

# NuMicro™ Series NM1510/520/530 Preliminary Technical Reference Manual

*The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.*

*Nuvoton is providing this document only for reference purposes of NuMicro™ microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.*

*All data and specifications are subject to change without notice.*

*For additional information or questions, please contact: Nuvoton Technology Corporation.*

[www.nuvoton.com](http://www.nuvoton.com)

TABLE OF CONTENTS

|      |   |     |
|------|---|-----|
| 1    | GENERAL DESCRIPTION .....                         | 12  |
| 2    | FEATURES .....                                    | 13  |
| 3    | BLOCK DIAGRAM .....                               | 17  |
| 4    | PARTS LIST .....                                  | 18  |
| 5    | PIN CONFIGURATION .....                           | 19  |
| 5.1  | LQFP 100-pin .....                                | 19  |
| 5.2  | LQFP 64-pin .....                                 | 20  |
| 5.3  | LQFP 48-pin .....                                 | 22  |
| 5.4  | Pin Description .....                             | 23  |
| 6    | ARM® CORTEX™-M0 CORE .....                        | 30  |
| 7    | SYSTEM MANAGEMENT .....                           | 32  |
| 7.1  | Overview .....                                    | 32  |
| 7.2  | System Reset .....                                | 33  |
| 7.3  | System Power Distribution .....                   | 34  |
| 7.4  | System Memory Map .....                           | 35  |
| 7.5  | System Manager Controller Register Map .....      | 37  |
| 7.6  | System Timer (SysTick) .....                      | 74  |
| 7.7  | Nested Vectored Interrupt Controller (NVIC) ..... | 78  |
| 7.8  | System Control Block Register .....               | 130 |
| 8    | CLOCK CONTROL .....                               | 138 |
| 8.1  | Overview .....                                    | 138 |
| 8.2  | Clock Generator .....                             | 139 |
| 8.3  | System Clock & SysTick Clock .....                | 140 |
| 8.4  | Peripherals Clock .....                           | 141 |
| 8.5  | Power-down mode (Deep Sleep Mode) Clock .....     | 142 |
| 8.6  | Frequency Divider Output .....                    | 143 |
| 8.7  | Register Map .....                                | 144 |
| 8.8  | Register Description .....                        | 145 |
| 9    | GENERAL PURPOSE I/O .....                         | 163 |
| 9.1  | Overview .....                                    | 163 |
| 9.2  | Features .....                                    | 163 |
| 9.3  | Functional Description .....                      | 164 |
| 9.4  | Enhanced PWM Port Output Driving Control .....    | 166 |
| 9.5  | Register Map .....                                | 167 |
| 9.6  | Register Description .....                        | 168 |
| 10   | TIMERS/COUNTERS .....                             | 182 |
| 10.1 | Overview .....                                    | 182 |
| 10.2 | Features .....                                    | 182 |
| 10.3 | Block Diagram .....                               | 183 |

|       |   |     |
|-------|---|-----|
| 10.4  | Function Description.....                           | 184 |
| 10.5  | Register Map .....                                  | 187 |
| 10.6  | Register Description .....                          | 189 |
| 11    | WATCHDOG TIMER (WDT) .....                          | 198 |
| 11.1  | Overview .....                                      | 198 |
| 11.2  | Features .....                                      | 198 |
| 11.3  | Block Diagram .....                                 | 199 |
| 11.4  | Register Map .....                                  | 201 |
| 11.5  | Register Description .....                          | 202 |
| 12    | WINDOW WATCHDOG TIMER (WWDT).....                   | 205 |
| 12.1  | Overview .....                                      | 205 |
| 12.2  | Features .....                                      | 205 |
| 12.3  | Block Diagram .....                                 | 206 |
| 12.4  | Functional Description.....                         | 207 |
| 12.5  | Register Map .....                                  | 209 |
| 12.6  | Register Description .....                          | 210 |
| 13    | BASIC PWM GENERATOR AND CAPTURE TIMER (BPWM).....   | 215 |
| 13.1  | Overview .....                                      | 215 |
| 13.2  | Features .....                                      | 216 |
| 13.3  | Block Diagram .....                                 | 217 |
| 13.4  | PWM-Timer Operation .....                           | 218 |
| 13.5  | Register Map .....                                  | 227 |
| 13.6  | Register Description .....                          | 228 |
| 14    | ENHANCED PWM GENERATOR FOR MOTOR DRIVE (EPWM) ..... | 243 |
| 14.1  | Overview .....                                      | 243 |
| 14.2  | Features .....                                      | 244 |
| 14.3  | PWM Operation.....                                  | 245 |
| 14.4  | PWM Brake .....                                     | 252 |
| 14.5  | PWM Port Output Driving Control .....               | 255 |
| 14.6  | PWM Modes.....                                      | 256 |
| 14.7  | Polarity Control.....                               | 258 |
| 14.8  | PWM Mask Output Function .....                      | 259 |
| 14.9  | Interrupt Architecture of Enhanced PWM.....         | 262 |
| 14.10 | Register Map .....                                  | 263 |
| 14.11 | Register Description .....                          | 264 |
| 15    | MOTOR DRIVE UNIT (MDU).....                         | 280 |
| 15.1  | Overview .....                                      | 280 |
| 15.2  | Features .....                                      | 280 |
| 15.3  | MDU Architecture .....                              | 281 |

|       |  |     |
|-------|--|-----|
| 15.4  | Operation of Motor Drive Unit .....                      | 282 |
| 15.5  | Register Map .....                                       | 283 |
| 15.6  | Register Description .....                               | 286 |
| 16    | HARDWARE DIVIDER.....                                    | 315 |
| 16.1  | Overview .....   | 315 |
| 16.2  | Features .....   | 315 |
| 16.3  | Register Map .....                                       | 316 |
| 16.4  | Register Description .....                               | 317 |
| 17    | ENHANCED INPUT CAPTURE TIMER.....                        | 323 |
| 17.1  | Overview .....   | 323 |
| 17.2  | Features .....   | 323 |
| 17.3  | Input Capture Timer/Counter Architecture .....           | 324 |
| 17.4  | Input Noise Filter .....                                 | 325 |
| 17.5  | Operation of Input Capture Timer/Counter .....           | 326 |
| 17.6  | Input Capture Timer/Counter Interrupt Architecture.....  | 329 |
| 17.7  | Register Map .....                                       | 330 |
| 17.8  | Register Description .....                               | 331 |
| 18    | QUADRATURE ENCODER INTERFACE (QEI).....                  | 341 |
| 18.1  | Overview .....   | 341 |
| 18.2  | Features .....   | 341 |
| 18.3  | QEI Architecture .....                                   | 342 |
| 18.4  | Input Noise Filter .....                                 | 343 |
| 18.5  | Operation of Quadrature Encoder Interface.....           | 344 |
| 18.6  | Compare Function.....                                    | 348 |
| 18.7  | Reload Counter by Pin IDX .....                          | 349 |
| 18.8  | Capture QEP Counter .....                                | 350 |
| 18.9  | QEI Interrupt Architecture .....                         | 352 |
| 18.10 | Register Map .....                                       | 353 |
| 18.11 | Register Description .....                               | 354 |
| 19    | UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART) ..... | 364 |
| 19.1  | Overview .....   | 364 |
| 19.2  | Features .....   | 366 |
| 19.3  | Block Diagram .....                                      | 367 |
| 19.4  | IrDA Mode .....  | 370 |
| 19.5  | LIN (Local Interconnection Network) mode.....            | 372 |
| 19.6  | RS-485 function mode.....                                | 381 |
| 19.7  | Register Map .....                                       | 383 |
| 19.8  | Register Description .....                               | 384 |

|      |  |     |
|------|--|-----|
| 20   | SERIAL PERIPHERAL INTERFACE (SPI) .....  | 411 |
| 20.1 | Overview .....                           | 411 |
| 20.2 | Features .....                           | 411 |
| 20.3 | Block Diagram .....                      | 412 |
| 20.4 | Functional Description .....             | 413 |
| 20.5 | Timing Diagram .....                     | 421 |
| 20.6 | Programming Examples .....               | 423 |
| 20.7 | Register Map .....                       | 425 |
| 20.8 | Register Description .....               | 426 |
| 21   | I <sup>2</sup> C SERIAL INTERFACE .....  | 440 |
| 21.1 | Overview .....                           | 440 |
| 21.2 | Features .....                           | 441 |
| 21.3 | Functional Description .....             | 442 |
| 21.4 | Protocol Registers .....                 | 445 |
| 21.5 | Register Map .....                       | 449 |
| 21.6 | Register Description .....               | 450 |
| 21.7 | Operation Modes .....                    | 460 |
| 22   | CONTROLLER AREA NETWORK (CAN) .....      | 466 |
| 22.1 | Overview .....                           | 466 |
| 22.2 | Features .....                           | 466 |
| 22.3 | Block Diagram .....                      | 467 |
| 22.4 | Functional Description .....             | 468 |
| 22.5 | Test Mode .....                          | 470 |
| 22.6 | Register Description .....               | 492 |
| 22.7 | Register Map .....                       | 493 |
| 22.8 | CAN Interface Reset State .....          | 495 |
| 23   | 12-BIT ANALOG-TO-DIGITAL CONVERTER ..... | 534 |
| 23.1 | Overview .....                           | 534 |
| 23.2 | Features .....                           | 535 |
| 23.3 | Block Diagram .....                      | 536 |
| 23.4 | Operation Procedure .....                | 538 |
| 23.5 | Register Map .....                       | 549 |
| 23.6 | Register Description .....               | 552 |
| 24   | ANALOG COMPARATOR .....                  | 583 |
| 24.1 | Overview .....                           | 583 |
| 24.2 | Features .....                           | 584 |
| 24.3 | Block Diagram .....                      | 585 |
| 24.4 | Interrupt Sources .....                  | 586 |
| 24.5 | Hysteresis Function .....                | 587 |

|      |   |     |
|------|---|-----|
| 24.6 | Register Map .....                            | 588 |
| 24.7 | Register Description .....                    | 589 |
| 25   | OP AMPLIFIER .....                            | 595 |
| 25.1 | Overview .....                                | 595 |
| 25.2 | Features .....                                | 595 |
| 25.3 | Block Diagram .....                           | 596 |
| 25.4 | Interrupt Sources.....                        | 597 |
| 25.5 | Register Map .....                            | 598 |
| 25.6 | Register Description .....                    | 599 |
| 26   | FLASH MEMORY CONTROL (FMC) .....              | 602 |
| 26.1 | Overview .....                                | 602 |
| 26.2 | Features .....                                | 603 |
| 26.3 | Block Diagram .....                           | 604 |
| 26.4 | Functional Description.....                   | 605 |
| 26.5 | In-Application-Programming (IAP) .....        | 613 |
| 26.6 | In-System-Programming (ISP) .....             | 614 |
| 26.7 | Register Map .....                            | 617 |
| 26.8 | Register Description .....                    | 618 |
| 27   | ELECTRICAL CHARACTERISTICS.....               | 627 |
| 27.1 | Absolute Maximum Ratings .....                | 627 |
| 27.2 | DC Electrical Characteristics.....            | 628 |
| 27.3 | AC Electrical Characteristics.....            | 631 |
| 27.4 | Analog Characteristics .....                  | 633 |
| 28   | PACKAGE DIMENSIONS .....                      | 637 |
| 28.1 | 100L LQFP (14x14x1.4 mm footprint 2.0mm)..... | 637 |
| 28.2 | 64L LQFP (10x10x1.4mm footprint 2.0 mm).....  | 638 |
| 28.3 | 48L LQFP (7x7x1.4mm footprint 2.0mm).....     | 639 |
| 29   | REVISION HISTORY .....                        | 640 |

List of Figures

|  |     |
|--|-----|
| Figure 5–1 NuMicro™ NM15xx Series LQFP-100 Pin Diagram .....               | 19  |
| Figure 5–2 NuMicro™ NM15xx Series LQFP-64 Pin Diagram .....                | 20  |
| Figure 5–3 NuMicro™ NM15xx Series LQFP-48 Pin Diagram .....                | 22  |
| Figure 6–1 Functional Controller Diagram .....                             | 30  |
| Figure 8–1 Clock Generator Block Diagram .....                             | 139 |
| Figure 8–2 System Clock Block Diagram .....                                | 140 |
| Figure 8–3 SysTick Clock Control Block Diagram .....                       | 140 |
| Figure 8–4 Clock Source of Frequency Divider .....                         | 143 |
| Figure 8–5 Block Diagram of Frequency Divider .....                        | 143 |
| Figure 9–1 Push-Pull Output.....   | 164 |
| Figure 9–2 Open-Drain Output.....  | 164 |
| Figure 9–3 Quasi bi-directional I/O Mode .....                             | 165 |
| Figure 9–4 PWM Output Driving Control .....                                | 166 |
| Figure 10-1 Timer Controller Block Diagram .....                           | 183 |
| Figure 10-2 Clock Source of Timer Controller .....                         | 183 |
| Figure 10-3 Continuous Counting Mode .....                                 | 185 |
| Figure 11–1 Watchdog Timer Clock Control.....                              | 199 |
| Figure 11–2 Watchdog Timer Block Diagram .....                             | 199 |
| Figure 11-3 Watchdog Timer Time-out Interval and Reset Period Timing ..... | 200 |
| Figure 12-1 Window Watchdog Timer Clock Control.....                       | 206 |
| Figure 12-2 Window Watchdog Timer Block Diagram .....                      | 206 |
| Figure 12-3 Window Watchdog Timer reset and reload behavior .....          | 208 |
| Figure 13-1 PWM Clock Source Control .....                                 | 217 |
| Figure 13-2 PWM Architecture Diagram .....                                 | 217 |
| Figure 13-3 Legend of Internal Comparator Output of PWM-Timer .....        | 218 |
| Figure 13-4 PWM-Timer Operation Timing.....                                | 219 |
| Figure 13-5 PWM Edge-aligned Interrupt Generate Timing Waveform.....       | 219 |
| Figure 13-6 Center-aligned Type Output Waveform.....                       | 220 |
| Figure 13-7 PWM Center-aligned Interrupt Generate Timing Waveform .....    | 221 |
| Figure 13-8 PWM Double Buffering Illustration.....                         | 222 |
| Figure 13-9 PWM Controller Output Duty Ratio.....                          | 223 |
| Figure 13-10 Paired-PWM Output with Dead-zone Generation Operation .....   | 223 |
| Figure 13-11 Capture Operation Timing .....                                | 224 |
| Figure 13-12 PWM Interrupt Architecture Diagram.....                       | 225 |
| Figure 14–1 PWM Block Diagram.....   | 244 |
| Figure 14–2 PWM Clock Source Control .....                                 | 245 |
| Figure 14–3 PWM Time-base Generator .....                                  | 245 |

|   |     |
|---|-----|
| Figure 14–4 Edge-aligned PWM .....  | 247 |
| Figure 14–5 PWM0 Edge aligned Waveform Output.....  | 247 |
| Figure 14–6 Edge-aligned Flow Diagram .....   | 248 |
| Figure 14–7 Center-aligned Mode .....   | 249 |
| Figure 14–8 Example PWM0 Center-aligned Waveform Output .....                             | 250 |
| Figure 14–9 Center-aligned Flow Diagram (INT_TYPE = 0) .....                              | 251 |
| Figure 14–10 PWM Brake Function .....   | 252 |
| Figure 14–11 PWM Brake Condition (Edge-aligned Mode).....                                 | 253 |
| Figure 14–12 PWM Brake Condition (Centre-aligned Mode) .....                              | 254 |
| Figure 14–13 PWM Output Driving Control .....   | 255 |
| Figure 14–14 Dead-Time Insertion .....  | 256 |
| Figure 14–15 Initial State and Polarity Control with Rising Edge Dead Time Insertion..... | 258 |
| Figure 14–16 Illustration of Mask Control .....   | 259 |
| Figure 14–17 Architecture of Enhanced PWM Interrupts .....                                | 262 |
| Figure 15–1 MDU Clock Source Control.....   | 281 |
| Figure 15–2 Motor Drive Unit Architecture.....  | 281 |
| Figure 15–3 PI Controller Architecture.....   | 282 |
| Figure 17–1 Input Capture Timer/Counter Clock Source Control.....                         | 324 |
| Figure 17–2 Input Capture Timer/Counter Architecture.....                                 | 324 |
| Figure 17–3 Noise Filter Sampling Clock Selection.....                                    | 325 |
| Figure 17–4 Input Capture Timer/Counter Functions Block .....                             | 327 |
| Figure 17–5 Input Capture Timer/Counter Interrupt Architecture Diagram .....              | 329 |
| Figure 18–1 QEI Clock Source Control.....   | 342 |
| Figure 18–2 QEI Block Diagram .....   | 342 |
| Figure 18–3 Noise Filter.....   | 343 |
| Figure 18–4 Noise Filter Sampling Clock Selection.....                                    | 343 |
| Figure 18–5 QEI/QEB/IDX Timing Requirement through Noise Filter.....                      | 343 |
| Figure 18–6 X4 Counting Mode .....  | 345 |
| Figure 18–7 X2 Counting Mode .....  | 346 |
| Figure 18–8 Compare Operation .....   | 348 |
| Figure 18–9 QEI_CNT Reload/Reset Control.....   | 349 |
| Figure 18–10 Trigger Control of Capturing QEP Counter.....                                | 350 |
| Figure 18–11 Capture and Latch QEP Counter.....   | 351 |
| Figure 18–12 Quadrature Encoder Interface Interrupt Architecture Diagram.....             | 352 |
| Figure 19-1 UART Clock Control Diagram.....   | 367 |
| Figure 19-2 UART Block Diagram.....   | 367 |
| Figure 19-3 Auto Flow Control Block Diagram.....  | 369 |
| Figure 19-4 IrDA Block Diagram .....  | 370 |



|  |     |
|--|-----|
| Figure 19-5 IrDA Timing Diagram .....  | 371 |
| Figure 19-6 Structure of LIN Frame .....   | 372 |
| Figure 19-7 Structure of LIN Byte .....  | 372 |
| Figure 19-8 Break detection in LIN mode .....  | 375 |
| Figure 19-9 LIN sync field measurement .....   | 378 |
| Figure 19-10 UA_BAUD update sequence in automatic resynchronization mode (LINS_DUM_EN = 1) ..... | 379 |
| Figure 19-11 UA_BAUD update sequence in automatic resynchronization mode (LINS_DUM_EN = 0) ..... | 379 |
| Figure 19-12 Structure of RS-485 Frame .....   | 382 |
| Figure 20-1 SPI Block Diagram.....   | 412 |
| Figure 20-2 SPI Master Mode Application Block Diagram.....                                       | 413 |
| Figure 20-3 SPI Slave Mode Application Block Diagram.....  | 413 |
| Figure 20-4 32-Bit in One Transaction .....  | 415 |
| Figure 20-5 Byte Reorder Function.....   | 416 |
| Figure 20-6 Timing Waveform for Byte Suspend.....  | 417 |
| Figure 20-7 FIFO Mode Block Diagram .....  | 418 |
| Figure 20-8 SPI Timing in Master Mode .....  | 421 |
| Figure 20-9 SPI Timing in Master Mode (Alternate Phase of SPI_CLK) .....                         | 421 |
| Figure 20-10 SPI Timing in Slave Mode .....  | 422 |
| Figure 20-11 SPI Timing in Slave Mode (Alternate Phase of SPI_CLK) .....                         | 422 |
| Figure 21-1 I <sup>2</sup> C Bus Timing .....  | 440 |
| Figure 21-2 I <sup>2</sup> C Protocol.....   | 442 |
| Figure 21-3 Master Transmits Data to Slave .....   | 442 |
| Figure 21-4 Master Reads Data from Slave .....   | 442 |
| Figure 21-5 START and STOP Condition .....   | 443 |
| Figure 21-6 Bit Transfer on the I <sup>2</sup> C Bus .....                                       | 444 |
| Figure 21-7 Acknowledge on the I <sup>2</sup> C Bus .....  | 444 |
| Figure 21-8 I <sup>2</sup> C Data Shifting Direction .....                                       | 446 |
| Figure 21-9 I <sup>2</sup> C Time out Count Block Diagram.....                                   | 448 |
| Figure 21-10 Legend for the Following Five Figures .....   | 460 |
| Figure 21-11 Master Transmitter Mode .....   | 461 |
| Figure 21-12 Master Receiver Mode .....  | 462 |
| Figure 21-13 Slave Receiver Mode .....   | 463 |
| Figure 21-14 Slave Transmitter Mode .....  | 464 |
| Figure 21-15 GC Mode .....   | 465 |
| Figure 22-1 CAN Peripheral Block Diagram .....   | 467 |
| Figure 22-2 CAN Core in Silent Mode .....  | 470 |

|   |     |
|---|-----|
| Figure 22-3 CAN Core in Loop Back Mode .....                                      | 470 |
| Figure 22-4 CAN Core in Loop Back Mode Combined with Silent Mode .....            | 471 |
| Figure 22-5 Data Transfer between IF <sup>n</sup> Registers and Message .....     | 474 |
| Figure 22-6 Application Software Handling of a FIFO Buffer .....                  | 479 |
| Figure 22-7 Bit Timing .....  | 481 |
| Figure 22-8 Propagation Time Segment .....  | 483 |
| Figure 22-9 Synchronization on “late” and “early” Edges .....                     | 485 |
| Figure 22-10 Filtering of Short Dominant Spikes .....                             | 486 |
| Figure 22-11 Structure of the CAN Core’s CAN Protocol Controller .....            | 488 |
| Figure 23-1 ADCA Converter Block Diagram .....                                    | 536 |
| Figure 23-2 ADCB Converter Block Diagram .....                                    | 537 |
| Figure 23-3 ADC Clock Control .....   | 537 |
| Figure 23-4 Single Sampling Mode Conversion Timing Diagram .....                  | 539 |
| Figure 23-5 SAMPLEA0~3 and SAMPLEB0~3 Control Block Diagram .....                 | 540 |
| Figure 23-6 SAMPLEA4~7 and SAMPLEB4~7 Control Block Diagram .....                 | 540 |
| Figure 23-7 SAMPLE Conversion Priority Arbitrator Diagram .....                   | 541 |
| Figure 23-8 PWM-triggered ADC Start .....   | 542 |
| Figure 23-9 Specific SAMPLE A/D EOC Signal for ADINT0 Interrupt .....             | 544 |
| Figure 23-10 Specific SAMPLE A/D EOC Signal for ADINT1 Interrupt .....            | 544 |
| Figure 23-11 Specific SAMPLE A/D EOC Signal for ADINT2 Interrupt .....            | 545 |
| Figure 23-12 Specific SAMPLE A/D EOC Signal for ADINT3 Interrupt .....            | 545 |
| Figure 23-13 Conversion Start Delay Timing Diagram .....                          | 546 |
| Figure 23-14 A/D Extend Sampling Timing Diagram .....                             | 547 |
| Figure 23-15 A/D Conversion Result Monitor Logics Diagram .....                   | 547 |
| Figure 23-16 A/D Controller Interrupts .....                                      | 548 |
| Figure 24-1 Analog Comparator Block Diagram .....                                 | 585 |
| Figure 24-2 Analog Comparator Controller Interrupt .....                          | 586 |
| Figure 24-3 Comparator Hysteresis function .....                                  | 587 |
| Figure 25-1 OP Amplifier Block Diagram .....                                      | 596 |
| Figure 25-2 OP Amplifier Interrupt Flags for Analog Comparator Interrupt .....    | 597 |
| Figure 26-1 Flash Memory Control Block Diagram .....                              | 604 |
| Figure 26-2 Flash Memory Organization .....                                       | 606 |
| Figure 26-3 Program Executing Range for boot from APROM and boot from LDROM ..... | 611 |
| Figure 26-4 Executable Range of Code with IAP Function Enabled .....              | 613 |
| Figure 26-5 Example Flow of Boot Selection by BS Bit .....                        | 614 |
| Figure 26-6 ISP Flow Example .....  | 615 |
| Figure 27-1 Typical Crystal Application Circuit .....                             | 631 |

List of Tables

|  |     |
|--|-----|
| Table 7-1 Address Space Assignments for On-Chip Controllers.....         | 36  |
| Table 7-2 System Exception Model .....                                   | 79  |
| Table 7-3 System Exception Map .....                                     | 80  |
| Table 7-4 Vector Table.....  | 80  |
| Table 8-1 Power-down mode Control Table .....                            | 146 |
| Table 11-1 Watchdog Timer Time out Interval Selection.....               | 200 |
| Table 12-1 Window Watchdog Timer Pre-scale Value Selection .....         | 207 |
| Table 12-2 WINCMP Setting Limitation .....                               | 208 |
| Table 18-1 Direction of Count .....                                      | 347 |
| Table 19-1 UART Baud Rate Equation .....                                 | 364 |
| Table 19-2 UART Baud Rate Setting Table.....                             | 365 |
| Table 19-3 LIN Header selection in master mode .....                     | 373 |
| Table 19-4 UART Interrupt Sources and Flags Table in Software Mode ..... | 399 |
| Table 19-5 Baud Rate Equation Table.....                                 | 401 |
| Table 21-1 I <sup>2</sup> C Status Code Description Table .....          | 447 |
| Table 22-1 Initialization of a Transmit Object .....                     | 476 |
| Table 22-2 Initialization of a Receive Object .....                      | 477 |
| Table 22-3 CAN Bit Time Parameters .....                                 | 482 |
| Table 22-4 CAN Register Map for Each Bit Function .....                  | 498 |
| Table 22-5 Error Code .....  | 502 |
| Table 22-6 Source of Interrupts .....                                    | 505 |
| Table 22-7 IF1 and IF2 Message Interface Register .....                  | 509 |
| Table 22-8 Structure of a Message Object in Message Memory.....          | 523 |
| Table 26-1 Memory Address Map.....                                       | 606 |
| Table 26-2 ISP Command List.....   | 616 |

### 1 GENERAL DESCRIPTION

The NuMicro™ NM15xx Series 32-bit microcontroller is embedded with the newest ARM® Cortex™-M0 core at a cost equivalent to traditional 8-bit microcontroller for industrial control and applications which need high performance.

The NuMicro™ NM15xx Series embedded with the Cortex™-M0 core runs up to 72 MHz and supports a variety of industrial control and applications which need high CPU performance. The NuMicro™ NM15xx Series provides 128K/64K/32K bytes embedded flash, 4 Kbytes data flash, 4 Kbytes flash for the ISP, and 16K/8K/4K/2K bytes embedded SRAM. This MCU includes advanced PWM function, MDU (motor drive unit) and input capture timer which are specially designed for motor driving application. It is also equipped with plenty of peripheral devices, such as Timers, Watchdog Timer, UART, SPI, I2C, PWM Timer, GPIO, 12-bit ADC, Low Voltage Detector and Brown-out detector. These useful functions make the NuMicro™ NM15xx Series powerful for a wide range of applications.

In addition, the NuMicro™ NM15xx Series is equipped with ISP (In-System Programming) and ICP (In-Circuit Programming) functions, which allow user to update the program memory without removing the chip from the actual end product.

## 2 FEATURES

- Core
  - ARM® Cortex™ -M0 core runs up to 72 MHz
  - One 24-bit system timer
  - Supports low power sleep-mode
  - Single-cycle 32-bit hardware multiplier
  - NVIC for the 32 interrupt inputs, each with 4-levels of priority
  - Supports Serial Wire Debug (SWD) support with two watchpoints and four breakpoints
- Memory
  - 128K/64K/32K bytes Flash for program memory (APROM)
  - 8 Kbytes Flash memory for ISP loader (LDROM)
  - 4 Kbytes Flash memory for Data Flash
  - Supports In-system program (ISP) and In-application program (IAP) application code update
  - 16K/8K/4K bytes SRAM for internal scratch-pad RAM
  - Supports 2 wire ICP update from ICE interface
  - Supports fast parallel programming mode by external programmer
- Clock Control
  - Programmable system clock source
  - Built-in internal 22.1184 MHz OSC (trimmed to 1%) for system operation
  - Built-in low power 10 kHz OSC for watchdog timer and wake-up in sleep mode
  - External 4~24 MHz crystal input
  - Supports one PLL, up to 72 MHz, for high performance system operation
- Hardware divider
  - Supports signed 32-bit dividend, 16-bit divisor operation
- GPIO port
  - Up to 8-bit, 11-port I/O
  - Bit control available
  - Four I/O modes:
    - ◆ Quasi bi-directional
    - ◆ Push-pull (output with high driver and sink current)
    - ◆ Open-drain
    - ◆ Input only with high impedance (default)
  - TTL/Schmitt trigger input selectable
  - I/O pins configurable as interrupt source with edge/level setting
  - INT0 and INT1 pins with individual interrupt vectors
- Timers
  - Supports four channel 32-bit timers; one 8-bit pre-scale counter with 24-bit up-timer for each timer

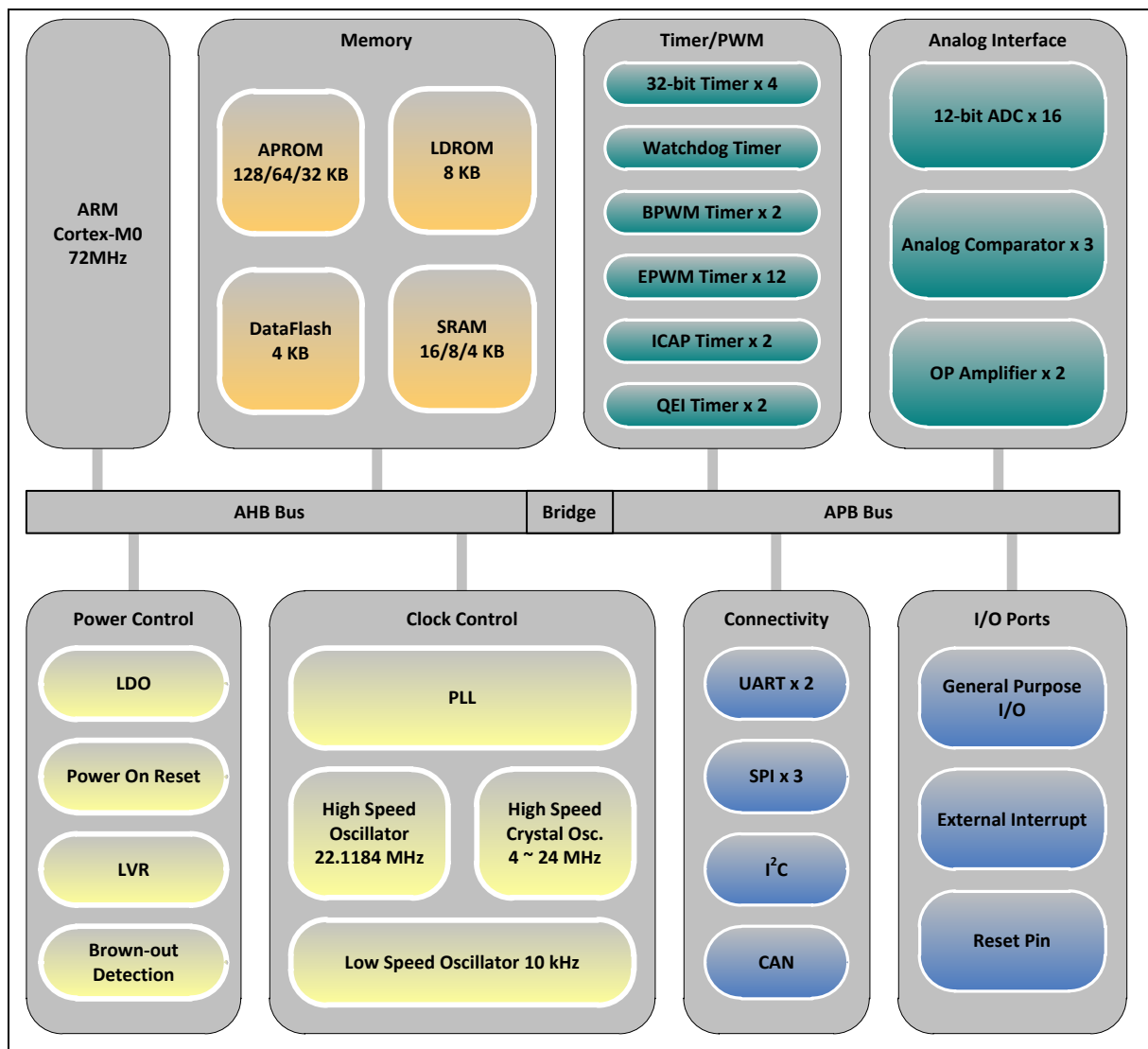
- 24-bit timer value is readable through TDR (Timer Data Register)
- Supports One-shot, Periodic and Toggle operation modes
- Supports event counter function
- Watchdog Timer
  - ON/OFF by hardware configuration or software
  - Multiple clock sources
  - Supports wake-up from Power-down or Sleep mode
  - Interrupt or reset selectable on watchdog time-out
  - Time-out reset delay period time can be selected
- WWDT (Window Watchdog Timer)
  - 6-bit down counter with 11-bit pre-scale for wide range window selected
- Basic PWM
  - 1 unit of 16-bit basic PWM, up to 2ch output
  - Alternative function as input capture timer
- Enhanced PWM
  - 2 units of 16-bit enhanced PWM, up to 6ch output with dead-zone control, brake and polarity control for motor drive
  - Default tri-state during any reset
- Enhanced Input Capture
  - Up to 2 units of 24-bit input capture
  - Each unit has 3 inputs: IC0, IC1 and IC2
- QEI (Quadrature Encoder Interface)
  - Up to 2 units of Quadrature Encoder Interface
  - Each unit has 3 inputs: QEIA, QEIB and IDX
- MDU (Motor Drive Unit)
  - Built-in PI + FOC + SVPWM
  - Output Space Vector PWM timing to PWM unit 0/1
- UART
  - Up to two 16550 compatible UART devices
  - Programmable baud-rate generator
  - Buffered receiving and transmitting, each with 16 bytes FIFO
  - Supports flow control (TX, RX, CTS and RTS)
  - Supports IrDA(SIR) function
  - Supports RS-485
- SPI
  - Up to three sets of SPI device
  - Supports SPI master/slave mode
  - Full duplex synchronous serial data transfer
  - Variable length of transfer data from 8 to 32 bits
  - MSB or LSB first data transfer

- Rx and Tx on both rising or falling edge of serial clock independently
- Supports Byte Suspend mode in 32-bit transmission
- I<sup>2</sup>C
  - Master/Slave up to 1 Mbit/s
  - Bi-directional data transfer between masters and slaves
  - Multi-master bus (no central master)
  - Arbitration between simultaneously transmitting masters
  - Programmable clocks allow versatile rate control
  - Multiple address recognition (four slave address with mask option)
- CAN
  - CAN 2.0B protocol compatible device
  - Support 11-bit identifier as well as 29-bit identifier
  - Bit rates up to 1Mbits/s
  - NRZ bit Coding/ Encoding
  - Error Detection & Status Report
  - Bit error, Form error, Stuffing error, 15-bit CRC detection, and Acknowledge error Interrupt
  - Bit Timing Synchronization
  - Acceptance filter extension
- ADC
  - Two A/D converters
  - Each ADC with up to 8 channel, 12-bit resolution with 10-bit accuracy
  - 16 result registers
  - Sampling rate up to 800ksps
  - Two operating modes:
    - Single Sampling mode: Only one specified channel can be sampled at one time.
    - Simultaneous Sampling mode: Allowing two ADC channels to be sampled simultaneously.
  - Two converting result digital comparators
  - Conversion start by software, external pins, or linked with Timer 0~3 or PWM module
- Up to three Analog Comparators
- Up to two OPA (operational amplifier)
- Brown-out detector
  - 4 levels: 4.4V/3.7V/2.7V/2.2V
  - Optional brown-out interrupt or reset

- Built-in LDO for Wide Operating Voltage Range: 2.5V to 5.5V
- Low Voltage Reset
- 96-bit unique ID
- Operating Temperature: -40°C ~105°C
- Develop tools: parallel writer or In-Circuit Programming (ICP) writer
- Packages:
  - All Green package (RoHS)
  - LQFP 100/64/48-pin



### 3 BLOCK DIAGRAM



### 4 PARTS LIST

|                   | NM1530VE3AE<br>NM1530VD3AE                       | NM1520RD2AE<br>NM1520RC2AE                     | NM1520LD2AE<br>NM1520LC2AE                | NM1521RD2AE                            | NM1510LC1AE                               |
|-------------------|--|--|---|--|---|
| <b>AP Flash</b>   | 128KB/64KB                                       | 64KB/32KB                                      | 64KB/32K                                  | 64KB                                   | 32KB                                      |
| <b>RAM</b>        | 16KB   | 8KB  | 8KB                                       | 8KB                                    | 4KB                                       |
| <b>Data Flash</b> | 4KB  | 4KB  | 4KB                                       | 4KB                                    | 4KB                                       |
| <b>Timer</b>      | 4 x 24-bit                                       | 4 x 24-bit                                     | 4 x 24-bit                                | 4 x 24-bit                             | 4 x 24-bit                                |
| <b>PWM</b>        | 6ch PWM0,<br>6ch PWM1<br>2ch PWM2                | 6ch PWM0,<br>6ch PWM1<br>1ch PWM2              | 6ch PWM0,<br>3ch PWM1                     | 6ch PWM0,<br>6ch PWM1<br>1ch PWM2      | 6ch PWM0,<br>3ch PWM1                     |
| <b>QEI IC</b>     | 3ch x 2 sets(QEI0,QEI1)<br>3ch x 2 sets(IC0,IC1) | 3ch x 1 set(QEI0,QEI1)<br>Pin shared with QEI0 | 3ch x 1 set(QEI0)<br>Pin shared with QEI0 | 3ch x 1 set(QEI0)<br>3ch x 1 set (IC1) | 3ch x 1 set(QEI0)<br>Pin shared with QEI0 |
| <b>UART</b>       | 2  | 2  | 2   | 2                                      | 2   |
| <b>SPI</b>        | 3  | 1  | 1   | 1                                      | 1   |
| <b>I2C</b>        | 1  | 1  | 1   | 1                                      | 1   |
| <b>CAN 2.0B</b>   | 1  | 1  | 1   | 1                                      | -   |
| <b>ADC</b>        | 8ch ADCA<br>8ch ADCB                             | 7ch ADCA<br>7ch ADCB                           | 5ch ADCA<br>4ch ADCB                      | 4ch ADCA<br>7ch ADCB                   | 5ch ADCA<br>4ch ADCB                      |
| <b>Comparator</b> | CMP0, CMP1, CMP2                                 | CMP1, CMP2                                     | CMP1                                      | CMP2                                   | CMP1                                      |
| <b>OP</b>         | OP0, OP1   | OP0, OP1                                       | OP0, OP1                                  | OP0, OP1                               | OP0, OP1                                  |
| <b>Package</b>    | LQFP100(14x14x1.4mm)                             | LQFP64(10x10x1.4mm)                            | LQFP48(7x7x1.4mm)                         | LQFP64(10x10x1.4mm)                    | LQFP48(7x7x1.4mm)                         |

## 5 PIN CONFIGURATION

### 5.1 LQFP 100-pin

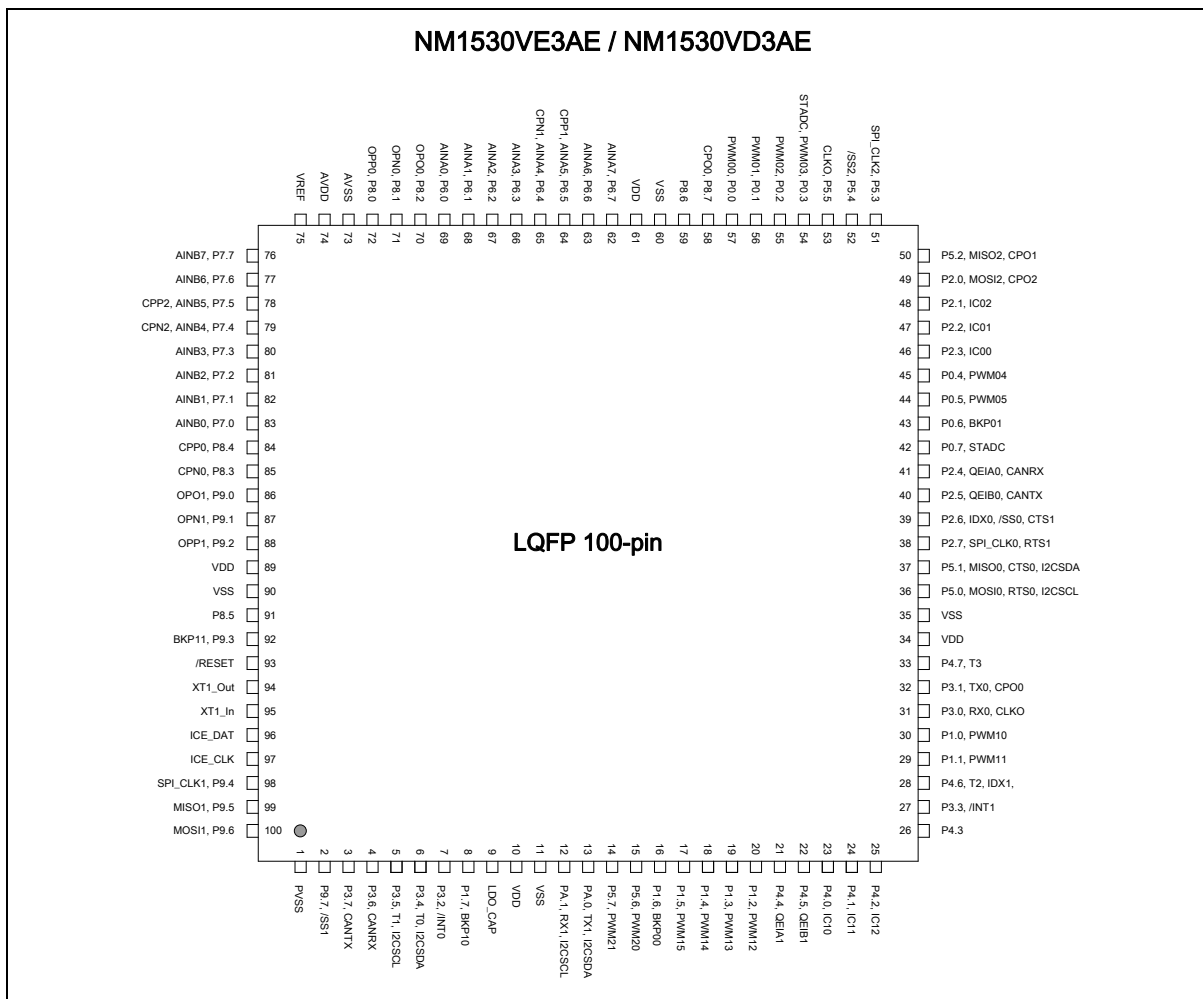


Figure 5–1 NuMicro™ NM15xx Series LQFP-100 Pin Diagram

## 5.2 LQFP 64-pin

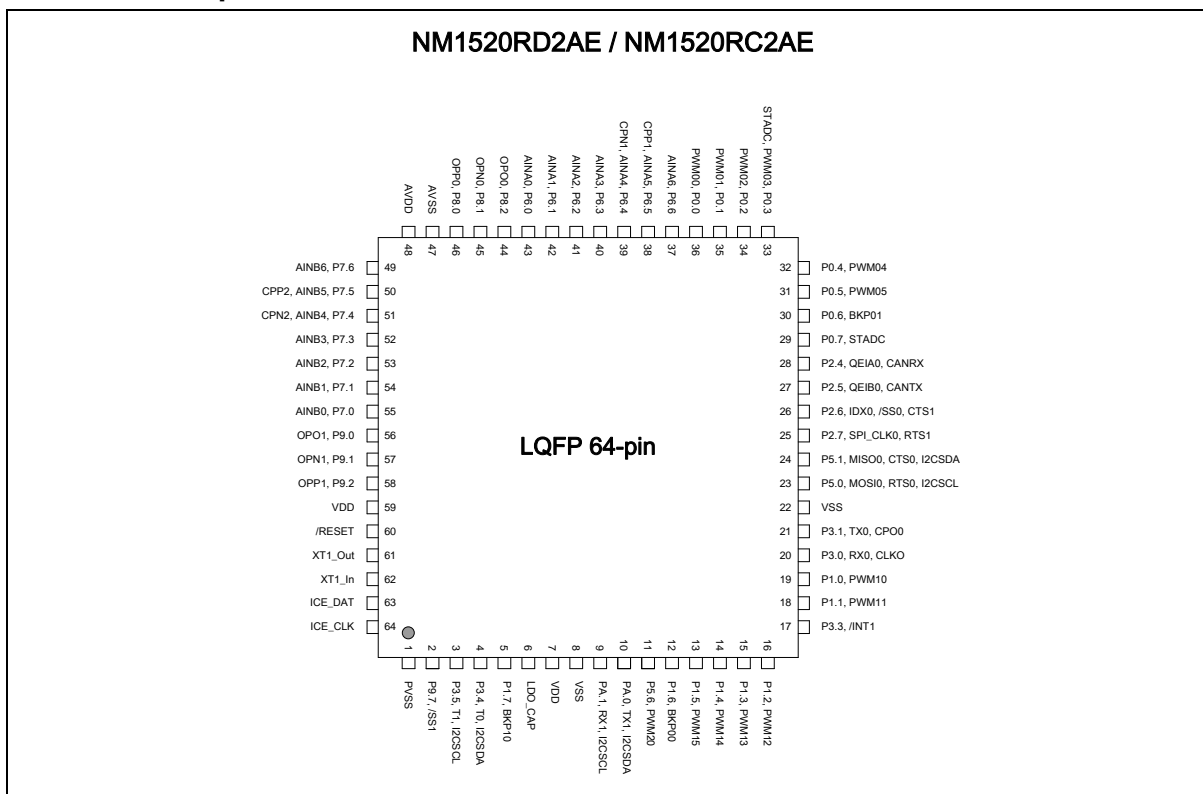
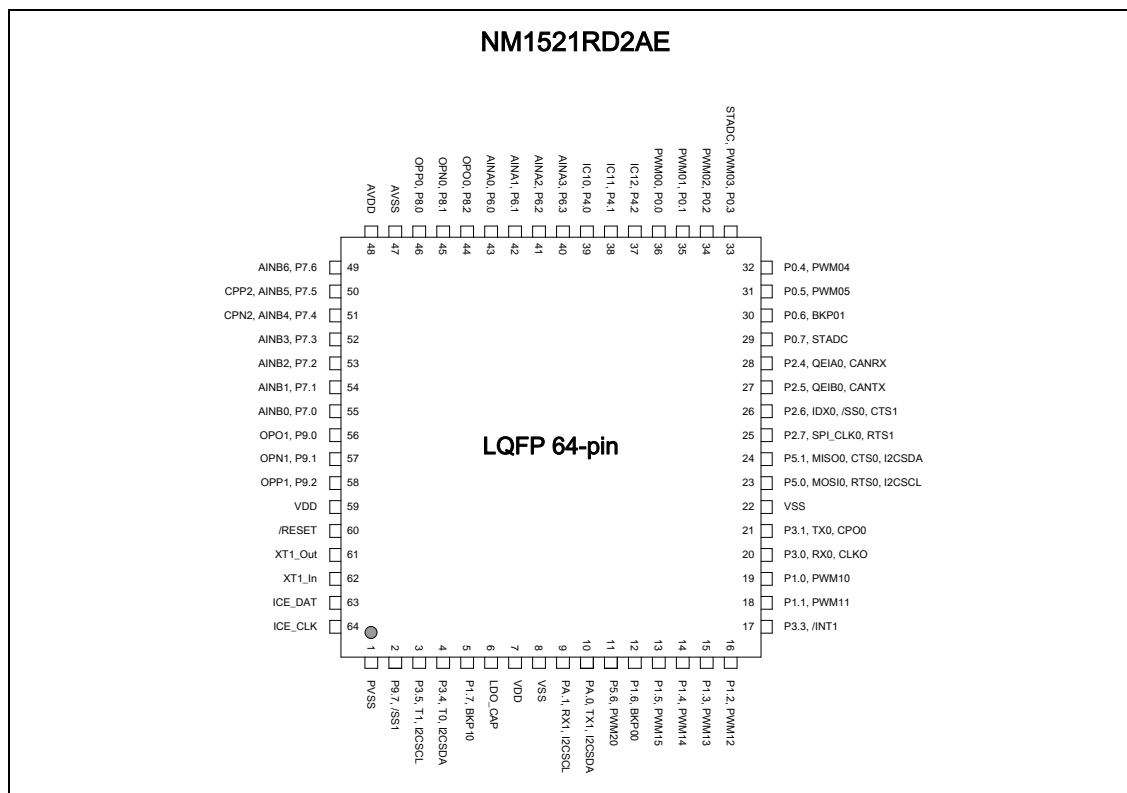


Figure 5–2 NuMicro™ NM15xx Series LQFP-64 Pin Diagram



### 5.3 LQFP 48-pin

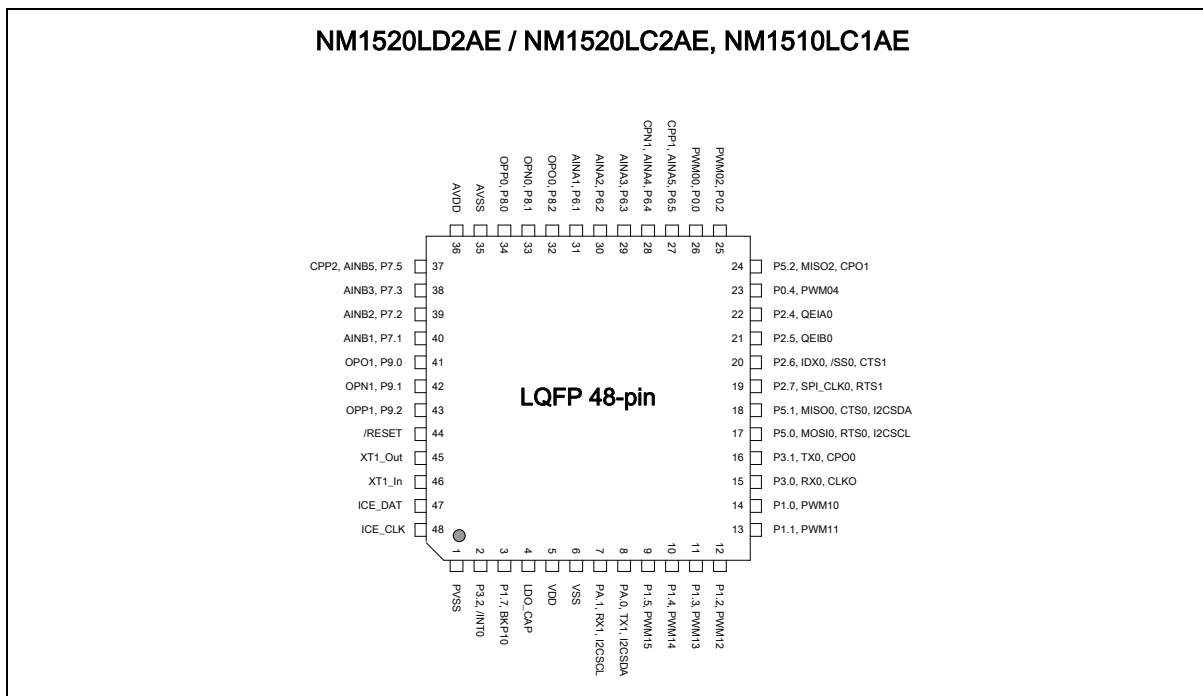


Figure 5–4 NuMicro™ NM15xx Series LQFP-48 Pin Diagram

### 5.4 Pin Description

| Pin Number |        |        | Pin Name         | Pin Type <sup>[1]</sup> | Description  |
|------------|--------|--------|------------------|-------------------------|--|
| 100-pin    | 64-pin | 48-pin |                  |                         |  |
| 10         | 7      | 5      | V <sub>DD</sub>  | P                       | <b>POWER SUPPLY:</b> Supply voltage <b>Digital</b> V <sub>DD</sub> for operation.  |
| 34         |        |        |                  |                         |  |
| 61         | 59     |        |                  |                         |  |
| 89         |        |        |                  |                         |  |
| 11         | 8      | 6      | V <sub>SS</sub>  | P                       | <b>GROUND:</b> Digital Ground potential.   |
| 35         |        |        |                  |                         |  |
| 60         | 22     |        |                  |                         |  |
| 90         |        |        |                  |                         |  |
| 9          | 6      | 4      | LDO_CAP          | P                       | <b>LDO:</b> LDO output pin<br><b>Note: It needs to be connected with a 10uF capacitor.</b>   |
| 1          | 1      | 1      | PVSS             | P                       | <b>PLL GROUND:</b> PLL Ground potential.   |
| 74         | 48     | 36     | AV <sub>DD</sub> | AP                      | Power supply for internal analog circuit   |
| 73         | 47     | 35     | AV <sub>SS</sub> | AP                      | Ground Pin for analog circuit  |
| 75         | -      | -      | Vref             | AP                      | Voltage reference input for ADC  |
| 93         | 60     | 44     | /RESET           | I<br>(ST)               | <b>RESET:</b> /RST pin is a Schmitt trigger input pin for hardware device reset. A “ <b>Low</b> ” on this pin for 768 clock counter of Internal RC 22.1184 MHz while the system clock is running will reset the device. /RST pin has an internal pull-up resistor allowing power-on reset by simply connecting an external capacitor to GND. |
| 94         | 61     | 45     | XT_OUT           | O                       | <b>CRYSTAL OUT:</b> This is the output pin from the internal inverting amplifier. It emits the inverted signal of XTAL1.   |
| 95         | 62     | 46     | XT_IN            | I<br>(ST)               | <b>CRYSTAL IN:</b> This is the input pin to the internal inverting amplifier. The system clock is from external crystal or resonator when FOSC[1:0] (CONFIG3[1:0]) are both logic 1 by default.  |
| 96         | 63     | 47     | ICE_DAT          | I/O                     | Serial Wired Debugger Data pin   |
| 97         | 64     | 48     | ICE_CLK          | I                       | Serial Wired Debugger Clock pin  |
| 57         | 36     | 26     | P0.0             | I/O                     | General purpose digital I/O pin  |
|            |        |        | PWM00            | O                       | PWM0 output of PWM Unit 0  |
| 56         | 35     | -      | P0.1             | I/O                     | General purpose digital I/O pin  |
|            |        |        | PWM01            | O                       | PWM1 output of PWM Unit 0  |
| 55         | 34     | 25     | P0.2             | I/O                     | General purpose digital I/O pin  |
|            |        |        | PWM02            | O                       | PWM2 output of PWM Unit 0  |
| 54         | 33     | -      | P0.3             | I/O                     | General purpose digital I/O pin  |
|            |        |        | PWM03            | O                       | PWM3 output of PWM Unit 0  |
|            |        |        | STADC            | I                       | ADC external trigger input   |

| Pin Number |        |        | Pin Name | Pin Type <sup>(1)</sup> | Description                          |
|------------|--------|--------|----------|-------------------------|--------------------------------------|
| 100-pin    | 64-pin | 48-pin |          |                         |                                      |
| 45         | 32     | 23     | P0.4     | I/O                     | General purpose digital I/O pin      |
|            |        |        | PWM04    | O                       | PWM4 output of PWM Unit 0            |
| 44         | 31     | -      | P0.5     | I/O                     | General purpose digital I/O pin      |
|            |        |        | PWM05    | O                       | PWM5 output of PWM Unit 0            |
| 43         | 30     | -      | P0.6     | I/O                     | General purpose digital I/O pin      |
|            |        |        | BKP01    | I                       | Brake input pin 1 of PWM Unit 0      |
| 42         | 29     | -      | P0.7     | I/O                     | General purpose digital I/O pin      |
|            |        |        | STADC    | I                       | ADC external trigger input           |
| 30         | 19     | 14     | P1.0     | I/O                     | General purpose digital I/O pin      |
|            |        |        | PWM10    | O                       | PWM0 output of PWM Unit 1            |
| 29         | 18     | 13     | P1.1     | I/O                     | General purpose digital I/O pin      |
|            |        |        | PWM11    | O                       | PWM1 output of PWM Unit 1            |
| 20         | 16     | 12     | P1.2     | I/O                     | General purpose digital I/O pin      |
|            |        |        | PWM12    | O                       | PWM2 output of PWM Unit 1            |
| 19         | 15     | 11     | P1.3     | I/O                     | General purpose digital I/O pin      |
|            |        |        | PWM13    | O                       | PWM3 output of PWM Unit 1            |
| 18         | 14     | 10     | P1.4     | I/O                     | General purpose digital I/O pin      |
|            |        |        | PWM14    | O                       | PWM4 output of PWM Unit 1            |
| 17         | 13     | 9      | P1.5     | I/O                     | General purpose digital I/O pin      |
|            |        |        | PWM15    | O                       | PWM5 output of PWM Unit 1            |
| 16         | 12     | -      | P1.6     | I/O                     | General purpose digital I/O pin      |
|            |        |        | BKP00    | I                       | Brake input pin 0 of PWM Unit 0      |
| 8          | 5      | 3      | P1.7     | I/O                     | General purpose digital I/O pin      |
|            |        |        | BKP10    | I                       | Brake input pin0 of PWM Unit 1       |
| 49         | -      | -      | P2.0     | I/O                     | General purpose digital I/O pin      |
|            |        |        | MOSI2    | I/O                     | SPI2 MOSI (Master Out, Slave In) pin |
|            |        |        | CPO2     | AO                      | Analog comparator 2 output pin       |
| 48         | -      | -      | P2.1     | I/O                     | General purpose digital I/O pin      |
|            |        |        | IC02     | I                       | Input 2 of Input Capture Unit 0      |
| 47         | -      | -      | P2.2     | I/O                     | General purpose digital I/O pin      |
|            |        |        | IC01     | I                       | Input 1 of Input Capture Unit 0      |
| 46         | -      | -      | P2.3     | I/O                     | General purpose digital I/O pin      |
|            |        |        | IC00     | I                       | Input 0 of Input Capture Unit 0      |



| Pin Number |        |        | Pin Name | Pin Type <sup>[1]</sup> | Description                                      |
|------------|--------|--------|----------|-------------------------|--|
| 100-pin    | 64-pin | 48-pin |          |                         |  |
| 41         | 28     | 22     | P2.4     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | QEIA0    | I                       | Quadarature encoder phase A input of QEI Unit 10 |
|            |        |        | CANRX    | I                       | CAN Bus RX Input (not supported in 48-pin)       |
| 40         | 27     | 21     | P2.5     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | QEIB0    | I                       | Quadarature encoder phase B input of QEI Unit 0  |
|            |        |        | CANTX    | O                       | CAN Bus TX Output (not supported in 48-pin)      |
| 39         | 26     | 20     | P2.6     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | IDX0     | I                       | Quadrature Encoder Index input of QEI Unit 0     |
|            |        |        | /SS0     | I/O                     | SPI0 slave select pin                            |
|            |        |        | CTS1     | I                       | UART1 CTS pin                                    |
| 38         | 25     | 19     | P2.7     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | SPI_CLK0 | I/O                     | SPI0 serial clock pin                            |
|            |        |        | RTS1     | O                       | UART1 RTS pin                                    |
| 31         | 20     | 15     | P3.0     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | RX0      | I                       | Data Receiver input pin for UART0                |
| 32         | 21     | 16     | P3.1     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | TX0      | O                       | Data transmitter output pin for UART0            |
|            |        |        | CPO0     | AO                      | Analog comparator 0 output                       |
| 7          | -      | 2      | P3.2     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | /INT0    | I                       | External Interrupt 0 input pin                   |
| 27         | 17     | -      | P3.3     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | /INT1    | I                       | External Interrupt 1 input pin                   |
| 6          | 4      | -      | P3.4     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | T0       | I/O                     | Timer0 external clock                            |
|            |        |        | I2CSDA   | I/O                     | I2C data input/output pin                        |
| 5          | 3      | -      | P3.5     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | T1       | I/O                     | Timer1 external clock                            |
|            |        |        | I2CSCL   | I/O                     | I2C clock output pin                             |
| 4          | -      | -      | P3.6     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | CANRX    | I                       | CAN Bus RX Input                                 |
| 3          | -      | -      | P3.7     | I/O                     | General purpose digital I/O pin                  |
|            |        |        | CANTX    | O                       | CAN Bus TX Output                                |
| 23         | -      | -      | P4.0     | I/O                     | General purpose digital I/O pin                  |

| Pin Number |        |        | Pin Name | Pin Type <sup>[1]</sup> | Description                                    |
|------------|--------|--------|----------|-------------------------|--|
| 100-pin    | 64-pin | 48-pin |          |                         |  |
|            |        |        | IC10     | I                       | Input 0 of Input Capture Unit 1                |
| 24         | -      | -      | P4.1     | I/O                     | General purpose digital I/O pin                |
|            |        |        | IC11     | I                       | Input 1 of Input Capture Unit 1                |
| 25         | -      | -      | P4.2     | I/O                     | General purpose digital I/O pin                |
|            |        |        | IC12     | I                       | Input 2 of Input Capture Unit 1                |
| 26         | -      | -      | P4.3     | I/O                     | General purpose digital I/O pin                |
| 21         | -      | -      | P4.4     | I/O                     | General purpose digital I/O pin                |
|            |        |        | QEIA1    | I                       | Quadrature encoder phase A input of QE1 Unit 1 |
| 22         | -      | -      | P4.5     | I/O                     | General purpose digital I/O pin                |
|            |        |        | QEIB1    | I                       | Quadrature encoder phase B input of QE1 Unit 1 |
| 28         | --     | -      | P4.6     | I/O                     | General purpose digital I/O pin                |
|            |        |        | T2       | I/O                     | Timer2 external clock                          |
|            |        |        | IDX1     | I                       | Quadrature Encoder Index input of QE1 Unit 1   |
| 33         | -      | -      | P4.7     | I/O                     | General purpose digital I/O pin                |
|            |        |        | T3       | I/O                     | Timer3 external clock                          |
| 36         | 23     | 17     | P5.0     | I/O                     | General purpose digital I/O pin                |
|            |        |        | MOSI0    | I/O                     | SPI0 MOSI (Master Out, Slave In) pin           |
|            |        |        | RTS0     | O                       | UART0 RTS pin                                  |
| 37         | 24     | 18     | P5.1     | I/O                     | General purpose digital I/O pin                |
|            |        |        | MISO0    | I/O                     | SPI0 MISO (Master In, Slave Out) pin           |
|            |        |        | CTS0     | I                       | UART0 CTS pin                                  |
| 50         | -      | 24     | P5.2     | I/O                     | General purpose digital I/O pin                |
|            |        |        | MISO2    | I/O                     | SPI2 MISO (Master In, Slave Out) pin           |
|            |        |        | CPO1     | AO                      | Analog comparator 1 output pin                 |
| 51         | -      | -      | P5.3     | I/O                     | General purpose digital I/O pin                |
|            |        |        | SPI_CLK2 | I/O                     | SPI2 serial clock pin                          |
| 52         | -      | -      | P5.4     | I/O                     | General purpose digital I/O pin                |
|            |        |        | /SS2     | I/O                     | SPI2 slave select pin                          |
| 53         | -      | -      | P5.5     | I/O                     | General purpose digital I/O pin                |
|            |        |        | CLKO     | O                       | Frequency Divider output pin                   |
| 15         | 11     | -      | P5.6     | I/O                     | General purpose digital I/O pin                |
|            |        |        | PWM20    | I/O                     | PWM0 output of PWM unit 2                      |
| 14         | -      | -      | P5.7     | I/O                     | General purpose digital I/O pin                |

| Pin Number |        |        | Pin Name | Pin Type <sup>[1]</sup> | Description                              |
|------------|--------|--------|----------|-------------------------|--|
| 100-pin    | 64-pin | 48-pin |          |                         |  |
|            |        |        | PWM21    | I/O                     | PWM1 output of PWM unit 2                |
| 69         | 42     | -      | P6.0     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINA0    | AI                      | ADC analog input 0 for sample-and-hold A |
| 68         | 42     | 31     | P6.1     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINA1    | AI                      | ADC analog input 1 for sample-and-hold A |
| 67         | 41     | 30     | P6.2     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINA2    | AI                      | ADC analog input 2 for sample-and-hold A |
| 66         | 40     | 29     | P6.3     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINA3    | AI                      | ADC analog input 3 for sample-and-hold A |
| 65         | 39     | 28     | P6.4     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINA4    | AI                      | ADC analog input 4 for sample-and-hold A |
|            |        |        | CPN1     | AI                      | Analog comparator 1 negative input       |
| 64         | 38     | 27     | P6.5     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINA5    | AI                      | ADC analog input 5 for sample-and-hold A |
|            |        |        | CPP1     | AI                      | Analog comparator 1 positive input       |
| 63         | 37     | -      | P6.6     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINA6    | AI                      | ADC analog input 6 for sample-and-hold A |
| 62         | -      | -      | P6.7     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINA7    | AI                      | ADC analog input 7 for sample-and-hold A |
| 83         | 55     | -      | P7.0     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINB0    | AI                      | ADC analog input 0 for sample-and-hold B |
| 82         | 54     | 40     | P7.1     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINB1    | AI                      | ADC analog input 1 for sample-and-hold B |
| 81         | 53     | 39     | P7.2     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINB2    | AI                      | ADC analog input 2 for sample-and-hold B |
| 80         | 52     | 38     | P7.3     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINB3    | AI                      | ADC analog input 3 for sample-and-hold B |
| 79         | 51     | -      | P7.4     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINB4    | AI                      | ADC analog input 4 for sample-and-hold B |
|            |        |        | CPN2     | AI                      | Analog comparator 2 negative input       |
| 78         | 50     | 37     | P7.5     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINB5    | AI                      | ADC analog input 5 for sample-and-hold B |
|            |        |        | CPP2     | AI                      | Analog comparator 2 positive input       |

| Pin Number |        |        | Pin Name | Pin Type <sup>(1)</sup> | Description                              |
|------------|--------|--------|----------|-------------------------|--|
| 100-pin    | 64-pin | 48-pin |          |                         |  |
| 77         | 49     | -      | P7.6     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINB6    | AI                      | ADC analog input 6 for sample-and-hold B |
| 76         | -      | -      | P7.7     | I/O                     | General purpose digital I/O pin          |
|            |        |        | AINB7    | AI                      | ADC analog input 7 for sample-and-hold B |
| 72         | 46     | 34     | P8.0     | I/O                     | General purpose digital I/O pin          |
|            |        |        | OPP0     | AI                      | OP Amplifier 0 positive input            |
| 71         | 45     | 33     | P8.1     | I/O                     | General purpose digital I/O pin          |
|            |        |        | OPN0     | AI                      | OP Amplifier 0 negative input            |
| 70         | 44     | 32     | P8.2     | I/O                     | General purpose digital I/O pin          |
|            |        |        | OPO0     | AO                      | OP Amplifier 0 output                    |
| 85         | -      | -      | P8.3     | I/O                     | General purpose digital I/O pin          |
|            |        |        | CPN0     | AI                      | Analog comparator negative input pin     |
| 84         | -      | -      | P8.4     | I/O                     | General purpose digital I/O pin          |
|            |        |        | CPP0     | AI                      | Analog comparator positive input pin     |
| 91         | -      | -      | P8.5     | I/O                     | General purpose digital I/O pin          |
| 59         | -      | -      | P8.6     | I/O                     | General purpose digital I/O pin          |
| 58         | -      | -      | P8.7     | I/O                     | General purpose digital I/O pin          |
|            |        |        | CPO0     | O                       | Analog comparator output pin             |
| 86         | 56     | 41     | P9.0     | I/O                     | General purpose digital I/O pin          |
|            |        |        | OPO1     | AO                      | OP Amplifier 1 output                    |
| 87         | 57     | 42     | P9.1     | I/O                     | General purpose digital I/O pin          |
|            |        |        | OPN1     | AI                      | OP Amplifier 1 negative input            |
| 88         | 58     | 43     | P9.2     | I/O                     | General purpose digital I/O pin          |
|            |        |        | OPP1     | AI                      | OP Amplifier 1 positive input            |
| 92         | -      | -      | P9.3     | I/O                     | General purpose digital I/O pin          |
|            |        |        | BKP11    | I                       | Brake input pin 1 of PWM Unit 1          |
| 98         | -      | -      | P9.4     | I/O                     | General purpose digital I/O pin          |
|            |        |        | SPI_CLK1 | I/O                     | SPI1 serial clock pin                    |
| 99         | -      | -      | P9.5     | I/O                     | General purpose digital I/O pin          |
|            |        |        | MISO1    | I/O                     | SPI1 MISO (Master In, Slave Out) pin     |
| 100        | -      | -      | P9.6     | I/O                     | General purpose digital I/O pin          |
|            |        |        | MOSI1    | I/O                     | SPI1 MOSI (Master Out, Slave In) pin     |
| 2          | 2      | -      | P9.7     | I/O                     | General purpose digital I/O pin          |

| Pin Number |        |        | Pin Name | Pin Type <sup>[1]</sup> | Description                           |
|------------|--------|--------|----------|-------------------------|---------------------------------------|
| 100-pin    | 64-pin | 48-pin |          |                         |                                       |
|            |        |        | /SS1     | I/O                     | SPI1 slave select pin                 |
| 13         | 10     | 8      | PA.0     | I/O                     | General purpose digital I/O pin       |
|            |        |        | TX1      | O                       | Data transmitter output pin for UART1 |
|            |        |        | I2CSDA   | I/O                     | I2C data input/output pin             |
| 12         | 9      | 7      | PA.1     | I/O                     | General purpose digital I/O pin       |
|            |        |        | RX1      | I                       | Data Receiver input pin for UART1     |
|            |        |        | I2CSCL   | I/O                     | I2C clock output pin                  |

**Note:** Pin Type I = Digital Input, O = Digital Output; AI = Analog Input; P = Power Pin; AP = Analog Power

## 6 ARM® CORTEX™-M0 CORE

The Cortex™-M0 processor is a configurable, multistage, 32-bit RISC processor, which has an AMBA AHB-Lite interface and includes an NVIC component. It also has optional hardware debug functionality. The processor can execute Thumb code and is compatible with other Cortex-M profile processor. The profile supports two modes –Thread mode and Handler mode. Handler mode is entered as a result of an exception. An exception return can only be issued in Handler mode. Thread mode is entered on Reset, and can be entered as a result of an exception return. Figure 6–1 shows the functional controller of processor.

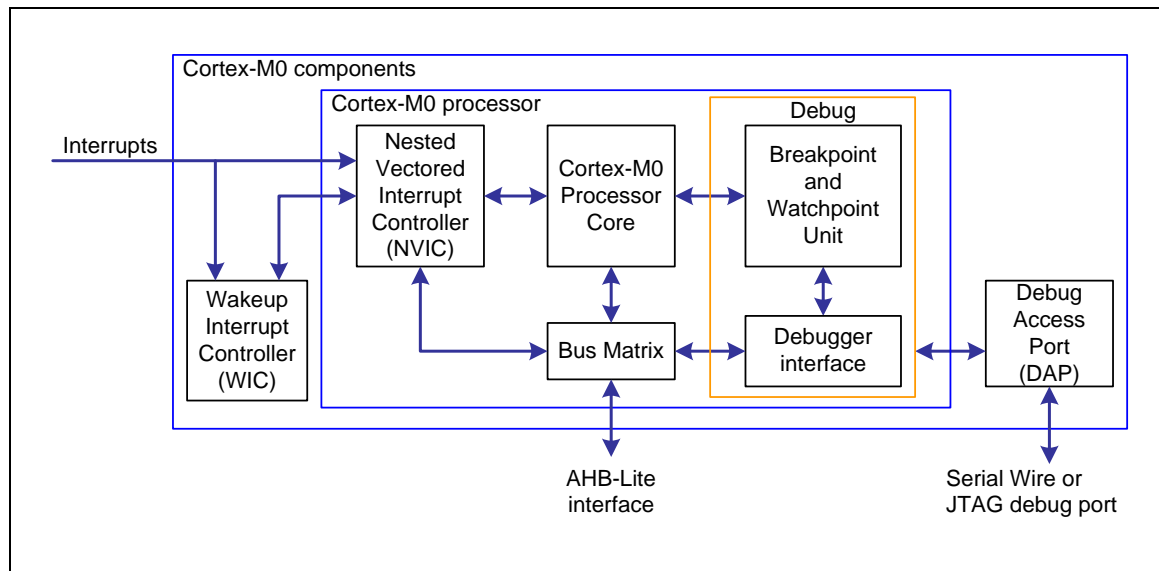


Figure 6–1 Functional Controller Diagram

The implemented device provides:

- A low gate count processor:
  - ◆ ARMv6-M Thumb® instruction set
  - ◆ Thumb-2 technology
  - ◆ ARMv6-M compliant 24-bit SysTick timer
  - ◆ A 32-bit hardware multiplier
  - ◆ The system interface supports little-endian data accesses
  - ◆ The ability to have deterministic, fixed-latency, interrupt handling
  - ◆ Load/store-multiples and multicycle-multiplies that can be abandoned and restarted to facilitate rapid interrupt handling
  - ◆ C Application Binary Interface compliant exception model. This is the ARMv6-M, C Application Binary Interface (C-ABI) compliant exception model that enables the use of pure C functions as interrupt handlers
  - ◆ Low power sleep-mode entry using Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or the return from interrupt sleep-on-exit feature

- NVIC:
  - ◆ 32 external interrupt inputs, each with four levels of priority
  - ◆ Dedicated Non-Maskable Interrupt (NMI) input
  - ◆ Supports for both level-sensitive and pulse-sensitive interrupt lines
  - ◆ Supports Wake-up Interrupt Controller (WIC) and providing ultra-low power sleep mode
- Debug support:
  - ◆ Four hardware breakpoints
  - ◆ Two watchpoints
  - ◆ Program Counter Sampling Register (PCSR) for non-intrusive code profiling
  - ◆ Single step and vector catch capabilities
- Bus interfaces:
  - ◆ Single 32-bit AMBA-3 AHB-Lite system interface that provides simple integration to all system peripherals and memory
  - ◆ Single 32-bit slave port that supports the DAP (Debug Access Port)

## 7 SYSTEM MANAGEMENT

### 7.1 Overview

System management includes the following sections:

- System Resets
- System Memory Map
- System management registers for Part Number ID, chip reset and on-chip controllers reset , multi-functional pin control
- System Timer (SysTick)
- Nested Vectored Interrupt Controller (NVIC)
- System Control registers



### 7.2 System Reset

The system reset can be issued by one of the following listed events. For these reset event flags can be read by RSTRC register.

- Hardware Reset
  - Power-On Reset (POR)
  - Low level on the Reset pin (nRST)
  - Watchdog Time-out Reset (WDT)
  - Low Voltage Reset (LVR)
  - Brown-out Detector Reset (BOD)
- Software Reset
  - MCU Reset - SYSRESETREQ (AIRCR[2])
  - Cortex-M0 Core One-shot Reset - CPU\_RST (IPRSTC1[1])
  - Chip One-shot Reset - CHIP\_RST (IPRSTC1[0])

**Note:** ISPCON.BS keeps the original value after MCU Reset and Cortex-M0 Core One-shot Reset.

### 7.3 System Power Distribution

In this chip, the power distribution is divided into two segments.

- Analog power from  $AV_{DD}$  and  $AV_{SS}$  provides the power for analog components operation.
- Digital power from  $V_{DD}$  and  $V_{SS}$  supplies the power to the I/O pins and internal regulator which provides a fixed 1.8V power for digital operation.

The output of internal voltage regulators, LDO\_CAP, requires an external capacitor which should be located close to the corresponding pin. Analog power ( $AV_{DD}$ ) should be the same voltage level of the digital power ( $V_{DD}$ ).

### 7.4 System Memory Map

The NuMicro™ NM15xx Series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in the following table. The detailed register definition, memory space, and programming detailed will be described in the following sections for each on-chip peripheral. The NuMicro™ NM15xx Series only supports little-endian data format.

| Address Space   | Token    | Controllers                                       |
|---|----------|---|
| <b>Flash and SRAM Memory Space</b>                        |          |   |
| 0x0000_0000 – 0x0001_FFFF                                 | FLASH_BA | FLASH Memory Space (128 KB)                       |
| 0x2000_0000 – 0x2000_3FFF                                 | SRAM_BA  | SRAM Memory Space (16 KB)                         |
| <b>AHB Controllers Space (0x5000_0000 – 0x501F_FFFF)</b>  |          |   |
| 0x5000_0000 – 0x5000_01FF                                 | GCR_BA   | System Global Control Registers                   |
| 0x5000_0200 – 0x5000_02FF                                 | CLK_BA   | Clock Control Registers                           |
| 0x5000_0300 – 0x5000_03FF                                 | INT_BA   | Interrupt Multiplexer Control Registers           |
| 0x5000_4000 – 0x5000_7FFF                                 | GPIO_BA  | GPIO Control Registers                            |
| 0x5000_C000 – 0x5000_FFFF                                 | FMC_BA   | Flash Memory Control Registers                    |
| 0x5001_4000 – 0x5001_7FFF                                 | DIV_BA   | Hardware Divider Register                         |
| <b>APB1 Controllers Space (0x4000_0000 ~ 0x400F_FFFF)</b> |          |   |
| 0x4000_4000 – 0x4000_7FFF                                 | WDT_BA   | Watchdog Timer Control Registers                  |
| 0x4000_4100 – 0x4000_7FFF                                 | WWDT_BA  | Window Watchdog Timer Control Registers           |
| 0x4001_0000 – 0x4001_3FFF                                 | TMR01_BA | Timer0/Timer1 Control Registers                   |
| 0x4002_0000 – 0x4002_3FFF                                 | I2C_BA   | I <sup>2</sup> C0 Interface Control Registers     |
| 0x4003_0000 – 0x4003_3FFF                                 | SPI0_BA  | SPI0 with master/slave function Control Registers |
| 0x4003_4000 – 0x4003_7FFF                                 | SPI1_BA  | SPI1 with master/slave function Control Registers |
| 0x4004_0000 – 0x4004_3FFF                                 | BPWM_BA  | Basic PWM Control Registers                       |
| 0x4005_0000 – 0x4005_3FFF                                 | UART0_BA | UART0 Control Registers                           |
| 0x400D_0000 – 0x400D_3FFF                                 | ACMP_BA  | Analog Comparator Control Registers               |
| 0x400E_0000 – 0x400E_3FFF                                 | ADC_BA   | Analog-Digital-Converter (ADC) Control Registers  |
| 0x400F_0000 – 0x400F_3FFF                                 | OPA_BA   | Operation Amplifier Control Registers             |
| <b>APB2 Controllers Space (0x4010_0000 ~ 0x401F_FFFF)</b> |          |   |
| 0x4011_0000 – 0x4011_3FFF                                 | TMR23_BA | Timer2/Timer3 Control Registers                   |

| Address Space   | Token    | Controllers                                       |
|---|----------|---|
| 0x4013_0000 – 0x4013_3FFF                                   | SPI2_BA  | SPI2 with master/slave function Control Registers |
| 0x4015_0000 – 0x4015_3FFF                                   | UART1_BA | UART1 Control Registers                           |
| 0x4018_0000 – 0x4018_3FFF                                   | CAN_BA   | CAN Bus Control Registers                         |
| 0x4019_0000 – 0x4019_3FFF                                   | EPWM0_BA | Enhanced PWM0 Control Registers                   |
| 0x4019_4000 – 0x4019_7FFF                                   | EPWM1_BA | Enhanced PWM1 Control Registers                   |
| 0x401B_0000 – 0x401B_3FFF                                   | CAP0_BA  | Input Capture 0 Control Registers                 |
| 0x401B_4000 – 0x401B_7FFF                                   | CAP1_BA  | Input Capture 1 Control Registers                 |
| 0x401C_0000 – 0x401C_3FFF                                   | QEI0_BA  | Quadrature Encoder Interface 0 Control Registers  |
| 0x401C_4000 – 0x401C_7FFF                                   | QEI1_BA  | Quadrature Encoder Interface 1 Control Registers  |
| 0x401D_0000 – 0x401D_3FFF                                   | MDU_BA   | Motor Drive Unit Control Registers                |
| <b>System Controllers Space (0xE000_E000 ~ 0xE000_EFFF)</b> |          |   |
| 0xE000_E010 – 0xE000_E01F                                   | SYST_BA  | System Timer Control Registers                    |
| 0xE000_E100 – 0xE000_E4EF                                   | NVIC_BA  | External Interrupt Controller Control Registers   |
| 0xE000_ED00 – 0xE000_ED3F                                   | SCS_BA   | System Control Registers                          |

Table 7-1 Address Space Assignments for On-Chip Controllers

### 7.5 System Manager Controller Register Map

R: read only, W: write only, R/W: both read and write

| Register                    | Offset       | R/W | Description  | Reset Value |
|-----------------------------|--------------|-----|--|-------------|
| <b>GCR Base Address:</b>    |              |     |  |             |
| <b>GCR_BA = 0x5000_0000</b> |              |     |  |             |
| <b>PDID</b>                 | GCR_BA+0x00  | R   | Part Device Identification Number Register           | 0x0035_X0XX |
| <b>RSTSRC</b>               | GCR_BA+0x04  | R/W | System Reset Source Register                         | 0x0000_00XX |
| <b>IPRSTC1</b>              | GCR_BA+0x08  | R/W | Peripheral Reset Control Register1                   | 0x0000_0000 |
| <b>IPRSTC2</b>              | GCR_BA+0x0C  | R/W | Peripheral Reset Control Register2                   | 0x0000_0000 |
| <b>BODCR</b>                | GCR_BA+0x18  | R/W | Brown Out Detector Control Register                  | 0x0000_008X |
| <b>TEMPCR</b>               | GCR_BA+0x1C  | R/W | Temperature Sensor Control Register                  | 0x0000_0000 |
| <b>BODDBCR</b>              | GCR_BA+0x20  | R/W | Brown Out Detector De-bounce Control Register        | 0x0000_0000 |
| <b>PORCR</b>                | GCR_BA+0x24  | R/W | Power-On-Reset Controller Register                   | 0x0000_00xx |
| <b>P0_MFP</b>               | GCR_BA+0x30  | R/W | P0 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P1_MFP</b>               | GCR_BA+0x34  | R/W | P1 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P2_MFP</b>               | GCR_BA+0x38  | R/W | P2 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P3_MFP</b>               | GCR_BA+0x3C  | R/W | P3 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P4_MFP</b>               | GCR_BA+0x40  | R/W | P4 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P5_MFP</b>               | GCR_BA+0x44  | R/W | P5 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P6_MFP</b>               | GCR_BA+0x48  | R/W | P6 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P7_MFP</b>               | GCR_BA+0x4C  | R/W | P7 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P8_MFP</b>               | GCR_BA+0x50  | R/W | P8 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>P9_MFP</b>               | GCR_BA+0x54  | R/W | P9 Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>PA_MFP</b>               | GCR_BA+0x58  | R/W | PA Multiple Function and Input Type Control Register | 0x0000_0000 |
| <b>REGWRPROT</b>            | GCR_BA+0x100 | R/W | Register Write-Protection Control Register           | 0x0000_0000 |

### Part Device ID Code Register (PDID)

| Register | Offset      | R/W | Description                                | Reset Value                |
|----------|-------------|-----|--|----------------------------|
| PDID     | GCR_BA+0x00 | R   | Part Device Identification Number Register | 0x0035_X0XX <sup>[1]</sup> |

[1] Every part number has a unique default reset value.

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| PDID[31:24] |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PDID[23:16] |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PDID[15:8]  |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PDID[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [31:0] | <p><b>PDID</b></p> <p><b>Part Device Identification Number</b></p> <p>This register reflects the device part number code. Software can read this register to identify which device is used.</p> |

### System Reset Source Register (RSTSRC)

This register provides specific information for software to identify this chip's reset source from last operation.

| Register | Offset      | R/W | Description                  | Reset Value |
|----------|-------------|-----|------------------------------|-------------|
| RSTSRC   | GCR_BA+0x04 | R/W | System Reset Source Register | 0x0000_00XX |

|          |          |          |          |          |          |            |          |
|----------|----------|----------|----------|----------|----------|------------|----------|
| 31       | 30       | 29       | 28       | 27       | 26       | 25         | 24       |
| Reserved |          |          |          |          |          |            |          |
| 23       | 22       | 21       | 20       | 19       | 18       | 17         | 16       |
| Reserved |          |          |          |          |          |            |          |
| 15       | 14       | 13       | 12       | 11       | 10       | 9          | 8        |
| Reserved |          |          |          |          |          |            |          |
| 7        | 6        | 5        | 4        | 3        | 2        | 1          | 0        |
| RSTS_CPU | Reserved | RSTS_SYS | RSTS_BOD | RSTS_LVR | RSTS_WDT | RSTS_RESET | RSTS_POR |

| Bits   | Description |  |
|--------|-------------|--|
| [31:8] | Reserved    | Reserved.  |
| [7]    | RSTS_CPU    | <b>CPU Reset Flag</b><br>The RSTS_CPU flag is set by hardware if software writes CPU_RST (IPRSTC1[1]) 1 to reset Cortex™-M0 core and Flash memory controller (FMC).<br>0 = No reset from CPU.<br>1 = Cortex™-M0 core and FMC are reset by software setting CPU_RST to 1.<br><b>Note:</b> Write 1 to clear this bit to 0.   |
| [6]    | Reserved    | Reserved.  |
| [5]    | RSTS_MCU    | <b>MCU Reset Flag</b><br>The RSTS_MCU flag is set by the "reset signal" from the Cortex™-M0 core to indicate the previous reset source.<br>0 = No reset from Cortex™-M0.<br>1 = The Cortex™-M0 had issued the reset signal to reset the system by writing 1 to bit SYSRESETREQ (AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE00ED0C) in system control registers of Cortex™-M0 core.<br><b>Note:</b> Write 1 to clear this bit to 0. |
| [4]    | RSTS_BOD    | <b>Brown-out Detector Reset Flag</b><br>The RSTS_BOD flag is set by the "reset signal" from the Brown-out Detector to indicate the previous reset source.<br>0 = No reset from BOD.<br>1 = The BOD had issued the reset signal to reset the system<br><b>Note:</b> Write 1 to clear this bit to 0.   |

| Bits | Description       |   |
|------|-------------------|---|
| [3]  | <b>RSTS_LVR</b>   | <p><b>Low Voltage Reset Flag</b></p> <p>The RSTS_LVR flag is set by the “reset signal” from the Low-Voltage-Reset controller to indicate the previous reset source.</p> <p>0 = No reset from LVR.</p> <p>1 = The LVR controller had issued the reset signal to reset the system.</p> <p><b>Note:</b> Write 1 to clear this bit to 0.</p>  |
| [2]  | <b>RSTS_WDT</b>   | <p><b>Watchdog Reset Flag</b></p> <p>The RSTS_WDT flag is set by the “reset signal” from the watchdog timer to indicate the previous reset source.</p> <p>0 = No reset from watchdog timer.</p> <p>1 = The watchdog timer had issued the reset signal to reset the system.</p> <p><b>Note:</b> Write 1 to clear this bit to 0.</p>  |
| [1]  | <b>RSTS_RESET</b> | <p><b>Reset Pin Reset Flag</b></p> <p>The RSTS_RESET flag is set by the “reset signal” from the nRST pin to indicate the previous reset source.</p> <p>0 = No reset from the nRST pin.</p> <p>1 = The nRST pin had issued the reset signal to reset the system.</p> <p><b>Note:</b> Write 1 to clear this bit to 0.</p>   |
| [0]  | <b>RSTS_POR</b>   | <p><b>Power-on Reset Flag</b></p> <p>The RSTS_POR flag is set by the “reset signal” from the Power-on Reset (POR) controller or bit CHIP_RST (IPRSTC1[0]) to indicate the previous reset source.</p> <p>0 = No reset from POR or CHIP_RST.</p> <p>1 = The Power-on Reset (POR) or CHIP_RST had issued the reset signal to reset the system.</p> <p><b>Note:</b> Write 1 to clear this bit to 0.</p> |



### Peripheral Reset Control Register1 (IPRSTC1)

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| IPRSTC1  | GCR_BA+0x08 | R/W | Peripheral Reset Control Register1 | 0x0000_0000 |

|          |    |    |          |          |    |         |          |
|----------|----|----|----------|----------|----|---------|----------|
| 31       | 30 | 29 | 28       | 27       | 26 | 25      | 24       |
| Reserved |    |    |          |          |    |         |          |
| 23       | 22 | 21 | 20       | 19       | 18 | 17      | 16       |
| Reserved |    |    |          |          |    |         |          |
| 15       | 14 | 13 | 12       | 11       | 10 | 9       | 8        |
| Reserved |    |    |          |          |    |         |          |
| 7        | 6  | 5  | 4        | 3        | 2  | 1       | 0        |
| Reserved |    |    | HDIV_RST | Reserved |    | CPU_RST | CHIP_RST |

| Bits   | Description |   |
|--------|-------------|---|
| [31:5] | Reserved    | Reserved.   |
| [4]    | HDIV_RST    | <p><b>HDIV Controller Reset (Write Protect)</b></p> <p>Set this bit to 1 will generate a reset signal to the hardware divider. User need to set this bit to 0 to release from the reset state.</p> <p>0 = Hardware divider controller normal operation.</p> <p>1 = Hardware divider controller reset.</p> <p><b>Note:</b> This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</p>  |
| [3:2]  | Reserved    | Reserved.   |
| [1]    | CPU_RST     | <p><b>Cortex-M0 Core One-shot Reset (Write Protect)</b></p> <p>Setting this bit will only reset the Cortex-M0 core and Flash Memory Controller (FMC), and this bit will automatically return 0 after the two clock cycles.</p> <p>0 = Cortex-M0 core normal operation.</p> <p>1 = Cortex-M0 core one-shot reset.</p> <p><b>Note:</b> This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</p>   |
| [0]    | CHIP_RST    | <p><b>Chip One-shot Reset (Write Protect)</b></p> <p>Setting this bit will reset the whole chip, including Cortex-M0 core and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles.</p> <p>The CHIP_RST is the same as the POR reset. All the chip controllers are reset and the chip setting from CONFIG0 are also reload.</p> <p>0 = Chip normal operation.</p> <p>1 = Chip one-shot reset.</p> <p><b>Note:</b> This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</p> |



### Peripheral Reset Control Register2 (IPRSTC2)

Setting these bits 1 will generate asynchronous reset signals to the corresponding IP controller. Users need to set these bits to 0 to release corresponding IP controller from reset state

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| IPRSTC2  | GCR_BA+0x0C | R/W | Peripheral Reset Control Register2 | 0x0000_0000 |

| 31       | 30       | 29        | 28        | 27       | 26       | 25        | 24        |
|----------|----------|-----------|-----------|----------|----------|-----------|-----------|
| QE11_RST | QE10_RST | OPA_RST   | ADC_RST   | CAP1_RST | CAP0_RST | Reserved  | CAN_RST   |
| 23       | 22       | 21        | 20        | 19       | 18       | 17        | 16        |
| Reserved | ACMP_RST | EPWM1_RST | EPWM0_RST | BPWM_RST | MDU_RST  | UART1_RST | UART0_RST |
| 15       | 14       | 13        | 12        | 11       | 10       | 9         | 8         |
| Reserved | SPI2_RST | SPI1_RST  | SPI0_RST  | Reserved |          |           | I2C_RST   |
| 7        | 6        | 5         | 4         | 3        | 2        | 1         | 0         |
| Reserved |          | TMR3_RST  | TMR2_RST  | TMR1_RST | TMR0_RST | GPIO_RST  | Reserved  |

| Bits | Description |  |
|------|-------------|--|
| [31] | QE11_RST    | <b>QE11 Controller Reset</b><br>0 = QE11 controller normal operation.<br>1 = QE11 controller reset.                                  |
| [30] | QE10_RST    | <b>QE10 Controller Reset</b><br>0 = QE10 controller normal operation.<br>1 = QE10 controller reset.                                  |
| [29] | OPA_RST     | <b>OPA0 and OPA1 Controller Reset</b><br>0 = OPA0 and OPA1 controller normal operation.<br>1 = OPA0 and OPA1 controller reset.       |
| [28] | ADC_RST     | <b>ADC Controller Reset</b><br>0 = ADC controller normal operation.<br>1 = ADC controller reset.                                     |
| [27] | CAP1_RST    | <b>Input Capture 1 Controller Reset</b><br>0 = Input capture 1 controller normal operation.<br>1 = Input capture 1 controller reset. |
| [26] | CAP0_RST    | <b>Input Capture 0 Controller Reset</b><br>0 = Input capture 0 controller normal operation.<br>1 = Input capture 0 controller reset. |
| [25] | Reserved    | Reserved.  |
| [24] | CAN_RST     | <b>CAN Controller Reset</b><br>0 = CAN controller normal operation.<br>1 = CAN controller reset.                                     |

| Bits   | Description |  |
|--------|-------------|--|
| [23]   | Reserved    | Reserved.  |
| [22]   | ACMP_RST    | <b>Analog Comparator Controller Reset</b><br>0 = Analog Comparator controller normal operation.<br>1 = Analog Comparator controller reset. |
| [21]   | EPWM1_RST   | <b>Enhanced PWM1 Controller Reset</b><br>0 = EPWM1 controller normal operation.<br>1 = EPWM1 controller reset.                             |
| [20]   | EPWM0_RST   | <b>Enhanced PWM0 Controller Reset</b><br>0 = EPWM0 controller normal operation.<br>1 = EPWM0 controller reset.                             |
| [19]   | BPWM_RST    | <b>Basic PWM Controller Reset</b><br>0 = Basic PWM controller normal operation.<br>1 = Basic PWM controller reset.                         |
| [18]   | MDU_RST     | <b>MDU Controller Reset</b><br>0 = MDU controller normal operation.<br>1 = MDU controller reset.   |
| [17]   | UART1_RST   | <b>UART1 Controller Reset</b><br>0 = UART1 controller normal operation<br>1 = UART1 controller reset                                       |
| [16]   | UART0_RST   | <b>UART0 Controller Reset</b><br>0 = UART0 controller normal operation.<br>1 = UART0 controller reset.                                     |
| [15]   | Reserved    | Reserved.  |
| [14]   | SPI2_RST    | <b>SPI2 Controller Reset</b><br>0 = SPI2 controller normal operation.<br>1 = SPI2 controller reset.  |
| [13]   | SPI1_RST    | <b>SPI1 Controller Reset</b><br>0 = SPI1 controller normal operation.<br>1 = SPI1 controller reset.  |
| [12]   | SPI0_RST    | <b>SPI0 Controller Reset</b><br>0 = SPI0 controller normal operation.<br>1 = SPI0 controller reset.  |
| [11:9] | Reserved    | Reserved.  |
| [8]    | I2C_RST     | <b>I2C Controller Reset</b><br>0 = I <sup>2</sup> C controller normal operation.<br>1 = I <sup>2</sup> C controller reset.                 |
| [7:6]  | Reserved    | Reserved.  |

| Bits | Description     |   |
|------|-----------------|---|
| [5]  | <b>TMR3_RST</b> | <b>Timer3 Controller Reset</b><br>0 = Timer3 controller normal operation.<br>1 = Timer3 controller reset. |
| [4]  | <b>TMR2_RST</b> | <b>Timer2 Controller Reset</b><br>0 = Timer2 controller normal operation.<br>1 = Timer2 controller reset. |
| [3]  | <b>TMR1_RST</b> | <b>Timer1 Controller Reset</b><br>0 = Timer1 controller normal operation.<br>1 = Timer1 controller reset. |
| [2]  | <b>TMR0_RST</b> | <b>Timer0 Controller Reset</b><br>0 = Timer0 controller normal operation.<br>1 = Timer0 controller reset. |
| [1]  | <b>GPIO_RST</b> | <b>GPIO Controller Reset</b><br>0 = GPIO controller normal operation.<br>1 = GPIO controller reset.       |
| [0]  | <b>Reserved</b> | Reserved.   |

|     |                 |   |
|-----|-----------------|---|
| [2] | <b>POR_TEST</b> | <b>POR Test Enable</b><br>When this bit set to 1,<br>GP6[1] will force as LVR output without deglitch circuit<br>GP6[2] will force as LVR output with deglitch circuit<br>GP6[3] will force as BOD output<br>GPA[0] will force as POR50 output<br>GPA[1] will force as POR18 output |
| [1] | <b>DLY_TEST</b> | <b>Delay Test Enable</b><br>Enable this bit   |
| [0] | <b>SELFTEST</b> | <b>Input capture 0 controller reset.</b><br>0 = I/O normal run.<br>1 = I/O self test mode enable. While enabled,<br>(1) Pin input -> Multi function without Pn_MFP switch.<br>(2) UART Rx and Tx internal loop back.<br>(3) BPWM internal loop back.<br>(4) EPWM -> ECAP.           |

### Brown-Out Detector Control Register (BODCR)

Partial of the BODCR control registers values are initiated by the flash configuration and partial bits are write-protected bit. Programming write-protected bits needs to write “59h”, “16h”, “88h” to address 0x5000\_0100 to disable register protection. Reference the register REGWRPROT at address GCR\_BA+0x100

| Register     | Offset      | R/W | Description                         | Reset Value |
|--------------|-------------|-----|-------------------------------------|-------------|
| <b>BODCR</b> | GCR_BA+0x18 | R/W | Brown Out Detector Control Register | 0x0000_008X |

|          |         |         |          |           |        |    |        |
|----------|---------|---------|----------|-----------|--------|----|--------|
| 31       | 30      | 29      | 28       | 27        | 26     | 25 | 24     |
| Reserved |         |         |          |           |        |    |        |
| 23       | 22      | 21      | 20       | 19        | 18     | 17 | 16     |
| Reserved |         |         |          |           |        |    |        |
| 15       | 14      | 13      | 12       | 11        | 10     | 9  | 8      |
| Reserved |         |         |          |           |        |    |        |
| 7        | 6       | 5       | 4        | 3         | 2      | 1  | 0      |
| LVR_EN   | BOD_OUT | BOD_LPM | BOD_INTF | BOD_RSTEN | BOD_VL |    | BOD_EN |

| Bits   | Description  |
|--------|--|
| [31:8] | <b>Reserved</b><br>Reserved.   |
| [7]    | <b>LVR_EN</b><br><b>Low Voltage Reset Enable Control (Write Protect)</b><br>The LVR function reset the chip when the input power voltage is lower than LVR circuit setting. LVR function is enabled by default.<br>0 = Low Voltage Reset function Disabled.<br>1 = Low Voltage Reset function Enabled – After enabling the bit, the LVR function will be active with 100us delay for LVR output stable (default).<br><b>Note:</b> This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100. |
| [6]    | <b>BOD_OUT</b><br><b>Brown-out Detector Output Status</b><br>0 = Brown-out Detector output status is 0, which means the detected voltage is higher than BOD_VL setting or BOD_EN is 0.<br>1 = Brown-out Detector output status is 1, which means the detected voltage is lower than BOD_VL setting. If the BOD_EN is 0, BOD function disabled, this bit always responds to 0.  |
| [5]    | <b>BOD_LPM</b><br><b>Brown-out Detector Low power Mode (Write Protect)</b><br>0 = BOD operated in Normal mode (default).<br>1 = BOD Low Power mode Enabled.<br><b>Note1:</b> The BOD consumes about 100 uA in Normal mode, and the low power mode can reduce the current to about 1/10 but slow the BOD response.<br><b>Note2:</b> This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.  |

| Bits  | Description      |   |
|-------|------------------|---|
| [4]   | <b>BOD_INTF</b>  | <p><b>Brown-out Detector Interrupt Flag</b></p> <p>0 = Brown-out Detector does not detect any voltage draft at <math>V_{DD}</math> down through or up through the voltage of BOD_VL setting.</p> <p>1 = When Brown-out Detector detects the <math>V_{DD}</math> is dropped down through the voltage of BOD_VL setting or the <math>V_{DD}</math> is raised up through the voltage of BOD_VL setting, this bit is set to 1 and the Brown-out interrupt is requested if Brown-out interrupt is enabled.</p> <p><b>Note:</b> Write 1 to clear this bit to 0.</p>   |
| [3]   | <b>BOD_RSTEN</b> | <p><b>Brown-out Reset Enable Control (Write Protect)</b></p> <p>0 = Brown-out "INTERRUPT" function Enabled.</p> <p>While the BOD function is enabled (BOD_EN high) and BOD interrupt function is enabled (BOD_RSTEN low), BOD will assert an interrupt if BOD_OUT is high. BOD interrupt will keep till to the BOD_EN set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BOD_EN low).</p> <p>1 = Brown-out "RESET" function Enabled.</p> <p><b>Note1:</b> While the Brown-out Detector function is enabled (BOD_EN high) and BOD reset function is enabled (BOD_RSTEN high), BOD will assert a signal to reset chip when the detected voltage is lower than the threshold (BOD_OUT high).</p> <p><b>Note2:</b> The default value is set by flash controller user configuration register config0 bit[20].</p> <p><b>Note3:</b> This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</p> |
| [2:1] | <b>BOD_VL</b>    | <p><b>Brown-out Detector Threshold Voltage Select (Write Protect)</b></p> <p>The default value is set by flash controller user configuration register config0 bit[22:21]</p> <p>00 = Brown-out voltage is 2.2V</p> <p>01 = Brown-out voltage is 2.7V</p> <p>10 = Brown-out voltage is 3.7V</p> <p>11 = Brown-out voltage is 4.4V</p> <p><b>Note:</b> This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</p>   |
| [0]   | <b>BOD_EN</b>    | <p><b>Brown-out Detector Enable Control (Write Protect)</b></p> <p>The default value is set by flash controller user configuration register config0 bit[23]</p> <p>0 = Brown-out Detector function Disabled.</p> <p>1 = Brown-out Detector function Enabled.</p> <p><b>Note:</b> This bit is the protected bit, and programming it needs to write "59h", "16h", and "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</p>   |

### Temperature Sensor Control Register (TEMPCR)

| Register | Offset      | R/W | Description                         | Reset Value |
|----------|-------------|-----|-------------------------------------|-------------|
| TEMPCR   | GCR_BA+0x1C | R/W | Temperature Sensor Control Register | 0x0000_0000 |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | VTEMP_EN |

| Bits   | Description   |
|--------|---|
| [31:1] | <b>Reserved</b><br>Reserved.  |
| [0]    | <b>Temperature Sensor Enable Control</b><br>This bit is used to enable/disable temperature sensor function.<br>0 = Temperature sensor function Disabled (default).<br>1 = Temperature sensor function Enabled.<br><b>Note:</b> After this bit is set to 1, the value of temperature sensor output can be obtained from the ADC conversion result. Please refer to the ADC chapter for detailed ADC conversion functional description. |



### Temperature Sensor Control Register (BODDBCR)

| Register       | Offset      | R/W | Description                                   | Reset Value |
|----------------|-------------|-----|---|-------------|
| <b>BODDBCR</b> | GCR_BA+0x20 | R/W | Brown Out Detector De-bounce Control Register | 0x0000_0000 |

|          |    |    |    |    |    |    |                  |
|----------|----|----|----|----|----|----|------------------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24               |
| Reserved |    |    |    |    |    |    |                  |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16               |
| Reserved |    |    |    |    |    |    |                  |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8                |
| Reserved |    |    |    |    |    |    |                  |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0                |
| Reserved |    |    |    |    |    |    | <b>BODDB_DIS</b> |

| Bits   | Description      |  |
|--------|------------------|--|
| [31:1] | <b>Reserved</b>  | Reserved.  |
| [0]    | <b>BODDB_DIS</b> | <b>Brown Out Detector De-bounce Disable</b><br>0 = Brown-out detector de-bounce function Enabled.<br>1 = Brown-out detector de-bounce function Disabled.<br>When Brown-out detector de-bounce function enable, the brown-out will have (5*SYS_CLK) period de-bounce ability. |

### Power-On-Reset Control Register (PORCR)

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| PORCR    | GCR_BA+0x24 | R/W | Power-On-Reset Controller Register | 0x0000_00XX |

|                    |    |    |    |    |    |    |    |
|--------------------|----|----|----|----|----|----|----|
| 31                 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved           |    |    |    |    |    |    |    |
| 23                 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved           |    |    |    |    |    |    |    |
| 15                 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| POR_DIS_CODE[15:8] |    |    |    |    |    |    |    |
| 7                  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| POR_DIS_CODE[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description  |   |
|---------|--------------|---|
| [31:16] | Reserved     | Reserved.   |
| [15:0]  | POR_DIS_CODE | <p><b>Power-on Reset Enable Control (Write Protect)</b></p> <p>When powered on, the POR circuit generates a reset signal to reset the whole chip function, but noise on the power may cause the POR active again. User can disable internal POR circuit to avoid unpredictable noise to cause chip reset by writing 0x5AA5 to this field.</p> <p>The POR function will be active again when this field is set to another value or chip is reset by other reset source, including:</p> <p>nRST, Watchdog reset, LVR reset, BOD reset, ICE reset command and the software-chip reset function.</p> <p><b>Note:</b> This bit is the protected bit, and programming it needs to write “59h”, “16h”, and “88h” to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100.</p> |

### Port 0 Multiple Function Pin Control Register (P0\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P0_MFP   | GCR_BA+0x30 | R/W | P0 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |    |    |    |             |    |    |    |
|--------------|----|----|----|-------------|----|----|----|
| 31           | 30 | 29 | 28 | 27          | 26 | 25 | 24 |
| Reserved     |    |    |    |             |    |    |    |
| 23           | 22 | 21 | 20 | 19          | 18 | 17 | 16 |
| P0_TYPE[7:0] |    |    |    |             |    |    |    |
| 15           | 14 | 13 | 12 | 11          | 10 | 9  | 8  |
| Reserved     |    |    |    | P0_ALT[3:0] |    |    |    |
| 7            | 6  | 5  | 4  | 3           | 2  | 1  | 0  |
| P0_MFP[7:0]  |    |    |    |             |    |    |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | P0_TYPE[m]  | <b>Port 0 Schmitt Trigger Input Enable</b><br>1 = Port 0 bit m Schmitt trigger input function Enabled.<br>0 = Port 0 bit m Schmitt trigger input function Disabled. |
| [15:12]            | -           | Reserved.   |
| [11]               | P0_ALT[3]   | <b>P0.3 Alternative Function</b><br>See P0_MFP[3].  |
| [10]               | P0_ALT[2]   | <b>P0.2 Alternative Function</b><br>See P0_MFP[2].  |
| [9]                | P0_ALT[1]   | <b>P0.1 Alternative Function</b><br>See P0_MFP[1].  |
| [8]                | P0_ALT[0]   | <b>P0.0 Alternative Function</b><br>See P0_MFP[0].  |
| [7]                | P0_MFP[7]   | <b>P0.7 Multi-function Selection</b><br>1 = The STADC function is selected.<br>0 = The GPIO P0.7 is selected.   |
| [6]                | P0_MFP[6]   | <b>P0.6 Multi-function Selection</b><br>1 = The BKP01 function is selected.<br>0 = The GPIO P0.6 is selected.   |
| [5]                | P0_MFP[5]   | <b>P0.5 Multi-function Selection</b><br>1 = The EPWM0.5 function is selected.<br>0 = The GPIO P0.5 is selected.   |

| Bits      | Description      |   |           |           |               |   |   |           |   |   |         |   |   |          |
|-----------|------------------|---|-----------|-----------|---------------|---|---|-----------|---|---|---------|---|---|----------|
| [4]       | <b>P0_MFP[4]</b> | <b>P0.4 Multi-function Selection</b><br>1 = The <b>EPWM0.4</b> function is selected.<br>0 = The <b>GPIO P0.4</b> is selected.   |           |           |               |   |   |           |   |   |         |   |   |          |
| [3]       | <b>P0_MFP[3]</b> | <b>P0.3 Multi-function Selection</b><br>This bit combined with <b>P0_ALT[3]</b> selects P0.3 multi-function. <table border="1"> <thead> <tr> <th>P0_ALT[3]</th><th>P0_MFP[3]</th><th>P0.3 Function</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>GPIO P0.3</td></tr> <tr> <td>0</td><td>1</td><td>EPWM0.3</td></tr> <tr> <td>1</td><td>1</td><td>STADC</td></tr> </tbody> </table>    | P0_ALT[3] | P0_MFP[3] | P0.3 Function | 0 | 0 | GPIO P0.3 | 0 | 1 | EPWM0.3 | 1 | 1 | STADC    |
| P0_ALT[3] | P0_MFP[3]        | P0.3 Function   |           |           |               |   |   |           |   |   |         |   |   |          |
| 0         | 0                | GPIO P0.3   |           |           |               |   |   |           |   |   |         |   |   |          |
| 0         | 1                | EPWM0.3   |           |           |               |   |   |           |   |   |         |   |   |          |
| 1         | 1                | STADC   |           |           |               |   |   |           |   |   |         |   |   |          |
| [2]       | <b>P0_MFP[2]</b> | <b>P0.2 Multi-function Selection</b><br>This bit combined with <b>P0_ALT[2]</b> selects P0.2 multi-function. <table border="1"> <thead> <tr> <th>P0_ALT[2]</th><th>P0_MFP[2]</th><th>P0.2 Function</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>GPIO P0.2</td></tr> <tr> <td>0</td><td>1</td><td>EPWM0.2</td></tr> <tr> <td>1</td><td>1</td><td>Reserved</td></tr> </tbody> </table> | P0_ALT[2] | P0_MFP[2] | P0.2 Function | 0 | 0 | GPIO P0.2 | 0 | 1 | EPWM0.2 | 1 | 1 | Reserved |
| P0_ALT[2] | P0_MFP[2]        | P0.2 Function   |           |           |               |   |   |           |   |   |         |   |   |          |
| 0         | 0                | GPIO P0.2   |           |           |               |   |   |           |   |   |         |   |   |          |
| 0         | 1                | EPWM0.2   |           |           |               |   |   |           |   |   |         |   |   |          |
| 1         | 1                | Reserved  |           |           |               |   |   |           |   |   |         |   |   |          |
| [1]       | <b>P0_MFP[1]</b> | <b>P0.1 Multi-function Selection</b><br>This bit combined with <b>P0_ALT[1]</b> selects P0.1 multi-function. <table border="1"> <thead> <tr> <th>P0_ALT[1]</th><th>P0_MFP[1]</th><th>P0.1 Function</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>GPIO P0.1</td></tr> <tr> <td>0</td><td>1</td><td>EPWM0.1</td></tr> <tr> <td>1</td><td>1</td><td>Reserved</td></tr> </tbody> </table> | P0_ALT[1] | P0_MFP[1] | P0.1 Function | 0 | 0 | GPIO P0.1 | 0 | 1 | EPWM0.1 | 1 | 1 | Reserved |
| P0_ALT[1] | P0_MFP[1]        | P0.1 Function   |           |           |               |   |   |           |   |   |         |   |   |          |
| 0         | 0                | GPIO P0.1   |           |           |               |   |   |           |   |   |         |   |   |          |
| 0         | 1                | EPWM0.1   |           |           |               |   |   |           |   |   |         |   |   |          |
| 1         | 1                | Reserved  |           |           |               |   |   |           |   |   |         |   |   |          |
| [0]       | <b>P0_MFP[0]</b> | <b>P0.0 Multi-function Selection</b><br>This bit combined with <b>P0_ALT[0]</b> selects P0.0 multi-function. <table border="1"> <thead> <tr> <th>P0_ALT[0]</th><th>P0_MFP[0]</th><th>P0.0 Function</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>GPIO P0.0</td></tr> <tr> <td>0</td><td>1</td><td>EPWM0.0</td></tr> <tr> <td>1</td><td>1</td><td>Reserved</td></tr> </tbody> </table> | P0_ALT[0] | P0_MFP[0] | P0.0 Function | 0 | 0 | GPIO P0.0 | 0 | 1 | EPWM0.0 | 1 | 1 | Reserved |
| P0_ALT[0] | P0_MFP[0]        | P0.0 Function   |           |           |               |   |   |           |   |   |         |   |   |          |
| 0         | 0                | GPIO P0.0   |           |           |               |   |   |           |   |   |         |   |   |          |
| 0         | 1                | EPWM0.0   |           |           |               |   |   |           |   |   |         |   |   |          |
| 1         | 1                | Reserved  |           |           |               |   |   |           |   |   |         |   |   |          |

### Port 1 Multiple Function Pin Control Register (P1\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P1_MFP   | GCR_BA+0x34 | R/W | P1 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| P1_TYPE[7:0] |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved     |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| P1_MFP[7:0]  |    |    |    |    |    |    |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | P1_TYPE[m]  | <b>Port 1 Schmitt Trigger Input Enable</b><br>1 = Port 1 bit m Schmitt trigger input function Enabled.<br>0 = Port 1 bit m Schmitt trigger input function Disabled. |
| [15:8]             | -           | Reserved.   |
| [7]                | P1_MFP[7]   | <b>P1.7 Multi-function Selection</b><br>1 = The BKP10 function is selected.<br>0 = The GPIO P1.7 is selected.   |
| [6]                | P1_MFP[6]   | <b>P1.6 Multi-function Selection</b><br>1 = The BKP00 function is selected.<br>0 = The GPIO P1.6 is selected.   |
| [5]                | P1_MFP[5]   | <b>P1.5 Multi-function Selection</b><br>1 = The EPWM1.5 function is selected.<br>0 = The GPIO P1.5 is selected.   |
| [4]                | P1_MFP[4]   | <b>P1.4 Multi-function Selection</b><br>1 = The EPWM1.4 function is selected.<br>0 = The GPIO P1.4 is selected.   |
| [3]                | P1_MFP[3]   | <b>P1.3 Multi-function Selection</b><br>1 = The EPWM1.3 function is selected.<br>0 = The GPIO P1.3 is selected.   |
| [2]                | P1_MFP[2]   | <b>P1.2 Multi-function Selection</b><br>1 = The EPWM1.2 function is selected.<br>0 = The GPIO P1.2 is selected.   |

| Bits | Description      |  |
|------|------------------|--|
| [1]  | <b>P1_MFP[1]</b> | <b>P1.1 Multi-function Selection</b><br>1 = The <b>EPWM1.1</b> function is selected.<br>0 = The GPIO P1.1 is selected. |
| [0]  | <b>P1_MFP[0]</b> | <b>P1.0 Multi-function Selection</b><br>1 = The <b>EPWM1.0</b> function is selected.<br>0 = The GPIO P1.0 is selected. |

### Port 2 Multiple Function Pin Control Register (P2\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P2_MFP   | GCR_BA+0x38 | R/W | P2 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |    |    |    |          |    |    |           |
|--------------|----|----|----|----------|----|----|-----------|
| 31           | 30 | 29 | 28 | 27       | 26 | 25 | 24        |
| Reserved     |    |    |    |          |    |    |           |
| 23           | 22 | 21 | 20 | 19       | 18 | 17 | 16        |
| P2_TYPE[7:0] |    |    |    |          |    |    |           |
| 15           | 14 | 13 | 12 | 11       | 10 | 9  | 8         |
| P2_ALT[7:4]  |    |    |    | Reserved |    |    | P2_ALT[0] |
| 7            | 6  | 5  | 4  | 3        | 2  | 1  | 0         |
| P2_MFP[7:0]  |    |    |    |          |    |    |           |

| Bits               | Description |   |           |               |
|--------------------|-------------|---|-----------|---------------|
| [31:24]            | -           | Reserved.   |           |               |
| [m+16]<br>m=0,1..7 | P2_TYPE[m]  | <b>Port 2 Schmitt Trigger Input Enable</b><br>1 = Port 2 bit m Schmitt trigger input function Enabled.<br>0 = Port 2 bit m Schmitt trigger input function Disabled. |           |               |
| [15]               | P2_ALT[7]   | <b>P2.7 Alternative Function</b><br>See P2_MFP[7].  |           |               |
| [14]               | P2_ALT[6]   | <b>P2.6 Alternative Function</b><br>See P2_MFP[6].  |           |               |
| [13]               | P2_ALT[5]   | <b>P2.5 Alternative Function</b><br>See P2_MFP[5].  |           |               |
| [12]               | P2_ALT[4]   | <b>P2.4 Alternative Function</b><br>See P2_MFP[4].  |           |               |
| [11:9]             | -           | Reserved.   |           |               |
| [8]                | P2_ALT[0]   | <b>P2.0 Alternative Function</b><br>See P2_MFP[0].  |           |               |
| [7]                | P2_MFP[7]   | <b>P2.7 Multi-function Selection</b><br>This bit combined with P2_ALT[7] selects P2.7 multi-function.   |           |               |
|                    |             | P2_ALT[7]   | P2_MFP[7] | P2.7 Function |
|                    |             | 0   | 0         | GPIO P2.7     |
|                    |             | 1   | 0         | SPI_CLK0      |
|                    |             | 1   | 1         | RTS1          |

| Bits | Description |  |           |
|------|-------------|--|-----------|
| [6]  | P2_MFP[6]   | <b>P2.6 Multi-function Selection</b><br>This bit combined with P2_ALT[6] selects P2.6 multi-function.        |           |
|      |             | P2_ALT[6]  | P2_MFP[6] |
|      |             | 0  | 0         |
|      |             | 0  | 1         |
|      |             | 1  | 0         |
| [5]  | P2_MFP[5]   | <b>P2.5 Multi-function Selection</b><br>This bit combined with P2_ALT[5] selects P2.5 multi-function.        |           |
|      |             | P2_ALT[5]  | P2_MFP[5] |
|      |             | 0  | 0         |
|      |             | 0  | 1         |
|      |             | 1  | 1         |
| [4]  | P2_MFP[4]   | <b>P2.4 Multi-function Selection</b><br>This bit combined with P2_ALT[4] selects P2.4 multi-function.        |           |
|      |             | P2_ALT[4]  | P2_MFP[4] |
|      |             | 0  | 0         |
|      |             | 0  | 1         |
|      |             | 1  | 1         |
| [3]  | P2_MFP[3]   | <b>P2.3 Multi-function Selection</b><br>1 = The IC00 function is selected.<br>0 = The GPIO P2.3 is selected. |           |
| [2]  | P2_MFP[2]   | <b>P2.2 Multi-function Selection</b><br>1 = The IC01 function is selected.<br>0 = The GPIO P2.2 is selected. |           |
| [1]  | P2_MFP[1]   | <b>P2.1 Multi-function Selection</b><br>1 = The IC02 function is selected.<br>0 = The GPIO P2.1 is selected. |           |
| [0]  | P2_MFP[0]   | <b>P2.0 Multi-function Selection</b><br>This bit combined with P2_ALT[0] selects P2.0 multi-function.        |           |
|      |             | P2_ALT[0]  | P2_MFP[0] |
|      |             | 0  | 0         |
|      |             | 0  | 1         |
|      |             | 1  | 0         |
|      |             | 1  | 1         |



### Port 3 Multiple Function Pin Control Register (P3\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P3_MFP   | GCR_BA+0x3C | R/W | P3 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |    |             |    |          |    |             |    |
|--------------|----|-------------|----|----------|----|-------------|----|
| 31           | 30 | 29          | 28 | 27       | 26 | 25          | 24 |
| Reserved     |    |             |    |          |    |             |    |
| 23           | 22 | 21          | 20 | 19       | 18 | 17          | 16 |
| P3_TYPE[5:0] |    |             |    |          |    |             |    |
| 15           | 14 | 13          | 12 | 11       | 10 | 9           | 8  |
| Reserved     |    | P3_ALT[5:4] |    | Reserved |    | P3_ALT[1:0] |    |
| 7            | 6  | 5           | 4  | 3        | 2  | 1           | 0  |
| P3_MFP[7:0]  |    |             |    |          |    |             |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | P3_TYPE[m]  | <b>Port 3 Schmitt Trigger Input Enable</b><br>1 = Port 3 bit m Schmitt trigger input function Enabled.<br>0 = Port 3 bit m Schmitt trigger input function Disabled. |
| [15:14]            | -           | Reserved.   |
| [13]               | P3_ALT[5]   | <b>P3.5 Alternative Function</b><br>See P3_MFP[5].  |
| [12]               | P3_ALT[4]   | <b>P3.4 Alternative Function</b><br>See P3_MFP[4].  |
| [11:10]            | -           | Reserved.   |
| [9]                | P3_ALT[1]   | <b>P3.1 Alternative Function</b><br>See P3_MFP[1].  |
| [8]                | P3_ALT[0]   | <b>P3.0 Alternative Function</b><br>See P3_MFP[0].  |
| [7]                | P3_MFP[7]   | <b>P3.7 Multi-function Selection</b><br>1 = The CANTX function is selected.<br>0 = The GPIO P3.7 is selected.   |
| [6]                | P3_MFP[6]   | <b>P3.6 Multi-function Selection</b><br>1 = The CANRX function is selected.<br>0 = The GPIO P3.6 is selected.   |

| Bits | Description |   |           |
|------|-------------|---|-----------|
| [5]  | P3_MFP[5]   | <b>P3.5 Multi-function Selection</b><br>This bit combined with P3_ALT[5] selects P3.5 multi-function.         |           |
|      |             | P3_ALT[5]   | P3_MFP[5] |
|      |             | 0   | 0         |
|      |             | 0   | 1         |
|      |             | 1   | 0         |
| [4]  | P3_MFP[4]   | <b>P3.4 Multi-function Selection</b><br>This bit combined with P3_ALT[4] selects P3.4 multi-function.         |           |
|      |             | P3_ALT[4]   | P3_MFP[4] |
|      |             | 0   | 0         |
|      |             | 0   | 1         |
|      |             | 1   | 0         |
| [3]  | P3_MFP[3]   | <b>P3.3 Multi-function Selection</b><br>1 = The /INT1 function is selected.<br>0 = The GPIO P3.3 is selected. |           |
|      |             |   |           |
|      |             |   |           |
|      |             |   |           |
|      |             |   |           |
| [2]  | P3_MFP[2]   | <b>P3.2 Multi-function Selection</b><br>1 = The /INT0 function is selected.<br>0 = The GPIO P3.2 is selected. |           |
|      |             |   |           |
|      |             |   |           |
|      |             |   |           |
|      |             |   |           |
| [1]  | P3_MFP[1]   | <b>P3.1 Multi-function Selection</b><br>This bit combined with P3_ALT[1] selects P3.1 multi-function.         |           |
|      |             | P3_ALT[1]   | P3_MFP[1] |
|      |             | 0   | 0         |
|      |             | 0   | 1         |
|      |             | 1   | 0         |
| [0]  | P3_MFP[0]   | <b>P3.0 Multi-function Selection</b><br>This bit combined with P3_ALT[0] selects P3.0 multi-function.         |           |
|      |             | P3_ALT[0]   | P3_MFP[0] |
|      |             | 0   | 0         |
|      |             | 0   | 1         |
|      |             | 1   | 0         |

### Port 4 Multiple Function Pin Control Register (P4\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P4_MFP   | GCR_BA+0x40 | R/W | P4 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |           |          |    |          |             |    |    |
|--------------|-----------|----------|----|----------|-------------|----|----|
| 31           | 30        | 29       | 28 | 27       | 26          | 25 | 24 |
| Reserved     |           |          |    |          |             |    |    |
| 23           | 22        | 21       | 20 | 19       | 18          | 17 | 16 |
| P4_TYPE[7:0] |           |          |    |          |             |    |    |
| 15           | 14        | 13       | 12 | 11       | 10          | 9  | 8  |
| Reserved     | P4_ALT[6] | Reserved |    |          |             |    |    |
| 7            | 6         | 5        | 4  | 3        | 2           | 1  | 0  |
| P4_MFP[7:4]  |           |          |    | Reserved | P4_MFP[2:0] |    |    |

| Bits               | Description |  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
|--------------------|-------------|--|--|-----------|-----------|---------------|---|---|-----------|---|---|----|---|---|------|---|---|---|
| [31:24]            | -           | Reserved.  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| [m+16]<br>m=0,1..7 | P4_TYPE[m]  | <b>Port 4 Schmitt Trigger Input Enable</b><br>1 = Port 4 bit m Schmitt trigger input function Enabled.<br>0 = Port 4 bit m Schmitt trigger input function Disabled.  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| [15]               | -           | Reserved.  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| [14]               | P4_ALT[6]   | <b>P4.6 Alternative Function.</b><br>See P4_MFP[6].  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| [13:8]             | -           | Reserved.  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| [7]                | P4_MFP[7]   | <b>P4.7 Multi-function Selection</b><br>1 = The T3 function is selected.<br>0 = The GPIO P4.7 is selected.   |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| [6]                | P4_MFP[6]   | <b>P4.6 Multi-function Selection</b><br>This bit combined with P4_ALT[6] selects P4.6 multi-function. <table><tr><th>P4_ALT[6]</th><th>P4_MFP[6]</th><th>P4.6 function</th></tr><tr><td>0</td><td>0</td><td>GPIO P4.6</td></tr><tr><td>0</td><td>1</td><td>T2</td></tr><tr><td>1</td><td>0</td><td>IDX1</td></tr><tr><td>1</td><td>1</td><td>-</td></tr></table> |  | P4_ALT[6] | P4_MFP[6] | P4.6 function | 0 | 0 | GPIO P4.6 | 0 | 1 | T2 | 1 | 0 | IDX1 | 1 | 1 | - |
| P4_ALT[6]          | P4_MFP[6]   | P4.6 function  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| 0                  | 0           | GPIO P4.6  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| 0                  | 1           | T2   |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| 1                  | 0           | IDX1   |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| 1                  | 1           | -  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |
| [5]                | P4_MFP[5]   | <b>P4.5 Multi-function Selection</b><br>1 = The QEIB1 function is selected.<br>0 = The GPIO P4.5 is selected.  |  |           |           |               |   |   |           |   |   |    |   |   |      |   |   |   |

| Bits | Description      |  |
|------|------------------|--|
| [4]  | <b>P4_MFP[4]</b> | <b>P4.4 Multi-function Selection</b><br>1 = The <b>QEIA1</b> function is selected.<br>0 = The GPIO P4.4 is selected. |
| [3]  | -                | <b>Reserved.</b>   |
| [2]  | <b>P4_MFP[2]</b> | <b>P4.2 Multi-function Selection</b><br>1 = The <b>IC12</b> function is selected.<br>0 = The GPIO P4.2 is selected.  |
| [1]  | <b>P4_MFP[1]</b> | <b>P4.1 Multi-function Selection</b><br>1 = The <b>IC11</b> function is selected.<br>0 = The GPIO P4.1 is selected.  |
| [0]  | <b>P4_MFP[0]</b> | <b>P4.0 Multi-function Selection</b><br>1 = The <b>IC10</b> function is selected.<br>0 = The GPIO P4.0 is selected.  |

### Port 5 Multiple Function Pin Control Register (P5\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P5_MFP   | GCR_BA+0x44 | R/W | P5 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |    |    |    |             |    |    |    |
|--------------|----|----|----|-------------|----|----|----|
| 31           | 30 | 29 | 28 | 27          | 26 | 25 | 24 |
| Reserved     |    |    |    |             |    |    |    |
| 23           | 22 | 21 | 20 | 19          | 18 | 17 | 16 |
| P5_TYPE[7:0] |    |    |    |             |    |    |    |
| 15           | 14 | 13 | 12 | 11          | 10 | 9  | 8  |
| Reserved     |    |    |    | P5_ALT[2:0] |    |    |    |
| 7            | 6  | 5  | 4  | 3           | 2  | 1  | 0  |
| P5_MFP[7:0]  |    |    |    |             |    |    |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | P5_TYPE[m]  | <b>Port 5 Schmitt Trigger Input Enable</b><br>1 = Port 5 bit m Schmitt trigger input function Enabled.<br>0 = Port 5 bit m Schmitt trigger input function Disabled. |
| [15:11]            | -           | Reserved.   |
| [10]               | P5_ALT[2]   | <b>P5.2 Alternative Function.</b><br>See P5_MFP[2].   |
| [9]                | P5_ALT[1]   | <b>P5.1 Alternative Function</b><br>See P5_MFP[1].  |
| [8]                | P5_ALT[0]   | <b>P5.0 Alternative Function</b><br>See P5_MFP[0].  |
| [7]                | P5_MFP[7]   | <b>P5.7 Multi-function Selection</b><br>1 = The BPWM1 function is selected.<br>0 = The GPIO P5.7 is selected.   |
| [6]                | P5_MFP[6]   | <b>P5.6 Multi-function Selection</b><br>1 = The BPWM0 function is selected.<br>0 = The GPIO P5.6 is selected.   |
| [5]                | P5_MFP[5]   | <b>P5.5 Multi-function Selection</b><br>1 = The CLKO function is selected.<br>0 = The GPIO P5.5 is selected.  |
| [4]                | P5_MFP[4]   | <b>P5.4 Multi-function Selection</b><br>1 = The /SS2 function is selected.<br>0 = The GPIO P5.4 is selected.  |

| Bits      | Description |  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
|-----------|-------------|--|-----------|-----------|---------------|---|---|-----------|---|---|-------|---|---|------|---|---|--------|
| [3]       | P5_MFP[3]   | <b>P5.3 Multi-function Selection</b><br>1 = The SPI_CLK2 function is selected.<br>0 = The GPIO P5.3 is selected.   |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| [2]       | P5_MFP[2]   | <b>P5.2 Multi-function Selection</b><br>This bit combined with P5_ALT[2] selects P5.2 multi-function. <table border="1"> <thead> <tr> <th>P5_ALT[2]</th><th>P5_MFP[2]</th><th>P5.2 Function</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>GPIO P5.2</td></tr> <tr> <td>0</td><td>1</td><td>MISO2</td></tr> <tr> <td>1</td><td>0</td><td>CPO1</td></tr> <tr> <td>1</td><td>1</td><td>-</td></tr> </tbody> </table>      | P5_ALT[2] | P5_MFP[2] | P5.2 Function | 0 | 0 | GPIO P5.2 | 0 | 1 | MISO2 | 1 | 0 | CPO1 | 1 | 1 | -      |
| P5_ALT[2] | P5_MFP[2]   | P5.2 Function  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 0         | 0           | GPIO P5.2  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 0         | 1           | MISO2  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 1         | 0           | CPO1   |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 1         | 1           | -  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| [1]       | P5_MFP[1]   | <b>P5.1 Multi-function Selection</b><br>This bit combined with P5_ALT[1] selects P5.1 multi-function. <table border="1"> <thead> <tr> <th>P5_ALT[1]</th><th>P5_MFP[1]</th><th>P5.1 Function</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>GPIO P5.1</td></tr> <tr> <td>0</td><td>1</td><td>MISO0</td></tr> <tr> <td>1</td><td>0</td><td>CTS0</td></tr> <tr> <td>1</td><td>1</td><td>I2CSDA</td></tr> </tbody> </table> | P5_ALT[1] | P5_MFP[1] | P5.1 Function | 0 | 0 | GPIO P5.1 | 0 | 1 | MISO0 | 1 | 0 | CTS0 | 1 | 1 | I2CSDA |
| P5_ALT[1] | P5_MFP[1]   | P5.1 Function  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 0         | 0           | GPIO P5.1  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 0         | 1           | MISO0  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 1         | 0           | CTS0   |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 1         | 1           | I2CSDA   |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| [0]       | P5_MFP[0]   | <b>P5.0 Multi-function Selection</b><br>This bit combined with P5_ALT[0] selects P5.0 multi-function. <table border="1"> <thead> <tr> <th>P5_ALT[0]</th><th>P5_MFP[0]</th><th>P5.0 Function</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>GPIO P5.0</td></tr> <tr> <td>0</td><td>1</td><td>MOSI0</td></tr> <tr> <td>1</td><td>0</td><td>RTS0</td></tr> <tr> <td>1</td><td>1</td><td>I2CSCL</td></tr> </tbody> </table> | P5_ALT[0] | P5_MFP[0] | P5.0 Function | 0 | 0 | GPIO P5.0 | 0 | 1 | MOSI0 | 1 | 0 | RTS0 | 1 | 1 | I2CSCL |
| P5_ALT[0] | P5_MFP[0]   | P5.0 Function  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 0         | 0           | GPIO P5.0  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 0         | 1           | MOSI0  |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 1         | 0           | RTS0   |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |
| 1         | 1           | I2CSCL   |           |           |               |   |   |           |   |   |       |   |   |      |   |   |        |

### Port 6 Multiple Function Pin Control Register (P6\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P6_MFP   | GCR_BA+0x48 | R/W | P6 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| P6_TYPE[7:0] |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved     |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| P6_MFP[7:0]  |    |    |    |    |    |    |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | P6_TYPE[m]  | <b>Port 6 Schmitt Trigger Input Enable</b><br>1 = Port 6 bit m Schmitt trigger input function Enabled.<br>0 = Port 6 bit m Schmitt trigger input function Disabled. |
| [15:8]             | -           | Reserved.   |
| [7]                | P6_MFP[7]   | <b>P6.7 Multi-function Selection</b><br>1 = The AINA7 function is selected.<br>0 = The GPIO P6.7 is selected.   |
| [6]                | P6_MFP[6]   | <b>P6.6 Multi-function Selection</b><br>1 = The AINA6 function is selected.<br>0 = The GPIO P6.6 is selected.   |
| [5]                | P6_MFP[5]   | <b>P6.5 Multi-function Selection</b><br>1 = The AINA5 or CPP1 function is selected.<br>0 = The GPIO P6.5 is selected.   |
| [4]                | P6_MFP[4]   | <b>P6.4 Multi-function Selection</b><br>1 = The AINA4 or CPN1 function is selected.<br>0 = The GPIO P6.4 is selected.   |
| [3]                | P6_MFP[3]   | <b>P6.3 Multi-function Selection</b><br>1 = The AINA3 function is selected.<br>0 = The GPIO P6.3 is selected.   |
| [2]                | P6_MFP[2]   | <b>P6.2 Multi-function Selection</b><br>1 = The AINA2 function is selected.<br>0 = The GPIO P6.2 is selected.   |

| Bits | Description      |  |
|------|------------------|--|
| [1]  | <b>P6_MFP[1]</b> | <b>P6.1 Multi-function Selection</b><br>1 = The <b>A1NA1</b> function is selected.<br>0 = The GPIO P6.1 is selected. |
| [0]  | <b>P6_MFP[0]</b> | <b>P6.0 Multi-function Selection</b><br>1 = The <b>A1NA0</b> function is selected.<br>0 = The GPIO P6.0 is selected. |



### Port 7 Multiple Function Pin Control Register (P7\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P7_MFP   | GCR_BA+0x4C | R/W | P7 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| P7_TYPE[7:0] |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved     |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| P7_MFP[7:0]  |    |    |    |    |    |    |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | P7_TYPE[m]  | <b>Port 7 Schmitt Trigger Input Enable</b><br>1 = Port 7 bit m Schmitt trigger input function Enabled.<br>0 = Port 7 bit m Schmitt trigger input function Disabled. |
| [15:8]             | -           | Reserved.   |
| [7]                | P7_MFP[7]   | <b>P7.7 Multi-function Selection</b><br>1 = The AINB7 function is selected.<br>0 = The GPIO P7.7 is selected.   |
| [6]                | P7_MFP[6]   | <b>P7.6 Multi-function Selection</b><br>1 = The AINB6 function is selected.<br>0 = The GPIO P7.6 is selected.   |
| [5]                | P7_MFP[5]   | <b>P7.5 Multi-function Selection</b><br>1 = The AINB5 or CPP2 function is selected.<br>0 = The GPIO P7.5 is selected.   |
| [4]                | P7_MFP[4]   | <b>P7.4 Multi-function Selection</b><br>1 = The AINB4 or CPN2 function is selected.<br>0 = The GPIO P7.4 is selected.   |
| [3]                | P7_MFP[3]   | <b>P7.3 Multi-function Selection</b><br>1 = The AINB3 function is selected.<br>0 = The GPIO P7.3 is selected.   |
| [2]                | P7_MFP[2]   | <b>P7.2 Multi-function Selection</b><br>1 = The AINB2 function is selected.<br>0 = The GPIO P7.2 is selected.   |

| Bits | Description      |  |
|------|------------------|--|
| [1]  | <b>P7_MFP[1]</b> | <b>P7.1 Multi-function Selection</b><br>1 = The <b>AINB1</b> function is selected.<br>0 = The GPIO P7.1 is selected. |
| [0]  | <b>P7_MFP[0]</b> | <b>P7.0 Multi-function Selection</b><br>1 = The <b>AINB0</b> function is selected.<br>0 = The GPIO P7.0 is selected. |

### Port 8 Multiple Function Pin Control Register (P8\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P8_MFP   | GCR_BA+0x50 | R/W | P8 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |          |    |             |    |    |    |    |
|--------------|----------|----|-------------|----|----|----|----|
| 31           | 30       | 29 | 28          | 27 | 26 | 25 | 24 |
| Reserved     |          |    |             |    |    |    |    |
| 23           | 22       | 21 | 20          | 19 | 18 | 17 | 16 |
| P8_TYPE[7:0] |          |    |             |    |    |    |    |
| 15           | 14       | 13 | 12          | 11 | 10 | 9  | 8  |
| Reserved     |          |    |             |    |    |    |    |
| 7            | 6        | 5  | 4           | 3  | 2  | 1  | 0  |
| P8_MFP[7]    | Reserved |    | P8_MFP[4:0] |    |    |    |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | P8_TYPE[m]  | <b>Port 8 Schmitt Trigger Input Enable</b><br>1 = Port 8 bit m Schmitt trigger input function Enabled.<br>0 = Port 8 bit m Schmitt trigger input function Disabled. |
| [15:8]             | -           | Reserved.   |
| [7]                | P8_MFP[7]   | <b>P8.7 Multi-function Selection</b><br>1 = The CPO function is selected.<br>0 = The GPIO P8.7 is selected.   |
| [6:5]              | -           | Reserved.   |
| [4]                | P8_MFP[4]   | <b>P8.4 Multi-function Selection</b><br>1 = The CPP function is selected.<br>0 = The GPIO P8.4 is selected.   |
| [3]                | P8_MFP[3]   | <b>P8.3 Multi-function Selection</b><br>1 = The CPN function is selected.<br>0 = The GPIO P8.3 is selected.   |
| [2]                | P8_MFP[2]   | <b>P8.2 Multi-function Selection</b><br>1 = The OPO0 function is selected.<br>0 = The GPIO P8.2 is selected.  |
| [1]                | P8_MFP[1]   | <b>P8.1 Multi-function Selection</b><br>1 = The OPN0 function is selected.<br>0 = The GPIO P8.1 is selected.  |

| Bits | Description      |   |
|------|------------------|---|
| [0]  | <b>P8_MFP[0]</b> | <b>P8.0 Multi-function Selection</b><br>1 = The <b>OPP0</b> function is selected.<br>0 = The GPIO P8.0 is selected. |

### Port 9 Multiple Function Pin Control Register (P9\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| P9_MFP   | GCR_BA+0x54 | R/W | P9 Multiple Function and Input Type Control Register | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| P9_TYPE[7:0] |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved     |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| P9_MFP[7:0]  |    |    |    |    |    |    |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | P9_TYPE[m]  | <b>Port 9 Schmitt Trigger Input Enable</b><br>1 = Port 9 bit m Schmitt trigger input function Enabled.<br>0 = Port 9 bit m Schmitt trigger input function Disabled. |
| [15:8]             | -           | Reserved.   |
| [7]                | P9_MFP[7]   | <b>P9.7 Multi-function Selection</b><br>1 = The /SS1 function is selected.<br>0 = The GPIO P9.7 is selected.  |
| [6]                | P9_MFP[6]   | <b>P9.6 Multi-function Selection</b><br>1 = The MOSI1 function is selected.<br>0 = The GPIO P9.6 is selected.   |
| [5]                | P9_MFP[5]   | <b>P9.5 Multi-function Selection</b><br>1 = The MISO1 function is selected.<br>0 = The GPIO P9.5 is selected.   |
| [4]                | P9_MFP[4]   | <b>P9.4 Multi-function Selection</b><br>1 = The SPI_CLK1 function is selected.<br>0 = The GPIO P9.4 is selected.  |
| [3]                | P9_MFP[3]   | <b>P9.3 Multi-function Selection</b><br>1 = The BKP11 function is selected.<br>0 = The GPIO P9.3 is selected.   |
| [2]                | P9_MFP[2]   | <b>P9.2 Multi-function Selection</b><br>1 = The OPP1 function is selected.<br>0 = The GPIO P9.2 is selected.  |

| Bits | Description      |   |
|------|------------------|---|
| [1]  | <b>P9_MFP[1]</b> | <b>P9.1 Multi-function Selection</b><br>1 = The <b>OPN1</b> function is selected.<br>0 = The GPIO P9.1 is selected. |
| [0]  | <b>P9_MFP[0]</b> | <b>P9.0 Multi-function Selection</b><br>1 = The <b>OPO1</b> function is selected.<br>0 = The GPIO P9.0 is selected. |

### Port A Multiple Function Pin Control Register (PA\_MFP)

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| PA_MFP   | GCR_BA+0x58 | R/W | PA Multiple Function and Input Type Control Register | 0x0000_0000 |

|          |    |    |    |    |    |              |    |
|----------|----|----|----|----|----|--------------|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25           | 24 |
| Reserved |    |    |    |    |    |              |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17           | 16 |
| Reserved |    |    |    |    |    | PA_TYPE[1:0] |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9            | 8  |
| Reserved |    |    |    |    |    | PA_ALT[1:0]  |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1            | 0  |
| Reserved |    |    |    |    |    | PA_MFP[1:0]  |    |

| Bits            | Description |  |           |               |
|-----------------|-------------|--|-----------|---------------|
| [31:18]         | -           | Reserved.  |           |               |
| [m+16]<br>m=0,1 | PA_TYPE[m]  | Port A Schmitt Trigger Input Enable<br>1 = Port A bit m Schmitt trigger input function Enabled.<br>0 = Port A bit m Schmitt trigger input function Disabled. |           |               |
| [15:10]         | -           | Reserved.  |           |               |
| [9]             | PA_ALT[1]   | PA.1 Alternative Function<br>See PA_MFP[1].  |           |               |
| [8]             | PA_ALT[0]   | PA.0 Alternative Function<br>See PA_MFP[0].  |           |               |
| [7:2]           |             | Reserved   |           |               |
| [1]             | PA_MFP[1]   | PA.1 Multi-function Selection<br>This bit combined with PA_ALT[1] selects PA.1 multi-function.   |           |               |
|                 |             | PA_ALT[1]  | PA_MFP[1] | PA.1 Function |
|                 |             | 0  | 0         | GPIO PA.1     |
|                 |             | 0  | 1         | RX1           |
|                 |             | 1  | 1         | I2CSCL        |
| [0]             | PA_MFP[0]   | PA.0 Multi-function Selection<br>This bit combined with PA_ALT[0] selects PA.0 multi-function.   |           |               |
|                 |             | PA_ALT[0]  | PA_MFP[0] | PA.0 Function |
|                 |             | 0  | 0         | GPIO PA.0     |
|                 |             | 0  | 1         | TX1           |
|                 |             | 1  | 1         | I2CSDA        |





### Register Write-Protection Control Register (REGWRPROT)

Some of the system control registers need to be protected to avoid inadvertent write and disturb the chip operation. These system control registers are protected after the power on reset till user to disable register protection. For user to program these protected registers, a register protection disable sequence needs to be followed by a special programming. The register protection disable sequence is writing the data “59h”, “16h” “88h” to the register REGWRPROT address at 0x5000\_0100 continuously. Any different data value, different sequence or any other write to other address during these three data writing will abort the whole sequence.

After the protection is disabled, user can check the protection disable bit at address 0x5000\_0100 bit0, “1” is protection disable, “0” is protection enable. Then user can update the target protected register value and then write any data to the address “0x5000\_0100” to enable register protection.

Write this register to disable/enable register protection, and reading it to get the REGPROTDIS status.

| Register  | Offset       | R/W | Description                                | Reset Value |
|-----------|--------------|-----|--|-------------|
| REGWRPROT | GCR_BA+0x100 | R/W | Register Write-Protection Control Register | 0x0000_0000 |

|                |    |    |    |    |    |    |            |
|----------------|----|----|----|----|----|----|------------|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24         |
| Reserved       |    |    |    |    |    |    |            |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16         |
| Reserved       |    |    |    |    |    |    |            |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8          |
| Reserved       |    |    |    |    |    |    |            |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0          |
| REGWRPROT[7:0] |    |    |    |    |    |    |            |
| -              |    |    |    |    |    |    | REGPROTDIS |

| Bits    | Description  |
|---------|--|
| [31:16] | - Reserved.  |
| [7:0]   | <b>Register Write-Protection Code (Write Only)</b><br>Some registers have write-protection function. Writing these registers have to disable the protected function by writing the sequence value “59h”, “16h”, “88h” to this field. After this sequence is completed, the REGPROTDIS bit will be set to 1 and write-protection registers can be normal write. |
| [0]     | <b>Register Write-Protection Disable index (Read Only)</b><br>0 = Write-protection Enabled for writing protected registers. Any write to the protected register is ignored.<br>1 = Write-protection Disabled for writing protected registers.<br><b>Note:</b> The bits which are write-protected will be noted as “(Write Protect)” beside the description.    |

## 7.6 System Timer (SysTick)

The Cortex™-M0 includes an integrated system timer, SysTick, which provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used as a Real Time Operating System (RTOS) tick timer or as a simple counter.

When system timer is enabled, it will count down from the value in the SysTick Current Value Register (SYST\_CVR) to 0, and reload (wrap) to the value in the SysTick Reload Value Register (SYST\_RVR) on the next clock cycle, then decrement on subsequent clocks. When the counter transitions to 0, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on reads.

The SYST\_CVR value is UNKNOWN on reset. Software should write to the register to clear it to 0 before enabling the feature. This ensures the timer will count from the SYST\_RVR value rather than an arbitrary value when it is enabled.

If the SYST\_RVR is 0, the timer will be maintained with a current value of 0 after it is reloaded with this value. This mechanism can be used to disable the feature independently from the timer enable bit.

For more detailed information, please refer to the “ARM® Cortex™-M0 Technical Reference Manual” and “ARM® v6-M Architecture Reference Manual”.

### 7.6.1 System Timer Control Register Map

R: read only, W: write only, R/W: both read and write

| Register                                    | Offset       | R/W | Description                         | Reset Value |
|---|--------------|-----|-------------------------------------|-------------|
| SYST Base Address:<br>SYST_BA = 0xE000_E010 |              |     |                                     |             |
| SYST_CSR                                    | SYST_BA+0x00 | R/W | SysTick Control and Status Register | 0x0000_0000 |
| SYST_RVR                                    | SYST_BA+0x04 | R/W | SysTick Reload Value Register       | 0xFFFF_FFFF |
| SYST_CVR                                    | SYST_BA+0x08 | R/W | SysTick Current Value Register      | 0xFFFF_FFFF |

### SysTick Control and Status (SYST\_CSR)

| Register | Offset       | R/W | Description                         | Reset Value |
|----------|--------------|-----|-------------------------------------|-------------|
| SYST_CSR | SYST_BA+0x00 | R/W | SysTick Control and Status Register | 0x0000_0000 |

|          |    |    |    |    |        |         |           |
|----------|----|----|----|----|--------|---------|-----------|
| 31       | 30 | 29 | 28 | 27 | 26     | 25      | 24        |
| Reserved |    |    |    |    |        |         |           |
| 23       | 22 | 21 | 20 | 19 | 18     | 17      | 16        |
| Reserved |    |    |    |    |        |         | COUNTFLAG |
| 15       | 14 | 13 | 12 | 11 | 10     | 9       | 8         |
| Reserved |    |    |    |    |        |         |           |
| 7        | 6  | 5  | 4  | 3  | 2      | 1       | 0         |
| Reserved |    |    |    |    | CLKSRC | TICKINT | ENABLE    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:17] | Reserved    | Reserved.  |
| [16]    | COUNTFLAG   | <b>SysTick flag.</b><br>COUNTFLAG is set when the counter transitions to zero.<br>COUNTFLAG is cleared by a read from this register or a write to the SYST_CVR.                          |
| [15:3]  | Reserved    | Reserved.  |
| [2]     | CLKSRC      | <b>System Tick Clock Source Selection</b><br>0 = Clock source is optional, refer to STCLK_S (CLKSEL0[5:3]).<br>1 = Core clock used for SysTick timer.                                    |
| [1]     | TICKINT     | <b>Enables SysTick exception request.</b><br>0 = Counting down to zero does not assert the SysTick exception request<br>1 = Counting down to zero asserts the SysTick exception request. |
| [0]     | ENABLE      | <b>Enables SysTick counter.</b><br>0 = SysTick counter Disabled<br>1 = SysTick counter Enabled.  |

### SysTick Reload Value Register (SYST\_RVR)

| Register | Offset       | R/W | Description                   | Reset Value |
|----------|--------------|-----|-------------------------------|-------------|
| SYST_RVR | SYST_BA+0x04 | R/W | SysTick Reload Value Register | 0xFFFF_XXXX |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved      |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| RELOAD[23:16] |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| RELOAD[15:8]  |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| RELOAD[7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |
|---------|-------------|
| [31:24] | Reserved    |
| [23:0]  | RELOAD      |

|   |  |
|---|--|
| Reserved.   |  |
| System Tick Reload Value  |  |
| Value to load into the Current Value register when the counter reaches 0. |  |

### SysTick Current Value Register (SYST\_CVR)

| Register | Offset       | R/W | Description                    | Reset Value |
|----------|--------------|-----|--------------------------------|-------------|
| SYST_CVR | SYST_BA+0x08 | R/W | SysTick Current Value Register | 0xFFFF_XXXX |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved       |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CURRENT[23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CURRENT[15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CURRENT[7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:24] | Reserved    | Reserved.  |
| [23:0]  | CURRENT     | <b>System Tick Current Value</b><br>Current counter value. This is the value of the counter at the time it is sampled. The counter does not provide read-modify-write protection. The register is write-clear. A software write of any value will clear the register to 0. Unsupported bits RAZ (see SysTick Reload Value register). |

## 7.7 Nested Vectored Interrupt Controller (NVIC)

The Cortex™-M0 provides an interrupt controller as an integral part of the exception mode, named as “Nested Vectored Interrupt Controller (NVIC)”, which is closely coupled to the processor core and provides following features:

- Nested and Vectored interrupt support
- Automatic processor state saving and restoration
- Reduced and deterministic interrupt latency

The NVIC prioritizes and handles all supported exceptions. All exceptions are handled in “Handler Mode”. This NVIC architecture supports 32 (IRQ[31:0]) discrete interrupts with 4 levels of priority. All of the interrupts and most of the system exceptions can be configured to different priority levels. When an interrupt occurs, the NVIC will compare the priority of the new interrupt to the current running one’s priority. If the priority of the new interrupt is higher than the current one, the new interrupt handler will override the current handler.

When an interrupt is accepted, the starting address of the interrupt service routine (ISR) is fetched from a vector table in memory. There is no need to determine which interrupt is accepted and branch to the starting address of the correlated ISR by software. While the starting address is fetched, NVIC will also automatically save processor state including the registers “PC, PSR, LR, R0~R3, R12” to the stack. At the end of the ISR, the NVIC will restore the mentioned registers from stack and resume the normal execution. Thus it will take less and deterministic time to process the interrupt request.

The NVIC supports “Tail Chaining” which handles back-to-back interrupts efficiently without the overhead of states saving and restoration and therefore reduces delay time in switching to pending ISR at the end of current ISR. The NVIC also supports “Late Arrival” which improves the efficiency of concurrent ISRs. When a higher priority interrupt request occurs before the current ISR starts to execute (at the stage of state saving and starting address fetching), the NVIC will give priority to the higher one without delay penalty. Thus it advances the real-time capability.

For more detailed information, please refer to the “ARM® Cortex™-M0 Technical Reference Manual” and “ARM® v6-M Architecture Reference Manual”.

### 7.7.1 Exception Model and System Interrupt Map

Table 7-2 lists the exception model supported by the NuMicro™ NM15xx Series. Software can set four levels of priority on some of these exceptions as well as on all interrupts. The highest user-configurable priority is denoted as “0” and the lowest priority is denoted as “3”. The default priority of all the user-configurable interrupts is “0”. Note that priority “0” is treated as the fourth priority on the system, after three system exceptions “Reset”, “NMI” and “Hard Fault”.

| Exception Name | Vector Number | Priority     |
|----------------|---------------|--------------|
| Reset          | 1             | -3           |
| NMI            | 2             | -2           |
| Hard Fault     | 3             | -1           |
| Reserved       | 4 ~ 10        | Reserved     |
| SVCall         | 11            | Configurable |
| Reserved       | 12 ~ 13       | Reserved     |
| PendSV         | 14            | Configurable |

| Exception Name           | Vector Number | Priority     |
|--------------------------|---------------|--------------|
| SysTick                  | 15            | Configurable |
| Interrupt (IRQ0 ~ IRQ31) | 16 ~ 47       | Configurable |

Table 7-2 Exception Model

| Exception Number | Vector Address | Interrupt Number (Bit in Interrupt Registers) | Interrupt Name   | Source Module              | Exception description   | Power Down wake-up |
|------------------|----------------|---|------------------|----------------------------|---|--------------------|
| 1 ~ 15           |                | -   | -                | -                          | System exceptions   | -                  |
| 16               | 0x40           | 0   | <b>BOD_INT</b>   | Brown-Out                  | Brown-Out low voltage detected interrupt                          | Yes                |
| 17               | 0x44           | 1   | <b>WDT_INT</b>   | WDT                        | Watchdog Timer interrupt  | Yes                |
| 18               | 0x48           | 2   | <b>EINT0_INT</b> | P3.2                       | External signal interrupt from P3.2 pin                           | Yes                |
| 19               | 0x4C           | 3   | <b>EINT1_INT</b> | P3.3                       | External signal interrupt from P3.3 pin                           | Yes                |
| 20               | 0x50           | 4   | <b>GPG0_INT</b>  | P0~P4 except P3.2 and P3.3 | External interrupt from GPIO group 0 (P0~P4) except P3.2 and P3.3 | Yes                |
| 21               | 0x54           | 5   | <b>GPG1_INT</b>  | P5~PA                      | External interrupt from GPIO group 1 (P5~PA)                      | Yes                |
| 22               | 0x58           | 6   | <b>BPWM_INT</b>  | BPWM0/1                    | Basic PWM0 and PWM1 interrupt                                     | No                 |
| 23               | 0x5C           | 7   | <b>ADC0_INT</b>  | ADC0                       | ADC0 interrupt  | No                 |
| 24               | 0x60           | 8   | <b>TMR0_INT</b>  | TMR0                       | Timer 0 interrupt   | No                 |
| 25               | 0x64           | 9   | <b>TMR1_INT</b>  | TMR1                       | Timer 1 interrupt   | No                 |
| 26               | 0x68           | 10  | <b>TMR2_INT</b>  | TMR2                       | Timer 2 interrupt   | No                 |
| 27               | 0x6C           | 11  | <b>TMR3_INT</b>  | TMR3                       | Timer 3 interrupt   | No                 |
| 28               | 0x70           | 12  | <b>UART0_INT</b> | UART0                      | UART0 interrupt   | Yes                |
| 29               | 0x74           | 13  | <b>UART1_INT</b> | UART1                      | UART1 interrupt   | Yes                |
| 30               | 0x78           | 14  | <b>SPI0_INT</b>  | SPI0                       | SPI0 interrupt  | No                 |
| 31               | 0x7C           | 15  | <b>SPI1_INT</b>  | SPI1                       | SPI1 interrupt  | No                 |
| 32               | 0x80           | 16  | <b>SPI2_INT</b>  | SPI2                       | SPI2 interrupt  | No                 |
| 33               | 0x84           | 17  | <b>MDU_INT</b>   | MDU                        | Motor drive unit interrupt  | No                 |
| 34               | 0x88           | 18  | <b>I2C_INT</b>   | I <sup>2</sup> C           | I <sup>2</sup> C interrupt  | Yes                |
| 35               | 0x8C           | 19  | <b>CKD_INT</b>   | CKD                        | CKD interrupt   | No                 |
| 36               | 0x90           | 20  | <b>CAN_INT</b>   | CAN                        | CAN interrupt   | No                 |

| Exception Number | Vector Address | Interrupt Number (Bit in Interrupt Registers) | Interrupt Name | Source Module | Exception description  | Power Down wake-up               |
|------------------|----------------|---|----------------|---------------|--|----------------------------------|
| 37               | 0x94           | 21  | EPWM0_INT      | EPWM0         | Enhanced PWM0 interrupt  | No                               |
| 38               | 0x98           | 22  | EPWM1_INT      | EPWM1         | Enhanced PWM1 interrupt  | No                               |
| 39               | 0x9C           | 23  | CAP0_INT       | CAP0          | Input capture 0 interrupt  | No                               |
| 40               | 0xA0           | 24  | CAP1_INT       | CAP1          | Input capture 1 interrupt  | No                               |
| 41               | 0xA4           | 25  | ACMP_INT       | ACMP          | Analog Comparator 0 or 1, or OP Amplifier digital output interrupt | Yes (only by analog comparator ) |
| 42               | 0xA8           | 26  | QEI0_INT       | QEI0          | QEI0 interrupt   | No                               |
| 43               | 0xAC           | 27  | QEI1_INT       | QEI1          | QEI1 interrupt   | No                               |
| 44               | 0xB0           | 28  | PWRWU_INT      | CLKC          | Clock controller interrupt for chip wake up from power-down state  | -                                |
| 45               | 0xB4           | 29  | ADC1_INT       | ADC1          | ADC1 interrupt   | No                               |
| 46               | 0xB8           | 30  | ADC2_INT       | ADC2          | ADC2 interrupt   | No                               |
| 47               | 0xBC           | 31  | ADC3_INT       | ADC3          | ADC3 interrupt   | No                               |

Table 7-3 System Interrupt Map Vector Table

### 7.7.2 Vector Table

When an interrupt is accepted, the processor will automatically fetch the starting address of the interrupt service routine (ISR) from a vector table in memory. For ARMv6-M, the vector table base address is fixed at 0x00000000. The vector table contains the initialization value for the stack pointer on reset, and the entry point addresses for all exception handlers. The vector number on previous page defines the order of entries in the vector table associated with exception handler entry as illustrated in previous section.

| Vector Table Word Offset (Bytes) | Description                                      |
|----------------------------------|--|
| 0                                | SP_main – The Main stack pointer                 |
| Vector Number                    | Exception Entry Pointer using that Vector Number |

Table 7-4 Vector Table

### 7.7.3 Operation Description

NVIC interrupts can be enabled and disabled by writing to their corresponding Interrupt Set-Enable or Interrupt Clear-Enable register bit-field. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current enabled state of the corresponding interrupts. When an interrupt is disabled, interrupt assertion will cause the interrupt to become Pending, however, the interrupt will not be activated. If an interrupt is Active when it is disabled, it remains in its Active state until cleared by reset or an exception return. Clearing the enable bit prevents new activations of the associated interrupt.

NVIC interrupts can be pending/un-pending using a complementary pair of registers to those used



to enable/disable the interrupts, named the Set-Pending Register and Clear-Pending Register respectively. The registers use a write-1-to-enable and write-1-to-clear policy, both registers reading back the current pended state of the corresponding interrupts. The Clear-Pending Register has no effect on the execution status of an Active interrupt.

NVIC interrupts are prioritized by updating an 8-bit field within a 32-bit register (each register supporting four interrupts).

The general registers associated with the NVIC are all accessible from a block of memory in the System Control Space and will be described in next section.

### 7.7.4 NVIC Control Registers

R: read only, W: write only, R/W: both read and write

| Register                     | Offset        | R/W | Description                                       | Reset Value |
|------------------------------|---------------|-----|---|-------------|
| <b>NVIC Base Address:</b>    |               |     |   |             |
| <b>NVIC_BA = 0xE000_E100</b> |               |     |   |             |
| <b>NVIC_ISER</b>             | NVIC_BA+0x000 | R/W | IRQ0 ~ IRQ31 Set-Enable Control Register          | 0x0000_0000 |
| <b>NVIC_ICER</b>             | NVIC_BA+0x080 | R/W | IRQ0 ~ IRQ31 Clear-Enable Control Register        | 0x0000_0000 |
| <b>NVIC_ISPR</b>             | NVIC_BA+0x100 | R/W | IRQ0 ~ IRQ31 Set-Pending Control Register         | 0x0000_0000 |
| <b>NVIC_ICPR</b>             | NVIC_BA+0x180 | R/W | IRQ0 ~ IRQ31 Clear-Pending Control Register       | 0x0000_0000 |
| <b>NVIC_IPR0</b>             | NVIC_BA+0x300 | R/W | IRQ0 ~ IRQ3 Interrupt Priority Control Register   | 0x0000_0000 |
| <b>NVIC_IPR1</b>             | NVIC_BA+0x304 | R/W | IRQ4 ~ IRQ7 Interrupt Priority Control Register   | 0x0000_0000 |
| <b>NVIC_IPR2</b>             | NVIC_BA+0x308 | R/W | IRQ8 ~ IRQ11 Interrupt Priority Control Register  | 0x0000_0000 |
| <b>NVIC_IPR3</b>             | NVIC_BA+0x30C | R/W | IRQ12 ~ IRQ15 Interrupt Priority Control Register | 0x0000_0000 |
| <b>NVIC_IPR4</b>             | NVIC_BA+0x310 | R/W | IRQ16 ~ IRQ19 Interrupt Priority Control Register | 0x0000_0000 |
| <b>NVIC_IPR5</b>             | NVIC_BA+0x314 | R/W | IRQ20 ~ IRQ23 Interrupt Priority Control Register | 0x0000_0000 |
| <b>NVIC_IPR6</b>             | NVIC_BA+0x318 | R/W | IRQ24 ~ IRQ27 Interrupt Priority Control Register | 0x0000_0000 |
| <b>NVIC_IPR7</b>             | NVIC_BA+0x31C | R/W | IRQ28 ~ IRQ31 Interrupt Priority Control Register | 0x0000_0000 |

### IRQ0 ~ IRQ31 Set-Enable Control Register (NVIC\_ISER)

| Register  | Offset        | R/W | Description                              | Reset Value |
|-----------|---------------|-----|--|-------------|
| NVIC_ISER | NVIC_BA+0x000 | R/W | IRQ0 ~ IRQ31 Set-Enable Control Register | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| SETENA[31:24] |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| SETENA[23:16] |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| SETENA[15:8]  |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| SETENA[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [31:0] | <p><b>SETENA</b></p> <p><b>Interrupt Enable</b></p> <p>The ISER enables interrupts, and shows the interrupts that are enabled. Each bit represents an IRQ number from IRQ0 ~ IRQ31 (Exception number from 16 ~ 47).</p> <p>Write:</p> <p>0 = No effect.</p> <p>1 = Interrupt Enabled.</p> <p>Read:</p> <p>0 = Interrupt Disabled.</p> <p>1 = Interrupt Enabled.</p> |

**IRQ0 ~ IRQ31 Clear-Enable Control Register (NVIC\_ICER)**

| Register  | Offset        | R/W | Description                                | Reset Value |
|-----------|---------------|-----|--|-------------|
| NVIC_ICER | NVIC_BA+0x080 | R/W | IRQ0 ~ IRQ31 Clear-Enable Control Register | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| CLRENA[31:24] |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CLRENA[23:16] |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CLRENA[15:8]  |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CLRENA[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [31:0] | <p><b>CLRENA</b></p> <p><b>Interrupt Clear Enable</b></p> <p>The ICER disables interrupts, and shows the interrupts that are enabled. Each bit represents an IRQ number from IRQ0 ~ IRQ31 (Exception number from 16 ~ 47).</p> <p>Write:</p> <p>0 = No effect.</p> <p>1 = Interrupt Disabled.</p> <p>Read:</p> <p>0 = Interrupt Disabled.</p> <p>1 = Interrupt Enabled.</p> |

### IRQ0 ~ IRQ31 Set-Pending Control Register (NVIC\_ISPR)

| Register  | Offset        | R/W | Description                               | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_ISPR | NVIC_BA+0x100 | R/W | IRQ0 ~ IRQ31 Set-Pending Control Register | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| SETPEND[31:24] |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| SETPEND[23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| SETPEND[15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| SETPEND[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description  |
|--------|--|
| [31:0] | <p><b>SETPEND</b></p> <p><b>Interrupt set-pending.</b></p> <p>The ISPR forces interrupts into the pending state, and shows the interrupts that are pending. Each bit represents an IRQ number from IRQ0 ~ IRQ31 (Exception number from 16 ~ 47).</p> <p>Write:</p> <p>0 = no effect.</p> <p>1 = changes interrupt state to pending.</p> <p>Read:</p> <p>0 = interrupt is not pending.</p> <p>1 = interrupt is pending.</p> |

**IRQ0 ~ IRQ31 Clear-Pending Control Register (NVIC\_ICPR)**

| Register  | Offset        | R/W | Description                                 | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_ICPR | NVIC_BA+0x180 | R/W | IRQ0 ~ IRQ31 Clear-Pending Control Register | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| CLRPEND[31:24] |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CLRPEND[23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CLRPEND[15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CLRPEND[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [31:0] | <p><b>CLRPEND</b></p> <p><b>Interrupt clear-pending.</b></p> <p>The ICPR removes the pending state from interrupts, and shows the interrupts that are pending. Each bit represents an IRQ number from IRQ0 ~ IRQ31 (Exception number from 16 ~ 47).</p> <p>Write:</p> <p>0 = no effect.</p> <p>1 = removes pending state an interrupt.</p> <p>Read:</p> <p>0 = interrupt is not pending.</p> <p>1 = interrupt is pending.</p> |

### IRQ0 ~ IRQ3 Interrupt Priority Register (NVIC\_IPR0)

| Register  | Offset        | R/W | Description                                     | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_IPR0 | NVIC_BA+0x300 | R/W | IRQ0 ~ IRQ3 Interrupt Priority Control Register | 0x0000_0000 |

|       |    |          |    |    |    |    |    |
|-------|----|----------|----|----|----|----|----|
| 31    | 30 | 29       | 28 | 27 | 26 | 25 | 24 |
| PRI_3 |    | Reserved |    |    |    |    |    |
| 23    | 22 | 21       | 20 | 19 | 18 | 17 | 16 |
| PRI_2 |    | Reserved |    |    |    |    |    |
| 15    | 14 | 13       | 12 | 11 | 10 | 9  | 8  |
| PRI_1 |    | Reserved |    |    |    |    |    |
| 7     | 6  | 5        | 4  | 3  | 2  | 1  | 0  |
| PRI_0 |    | Reserved |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:30] | PRI_3       | <b>Priority of IRQ3</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [29:24] | Reserved    | Reserved  |
| [23:22] | PRI_2       | <b>Priority of IRQ2</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [21:16] | Reserved    | Reserved  |
| [15:14] | PRI_1       | <b>Priority of IRQ1</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [13:8]  | Reserved    | Reserved  |
| [7:6]   | PRI_0       | <b>Priority of IRQ0</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [5:0]   | Reserved    | Reserved  |

### IRQ4 ~ IRQ7 Interrupt Priority Register (NVIC IPR1)

| Register  | Offset        | R/W | Description                                     | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_IPR1 | NVIC_BA+0x304 | R/W | IRQ4 ~ IRQ7 Interrupt Priority Control Register | 0x0000_0000 |

| 31    | 30 | 29       | 28 | 27 | 26 | 25 | 24 |
|-------|----|----------|----|----|----|----|----|
| PRI_7 |    | Reserved |    |    |    |    |    |
| 23    | 22 | 21       | 20 | 19 | 18 | 17 | 16 |
| PRI_6 |    | Reserved |    |    |    |    |    |
| 15    | 14 | 13       | 12 | 11 | 10 | 9  | 8  |
| PRI_5 |    | Reserved |    |    |    |    |    |
| 7     | 6  | 5        | 4  | 3  | 2  | 1  | 0  |
| PRI_4 |    | Reserved |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:30] | PRI_7       | <b>Priority of IRQ7</b><br>“0” denotes the highest priority and “3” denotes lowest priority |
| [29:24] | Reserved    | Reserved  |
| [23:22] | PRI_6       | <b>Priority of IRQ6</b><br>“0” denotes the highest priority and “3” denotes lowest priority |
| [21:16] | Reserved    | Reserved  |
| [15:14] | PRI_5       | <b>Priority of IRQ5</b><br>“0” denotes the highest priority and “3” denotes lowest priority |
| [13:8]  | Reserved    | Reserved  |
| [7:6]   | PRI_4       | <b>Priority of IRQ4</b><br>“0” denotes the highest priority and “3” denotes lowest priority |
| [5:0]   | Reserved    | Reserved  |

### IRQ8 ~ IRQ11 Interrupt Priority Register (NVIC\_IPR2)

| Register  | Offset        | R/W | Description                                      | Reset Value |
|-----------|---------------|-----|--|-------------|
| NVIC_IPR2 | NVIC_BA+0x308 | R/W | IRQ8 ~ IRQ11 Interrupt Priority Control Register | 0x0000_0000 |

|        |    |          |    |    |    |    |    |
|--------|----|----------|----|----|----|----|----|
| 31     | 30 | 29       | 28 | 27 | 26 | 25 | 24 |
| PRI_11 |    | Reserved |    |    |    |    |    |
| 23     | 22 | 21       | 20 | 19 | 18 | 17 | 16 |
| PRI_10 |    | Reserved |    |    |    |    |    |
| 15     | 14 | 13       | 12 | 11 | 10 | 9  | 8  |
| PRI_9  |    | Reserved |    |    |    |    |    |
| 7      | 6  | 5        | 4  | 3  | 2  | 1  | 0  |
| PRI_8  |    | Reserved |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | PRI_11      | <b>Priority of IRQ11</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [29:24] | Reserved    | Reserved   |
| [23:22] | PRI_10      | <b>Priority of IRQ10</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [21:16] | Reserved    | Reserved   |
| [15:14] | PRI_9       | <b>Priority of IRQ9</b><br>"0" denotes the highest priority and "3" denotes lowest priority  |
| [13:8]  | Reserved    | Reserved   |
| [7:6]   | PRI_8       | <b>Priority of IRQ8</b><br>"0" denotes the highest priority and "3" denotes lowest priority  |
| [5:0]   | Reserved    | Reserved   |



### IRQ12 ~ IRQ15 Interrupt Priority Register (NVIC IPR3)

| Register  | Offset        | R/W | Description                                       | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_IPR3 | NVIC_BA+0x30C | R/W | IRQ12 ~ IRQ15 Interrupt Priority Control Register | 0x0000_0000 |

|        |    |          |    |    |    |    |    |
|--------|----|----------|----|----|----|----|----|
| 31     | 30 | 29       | 28 | 27 | 26 | 25 | 24 |
| PRI_15 |    | Reserved |    |    |    |    |    |
| 23     | 22 | 21       | 20 | 19 | 18 | 17 | 16 |
| PRI_14 |    | Reserved |    |    |    |    |    |
| 15     | 14 | 13       | 12 | 11 | 10 | 9  | 8  |
| PRI_13 |    | Reserved |    |    |    |    |    |
| 7      | 6  | 5        | 4  | 3  | 2  | 1  | 0  |
| PRI_12 |    | Reserved |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | PRI_15      | <b>Priority of IRQ15</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [29:24] | Reserved    | Reserved   |
| [23:22] | PRI_14      | <b>Priority of IRQ14</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [21:16] | Reserved    | Reserved   |
| [15:14] | PRI_13      | <b>Priority of IRQ13</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [13:8]  | Reserved    | Reserved   |
| [7:6]   | PRI_12      | <b>Priority of IRQ12</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [5:0]   | Reserved    | Reserved   |

### IRQ16 ~ IRQ19 Interrupt Priority Register (NVIC\_IPR4)

| Register  | Offset        | R/W | Description                                       | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_IPR4 | NVIC_BA+0x310 | R/W | IRQ16 ~ IRQ19 Interrupt Priority Control Register | 0x0000_0000 |

|        |    |          |    |    |    |    |    |
|--------|----|----------|----|----|----|----|----|
| 31     | 30 | 29       | 28 | 27 | 26 | 25 | 24 |
| PRI_19 |    | Reserved |    |    |    |    |    |
| 23     | 22 | 21       | 20 | 19 | 18 | 17 | 16 |
| PRI_18 |    | Reserved |    |    |    |    |    |
| 15     | 14 | 13       | 12 | 11 | 10 | 9  | 8  |
| PRI_17 |    | Reserved |    |    |    |    |    |
| 7      | 6  | 5        | 4  | 3  | 2  | 1  | 0  |
| PRI_16 |    | Reserved |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | PRI_19      | <b>Priority of IRQ19</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [29:24] | Reserved    | Reserved   |
| [23:22] | PRI_18      | <b>Priority of IRQ18</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [21:16] | Reserved    | Reserved   |
| [15:14] | PRI_17      | <b>Priority of IRQ17</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [13:8]  | Reserved    | Reserved   |
| [7:6]   | PRI_16      | <b>Priority of IRQ16</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [5:0]   | Reserved    | Reserved   |

### IRQ20 ~ IRQ23 Interrupt Priority Register (NVIC IPR5)

| Register  | Offset        | R/W | Description                                       | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_IPR5 | NVIC_BA+0x314 | R/W | IRQ20 ~ IRQ23 Interrupt Priority Control Register | 0x0000_0000 |

| 31     | 30 | 29       | 28 | 27 | 26 | 25 | 24 |
|--------|----|----------|----|----|----|----|----|
| PRI_23 |    | Reserved |    |    |    |    |    |
| 23     | 22 | 21       | 20 | 19 | 18 | 17 | 16 |
| PRI_22 |    | Reserved |    |    |    |    |    |
| 15     | 14 | 13       | 12 | 11 | 10 | 9  | 8  |
| PRI_21 |    | Reserved |    |    |    |    |    |
| 7      | 6  | 5        | 4  | 3  | 2  | 1  | 0  |
| PRI_20 |    | Reserved |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | PRI_23      | <b>Priority of IRQ23</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [29:24] | Reserved    | Reserved   |
| [23:22] | PRI_22      | <b>Priority of IRQ22</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [21:16] | Reserved    | Reserved   |
| [15:14] | PRI_21      | <b>Priority of IRQ21</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [13:8]  | Reserved    | Reserved   |
| [7:6]   | PRI_20      | <b>Priority of IRQ20</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [5:0]   | Reserved    | Reserved   |

### IRQ24 ~ IRQ27 Interrupt Priority Register (NVIC\_IPR6)

| Register  | Offset        | R/W | Description                                       | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_IPR6 | NVIC_BA+0x318 | R/W | IRQ24 ~ IRQ27 Interrupt Priority Control Register | 0x0000_0000 |

|        |    |          |    |    |    |    |    |
|--------|----|----------|----|----|----|----|----|
| 31     | 30 | 29       | 28 | 27 | 26 | 25 | 24 |
| PRI_27 |    | Reserved |    |    |    |    |    |
| 23     | 22 | 21       | 20 | 19 | 18 | 17 | 16 |
| PRI_26 |    | Reserved |    |    |    |    |    |
| 15     | 14 | 13       | 12 | 11 | 10 | 9  | 8  |
| PRI_25 |    | Reserved |    |    |    |    |    |
| 7      | 6  | 5        | 4  | 3  | 2  | 1  | 0  |
| PRI_24 |    | Reserved |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | PRI_27      | <b>Priority of IRQ27</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [29:24] | Reserved    | Reserved   |
| [23:22] | PRI_26      | <b>Priority of IRQ26</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [21:16] | Reserved    | Reserved   |
| [15:14] | PRI_25      | <b>Priority of IRQ25</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [13:8]  | Reserved    | Reserved   |
| [7:6]   | PRI_24      | <b>Priority of IRQ24</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [5:0]   | Reserved    | Reserved   |

### IRQ28 ~ IRQ31 Interrupt Priority Register (NVIC IPR7)

| Register  | Offset        | R/W | Description                                       | Reset Value |
|-----------|---------------|-----|---|-------------|
| NVIC_IPR7 | NVIC_BA+0x31C | R/W | IRQ28 ~ IRQ31 Interrupt Priority Control Register | 0x0000_0000 |

|        |    |          |    |    |    |    |    |
|--------|----|----------|----|----|----|----|----|
| 31     | 30 | 29       | 28 | 27 | 26 | 25 | 24 |
| PRI_31 |    | Reserved |    |    |    |    |    |
| 23     | 22 | 21       | 20 | 19 | 18 | 17 | 16 |
| PRI_30 |    | Reserved |    |    |    |    |    |
| 15     | 14 | 13       | 12 | 11 | 10 | 9  | 8  |
| PRI_29 |    | Reserved |    |    |    |    |    |
| 7      | 6  | 5        | 4  | 3  | 2  | 1  | 0  |
| PRI_28 |    | Reserved |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | PRI_31      | <b>Priority of IRQ31</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [29:24] | Reserved    | Reserved   |
| [23:22] | PRI_30      | <b>Priority of IRQ30</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [21:16] | Reserved    | Reserved   |
| [15:14] | PRI_29      | <b>Priority of IRQ29</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [13:8]  | Reserved    | Reserved   |
| [7:6]   | PRI_28      | <b>Priority of IRQ28</b><br>"0" denotes the highest priority and "3" denotes lowest priority |
| [5:0]   | Reserved    | Reserved   |

### 7.7.5 Interrupt Source Control Registers

Besides the interrupt control registers associated with the NVIC, NuMicro™ NM15xx Series also implement some specific control registers to facilitate the interrupt functions, including “interrupt source identification”, “NMI source selection” and “interrupt test mode”, as described below.

**R:** read only, **W:** write only, **R/W:** both read and write

| Register                    | Offset      | R/W | Description  | Reset Value |
|-----------------------------|-------------|-----|--|-------------|
| <b>INT Base Address:</b>    |             |     |  |             |
| <b>INT_BA = 0x5000_0300</b> |             |     |  |             |
| <b>IRQ0_SRC</b>             | INT_BA+0x00 | R   | IRQ0 (BOD) interrupt source identity               | 0xFFFF_FFFF |
| <b>IRQ1_SRC</b>             | INT_BA+0x04 | R   | IRQ1 (WDT) interrupt source identity               | 0xFFFF_FFFF |
| <b>IRQ2_SRC</b>             | INT_BA+0x08 | R   | IRQ2 (EINT0) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ3_SRC</b>             | INT_BA+0x0C | R   | IRQ3 (EINT1) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ4_SRC</b>             | INT_BA+0x10 | R   | IRQ4 (P0-P4) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ5_SRC</b>             | INT_BA+0x14 | R   | IRQ5 (P5-PA) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ6_SRC</b>             | INT_BA+0x18 | R   | IRQ6 (BPWM) interrupt source identity              | 0xFFFF_FFFF |
| <b>IRQ7_SRC</b>             | INT_BA+0x1C | R   | IRQ7 (ADC0) interrupt source identity              | 0xFFFF_FFFF |
| <b>IRQ8_SRC</b>             | INT_BA+0x20 | R   | IRQ8 (TMR0) interrupt source identity              | 0xFFFF_FFFF |
| <b>IRQ9_SRC</b>             | INT_BA+0x24 | R   | IRQ9 (TMR1) interrupt source identity              | 0xFFFF_FFFF |
| <b>IRQ10_SRC</b>            | INT_BA+0x28 | R   | IRQ10 (TMR2) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ11_SRC</b>            | INT_BA+0x2C | R   | IRQ11 (TMR3) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ12_SRC</b>            | INT_BA+0x30 | R   | IRQ12 (UART0) interrupt source identity            | 0xFFFF_FFFF |
| <b>IRQ13_SRC</b>            | INT_BA+0x34 | R   | IRQ13 (UART1) interrupt source identity            | 0xFFFF_FFFF |
| <b>IRQ14_SRC</b>            | INT_BA+0x38 | R   | IRQ14 (SPI0) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ15_SRC</b>            | INT_BA+0x3C | R   | IRQ15 (SPI1) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ16_SRC</b>            | INT_BA+0x40 | R   | IRQ16 (SPI2) interrupt source identity             | 0xFFFF_FFFF |
| <b>IRQ17_SRC</b>            | INT_BA+0x44 | R   | IRQ17 (MDU) interrupt source identity              | 0xFFFF_FFFF |
| <b>IRQ18_SRC</b>            | INT_BA+0x48 | R   | IRQ18 (I <sup>2</sup> C) interrupt source identity | 0xFFFF_FFFF |
| <b>IRQ19_SRC</b>            | INT_BA+0x4C | R   | IRQ19 (CKD) interrupt source identity              | 0xFFFF_FFFF |
| <b>IRQ20_SRC</b>            | INT_BA+0x50 | R   | IRQ20 (CAN) interrupt source identity              | 0xFFFF_FFFF |
| <b>IRQ21_SRC</b>            | INT_BA+0x54 | R   | IRQ21 (EPWM0) interrupt source identity            | 0xFFFF_FFFF |
| <b>IRQ22_SRC</b>            | INT_BA+0x58 | R   | IRQ22 (EPWM1) interrupt source identity            | 0xFFFF_FFFF |
| <b>IRQ23_SRC</b>            | INT_BA+0x5C | R   | IRQ23 (CAP0) interrupt source identity             | 0xFFFF_FFFF |

|                  |             |     |  |             |
|------------------|-------------|-----|--|-------------|
| <b>IRQ24_SRC</b> | INT_BA+0x60 | R   | IRQ24 (CAP1) interrupt source identity       | 0XXXXX_XXXX |
| <b>IRQ25_SRC</b> | INT_BA+0x64 | R   | IRQ25 (ACMP) interrupt source identity       | 0XXXXX_XXXX |
| <b>IRQ26_SRC</b> | INT_BA+0x68 | R   | IRQ26 (QE10) interrupt source identity       | 0XXXXX_XXXX |
| <b>IRQ27_SRC</b> | INT_BA+0x6C | R   | IRQ27 (QE11) interrupt source identity       | 0XXXXX_XXXX |
| <b>IRQ28_SRC</b> | INT_BA+0x70 | R   | IRQ28 (PWRWU) interrupt source identity      | 0XXXXX_XXXX |
| <b>IRQ29_SRC</b> | INT_BA+0x74 | R   | IRQ29 (ADC1) interrupt source identity       | 0XXXXX_XXXX |
| <b>IRQ30_SRC</b> | INT_BA+0x78 | R   | IRQ30 (ADC2) interrupt source identity       | 0XXXXX_XXXX |
| <b>IRQ31_SRC</b> | INT_BA+0x7C | R   | IRQ31 (ADC3) interrupt source identity       | 0XXXXX_XXXX |
| <b>NMI_SEL</b>   | INT_BA+0x80 | R/W | NMI Interrupt Source Select Control Register | 0x0000_0000 |
| <b>MCU_IRQ</b>   | INT_BA+0x84 | R/W | MCU interrupt request source register        | 0x0000_0000 |
| <b>MCU_IRQCR</b> | INT_BA+0x88 | R/W | MCU Interrupt Request Control Register       | 0x0000_0000 |

### IRQ0 (BOD) interrupt source identity (IRQ0\_SRC)

| Register | Offset      | R/W | Description                          | Reset Value |
|----------|-------------|-----|--------------------------------------|-------------|
| IRQ0_SRC | INT_BA+0x00 | R   | IRQ0 (BOD) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |         |
|----------|----|----|----|----|----|----|---------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24      |
| Reserved |    |    |    |    |    |    |         |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16      |
| Reserved |    |    |    |    |    |    |         |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8       |
| Reserved |    |    |    |    |    |    |         |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0       |
| Reserved |    |    |    |    |    |    | BOD_INT |

| Bits   | Description |                                |
|--------|-------------|--------------------------------|
| [31:1] | Reserved    | Reserved                       |
| [0]    | BOD_INT     | Identify BOD interrupt source. |



### IRQ1 (WDT) interrupt source identity (IRQ1\_SRC)

| Register | Offset      | R/W | Description                          | Reset Value |
|----------|-------------|-----|--------------------------------------|-------------|
| IRQ1_SRC | INT_BA+0x04 | R   | IRQ1 (WDT) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |          |         |
|----------|----|----|----|----|----|----------|---------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25       | 24      |
| Reserved |    |    |    |    |    |          |         |
| 23       | 22 | 21 | 20 | 19 | 18 | 17       | 16      |
| Reserved |    |    |    |    |    |          |         |
| 15       | 14 | 13 | 12 | 11 | 10 | 9        | 8       |
| Reserved |    |    |    |    |    |          |         |
| 7        | 6  | 5  | 4  | 3  | 2  | 1        | 0       |
| Reserved |    |    |    |    |    | WWDT_INT | WDT_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:2] | Reserved    | Reserved                        |
| [1]    | WWDT_INT    | Identify WWDT interrupt source. |
| [0]    | WDT_INT     | Identify WDT interrupt source.  |

### IRQ2 (EINT0) interrupt source identity (IRQ2\_SRC)

| Register | Offset      | R/W | Description                            | Reset Value |
|----------|-------------|-----|--|-------------|
| IRQ2_SRC | INT_BA+0x08 | R   | IRQ2 (EINT0) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |       |
|----------|----|----|----|----|----|----|-------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24    |
| Reserved |    |    |    |    |    |    |       |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16    |
| Reserved |    |    |    |    |    |    |       |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8     |
| Reserved |    |    |    |    |    |    |       |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0     |
| Reserved |    |    |    |    |    |    | EINT0 |

| Bits   | Description |  |
|--------|-------------|--|
| [31:1] | Reserved    | Reserved   |
| [0]    | EINT0       | Identify EINT0 interrupt source.<br>EINT0 is external interrupt 0 from P3.2. |

**IRQ3 (EINT1) interrupt source identity (IRQ3\_SRC)**

| Register | Offset      | R/W | Description                            | Reset Value |
|----------|-------------|-----|--|-------------|
| IRQ3_SRC | INT_BA+0x0C | R   | IRQ3 (EINT1) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |       |
|----------|----|----|----|----|----|----|-------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24    |
| Reserved |    |    |    |    |    |    |       |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16    |
| Reserved |    |    |    |    |    |    |       |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8     |
| Reserved |    |    |    |    |    |    |       |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0     |
| Reserved |    |    |    |    |    |    | EINT1 |

| Bits   | Description |  |
|--------|-------------|--|
| [31:1] | Reserved    | Reserved   |
| [0]    | EINT1       | Identify EINT1 interrupt source.<br>EINT1 is external interrupt from P3.3. |

### IRQ4 (P0-P4) interrupt source identity (IRQ4\_SRC)

| Register | Offset      | R/W | Description                            | Reset Value |
|----------|-------------|-----|--|-------------|
| IRQ4_SRC | INT_BA+0x10 | R   | IRQ4 (P0-P4) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |        |        |        |        |        |
|----------|----|----|--------|--------|--------|--------|--------|
| 31       | 30 | 29 | 28     | 27     | 26     | 25     | 24     |
| Reserved |    |    |        |        |        |        |        |
| 23       | 22 | 21 | 20     | 19     | 18     | 17     | 16     |
| Reserved |    |    |        |        |        |        |        |
| 15       | 14 | 13 | 12     | 11     | 10     | 9      | 8      |
| Reserved |    |    |        |        |        |        |        |
| 7        | 6  | 5  | 4      | 3      | 2      | 1      | 0      |
| Reserved |    |    | P4_INT | P3_INT | P2_INT | P1_INT | P0_INT |

| Bits   | Description |                               |
|--------|-------------|-------------------------------|
| [31:5] | Reserved    | Reserved                      |
| [4]    | P4_INT      | Identify P4 interrupt source. |
| [3]    | P3_INT      | Identify P3 interrupt source. |
| [2]    | P2_INT      | Identify P2 interrupt source. |
| [1]    | P1_INT      | Identify P1 interrupt source. |
| [0]    | P0_INT      | Identify P0 interrupt source. |

**IRQ5 (P5-PA) interrupt source identity (IRQ5\_SRC)**

| Register | Offset      | R/W | Description                            | Reset Value |
|----------|-------------|-----|--|-------------|
| IRQ5_SRC | INT_BA+0x14 | R   | IRQ5 (P5-PA) interrupt source identity | 0xFFFF_XXXX |

|          |    |        |        |        |        |        |        |
|----------|----|--------|--------|--------|--------|--------|--------|
| 31       | 30 | 29     | 28     | 27     | 26     | 25     | 24     |
| Reserved |    |        |        |        |        |        |        |
| 23       | 22 | 21     | 20     | 19     | 18     | 17     | 16     |
| Reserved |    |        |        |        |        |        |        |
| 15       | 14 | 13     | 12     | 11     | 10     | 9      | 8      |
| Reserved |    |        |        |        |        |        |        |
| 7        | 6  | 5      | 4      | 3      | 2      | 1      | 0      |
| Reserved |    | PA_INT | P9_INT | P8_INT | P7_INT | P6_INT | P5_INT |

| Bits   | Description |                               |
|--------|-------------|-------------------------------|
| [31:6] | Reserved    | Reserved                      |
| [5]    | PA_INT      | Identify PA interrupt source. |
| [4]    | P9_INT      | Identify P9 interrupt source. |
| [3]    | P8_INT      | Identify P8 interrupt source. |
| [2]    | P7_INT      | Identify P7 interrupt source. |
| [1]    | P6_INT      | Identify P6 interrupt source. |
| [0]    | P5_INT      | Identify P5 interrupt source. |

### IRQ6 (BPWM) interrupt source identity (IRQ6\_SRC)

| Register | Offset      | R/W | Description                           | Reset Value |
|----------|-------------|-----|---------------------------------------|-------------|
| IRQ6_SRC | INT_BA+0x18 | R   | IRQ6 (BPWM) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |           |           |
|----------|----|----|----|----|----|-----------|-----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25        | 24        |
| Reserved |    |    |    |    |    |           |           |
| 23       | 22 | 21 | 20 | 19 | 18 | 17        | 16        |
| Reserved |    |    |    |    |    |           |           |
| 15       | 14 | 13 | 12 | 11 | 10 | 9         | 8         |
| Reserved |    |    |    |    |    |           |           |
| 7        | 6  | 5  | 4  | 3  | 2  | 1         | 0         |
| Reserved |    |    |    |    |    | BPWM1_INT | BPWM0_INT |

| Bits   | Description                                   |
|--------|---|
| [31:2] | Reserved                                      |
| [1]    | BPWM1_INT<br>Identify BPWM1 interrupt source. |
| [0]    | BPWM0_INT<br>Identify BPWM0 interrupt source. |

### IRQ7 (ADC0) interrupt source identity (IRQ7\_SRC)

| Register | Offset      | R/W | Description                           | Reset Value |
|----------|-------------|-----|---------------------------------------|-------------|
| IRQ7_SRC | INT_BA+0x1C | R   | IRQ7 (ADC0) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | ADC0_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | ADC0_INT    | Identify ADC0 interrupt source. |

### IRQ8 (TMR0) interrupt source identity (IRQ8\_SRC)

| Register | Offset      | R/W | Description                           | Reset Value |
|----------|-------------|-----|---------------------------------------|-------------|
| IRQ8_SRC | INT_BA+0x20 | R   | IRQ8 (TMR0) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | TMR0_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | TMR0_INT    | Identify TMR0 interrupt source. |



### IRQ9 (TMR1) interrupt source identity (IRQ9\_SRC)

| Register | Offset      | R/W | Description                           | Reset Value |
|----------|-------------|-----|---------------------------------------|-------------|
| IRQ9_SRC | INT_BA+0x24 | R   | IRQ9 (TMR1) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | TMR1_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | TMR1_INT    | Identify TMR1 interrupt source. |

### IRQ10 (TMR2) interrupt source identity (IRQ10\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ10_SRC | INT_BA+0x28 | R   | IRQ10 (TMR2) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | TMR2_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | TMR2_INT    | Identify TMR2 interrupt source. |

**IRQ11 (TMR3) interrupt source identity (IRQ11\_SRC)**

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ11_SRC | INT_BA+0x2C | R   | IRQ11 (TMR3) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | TMR3_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | TMR3_INT    | Identify TMR3 interrupt source. |

### IRQ12 (UART0) interrupt source identity (IRQ12\_SRC)

| Register  | Offset      | R/W | Description                             | Reset Value |
|-----------|-------------|-----|---|-------------|
| IRQ12_SRC | INT_BA+0x30 | R   | IRQ12 (UART0) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |           |
|----------|----|----|----|----|----|----|-----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24        |
| Reserved |    |    |    |    |    |    |           |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16        |
| Reserved |    |    |    |    |    |    |           |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8         |
| Reserved |    |    |    |    |    |    |           |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0         |
| Reserved |    |    |    |    |    |    | UART0_INT |

| Bits   | Description |                                  |
|--------|-------------|----------------------------------|
| [31:1] | Reserved    | Reserved                         |
| [0]    | UART0_INT   | Identify UART0 interrupt source. |

### IRQ13 (UART1) interrupt source identity (IRQ13\_SRC)

| Register  | Offset      | R/W | Description                             | Reset Value |
|-----------|-------------|-----|---|-------------|
| IRQ13_SRC | INT_BA+0x34 | R   | IRQ13 (UART1) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |           |
|----------|----|----|----|----|----|----|-----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24        |
| Reserved |    |    |    |    |    |    |           |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16        |
| Reserved |    |    |    |    |    |    |           |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8         |
| Reserved |    |    |    |    |    |    |           |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0         |
| Reserved |    |    |    |    |    |    | UART1_INT |

| Bits   | Description |                                  |
|--------|-------------|----------------------------------|
| [31:1] | Reserved    | Reserved                         |
| [0]    | UART1_INT   | Identify UART1 interrupt source. |

### IRQ14 (SPI0) interrupt source identity (IRQ14\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ14_SRC | INT_BA+0x38 | R   | IRQ14 (SPI0) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | SPI0_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | SPI0_INT    | Identify SPI0 interrupt source. |

### IRQ15 (SPI1) interrupt source identity (IRQ15\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ15_SRC | INT_BA+0x3C | R   | IRQ15 (SPI1) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | SPI1_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | SPI1_INT    | Identify SPI1 interrupt source. |

### IRQ16 (SPI2) interrupt source identity (IRQ16\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ16_SRC | INT_BA+0x40 | R   | IRQ16 (SPI2) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | SPI2_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | SPI2_INT    | Identify SPI2 interrupt source. |



**IRQ17 (MDU) interrupt source identity (IRQ17\_SRC)**

| Register  | Offset      | R/W | Description                           | Reset Value |
|-----------|-------------|-----|---------------------------------------|-------------|
| IRQ17_SRC | INT_BA+0x44 | R   | IRQ17 (MDU) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |         |
|----------|----|----|----|----|----|----|---------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24      |
| Reserved |    |    |    |    |    |    |         |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16      |
| Reserved |    |    |    |    |    |    |         |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8       |
| Reserved |    |    |    |    |    |    |         |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0       |
| Reserved |    |    |    |    |    |    | MDU_INT |

| Bits   | Description |                                |
|--------|-------------|--------------------------------|
| [31:1] | Reserved    | Reserved                       |
| [0]    | MDU_INT     | Identify MDU interrupt source. |

### IRQ18 (I2C) interrupt source identity (IRQ18\_SRC)

| Register  | Offset      | R/W | Description                           | Reset Value |
|-----------|-------------|-----|---------------------------------------|-------------|
| IRQ18_SRC | INT_BA+0x48 | R   | IRQ18 (I2C) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |         |
|----------|----|----|----|----|----|----|---------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24      |
| Reserved |    |    |    |    |    |    |         |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16      |
| Reserved |    |    |    |    |    |    |         |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8       |
| Reserved |    |    |    |    |    |    |         |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0       |
| Reserved |    |    |    |    |    |    | I2C_INT |

| Bits   | Description |
|--------|-------------|
| [31:1] | Reserved    |
| [0]    | I2C_INT     |

### IRQ20 (CAN) interrupt source identity (IRQ20\_SRC)

| Register  | Offset      | R/W | Description                           | Reset Value |
|-----------|-------------|-----|---------------------------------------|-------------|
| IRQ20_SRC | INT_BA+0x50 | R   | IRQ20 (CAN) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |         |
|----------|----|----|----|----|----|----|---------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24      |
| Reserved |    |    |    |    |    |    |         |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16      |
| Reserved |    |    |    |    |    |    |         |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8       |
| Reserved |    |    |    |    |    |    |         |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0       |
| Reserved |    |    |    |    |    |    | CAN_INT |

| Bits   | Description |
|--------|-------------|
| [31:1] | Reserved    |
| [0]    | CAN_INT     |

### IRQ21 (EPWM0) interrupt source identity (IRQ21\_SRC)

| Register  | Offset      | R/W | Description                             | Reset Value |
|-----------|-------------|-----|---|-------------|
| IRQ21_SRC | INT_BA+0x54 | R   | IRQ21 (EPWM0) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |           |
|----------|----|----|----|----|----|----|-----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24        |
| Reserved |    |    |    |    |    |    |           |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16        |
| Reserved |    |    |    |    |    |    |           |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8         |
| Reserved |    |    |    |    |    |    |           |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0         |
| Reserved |    |    |    |    |    |    | EPWM0_INT |

| Bits   | Description |                                  |
|--------|-------------|----------------------------------|
| [31:1] | Reserved    | Reserved                         |
| [0]    | EPWM0_INT   | Identify EPWM0 interrupt source. |

### IRQ22 (EPWM1) interrupt source identity (IRQ22\_SRC)

| Register  | Offset      | R/W | Description                             | Reset Value |
|-----------|-------------|-----|---|-------------|
| IRQ22_SRC | INT_BA+0x58 | R   | IRQ22 (EPWM1) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |           |
|----------|----|----|----|----|----|----|-----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24        |
| Reserved |    |    |    |    |    |    |           |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16        |
| Reserved |    |    |    |    |    |    |           |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8         |
| Reserved |    |    |    |    |    |    |           |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0         |
| Reserved |    |    |    |    |    |    | EPWM1_INT |

| Bits   | Description |                                  |
|--------|-------------|----------------------------------|
| [31:1] | Reserved    | Reserved                         |
| [0]    | EPWM1_INT   | Identify EPWM1 interrupt source. |

### IRQ23 (CAP0) interrupt source identity (IRQ23\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ23_SRC | INT_BA+0x5C | R   | IRQ23 (CAP0) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | CAP0_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | CAP0_INT    | Identify CAP0 interrupt source. |

### IRQ24 (CAP1) interrupt source identity (IRQ24\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ24_SRC | INT_BA+0x60 | R   | IRQ24 (CAP1) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | CAP1_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | CAP1_INT    | Identify CAP1 interrupt source. |

### IRQ25 (ACMP) interrupt source identity (IRQ25\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ25_SRC | INT_BA+0x64 | R   | IRQ25 (ACMP) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | ACMP_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | ACMP_INT    | Identify ACMP interrupt source. |



**IRQ26 (QEIO) interrupt source identity (IRQ26\_SRC)**

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ26_SRC | INT_BA+0x68 | R   | IRQ26 (QEIO) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | QEIO_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | QEIO_INT    | Identify QEIO interrupt source. |

### IRQ27 (QE1) interrupt source identity (IRQ27\_SRC)

| Register  | Offset      | R/W | Description                           | Reset Value |
|-----------|-------------|-----|---------------------------------------|-------------|
| IRQ27_SRC | INT_BA+0x6C | R   | IRQ27 (QE1) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |     |
|----------|----|----|----|----|----|----|-----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24  |
| Reserved |    |    |    |    |    |    |     |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16  |
| Reserved |    |    |    |    |    |    |     |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8   |
| Reserved |    |    |    |    |    |    |     |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0   |
| Reserved |    |    |    |    |    |    | QE1 |

| Bits   | Description |                                |
|--------|-------------|--------------------------------|
| [31:1] | Reserved    | Reserved                       |
| [0]    | QE1_INT     | Identify QE1 interrupt source. |

### IRQ28 (PWRWU) interrupt source identity (IRQ28\_SRC)

| Register  | Offset      | R/W | Description                             | Reset Value |
|-----------|-------------|-----|---|-------------|
| IRQ28_SRC | INT_BA+0x70 | R   | IRQ28 (PWRWU) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |           |
|----------|----|----|----|----|----|----|-----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24        |
| Reserved |    |    |    |    |    |    |           |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16        |
| Reserved |    |    |    |    |    |    |           |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8         |
| Reserved |    |    |    |    |    |    |           |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0         |
| Reserved |    |    |    |    |    |    | PWRWU_INT |

| Bits   | Description |                                  |
|--------|-------------|----------------------------------|
| [31:1] | Reserved    | Reserved                         |
| [0]    | PWRWU_INT   | Identify PWRWU interrupt source. |

### IRQ29 (ADC1) interrupt source identity (IRQ29\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ29_SRC | INT_BA+0x74 | R   | IRQ29 (ADC1) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | ADC1_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | ADC1_INT    | Identify ADC1 interrupt source. |

**IRQ30 (ADC2) interrupt source identity (IRQ30\_SRC)**

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ30_SRC | INT_BA+0x78 | R   | IRQ30 (ADC2) interrupt source identity | 0xFFFF_XXXX |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | ADC2_INT |

| Bits   | Description |                                 |
|--------|-------------|---------------------------------|
| [31:1] | Reserved    | Reserved                        |
| [0]    | ADC2_INT    | Identify ADC2 interrupt source. |

### IRQ31 (ADC3) interrupt source identity (IRQ31\_SRC)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| IRQ31_SRC | INT_BA+0x7C | R   | IRQ31 (ADC3) interrupt source identity | 0xFFFF_FFFF |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | ADC3_INT |

| Bits   | Description |
|--------|-------------|
| [31:1] | Reserved    |
| [0]    | ADC3_INT    |

**NMI Interrupt Source Select Control Register (NMI\_SEL)**

| Register | Offset      | R/W | Description                                  | Reset Value |
|----------|-------------|-----|--|-------------|
| NMI_SEL  | INT_BA+0x80 | R/W | NMI Interrupt Source Select Control Register | 0x0000_0000 |

|    |    |    |              |    |    |    |        |
|----|----|----|--------------|----|----|----|--------|
| 31 | 30 | 29 | 28           | 27 | 26 | 25 | 24     |
| -  |    |    |              |    |    |    |        |
| 23 | 22 | 21 | 20           | 19 | 18 | 17 | 16     |
| -  |    |    |              |    |    |    |        |
| 15 | 14 | 13 | 12           | 11 | 10 | 9  | 8      |
| -  |    |    |              |    |    |    | MNI_EN |
| 7  | 6  | 5  | 4            | 3  | 2  | 1  | 0      |
| -  |    |    | NMI_SEL[4:0] |    |    |    |        |

| Bits   | Description |   |
|--------|-------------|---|
| [31:9] | -           | Reserved.   |
| [8]    | NMI_EN      | <b>NMI Interrupt Enable</b><br>0 = IRQ0~31 assigned to NMI Disabled. (NMI still can be software triggered by setting its pending flag.)<br>1 = IRQ0~31 assigned to NMI Enabled.   |
| [7:5]  | -           | Reserved.   |
| [4:0]  | NMI_SEL     | <b>NMI Interrupt Source Selection</b><br>The NMI interrupt to Cortex-M0 can be selected from one of IRQ0~IRQ31 by setting NMI_SEL with IRQ number. The default NMI interrupt is assigned as IRQ0 interrupt if NMI is enabled by setting NMI_SEL[8]. |

**MCU Interrupt Request Source Register (MCU\_IRQ)**

| Register | Offset      | R/W | Description                           | Reset Value |
|----------|-------------|-----|---------------------------------------|-------------|
| MCU_IRQ  | INT_BA+0x84 | R/W | MCU interrupt request source register | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| MCU_IRQ[31:24] |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| MCU_IRQ[23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| MCU_IRQ[15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| MCU_IRQ[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [31:0] | <p><b>MCU_IRQ Source Register</b></p> <p>The MCU_IRQ collects all the interrupts from the peripherals and generates the synchronous interrupt to Cortex-M0.</p> <p>The MCU_IRQ collects all interrupts from each peripheral and synchronizes them and then interrupts the Cortex-M0.</p> <p>When the MCU_IRQ[n] is 0:</p> <p>1 = Generate an interrupt to Cortex_M0 NVIC[n].</p> <p>0 = No effect.</p> <p>When the MCU_IRQ[n] is 1 (means an interrupt is assert):</p> <p>1 = Clear the interrupt and MCU_IRQ[n].</p> <p>0 = No effect.</p> |



### MCU Interrupt Request Control Register (MCU\_IRQCR)

| Register  | Offset      | R/W | Description                            | Reset Value |
|-----------|-------------|-----|--|-------------|
| MCU_IRQCR | INT_BA+0x88 | R/W | MCU Interrupt Request Control Register | 0x0000_0000 |

|    |    |    |    |    |    |    |          |
|----|----|----|----|----|----|----|----------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
|    |    |    |    |    |    |    | FAST_IRQ |

| Bits   | Description  |
|--------|--|
| [31:1] | - Reserved.  |
| [0]    | <b>Fast IRQ Latency Enable</b><br>1= MCU IRQ latency will not fixed, MCU will enter IRQ handler as soon as possible when interrupt happened.<br>0= MCU IRQ latency is fixed at 13 HCLK, MCU will enter IRQ handler after this fixed latency when interrupt happened. |

### 7.8 System Control Block Register

Cortex-M0 status and operating mode control are managed System Control Block Registers including CPUID, Cortex-M0 interrupt priority and Cortex-M0 power management.

**R**: read only, **W**: write only, **R/W**: both read and write

| Register                    | Offset      | R/W | Description                                      | Reset Value |
|-----------------------------|-------------|-----|--|-------------|
| <b>SCB Base Address:</b>    |             |     |  |             |
| <b>SCB_BA = 0xE000_ED00</b> |             |     |  |             |
| <b>CPUID</b>                | SCB_BA+0x00 | R   | CPUID Register                                   | 0x410C_C200 |
| <b>ICSR</b>                 | SCB_BA+0x04 | R/W | Interrupt Control State Register                 | 0x0000_0000 |
| <b>AIRCR</b>                | SCB_BA+0x0C | R/W | Application Interrupt and Reset Control Register | 0xFA05_0000 |
| <b>SCR</b>                  | SCB_BA+0x10 | R/W | System Control Register                          | 0x0000_0000 |
| <b>SHPR2</b>                | SCB_BA+0x1C | R/W | System Handler Priority Register 2               | 0x0000_0000 |
| <b>SHPR3</b>                | SCB_BA+0x20 | R/W | System Handler Priority Register 3               | 0x0000_0000 |

### CPUID Register (CPUID)

| Register     | Offset      | R/W | Description    | Reset Value |
|--------------|-------------|-----|----------------|-------------|
| <b>CPUID</b> | SCB_BA+0x00 | R   | CPUID Register | 0x410C_C200 |

|                  |    |    |    |               |    |    |    |
|------------------|----|----|----|---------------|----|----|----|
| 31               | 30 | 29 | 28 | 27            | 26 | 25 | 24 |
| Implementer[7:0] |    |    |    |               |    |    |    |
| 23               | 22 | 21 | 20 | 19            | 18 | 17 | 16 |
| Variant[3:0]     |    |    |    | Constant[3:0] |    |    |    |
| 15               | 14 | 13 | 12 | 11            | 10 | 9  | 8  |
| Partno[11:4]     |    |    |    |               |    |    |    |
| 7                | 6  | 5  | 4  | 3             | 2  | 1  | 0  |
| Partno[3:0]      |    |    |    | Revision[3:0] |    |    |    |

| Bits    | Description        |  |
|---------|--------------------|--|
| [31:24] | <b>Implementer</b> | Implementer code assigned by ARM. (ARM = 0x41)     |
| [23:20] | <b>Variant</b>     | Read as 0x0.                                       |
| [19:16] | <b>Constant</b>    | Read as 0xC corresponding to ARMv6-M architecture. |
| [15:4]  | <b>Partno</b>      | Reads as 0xC20 corresponding to Cortex-M0          |
| [3:0]   | <b>Revision</b>    | Reads as 0x0.                                      |

### Interrupt Control State Register (ICSR)

The ICSR provides (1)a set-pending bit for the NMI exception, and (2)set-pending and clear-pending bits for the PendSV and SysTick exceptions. And the ICSR also indicates (1)the exception number of the exception being processed, (2)whether there are preempted active exceptions, (3)the exception number of the highest priority pending exception, and (4)whether any interrupts are pending.

| Register | Offset      | R/W | Description                      | Reset Value |
|----------|-------------|-----|----------------------------------|-------------|
| ICSR     | SCB_BA+0x04 | R/W | Interrupt Control State Register | 0x0000_0000 |

| 31               | 30          | 29              | 28 | 27        | 26        | 25               | 24        |
|------------------|-------------|-----------------|----|-----------|-----------|------------------|-----------|
| NMIPENDSET       | -           |                 |    | PENDSVSET | PENDSVCLR | PENDSTSET        | PENDSTCLR |
| 23               | 22          | 21              | 20 | 19        | 18        | 17               | 16        |
| -                | ISR_PENDING | -               |    |           |           | VECTPENDING[5:4] |           |
| 15               | 14          | 13              | 12 | 11        | 10        | 9                | 8         |
| VECTPENDING[3:0] |             |                 |    | -         |           |                  |           |
| 7                | 6           | 5               | 4  | 3         | 2         | 1                | 0         |
| -                |             | VECTACTIVE[5:0] |    |           |           |                  |           |

| Bits    | Description  |
|---------|--|
| [31]    | <p><b>NMIPENDSET</b></p> <p><b>NMI set-pending bit.</b></p> <p>Write:</p> <p>0 = no effect</p> <p>1 = changes NMI exception state to pending.</p> <p>Read:</p> <p>0 = NMI exception is not pending</p> <p>1 = NMI exception is pending.</p> <p>Because NMI is the highest-priority exception, normally the processor enters the NMI exception handler as soon as it detects a write of 1 to this bit. Entering the handler then clears this bit to 0. This means a read of this bit by the NMI exception handler returns 1 only if the NMI signal is reasserted while the processor is executing that handler.</p> |
| [30:29] | <p>-</p> <p><b>Reserved.</b></p>   |
| [28]    | <p><b>PENDSVSET</b></p> <p><b>PendSV set-pending bit.</b></p> <p>Write:</p> <p>0 = no effect.</p> <p>1 = changes PendSV exception state to pending.</p> <p>Read:</p> <p>0 = PendSV exception is not pending.</p> <p>1 = PendSV exception is pending.</p> <p>Writing 1 to this bit is the only way to set the PendSV exception state to pending.</p>  |

| Bits    | Description        |   |
|---------|--------------------|---|
| [27]    | <b>PENDSVCLR</b>   | <b>PendSV clear-pending bit. (Write only)</b><br>Write:<br>0 = no effect.<br>1 = removes the pending state from the PendSV exception.   |
| [26]    | <b>PENDSTSET</b>   | <b>SysTick exception set-pending bit.</b><br>Write:<br>0 = no effect.<br>1 = changes SysTick exception state to pending.<br>Read:<br>0 = SysTick exception is not pending.<br>1 = SysTick exception is pending.   |
| [25]    | <b>PENDSTCLR</b>   | <b>SysTick exception clear-pending bit. (Write only)</b><br>Write:<br>0 = no effect<br>1 = removes the pending state from the SysTick exception.  |
| [24:23] | -                  | <b>Reserved.</b>  |
| [22]    | <b>ISRPENDING</b>  | <b>Interrupt pending flag. (Read only)</b><br>Indicates if an external configurable (NVIC generated) interrupt is pending.<br>0 = interrupt not pending.<br>1 = interrupt pending.  |
| [21:18] | -                  | <b>Reserved.</b>  |
| [17:12] | <b>VECTPENDING</b> | <b>Vector pending indicator. (Read only)</b><br>This field indicates the exception number of the highest priority pending enabled exception:<br>0 = no pending exceptions.<br>Nonzero = the exception number of the highest priority pending enabled exception. |
| [11:6]  | -                  | <b>Reserved.</b>  |
| [5:0]   | <b>VECTACTIVE</b>  | <b>Vector active indicator. (Read only)</b><br>This field contains the active exception number:<br>0 = Thread mode.<br>Nonzero = The exception number of the currently active exception.  |

### Application Interrupt and Reset Control Register (AIRCR)

| Register | Offset      | R/W | Description                                      | Reset Value |
|----------|-------------|-----|--|-------------|
| AIRCR    | SCB_BA+0x0C | R/W | Application Interrupt and Reset Control Register | 0xFA05_0000 |

|                 |    |    |    |    |             |    |    |
|-----------------|----|----|----|----|-------------|----|----|
| 31              | 30 | 29 | 28 | 27 | 26          | 25 | 24 |
| VECTORKEY[15:8] |    |    |    |    |             |    |    |
| 23              | 22 | 21 | 20 | 19 | 18          | 17 | 16 |
| VECTKEY[7:0]    |    |    |    |    |             |    |    |
| 15              | 14 | 13 | 12 | 11 | 10          | 9  | 8  |
| -               |    |    |    |    |             |    |    |
| 7               | 6  | 5  | 4  | 3  | 2           | 1  | 0  |
| -               |    |    |    |    | SYSRESETREQ | -  |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | VECTKEY     | Register key. (Write only)<br>Reads as Unknown. On writes, write 0x05FA to VECTKEY, otherwise the write is ignored. |
| [15:3]  | -           | Reserved.   |
| [2]     | SYSRESETREQ | System reset request. (Write only)<br>0 = no effect.<br>1 = requests a system level reset.                          |
| [1:0]   | -           | Reserved.   |

### System Control Register (SCR)

| Register | Offset      | R/W | Description             | Reset Value |
|----------|-------------|-----|-------------------------|-------------|
| SCR      | SCB_BA+0x10 | R/W | System Control Register | 0x0000_0000 |

|    |    |    |           |    |           |             |    |
|----|----|----|-----------|----|-----------|-------------|----|
| 31 | 30 | 29 | 28        | 27 | 26        | 25          | 24 |
| -  |    |    |           |    |           |             |    |
| 23 | 22 | 21 | 20        | 19 | 18        | 17          | 16 |
| -  |    |    |           |    |           |             |    |
| 15 | 14 | 13 | 12        | 11 | 10        | 9           | 8  |
| -  |    |    |           |    |           |             |    |
| 7  | 6  | 5  | 4         | 3  | 2         | 1           | 0  |
| -  |    |    | SEVONPEND | -  | SLEEPDEEP | SLEEPONEXIT | -  |

| Bits   | Description |   |
|--------|-------------|---|
| [31:5] | -           | <b>Reserved.</b>  |
| [4]    | SEVONPEND   | <b>Send Event on Pending Bit</b><br>0 = Only enabled interrupts or events can wake up the processor, disabled interrupts are excluded.<br>1 = Enabled events and all interrupts, including disabled interrupts, can wakeup the processor.<br>When an event or interrupt enters pending state, the event signal wakes up the processor from WFE. If the processor is not waiting for an event, the event is registered and affects next WFE.<br>The processor also wakes up on execution of an SEV instruction or an external event. |
| [3]    | -           | <b>Reserved.</b>  |
| [2]    | SLEEPDEEP   | <b>Deep Sleep Mode Enable</b><br>This bit controls whether the processor uses sleep or deep sleep as its low power mode:<br>0 = Sleep.<br>1 = Deep sleep.   |
| [1]    | SLEEPONEXIT | <b>Sleep-on-Exit Enable</b><br>This bit controls sleep-on-exit when returning from Handler mode to Thread mode:<br>0 = Do not sleep when returning to Thread mode.<br>1 = Enter Sleep, or Deep Sleep, on return from an ISR to Thread mode.<br>Setting this bit to 1 enables an interrupt driven application to avoid returning to an empty main application.   |
| [0]    | -           | <b>Reserved.</b>  |

### System Handler Priority Register 2 (SHPR2)

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| SHPR2    | SCB_BA+0x1C | R/W | System Handler Priority Register 2 | 0x0000_0000 |

|        |    |    |    |    |    |    |    |
|--------|----|----|----|----|----|----|----|
| 31     | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| PRI_11 |    | -  |    |    |    |    |    |
| 23     | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -      |    |    |    |    |    |    |    |
| 15     | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -      |    |    |    |    |    |    |    |
| 7      | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| -      |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:30] | PRI_11      | Priority of System Handler 11, SVCall<br>"0" denotes the highest priority and "3" denotes the lowest priority |
| [29:0]  | -           | Reserved.   |



### System Handler Priority Register 3 (SHPR3)

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| SHPR3    | SCB_BA+0x20 | R/W | System Handler Priority Register 3 | 0x0000_0000 |

|        |    |    |    |    |    |    |    |
|--------|----|----|----|----|----|----|----|
| 31     | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| PRI_15 |    | -  |    |    |    |    |    |
| 23     | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| PRI_14 |    | -  |    |    |    |    |    |
| 15     | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -      |    |    |    |    |    |    |    |
| 7      | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| -      |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | PRI_15      | Priority of System Handler 15, SysTick<br>"0" denotes the highest priority and "3" denotes the lowest priority |
| [29:24] | -           | Reserved.  |
| [23:22] | PRI_14      | Priority of System Handler 14, PendSV<br>"0" denotes the highest priority and "3" denotes the lowest priority  |
| [21:0]  | -           | Reserved.  |

## 8 CLOCK CONTROL

### 8.1 Overview

The clock controller generates the clocks for the whole chip, including system clocks and all peripheral clocks. The clock controller also implements the power control function with the individually clock ON/OFF control, clock source selection and clock divider. The chip enters Power-down mode when Cortex™-M0 core executes the WFI instruction only if the SLEEPDEEP (SCR[2]) bit are both set to 1.. After that, chip enter Power-down mode and wait for waking-up interrupt source triggered to leave power-down mode. In the Power-down mode, the clock controller turns off the external 4~24 MHz high speed crystal and internal 22.1184 MHz high speed oscillator to reduce the overall system power consumption.

## 8.2 Clock Generator

The clock generator consists of 4 clock sources which are listed below:

- External 4~24 MHz high speed crystal.
- Internal 22.1184 MHz high speed oscillator.
- Programmable PLL output. (PLL source consists of external 4~24 MHz high speed crystal and internal 22.1184 MHz high speed oscillator.)
- Internal 10 kHz low speed oscillator.

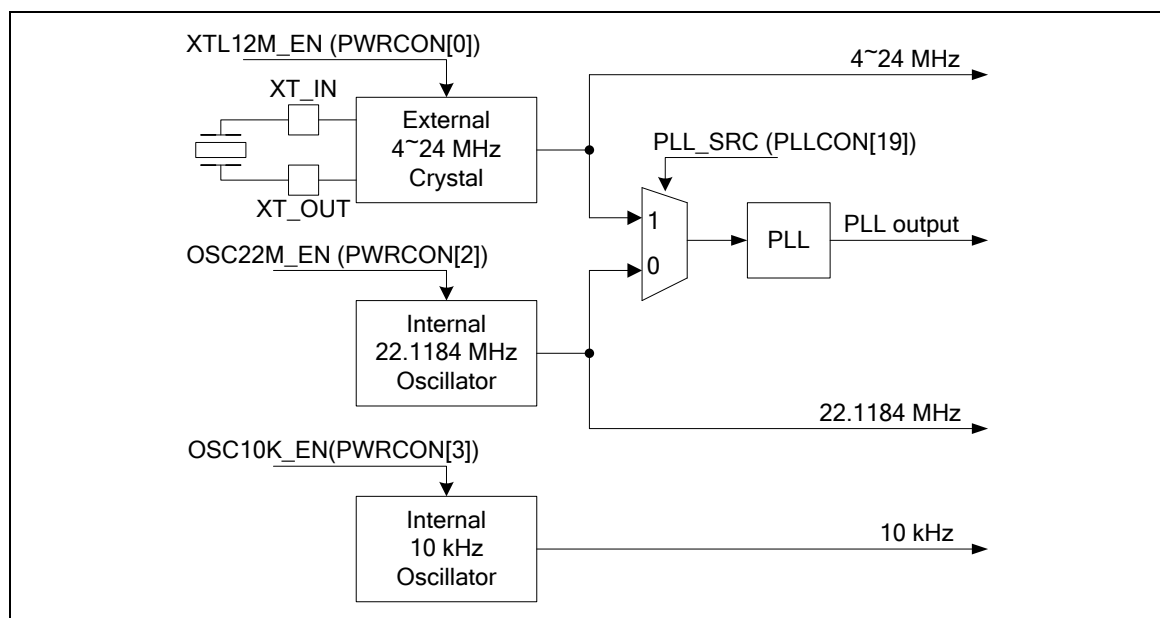


Figure 8-1 Clock Generator Block Diagram

### 8.3 System Clock & SysTick Clock

The system clock has 4 clock sources which were generated from clock generator block. The clock source switch depends on the register HCLK\_S (CLKSEL0[2:0]). The block diagram is showed in Figure 8–2.

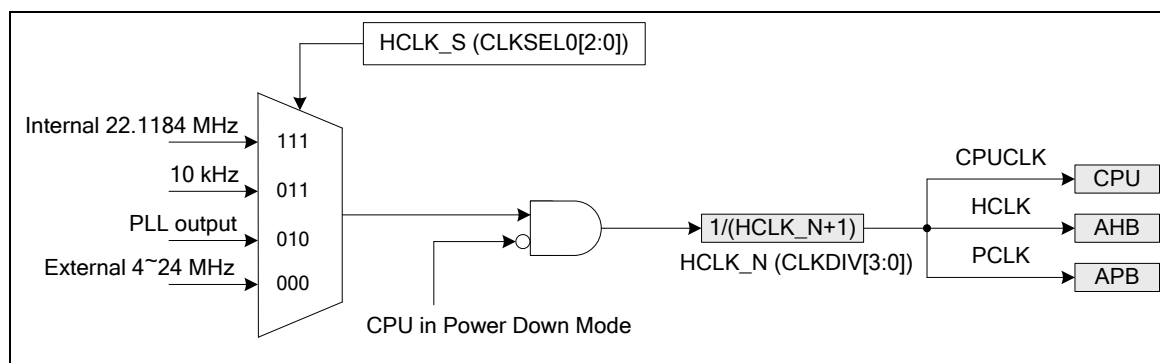


Figure 8–2 System Clock Block Diagram

The clock source of SysTick clock (STCLK) in Cortex-M0 core comes from 4 clock sources. The clock source switch depends on the setting of the register STCLK\_S (CLKSEL0[5:3]). The block diagram is showed in Figure 8–3.

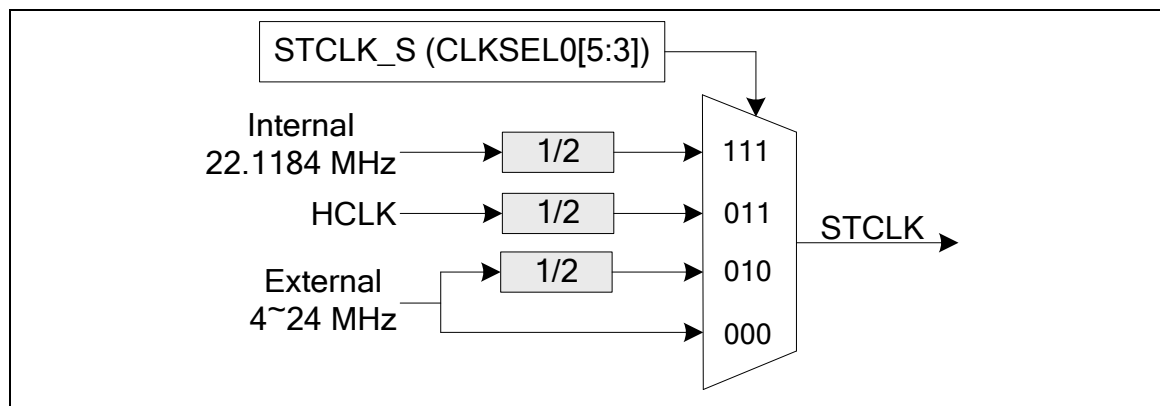


Figure 8–3 SysTick Clock Control Block Diagram

### 8.4 Peripherals Clock

The peripherals clock had different clock source switch setting which depends on the different peripheral. Please refer the CLKSEL1 and CLKSEL2 register description in 5.3.7.

### 8.5 Power-down mode (Deep Sleep Mode) Clock

When chip enters Power-down mode, system clocks, some clock sources, and some peripheral clocks will be disabled. Some clock sources and peripherals clock are still active in Power-down mode.

Clocks and peripherals which still keep active:

- Clock Generator
  - ◆ Internal 10 kHz oscillator clock.
- Peripherals Clock (when these IP adopt 10 kHz as clock source)
  - ◆ Watchdog Timer.

## 8.6 Frequency Divider Output

This device is equipped a frequency divider which is composed by 16 chained divide-by-2 shift registers. One of the 16 shift register outputs selected by a sixteen to one multiplexer is reflected to CLKO function pin. Therefore there are 16 options of divided clocks.

The output formula is  $F_{CLKO} = F_{FRQDIV\_CLK} / 2^{(N+1)}$ , where  $F_{FRQDIV\_CLK}$  is the input clock frequency,  $F_{CLKO}$  is the clock divider output frequency and N is the 4-bit value in FSEL (FRQDIV[3:0]).

When write 1 to DIVIDER\_EN (FRQDIV[4]), the chained counter starts to count. When write 0 to DIVIDER\_EN (FRQDIV[4]), the chained counter continuously runs till divided clock reaches low state and stay in low state.

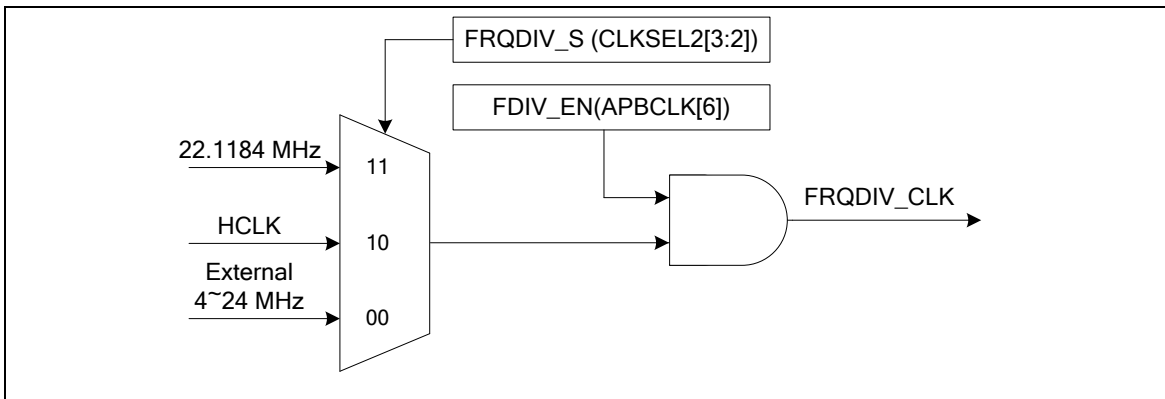


Figure 8-4 Clock Source of Frequency Divider

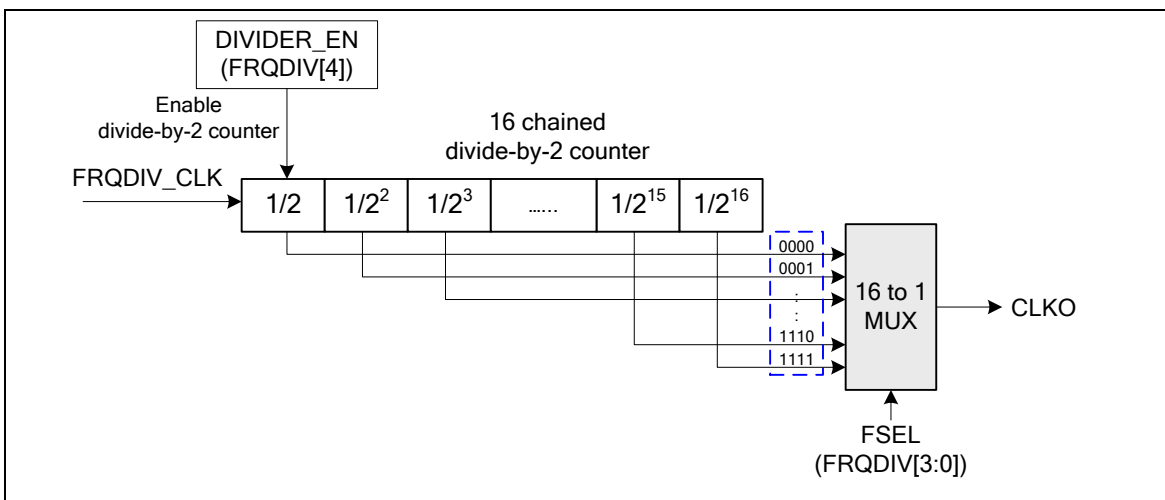


Figure 8-5 Block Diagram of Frequency Divider

### 8.7 Register Map

R: read only, W: write only, R/W: both read and write

| Register                                  | Offset      | R/W | Description                               | Reset Value |
|---|-------------|-----|---|-------------|
| CLK Base Address:<br>CLK_BA = 0x5000_0200 |             |     |   |             |
| PWRCON                                    | CLK_BA+0x00 | R/W | System Power Down Control Register        | 0x0000_000C |
| AHBCLK                                    | CLK_BA+0x04 | R/W | AHB Devices Clock Enable Control Register | 0x0000_0004 |
| APBCLK                                    | CLK_BA+0x08 | R/W | APB Devices Clock Enable Control Register | 0x0000_0001 |
| CLKSTATUS                                 | CLK_BA+0x0C | R/W | Clock status monitor Register             | 0x0000_0018 |
| CLKSEL0                                   | CLK_BA+0x10 | R/W | Clock Source Select Control Register 0    | 0x0000_000X |
| CLKSEL1                                   | CLK_BA+0x14 | R/W | Clock Source Select Control Register 1    | 0xFFFF_FFFF |
| CLKDIV                                    | CLK_BA+0x18 | R/W | Clock Divider Number Register             | 0x0000_0000 |
| CLKSEL2                                   | CLK_BA+0x1C | R/W | Clock Source Select Control Register 2    | 0xFFFF_FFFF |
| PLLCON                                    | CLK_BA+0x20 | R/W | PLL Control Register                      | 0x0005_C22E |
| FRQDIV                                    | CLK_BA+0x24 | R/W | Frequency Divider Control Register        | 0x0000_0000 |
| CLKDCR                                    | CLK_BA+0x28 | R/W | Clock Detect Control Register             | 0x0000_0000 |
| CLKDSR                                    | CLK_BA+0x2C | R/W | Clock Detect Status Register              | 0x0000_0000 |



## 8.8 Register Description

### Power Down Control Register (PWRCON)

Except the BIT[6], all the other bits are protected, program these bits need to write “59h”, “16h”, “88h” to address 0x5000\_0100 to disable register protection. Reference the register REGWRPROT at address GCR\_BA+0x100.

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| PWRCON   | CLK_BA+0x00 | R/W | System Power Down Control Register | 0x0000_000C |

|    |           |              |    |           |           |    |           |
|----|-----------|--------------|----|-----------|-----------|----|-----------|
| 31 | 30        | 29           | 28 | 27        | 26        | 25 | 24        |
| -  |           |              |    |           |           |    |           |
| 23 | 22        | 21           | 20 | 19        | 18        | 17 | 16        |
| -  |           |              |    |           |           |    |           |
| 15 | 14        | 13           | 12 | 11        | 10        | 9  | 8         |
| -  |           |              |    |           |           |    |           |
| 7  | 6         | 5            | 4  | 3         | 2         | 1  | 0         |
| -  | PD_WU_STS | PD_WU_INT_EN | -  | OSC10K_EN | OSC22M_EN | -  | XTL12M_EN |

| Bits   | Description  |
|--------|--|
| [31:7] | - Reserve  |
| [6]    | <b>PD_WU_STS</b><br>Power-down Mode Wake-up Interrupt Status<br>Set by “power down wake up event”, it indicates that resume from Power-down mode”<br>Write 1 to clear the bit to zero.             |
| [5]    | <b>PD_WU_INT_EN</b><br>Power-down Mode Wake-up Interrupt Enable (Write-protection Bit)<br>0 = Disabled.<br>1 = Enabled.<br>The interrupt will occur when both PD_WU_STS and PD_WU_INT_EN are high. |
| [4]    | - Reserved.  |
| [3]    | <b>OSC10K_EN</b><br>Internal 10 kHz Oscillator Enable (Write-protection Bit)<br>1 = 10 kHz Oscillation Enabled.<br>0 = 10 kHz Oscillation Disabled.  |
| [2]    | <b>OSC22M_EN</b><br>Internal 22.1184 MHz Oscillator Enable (Write-protection Bit)<br>1 = 22.1184 MHz Oscillation Enabled.<br>0 = 22.1184 MHz Oscillation Disabled.                                 |
| [1]    | - Reserved.  |

| Bits | Description      |   |
|------|------------------|---|
| [0]  | <b>XTL12M_EN</b> | <b>External 4~24 MHz Crystal Enable (Write-protection Bit)</b><br>The bit default value is set by flash controller user configuration register config0 [26:24]. When the default clock source is from external 4~24 MHz crystal, this bit is set to 1 automatically.<br>1 = External 4~24 MHz crystal Enabled.<br>0 = External 4~24 MHz crystal Disabled. |

| Chip Operating Mode                            | SLEEPDEEP (SCR[2]) | CPU Run WFI Instruction | Clock Disable   |
|--|--------------------|-------------------------|---|
| Normal Running Mode                            | 0                  | NO                      | All Clock are disabled by control register                  |
| IDLE Mode<br>(CPU entry Sleep Mode)            | 0                  | YES                     | Only CPU clock is disabled                                  |
| Power_down Mode<br>(CPU entry deep sleep mode) | 1                  | YES                     | Most Clock are disabled except 10K WDT clock still enabled. |

Table 8-1 Power-down mode Control Table

### AHB Devices Clock Enable Control Register (AHBCLK)

These bits for this register are used to enable/disable clock for ISP clock.

| Register | Offset      | R/W | Description                               | Reset Value |
|----------|-------------|-----|---|-------------|
| AHBCLK   | CLK_BA+0x04 | R/W | AHB Devices Clock Enable Control Register | 0x0000_0005 |

|    |    |    |        |    |        |    |    |
|----|----|----|--------|----|--------|----|----|
| 31 | 30 | 29 | 28     | 27 | 26     | 25 | 24 |
| -  |    |    |        |    |        |    |    |
| 23 | 22 | 21 | 20     | 19 | 18     | 17 | 16 |
| -  |    |    |        |    |        |    |    |
| 15 | 14 | 13 | 12     | 11 | 10     | 9  | 8  |
| -  |    |    |        |    |        |    |    |
| 7  | 6  | 5  | 4      | 3  | 2      | 1  | 0  |
| -  |    |    | DIV_EN |    | ISP_EN | -  |    |

| Bits   | Description |  |
|--------|-------------|--|
| [31:3] | -           | Reserved.  |
| [4]    | DIV_EN      | <b>Hardware Divider Controller Clock Enable Control</b><br>1 = Hardware Divider engine clock Enabled.<br>0 = Hardware Divider engine clock Disabled. |
| [3]    |             | Reserved.  |
| [2]    | ISP_EN      | <b>Flash ISP Controller Clock Enable Control</b><br>1 = Flash ISP engine clock Enabled.<br>0 = Flash ISP engine clock Disabled.                      |
| [1:0]  | -           | Reserved.  |

**APB Devices Clock Enable Control Register (APBCLK)**

These bits of this register are used to enable/disable clock for peripheral controller clocks.

| Register | Offset      | R/W | Description                               | Reset Value |
|----------|-------------|-----|---|-------------|
| APBCLK   | CLK_BA+0x08 | R/W | APB Devices Clock Enable Control Register | 0x0000_0001 |

| 31      | 30      | 29       | 28       | 27      | 26      | 25       | 24       |
|---------|---------|----------|----------|---------|---------|----------|----------|
| QE11_EN | QE10_EN | OPA_EN   | ADC_EN   | CAP1_EN | CAP0_EN | -        | CAN_EN   |
| 23      | 22      | 21       | 20       | 19      | 18      | 17       | 16       |
| -       | ACMP_EN | EPWM1_EN | EPWM0_EN | BPWM_EN | MDU_EN  | UART1_EN | UART0_EN |
| 15      | 14      | 13       | 12       | 11      | 10      | 9        | 8        |
| -       | SPI2_EN | SPI1_EN  | SPI0_EN  | -       |         |          | I2C_EN   |
| 7       | 6       | 5        | 4        | 3       | 2       | 1        | 0        |
| -       | FDIV_EN | TMR3_EN  | TMR2_EN  | TMR1_EN | TMR0_EN |          | WDT_EN   |

| Bits | Description |  |
|------|-------------|--|
| [31] | QE11_EN     | <b>QE11 Clock Enable</b><br>1 = QE11 clock Enabled.<br>0 = QE11 clock Disabled.                                  |
| [30] | QE10_EN     | <b>QE10 Clock Enable</b><br>1 = QE10 clock Enabled.<br>0 = QE10 clock Disabled.                                  |
| [29] | OPA_EN      | <b>OPA0 and OPA1 Clock Enable</b><br>1 = OPA0 and OPA1 clock Enabled.<br>0 = OPA0 and OPA1 clock Disabled.       |
| [28] | ADC_EN      | <b>ADC Clock Enable</b><br>1 = ADC clock Enabled.<br>0 = ADC clock Disabled.                                     |
| [27] | CAP1_EN     | <b>Input Capture 1 Clock Enable</b><br>1 = Input capture 1 clock Enabled.<br>0 = Input capture 1 clock Disabled. |
| [26] | CAP0_EN     | <b>Input Capture 0 Clock Enable</b><br>1 = Input capture 0 clock Enabled.<br>0 = Input capture 0 clock Disabled. |
| [25] | -           | <b>Reserved.</b>   |
| [24] | CAN_EN      | <b>CAN Bus Controller Clock Enable</b><br>1 = CAN clock Enabled.<br>0 = CAN clock Disabled.                      |

| Bits   | Description     |  |
|--------|-----------------|--|
| [23]   | -               | <b>Reserved.</b>   |
| [22]   | <b>ACMP_EN</b>  | <b>Analog comparator Clock Enable</b><br>1 = Analog comparator clock Enabled.<br>0 = Analog comparator clock Disabled. |
| [21]   | <b>EPWM1_EN</b> | <b>Enhanced PWM1 Clock Enable</b><br>1 = Enhanced PWM1 clock Enabled.<br>0 = Enhanced PWM1 clock Disabled.             |
| [20]   | <b>EPWM0_EN</b> | <b>Enhanced PWM0 Clock Enable</b><br>1 = Enhanced PWM0 clock Enabled.<br>0 = Enhanced PWM0 clock Disabled.             |
| [19]   | <b>BPWM_EN</b>  | <b>Basic PWM Clock Enable</b><br>1 = Basic PWM clock Enabled.<br>0 = Basic PWM clock Disabled.                         |
| [18]   | <b>MDU_EN</b>   | <b>MDU Clock Enable</b><br>1 = MDU clock Enabled.<br>0 = MDU clock Disabled.   |
| [17]   | <b>UART1_EN</b> | <b>UART1 clock enable.</b><br>1 = UART1 clock Enabled.<br>0 = UART1 clock Disabled.                                    |
| [16]   | <b>UART0_EN</b> | <b>UART0 Clock Enable</b><br>1 = UART0 clock Enabled.<br>0 = UART0 clock Disabled.                                     |
| [15]   | -               | <b>Reserved.</b>   |
| [14]   | <b>SPI2_EN</b>  | <b>SPI2 Clock Enable</b><br>1 = SPI2 clock Enabled.<br>0 = SPI2 clock Disabled.  |
| [13]   | <b>SPI1_EN</b>  | <b>SPI1 Clock Enable</b><br>1 = SPI1 clock Enabled.<br>0 = SPI1 clock Disabled.  |
| [12]   | <b>SPI0_EN</b>  | <b>SPI0 Clock Enable</b><br>1 = SPI0 clock Enabled.<br>0 = SPI0 clock Disabled.  |
| [11:9] | -               | <b>Reserved.</b>   |
| [8]    | <b>I2C_EN</b>   | <b>I<sup>2</sup>C Clock Enable</b><br>1 = I <sup>2</sup> C clock Enabled.<br>0 = I <sup>2</sup> C clock Disabled.      |
| [7]    | -               | <b>Reserved.</b>   |

| Bits | Description    |  |
|------|----------------|--|
| [6]  | <b>FDIV_EN</b> | <b>Frequency Divider Output Clock Enable</b><br>1 = Frequency divider output clock Enabled.<br>0 = Frequency divider output clock Disabled.  |
| [5]  | <b>TMR3_EN</b> | <b>Timer3 Clock Enable</b><br>1 = Timer3 clock Enabled.<br>0 = Timer3 clock Disabled.  |
| [4]  | <b>TMR2_EN</b> | <b>Timer2 Clock Enable</b><br>1 = Timer2 clock Enabled.<br>0 = Timer2 clock Disabled.  |
| [3]  | <b>TMR1_EN</b> | <b>Timer1 Clock Enable</b><br>1 = Timer1 clock Enabled.<br>0 = Timer1 clock Disabled.  |
| [2]  | <b>TMR0_EN</b> | <b>Timer0 Clock Enable</b><br>1 = Timer0 clock Enabled.<br>0 = Timer0 clock Disabled.  |
| [1]  | -              | <b>Reserved.</b>   |
| [0]  | <b>WDT_EN</b>  | <b>Watchdog Timer Clock Enable (Write-protection Bit)</b><br>This bit is the protected bit. It means programming this needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100.<br>1 = Watchdog Timer clock Enabled.<br>0 = Watchdog Timer clock Disabled. |

### Clock status Register (CLKSTATUS)

These bits of this register are used to monitor if the chip clock source stable or not, and whether clock switch failed.

| Register  | Offset      | R/W | Description                   | Reset Value |
|-----------|-------------|-----|-------------------------------|-------------|
| CLKSTATUS | CLK_BA+0x0C | R/W | Clock status monitor Register | 0x0000_0018 |

|             |    |    |            |            |         |    |         |
|-------------|----|----|------------|------------|---------|----|---------|
| 31          | 30 | 29 | 28         | 27         | 26      | 25 | 24      |
| -           |    |    |            |            |         |    |         |
| 23          | 22 | 21 | 20         | 19         | 18      | 17 | 16      |
| -           |    |    |            |            |         |    |         |
| 15          | 14 | 13 | 12         | 11         | 10      | 9  | 8       |
| -           |    |    |            |            |         |    |         |
| 7           | 6  | 5  | 4          | 3          | 2       | 1  | 0       |
| CLK_SW_FAIL | -  |    | OSC22M_STB | OSC10K_STB | PLL_STB | -  | XTL_STB |

| Bits   | Description  |
|--------|--|
| [31:8] | - Reserved.  |
| [7]    | <b>CLK_SW_FAIL</b><br><b>Clock Switching Fail Flag</b><br>1 = Clock switching failure.<br>0 = Clock switching success.<br>This bit is an index that if current system clock source is match as user defined at HCLK_S (CLKSEL[2:0]). When user switch system clock, the system clock source will keep old clock until the new clock is stable. During the period that waiting new clock stable, this bit will be an index shows system clock source is not match as user wanted.<br>This bit is read only. |
| [6:5]  | - Reserved.  |
| [4]    | <b>OSC22M_STB</b><br><b>Internal 22.1184M Hz Oscillator Clock Source Stable Flag</b><br>1 = Internal 22.1184M Hz oscillator clock is stable.<br>0 = Internal 22.1184M Hz oscillator clock is not stable or disabled.<br>This bit is read only.   |
| [3]    | <b>OSC10K_STB</b><br><b>Internal 10k Hz Clock Source Stable Flag</b><br>1 = Internal 10k Hz oscillator clock is stable.<br>0 = Internal 10k Hz oscillator clock is not stable or disabled.<br>This bit is read only.   |
| [2]    | <b>PLL_STB</b><br><b>PLL Clock Source Stable Flag</b><br>1 = PLL clock is stable.<br>0 = PLL clock is not stable or disabled.<br>This bit is read only.  |

| Bits | Description    |   |
|------|----------------|---|
| [1]  | -              | Reserved.   |
| [0]  | <b>XTL_STB</b> | <b>External 4~24 MHz Crystal Clock Source Stable Flag</b><br>1 = External 4~24 MHz crystal clock is stable.<br>0 = External 4~24 MHz crystal clock is not stable or disabled.<br>This bit is read only. |



**Clock Source Select Control Register 0 (CLKSEL0)**

| Register | Offset      | R/W | Description                            | Reset Value |
|----------|-------------|-----|--|-------------|
| CLKSEL0  | CLK_BA+0x10 | R/W | Clock Source Select Control Register 0 | 0x0000_000X |

|    |    |         |    |    |        |    |    |
|----|----|---------|----|----|--------|----|----|
| 31 | 30 | 29      | 28 | 27 | 26     | 25 | 24 |
| -  |    |         |    |    |        |    |    |
| 23 | 22 | 21      | 20 | 19 | 18     | 17 | 16 |
| -  |    |         |    |    |        |    |    |
| 15 | 14 | 13      | 12 | 11 | 10     | 9  | 8  |
| -  |    |         |    |    |        |    |    |
| 7  | 6  | 5       | 4  | 3  | 2      | 1  | 0  |
| -  |    | STCLK_S |    |    | HCLK_S |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:6] | -           | Reserved.   |
| [5:3]  | STCLK_S     | <b>Cortex_M0 SysTick Clock Source Selection (Write-protection Bits)</b><br>If SYST_CSR[2]=0, SysTick uses listed clock source below<br>These bits are protected bit. It means programming this bit needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100.<br>000 = Clock source from external 4~24 MHz crystal clock<br>001 = Reserved.<br>010 = Clock source from external 4~24 MHz crystal clock/2<br>011 = Clock source from HCLK/2<br>111 = Clock source from internal 22.1184 MHz oscillator clock/2<br>Others = Reserved.  |
| [2:0]  | HCLK_S      | <b>HCLK Clock Source Selection (Write-protection Bits)</b><br>1. Before clock switching, the related clock sources (both pre-select and new-select) must be turn on<br>2. The 3-bit default value is reloaded from the value of CFOSC (Config0[26:24]) in user configuration register of Flash controller by any reset. Therefore the default value is either 000b or 111b.<br>3. These bits are protected bit, It means programming this bit needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100.<br>000 = Clock source from external 4~24 MHz crystal clock<br>001 = Reserved.<br>010 = Clock source from PLL clock<br>011 = Clock source from internal 10 kHz oscillator clock<br>111 = Clock source from internal 22.1184 MHz oscillator clock<br>Others = Reserved. |

**Clock Source Select Control Register 1 (CLKSEL1)**

Before clock switching, the related clock sources (pre-selected and new-selected) must be turned on.

| Register | Offset      | R/W | Description                            | Reset Value |
|----------|-------------|-----|--|-------------|
| CLKSEL1  | CLK_BA+0x14 | R/W | Clock Source Select Control Register 1 | 0xFFFF_FFFF |

|    |        |        |        |    |    |        |    |
|----|--------|--------|--------|----|----|--------|----|
| 31 | 30     | 29     | 28     | 27 | 26 | 25     | 24 |
| -  |        |        |        |    |    | UART_S |    |
| 23 | 22     | 21     | 20     | 19 | 18 | 17     | 16 |
| -  |        |        |        |    |    |        |    |
| 15 | 14     | 13     | 12     | 11 | 10 | 9      | 8  |
| -  |        |        |        |    |    |        |    |
| 7  | 6      | 5      | 4      | 3  | 2  | 1      | 0  |
| -  | SPI2_S | SPI1_S | SPI0_S |    |    | WDT_S  |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:26] | -           | Reserved.   |
| [25:24] | UART_S      | <b>UART Clock Source Selection</b><br>00 = Clock source from external 4~24 MHz crystal clock<br>01 = Clock source from PLL clock<br>10 = Reserved.<br>11 = Clock source from internal 22.1184 MHz oscillator clock  |
| [23:7]  | -           | Reserved.   |
| [6]     | SPI2_S      | <b>SPI2 Clock Source Selection</b><br>1 = Clock source from HCLK<br>0 = Clock source from PLL clock   |
| [5]     | SPI1_S      | <b>SPI1 Clock Source Selection</b><br>1 = Clock source from HCLK<br>0 = Clock source from PLL clock   |
| [4]     | SPI0_S      | <b>SPI0 Clock Source Selection</b><br>1 = Clock source from HCLK<br>0 = Clock source from PLL clock   |
| [3:2]   | Reserved    | Reserved  |
| [1:0]   | WDT_S       | <b>Watchdog Timer Clock Source Selection (Write-protection Bits)</b><br>These bits are protected bits, programing them need to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100. |

|  |  |   |
|--|--|---|
|  |  | 00 = Clock source from HCLK/128 clock<br>01 = Clock source from HCLK/512 clock<br>10 = Clock source from HCLK/2048 clock<br>11 = Clock source from internal 10 kHz oscillator clock |
|--|--|---|

### Clock Source Select Control Register 2 (CLKSEL2)

Before clock switching, the related clock sources (pre-selected and new-selected) must be turned on.

| Register | Offset      | R/W | Description                            | Reset Value |
|----------|-------------|-----|--|-------------|
| CLKSEL2  | CLK_BA+0x1C | R/W | Clock Source Select Control Register 2 | 0xFFFF_FFFF |

|    |    |    |    |          |    |        |    |
|----|----|----|----|----------|----|--------|----|
| 31 | 30 | 29 | 28 | 27       | 26 | 25     | 24 |
| -  |    |    |    |          |    |        |    |
| 23 | 22 | 21 | 20 | 19       | 18 | 17     | 16 |
| -  |    |    |    |          |    | WWDT_S |    |
| 15 | 14 | 13 | 12 | 11       | 10 | 9      | 8  |
| -  |    |    |    |          |    |        |    |
| 7  | 6  | 5  | 4  | 3        | 2  | 1      | 0  |
| -  |    |    |    | FRQDIV_S |    | -      |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:18] | -           | Reserved.  |
| [17:16] | WWDT_S      | <b>Window Watchdog Timer Clock Source Selection</b><br>00 = Reserved<br>01 = Reserved<br>10 = Clock source from HCLK/2048 clock<br>11 = Clock source from internal 10 kHz low speed oscillator clock                   |
| [15:4]  | -           | Reserved.  |
| [3:2]   | FRQDIV_S    | <b>Clock Divider Clock Source Selection</b><br>00 = Clock source from external 4~24 MHz crystal clock<br>01 = Reserved.<br>10 = Clock source from HCLK<br>11 = Clock source from internal 22.1184 MHz oscillator clock |
| [1:0]   | -           | Reserved.  |

### Clock Divider Register (CLKDIV)

| Register | Offset      | R/W | Description                   | Reset Value |
|----------|-------------|-----|-------------------------------|-------------|
| CLKDIV   | CLK_BA+0x18 | R/W | Clock Divider Number Register | 0x0000_0000 |

|       |    |    |    |        |    |    |    |
|-------|----|----|----|--------|----|----|----|
| 31    | 30 | 29 | 28 | 27     | 26 | 25 | 24 |
| -     |    |    |    |        |    |    |    |
| 23    | 22 | 21 | 20 | 19     | 18 | 17 | 16 |
| ADC_N |    |    |    |        |    |    |    |
| 15    | 14 | 13 | 12 | 11     | 10 | 9  | 8  |
| -     |    |    |    | UART_N |    |    |    |
| 7     | 6  | 5  | 4  | 3      | 2  | 1  | 0  |
| -     |    |    |    | HCLK_N |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:24] | -           | Reserved.   |
| [23:16] | ADC_N       | ADC clock divider.<br>The ADC clock frequency = (ADC clock source frequency) / (ADC_N + 1)    |
| [15:12] | -           | Reserved.   |
| [11:8]  | UART_N      | UART Clock Divider<br>The UART clock frequency = (UART clock source frequency) / (UART_N + 1) |
| [7:4]   | -           | Reserved.   |
| [3:0]   | HCLK_N      | HCLK Clock Divider<br>The HCLK clock frequency = (HCLK clock source frequency) / (HCLK_N + 1) |

### PLL Control Register ( PLLCON )

The PLL reference clock input is from the external 4~24 MHz crystal clock input or from the internal 22.1184 MHz oscillator. These registers are use to control the PLL output frequency and PLL operating mode

| Register | Offset      | R/W | Description          | Reset Value |
|----------|-------------|-----|----------------------|-------------|
| PLLCON   | CLK_BA+0x20 | R/W | PLL Control Register | 0x0005_C22E |

|        |    |       |         |         |    |    |       |
|--------|----|-------|---------|---------|----|----|-------|
| 31     | 30 | 29    | 28      | 27      | 26 | 25 | 24    |
| -      |    |       |         |         |    |    |       |
| 23     | 22 | 21    | 20      | 19      | 18 | 17 | 16    |
| -      |    |       | FCO_SEL | PLL_SRC | OE | BP | PD    |
| 15     | 14 | 13    | 12      | 11      | 10 | 9  | 8     |
| OUT_DV |    | IN_DV |         |         |    |    | FB_DV |
| 7      | 6  | 5     | 4       | 3       | 2  | 1  | 0     |
| FB_DV  |    |       |         |         |    |    |       |

| Bits    | Description |   |
|---------|-------------|---|
| [31:20] | -           | Reserved.   |
| [20]    | FCO_SEL     | <b>PLL FCO Selection</b><br>1 = When the FCO frequency range between 200MHz to 500MHz, this bit should be set as 1<br>0 = When the FCO frequency range between 100MHz and 200MHz, this bit should be set as 0 |
| [19]    | PLL_SRC     | <b>PLL Source Clock Selection</b><br>1 = PLL source clock from internal 22.1184 MHz oscillator<br>0 = PLL source clock from external 4~24 MHz crystal   |
| [18]    | OE          | <b>PLL OE (FOUT enable) Pin Control</b><br>0 = PLL FOUT enable<br>1 = PLL FOUT is fixed low   |
| [17]    | BP          | <b>PLL Bypass Control</b><br>0 = PLL is in normal mode (default)<br>1 = PLL clock output is same as clock input (XTALin)  |
| [16]    | PD          | <b>Power-down Mode</b><br>0 = PLL is in normal mode<br>1 = PLL is in power-down mode (default)  |
| [15:14] | OUT_DV      | <b>PLL Output Divider Control Pins</b><br>Refer to the formulas below the table.  |

| Bits   | Description |   |
|--------|-------------|---|
| [13:9] | IN_DV       | PLL Input Divider Control Pins<br>Refer to the formulas below the table.    |
| [8:0]  | FB_DV       | PLL Feedback Divider Control Pins<br>Refer to the formulas below the table. |

### Output Clock Frequency Setting

$$F_{OUT} = F_{IN} \times \frac{NF}{NR} \times \frac{1}{NO}$$

Constrain:

$$1. \quad 3.2MHz < F_{IN} < 150MHz$$

$$2. \quad 800KHz < \frac{F_{IN}}{2 * NR} < 8MHz$$

$$200MHz < FCO = F_{IN} * \frac{NF}{NR} < 500MHz, (FCO\_SEL = 1)$$

$$3. \quad 100MHz < FCO = F_{IN} * \frac{NF}{NR} < 200MHz, (FCO\_SEL = 0)$$

| Symbol    | Description  |
|-----------|--|
| $F_{OUT}$ | Output Clock Frequency   |
| $F_{IN}$  | Input (Reference) Clock Frequency  |
| $NR$      | Input Divider (IN_DV + 2)  |
| $NF$      | Feedback Divider (FB_DV + 2)   |
| $NO$      | OUT_DV = "00" : NO = 1<br>OUT_DV = "01" : NO = 2<br>OUT_DV = "10" : NO = 2<br>OUT_DV = "11" : NO = 4 |

### Default Frequency Setting

The default value : 0xC22E

$F_{IN} = 12 \text{ MHz}$

$NR = (1+2) = 3$

$NF = (46+2) = 48$

$NO = 4$

$F_{OUT} = 12/4 \times 48 \times 1/3 = 48MHz$

### Frequency Divider Control Register (FRQDIV)

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| FRQDIV   | CLK_BA+0x24 | R/W | Frequency Divider Control Register | 0x0000_0000 |

|    |    |      |            |      |    |    |    |
|----|----|------|------------|------|----|----|----|
| 31 | 30 | 29   | 28         | 27   | 26 | 25 | 24 |
| -  |    |      |            |      |    |    |    |
| 23 | 22 | 21   | 20         | 19   | 18 | 17 | 16 |
| -  |    |      |            |      |    |    |    |
| 15 | 14 | 13   | 12         | 11   | 10 | 9  | 8  |
| -  |    |      |            |      |    |    |    |
| 7  | 6  | 5    | 4          | 3    | 2  | 1  | 0  |
| -  |    | DIV1 | DIVIDER_EN | FSEL |    |    |    |

| Bits   | Description |  |
|--------|-------------|--|
| [31:6] | -           | Reserved.  |
| [5]    | DIV1        | <b>Frequency Divider Divide 1 Enable</b><br>0 = Frequency divider will output clock with source frequency divide by FSEL<br>1 = Frequency divider will output clock with source frequency.   |
| [4]    | DIVIDER_EN  | <b>Frequency Divider Enable Bit</b><br>0 = Frequency divider Disabled.<br>1 = Frequency divider Enabled.   |
| [3:0]  | FSEL        | <b>Frequency Divider Output Selection Bits</b><br>The output formula is:<br>$F_{CLKO} = F_{FRQDIV\_CLK} / 2^{(N+1)}$ where $F_{FRQDIV\_CLK}$ is the input clock frequency, $F_{CLKO}$ is the clock divider output frequency and N is the 4-bit value in FSEL[3:0]. |



### Clock Detect Control Register (CKDCR)

| Register | Offset      | R/W | Description                   | Reset Value |
|----------|-------------|-----|-------------------------------|-------------|
| CKDCR    | CLK_BA+0x28 | R/W | Clock Detect Control Register | 0x0000_0000 |

|    |    |    |    |    |    |         |        |
|----|----|----|----|----|----|---------|--------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25      | 24     |
| -  |    |    |    |    |    |         |        |
| 23 | 22 | 21 | 20 | 19 | 18 | 17      | 16     |
| -  |    |    |    |    |    |         |        |
| 15 | 14 | 13 | 12 | 11 | 10 | 9       | 8      |
| -  |    |    |    |    |    |         |        |
| 7  | 6  | 5  | 4  | 3  | 2  | 1       | 0      |
| -  |    |    |    |    |    | CKD_IEN | CKD_EN |

| Bits   | Description |   |
|--------|-------------|---|
| [31:2] | -           | Reserved.   |
| [1]    | CKD_IEN     | <b>Clock Detect Interrupt Enable</b><br>0 = Clock detect interrupt Disabled.<br>1 = Clock detect interrupt Enabled. |
| [0]    | CKD_EN      | <b>Clock Detect Enable</b><br>0 = Clock detect Disabled.<br>1 = Clock detect Enabled                                |

### Clock Detect Status Register (CKDSR)

| Register | Offset      | R/W | Description                  | Reset Value |
|----------|-------------|-----|------------------------------|-------------|
| CKDSR    | CLK_BA+0x2C | R/W | Clock Detect Status Register | 0x0000_0000 |

|    |    |    |    |    |    |        |           |
|----|----|----|----|----|----|--------|-----------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25     | 24        |
| -  |    |    |    |    |    |        |           |
| 23 | 22 | 21 | 20 | 19 | 18 | 17     | 16        |
| -  |    |    |    |    |    |        |           |
| 15 | 14 | 13 | 12 | 11 | 10 | 9      | 8         |
| -  |    |    |    |    |    |        |           |
| 7  | 6  | 5  | 4  | 3  | 2  | 1      | 0         |
| -  |    |    |    |    |    | CKSTOP | CKSTOP_IF |

| Bits   | Description |   |
|--------|-------------|---|
| [31:2] | -           | Reserved.   |
| [1]    | CKSTOP      | <b>Clock Stop Status</b><br>This bit reflect clock detect circuit result  |
| [0]    | CKSTOP_IF   | <b>Clock Stop Interrupt Flag</b><br>When system clock stop is detected, this bit will be set. Software can write 1 to clear this bit. |

## 9 GENERAL PURPOSE I/O

### 9.1 Overview

The NuMicro™ NM15xx Series has up to 82 General Purpose I/O pins to be shared with other function pins depending on the chip configuration. These 82 pins are arranged in 10 ports named as P0, P1, P2, P3, P4, P5, P6, P7, P8, P9 and PA. The P0/1/2/3/4/5/6/7/8/9 port has the maximum of 8 pins and PA port has the maximum of 2 pins. Each of the 82 pins is independent and has the corresponding register bits to control the pin mode function and data.

The I/O type of each of I/O pins can be configured by software individually as input, output, open-drain or Quasi-bidirectional mode. After reset, the I/O mode of all pins are stay at input mode. In Quasi-bidirectional mode, I/O pin has a very weak individual pull-up resistor which is about 110~300 K $\Omega$  for  $V_{DD}$  is from 5.0 V to 2.5 V.

### 9.2 Features

- Four I/O modes:
  - ◆ Quasi-bidirectional
  - ◆ Push-Pull output
  - ◆ Open-Drain output
  - ◆ Input only with high impedance
- TTL/Schmitt trigger input selectable
- I/O pin configured as interrupt source with edge/level setting
- I/O pin internal pull-up resistor enabled only in Quasi-bidirectional I/O mode
- Enabling pin interrupt function will also enable the pin wake-up function

## 9.3 Functional Description

### 9.3.1 Input Only Mode

Setting PMDm[1:0] in Pn\_PMD to 00b as the Pn[m] pin is in Input mode and the I/O pin is in tri-state (high impedance) without output drive capability. The Pn\_PIN value reflects the status of the corresponding port pins.

### 9.3.2 Push-Pull Output Mode

Setting PMDm[1:0] in Pn\_PMD to 01b as the Pn[m] pin is in Push-pull Output mode and the I/O pin supports digital output function with source/sink current capability. The bit value of Pn\_DOUT[m] is driven on the pin.

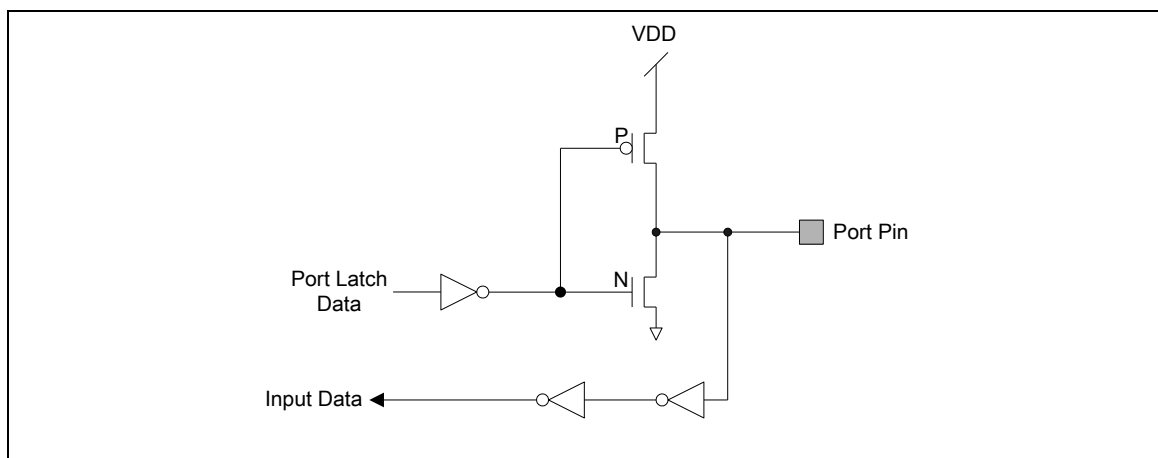


Figure 9–1 Push-Pull Output

### 9.3.3 Open-Drain Mode

Setting PMDm[1:0] in Pn\_PMD to 10b as the Pn[m] pin is in Open-drain mode and the digital output function of I/O pin supports only sink current capability, an additional pull-up resistor is needed for driving high state. If the bit value of Pn\_DOUT[m] is 0, the pin drive a “low” output on the pin. If the bit value of Pn\_DOUT[m] is 1, the pin output drives high that is controlled by external pull high resistor.

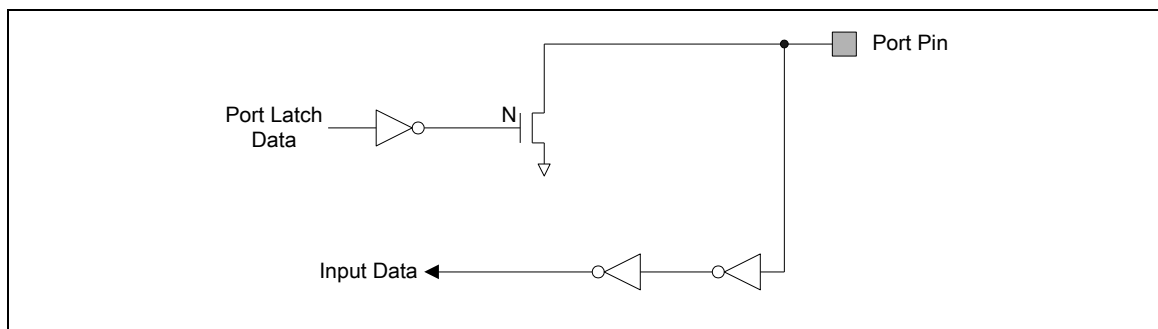


Figure 9–2 Open-Drain Output

### 9.3.4 Quasi Bi-directional Mode

Setting PMDm[1:0] in Pn\_PMD to 11b as the Pn[m] pin is in Quasi-bidirectional mode and the I/O pin supports digital output and input function at the same time but the source current is only up to hundreds uA. Before the digital input function is performed, value of Pn\_DOUT[m] must be set to 1. The quasi-bidirectional output is common on the 80C51 and most of its derivatives. If the bit value of Pn\_DOUT[m] is 0, the pin drive a “low” output on the pin. If the bit value of Pn\_DOUT[m] is 1, the pin will check the pin value. If pin value is high, no action takes. If pin state is low, then pin will drive strong high with 2 clock cycles on the pin and then disable the strong output drive and then the pin status is control by internal pull-up resistor. Note that the source current capability in quasi-bidirectional mode is only about 200uA to 30uA if VDD is from 5.0V to 2.5V.

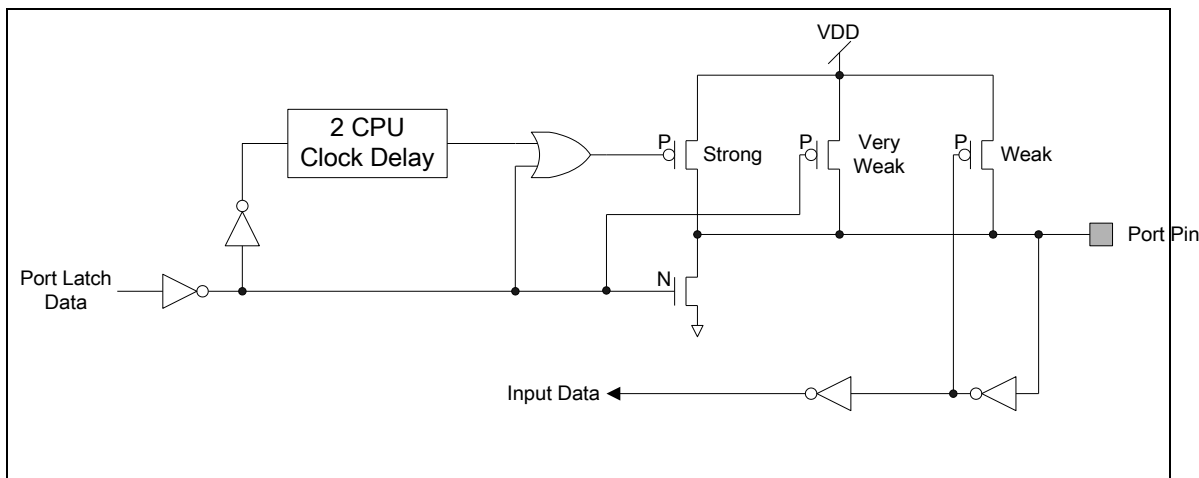


Figure 9–3 Quasi bi-directional I/O Mode

## 9.4 Enhanced PWM Port Output Driving Control

There are two enhanced PWM units each unit has six output pins in this device. The PWM port outputs are P0.0~P0.5 and P1.0~P1.5 for unit 0 and unit 1, respectively.

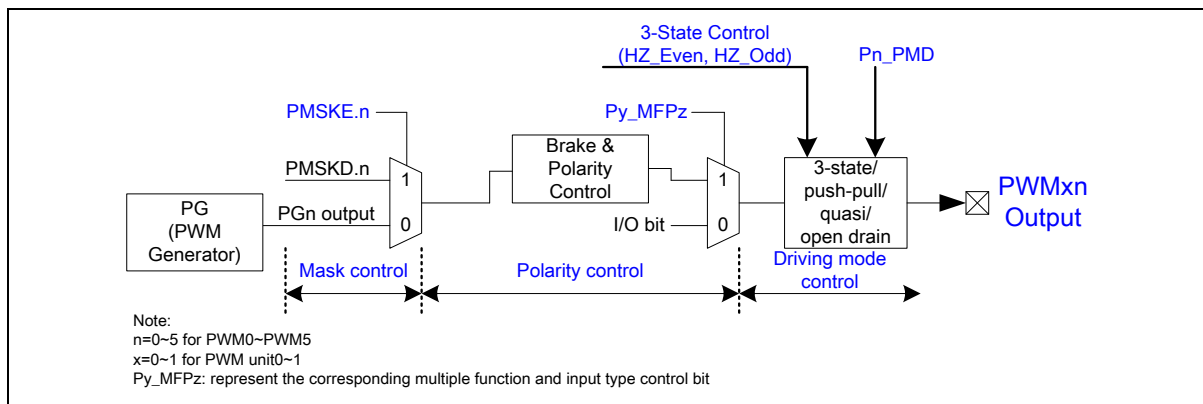


Figure 9-4 PWM Output Driving Control

The driving type of PWM output ports can be initialized as Tri-state type or other type dependent with Pn\_PMD register setting after any reset. As show in the above diagram, PWM output structures are controllable through config-bits, register bits PWMPOEN[HZ\_Even0/1, HZ\_Odd0/1] and Pn\_PMD mode registers.

| HZ_Even/HZ_Odd<br>(In PWMPOEN Register) | Even/Odd PWM Outputs<br>Drive Mode |
|---|------------------------------------|
| 0                                       | Depend on Pn_PMD                   |
| 1                                       | Driving mode = Hi-Z.               |

**Note:** Register bits for HZ\_Even and HZ\_Odd are latched from config0 during all reset.

### 9.5 Register Map

R: read only, W: write only, R/W: both read and write

| Register  | Offset                         | R/W | Description                          | Reset Value |
|---|--------------------------------|-----|--------------------------------------|-------------|
| <b>GPIO Base Address:</b><br><b>GPx_BA = 0x5000_4000 + (0x40 * x)</b><br>x = 0, 1..10<br><b>GPIOB_BA = 0x5000_4300</b><br>n = 0, 1..10<br>m = 0, 1..7 |                                |     |                                      |             |
| <b>Px_PMD</b>   | GPx_BA+0x00                    | R/W | GPIO Port Bit Mode Control           | 0x0000_0000 |
| <b>Px_OFFD</b>  | GPx_BA+0x04                    | R/W | GPIO Port Bit Off Digital Enable     | 0x0000_0000 |
| <b>Px_DOUT</b>  | GPx_BA+0x08                    | R/W | GPIO Port Data Output                | 0x0000_00FF |
| <b>Px_DMASK</b>   | GPx_BA+0x0C                    | R/W | GPIO Port Data Output Write Mask     | 0x0000_0000 |
| <b>Px_PIN</b>   | GPx_BA+0x10                    | R   | GPIO Port Pin Value                  | 0x0000_00XX |
| <b>Px_DBEN</b>  | GPx_BA+0x14                    | R/W | GPIO Port De-bounce Enable           | 0x0000_0000 |
| <b>Px_IMD</b>   | GPx_BA+0x18                    | R/W | GPIO Port Interrupt Mode Select      | 0x0000_0000 |
| <b>Px_IEN</b>   | GPx_BA+0x1C                    | R/W | GPIO Port Interrupt Enable           | 0x0000_0000 |
| <b>Px_ISF</b>   | GPx_BA+0x20                    | R/W | GPIO Port Interrupt Source Flag      | 0xFFFF_FFFF |
| <b>DBNCECON</b>   | GP0_BA+0x2E0                   | R/W | External Interrupt De-bounce Control | 0x0000_0000 |
| <b>PWMPDEN</b>  | GP0_BA+0x2E4                   | R/W | PWM Port Output Enable               | 0x0000_00XX |
| <b>Pn_m</b><br>n=0,1..10<br>m=0,1..7  | GPIOB_BA+(0x20 * n + 0x04 * m) | R/W | GPIO Port n bit m I/O value          | 0x0000_000X |

## 9.6 Register Description

### GPIO Port I/O Mode Control (Px\_PMD, x = 0~10(0x0A))

| Register | Offset      | R/W | Description                | Reset Value |
|----------|-------------|-----|----------------------------|-------------|
| Px_PMD   | GPx_BA+0x00 | R/W | GPIO Port Bit Mode Control | 0x0000_0000 |

|      |    |      |    |      |    |      |    |
|------|----|------|----|------|----|------|----|
| 31   | 30 | 29   | 28 | 27   | 26 | 25   | 24 |
| -    |    |      |    |      |    |      |    |
| 23   | 22 | 21   | 20 | 19   | 18 | 17   | 16 |
| -    |    |      |    |      |    |      |    |
| 15   | 14 | 13   | 12 | 11   | 10 | 9    | 8  |
| PMD7 |    | PMD6 |    | PMD5 |    | PMD4 |    |
| 7    | 6  | 5    | 4  | 3    | 2  | 1    | 0  |
| PMD3 |    | PMD2 |    | PMD1 |    | PMD0 |    |

| Bits                   | Description |   |
|------------------------|-------------|---|
| [31:16]                | -           | Reserved.   |
| [2m+1:2m]<br>m=0,1..15 | PMDm        | <b>Port x bit m Mode Control</b><br>Determine each I/O mode of Px pins.<br>00 = GPIO port [n] pin is in Input mode<br>01 = GPIO port [n] pin is in Push-pull Output mode<br>10 = GPIO port [n] pin is in Open-drain Output mode<br>11 = GPIO port [n] pin is in Quasi-bidirectional mode<br><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9. |



### GPIO Port Bit OFF Digital Enable (Px\_OFFD, x = 0~10(0x0A))

| Register | Offset      | R/W | Description                      | Reset Value |
|----------|-------------|-----|----------------------------------|-------------|
| Px_OFFD  | GPx_BA+0x04 | R/W | GPIO Port Bit Off Digital Enable | 0x0000_0000 |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -         |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| OFFD[7:0] |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -         |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| -         |    |    |    |    |    |    |    |

| Bits               | Description |  |
|--------------------|-------------|--|
| [31:24]            | -           | Reserved.  |
| [m+16]<br>m=0,1..7 | OFFD[m]     | <p><b>Port x bit m off digital input path.</b></p> <p>Each of these bits is used to turn off the digital input path of port n bit m pin. If input is analog signal, users can turn off digital input path to avoid input current leakage.</p> <p>1 = I/O digital input path Disabled (digital input tied to low).</p> <p>0 = I/O digital input path Enabled.</p> <p><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9</p> |
| [15:0]             | -           | Reserved.  |

### GPIO Port Data Output (Px\_DOUT, x = 0~10(0x0A))

| Register | Offset      | R/W | Description           | Reset Value |
|----------|-------------|-----|-----------------------|-------------|
| Px_DOUT  | GPx_BA+0x08 | R/W | GPIO Port Data Output | 0x0000_00FF |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -         |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -         |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -         |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DOUT[7:0] |    |    |    |    |    |    |    |

| Bits            | Description |  |
|-----------------|-------------|--|
| [31:8]          | -           | Reserved.  |
| [m]<br>m=0,1..7 | DOUT[m]     | <p><b>Port x bit m output.</b></p> <p>Each of these bits control the status of port n bit m when this pin is configured as output, open-drain, or quasi bi-directional mode.</p> <p>1 = GPIO port [0/1/2/3/4/5/6/7/8/9/A] Pin[m] will drive High if the GPIO pin is configured as Push-pull output or Quasi-bidirectional mode.</p> <p>0 = GPIO port [0/1/2/3/4/5/6/7/8/9/A] Pin[m] will drive Low if the GPIO pin is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode.</p> <p><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9</p> |

**GPIO Port Data Output Write Mask (Px\_DMASK, x = 0~10(0x0A))**

| Register | Offset      | R/W | Description                      | Reset Value |
|----------|-------------|-----|----------------------------------|-------------|
| Px_DMASK | GPx_BA+0x0C | R/W | GPIO Port Data Output Write Mask | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -          |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DMASK[7:0] |    |    |    |    |    |    |    |

| Bits            | Description |  |
|-----------------|-------------|--|
| [31:8]          | -           | Reserved.  |
| [m]<br>m=0,1..7 | DMASK[m]    | <p><b>Port x bit m data output write mask</b></p> <p>These bits are used to protect the corresponding register of Pn_DOUT[m]. When set the DMASK[m] to 1, the writing to Pn_DOUT[m] bit is ignored. The write to port pin latch is masked.</p> <p>1 = Corresponding Pn_DOUT[m] bit protected.</p> <p>0 = Corresponding Pn_DOUT[m] bit can be updated</p> <p><b>Note:</b> This function only protects the corresponding Px_DOUT[m] bit, and will not protect the corresponding bit control register (P0m, P1m, P2m, P3m, P4m, P5m, P6m, P7m, P8m, P9m and PAm).</p> <p><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9</p> |

### GPIO Port Pin Value (Px\_PIN, x = 0~10(0x0A))

| Register | Offset      | R/W | Description         | Reset Value |
|----------|-------------|-----|---------------------|-------------|
| Px_PIN   | GPx_BA+0x10 | R   | GPIO Port Pin Value | 0x0000_00XX |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -        |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -        |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -        |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PIN[7:0] |    |    |    |    |    |    |    |

| Bits            | Description |   |
|-----------------|-------------|---|
| [31:8]          | -           | Reserved.   |
| [m]<br>m=0,1..7 | PIN[m]      | <b>Port x Bit m Pin Value</b><br>Each bit of the register reflects the actual status of the respective port pin. If bit is 1, it indicates the corresponding pin status is high, else the pin status is low<br><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9 |

**GPIO Port De-bounce Enable (Px\_DBEN, x = 0~10(0x0A))**

| Register | Offset      | R/W | Description                | Reset Value |
|----------|-------------|-----|----------------------------|-------------|
| Px_DBEN  | GPx_BA+0x14 | R/W | GPIO Port De-bounce Enable | 0x0000_0000 |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -         |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -         |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -         |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DBEN[7:0] |    |    |    |    |    |    |    |

| Bits            | Description |  |
|-----------------|-------------|--|
| [31:8]          | -           | Reserved.  |
| [m]<br>m=0,1..7 | DBEN[m]     | <p><b>Port x Bit m Input De-bounce Enable</b></p> <p>DBEN[m] is used to enable the de-bounce function for each corresponding bit. DBEN[m] is valid for “edge-triggered” interrupt only and is ignored for “level triggered” interrupt.</p> <p>If the input signal pulse width can't be sampled by continuous two de-bounce sample cycle The input signal transition is seen as the signal bounce and will not trigger the interrupt. The de-bounce clock is controlled by DBNCECON register.</p> <p>1 = Bit[m] de-bounce function Enabled.<br/>0 = Bit[m] de-bounce function Disabled.</p> <p><b>Note:</b> Max. m = 1 for PA; Max. n = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9</p> |

**GPIO Port Interrupt Mode Select (Px\_IMD, x = 0~10(0x0A))**

| Register | Offset      | R/W | Description                     | Reset Value |
|----------|-------------|-----|---------------------------------|-------------|
| Px_IMD   | GPx_BA+0x18 | R/W | GPIO Port Interrupt Mode Select | 0x0000_0000 |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -        |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -        |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -        |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| IMD[7:0] |    |    |    |    |    |    |    |

| Bits            | Description  |
|-----------------|--|
| [31:8]          | - Reserved.  |
| [m]<br>m=0,1..7 | <p><b>Port x bit m Edge or Level Triggered Interrupt Control</b></p> <p>IMD[m] decides the pin interrupt triggered by level or edge. If the interrupt is by edge trigger, the trigger source can be controlled by de-bounce. If the interrupt is by level trigger, the input source is sampled by one HCLK clock and generates the interrupt.</p> <p>1 = Level triggered interrupt.<br/>0 = Edge triggered interrupt.</p> <p>If set pin as the level trigger interrupt, then only one level can be set on the registers Pn_IEN. If set both the level to trigger interrupt, the setting is ignored and no interrupt will occur</p> <p>The de-bounce function is valid for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored.</p> <p><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9</p> |

**GPIO Port Interrupt Enable (Px\_IEN, x = 0~10(0x0A))**

| Register | Offset      | R/W | Description                | Reset Value |
|----------|-------------|-----|----------------------------|-------------|
| Px_IEN   | GPx_BA+0x1C | R/W | GPIO Port Interrupt Enable | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| IR_EN[7:0] |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -          |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| IF_EN[7:0] |    |    |    |    |    |    |    |

| Bits               | Description |   |
|--------------------|-------------|---|
| [31:24]            | -           | Reserved.   |
| [m+16]<br>m=0,1..7 | IR_EN[m]    | <b>Port x Bit m Interrupt Enabled for Rising Edge or High Level Input</b><br>IR_EN[m] enables the interrupt for each of the corresponding input of Port x. Set bit to 1 also enable the pin wakeup function.<br>1 = Port n bit m high-level or rising edge interrupt Enabled.<br>0 = Port n bit m high-level or rising edge interrupt Disabled.<br><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9 |
| [15:8]             | -           | Reserved.   |
| [m]<br>m=0,1..7    | IF_EN[m]    | <b>Port x Bit m Interrupt Enabled for Falling Edge or Low Level Input</b><br>IF_EN[n] enables the interrupt for each of the corresponding input of Port x. Set bit to 1 also enable the pin wakeup function.<br>1 = Port n bit m low-level or falling edge interrupt Enabled.<br>0 = Port n bit m low-level or falling edge interrupt Disabled.<br><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9 |

### GPIO Port Interrupt Source Flag (Px\_ISF, x = 0~10(0x0A))

| Register | Offset      | R/W | Description                     | Reset Value |
|----------|-------------|-----|---------------------------------|-------------|
| Px_ISF   | GPx_BA+0x20 | R/W | GPIO Port Interrupt Source Flag | 0xFFFF_XXXX |

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -           |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -           |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -           |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| IF_ISF[7:0] |    |    |    |    |    |    |    |

| Bits            | Description  |
|-----------------|--|
| [31:8]          | - Reserved.  |
| [m]<br>m=0,1..7 | <p><b>IF_ISF [m]</b></p> <p><b>Port x Bit m Trigger Source Indicator</b></p> <p>These bits are edge/level trigger event indicators of each pin of Port x, if GPG0_INT or GPG1_INT enable bit is set, the corresponding interrupt service routine will be served. Note that P3.2 and P3.3 will <b>only</b> vector to INT0_INT and INT1_INT ISRs if its own interrupt is enabled.</p> <p>Read :</p> <p>1 = Indicates Port x bit m detects a trigger event.</p> <p>0 = No interrupt at Port x.</p> <p>Write :</p> <p>1= Clear the corresponding pending interrupt.</p> <p>0= No effect.</p> <p><b>Note:</b> Max. m = 1 for PA; Max. m = 7 for P0/P1/P2/P3/P4/P5/P6/P7/P8/P9</p> |



**External Interrupt De-bounce Control (DBNCECON)**

| Register | Offset       | R/W | Description                          | Reset Value |
|----------|--------------|-----|--------------------------------------|-------------|
| DBNCECON | GP0_BA+0x2E0 | R/W | External Interrupt De-bounce Control | 0x0000_0000 |

|    |    |         |          |          |    |    |    |
|----|----|---------|----------|----------|----|----|----|
| 31 | 30 | 29      | 28       | 27       | 26 | 25 | 24 |
| -  |    |         |          |          |    |    |    |
| 23 | 22 | 21      | 20       | 19       | 18 | 17 | 16 |
| -  |    |         |          |          |    |    |    |
| 15 | 14 | 13      | 12       | 11       | 10 | 9  | 8  |
| -  |    |         |          |          |    |    |    |
| 7  | 6  | 5       | 4        | 3        | 2  | 1  | 0  |
| -  |    | ICLK_ON | DBCLKSRC | DBCLKSEL |    |    |    |

| Bits | Description |   |
|------|-------------|---|
| [5]  | ICLK_ON     | <b>Interrupt Clock On Mode</b><br>1 = All I/O pins edge detection circuit is always active after reset.<br>0 = Edge detection circuit is active only if I/O pin corresponding Pn_IEN bit is set to 1.<br>It is recommended to turn off this bit to save system power if no special application concern. |
| [4]  | DBCLKSRC    | <b>De-bounce Counter Clock Source Selection</b><br>1 = De-bounce counter clock source is the internal 10 kHz low speed oscillator.<br>0 = De-bounce counter clock source is the HCLK.   |

| Bits  | Description                                    |   |   |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|-------|--|---|---|-------------|-----|---|-----|--|-----|--|-----|--|-----|---|-----|---|-----|---|-----|--|-----|--|-----|--|-----|---|-----|--|-----|---|-----|---|-----|---|-----|--|
| [3:0] | DBCLKSEL                                       | De-bounce sampling cycle selection.   |   |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | <table><tr><th>DBCLKSEL</th><th>Description</th></tr><tr><td>0x0</td><td>Sample interrupt input once per 1 clock</td></tr><tr><td>0x1</td><td>Sample interrupt input once per 2 clocks</td></tr><tr><td>0x2</td><td>Sample interrupt input once per 4 clocks</td></tr><tr><td>0x3</td><td>Sample interrupt input once per 8 clocks</td></tr><tr><td>0x4</td><td>Sample interrupt input once per 16 clocks</td></tr><tr><td>0x5</td><td>Sample interrupt input once per 32 clocks</td></tr><tr><td>0x6</td><td>Sample interrupt input once per 64 clocks</td></tr><tr><td>0x7</td><td>Sample interrupt input once per 128 clocks</td></tr><tr><td>0x8</td><td>Sample interrupt input once per 256 clocks</td></tr><tr><td>0x9</td><td>Sample interrupt input once per 2*256 clocks</td></tr><tr><td>0xA</td><td>Sample interrupt input once per 4*256clocks</td></tr><tr><td>0xB</td><td>Sample interrupt input once per 8*256 clocks</td></tr><tr><td>0xC</td><td>Sample interrupt input once per 16*256 clocks</td></tr><tr><td>0xD</td><td>Sample interrupt input once per 32*256 clocks</td></tr><tr><td>0xE</td><td>Sample interrupt input once per 64*256 clocks</td></tr><tr><td>0xF</td><td>Sample interrupt input once per 128*256 clocks</td></tr></table> | DBCLKSEL                                      | Description | 0x0 | Sample interrupt input once per 1 clock | 0x1 | Sample interrupt input once per 2 clocks | 0x2 | Sample interrupt input once per 4 clocks | 0x3 | Sample interrupt input once per 8 clocks | 0x4 | Sample interrupt input once per 16 clocks | 0x5 | Sample interrupt input once per 32 clocks | 0x6 | Sample interrupt input once per 64 clocks | 0x7 | Sample interrupt input once per 128 clocks | 0x8 | Sample interrupt input once per 256 clocks | 0x9 | Sample interrupt input once per 2*256 clocks | 0xA | Sample interrupt input once per 4*256clocks | 0xB | Sample interrupt input once per 8*256 clocks | 0xC | Sample interrupt input once per 16*256 clocks | 0xD | Sample interrupt input once per 32*256 clocks | 0xE | Sample interrupt input once per 64*256 clocks | 0xF | Sample interrupt input once per 128*256 clocks |
|       |  | DBCLKSEL  | Description                                   |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x0   | Sample interrupt input once per 1 clock       |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x1   | Sample interrupt input once per 2 clocks      |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x2   | Sample interrupt input once per 4 clocks      |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x3   | Sample interrupt input once per 8 clocks      |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x4   | Sample interrupt input once per 16 clocks     |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x5   | Sample interrupt input once per 32 clocks     |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x6   | Sample interrupt input once per 64 clocks     |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x7   | Sample interrupt input once per 128 clocks    |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x8   | Sample interrupt input once per 256 clocks    |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0x9   | Sample interrupt input once per 2*256 clocks  |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0xA   | Sample interrupt input once per 4*256clocks   |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0xB   | Sample interrupt input once per 8*256 clocks  |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0xC   | Sample interrupt input once per 16*256 clocks |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
|       |  | 0xD   | Sample interrupt input once per 32*256 clocks |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
| 0xE   | Sample interrupt input once per 64*256 clocks  |   |   |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |
| 0xF   | Sample interrupt input once per 128*256 clocks |   |   |             |     |   |     |  |     |  |     |  |     |   |     |   |     |   |     |  |     |  |     |  |     |   |     |  |     |   |     |   |     |   |     |  |

**PWM Port Output Enable (PWMPOEN)**

| Register | Offset       | R/W | Description            | Reset Value |
|----------|--------------|-----|------------------------|-------------|
| PWMPOEN  | GP0_BA+0x2E4 | R/W | PWM Port Output Enable | 0x0000_00XX |

|    |    |    |         |         |          |         |          |
|----|----|----|---------|---------|----------|---------|----------|
| 31 | 30 | 29 | 28      | 27      | 26       | 25      | 24       |
| -  |    |    |         |         |          |         |          |
| 23 | 22 | 21 | 20      | 19      | 18       | 17      | 16       |
| -  |    |    |         |         |          |         |          |
| 15 | 14 | 13 | 12      | 11      | 10       | 9       | 8        |
| -  |    |    |         |         |          |         |          |
| 7  | 6  | 5  | 4       | 3       | 2        | 1       | 0        |
| -  |    |    | HZ_BPWM | HZ_Odd1 | HZ_Even1 | HZ_Odd0 | HZ_Even0 |

| Bits   | Description |  |
|--------|-------------|--|
| [31:5] | -           | -  |
| [4]    | HZ_BPWM     | <b>Basic PWM Ports Output Control</b><br>1 = The driving mode of Basic PWM ports are forced in tri-state (Hi-Z) all the time.<br>0 = The driving mode of Basic PWM ports are controlled by GPIO mode register (Pn_PMD) or multi-function register (Pn_MFP).<br>The initial value is loaded from config0[CHZ_BPWM] after any reset.                 |
| [3]    | HZ_Odd1     | <b>PWM Unit1 Odd Ports Output Control</b><br>1 = The driving mode of PWM unit1 odd ports are forced in tri-state (Hi-Z) all the time.<br>0 = The driving mode of PWM unit1 odd ports are controlled by GPIO mode register (Pn_PMD) or multi-function register (Pn_MFP).<br>The initial value is loaded from config0[CHZ_Odd1] after any reset.     |
| [2]    | HZ_Even1    | <b>PWM Unit1 Even Ports Output Control</b><br>1 = The driving mode of PWM unit1 even ports are forced in tri-state (Hi-Z) all the time.<br>0 = The driving mode of PWM unit1 even ports are controlled by GPIO mode register (Pn_PMD) or multi-function register (Pn_MFP).<br>The initial value is loaded from config0[CHZ_Even1] after any reset. |
| [1]    | HZ_Odd0     | <b>PWM Unit0 Odd Ports Output Control</b><br>1 = The driving mode of PWM unit0 odd ports are forced in tri-state (Hi-Z) all the time.<br>0 = The driving mode of PWM unit0 odd ports are controlled by GPIO mode register (Pn_PMD) or multi-function register (Pn_MFP).<br>The initial value is loaded from config0[CHZ_Odd0] after any reset.     |

| Bits | Description |   |
|------|-------------|---|
| [0]  | HZ_Even0    | <p><b>PWM Unit0 Even Ports Output Control</b></p> <p>1 = The driving mode of PWM unit0 even ports are forced in tri-state (Hi-Z) all the time.</p> <p>0 = The driving mode of PWM unit0 even ports are controlled by GPIO mode register (Pn_PMD) or multi-function register (Pn_MFP).</p> <p>The initial value is loaded from config0[CHZ_Even0] after any reset.</p> |

**GPIO Port n Bit m I/O value (Pn\_m, n = 0~10(0x0A), m = 0~7)**

| Register | Offset                         | R/W | Description                 | Reset Value |
|----------|--------------------------------|-----|-----------------------------|-------------|
| Pn_m     | GPIOB_BA+(0x20 * n + 0x04 * m) | R/W | GPIO Port n bit m I/O value | 0x0000_000X |

|    |    |    |    |    |    |    |     |
|----|----|----|----|----|----|----|-----|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24  |
| -  |    |    |    |    |    |    |     |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16  |
| -  |    |    |    |    |    |    |     |
| 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8   |
| -  |    |    |    |    |    |    |     |
| 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0   |
| -  |    |    |    |    |    |    | Pnm |

| Bits   | Description |  |
|--------|-------------|--|
| [31:1] | -           | Reserved.  |
| [0]    | Pnm         | <p><b>Port n bit m (Pnm) value.</b></p> <p>Write:</p> <p>1 = Set Pnm latch to output high.</p> <p>0 = Clear Pnm latch to output low.</p> <p>Read:</p> <p>1 = Port pin of Pnm is a high level.</p> <p>0 = Port pin of Pnm is a low level.</p> <p>For example: a writing of P00 reflects the value of bit P0_DOUT[0], a reading returns the value of P0_PIN[0].</p> <p><b>Note:</b> The write operation will not be affected by register GPIOx_DMASK</p> |

## 10 TIMERS/COUNTERS

### 10.1 Overview

The timer controller includes four 32-bit timers, TIMER0~TIMER3, which allows user to easily implement a timer control for applications. The timer can perform functions, such as frequency measurement, event counting, interval measurement, clock generation, and delay timing. The timer can generate an interrupt signal upon timeout, or provide the current value during operation.

### 10.2 Features

- Four sets of 32-bit timers with 24-bit up counter and one 8-bit pre-scale counter
- Independent clock source for each timer
- Provides one-shot, periodic, toggle and continuous counting operation modes
- Time out period = (Period of timer clock input) \* (8-bit pre-scale counter + 1) \* (24-bit TCMP)
- Maximum counting cycle time =  $(1 / T \text{ MHz}) * (2^8) * (2^{24})$ , T is the period of timer clock
- 24-bit up counter value is readable through TDR (Timer Data Register)
- Supports event counting function to count the event from external pin
- Supports external pin capture function for interval measurement
- Supports external pin capture function for reset timer counter.



## 10.4 Function Description

Timer controller provides One-shot, Period, Toggle and Continuous Counting operation modes. The event counting function is also provided to count the events/counts from external pin and external pin capture function for interval measurement or reset timer counter. Each operating function mode is shown as follows:

### 10.4.1 One-shot mode

If the timer is operated in One-shot mode (TCSR MODE[1:0] is 0x0) and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, the TIF (TISR[0] timer interrupt flag) will set to 1. If IE (TCSR[29] interrupt enable bit) is set to 1, and TIF (TISR[0] timer interrupt flag) is 1 then the interrupt signal is generated and sent to NVIC to inform CPU for indicating that the timer counting overflow happens. If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, TIF (TISR[0] timer interrupt flag) will set to 1, timer counting operation stops and the timer counter value (TDR value) goes back to counting initial value then CEN (TCSR[30] timer enable bit) is cleared to 0 by timer controller automatically. That is to say, timer operates timer counting and compares with TCMPR value function only one time after programming the timer compare register (TCMPR) value and CEN (TCSR[30] timer enable bit) is set to 1. So, this operating mode is called One-Shot mode.

### 10.4.2 Periodic mode

If the timer is operated in Period mode (TCSR MODE[1:0] is 0x1) and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, the TIF (TISR[0] timer interrupt flag) will set to 1. If IE (TCSR[29] interrupt enable bit) is set to 1, and TIF (TISR[0] timer interrupt flag) is 1 then the interrupt signal is generated and sent to NVIC to inform CPU for indicating that the timer counting overflow happens. If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, TIF (TISR[0] timer interrupt flag) will set to 1, the timer counter value (TDR value) goes back to counting initial value and CEN (TCSR[30] timer enable bit) is kept at 1 (counting enable continuously) and timer counter operates up counting again. If TIF (TISR[0] timer interrupt flag) is cleared by software, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value again, TIF (TISR[0] timer interrupt flag) will set to 1 also. That is to say, timer operates timer counting and compares with TCMPR value function periodically. The timer counting operation does not stop until the CEN (TCSR[30] timer enable bit) is set to 0. The interrupt signal is also generated periodically. So, this operating mode is called Periodic mode.

### 10.4.3 Toggle mode

If the timer is operated in Toggle mode (TCSR MODE[1:0] is 0x2) and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, the TIF (TISR[0] timer interrupt flag) will set to 1. If IE (TCSR[29] interrupt enable bit) is set to 1, and TIF (TISR[0] timer interrupt flag) is 1 then the interrupt signal is generated and sent to NVIC to inform CPU for indicating that the timer counting overflow happens. If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, TIF (TISR[0] timer interrupt flag) will set to 1, toggle out signal (TMx pin) is set to 1, the timer counter value (TDR value) goes back to counting initial value and CEN (TCSR[30] timer enable bit) is kept at 1 (counting enable continuously) and timer counter operates up counting again. If TIF (TISR[0] timer interrupt flag) is cleared by software, once the timer counter



value (TDR value) reaches timer compare register (TCMPR) value again, TIF (TISR[0] timer interrupt flag) will set to 1 also and toggle out signal (TMx pin) is set to 0. The timer counting operation does not stop until the CEN (TCSR[30] timer enable bit) is set to 0. Thus, the toggle output signal (TMx pin) is changing back and forth with 50% duty cycle. So, this operating mode is called Toggle mode.

#### 10.4.4 Continuous Counting Mode

If the timer is operated in Continuous Counting mode (TCSR MODE[1:0] is 0x3) and CEN (TCSR[30] timer enable bit) is set to 1, the timer counter starts up counting. Once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, the TIF (TISR[0] timer interrupt flag) will set to 1. If IE (TCSR[29] interrupt enable bit) is set to 1, and TIF (TISR[0] timer interrupt flag) is 1 then the interrupt signal is generated and sent to NVIC to inform CPU for indicating that the timer counting overflow happens. If IE (TCSR[29] interrupt enable bit) is set to 0, no interrupt signal is generated.

In this operating mode, once the timer counter value (TDR value) reaches timer compare register (TCMPR) value, TIF (TISR[0] timer interrupt flag) will set to 1 and CEN (TCSR[30] timer enable bit) is kept at 1 (counting enable continuously) and timer counter continuous counting without reload the timer counter value (TDR value) to counting initial value. User can change different timer compare register (TCMPR) value immediately without disabling timer counter and restarting timer counter counting.

For example, timer compare register (TCMPR) value is set as 80, first. (The timer compare register (TCMPR) should be less than  $2^{24}$  and be greater than 1). Once the timer counter value (TDR value) reaches to 80, TIF (TISR[0] timer interrupt flag) will set to 1 and (TCSR[30] timer enable bit) is kept at 1 (counting enable continuously) and timer counter value (TDR value) will not goes back to 0, it continues counting to 81, 82, 83, ... to  $(2^{24} - 1)$  then 0, 1, 2, 3, ... to  $2^{24} - 1$  again and again. Next, if user programs timer compare register (TCMPR) value as 200 and the TIF (TISR[0] timer interrupt flag) is cleared to 0, then TIF (TISR[0] timer interrupt flag) will set to 1 again when timer counter value (TDR value) reaches to 200. At last, user programs timer compare register (TCMPR) value as 500 and clears TIF (TISR[0] timer interrupt flag) to 0, then TIF (TISR[0] timer interrupt flag) will set to 1 again when timer counter value (TDR value) reaches to 500. In this mode, the timer counter value (TDR value) is keeping up counting always even if TIF (TISR[0] timer interrupt flag) is 1. So, this operation mode is called as Continuous Counting mode.

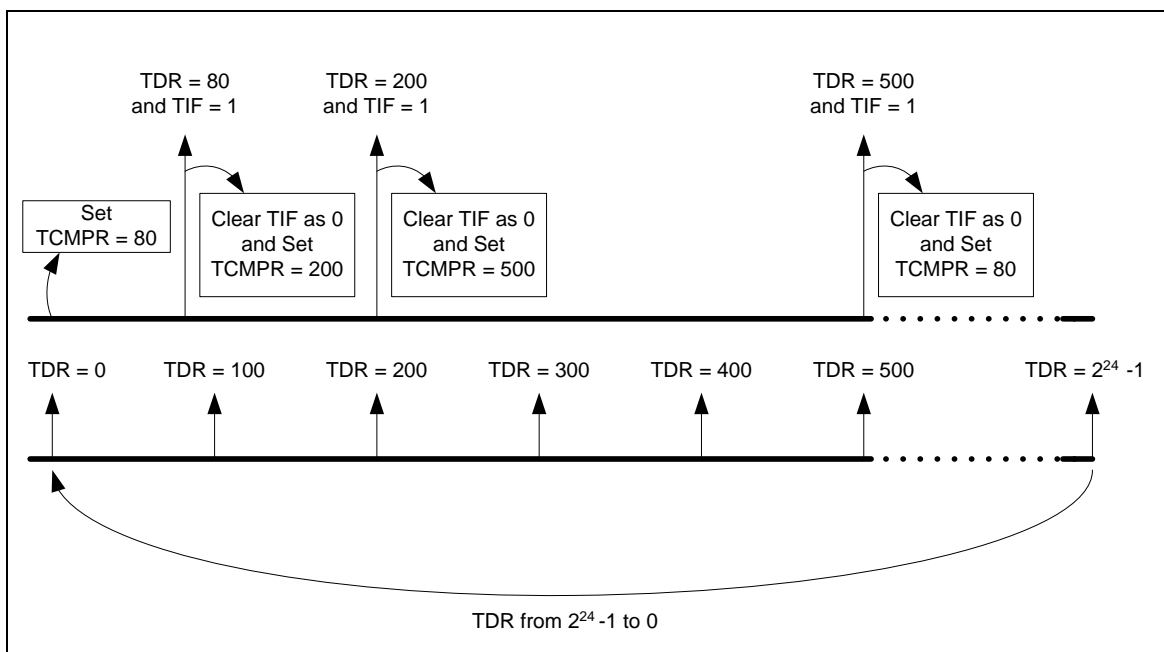


Figure 10-3 Continuous Counting Mode

#### 10.4.5 Event Counting Function

An application which can count the events/counts from TMx pin (event counting pin) is called as event counting function. In this mode, most of the timer control registers are the same with the timer operating function mode except TCSR MODE[1:0] must set to 0x2 (toggle out mode will be disabled) and the clock source of timer controller, TMRx\_CLK, in Figure 10-2 should be set as HCLK. When status transition on TMx pin, the event counter value (TDR value) will be counted according to TX\_PHASE (TEXCON[0] timer external count phase) setting. TCDB (TEXCON[7] timer counter pin de-bounce enable) bit is for enabled or disabled edge detection de-bounce circuit of TMx pin. The max frequency of event counting source on TMx pin should be less than 1/3 HCLK if TCDB (TEXCON[7] timer counter pin de-bounce enable) is 0 or less than 1/8 HCLK if TCDB (TEXCON[7] timer counter pin de-bounce enable) is 1. Otherwise, the event counter value (TDR value) will not be counted normally.

#### 10.4.6 External Pin Capture Function

In this mode, timer will monitor the transition of TMx pin (external capture pin) to save the timer counter value (TDR value) to timer capture value (TCAP value) or reset the timer counter value (TDR value) to 0. And the clock source of timer controller, TMRx\_CLK, in Figure 10-2 should be set as HCLK.

If RSTCAPSEL (TEXCON[4] timer external reset counter / capture mode select) bit is 0, the transition on TMx pin is used as timer counter capture function. And when the transition of TMx pin matches the TEX\_EDGE (TEXCON[2:1] timer external pin edge detect) setting, TEXIF (TEXISR[0] timer external interrupt flag) will set to 1 and the timer counter value (TDR value) will be saved into timer capture value (TCAP value).

If RSTCAPSEL (TEXCON[4] timer external reset counter / capture mode select) bit is 1, the transition on TMx pin is used as timer counter reset function. In this mode, once the transition of TMx pin matches TEX\_EDGE (TEXCON[2:1] timer external pin edge detect) setting, TEXIF (TEXISR[0] timer external interrupt flag) will set to 1 and the timer counter value (TDR value) will be reset to 0.

The max frequency of external capture source on TMx pin should be less than 1/3 HCLK if TEXDB (TEXCON[6] timer external capture pin de-bounce enable) is 0 or less than 1/8 HCLK if TEXDB (TEXCON[6] timer external capture pin de-bounce enable) is 1. Otherwise, the transition on TMx pin will not be detected and TEXIF (TEXISR[0] timer external interrupt flag) will not set to 1 normally.

### 10.5 Register Map

R: read only, W: write only, R/W: both read and write

| Register                      | Offset        | R/W | Description                               | Reset Value |
|-------------------------------|---------------|-----|---|-------------|
| <b>TIMER Base Address:</b>    |               |     |   |             |
| <b>TMR01_BA = 0x4001_0000</b> |               |     |   |             |
| <b>TMR23_BA = 0x4011_0000</b> |               |     |   |             |
| <b>TCSR0</b>                  | TMR01_BA+0x00 | R/W | Timer0 Control and Status Register        | 0x0000_0005 |
| <b>TCMPR0</b>                 | TMR01_BA+0x04 | R/W | Timer0 Compare Register                   | 0x0000_0000 |
| <b>TISR0</b>                  | TMR01_BA+0x08 | R/W | Timer0 Interrupt Status Register          | 0x0000_0000 |
| <b>TDR0</b>                   | TMR01_BA+0x0C | R   | Timer0 Data Register                      | 0x0000_0000 |
| <b>TCAP0</b>                  | TMR01_BA+0x10 | R   | Timer0 Capture Data Register              | 0x0000_0000 |
| <b>TEXCON0</b>                | TMR01_BA+0x14 | R/W | Timer0 External Control Register          | 0x0000_0000 |
| <b>TEXISR0</b>                | TMR01_BA+0x18 | R/W | Timer0 External Interrupt Status Register | 0x0000_0000 |
| <b>TCSR1</b>                  | TMR01_BA+0x20 | R/W | Timer1 Control and Status Register        | 0x0000_0005 |
| <b>TCMPR1</b>                 | TMR01_BA+0x24 | R/W | Timer1 Compare Register                   | 0x0000_0000 |
| <b>TISR1</b>                  | TMR01_BA+0x28 | R/W | Timer1 Interrupt Status Register          | 0x0000_0000 |
| <b>TDR1</b>                   | TMR01_BA+0x2C | R   | Timer1 Data Register                      | 0x0000_0000 |
| <b>TCAP1</b>                  | TMR01_BA+0x30 | R   | Timer1 Capture Data Register              | 0x0000_0000 |
| <b>TEXCON1</b>                | TMR01_BA+0x34 | R/W | Timer1 External Control Register          | 0x0000_0000 |
| <b>TEXISR1</b>                | TMR01_BA+0x38 | R/W | Timer1 External Interrupt Status Register | 0x0000_0000 |
| <b>TCSR2</b>                  | TMR23_BA+0x00 | R/W | Timer2 Control and Status Register        | 0x0000_0005 |
| <b>TCMPR2</b>                 | TMR23_BA+0x04 | R/W | Timer2 Compare Register                   | 0x0000_0000 |
| <b>TISR2</b>                  | TMR23_BA+0x08 | R/W | Timer2 Interrupt Status Register          | 0x0000_0000 |
| <b>TDR2</b>                   | TMR23_BA+0x0C | R   | Timer2 Data Register                      | 0x0000_0000 |
| <b>TCAP2</b>                  | TMR23_BA+0x10 | R   | Timer2 Capture Data Register              | 0x0000_0000 |
| <b>TEXCON2</b>                | TMR23_BA+0x14 | R/W | Timer2 External Control Register          | 0x0000_0000 |
| <b>TEXISR2</b>                | TMR23_BA+0x18 | R/W | Timer2 External Interrupt Status Register | 0x0000_0000 |
| <b>TCSR3</b>                  | TMR23_BA+0x20 | R/W | Timer3 Control and Status Register        | 0x0000_0005 |
| <b>TCMPR3</b>                 | TMR23_BA+0x24 | R/W | Timer3 Compare Register                   | 0x0000_0000 |
| <b>TISR3</b>                  | TMR23_BA+0x28 | R/W | Timer3 Interrupt Status Register          | 0x0000_0000 |
| <b>TDR3</b>                   | TMR23_BA+0x2C | R   | Timer3 Data Register                      | 0x0000_0000 |
| <b>TCAP3</b>                  | TMR23_BA+0x30 | R   | Timer3 Capture Data Register              | 0x0000_0000 |

|                |               |     |   |             |
|----------------|---------------|-----|---|-------------|
| <b>TEXCON3</b> | TMR23_BA+0x34 | R/W | Timer3 External Control Register          | 0x0000_0000 |
| <b>TEXISR3</b> | TMR23_BA+0x38 | R/W | Timer3 External Interrupt Status Register | 0x0000_0000 |

### 10.6 Register Description

#### Timer Control and Status Register (TCSR)

| Register | Offset        | R/W | Description                        | Reset Value |
|----------|---------------|-----|------------------------------------|-------------|
| TCSR0    | TMR01_BA+0x00 | R/W | Timer0 Control and Status Register | 0x0000_0005 |
| TCSR1    | TMR01_BA+0x20 | R/W | Timer1 Control and Status Register | 0x0000_0005 |
| TCSR2    | TMR23_BA+0x00 | R/W | Timer2 Control and Status Register | 0x0000_0005 |
| TCSR3    | TMR23_BA+0x20 | R/W | Timer3 Control and Status Register | 0x0000_0005 |

| 31            | 30  | 29 | 28        | 27 | 26   | 25   | 24     |
|---------------|-----|----|-----------|----|------|------|--------|
| DBGACK_TMR    | CEN | IE | MODE[1:0] |    | CRST | CACT | CTB    |
| 23            | 22  | 21 | 20        | 19 | 18   | 17   | 16     |
| Reserved      |     |    |           |    |      |      | TDR_EN |
| 15            | 14  | 13 | 12        | 11 | 10   | 9    | 8      |
| Reserved      |     |    |           |    |      |      |        |
| 7             | 6   | 5  | 4         | 3  | 2    | 1    | 0      |
| PRESCALE[7:0] |     |    |           |    |      |      |        |

| Bits | Description |   |
|------|-------------|---|
| [31] | DBGACK_TMR  | <b>ICE Debug Mode Acknowledge Disable (Write-protection Bit)</b><br>0 = ICE debug mode acknowledgement effects TIMER counting.<br>TIMER counter will be held while CPU is held by ICE.<br>1 = ICE debug mode acknowledgement disabled.<br>TIMER counter will keep going no matter CPU is held by ICE or not.  |
| [30] | CEN         | <b>Timer Enable Bit</b><br>1 = Starts counting<br>0 = Stops/Suspends counting<br><b>Note1:</b> In stop status, and then set CEN to 1 will enable the 24-bit up counter to keep counting from the last stop counting value.<br><b>Note2:</b> This bit is auto-cleared by hardware in one-shot mode (TCSR [28:27] = 00) when the timer interrupt flag (TISR[0] TIF) is generated. |
| [29] | IE          | <b>Interrupt Enable Bit</b><br>1 = Timer Interrupt Enabled<br>0 = Timer Interrupt Disabled<br>If this bit is enabled, when the timer interrupt flag (TISR[0] TIF) is set to 1, the timer interrupt signal is generated and inform to CPU.   |

|         |                 |  |  |
|---------|-----------------|--|--|
| [28:27] | <b>MODE</b>     | <b>Timer Operating Mode</b>  |  |
|         |                 | <b>MODE</b>  | <b>Timer Operating Mode</b>                        |
|         |                 | 00   | The timer is operated in One-shot mode.            |
|         |                 | 01   | The timer is operated in Periodic mode.            |
|         |                 | 10   | The timer is operated in Toggle mode.              |
|         |                 | 11   | The timer is operated in Continuous Counting mode. |
| [26]    | <b>CRST</b>     | <b>Timer Reset Bit</b><br>Setting this bit will reset the 24-bit up counter value (TDR) and also force CEN (TCSR[30] timer enable bit) to 0 if CACT (TCSR[25] timer active status bit) is 1.<br>0 = No effect<br>1 = Reset 8-bit pre-scale counter, 24-bit up counter value and CEN bit  |  |
| [25]    | <b>CACT</b>     | <b>Timer Active Status Bit (Read Only)</b><br>This bit indicates the 24-bit up counter status.<br>0 = 24-bit up counter is not active<br>1 = 24-bit up counter is active   |  |
| [24]    | <b>CTB</b>      | <b>Counter Mode Enable Bit</b><br>This bit is for external counting pin function enabled. When timer is used as an event counter, this bit should be set to 1 and select HCLK as timer clock source. Please refer to 10.4.5 for detail description.<br>1 = External counter mode Enabled<br>0 = External counter mode Disabled |  |
| [23:17] | <b>Reserved</b> | Reserved   |  |
| [16]    | <b>TDR_EN</b>   | <b>Data Load Enable</b><br>When this bit is set, timer counter value (TDR) will be updated continuously to monitor internal 24-bit up counter value as the counter is counting.<br>1 = Timer Data Register update Enabled while timer counter is active<br>0 = Timer Data Register update Disabled                             |  |
| [15:8]  | <b>Reserved</b> | Reserved   |  |
| [7:0]   | <b>PRESCALE</b> | <b>Pre-scale Counter</b><br>Timer input clock source is divided by (PRESCALE+1) before it is fed to the timer up counter. If this field is 0 (PRESCALE = 0), then there is no scaling.   |  |

### Timer Compare Register (TCMPR)

| Register | Offset        | R/W | Description             | Reset Value |
|----------|---------------|-----|-------------------------|-------------|
| TCMPR0   | TMR01_BA+0x04 | R/W | Timer0 Compare Register | 0x0000_0000 |
| TCMPR1   | TMR01_BA+0x24 | R/W | Timer1 Compare Register | 0x0000_0000 |
| TCMPR2   | TMR23_BA+0x04 | R/W | Timer2 Compare Register | 0x0000_0000 |
| TCMPR3   | TMR23_BA+0x24 | R/W | Timer3 Compare Register | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| TCMP [23:16] |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| TCMP [15:8]  |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TCMP [7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:24] | Reserved    | Reserved  |
| [23:0]  | TCMP        | <p><b>Timer Compared Value</b></p> <p>TCMP is a 24-bit compared value register. When the internal 24-bit up counter value is equal to TCMP value, the TIF (TISR[0] timer interrupt flag) will set to 1.</p> <p>Time out period = (Period of timer clock input) * (8-bit PRESCALE + 1) * (24-bit TCMP)</p> <p><b>Note1:</b> Never write 0x0 or 0x1 in TCMP field, or the core will run into unknown state.</p> <p><b>Note2:</b> When timer is operating at continuous counting mode, the 24-bit up counter will keep counting continuously even if software writes a new value into TCMP field. But if timer is operating at other modes, the 24-bit up counter will restart counting and using newest TCMP value to be the timer compared value if software writes a new value into TCMP field.</p> |

### Timer Interrupt Status Register (TISR)

| Register     | Offset        | R/W | Description                      | Reset Value |
|--------------|---------------|-----|----------------------------------|-------------|
| <b>TISR0</b> | TMR01_BA+0x08 | R/W | Timer0 Interrupt Status Register | 0x0000_0000 |
| <b>TISR1</b> | TMR01_BA+0x28 | R/W | Timer1 Interrupt Status Register | 0x0000_0000 |
| <b>TISR2</b> | TMR23_BA+0x08 | R/W | Timer2 Interrupt Status Register | 0x0000_0000 |
| <b>TISR3</b> | TMR23_BA+0x28 | R/W | Timer3 Interrupt Status Register | 0x0000_0000 |

|          |    |    |    |    |    |    |            |
|----------|----|----|----|----|----|----|------------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24         |
| Reserved |    |    |    |    |    |    |            |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16         |
| Reserved |    |    |    |    |    |    |            |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8          |
| Reserved |    |    |    |    |    |    |            |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0          |
| Reserved |    |    |    |    |    |    | <b>TIF</b> |

| Bits   | Description |   |
|--------|-------------|---|
| [31:1] | Reserved    | Reserved  |
| [0]    | <b>TIF</b>  | <b>Timer Interrupt Flag</b><br>This bit indicates the interrupt flag status of Timer.<br>And this bit is set by hardware when the timer counter value matches the timer compared value (TCMP value). It is cleared by software writing 1 to this bit. |



### Timer Data Register (TDR)

| Register    | Offset        | R/W | Description          | Reset Value |
|-------------|---------------|-----|----------------------|-------------|
| <b>TDR0</b> | TMR01_BA+0x0C | R   | Timer0 Data Register | 0x0000_0000 |
| <b>TDR1</b> | TMR01_BA+0x2C | R   | Timer1 Data Register | 0x0000_0000 |
| <b>TDR2</b> | TMR23_BA+0x0C | R   | Timer2 Data Register | 0x0000_0000 |
| <b>TDR3</b> | TMR23_BA+0x2C | R   | Timer3 Data Register | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved   |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| TDR[23:16] |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| TDR[15:8]  |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TDR[7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:24] | Reserved    | Reserved  |
| [23:0]  | TDR         | <b>Timer Data Register</b><br>If TDR_EN (TCSR[16]) is set to 1, TDR will be updated continuously to monitor 24-bit timer counter value. |

### Timer Capture Data Register (TCAP)

| Register     | Offset        | R/W | Description                  | Reset Value |
|--------------|---------------|-----|------------------------------|-------------|
| <b>TCAP0</b> | TMR01_BA+0x10 | R   | Timer0 Capture Data Register | 0x0000_0000 |
| <b>TCAP1</b> | TMR01_BA+0x30 | R   | Timer1 Capture Data Register | 0x0000_0000 |
| <b>TCAP2</b> | TMR23_BA+0x10 | R   | Timer2 Capture Data Register | 0x0000_0000 |
| <b>TCAP3</b> | TMR23_BA+0x30 | R   | Timer3 Capture Data Register | 0x0000_0000 |

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved    |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| TCAP[23:16] |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| TCAP[15:8]  |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TCAP[7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:24] | Reserved    | Reserved   |
| [23:0]  | TCAP        | <b>Timer Capture Data Register</b><br>When TEXEN (TEXCON[3] timer external pin enable) bit is set, RSTCAPSEL (TEXCON[4] timer external reset counter/capture mode select) bit is 0, and a transition on TMx pin matched the TEX_EDGE (TEXCON[2:1] timer external pin edge detect) setting, TEXIF (TEXISR[0] timer external interrupt flag) will set to 1 and the current timer counter value (TDR value) will be auto-loaded into this TCAP field. |

Timer External Control Register (TEXCON)

| Register | Offset        | R/W | Description                      | Reset Value |
|----------|---------------|-----|----------------------------------|-------------|
| TEXCON0  | TMR01_BA+0x14 | R/W | Timer0 External Control Register | 0x0000_0000 |
| TEXCON1  | TMR01_BA+0x34 | R/W | Timer1 External Control Register | 0x0000_0000 |
| TEXCON2  | TMR23_BA+0x14 | R/W | Timer2 External Control Register | 0x0000_0000 |
| TEXCON3  | TMR23_BA+0x34 | R/W | Timer3 External Control Register | 0x0000_0000 |

|          |       |        |           |       |          |    |          |
|----------|-------|--------|-----------|-------|----------|----|----------|
| 31       | 30    | 29     | 28        | 27    | 26       | 25 | 24       |
| Reserved |       |        |           |       |          |    |          |
| 23       | 22    | 21     | 20        | 19    | 18       | 17 | 16       |
| Reserved |       |        |           |       |          |    |          |
| 15       | 14    | 13     | 12        | 11    | 10       | 9  | 8        |
| Reserved |       |        |           |       |          |    |          |
| 7        | 6     | 5      | 4         | 3     | 2        | 1  | 0        |
| TCDB     | TEXDB | TEXIEN | RSTCAPSEL | TEXEN | TEX_EDGE |    | TX_PHASE |

| Bits   | Description |  |
|--------|-------------|--|
| [31:8] | Reserved    | Reserved   |
| [7]    | TCDB        | <b>Timer Counter pin De-bounce Enable</b><br>1 = TMx pin de-bounce Enabled.<br>0 = TMx pin de-bounce Disabled.<br>If this bit is enabled, the edge detection of TMx pin is detected with de-bounce circuit.  |
| [6]    | TEXDB       | <b>Timer External Capture pin De-bounce Enable</b><br>1 = TMx pin de-bounce Enabled.<br>0 = TMx pin de-bounce Disabled.<br>If this bit is enabled, the edge detection of TMx pin is detected with de-bounce circuit.   |
| [5]    | TEXIEN      | <b>Timer External interrupt Enable</b><br>1 = TMx pin detection Interrupt Enabled<br>0 = TMx pin detection Interrupt Disabled.<br>TEXIEN is used to enable timer external interrupt. If TEXIEN enabled, timer will rise an interrupt when TEXIF = 1.   |
| [4]    | RSTCAPSEL   | <b>Timer External Reset Counter / Capture Mode Select</b><br>1 = Transition on TMx pin is using to reset the 24-bit timer counter.<br>0 = Transition on TMx pin is using to save the 24-bit timer counter value (TDR value) to timer capture value (TCAP value) if TEXIF (TEXISR[0] timer external interrupt flag) is set to 1 |
| [3]    | TEXEN       | <b>Timer External Pin Enable</b><br>This bit enables the RSTCAPSEL (TEXCON[4] timer external reset counter/capture mode select ) function on the TMx pin.  |

|       |                 |   |
|-------|-----------------|---|
|       |                 | 1 = RSTCAPSEL function of TMx pin is active.<br>0 = RSTCAPSEL function of TMx pin will be ignored.  |
| [2:1] | <b>TEX_EDGE</b> | <b>Timer External Pin Edge Detect</b><br>00 = A 1 to 0 transition on TMx pin will be detected.<br>01 = A 0 to 1 transition on TMx pin will be detected.<br>10 = Either 1 to 0 or 0 to 1 transition on TMx pin will be detected.<br>11 = Reserved. |
| [0]   | <b>TX_PHASE</b> | <b>Timer External Count Phase</b><br>This bit indicates the detection phase of external counting pin.<br>1 = A rising edge of external counting pin will be counted.<br>0 = A falling edge of external counting pin will be counted.              |

**Timer External Interrupt Status Register (TEXISR)**

| Register | Offset        | R/W | Description                               | Reset Value |
|----------|---------------|-----|---|-------------|
| TEXISR0  | TMR01_BA+0x18 | R/W | Timer0 External Interrupt Status Register | 0x0000_0000 |
| TEXISR1  | TMR01_BA+0x38 | R/W | Timer1 External Interrupt Status Register | 0x0000_0000 |
| TEXISR2  | TMR23_BA+0x18 | R/W | Timer2 External Interrupt Status Register | 0x0000_0000 |
| TEXISR3  | TMR23_BA+0x38 | R/W | Timer3 External Interrupt Status Register | 0x0000_0000 |

|          |    |    |    |    |    |    |       |
|----------|----|----|----|----|----|----|-------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24    |
| Reserved |    |    |    |    |    |    |       |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16    |
| Reserved |    |    |    |    |    |    |       |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8     |
| Reserved |    |    |    |    |    |    |       |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0     |
| Reserved |    |    |    |    |    |    | TEXIF |

| Bits   | Description |  |
|--------|-------------|--|
| [31:1] | Reserved    | Reserved   |
| [0]    | TEXIF       | <b>Timer External Interrupt Flag</b><br>This bit indicates the timer external interrupt flag status.<br>When TEXEN (TEXCON[3] timer external pin enable) bit is set, RSTCAPSEL (TEXCON[4] timer external reset counter/capture mode select) bit is 0, and a transition on TMx pin matched the TEX_EDGE (TEXCON[2:1] timer external pin edge detect) setting, this bit will set to 1 by hardware.<br>And it is cleared by software writing 1 to this bit. |

## 11 WATCHDOG TIMER (WDT)

### 11.1 Overview

The purpose of Watchdog Timer is to perform a system reset when system runs into an unknown state. This prevents system from hanging for an infinite period of time. Besides, this Watchdog Timer supports the function to wake-up system from Idle/Power-down mode.

### 11.2 Features

- 18-bit free running up counter for Watchdog Timer time out interval.
- Selectable time out interval ( $2^4 \sim 2^{18}$ ) and the time out interval is 104 ms ~ 26.3168 s (if WDT\_CLK = 10 kHz).
- System keep in reset state for a period of  $(1 / \text{WDT\_CLK}) * 63$
- Supports selectable Watchdog Timer reset delay period, it includes  $(1024+2)$ 、 $(128+2)$ 、 $(16+2)$  or  $(1+2)$  WDT\_CLK reset delay period.
- Supports force Watchdog Timer enabled after chip powered on or reset while CWDTEN (Config0[31] watchdog enable) bit is set to 0.
- Supports Watchdog Timer time out wake-up function when WDT clock source is selected to 10 kHz low speed oscillator

### 11.3 Block Diagram

The Watchdog Timer clock control and block diagram are shown as following.

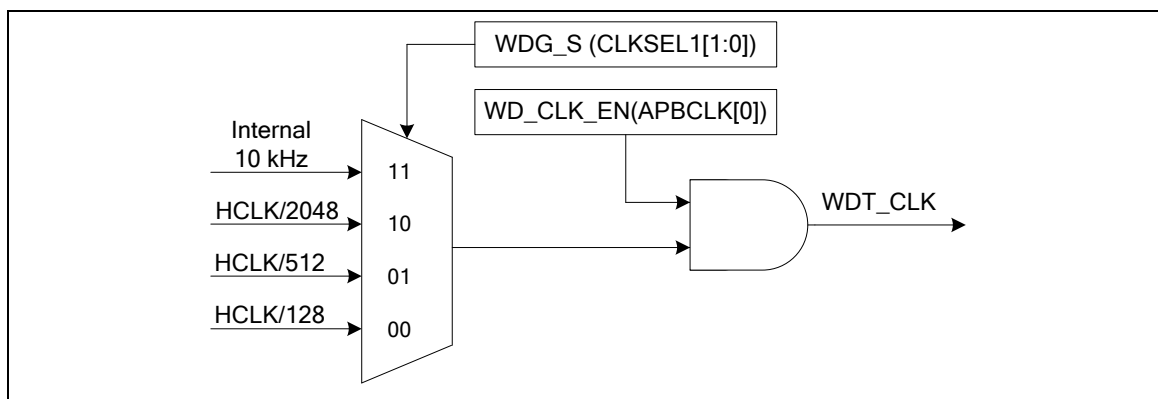


Figure 11-1 Watchdog Timer Clock Control

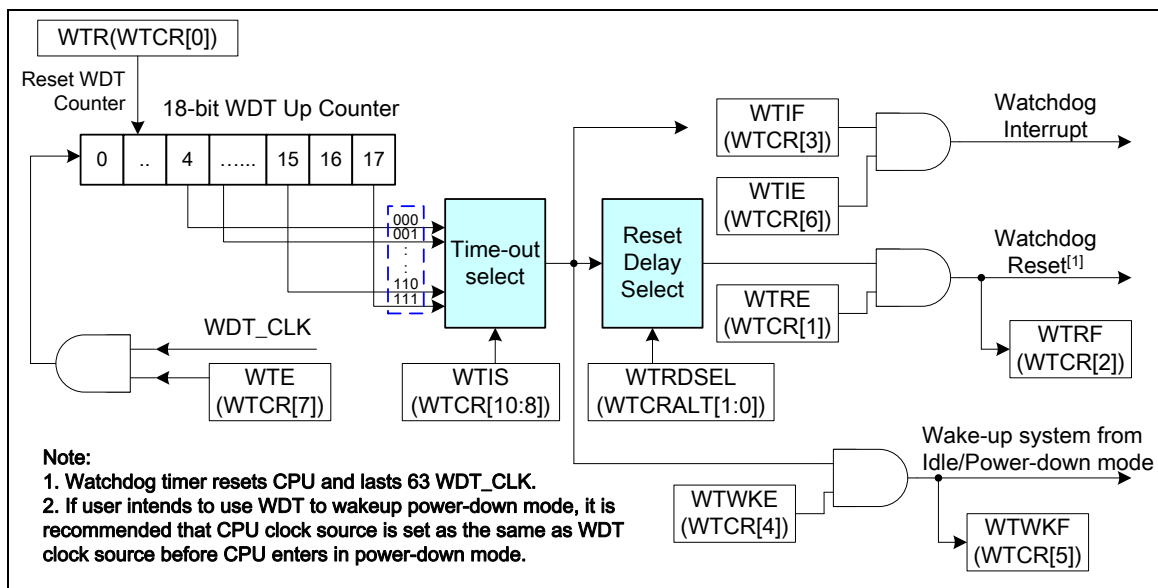


Figure 11-2 Watchdog Timer Block Diagram

#### 11.3.1 Functional Description

The Watchdog Timer (WDT) includes an 18-bit free running up counter with programmable time-out intervals. Table 5-7 shows the WDT time-out interval selection and Figure 11-3 shows the timing of WDT time-out interval and reset period.

Setting WTE (WDTCCR[7] WDT enable) bit to 1 will enable the WDT function and the WDT counter to start counting up. When the counter reaches the selected time-out interval (WTIS settings), WTIF (WTCR[3] WDT interrupt flag) will be set to 1 immediately, in the meanwhile, a specified WDT reset delay period (WTCRALT[1:0] WTRDSEL) follows the WTIF is setting to 1. User must set WTR (WDTCCR[0] Reset WDT counter) 1 to reset the 18-bit WDT counter value to avoid generate WDT time-out reset signal before the WDT reset delay period expires. And WTR bit is cleared automatically by hardware after WDT counter is reset.

There are eight time-out interval period can be selected by setting WTIS (WTCR[10:8] WDT interval selection). If the WDT counter value has not been cleared after the specific WDT reset delay period expires, the WDT control will set WTRF (WTCR[2] WDT reset flag) to 1 if WTRE

(WTCR[1] WDT reset enable) bit is enabled, then chip will reset immediately. This reset period will keep last 63 WDT clocks ( $T_{RST}$ ) then chip restarts executing program from reset vector (0x0000\_0000). And WTRF bit will not be cleared after WDT time-out reset the chip, user can check WTRF bit by software to recognize the system has been reset by WDT time-out reset or not.

The WDT also provides system wake-up function from Idle/Power-Down mode while WTIE (WTCR[6] WDT interrupt enable) bit is enabled and WTIF (WTCR[3] WDT interrupt flag) is set to 1.

| WTIS | Time out Interval Period<br>$T_{TIS}$ | Reset Delay Period<br>$T_{RSTD}$ |
|------|---------------------------------------|----------------------------------|
| 000  | $2^4 * T_{WDT}$                       | $(1/16/128/1024 + 2) * T_{WDT}$  |
| 001  | $2^6 * T_{WDT}$                       | $(1/16/128/1024 + 2) * T_{WDT}$  |
| 010  | $2^8 * T_{WDT}$                       | $(1/16/128/1024 + 2) * T_{WDT}$  |
| 011  | $2^{10} * T_{WDT}$                    | $(1/16/128/1024 + 2) * T_{WDT}$  |
| 100  | $2^{12} * T_{WDT}$                    | $(1/16/128/1024 + 2) * T_{WDT}$  |
| 101  | $2^{14} * T_{WDT}$                    | $(1/16/128/1024 + 2) * T_{WDT}$  |
| 110  | $2^{16} * T_{WDT}$                    | $(1/16/128/1024 + 2) * T_{WDT}$  |
| 111  | $2^{18} * T_{WDT}$                    | $(1/16/128/1024 + 2) * T_{WDT}$  |

Table 11-1 Watchdog Timer Time out Interval Selection

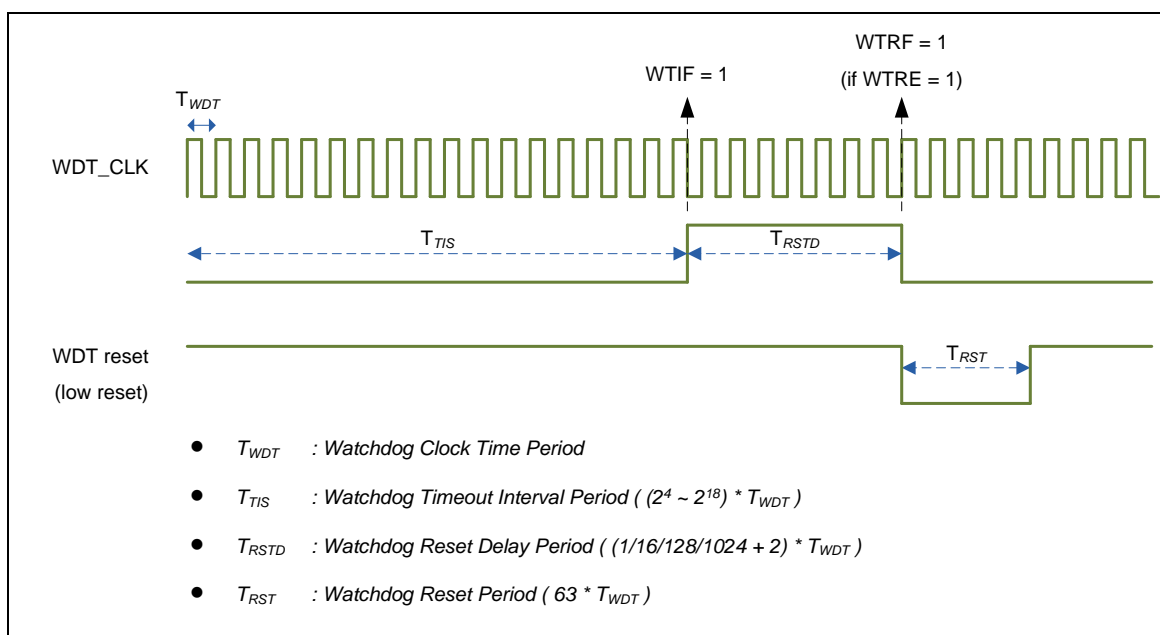


Figure 11-3 Watchdog Timer Time-out Interval and Reset Period Timing



### 11.4 Register Map

R: read only, W: write only, R/W: both read and write

| Register                    | Offset      | R/W | Description                                 | Reset Value |
|-----------------------------|-------------|-----|---|-------------|
| <b>WDT Base Address:</b>    |             |     |   |             |
| <b>WDT_BA = 0x4000_4000</b> |             |     |   |             |
| <b>WTCR</b>                 | WDT_BA+0x00 | R/W | Watchdog Timer Control Register             | 0x0000_0700 |
| <b>WTCRALT</b>              | WDT_BA+0x04 | R/W | Watchdog Timer Alternative Control Register | 0x0000_0000 |

## 11.5 Register Description

### Watchdog Timer Control Register (WTCR)

| Register | Offset      | R/W | Description                     | Reset Value |
|----------|-------------|-----|---------------------------------|-------------|
| WTCR     | WDT_BA+0x00 | R/W | Watchdog Timer Control Register | 0x0000_0700 |

**Note:** All bits that can be written in this register are write-protected. To program it needs to write “59h”, “16h”, “88h” to address 0x5000\_0100 to disable register protection. Refer to the register REGWRPROT at address GCR\_BA+0x100.

| 31         | 30       | 29    | 28    | 27   | 26   | 25   | 24  |
|------------|----------|-------|-------|------|------|------|-----|
| DBGACK_WDT | Reserved |       |       |      |      |      |     |
| 23         | 22       | 21    | 20    | 19   | 18   | 17   | 16  |
| Reserved   |          |       |       |      |      |      |     |
| 15         | 14       | 13    | 12    | 11   | 10   | 9    | 8   |
| Reserved   |          |       |       |      | WTIS |      |     |
| 7          | 6        | 5     | 4     | 3    | 2    | 1    | 0   |
| WTE        | WTIE     | WTWKF | WTWKE | WTIF | WTRF | WTRE | WTR |

| Bits    | Description   |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
|---------|---|------|--------------------------|-----|-----------------|-----|-----------------|-----|-----------------|-----|--------------------|-----|--------------------|-----|--------------------|-----|--------------------|-----|--------------------|
| [31]    | <b>DBGACK_WDT</b><br><b>ICE Debug Mode Acknowledge Disable (Write-protection Bit)</b><br>0 = ICE debug mode acknowledgement affects Watchdog Timer counting.<br>Watchdog Timer counter will be held while CPU is held by ICE.<br>1 = ICE debug mode acknowledgement Disabled.<br>Watchdog Timer counter will keep going no matter CPU is held by ICE or not.  |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| [30:11] | Reserved  |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| [10:8]  | <b>WTIS</b><br><b>Watchdog Timer Interval Selection (Write-protection Bits)</b><br>These three bits select the time-out interval period for the Watchdog Timer. <table border="1"> <thead> <tr> <th>WTIS</th><th>Time-out Interval Period</th></tr> </thead> <tbody> <tr> <td>000</td><td><math>2^4 * T_{WDT}</math></td></tr> <tr> <td>001</td><td><math>2^6 * T_{WDT}</math></td></tr> <tr> <td>010</td><td><math>2^8 * T_{WDT}</math></td></tr> <tr> <td>011</td><td><math>2^{10} * T_{WDT}</math></td></tr> <tr> <td>100</td><td><math>2^{12} * T_{WDT}</math></td></tr> <tr> <td>101</td><td><math>2^{14} * T_{WDT}</math></td></tr> <tr> <td>110</td><td><math>2^{16} * T_{WDT}</math></td></tr> <tr> <td>111</td><td><math>2^{18} * T_{WDT}</math></td></tr> </tbody> </table> | WTIS | Time-out Interval Period | 000 | $2^4 * T_{WDT}$ | 001 | $2^6 * T_{WDT}$ | 010 | $2^8 * T_{WDT}$ | 011 | $2^{10} * T_{WDT}$ | 100 | $2^{12} * T_{WDT}$ | 101 | $2^{14} * T_{WDT}$ | 110 | $2^{16} * T_{WDT}$ | 111 | $2^{18} * T_{WDT}$ |
| WTIS    | Time-out Interval Period  |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| 000     | $2^4 * T_{WDT}$   |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| 001     | $2^6 * T_{WDT}$   |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| 010     | $2^8 * T_{WDT}$   |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| 011     | $2^{10} * T_{WDT}$  |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| 100     | $2^{12} * T_{WDT}$  |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| 101     | $2^{14} * T_{WDT}$  |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| 110     | $2^{16} * T_{WDT}$  |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| 111     | $2^{18} * T_{WDT}$  |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |
| [7]     | <b>WTE</b><br><b>Watchdog Timer Enable (Write-protection Bit)</b>   |      |                          |     |                 |     |                 |     |                 |     |                    |     |                    |     |                    |     |                    |     |                    |

|     |       |  |
|-----|-------|--|
|     |       | <p>0 = Watchdog Timer Disabled (This action will reset the internal counter).<br/>1 = Watchdog Timer Enabled</p> <p><b>Note:</b> If CWDTEN (Config0[31] watchdog enable) bit is set to 0, this bit is forced as 1 and software cannot change this bit to 0.</p>  |
| [6] | WTIE  | <p><b>Watchdog Timer Interrupt Enable (Write-protection Bit)</b></p> <p>If this bit is enabled, the WDT time-out interrupt signal is generated and inform to CPU.</p> <p>0 = Watchdog Timer interrupt Disabled.<br/>1 = Watchdog Timer interrupt Enabled.</p>  |
| [5] | WTWKF | <p><b>Watchdog Timer Wake-up Flag</b></p> <p>This bit indicates the interrupt wake-up flag status of WDT</p> <p>0 = Watchdog Timer does not cause chip wake-up.<br/>1 = Chip wake-up from Idle or Power-down mode if WDT time-out interrupt signal generated.</p> <p>This bit is cleared by writing 1 to this bit..</p>  |
| [4] | WTWKE | <p><b>Watchdog Timer Wake-up Function Enable bit (Write-protection Bit)</b></p> <p>If this bit is set to 1, while WDT interrupt flag (WTCR[3] WTIF) is generated to 1 and WTIE (WTCR[6] WDT interrupt enable) is enabled, the WDT time-out interrupt signal will generate a wake-up trigger event to chip.</p> <p>0 = Wake-up trigger event Disabled if WDT time-out interrupt signal generated<br/>1 = Wake-up trigger event Enabled if WDT time-out interrupt signal generated.</p> <p><b>Note:</b> Chip can be woken-up by WDT time-out interrupt signal generated only if WDT clock source is selected to 10 kHz oscillator.</p> |
| [3] | WTIF  | <p><b>Watchdog Timer Interrupt Flag</b></p> <p>This bit will set to 1 while WDT counter value reaches the selected WDT time-out interval</p> <p>0 = Watchdog Timer time-out interrupt did not occur.<br/>1 = Watchdog Timer time-out interrupt occurred.</p> <p><b>Note:</b> This bit is cleared by writing 1 to this bit.</p>   |
| [2] | WTRF  | <p><b>Watchdog Timer Reset Flag</b></p> <p>This bit indicates the system has been reset by WDT time-out reset or not.</p> <p>0 = Watchdog Timer time-out reset did not occur.<br/>1 = Watchdog Timer time-out reset occurred.</p> <p><b>Note:</b> This bit is cleared by writing 1 to this bit.</p>  |
| [1] | WTRE  | <p><b>Watchdog Timer Reset Enable (Write-protection Bit)</b></p> <p>Setting this bit will enable the Watchdog Timer time-out reset function If the WDT counter value has not been cleared after the specific WDT reset delay period expires..</p> <p>0 = Watchdog Timer time-out reset function Disabled.<br/>1 = Watchdog Timer time-out reset function Enabled.</p>  |
| [0] | WTR   | <p><b>Reset Watchdog Timer Counter (Write-protection Bit)</b></p> <p>0 = No effect.<br/>1 = Reset the internal 18-bit WDT counter.</p> <p><b>Note:</b> This bit will be automatically cleared by hardware.</p>   |

### Watchdog Timer Alternative Control Register (WTCRALT)

| Register | Offset      | R/W | Description                                 | Reset Value |
|----------|-------------|-----|---|-------------|
| WTCRALT  | WDT_BA+0x04 | R/W | Watchdog Timer Alternative Control Register | 0x0000_0000 |

|          |    |    |    |    |    |         |    |
|----------|----|----|----|----|----|---------|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25      | 24 |
| Reserved |    |    |    |    |    |         |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17      | 16 |
| Reserved |    |    |    |    |    |         |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9       | 8  |
| Reserved |    |    |    |    |    |         |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1       | 0  |
| Reserved |    |    |    |    |    | WTRDSEL |    |

| Bits   | Description  |
|--------|--|
| [31:2] | Reserved   |
| [1:0]  | <p><b>WTRDSEL</b></p> <p><b>Watchdog Timer Reset Delay Select (Write-protection Bits)</b></p> <p>When WDT time-out happened, software has a time named WDT reset delay period to clear WDT counter to prevent WDT time-out reset happened. Software can select a suitable value of WDT reset delay period for different WDT time-out period.</p> <p>These bits are protected bit. It means programming this bit needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Reference the register REGWRPROT at address GCR_BA+0x100.</p> <p>00 = Watchdog Timer reset delay period is (1024+2) * WDT_CLK<br/>           01 = Watchdog Timer reset delay period is (128+2) * WDT_CLK<br/>           10 = Watchdog Timer reset delay period is (16+2) * WDT_CLK<br/>           11 = Watchdog Timer reset delay period is (1+2) * WDT_CLK</p> <p>This register will be reset to 0 if WDT time-out reset happened</p> |

## 12 WINDOW WATCHDOG TIMER (WWDT)

### 12.1 Overview

The purpose of Window Watchdog Timer is to perform a system reset within a specified window period to prevent software run to uncontrollable status by any unpredictable condition.

### 12.2 Features

- 6-bit down counter (WWDTVAL[5:0]) and 6-bit compare value (WWDTCCR[21:16] – WINCMP value) to make the window period flexible
- Selectable maximum 11-bit WWDT clock pre-scale (WWDTCCR[11:8] – PERIODSEL value) to make WWDT time out interval variable



## 12.4 Functional Description

The Window Watchdog Timer includes a 6-bit down counter with programmable pre-scale value to define different time out intervals.

The clock source of 6-bit Window Watchdog Timer is based on system clock divide 2048 (HCLK/2048) or internal 10 kHz oscillator with a programmable maximum 11-bit pre-scale value. And the programmable 11-bit pre-scale value is controlled by PERIODSEL (WWDTCCR[11:8] WWDT pre-scale period select) and the correlate of PERIODSEL and pre-scale value are listed in Table 12-1.

| PERIODSEL | Prescaler Value | Time out Period        | Max. Time out Interval<br>(WWDT_CLK=10 kHz) |
|-----------|-----------------|------------------------|---|
| 0000      | 1               | $1 * 64 * T_{WWDT}$    | 6.4 ms                                      |
| 0001      | 2               | $2 * 64 * T_{WWDT}$    | 12.8 ms                                     |
| 0010      | 4               | $4 * 64 * T_{WWDT}$    | 25.6 ms                                     |
| 0011      | 8               | $8 * 64 * T_{WWDT}$    | 51.2 ms                                     |
| 0100      | 16              | $16 * 64 * T_{WWDT}$   | 102.4 ms                                    |
| 0101      | 32              | $32 * 64 * T_{WWDT}$   | 204.8 ms                                    |
| 0110      | 64              | $64 * 64 * T_{WWDT}$   | 409.6 ms                                    |
| 0111      | 128             | $128 * 64 * T_{WWDT}$  | 819.2 ms                                    |
| 1000      | 192             | $192 * 64 * T_{WWDT}$  | 1.2288 s                                    |
| 1001      | 256             | $256 * 64 * T_{WWDT}$  | 1.6384 s                                    |
| 1010      | 384             | $384 * 64 * T_{WWDT}$  | 2.4576 s                                    |
| 1011      | 512             | $512 * 64 * T_{WWDT}$  | 3.2768 s                                    |
| 1100      | 768             | $768 * 64 * T_{WWDT}$  | 4.9152 s                                    |
| 1101      | 1024            | $1024 * 64 * T_{WWDT}$ | 6.5536 s                                    |
| 1110      | 1536            | $1536 * 64 * T_{WWDT}$ | 9.8304 s                                    |
| 1111      | 2048            | $2048 * 64 * T_{WWDT}$ | 13.1072 s                                   |

Table 12-1 Window Watchdog Timer Pre-scale Value Selection

Window Watchdog Timer can be enabled only once by software setting WWDTEN (WWDTCCR[0] WWDT enable) bit to 1 after chip power on or reset and the WWDT down counter will start counting from 0x3F and cannot be stopped by software unless chip has been reset again.

During WWDT counter down counting, the WWDITF (WWDTSR[0] WWDT compare match interrupt flag) is set to 1 if the WWDT counter value is equal to WINCMP (WWDTCCR[21:16] WWDT window compare register) value; if WWDITIE (WWDTCR[1] WWDT interrupt enable) is also set to 1 by software, the WWDT time-out interrupt signal is generated also while WWDITF is set to 1 by hardware.

The WWDT time-out reset signal is generated when the WWDT counter value reaches to 0. Before WWDT counter down counting to 0, software can write 0x00005AA5 to WWDTRLD register to reload WWDT internal counter value to 0x3F to prevent WWDT time-out reset happen when current WWDT counter value (WWDTCVR value) is equal or smaller than WINCMP value. If current WWDT counter value (WWDTCVR value) larger than WINCMP value and software write

0x00005AA5 to WWDTRLD register, WWDT reset signal will generate to cause chip be reset. Figure 12-3 shows the reset and reload behavior of WWDT.

To prevent program runs to disable Window Watchdog Timer counter counting unexpected, the control register WWDTCR can only be written once after chip is powered on or reset. Software cannot disable Window Watchdog Timer counter counting (WWDTEN), change time-out pre-scale period (PERIODSEL) or change window compare value (WINCMP) while WWDTEN bit has been enabled by software unless chip is reset.

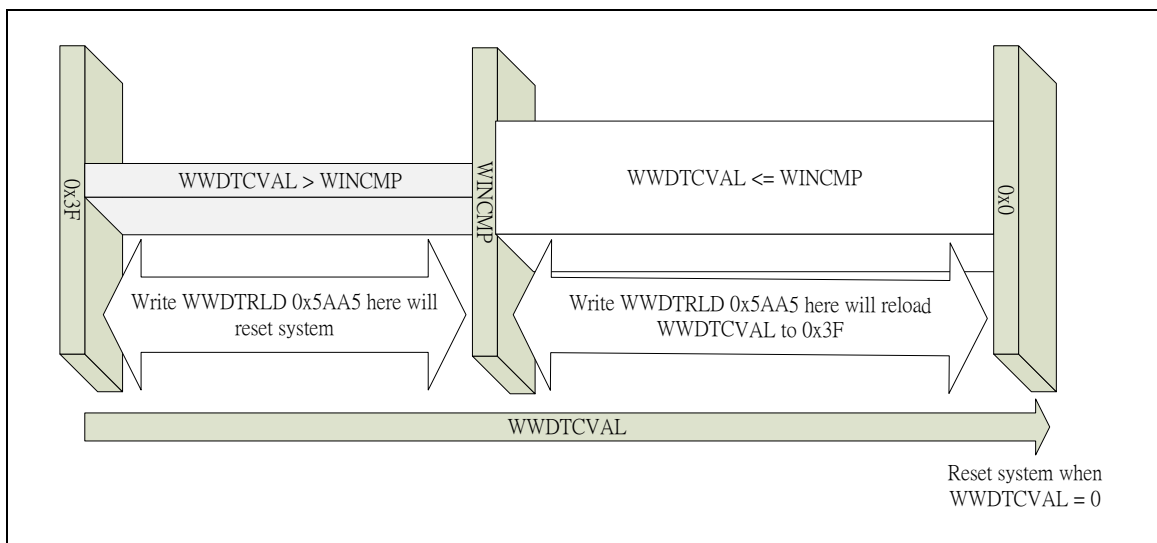


Figure 12-3 Window Watchdog Timer reset and reload behavior

When software writes 0x00005AA5 to WWDTRLD register to reload WWDT counter value to 0x3F, it needs 3 WWDT clocks to sync reload command to actually perform reload action. It means if software set PERIODSEL (WWDTCR[11:8] WWDT pre-scale period select) to 0, the pre-scale value should be as 1, and the WINCMP (WWDTCR[21:16] WWDT window compare register) value must larger than 2, otherwise software writes WWDTRLD to reload WWDT counter value to 0x3F is unavailable and WWDT time-out reset always happened. Table 12-2 shows the limitation of WINCMP.

| Pre-scale Value | Valid WINCMP Value |
|-----------------|--------------------|
| 1               | 0x3 ~ 0x3F         |
| 2               | 0x2 ~ 0x3F         |
| Others          | 0x0 ~ 0x3F         |

Table 12-2 WINCMP Setting Limitation



### 12.5 Register Map

R: read only, W: write only, R/W: both read and write

| Register                     | Offset       | R/W | Description                                   | Reset Value |
|------------------------------|--------------|-----|---|-------------|
| <b>WWDT Base Address:</b>    |              |     |   |             |
| <b>WWDT_BA = 0x4000_4100</b> |              |     |   |             |
| <b>WWDTRLD</b>               | WWDT_BA+0x00 | W   | Window Watchdog Timer Reload Counter Register | 0x0000_0000 |
| <b>WWDTCR</b>                | WWDT_BA+0x04 | R/W | Window Watchdog Timer Control Register        | 0x003F_0800 |
| <b>WWDTSR</b>                | WWDT_BA+0x08 | R/W | Window Watchdog Timer Status Register         | 0x0000_0000 |
| <b>WWDTCVR</b>               | WWDT_BA+0x0C | R   | Window Watchdog Timer Counter Value Register  | 0x0000_003F |

### 12.6 Register Description

#### Window Watchdog Timer Reload Counter Register (WWDTRLD)

| Register | Offset       | R/W | Description                                   | Reset Value |
|----------|--------------|-----|---|-------------|
| WWDTRLD  | WWDT_BA+0x00 | W   | Window Watchdog Timer Reload Counter Register | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| WWDTRLD[31:24] |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| WWDTRLD[23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| WWDTRLD[15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| WWDTRLD[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description  |
|--------|--|
| [31:0] | <p><b>WWDTRLD</b></p> <p><b>WWDTRLD Reload Counter Register</b></p> <p>Writing 0x00005AA5 to this register will reload the Window Watchdog Timer counter value to 0x3F.</p> <p><b>Note:</b> Software can only write WWDTRLD to reload WWDTRLD counter value when current WWDTRLD counter value between 0 and WINCMP. If software writes WWDTRLD when current WWDTRLD counter value larger than WINCMP, WWDTRLD reset signal will generate immediately.</p> |

Window Watchdog Timer Control Register (WWDTCR)

| Register | Offset         | R/W | Description                            | Reset Value |
|----------|----------------|-----|--|-------------|
| WWDTCR   | WWDTCR_BA+0x04 | R/W | Window Watchdog Timer Control Register | 0x003F_0800 |

**Note:** This register can be written only one time after chip is powered on or reset.

|             |          |        |    |           |    |        |        |
|-------------|----------|--------|----|-----------|----|--------|--------|
| 31          | 30       | 29     | 28 | 27        | 26 | 25     | 24     |
| DBGACK_WWDT | Reserved |        |    |           |    |        |        |
| 23          | 22       | 21     | 20 | 19        | 18 | 17     | 16     |
| Reserved    |          | WINCMP |    |           |    |        |        |
| 15          | 14       | 13     | 12 | 11        | 10 | 9      | 8      |
| Reserved    |          |        |    | PERIODSEL |    |        |        |
| 7           | 6        | 5      | 4  | 3         | 2  | 1      | 0      |
| Reserved    |          |        |    |           |    | WWDTIE | WWDTEN |

| Bits    | Description |  |                 |                              |  |
|---------|-------------|--|-----------------|------------------------------|--|
| [31]    | DBGACK_WWDT | <b>ICE debug mode acknowledge Disable</b><br>0 = WWDT counter stopped if system is in Debug mode.<br>1 = WWDT still counted even system is in Debug mode.  |                 |                              |  |
| [30:22] | Reserved    | Reserved   |                 |                              |  |
| [21:16] | WINCMP      | <b>WWDT Window Compare Register</b><br>Set this register to adjust the valid reload window.<br><b>Note:</b> Software can only write WWDTRLD to reload WWDT counter value when current WWDT counter value between 0 and WINCMP. If Software writes WWDTRLD when current WWDT counter value larger than WINCMP, WWDT reset signal will generate immediately. |                 |                              |  |
| [15:12] | Reserved    | Reserved   |                 |                              |  |
| [11:8]  | PERIODSEL   | <b>WWDT Pre-scale Period Select</b><br>These 4-bit select the pre-scale period for the WWDT counter period.  |                 |                              |  |
|         |             | PERIODSEL  | Pre-scale Value | Time out Period              | Max. Time out Interval (WWDT_CLK=10 kHz) |
|         |             | 0000   | 1               | 1 * 64 * T <sub>WWDT</sub>   | 6.4 ms                                   |
|         |             | 0001   | 2               | 2 * 64 * T <sub>WWDT</sub>   | 12.8 ms                                  |
|         |             | 0010   | 4               | 4 * 64 * T <sub>WWDT</sub>   | 25.6 ms                                  |
|         |             | 0011   | 8               | 8 * 64 * T <sub>WWDT</sub>   | 51.2 ms                                  |
|         |             | 0100   | 16              | 16 * 64 * T <sub>WWDT</sub>  | 102.4 ms                                 |
|         |             | 0101   | 32              | 32 * 64 * T <sub>WWDT</sub>  | 204.8 ms                                 |
|         |             | 0110   | 64              | 64 * 64 * T <sub>WWDT</sub>  | 409.6 ms                                 |
|         |             | 0111   | 128             | 128 * 64 * T <sub>WWDT</sub> | 819.2 ms                                 |

|       |                 |  |      |                               |           |
|-------|-----------------|--|------|-------------------------------|-----------|
|       |                 | 1000   | 192  | $192 * 64 * T_{\text{WWDT}}$  | 1.2288 s  |
|       |                 | 1001   | 256  | $256 * 64 * T_{\text{WWDT}}$  | 1.6384 s  |
|       |                 | 1010   | 384  | $384 * 64 * T_{\text{WWDT}}$  | 2.4576 s  |
|       |                 | 1011   | 512  | $512 * 64 * T_{\text{WWDT}}$  | 3.2768 s  |
|       |                 | 1100   | 768  | $768 * 64 * T_{\text{WWDT}}$  | 4.9152 s  |
|       |                 | 1101   | 1024 | $1024 * 64 * T_{\text{WWDT}}$ | 6.5536 s  |
|       |                 | 1110   | 1536 | $1536 * 64 * T_{\text{WWDT}}$ | 9.8304 s  |
|       |                 | 1111   | 2048 | $2048 * 64 * T_{\text{WWDT}}$ | 13.1072 s |
| [7:2] | <b>Reserved</b> | Reserved   |      |                               |           |
| [1]   | <b>WWDTIE</b>   | <b>WWDT Interrupt Enable</b><br>Setting this bit to enable the Window Watchdog Timer time-out interrupt function.<br>0 = WWDT time-out interrupt function Disabled if WWDTIF (WWDTSR[0] WWDT compare match interrupt flag) is 1.<br>1 = WWDT time-out interrupt function Enabled if WWDTIF (WWDTSR[0] WWDT compare match interrupt flag) is 1. |      |                               |           |
| [0]   | <b>WWDTEN</b>   | <b>WWDT Enable</b><br>Set this bit to enable Window Watchdog Timer counter counting.<br>0 = Window Watchdog Timer counter is stopped.<br>1 = Window Watchdog Timer counter is starting counting.   |      |                               |           |

**Window Watchdog Timer Status Register (WWDTSR)**

| Register | Offset       | R/W | Description                           | Reset Value |
|----------|--------------|-----|---------------------------------------|-------------|
| WWDTSR   | WWDT_BA+0x08 | R/W | Window Watchdog Timer Status Register | 0x0000_0000 |

|          |    |    |    |    |    |        |        |
|----------|----|----|----|----|----|--------|--------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25     | 24     |
| Reserved |    |    |    |    |    |        |        |
| 23       | 22 | 21 | 20 | 19 | 18 | 17     | 16     |
| Reserved |    |    |    |    |    |        |        |
| 15       | 14 | 13 | 12 | 11 | 10 | 9      | 8      |
| Reserved |    |    |    |    |    |        |        |
| 7        | 6  | 5  | 4  | 3  | 2  | 1      | 0      |
| Reserved |    |    |    |    |    | WWDTRF | WWDTIF |

| Bits   | Description |   |
|--------|-------------|---|
| [31:2] | Reserved    | Reserved  |
| [1]    | WWDTRF      | <b>WWDT Reset Flag</b><br>When WWDT counter counts down to 0 or writes WWDTRLN during current WWDT counter value larger than WINCMP, chip will be reset and this bit is set to 1. This bit will be cleared to 0 by writing 1 to itself. |
| [0]    | WWDTIF      | <b>WWDT Compare Match Interrupt Flag</b><br>When current WWDT counter value matches to WWCMP, this bit is set to 1. This bit will be cleared by writing 1 to itself.  |

### Window Watchdog Timer Counter Value Register (WWDTTCVR)

| Register | Offset       | R/W | Description                                  | Reset Value |
|----------|--------------|-----|--|-------------|
| WWDTTCVR | WWDT_BA+0x0C | R   | Window Watchdog Timer Counter Value Register | 0x0000_003F |

|          |    |           |    |    |    |    |    |
|----------|----|-----------|----|----|----|----|----|
| 31       | 30 | 29        | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |           |    |    |    |    |    |
| 23       | 22 | 21        | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |           |    |    |    |    |    |
| 15       | 14 | 13        | 12 | 11 | 10 | 9  | 8  |
| Reserved |    |           |    |    |    |    |    |
| 7        | 6  | 5         | 4  | 3  | 2  | 1  | 0  |
| Reserved |    | WWDTTCVAL |    |    |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:6] | Reserved    | Reserved  |
| [5:0]  | WWDTTCVAL   | <b>WWDT Counter Value</b><br>This register reflects the current WWDT counter value and this register is read only |

## 13 BASIC PWM GENERATOR AND CAPTURE TIMER (BPWM)

### 13.1 Overview

The NuMicro™ NM15xx series has 1 sets of BPWM group supporting 1 sets of PWM generators that can be configured as 2 independent PWM outputs, PWM20~PWM21, or as 1 complementary PWM pairs, (PWM20, PWM21) with programmable Dead-zone generators.

The PWM generator has one 8-bit pre-scalar, one clock divider with 5 divided frequencies (1, 1/2, 1/4, 1/8, 1/16), two PWM Timers including two clock selectors, two 16-bit PWM down-counters for PWM period control, two 16-bit comparators for PWM duty control and one dead-zone generator. The PWM generator provides two independent PWM interrupt flags which are set by hardware when the corresponding PWM period down counter reaches zero. Each PWM interrupt source with its corresponding enable bit can cause CPU to request PWM interrupt. The PWM generators can be configured as one-shot mode to produce only one PWM cycle signal or auto-reload mode to output PWM waveform continuously.

When PCR.DZEN01 is set, PWM20 and PWM21 perform complementary; the paired PWM timing, period, duty and dead-time are determined by PWM0 timer and Dead-zone generator 0. Refer to Figure 13-1 for the architecture of Basic PWM Timers.

To prevent PWM driving output pin from glitches, the 16-bit period down counter and 16-bit comparator are implemented with double buffer. When user writes data to counter/comparator buffer registers the updated value will be load into the 16-bit down counter/ comparator at the time down counter reaching zero. The double buffering feature avoids glitch at PWM outputs.

When the 16-bit period down counter reaches zero, the interrupt request is generated. If PWM-timer is set as auto-reload mode, when the down counter reaches zero, it is reloaded with PWM Counter Register (CNRx) automatically then start decreasing, repeatedly. If the PWM-timer is set as one-shot mode, the down counter will stop and generate one interrupt request when it reaches zero.

The value of PWM counter comparator is used for pulse high width modulation. The counter control logic changes the output to high level when down-counter value matches the value of compare register.

The alternate feature of the PWM timer is digital input capture function. If capture function is enabled the PWM output pin is switched as capture input mode. The capture20 and PWM20 share one timer which is included in PWM20; and the capture21 and PWM21 share PWM21 timer, and etc. Therefore user must set the PWM timer before enable capture feature. After capture feature is enabled, the capture always latched PWM counter to Capture Rising Latch Register (CRLR) when input channel has a rising transition and latched PWM-counter to Capture Falling Latch Register (CFLR) when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting CCR.CRL\_IE0[1] (Rising latch Interrupt enable) and CCR.CFL\_IE0[2] (Falling latch Interrupt enable) to decide the condition of interrupt occur. Capture channel 1 has the same feature by setting CCR.CRL\_IE1[17] and CCR.CFL\_IE1[18].

The maximum captured frequency that PWM can capture is confined by the capture interrupt latency. When capture interrupt occurred, software will do at least three steps, they are: Read PIIR to get interrupt source and Read PWM\_CRLx/PWM\_CFLx(x=0~1) to get capture value and finally write 1 to clear PIIR to zero. If interrupt latency will take time T0 to finish, the capture signal mustn't transition during this interval (T0). In this case, the maximum capture frequency will be 1/T0. For example:

HCLK = 50MHz, BPWM\_CLK = 25MHz, Interrupt latency is 900 ns

So the maximum capture frequency will is 1/900ns ≈ 1000 kHz

### 13.2 Features

#### 13.2.1 PWM Function:

- One PWM generator which supports one 8-bit pre-scalar, one clock divider, two PWM timers (down counter), one dead-zone generator and two PWM outputs.
- Up to 16-bit resolution
- PWM Interrupt request synchronized with PWM period
- One-shot or Auto-reload mode PWM
- Edge-aligned type or Center-aligned type option

#### 13.2.2 Capture Function:

- Timing control logic shared with PWM generators
- Supports 2 Capture input channels shared with 2 PWM output channels
- Each channel supports one rising latch register (CRLR), one falling latch register (CFLR) and Capture interrupt flag (CAPIFx)



### 13.3 Block Diagram

The Figure 13-1 and Figure 13-2 illustrate the architecture of PWM

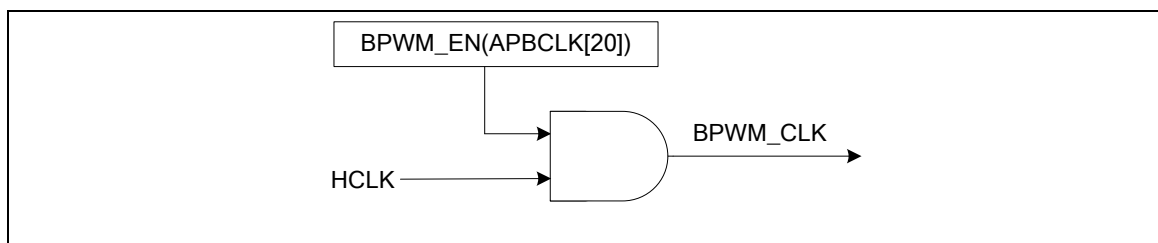


Figure 13-1 PWM Clock Source Control

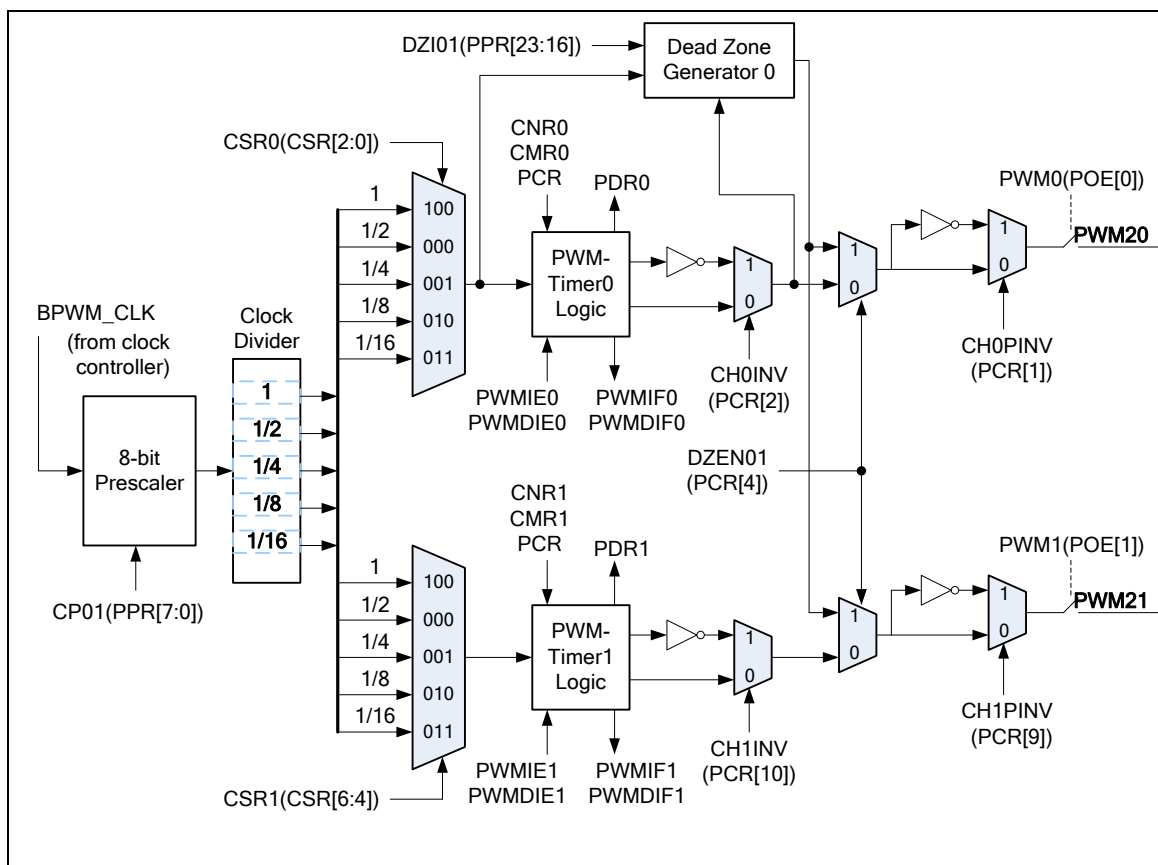


Figure 13-2 PWM Architecture Diagram

## 13.4 PWM-Timer Operation

PWM controller supports 2 operation types: Edge-aligned and Center-aligned type.

### 13.4.1 Edge-aligned PWM (down-counter)

In Edge-aligned PWM Output mode, the 16 bits PWM counter will starts down-counting from CNRn to match with the value of the duty cycle CMRn (old), when this happen it will toggle the PWMn generator output to low. The counter will continue down-counting to 0, at this moment, it toggles the PWMn generator output to high and CMRn(new) and CNRn(new) are updated with CHnMODE=1 and request the PWM interrupt if PWM interrupt is enabled(PIER.n=1).

The PWM period and duty control are configured by PWM down-counter register (CNR) and PWM comparator register (CMR). The PWM-timer timing operation is shown in Figure 13-4. The pulse width modulation follows the formula below and the legend of PWM-Timer Comparator is shown as Figure 13-3. Note that the corresponding GPIO pins must be configured as PWM function (enable POE and disable CAPENR) for the corresponding PWM channel.

- PWM frequency =  $\text{BPWM\_CLK} / [(\text{prescale} + 1) * (\text{clock divider}) * (\text{CNR} + 1)]$ .
- Duty ratio =  $(\text{CMR} + 1) / (\text{CNR} + 1)$
- $\text{CMR} \geq \text{CNR}$ : PWM output is always high
- $\text{CMR} < \text{CNR}$ : PWM low width =  $(\text{CNR} - \text{CMR})$  unit[1]; PWM high width =  $(\text{CMR} + 1)$  unit
- $\text{CMR} = 0$ : PWM low width =  $(\text{CNR})$  unit; PWM high width = 1 unit

**Note:** [1] Unit = one PWM clock cycle.

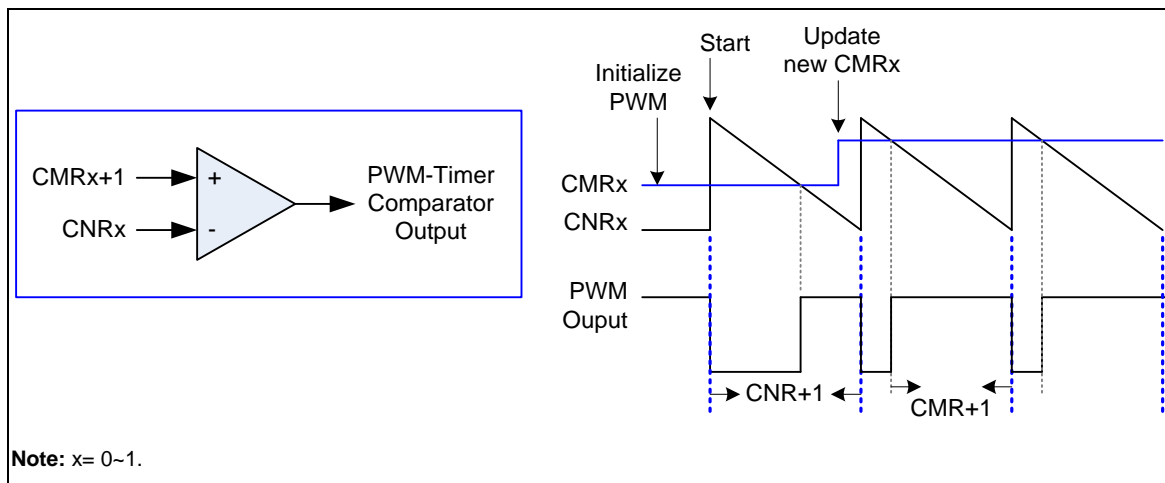


Figure 13-3 Legend of Internal Comparator Output of PWM-Timer

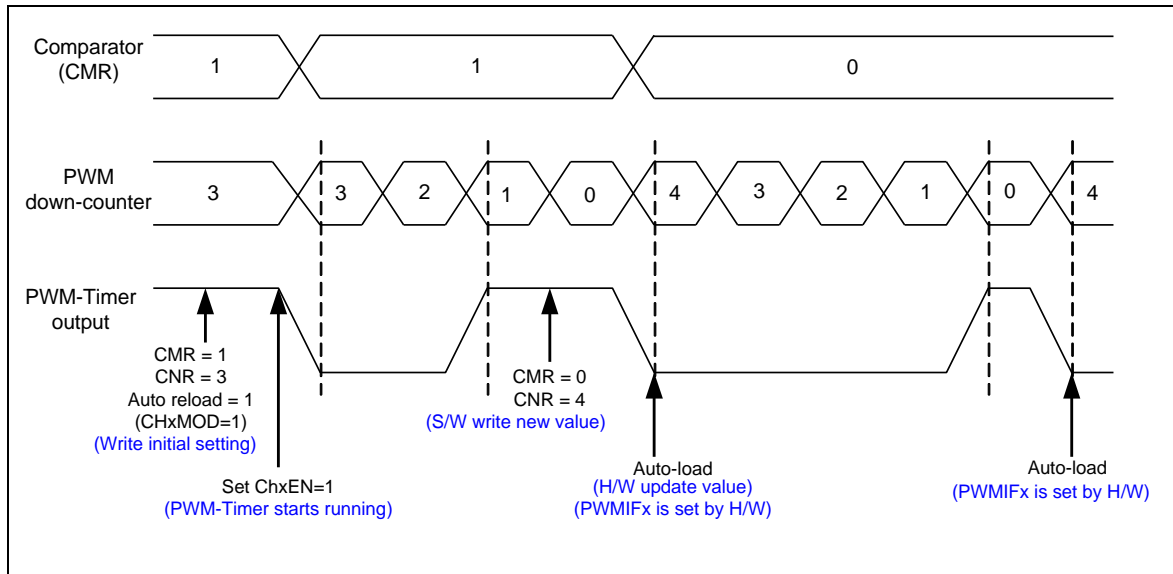


Figure 13-4 PWM-Timer Operation Timing

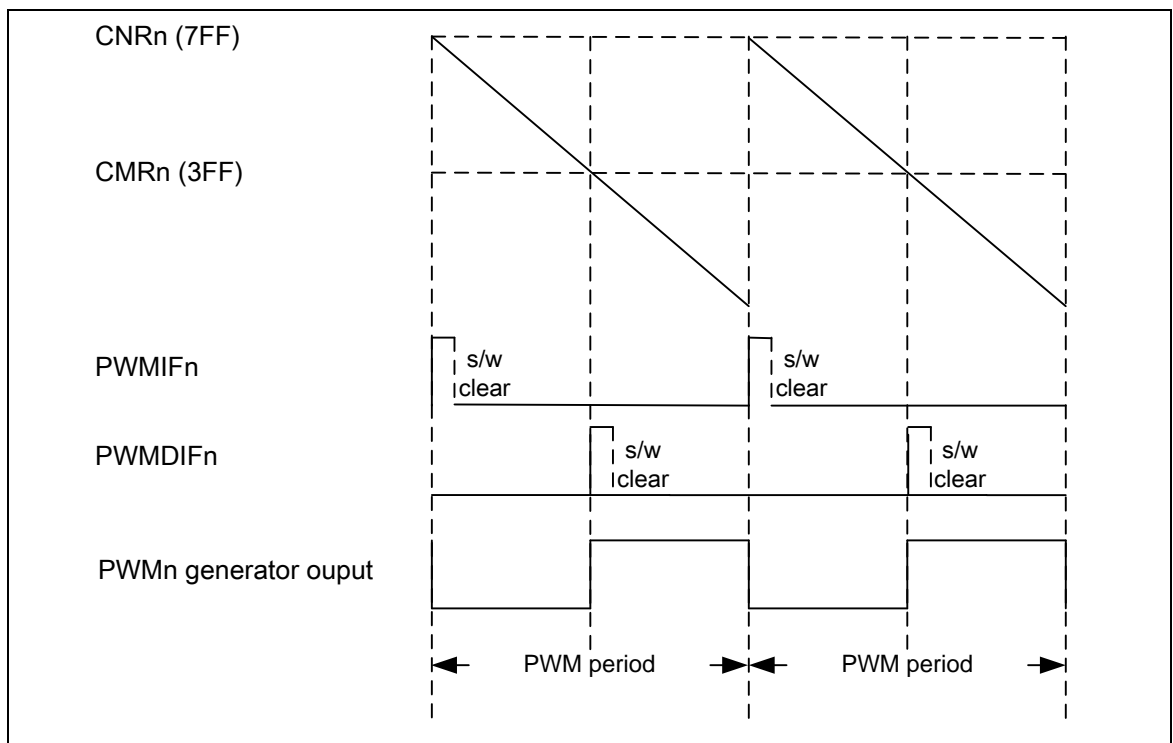


Figure 13-5 PWM Edge-aligned Interrupt Generate Timing Waveform

### 13.4.2 Center-aligned PWM (up/down-counter)

The Center-aligned PWM signals are produced by the module when the PWM time base is configured in an Up/Down Counting mode. The PWM counter will start counting-up from 0 to match the value of CMRn (old); this will cause the toggling of the PWMn generator output to low. The counter will continue counting to match with the CNRn (old). Upon reaching this states counter is configured automatically to down counting, when PWM counter matches the CMRn (old) value again the PWMn generator output toggles to high. Once the PWM counter underflows it will update the PWM period register CNRn(new) and duty cycle register CMRn(new) with CHnMODE = 1.

In Center-aligned type, the PWM period interrupt is requested at down-counter underflow if INTTYPE (PIER[17:16]) = 0, i.e. at start (end) of each PWM cycle or at up-counter matching with CNRn if INTTYPE (PIER[17:16]) = 1, i.e. at center point of PWM cycle.

- PWM frequency =  $\text{BPWM\_CLK} / [(\text{prescale} + 1) * (\text{clock divider}) * (\text{CNR} + 1)]$ .
- Duty ratio =  $[(2 \times \text{CMR}) + 1] / [2 \times (\text{CNR} + 1)]$
- $\text{CMR} > \text{CNR}$ : PWM output is always high
- $\text{CMR} \leq \text{CNR}$ : PWM low width =  $2 \times (\text{CNR} - \text{CMR}) + 1$  unit[1]; PWM high width =  $(2 \times \text{CMR}) + 1$  unit
- $\text{CMR} = 0$ : PWM low width =  $2 \times \text{CNR} + 1$  unit; PWM high width = 1 unit

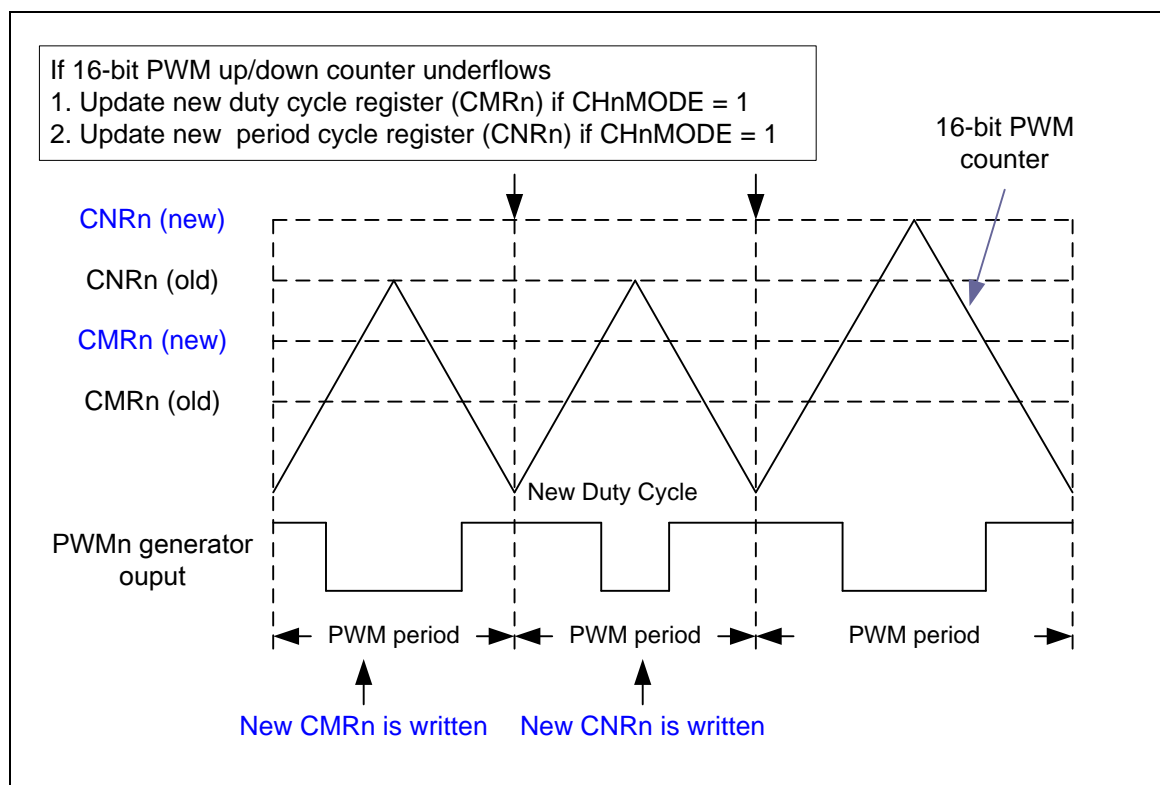


Figure 13-6 Center-aligned Type Output Waveform

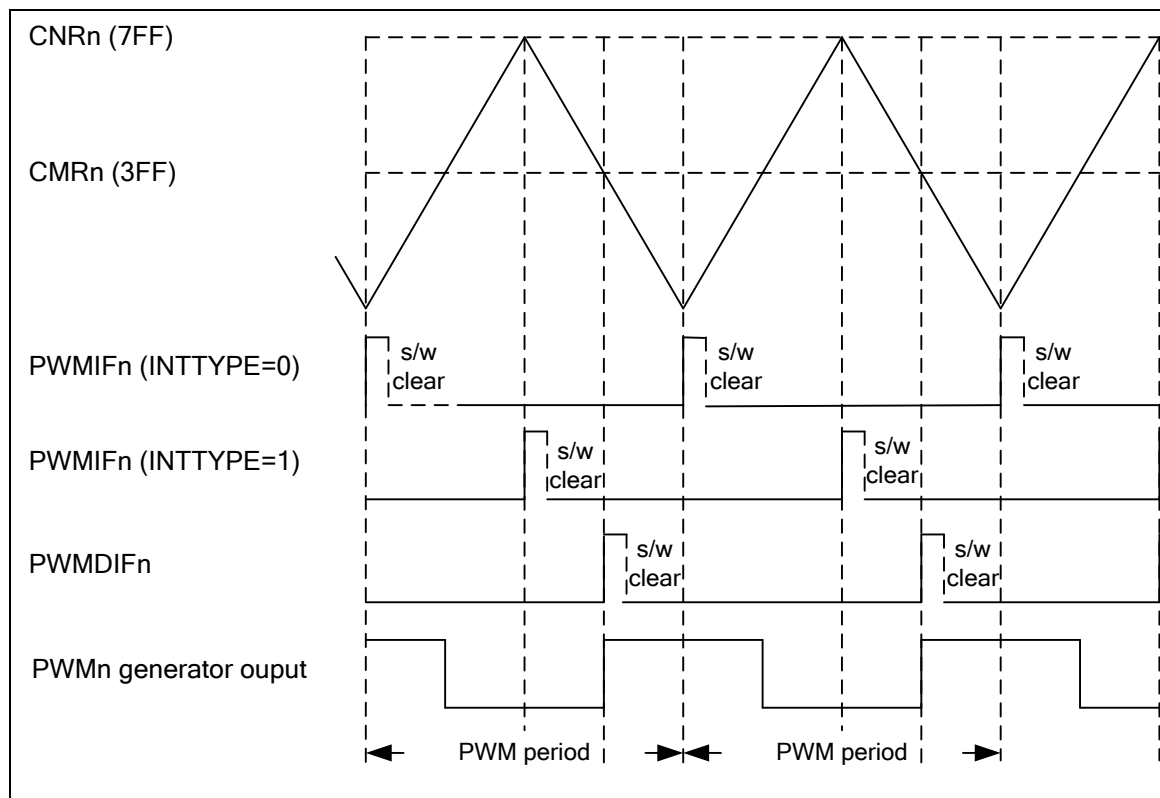


Figure 13-7 PWM Center-aligned Interrupt Generate Timing Waveform

### 13.4.3 PWM Double Buffering, Auto-reload and One-shot Operation

PWM Timers have double buffering function the reload value is updated at the start of next period without affecting current timer operation. The PWM counter value can be written into CNRx and current PWM counter value can be read from PDRx.

PWM0 will operate in One-shot mode if CH0MOD bit is set to 0, and operate in Auto-reload mode if CH0MOD bit is set to 1. It is recommend that switch PWM0 operating mode before set CH0EN bit to 1 to enable PWM0 counter start running because the content of CNR0 and CMR0 will be cleared to 0 to reset the PWM0 period and duty setting when PWM0 operating mode is changed. As PWM0 operate in One-shot mode, CMR0 and CNR0 should be written first and then set CH0EN bit to 1 to enable PWM0 counter start running. After PWM0 counter down count from CNR0 value to 0, CNR0 and CMR0 will be cleared to 0 by hardware and PWM counter will be held. Software need to write new CMR0 and CNR0 value to set next one-shot period and duty. When re-start next one-shot operation, the CMR0 should be written first because PWM0 counter will auto re-start counting when CNR0 is written a non-zero value. As PWM0 operates at auto-reload mode, CMR0 and CNR0 should be written first and then set CH0EN bit to 1 to enable PWM0 counter start running. The value of CNR0 will reload to PWM0 counter when it down count reaches 0. If CNR0 is set to 0, PWM0 counter will be held. PWM1 performs the same function as PWM0.

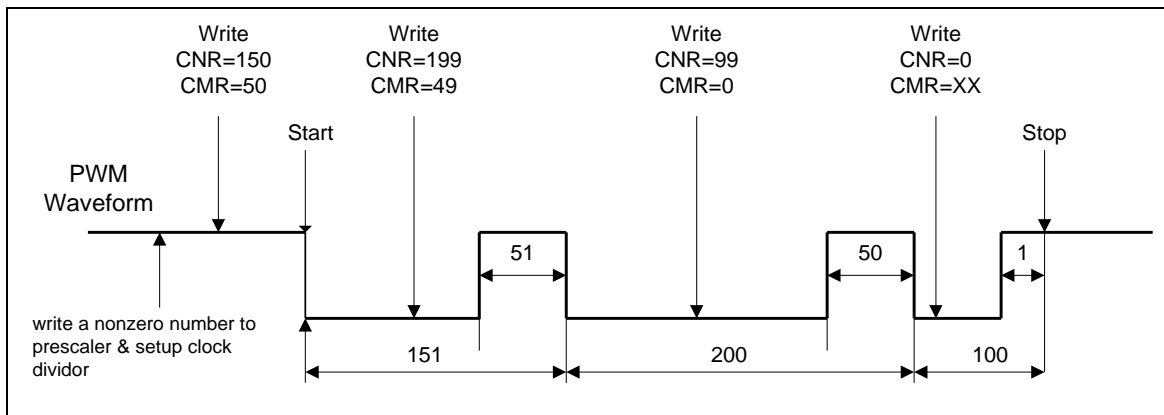


Figure 13-8 PWM Double Buffering Illustration

#### 13.4.4 Modulate Duty Ratio

The double buffering function allows CMRx written at any point in current cycle. The loaded value will take effect from next cycle.

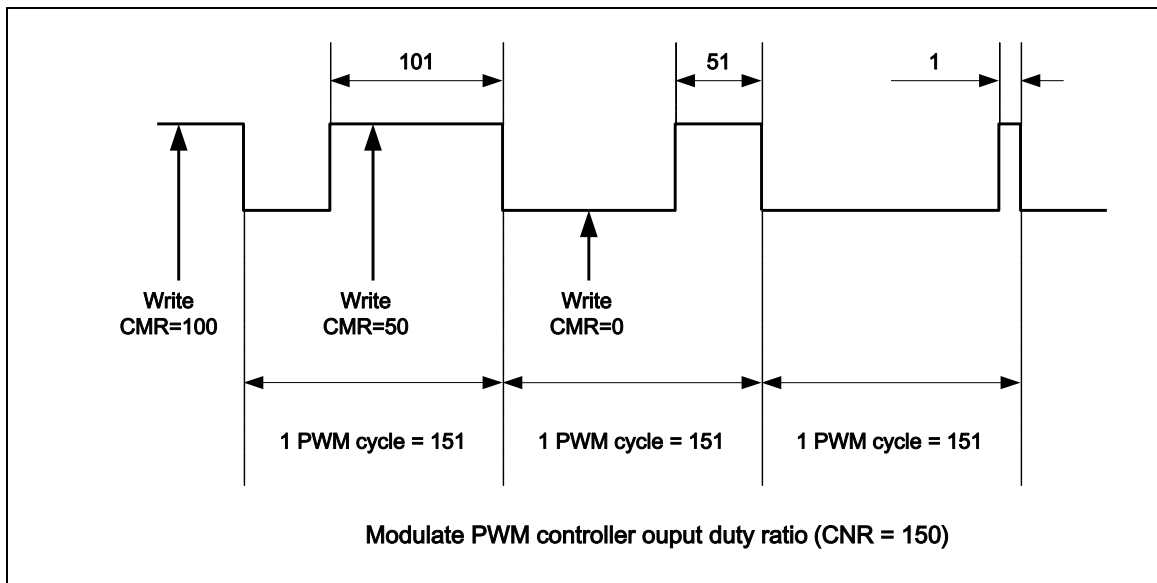


Figure 13-9 PWM Controller Output Duty Ratio

#### 13.4.5 Dead-Zone Generator

The PWM controller is implemented with Dead-zone generator. They are built for power device protection. This function generates a programmable time gap to delay PWM rising output. User can program PPR.DZI to determine the Dead-zone interval.

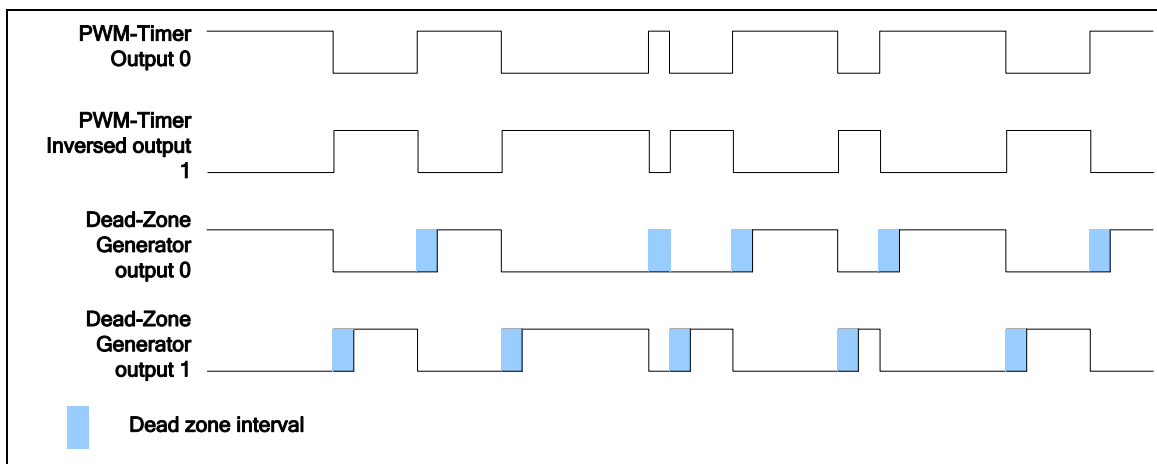


Figure 13-10 Paired-PWM Output with Dead-zone Generation Operation

### 13.4.6 Capture Operation

The Capture 0 and PWM 0 share one timer that included in PWM 0; and the Capture 1 and PWM 1 share another timer, and etc. The capture always latches PWM-counter to CRLRx when input channel has a rising transition and latches PWM-counter to CFLRx when input channel has a falling transition. Capture channel 0 interrupt is programmable by setting CCR[1] (Rising latch Interrupt enable) and CCR[2] (Falling latch Interrupt enable) to decide the condition of interrupt occur. Capture channel 1 has the same feature by setting CCR[17] and CCR[18], and etc. Whenever the Capture controller issues a capture interrupt, the corresponding PWM counter will be reloaded with CNRx at this moment. Note that the corresponding GPIO pins must be configured as capture function (POE disabled and CAPENR enabled) for the corresponding capture channel.

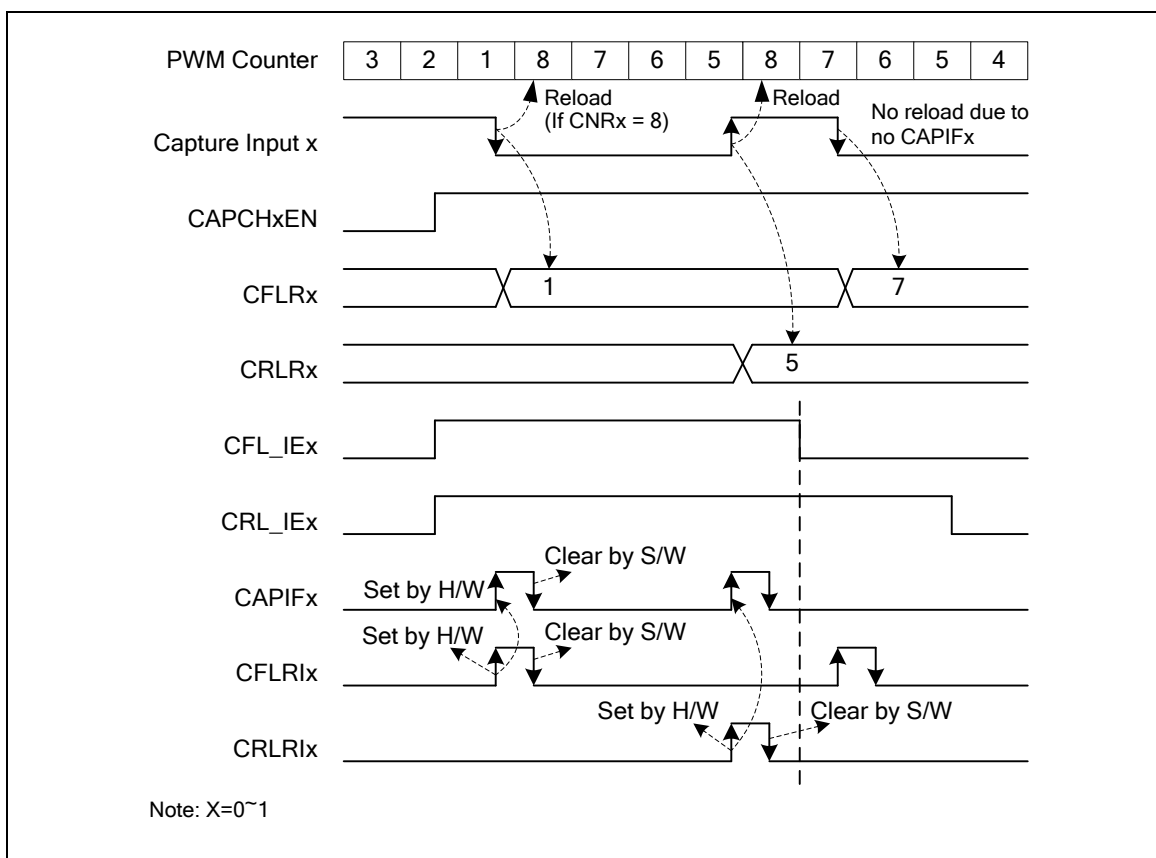


Figure 13-11 Capture Operation Timing

In this case, the CNR is 8:

1. The PWM counter will be reloaded with CNRx when a capture interrupt flag (CAPIFx) is set.
2. The channel low pulse width is  $(CNR + 1 - CRLR)$ .
3. The channel high pulse width is  $(CNR + 1 - CFLR)$ .



### 13.4.7 PWM-Timer Interrupt Architecture

There are two PWM interrupts, BPWM0\_INT and BPWM1\_INT. BPWM0 and capture0 share one interrupt, BPWM1 and capture1 share the same interrupt. Therefore, PWM function and Capture function in the same channel cannot be used at the same time. Figure 13-12 demonstrates the architecture of PWM Timer interrupts.

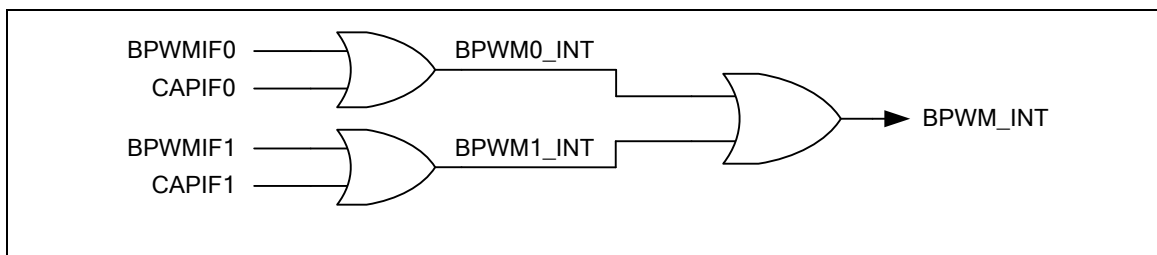


Figure 13-12 PWM Interrupt Architecture Diagram

### 13.4.8 PWM-Timer Start Procedure

The following procedure is recommended for starting a PWM drive.

1. Set clock source divider select register (CSR)
2. Set prescaler (PPR)
3. Set inverter on/off, Dead-zone generator on/off, Auto-reload/One-shot mode and Stop PWM-timer (PCR)
4. Set comparator register (CMR) for setting PWM duty.
5. Set PWM down-counter register (CNR) for setting PWM period.
6. Set interrupt enable register (PIER) (optional)
7. Set corresponding GPIO pins as PWM function (enable POE and disable CAPENR) for the corresponding PWM channel.
8. Enable PWM timer start running (Set CHxEN = 1 in PCR)

### 13.4.9 PWM-Timer Re-Start Procedure in Single-shot mode

After PWM waveform is generated once in PWM One-shot mode, PWM-Timer will be stopped automatically. The following procedure is recommended for re-starting PWM single-shot waveform.

- Set comparator register (CMR) for setting PWM duty.
- Set PWM down-counter register (CNR) for setting PWM period. After setting CNR, PWM wave will be generated.

### 13.4.10 PWM-Timer Stop Procedure

#### Method 1:

Set 16-bit counter (CNR) as 0, and monitor PDR (current value of 16-bit down-counter). When PDR reaches to 0, disable PWM-Timer (CHxEN in PCR). **(Recommended)**

#### Method 2:

Set 16-bit counter (CNR) as 0. When interrupt request happened, disable PWM-Timer (CHxEN in PCR). **(Recommended)**

#### Method 3:

Disable PWM-Timer directly ((CHxEN in PCR). **(Not recommended)**

The reason why method 3 is not recommended is that disable CHxEN will immediately stop PWM output signal and lead to change the duty of the PWM output, this may cause damage to the control circuit of motor

### 13.4.11 Capture Start Procedure

1. Set clock source divider select register (CSR)
2. Set prescaler (PPR)
3. Set channel enabled, rising/falling interrupt enable and input signal inverter on/off (CCR)
4. Set Auto-reload mode, Edge-aligned type and Stop PWM-timer (PCR)
5. Set PWM down-counter (CNR)
6. Enable PWM timer start running (Set CHxEN = 1 in PCR)
7. Set corresponding GPIO pins as capture function (disable POE and enable CAPENR) for the corresponding PWM channel.

### 13.5 Register Map

R: read only, W: write only, R/W: both read and write

| Register                     | Offset       | R/W | Description  | Reset Value |
|------------------------------|--------------|-----|--|-------------|
| <b>BPWM Base Address:</b>    |              |     |  |             |
| <b>BPWM_BA = 0x4004_0000</b> |              |     |  |             |
| <b>PPR</b>                   | BPWM_BA+0x00 | R/W | Basic PWM Pre-scalar Register                        | 0x0000_0000 |
| <b>CSR</b>                   | BPWM_BA+0x04 | R/W | Basic PWM Clock Source Divider Select Register       | 0x0000_0000 |
| <b>PCR</b>                   | BPWM_BA+0x08 | R/W | Basic PWM Control Register                           | 0x0000_0000 |
| <b>CNR0</b>                  | BPWM_BA+0x0C | R/W | Basic PWM Counter Register 0                         | 0x0000_0000 |
| <b>CMR0</b>                  | BPWM_BA+0x10 | R/W | Basic PWM Comparator Register 0                      | 0x0000_0000 |
| <b>PDR0</b>                  | BPWM_BA+0x14 | R   | Basic PWM Data Register 0                            | 0x0000_0000 |
| <b>CNR1</b>                  | BPWM_BA+0x18 | R/W | Basic PWM Counter Register 1                         | 0x0000_0000 |
| <b>CMR1</b>                  | BPWM_BA+0x1C | R/W | Basic PWM Comparator Register 1                      | 0x0000_0000 |
| <b>PDR1</b>                  | BPWM_BA+0x20 | R   | Basic PWM Data Register 1                            | 0x0000_0000 |
| <b>PIER</b>                  | BPWM_BA+0x40 | R/W | Basic PWM Interrupt Enable Register                  | 0x0000_0000 |
| <b>PIIR</b>                  | BPWM_BA+0x44 | R/W | Basic PWM Interrupt Indication Register              | 0x0000_0000 |
| <b>CCR</b>                   | BPWM_BA+0x50 | R/W | Basic PWM Capture Control Register                   | 0x0000_0000 |
| <b>CRLR0</b>                 | BPWM_BA+0x58 | R   | Basic PWM Capture Rising Latch Register (Channel 0)  | 0x0000_0000 |
| <b>CFLR0</b>                 | BPWM_BA+0x5C | R   | Basic PWM Capture Falling Latch Register (Channel 0) | 0x0000_0000 |
| <b>CRLR1</b>                 | BPWM_BA+0x60 | R   | Basic PWM Capture Rising Latch Register (Channel 1)  | 0x0000_0000 |
| <b>CFLR1</b>                 | BPWM_BA+0x64 | R   | Basic PWM Capture Falling Latch Register (Channel 1) | 0x0000_0000 |
| <b>CAPENR</b>                | BPWM_BA+0x78 | R/W | Basic PWM Capture Input Enable Register              | 0x0000_0000 |
| <b>POE</b>                   | BPWM_BA+0x7C | R/W | Basic PWM Output Enable                              | 0x0000_0000 |

### 13.6 Register Description

#### PWM Pre-Scale Register (PPR)

| Register | Offset       | R/W | Description                   | Reset Value |
|----------|--------------|-----|-------------------------------|-------------|
| PPR      | BPWM_BA+0x00 | R/W | Basic PWM Pre-scalar Register | 0x0000_0000 |

|       |    |    |    |    |    |    |    |
|-------|----|----|----|----|----|----|----|
| 31    | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -     |    |    |    |    |    |    |    |
| 23    | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| DZI01 |    |    |    |    |    |    |    |
| 15    | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -     |    |    |    |    |    |    |    |
| 7     | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CP01  |    |    |    |    |    |    |    |

| Bits    | Description  |  |
|---------|--------------|--|
| [31:24] | -            | <b>Reserved.</b>   |
| [23:16] | <b>DZI01</b> | <b>Dead-zone Interval for Pair of Channel 0 and Channel 1</b><br>These 8-bit determine the Dead-zone length.<br>The unit time of Dead-zone length = [(prescale+1)*(clock source divider)] / BPWM_CLK                                   |
| [15:8]  | -            | <b>Reserved.</b>   |
| [7:0]   | <b>CP01</b>  | <b>Clock Prescaler</b><br>Clock input is divided by (CP01 + 1) before it is fed to the corresponding PWM-timer<br>If CP01=0, then the clock prescaler 0 output clock will be stopped. So corresponding PWM-timer will also be stopped. |

### PWM Clock Source Divider Select Register (CSR)

| Register | Offset       | R/W | Description                                    | Reset Value |
|----------|--------------|-----|--|-------------|
| CSR      | BPWM_BA+0x04 | R/W | Basic PWM Clock Source Divider Select Register | 0x0000_0000 |

|          |      |    |    |          |      |    |    |
|----------|------|----|----|----------|------|----|----|
| 31       | 30   | 29 | 28 | 27       | 26   | 25 | 24 |
| Reserved |      |    |    |          |      |    |    |
| 23       | 22   | 21 | 20 | 19       | 18   | 17 | 16 |
| Reserved |      |    |    |          |      |    |    |
| 15       | 14   | 13 | 12 | 11       | 10   | 9  | 8  |
| Reserved |      |    |    |          |      |    |    |
| 7        | 6    | 5  | 4  | 3        | 2    | 1  | 0  |
| Reserved | CSR1 |    |    | Reserved | CSR0 |    |    |

| Bits   | Description |  |                        |                        |     |   |     |    |     |   |     |   |     |   |
|--------|-------------|--|------------------------|------------------------|-----|---|-----|----|-----|---|-----|---|-----|---|
| [31:7] | Reserved    | Reserved   |                        |                        |     |   |     |    |     |   |     |   |     |   |
| [6:4]  | CSR1        | <b>PWM Timer 1 Clock Source Divider Selection</b><br>Select clock source divider for PWM timer 1.  |                        |                        |     |   |     |    |     |   |     |   |     |   |
|        |             | <table><tr><th>CSR1</th><th>Input Clock Divided by</th></tr><tr><td>100</td><td>1</td></tr><tr><td>011</td><td>16</td></tr><tr><td>010</td><td>8</td></tr><tr><td>001</td><td>4</td></tr><tr><td>000</td><td>2</td></tr></table> | CSR1                   | Input Clock Divided by | 100 | 1 | 011 | 16 | 010 | 8 | 001 | 4 | 000 | 2 |
|        |             | CSR1   | Input Clock Divided by |                        |     |   |     |    |     |   |     |   |     |   |
|        |             | 100  | 1                      |                        |     |   |     |    |     |   |     |   |     |   |
|        |             | 011  | 16                     |                        |     |   |     |    |     |   |     |   |     |   |
|        |             | 010  | 8                      |                        |     |   |     |    |     |   |     |   |     |   |
|        |             | 001  | 4                      |                        |     |   |     |    |     |   |     |   |     |   |
| 000    | 2           |  |                        |                        |     |   |     |    |     |   |     |   |     |   |
| [3]    | Reserved    | Reserved   |                        |                        |     |   |     |    |     |   |     |   |     |   |
| [2:0]  | CSR0        | <b>PWM Timer 0 Clock Source Divider Selection</b><br>Select clock source divider for PWM timer 0.<br>(Table is the same as CSR1)   |                        |                        |     |   |     |    |     |   |     |   |     |   |
|        |             |  |                        |                        |     |   |     |    |     |   |     |   |     |   |

**PWM Control Register (PCR)**

| Register | Offset       | R/W | Description                | Reset Value |
|----------|--------------|-----|----------------------------|-------------|
| PCR      | BPWM_BA+0x08 | R/W | Basic PWM Control Register | 0x0000_0000 |

| 31       | 30        | 29       | 28     | 27     | 26     | 25      | 24    |
|----------|-----------|----------|--------|--------|--------|---------|-------|
| Reserved | PWM01TYPE | Reserved |        |        |        |         |       |
| 23       | 22        | 21       | 20     | 19     | 18     | 17      | 16    |
| Reserved |           |          |        |        |        |         |       |
| 15       | 14        | 13       | 12     | 11     | 10     | 9       | 8     |
| Reserved |           |          |        | CH1MOD | CH1INV | CH1PINV | CH1EN |
| 7        | 6         | 5        | 4      | 3      | 2      | 1       | 0     |
| Reserved |           |          | DZEN01 | CH0MOD | CH0INV | CH0PINV | CH0EN |

| Bits    | Description |   |
|---------|-------------|---|
| [31]    | Reserved    | Reserved  |
| [30]    | PWM01TYPE   | <b>PWM01 Aligned Type Selection Bit</b><br>0 = Edge-aligned type.<br>1 = Center-aligned type.   |
| [29:12] | Reserved    | Reserved  |
| [11]    | CH1MOD      | <b>PWM-Timer 1 Auto-reload/One-Shot Mode</b><br>1 = Auto-reload mode<br>0 = One-shot mode<br><b>Note:</b> If there is a transition at this bit, it will cause CNR1 and CMR1 be cleared. |
| [10]    | CH1INV      | <b>PWM-Timer 1 Output Inverter Enable</b><br>1 = Inverter Enabled<br>0 = Inverter Disabled  |
| [9]     | CH1PINV     | <b>PWM-Timer 1 Output Polar Inverse Enable</b><br>1 = PWM1 output polar inverse Enabled<br>0 = PWM1 output polar inverse Disabled   |
| [8]     | CH1EN       | <b>PWM-Timer 1 Enable</b><br>1 = Corresponding PWM-Timer Start Running<br>0 = Corresponding PWM-Timer Stopped   |
| [7:5]   | Reserved    | Reserved  |
| [4]     | DZEN01      | <b>Dead-zone 0 Generator Enable</b><br>1 = Enabled<br>0 = Disabled<br><b>Note:</b> When Dead-zone generator is enabled, the pair of PWM0 and PWM1 becomes a complementary pair.         |

|     |         |   |
|-----|---------|---|
| [3] | CH0MOD  | <b>PWM-Timer 0 Auto-reload/One-Shot Mode</b><br>1 = Auto-reload mode<br>0 = One-shot mode<br><b>Note:</b> If there is a transition at this bit, it will cause CNR0 and CMR0 be cleared. |
| [2] | CH0INV  | <b>PWM-Timer 0 Output Inverter Enable</b><br>1 = Inverter Enabled<br>0 = Inverter Disabled  |
| [1] | CH0PINV | <b>PWM-Timer 0 Output Polar Inverse Enable</b><br>1 = PWM0 output polar inverse Enabled<br>0 = PWM0 output polar inverse Disabled   |
| [0] | CH0EN   | <b>PWM-Timer 0 Enable</b><br>1 = The corresponding PWM-Timer starts running<br>0 = The corresponding PWM-Timer stops running  |

### PWM Counter Register 1-0 (CNR1-0)

| Register    | Offset       | R/W | Description                  | Reset Value |
|-------------|--------------|-----|------------------------------|-------------|
| <b>CNR0</b> | BPWM_BA+0x0C | R/W | Basic PWM Counter Register 0 | 0x0000_0000 |
| <b>CNR1</b> | BPWM_BA+0x18 | R/W | Basic PWM Counter Register 1 | 0x0000_0000 |

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved    |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved    |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CNRx [15:8] |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CNRx [7:0]  |    |    |    |    |    |    |    |

| Bits    | Description     |   |
|---------|-----------------|---|
| [31:16] | <b>Reserved</b> | Reserved  |
| [15:0]  | <b>CNRx</b>     | <p><b>PWM Timer Loaded Value</b></p> <p>CNR determines the PWM period.</p> <p>PWM frequency = <math>\text{BPWM\_CLK} / [(\text{prescale} + 1) * (\text{clock divider}) * (\text{CNR} + 1)]</math>.</p> <p>For Edge-aligned type:</p> <ul style="list-style-type: none"> <li>• Duty ratio = <math>(\text{CMR} + 1) / (\text{CNR} + 1)</math>.</li> <li>• <math>\text{CMR} \geq \text{CNR}</math>: PWM output is always high.</li> <li>• <math>\text{CMR} &lt; \text{CNR}</math>: PWM low width = <math>(\text{CNR} - \text{CMR})</math> unit; PWM high width = <math>(\text{CMR} + 1)</math> unit.</li> <li>• <math>\text{CMR} = 0</math>: PWM low width = <math>(\text{CNR})</math> unit; PWM high width = 1 unit</li> </ul> <p>For Center-aligned type:</p> <ul style="list-style-type: none"> <li>• Duty ratio = <math>[(2 \times \text{CMR}) + 1] / [2 \times (\text{CNR} + 1)]</math>.</li> <li>• <math>\text{CMR} &gt; \text{CNR}</math>: PWM output is always high.</li> <li>• <math>\text{CMR} \leq \text{CNR}</math>: PWM low width = <math>2 \times (\text{CNR} - \text{CMR}) + 1</math> unit; PWM high width = <math>(2 \times \text{CMR}) + 1</math> unit.</li> <li>• <math>\text{CMR} = 0</math>: PWM low width = <math>2 \times \text{CNR} + 1</math> unit; PWM high width = 1 unit</li> </ul> <p>(Unit = one PWM clock cycle)</p> <p><b>Note:</b> Any write to CNR will take effect in next PWM cycle.</p> <p><b>Note:</b> When PWM operating at Center-aligned type, CNR value should be set between 0x0000 to 0xFFFE. If CNR equal to 0xFFFF, the PWM will work unpredictable.</p> <p><b>Note:</b> When CNR value is set to 0, PWM output is always high.</p> |



**PWM Comparator Register 1-0 (CMR1-0)**

| Register    | Offset       | R/W | Description                     | Reset Value |
|-------------|--------------|-----|---------------------------------|-------------|
| <b>CMR0</b> | BPWM_BA+0x10 | R/W | Basic PWM Comparator Register 0 | 0x0000_0000 |
| <b>CMR1</b> | BPWM_BA+0x1C | R/W | Basic PWM Comparator Register 1 | 0x0000_0000 |

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved    |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved    |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CMRx [15:8] |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CMRx [7:0]  |    |    |    |    |    |    |    |

| Bits    | Description     |  |
|---------|-----------------|--|
| [31:16] | <b>Reserved</b> | Reserved   |
| [15:0]  | <b>CMRx</b>     | <p><b>PWM Comparator Register</b></p> <p>CMR determines the PWM duty.</p> <p>PWM frequency = <math>BPWM\_CLK / [(prescale+1) * (clock\ divider) * (CNR+1)]</math></p> <p>For Edge-aligned type:</p> <ul style="list-style-type: none"> <li>Duty ratio = <math>(CMR+1)/(CNR+1)</math>.</li> <li>CMR <math>\geq</math> CNR: PWM output is always high.</li> <li>CMR &lt; CNR: PWM low width = (CNR-CMR) unit; PWM high width = (CMR+1) unit.</li> <li>CMR = 0: PWM low width = (CNR) unit; PWM high width = 1 unit</li> </ul> <p>For Center-aligned type:</p> <ul style="list-style-type: none"> <li>Duty ratio = <math>[(2 \times CMR) + 1] / [2 \times (CNR+1)]</math>.</li> <li>CMR &gt; CNR: PWM output is always high.</li> <li>CMR <math>\leq</math> CNR: PWM low width = <math>2 \times (CNR-CMR) + 1</math> unit; PWM high width = <math>(2 \times CMR) + 1</math> unit.</li> <li>CMR = 0: PWM low width = <math>2 \times CNR + 1</math> unit; PWM high width = 1 unit</li> </ul> <p>(Unit = one PWM clock cycle)</p> <p><b>Note:</b> Any write to CNR will take effect in next PWM cycle.</p> |

### PWM Data Register 1-0 (PDR 1-0)

| Register    | Offset       | R/W | Description               | Reset Value |
|-------------|--------------|-----|---------------------------|-------------|
| <b>PDR0</b> | BPWM_BA+0x14 | R   | Basic PWM Data Register 0 | 0x0000_0000 |
| <b>PDR1</b> | BPWM_BA+0x20 | R   | Basic PWM Data Register 1 | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved   |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved   |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PDRx[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PDRx[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | Reserved    | Reserved  |
| [15:0]  | PDRx        | <b>PWM Data Register</b><br>User can monitor PDR to know the current value in 16-bit counter. |

### PWM Interrupt Enable Register (PIER)

| Register | Offset       | R/W | Description                         | Reset Value |
|----------|--------------|-----|-------------------------------------|-------------|
| PIER     | BPWM_BA+0x40 | R/W | Basic PWM Interrupt Enable Register | 0x0000_0000 |

|          |    |    |    |    |    |          |          |
|----------|----|----|----|----|----|----------|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25       | 24       |
| Reserved |    |    |    |    |    |          |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17       | 16       |
| Reserved |    |    |    |    |    |          | INTTYPE  |
| 15       | 14 | 13 | 12 | 11 | 10 | 9        | 8        |
| Reserved |    |    |    |    |    | BPWMDIE1 | BPWMDIE0 |
| 7        | 6  | 5  | 4  | 3  | 2  | 1        | 0        |
| Reserved |    |    |    |    |    | BPWMPIE1 | BPWMPIE0 |

| Bits    | Description |  |
|---------|-------------|--|
| [31:17] | Reserved    | Reserved   |
| [16]    | INTTYPE     | <b>BPWM Interrupt Period Type Selection Bit</b><br>1 = BPWMIFn will be set if BPWM counter matches CNRn register.<br>0 = BPWMIFn will be set if BPWM counter underflow.<br><b>Note:</b> This bit is effective when BPWM in Center-aligned type only. |
| [15:10] | Reserved    | Reserved   |
| [9]     | BPWMDIE1    | <b>BPWM channel 1 Duty Interrupt Enable</b><br>1 = Enabled<br>0 = Disabled   |
| [8]     | BPWMDIE0    | <b>BPWM channel 0 Duty Interrupt Enable</b><br>1 = Enabled<br>0 = Disabled   |
| [7:2]   | Reserved    | Reserved   |
| [1]     | BPWMIE1     | <b>BPWM channel 1 Period Interrupt Enable</b><br>1 = Enabled<br>0 = Disabled   |
| [0]     | BPWMIE0     | <b>BPWM channel 0 Period Interrupt Enable</b><br>1 = Enabled<br>0 = Disabled   |

**PWM Interrupt Indication Register (PIIR)**

| Register | Offset       | R/W | Description                             | Reset Value |
|----------|--------------|-----|---|-------------|
| PIIR     | BPWM_BA+0x44 | R/W | Basic PWM Interrupt Indication Register | 0x0000_0000 |

|          |    |    |    |    |    |         |         |
|----------|----|----|----|----|----|---------|---------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25      | 24      |
| Reserved |    |    |    |    |    |         |         |
| 23       | 22 | 21 | 20 | 19 | 18 | 17      | 16      |
| Reserved |    |    |    |    |    |         |         |
| 15       | 14 | 13 | 12 | 11 | 10 | 9       | 8       |
| Reserved |    |    |    |    |    | PWMDIF1 | PWMDIF0 |
| 7        | 6  | 5  | 4  | 3  | 2  | 1       | 0       |
| Reserved |    |    |    |    |    | PWMIF1  | PWMIF0  |

| Bits    | Description   |
|---------|---|
| [31:10] | Reserved  |
| [9]     | <b>BPWMDIF1</b><br><b>BPWM Channel 1 Duty Interrupt Flag</b><br>Flag is set by hardware when channel 1 BPWM counter down count and reaches CMR1, software can clear this bit by writing a one to it.<br><b>Note:</b> If CMR equal to CNR, this flag is not working in Edge-aligned type selection |
| [8]     | <b>BPWMDIF0</b><br><b>BPWM Channel 0 Duty Interrupt Flag</b><br>Flag is set by hardware when channel 0 BPWM counter down count and reaches CMR0, software can clear this bit by writing a one to it.<br><b>Note:</b> If CMR equal to CNR, this flag is not working in Edge-aligned type selection |
| [7:2]   | Reserved  |
| [1]     | <b>BPWMIF1</b><br><b>BPWM channel 1 Period Interrupt Status</b><br>This bit is set by hardware when BPWM1 counter reaches the requirement of interrupt (depend on INTTYPE bit of PIER register), software can write 1 to clear this bit to 0.   |
| [0]     | <b>BPWMIF0</b><br><b>BPWM channel 0 Period Interrupt Status</b><br>This bit is set by hardware when BPWM0 counter reaches the requirement of interrupt (depend on INTTYPE bit of PIER register), software can write 1 to clear this bit to 0.   |

**Note:** User can clear each interrupt flag by writing 1 to corresponding bit in PIIR.

**Capture Control Register (CCR)**

| Register | Offset       | R/W | Description                        | Reset Value |
|----------|--------------|-----|------------------------------------|-------------|
| CCR      | BPWM_BA+0x50 | R/W | Basic PWM Capture Control Register | 0x0000_0000 |

|          |       |          |        |          |         |         |      |
|----------|-------|----------|--------|----------|---------|---------|------|
| 31       | 30    | 29       | 28     | 27       | 26      | 25      | 24   |
| Reserved |       |          |        |          |         |         |      |
| 23       | 22    | 21       | 20     | 19       | 18      | 17      | 16   |
| CFLR1    | CRLR1 | Reserved | CAPIF1 | CAPCH1EN | CFL_IE1 | CRL_IE1 | INV1 |
| 15       | 14    | 13       | 12     | 11       | 10      | 9       | 8    |
| Reserved |       |          |        |          |         |         |      |
| 7        | 6     | 5        | 4      | 3        | 2       | 1       | 0    |
| CFLR0    | CRLR0 | Reserved | CAPIF0 | CAPCH0EN | CFL_IE0 | CRL_IE0 | INV0 |

| Bits    | Description |   |
|---------|-------------|---|
| [31:24] | Reserved    | Reserved  |
| [23]    | CFLR1       | <b>CFLR1 Latched Indicator Bit</b><br>When BPWM input channel 1 has a falling transition, CFLR1 was latched with the value of BPWM down-counter and this bit is set by hardware.<br>Software can write 1 to clear this bit to 0   |
| [22]    | CRLR1       | <b>CRLR1 Latched Indicator Bit</b><br>When BPWM input channel 1 has a rising transition, CRLR1 was latched with the value of BPWM down-counter and this bit is set by hardware.<br>Software can write 1 to clear this bit to 0.   |
| [5]     | Reserved    | Reserved  |
| [20]    | CAPIF1      | <b>Channel 1 Capture Interrupt Indication Flag</b><br>If PWM channel 1 rising latch interrupt is enabled (CRL_IE1 = 1), a rising transition occurs at BPWM channel 1 will result in CAPIF1 to high; Similarly, a falling transition will cause CAPIF1 to be set high if BPWM channel 1 falling latch interrupt is enabled (CFL_IE1 = 1).<br>Write 1 to clear this bit to 0. |
| [19]    | CAPCH1EN    | <b>Channel 1 Capture Function Enable</b><br>1 = Capture function on BPWM channel 1 Enabled<br>0 = Capture function on BPWM channel 1 Disabled<br>When Enabled, Capture latched the BPWM-counter and saved to CRLR (Rising latch) and CFLR (Falling latch).<br>When Disabled, Capture does not update CRLR and CFLR, and disable PWM channel 1 Interrupt.                    |
| [18]    | CFL_IE1     | <b>Channel 1 Falling Latch Interrupt Enable</b><br>1 = Falling latch interrupt Enabled<br>0 = Falling latch interrupt Disabled<br>When Enabled, if Capture detects BPWM channel 1 has falling transition, Capture will issue an Interrupt.  |

|        |                 |   |
|--------|-----------------|---|
| [17]   | <b>CRL_IE1</b>  | <b>Channel 1 Rising Latch Interrupt Enable</b><br>1 = Rising latch interrupt Enabled<br>0 = Rising latch interrupt Disabled<br>When Enabled, if Capture detects BPWM channel 1 has rising transition, Capture will issue an Interrupt.  |
| [16]   | <b>INV1</b>     | <b>Channel 1 Inverter Enable</b><br>1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer<br>0 = Inverter Disabled   |
| [15:8] | <b>Reserved</b> | Reserved  |
| [7]    | <b>CFLRI0</b>   | <b>CFLR0 Latched Indicator Bit</b><br>When BPWM input channel 0 has a falling transition, CFLR0 was latched with the value of BPWM down-counter and this bit is set by hardware.<br>Software can write 1 to clear this bit to 0.  |
| [6]    | <b>CRLRI0</b>   | <b>CRLR0 Latched Indicator Bit</b><br>When BPWM input channel 0 has a rising transition, CRLR0 was latched with the value of BPWM down-counter and this bit is set by hardware.<br>Software can write 1 to clear this bit to 0.   |
| [5]    | <b>Reserved</b> | Reserved  |
| [4]    | <b>CAPIF0</b>   | <b>Channel 0 Capture Interrupt Indication Flag</b><br>If PWM channel 0 rising latch interrupt is enabled (CRL_IE0 = 1), a rising transition occurs at BPWM channel 0 will result in CAPIF0 to high; Similarly, a falling transition will cause CAPIF0 to be set high if BPWM channel 0 falling latch interrupt is enabled (CFL_IE0 = 1).<br>Write 1 to clear this bit to 0. |
| [3]    | <b>CAPCH0EN</b> | <b>Channel 0 Capture Function Enable</b><br>1 = Capture function on BPWM channel 0 Enabled<br>0 = Capture function on BPWM channel 0 Disabled<br>When Enabled, Capture latched the BPWM-counter value and saved to CRLR (Rising latch) and CFLR (Falling latch).<br>When Disabled, Capture does not update CRLR and CFLR, and disable BPWM channel 0 Interrupt.             |
| [2]    | <b>CFL_IE0</b>  | <b>Channel 0 Falling Latch Interrupt Enable</b><br>1 = Falling latch interrupt Enabled<br>0 = Falling latch interrupt Disabled<br>When Enabled, if Capture detects BPWM channel 0 has falling transition, Capture will issue an Interrupt.  |
| [1]    | <b>CRL_IE0</b>  | <b>Channel 0 Rising Latch Interrupt Enable</b><br>1 = Rising latch interrupt Enabled<br>0 = Rising latch interrupt Disabled<br>When Enabled, if Capture detects BPWM channel 0 has rising transition, Capture will issue an Interrupt.  |
| [0]    | <b>INV0</b>     | <b>Channel 0 Inverter Enable</b><br>1 = Inverter Enabled. Reverse the input signal from GPIO before fed to Capture timer<br>0 = Inverter Disabled   |

### Capture Rising Latch Register1-0 (CRLR1-0)

| Register     | Offset       | R/W | Description   | Reset Value |
|--------------|--------------|-----|---|-------------|
| <b>CRLR0</b> | BPWM_BA+0x58 | R   | Basic PWM Capture Rising Latch Register (Channel 0) | 0x0000_0000 |
| <b>CRLR1</b> | BPWM_BA+0x60 | R   | Basic PWM Capture Rising Latch Register (Channel 1) | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved     |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CRLRx [15:8] |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CRLRx [7:0]  |    |    |    |    |    |    |    |

| Bits    | Description     |  |
|---------|-----------------|--|
| [31:16] | <b>Reserved</b> | Reserved   |
| [15:0]  | <b>CRLRx</b>    | <b>Capture Rising Latch Register</b><br>Latch the BPWM counter when Channel 0/1 has rising transition. |

### Capture Falling Latch Register1-0 (CFLR1-0)

| Register     | Offset       | R/W | Description  | Reset Value |
|--------------|--------------|-----|--|-------------|
| <b>CFLR0</b> | BPWM_BA+0x5C | R   | Basic PWM Capture Falling Latch Register (Channel 0) | 0x0000_0000 |
| <b>CFLR1</b> | BPWM_BA+0x64 | R   | Basic PWM Capture Falling Latch Register (Channel 1) | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved     |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CFLRx [15:8] |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CFLRx [7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | Reserved    | Reserved   |
| [15:0]  | CFLRx       | <b>Capture Falling Latch Register</b><br>Latch the BPWM counter when Channel 0/1 has Falling transition. |



### Capture Input Enable Register (CAPENR)

| Register | Offset       | R/W | Description                             | Reset Value |
|----------|--------------|-----|---|-------------|
| CAPENR   | BPWM_BA+0x78 | R/W | Basic PWM Capture Input Enable Register | 0x0000_0000 |

|          |    |    |    |    |    |        |        |
|----------|----|----|----|----|----|--------|--------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25     | 24     |
| Reserved |    |    |    |    |    |        |        |
| 23       | 22 | 21 | 20 | 19 | 18 | 17     | 16     |
| Reserved |    |    |    |    |    |        |        |
| 15       | 14 | 13 | 12 | 11 | 10 | 9      | 8      |
| Reserved |    |    |    |    |    |        |        |
| 7        | 6  | 5  | 4  | 3  | 2  | 1      | 0      |
| Reserved |    |    |    |    |    | CINEN1 | CINEN0 |

| Bits   | Description |  |
|--------|-------------|--|
| [31:2] | Reserved    | Reserved   |
| [1]    | CINEN1      | <b>Channel 1 Capture Input Enable</b><br>1 = BPWM Channel 1 capture input path Enabled. The input of BPWM channel 1 capture function comes from correlative multifunction pin if GPIO multi-function is set as PWM21.<br>0 = BPWM Channel 1 capture input path Disabled. The input of BPWM channel 1 capture function is always regarded as 0. |
| [0]    | CINEN0      | <b>Channel 0 Capture Input Enable</b><br>1 = BPWM Channel 0 capture input path Enabled. The input of BPWM channel 0 capture function comes from correlative multifunction pin if GPIO multi-function is set as PWM20.<br>0 = BPWM Channel 0 capture input path Disabled. The input of BPWM channel 0 capture function is always regarded as 0. |

### PWM Output Enable Register (POE)

| Register | Offset       | R/W | Description             | Reset Value |
|----------|--------------|-----|-------------------------|-------------|
| POE      | BPWM_BA+0x7C | R/W | Basic PWM Output Enable | 0x0000_0000 |

|          |    |    |    |    |    |      |      |
|----------|----|----|----|----|----|------|------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25   | 24   |
| Reserved |    |    |    |    |    |      |      |
| 23       | 22 | 21 | 20 | 19 | 18 | 17   | 16   |
| Reserved |    |    |    |    |    |      |      |
| 15       | 14 | 13 | 12 | 11 | 10 | 9    | 8    |
| Reserved |    |    |    |    |    |      |      |
| 7        | 6  | 5  | 4  | 3  | 2  | 1    | 0    |
| Reserved |    |    |    |    |    | POE1 | POE0 |

| Bits   | Description |  |
|--------|-------------|--|
| [31:2] | Reserved    | Reserved   |
| [1]    | POE1        | <b>Channel 1 Output Enable Register</b><br>1 = BPWM channel 1 output to pin Enabled<br>0 = BPWM channel 1 output to pin Disabled<br><b>Note:</b> The corresponding GPIO pin must also be switched to BPWM function |
| [0]    | POE0        | <b>Channel 0 Output Enable Register</b><br>1 = BPWM channel 0 output to pin Enabled<br>0 = BPWM channel 0 output to pin Disabled<br><b>Note:</b> The corresponding GPIO pin must also be switched to BPWM function |

## 14 ENHANCED PWM GENERATOR FOR MOTOR DRIVE (EPWM)

### 14.1 Overview

This device has two built-in PWM units with the same architecture whose function is specially designed for driving motor control applications. Using the PWM, input capture module and QEI controller with proper control flow by software can easily drive the 3-phase Brushless DC motor, 3-phase AC induction motor, and DC motor.

## 14.2 Features

Each unit supports the features listed below:

- **Three** independent 16-bit PWM duty control units with maximum 6 port pins:
  - **Three independent PWM output:**  
PWM00, PWM02 and PWM04 for Unit 0  
PWM10, PWM12 and PWM14 for Unit 1
  - **Three complementary PWM pairs**, with each pin in a pair mutually complement to each other and capable of programmable dead-time insertion:  
(PWMx0,PWMx1), (PWMx2,PWMx3) and (PWMx4,PWMx5) where x=0~1.
  - 3 synchronous PWM pairs, with each pin in a pair in-phase:  
(PWM0,PWM1), (PWM2,PWM3) and (PWM4,PWM5)
- Group control bit:  
PWM2 and PWM4 are synchronized with PWM0
- Supports Edge-aligned mode and Center-aligned mode
- Programmable dead-time insertion between complementary paired PWMs
- Each pin of from PWM0 to PWM5 has independent polarity setting control
- Mask output control for Electrically Commutated Motor operation
- Tri-state output at reset and brake state
- Hardware brake protections.
- Two Interrupt Sources:
  - Interrupt is synchronously requested at PWM frequency when up/down counter comparison matched (Edge- and Center-aligned modes) or underflow (Center-aligned mode).
  - Interrupt is requested when external brake pins asserted
- The PWM signals before polarity control stage are defined in view of positive logic. The PWM ports is active high or active low are controlled by polarity control register.
- High Source/Sink current

After CPU reset, the internal output of the each PWM channels depends on the polarity setting. The interval between successive outputs is controlled by a 16-bit up/down counter which uses a software selectable clock source with configurable internal clock pre-scalar as its input. The PWM counter clock has the frequency as the clock source  $F_{PWM} = EPWMx\_CLK/Pre\text{-}scalar$ ; Here the EPWMx\_CLK synchronized with CPU clock HCLK.

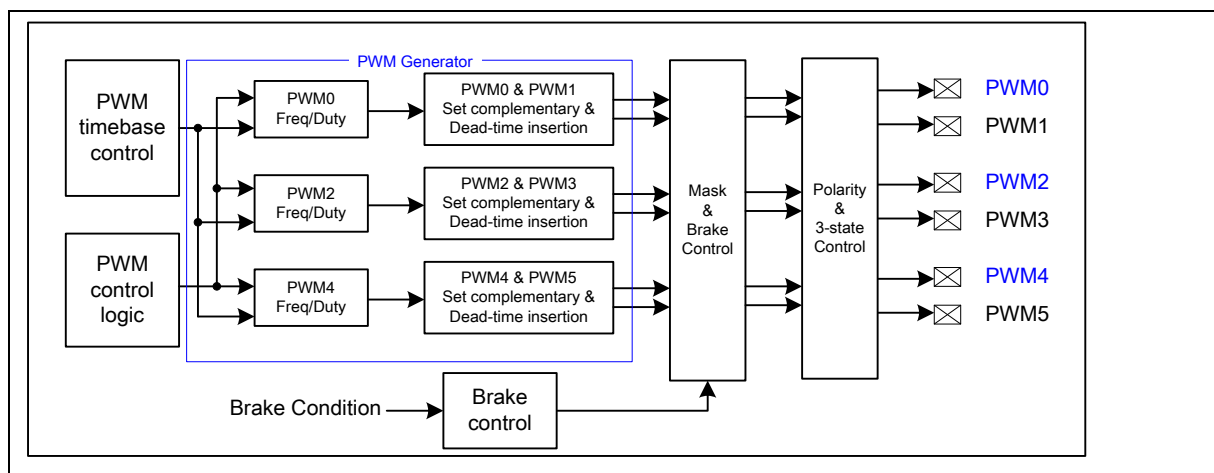


Figure 14-1 PWM Block Diagram

### 14.3 PWM Operation

The following diagrams show the PWM clock source control and PWM time-base generator.

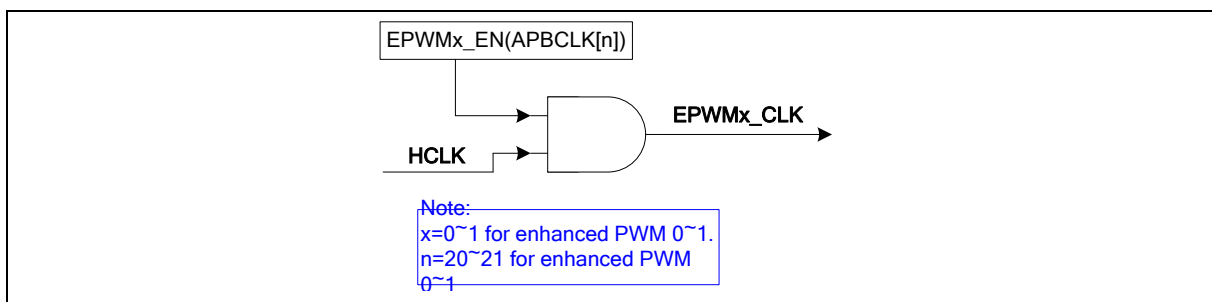


Figure 14-2 PWM Clock Source Control

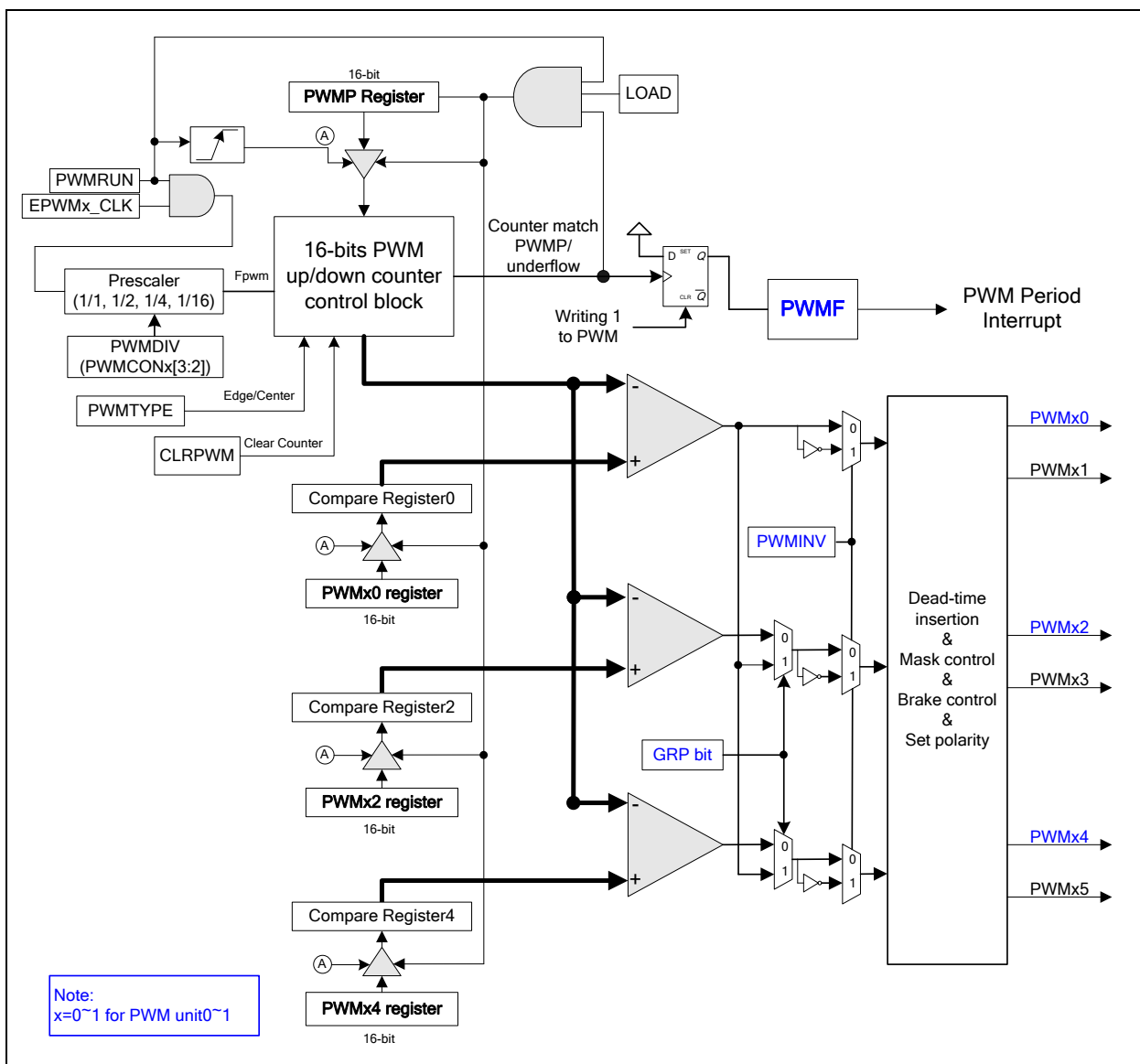


Figure 14-3 PWM Time-base Generator

The overall functioning of the PWM module is controlled by the contents of the PWMCONx0, PWMB

and PWMCONx2 registers. The operation of most of the control bits is straightforward. **PWMRUN** (PWMCONx0[7]) allows the PWM to be either in the run or idle state. The transfer of the data from the PWMP register to 16-bit PWM counter (PWMCNT) will occur on the rising edge of PWMRUN or during PWMRUN with Load (PWMCONx0[6]) and PWM counter match/underflow occurs. The transfer of the data from the PWMn registers to the compare registers is controlled by the Load bit (PWMCONx0[6]) (with the condition that PWMRUN = 1 and match/underflow occurs). [x=0-1].

**Note:**

1. A compare value greater than the counter reloaded value resulted in the PWM output being permanently high. In addition there are two special cases. If compare register is set to 0x0000, the PWMn output will stay at low, and if compare register is set to 0xFFFF, the PWMn output will stuck at high until there is a change in the compare register. [n = 0-3].
2. During ICP, ISP or ICE mode, PWM pins will be tri-stated. PWM operation will stop and module reset. When exit from ICP, ISP or ICE mode, the PWM pins will follow the control register settings.
3. In ICE mode, the condition which causes CPU stops or pauses running will force the PWMn output pins in tri-stated, when CPU runs the PWMn pins will follow the control register settings.

When a PWM period value is written to the PWMP register by software, the value is saved in the holding register first, and then the value of the holding register will be reloaded to the actual PWM period when the following conditions are met: Load = 1, PWMRUN = 1 and PWM match/underflow. The width of each PWM output pulse is determined by the value in the appropriate compare register. Each PWM registers of **PWMPx**, **PWMx0**, **PWMx2** and **PWMx4**, in the format of 16-bit width, decides the PWM period and each channel's duty cycle. If the PWMINV (Inverse PWM Comparator Output) is set to high the PWM comparator output signals will be inversed, therefore the PWM Duty (in percentage) is changed to (1-Duty) before PWMINV is set to high and PWM Duty registers, PWM0/2/4 represent Duty-off time.

Note that the duty registers PWMx0/2/4 and the period registers PWMPx are double-buffered registers used to set the duty cycle and counting period for the PWM time base respectively. For the 1<sup>st</sup> buffer it is accessible by user while the 2<sup>nd</sup> buffer holds the actual compare value used in the present period. Load bit must be set to 1 to enable the value to be loaded in to the 2<sup>nd</sup> buffer register when counter underflow/match.

### 14.3.1 PWM Operation Mode

This device supports two operation modes: Edge-aligned and Center-aligned mode.

The following equations show the formula for period and duty for each PWM operation mode:

**Edge-aligned:**

Period = (PWMP + 1) \* EPWMx\_CLK period/pre-scalar

Duty = (duty + 1) \* EPWMx\_CLK period /pre-scalar

**Center-aligned:**

Period = (PWMP\* 2) \* EPWMx\_CLK period/pre-scalar

Duty = (duty\*2 + 1) \* EPWMx\_CLK period/pre-scalar

**Note:** "duty" refers to PWM0/2/4/6 register value.

**Edge aligned PWM (Up Counter)**

In Edge-aligned PWM Output mode, the 16-bits PWM counter will starts counting from 0 to match with the value of the duty cycle PWM0 (old), when this happen it will toggle the PWM0 generator output to low. The counter will continue counting to match with the value of the period register PWMP (old), at this moment, it toggles the PWM0 generator output to high and new PWM0 (new) and PWMP(new) are updated with Load=1 and request the PWM interrupt if PWM interrupt is enabled(EIE1.6=1).

Figure 14–4, Figure 14–5 and Figure 14–6 depict the Edge-aligned PWM timing and operation flow.

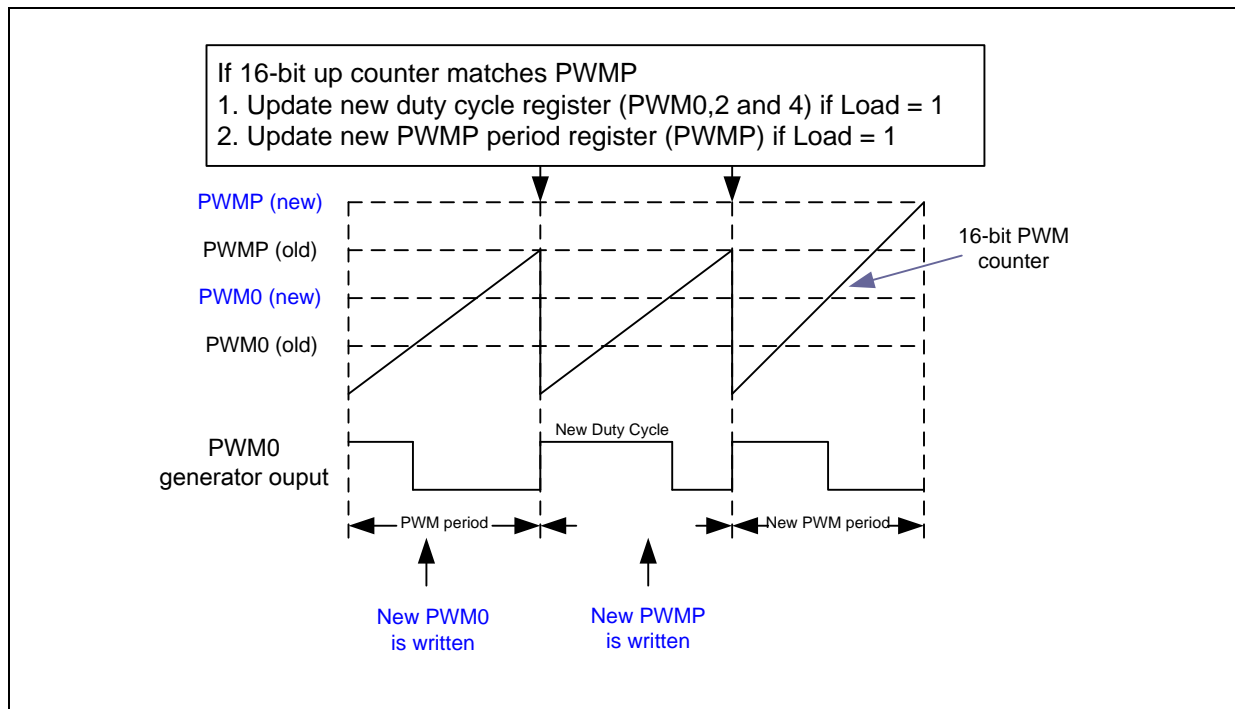


Figure 14-4 Edge-aligned PWM

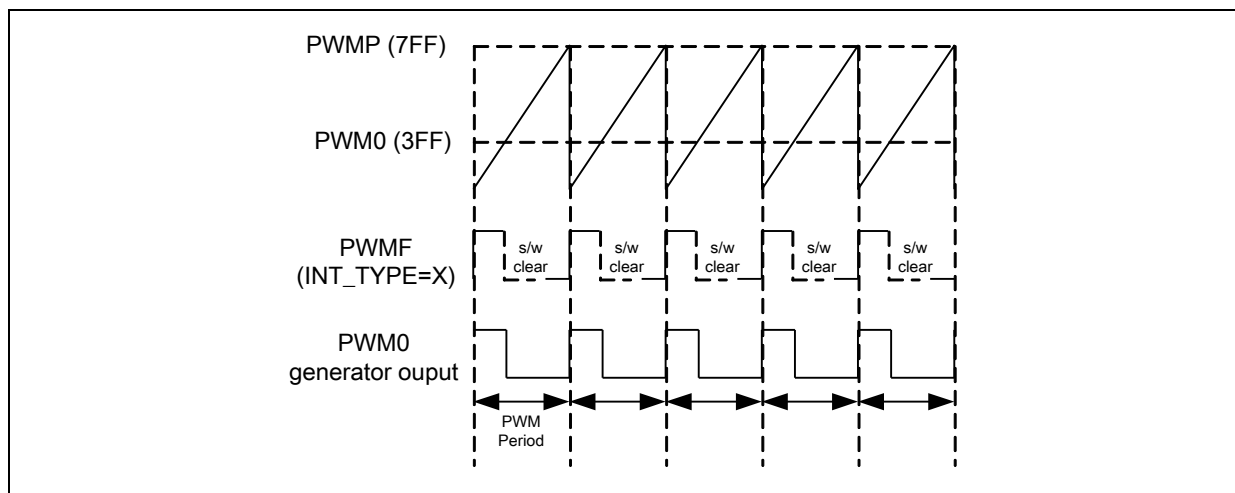


Figure 14-5 PWM0 Edge aligned Waveform Output

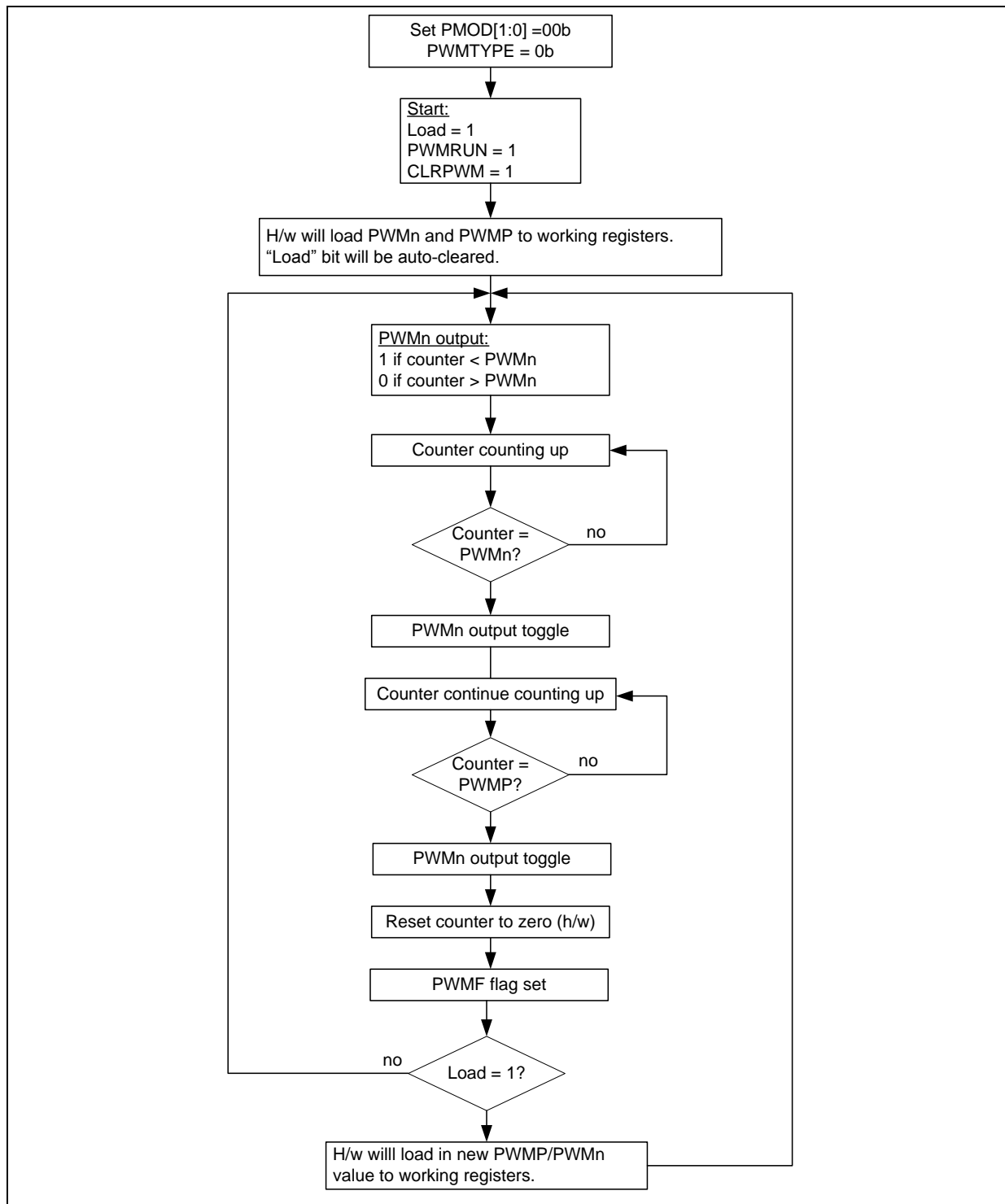


Figure 14–6 Edge-aligned Flow Diagram



### Center-Aligned PWM (up/down counter)

The Center-aligned PWM signals are produced by the module when the PWM time base is configured in Up/Down Counting mode. The PWM counter will start counting-up from 0 to match the value of PWM0 (old), and this will cause the toggling of the PWM0 generator output to low. The counter will continue counting to match with the PWMP (old). Upon reaching this state the counter is configured automatically to down counting, when PWM counter matches the PWM0 (old) value again the PWM0 generator output toggles to high. Once the PWM counter underflows it will update the PWM period register PWMP(new) and duty cycle register PWM0(new) with Load = 1.

In Center-aligned mode, the PWM interrupt is requested at down-counter underflow if INT\_TYPE (PWMCONx0[8]) = 0, i.e. at start (end) of each PWM cycle or at up-counter matching with PWMP if INT\_TYPE (PWMCONx0[0]) = 1, i.e. at center point of PWM cycle.

Figure 14–7, Figure 14–8 and Figure 14–9 depict the Center-aligned PWM timing and operation flow.

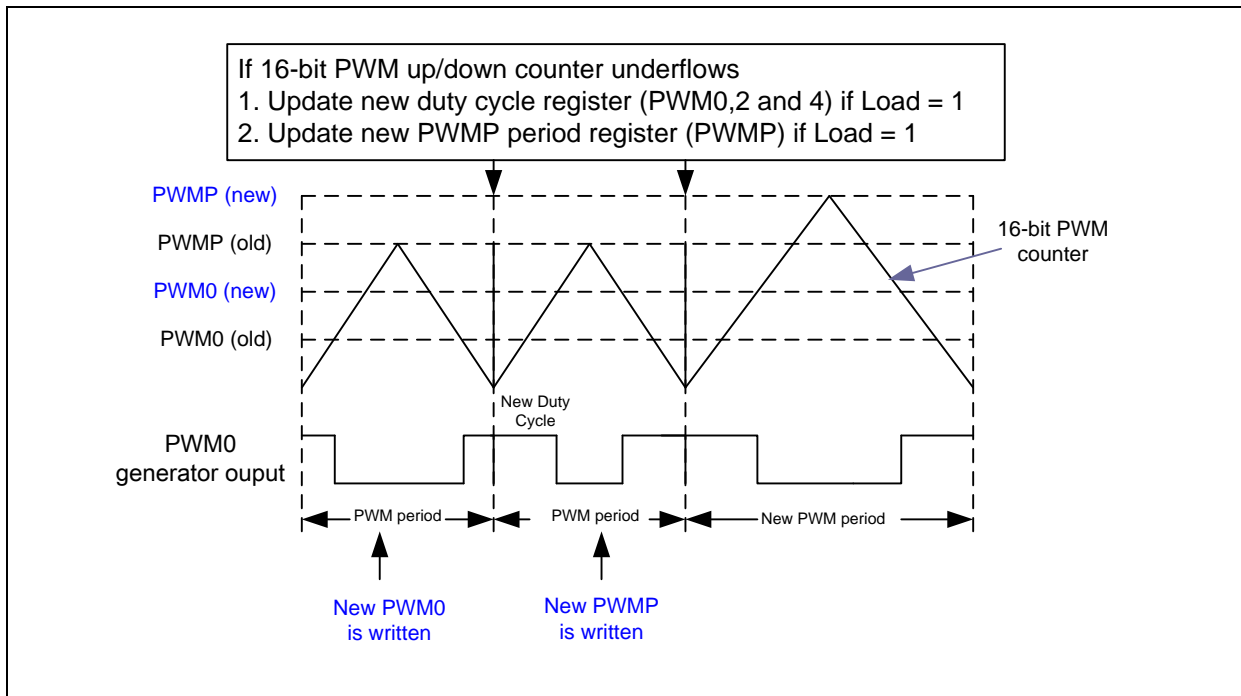


Figure 14–7 Center-aligned Mode

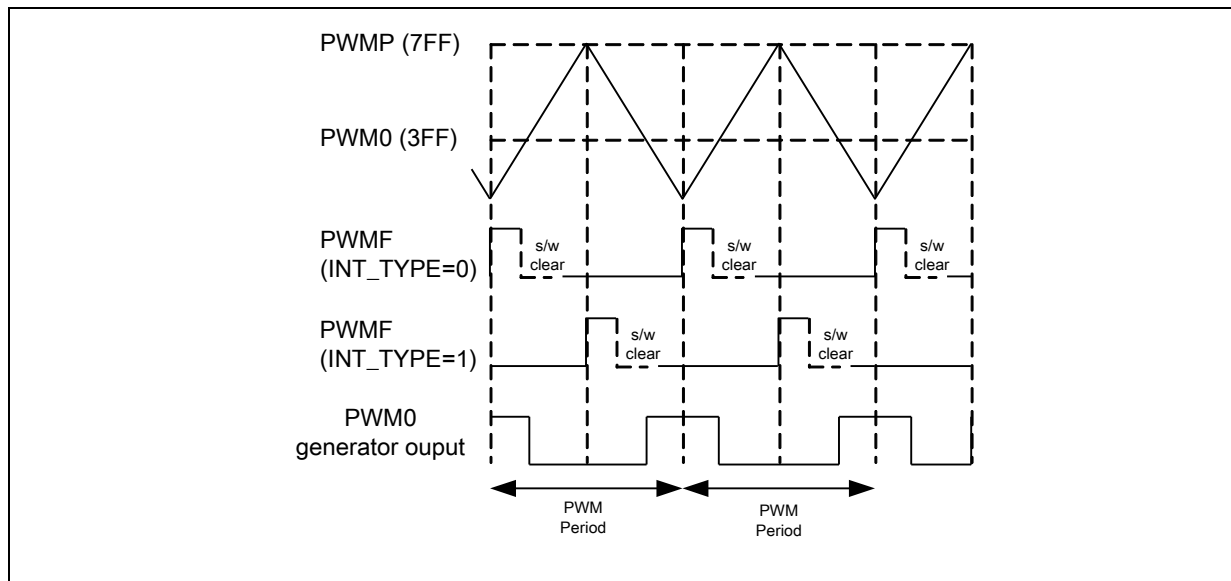


Figure 14–8 Example PWM0 Center-aligned Waveform Output

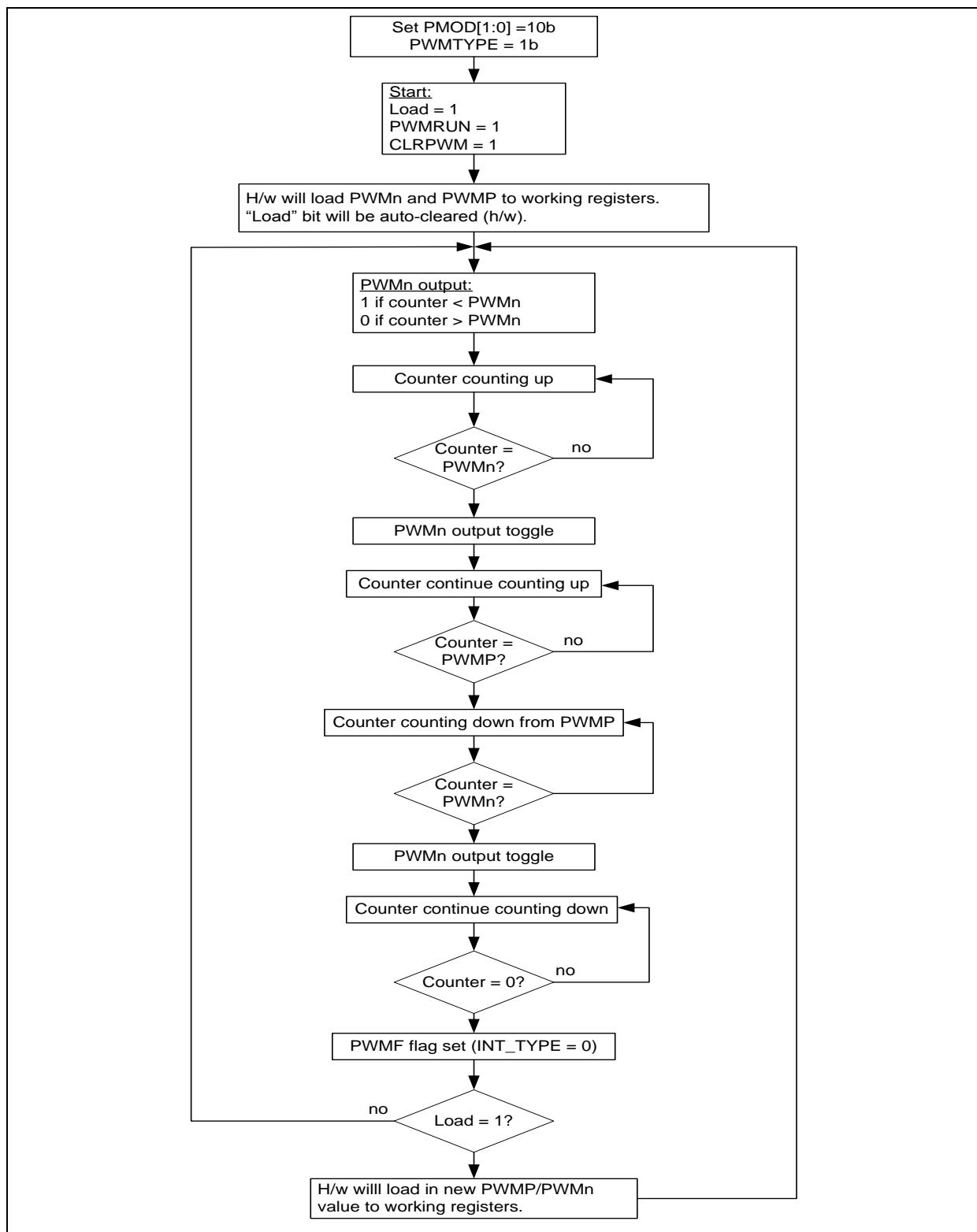


Figure 14–9 Center-aligned Flow Diagram (INT\_TYPE = 0)

## 14.4 PWM Brake

This device supports two brake detectors, BK0 and BK1, and each of them has 4 brake signals, one external brake pin (BKPx0 for BK0 and BKPx1 for BK1), and three analog comparator outputs. Both external brake pins have each 4-degree digital filter that is user controllable through BKxFILT.1-0 bits (x=0 and 1 for BK0 and BK1). The Brake function is controlled by the contents of the SFR PWMCON register.

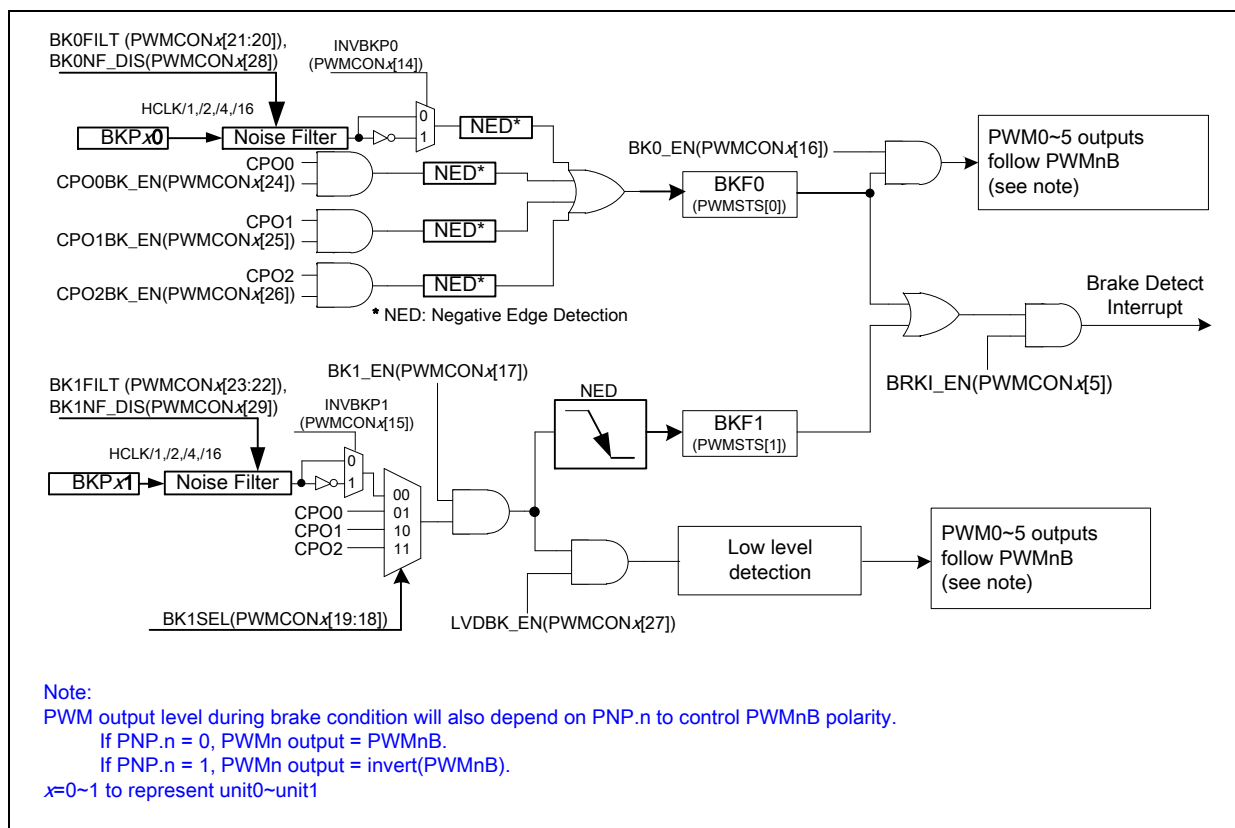


Figure 14–10 PWM Brake Function

The following table summarizes the effect of each brake pins. Figure 14–11 and Figure 14–12 illustrate the brake signals vs. PWM operation.

| Brake Source   | Brake Trigger Condition                  | Actions  |
|--|--|--|
| BKP0<br>(I/O pin);<br>CPO0<br>(Comparator0 output);<br>CPO1<br>(Comparator1 output);<br>CPO2<br>(Comparator2 output) | A falling edge at brake 0 edge detector. | <ul style="list-style-type: none"> <li>BKF0 and BCLK0 flags set by the falling edge at brake 0 edge detector; but cleared by S/W.</li> <li>PWM0~5 outputs follow PWMnB bits. PNP bits are also able to control the polarity of PWMnB. If PNP.n=0, PWMn pin output = PWMnB. If PNP.n=1, PWMn pin output = invert(PWMnB).</li> <li>PWM0~5 will keep in PWMnB state before BCLK0 flag is cleared by S/W.</li> <li>PWMRUN bit remain asserted to keep PWM generators running.</li> <li>If the BCLK0 flag is cleared, the brake state will be released on next PWM cycle/period.</li> </ul> |

|  |   |   |
|--|---|---|
|  |   | <ul style="list-style-type: none"> <li>If PWMRUN is cleared to 0 before the BCLK0 flag is cleared, PWM0~5 will remain in PWMnB state.</li> </ul>  |
| BKP1<br>(I/O pin);<br>CPO0<br>(Comparator0 output);<br>CPO1<br>(Comparator1 output);<br>CPO2<br>(Comparator2 output) | A falling edge at brake 1 edge detector;<br><br>Low level state | <ol style="list-style-type: none"> <li>BKF1 flag is set by the falling edge at brake 1 edge detector; but cleared by S/W.</li> <li>If LVDBK_EN=1 and Brake 1 signal detected as low level, PWM0~5 pin outputs follow PWMnB bits. Otherwise, PWM0~5 will continue follow PWM generators' output. In both situations, PNP bits are also able to control port polarity.</li> <li>PWMRUN bit remain asserted to keep PWM generators running.</li> <li>PWM0~5 resume if BKP1 pin state returns to high state. PWM0~5 will resume on start of next PWM cycle/period.</li> </ol> |

**Note:** The brake enable bits, BK0\_EN and BK1\_EN, must be set in order for the above to be effective.

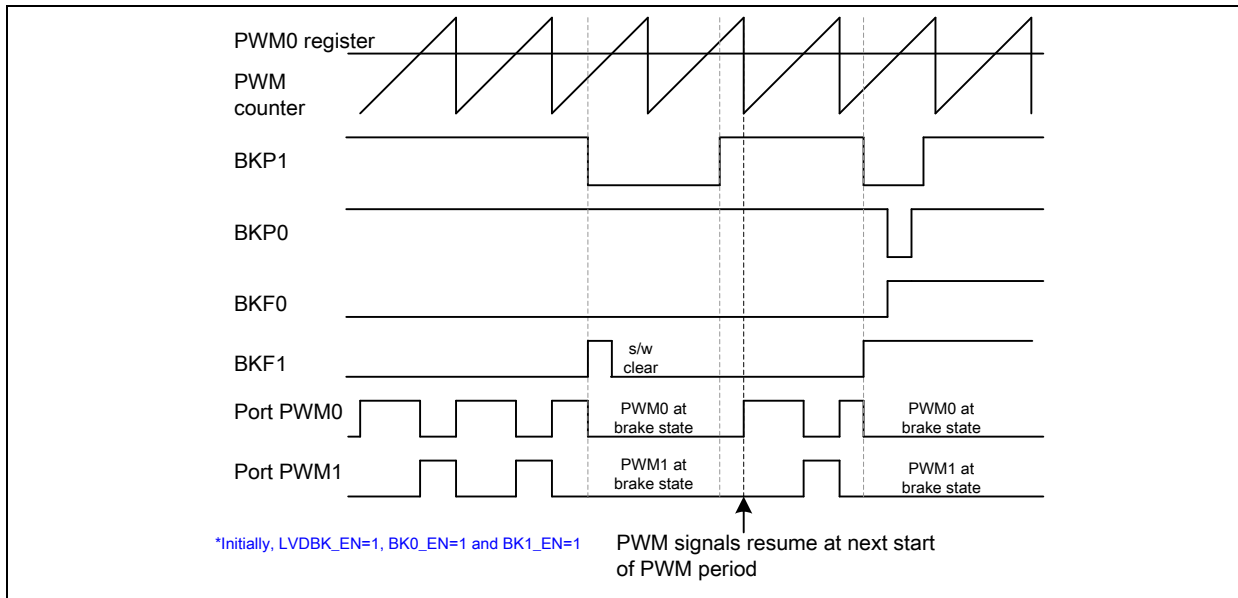


Figure 14–11 PWM Brake Condition (Edge-aligned Mode)

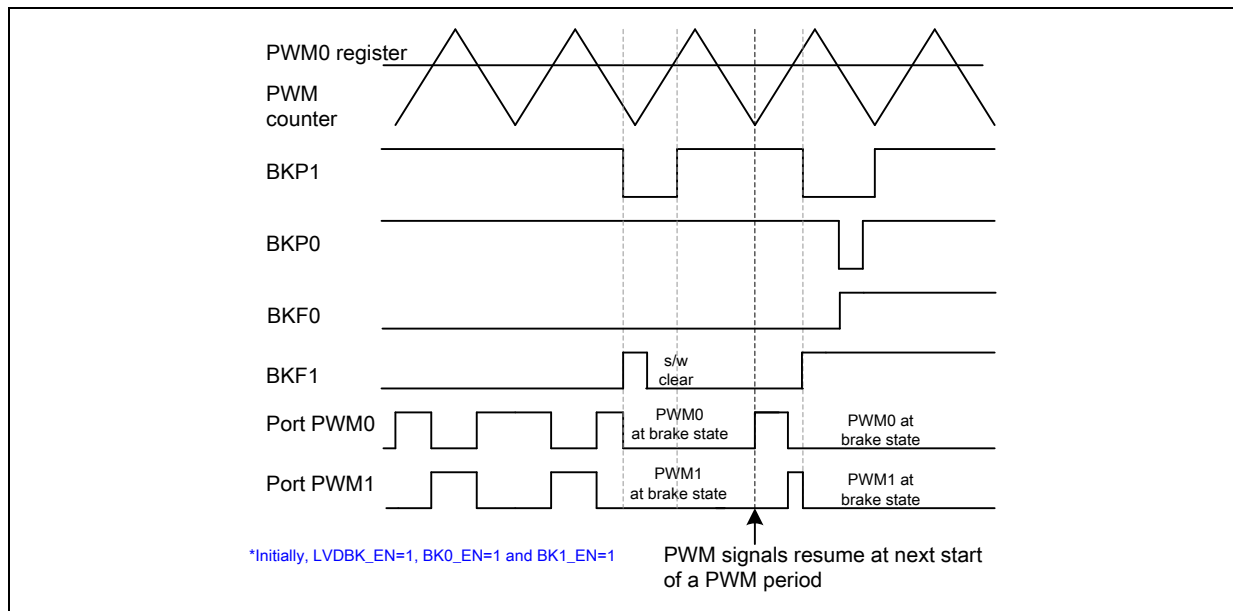


Figure 14–12 PWM Brake Condition (Centre-aligned Mode)

Since both brake conditions being asserted will automatically cause BKF<sub>n</sub> flag to be set, the user program can poll these brake flag bits or enable PWM's brake interrupt to determine which condition will cause a brake to occur.

## 14.5 PWM Port Output Driving Control

There are two enhanced PWM units and each unit has six output pins in this device. The PWM port outputs are P0.0~P0.5 and P1.0~P1.5 for unit 0 and unit 1, respectively.

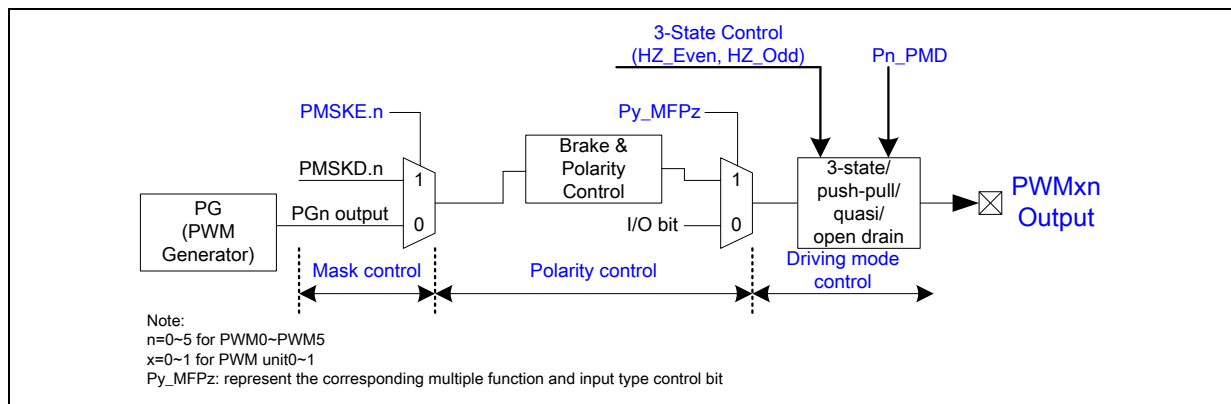


Figure 14-13 PWM Output Driving Control

The driving type of PWM output ports can be initialized as Tri-state type or other types depending on the Pn\_PMD register setting after any reset. As shown in the figure above, the PWM output structures are controllable through config-bit, register bits PWMPOEN[HZ\_Even0/1, HZ\_Odd0/1] and Pn\_PMD mode registers.

| HZ_Even/HZ_Odd<br>(in PWMPOEN Register) | Even/Odd PWM Outputs Drive Mode |
|---|---------------------------------|
| 0                                       | Depending on Pn_PMD             |
| 1                                       | Driving mode = Hi-Z.            |

**Note:** Register bits for HZ\_Even and HZ\_Odd are latched from config0 during all reset.

## 14.6 PWM Modes

This powerful PWM unit supports Independent mode which may be applied to DC and BLDC motor system, Complementary mode with dead-time insertion which may be used in the application of AC induction motor and synchronous motor, Synchronous mode that makes both pins of each pair are in phase. Besides, the Group mode, forces the PWM0, PWM2 and PWM4 synchronous with PWM0 generator, may simplify updating duty control in DC and BLDC motor applications.

### 14.6.1 Independent Mode

Independent mode is enabled when PMOD.1-0 = 00b.

By default, the PWM is operating in Independent mode with three PWM even channels outputs: PWM0, PWM2 and PWM4. Each channel is running off its own duty-cycle generator module. The states of PWM1, PWM3 and PWM5 are reset to 0 by default if PWM Mask output function is not enabled (PMSKE=0x00).

### 14.6.2 Complementary Mode

Complementary mode is enabled when PMOD.1-0 = 01b.

In this module there are three duty-cycle generators utilized for complementary mode, with total of three PWM output pair pins in this module. The total six PWM outputs are grouped into output pairs of even and odd numbered outputs. In complimentary modes, the internal odd PWM signal PGx must always be the complement of the corresponding even PWM signal. For example, PG1 will be the complement of PG0. PG3 will be the complement of PG2 and PG5 will be the complement of PG4. The time base for the PWM module is provided by its own 16-bit timer, which also incorporates selectable pre-scalar options.

### 14.6.3 Dead-Time Insertion

The dead time generator inserts an “off” period called “dead time” between the turnings off of one pin to the turning on of the complementary pin of the paired pins. This is to prevent damage to the power switching devices that will be connected to the PWM output pins. The complementary output pair mode has an **11-bit down counter** used to produce the dead time insertion. The complementary outputs are delayed until the timer counts down to 0.

The dead-time can be calculated from the following formula:

$$\text{Dead-time} = \text{EPWMx\_CLK} * (\text{DTCNT}[10:0] + 1).$$

The timing diagram below indicates the dead time insertion for one pair of PWM signals.

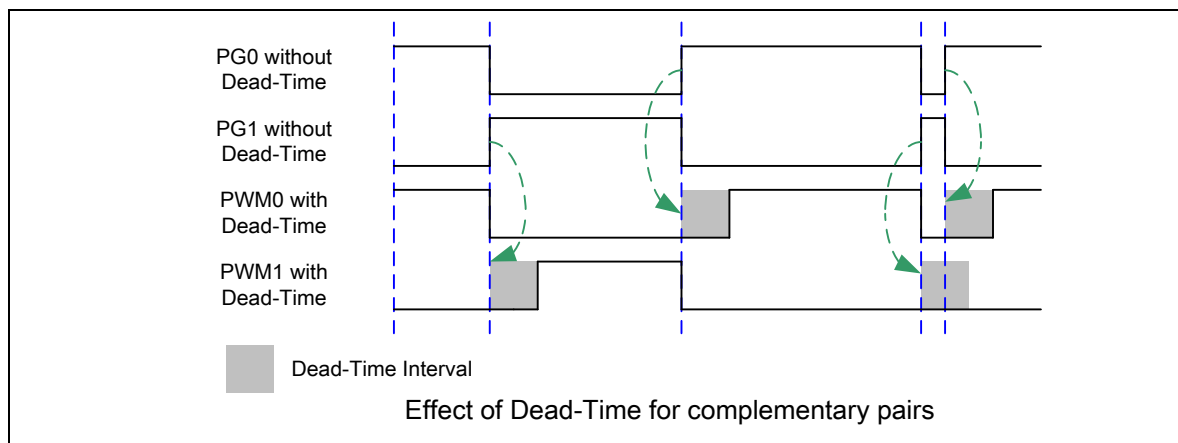


Figure 14–14 Dead-Time Insertion

The PWM Dead-time Control Register, PDTC, **has write-protection**. In the power inverter applications, a dead time insertion avoids the upper and lower switches of the half bridge from being



active at the same time. Hence the dead time control is crucial to proper operation of a system. Some amount of time must be provided between turning off of one PWM output in a complementary pair and turning on the other transistor as the power output devices cannot switch instantaneously.

### 14.6.4 Synchronous Mode

Synchronous mode is enabled when  $PMOD.1-0 = 10b$ .

In the synchronization mode the PWM pair signals from PWM Generator are in-phase.

$PG1=PG0$ ,  $PG3=PG2$  and  $PG5=PG4$ .

### 14.6.5 Group Mode

Group mode is enabled when  $GRP (PWMCONx0[13]) = 1$ .

This device supports Group Mode control. This control allows all even PWM channels output to be duty controllable by PWM0 duty register.

If  $GRP = 1$ , both  $(PG2, PG3)$  and  $(PG4, PG5)$  pairs will follow  $(PG0, PG1)$ , which imply;

$PG4 = PG2 = PG0$ ;

$PG5 = PG3 = PG1 = \text{invert}(PG0)$  if Complementary mode is enabled ( $PMOD.1-0=01b$ )

## 14.7 Polarity Control

Each PWM port of from PWM0 to PWM5 has independent polarity control to configure the polarity of active state of PWM output. By default, the PWM output is active high. This implies the PWM OFF state is low and ON state is high. This is controllable through the PWM Negative Polarity Control Register, PNP, on each individual PWM channel.

The following diagram show the initial state before PWM starts with different polarity settings.

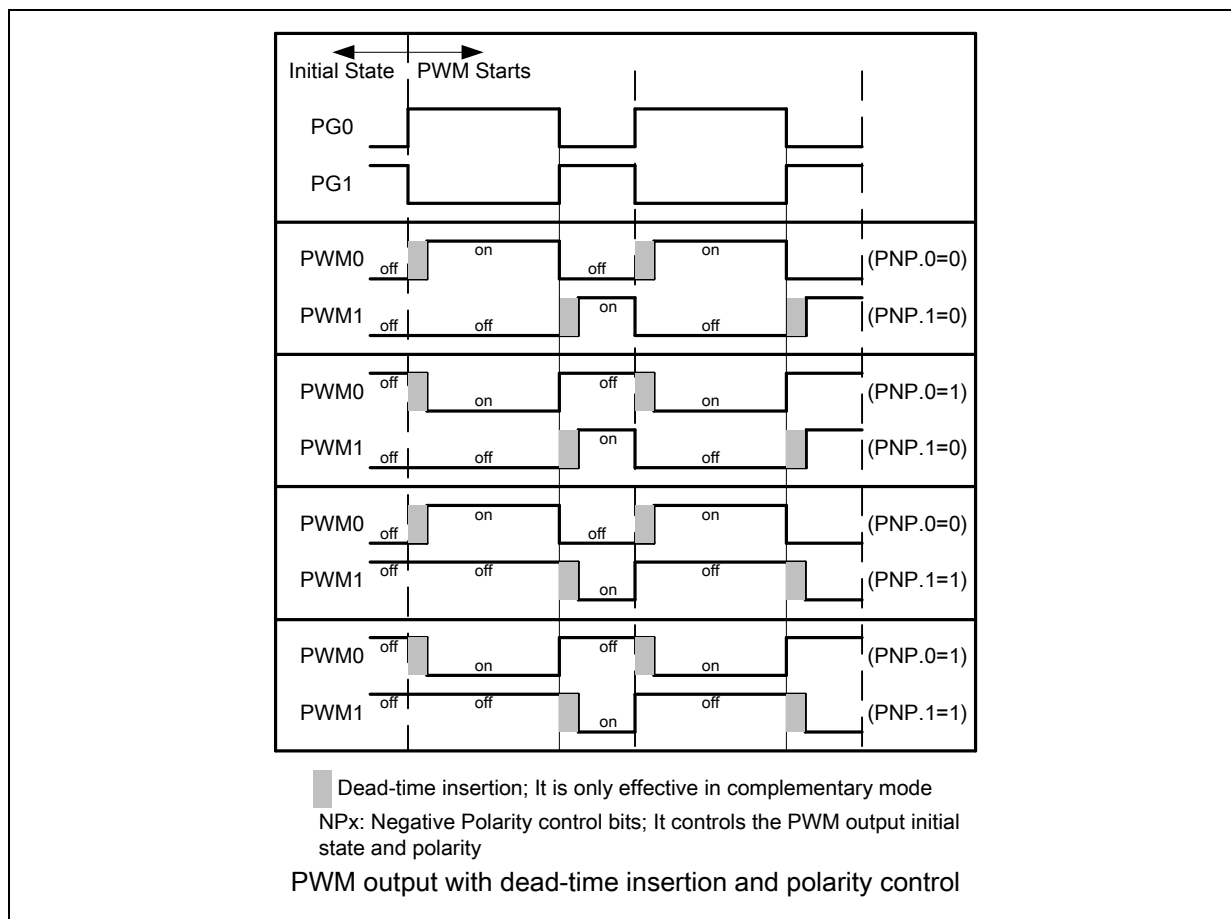


Figure 14–15 Initial State and Polarity Control with Rising Edge Dead Time Insertion

## 14.8 PWM Mask Output Function

Each of the PWM output channels can be manually overridden by using the appropriate bits in the PWM Mask Enable Control Register (PMSKE) and PWM Masked Data Register (PMSKD) to drive the PWM I/O pins to specified logic states independent of the duty cycle comparison units. The PWM mask bits are useful when controlling various types of Electrically Commutated Motor (ECM) like a BLDC motor. The PMSKE register contains six bits, PMSKE[5:0] determine which PWM I/O pins will be overridden. On reset PMSKE is 00H. The PMSKD register contains six bits, PMSKD[5:0] determine the state of the PWM I/O pins when a particular output is masked via the PMSKD bits. On reset PMSKD is 00H. The PMSKE[5:0] bits are active-high. When the PMSKE[5:0] bits are set, the corresponding PMSKD[5:0] bit will have effect on the PWM channel. When one of the PMSKE bits is sets, the output on the corresponding PWM I/O pin will be determined by the state of the PMSKD bit and polarity control bit.

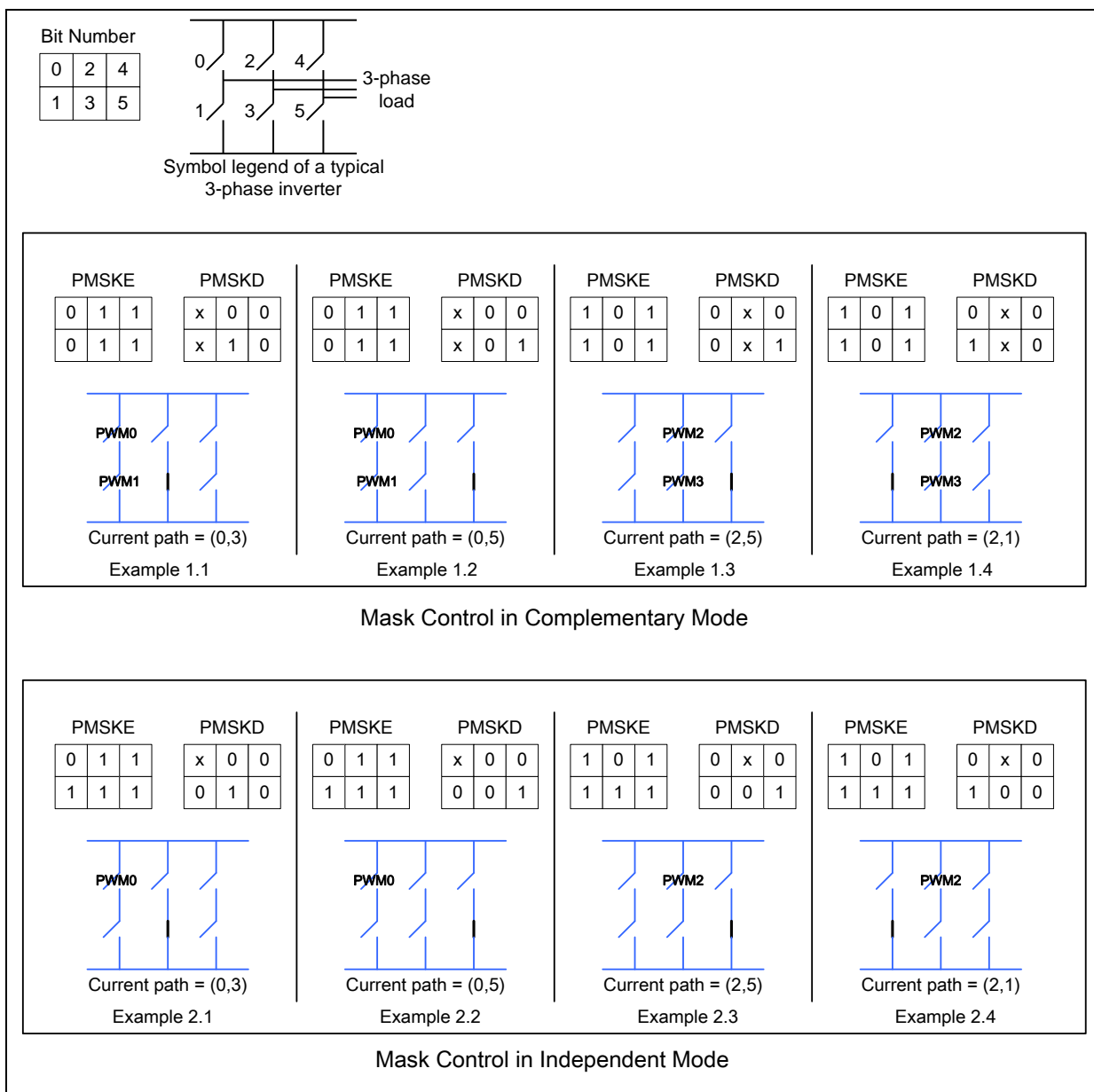


Figure 14–16 Illustration of Mask Control

Figure above shows example of how PWM mask control can be used for the override feature.

**In example 1.1; PMSKE[5:0] = 11 1100b, PMSKD[5:0] = 0010xxb (Complementary Mode)**

PWM channels 2-5 are masked from PWM frequency/duty generators.

PWM channels 2-5 outputs are determined by state of PMSKD bits.

PWM channels 0 and 1 follow PWM generator.

Switch 0 (On/Off) : Control by PWM0 (pwm0 frequency/duty generator).

Switch 1 (On/Off) : Control by PWM1 (inverted of PWM0, complementary mode).

Switch 2 (Off) : PMSKD.2 = 0.

Switch 3 (On) : PMSKD.3 = 1.

Switch 4 (Off) : PMSKD.4 = 0.

Switch 5 (Off) : PMSKD.5 = 0.

**In example 1.3; PMSKE[5:0] = 11 0011b, PMSKD[5:0] = 10xx00b (Complementary Mode)**

PWM channels 0, 1, 4 and 5 are masked from PWM frequency/duty generators.

PWM channels 0, 1, 4 and 5 outputs are determined by state of PMSKD bits.

PWM channels 2 and 3 follow PWM generator.

Switch 0 (Off) : PMSKD.0 = 0.

Switch 1 (Off) : PMSKD.1 = 0.

Switch 2 (On/Off) : Control by PWM2 (pwm2 frequency/duty generator).

Switch 3 (On/Off) : Control by PWM3 (inverted of PWM2, complementary mode).

Switch 4 (Off) : PMSKD.4 = 0.

Switch 5 (On) : PMSKD.5 = 1.

**In example 2.1; PMSKE[5:0] = 11 1110b, PMSKD[5:0] = 00100xb (Independent Mode)**

PWM channels 1-5 are masked from PWM frequency/duty generators.

PWM channels 1-5 outputs are determined by state of PMSKD bits.

PWM channel 0 follow PWM generator.

Switch 0 (On/Off) : Control by PWM0 (pwm0 frequency/duty generator).

Switch 1 (Off) : PMSKD.1 = 0.

Switch 2 (Off) : PMSKD.2 = 0.

Switch 3 (On) : PMSKD.3 = 1.

Switch 4 (Off) : PMSKD.4 = 0.

Switch 5 (Off) : PMSKD.5 = 0.

**In example 2.3; PMSKE[5:0] = 11 1011b, PMSKD[5:0] = 100x00b (Independent Mode)**

PWM channels 0,1,3,4 and 5 are masked from PWM frequency/duty generators.

PWM channels 0,1,3,4 and 5 outputs are determined by state of PMSKD bits.

PWM channel 2 follow PWM generator.

Switch 0 (Off) : PMSKD.0 = 0.

Switch 1 (Off) : PMSKD.1 = 0.

Switch 2 (On/Off) : Control by PWM2 (pwm2 frequency/duty generator).

Switch 3 (Off) : PMSKD.3 = 0.

Switch 4 (Off) : PMSKD.4 = 0.

Switch 5 (On) : PMSKD.5 = 1.

### 14.9 Interrupt Architecture of Enhanced PWM

There are six interrupt sources for each PWM unit, including PWM period flag (PWMF), Brake0 flag (BKF0), Brake1 flag (BKF1), PWM0 edge flag, PWM2 edge flag and PWM4 edge flag. The bit BRKI\_EN (PWMCONx[5]) controls the brake interrupt enable; the bit PWMI\_EN (PWMCONx[4]) controls the PWM periodic interrupt enable; the bit PWM0EI\_EN (PWMEICx[0]) controls the PWM0 edge interrupt enable; the bit PWM2EI\_EN (PWMEICx[1]) controls the PWM2 edge interrupt enable; the bit PWM4EI\_EN (PWMEICx[2]) controls the PWM4 edge interrupt enable.

**Note:** All the interrupt flags are set by hardware and must be cleared by writing 1 to flags through software.

Figure 14–17 demonstrates the architecture of enhanced PWM interrupts.

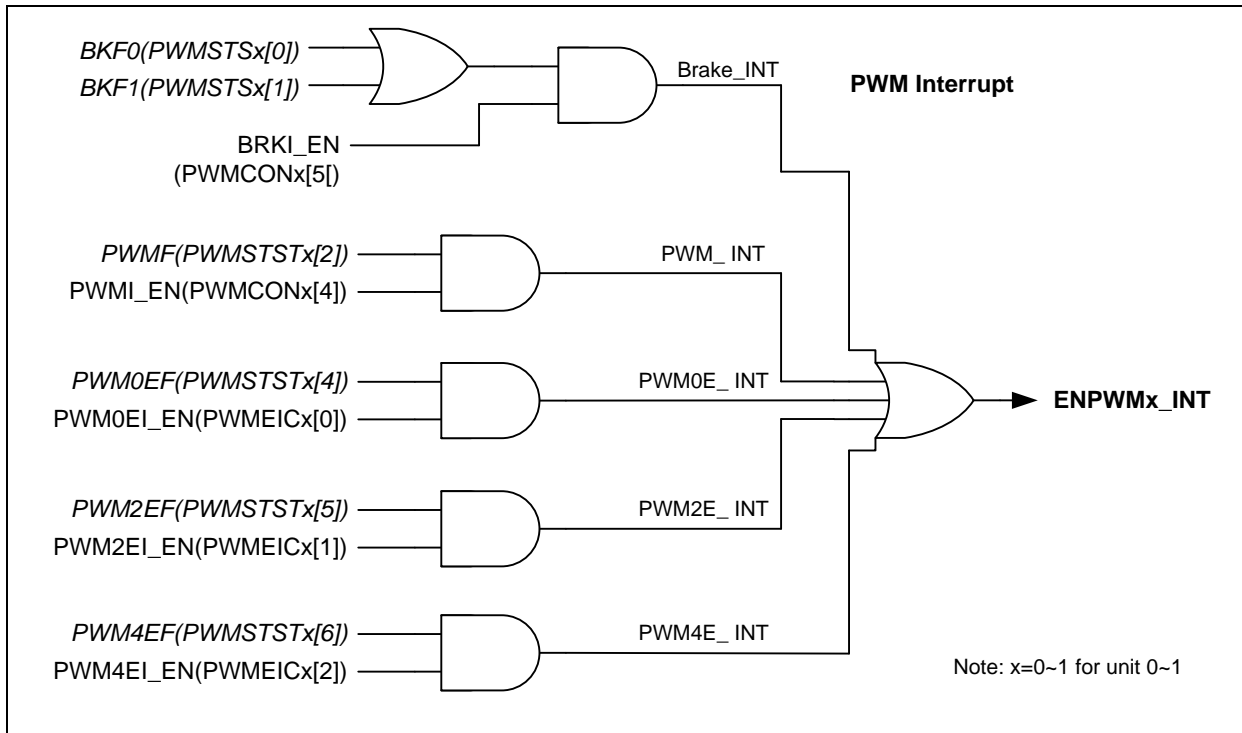


Figure 14–17 Architecture of Enhanced PWM Interrupts

### 14.10 Register Map

R: read only, W: write only, R/W: both read and write

| Register   | Offset        | R/W | Description                          | Reset Value |
|--|---------------|-----|--------------------------------------|-------------|
| <b>EPWM Base Address:</b><br>$EPWMx\_BA = 0x4019\_0000 + (0x4000 * x)$<br>$x = 0, 1$ |               |     |                                      |             |
| <b>PWMCON</b>  | EPWMx_BA+0x00 | R/W | EPWM Control Register                | 0x0000_0000 |
| <b>PWMSTS</b>  | EPWMx_BA+0x04 | R/W | EPWM Status Register                 | 0x0000_0000 |
| <b>PWMP</b>  | EPWMx_BA+0x08 | R/W | EPWM Period Register                 | 0x0000_0000 |
| <b>PWM0</b>  | EPWMx_BA+0x0C | R/W | EPWM PWM0 Duty Register              | 0x0000_0000 |
| <b>PWM2</b>  | EPWMx_BA+0x10 | R/W | EPWM PWM2 Duty Register              | 0x0000_0000 |
| <b>PWM4</b>  | EPWMx_BA+0x14 | R/W | EPWM PWM4 Duty Register              | 0x0000_0000 |
| <b>PMSKE</b>   | EPWMx_BA+0x18 | R/W | EPWM Mask Mode Enable Register       | 0x0000_0000 |
| <b>PMSKD</b>   | EPWMx_BA+0x1C | R/W | EPWM Mask Mode Data Register         | 0x0000_0000 |
| <b>PDTC</b>  | EPWMx_BA+0x2C | R/W | EPWM Dead-time Control Register      | 0x0000_0000 |
| <b>PWMB</b>  | EPWMx_BA+0x30 | R/W | EPWM Brake Output                    | 0x0000_0000 |
| <b>PNPC</b>  | EPWMx_BA+0x34 | R/W | EPWM Negative Polarity Control       | 0x0000_0000 |
| <b>PWMFCNT</b>   | EPWMx_BA+0x3C | R/W | EPWMF Compared Counter               | 0x0000_0000 |
| <b>PWMEIC</b>  | EPWMx_BA+0x40 | R/W | EPWM Edge Interrupt Control Register | 0x0000_0000 |

### 14.11 Register Description

#### EPWM Period Register (PWMP)

| Register | Offset        | R/W | Description          | Reset Value |
|----------|---------------|-----|----------------------|-------------|
| PWMP     | EPWMx_BA+0x08 | R/W | EPWM Period Register | 0x0000_0000 |

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -           |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -           |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PWMP [15:8] |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PWMP [7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | PWMP        | <b>PWM Period Register</b><br><u>Edge-aligned:</u><br>$\text{Period} = (\text{PWMP} + 1) * \text{EPWMx\_CLK period/pre-scalar}$<br>$\text{Duty} = (\text{Duty} + 1) * \text{EPWMx\_CLK period /pre-scalar}$<br><u>Centre-aligned:</u><br>$\text{Period} = (\text{PWMP} * 2) * \text{EPWMx\_CLK period/pre-scalar}$<br>$\text{Duty} = (\text{Duty} * 2 + 1) * \text{EPWMx\_CLK period/pre-scalar}$ |



**EPWM Duty Register (PWM0/2/4)**

| Register    | Offset        | R/W | Description             | Reset Value |
|-------------|---------------|-----|-------------------------|-------------|
| <b>PWM0</b> | EPWMx_BA+0x0C | R/W | EPWM PWM0 Duty Register | 0x0000_0000 |
| <b>PWM2</b> | EPWMx_BA+0x10 | R/W | EPWM PWM2 Duty Register | 0x0000_0000 |
| <b>PWM4</b> | EPWMx_BA+0x14 | R/W | EPWM PWM4 Duty Register | 0x0000_0000 |

|                 |    |    |    |    |    |    |    |
|-----------------|----|----|----|----|----|----|----|
| 31              | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -               |    |    |    |    |    |    |    |
| 23              | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -               |    |    |    |    |    |    |    |
| 15              | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PWM_Duty [15:8] |    |    |    |    |    |    |    |
| 7               | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PWM_Duty [7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | PWM_Duty    | <b>PWM Duty Register</b><br><u>Edge-aligned:</u><br>$\text{Period} = (\text{PWMP} + 1) * \text{EPWMx\_CLK period/pre-scalar}$<br>$\text{Duty} = (\text{Duty} + 1) * \text{EPWMx\_CLK period /pre-scalar}$<br><u>Centre-aligned:</u><br>$\text{Period} = (\text{PWMP} * 2) * \text{EPWMx\_CLK period/pre-scalar}$<br>$\text{Duty} = (\text{Duty} * 2 + 1) * \text{EPWMx\_CLK period/pre-scalar}$ |

### EPWM Mask Enable Control Register (PMSKE)

| Register     | Offset        | R/W | Description                    | Reset Value |
|--------------|---------------|-----|--------------------------------|-------------|
| <b>PMSKE</b> | EPWMx_BA+0x18 | R/W | EPWM Mask Mode Enable Register | 0x0000_0000 |

|    |    |                    |    |    |    |    |    |
|----|----|--------------------|----|----|----|----|----|
| 31 | 30 | 29                 | 28 | 27 | 26 | 25 | 24 |
| -  |    |                    |    |    |    |    |    |
| 23 | 22 | 21                 | 20 | 19 | 18 | 17 | 16 |
| -  |    |                    |    |    |    |    |    |
| 15 | 14 | 13                 | 12 | 11 | 10 | 9  | 8  |
| -  |    |                    |    |    |    |    |    |
| 7  | 6  | 5                  | 4  | 3  | 2  | 1  | 0  |
| -  |    | <b>PMSKE [5:0]</b> |    |    |    |    |    |

| Bits   | Description  |   |
|--------|--------------|---|
| [31:6] | -            | <b>Reserved.</b>  |
| [5:0]  | <b>PMSKE</b> | <p><b>PWM Mask Enable Bit</b></p> <p>The PWM generator signal will be masked when this bit is enabled. The corresponding PWMn channel will be output with PMD.n data.</p> <p>0 = PWM generator signal is output to next stage.</p> <p>1 = PWM generator signal is masked and PMD.n is output to next stage.</p> <p><b>Note:</b> n = 0~5</p> |

### EPWM Mask Data Register (PMSKD)

| Register     | Offset        | R/W | Description                  | Reset Value |
|--------------|---------------|-----|------------------------------|-------------|
| <b>PMSKD</b> | EPWMx_BA+0x1C | R/W | EPWM Mask Mode Data Register | 0x0000_0000 |

|    |    |                    |    |    |    |    |    |
|----|----|--------------------|----|----|----|----|----|
| 31 | 30 | 29                 | 28 | 27 | 26 | 25 | 24 |
| -  |    |                    |    |    |    |    |    |
| 23 | 22 | 21                 | 20 | 19 | 18 | 17 | 16 |
| -  |    |                    |    |    |    |    |    |
| 15 | 14 | 13                 | 12 | 11 | 10 | 9  | 8  |
| -  |    |                    |    |    |    |    |    |
| 7  | 6  | 5                  | 4  | 3  | 2  | 1  | 0  |
| -  |    | <b>PMSKD [5:0]</b> |    |    |    |    |    |

| Bits   | Description  |  |
|--------|--------------|--|
| [31:6] | -            | <b>Reserved.</b>   |
| [5:0]  | <b>PMSKD</b> | <b>PWM Mask Data Bit</b><br>This data bit control the state of PWMn output pin, if corresponding PME.n = 1.<br>0 = Output logic low to PWMn.<br>1 = Output logic high to PWMn.<br><b>Note:</b> n = 0~5 |

### EPWM Control Register (PWMCON)

| Register | Offset        | R/W | Description           | Reset Value |
|----------|---------------|-----|-----------------------|-------------|
| PWMCON   | EPWMx_BA+0x00 | R/W | EPWM Control Register | 0x0000_0000 |

| 31      | 30      | 29        | 28        | 27       | 26        | 25        | 24        |
|---------|---------|-----------|-----------|----------|-----------|-----------|-----------|
| CLDMD   | AUTOLD  | BK1NF_DIS | BK0NF_DIS | LVDBK_EN | CPO2BK_EN | CPO1BK_EN | CPO0BK_EN |
| 23      | 22      | 21        | 20        | 19       | 18        | 17        | 16        |
| BK1FILT |         | BK0FILT   |           | BK1SEL   |           | BK1_EN    | BK0_EN    |
| 15      | 14      | 13        | 12        | 11       | 10        | 9         | 8         |
| INVBKP1 | INVBKP0 | GRP       | PWMTYPE   | CLEARPWM | -         | PWMINV    | INT_TYPE  |
| 7       | 6       | 5         | 4         | 3        | 2         | 1         | 0         |
| PWMRUN  | LOAD    | BRKI_EN   | PWMI_EN   | PWMDIV   |           | PMOD      |           |

| Bits | Description |   |
|------|-------------|---|
| [31] | CLDMD       | <b>Center Reload Mode Enable</b><br>1 = EPWM reload duty register at the center point of PWM counter.<br>0 = EPWM reload duty register at the period point of PWM counter.<br>This bit only works when EPWM operating in Center-aligned mode. |
| [30] | AUTOLD      | <b>Auto Load Enable Bit</b><br>1 = PWM duty registers PWMx0~PWMx4 are auto-load from motor drive unit (MDU) when MDU update a set of new duty values for PWM unit.<br>0 = PWM duty registers PWMx0~PWMx4 are updated by software.             |
| [29] | BK1NF_DIS   | <b>PWM Brake 1 Noise Filter Disable</b><br>1 = Noise filter of PWM Brake 1 Disabled<br>0 = Noise filter of PWM Brake 1 Enabled  |
| [28] | BK0NF_DIS   | <b>PWM Brake 0 Noise Filter Disable</b><br>1 = Noise filter of PWM Brake 0 Disabled<br>0 = Noise filter of PWM Brake 0 Enabled  |
| [27] | LVDBK_EN    | <b>Low-level Detection Trigger PWM Brake Function 1 Enable</b><br>1 = Brake Function 1 triggered by Low-level detection Enabled<br>0 = Brake Function 1 triggered by Low-level detection Disabled   |
| [26] | CPO2BK_EN   | <b>CPO2 Digital Output as Brake 0 Source Enable</b><br>1 = CPO2 as one brake source in Brake 0 Enabled<br>0 = CPO2 as one brake source in Brake 0 Disabled  |

| Bits        | Description                                 |  |             |                   |    |   |    |  |    |  |    |  |
|-------------|---|--|-------------|-------------------|----|---|----|--|----|--|----|--|
| [25]        | CPO1BK_EN                                   | <b>Enable CPO1 Digital Output as Brake 0 Source</b><br>1 = CPO1 as one brake source in Brake 0 Enabled<br>0 = CPO1 as one brake source in Brake 0 Disabled   |             |                   |    |   |    |  |    |  |    |  |
| [24]        | CPO0BK_EN                                   | <b>Enable CPO0 Digital Output as Brake0 Source</b><br>1 = CPO0 as one brake source in Brake 0 Enabled<br>0 = CPO0 as one brake source in Brake 0 Disabled  |             |                   |    |   |    |  |    |  |    |  |
| [23:22]     | BK1FILT                                     | <b>Brake 1 (BKPx1 pin) Edge Detector Filter Clock Selection</b> <table><tr><th>BK1FLT[1:0]</th><th></th></tr><tr><td>00</td><td>Filter clock = HCLK</td></tr><tr><td>01</td><td>Filter clock = HCLK/2</td></tr><tr><td>10</td><td>Filter clock = HCLK/4</td></tr><tr><td>11</td><td>Filter clock = HCLK/16</td></tr></table>   | BK1FLT[1:0] |                   | 00 | Filter clock = HCLK                         | 01 | Filter clock = HCLK/2                  | 10 | Filter clock = HCLK/4                  | 11 | Filter clock = HCLK/16                 |
| BK1FLT[1:0] |   |  |             |                   |    |   |    |  |    |  |    |  |
| 00          | Filter clock = HCLK                         |  |             |                   |    |   |    |  |    |  |    |  |
| 01          | Filter clock = HCLK/2                       |  |             |                   |    |   |    |  |    |  |    |  |
| 10          | Filter clock = HCLK/4                       |  |             |                   |    |   |    |  |    |  |    |  |
| 11          | Filter clock = HCLK/16                      |  |             |                   |    |   |    |  |    |  |    |  |
| [21:20]     | BK0FILT                                     | <b>Brake 0 (BKPx0 pin) Edge Detector Filter Clock Selection</b> <table><tr><th>BK0FLT[1:0]</th><th></th></tr><tr><td>00</td><td>Filter clock = HCLK</td></tr><tr><td>01</td><td>Filter clock = HCLK/2</td></tr><tr><td>10</td><td>Filter clock = HCLK/4</td></tr><tr><td>11</td><td>Filter clock = HCLK/16</td></tr></table>   | BK0FLT[1:0] |                   | 00 | Filter clock = HCLK                         | 01 | Filter clock = HCLK/2                  | 10 | Filter clock = HCLK/4                  | 11 | Filter clock = HCLK/16                 |
| BK0FLT[1:0] |   |  |             |                   |    |   |    |  |    |  |    |  |
| 00          | Filter clock = HCLK                         |  |             |                   |    |   |    |  |    |  |    |  |
| 01          | Filter clock = HCLK/2                       |  |             |                   |    |   |    |  |    |  |    |  |
| 10          | Filter clock = HCLK/4                       |  |             |                   |    |   |    |  |    |  |    |  |
| 11          | Filter clock = HCLK/16                      |  |             |                   |    |   |    |  |    |  |    |  |
| [19:18]     | BK1SEL                                      | <b>Brake Function 1 Source Selection</b> <table><tr><th>BK1SEL[1:0]</th><th>Brake Signal From</th></tr><tr><td>00</td><td>From external pin BKPx1 (x=0~1 for unit0~1)</td></tr><tr><td>01</td><td>From analog comparator 0 output (CPO0)</td></tr><tr><td>10</td><td>From analog comparator 1 output (CPO1)</td></tr><tr><td>11</td><td>From analog comparator 2 output (CPO2)</td></tr></table> | BK1SEL[1:0] | Brake Signal From | 00 | From external pin BKPx1 (x=0~1 for unit0~1) | 01 | From analog comparator 0 output (CPO0) | 10 | From analog comparator 1 output (CPO1) | 11 | From analog comparator 2 output (CPO2) |
| BK1SEL[1:0] | Brake Signal From                           |  |             |                   |    |   |    |  |    |  |    |  |
| 00          | From external pin BKPx1 (x=0~1 for unit0~1) |  |             |                   |    |   |    |  |    |  |    |  |
| 01          | From analog comparator 0 output (CPO0)      |  |             |                   |    |   |    |  |    |  |    |  |
| 10          | From analog comparator 1 output (CPO1)      |  |             |                   |    |   |    |  |    |  |    |  |
| 11          | From analog comparator 2 output (CPO2)      |  |             |                   |    |   |    |  |    |  |    |  |
| [17]        | BKEN1                                       | <b>BKPx1 Pin Trigger Brake Function Enable</b><br>1 = PWMx Brake Function 1 Enabled.<br>0 = PWMx Brake Function 1 Disabled.<br><b>Note:</b> x=0~1 for PWM unit0~1.   |             |                   |    |   |    |  |    |  |    |  |
| [16]        | BKEN0                                       | <b>BKPx0 Pin Trigger Brake Function0 Enable</b><br>1 = PWMx Brake Function 0 Enabled.<br>0 = PWMx Brake Function 0 Disabled.<br><b>Note:</b> x = 0~1 for PWM unit0~1.  |             |                   |    |   |    |  |    |  |    |  |
| [15]        | INVBKP1                                     | <b>Inverse BKP1 State</b><br>0 = The state of pin BKPx1 is passed to the negative edge detector.<br>1 = The inversed state of pin BKPx1 is passed to the negative edge detector.   |             |                   |    |   |    |  |    |  |    |  |

| Bits | Description     |  |
|------|-----------------|--|
| [14] | <b>INVBKP0</b>  | <b>Inverse BKP0 State</b><br>0 = The state of pin BKPx0 is passed to the negative edge detector.<br>1 = The inversed state of pin BKPx0 is passed to the negative edge detector.   |
| [13] | <b>GRP</b>      | <b>Group bit</b><br>1 = Unify the signals timing of PWM0, PWM2 and PWM4 in the same phase which is controlled by PWM0.<br>0 = The signals timing of PWM0, PWM2 and PWM4 are independent.   |
| [12] | <b>PWMTYPE</b>  | <b>PWM Aligned Type Selection Bit.</b><br>0 = Edge-aligned type.<br>1 = Centre-aligned type.   |
| [11] | <b>CLRPWM</b>   | <b>Clear PWM Counter Control Bit.</b><br>1 = Clear 16-bit PWM counter to 000H.<br><b>Note:</b> It is automatically cleared by hardware.  |
| [10] | <b>Reserved</b> | Reserved   |
| [9]  | <b>PWMINV</b>   | <b>Inverse PWM Comparator Output</b><br>When PWMINV is set to high the PWM comparator output signals will be inversed, therefore the PWM Duty (in percentage) is changed to (1-Duty) before PWMINV is set to high.<br>0 = Not inverse PWM comparator output.<br>1 = Inverse PWM comparator output.   |
| [8]  | <b>INT_TYPE</b> | <b>PWM Interrupt Type Selection Bit</b><br>0 = PWMF will be set if PWM counter underflow.<br>1 = PWMF will be set if PWM counter matches PWMP register.<br><b>Note:</b> This bit is effective when PWM is in Center-aligned mode only.   |
| [7]  | <b>PWMRUN</b>   | <b>Start PWMRUN Control Bit</b><br>0 = The PWM stops running.<br>1 = The PWM counter starts running.   |
| [6]  | <b>LOAD</b>     | <b>Reload PWM period registers (PWMP) and PWM Duty Registers (PWM0~3) Control Bit</b><br>0 = No action if written with 0. The value of PWM period register (PWMP) and PWM duty registers (PWMn0~PWMn3) are not loaded to PWM counter and Comparator registers.<br>1 = Hardware will update the value of PWM period register (PWMP) and PWM duty registers (PWMn0~PWMn3) to PWM Counter and Comparator register at the time of PWM Counter matches PWMP in Edge- and Center-aligned modes or at the time of PWM Counter down counts with underflow in Center-aligned mode.<br><b>Note1:</b> n=0-1 for PWM unit0-1.<br><b>Note2:</b> This bit is written by software, cleared by hardware, and always read as 0. |
| [5]  | <b>BRKI_EN</b>  | <b>Enable Brake0 and Brak1 Interrupt</b><br>1 = Flags BKF0 and BKF1 Enabled to trigger PWM interrupt.<br>0 = Flags BFK0 and BFK1 Disabled to trigger PWM interrupt.  |

| Bits        | Description              |  |             |             |    |                       |    |                         |    |                         |    |                          |
|-------------|--------------------------|--|-------------|-------------|----|-----------------------|----|-------------------------|----|-------------------------|----|--------------------------|
| [4]         | PWMI_EN                  | <b>Enable PWM Interrupt</b><br>1 = Flag PWMF Enabled to trigger PWM interrupt.<br>0 = Flag PWMF Disabled to trigger PWM interrupt.   |             |             |    |                       |    |                         |    |                         |    |                          |
| [3:2]       | PWMDIV                   | <b>PWM Clock Pre-divider Selection</b> <table><tr><th>PWMDIV[1:0]</th><th>Description</th></tr><tr><td>00</td><td>PWM clock = EPWMx_CLK</td></tr><tr><td>01</td><td>PWM clock = EPWMx_CLK/2</td></tr><tr><td>10</td><td>PWM clock = EPWMx_CLK/4</td></tr><tr><td>11</td><td>PWM clock = EPWMx_CLK/16</td></tr></table> | PWMDIV[1:0] | Description | 00 | PWM clock = EPWMx_CLK | 01 | PWM clock = EPWMx_CLK/2 | 10 | PWM clock = EPWMx_CLK/4 | 11 | PWM clock = EPWMx_CLK/16 |
| PWMDIV[1:0] | Description              |  |             |             |    |                       |    |                         |    |                         |    |                          |
| 00          | PWM clock = EPWMx_CLK    |  |             |             |    |                       |    |                         |    |                         |    |                          |
| 01          | PWM clock = EPWMx_CLK/2  |  |             |             |    |                       |    |                         |    |                         |    |                          |
| 10          | PWM clock = EPWMx_CLK/4  |  |             |             |    |                       |    |                         |    |                         |    |                          |
| 11          | PWM clock = EPWMx_CLK/16 |  |             |             |    |                       |    |                         |    |                         |    |                          |
| [1:0]       | PWMMOD                   | <b>PWM Mode Selection</b> <table><tr><th>PWMMOD[1:0]</th><th>Description</th></tr><tr><td>00</td><td>Independent mode</td></tr><tr><td>01</td><td>Pair/Complementary mode</td></tr><tr><td>10</td><td>Synchronized mode</td></tr><tr><td>11</td><td>Reserved</td></tr></table>   | PWMMOD[1:0] | Description | 00 | Independent mode      | 01 | Pair/Complementary mode | 10 | Synchronized mode       | 11 | Reserved                 |
| PWMMOD[1:0] | Description              |  |             |             |    |                       |    |                         |    |                         |    |                          |
| 00          | Independent mode         |  |             |             |    |                       |    |                         |    |                         |    |                          |
| 01          | Pair/Complementary mode  |  |             |             |    |                       |    |                         |    |                         |    |                          |
| 10          | Synchronized mode        |  |             |             |    |                       |    |                         |    |                         |    |                          |
| 11          | Reserved                 |  |             |             |    |                       |    |                         |    |                         |    |                          |

EPWM Status Register (PWMSTS)

| Register | Offset        | R/W | Description          | Reset Value |
|----------|---------------|-----|----------------------|-------------|
| PWMSTS   | EPWMx_BA+0x04 | R/W | EPWM Status Register | 0x0000_0000 |

|    |        |        |        |    |      |        |        |
|----|--------|--------|--------|----|------|--------|--------|
| 31 | 30     | 29     | 28     | 27 | 26   | 25     | 24     |
| -  |        |        |        |    |      | BK1STS | BK0STS |
| 23 | 22     | 21     | 20     | 19 | 18   | 17     | 16     |
| -  |        |        |        |    |      |        |        |
| 15 | 14     | 13     | 12     | 11 | 10   | 9      | 8      |
| -  |        |        |        |    |      |        | BK0STS |
| 7  | 6      | 5      | 4      | 3  | 2    | 1      | 0      |
| -  | PWM4EF | PWM2EF | PWM0EF | -  | PWMF | BKF1   | BKF0   |

| Bits    | Description |  |
|---------|-------------|--|
| [31:26] | -           | Reserved.  |
| [25]    | BK1STS      | <b>Brake 1 Status (Read Only)</b><br>1 = PWM is in Brake 1 state.<br>0 = PWM had been out of Brake 1 state   |
| [24]    | BK0STS      | <b>Brake 0 Status (Read Only)</b><br>1 = PWM is in Brake 0 state.<br>0 = PWM had been out of Brake 0 state.  |
| [23:9]  | -           | Reserved.  |
| [8]     | BK0STS      | <b>PWM Brake0 Locked</b><br>1 = When PWM Brake detects a falling signal at BK0, this flag will be set to high to indicate the Brake0 state is locked.<br>0 = Brake 0 state is released.<br><b>Note:</b> This bit must be cleared by writing 1 to itself through software.  |
| [7]     | -           | Reserved.  |
| [6]     | PWM4EF      | <b>PWMx4 Edge Flag</b><br>0 = PWMx4 not toggled.<br>1 = Hardware will set this flag to high at the time of PWMx4 rising or falling. If EINT4_TYPE = 0, this bit is set when PWMx4 falling is detected. If EINT4_TYPE = 1, this bit is set when PWMx4 rising is detected.<br><b>Note:</b> This bit must be cleared by writing 1 to itself through software. |



|     |               |   |
|-----|---------------|---|
| [5] | <b>PWM2EF</b> | <p><b>PWMx2 Edge Flag</b></p> <p>0 = PWMx2 not toggled.</p> <p>1 = Hardware will set this flag to high at the time of PWMx2 rising or falling. If EINT2_TYPE = 0, this bit is set when PWMx2 falling is detected. If EINT2_TYPE = 1, this bit is set when PWMx2 rising is detected.</p> <p><b>Note:</b> This bit must be cleared by writing 1 to itself through software.</p>   |
| [4] | <b>PWM0EF</b> | <p><b>PWMx0 Edge Flag</b></p> <p>0 = PWMx0 not toggled.</p> <p>1 = Hardware will set this flag to high at the time of PWMx0 rising or falling. If EINT0_TYPE = 0, this bit is set when PWMx0 falling is detected. If EINT0_TYPE = 1, this bit is set when PWMx0 rising is detected.</p> <p><b>Note:</b> This bit must be cleared by writing 1 to itself through software.</p>   |
| [3] | -             | <b>Reserved.</b>  |
| [2] | <b>PWMF</b>   | <p><b>PWM Period Flag.</b></p> <p>0 = The PWM Counter has not up counted to the value of PWMP or down counted with underflow.</p> <p>1 = Hardware will set this flag to high at the time of PWM Counter matches PWMP in Edge- and Center-aligned modes or at the time of PWM Counter down counts with underflow in Center-aligned mode.</p> <p><b>Note:</b> This bit must be cleared by writing 1 to itself through software.</p> |
| [1] | <b>BKF1</b>   | <p><b>PWM Brake1 Flag</b></p> <p>1 = When PWM Brake 1 detects a falling signal at pin BKP1, this flag will be set to high.</p> <p>0 = PWM Brake 1 is able to poll falling signal at BKP1 and has not recognized any one</p> <p><b>Note:</b> This bit must be cleared by writing 1 to itself through software.</p>   |
| [0] | <b>BKF0</b>   | <p><b>PWM Brake0 Flag</b></p> <p>1 = When PWM Brake 0 detects a falling signal at BKP0, this flag will be set to high</p> <p>0 = PWM Brake 0 is able to poll falling signal at BKP0 and has not recognized any one</p> <p><b>Note:</b> This bit must be cleared by writing 1 to itself through software.</p>  |

**EPWM Dead-time Control Register (PDTC)**

| Register | Offset        | R/W | Description                     | Reset Value |
|----------|---------------|-----|---------------------------------|-------------|
| PDTC     | EPWMx_BA+0x2C | R/W | EPWM Dead-time Control Register | 0x0000_0000 |

|            |    |    |    |    |             |       |       |
|------------|----|----|----|----|-------------|-------|-------|
| 31         | 30 | 29 | 28 | 27 | 26          | 25    | 24    |
| -          |    |    |    |    |             |       |       |
| 23         | 22 | 21 | 20 | 19 | 18          | 17    | 16    |
| -          |    |    |    |    | DTEN4       | DTEN2 | DTEN0 |
| 15         | 14 | 13 | 12 | 11 | 10          | 9     | 8     |
| -          |    |    |    |    | DTCNT[10:8] |       |       |
| 7          | 6  | 5  | 4  | 3  | 2           | 1     | 0     |
| DTCNT[7:0] |    |    |    |    |             |       |       |

| Bits    | Description |   |
|---------|-------------|---|
| [31:19] | -           | Reserved.   |
| [18]    | DTEN4       | <p><b>Enable Dead-time Insertion for PWMx Pair (PWM4, PWM5)</b></p> <p>Dead-time insertion is only active when this pair of complementary PWM is enabled. If dead-time insertion is inactive, the outputs of pin pair are complementary without any delay.</p> <p>0 = Dead-time insertion Disabled on the pin pair (PWM4, PWM5).<br/>1 = Dead-time insertion Enabled on the pin pair (PWM4, PWM5).</p> <p><b>Note:</b> x=0~1 for PWM unit0~1.</p> |
| [17]    | DTEN2       | <p><b>Enable Dead-time Insertion for PWMx Pair (PWM2, PWM3)</b></p> <p>Dead-time insertion is only active when this pair of complementary PWM is enabled. If dead-time insertion is inactive, the outputs of pin pair are complementary without any delay.</p> <p>0 = Dead-time insertion Disabled on the pin pair (PWM2, PWM3).<br/>1 = Dead-time insertion Enabled on the pin pair (PWM2, PWM3).</p> <p><b>Note:</b> x=0~1 for PWM unit0~1.</p> |
| [16]    | DTEN0       | <p><b>Enable Dead-time Insertion for PWMx Pair (PWM0, PWM1)</b></p> <p>Dead-time insertion is only active when this pair of complementary PWM is enabled. If dead-time insertion is inactive, the outputs of pin pair are complementary without any delay.</p> <p>0 = Dead-time insertion Disabled on the pin pair (PWM0, PWM1).<br/>1 = Dead-time insertion Enabled on the pin pair (PWM0, PWM1).</p> <p><b>Note:</b> x=0~1 for PWM unit0~1.</p> |
| [15:11] | -           | Reserved.   |

| Bits   | Description  |  |
|--------|--------------|--|
| [10:0] | <b>DTCNT</b> | <b>Dead-time Counter</b><br>The dead-time can be calculated according to the following formula:<br>$\text{Dead-time} = \text{EPWMx\_CLK} * (\text{DTCNT}.[10:0] + 1).$ |

### EPWM Brake Output (PWMB)

| Register | Offset        | R/W | Description       | Reset Value |
|----------|---------------|-----|-------------------|-------------|
| PWMB     | EPWMx_BA+0x30 | R/W | EPWM Brake Output | 0x0000_0000 |

|    |    |           |    |    |    |    |    |
|----|----|-----------|----|----|----|----|----|
| 31 | 30 | 29        | 28 | 27 | 26 | 25 | 24 |
| -  |    |           |    |    |    |    |    |
| 23 | 22 | 21        | 20 | 19 | 18 | 17 | 16 |
| -  |    |           |    |    |    |    |    |
| 15 | 14 | 13        | 12 | 11 | 10 | 9  | 8  |
| -  |    |           |    |    |    |    |    |
| 7  | 6  | 5         | 4  | 3  | 2  | 1  | 0  |
| -  |    | PWMB[5:0] |    |    |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:6] | -           | Reserved.   |
| [5:0]  | PWMB        | <p><b>PWM Brake Output</b></p> <p>When PWM Brake is asserted, the PWM0~5 output state before polarity control will follow PWMB bit0~5 setting, respectively.</p> <p>0 = PWMn output before polarity control is low when Brake is asserted.</p> <p>1 = PWMn output before polarity control is high when Brake is asserted.</p> <p><b>Note:</b> n = 0~5</p> |

### EPWM Negative Polarity Control (PNPC)

| Register    | Offset        | R/W | Description                    | Reset Value |
|-------------|---------------|-----|--------------------------------|-------------|
| <b>PNPC</b> | EPWMx_BA+0x34 | R/W | EPWM Negative Polarity Control | 0x0000_0000 |

|    |    |            |    |    |    |    |    |
|----|----|------------|----|----|----|----|----|
| 31 | 30 | 29         | 28 | 27 | 26 | 25 | 24 |
| -  |    |            |    |    |    |    |    |
| 23 | 22 | 21         | 20 | 19 | 18 | 17 | 16 |
| -  |    |            |    |    |    |    |    |
| 15 | 14 | 13         | 12 | 11 | 10 | 9  | 8  |
| -  |    |            |    |    |    |    |    |
| 7  | 6  | 5          | 4  | 3  | 2  | 1  | 0  |
| -  |    | PNPn [5:0] |    |    |    |    |    |

| Bits   | Description |  |
|--------|-------------|--|
| [31:6] | -           | Reserved.  |
| [5:0]  | <b>PNPn</b> | <b>PWM Negative Polarity Control</b><br>The register bit controls polarity/active state of real PWM output.<br>0 = PWMn output is active high.<br>1 = PWMn output is active low.<br><b>Note:</b> n = 0~5 |

**EPWMF Compared Counter (PWMFCNT)**

| Register | Offset        | R/W | Description            | Reset Value |
|----------|---------------|-----|------------------------|-------------|
| PWMFCNT  | EPWMx_BA+0x3C | R/W | EPWMF Compared Counter | 0x0000_0000 |

|    |    |    |    |               |    |    |    |
|----|----|----|----|---------------|----|----|----|
| 31 | 30 | 29 | 28 | 27            | 26 | 25 | 24 |
| -  |    |    |    |               |    |    |    |
| 23 | 22 | 21 | 20 | 19            | 18 | 17 | 16 |
| -  |    |    |    |               |    |    |    |
| 15 | 14 | 13 | 12 | 11            | 10 | 9  | 8  |
| -  |    |    |    |               |    |    |    |
| 7  | 6  | 5  | 4  | 3             | 2  | 1  | 0  |
| -  |    |    |    | PWMFCNT [3:0] |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:4] | -           | Reserved.   |
| [3:0]  | PWMFCNT     | <p><b>PWMF Compared Counter</b></p> <p>The register sets the count number which defines how many times of PWM period occurs to set bit PWMF to request the PWM period interrupt.</p> <p>PWMF will be set in every (1 + PWMFCNT[3:0]) time of PWM period or center point defined by INT_TYPE at PWMCON[8] occurs</p> |

### EPWM Edge Interrupt Control Register (PWMEIC)

| Register | Offset        | R/W | Description                          | Reset Value |
|----------|---------------|-----|--------------------------------------|-------------|
| PWMEIC   | EPWMx_BA+0x40 | R/W | EPWM Edge Interrupt Control Register | 0x0000_0000 |

|    |    |    |    |    |            |            |            |
|----|----|----|----|----|------------|------------|------------|
| 31 | 30 | 29 | 28 | 27 | 26         | 25         | 24         |
| -  |    |    |    |    |            |            |            |
| 23 | 22 | 21 | 20 | 19 | 18         | 17         | 16         |
| -  |    |    |    |    |            |            |            |
| 15 | 14 | 13 | 12 | 11 | 10         | 9          | 8          |
| -  |    |    |    |    | EINT4_TYPE | EINT2_TYPE | EINT0_TYPE |
| 7  | 6  | 5  | 4  | 3  | 2          | 1          | 0          |
| -  |    |    |    |    | PWM4EI_EN  | PWM2EI_EN  | PWM0EI_EN  |

| Bits    | Description |   |
|---------|-------------|---|
| [31:11] | -           | Reserved.   |
| [10]    | EINT4_TYPE  | <b>PWMx4 Edge Interrupt Type</b><br>1 = PWM4EF will be set if rising edge is detected at PWMx4.<br>0 = PWM4EF will be set if falling edge is detected at PWMx4. |
| [9]     | EINT2_TYPE  | <b>PWMx2 Edge Interrupt Type</b><br>1 = PWM2EF will be set if rising edge is detected at PWMx2.<br>0 = PWM2EF will be set if falling edge is detected at PWMx2. |
| [8]     | EINT0_TYPE  | <b>PWMx0 Edge Interrupt Type</b><br>1 = PWM0EF will be set if rising edge is detected at PWMx0.<br>0 = PWM0EF will be set if falling edge is detected at PWMx0. |
| [7:3]   | -           | Reserved.   |
| [2]     | PWM4EI_EN   | <b>Enable PWMx4 Edge Interrupt</b><br>1 = Flag PWM4EF Enabled to trigger PWM interrupt.<br>0 = Flag PWM4EF Disabled to trigger PWM interrupt.                   |
| [1]     | PWM2EI_EN   | <b>Enable PWMx2 Edge Interrupt</b><br>1 = Flag PWM2EF Enabled to trigger PWM interrupt.<br>0 = Flag PWM2EF Disabled to trigger PWM interrupt.                   |
| [0]     | PWM0EI_EN   | <b>Enable PWMx0 Edge Interrupt</b><br>1 = Flag PWM0EF Enabled to trigger PWM interrupt.<br>0 = Flag PWM0EF Disabled to trigger PWM interrupt.                   |

## 15 MOTOR DRIVE UNIT (MDU)

### 15.1 Overview

An efficient motor control scheme, Field Orientated Control (FOC), is widely applied to a three phases AC Induction Motor (ACIM) and Permanent Magnet Synchronous Motor (PMSM). This device provides a motor drive unit (MDU) which performs the Field Orientated Control (FOC) through hardware and offers the Space Vector PWM duty timing to PWM unit0 to produce the PWM signals to power inverter.

### 15.2 Features

- Clarke and Inverse Clarke Transformation
- Park and Inverse Park Transformation
- Two built-in PI (Proportional and Integral) controllers
- Space-vector PWM (SVPWM) timing generator
- Arithmetic operation in fixed point Q-15 format
- Auto update PWM Unit0 duty registers



### 15.3 MDU Architecture

The Motor Drive Unit comprises several function blocks including Clarke, Park, Inverse Clarke and Inverse Park transformations, two PI controllers and SVPWM timing generator.

The following figures illustrate the MDU clock source control and the architecture of Motor Drive Unit Controller.

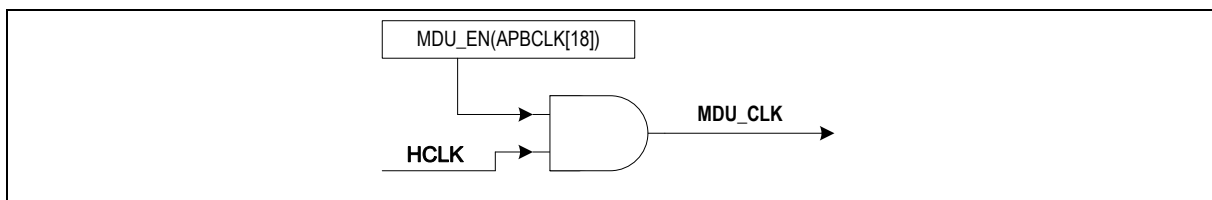


Figure 15-1 MDU Clock Source Control

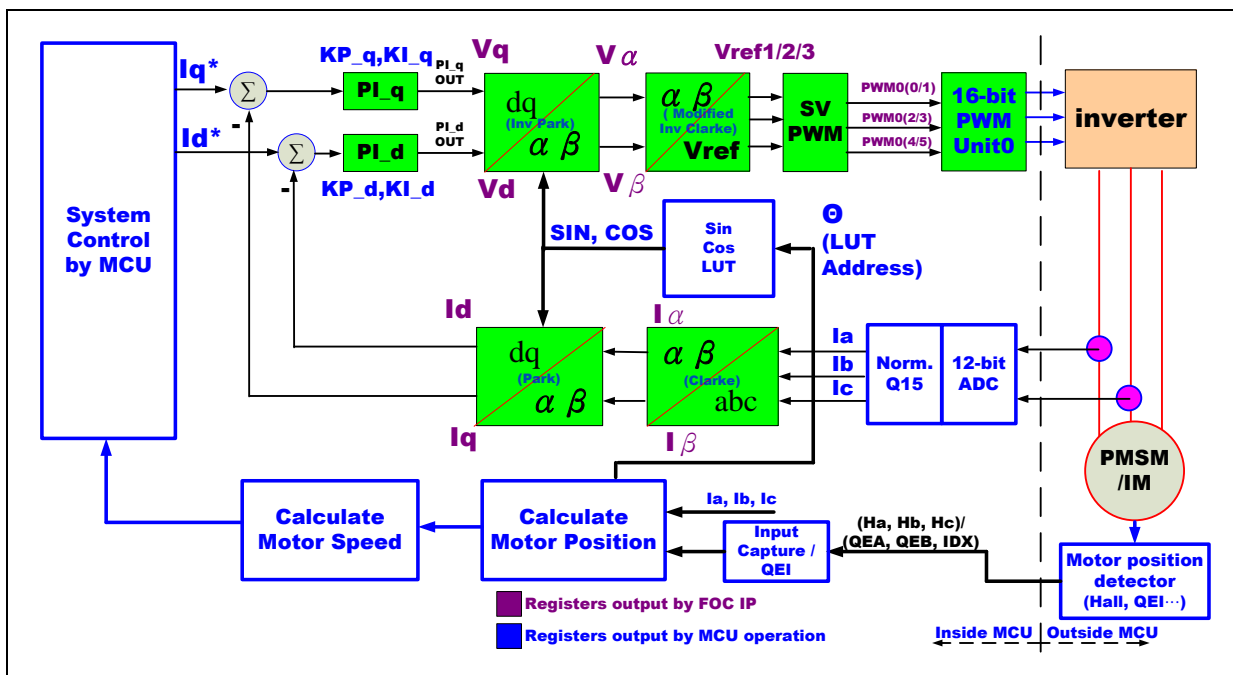


Figure 15-2 Motor Drive Unit Architecture

## 15.4 Operation of Motor Drive Unit

The Field Orientated Control (FOC) is based on projections which transform a three phase time and speed dependent system into a two co-ordinate (d and q co-ordinates) time invariant system.

All the arithmetical values in MDU, except KP0 and KP1, are normalized in a 16-bit Q15 format with one sign bit in MSB. KP0 and KP1 are represented in 18-bit I2Q15 format.

### 15.4.1 Clarke and Inverse Clarke Transformation

The Clarke transformation function block transforms the three variables registers Ia, Ib and Ic from abc-axis to stationary  $\alpha\beta$ -axis and produces two variables stored in registers Ialfa and Ibeta. The inverse Clarke transformation takes the inverse operation.

### 15.4.2 Park and Inverse Park Transformation

The Park transformation function block transforms the two variables Valfa and Vbeta from stationary  $\alpha\beta$ -axis to rotating **dq**-axis and produces two variables stored in registers Vd and Vq. The inverse Park transformation takes the inverse operation.

### 15.4.3 PI Controller

The PI controller executes current control by proportional and integral scheme to obtain a proper controlled phase voltage applied to the Motor.

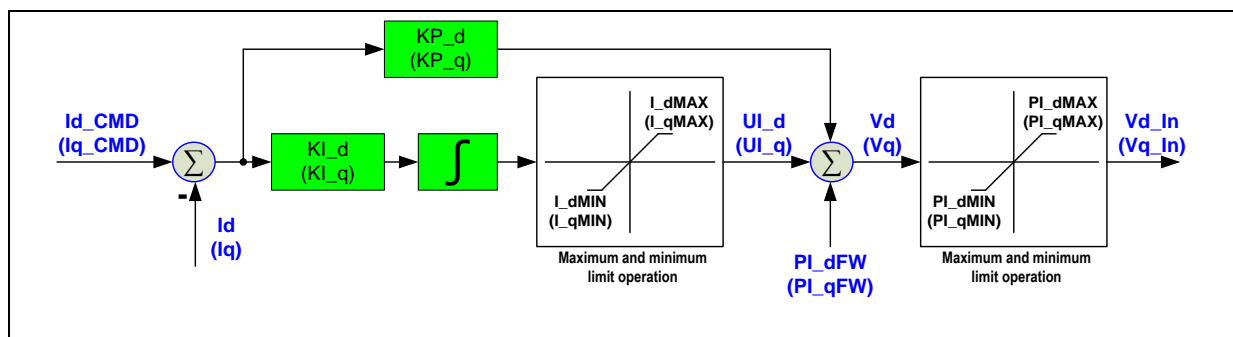


Figure 15-3 PI Controller Architecture

### 15.4.4 SVPWM Timing Generator

If the control bit SVPWM\_DIS (MDUCON[17]) is cleared, the inverse Clarke Transformation is integrated into the Space Vector PWM function block to produce three PWM duties which are applied to PWM unit0 duty double buffer registers. Consequently, PWM unit 0 may output the frequency and amplitude adjustable sinusoidal signal modulated by SVPWM scheme.

If the control bit SVPWM\_DIS is set, it will produce 3 balanced sinusoidal PWM (SPWM) duties for PWM modules.

### 15.5 Register Map

R: read only, W: write only, R/W: both read and write

| Register  | Offset       | R/W | Description  | Reset Value |
|---|--------------|-----|--|-------------|
| <b>MDU Base Address:</b><br><b>MDUx_BA = 0x401D_0000 + (0x4000 * x)</b><br>x = 0, 1<br><b>MDUG_BA = 0x401D_0800</b> |              |     |  |             |
| <b>Ia</b>   | MDUx_BA+0x00 | R/W | Phase A Current Register, represented in Q-15 Format                                 | 0x0000_0000 |
| <b>Ib</b>   | MDUx_BA+0x04 | R/W | Phase B Current Register, represented in Q-15 Format                                 | 0x0000_0000 |
| <b>Ic</b>   | MDUx_BA+0x08 | R/W | Phase C Current Register, represented in Q-15 Format                                 | 0x0000_0000 |
| <b>Ialfa</b>  | MDUx_BA+0x0C | R/W | Clarke Transformation Output Register, the Current Along alpha-axis                  | 0x0000_0000 |
| <b>Ibeta</b>  | MDUx_BA+0x10 | R/W | Clarke Transformation Output Register, the Current Along beta-axis                   | 0x0000_0000 |
| <b>SIN</b>  | MDUx_BA+0x14 | R/W | Trigonometric Value of SIN Register, represented in Q-15 Format                      | 0x0000_0000 |
| <b>COS</b>  | MDUx_BA+0x18 | R/W | Trigonometric Value of COS Register, represented in Q-15 Format                      | 0x0000_0000 |
| <b>Id</b>   | MDUx_BA+0x1C | R/W | Park Transformation Output Register, the Current Along d-axis.                       | 0x0000_0000 |
| <b>Iq</b>   | MDUx_BA+0x20 | R/W | Park Transformation Output Register, the Current Along q-axis.                       | 0x0000_0000 |
| <b>Id_CMD</b>   | MDUx_BA+0x24 | R/W | PI_d Controller Command Input Register, represented in Q-15 Format.                  | 0x0000_0000 |
| <b>Iq_CMD</b>   | MDUx_BA+0x28 | R/W | PI_q Controller Command Input Register, Represented in Q-15 Format.                  | 0x0000_0000 |
| <b>KP_d</b>   | MDUx_BA+0x2C | R/W | Parameter KP for PI_d Controller, represented in <b>18-bit I2Q-15 (Q2.15) Format</b> | 0x0000_0000 |
| <b>KI_d</b>   | MDUx_BA+0x30 | R/W | Parameter KI for PI_d Controller, represented in <b>18-bit I2Q-15 (Q2.15) Format</b> | 0x0000_0000 |
| <b>KP_q</b>   | MDUx_BA+0x34 | R/W | Parameter KP for PI_q Controller, represented in <b>18-bit I2Q-15 (Q2.15) Format</b> | 0x0000_0000 |
| <b>KI_q</b>   | MDUx_BA+0x38 | R/W | Parameter KI for PI_q Controller, Represented in <b>18-bit I2Q-15 (Q2.15) Format</b> | 0x0000_0000 |
| <b>PI_dFW</b>   | MDUx_BA+0x3C | R/W | PI_d Controller Feed Forward Data Register, represented in Q-15 Format.              | 0x0000_0000 |
| <b>PI_qFW</b>   | MDUx_BA+0x40 | R/W | PI_q Controller Feed Forward Data Register, represented in Q-15 Format               | 0x0000_0000 |
| <b>UI_d</b>   | MDUx_BA+0x44 | R/W | D-axis I Control Output Data Register, represented in Q-15 Format                    | 0x0000_0000 |
| <b>UI_q</b>   | MDUx_BA+0x48 | R/W | Q-axis I Control Output Data Register, represented in Q-15 Format                    | 0x0000_0000 |
| <b>Vd</b>   | MDUx_BA+0x4C | R/W | PI_d Controller Output of d-axis, represented in Q-15 Format                         | 0x0000_0000 |
| <b>Vq</b>   | MDUx_BA+0x50 | R/W | PI_q Controller Output of q-axis, represented in Q-15 Format                         | 0x0000_0000 |

| Register  | Offset        | R/W | Description   | Reset Value |
|---|---------------|-----|---|-------------|
| <b>MDU Base Address:</b><br><b>MDUx_BA = 0x401D_0000 + (0x4000 * x)</b><br>x = 0, 1<br><b>MDUG_BA = 0x401D_0800</b> |               |     |   |             |
| <b>Valfa</b>  | MDUx_BA+0x54  | R/W | Inverse Park Transformation Output, the Voltage Along the alpha-axis                                  | 0x0000_0000 |
| <b>Vbeta</b>  | MDUx_BA+0x58  | R/W | Inverse Park Transformation Output, the Voltage Along the beta-axis                                   | 0x0000_0000 |
| <b>Vref1</b>  | MDUx_BA+0x5C  | R/W | Inverse Clarke Transformation Output Register   | 0x0000_0000 |
| <b>Vref2</b>  | MDUx_BA+0x60  | R/W | Inverse Clarke Transformation Output Register   | 0x0000_0000 |
| <b>Vref3</b>  | MDUx_BA+0x64  | R/W | Inverse Clarke Transformation Output Register   | 0x0000_0000 |
| <b>SVPDATA0</b>   | MDUx_BA+0x68  | R/W | SVPWM output Data Register (it is an unsigned 16-bit value and denotes Taon time)                     | 0x0000_0000 |
| <b>SVPDATA2</b>   | MDUx_BA+0x6C  | R/W | SVPWM output Data Register (it is an unsigned 16-bit value and denotes Tbon time)                     | 0x0000_0000 |
| <b>SVPDATA4</b>   | MDUx_BA+0x70  | R/W | SVPWM output Data Register (it is an unsigned 16-bit value and denotes Tcon time)                     | 0x0000_0000 |
| <b>SVPDATA0N</b>  | MDUx_BA+0x74  | R/W | An unsigned 16-bit data Normalizes SVPDATA0 to PWMNORM and within the limit between PWMMAX and PWMMIN | 0x0000_0000 |
| <b>SVPDATA2N</b>  | MDUx_BA+0x78  | R/W | An unsigned 16-bit data Normalizes SVPDATA2 to PWMNORM and within the limit between PWMMAX and PWMMIN | 0x0000_0000 |
| <b>SVPDATA4N</b>  | MDUx_BA+0x7C  | R/W | An unsigned 16-bit data Normalizes SVPDATA4 to PWMNORM and within the limit between PWMMAX and PWMMIN | 0x0000_0000 |
| <b>MDUCON</b>   | MDUx_BA+0x80  | R/W | Motor Driver Unit Control Register  | 0x0000_0000 |
| <b>MDUSTS</b>   | MDUx_BA+0x84  | R/W | Motor Driver Unit Status Register   | 0x0000_0700 |
| <b>PI_dMAX</b>  | MDUx_BA+0x100 | R/W | Maximum limit value of PI_d Controller in Q-15 Format   | 0x0000_7FFF |
| <b>PI_dMIN</b>  | MDUx_BA+0x104 | R/W | Minimum limit value of PI_d Controller in Q-15 Format   | 0x0000_8000 |
| <b>PI_qMAX</b>  | MDUx_BA+0x108 | R/W | Maximum limit value of PI_q Controller in Q-15 Format   | 0x0000_7FFF |
| <b>PI_qMIN</b>  | MDUx_BA+0x10C | R/W | Minimum limit value of PI_q Controller in Q-15 Format   | 0x0000_8000 |
| <b>I_dMAX</b>   | MDUx_BA+0x110 | R/W | Maximum limit value of integral_d Controller in Q-15 Format   | 0x0000_7FFF |
| <b>I_dMIN</b>   | MDUx_BA+0x114 | R/W | Minimum limit value of integral_d Controller in Q-15 Format   | 0x0000_8000 |
| <b>I_qMAX</b>   | MDUx_BA+0x118 | R/W | Maximum limit value of integral_q Controller in Q-15 Format   | 0x0000_7FFF |
| <b>I_qMIN</b>   | MDUx_BA+0x11C | R/W | Minimum limit value of integral_q Controller in Q-15 Format   | 0x0000_8000 |
| <b>Vd_In</b>  | MDUx_BA+0x120 | R   | Inverse Park Transformation Input of d_axis represented in Q-15 Format                                | 0x0000_0000 |
| <b>Vq_In</b>  | MDUx_BA+0x124 | R   | Inverse Park Transformation Input of q_axis represented in Q-15 Format                                | 0x0000_0000 |
| <b>PWMNORM</b>  | MDUx_BA+0x128 | R/W | An Unsigned 16-bit PWM Normalization Value  | 0x0000_0000 |

| Register  | Offset        | R/W | Description  | Reset Value |
|---|---------------|-----|--|-------------|
| <b>MDU Base Address:</b><br><b>MDUx_BA = 0x401D_0000 + (0x4000 * x)</b><br>x = 0, 1<br><b>MDUG_BA = 0x401D_0800</b> |               |     |  |             |
| <b>PWMMAX</b>   | MDUx_BA+0x12C | R/W | An unsigned 16-bit Maximum Limit Value of SVPWM Output | 0x0000_FFFF |
| <b>PWMMIN</b>   | MDUx_BA+0x130 | R/W | An unsigned 16-bit Minimum Limit Value of SVPWM Output | 0x0000_0000 |
| <b>Id_Err</b>   | MDUx_BA+0x134 | R   | Error of Id current = Id_CMD – Id; Read only           | 0x0000_0000 |
| <b>Iq_Err</b>   | MDUx_BA+0x138 | R   | Error of Id current = Iq_CMD – Iq; Read only.          | 0x0000_0000 |
| <b>MDUSCON</b>  | MDUG_BA+0x00  | R/W | MDU Switch Control                                     | 0x0000_0000 |
| <b>MDUSSTS</b>  | MDUG_BA+0x04  | R/W | MDU Switch Status                                      | 0x0000_0000 |

### 15.6 Register Description

#### Phase A/B/C Current Register (Ia / Ib / Ic)

| Register | Offset       | R/W | Description  | Reset Value |
|----------|--------------|-----|--|-------------|
| Ia       | MDUx_BA+0x00 | R/W | Phase A Current Register, Represented in Q-15 Format | 0x0000_0000 |
| Ib       | MDUx_BA+0x04 | R/W | Phase B Current Register, Represented in Q-15 Format | 0x0000_0000 |
| Ic       | MDUx_BA+0x08 | R/W | Phase C Current Register, Represented in Q-15 Format | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | DATA        | <b>Motor Phase Current Data Register</b><br>This register stores the phase current data represented in Q-15 format.<br>The MCU must update registers Ia, Ib and Ic in the beginning of a FOC process flow. |

### Clarke Transformation Output Register (Ialfa / Ibeta)

| Register     | Offset       | R/W | Description   | Reset Value |
|--------------|--------------|-----|---|-------------|
| <b>Ialfa</b> | MDUx_BA+0x0C | R/W | Clarke Transformation Output Register, the Current Along alpha-axis | 0x0000_0000 |
| <b>Ibeta</b> | MDUx_BA+0x10 | R/W | Clarke Transformation Output Register, the Current Along beta-axis  | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | <b>Reserved.</b>   |
| [15:0]  | <b>DATA</b> | <b>Clarke Transformation Output Data Register</b><br>This data content is produced by Clarke Transformation function block and stored in Q-15 format. It is also writable by software. |

### Trigonometric Value Register (SIN / COS)

| Register   | Offset       | R/W | Description   | Reset Value |
|------------|--------------|-----|---|-------------|
| <b>SIN</b> | MDUx_BA+0x14 | R/W | Trigonometric Value of SIN Register, Represented in Q-15 Format | 0x0000_0000 |
| <b>COS</b> | MDUx_BA+0x18 | R/W | Trigonometric Value of COS Register, Represented in Q-15 Format | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | <b>DATA</b> | <b>Trigonometric Value of SIN / COS Data Register</b><br>The trigonometric value of SIN and COS, which is represented in Q-15 format, must be preset before operating Park and inverse Park transformation. |



### Park Transformation Output Register (Id / Iq)

| Register  | Offset       | R/W | Description  | Reset Value |
|-----------|--------------|-----|--|-------------|
| <b>Id</b> | MDUx_BA+0x1C | R/W | Park Transformation Output Register, the Current Along d-axis. | 0x0000_0000 |
| <b>Iq</b> | MDUx_BA+0x20 | R/W | Park Transformation Output Register, the Current Along q-axis. | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | <b>Reserved.</b>   |
| [15:0]  | <b>DATA</b> | <b>Park Transformation Output Data Register</b><br>This data content is produced by Park Transformation function block and stored in Q-15 format. It is also writable by software. |

### PI Controller Command Input Register (Id\_CMD / Iq\_CMD)

| Register      | Offset       | R/W | Description   | Reset Value |
|---------------|--------------|-----|---|-------------|
| <b>Id_CMD</b> | MDUx_BA+0x24 | R/W | PI_d Controller Command Input Register, Represented in Q-15 Format. | 0x0000_0000 |
| <b>Iq_CMD</b> | MDUx_BA+0x28 | R/W | PI_q Controller Command Input Register, Represented in Q-15 Format. | 0x0000_0000 |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -         |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -         |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CMD[15:8] |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CMD [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | <b>CMD</b>  | <b>PI Controller Command Input Register</b><br>The register content stands for the PI controller command input. It must be normalized in Q-15 format. |

**PI Controller Parameter Register (KP<sub>q</sub> / KI<sub>q</sub> / KP<sub>d</sub> / KI<sub>d</sub>)**

| Register        | Offset       | R/W | Description   | Reset Value |
|-----------------|--------------|-----|---|-------------|
| KP <sub>d</sub> | MDUx_BA+0x2C | R/W | Parameter KP for PI <sub>d</sub> Controller, Represented in <b>18-bit I2Q-15 (Q2.15) Format</b> | 0x0000_0000 |
| KI <sub>d</sub> | MDUx_BA+0x30 | R/W | Parameter KI for PI <sub>d</sub> Controller, Represented in <b>18-bit I2Q-15 (Q2.15) Format</b> | 0x0000_0000 |
| KP <sub>q</sub> | MDUx_BA+0x34 | R/W | Parameter KP for PI <sub>q</sub> Controller, Represented in <b>18-bit I2Q-15 (Q2.15) Format</b> | 0x0000_0000 |
| KI <sub>q</sub> | MDUx_BA+0x38 | R/W | Parameter KI for PI <sub>q</sub> Controller, Represented in <b>18-bit I2Q-15 (Q2.15) Format</b> | 0x0000_0000 |

|            |    |    |    |    |    |             |    |
|------------|----|----|----|----|----|-------------|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25          | 24 |
| -          |    |    |    |    |    |             |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17          | 16 |
| -          |    |    |    |    |    | DATA[17:16] |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9           | 8  |
| DATA[15:8] |    |    |    |    |    |             |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1           | 0  |
| DATA [7:0] |    |    |    |    |    |             |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:18] | -           | Reserved.   |
| [17:0]  | DATA        | <b>PI Controller Parameter Register</b><br>The KP <sub>d</sub> and KP <sub>q</sub> stand for the proportional parameter of PI controller, the KI <sub>d</sub> and KI <sub>q</sub> stand for the integral parameter of PI controller. The register data value must be stored in Q-15 format. |

**PI Controller Feed Forward Data Register (PI\_dFW / PI\_qFW)**

| Register | Offset       | R/W | Description                                | Reset Value |
|----------|--------------|-----|--|-------------|
| PI_dFW   | MDUX_BA+0x3C | R/W | PI_d Controller Feed Forward Data Register | 0x0000_0000 |
| PI_qFW   | MDUX_BA+0x40 | R/W | PI_q Controller Feed Forward Data Register | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | DATA        | <b>PI Controller Feed Forward Data Register</b><br>Software presets the feed forward value of the PI controller in this register. |

**Integral Controller Output Data Register (UI\_d / UI\_q)**

| Register | Offset       | R/W | Description                           | Reset Value |
|----------|--------------|-----|---------------------------------------|-------------|
| UI_d     | MDUX_BA+0x44 | R/W | D-axis I Control Output Data Register | 0x0000_0000 |
| UI_q     | MDUX_BA+0x48 | R/W | Q-axis I Control Output Data Register | 0x0000_0000 |

|                         |    |    |                           |    |    |    |    |
|-------------------------|----|----|---------------------------|----|----|----|----|
| 31                      | 30 | 29 | 28                        | 27 | 26 | 25 | 24 |
| -                       |    |    |                           |    |    |    |    |
| 23                      | 22 | 21 | 20                        | 19 | 18 | 17 | 16 |
| -                       |    |    | UI_d[20:16] / UI_q[20:16] |    |    |    |    |
| 15                      | 14 | 13 | 12                        | 11 | 10 | 9  | 8  |
| UI_d[15:8] / UI_q[15:8] |    |    |                           |    |    |    |    |
| 7                       | 6  | 5  | 4                         | 3  | 2  | 1  | 0  |
| UI_d [7:0] / UI_q[7:0]  |    |    |                           |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | DATA        | <p><b>Integral Controller Output Data Register</b></p> <p>The PI function block content I (Integral) controller and P (Propotional) controller, and I controller output data will be used as the next PI control flow. Here is the register to save the output data of I controller and will update when PI controller finished once operation. <a href="#">Here, the register content is a Q-15 format value (PI_Q20_EN = 0) or Q-20 format value(PI_Q20_EN = 1).</a></p> |

**PI Controller Output Register (Vd / Vq)**

| Register | Offset       | R/W | Description  | Reset Value |
|----------|--------------|-----|--|-------------|
| Vd       | MDUx_BA+0x4C | R/W | PI_d Controller Output of d-axis, Represented in Q-15 Format | 0x0000_0000 |
| Vq       | MDUx_BA+0x50 | R/W | PI_q Controller Output of q-axis, Represented in Q-15 Format | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | DATA        | <b>PI Controller Output Data Register (Inverse Park Transformation Input)</b><br>This data content is produced by PI controller function block and stored in Q-15 format. It is also writable by software. |

### Inverse Park Transformation Output Register (Valfa / Vbeta)

| Register     | Offset       | R/W | Description  | Reset Value |
|--------------|--------------|-----|--|-------------|
| <b>Valfa</b> | MDUx_BA+0x54 | R/W | Inverse Park Transformation Output, the Voltage Along the alpha-axis | 0x0000_0000 |
| <b>Vbeta</b> | MDUx_BA+0x58 | R/W | Inverse Park Transformation Output, the Voltage Along the beta-axis  | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | <b>DATA</b> | <b>Inverse Park Transformation Output Data Register</b><br>This data content is produced by inverse Park Transformation function block and stored in Q-15 format. It is also writable by software. |

### Inverse Clarke Transformation Output Register (Valfa / Vbeta)

| Register | Offset       | R/W | Description                                   | Reset Value |
|----------|--------------|-----|---|-------------|
| Vref1    | MDUx_BA+0x5C | R/W | Inverse Clarke Transformation Output Register | 0x0000_0000 |
| Vref2    | MDUx_BA+0x60 | R/W | Inverse Clarke Transformation Output Register | 0x0000_0000 |
| Vref3    | MDUx_BA+0x64 | R/W | Inverse Clarke Transformation Output Register | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | DATA        | <b>Inverse Clarke Transformation Output Data Register</b><br>This data content is produced by inverse Clarke Transformation function block and stored in Q-15 format. It is also writable by software. |



**SVPWM Output Data Register (SVPWM0 / SVPWM2 / SVPWM4)**

| Register        | Offset       | R/W | Description   | Reset Value |
|-----------------|--------------|-----|---|-------------|
| <b>SVPDATA0</b> | MDUx_BA+0x68 | R/W | SVPWM output Data Register (it is an unsigned 16-bit value and denotes Taon time) | 0x0000_0000 |
| <b>SVPDATA2</b> | MDUx_BA+0x6C | R/W | SVPWM output Data Register (it is an unsigned 16-bit value and denotes Tbon time) | 0x0000_0000 |
| <b>SVPDATA4</b> | MDUx_BA+0x70 | R/W | SVPWM output Data Register (it is an unsigned 16-bit value and denotes Tcon time) | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description   |
|---------|---|
| [31:16] | - Reserved.   |
| [15:0]  | <b>DATA</b><br><b>SVPWM Output Data Register</b><br>This data content is produced by SVPWM function block and stored in Q-15 format. It is also writable by software. |

**Normalized SVPWM Output Register (SVPWM0N / SVPWM2N / SVPWM4N)**

| Register         | Offset       | R/W | Description   | Reset Value |
|------------------|--------------|-----|---|-------------|
| <b>SVPDATA0N</b> | MDUX_BA+0x74 | R/W | An unsigned 16-bit data Normalizes SVPDATA0 to PWMNORM and within the limit between PWMMAX and PWMMIN | 0x0000_0000 |
| <b>SVPDATA2N</b> | MDUX_BA+0x78 | R/W | An unsigned 16-bit data Normalizes SVPDATA2 to PWMNORM and within the limit between PWMMAX and PWMMIN | 0x0000_0000 |
| <b>SVPDATA4N</b> | MDUX_BA+0x7C | R/W | An unsigned 16-bit data Normalizes SVPDATA4 to PWMNORM and within the limit between PWMMAX and PWMMIN | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | <b>DATA</b> | <b>Normalized SVPDATA Register</b><br>The SVPWM function block will normalize the value of SVP0/2/4 to the range between 0000h and the value of PWMNORM and limit the PWM duty value within PWMMAX and PWMMIN. If SVPWM function block auto mode is active, each new SVPDATA0N/2N/4N will be reload to PWM unit 0 duty registers PWM0/2/4 after SVPWM operation is completed if PWMRUN and LOAD bits are enabled in PWM unit 0. <b>Here, the register content is a 16-bit unsigned integer.</b> |

### Motor Driver Unit Control Register (MDUCON)

| Register      | Offset       | R/W | Description                        | Reset Value |
|---------------|--------------|-----|------------------------------------|-------------|
| <b>MDUCON</b> | MDUX_BA+0x80 | R/W | Motor Driver Unit Control Register | 0x0000_0000 |

| 31            | 30               | 29               | 28               | 27             | 26             | 25               | 24             |
|---------------|------------------|------------------|------------------|----------------|----------------|------------------|----------------|
| <b>MDU_IE</b> | <b>SVPWM_IE</b>  | <b>INVCK_IE</b>  | <b>INVPK_IE</b>  | <b>PIQ_IE</b>  | <b>PID_IE</b>  | <b>PK_IE</b>     | <b>CK_IE</b>   |
| 23            | 22               | 21               | 20               | 19             | 18             | 17               | 16             |
| --            |                  |                  | <b>PI_Q20_EN</b> |                |                | <b>SVPWM_DIS</b> | <b>ATCLRFG</b> |
| 15            | 14               | 13               | 12               | 11             | 10             | 9                | 8              |
| -             | <b>SVPWMAUTO</b> | <b>INVCKAUTO</b> | <b>INVPKAUTO</b> | <b>PIQAUTO</b> | <b>PIDAUTO</b> | <b>PKAUTO</b>    | -              |
| 7             | 6                | 5                | 4                | 3              | 2              | 1                | 0              |
| -             | <b>SVPWMSTR</b>  | <b>INVCKSTR</b>  | <b>INVPKSTR</b>  | <b>PIQSTR</b>  | <b>PIDSTR</b>  | <b>PKSTR</b>     | <b>CKSTR</b>   |

| Bits | Description     |   |
|------|-----------------|---|
| [31] | <b>MDU_IE</b>   | <b>MDU Interrupt Enable Bit</b><br>1 = MDU interrupt Enabled.<br>0 = MDU interrupt Disabled.  |
| [30] | <b>SVPWM_IE</b> | <b>Clark Block Interrupt Enable Bit</b><br>1 = SVPWM block interrupt Enabled. Interrupt will generate if MDUINT_EN and SVPWM_IE is set to 1 and SVPWMCPPF is set<br>0 = SVPWM block interrupt Disabled. |
| [29] | <b>INVCK_IE</b> | <b>Inv-Clark Block Interrupt Enable Bit</b><br>1 = Inv-Clark block interrupt Enabled.<br>0 = Inv-Clark block interrupt Disabled.  |
| [28] | <b>INVPK_IE</b> | <b>Inv-Park Block Interrupt Enable Bit</b><br>1 = Inv-Park block interrupt Enabled.<br>0 = Inv-Park block interrupt Disabled.   |
| [27] | <b>PIQ_IE</b>   | <b>Q-axis PI Block Interrupt Enable Bit</b><br>1 = Q-axis PI controller block interrupt Enabled.<br>0 = Q-axis PI controller block interrupt Disabled.  |
| [26] | <b>PID_IE</b>   | <b>D-axis PI Block Interrupt Enable Bit</b><br>1 = D-axis PI controller block interrupt Enabled.<br>0 = D-axis PI controller block interrupt Disabled.  |
| [25] | <b>PK_IE</b>    | <b>Park Block Interrupt Enable Bit</b><br>1 = Park block interrupt Enabled.<br>0 = Park block interrupt Disabled.   |

| Bits    | Description |  |
|---------|-------------|--|
| [24]    | CK_IE       | <b>Clark Block Interrupt Enable Bit</b><br>1 = Clark block interrupt Enabled.<br>0 = Clark block interrupt Disabled.   |
| [23:21] | -           | <b>Reserved.</b>   |
| [20]    | PI_Q20_EN   | <b>PI Controller Q20 format Enable Bit</b><br>1 = PI controller Q20 format Enabled.<br>0 = PI controller Q20 format Disabled.  |
| [19:18] | -           | <b>Reserved.</b>   |
| [17]    | SVPWM_DIS   | <b>SVPWM Block Function Disable Bit</b><br>1 = SVPWM operation Disabled.<br>0 = SVPWM function block produces the Space Vector PWM timing for PWM duty registers.  |
| [16]    | ATCLRFG     | <b>Auto-Clear Flag Control Bit</b><br>When this bit is set to high, the complete flag of <b>each function block in auto-mode</b> is cleared automatically after SVPWM function block has updated PWM duty registers. The auto-clear flag function helps the FOC control flow runs smoothly without the need of software clearing flags.<br>1 = Auto-clear flag function Enabled.<br>0 = Auto-clear flag function Disabled. |
| [15]    | -           | <b>Reserved.</b>   |
| [14]    | SVPWMAUTO   | <b>SVPWM Auto-mode Enable Bit</b><br>Setting this bit will enable auto-mode of the function block. If the auto-mode is active, the SVPWM function block is allowed to accept the start trigger from Inverse Clarke block and SVPWM function proceeds completed, it will automatically update the duty registers in PWM unit0 and set the reload bit (LOAD).<br>1 = Auto-mode Enabled.<br>0 = Auto-mode Disabled.           |
| [13]    | INVCKAUTO   | <b>Inverse Clarke Transformation Auto-mode Enable Bit</b><br>Setting this bit will enable auto-mode of the function block. If the auto-mode is active, the current function block is allowed to accept the previous start trigger and will trigger the start bit of the next function block after current function is completed.<br>1 = Auto-mode Enabled.<br>0 = Auto-mode Disabled.                                      |
| [12]    | INVPKAUTO   | <b>Inverse Park Transformation Auto-mode Enable Bit</b><br>Setting this bit will enable auto-mode of the function block. If the auto-mode is active, the current function block is allowed to accept the previous start trigger and will trigger the start bit of the next function block after current function is completed.<br>1 = Auto-mode Enabled.<br>0 = Auto-mode Disabled.  |

| Bits  | Description |  |
|-------|-------------|--|
| [11]  | PIQAUTO     | <b>D-axis PI Controller Auto-mode Enable Bit</b><br>Setting this bit will enable auto-mode of the function block. If the auto-mode is active, the current function block is allowed to accept the previous start trigger and will trigger the start bit of the next function block after current function is completed.<br>1 = Auto-mode Enabled.<br>0 = Auto-mode Disabled. |
| [10]  | PIDAUTO     | <b>D-axis PI Controller Auto-mode Enable Bit</b><br>Setting this bit will enable auto-mode of the function block. If the auto-mode is active, the current function block is allowed to accept the previous start trigger and will trigger the start bit of the next function block after current function is completed.<br>1 = Auto-mode Enabled.<br>0 = Auto-mode Disabled. |
| [9]   | PKAUTO      | <b>Park Transformation Auto-mode Enable Bit</b><br>Setting this bit will enable auto-mode of the function block. If the auto-mode is active, the current function block is allowed to accept the previous start trigger and will trigger the start bit of the next function block after current function is completed.<br>1 = Auto-mode Enabled.<br>0 = Auto-mode Disabled.  |
| [7:8] | -           | <b>Reserved.</b>   |
| [6]   | SVPWMSTR    | <b>SVPWM Start Bit (Write Only, Read 0)</b><br>Setting this bit to high will start proceeding SVPWM timing calculating and it is automatically cleared when the process is completed by hardware. Writing it to 0 will stop and reset the function block.<br>1 = Start the process<br>0 = No action  |
| [5]   | INVCKSTR    | <b>Inverse Clarke Transformation Start Bit (Write Only, Read 0)</b><br>Setting this bit to high will start proceeding Inverse Clarke Transformation and it is automatically cleared when the transform process is completed by hardware. Writing it to 0 will stop and reset the function block.<br>1 = Start the process<br>0 = No action                                   |
| [4]   | INVPKSTR    | <b>Inverse Park Transformation Start Bit (Write Only, Read 0)</b><br>Setting this bit to high will start proceeding Inverse Park Transformation and it is automatically cleared when the transform process is completed by hardware. Writing it to 0 will stop and reset the function block.<br>1 = Start the process<br>0 = No action                                       |
| [3]   | PIQSTR      | <b>Q-axis PI Controller Start Bit (Write Only, Read 0)</b><br>Setting this bit to high will start proceeding Q-axis PI Control and it is automatically cleared when the process is completed by hardware. Writing it to 0 will stop and reset the function block.<br>1 = Start the process<br>0 = No action  |

| Bits | Description   |  |
|------|---------------|--|
| [2]  | <b>PIDSTR</b> | <b>D-axis PI Controller Start Bit (Write Only, Read 0)</b><br>Setting this bit to high will start proceeding D-axis PI Control and it is automatically cleared when the process is completed by hardware. Writing it to 0 will stop and reset the function block.<br>1 = Start the process<br>0 = No action                |
| [1]  | <b>PKSTR</b>  | <b>Park Transformation Start Bit (Write Only, Read 0)</b><br>Setting this bit to high will start proceeding Park Transformation and it is automatically cleared when the transform process is completed by hardware. Writing it to 0 will stop and reset the function block.<br>1 = Start the process<br>0 = No action     |
| [0]  | <b>CKSTR</b>  | <b>Clarke Transformation Start Bit (Write Only, Read 0)</b><br>Setting this bit to high will start proceeding Clarke Transformation and it is automatically cleared when the transform process is completed by hardware. Writing it to 0 will stop and reset the function block.<br>1 = Start the process<br>0 = No action |

**Motor Driver Unit Status Register (MDUSTS)**

| Register | Offset       | R/W | Description                       | Reset Value |
|----------|--------------|-----|-----------------------------------|-------------|
| MDUSTS   | MDUx_BA+0x84 | R/W | Motor Driver Unit Status Register | 0x0000_0700 |

|    |          |          |          |         |         |       |       |
|----|----------|----------|----------|---------|---------|-------|-------|
| 31 | 30       | 29       | 28       | 27      | 26      | 25    | 24    |
| -  |          |          |          |         |         |       |       |
| 23 | 22       | 21       | 20       | 19      | 18      | 17    | 16    |
| -  |          |          |          |         |         |       |       |
| 15 | 14       | 13       | 12       | 11      | 10      | 9     | 8     |
| -  |          |          |          |         | ZONE    |       |       |
| 7  | 6        | 5        | 4        | 3       | 2       | 1     | 0     |
| -  | SVPWMCPF | INVCKCPF | INVPKCPF | PI_dCPF | PI_qCPF | PKCPF | CKCPF |

| Bits    | Description |   |
|---------|-------------|---|
| [31:11] | -           | Reserved.   |
| [10:8]  | ZONE        | <b>Zone Number Indicator</b><br>A 360 degree space is equally divided into six zones where each zone is with 60 degree space. The SVPWM function block will internally produce the zone number during the operation which indicates which zone the motor electric angle stays in the moment.  |
| [7]     | -           | Reserved.   |
| [6]     | SVPWMCPF    | <b>SVPWM Complete Flag</b><br>When the function block completes the process, the corresponding complete flag will be set by hardware. This flag can be cleared by a writing one to itself.<br><b>Note:</b> If both the control bits ATCLRFG (Auto-clear flag control bit) and SVPWMAUTO (Auto-mode enable bit) are set to high, the complete flag is cleared automatically after SVPWM function block has set PWM LOAD bit to high.                         |
| [5]     | INVCKCPF    | <b>Inverse Clarke Transformation Complete Flag</b><br>When the function block completes the process, the corresponding complete flag will be set by hardware. This flag can be cleared by a writing one to itself.<br><b>Note:</b> If both the control bits ATCLRFG (Auto-clear flag control bit) and INVCKAUTO (Auto-mode enable bit) are set to high, the complete flag is cleared automatically after SVPWM function block has set PWM LOAD bit to high. |
| [4]     | INVPKCPF    | <b>Inverse Park Transformation Complete Flag</b><br>When the function block completes the process, the corresponding complete flag will be set by hardware. This flag can be cleared by a writing one to itself.<br><b>Note:</b> If both the control bits ATCLRFG (Auto-clear flag control bit) and INVPKAUTO (Auto-mode enable bit) are set to high, the complete flag is cleared automatically after SVPWM function block has set PWM LOAD bit to high.   |

| Bits | Description    |   |
|------|----------------|---|
| [3]  | <b>PI_dCPF</b> | <p><b>PI_d Controller Complete Flag</b></p> <p>When the function block completes the process, the corresponding complete flag will be set by hardware. This flag can be cleared by a writing one to itself.</p> <p><b>Note:</b> If both the control bits ATCLRFG (Auto-clear flag control bit) and PI_dAUTO (Auto-mode enable bit) are set to high, the complete flag is cleared automatically after SVPWM function block has set PWM LOAD bit to high.</p>     |
| [2]  | <b>PI_qCPF</b> | <p><b>PI_q Controller Complete Flag</b></p> <p>When the function block completes the process, the corresponding complete flag will be set by hardware. This flag can be cleared by a writing one to itself.</p> <p><b>Note:</b> If both the control bits ATCLRFG (Auto-clear flag control bit) and PI_qAUTO (Auto-mode enable bit) are set to high, the complete flag is cleared automatically after SVPWM function block has set PWM LOAD bit to high.</p>     |
| [1]  | <b>PKCPF</b>   | <p><b>Park Transformation Complete Flag</b></p> <p>When the function block completes the process, the corresponding complete flag will be set by hardware. This flag can be cleared by a writing one to itself.</p> <p><b>Note:</b> If both the control bits ATCLRFG (Auto-clear flag control bit) and PKAUTO (Auto-mode enable bit) are set to high, the complete flag is cleared automatically after SVPWM function block has set PWM LOAD bit to high.</p>   |
| [0]  | <b>CKCPF</b>   | <p><b>Clarke Transformation Complete Flag</b></p> <p>When the function block completes the process, the corresponding complete flag will be set by hardware. This flag can be cleared by a writing one to itself.</p> <p><b>Note:</b> If both the control bits ATCLRFG (Auto-clear flag control bit) and CKAUTO (Auto-mode enable bit) are set to high, the complete flag is cleared automatically after SVPWM function block has set PWM LOAD bit to high.</p> |



### Limitation of PI Controller Output (PI\_dMAX / PI\_qMAX)

| Register | Offset        | R/W | Description   | Reset Value |
|----------|---------------|-----|---|-------------|
| PI_dMAX  | MDUX_BA+0x100 | R/W | Maximum limit value of PI_d Controller in Q-15 Format | 0x0000_7FFF |
| PI_qMAX  | MDUX_BA+0x108 | R/W | Maximum limit value of PI_q Controller in Q-15 Format | 0x0000_7FFF |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -         |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -         |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| MAX[15:8] |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| MAX[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | MAX         | <b>Maximum Limit of PI Controller and I Controller Output</b><br>The 16-bit value represented in Q-15 format limits the maximum value of the PI controller and I controller output. |

### Limitation of PI Controller Output (PI\_dMIN / PI\_qMIN)

| Register | Offset        | R/W | Description   | Reset Value |
|----------|---------------|-----|---|-------------|
| PI_dMIN  | MDUX_BA+0x104 | R/W | Minimum limit value of PI_d Controller in Q-15 Format | 0x0000_8000 |
| PI_qMIN  | MDUX_BA+0x10C | R/W | Minimum limit value of PI_q Controller in Q-15 Format | 0x0000_8000 |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -         |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -         |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| MIN[15:8] |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| MIN[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | MIN         | <b>Minimum Limit of PI Controller and I Controller Output</b><br>The 16-bit value represented in Q-15 format limits the minimum value of the PI controller and I controller output. |

**Limitation of Integral Controller Output (I\_dMAX / I\_qMAX)**

| Register | Offset        | R/W | Description   | Reset Value |
|----------|---------------|-----|---|-------------|
| I_dMAX   | MDUx_BA+0x110 | R/W | Maximum limit value of integral_d Controller in Q-15 Format | 0x0000_7FFF |
| I_qMAX   | MDUx_BA+0x118 | R/W | Maximum limit value of integral_q Controller in Q-15 Format | 0x0000_7FFF |

|           |    |    |            |    |    |    |    |
|-----------|----|----|------------|----|----|----|----|
| 31        | 30 | 29 | 28         | 27 | 26 | 25 | 24 |
| -         |    |    |            |    |    |    |    |
| 23        | 22 | 21 | 20         | 19 | 18 | 17 | 16 |
| -         |    |    | MAX[20:16] |    |    |    |    |
| 15        | 14 | 13 | 12         | 11 | 10 | 9  | 8  |
| MAX[15:8] |    |    |            |    |    |    |    |
| 7         | 6  | 5  | 4          | 3  | 2  | 1  | 0  |
| MAX[7:0]  |    |    |            |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | MAX         | <b>Maximum Limit of PI Controller and I Controller Output</b><br>The 16-bit value represented in Q-15 format limits the maximum value of the integral controller output. <b>Here, the register content is a Q-15 format value (PI_Q20_EN = 0) or Q-20 format value(PI_Q20_EN = 1).</b> |

**Limitation of Integral Controller Output (I\_dMIN / I\_qMIN)**

| Register | Offset        | R/W | Description   | Reset Value |
|----------|---------------|-----|---|-------------|
| I_dMIN   | MDUx_BA+0x114 | R/W | Minimum limit value of integral_d Controller in Q-15 Format | 0x0000_8000 |
| I_qMIN   | MDUx_BA+0x11C | R/W | Minimum limit value of integral_q Controller in Q-15 Format | 0x0000_8000 |

|           |    |    |            |    |    |    |    |
|-----------|----|----|------------|----|----|----|----|
| 31        | 30 | 29 | 28         | 27 | 26 | 25 | 24 |
| -         |    |    |            |    |    |    |    |
| 23        | 22 | 21 | 20         | 19 | 18 | 17 | 16 |
| -         |    |    | MIN[20:16] |    |    |    |    |
| 15        | 14 | 13 | 12         | 11 | 10 | 9  | 8  |
| MIN[15:8] |    |    |            |    |    |    |    |
| 7         | 6  | 5  | 4          | 3  | 2  | 1  | 0  |
| MIN[7:0]  |    |    |            |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | MIN         | <b>Minimum Limit of PI Controller and I Controller Output</b><br>The 16-bit value represented in Q-15 format limits the minimum value of the integral controller output. |

### Inverse Park Transformation Input Register (Vd\_In / Vq\_In)

| Register | Offset        | R/W | Description  | Reset Value |
|----------|---------------|-----|--|-------------|
| Vd_In    | MDUX_BA+0x120 | R   | Inverse Park Transformation Input of d_axis represented in Q-15 Format | 0x0000_0000 |
| Vq_In    | MDUX_BA+0x124 | R   | Inverse Park Transformation Input of q_axis represented in Q-15 Format | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DATA[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DATA [7:0] |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | DATA        | <b>Inverse Park Transformation Input Data (Read Only)</b><br>This data content is produced by the PI controller with maximum and minimum limit operation and stored in Q-15 format. It is a read only register |

### PWM Normalization Data Register (PWMNORM)

| Register | Offset        | R/W | Description                                | Reset Value |
|----------|---------------|-----|--|-------------|
| PWMNORM  | MDUX_BA+0x128 | R/W | An Unsigned 16-bit PWM Normalization Value | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -             |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -             |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PWMNORM[15:8] |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PWMNORM[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:0]  | PWMNORM     | <b>PWM Normalization Data</b><br>The SVPWM function block will normalize the value of SVP0/2/4 to the range between 0000h and the value of PWMNORM and limit the PWM duty value within PWMMAX and PWMMIN. <a href="#">Here, the register content is a 16-bit unsigned integer.</a> |

**Maximum Limit of SVPWM Output (PWMMAX)**

| Register | Offset        | R/W | Description  | Reset Value |
|----------|---------------|-----|--|-------------|
| PWMMAX   | MDUX_BA+0x12C | R/W | An unsigned 16-bit Maximum Limit Value of SVPWM Output | 0x0000_FFFF |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -            |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -            |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PWMMAX[15:8] |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PWMMAX[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | PWMMAX      | <b>Maximum Limit of SVPWM Output Data Register</b><br>The SVPWM function block will normalize the value of SVP0/2/4 to the range between 0000h and the value of PWMNORM and limit the PWM duty value within PWMMAX and PWMMIN. <a href="#">Here, the register content is a 16-bit unsigned integer.</a> |

### Minimum Limit of SVPWM Output (PWMMIN)

| Register | Offset        | R/W | Description  | Reset Value |
|----------|---------------|-----|--|-------------|
| PWMMIN   | MDUX_BA+0x130 | R/W | An unsigned 16-bit Minimum Limit Value of SVPWM Output | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -            |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -            |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| PWMMIN[15:8] |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| PWMMIN[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | PWMMIN      | <b>Maximum Limit of SVPWM Output Data Register</b><br>The SVPWM function block will normalize the value of SVP0/2/4 to the range between 0000h and the value of PWMNORM and limit the PWM duty value within PWMMAX and PWMMIN. <a href="#">Here, the register content is a 16-bit unsigned integer.</a> |



### MDU Switch Control (MDUSCON)

| Register       | Offset       | R/W | Description        | Reset Value |
|----------------|--------------|-----|--------------------|-------------|
| <b>MDUSCON</b> | MDUG_BA+0x00 | R/W | MDU Switch Control | 0x0000_0000 |

|    |    |    |    |    |    |    |             |
|----|----|----|----|----|----|----|-------------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24          |
| -  |    |    |    |    |    |    |             |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16          |
| -  |    |    |    |    |    |    |             |
| 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8           |
| -  |    |    |    |    |    |    |             |
| 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0           |
| -  |    |    |    |    |    |    | <b>MDUS</b> |

| Bits   | Description |  |
|--------|-------------|--|
| [31:1] | -           | Reserved.  |
| [0]    | <b>MDUS</b> | <b>MDU Switch Control Register</b><br>1 = MDU is controlled by MDU1 control register<br>0 = MDU is controlled by MDU0 control register |

### MDU Switch Control (MDUSSTS)

| Register       | Offset       | R/W | Description       | Reset Value |
|----------------|--------------|-----|-------------------|-------------|
| <b>MDUSSTS</b> | MDUG_BA+0x04 | R/W | MDU Switch Status | 0x0000_0000 |

|    |    |    |    |    |        |          |          |
|----|----|----|----|----|--------|----------|----------|
| 31 | 30 | 29 | 28 | 27 | 26     | 25       | 24       |
| -  |    |    |    |    |        |          |          |
| 23 | 22 | 21 | 20 | 19 | 18     | 17       | 16       |
| -  |    |    |    |    |        |          |          |
| 15 | 14 | 13 | 12 | 11 | 10     | 9        | 8        |
| -  |    |    |    |    |        |          |          |
| 7  | 6  | 5  | 4  | 3  | 2      | 1        | 0        |
| -  |    |    |    |    | MDUACT | MDU1BUSY | MDU0BUSY |

| Bits   | Description     |   |
|--------|-----------------|---|
| [31:3] | -               | Reserved.   |
| [2]    | <b>MDUACT</b>   | <b>MDU Active Set</b><br>1 = MDU1 is active<br>0 = MDU0 is active |
| [1]    | <b>MDU1BUSY</b> | <b>MDU1 Busy Register</b><br>1 = MDU1 is busy<br>0 = MDU1 is idle |
| [0]    | <b>MDU0BUSY</b> | <b>MDU0 Busy Register</b><br>1 = MDU0 is busy<br>0 = MDU0 is idle |

## 16 HARDWARE DIVIDER

### 16.1 Overview

The hardware divider is useful to the high performance application. The hardware divider is a signed, integer divider with both quotient and remainder outputs.

### 16.2 Features

- Signed (two's complement) integer calculation.
- 32-bit dividend with 16-bit divisor calculation capacity.
- Both 32-bit quotient and 16-bit remainder outputs.
- Divided by 0 warning flag.
- 7 HCLK clocks taken for one cycle calculation.
- Software triggered with finish flag.
- Triggered by MDU under background automatically.

### 16.3 Register Map

R: read only, W: write only, R/W: both read and write

| Register                                  | Offset      | R/W | Description                              | Reset Value |
|---|-------------|-----|--|-------------|
| DIV Base Address:<br>DIV_BA = 0x5001_4000 |             |     |  |             |
| <b>DIVCON</b>                             | DIV_BA+0x00 | W   | Divider Control Register                 | 0x0000_0000 |
| <b>DIVIDEND</b>                           | DIV_BA+0x04 | R/W | Dividend Source Register (Signed 32-bit) | 0x0000_0000 |
| <b>DIVISOR</b>                            | DIV_BA+0x08 | R/W | Divisor Source Register (Signed 16-bit)  | 0x0000_FFFF |
| <b>DIVQUO</b>                             | DIV_BA+0x0C | R   | Quotient Result Register (Signed 32-bit) | 0x0000_0000 |
| <b>DIVREM</b>                             | DIV_BA+0x10 | R   | Reminder Result Register (Signed 16-bit) | 0x0000_0000 |
| <b>DIVSTS</b>                             | DIV_BA+0x14 | R/W | Divider Status Register                  | 0x0000_0001 |

## 16.4 Register Description

### Divider Control Register (DIVCON)

| Register | Offset      | R/W | Description              | Reset Value |
|----------|-------------|-----|--------------------------|-------------|
| DIVCON   | DIV_BA+0x00 | W   | Divider Control Register | 0x0000_0000 |

|    |    |    |    |    |    |    |       |
|----|----|----|----|----|----|----|-------|
| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24    |
| -  |    |    |    |    |    |    |       |
| 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16    |
| -  |    |    |    |    |    |    |       |
| 15 | 14 | 13 | 12 | 11 | 10 | 9  | 8     |
| -  |    |    |    |    |    |    |       |
| 7  | 6  | 5  | 4  | 3  | 2  | 1  | 0     |
| -  |    |    |    |    |    |    | DIVST |

| Bits   | Description  |
|--------|--|
| [31:1] | Reserved.  |
| [0]    | <b>DIVST</b><br><b>Divider Start.</b><br>Set this to 1 will start trigger divider operation once. This bit will be auto clear by hardware will calculation complete. |

### Divider Dividend Source Register (DIVIDEND)

| Register        | Offset      | R/W | Description                              | Reset Value |
|-----------------|-------------|-----|--|-------------|
| <b>DIVIDEND</b> | DIV_BA+0x04 | R/W | Dividend Source Register (Signed 32-bit) | 0x0000_0000 |

|                 |    |    |    |    |    |    |    |
|-----------------|----|----|----|----|----|----|----|
| 31              | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Dividend[31:24] |    |    |    |    |    |    |    |
| 23              | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Dividend[23:16] |    |    |    |    |    |    |    |
| 15              | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Dividend[15:8]  |    |    |    |    |    |    |    |
| 7               | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Dividend[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description  |
|--------|--|
| [31:0] | <div>Dividend</div> <div>Dividend Source.</div> <div>This register is given the dividend (signed 32-bit) of divider before calculation starts.</div> |

**Divider Divisor Source Register (DIVISOR)**

| Register       | Offset      | R/W | Description                             | Reset Value |
|----------------|-------------|-----|---|-------------|
| <b>DIVISOR</b> | DIV_BA+0x08 | R/W | Divisor Source Resister (Signed 16-bit) | 0x0000_FFFF |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -             |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -             |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Divisor[15:8] |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Divisor[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description  |
|---------|--|
| [31:16] | - Reserved.  |
| [15:0]  | <b>Divisor Source.</b><br>This register is given the divisor of divider before calculation starts.<br><b>Note:</b> When this register is written, hardware divider will start calculation. |

### Divider Quotient Result Register (DIVQUO)

| Register | Offset      | R/W | Description                              | Reset Value |
|----------|-------------|-----|--|-------------|
| DIVQUO   | DIV_BA+0x0C | R   | Quotient Result Register (Signed 32-bit) | 0x0000_0000 |

|                 |    |    |    |    |    |    |    |
|-----------------|----|----|----|----|----|----|----|
| 31              | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Quotient[31:24] |    |    |    |    |    |    |    |
| 23              | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Quotient[23:16] |    |    |    |    |    |    |    |
| 15              | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Quotient[15:8]  |    |    |    |    |    |    |    |
| 7               | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Quotient[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:0] | Quotient    | <b>Quotient Result</b><br>This register holds the quotient (signed 32-bit) result of divider after calculation completes. |



### Divider Reminder Result Register (DIVREM)

| Register      | Offset      | R/W | Description                              | Reset Value |
|---------------|-------------|-----|--|-------------|
| <b>DIVREM</b> | DIV_BA+0x10 | R   | Reminder Result Register (Signed 16-bit) | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -              |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -              |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reminder[15:8] |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Reminder[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:0]  | Reminder    | <b>Reminder Result</b><br>This register holds the reminder (signed 16-bit) result of divider after calculation completes. |

### Divider Status Register (DIVSTS)

| Register      | Offset      | R/W | Description             | Reset Value |
|---------------|-------------|-----|-------------------------|-------------|
| <b>DIVSTS</b> | DIV_BA+0x14 | R/W | Divider Status Register | 0x0000_0001 |

|    |    |    |    |    |              |             |                   |
|----|----|----|----|----|--------------|-------------|-------------------|
| 31 | 30 | 29 | 28 | 27 | 26           | 25          | 24                |
| -  |    |    |    |    |              |             |                   |
| 23 | 22 | 21 | 20 | 19 | 18           | 17          | 16                |
| -  |    |    |    |    |              |             |                   |
| 15 | 14 | 13 | 12 | 11 | 10           | 9           | 8                 |
| -  |    |    |    |    |              |             |                   |
| 7  | 6  | 5  | 4  | 3  | 2            | 1           | 0                 |
| -  |    |    |    |    | <b>DIVFF</b> | <b>DIV0</b> | <b>DIV_FINISH</b> |

| Bits   | Description       |   |
|--------|-------------------|---|
| [31:3] | -                 | <b>Reserved.</b>  |
| [2]    | <b>DIVFF</b>      | <b>Divider Operation Finish Flag</b><br>When divider calculation has finished, this bit is set to 1. This bit is cleared to 0 by writing 1 to it through software |
| [1]    | <b>DIV0</b>       | <b>Divisor Zero Warning.</b><br>1 = The divisor is 0.<br>0 = The divisor is not 0.<br>This register is read only.   |
| [0]    | <b>DIV_FINISH</b> | <b>Divider Operation Finished</b><br>1 = The divider calculation finished.<br>0 = The divider calculation not finished yet.<br>This register is read only.        |

## 17 ENHANCED INPUT CAPTURE TIMER

### 17.1 Overview

This device provides up to two units of Input Capture Timer/Counter which capture function can detect the digital edge changed signal at channel inputs. Each unit has three input capture channels. The timer/counter is equipped with up counting, reload and compare-match capabilities.

### 17.2 Features

- Up to two Input Capture Timer/Counter Units, Input Capture 0 and Input Capture 1.
- Each unit has own interrupt vector
- 24-bit Input Capture up-counting timer/counter
- With noise filter in front end of input ports
- Edge detector with three options
  - Rising edge detection
  - Falling edge detection
  - Both edge detection
- Each input channel is supported with one capture counter hold register
- Captured event reset/reload capture counter option
- Supports the compare-match function

### 17.3 Input Capture Timer/Counter Architecture

Each of the input capture timer/counter unit supports 3 input channels with three programmable input signal sources. The port pins IC0 to IC2 can be fed to the inputs of capture unit, besides, the QEI controller input signals (CHA, CHB and CHX), analog comparator outputs (CPO), OPA digital output (OPDOx), and ADC compare output (ADCMPOx) can also be internally routed to the capture inputs by software configuration. The following figures illustrate the architecture of Input Capture.

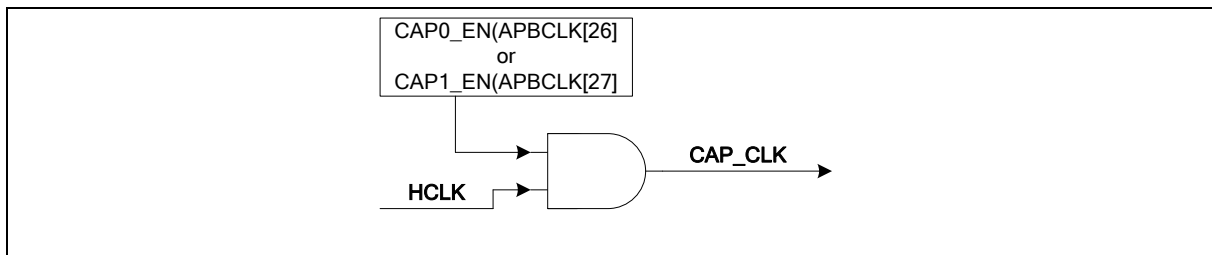


Figure 17-1 Input Capture Timer/Counter Clock Source Control

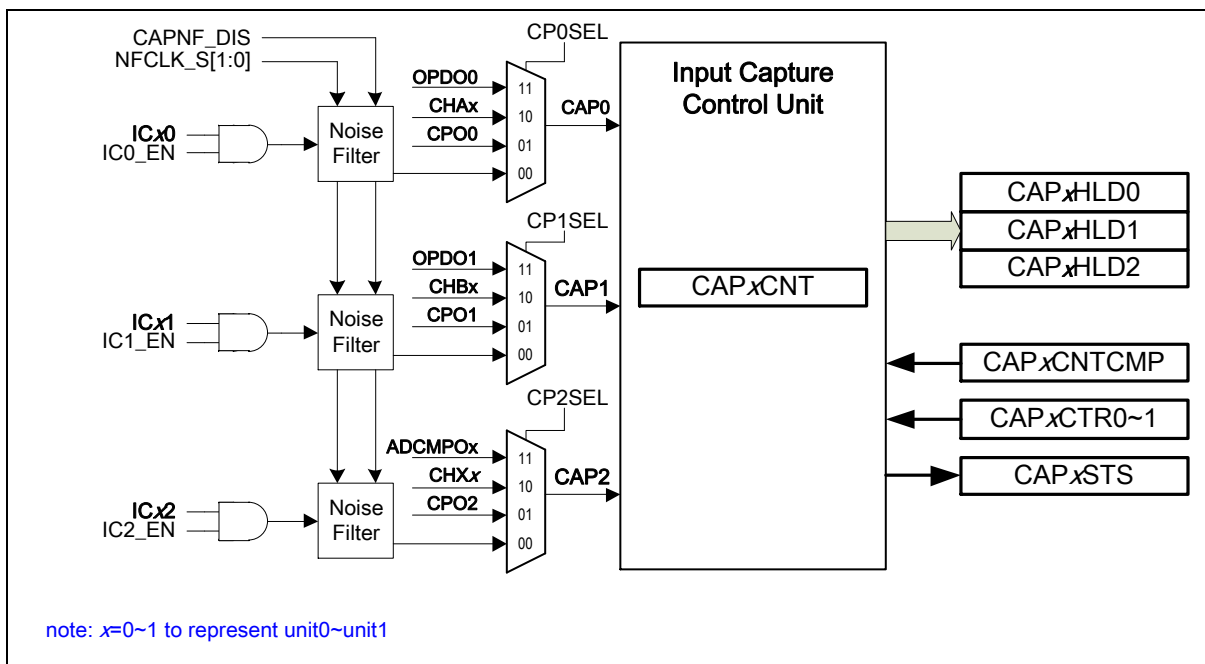


Figure 17-2 Input Capture Timer/Counter Architecture

## 17.4 Input Noise Filter

The architecture of input noise filter is similar to the QEI controller with four sampling rate options. Refer to 18.4 for the detail.

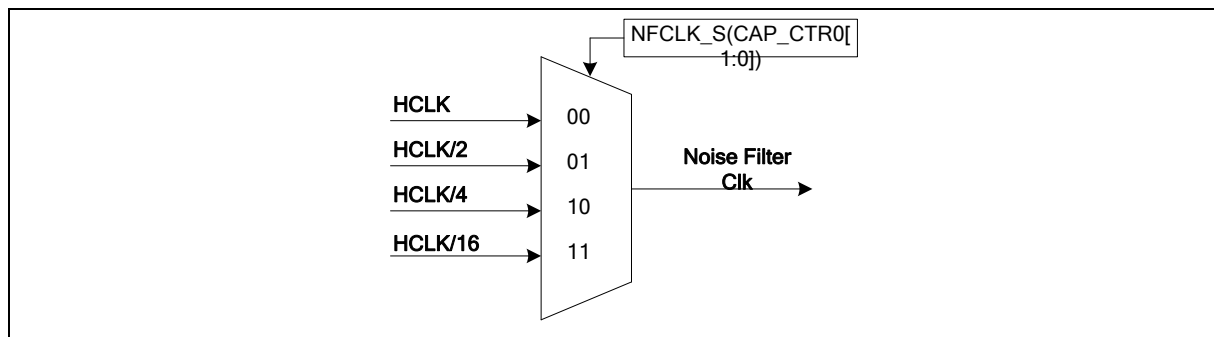


Figure 17–3 Noise Filter Sampling Clock Selection

## 17.5 Operation of Input Capture Timer/Counter

The capture modules are functioned to detect and measure pulse width and period of a square wave. The input channel 0 to 2 have their own edge detector but share with one capture timer/counter i.e. CAP\_CNT. The trigger option is programmable through CAPEDG in CAP\_CTR1 register. It supports positive edge, negative edge and both edge triggers. Each capture module consists of an enable control bit, IC0\_EN to IC2\_EN. The capture counter (CAP\_CNT) serves as a 24-bit up counter. It supports reload and compared modes. The Input Capture Timer/Counter Enable bit (CAP\_EN) must be set to enable Input Capture Timer/Counter functions. More details are described in next sections.

### 17.5.1 Capture Function

Each time the capture input trigger is validated, the content of the free running 24 bits capture counter CAP\_CNT will be captured/transferred into the capture hold registers, CAP\_HLD0~2, depending which channel trigger. This action also causes the CAPTF flag bits in CAP\_STS to be set, which will also generate an interrupt (if enabled by CAPTFx\_IEN bit in CAP\_CTR0 register). The CAPTF0~2 flags are logical “OR” to the interrupt module. Flag is set by hardware and clear by software. Software will have to resolve on the priority of the interrupt flags.

Setting the CPTCLR bit (CAP\_CTR0[26]), will allow hardware to reset capture counter (CAP\_CNT) automatically after the value of CAP\_CNT has been captured. **Priority is given to reset counter after capture the counter value into the capture register.**

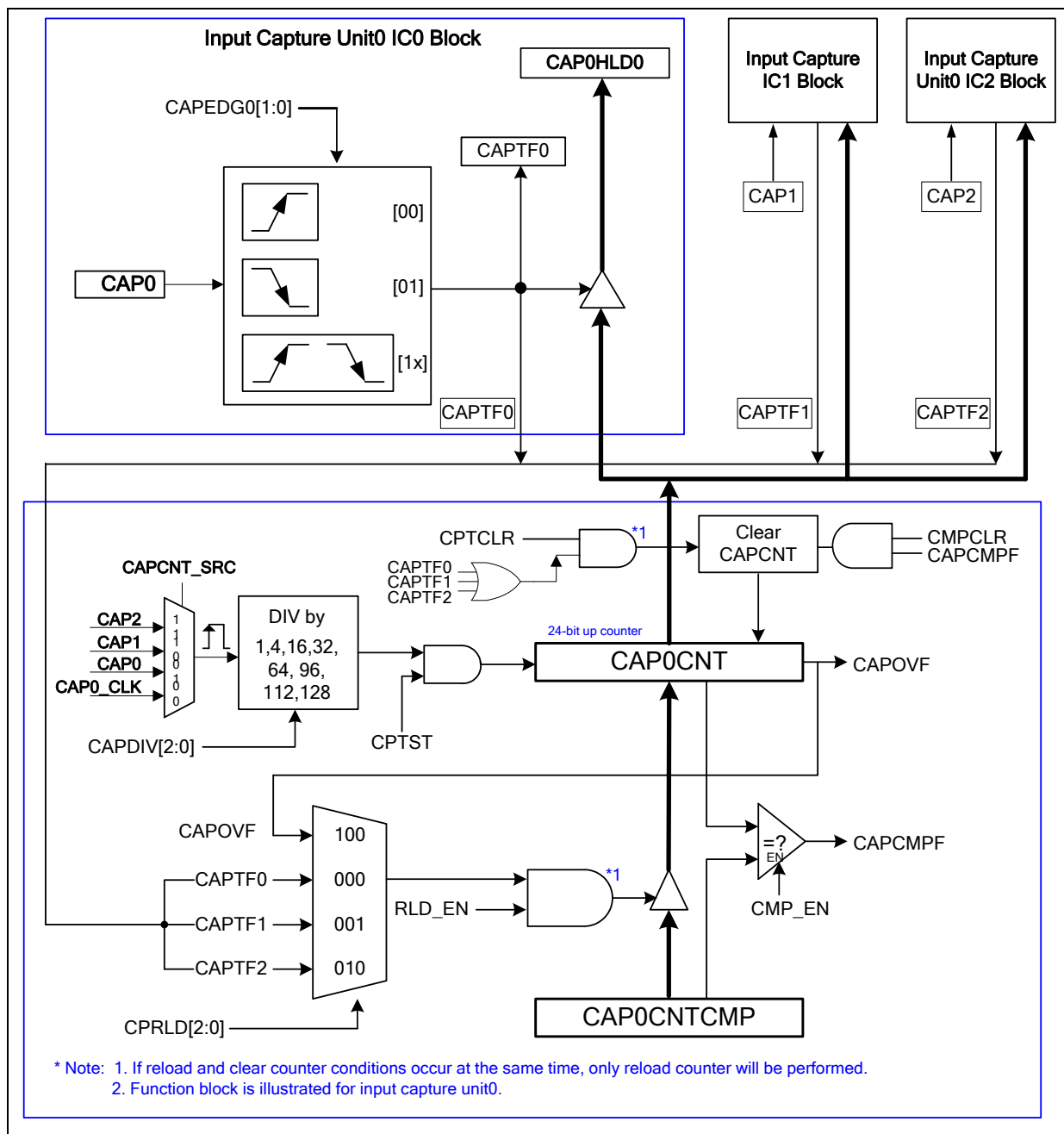


Figure 17-4 Input Capture Timer/Counter Functions Block

### 17.5.2 Compare Mode

The compare function is enabled by setting the CMP\_EN (CAP\_CTR0[28]) bit to 1. CAP\_CNTCMP will serve as a compare register. As CAP\_CNT counting up, upon matching with CAP\_CNTCMP value, CAPCMPF (CAP\_STS[4]) will be set, which will generate an interrupt request if capture compare interrupt enable bit, CAPCMPI\_EN, is set. And then the timer reload from 0 and starts counting again.

Setting the CMPCLR bit (CAP\_CTR[25]), will allow hardware to reset capture counter automatically after a match has occurred.

### 17.5.3 Reload Mode

Input Capture Timer/Counter can also be configured for Reload mode. The reload function is enabled

by setting the RLD\_EN bit (CAP\_CTR0[27]) to 1. In this mode, CAP\_CNTCMP serves as a reload register. When CAP\_CNT overflows, a reload is generated that causes the contents of the CAP\_CNTCMP register to be reloaded into the CAP\_CNT register, if RLD\_EN is set. However, if RLD\_EN = 0, CAP\_CNT will be reload with 0, and count up again.

Alternatively, other reload source is also possible by the capture inputs by configuring the CPRLD (CAP\_CTR1[10:8]). This action also sets the CAPTFx flag bits in the CAP\_STS register.



### 17.6 Input Capture Timer/Counter Interrupt Architecture

There are five interrupt sources for one input capture unit, each one has an interrupt flag and enable control bit, which can trigger Input Capture Timer/Counter Interrupt. **Note that all the interrupt flags are set by hardware and must be cleared by software.**

Figure 17–5 Demonstrates the architecture of Input Capture Timer/Counter interrupts.

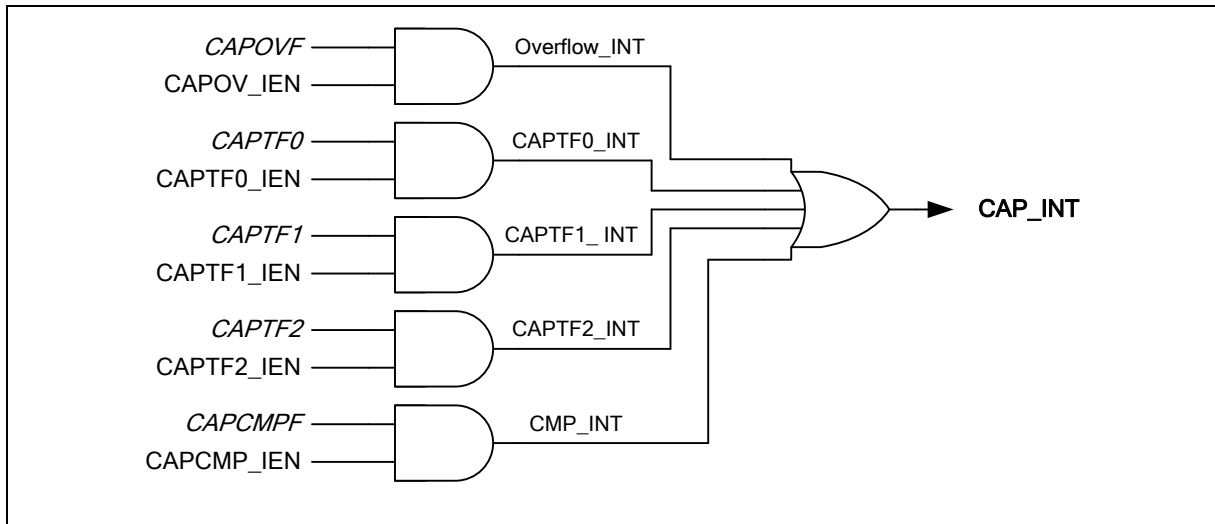


Figure 17–5 Input Capture Timer/Counter Interrupt Architecture Diagram

### 17.7 Register Map

R: read only, W: write only, R/W: both read and write

| Register   | Offset       | R/W | Description                               | Reset Value |
|--|--------------|-----|---|-------------|
| <b>CAP Base Address:</b><br>$\text{CAPx\_BA} = 0x401B\_0000 + (0x4000 * x)$ $x = 0, 1$ |              |     |   |             |
| <b>CAP_CNT</b>   | CAPx_BA+0x00 | R/W | Input Capture Counter (24-bit up counter) | 0x0000_0000 |
| <b>CAP_HLD0</b>  | CAPx_BA+0x04 | R/W | Input Capture Counter Hold Register 0     | 0x0000_0000 |
| <b>CAP_HLD1</b>  | CAPx_BA+0x08 | R/W | Input Capture Counter Hold Register 1     | 0x0000_0000 |
| <b>CAP_HLD2</b>  | CAPx_BA+0x0C | R/W | Input Capture Counter Hold Register 2     | 0x0000_0000 |
| <b>CAP_CNTCMP</b>  | CAPx_BA+0x10 | R/W | Input Capture Counter Compare Register    | 0x0000_0000 |
| <b>CAP_CTR0</b>  | CAPx_BA+0x14 | R/W | Input Capture Control Register 0          | 0x0000_0000 |
| <b>CAP_CTR1</b>  | CAPx_BA+0x18 | R/W | Input Capture Control Register 1          | 0x0000_0000 |
| <b>CAP_STS</b>   | CAPx_BA+0x1C | R/W | Input Capture Status Register             | 0x0000_0000 |

## 17.8 Register Description

### Input Capture Counter (CAP\_CNT)

| Register | Offset       | R/W | Description                               | Reset Value |
|----------|--------------|-----|---|-------------|
| CAP_CNT  | CAPx_BA+0x00 | R/W | Input Capture Counter (24-bit up counter) | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -              |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CAP_CNT[23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CAP_CNT[15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CAP_CNT[7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:24] | -           | Reserved.  |
| [23:0]  | CAP_CNT     | <b>Input Capture Timer/Counter</b><br>The input Capture Timer/Counter is a 24-bit up-counting counter. The clock source for the counter is from the clock divider output which the CAP_CLK is software optionally divided by 1,4,16 or 32. |

### Input Capture Counter Hold Register (CAP\_HLD0~2)

| Register | Offset       | R/W | Description                           | Reset Value |
|----------|--------------|-----|---------------------------------------|-------------|
| CAP_HLD0 | CAPx_BA+0x04 | R/W | Input Capture Counter Hold Register 0 | 0x0000_0000 |
| CAP_HLD1 | CAPx_BA+0x08 | R/W | Input Capture Counter Hold Register 1 | 0x0000_0000 |
| CAP_HLD2 | CAPx_BA+0x0C | R/W | Input Capture Counter Hold Register 2 | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -              |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CAP_HLD[23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CAP_HLD[15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CAP_HLD[7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:24] | -           | Reserved.   |
| [23:0]  | CAP_HLD     | <b>Input Capture Counter Hold Register</b><br>When an active input capture channel detects a valid edge signal change, the CAPCNT value is latched into the corresponding holding register. Each input channel has itself holding register named by CAP_HLDx where x is from 0 to 2 to indicate inputs from IC0 to IC2, respectively. |

**Input Capture Counter Compare Register (CAP\_CNTCMP)**

| Register   | Offset       | R/W | Description                            | Reset Value |
|------------|--------------|-----|--|-------------|
| CAP_CNTCMP | CAPx_BA+0x10 | R/W | Input Capture Counter Compare Register | 0x0000_0000 |

|                   |    |    |    |    |    |    |    |
|-------------------|----|----|----|----|----|----|----|
| 31                | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -                 |    |    |    |    |    |    |    |
| 23                | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| CAP_CNTCMP[23:16] |    |    |    |    |    |    |    |
| 15                | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| CAP_CNTCMP[15:8]  |    |    |    |    |    |    |    |
| 7                 | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| CAP_CNTCMP[7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:24] | -           | Reserved.   |
| [23:0]  | CAP_CNTCMP  | <b>Input Capture Counter Compare Register</b><br>If the compare function is enabled (CMP_EN = 1), the compare register is loaded with the value that the compare function compares the capture counter (CAP_CNT) with.<br>If the reload control is enabled (RLD_EN = 1), an overflow event or capture events will trigger the hardware to reload CAP_CNTCMP into CAP_CNT. |

### Input Capture Timer/Counter Control Register (CAP\_CTR0)

| Register | Offset       | R/W | Description                      | Reset Value |
|----------|--------------|-----|----------------------------------|-------------|
| CAP_CTR0 | CAPx_BA+0x14 | R/W | Input Capture Control Register 0 | 0x0000_0000 |

|    |        |            |           |           |            |            |            |
|----|--------|------------|-----------|-----------|------------|------------|------------|
| 31 | 30     | 29         | 28        | 27        | 26         | 25         | 24         |
| -  | -      | CAP_EN     | CMP_EN    | RLD_EN    | CPTCLR     | CMPCLR     | CPST       |
| 23 | 22     | 21         | 20        | 19        | 18         | 17         | 16         |
| -  | -      | CAPCMP_IEN | CAPOV_IEN | -         | CAPTF2_IEN | CAPTF1_IEN | CAPTF0_IEN |
| 15 | 14     | 13         | 12        | 11        | 10         | 9          | 8          |
| -  | -      | CAPSEL2    |           | CAPSEL1   |            | CAPSEL0    |            |
| 7  | 6      | 5          | 4         | 3         | 2          | 1          | 0          |
| -  | IC2_EN | IC1_EN     | IC0_EN    | CAPNF_DIS | -          | NFCLK_S    |            |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | -           | <b>Reserved.</b>   |
| [29]    | CAP_EN      | <b>Input Capture Timer/Counter Enable Bit</b><br>1 = Input Capture function Enabled.<br>0 = Input Capture function Disabled.   |
| [28]    | CMP_EN      | <b>The Compare Function Enable Bit</b><br>The compare function in input capture timer/counter is to compare the dynamic counting CAP_CNT with the compare register CAP_CNTCMP, if CAP_CNT value reaches CAP_CNTCMP, the flag CAPCMPF will be set.<br>1 = Compare function Enabled<br>0 = Compare function Disabled   |
| [27]    | RLD_EN      | <b>The Reload Function Enable Bit</b><br>Setting this bit to enable reload function. If the reload control is enabled, an overflow event (CAPOVF) or capture events (CAPTFx) will trigger the hardware to reload CAP_CNTCMP into CAP_CNT.<br>1 = Reload function Enabled<br>0 = Reload function Disabled   |
| [26]    | CPTCLR      | <b>Input Capture Counter Clear by Capture Events Control Bit</b><br>If this bit is set to 1, the capture counter (CAP_CNT) will be cleared to 0 when any one of capture events (CAPTF0~3) occurs.<br>1 = Capture events (CAPTF0~3) can clear capture counter (CAP_CNT) Enabled<br>0 = Capture events (CAPTF0~3) can clear capture counter (CAP_CNT) Disabled |

| Bits    | Description       |   |
|---------|-------------------|---|
| [25]    | <b>CMPCLR</b>     | <b>Input Capture Counter Clear by Compare-match Control Bit</b><br>If this bit is set to 1, the capture counter (CAP_CNT) will be cleared to 0 when the compare-match event (CAMCMPF = 1) occurs.<br>1 = Compare-match event (CAMCMPF) can clear capture counter (CAP_CNT) Enabled<br>0 = Compare-match event (CAMCMPF) can clear capture counter (CAP_CNT) Disabled  |
| [24]    | <b>CPTST</b>      | <b>Input Capture Counter Start Bit</b><br>Setting this bit to 1, the capture counter (CAP_CNT) starts up-counting synchronously with capture clock input (CAP_CLK).<br>1 = CAP_CNT starts up-counting<br>0 = CAP_CNT stop counting.   |
| [23:22] | -                 | <b>Reserved.</b>  |
| [21]    | <b>CAPCMP_IEN</b> | <b>Enable CAPCMPF Trigger Input Capture Interrupt</b><br>1 = Enabling flag CAPCMPF can trigger Input Capture interrupt<br>0 = Disabling flag CAPCMPF can trigger Input Capture interrupt  |
| [20]    | <b>CAPOV_IEN</b>  | <b>Enable CAPOVF Trigger Input Capture Interrupt</b><br>1 = Enabling flag OVUNF can trigger Input Capture interrupt<br>0 = Disabling flag OVUNF can trigger Input Capture interrupt   |
| [19]    | -                 | <b>Reserved.</b>  |
| [18]    | <b>CAPTF2_IEN</b> | <b>Enable Input Capture Channel 2 Interrupt</b><br>1 = Enabling flag CAPTF2 can trigger Input Capture interrupt<br>0 = Disabling flag CAPTF2 can trigger Input Capture interrupt  |
| [17]    | <b>CAPTF1_IEN</b> | <b>Enable Input Capture Channel 1 Interrupt</b><br>1 = Enabling flag CAPTF1 can trigger Input Capture interrupt<br>0 = Disabling flag CAPTF1 can trigger Input Capture interrupt  |
| [16]    | <b>CAPTF0_IEN</b> | <b>Enable Input Capture Channel 0 Interrupt</b><br>1 = Enabling flag CAPTF0 can trigger Input Capture interrupt<br>0 = Disabling flag CAPTF0 can trigger Input Capture interrupt  |
| [15:14] | -                 | <b>Reserved.</b>  |
| [13:12] | <b>CAPSEL2</b>    | <b>CAP2 Input Source Selection Bit</b><br>11 = CAP2 input is from signal ADCMPO <sub>x</sub> (ADC compare output <i>x</i> ).<br>10 = CAP2 input is from signal CHX of QEI controller unit <i>x</i> .<br>01 = CAP2 input is from signal CPO2 (Analog comparator 2 output)<br>00 = CAP2 input is from port pin IC2.<br><b>Note:</b> Input capture unit <i>n</i> matches QEI or comparator unit <i>x</i> , where <i>x</i> = 0~1. |

| Bits         | Description        |   |              |                    |    |         |    |           |    |           |    |            |
|--------------|--------------------|---|--------------|--------------------|----|---------|----|-----------|----|-----------|----|------------|
| [11:10]      | <b>CAPSEL1</b>     | <b>CAP1 Input Source Selection Bit</b><br>11 = CAP1 input is from signal OPDO1 (OP1 digital output).<br>10 = CAP1 input is from signal CHB of QEI controller unit x.<br>01 = CAP1 input is from signal CPO1 (Analog comparator 1 output)<br>00 = CAP1 input is from port pin IC1.<br><b>Note:</b> Input capture unit n matches QEI or comparator unit x, where x = 0~1. |              |                    |    |         |    |           |    |           |    |            |
| [9:8]        | <b>CAPSEL0</b>     | <b>CAP0 Input Source Selection Bit</b><br>11 = CAP0 input is from signal OPDO0 (OP0 digital output).<br>10 = CAP0 input is from signal CHA of QEI controller unit x.<br>01 = CAP0 input is from signal CPO0 (Analog comparator 0 output)<br>00 = CAP0 input is from port pin IC0.<br><b>Note:</b> Input capture unit n matches QEI or comparator unit x, where x = 0~1. |              |                    |    |         |    |           |    |           |    |            |
| [7]          | -                  | <b>Reserved.</b>  |              |                    |    |         |    |           |    |           |    |            |
| [6]          | <b>IC2_EN</b>      | <b>Enable Port Pin IC2 Input to Input Capture Unit</b><br>1 = IC2 input to Input Capture Unit Enabled<br>0 = IC2 input to Input Capture Unit Disabled   |              |                    |    |         |    |           |    |           |    |            |
| [5]          | <b>IC1_EN</b>      | <b>Enable Port Pin IC1 Input to Input Capture Unit</b><br>1 = IC1 input to Input Capture Unit Enabled<br>0 = IC1 input to Input Capture Unit Disabled   |              |                    |    |         |    |           |    |           |    |            |
| [4]          | <b>IC0_EN</b>      | <b>Enable Port Pin IC0 Input to Input Capture Unit</b><br>1 = IC0 input to Input Capture Unit Enabled<br>0 = IC0 input to Input Capture Unit Disabled   |              |                    |    |         |    |           |    |           |    |            |
| [3]          | <b>CAPNF_DIS</b>   | <b>Disable Input Capture Noise Filter</b><br>1 = The noise filter of Input Capture Disabled<br>0 = Noise filter of Input Capture Enabled  |              |                    |    |         |    |           |    |           |    |            |
| [2]          | -                  | <b>Reserved.</b>  |              |                    |    |         |    |           |    |           |    |            |
| [1:0]        | <b>NFCLK_S</b>     | <b>Noise Filter Clock Pre-divided Selection</b><br>To determine the sampling frequency of the Noise Filter clock <table><tr><th>NFCLK_S[1:0]</th><th>Noise Filter Clock</th></tr><tr><td>00</td><td>CAP_CLK</td></tr><tr><td>01</td><td>CAP_CLK/2</td></tr><tr><td>10</td><td>CAP_CLK/4</td></tr><tr><td>11</td><td>CAP_CLK/16</td></tr></table>                        | NFCLK_S[1:0] | Noise Filter Clock | 00 | CAP_CLK | 01 | CAP_CLK/2 | 10 | CAP_CLK/4 | 11 | CAP_CLK/16 |
| NFCLK_S[1:0] | Noise Filter Clock |   |              |                    |    |         |    |           |    |           |    |            |
| 00           | CAP_CLK            |   |              |                    |    |         |    |           |    |           |    |            |
| 01           | CAP_CLK/2          |   |              |                    |    |         |    |           |    |           |    |            |
| 10           | CAP_CLK/4          |   |              |                    |    |         |    |           |    |           |    |            |
| 11           | CAP_CLK/16         |   |              |                    |    |         |    |           |    |           |    |            |



**Input Capture Timer/Counter Control Register (CAP\_CTR1)**

| Register | Offset       | R/W | Description                      | Reset Value |
|----------|--------------|-----|----------------------------------|-------------|
| CAP_CTR1 | CAPx_BA+0x18 | R/W | Input Capture Control Register 1 | 0x0000_0000 |

|    |        |         |    |         |         |            |    |
|----|--------|---------|----|---------|---------|------------|----|
| 31 | 30     | 29      | 28 | 27      | 26      | 25         | 24 |
| -  |        |         |    |         |         |            |    |
| 23 | 22     | 21      | 20 | 19      | 18      | 17         | 16 |
| -  |        |         |    |         |         | CAPCNT_SRC |    |
| 15 | 14     | 13      | 12 | 11      | 10      | 9          | 8  |
| -  | CAPDIV |         |    | -       | CPRLD_S |            |    |
| 7  | 6      | 5       | 4  | 3       | 2       | 1          | 0  |
| -  |        | CAPEDG2 |    | CAPEDG1 |         | CAPEDG0    |    |

| Bits            | Description                |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
|-----------------|----------------------------|--|-----------------|----------------------------|-----|-------------------|-----|-----------|-----|------------|-----|------------|-----|------------|-----|------------|-----|-------------|-----|-------------|
| [31:18]         | -                          | <b>Reserved.</b>   |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| [17:16]         | <b>CAPCNT_SRC</b>          | <p>Capture Timer/Counter Clock Source Select<br/>Select the capture timer/counter clock source</p> <table><tr><th>CAPCNT_SRC[1:0]</th><th>Capture Timer Clock Source</th></tr><tr><td>00</td><td>CAP_CLK (Default)</td></tr><tr><td>01</td><td>CAP0</td></tr><tr><td>10</td><td>CAP1</td></tr><tr><td>11</td><td>CAP2</td></tr></table>  | CAPCNT_SRC[1:0] | Capture Timer Clock Source | 00  | CAP_CLK (Default) | 01  | CAP0      | 10  | CAP1       | 11  | CAP2       |     |            |     |            |     |             |     |             |
| CAPCNT_SRC[1:0] | Capture Timer Clock Source |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 00              | CAP_CLK (Default)          |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 01              | CAP0                       |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 10              | CAP1                       |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 11              | CAP2                       |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| [15]            | -                          | <b>Reserved.</b>   |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| [14:12]         | <b>CAPDIV</b>              | <p><b>Capture Timer Clock Divide Selection</b><br/>The capture timer clock has a pre-divider with four divided options controlled by CAPDIV[2:0].</p> <table><tr><th>CAPDIV[2:0]</th><th>Capture Timer Clock</th></tr><tr><td>000</td><td>CAP_CLK/1</td></tr><tr><td>001</td><td>CAP_CLK/4</td></tr><tr><td>010</td><td>CAP_CLK/16</td></tr><tr><td>011</td><td>CAP_CLK/32</td></tr><tr><td>100</td><td>CAP_CLK/64</td></tr><tr><td>101</td><td>CAP_CLK/96</td></tr><tr><td>110</td><td>CAP_CLK/112</td></tr><tr><td>111</td><td>CAP_CLK/128</td></tr></table> | CAPDIV[2:0]     | Capture Timer Clock        | 000 | CAP_CLK/1         | 001 | CAP_CLK/4 | 010 | CAP_CLK/16 | 011 | CAP_CLK/32 | 100 | CAP_CLK/64 | 101 | CAP_CLK/96 | 110 | CAP_CLK/112 | 111 | CAP_CLK/128 |
| CAPDIV[2:0]     | Capture Timer Clock        |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 000             | CAP_CLK/1                  |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 001             | CAP_CLK/4                  |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 010             | CAP_CLK/16                 |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 011             | CAP_CLK/32                 |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 100             | CAP_CLK/64                 |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 101             | CAP_CLK/96                 |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 110             | CAP_CLK/112                |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |
| 111             | CAP_CLK/128                |  |                 |                            |     |                   |     |           |     |            |     |            |     |            |     |            |     |             |     |             |

| Bits         | Description                          |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
|--------------|--------------------------------------|--|--------------|-------------------|-----|--------------------|-----|---------------------|-----|--------------------------------------|-----|--------|--------|----------|
| [11]         | -                                    | Reserved.  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| [10:8]       | CPRLD_S                              | <b>CAPCNT Reload Trigger Source Selection</b><br>If the reload function is enabled (RLD_EN = 1), when a reload trigger event comes, the CAPCNT is reloaded with CAPCNTCMP.<br>CPRLD_S[2:0] determines the CAPCNT reload trigger source<br><table><tr><th>CPRLD_S[2:0]</th><th>Reload Trigger by</th></tr><tr><td>000</td><td>CAPTF0</td></tr><tr><td>001</td><td>CAPTF1</td></tr><tr><td>010</td><td>CAPTF2</td></tr><tr><td>100</td><td>CAPOVF</td></tr><tr><td>Others</td><td>Reserved</td></tr></table> | CPRLD_S[2:0] | Reload Trigger by | 000 | CAPTF0             | 001 | CAPTF1              | 010 | CAPTF2                               | 100 | CAPOVF | Others | Reserved |
| CPRLD_S[2:0] | Reload Trigger by                    |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 000          | CAPTF0                               |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 001          | CAPTF1                               |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 010          | CAPTF2                               |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 100          | CAPOVF                               |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| Others       | Reserved                             |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| [7:6]        | -                                    | Reserved.  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| [5:4]        | CAPEDG2                              | <b>Channel 2 Captured Edge Selection</b><br>Input capture can detect falling edge change only, rising edge change only or one of both edge change<br><table><tr><th>CAPEDG2[1:0]</th><th>Detected Edge</th></tr><tr><td>00</td><td>Detect rising edge</td></tr><tr><td>01</td><td>Detect falling edge</td></tr><tr><td>1x</td><td>Detect either rising or falling edge</td></tr></table>   | CAPEDG2[1:0] | Detected Edge     | 00  | Detect rising edge | 01  | Detect falling edge | 1x  | Detect either rising or falling edge |     |        |        |          |
| CAPEDG2[1:0] | Detected Edge                        |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 00           | Detect rising edge                   |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 01           | Detect falling edge                  |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 1x           | Detect either rising or falling edge |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| [3:2]        | CAPEDG1                              | <b>Channel 1 Captured Edge Selection</b><br>Input capture can detect falling edge change only, rising edge change only or one of both edge change<br><table><tr><th>CAPEDG1[1:0]</th><th>Detected Edge</th></tr><tr><td>00</td><td>Detect rising edge</td></tr><tr><td>01</td><td>Detect falling edge</td></tr><tr><td>1x</td><td>Detect either rising or falling edge</td></tr></table>   | CAPEDG1[1:0] | Detected Edge     | 00  | Detect rising edge | 01  | Detect falling edge | 1x  | Detect either rising or falling edge |     |        |        |          |
| CAPEDG1[1:0] | Detected Edge                        |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 00           | Detect rising edge                   |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 01           | Detect falling edge                  |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 1x           | Detect either rising or falling edge |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| [1:0]        | CAPEDG0                              | <b>Channel 0 Captured Edge Selection</b><br>Input capture can detect falling edge change only, rising edge change only or one of both edge change<br><table><tr><th>CAPEDG0[1:0]</th><th>Detected Edge</th></tr><tr><td>00</td><td>Detect rising edge</td></tr><tr><td>01</td><td>Detect falling edge</td></tr><tr><td>1x</td><td>Detect either rising or falling edge</td></tr></table>   | CAPEDG0[1:0] | Detected Edge     | 00  | Detect rising edge | 01  | Detect falling edge | 1x  | Detect either rising or falling edge |     |        |        |          |
| CAPEDG0[1:0] | Detected Edge                        |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 00           | Detect rising edge                   |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 01           | Detect falling edge                  |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |
| 1x           | Detect either rising or falling edge |  |              |                   |     |                    |     |                     |     |                                      |     |        |        |          |

**Input Capture Timer/Counter Status Register (CAP\_STS)**

| Register | Offset       | R/W | Description                   | Reset Value |
|----------|--------------|-----|-------------------------------|-------------|
| CAP_STS  | CAPx_BA+0x1C | R/W | Input Capture Status Register | 0x0000_0000 |

|    |    |        |         |    |        |        |        |
|----|----|--------|---------|----|--------|--------|--------|
| 31 | 30 | 29     | 28      | 27 | 26     | 25     | 24     |
| -  |    |        |         |    |        |        |        |
| 23 | 22 | 21     | 20      | 19 | 18     | 17     | 16     |
| -  |    |        |         |    |        |        |        |
| 15 | 14 | 13     | 12      | 11 | 10     | 9      | 8      |
| -  |    |        |         |    |        |        |        |
| 7  | 6  | 5      | 4       | 3  | 2      | 1      | 0      |
| -  |    | CAPOVF | CAPCMPF | -  | CAPTF2 | CAPTF1 | CAPTF0 |

| Bits   | Description    |   |
|--------|----------------|---|
| [31:6] | -              | <b>Reserved.</b>  |
| [5]    | <b>CAPOVF</b>  | <b>Input Capture Counter Overflow Flag</b><br>Flag is set by hardware when input capture up counter (CAP_CNT) overflows from 0x00FF_FFFF to 0.<br>1 = CAP_CNT overflows.<br>0 = No overflow occurs in CAP_CNT.<br><b>Note:</b> This bit is only cleared by writing 1 to itself through software.  |
| [4]    | <b>CAPCMPF</b> | <b>Input Capture Compare-match Flag</b><br>If the input capture compare function is enabled, the flag is set by hardware while capture counter (CAP_CNT) up counts and reach to the CAP_CNTCMP value.<br>1 = CAP_CNT counts to the same as CAP_CNTCMP value.<br>0 = CAP_CNT does not match with CAP_CNTCMP value.<br><b>Note:</b> This bit is only cleared by writing 1 to itself through software. |
| [3]    | -              | <b>Reserved.</b>  |
| [2]    | <b>CAPTF2</b>  | <b>Input Capture Channel 2 Captured Flag</b><br>When the input capture channel 2 detects a valid edge change at CAP2 input, it will set flag CAPTF2 to high.<br>1 = A valid edge change is detected at CAP2 input.<br>0 = No valid edge change is detected at CAP2 input.<br><b>Note:</b> This bit is only cleared by writing 1 to itself through software.   |

| Bits | Description   |  |
|------|---------------|--|
| [1]  | <b>CAPTF1</b> | <p><b>Input Capture Channel 1 Captured Flag</b></p> <p>When the input capture channel 1 detects a valid edge change at CAP1 input, it will set flag CAPTF1 to high.</p> <p>1 = A valid edge change is detected at CAP1 input.</p> <p>0 = No valid edge change is detected at CAP1 input.</p> <p><b>Note:</b> This bit is only cleared by writing 1 to itself through software.</p> |
| [0]  | <b>CAPTF0</b> | <p><b>Input Capture Channel 0 Captured Flag</b></p> <p>When the input capture channel 0 detects a valid edge change at CAP0 input, it will set flag CAPTF0 to high.</p> <p>1 = A valid edge change is detected at CAP0 input.</p> <p>0 = No valid edge change is detected at CAP0 input.</p> <p><b>Note:</b> This bit is only cleared by writing 1 to itself through software.</p> |

## 18 QUADRATURE ENCODER INTERFACE (QEI)

### 18.1 Overview

There are two QEI controllers in this device. The Quadrature Encoder Interface (QEI) decodes speed of rotation and motion sensor information. It can be used in any application that uses a quadrature encoder for feedback.

### 18.2 Features

- Up to two QEI controllers, QEI0 and QEI1.
- Two QEI phase inputs, QEA and QEB; One Index input.
- One QEI control register (QEI\_CTR) and one QEI Status Register (QEI\_STS)
- Four Quadrature encoder pulse counter operation modes
  - Mode0: x4 free-counting mode
  - Mode1: x2 free-counting mode
  - Mode2: x4 compare-counting mode
  - Mode3: x2 compare-counting mode
- Encoder Pulse Width measurement mode

### 18.3 QEI Architecture

The QEI controller inputs, QEA and QEB, accept the outputs from a quadrature encoded source, such as incremental optical shaft encoder. Two channels, A and B, nominally 90 degrees out of phase, are required. A quadrature encoder usually provides an index signal (to pin IDX) which can be used to indicate an absolute position. There is a noise filter and polarity control for each signal before QEI control unit.

The following figures illustrate the architecture of Quadrature Encoder Interface Controller.

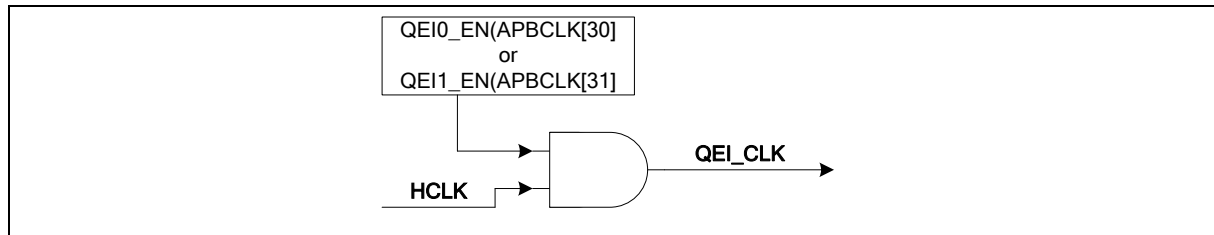


Figure 18-1 QEI Clock Source Control

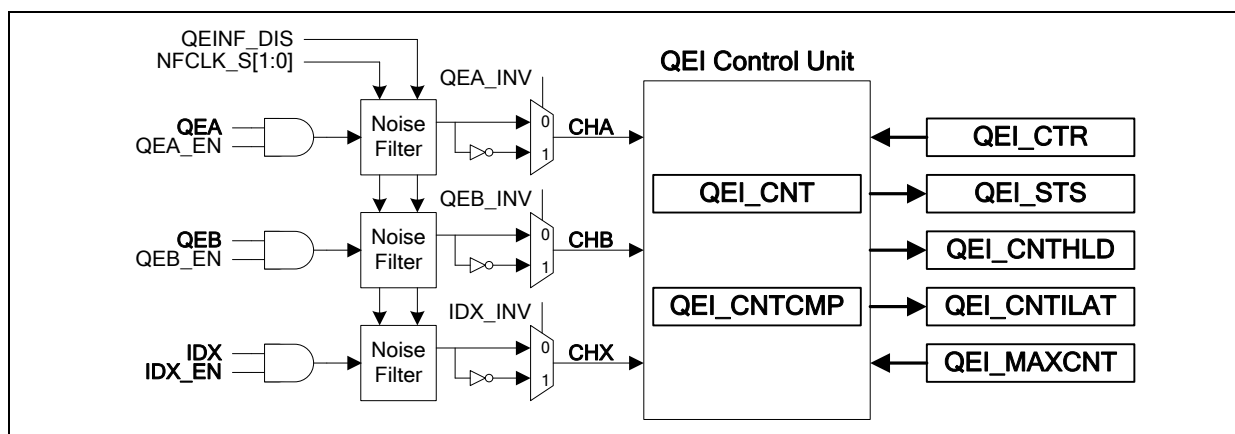


Figure 18-2 QEI Block Diagram

The QEI control logic detects the relation of phase lead/lag between the filtered signals CHA and CHB and CHX to produce direction indication bit (DIR) and clock (QCLK) to control pulse counter. The comparator/reload logic compares the pulse counter and maximum count and control the function of reloading pulse counter in compare-counting mode. In Free-counting mode the pulse counter (QEI\_CNT) will count until the 0xFFFF\_FFFF value; while in Compare-counting mode the pulse counter will counts until the QEI\_MAXCNT value and the pulse counter will be reset to 0 to restart the next cyclic counting.

## 18.4 Input Noise Filter

Each pin of QEI inputs is equipped a noise filter which can filter the unwanted noise from. The QEA, QEB and IDX noise filters can be disabled through bits QEINF\_DIS. If enabled, the capture logic required to sample 4 consecutive same capture input value in order to recognize an edge as a capture event. A possible implementation of digital noise filter is as follow; the interval between pulses requirement for input capture is 4 QEI\_CLK clocks width. Any pulse width less than or equal to 3 QEI\_CLK clocks will not have any trigger. CHA, CHB and CHX are the outputs of QEA, QEB and IDX respectively after going through noise filter and polarity control. See Figure 18–2 and Figure 18–3. If the noise filter is disabled the input signals QEA, QEB and IDX are passed to the internal signals CHA, CHB and CHX respectively without any delay.

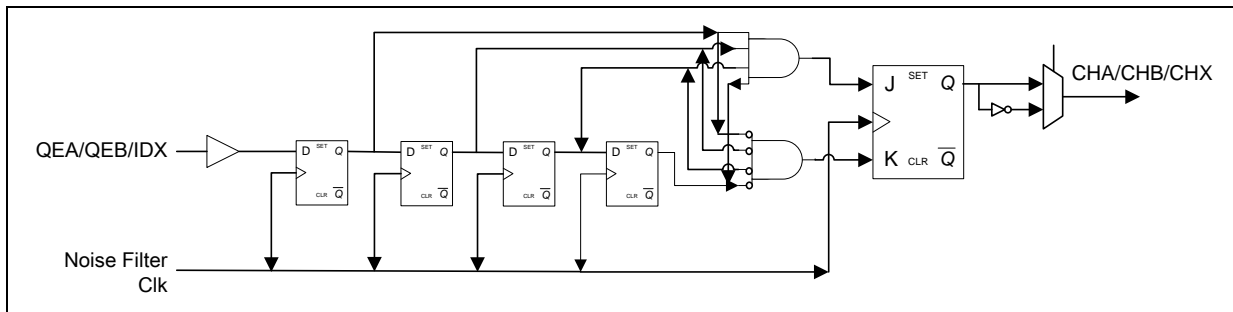


Figure 18–3 Noise Filter

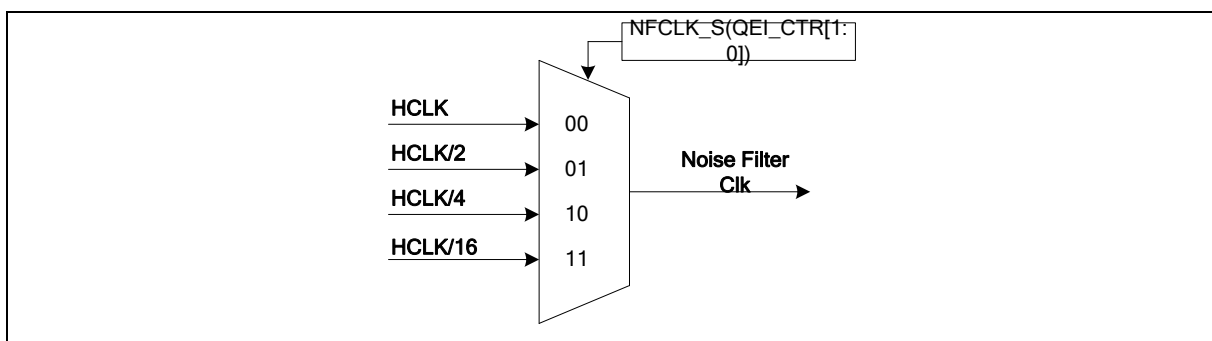


Figure 18–4 Noise Filter Sampling Clock Selection

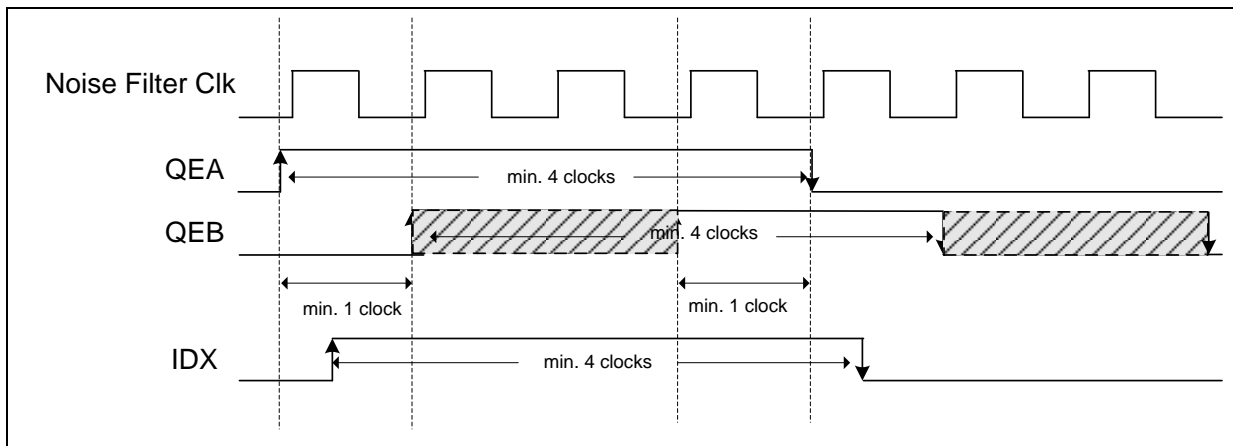


Figure 18–5 QEA/QEB/IDX Timing Requirement through Noise Filter

## 18.5 Operation of Quadrature Encoder Interface

There are four Quadrature encoder pulse counter operation modes

- Mode0: x4 free-counting mode
- Mode1: x2 free-counting mode
- Mode2: x4 compare-counting mode
- Mode3: x2 compare-counting mode

### 18.5.1 Free-counting mode

The quadrature encoder pulse counter (QEI\_CNT) up or down counts according direction indication bit (DIR in QEI\_STS [27]). When overflow or underflow occurs, it sets flag OVUNF (QEI\_STS[24]). Refer to Figure 18–6 and Figure 18–7 for detailed timing.

### 18.5.2 Compare-counting mode

Pulse counter up or down counts according to direction indication bit (DIR). On up counting, flag OVUNF will be asserted when QEI\_CNT overflows from QEI\_MAXCNT to 0 on **the next CHA edge for x2 counting mode, and on CHA/CHB edge for x4 counting mode**. On down counting, flag OVUNF will be asserted when QEI\_CNT underflows from 0 to QEI\_MAXCNT **on the next CHA edge for x2 counting mode, and on CHA/CHB edge for x4 counting mode**. This mode provides the position of a rotor to user. If a quadrature encoder output 1024 pulses to CHA per round, user can write QEI\_MAXCNT and QEI\_CNTCMP with 4095 in x4 mode or 2047 in x2 mode and reset QEI\_CNT at initial before compare-counting mode is active. When the QEI\_CNT overflows from QEI\_CNTCMP, here QEI\_CNTCMP should be preset the same value as QEI\_MAXCNT, it means rotor runs one round on next CHA/CHB edge. Refer to Figure 18–6 and Figure 18–7 for detailed timing.

### 18.5.3 X4/X2 counting modes

In **x4 counting mode**, the pulse counter increases or decreases one on every CHA and CHB edge based on the phase relationship of CHA and CHB signals.

QEI x4 Counting mode provides for a finer resolution of the rotor position, since the counter increments or decrements more frequently for each QEA/QEB input pulse pair than in QEI x2 mode. This mode is selected by setting the QEI Counting Mode Selection bits (QEI\_CTR[9:8]) to [0:0] or [1:0]. In this mode, the QEI logic detects every edge on every QEA and QEB input edges.



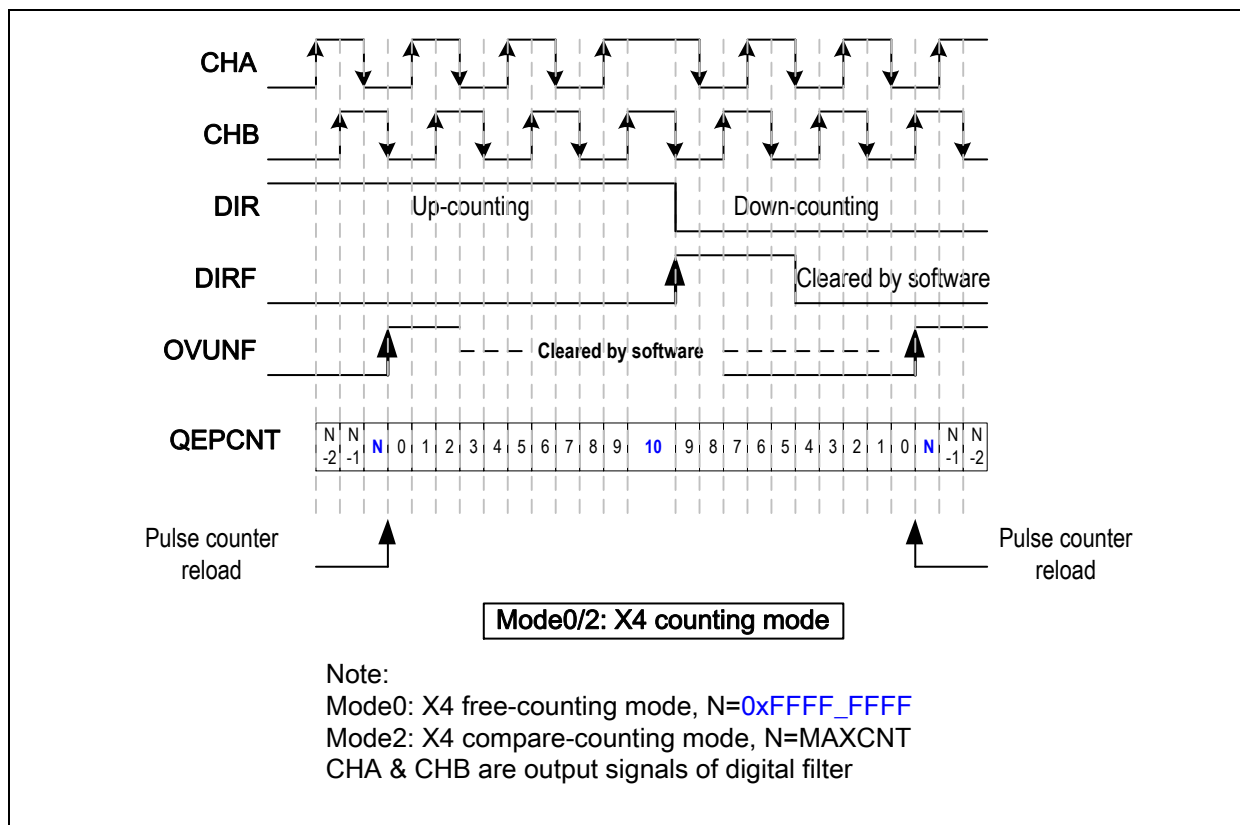


Figure 18–6 X4 Counting Mode

In **x2 counting mode**, the pulse counter increases or decreases one on every CHA edge based on the phase relationship of CHA and CHB signals.

QEI x2 Counting mode is selected by setting the QEI Counting Mode Selection bits (QEI\_CTR[9:8]) to [0:1] or [1:1]. In this mode, the QEI logic detects every edge on the QEA input only. Every rising and falling edge on the QEA signal clocks the pulse counter.

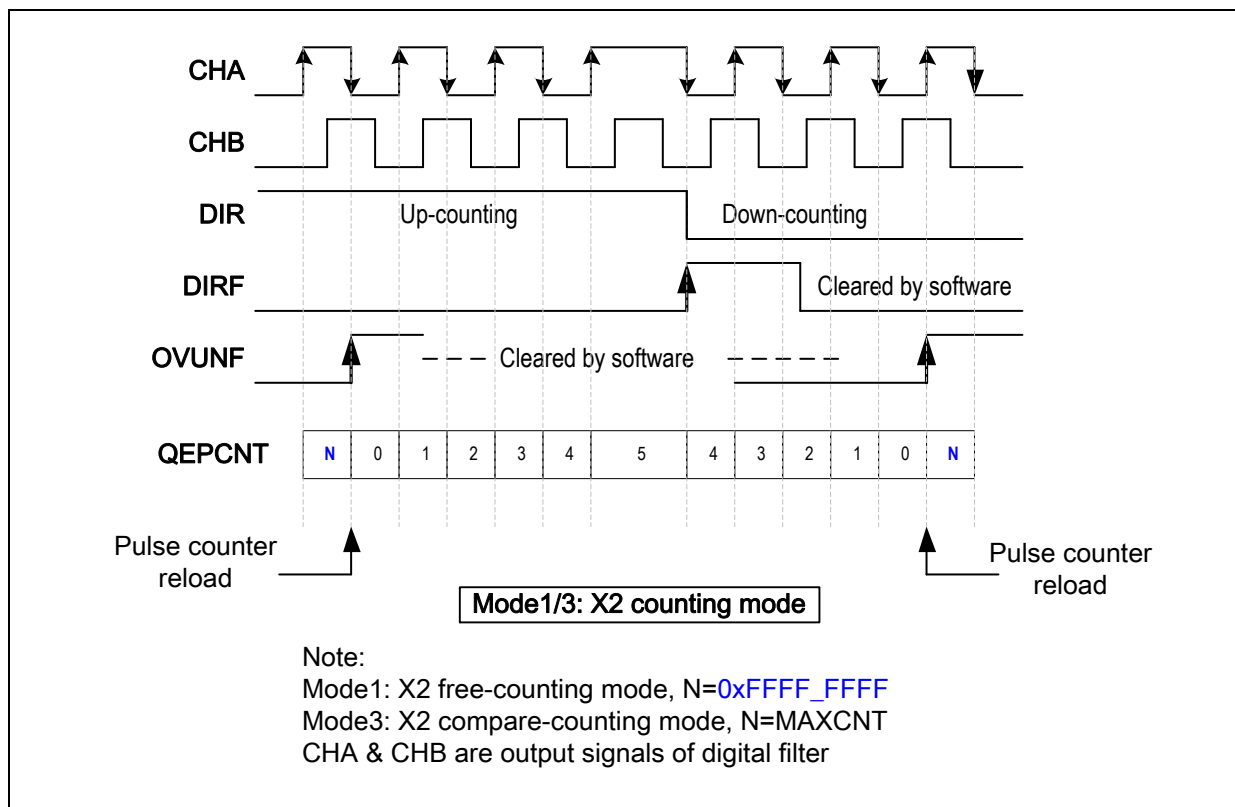


Figure 18–7 X2 Counting Mode

#### 18.5.4 Direction of Count

If CHA lead CHB, the pulse counter is increased by 1. If CHA lags CHB, the pulse counter is decreased by 1. The QEI control logic generates a signal that sets the DIR bit (QEI\_STS[27]); this in turn determines the direction of the count. When CHA leads CHB, DIR is set as 1, and the position counter increments on every active edge. When CHA lags CHB, DIR is cleared, and the position counter decrements on every active edge. Refer to below table.

| Current Signal Detected | Previous Signal Detected |     |         |     | DIR<br>(Counting Direction) |
|-------------------------|--------------------------|-----|---------|-----|-----------------------------|
|                         | Rising                   |     | Falling |     |                             |
|                         | CHA                      | CHB | CHA     | CHB |                             |
| CHA rising              |                          |     |         | ✓   | 1 (Increment)               |
|                         |                          | ✓   |         |     | 0 (Decrement)               |
|                         |                          |     | ✓       |     | Toggle (direction change)   |
| CHA falling             |                          |     |         | ✓   | 0 (Decrement)               |

|             |   |   |   |   |                           |
|-------------|---|---|---|---|---------------------------|
|             |   | ✓ |   |   | 1 (Increment)             |
|             | ✓ |   |   |   | Toggle (direction change) |
| CHB rising  | ✓ |   |   |   | 1 (Increment)             |
|             |   |   | ✓ |   | 0 (Decrement)             |
|             |   |   |   | ✓ | Toggle (direction change) |
| CHB falling |   |   | ✓ |   | 1 (Increment)             |
|             | ✓ |   |   |   | 0 (Decrement)             |
|             |   | ✓ |   |   | Toggle (direction change) |

Table 18-1 Direction of Count

### 18.5.5 Up-Counting

Under the forward direction the DIR bit is 1 when up-counting. Software needs to clear the OVUNF flag. For the free-counting mode the QE1\_CNT counter will counts until it matches 0xFFFF\_FFFF and next edges on the forward direction will set bit OVUNF high and reset QE1\_CNT to 0. For compare-counting mode the QE1\_CNT counter counts until the QE1\_MAXCNT value and next edges on the forward direction will set bit OVUNF high and reset QE1\_CNT to 0. Changes of direction trigger a down-count and QE1\_CNT decreasing in counter value. For X2 mode, only CHA edge will set OVUNF while for X4 mode both CHA and CHB edges will set OVUNF.

### 18.5.6 Down-Counting

A change of direction will causes the counter to down-count for x2/x4 counting mode. It is indicated with the DIR bit as 0 and DIRF flag is set to 1. At this stage the QE1\_CNT will starts to down-count. In free-counting mode the pulse counter will reload with 0xFFFF\_FFFF when it down counts to 0 and sets OVUNF to high in the next edge. The pulse counter will reload with QE1\_MAXCNT when it down counts to 0 in Compare-counting mode and sets OVUNF to high in the next edge. For X2 mode, only CHA edge will set OVUNF while for X4 mode both CHA and CHB edges will set OVUNF.

## 18.6 Compare Function

The compare function in QEI controller is to compare the dynamic counting QEI\_CNT with the compare register QEI\_CNTCMP. When QEI\_CNT up or down counts and reaches QEI\_CNTCMP, the flag CMPF will be set. Set bit CMP\_EN (QEI\_CTR[28]) to one to enable the compare function otherwise disable it.

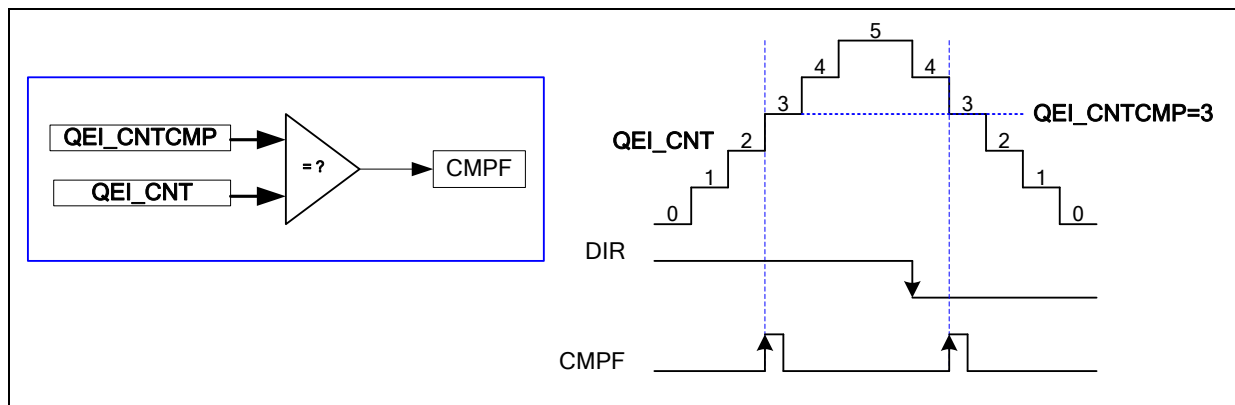


Figure 18–8 Compare Operation

## 18.7 Reload Counter by Pin IDX

The QEI\_CNT counter can be reset to 0 or reload with the content of QEI\_MAXCNT by the signal CHX (the filtered and polarity-set output of pin IDX) trigger. When the IDX Reload bit **IDXRLD\_EN** (QEI\_CTR[27]) is set, a rising edge of CHX causes QEI controller to reset the QEI\_CNT to 0 if the counter is in up-counting; if the counter is in down-counting the rising edge of CHX causes the QEI controller reload the QEI\_CNT with the content of QEI\_MAXCNT. Refer to Figure 18–9 for the detail.

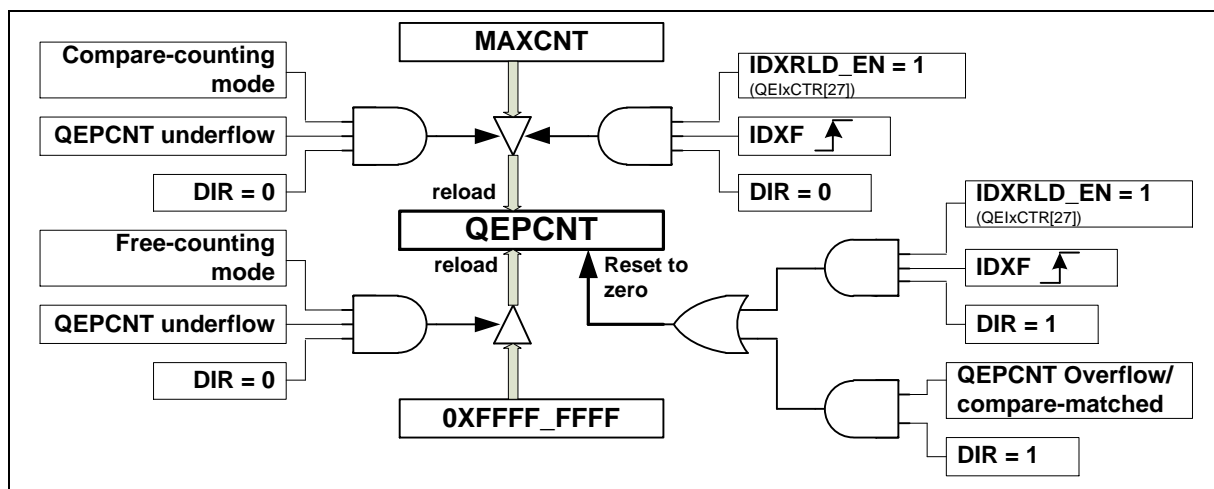


Figure 18–9 QEI\_CNT Reload/Reset Control

### 18.8 Capture QEP Counter

If the bit **HOLDCNT** (QEI\_CTR[24]) is set, the QEI\_CNT content will be captured into QEI Counter Hold Register (QEI\_CNTHLD), the data will be hold until the next HOLDCNT trigger comes. The bit HOLDCNT can be set by writing 1 to itself through software or the rising edge of timers interrupt flags (TIF in TISRx[0]). **The bit HOLDCNT is auto cleared by hardware after QEI\_CNTHLD captures the content of QEI counter.**

If the bit **IDXLAT** (QEI\_CTR[25]) is set, the QEI\_CNT content will be latched into QEI Counter Index Latch Register (QEI\_CNTILAT) at every rising edge of CHX signal.

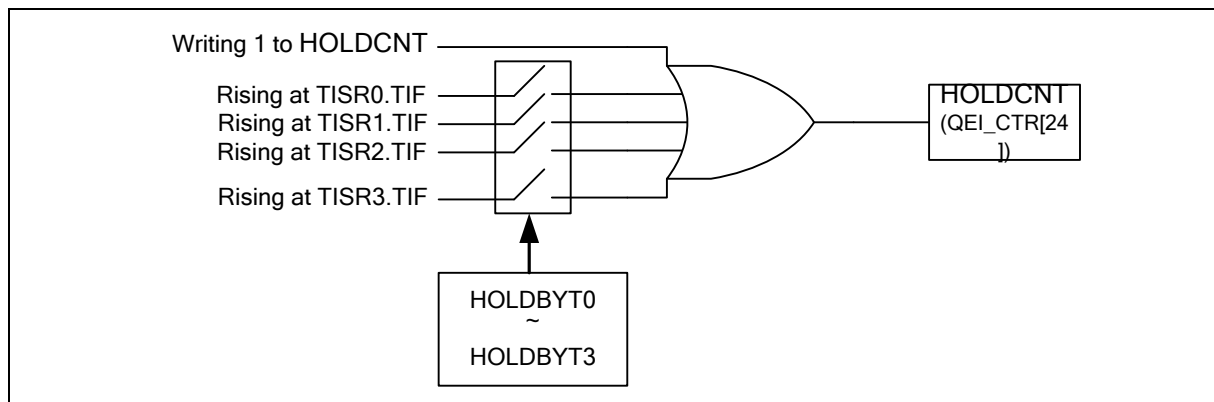


Figure 18–10 Trigger Control of Capturing QEP Counter

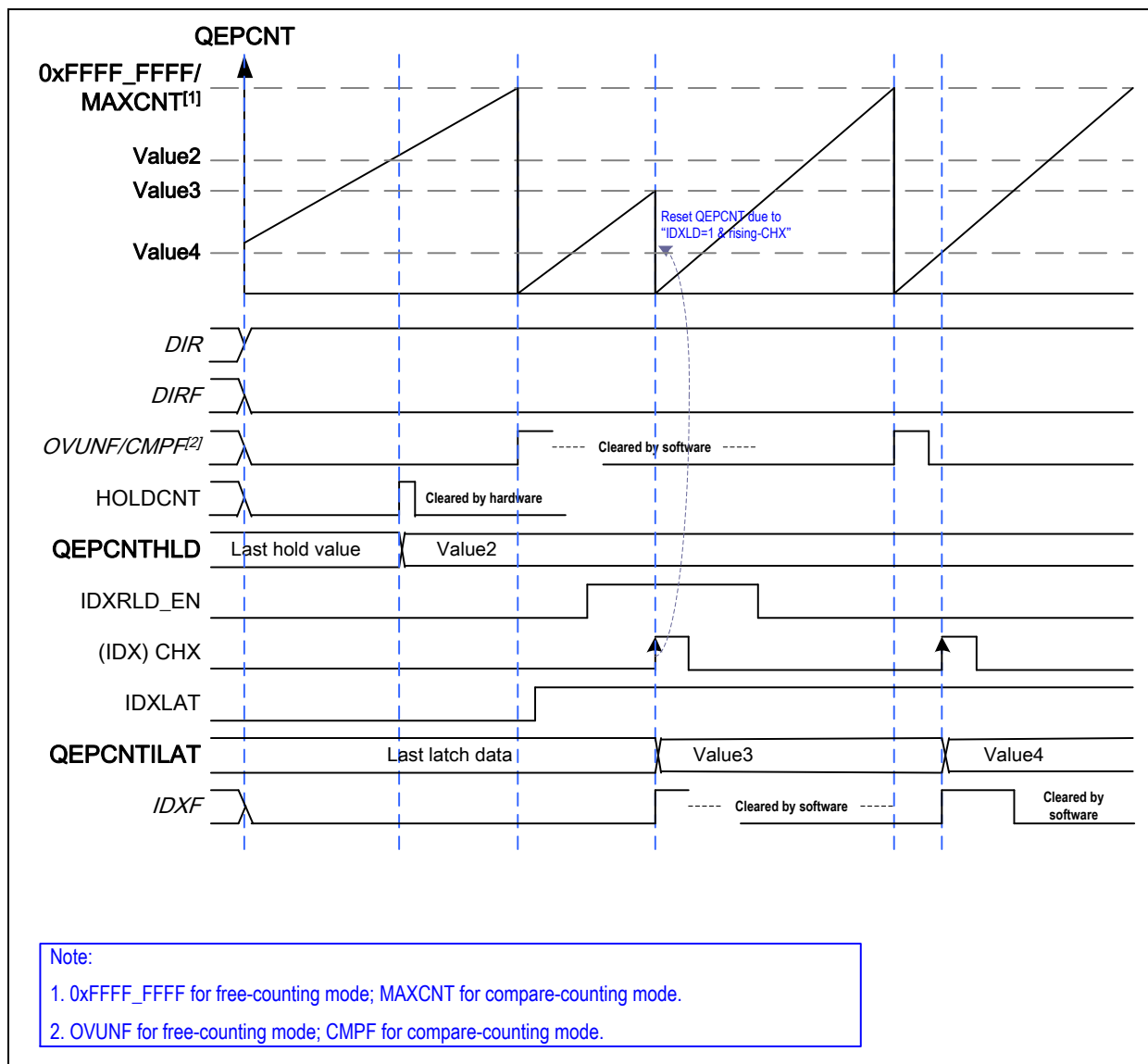


Figure 18–11 Capture and Latch QEP Counter

### 18.9 QEI Interrupt Architecture

There are four interrupt sources, each one of them has an interrupt flag and enable control bit, can trigger QEI Interrupt. When QEI counter is up-counting and QEI\_CNT overflows or down-counting and underflows, the Overflow/Underflow flag (OVUNF in QEI\_STS[24]) will be set by hardware and it will trigger QEI Interrupt request if bit OVUN\_IEN (QEI\_CTR[16]) is high. When QEI controller detects the encoder rotation change, it toggles the direction indication bit DIR (QEI\_STS[27]) and the flag DIRF (QEI\_STS[26]) will be set by hardware that requests the QEI interrupt if bit DIR\_IEN (QEI\_CTR[17]) is set. When the QEI counter counting value is equal to the value of QEI Counter Compare Register (QEI\_CNTCMP), the flag CMPF (QEI\_STS[19]) will be set by hardware and the QEI Interrupt will be requested if bit CMP\_IEN (QEI\_CTR[18]) is high. When QEI controller detects a rising edge at signal CHX (the filtered and polarity-set output of pin IDX), the flag IDXF will set by hardware and the QEI interrupt will be requested if bit IDX\_IEN (QEI\_CTR[19]) is set. **Note that the four flags, OVUNF, DIRF, CMPF and IDXF are set by hardware and must be cleared by software.** Figure 18–12 demonstrates the architecture of Quadrature Encoder Interface Controller interrupts.

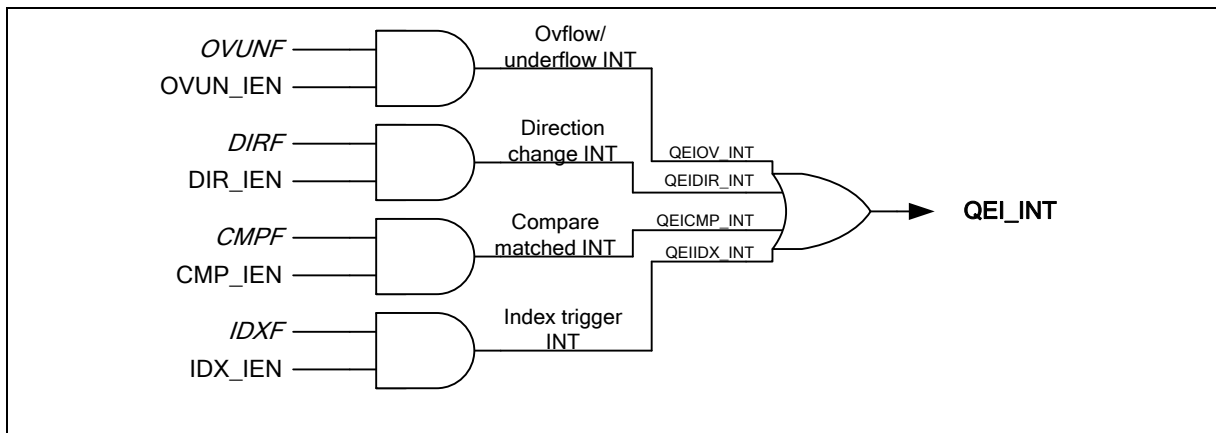


Figure 18–12 Quadrature Encoder Interface Interrupt Architecture Diagram



### 18.10 Register Map

R: read only, W: write only, R/W: both read and write

| Register   | Offset       | R/W | Description                            | Reset Value |
|--|--------------|-----|--|-------------|
| <b>QEI Base Address:</b><br>$QEIx\_BA = 0x401C\_0000 + (0x4000 * x)$<br>$x = 0, 1$ |              |     |  |             |
| <b>QEI_CNT</b>   | QEIx_BA+0x00 | R/W | QEI Pulse Counter                      | 0x0000_0000 |
| <b>QEI_CNTHLD</b>  | QEIx_BA+0x04 | R/W | QEI Pulse Counter Hold Register        | 0x0000_0000 |
| <b>QEI_CNTILAT</b>   | QEIx_BA+0x08 | R/W | QEI Pulse Counter Index Latch Register | 0x0000_0000 |
| <b>QEI_CNTCMP</b>  | QEIx_BA+0x0C | R/W | QEI Pulse Counter Compare Register     | 0x0000_0000 |
| <b>QEI_MAXCNT</b>  | QEIx_BA+0x14 | R/W | QEI Pre-set Maximum Count Register     | 0x0000_0000 |
| <b>QEI_CTR</b>   | QEIx_BA+0x18 | R/W | QEI Control Register                   | 0x0000_0000 |
| <b>QEI_STS</b>   | QEIx_BA+0x2C | R/W | QEI Status Register                    | 0x0000_0000 |

### 18.11 Register Description

#### QEI Pulse Counter (QEI\_CNT)

| Register | Offset       | R/W | Description       | Reset Value |
|----------|--------------|-----|-------------------|-------------|
| QEI_CNT  | QEIx_BA+0x00 | R/W | QEI Pulse Counter | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| QEI_CNT[31:24] |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| QEI_CNT[23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| QEI_CNT[15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| QEI_CNT[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [31:0] | <p><b>QEI_CNT</b></p> <p><b>Quadrature Encoder Pulse Counter</b></p> <p>A 24-bit up/down counter. When an effective phase pulse is detected, this counter is increased by one if the bit DIR in EQICTR is one or decreased by one if the bit DIR is 0. This register performs an integrator which count value is proportional to the encoder position. The pulse counter may be initialized to a predetermined value by one of three events occurs:</p> <ol style="list-style-type: none"> <li>1. Software written if QEI_EN (QEI_CTR[29]) = 0.</li> <li>2. Compare-match event if QEI_EN=1 and QEI is in compare-counting mode.</li> <li>3. Index signal change if QEI_EN=1 and IDXRLD_EN (QEI_CTR[27])=1</li> </ol> |

**QE Pulse Counter Hold Register (QEI\_CNTHLD)**

| Register   | Offset       | R/W | Description                    | Reset Value |
|------------|--------------|-----|--------------------------------|-------------|
| QEI_CNTHLD | QEIx_BA+0x04 | R/W | QE Pulse Counter Hold Register | 0x0000_0000 |

|                   |    |    |    |    |    |    |    |
|-------------------|----|----|----|----|----|----|----|
| 31                | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| QEI_CNTHLD[31:24] |    |    |    |    |    |    |    |
| 23                | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| QEI_CNTHLD[23:16] |    |    |    |    |    |    |    |
| 15                | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| QEI_CNTHLD[15:8]  |    |    |    |    |    |    |    |
| 7                 | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| QEI_CNTHLD[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description  |
|--------|--|
| [31:0] | <p><b>QEI_CNTHLD</b></p> <p><b>Quadrature Encoder Pulse Counter Hold Register</b></p> <p>When bit HOLDCNT (QEIXCTR[24]) goes from low to high, the QEPCNT value is copied into QEPCNTHLD register.</p> |

**QEI Pulse Counter Index Latch Register (QEI\_CNTILAT)**

| Register    | Offset       | R/W | Description                            | Reset Value |
|-------------|--------------|-----|--|-------------|
| QEI_CNTILAT | QEIx_BA+0x08 | R/W | QEI Pulse Counter Index Latch Register | 0x0000_0000 |

|                    |    |    |    |    |    |    |    |
|--------------------|----|----|----|----|----|----|----|
| 31                 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -                  |    |    |    |    |    |    |    |
| 23                 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| QEI_CNTILAT[23:16] |    |    |    |    |    |    |    |
| 15                 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| QEI_CNTILAT[15:8]  |    |    |    |    |    |    |    |
| 7                  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| QEI_CNTILAT[7:0]   |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:24] | -           | Reserved.  |
| [23:0]  | QEI_CNTILAT | <b>Quadrature Encoder Pulse Counter Index Latch Register</b><br>When bit IDXFL (QEI_STS[18]) is set, the QEPI_CNT value is copied into QEI_CNTILAT register. |

### QEI Pulse Counter Compare Register (QEI\_CNTCMP)

| Register   | Offset       | R/W | Description                        | Reset Value |
|------------|--------------|-----|------------------------------------|-------------|
| QEI_CNTCMP | QEIx_BA+0x0C | R/W | QEI Pulse Counter Compare Register | 0x0000_0000 |

|                    |    |    |    |    |    |    |    |
|--------------------|----|----|----|----|----|----|----|
| 31                 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| QEI_CNTILAT[31:24] |    |    |    |    |    |    |    |
| 23                 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| QEI_CNTCMP[23:16]  |    |    |    |    |    |    |    |
| 15                 | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| QEI_CNTCMP[15:8]   |    |    |    |    |    |    |    |
| 7                  | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| QEI_CNTCMP[7:0]    |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [31:0] | <p><b>QEI_CNTCMP</b></p> <p><b>Quadrature Encoder Pulse Counter Compare Register</b></p> <p>if the QEI controller is in the compare-counting mode (CMP_EN in QEI_CTR[28] =1), when the value of QEI_CNT matches the value of QEI_CNTCMP the bit CMPF will be set. This register is software writable.</p> |

### QEI Pre-set Maximum Count Register (QEI\_MAXCNT)

| Register   | Offset       | R/W | Description                        | Reset Value |
|------------|--------------|-----|------------------------------------|-------------|
| QEI_MAXCNT | QEIx_BA+0x14 | R/W | QEI Pre-set Maximum Count Register | 0x0000_0000 |

|                   |    |    |    |    |    |    |    |
|-------------------|----|----|----|----|----|----|----|
| 31                | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| QEI_MAXCNT[31:24] |    |    |    |    |    |    |    |
| 23                | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| QEI_MAXCNT[23:16] |    |    |    |    |    |    |    |
| 15                | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| QEI_MAXCNT[15:8]  |    |    |    |    |    |    |    |
| 7                 | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| QEI_MAXCNT[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:0] | QEI_MAXCNT  | <b>Quadrature Encoder Preset Maximum Count Register</b><br>This register value determined by user stores the maximum value which may be the number of the quadrature encoder pulses in a revolution for the QEI controller compare-counting mode. |

QEI Control Register (QEI\_CTR)

| Register | Offset       | R/W | Description          | Reset Value |
|----------|--------------|-----|----------------------|-------------|
| QEI_CTR  | QEIx_BA+0x18 | R/W | QEI Control Register | 0x0000_0000 |

|          |          |          |          |           |         |         |          |
|----------|----------|----------|----------|-----------|---------|---------|----------|
| 31       | 30       | 29       | 28       | 27        | 26      | 25      | 24       |
| -        | -        | QEI_EN   | CMP_EN   | IDXRDL_EN | -       | IDXLAT  | HOLDCNT  |
| 23       | 22       | 21       | 20       | 19        | 18      | 17      | 16       |
| HOLDBYT3 | HOLDBYT2 | HOLDBYT1 | HOLDBYT0 | IDX_IEN   | CMP_IEN | DIR_IEN | OVUN_IEN |
| 15       | 14       | 13       | 12       | 11        | 10      | 9       | 8        |
| -        | IDX_INV  | QEB_INV  | QEA_INV  | -         | -       | QEIMODE |          |
| 7        | 6        | 5        | 4        | 3         | 2       | 1       | 0        |
| -        | IDX_EN   | QEB_EN   | QEA_EN   | QEINF_DIS | -       | NFCLK_S |          |

| Bits    | Description   |
|---------|---|
| [31:30] | - Reserved.   |
| [29]    | <b>QEI_EN</b><br>Quadrature Encoder Interface Controller Enable Bit<br>1 = QEI controller function Enabled<br>0 = QEI controller function Disabled  |
| [28]    | <b>CMP_EN</b><br>The Compare Function Enable Bit<br>The compare function in QEI controller is to compare the dynamic counting QEPCNT with the compare register QEI_CNTCMP, if QEI_CNT value reaches QEI_CNTCMP, the flag CMPF will be set.<br>1 = The compare function Enabled<br>0 = The compare function Disabled   |
| [27]    | <b>IDXRDL_EN</b><br>Index Trigger QEI_CNT Reload Enable Bit<br>When this bit is high and a rising edge comes on signal CHX, the QEI_CNT will be reset to 0 if the counter is in up-counting type (DIR = 1); while the QEI_CNT will be reloaded with QEI_MAXCNT content if the counter is in down-counting type (DIR = 0).<br>1 = The QEI_CNT re-initialized by Index signal Enabled<br>0 = The reload function Disabled |
| [26]    | -   |
| [25]    | <b>IDXLAT</b><br>Index Latch QEI_CNT Enable Bit<br>If this bit is set to high, the QEI_CNT content will be latched into QEI_CNTILAT at every rising on signal CHX.<br>1 = The index signal latch QEI counter function Enabled<br>0 = The index signal latch QEI counter function Disabled   |

| Bits | Description     |  |
|------|-----------------|--|
| [24] | <b>HOLDCNT</b>  | <b>Hold QEI_CNT Control Bit</b><br>When this bit is set from low to high, the QEI_CNT value is copied into QEI_CNTHLD. This bit may be set by writing 1 to itself through software or Timer0~Timer3 interrupt flag (TISTR.TIF).<br>1 = QEI_CNT content captured and stored in QEI_CNTHLD.<br>0 = No operation.<br><b>Note:</b> This bit is automatically cleared after QEI_CNTHLD holds QEI_CNT value. |
| [23] | <b>HOLDBYT3</b> | <b>Hold QEI_CNT by Timer 3</b><br>1 = A rising edge of bit TISR3.TIF in timer 3 sets HOLDCNT to 1.<br>0 = TISR3.TIF has no effect on HOLDCNT.  |
| [22] | <b>HOLDBYT2</b> | <b>Hold QEI_CNT by Timer 2</b><br>1 = A rising edge of bit TISR2.TIF in timer 2 sets HOLDCNT to 1.<br>0 = TISR2.TIF has no effect on HOLDCNT.  |
| [21] | <b>HOLDBYT1</b> | <b>Hold QEI_CNT by Timer 1</b><br>1 = A rising edge of bit TISR1.TIF in timer 1 sets HOLDCNT to 1.<br>0 = TISR1.TIF has no effect on HOLDCNT.  |
| [20] | <b>HOLDBYT0</b> | <b>Hold QEI_CNT by Timer 0</b><br>1 = A rising edge of bit TISR0.TIF in timer 0 sets HOLDCNT to 1.<br>0 = TISR0.TIF has no effect on HOLDCNT.  |
| [19] | <b>IDX_IEN</b>  | <b>Enable IDX Trigger QEI Interrupt</b><br>1 = The IDX can trigger QEI interrupt Enabled.<br>0 = The IDX can trigger QEI interrupt Disabled.   |
| [18] | <b>CMP_IEN</b>  | <b>Enable CMPF Trigger QEI Interrupt</b><br>1 = The CMPF can trigger QEI controller interrupt Enabled.<br>0 = The CMPF can trigger QEI controller interrupt Disabled.  |
| [17] | <b>DIR_IEN</b>  | <b>Enable DIRF Trigger QEI Interrupt</b><br>1 = The DIRF can trigger QEI controller interrupt Enabled.<br>0 = The DIRF can trigger QEI controller interrupt Disabled.  |
| [16] | <b>OVUN_IEN</b> | <b>Enable OVUNF Trigger QEI Interrupt</b><br>1 = The OVUNF can trigger QEI controller interrupt Enabled.<br>0 = The OVUNF can trigger QEI controller interrupt Disabled.   |
| [15] | -               | <b>Reserved.</b>   |
| [14] | <b>IDX_INV</b>  | <b>Inverse IDX Input Polarity</b><br>1 = IDX input polarity is inversed to QEI controller.<br>0 = Not inverse IDX input polarity.  |
| [13] | <b>QEB_INV</b>  | <b>Inverse QEB Input Polarity</b><br>1 = QEB input polarity is inversed to QEI controller.<br>0 = Not inverse QEB input polarity.  |



| Bits         | Description              |   |              |                    |    |                       |    |                       |    |                          |    |                          |
|--------------|--------------------------|---|--------------|--------------------|----|-----------------------|----|-----------------------|----|--------------------------|----|--------------------------|
| [12]         | QEA_INV                  | <b>Inverse QEA Input Polarity</b><br>1 = QEA input polarity is inversed to QEI controller.<br>0 = Not inverse QEA input polarity  |              |                    |    |                       |    |                       |    |                          |    |                          |
| [11:10]      | -                        | <b>Reserved.</b>  |              |                    |    |                       |    |                       |    |                          |    |                          |
| [9:8]        | QEIMODE                  | <b>QEI Counting Mode Selection</b><br>There are four quadrature encoder pulse counter operation modes <table><tr><th>QEIMODE[1:0]</th><th>QEI Counting Mode</th></tr><tr><td>00</td><td>X4 Free-counting Mode</td></tr><tr><td>01</td><td>X2 Free-counting Mode</td></tr><tr><td>10</td><td>X4 Compare-counting Mode</td></tr><tr><td>11</td><td>X2 Compare-counting Mode</td></tr></table> | QEIMODE[1:0] | QEI Counting Mode  | 00 | X4 Free-counting Mode | 01 | X2 Free-counting Mode | 10 | X4 Compare-counting Mode | 11 | X2 Compare-counting Mode |
| QEIMODE[1:0] | QEI Counting Mode        |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| 00           | X4 Free-counting Mode    |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| 01           | X2 Free-counting Mode    |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| 10           | X4 Compare-counting Mode |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| 11           | X2 Compare-counting Mode |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| [7]          | -                        | <b>Reserved.</b>  |              |                    |    |                       |    |                       |    |                          |    |                          |
| [6]          | IDX_EN                   | <b>Enable IDX Input to QEI Controller</b><br>1 = The IDX input to QEI Controller Enabled<br>0 = The IDX input to QEI Controller Disabled  |              |                    |    |                       |    |                       |    |                          |    |                          |
| [5]          | QEB_EN                   | <b>Enable QEB Input to QEI Controller</b><br>1 = The QEB input to QEI Controller Enabled.<br>0 = The QEB input to QEI Controller Disabled.  |              |                    |    |                       |    |                       |    |                          |    |                          |
| [4]          | QEA_EN                   | <b>Enable QEA Input to QEI Controller</b><br>1 = The QEA input to QEI Controller Enabled.<br>0 = The QEA input to QEI Controller Disabled.  |              |                    |    |                       |    |                       |    |                          |    |                          |
| [3]          | QEINF_DIS                | <b>Disable QEI Controller Input Noise Filter</b><br>1 = The noise filter of QEI controller Disabled.<br>0 = The noise filter of QEI controller Enabled.   |              |                    |    |                       |    |                       |    |                          |    |                          |
| [2]          | -                        | <b>Reserved.</b>  |              |                    |    |                       |    |                       |    |                          |    |                          |
| [1:0]        | NFCLK_S                  | <b>Noise Filter Clock Pre-divided Selection</b><br>To determine the sampling frequency of the Noise Filter clock: <table><tr><th>NFCLK_S[1:0]</th><th>Noise Filter Clock</th></tr><tr><td>00</td><td>QEI_CLK</td></tr><tr><td>01</td><td>QEI_CLK/2</td></tr><tr><td>10</td><td>QEI_CLK/4</td></tr><tr><td>11</td><td>QEI_CLK/16</td></tr></table>   | NFCLK_S[1:0] | Noise Filter Clock | 00 | QEI_CLK               | 01 | QEI_CLK/2             | 10 | QEI_CLK/4                | 11 | QEI_CLK/16               |
| NFCLK_S[1:0] | Noise Filter Clock       |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| 00           | QEI_CLK                  |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| 01           | QEI_CLK/2                |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| 10           | QEI_CLK/4                |   |              |                    |    |                       |    |                       |    |                          |    |                          |
| 11           | QEI_CLK/16               |   |              |                    |    |                       |    |                       |    |                          |    |                          |

### QEI Status Register (QEI\_STS)

| Register | Offset       | R/W | Description         | Reset Value |
|----------|--------------|-----|---------------------|-------------|
| QEI_STS  | QEIx_BA+0x2C | R/W | QEI Status Register | 0x0000_0100 |

|    |    |    |    |      |       |      |      |
|----|----|----|----|------|-------|------|------|
| 31 | 30 | 29 | 28 | 27   | 26    | 25   | 24   |
| -  |    |    |    |      |       |      |      |
| 23 | 22 | 21 | 20 | 19   | 18    | 17   | 16   |
| -  |    |    |    |      |       |      |      |
| 15 | 14 | 13 | 12 | 11   | 10    | 9    | 8    |
| -  |    |    |    |      |       |      | DIR  |
| 7  | 6  | 5  | 4  | 3    | 2     | 1    | 0    |
| -  |    |    |    | DIRF | OVUNF | CMPF | IDXF |

| Bits   | Description |  |
|--------|-------------|--|
| [31:9] | -           | <b>Reserved.</b>   |
| [8]    | DIR         | <b>QEI Counter Counting Direction Indication Bit</b><br>1 = QEI Counter is in up-counting.<br>0 = QEI Counter is in down-counting.<br><b>Note:</b> This bit is set/reset by hardware according to the phase detection between CHA and CHB.   |
| [7:4]  | -           | <b>Reserved.</b>   |
| [3]    | DIRF        | <b>Direction Change Flag</b><br>Flag is set by hardware while QEI counter counting direction is changed, software can clear this bit by writing a one to it.<br>1 = QEI counter counting direction is changed.<br>0 = No change in QEI counter counting direction.<br><b>Note:</b> This bit is only cleared by writing 1 to itself through software.   |
| [2]    | OVUNF       | <b>QEI Counter Overflow or Underflow Flag</b><br>Flag is set by hardware while QEI_CNT overflows from 0xFFFF_FFFF to 0 in Free-counting mode or from the QEI_MAXCNT value to 0 in compare-counting mode. Similarly, the flag is set while QEI counter underflows from 0 to 0xFFFF_FFFF or QEI_MAXCNT.<br>1 = QEI counter occurred counting overflow or underflow.<br>0 = No overflow or underflow occurred in QEI counter.<br><b>Note:</b> This bit is only cleared by writing 1 to itself through software. |

| Bits | Description |  |
|------|-------------|--|
| [1]  | <b>CMPF</b> | <p><b>Compare-match Flag</b></p> <p>If the QEI compare function is enabled, the flag is set by hardware while QEI counter up or down counts and reach to the QEI_CNTCMP value.</p> <p>1 = QEI counter counts to the same as QEI_CNTCMP value.</p> <p>0 = QEI counter does not match with QEI_CNTCMP value.</p> <p><b>Note:</b> This bit is only cleared by writing 1 to itself through software.</p> |
| [0]  | <b>IDXF</b> | <p><b>IDX Detected Flag</b></p> <p>When the QEI controller detects a rising edge on signal CHX it will set flag IDXF to high.</p> <p>1 = A rising edge occurred on signal CHX.</p> <p>0 = No rising edge detected on signal CHX.</p> <p><b>Note:</b> This bit is only cleared by writing 1 to itself through software.</p>   |

## 19 UNIVERSAL ASYNCHRONOUS RECEIVER TRANSMITTER (UART)

### 19.1 Overview

The Universal Asynchronous Receiver/Transmitter (UART) performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU. The UART controller also supports IrDA SIR Function, LIN master/slave mode function and RS-485 mode functions. Each UART channel supports seven types of interrupts including

- Transmitter FIFO empty interrupt (INT\_THRE)
- Receiver threshold level reached interrupt (INT\_RDA),
- Line status interrupt (parity error or frame error or break interrupt) (INT\_RLS),
- Receiver buffer time out interrupt (INT\_TOUT),
- MODEM/Wake-up status interrupt (INT\_MODEM),
- Buffer error interrupt (INT\_BUF\_ERR)
- LIN interrupt (INT\_LIN)

The UART controller is built-in with a 16-byte transmitter FIFO (TX\_FIFO) and a 16-byte receiver FIFO (RX\_FIFO) that reduces the number of interrupts presented to the CPU. The CPU can read the status of the UART at any time during the operation. The reported status information includes the type and condition of the transfer operations being performed by the UART, as well as 4 error conditions (parity error, frame error, break interrupt and buffer error) probably occur while receiving data. The UART includes a programmable baud rate generator that is capable of dividing clock input by divisors to produce the serial clock that transmitter and receiver need. The baud rate equation is  $\text{Baud Rate} = \text{UART\_CLK} / M * [\text{BRD} + 2]$ , where M and BRD are defined in Baud Rate Divider Register (UA\_BAUD). Table 19-1 lists the equations in the various conditions and Table 19-2 lists the UART baud rate setting table.

| Mode | DIV_X_EN | DIV_X_ONE | Divider X  | BRD | Baud Rate Equation                                     |
|------|----------|-----------|------------|-----|--|
| 0    | 0        | 0         | Don't care | A   | $\text{UART\_CLK} / [16 * (A+2)]$                      |
| 1    | 1        | 0         | B          | A   | $\text{UART\_CLK} / [(B+1) * (A+2)]$ , B must $\geq 8$ |
| 2    | 1        | 1         | Don't care | A   | $\text{UART\_CLK} / (A+2)$ , A must $\geq 3$           |

Table 19-1 UART Baud Rate Equation

| System clock = internal 22.1184 MHz high speed oscillator |           |             |                      |                            |           |             |
|---|-----------|-------------|----------------------|----------------------------|-----------|-------------|
| Baud Rate   | Mode 0    |             | Mode 1               |                            | Mode 2    |             |
|   | Parameter | Register    | Parameter            | Register                   | Parameter | Register    |
| 921600  | x         | x           | A=0,B=11             | 0x2B00_0000                | A=22      | 0x3000_0016 |
| 460800  | A=1       | 0x0000_0001 | A=1,B=15<br>A=2,B=11 | 0x2F00_0001<br>0x2B00_0002 | A=46      | 0x3000_002E |
| 230400  | A=4       | 0x0000_0004 | A=4,B=15<br>A=6,B=11 | 0x2F00_0004<br>0x2B00_0006 | A=94      | 0x3000_005E |

|        |       |             |                                       |   |        |             |
|--------|-------|-------------|---------------------------------------|---|--------|-------------|
| 115200 | A=10  | 0x0000_000A | A=10,B=15<br>A=14,B=11                | 0x2F00_000A<br>0x2B00_000E                | A=190  | 0x3000_00BE |
| 57600  | A=22  | 0x0000_0016 | A=22,B=15<br>A=30,B=11                | 0x2F00_0016<br>0x2B00_001E                | A=382  | 0x3000_017E |
| 38400  | A=34  | 0x0000_0022 | A=62,B=8<br>A=46,B=11<br>A=34,B=15    | 0x2800_003E<br>0x2B00_002E<br>0x2F00_0022 | A=574  | 0x3000_023E |
| 19200  | A=70  | 0x0000_0046 | A=126,B=8<br>A=94,B=11<br>A=70,B=15   | 0x2800_007E<br>0x2B00_005E<br>0x2F00_0046 | A=1150 | 0x3000_047E |
| 9600   | A=142 | 0x0000_008E | A=254,B=8<br>A=190,B=11<br>A=142,B=15 | 0x2800_00FE<br>0x2B00_00BE<br>0x2F00_008E | A=2302 | 0x3000_08FE |
| 4800   | A=286 | 0x0000_011E | A=510,B=8<br>A=382,B=11<br>A=286,B=15 | 0x2800_01FE<br>0x2B00_017E<br>0x2F00_011E | A=4606 | 0x3000_11FE |

Table 19-2 UART Baud Rate Setting Table

The UART controller support auto-flow control function that uses two low-level signals, nCTS (clear-to-send) and nRTS (request-to-send), to control the flow of data transfer between the chip and external devices (ex: Modem). When auto-flow is enabled, the UART is not allowed to receive data until the UART asserts nRTS to external device. When the number of bytes in the RX FIFO equals the value of RTS\_TRI\_LEV (UA\_FCR [19:16]), the nRTS is de-asserted. The UART sends data out when UART controller detects nCTS is asserted from external device. If a valid asserted nCTS is not detected the UART controller will not send data out.

The UART controllers also provides Serial IrDA (SIR, Serial Infrared) function (User must set IrDA\_EN (UA\_FUN\_SEL [1]) to enable IrDA function). The SIR specification defines a short-range infrared asynchronous serial transmission mode with 1 start bit, 8 data bits, and 1 stop bit. The maximum data rate supports up to 115.2 Kbps (half duplex). The IrDA SIR block contains an IrDA SIR Protocol encoder/decoder. The IrDA SIR Protocol encoder/decoder is half-duplex only. So it cannot transmit and receive data at the same time. The IrDA SIR physical layer specifies a minimum 10ms transfer delay between transmission and reception, and this delay feature must be implemented by software.

The alternate function of UART controllers is LIN (Local Interconnect Network) function. The LIN mode is selected by setting the UA\_FUN\_SEL[1:0] to '01'. In LIN mode, 1 start bit and 8 data bits format with 1 stop bit are required in accordance with the LIN standard.

For the NuMicro™ NM15xx Series, another alternate function of UART controllers is RS-485 9-bit mode, and direction control provided by nRTS pin or can program GPIO (PB.2 for UART0\_nRTS and PB.6 for UART1\_nRTS) to implement the function by software. The RS-485 mode is selected by setting the UA\_FUN\_SEL register to select RS-485 function. The RS-485 transceiver control is implemented using the nRTS control signal from an asynchronous serial port to enable the RS-485 transceiver. In RS-485 mode, many characteristics of the receiving and transmitting are same as UART.

## 19.2 Features

- Full duplex, asynchronous communications
- Separates receive / transmit 16 bytes entry FIFO for data payloads
- Supports hardware auto flow control/flow control function (nCTS, nRTS) and programmable nRTS flow control trigger level
- Programmable receiver buffer trigger level
- Supports programmable baud-rate generator for each channel individually
- Supports nCTS wake-up function
- Supports 7-bit receiver buffer time out detection function
- Programmable transmitting data delay time between the last stop and the next start bit by setting UA\_TOR [DLY] register
- Supports break error, frame error, parity error and receive / transmit buffer overflow detect function
- Fully programmable serial-interface characteristics
  - Programmable data bit length, 5-, 6-, 7-, 8-bit character
  - Programmable parity bit, even, odd, no parity or stick parity bit generation and detection
  - Programmable stop bit length, 1, 1.5, or 2 stop bit generation
- IrDA SIR function mode
  - Supports 3-/16-bit duration for normal mode
- LIN function mode
  - Supports LIN master/slave mode
  - Supports programmable break generation function for transmitter
  - Supports break detect function for receiver
- RS-485 function mode
  - Supports RS-485 9-bit mode
  - Supports hardware or software direct enable control provided by nRTS pin

### 19.3 Block Diagram

The UART clock control and block diagram are shown in Figure 5-67 and Figure 5-68 respectively.

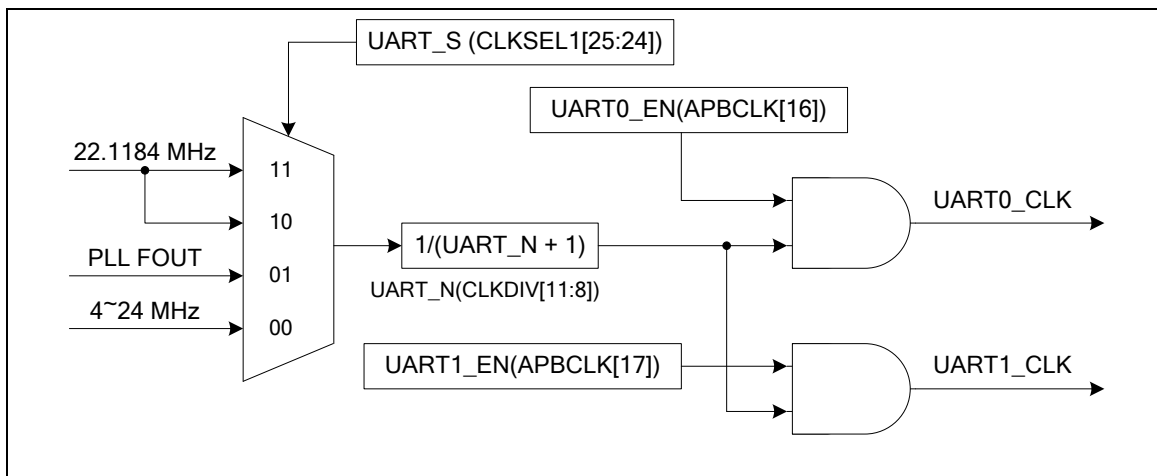


Figure 19-1 UART Clock Control Diagram

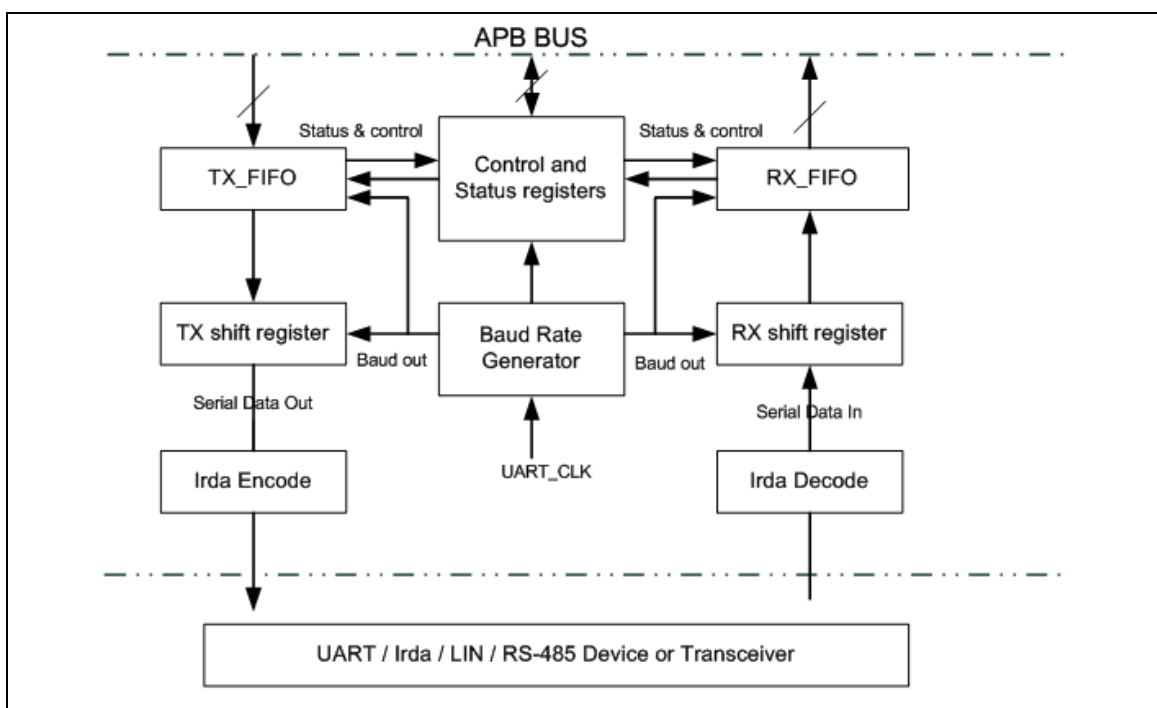


Figure 19-2 UART Block Diagram

### **TX\_FIFO**

The transmitter is buffered with a 16 byte FIFO to reduce the number of interrupts presented to the CPU.

### **RX\_FIFO**

The receiver is buffered with a 16 byte FIFO (plus three error bits per byte) to reduce the number of interrupts presented to the CPU.

### **TX shift Register**

This block is the shifting the transmitting data out of serially control.

### **RX shift Register**

This block is the shifting the receiving data in of serially control.

### **Modem Control Register**

This register controls the interface to the MODEM or data set (or a peripheral device emulating a MODEM).

### **Baud Rate Generator**

Divide the external clock by the divisor to get the desired baud rate. Refer to baud rate equation.

### **IrDA Encode**

This block is IrDA encode control block.

### **IrDA Decode**

This block is IrDA decode control block.

### **Control and Status Register**

This field is register set that including the FIFO control registers (UA\_FCR), FIFO status registers (UA\_FSR), and line control register (UA\_LCR) for transmitter and receiver. The time out control register (UA\_TOR) identifies the condition of time out interrupt. This register set also includes the interrupt enable register (UA\_IER) and interrupt status register (UA\_ISR) to enable or disable the responding interrupt and to identify the occurrence of the responding interrupt. There are seven types of interrupts, transmitter FIFO empty interrupt (INT\_THRE), receiver threshold level reaching interrupt (INT\_RDA), line status interrupt (parity error or framing error or break interrupt) (INT\_RLS), time out interrupt (INT\_TOUT), MODEM/Wake-up status interrupt (INT\_MODEM), Buffer error interrupt (INT\_BUF\_ERR) and LIN bus interrupt.



The following diagram demonstrates the auto-flow control block.

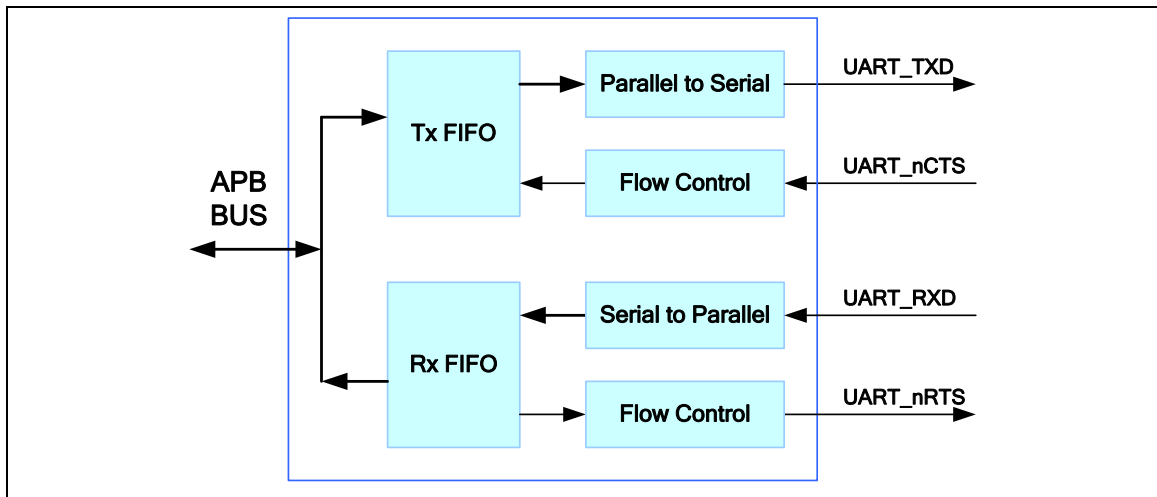


Figure 19-3 Auto Flow Control Block Diagram

## 19.4 IrDA Mode

The UART supports IrDA SIR (Serial Infrared) Transmit Encoder and Receive Decoder, and IrDA mode is selected by setting the in **UA\_FUN\_SEL** register.

In IrDA mode, the UA\_BAUD [DIV\_X\_EN] register must be disabled.

**Baud Rate = Clock / (16 \* BRD)**, where BRD is Baud Rate Divider in UA\_BAUD register.

The following diagram demonstrates the IrDA control block.

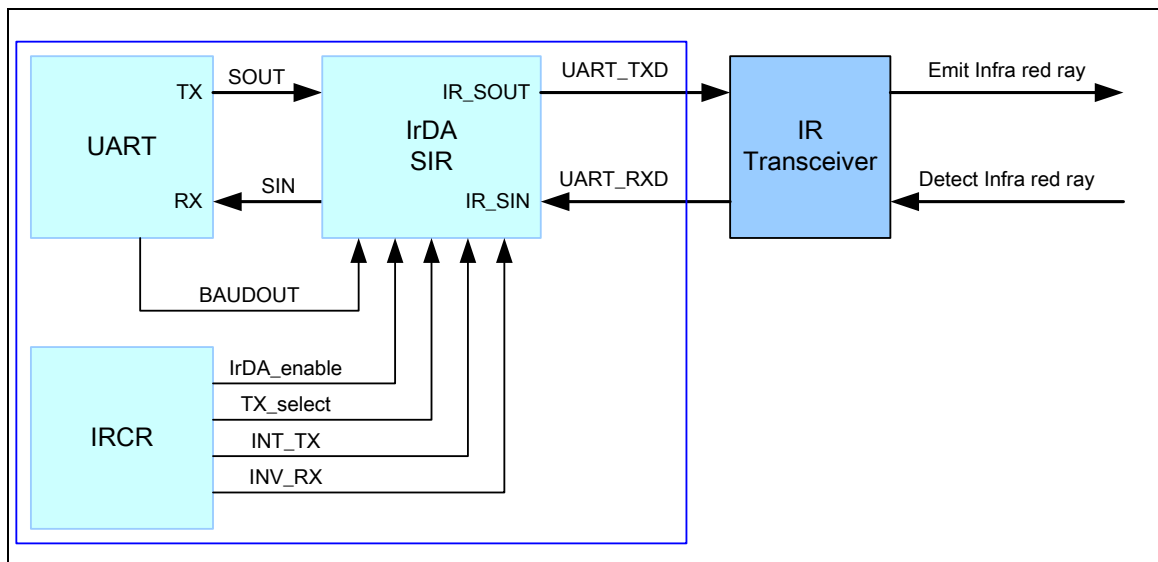


Figure 19-4 IrDA Block Diagram

### 19.4.1.1 IrDA SIR Transmit Encoder

The IrDA SIR Transmit Encoder modulates Non-Return-to Zero (NRZ) transmit bit stream output from UART. The IrDA SIR physical layer specifies the use of Return-to-Zero, Inverted (RZI) modulation scheme which represents logic 0 as an infra light pulse. The modulated output pulse stream is transmitted to an external output driver and infrared Light Emitting Diode.

In Normal mode, the transmitted pulse width is specified as 3/16 period of baud rate.

### 19.4.1.2 IrDA SIR Receive Decoder

The IrDA SIR Receive Decoder demodulates the Return-to-Zero bit stream from the input detector and outputs the NRZ serial bits stream to the UART received data input. The decoder input is normally high in the idle state. (Because of this, IRCR bit 6 should be set as 1 by default)

A start bit is detected when the decoder input is LOW

### 19.4.1.3 IrDA SIR Operation

The IrDA SIR Encoder/Decoder provides functionality which converts between UART data stream and half duplex serial SIR interface. The following diagram is IrDA encoder/decoder waveform:

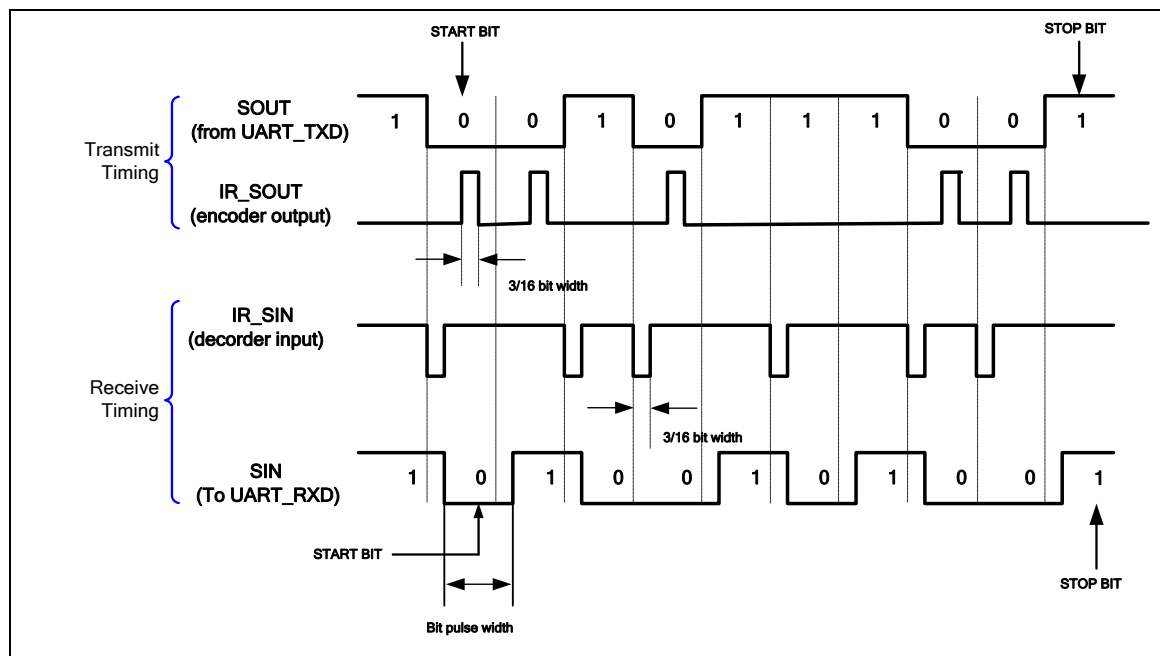


Figure 19-5 IrDA Timing Diagram

## 19.5 LIN (Local Interconnection Network) mode

The UART supports LIN function, and LIN mode is selected by setting the UA\_FUN\_SEL[1:0] to '01'. The UART support LIN break/delimiter generation and break/delimiter detection in LIN master mode, support header detection and automatic resynchronization in LIN slave mode.

### 19.5.1 Structure of LIN Frame

According to the LIN protocol, all information is transmitted packed as frames; a frame consist (provided by the master task) a header and a response (provided by a slave task). That is any communication on the LIN bus is started by the master sending a header, followed by the response. The header (provided by the master task) consists of a break field and sync field followed by a frame identifier (frame ID). The frame identifier uniquely defines the purpose of the frame. The slave task appointed for providing the response associated with the frame ID and the response consists of a data field and a checksum field. The following diagram is the structure of LIN Frame.

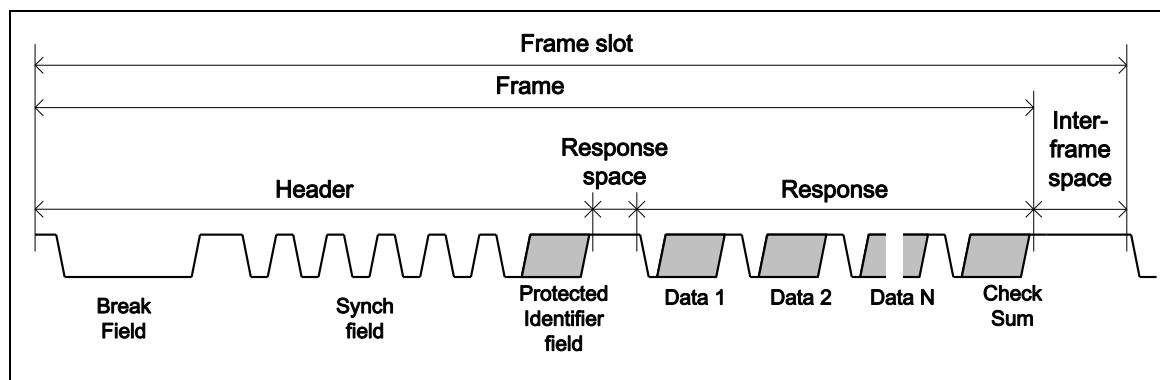


Figure 19-6 Structure of LIN Frame

### 19.5.2 Structure of LIN Byte

In LIN mode, each byte field is initiated by a START bit with value zero (dominant), followed by 8 data bits (UA\_LCR [WLS] = 2'b11) and no parity bit, LSB is first and ended by 1 stop bit (UA\_LCR [NSB] = 1) with value one (recessive) in accordance with the LIN standard. Structure of Byte is shown as follows.

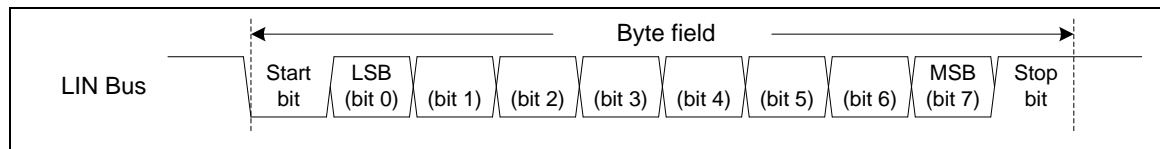


Figure 19-7 Structure of LIN Byte

### 19.5.3 LIN Master Mode

The UART controller support LIN master mode by setting the UA\_FUN\_SEL register. To enable and initialize the LIN master mode, the following steps are necessary.

1. Select the desired baud-rate by setting the UA\_BAUD register.
2. Configure the data length to 8 bits by setting UA\_LCR[WLS] = 11 and disable parity check by clearing UA\_LCR[PBE] bit and configure the stop bit to 1 by clearing UA\_LCR[NSB] bit.

3. Select LIN function mode by setting UA\_FUN\_SEL register.

A complete header consists of a break field and sync field followed by a frame identifier (frame ID). The UART controller can be selected header sending by three header selected mode. The header selected mode can be “break field” or “break field and sync field” or “break field, sync field and frame ID field” by setting LIN\_HEAD\_SEL in UA\_LIN\_CTL register. If the header selected is “break field”, software must handle the following sequence to sending a complete header to bus by filled sync data (0x55) and frame ID data to UA\_THR register. If the header selected is “break field and sync field”, software must handle the sequence to sending a complete header to bus by filled frame ID data to UA\_THR register, and if the header selected is “break field, sync field and frame ID field”, hardware will control the header sending sequence automatically but software must filled frame ID data to UA\_LIN\_CTL [LIN\_PID] register. When operating in header selected is “break field, sync field and frame ID field” mode, the frame ID parity bit can be calculated by software or hardware that depended on UA\_LIN\_CTL [LIN\_IDPEN] bit setting or not.

| LIN_HEAD_SEL | Break field      | Sync field       | ID field   |
|--------------|------------------|------------------|--|
| 0            | Generated by H/W | Handled by S/W   | Handled by S/W   |
| 1            | Generated by H/W | Generated by H/W | Handled by S/W   |
| 2            | Generated by H/W | Generated by H/W | Generated by H/W<br>(But S/W needs to fill ID to UA_LIN_CTL [LIN_PID] first) |

Table 19-3 LIN Header selection in master mode

When operating in LIN data transmission, software can monitor the LIN bus transfer state by hardware or software. User can enable hardware monitoring by setting UA\_LIN\_CTL [BIT\_ERR\_EN] to “1”, if the input pin (SIN) state is not equal to the output pin (SOUT) state in LIN transmitter state that the hardware wills generator an interrupt to CPU. Software also can monitor the LIN bus transfer state by checking the read back data in UA\_RBR register. The following sequence is a program sequence example:

Procedure without software error monitoring in master mode

1. Fill Protected Identifier to [UA\_LIN\_CTL]LIN\_PAD.
2. Choose the hardware transmission header field include “break field + sync field + Protected identifier field” by setting UA\_LIN\_CTL [LIN\_HEAD\_SEL] = 10
3. Choose the hardware transmission header field by setting UA\_LIN\_CTL [LIN\_HEAD\_SEL]).
4. Request header transmission by setting the LIN\_SHD bit in the UA\_LIN\_CTL register.
5. Wait until UA\_LIN\_CTL[LIN\_SHD] be cleared by hardware.
6. Wait until UA\_FSR[TE\_FLAG] set to “1” by hardware

**Note1:** The break field + break/sync delimiter default setting is 13 dominant bits(break field) and 1 recessive bit(break/sync delimiter), software can change it by setting UA\_LIN\_CTL [LIN\_BKFL] and UA\_LIN\_CTL [LIN\_BS\_LEN], to change the dominant bits.

**Note2:** The break/sync delimiter length default setting is 1 bit time and the inter-byte spaces default setting is also 1 bit time, software can change them by setting UA\_LIN\_CTL [LIN\_BS\_LEN] and UA\_TOR [DLY].

**Note3:** If the header includes “break field, sync field and frame ID field”, software must fill frame ID in UA\_LIN\_CTL [LIN\_PID] register before trigger header transmission (setting the LIN\_SHD bit in the UA\_LIN\_CTL register). The frame ID parity can be generated by software or hardware depends on UA\_LIN\_CTL [LIN\_IDPEN]. If the parity generated by software (UA\_LIN\_CTL [LIN\_IDPEN] = 0), software must fill 8 bit data (include 2 bit parity) in this field, and if the parity generated by hardware (UA\_LIN\_CTL [LIN\_IDPEN] = 1), software fill ID0~ID5, hardware will calculi P0 and P1.

#### Procedure with software error monitoring in master mode

1. Choose the hardware transmission header field only include “break field” by setting UA\_LIN\_CTL [LIN\_HEAD\_SEL] = 0x00.
2. Enable break detection function by setting LIN\_BKDET\_EN bit in UA\_LIN\_CTL.
3. Request break + break/sync delimiter transmission by setting the LIN\_SHD bit in the UA\_LIN\_CTL register.
4. Wait until the LIN\_BKDET\_F flag in the UA\_LIN\_SR register be set to “1” by hardware..
5. Request sync field transmission by writing 0x55 into UA\_THR register.
6. Wait until the RDA\_IF flag in the UA\_ISR register be set to “1” by hardware and then read back the UA\_RBR register.
7. Request header frame ID transmission by writing the protected identifier value to UA\_THR register.
8. Wait until the RDA\_IF flag in the UA\_ISR register be set to “1” by hardware and then read back the UA\_RBR register.

### LIN break and delimiter detection

When software enable the break detection function by setting LIN\_BKDET\_EN bit in UA\_LIN\_CTL register, the break detection circuit is activated. The break detection circuit is totally independent from the UART receiver.

When enable break detection function, the circuit looks at the input SIN pin for a start signal. If circuit detected consecutive dominant greater than 11 bits dominant followed by a recessive bit(delimiter), the UA\_LIN\_SR [LIN\_BKDET\_F] flag is set at the end of break field. If the UA\_IER [LIN\_IEN] bit =1, an interrupt will be generated. The behavior of the break detection and break flag is shown in following figure.

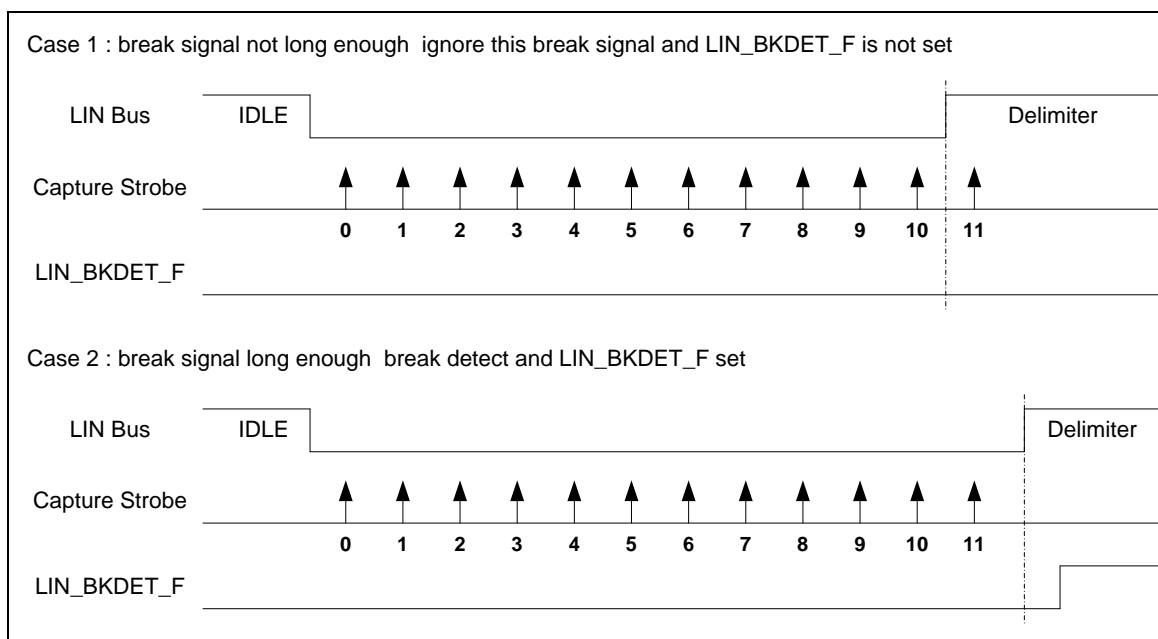


Figure 19-8 Break detection in LIN mode

### LIN break and delimiter detection

The LIN master can transmitter response (master is the publisher of the response) and receive response (master is the subscriber of the response). When the master is the publisher of the response, the master send response by writing UA\_THR register, and if the master is the subscriber of the response, the master will received response from other slave node.

#### 19.5.4 LIN Slave Mode

The UART controller support LIN slave mode by setting the LINS\_EN bit in UA\_LIN\_CTL register. To enable and initialize the LIN slave mode, the following steps are necessary:

1. Select the desired baud-rate by setting the UA\_BAUD register.
2. Configure the data length to 8 bits by setting UA\_LCR[WLS] = 11 and disable parity check by clearing UA\_LCR[PBE] bit and configure the stop bit to 1 by clearing UA\_LCR[NSB] bit.
3. Select LIN function mode by setting UA\_FUN\_SEL register.
4. Enable LIN slave mode by setting the LINS\_EN bit in UA\_LIN\_CTL.

#### LIN header reception

According to the LIN protocol, a slave node must wait for a valid header which came from the master node. Then application will take one of following actions (depend on the master header frame ID value)

- Receive the response.
- Transmit the response.
- Ignore the response and wait for next header.

In LIN slave mode, user can enable slave header detection function by setting LINS\_HDET\_EN bit in UA\_LIN\_CTL register to detect complete frame header (receive “break field”, “sync field” and “frame ID field”). When a LIN header is received, the LINS\_HDET\_F flag in UA\_LIN\_SR register will be set (If the UA\_IER [LIN\_IEN] bit =1, an interrupt will be generated). User can enable frame ID parity check function by setting LIN\_IDPEN bit in UA\_LIN\_CTL register. If only received frame ID parity is not correct (break and sync field are correct), the [UA\_LIN\_SR]LIN\_IDPERR\_F flag (If the UA\_IER [LIN\_IEN] bit =1, an interrupt will be generated) and UA\_LIN\_SR [LINS\_HDET\_F] both will be set. User also can put LIN in mute mode by setting LIN\_MUTE\_EN bit in UA\_LIN\_CTL register. This mode allows detection of headers only (break + sync + frame ID) and prevents the reception of any other characters. In order to avoid bit rate tolerance, the controller support automatic resynchronization function to avoid clock deviation error, user can enable this feature by setting LINS\_ARS\_EN bit in UA\_LIN\_CTL register.

#### LIN response transmission

LIN slave node can transmit response and receive response. When slave node is the publisher of the response, the slave node send response by filling data to UA\_THR register, and if the slave node is the subscriber of the response, the slave node received data from LIN bus.



### LIN header time-out error

The LIN slave controller contains a header time-out counter. If the entire header is not received within the maximum time limit of 57 bit times, the header error flag (UA\_FSR [LIN\_HERR\_F]) will be set. The time-out counter is enabled at each break detect edge and stopped in the following conditions.

- A LIN frame ID field has been received.
- The header error flag assert.
- Software write 1 to LINS\_SYNC\_F bit in UA\_LIN\_SR register to re-search new frame header.

### Mute mode and LIN exit from mute mode condition

In mute mode, LIN slave node will not receive any data until specified condition occurred. It allows detection of headers only and prevents the reception of any other characters. User can enable mute mode by setting LIN\_MUTE\_EN bit in UA\_LIN\_CTL register and exit from mute mode condition can be selected by LIN\_HEAD\_SEL in UA\_LIN\_CTL register.

**Note:** It is recommended to set LIN slave node to mute mode after checksum transmission.

LIN slave controller exit from mute mode conditions is shown as follows: If [UA\_LIN\_CTL] LIN\_HEAD\_SEL is set to “break field”, when LIN slave controller detects a valid LIN break + delimiter, the controller will enable the receiver (exit from mute mode) and subsequent data(sync data, frame ID data, response data) are received in RX-FIFO.

If [UA\_LIN\_CTL]LIN\_HEAD\_SEL is set to “break field and sync field”, when LIN slave controller detects a valid LIN break + delimiter followed by a valid sync field without frame error, the controller will enable the receiver (exit from mute mode) and subsequent data(ID data, response data) are received in RX-FIFO. If [UA\_LIN\_CTL]LIN\_HEAD\_SEL is set to “break field, sync field and ID field”, when LIN slave controller detects a valid LIN break + delimiter and valid sync field without frame error followed by ID data without frame error and received ID data matched UA\_LIN\_CTL [LIN\_PID] value. The controller will enable the receiver (exit from mute mode) and subsequent data(response data) are received in RX-FIFO.

### Slave mode without automatic resynchronization

User can disable automatic resynchronization function to fix the communication baud rate. When operating in without automatic resynchronization mode, software need some initial process, and the initialization process flow of without automatic resynchronization mode is shown as follows:

1. Select the desired baud-rate by setting the UA\_BAUD register.
2. Select LIN function mode by setting UA\_FUN\_SEL register.
3. Disable automatic resynchronization function by setting UA\_LIN\_CTL[LINS\_ARS\_EN] = 0.
4. Enable LIN slave mode by setting the LINS\_EN bit in UA\_LIN\_CTL.

### Slave mode with automatic resynchronization

User can enable automatic resynchronization function by setting LINS\_ARS\_EN bit in UA\_LIN\_CTL register. In automatic resynchronization mode, the controller will adjust the baud rate generator after each sync field reception. The initialization process flow of automatic resynchronization mode is shown as follows:

1. Select the desired baud-rate by setting the UA\_BAUD register.
2. Select LIN function mode by setting UA\_FUN\_SEL register.
3. Enable automatic resynchronization function by setting UA\_LIN\_CTL[LINS\_ARS\_EN] = 1.
4. Enable LIN slave mode by setting the LINS\_EN bit in UA\_LIN\_CTL.

When automatic resynchronization function is enabled, after each LIN break field, the time duration between five falling edges is sampled on engine clock and the result of this measurement is stored in an internal 13-bit register and the UA\_BAUD register value will automatically updated at the end of the fifth falling edge. If the measure timer (13bit) overflow before five falling edges, then the header error flag (UA\_LIN\_SR [LIN\_HERR\_F]) will be set.

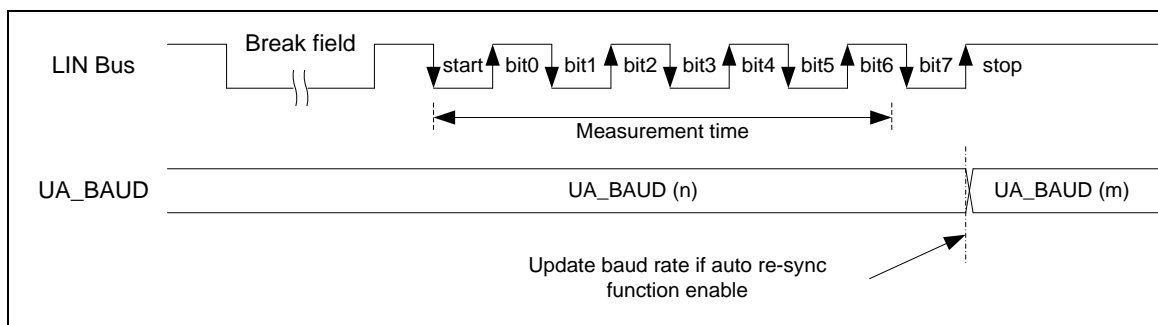


Figure 19-9 LIN sync field measurement

When operating in automatic resynchronization mode, software must select the desired baud rate by setting the UA\_BAUD register and hardware will store it at internal TEMP\_REG register, after each LIN break field, the time duration between five falling edges is sampled on engine clock and the result of this measurement is stored in an internal 13-bit register (BAUD\_LIN) and the result will be updated to UA\_BAUD register automatically.

In order to guarantee the transmission baud rate, the baud rate generator must reload the initial value before each new break reception. The initial value is programmed by the application during initialization (TEMP\_REG). User can setting UA\_LIN\_CTL [LINS\_DUM\_EN] bit to enable auto reload initial baud rate value function. If the LINS\_DUM\_EN is set, when received the next character, hardware will auto reload the initial value to UA\_BAUD, and when the UA\_BAUD be updated, the LINS\_DUM\_EN bit will be cleared automatically. The behavior of LIN updated method as shown as following figure.

**Note1:** It is recommended setting the LINS\_DUM\_EN bit before every checksum reception.

**Note2:** When header error been detected, user must writing 1 to LINS\_SYNC\_F bit in UA\_LIN\_SR register to re-search new frame header. When user writing 1 to it, hardware will reload the initial baud-rate (TEMP\_REG) and re-search new frame header.

**Note3:** When operation in automatic resynchronization mode, the baud rate setting must be mode2 (UA\_BAUD [DIV\_X\_EN] and UA\_BAUD [DIV\_X\_ONE] must be 1).

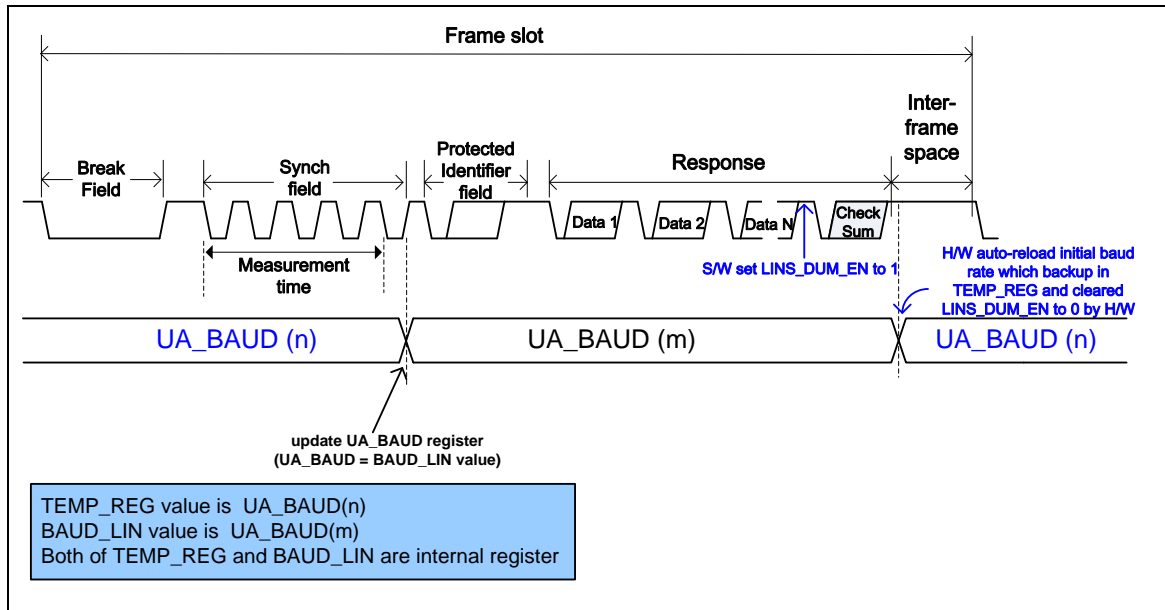


Figure 19-10 UA\_BAUD update sequence in automatic resynchronization mode (LINS\_DUM\_EN = 1)

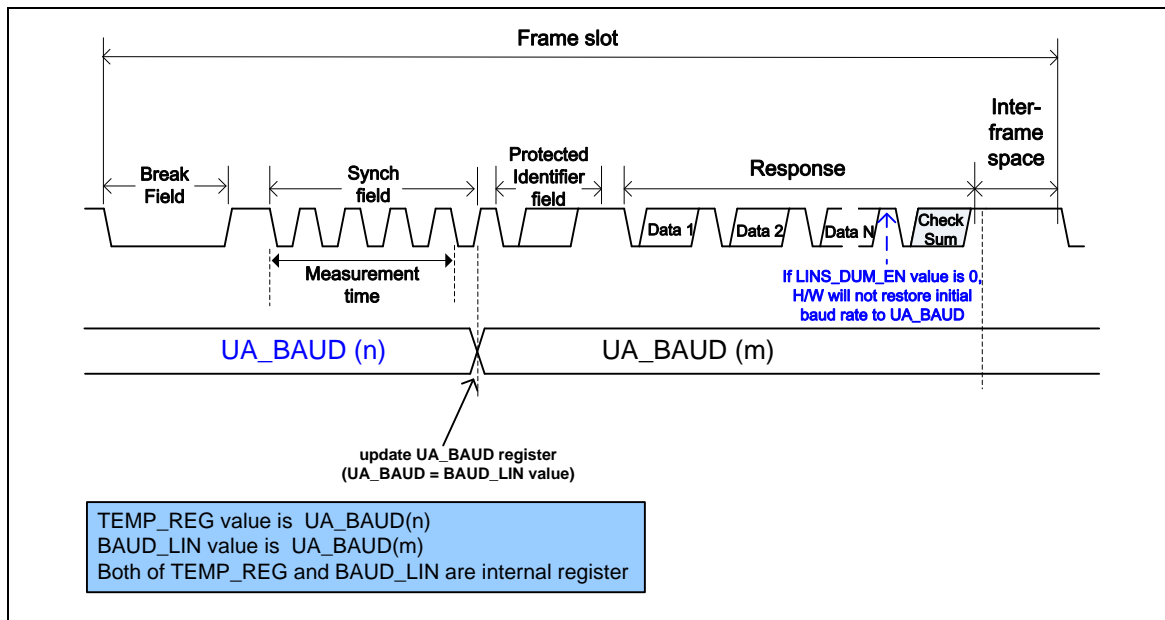


Figure 19-11 UA\_BAUD update sequence in automatic resynchronization mode (LINS\_DUM\_EN = 0)

### Deviation error on the sync field

When operating in automatic resynchronization mode, the controller will check the deviation error on the sync field. The deviation error is checked by comparing the current baud rate with the received sync field. Two checks are performed in parallel.

Check1: Based on measurement between the first falling edge and the last falling edge of the sync field.

- If the difference more than 14.84%, the header error flag UA\_LIN\_SR[LINS\_HERR\_F]) will be set.
- If the difference less than 14.06%, the header error flag UA\_LIN\_SR[LINS\_HERR\_F]) will not be set.
- If the difference between 14.84% and 14.06%, the header error flag UA\_LIN\_SR[LINS\_HERR\_F]) may either set or not (it is depending on the data dephasing).

Check2: Based on measurement of time between each falling edge of the sync field.

- If the difference more than 18.75%, the header error flag UA\_LIN\_SR[LINS\_HERR\_F]) will be set.
- If the difference less than 15.62%, the header error flag UA\_LIN\_SR[LINS\_HERR\_F]) will not be set.
- If the difference between 18.75% and 15.62%, the header error flag UA\_LIN\_SR[LINS\_HERR\_F]) may either set or not (it is depending on the data dephasing).

**Note:** The deviation check is based on the current baud-rate clock. Therefore, in order to guarantee correct deviation checking, the baud-rate must reload the nominal value before each new break reception by setting UA\_LIN\_CTL [LINS\_DUM\_EN] register (It is recommend setting the LINS\_DUM\_EN bit before every checksum reception)

### LIN header error detection

In LIN slave function mode, when user enables header detection function by setting LINS\_HDET\_EN bit in UA\_LIN\_CTL register, the hardware will handle the header detect flow. If the header has error, the LIN header error flag (UA\_LIN\_SR [LIN\_HERR\_F]) will be set and an interrupt is generated if the UA\_IER [LIN\_IEN] bit is set. When header error been detect, user must to reset the detect circuit to re-search new frame header by writing 1 to LINS\_SYNC\_F bit in UA\_LIN\_SR register to re-search new frame header.

The LIN header error flag (UA\_LIN\_SR [LIN\_HERR\_F]) is set if one of the following conditions occurs:

- Break Delimiter is too short (less than 0.5 bit time).
- Frame error in sync field or Identifier field.
- The sync field data is not 0x55 (without automatic resynchronization mode).
- The sync field deviation error (with automatic resynchronization mode).
- The sync field measure time-out (with automatic resynchronization mode).

LIN header reception time-out

## 19.6 RS-485 function mode

The UART support **RS-485 9-bit mode function**. The RS-485 mode is selected by setting the register UA\_FUN\_SEL[1:0]. The RS-485 transceiver control is implemented using the nRTS control signal from an asynchronous serial port. In RS-485 mode, many characteristics of the RX and TX are same as UART.

In RS-485 mode, the controller can configuration of it as an RS-485 addressable slave and the RS-485 master transmitter will identify an address character by setting the parity (9<sup>th</sup> bit) to 1. For data characters, the parity is set to 0. Software can use UA\_LCR register to control the 9-th bit (When the PBE, EPE and SPE are set, the 9-th bit is transmitted 0 and when PBE and SPE are set and EPE is cleared, the 9-th bit is transmitted 1).

The controller supports three operation modes that are RS-485 Normal Multidrop Operation Mode (NMM), RS-485 Auto Address Detection Operation Mode (AAD) and RS-485 Auto Direction Control Operation Mode (AUD). Software can choose any operation mode by programming UA\_ALT\_CSR register, and software can driving the transfer delay time between the last stop bit leaving the TX-FIFO and the de-assertion of by setting UA\_TOR [DLY] register.

**Note:** when RS485 NMM or AAD mode is selected, the RS485 clock operating frequency should be less than or equal to half of PCLK clock operation frequency. Otherwise, RS485 cannot receive correct data.

### RS-485 Normal Multidrop Operation Mode (NMM)

In RS-485 Normal Multidrop operation mode, in first, software must decide the data which before the address byte be detected will be stored in RX-FIFO or not. If software want to ignore any data before address byte detected, the flow is set UA\_FCR[RX\_DIS] and then enable UA\_ALT\_CSR [RS485\_NMM] and the receiver will ignore any data until an address byte is detected (bit9 =1) and the address byte data will be stored in the RX-FIFO. If software wants to receive any data before address byte detected, the flow disables UA\_FCR[RX\_DIS] and then enables UA\_ALT\_CSR[RS485\_NMM] and the receiver will received any data.

If an address byte is detected (bit9 =1), it will generate an interrupt to CPU and UA\_FCR[RX\_DIS] can decide whether accept the following data bytes are stored in the RX-FIFO. If software disable receiver by setting UA\_FCR[RX\_DIS] register, when a next address byte be detected, the controller will clear the UA\_FCR[RX\_DIS] bit and the address byte data will be stored in the RX-FIFO.

### RS-485 Auto Address Detection Operation Mode (AAD)

In RS-485 Auto Address Detection Operation mode, the receiver will ignore any data until an address byte is detected (bit9 =1) and the address byte data match the UA\_ALT\_CSR [ADDR\_MATCH] value. The address byte data will be stored in the RX-FIFO. The all received byte data will be accepted and stored in the RX-FIFO until and address byte data not match the UA\_ALT\_CSR[ADDR\_MATCH] value.

### RS-485 Auto Direction Mode (AUD)

Another option function of RS-485 controllers is **RS-485 auto direction control function**. The RS-485 transceiver control is implemented using the nRTS control signal from an asynchronous serial port. The nRTS line is connected to the RS-485 transceiver enable pin such that setting the nRTS line to high (logic 1) enables the RS-485 transceiver. Setting the nRTS line to low (logic 0) puts the transceiver into the tri-state condition to disable. User can setting LEV\_RTS in UA\_MCR register to change the nRTS driving level.

**Program Sequence Example:**

1. Program FUN\_SEL in UA\_FUN\_SEL to select RS-485 function.
2. Program the RX\_DIS bit in UA\_FCR register to determine enable or disable the receiver of RS-485 controller.
3. Program the RS-485\_NMM or RS-485\_AAD mode.
4. If the RS-485\_AAD mode is selected, the ADDR\_MATCH is programmed for auto address match value.
5. Determine auto direction control by programming RS-485\_AUD.

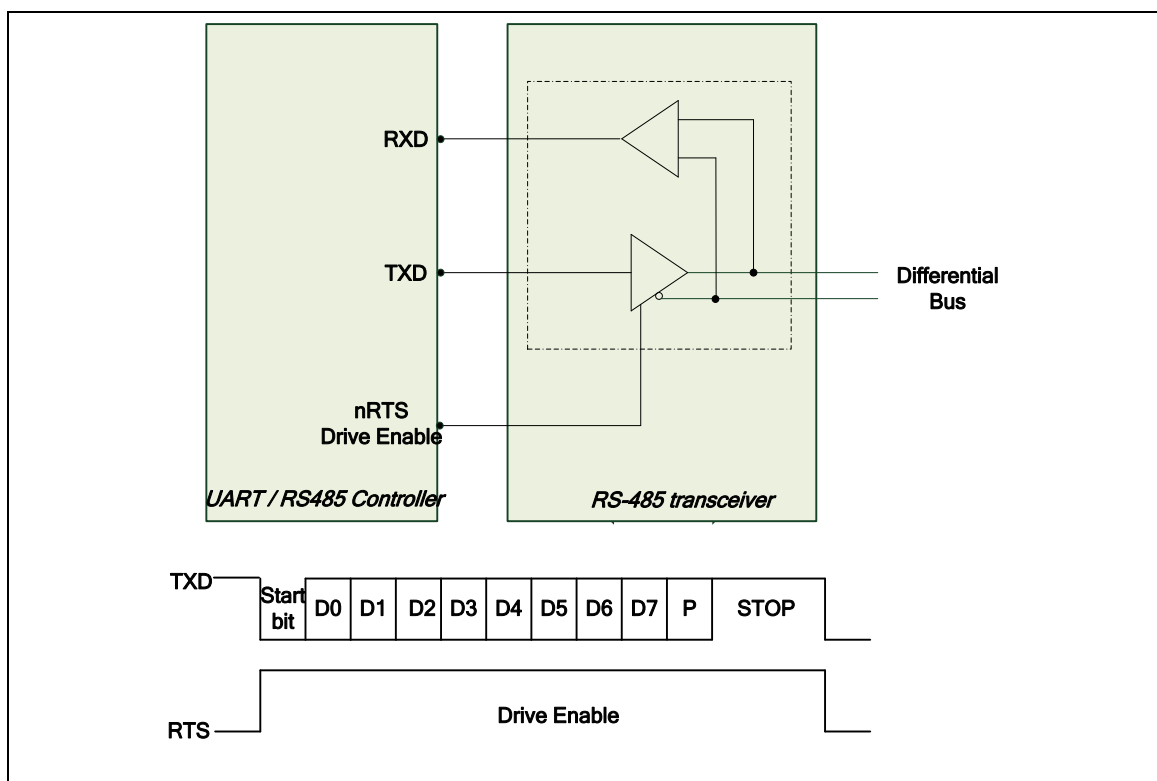


Figure 19-12 Structure of RS-485 Frame

### 19.7 Register Map

R: read only, W: write only, R/W: both read and write

| Register  | Offset        | R/W | Description                            | Reset Value |
|---|---------------|-----|--|-------------|
| <b>UART Base Address:</b><br>$\text{UARTx\_BA} = 0x4005\_0000 + (0x10\_0000 * x)$<br>$x = 0, 1$ |               |     |  |             |
| <b>UA_RBR</b>   | UARTx_BA+0x00 | R   | UART Receive Buffer Register           | Undefined   |
| <b>UA_THR</b>   | UARTx_BA+0x00 | W   | UART Transmit Holding Register         | Undefined   |
| <b>UA_IER</b>   | UARTx_BA+0x04 | R/W | UART Interrupt Enable Register         | 0x0000_0000 |
| <b>UA_FCR</b>   | UARTx_BA+0x08 | R/W | UART FIFO Control Register             | 0x0000_0000 |
| <b>UA_LCR</b>   | UARTx_BA+0x0C | R/W | UART Line Control Register             | 0x0000_0000 |
| <b>UA_MCR</b>   | UARTx_BA+0x10 | R/W | UART Modem Control Register            | 0x0000_0200 |
| <b>UA_MSR</b>   | UARTx_BA+0x14 | R/W | UART Modem Status Register             | 0x0000_0110 |
| <b>UA_FSR</b>   | UARTx_BA+0x18 | R/W | UART FIFO Status Register              | 0x1040_4000 |
| <b>UA_ISR</b>   | UARTx_BA+0x1C | R/W | UART Interrupt Status Register         | 0x0000_0002 |
| <b>UA_TOR</b>   | UARTx_BA+0x20 | R/W | UART Time Out Register                 | 0x0000_0000 |
| <b>UA_BAUD</b>  | UARTx_BA+0x24 | R/W | UART Baud Rate Divisor Register        | 0x0F00_0000 |
| <b>UA_IRCR</b>  | UARTx_BA+0x28 | R/W | UART IrDA Control Register             | 0x0000_0040 |
| <b>UA_ALT_CSR</b>   | UARTx_BA+0x2C | R/W | UART Alternate Control/Status Register | 0x0000_0000 |
| <b>UA_FUN_SEL</b>   | UARTx_BA+0x30 | R/W | UART Function Select Register          | 0x0000_0000 |
| <b>UA_LIN_CTL</b>   | UARTx_BA+0x34 | R/W | UART LIN Control Register              | 0x000C_0000 |
| <b>UA_LIN_SR</b>  | UARTx_BA+0x38 | R/W | UART LIN Status Register               | 0x0000_0000 |

### 19.8 Register Description

#### Receive Buffer Register (UA\_RBR)

| Register | Offset        | R/W | Description                  | Reset Value |
|----------|---------------|-----|------------------------------|-------------|
| UA_RBR   | UARTx_BA+0x00 | R   | UART Receive Buffer Register | Undefined   |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| RBR      |    |    |    |    |    |    |    |

| Bits   | Description |  |
|--------|-------------|--|
| [31:8] | Reserved    | Reserved   |
| [7:0]  | RBR         | <b>Receive Buffer Register (Read Only)</b><br>By reading this register, the UART will return an 8-bit data received from UART_RXD pin (LSB first). |



### Transmit Holding Register (UA\_THR)

| Register | Offset        | R/W | Description                    | Reset Value |
|----------|---------------|-----|--------------------------------|-------------|
| UA_THR   | UARTx_BA+0x00 | W   | UART Transmit Holding Register | Undefined   |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| THR      |    |    |    |    |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | Reserved  |
| [7:0]  | THR         | <b>Transmit Holding Register</b><br>By writing to this register, the UART will send out an 8-bit data through the UART_TXD pin (LSB first). |

**Interrupt Enable Register (UA\_IER)**

| Register | Offset        | R/W | Description                    | Reset Value |
|----------|---------------|-----|--------------------------------|-------------|
| UA_IER   | UARTx_BA+0x04 | R/W | UART Interrupt Enable Register | 0x0000_0000 |

|          |         |             |             |             |          |          |         |
|----------|---------|-------------|-------------|-------------|----------|----------|---------|
| 31       | 30      | 29          | 28          | 27          | 26       | 25       | 24      |
| Reserved |         |             |             |             |          |          |         |
| 23       | 22      | 21          | 20          | 19          | 18       | 17       | 16      |
| Reserved |         |             |             |             |          |          |         |
| 15       | 14      | 13          | 12          | 11          | 10       | 9        | 8       |
| Reserved |         | AUTO_CTS_EN | AUTO_RTS_EN | TIME_OUT_EN | Reserved |          | LIN_IEN |
| 7        | 6       | 5           | 4           | 3           | 2        | 1        | 0       |
| Reserved | WAKE_EN | BUF_ERR_IEN | TOUT_IEN    | MODEM_IEN   | RLS_IEN  | THRE_IEN | RDA_IEN |

| Bits    | Description |  |
|---------|-------------|--|
| [31:14] | Reserved    | Reserved   |
| [13]    | AUTO_CTS_EN | <b>nCTS Auto Flow Control Enable</b><br>1 = nCTS auto flow control Enabled<br>0 = nCTS auto flow control Disabled<br>When nCTS auto-flow is enabled, the UART will send data to external device when nCTS input assert (UART will not send data to device until nCTS is asserted). |
| [12]    | AUTO_RTS_EN | <b>nRTS Auto Flow Control Enable</b><br>1 = nRTS auto flow control Enabled<br>0 = nRTS auto flow control Disabled<br>When nRTS auto-flow is enabled, if the number of bytes in the RX FIFO equals the UA_FCR [RTS_TRI_LEV], the UART will de-assert nRTS signal.                   |
| [11]    | TIME_OUT_EN | <b>Time Out Counter Enable</b><br>1 = Time out counter Enabled<br>0 = Time out counter Disabled  |
| [10:9]  | Reserved    | Reserved   |
| [8]     | LIN_IEN     | <b>LIN Bus Interrupt Enable</b><br>1 = Lin bus interrupt Enabled<br>0 = Lin bus interrupt Disabled<br><b>Note:</b> This field is used for LIN function mode.   |
| [7]     | Reserved    | Reserved   |
| [6]     | WAKE_EN     | <b>UART Wake-up Function Enable</b><br>1 = UART wake-up function Enabled, when the chip is in Power-down mode, an external nCTS change will wake-up chip from Power-down mode.   |

|     |                    |  |
|-----|--------------------|--|
|     |                    | 0 = UART wake-up function Disabled   |
| [5] | <b>BUF_ERR_IEN</b> | <b>Buffer Error Interrupt Enable</b><br>1 = INT_BUF_ERR Enabled<br>0 = Mask off INT_BUF_ERR Masked off     |
| [4] | <b>TOUT_IEN</b>    | <b>RX Time Out Interrupt Enable</b><br>1 = INT_TOUT Enabled<br>0 = INT_TOUT Masked off                     |
| [3] | <b>MODEM_IEN</b>   | <b>Modem Status Interrupt Enable</b><br>1 = INT_MODEM Enabled<br>0 = INT_MODEM Masked off                  |
| [2] | <b>RLS_IEN</b>     | <b>Receive Line Status Interrupt Enable</b><br>1 = INT_RLS Enabled<br>0 = INT_RLS Masked off               |
| [1] | <b>THRE_IEN</b>    | <b>Transmit Holding Register Empty Interrupt Enable</b><br>1 = INT_THRE Enabled<br>0 = INT_THRE Masked off |
| [0] | <b>RDA_IEN</b>     | <b>Receive Data Available Interrupt Enable</b><br>1 = INT_RDA Enabled<br>0 = INT_RDA Masked off            |

### FIFO Control Register (UA\_FCR)

| Register | Offset        | R/W | Description                | Reset Value |
|----------|---------------|-----|----------------------------|-------------|
| UA_FCR   | UARTx_BA+0x08 | R/W | UART FIFO Control Register | 0x0000_0000 |

|          |    |    |    |             |     |     |          |
|----------|----|----|----|-------------|-----|-----|----------|
| 31       | 30 | 29 | 28 | 27          | 26  | 25  | 24       |
| Reserved |    |    |    |             |     |     |          |
| 23       | 22 | 21 | 20 | 19          | 18  | 17  | 16       |
| Reserved |    |    |    | RTS_TRI_LEV |     |     |          |
| 15       | 14 | 13 | 12 | 11          | 10  | 9   | 8        |
| Reserved |    |    |    |             |     |     | RX_DIS   |
| 7        | 6  | 5  | 4  | 3           | 2   | 1   | 0        |
| RFITL    |    |    |    | Reserved    | TFR | RFR | Reserved |

| Bits        | Description  |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
|-------------|--|-------------|--------------------------------|------|----|------|----|------|----|------|----|------|---------------------------------|------|---------------------------------|------|---------------------------------|--------|---------------------------------|
| [31:20]     | Reserved   |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| [19:16]     | <p><b>RTS_TRI_LEV</b></p> <p>nRTS Trigger Level for Auto-flow Control Use</p> <table> <tr> <th>RTS_TRI_LEV</th><th>Trigger Level (Bytes)</th></tr> <tr> <td>0000</td><td>01</td></tr> <tr> <td>0001</td><td>04</td></tr> <tr> <td>0010</td><td>08</td></tr> <tr> <td>0011</td><td>14</td></tr> <tr> <td>0100</td><td>30/14 (High Speed/Normal Speed)</td></tr> <tr> <td>0101</td><td>46/14 (High Speed/Normal Speed)</td></tr> <tr> <td>0110</td><td>62/14 (High Speed/Normal Speed)</td></tr> <tr> <td>others</td><td>62/14 (High Speed/Normal Speed)</td></tr> </table> <p><b>Note:</b> This field is used for auto nRTS flow control.</p> | RTS_TRI_LEV | Trigger Level (Bytes)          | 0000 | 01 | 0001 | 04 | 0010 | 08 | 0011 | 14 | 0100 | 30/14 (High Speed/Normal Speed) | 0101 | 46/14 (High Speed/Normal Speed) | 0110 | 62/14 (High Speed/Normal Speed) | others | 62/14 (High Speed/Normal Speed) |
| RTS_TRI_LEV | Trigger Level (Bytes)  |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| 0000        | 01   |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| 0001        | 04   |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| 0010        | 08   |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| 0011        | 14   |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| 0100        | 30/14 (High Speed/Normal Speed)  |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| 0101        | 46/14 (High Speed/Normal Speed)  |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| 0110        | 62/14 (High Speed/Normal Speed)  |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| others      | 62/14 (High Speed/Normal Speed)  |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| [15:9]      | Reserved   |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| [8]         | <p><b>RX_DIS</b></p> <p><b>Receiver Disable Register.</b></p> <p>The receiver is disabled or not (set 1 to disable receiver)</p> <p>1 = Receiver Disabled</p> <p>0 = Receiver Enabled</p> <p><b>Note:</b> This field is used for RS-485 Normal Multi-drop mode. It should be programmed before UA_ALT_CSR [RS-485_NMM] is programmed.</p>  |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| [7:4]       | <p><b>RFITL</b></p> <p><b>RX FIFO Interrupt (INT_RDA) Trigger Level</b></p> <p>When the number of bytes in the receive FIFO equals the RFITL, the RDA_IF will be set (if UA_IER [RDA_IEN] enabled, and an interrupt will be generated).</p> <table> <tr> <th>RFITL</th><th>INTR_RDA Trigger Level (Bytes)</th></tr> </table>   | RFITL       | INTR_RDA Trigger Level (Bytes) |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |
| RFITL       | INTR_RDA Trigger Level (Bytes)   |             |                                |      |    |      |    |      |    |      |    |      |                                 |      |                                 |      |                                 |        |                                 |

|     |          |  |                                 |
|-----|----------|--|---------------------------------|
|     |          | 0000   | 01                              |
|     |          | 0001   | 04                              |
|     |          | 0010   | 08                              |
|     |          | 0011   | 14                              |
|     |          | 0100   | 30/14 (High Speed/Normal Speed) |
|     |          | 0101   | 46/14 (High Speed/Normal Speed) |
|     |          | 0110   | 62/14 (High Speed/Normal Speed) |
|     |          | others   | 62/14 (High Speed/Normal Speed) |
| [3] | Reserved | Reserved   |                                 |
| [2] | TFR      | <b>TX Field Software Reset</b><br>When TX_RST is set, all the byte in the transmit FIFO and TX internal state machine are cleared.<br>1 = Reset the TX internal state machine and pointers.<br>0 = No effect.<br><b>Note:</b> This bit will automatically clear at least 3 UART engine clock cycles. |                                 |
| [1] | RFR      | <b>RX Field Software Reset</b><br>When RX_RST is set, all the byte in the receiver FIFO and RX internal state machine are cleared.<br>1 = Reset the RX internal state machine and pointers.<br>0 = No effect.<br><b>Note:</b> This bit will automatically clear at least 3 UART engine clock cycles. |                                 |
| [0] | Reserved | Reserved   |                                 |

**Line Control Register (UA\_LCR)**

| Register | Offset        | R/W | Description                | Reset Value |
|----------|---------------|-----|----------------------------|-------------|
| UA_LCR   | UARTx_BA+0x0C | R/W | UART Line Control Register | 0x0000_0000 |

|          |     |     |     |     |     |     |    |
|----------|-----|-----|-----|-----|-----|-----|----|
| 31       | 30  | 29  | 28  | 27  | 26  | 25  | 24 |
| Reserved |     |     |     |     |     |     |    |
| 23       | 22  | 21  | 20  | 19  | 18  | 17  | 16 |
| Reserved |     |     |     |     |     |     |    |
| 15       | 14  | 13  | 12  | 11  | 10  | 9   | 8  |
| Reserved |     |     |     |     |     |     |    |
| 7        | 6   | 5   | 4   | 3   | 2   | 1   | 0  |
| Reserved | BCB | SPE | EPE | PBE | NSB | WLS |    |

| Bits     | Description      |   |          |                  |    |       |
|----------|------------------|---|----------|------------------|----|-------|
| [31:7]   | Reserved         | Reserved  |          |                  |    |       |
| [6]      | BCB              | <b>Break Control Bit</b><br>When this bit is set to logic 1, the serial data output (TX) is forced to the Spacing State (logic 0). This bit acts only on TX and has no effect on the transmitter logic.   |          |                  |    |       |
| [5]      | SPE              | <b>Stick Parity Enable</b><br>1 = If PBE (UA_LCR[3]) and EBE (UA_LCR[4]) are logic 1, the parity bit is transmitted and checked as logic 0. If PBE (UA_LCR[3]) is 1 and EBE (UA_LCR[4]) is 0 then the parity bit is transmitted and checked as 1<br>0 = Stick parity Disabled   |          |                  |    |       |
| [4]      | EPE              | <b>Even Parity Enable</b><br>1 = Even number of logic 1's is transmitted and checked in each word<br>0 = Odd number of logic 1's is transmitted and checked in each word<br>This bit has effect only when PBE (UA_LCR[3]) is set.   |          |                  |    |       |
| [3]      | PBE              | <b>Parity Bit Enable</b><br>1 = Parity bit is generated on each outgoing character and is checked on each incoming data.<br>0 = No parity bit.  |          |                  |    |       |
| [2]      | NSB              | <b>Number of “STOP bit”</b><br>1 = When select 5-bit word length, 1.5 “STOP bit” is generated in the transmitted data. When select 6-, 7- and 8-bti word length, 2 “STOP bit” is generated in the transmitted data.<br>0 = One “ STOP bit” is generated in the transmitted data |          |                  |    |       |
| [1:0]    | WLS              | <b>Word Length Selection</b> <table><tr><th>WLS[1:0]</th><th>Character Length</th></tr><tr><td>00</td><td>5-bit</td></tr></table>   | WLS[1:0] | Character Length | 00 | 5-bit |
| WLS[1:0] | Character Length |   |          |                  |    |       |
| 00       | 5-bit            |   |          |                  |    |       |

|  |  |    |       |  |
|--|--|----|-------|--|
|  |  | 01 | 6-bit |  |
|  |  | 10 | 7-bit |  |
|  |  | 11 | 8-bit |  |

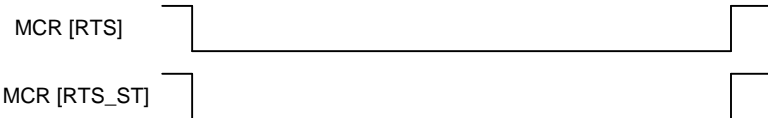
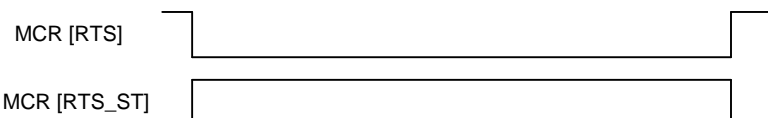
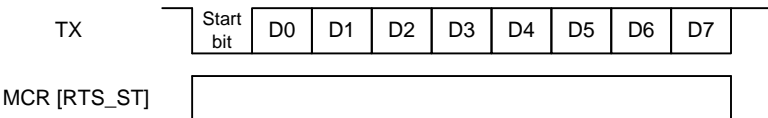
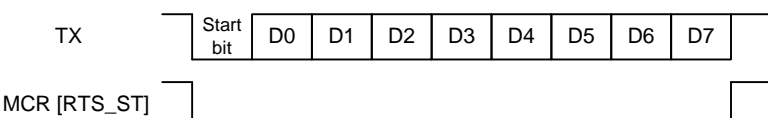
### MODEM Control Register (UA\_MCR)

| Register | Offset        | R/W | Description                 | Reset Value |
|----------|---------------|-----|-----------------------------|-------------|
| UA_MCR   | UARTx_BA+0x10 | R/W | UART Modem Control Register | 0x0000_0200 |

|          |    |        |          |    |    |         |          |
|----------|----|--------|----------|----|----|---------|----------|
| 31       | 30 | 29     | 28       | 27 | 26 | 25      | 24       |
| Reserved |    |        |          |    |    |         |          |
| 23       | 22 | 21     | 20       | 19 | 18 | 17      | 16       |
| Reserved |    |        |          |    |    |         |          |
| 15       | 14 | 13     | 12       | 11 | 10 | 9       | 8        |
| Reserved |    | RTS_ST | Reserved |    |    | LEV_RTS | Reserved |
| 7        | 6  | 5      | 4        | 3  | 2  | 1       | 0        |
| Reserved |    |        |          |    |    | RTS     | Reserved |

| Bits    | Description |  |
|---------|-------------|--|
| [31:14] | Reserved    | Reserved   |
| [13]    | RTS_ST      | nRTS Pin State (Read Only)<br>This bit is the output pin status of nRTS. |
| [12:10] | Reserved    | Reserved   |



|       |          |   |
|-------|----------|---|
| [9]   | LEV_RTS  | <p><b>nRTS Trigger Level</b></p> <p>This bit can change the nRTS trigger level.</p> <p>1 = High level triggered</p> <p>0 = Low level triggered</p> <p><u>UART Mode : MCR[LEV_RTS] = 1</u></p>  <p><u>UART Mode : MCR[LEV_RTS] = 0</u></p>  <p><u>RS-485 Mode : MCR[LEV_RTS] = 0</u></p>  <p><u>RS-485 Mode : MCR[LEV_RTS] = 1</u></p>  |
| [8:2] | Reserved | Reserved  |
| [1]   | RTS      | <p><b>nRTS (Request-To-Send) Signal</b></p> <p>0 = Drive nRTS pin to logic 1 (If the <b>LEV_RTS</b> set to low level triggered).</p> <p>1 = Drive nRTS pin to logic 0 (If the <b>LEV_RTS</b> set to low level triggered).</p> <p>0 = Drive nRTS pin to logic 0 (If the <b>LEV_RTS</b> set to high level triggered).</p> <p>1 = Drive nRTS pin to logic 1 (If the <b>LEV_RTS</b> set to high level triggered).</p>   |
| [0]   | Reserved | Reserved  |

### Modem Status Register (UA\_MSR)

| Register | Offset        | R/W | Description                | Reset Value |
|----------|---------------|-----|----------------------------|-------------|
| UA_MSR   | UARTx_BA+0x14 | R/W | UART Modem Status Register | 0x0000_0110 |

|          |    |    |        |          |    |    |         |
|----------|----|----|--------|----------|----|----|---------|
| 31       | 30 | 29 | 28     | 27       | 26 | 25 | 24      |
| Reserved |    |    |        |          |    |    |         |
| 23       | 22 | 21 | 20     | 19       | 18 | 17 | 16      |
| Reserved |    |    |        |          |    |    |         |
| 15       | 14 | 13 | 12     | 11       | 10 | 9  | 8       |
| Reserved |    |    |        |          |    |    | LEV_CTS |
| 7        | 6  | 5  | 4      | 3        | 2  | 1  | 0       |
| Reserved |    |    | CTS_ST | Reserved |    |    | DCTS_F  |

| Bits   | Description |  |
|--------|-------------|--|
| [31:9] | Reserved    | Reserved   |
| [8]    | LEV_CTS     | <b>nCTS Trigger Level</b><br>This bit can change the nCTS trigger level.<br>1 = High level triggered<br>0 = Low level triggered  |
| [7:5]  | Reserved    | Reserved   |
| [4]    | CTS_ST      | <b>nCTS Pin Status (Read Only)</b><br>This bit is the pin status of nCTS.  |
| [3:1]  | Reserved    | Reserved   |
| [0]    | DCTS_F      | <b>Detect nCTS State Change Flag (Read Only)</b><br>This bit is set whenever nCTS input has change state, and it will generate Modem interrupt to CPU when UA_IER [MODEM_IEN] is set to 1.<br>Write 1 to clear this bit to 0 |

### FIFO Status Register (UA\_FSR)

| Register | Offset        | R/W | Description               | Reset Value |
|----------|---------------|-----|---------------------------|-------------|
| UA_FSR   | UARTx_BA+0x18 | R/W | UART FIFO Status Register | 0x1040_4000 |

| 31       | 30       | 29         | 28      | 27             | 26       | 25 | 24         |
|----------|----------|------------|---------|----------------|----------|----|------------|
| Reserved |          |            | TE_FLAG | Reserved       |          |    | TX_OVER_IF |
| 23       | 22       | 21         | 20      | 19             | 18       | 17 | 16         |
| TX_FULL  | TX_EMPTY | TX_POINTER |         |                |          |    |            |
| 15       | 14       | 13         | 12      | 11             | 10       | 9  | 8          |
| RX_FULL  | RX_EMPTY | RX_POINTER |         |                |          |    |            |
| 7        | 6        | 5          | 4       | 3              | 2        | 1  | 0          |
| Reserved | BIF      | FEF        | PEF     | RS485_ADD_DETF | Reserved |    | RX_OVER_IF |

| Bits    | Description  |
|---------|--|
| [31:29] | <b>Reserved</b><br>Reserved  |
| [28]    | <b>TE_FLAG</b><br><b>Transmitter Empty Flag (Read Only)</b><br>Bit is set by hardware when TX FIFO (UA_THR) is empty and the STOP bit of the last byte has been transmitted.<br>Bit is cleared automatically when TX FIFO is not empty or the last byte transmission has not completed.  |
| [27:25] | <b>Reserved</b><br>Reserved  |
| [24]    | <b>TX_OVER_IF</b><br><b>TX Overflow Error Interrupt Flag (Read Only)</b><br>If TX FIFO (UA_THR) is full, an additional write to UA_THR will cause this bit to logic 1.<br><b>Note:</b> This bit is read only, but can be cleared by writing '1' to it.   |
| [23]    | <b>TX_FULL</b><br><b>Transmitter FIFO Full (Read Only)</b><br>This bit indicates TX FIFO is full or not.<br>This bit is set when the number of usage in TX FIFO Buffer is equal to 16, otherwise is cleared by hardware.   |
| [22]    | <b>TX_EMPTY</b><br><b>Transmitter FIFO Empty (Read Only)</b><br>This bit indicates TX FIFO is empty or not.<br>When the last byte of TX FIFO has been transferred to Transmitter Shift Register, hardware sets this bit high. It will be cleared when writing data into THR (TX FIFO not empty).   |
| [21:16] | <b>TX_POINTER</b><br><b>TX FIFO Pointer (Read Only)</b><br>This field indicates the TX FIFO Buffer Pointer. When CPU writes one byte into UA_THR, then TX_POINTER increases one. When one byte of TX FIFO is transferred to Transmitter Shift Register, then TX_POINTER decreases one.<br>The Maximum value shown in TX_POINTER is 15. When the using level of TX FIFO Buffer equal to 16, the TX_FULL bit is set to 1 and TX_POINTER will show 0. As one byte of TX FIFO is transferred to Transmitter Shift Register, the TX_FULL bit is clear to 0 and TX_POINTER will show 15. |

|        |               |   |
|--------|---------------|---|
| [15]   | RX_FULL       | <b>Receiver FIFO Full (Read Only)</b><br>This bit initiates RX FIFO is full or not.<br>This bit is set when the number of usage in RX FIFO Buffer is equal to 16, otherwise is cleared by hardware.   |
| [14]   | RX_EMPTY      | <b>Receiver FIFO Empty (Read Only)</b><br>This bit initiate RX FIFO empty or not.<br>When the last byte of RX FIFO has been read by CPU, hardware sets this bit high. It will be cleared when UART receives any new data.   |
| [13:8] | RX_POINTER    | <b>RX FIFO Pointer (Read Only)</b><br>This field indicates the RX FIFO Buffer Pointer. When UART receives one byte from external device, then RX_POINTER increases one. When one byte of RX FIFO is read by CPU, then RX_POINTER decreases one.<br>The Maximum value shown in RX_POINTER is 15. When the using level of RX FIFO Buffer equal to 16, the RX_FULL bit is set to 1 and RX_POINTER will show 0. As one byte of RX FIFO is read by CPU, the RX_FULL bit is clear to 0 and RX_POINTER will show 15. |
| [7]    | Reserved      | Reserved  |
| [6]    | BIF           | <b>Break Interrupt Flag (Read Only)</b><br>This bit is set to a logic 1 whenever the received data input(RX) is held in the "spacing state" (logic 0) for longer than a full word transmission time (that is, the total time of "start bit" + data bits + parity + stop bits) and is reset whenever the CPU writes 1 to this bit.<br><b>Note:</b> This bit is read only, but can be cleared by writing '1' to it.   |
| [5]    | FEF           | <b>Framing Error Flag (Read Only)</b><br>This bit is set to logic 1 whenever the received character does not have a valid "stop bit" (that is, the stop bit following the last data bit or parity bit is detected as a logic 0), and is reset whenever the CPU writes 1 to this bit.<br><b>Note:</b> This bit is read only, but can be cleared by writing '1' to it.  |
| [4]    | PEF           | <b>Parity Error Flag (Read Only)</b><br>This bit is set to logic 1 whenever the received character does not have a valid "parity bit", and is reset whenever the CPU writes 1 to this bit.<br><b>Note:</b> This bit is read only, but can be cleared by writing '1' to it.  |
| [3]    | RS485_ADD_DET | <b>RS-485 Address Byte Detection Flag (Read Only)</b><br>This bit sets to 1 while UA_ALT_CSR[RS485_ADD_EN] sets to 1 to enable Address detection mode and receive detect a data with an address bit(bit 9 = '1').<br><b>Note:</b> This field is used for RS-485 function mode.<br><b>Note:</b> This bit is read only, but can be cleared by writing '1' to it.  |
| [2:1]  | Reserved      | Reserved  |
| [0]    | RX_OVER_IF    | <b>RX Overflow Error IF (Read Only)</b><br>This bit is set when RX FIFO overflow.<br>If the number of bytes of received data is greater than RX_FIFO (UA_RBR) size, 16 bytes of UART, this bit will be set.<br><b>Note:</b> This bit is read only, but can be cleared by writing '1' to it.   |

**Interrupt Status Control Register (UA\_ISR)**

| Register | Offset        | R/W | Description                    | Reset Value |
|----------|---------------|-----|--------------------------------|-------------|
| UA_ISR   | UARTx_BA+0x1C | R/W | UART Interrupt Status Register | 0x0000_0002 |

|          |          |             |          |           |         |          |         |
|----------|----------|-------------|----------|-----------|---------|----------|---------|
| 31       | 30       | 29          | 28       | 27        | 26      | 25       | 24      |
| Reserved |          |             |          |           |         |          |         |
| 23       | 22       | 21          | 20       | 19        | 18      | 17       | 16      |
| Reserved |          |             |          |           |         |          |         |
| 15       | 14       | 13          | 12       | 11        | 10      | 9        | 8       |
| LIN_INT  | Reserved | BUF_ERR_INT | TOUT_INT | MODEM_INT | RLS_INT | THRE_INT | RDA_INT |
| 7        | 6        | 5           | 4        | 3         | 2       | 1        | 0       |
| LIN_IF   | Reserved | BUF_ERR_IF  | TOUT_IF  | MODEM_IF  | RLS_IF  | THRE_IF  | RDA_IF  |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | Reserved    | Reserved   |
| [15]    | LIN_INT     | <b>LIN Bus Interrupt Indicator (Read Only)</b><br>This bit is set if LIN_IEN and LIN_IF are both set to 1.<br>1 = The LIN Bus interrupt is generated<br>0 = No LIN Bus interrupt is generated                    |
| [14]    | Reserved    | Reserved   |
| [13]    | BUF_ERR_INT | <b>Buffer Error Interrupt Indicator (Read Only)</b><br>This bit is set if BUF_ERR_IEN and BUF_ERR_IF are both set to 1.<br>1 = Buffer error interrupt is generated<br>0 = No buffer error interrupt is generated |
| [12]    | TOUT_INT    | <b>Time Out Interrupt Indicator (Read Only)</b><br>This bit is set if TOUT_IEN and TOUT_IF are both set to 1.<br>1 = Tout interrupt is generated<br>0 = No Tout interrupt is generated                           |
| [11]    | MODEM_INT   | <b>MODEM Status Interrupt Indicator (Read Only)</b><br>This bit is set if MODEM_IEN and MODEM_IF are both set to 1.<br>1 = Modem interrupt is generated<br>0 = No Modem interrupt is generated                   |
| [10]    | RLS_INT     | <b>Receive Line Status Interrupt Indicator (Read Only).</b><br>This bit is set if RLS_IEN and RLS_IF are both set to 1.<br>1 = RLS interrupt is generated<br>0 = No RLS interrupt is generated                   |
| [9]     | THRE_INT    | <b>Transmit Holding Register Empty Interrupt Indicator (Read Only).</b><br>This bit is set if THRE_IEN and THRE_IF are both set to 1.  |

|     |                   |   |
|-----|-------------------|---|
|     |                   | 1 = THRE interrupt is generated<br>0 = No THRE interrupt is generated   |
| [8] | <b>RDA_INT</b>    | <b>Receive Data Available Interrupt Indicator (Read Only).</b><br>This bit is set if RDA_IEN and RDA_IF are both set to 1.<br>1 = RDA interrupt is generated<br>0 = No RDA interrupt is generated   |
| [7] | <b>LIN_IF</b>     | <b>LIN Bus Flag (Read Only)</b><br>This bit is set when LIN slave header detect (LINS_HDET_F=1), LIN break detect (LIN_BKDET_F=1), bit error detect (BIT_ERR_F=1), LIN slave ID parity error (LINS_IDPERR_F) or LIN slave header error detect (LINS_HERR_F) If UA_IER [LIN_IEN] is enabled the LIN interrupt will be generated.<br><b>Note:</b> This bit is cleared when LINS_HDET_F, LIN_BKDET_F, BIT_ERR_F, LINS_IDPENR_F and LINS_HERR_F all are cleared   |
| [6] | <b>Reserved</b>   | Reserved  |
| [5] | <b>BUF_ERR_IF</b> | <b>Buffer Error Interrupt Flag (Read Only)</b><br>This bit is set when the TX or RX FIFO overflows or Break Interrupt Flag or Parity Error Flag or Frame Error Flag (TX_OVER_IF or RX_OVER_IF or BIF or PEF or FEF) is set. When BUF_ERR_IF is set, the transfer is not correct. If UA_IER [BUF_ERR_IEN] is enabled, the buffer error interrupt will be generated.  |
| [4] | <b>TOUT_IF</b>    | <b>Time Out Interrupt Flag (Read Only)</b><br>This bit is set when the RX FIFO is not empty and no activities occurred in the RX FIFO and the time out counter equal to TOIC. If UA_IER [TOUT_IEN] is enabled, the Tout interrupt will be generated.<br><b>Note:</b> This bit is read only and user can read UA_RBR (RX is in active) to clear it.  |
| [3] | <b>MODEM_IF</b>   | <b>MODEM Interrupt Flag (Read Only)</b><br>This bit is set when the nCTS pin has state change (DCTS_F=1). If UA_IER [MODEM_IEN] is enabled, the Modem interrupt will be generated.<br><b>Note:</b> This bit is read only and reset to 0 when bit DCTS_F is cleared by a write 1 on DCTS_F.  |
| [2] | <b>RLS_IF</b>     | <b>Receive Line Interrupt Flag (Read Only).</b><br>This bit is set when the RX receive data have parity error, frame error or break error (at least one of 3 bits, BIF, FEF and PEF, is set). If UA_IER [RLS_IEN] is enabled, the RLS interrupt will be generated.<br><b>Note:</b> In RS-485 function mode, this field is set include "receiver detect and received address byte character (bit9 = '1') bit". At the same time, the bit of UA_FSR[RS485_ADD_DET_F] is also set.<br><b>Note:</b> This bit is read only and reset to 0 when all bits of BIF, FEF and PEF are cleared. |
| [1] | <b>THRE_IF</b>    | <b>Transmit Holding Register Empty Interrupt Flag (Read Only).</b><br>This bit is set when the last data of TX FIFO is transferred to Transmitter Shift Register. If UA_IER [THRE_IEN] is enabled, the THRE interrupt will be generated.<br><b>Note:</b> This bit is read only and it will be cleared when writing data into THR (TX FIFO not empty).   |
| [0] | <b>RDA_IF</b>     | <b>Receive Data Available Interrupt Flag (Read Only).</b><br>When the number of bytes in the RX FIFO equals the RFITL then the RDA_IF will be set. If UA_IER [RDA_IEN] is enabled, the RDA interrupt will be generated.<br><b>Note:</b> This bit is read only and it will be cleared when the number of unread bytes of RX FIFO drops below the threshold level (RFITL).  |

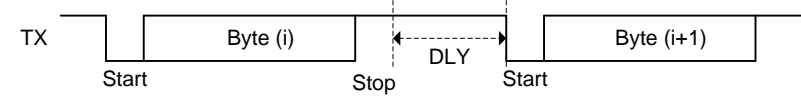
| UART Interrupt Source                                | Interrupt Enable Bit | Interrupt Indicator to Interrupt Controller | Interrupt Flag   | Flag Cleared by  |
|--|----------------------|---|--|--|
| LIN interrupt  | LIN_IEN              | LIN_INT                                     | <b>LIN_IF</b>  | Write '1' to LINS_HDET_F/LIN_BKDET_F/BIT_ERR_F/LINS_IDPERR_F/LINS_HERR_F |
| Buffer Error Interrupt (INT_BUF_ERR)                 | BUF_ERR_IEN          | BUF_ERR_INT                                 | <b>BUF_ERR_IF</b> = (TX_OVER_IF or RX_OVER_IF)         | Write '1' to TX_OVER_IF/RX_OVER_IF                                       |
| RX Timeout Interrupt (INT_TOUT)                      | TOUT_IEN             | TOUT_INT                                    | <b>TOUT_IF</b>   | Read UA_RBR  |
| Modem Status Interrupt (INT_MODEM)                   | MODEM_IEN            | MODEM_INT                                   | <b>MODEM_IF</b> = (DCTS_F)                             | Write '1' to DCTS_F  |
| Receive Line Status Interrupt (INT_RLS)              | RLS_IEN              | RLS_INT                                     | <b>RLS_IF</b> = (BIF or FEF or PEF or RS-485_ADD_DETF) | Write '1' to BIF/FEF/PEF/RS-485_ADD_DETF                                 |
| Transmit Holding Register Empty Interrupt (INT_THRE) | THRE_IEN             | THRE_INT                                    | <b>THRE_IF</b>   | Write UA_THR   |
| Receive Data Available Interrupt (INT_RDA)           | RDA_IEN              | RDA_INT                                     | <b>RDA_IF</b>  | Read UA_RBR  |

Table 19-4 UART Interrupt Sources and Flags Table in Software Mode

**Time out Register (UA TOR)**

| Register | Offset        | R/W | Description            | Reset Value |
|----------|---------------|-----|------------------------|-------------|
| UA_TOR   | UARTx_BA+0x20 | R/W | UART Time Out Register | 0x0000_0000 |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DLY      |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TOIC     |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | Reserved    | Reserved   |
| [15:8]  | DLY         | <b>TX Delay Time Value</b><br>This field is used to programming the transfer delay time between the last stop bit and next start bit.<br>  |
| [7:0]   | TOIC        | <b>Time Out Interrupt Comparator</b><br>The time out counter resets and starts counting (the counting clock = baud rate) whenever the RX FIFO receives a new data word. Once the content of time out counter (TOUT_CNT) is equal to that of time out interrupt comparator (TOIC), a receiver time out interrupt (INT_TOUT) is generated if UA_IER [TOUT_IEN] enabled. A new incoming data word or RX FIFO empty will clear INT_TOUT. In order to avoid receiver time out interrupt generation immediately during one character is being received, TOIC value should be set between 40 and 255. So, for example, if TOIC is set with 40, the time out interrupt is generated after four characters are not received when 1 stop bit and no parity check is set for UART transfer. |



### Baud Rate Divider Register (UA\_BAUD)

| Register | Offset        | R/W | Description                     | Reset Value |
|----------|---------------|-----|---------------------------------|-------------|
| UA_BAUD  | UARTx_BA+0x24 | R/W | UART Baud Rate Divisor Register | 0x0F00_0000 |

|          |    |          |           |           |    |    |    |
|----------|----|----------|-----------|-----------|----|----|----|
| 31       | 30 | 29       | 28        | 27        | 26 | 25 | 24 |
| Reserved |    | DIV_X_EN | DIV_X_ONE | DIVIDER_X |    |    |    |
| 23       | 22 | 21       | 20        | 19        | 18 | 17 | 16 |
| Reserved |    |          |           |           |    |    |    |
| 15       | 14 | 13       | 12        | 11        | 10 | 9  | 8  |
| BRD      |    |          |           |           |    |    |    |
| 7        | 6  | 5        | 4         | 3         | 2  | 1  | 0  |
| BRD      |    |          |           |           |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:30] | Reserved    | Reserved   |
| [29]    | DIV_X_EN    | <b>Divider X Enable</b><br>The BRD = Baud Rate Divider, and the baud rate equation is<br>$\text{Baud Rate} = \text{Clock} / [M * (\text{BRD} + 2)]$ ; The default value of M is 16.<br>1 = Divider X Enabled (the equation of $M = X+1$ , but DIVIDER_X [27:24] must $\geq 8$ ).<br>0 = Divider X Disabled (the equation of $M = 16$ )<br>Refer to Table 19-1 for more information.<br><b>Note:</b> In IrDA mode, this bit must disable. |
| [28]    | DIV_X_ONE   | <b>Divider X Equal to 1</b><br>1 = Divider M = 1 (the equation of $M = 1$ , but BRD [15:0] must $\geq 3$ ).<br>0 = Divider M = X (the equation of $M = X+1$ , but DIVIDER_X [27:24] must $\geq 8$ )<br>Refer to Table 19-1 for more information.   |
| [27:24] | DIVIDER_X   | <b>Divider X</b><br>The baud rate divider $M = X+1$ .  |
| [23:16] | Reserved    | Reserved   |
| [15:0]  | BRD         | <b>Baud Rate Divider</b><br>The field indicates the baud rate divider  |

| Mode | DIV_X_EN | DIV_X_ONE | DIVIDER_X  | BRD | Baud rate equation                                     |
|------|----------|-----------|------------|-----|--|
| 0    | Disable  | 0         | Don't care | A   | $\text{UART\_CLK} / [16 * (A+2)]$                      |
| 1    | Enable   | 0         | B          | A   | $\text{UART\_CLK} / [(B+1) * (A+2)]$ , B must $\geq 8$ |
| 2    | Enable   | 1         | Don't care | A   | $\text{UART\_CLK} / (A+2)$ , A must $\geq 3$           |

Table 19-5 Baud Rate Equation Table

**IrDA Control Register (IRCR)**

| Register | Offset        | R/W | Description                | Reset Value |
|----------|---------------|-----|----------------------------|-------------|
| UA_IRCR  | UARTx_BA+0x28 | R/W | UART IrDA Control Register | 0x0000_0040 |

|          |        |        |          |    |    |           |          |
|----------|--------|--------|----------|----|----|-----------|----------|
| 31       | 30     | 29     | 28       | 27 | 26 | 25        | 24       |
| Reserved |        |        |          |    |    |           |          |
| 23       | 22     | 21     | 20       | 19 | 18 | 17        | 16       |
| Reserved |        |        |          |    |    |           |          |
| 15       | 14     | 13     | 12       | 11 | 10 | 9         | 8        |
| Reserved |        |        |          |    |    |           |          |
| 7        | 6      | 5      | 4        | 3  | 2  | 1         | 0        |
| Reserved | INV_RX | INV_TX | Reserved |    |    | TX_SELECT | Reserved |

| Bits   | Description |  |
|--------|-------------|--|
| [31:7] | Reserved    | Reserved   |
| [6]    | INV_RX      | INV_RX<br>1 = Inverse RX input signal<br>0 = No inversion              |
| [5]    | INV_TX      | INV_TX<br>1 = Inverse TX output signal<br>0 = No inversion             |
| [4:2]  | Reserved    | Reserved   |
| [1]    | TX_SELECT   | TX_SELECT<br>1 = IrDA transmitter Enabled<br>0 = IrDA receiver Enabled |
| [0]    | Reserved    | Reserved   |

**Note:** In IrDA mode, the UA\_BAUD [DIV\_X\_EN] register must be disabled (the baud equation must be Clock / 16 \* (BRD)

### UART Alternate Control/Status Register (UA\_ALT\_CSR)

| Register   | Offset        | R/W | Description                            | Reset Value |
|------------|---------------|-----|--|-------------|
| UA_ALT_CSR | UARTx_BA+0x2C | R/W | UART Alternate Control/Status Register | 0x0000_0000 |

|              |           |          |    |          |           |           |           |
|--------------|-----------|----------|----|----------|-----------|-----------|-----------|
| 31           | 30        | 29       | 28 | 27       | 26        | 25        | 24        |
| ADDR_MATCH   |           |          |    |          |           |           |           |
| 23           | 22        | 21       | 20 | 19       | 18        | 17        | 16        |
| Reserved     |           |          |    |          |           |           |           |
| 15           | 14        | 13       | 12 | 11       | 10        | 9         | 8         |
| RS485_ADD_EN | Reserved  |          |    |          | RS485_AUD | RS485_AAD | RS485_NMM |
| 7            | 6         | 5        | 4  | 3        | 2         | 1         | 0         |
| LIN_TX_EN    | LIN_RX_EN | Reserved |    | LIN_BKFL |           |           |           |

| Bits    | Description  |  |
|---------|--------------|--|
| [31:24] | ADDR_MATCH   | <b>Address Match Value Register</b><br>This field contains the RS-485 address match values.<br><b>Note:</b> This field is used for RS-485 auto address detection mode.   |
| [23:16] | Reserved     | Reserved   |
| [15]    | RS485_ADD_EN | <b>RS-485 Address Detection Enable</b><br>This bit is used to enable RS-485 Address Detection mode.<br>1 = Address detection mode Enabled<br>0 = Address detection mode Disabled<br><b>Note:</b> This field is used for RS-485 any operation mode.                 |
| [14:11] | Reserved     | Reserved   |
| [10]    | RS485_AUD    | <b>RS-485 Auto Direction Mode (AUD)</b><br>1 = RS-485 Auto Direction Operation mode (AUO) Enabled<br>0 = RS-485 Auto Direction Operation mode (AUO) Disabled<br><b>Note:</b> It can be active with RS-485_AAD or RS-485_NMM operation mode.                        |
| [9]     | RS485_AAD    | <b>RS-485 Auto Address Detection Operation Mode (AAD)</b><br>1 = RS-485 Auto Address Detection Operation mode (AAD) Enabled<br>0 = RS-485 Auto Address Detection Operation mode (AAD) Disabled<br><b>Note:</b> It cannot be active with RS-485_NMM operation mode. |
| [8]     | RS485_NMM    | <b>RS-485 Normal Multi-drop Operation Mode (NMM)</b><br>1 = RS-485 Normal Multi-drop Operation mode (NMM) Enabled<br>0 = RS-485 Normal Multi-drop Operation mode (NMM) Disabled<br><b>Note:</b> It cannot be active with RS-485_AAD operation mode.                |
| [7]     | LIN_TX_EN    | LIN TX Break Mode Enable   |

|       |                  |  |
|-------|------------------|--|
|       |                  | 1 = LIN TX Break mode Enabled<br>0 = LIN TX Break mode Disabled<br><b>Note:</b> When TX break field transfer operation finished, this bit will be cleared automatically.   |
| [6]   | <b>LIN_RX_EN</b> | <b>LIN RX Enable</b><br>1 = LIN RX mode Enabled<br>0 = LIN RX mode Disabled  |
| [5:4] | <b>Reserved</b>  | Reserved   |
| [3:0] | <b>LIN_BKFL</b>  | <b>UART LIN Break Field Length</b><br>This field indicates a 4-bit LIN TX break field count.<br><b>Note1:</b> This break field length is UA_LIN_BKFL + 1<br><b>Note2:</b> According to LIN spec, the reset value is 0xC (break field length = 13). |

### UART Function Select Register (UA\_FUN\_SEL)

| Register   | Offset        | R/W | Description                   | Reset Value |
|------------|---------------|-----|-------------------------------|-------------|
| UA_FUN_SEL | UARTx_BA+0x30 | R/W | UART Function Select Register | 0x0000_0000 |

|          |    |    |    |    |    |         |    |
|----------|----|----|----|----|----|---------|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25      | 24 |
| Reserved |    |    |    |    |    |         |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17      | 16 |
| Reserved |    |    |    |    |    |         |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9       | 8  |
| Reserved |    |    |    |    |    |         |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1       | 0  |
| Reserved |    |    |    |    |    | FUN_SEL |    |

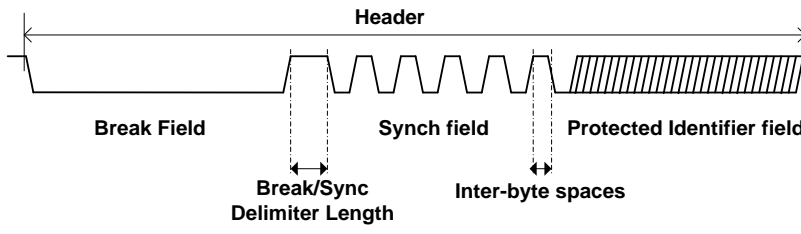
| Bits   | Description |  |
|--------|-------------|--|
| [31:2] | Reserved    | Reserved   |
| [1:0]  | FUN_SEL     | <b>Function Select Enable</b><br>00 = UART function Enabled<br>01 = LIN function Enabled<br>10 = IrDA function Enabled<br>11 = RS-485 function Enabled |

**UART LIN Control Register (UA\_LIN\_CTL)**

| Register   | Offset        | R/W | Description               | Reset Value |
|------------|---------------|-----|---------------------------|-------------|
| UA_LIN_CTL | UARTx_BA+0x34 | R/W | UART LIN Control Register | 0x000C_0000 |

|              |    |            |             |             |              |              |         |
|--------------|----|------------|-------------|-------------|--------------|--------------|---------|
| 31           | 30 | 29         | 28          | 27          | 26           | 25           | 24      |
| LIN_PID      |    |            |             |             |              |              |         |
| 23           | 22 | 21         | 20          | 19          | 18           | 17           | 16      |
| LIN_HEAD_SEL |    | LIN_BS_LEN |             | LIN_BKFL    |              |              |         |
| 15           | 14 | 13         | 12          | 11          | 10           | 9            | 8       |
| Reserved     |    |            | BIT_ERR_EN  | LIN_RX_DIS  | LIN_BKDET_EN | LIN_IDPEN    | LIN_SHD |
| 7            | 6  | 5          | 4           | 3           | 2            | 1            | 0       |
| Reserved     |    |            | LIN_MUTE_EN | LINS_DUM_EN | LINS_ARS_EN  | LINS_HDET_EN | LINS_EN |

| Bits    | Description  |   |     |       |     |     |     |     |     |     |    |    |
|---------|--------------|---|-----|-------|-----|-----|-----|-----|-----|-----|----|----|
| [31:24] | LIN_PID      | <p><b>LIN PID Register</b></p> <p>This field contains the LIN frame ID value when in LIN function mode, the frame ID parity can be generated by software or hardware depends on UA_LIN_CTL [LIN_IDPEN].</p> <p>If the parity generated by hardware (UA_LIN_CTL [LIN_IDPEN] = 1), user fill ID0~ID5, (LIN_PID[24:29]hardware will calculate P0(LIN_PID[30]) and P1(LIN_PID[31]), otherwise user must filled frame ID and parity in this field.</p> <table><tr><td>PID</td><td>Start</td><td>ID0</td><td>ID1</td><td>ID2</td><td>ID3</td><td>ID4</td><td>ID5</td><td>P0</td><td>P1</td></tr></table> <p>P0 = ID0 xor ID1 xor ID2 xor ID4<br/>P1 = ~(ID1 xor ID3 xor ID4 xor ID5)</p> <p><b>Note1:</b> User can filled any 8-bit value to this field and the bit 24 indicates ID0 (LSB first)</p> <p><b>Note2:</b> This field can be used for LIN master mode or slave mode.</p> | PID | Start | ID0 | ID1 | ID2 | ID3 | ID4 | ID5 | P0 | P1 |
| PID     | Start        | ID0   | ID1 | ID2   | ID3 | ID4 | ID5 | P0  | P1  |     |    |    |
| [23:22] | LIN_HEAD_SEL | <p><b>LIN Header Select</b></p> <p>00 = The LIN header includes “break field”.</p> <p>01 = The LIN header includes “break field” and “sync field”.</p> <p>10 = The LIN header includes “break field”, “sync field” and “frame ID field”.</p> <p>11 = Reserved.</p> <p><b>Note:</b> This bit is used to master mode for LIN to sending header field (LIN_SHD = 1) or used to slave to indicates exit from mute mode condition(LIN_MUTE_EN).</p>  |     |       |     |     |     |     |     |     |    |    |
| [21:20] | LIN_BS_LEN   | <p><b>LIN Break/Sync Delimiter Length</b></p> <p>00 = The LIN break/sync delimiter length is 1 bit time.</p> <p>10 = The LIN break/sync delimiter length is 2 bit time.</p> <p>10 = The LIN break/sync delimiter length is 3 bit time.</p>  |     |       |     |     |     |     |     |     |    |    |

|         |                     |  |
|---------|---------------------|--|
|         |                     | <p>11 = The LIN break/sync delimiter length is 4 bit time.</p>  <p><b>Note:</b> This bit used for LIN master to sending header field.</p>  |
| [19:16] | <b>LIN_BKFL</b>     | <p><b>LIN Break Field Length</b></p> <p>This field indicates a 4-bit LIN TX break field count.</p> <p><b>Note1:</b> These registers are shadow registers of UA_ALT_CSR [LIN_BKFL], User can read/write it by setting UA_ALT_CSR [LIN_BKFL] or UA_LIN_CTL [LIN_BKFL].</p> <p><b>Note2:</b> This break field length is LIN_BKFL + 1.</p> <p><b>Note3:</b> According to LIN spec, the reset value is 0XC (break field length = 13).</p> |
| [15:13] | <b>Reserved</b>     | Reserved   |
| [12]    | <b>BIT_ERR_EN</b>   | <p><b>Bit Error Detect Enable</b></p> <p>1 = Bit error detection Enabled.</p> <p>0 = Bit error detection function Disabled.</p> <p><b>Note:</b> In LIN function mode, when occur bit error, the UA_LIN_SR [BIT_ERR_F] flag will be asserted. If the UA_IER[LIN_IEN] = 1, an interrupt will be generated.</p>   |
| [11]    | <b>LIN_RX_DIS</b>   | <p>If the receiver is be enabled (LIN_RX_DIS = 0), all received byte data will be accepted and stored in the RX-FIFO, and if the receiver is disabled (LIN_RX_DIS = 1), all received byte data will be ignore.</p> <p>1 = Bit error detection Enabled.</p> <p>0 = Error detection function Disabled.</p> <p><b>Note:</b> This bit is only valid when operating in LIN function mode (UA_FUN_SEL[FUN_SEL] = 01)</p>                   |
| [10]    | <b>LIN_BKDET_EN</b> | <p><b>LIN Break Detection Enable</b></p> <p>When detect consecutive dominant greater than 11 bits, and are followed by a delimiter character, the LIN_BKDET_F flag is set in UA_LIN_SR register at the end of break field. If the UA_IER [LIN_IEN] = 1, an interrupt will be generated.</p> <p>1 = LIN break detection Enabled.</p> <p>0 = LIN break detection. Disabled</p>   |
| [9]     | <b>LIN_IDPEN</b>    | <p><b>LIN ID Parity Enable</b></p> <p>1 = LIN frame ID parity Enabled.</p> <p>0 = LIN frame ID parity Disabled.</p> <p><b>Note1:</b> This bit can be used for LIN master to sending header field (LIN_SHD = 1 and LIN_HEAD_SEL = 10) or be used for enable LIN slave received frame ID parity checked.</p> <p><b>Note2:</b> This bit is only use when operation header transmitter is in LIN_HEAD_SEL = 10.</p>                      |
| [8]     | <b>LIN_SHD</b>      | <p><b>LIN TX Send Header Enable</b></p> <p>The LIN TX header can be "break field" or "break and sync field" or "break, sync and frame ID field", it is depend on setting LIN_HEAD_SEL register.</p> <p>1 = Send LIN TX header Enabled.</p>   |

|       |              |  |
|-------|--------------|--|
|       |              | <p>0 = Send LIN TX header Disabled.</p> <p><b>Note1:</b> These registers are shadow registers of UA_ALT_CSR [LIN_SHD]; user can read/write it by setting UA_ALT_CSR [LIN_SHD] or UA_LIN_CTL [LIN_SHD].</p> <p><b>Note2:</b> When transmitter header field (it may be “break” or “break + sync” or “break + sync + frame ID” selected by LIN_HEAD_SEL field) transfer operation finished, this bit will be cleared automatically.</p>   |
| [7:5] | Reserved     | Reserved   |
| [4]   | LIN_MUTE_EN  | <p><b>LIN Mute Mode Enable</b></p> <p>1 = LIN mute mode Enabled.</p> <p>0 = LIN mute mode Disabled.</p> <p><b>Note:</b> The exit from mute mode condition and each control and interactions of this field are explained in LIN slave mode.</p>   |
| [3]   | LINS_DUM_EN  | <p><b>LIN Slave Divider Update Method Enable</b></p> <p>1 = UA_BAUD is updated at the next received character. User must set the bit before checksum reception.</p> <p>0 = UA_BAUD is updated as soon as UA_BAUD is writing by software (if no automatic resynchronization update occurs at the same time)</p> <p><b>Note1:</b> This bit only valid when in LIN slave mode (LINS_EN = 1).</p> <p><b>Note2:</b> This bit used for LIN slave automatic resynchronization mode. (for non-automatic resynchronization mode, this bit should be kept cleared)</p> <p><b>Note3:</b> The control and interactions of this field are explained in Slave mode with automatic resynchronization.</p> |
| [2]   | LINS_ARS_EN  | <p><b>LIN Slave Automatic Resynchronization Mode Enable</b></p> <p>1 = LIN automatic resynchronization Enabled.</p> <p>0 = LIN automatic resynchronization Disabled.</p> <p><b>Note1:</b> This bit only valid when in LIN slave mode (LINS_EN = 1).</p> <p><b>Note2:</b> When operation in automatic resynchronization mode, the baud rate setting must be mode2 (UA_BAUD [DIV_X_EN] and UA_BAUD [DIV_X_ONE] must be 1).</p> <p><b>Note3:</b> The control and interactions of this field are explained in Slave mode with automatic resynchronization.</p>   |
| [1]   | LINS_HDET_EN | <p><b>LIN Slave Header Detection Enable</b></p> <p>1 = LIN slave header detection Enabled.</p> <p>0 = LIN slave header detection Disabled.</p> <p><b>Note1:</b> This bit only valid when in LIN slave mode (LINS_EN = 1).</p> <p><b>Note2:</b> In LIN function mode, when detect header field (break + sync + frame ID), UA_LIN_SR [LINS_HDET_F] flag will be asserted. If the UA_IER[LIN_IEN] = 1, an interrupt will be generated.</p>  |
| [0]   | LINS_EN      | <p><b>LIN Slave Mode Enable</b></p> <p>1 = LIN slave mode Enabled.</p> <p>0 = LIN slave mode Disabled.</p>   |



### UART LIN Status Register (UA\_LIN\_SR)

| Register  | Offset        | R/W | Description              | Reset Value |
|-----------|---------------|-----|--------------------------|-------------|
| UA_LIN_SR | UARTx_BA+0x38 | R/W | UART LIN Status Register | 0x0000_0000 |

|          |    |    |    |             |                |             |             |
|----------|----|----|----|-------------|----------------|-------------|-------------|
| 31       | 30 | 29 | 28 | 27          | 26             | 25          | 24          |
| Reserved |    |    |    |             |                |             |             |
| 23       | 22 | 21 | 20 | 19          | 18             | 17          | 16          |
| Reserved |    |    |    |             |                |             |             |
| 15       | 14 | 13 | 12 | 11          | 10             | 9           | 8           |
| Reserved |    |    |    |             |                | BIT_ERR_F   | LIN_BKDET_F |
| 7        | 6  | 5  | 4  | 3           | 2              | 1           | 0           |
| Reserved |    |    |    | LINS_SYNC_F | LINS_IDPER_R_F | LINS_HERR_F | LINS_HDET_F |

| Bits    | Description |  |
|---------|-------------|--|
| [31:10] | Reserved    | Reserved   |
| [9]     | BIT_ERR_F   | <b>Bit Error Detect Status Flag (Read Only)</b><br>At TX transfer state, hardware will monitoring the bus state, if the input pin (SIN) state not equals to the output pin (SOUT) state, BIT_ERR_F will be set.<br>When occur bit error, if the UA_IER[LIN_IEN] = 1, an interrupt will be generated.<br><b>Note1:</b> This bit is read only, but it can be cleared by writing 1 to it.<br><b>Note2:</b> This bit is only valid when enable bit error detection function (UA_LIN_CTL [BIT_ERR_EN] == 1).  |
| [8]     | LIN_BKDET_F | <b>LIN Break Detection Flag (Read Only)</b><br>This bit is set by hardware when a break is detected and be cleared by software writing 1 to it.<br>1 = LIN break detected.<br>0 = LIN break not detected.<br><b>Note1:</b> This bit is read only, but it can be cleared by writing 1 to it.<br><b>Note2:</b> This bit is only valid when enable LIN break detection function (UA_ALT_CSR [LIN_BKDET_EN])   |
| [7:4]   | Reserved    | Reserved   |
| [3]     | LINS_SYNC_F | <b>LIN Slave Sync Field</b><br>This bit indicates that the LIN sync field is being analyzed in automatic resynchronization mode. When the receiver header have some error been detect, user must to reset the internal circuit to re-search new frame header by writing 1 to this bit.<br>1 = The current character is at LIN sync state.<br>0 = The current character is not at LIN sync state.<br><b>Note1:</b> This bit only valid when in LIN slave mode (LINS_EN = 1).<br><b>Note2:</b> This bit is read only, but it can be cleared by writing 1 to it.<br><b>Note3:</b> When user writing 1 to it, hardware will reload the initial baud-rate and re- |

|     |               |  |
|-----|---------------|--|
|     |               | search new frame header/   |
| [2] | LINS_IDPERR_F | <p><b>LIN Slave ID Parity Error Flag (Read Only)</b></p> <p>This bit is set by hardware when receipted frame ID parity is not correct.</p> <p>1 = Receipted frame ID parity is not correct.</p> <p>0 = no active.</p> <p><b>Note1:</b> This bit is read only, but it can be cleared by writing 1 to it.</p> <p><b>Note2:</b> This bit is only valid when in LIN slave mode (UA_LIN_CTL [LINS_EN] = 1) and enable LIN frame ID parity check function (UA_LIN_CTL [LIN_IDPEN])</p>   |
| [1] | LINS_HERR_F   | <p><b>LIN Slave Header Error Flag (Read Only)</b></p> <p>This bit is set by hardware when a LIN header error is detected in LIN slave mode and be cleared by writing 1 to it. The header errors include "break delimiter is too short", "frame error in sync field or Identifier field", "sync field data is not 0x55 without automatic resynchronization mode", "sync field deviation error with automatic resynchronization mode", "sync field measure time-out with automatic resynchronization mode" and "LIN header reception time-out".</p> <p>1 = LIN header error detected.</p> <p>0 = LIN header error not detected.</p> <p><b>Note1:</b> This bit is read only, but it can be cleared by writing 1 to it.</p> <p><b>Note2:</b> This bit is only valid when in LIN slave mode (UA_LIN_CTL [LINS_EN] = 1) and enable LIN slave header detection function (UA_LIN_CTL [LINS_HDET_EN])</p> |
| [0] | LINS_HDET_F   | <p><b>LIN Slave Header Detection Flag (Read Only)</b></p> <p>This bit is set by hardware when a LIN header is detected in LIN slave mode and be cleared by writing 1 to it.</p> <p>1 = LIN header detected (break + sync + frame ID).</p> <p>0 = LIN header not detected.</p> <p><b>Note1:</b> This bit is read only, but it can be cleared by writing 1 to it.</p> <p><b>Note2:</b> This bit is only valid when in LIN slave mode (UA_LIN_CTL [LINS_EN] = 1) and enable LIN slave header detection function (UA_LIN_CTL [LINS_HDET_EN])</p> <p><b>Note3:</b> When enable ID parity check (UA_LIN_CTL [LIN_IDPEN] = 1), if hardware detect complete header ("break + sync + frame ID"), the LINS_HDET_F will be set whether the frame ID correct or not.</p>   |

## 20 SERIAL PERIPHERAL INTERFACE (SPI)

### 20.1 Overview

The Serial Peripheral Interface (SPI) is a synchronous serial data communication protocol that operates in full duplex mode. Devices communicate in Master/Slave mode with the 4-wire bi-direction interface. The NuMicro™ NM15xx series contains up to three sets of SPI controllers performing a serial-to-parallel conversion on data received from a peripheral device, and a parallel-to-serial conversion on data transmitted to a peripheral device. Each set of SPI controller can be configured as a master or a slave device.

### 20.2 Features

- Up to three sets of SPI controller
- Supports Master or Slave mode operation
- Configurable bit length of a transfer word from 8 to 32-bit
- Provides separate 8-layer depth transmit and receive FIFO buffers
- Supports MSB first or LSB first transfer sequence
- Supports the byte reorder function
- Supports Byte or Word Suspend mode
- Supports 3-wire, no slave select signal, bi-direction interface

### 20.3 Block Diagram

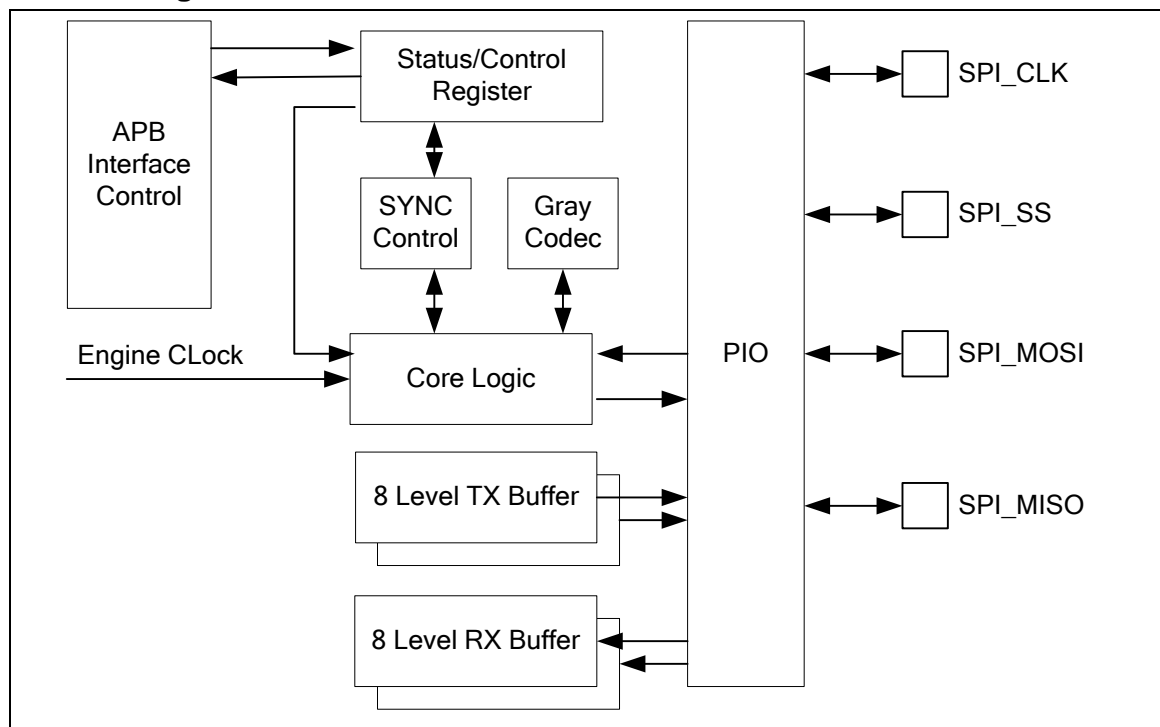


Figure 20-1 SPI Block Diagram

## 20.4 Functional Description

### SPI Engine Clock and SPI Serial Clock

The SPI controller needs the SPI engine clock to drive the SPI logic unit to perform the data transfer. The SPI engine clock rate is determined by the settings of clock source and clock divisor. The SPI\_SS bit of CLKSEL1 register determines the clock source of the SPI engine clock. The clock source can be HCLK or PLL output clock. The DIVIDER setting of SPI\_DIVIDER register determines the divisor of the clock rate calculation.

In Master mode, the output frequency of the SPI serial clock output pin is equal to the SPI engine clock rate. In general, the SPI serial clock denotes as SPI clock. In Slave mode, the SPI serial clock is provided by an off-chip master device. The SPI engine clock rate of slave device must be faster than the SPI serial clock rate of the master device connected together. The frequency of SPI engine clock cannot be faster than the APB clock rate regardless of Master or Slave mode.

### Master/Slave Mode

The SPI controller can be set as Master or Slave mode by setting the SLAVE bit (SPI\_CNTRL[18]) to communicate with the off-chip SPI Slave or Master device. The application block diagrams in Master and Slave mode are shown below.

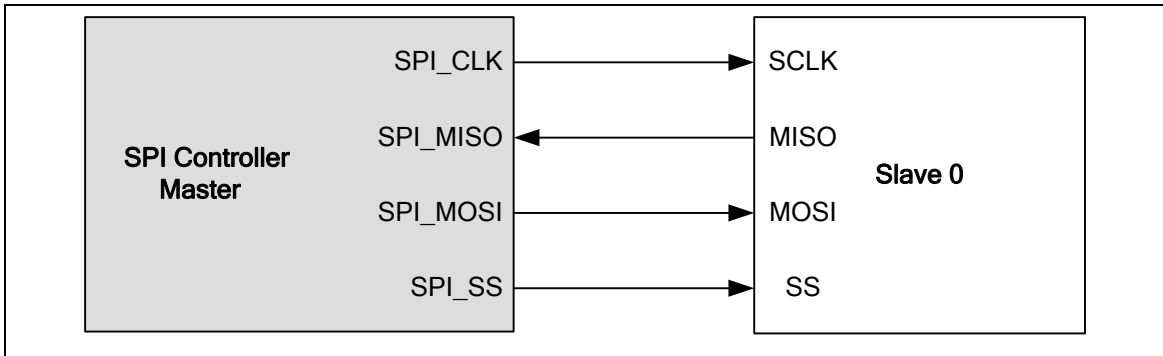


Figure 20-2 SPI Master Mode Application Block Diagram

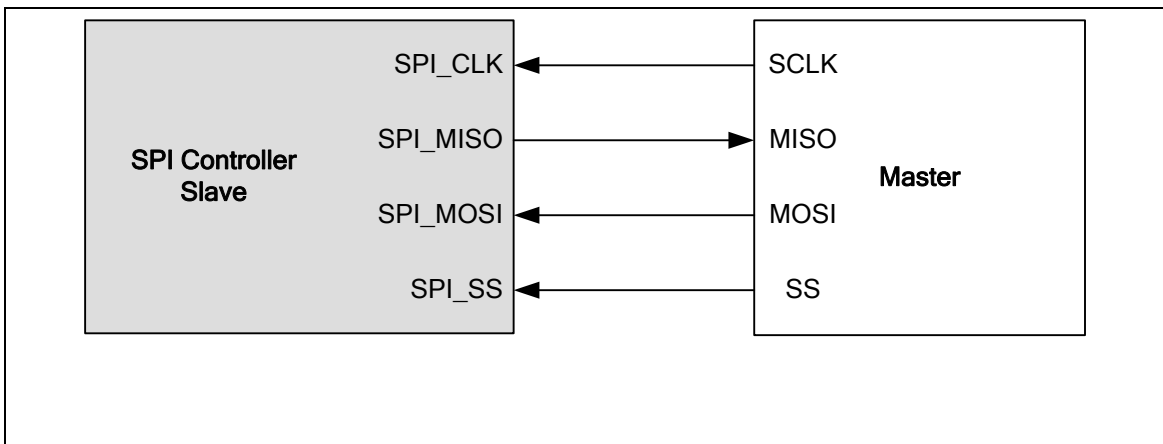


Figure 20-3 SPI Slave Mode Application Block Diagram

### Slave Selection

In Master mode, this SPI controller can drive one slave devices through the slave select output pin SPI\_SS. In Slave mode, the off-chip master device drives the slave select signal from the SPI\_SS input port to this SPI controller. In Master/Slave mode, the active state of slave select signal can be programmed to low or high active in SS\_LVL bit (SPI\_SSR[2]), and the SS\_LTRIG bit (SPI\_SSR[4]) defines the slave select signal SPI\_SS is level-triggered or edge-triggered. The selection of trigger conditions depends on what type of peripheral slave/master device is connected.

In Slave mode, if the SS\_LTRIG bit is configured as level trigger, the LTRIG\_FLAG bit (SPI\_SSR[5]) is used to indicate if the received bits among one transaction meets the requirement defined in TX\_BIT\_LEN.

### Level-trigger/Edge-trigger

In Slave mode, the slave select signal can be configured as level-trigger or edge-trigger. For edge-trigger, the data transfer starts from an active edge and ends on an inactive edge. If the master does not send an inactive edge to slave, the transfer procedure will not be completed and the unit transfer interrupt flag of slave will not be set. In level-trigger, the following two conditions will terminate the transfer procedure and the unit transfer interrupt flag of slave will be set. The first condition is that if the number of transferred bits matches the settings of TX\_BIT\_LEN, the unit transfer interrupt flag of slave will be set. As to the second condition, if the master set the slave select pin to inactive level during the transfer is in progress, it will force slave device to terminate the current transfer no matter how many bits have been transferred and the unit transfer interrupt flag will be set. User can read the status of LTRIG\_FLAG bit to see if the data has been completely transferred.

### Automatic Slave Selection

In Master mode, if the bit AUTOSS (SPI\_SSR[3]) is set, the slave select signals will be generated automatically and output to the SPI\_SS pin according to whether SSR (SPI\_SSR[0]) is enabled or not. This means that the slave select signal, which is selected in SSR, will be asserted by the SPI controller when the SPI data transfer is started by setting the GO\_BUSY bit (SPI\_CNTRL[0]) and will be de-asserted after the data transfer is finished. If the AUTOSS bit is cleared, the slave select output signal will be asserted/de-asserted by manual setting/clearing the related bit of SPI\_SSR[0]. The active state of the slave select output signal is specified in SS\_LVL bit (SPI\_SSR[2]).

In Master mode, if the value of SP\_CYCLE[3:0] is less than 3 and the AUTOSS is set as 1, the slave select signal will be kept in active state between two successive transactions.

In Slave mode, to recognize the inactive state of the slave select signal, the inactive period of the slave select signal must be larger than or equal to 6 engine clock periods between two successive transactions.

### Clock Polarity

The CLKP bit (SPI\_CNTRL[11]) defines the SPI clock idle state. If CLKP = 1, the SPI clock is idle at high state; if CLKP = 0, it is idle at low state.

### Transmit/Receive Bit Length

The bit length of a transaction word is defined in TX\_BIT\_LEN bit field (SPI\_CNTRL[7:3]) and can be configured up to 32-bit length in a transaction word for transmitting and receiving.

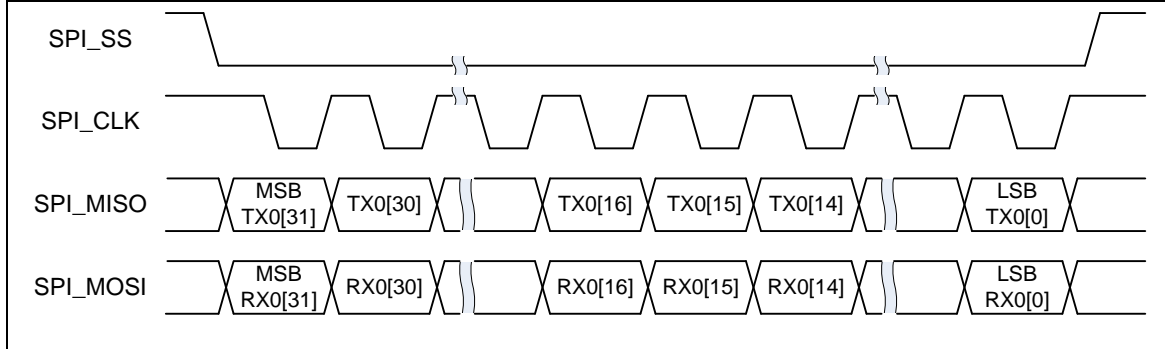


Figure 20-4 32-Bit in One Transaction

### LSB/MSB First

The LSB bit (SPI\_CNTRL[10]) defines the bit transfer sequence in a transaction. If the LSB bit is set to 1, the transfer sequence is LSB first. The bit 0 will be transferred firstly. If the LSB bit is cleared to 0, the transfer sequence is MSB first.

### Transmit Edge

The TX\_NEG bit (SPI\_CNTRL[2]) defines the data transmitted out either on negative edge or on positive edge of SPI clock.

### Receive Edge

The Rx\_NEG bit (SPI\_CNTRL[1]) defines the data received either on negative edge or on positive edge of SPI clock.

**Note:** The settings of TX\_NEG and RX\_NEG are mutual exclusive. In other words, do not transmit and receive data at the same clock edge.

### Word Suspend

The four bit fields of SP\_CYCLE (SPI\_CNTRL[15:12]) provide a configurable suspend interval, 0.5 ~ 15.5 SPI clock periods, between two successive transaction words in Master mode. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value of SP\_CYCLE is 0x3 (3.5 SPI clock cycles). This SP\_CYCLE setting will not take effect to the word suspend interval if FIFO mode is disabled by software.

## Byte Reorder

When the transfer is set as MSB first (LSB = 0) and the REORDER bit is set to 1, the data stored in the TX buffer and RX buffer will be rearranged in the order as [BYTE0, BYTE1, BYTE2, BYTE3] in 32-bit Transfer mode (TX\_BIT\_LEN = 0). The sequence of transmitted/received data will be BYTE0, BYTE1, BYTE2, and then BYTE3. If the TX\_BIT\_LEN is set as 24-bit transfer mode, the data in TX buffer and RX buffer will be rearranged as [unknown byte, BYTE0, BYTE1, BYTE2]. The SPI controller will transmit/receive data with the sequence of BYTE0, BYTE1 and then BYTE2. Each byte will be transmitted/received with MSB first. The rule of 16-bit mode is the same as above. Byte reorder function is only available when TX\_BIT\_LEN is configured as 16, 24, and 32 bits.

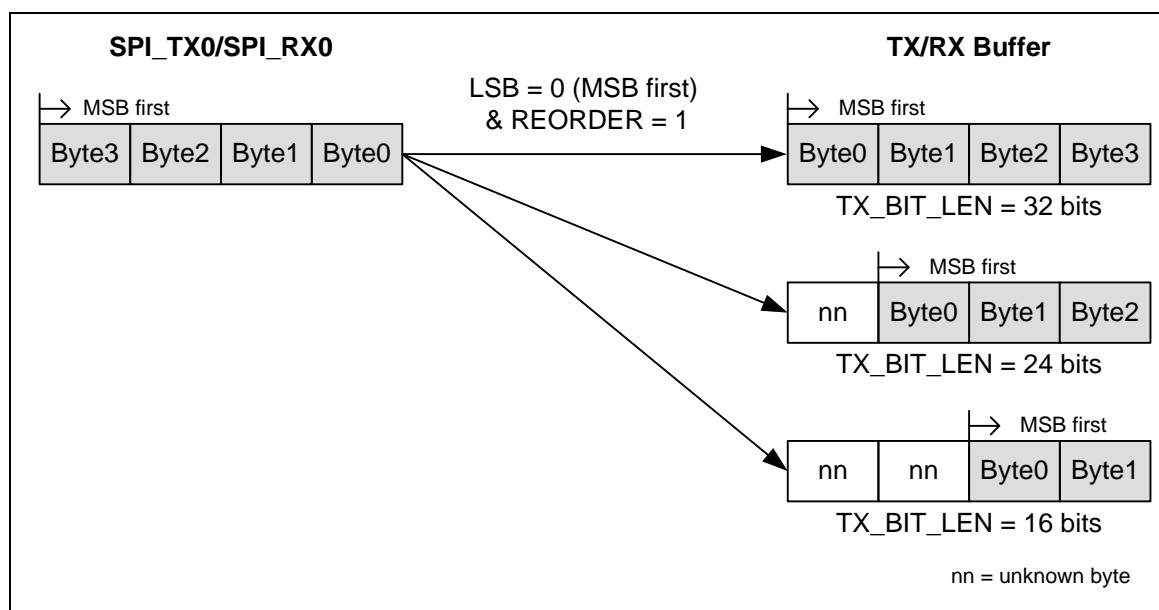


Figure 20-5 Byte Reorder Function



### Byte Suspend

In Master mode, if SPI\_CNTRL[19] is set to 1, a suspend interval of 0.5 ~ 15.5 SPI clock periods will be inserted by hardware between two successive bytes in a transaction word. Both settings of byte suspend interval and word suspend interval are configured in SP\_CYCLE.

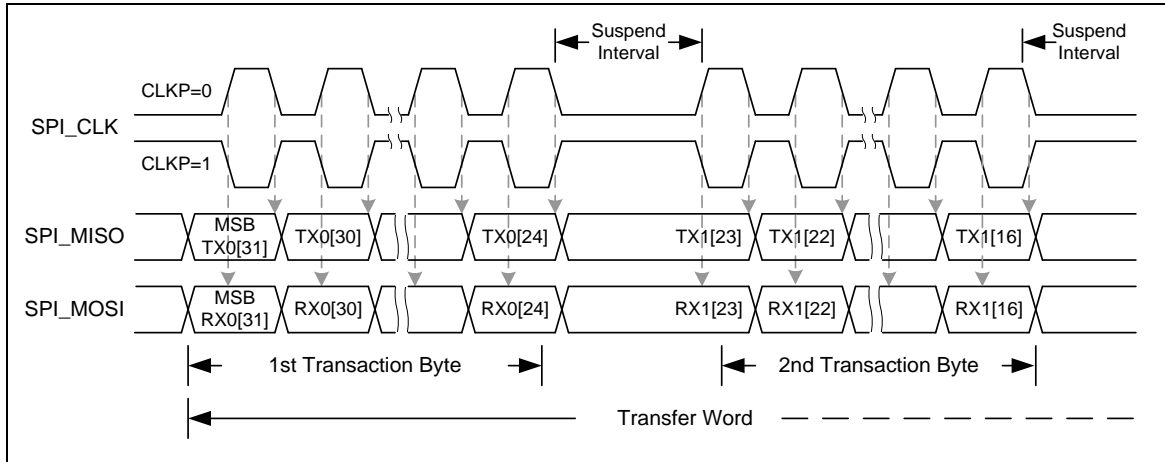


Figure 20-6 Timing Waveform for Byte Suspend

### 3-Wire Mode

When the NOSLVSEL bit is set by software to enable the Slave 3-wire mode, the SPI controller can work with no slave select signal in Slave mode. The NOSLVSEL bit only takes effect in Slave mode. Only three pins, SPI\_CLK, SPI\_MISO, and SPI\_MOSI, are required to communicate with a SPI master. The SPI\_SS pin can be configured as a GPIO. When the NOSLVSEL bit is set to 1, the SPI slave will be ready to transmit/receive data after the GO\_BUSY bit is set to 1. In Slave 3-wire mode, the SS\_LTRIG, SPI\_SSR[4], should be set as 1.

## FIFO Mode

The SPI controller supports FIFO mode when the FIFO bit in SPI\_CNTRL[21] is set as 1. The SPI controllers equip with eight 32-bit wide transmit and receive FIFO buffers.

The transmit FIFO buffer is an 8-layer depth, 32-bit wide, first-in, first-out register buffer. Data can be written to the transmit FIFO buffer through software by writing the SPI\_TX register. The data stored in the transmit FIFO buffer will be read and sent out by the transmission control logic. If the 8-layer transmit FIFO buffer is full, the TX\_FULL bit will be set to 1. When the SPI transmission logic unit draws out the last datum of the transmit FIFO buffer, so that the 8-layer transmit FIFO buffer is empty, the TX\_EMPTY bit will be set to 1. Notice that the TX\_EMPTY flag is set to 1 while the last transaction is still in progress. In Master mode, both the GO\_BUSY bit and TX\_EMPTY bit should be checked by software to make sure whether the SPI is in idle or not.

The received FIFO buffer is also an 8-layer depth, 32-bit wide, first-in, first-out register buffer. The receive control logic will store the received data to this buffer. The FIFO buffer data can be read from SPI\_RX register by software. There are FIFO related status bits, like RX\_EMPTY and RX\_FULL, to indicate the current status of FIFO buffer.

In FIFO mode, the transmitting and receiving threshold can be set through software by setting the TX\_THRESHOLD and RX\_THRESHOLD settings. When the count of valid data stored in transmit FIFO buffer is less than or equal to TX\_THRESHOLD setting, the TX\_INTSTS bit will be set to 1. When the count of valid data stored in receive FIFO buffer is larger than RX\_THRESHOLD setting, the RX\_INTSTS bit will be set to 1.

In FIFO mode, 8 data can be written to the SPI transmit FIFO buffer by software in advance. When the SPI controller operates with FIFO mode, the GO\_BUSY bit of SPI\_CNTRL register will be controlled by hardware, and the content of SPI\_CNTRL register should not be modified by software unless the FIFO bit is cleared to disable FIFO mode.

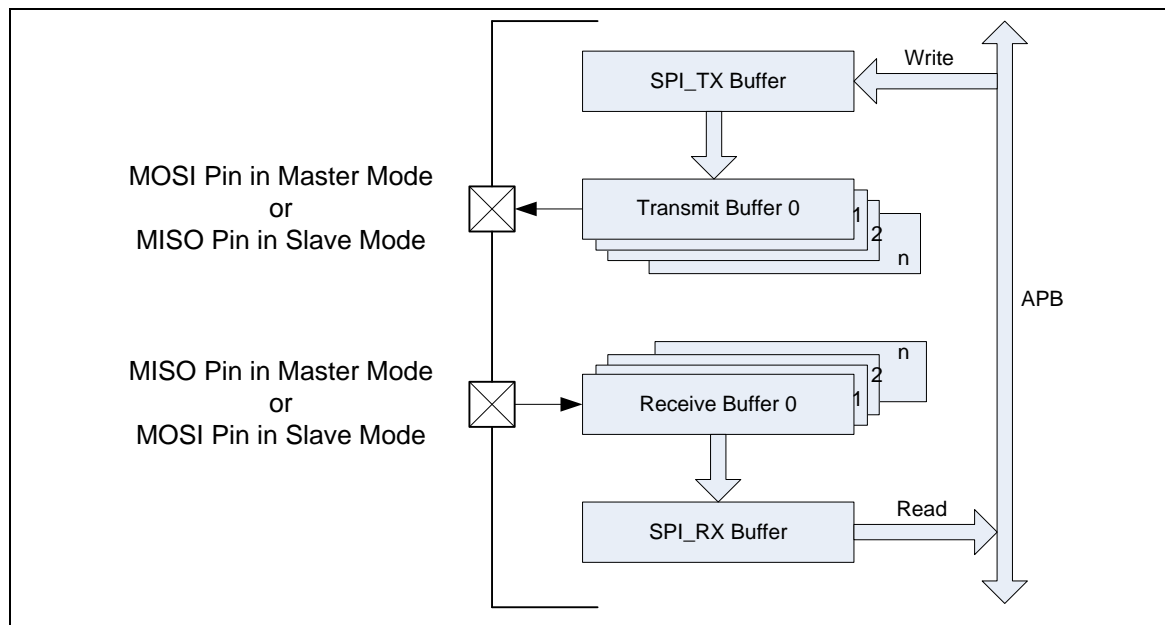


Figure 20-7 FIFO Mode Block Diagram

In Master mode, when the FIFO bit is set to 1 and the first datum is written to the SPI\_TX register, the TX\_EMPTY flag will be cleared to 0. The transmission immediately starts as long as the transmit FIFO buffer is not empty. User can write the next data into SPI\_TX register immediately. The SPI controller will insert a suspend interval between two successive transactions in FIFO

mode and the period of suspend interval is decided by the setting of SP\_CYCLE (SPI\_CNTRL [15:12]). User can write data into SPI\_TX register as long as the TX\_FULL flag is 0.

The subsequent transactions will be triggered automatically if the transmitted data are updated in time. If the SPI\_TX register does not be updated after all data transfer are done, the transfer will stop.

In Master mode, during receiving operation, the serial data are received from SPI\_MISO0/1 pin and stored to receive FIFO buffer. The RX\_EMPTY flag will be cleared to 0 while the receive FIFO buffer contains unread data. The received data can be read by software from SPI\_RX register as long as the RX\_EMPTY flag is 0. If the receive FIFO buffer contains 8 unread data, the RX\_FULL flag will be set to 1. The SPI controller will stop receiving data until the SPI\_RX register is read by software.

In Slave mode, when the FIFO bit is set as 1, the GO\_BUSY bit will be set as 1 by hardware automatically.

In Slave mode, during transmission operation, when data is written to the SPI\_TX register by software, the data will be loaded into transmit FIFO buffer and the TX\_EMPTY flag will be set to 0. The transmission will start when the slave device receives clock signal from master. Data can be written to SPI\_TX register as long as the TX\_FULL flag is 0. After all data have been drawn out by the SPI transmission logic unit and the SPI\_TX register is not updated by software, the TX\_EMPTY flag will be set to 1.

In Slave mode, during receiving operation, the serial data is received from SPI\_MOSI pin and stored to SPI\_RX register. The reception mechanism is similar to Master mode reception operation.

## Interrupt

### ■ SPI unit transfer interrupt

As the SPI controller finishes a unit transfer, the unit transfer interrupt flag IF (SPI\_CNTRL[16]) will be set to 1. The unit transfer interrupt event will generate an interrupt to CPU if the unit transfer interrupt enable bit IE (SPI\_CNTRL[17]) is set. The unit transfer interrupt flag can be cleared only by writing 1 to it.

### ■ SPI slave 3-wire mode start interrupt

In 3-wire mode, the slave 3-wire mode start interrupt flag, SLV\_START\_INTSTS, will be set to 1 when the slave senses the SPI clock signal. The SPI controller will issue an interrupt if the SSTA\_INTEN is set to 1. If the count of the received bits is less than the setting of TX\_BIT\_LEN and there is no more SPI clock input over the expected time period which is defined by the user, the user can set the SLV\_ABORT bit to abort the current transfer. The unit transfer interrupt flag, IF, will be set to 1 if the software set the SLV\_ABORT bit.

### ■ Receive FIFO time out interrupt

In FIFO mode, there is time out function to inform user. If there is a received data in the FIFO and it does not be read by software over 64 SPI engine clock periods in Master mode or over 576 SPI engine clock periods in Slave mode, it will send a time out interrupt to the system if the time out interrupt enable bit, FIFO\_CTL[21], is set to 1.

### ■ Transmit FIFO interrupt

In FIFO mode, if the valid data count of the transmit FIFO buffer is less than or equal to the setting value of TX\_THRESHOLD, the transmit FIFO interrupt flag will be set to 1. The SPI controller will generate a transmit FIFO interrupt to the system if the transmit FIFO interrupt enable bit, SPI\_FIFO\_CTL[3], is set to 1.

### ■ Receive FIFO interrupt

In FIFO mode, if the valid data count of the receive FIFO buffer is larger than the setting value of RX\_THRESHOLD, the receive FIFO interrupt flag will be set to 1. The SPI controller will generate a receive FIFO interrupt to the system if the receive FIFO interrupt enable bit, SPI\_FIFO\_CTL[2], is set to 1.

## 20.5 Timing Diagram

The active state of slave select signal can be defined by setting the SS\_LVL bit (SPI\_SSR[2]) and SS\_LTRIG bit (SPI\_SSR[4]). The SPI clock idle state can be configured as high or low state by setting the CLKP bit (SPI\_CNTRL[11]). It also provides the bit length of a transaction word in TX\_BIT\_LEN (SPI\_CNTRL[7:3]), and transmit/receive data from MSB or LSB first in LSB bit (SPI\_CNTRL[10]). User can also select which edge of SPI clock to transmit/receive data in TX\_NEG/RX\_NEG (SPI\_CNTRL[2:1]). Four SPI timing diagrams for master/slave operations and the related settings are shown below.

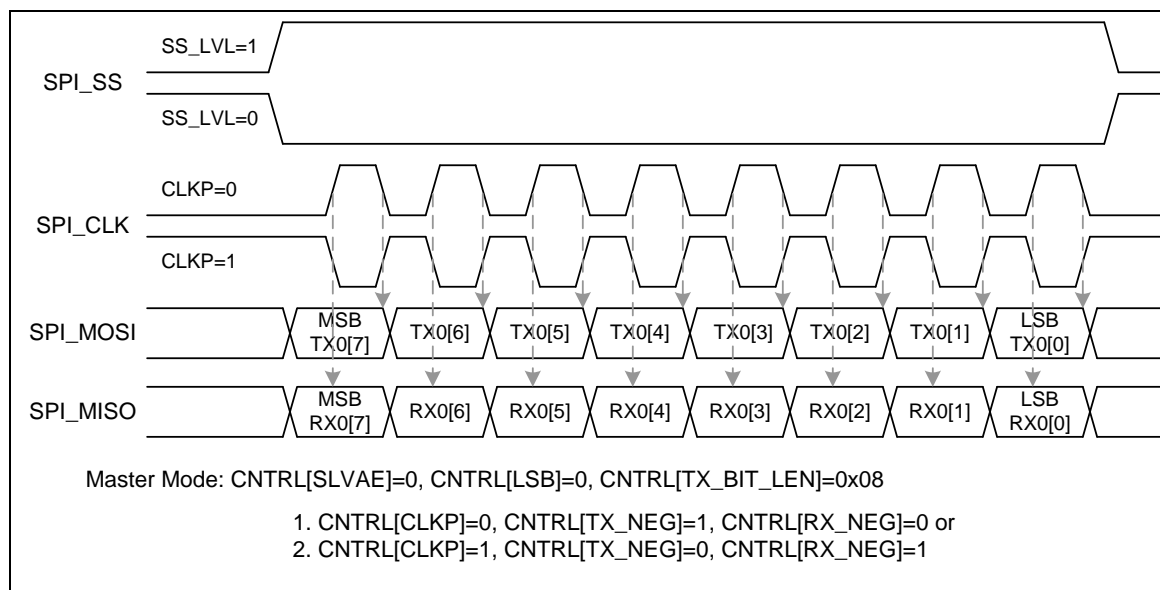


Figure 20-8 SPI Timing in Master Mode

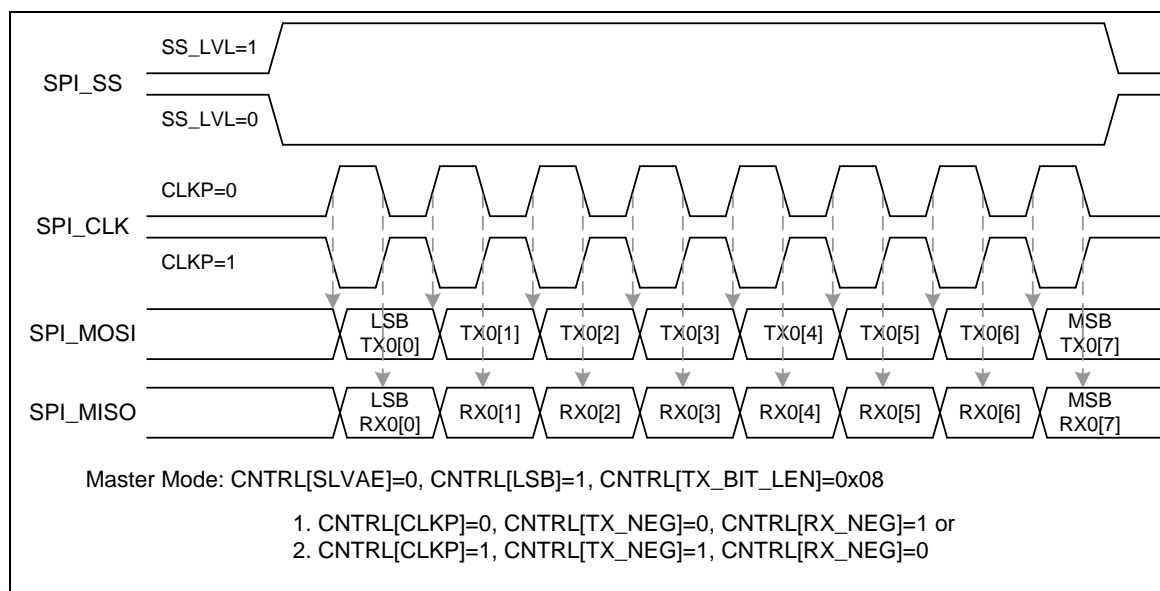


Figure 20-9 SPI Timing in Master Mode (Alternate Phase of SPI\_CLK)

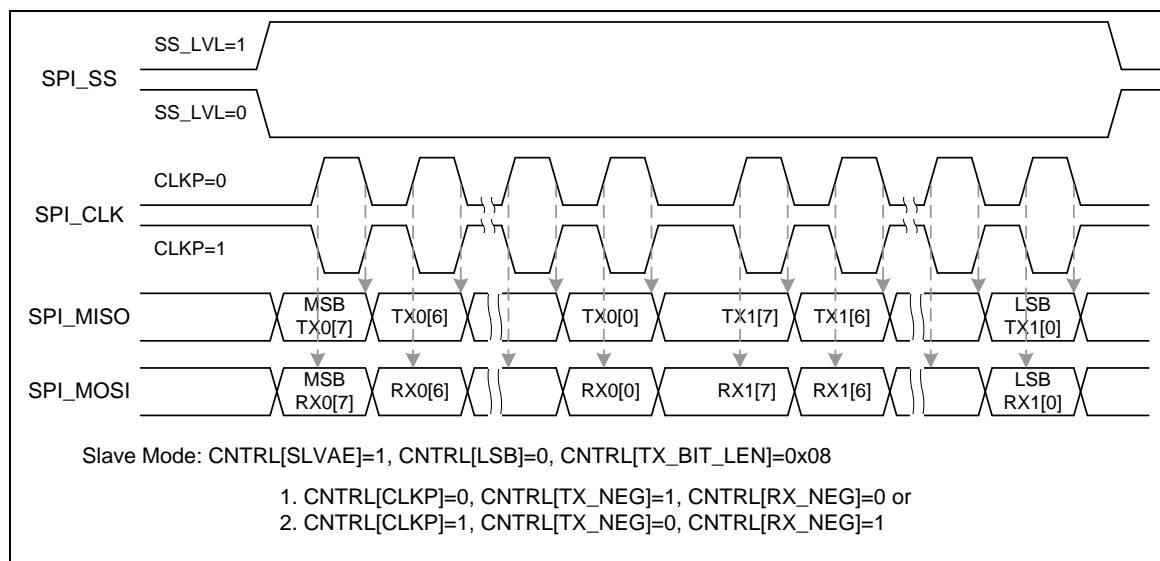


Figure 20-10 SPI Timing in Slave Mode

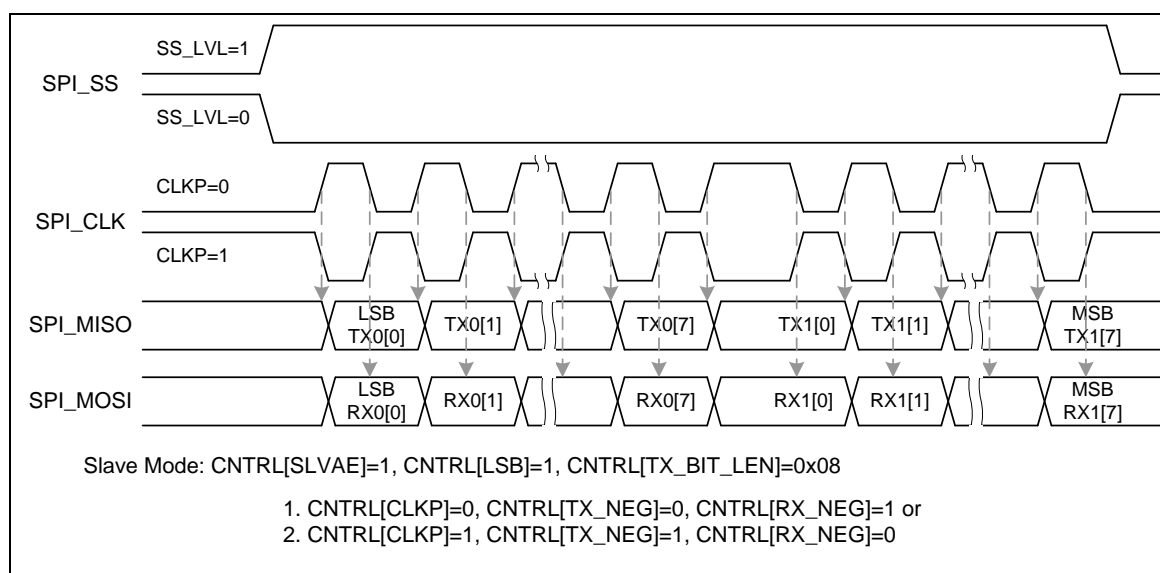


Figure 20-11 SPI Timing in Slave Mode (Alternate Phase of SPI\_CLK)

## 20.6 Programming Examples

**Example 1:** The SPI controller is set as a master to access an off-chip slave device with the following specifications:

- Data bit is latched on positive edge of SPI clock.
- Data bit is driven on negative edge of SPI clock.
- Data is transferred from MSB first.
- SPI\_CLK is idle at low state.
- Only one byte of data to be transmitted/received in a transaction.
- Uses the first SPI slave select pin to connect with an off-chip slave device. The slave select signal is active low.

The operation flow is as follows.

- 2) Set the DIVIDER (SPI\_DIVIDER [7:0]) register to determine the output frequency of SPI clock.
- 3) Write the SPI\_SSR register a proper value for the related settings of Master mode:
  1. Disable the Automatic Slave Select bit AUTOSS(SPI\_SSR[3] = 0).
  2. Select low level trigger output of slave select signal in the Slave Select Active Level bit SS\_LVL (SPI\_SSR[2] = 0) and Slave Select Level Trigger bit SS\_LTRIG (SPI\_SSR[4] = 1).
  3. Select slave select signal to be output active at the I/O pin by setting the Slave Select Register bits SSR[0] (SPI\_SSR[0]) to active the off-chip slave device.
- 4) Write the related settings into the SPI\_CNTRL register to control the SPI master actions
  1. Set this SPI controller as master device in SLAVE bit (SPI\_CNTRL[18] = 0).
  2. Force the SPI clock idle state at low in CLKP bit (SPI\_CNTRL[11] = 0).
  3. Select data transmitted at negative edge of SPI clock in TX\_NEG bit (SPI\_CNTRL[2] = 1).
  4. Select data latched at positive edge of SPI clock in RX\_NEG bit (SPI\_CNTRL[1] = 0).
  5. Set the bit length of word transfer as 8-bit in TX\_BIT\_LEN bit field. (SPI\_CNTRL[7:3] = 0x08).
  6. Set MSB transfer first in MSB bit (SPI\_CNTRL[10] = 0).
- 5) If this SPI master attempts to transmit (write) one byte data to the off-chip slave device, write the byte data that will be transmitted into the SPI\_TX register.
- 6) If this SPI master just only attempts to receive (read) one byte data from the off-chip slave device and does not care what data will be transmitted, the SPI\_TX register does not need to be updated by software.
- 7) Enable the GO\_BUSY bit (SPI\_CNTRL [0] = 1) to start the data transfer with the SPI interface.
- 8) Waiting for SPI interrupt (if the Interrupt Enable IE bit is set) or just polling the GO\_BUSY bit till it is cleared to 0 by hardware automatically.
- 9) Read out the received one byte data from SPI\_RX[7:0].
- 10) Go to 4) to continue another data transfer or set SSR [0] to 0 to inactivate the off-chip slave device.

**Example 2:** The SPI controller is set as a slave device and connects with an off-chip master device. The off-chip master device communicates with the on-chip SPI slave controller through the SPI interface with the following specifications:

- Data bit is latched on positive edge of SPI clock.
- Data bit is driven on negative edge of SPI clock.
- Data is transferred from LSB first.
- SPI\_CLK is idle at high state.
- Only one byte of data to be transmitted/received in a transaction.
- Slave select signal is high level trigger.

The operation flow is as follows.

- 1) Write the SPI\_SSR register a proper value for the related settings of Slave mode:  
Select high level and level trigger for the input of slave select signal by setting the Slave Select Active Level bit SS\_LVL (SPI\_SSR[2] = 1) and the Slave Select Level Trigger bit SS\_LTRIG (SPI\_SSR[4] = 1).
- 2) Write the related settings into the SPI\_CNTRL register to control this SPI slave actions
  1. Set the SPI controller as slave device in SLAVE bit (SPI\_CNTRL[18] = 1).
  2. Select the SPI clock idle state at high in CLKP bit (SPI\_CNTRL[11] = 1).
  3. Select data transmitted at negative edge of SPI clock in TX\_NEG bit (SPI\_CNTRL[2] = 1).
  4. Select data latched at positive edge of SPI clock in RX\_NEG bit (SPI\_CNTRL[1] = 0).
  5. Set the bit length of word transfer as 8-bit in TX\_BIT\_LEN bit field (SPI\_CNTRL[7:3] = 0x08).
  6. Set LSB transfer first in LSB bit (SPI\_CNTRL[10] = 1).
- 3) If this SPI slave attempts to transmit (be read) one byte data to the off-chip master device, write the byte data that will be transmitted into the SPI\_TX register.
- 4) If this SPI slave just only attempts to receive (be written) one byte data from the off-chip master device and does not care what data will be transmitted, the SPI\_TX register does not need to be updated by software.
- 5) Enable the GO\_BUSY bit (SPI\_CNTRL[0] = 1) to wait for the slave select trigger input and SPI clock input from the off-chip master device to start the data transfer at the SPI interface.
- 6) Waiting for SPI interrupt (if the Interrupt Enable IE bit is set), or just polling the GO\_BUSY bit till it is cleared to 0 by hardware automatically.
- 7) Read out the received one byte data from SPI\_RX[7:0].
- 8) Go to 3) to continue another data transfer or stop data transfer.



### 20.7 Register Map

R: read only, W: write only, R/W: both read and write

| Register   | Offset       | R/W | Description                   | Reset Value |
|--|--------------|-----|-------------------------------|-------------|
| <b>SPI Base Address:</b><br><b>SPI0_BA = 0x4003_0000</b><br><b>SPI1_BA = 0x4003_4000</b><br><b>SPI2_BA = 0x4013_0000</b> |              |     |                               |             |
| <b>SPI_CNTRL</b><br>x=0,1,2  | SPIx_BA+0x00 | R/W | Control and Status Register   | 0x0500_3004 |
| <b>SPI_DIVIDER</b><br>x=0,1,2  | SPIx_BA+0x04 | R/W | Clock Divider Register        | 0x0000_0000 |
| <b>SPI_SSR</b><br>x=0,1,2  | SPIx_BA+0x08 | R/W | Slave Select Register         | 0x0000_0000 |
| <b>SPI_RX</b><br>x=0,1,2   | SPIx_BA+0x10 | R   | Data Receive Register         | 0x0000_0000 |
| <b>SPI_TX</b><br>x=0,1,2   | SPIx_BA+0x20 | W   | Data Transmit Register        | 0x0000_0000 |
| <b>SPI_CNTRL2</b><br>x=0,1,2   | SPIx_BA+0x3C | R/W | Control and Status Register 2 | 0x0000_1000 |
| <b>SPI_FIFO_CTL</b><br>x=0,1,2   | SPIx_BA+0x40 | R/W | SPI FIFO Control Register     | 0x4400_0000 |
| <b>SPI_STATUS</b><br>x=0,1,2   | SPIx_BA+0x44 | R/W | SPI Status Register           | 0x0500_0000 |

### 20.8 Register Description

#### SPI Control and Status Register (SPI\_CNTRL)

| Register  | Offset       | R/W | Description                 | Reset Value |
|-----------|--------------|-----|-----------------------------|-------------|
| SPI_CNTRL | SPIx_BA+0x00 | R/W | Control and Status Register | 0x0500_3004 |

| 31         | 30 | 29   | 28       | 27      | 26       | 25       | 24       |
|------------|----|------|----------|---------|----------|----------|----------|
| Reserved   |    |      |          | TX_FULL | TX_EMPTY | RX_FULL  | RX_EMPTY |
| 23         | 22 | 21   | 20       | 19      | 18       | 17       | 16       |
| Reserved   |    | FIFO | Reserved | REORDER | SLAVE    | IE       | IF       |
| 15         | 14 | 13   | 12       | 11      | 10       | 9        | 8        |
| SP_CYCLE   |    |      |          | CLKP    | LSB      | Reserved |          |
| 7          | 6  | 5    | 4        | 3       | 2        | 1        | 0        |
| TX_BIT_LEN |    |      |          |         | TX_NEG   | RX_NEG   | GO_BUSY  |

| Bits    | Description |   |
|---------|-------------|---|
| [31:28] | Reserved    | Reserved  |
| [27]    | TX_FULL     | <b>Transmit FIFO Buffer Full Indicator (Read Only)</b><br>It is a mutual mirror bit of SPI_STATUS[27].<br>1 = Transmit FIFO buffer is full.<br>0 = Transmit FIFO buffer is not full.    |
| [26]    | TX_EMPTY    | <b>Transmit FIFO Buffer Empty Indicator (Read Only)</b><br>It is a mutual mirror bit of SPI_STAUTS[26].<br>1 = Transmit FIFO buffer is empty.<br>0 = Transmit FIFO buffer is not empty. |
| [25]    | RX_FULL     | <b>Receive FIFO Buffer Full Indicator (Read Only)</b><br>It is a mutual mirror bit of SPI_STATUS[25].<br>1 = Receive FIFO buffer is full.<br>0 = Receive FIOF buffer is not full.       |
| [24]    | RX_EMPTY    | <b>Receive FIFO Buffer Empty Indicator (Read Only)</b><br>It is a mutual mirror bit of SPI_CNTRL[24].<br>1 = Receive FIFO buffer is empty.<br>0 = Receive FIFO buffer is not empty.     |
| [23:22] | Reserved    | Reserved<br><b>Note:</b> These bits should always keep at 0, or SPI will work unpredictable   |
| [21]    | FIFO        | <b>FIFO Mode Enable</b><br>1 = FIFO mode Enabled.   |

|         |          |  |
|---------|----------|--|
|         |          | <p>0 = FIFO mode Disabled.</p> <p><b>Note:</b></p> <ol style="list-style-type: none"> <li>Before enabling FIFO mode, the other related settings should be set in advance.</li> <li>In Master mode, if the FIFO mode is enabled, the GO_BUSY bit will be set to 1 automatically after writing data to the transmit FIFO buffer; the GO_BUSY bit will be cleared to 0 automatically when the SPI controller is in idle. If all data stored at transmit FIFO buffer are sent out, the TX_EMPTY bit will be set to 1 and the GO_BUSY bit will be cleared to 0.</li> </ol>  |
| [20]    | Reserved | Reserved   |
| [19]    | REORDER  | <p><b>Byte Reorder Function Enable</b></p> <p>1 = Byte reorder function Enabled. A byte suspend interval will be inserted among each byte. The period of the byte suspend interval depends on the setting of SP_CYCLE.</p> <p>0 = Byte reorder function Disabled.</p> <p><b>Note:</b></p> <ol style="list-style-type: none"> <li>Byte reorder function is only available if TX_BIT_LEN is defined as 16, 24, and 32 bits.</li> <li>In Slave mode with level-trigger configuration, the slave select pin must be kept at active state during the byte suspend interval.</li> </ol>  |
| [18]    | SLAVE    | <p><b>Slave Mode Enable</b></p> <p>1 = Slave mode.</p> <p>0 = Master mode.</p>   |
| [17]    | IE       | <p><b>Unit Transfer Interrupt Enable</b></p> <p>1 = SPI unit transfer interrupt Enabled.</p> <p>0 = SPI unit transfer interrupt Disabled.</p>  |
| [16]    | IF       | <p><b>Unit Transfer Interrupt Flag</b></p> <p>1 = SPI controller has finished one unit transfer.</p> <p>0 = No transaction has been finished since this bit was cleared to 0.</p> <p><b>Note:</b> This bit will be cleared by writing 1 to itself.</p>   |
| [15:12] | SP_CYCLE | <p><b>Suspend Interval (Master Only)</b></p> <p>The four bits provide configurable suspend interval between two successive transmit/receive transaction in a transfer. The definition of the suspend interval is the interval between the last clock edge of the preceding transaction word and the first clock edge of the following transaction word. The default value is 0x3. The period of the suspend interval is obtained according to the following equation.</p> $(SP\_CYCLE[3:0] + 0.5) * \text{period of SPI clock cycle}$ <p>Example:</p> <p>SP_CYCLE = 0x0 ... 0.5 SPI clock cycle</p> <p>SP_CYCLE = 0x1 ... 1.5 SPI clock cycle</p> <p>.....</p> <p>SP_CYCLE = 0xE ... 14.5 SPI clock cycle</p> <p>SP_CYCLE = 0xF ... 15.5 SPI clock cycle</p> |
| [11]    | CLKP     | <p><b>Clock Polarity</b></p> <p>1 = SPI clock is idle high.</p> <p>0 = SPI clock is idle low.</p>  |
| [10]    | LSB      | <p><b>Send LSB First</b></p> <p>1 = The LSB, bit 0 of the SPI TX0/1 register, is sent first to the SPI data output pin, and the first bit received from the SPI data input pin will be put in the LSB position of</p>  |

|       |                   |  |
|-------|-------------------|--|
|       |                   | <p>the RX register (bit 0 of SPI_RX).</p> <p>0 = The MSB, which bit of transmit/receive register depends on the setting of TX_BIT_LEN, is transmitted/received first.</p>  |
| [9:8] | <b>Reserved</b>   | Reserved   |
| [7:3] | <b>TX_BIT_LEN</b> | <p><b>Transmit Bit Length</b></p> <p>This field specifies how many bits can be transmitted / received in one transaction. The minimum bit length is 8 bits and can up to 32 bits.</p> <p>TX_BIT_LEN = 0x08 ... 8 bits</p> <p>TX_BIT_LEN = 0x09 ... 9 bits</p> <p>.....</p> <p>TX_BIT_LEN = 0x1F ... 31 bits</p> <p>TX_BIT_LEN = 0x00 ... 32 bits</p>   |
| [2]   | <b>TX_NEG</b>     | <p><b>Transmit on Negative Edge</b></p> <p>1 = Transmitted data output signal is changed on the falling edge of SPI clock.</p> <p>0 = Transmitted data output signal is changed on the rising edge of SPI clock.</p>   |
| [1]   | <b>RX_NEG</b>     | <p><b>Receive on Negative Edge</b></p> <p>1 = Received data input signal is latched on the falling edge of SPI clock.</p> <p>0 = Received data input signal is latched on the rising edge of SPI clock.</p>  |
| [0]   | <b>GO_BUSY</b>    | <p><b>SPI Transfer Control Bit and Busy Status</b></p> <p>1 = In Master mode, writing 1 to this bit to start the SPI data transfer; in Slave mode, writing 1 to this bit indicates that the slave is ready to communicate with a master.</p> <p>0 = Data transfer stopped.</p> <p>If FIFO mode is disabled, during the data transfer, this bit keeps the value of 1. As the transfer is finished, this bit will be cleared automatically. Software can read this bit to check if the SPI is in busy status.</p> <p>In FIFO mode, this bit will be controlled by hardware. Software should not modify this bit. In Slave mode, this bit always returns 1 when this register is read by software. In Master mode, this bit reflects the busy or idle status of SPI.</p> <p><b>Note:</b><br/>When FIFO mode is disabled, all configurations should be set before writing 1 to this GO_BUSY bit.</p> |

**SPI Divider Register (SPI\_DIVIDER)**

| Register           | Offset       | R/W | Description            | Reset Value |
|--------------------|--------------|-----|------------------------|-------------|
| <b>SPI_DIVIDER</b> | SPIx_BA+0x04 | R/W | Clock Divider Register | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved     |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved     |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DIVIDER[7:0] |    |    |    |    |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | Reserved  |
| [7:0]  | DIVIDER     | <p><b>Clock Divider 1 Register</b></p> <p>The value in this field is the frequency divider for generating the SPI engine clock, <math>f_{spi\_eclk}</math>, and the SPI serial clock of SPI master. The frequency is obtained according to the following equation.</p> $f_{spi\_eclk} = \frac{f_{system\_clock}}{(DIVIDER + 1) * 2}$ <p>where</p> <p><math>f_{spi\_clock\_src}</math> is the SPI engine clock source, which is defined in the CLKSEL1 register.</p> |

### SPI Slave Select Register (SPI SSR)

| Register | Offset       | R/W | Description           | Reset Value |
|----------|--------------|-----|-----------------------|-------------|
| SPI_SSR  | SPIx_BA+0x08 | R/W | Slave Select Register | 0x0000_0000 |

|          |    |            |          |        |        |          |     |
|----------|----|------------|----------|--------|--------|----------|-----|
| 31       | 30 | 29         | 28       | 27     | 26     | 25       | 24  |
| Reserved |    |            |          |        |        |          |     |
| 23       | 22 | 21         | 20       | 19     | 18     | 17       | 16  |
| Reserved |    |            |          |        |        |          |     |
| 15       | 14 | 13         | 12       | 11     | 10     | 9        | 8   |
| Reserved |    |            |          |        |        |          |     |
| 7        | 6  | 5          | 4        | 3      | 2      | 1        | 0   |
| Reserved |    | LTRIG_FLAG | SS_LTRIG | AUTOSS | SS_LVL | Reserved | SSR |

| Bits   | Description |  |
|--------|-------------|--|
| [31:6] | Reserved    | Reserved   |
| [5]    | LTRIG_FLAG  | <b>Level Trigger Accomplish Flag</b><br>In Slave mode, this bit indicates whether the received bit number meets the requirement or not after the current transaction done.<br>1 = Transferred bit length meets the specified requirement which defined in TX_BIT_LEN.<br>0 = Transferred bit length of one transaction does not meet the specified requirement.<br><b>Note:</b> This bit is READ only. As the GO_BUSY bit is set to 1 by software, the LTRIG_FLAG will be cleared to 0 after 4 SPI engine clock periods plus 1 system clock period. In FIFO mode, this bit has no meaning. |
| [4]    | SS_LTRIG    | <b>Slave Select Level Trigger Enable (Slave Only)</b><br>1 = Slave select signal is level-trigger. The SS_LVL bit decides the signal is active low or active high.<br>0 = Slave select signal is edge-trigger. This is the default value. The SS_LVL bit decides the signal is active after a falling-edge or rising-edge.   |
| [3]    | AUTOSS      | <b>Automatic Slave Select Function Enable (Master Only)</b><br>1 = If this bit is set, SPI_SS signals will be generated automatically. It means that device/slave select signal, which is set in SPI_SSR[0], will be asserted by the SPI controller when transmit/receive is started, and will be de-asserted after each transmit/receive is finished.<br>0 = If this bit is cleared, slave select signals will be asserted/de-asserted by setting /clearing the corresponding bits of SPI_SSR[0].   |
| [2]    | SS_LVL      | <b>Slave Select Active Level</b><br>This bit defines the active status of slave select signal (SPI_SS).<br>1 = The slave select signal SPI_SS is active on high-level/rising-edge.<br>0 = The slave select signal SPI_SS is active on low-level/falling-edge.  |
| [1]    | Reserved    | Reserved   |
| [0]    | SSR         | <b>Slave Select Control Bits (Master Only)</b>   |

|  |  |   |
|--|--|---|
|  |  | <p>If AUTOSS bit is cleared, writing 1 to any bit of this field sets the proper SPI_SS line to an active state and writing 0 sets the line back to inactive state.</p> <p>If the AUTOSS bit is set, writing 0 to any bit location of this field will keep the corresponding SPI_SS line at inactive state; writing 1 to any bit location of this field will select appropriate SPI_SS line to be automatically driven to active state for the duration of the transmit/receive, and will be driven to inactive state for the rest of the time. The active state of SPI_SS is specified in SS_LVL.</p> |
|--|--|---|

### SPI Data Receive Register (SPI\_RX)

| Register | Offset       | R/W | Description           | Reset Value |
|----------|--------------|-----|-----------------------|-------------|
| SPI_RX   | SPIx_BA+0x10 | R   | Data Receive Register | 0x0000_0000 |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| RX[31:24] |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| RX[23:16] |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| RX[15:8]  |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| RX[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [31:0] | <p><b>Data Receive Register</b></p> <p>The data receive register holds the datum received from SPI data input pin. If FIFO mode is disabled, the last received data can be accessed through software by reading this register. If the FIFO bit is set as 1 and the RX_EMPTY bit, SPI_CNTRL[24] or SPI_STATUS[24], is not set to 1, the receive FIFO buffer can be accessed through software by reading this register. This is a read-only register.</p> |



**SPI Data Transmit Register (SPI\_TX)**

| Register | Offset       | R/W | Description            | Reset Value |
|----------|--------------|-----|------------------------|-------------|
| SPI_TX   | SPIx_BA+0x20 | W   | Data Transmit Register | 0x0000_0000 |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| TX[31:24] |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| TX[23:16] |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| TX[15:8]  |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TX[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description |  |
|--------|-------------|--|
| [31:0] | TX          | <p><b>Data Transmit Register</b></p> <p>The data transmit registers hold the data to be transmitted in the next transfer. The number of valid bits depends on the setting of transmit bit length field of the SPI_CNTRL register.</p> <p>For example, if TX_BIT_LEN is set to 0x08, the bits TX[7:0] will be transmitted in next transfer. If TX_BIT_LEN is set to 0x00, the SPI controller will perform a 32-bit transfer.</p> <p><b>Note:</b> When the SPI controller is configured as a slave device and FIFO mode is disabled, if the SPI controller attempts to transmit data to a master, the transmit data register should be updated by software before setting the GO_BUSY bit to 1</p> |

**SPI Control and Status Register 2 (SPI\_CNTRL2)**

| Register   | Offset       | R/W | Description                   | Reset Value |
|------------|--------------|-----|-------------------------------|-------------|
| SPI_CNTRL2 | SPIx_BA+0x3C | R/W | Control and Status Register 2 | 0x0000_1000 |

|          |    |    |    |                  |            |           |            |
|----------|----|----|----|------------------|------------|-----------|------------|
| 31       | 30 | 29 | 28 | 27               | 26         | 25        | 24         |
| Reserved |    |    |    |                  |            |           |            |
| 23       | 22 | 21 | 20 | 19               | 18         | 17        | 16         |
| Reserved |    |    |    |                  |            |           | SS_INT_OPT |
| 15       | 14 | 13 | 12 | 11               | 10         | 9         | 8          |
| Reserved |    |    |    | SLV_START_INTSTS | SSTA_INTEN | SLV_ABORT | NOSLVSEL   |
| 7        | 6  | 5  | 4  | 3                | 2          | 1         | 0          |
| Reserved |    |    |    |                  |            |           |            |

| Bits    | Description      |   |
|---------|------------------|---|
| [31:17] | Reserved         | Reserved  |
| [16]    | SS_INT_OPT       | <b>Slave Select Inactive Interrupt Option</b><br>This setting is only available if the SPI controller is configured as level trigger slave device.<br>1 = As the slave select signal goes to inactive level, the IF bit will be set to 1.<br>0 = As the slave select signal goes to inactive level, the IF bit will NOT be set to 1.  |
| [15:14] | Reserved         | Reserved  |
| [13]    | Reserved         | Reserved<br><b>Note:</b> This bit should always keep at 0, or SPI will work unpredictable   |
| [12]    | Reserved         | Reserved  |
| [11]    | SLV_START_INTSTS | <b>Slave 3-Wire Mode Start Interrupt Status</b><br>This bit indicates if a transaction has started in Slave 3-wire mode. It is a mutual mirror bit of SPI_STATUS[11].<br>1 = A transaction has started in Slave 3-wire mode. It will be cleared automatically when a transaction is done or by writing 1 to this bit.<br>0 = Slave has not detected any SPI clock transition since the SSTA_INTEN bit was set to 1. |
| [10]    | SSTA_INTEN       | <b>Slave 3-Wire Mode Start Interrupt Enable</b><br>Used to enable interrupt when the transfer has started in Slave 3-wire mode. If there is no transfer done interrupt over the time period which is defined by user after the transfer start, the user can set the SLV_ABORT bit to force the transfer done.<br>1 = Transaction start interrupt Enabled. It will be cleared to 0 as the current                    |

|       |           |   |
|-------|-----------|---|
|       |           | transfer is done or the SLV_START_INTSTS bit is cleared.<br>0 = Transaction start interrupt Disabled.   |
| [9]   | SLV_ABORT | <b>Slave 3-Wire Mode Abort Control</b><br>In normal operation, there is an interrupt event when the received data meet the required bits which defined in TX_BIT_LEN.<br>If the received bits are less than the requirement and there is no more SPI clock input over the one transfer time in Slave 3-wire mode, the user can set this bit to force the current transfer done and then the user can get a transfer done interrupt event.<br><b>Note:</b> This bit will be cleared to 0 automatically by hardware after it is set to 1 by software. |
| [8]   | NOSLVSEL  | <b>Slave 3-Wire Mode Enable</b><br>This is used to ignore the slave select signal in Slave mode. The SPI controller can work with 3-wire interface including SPI_CLK, SPI_MISO, and SPI_MOSI.<br>1 = 3-wire bi-direction interface.<br>0 = 4-wire bi-direction interface.<br><b>Note:</b> In Slave 3-wire mode, the SS_LTRIG, SPI_SSR[4] will be set as 1 automatically.  |
| [7:0] | Reserved  | Reserved  |

### SPI FIFO Control Register (SPI\_FIFO\_CTL)

| Register     | Offset       | R/W | Description               | Reset Value |
|--------------|--------------|-----|---------------------------|-------------|
| SPI_FIFO_CTL | SPIx_BA+0x40 | R/W | SPI FIFO Control Register | 0x4400_0000 |

| 31       | 30           | 29            | 28       | 27       | 26           | 25     | 24     |
|----------|--------------|---------------|----------|----------|--------------|--------|--------|
| Reserved | TX_THRESHOLD |               |          | Reserved | RX_THRESHOLD |        |        |
| 23       | 22           | 21            | 20       | 19       | 18           | 17     | 16     |
| Reserved |              | TIMEOUT_INTEN | Reserved |          |              |        |        |
| 15       | 14           | 13            | 12       | 11       | 10           | 9      | 8      |
| Reserved |              |               |          |          |              |        |        |
| 7        | 6            | 5             | 4        | 3        | 2            | 1      | 0      |
| Reserved | RXOV_INTEN   | Reserved      |          | TX_INTEN | RX_INTEN     | TX_CLR | RX_CLR |

| Bits    | Description   |   |
|---------|---------------|---|
| [31]    | Reserved      | Reserved  |
| [30:28] | TX_THRESHOLD  | <b>Transmit FIFO Threshold</b><br>If the valid data count of the transmit FIFO buffer is less than or equal to the TX_THRESHOLD setting, the TX_INTSTS bit will be set to 1, else the TX_INTSTS bit will be cleared to 0. |
| [27]    | Reserved      | Reserved  |
| [26:24] | RX_THRESHOLD  | <b>Receive FIFO Threshold</b><br>If the valid data count of the receive FIFO buffer is larger than the RX_THRESHOLD setting, the RX_INTSTS bit will be set to 1, else the RX_INTSTS bit will be cleared to 0.             |
| [23:22] | Reserved      | Reserved  |
| [21]    | TIMEOUT_INTEN | <b>Receive FIFO Time-out Interrupt Enable</b><br>1 = Time-out interrupt Enabled.<br>0 = Time-out interrupt Disabled.  |
| [20:7]  | Reserved      | Reserved  |
| [6]     | RXOV_INTEN    | <b>Receive FIFO Overrun Interrupt Enable</b><br>1 = Receive FIFO overrun interrupt Enabled.<br>0 = Receive FIFO overrun interrupt Disabled.   |
| [5:4]   | Reserved      | Reserved  |
| [3]     | TX_INTEN      | <b>Transmit Threshold Interrupt Enable</b>  |

|     |                 |  |
|-----|-----------------|--|
|     |                 | 1 = TX threshold interrupt Enabled.<br>0 = TX threshold interrupt Disabled.  |
| [2] | <b>RX_INTEN</b> | <b>Receive Threshold Interrupt Enable</b><br>1 = RX threshold interrupt Enabled.<br>0 = RX threshold interrupt Disabled.   |
| [1] | <b>TX_CLR</b>   | <b>Clear Transmit FIFO Buffer</b><br>1 = Clear transmit FIFO buffer. The TX_FULL flag will be cleared to 0 and the TX_EMPTY flag will be set to 1. This bit will be cleared to 0 by hardware after it is set to 1 by software.<br>0 = No effect. |
| [0] | <b>RX_CLR</b>   | <b>Clear Receive FIFO Buffer</b><br>1 = Clear receive FIFO buffer. The RX_FULL flag will be cleared to 0 and the RX_EMPTY flag will be set to 1. This bit will be cleared to 0 by hardware after it is set to 1 by software.<br>0 = No effect.   |

### SPI Status Register (SPI\_STATUS)

| Register   | Offset       | R/W | Description         | Reset Value |
|------------|--------------|-----|---------------------|-------------|
| SPI_STATUS | SPIx_BA+0x44 | R/W | SPI Status Register | 0x0500_0000 |

|               |    |    |           |                  |            |          |           |
|---------------|----|----|-----------|------------------|------------|----------|-----------|
| 31            | 30 | 29 | 28        | 27               | 26         | 25       | 24        |
| TX_FIFO_COUNT |    |    |           | TX_FULL          | TX_EMPTY   | RX_FULL  | RX_EMPTY  |
| 23            | 22 | 21 | 20        | 19               | 18         | 17       | 16        |
| Reserved      |    |    | TIMEOUT   | Reserved         |            |          | IF        |
| 15            | 14 | 13 | 12        | 11               | 10         | 9        | 8         |
| RX_FIFO_COUNT |    |    |           | SLV_START_INTSTS | Reserved   |          |           |
| 7             | 6  | 5  | 4         | 3                | 2          | 1        | 0         |
| Reserved      |    |    | TX_INTSTS | Reserved         | RX_OVERRUN | Reserved | RX_INTSTS |

| Bits    | Description   |  |
|---------|---------------|--|
| [31:28] | TX_FIFO_COUNT | <b>Transmit FIFO Data Count (Read Only)</b><br>This bit field indicates the valid data count of transmit FIFO buffer.  |
| [27]    | TX_FULL       | <b>Transmit FIFO Buffer Full Indicator (Read Only)</b><br>It is a mutual mirror bit of SPI_CNTRL[27].<br>1 = Transmit FIFO buffer is full.<br>0 = Transmit FIFO buffer is not full.  |
| [26]    | TX_EMPTY      | <b>Transmit FIFO Buffer Empty Indicator (Read Only)</b><br>It is a mutual mirror bit of SPI_CNTRL[26].<br>1 = Transmit FIFO buffer is empty.<br>0 = Transmit FIFO buffer is not empty.   |
| [25]    | RX_FULL       | <b>Receive FIFO Buffer Empty Indicator (Read Only)</b><br>It is a mutual mirror bit of SPI_CNTRL[24].<br>1 = Receive FIFO buffer is empty.<br>0 = Receive FIFO buffer is not empty.  |
| [24]    | RX_EMPTY      | <b>Receive FIFO Buffer Empty Indicator (Read Only)</b><br>It is a mutual mirror bit of SPI_CNTRL[24].<br>1 = Receive FIFO buffer is empty.<br>0 = Receive FIFO buffer is not empty.  |
| [23:21] | Reserved      | Reserved   |
| [20]    | TIMEOUT       | <b>Time-out Interrupt Flag</b><br>1 = Receive FIFO buffer is not empty and no read operation on receive FIFO buffer over 64 SPI clock period in Master mode or over 576 SPI engine clock period in Slave mode. When the received FIFO buffer is read by software, the time-out |

|         |                  |   |
|---------|------------------|---|
|         |                  | <p>status will be cleared automatically.</p> <p>0 = No receive FIFO time-out event.</p> <p><b>Note:</b> This bit will be cleared by writing 1 to itself.</p>  |
| [19:17] | Reserved         | Reserved  |
| [16]    | IF               | <p><b>SPI Unit Transfer Interrupt Flag</b></p> <p>It is a mutual mirror bit of SPI_CNTRL[16].</p> <p>1 = SPI controller has finished one unit transfer.</p> <p>0 = No transaction has been finished since this bit was cleared to 0.</p> <p><b>Note:</b> This bit will be cleared by writing 1 to itself.</p>   |
| [15:12] | RX_FIFO_COUNT    | <p><b>Receive FIFO Data Count (Read Only)</b></p> <p>This bit field indicates the valid data count of receive FIFO buffer.</p>  |
| [11]    | SLV_START_INTSTS | <p><b>Slave Start Interrupt Status</b></p> <p>It is used to dedicate if a transaction has started in Slave 3-wire mode. It is a mutual mirror bit of SPI_CNTRL2[11].</p> <p>1 = A transaction has started in Slave 3-wire mode. It will be cleared as a transaction is done or by writing 1 to this bit.</p> <p>0 = Slave has not detected any SPI clock transition since the SSTA_INTEN bit was set to 1.</p>                    |
| [10:5]  | Reserved         | Reserved  |
| [4]     | TX_INTSTS        | <p><b>Transmit FIFO Threshold Interrupt Status (Read Only)</b></p> <p>1 = The valid data count within the transmit FIFO buffer is less than or equal to the setting value of TX_THRESHOLD.</p> <p>0 = The valid data count within the transmit FIFO buffer is larger than the setting value of TX_THRESHOLD.</p> <p><b>Note:</b> If TX_INTEN = 1 and TX_INTSTS = 1, the SPI controller will generate a SPI interrupt request.</p> |
| [3]     | Reserved         | Reserved  |
| [2]     | RX_OVERRUN       | <p><b>Receive FIFO Overrun Status</b></p> <p>When the receive FIFO buffer is full, the follow-up data will be dropped and this bit will be set to 1.</p> <p><b>Note:</b> This bit will be cleared by writing 1 to itself.</p>   |
| [1]     | Reserved         | Reserved  |
| [0]     | RX_INTSTS        | <p><b>Receive FIFO Threshold Interrupt Status (Read Only)</b></p> <p>1 = The valid data count within the receive FIFO buffer is larger than the setting value of RX_THRESHOLD.</p> <p>0 = The valid data count within the Rx FIFO buffer is smaller than or equal to the setting value of RX_THRESHOLD.</p> <p><b>Note:</b> If RX_INTEN = 1 and RX_INTSTS = 1, the SPI controller will generate a SPI interrupt request.</p>      |

## 21 I<sup>2</sup>C SERIAL INTERFACE

### 21.1 Overview

I<sup>2</sup>C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. The I<sup>2</sup>C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

Data is transferred between a Master and a Slave. Data bits transfer on the SCL and SDA lines are synchronously on a byte-by-byte basis. Each data byte is 8-bit long. There is one SCL clock pulse for each data bit with the MSB being transmitted first, and an acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (START or STOP). Please refer to the Figure 21-1 for more detailed I<sup>2</sup>C BUS Timing.

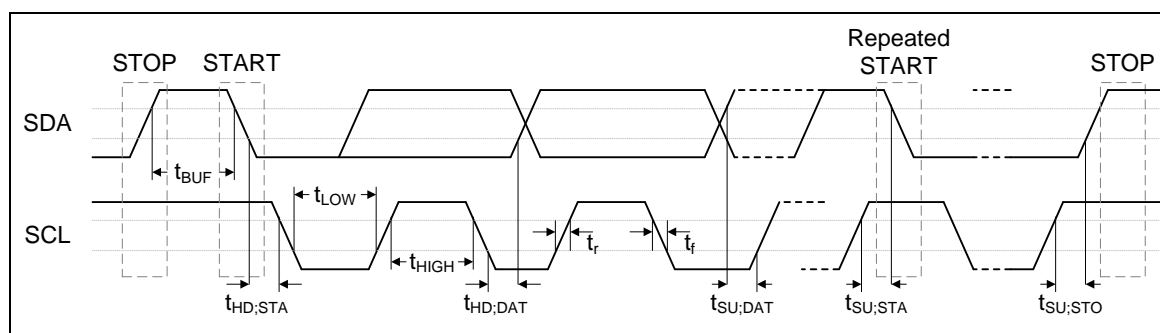


Figure 21-1 I<sup>2</sup>C Bus Timing

The device's on-chip I<sup>2</sup>C logic provides the serial interface that meets the I<sup>2</sup>C bus standard mode specification. The I<sup>2</sup>C port handles byte transfers autonomously. To enable this port, the bit ENS1 in I2CON should be set to '1'. The I<sup>2</sup>C H/W interfaces to the I<sup>2</sup>C bus via two pins: SDA and SCL. Pull-up resistor is needed for I<sup>2</sup>C operation as these SDA and SCL are open drain pins. When the I/O pins are used as I<sup>2</sup>C port, user must set the pins function to I<sup>2</sup>C in advance.



## 21.2 Features

The I<sup>2</sup>C bus uses two wires (SDA and SCL) to transfer information between devices connected to the bus. The main features of the bus are:

- Master/Slave mode
- Bidirectional data transfer between masters and slaves
- Multi-master bus (no central master)
- Arbitration between simultaneously transmitting masters without corruption of serial data on the bus
- Serial clock synchronization allows devices with different bit rates to communicate via one serial bus
- Serial clock synchronization can be used as a handshake mechanism to suspend and resume serial transfer
- Built-in a 14-bit time out counter requested the I<sup>2</sup>C interrupt if the I<sup>2</sup>C bus hangs up and timer-out counter overflows.
- External pull-up resistors are needed for high output
- Programmable clocks allow versatile rate control
- Supports 7-bit addressing mode
- Supports multiple address recognition ( four slave address with mask option)

## 21.3 Functional Description

### 21.3.1.1 I<sup>2</sup>C Protocol

Normally, a standard communication consists of four parts:

1. START or Repeated START signal generation
2. Slave address and R/W bit transfer
3. Data transfer
4. STOP signal generation

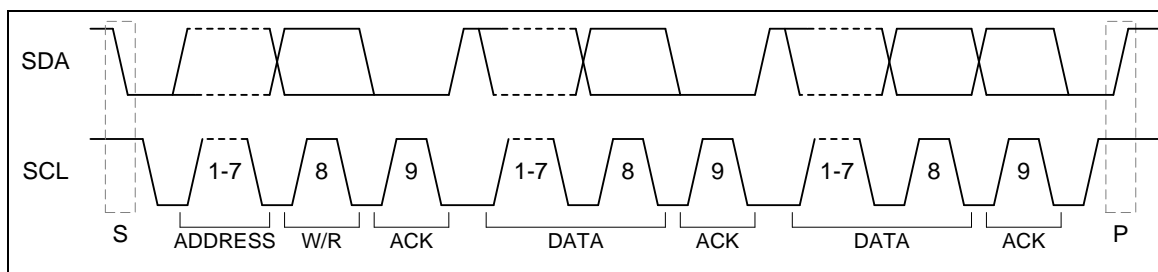


Figure 21-2 I<sup>2</sup>C Protocol

### 21.3.1.2 Data transfer on the I<sup>2</sup>C-bus

Figure 21-3 shows a master transmits data to slave. A master addresses a slave with a 7-bit address and 1-bit write index to denote master wants to transmit data to slave. The master keep transmitting data after slave returns acknowledge to master.

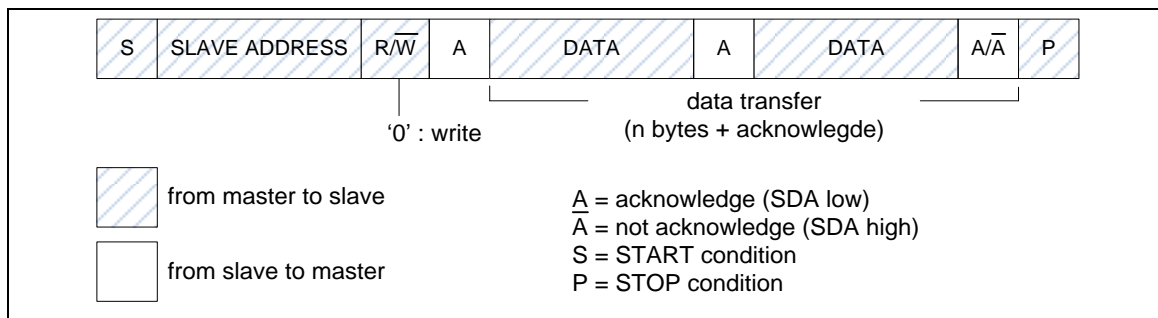


Figure 21-3 Master Transmits Data to Slave

Figure 21-4 shows a master read data from slave. A master addresses a slave with a 7-bit address and 1-bit read index to denote master wants to read data from slave. The slave will start transmitting data after slave returns acknowledge to master.

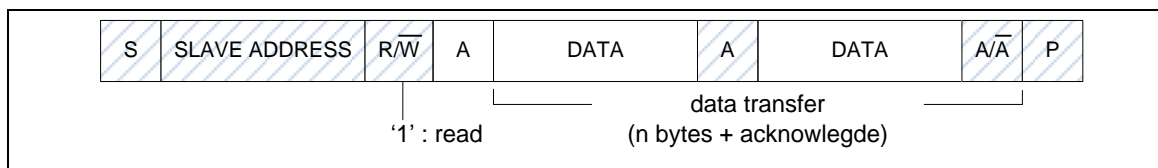


Figure 21-4 Master Reads Data from Slave

### 21.3.2 START or Repeated START signal

When the bus is free/idle, meaning no master device is engaging the bus (both SCL and SDA lines are high), a master can initiate a transfer by sending a START signal. A START signal, usually referred to as the S-bit, is defined as a HIGH to LOW transition on the SDA line while SCL is HIGH. The START signal denotes the beginning of a new data transmission.

A Repeated START (Sr) is no STOP signal between two START signals. The master uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

#### STOP signal

The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the P-bit, is defined as a LOW to HIGH transition on the SDA line while SCL is HIGH.

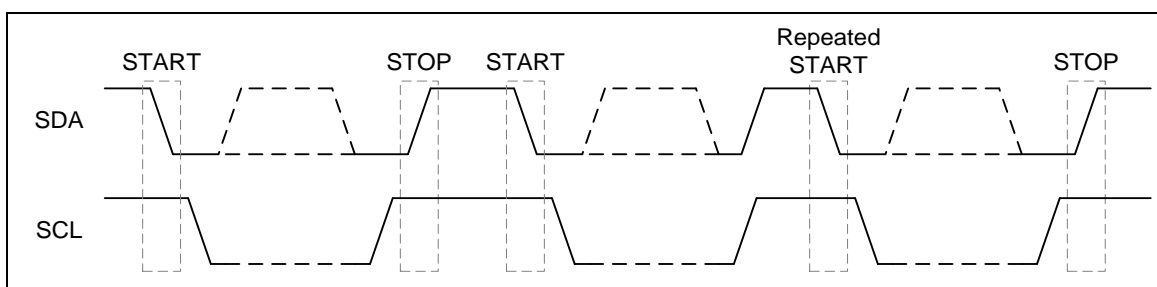


Figure 21-5 START and STOP Condition

### 21.3.3 Slave Address Transfer

The first byte of data transferred by the master immediately after the START signal is the slave address. This is a 7-bit calling address followed by a RW bit. The RW bit signals the slave the data transfer direction. No two slaves in the system can have the same address. Only the slave with an address that matches the one transmitted by the master will respond by returning an acknowledge bit by pulling the SDA low at the 9th SCL clock cycle.

### 21.3.4 Data Transfer

When slave receives a correct address with a RW bit, the data will follow RW bit specified to transfer. Each transferred byte is followed by an acknowledge bit on the 9th SCL clock cycle. If the slave signals a Not Acknowledge (NACK), the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

If the master, as the receiving device, does Not Acknowledge (NACK) the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal.

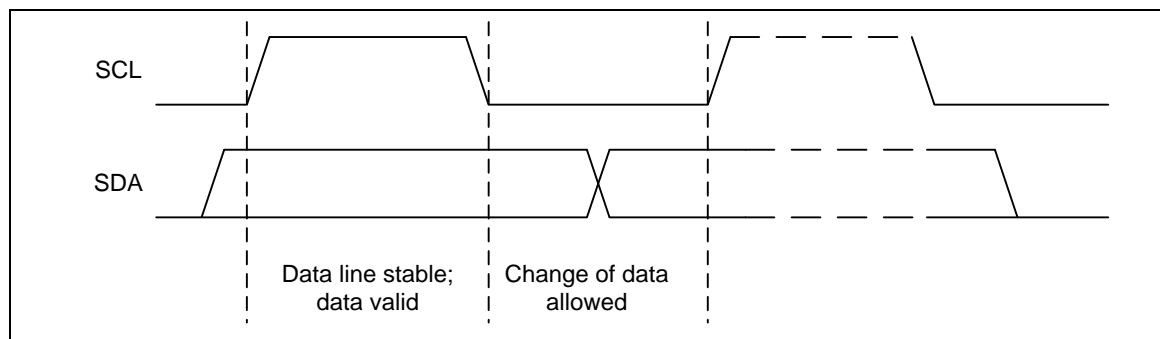


Figure 21-6 Bit Transfer on the I<sup>2</sup>C Bus

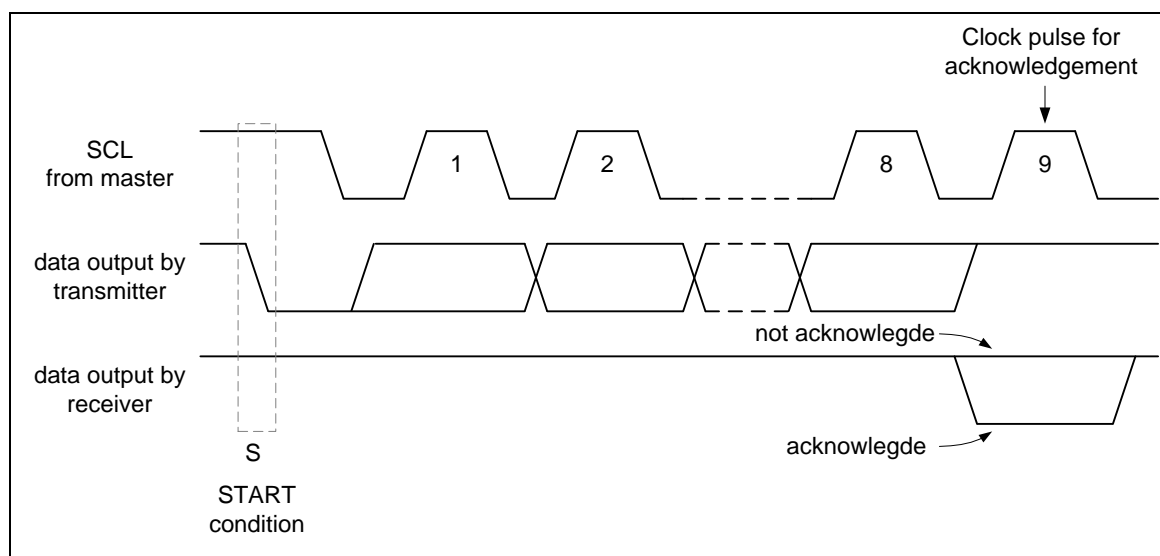


Figure 21-7 Acknowledge on the I<sup>2</sup>C Bus

## 21.4 Protocol Registers

The CPU interfaces connect to the I<sup>2</sup>C port through the following thirteen special function registers: I2CON (control register), I2CSTATUS (status register), I2CDAT (data register), I2CADDRn (address registers, n=0~3), I2CADMn (address mask registers, n=0~3), I2CLK (clock rate register) and I2CTOC (Time-out counter register). All bit 31~ bit 8 of these I<sup>2</sup>C special function registers are reserved. These bits do not have any functions and are all 0 if read them back.

When I<sup>2</sup>C port is enabled by setting ENS1 (I2CON [6]) to high, the internal states will be controlled by I2CON and I<sup>2</sup>C logic hardware. Once a new status code is generated and stored in I2CSTATUS, the I<sup>2</sup>C Interrupt Flag bit SI (I2CON [3]) will be set automatically. If the Enable Interrupt bit EI (I2CON [7]) is set at this time, the I<sup>2</sup>C interrupt will be generated. The bit field I2CSTATUS[7:3] stores the internal state code, the lowest 3 bits of I2CSTATUS are always zero and the content keeps stable until SI is cleared by software. The base address is 0x4002\_0000 and 0x4012\_0000.

### 21.4.1 Address Registers (I2CADDR)

I<sup>2</sup>C port is equipped with four slave address registers I2CADDRn (n=0~3). The contents of the register are irrelevant when I<sup>2</sup>C is in Master mode. In the Slave mode, the bit field I2CADDRn[7:1] must be loaded with the chip's own slave address. The I<sup>2</sup>C hardware will react if the contents of I2CADDRn are matched with the received slave address.

The I<sup>2</sup>C ports support the "General Call" function. If the GC bit (I2CADDRn [0]) is set the I<sup>2</sup>C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

When GC bit is set and the I<sup>2</sup>C is in Slave mode, it can receive the general call address by 00H after Master send general call address to I<sup>2</sup>C bus, then it will follow status of GC mode.

I<sup>2</sup>C bus controllers support multiple address recognition with four address mask registers I2CADMn (n=0~3). When the bit in the address mask register is set to one, it means the received corresponding address bit is don't care. If the bit is set to 0, that means the received corresponding register bit should be exact the same as address register.

### 21.4.2 Data Register (I2CDAT)

This register contains a byte of serial data to be transmitted or a byte which just has been received. The CPU can read from or write to this 8-bit (I2CDAT [7:0]) directly while it is not in the process of shifting a byte. When I<sup>2</sup>C is in a defined state and the serial interrupt flag (SI) is set data in I2CDAT [7:0] remains stable. While data is being shifted out, data on the bus is simultaneously being shifted in; I2CDAT [7:0] always contains the last data byte present on the bus.

The acknowledge bit is controlled by the I<sup>2</sup>C hardware and cannot be accessed by the CPU. Serial data is shifted into I2CDAT [7:0] on the rising edges of serial clock pulses on the SCL line. When a byte has been shifted into I2CDAT [7:0], the serial data is available in I2CDAT [7:0], and the acknowledge bit (ACK or NACK) is returned by the control logic during the ninth clock pulse. In order to monitor bus status while sending data, the bus data will be shifted into I2CDAT[7:0] when sending I2CDAT[7:0] to bus. In case of sending data, serial data bits are shifted out from I2CDAT [7:0] on the falling edge of SCL clocks, and is shifted into I2CDAT [7:0] on the rising edge of SCL clocks.

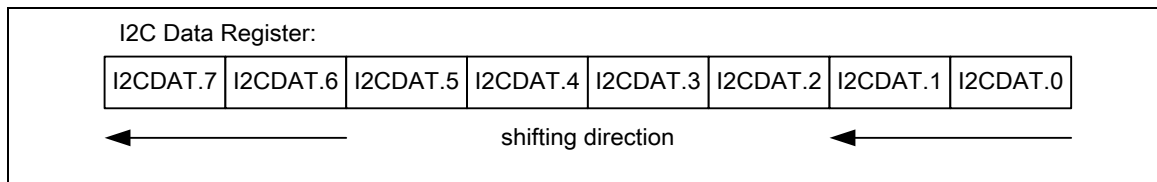


Figure 21-8 I<sup>2</sup>C Data Shifting Direction

### 21.4.3 Control Register (I2CON)

The CPU can read from and write to I2CON [7:0] directly. Two bits are affected by hardware: the SI bit is set when the I<sup>2</sup>C hardware requests a serial interrupt, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when ENS1 = 0.

- EI** Enable Interrupt.
- ENS1** Set to enable I<sup>2</sup>C serial function controller. When ENS1=1 the I<sup>2</sup>C serial function enables. The multi-function pin function must be set to I<sup>2</sup>C function.
- STA** I<sup>2</sup>C START Control Bit. Setting STA to logic 1 to enter Master mode, the I<sup>2</sup>C hardware sends a START or repeat START condition to bus when the bus is free.
- STO** I<sup>2</sup>C STOP Control Bit. In Master mode, setting STO to transmit a STOP condition to bus then I<sup>2</sup>C hardware will check the bus condition if a STOP condition is detected this flag will be cleared by hardware automatically. In Slave mode, setting STO resets I<sup>2</sup>C hardware to the defined "not addressed" Slave mode. This means it is NO LONGER in the slave receiver mode to receive data from the master.
- SI** I<sup>2</sup>C Interrupt Flag. When a new I<sup>2</sup>C state is present in the I2CSTATUS register, the SI flag is set by hardware, and if bit EI (I2CON [7]) is set, the I<sup>2</sup>C interrupt is requested. SI must be cleared by software. Clear SI is by writing 1 to this bit. All states are listed in section 5.6.6
- AA** Assert Acknowledge Control Bit. When AA=1 prior to address or data received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.) A slave is acknowledging the address sent from master, 2.) The receiver devices are acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line.

#### 21.4.4 Status Register (I2CSTATUS)

I2CSTATUS [7:0] is an 8-bit read-only register. The three least significant bits are always 0. The bit field I2CSTATUS [7:3] contain the status code and there are 26 possible status codes. All states are listed in section 5.6.6. When I2CSTATUS [7:0] is F8H, no serial interrupt is requested. All other I2CSTATUS [7:3] values correspond to defined I<sup>2</sup>C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2CSTATUS[7:3] one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software.

In addition, state 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the format frame. Examples of illegal positions are during the serial transfer of an address byte, a data byte or an acknowledge bit. To recover I<sup>2</sup>C from bus error, STO should be set and SI should be clear to enter Not Addressed Slave mode. Then clear STO to release bus and to wait new communication. I<sup>2</sup>C bus cannot recognize stop condition during this action when bus error occurs.

| Master mode |   | Slave mode |                                     |
|-------------|---|------------|-------------------------------------|
| STATUS      | Description   | STATUS     | Description                         |
| 0x08        | Start   | 0xA0       | Slave Transmit Repeat Start or Stop |
| 0x10        | Master Repeat Start   | 0xA8       | Slave Transmit Address ACK          |
| 0x18        | Master Transmit Address ACK   | 0xB0       | Slave Transmit Arbitration Lost     |
| 0x20        | Master Transmit Address NACK  | 0xB8       | Slave Transmit Data ACK             |
| 0x28        | Master Transmit Data ACK  | 0xC0       | Slave Transmit Data NACK            |
| 0x30        | Master Transmit Data NACK   | 0xC8       | Slave Transmit Last Data ACK        |
| 0x38        | Master Arbitration Lost   | 0x60       | Slave Receive Address ACK           |
| 0x40        | Master Receive ACK  | 0x68       | Slave Receive Arbitration Lost      |
| 0x48        | Master Receive NACK   | 0x80       | Slave Receive Data ACK              |
| 0x50        | Master Receive ACK  | 0x88       | Slave Receive Data NACK             |
| 0x58        | Master Receive NACK   | 0x70       | GC mode Address ACK                 |
| 0x00        | Bus error   | 0x78       | GC mode Arbitration Lost            |
|             |   | 0x90       | GC mode Data ACK                    |
|             |   | 0x98       | GC mode Data NACK                   |
| 0xF8        | Bus Released<br><b>Note:</b> Status "0xF8" exists in both Master/Slave modes, and it won't raise interrupt. |            |                                     |

Table 21-1 I<sup>2</sup>C Status Code Description Table

#### 21.4.5 I<sup>2</sup>C Clock Baud Rate Bits (I2CLK)

The data baud rate of I<sup>2</sup>C is determined by I2CLK [7:0] register when I<sup>2</sup>C is in Master mode. It is not important when I<sup>2</sup>C is in Slave mode. In the Slave mode, I<sup>2</sup>C will automatically synchronize with any clock frequency from master I<sup>2</sup>C device.

The data baud rate of I<sup>2</sup>C setting is Data Baud Rate of I<sup>2</sup>C = (system clock) / (4x (I2CLK [7:0] +1)). If system clock = 16 MHz, the I2CLK [7:0] = 40 (28H), so data baud rate of I<sup>2</sup>C = 16 MHz/ (4x (40 +1)) = 97.5 Kbits/sec.

#### 21.4.6 The I<sup>2</sup>C Time out Counter Register (I2CTOC)

There is a 14-bit time out counter which can be used to deal with the I<sup>2</sup>C bus hang-up. If the time out counter is enabled, the counter starts up counting until it overflows (TIF=1) and generates I<sup>2</sup>C interrupt to CPU or stops counting by clearing ENTI to 0. When time out counter is enabled, write 1 to SI flag will reset counter and re-start up counting after SI is cleared. If I<sup>2</sup>C bus hangs up, it causes the I2CSTATUS and flag SI are not updated for a period, the 14-bit time-out counter may overflow and acknowledge CPU the I<sup>2</sup>C interrupt. Refer to the Figure 21-9 for the 14-bit time out counter. User may write 1 to clear TIF to 0.

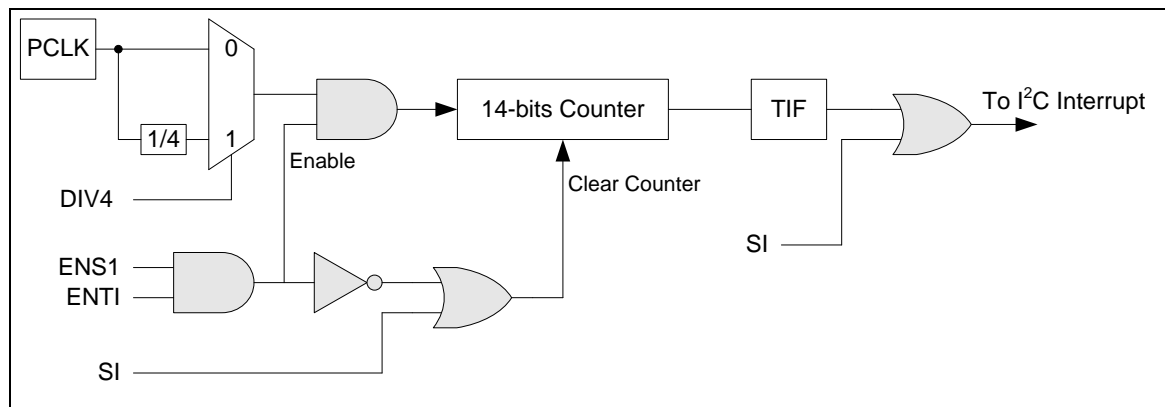


Figure 21-9 I<sup>2</sup>C Time out Count Block Diagram



### 21.5 Register Map

R: read only, W: write only, R/W: both read and write

| Register   | Offset       | R/W | Description                                   | Reset Value |
|--|--------------|-----|---|-------------|
| <b>I2C Base Address:</b><br>$I2Cx\_BA = 0x4002\_0000 + (0x10\_0000 * x)$<br>$x = 0, 1$ |              |     |   |             |
| <b>I2CON</b>   | I2Cx_BA+0x00 | R/W | I <sup>2</sup> C Control Register             | 0x0000_0000 |
| <b>I2CADDR0</b>  | I2Cx_BA+0x04 | R/W | I <sup>2</sup> C Slave Address Register0      | 0x0000_0000 |
| <b>I2CDAT</b>  | I2Cx_BA+0x08 | R/W | I <sup>2</sup> C Data Register                | 0x0000_0000 |
| <b>I2CSTATUS</b>   | I2Cx_BA+0x0C | R   | I <sup>2</sup> C Status Register              | 0x0000_00F8 |
| <b>I2CLK</b>   | I2Cx_BA+0x10 | R/W | I <sup>2</sup> C Clock Divided Register       | 0x0000_0000 |
| <b>I2CTOC</b>  | I2Cx_BA+0x14 | R/W | I <sup>2</sup> C Time Out Counter Register    | 0x0000_0000 |
| <b>I2CADDR1</b>  | I2Cx_BA+0x18 | R/W | I <sup>2</sup> C Slave Address Register1      | 0x0000_0000 |
| <b>I2CADDR2</b>  | I2Cx_BA+0x1C | R/W | I <sup>2</sup> C Slave Address Register2      | 0x0000_0000 |
| <b>I2CADDR3</b>  | I2Cx_BA+0x20 | R/W | I <sup>2</sup> C Slave Address Register3      | 0x0000_0000 |
| <b>I2CADM0</b>   | I2Cx_BA+0x24 | R/W | I <sup>2</sup> C Slave Address Mask Register0 | 0x0000_0000 |
| <b>I2CADM1</b>   | I2Cx_BA+0x28 | R/W | I <sup>2</sup> C Slave Address Mask Register1 | 0x0000_0000 |
| <b>I2CADM2</b>   | I2Cx_BA+0x2C | R/W | I <sup>2</sup> C Slave Address Mask Register2 | 0x0000_0000 |
| <b>I2CADM3</b>   | I2Cx_BA+0x30 | R/W | I <sup>2</sup> C Slave Address Mask Register3 | 0x0000_0000 |
| <b>I2CWKUPCON</b>  | I2Cx_BA+0x3C | R/W | I <sup>2</sup> C Wake Up Control Register     | 0x0000_0000 |
| <b>I2CWKUPSTS</b>  | I2Cx_BA+0x40 | R/W | I <sup>2</sup> C Wake Up Status Register      | 0x0000_0000 |

## 21.6 Register Description

### I<sup>2</sup>C Control Register (I2CON)

| Register | Offset       | R/W | Description                       | Reset Value |
|----------|--------------|-----|-----------------------------------|-------------|
| I2CON    | I2Cx_BA+0x00 | R/W | I <sup>2</sup> C Control Register | 0x0000_0000 |

|          |      |     |     |    |    |          |    |
|----------|------|-----|-----|----|----|----------|----|
| 31       | 30   | 29  | 28  | 27 | 26 | 25       | 24 |
| Reserved |      |     |     |    |    |          |    |
| 23       | 22   | 21  | 20  | 19 | 18 | 17       | 16 |
| Reserved |      |     |     |    |    |          |    |
| 15       | 14   | 13  | 12  | 11 | 10 | 9        | 8  |
| Reserved |      |     |     |    |    |          |    |
| 7        | 6    | 5   | 4   | 3  | 2  | 1        | 0  |
| EI       | ENS1 | STA | STO | SI | AA | Reserved |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | Reserved  |
| [7]    | EI          | <b>Enable Interrupt</b><br>1 = I <sup>2</sup> C interrupt Enabled.<br>0 = I <sup>2</sup> C interrupt Disabled.  |
| [6]    | ENS1        | <b>I<sup>2</sup>C Controller Enable Bit</b><br>1 = Enabled<br>0 = Disabled<br>Set to enable I <sup>2</sup> C serial function controller. When ENS1 =1 the I <sup>2</sup> C serial function enables. The multi-function pin function must set to I <sup>2</sup> C function first.  |
| [5]    | STA         | <b>I<sup>2</sup>C START Control Bit</b><br>Setting STA to logic 1 to enter Master mode, the I <sup>2</sup> C hardware sends a START or repeat START condition to bus when the bus is free.  |
| [4]    | STO         | <b>I<sup>2</sup>C STOP Control Bit</b><br>In Master mode, setting STO to transmit a STOP condition to bus then I <sup>2</sup> C hardware will check the bus condition if a STOP condition is detected this bit will be cleared by hardware automatically. In Slave mode, setting STO resets I <sup>2</sup> C hardware to the defined "not addressed" Slave mode. This means it is NO LONGER in the slave receiver mode to receive data from the master transmit device. |
| [3]    | SI          | <b>I<sup>2</sup>C Interrupt Flag</b><br>When a new I <sup>2</sup> C state is present in the I2CSTATUS register, the SI flag is set by hardware, and if bit EI (I2CON [7]) is set, the I <sup>2</sup> C interrupt is requested. SI must be cleared by software. Clear SI is by writing 1 to this bit.  |
| [2]    | AA          | <b>Assert Acknowledge Control Bit</b><br>When AA=1 prior to address or data received, an acknowledged (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when 1.) A slave is acknowledging the address sent from master, 2.) The receiver devices are   |

|       |          |  |
|-------|----------|--|
|       |          | acknowledging the data sent by transmitter. When AA=0 prior to address or data received, a Not acknowledged (high level to SDA) will be returned during the acknowledge clock pulse on the SCL line. |
| [1:0] | Reserved | Reserved   |

### I<sup>2</sup>C Data Register (I2CDAT)

| Register | Offset       | R/W | Description                    | Reset Value |
|----------|--------------|-----|--------------------------------|-------------|
| I2CDAT   | I2Cx_BA+0x08 | R/W | I <sup>2</sup> C Data Register | 0x0000_0000 |

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved    |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved    |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved    |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| I2CDAT[7:0] |    |    |    |    |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | Reserved  |
| [7:0]  | I2CDAT      | I <sup>2</sup> C Data Register<br>Bit [7:0] is located with the 8-bit transferred data of I <sup>2</sup> C serial port. |

### I<sup>2</sup>C Status Register (I2CSTATUS)

| Register  | Offset       | R/W | Description                      | Reset Value |
|-----------|--------------|-----|----------------------------------|-------------|
| I2CSTATUS | I2Cx_BA+0x0C | R   | I <sup>2</sup> C Status Register | 0x0000_00F8 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved       |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved       |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved       |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| I2CSTATUS[7:0] |    |    |    |    |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | Reserved  |
| [7:0]  | I2CSTATUS   | <p><b>I<sup>2</sup>C Status Register</b></p> <p>The status register of I<sup>2</sup>C:</p> <p>The three least significant bits are always 0. The five most significant bits contain the status code. There are 26 possible status codes. When I2CSTATUS contains F8H, no serial interrupt is requested. All other I2CSTATUS values correspond to defined I<sup>2</sup>C states. When each of these states is entered, a status interrupt is requested (SI = 1). A valid status code is present in I2CSTATUS one cycle after SI is set by hardware and is still present one cycle after SI has been reset by software. In addition, states 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position in the formation frame. Example of illegal position are during the serial transfer of an address byte, a data byte or an acknowledge bit.</p> |

### I<sup>2</sup>C Clock Divided Register (I2CLK)

| Register | Offset       | R/W | Description                             | Reset Value |
|----------|--------------|-----|---|-------------|
| I2CLK    | I2Cx_BA+0x10 | R/W | I <sup>2</sup> C Clock Divided Register | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved   |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved   |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved   |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| I2CLK[7:0] |    |    |    |    |    |    |    |

| Bits   | Description |  |
|--------|-------------|--|
| [31:8] | Reserved    | Reserved   |
| [7:0]  | I2CLK       | <b>I<sup>2</sup>C clock divided Register</b><br>The I <sup>2</sup> C clock rate bits: Data Baud Rate of I <sup>2</sup> C = (system clock) / (4x (I2CLK+1)).<br><b>Note:</b> The minimum value of I2CLK is 4. |

### I<sup>2</sup>C Time Out Counter Register (I2CTOC)

| Register | Offset       | R/W | Description                                | Reset Value |
|----------|--------------|-----|--|-------------|
| I2CTOC   | I2Cx_BA+0x14 | R/W | I <sup>2</sup> C Time Out Counter Register | 0x0000_0000 |

|          |    |    |    |    |      |      |     |
|----------|----|----|----|----|------|------|-----|
| 31       | 30 | 29 | 28 | 27 | 26   | 25   | 24  |
| Reserved |    |    |    |    |      |      |     |
| 23       | 22 | 21 | 20 | 19 | 18   | 17   | 16  |
| Reserved |    |    |    |    |      |      |     |
| 15       | 14 | 13 | 12 | 11 | 10   | 9    | 8   |
| Reserved |    |    |    |    |      |      |     |
| 7        | 6  | 5  | 4  | 3  | 2    | 1    | 0   |
| Reserved |    |    |    |    | ENTI | DIV4 | TIF |

| Bits   | Description |  |
|--------|-------------|--|
| [31:3] | Reserved    | Reserved   |
| [2]    | ENTI        | <b>Time out Counter Enable</b><br>1 = Time out counter Enabled<br>0 = Time out counter Disabled<br>When Enabled, the 14-bit time out counter will start counting when SI is clear. Write 1 to SI flag will reset counter and re-start up counting after SI is cleared. |
| [1]    | DIV4        | <b>Time out Counter Input Clock is Divided by 4</b><br>1 = The time out counter input clock divided by 4 Enabled<br>0 = The time out counter input clock divided by 4 Disabled<br>When Enabled, the time out period is extend 4 times.                                 |
| [0]    | TIF         | <b>Time out Flag</b><br>This bit is set by H/W when I <sup>2</sup> C time out happened and it can interrupt CPU if I <sup>2</sup> C interrupt enable bit (EI) is set to 1.<br>Software can write 1 to clear this bit.  |

**I<sup>2</sup>C Slave Address Register (I2CADDRx)**

| Register        | Offset       | R/W | Description                              | Reset Value |
|-----------------|--------------|-----|--|-------------|
| <b>I2CADDR0</b> | I2Cx_BA+0x04 | R/W | I <sup>2</sup> C Slave Address Register0 | 0x0000_0000 |
| <b>I2CADDR1</b> | I2Cx_BA+0x18 | R/W | I <sup>2</sup> C Slave Address Register1 | 0x0000_0000 |
| <b>I2CADDR2</b> | I2Cx_BA+0x1C | R/W | I <sup>2</sup> C Slave Address Register2 | 0x0000_0000 |
| <b>I2CADDR3</b> | I2Cx_BA+0x20 | R/W | I <sup>2</sup> C Slave Address Register3 | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved     |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Reserved     |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| I2CADDR[7:1] |    |    |    |    |    |    | GC |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | Reserved  |
| [7:1]  | I2CADDR     | <b>I<sup>2</sup>C Address Register</b><br>The content of this register is irrelevant when I <sup>2</sup> C is in Master mode. In Slave mode, the seven most significant bits must be loaded with the chip's own address. The I <sup>2</sup> C hardware will react if one of the addresses is matched. |
| [0]    | GC          | <b>General Call Function</b><br>0 = General Call function Disabled.<br>1 = General Call function Enabled.   |



### I<sup>2</sup>C Slave Address Mask Register (I2CADMx)

| Register | Offset       | R/W | Description                                   | Reset Value |
|----------|--------------|-----|---|-------------|
| I2CADM0  | I2Cx_BA+0x24 | R/W | I <sup>2</sup> C Slave Address Mask Register0 | 0x0000_0000 |
| I2CADM1  | I2Cx_BA+0x28 | R/W | I <sup>2</sup> C Slave Address Mask Register1 | 0x0000_0000 |
| I2CADM2  | I2Cx_BA+0x2C | R/W | I <sup>2</sup> C Slave Address Mask Register2 | 0x0000_0000 |
| I2CADM3  | I2Cx_BA+0x30 | R/W | I <sup>2</sup> C Slave Address Mask Register3 | 0x0000_0000 |

|             |    |    |    |    |    |    |          |
|-------------|----|----|----|----|----|----|----------|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved    |    |    |    |    |    |    |          |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved    |    |    |    |    |    |    |          |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved    |    |    |    |    |    |    |          |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| I2CADM[7:1] |    |    |    |    |    |    | Reserved |

| Bits   | Description |  |
|--------|-------------|--|
| [31:8] | Reserved    | Reserved   |
| [7:1]  | I2CADM      | <b>I<sup>2</sup>C Address Mask register</b><br>1 = Mask Enabled (the received corresponding address bit is don't care.)<br>0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.)<br>I <sup>2</sup> C bus controller supports multiple address recognition with four address mask registers. When the bit in the address mask register is set to one, it means the received corresponding address bit is don't-care. If the bit is set to 0, that means the received corresponding register bit should be exact the same as address register. |
| [0]    | Reserved    | Reserved   |

### I<sup>2</sup>C Wake Up Control Register (I2C WKUPCON)

| Register    | Offset       | R/W | Description                               | Reset Value |
|-------------|--------------|-----|---|-------------|
| I2C WKUPCON | I2Cx_BA+0x3C | R/W | I <sup>2</sup> C Wake Up Control Register | 0x0000_0000 |

|          |    |    |    |    |    |    |        |
|----------|----|----|----|----|----|----|--------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24     |
| Reserved |    |    |    |    |    |    |        |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16     |
| Reserved |    |    |    |    |    |    |        |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8      |
| Reserved |    |    |    |    |    |    |        |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0      |
| Reserved |    |    |    |    |    |    | WKUPEN |

| Bits   | Description |  |
|--------|-------------|--|
| [31:1] | Reserved    | Reserved   |
| [0]    | WKUPEN      | <b>I<sup>2</sup>C Wake-up Function Enable</b><br>1 = I <sup>2</sup> C wake up function Enabled<br>0 = I <sup>2</sup> C wake up function Disabled |

### I<sup>2</sup>C Wake Up Status Register (I2CWKUPSTS)

| Register   | Offset       | R/W | Description                              | Reset Value |
|------------|--------------|-----|--|-------------|
| I2CWKUPSTS | I2Cx_BA+0x40 | R/W | I <sup>2</sup> C Wake Up Status Register | 0x0000_0000 |

|          |    |    |    |    |    |    |        |
|----------|----|----|----|----|----|----|--------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24     |
| Reserved |    |    |    |    |    |    |        |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16     |
| Reserved |    |    |    |    |    |    |        |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8      |
| Reserved |    |    |    |    |    |    |        |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0      |
| Reserved |    |    |    |    |    |    | WKUPIF |

| Bits   | Description |   |
|--------|-------------|---|
| [31:1] | Reserved    | Reserved  |
| [0]    | WKUPIF      | I <sup>2</sup> C Wake Up Interrupt Flag<br>When chip is woken up from Power-down mode by I <sup>2</sup> C, this bit is set to 1. Software can write 1 to clear this bit |

## 21.7 Operation Modes

The on-chip I<sup>2</sup>C ports support five operation modes, Master Transmitter, Master Receiver, Slave Transmitter, Slave Receiver, and GC Call.

In a given application, I<sup>2</sup>C port may operate as a master or as a slave. In Slave mode, the I<sup>2</sup>C port hardware looks for its own slave address and the general call address. If one of these addresses is detected, and if the slave is willing to receive or transmit data from/to master (by setting the AA bit), acknowledge bit will be transmitted out on the 9th clock, hence an interrupt is requested on both master and slave devices if interrupt is enabled. When the microcontroller wishes to become the bus master, hardware waits until the bus is free before entering Master mode so that a possible slave action is not be interrupted. If bus arbitration is lost in Master mode, I<sup>2</sup>C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

Bits STA, STO and AA in I2CON register will determine the next state of the I<sup>2</sup>C hardware after SI flag is cleared. Upon completion of the new action, a new status code will be updated and the SI flag will be set. If the I<sup>2</sup>C interrupt control bit EI (I2CON [7]) is set, appropriate action or software branch of the new status code can be performed in the Interrupt service routine.

In the following figure about the five operation modes, detailed data flow is represented. The legend for those data flow figures is shown in Figure 21-10

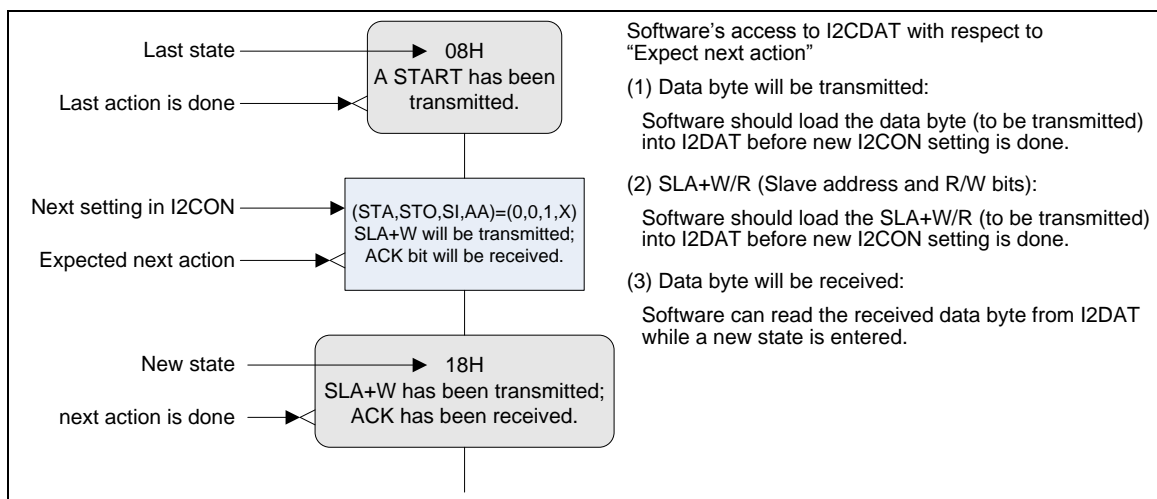


Figure 21-10 Legend for the Following Five Figures

### 21.7.1 Master Transmitter Mode

As shown in Figure 21-11, in master transmitter mode, serial data output through SDA while SCL outputs the serial clock. The first byte transmitted contains the slave address of the receiving device (7-bit) and the data direction bit. In this case the data direction bit (R/W) will be logic 0, and it is represented by "W" in the Figure 21-11. Thus the first byte transmitted is SLA+W. Serial data is transmitted 8-bit at a time. After each byte is transmitted, an acknowledge bit is received. START and STOP conditions are output to indicate the beginning and the end of a serial transfer.

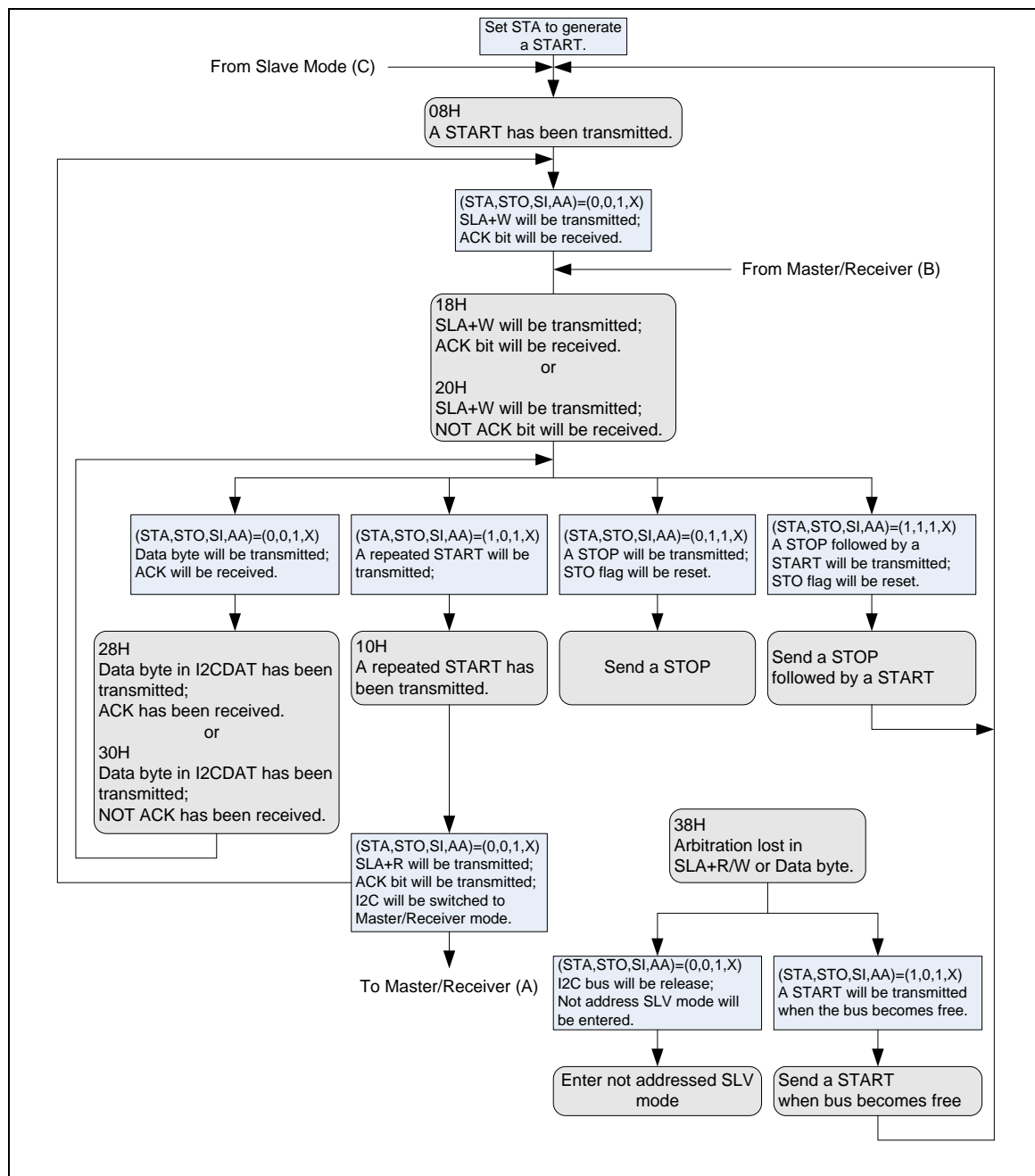


Figure 21-11 Master Transmitter Mode

### 21.7.2 Master Receiver Mode

As shown in Figure 21-12, in this case the data direction bit (R/W) will be logic 1, and it is represented by “R” in the Figure 21-12. Thus the first byte transmitted is SLA+R. Serial data is received via SDA while SCL outputs the serial clock. Serial data is received 8-bit at a time. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are output to indicate the beginning and end of a serial transfer.

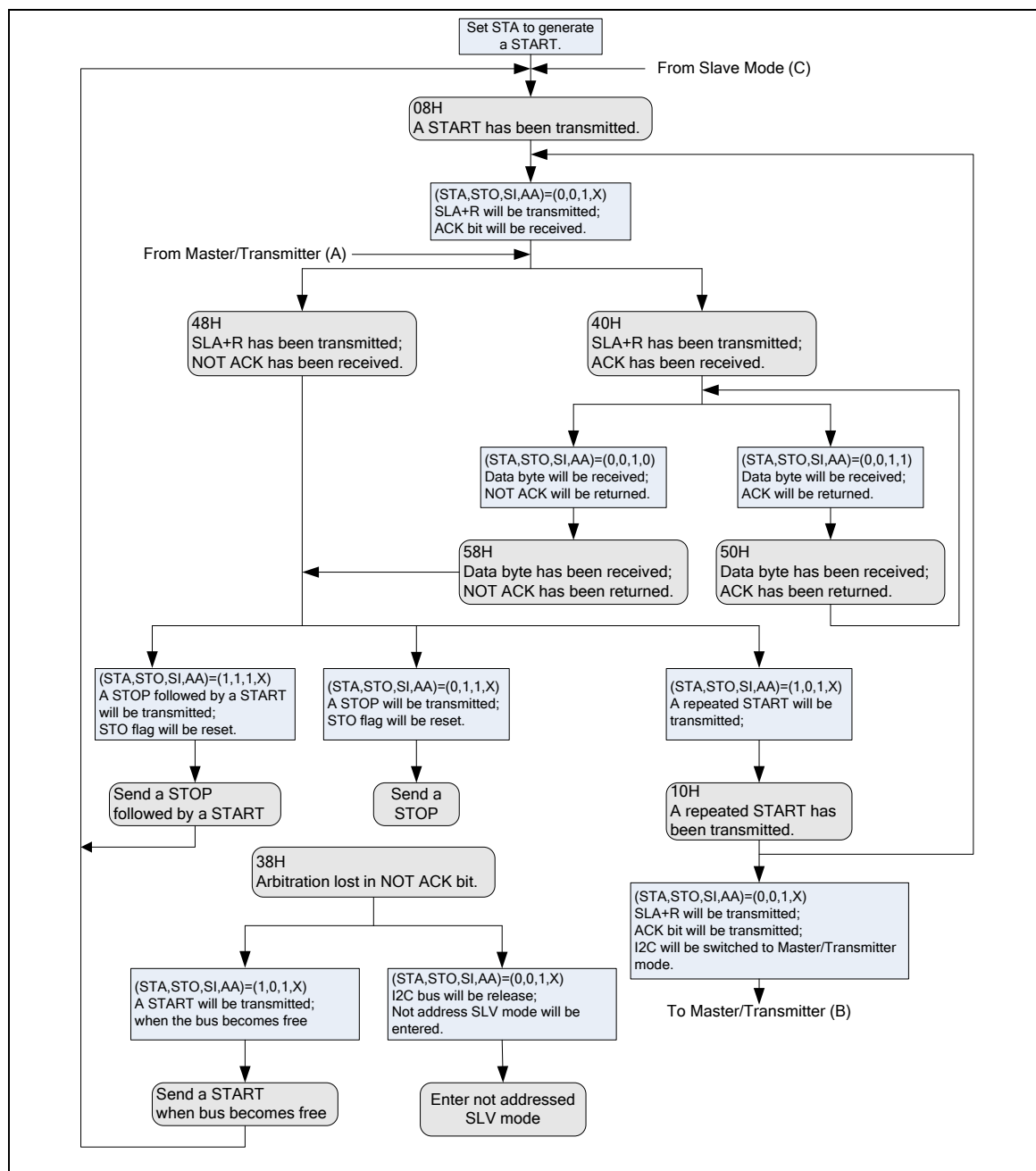


Figure 21-12 Master Receiver Mode

### 21.7.3 Slave Receiver Mode

As shown in Figure 21-13, serial data and the serial clock are received through SDA and SCL. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and direction bit.

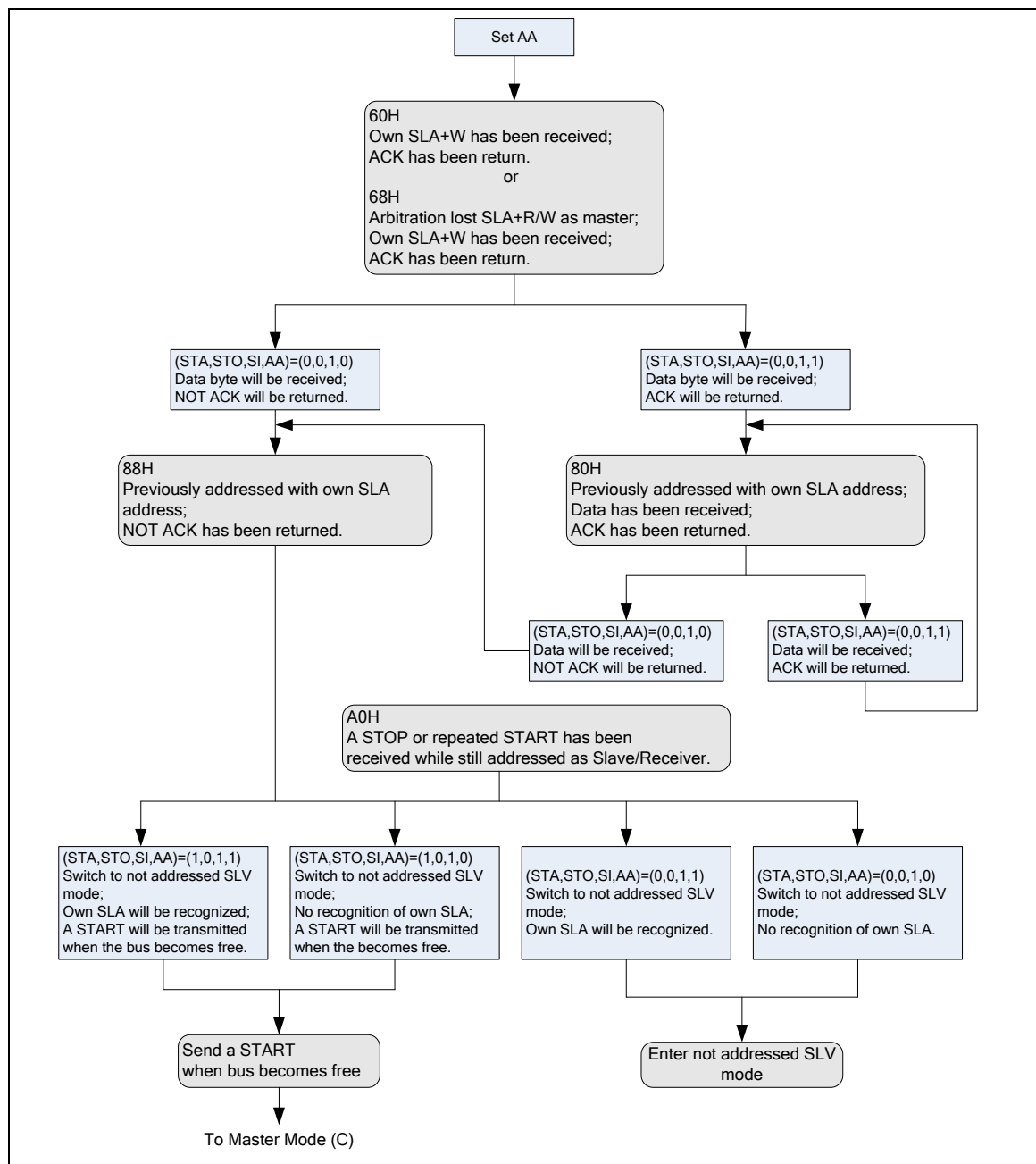


Figure 21-13 Slave Receiver Mode

#### 21.7.4 Slave Transmitter Mode

As shown in Figure 21-14, the first byte is received and handled as in the slave receiver mode. However, in this mode, the direction bit will indicate that the transfer direction is reversed. Serial data is transmitted via SDA while the serial clock is input through SCL. START and STOP conditions are recognized as the beginning and end of a serial transfer.

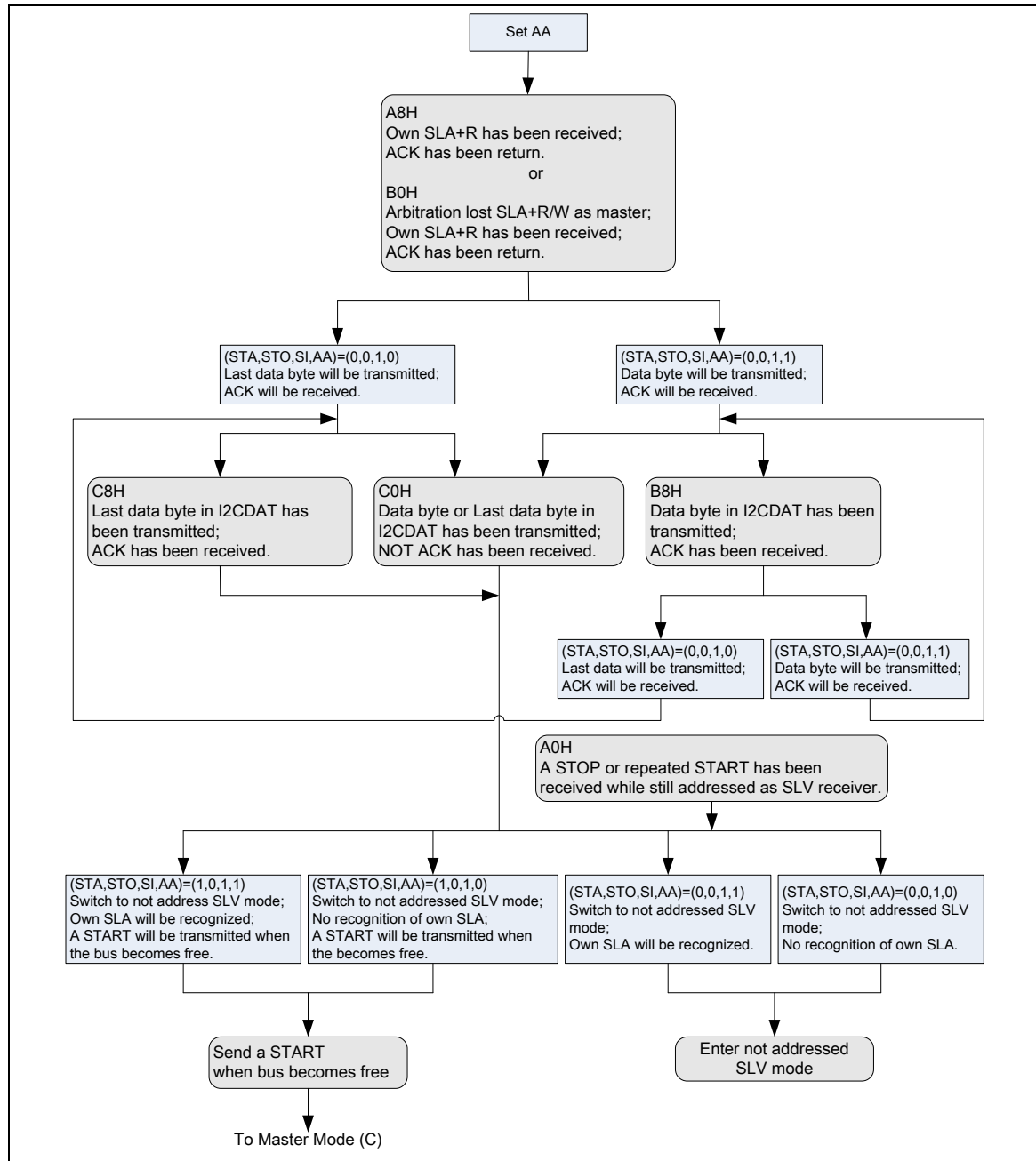


Figure 21-14 Slave Transmitter Mode

### 21.7.5 General Call (GC) Mode

As shown in Figure 21-15, if the GC bit (I2CADDRn [0]) is set, the I<sup>2</sup>C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function. When GC bit is set and the I<sup>2</sup>C is in Slave mode, it can receive the general call address by 00H after Master send general call address to I<sup>2</sup>C bus, then it will follow status of GC mode. Serial data and the serial clock are received through SDA and SCL. After each byte is received, an acknowledge bit is transmitted. START and STOP conditions are recognized as the beginning and end of a serial transfer. Address recognition is performed by hardware after reception of the slave address and direction bit.



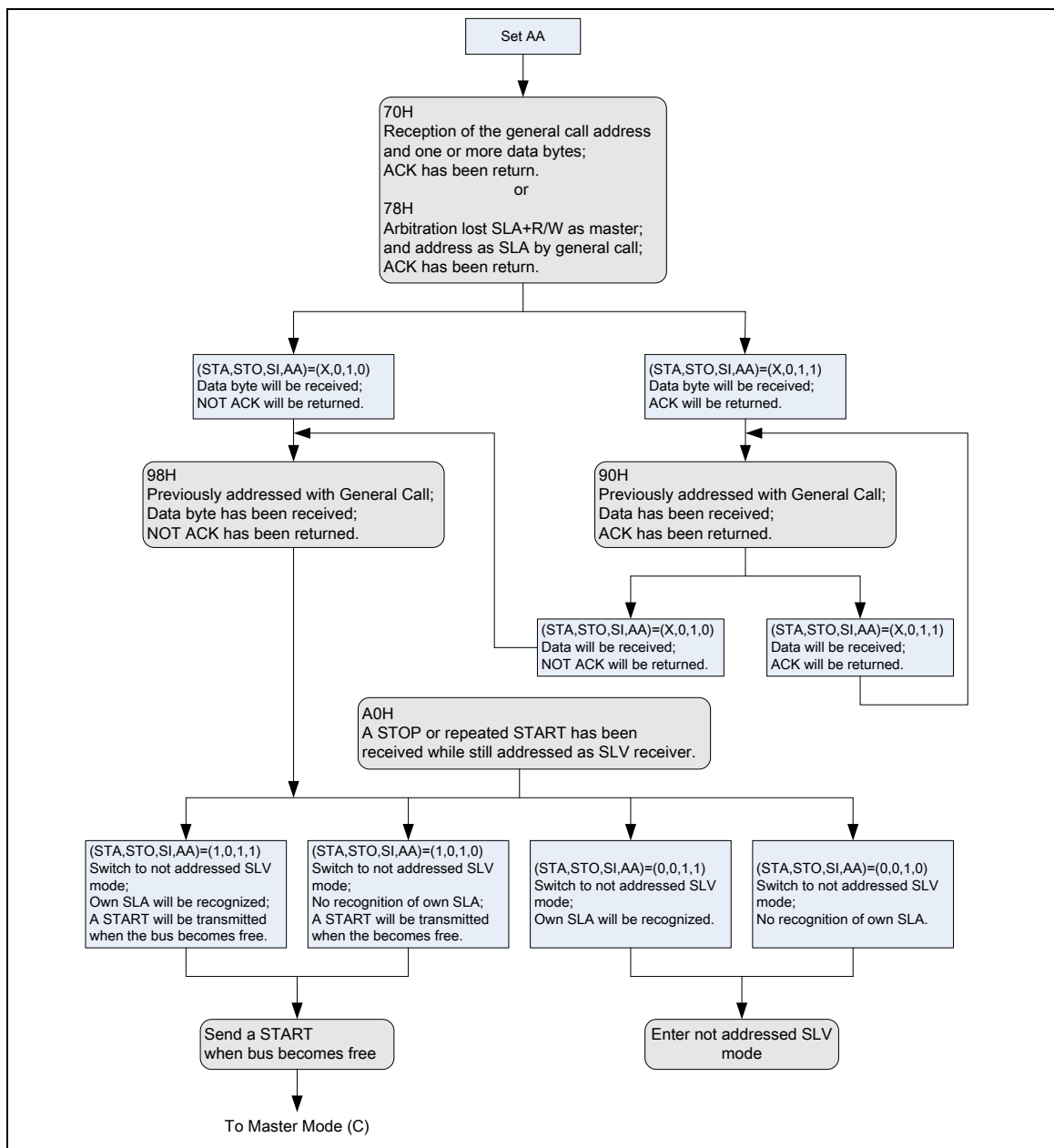


Figure 21-15 GC Mode

## 22 CONTROLLER AREA NETWORK (CAN)

### 22.1 Overview

The C\_CAN consists of the CAN Core, Message RAM, Message Handler, Control Registers and Module Interface (Refer **Figure 22-1 CAN Peripheral Block Diagram**)

). The CAN Core performs communication according to the CAN protocol version 2.0 part A and B. The bit rate can be programmed to values up to 1MBit/s. For the connection to the physical layer, additional transceiver hardware is required.

For communication on a CAN network, individual Message Objects are configured. The Message Objects and Identifier Masks for acceptance filtering of received messages are stored in the Message RAM. All functions concerning the handling of messages are implemented in the Message Handler. These functions include acceptance filtering, the transfer of messages between the CAN Core and the Message RAM, and the handling of transmission requests as well as the generation of the module interrupt.

The register set of the C\_CAN can be accessed directly by software through the module interface. These registers are used to control/configure the CAN Core and the Message Handler and to access the Message RAM.

### 22.2 Features

- Supports CAN protocol version 2.0 part A and B.
- Bit rates up to 1 MBit/s.
- 32 Message Objects.
- Each Message Object has its own identifier mask.
- Programmable FIFO mode (concatenation of Message Objects).
- Maskable interrupt.
- Disabled Automatic Re-transmission mode for Time Triggered CAN applications.
- Programmable loop-back mode for self-test operation.
- 16-bit module interfaces to the AMBA APB bus.
- Supports wake-up function.

## 22.3 Block Diagram

The C\_CAN interfaces with the AMBA APB bus. Figure 22-1 CAN Peripheral Block Diagram shows the block diagram of the C\_CAN.

### 1) CAN Core

CAN Protocol Controller and Rx/Tx Shift Register for serial/parallel conversion of messages.

### 2) Message RAM

Stores Message Objects and Identifier Masks

### 3) Registers

All registers used to control and to configure the C\_CAN.

### 4) Message Handler

State Machine that controls the data transfer between the Rx/Tx Shift Register of the CAN Core and the Message RAM as well as the generation of interrupts as programmed in the Control and Configuration Registers.

### 5) Module Interface

C\_CAN interfaces to the AMBA APB 16-bit bus from ARM.

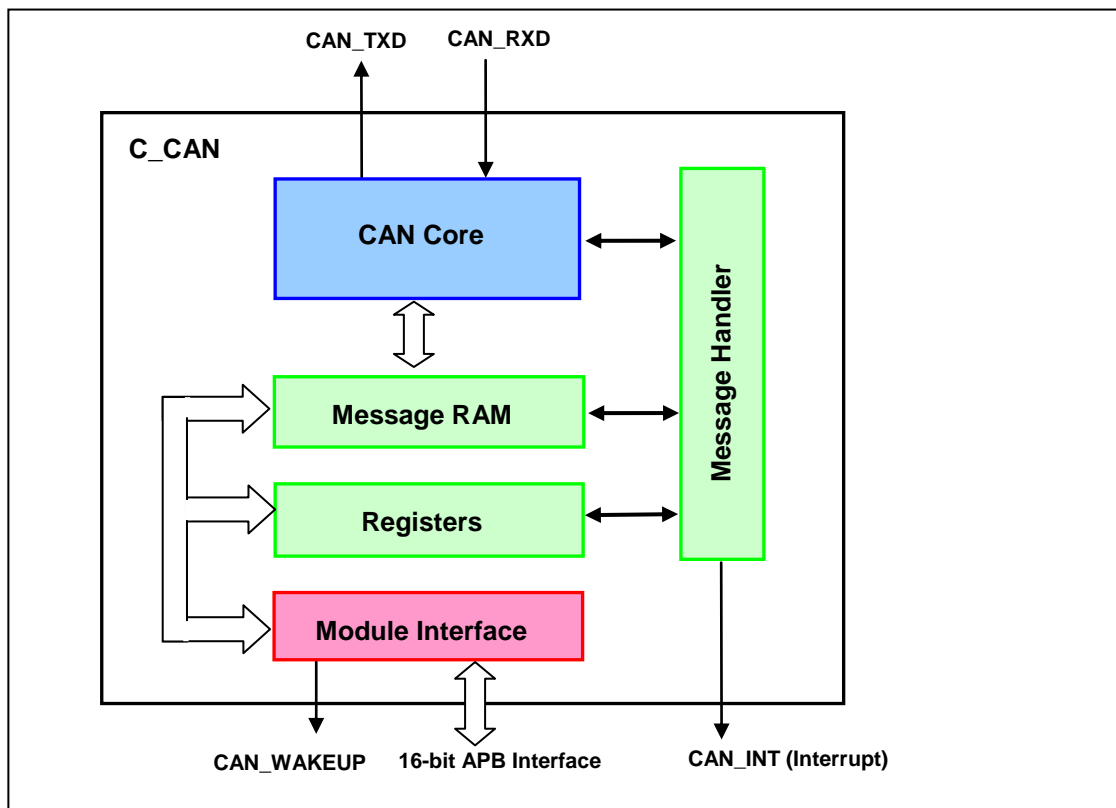


Figure 22-1 CAN Peripheral Block Diagram

## 22.4 Functional Description

### 22.4.1 Software Initialization

The software initialization is started by setting the **Init** bit in the CAN Control Register, either by a software or a hardware reset, or by going to *Bus\_Off* state.

While the **Init** bit is set, all messages transfer to and from the CAN bus are stopped and the status of the **CAN\_TXD** output pin is recessive (HIGH). The Error Management Logic (EML) counters are unchanged. Setting the **Init** bit does not change any configuration register.

To initialize the CAN Controller, software has to set up the Bit Timing Register and each Message Object. If a Message Object is not required, the corresponding **MsgVal** bit should be cleared. Otherwise, the entire Message Object has to be initialized.

Access to the Bit Timing Register and to the Baud Rate Prescaler Extension Register for configuring bit timing is enabled when both the **Init** and **CCE** bits in the CAN Control Register are set.

Resetting the **Init** bit (by software only) finishes software initialization. Later, the Bit Stream Processor (BSP) (see Section 5.13.6.10: Configuring the Bit Timing) synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (Bus Idle) before it can take part in bus activities and start the message transfer.

The initialization of the Message Objects is independent of **Init** and can be done on the fly, but the Message Objects should all be configured to particular identifiers or set to not valid before the BSP starts the message transfer.

To change the configuration of a Message Object during normal operation, software has to start by resetting the corresponding **MsgVal** bit. When the configuration is completed, **MsgVal** is set again.

### 22.4.2 CAN Message Transfer

Once the C\_CAN is initialized and **Init** bit is reset to 0, the C\_CAN Core synchronizes itself to the CAN bus and starts the message transfer.

Received messages are stored in their appropriate Message Objects if they pass the Message Handler's acceptance filtering. The whole message including all arbitration bits, DLC and eight data bytes are stored in the Message Object. If the Identifier Mask is used, the arbitration bits which are masked to "don't care" may be overwritten in the Message Object.

Software can read or write each message any time through the Interface Registers and the Message Handler guarantees data consistency in case of concurrent accesses.

Messages to be transmitted are updated by the application software. If a permanent Message Object (arbitration and control bits are set during configuration) exists for the message, only the data bytes are updated and the **TxRqst** bit with **NewDat** bit are set to start the transmission. If several transmit messages are assigned to the same Message Object (when the number of Message Objects is not sufficient), the whole Message Object has to be configured before the transmission of this message is requested.

The transmission of any number of Message Objects may be requested at the same time. Message objects are transmitted subsequently according to their internal priority. Messages may be updated or set to not valid any time, even when their requested transmission is still pending. The old data will be discarded when a message is updated before its pending transmission has started.

Depending on the configuration of the Message Object, the transmission of a message may be requested autonomously by the reception of a remote frame with a matching identifier.

### 22.4.3 Disabled Automatic Re-Transmission Mode

In accordance with the CAN Specification (see ISO11898, 6.3.3 Recovery Management), the C\_CAN provides means for automatic retransmission of frames that have lost arbitration or have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed. This means that, by default, automatic retransmission is enabled. It can be disabled to enable the C\_CAN to work within a Time Triggered CAN (TTCAN, see ISO11898-1) environment.

The Disabled Automatic Retransmission mode is enabled by setting the Disable Automatic Retransmission (**DAR**) bit in the CAN Control Register to one. In this operation mode, the programmer has to consider the different behavior of bits TxRqst and NewDat in the Control Registers of the Message Buffers:

- When a transmission starts, bit **TxRqst** of the respective Message Buffer is cleared, while bit **NewDat** remains set.
- When the transmission completed successfully, bit **NewDat** is cleared.
- When a transmission failed (lost arbitration or error), bit **NewDat** remains set.
- • To restart the transmission, software should set the bit **TxRqst** again.

## 22.5 Test Mode

Test Mode is entered by setting the Test bit in the CAN Control Register. In Test Mode, bits Tx1, Tx0, LBack, Silent and Basic in the Test Register are writeable. Bit Rx monitors the state of the CAN\_RX pin and therefore is only readable. All Test Register functions are disabled when the Test bit is cleared.

### 22.5.1 Silent Mode

The CAN Core can be set in Silent Mode by programming the **Silent** bit in the Test Register to one. In Silent Mode, the C\_CAN is able to receive valid data frames and valid remote frames, but it sends only recessive bits on the CAN bus and it cannot start a transmission. If the CAN Core is required to send a dominant bit (ACK bit, Error Frames), the bit is rerouted internally so that the CAN Core monitors this dominant bit, although the CAN bus may remain in recessive state. The Silent Mode can be used to analysis the traffic on a CAN bus without affecting it by the transmission of *dominant* bits. Figure 22-2 shows the connection of signals **CAN\_TXD** and **CAN\_RXD** to the CAN Core in Silent Mode.

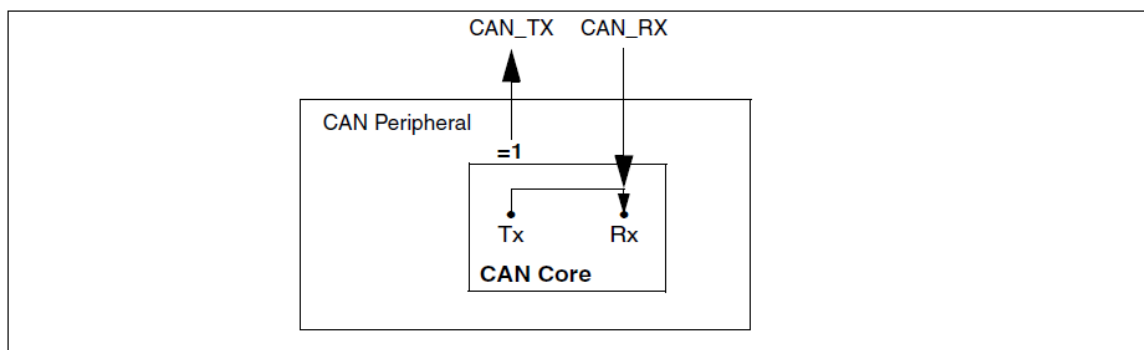


Figure 22-2 CAN Core in Silent Mode

### 22.5.2 Loop Back Mode

The CAN Core can be set in Loop Back Mode by programming the Test Register bit **LBack** to one. In Loop Back Mode, the CAN Core treats its own transmitted messages as received messages and stores them into a Receive Buffer (if they pass acceptance filtering). Figure 22-3 shows the connection of signals, **CAN\_TXD** and **CAN\_RXD**, to the CAN Core in Loop Back Mode.

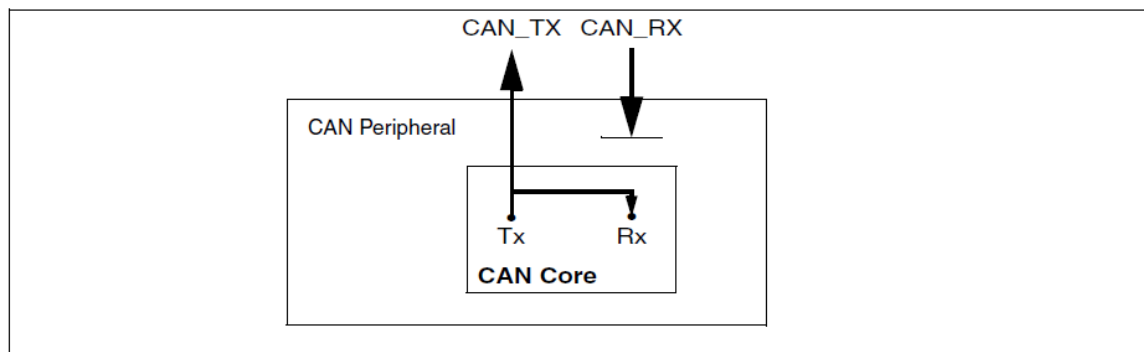


Figure 22-3 CAN Core in Loop Back Mode

This mode is provided for self-test functions. To be independent from external stimulation, the CAN Core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back Mode. In this mode, the CAN Core performs an internal feedback from its Tx output to its Rx input. The actual value of the **CAN\_RXD** input pin is disregarded by the CAN Core. The transmitted messages can be monitored on the **CAN\_TXD** pin.

### 22.5.3 Loop Back Combined with Silent Mode

It is also possible to combine Loop Back Mode and Silent Mode by programming bits **LBack** and **Silent** to one at the same time. This mode can be used for a “Hot Selftest”, which means that C\_CAN can be tested without affecting a running CAN system connected to the **CAN\_TXD** and **CAN\_RXD** pins. In this mode, the **CAN\_RXD** pin is disconnected from the CAN Core and the **CAN\_TXD** pin is held recessive. Figure 22-4 shows the connection of signals **CAN\_TXD** and **CAN\_RXD** to the CAN Core in case of the combination of Loop Back Mode with Silent Mode.

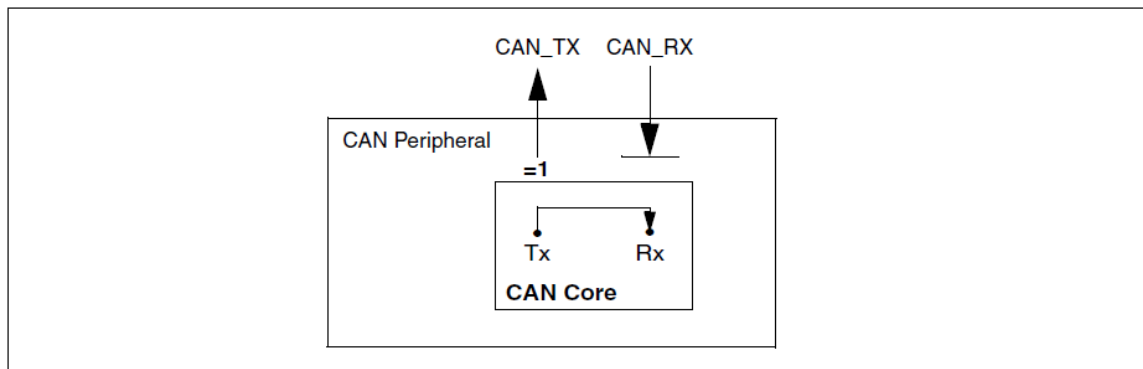


Figure 22-4 CAN Core in Loop Back Mode Combined with Silent Mode

### 22.5.4 Basic Mode

The CAN Core can be set in Basic Mode by programming the Test Register bit **Basic** to one. In this mode, the C\_CAN runs without the Message RAM.

The IF1 Registers are used as Transmit Buffer. The transmission of the contents of the IF1 Registers is requested by writing the **Busy** bit of the IF1 Command Request Register to one. The IF1 Registers are locked while the **Busy** bit is set. The Busy bit indicates that the transmission is pending.

As soon the CAN bus is idle, the IF1 Registers are loaded into the shift register of the CAN Core and the transmission is started. When the transmission has been completed, the **Busy** bit is reset and the locked IF1 Registers are released.

A pending transmission can be aborted at any time by resetting the **Busy** bit in the IF1 Command Request Register while the IF1 Registers are locked. If software has reset the **Busy** bit, a possible retransmission in case of lost arbitration or in case of an error is disabled.

The IF2 Registers are used as a Receive Buffer. After the reception of a message the contents of the shift register is stored into the IF2 Registers, without any acceptance filtering.

Additionally, the actual contents of the shift register can be monitored during the message transfer. Each time a read Message Object is initiated by writing the **Busy** bit of the IF2 Command Request Register to one, the contents of the shift register are stored into the IF2 Registers.

In Basic Mode, the evaluation of all Message Object related control and status bits and the control bits of the IFn Command Mask Registers are turned off. The message number of the Command request registers is not evaluated. The **NewDat** and **MsgLst** bits in the IF2 Message Control Register retain their function, **DLC3-0** indicates the received DLC, and the other control bits are read as '0'.

### 22.5.5 Software Control of CAN\_TX Pin

Four output functions are available for the CAN transmit pin, **CAN\_TXD**. In addition to its default



function (serial data output), the CAN transmit pin can drive the CAN Sample Point signal to monitor CAN\_Core's bit timing and it can drive constant dominant or recessive values. The latter two functions, combined with the readable CAN receive pin **CAN\_RXD**, can be used to check the physical layer of the CAN bus.

The output mode for the **CAN\_TXD** pin is selected by programming the **Tx1** and **Tx0** bits of the CAN Test Register.

The three test functions of the **CAN\_TXD** pin interfere with all CAN protocol functions. **CAN\_TXD** must be left in its default function when CAN message transfer or any of the test modes (Loop Back Mode, Silent Mode, or Basic Mode) are selected.

## 22.5.6 CAN Communications

### 22.5.7 Managing Message Objects

The configuration of the Message Objects in the Message RAM (with the exception of the bits the CAN Control Register **MsgVal**, **NewDat**, **IntPnd**, and **TxRqst**) will not be affected by resetting the chip. All the Message Objects must be initialized by the application software or they must be "not valid" (**MsgVal** = '0') and the bit timing must be configured before the application software clears the Init bit in ter.

The configuration of a Message Object is done by programming Mask, Arbitration, Control and Data fields of one of the two interface registers to the desired values. By writing to the corresponding IFn Command Request Register, the IFn Message Buffer Registers are loaded into the addressed Message Object in the Message RAM.

When the Init bit in the CAN Control Register is cleared, the CAN Protocol Controller state machine of the CAN\_Core and the state machine of the Message Handler control the internal data flow of the C\_CAN. Received messages that pass the acceptance filtering are stored into the Message RAM, messages with pending transmission request are loaded into the CAN\_Core's Shift Register and are transmitted through the CAN bus.

The application software reads received messages and updates messages to be transmitted through the IFn Interface Registers. Depending on the configuration, the application software is interrupted on certain CAN message and CAN error events.

### 22.5.8 Message Handler State Machine

The Message Handler controls the data transfer between the Rx/Tx Shift Register of the CAN Core, the Message RAM and the IFn Registers.

The Message Handler FSM controls the following functions:

- Data Transfer from IFn Registers to the Message RAM
- Data Transfer from Message RAM to the IFn Registers
- Data Transfer from Shift Register to the Message RAM
- Data Transfer from Message RAM to Shift Register
- Data Transfer from Shift Register to the Acceptance Filtering unit
- Scanning of Message RAM for a matching Message Object
- Handling of TxRqst flags
- Handling of interrupts.



### 22.5.9 Data Transfer from/to Message RAM

When the application software initiates a data transfer between the IFn Registers and Message RAM, the Message Handler sets the Busy bit in the respective Command Request Register (CAN\_IFn\_CRR) to '1'. After the transfer has completed, the Busy bit is again cleared (see Figure 22-5).

The respective Command Mask Register specifies whether a complete Message Object or only parts of it will be transferred. Due to the structure of the Message RAM, it is not possible to write single bits/bytes of one Message Object. It is always necessary to write a complete Message Object into the Message RAM. Therefore, the data transfer from the IFn Registers to the Message RAM requires a read-modify-write cycle. First, those parts of the Message Object that are not to be changed are read from the Message RAM and then the complete contents of the Message Buffer Registers are written into the Message Object.

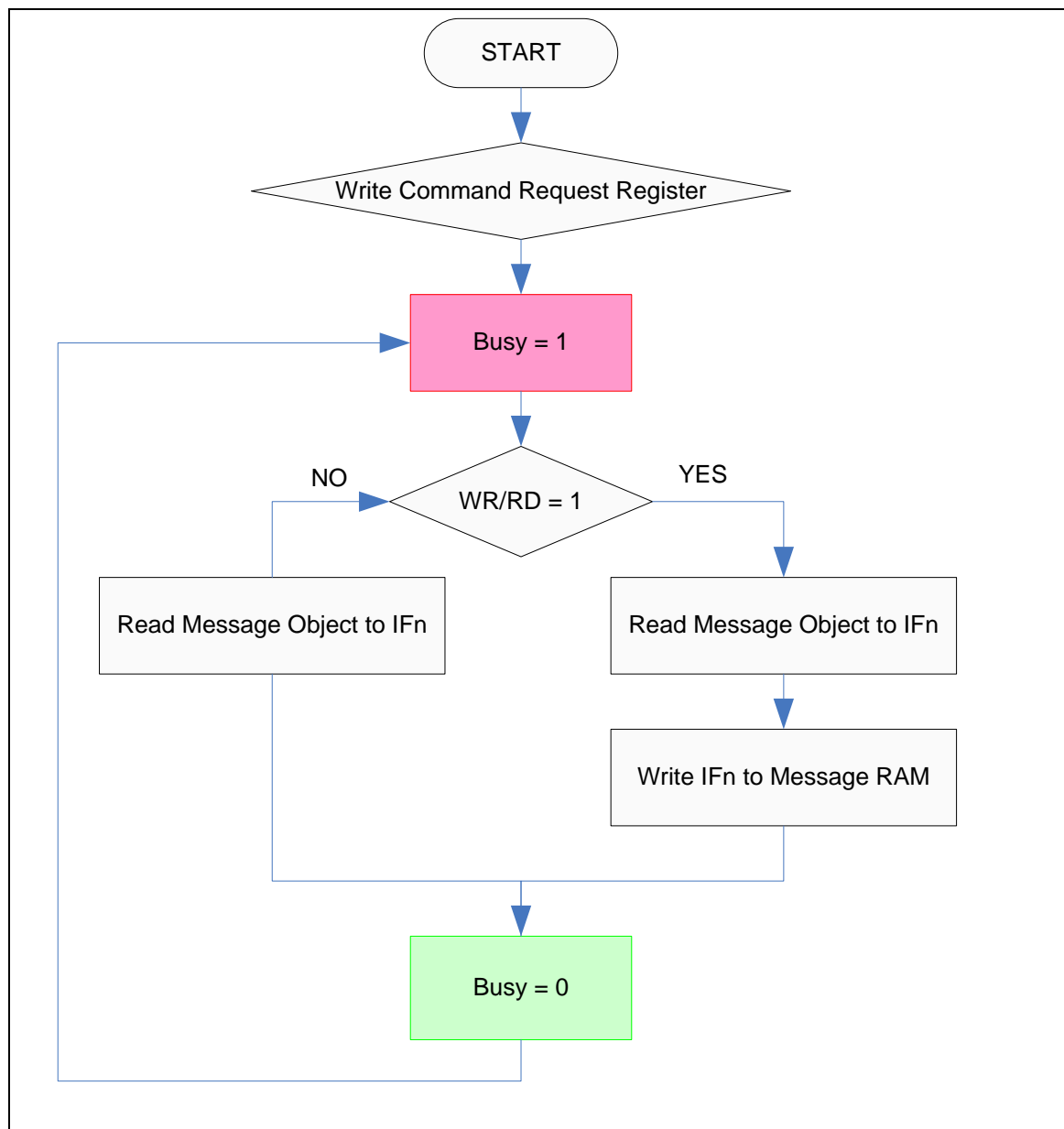


Figure 22-5 Data Transfer between IFn Registers and Message

After a partial write of a Message Object, the Message Buffer Registers that are not selected in the Command Mask Register will set the actual contents of the selected Message Object.

After a partial read of a Message Object, the Message Buffer Registers that are not selected in the Command Mask Register will be left unchanged.

#### 22.5.10 Message Transmission

If the shift register of the CAN Core cell is ready for loading and if there is no data transfer between the IFn Registers and Message RAM, the **MsgVal** bits in the Message Valid Register and **TxRqst** bits in the Transmission Request Register are evaluated. The valid Message Object with the highest priority pending transmission request is loaded into the shift register by the Message Handler and the transmission is started. The **NewDat** bit of the Message Object is reset.

After a successful transmission and also if no new data was written to the Message Object (**NewDat** = '0') since the start of the transmission, the **TxRqst** bit of the Message Control register (CAN\_IFn\_MCR) will be reset. If **TxE** bit of the Message Control register (CAN\_IFn\_MCR) is set, **IntPnd** bit of the Interrupt Identifier register will be set after a successful transmission. If the C\_CAN has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. Meanwhile, if the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

### 22.5.11 Acceptance Filtering of Received Messages

When the arbitration and control field (Identifier + IDE + RTR + DLC) of an incoming message is completely shifted into the Rx/Tx Shift Register of the CAN Core, the Message Handler FSM starts the scanning of the Message RAM for a matching valid Message Object.

To scan the Message RAM for a matching Message Object, the Acceptance Filtering unit is loaded with the arbitration bits from the CAN Core shift register. The arbitration and mask fields (including **MsgVal**, **UMask**, **NewDat**, and **EoB**) of Message Object 1 are then loaded into the Acceptance Filtering unit and compared with the arbitration field from the shift register. This is repeated with each following Message Object until a matching Message Object is found or until the end of the Message RAM is reached.

If a match occurs, the scan is stopped and the Message Handler FSM proceeds depending on the type of frame (Data Frame or Remote Frame) received.

#### Reception of Data Frame

The Message Handler FSM stores the message from the CAN Core shift register into the respective Message Object in the Message RAM. Not only the data bytes, but all arbitration bits and the Data Length Code are stored into the corresponding Message Object. This is done to keep the data bytes connected with the identifier even if arbitration mask registers are used.

The **NewDat** bit is set to indicate that new data (not yet seen by software) has been received. The application software should reset **NewDat** bit when the Message Object has been read. If at the time of reception, the **NewDat** bit was already set, **MsgLst** is set to indicate that the previous data (supposedly not seen by software) is lost. If the **RxE** bit is set, the **IntPnd** bit is set, causing the Interrupt Register to point to this Message Object.

The **TxRqst** bit of this Message Object is reset to prevent the transmission of a Remote Frame, while the requested Data Frame has just been received.

#### Reception of Remote Frame

When a Remote Frame is received, three different configurations of the matching Message Object have to be considered:

- 1) Dir = '1' (direction = transmit), **RmtEn** = '1', **UMask** = '1' or '0'

At the reception of a matching Remote Frame, the **TxRqst** bit of this Message Object is set. The rest of the Message Object remains unchanged.

- 2) Dir = '1' (direction = transmit), **RmtEn** = '0', **UMask** = '0'

At the reception of a matching Remote Frame, the **TxRqst** bit of this Message Object remains unchanged; the Remote Frame is ignored.

- 3) Dir = '1' (direction = transmit), **RmtEn** = '0', **UMask** = '1'

At the reception of a matching Remote Frame, the **TxRqst** bit of this Message Object is reset. The arbitration and control field (Identifier + IDE + RTR + DLC) from the shift register is stored

in the Message Object of the Message RAM and the **NewDat** bit of this Message Object is set. The data field of the Message Object remains unchanged; the Remote Frame is treated similar to a received Data Frame.

### 22.5.12 Receive/Transmit Priority

The receive/transmit priority for the Message Objects is attached to the message number. Message Object 1 has the highest priority, while Message Object 32 has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding Message Object

### 22.5.13 Configuring a Transmit Object

The following table shows how a Transmit Object should be initialized.

| Ms | Arb   | Data  | Mask  | EOB | Dir | NewDat | MsgLst | RxIE | TxIE  | IntPnd | RmtEn | TxRqst |
|----|-------|-------|-------|-----|-----|--------|--------|------|-------|--------|-------|--------|
| 1  | appl. | appl. | appl. | 1   | 1   | 0      | 0      | 0    | appl. | 0      | appl. | 0      |

Table 22-1 Initialization of a Transmit Object

**Note:** appl. = application software.

The Arbitration Register values (**ID28-0** and **Xtd** bit) are provided by the application. They define the identifier and type of the outgoing message. If an 11-bit Identifier ("Standard Frame") is used, it is programmed to ID28 - ID18. The ID17 - ID0 can then be disregarded.

If the **TxIE** bit is set, the **IntPnd** bit will be set after a successful transmission of the Message Object.

If the **RmtEn** bit is set, a matching received Remote Frame will cause the **TxRqst** bit to be set; the Remote Frame will autonomously be answered by a Data Frame.

The Data Register values (**DLC3-0**, **Data0-7**) are provided by the application, **TxRqst** and **RmtEn** may not be set before the data is valid.

The Mask Registers (**Msk28-0**, **UMask**, **MXtd**, and **MDir** bits) may be used (**UMask**='1') to allow groups of Remote Frames with similar identifiers to set the **TxRqst** bit. The Dir bit should not be masked.

### 22.5.14 Updating a Transmit Object

The software may update the data bytes of a Transmit Object any time through the IFn Interface registers, neither **MsgVal** nor **TxRqst** have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes of the corresponding IFn Data A Register or IFn Data B Register have to be valid before the contents of that register are transferred to the Message Object. Either the application software has to write all four bytes into the IFn Data Register or the Message Object is transferred to the IFn Data Register before software writes the new data bytes.

When only the (eight) data bytes are updated, first 0x0087 is written to the Command Mask Register and then the number of the Message Object is written to the Command Request Register, concurrently updating the data bytes and setting **TxRqst**.

To prevent the reset of **TxRqst** at the end of a transmission that may already be in progress while

the data is updated, **NewDat** has to be set together with **TxRqst**.

When **NewDat** is set together with **TxRqst**, **NewDat** will be reset as soon as the new transmission has started.

### 22.5.15 Configuring a Receive Object

The following table shows how a Receive Object should be initialized.

| MsgVal | Arb   | Data  | Mask  | EOB | Dir | NewDat | MsgLst | RxIE  | TxIE | IntPnd | RmtEn | TxRqst |
|--------|-------|-------|-------|-----|-----|--------|--------|-------|------|--------|-------|--------|
| 1      | appl. | appl. | appl. | 1   | 0   | 0      | 0      | appl. | 0    | 0      | 0     | 0      |

Table 22-2 Initialization of a Receive Object

The Arbitration Registers values (**ID28-0** and **Xtd** bit) are provided by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier ("Standard Frame") is used, it is programmed to ID28 - ID18. Then ID17 - ID0 can be disregarded. When a Data Frame with an 11-bit Identifier is received, ID17 - ID0 will be set to '0'.

If the **RxIE** bit is set, the **IntPnd** bit will be set when a received Data Frame is accepted and stored in the Message Object.

The Data Length Code (DLC3-0) is provided by the application. When the Message Handler stores a Data Frame in the Message Object, it will store the received Data Length Code and eight data bytes. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by unspecified values.

The Mask Registers (Msk28-0, **UMask**, **MXtd**, and **MDir** bits) may be used (**UMask**='1') to allow groups of Data Frames with similar identifiers to be accepted. The Dir bit should not be masked in typical applications.

### 22.5.16 Handling Received Messages

The application software may read a received message any time through the IFn Interface registers. The data consistency is guaranteed by the Message Handler state machine.

Typically, software will write first 0x007F to the Command Mask Register and then the number of the Message Object to the Command Request Register. This combination will transfer the whole received message from the Message RAM into the Message Buffer Register. Additionally, the bits **NewDat** and **IntPnd** are cleared in the Message RAM (not in the Message Buffer).

If the Message Object uses masks for acceptance filtering, the arbitration bits shows which of the matching messages have been received.

The actual value of **NewDat** shows whether a new message has been received since the last time this Message Object was read. The actual value of **MsgLst** shows whether more than one message has been received since the last time this Message Object was read. **MsgLst** will not be automatically reset.

By means of a Remote Frame, software may request another CAN node to provide new data for a receive object. Setting the **TxRqst** bit of a receive object will cause the transmission of a Remote Frame with the receive object's identifier. This Remote Frame triggers the other CAN node to start the transmission of the matching Data Frame. If the matching Data Frame is received before the Remote Frame could be transmitted, the **TxRqst** bit is automatically reset.

**22.5.17 Configuring a FIFO Buffer**

With the exception of the EoB bit, the configuration of Receive Objects belonging to a FIFO Buffer is the same as the configuration of a (single) Receive Object, see Section 5.13.6.5: Configuring a Receive Object.

To concatenate two or more Message Objects into a FIFO Buffer, the identifiers and masks (if used) of these Message Objects have to be programmed to matching values. Due to the implicit priority of the Message Objects, the Message Object with the lowest number will be the first Message Object of the FIFO Buffer. The **EoB** bit of all Message Objects of a FIFO Buffer except the last have to be programmed to 0. The **EoB** bits of the last Message Object of a FIFO Buffer is set to one, configuring it as the End of the Block.

**22.5.18 Receiving Messages with FIFO Buffers**

Received messages with identifiers matching to a FIFO Buffer are stored into a Message Object of this FIFO Buffer starting with the Message Object with the lowest message number.

When a message is stored into a Message Object of a FIFO Buffer, the **NewDat** bit of this Message Object is set. By setting **NewDat** while **EoB** is 0, the Message Object is locked for further write access by the Message Handler until the application software has written the **NewDat** bit back to 0.

Messages are stored into a FIFO Buffer until the last Message Object of this FIFO Buffer is reached. If none of the preceding Message Objects is released by writing **NewDat** to 0, all further messages for this FIFO Buffer will be written into the last Message Object of the FIFO Buffer and therefore overwrite previous messages.

**22.5.19 Reading from a FIFO Buffer**

When the application software transfers the contents of a Message Object to the IFn Message Buffer register by writing its number to the IFn Command Request Register, the corresponding Command Mask Register should be programmed in such a way that bits **NewDat** and **IntPnd** are reset to 0 (**TxRqst/NewDat** = '1' and **ClrIntPnd** = '1'). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

To assure the correct function of a FIFO Buffer, the application software should read the Message Objects starting at the FIFO Object with the lowest message number.

Figure 22-6 shows how a set of Message Objects which are concatenated to a FIFO Buffer can be handled by the application software.

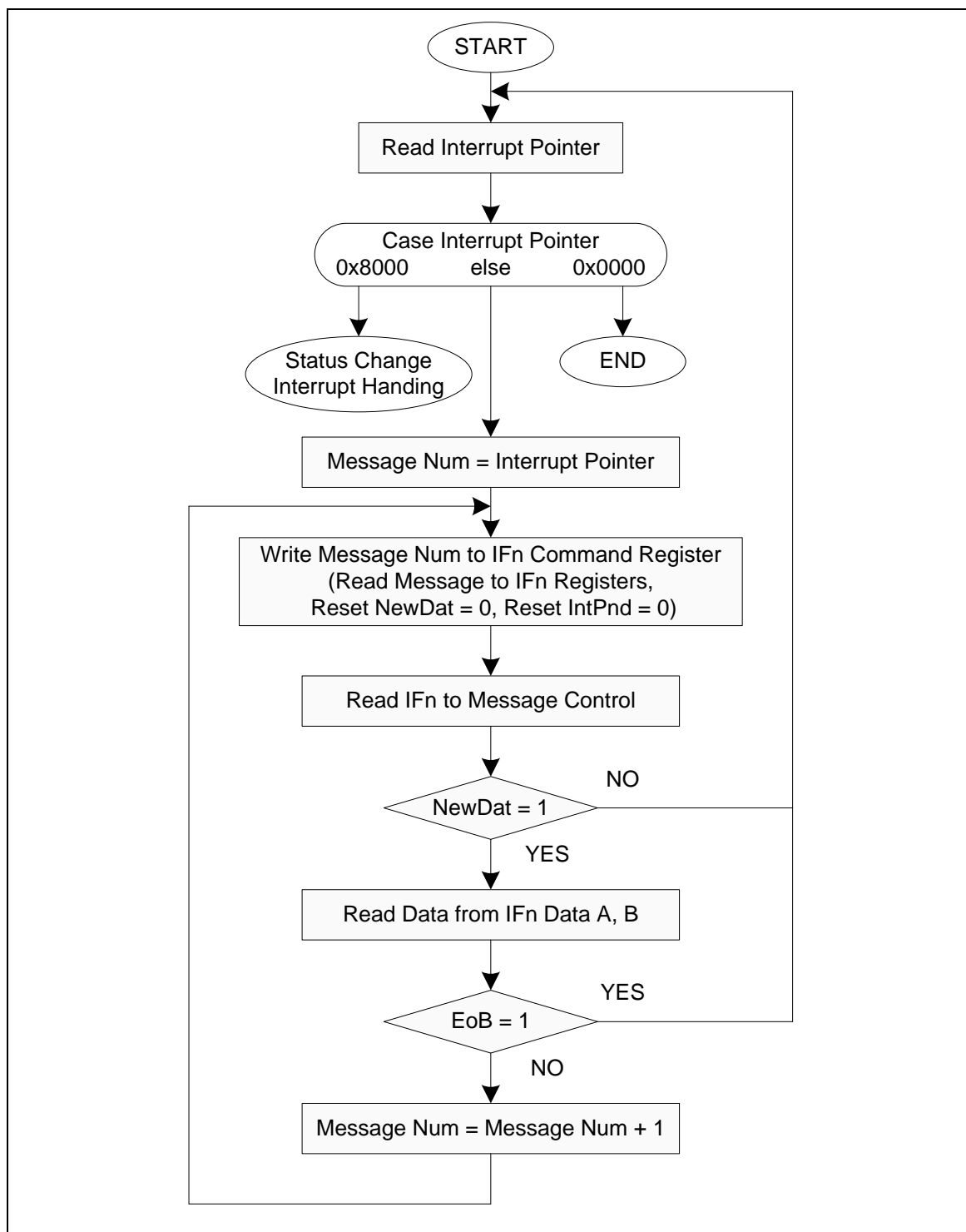


Figure 22-6 Application Software Handling of a FIFO Buffer



### 22.5.20 Handling Interrupts

If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the application software has cleared it.

The Status Interrupt has the highest priority. Among the message interrupts, interrupt priority of the Message Object decreases with increasing message number.

A message interrupt is cleared by clearing the **IntPnd** bit of the Message Object. The Status Interrupt is cleared by reading the Status Register.

The interrupt identifier, **IntId**, in the Interrupt Register, indicates the cause of the interrupt. When no interrupt is pending, the register will hold the value 0. If the value of the Interrupt Register is different from 0, then there is an interrupt pending and, if IE is set, the CAN\_INT interrupt signal is active. The interrupt remains active until the Interrupt Register is back to value 0 (the cause of the interrupt is reset) or until IE is reset.

The value 0x8000 indicates that an interrupt is pending because the CAN Core has updated (not necessarily changed) the Status Register (Error Interrupt or Status Interrupt). This interrupt has the highest priority. The application software can update (reset) the status bits **RxOk**, **TxOk** and **LEC**, but a write access of software to the Status Register can never generate or reset an interrupt.

All other values indicate that the source of the interrupt is one of the Message Objects. **IntId** points to the pending message interrupt with the highest interrupt priority.

The application software controls whether a change of the Status Register may cause an interrupt (bits EIE and SIE in the CAN Control Register) and whether the interrupt line becomes active when the Interrupt Register is different from 0 (bit IE in the CAN Control Register). The Interrupt Register will be updated even when IE is reset.

The application software has two possibilities to follow the source of a message interrupt. First, it can follow the **IntId** in the Interrupt Register and second it can poll the Interrupt Pending Register.

An interrupt service routine that is reading the message that is the source of the interrupt may read the message and reset the Message Object's **IntPnd** at the same time (bit **ClrIntPnd** in the Command Mask Register). When **IntPnd** is cleared, the Interrupt Register will point to the next Message Object with a pending interrupt.

### 22.5.21 Configuring the Bit Timing

Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of a CAN network can be reduced significantly.

In many cases, the CAN bit synchronization will amend a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. However, in the case of arbitration, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point may cause one of the transmitters to become error passive.

The analysis of such sporadic errors requires a detailed knowledge of the CAN bit synchronization inside a CAN node and interaction of the CAN nodes on the CAN bus.

### 22.5.22 Bit Time and Bit Rate

CAN supports bit rates in the range of lower than 1 kBit/s up to 1000 kBit/s. Each member of the CAN network has its own clock generator, usually a quartz oscillator. The timing parameter of the bit time (i.e. the reciprocal of the bit rate) can be configured individually for each CAN



node, creating a common bit rate even though the oscillator periods of the CAN nodes ( $f_{osc}$ ) may be different.

The frequencies of these oscillators are not absolutely stable, small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by re-synchronizing to the bit stream.

According to the CAN specification, the bit time is divided into four segments (see Figure 22-7). The Synchronization Segment, the Propagation Time Segment, the Phase Buffer Segment 1 and the Phase Buffer Segment 2. Each segment consists of a specific, programmable number of time quanta. The length of the time quantum ( $t_q$ ), which is the basic time unit of the bit time, is defined by the CAN controller's APB clock  $f_{APB}$  and the BRP bit of the Bit Timing Register (CAN\_BTR):  
$$t_q = BRP / f_{APB}$$

The Synchronization Segment, Sync\_Seg, is that part of the bit time where edges of the CAN bus level are expected to occur. The distance between an edge that occurs outside of Sync\_Seg, and the Sync\_Seg is called the phase error of that edge. The Propagation Time Segment, Prop\_Seg, is intended to compensate for the physical delay times within the CAN network. The Phase Buffer Segments Phase\_Seg1 and Phase\_Seg2 surround the Sample Point. The (Re-)Synchronization Jump Width (SJW) defines how far a re-synchronization may move the Sample Point inside the limits defined by the Phase Buffer Segments to compensate for edge phase errors.

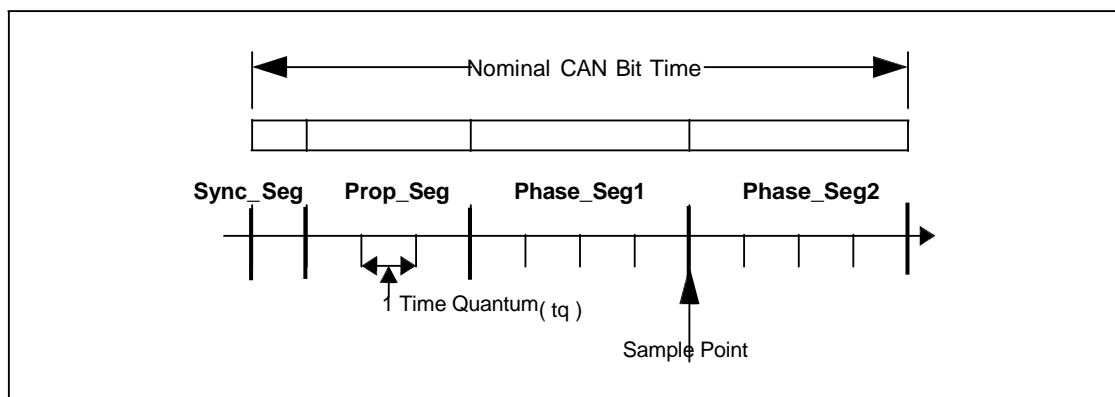


Figure 22-7 Bit Timing

| Parameter  | Range          | Remark  |
|--|----------------|---|
| BRP  | [1 .. 32]      | Defines the length of the time quantum $t_q$            |
| Sync_Seg   | 1 $t_q$        | Fixed length, synchronization of bus input to APB clock |
| Prop_Seg   | [1.. 8] $t_q$  | Compensates for the physical delay times                |
| Phase_Seg1   | [1..8] $t_q$   | May be lengthened temporarily by synchronization        |
| Phase_Seg2   | [1.. 8] $t_q$  | May be shortened temporarily by synchronization         |
| SJW  | [1 .. 4] $t_q$ | May not be longer than either Phase Buffer Segment      |
| This table describes the minimum programmable ranges required by the CAN protocol. |                |   |

Table 22-3 CAN Bit Time Parameters

A given bit rate may be met by different bit time configurations, but for the proper function of the CAN network the physical delay times and the oscillator's tolerance range have to be considered.

### 22.5.23 Propagation Time Segment

This part of the bit time is used to compensate physical delay times within the network. These delay times consist of the signal propagation time on the bus and the internal delay time of the CAN nodes.

Any CAN node synchronized to the bit stream on the CAN bus will be out of phase with the transmitter of that bit stream, caused by the signal propagation time between the two nodes. The CAN protocol's non-destructive bitwise arbitration and the dominant acknowledge bit provided by receivers of CAN messages requires that a CAN node transmitting a bit stream must also be able to receive dominant bits transmitted by other CAN nodes that are synchronized to that bit stream. The example in Figure 22-8 shows the phase shift and propagation times between two CAN nodes.

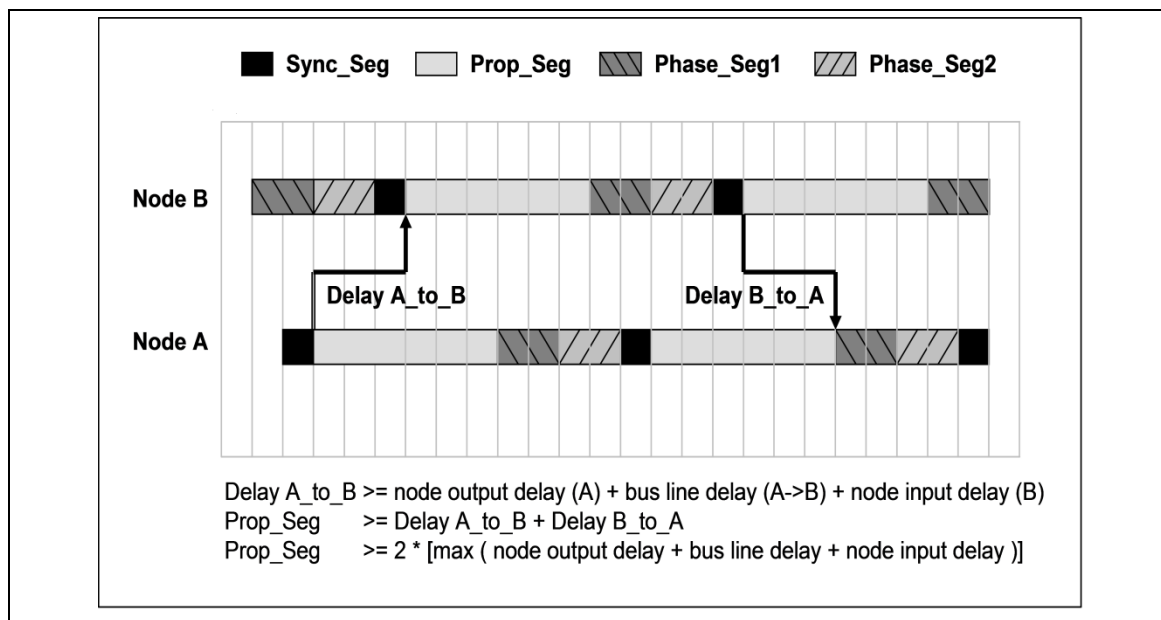


Figure 22-8 Propagation Time Segment

In this example, both nodes A and B are transmitters, performing an arbitration for the CAN bus. Node A has sent its Start of Frame bit less than one bit time earlier than node B, therefore node B has synchronized itself to the received edge from recessive to dominant. Since node B has received this edge delay (A\_to\_B) after it has been transmitted, B's bit timing segments are shifted with respect to A. Node B sends an identifier with higher priority and so it will win the arbitration at a specific identifier bit when it transmits a dominant bit while node A transmits a recessive bit. The dominant bit transmitted by node B will arrive at node A after the delay (B\_to\_A).

Due to oscillator tolerances, the actual position of node A's Sample Point can be anywhere inside the nominal range of node A's Phase Buffer Segments, so the bit transmitted by node B must arrive at node A before the start of Phase\_Seg1. This condition defines the length of Prop\_Seg.

If the edge from recessive to dominant transmitted by node B arrives at node A after the start of Phase\_Seg1, it can happen that node A samples a recessive bit instead of a dominant bit, resulting in a bit error and the destruction of the current frame by an error flag.

The error occurs only when two nodes arbitrate for the CAN bus that have oscillators of opposite ends of the tolerance range and that are separated by a long bus line. This is an example of a minor error in the bit timing configuration (Prop\_Seg too short) that causes sporadic bus errors.

Some CAN implementations provide an optional 3 Sample Mode but the C\_CAN does not. In this mode, the CAN bus input signal passes a digital low-pass filter, using three samples and a majority logic to determine the valid bit value. This results in an additional input delay of  $1 t_q$ , requiring a longer Prop\_Seg.

#### 22.5.24 Phase Buffer Segments and Synchronization

The Phase Buffer Segments (Phase\_Seg1 and Phase\_Seg2) and the Synchronization Jump Width (SJW) are used to compensate for the oscillator tolerance. The Phase Buffer Segments may be lengthened or shortened by synchronization.

Synchronizations occur on edges from recessive to dominant, their purpose is to control the distance between edges and Sample Points.

Edges are detected by sampling the actual bus level in each time quantum and comparing it with the bus level at the previous Sample Point. A synchronization may be done only if a recessive bit was sampled at the previous Sample Point and if the bus level at the actual time quantum is dominant.

An edge is synchronous if it occurs inside of Sync\_Seg, otherwise the distance between edge and the end of Sync\_Seg is the edge phase error, measured in time quanta. If the edge occurs before Sync\_Seg, the phase error is negative, else it is positive.

Two types of synchronization exist, Hard Synchronization and Re-synchronization.

A Hard Synchronization is done once at the start of a frame and inside a frame only when Re-synchronizations occur.

- **Hard Synchronization**

After a hard synchronization, the bit time is restarted with the end of Sync\_Seg, regardless of the edge phase error. Thus hard synchronization forces the edge, which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time.

- **Bit Re-synchronization**

Re-synchronization leads to a shortening or lengthening of the bit time such that the position of the sample point is shifted with regard to the edge.

When the phase error of the edge which causes Re-synchronization is positive, Phase\_Seg1 is lengthened. If the magnitude of the phase error is less than SJW, Phase\_Seg1 is lengthened by the magnitude of the phase error, else it is lengthened by SJW.

When the phase error of the edge, which causes Re-synchronization is negative, Phase\_Seg2 is shortened. If the magnitude of the phase error is less than SJW, Phase\_Seg2 is shortened by the magnitude of the phase error, else it is shortened by SJW.

When the magnitude of the phase error of the edge is less than or equal to the programmed value of SJW, the results of Hard Synchronization and Re-synchronization are the same. If the magnitude of the phase error is larger than SJW, the Re-synchronization cannot compensate the phase error completely, an error (phase error - SJW) remains.

Only one synchronization may be done between two Sample Points. The Synchronizations maintain a minimum distance between edges and Sample Points, giving the bus level time to stabilize and filtering out spikes that are shorter than (Prop\_Seg + Phase\_Seg1).

Apart from noise spikes, most synchronizations are caused by arbitration. All nodes synchronize “hard” on the edge transmitted by the “leading” transceiver that started transmitting first, but they cannot become ideally synchronized due to propagation delay times. The “leading” transmitter does not necessarily win the arbitration, therefore the receivers have to synchronize themselves to different transmitters that subsequently “take the lead” and that are differently synchronized to the previously “leading” transmitter. The same happens at the acknowledge field, where the transmitter and some of the receivers will have to synchronize to that receiver that “takes the lead” in the transmission of the dominant acknowledge bit.

Synchronizations after the end of the arbitration will be caused by oscillator tolerance, when the differences in the oscillator’s clock periods of transmitter and receivers sum up during the time between synchronizations (at most ten bits). These summarized differences may not be longer than the SJW, limiting the oscillator’s tolerance range.

The examples in Figure 22-9 show how the Phase Buffer Segments are used to compensate for phase errors. There are three drawings of each two consecutive bit timings. The upper drawing

shows the synchronization on a “late” edge, the lower drawing shows the synchronization on an “early” edge, and the middle drawing is the reference without synchronization.

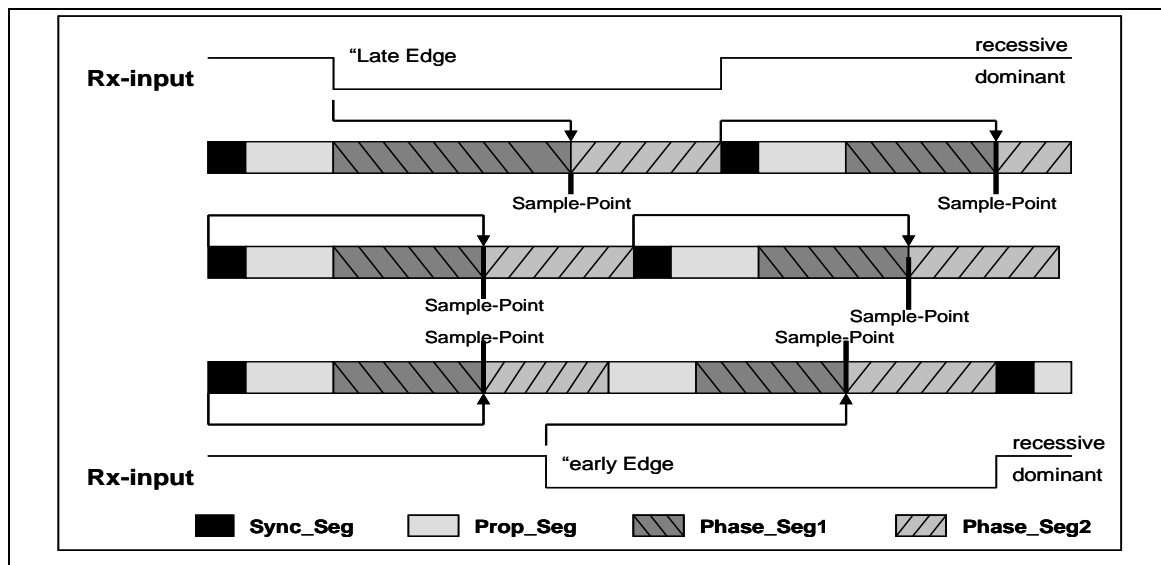


Figure 22-9 Synchronization on “late” and “early” Edges

In the first example, an edge from recessive to dominant occurs at the end of Prop\_Seg. The edge is “late” since it occurs after the Sync\_Seg. Reacting to the “late” edge, Phase\_Seg1 is lengthened so that the distance from the edge to the Sample Point is the same as it would have been from the Sync\_Seg to the Sample Point if no edge had occurred. The phase error of this “late” edge is less than SJW, so it is fully compensated and the edge from dominant to recessive at the end of the bit, which is one nominal bit time long, occurs in the Sync\_Seg.

In the second example, an edge from recessive to dominant occurs during Phase\_Seg2. The edge is “early” since it occurs before a Sync\_Seg. Reacting to the “early” edge, Phase\_Seg2 is shortened and Sync\_Seg is omitted, so that the distance from the edge to the Sample Point is the same as it would have been from an Sync\_Seg to the Sample Point if no edge had occurred. As in the previous example, the magnitude of this “early” edge’s phase error is less than SJW, so it is fully compensated.

The Phase Buffer Segments are lengthened or shortened temporarily only; at the next bit time, the segments return to their nominal programmed values.

In these examples, the bit timing is seen from the point of view of the CAN implementation’s state machine, where the bit time starts and ends at the Sample Points. The state machine omits Sync\_Seg when synchronizing on an “early” edge because it cannot subsequently redefine that time quantum of Phase\_Seg2 where the edge occurs to be the Sync\_Seg.

The examples in Figure 22-10 show how short dominant noise spikes are filtered by synchronizations. In both examples the spike starts at the end of Prop\_Seg and has the length of (Prop\_Seg + Phase\_Seg1).

In the first example, the Synchronisation Jump Width is greater than or equal to the phase error of the spike’s edge from recessive to dominant. Therefore the Sample Point is shifted after the end of the spike; a recessive bus level is sampled.

In the second example, SJW is shorter than the phase error, so the Sample Point cannot be shifted far enough; the dominant spike is sampled as actual bus level.

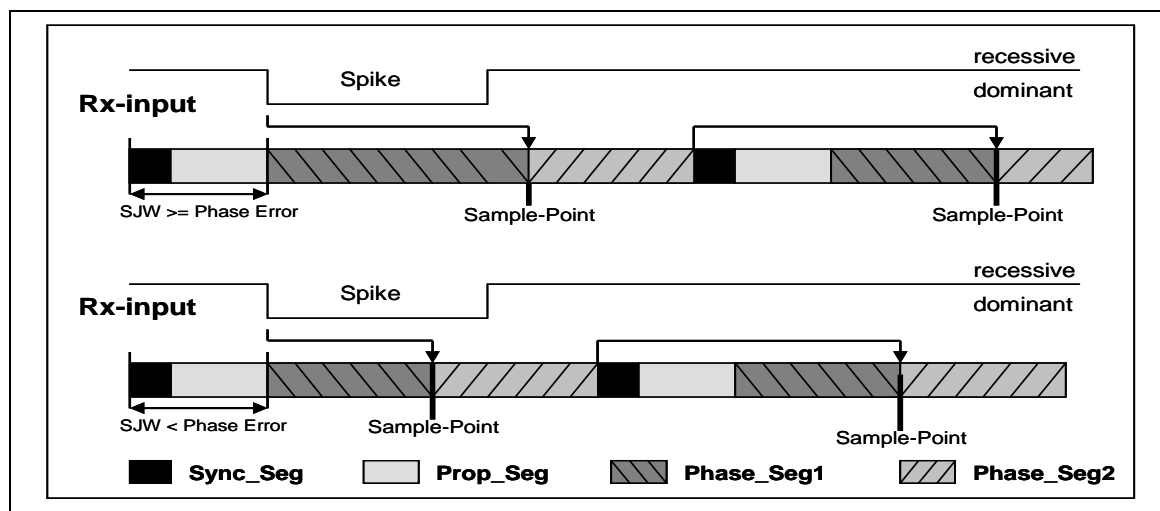


Figure 22-10 Filtering of Short Dominant Spikes

### 22.5.25 Oscillator Tolerance Range

The oscillator tolerance range was increased when the CAN protocol was developed from version 1.1 to version 1.2 (version 1.0 was never implemented in silicon). The option to synchronize on edges from dominant to recessive became obsolete, only edges from recessive to dominant are considered for synchronization. The protocol update to version 2.0 (A and B) had no influence on the oscillator tolerance.

The tolerance range  $df$  for an oscillator frequency  $f_{osc}$  around the nominal frequency  $f_{nom}$  is:

$$(1 - df) \times f_{nom} \leq f_{osc} \leq (1 + df) \times f_{nom}$$

It depends on the proportions of Phase\_Seg1, Phase\_Seg2, SJW, and the bit time. The maximum tolerance  $df$  is defined by two conditions (both shall be met):

$$I: df \leq \frac{\text{Min}(\text{Phase\_Seg1}, \text{Phase\_Seg2})}{2 \times (13 \times \text{bit\_time} - \text{Phase\_Seg2})}$$

$$II: df \leq \frac{\text{SJW}}{20 \times \text{bit\_time}}$$

**Note:** These conditions base on the APB clock =  $f_{osc}$ .

It has to be considered that SJW may not be larger than the smaller of the Phase Buffer Segments and that the Propagation Time Segment limits that part of the bit time that may be used for the Phase Buffer Segments.

The combination Prop\_Seg = 1 and Phase\_Seg1 = Phase\_Seg2 = SJW = 4 allows the largest possible oscillator tolerance of 1.58%. This combination with a Propagation Time Segment of only 10% of the bit time is not suitable for short bit times; it can be used for bit rates of up to 125 kBit/s (bit time = 8  $\mu$ s) with a bus length of 40 m.

#### 22.5.26 Configuring the CAN Protocol Controller

In most CAN implementations and also in the C\_CAN, the bit timing configuration is programmed in two register bytes. The sum of Prop\_Seg and Phase\_Seg1 (as TSEG1) is combined with Phase\_Seg2 (as TSEG2) in one byte, SJW and BRP are combined in the other byte.

In these bit timing registers, the four components TSEG1, TSEG2, SJW, and BRP have to be programmed to a numerical value that is one less than its functional value. Therefore, instead of values in the range of [1..n], values in the range of [0..n-1] are programmed. That way, e.g. SJW (functional range of [1..4]) is represented by only two bits.

Therefore the length of the bit time is (programmed values)  $[TSEG1 + TSEG2 + 3] t_q$  or (functional values)  $[Sync\_Seg + Prop\_Seg + Phase\_Seg1 + Phase\_Seg2] t_q$ .



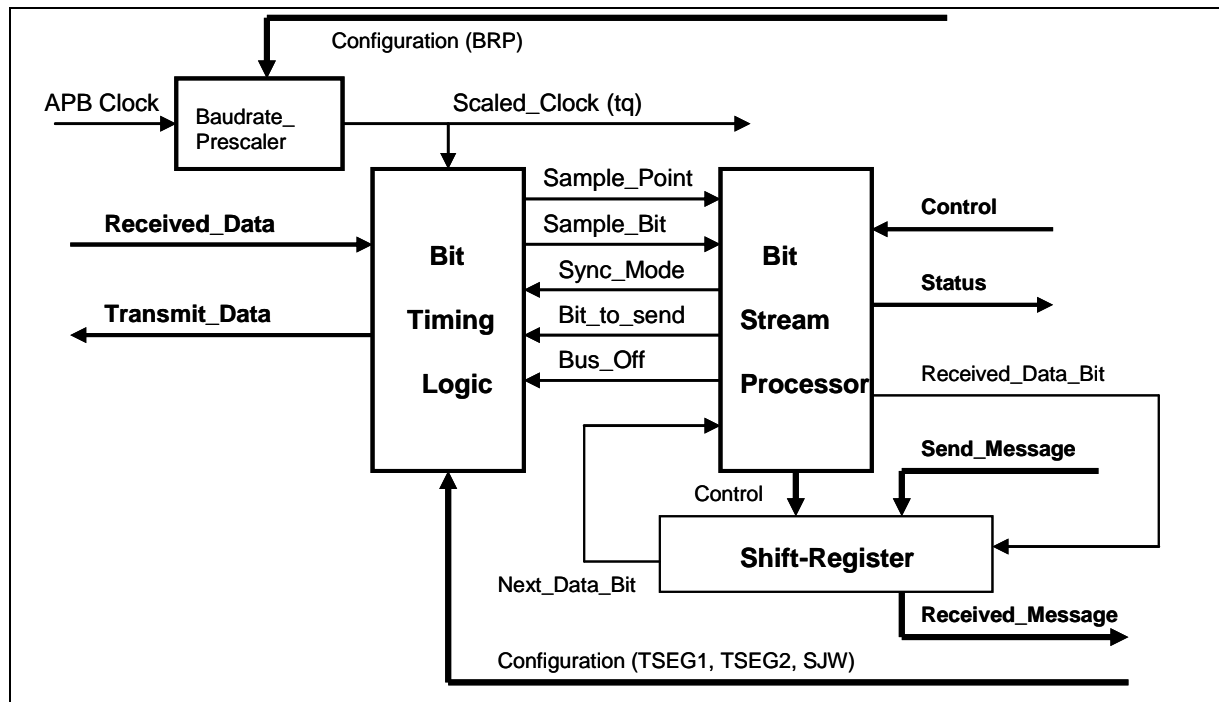


Figure 22-11 Structure of the CAN Core's CAN Protocol Controller

The data in the bit timing registers is the configuration input of the CAN protocol controller. The Baud Rate Prescaler (configured by BRP) defines the length of the time quantum, the basic time unit of the bit time; the Bit Timing Logic (configured by TSEG1, TSEG2, and SJW) defines the number of time quanta in the bit time.

The processing of the bit time, the calculation of the position of the Sample Point, and occasional synchronizations are controlled by the BTL (Bit Timing Logic) state machine, which is evaluated once each time quantum. The rest of the CAN protocol controller, the BSP (Bit Stream Processor) state machine is evaluated once each bit time, at the Sample Point.

The Shift Register sends the messages serially and parallelizes received messages. Its loading and shifting is controlled by the BSP.

The BSP translates messages into frames and vice versa. It generates and discards the enclosing fixed format bits, inserts and extracts stuff bits, calculates and checks the CRC code, performs the error management, and decides which type of synchronization is to be used. It is evaluated at the Sample Point and processes the sampled bus input bit. The time that is needed to calculate the next bit to be sent after the Sample point (e.g. data bit, CRC bit, stuff bit, error flag, or idle) is called the Information Processing Time (IPT).

The IPT is application specific but may not be longer than  $2 t_q$ ; the IPT for the C\_CAN is  $0 t_q$ . Its length is the lower limit of the programmed length of Phase\_Seg2. In case of a synchronization, Phase\_Seg2 may be shortened to a value less than IPT, which does not affect bus timing.



### 22.5.27 Calculating Bit Timing Parameters

Usually, the calculation of the bit timing configuration starts with a desired bit rate or bit time. The resulting bit time (1/bit rate) must be an integer multiple of the APB clock period.

The bit time may consist of 4 to 25 time quanta, the length of the time quantum  $t_q$  is defined by the Baud Rate Prescaler with  $t_q = (\text{Baud Rate Prescaler})/f_{\text{apb\_clk}}$ . Several combinations may lead to the desired bit time, allowing iterations of the following steps.

First part of the bit time to be defined is the Prop\_Seg. Its length depends on the delay times measured in the APB clock. A maximum bus length as well as a maximum node delay has to be defined for expandible CAN bus systems. The resulting time for Prop\_Seg is converted into time quanta (rounded up to the nearest integer multiple of  $t_q$ ).

The Sync\_Seg is 1  $t_q$  long (fixed), leaving (bit time – Prop\_Seg – 1)  $t_q$  for the two Phase Buffer Segments. If the number of remaining  $t_q$  is even, the Phase Buffer Segments have the same length, Phase\_Seg2 = Phase\_Seg1, else Phase\_Seg2 = Phase\_Seg1 + 1.

The minimum nominal length of Phase\_Seg2 has to be regarded as well. Phase\_Seg2 may not be shorter than the IPT of the CAN controller, which, depending on the actual implementation, is in the range of  $[0..2] t_q$ .

The length of the Synchronization Jump Width is set to its maximum value, which is the minimum of 4 and Phase\_Seg1.

The oscillator tolerance range necessary for the resulting configuration is calculated by the formulas given in Section 5.13.6.10.4: Oscillator Tolerance Range

If more than one configuration is possible, that configuration allowing the highest oscillator tolerance range should be chosen.

CAN nodes with different system clocks require different configurations to come to the same bit rate. The calculation of the propagation time in the CAN network, based on the nodes with the longest delay times, is done once for the whole network.

The oscillator tolerance range of the CAN systems is limited by that node with the lowest tolerance range.

The calculation may show that bus length or bit rate have to be decreased or that the stability of the oscillator frequency has to be increased in order to find a protocol compliant configuration of the CAN bit timing. The resulting configuration is written into the Bit Timing Register: (Phase\_Seg2-1) & (Phase\_Seg1+Prop\_Seg-1) & (SynchronisationJumpWidth-1)&(Prescaler-1)

## Example for Bit Timing at High Baud Rate

In this example, the frequency of APB\_CLK is 10 MHz, BRP is 0, the bit rate is 1 MBit/s.

|                           |      |    |   |
|---------------------------|------|----|---|
| $T_q$                     | 100  | ns | $= t_{APB\_CLK}$  |
| delay of bus driver       | 50   | ns |   |
| delay of receiver circuit | 30   | ns |   |
| delay of bus line (40m)   | 220  | ns |   |
| $t_{Prop}$                | 600  | ns | $= 6 \cdot t_q$   |
| $t_{SJW}$                 | 100  | ns | $= 1 \cdot t_q$   |
| $t_{TSeg1}$               | 700  | ns | $= t_{Prop} + t_{SJW}$  |
| $t_{TSeg2}$               | 200  | ns | $= \text{Information Processing Time} + 1 \cdot t_q$  |
| $t_{Sync-Seg}$            | 100  | ns | $= 1 \cdot t_q$   |
| bit time                  | 1000 | ns | $= t_{Sync-Seg} + t_{TSeg1} + t_{TSeg2}$  |
| tolerance for APB_CLK     | 0.39 | %  | $= \frac{Min(PB1, PB2)}{2 \times 13 \times (bit\ time - PB2)}$ $= \frac{0.1us}{2 \times (13 \times (1us - 0.2us))}$ |

In this example, the concatenated bit time parameters are (2-1)<sub>3</sub>&(7-1)<sub>4</sub>&(1-1)<sub>2</sub>&(1-1)<sub>6</sub>, the Bit Timing Register is programmed to= 0x1600.

### Example for Bit Timing at Low Baud Rate

In this example, the frequency of APB\_CLK is 2 MHz, BRP is 1, the bit rate is 100 KBit/s.

|                           |      |  |
|---------------------------|------|--|
| $t_q$                     | 1    | $s = 2 \cdot t_{APB\_CLK}$   |
| delay of bus driver       | 200  | ns   |
| delay of receiver circuit | 80   | ns   |
| delay of bus line (40m)   | 220  | ns   |
| $t_{Prop}$                | 1    | $s = 1 \cdot t_q$  |
| $t_{SJW}$                 | 4    | $s = 4 \cdot t_q$  |
| $t_{TSeg1}$               | 5    | $s = t_{Prop} + t_{SJW}$   |
| $t_{TSeg2}$               | 4    | $s = \text{Information Processing Time} + 3 \cdot t_q$   |
| $t_{Sync-Seg}$            | 1    | $s = 1 \cdot t_q$  |
| bit time                  | 10   | $s = t_{Sync-Seg} + t_{TSeg1} + t_{TSeg2}$   |
| tolerance for APB_CLK     | 1.58 | $\% = \frac{Min(PB1, PB2)}{2 \times 13 \times (bit\ time - PB2))}$ $= \frac{4us}{2 \times (13 \times (10us - 4us))}$ |

In this example, the concatenated bit time parameters are  $(4-1)_3 \& (5-1)_4 \& (4-1)_2 \& (2-1)_6$ , the Bit Timing Register is programmed to= 0x34C1.

### 22.6 Register Description

The C\_CAN allocates an address space of 256 bytes. The registers are organized as 16-bit registers.

The two sets of interface registers (IF1 and IF2) control software access to the Message RAM. They buffer the data to be transferred to and from the RAM, avoiding conflicts between software accesses and message reception/transmission.

### 22.7 Register Map

R: read only, W: write only, R/W: both read and write\

| Register                    | Offset      | R/W | Description                            | Reset Value                |
|-----------------------------|-------------|-----|--|----------------------------|
| <b>CAN Base Address:</b>    |             |     |  |                            |
| <b>CAN_BA = 0x4018_0000</b> |             |     |  |                            |
| <b>CAN_CON</b>              | CAN_BA+0x00 | R/W | Control Register                       | 0x0000_0001                |
| <b>CAN_STATUS</b>           | CAN_BA+0x04 | R/W | Status Register                        | 0x0000_0000                |
| <b>CAN_ERR</b>              | CAN_BA+0x08 | R   | Error Counter Register                 | 0x0000_0000                |
| <b>CAN_BTMR</b>             | CAN_BA+0x0C | R/W | Bit Timing Register                    | 0x0000_2301                |
| <b>CAN_IIDR</b>             | CAN_BA+0x10 | R   | Interrupt Identifier Register          | 0x0000_0000                |
| <b>CAN_TEST</b>             | CAN_BA+0x14 | R/W | Test Register                          | 0x0000_00x0 <sup>[1]</sup> |
| <b>CAN_BRPE</b>             | CAN_BA+0x18 | R/W | Baud Rate Prescaler Extension Register | 0x0000_0000                |
| <b>CAN_IF1_CREQ</b>         | CAN_BA+0x20 | R/W | IF1 Command Request Register           | 0x0000_0001                |
| <b>CAN_IF2_CREQ</b>         | CAN_BA+0x80 | R/W | IF2 Command Request Register           | 0x0000_0001                |
| <b>CAN_IF1_CMASK</b>        | CAN_BA+0x24 | R/W | IF1 Command Mask Register              | 0x0000_0000                |
| <b>CAN_IF2_CMASK</b>        | CAN_BA+0x84 | R/W | IF2 Command Mask Register              | 0x0000_0000                |
| <b>CAN_IF1_MASK1</b>        | CAN_BA+0x28 | R/W | IF1 Mask 1 Register                    | 0x0000_FFFF                |
| <b>CAN_IF2_MASK1</b>        | CAN_BA+0x88 | R/W | IF2 Mask 1 Register                    | 0x0000_FFFF                |
| <b>CAN_IF1_MASK2</b>        | CAN_BA+0x2C | R/W | IF1 Mask 2 Register                    | 0x0000_FFFF                |
| <b>CAN_IF2_MASK2</b>        | CAN_BA+0x8C | R/W | IF2 Mask 2 Register                    | 0x0000_FFFF                |
| <b>CAN_IF1_ARB1</b>         | CAN_BA+0x30 | R/W | IF1 Arbitration 1 Register             | 0x0000_0000                |
| <b>CAN_IF2_ARB1</b>         | CAN_BA+0x90 | R/W | IF2 Arbitration 1 Register             | 0x0000_0000                |
| <b>CAN_IF1_ARB2</b>         | CAN_BA+0x34 | R/W | IF1 Arbitration 2 Register             | 0x0000_0000                |
| <b>CAN_IF2_ARB2</b>         | CAN_BA+0x94 | R/W | IF2 Arbitration 2 Register             | 0x0000_0000                |
| <b>CAN_IF1_MCON</b>         | CAN_BA+0x38 | R/W | IF1 Message Control Register           | 0x0000_0000                |
| <b>CAN_IF2_MCON</b>         | CAN_BA+0x98 | R/W | IF2 Message Control Register           | 0x0000_0000                |
| <b>CAN_IF1_DAT_A1</b>       | CAN_BA+0x3C | R/W | IF1 Data A1 Register                   | 0x0000_0000                |
| <b>CAN_IF1_DAT_A2</b>       | CAN_BA+0x40 | R/W | IF1 Data A2 Register                   | 0x0000_0000                |
| <b>CAN_IF1_DAT_B1</b>       | CAN_BA+0x44 | R/W | IF1 Data B1 Register                   | 0x0000_0000                |
| <b>CAN_IF1_DAT_B2</b>       | CAN_BA+0x48 | R/W | IF1 Data B2 Register                   | 0x0000_0000                |
| <b>CAN_IF2_DAT_A1</b>       | CAN_BA+0x9C | R/W | IF2 Data A1 Register                   | 0x0000_0000                |
| <b>CAN_IF2_DAT_A2</b>       | CAN_BA+0xA0 | R/W | IF2 Data A2 Register                   | 0x0000_0000                |

|                       |              |     |                                 |             |
|-----------------------|--------------|-----|---------------------------------|-------------|
| <b>CAN_IF2_DAT_B1</b> | CAN_BA+0xA4  | R/W | IF2 Data B1 Register            | 0x0000_0000 |
| <b>CAN_IF2_DAT_B2</b> | CAN_BA+0xA8  | R/W | IF2 Data B2 Register            | 0x0000_0000 |
| <b>CAN_TXREQ1</b>     | CAN_BA+0x100 | R   | Transmission Request Register 1 | 0x0000_0000 |
| <b>CAN_TXREQ2</b>     | CAN_BA+0x104 | R   | Transmission Request Register 2 | 0x0000_0000 |
| <b>CAN_NDAT1</b>      | CAN_BA+0x120 | R   | New Data Register 1             | 0x0000_0000 |
| <b>CAN_NDAT2</b>      | CAN_BA+0x124 | R   | New Data Register 2             | 0x0000_0000 |
| <b>CAN_IPND1</b>      | CAN_BA+0x140 | R   | Interrupt Pending Register 1    | 0x0000_0000 |
| <b>CAN_IPND2</b>      | CAN_BA+0x144 | R   | Interrupt Pending Register 2    | 0x0000_0000 |
| <b>CAN_MVLD1</b>      | CAN_BA+0x160 | R   | Message Valid Register 1        | 0x0000_0000 |
| <b>CAN_MVLD2</b>      | CAN_BA+0x164 | R   | Message Valid Register 2        | 0x0000_0000 |
| <b>CAN_WU_EN</b>      | CAN_BA+0x168 | R/W | Wake-Up Enable Register         | 0x0000_0000 |
| <b>CAN_WU_STATUS</b>  | CAN_BA+0x16C | R/W | Wake-Up Status Register         | 0x0000_0000 |

**Note:** 1. 0x00 & 0br0000000, where r signifies the actual value of the CAN\_RX

2. IFn: The two sets of Message Interface Registers – IF1 and IF2, have identical function

## 22.8 CAN Interface Reset State

After hardware reset, the C\_CAN registers hold the reset values given in the register description in [CAN register map](#).

Additionally the *bus-off* state is reset and the output CAN\_TXD is set to recessive (HIGH). The value 0x0001 (Init = '1') in the CAN Control Register enables software initialization. The C\_CAN does not influence the CAN bus until the application software resets the Init bit to '0'.

The data stored in the Message RAM is not affected by a hardware reset. After powering on, the contents of the Message RAM are undefined.

### CAN Register Map for Each Bit Function

| Addr<br>offset | Register Name | 1<br>5    | 1<br>4   | 1<br>3 | 1<br>2   | 1<br>1 | 1<br>0 | 9 | 8 | 7      | 6        | 5              | 4     | 3       | 2         | 1       | 0        |        |
|----------------|---------------|-----------|----------|--------|----------|--------|--------|---|---|--------|----------|----------------|-------|---------|-----------|---------|----------|--------|
| 00h            | CAN_CON       | Reserved  |          |        |          |        |        |   |   |        | Test     | CCE            | DAR   | Res     | EIE       | SIE     | IE       | Init   |
| 04h            | CAN_STATUS    | Reserved  |          |        |          |        |        |   |   |        | BOff     | EWarn          | EPass | RxOk    | TxOk      | LEC     |          |        |
| 08h            | CAN_ERR       | RP        | REC6-0   |        |          |        |        |   |   | TEC7-0 |          |                |       |         |           |         |          |        |
| 0Ch            | CAN_BTIME     | Res       | TSeg2    |        |          | TSeg1  |        |   |   | SJW    |          | BRP            |       |         |           |         |          |        |
| 10h            | CAN_IIDR      | IntId15-8 |          |        |          |        |        |   |   |        | IntId7-0 |                |       |         |           |         |          |        |
| 14h            | CAN_TEST      | Reserved  |          |        |          |        |        |   |   |        | Rx       | Tx1            | Tx0   | LBack   | Silent    | Basic   | Reserved |        |
| 18h            | CAN_BRPE      | Reserved  |          |        |          |        |        |   |   |        |          |                |       | BRPE    |           |         |          |        |
| 20h            | CAN_IF1_CREQ  | Busy      | Reserved |        |          |        |        |   |   |        |          | Message Number |       |         |           |         |          |        |
| 24h            | CAN_IF1_CMASK | Reserved  |          |        |          |        |        |   |   |        | WR/RD    | Mask           | Arb   | Control | ClrIntPnd | TxRqst/ | Data A   | Data B |
| 28h            | CAN_IF1_MASK1 | Msk15-0   |          |        |          |        |        |   |   |        |          |                |       |         |           |         |          |        |
| 2Ch            | CAN_IF1_MASK2 | MXtd      | MDir     | Res    | Msk28-16 |        |        |   |   |        |          |                |       |         |           |         |          |        |
| 30h            | CAN_IF1_ARB1  | ID15-0    |          |        |          |        |        |   |   |        |          |                |       |         |           |         |          |        |
| 34h            | CAN_IF1_ARB2  | MsgVal    | Xtd      | Dir    | ID28-16  |        |        |   |   |        |          |                |       |         |           |         |          |        |



| Addr<br>offset | Register Name  | 1<br>5   | 1<br>4   | 1<br>3 | 1<br>2   | 1<br>1 | 1<br>0 | 9     | 8      | 7     | 6        | 5              | 4       | 3        | 2       | 1      | 0      |
|----------------|----------------|----------|----------|--------|----------|--------|--------|-------|--------|-------|----------|----------------|---------|----------|---------|--------|--------|
| 38h            | CAN_IF1_MCON   | NewDat   | MsgLst   | IntPnd | UMask    | TxE    | RxE    | RmtEn | TxRqst | EoB   | Reserved |                |         | DLC3-0   |         |        |        |
| 3Ch            | CAN_IF1_DAT_A1 | Data1    |          |        |          |        |        |       |        | Data0 |          |                |         |          |         |        |        |
| 40h            | CAN_IF1_DAT_A2 | Data3    |          |        |          |        |        |       |        | Data2 |          |                |         |          |         |        |        |
| 44h            | CAN_IF1_DAT_B1 | Data5    |          |        |          |        |        |       |        | Data4 |          |                |         |          |         |        |        |
| 48h            | CAN_IF1_DAT_B2 | Data7    |          |        |          |        |        |       |        | Data6 |          |                |         |          |         |        |        |
| 80h            | CAN_IF2_CREQ   | Busy     | Reserved |        |          |        |        |       |        |       |          | Message Number |         |          |         |        |        |
| 84h            | CAN_IF2_CMASK  | Reserved |          |        |          |        |        |       |        | WR/RD | Mask     | Arb            | Control | CiIntPnd | TxRqst/ | Data A | Data B |
| 88h            | CAN_IF2_MASK1  | Msk15-0  |          |        |          |        |        |       |        |       |          |                |         |          |         |        |        |
| 8Ch            | CAN_IF2_MASK2  | MXtd     | MDir     | Res.   | Msk28-16 |        |        |       |        |       |          |                |         |          |         |        |        |
| 90h            | CAN_IF2_ARB1   | ID15-0   |          |        |          |        |        |       |        |       |          |                |         |          |         |        |        |
| 94h            | CAN_IF2_ARB2   | MsgVal   | Xtd      | Dir    | ID28-16  |        |        |       |        |       |          |                |         |          |         |        |        |
| 98h            | CAN_IF2_MCON   | NewDat   | MsgLst   | IntPnd | UMask    | TxE    | RxE    | RmtEn | TxRqst | EoB   | Reserved |                |         | DLC3-0   |         |        |        |
| 9Ch            | CAN_IF2_DAT_A1 | Data1    |          |        |          |        |        |       |        | Data0 |          |                |         |          |         |        |        |

| Addr<br>offset | Register Name  | 1<br>5      | 1<br>4 | 1<br>3 | 1<br>2 | 1<br>1 | 1<br>0 | 9 | 8 | 7     | 6 | 5 | 4 | 3 | 2 | 1 | 0             |
|----------------|----------------|-------------|--------|--------|--------|--------|--------|---|---|-------|---|---|---|---|---|---|---------------|
| A0h            | CAN_IF2_DAT_A2 | Data3       |        |        |        |        |        |   |   | Data2 |   |   |   |   |   |   |               |
| A4h            | CAN_IF2_DAT_B1 | Data5       |        |        |        |        |        |   |   | Data4 |   |   |   |   |   |   |               |
| A8h            | CAN_IF2_DAT_B2 | Data7       |        |        |        |        |        |   |   | Data6 |   |   |   |   |   |   |               |
| 100h           | CAN_TXREQ1     | TxRqst16-1  |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |
| 104h           | CAN_TXREQ2     | TxRqst32-17 |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |
| 120h           | CAN_NDAT1      | NewDat16-1  |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |
| 124h           | CAN_NDAT2      | NewDat32-17 |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |
| 140h           | CAN_IPND1      | IntPnd16-1  |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |
| 144h           | CAN_IPND2      | IntPnd32-17 |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |
| 160h           | CAN_MVLD1      | MsgVal16-1  |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |
| 164h           | CAN_MVLD2      | MsgVal32-17 |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |
| 168h           | CAN_WU_EN      | Reserved    |        |        |        |        |        |   |   |       |   |   |   |   |   |   | WAKUP<br>EN   |
| 16Ch           | CAN_WU_STATUS  | Reserved    |        |        |        |        |        |   |   |       |   |   |   |   |   |   | WAKUP<br>_STS |
| 170h           | CAN_RAM_CEN    | Reserved    |        |        |        |        |        |   |   |       |   |   |   |   |   |   | RAM_<br>CEN   |
| Others         | Reserved       | Reserved    |        |        |        |        |        |   |   |       |   |   |   |   |   |   |               |

Table 22-4 CAN Register Map for Each Bit Function

**Note:** Reserved bits are read as 0' except for IFn Mask 2 Register where they are read as '1'.

Res. = Reserved

### Control Register (CAN\_CON)

| Register | Offset      | R/W | Description      | Reset Value |
|----------|-------------|-----|------------------|-------------|
| CAN_CON  | CAN_BA+0x00 | R/W | Control Register | 0x0000_0001 |

|          |     |     |          |     |     |    |      |
|----------|-----|-----|----------|-----|-----|----|------|
| 31       | 30  | 29  | 28       | 27  | 26  | 25 | 24   |
| Reserved |     |     |          |     |     |    |      |
| 23       | 22  | 21  | 20       | 19  | 18  | 17 | 16   |
| Reserved |     |     |          |     |     |    |      |
| 15       | 14  | 13  | 12       | 11  | 10  | 9  | 8    |
| Reserved |     |     |          |     |     |    |      |
| 7        | 6   | 5   | 4        | 3   | 2   | 1  | 0    |
| Test     | CCE | DAR | Reserved | EIE | SIE | IE | Init |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | <b>Reserved</b><br>There are reserved bits.<br>These bits are always read as '0' and must always be written with '0'  |
| [7]    | Test        | <b>Test Mode Enable</b><br>1 = Test mode.<br>0 = Normal operation.  |
| [6]    | CCE         | <b>Configuration Change Enable</b><br>1 = Write access to the Bit Timing Register (CAN_BTTIME & CAN_BRP) allowed. (while Init bit =1).<br>0 = No write access to the Bit Timing Register.   |
| [5]    | DAR         | <b>Disable Automatic Re-transmission</b><br>1 = Automatic Retransmission Disabled.<br>0 = Automatic Retransmission of disturbed messages Enabled.   |
| [4]    | Reserved    | <b>Reserved</b><br>This is a reserved bit. This bit is always read as '0' and must always be written with '0'.  |
| [3]    | EIE         | <b>Error Interrupt Enable</b><br>1 = Enabled - A change in the bits BOff or EWarn in the Status Register will generate an interrupt.<br>0 = Disabled - No Error Status Interrupt will be generated.                                     |
| [2]    | SIE         | <b>Status Change Interrupt Enable</b><br>1 = Enabled - An interrupt will be generated when a message transfer is successfully completed or a CAN bus error is detected.<br>0 = Disabled - No Status Change Interrupt will be generated. |
| [1]    | IE          | <b>Module Interrupt Enable</b><br>1 = Enabled.  |

|     |             |   |
|-----|-------------|---|
|     |             | 0 = Disabled.   |
| [0] | <b>Init</b> | <b>Init Initialization</b><br>1 = Initialization is started.<br>0 = Normal operation. |

**Note:** The bus-off recovery sequence (see CAN Specification Rev. 2.0) cannot be shortened by setting or resetting the Init bit. If the device goes in the bus-off state, it will set Init of its own accord, stopping all bus activities. Once Init has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 \* 11 consecutive recessive bits) before resuming normal operations. At the end of the bus-off recovery sequence, the Error Management Counters will be reset.

During the waiting time after resetting Init, each time a sequence of 11 recessive bits has been monitored, a Bit0Error code is written to the Status Register, enabling the CPU to readily check whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the proceeding of the bus-off recovery sequence.

Status Register (CAN\_STATUS)

| Register   | Offset      | R/W | Description     | Reset Value |
|------------|-------------|-----|-----------------|-------------|
| CAN_STATUS | CAN_BA+0x04 | R/W | Status Register | 0x0000_0000 |

|          |       |       |      |      |     |    |    |
|----------|-------|-------|------|------|-----|----|----|
| 31       | 30    | 29    | 28   | 27   | 26  | 25 | 24 |
| Reserved |       |       |      |      |     |    |    |
| 23       | 22    | 21    | 20   | 19   | 18  | 17 | 16 |
| Reserved |       |       |      |      |     |    |    |
| 15       | 14    | 13    | 12   | 11   | 10  | 9  | 8  |
| Reserved |       |       |      |      |     |    |    |
| 7        | 6     | 5     | 4    | 3    | 2   | 1  | 0  |
| BOFF     | EWarn | EPass | RxOK | TxOK | LEC |    |    |

| Bits   | Description   |
|--------|---|
| [31:8] | <b>Reserved</b><br><b>Reserved</b><br>This is a reserved bit. This bit is always read as '0' and must always be written with '0'.   |
| [7]    | <b>BOff</b><br><b>Bus-off Status (Read Only)</b><br>1 = CAN module is in bus-off state.<br>0 = CAN module is not in bus-off state.  |
| [6]    | <b>EWarn</b><br><b>Error Warning Status (Read Only)</b><br>1 = At least one of the error counters in the EML has reached the error warning limit of 96.<br>0 = Both error counters are below the error warning limit of 96.   |
| [5]    | <b>EPass</b><br><b>Error Passive (Read Only)</b><br>1 = CAN Core is in the error passive state as defined in the CAN Specification.<br>0 = CAN Core is error active.  |
| [4]    | <b>RxOK</b><br><b>Received a Message Successfully</b><br>1 = A message has been successfully received since this bit was last reset by the CPU (independent of the result of acceptance filtering).<br>0 = No message has been successfully received since this bit was last reset by the CPU. This bit is never reset by the CAN Core.   |
| [3]    | <b>TxOK</b><br><b>Transmitted a Message Successfully</b><br>1 = Since this bit was last reset by the CPU, a message has been successfully (error free and acknowledged by at least one other node) transmitted.<br>0 = Since this bit was reset by the CPU, no message has been successfully transmitted. This bit is never reset by the CAN Core.  |
| [2:0]  | <b>LEC</b><br><b>Last Error Code (Type of the Last Error to Occur on the CAN Bus)</b><br>The LEC field holds a code, which indicates the type of the last error to occur on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. The unused code '7' may be written by the CPU to check for updates. <b>Error! Reference source not found.</b> describes the error code. |

| Error Code | Meanings  |
|------------|---|
| 0          | No Error  |
| 1          | Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.  |
| 2          | Form Error: A fixed format part of a received frame has the wrong format.   |
| 3          | AckError: The message this CAN Core transmitted was not acknowledged by another node.   |
| 4          | Bit1Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant.  |
| 5          | Bit0Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), though the device wanted to send a dominant level (data or identifier bit logical value '0'), but the monitored Bus value was recessive. During bus-off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceedings of the bus-off recovery sequence (indicating the bus is not stuck at <i>dominant</i> or continuously disturbed). |
| 6          | CRCErrror: The CRC check sum was incorrect in the message received, the CRC received for an incoming message does not match with the calculated CRC for the received data.  |
| 7          | Unused: When the LEC shows the value '7', no CAN bus event was detected since the CPU wrote this value to the LEC.  |

Table 22-5 Error Code

### Status Interrupts

A Status Interrupt is generated by bits **BOff** and **EWarn** (Error Interrupt) or by **RxOk**, **TxOk**, and **LEC** (Status Change Interrupt) assumed that the corresponding enable bits in the CAN Control Register are set. A change of bit **EPass** or a write to **RxOk**, **TxOk**, or **LEC** will never generate a Status Interrupt.

Reading the Status Register will clear the Status Interrupt value (8000h) in the Interrupt Register, if it is pending.

**Error Counter Register (CAN\_ERR)**

| Register | Offset      | R/W | Description            | Reset Value |
|----------|-------------|-----|------------------------|-------------|
| CAN_ERR  | CAN_BA+0x08 | R   | Error Counter Register | 0x0000_0000 |

|          |          |    |    |    |    |    |    |
|----------|----------|----|----|----|----|----|----|
| 31       | 30       | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |          |    |    |    |    |    |    |
| 23       | 22       | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |          |    |    |    |    |    |    |
| 15       | 14       | 13 | 12 | 11 | 10 | 9  | 8  |
| RP       | REC[6:0] |    |    |    |    |    |    |
| 7        | 6        | 5  | 4  | 3  | 2  | 1  | 0  |
| TEC[7:0] |          |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | Reserved    | <b>Reserved</b><br>This is a reserved bit. This bit is always read as '0' and must always be written with '0'.  |
| [15]    | RP          | <b>Receive Error Passive</b><br>1 = The Receive Error Counter has reached the error passive level as defined in the CAN Specification.<br>0 = The Receive Error Counter is below the error passive level. |
| [14:8]  | REC         | <b>Receive Error Counter</b><br>Actual state of the Receive Error Counter. Values between 0 and 127.  |
| [7:0]   | TEC         | <b>Transmit Error Counter</b><br>Actual state of the Transmit Error Counter. Values between 0 and 255.  |

**Bit Timing Register (CAN\_BTME)**

| Register | Offset      | R/W | Description         | Reset Value |
|----------|-------------|-----|---------------------|-------------|
| CAN_BTME | CAN_BA+0x0C | R/W | Bit Timing Register | 0x0000_2301 |

|          |       |     |    |       |    |    |    |
|----------|-------|-----|----|-------|----|----|----|
| 31       | 30    | 29  | 28 | 27    | 26 | 25 | 24 |
| Reserved |       |     |    |       |    |    |    |
| 23       | 22    | 21  | 20 | 19    | 18 | 17 | 16 |
| Reserved |       |     |    |       |    |    |    |
| 15       | 14    | 13  | 12 | 11    | 10 | 9  | 8  |
| Reserved | TSeg2 |     |    | TSeg1 |    |    |    |
| 7        | 6     | 5   | 4  | 3     | 2  | 1  | 0  |
| SJW      |       | BRP |    |       |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:15] | Reserved    | <b>Reserved</b><br>This is a reserved bit. This bit is always read as '0' and must always be written with '0'.  |
| [14:12] | TSeg2       | <b>Time Segment After sample Point</b><br>0x0-0x7: Valid values for TSeg2 are [0 ... 7]. The actual interpretation by hardware of this value is such that one more than the value programmed here is used.  |
| [11:8]  | TSeg1       | <b>Time Segment before the sample Point Minus Sync_seg</b><br>0x01-0x0F: valid values for TSeg1 are [1 ... 15]. The actual interpretation by hardware of this value is such that one more than the value programmed is used.  |
| [7:6]   | SJW         | <b>(Re)Synchronization Jump Width</b><br>0x0-0x3: Valid programmed values are [0 ... 3]. The actual interpretation by hardware of this value is such that one more than the value programmed here is used.  |
| [5:0]   | BRP         | <b>Baud Rate Prescaler</b><br>0x01-0x3F: The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Baud Rate Prescaler are [ 0 ... 63 ]. The actual interpretation by hardware of this value is such that one more than the value programmed here is used. |

Note With a module clock APB\_CLK of 8 MHz, the reset value of 0x2301 configures the C\_CAN for a bit rate of 500 kBit/s. The registers are only writable if bits CCE and Init in the CAN Control Register are set.



### Interrupt Identify Register (CAN\_IIDR)

| Register | Offset      | R/W | Description                   | Reset Value |
|----------|-------------|-----|-------------------------------|-------------|
| CAN_IIDR | CAN_BA+0x10 | R   | Interrupt Identifier Register | 0x0000_0000 |

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved    |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved    |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| IntId[15:8] |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| IntId[7:0]  |    |    |    |    |    |    |    |

| Bits   | Description   |
|--------|---|
| [15:0] | <p><b>IntId</b></p> <p><b>Interrupt Identifier (Indicates the source of the interrupt. Refer to Error! Reference source not found.)</b></p> <p>If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority, disregarding their chronological order. An interrupt remains pending until the application software has cleared it. If <b>IntId</b> is different from 0x0000 and IE is set, the IRQ interrupt signal to the EIC is active. The interrupt remains active until <b>IntId</b> is back to value 0x0000 (the cause of the interrupt is reset) or until <b>IE</b> is reset.</p> <p>The Status Interrupt has the highest priority. Among the message interrupts, the Message Object's interrupt priority decreases with increasing message number.</p> <p>A message interrupt is cleared by clearing the Message Object's <b>IntPnd</b> bit. The Status Interrupt is cleared by reading the Status Register.</p> |

| IntId Value   | Meanings   |
|---------------|--|
| 0x0000        | No Interrupt is Pending                              |
| 0x0001-0x0020 | Number of Message Object which caused the interrupt. |
| 0x0021-0x7FFF | Unused   |
| 0x8000        | Status Interrupt                                     |
| 0x8001-0xFFFF | Unused   |

Table 22-6 Source of Interrupts

### Test Register (CAN TEST)

| Register | Offset      | R/W | Description   | Reset Value |
|----------|-------------|-----|---------------|-------------|
| CAN_TEST | CAN_BA+0x14 | R/W | Test Register | 0x0000_00x0 |

|          |         |    |       |        |       |     |    |
|----------|---------|----|-------|--------|-------|-----|----|
| 31       | 30      | 29 | 28    | 27     | 26    | 25  | 24 |
| Reserved |         |    |       |        |       |     |    |
| 23       | 22      | 21 | 20    | 19     | 18    | 17  | 16 |
| Reserved |         |    |       |        |       |     |    |
| 15       | 14      | 13 | 12    | 11     | 10    | 9   | 8  |
| Reserved |         |    |       |        |       |     |    |
| 7        | 6       | 5  | 4     | 3      | 2     | 1   | 0  |
| Rx       | Tx[1:0] |    | LBack | Silent | Basic | Res |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.  |
| [7]    | Rx          | <b>Monitors the actual value of CAN_RXD Pin (Read Only)</b><br>1 = The CAN bus is recessive (CAN_RXD = '1').<br>0 = The CAN bus is dominant (CAN_RXD = '0').  |
| [6:5]  | Tx          | <b>Tx[1:0]: Control of CAN_TXD pin</b><br>00 = Reset value, CAN_TXD is controlled by the CAN Core<br>01 = Sample Point can be monitored at CAN_TXD pin<br>10 = CAN_TXD pin drives a dominant ('0') value.<br>11 = CAN_TXD pin drives a recessive ('1') value. |
| [4]    | LBack       | <b>Loop Back Mode</b><br>1 = Loop Back Mode Enabled.<br>0 = Loop Back Mode Disabled.  |
| [3]    | Silent      | <b>Silent Mode</b><br>1 = The module is in Silent mode.<br>0 = Normal operation.  |
| [2]    | Basic       | <b>Basic Mode</b><br>1= IF1 Registers used as Tx Buffer, IF2 Registers used as Rx Buffer.<br>0 = Basic Mode disabled.   |
| [1:0]  | Res         | <b>Reserved</b><br>There are reserved bits.<br>These bits are always read as '0' and must always be written with '0'.   |

Reset value: 0000 0000 R000 0000 b (R: current value of RX pin)

Write access to the Test Register is enabled by setting the Test bit in the CAN Control Register. The different test functions may be combined, but **Tx[1-0]** “00” disturbs message transfer.

### Baud Rate Prescaler Extension REGISTER (CAN\_BRPE)

| Register | Offset      | R/W | Description                            | Reset Value |
|----------|-------------|-----|--|-------------|
| CAN_BRPE | CAN_BA+0x18 | R/W | Baud Rate Prescaler Extension Register | 0x0000_0000 |

|          |    |    |    |      |    |    |    |
|----------|----|----|----|------|----|----|----|
| 31       | 30 | 29 | 28 | 27   | 26 | 25 | 24 |
| Reserved |    |    |    |      |    |    |    |
| 23       | 22 | 21 | 20 | 19   | 18 | 17 | 16 |
| Reserved |    |    |    |      |    |    |    |
| 15       | 14 | 13 | 12 | 11   | 10 | 9  | 8  |
| Reserved |    |    |    |      |    |    |    |
| 7        | 6  | 5  | 4  | 3    | 2  | 1  | 0  |
| Reserved |    |    |    | BRPE |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:4] | Reserved    | Reserved<br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.   |
| [3:0]  | BRPE        | <b>BRPE: Baud Rate Prescaler Extension</b><br>0x00-0x0F: By programming <b>BRPE</b> , the Baud Rate Prescaler can be extended to values up to 1023. The actual interpretation by hardware is that one more than the value programmed by <b>BRPE</b> (MSBs) and <b>BTIME</b> (LSBs) is used. |

### Message Interface Register Sets

There are two sets of Interface Registers, which are used to control the CPU access to the Message RAM. The Interface Registers avoid conflict between the CPU accesses to the Message RAM and CAN message reception and transmission by buffering the data to be transferred. A complete Message Object or parts of the Message Object may be transferred between the Message RAM and the IF $n$  Message Buffer registers in one single transfer.

The function of the two interface register sets is identical except for the Basic test mode. They can be used the way one set of registers is used for data transfer to the Message RAM while the other set of registers is used for the data transfer from the Message RAM, allowing both processes to be interrupted by each other. Table 22-7 (IF1 and IF2 Message Interface Register Set) provides an overview of the two Interface Register sets.

Each set of Interface Registers consists of Message Buffer Registers controlled by their own Command Registers. The Command Mask Register specifies the direction of the data transfer and which parts of a Message Object will be transferred. The Command Request Register is used to select a Message Object in the Message RAM as target or source for the transfer and to start the action specified in the Command Mask Register.

| Address     | IF1 Register Set    | Address     | IF2 Register Set    |
|-------------|---------------------|-------------|---------------------|
| CAN_BA+0x20 | IF1 Command Request | CAN_BA+0x80 | IF2 Command Request |
| CAN_BA+0x24 | IF1 Command Mask    | CAN_BA+0x84 | IF2 Command Mask    |
| CAN_BA+0x28 | IF1 Mask 1          | CAN_BA+0x88 | IF2 Mask 1          |
| CAN_BA+0x2C | IF1 Mask 2          | CAN_BA+0x8C | IF2 Mask 2          |
| CAN_BA+0x30 | IF1 Arbitration 1   | CAN_BA+0x90 | IF2 Arbitration 1   |
| CAN_BA+0x34 | IF1 Arbitration 2   | CAN_BA+0x94 | IF2 Arbitration 2   |
| CAN_BA+0x38 | IF1 Message Control | CAN_BA+0x98 | IF2 Message Control |
| CAN_BA+0x3C | IF1 Data A 1        | CAN_BA+0x9C | IF2 Data A 1        |
| CAN_BA+0x40 | IF1 Data A 2        | CAN_BA+0xA0 | IF2 Data A 2        |
| CAN_BA+0x44 | IF1 Data B 1        | CAN_BA+0xA4 | IF2 Data B 1        |
| CAN_BA+0x48 | IF1 Data B 2        | CAN_BA+0xA8 | IF2 Data B 2        |

Table 22-7 IF1 and IF2 Message Interface Register

**IFn Command Request Register (CAN\_IFn\_CREQ)**

| Register     | Offset      | R/W | Description                  | Reset Value |
|--------------|-------------|-----|------------------------------|-------------|
| CAN_IF1_CREQ | CAN_BA+0x20 | R/W | IF1 Command Request Register | 0x0000_0001 |
| CAN_IF2_CREQ | CAN_BA+0x80 | R/W | IF2 Command Request Register | 0x0000_0001 |

|          |     |                |    |    |    |    |    |
|----------|-----|----------------|----|----|----|----|----|
| 31       | 30  | 29             | 28 | 27 | 26 | 25 | 24 |
| Reserved |     |                |    |    |    |    |    |
| 23       | 22  | 21             | 20 | 19 | 18 | 17 | 16 |
| Reserved |     |                |    |    |    |    |    |
| 15       | 14  | 13             | 12 | 11 | 10 | 9  | 8  |
| Busy     | Res |                |    |    |    |    |    |
| 7        | 6   | 5              | 4  | 3  | 2  | 1  | 0  |
| Res      |     | Message Number |    |    |    |    |    |

| Bits   | Description    |  |
|--------|----------------|--|
| [15]   | Busy           | <b>Busy Flag</b><br>1 = Writing to the IFn Command Request Register is in progress. This bit can only be read by software.<br>0 = Read/write action has been finished.   |
| [14:6] | Reserved       | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.   |
| [5:0]  | Message Number | <b>Message Number</b><br>0x01-0x20: Valid Message Number, the Message Object in the Message RAM is selected for data transfer.<br>0x00: Not a valid Message Number, interpreted as 0x20.<br>0x21-0x3F: Not a valid Message Number, interpreted as 0x01-0x1F. |

A message transfer is started as soon as the application software has written the message number to the Command Request Register. With this write operation, the Busy bit is automatically set to notify the CPU that a transfer is in progress. After a waiting time of 3 to 6 APB\_CLK periods, the transfer between the Interface Register and the Message RAM is completed. The Busy bit is cleared.

**Note:**When a Message Number that is not valid is written into the Command Request Register, the Message Number will be transformed into a valid value and that Message Object will be transferred.

### IFn Command Mask Register (CAN IFn\_CMASK)

The control bits of the IFn Command Mask Register specify the transfer direction and select which of the IFn Message Buffer Registers are source or target of the data transfer.

| Register      | Offset      | R/W | Description               | Reset Value |
|---------------|-------------|-----|---------------------------|-------------|
| CAN_IF1_CMASK | CAN_BA+0x24 | R/W | IF1 Command Mask Register | 0x0000_0000 |
| CAN_IF2_CMASK | CAN_BA+0x84 | R/W | IF2 Command Mask Register | 0x0000_0000 |

|          |      |     |         |           |                   |       |       |
|----------|------|-----|---------|-----------|-------------------|-------|-------|
| 31       | 30   | 29  | 28      | 27        | 26                | 25    | 24    |
| Reserved |      |     |         |           |                   |       |       |
| 23       | 22   | 21  | 20      | 19        | 18                | 17    | 16    |
| Reserved |      |     |         |           |                   |       |       |
| 15       | 14   | 13  | 12      | 11        | 10                | 9     | 8     |
| Reserved |      |     |         |           |                   |       |       |
| 7        | 6    | 5   | 4       | 3         | 2                 | 1     | 0     |
| WR_RD    | Mask | Arb | Control | CtrIntPnd | TxRqst/<br>NewDat | DAT_A | DAT_B |

| Bits   | Description |   |
|--------|-------------|---|
| [31:8] | Reserved    | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.  |
| [7]    | WR_RD       | <b>Write / Read</b><br>1 = Write: Transfer data from the selected Message Buffer Registers to the Message Object addressed by the Command Request Register.<br>0 = Read: Transfer data from the Message Object addressed by the Command Request Register into the selected Message Buffer Registers.                        |
| [6]    | Mask        | <b>Access Mask Bits</b><br><u>Direction = Write</u><br>1 = Transfer <b>Identifier Mask + MDir + MXtd</b> to Message Object.<br>0 = Mask bits unchanged.<br><u>Direction = Read</u><br>1 = Transfer <b>Identifier Mask + MDir + MXtd</b> to IFn Message Buffer Register.<br>0 = Mask bits unchanged.                         |
| [5]    | Arb         | <b>Access Arbitration Bits</b><br><u>Direction = Write</u><br>1 = Transfer <b>Identifier + Dir + Xtd + MsgVal</b> to Message Object<br>0 = Arbitration bits unchanged.<br><u>Direction = Read</u><br>1 = Transfer <b>Identifier + Dir + Xtd + MsgVal</b> to IFn Message Buffer Register.<br>0 = Arbitration bits unchanged. |
| [4]    | Control     | <b>Control Access Control Bits</b>  |

|     |                      |  |
|-----|----------------------|--|
|     |                      | <u>Direction = Write</u><br>1 = Transfer Control Bits to Message Object.<br>0 = Control Bits unchanged<br><u>Direction = Read</u><br>1 = Transfer Control Bits to IFn Message Buffer Register.<br>0 = Control Bits unchanged.  |
| [3] | <b>CIntPnd</b>       | <b>Clear Interrupt Pending Bit</b><br><u>Direction = Write</u><br>When writing to a Message Object, this bit is ignored.<br><u>Direction = Read</u><br>1 = Clear <b>IntPnd</b> bit in the Message Object.<br>0 = <b>IntPnd</b> bit remains unchanged.  |
| [2] | <b>TxRqst/NewDat</b> | <b>Access Transmission Request Bit</b> when <u>Direction = Write</u><br>1 = Set TxRqst bit.<br>0 = TxRqst bit unchanged.<br><b>Note:</b> If a transmission is requested by programming bit <b>TxRqst/NewDat</b> in the IFn Command Mask Register, bit TxRqst in the IFn Message Control Register will be ignored.<br><b>Access New Data Bit</b> when <u>Direction = Read</u><br>1 = Clear <b>NewDat</b> bit in the Message Object<br>0 = <b>NewDat</b> bit remains unchanged.<br><b>Note:</b> A read access to a Message Object can be combined with the reset of the control bits <b>IntPnd</b> and <b>NewDat</b> . The values of these bits transferred to the IFn Message Control Register always reflect the status before resetting these bits. |
| [1] | <b>DAT_A</b>         | <b>Access Data Bytes [3:0]</b><br><u>Direction = Write</u><br>1 = Transfer Data Bytes [3:0] to Message Object<br>0 = Data Bytes [3:0] unchanged.<br><u>Direction = Read</u><br>1 = Transfer Data Bytes [3:0] to IFn Message Buffer Register.<br>0 = Data Bytes [3:0] unchanged.  |
| [0] | <b>DAT_B</b>         | <b>Access Data Bytes [7:4]</b><br><u>Direction = Write</u><br>1 = Transfer Data Bytes [7:4] to Message Object.<br>0 = Data Bytes [7:4] unchanged.<br><u>Direction = Read</u><br>1 = Transfer Data Bytes [7:4] to IFn Message Buffer Register.<br>0 = Data Bytes [7:4] unchanged.   |



**IFn Mask 1 Register (CAN\_IFn\_MASK1)**

| Register      | Offset      | R/W | Description         | Reset Value |
|---------------|-------------|-----|---------------------|-------------|
| CAN_IF1_MASK1 | CAN_BA+0x28 | R/W | IF1 Mask 1 Register | 0x0000_FFFF |
| CAN_IF2_MASK1 | CAN_BA+0x88 | R/W | IF2 Mask 1 Register | 0x0000_FFFF |

|           |    |    |    |    |    |    |    |
|-----------|----|----|----|----|----|----|----|
| 31        | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved  |    |    |    |    |    |    |    |
| 23        | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved  |    |    |    |    |    |    |    |
| 15        | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Msk[15:8] |    |    |    |    |    |    |    |
| 7         | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Msk[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | Reserved    | Reserved<br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.   |
| [15:0]  | Msk[15:0]   | Identifier Mask 15-0<br>1 = The corresponding identifier bit is used for acceptance filtering.<br>0 = The corresponding bit in the identifier of the message object cannot inhibit the match in the acceptance filtering. |

### IFn Mask 2 Register (CAN\_IFn\_MASK2)

| Register      | Offset      | R/W | Description         | Reset Value |
|---------------|-------------|-----|---------------------|-------------|
| CAN_IF1_MASK2 | CAN_BA+0x2C | R/W | IF1 Mask 2 Register | 0x0000_FFFF |
| CAN_IF2_MASK2 | CAN_BA+0x8C | R/W | IF2 Mask 2 Register | 0x0000_FFFF |

|            |      |          |            |    |    |    |    |
|------------|------|----------|------------|----|----|----|----|
| 31         | 30   | 29       | 28         | 27 | 26 | 25 | 24 |
| Reserved   |      |          |            |    |    |    |    |
| 23         | 22   | 21       | 20         | 19 | 18 | 17 | 16 |
| Reserved   |      |          |            |    |    |    |    |
| 15         | 14   | 13       | 12         | 11 | 10 | 9  | 8  |
| MXtd       | MDir | Reserved | Msk[28:24] |    |    |    |    |
| 7          | 6    | 5        | 4          | 3  | 2  | 1  | 0  |
| Msk[23:16] |      |          |            |    |    |    |    |

| Bits    | Description   |
|---------|---|
| [31:16] | <b>Reserved</b><br><b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.   |
| [15]    | <b>MXtd</b><br><b>Mask Extended Identifier</b><br>1 = The extended identifier bit (IDE) is used for acceptance filtering.<br>0 = The extended identifier bit (IDE) has no effect on the acceptance filtering.<br><b>Note:</b> When 11-bit ("standard") Identifiers are used for a Message Object, the identifiers of received Data Frames are written into bits <b>ID28</b> to <b>ID18</b> . For acceptance filtering, only these bits together with mask bits <b>Msk28</b> to <b>Msk18</b> are considered. |
| [14]    | <b>MDir</b><br><b>Mask Message Direction</b><br>1 = The message direction bit ( <b>Dir</b> ) is used for acceptance filtering.<br>0 = The message direction bit ( <b>Dir</b> ) has no effect on the acceptance filtering.   |
| [13]    | <b>Reserved</b><br><b>Reserved</b><br>This is reserved bit. The bit is always read as '1' and must always be written with '1'.  |
| [12:0]  | <b>Msk[28:16]</b><br><b>Identifier Mask 28-16</b><br>1 = The corresponding identifier bit is used for acceptance filtering.<br>0 = The corresponding bit in the identifier of the message object cannot inhibit the match in the acceptance filtering.  |

### IFn Arbitration 1 Register (CAN IFn\_ARB1)

| Register     | Offset      | R/W | Description                | Reset Value |
|--------------|-------------|-----|----------------------------|-------------|
| CAN_IF1_ARB1 | CAN_BA+0x30 | R/W | IF1 Arbitration 1 Register | 0x0000_0000 |
| CAN_IF2_ARB1 | CAN_BA+0x90 | R/W | IF2 Arbitration 1 Register | 0x0000_0000 |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| ID[15:8] |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| ID[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | Reserved    | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.                       |
| [15:0]  | ID[15:0]    | <b>Message Identifier 15-0</b><br>ID28 - ID0, 29-bit Identifier ("Extended Frame").<br>ID28 - ID18, 11-bit Identifier ("Standard Frame") |

IFn Arbitration 2 Register (CAN IFn\_ARB2)

| Register     | Offset      | R/W | Description                | Reset Value |
|--------------|-------------|-----|----------------------------|-------------|
| CAN_IF1_ARB2 | CAN_BA+0x34 | R/W | IF1 Arbitration 2 Register | 0x0000_0000 |
| CAN_IF2_ARB2 | CAN_BA+0x94 | R/W | IF2 Arbitration 2 Register | 0x0000_0000 |

|           |     |     |           |    |    |    |    |
|-----------|-----|-----|-----------|----|----|----|----|
| 31        | 30  | 29  | 28        | 27 | 26 | 25 | 24 |
| Reserved  |     |     |           |    |    |    |    |
| 23        | 22  | 21  | 20        | 19 | 18 | 17 | 16 |
| Reserved  |     |     |           |    |    |    |    |
| 15        | 14  | 13  | 12        | 11 | 10 | 9  | 8  |
| MsgVal    | Xtd | Dir | ID[28:24] |    |    |    |    |
| 7         | 6   | 5   | 4         | 3  | 2  | 1  | 0  |
| ID[23:16] |     |     |           |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | Reserved    | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.  |
| [15]    | MsgVal      | <b>Message Valid</b><br>1 = The Message Object is configured and should be considered by the Message Handler.<br>0 = The Message Object is ignored by the Message Handler.<br><b>Note:</b> The application software must reset the <b>MsgVal</b> bit of all unused Messages Objects during the initialization before it resets bit <b>Init</b> in the CAN Control Register. This bit must also be reset before the identifier <b>Id28-0</b> , the control bits <b>Xtd</b> , <b>Dir</b> , or the Data Length Code <b>DLC3-0</b> are modified, or if the Messages Object is no longer required. |
| [14]    | Xtd         | <b>Extended Identifier</b><br>1 = The 29-bit ("extended") Identifier will be used for this Message Object.<br>0 = The 11-bit ("standard") Identifier will be used for this Message Object.  |
| [13]    | Dir         | <b>Message Direction</b><br>1 = Direction is transmit<br>On <b>TxRqst</b> , the respective Message Object is transmitted as a Data Frame. On reception of a Remote Frame with matching identifier, the <b>TxRqst</b> bit of this Message Object is set (if <b>RmtEn</b> = one).<br>0 = Direction is receive<br>On <b>TxRqst</b> , a Remote Frame with the identifier of this Message Object is transmitted. On reception of a Data Frame with matching identifier, that message is stored in this Message Object.   |
| [12:0]  | ID[28:16]   | <b>Message Identifier 28-16</b><br>ID28 - ID0, 29-bit Identifier ("Extended Frame").<br>ID28 - ID18, 11-bit Identifier ("Standard Frame")   |

**IFn Message Control Register (CAN\_IFn\_MCON)**

| Register     | Offset      | R/W | Description                  | Reset Value |
|--------------|-------------|-----|------------------------------|-------------|
| CAN_IF1_MCON | CAN_BA+0x38 | R/W | IF1 Message Control Register | 0x0000_0000 |
| CAN_IF2_MCON | CAN_BA+0x98 | R/W | IF2 Message Control Register | 0x0000_0000 |

|          |          |        |       |          |      |       |        |
|----------|----------|--------|-------|----------|------|-------|--------|
| 31       | 30       | 29     | 28    | 27       | 26   | 25    | 24     |
| Reserved |          |        |       |          |      |       |        |
| 23       | 22       | 21     | 20    | 19       | 18   | 17    | 16     |
| Reserved |          |        |       |          |      |       |        |
| 15       | 14       | 13     | 12    | 11       | 10   | 9     | 8      |
| NewDat   | MsgLst   | IntPnd | UMask | TxIE     | RxIE | RmtEn | TxRqst |
| 7        | 6        | 5      | 4     | 3        | 2    | 1     | 0      |
| EoB      | Reserved |        |       | DLC[3:0] |      |       |        |

| Bits    | Description   |
|---------|---|
| [31:16] | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.  |
| [15]    | <b>NewDat</b><br>1 = The Message Handler or the application software has written new data into the data portion of this Message Object.<br>0 = No new data has been written into the data portion of this Message Object by the Message Handler since last time this flag was cleared by the application software.              |
| [14]    | <b>MsgLst</b><br><b>Message Lost (only valid for Message Objects with direction = receive)</b><br>1 = The Message Handler stored a new message into this object when <b>NewDat</b> was still set, the CPU has lost a message.<br>0 = No message lost since last time this bit was reset by the CPU.                             |
| [13]    | <b>IntPnd</b><br><b>Interrupt Pending</b><br>1 = This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority.<br>0 = This message object is not the source of an interrupt.              |
| [12]    | <b>UMask</b><br><b>Use Acceptance Mask</b><br>1 = Use Mask (Msk28-0, MXtd, and MDir) for acceptance filtering.<br>0 = Mask ignored.<br><b>Note:</b> If the <b>UMask</b> bit is set to one, the Message Object's mask bits have to be programmed during initialization of the Message Object before <b>MsgVal</b> is set to one. |
| [11]    | <b>TxIE</b><br><b>Transmit Interrupt Enable</b><br>1 = <b>IntPnd</b> will be set after a successful transmission of a frame.<br>0 = <b>IntPnd</b> will be left unchanged after the successful transmission of a frame.  |
| [10]    | <b>RxIE</b><br><b>Receive Interrupt Enable</b>  |

|       |                 |   |
|-------|-----------------|---|
|       |                 | <p>1 = <b>IntPnd</b> will be set after a successful reception of a frame.</p> <p>0 = <b>IntPnd</b> will be left unchanged after a successful reception of a frame.</p>  |
| [9]   | <b>RmtEn</b>    | <p><b>Remote Enable</b></p> <p>1 = At the reception of a Remote Frame, <b>TxRqst</b> is set.</p> <p>0 = At the reception of a Remote Frame, <b>TxRqst</b> is left unchanged.</p>  |
| [8]   | <b>TxRqst</b>   | <p><b>Transmit Request</b></p> <p>1 = The transmission of this Message Object is requested and is not yet done.</p> <p>0 = This Message Object is not waiting for transmission.</p>   |
| [7]   | <b>EoB</b>      | <p><b>End of Buffer</b></p> <p>1 = Single Message Object or last Message Object of a FIFO Buffer.</p> <p>0 = Message Object belongs to a FIFO Buffer and is not the last Message Object of that FIFO Buffer.</p> <p><b>Note:</b> This bit is used to concatenate two or more Message Objects (up to 32) to build a FIFO Buffer. <b>For single Message Objects (not belonging to a FIFO Buffer), this bit must always be set to one.</b></p>   |
| [6:4] | <b>Reserved</b> | <p><b>Reserved</b></p> <p>There are reserved bits. These bits are always read as '0' and must always be written with '0'.</p>   |
| [3:0] | <b>DLC</b>      | <p><b>Data Length Code</b></p> <p>0-8: Data Frame has 0-8 data bytes.</p> <p>9-15: Data Frame has 8 data bytes</p> <p><b>Note:</b> The Data Length Code of a Message Object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the Message Handler stores a data frame, it will write the DLC to the value given by the received message.</p> <p>Data 0: 1st data byte of a CAN Data Frame</p> <p>Data 1: 2nd data byte of a CAN Data Frame</p> <p>Data 2: 3rd data byte of a CAN Data Frame</p> <p>Data 3: 4th data byte of a CAN Data Frame</p> <p>Data 4: 5th data byte of a CAN Data Frame</p> <p>Data 5: 6th data byte of a CAN Data Frame</p> <p>Data 6: 7th data byte of a CAN Data Frame</p> <p>Data 7 : 8th data byte of a CAN Data Frame</p> <p><b>Note:</b> The <b>Data0</b> Byte is the first data byte shifted into the shift register of the CAN Core during a reception while the <b>Data7</b> byte is the last. When the Message Handler stores a Data Frame, it will write all the eight data bytes into a Message Object. If the Data Length Code is less than 8, the remaining bytes of the Message Object will be overwritten by <b>unspecified values</b>.</p> |

**IFn Data A1 Register (CAN IFn DAT A1)**

| Register       | Offset      | R/W | Description          | Reset Value |
|----------------|-------------|-----|----------------------|-------------|
| CAN_IF1_DAT_A1 | CAN_BA+0x3C | R/W | IF1 Data A1 Register | 0x0000_0000 |
| CAN_IF2_DAT_A1 | CAN_BA+0x9C | R/W | IF2 Data A1 Register | 0x0000_0000 |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Data1    |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Data0    |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | Reserved    | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'. |
| [15:8]  | Data1       | <b>Data byte 1</b><br>2nd data byte of a CAN Data Frame  |
| [7:0]   | Data0       | <b>Data byte 0</b><br>1st data byte of a CAN Data Frame  |

**IFn Data A2 Register (CAN IFn DAT A2)**

| Register       | Offset      | R/W | Description          | Reset Value |
|----------------|-------------|-----|----------------------|-------------|
| CAN_IF1_DAT_A2 | CAN_BA+0x40 | R/W | IF1 Data A2 Register | 0x0000_0000 |
| CAN_IF2_DAT_A2 | CAN_BA+0xA0 | R/W | IF2 Data A2 Register | 0x0000_0000 |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Data3    |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Data2    |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | Reserved    | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'. |
| [15:8]  | Data3       | <b>Data byte 3</b><br>4th data byte of CAN Data Frame  |
| [7:0]   | Data2       | <b>Data byte 2</b><br>3rd data byte of CAN Data Frame  |



**IFn Data B1 Register (CAN IFn DAT B1)**

| Register       | Offset      | R/W | Description          | Reset Value |
|----------------|-------------|-----|----------------------|-------------|
| CAN_IF1_DAT_B1 | CAN_BA+0x44 | R/W | IF1 Data B1 Register | 0x0000_0000 |
| CAN_IF2_DAT_B1 | CAN_BA+0xA4 | R/W | IF2 Data B1 Register | 0x0000_0000 |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Data5    |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Data4    |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | Reserved    | Reserved<br>There are reserved bits. These bits are always read as '0' and must always be written with '0'. |
| [15:8]  | Data5       | Data byte 5<br>6th data byte of CAN Data Frame  |
| [7:0]   | Data4       | Data byte 4<br>5th data byte of CAN Data Frame  |

**IFn Data B2 Register (CAN IFn DAT B2)**

| Register       | Offset      | R/W | Description          | Reset Value |
|----------------|-------------|-----|----------------------|-------------|
| CAN_IF1_DAT_B2 | CAN_BA+0x48 | R/W | IF1 Data B2 Register | 0x0000_0000 |
| CAN_IF2_DAT_B2 | CAN_BA+0xA8 | R/W | IF2 Data B2 Register | 0x0000_0000 |

|          |    |    |    |    |    |    |    |
|----------|----|----|----|----|----|----|----|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |    |    |    |    |    |    |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |    |    |    |    |    |    |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| Data7    |    |    |    |    |    |    |    |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| Data6    |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | Reserved    | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'. |
| [15:8]  | Data7       | <b>Data byte 7</b><br>8th data byte of CAN Data Frame.   |
| [7:0]   | Data6       | <b>Data byte 6</b><br>7th data byte of CAN Data Frame.   |

In a CAN Data Frame, Data0 is the first, Data7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first.

### Message Object in the Message Memory

There are 32 Message Objects in the Message RAM. To avoid conflicts between application software access to the Message RAM and CAN message reception and transmission, the CPU cannot directly access the Message Objects, these accesses are handled through the IF $n$  Interface Registers. Provides an overview of the structures of a Message Object.

| Message Object |               |      |      |              |        |       |        |       |       |        |       |        |
|----------------|---------------|------|------|--------------|--------|-------|--------|-------|-------|--------|-------|--------|
| UMask          | Msk<br>[28:0] | MXtd | MDir | EoB          | NewDat |       | MsgLst | RxIE  | TxIE  | IntPnd | RmtEn | TxRqst |
| MsgVal         | ID<br>[28:0]  | Xtd  | Dir  | DLC<br>[3:0] | Data0  | Data1 | Data2  | Data3 | Data4 | Data5  | Data6 | Data7  |

Table 22-8 Structure of a Message Object in Message Memory

The Arbitration Registers **ID28-0**, **Xtd**, and **Dir** are used to define the identifier and type of outgoing messages and are used (together with the mask registers **Msk28-0**, **MXtd**, and **MDir**) for acceptance filtering of incoming messages. A received message is stored in the valid Message Object with matching identifier and Direction = *receive* (Data Frame) or Direction = *transmit* (Remote Frame). Extended frames can be stored only in Message Objects with **Xtd** = one, standard frames in Message Objects with **Xtd** = 0. If a received message (Data Frame or Remote Frame) matches with more than one valid Message Object, it is stored into that with the lowest message number.

### Message Handler Registers

All Message Handler registers are read-only. Their contents (**TxRqst**, **NewDat**, **IntPnd**, and **MsgVal** bits of each Message Object and the Interrupt Identifier) are status information provided by the Message Handler FSM.

### Transmission Request Register 1 (CAN\_TXREQ1)

These registers hold the **TxRqst** bits of the 32 Message Objects. By reading the **TxRqst** bits, software can check which Message Object in a Transmission Request is pending. The **TxRqst** bit of a specific Message Object can be set/reset by the application software through the IFn Message Interface Registers or by the Message Handler after reception of a Remote Frame or after a successful transmission.

| Register   | Offset       | R/W | Description                     | Reset Value |
|------------|--------------|-----|---------------------------------|-------------|
| CAN_TXREQ1 | CAN_BA+0x100 | R   | Transmission Request Register 1 | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved     |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| TxRqst[16:9] |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TxRqst[8:1]  |    |    |    |    |    |    |    |

| Bits    | Description  |
|---------|--|
| [31:16] | <b>Reserved</b><br><b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.  |
| [15:0]  | <b>Transmission Request Bits 16-1 (of all Message Objects)</b><br>1 = The transmission of this Message Object is requested and is not done yet.<br>0 = This Message Object is not waiting for transmission.<br>These bits are read only. |

Transmission Request Register 2 (CAN\_TXREQ2)

| Register   | Offset       | R/W | Description                     | Reset Value |
|------------|--------------|-----|---------------------------------|-------------|
| CAN_TXREQ2 | CAN_BA+0x104 | R   | Transmission Request Register 2 | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved      |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved      |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| TxRqst[32:25] |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| TxRqst[24:17] |    |    |    |    |    |    |    |

| Bits    | Description  |
|---------|--|
| [31:16] | <p><b>Reserved</b></p> <p><b>Reserved</b></p> <p>There are reserved bits. These bits are always read as '0' and must always be written with '0'.</p>   |
| [15:0]  | <p><b>TxRqst[32:17]</b></p> <p><b>Transmission Request Bits 32-17 (of all Message Objects)</b></p> <p>1 = The transmission of this Message Object is requested and is not done yet.</p> <p>0 = This Message Object is not waiting for transmission.</p> <p>These bits are read only.</p> |

**New Data Register 1 (CAN\_NDAT1)**

These registers hold the **NewDat** bits of the 32 Message Objects. By reading out the **NewDat** bits, software can check for which Message Object the data portion was updated. The **NewDat** bit of a specific Message Object can be set/reset by software through the IFn Message Interface Registers or by the Message Handler after reception of a Data Frame or after a successful transmission.

| Register  | Offset       | R/W | Description         | Reset Value |
|-----------|--------------|-----|---------------------|-------------|
| CAN_NDAT1 | CAN_BA+0x120 | R   | New Data Register 1 | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved      |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved      |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| NewData[16:9] |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| NewData[8:1]  |    |    |    |    |    |    |    |

| Bits    | Description   |   |
|---------|---------------|---|
| [31:16] | Reserved      | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.  |
| [15:0]  | NewData[16:1] | <b>New Data Bits 16-1 (of all Message Objects)</b><br>1 = The Message Handler or the application software has written new data into the data portion of this Message Object.<br>0 = No new data has been written into the data portion of this Message Object by the Message Handler since the last time this flag was cleared by the application software. |

**New Data Register 2 (CAN\_NDAT2)**

| Register  | Offset       | R/W | Description         | Reset Value |
|-----------|--------------|-----|---------------------|-------------|
| CAN_NDAT2 | CAN_BA+0x124 | R   | New Data Register 2 | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved       |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved       |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| NewData[32:25] |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| NewData[24:17] |    |    |    |    |    |    |    |

| Bits    | Description   |
|---------|---|
| [31:16] | <p><b>Reserved</b></p> <p><b>Reserved</b></p> <p>There are reserved bits. These bits are always read as '0' and must always be written with '0'.</p>  |
| [15:0]  | <p><b>New Data Bits 32-17 (of all Message Objects)</b></p> <p>1 = The Message Handler or the application software has written new data into the data portion of this Message Object.</p> <p>0 = No new data has been written into the data portion of this Message Object by the Message Handler since the last time this flag was cleared by the application software.</p> |

### Interrupt Pending Register 1 (CAN\_IPND1)

These registers contain the **IntPnd** bits of the 32 Message Objects. By reading the **IntPnd** bits, software can check for which Message Object an interrupt is pending. The **IntPnd** bit of a specific Message Object can be set/reset by the application software through the IFn Message Interface Registers or by the Message Handler after reception or after a successful transmission of a frame. This will also affect the value of **IntId** in the Interrupt Register.

| Register  | Offset       | R/W | Description                  | Reset Value |
|-----------|--------------|-----|------------------------------|-------------|
| CAN_IPND1 | CAN_BA+0x140 | R   | Interrupt Pending Register 1 | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved     |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| IntPnd[16:9] |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| IntPnd[8:1]  |    |    |    |    |    |    |    |

| Bits    | Description  |
|---------|--|
| [31:16] | <b>Reserved</b><br><b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.  |
| [15:0]  | <b>Interrupt Pending Bits 16-1 (of all Message Objects)</b><br><b>IntPnd[16:1]</b><br>1 = This message object is the source of an interrupt.<br>0 = This message object is not the source of an interrupt. |



**Interrupt Pending Register 2 (CAN\_IPND2)**

| Register  | Offset       | R/W | Description                  | Reset Value |
|-----------|--------------|-----|------------------------------|-------------|
| CAN_IPND2 | CAN_BA+0x144 | R   | Interrupt Pending Register 2 | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved      |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved      |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| IntPnd[32:25] |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| IntPnd[24:17] |    |    |    |    |    |    |    |

| Bits    | Description  |
|---------|--|
| [31:16] | <p><b>Reserved</b></p> <p><b>Reserved</b></p> <p>There are reserved bits. These bits are always read as '0' and must always be written with '0'.</p>   |
| [15:0]  | <p><b>IntPnd[32:17]</b></p> <p><b>Interrupt Pending Bits 32-17(of all Message Objects)</b></p> <p>1 = This message object is the source of an interrupt.</p> <p>0 = This message object is not the source of an interrupt.</p> |

### Message Valid Register 1 (CAN\_MVLD1)

These registers hold the **MsgVal** bits of the 32 Message Objects. By reading the **MsgVal** bits, the application software can check which Message Object is valid. The **MsgVal** bit of a specific Message Object can be set/reset by the application software via the IFn Message Interface Registers.

| Register  | Offset       | R/W | Description              | Reset Value |
|-----------|--------------|-----|--------------------------|-------------|
| CAN_MVLD1 | CAN_BA+0x160 | R   | Message Valid Register 1 | 0x0000_0000 |

|              |    |    |    |    |    |    |    |
|--------------|----|----|----|----|----|----|----|
| 31           | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved     |    |    |    |    |    |    |    |
| 23           | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved     |    |    |    |    |    |    |    |
| 15           | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| MsgVal[16:9] |    |    |    |    |    |    |    |
| 7            | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| MsgVal[8:1]  |    |    |    |    |    |    |    |

| Bits    | Description  |
|---------|--|
| [31:16] | <b>Reserved</b><br><b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.  |
| [15:0]  | <b>MsgVal[16:1]</b><br><b>Message Valid Bits 16-1 (of all Message Objects) (Read Only)</b><br>1 = This Message Object is configured and should be considered by the Message Handler.<br>0 = This Message Object is ignored by the Message Handler.<br>Ex. CAN_MVLD1[0] means Message object No.1 is valid or not. If CAN_MVLD1[0] is set, message object No.1 is configured. |

**Message Valid Register 2 (CAN\_MVLD2)**

| Register  | Offset       | R/W | Description              | Reset Value |
|-----------|--------------|-----|--------------------------|-------------|
| CAN_MVLD2 | CAN_BA+0x164 | R   | Message Valid Register 2 | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| Reserved      |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Reserved      |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| MsgVal[32:25] |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| MsgVal[24:17] |    |    |    |    |    |    |    |

| Bits    | Description  |
|---------|--|
| [31:16] | <p><b>Reserved</b></p> <p><b>Reserved</b></p> <p>There are reserved bits. These bits are always read as '0' and must always be written with '0'.</p>   |
| [15:0]  | <p><b>Message Valid Bits 32-17 (of all Message Objects) (Read only)</b></p> <p>1 = This Message Object is configured and should be considered by the Message Handler.</p> <p>0 = This Message Object is ignored by the Message Handler.</p> <p>Ex.CAN_MVLD2[15] means Message object No.32 is valid or not. If CAN_MVLD2[15] is set, message object No.32 is configured.</p> |

### Wake-Up Enable Register (CAN\_WU\_EN)

| Register  | Offset       | R/W | Description             | Reset Value |
|-----------|--------------|-----|-------------------------|-------------|
| CAN_WU_EN | CAN_BA+0x168 | R/W | Wake-Up Enable Register | 0x0000_0000 |

|          |    |    |    |    |    |    |          |
|----------|----|----|----|----|----|----|----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24       |
| Reserved |    |    |    |    |    |    |          |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16       |
| Reserved |    |    |    |    |    |    |          |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8        |
| Reserved |    |    |    |    |    |    |          |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0        |
| Reserved |    |    |    |    |    |    | WAKUP_EN |

| Bits   | Description |  |
|--------|-------------|--|
| [31:1] | Reserved    | <b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.   |
| [0]    | WAKUP_EN    | <b>Wake Up Enable</b><br>1 = The wake-up function Enabled.<br>0 = The wake-up function Disabled.<br><b>Note:</b> User can wake up system when there is a falling edge in the CAN_RXD pin.. |

### Wake-Up Status Register (CAN\_WU\_STATUS)

| Register      | Offset       | R/W | Description             | Reset Value |
|---------------|--------------|-----|-------------------------|-------------|
| CAN_WU_STATUS | CAN_BA+0x16C | R/W | Wake-Up Status Register | 0x0000_0000 |

|          |    |    |    |    |    |    |           |
|----------|----|----|----|----|----|----|-----------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24        |
| Reserved |    |    |    |    |    |    |           |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16        |
| Reserved |    |    |    |    |    |    |           |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8         |
| Reserved |    |    |    |    |    |    |           |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0         |
| Reserved |    |    |    |    |    |    | WAKUP_STS |

| Bits   | Description   |
|--------|---|
| [31:1] | <b>Reserved</b><br><b>Reserved</b><br>There are reserved bits. These bits are always read as '0' and must always be written with '0'.         |
| [0]    | <b>Wake Up Status</b><br>1 = Wake-up event occurred.<br>0 = No wake-up event occurred.<br><b>Note:</b> The bit can be cleared by writing '0'. |

## 23 12-BIT ANALOG-TO-DIGITAL CONVERTER

### 23.1 Overview

The NuMicro™ NM15xx Series contains two 12-bit successive approximation analog-to-digital converters (SAR A/D converter) with 16 input channels. The two A/D converters ADCA and ADCB (EADC0 and EADC1) can be sampled with Simultaneous or Single Sampling mode. The A/D converters can be started by software, PWM triggers, timer0~3 overflow pulse triggers, ADINT0, ADINT1 interrupt EOC pulse trigger and external STADC pin input signal.

**Note:** The analog input port pins must be configured as input type before the ADC function is enabled.

### 23.2 Features

- Analog input voltage range: 0~Vref (Max to 5.0V).
- 12-bit resolution and 10-bit accuracy is guaranteed.
- Up to 16 single-end analog input channels.
- Two SAR ADC converters.
- Four ADC interrupts with individual interrupt vector addresses.
- Maximum ADC clock frequency is use 16MHz.
- Up to 1.6M SPS conversion rate, each of ADC converter conversion time less than 1.25μs.
- Two operating modes
  - ◆ Single sampling mode: two ADC converters run at normal operation.
  - ◆ Simultaneous sampling mode: Allow two ADC converters can be sampled simultaneously.
- An A/D conversion can be started by:
  - ◆ Writing 1 to ADSTx bit ( x = 0~15) through software
  - ◆ External pin STADC
  - ◆ Timer0~3 overflow pulse triggers
  - ◆ ADINT0, ADINT1 interrupt EOC pulse triggers
  - ◆ PWM triggers
- Conversion results are held in 16 data registers with valid and overrun indicators.
- SAMPLEA0~7 ADC control logic modules, each of them is configurable for ADCA converter channel AINA0~7 and trigger source.
- SAMPLEB0~7 ADC control logic modules, each of them is configurable for ADCB converter channel AINB0~7 and trigger source.
- Channel AINA0 supports 2 input sources: external analog voltage and internal OP0 Amplifier output voltage.
- Channel AINB0 supports 2 input sources: external analog voltage and internal OP1 Amplifier output voltage.
- Channel AINA7 supports 4 input sources: external analog voltage, internal fixed band-gap voltage, internal temperature sensor output, and analog ground.

### 23.3 Block Diagram

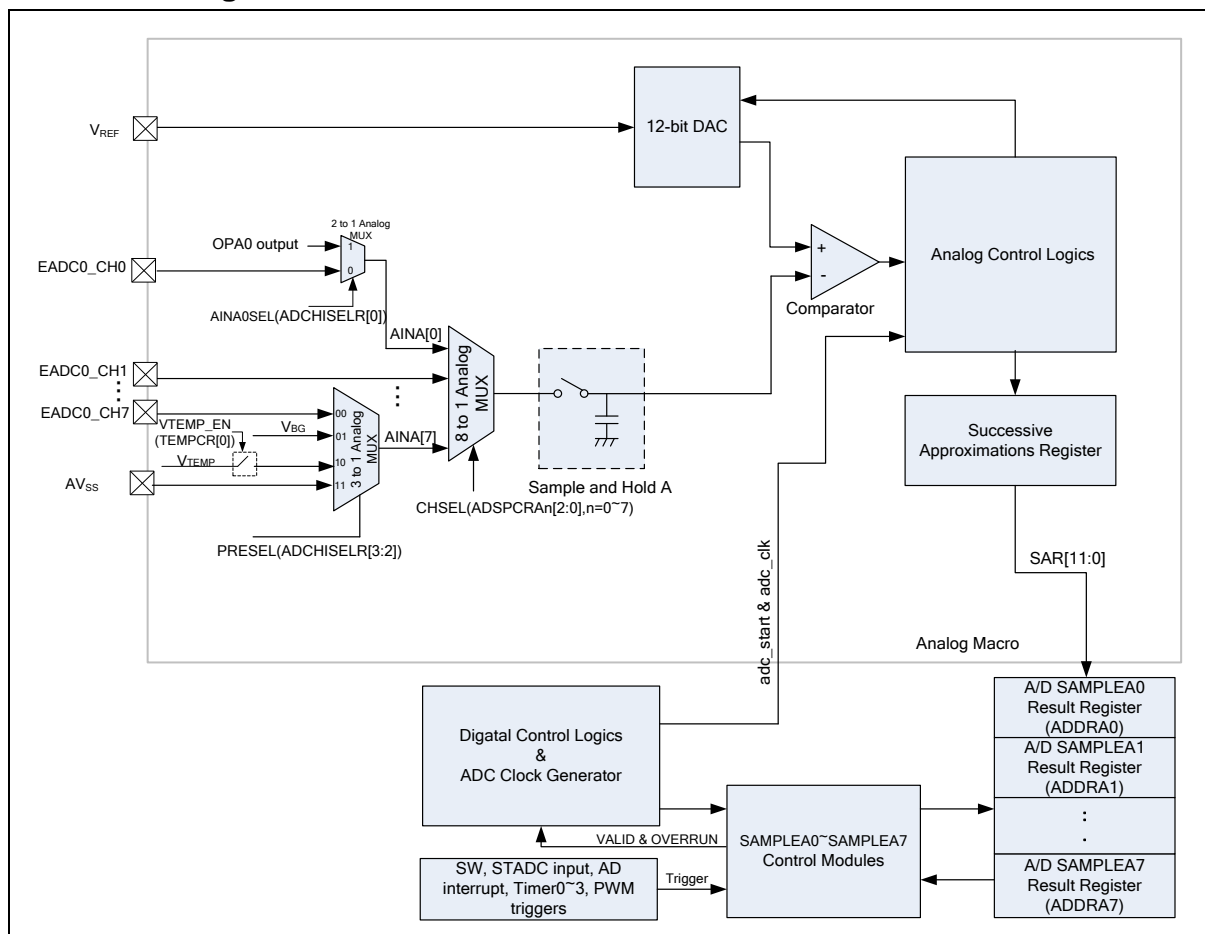


Figure 23-1 ADCA Converter Block Diagram



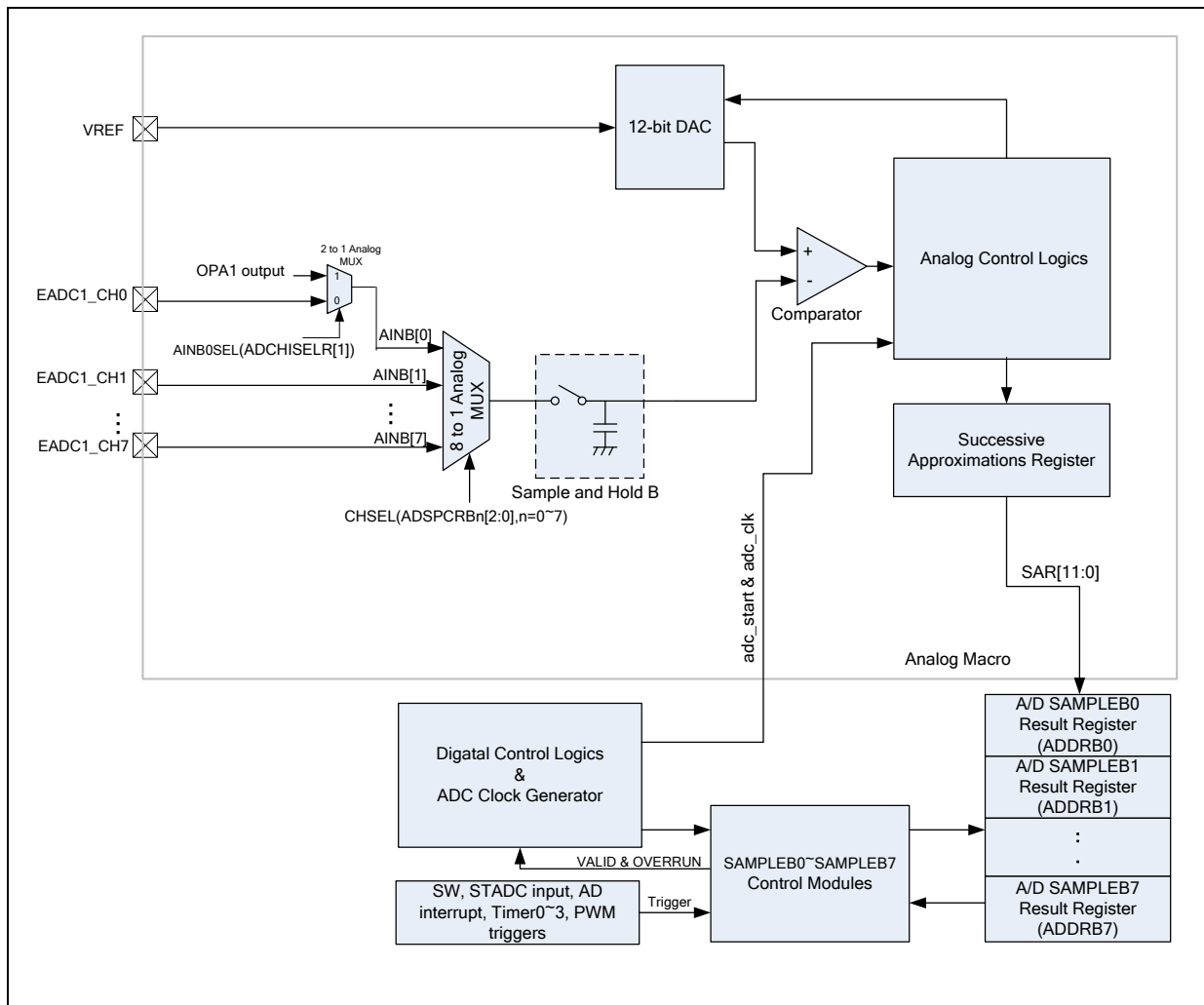


Figure 23-2 ADCB Converter Block Diagram

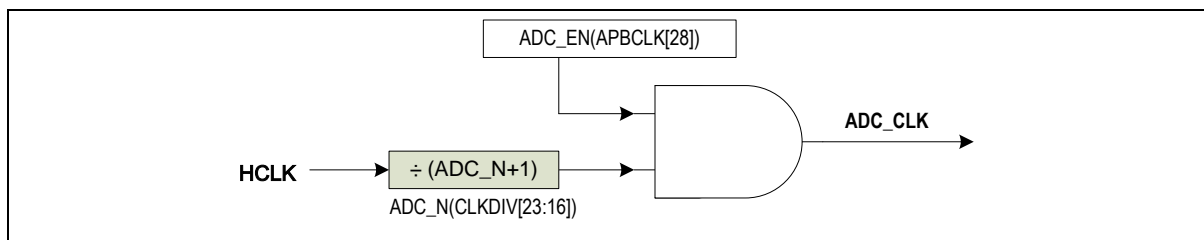


Figure 23-3 ADC Clock Control

### 23.4 Operation Procedure

There are two ADC converters, and each ADC converter consists of an eight-channel input select function and eight SAMPLE control modules, the A/D converter operates by successive approximation with 12-bit resolution.

The A/D operation is based on SAMPLEA0~7(ADCA converter) and SAMPLEB0~7(ADCB converter) control logic modules, and each of them has its configuration to decide which trigger source to start the conversion and which channel to convert. Different SAMPLE modules can be configured for the same channel, trigger source. It provides user a flexible means to get the over-sampling results.

The ADC conversion trigger sources are listed below:

- Writing 1 to ADSTx bit ( x = 0~15) by software
- External pin STADC
- Timer0~3 overflow pulse triggers
- ADINT0, ADINT1 ADC interrupt EOC pulse triggers
- PWM triggers

The ADINT0, ADINT1 interrupt pulses are generated whenever the specific SAMPLE A/D EOC (End of conversion) pulse is generated. The ADINT0, 1 interrupt pulse triggers can be fed back to trigger another A/D conversion, and is useful if a continuous scan conversion is needed.

#### 23.4.1 ADC Clock Generator

The maximum sampling rate is up to 800 kHz and the conversion time is less than 1.25μs. It needs 20 ADC clocks to complete an A/D conversion. The ADC engine clock source is from HCLK clock, the ADC clock frequency is divided by an 8-bit pre-scalar with the formula:

The ADC clock frequency = (HCLK) / (ADC\_N+1);  
where the 8-bit ADC\_N is located in register CLKDIV[23:16].

In generally, software can set ADC\_N to get 16 MHz or slightly less.

#### 23.4.2 ADC Single Sampling Mode

When a ADC conversion is performed on the SAMPLEx specified single channel, the operations are as follows:

- A/D conversion is started when the ADSTx bit in ADSSTR is set to 1 by software or other trigger inputs.
- When A/D conversion is finished, the 12-bit result is stored in the ADC data register ADDR<sub>x</sub> corresponding to the SAMPLE<sub>x</sub>.
- On completion of conversion, the ADF<sub>n</sub> bit in ADSR is set to 1 and ADC interrupt (ADINT<sub>n</sub>) is requested if the ADIE<sub>n</sub> bit is set to 1.
- The ADSTx bit remains 1 during A/D conversion. When A/D conversion ends, the ADSTx bit is automatically cleared to 0 and the A/D converter will do another pending conversion.

**Note:** If software or other trigger enables more than one channel in single or simultaneous sampling mode, the SAMPLE specified channel with highest priority is converted and other enabled channels will be pended.

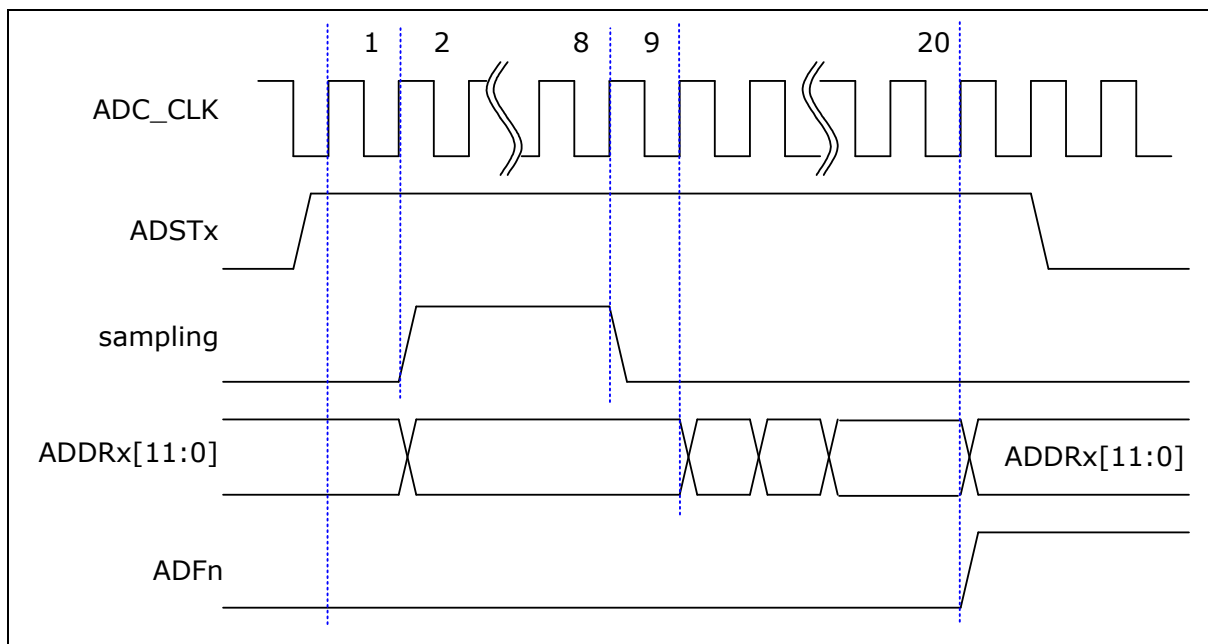


Figure 23–4 Single Sampling Mode Conversion Timing Diagram

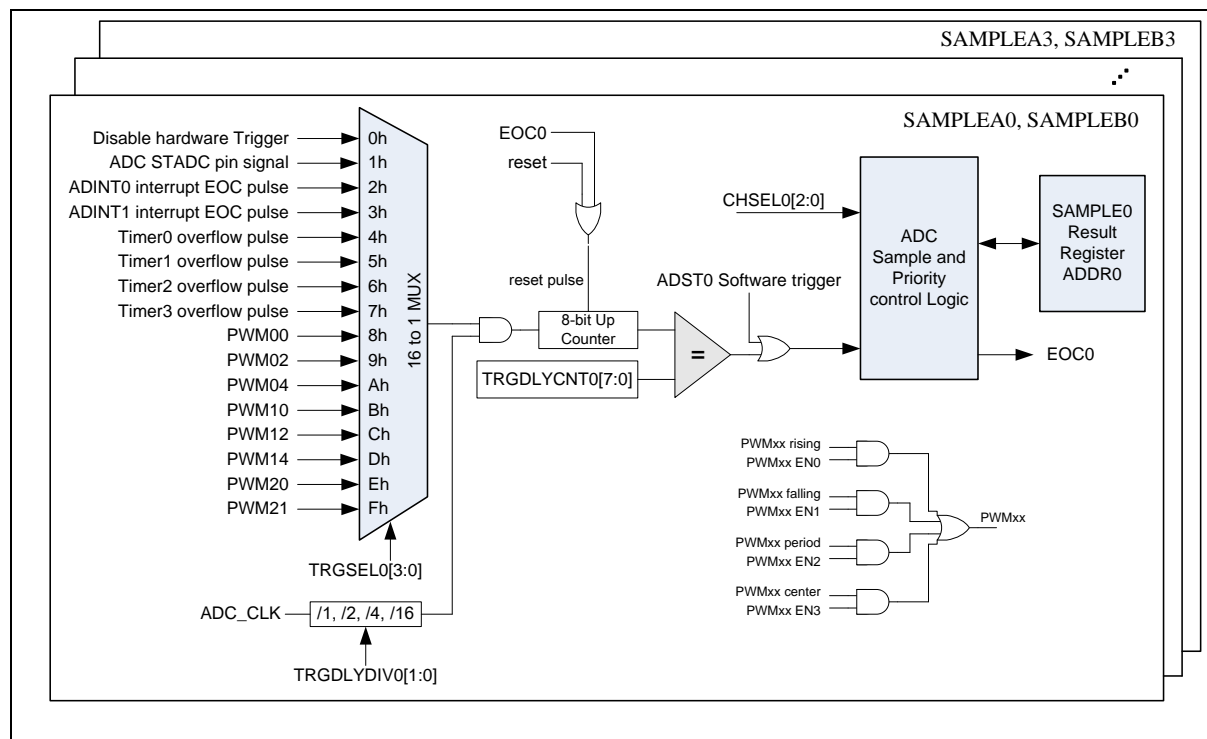


Figure 23-5 SAMPLEA0~3 and SAMPLEB0~3 Control Block Diagram

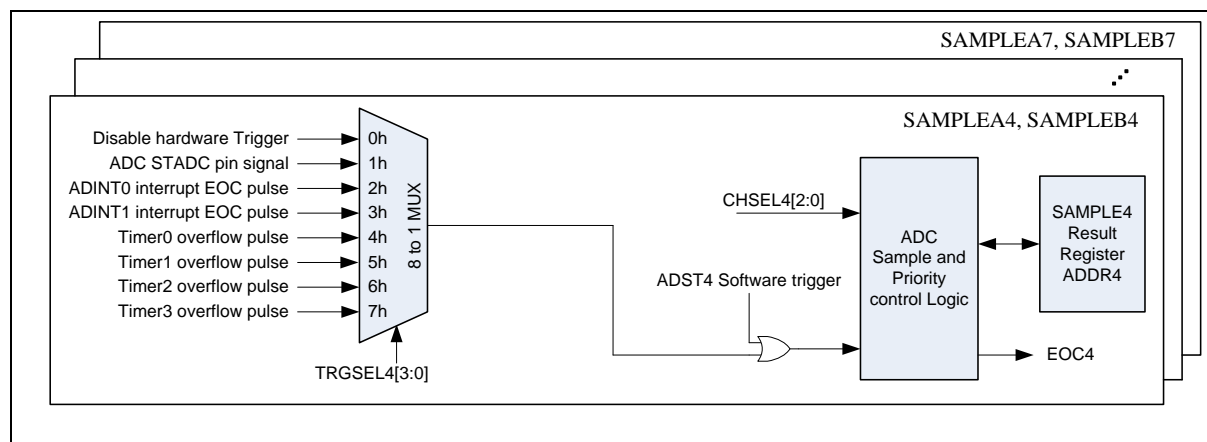


Figure 23-6 SAMPLEA4~7 and SAMPLEB4~7 Control Block Diagram

### 23.4.3 ADC Simultaneous Sampling Mode

The NuMicro™ NM15XX has two ADCs that allow two different ADC channels to be simultaneously sampled. The priority rules are same as the single sampling mode.

The same numbered of SAMPLEA and the SAMPLEB are coupled together. For example, SAMPLEA0 and SAMPLEB0 are coupled when simultaneous sampling mode enable bit SIMUSEL0 bit = 1.

The ADC simultaneous sampling mode conversion operations are described below:

1. Only SAMPLEA trigger can start a pair of conversions.
2. SAMPLEA assign an A-channel and SAMPLEB also assign a B-channel in simultaneous sampling mode. The SAMPLEB specified trigger will be ignored.

- The A-channel conversion result is stored in specific SAMPLEA ADDRA register, and the conversion result of the B-channel is placed in the same numbered SAMPLEB ADDRb register.

For example:

If SIMUSEL2 bit = 1 and SAMPLEA2 is configured to sample AINA[4] (CHSELA2 = 4), it defines the SAMPLEA2 and same numbered SAMPLEB2 are coupled at simultaneous sampling conversion mode, if SAMPLEB2 is configured to sample AINB[3] (CHSELB2 = 3), the pair of ADC conversion channels are AINA[4] and AINA[3].

After a channel pair (AINA[4], AINB[3]) of ADC conversion completes, the results of those two ADC conversions will be placed in registers ADDRA2 and ADDRb2.

#### 23.4.4 ADC Conversion Priority

There are two priority groups of converter for determining the conversion order when multiple SAMPLE trigger flags are set at the same time. SAMPLEA0~7 priority group is for ADCA converter, and SAMPLEB0~7 priority group is for ADCB converter.

The SAMPLE with lower number has higher priority than the higher number SAMPLE, if two SAMPLEs are triggered at the same time; the SAMPLE with lower number will start to convert ADC first.

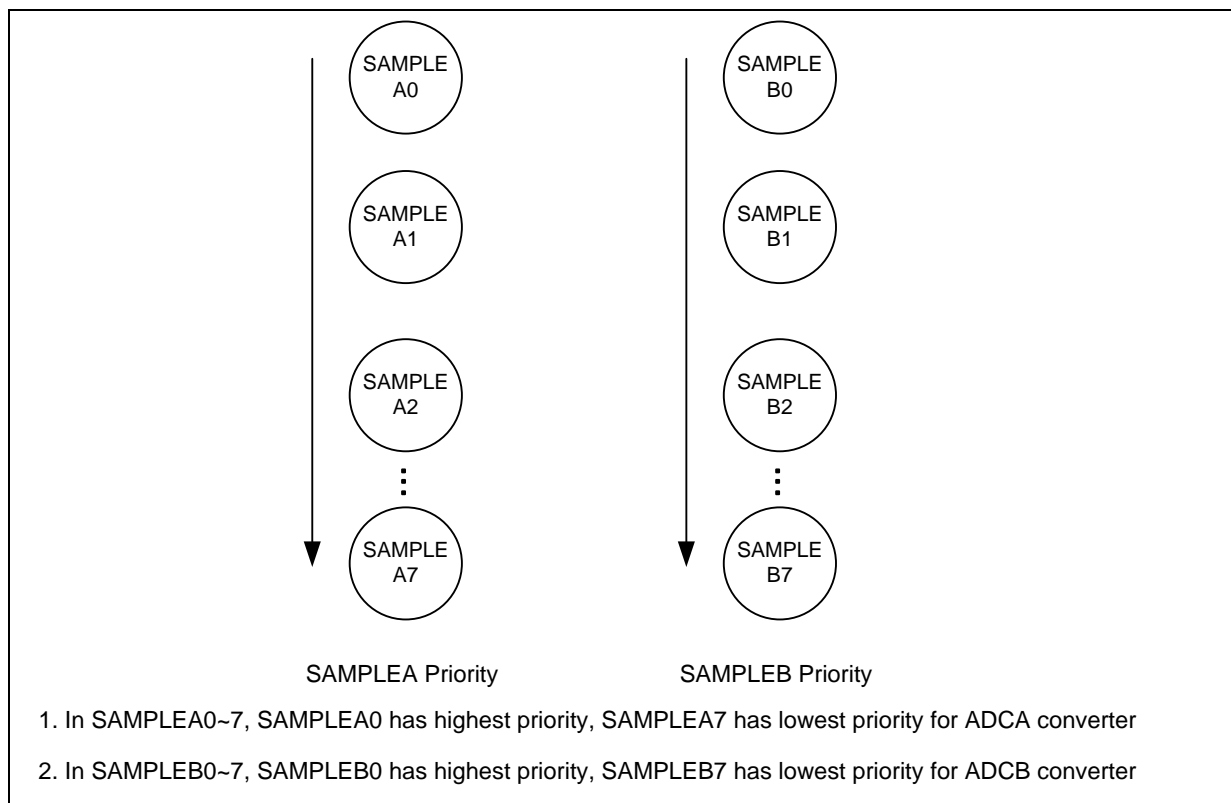


Figure 23-7 SAMPLE Conversion Priority Arbitrator Diagram

#### 23.4.5 ADC Start Synchronous with PWM

Besides starting ADC conversion by software, ADCINT0, 1 interrupt pulse, and external pin STADC, this device has a new feature to allow PWM channels to trigger the ADC start. User may configure the PWM trigger types: rising, falling PWM edge or center point of PWM (Center-aligned mode only) to trigger ADC start. The device also allows user to configure the amount of delay prior to ADC start after hardware detected the PWM edge. The figure below shows the programmable delay time for PWM-triggered ADC to start conversion.

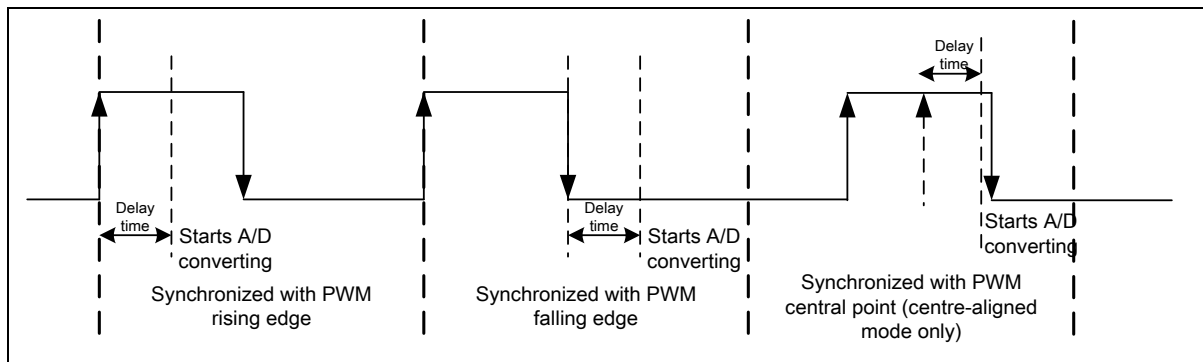


Figure 23–8 PWM-triggered ADC Start

### 23.4.6 ADC SAMPLE End of Conversion Interrupt Operation

There are 4 ADC interrupts ADINT0~3, and each of these interrupts has its own interrupt vector address and can be configured to select a specific SAMPLE EOC pulse (SAMPLEA0~7, SAMPLEB0~7 End of conversion pulses) as its interrupt trigger source. Each of them has its own interrupt overwritten flag ADFOVn(ADIFOVR[n], n=0~3). This flag set to 1 if the corresponding interrupt is overwritten to 1.

When ADIE0(ADCR[2]) = 1 and ADINT0SRCTL[7:0]=0xFF, all of SAMPLEA EOC pulses can cause an ADINT0 interrupt.

If ADIE1(ADCR[3]) = 1 and ADINT1SRCTL[15:8]=0xFF, all of SAMPLEB EOC pulses also can cause an ADINT1 interrupt.

The ADINT0, ADINT1 interrupt pulses are generated whenever the specific SAMPLE A/D EOC (End of conversion) pulse is generated. It also can be the SAMPLE conversion trigger sources, and user can use it to do the ADC continuous scan conversion.

Example for "Continuous scan":

Step1. If ADC SAMPLEA2's EOCA2 pulse is selected as ADINT0 interrupt trigger (ADINT0SRCTL[7:0] = 0x04) and ADINT0 is selected as SAMPLEA0, B1, A2 hardware conversion trigger.

Step2. Set software trigger ADST(ADSSTR[2]) bit to 1 to start a SAMPLEA2 ADC conversion, after the conversion completes, it generates an EOCA2 pulse signal and ADINT0 interrupt pulse at end of SAMPLEA2 ADC conversion, ADINT0 interrupt pulse will trigger the SAMPLEA0, B1, A2 to start the ADC conversions.

Step3. ADINT0 interrupt pulse repeats to trigger SAMPLEA0, B1, A2 ADC conversions automatically.

Step4. Clear TRGSEL(ADSPCRA2[7:4]) to 0 to disable SAMPLEA2 module's ADINT0 interrupt pulse hardware trigger, if needs to stop the continuous scan.

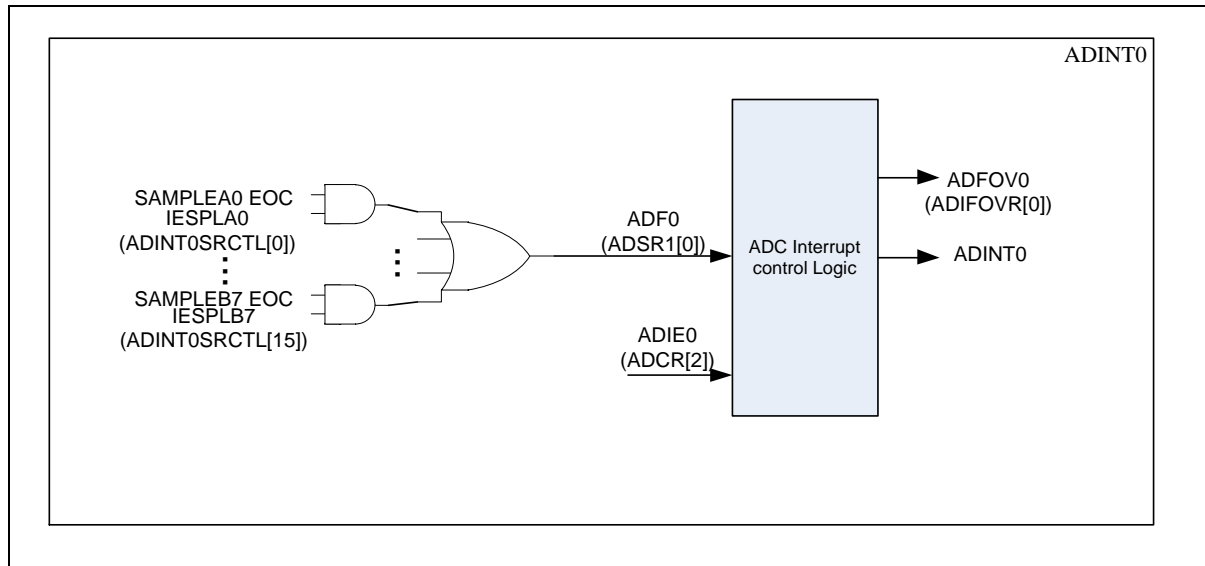


Figure 23-9 Specific SAMPLE A/D EOC Signal for ADINT0 Interrupt

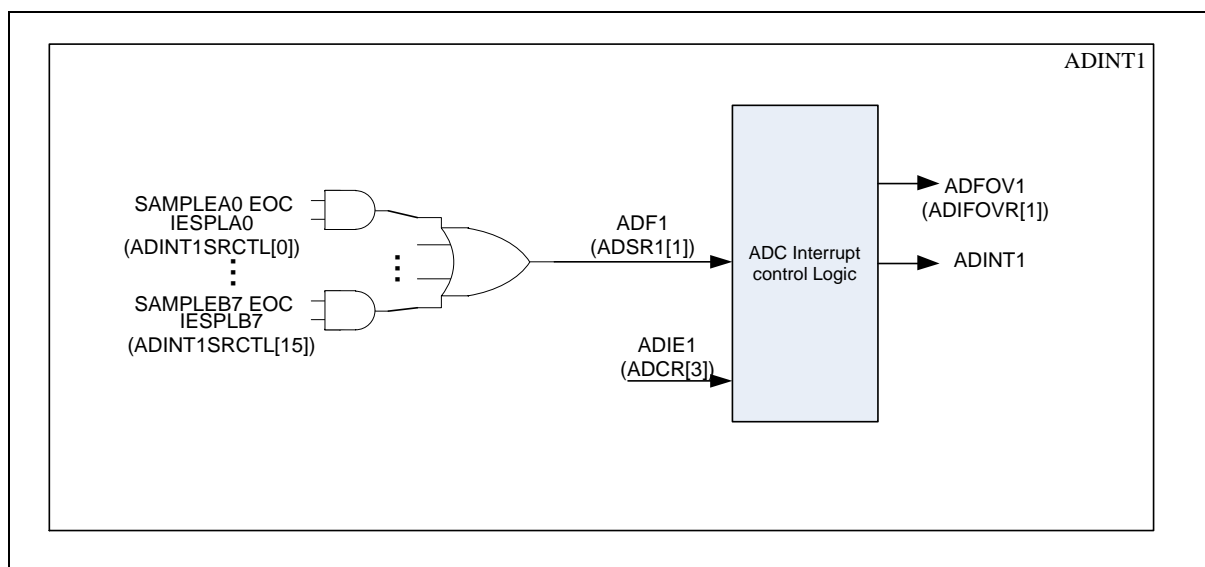


Figure 23-10 Specific SAMPLE A/D EOC Signal for ADINT1 Interrupt



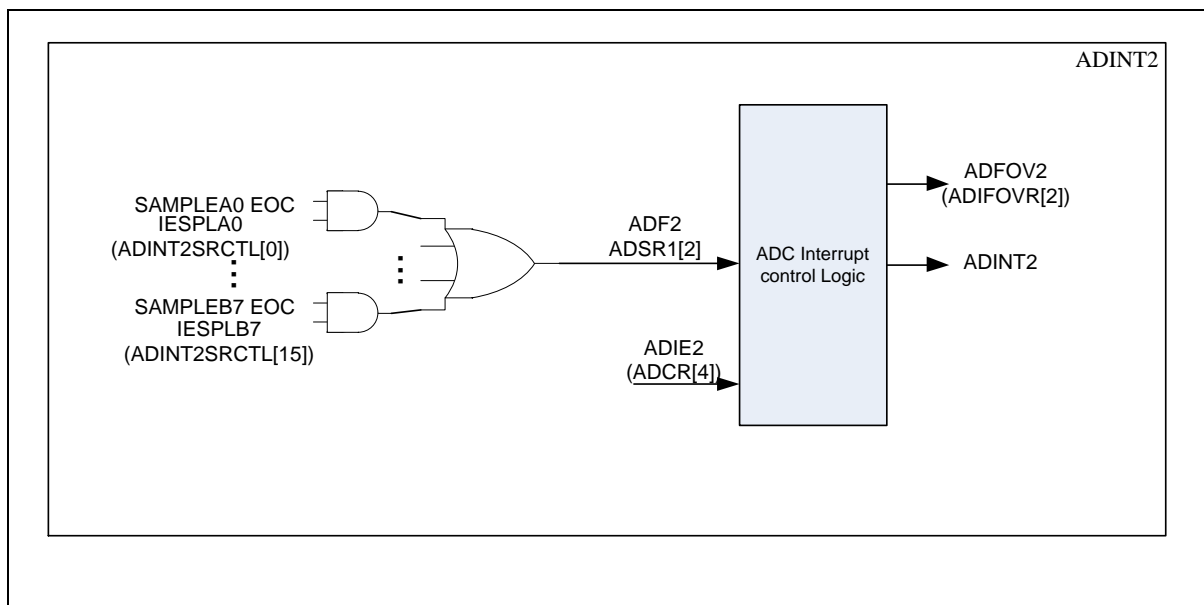


Figure 23-11 Specific SAMPLE A/D EOC Signal for ADINT2 Interrupt

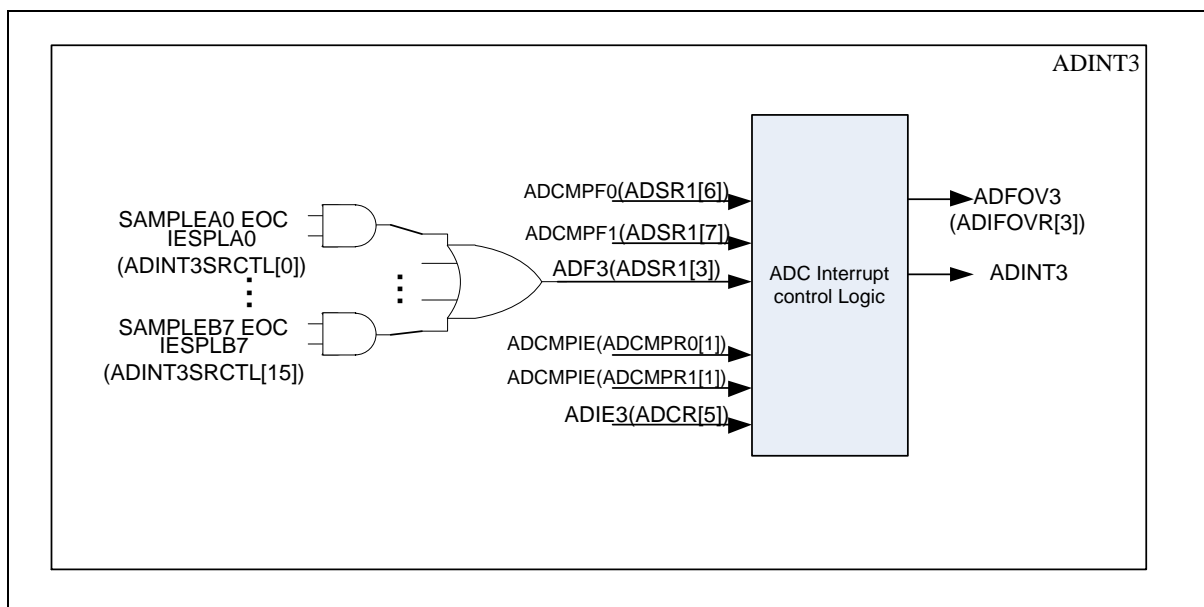


Figure 23-12 Specific SAMPLE A/D EOC Signal for ADINT3 Interrupt

### 23.4.7 Input Sampling and A/D Conversion Time

The A/D converter samples the analog input when A/D conversion start delay time ( $T_d$ ) has passed after ADST bit in ADSSTR is set to 1, and then starts conversion. Since the ADC clock is generated by PCLK divided by (N+1), the maximum delay time from APB write to A/D start sampling analog input time is 2N PCLKs. The start delay time is shown below:

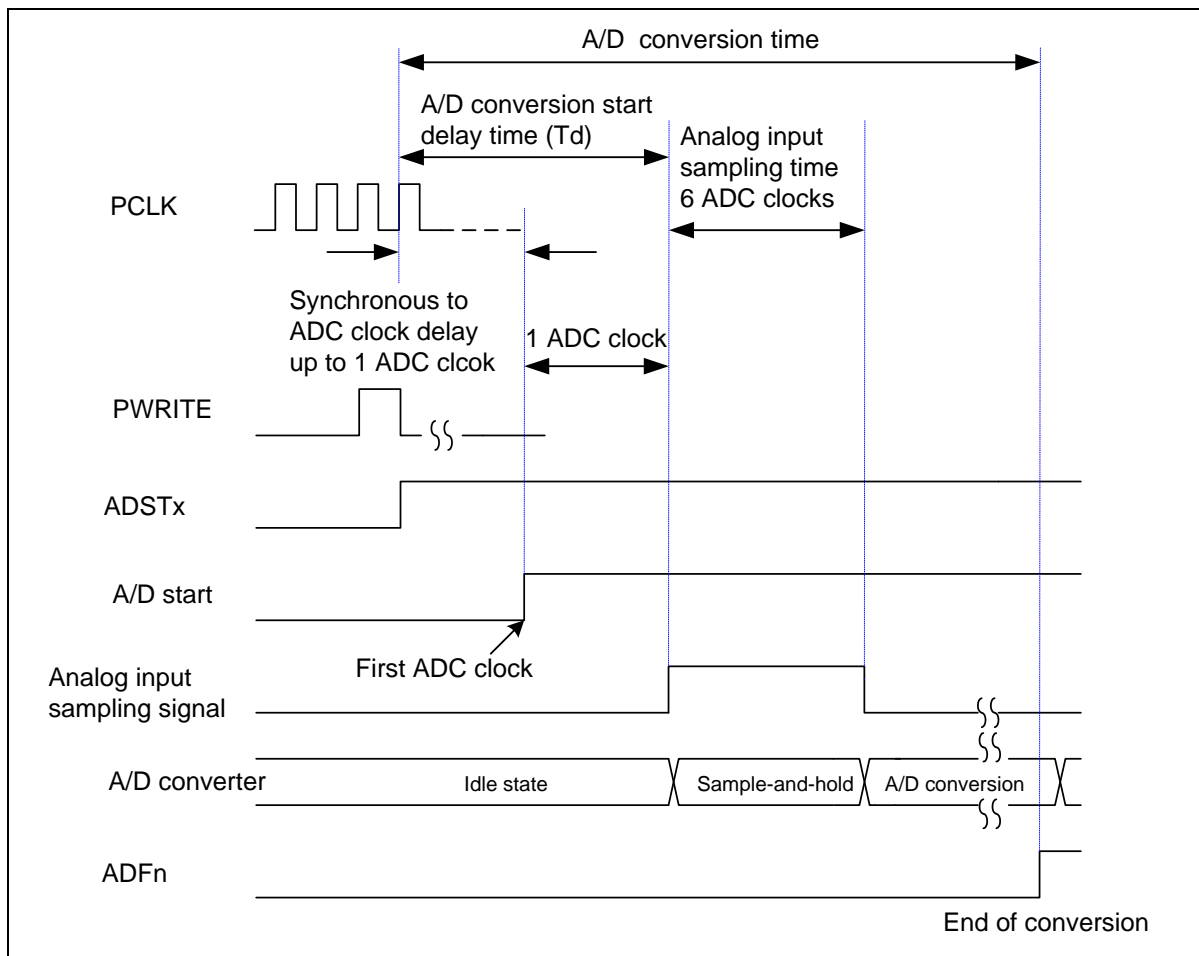


Figure 23-13 Conversion Start Delay Timing Diagram

A/D conversion can be triggered by external pin STADC request. Setting the TRGSEL[x:0] bits to 01h is to select external trigger input from the STADC pin. An 8-bit sampling counter is used to deglitch. If edge trigger condition is selected, the high and low state must be kept at least 4 HCLKs. The pulse that is shorter than this specification will be ignored.

### 23.4.8 A/D Extend Sampling Time

When A/D operation at high ADC clock rate, the sampling time of analog input voltage may not enough if the analog channel has heavy loading to cause fully charge time is longer. SW can set A/D extend sampling time by writing ADEST field in ADTCR register. The A/D extend sampling time is present between A/D controller judge which channel to be converting and A/D start to conversion. The range of extend sampling time is from 0 ~255 ADC clock. The extend sampling time is shown below.

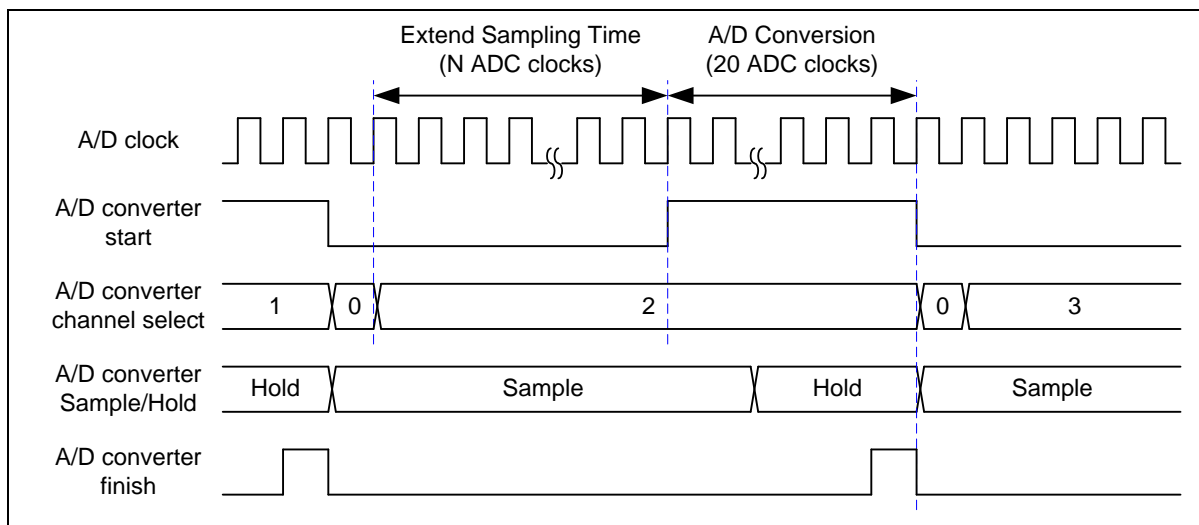


Figure 23-14 A/D Extend Sampling Timing Diagram

### 23.4.9 Conversion Result Monitored by Compare Mode

The NuMicro™ NM15XX controller provides two sets of compare register ADCMPR0 and 1 to monitor maximum two specified SAMPLEA0~3, SAMPLEB0~3 conversion results from A/D conversion module, refer to Figure below. Software can select which SAMPLE module result to be monitored by setting the CMPSMPL(ADCMPRx[5:3]) and CMPCOND bit used to check if the conversion result is less than the specified value or greater than (equal to) value specified in CMPD[11:0]. When the conversion of the SAMPLE specified by CMPSMPL is completed, the comparing action will be triggered once automatically. When the compare result meets the setting, compare match counter will increase 1, when counter value reach the setting of (CMPMATCNT+1) then ADCMPF bit will be set to 1, if ADCMPIE bit is set, an ADINT3 interrupt request is generated. Software can use it to monitor the external analog input pin voltage transition. Detailed logics diagram is shown below.

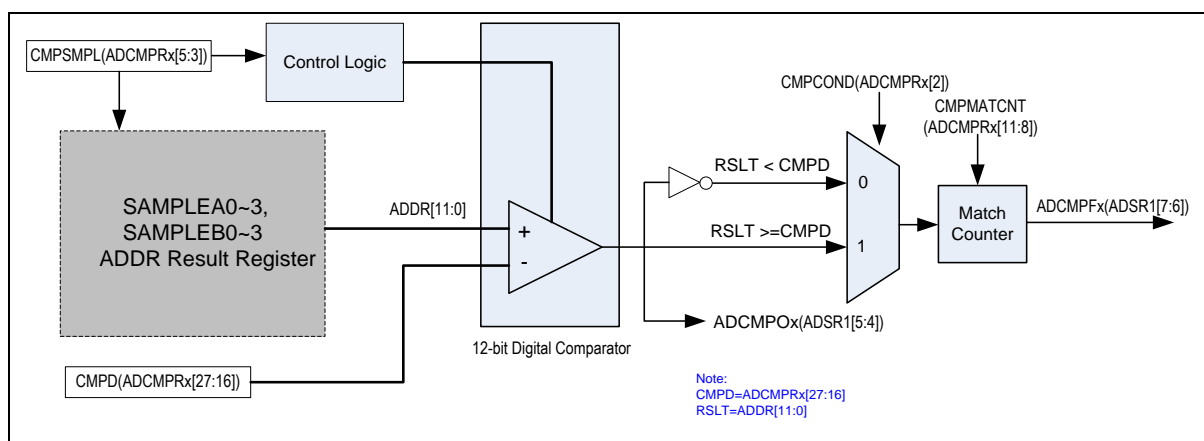


Figure 23-15 A/D Conversion Result Monitor Logics Diagram

### 23.4.10 Interrupt Sources

The A/D converter generates a conversion and ADFn flag in the ADSR1 register upon the end of specific SAMPLE A/D conversion. If the ADIE<sub>n</sub> bit in ADCR is set, the conversion end interrupt request ADINT<sub>n</sub> is generated.

If ADCMPIEx bit is enabled, when A/D conversion result meets setting in ADCMPRx register, monitor

interrupt is generated, ADINT3 will be set also. CPU can to clear ADCMPF<sub>x</sub> and ADF<sub>n</sub> flag to stop interrupt request.

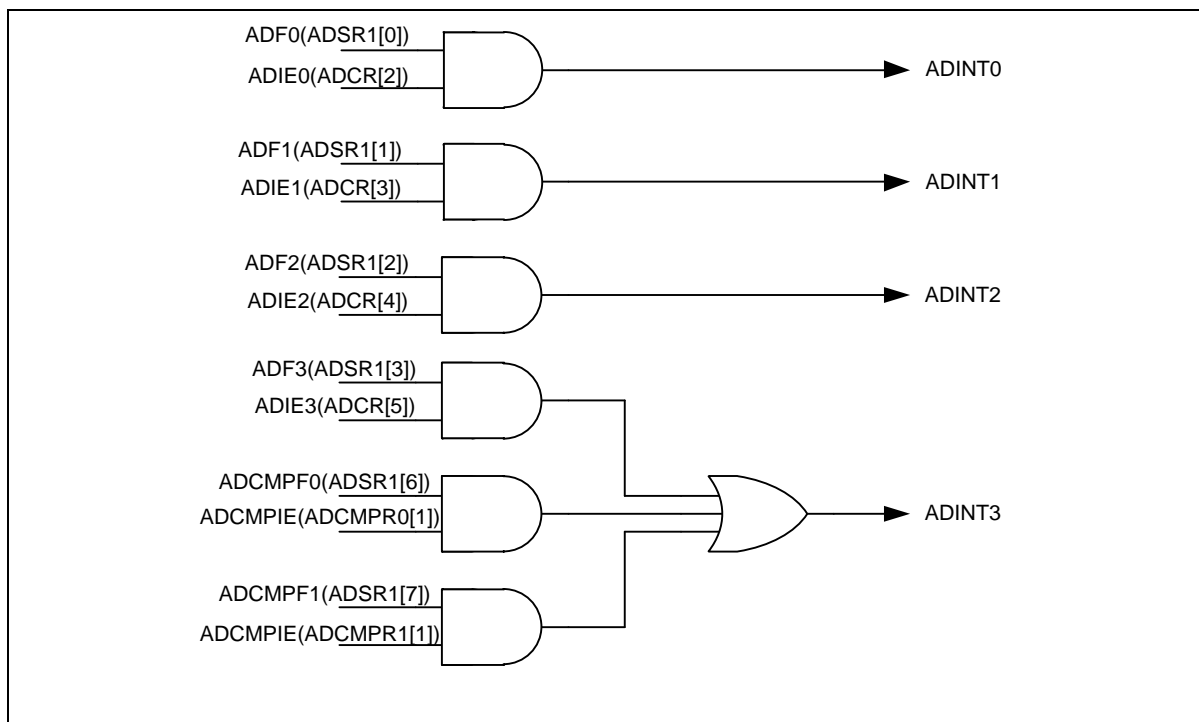


Figure 23–16 A/D Controller Interrupts

### 23.5 Register Map

R: read only, W: write only, R/W: both read and write, C: Only value 0 can be written

| Register                    | Offset      | R/W | Description   | Reset Value |
|-----------------------------|-------------|-----|---|-------------|
| <b>ADC Base Address:</b>    |             |     |   |             |
| <b>ADC_BA = 0x400E_0000</b> |             |     |   |             |
| <b>ADDRA0</b>               | ADC_BA+0x00 | R   | A/D Data Register 0 for SAMPLEA0                      | 0x0000_0000 |
| <b>ADDRA1</b>               | ADC_BA+0x04 | R   | A/D Data Register 1 for SAMPLEA1                      | 0x0000_0000 |
| <b>ADDRA2</b>               | ADC_BA+0x08 | R   | A/D Data Register 2 for SAMPLEA2                      | 0x0000_0000 |
| <b>ADDRA3</b>               | ADC_BA+0x0C | R   | A/D Data Register 3 for SAMPLEA3                      | 0x0000_0000 |
| <b>ADDRA4</b>               | ADC_BA+0x10 | R   | A/D Data Register 4 for SAMPLEA4                      | 0x0000_0000 |
| <b>ADDRA5</b>               | ADC_BA+0x14 | R   | A/D Data Register 5 for SAMPLEA5                      | 0x0000_0000 |
| <b>ADDRA6</b>               | ADC_BA+0x18 | R   | A/D Data Register 6 for SAMPLEA6                      | 0x0000_0000 |
| <b>ADDRA7</b>               | ADC_BA+0x1C | R   | A/D Data Register 7 for SAMPLEA7                      | 0x0000_0000 |
| <b>ADDRB0</b>               | ADC_BA+0x20 | R   | A/D Data Register 8 for SAMPLEB0                      | 0x0000_0000 |
| <b>ADDRB1</b>               | ADC_BA+0x24 | R   | A/D Data Register 9 for SAMPLEB1                      | 0x0000_0000 |
| <b>ADDRB2</b>               | ADC_BA+0x28 | R   | A/D Data Register 10 for SAMPLEB2                     | 0x0000_0000 |
| <b>ADDRB3</b>               | ADC_BA+0x2C | R   | A/D Data Register 11 for SAMPLEB3                     | 0x0000_0000 |
| <b>ADDRB4</b>               | ADC_BA+0x30 | R   | A/D Data Register 12 for SAMPLEB4                     | 0x0000_0000 |
| <b>ADDRB5</b>               | ADC_BA+0x34 | R   | A/D Data Register 13 for SAMPLEB5                     | 0x0000_0000 |
| <b>ADDRB6</b>               | ADC_BA+0x38 | R   | A/D Data Register 14 for SAMPLEB6                     | 0x0000_0000 |
| <b>ADDRB7</b>               | ADC_BA+0x3C | R   | A/D Data Register 15 for SAMPLEB7                     | 0x0000_0000 |
| <b>ADCR</b>                 | ADC_BA+0x40 | R/W | A/D Control Register                                  | 0x0000_0000 |
| <b>ADCHISELR</b>            | ADC_BA+0x44 | R/W | A/D Channel Input Sources Select Register             | 0x0000_0000 |
| <b>ADSSTR</b>               | ADC_BA+0x48 | W   | A/D SAMPLE Software Start Register                    | 0x0000_0000 |
| <b>ADSTPFR</b>              | ADC_BA+0x4C | R   | A/D SAMPLE Start of Conversion Pending Flag Register  | 0x0000_0000 |
| <b>ADIFOVR</b>              | ADC_BA+0x50 | R/W | A/D ADINT3~0 Interrupt Flag Over Run Register         | 0x0000_0000 |
| <b>ADSPOVFR</b>             | ADC_BA+0x54 | R/W | A/D SAMPLE Start of Conversion Over Run Flag Register | 0x0000_0000 |
| <b>ADSPCRA0</b>             | ADC_BA+0x58 | R/W | A/D SAMPLEA0 Control Register                         | 0x0000_0000 |
| <b>ADSPCRA1</b>             | ADC_BA+0x5C | R/W | A/D SAMPLEA1 Control Register                         | 0x0000_0000 |
| <b>ADSPCRA2</b>             | ADC_BA+0x60 | R/W | A/D SAMPLEA2 Control Register                         | 0x0000_0000 |
| <b>ADSPCRA3</b>             | ADC_BA+0x64 | R/W | A/D SAMPLEA3 Control Register                         | 0x0000_0000 |
| <b>ADSPCRA4</b>             | ADC_BA+0x68 | R/W | A/D SAMPLEA4 Control Register                         | 0x0000_0000 |

| Register                    | Offset       | R/W | Description   | Reset Value |
|-----------------------------|--------------|-----|---|-------------|
| <b>ADC Base Address:</b>    |              |     |   |             |
| <b>ADC_BA = 0x400E_0000</b> |              |     |   |             |
| <b>ADSPCRA5</b>             | ADC_BA+0x6C  | R/W | A/D SAMPLEA5 Control Register                           | 0x0000_0000 |
| <b>ADSPCRA6</b>             | ADC_BA+0x70  | R/W | A/D SAMPLEA6 Control Register                           | 0x0000_0000 |
| <b>ADSPCRA7</b>             | ADC_BA+0x74  | R/W | A/D SAMPLEA7 Control Register                           | 0x0000_0000 |
| <b>ADSPCRB0</b>             | ADC_BA+0x78  | R/W | A/D SAMPLEB0 Control Register                           | 0x0000_0000 |
| <b>ADSPCRB1</b>             | ADC_BA+0x7C  | R/W | A/D SAMPLEB1 Control Register                           | 0x0000_0000 |
| <b>ADSPCRB2</b>             | ADC_BA+0x80  | R/W | A/D SAMPLEB2 Control Register                           | 0x0000_0000 |
| <b>ADSPCRB3</b>             | ADC_BA+0x84  | R/W | A/D SAMPLEB3 Control Register                           | 0x0000_0000 |
| <b>ADSPCRB4</b>             | ADC_BA+0x88  | R/W | A/D SAMPLEB4 Control Register                           | 0x0000_0000 |
| <b>ADSPCRB5</b>             | ADC_BA+0x8C  | R/W | A/D SAMPLEB5 Control Register                           | 0x0000_0000 |
| <b>ADSPCRB6</b>             | ADC_BA+0x90  | R/W | A/D SAMPLEB6 Control Register                           | 0x0000_0000 |
| <b>ADSPCRB7</b>             | ADC_BA+0x94  | R/W | A/D SAMPLEB7 Control Register                           | 0x0000_0000 |
| <b>ADSMSELR</b>             | ADC_BA+0xA4  | R/W | A/D SAMPLE Simultaneous Mode Select Register            | 0x0000_0000 |
| <b>ADCMPR0</b>              | ADC_BA+0xA8  | R/W | A/D Result Compare Register 0                           | 0x0000_0000 |
| <b>ADCMPR1</b>              | ADC_BA+0xAC  | R/W | A/D Result Compare Register 1                           | 0x0000_0000 |
| <b>ADSR0</b>                | ADC_BA+0xB0  | R   | A/D Status Register 0                                   | 0x0000_0000 |
| <b>ADSR1</b>                | ADC_BA+0xB4  | R/W | A/D Status Register 1                                   | 0x0000_0000 |
| <b>ADTCR</b>                | ADC_BA+0xB8  | R/W | A/D Timing Control Register                             | 0x0000_0000 |
| <b>ADDRA0B</b>              | ADC_BA+0x100 | R   | A/D double Data Register 0 for SAMPLEA0                 | 0x0000_0000 |
| <b>ADDRA1B</b>              | ADC_BA+0x104 | R   | A/D double Data Register 1 for SAMPLEA1                 | 0x0000_0000 |
| <b>ADDRA2B</b>              | ADC_BA+0x108 | R   | A/D double Data Register 2 for SAMPLEA2                 | 0x0000_0000 |
| <b>ADDRA3B</b>              | ADC_BA+0x10C | R   | A/D double Data Register 3 for SAMPLEA3                 | 0x0000_0000 |
| <b>ADDRB0B</b>              | ADC_BA+0x120 | R   | A/D double Data Register 0 for SAMPLEB0                 | 0x0000_0000 |
| <b>ADDRB1B</b>              | ADC_BA+0x124 | R   | A/D double Data Register 1 for SAMPLEB1                 | 0x0000_0000 |
| <b>ADDRB2B</b>              | ADC_BA+0x128 | R   | A/D double Data Register 2 for SAMPLEB2                 | 0x0000_0000 |
| <b>ADDRB3B</b>              | ADC_BA+0x12C | R   | A/D double Data Register 3 for SAMPLEB3                 | 0x0000_0000 |
| <b>ADDBM</b>                | ADC_BA+0x130 | R/W | A/D Timing Control Register                             | 0x0000_0000 |
| <b>ADINT0SRCTL</b>          | ADC_BA+0x134 | R/W | A/D interrupt source enable control register for ADINT0 | 0x0000_0000 |
| <b>ADINT1SRCTL</b>          | ADC_BA+0x138 | R/W | A/D interrupt source enable control register for ADINT1 | 0x0000_0000 |
| <b>ADINT2SRCTL</b>          | ADC_BA+0x13C | R/W | A/D interrupt source enable control register for ADINT2 | 0x0000_0000 |

| Register                    | Offset       | R/W | Description   | Reset Value |
|-----------------------------|--------------|-----|---|-------------|
| <b>ADC Base Address:</b>    |              |     |   |             |
| <b>ADC_BA = 0x400E_0000</b> |              |     |   |             |
| <b>ADINT3SRCTL</b>          | ADC_BA+0x140 | R/W | A/D interrupt source enable control register for ADINT3 | 0x0000_0000 |
| <b>SMPA0TRGEN</b>           | ADC_BA+0x144 | R/W | A/D trigger condition for SAMPLEA0                      | 0x0000_0000 |
| <b>SMPA1TRGEN</b>           | ADC_BA+0x148 | R/W | A/D trigger condition for SAMPLEA1                      | 0x0000_0000 |
| <b>SMPA2TRGEN</b>           | ADC_BA+0x14C | R/W | A/D trigger condition for SAMPLEA2                      | 0x0000_0000 |
| <b>SMPA3TRGEN</b>           | ADC_BA+0x150 | R/W | A/D trigger condition for SAMPLEA3                      | 0x0000_0000 |
| <b>SMPB0TRGEN</b>           | ADC_BA+0x154 | R/W | A/D trigger condition for SAMPLEB0                      | 0x0000_0000 |
| <b>SMPB1TRGEN</b>           | ADC_BA+0x158 | R/W | A/D trigger condition for SAMPLEB1                      | 0x0000_0000 |
| <b>SMPB2TRGEN</b>           | ADC_BA+0x15C | R/W | A/D trigger condition for SAMPLEB2                      | 0x0000_0000 |
| <b>SMPB3TRGEN</b>           | ADC_BA+0x160 | R/W | A/D trigger condition for SAMPLEB3                      | 0x0000_0000 |

### 23.6 Register Description

#### A/D Data Registers (ADDRA0~7, ADDR0~7)

| Register | Offset      | R/W | Description                       | Reset Value |
|----------|-------------|-----|-----------------------------------|-------------|
| ADDRA0   | ADC_BA+0x00 | R   | A/D Data Register 0 for SAMPLEA0  | 0x0000_0000 |
| ADDRA1   | ADC_BA+0x04 | R   | A/D Data Register 1 for SAMPLEA1  | 0x0000_0000 |
| ADDRA2   | ADC_BA+0x08 | R   | A/D Data Register 2 for SAMPLEA2  | 0x0000_0000 |
| ADDRA3   | ADC_BA+0x0C | R   | A/D Data Register 3 for SAMPLEA3  | 0x0000_0000 |
| ADDRA4   | ADC_BA+0x10 | R   | A/D Data Register 4 for SAMPLEA4  | 0x0000_0000 |
| ADDRA5   | ADC_BA+0x14 | R   | A/D Data Register 5 for SAMPLEA5  | 0x0000_0000 |
| ADDRA6   | ADC_BA+0x18 | R   | A/D Data Register 6 for SAMPLEA6  | 0x0000_0000 |
| ADDRA7   | ADC_BA+0x1C | R   | A/D Data Register 7 for SAMPLEA7  | 0x0000_0000 |
| ADDRB0   | ADC_BA+0x20 | R   | A/D Data Register 8 for SAMPLEB0  | 0x0000_0000 |
| ADDRB1   | ADC_BA+0x24 | R   | A/D Data Register 9 for SAMPLEB1  | 0x0000_0000 |
| ADDRB2   | ADC_BA+0x28 | R   | A/D Data Register 10 for SAMPLEB2 | 0x0000_0000 |
| ADDRB3   | ADC_BA+0x2C | R   | A/D Data Register 11 for SAMPLEB3 | 0x0000_0000 |
| ADDRB4   | ADC_BA+0x30 | R   | A/D Data Register 12 for SAMPLEB4 | 0x0000_0000 |
| ADDRB5   | ADC_BA+0x34 | R   | A/D Data Register 13 for SAMPLEB5 | 0x0000_0000 |
| ADDRB6   | ADC_BA+0x38 | R   | A/D Data Register 14 for SAMPLEB6 | 0x0000_0000 |
| ADDRB7   | ADC_BA+0x3C | R   | A/D Data Register 15 for SAMPLEB7 | 0x0000_0000 |

|           |    |    |    |             |    |       |         |
|-----------|----|----|----|-------------|----|-------|---------|
| 31        | 30 | 29 | 28 | 27          | 26 | 25    | 24      |
| -         |    |    |    |             |    |       |         |
| 23        | 22 | 21 | 20 | 19          | 18 | 17    | 16      |
| -         |    |    |    |             |    | VALID | OVERRUN |
| 15        | 14 | 13 | 12 | 11          | 10 | 9     | 8       |
| -         |    |    |    | RSLT [11:8] |    |       |         |
| 7         | 6  | 5  | 4  | 3           | 2  | 1     | 0       |
| RSLT[7:0] |    |    |    |             |    |       |         |



| Bits    | Description    |   |
|---------|----------------|---|
| [31:18] | -              | <b>Reserved.</b>  |
| [17]    | <b>VALID</b>   | <b>Valid Flag</b><br>1 = Data in RSLT[11:0] bits is valid.<br>0 = Data in RSLT[11:0] bits is not valid.<br>This bit is set to 1 when corresponding SAMPLE channel analog input conversion is completed and cleared by hardware after ADDR register is read.   |
| [16]    | <b>OVERRUN</b> | <b>Over Run Flag</b><br>1 = Data in RSLT[11:0] is overwritten.<br>0 = Data in RSLT[11:0] is the recent conversion result.<br>If converted data in RSLT[11:0] has not been read before new conversion result is loaded to this register, OVERRUN is set to 1. It is cleared by hardware after ADDR register is read. |
| [15:12] | -              | <b>Reserved.</b>  |
| [11:0]  | <b>RSLT</b>    | <b>A/D Conversion Result</b><br>This field contains 12-bit conversion result.   |

**A/D Control Register (ADCR)**

| Register | Offset      | R/W | Description          | Reset Value |
|----------|-------------|-----|----------------------|-------------|
| ADCR     | ADC_BA+0x40 | R/W | A/D Control Register | 0x0000_0000 |

|    |    |       |       |       |       |         |       |
|----|----|-------|-------|-------|-------|---------|-------|
| 31 | 30 | 29    | 28    | 27    | 26    | 25      | 24    |
| -  |    |       |       |       |       |         |       |
| 23 | 22 | 21    | 20    | 19    | 18    | 17      | 16    |
| -  |    |       |       |       |       |         |       |
| 15 | 14 | 13    | 12    | 11    | 10    | 9       | 8     |
| -  |    |       |       |       |       |         |       |
| 7  | 6  | 5     | 4     | 3     | 2     | 1       | 0     |
| -  |    | ADIE3 | ADIE2 | ADIE1 | ADIE0 | ADRESET | AD_EN |

| Bits   | Description |  |
|--------|-------------|--|
| [31:6] | -           | Reserved.  |
| [5]    | ADIE3       | <p><b>Specific SAMPLE A/D ADINT3 Interrupt Enable</b></p> <p>1 = Specific SAMPLE A/D ADINT3 interrupt function Enabled.<br/>0 = Specific SAMPLE A/D ADINT3 interrupt function Disabled.</p> <p>The A/D converter generates a conversion end ADF3 flag in ADSR1 register upon the end of specific SAMPLE A/D conversion. If ADIE3 bit is set then conversion end interrupt request ADINT3 is generated.</p> |
| [4]    | ADIE2       | <p><b>Specific SAMPLE A/D ADINT2 Interrupt Enable</b></p> <p>1 = Specific SAMPLE A/D ADINT2 interrupt function Enabled.<br/>0 = Specific SAMPLE A/D ADINT2 interrupt function Disabled.</p> <p>The A/D converter generates a conversion end ADF2 flag in ADSR1 register upon the end of specific SAMPLE A/D conversion. If ADIE2 bit is set then conversion end interrupt request ADINT2 is generated.</p> |
| [3]    | ADIE1       | <p><b>Specific SAMPLE A/D ADINT1 Interrupt Enable</b></p> <p>1 = Specific SAMPLE A/D ADINT1 interrupt function Enabled.<br/>0 = Specific SAMPLE A/D ADINT1 interrupt function Disabled.</p> <p>The A/D converter generates a conversion end ADF1 flag in ADSR1 register upon the end of specific SAMPLE A/D conversion. If ADIE1 bit is set then conversion end interrupt request ADINT1 is generated.</p> |
| [2]    | ADIE0       | <p><b>Specific SAMPLE A/D ADINT0 Interrupt Enable</b></p> <p>1 = Specific SAMPLE A/D ADINT0 interrupt function Enabled.<br/>0 = Specific SAMPLE A/D ADINT0 interrupt function Disabled.</p> <p>The A/D converter generates a conversion end ADF0 flag in ADSR1 register upon the end of specific SAMPLE A/D conversion. If ADIE0 bit is set then conversion end interrupt request ADINT0 is generated.</p> |

| Bits | Description    |  |
|------|----------------|--|
| [1]  | <b>ADRESET</b> | <p><b>ADCA, ADCB A/D Converter control circuits reset</b></p> <p>0 = Writing 0 has no effect.</p> <p>1 = Writing 1 will cause ADC control circuits reset to initial state, but not change the ADC registers value.</p> <p>ADRESET bit remains 1 during ADC reset, when ADC reset end, the ADRESET bit is automatically cleared to 0.</p> |
| [0]  | <b>AD_EN</b>   | <p><b>A/D Converter Enable</b></p> <p>1 = Enabled.</p> <p>0 = Disabled.</p> <p>Before starting the A/D conversion function, this bit should be set to 1. Clear it to 0 to disable A/D converter analog circuit power consumption.</p>  |

### A/D Channel Input Select Register (ADCHISELR)

| Register  | Offset      | R/W | Description                               | Reset Value |
|-----------|-------------|-----|---|-------------|
| ADCHISELR | ADC_BA+0x44 | R/W | A/D Channel Input Sources Select Register | 0x0000_0000 |

|    |    |    |    |             |    |          |          |
|----|----|----|----|-------------|----|----------|----------|
| 31 | 30 | 29 | 28 | 27          | 26 | 25       | 24       |
| -  |    |    |    |             |    |          |          |
| 23 | 22 | 21 | 20 | 19          | 18 | 17       | 16       |
| -  |    |    |    |             |    |          |          |
| 15 | 14 | 13 | 12 | 11          | 10 | 9        | 8        |
| -  |    |    |    |             |    |          |          |
| 7  | 6  | 5  | 4  | 3           | 2  | 1        | 0        |
| -  |    |    |    | PRESEL[1:0] |    | AINB0SEL | AINA0SEL |

| Bits   | Description |   |
|--------|-------------|---|
| [31:4] | -           | Reserved.   |
| [3:2]  | PRESEL      | <b>A/D Channel AINA[7] Analog Input Selection</b><br>00 = Analog Input Channel AINA7<br>01 = Band-gap (VBG) Analog Input<br>10 = VTEMP Internal Temperature Sensor Analog Input<br>11 = Analog ground |
| [1]    | AINB0SEL    | <b>A/D Channel AINB[0] Analog Input Selection</b><br>1 = OP Amplifier 1 output is selected as the A/D AINB[0] input signal<br>0 = AINB[0] pin P7.0/AINB0 is selected as the A/D AINB[0] input signal  |
| [0]    | AINA0SEL    | <b>A/D Channel AINA[0] Analog Input Selection</b><br>1 = OP Amplifier 0 output is selected as the ADC AINA[0] input signal<br>0 = AINA[0] pin P6.0/AINA0 is selected as the ADC AINA[0] input signal  |

### A/D SAMPLE Software Start Register (ADSSTR)

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| ADSSTR   | ADC_BA+0x48 | W   | A/D SAMPLE Software Start Register | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| ADST[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| ADST[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15:8]  | ADST[15:8]  | <b>A/D SAMPLEB7~0 Software Force to Start ADC Conversion Register</b><br>0 = No effect.<br>1 = Cause an ADC conversion when the priority is given to SAMPLEB. |
| [7:0]   | ADST[7:0]   | <b>A/D SAMPLEA7~0 Software Force to Start ADC Conversion Register</b><br>0 = No effect.<br>1 = Cause an ADC conversion when the priority is given to SAMPLEA. |

**A/D SAMPLE Start of Conversion Pending Flag Register (ADSTPFR)**

| Register | Offset      | R/W | Description  | Reset Value |
|----------|-------------|-----|--|-------------|
| ADSTPFR  | ADC_BA+0x4C | R   | A/D SAMPLE Start of Conversion Pending Flag Register | 0x0000_0000 |

|            |    |    |    |    |    |    |    |
|------------|----|----|----|----|----|----|----|
| 31         | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -          |    |    |    |    |    |    |    |
| 23         | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -          |    |    |    |    |    |    |    |
| 15         | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| STPF[15:8] |    |    |    |    |    |    |    |
| 7          | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| STPF[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:16] | -           | Reserved.  |
| [15:8]  | STPF[15:8]  | <b>A/D SAMPLEB7~0 Start Conversion Pending Flag</b><br>0 = No pending conversion for SAMPLEB<br>1 = SAMPLEB ADC start of conversion is pending.<br>This bit remains 1 during pending state, when the respective ADC conversion is started, the STPFx bit is automatically cleared to 0.          |
| [7:0]   | STPF[7:0]   | <b>A/D SAMPLEA7~0 Start Conversion Pending Flag</b><br>0 = There is no pending conversion for SAMPLEA<br>1 = SAMPLEA ADC start of conversion is pending.<br>This bit remains 1 during pending state, when the respective ADC conversion is started, the STPFx bit is automatically cleared to 0. |

### A/D Interrupt Flag Over Run Register (ADIFOVR)

| Register | Offset      | R/W | Description                                   | Reset Value |
|----------|-------------|-----|---|-------------|
| ADIFOVR  | ADC_BA+0x50 | R/W | A/D ADINT3~0 Interrupt Flag Over Run Register | 0x0000_0000 |

|    |    |    |    |            |    |    |    |
|----|----|----|----|------------|----|----|----|
| 31 | 30 | 29 | 28 | 27         | 26 | 25 | 24 |
| -  |    |    |    |            |    |    |    |
| 23 | 22 | 21 | 20 | 19         | 18 | 17 | 16 |
| -  |    |    |    |            |    |    |    |
| 15 | 14 | 13 | 12 | 11         | 10 | 9  | 8  |
| -  |    |    |    |            |    |    |    |
| 7  | 6  | 5  | 4  | 3          | 2  | 1  | 0  |
| -  |    |    |    | ADFOV[3:0] |    |    |    |

| Bits   | Description |   |
|--------|-------------|---|
| [31:4] | -           | Reserved.   |
| [3]    | ADFOV3      | <b>A/D ADINT3 Interrupt flag over run bit</b><br>1 = ADINT3 interrupt pulse received when ADF3 is 1<br>0 = ADINT3 interrupt flag is not over run<br>It is cleared by write 1. |
| [2]    | ADFOV2      | <b>A/D ADINT2 Interrupt flag over run bit</b><br>1 = ADINT2 interrupt flag is overwrite to 1<br>0 = ADINT2 interrupt flag is not over run<br>It is cleared by write 1.        |
| [1]    | ADFOV1      | <b>A/D ADINT1 Interrupt flag over run bit</b><br>1 = ADINT1 interrupt flag is overwrite to 1<br>0 = ADINT1 interrupt flag is not over run<br>It is cleared by write 1.        |
| [0]    | ADFOV0      | <b>A/D ADINT0 Interrupt flag over run bit</b><br>1 = ADINT0 interrupt flag is overwrite to 1<br>0 = ADINT0 interrupt flag is not over run<br>It is cleared by write 1.        |

### A/D SAMPLE Over Run Flag Register (ADSPOVFR)

| Register | Offset      | R/W | Description   | Reset Value |
|----------|-------------|-----|---|-------------|
| ADSPOVFR | ADC_BA+0x54 | R/W | A/D SAMPLE Start of Conversion Over Run Flag Register | 0x0000_0000 |

|             |    |    |    |    |    |    |    |
|-------------|----|----|----|----|----|----|----|
| 31          | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -           |    |    |    |    |    |    |    |
| 23          | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| -           |    |    |    |    |    |    |    |
| 15          | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| SPOVF[15:8] |    |    |    |    |    |    |    |
| 7           | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| SPOVF[7:0]  |    |    |    |    |    |    |    |

| Bits    | Description   |
|---------|---|
| [31:16] | - Reserved.   |
| [15:8]  | <b>SPOVF[15:8]</b><br><b>A/D SAMPLEB7~0 Start Conversion Over Run Flag</b><br>1 = A new SAMPLEB event is generated while an old one event is pending.<br>0 = No SAMPLE event over run.<br>It is cleared by writing 1. |
| [7:0]   | <b>SPOVF[7:0]</b><br><b>A/D SAMPLEA7~0 Start Conversion Over Run Flag</b><br>1 = A new SAMPLEA event is generated while an old one event is pending.<br>0 = No SAMPLE event over run.<br>It is cleared by writing 1.  |



### A/D SAMPLEA0~3, SAMPLEB0~3 Control Registers (ADSPCRx0 ~ ADSPCRx3)

| Register | Offset      | R/W | Description                   | Reset Value |
|----------|-------------|-----|-------------------------------|-------------|
| ADSPCRA0 | ADC_BA+0x58 | R/W | A/D SAMPLEA0 Control Register | 0x0000_0000 |
| ADSPCRA1 | ADC_BA+0x5C | R/W | A/D SAMPLEA1 Control Register | 0x0000_0000 |
| ADSPCRA2 | ADC_BA+0x60 | R/W | A/D SAMPLEA2 Control Register | 0x0000_0000 |
| ADSPCRA3 | ADC_BA+0x64 | R/W | A/D SAMPLEA3 Control Register | 0x0000_0000 |
| ADSPCRB0 | ADC_BA+0x78 | R/W | A/D SAMPLEB0 Control Register | 0x0000_0000 |
| ADSPCRB1 | ADC_BA+0x7C | R/W | A/D SAMPLEB1 Control Register | 0x0000_0000 |
| ADSPCRB2 | ADC_BA+0x80 | R/W | A/D SAMPLEB2 Control Register | 0x0000_0000 |
| ADSPCRB3 | ADC_BA+0x84 | R/W | A/D SAMPLEB3 Control Register | 0x0000_0000 |

|                |    |        |        |    |            |                |    |
|----------------|----|--------|--------|----|------------|----------------|----|
| 31             | 30 | 29     | 28     | 27 | 26         | 25             | 24 |
| -              |    |        |        |    |            |                |    |
| 23             | 22 | 21     | 20     | 19 | 18         | 17             | 16 |
| -              |    | EXTFEN | EXTREN | -  |            | TRGDLYDIV[1:0] |    |
| 15             | 14 | 13     | 12     | 11 | 10         | 9              | 8  |
| TRGDLYCNT[7:0] |    |        |        |    |            |                |    |
| 7              | 6  | 5      | 4      | 3  | 2          | 1              | 0  |
| TRGSEL[3:0]    |    |        |        | -  | CHSEL[2:0] |                |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:20] | -           | Reserved.   |
| [21]    | EXTFEN      | <b>A/D external trigger falling edge enabled</b><br>0 = Falling edge Disabled when A/D selects STADC as trigger source.<br>1 = Falling edge Enabled when A/D selects STADC as trigger source. |
| [20]    | EXTREN      | <b>A/D external trigger rising edge enabled</b><br>0 = Rising edge Disabled when A/D selects STADC as trigger source.<br>1 = Rising edge Enabled when A/D selects STADC as trigger source.    |
| [19:18] | -           | Reserved.   |
| [17:16] | TRGDLYDIV   | <b>A/D SAMPLE Start Conversion Trigger Delay Clock Divider Selection</b><br>Trigger delay clock frequency:<br>00 = ADC_CLK/1<br>01 = ADC_CLK/2<br>10 = ADC_CLK/4<br>11 = ADC_CLK/16           |

| Bits   | Description |  |
|--------|-------------|--|
| [15:8] | TRGDLYCNT   | <b>A/D SAMPLE Start Conversion Trigger Delay Time</b><br>Trigger delay time = (TRGDLYCNT + 4) x Trigger delay clock period   |
| [7:4]  | TRGSEL      | <b>A/D SAMPLE Start Conversion Trigger Source Selection</b><br>0H = Disable hardware trigger<br>1H = External trigger from STADC pin input<br>2H = ADC ADINT0 interrupt EOC pulse trigger<br>3H = ADC ADINT1 interrupt EOC pulse trigger<br>4H = Timer0 overflow pulse trigger<br>5H = Timer1 overflow pulse trigger<br>6H = Timer2 overflow pulse trigger<br>7H = Timer3 overflow pulse trigger<br>8H = PWM00 trigger<br>9H = PWM02 trigger<br>AH = PWM04 trigger<br>BH = PWM10 trigger<br>CH = PWM12 trigger<br>DH = PWM14 trigger<br>EH = PWM20 trigger<br>FH = PWM21 trigger |
| 3      | -           | <b>Reserved.</b>   |
| [2:0]  | CHSEL       | <b>A/D SAMPLEA,B Channel Selection</b><br>0H = AINx[0]<br>1H = AINx[1]<br>2H = AINx[2]<br>3H = AINx[3]<br>4H = AINx[4]<br>5H = AINx[5]<br>6H = AINx[6]<br>7H = AINx[7]   |

### A/D SAMPLEA4~7, SAMPLEB4~7 Control Registers (ADSPCRx4 ~ ADSPCRx7)

| Register | Offset      | R/W | Description                   | Reset Value |
|----------|-------------|-----|-------------------------------|-------------|
| ADSPCRA4 | ADC_BA+0x68 | R/W | A/D SAMPLEA4 Control Register | 0x0000_0000 |
| ADSPCRA5 | ADC_BA+0x6C | R/W | A/D SAMPLEA5 Control Register | 0x0000_0000 |
| ADSPCRA6 | ADC_BA+0x70 | R/W | A/D SAMPLEA6 Control Register | 0x0000_0000 |
| ADSPCRA7 | ADC_BA+0x74 | R/W | A/D SAMPLEA7 Control Register | 0x0000_0000 |
| ADSPCRB4 | ADC_BA+0x88 | R/W | A/D SAMPLEB4 Control Register | 0x0000_0000 |
| ADSPCRB5 | ADC_BA+0x8C | R/W | A/D SAMPLEB5 Control Register | 0x0000_0000 |
| ADSPCRB6 | ADC_BA+0x90 | R/W | A/D SAMPLEB6 Control Register | 0x0000_0000 |
| ADSPCRB7 | ADC_BA+0x94 | R/W | A/D SAMPLEB7 Control Register | 0x0000_0000 |

|    |             |        |        |    |            |    |    |
|----|-------------|--------|--------|----|------------|----|----|
| 31 | 30          | 29     | 28     | 27 | 26         | 25 | 24 |
| -  |             |        |        |    |            |    |    |
| 23 | 22          | 21     | 20     | 19 | 18         | 17 | 16 |
| -  |             | EXTFEN | EXTREN | -  |            |    |    |
| 15 | 14          | 13     | 12     | 11 | 10         | 9  | 8  |
| -  |             |        |        |    |            |    |    |
| 7  | 6           | 5      | 4      | 3  | 2          | 1  | 0  |
| -  | TRGSEL[2:0] |        |        | -  | CHSEL[2:0] |    |    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:22] | -           | <b>Reserved.</b>  |
| [21]    | EXTFEN      | <b>A/D external trigger falling edge enabled</b><br>0 = Falling edge Disabled when A/D selects STADC as trigger source.<br>1 = Falling edge Enabled when A/D selects STADC as trigger source. |
| [20]    | EXTREN      | <b>A/D external trigger rising edge enabled</b><br>0 = Rising edge Disabled when A/D selects STADC as trigger source.<br>1 = Rising edge Enabled when A/D selects STADC as trigger source.    |
| [19:7]  | -           | <b>Reserved.</b>  |

| Bits  | Description   |  |
|-------|---------------|--|
| [6:4] | <b>TRGSEL</b> | <b>A/D SAMPLE Start Conversion Trigger Source Selection</b><br>0H = Disable hardware trigger<br>1H = External trigger from STADC pin input<br>2H = ADC ADINT0 interrupt EOC pulse trigger<br>3H = ADC ADINT1 interrupt EOC pulse trigger<br>4H = Timer0 overflow pulse trigger<br>5H = Timer1 overflow pulse trigger<br>6H = Timer2 overflow pulse trigger<br>7H = Timer3 overflow pulse trigger |
| 3     | -             | <b>Reserved.</b>   |
| [2:0] | <b>CHSEL</b>  | <b>A/D SAMPLEA,B Channel Selection</b><br>0H = AINx[0]<br>1H = AINx[1]<br>2H = AINx[2]<br>3H = AINx[3]<br>4H = AINx[4]<br>5H = AINx[5]<br>6H = AINx[6]<br>7H = AINx[7]   |

**A/D Simultaneous Sampling Mode Select Register (ADSMSELR)**

| Register | Offset      | R/W | Description                                  | Reset Value |
|----------|-------------|-----|--|-------------|
| ADSMSELR | ADC_BA+0xA4 | R/W | A/D SAMPLE Simultaneous Mode Select Register | 0x0000_0000 |

|          |          |          |          |          |          |          |          |
|----------|----------|----------|----------|----------|----------|----------|----------|
| 31       | 30       | 29       | 28       | 27       | 26       | 25       | 24       |
| -        |          |          |          |          |          |          |          |
| 23       | 22       | 21       | 20       | 19       | 18       | 17       | 16       |
| -        |          |          |          |          |          |          |          |
| 15       | 14       | 13       | 12       | 11       | 10       | 9        | 8        |
| -        |          |          |          |          |          |          |          |
| 7        | 6        | 5        | 4        | 3        | 2        | 1        | 0        |
| SIMUSEL7 | SIMUSEL6 | SIMUSEL5 | SIMUSEL4 | SIMUSEL3 | SIMUSEL2 | SIMUSEL1 | SIMUSEL0 |

| Bits   | Description   |
|--------|---|
| [31:8] | - Reserved.   |
| [7]    | <p><b>SIMUSEL7</b></p> <p><b>A/D SAMPLEA7, SAMPLEB7 Simultaneous Sampling Mode Selection</b><br/>                     0 = SAMPLEA7, SAMPLEB7 are in single sampling mode, both SAMPLEA7 and SAMPLEB7's 3 bits of CHSEL define the ADC channels to be converted<br/>                     1 = SAMPLEA7, SAMPLEB7 are in simultaneous sampling mode, Only SAMPLEA7 can trigger both the ADC conversions of SAMPLEA7 and SAMPLEB7, SAMPLEB7.trigger select TRGSEL is ignored. if SAMPLEA7's CHSEL = 1, SAMPLEB7's CHSEL = 3, the pair of channels are AINA[1], AINB[3], they will do the ADC conversion at the same time.</p>     |
| [6]    | <p><b>SIMUSEL6</b></p> <p><b>A/D SAMPLEA6, SAMPLEB6 Simultaneous Sampling Mode Selection</b><br/>                     0 = SAMPLEA6, SAMPLEB6 are in single sampling mode, both SAMPLEA6 and SAMPLEB6's 3 bits of CHSEL define the ADC channels to be converted<br/>                     1 = SAMPLEA6, SAMPLEB6 are in simultaneous sampling mode, Only SAMPLEA6 can trigger both the ADC conversions of SAMPLEA6 and SAMPLEB6, SAMPLEB6.trigger select TRGSEL is ignored. if SAMPLEA6's CHSEL = 1, and SAMPLEB6's CHSEL = 3, the pair of channels are AINA[1], AINB[3], they will do the ADC conversion at the same time.</p> |
| [5]    | <p><b>SIMUSEL5</b></p> <p><b>A/D SAMPLEA5, SAMPLEB5 Simultaneous Sampling Mode Selection</b><br/>                     0 = SAMPLEA5, SAMPLEB5 are in single sampling mode, both SAMPLEA5 and SAMPLEB5's 3 bits of CHSEL define the ADC channels to be converted<br/>                     1 = SAMPLEA5, SAMPLEB5 are in simultaneous sampling mode, Only SAMPLEA5 can trigger both the ADC conversions of SAMPLEA5 and SAMPLEB5, SAMPLEB5.trigger select TRGSEL is ignored. if SAMPLEA5's CHSEL = 1, and SAMPLEB5's CHSEL = 3, the pair of channels are AINA[1], AINB[3], they will do the ADC conversion at the same time.</p> |

| Bits | Description     |  |
|------|-----------------|--|
| [4]  | <b>SIMUSEL4</b> | <p><b>A/D SAMPLEA4, SAMPLEB4 Simultaneous Sampling Mode Selection</b></p> <p>0 = SAMPLEA4, SAMPLEB4 are in single sampling mode, both SAMPLEA4 and SAMPLEB4's 3 bits of CHSEL define the ADC channels to be converted</p> <p>1 = SAMPLEA4, SAMPLEB4 are in simultaneous sampling mode, Only SAMPLEA4 can trigger both the ADC conversions of SAMPLEA4 and SAMPLEB4, SAMPLEB4.trigger select TRGSEL is ignored. if SAMPLEA4's CHSEL = 1, and SAMPLEB4's CHSEL = 3, the pair of channels are AINA[1], AINB[3], they will do the ADC conversion at the same time.</p> |
| [3]  | <b>SIMUSEL3</b> | <p><b>A/D SAMPLEA3, SAMPLEB3 Simultaneous Sampling Mode Selection</b></p> <p>0 = SAMPLEA3, SAMPLEB3 are in single sampling mode, both SAMPLEA3 and SAMPLEB3's 3 bits of CHSEL define the ADC channels to be converted</p> <p>1 = SAMPLEA3, SAMPLEB3 are in simultaneous sampling mode, Only SAMPLEA3 can trigger both the ADC conversions of SAMPLEA3 and SAMPLEB3, SAMPLEB3.trigger select TRGSEL is ignored. if SAMPLEA3's CHSEL = 1, and SAMPLEB3's CHSEL = 3, the pair of channels are AINA[1], AINB[3], they will do the ADC conversion at the same time.</p> |
| [2]  | <b>SIMUSEL2</b> | <p><b>A/D SAMPLEA2, SAMPLEB2 Simultaneous Sampling Mode Selection</b></p> <p>0 = SAMPLEA2, SAMPLEB2 are in single sampling mode, both SAMPLEA2 and SAMPLEB2's 3 bits of CHSEL define the ADC channels to be converted</p> <p>1 = SAMPLEA2, SAMPLEB2 are in simultaneous sampling mode, Only SAMPLEA2 can trigger both the ADC conversions of SAMPLEA2 and SAMPLEB2, SAMPLEB2.trigger select TRGSEL is ignored. if SAMPLEA2's CHSEL = 1, and SAMPLEB2's CHSEL = 3, the pair of channels are AINA[1], AINB[3], they will do the ADC conversion at the same time.</p> |
| [1]  | <b>SIMUSEL1</b> | <p><b>A/D SAMPLEA1, SAMPLEB1 Simultaneous Sampling Mode Selection</b></p> <p>0 = SAMPLEA1, SAMPLEB1 are in single sampling mode, both SAMPLEA1 and SAMPLEB1's 3 bits of CHSEL define the ADC channels to be converted</p> <p>1 = SAMPLEA1, SAMPLEB1 are in simultaneous sampling mode, Only SAMPLEA1 can trigger both the ADC conversions of SAMPLEA1 and SAMPLEB1, SAMPLEB1.trigger select TRGSEL is ignored. if SAMPLEA1's CHSEL = 1, and SAMPLEB1's CHSEL = 3, the pair of channels are AINA[1], AINB[3], they will do the ADC conversion at the same time.</p> |
| [0]  | <b>SIMUSEL0</b> | <p><b>A/D SAMPLEA0, SAMPLEB0 Simultaneous Sampling Mode Selection</b></p> <p>0 = SAMPLEA0, SAMPLEB0 are in single sampling mode, both SAMPLEA0 and SAMPLEB0's 3 bits of CHSEL define the ADC channels to be converted</p> <p>1 = SAMPLEA0, SAMPLEB0 are in simultaneous sampling mode, Only SAMPLEA0 can trigger both the ADC conversions of SAMPLEA0 and SAMPLEB0, SAMPLEB0.trigger select TRGSEL is ignored. if SAMPLEA0's CHSEL = 1, and SAMPLEB0's CHSEL = 3, the pair of channels are AINA[1], AINB[3], they will do the ADC conversion at the same time.</p> |

**A/D Result Compare Register 0/1 (ADCMPR0/1)**

| Register | Offset      | R/W | Description                   | Reset Value |
|----------|-------------|-----|-------------------------------|-------------|
| ADCMPR0  | ADC_BA+0xA8 | R/W | A/D Result Compare Register 0 | 0x0000_0000 |
| ADCMPR1  | ADC_BA+0xAC | R/W | A/D Result Compare Register 1 | 0x0000_0000 |

|           |    |         |    |            |         |         |          |
|-----------|----|---------|----|------------|---------|---------|----------|
| 31        | 30 | 29      | 28 | 27         | 26      | 25      | 24       |
| -         |    |         |    | CMPD[11:8] |         |         |          |
| 23        | 22 | 21      | 20 | 19         | 18      | 17      | 16       |
| CMPD[7:0] |    |         |    |            |         |         |          |
| 15        | 14 | 13      | 12 | 11         | 10      | 9       | 8        |
| -         |    |         |    | CMPMATCNT  |         |         |          |
| 7         | 6  | 5       | 4  | 3          | 2       | 1       | 0        |
| -         |    | CMPSMPL |    |            | CMPCOND | ADCMPIE | ADCMP_EN |

| Bits    | Description      |   |
|---------|------------------|---|
| [31:28] | -                | <b>Reserved.</b>  |
| [27:16] | <b>CMPD</b>      | <b>Comparison Data</b><br>The 12 bits data is used to compare with the conversion result of specified SAMPLE. Software can use it to monitor the external analog input pin voltage transition without imposing a load on software.  |
| [15:12] | -                | <b>Reserved.</b>  |
| [11:8]  | <b>CMPMATCNT</b> | <b>Compare Match Count</b><br>When the specified A/D SAMPLE analog conversion result matches the compare condition defined by CMPCOND, the internal match counter will increase 1. When the internal counter reaches the value to (CMPMATCNT +1), the CMPF bit will be set.   |
| [7:6]   | -                | <b>Reserved.</b>  |
| [5:3]   | <b>CMPSMPL</b>   | <b>Compare SAMPLE Selection</b><br>000 = SAMPLEA0 conversion result ADDRA0 is selected to be compared.<br>001 = SAMPLEA1 conversion result ADDRA1 is selected to be compared.<br>010 = SAMPLEA2 conversion result ADDRA2 is selected to be compared.<br>011 = SAMPLEA3 conversion result ADDRA3 is selected to be compared.<br>100 = SAMPLEB0 conversion result ADDR0B is selected to be compared.<br>101 = SAMPLEB1 conversion result ADDR1B is selected to be compared.<br>110 = SAMPLEB2 conversion result ADDR2B is selected to be compared.<br>111 = SAMPLEB3 conversion result ADDR3B is selected to be compared. |

| Bits | Description     |  |
|------|-----------------|--|
| [2]  | <b>CMPCOND</b>  | <p><b>Compare Condition</b></p> <p>1= Set the compare condition as that when a 12-bit A/D conversion result is greater or equal to the 12-bit CMPD (ADCMPRx[27:16]), the internal match counter will increase one.</p> <p>0= Set the compare condition as that when a 12-bit A/D conversion result is less than the 12-bit CMPD (ADCMPRx[27:16]), the internal match counter will increase one.</p> <p><b>Note:</b> When the internal counter reaches the value to (CMPMATCNT +1), the CMPF bit will be set.</p> |
| [1]  | <b>ADCMPIE</b>  | <p><b>A/D Result Compare Interrupt Enable</b></p> <p>1 = Compare function interrupt Enabled.</p> <p>0 = Compare function interrupt Disabled.</p> <p>If the compare function is enabled and the compare condition matches the setting of CMPCOND and CMPMATCNT, ADCMPF bit will be asserted, in the meanwhile, if ADCMPIE is set to 1, a compare interrupt request is generated.</p>  |
| [0]  | <b>ADCMP_EN</b> | <p><b>A/D Result Compare Enable</b></p> <p>1 = Compare Enabled.</p> <p>0 = Compare Disabled.</p> <p>Set this bit to 1 to enable compare CMPD[11:0] with specified SAMPLE conversion result when converted data is loaded into ADDR register.</p>   |



### A/D Status Register 0 (ADSR0)

| Register | Offset      | R/W | Description           | Reset Value |
|----------|-------------|-----|-----------------------|-------------|
| ADSR0    | ADC_BA+0xB0 | R   | A/D Status Register 0 | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| OVERRUN[15:8] |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| OVERRUN[7:0]  |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| VALID[15:8]   |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| VALID[7:0]    |    |    |    |    |    |    |    |

| Bits    | Description   |   |
|---------|---------------|---|
| [31:24] | OVERRUN[15:8] | <b>ADDRB7~0 Over Run Flag</b><br>It is a mirror to OVERRUN bit in SAMPLEB A/D result data register ADDRbX |
| [23:16] | OVERRUN[7:0]  | <b>ADDRA7~0 Over Run Flag</b><br>It is a mirror to OVERRUN bit in SAMPLEA A/D result data register ADDRAX |
| [15:8]  | VALID[15:8]   | <b>ADDRB7~0 Data Valid Flag</b><br>It is a mirror of VALID bit in SAMPLEB A/D result data register ADDRbX |
| [7:0]   | VALID[7:0]    | <b>ADDRA7~0 Data Valid Flag</b><br>It is a mirror of VALID bit in SAMPLEA A/D result data register ADDRAX |

### A/D Status Register 1 (ADSR1)

| Register | Offset      | R/W | Description           | Reset Value |
|----------|-------------|-----|-----------------------|-------------|
| ADSR1    | ADC_BA+0xB4 | R/W | A/D Status Register 1 | 0x0000_0000 |

| 31       | 30       | 29      | 28      | 27       | 26     | 25     | 24     |
|----------|----------|---------|---------|----------|--------|--------|--------|
| -        |          |         |         | AOVERRUN | AVALID | ASPOVF | AADFOV |
| 23       | 22       | 21      | 20      | 19       | 18     | 17     | 16     |
| -        | CHANNELB |         |         | -        |        |        | BUSYB  |
| 15       | 14       | 13      | 12      | 11       | 10     | 9      | 8      |
| -        | CHANNELA |         |         | -        |        |        | BUSYA  |
| 7        | 6        | 5       | 4       | 3        | 2      | 1      | 0      |
| ADCMPPF1 | ADCMPPF0 | ADCMPO1 | ADCMPO0 | ADF3     | ADF2   | ADF1   | ADF0   |

| Bits    | Description |  |
|---------|-------------|--|
| [31:28] | -           | Reserved.  |
| [27]    | AOVERRUN    | <b>All SAMPLE A/D Result Data Register Over Run Flags Check</b><br>1 = Any one of SAMPLE data register over run flag OVERRUNx is set to 1.<br>0 = None of SAMPLE data register over run flag OVERRUNx is set to 1.<br>This bit will keep 1 when any OVERRUNx Flag is equal to 1. |
| [26]    | AVALID      | <b>All SAMPLE A/D Result Data Register ADDR Data Valid Flag Check</b><br>1 = Any one of SAMPLE data register valid flag VALIDx is set to 1.<br>0 = None of SAMPLE data register valid flag VALIDx is set to 1.<br>This bit will keep 1 when any VALIDx Flag is equal to 1.       |
| [25]    | ASPOVF      | <b>All A/D SAMPLE Start Conversion Over Run Flags Check</b><br>1 = Any one of SAMPLE event over run flag SPOVFx is set to 1.<br>0 = None of SAMPLE event over run flag SPOVFx is set to 1.<br>This bit will keep 1 when any SPOVFx Flag is equal to 1.                           |
| [24]    | AADFOV      | <b>All A/D Interrupt Flag Over Run Bits Check</b><br>1 = Any one of ADINT interrupt flag ADFOVx is overwritten to 1<br>0 = None of ADINT interrupt flag ADFOVx is overwritten to 1<br>This bit will keep 1 when any ADFOVx Flag is equal to 1.                                   |
| [23]    | -           | Reserved.  |

| Bits    | Description     |   |
|---------|-----------------|---|
| [22:20] | <b>CHANNELB</b> | <b>Current Conversion Channel</b><br>This filed reflects ADCB current conversion channel when BUSYB=1. When BUSYB=0, it shows the last converted channel.<br>It is read only.<br>00H = AINB[0]<br>01H = AINB[1]<br>02H = AINB[2]<br>03H = AINB[3]<br>04H = AINB[4]<br>05H = AINB[5]<br>06H = AINB[6]<br>07H = AINB[7] |
| [19:17] | -               | <b>Reserved.</b>  |
| [16]    | <b>BUSYB</b>    | <b>BUSY/IDLE</b><br>1 = A/D converter B (ADCB) is busy at conversion.<br>0 = A/D converter B (ADCB) is in idle state.<br>It is read only.   |
| [14:12] | <b>CHANNELA</b> | <b>Current Conversion Channel</b><br>This filed reflects ADCA current conversion channel when BUSYA=1. When BUSYA=0, it shows the last converted channel.<br>It is read only.<br>00H = AINA[0]<br>01H = AINA[1]<br>02H = AINA[2]<br>03H = AINA[3]<br>04H = AINA[4]<br>05H = AINA[5]<br>06H = AINA[6]<br>07H = AINA[7] |
| [11:9]  | -               | <b>Reserved.</b>  |
| [8]     | <b>BUSYA</b>    | <b>BUSY/IDLE</b><br>1 = A/D converter A (ADCA) is busy at conversion.<br>0 = A/D converter A (ADCA) is in idle state.<br>It is read only.   |
| [7]     | <b>ADCMPPF1</b> | <b>ADC Compare 1 Flag</b><br>When the specific SAMPLE A/D conversion result meets setting condition in ADCMPR1 then this bit is set to 1. And it is cleared by write 1.<br>1 = Conversion result in ADDR meets ADCMPR1 setting<br>0 = Conversion result in ADDR does not meet ADCMPR1 setting                         |

| Bits | Description    |  |
|------|----------------|--|
| [6]  | <b>ADCMPO0</b> | <b>ADC Compare 0 Flag</b><br>When the specific SAMPLE A/D conversion result meets setting condition in ADCMPR0 then this bit is set to 1. And it is cleared by write 1.<br>1 = Conversion result in ADDR meets ADCMPR0 setting<br>0 = Conversion result in ADDR does not meet ADCMPR0 setting  |
| [5]  | <b>ADCMPO1</b> | <b>ADC Compare 1 Output Status Bit</b><br>The 12 bits compare1 data (ADCMPR1[27:16]) is used to compare with conversion result of specified SAMPLE. Software can use it to monitor the external analog input pin voltage status.<br>1 = Conversion result in ADDR great than or equal ADCMPR1[27:16] setting<br>0 = Conversion result in ADDR less than ADCMPR1[27:16] setting       |
| [4]  | <b>ADCMPO0</b> | <b>ADC Compare 0 Output Status Bit</b><br>The 12 bits compare0 data (ADCMPR0[27:16]) is used to compare with conversion result of specified SAMPLE. Software can use it to monitor the external analog input pin voltage status.<br>1 = Conversion result in ADDR is great than or equal ADCMPR0[27:16] setting<br>0 = Conversion result in ADDR is less than ADCMPR0[27:16] setting |
| [3]  | <b>ADF3</b>    | <b>A/D ADINT3 Interrupt Flag</b><br>1 = ADINT3 interrupt pulse received<br>0 = No ADINT3 interrupt pulse received<br>It is cleared by writing 1.<br>This bit indicates whether an A/D conversion of specific SAMPLE has been completed   |
| [2]  | <b>ADF2</b>    | <b>A/D ADINT2 Interrupt Flag</b><br>1 = ADINT2 interrupt pulse received<br>0 = No ADINT2 interrupt pulse received<br>It is cleared by writing 1.<br>This bit indicates whether an A/D conversion of specific SAMPLE has been completed.  |
| [1]  | <b>ADF1</b>    | <b>A/D ADINT1 Interrupt Flag</b><br>1 = ADINT1 interrupt pulse has been received<br>0 = no ADINT1 interrupt pulse received<br>It is cleared by writing 1.<br>This bit indicates whether an A/D conversion of specific SAMPLE has been completed.   |
| [0]  | <b>ADF0</b>    | <b>A/D ADINT0 Interrupt Flag</b><br>1 = ADINT0 interrupt pulse received.<br>0 = No ADINT0 interrupt pulse received.<br>It is cleared by writing 1.<br>This bit indicates whether an A/D conversion of specific SAMPLE has been completed.  |

### A/D Timing Control Register (ADTCR)

| Register | Offset      | R/W | Description                 | Reset Value |
|----------|-------------|-----|-----------------------------|-------------|
| ADTCR    | ADC_BA+0xB8 | R/W | A/D Timing Control Register | 0x0000_0000 |

|        |    |    |    |    |    |    |    |
|--------|----|----|----|----|----|----|----|
| 31     | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| -      |    |    |    |    |    |    |    |
| 23     | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| ADBEST |    |    |    |    |    |    |    |
| 15     | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| -      |    |    |    |    |    |    |    |
| 7      | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| ADAEST |    |    |    |    |    |    |    |

| Bits    | Description |  |
|---------|-------------|--|
| [31:24] | -           | Reserved.  |
| [23:16] | ADBEST      | <b>ADCB Extend Sampling Time</b><br>When A/D converting at high conversion rate, the sampling time of analog input voltage may not enough if input channel loading is heavy, SW can extend A/D sampling time after trigger source is coming to get enough sampling time.<br>The range of start delay time is from 0~255 ADC clock. |
| [15:8]  | -           | Reserved.  |
| [7:0]   | ADAEST      | <b>ADCA Extend Sampling Time</b><br>When A/D converting at high conversion rate, the sampling time of analog input voltage may not enough if input channel loading is heavy, SW can extend A/D sampling time after trigger source is coming to get enough sampling time.<br>The range of start delay time is from 0~255 ADC clock. |

**A/D Double Data Registers for A/D Data Registers (ADDRA0~3, ADDR0~3)**

| Register | Offset       | R/W | Description                             | Reset Value |
|----------|--------------|-----|---|-------------|
| ADDRA0B  | ADC_BA+0x100 | R   | A/D double Data Register 0 for SAMPLEA0 | 0x0000_0000 |
| ADDRA1B  | ADC_BA+0x104 | R   | A/D double Data Register 1 for SAMPLEA1 | 0x0000_0000 |
| ADDRA2B  | ADC_BA+0x108 | R   | A/D double Data Register 2 for SAMPLEA2 | 0x0000_0000 |
| ADDRA3B  | ADC_BA+0x10C | R   | A/D double Data Register 3 for SAMPLEA3 | 0x0000_0000 |
| ADDRB0B  | ADC_BA+0x120 | R   | A/D double Data Register 0 for SAMPLEB0 | 0x0000_0000 |
| ADDRB1B  | ADC_BA+0x124 | R   | A/D double Data Register 1 for SAMPLEB1 | 0x0000_0000 |
| ADDRB2B  | ADC_BA+0x128 | R   | A/D double Data Register 2 for SAMPLEB2 | 0x0000_0000 |
| ADDRB3B  | ADC_BA+0x12C | R   | A/D double Data Register 3 for SAMPLEB3 | 0x0000_0000 |

|             |    |    |    |               |    |    |       |
|-------------|----|----|----|---------------|----|----|-------|
| 31          | 30 | 29 | 28 | 27            | 26 | 25 | 24    |
| -           |    |    |    |               |    |    |       |
| 23          | 22 | 21 | 20 | 19            | 18 | 17 | 16    |
| -           |    |    |    |               |    |    | VALID |
| 15          | 14 | 13 | 12 | 11            | 10 | 9  | 8     |
| -           |    |    |    | RSLTDB [11:8] |    |    |       |
| 7           | 6  | 5  | 4  | 3             | 2  | 1  | 0     |
| RSLTDB[7:0] |    |    |    |               |    |    |       |

| Bits    | Description |   |
|---------|-------------|---|
| [31:17] | -           | Reserved.   |
| [16]    | VALID       | <b>Valid Flag</b><br>1 = Double data in RSLT[11:0] bits is valid.<br>0 = Double data in RSLT[11:0] bits is not valid.<br>This bit is set to 1 when corresponding SAMPLE channel analog input conversion is completed and cleared by hardware after ADDR register is read. |
| [15:12] | -           | Reserved.   |
| [11:0]  | RSLT        | <b>A/D Conversion Result</b><br>This field contains 12 bits conversion result.  |

### A/D Double Buffer Mode select (ADDBM)

| Register | Offset       | R/W | Description                 | Reset Value |
|----------|--------------|-----|-----------------------------|-------------|
| ADDBM    | ADC_BA+0x130 | R/W | A/D Timing Control Register | 0x0000_0000 |

|    |    |    |    |       |       |       |       |
|----|----|----|----|-------|-------|-------|-------|
| 31 | 30 | 29 | 28 | 27    | 26    | 25    | 24    |
| -  |    |    |    |       |       |       |       |
| 23 | 22 | 21 | 20 | 19    | 18    | 17    | 16    |
| -  |    |    |    |       |       |       |       |
| 15 | 14 | 13 | 12 | 11    | 10    | 9     | 8     |
| -  |    |    |    | DBMB3 | DBMB2 | DBMB1 | DBMB0 |
| 7  | 6  | 5  | 4  | 3     | 2     | 1     | 0     |
| -  |    |    |    | DBMA3 | DBMA2 | DBMA1 | DBMA0 |

| Bits    | Description |   |
|---------|-------------|---|
| [31:12] | -           | Reserved.   |
| [11]    | DBMB3       | <b>Double buffer mode for SAMPLE B3</b><br>1 =SampleB3 has two sample result registers.<br>0 = SampleB3 has one sample result register. (default)   |
| [10]    | DBMB2       | <b>Double buffer mode for SAMPLE B2</b><br>1 =SampleB2 has two sample result registers.<br>0 = SampleB2 has one sample result register. (default).. |
| [9]     | DBMB1       | <b>Double buffer mode for SAMPLE B1</b><br>1 =SampleB1 has two sample result registers.<br>0 = SampleB1 has one sample result register. (default).  |
| [8]     | DBMB0       | <b>Double buffer mode for SAMPLE B0</b><br>1 =SampleB0 has two sample result registers.<br>0 = SampleB0 has one sample result register. (default)   |
| [7:4]   | -           | Reserved.   |
| [3]     | DBMA3       | <b>Double buffer mode for SAMPLE A3</b><br>1 =SampleA3 has two sample result registers.<br>0 = SampleA3 has one sample result register. (default).  |
| [2]     | DBMA2       | <b>Double buffer mode for SAMPLE A2</b><br>1 =SampleA2 has two sample result registers.<br>0 = SampleA2 has one sample result register. (default).. |

| Bits | Description  |  |
|------|--------------|--|
| [1]  | <b>DBMA1</b> | <b>Double buffer mode for SAMPLE A1</b><br>1 =SampleA1 has two sample result registers.<br>0 = SampleA1 has one sample result register. (default). |
| [0]  | <b>DBMA0</b> | <b>Double buffer mode for SAMPLE A0</b><br>1 =SampleA0 has two sample result registers.<br>0 = SampleA0 has one sample result register. (default). |



### A/D interrupt Source Enable Control Register. (ADINT0SRCTL ~ ADINT3SRCTL)

| Register    | Offset       | R/W | Description   | Reset Value |
|-------------|--------------|-----|---|-------------|
| ADINT0SRCTL | ADC_BA+0x134 | R/W | A/D interrupt source enable control register for ADINT0 | 0x0000_0000 |
| ADINT1SRCTL | ADC_BA+0x138 | R/W | A/D interrupt source enable control register for ADINT1 | 0x0000_0000 |
| ADINT2SRCTL | ADC_BA+0x13C | R/W | A/D interrupt source enable control register for ADINT2 | 0x0000_0000 |
| ADINT3SRCTL | ADC_BA+0x140 | R/W | A/D interrupt source enable control register for ADINT3 | 0x0000_0000 |

|         |         |         |         |         |         |         |         |
|---------|---------|---------|---------|---------|---------|---------|---------|
| 31      | 30      | 29      | 28      | 27      | 26      | 25      | 24      |
| -       |         |         |         |         |         |         |         |
| 23      | 22      | 21      | 20      | 19      | 18      | 17      | 16      |
| -       |         |         |         |         |         |         |         |
| 15      | 14      | 13      | 12      | 11      | 10      | 9       | 8       |
| IESPLB7 | IESPLB6 | IESPLB5 | IESPLB4 | IESPLB3 | IESPLB2 | IESPLB1 | IESPLB0 |
| 7       | 6       | 5       | 4       | 3       | 2       | 1       | 0       |
| IESPLA7 | IESPLA6 | IESPLA5 | IESPLA4 | IESPLA3 | IESPLA2 | IESPLA1 | IESPLA0 |

| Bits    | Description |   |
|---------|-------------|---|
| [31:16] | -           | Reserved.   |
| [15]    | IESPLB7     | <b>SAMPLE B7 Interrupt Mask Enable</b><br>1 = SAMPLE B7 interrupt mask Enabled.<br>0 = SAMPLE B7 interrupt mask Disabled. |
| [14]    | IESPLB6     | <b>SAMPLE B6 Interrupt Mask Enable</b><br>1 = SAMPLE B6 interrupt mask Enabled.<br>0 = SAMPLE B6 interrupt mask Disabled. |
| [13]    | IESPLB5     | <b>SAMPLE B5 Interrupt Mask Enable</b><br>1 = SAMPLE B5 interrupt mask Enabled.<br>0 = SAMPLE B5 interrupt mask Disabled. |
| [12]    | IESPLB4     | <b>SAMPLE B4 Interrupt Mask Enable</b><br>1 = SAMPLE B4 interrupt mask Enabled.<br>0 = SAMPLE B4 interrupt mask Disabled. |
| [11]    | IESPLB3     | <b>SAMPLE B3 interrupt mask Enable</b><br>1 = SAMPLE B3 interrupt mask Enabled.<br>0 = SAMPLE B3 interrupt mask Disabled. |
| [10]    | IESPLB2     | <b>SAMPLE B2 interrupt mask Enable</b><br>1 = SAMPLE B2 interrupt mask Enabled.<br>0 = SAMPLE B2 interrupt mask Disabled. |

| Bits | Description    |   |
|------|----------------|---|
| [9]  | <b>IESPLB1</b> | <b>SAMPLE B1 interrupt mask Enable</b><br>1 = SAMPLE B1 interrupt mask Enabled.<br>0 = SAMPLE B1 interrupt mask Disabled. |
| [8]  | <b>IESPLB0</b> | <b>SAMPLE B0 interrupt mask Enable</b><br>1 = SAMPLE B0 interrupt mask Enabled.<br>0 = SAMPLE B0 interrupt mask Disabled. |
| [7]  | <b>IESPLA7</b> | <b>SAMPLE A7 interrupt mask Enable</b><br>1 = SAMPLE A7 interrupt mask Enabled.<br>0 = SAMPLE A7 interrupt mask Disabled. |
| [6]  | <b>IESPLA6</b> | <b>SAMPLE A6 interrupt mask Enable</b><br>1 = SAMPLE A6 interrupt mask Enabled.<br>0 = SAMPLE A6 interrupt mask Disabled. |
| [5]  | <b>IESPLA5</b> | <b>SAMPLE A5 interrupt mask Enable</b><br>1 = SAMPLE A5 interrupt mask Enabled.<br>0 = SAMPLE A5 interrupt mask Disabled. |
| [4]  | <b>IESPLA4</b> | <b>SAMPLE A4 interrupt mask Enable</b><br>1 = SAMPLE A4 interrupt mask Enabled.<br>0 = SAMPLE A4 interrupt mask Disabled. |
| [3]  | <b>IESPLA3</b> | <b>SAMPLE A3 interrupt mask Enable</b><br>1 = SAMPLE A3 interrupt mask Enabled.<br>0 = SAMPLE A3 interrupt mask Disabled. |
| [2]  | <b>IESPLA2</b> | <b>SAMPLE A2 interrupt mask Enable</b><br>1 = SAMPLE A2 interrupt mask Enabled.<br>0 = SAMPLE A2 interrupt mask Disabled. |
| [1]  | <b>IESPLA1</b> | <b>SAMPLE A1 interrupt mask Enable</b><br>1 = SAMPLE A1 interrupt mask Enabled.<br>0 = SAMPLE A1 interrupt mask Disabled. |
| [0]  | <b>IESPLA0</b> | <b>SAMPLE A0 interrupt mask Enable</b><br>1 = SAMPLE A0 interrupt mask Enabled.<br>0 = SAMPLE A0 interrupt mask Disabled. |

### A/D Trigger Condition Control Register. (SMPA0TRGEN ~ SMPA3TRGEN, SMPB0TRGEN ~ SMPB3TRGEN )

| Register          | Offset       | R/W | Description                        | Reset Value |
|-------------------|--------------|-----|------------------------------------|-------------|
| <b>SMPA0TRGEN</b> | ADC_BA+0x144 | R/W | A/D trigger condition for SAMPLEA0 | 0x0000_0000 |
| <b>SMPA1TRGEN</b> | ADC_BA+0x148 | R/W | A/D trigger condition for SAMPLEA1 | 0x0000_0000 |
| <b>SMPA2TRGEN</b> | ADC_BA+0x14C | R/W | A/D trigger condition for SAMPLEA2 | 0x0000_0000 |
| <b>SMPA3TRGEN</b> | ADC_BA+0x150 | R/W | A/D trigger condition for SAMPLEA3 | 0x0000_0000 |
| <b>SMPB0TRGEN</b> | ADC_BA+0x154 | R/W | A/D trigger condition for SAMPLEB0 | 0x0000_0000 |
| <b>SMPB1TRGEN</b> | ADC_BA+0x158 | R/W | A/D trigger condition for SAMPLEB1 | 0x0000_0000 |
| <b>SMPB2TRGEN</b> | ADC_BA+0x15C | R/W | A/D trigger condition for SAMPLEB2 | 0x0000_0000 |
| <b>SMPB3TRGEN</b> | ADC_BA+0x160 | R/W | A/D trigger condition for SAMPLEB3 | 0x0000_0000 |

| 31              | 30              | 29              | 28              | 27              | 26              | 25              | 24              |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| <b>PWM21CEN</b> | <b>PWM21PEN</b> | <b>PWM21FEN</b> | <b>PWM21REN</b> | <b>PWM20CEN</b> | <b>PWM20PEN</b> | <b>PWM20FEN</b> | <b>PWM20REN</b> |
| 23              | 22              | 21              | 20              | 19              | 18              | 17              | 16              |
| <b>PWM14CEN</b> | <b>PWM14PEN</b> | <b>PWM14FEN</b> | <b>PWM14REN</b> | <b>PWM12CEN</b> | <b>PWM12PEN</b> | <b>PWM12FEN</b> | <b>PWM12REN</b> |
| 15              | 14              | 13              | 12              | 11              | 10              | 9               | 8               |
| <b>PWM10CEN</b> | <b>PWM10PEN</b> | <b>PWM10FEN</b> | <b>PWM10REN</b> | <b>PWM04CEN</b> | <b>PWM04PEN</b> | <b>PWM04FEN</b> | <b>PWM04REN</b> |
| 7               | 6               | 5               | 4               | 3               | 2               | 1               | 0               |
| <b>PWM02CEN</b> | <b>PWM02PEN</b> | <b>PWM02FEN</b> | <b>PWM02REN</b> | <b>PWM00CEN</b> | <b>PWM00PEN</b> | <b>PWM00FEN</b> | <b>PWM00REN</b> |

| Bits | Description  |
|------|--|
| [31] | <b>PWM21CEN</b><br><b>PWM21 Center Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [30] | <b>PWM21PEN</b><br><b>PWM21 Period Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [29] | <b>PWM21FEN</b><br><b>PWM21 Falling Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled. |
| [28] | <b>PWM21REN</b><br><b>PWM21 Rising Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.  |

| Bits | Description      |   |
|------|------------------|---|
| [27] | <b>PWM20CEN</b>  | <b>PWM20 Center Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [26] | <b>PWM20PEN</b>  | <b>PWM20 Period Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [25] | <b>PWM20FEN</b>  | <b>PWM20 Falling Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled. |
| [24] | <b>PWM20REN</b>  | <b>PWM20 Rising Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.  |
| [23] | <b>PWM14CEN</b>  | <b>PWM14 Center Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [22] | <b>PWM14PEN</b>  | <b>PWM14 Period Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [21] | <b>PWM14FEN</b>  | <b>PWM14 Falling Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled. |
| [20] | <b>PWM14REN</b>  | <b>PWM14 Rising Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.  |
| [19] | <b>PWM12CEN</b>  | <b>PWM12 Center Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [18] | <b>PWM12PEN</b>  | <b>PWM12 Period Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [17] | <b>PWM120FEN</b> | <b>PWM12 Falling Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled. |
| [16] | <b>PWM12REN</b>  | <b>PWM12 Rising Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.  |

| Bits | Description     |   |
|------|-----------------|---|
| [15] | <b>PWM10CEN</b> | <b>PWM10 Center Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [14] | <b>PWM10PEN</b> | <b>PWM10 Period Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [13] | <b>PWM10FEN</b> | <b>PWM10 Falling Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled. |
| [12] | <b>PWM10REN</b> | <b>PWM10 Rising Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.  |
| [11] | <b>PWM04CEN</b> | <b>PWM04 Center Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [10] | <b>PWM04PEN</b> | <b>PWM04 Period Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [9]  | <b>PWM04FEN</b> | <b>PWM04 Falling Rdge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled. |
| [8]  | <b>PWM04REN</b> | <b>PWM04 Rising Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.  |
| [7]  | <b>PWM02CEN</b> | <b>PWM02 Center Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [6]  | <b>PWM02PEN</b> | <b>PWM02 Period trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [5]  | <b>PWM02FEN</b> | <b>PWM02 Falling Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled. |
| [4]  | <b>PWM02REN</b> | <b>PWM02 Rising Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.  |

| Bits | Description     |   |
|------|-----------------|---|
| [3]  | <b>PWM00CEN</b> | <b>PWM00 Center Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [2]  | <b>PWM00PEN</b> | <b>PWM00 Period Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.       |
| [1]  | <b>PWM00FEN</b> | <b>PWM00 Falling Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled. |
| [0]  | <b>PWM00REN</b> | <b>PWM00 Rising Edge Trigger Enable</b><br>1 = Enabled.<br>0 = Disabled.  |

## 24 ANALOG COMPARATOR

### 24.1 Overview

The NuMicro™ NM15xx Series contains three comparators which can be used in a number of different configurations. The comparator output is logic 1 when positive input voltage is greater than negative input voltage; otherwise the output is logic 0. Each comparator can be configured to cause an interrupt when the comparator output value changes. The block diagram is shown in Figure 24–1.

**Note:** the analog input port pins must be configured as input type before Analog Comparator function is enabled.

### 24.2 Features

- Analog input voltage range: 0~  $V_{DDA}$
- Supports Hysteresis function
- Supports optional internal reference voltage input at negative end
- Supports optional OP amplifier output voltage input at positive end
- Supports comparator output inverse function
- Supports the comparator output can be the brake source for EPWM function



### 24.3 Block Diagram

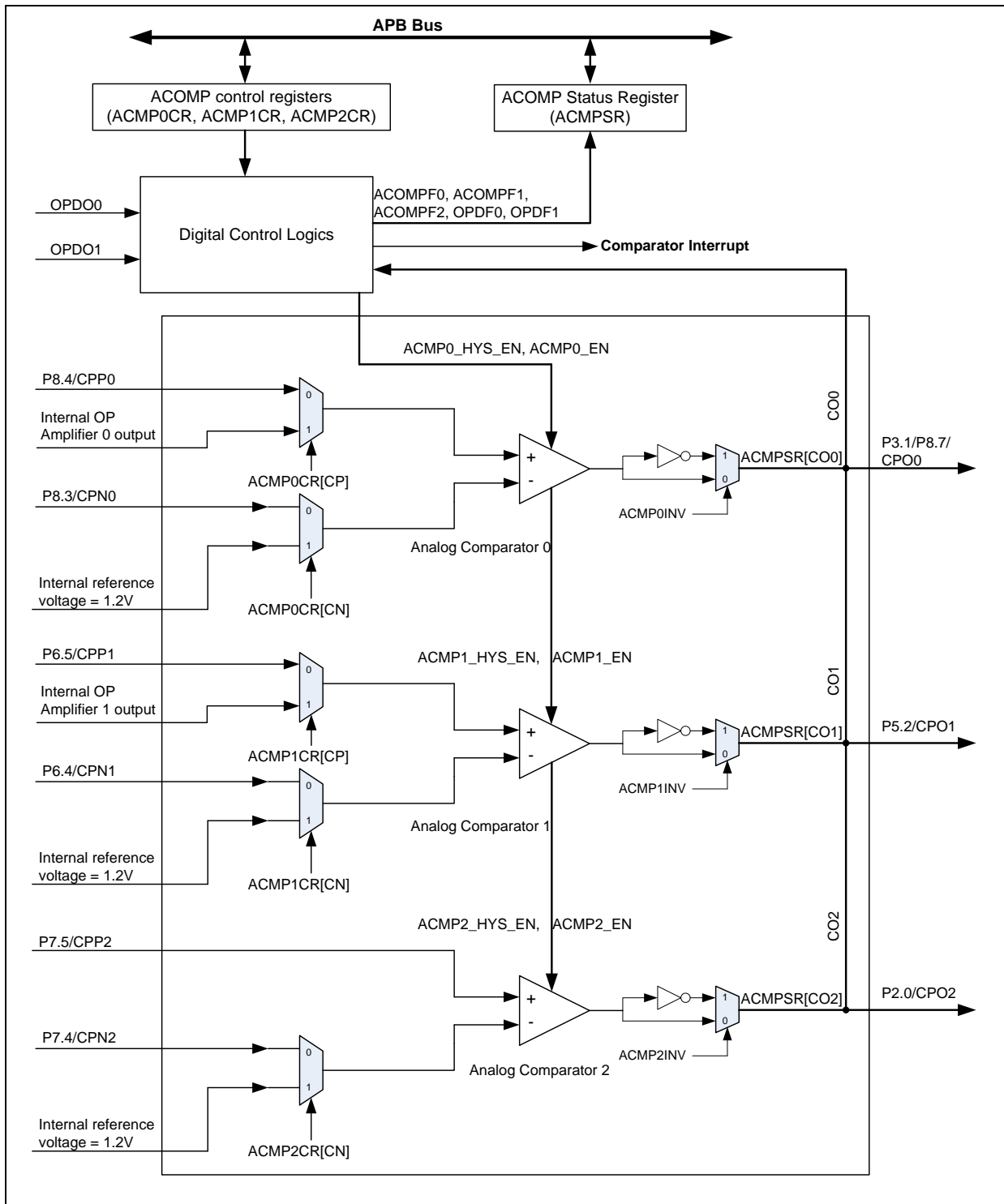


Figure 24-1 Analog Comparator Block Diagram

## 24.4 Interrupt Sources

The output of comparators are sampled by PCLK and reflected at CO1, CO2 and CO2 of ACMPSR register. If ACMPIEx bit in ACMPCR is set to 1, the comparator interrupt will be enabled. As the output state of comparator is changed, the comparator interrupt will be asserted and the corresponding flag, ACMPF<sub>x</sub>, will be set. Software can clear the flag to 0 by writing 1 to it.

OPDF0, OPDF1 interrupt flag are set respectively by hardware whenever the OP0, 1 amplifier Schmitt trigger non-inverting buffer output change states. Software can clear the flag to 0 by writing 1 to it.

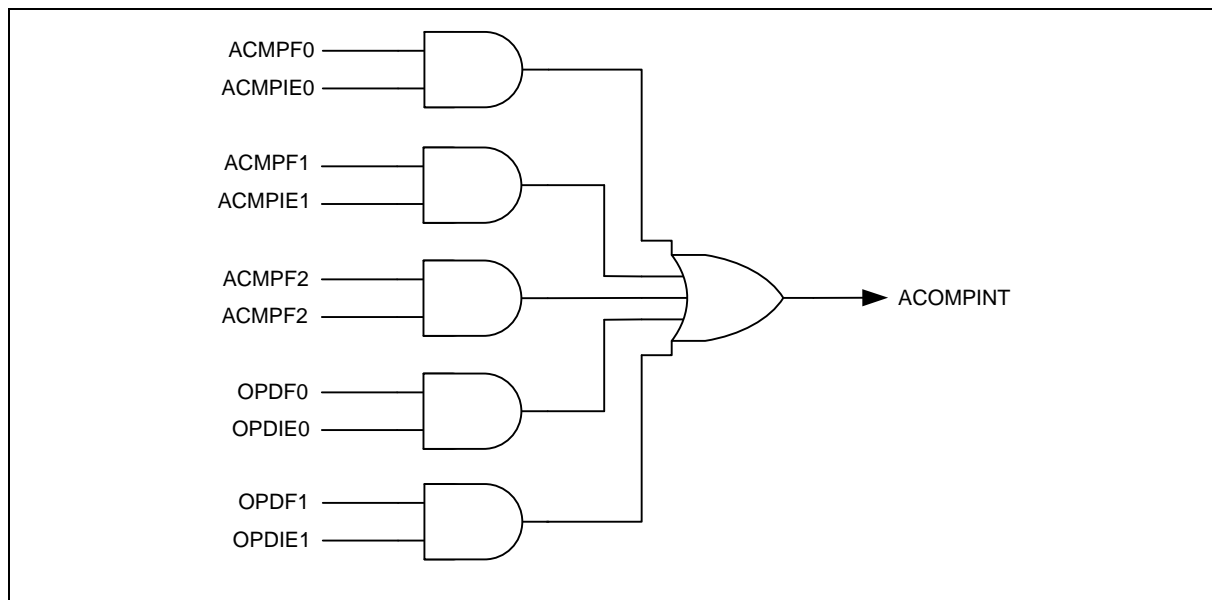


Figure 24–2 Analog Comparator Controller Interrupt

## 24.5 Hysteresis Function

The analog comparator provides hysteresis function to make the comparator output transition more stable. If comparator output is 0, it will not change to 1 until the positive input voltage exceeds the negative input voltage by a positive hysteresis voltage. Similarly, if comparator output is 1, it will not change to 0 until the positive input voltage drops below the negative input voltage by a negative hysteresis voltage.

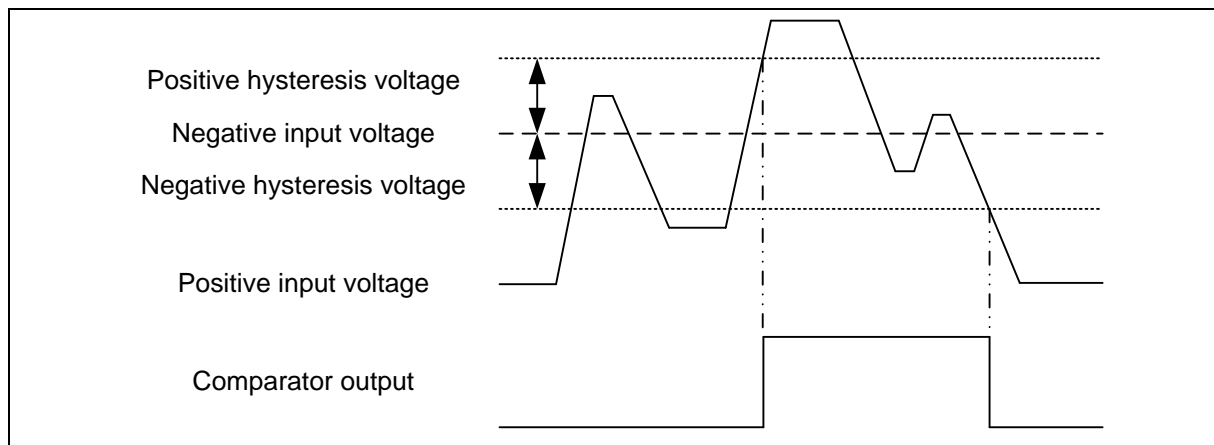


Figure 24-3 Comparator Hysteresis function

### 24.6 Register Map

R: read only, **W**: write only, **R/W**: both read and write, **C**: Only value 0 can be written

| Register  | Offset       | R/W | Description                          | Reset Value |
|---|--------------|-----|--------------------------------------|-------------|
| <b>ACMP Base Address:</b><br><b>ACMP_BA = 0x400D_0000</b> |              |     |                                      |             |
| <b>ACMP0CR</b>  | ACMP_BA+0x00 | R/W | Analog Comparator 0 Control Register | 0x0000_0000 |
| <b>ACMP1CR</b>  | ACMP_BA+0x04 | R/W | Analog Comparator 1 Control Register | 0x0000_0000 |
| <b>ACMP2CR</b>  | ACMP_BA+0x08 | R/W | Analog Comparator 2 Control Register | 0x0000_0000 |
| <b>ACMPSR</b>   | ACMP_BA+0x0C | R/W | Analog Comparator Status Register    | 0x0000_0000 |

## 24.7 Register Description

### Analog CMP 0 Control Register (ACMP0CR)

| Register | Offset       | R/W | Description                          | Reset Value |
|----------|--------------|-----|--------------------------------------|-------------|
| ACMP0CR  | ACMP_BA+0x00 | R/W | Analog Comparator 0 Control Register | 0x0000_0000 |

|    |          |    |     |     |              |         |          |
|----|----------|----|-----|-----|--------------|---------|----------|
| 31 | 30       | 29 | 28  | 27  | 26           | 25      | 24       |
| -  |          |    |     |     |              |         |          |
| 23 | 22       | 21 | 20  | 19  | 18           | 17      | 16       |
| -  |          |    |     |     |              |         |          |
| 15 | 14       | 13 | 12  | 11  | 10           | 9       | 8        |
| -  |          |    |     |     |              |         |          |
| 7  | 6        | 5  | 4   | 3   | 2            | 1       | 0        |
| -  | ACMP0INV | -  | CN0 | CP0 | ACMP0_HYS_EN | ACMPIE0 | ACMP0_EN |

| Bits   | Description   |
|--------|---|
| [31:7] | - Reserved.   |
| [6]    | <b>ACMP0INV</b><br><b>Analog Comparator 0 output inverse select</b><br>1 = The comparator output inverse function Enabled.<br>0 = The comparator output inverse function Disabled   |
| [5]    | - Reserved.   |
| [4]    | <b>CN0</b><br><b>Analog Comparator 0 negative input select</b><br>1 = The internal comparator reference voltage (Vref=1.2V) is selected as the negative comparator input<br>0 = The comparator reference pin P8.3/CPN0 is selected as the negative comparator input |
| [3]    | <b>CP0</b><br><b>Analog Comparator 0 positive input select</b><br>1 = The internal OP amplifier 0 output is selected as the positive comparator input<br>0 = The comparator reference pin P8.4/CPN0 is selected as the positive comparator input                    |
| [2]    | <b>ACMP0_HYS_EN</b><br><b>CMP Hysteresis Enable</b><br>1 = Hysteresis function Enabled. The typical range is 20mV.<br>0 = Hysteresis function Disabled (Default)..  |
| [1]    | <b>ACMPIE0</b><br><b>Analog Comparator 0 Interrupt Enable</b><br>1 = Interrupt function Enabled<br>0 = Interrupt function Disabled  |

| Bits | Description |   |
|------|-------------|---|
| [0]  | ACMP0_EN    | <b>Analog Comparator 0 Enable</b><br>1 = Enabled<br>0 = Disabled<br>Comparator output need wait stable 10 us after ACMP0_EN is set. |

### Analog CMP 1 Control Register (ACMP1CR)

| Register | Offset       | R/W | Description                          | Reset Value |
|----------|--------------|-----|--------------------------------------|-------------|
| ACMP1CR  | ACMP_BA+0x04 | R/W | Analog Comparator 1 Control Register | 0x0000_0000 |

|    |          |    |     |     |              |         |          |
|----|----------|----|-----|-----|--------------|---------|----------|
| 31 | 30       | 29 | 28  | 27  | 26           | 25      | 24       |
| -  |          |    |     |     |              |         |          |
| 23 | 22       | 21 | 20  | 19  | 18           | 17      | 16       |
| -  |          |    |     |     |              |         |          |
| 15 | 14       | 13 | 12  | 11  | 10           | 9       | 8        |
| -  |          |    |     |     |              |         |          |
| 7  | 6        | 5  | 4   | 3   | 2            | 1       | 0        |
| -  | ACMP1INV | -  | CN1 | CP1 | ACMP1_HYS_EN | ACMPIE1 | ACMP1_EN |

| Bits   | Description |   |
|--------|-------------|---|
| [31:7] | -           | Reserved.   |
| [6]    | ACMP1INV    | <b>Analog Comparator 1 output inverse select</b><br>1 = The comparator output inverse function Enabled.<br>0 = The comparator output inverse function Disabled  |
| [5]    | -           | Reserved.   |
| [4]    | CN1         | <b>Analog Comparator 1 negative input select</b><br>1 = The internal comparator reference voltage (Vref=1.2V) is selected as the negative comparator input<br>0 = The comparator reference pin P6.4/CPN1 is selected as the negative comparator input |
| [3]    | CP1         | <b>Analog Comparator 1 positive input select</b><br>1 = The internal OP amplifier 1 output is selected as the positive comparator input<br>0 = The comparator reference pin P6.5/CP1 is selected as the positive comparator input                     |

| Bits | Description  |   |
|------|--------------|---|
| [2]  | ACMP1_HYS_EN | <b>CMP Hysteresis Enable</b><br>1 = Hysteresis function Enabled. The typical range is 20mV.<br>0 = Hysteresis function Disabled (Default).. |
| [1]  | ACMPIE1      | <b>Analog Comparator 1 Interrupt Enable</b><br>1 = Interrupt function Enabled<br>0 = Interrupt function Disabled                            |
| [0]  | ACMP1_EN     | <b>Analog Comparator 1 Enable</b><br>1 = Enabled<br>0 = Disabled<br>Comparator output need wait stable 10 us after ACMP1_EN is set.         |

#### Analog CMP 2 Control Register (ACMP2CR)

| Register | Offset       | R/W | Description                          | Reset Value |
|----------|--------------|-----|--------------------------------------|-------------|
| ACMP2CR  | ACMP_BA+0x08 | R/W | Analog Comparator 2 Control Register | 0x0000_0000 |

|    |          |    |     |    |              |         |          |
|----|----------|----|-----|----|--------------|---------|----------|
| 31 | 30       | 29 | 28  | 27 | 26           | 25      | 24       |
| -  |          |    |     |    |              |         |          |
| 23 | 22       | 21 | 20  | 19 | 18           | 17      | 16       |
| -  |          |    |     |    |              |         |          |
| 15 | 14       | 13 | 12  | 11 | 10           | 9       | 8        |
| -  |          |    |     |    |              |         |          |
| 7  | 6        | 5  | 4   | 3  | 2            | 1       | 0        |
| -  | ACMP2INV | -  | CN2 | -  | ACMP2_HYS_EN | ACMPIE2 | ACMP2_EN |

| Bits   | Description |  |
|--------|-------------|--|
| [31:7] | -           | Reserved.  |
| [6]    | ACMP2INV    | <b>Analog Comparator 2 output inverse select</b><br>1 = The comparator output inverse function Enabled.<br>0 = The comparator output inverse function Disabled |
| [5]    | -           | Reserved.  |

| Bits | Description         |   |
|------|---------------------|---|
| [4]  | <b>CN2</b>          | <b>Analog Comparator 2 negative input select</b><br>1 = The internal comparator reference voltage (Vref=1.2V) is selected as the negative comparator input<br>0 = The comparator reference pin P7.4/CPN2 is selected as the negative comparator input |
| [3]  | -                   | <b>Reserved.</b>  |
| [2]  | <b>ACMP2_HYS_EN</b> | <b>CMP Hysteresis Enable</b><br>1 = Hysteresis function Enabled. The typical range is 20mV.<br>0 = Hysteresis function Disabled (Default)..   |
| [1]  | <b>ACMPIE2</b>      | <b>Analog Comparator 2 Interrupt Enable</b><br>1 = Interrupt function Enabled<br>0 = Interrupt function Disabled  |
| [0]  | <b>ACMP2_EN</b>     | <b>Analog Comparator 2 Enable</b><br>1 = Enabled<br>0 = Disabled<br>Comparator output need wait stable 10 us after ACMP2_EN is first set.   |



Analog CMP Status Register (ACMPSR)

| Register | Offset       | R/W | Description                       | Reset Value |
|----------|--------------|-----|-----------------------------------|-------------|
| ACMPSR   | ACMP_BA+0x0C | R/W | Analog Comparator Status Register | 0x0000_0000 |

|    |    |       |       |    |        |        |        |
|----|----|-------|-------|----|--------|--------|--------|
| 31 | 30 | 29    | 28    | 27 | 26     | 25     | 24     |
| -  |    |       |       |    |        |        |        |
| 23 | 22 | 21    | 20    | 19 | 18     | 17     | 16     |
| -  |    |       |       |    |        |        |        |
| 15 | 14 | 13    | 12    | 11 | 10     | 9      | 8      |
| -  |    |       |       |    | CO2    | CO1    | CO0    |
| 7  | 6  | 5     | 4     | 3  | 2      | 1      | 0      |
| -  |    | OPDF1 | OPDF0 |    | ACMPF2 | ACMPF1 | ACMPF0 |

| Bits    | Description |   |
|---------|-------------|---|
| [31:11] | -           | Reserved.   |
| [10]    | CO2         | <b>Compare 2 Output</b><br>Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (ACMP2_EN = 0).  |
| [9]     | CO1         | <b>Compare 1 Output</b><br>Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (ACMP1_EN = 0).  |
| [8]     | CO0         | <b>Compare 0 Output</b><br>Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (ACMP0_EN = 0).  |
| [7:6]   | -           | Reserved.   |
| [5]     | OPDF1       | <b>OP Amplifier 1 Schmitt Trigger Digital Output Interrupt Flag</b><br>OPDF1 interrupt flag is set by hardware whenever the OP amplifier 1 Schmitt trigger non-inverting buffer digital output changes state. <b>Note: This bit is a copy of OPASR[5] and writing 1 to either ACMPSR[5] or OPASR[5] can clear this bit.</b> |
| [4]     | OPDF0       | <b>OP Amplifier 0 Schmitt Trigger Digital Output Interrupt Flag</b><br>OPDF0 interrupt flag is set by hardware whenever the OP amplifier 0 Schmitt trigger non-inverting buffer digital output changes state. <b>Note: This bit is a copy of OPASR[4] and writing 1 to either ACMPSR[4] or OPASR[4] can clear this bit.</b> |
| [3]     |             | Reserved  |
| [2]     | ACMPF2      | <b>Compare 2 Flag</b><br>This bit is set by hardware whenever the comparator 2 output changes state. This will cause an interrupt if ACMPCR2[1] is set to 1.<br><br>Write 1 to clear this bit to 0.   |

|     |               |   |
|-----|---------------|---|
| [1] | <b>ACMPF1</b> | <b>Compare 1 Flag</b><br>This bit is set by hardware whenever the comparator 1 output changes state. This will cause an interrupt if ACMPCR1[1] is set to 1.<br>Write 1 to clear this bit to 0. |
| [0] | <b>ACMPF0</b> | <b>Compare 0 Flag</b><br>This bit is set by hardware whenever the comparator 0 output changes state. This will cause an interrupt if ACMPCR0[1] is set to 1.<br>Write 1 to clear this bit to 0. |

## 25 OP AMPLIFIER

### 25.1 Overview

This device integrated two operational amplifiers. It can be enabled through OPx\_EN bit. User can measure the outputs of the OP amplifier as the OP amplifier output to the integrated A/D converter channel AINA[0] and AINB[0], where digital results can be taken.

**Note:** The analog input port pins must be configured as input type before the OP amplifier function is enabled.

### 25.2 Features

- Analog input voltage range: 0~Vdd.
- Two analog OP amplifiers.
- Software enabled to connect OP amplifier 0,1 outputs to A/D converter channel AINA[0], AINB[0] respectively.
- Schmitt trigger buffer outputs of OP amplifier 0, 1 can be one of the comparator interrupt sources.
- OP amplifier 0 output can be an optional input source of integrated comparator 0 positive input.
- OP amplifier 1 output can be an optional input source of integrated comparator 1 positive input.

### 25.3 Block Diagram

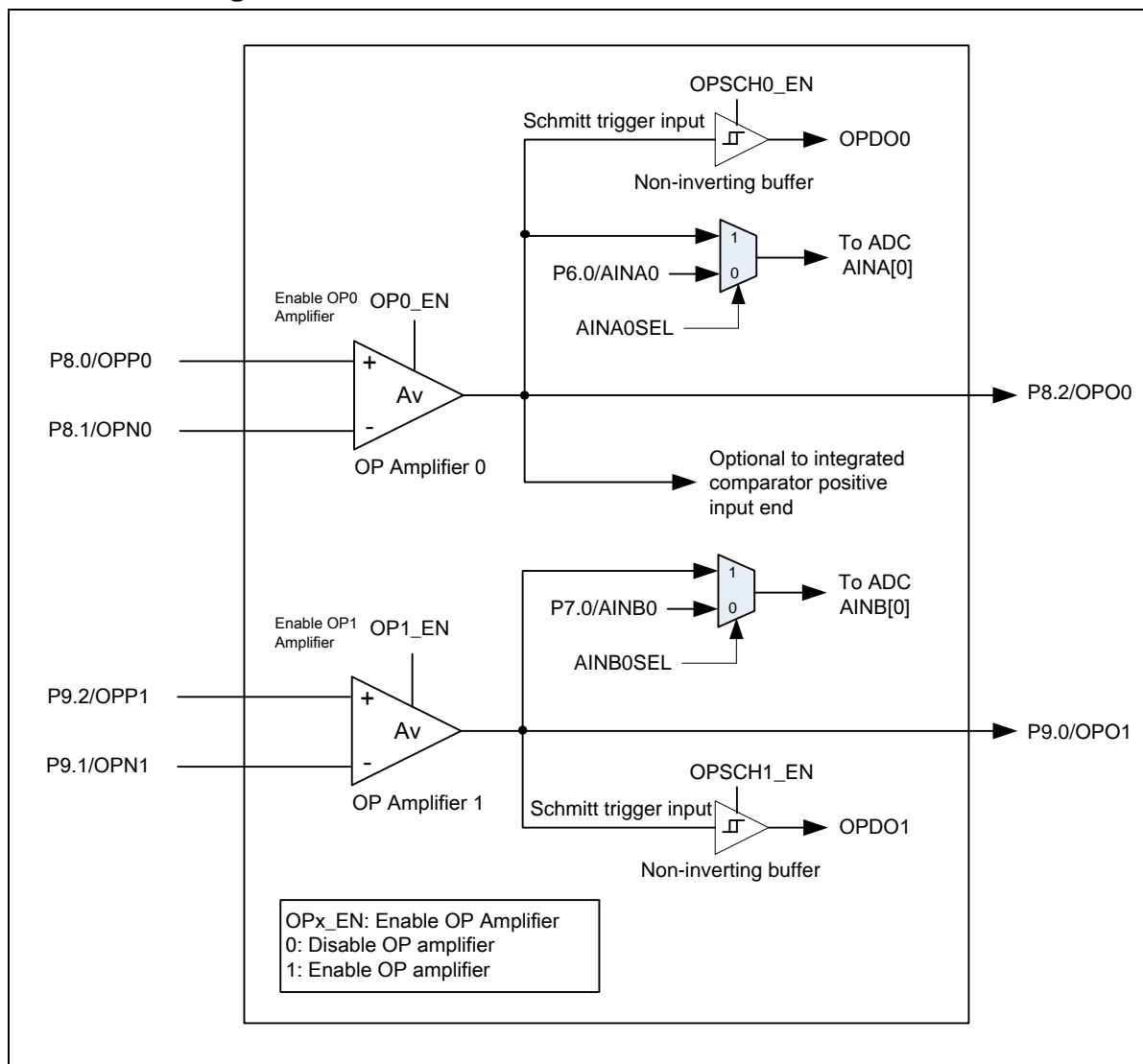


Figure 25-1 OP Amplifier Block Diagram

## 25.4 Interrupt Sources

The OPDF0, OPDF1 interrupt flags are set respectively by hardware whenever the OP0, 1 amplifier Schmitt trigger non-inverting buffer output change states. The flag bit is cleared by writing 1 to itself. Schmitt trigger buffer outputs of the OP amplifier0, 1 can be one of the comparator interrupt sources.

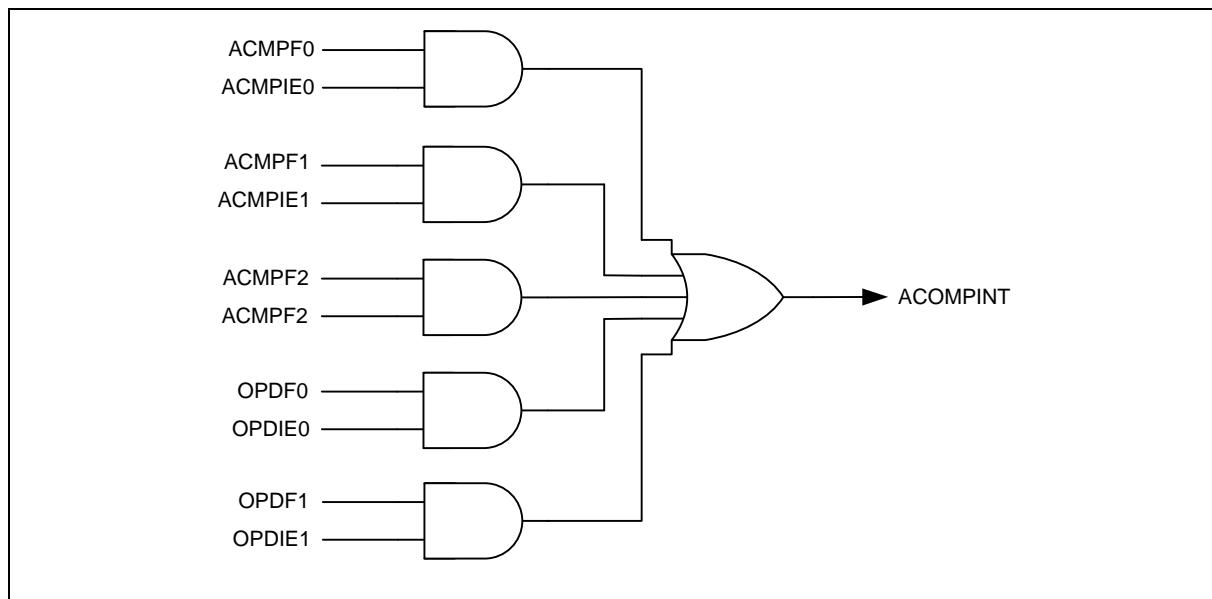


Figure 25–2 OP Amplifier Interrupt Flags for Analog Comparator Interrupt

### 25.5 Register Map

R: read only, W: write only, R/W: both read and write, C: Only value 0 can be written

| Register                                  | Offset      | R/W | Description                   | Reset Value |
|---|-------------|-----|-------------------------------|-------------|
| OPA Base Address:<br>OPA_BA = 0x400F_0000 |             |     |                               |             |
| OPACR                                     | OPA_BA+0x00 | R/W | OP Amplifier Control Register | 0x0000_0000 |
| OPASR                                     | OPA_BA+0x04 | R/W | OP Amplifier Status Register  | 0x0000_0000 |

## 25.6 Register Description

### OPA Control Register (OPACR)

| Register | Offset      | R/W | Description                   | Reset Value |
|----------|-------------|-----|-------------------------------|-------------|
| OPACR    | OPA_BA+0x00 | R/W | OP Amplifier Control Register | 0x0000_0000 |

|    |    |           |           |    |    |        |        |
|----|----|-----------|-----------|----|----|--------|--------|
| 31 | 30 | 29        | 28        | 27 | 26 | 25     | 24     |
| -  |    |           |           |    |    |        |        |
| 23 | 22 | 21        | 20        | 19 | 18 | 17     | 16     |
| -  |    |           |           |    |    |        |        |
| 15 | 14 | 13        | 12        | 11 | 10 | 9      | 8      |
| -  |    |           |           |    |    | OPDIE1 | OPDIE0 |
| 7  | 6  | 5         | 4         | 3  | 2  | 1      | 0      |
| -  | -  | OPSCH1_EN | OPSCH0_EN | -  | -  | OP1_EN | OP0_EN |

| Bits    | Description  |
|---------|--|
| [31:10] | - Reserved.  |
| [9]     | <b>OP Amplifier 1 Schmitt Trigger Digital Output Interrupt Enable</b><br>1 = OP Amplifier 1 digital output interrupt function Enabled.<br>0 = OP Amplifier 1 digital output interrupt function Disabled.<br>The OPDF1 interrupt flag is set by hardware whenever the OP amplifier 1 Schmitt trigger non-inverting buffer digital output changes state, in the meanwhile, if OPDIE1 is set to 1, a comparator interrupt request is generated. |
| [8]     | <b>OP Amplifier 0 Schmitt Trigger Digital Output Interrupt Enable</b><br>1 = OP Amplifier 0 digital output interrupt function Enabled.<br>0 = OP Amplifier 0 digital output interrupt function Disabled.<br>The OPDF0 interrupt flag is set by hardware whenever the OP amplifier 0 Schmitt trigger non-inverting buffer digital output changes state, in the meanwhile, if OPDIE0 is set to 1, a comparator interrupt request is generated. |
| [7:6]   | - Reserved.  |
| [5]     | <b>OP Amplifier 1 Schmitt Trigger Non-inverting Buffer Enable</b><br>1 = Enabled<br>0 = Disabled   |
| [4]     | <b>OP Amplifier 0 Schmitt Trigger Non-inverting Buffer Enable</b><br>1 = Enabled<br>0 = Disabled   |
| [3:2]   | - Reserved.  |

| Bits | Description   |   |
|------|---------------|---|
| [1]  | <b>OP1_EN</b> | <b>OP Amplifier 1 Enable</b><br>1 = OP Amplifier 1 Enabled.<br>0 = OP Amplifier 1 Disabled.<br>The OP Amplifier 1 output needs to wait stable 20μs after OP1_EN is first set. |
| [0]  | <b>OP0_EN</b> | <b>OP Amplifier 0 Enable</b><br>1 = OP Amplifier 0 Enabled<br>0 = OP Amplifier 0 Disabled<br>The OP Amplifier 0 output needs to wait stable 20μs after OP0_EN is first set.   |



### OPA Status Register (OPASR)

| Register | Offset      | R/W | Description                  | Reset Value |
|----------|-------------|-----|------------------------------|-------------|
| OPASR    | OPA_BA+0x04 | R/W | OP Amplifier Status Register | 0x0000_0000 |

|    |    |       |       |    |    |       |       |
|----|----|-------|-------|----|----|-------|-------|
| 31 | 30 | 29    | 28    | 27 | 26 | 25    | 24    |
| -  |    |       |       |    |    |       |       |
| 23 | 22 | 21    | 20    | 19 | 18 | 17    | 16    |
| -  |    |       |       |    |    |       |       |
| 15 | 14 | 13    | 12    | 11 | 10 | 9     | 8     |
| -  |    |       |       |    |    |       |       |
| 7  | 6  | 5     | 4     | 3  | 2  | 1     | 0     |
| -  |    | OPDF1 | OPDF0 | -  |    | OPDO1 | OPDO0 |

| Bits   | Description |  |
|--------|-------------|--|
| [31:6] | -           | Reserved.  |
| [5]    | OPDF1       | <b>OP Amplifier 1 Schmitt Trigger Digital Output Interrupt Flag</b><br>OPDF1 interrupt flag is set by hardware whenever the OP amplifier 1 Schmitt trigger non-inverting buffer digital output changes state. <b>This bit is cleared by writing 1 to itself.</b> |
| [4]    | OPDF0       | <b>OP Amplifier 0 Schmitt Trigger Digital Output Interrupt Flag</b><br>OPDF0 interrupt flag is set by hardware whenever the OP amplifier 0 Schmitt trigger non-inverting buffer digital output changes state. <b>This bit is cleared by writing 1 to itself.</b> |
| [3:2]  | -           | Reserved.  |
| [1]    | OPDO1       | <b>OP Amplifier 1 Digital Output</b><br>Synchronized to the APB clock to allow reading by software. Cleared when the Schmitt trigger buffer is disabled (OPSCH1_EN = 0).   |
| [0]    | OPDO0       | <b>OP Amplifier 0 Digital Output</b><br>Synchronized to the APB clock to allow reading by software. Cleared when the Schmitt trigger buffer is disabled (OPSCH0_EN = 0).   |

## 26 FLASH MEMORY CONTROL (FMC)

### 26.1 Overview

The NuMicro™ NM15xx Series has 128K/64K/32K bytes on-chip embedded Flash for application program memory (APROM) that can be updated through ISP procedure. The In-System-Programming (ISP) function enables user to update program memory when chip is soldered on PCB. After chip is powered on, Cortex™-M0 CPU fetches code from APROM or LDROM decided by boot select (CBS) in Config0. By the way, the NuMicro™ NM15xx Series also provides additional DATA Flash for user to store some application dependent data. For 128K bytes APROM device, the data flash is shared with original 128K program memory and its start address is configurable in Config1. For 64K/32K bytes APROM device, the data flash is fixed at 4K.

### 26.2 Features

- Runs up to 72 MHz and optional up to 50 MHz with zero wait state for continuous address read access
- All embedded flash memory supports 512 bytes page erase
- 128K/64K/32KB application program memory (APROM)
- 8 KB In-System-Programming (ISP) loader program memory (LDROM)
- 4KB data flash for 64K/32KB APROM device
- Configurable data flash size for 128KB APROM device
- Configurable or fixed 4 KB data flash with 512 bytes page erase unit
- Support In-Application-Programming (IAP) to switch code between APROM and LDROM without reset
- In-System-Programming (ISP) to update on-chip Flash

### 26.3 Block Diagram

The flash memory controller consist of AHB slave interface, ISP control logic, writer interface and flash macro interface timing control logic. The block diagram of flash memory controller is shown as follows:

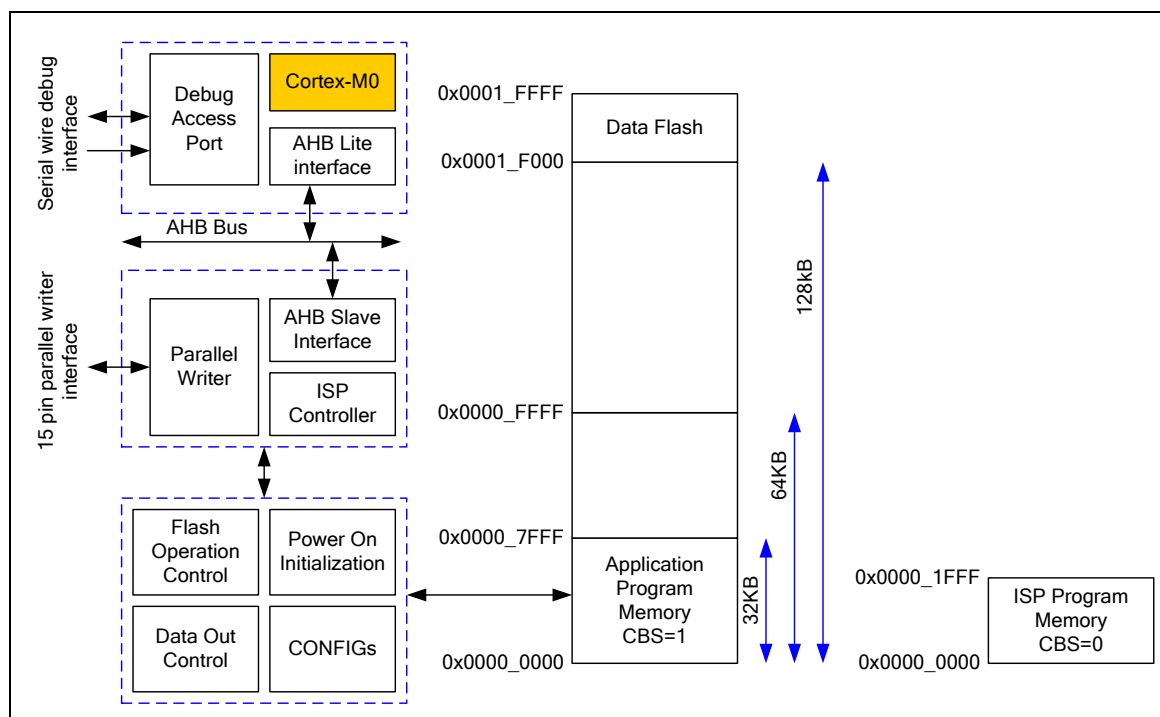


Figure 26–1 Flash Memory Control Block Diagram

## 26.4 Functional Description

### 26.4.1 Flash Memory Organization

The NuMicro™ NM15xx Series flash memory consists of program memory (APROM), data flash, ISP loader program memory (LDROM), and user configuration.

Program memory is main memory for user applications and called APROM. User can write their application to APROM and set system to boot from APROM.

ISP loader program memory is designed for a loader to implement In-System-Programming function. LDROM is independent to APROM and system can also be set to boot from LDROM. Therefore, user can use LDROM to avoid system boot fail when code of APROM was corrupted.

Data flash is used for user to store data. It can be read by ISP read or memory read and programmed through ISP procedure. The size of each erase unit is 512 bytes. For 128 KB APROM device, the data flash and application program share the same 128 KB memory, if DFEN (Data Flash Enable) bit in Config0 is enabled, the data flash base address is defined by DFBADR and its size is (0x20000 - DFBADR). At the same time, the APROM size will be (128 KB – data flash size). For 64/32KB APROM devices, data flash size is always 4 KB and start address is fixed at 0x0001\_F000.

User configuration provides several bytes to control system logic, such as flash security lock, boot select, Brown-out voltage level, data flash base address, etc.... User configuration works like a fuse for power on setting and loaded from flash memory to its corresponding control registers during chip powered on.

In NuMicro™ Family, the flash memory organization is different to system memory map. Flash memory organization is used when user using ISP command to read, program or erase flash memory. System memory map is used when CPU access flash memory to fetch code or data. For example, When system is set to boot from LDROM by CBS = 01b, CPU will be able to fetch code of LDROM from 0x0 ~ 0x1FFF. However, if user want to read LDROM by ISP, they still need to read the address of LDROM as 0x0010\_0000 ~ 0x0010\_1FFF.

Table 26-1 and Figure 26-2 show the address mapping information of APROM, LDROM, data flash and user configuration for 32/64KB and 128 KB devices.

| Block Name         | Device Type  | Size               |                          | Start Address | End Address                   |
|--------------------|--------------|--------------------|--------------------------|---------------|-------------------------------|
| APROM              | 32 KB        | 32 KB              |                          | 0x0000_0000   | 0x0000_7FFF                   |
|                    | 64 KB        | 64 KB              |                          | 0x0000_0000   | 0x0000_FFFF                   |
|                    | 128 KB       | Data Flash Enable  | 128 KB - Data Flash Size | 0x0000_0000   | 0x20000 – Data Flash Size - 1 |
|                    |              | Data Flash Disable | 128 KB                   | 0x0000_0000   | 0x0001_FFFF                   |
| Data Flash         | 32 KB        | 4 KB               |                          | 0x0001_F000   | 0x0001_FFFF                   |
|                    | 64 KB        | 4 KB               |                          | 0x0001_F000   |                               |
|                    | 128 KB       | Data Flash Enable  | 0x20000-DFBADR           | DFBADR        |                               |
|                    |              | Data Flash Disable | 0 KB                     | N/A           | N/A                           |
| LDROM              | 32/64/128 KB | 8 KB               |                          | 0x0010_0000   | 0x0010_1FFF                   |
| User Configuration | 32/64/128 KB | 2 words            |                          | 0x0030_0000   | 0x0030_0004                   |

Table 26-1 Memory Address Map

The Flash memory organization is shown as Figure 26-2:

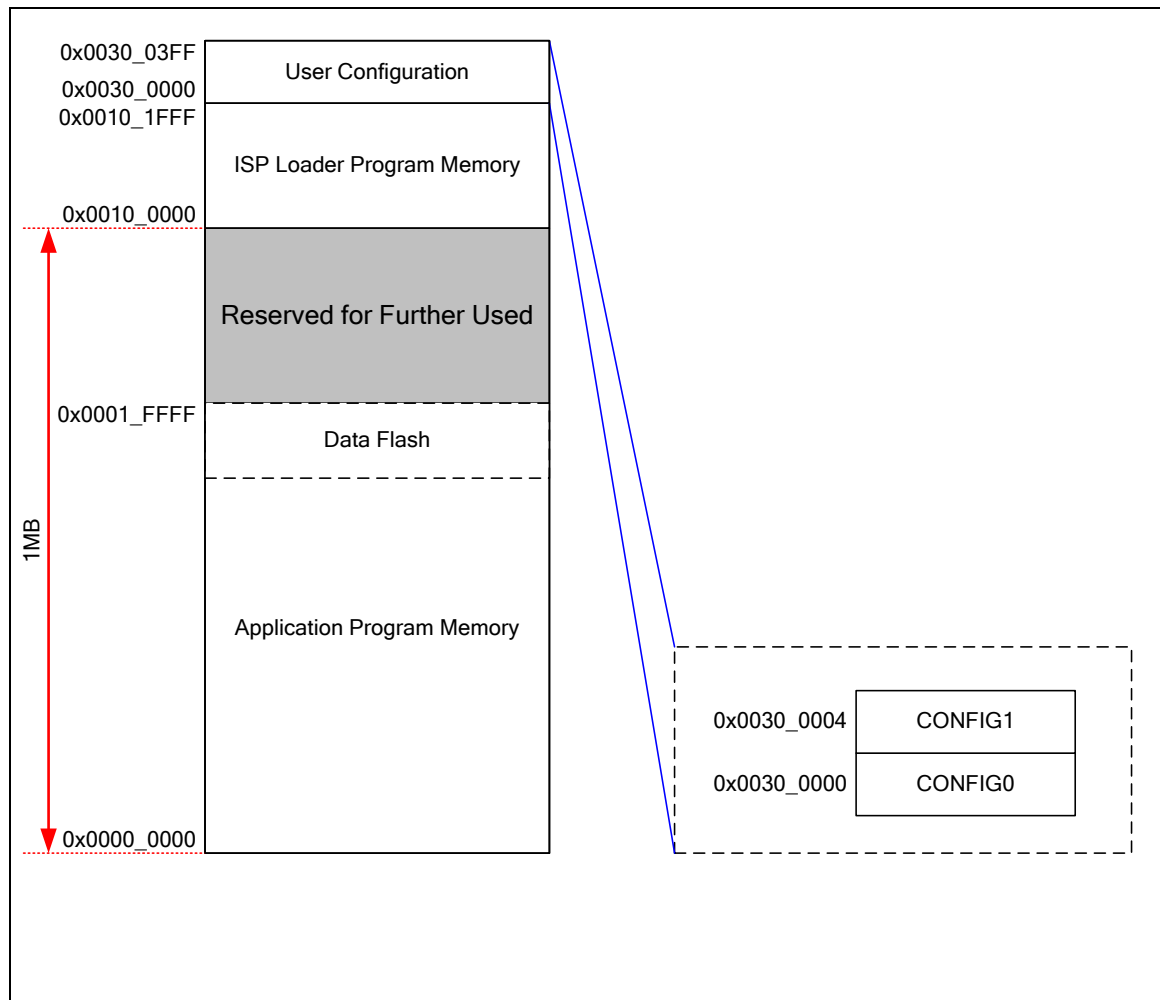


Figure 26-2 Flash Memory Organization

### 26.4.2 User Configuration

User configuration is internal programmable configuration area for boot options. The user configuration is located at 0x300000 of Flash Memory Organization and they are two 32 bits words. Any change on user configuration will take effect after system reboot.

### Config0 (Address = 0x0030 0000)

| 31       | 30       | 29       | 28       | 27       | 26        | 25       | 24        |
|----------|----------|----------|----------|----------|-----------|----------|-----------|
| CWDTEN   | CWDTPDEN | Reserved |          |          | CFOSC     |          |           |
| 23       | 22       | 21       | 20       | 19       | 18        | 17       | 16        |
| CBODEN   | CBOV1    | CBOV0    | CBORST   | Reserved |           |          |           |
| 15       | 14       | 13       | 12       | 11       | 10        | 9        | 8         |
| Reserved |          |          | CHZ_BPWM | CHZ_Odd1 | CHZ_Even1 | CHZ_Odd0 | CHZ_Even0 |
| 7        | 6        | 5        | 4        | 3        | 2         | 1        | 0         |
| CBS      |          | Reserved |          |          |           | LOCK     | DFEN      |

| Config0 Bits | Description   |   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
|--------------|---|---|--|-----------|--------------|-------------------|---|-----|---|--------|----------|-------|---|---|-------|---|---|-------|
| [31]         | CWDTEN  | <b>Watchdog Enable</b><br>0 = Watchdog Timer Enabled and force Watchdog Timer clock source as OSC10K after chip powered on.<br>1 = Watchdog Timer Disabled after chip powered on.   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| [30]         | CWDTPDEN  | <b>Watchdog Clock Power Down Enable</b><br>0 = OSC10K Watchdog Timer clock source is forced to be always enabled.<br>1 = OSC10K Watchdog Timer clock source is controlled by OSC10K_EN (PWRCON[3]) when chip enters power down.<br><b>Note:</b> This bit only works at CWDTEN is set to 0   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| [29:27]      | Reserved  | Reserved  |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| [26:24]      | CFOSC   | <b>CPU Clock Source Selection After Reset</b> <table><tr><th>FOSC[2:0]</th><th>Clock Source</th></tr><tr><td>000</td><td>External 4~24 MHz high speed crystal oscillator clock</td></tr><tr><td>111</td><td>Internal RC 22.1184 MHz high speed oscillator clock</td></tr><tr><td>Others</td><td>Reserved</td></tr></table> <p>The value of CFOSC will be load to CLKSEL0.HCLK_S[2:0] in system register after any reset occurs.</p> |  | FOSC[2:0] | Clock Source | 000               | External 4~24 MHz high speed crystal oscillator clock | 111 | Internal RC 22.1184 MHz high speed oscillator clock | Others | Reserved |       |   |   |       |   |   |       |
| FOSC[2:0]    | Clock Source  |   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| 000          | External 4~24 MHz high speed crystal oscillator clock |   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| 111          | Internal RC 22.1184 MHz high speed oscillator clock   |   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| Others       | Reserved  |   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| [23]         | CBODEN  | <b>Brown-out Detector Enable</b><br>0= Brown-out detect Enabled after powered on.<br>1= Brown-out detect Disabled after powered on.   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| [22:21]      | CBOV1-0   | <b>Brown-out Voltage Selection</b> <table><tr><th>CBOV1</th><th>CBOV0</th><th>Brown-out Voltage</th></tr><tr><td>1</td><td>1</td><td>4.4 V</td></tr><tr><td>1</td><td>0</td><td>3.7 V</td></tr><tr><td>0</td><td>1</td><td>2.7 V</td></tr><tr><td>0</td><td>0</td><td>2.2 V</td></tr></table>   |  | CBOV1     | CBOV0        | Brown-out Voltage | 1   | 1   | 4.4 V   | 1      | 0        | 3.7 V | 0 | 1 | 2.7 V | 0 | 0 | 2.2 V |
| CBOV1        | CBOV0   | Brown-out Voltage   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| 1            | 1   | 4.4 V   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| 1            | 0   | 3.7 V   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| 0            | 1   | 2.7 V   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |
| 0            | 0   | 2.2 V   |  |           |              |                   |   |     |   |        |          |       |   |   |       |   |   |       |



|         |                  |  |
|---------|------------------|--|
| [20]    | <b>CBORST</b>    | <b>Brown-out Reset Enable</b><br>0 = Brown-out reset Enabled after powered on<br>1 = Brown-out reset Disabled after powered on   |
| [19:13] | <b>Reserved</b>  | Reserved   |
| [12]    | <b>CHZ_BPWM</b>  | <b>Basic PWM Ports Tri-state Driving Control</b><br>0 = Basic PWM ports driving mode is controlled by GOPIO mode registers (GPIO_PMD).<br>1 = Basic PWM ports driving mode is forced in tri-state all the time.<br>This bit will be load to bit PWMPOEN.HZ_BPWM after any reset.                 |
| [11]    | <b>CHZ_Odd1</b>  | <b>PWM Unit1 odd Ports Tri-state Driving Control</b><br>0 = PWM unit1 odd ports driving mode is controlled by GOPIO mode registers (GPIO_PMD).<br>1 = PWM unit1 odd ports driving mode is forced in tri-state all the time.<br>This bit will be load to bit PWMPOEN.HZ_Odd1 after any reset.     |
| [10]    | <b>CHZ_Even1</b> | <b>PWM Unit0 Even Ports Tri-state Driving Control</b><br>0 = PWM unit1 even ports driving mode is controlled by GOPIO mode registers (GPIO_PMD).<br>1 = PWM unit1 even ports driving mode is forced in tri-state all the time.<br>This bit will be load to bit PWMPOEN.HZ_Even1 after any reset. |
| [9]     | <b>CHZ_Odd0</b>  | <b>PWM Unit0 Odd Ports Tri-state Driving Control</b><br>0 = PWM unit0 odd ports driving mode is controlled by GOPIO mode registers (GPIO_PMD).<br>1 = PWM unit0 odd ports driving mode is forced in tri-state all the time.<br>This bit will be load to bit PWMPOEN.HZ_Odd0 after any reset.     |
| [8]     | <b>CHZ_Even0</b> | <b>PWM Unit0 Even Ports Tri-state Driving Control</b><br>0 = PWM unit0 even ports driving mode is controlled by GOPIO mode registers (GPIO_PMD).<br>1 = PWM unit0 even ports driving mode is forced in tri-state all the time.<br>This bit will be load to bit PWMPOEN.HZ_Even0 after any reset. |

|       |                 |   |  |
|-------|-----------------|---|--|
| [7:6] | <b>CBS</b>      | <b>Chip Boot Selection</b>  |  |
|       |                 | <b>CBS[1:0]</b>   | <b>Boot Selection</b>  |
|       |                 | 11  | Chip booting from APROM and program executing range only including APROM. LDROM cannot be access by program directly, except by through ISP.<br>APROM is write-protected in this mode.   |
|       |                 | 01  | Chip booting from LDROM, program executing range only including LDROM; APROM cannot be access by program directly, except by through ISP.<br>LDROM is write-protected in this mode.  |
|       |                 | 10  | Chip booting from APROM, program executing range including LDROM and APROM<br>LDROM address is mapping to 0x0010_0000~0x0010_0FFF<br>The address 0x0000_0000 ~ 0x0000_01FF can be re-mapping to any other page within executing range though ISP command.  |
|       |                 | 00  | Chip booting from LDROM, program executing range including LDROM and most of APROM (all but except first 512 bytes, because the first 512 bytes is mapped from LDROM)<br>LDROM address is mapping to 0x0010_0000 ~ 0x0010_0FFF, and also the first 512 bytes of LDROM is mapping to the address 0x0000_0000 ~ 0x0000_01FF.<br>The address 0x0000_0000 ~ 0x0000_01FF can be re-mapping to any other page within executing range though ISP command. |
| [5:2] | <b>Reserved</b> | Reserved  |  |
| [1]   | <b>LOCK</b>     | <b>Security Lock</b><br>0 = Flash data is locked<br>1 = Flash data is not locked<br>When flash data is locked, only device ID, Config0 and Config1 can be read by writer and ICP through serial debug interface. Others data is locked as 0xFFFF_FFFF. ISP can read data anywhere regardless of LOCK bit value.<br>User need to erase whole chip by ICP/Writer tool or erase user configuration by ISP to unlock. |  |
| [0]   | <b>DFEN</b>     | <b>Data Flash Enable (Only for 128 KB APROM Device)</b><br>0 = Data flash Enabled.<br>1 = Data flash Disabled.  |  |

**Config1 (Address = 0x0030 0004)**

|           |           |           |           |           |           |           |           |
|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|
| 31        | 30        | 29        | 28        | 27        | 26        | 25        | 24        |
| Reserved  |           |           |           |           |           |           |           |
| 23        | 22        | 21        | 20        | 19        | 18        | 17        | 16        |
| Reserved  |           |           |           | DFBADR.19 | DFBADR.18 | DFBADR.17 | DFBADR.16 |
| 15        | 14        | 13        | 12        | 11        | 10        | 9         | 8         |
| DFBADR.15 | DFBADR.14 | DFBADR.13 | DFBADR.12 | DFBADR.11 | DFBADR.10 | DFBADR.9  | DFBADR.8  |
| 7         | 6         | 5         | 4         | 3         | 2         | 1         | 0         |
| DFBADR.7  | DFBADR.6  | DFBADR.5  | DFBADR.4  | DFBADR.3  | DFBADR.2  | DFBADR.1  | DFBADR.0  |

| Config1 Bits | Description |  |
|--------------|-------------|--|
| [31:20]      | Reserved    | Reserved (It is mandatory to program 0x00 to these Reserved bits)  |
| [19:0]       | DFBADR      | <b>Data Flash Base Address (Only for 128 KB APROM Device)</b><br>For 128 KB APROM device, its data flash base address is defined by user. Since on-chip flash erase unit is 512 bytes, it is mandatory to keep bit 8-0 as 0. This configuration is only valid for 128 KB flash device. |

### 26.4.3 Boot Selection

The NuMicro™ NM15xx Series provides In-System-Programming (ISP) feature to enable user to update program memory by a stand-alone ISP firmware. A dedicated 8 KB program memory (LDROM) is used to store ISP firmware. Users can select to start program fetch from APROM or LDROM by CBS[1] in Config0.

In addition to set boot from APROM or LDROM, CBS in Config0 also used to control system memory map after booting. When CBS[0] = 1 and set CBS[1] = 1 to boot from APROM, the application in APROM will not be able to access LDROM by memory read. In other words, when CBS[0] = 1 and set CBS[1] = 0 to boot from LDROM, the software executed in LDROM will not be able to access APROM by memory read. Figure 26-3 shows the memory map when boot from APROM and LDROM.

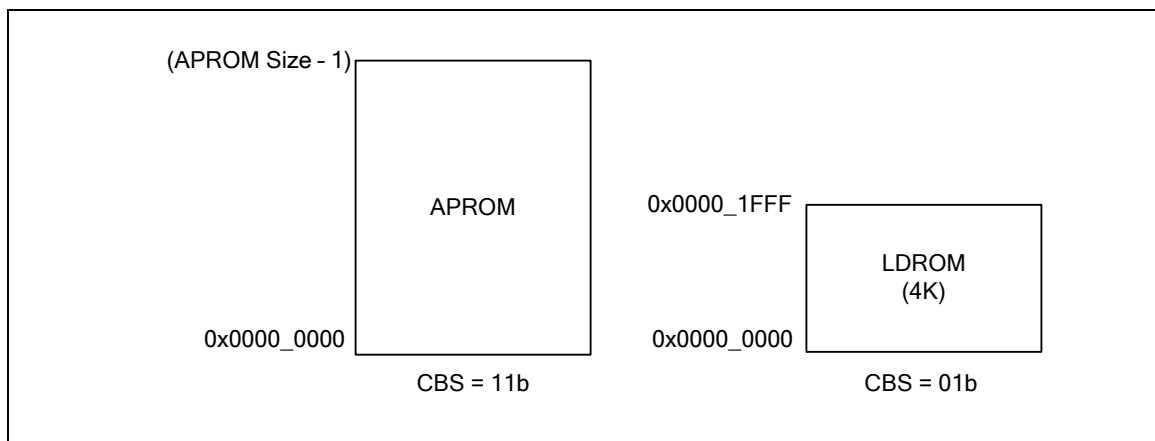


Figure 26-3 Program Executing Range for boot from APROM and boot from LDROM

For the application that software needs to execute code in APROM and call the functions in LDROM or to execute code in LDROM and call the APROM function without changing boot mode, CBS[0] needs to be set as 0 and this is called In-Application-Programming(IAP).

## 26.5 In-Application-Programming (IAP)

The NuMicro™NM15xx Series provides In-application-programming (IAP) function for user to switch the code executing between APROM and LDROM without a reset. User can enable the IAP function by re-booting chip and setting the chip boot selection bits in Config0 (CBS[1:0]) as 10b or 00b.

In the case that the chip boots from APROM with the IAP function enabled (CBS[1:0] = 10b), the executable range of code includes all of APROM and LDROM. The address space of APROM is kept as the original size but the address space of the 8 KB LDROM is mapped to 0x0010\_0000~0x0010\_1FFF.

In the case that the chip boots from LDROM with the IAP function enabled (CBS[1:0] = 00b), the executable range of code includes all of LDROM and almost all of APROM except for its first page. User cannot access the first page of APROM because the first page of executable code range becomes the mirror of the first page of LDROM as set by default. Meanwhile, the address space of 8 KB LDROM is mapped to 0x0010\_0000~0x0010\_1FFF.

Please refer to Figure 26-4 for the address map while IAP is activating.

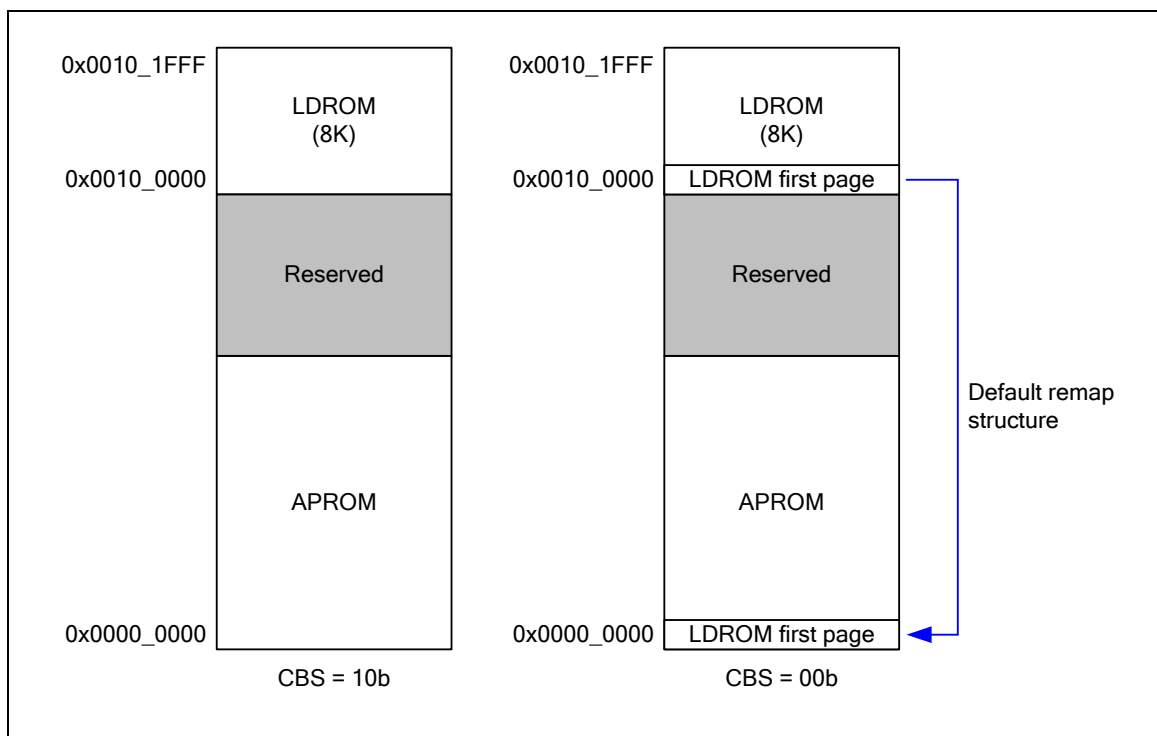


Figure 26-4 Executable Range of Code with IAP Function Enabled

When chip boots with the IAP function enabled, any other page within the executable range of code can be mirrored to the first page of executable code (0x0000\_0000~0x0000\_01FF) any time. User can change the remap address of the first executing page by filling the target remap address to ISPADR and then go through ISP procedure with the Vector Page Re-map command. After changing the remap address, user can check if the change is successful by reading the VECMAP field in the ISPSTA register.

## 26.6 In-System-Programming (ISP)

The NuMicro™ NM15xx Series supports ISP mode allowing a device to be reprogrammed under software control and avoid system fail risk when download or programming fail. Furthermore, the capability to update the application firmware makes a wide range of applications possible.

ISP provides the ability to update system firmware on board. Various peripheral interfaces let ISP loader in LDROM to receive new program code easily. The most common method to perform ISP is via UART along with the ISP loader in LDROM. General speaking, PC transfers the new APROM code through serial port. Then ISP loader receives it and re-programs into APROM through ISP commands.

### 26.6.1 ISP Procedure

The NuMicro™ NM15xx Series supports booting from APROM or LDROM initially defined by user configuration. The change of user configuration needs to reboot system to make it take effect. If user wants to switch between APROM or LDROM mode without changing user configuration, he needs to control BS bit of ISPCON control register, then reset CPU by IPRSTC1 control register. The boot switching flow by BS bit is as the following figure.

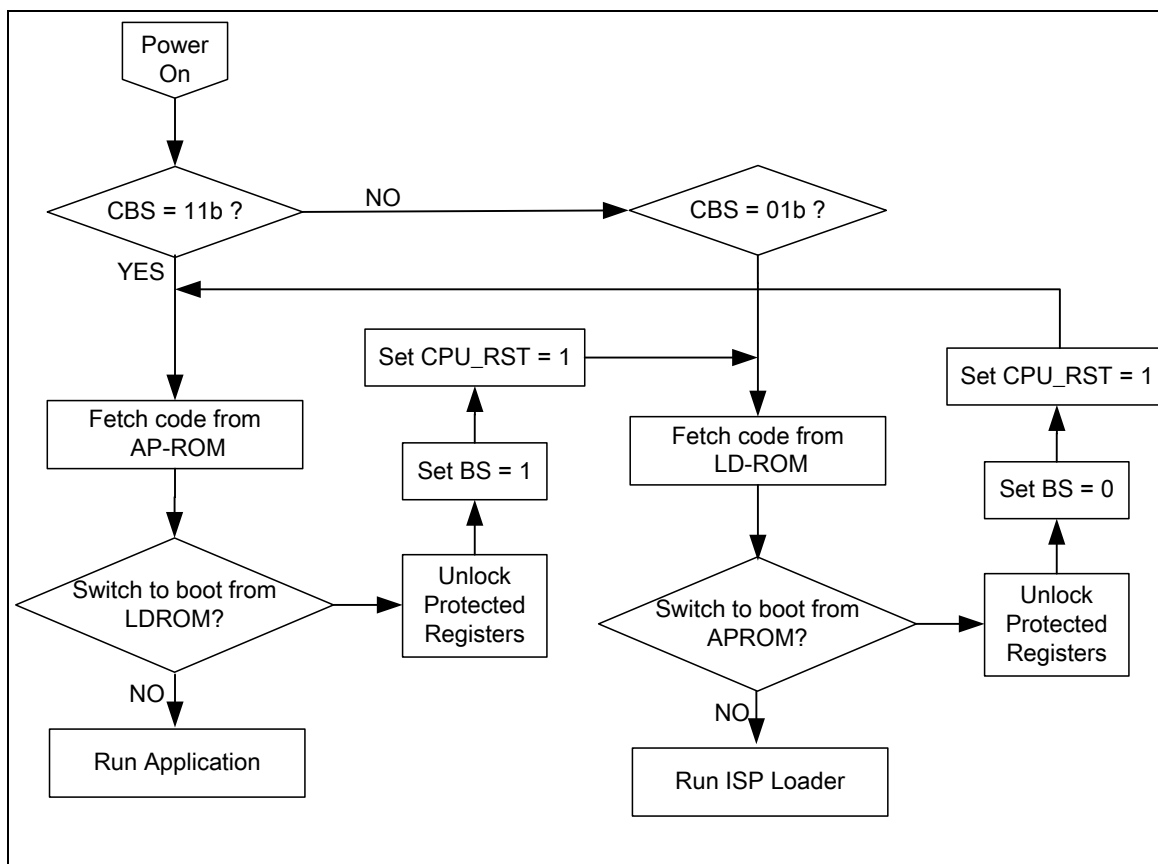


Figure 26-5 Example Flow of Boot Selection by BS Bit

Update APROM by the software in LDROM or update LDROM by the software in APROM can avoid system fail when update fails.

ISP controller supports to read, erase and program embedded flash memory. Several control bits of ISP controller are write-protected, thus it is necessary to unlock before we can setting them. To unlock protected register bits, software needs to write 0x59, 0x16 and 0x88 sequentially to REGWRPROT. If register unlock successfully, the value of REGWRPROT will be 1. Unlock sequence must not be interrupted by other access; otherwise it may fail to unlock.

After unlock protected register bits, user need to set ISPCON control register to decide to update LDROM, User Configuration, APROM and enable ISP controller.

Once ISPCON register is set properly, user can set ISPCMD for erase, read or program. Set ISPADR for target flash memory based on flash memory origination. ISPDAT can be used to set the data to program or used to return the read data according to ISPCMD.

Finally, we can set ISPGO bit of ISPTRG control register to perform relative ISP function. ISPGO bit is self-clear when ISP function has been done. In order to make sure ISP function has been finished before CPU go ahead, ISB instruction is used right after ISPGO setting.

Several error conditions are checked after ISP finished. If error condition occurs, ISP operation is not been started and ISP fail flag will be set instead of. ISPPF flag can only be cleared by software. The next ISP procedure can be started even ISPPF bit keeps at 1. Therefore, it is recommended to check the ISPPF bit and clear it after each ISP operation if it is set to 1.

When ISPGO bit is set, CPU will wait for ISP operation finish, during this period; peripheral still keeps working as usual. If any interrupt request occur, CPU will not service it till ISP operation finish. When ISP operation is finished, the ISPGO bit will be cleared by hardware automatically. User can check whether ISP operation is finished or not by ISPGO bit. User should add ISB instruction next to the instruction which set 1 to ISPGO bit to ensure correct execution of the instructions following ISP operation.

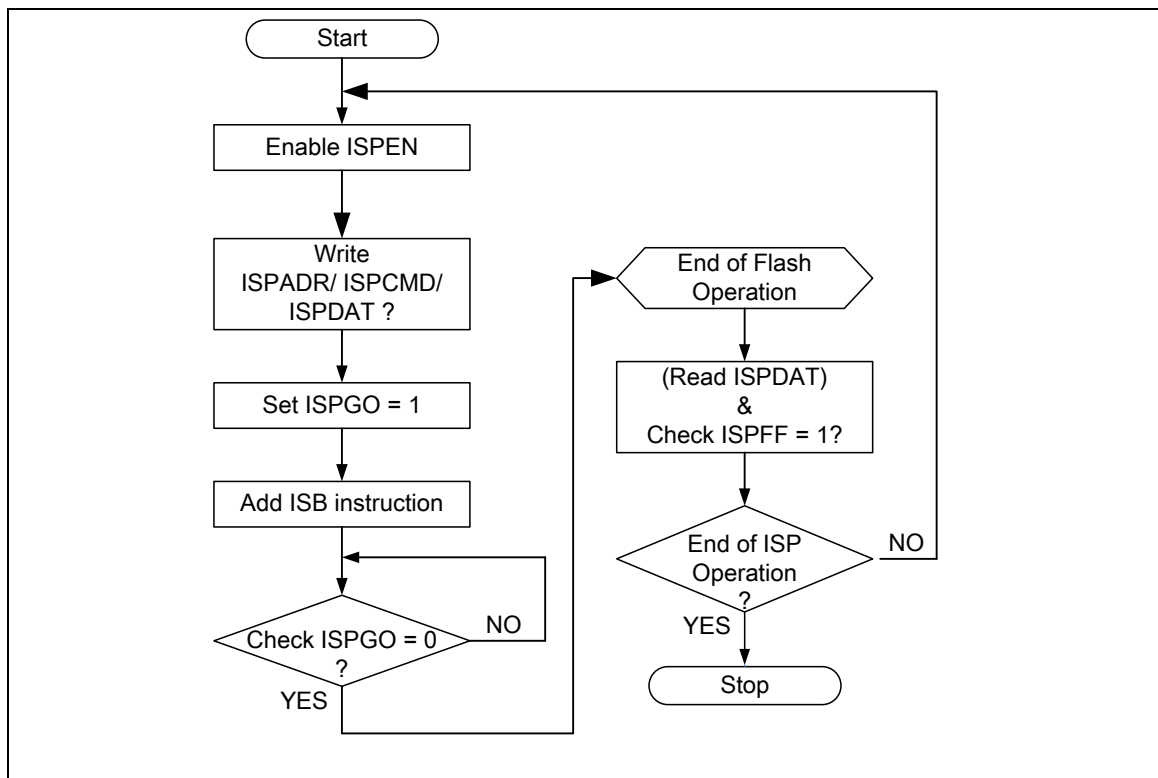


Figure 26-6 ISP Flow Example

| ISP Command        | ISPCMD | ISPADR   | ISPDAT           |
|--------------------|--------|--|------------------|
| FLASH Page Erase   | 0x22   | Valid address of flash memory origination.<br>It must be 512 bytes page alignment. | N/A              |
| FLASH Program      | 0x21   | Valid address of flash memory origination  | Programming Data |
| FLASH Read         | 0x00   | Valid address of flash memory origination  | Return Data      |
| Read Unique ID     | 0x04   | 0x0000_0000  | Unique ID Word 0 |
|                    |        | 0x0000_0004  | Unique ID Word 1 |
|                    |        | 0x0000_0008  | Unique ID Word 2 |
| Vector Page Re-Map | 0x2E   | Page in APROM or LDROM<br>It must be 512 bytes page alignment                      | N/A              |

Table 26-2 ISP Command List



### 26.7 Register Map

R: read only, W: write only, R/W: both read and write

| Register                    | Offset      | R/W | Description                        | Reset Value |
|-----------------------------|-------------|-----|------------------------------------|-------------|
| <b>FMC Base Address:</b>    |             |     |                                    |             |
| <b>FMC_BA = 0x5000_C000</b> |             |     |                                    |             |
| <b>ISPCON</b>               | FMC_BA+0x00 | R/W | ISP Control Register               | 0x0000_0000 |
| <b>ISPADR</b>               | FMC_BA+0x04 | R/W | ISP Address Register               | 0x0000_0000 |
| <b>ISPDAT</b>               | FMC_BA+0x08 | R/W | ISP Data Register                  | 0x0000_0000 |
| <b>ISPCMD</b>               | FMC_BA+0x0C | R/W | ISP Command Register               | 0x0000_0000 |
| <b>ISPTRG</b>               | FMC_BA+0x10 | R/W | ISP Trigger Control Register       | 0x0000_0000 |
| <b>DFBADR</b>               | FMC_BA+0x14 | R   | Data Flash Base Address            | 0x000X_XXXX |
| <b>FATCON</b>               | FMC_BA+0x18 | R/W | Flash Access Time Control Register | 0x0000_0000 |
| <b>ISPSTA</b>               | FMC_BA+0x40 | R/W | ISP Status Register                | 0x0000_0000 |

### 26.8 Register Description

#### ISP Control Register (ISPCON)

| Register | Offset      | R/W | Description          | Reset Value |
|----------|-------------|-----|----------------------|-------------|
| ISPCON   | FMC_BA+0x00 | R/W | ISP Control Register | 0x0000_0000 |

|          |       |       |        |       |          |    |       |
|----------|-------|-------|--------|-------|----------|----|-------|
| 31       | 30    | 29    | 28     | 27    | 26       | 25 | 24    |
| Reserved |       |       |        |       |          |    |       |
| 23       | 22    | 21    | 20     | 19    | 18       | 17 | 16    |
| Reserved |       |       |        |       |          |    |       |
| 15       | 14    | 13    | 12     | 11    | 10       | 9  | 8     |
| Reserved |       |       |        |       |          |    |       |
| 7        | 6     | 5     | 4      | 3     | 2        | 1  | 0     |
| Reserved | ISPPF | LDUEN | CFGUEN | APUEN | Reserved | BS | ISPEN |

| Bits   | Description  |
|--------|--|
| [31:7] | Reserved   |
| [6]    | <b>ISP Fail Flag (Write-protection Bit)</b><br>This bit is set by hardware when a triggered ISP meets any of the following conditions:<br>(1) APROM writes to itself if APUEN is set to 0<br>(2) LDROM writes to itself if LDUEN is set to 0<br>(3) CONFIG is erased/programmed if CFGUEN is set to 0<br>(4) Destination address is illegal, such as over an available range<br>Write 1 to clear to this bit to 0. |
| [5]    | <b>LDROM Update Enable (Write-protection Bit)</b><br>LDROM update enable bit.<br>1 = LDROM can be updated when chip runs in APROM.<br>0 = LDROM cannot be updated.   |
| [4]    | <b>Enable Config-bits Update by ISP (Write-protection Bit)</b><br>1 = ISP update config-bits Enabled.<br>0 = ISP update config-bits Disabled.  |
| [3]    | <b>APROM Update Enable (Write-protection Bit)</b><br>1 = APROM can be updated when chip runs in APROM.<br>0 = APROM cannot be updated when chip runs in APROM.   |
| [2]    | Reserved   |

|     |              |  |
|-----|--------------|--|
| [1] | <b>BS</b>    | <b>Boot Select (Write-protection Bit)</b><br>Set/clear this bit to select next booting from LDROM/APROM, respectively. This bit also functions as chip booting status flag, which can be used to check where chip booted from. This bit is initiated with the inversed value of CBS in Config0 after any reset is happened except CPU reset (RSTS_CPU is 1) or system reset (RSTS_SYS) is happened<br>1 = Boot from LDROM<br>0 = Boot from APROM |
| [0] | <b>ISPEN</b> | <b>ISP Enable (Write-protection Bit)</b><br>ISP function enable bit. Set this bit to enable ISP function.<br>1 = ISP function Enabled.<br>0 = ISP function Disabled.   |

### ISP Address (ISPADR)

| Register | Offset      | R/W | Description          | Reset Value |
|----------|-------------|-----|----------------------|-------------|
| ISPADR   | FMC_BA+0x04 | R/W | ISP Address Register | 0x0000_0000 |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| ISPADR[31:24] |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| ISPADR[23:16] |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| ISPADR[15:8]  |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| ISPADR[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description |  |
|--------|-------------|--|
| [31:0] | ISPADR      | <p><b>ISP Address</b></p> <p>The NuMicro™ NM15xx Series has a maximum 32Kx32 (128 KB) of embedded Flash, which supports word program only. ISPADR[1:0] must be kept 00b for ISP operation.</p> |

**ISP Data Register (ISPDAT)**

| Register | Offset      | R/W | Description       | Reset Value |
|----------|-------------|-----|-------------------|-------------|
| ISPDAT   | FMC_BA+0x08 | R/W | ISP Data Register | 0x0000_0000 |

|                |    |    |    |    |    |    |    |
|----------------|----|----|----|----|----|----|----|
| 31             | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| ISPDAT[31:24]  |    |    |    |    |    |    |    |
| 23             | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| ISPDAT [23:16] |    |    |    |    |    |    |    |
| 15             | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| ISPDAT [15:8]  |    |    |    |    |    |    |    |
| 7              | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| ISPDAT [7:0]   |    |    |    |    |    |    |    |

| Bits   | Description  |
|--------|--|
| [31:0] | <div>ISPDAT</div> <div>ISP Data</div> <div>Write data to this register before ISP program operation</div> <div>Read data from this register after ISP read operation</div> |

### ISP Command (ISPCMD)

| Register | Offset      | R/W | Description          | Reset Value |
|----------|-------------|-----|----------------------|-------------|
| ISPCMD   | FMC_BA+0x0C | R/W | ISP Command Register | 0x0000_0000 |

|          |    |        |    |    |    |    |    |
|----------|----|--------|----|----|----|----|----|
| 31       | 30 | 29     | 28 | 27 | 26 | 25 | 24 |
| Reserved |    |        |    |    |    |    |    |
| 23       | 22 | 21     | 20 | 19 | 18 | 17 | 16 |
| Reserved |    |        |    |    |    |    |    |
| 15       | 14 | 13     | 12 | 11 | 10 | 9  | 8  |
| Reserved |    |        |    |    |    |    |    |
| 7        | 6  | 5      | 4  | 3  | 2  | 1  | 0  |
| Reserved |    | ISPCMD |    |    |    |    |    |

| Bits       | Description |  |             |        |      |      |                    |      |         |      |            |      |
|------------|-------------|--|-------------|--------|------|------|--------------------|------|---------|------|------------|------|
| [31:6]     | Reserved    | Reserved   |             |        |      |      |                    |      |         |      |            |      |
| [5:0]      | ISPCMD      | <b>ISP Command</b><br>ISP command table is shown below:  |             |        |      |      |                    |      |         |      |            |      |
|            |             | <table><tr><th>ISP command</th><th>ISPCMD</th></tr><tr><td>Read</td><td>0x00</td></tr><tr><td>Vector Page Re-Map</td><td>0x2E</td></tr><tr><td>Program</td><td>0x21</td></tr><tr><td>Page Erase</td><td>0x22</td></tr></table> | ISP command | ISPCMD | Read | 0x00 | Vector Page Re-Map | 0x2E | Program | 0x21 | Page Erase | 0x22 |
|            |             | ISP command  | ISPCMD      |        |      |      |                    |      |         |      |            |      |
|            |             | Read   | 0x00        |        |      |      |                    |      |         |      |            |      |
|            |             | Vector Page Re-Map   | 0x2E        |        |      |      |                    |      |         |      |            |      |
|            |             | Program  | 0x21        |        |      |      |                    |      |         |      |            |      |
| Page Erase | 0x22        |  |             |        |      |      |                    |      |         |      |            |      |
|            |             |  |             |        |      |      |                    |      |         |      |            |      |
|            |             |  |             |        |      |      |                    |      |         |      |            |      |
|            |             |  |             |        |      |      |                    |      |         |      |            |      |

**ISP Trigger Control Register (ISPTRG)**

| Register | Offset      | R/W | Description                  | Reset Value |
|----------|-------------|-----|------------------------------|-------------|
| ISPTRG   | FMC_BA+0x10 | R/W | ISP Trigger Control Register | 0x0000_0000 |

|          |    |    |    |    |    |    |       |
|----------|----|----|----|----|----|----|-------|
| 31       | 30 | 29 | 28 | 27 | 26 | 25 | 24    |
| Reserved |    |    |    |    |    |    |       |
| 23       | 22 | 21 | 20 | 19 | 18 | 17 | 16    |
| Reserved |    |    |    |    |    |    |       |
| 15       | 14 | 13 | 12 | 11 | 10 | 9  | 8     |
| Reserved |    |    |    |    |    |    |       |
| 7        | 6  | 5  | 4  | 3  | 2  | 1  | 0     |
| Reserved |    |    |    |    |    |    | ISPGO |

| Bits   | Description  |
|--------|--|
| [31:1] | Reserved   |
| [0]    | <p><b>ISP Start Trigger (Write-protection Bit)</b></p> <p>Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished.</p> <p>1 = ISP progressed.</p> <p>0 = ISP operation finished.</p> <p>This bit is the protected bit, It means programming this bit needs to write "59h", "16h", "88h" to address 0x5000_0100 to disable register protection. Refer to the register REGWRPROT at address GCR_BA+0x100</p> |

### Data Flash Base Address Register (DFBADR)

| Register | Offset      | R/W | Description             | Reset Value |
|----------|-------------|-----|-------------------------|-------------|
| DFBADR   | FMC_BA+0x14 | R   | Data Flash Base Address | 0x000X_XXXX |

|               |    |    |    |    |    |    |    |
|---------------|----|----|----|----|----|----|----|
| 31            | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
| DFBADR[31:23] |    |    |    |    |    |    |    |
| 23            | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| DFBADR[23:16] |    |    |    |    |    |    |    |
| 15            | 14 | 13 | 12 | 11 | 10 | 9  | 8  |
| DFBADR[15:8]  |    |    |    |    |    |    |    |
| 7             | 6  | 5  | 4  | 3  | 2  | 1  | 0  |
| DFBADR[7:0]   |    |    |    |    |    |    |    |

| Bits   | Description  |
|--------|--|
| [31:0] | <p><b>DFBADR</b></p> <p><b>Data Flash Base Address</b></p> <p>This register indicates data flash start address. It is read only.</p> <p>For 128 KB flash memory device, the data flash size is defined by user configuration, register content is loaded from Config1 when chip is powered on but for 64/32 KB device, it is fixed at 0x0001_F000.</p> |



**Flash Access Time Control Register (FATCON)**

| Register | Offset      | R/W | Description                        | Reset Value |
|----------|-------------|-----|------------------------------------|-------------|
| FATCON   | FMC_BA+0x18 | R/W | Flash Access Time Control Register | 0x0000_0000 |

|          |            |          |            |    |    |    |    |
|----------|------------|----------|------------|----|----|----|----|
| 31       | 30         | 29       | 28         | 27 | 26 | 25 | 24 |
| Reserved |            |          |            |    |    |    |    |
| 23       | 22         | 21       | 20         | 19 | 18 | 17 | 16 |
| Reserved |            |          |            |    |    |    |    |
| 15       | 14         | 13       | 12         | 11 | 10 | 9  | 8  |
| Reserved |            |          |            |    |    |    |    |
| 7        | 6          | 5        | 4          | 3  | 2  | 1  | 0  |
| Reserved | FOM_SEL[1] | Reserved | FOM_SEL[0] |    |    |    |    |

| Bits   | Description |  |
|--------|-------------|--|
| [31:7] | Reserved    | Reserved   |
| [6]    | FOM_SEL[1]  | <b>Chip Frequency Optimization Mode Select (Write-protection Bit)</b><br>When chip operation frequency is lower than 25 MHz, chip can work more efficiently by setting FOM_SEL[1:0] = 01<br>When chip operation frequency is lower than 50 MHz, chip can work more efficiently by setting FOM_SEL[1:0] = 00<br>When chip operation frequency is over than 50 MHz, chip only can work by setting FOM_SEL[1:0] = 11<br>11 = High frequency optimization mode Enabled.<br>10 = Reserved.<br>01 = Low frequency optimization mode Enabled.<br>00 = Middle frequency optimization mode Enabled. |
| [5]    | Reserved    | Reserved   |
| [4]    | FOM_SEL[0]  | <b>Chip Frequency Optimization Mode Select (Write-protection Bit)</b><br>When chip operation frequency is lower than 25 MHz, chip can work more efficiently by setting FOM_SEL[1:0] = 01<br>When chip operation frequency is lower than 50 MHz, chip can work more efficiently by setting FOM_SEL[1:0] = 00<br>When chip operation frequency is over than 50 MHz, chip only can work by setting FOM_SEL[1:0] = 11<br>11 = High frequency optimization mode Enabled.<br>10 = Reserved.<br>01 = Low frequency optimization mode Enabled.<br>00 = Middle frequency optimization mode Enabled. |
| [3:0]  | Reserved    | Reserved   |

### ISP Status Register (ISPSTA)

| Register | Offset      | R/W | Description         | Reset Value |
|----------|-------------|-----|---------------------|-------------|
| ISPSTA   | FMC_BA+0x40 | R/W | ISP Status Register | 0x0000_0000 |

|             |       |          |    |              |     |    |          |
|-------------|-------|----------|----|--------------|-----|----|----------|
| 31          | 30    | 29       | 28 | 27           | 26  | 25 | 24       |
| Reserved    |       |          |    |              |     |    |          |
| 23          | 22    | 21       | 20 | 19           | 18  | 17 | 16       |
| Reserved    |       |          |    | VECMAP[11:7] |     |    |          |
| 15          | 14    | 13       | 12 | 11           | 10  | 9  | 8        |
| VECMAP[6:0] |       |          |    |              |     |    | Reserved |
| 7           | 6     | 5        | 4  | 3            | 2   | 1  | 0        |
| Reserved    | ISPFF | Reserved |    |              | CBS |    | ISPGO    |

| Bits    | Description |   |
|---------|-------------|---|
| [31:21] | Reserved    | Reserved  |
| [20:9]  | VECMAP      | <b>Vector Page Mapping Address (Read Only)</b><br>The current flash address space 0x0000_0000~0x0000_01FF is mapping to address {VECMAP[11:0], 9'h000} ~ {VECMAP[11:0], 9'h1FF}   |
| [8:7]   | Reserved    | Reserved  |
| [6]     | ISPFF       | <b>ISP Fail Flag (Write-protection Bit)</b><br>This bit is set by hardware when a triggered ISP meets any of the following conditions:<br>(1) APROM writes to itself<br>(2) LDROM writes to itself<br>(3) CONFIG is erased/programmed if CFGUEN is set to 0<br>(4) Destination address is illegal, such as over an available range<br>Write 1 to clear this bit.<br><b>Note:</b> This bit function is the same as ISPCON bit6 |
| [5:3]   | Reserved    | Reserved  |
| [2:1]   | CBS         | <b>Chip Boot Selection (Read Only)</b><br>This is a mirror of CBS in Config0.   |
| [0]     | ISPGO       | <b>ISP Start Trigger (Read Only)</b><br>Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished.<br>1 = ISP operation progressed.<br>0 = ISP operation finished.<br><b>Note:</b> This bit is the same with ISPTRG bit0  |

## 27 ELECTRICAL CHARACTERISTICS

### 27.1 Absolute Maximum Ratings

| SYMBOL                                    | PARAMETER           | MIN     | MAX     | UNIT |
|---|---------------------|---------|---------|------|
| DC Power Supply                           | VDD-VSS             | -0.3    | +6.3    | V    |
| Input Voltage                             | VIN                 | VSS-0.3 | VDD+0.3 | V    |
| Oscillator Frequency                      | 1/t <sub>CLCL</sub> | 4       | 24      | MHz  |
| Operating Temperature                     | TA                  | -40     | 105     | °C   |
| Storage Temperature                       | TST                 | -55     | +150    | °C   |
| Maximum Current into VDD                  |                     | -       | 120     | mA   |
| Maximum Current out of VSS                |                     |         | 120     | mA   |
| Maximum Current sunk by a I/O pin         |                     |         | 35      | mA   |
| Maximum Current sourced by a I/O pin      |                     |         | 35      | mA   |
| Maximum Current sunk by total I/O pins    |                     |         | 100     | mA   |
| Maximum Current sourced by total I/O pins |                     |         | 100     | mA   |

**Note:** Exposure to conditions beyond those listed under absolute maximum ratings may adversely affects the lift and reliability of the device.

### 27.2 DC Electrical Characteristics

| PARAMETER                            | SYM.                                  | SPECIFICATION |      |                  |      | TEST CONDITIONS  |
|--------------------------------------|---------------------------------------|---------------|------|------------------|------|--|
|                                      |                                       | MIN.          | TYP. | MAX.             | UNIT |  |
| Operation voltage                    | V <sub>DD</sub>                       | 2.5           | -    | 5.5              | V    | V <sub>DD</sub> = 2.5V ~ 5.5V  |
| Power Ground                         | V <sub>SS</sub> /<br>AV <sub>SS</sub> | -0.3          | -    | -                | V    |  |
| LDO Output Voltage                   | V <sub>LDO</sub>                      | 1.62          | 1.8  | 1.98             | V    | V <sub>DD</sub> ≥ 2.5V   |
| Analog Operating Voltage             | AV <sub>DD</sub>                      | 2.5           | -    | V <sub>DD</sub>  | V    |  |
| Analog Reference Voltage             | V <sub>ref</sub>                      | 1.2           | -    | AV <sub>DD</sub> | V    |  |
| Operating Current<br>Normal Run Mode | I <sub>DD1</sub>                      | -             | -    | 0.61F+8.1        | mA   | V <sub>DD</sub> = 5V,<br>enable all IP and PLL, external XTAL              |
|                                      | I <sub>DD2</sub>                      | -             | -    | 0.32F+7.7        | mA   | V <sub>DD</sub> = 5V,<br>disable all IP and enable PLL,<br>external XTAL   |
|                                      | I <sub>DD3</sub>                      | -             | -    | 0.58F+8.1        | mA   | V <sub>DD</sub> = 3.3V,<br>enable all IP and PLL, external XTAL            |
|                                      | I <sub>DD4</sub>                      | -             | -    | 0.29F+7.7        | mA   | V <sub>DD</sub> = 3.3V,<br>disable all IP and enable PLL,<br>external XTAL |
|                                      | I <sub>DD5</sub>                      | -             | -    | 0.66F+0.4        | mA   | V <sub>DD</sub> = 5V,<br>enable all IP and disable PLL,<br>external XTAL   |
|                                      | I <sub>DD6</sub>                      | -             | -    | 0.40F+0.2        | mA   | V <sub>DD</sub> = 5V,<br>disable all IP and disable PLL,<br>external XTAL  |
|                                      | I <sub>DD7</sub>                      | -             | -    | 0.60F+0.4        | mA   | V <sub>DD</sub> = 3.3V,<br>enable all IP and disable PLL,<br>external XTAL |
|                                      | I <sub>DD8</sub>                      | -             | -    | 0.35F+0.2        | mA   | V <sub>DD</sub> = 3.3V<br>disable all IP and disable PLL,<br>external XTAL |

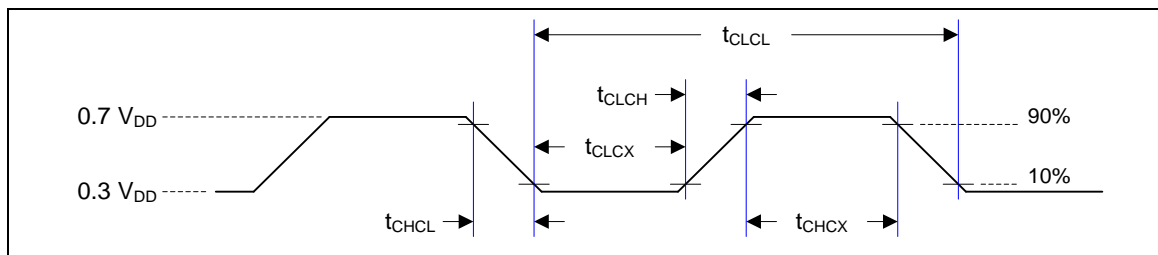
| PARAMETER   | SYM.                           | SPECIFICATION |      |           |      | TEST CONDITIONS  |
|---|--------------------------------|---------------|------|-----------|------|--|
|   |                                | MIN.          | TYP. | MAX.      | UNIT |  |
| Operating Current<br>Idle Mode                                | I <sub>IDLE1</sub>             | -             | -    | 0.39F+8.1 | mA   | V <sub>DD</sub> = 5V,<br>enable all IP and PLL, external XTAL              |
|   | I <sub>IDLE2</sub>             | -             | -    | 0.09F+7.7 | mA   | V <sub>DD</sub> = 5V,<br>disable all IP and enable PLL,<br>external XTAL   |
|   | I <sub>IDLE3</sub>             | -             | -    | 0.37F+8.1 | mA   | V <sub>DD</sub> = 3.3V,<br>enable all IP and PLL, external XTAL            |
|   | I <sub>IDLE4</sub>             | -             | -    | 0.08F+7.7 | mA   | V <sub>DD</sub> = 3.3V,<br>disable all IP and enable PLL,<br>external XTAL |
|   | I <sub>IDLE5</sub>             | -             | -    | 0.43F+0.5 | mA   | V <sub>DD</sub> = 5V,<br>enable all IP and disable PLL,<br>external XTAL   |
|   | I <sub>IDLE6</sub>             | -             | -    | 0.18F+0.4 | mA   | V <sub>DD</sub> = 5V,<br>disable all IP and disable PLL,<br>external XTAL  |
|   | I <sub>IDLE7</sub>             | -             | -    | 0.38F+0.5 | mA   | V <sub>DD</sub> = 3.3V,<br>enable all IP and disable PLL,<br>external XTAL |
|   | I <sub>IDLE8</sub>             | -             | -    | 0.13F+0.4 | mA   | V <sub>DD</sub> = 3.3V<br>disable all IP and disable PLL,<br>external XTAL |
| Standby Current<br>Power-down Mode                            | I <sub>PWD</sub>               | -             | -    | 25        | μA   | V <sub>DD</sub> = 5.5V, No load<br>@ Disable BOV function, 25°C            |
| Logic 0 Input Current<br>(Quasi-bidirectional mode)           | I <sub>IL</sub>                | -             | -    | -75       | μA   |  |
| Input Leakage Current<br>(input only)                         | I <sub>LK</sub>                | -             | -    | 2         | μA   |  |
| Logic 1 to 0 Transition<br>Current (Quasi-bidirectional mode) | I <sub>TL</sub> <sup>[3]</sup> | -             | -    | -660      | μA   | V <sub>DD</sub> = 5.5V, V <sub>IN</sub> <2.0V                              |
| Internal Pull-High Resistor<br>of /RESET <sup>[1]</sup>       | R <sub>RST</sub>               | 15            | -    | -         | kΩ   |  |

| PARAMETER   | SYM.      | SPECIFICATION   |             |                 |               | TEST CONDITIONS  |
|---|-----------|-----------------|-------------|-----------------|---------------|--|
|   |           | MIN.            | TYP.        | MAX.            | UNIT          |  |
| Input Low Voltage (TTL input)                         | $V_{IL}$  | -0.3            | -           | $0.2V_{DD}-0.1$ | V             |  |
| Input Low Voltage (Schmitt input)                     | $V_{IL1}$ | -0.3            |             | $0.3V_{DD}$     | V             |  |
| Input Low Voltage (/RESET, XTAL in)                   | $V_{IL2}$ | -0.3            |             | $0.15V_{DD}$    | V             |  |
| Input High Voltage (TTL input)                        | $V_{IH}$  | $0.2V_{DD}+0.9$ | -           | $V_{DD}+0.3$    | V             |  |
| Input High Voltage (Schmitt input, /RESET, XTAL in)   | $V_{IH1}$ | $0.7V_{DD}$     | -           | $V_{DD}+0.3$    | V             |  |
| Hysteresis voltage of (Schmitt input)                 | $V_{HY}$  | -               | $0.2V_{DD}$ | -               | V             |  |
| Band-gap Voltage                                      | $V_{BG}$  | 1.22            | 1.25        | 1.28            | V             | $V_{DD} = 2.5\text{ V} \sim 5.5\text{ V}$ , $T_A = 25^\circ\text{C}$ |
| Source Current (Quasi-bidirectional Mode)             | $I_{OH}$  | -360            | -           | -               | $\mu\text{A}$ | $V_{DD} = 4.5\text{V}$ , $V_S = 2.4\text{V}$                         |
|   |           | -60             | -           | -               | $\mu\text{A}$ | $V_{DD} = 2.7\text{V}$ , $V_S = 2.2\text{V}$                         |
|   |           | -50             | -           | -               | $\mu\text{A}$ | $V_{DD} = 2.5\text{V}$ , $V_S = 2.0\text{V}$                         |
| Source Current (Push-pull Mode)                       | $I_{OH1}$ | -25             | -           | -               | mA            | $V_{DD} = 4.5\text{V}$ , $V_S = 2.4\text{V}$                         |
|   |           | -4              | -           | -               | mA            | $V_{DD} = 2.7\text{V}$ , $V_S = 2.2\text{V}$                         |
|   |           | -3              | -           | -               | mA            | $V_{DD} = 2.5\text{V}$ , $V_S = 2.0\text{V}$                         |
| Sink Current (Quasi-bidirectional and Push-pull Mode) | $I_{OL}$  | 16              | -           | -               | mA            | $V_{DD} = 4.5\text{V}$ , $V_S = 0.45\text{V}$                        |
|   |           | 10              | -           | -               | mA            | $V_{DD} = 2.7\text{V}$ , $V_S = 0.45\text{V}$                        |
|   |           | 9               | -           | -               | mA            | $V_{DD} = 2.5\text{V}$ , $V_S = 0.45\text{V}$                        |

**Note:**

1. /RESET pin is a Schmitt trigger input.
2. Crystal Input is a CMOS input.
3. I/O pin can source a transition current when they are being externally driven from 1 to 0. In the condition of  $V_{DD}=5.5\text{V}$ , the transition current reaches its maximum value when  $V_{IN}$  approximates to 2V.

## 27.3 AC Electrical Characteristics



**Note:** Duty cycle is 50%.

| SYMBOL            | PARAMETER       | CONDITION | MIN. | TYP. | MAX. | UNIT |
|-------------------|-----------------|-----------|------|------|------|------|
| t <sub>CHCX</sub> | Clock High Time |           | 10   | -    | -    | nS   |
| t <sub>CLCX</sub> | Clock Low Time  |           | 10   | -    | -    | nS   |
| t <sub>CLCH</sub> | Clock Rise Time |           | 2    | -    | 15   | nS   |
| t <sub>CHCL</sub> | Clock Fall Time |           | 2    | -    | 15   | nS   |

### 27.3.1 External 4~24MHz Crystal

| PARAMETER                         | CONDITION                      | MIN. | TYP.. | MAX. | UNIT |
|-----------------------------------|--------------------------------|------|-------|------|------|
| Operation Voltage V <sub>DD</sub> | -                              | 2.5  | -     | 5.5  | V    |
| Temperature                       | -                              | -40  | -     | 85   | °C   |
| Operating Current                 | 12 MHz at V <sub>DD</sub> = 5V | -    | 1     | -    | mA   |
| Clock Frequency                   | External crystal               | 4    |       | 24   | MHz  |

#### 27.3.1.1 Typical Crystal Application Circuits

| CRYSTAL        | C1       | C2       | R       |
|----------------|----------|----------|---------|
| 4 MHz ~ 24 MHz | 10~20 pF | 10~20 pF | without |

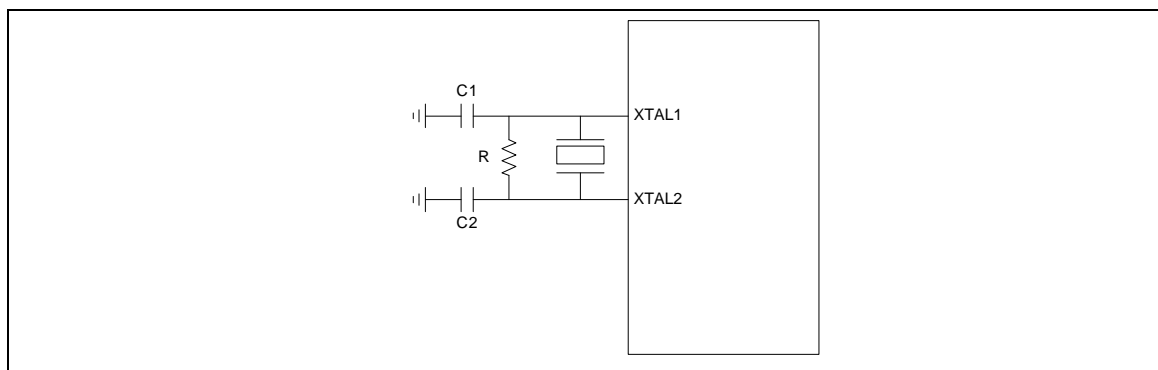


Figure 27–1 Typical Crystal Application Circuit

### 27.3.2 Internal 22.1184 MHz Oscillator

| PARAMETER                        | CONDITION                                     | MIN. | TYP.    | MAX. | UNIT |
|----------------------------------|---|------|---------|------|------|
| Supply voltage                   | -   | 2.5  | -       | 5.5  | V    |
| Frequency<br>(After calibration) | -   | -    | 22.1184 | -    | MHz  |
|                                  | +25°C; V <sub>DD</sub> = 5V                   | -1   | -       | +1   | %    |
|                                  | -40 to +105°C;<br>V <sub>DD</sub> = 2.5V~5.5V | -2   | -       | +2   | %    |
| Operation Current                | V <sub>DD</sub> =5V                           | -    | 500     | -    | uA   |

### 27.3.3 Internal 10 kHz Oscillator

| PARAMETER                                | CONDITION                                     | MIN. | TYP. | MAX. | UNIT |
|--|---|------|------|------|------|
| Supply voltage                           | -   | 2.5  | -    | 5.5  | V    |
| Center Frequency                         | -   | -    | 10   | -    | kHz  |
| Calibrated Internal Oscillator Frequency | +25°C; V <sub>DD</sub> =5 V                   | -30  | -    | +30  | %    |
|  | -40°C ~+85°C;<br>V <sub>DD</sub> =2.5 V~5.5 V | -50  | -    | +50  | %    |



## 27.4 Analog Characteristics

## 27.4.1 Specification of 12-bit SARADC

| PARAMETER                       | SYMBOL    | CONDITON    | MIN.       | TYP.  | MAX.  | UNIT  |
|---------------------------------|-----------|-------------|------------|-------|-------|-------|
| Resolution                      | -         |             | 12         |       |       | Bit   |
| Differential nonlinearity error | DNL       |             | -          | -1~+2 | -1~+4 | LSB   |
| Integral nonlinearity error     | INL       |             | -          | ±1.5  | ±4    | LSB   |
| Offset error                    | EO        |             | -          | +3    | +5    | LSB   |
| Full scale error                | EG        |             | -          | -3    | -6    | LSB   |
| Absolute error                  | EA        |             |            | -     | ±4    |       |
| Monotonic                       | -         |             | Guaranteed |       |       |       |
| ADC clock frequency             | $F_{ADC}$ | AVDD = 4.5V | -          | -     | 16    | MHz   |
|                                 |           | AVDD = 2.5V | -          | -     | 8     |       |
| Sample rate                     | $F_S$     | AVDD = 4.5V | -          | -     | 800   | kps   |
|                                 |           | AVDD = 2.5V | -          | -     | 400   |       |
| Sample time                     | $T_S$     |             | -          | 8     | -     | Clock |
| Conversion time                 | $T_{ADC}$ |             | -          | 12    | -     | Clock |
| Supply voltage                  | AVDD      |             | 2.5        | -     | 5.5   | V     |
| VRFE voltage                    | VREF      |             | 2.0        |       | AVDD  |       |
| Supply current                  | $I_{DDA}$ |             | -          | 1.5   | -     | mA    |
| Reference current               | $I_{RFE}$ |             | -          | 1     | -     | mA    |
| Input voltage                   | $V_{IN}$  |             | 0          | -     | VREF  | V     |
| Resistance                      | $R_{IN}$  |             |            | 6     |       | kΩ    |
| Capacitance                     | $C_{IN}$  |             | -          | 5     | -     | pF    |

## 27.4.2 Specification of LDO

| PARAMETER              | MIN. | TYP. | MAX. | UNIT | NOTE                   |
|------------------------|------|------|------|------|------------------------|
| Input Voltage $V_{DD}$ | 2.5  |      | 5.5  | V    | $V_{DD}$ input voltage |
| Output Voltage         | 1.62 | 1.8  | 1.98 | V    | $V_{DD} > 2.5$ V       |
| Operating Temperature  | -40  | 25   | 105  | °C   |                        |
| Cbp                    | -    | 1    | -    | μF   | $R_{ESR} = 1 \Omega$   |

**Note:**

1. It is recommended that a 10 μF or higher capacitor and a 100 nF bypass capacitor are connected between  $V_{DD}$  and the closest  $V_{SS}$  pin of the device.
2. To ensure power stability, a 1 μF or higher capacitor must be connected between LDO\_CAP pin and the closest  $V_{SS}$  pin of the device.

### 27.4.3 Specification of Low Voltage Reset

| PARAMETER             | CONDITION               | MIN. | TYP. | MAX. | UNIT |
|-----------------------|-------------------------|------|------|------|------|
| Operation Voltage     | -                       | 0    | -    | 5.5  | V    |
| Quiescent Current     | AV <sub>DD</sub> =5.5 V | -    | 1    | 5    | μA   |
| Operation Temperature | -                       | -40  | 25   | 105  | °C   |
| Threshold Voltage     | -                       | 1.6  | 2.0  | 2.4  | V    |
| Hysteresis            | -                       | 0    | 0    | 0    | V    |

### 27.4.4 Specification of Brown-out Detector

| PARAMETER         | CONDITION               | MIN. | TYP. | MAX. | UNIT |
|-------------------|-------------------------|------|------|------|------|
| Operation Voltage | -                       | 0    | -    | 5.5  | V    |
| Temperature       | -                       | -40  | 25   | 105  | °C   |
| Quiescent Current | AV <sub>DD</sub> =5.5 V | -    | -    | 125  | μA   |
| Brown-out Voltage | BOD_VL[1:0]=11          | 4.2  | 4.4  | 4.6  | V    |
|                   | BOD_VL [1:0]=10         | 3.5  | 3.7  | 3.9  | V    |
|                   | BOD_VL [1:0]=01         | 2.6  | 2.7  | 2.8  | V    |
|                   | BOD_VL [1:0]=00         | 2.1  | 2.2  | 2.3  | V    |
| Hysteresis        | -                       | 30   | -    | 150  | mV   |

### 27.4.5 Specification of Power-On Reset (5V)

| PARAMETER             | CONDITION           | MIN. | TYP. | MAX. | UNIT |
|-----------------------|---------------------|------|------|------|------|
| Operation Temperature | -                   | -40  | 25   | 105  | °C   |
| Reset Voltage         | V+                  | -    | 2    | -    | V    |
| Quiescent Current     | Vin > reset voltage | -    | 1    | -    | nA   |

### 27.4.6 Specification of Temperature Sensor

| PARAMETER                        | CONDITIONS | MIN. | TYP.  | MAX. | UNIT  |
|----------------------------------|------------|------|-------|------|-------|
| Operation Voltage <sup>[1]</sup> |            | 2.5  | -     | 5.5  | V     |
| Operation Temperature            |            | -40  | -     | 105  | °C    |
| Current Consumption              |            | 6.4  | -     | 10.5 | μA    |
| Gain                             |            |      | -1.76 |      | mV/°C |
| Offset Voltage                   | Temp=0 °C  |      | 720   |      | mV    |

### 27.4.7 Specification of Comparator

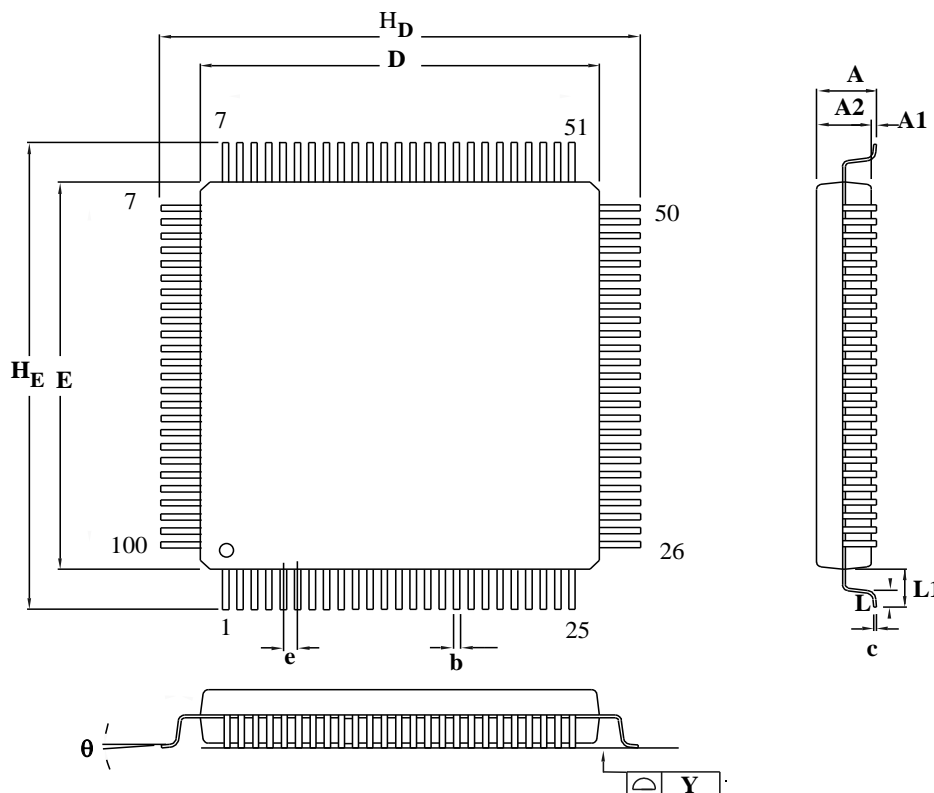
| PARAMETER                          | CONDITION  | MIN. | TYP. | MAX.                 | UNIT |
|------------------------------------|--|------|------|----------------------|------|
| Operation Voltage AV <sub>DD</sub> | -  | 2.5  |      | 5.5                  | V    |
| Operation Temperature              | -  | -40  | 25   | 85                   | °C   |
| Operation Current                  | V <sub>DD</sub> =3.0 V   | -    | 20   | 40                   | μA   |
| Input Offset Voltage               | -  | -    | 5    | 15                   | mV   |
| Output Swing                       | -  | 0.1  | -    | V <sub>DD</sub> -0.1 | V    |
| Input Common Mode Range            | -  | 0.1  | -    | V <sub>DD</sub> -1.2 | V    |
| DC Gain                            | -  | -    | 70   | -                    | dB   |
| Propagation Delay                  | V <sub>CM</sub> =1.2 V and<br>V <sub>DIFF</sub> =0.1 V   | -    | 200  | -                    | ns   |
| Comparison Voltage                 | 20 mV at V <sub>CM</sub> =1 V<br>50 mV at V <sub>CM</sub> =0.1 V<br>50 mV at V <sub>CM</sub> =V <sub>DD</sub> -1.2<br>10 mV for non-hysteresis | 10   | 20   | -                    | mV   |
| Hysteresis                         | V <sub>CM</sub> =0.4 V ~ V <sub>DD</sub> -1.2 V  | -    | ±10  | -                    | mV   |
| Wake-up Time                       | C <sub>INP</sub> =1.3 V<br>C <sub>INN</sub> =1.2 V   | -    | -    | 2                    | μs   |

## 27.4.8 Specification of OP Amplifier

| PARAMETER                  | CONDITION                        | MIN. | TYP. | MAX.    | UNIT  |
|----------------------------|----------------------------------|------|------|---------|-------|
| AVDD                       | -                                | 3.0  | 3.3  | 5.5     | V     |
| Input offset voltage       | -                                | -    | 2    | 5       | mV    |
| Input offset average drift | -                                | -    | -    | 1       | uV/°C |
| Output swing               | -                                | 0.1  | -    | VDD-0.1 | V     |
| Input common mode range    | -                                | 0.1  | -    | VDD-1.2 | V     |
| DC gain                    | -                                | -    | 80   | -       | dB    |
| Unity gain freq.           | AVDD=5V                          | -    | -    | 5       | MHz   |
| Phase margin               | -                                | -    | 50°  | -       | °     |
| PSRR+                      | AVDD=5V                          | -    | 90   | -       | dB    |
| CMRR                       | AVDD=5V                          | -    | 90   | -       | dB    |
| Slew rate                  | AVDD=5V, RLOAD=33K,<br>CLOAD=50p | 6.0  | -    | -       | V/us  |
| Wake up time               | -                                | -    | -    | 1       | us    |
| Quiescent current          | -                                | -    | -    | 2       | mA    |

## 28 PACKAGE DIMENSIONS

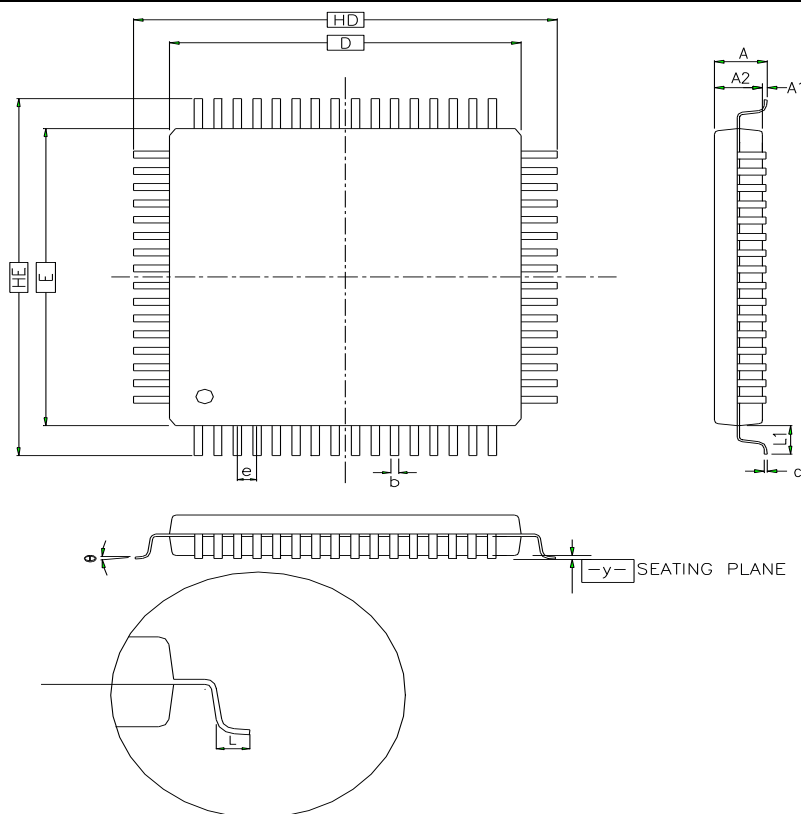
### 28.1 100L LQFP (14x14x1.4 mm footprint 2.0mm)



Controlling Dimension : Millimeters

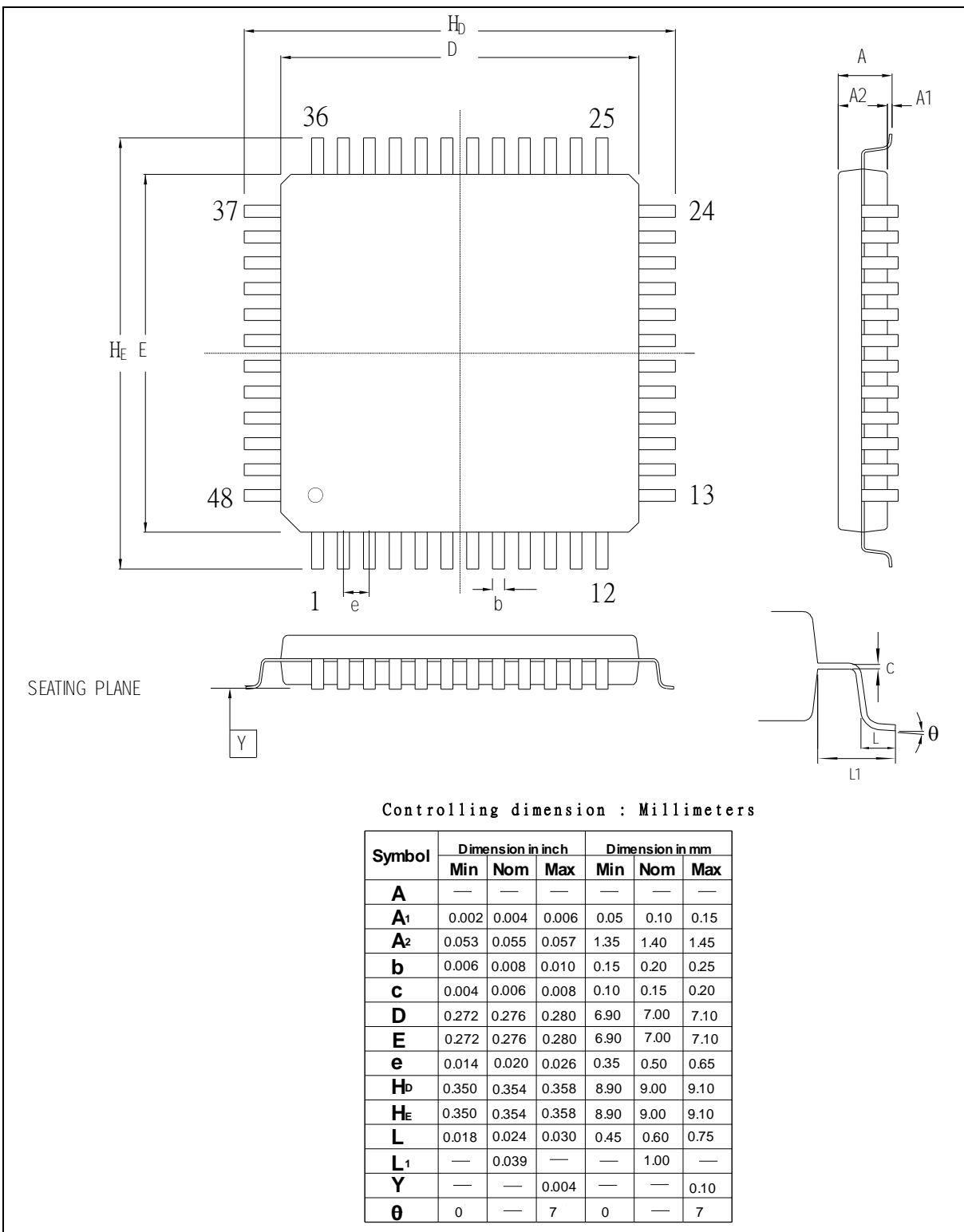
| Symbol   | Dimension in inch |       |       | Dimension in mm |       |       |
|----------|-------------------|-------|-------|-----------------|-------|-------|
|          | Min               | Nom   | Max   | Min             | Nom   | Max   |
| A        | —                 | —     | 0.063 | —               | —     | 1.60  |
| A1       | 0.002             | —     | —     | 0.05            | —     | —     |
| A2       | 0.053             | 0.055 | 0.057 | 1.35            | 1.40  | 1.45  |
| b        | 0.007             | 0.009 | 0.011 | 0.17            | 0.22  | 0.27  |
| c        | 0.004             | 0.006 | 0.008 | 0.10            | 0.15  | 0.20  |
| D        | 0.547             | 0.551 | 0.556 | 13.90           | 14.00 | 14.10 |
| E        | 0.547             | 0.551 | 0.556 | 13.90           | 14.00 | 14.10 |
| e        | —                 | 0.020 | —     | —               | 0.50  | —     |
| $H_D$    | 0.622             | 0.630 | 0.638 | 15.80           | 16.00 | 16.20 |
| $H_E$    | 0.622             | 0.630 | 0.638 | 15.80           | 16.00 | 16.20 |
| L        | 0.018             | 0.024 | 0.030 | 0.45            | 0.60  | 0.75  |
| L1       | —                 | 0.039 | —     | —               | 1.00  | —     |
| y        | —                 | —     | 0.004 | —               | —     | 0.10  |
| $\theta$ | 0°                | —     | 7°    | 0°              | —     | 7°    |

## 28.2 64L LQFP (10x10x1.4mm footprint 2.0 mm)



| Symbol               | Dimension in inch |       |       | Dimension in mm |       |      |
|----------------------|-------------------|-------|-------|-----------------|-------|------|
|                      | Min               | Nom   | Max   | Min             | Nom   | Max  |
| <b>A</b>             | —                 | —     | 0.063 | —               | —     | 1.60 |
| <b>A<sub>1</sub></b> | 0.002             | —     | 0.006 | 0.05            | —     | 0.15 |
| <b>A<sub>2</sub></b> | 0.053             | 0.055 | 0.057 | 1.35            | 1.40  | 1.45 |
| <b>b</b>             | 0.007             | 0.008 | 0.011 | 0.17            | 0.20  | 0.27 |
| <b>c</b>             | 0.004             | —     | 0.008 | 0.09            | —     | 0.20 |
| <b>D</b>             | —                 | 0.393 | —     | —               | 10.00 | —    |
| <b>E</b>             | —                 | 0.393 | —     | —               | 10.00 | —    |
| <b>e</b>             | —                 | 0.020 | —     | —               | 0.50  | —    |
| <b>H<sub>D</sub></b> | —                 | 0.472 | —     | —               | 12.00 | —    |
| <b>H<sub>E</sub></b> | —                 | 0.472 | —     | —               | 12.00 | —    |
| <b>L</b>             | 0.018             | 0.024 | 0.030 | 0.45            | 0.60  | 0.75 |
| <b>L<sub>1</sub></b> | —                 | 0.039 | —     | —               | 1.00  | —    |
| <b>y</b>             | —                 | 0.004 | —     | —               | 0.10  | —    |
| <b>θ</b>             | 0                 | 3.5   | 7     | 0               | 3.5   | 7    |

28.3 48L LQFP (7x7x1.4mm footprint 2.0mm)



### 29 REVISION HISTORY

| REVISION | DATE       | PAGE | DESCRIPTION   |
|----------|------------|------|---|
| V0.1     | 2011/8/30  |      | Preliminary release.  |
| V0.2     | 2012/03/20 |      | 1. Remove MT510 part<br>2. Update the package and pin assignment.   |
| V0.3     | 2012/10/08 |      | 1. Update the part numbers with MT510, MT520 and MT530.<br>2. Update the pin assignment and description<br>3. Update the general purpose I/O<br>4. Update the timer/counter<br>5. Update the basic PWM function<br>6. Update the enhance PWM function<br>7. Update the analog comparator function |
| V0.3.1   | 2013/08/07 |      | 1. Modify the content of register AHBCLK  |
| V0.4.    | 2014/04/09 |      | 1. Change part name from MT5xx to NM15xx.<br>2. Modified CAP_CTR1.CAPDIV<br>3. Modified some ADC registers:<br>a. Remove register ADITSSELR<br>b. Modified registers: ADSPCRA0~7/ADSPCRB0~7<br>c. Added registers: SMPAnTRGEN and SMPBnTRGEN, n=0~3.<br>4. Modified register ACMPSR               |
| V0.5     | 2014/06/19 |      | 1. Update the Parts List in chapter 4.<br>2. Update the diagrams of pin configuration in chapter 5.   |
| V0.6     | 2014/7/18  |      | 1. Update the Specification of 12-bit SARADC  |
| V0.7     | 2014/7/25  |      | 1. Modified the MDU block diagram in Figure 15–2.<br>2. Correct P7.4 and P7.5 about comparator input in Pin Description.  |
| V0.8     | 2015/2/25  |      | 1. Correct the content of LDROM size in chapter 2.<br>2. Correct the content of Table 26-1.   |
| V0.9     | 2016/03/01 |      | 1. Correct pin assignment of IC10~IC12. Modify chapter4 of <u>part list</u> , chapter5 of <u>pin configuration</u> and the description of <u>P0_MFP</u>   |
| V0.9.1   | 2016/03/22 |      | 1. Modify the diagram in section 23.3 and the content of section 23.4.6   |
| V0.9.2   | 2016/07/13 |      | 1. Add band-gap voltage spec  |
| V0.9.3   | 2017/05/22 |      | 1. Add new part of NM1521 LQFP64  |



### Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

---

Please note that all data and specifications are subject to change without notice.  
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.