ARM Cortex® -M0

32-bit Microcontroller

NuMicro® Family
NDA102 Series
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

TABLE OF CONTENTS

1 GENERAL DESCRIPTION 11
2 FEATURES 13
3 ABBREVIATIONS 17
4 PARTS INFORMATION LIST AND PIN CONFIGURATION 18
4.1 NuMicro® NDA102 Naming Rule 18
4.2 NuMicro® NDA102 Series Selection Guide 19
4.3 Pin Configuration 20
4.3.1 TSSOP 28-Pin 20
4.4 Pin Description 21
4.4.1 NDA102 Series Pin Description 21
4.4.2 GPIO Multi-function Pin Summary 25
5 BLOCK DIAGRAM 28
5.1 NuMicro® NDA102 Block Diagram 28
6 FUNCTIONAL DESCRIPTION 29
6.1 ARM® Cortex®-M0 Core 29
6.1.1 Overview 29
6.1.2 Features 29
6.2 System Manager 31
6.2.1 Overview 31
6.2.2 System Reset 31
6.2.3 Power Modes and Wake-up Sources 37
6.2.4 System Power Architecture 40
6.2.5 System Memory Mapping 41
6.2.6 Register Protection 42
6.2.7 Memory Organization 44
6.2.8 Register Map 47
6.2.9 Register Description 48
6.2.10 System Timer (SysTick) 68
6.2.11 Nested Vectored Interrupt Control (NVIC) 73
6.2.12 System Control Registers 93
6.3 Clock Controller 102
6.3.1 Overview 102
6.3.2 Auto Trim 104
6.3.3 System Clock and SysTick Clock 104
6.3.4 Peripherals Clock Source Selection 105
6.3.5 Power-down Mode Clock 107
6.3.6 Frequency Divider Output 107
6.3.7 Register Map 109
6.3.8 Register Description 110
6.4 Flash Memory Controller (FMC) 122
6.4.1 Overview 122

6.4.2 Features 122
6.4.3 Block Diagram 122
6.4.4 Functional Description 124
6.4.5 Register Map 139
6.4.6 Register Description 140
6.5 General Purpose I/O (GPIO) 151
6.5.1 Overview 151
6.5.2 Features 151
6.5.3 Block Diagram 152
6.5.4 Basic Configuration 152
6.5.5 Functional Description 152
6.5.6 GPIO Interrupt and Wake-up Function 155
6.5.7 Register Map 156
6.5.8 Register Description 159
6.6 Timer Controller (TIMER) 176
6.6.1 Overview 176
6.6.2 Features 176
6.6.3 Block Diagram 177
6.6.4 Basic Configuration 178
6.6.5 Functional Description 178
6.6.6 Register Map 185
6.6.7 Register Description 186
6.7 Enhanced Input Capture Timer (ECAP) 198
6.7.1 Overview 198
6.7.2 Features 198
6.7.3 Block Diagram 198
6.7.4 Input Noise Filter 199
6.7.5 Operation of Input Capture Timer/Counter 200
6.7.6 Input Capture Timer/Counter Interrupt Architecture....202
6.7.7 Register Map 203
6.7.8 Register Description 204
6.8 Enhanced PWM Generator (EPWM) 214
6.8.1 Overview 214
6.8.2 Features 214
6.8.3 Block Diagram 215
6.8.4 Basic Configuration 217
6.8.5 Functional Description 217
6.8.6 Register Map 236
6.8.7 Register Description 237
6.9 Basic PWM Generator (BPWM) 264
6.9.1 Overview 264
6.9.2 Features 264
6.9.3 Block Diagram 264
6.9.4 PWM-Timer Operation....265
6.9.5 Register Map 273
6.9.6 Register Description 274

6.10 Watchdog Timer (WDT) 284
6.10.1 Overview 284
6.10.2 Features 284
6.10.3 Block Diagram 284
6.10.4 Clock Control 285
6.10.5 Basic Configuration 285
6.10.6 Functional Description 285
6.10.7 Registers Map 287
6.10.8 Register Description 288
6.11 USCI – Universal Serial Control Interface Controller 290
6.11.1 Overview 290
6.11.2 Features 290
6.11.3 Block Diagram 290
6.11.4 Functional Description 291
6.12 USCI – UART Mode 302
6.12.1 Overview 302
6.12.2 Features 302
6.12.3 Block Diagram 302
6.12.4 Basic Configuration 303
6.12.5 Functional Description 303
6.12.6 Register Map 313
6.12.7 Register Description 314
6.13 USCI – SPI Mode 335
6.13.1 Overview 335
6.13.2 Features 335
6.13.3 Block Diagram 336
6.13.4 Basic Configuration 336
6.13.5 Functional Description 337
6.13.6 Register Map 350
6.13.7 Register Description 351
6.14 USCI – I^2C Mode 373
6.14.1 Overview 373
6.14.2 Features 373
6.14.3 Block Diagram 374
6.14.4 Basic Configuration 374
6.14.5 Functional Description 374
6.14.6 Register Map 392
6.14.7 Register Description 393
6.15 Hardware Divider (HDIV) 411
6.15.1 Overview 411
6.15.2 Features 411
6.15.3 Basic Configuration 411
6.15.4 Functional Description 412
6.15.5 Register Map 413
6.15.6 Register Description 414

6.16 Analog to Digital Converter (ADC) 419
1.1.1 Overview 419
1.1.2 Features 419
1.1.3 Block Diagram 419
6.16.1 Basic Configuration 420
6.16.2 Functional Description 420
6.16.3 Register Map 424
6.16.4 Register Description 425
6.17 Analog Comparator (ACMP) 441
6.17.1 Overview 441
6.17.2 Features 441
6.17.3 Block Diagram 442
6.17.4 Basic Configuration 442
6.17.5 Functional Description 443
6.17.6 Comparator Reference Voltage (CRV) 444
6.17.7 Register Map 445
6.17.8 Register Description 446
6.18 Programmable Gain Amplifier (PGA) 456
6.18.1 Overview 456
1.1.4 Features 456
1.1.5 Block Diagram 456
1.1.6 Register Map 456
6.18.2 Register Description 457
7 APPLICATION CIRCUIT 458
8 ELECTRICAL CHARACTERISTICS 459
9 PACKAGE DIMENSIONS 460
1.2 28-Pin TSSOP (4.4x9.7x1.0 mm) 460
10 REVISION HISTORY 461

List of Figures

Figure 4.1-1 NuMicro® NDA102 Series Selection Code 18
Figure 4.3-1 NuMicro® NDA102 Series TSSOP 28-pin Diagram 20
Figure 4.3-2 NuMicro® NDA102 Series TSSOP 28-pin Multi-function Diagram 20
Figure 5.1-1 NuMicro® NDA102 Block Diagram 28
Figure 6.1-1 Functional Block Diagram 29
Figure 6.2-1 System Reset Resources 32
Figure 6.2-2 nRESET Reset Waveform 34
Figure 6.2-3 Power-on Reset (POR) Waveform 34
Figure 6.2-4 Low Voltage Reset (LVR) Waveform 35
Figure 6.2-5 Brown-out Detector (BOD) Waveform 36
Figure 6.2-6 Power Mode State Machine 37
Figure 6.2-7 NuMicro® NDA102 Series Power Architecture Diagram 40
Figure 6.2-8 NuMicro® NDA102 Flash, Security and Configuration Map 44
Figure 6.2-9 SRAM Block Diagram 46
Figure 6.3-1 Clock Generator Block Diagram 102
Figure 6.3-2 Clock Generator Global View Diagram 103
Figure 6.3-3 System Clock Block Diagram 104
Figure 6.3-4 SysTick Clock Control Block Diagram 105
Figure 6.3-5 Peripherals Bus Clock Source Selection for PCLK 106
Figure 6.3-6 Clock Source of Frequency Divider 108
Figure 6.3-7 Block Diagram of Frequency Divider 108
Figure 6.4-1 Flash Memory Control Block Diagram 123
Figure 6.4-2 Flash Memory Organization 125
Figure 6.4-3 Data Flash Shared with APROM 126
Figure 6.4-4 SPROM Security Mode 128
Figure 6.4-5 Boot Select (BS) for Power-on Action 133
Figure 6.4-6 Flash Memory Mapping of CBS in CONFIG0 134
Figure 6.4-7 Executable Range of Code with IAP Function Enabled 135
Figure 6.4-8 Example Flow of Boot Selection by BS Bit when CBS[0] = 1 136
Figure 6.4-9 ISP Flow Example 138
Figure 6.5-1 I/O Pin Block Diagram 151
Figure 6.5-2 GPIO Controller Block Diagram 152
Figure 6.5-3 Push-Pull Output 153
Figure 6.5-4 Open-Drain Output 154
Figure 6.5-5 Quasi-Bidirectional I/O Mode 154

Figure 6.6-1 Timer Controller Block Diagram 177
Figure 6.6-2 Clock Source of Timer Controller 177
Figure 6.6-3 Continuous Counting Mode 179
Figure 6.6-4 Free-Counting Capture Mode 180
Figure 6.6-5 External Reset Counter Mode 181
Figure 6.6-6 Continuous Capture Mode Block 183
Figure 6.6-7 Continuous Capture Mode Behavior 184
Figure 6.7-1 Enhanced Input Capture Timer/Counter Architecture 198
Figure 6.7-2 Enhanced Input Capture Timer/Counter Clock Source Control 199
Figure 6.7-3 Noise Filter Sampling Clock Selection 199
Figure 6.7-4 Noise Filter 199
Figure 6.7-5 Enhanced Input Capture Timer/Counter Functions Block 201
Figure 6.7-6 Enhanced Input Capture Timer/Counter Interrupt Architecture Diagram 202
Figure 6.8-1 EPWM Clock Source 215
Figure 6.8-2 EPWM Block Diagram 216
Figure 6.8-3 EPWM Generator 0/1 Architecture Diagram 216
Figure 6.8-4 EPWM Generator 2/3 Architecture Diagram 216
Figure 6.8-5 EPWM Generator 4/5 Architecture Diagram 217
Figure 6.8-6 EPWM Edge-aligned Waveform Output 218
Figure 6.8-7 EPWM Edge-aligned Mode Operation Timing 219
Figure 6.8-8 EPWM Edge-aligned Interrupt Diagram 220
Figure 6.8-9 EPWM Edge-aligned Flow Diagram 221
Figure 6.8-10 EPWM Legend of Internal Comparator Output of PWM-Timer 222
Figure 6.8-11 EPWM-Timer Operation Timing 222
Figure 6.8-12 EPWM Center-aligned Type 223
Figure 6.8-13 EPWM Center-aligned Mode Operation Timing 224
Figure 6.8-14 EPWM Center-aligned Waveform Output 225
Figure 6.8-15 EPWM Center-aligned Interrupt Diagram 225
Figure 6.8-16 EPWM Center-aligned Flow Diagram 226
Figure 6.8-17 EPWM Dead-time Insertion 228
Figure 6.8-18 EPWM Asymmetric Mode Timing Diagram 229
Figure 6.8-19 EPWM One-Shot Mode Architecture 229
Figure 6.8-20 EPWM Initial State and Polarity Control with Rising Edge Dead-time Insertion ... 230
Figure 6.8-21 EPWM Interrupt Architecture 231
Figure 6.8-22 EPWM Brake Architecture 232
Figure 6.8-23 EPWM 3-phase Motor Mask Diagram 233

Figure 6.8-24 EPWM 3-phase Motor Mask Example 1 234
Figure 6.8-25 EPWM 3-phase Motor Mask Example 2 234
Figure 6.9-1 PWM Clock Source Control 265
Figure 6.9-2 PWM Architecture Diagram 265
Figure 6.9-3 Legend of Internal Comparator Output of PWM-Timer 266
Figure 6.9-4 PWM-Timer Operation Timing 267
Figure 6.9-5 PWM Edge-aligned Interrupt Generate Timing Waveform 267
Figure 6.9-6 Center-aligned Type Output Waveform 268
Figure 6.9-7 PWM Center-aligned Interrupt Generate Timing Waveform 269
Figure 6.9-8 PWM Double Buffering Illustration 270
Figure 6.9-9 PWM Controller Output Duty Ratio 270
Figure 6.9-10 Paired-PWM Output with Dead-zone Generation Operation 271
Figure 6.9-11 PWM Interrupt Architecture Diagram 271
Figure 6.10-1 Watchdog Timer Block Diagram 284
Figure 6.10-2 Watchdog Timer Clock Control Diagram 285
Figure 6.10-3 Watchdog Timer Time-out Interval and Reset Period Timing 286
Figure 6.11-1 USCI Block Diagram 290
Figure 6.11-2 Input Conditioning for USCIx_DAT[1:0] and USCIx_CTL[1:0] 292
Figure 6.11-3 Input Conditioning for USCIx_CLK 292
Figure 6.11-4 Block Diagram of Data Buffering 294
Figure 6.11-5 Data Access Structure 294
Figure 6.11-6 Transmit Data Path 295
Figure 6.11-7 Receive Data Path 296
Figure 6.11-8 Protocol-Relative Clock Generator 297
Figure 6.11-9 Basic Clock Divider Counter 298
Figure 6.11-10 Block of Timing Measurement Counter 298
Figure 6.11-11 Sample Time Counter 299
Figure 6.11-12 Event and Interrupt Structure 300
Figure 6.12-1 USCI - UART Mode Block Diagram 302
Figure 6.12-2 UART Signal Connection for Full-Duplex Communication 303
Figure 6.12-3 UART Standard Frame Format 305
Figure 6.12-4 UART Bit Timing (Data Sample Time) 307
Figure 6.12-5 UART Auto Baud Rate Control 309
Figure 6.12-6 Incoming Data Wake-Up 310
Figure 6.13-1 SPI Master Mode Application Block Diagram (x=0, 1) 335
Figure 6.13-2 SPI Slave Mode Application Block Diagram (x=0, 1) 335

Figure 6.13-3 USCI - SPI Mode Block Diagram 336
Figure 6.13-4 4-Wire Full-Duplex SPI Communication Signals (Master Mode) 337
Figure 6.13-5 4-Wire Full-Duplex SPI Communication Signals (Slave Mode) 338
Figure 6.13-6 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x0) .. 339
Figure 6.13-7 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x1) .. 339
Figure 6.13-8 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x2) .. 340
Figure 6.13-9 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x3) .. 340
Figure 6.13-10 16-bit data Length in One Word Transaction with MSB First Format 341
Figure 6.13-11 Word Suspend Interval between Two Transaction Words 342
Figure 6.13-12 Auto Slave Select (SUSPITV ≧ 0x3) 343
Figure 6.13-13 Auto Slave Select (SUSPITV < 0x3) 343
Figure 6.13-14 SPI Timing in Master Mode 346
Figure 6.13-15 SPI Timing in Master Mode (Alternate Phase of Serial Bus Clock) 346
Figure 6.13-16 SPI Timing in Slave Mode 347
Figure 6.13-17 SPI Timing in Slave Mode (Alternate Phase of Serial Bus Clock) 347
Figure 6.14-1 I^2C Bus Timing 373
Figure 6.14-2 USCI - I²C Mode Block Diagram 374
Figure 6.14-3 I^2C Protocol 375
Figure 6.14-4 START and STOP Conditions 375
Figure 6.14-5 Bit Transfer on I^2C Bus 377
Figure 6.14-6 Acknowledge on I^2C Bus 377
Figure 6.14-7 Arbitration Lost 379
Figure 6.14-8 Control I^2C Bus according to Current I^2C Status 381
Figure 6.14-9 Master Transmits Data to Slave with a 7-bit Address 382
Figure 6.14-10 Master Reads Data from Slave with a 7-bit Address 382
Figure 6.14-11 Master Transmits Data to Slave by 10-bit Address 382
Figure 6.14-12 Master Reads Data from Slave by 10-bit Address 383
Figure 6.14-13 Master Transmitter Mode Control Flow with 7-bit Address 383
Figure 6.14-14 Master Receiver Mode Control Flow with 7-bit Address 384
Figure 6.14-15 Save Mode Control Flow with 7-bit Address 385
Figure 6.14-16 GC Mode with 7-bit Address 387
Figure 6.14-17 Setup Time Wrong Adjustment 388
Figure 6.14-18 Hold Time Wrong Adjustment 389
Figure 6.14-19 I^2C Time-out Count Block Diagram 389
Figure 6.14-20 EEPROM Random Read 390
Figure 6.14-21 Protocol of EEPROM Random Read 391

Figure 6.15-1 Hardware Divider Operation Flow 412
Figure 6.16-1 ADC Control Block Diagram 419
Figure 6.16-2 ADC Peripheral Clock Control 420
Figure 6.16-3 Single Mode Conversion Timing Diagram 421
Figure 6.16-4 ADC Hardware Trigger Source 422
Figure 6.16-5 Independent Sample Mode Conversion Timing Diagram 422
Figure 6.16-6 Simultaneous Simple Mode Conversion Timing Diagram 423
Figure 6.16-7 Simultaneous Sequential 4R Mode Conversion Timing Diagram 423
Figure 6.17-1 Analog Comparator Block Diagram 442
Figure 6.17-2 Analog Comparator Controller Interrupt Sources 443
Figure 6.17-3 Comparator Hysteresis Function 443
Figure 6.17-4 Comparator Reference Voltage Block Diagram 444
Figure 6.18-1 OP Amplifier Block Diagram 456

List of Tables

Table 3-1 List of Abbreviations 17
Table 4.2-1 NuMicro® NDA102 Series Selection Guide 19
Table 4.4-1 TSSOP28 Pin Description 24
Table 4.4-2 TSSOP28 Multi-function Pin Summary 27
Table 6.2-1 Reset Value of Registers 33
Table 6.2-2 Power Mode Difference Table 37
Table 6.2-3 Clocks in Power Modes 38
Table 6.2-4 Condition of Entering Power-down Mode Again 39
Table 6.2-5 Memory Mapping Table 41
Table 6.2-6 Protected Registers 43
Table 6.2-7 Address Space Assignments for On-Chip Modules 45
Table 6.2-8 Exception Model 74
Table 6.2-9 System Interrupt Map Vector Table 75
Table 6.2-10 Vector Table Format 75
Table 6.3-1 Peripheral Clock Source Selection Table 107
Table 6.3-2 Power-down Mode Control Table 112
Table 6.4-1 Flash Memory Address Map 124
Table 6.4-2 Data Flash Table 125
Table 6.4-3 Boot Selection 134
Table 6.4-4 Boot Selection and Supports Function 134
Table 6.4-5 ISP Command Table 138
Table 6.6-1 Input Capture Mode Operation 182
Table 6.10-1 Watchdog Timer Time-out Interval Period Selection 286
Table 6.11-1 Input Signals for Different Protocols 291
Table 6.11-2 Output Signals for Different Protocols 293
Table 6.11-3 Data Transfer Events and Interrupt Handling 300
Table 6.11-4 Protocol-specific Events and Interrupt Handling 301
Table 6.12-1 Input Signals for UART Protocols 304
Table 6.12-2 Output Signals for Different Protocols 304
Table 6.12-3 Baud rate Relationship 308
Table 6.13-1 SPI Communication Signals (x=0, 1) 337
Table 6.13-2 Serial Bus Clock Configuration 338
Table 6.14-1 Relationship between I^2C Baud Rate and PCLK 388

1 GENERAL DESCRIPTION

The NuMicro® NDA102 series 32-bit microcontrollers are embedded with ARM® Cortex®-M0 core

for industrial applications which need high performance, high integration, and low cost. The Cortex®-M0 is the newest ARM® embedded processor with 32-bit performance at a cost equivalent to the traditional 8-bit microcontroller.

The NDA102 series can run up to 48 MHz and operate at 2.1V ~ 5.5V, -40℃ ~ 105℃, and thus can support a variety of industrial control applications which need high CPU performance. The NDA102 offers 29.5 Kbytes embedded program Flash, size configurable Data Flash (shared with program Flash), 2 Kbytes Flash for the ISP, 1.5 Kbytes SPROM for security, and 4 Kbytes SRAM.

Many system level peripheral functions, such as I/O Port, Timer, UART, SPI, I^2C, PWM, ADC, Watchdog Timer, Analog Comparator and Brown-out Detector, have been incorporated into the NDA102 to reduce component count, board space and system cost. These useful functions make the NDA102 powerful for a wide range of applications.

Additionally, the NDA102 series is equipped with ISP (In-System Programming) and ICP (In-Circuit Programming) functions, which allow the user to update program memory without removing the chip from the actual end product.

2 FEATURES

- Core
  - ARM® Cortex®-M0 core running up to 48 MHz
  - One 24-bit system timer
  - Supports low power Idle mode
  - A single-cycle 32-bit hardware multiplier
  - NVIC for the 32 interrupt inputs, each with 4-level of priority
  - Supports Serial Wire Debug (SWD) interface and two watchpoints/four breakpoints
- Built-in LDO for wide operating voltage ranged: 2.1V to 5.5V
- Memory
  - 29.5 Kbytes Flash memory for program memory (APROM)
  - Configurable Flash memory for data memory (Data Flash)
  - 2 KB Flash memory for loader (LDROM)
  - Three 0.5 KB Flash memory for security protection (SPROM)
  - 4 KB SRAM for internal scratch-pad RAM (SRAM)
- Clock Control
  - Programmable system clock source
    - ◆ Switch clock sources on-the-fly
  - 4 ~ 24 MHz external crystal input (HXT)
  - 32.768 kHz external crystal input (LXT) for idle wake-up and system operation clock
  - 48 MHz internal oscillator (HIRC) (±1% accuracy at 25^0C, 5V)
    - ◆ Dynamically calibrating the HIRC OSC to 48 MHz ±1% from -40℃ to 105℃ by external 32.768K crystal oscillator (LXT)
  - 10 kHz internal low-power oscillator (LIRC) for Watchdog Timer and idle wake-up
- I/O Port
  - Up to 22 general-purpose I/O (GPIO) pins and 1 Reset pin for QFN-33 package
  - Four I/O modes:
    - ◆ Quasi-bidirectional input/output
    - ◆ Push-Pull output
    - ◆ Open-Drain output
    - ◆ Input only with high impendence
  - Optional TTL/Schmitt trigger input
  - I/O pin can be configured as interrupt source with edge/level setting
  - Supports high driver and high sink I/O mode

- Supports software selectable slew rate control
- **GPIO built-in Pull-up/Pull-low resistor for selection.**

- Timer
  - Provides two channel 32-bit Timers; one 8-bit pre-scalar counter with 24-bit up-timer for each timer
  - Independent clock source for each timer
  - Provides One-shot, Periodic, Toggle and Continuous operation modes
  - 24-bit up counter value is readable through CNT (Timer Data Register)
  - Provides trigger counting/free counting/counter reset function triggered by external capture pin or internal comparator signal
  - Supports event counter function
  - Supports Toggle Output mode
  - Supports wake-up from Idle or Power-down mode
  - Timer0, Timer1 and Systick provided with Continuous capture function to capture at most 4 edges continuously on one signal
- Continuous Capture
  - Timer0, Timer1 and Systick have support Continuous capture function can Continuous Capture 4 edge on one signal
- Enhanced Input Capture
  - One unit of 24-bit input capture counter
  - Capture surce:
    - I/O inputs: ECAP0, ECAP1 and ECAP2
    - PWM Trigger
    - ADC Trigger
- WDT (Watchdog Timer)
  - Programmable clock source and time-out period
  - Supports wake-up function in Power-down mode and Idle mode
  - Interrupt or reset selectable on watchdog time-out
- PWM
  - Supports a built-in 16-bit PWM clock generators, providing six PWM outputs or three complementary paired PWM outputs
  - Shared same as clock source, clock divider, period and dead-zone generator
  - Supports group/synchronous/independent/ complementary modes
  - Supports One-shot or Auto-reload mode
  - Supports Edge-aligned and Center-aligned type
  - Supports Asymmetric mode
  - Programmable dead-zone insertion between complementary channels
  - Each output has independent polarity setting control

- Hardware fault brake and software brake protections
- Supports rising, falling, central, period, and fault break interrupts
- Supports duty/period trigger A/D conversion
- Timer comparing matching event trigger PWM to do phase change
- Supports comparator event trigger PWM to force PWM output low for current period
- Provides interrupt accumulation function

- USCI (Universal Serial Control Interface Controller)
  - Two USCI devices
  - Supports to be configured as UART, SPI or I²C individually
  - Supports programmable baud-rate generator
- ADC (Analog-to-Digital Converter)
  - 12-bit ADC with 700 kSPS
  - Supports 2 sample/hold
  - Up to 8-ch single-end input from I/O and one internal input from band-gap.
  - Conversion started either by software trigger, PWM trigger, ACMP trigger or external pin trigger
  - Supports temperature sensor for measurement chip temperature
  - Supports Simultaneous and Sequential function to continuous conversion 4 channels maximum.
- Programmable Gain Amplifier (PGA)
  - Supports 8 level gain selects from 1, 2, 3, 5, 7, 9, 11 and 13
  - Unity gain frequency up to 8 MHz
- Analog Comparator
  - Two analog comparators with programmable 16-level internal voltage reference
  - Built-in CRV (comparator reference voltage)
  - Supports Hysteresis function
  - Interrupt when compared results changed
- Hardware Divider
  - Signed (two's complement) integer calculation
  - 32-bit dividend with 16-bit divisor calculation capacity
  - 32-bit quotient and 32-bit remainder outputs (16-bit remainder with sign extends to 32-bit)
  - Divided by zero warning flag
  - 6 HCLK clocks taken for one cycle calculation
  - Waiting for calculation ready automatically when reading quotient and remainder
- ISP (In-System Programming) and ICP (In-Circuit Programming)

- BOD (Brown-out Detector)
  - 8 programmable threshold levels: 4.3V/4.0V/3.7V/3.0V/2.7V/2.4V/2.2V/2.0V
  - Supports Brown-out interrupt and reset option
- 96-bit unique ID
- LVR (Low Voltage Reset)
- Operating Temperature: -40℃~105℃
- Reliability: EFT > ± 4KV, ESD HBM pass 4KV
- Packages:
  - Green package (RoHS)
  - 28-pin TSSOP

3 ABBREVIATIONS

Acronym	Description
ACMP	Analog Comparator Controller
ADC	Analog-to-Digital Converter
AHB	Advanced High-Performance Bus
APB	Advanced Peripheral Bus
BOD	Brown-out Detection
BPWM	Basic Pulse Width Modulation
DAP	Debug Access Port
EPWM	Enhanced Pulse Width Modulation
FIFO	First In, First Out
FMC	Flash Memory Controller
GPIO	General-Purpose Input/Output
HCLK	The Clock of Advanced High-Performance Bus
HIRC	48 MHz Internal High Speed RC Oscillator
HXT	4~24 MHz External High Speed Crystal Oscillator
ICP	In Circuit Programming
ISP	In System Programming
ISR	Interrupt Service Routine
LDO	Low Dropout Regulator
LIRC	10 kHz internal low speed RC oscillator (LIRC)
LXT	32.768 kHz External Low Speed Crystal Oscillator
NVIC	Nested Vectored Interrupt Controller
PCLK	The Clock of Advanced Peripheral Bus
PWM	Pulse Width Modulation
SPI	Serial Peripheral Interface
SPS	Samples per Second
TMR	Timer Controller
UART	Universal Asynchronous Receiver/Transmitter
UCID	Unique Customer ID
WDT	Watchdog Timer

Table 3-1 List of Abbreviations

4 PARTS INFORMATION LIST AND PIN CONFIGURATION

4.1 NuMicro® NDA102 Naming Rule

Figure 4.1-1 NuMicro® NDA102 Series Selection Code

4.2 NuMicro® NDA102 Series Selection Guide

* USCI can be set to UART, SPI or I^2C

Part Number	APROM	RAM	Data Flash	ISP Loader ROM	I/O	Timer	Connectivity	Comp.	PWM	ADC	PGA	ISP ICP IAP	IRC 10 kHz 48 MHz	Package
							USCI*							
NDA102EC1	29.5 KB	4 KB	Configurable	2.5 KB	up to 22	2x32-bit	2	2	8	8x12-bit	v	v	v	TSSOP28

Table 4.2-1 NuMicro® NDA102 Series Selection Guide

4.3 Pin Configuration

4.3.1 TSSOP 28-Pin

Figure 4.3-1 NuMicro® NDA102 Series TSSOP 28-pin Diagram

Figure 4.3-2 NuMicro® NDA102 Series TSSOP 28-pin Multi-function Diagram

4.4 Pin Description

4.4.1 NDA102 Series Pin Description

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFP)

PA.0 MFP0 means SYS_GPA_MFP[3:0]=0x0.

PA.4 MFP5 means SYS_GPA_MFP[19:16]=0x5.

MFP only configures the ouput data or input data of PAD; the direction of PAD is configured by PMD.

The priority of MFP in the same multi-function was GPA > GPB > GPC > GPD.

The type A of multi-function needs to be configured to be input port.

4.4.1.1 NDA102 Series TSSOP28 Pin Description

Pin No.	Pin Name	Type	MFP*	Description
1	V_{DD}	A	MFP0	Power supply for I/O ports and LDO source for internal PLL and digital function.
2	PD.6	I/O	MFP0	General purpose digital I/O pin.
	UART0_RXD	I	MFPB	Data receiver input pin for UART0.
3	PB.0	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH0	A	MFP2	ADC0 channel0 analog input.
	ACMP0_P0	A	MFP4	Analog comparator0 positive input pin.
	ECAP_P0	I	MFP7	Enhanced Input Capture input pin
4	PB.1	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH1	A	MFP2	ADC0 channel1 analog input.
	ACMP0_P1	A	MFP4	Analog comparator0 positive input pin.
	ECAP_P1	I	MFP7	Enhanced Input Capture input pin
5	PB.2	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH2	A	MFP2	ADC0 channel2 analog input.
	BPWM_CH1	I/O	MFP3	PWM channel1 output/capture input.
	ACMP0_P2	A	MFP4	Analog comparator0 positive input pin.
	ECAP_P2	I	MFP7	Enhanced Input Capture input pin
6	PB.4	I/O	MFP0	General purpose digital I/O pin.
	ADC1_CH0	A	MFP2	ADC1 channel0 analog input.
	ACMP0_N	A	MFP4	Analog comparator0 negative input pin.
	TM1	I/O	MFP7	Timer1 event counter input / toggle output
7	PC.1	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH4	A	MFP2	ADC0 channel4 analog input.

Pin No.	Pin Name	Type	MFP*	Description
	STADC	I	MFP3	ADC external trigger input.
	ACMP0_P3	A	MFP4	Analog comparator0 positive input pin.
	ACMP1_P1	A	MFP5	Analog comparator1 positive input pin.
	SPI0_MOSI	I/O	MFP9	SPI0 1st MOSI (Master Out, Slave In) pin.
	SPI1_MISO	I/O	MFPA	SPI1 MISO (Master In, Slave Out) pin.
8	nRESET	I	MFP0	External reset input: active LOW, with an internal pull-up. Set this pin low reset to initial state.
9	PB.3	I/O	MFP0	General purpose digital I/O pin.
	ACMP1_N	A	MFP5	Analog comparator1 negative input pin.
	PGA_I	A	MFP6	PGA input pin
	TM0	I/O	MFP7	Timer0event counter input / toggle output
10	PC.2	I/O	MFP0	General purpose digital I/O pin.
	ADC1_CH2	A	MFP2	ADC1 channel2 analog input.
	BRAKE	I	MFP3	Brake input pin of EPWM.
	CCAP_P1	I	MFP7	Timer Continuous Capture input pin
	I2C1_SDA	I/O	MFP8	I^2C1 data input/output pin.
	SPI0_MISO	I/O	MFP9	SPI0 1st MISO (Master In, Slave Out) pin.
	SPI1_MOSI	I/O	MFPA	SPI1 MOSI (Master Out, Slave In) pin.
	UART1_RXD	I	MFPB	Data receiver input pin for UART1.
11	PD.2	I/O	MFP0	General purpose digital I/O pin.
	ICE_DAT	I/O	MFP1	Serial wired debugger data pin
	ADC1_CH1	A	MFP2	ADC1 channel1 analog input.
	CCAP_P0	I	MFP7	Timer Continuous Capture input pin
	I2C0_SDA	I/O	MFP8	I^2C0 data input/output pin.
	SPI0_MOSI	I/O	MFP9	SPI0 1st MOSI (Master Out, Slave In) pin.
	SPI1_MISO	I/O	MFPA	SPI1 MISO (Master In, Slave Out) pin.
	UART0_RXD	I	MFPB	Data receiver input pin for UART0.
12	PD.3	I/O	MFP0	General purpose digital I/O pin.
	BPWM_CH1	I/O	MFP3	PWM channel1 output/capture input.
	UART1_TXD	O	MFPB	Data transmitter output pin for UART1.
13	NC			No Connection
14	NC			No Connection
15	PC.0	I/O	MFP0	General purpose digital I/O pin.
	ADC0_CH3	A	MFP2	ADC0 channel3 analog input.

Pin No.	Pin Name	Type	MFP*	Description
	BPWM_CH0	I/O	MFP3	PWM channel0 output/capture input.
	ACMP1_P0	A	MFP5	Analog comparator1 positive input pin.
	I2C1_SCL	I/O	MFP8	I^2C1 clock pin.
	SPI0_SS	I/O	MFP9	SPI0 slave select pin.
	SPI1_CLK	I/O	MFPA	SPI1 serial clock pin
	UART1_TXD	O	MFPB	Data transmitter output pin for UART1.
16	PD.4	I/O	MFP0	General purpose digital I/O pin.
	BPWM_CH0	I/O	MFP3	PWM channel0 output/capture input.
	UART1_RXD	I	MFPB	Data receiver input pin for UART1.
17	PD.1	I/O	MFP0	General purpose digital I/O pin.
	ICE_CLK	I	MFP1	Serial wired debugger clock pin
	ACMP1_P2	A	MFP5	Analog comparator1 positive input pin.
	I2C0_SCL	I/O	MFP8	I^2C0 clock pin.
	SPI0_CLK	I/O	MFP9	SPI0 serial clock pin.
	SPI1_SS	I/O	MFPA	SPI1 slave select pin
	UART0_TXD	O	MFPB	Data transmitter output pin for UART0.
18	PC.3	I/O	MFP0	General purpose digital I/O pin.
	ACMP1_O	O	MFP5	Analog comparator1 output.
	PGA_O	A	MFP6	PGA output pin
	SPI0_CLK	I/O	MFP9	SPI0 serial clock pin.
	SPI1_SS	I/O	MFPA	SPI1 slave select pin
19	PD.5	I/O	MFP0	General purpose digital I/O pin.
	UART0_TXD	O	MFPB	Data transmitter output pin for UART0.
20	PA.5	I/O	MFP0	General purpose digital I/O pin.
	XT_OUT	O	MFP1	External 4~24 MHz (high speed) crystal output pin.
	EPWM_CH5	I/O	MFP3	PWM channel5 output/capture input.
	ACMP0_O	O	MFP4	Analog comparator0 output.
21	PA.4	I/O	MFP0	General purpose digital I/O pin.
	XT_IN	I	MFP1	External 4~24 MHz (high speed) crystal input pin.
	EPWM_CH4	I/O	MFP3	PWM channel4 output/capture input.
22	PA.3	I/O	MFP0	General purpose digital I/O pin.
	EPWM_CH3	I/O	MFP3	PWM channel3 output/capture input.
	I2C0_SCL	I/O	MFP8	I^2C0 clock pin.

Pin No.	Pin Name	Type	MFP*	Description
	SPI0_CLK	I/O	MFP9	SPI0 serial clock pin.
	SPI1_SS	I/O	MFPA	SPI1 slave select pin
	UART0_TXD	O	MFPB	Data transmitter output pin for UART0.
23	PA.2	I/O	MFP0	General purpose digital I/O pin.
	EPWM_CH2	I/O	MFP3	PWM channel2 output/capture input.
	I2C0_SDA	I/O	MFP8	I^2C0 data input/output pin.
	SPI0_MOSI	I/O	MFP9	SPI0 1st MOSI (Master Out, Slave In) pin.
	SPI1_MISO	I/O	MFPA	SPI1 MISO (Master In, Slave Out) pin.
	UART0_RXD	I	MFPB	Data receiver input pin for UART0.
24	PA.1	I/O	MFP0	General purpose digital I/O pin.
	EPWM_CH1	I/O	MFP3	PWM channel1 output/capture input.
	I2C1_SDA	I/O	MFP8	I^2C1 data input/output pin.
	SPI0_MISO	I/O	MFP9	SPI0 1st MISO (Master In, Slave Out) pin.
	SPI1_MOSI	I/O	MFPA	SPI1 MOSI (Master Out, Slave In) pin.
	UART1_RXD	I	MFPB	Data receiver input pin for UART1.
25	PA.0	I/O	MFP0	General purpose digital I/O pin.
	CLKO	O	MFP1	Clock Out
	EPWM_CH0	I/O	MFP3	PWM channel0 output/capture input.
	I2C1_SCL	I/O	MFP8	I^2C1 clock pin.
	SPI0_SS	I/O	MFP9	SPI0 slave select pin.
	SPI1_CLK	I/O	MFPA	SPI1 serial clock pin
	UART1_TXD	O	MFPB	Data transmitter output pin for UART1.
26	PC.4	I/O	MFP0	General purpose digital I/O pin.
	ECAP_P3	I	MFP7	Enhanced Input Capture input pin
27	LDO_CAP	A	MFP0	LDO output pin.
28	V_{SS}	A	MFP0	Ground pin for digital circuit.

Table 4.4-1 TSSOP28 Pin Description

4.4.2 GPIO Multi-function Pin Summary

MFP* = Multi-function pin. (Refer to section SYS_GPx_MFP)

PA.0 MFP0 means SYS_GPA_MFP[3:0]=0x0.

PA.4 MFP5 means SYS_GPA_MFP[19:16]=0x5.

Group	Pin Name	GPIO	MFP*	Type	Description
ACMP0	ACMP0_P0	PB.0	MFP4	A	Comparator0 positive input pin.
	ACMP0_P1	PB.1	MFP4	A	Comparator0 positive input pin.
	ACMP0_P2	PB.2	MFP4	A	Comparator0 positive input pin.
	ACMP0_N	PB.4	MFP4	A	Comparator0 negative input pin.
	ACMP0_P3	PC.1	MFP4	A	Comparator0 positive input pin.
	ACMP0_O	PA.5	MFP4	O	Comparator0 output pin.
ACMP1	ACMP1_P1	PC.1	MFP5	A	Comparator1 positive input pin.
	ACMP1_N	PB.3	MFP5	A	Comparator1 negative input pin.
	ACMP1_O	PC.3	MPF5	O	Comparator1 output pin.
	ACMP1_P2	PD.1	MFP5	A	Comparator1 positive input pin.
	ACMP1_P0	PC.0	MFP5	A	Comparator1 positive input pin.
ADC0	ADC0_CH0	PB.0	MFP2	A	ADC0 analog input channel 0.
	ADC0_CH1	PB.1	MFP2	A	ADC0 analog input channel 1.
	ADC0_CH2	PB.2	MFP2	A	ADC0 analog input channel 2.
	ADC0_CH4	PC.1	MFP2	A	ADC0 analog input channel 4.
	ADC0_CH3	PC.0	MFP2	A	ADC0 analog input channel 3.
ADC1	ADC1_CH0	PB.4	MFP2	A	ADC1 analog input channel 0.
	ADC1_CH2	PC.2	MFP2	A	ADC1 analog input channel 2.
	ADC1_CH1	PD.2	MFP2	A	ADC1 analog input channel 1.
BPWM	BPWM_CH1	PB.2	MFP3	O	Basic PWM channel 1 output
	BPWM_CH0	PC.0	MFP3	O	Basic PWM channel 0 output
	BPWM_CH1	PD.3	MFP3	O	Basic PWM channel 1 output
	BPWM_CH0	PD.4	MFP3	O	Basic PWM channel 0 output
CCAP	CCAP_P1	PC.2	MFP7	I	Continuous Capture Input
	CCAP_P0	PD.2	MFP7	I	Continuous Capture Input
CLKO	CLKO	PA.0	MFP1	O	Clock output pin.
ECAP	ECAP_P0	PB.0	MFP7	I	Input capture channel 0
	ECAP_P1	PB.1	MFP7	I	Input capture channel 1
	ECAP_P2	PB.2	MFP7	I	Input capture channel 2

EPWM	BRAKE	PC.2	MFP3	I	EPWM brake pin.
	EPWM_CH5	PA.5	MFP3	O	Enhanced PWM output pin.
	EPWM_CH4	PA.4	MFP3	O	Enhanced PWM output pin.
	EPWM_CH3	PA.3	MFP3	O	Enhanced PWM output pin.
	EPWM_CH2	PA.2	MFP3	O	Enhanced PWM output pin.
	EPWM_CH1	PA.1	MFP3	O	Enhanced PWM output pin.
	EPWM_CH0	PA.0	MFP3	O	Enhanced PWM output pin.
I^2C	I2C1_SDA	PC.2	MFP8	I/O	I^2C1 data pin.
	I2C0_SDA	PD.2	MFP8	I/O	I^2C0 data pin.
	I2C0_SCL	PD.1	MFP8	I/O	I^2C0 clock pin.
	I2C1_SCL	PC.0	MFP8	I/O	I^2C1 clock pin.
	I2C0_SCL	PA.3	MFP8	I/O	I^2C0 clock pin.
	I2C0_SDA	PA.2	MFP8	I/O	I^2C0 data pin.
	I2C1_SDA	PA.1	MFP8	I/O	I^2C1 data pin.
	I2C1_SCL	PA.0	MFP8	I/O	I^2C1 clock pin.
ICE	ICE_DAT	PD.2	MFP1	I/O	Serial wired debugger data pin
	ICE_CLK	PD.1	MFP1	I	Serial wired debugger clock pin
nRESET	nRESET			I	External reset pin, internal pull-high.
PGA	PGA_I	PB.3	MFP6	A	PGA analog input pin.
	PGA_O	PC.3	MFP6	A	PGA analog output pin.
SPI0	SPI0_MOSI	PC.1	MFP9	I/O	SPI0 MOSI (Master Out, Slave In) pin.
	SPI0_MISO	PC.2	MFP9	I/O	SPI0 MISO (Master In, Slave Out) pin.
	SPI0_MOSI	PD.2	MFP9	I/O	SPI0 MOSI (Master Out, Slave In) pin.
	SPI0_CLK	PC.3	MFP9	I/O	SPI0 clock pin.
	SPI0_CLK	PD.1	MFP9	I/O	SPI0 clock pin.
	SPI0_SS	PC.0	MFP9	I	SPI0 slave selection pin.
	SPI0_CLK	PA.3	MFP9	I/O	SPI0 clock pin.
	SPI0_MOSI	PA.2	MFP9	I/O	SPI0 MOSI (Master Out, Slave In) pin.
	SPI0_MISO	PA.1	MFP9	I/O	SPI0 MISO (Master In, Slave Out) pin.
	SPI0_SS	PA.0	MFP9	I	SPI0 slave selection pin.
SPI1	SPI1_MISO	PC.1	MFPA	I/O	SPI1 MISO (Master In, Slave Out) pin
	SPI1_MOSI	PC.2	MFPA	I/O	SPI1 MOSI (Master Out, Slave In) pin.
	SPI1_MISO	PD.2	MFPA	I/O	SPI1 MISO (Master In, Slave Out) pin
	SPI1_SS	PC.3	MFPA	I/O	SPI1 Slave Select

	SPI1_SS	PD.1	MFPA	I/O	SPI1 Slave Select
	SPI1_CLK	PC.0	MFPA	I/O	SPI1 clock pin.
	SPI1_SS	PA.3	MFPA	I	SPI1 slave selection pin.
	SPI1_MISO	PA.2	MFPA	I/O	SPI1 MISO (Master In, Slave Out) pin.
	SPI1_MOSI	PA.1	MFPA	I/O	SPI1 MOSI (Master Out, Slave In) pin.
	SPI1_CLK	PA.0	MFPA	I/O	SPI1 clock pin.
STADC	STADC	PC.1	MFP3	I	External ADC trigger input pin.
TM0	TM0	PB.3	MFP7	I	Timer0 event counter input / toggle output
TM1	TM1	PB.4	MFP7	I	Timer1 event counter input / toggle output
UART0	UART0_RXD	PD.2	MFPB	I	UART0 data receiver input pin.
	UART0_TXD	PD.1	MFPB	O	UART0 data transmitter output pin.
	UART0_TXD	PA.3	MFPB	O	UART0 data transmitter output pin.
	UART0_RXD	PA.2	MFPB	I	UART0 data receiver input pin.
	UART0_TXD	PD.5	MFPB	O	UART0 data transmitter output pin.
	UART0_RXD	PD.6	MFPB	I	UART0 data receiver input pin.
UART1	UART1_RXD	PC.2	MFPB	I	UART1 data receiver input pin.
	UART1_TXD	PC.0	MFPB	O	UART1 data transmitter output pin.
	UART1_RXD	PA.1	MFPB	I	UART1 data receiver input pin.
	UART1_TXD	PA.0	MFPB	O	UART1 data transmitter output pin.
	UART1_TXD	PD.3	MFPB	O	UART1 data transmitter output pin.
	UART1_RXD	PD.4	MFPB	I	UART1 data receiver input pin.
XT	XT_OUT	PA.5	MPF1	A	External crystal output pin.
	XT_IN	PA.4	MFP1	A	External crystal input pin.

Table 4.4-2 TSSOP28 Multi-function Pin Summary

5 BLOCK DIAGRAM

5.1 NuMicro® NDA102 Block Diagram

Figure 5.1-1 NuMicro® NDA102 Block Diagram

6 FUNCTIONAL DESCRIPTION

6.1 ARM® Cortex®-M0 Core

6.1.1 Overview

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 6.1-1 shows the functional controller of processor.

Figure 6.1-1 Functional Block Diagram

6.1.2 Features

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
  - System interface supported with little-endian data accesses
  - 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 the Wait For Interrupt (WFI), Wait For Event (WFE) instructions, or 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)

6.2 System Manager

6.2.1 Overview

System management includes the following sections:

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

6.2.2 System Reset

The system reset can be issued by one of the events listed below. These reset event flags can be read from SYS_RSTSTS register to determine the reset source. Hardware reset can reset chip through peripheral reset signals. Software reset can trigger reset through control registers.

- Hardware Reset Sources
  - Power-on Reset (POR)
  - Low level on the nRESET pin
  - Watchdog Timer Time-out Reset (WDT)
  - Low Voltage Reset (LVR)
  - Brown-out Detector Reset (BOD Reset)
- Software Reset Sources
  - CHIP Reset will reset whole chip by writing 1 to CHIPRST (SYS_IPRST0[0])
  - MCU Reset to reboot but keeping the booting setting from APROM or LDROM by writing 1 to SYSRESETREQ (SCS_AIRCR[2])
  - CPU Reset for Cortex®-M0 core Only by writing 1 to CPURST (SYS_IPRST0[1])

Figure 6.2-1 System Reset Resources

There are a total of 9 reset sources in the NuMicro® family. In general, CPU reset is used to reset Cortex®-M0 only; the other reset sources will reset Cortex®-M0 and all peripherals. However, there are small differences between each reset source and they are listed in Table 6.2-5.

Reset Sources / Register	POR	nRESET	WDT	LVR	BOD	CHIP	MCU	CPU
SYS_RSTSTS	0x001	Bit 1 = 1	Bit 2 = 1	0x001	Bit 4 = 1	Bit 0 = 1	Bit 5 = 1	Bit 7 = 1
CHIPRST (SYS_IPRST0[0])	0x0	-	-	-	-	-	-	-
BODEN (SYS_BODCTL[0])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	Reload from CONFIG0	Reload from CONFIG0	-
BODVL (SYS_BODCTL[2:1])								
BODRSTEN (SYS_BODCTL[3])								
XTLEN (CLK_PWRCTL[1:0])	0x0	0x0	0x0	0x0	0x0	0x0	0x0	
WDTCKEN (CLK_APBCLK0[0])	0x1	-	0x1	-	-	0x1	-	-

HCLKSEL (CLK_CLKSEL0[1:0])	0x8	0x8	0x8	0x8	0x8	0x8	0x8	-
WDTSEL (CLK_CLKSEL1[1:0])	0x3	0x3	-	-	-	-	-	-
XLTSTB (CLK_STATUS[0])	0x0	-	-	-	-	-	-	-
LIRCSTB (CLK_STATUS[3])	0x0							
HIRCSTB (CLK_STATUS[4])	0x0	-	-	-	-	-	-	-
CLKSFAIL (CLK_STATUS[7])	0x0	0x0	-	-	-	-	-	-
WDT_CTL	0x0700	0x0700	0x0700	0x0700	0x0700	0x0700	-	-
WDT_ALTCTL	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDT_RLDCNT	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDT_CTL	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	0x3F0800	-	-
WWDT_STATUS	0x0000	0x0000	0x0000	0x0000	0x0000	0x0000	-	-
WWDT_CNT	0x3F	0x3F	0x3F	0x3F	0x3F	0x3F	-	-
BS (FMC_ISPCTL[1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	-
ISPEN (FMC_ISPCTL[16])								
FMC_DFBA	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	Reload from CONFIG1	-	-
CBS (FMC_ISPSTS[2:1])	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	Reload from CONFIG0	-	-
VECMAP (FMC_ISPSTS[20:9])	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	Reload base on CONFIG0	-	-
Other Peripheral Registers	Reset Value							
FMC Registers	Reset Value							
Note: '-' means that the value of register keeps original setting.								

Table 6.2-1 Reset Value of Registers

6.2.2.1 nRESET Reset

The nRESET reset means to generate a reset signal by pulling low nRESET pin, which is an asynchronous reset input pin and can be used to reset system at any time. When the nRESET voltage is lower than 0.2 V_{DD} and the state keeps longer than 16.8 us (glitch filter), chip will be reset. The nRESET reset will control the chip in reset state until the nRESET voltage rises above

0.7 V_{DD} and the state keeps longer than 36 us (glitch filter). The PINRF (SYS_RSTSTS[1]) will be set to 1 if the previous reset source is nRESET reset. Figure 6.2-2 shows the nRESET reset waveform.

Figure 6.2-2 nRESET Reset Waveform

6.2.2.2 Power-On Reset (POR)

The Power-on reset (POR) is used to generate a stable system reset signal and forces the system to be reset when power-on to avoid unexpected behavior of MCU. When applying the power to MCU, the POR module will detect the rising voltage and generate reset signal to system until the voltage is ready for MCU operation. At POR reset, the PORF (SYS_RSTSTS[0]) will be set to 1 to indicate there is a POR reset event. The PORF (SYS_RSTSTS[0]) bit can be cleared by writing 1 to it. Figure 6.2-3 shows the waveform of Power-On reset.

Figure 6.2-3 Power-on Reset (POR) Waveform

6.2.2.3 Low Voltage Reset (LVR)

Low Voltage Reset detects AV_{DD} during system operation. When the AV_{DD} voltage is lower than V_{LVR} and the state keeps longer than De-glitch time (16*HCLK cycles), chip will be reset. The LVR reset will control the chip in reset state until the AV_{DD} voltage rises above V_{LVR} and the state

keeps longer than De-glitch time. The PINRF (SYS_RSTSTS[1]) will be set to 1 if the previous reset source is nRESET reset. Figure 6.2-4 shows the Low Voltage Reset waveform.

Figure 6.2-4 Low Voltage Reset (LVR) Waveform

6.2.2.4 Brown-out Detector Reset (BOD Reset)

If the Brown-out Detector (BOD) function is enabled by setting the Brown-out Detector Enable Bit BODEN (SYS_BODCTL[0]), Brown-Out Detector function will detect AV_{DD} during system operation. When the AV_{DD} voltage is lower than V_{BOD} which is decided by BODEN (SYS_BODCTL[0]) and BODVL (SYS_BODCTL[2:1]) and the state keeps longer than De-glitch time (Max(20*HCLK cycles, 1*LIRC cycle)), chip will be reset. The BOD reset will control the chip in reset state until the AV_{DD} voltage rises above V_{BOD} and the state keeps longer than De-glitch time. The default value of BODEN, BODVL and BODRSTEN is set by Flash controller user configuration register CBODEN (CONFIG0[12]), CBOV (CONFIG0[15:13]) and CBORST (CONFIG0[12]) respectively. User can determine the initial BOD setting by setting the CONFIG0 register. Figure 6.2-5 shows the Brown-Out Detector waveform.

Figure 6.2-5 Brown-out Detector (BOD) Waveform

6.2.2.5 Watchdog Timer Reset

In most industrial applications, system reliability is very important. To automatically recover the MCU from failure status is one way to improve system reliability. The watchdog timer (WDT) is widely used to check if the system works fine. If the MCU is crashed or out of control, it may cause the watchdog time-out. User may decide to enable system reset during watchdog time-out to recover the system and take action for the system crash/out-of-control after reset.

Software can check if the reset is caused by watchdog time-out to indicate the previous reset is a watchdog reset and handle the failure of MCU after watchdog time-out reset by checking WDTRF (SYS_RSTSTS[2]).

6.2.2.6 CPU Reset, CHIP Reset and SYSTEM Reset

The CPU Reset means only Cortex®-M0 core is reset and all other peripherals remain the same status after CPU reset. User can set the CPURST (SYS_IPRST0[1]) to 1 to assert the CPU Reset signal.

The CHIP Reset is same with Power-On Reset. The CPU and all peripherals are reset and BS (FMC_ISPCTL[1]) bit is automatically reloaded from CONFIG setting. User can set the CHIPRST (SYS_IPRST0[0]) to 1 to assert the CHIP Reset signal.

The MCU Reset is similar with CHIP Reset. The difference is that BS (FMC_ISPCTL[1]) will not be reloaded from CONFIG setting and keep its original software setting for booting from APROM or LDROM. User can set the SYSRESETREQ (SCS_AIRCR[2]) to 1 to assert the MCU Reset.

6.2.3 Power Modes and Wake-up Sources

There are several wake-up sources in Idle mode and Power-down mode. Table 6.2-2 lists the available clocks for each power mode.

Power Mode	Normal Mode	Idle Mode	Power-Down Mode
Definition	CPU is in active state	CPU is in sleep state	CPU is in sleep state and all clocks stop except LXT and LIRC. SRAM content retended.
Entry Condition	Chip is in normal mode after system reset released	CPU executes WFI instruction.	CPU sets sleep mode enable and power down enable and executes WFI instruction.
Wake-up Sources	N/A	All interrupts	WDT, I²C, Timer, UART, SPI, ACMP, BOD and GPIO
Available Clocks	All	All except CPU clock	LXT and LIRC
After Wake-up	N/A	CPU back to normal mode	CPU back to normal mode

Table 6.2-2 Power Mode Difference Table

Figure 6.2-6 Power Mode State Machine

1. LXT (32768 Hz XTL) ON or OFF depends on SW setting in run mode.

2. LIRC (10 kHz OSC) ON or OFF depends on S/W setting in run mode.

3. If TIMER clock source is selected as LIRC/LXT and LIRC/LXT is on.

4. If WDT clock source is selected as LIRC and LIRC is on.

	Normal Mode	Idle Mode	Power-Down Mode
HXT (4~20 MHz XTL)	ON	ON	Halt
HIRC (12/16 MHz OSC)	ON	ON	Halt

LXT (32768 Hz XTL)	ON	ON	ON/OFF[1]
LIRC (10 kHz OSC)	ON	ON	ON/OFF[2]
LDO	ON	ON	ON
CPU	ON	Halt	Halt
HCLK/PCLK	ON	ON	Halt
SRAM retention	ON	ON	ON
FLASH	ON	ON	Halt
GPIO	ON	ON	Halt
TIMER	ON	ON	ON/OFF[3]
BPWM	ON	ON	Halt
EPWM	ON	ON	Halt
WDT	ON	ON	ON/OFF[4]
USCI	ON	ON	Halt
ADC	ON	ON	Halt
ACMP	ON	ON	Halt
ECAP	ON	ON	Halt
HDIV	ON	ON	Halt
PGA	ON	ON	Halt

Table 6.2-3 Clocks in Power Modes

Wake-up sources in Power-down mode:

WDT, I²C, Timer, UART, SPI, BOD, ACMP and GPIO

After chip enters power down, the following wake-up sources can wake chip up to normal mode. Table 6.2-4 lists the condition about how to enter Power-down mode again for each peripheral.

*User needs to wait this condition before setting PDEN (CLK_PWRCTL[7]) and execute WFI to enter Power-down mode.

Wake-Up Source	Wake-Up Condition	System Can Enter Power-Down Mode Again Condition*
BOD	Brown-Out Detector Interrupt	After software writes 1 to clear SYS_BODCTL[BODIF].
GPIO	GPIO Interrupt	After software write 1 to clear the Px_INTSRC[n] bit.
TIMER	Timer Interrupt	After software writes 1 to clear TWKF (TIMERx_INTSTS[1]) and TIF (TIMERx_INTSTS[0]).
WDT	WDT Interrupt	After software writes 1 to clear WKF (WDT_CTL[5]) (Write Protect).
USCI UART	Incoming data wake-up	After software writes 1 to clear WKF (UUART_WKSTS[0]).

USCI SPI	SS transaction wake-up	After software writes 1 to clear WKF (USPI_WKSTS[0]).
USCI I^2C	Data toggle	After software writes 1 to clear WKF (UI2C_WKSTS[0]).
	Address match	After software writes 1 to clear WKAKDONE (UI2C_PROTSTS[16], then writes 1 to clear WKF (UI2C_WKSTS[0]).
ACMP	Comparator Power-down Wake-Up Interrupt	After software writes 1 to clear ACMPF0 (ACMP_STATUS[0]) and ACMPF1 (ACMP_STATUS[1]).

Table 6.2-4 Condition of Entering Power-down Mode Again

6.2.4 System Power Architecture

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

- Analog power from AV_{DD} and AV_{SS} provides the power for analog components operation. AV_{DD} must be equal to V_{DD} to avoid leakage current.
- Digital power from V_{DD} and V_{SS} supplies power to the I/O pins and internal regulator which provides a fixed 1.5V power for digital operation.
- A built-in capacitor for internal voltage regulator

The output of internal voltage regulator, LDO, does not require an external capacitor and doesn't bond out to external pin. Analog power (AV_{DD}) should be the same voltage level of the digital power (V_{DD}).

Figure 6.2-7 NuMicro® NDA102 Series Power Architecture Diagram

6.2.5 System Memory Mapping

	NDA102EC1	
4 GB	Reserved	0xFFFF_FFFF | 0xE000_F000
	System Control	0xE000_EFFF 0xE000_E000
	Reserved	0xE000_E00F | 0x6002_0000
	Reserved	0x6001_FFFF 0x6000_0000
	Reserved	0x5FFF_FFFF | 0x5020_0000
	AHB	0x501F_FFFF 0x5000_0000
	Reserved	0x4FFF_FFFF | 0x4020_0000
	APB	0x401F_FFFF | 0x4000_0000
1 GB	Reserved	0x3FFF_FFFF | 0x2000_1000
	4 KB SRAM	0x2000_0FFF 0x2000_0000
0.5 GB	Reserved	0x1FFF_FFFF | 0x0000_7600
	29.5 KB on-chip Flash (NDA102)	0x0000_75FF |
0 GB		0x0000_0000

System Control

System Control	0xE000_ED00	SCS_BA
External Interrupt Control	0xE000_E100	SCS_BA
System Timer Control	0xE000_E010	SCS_BA

AHB peripherals

FMC	0x5000_C000	FMC_BA
GPIO Control	0x5000_4000	GP_BA
Interrupt Multiplexer Control	0x5000_0300	INT_BA
Clock Control	0x5000_0200	CLK_BA
System Global Control	0x5000_0000	SYS_BA

APB peripherals

ECAP Control	0x401B_0000	ECAP_BA
USCI1 Control	0x4017_0000	USCI1_BA
BPWM Control	0x4014_0000	BPWM_BA
PGA Control	0x400F_0000	PGA_BA
ADC Control	0x400E_0000	ADC_BA
ACMP 0/1 Control	0x400D_0000	ACMP_BA
USCI0 Control	0x4007_0000	USCI0_BA
EPWM Control	0x4004_0000	EPWM_BA
Timer0/Timer1 Control	0x4001_0000	TMR01_BA
WDT Control	0x4000_4000	WDT_BA

Table 6.2-5 Memory Mapping Table

6.2.6 Register Protection

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 SYS_REGLCTL 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 REGLCTL (SYS_REGLCTL [0]), “1” is protection disable, “0” is protection enable. Then user can update the target protected register value and then write any data to SYS_REGLCTL to enable register protection.

The protected registers are listed in Table 6.2-6.

Register	Bit	Description
SYS_IPRST0	[1] CPURST	Processor Core One-shot Reset (Write Protect)
	[0] CHIPRST	Chip One-shot Reset (Write Protect)
SYS_BODCTL	[15] LVREN	Low Voltage Reset Enable Control (Write Protect)
	[6] BODLPM	Brown-out Detector Low Power Mode (Write Protect)
	[4] BODRSTEN	Brown-out Reset Enable Control (Write Protect)
	[3:1] BODVL	Brown-out Detector Threshold Voltage Selection (Write Protect)
	[0] BODEN	Brown-out Detector Enable Control (Write Protect)
SYS_PORCTL	[15:0] POROFF	Power-on Reset Enable Control (Write Protect)
INT_NMICTL	[8] NMISELEN	NMI Interrupt Enable Control (Write Protected)
CLK_PWRCTL	[11:10] HXTGAIN	HXT Gain Control (Write Protect)
	[7] PDEN	System Power-down Enable Control (Write Protect)
	[5] PDWKIEN	Power-down Mode Wake-up Interrupt Enable Control (Write Protect)
	[4] PDWKDLY	Wake-up Delay Counter Enable Control (Write Protect)
	[3] LIRCEN	LIRC Enable Control (Write Protect)
	[2] HIRCEN	HIRC Enable Control (Write Protect)
	[1:0] XTLEN	XTL Enable Control (Write Protect)
CLK_APBCLK	[0] WDTCKEN	Watchdog Timer Clock Enable Control (Write Protect)
CLK_CLKSEL0	[4:3] STCLKSEL	Cortex®-M0 SysTick Clock Source Selection (Write Protect)
	[1:0] HCLKSEL	HCLK Clock Source Selection (Write Protect)
CLK_CLKSEL1	[1:0] WDTSEL	Watchdog Timer Clock Source Selection (Write Protect)
FMC_ISPCTL	[6] ISPFF	ISP Fail Flag (Write Protect)
	[5] LDUEN	LDROM Update Enable Control (Write Protect)
	[4] CFGUEN	CONFIG Update Enable Control (Write Protect)

	[3] APUEN	APROM Update Enable Control (Write Protect)
	[2] SPUEN	SPROM Update Enable Control (Write Protect)
	[1] BS	Boot Select (Write Protect)
	[0] ISPEN	ISP Enable Control (Write Protect)
FMC_ISPTRG	[0] ISPGO	ISP Start Trigger (Write Protect)
FMC_ISPSTS	[6] ISPFF	ISP Fail Flag (Write Protect)
TIMER0_CTL	[31] ICEDEBUG	ICE Debug Mode Acknowledge Disable Control (Write Protect)
TIMER1_CTL	[31] ICEDEBUG	ICE Debug Mode Acknowledge Disable Control (Write Protect)
WDT_CTL	[31] ICEDEBUG	ICE Debug Mode Acknowledge Disable Control (Write Protect)
	[7] WDTEN	Watchdog Timer Enable Control (Write Protect)
	[6] INTEN	Watchdog Timer Time-out Interrupt Enable Control (Write Protect)
	[4] WKEN	Watchdog Timer Time-out Wake-up Function Control (Write Protect)
	[1] RSTEN	Watchdog Timer Time-out Reset Enable Control (Write Protect)
	[0] RSTCNT	Reset Watchdog Timer Up Counter (Write Protect)

Table 6.2-6 Protected Registers

6.2.7 Memory Organization

6.2.7.1 Overview

The NuMicro® NDA102 series provides 4G-byte addressing space. The addressing space assigned to each on-chip controllers is shown in Figure 6.2-8. The detailed register definition, addressing space, and programming details will be described in the following sections for each on-chip peripheral. The NDA102 series only supports little-endian data format.

Figure 6.2-8 NuMicro® NDA102 Flash, Security and Configuration Map

6.2.7.2 System Memory Map

The NDA102 series provides 4G-byte addressing space. The memory locations assigned to each on-chip controllers are shown in Table 6.2-7. The detailed register definition, memory space, and programming will be described in the following sections for each on-chip peripheral. The NDA102 series only supports little-endian data format.

The memory locations assigned to each on-chip controllers are shown in Table 6.2-7.

Address Space	Token	Controllers
Flash and SRAM Memory Space		
0x0000_0000 – 0x0000_75FF	FLASH_BA	FLASH Memory Space (29.5KB)
0x0010_0000 – 0x0010_07FF	LD_BA	Loader Memory Space (2 KB)
0x0020_0000 – 0x0020_01FF	SP0_BA	Security Program Memory 0 Space (0.5 KB)
0x0024_0000 – 0x0024_01FF	SP1_BA	Security Program Memory 1 Space (0.5 KB)
0x0028_0000 – 0x0028_01FF	SP2_BA	Security Program Memory 2 Space (0.5 KB)
0x2000_0000 – 0x2000_0FFF	SRAM_BA	SRAM Memory Space (4 KB)
AHB Modules Space (0x5000_0000 – 0x501F_FFFF)		
0x5000_0000 – 0x5000_01FF	SYS_BA	System 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	HDIV_BA	Hardware Divider Control Register
APB Controllers Space (0x4000_0000 ~ 0x401F_FFFF)		
0x4000_4000 – 0x4000_7FFF	WDT_BA	Watchdog Timer Control Registers
0x4001_0000 – 0x4001_3FFF	TMR01_BA	Timer0/Timer1 Control Registers
0x4004_0000 – 0x4004_3FFF	EPWM_BA	Enhance PWM Control Registers
0x4007_0000 – 0x4007_3FFF	USCI0_BA	USCI0 Control Registers
0x400D_0000 – 0x400D_3FFF	ACMP_BA	Analog Comparator 0/1 Control Registers
0x400E_0000 – 0x400E_3FFF	ADC_BA	ADC Control Registers
0x400F_0000 – 0x400F_3FFF	PGA_BA	Programmable Gain Amplifier Control Register
0x4014_0000 – 0x4014_3FFF	BPWM_BA	Basic PWM Control Registers
0x4017_0000 – 0x4017_3FFF	USCI1_BA	USCI1 Control Registers
0x401B_0000 – 0x401B_3FFF	ECAP_BA	Enhanced Input Capture Timer Register
System Controllers Space (0xE000_E000 ~ 0xE000_EFFF)		
0xE000_E010 – 0xE000_E0FF	SCS_BA	System Timer Control Registers
0xE000_E100 – 0xE000_ECFF	SCS_BA	External Interrupt Controller Control Registers
0xE000_ED00 – 0xE000_ED8F	SCS_BA	System Control Registers

Table 6.2-7 Address Space Assignments for On-Chip Modules

6.2.7.3 SRAM Memory Organization

The NDA102 supports embedded SRAM with total 4 Kbytes size.

- Supports total 4 Kbytes SRAM
- Supports byte / half word / word write
- Supports oversize response error

Figure 6.2-9 SRAM Block Diagram

6.2.8 Register Map

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

Register	Offset	R/W	Description	Reset Value
SYS Base Address: **SYS_BA = 0x5000_0000**				
SYS_PDID	SYS_BA+0x00	R	Part Device Identification Number Register	0xXXXX_XXXX [1]
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status Register	0x0000_00XX
SYS_IPRST0	SYS_BA+0x08	R/W	Peripheral Reset Control Register 0	0x0000_0000
SYS_IPRST1	SYS_BA+0x0C	R/W	Peripheral Reset Control Register 1	0x0000_0000
SYS_WAIT	SYS_BA+0x10	R/W	HCLK Wait State Cycle Control Register	0x0000_0001
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-Out Detector Control Register	0x0000_80XX
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source Control Register	0x0000_0000
SYS_PORCTL	SYS_BA+0x24	R/W	Power-On-Reset Controller Register	0x0000_00XX
SYS_GPA_MFP	SYS_BA+0x30	R/W	GPIOA Multiple Function Control Register	0x0000_0000
SYS_GPB_MFP	SYS_BA+0x34	R/W	GPIOB Multiple Function Control Register	0x0000_0000
SYS_GPC_MFP	SYS_BA+0x38	R/W	GPIOC Multiple Function Control Register	0x0000_0000
SYS_GPD_MFP	SYS_BA+0x3C	R/W	GPIOD Multiple Function Control Register	0x0000_0111
SYS_IRCTCTL	SYS_BA+0x80	R/W	HIRC Trim Control Register	0x0000_0030
SYS_IRCTIEN	SYS_BA+0x84	R/W	HIRC Trim Interrupt Enable Register	0x0000_0000
SYS_IRCTISTS	SYS_BA+0x88	R/W	HIRC Trim Interrupt Status Register	0x0000_0000
SYS_REGLCTL	SYS_BA+0x100	R/W	Register Write-Protection Control Register	0x0000_0000
SYS_TSOFFSET	SYS_BA+0x114	R	Temperature sensor offset Register	0x0XXX_0XXX

6.2.9 Register Description

Part Device Identification Number Register (SYS_PDID)

Register	Offset	R/W	Description	Reset Value
SYS_PDID	SYS_BA+0x00	R	Part Device Identification Number Register	0xXXXX_XXXX [1]

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

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

Bits	Description	
[31:0]	**PDID**	**Part Device Identification Number (Read Only)** This register reflects device part number code. Software can read this register to identify which device is used.

NuMicro® NDA102 Series	Part Device Identification Number
NDA102EC1	0x00B05740

Table 6.2-1 Part Device Identification Number

System Reset Status Register (SYS_RSTSTS)

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

Register	Offset	R/W	Description	Reset Value
SYS_RSTSTS	SYS_BA+0x04	R/W	System Reset Status 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**
CPURF	**Reserved**	**SYSRF**	**BODRF**	**LVRF**	**WDTRF**	**PINRF**	**PORF**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:8]	**Reserved**	Reserved.
[7]	**CPURF**	**CPU Reset Flag** The CPU reset flag is set by hardware if software writes CPURST (SYS_IPRST0[1]) 1 to reset Cortex®-M0 Core and Flash Memory Controller (FMC). 0 = No reset from CPU. 1 = The Cortex®-M0 Core and FMC are reset by software setting CPURST to 1. **Note:** Write 1 to clear this bit to 0.
[6]	**Reserved**	Reserved.
[5]	**SYSRF**	**System Reset Flag** The system reset flag is set by the "Reset Signal" from the Cortex®-M0 Core to indicate the previous reset source. 0 = No reset from Cortex®-M0. 1 = The Cortex®-M0 had issued the reset signal to reset the system by writing 1 to the bit SYSRESETREQ(AIRCR[2], Application Interrupt and Reset Control Register, address = 0xE000ED0C) in system control registers of Cortex®-M0 core. **Note:** Write 1 to clear this bit to 0.
[4]	**BODRF**	**BOD Reset Flag** The BOD reset flag is set by the "Reset Signal" from the Brown-Out Detector to indicate the previous reset source. 0 = No reset from BOD. 1 = The BOD had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.

Bits	Description	
[3]	**LVRF**	**LVR Reset Flag** The LVR reset flag is set by the "Reset Signal" from the Low Voltage Reset Controller to indicate the previous reset source. 0 = No reset from LVR. 1 = LVR controller had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[2]	**WDTRF**	**WDT Reset Flag** The WDT reset flag is set by the "Reset Signal" from the Watchdog Timer or Window Watchdog Timer to indicate the previous reset source. 0 = No reset from watchdog timer or window watchdog timer. 1 = The watchdog timer or window watchdog timer had issued the reset signal to reset the system. **Note1:** Write 1 to clear this bit to 0. **Note2:** Watchdog Timer register RSTF(WDT_CTL[2]) bit is set if the system has been reset by WDT time-out reset. Window Watchdog Timer register WWDTRF(WWDT_STATUS[1]) bit is set if the system has been reset by WWDT time-out reset.
[1]	**PINRF**	**NRESET Pin Reset Flag** The nRESET pin reset flag is set by the "Reset Signal" from the nRESET Pin to indicate the previous reset source. 0 = No reset from nRESET pin. 1 = Pin nRESET had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.
[0]	**PORF**	**POR Reset Flag** The POR reset flag is set by the "Reset Signal" from the Power-on Reset (POR) Controller or bit CHIPRST (SYS_IPRST0[0]) to indicate the previous reset source. 0 = No reset from POR or CHIPRST. 1 = Power-on Reset (POR) or CHIPRST had issued the reset signal to reset the system. **Note:** Write 1 to clear this bit to 0.

Peripheral Reset Control Register 0 (SYS_IPRST0)

Register	Offset	R/W	Description	Reset Value
SYS_IPRST0	SYS_BA+0x08	R/W	Peripheral Reset Control Register 0	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						**CPURST**	**CHIPRST**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**CPURST**	**Processor Core One-shot Reset (Write Protect)** Setting this bit will only reset the processor core and Flash Memory Controller(FMC), and this bit will automatically return to 0 after the 2 clock cycles. 0 = Processor core normal operation. 1 = Processor core one-shot reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**CHIPRST**	**Chip One-shot Reset (Write Protect)** Setting this bit will reset the whole chip, including Processor core and all peripherals, and this bit will automatically return to 0 after the 2 clock cycles. The CHIPRST is same as the POR reset; all the chip controllers are reset and the chip setting from Flash is also reloaded. About the difference between CHIPRST and SYSRESETREQ(AIRCR[2]), please refer to section 6.2.2 0 = Chip normal operation. 1 = Chip one-shot reset. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Peripheral Reset Control Register 1 (SYS_IPRST1)

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

Register	Offset	R/W	Description	Reset Value
SYS_IPRST1	SYS_BA+0x0C	R/W	Peripheral Reset Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**ACMPRST**	**Reserved**	**ADCRST**	**Reserved**		**USCI1RST**	**USCI0RST**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**EPWMRST**		**Reserved**		**BPWMRST**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**PGARST**	**Reserved**		**CAPRST**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**TMR1RST**	**TMR0RST**	**GPIORST**	**Reserved**

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**ACMPRST**	**ACMP Controller Reset** 0 = ACMP controller normal operation. 1 = ACMP controller reset.
[29]	**Reserved**	Reserved.
[28]	**ADCRST**	**ADC Controller Reset** 0 = ADC controller normal operation. 1 = ADC controller reset.
[27:26]	**Reserved**	Reserved.
[25]	**USCI1RST**	**USCI1 Controller Reset** 0 = USCI1 controller normal operation. 1 = USCI1 controller reset.
[24]	**USCI0RST**	**USCI0 Controller Reset** 0 = USCI0 controller normal operation. 1 = USCI0 controller reset.
[23:21]	**Reserved**	Reserved.
[20]	**EPWMRST**	**Enhanced PWM Controller Reset** 0 = EPWM controller normal operation. 1 = EPWM controller reset.
[19:17]	**Reserved**	Reserved.

[16]	**BPWMRST**	**Basic PWM Controller Reset** 0 = BPWM controller normal operation. 1 = BPWM controller reset.
[15:13]	**Reserved**	Reserved.
[12]	**PGARST**	**PGA Controller Reset** 0 = PGA controller normal operation. 1 = PGA controller reset.
[11:9]	**Reserved**	Reserved.
[8]	**CAPRST**	**ECAP Controller Reset** 0 = ECAP controller normal operation. 1 = ECAP controller reset.
[7:4]	**Reserved**	Reserved.
[3]	**TMR1RST**	**Timer1 Controller Reset** 0 = Timer1 controller normal operation. 1 = Timer1 controller reset.
[2]	**TMR0RST**	**Timer0 Controller Reset** 0 = Timer0 controller normal operation. 1 = Timer0 controller reset.
[1]	**GPIORST**	**GPIO Controller Reset** 0 = GPIO controller normal operation. 1 = GPIO controller reset.
[0]	**Reserved**	Reserved.

HCLK Wait State Cycle Control Register (SYS_WAIT)

Register	Offset	R/W	Description	Reset Value
SYS_WAIT	SYS_BA+0x10	R/W	HCLK Wait State Cycle 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**
Reserved							**HCLKWS**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**HCLKWS**	**HCLK Wait State Cycle Control Bit** This bit is used to enable/disable HCLK wait state when access Flash. 0 = No wait state. 1 = One wait state inserted when CPU access Flash. **Note:** When HCLK frequency is faster than 48MHz, insert one wait state is necessary.

Brown-out Detector Control Register (SYS_BODCTL)

Partial of the SYS_BODCTL control registers values are initiated by the Flash configuration and partial bits are write-protected bit.

Register	Offset	R/W	Description	Reset Value
SYS_BODCTL	SYS_BA+0x18	R/W	Brown-Out Detector Control Register	0x0000_80XX

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**
LVREN	**Reserved**						
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BODOUT	**BODLPM**	**BODIF**	**BODRSTEN**	**BODVL**			**BODEN**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15]	**LVREN**	**Low Voltage Reset Enable Bit (Write Protect)** The LVR function resets the chip when the input power voltage is lower than LVR circuit setting. LVR function is enabled by default. 0 = Low Voltage Reset function Disabled. 1 = Low Voltage Reset function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[14:8]	**Reserved**	Reserved.
[7]	**BODOUT**	**Brown-out Detector Output Status** 0 = Brown-out Detector output status is 0. It means the detected voltage is higher than BODVL setting or BODEN is 0. 1 = Brown-out Detector output status is 1. It means the detected voltage is lower than BODVL setting. If the BODEN is 0, BOD function is disabled. This bit always responds 0000.
[6]	**BODLPM**	**Brown-out Detector Low Power Mode (Write Protect)** 0 = BOD operate in normal mode (default). 1 = BOD Low Power mode Enabled. **Note1:** The BOD consumes about 100uA in normal mode, the low power mode can reduce the current to about 1/10 but slow the BOD response. **Note2:** This bit is write protected. Refer to the SYS_REGLCTL register.

Bits	Description	
[5]	**BODIF**	**Brown-out Detector Interrupt Flag** 0 = Brown-out Detector does not detect any voltage draft at V_{DD} down through or up through the voltage of BODVL setting. 1 = When Brown-out Detector detects the V_{DD} is dropped down through the voltage of BODVL setting or the V_{DD} is raised up through the voltage of BODVL setting, this bit is set to 1 and the brown-out interrupt is requested if brown-out interrupt is enabled. **Note:** Write 1 to clear this bit to 0.
[4]	**BODRSTEN**	**Brown-out Reset Enable Bit (Write Protect)** The default value is set by Flash controller user configuration register CBORST(CONFIG0[12]) bit . 0 = Brown-out "INTERRUPT" function Enabled. 1 = Brown-out "RESET" function Enabled. **Note1:** While the Brown-out Detector function is enabled (BODEN high) and BOD reset function is enabled (BODRSTEN high), BOD will assert a signal to reset chip when the detected voltage is lower than the threshold (BODOUT high). While the BOD function is enabled (BODEN high) and BOD interrupt function is enabled (BODRSTEN low), BOD will assert an interrupt if BODOUT is high. BOD interrupt will keep till to the BODEN set to 0. BOD interrupt can be blocked by disabling the NVIC BOD interrupt or disabling BOD function (set BODEN low). BOD will wake CPU up when BODOUT is high in power-down mode. **Note2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3:1]	**BODVL**	**Brown-out Detector Threshold Voltage Selection (Write Protect)** The default value is set by Flash controller user configuration register CBOV (CONFIG0 [15:13]). 000 = Brown-Out Detector threshold voltage is 2.0V. 001 = Brown-Out Detector threshold voltage is 2.2V. 010 = Brown-Out Detector threshold voltage is 2.4V. 011 = Brown-Out Detector threshold voltage is 2.7V. 100 = Brown-Out Detector threshold voltage is 3.0V. 101 = Brown-Out Detector threshold voltage is 3.7V. 110 = Brown-Out Detector threshold voltage is 4.0V. 111 = Brown-Out Detector threshold voltage is 4.3V. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[0]	**BODEN**	**Brown-out Detector Enable Bit (Write Protect)** The default value is set by Flash controller user configuration register CBODEN (CONFIG0 [12]). 0 = Brown-out Detector function Disabled. 1 = Brown-out Detector function Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Internal Voltage Source Control Register (SYS_IVSCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IVSCTL	SYS_BA+0x1C	R/W	Internal Voltage Source 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							**VTEMPEN**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**VTEMPEN**	**Temperature Sensor Enable Bit** This bit is used to enable/disable temperature sensor function. 0 = Temperature sensor function Disabled (default). 1 = Temperature sensor function Enabled. **Note:** After this bit is set to 1, the value of temperature sensor output can be obtained from A/D conversion result. Please refer to ADC function chapter for details.

Power-on Reset Controller Register (SYS_PORCTL)

Register	Offset	R/W	Description	Reset Value
SYS_PORCTL	SYS_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**
POROFF							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
POROFF							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**POROFF**	**Power-on Reset Enable Bit (Write Protect)** 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. The POR function will be active again when this field is set to another value or chip is reset by other reset source, including: nRESET, Watchdog, LVR reset, BOD reset, ICE reset command and the software-chip reset function. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

GPIOA Multiple Function Control Register (SYS_GPA_MFP)

Please refer to GPIO Multi-function Pin Summary

Register	Offset	R/W	Description	Reset Value
SYS_GPA_MFP	SYS_BA+0x30	R/W	GPIOA Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PA5MFP				**PA4MFP**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PA3MFP				**PA2MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PA1MFP				**PA0MFP**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:20]	**PA5MFP**	**PA.5 Multi-function Pin Selection**
[19:16]	**PA4MFP**	**PA.4 Multi-function Pin Selection**
[15:12]	**PA3MFP**	**PA.3 Multi-function Pin Selection**
[11:8]	**PA2MFP**	**PA.2 Multi-function Pin Selection**
[7:4]	**PA1MFP**	**PA.1 Multi-function Pin Selection**
[3:0]	**PA0MFP**	**PA.0 Multi-function Pin Selection**

GPIOB Multiple Function Control Register (SYS_GPB_MFP)

Please refer to GPIO Multi-function Pin Summary

Register	Offset	R/W	Description	Reset Value
SYS_GPB_MFP	SYS_BA+0x34	R/W	GPIOB Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**PB4MFP**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PB3MFP				**PB2MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PB1MFP				**PB0MFP**			

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**PB4MFP**	**PB.4 Multi-function Pin Selection**
[15:12]	**PB3MFP**	**PB.3 Multi-function Pin Selection**
[11:8]	**PB2MFP**	**PB.2 Multi-function Pin Selection**
[7:4]	**PB1MFP**	**PB.1 Multi-function Pin Selection**
[3:0]	**PB0MFP**	**PB.0 Multi-function Pin Selection**

GPIOC Multiple Function Control Register (SYS_GPC_MFP)

Please refer to GPIO Multi-function Pin Summary

Register	Offset	R/W	Description	Reset Value
SYS_GPC_MFP	SYS_BA+0x38	R/W	GPIOC Multiple Function Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**PC4MFP**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PC3MFP				**PC2MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PC1MFP				**PC0MFP**			

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19:16]	**PC4MFP**	**PC.4 Multi-function Pin Selection**
[15:12]	**PC3MFP**	**PC.3 Multi-function Pin Selection**
[11:8]	**PC2MFP**	**PC.2 Multi-function Pin Selection**
[7:4]	**PC1MFP**	**PC.1 Multi-function Pin Selection**
[3:0]	**PC0MFP**	**PC.0 Multi-function Pin Selection**

GPIOD Multiple Function Control Register (SYS_GPD_MFP)

Please refer to GPIO Multi-function Pin Summary

Register	Offset	R/W	Description	Reset Value
SYS_GPD_MFP	SYS_BA+0x3C	R/W	GPIOD Multiple Function Control Register	0x0000_0111

31	30	29	28	27	26	25	24
Reserved				**PD6MFP**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
PD5MFP				**PD4MFP**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
PD3MFP				**PD2MFP**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PD1MFP				**Reserved**			

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:24]	**PD6MFP**	**PD.6 Multi-function Pin Selection**
[23:20]	**PD5MFP**	**PD.5 Multi-function Pin Selection**
[19:16]	**PD4MFP**	**PD.4 Multi-function Pin Selection**
[15:12]	**PD3MFP**	**PD.3 Multi-function Pin Selection**
[11:8]	**PD2MFP**	**PD.2 Multi-function Pin Selection**
[7:4]	**PD1MFP**	**PD.1 Multi-function Pin Selection**
[3:0]	**Reserved**	Reserved.

HIRC Trim Control Register (SYS_IRCTCTL)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTCTL	SYS_BA+0x80	R/W	HIRC Trim Control Register	0x0000_0030

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**
RETRYCNT		**LOOPSEL**		**Reserved**			**FREQSEL**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:6]	**RETRYCNT**	**Trim Value Update Limitation Count** This field defines that how many times the auto trim circuit will try to update the HIRC trim value before the frequency of HIRC locked. Once the HIRC locked, the internal trim value update counter will be reset. If the trim value update counter reached this limitation value and frequency of HIRC still doesn't lock, the auto trim operation will be disabled and FREQSEL will be cleared to 00. 00 = Trim retry count limitation is 64 loops. 01 = Trim retry count limitation is 128 loops. 10 = Trim retry count limitation is 256 loops. 11 = Trim retry count limitation is 512 loops.
[5:4]	**LOOPSEL**	**Trim Calculation Loop Selection** This field defines that trim value calculation is based on how many 32.768 kHz clock. 00 = Trim value calculation is based on average difference in 4 32.768 kHz clock. 01 = Trim value calculation is based on average difference in 8 32.768 kHz clock. 10 = Trim value calculation is based on average difference in 16 32.768 kHz clock. 11 = Trim value calculation is based on average difference in 32 32.768 kHz clock. **Note:** For example, if LOOPSEL is set as 00, auto trim circuit will calculate trim value based on the average frequency difference in 4 32.768 kHz clock.
[3:1]	**Reserved**	Reserved.
[0]	**FREQSEL**	**Trim Frequency Selection** This field indicates the target frequency of 48 MHz internal high speed RC oscillator (HIRC) auto trim. 0 = HIRC auto trim function Disabled. 1 = HIRC auto trim function and trim HIRC to 48 MHz Enabled.

HIRC Trim Interrupt Enable Register (SYS_IRCTIEN)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTIEN	SYS_BA+0x84	R/W	HIRC Trim 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**CLKEIEN**	**TFAILIEN**	**Reserved**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKEIEN**	**Clock Error Interrupt Enable Bit** This bit controls if CPU would get an interrupt while clock is inaccuracy during auto trim operation. If this bit is set to1, and CLKERRIF(SYS_IRCTSTS[2]) is set during auto trim operation, an interrupt will be triggered to notify the clock frequency is inaccuracy. 0 = CLKERRIF(SYS_IRCTSTS[2]) status to trigger an interrupt to CPU Disabled. 1 = CLKERRIF(SYS_IRCTSTS[2]) status to trigger an interrupt to CPU Enabled.
[1]	**TFAILIEN**	**Trim Failure Interrupt Enable Bit** This bit controls if an interrupt will be triggered while HIRC trim value update limitation count reached and HIRC frequency still not locked on target frequency set by FREQSEL(SYS_IRCTCTL[0]). If this bit is high and TFAILIF(SYS_IRCTSTS[1]) is set during auto trim operation, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. 0 = TFAILIF(SYS_IRCTSTS[1]) status to trigger an interrupt to CPU Disabled. 1 = TFAILIF(SYS_IRCTSTS[1]) status to trigger an interrupt to CPU Enabled.
[0]	**Reserved**	Reserved.

HIRC Trim Interrupt Status Register (SYS_IRCTISTS)

Register	Offset	R/W	Description	Reset Value
SYS_IRCTISTS	SYS_BA+0x88	R/W	HIRC Trim 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					**CLKERRIF**	**TFAILIF**	**FREQLOCK**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**CLKERRIF**	**Clock Error Interrupt Status** When the frequency of 32.768 kHz external low speed crystal oscillator (LXT) or 48 MHz internal high speed RC oscillator (HIRC) is shift larger to unreasonable value, this bit will be set and to be an indicate that clock frequency is inaccuracy If this bit is set and CLKEIEN(SYS_IRCTIEN[2]) is high, an interrupt will be triggered to notify the clock frequency is inaccuracy. Write 1 to clear this to 0. 0 = Clock frequency is accuracy. 1 = Clock frequency is inaccuracy.
[1]	**TFAILIF**	**Trim Failure Interrupt Status** This bit indicates that HIRC trim value update limitation count reached and the HIRC clock frequency still doesn't be locked. Once this bit is set, the auto trim operation stopped and FREQSEL(SYS_iRCTCTL[1:0]) will be cleared to 00 by hardware automatically. If this bit is set and TFAILIEN(SYS_IRCTIEN[1]) is high, an interrupt will be triggered to notify that HIRC trim value update limitation count was reached. Write 1 to clear this to 0. 0 = Trim value update limitation count does not reach. 1 = Trim value update limitation count reached and HIRC frequency still not locked.
[0]	**FREQLOCK**	**HIRC Frequency Lock Status** This bit indicates the HIRC frequency is locked. This is a status bit and doesn't trigger any interrupt. 0 = The internal high-speed oscillator frequency doesn't lock at 48 MHz yet. 1 = The internal high-speed oscillator frequency locked at 48 MHz.

Register Lock Control Register (SYS_REGLCTL)

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 SYS_REGLCTL 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, and 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.

This register is writen to disable/enable register protection and read for the REGLCTL status.

Register	Offset	R/W	Description	Reset Value
SYS_REGLCTL	SYS_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**
REGLCTL							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7:1]	**REGPROTDIS**	**Register Write-protection Code (Write Only)** Some registers have write-protection function. Writing these registers have to disable the protected function by writing the sequence value 0x59, 0x16, 0x88 to this field. After this sequence is completed, the SYS_REGLCTL bit will be set to 1 and write-protection registers can be normal write.
[0]	**REGLCTL**	**Register Lock Control Disable Index (Read Only)** 0 = Write-protection Enabled for writing protected registers. Any write to the protected register is ignored. 1 = Write-protection Disabled for writing protected registers. Please refer to section 6.2.6 for detail descriptions. **Note:** The bits which are write-protected will be noted as” **(Write Protect)**” beside the description.

Temperature Sensor Offset Register (SYS_TSOFFSET)

Register	Offset	R/W	Description	Reset Value
SYS_TSOFFSET	SYS_BA+0x114	R	Temperature sensor offset Register	0x0XXX_0XXX

31	30	29	28	27	26	25	24
Reserved			**VTEMP1**				
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
VTEMP1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**VTEMP0**				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VTEMP0							

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**VTEMP1**	**Temperature Sensor Offset Value** This field reflects temperature sensor output voltage offset at 125°C.
[15:12]	**Reserved**	Reserved.
[11:0]	**VTEMP0**	**Temperature Sensor Offset Value** This field reflects temperature sensor output voltage offset at 25°C.

6.2.10 System Timer (SysTick)

The Cortex®-M0 includes an integrated system timer, SysTick, which provides a simple, 24-bit cleared-on-write, decrementing, wrap-on-zero counter with a flexible control mechanism. The counter can be used in several different ways, for example:

An RTOS tick timer fires at a programmable rate (for example 100Hz) and invokes a SysTick routine.

A high-speed alarm timer uses Core clock.

A variable rate alarm or signal timer – the duration range is dependent on the reference clock used and the dynamic range of the counter.

A simple counter can be used by software to measure task completion time.

An internal Clock Source control based on missing/meeting durations. The COUNTFLAG bit-field in the control and status register can be used to determine if an action completed within a set duration, as part of a dynamic clock management control loop.

When enabled, the timer 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 edge, and then decrement on subsequent clocks. When the counter transitions to 0, the COUNTFLAG status bit is set. The COUNTFLAG bit clears on read.

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 zero, the timer will be maintained with a current value of zero 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”.

6.2.10.1 System Timer Control Register Map

R: read only**, W:** write only**, R/W:** both read and write**, W&C:** write 1 to clear

Register	Offset	R/W	Description	Reset Value
SCS Base Address: **SCS_BA = 0xE000_E000**				
SYST_CTL	SCS_BA+0x10	R/W	SysTick Control and Status	0x0000_0004
SYST_RVR	SCS_BA+0x14	R/W	SysTick Reload Value Register	0xXXXX_XXXX
SYST_CVR	SCS_BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

6.2.10.2 System Timer Control Register Description

SysTick Control and Status （SYST_CTL）

Register	Offset	R/W	Description	Reset Value
SYST_CTL	SCS_BA+0x10	R/W	SysTick Control and Status	0x0000_0004

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**	**System Tick Counter Flag** Return 1 If Timer Counted to 0 Since Last Time this Register Was Read 0 = COUNTFLAG is cleared on read or by a write to the Current Value register. 1 = COUNTFLAG is set by a count transition from 1 to 0.
[15:3]	**Reserved**	Reserved.
[2]	**CLKSRC**	**System Tick Clock Source Select Bit** 0 = Clock source is optional, refer to STCLKSEL. 1 = Core clock used for SysTick timer.
[1]	**TICKINT**	**System Tick Interrupt Enable Bit** 0 = Counting down to 0 will not cause the SysTick exception to be pended. User can use COUNTFLAG to determine if a count to zero has occurred. 1 = Counting down to 0 will cause the SysTick exception to be pended. Clearing the SysTick Current Value register by a register write in software will not cause SysTick to be pended.
[0]	**ENABLE**	**System Tick Counter Enable Bit** 0 = System Tick counter Disabled. 1 = System Tick counter will operate in a multi-shot manner.

SysTick Reload Value Register (SYST_RVR）

Register	Offset	R/W	Description	Reset Value
SYST_RVR	SCS_BA+0x14	R/W	SysTick Reload Value Register	0xXXXX_XXXX

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**RELOAD**	**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	SCS_BA+0x18	R/W	SysTick Current Value Register	0xXXXX_XXXX

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CURRENT**	**System Tick Current Value** 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.

6.2.11 Nested Vectored Interrupt Control (NVIC)

6.2.11.1 Overview

The Cortex®-M0 CPU 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.

6.2.11.2 Features

- Nested and Vectored interrupt support
- Automatic processor state saving and restoration
- Dynamic priority change
- 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".

The processor automatically stacks its state on exception entry and unstacks this state on exception exit, with no instruction overhead. This provides low latency exception handling.

6.2.11.3 Exception Model and System Interrupt Map

Table 6.2-8 lists the exception model supported by the NDA102 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 the 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
SysTick	15	Configurable
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	Configurable

Table 6.2-8 Exception Model

Vector Number	Interrupt Number (Bit In Interrupt Registers)	Interrupt Name	Interrupt Description
0 ~ 15	-	-	System exceptions
16	0	**BOD_OUT**	Brown-Out low voltage detected interrupt
17	1	**WDTPINT**	Watchdog Timer interrupt
18	2	**USCI0**	USCI0 interrupt
19	3	**USCI1**	USCI1 interrupt
20	4	**GP_INT**	External interrupt from GPA ~ GPD pins
21	5	**EPWM_INT**	EPWM interrupt
22	6	**BRAKE0_INT**	EPWM brake interrupt from PWM0 or PWM_BRAKE pin
23	7	**BRAKE1_INT**	EPWM brake interrupt from PWM1
24	8	**BPWM0_INT**	BPWM0 interrupt
25	9	**BPWM1_INT**	BPWM1 interrupt
26	10	Reserved	Reserved
27	11	Reserved	Reserved
28	12	Reserved	Reserved
29	13	Reserved	Reserved
30	14	Reserved	Reserved
31	15	**ECAP_INT**	Enhanced Input Capture interrupt
32	16	**CCAP_INT**	Continues Input Capture interrupt
33	17	Reserved	Reserved

34	18	Reserved	Reserved
35	19	Reserved	Reserved
36	20	Reserved	Reserved
37	21	**HIRCTRIM_INT**	HIRC TRIM interrupt
38	22	**TMR0_INT**	Timer 0 interrupt
39	23	**TMR1_INT**	Timer 1 interrupt
40	24	Reserved	Reserved
41	25	Reserved	Reserved
42	26	**ACMP_INT**	Analog Comparator 0 or Comparator 1 interrupt
43	27	Reserved	Reserved
44	28	**PWRWU_INT**	Chip wake-up from Power-down state interrupt
45	29	**ADC0_INT**	ADC0 interrupt
46	30	**ADC1_INT**	ADC1 interrupt
47	31	**ADCWCMP_INT**	ADC Window Compare interrupt

Table 6.2-9 System Interrupt Map Vector Table

6.2.11.4 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 based 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 the exception handler entry as illustrated in previous section.

Vector Table Word Offset (Bytes)	Description
0x00	Initial Stack Pointer Value
Exception Number * 0x04	Exception Entry Pointer using that Exception Number

Table 6.2-10 Vector Table Format

6.2.11.5 Operation Description

The 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 activate. 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 pended/un-pended 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.

6.2.11.6 NVIC Control Registers Map

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

Register	Offset	R/W	Description	Reset Value
SCS Base Address: **SCS_BA = 0xE000_E000**				
NVIC_ISER	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000
NVIC_IPR0	SCS_BA+0x400	R/W	IRQ0 ~ IRQ3 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR1	SCS_BA+0x404	R/W	IRQ4 ~ IRQ7 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR2	SCS_BA+0x408	R/W	IRQ8 ~ IRQ11 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR3	SCS_BA+0x40C	R/W	IRQ12 ~ IRQ15 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR4	SCS_BA+0x410	R/W	IRQ16 ~ IRQ19 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR5	SCS_BA+0x414	R/W	IRQ20 ~ IRQ23 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR6	SCS_BA+0x418	R/W	IRQ24 ~ IRQ27 Interrupt Priority Control Register	0x0000_0000
NVIC_IPR7	SCS_BA+0x41C	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	SCS_BA+0x100	R/W	IRQ0 ~ IRQ31 Set-Enable Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**SETENA**	**Interrupt Enable Register** Enable one or more interrupts. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Write operation: 0 = No effect. 1 = Write 1 to enable associated interrupt. Read operation: 0 = Associated interrupt status Disabled. 1 = Associated interrupt status Enabled. Read value indicates the current enable status.

IRQ0 ~ IRQ31 Clear-enable Control Register (NVIC_ICER)

Register	Offset	R/W	Description	Reset Value
NVIC_ICER	SCS_BA+0x180	R/W	IRQ0 ~ IRQ31 Clear-Enable Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CLRENA**	**Interrupt Disable Register** Disable one or more interrupts. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Write operation: 0 = No effect. 1 = Write 1 to disable associated interrupt. Read operation: 0 = Associated interrupt status Disabled. 1 = Associated interrupt status Enabled. **Note:** Read value indicates the current enable status.

IRQ0 ~ IRQ31 Set-pending Control Register (NVIC_ISPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ISPR	SCS_BA+0x200	R/W	IRQ0 ~ IRQ31 Set-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**SETPEND**	**Set Interrupt Pending Register** Write operation: 0 = No effect. 1 = Write 1 to set pending state. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Read operation: 0 = Associated interrupt in not in pending status. 1 = Associated interrupt is in pending status. **Note:** Read value indicates the current pending status.

IRQ0 ~ IRQ31 Clear-pending Control Register (NVIC_ICPR)

Register	Offset	R/W	Description	Reset Value
NVIC_ICPR	SCS_BA+0x280	R/W	IRQ0 ~ IRQ31 Clear-Pending Control Register	0x0000_0000

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

Bits	Description	
[31:0]	**CLRPEND**	**Clear Interrupt Pending Register** Write operation: 0 = No effect. 1 = Write 1 to clear pending state. Each bit represents an interrupt number from IRQ0 ~ IRQ31 (Vector number from 16 ~ 47). Read operation: 0 = Associated interrupt in not in pending status. 1 = Associated interrupt is in pending status. **Note:** Read value indicates the current pending status.

IRQ0 ~ IRQ3 Interrupt Priority Register (NVIC_IPR0)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR0	SCS_BA+0x400	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**	**Priority of IRQ3** 0 denotes the highest priority and 3 denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_2**	**Priority of IRQ2** 0 denotes the highest priority and 3 denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_1**	**Priority of IRQ1** 0 denotes the highest priority and 3 denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_0**	**Priority of IRQ0** 0 denotes the highest priority and 3 denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ4 ~ IRQ7 Interrupt Priority Register (NVIC_IPR1)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR1	SCS_BA+0x404	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**	**Priority of IRQ7** 0 denotes the highest priority and 3 denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_6**	**Priority of IRQ6** 0 denotes the highest priority and 3 denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_5**	**Priority of IRQ5** 0 denotes the highest priority and 3 denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_4**	**Priority of IRQ4** 0 denotes the highest priority and 3 denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ8 ~ IRQ11 Interrupt Priority Register (NVIC_IPR2)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR2	SCS_BA+0x408	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**	**Priority of IRQ11** 0 denotes the highest priority and 3 denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_10**	**Priority of IRQ10** 0 denotes the highest priority and 3 denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_9**	**Priority of IRQ9** 0 denotes the highest priority and 3 denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_8**	**Priority of IRQ8** 0 denotes the highest priority and 3 denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ12 ~ IRQ15 Interrupt Priority Register (NVIC_IPR3)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR3	SCS_BA+0x40C	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**	**Priority of IRQ15** 0 denotes the highest priority and 3 denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_14**	**Priority of IRQ14** 0 denotes the highest priority and 3 denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_13**	**Priority of IRQ13** 0 denotes the highest priority and 3 denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_12**	**Priority of IRQ12** 0 denotes the highest priority and 3 denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ16 ~ IRQ19 Interrupt Priority Register (NVIC_IPR4)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR4	SCS_BA+0x410	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**	**Priority of IRQ19** 0 denotes the highest priority and 3 denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_18**	**Priority of IRQ18** 0 denotes the highest priority and 3 denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_17**	**Priority of IRQ17** 0 denotes the highest priority and 3 denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_16**	**Priority of IRQ16** 0 denotes the highest priority and 3 denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ20 ~ IRQ23 Interrupt Priority Register (NVIC_IPR5)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR5	SCS_BA+0x414	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**	**Priority of IRQ23** 0 denotes the highest priority and 3 denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_22**	**Priority of IRQ22** 0 denotes the highest priority and 3 denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_21**	**Priority of IRQ21** 0 denotes the highest priority and 3 denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_20**	**Priority of IRQ20** 0 denotes the highest priority and 3 denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ24 ~ IRQ27 Interrupt Priority Register (NVIC_IPR6)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR6	SCS_BA+0x418	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**	**Priority of IRQ27** 0 denotes the highest priority and 3 denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_26**	**Priority of IRQ26** 0 denotes the highest priority and 3 denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_25**	**Priority of IRQ25** 0 denotes the highest priority and 3 denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_24**	**Priority of IRQ24** 0 denotes the highest priority and 3 denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

IRQ28 ~ IRQ31 Interrupt Priority Register (NVIC_IPR7)

Register	Offset	R/W	Description	Reset Value
NVIC_IPR7	SCS_BA+0x41C	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**	**Priority of IRQ31** 0 denotes the highest priority and 3 denotes the lowest priority.
[29:24]	**Reserved**	Reserved.
[23:22]	**PRI_30**	**Priority of IRQ30** 0 denotes the highest priority and 3 denotes the lowest priority.
[21:16]	**Reserved**	Reserved.
[15:14]	**PRI_29**	**Priority of IRQ29** 0 denotes the highest priority and 3 denotes the lowest priority.
[13:8]	**Reserved**	Reserved.
[7:6]	**PRI_28**	**Priority of IRQ28** 0 denotes the highest priority and 3 denotes the lowest priority.
[5:0]	**Reserved**	Reserved.

6.2.11.7 Interrupt Source Control Registers

Besides the interrupt control registers associated with the NVIC, the NDA102 series also implements some specific control registers to facilitate the interrupt functions, including ”NMI source selection” and “IRQ number identity”, which are described below.

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

Register	Offset	R/W	Description	Reset Value
INT Base Address: **INT_BA = 0x5000_0300**				
INT_NMICTL	INT_BA+0x80	R/W	NMI Source Interrupt Select Control Register	0x0000_0000
INT_IRQSTS	INT_BA+0x84	R/W	MCU IRQ Number Identity Register	0x0000_0000

NMI Interrupt Source Select Control Register (INT_NMICTL)

Register	Offset	R/W	Description	Reset Value
INT_NMICTL	INT_BA+0x80	R/W	NMI Source Interrupt Select 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							**NMISELEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**NMISEL**				

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**NMISELEN**	**NMI Interrupt Enable Bit (Write Protected)** 0 = NMI interrupt Disabled. 1 = NMI interrupt Enabled. **Note:** This bit is the protected bit, and programming it needs to write 0x59, 0x16, and 0x88 to address 0x5000_0100 to disable register protection. Refer to the register SYS_REGLCTL at address SYS_BA+0x100.
[7:5]	**Reserved**	Reserved.
[4:0]	**NMISEL**	**NMI Interrupt Source Selection** The NMI interrupt to Cortex®-M0 can be selected from one of the peripheral interrupt by setting NMTSEL.

MCU Interrupt Request Source Register (INT_IRQSTS)

Register	Offset	R/W	Description	Reset Value
INT_IRQSTS	INT_BA+0x84	R/W	MCU IRQ Number Identity Register	0x0000_0000

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

Bits	Description	
[31:0]	**IRQ**	**MCU IRQ Source Register** The IRQ collects all the interrupts from the peripherals and generates the synchronous interrupt to Cortex®-M0 core. There is one mode to generate interrupt to Cortex®-M0 - the normal mode. The IRQ collects all interrupts from each peripheral and synchronizes them then interrupts the Cortex®-M0. When the IRQ[n] is 0, setting IRQ[n] to 1 will generate an interrupt to Cortex®-M0 NVIC[n]. When the IRQ[n] is 1 (i.e. an interrupt is assert), setting 1 to the MCU_bit[n] will clear the interrupt and setting IRQ[n] 0 has no effect.

6.2.12 System Control Registers

Key control and status features of Cortex®-M0 are managed centrally in a System Control Block within the System Control Registers.

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

6.2.12.1 System Control Register Memory Map

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

Register	Offset	R/W	Description	Reset Value
SCS Base Address: **SCS_BA = 0xE000_E000**				
SCS_CPUID	SCS_BA+0xD00	R	CPUID Base Register	0x410C_C200
SCS_ICSR	SCS_BA+0xD04	R/W	Interrupt Control State Register	0x0000_0000
SCS_AIRCR	SCS_BA+0xD0C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000
SCS_SCR	SCS_BA+0xD10	R/W	System Control Register	0x0000_0000
SCS_SHPR2	SCS_BA+0xD1C	R/W	System Handler Priority Register 2	0x0000_0000
SCS_SHPR3	SCS_BA+0xD20	R/W	System Handler Priority Register 3	0x0000_0000

6.2.12.2 System Control Register Description

CPUID Base Register (CPUID)

Register	Offset	R/W	Description	Reset Value
SCS_CPUID	SCS_BA+0xD00	R	CPUID Base Register	0x410C_C200

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

Bits	Description	
[31:24]	**IMPLEMENTER**	**Implementer Code** Implementer code assigned by ARM (ARM = 0x41).
[23:20]	**Reserved**	Reserved.
[19:16]	**PART**	**Architecture of the Processor** Reads as 0xC for ARMv6-M parts
[15:4]	**PARTNO**	**Part Number of the Processor** Reads as 0xC20.
[3:0]	**REVISION**	**Revision Number** Reads as 0x0

Interrupt Control State Register (ICSR)

Register	Offset	R/W	Description	Reset Value
SCS_ICSR	SCS_BA+0xD04	R/W	Interrupt Control State Register	0x0000_0000

31	30	29	28	27	26	25	24
NMIPENDSET	**Reserved**		**PENDSVSET**	**PENDSVCLR**	**PENDSTSET**	**PENDSTCLR**	**Reserved**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ISRPREEMPT	**ISRPENDING**	**Reserved**	**VECTPENDING**				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VECTPENDING				**Reserved**			**VECTACTIVE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
VECTACTIVE							

Bits	Description	
[31]	**NMIPENDSET**	**NMI Set-pending Bit** Write Operation: 0 = No effect. 1 = Changes NMI exception state to pending. Read Operation: 0 = NMI exception not pending. 1 = NMI exception pending. **Note:** Because NMI is the highest-priority exception, normally the processor entersthe NMI exception handler as soon as it detects a write of 1 to this bit. Entering thehandler then clears this bit to 0. This means a read of this bit by the NMI exceptionhandler returns 1 only if the NMI signal is reasserted while the processor is executingthat handler.
[30:29]	**Reserved**	Reserved.
[28]	**PENDSVSET**	**PendSV Set-pending Bit** Write Operation: 0 = No effect. 1 = Changes PendSV exception state to pending. Read Operation: 0 = PendSV exception is not pending. 1 = PendSV exception is pending. **Note:** Writing 1 to this bit is the only way to set the PendSV exception state to pending
[27]	**PENDSVCLR**	**PendSV Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes the pending state from the PendSV exception. This bit is write-only. To clear the PENDSV bit, you must “write 0 to PENDSVSET andwrite 1 to PENDSVCLR” at the same time.
[26]	**PENDSTSET**	**SysTick Exception Set-pending Bit**

		Write Operation: 0 = No effect. 1 = Changes SysTick exception state to pending. Read Operation: 0 = SysTick exception is not pending. 1 = SysTick exception is pending.
[25]	**PENDSTCLR**	**SysTick Exception Clear-pending Bit** Write Operation: 0 = No effect. 1 = Removes the pending state from the SysTick exception. **Note:** This bit is write-only. When you want to clear PENDST bit, you must "write 0 toPENDSTSET and write 1 to PENDSTCLR" at the same time.
[24]	**Reserved**	Reserved.
[23]	**ISRPREEMPT**	**Interrupt Preempt Bit(Read Only)** If set, a pending exception will be serviced on exit from the debug halt state
[22]	**ISRPENDING**	**Interrupt Pending Flag,Excluding NMI and Faults (Read Only)** 0 = Interrupt not pending. 1 = Interrupt pending.
[21]	**Reserved**	Reserved.
[20:12]	**VECTPENDING**	**Exception Number of the Highest Priority Pending Enabled Exception** 0 = No pending exceptions. Non-zero = Exception number of the highest priority pending enabled exception.
[11:9]	**Reserved**	Reserved.
[8:0]	**VECTACTIVE**	**Contains the Active Exception Number** 0 = Thread mode. Non-zero = Exception number of the currently active exception.

Application Interrupt and Reset Control Register (AIRCR)

Register	Offset	R/W	Description	Reset Value
SCS_AIRCR	SCS_BA+0xD0C	R/W	Application Interrupt and Reset Control Register	0xFA05_0000

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

Bits	Description	
[31:16]	**VECTORKEY**	**Register Access Key** Write Operation: When writing to this register, the VECTORKEY field need to be set to 0x05FA, otherwise the write operation would be ignored. The VECTORKEY filed is used to prevent accidental write to this register from resetting the system or clearing of the exception status. Read Operation: Read as 0xFA05.
[15:3]	**Reserved**	Reserved.
[2]	**SYSRESETREQ**	**System Reset Request** Writing this bit 1 will cause a reset signal to be asserted to the chip to indicate a reset is requested. The bit is a write only bit and self-clears as part of the reset sequence.
[1]	**VECTCLRACTIVE**	**Exception Active Status Clear Bit** Reserved for debug use. When writing to the register, user must write 0 to this bit, otherwise behavior is unpredictable.
[0]	**Reserved**	Reserved.

System Control Register (SCR)

Register	Offset	R/W	Description	Reset Value
SCS_SCR	SCS_BA+0xD10	R/W	System 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			**SEVONPEND**	**Reserved**	**SLEEPDEEP**	**SLEEPONEXIT**	**Reserved**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**SEVONPEND**	**Send Event on Pending Bit** 0 = Only enabled interrupts or events can wake-up the processor, disabled interrupts areexcluded. 1 = Enabled events and all interrupts, including disabled interrupts, can wake-up theprocessor. When an event or interrupt enters pending state, the event signal wakes up the processorfrom WFE. If the processor is not waiting for an event, the event is registered and affectsthe next WFE. The processor also wakes up on execution of an SEV instruction or an external event.
[3]	**Reserved**	Reserved.
[2]	**SLEEPDEEP**	**Processor Deep Sleep and Sleep Mode Selection** Controls whether the processor uses sleep or deep sleep as its low power mode: 0 = Sleep mode. 1 = Deep Sleep mode.
[1]	**SLEEPONEXIT**	**Sleep-on-exit Enable Bit** This bit indicates sleep-on-exit when returning from Handler mode to Thread mode. 0 = Do not sleep when returning to Thread mode. 1 = Enter Sleep or Deep Sleep when returning from ISR to Thread mode.Setting this bit to 1 enables an interrupt driven application to avoid returning to an emptymain application.
[0]	**Reserved**	Reserved.

System Handler Priority Register 2 (SHPR2)

Register	Offset	R/W	Description	Reset Value
SCS_SHPR2	SCS_BA+0xD1C	R/W	System Handler Priority Register 2	0x0000_0000

31	30	29	28	27	26	25	24
PRI_11		**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							

Bits	Description	
[31:30]	**PRI_11**	**Priority of System Handler 11 – SVCall** "0" denotes the highest priority and "3" denotes the lowest priority.
[29:0]	**Reserved**	Reserved.

System Handler Priority Register 3 (SHPR3)

Register	Offset	R/W	Description	Reset Value
SCS_SHPR3	SCS_BA+0xD20	R/W	System Handler Priority Register 3	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**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							

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

6.3 Clock Controller

6.3.1 Overview

The clock controller generates 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 the Cortex®-M0 core executes the WFI instruction only if the PDEN (CLK_PWRCTL[7]) bit set to 1. After that, chip enters Power-down mode and waits for wake-up interrupt source triggered to exit Power-down mode. In Power-down mode, the clock controller turns off the 4~24 MHz external high speed crystal (HXT) and 48 MHz internal high speed RC oscillator (HIRC) to reduce the overall system power consumption. Figure 6.3-2 shows the clock generator and the overview of the clock source control.

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

- 32.768 kHz external low speed crystal oscillator (LXT)
- 4~24 MHz external high speed crystal oscillator (HXT)
- 48 MHz internal high speed RC oscillator (HIRC)
- 10 kHz internal low speed RC oscillator (LIRC)

Figure 6.3-1 Clock Generator Block Diagram

Figure 6.3-2 Clock Generator Global View Diagram

6.3.2 Auto Trim

This chip supports auto-trim function: the HIRC trim (48 MHz RC oscillator), according to the accurate LXT (32.768 kHz crystal oscillator), automatically gets accurate HIRC output frequency, 0.25 % deviation within all temperature ranges.

For instance, the system needs an accurate 48MHz clock. In such case, if users do not want to use PLL as the system clock source, they need to solder 32.768 kHz crystal in system, and set FREQSEL (SYS_IRCTCTL[0] trim frequency selection) to “1”, and the auto-trim function will be enabled. Interrupt status bit FREQLOCK (SYS_IRCTISTS[0] HIRC frequency lock status) “1” indicates the HIRC output frequency is accurate within 0.25% deviation. To get better results, it is recommended to set both LOOPSEL (SYS_IRCTCTL[5:4] trim calculation loop) and RETRYCNT (SYS_IRCTCTL[7:6] trim value update limitation count) to “11”.

6.3.3 System Clock and SysTick Clock

The system clock has three clock sources which were generated from clock generator block. The clock source switch depends on the register HCLKSEL (CLK_CLKSEL0[1:0]). The block diagram is shown in Figure 6.3-3.

Figure 6.3-3 System Clock Block Diagram

The clock source of SysTick in the Cortex®-M0 core can use CPU clock or external clock CLKSRC(SYST_CTL[2]). If using external clock, the SysTick clock (STCLK) has 4 clock sources. The clock source switch depends on the setting of the register STCLKSEL (CLK_CLKSEL0[4:3]). The block diagram is shown in Figure 6.3-4.

Figure 6.3-4 SysTick Clock Control Block Diagram

6.3.4 Peripherals Clock Source Selection

The peripheral clock has different clock source switch settings depending on different peripherals. Please refer to the CLK_CLKSEL1 and CLK_APBCLK register description in section 6.3.8. Please note that, while switching clock source from one to another, user must wait until both clock sources are running stabled.

Figure 6.3-5 Peripherals Bus Clock Source Selection for PCLK

	Peripheral Clock Selectable	Ext. CLK (HXT Or LXT)	HIRC	LIRC	HCLK
WDT	Yes	Yes	No	Yes	Yes
WWDT	Yes	Yes	No	Yes	Yes
Timer0	Yes	Yes	Yes	Yes	Yes
Timer1	Yes	Yes	Yes	Yes	Yes
USCI0	Yes	Yes	Yes	Yes	Yes
USCI1	Yes	Yes	Yes	Yes	Yes
ADC	Yes	Yes	Yes	No	Yes
ACMP	No	No	No	No	Yes
ECAP	No	No	No	No	Yes
EBWM	No	No	No	No	Yes
BPWM	No	No	No	No	Yes
HDIV	No	No	No	No	Yes

Table 6.3-1 Peripheral Clock Source Selection Table

Note: For the peripherals those peripheral clock are not selectable, its clock source is fixed to PCLK.

6.3.5 Power-down Mode Clock

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

The clocks still kept active are listed below:

- Clock Generator
- 10 kHz internal low speed oscillator (LIRC) clock
- 32.768 kHz external low speed crystal oscillator (LXT) clock (If PDLXT = 1 and XTLEN[1:0] = 10)
- Peripherals Clock (When 10 kHz low speed oscillator is adopted as clock source)
  - Watchdog Clock
  - Timer 0/1 Clock

6.3.6 Frequency Divider Output

This device is equipped with a power-of-2 frequency divider which is composed of 16 chained divide-by-2 shift registers. One of the 16 shift register outputs selected by a sixteen to one

multiplexer is reflected to the CKO pin. Therefore there are 16 options of power-of-2 divided clocks with the frequency from $F_{in}/2^1$ to $F_{in}/2^{16}$ where F_{in} is input clock frequency to the clock divider.

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

When writing 1 to CLKOEN (CLK_CLKOCTL[4]), the chained counter starts to count. When writing 0 to CLKOEN (CLK_CLKOCTL[4]), the chained counter continuously runs till divided clock reaches low state and stay in low state.

if DIV1EN(CLK_CLKOCTL[5]) set to 1, the frequency divider clock will bypass power-of-2 frequency divider. The frequency divider clock will be output to CLKO pin directly.

Figure 6.3-6 Clock Source of Frequency Divider

Figure 6.3-7 Block Diagram of Frequency Divider

6.3.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
CLK Base Address: **CLK_BA = 0x5000_0200**				
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_001C
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0014
CLK_APBCLK	CLK_BA+0x08	R/W	APB Devices Clock Enable Control Register	0x0000_0001
CLK_CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0000_001B
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xC307_7733
CLK_CLKDIV	CLK_BA+0x20	R/W	Clock Divider Number Register	0x0000_0000
CLK_STATUS	CLK_BA+0x50	R	Clock Status Monitor Register	0x0000_00XX
CLK_CLKOCTL	CLK_BA+0x60	R/W	Clock Output Control Register	0x0000_0000

6.3.8 Register Description

Power-down Control Register (CLK_PWRCTL)

Except the BIT[6], all the other bits are protected, and programming these bits need to write 0x59, 0x16, 0x88 to address 0x5000_0100 to disable register protection. Refer to the SYS_REGLCTL register at address SYS_BA + 0x100.

Register	Offset	R/W	Description	Reset Value
CLK_PWRCTL	CLK_BA+0x00	R/W	System Power-down Control Register	0x0000_001C

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				**HXTGAIN**		**PDLXT**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PDEN	**PDWKIF**	**PDWKIEN**	**PDWKDLY**	**LIRCEN**	**HIRCEN**	**XTLEN**	

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:10]	**HXTGAIN**	**HXT Gain Control (Write Protect)** This is a protected register. Please refer to open lock sequence to program it. Gain control is used to enlarge the gain of crystal to make sure crystal work normally. If gain control is enabled, crystal will consume more power than gain control off. 00 = HXT frequency is lower than from 8 MHz. 01 = HXT frequency is from 8 MHz to 12 MHz. 10 = HXT frequency is from 12 MHz to 16 MHz. 11 = HXT frequency is higher than 16 MHz. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[9]	**PDLXT**	**LXT Alive in Power-down** 0 = LXT will be turned off automatically when chip enters Power-down. 1 = If XTLEN[1:0] are 0x2, LXT keeps active in Power-down.
[8]	**Reserved**	Reserved.
[7]	**PDEN**	**System Power-down Enable Bit (Write Protect)** When this bit is set to 1, Power-down mode is enabled. When chip wakes up from Power-down mode, this bit is auto cleared. Users need to set this bit again for next Power-down. In Power-down mode, HXT and the HIRC will be disabled in this mode, but LXT and LIRC are not controlled by Power-down mode. In Power-down mode, the system clocks are disabled, and ignored the clock source selection. The clocks of peripheral are not controlled by Power-down mode, if the

		peripheral clock source is from LXT or LIRC. 0 = Chip operating normally or chip in idle mode because of WFI/WFE command. 1 = Chip enters Power-down mode when CPU sleep command WFI/WFE. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[6]	**PDWKIF**	**Power-down Mode Wake-up Interrupt Status** Set by "Power-down wake-up event", it indicates that resume from Power-down mode" The flag is set if the GPIO, USCI01, WDT, ACMP01, BOD, TMR01 wake-up occurred. **Note1:** Write 1 to clear the bit to 0. **Note2:** This bit works only if PDWKIEN (CLK_PWRCTL[5]) set to 1.
[5]	**PDWKIEN**	**Power-down Mode Wake-up Interrupt Enable Bit (Write Protect)** 0 = Power-down mode wake-up interrupt Disabled. 1 = Power-down mode wake-up interrupt Enabled. **Note1:** The interrupt will occur when both PDWKIF and PDWKIEN are high. **Note2:** This bit is write protected. Refer to the SYS_REGLCTL register.
[4]	**PDWKDLY**	**Wake-up Delay Counter Enable Bit (Write Protect)** When the chip wakes up from Power-down mode, the clock control will delay certain clock cycles to wait system clock stable. The delayed clock cycle is 4096 clock cycles when chip works at 4~24 MHz external high speed crystal oscillator (HXT), and 256 clock cycles when chip works at 48 MHz internal high speed RC oscillator (HIRC). 0 = Clock cycles delay Disabled. 1 = Clock cycles delay Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[3]	**LIRCEN**	**LIRC Enable Bit (Write Protect)** 0 = 10 kHz internal low speed RC oscillator (LIRC) Disabled. 1 = 10 kHz internal low speed RC oscillator (LIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**HIRCEN**	**HIRC Enable Bit (Write Protect)** 0 = 48 MHz internal high speed RC oscillator (HIRC) Disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC) Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[1:0]	**XTLEN**	**XTL Enable Bit (Write Protect)** These two bits are default set to "00" and the XT_IN and XT_OUT pins are GPIO. 00 = XT_IN and XT_OUT are GPIO, disable both LXT & HXT (default). 01 = HXT Enabled. 10 = LXT Enabled. 11 = XT_IN is external clock input pin, XT_OUT is GPIO. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Register/Instruction Mode	SLEEPDEEP (SCS_SCR[2])	PDEN (CLK_PWRCTL [7])	CPU Run WFI Instruction	Clock Disable
Normal operation	0	0	NO	All clocks are controlled by control register.
Idle mode (CPU enters Sleep mode)	0	0	YES	Only CPU clock is disabled.
Power-down mode (CPU enters Deep Sleep mode)	1	1	YES	Most clocks are disabled except LIRC/LXT, and only RTC/WDT/Timer peripheral clocks still enable if their clock sources are selected as LIRC/LXT.

Table 6.3-2 Power-down Mode Control Table

When the chip enters Power-down mode, user can wake up chip by some interrupt sources. User should enable the related interrupt sources and NVIC IRQ enable bits (NVIC_ISER) before set PDEN bit in CLK_PWRCTL[7] to ensure chip can enter Power-down and wake-up successfully.

AHB Devices Clock Enable Control Register (CLK_AHBCLK)

The bits in this register are used to enable/disable clock for system clock, AHB bus devices clock.

Register	Offset	R/W	Description	Reset Value
CLK_AHBCLK	CLK_BA+0x04	R/W	AHB Devices Clock Enable Control Register	0x0000_0014

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			**HDIVCKEN**	**Reserved**	**ISPCKEN**	**Reserved**	

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**HDIVCKEN**	**Hardware Divider Controller Clock Enable Bit** 0 = HDIV peripheral clock Disabled. 1 = HDIV peripheral clock Enabled.
[3]	**Reserved**	Reserved.
[2]	**ISPCKEN**	**Flash ISP Controller Clock Enable Bit** 0 = Flash ISP peripheral clock Disabled. 1 = Flash ISP peripheral clock Enabled.
[1:0]	**Reserved**	Reserved.

APB Devices Clock Enable Control Register (CLK_APBCLK)

The bits in this register are used to enable/disable clock for peripheral controller clocks.

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

31	30	29	28	27	26	25	24
Reserved	**ACMPCKEN**	**Reserved**	**ADCCKEN**	**Reserved**		**USCI1CKEN**	**USCI0CKEN**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BPWMCKEN	**Reserved**		**EPWMCKEN**	**Reserved**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved			**PGACKEN**	**Reserved**		**ECAPCKEN**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**CLKOCKEN**	**Reserved**		**TMR1CKEN**	**TMR0CKEN**	**Reserved**	**WDTCKEN**

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**ACMPCKEN**	**Analog Comparator Clock Enable Bit** 0 = Analog comparator clock Disabled. 1 = Analog comparator clock Enabled.
[29]	**Reserved**	Reserved.
[28]	**ADCCKEN**	**Analog-digital-converter (ADC) Clock Enable Bit** 0 = ADC clock Disabled. 1 = ADC clock Enabled.
[27:26]	**Reserved**	Reserved.
[25]	**USCI1CKEN**	**USCI1 Clock Enable Bit** 0 = USCI1 clock Disabled. 1 = USCI1 clock Enabled.
[24]	**USCI0CKEN**	**USCI0 Clock Enable Bit** 0 = USCI0 clock Disabled. 1 = USCI0 clock Enabled.
[23]	**BPWMCKEN**	**Basic PWM Channel 0/1 Clock Enable Bit** 0 = BPWM channel 0/1 clock Disabled. 1 = BPWM channel 0/1 clock Enabled.
[22:21]	**Reserved**	Reserved.
[20]	**EPWMCKEN**	**Enhanced PWM Clock Enable Bit** 0 = EPWM clock Disabled. 1 = EPWM clock Enabled.

[19:13]	**Reserved**	Reserved.
[12]	**PGACKEN**	**PGA Clock Enable Bit** 0 = PGA clock Disabled. 1 = PGA clock Enabled.
[11:9]	**Reserved**	Reserved.
[8]	**ECAPCKEN**	**Input Capture Clock Enable Bit** 0 = ECAP clock Disabled. 1 = ECAP clock Enabled.
[7]	**Reserved**	Reserved.
[6]	**CLKOCKEN**	**CLKO Clock Enable Bit** 0 = CLKO clock Disabled. 1 = CLKO clock Enabled.
[5:4]	**Reserved**	Reserved.
[3]	**TMR1CKEN**	**Timer1 Clock Enable Bit** 0 = Timer1 clock Disabled. 1 = Timer1 clock Enabled.
[2]	**TMR0CKEN**	**Timer0 Clock Enable Bit** 0 = Timer0 clock Disabled. 1 = Timer0 clock Enabled.
[1]	**Reserved**	Reserved.
[0]	**WDTCKEN**	**Watchdog Timer Clock Enable Bit (Write Protect)** 0 = Watchdog timer clock Disabled. 1 = Watchdog timer clock Enabled. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Clock Source Select Control Register 0 (CLK_CLKSEL0)

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL0	CLK_BA+0x10	R/W	Clock Source Select Control Register 0	0x0000_001B

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			**STCLKSEL**		**Reserved**	**HCLKSEL**	

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:3]	**STCLKSEL**	**Cortex®-M0 SysTick Clock Source Selection (Write Protect)** If SYST_CTL[2]=0, SysTick uses the clock source listed below. 00 = Clock source from HXT/LXT. 01 = Clock source from (HXT or LXT)/2. 10 = Clock source from HCLK/2. 11 = Clock source from HIRC/2. Other = Reserved. **Note:** if SysTick clock source is not from HCLK (i.e. SYST_CTL[2] = 0), SysTick clock source must less than or equal to HCLK/2. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.
[2]	**Reserved**	Reserved.
[1:0]	**HCLKSEL**	**HCLK Clock Source Selection (Write Protect)** Before clock switching, the related clock sources (both pre-select and new-select) must be turned on. 00 = Clock source from HXT/LXT. 01 = Clock source from LIRC. 11= Clock source from HIRC. Other = Reserved. **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Clock Source Select Control Register 1 (CLK_CLKSEL1)

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

Register	Offset	R/W	Description	Reset Value
CLK_CLKSEL1	CLK_BA+0x14	R/W	Clock Source Select Control Register 1	0xC307_7733

31	30	29	28	27	26	25	24
CLKOSEL		**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**TMR1SEL**			**Reserved**	**TMR0SEL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**ADCSEL**		**Reserved**		**WDTSEL**	

Bits	Description	
[31:30]	**CLKOSEL**	**Clock Divider Clock Source Selection** 00 = Clock source from external crystal oscillator (HXT or LXT). 01 = Reserved. 10 = Clock source from HCLK. 11 = Clock source from 48 MHz internal high speed RC oscillator (HIRC).
[29:15]	**Reserved**	Reserved.
[14:12]	**TMR1SEL**	**TIMER1 Clock Source Selection** 000 = Clock source from external crystal oscillator (HXT or LXT). 001 = Clock source from 10 kHz internal low speed RC oscillator (LIRC). 010 = Clock source from HCLK. 011 = Clock source from external clock T1 pin. 111 = Clock source from 48 MHz internal high speed RC oscillator (HIRC). Others = Reserved.
[11]	**Reserved**	Reserved.
[10:8]	**TMR0SEL**	**TIMER0 Clock Source Selection** 000 = Clock source from external crystal oscillator (HXT or LXT). 001 = Clock source from 10 kHz internal low speed RC oscillator (LIRC). 010 = Clock source from HCLK. 011 = Clock source from external clock T0 pin. 111 = Clock source from 48 MHz internal high speed RC oscillator (HIRC). Others = Reserved.
[7:6]	**Reserved**	Reserved.

[5:4]	**ADCSEL**	**ADC Peripheral Clock Source Selection** 00 = Clock source from external crystal oscillator (HXT or LXT). 01 = Reserved. 10 = Clock source is from HCLK. 11 = Clock source from 48 MHz internal high speed RC oscillator (HIRC).
[3:2]	**Reserved**	Reserved.
[1:0]	**WDTSEL**	**Watchdog Timer Clock Source Selection (Write Protect)** 00 = Clock source from external crystal oscillator (HXT or LXT). 01 = Reserved. 10 = Clock source from HCLK0/2048. 11 = Clock source from 10 kHz internal low speed RC oscillator (LIRC). **Note:** This bit is write protected. Refer to the SYS_REGLCTL register.

Clock Divider Number Register (CLK_CLKDIV)

Register	Offset	R/W	Description	Reset Value
CLK_CLKDIV	CLK_BA+0x20	R/W	Clock Divider Number Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**ADCDIV**	**ADC Clock Divide Number From ADC Clock Source** ADC clock frequency = (ADC clock source frequency) / (ADCDIV + 1).
[15:4]	**Reserved**	Reserved.
[3:0]	**HCLKDIV**	**HCLK Clock Divide Number From HCLK Clock Source** HCLK clock frequency = (HCLK clock source frequency) / (HCLKDIV + 1).

Clock Status Monitor Register (CLK_STATUS)

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

Register	Offset	R/W	Description	Reset Value
CLK_STATUS	CLK_BA+0x50	R	Clock Status Monitor 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**
CLKSFAIL	**Reserved**		**HIRCSTB**	**LIRCSTB**	**Reserved**		**XTLSTB**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**CLKSFAIL**	**Clock Switching Fail Flag (Read Only)** This bit is updated when software switches system clock source. If switch target clock is stable, this bit will be set to 0. If switch target clock is not stable, this bit will be set to 1. 0 = Clock switching success. 1 = Clock switching failure. **Note:** Write 1 to clear the bit to 0.
[6:5]	**Reserved**	Reserved.
[4]	**HIRCSTB**	**HIRC Clock Source Stable Flag (Read Only)** 0 = 48 MHz internal high speed RC oscillator (HIRC) clock is not stable or disabled. 1 = 48 MHz internal high speed RC oscillator (HIRC) clock is stabe and enabled.
[3]	**LIRCSTB**	**LIRC Clock Source Stable Flag (Read Only)** 0 = 10 kHz internal low speed RC oscillator (LIRC) clock is not stable or disabled. 1 = 10 kHz internal low speed RC oscillator (LIRC) clock is stable and enabled.
[2:1]	**Reserved**	Reserved.
[0]	**XTLSTB**	**XTL Clock Source Stable Flag (Read Only)** 0 = External crystal oscillator (HXT or LXT) clock is not stable or disabled. 1 = External crystal oscillator (HXT or LXT) clock is stable and enabled.

Clock Output Control Register (CLK_CLKOCTL)

Register	Offset	R/W	Description	Reset Value
CLK_CLKOCTL	CLK_BA+0x60	R/W	Clock Output 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		DIV1EN	CLKOEN	FREQSEL			

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**DIV1EN**	**Clock Output Divide One Enable Bit** 0 = Clock Output will output clock with source frequency divided by FREQSEL. 1 = Clock Output will output clock with source frequency.
[4]	**CLKOEN**	**Clock Output Enable Bit** 0 = Clock Output function Disabled. 1 = Clock Output function Enabled.
[3:0]	**FREQSEL**	**Clock Output Frequency Selection** The formula of output frequency is $F_{out} = F_{in}/2^{(N+1)}$. F_{in} is the input clock frequency. F_{out} is the frequency of divider output clock. N is the 4-bit value of FREQSEL[3:0].

6.4 Flash Memory Controller (FMC)

6.4.1 Overview

The NDA102 series is equipped with 29.5 Kbytes on chip embedded Flash for application program memory (APROM) that can be updated through ISP procedure. In System Programming (ISP) function enables user to update program memory when chip is soldered on PCB. After chip powered on Cortex®-M0 CPU fetches code from APROM or LDROM decided by boot select (CBS) in Config0. By the way, the NDA102 series also provides Data Flash Region, where the Data Flash is shared with original program memory and its start address is configurable and defined by user in Config1. The Data Flash size is defined by user depending on the application request. Security program memory (SPROM) provides user to protect any program code within SPROM.

6.4.2 Features

- Running up to 48 MHz with one wait state and 24 MHz without wait state for discontinuous address read access
- 29.5 Kbytes application program memory (APROM)
- 2 Kbytes in system programming (ISP) loader program memory (LDROM)
- Programmable Data Flash start address and memory size with 512 bytes page erase unit
- Three 512 bytes security program memory (SPROM)
- Supports In-System-Programming (ISP) / In-Application-Programming (IAP) to update embedded Flash memory.

6.4.3 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 in Figure 6.4-1.

Figure 6.4-1 Flash Memory Control Block Diagram

6.4.4 Functional Description

6.4.4.1 Flash Memory Organization

The NDA102 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 register. The size of each erase unit is 512 bytes. Data Flash is shared with original program memory, the size and start address are defined by user depending on the application request.

User configuration provides several bytes to control system logic, such as Flash security lock, boot selection, 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 register during chip powered on.

In the 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[1:0] = 1, CPU will be able to fetch code on LDROM from 0x0000 ~ 0x07FF. However, if user want to read LDROM by ISP, they still need to read the address of LDROM as 0x0010_0000 ~ 0x0010_07FF.

Table 6.4-1 shows the address mapping information of APROM, LDROM, Data Flash and user configuration.

Block Name	DFEN	Size	Start Address	End Address
APROM	0	(29.5-0.5*N) Kbytes	0x0000_0000	DFBA-1
APROM	1	29.5 Kbytes	0x0000_0000	0x0000_75FF
Data Flash	0	0.5*N Kbytes	DFBA	0x0000_75FF
Data Flash	1	N/A	N/A	N/A
LDROM	x	2 Kbytes	0x0010_0000	0x0010_07FF
SPROM0	x	0.5 Kbytes	0x0020_0000	0x0020_01FF
SPROM1	x	0.5 Kbytes	0x0024_0000	0x0024_01FF
SPROM2	x	0.5 Kbytes	0x0028_0000	0x0028_01FF
User Configuration	x	2 words	0x0030_0000	0x0030_0004

Table 6.4-1 Flash Memory Address Map

The Flash memory organization is shown in Figure 6.4-2:

Figure 6.4-2 Flash Memory Organization

6.4.4.2 Data Flash

The NDA102 series provides Data Flash for user to store data which is read/write thru ISP registers. The erase unit is 512 bytes. When a word will be changed, all 128 words need to be copied to another page or SRAM in advance. The Data Flash base address is defined by DFBA if DFEN bit in Config0 is enabled. For example, for 4K/2K/1K/0KB Data Flash, the DFBA setting value is listed in Table 6.4-2.

Data Flash **APROM**	**4KB** **(DFEN=0)**	**2KB** **(DFEN=0)**	**1KB** **(DFEN=0)**	**0KB** **(DFEN=1)**
29.5K Flash	DFBA=0x0000_6600	DFBA=0x0000_6E00	DFBA=0x0000_7200	DFEN=1

Table 6.4-2 Data Flash Table

Figure 6.4-3 Data Flash Shared with APROM

6.4.4.3 Security Program Memory (SPROM)

The NDA102 series provides security program memory for user to store instruction of security. It is read/write through ISP procedure and ICE, and this memory cannot be erased by “whole chip erase command” but “page erase command”. The last byte of SPROM memory is used to identify the code is secured or non-secured. Please refer to Table 6.4-4, which shows that security program memory only allows CPU performs instruction fetch and page-erase operation when it is secured code.

- (The last byte= 0xFF): Non-secured code

	ICE Debug	ISP/IAP	CPU Data	CPU Instruction
Whole chip erase	-	-	-	-
Page-erase	-	√	-	-
Program	-	√	-	-
Read	√	√	√	√

- (The last byte=Others): Secured code

	ICE Debug	ISP/IAP	CPU Data	CPU Instruction
Whole chip erase	-	-	-	-
Page-erase	-	√	-	-
Program	-	-	-	-
Read	00h	00h	00h	CPU Instruction

	SPROM0/1/2			
	0x200000 ~ 0x2001FF/0x240000 ~ 0x2401FF/0x280000 ~ 0x2801FF			
	ISP/IAP/ICP/Writer		ICE	
	Lock	unLock	Lock	unLock
whole chip erase	-	-	-	-
page erase	√	√	-	-
program	-	√	-	-
read instruction	√	√	√	√
read data	00h	√	00h	√

Figure 6.4-4 SPROM Security Mode

6.4.4.4 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
Reserved				GPA5RINI		GPA4RINI	
23	22	21	20	19	18	17	16
GPA3RINI		GPA2RINI		GPA1RINI		GPA0RINI	
15	14	13	12	11	10	9	8
CBOV			CBORST	CBODEN	CIOINI	Reserved	
7	6	5	4	3	2	1	0
CBS		Reserved				LOCK	DFEN

Config0	Address = 0x0030_0000	
Bits	**Description**	
[31:28]	**Reserved**	Reserved.
[27:26]	**GPA5RINI**	**POWER-oN Pull Resistor Initial State Selection** 00 = Pull-low Resistor enabled. 01 = Pull-high Resistor enabled. 1x = Pull-high/low Resistor disabled. GPA5 set as this state mode after power-on.
[25:24]	**GPA4RINI**	**POWER-oN Pull Resistor Initial State Selection** 00 = Pull-low Resistor enabled. 01 = Pull-high Resistor enabled. 1x = Pull-high/low Resistor disabled. GPA4 set as this state mode after power-on.
[23:22]	**GPA3RINI**	**POWER-oN Pull Resistor Initial State Selection** 00 = Pull-low Resistor enabled. 01 = Pull-high Resistor enabled. 1x = Pull-high/low Resistor disabled. GPA3 set as this state mode after power-on.
[21:20]	**GPA2RINI**	**POWER-oN Pull Resistor Initial State Selection** 00 = Pull-low Resistor enabled. 01 = Pull-high Resistor enabled. 1x = Pull-high/low Resistor disabled. GPA2 set as this state mode after power-on.
[19:18]	**GPA1RINI**	**POWER-oN Pull Resistor Initial State Selection** 00 = Pull-low Resistor enabled. 01 = Pull-high Resistor Enabled. 1x = Pull-high/low Resistor Disabled. **Note:** GPA1 is set as this state mode after power-on.
[17:16]	**GPA0RINI**	**POWER-oN Pull Resistor Initial State Selection** 00 = Pull-low Resistor Enabled.

Config0	Address = 0x0030_0000	
Bits	**Description**	
		01 = Pull-high Resistor Enabled. 1x = Pull-high/low Resistor Disabled. GPA0 is set as this state mode after power-on.
[15:13]	**CBOV**	**Brown-out Voltage Selection** Brown-out voltages are as follows: 000 = 2.0V. 001 = 2.2V. 010 = 2.4V. 011 = 2.7V. 100 = 3.0V. 101 = 3.7V. 110 = 4.0V. 111 = 4.3V.
[12]	**CBORST**	**Brown-out Reset Enable Bit** 0 = Brown-out reset Enabled after power-on. 1 = Brown-out reset Disabled after power-on.
[11]	**CBODEN**	**Brown-out Voltage Enable Bit** 0 = Brown-out Voltage Enabled. 1 = Brown-out Voltage Disabled.
[10]	**CIOINI**	**POWER-oN Initial State Selection** 0 = Power-on mode. 1 = Input tri-state mode. **Note:** All GPIO are set as this state mode after power-on.
[9:8]	**Reserved**	Reserved.
[7:6]	**CBS**	**Chip Boot Selection** 00 = LDROM with IAP function. 01 = LDROM without IAP function. 10 = APROM with IAP function. 11 = APROM without IAP function. For the NDA102 series, user can set CBS[0] = 0 to support IAP function. When CBS[0] = 0, the LDROM is mapping to address 0x100000 and APROM is mapping to address 0x0. User could access them by their address without boot switching. In other words, if IAP function is supported, the code in LDROM and APROM can be called by each other. **Note1:** The BS bit of FMC_ISPCTL can only be used to control boot switching when CBS[0] = 1. **Note2:** VECMAP can only be used to remap page 0 of APROM or LDROM to 0x0~0x1ff when CBS[0] = 0.
[5:2]	**Reserved**	Reserved.
[1]	**LOCK**	**Security Lock** 0 = Flash data locked. 1 = Flash data unlocked. When Flash data is locked, only device ID, unique ID, CRC checksum user configuration can be read by writer and ICP through serial debug interface. Others data is locked as 0xFFFFFFFF. ISP can read data anywhere regardless of LOCK bit value.

Config0	Address = 0x0030_0000	
Bits	**Description**	
[0]	**DFEN**	**Data Flash Enabled** 0 = Data Flash Enabled. 1 = Data Flash Disabled.

Note: The reserved bits of user configuration should be kept as '1'.

Config1 (Address = 0x0030_0004)

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		DFBA					
7	6	5	4	3	2	1	0
DFBA							

Config1	Address = 0x0030_0004	
Bits	**Description**	
[31:14]	**Reserved**	**Reserved**
[13:0]	**DFBA**	**Data Flash Base Address** The 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.

6.4.4.5 Brown-out Detection

The NDA102 series includes the brown-out detection function for monitoring the voltage on V_{DD} pin. If V_{DD} voltage falls below level setting CBOV, the BOD event will be triggered when BOD enabled. User can decide to use BOD reset by enable CBORST or just enable BOD interrupt by NVIC when BOD detected. Because BOD reset is issued whenever V_{DD} voltage falls below the level setting of CBOV, user must make sure the CBOV setting to avoid BOD reset actived after BOD reset enabled.

6.4.4.6 Boot Selection

The NDA102 series provides in system programming (ISP) feature to support to update program memory when chip is mounted on PCB. A dedicated 2 Kbytes program memory (LDROM) is used to store ISP firmware. Users can select to start program fetch from APROM or LDROM by (CBS) in CONFIG0.

Figure 6.4-5 Boot Select (BS) for Power-on Action

CBS[1:0]	Boot Selection
00	**LDROM with IAP function** Chip booting from LDROM, program executing range including SPROM, LDROM and APROM (except APROM's first page). LDROM address is mapping to 0x0010_0000 ~ 0x0010_07FF, and the first 512 bytes of LDROM is mapping to the address 0x0000_0000 ~ 0x0000_01FF at the same time. Both APROM and LDROM are programmable in this mode no matter the code is currently running on LDROM or APROM. Data Flash is not functioned in this mode.
01	**LDROM without IAP function** Chip booting from LDROM, program executing range only including SPROM and LDROM. APROM can only be accessed by ISP commands. LDROM is write-protected in this mode.
10	**APROM with IAP function** Chip booting from APROM, program executing range including SPROM, LDROM and APROM. LDROM address is mapping to 0x0010_0000~0x0010_07FF. Both APROM and LDROM are programmable in this mode no matter the code is currently running on LDROM or APROM. Data Flash is not functioned in this mode.
11	**APROM without IAP function** Chip booting from APROM and program executing range only including SPROM and APROM. LDROM can only

	be access by ISP commands. APROM is write-protected in this mode.

Table 6.4-3 Boot Selection

CBS[1:0]	Boot From	Vector Re-Map	Run In LDROM Write To APROM	Run In APROM Write To LDROM	Run In LDROM Write To LDROM	Run In APROM Write To APROM
00	LDROM	Yes	Yes	-	Yes	-
01	LDROM	-	Yes	-	Yes	-
10	APROM	Yes	-	Yes	-	Yes
11	APROM	-	-	Yes	-	Yes

Table 6.4-4 Boot Selection and Supports Function

Figure 6.4-6 Flash Memory Mapping of CBS in CONFIG0

6.4.4.7 In Application Programming

The NDA102 series provides In-application-programming (IAP) function for user to switch the code executing between APROM and LDROM without reset. User can enable the IAP function by re-booting chip and setting the chip boot selection bits in Config0 (CBS[1:0]) as 10’b or 00’b.

In the case that the chip boots from APROM with the IAP function enabled (CBS[1:0] = 10’b), 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 2 KB LDROM is mapped to 0x0010_0000~ 0x0010_07FF.

In the case that the chip boots from LDROM with the IAP function enabled (CBS[1:0] = 00’b), 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 2 KB LDROM is mapped to 0x0010_0000~0x0010_07FF.

Please refer to Figure 6.4-7 for the address map while IAP is activating.

Figure 6.4-7 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 FMC_ISPADDR 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 FMC_ISPSTS register.

6.4.4.8 In System Programming (ISP)

The NDA102 series supports In-System-Programming which allows a device to be reprogrammed under software control and avoids system fail risk when download or programming fail. Furthermore, the capability to update the application firmware makes a wide range of applications possible.

To supports In-System-Programming, the NDA102 includes LDROM and ISP controller. User can implement their ISP loader programming in LDROM and this loader can programming user application code (APROM) through ISP register. In other words, the loader could provide the ability to update system firmware on board. By ISP loader, various hardware peripheral interfaces make it be easier to receive new program code. 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.

ISP Registers Control Procedure

The NDA102 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 with CBS[0] = 1, he needs to control BS bit of FMC_ISPCTL control register, then reset CPU by SYS_IPRST1 control register. The boot switching flow by BS bit is shown in Figure 6.4-8. Boot switching function by BS bit is only valid when CBS[0] = 1.

Figure 6.4-8 Example Flow of Boot Selection by BS Bit when CBS[0] = 1

Updating APROM by software in LDROM or updating LDROM by software in APROM can avoid a system failure when update fails.

The 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 set them. To unlock the protected register bits, software needs to write 0x59, 0x16 and 0x88 sequentially to REGWRPROT. If register is unlocked successfully, the value of REGWRPROT will be 1. The unlock sequence must not be interrupted by other access; otherwise it may fail to unlock.

After unlocking the protected register bits, user needs to set the FMC_ISPCTL control register to decide to update LDROM, User Configuration, APROM and enable ISP controller.

Once the FMC_ISPCTL register is set properly, user can set FMC_ISPCMD for erase, read or programming. Set ISPADR for target Flash memory based on Flash memory origination. FMC_ISPDAT can be used to set the data to program or used to return the read data according to FMC_ISPCMD.

Finally, set ISPGO bit of FMC_ISPTRG control register to perform the relative ISP register function. The ISPGO bit is self-cleared when ISP register function has been done. To make sure ISP register function has been finished before CPU goes ahead, ISP instruction is used right after ISPGO setting.

Several error conditions are checked after ISP register function is completed. If an error condition occurs, ISP register operation is not started and the ISP fail flag will be set instead. ISPFF flag can only be cleared by software. The next ISP register control procedure can be started even ISPFF bit is kept as 1. Therefore, it is recommended to check the ISPFF bit and clear it after each ISP register operation if it is set to 1.

When the ISPGO bit is set, CPU will wait for ISP operation to finish during this period; the peripheral still keeps working as usual. If any interrupt request occurs, CPU will not service it till ISP operation is finished. When ISP operation is finished, the ISPBUSY bit will be cleared by hardware automatically. User can check whether ISP operation is finished or not by the ISPBUSY bit. User should add ISP instruction next to the instruction in which the ISPGO bit is set 1 to ensure correct execution of the instructions following ISP operation.

Figure 6.4-9 ISP Flow Example

Table 6.4-5 lists ISP commands supported by the NDA102 series.

ISP Command	FMC_ISPCMD	FMC_ISPADDR	FMC_ISPDAT
FLASH Page Erase	0x22	Valid address of Flash memory origination. It must be 512 bytes page alignment.	Don't care
SPROM Page Erase	0x22	Valid address of Flash memory origination. It must be 512 bytes page alignment.	0x0055AA03
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
Read Company ID	0x0B	Don't care	Company ID (0xDA)
Vector Page Re-Map	0x2E	Page in APROM or LDROM It must be 512 bytes page alignment	Don't care
CRC Calculation	0x2D	Start address of CRC calculation.	CRC calculating range
Read CRC	0xD	0x0000_0000	CRC value

Table 6.4-5 ISP Command Table

6.4.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
FMC Base Address: **FMC_BA = 0x5000_C000**				
FMC_ISPCTL	FMC_BA+0x00	R/W	ISP Control Register	0x0000_0000
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000
FMC_ISPCMD	FMC_BA+0x0C	R/W	ISP Command Register	0x0000_0000
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger Register	0x0000_0000
FMC_DFBA	FMC_BA+0x14	R	Data Flash Start Address	0x0000_3800
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP Status Register	0xXXXX_XXXX
FMC_CRCSEED	FMC_BA+0x50	R/W	ISP CRC Seed Register	0xFFFF_FFFF
FMC_CRCCV	FMC_BA+0x54	R	ISP CRC Current Value Register	0xXXXX_XXXX

6.4.6 Register Description

ISP Control Register (FMC_ISPCTL)

Register	Offset	R/W	Description	Reset Value
FMC_ISPCTL	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	**ISPFF**	**LDUEN**	**CFGUEN**	**APUEN**	**SPUEN**	**BS**	**ISPEN**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**ISPFF**	**ISP Fail Flag (Write Protect)** This bit is set by hardware when a triggered ISP meets any of the following conditions: (1) APROM writes to itself if APUEN is set to 0. (2) LDROM writes to itself if LDUEN is set to 0. (3) SPROM writes to itself if SPUEN is set to 0. (4) CONFIG is erased/programmed if CFGUEN is set to 0. (5) Destination address is illegal, such as over an available range. **Note:** Write 1 to clear this bit to 0.
[5]	**LDUEN**	**LDROM Update Enable Bit (Write Protect)** 0 = LDROM cannot be updated. 1 = LDROM can be updated when the MCU runs in APROM.
[4]	**CFGUEN**	**CONFIG Update Enable Bit (Write Protect)** Writing this bit to 1 enables software to update CONFIG value by ISP register control procedure regardless of program code is running in APROM or LDROM. 0 = ISP update User Configuration Disabled. 1 = ISP update User Configuration Enabled.
[3]	**APUEN**	**APROM Update Enable Bit (Write Protect)** 0 = APROM cannot be updated when chip runs in APROM. 1 = APROM can be updated when chip runs in APROM.
[2]	**SPUEN**	**SPROM Update Enable Bit (Write Protect)** 0 = SPROM cannot be updated. 1 = SPROM can be updated when the MCU runs in APROM.

Bits	Description	
[1]	**BS**	**Boot Select (Write Protect)** 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 (CPURF is 1) or system reset (SYSRF) is happened. 0 = Boot from APROM. 1 = Boot from LDROM.
[0]	**ISPEN**	**ISP Enable Bit (Write Protect)** Set this bit to enable ISP function. 0 = ISP function Disabled. 1 = ISP function Enabled.

ISP Address (FMC_ISPADDR)

Register	Offset	R/W	Description	Reset Value
FMC_ISPADDR	FMC_BA+0x04	R/W	ISP Address Register	0x0000_0000

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

Bits	Description	
[31:0]	**ISPADR**	**ISP Address** The NDA102 series supports word program only. ISPADR[1:0] must be kept 00 for ISP operation.

FMC_ISPDAT (ISP Data Register)

Register	Offset	R/W	Description	Reset Value
FMC_ISPDAT	FMC_BA+0x08	R/W	ISP Data Register	0x0000_0000

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

Bits	Description	
[31:0]	**ISPDAT**	**ISP Data** Write data to this register before ISP program operation. Read data from this register after ISP read operation.

ISP Command (FMC_ISPCMD)

Register	Offset	R/W	Description	Reset Value
FMC_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		**CMD**					

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5:0]	**CMD**	**ISP Command** ISP commands are shown below: 0x00 = Read. 0x04 = Read Unique ID. 0x0B = Read Company ID (0xDA). 0x0D = Read CRC32 Checksum Result After Calculating. 0x21 = Program. 0x22 = Page Erase. 0x2D = Run Memory CRC32 Checksum Calculation. 0x2E = Set Vector Page Re-Map.

ISP Trigger Control Register (FMC_ISPTRG)

Register	Offset	R/W	Description	Reset Value
FMC_ISPTRG	FMC_BA+0x10	R/W	ISP Trigger 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**	Reserved.
[0]	**ISPGO**	**ISP Start Trigger (Write Protect)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation is finished. 1 = ISP operation is progressed.

Data Flash Base Address Register (FMC_DFBA)

Register	Offset	R/W	Description	Reset Value
FMC_DFBA	FMC_BA+0x14	R	Data Flash Start Address	0x0000_3800

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

Bits	Description	
[31:0]	**DFBA**	**Data Flash Base Address** This register indicates Data Flash start address. It is a read only register. The Data Flash start address is defined by user. Since on chip Flash erase unit is 512 bytes, it is mandatory to keep bit 8-0 as 0.

Example:

Data Flash	4KB (DFEN=0)	2KB (DFEN=0)	1KB (DFEN=0)	0KB (DFEN=1)
17.5K Flash	DFBA=0x0000_3600	DFBA=0x0000_3E00	DFBA=0x0000_4200	DFEN=1
8K Flash	DFBA=0x0000_1000	DFBA=0x0000_1800	DFBA=0x0000_1C00	DFEN=1
4K Flash	Forbidden	DFBA=0x0000_0800	DFBA=0x0000_0C00	DFEN=1

ISP Status Register (FMC_ISPSTS)

Register	Offset	R/W	Description	Reset Value
FMC_ISPSTS	FMC_BA+0x40	R/W	ISP Status Register	0xXXXX_XXXX

31	30	29	28	27	26	25	24
SCODE			**Reserved**				
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved			**VECMAP**				
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
VECMAP							**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**ISPFF**	**Reserved**			**CBS**		**ISPBUSY**

Bits	Description	
[31:29]	**SCODE**	**Security Code Active Flag** This bit field set by hardware when detecting SPROM secured code is active at Flash initiation, or software writes 1 to this bit to make secured code active; this bit is clear by SPROM page erase operation. 000 = SPROM0/1/2 secured code are inactive. 001 = SPROM0 secured code is active. 010 = SPROM1 secured code is active. 100 = SPROM2 secured code is active. 111 = SPROM0/1/2 Secured code are active.
[28:21]	**Reserved**	Reserved.
[20:9]	**VECMAP**	**Vector Page Mapping Address (Read Only)** 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**	**ISP Fail Flag (Write Protect)** This bit is set by hardware when a triggered ISP meets any of the following conditions: (1) APROM writes to itself if APUEN is set to 0. (2) LDROM writes to itself if LDUEN is set to 0. (3) SPROM writes to itself if SPUEN is set to 0. (4) CONFIG is erased/programmed if CFGUEN is set to 0. (5) Destination address is illegal, such as over an available range. **Note:** Write 1 to clear this bit to 0.
[5:3]	**Reserved**	Reserved.
[2:1]	**CBS**	**Config Boot Selection (Read Only)** This is a mirror of CBS in CONFIG0.

[0]	**ISPBUSY**	**ISP Start Trigger (Read Only)** Write 1 to start ISP operation and this bit will be cleared to 0 by hardware automatically when ISP operation is finished. 0 = ISP operation is finished. 1 = ISP operation is progressed. **Note:** This bit is the same with FMC_ISPTRG bit 0.

ISP CRC Seed Register (FMC_CRCSEED)

Register	Offset	R/W	Description	Reset Value
FMC_CRCSEED	FMC_BA+0x50	R/W	ISP CRC Seed Register	0xFFFF_FFFF

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

Bits	Description	
[31:0]	**CRCSEED**	**CRC Seed Data** This register was provided to be the initial value for CRC operation. Write data to this register before ISP CRC operation. Read data from this register after ISP CRC read operation.

ISP CRC Current Value Register (FMC_CRCCV)

Register	Offset	R/W	Description	Reset Value
FMC_CRCCV	FMC_BA+0x54	R	ISP CRC Current Value Register	0xXXXX_XXXX

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

Bits	Description	
[31:0]	**CRCCV**	**CRC Current Value** This register provided current value of CRC durning calculation.

6.5 General Purpose I/O (GPIO)

6.5.1 Overview

The NDA102 series has up to 22 General Purpose I/O pins. These pins could be shared with other functions depending on the chip configuration. 22 pins are arranged in 4 ports named as PA, PB, PC, and PD. Each of the 22 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, Push-pull output, Open-drain output or Quasi-bidirectional mode. After the chip is reset, the I/O mode of all pins are depending on CIOIN (CONFIG0[10]). Each I/O pin has a very weakly individual pull-up resistor which is about 110 kΩ ~ 300 kΩ for V_{DD} is from 5.0 V to 2.5 V.

Figure 6.5-1 I/O Pin Block Diagram

6.5.2 Features

- Four I/O modes:
  - Quasi-bidirectional mode
  - Push-Pull Output mode
  - Open-Drain Output mode
  - Input only with high impendence mode
- TTL/Schmitt trigger input selectable
- I/O pin can be configured as interrupt source with edge/level setting

- Supports High Drive and High Sink I/O mode
- Supports software selectable slew rate control
- Configurable default I/O mode of all pins after reset by CIOINI (CONFIG0[10]) setting
  - CIOIN = 0, all GPIO pins in Quasi-bidirectional mode after chip reset
  - CIOIN = 1, all GPIO pins in input mode after chip reset
- GPIOA supports the pull-up and pull-low resistor enabled in four I/O modes
- GPIOB to GPIOD internal pull-up resistor enabled only in Quasi-bidirectional I/O mode
- Enabling the pin interrupt function will also enable the wake-up function

6.5.3 Block Diagram

Figure 6.5-2 GPIO Controller Block Diagram

6.5.4 Basic Configuration

The GPIO pin functions are configured in SYS_GPA_MFP, SYS_GPB_MFP, SYS_GPC_MFP, and SYS_GPD_MFP registers.

6.5.5 Functional Description

6.5.5.1 Input Mode

Set MODEn (Px_MODE[2n+1:2n]) to 00 as the Px.n pin is in Input mode and the I/O pin is in tri-state (high impedance) without output drive capability. The PIN (Px_PIN[n]) value reflects the status of the corresponding port pins.

6.5.5.2 Push-pull Output Mode

Set MODEn (Px_MODE[2n+1:2n]) to 01 as the Px.n pin is in Push-pull Output mode and the I/O pin supports digital output function with source/sink current capability. The bit value in the corresponding DOUT (Px_DOUT[n]) is driven on the pin.

Figure 6.5-3 Push-Pull Output

6.5.5.3 Open-drain Mode

Set MODEn (Px_MODE[2n+1:2n]) to 10 as the Px.n pin is in Open-drain mode and the digital output function of I/O pin supports only sink current capability, an external pull-up register is needed for driving high state. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 1, the pin output drives high that is controlled by external pull high resistor.

Figure 6.5-4 Open-Drain Output

6.5.5.4 Quasi-bidirectional Mode

Set MODEn (Px_MODE[2n+1:2n]) to 11 as the Px.n 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 the corresponding DOUT (Px_DOUT[n]) bit must be set to 1. The quasi-bidirectional output is common on the 80C51 and most of its derivatives. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 0, the pin drive a low output on the pin. If the bit value in the corresponding DOUT (Px_DOUT[n]) bit is 1, the pin will check the pin value. If pin value is high, no action takes. If pin state is low, the pin will drive strong high with 2 clock cycles on the pin and then disable the strong output drive. Meanwhile, the pin status is controlled by internal pull-up resistor. Note that the source current capability in quasi-bidirectional mode is only about 200 uA to 30 uA for V_{DD} is from 5.0 V to 2.5 V.

Figure 6.5-5 Quasi-Bidirectional I/O Mode

6.5.6 GPIO Interrupt and Wake-up Function

Each GPIO pin can be set as chip interrupt source by setting correlative RHIEN (Px_INTEN[n+16])/ FLIEN (Px_INTEN[n]) bit and TYPE (Px_INTTYPE[n]). There are five types of interrupt conditions to be selected: low level trigger, high level trigger, falling edge trigger, rising edge trigger and both rising and falling edge trigger. For edge trigger condition, user can enable input signal de-bounce function to prevent unexpected interrupt happened which caused by noise. The de-bounce clock source and sampling cycle period can be set through DBCLKSRC (GPIO_DBCTL[4]) and DBCLKSEL (GPIO_DBCTL[3:0]) register.

The GPIO can also be the chip wake-up source when chip enters Idle/Power-down mode. The setting of wake-up trigger condition is the same as GPIO interrupt trigger.

1. To ensure the I/O status before entering Idle/Power-down mode

When using toggle GPIO to wake-up system, user must make sure the I/O status before entering Idle/Power-down mode according to the relative wake-up settings.

For example, if configuring the wake-up event occurred by I/O rising edge/high level trigger, user must make sure the I/O status of specified pin is at low level before entering Idle/Power-down mode; and if configuring I/O falling edge/low level trigger to trigger a wake-up event, user must make sure the I/O status of specified pin is at high level before entering Power-down mode.

2. To disable the I/O de-bounce function before entering Idle/Power-down mode

If the specified wake-up I/O pin with enabling input signal de-bounce function, system will encounter two GPIO interrupt events while the system is woken up by this GPIO pin. One interrupt event is caused by wake-up function, the other is caused by I/O input de-bounce function. User should be disable the de-bounce function before entering Idle/Power-down mode to avoid the second interrupt event occurred after system woken up.

6.5.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
GPIO Base Address: **GPIO_BA = 0x5000_4000**				
PA_MODE	GPIO_BA+0x000	R/W	PA I/O Mode Control	0x0000_0XXX
PA_DINOFF	GPIO_BA+0x004	R/W	PA Digital Input Path Disable Control	0x0000_0000
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_003F
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_00XX
PA_DBEN	GPIO_BA+0x014	R/W	PA De-Bounce Enable Control Register	0x0000_0000
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Trigger Type Control	0x0000_0000
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control Register	0x0000_0000
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_00XX
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable Register	0x0000_0000
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control Register	0x0000_0000
PA_PLEN	GPIO_BA+0x02C	R/W	PA Pull-Low Control Register	0x0000_0000
PA_PHEN	GPIO_BA+0x030	R/W	PA Pull-High Control Register	0x0000_003F
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0x0000_0XXX
PB_DINOFF	GPIO_BA+0x044	R/W	PB Digital Input Path Disable Control	0x0000_0000
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_001F
PB_DATMSK	GPIO_BA+0x04C	R/W	PB Data Output Write Mask	0x0000_0000
PB_PIN	GPIO_BA+0x050	R	PB Pin Value	0x0000_00XX
PB_DBEN	GPIO_BA+0x054	R/W	PB De-Bounce Enable Control Register	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Trigger Type Control	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control Register	0x0000_0000
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_00XX
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable Register	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control Register	0x0000_0000
PB_PLEN	GPIO_BA+0x06C	R/W	PB Pull-Low Control Register	0x0000_0000

PB_PHEN	GPIO_BA+0x070	R/W	PB Pull-High Control Register	0x0000_001F
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0x0000_0XXX
PC_DINOFF	GPIO_BA+0x084	R/W	PC Digital Input Path Disable Control	0x0000_0000
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_001F
PC_DATMSK	GPIO_BA+0x08C	R/W	PC Data Output Write Mask	0x0000_0000
PC_PIN	GPIO_BA+0x090	R	PC Pin Value	0x0000_00XX
PC_DBEN	GPIO_BA+0x094	R/W	PC De-Bounce Enable Control Register	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Trigger Type Control	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control Register	0x0000_0000
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_00XX
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable Register	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control Register	0x0000_0000
PC_PLEN	GPIO_BA+0x0AC	R/W	PC Pull-Low Control Register	0x0000_0000
PC_PHEN	GPIO_BA+0x0B0	R/W	PC Pull-High Control Register	0x0000_001F
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	0x0000_00XX
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD Digital Input Path Disable Control	0x0000_0000
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_007F
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	0x0000_0000
PD_PIN	GPIO_BA+0x0D0	R	PD Pin Value	0x0000_00XX
PD_DBEN	GPIO_BA+0x0D4	R/W	PD De-Bounce Enable Control Register	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt Trigger Type Control	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control Register	0x0000_0000
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	0x0000_00XX
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger Enable Register	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate Control Register	0x0000_0000
PD_PLEN	GPIO_BA+0x0EC	R/W	PD Pull-Low Control Register	0x0000_0000
PD_PHEN	GPIO_BA+0x0F0	R/W	PD Pull-High Control Register	0x0000_007F
GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x0000_0020
PAn_PDIO n=0,1..5	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X

PBn_PDIO n=0,1..4	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO n=0,1..4	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO n=0,1..6	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X

6.5.8 Register Description

Port A-D I/O Mode Control (Px_MODE)

Register	Offset	R/W	Description	Reset Value
PA_MODE	GPIO_BA+0x000	R/W	PA I/O Mode Control	0x0000_0XXX
PB_MODE	GPIO_BA+0x040	R/W	PB I/O Mode Control	0x0000_0XXX
PC_MODE	GPIO_BA+0x080	R/W	PC I/O Mode Control	0x0000_0XXX
PD_MODE	GPIO_BA+0x0C0	R/W	PD I/O Mode Control	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**
MODE7		**MODE6**		**MODE5**		**MODE4**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
MODE3		**MODE2**		**MODE1**		**MODE0**	

Bits	Description	
[31:16]	**Reserved**	Reserved.
[2n+1:2n] n=0,1..7	**MODEn**	**Port A-d I/O Pin[n] Mode Control** Determine each I/O mode of Px.n pins. 00 = Px.n is in Input mode. 01 = Px.n is in Push-pull Output mode. 10 = Px.n is in Open-drain Output mode. 11 = Px.n is in Quasi-bidirectional mode. **Note1:** The initial value of this field is defined by CIOINI (CONFIG0 [10]). If CIOINI is set to 0, the default value is 0xFFFF_FFFF and all pins will be quasi-bidirectional mode after chip powered on. If CIOINI is set to 1, the default value is 0x0000_0000 and all pins will be input mode after chip powered on. **Note2:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D Digital Input Path Disable Control (Px_DINOFF)

Register	Offset	R/W	Description	Reset Value
PA_DINOFF	GPIO_BA+0x004	R/W	PA Digital Input Path Disable Control	0x0000_0000
PB_DINOFF	GPIO_BA+0x044	R/W	PB Digital Input Path Disable Control	0x0000_0000
PC_DINOFF	GPIO_BA+0x084	R/W	PC Digital Input Path Disable Control	0x0000_0000
PD_DINOFF	GPIO_BA+0x0C4	R/W	PD Digital Input Path Disable Control	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[n+16] n=0,1..7	**DINOFFn**	**Port A-d Pin[n] Digital Input Path Disable Bits** Each of these bits is used to control if the digital input path of corresponding Px.n pin is disabled. If input is analog signal, users can disable Px.n digital input path to avoid input current leakage. 0 = Px.n digital input path Enabled. 1 = Px.n digital input path Disabled (digital input tied to low). **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.
[15:0]	**Reserved**	Reserved.

Port A-D Data Output Value (Px_DOUT)

Register	Offset	R/W	Description	Reset Value
PA_DOUT	GPIO_BA+0x008	R/W	PA Data Output Value	0x0000_003F
PB_DOUT	GPIO_BA+0x048	R/W	PB Data Output Value	0x0000_001F
PC_DOUT	GPIO_BA+0x088	R/W	PC Data Output Value	0x0000_001F
PD_DOUT	GPIO_BA+0x0C8	R/W	PD Data Output Value	0x0000_007F

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**
DOUTn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**DOUTn**	**Port A-d Pin[n] Output Value** Each of these bits controls the status of a Px.n pin when the Px.n is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 0 = Px.n will drive Low if the Px.n pin is configured as Push-pull output, Open-drain output or Quasi-bidirectional mode. 1 = Px.n will drive High if the Px.n pin is configured as Push-pull output or Quasi-bidirectional mode. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D Data Output Write Mask (Px_DATMSK)

Register	Offset	R/W	Description	Reset Value
PA_DATMSK	GPIO_BA+0x00C	R/W	PA Data Output Write Mask	0x0000_0000
PB_DATMSK	GPIO_BA+0x04C	R/W	PB Data Output Write Mask	0x0000_0000
PC_DATMSK	GPIO_BA+0x08C	R/W	PC Data Output Write Mask	0x0000_0000
PD_DATMSK	GPIO_BA+0x0CC	R/W	PD Data Output Write Mask	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**
DATMSKn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**DATMSKn**	**Port A-d Pin[n] Data Output Write Mask** These bits are used to protect the corresponding DOUT (Px_DOUT[n]) bit. When the DATMSK (Px_DATMSK[n]) bit is set to 1, the corresponding DOUT (Px_DOUT[n]) bit is protected. If the write signal is masked, writing data to the protect bit is ignored. 0 = Corresponding DOUT (Px_DOUT[n]) bit can be updated. 1 = Corresponding DOUT (Px_DOUT[n]) bit protected. **Note1:** This function only protects the corresponding DOUT (Px_DOUT[n]) bit, and will not protect the corresponding PDIO (Pxn_PDIO[0]) bit. **Note2:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D Pin Value (Px_PIN)

Register	Offset	R/W	Description	Reset Value
PA_PIN	GPIO_BA+0x010	R	PA Pin Value	0x0000_00XX
PB_PIN	GPIO_BA+0x050	R	PB Pin Value	0x0000_00XX
PC_PIN	GPIO_BA+0x090	R	PC Pin Value	0x0000_00XX
PD_PIN	GPIO_BA+0x0D0	R	PD Pin Value	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**
PINn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**PINn**	**Port A-d Pin[n] Pin Value** Each bit of the register reflects the actual status of the respective Px.n pin. If the bit is 1, it indicates the corresponding pin status is high; else the pin status is low. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D De-bounce Enable Control Register (Px_DBEN)

Register	Offset	R/W	Description	Reset Value
PA_DBEN	GPIO_BA+0x014	R/W	PA De-Bounce Enable Control Register	0x0000_0000
PB_DBEN	GPIO_BA+0x054	R/W	PB De-Bounce Enable Control Register	0x0000_0000
PC_DBEN	GPIO_BA+0x094	R/W	PC De-Bounce Enable Control Register	0x0000_0000
PD_DBEN	GPIO_BA+0x0D4	R/W	PD De-Bounce Enable 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**
DBENn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**DBENn**	**Port A-d Pin[n] Input Signal De-bounce Enable Bits** The DBEN[n] bit is used to enable the de-bounce function for each corresponding bit. If the input signal pulse width cannot 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 source is controlled by DBCLKSRC (GPIO_DBCTL [4]), one de-bounce sample cycle period is controlled by DBCLKSEL (GPIO_DBCTL [3:0]). 0 = Px.n de-bounce function Disabled. 1 = Px.n de-bounce function Enabled. The de-bounce function is valid only for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D Interrupt Type Control (Px_INTTYPE)

Register	Offset	R/W	Description	Reset Value
PA_INTTYPE	GPIO_BA+0x018	R/W	PA Interrupt Trigger Type Control	0x0000_0000
PB_INTTYPE	GPIO_BA+0x058	R/W	PB Interrupt Trigger Type Control	0x0000_0000
PC_INTTYPE	GPIO_BA+0x098	R/W	PC Interrupt Trigger Type Control	0x0000_0000
PD_INTTYPE	GPIO_BA+0x0D8	R/W	PD Interrupt Trigger Type Control	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**
TYPEn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**TYPEn**	**Port A-d Pin[n] Edge or Level Detection Interrupt Trigger Type Control** TYPE (Px_INTTYPE[n]) bit is used to control the triggered interrupt is by level trigger or by edge trigger. 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. 0 = Edge trigger interrupt. 1 = Level trigger interrupt. If the pin is set as the level trigger interrupt, only one level can be set on the registers RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n]). If both levels to trigger interrupt are set, the setting is ignored and no interrupt will occur. The de-bounce function is valid only for edge triggered interrupt. If the interrupt mode is level triggered, the de-bounce enable bit is ignored. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D Interrupt Enable Control Register (Px_INTEN)

Register	Offset	R/W	Description	Reset Value
PA_INTEN	GPIO_BA+0x01C	R/W	PA Interrupt Enable Control Register	0x0000_0000
PB_INTEN	GPIO_BA+0x05C	R/W	PB Interrupt Enable Control Register	0x0000_0000
PC_INTEN	GPIO_BA+0x09C	R/W	PC Interrupt Enable Control Register	0x0000_0000
PD_INTEN	GPIO_BA+0x0DC	R/W	PD Interrupt Enable Control Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[n+16] n=0,1..7	**RHIENn**	**Port A-d Pin[n] Rising Edge or High Level Interrupt Trigger Type Enable Bits** The RHIEN (Px_INTEN[n+16]) bit is used to enable the interrupt for each of the corresponding input Px.n pin. Set bit to 1 also enable the pin wake-up function. When setting the RHIEN (Px_INTEN[n+16]) bit to 1 : If the interrupt is level trigger (TYPE (Px_INTTYPE[n]) bit is set to 1), the input Px.n pin will generate the interrupt while this pin state is at high level. If the interrupt is edge trigger (TYPE (Px_INTTYPE[n]) bit is set to 0), the input Px.n pin will generate the interrupt while this pin state changed from low to high. 0 = Px.n level high or low to high interrupt Disabled. 1 = Px.n level high or low to high interrupt Enabled. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.
[15:8]	**Reserved**	Reserved.
[n] n=0,1..7	**FLIENn**	**Port A-d Pin[n] Falling Edge or Low Level Interrupt Trigger Type Enable Bits** The FLIEN (Px_INTEN[n]) bit is used to enable the interrupt for each of the corresponding input Px.n pin. Set bit to 1 also enable the pin wake-up function. When setting the FLIEN (Px_INTEN[n]) bit to 1 : If the interrupt is level trigger (TYPE (Px_INTTYPE[n]) bit is set to 1), the input Px.n pin will generate the interrupt while this pin state is at low level.

		If the interrupt is edge trigger(TYPE (Px_INTTYPE[n]) bit is set to 0), the input Px.n pin will generate the interrupt while this pin state changed from high to low. 0 = Px.n level low or high to low interrupt Disabled. 1 = Px.n level low or high to low interrupt Enabled. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D Interrupt Source Flag (Px_INTSRC)

Register	Offset	R/W	Description	Reset Value
PA_INTSRC	GPIO_BA+0x020	R/W	PA Interrupt Source Flag	0x0000_00XX
PB_INTSRC	GPIO_BA+0x060	R/W	PB Interrupt Source Flag	0x0000_00XX
PC_INTSRC	GPIO_BA+0x0A0	R/W	PC Interrupt Source Flag	0x0000_00XX
PD_INTSRC	GPIO_BA+0x0E0	R/W	PD Interrupt Source Flag	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**
INTSRCn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**INTSRCn**	**Port A-d Pin[n] Interrupt Source Flag** Write Operation : 0 = No action. 1 = Clear the corresponding pending interrupt. Read Operation : 0 = No interrupt at Px.n. 1 = Px.n generates an interrupt. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D Input Schmitt Trigger Enable Register (Px_SMTEN)

Register	Offset	R/W	Description	Reset Value
PA_SMTEN	GPIO_BA+0x024	R/W	PA Input Schmitt Trigger Enable Register	0x0000_0000
PB_SMTEN	GPIO_BA+0x064	R/W	PB Input Schmitt Trigger Enable Register	0x0000_0000
PC_SMTEN	GPIO_BA+0x0A4	R/W	PC Input Schmitt Trigger Enable Register	0x0000_0000
PD_SMTEN	GPIO_BA+0x0E4	R/W	PD Input Schmitt Trigger 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**
SMTENn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**SMTENn**	**Port A-d Pin[n] Input Schmitt Trigger Enable Bits** 0 = Px.n input schmitt trigger function Disabled. 1 = Px.n input schmitt trigger function Enabled. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port A-D High Slew Rate Control Register (Px_SLEWCTL)

Register	Offset	R/W	Description	Reset Value
PA_SLEWCTL	GPIO_BA+0x028	R/W	PA High Slew Rate Control Register	0x0000_0000
PB_SLEWCTL	GPIO_BA+0x068	R/W	PB High Slew Rate Control Register	0x0000_0000
PC_SLEWCTL	GPIO_BA+0x0A8	R/W	PC High Slew Rate Control Register	0x0000_0000
PD_SLEWCTL	GPIO_BA+0x0E8	R/W	PD High Slew Rate 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**
HSRENn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**HSRENn**	**Port A-d Pin[n] High Slew Rate Control** 0 = Px.n output with basic slew rate. 1 = Px.n output with higher slew rate. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port Pull-low Resistor Control Register (Px_PLEN)

Register	Offset	R/W	Description	Reset Value
PA_PLEN	GPIO_BA+0x02C	R/W	PA Pull-Low Control Register	0x0000_0000
PB_PLEN	GPIO_BA+0x06C	R/W	PB Pull-Low Control Register	0x0000_0000
PC_PLEN	GPIO_BA+0x0AC	R/W	PC Pull-Low Control Register	0x0000_0000
PD_PLEN	GPIO_BA+0x0EC	R/W	PD Pull-Low 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**
PLENn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**PLENn**	**Port A-d Pull-low Resistor Control** 0 = Pull-Low Resistor Disabled. 1 = Pull-Low Resistor Enabled. **Note:** The initial value of **PA_PLEN** were defined by GPAn_RINI (CONFIG0[27:16]). Selected pins will be configured after chip powered on. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Port Pull High Resistor Control Register (Px_PHEN)

Register	Offset	R/W	Description	Reset Value
PA_PHEN	GPIO_BA+0x030	R/W	PA Pull-High Control Register	0x0000_003F
PB_PHEN	GPIO_BA+0x070	R/W	PB Pull-High Control Register	0x0000_001F
PC_PHEN	GPIO_BA+0x0B0	R/W	PC Pull-High Control Register	0x0000_001F
PD_PHEN	GPIO_BA+0x0F0	R/W	PD Pull-High Control Register	0x0000_007F

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**
PHENn							

Bits	Description	
[31:8]	**Reserved**	Reserved.
[n] n=0,1..7	**PHENn**	**Port A-d Pull-high Resistor Control** 0 = Pull-High Resistor Enabled. 1 = Pull-High Resistor Disabled. **Note:** The initial value of **PA_PHEN** were defined by GPAn_RINI (CONFIG0[27:16]). Selected pins will be configured after chip powered on. **Note:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

Interrupt De-bounce Control Register (GPIO_DBCTL)

Register	Offset	R/W	Description	Reset Value
GPIO_DBCTL	GPIO_BA+0x440	R/W	Interrupt De-bounce Control Register	0x0000_0020

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		**ICLKON**	**DBCLKSRC**	**DBCLKSEL**			

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**ICLKON**	**Interrupt Clock on Mode** 0 = Edge detection circuit is active only if I/O pin corresponding RHIEN (Px_INTEN[n+16])/FLIEN (Px_INTEN[n]) bit is set to 1. 1 = All I/O pins edge detection circuit is always active after reset. **Note:** It is recommended to disable this bit to save system power if no special application concern.
[4]	**DBCLKSRC**	**De-bounce Counter Clock Source Selection** 0 = De-bounce counter clock source is the HCLK. 1 = De-bounce counter clock source is the 10 kHz internal low speed RC oscillator (LIRC).

Bits	Description	
[3:0]	**DBCLKSEL**	**De-bounce Sampling Cycle Selection** 0000 = Sample interrupt input once per 1 clocks. 0001 = Sample interrupt input once per 2 clocks. 0010 = Sample interrupt input once per 4 clocks. 0011 = Sample interrupt input once per 8 clocks. 0100 = Sample interrupt input once per 16 clocks. 0101 = Sample interrupt input once per 32 clocks. 0110 = Sample interrupt input once per 64 clocks. 0111 = Sample interrupt input once per 128 clocks. 1000 = Sample interrupt input once per 256 clocks. 1001 = Sample interrupt input once per 2*256 clocks. 1010 = Sample interrupt input once per 4*256 clocks. 1011 = Sample interrupt input once per 8*256 clocks. 1100 = Sample interrupt input once per 16*256 clocks. 1101 = Sample interrupt input once per 32*256 clocks. 1110 = Sample interrupt input once per 64*256 clocks. 1111 = Sample interrupt input once per 128*256 clocks.

GPIO Px.n Pin Data Input/Outut Register (Pxn_PDIO)

Register	Offset	R/W	Description	Reset Value
PAn_PDIO n=0,1..5	GPIO_BA+0x800+(0x04 * n)	R/W	GPIO PA.n Pin Data Input/Output Register	0x0000_000X
PBn_PDIO n=0,1..4	GPIO_BA+0x840+(0x04 * n)	R/W	GPIO PB.n Pin Data Input/Output Register	0x0000_000X
PCn_PDIO n=0,1..4	GPIO_BA+0x880+(0x04 * n)	R/W	GPIO PC.n Pin Data Input/Output Register	0x0000_000X
PDn_PDIO n=0,1..6	GPIO_BA+0x8C0+(0x04 * n)	R/W	GPIO PD.n Pin Data Input/Output Register	0x0000_000X

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							**PDIO**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**PDIO**	**GPIO Px.N Pin Data Input/Output** Writing this bit can control one GPIO pin output value. 0 = Corresponding GPIO pin set to low. 1 = Corresponding GPIO pin set to high. Read this register to get GPIO pin status. For example, writing PA0_PDIO will reflect the written value to bit DOUT (Px_DOUT[0]), reading PA0_PDIO will return the value of PIN (PA_PIN[0]). **Note1:** The writing operation will not be affected by register DATMSK (Px_DATMSK[n]). **Note2:** Max. n=5 for port A. Max. n=4 for port B. Max. n=4 for port C. Max. n=6 for port D. n=0 is reserved.

6.6 Timer Controller (TIMER)

6.6.1 Overview

The Timer Controller includes two 32-bit timers, TIMER0 ~ TIMER1, allowing user to easily implement a timer control for applications. The timer can perform functions, such as frequency measurement, delay timing, clock generation, and event counting by external input pins, and interval measurement by external capture pins.

6.6.2 Features

- Supports two sets of 32-bit timers with 24-bit up-timer and one 8-bit pre-scale counter
- Supports independent clock source for each channel (TMR0_CLK, TMR1_CLK)
- Supports four timer counting modes: one-shot, periodic, toggle and continuous counting
- Time-out period = (period of timer clock input) * (8-bit pre-scale counter + 1) * (24-bit CMPDAT)
- Supports maximum counting cycle time = (1 / T MHz) * (2^8) * (2^{24}); T is the period of timer clock
- 24-bit up counter value is readable through TIMERx_CNT (Timer Data Register)
- Supports event counting function to count the event from external pin (TM0, TM1)
- Supports internal capture triggered while internal ACMP output signal transition
- Supports chip wake-up from Idle/Power-down mode if a timer interrupt signal is generated

6.6.3 Block Diagram

Each channel is equipped with an 8-bit pre-scale counter, a 24-bit up-timer, a 24-bit compare register and an interrupt request signal. Refer to Figure 6.6-1. There are five options of clock sources for each channel. Figure 6.6-2 illustrates the clock source control function.

Figure 6.6-1 Timer Controller Block Diagram

Figure 6.6-2 Clock Source of Timer Controller

6.6.4 Basic Configuration

The peripheral clock source of Tiimer0 ~ Timer1 can be enabled in TMRxCKEN (CLK_APBCLK[3:2]) and selected as different frequency in TMR0SEL (CLK_CLKSEL1[10:8]) for Timer0, TMR1SEL (CLK_CLKSEL1[14:12]) for Timer1.

6.6.5 Functional Description

The 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.

6.6.5.1 Timer Interrupt Flag

Timer controller supports two interrupt flags; one is TIF (TIMERx_INTSTS[0]) flag and its set while timer counter value CNT (TIMERx_CNT[23:0]) matches the timer compared value CMPDAT (TIMERx_CMP[23:0]), the other is CAPIF (TIMERx_EINTSTS[0]) flag and its set when the transition on the ACMPOx (ACMP_STATUS[3] and ACMP_STATUS[2]) associated CAPEDGE (TIMERx_EXTCTL[2:1]) setting.

6.6.5.2 Timer Counting Operation Mode

Timer controller provides four timer counting modes: one-shot, periodic, toggle-output and continuous counting operation modes:

6.6.5.3 One–shot Mode

If timer controller is configured at one-shot OPMODE (TIMERx_CTL[28:27] is 00) and CNTEN (TIMERx_CTL[30]) bit is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF(TIMERx_INTSTS[0]) flag will be set to 1, CNT (TIMERx_CNT[23:0]) value and CNTEN bit is cleared by timer controller then timer counting operation stops. In the meantime, if the INTEN (TIMERx_CTL[29]) bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also.

6.6.5.4 Periodic Mode

If timer controller is configured at periodic OPMODE (TIMERx_CTL[28:27] is 01) and CNTEN (TIMERX_CTL[30]) bit is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF flag will be set to 1, CNT value will be cleared by timer controller and timer counter operates counting again. In the meantime, if the INTEN bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. In this mode, timer controller operates counting and compares with CMPDAT value periodically until the CNTEN bit is cleared by software.

6.6.5.5 Toggle-Output Mode

If timer controller is configured at toggle-out OPMODE (TIMERx_CTL[28:27] is 10) and CNTEN

(TIMERX_CTL[30]) bit is set, the timer counter starts up counting. The counting operation of toggle-out mode is almost the same as periodic mode, except toggle-out mode has associated output pin to output signal while specify TIF bit is set. Thus, the toggle-output signal on output pin is changing back and forth with 50% duty cycle.

6.6.5.6 Continuous Counting Mode

If timer controller is configured at continuous counting OPMODE (TIMERx_CTL[28:27] is 11) and CNTEN bit is set, the timer counter starts up counting. Once the CNT (TIMERx_CNT[23:0]) value reaches CMPDAT (TIMERx_CMP[23:0]) value, the TIF flag will be set to 1 and CNT value keeps up counting. In the meantime, if the INTEN bit is enabled, the timer interrupt signal is generated and sent to NVIC to inform CPU also. User can change different CMPDAT value immediately without disabling timer counting and restarting timer counting in this mode.

For example, CMPDAT value is set as 80, first. The TIF flag will set to 1 when CNT value is equal to 80, timer counter is kept counting and CNT value will not goes back to 0, it continues to count 81, 82, 83,⋯ to 2^{24} -1, 0, 1, 2, 3, ⋯ to 2^{24} -1 again and again. Next, if software programs CMPDAT value as 200 and clears TIF flag, the TIF flag will set to 1 again when CNT value reaches to 200. At last, software programs CMPDAT as 500 and clears TIF flag, the TIF flag will set to 1 again when CNT value reaches to 500.

In this mode, the timer counting is continuous. Thus, this operation mode is called as continuous counting mode.

Figure 6.6-3 Continuous Counting Mode

6.6.5.7 Event Counting Mode

Timer controller also provides an application which can count the input event from TMx pin (x= 0~1) and the number of event will reflect to CNT (TIMERx_CNT[23:0]) value. It is also called as event counting function. In this function, EXTCNTEN (TIMERx_CTL[24]) bit should be set and the timer peripheral clock source should be set as HCLK.

Software can enable or disable TMx pin de-bounce circuit by ECNTDBEN (TIMERx_EXTCTL[7]) bit. The input event frequency should be less than 1/3 HCLK if TMx pin de-bounce disabled or

less than 1/8 HCLK if TMx pin de-bounce enabled to assure the returned CNT value is incorrect, and software can also select edge detection phase of TMx pin by CNTPHASE (TIMERx_EXTCTL[0]) bit.

In event counting mode, the timer counting operation mode can be selected as one-shot, periodic and continuous counting mode to counts the CNT value by input event from TMx pin.

6.6.5.8 Capture Function

The capture or reset function is provided to capture or reset timer counter value. The capture function with free-counting capture mode and trigger-counting capture mode are configured by CAPMODE (TIMERx_EXTCTL[8]). The free-counting capture mode, reset mode, trigger-counting capture mode are described as follows.

6.6.5.9 Free-Counting Capture Mode

The event capture function is used to load CNT (TIMERx_CNT[23:0]) value to CAPDAT (TIMERx_CAP[23:0]) value while edge transition detected on ACMPOx (x= 0~1). In this mode, CAPMODE (TIMERx_EXTCTL[8]) and CAPFUNCS (TIMERx_EXTCTL[4]) should be as 0 for select ACMPOx (x= 0~1) transition is using to trigger capture function and the timer peripheral clock source should be set as HCLK.

This mode can select edge transition detection of ACMPOx (x= 0~1) by setting CAPEDGE (TIMERx_EXTCTL[2:1]).

In Free-Counting capture mode, user does not consider what timer counting operation mode is selected, the capture event occurred only if edge transition on ACMPOx (x= 0~1) is detected.

Users must consider the Timer will keep register TIMERx_CAP unchanged and drop the new capture value, if the CPU does not clear the CAPIF (TIMERx_EINTSTS[0]) status. The operation method is described in Table 6.6-1.

Figure 6.6-4 Free-Counting Capture Mode

6.6.5.10 Reset Counter Mode

The timer controller also provides reset counter function to reset CNT (TIMERx_CNT[23:0]) value while edge transition detected on ACMPOx (x= 0~1). In this mode, most the settings are the same as Free-Counting capture mode except CAPFUNCS (TIMERx_EXTCTL[4]) should be as 1 for

select ACMPOx (x= 0~1) transition is using to trigger reset counter value. The operation method is also described in Table 6.6-1.

Figure 6.6-5 External Reset Counter Mode

6.6.5.11 Trigger-Counting Capture Mode

If CAPMODE (TIMERx_EXTCTL[8]) is set to 1, CAPEN (TIMERx_EXTCTL[3]) is set to 1 and CAPFUNCS (TIMERx_EXTCTL[4]) is set to 0, the CNT will be reset to 0 then captured into CAPDAT register when ACMPOx (x= 0~1) trigger condition occurred. The ACMPOx trigger edge can be chosen by CAPEDGE (TIMERx_EXTCTL[2:1]). The detailed operation method is described in Table 6.6-1. When ACMPOx (x= 0~1) trigger occurred, CAPIF (TIMERx_EINTSTS[0]) is set to 1, and the interrupt signal is generated, then sent to NVIC to inform CPU if CAPIEN (TIMERx_EXTCTL[5]) is 1.

Function	CAPMODE (TIMERx_EXTCTL[8])	CAPFUNCS TIMERx_EXTCTL[4])	CAPEDGE TIMERx_EXTCTL[2:1])	Operation Description
Free-counting Capture Mode	0	0	00	A 1 to 0 transition on ACMPOx (x= 0~1) pin is detected. CNT is captured to CAPDAT.
	0	0	01	A 0 to 1 transition on ACMPOx (x= 0~1) pin is detected. CNT is captured to CAPDAT.
	0	0	10	Either 1 to 0 or 0 to 1 transition on ACMPOx (x= 0~1) pin is detected. CNT is captured to CAPDAT.
	0	0	11	Reserved
Reset Counter Mode	0	1	00	An 1 to 0 transition on ACMPOx (x= 0~1) pin is detected. CNT is reset to 0.
	0	1	01	A 0 to 1 transition on ACMPOx (x= 0~1) pin is detected. CNT is reset to 0.
	0	1	10	Either 1 to 0 or 0 to 1 transition on ACMPOx (x= 0~1) pin is detected. CNT is reset to 0.
	0	1	11	Reserved
Trigger-Counting Capture Mode	1	0	00	Falling Edge Trigger: The 1st 1 to 0 transition on ACMPOx (x= 0~1)

				pin is detected to reset CNT as 0 and then starts counting, while the 2nd 1 to 0 transition stops counting.
	1	0	01	Rising Edge Trigger: The 1st 0 to 1 transition to on ACMPOx (x= 0~1) pin is detected to reset CNT as 0 and then starts counting, while the 2nd 0 to 1 transition stops counting.
	1	0	10	Level Change Trigger: An 1 to 0 transition on ACMPOx (x= 0~1) pin is detected to reset CNT as 0 and then starts counting, while 0 to 1 transition stops counting.
	1	0	11	Level Change Trigger: A 0 to 1 transition on ACMPOx (x= 0~1) pin is detected to reset CNT as 0 and then starts counting, while 1 to 0 transition stops counting.

Table 6.6-1 Input Capture Mode Operation

6.6.5.12 Continuous Capture Mode

Timer0/1 provide Continuous Capture mode to obtain timer counter value automatically when edges occurs on input capture signal.

When CCAPEN (TIMER_CCAPCTL[0]) set to high, the Continuous Capture mode will be enabled. CNTSEL (TIMER_CCAPCTL[3:2]) is 00, means timer0 counter value will be latched when edge of input capture signal changed. CNTSEL is 1 means timer1 counter value will be latched.

After CCAPEN enabled, if edge of input capture signal changed, the first rising edge will latch the timer counter value to TIMER_CCAP0 then set CAPR1F (TIMER_CCAPCTL[8]) to high. When CAPR1F is high, the first falling edge will latch the timer counter value to TIMER_CCAP1 and then set CAPF1F (TIMER_CCAPCTL[9]) to high. When CAPF1F is high, the second rising edge will latch timer counter value to TIMER_CCAP2 and then set CAPR2F (TIMER_CCAPCTL[10]) to high.

When CAPR2F is high, the second falling edge will latch the timer counter value to TIMER_CCAP3 and then set CAPF2F (TIMER_CCAPCTL[11]) to high.

Figure 6.6-6 Continuous Capture Mode Block

Figure 6.6-7 Continuous Capture Mode Behavior

6.6.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
TMR Base Address **TMR_BA = 0x4001_0000**				
TIMER0_CTL	TMR_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TIMER0_CMP	TMR_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TIMER0_INTSTS	TMR_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER0_CNT	TMR_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TIMER0_CAP	TMR_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TIMER0_EXTCTL	TMR_BA+0x14	R/W	Timer0 Extended Event Control Register	0x0000_0000
TIMER0_EINTSTS	TMR_BA+0x18	R/W	Timer0 Extended Event Interrupt Status Register	0x0000_0000
TIMER1_CTL	TMR_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005
TIMER1_CMP	TMR_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000
TIMER1_INTSTS	TMR_BA+0x28	R/W	Timer1 Interrupt Status Register	0x0000_0000
TIMER1_CNT	TMR_BA+0x2C	R	Timer1 Data Register	0x0000_0000
TIMER1_CAP	TMR_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000
TIMER1_EXTCTL	TMR_BA+0x34	R/W	Timer1 Extended Event Control Register	0x0000_0000
TIMER1_EINTSTS	TMR_BA+0x38	R/W	Timer1 Extended Event Interrupt Status Register	0x0000_0000
TIMER_CCAPCTL	TMR_BA+0x40	R/W	Timer Continuous Capture Control Register	0x0000_0000
TIMER_CCAP0	TMR_BA+0x44	R	Timer Continuous Capture Data Register 0	0x0000_0000
TIMER_CCAP1	TMR_BA+0x48	R	Timer Continuous Capture Data Register 1	0x0000_0000
TIMER_CCAP2	TMR_BA+0x4C	R	Timer Continuous Capture Data Register 2	0x0000_0000
TIMER_CCAP3	TMR_BA+0x50	R	Timer Continuous Capture Data Register 3	0x0000_0000

6.6.7 Register Description

Timer Control Register (TIMERx_CTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_CTL	TMR_BA+0x00	R/W	Timer0 Control and Status Register	0x0000_0005
TIMER1_CTL	TMR_BA+0x20	R/W	Timer1 Control and Status Register	0x0000_0005

31	30	29	28	27	26	25	24
ICEDEBUG	**CNTEN**	**INTEN**	**OPMODE**		**RSTCNT**	**ACTSTS**	**EXTCNTEN**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
WKEN	**Reserved**					**CMPCTL**	**CNTDATEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PSC							

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disabl Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects TIMER counting. Timer counter will be held while CPU is held by ICE. 1 = ICE debug mode acknowledgement Disabled. Timer counter will keep going no matter CPU is held by ICE or not.
[30]	**CNTEN**	**Timer Enable Bit** 0 = Stops/Suspends counting. 1 = Starts counting. **Note1:** In stop status, and then set CNTEN to 1 will enable the 24-bit up counter to keep counting from the last stop counting value. **Note2:** This bit is auto-cleared by hardware in one-shot mode (TIMERx_CTL[28:27] = 00) when the timer interrupt flag (TIF) is generated.
[29]	**INTEN**	**Interrupt Enable Bit** 0 = Timer Interrupt function Disabled. 1 = Timer Interrupt function Enabled. **Note:** If this bit is enabled, when the timer interrupt flag (TIF) is set to 1, the timer interrupt signal is generated and inform to CPU.
[28:27]	**OPMODE**	**Timer Operating Mode** 00 = The timer is operating in One-shot mode. The associated interrupt signal is generated once (if INTEN is enabled) and CNTEN is automatically cleared by hardware. 01 = The timer is operating in Periodic mode. The associated interrupt signal is generated periodically (if INTEN is enabled). 10 = The timer is operating in Toggle mode. The interrupt signal is generated periodically (if INTEN is enabled). The associated signal (tout) is changing back and forth with 50% duty cycle.

Bits	Description	
		11 = The timer is operating in Continuous Counting mode. The associated interrupt signal is generated when TIMERx_CNT = TIMERx_CMP (if INTEN is enabled). However, the 24-bit up-timer counts continuously. Please refer to 6.12.5.2 for detailed description about Continuous Counting mode operation.
[26]	**RSTCNT**	**Timer Reset** 0 = No effect. 1 = Reset 8-bit PSC counter, 24-bit up counter value and CNTEN bit if ACTSTS is 1.
[25]	**ACTSTS**	**Timer Active Status (Read Only)** This bit indicates the 24-bit up counter status. 0 = 24-bit up counter is not active. 1 = 24-bit up counter is active.
[24]	**EXTCNTEN**	**Counter Mode Enable Bit** 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 section “Event Counting Mode” for detailed description. 0 = External event counter mode Disabled. 1 = External event counter mode Enabled.
[23]	**WKEN**	**Wake-up Enable Bit** When WKEN is set and the TIF or CAPIF is set, the timer controller will generator a wake-up trigger event to CPU. 0 = Wake-up trigger event Disabled. 1 = Wake-up trigger event Enabled.
[22:18]	**Reserved**	Reserved.
[17]	**CMPCTL**	**TIMERx_CMP Mode Control** 0 = In One-shot or Periodic mode, when writing new CMPDAT, the timer counter will reset. 1 = In One-shot or Periodic mode, when write new CMPDAT if new CMPDAT > CNT (TIMERx_CNT[23:0])(current counter), the timer counter keeps counting and will not reset. If new CMPDAT <= CNT(current counter), timer counter will be reset.
[16]	**CNTDATEN**	**Data Load Enable Bit** When CNTDATEN is set, CNT (TIMERx_CNT[23:0]) (Timer Data Register) will be updated continuously with the 24-bit up-timer value as the timer is counting. 0 = Timer Data Register update Disabled. 1 = Timer Data Register update Enabled while Timer counter is active.
[15:8]	**Reserved**	Reserved.
[7:0]	**PSC**	**Prescale Counter** Timer input clock source is divided by (PSC+1) before it is fed to the Timer up counter. If this field is 0 (PSC = 0), then there is no scaling.

Timer Compare Register (TIMERx_CMP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CMP	TMR_BA+0x04	R/W	Timer0 Compare Register	0x0000_0000
TIMER1_CMP	TMR_BA+0x24	R/W	Timer1 Compare Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CMPDAT**	**Timer Compared Value** CMPDAT is a 24-bit compared value register. When the internal 24-bit up counter value is equal to CMPDAT value, the TIF flag will set to 1. Time-out period = (Period of Timer clock source) * (8-bit PSC + 1) * (24-bit CMPDAT). **Note1:** Never write 0x0 or 0x1 in CMPDAT field, or the core will run into unknown state. **Note2:** 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 CMPDAT field. But if Timer is operating at other modes except Periodic mode on M05xxDN/DE, the 24-bit up counter will restart counting and using newest CMPDAT value to be the timer compared value if software writes a new value into CMPDAT field.

Timer Interrupt Status Register (TIMERx_INTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_INTSTS	TMR_BA+0x08	R/W	Timer0 Interrupt Status Register	0x0000_0000
TIMER1_INTSTS	TMR_BA+0x28	R/W	Timer1 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						**TWKF**	**TIF**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**TWKF**	**Timer Wake-up Flag** This bit indicates the interrupt wake-up flag status of Timer. 0 = Timer does not cause chip wake-up. 1 = Chip wake-up from Idle or Power-down mode if Timer time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[0]	**TIF**	**Timer Interrupt Flag** This bit indicates the interrupt flag status of Timer while CNT (TIMERx_CNT[23:0]) value reaches to CMPDAT value. 0 = No effect. 1 = CNT value matches the CMPDAT value. **Note:** This bit is cleared by writing 1 to it.

Timer Data Register (TIMERx_CNT)

Register	Offset	R/W	Description	Reset Value
TIMER0_CNT	TMR_BA+0x0C	R	Timer0 Data Register	0x0000_0000
TIMER1_CNT	TMR_BA+0x2C	R	Timer1 Data Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Timer Data Register** If CNTDATEN is set to 1, CNT register value will be updated continuously to monitor 24-bit up counter value.

Timer Capture Data Register (TIMERx_CAP)

Register	Offset	R/W	Description	Reset Value
TIMER0_CAP	TMR_BA+0x10	R	Timer0 Capture Data Register	0x0000_0000
TIMER1_CAP	TMR_BA+0x30	R	Timer1 Capture Data Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CAPDAT**	**Timer Capture Data Register** When CAPEN (TIMERx_EXTCTL[3]) bit is set, CAPFUNCS (TIMERx_EXTCTL[4]) bit is 0, and a transition on ACMPOx matched the CAPEDGE (TIMERx_EXTCTL[2:1]) setting, CAPIF (TIMERx_EINTSTS[0]) will set to 1 and the current timer counter value CNT (TIMERx_CNT[23:0]) will be auto-loaded into this CAPDAT field.

Timer Extended Event Control Register (TIMERx_EXTCTL)

Register	Offset	R/W	Description	Reset Value
TIMER0_EXTCTL	TMR_BA+0x14	R/W	Timer0 Extended Event Control Register	0x0000_0000
TIMER1_EXTCTL	TMR_BA+0x34	R/W	Timer1 Extended Event 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							**CAPMODE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ECNTDBEN	**Reserved**	**CAPIEN**	**CAPFUNCS**	**CAPEN**	**CAPEDGE**		**CNTPHASE**

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8]	**CAPMODE**	**Capture Mode Select Bit** 0 = Timer counter reset function or free-counting mode of timer capture function. 1 = Trigger-counting mode of timer capture function.
[7]	**ECNTDBEN**	**Timer Counter Input Pin De-bounce Enable Bit** 0 = TMx (x = 0~1) pin de-bounce Disabled. 1 = TMx (x = 0~1) pin de-bounce Enabled. If this bit is enabled, the edge detection of TMx (x = 0~1) pin is detected with de-bounce circuit.
[6]	**Reserved**	Reserved.
[5]	**CAPIEN**	**Timer Capture Interrupt Enable Bit** 0 = Timer Capture Interrupt Disabled. 1 = Timer Capture Interrupt Enabled. **Note:** CAPIEN is used to enable timer capture interrupt. If CAPIEN enabled, timer will generate an interrupt when CAPIF (TIMERx_EINTSTS[0]) is 1. For example, while CAPIEN = 1, CAPEN = 1, and CAPEDGE = 00, an 1 to 0 transition on the ACMPOx will cause the CAPIF to be set then the interrupt signal is generated and sent to NVIC to inform CPU.
[4]	**CAPFUNCS**	**Capture Function Select Bit** 0 = Capture Mode Enabled. 1 = Reset Mode Enabled. **Note1:** When CAPFUNCS is 0, transition on ACMPOx is using to save the 24-bit timer counter value to CAPDAT register. **Note2:** When CAPFUNCS is 1, transition on ACMPOx is using to reset the 24-bit timer counter value.

Bits	Description	
[3]	**CAPEN**	**Timer Capture Function Enable Bit** This bit enables the Timer Capture Function 0 = Timer Capture Function Disabled. 1 = Timer Capture Function Enabled.
[2:1]	**CAPEDGE**	**Timer Capture Pin Edge Detection** 00 = A falling edge on ACMPOx will be detected. 01 = A rising edge on ACMPOx will be detected. 10 = Either rising or falling edge on ACMPOx will be detected. 11 = Reserved.
[0]	**CNTPHASE**	**Timer External Count Pin Phase Detect Selection** This bit indicates the detection phase of TMx (x = 0~1) pin. 0 = A falling edge of TMx (x = 0~1) pin will be counted. 1 = A rising edge of TMx (x = 0~1) pin will be counted.

Timer Extended Event Interrupt Status Register (TIMERx_EINTSTS)

Register	Offset	R/W	Description	Reset Value
TIMER0_EINTSTS	TMR_BA+0x18	R/W	Timer0 Extended Event Interrupt Status Register	0x0000_0000
TIMER1_EINTSTS	TMR_BA+0x38	R/W	Timer1 Extended Event 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							**CAPIF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**CAPIF**	**Timer Capture Interrupt Flag** This bit indicates the timer external capture interrupt flag status. 0 = Timer Cpautre interrupt did not occur. 1 = Timer Capture interrupt occurred. **Note1:** This bit is cleared by writing 1 to it. **Note2:** When CAPEN (TIMERx_EXTCTL[3]) bit is set, CAPFUNCS (TIMERx_EXTCTL[4]) bit is 0, and a transition on ACMPOx matched the CAPEDGE (TIMERx_EXTCTL[2:1]) setting, this bit will set to 1 by hardware. **Note3:** There is a new incoming capture event detected before CPU clearing the CAPIF status. If the above condition occurred, the Timer will keep register TIMERx_CAP unchanged and drop the new capture value.

Timer Continuous Capture Control Register (TIMER_CCAPCTL)

Register	Offset	R/W	Description	Reset Value
TIMER_CCAPCTL	TMR_BA+0x40	R/W	Timer Continuous Capture Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**CCAPIEN**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CAPF2F**	**CAPR2F**	**CAPF1F**	**CAPR1F**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**CAPCHSEL**	**CNTSEL**		**INV**	**CCAPEN**

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**CCAPIEN**	**Capture Interrupt Enable Bit** 00 = Interrupt Disabled. 01 = Capture Rising Edge 1 and Falling Edge 1 interrupt Enabled. 10 = Capture Rising Edge 1, Falling dege1 and Rising Edge 2 interrupt Enabled. 11 = Capture Rising Edge 1, Falling dege1, Rising Edge 2 and Falling Edge 2 interrupt Enabled.
[15:12]	**Reserved**	Reserved.
[11]	**CAPF2F**	**Capture Falling Edge 2 Flag** Second falling edge already captured, this bit will be set to 1 0 = None. 1 = CAPDAT(TIMER_CCAP3[23:0]) data is ready for read. **Note:** This bit is cleared by hardware automatically when writing 1 to this bit.
[10]	**CAPR2F**	**Capture Rising Edge 2 Flag** Second rising edge already captured, this bit will be set to 1. 0 = None. 1 = CAPDAT(TIMER_CCAP2[23:0]) data is ready for read. **Note:** This bit is cleared by hardware automatically when writing 1 to this bit.
[9]	**CAPF1F**	**Capture Falling Edge 1 Flag** First falling edge already captured, this bit will be set to 1. 0 = None. 1 = CAPDAT(TIMER_CCAP1[23:0]) data is ready for read. **Note:** This bit is cleared by hardware automatically when writing 1 to this bit.

Bits	Description	
[8]	**CAPR1F**	**Capture Rising Edge 1 Flag** First rising edge already captured, this bit will be set to 1. 0 = None. 1 = CAPDAT(TIMER_CCAP0[23:0]) data is ready for read. **Note:** This bit is cleared by hardware automatically when writing 1 to this bit.
[7:5]	**Reserved**	Reserved.
[4]	**CAPCHSEL**	**Capture Timer Channel Selection** Select the channel to be the continuous capture event. 0 = PD.2. 1 = PC.2.
[3:2]	**CNTSEL**	**Capture Timer Selection** Select the timer to continuous capture the input signal. 00 = TIMER0. 01 = TIMER1. 10 = SysTick. 11 = Reserved.
[1]	**INV**	**Input Signal Inverse** Invert the input signal which be captured. 0 = None. 1 = Inverse.
[0]	**CCAPEN**	**Continuous Capture Enable Bit** This bit is to be enabled the continuous capture function. 0 = Continuous capture function Disabled. 1 = Continuous capture function Enabled. **Note:** This bit is cleared by hardware automatically when capture operation finish or writing 0 to it

Timer Continuous Capture Register 0-3 (TIMER_CCAP0-3)

Register	Offset	R/W	Description	Reset Value
TIMER_CCAP0	TMR_BA+0x44	R	Timer Continuous Capture Data Register 0	0x0000_0000
TIMER_CCAP1	TMR_BA+0x48	R	Timer Continuous Capture Data Register 1	0x0000_0000
TIMER_CCAP2	TMR_BA+0x4C	R	Timer Continuous Capture Data Register 2	0x0000_0000
TIMER_CCAP3	TMR_BA+0x50	R	Timer Continuous Capture Data Register 3	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CAPDAT**	**Timer Continuous Capture Data Register** TIMER_CCAP0 store the timer count value of first rising edge TIMER_CCAP1 store the timer count value of first falling edge TIMER_CCAP2 store the timer count value of second rising edge TIMER_CCAP3 store the timer count value of second falling edge

6.7 Enhanced Input Capture Timer (ECAP)

6.7.1 Overview

This device provides an Input Capture Timer/Counter whose capture function can detect the digital edge-changed signal at channel inputs. This unit has three input capture channels. The timer/counter is equipped with up counting, reload and compare-match capabilities.

6.7.2 Features

- 24-bit Input Capture up-counting timer/counter.
- 3 input channels thatl has its own capture counter hold register.
- Noise filter in front end of input ports.
- Edge detector with three options.
  - Rising edge detection.
  - Falling edge detection.
  - Both edge detection.
- Supports ADC compare output and ACMP output as input sources
- Captured events reset and/or reload capture counter.
- Supports compare-match function.
- Supports interrupt function.

6.7.3 Block Diagram

Figure 6.7-1 Enhanced Input Capture Timer/Counter Architecture

Figure 6.7-2 Enhanced Input Capture Timer/Counter Clock Source Control

Figure 6.7-1 illustrates the architecture of the Input Capture. This input capture timer/counter unit supports 3 input channels with programmable input signal sources. The port pins ECAP_P0 to ECAP_P2 can be fed to the inputs of capture unit through noise filter or bypass it (CAPNFDIS = 1). Besides, the analog comparator outputs (ACMPn_ECAP), and ADC compare output (ADC_ECAP) also can be internally routed to the capture inputs by setting the register ECAP_CTL0(CAPSEL0~ CAPSEL 2).

6.7.4 Input Noise Filter

Figure 6.7-3 shows the architecture of Noise-Filter with four sampling rate options.

Figure 6.7-3 Noise Filter Sampling Clock Selection

If enabled, the capture logic is required to sample 4 consecutive same capture input value to recognize an edge as a capture event. A possible implementation of digital noise filter is as Figure 6.7-4.

Figure 6.7-4 Noise Filter

6.7.5 Operation of Input Capture Timer/Counter

The Input Capture Timer/Counter unit consists of 2 main functional blocks, Capture block and Operation block. There are 3 Input Capture units in Capture block for 3 input channel.

The capture units function as detecting and measuring the pulse width and the period of a square wave. The input channel 0 to 2 have their own edge detectors, which are in Input Capture block but share with one capture timer/counter, ECAP_CNT, which is in Operation block. The edge trigger option is programmable through CAPEDG (ECAP_CTL1[5,4], [3,2], [1,0]) register supporting positive edge, negative edge and both edge triggers. Each capture unit consists of an enable control bit, IC0EN ~ IC2EN (ECAP_CTL0[6:4]) to enable/disable each input channel and a status bit CAP0 ~ CAP2 (ECAP_STATUS[10:8]) to let software monitor the current status of each channel.

The Input Capture supports reload mode and compare mode. For both mode, the capture counter (ECAP_CNT) serves as a 24-bit up-counting counter whose clock comes from the output of the clock divider and is gated with CPTST, and the clock source of the clock divider, which can be set by CAPDIV[2:0] to divide clock by 1,4,16,32,64,96,112 and 128, is programmable (by setting CNTSRC[1:0]) to be from system clock source, ECAP_CLK or input channel CAP0 ~ CAP2. In reload mode, ECAP_CNTCMP serves as a reload register while in compare mode ECAP_CNTCMP serves as a compare register. The Input Capture Timer/Counter Enable bit (CAPEN) must be set to enable Input Capture Timer/Counter functions. More details of operation are described in the following.

6.7.5.1 Capture Function

When the capture input detects a valid edge change, it triggers a valid capture event (CAPTE0~2) so that the content of the free running 24-bit capture counter ECAP_CNT will be captured/transferred into the capture hold registers, ECAP_HLD0~2 depending on which channel is triggered. This event also causes the corresponding flag CAPTFx(ECAP_STS[2:0]) to be set, which will generate an interrupt if the corresponding interrupt enable bit CAPTFxIEN (ECAP_CTL0[18:16]) is set. Triggered Flags are set by hardware and should be cleared by software. Software can read the register ECAP_STS to get the status of flags and has to write 1 to the corresponding bit(s) of ECAP_STS to clear flag(s).

In addition, setting the CPTCLR(ECAP_CTL0[26]) will allow hardware to reset capture counter (ECAP_CNT) automatically whenever the event happens.

Figure 6.7-5 Enhanced Input Capture Timer/Counter Functions Block

6.7.5.2 Compare Mode

The compare function is enabled by setting the CMPEN (ECAP_CTL0[28]) bit to 1, and ECAP_CNTCMP will serve as a compare register. As ECAP_CNT counting up, upon matching ECAP_CNTCMP value, the flag, CAPCMPF (ECAP_STS[4]), will be set, which will generate an interrupt, CMP_INT, if compare interrupt enable bit, CAPCMPIEN (ECAP_CTL0[21]), is set.

Besides, setting the CMPCLR (ECAP_CTL0[25]) will allow hardware to make capture counter cleared to zero automatically after a compare-match event occurs.

6.7.5.3 Reload Mode

The Input Capture Timer/Counter can also be configured for reload mode. The reload function is enabled by setting RLDEN (ECAP_CTL0[27]) to 1.

In this mode, ECAP_CNTCMP serves as a reload register. A reload event is generated and causes the content of the ECAP_CNTCMP register to be loaded into the ECAP_CNT register when ECAP_CNT overflows, CAPTF2, CAPTF1 or CAPTF0 setting by configuring the CPRLDS (ECAP_CTL1[10:8]).

One thing should be noted is that if CPTCLR as well as RLDEN are set, when a valid trigger event (CAPTFx) occurs, only RELAOD function will be executed.

6.7.6 Input Capture Timer/Counter Interrupt Architecture

Figure 6.7-6 demonstrates the architecture of Input Capture Timer/Counter interrupt module. There are 5 interrupt sources (OVF_INT, CMP_INT, CAPTF0_INT~CAPTF2_INT), which are logical 'OR' together, in a input capture unit, and each one has an interrupt flag (CAPOVF, CAPCMPF, CAPTF0~CAPTF2), which can trigger Interrupt (ECAP_INT), as well as an the enable control bit (CAPOVIEN, CAPCMPIEN, CAPTF0IEN ~ CAPTF2IEN) to enable/disable the flag.

Note that all the interrupt flags are set by hardware and must be cleared by software by writing 1 to the bit (ECAP_STS[5:4],[2:0]) corresponding to the flag.

Figure 6.7-6 Enhanced Input Capture Timer/Counter Interrupt Architecture Diagram

6.7.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
ECAP Base Address: **ECAP_BA = 0x401B_0000**				
ECAP_CNT	ECAP_BA+0x00	R/W	Input Capture Counter	0x0000_0000
ECAP_HLD0	ECAP_BA+0x04	R/W	Input Capture Counter Hold Register 0	0x0000_0000
ECAP_HLD1	ECAP_BA+0x08	R/W	Input Capture Counter Hold Register 1	0x0000_0000
ECAP_HLD2	ECAP_BA+0x0C	R/W	Input Capture Counter Hold Register 2	0x0000_0000
ECAP_CNTCMP	ECAP_BA+0x10	R/W	Input Capture Counter Compare Register	0x0000_0000
ECAP_CTL0	ECAP_BA+0x14	R/W	Input Capture Control Register 0	0x0000_0000
ECAP_CTL1	ECAP_BA+0x18	R/W	Input Capture Control Register 1	0x0000_0000
ECAP_STS	ECAP_BA+0x1C	R/W	Input Capture Status Register	0x0000_0000

6.7.8 Register Description

Input Capture Counter (ECAP_CNT)

Register	Offset	R/W	Description	Reset Value
ECAP_CNT	ECAP_BA+0x00	R/W	Input Capture Counter	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CNT**	**Input Capture Timer/Counter** The input Capture Timer/Counter is a 24-bit up-counting counter. The clock source for the counter is from thme clock divider.

Input Capture Counter Hold Register 0-2 (ECAP_HLD0-2)

Register	Offset	R/W	Description	Reset Value
ECAP_HLD0	ECAP_BA+0x04	R/W	Input Capture Counter Hold Register 0	0x0000_0000
ECAP_HLD1	ECAP_BA+0x08	R/W	Input Capture Counter Hold Register 1	0x0000_0000
ECAP_HLD2	ECAP_BA+0x0C	R/W	Input Capture Counter Hold Register 2	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**HOLD**	**Input Capture Counter Hold Register** When an active input capture channel detects a valid edge signal change, the ECAP_CNT value is latched into the corresponding holding register. Each input channel has itself holding register named by ECAP_HLDx where x is from 0 to 2 to indicate inputs from CAP0 to CAP2, respectively.

Input Capture Counter Compare Register (ECAP_CNTCMP)

Register	Offset	R/W	Description	Reset Value
ECAP_CNTCMP	ECAP_BA+0x10	R/W	Input Capture Counter Compare Register	0x0000_0000

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

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:0]	**CNTCMP**	**Input Capture Counter Compare Register** If the compare function is enabled (CMPEN = 1), t this register (ECAP_CNTCMP) is used to compare with the capture counter (ECAP_CNT). If the reload control is enabled (RLDEN = 1), an overflow event or capture events will trigger the hardware to load the value of this register (ECAP_CNTCMP) into ECAP_CNT.

Input Capture Timer/Counter Control Register (ECAP_CTL0)

Register	Offset	R/W	Description	Reset Value
ECAP_CTL0	ECAP_BA+0x14	R/W	Input Capture Control Register 0	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**CAPPHGEN**	**CAPEN**	**CMPEN**	**RLDEN**	**CPTCLR**	**CMPCLR**	**CPTST**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved		**CAPCMPIEN**	**CAPOVIEN**	**Reserved**	**CAPTF2IEN**	**CAPTF1IEN**	**CAPTF0IEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CAPSEL2**		**CAPSEL1**		**CAPSEL0**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**IC2EN**	**IC1EN**	**IC0EN**	**CAPNFDIS**	**Reserved**	**NFCLKS**	

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**CAPPHGEN**	**Input Capture Flag Trigger PWM Phase Change Function Enable Bit** 0 = CAPTF2, CAPTF1 or CAPTF0 trigger PWM phase change function Disabled. 1 = CAPTF2, CAPTF1 or CAPTF0 trigger PWM phase change function Enabled.
[29]	**CAPEN**	**Input Capture Timer/Counter Enable Bit** 0 = Input Capture function Disabled. 1 = Input Capture function Enabled.
[28]	**CMPEN**	**The Compare Function Enable Bit** The compare function in input capture timer/counter is to compare the dynamic counting ECAP_CNT with the compare register ECAP_CNTCMP, if ECAP_CNT value reaches ECAP_CNTCMP, the flag CAPCMPF will be set. 0 = Compare function Disabled. 1 = Compare function Enabled.
[27]	**RLDEN**	**The Reload Function Enable Bit** 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 ECAP_CNTCMP into ECAP_CNT. 0 =Reload function Disabled. 1 = Reload function Enabled.
[26]	**CPTCLR**	**Input Capture Counter Clear by Capture Events Control** If this bit is set to 1, the capture counter (ECAP_CNT) will be cleared to 0 when any one of capture events (CAPTF0~3) occurs. 0 = Capture events (CAPTF0~3) can clear capture counter (ECAP_CNT) Disabled. 1 = Capture events (CAPTF0~3) can clear capture counter (ECAP_CNT) Enabled.

Bits	Description	
[25]	**CMPCLR**	**Input Capture Counter Clear by Compare-match Control** If this bit is set to 1, the capture counter (ECAP_CNT) will be cleared to 0 when the compare-match event (CAPCMPF = 1) occurs. 0 = Compare-match event (CAPCMPF) can clear capture counter (ECAP_CNT) Disabled. 1 = Compare-match event (CAPCMPF) can clear capture counter (ECAP_CNT) Enabled.
[24]	**CPTST**	**Input Capture Counter Start Bit** Setting this bit to 1, the capture counter (ECAP_CNT) starts up-counting synchronously with capture clock input (CAP_CLK). 0 = ECAP_CNT stop counting. 1 = ECAP_CNT starts up-counting.
[23:22]	**Reserved**	Reserved.
[21]	**CAPCMPIEN**	**Enable CAPCMPF Trigger Input Capture Interrupt** 0 = Disabling flag CAPCMPF can trigger Input Capture interrupt. 1 = Enabling flag CAPCMPF can trigger Input Capture interrupt.
[20]	**CAPOVIEN**	**Enable CAPOVF Trigger Input Capture Interrupt** 0 = Disabling flag CAPOVF can trigger Input Capture interrupt. 1 = Enabling flag CAPOVF can trigger Input Capture interrupt.
[19]	**Reserved**	Reserved.
[18]	**CAPTF2IEN**	**Enable Input Capture Channel 2 Interrupt** 0 = Disabling flag CAPTF2 can trigger Input Capture interrupt. 1 = Enabling flag CAPTF2 can trigger Input Capture interrupt.
[17]	**CAPTF1IEN**	**Enable Input Capture Channel 1 Interrupt** 0 = Disabling flag CAPTF1 can trigger Input Capture interrupt. 1 = Enabling flag CAPTF1 can trigger Input Capture interrupt.
[16]	**CAPTF0IEN**	**Enable Input Capture Channel 0 Interrupt** 0 = Disabling flag CAPTF0 can trigger Input Capture interrupt. 1 = Enabling flag CAPTF0 can trigger Input Capture interrupt.
[15:14]	**Reserved**	Reserved.
[13:12]	**CAPSEL2**	**CAP2 Input Source Selection** 00 = CAP2 input is from port pin ECAP_P2. 01 = CAP2 input is from signal ACMP0_O (Analog comparator 0 output). 10 = CAP2 input is from signal ACMP1_O (Analog comparator 1 output). 11 = CAP2 input is from signal ADC_CPR (ADC compare output).
[11:10]	**CAPSEL1**	**CAP1 Input Source Selection** 00 = CAP1 input is from port pin ECAP_P1. 01 = CAP1 input is from signal ACMP0_O (Analog comparator 0 output). 10 = CAP1 input is from signal ACMP1_O (Analog comparator 1 output). 11 = CAP1 input is from signal ADC_CPR (ADC compare output).

Bits	Description	
[9:8]	**CAPSEL0**	**CAP0 Input Source Selection** 00 = CAP0 input is from port pin ECAP_P0. 01 = CAP0 input is from signal ACMP0_O (Analog comparator 0 output). 10 = CAP0 input is from signal ACMP1_O (Analog comparator 1 output). 11 = CAP0 input is from signal ADC_CPR (ADC compare output).
[7]	**Reserved**	Reserved.
[6]	**IC2EN**	**Enable Port Pin IC2 Input to Input Capture Unit** 0 = IC2 input to Input Capture Unit Disabled. 1 = IC2 input to Input Capture Unit Enabled.
[5]	**IC1EN**	**Enable Port Pin IC1 Input to Input Capture Unit** 0 = IC1 input to Input Capture Unit Disabled. 1 = IC1 input to Input Capture Unit Enabled.
[4]	**IC0EN**	**Enable Port Pin IC0 Input to Input Capture Unit** 0 = IC0 input to Input Capture Unit Disabled. 1 = IC0 input to Input Capture Unit Enabled.
[3]	**CAPNFDIS**	**Disable Input Capture Noise Filter** 0 = Noise filter of Input Capture Enabled. 1 = The noise filter of Input Capture Disabled.
[2]	**Reserved**	Reserved.
[1:0]	**NFCLKS**	**Noise Filter Clock Pre-divided Selection** To determine the sampling frequency of the Noise Filter clock 00 = ECAP_CLK. 01 = ECAP_CLK / 2. 10 = ECAP_CLK / 4. 11 = ECAP_CLK / 16.

Input Capture Timer/Counter Control Register (ECAP_CTL1)

Register	Offset	R/W	Description	Reset Value
ECAP_CTL1	ECAP_BA+0x18	R/W	Input Capture Control Register 1	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**CNTSRC**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**CAPDIV**			**Reserved**	**CPRLDS**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CAPEDG2**		**CAPEDG1**		**CAPEDG0**	

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17:16]	**CNTSRC**	**Capture Timer/Counter Clock Source Selection** Select the capture timer/counter clock source 00 = ECAP_CLK (Default). 01 = CAP0. 10 = CAP1. 11 = CAP2.
[15]	**Reserved**	Reserved.
[14:12]	**CAPDIV**	**Capture Timer Clock Divide Selection** The capture timer clock has a pre-divider with four divided options controlled by CAPDIV[2:0]. 000 = CAPCLK / 1. 001 = CAPCLK / 4. 010 = CAPCLK / 16. 011 = CAPCLK / 32. 100 = CAPCLK / 64. 101 = CAPCLK / 96. 110 = CAPCLK / 112. 111 = CAPCLK / 128.
[11]	**Reserved**	Reserved.

Bits	Description	
[10:8]	**CPRLDS**	**ECAP_CNT Reload Trigger Source Selection** If the reload function is enabled (RLDEN = 1), when a reload trigger event comes, the ECAP_CNT is reloaded with ECAP_CNTCMP. CPRLDS[2:0] determines the ECAP_CNT reload trigger source 000 = CAPTF0. 001 = CAPTF1. 010 = CAPTF2. 100 = CAPOVF. Other = Reserved.
[7:6]	**Reserved**	Reserved.
[5:4]	**CAPEDG2**	**Channel 2 Captured Edge Selection** Input capture can detect falling edge change only, rising edge change only or one of both edge change 00 = Detect rising edge. 01 = Detect falling edge. 1x = Detect either rising or falling edge.
[3:2]	**CAPEDG1**	**Channel 1 Captured Edge Selection** Input capture can detect falling edge change only, rising edge change only or one of both edge change 00 = Detect rising edge. 01 = Detect falling edge. 1x = Detect either rising or falling edge.
[1:0]	**CAPEDG0**	**Channel 0 Captured Edge Selection** Input capture can detect falling edge change only, rising edge change only or one of both edge change 00 = Detect rising edge. 01 = Detect falling edge. 1x = Detect either rising or falling edge.

Input Capture Timer/Counter Status Register (ECAP_STS)

Register	Offset	R/W	Description	Reset Value
ECAP_STS	ECAP_BA+0x1C	R/W	Input Capture 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					**CAP2**	**CAP1**	**CAP0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CAPOVF**	**CAPCMPF**	**Reserved**	**CAPTF2**	**CAPTF1**	**CAPTF0**

Bits	Description	
[31:11]	**Reserved**	Reserved.
[10]	**CAP2**	**Input Capture Pin 2 Status (Read Only)** Input capture pin 2 (ECAP_P2) status. (The bit is read only and write is ignored)
[9]	**CAP1**	**Input Capture Pin 1 Status (Read Only)** Input capture pin 1 (ECAP_P1) status (The bit is read only and write is ignored)
[8]	**CAP0**	**Input Capture Pin 0 Status (Read Only)** Input capture pin 0 (ECAP_P0) status (The bit is read only and write is ignored)
[7:6]	**Reserved**	Reserved.
[5]	**CAPOVF**	**Input Capture Counter Overflow Flag** Flag is set by hardware when counter (ECAP_CNT) overflows from 0x00FF_FFFF to zero. 0 = No overflow event has occurred since last clear. 1 = Overflow event(s) has/have occurred since last clear. **Note:** This bit is only cleared by writing 1 to it.
[4]	**CAPCMPF**	**Input Capture Compare-match Flag** If the input capture compare function is enabled, the flag is set by hardware when capture counter (ECAP_CNT) up counts and reaches the ECAP_CNTCMP value. 0 = ECAP_CNT has not matched ECAP_CNTCMP value since last clear. 1 = ECAP_CNT has matched ECAP_CNTCMP value at least once since last clear. **Note:** This bit is only cleared by writing 1 to it.
[3]	**Reserved**	Reserved.

Bits	Description	
[2]	**CAPTF2**	**Input Capture Channel 2 Captured Flag** When the input capture channel 2 detects a valid edge change at CAP2 input, it will set flag CAPTF2 to high. 0 = No valid edge change has been detected at CAP2 input since last clear. 1 = At least a valid edge change has been detected at CAP2 input since last clear. **Note:** This bit is only cleared by writing 1 to it.
[1]	**CAPTF1**	**Input Capture Channel 1 Captured Flag** When the input capture channel 1 detects a valid edge change at CAP1 input, it will set flag CAPTF1 to high. 0 = No valid edge change has been detected at CAP1 input since last clear. 1 = At least a valid edge change has been detected at CAP1 input since last clear. **Note:** This bit is only cleared by writing 1 to it.
[0]	**CAPTF0**	**Input Capture Channel 0 Captured Flag** When the input capture channel 0 detects a valid edge change at CAP0 input, it will set flag CAPTF0 to high. 0 = No valid edge change has been detected at CAP0 input since last clear. 1 = At least a valid edge change has been detected at CAP0 input since last clear. **Note:** This bit is only cleared by writing 1 to it.

6.8 Enhanced PWM Generator (EPWM)

6.8.1 Overview

The NDA102 series has built in one PWM unit which is specially designed for motor driving control applications. The PWM unit supports six PWM generators which can be configured as six independent PWM outputs, PWM0~PWM5, or as three complementary PWM pairs, (PWM0, PWM1), (PWM2, PWM3) and (PWM4, PWM5) with three programmable dead-zone generators.

Every complementary PWM pairs share one clock divider providing nine divided frequencies (1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256) for each channel. Each PWM output shares one 16-bit counter for PWM period control, and 16-bit comparators for PWM duty control. The six PWM generators provide fourteen independent PWM interrupt flags which are set by hardware when the corresponding PWM period counter comparison matched period and duty. Each PWM interrupt source with its corresponding enable bit can 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.

To prevent PWM driving output pin with unsteady waveform, the 16-bit period up counter and 16-bit comparator are implemented with double buffer. When user writes data to counter/comparator buffer registers, the updated value will be loaded into the 16-bit counter/comparator at the end of current period. The double buffering feature avoids glitch at PWM outputs.

Besides PWM, Motor controlling also need Timer, ACMP and ADC to work together. To control motor more precisely, some registers are provided to configure not only PWM but also Timer, ADC and ACMP. By doing so, it can save more CPU time and control motor with ease especially in BLDC.

6.8.2 Features

- Supports one PWM clock timer and one 9 level Divider (1, 1/2, 1/4, 1/8, 1/16, 1/32, 1/64, 1/128, 1/256).
- Supports six independent 16-bit PWM duty control units with maximum six port pins:
  - Six independent PWM outputs – PWM0, PWM1, PWM2, PWM3, PWM4, and PWM5
  - Three complementary PWM pairs, with each pin in a pair mutually complement to each other and capable of programmable dead-zone insertion – (PWM0, PWM1), (PWM2, PWM3) and (PWM4, PWM5)
  - Three synchronous PWM pairs, with each pin in a pair in-phase – (PWM0, PWM1), (PWM2, PWM3) and (PWM4, PWM5)
- Supports group function.
- Supports one-shot (only edge alignment mode) or auto-reload mode PWM
- Supports 16-bit resolution PWM counter
- Supports Edge-aligned and Center-aligned mode
- Supports Programmable dead-zone insertion between complementary paired PWMs
- Supports hardware fault brake protections
  - Two Interrupt source types:
    - one type is brake directed, and one type can resume from brake.

- fault brake source:
  - BRK0: ACMP0, ACMP1, EADC and External pin (BRAKE).
  - BRK1: ACMP0, ACMP1, EADC and External pin (BRAKE).

- The PWM signals before polarity control stage are defined in the view of positive logic. The PWM ports is active high or active low are controlled by polarity control register.
- Supports independently falling CMPDAT matching, central matching (in Center-aligned mode), rising CMPDAT matching (in Center-aligned mode), period matching to trigger EADC conversion
- Supports ACMP output event trigger PWM to force PWM output at most one period low, this feature is usually for step motor control
- Supports interrupt accumulation function

6.8.3 Block Diagram

Figure 6.8-1 shows the PWM clock source.

Figure 6.8-1 EPWM Clock Source

The overall functioning of the EPWM module is shown in Figure 6.8-2.

Figure 6.8-2 EPWM Block Diagram

Figure 6.8-3 illustrates the architecture of PWM in pair (e.g. PWM-Timer 0/1 are in one pair and PWM-Timer 2/3 are in another one, and PWM-Timer 4/5 are in one pair.).

Figure 6.8-3 EPWM Generator 0/1 Architecture Diagram

Figure 6.8-4 EPWM Generator 2/3 Architecture Diagram

Figure 6.8-5 EPWM Generator 4/5 Architecture Diagram

6.8.4 Basic Configuration

The PWM pin functions are configured in SYS_GPA_MFP registers.

The PWM clock can be enabled in CLK_APBCLK[20]. The PWM clock source is HCLK.

6.8.5 Functional Description

6.8.5.1 PWM-Timer Operation

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

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

<u>Edge aligned (Down Counter)</u>

Duty ratio = CMPDAT / (PERIOD+1)

Duty = CMPDAT * (clock period)

Period = (PERIOD+1) * (clock period)

<u>Center aligned (Down and Up Counter):</u>

Duty ratio = 2 * (CMPDAT / (PERIOD+1))

Duty = 2 * CMPDAT * (clock period)

Period = 2 * (PERIOD+1) * (clock period)

<u>Edge aligned PWM Type (Down Counter)</u>

In Edge-aligned PWM Output mode, the 16-bit PWM counter will start counting-down from PERIOD to match with the value of the duty cycle CMPDATn (old); when this happens it will toggle the EPWM_CHn generator output to high. The counter will continue counting-down to zero;

at this moment, it toggles the EPWM_CHn generator output to low and CMPDATn (new) and PERIOD (new) are updated with CNTMODE=1 and requests the PWM interrupt if PWM interrupt is enabled (EPWM_INTEN).

Figure 6.8-6 shows the Edge-aligned PWM timing and operation flow.

Figure 6.8-6 EPWM Edge-aligned Type

Figure 6.8-6 EPWM Edge-aligned Waveform Output

Edge Aligned mode :
Period = (PERIOD + 1) * clock cycle
Duty ratio = CMPDAT / (PERIOD + 1)
CMRDAT > PERIOD --> always High
CMRDAT = 0 --> always Low

Edge Aligned mode
PERIOD = 99
Period = PERIOD + 1 = 99 + 1 = 100

Figure 6.8-7 EPWM Edge-aligned Mode Operation Timing

Figure 6.8-8 EPWM Edge-aligned Interrupt Diagram

Figure 6.8-9 EPWM Edge-aligned Flow Diagram

The EPWM period and duty control are decided by PWM down-counter register (PERIOD) and PWM comparator register (CMPDATn). The PWM-Timer timing operation is shown in Figure 6.8-11 EPWM-Timer Operation Timing. The pulse width modulation follows the formula below and the legend of PWM-Timer Comparator is shown in Figure 6.8-10 EPWM Legend of Internal Comparator Output of PWM-Timer. Note that the corresponding GPIO pins must be configured as PWM function for the corresponding PWM channel.

PWM frequency = HCLK / (clock divider) / (PERIOD+1)

Period = (PERIOD + 1) unit

Duty ratio = CMPDAT / (PERIOD+1)

CMPDAT > PERIOD: PWM output is always high

CMPDAT <= PERIOD: PWM output high duty = (CMPDAT) unit

CMPDAT = 0: PWM always low

Note: 1. Unit = one PWM clock cycle.

Note: n= 0 ~ 5.

Figure 6.8-10 EPWM Legend of Internal Comparator Output of PWM-Timer

Figure 6.8-11 EPWM-Timer Operation Timing

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 Down/Up Counting mode. The PWM counter first state is Down-counter mode which it will start from PERIOD plus one value and decrase to match the value of CMPDATn (old); this will cause the toggling of the EPWM_CHn generator output to high. The counter will continue counting to 0. Upon reaching this state counter is configured automatically to Up counting, when EPWM counter matches the CMPDATn (old) value again the PWMn generator output toggles to low. Once the EPWM counter overflows it will update the EPWM period register PERIOD (new) and duty cycle register CMPDATn (new) with CNTMODE = 1.

In Center-aligned mode, the EPWM has 4 types interrupt as Period interrupt (PIF), Up interrupt (CMPUIF), Central interrupt (CIF), and Down interrupt (CMPDIF).

PWM frequency = HCLK/(clock divider))/(2*(PERIOD+1))

Period = 2 x (PERIOD + 1) unit

Duty ratio = 2 x (CMPDAT / (PERIOD + 1))

CMPDAT > PERIOD: PWM output is always high

CMPDAT <= PERIOD: PWM output high duty = (2 x CMPDAT) unit

CMPDAT = 0: PWM always low

Note: 1. Unit = one PWM clock cycle.

Figure 6.8-12 shows the Center-aligned PWM timing and operation flow.

Figure 6.8-12 EPWM Center-aligned Type

Figure 6.8-13 EPWM Center-aligned Mode Operation Timing

Figure 6.8-14 EPWM Center-aligned Waveform Output

Figure 6.8-15 EPWM Center-aligned Interrupt Diagram

Figure 6.8-16 EPWM Center-aligned Flow Diagram

6.8.5.2 EPWM Port Output Control

EPWM unit has six output pins in this device. The PWM port outputs are PA.0~PA.5.

The driving type of PWM output ports can be initialized as Tri-state type or other types depending on the PA_MODE register setting after any reset, as shown in Figure 6.8-10.

6.8.5.3 Independent Mode

The EPWM is set as independent mode when MODE (EPWM_CTL[29:28]) = 00, there are six PWM channel outputs. Each channel is running its own duty-cycle.

6.8.5.4 Complementary Mode

Complementary mode is enabled when MODE (EPWM_CTL[29:28]) = 01.

In this module there are three duty-cycle generators utilized for complementary mode, with total of three EPWM output pair pins in this module. The total six EPWM outputs are grouped into output pairs of even and odd numbered outputs. In complimentary modes, the internal odd PWM signal EPWM_CHn, always be the complement of the corresponding even PWM signal. For example, EPWM_CH1 will be the complement of EPWM_CH0. EPWM_CH3 will be the complement of EPWM_CH2 and EPWM_CH5 will be the complement of EPWM_CH4.

6.8.5.5 Synchronized Mode

Synchronized mode is enabled when MODE (EPWM_CTL[29:28]) = 10.

In this mode, there are three PWM channel duty-cycle by setting, EPWM_CH0, EPWM_CH2,and EPWM_CH4. The total six PWM outputs are grouped into output pairs of even and odd numbered outputs. In synchronized mode, the internal odd PWM (1/3/5) signal must always be synchronize to the corresponding even PWM (0/2/4) signal.

6.8.5.6 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 8-bit down counter used to produce the dead-time insertion. The complementary outputs are delayed until the counter counts down to zero.

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

dead-time = PWM_CLK * (DTCNTnm+1). where nm, could be 01, 23, 45

The timing diagram as shown in Figure 6.8-17 indicates the dead-time insertion for one pair of PWM signals.

Figure 6.8-17 EPWM Dead-time Insertion

6.8.5.7 Group Mode

Group mode is enabled when GROUPEN (EPWM_CTL[30]) = 1.

This device supports Group mode control which allows all even PWM channels output to be duty controllable by EPWM_CH0 duty register.

If GROUPEN = 1, both (EPWM_CH2, EPWM_CH3) and (EPWM_CH4, EPWM_CH5) pairs will follow (EPWM_CH0, EPWM_CH1), which imply;

EPWM_CH4 = EPWM_CH2 = EPWM_CH0;

EPWM_CH5 = EPWM_CH3 = EPWM_CH1 = invert (EPWM_CH0) if Complementary mode is enabled when MODE (EPWM_CTL[29:28]) = 01.

Note: For applications, please do not use Group and Synchronous mode simultaneously because the Synchronous mode will be inactive.

6.8.5.8 Asymmetric Mode

Asymmetric mode only works under Center-aligned type. Asymmetric mode is enabled when ASYMEN (EPWM_CTL[20]) = 1. In this mode EPWM counter will compare with another compared value CMPU (EPWM_CMPDATn[31:16]) when counting up. If CMPU is not equal to the CMP, the EPWM will generate asymmetric waveform and set CMPUIFn (EPWM_INTSTS[13:8]) of the corresponding channel n.

Figure 6.8-18 EPWM Asymmetric Mode Timing Diagram

6.8.5.9 One-Shot Mode

The EPWM set as one-shot mode when CNTMODE (EPWM_CTL[8]) = 0. The EPWM output one pulse in one period when EPWM start run is set. Figure 6.8-19 shows one-shot mode status.

Figure 6.8-19 EPWM One-Shot Mode Architecture

6.8.5.10 Polarity Control

Each PWM port from PWM0_CH0 to PWM0_CH5 has independent polarity control to configure the polarity of active state of PWM output. By default, the PWM output is active high.

Figure 6.8-20 shows the initial state before PWM starts with different polarity settings.

Figure 6.8-20 EPWM Initial State and Polarity Control with Rising Edge Dead-time Insertion

6.8.5.11 Interrupt Architecture

There are sixteen interrupt sources for EPWM unit, which are PIF (EPWM_INTSTS[0]) PWM counter counts to period interrupt flag; CIF (EPWM_INTSTS[18]) PWM counter counts to central point of center-aligned type interrupt flag; CMPDIFn (EPWM_INTSTS[29:24]) PWM counter down-counts to CMPn (EPWM_CMPDATn[15:0]) interrupt flag; CMPUIFn (EPWM_INTSTS[13:8]) PWM counter up-counts to CMPUn (EPWM_CMPDATn[31:16]) interrupt flag, if operating in asymmetric type it up count to CMPUn (PWM_CMPDATn[31:16]); BRK0IF (PWM_INTSTS[16]) Brake0 interrupt flag, BRK1IF (PWM_INTSTS[17]) Brake1 interrupt flag.

The bits BRK0IEN (EPWM_INTEN[16]) and BRK1IEN (EPWM_INTEN[17]) control the brake interrupt enable; the bit PIEN (EPWM_INTEN[0]) control the PIF interrupt enable; the bit CIEN (EPWM_INTEN[18]) control the CIF interrupt enable; the bits CMPUIENn (EPWM_INTEN[13:8]) control the CMPUIFn interrupt enable; and the bits CMPDIENn (EPWM_INTEN[29:24]) control the CMPDIFn interrupt enable. Note that all the interrupt flags are set by hardware and must be cleared by software.

Figure 6.8-21 EPWM Interrupt Architecture

Note: For the BRKnIF's interrupt architecture illustration, see Figure 6.8-22.

6.8.5.12 EPWM Brake

This device supports two brake detectors, BRK0 and BRK1, and each of them has 4 brake signals, one external brake pin (BRAKE connected to BRK0 and BRK1 both), two analog comparator outputs and one ADC output. External brake pins have digital filter. The Brake function is controlled by the contents of the EPWM_BRKCTL register.

Figure 6.8-22 EPWM Brake Architecture

The BRK0 block will keep EPWM output to brake define value (set by EPWM_BRKCTL[29:24]) and need initial EPWM function again to release EPWM brake signal.

The BRK1 block has resume function, when BRK1 brake active, it will resume by itself with 12-bits delay counter.

Since both brake conditions being asserted will automatically cause BRKnIF (n:0,1) flag to be set, the user program can poll these brake flag bits or enable EPWM's brake interrupt (EPWM_INTEN) to determine which condition will cause a brake to occur.

6.8.5.13 EPWM Phase Change Function

The phase change function can be used to trigger PWM by TIMER module with ACMP by selectable TRGSEL (EPWM_PHCHG[22:20]) registers. To use Timer (or ACMP) trigger EPWM, by configuring both EPWM_PHCHG and EPWM_PHCHGNXT register. Each time when time-out event coming, EPWM_PHCHG's value will be updated by EPWM_PHCHGNXT's value automatically, EPWM_PHCHG's bit field is identical with EPWM_PHCHGNXT's, each time when EPWM_PHCHG updated, the related function will also change.

Besides trigger EPWM, phase change register also with mask control bits to change the phase of EPWM output. By setting 1 to corresponding channel's MSKENn (EPWM_PHCHG[13:8]) to enable channel's mask function, then corresponding channel will output level of MSKDATn (EPWM_PHCHG[5:0]).

1.1.1.1.1 EPWM Mask Output Function

In Phase Change function, each of the EPWM output can be manually overridden by using the appropriate bits in the EPWM Mask Enable function (MSKENn, n:0~5) and EPWM Mask Data register (MSKDATn, n:0~5) to drive EPWM pins to specified logic states independent of duty

cycle comparison units. The MSKENn register contains six bits, MSKEN[5:0] (EPWM_PHCHG[13:8] / EPWM_PHCHGNXT[13:8]) determine which PWM I/O pins will be overridden. On reset MSKENn is 00H. The MSKDATn register contains six bits, MSKDAT[5:0] (EPWM_PHCHG[5:0] / EPWM_PHCHGNXT[5:0]) determine the state of the PWM I/O pins when a particular output is masked via the MSKDATn bits. On reset MSKDATn is 00H. When the MSKEN[5:0] bits are set, the corresponding MSKDAT[5:0] bit will have effect on the PWM channel.

Figure 6.8-23 EPWM 3-phase Motor Mask Diagram

For example 1, Motor activating path is path 0 connects path 3, and path 0 is used as EPWM0 and path 3 is ON (short).

PWM channel 0 follow PWM generator.

PWM channels 1-5 are masked by MSKENn bits,.

PWM channels 1-5 outputs are determined by state of MSKDATn bits.

Switch 0 (On/Off)	**Control By EPWM0 (EPWM0 Frequency/Duty Generator).**
Switch 1 (Off)	MSKDAT1 = 0
Switch 2 (Off)	MSKDAT2 = 0
Switch 3 (On)	MSKDAT3 = 1
Switch 4 (Off)	MSKDAT4 = 0
Switch 5 (Off)	MSKDAT5 = 0

Figure 6.8-24 EPWM 3-phase Motor Mask Example 1

For example 2, Motor activating path is path 2 connects path 5, and path 2 is used as EPWM2 and path 5 is ON (short).

PWM channel 2 follow PWM generator.

PWM channels 0-1, 3-5 are masked by MSKENn bits,.

PWM channels 0-1. 3-5 outputs are determined by state of MSKDATn bits.

Switch 0 (Off)	**MSKDAT0 = 0**
Switch 1 (Off)	MSKDAT1 = 0
Switch 2 (On/Off)	**Control by EPWM2 (EPWM2 frequency/duty generator).**
Switch 3 (Off)	MSKDAT3 = 0
Switch 4 (Off)	MSKDAT4 = 0
Switch 5 (On)	MSKDAT5 = 1

Figure 6.8-25 EPWM 3-phase Motor Mask Example 2

1.1.1.1.2 EPWM Phase Change Hall_State Sensor Mode

The EPWM can also direct check Hall sensor state to change motor phase. When TRGSEL = 011b (EPWM_PHCHGNXT[22:20]) trigger by next Hall state, the Phase-Change controller will check CAPn_PCHG (n:0,2) status. If matched HALLSTS (EPWM_PHCHGNXT[18:16]) setting value then EPWM output data can be copied to EPWM_PHCHG from EPWM_PHCHGNEXT and change status of MOTOR simultaneously.

6.8.6 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
EPWM Base Address: **EPWM_BA = 0x4004_0000**				
EPWM_NPCTL	EPWM_BA+0x00	R/W	EPWM Negative Polarity Control Register	0x0000_0000
EPWM_CLKDIV	EPWM_BA+0x04	R/W	EPWM Clock Select Register	0x0000_0000
EPWM_CTL	EPWM_BA+0x08	R/W	EPWM Control Register	0x0000_0000
EPWM_PERIOD	EPWM_BA+0x0C	R/W	EPWM Period Counter Register	0x0000_0000
EPWM_CMPDAT0	EPWM_BA+0x24	R/W	EPWM Comparator Register 0	0x0000_0000
EPWM_CMPDAT1	EPWM_BA+0x28	R/W	EPWM Comparator Register 1	0x0000_0000
EPWM_CMPDAT2	EPWM_BA+0x2C	R/W	EPWM Comparator Register 2	0x0000_0000
EPWM_CMPDAT3	EPWM_BA+0x30	R/W	EPWM Comparator Register 3	0x0000_0000
EPWM_CMPDAT4	EPWM_BA+0x34	R/W	EPWM Comparator Register 4	0x0000_0000
EPWM_CMPDAT5	EPWM_BA+0x38	R/W	EPWM Comparator Register 5	0x0000_0000
EPWM_CNT	EPWM_BA+0x3C	R	EPWM Data Register	0x0000_0000
EPWM_INTEN	EPWM_BA+0x54	R/W	EPWM Interrupt Enable Register	0x0000_0000
EPWM_INTSTS	EPWM_BA+0x58	R/W	EPWM Interrupt Status Register	0x0000_0000
EPWM_RESDLY	EPWM_BA+0x5C	R/W	EPWM BRK Low Voltage Detect Resume Delay	0x0000_0000
EPWM_BRKCTL	EPWM_BA+0x60	R/W	EPWM Fault Brake Control Register	0x0000_0000
EPWM_DTCTL	EPWM_BA+0x64	R/W	EPWM Dead-zone Interval Register	0x0000_0000
EPWM_PHCHG	EPWM_BA+0x78	R/W	EPWM Phase Changed Register	0x0000_0000
EPWM_PHCHGNXT	EPWM_BA+0x7C	R/W	EPWM Next Phase Change Register	0x0000_0000
EPWM_PHCHGALT	EPWM_BA+0x80	R/W	EPWM Phase Change Alternative Control Register	0x0000_0000
EPWM_IFA	EPWM_BA+0x84	R/W	EPWM Period Interrupt Accumulation Control Register	0x0000_00F0

6.8.7 Register Description

EPWM Negative Polarity Control Register (EPWM_NPCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_NPCTL	EPWM_BA+0x00	R/W	EPWM Negative Polarity 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		**NEGPOLAR5**	**NEGPOLAR4**	**NEGPOLAR3**	**NEGPOLAR2**	**NEGPOLAR1**	**NEGPOLAR0**

Bits	Description	
[31:6]	**Reserved**	Reserved.
[5]	**NEGPOLAR5**	**PWM5 Negative Polarity Control** The register bit controls polarity/active state of real PWM output. 0 = PWM output is active high. 1 = PWM output is active low.
[4]	**NEGPOLAR4**	**PWM4 Negative Polarity Control** The register bit controls polarity/active state of real PWM output. 0 = PWM output is active high. 1 = PWM output is active low.
[3]	**NEGPOLAR3**	**PWM3 Negative Polarity Control** The register bit controls polarity/active state of real PWM output. 0 = PWM output is active high. 1 = PWM output is active low.
[2]	**NEGPOLAR2**	**PWM2 Negative Polarity Control** The register bit controls polarity/active state of real PWM output. 0 = PWM output is active high. 1 = PWM output is active low.
[1]	**NEGPOLAR1**	**PWM1 Negative Polarity Control** The register bit controls polarity/active state of real PWM output. 0 = PWM output is active high. 1 = PWM output is active low.

Bits	Description	
[0]	**NEGPOLAR0**	**PWM0 Negative Polarity Control** The register bit controls polarity/active state of real PWM output. 0 = PWM output is active high. 1 = PWM output is active low.

EPWM Clock Selector Register (EPWM_CLKDIV)

Register	Offset	R/W	Description	Reset Value
EPWM_CLKDIV	EPWM_BA+0x04	R/W	EPWM Clock 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				**CLKDIV**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**CLKDIV**	**EPWM Clock Divider (9 Step Divider)** Select clock input for PWM timer 0000 = 1 (HCLK / 2^0). 0001 = 1/2 (HCLK / 2^1). 0010 = 1/4 (HCLK / 2^2). 0011 = 1/8 (HCLK / 2^3). 0100 = 1/16 (HCLK / 2^4). 0101 = 1/32 (HCLK / 2^5). 0110 = 1/64 (HCLK / 2^6). 0111 = 1/128 (HCLK / 2^7). 1000 = 1/256 (HCLK / 2^8). 1001~ 1111 = Reserved.

EPWM Control Register (EPWM_CTL)

Register	Offset	R/W	Description	Reset Value
EPWM_CTL	EPWM_BA+0x08	R/W	EPWM Control Register	0x0000_0000

31	30	29	28	27	26	25	24
CNTTYPE	**GROUPEN**	**MODE**		**CNTCLR**	**DTCNT45**	**DTCNT23**	**DTCNT01**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DBGTRIOFF	**Reserved**		**ASYMEN**	**Reserved**		**HCUPDT**	
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							**CNTMODE**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**CNTEN5**	**CNTEN4**	**CNTEN3**	**CNTEN2**	**CNTEN1**	**CNTEN0**

Bits	Description	
[31]	**CNTTYPE**	**PWM Aligned Type Selection** 0 = Edge-aligned type. 1 = Center-aligned type.
[30]	**GROUPEN**	**Group Bit** 0 = The signals timing of PWM0, PWM2 and PWM4 are independent. 1 = Unify the signals timing of PWM0, PWM2 and PWM4 in the same phase which is controlled by PWM0.
[29:28]	**MODE**	**PWM Operating Mode Selection** 00 = Independent mode. 01 = Complementary mode. 10 = Reserved. 11 = Reserved.
[27]	**CNTCLR**	**Clear PWM Counter Control Bit** 0 = Do not clear PWM counter. 1 = 16-bit PWM counter cleared to 0x000. **Note:** It is automatically cleared by hardware.
[26]	**DTCNT45**	**Dead-zone 4 Generator Enable/Disable (PWM4 and PWM5 Pair for PWM Group)** 0 = Dead-zone 4 Generator Disabled. 1 = Dead-zone 4 Generator Enabled. **Note:** When the dead-zone generator is enabled, the pair of PWM4 and PWM5 becomes a complementary pair for PWM group.
[25]	**DTCNT23**	**Dead-zone 2 Generator Enable/Disable (PWM2 and PWM3 Pair for PWM Group)** 0 = Dead-zone 2 Generator Disabled. 1 = Dead-zone 2 Generator Enabled. **Note:** When the dead-zone generator is enabled, the pair of PWM2 and PWM3 becomes a complementary pair for PWM group.

Bits	Description	
[24]	**DTCNT01**	**Dead-zone 0 Generator Enable/Disable (PWM0 and PWM1 Pair for PWM Group)** 0 = Dead-zone 0 Generator Disabled. 1 = Dead-zone 0 Generator Enabled. **Note:** When the dead-zone generator is enabled, the pair of PWM0 and PWM1 becomes a complementary pair for PWM group.
[23]	**DBGTRIOFF**	**PWM Debug Mode Configuration Bit (Available in DEBUG Mode Only)** 0 = Safe mode: The timer is frozen and PWM outputs are shut down Safe state for the inverter. The timer can still be re-started from where it stops. 1 = Normal mode: The timer continues to operate normally May be dangerous in some cases since a constant duty cycle is applied to the inverter (no more interrupts serviced).
[22:21]	**Reserved**	Reserved.
[20]	**ASYMEN**	**Asymmetric Mode in Center-aligned Type** 0 = Symmetric mode in center-aligned type. 1 = Asymmetric mode in center-aligned type.
[19:18]	**Reserved**	Reserved.
[17:16]	**HCUPDT**	**Half Cycle Update Enable for Center-aligned Type** 00= Update PERIOD & CMP at pwm_counter = PERIOD (Period). 01 = Update PERIOD & CMP at pwm_counter = 0. 10 = Update PERIOD & CMP at half cycle (counter = 0 & PERIOD, both update). 11 = Update PERIOD & CMP at pwm_counter = PERIOD (Period).
[15:9]	**Reserved**	Reserved.
[8]	**CNTMODE**	**PWM-timer Auto-reload/One-shot Mode** 0 = One-shot mode. 1 = Auto-reload mode.
[7:6]	**Reserved**	Reserved.
[5]	**CNTEN5**	**PWM-timer 5 Enable/Disable Start Run** 0 = Corresponding PWM-timer running Stopped. 1 = Corresponding PWM-timer start run Enabled.
[4]	**CNTEN4**	**PWM-timer 4 Enable/Disable Start Run** 0 = Corresponding PWM-timer running Stopped. 1 = Corresponding PWM-timer start run Enabled.
[3]	**CNTEN3**	**PWM-timer 3 Enable/Disable Start Run** 0 = Corresponding PWM-timer running Stopped. 1 = Corresponding PWM-timer start run Enabled.
[2]	**CNTEN2**	**PWM-timer 2 Enable/Disable Start Run** 0 = Corresponding PWM-timer running Stopped. 1 = Corresponding PWM-timer start run Enabled.
[1]	**CNTEN1**	**PWM-timer 1 Enable/Disable Start Run** 0 = Corresponding PWM-timer running Stopped. 1 = Corresponding PWM-timer start run Enabled.

Bits	Description	
[0]	**CNTEN0**	**PWM-timer 0 Enable/Disable Start Run** 0 = Corresponding PWM-timer running Stopped. 1 = Corresponding PWM-timer start run Enabled.

EPWM Period Counter Register (EPWM_PERIOD)

Register	Offset	R/W	Description	Reset Value
EPWM_PERIOD	EPWM_BA+0x0C	R/W	EPWM Period 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**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**PWM Counter/Timer Loaded Value** PERIODn determines the PWM Period. Edge-aligned mode: where xy, could be 01, 23, 45 depending on the selected PWM channel. PWM frequency = PWMxy_CLK[HCLK]/clock divider[EPWM_CLKDIV]. PWM Clock Cycle = 1 / PWM Freq. Period = PWM Clock Cycle * (PERIOD+1). Duty = PWM Clock Cycle * CMPn. Duty ratio = CMPn/(PERIOD+1). CMPn >= PERIOD: PWM output is always high. CMPn < PERIOD: PWM low width = (PERIODn-CMPn) unit; PWM high width = (CMP) unit. CMPn = 0: PWM always output low. Center-aligned mode: where xy, could be 01, 23, 45 depending on the selected PWM channel. Period = 1/ (PWMxy_CLK[HCLK]/clock divider[EPWM_CLKDIV] /(2*(PERIOD+1)). Duty ratio = 2* (CMPn/(PERIOD+1)). CMPn >= PERIOD: PWM output is always high. CMPn < PERIOD: PWM low width = (PERIOD - CMPn) x 2 unit; PWM high width = CMP x 2 unit. CMPn = 0: PWM always low. (Unit = One PWM clock cycle). **Note:** Any write to PERIODn will take effect in next PWM cycle.

EPWM Comparator Register 0-5 (EPWM_CMPDAT0-5)

Register	Offset	R/W	Description	Reset Value
EPWM_CMPDAT0	EPWM_BA+0x24	R/W	EPWM Comparator Register 0	0x0000_0000
EPWM_CMPDAT1	EPWM_BA+0x28	R/W	EPWM Comparator Register 1	0x0000_0000
EPWM_CMPDAT2	EPWM_BA+0x2C	R/W	EPWM Comparator Register 2	0x0000_0000
EPWM_CMPDAT3	EPWM_BA+0x30	R/W	EPWM Comparator Register 3	0x0000_0000
EPWM_CMPDAT4	EPWM_BA+0x34	R/W	EPWM Comparator Register 4	0x0000_0000
EPWM_CMPDAT5	EPWM_BA+0x38	R/W	EPWM Comparator Register 5	0x0000_0000

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

Bits	Description	
[31:16]	**CMPU**	**PWM Comparator Register for UP Counter in Center-aligned Asymmetric Mode** CMPU > PERIOD: @ up counter PWM output is keep to Max. duty. CMPU <= PERIOD: (CMPUn + CMPn) unit. CMP <= PERIOD: PWM output high duty = (CMP + CMPU) unit. Others: PWM output is always low (Unit = One PWM clock cycle).

Bits	Description	
[15:0]	**CMP**	**PWM Comparator Register** CMP determines the PWM Duty. Edge-aligned mode: where xy, could be 01, 23, 45 depending on the selected PWM channel. Period = (PERIOD + 1) unit. Duty ratio = CMP / (PERIOD+1). CMP > PERIOD: PWM output is always high CMP <= PERIOD: PWM output high duty = (CMP) unit. CMP = 0: PWM always low. Center-aligned mode: where xy, could be 01, 23, 45 depending on the selected PWM channel. Period = 2 x (PERIOD + 1) unit. Duty ratio = 2 x (CMP / PERIOD). CMP > PERIOD: PWM output is always high CMP <= PERIOD: PWM output high duty = (2 x CMP) unit. CMP = 0: PWM always low. (Unit = One PWM clock cycle.) **Note:** Any write to CMPn will take effect in next PWM cycle.

EPWM Data Register (EPWM_CNT)

Register	Offset	R/W	Description	Reset Value
EPWM_CNT	EPWM_BA+0x3C	R	EPWM Data Register	0x0000_0000

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

Bits	Description	
[31]	**CNTDIR**	**PWM Counter (Up/Down) Direction** 0 = PWM counter is down counting. 1 = PWM counter is up counting.
[30:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**PWM Data** User can monitor CNT to know the current value in 16-bit down counter.

EPWM Interrupt Enable Register (EPWM_INTEN)

Register	Offset	R/W	Description	Reset Value
EPWM_INTEN	EPWM_BA+0x54	R/W	EPWM Interrupt Enable Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		**CMPDIEN5**	**CMPDIEN4**	**CMPDIEN3**	**CMPDIEN2**	**CMPDIEN1**	**CMPDIEN0**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**CIEN**	**BRK1IEN**	**BRK0IEN**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**CMPUIEN5**	**CMPUIEN4**	**CMPUIEN3**	**CMPUIEN2**	**CMPUIEN1**	**CMPUIEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**PIEN**

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**CMPDIEN5**	**PWM Channel 5 DOWN Interrupt Enable Bit** DOWN for Edge-aligned and Center-aligned 0 = Interrupt compare Disabled. 1 = Interrupt when EPWM_CH5 PWM DOWN counter reaches EPWM_CMPDAT5 Enabled.
[28]	**CMPDIEN4**	**PWM Channel 4 DOWN Interrupt Enable Bit** DOWN for Edge-aligned and Center-aligned 0 = Interrupt compare Disabled. 1 = interrupt when EPWM_CH4 PWM DOWN counter reaches EPWM_CMPDAT4 Enabled.
[27]	**CMPDIEN3**	**PWM Channel 3 DOWN Interrupt Enable Bit** DOWN for Edge-aligned and Center-aligned 0 = Interrupt compare Disabled. 1 = interrupt when EPWM_CH3 PWM DOWN counter reaches EPWM_CMPDAT3 Enabled.
[26]	**CMPDIEN2**	**PWM Channel 2 DOWN Interrupt Enable Bit** DOWN for Edge-aligned and Center-aligned 0 = Interrupt compare Disabled. 1 = interrupt when EPWM_CH2 PWM DOWN counter reaches EPWM_CMPDAT2 Enabled.
[25]	**CMPDIEN1**	**PWM Channel 1 DOWN Interrupt Enable Bit** DOWN for Edge-aligned and Center-aligned 0 = Interrupt compare Disabled. 1 = interrupt when EPWM_CH1 PWM DOWN counter reaches EPWM_CMPDAT1 Enabled.

Bits	Description	
[24]	**CMPDIEN0**	**PWM Channel 0 DOWN Interrupt Enable Bit** DOWN for Edge-aligned and Center-aligned 0 = Interrupt compare Disabled. 1 = interrupt when EPWM_CH0 PWM DOWN counter reaches EPWM_CMPDAT0 Enabled.
[23:19]	**Reserved**	Reserved.
[18]	**CIEN**	**PWM Central Interrupt Enable Bit** for Center-aligned only 0 = Interrupt when EPWM Central Enabled. 1 = Interrupt when EPWM Central Enabled.
[17]	**BRK1IEN**	**Fault Brake1 Interrupt Enable Bit** 0 = BRK1IF trigger PWM interrupt Disabed. 1 = BRK1IF trigger PWM interrupt Enabled .
[16]	**BRK0IEN**	**Fault Brake0 Interrupt Enable Bit** 0 = BRK0IF trigger PWM interrupt Disabled. 1 = BRK0IF trigger PWM interrupt Enabled.
[15:14]	**Reserved**	Reserved.
[13]	**CMPUIEN5**	**PWM Channel 5 UP Interrupt Enable Bit** UP for Center-aligned only 0 = PWM Channel 5 UP Interrupt Disabled. 1 = Interrupt when EPWM_CH5 PWM UP counter reaches EPWM_CMPDAT5 Enabled.
[12]	**CMPUIEN4**	**PWM Channel 4 UP Interrupt Enable Bit** UP for Center-aligned only 0 = EPWM_CH4 PWM UP counter reaches EPWM_CMPDAT4 interrupt Disabled. 1 = EPWM_CH4 PWM UP counter reaches EPWM_CMPDAT4 interrupt Enabled.
[11]	**CMPUIEN3**	**PWM Channel 3 UP Interrupt Enable Bit** UP for Center-aligned only 0 = EPWM_CH3 PWM UP counter reaches EPWM_CMPDAT3 interrupt Disabled. 1 = EPWM_CH3 PWM UP counter reaches EPWM_CMPDAT3 interrupt Enabled.
[10]	**CMPUIEN2**	**PWM Channel 2 UP Interrupt Enable Bit** UP for Center-aligned only 0 = EPWM_CH2 PWM UP counter reaches EPWM_CMPDAT2 interrupt Disabled. 1 = EPWM_CH2 PWM UP counter reaches EPWM_CMPDAT2 interrupt Enabled.
[9]	**CMPUIEN1**	**PWM Channel 1 UP Interrupt Enable Bit** UP for Center-aligned only 0 = EPWM_CH1 PWM UP counter reaches EPWM_CMPDAT1 interrupt Disabled. 1 = EPWM_CH1 PWM UP counter reaches EPWM_CMPDAT1 interrupt Enabled.
[8]	**CMPUIEN0**	**PWM Channel 0 UP Interrupt Enable Bit** UP for Center-aligned only 0 = EPWM_CH0 PWM UP counter reaches EPWM_CMPDAT0 interrupt Disabled. 1 = EPWM_CH0 PWM UP counter reaches EPWM_CMPDAT0 interrupt Enabled.

Bits	Description	
[7:1]	**Reserved**	Reserved.
[0]	**PIEN**	**PWM Channel 0 Period Interrupt Enable Bit** for Edge-aligned and Center-aligned 0 = EPWM Period interrupt Disabled . 1 = EPWM Period interrupt Enabled .

EPWM Interrupt Status Register (EPWM_INTSTS)

Register	Offset	R/W	Description	Reset Value
EPWM_INTSTS	EPWM_BA+0x58	R/W	EPWM Interrupt Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		CMPDIF5	CMPDIF4	CMPDIF3	CMPDIF2	CMPDIF1	CMPDIF0
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					CIF	BRK1IF	BRK0IF
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		CMPUIF5	CMPUIF4	CMPUIF3	CMPUIF2	CMPUIF1	CMPUIF0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							PIF

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**CMPDIF5**	**PWM Channel 5 DOWN Interrupt Flag** Flag is set by hardware when a channel 5 PWM DOWN counter reaches EPWM_CMPDAT5. Software can write 1 to clear this bit.
[28]	**CMPDIF4**	**PWM Channel 4 DOWN Interrupt Flag** Flag is set by hardware when a channel 4 PWM DOWN counter reaches EPWM_CMPDAT4. Software can write 1 to clear this bit.
[27]	**CMPDIF3**	**PWM Channel 3 DOWN Interrupt Flag** Flag is set by hardware when a channel 3 PWM DOWN counter reaches EPWM_CMPDAT3. Software can write 1 to clear this bit.
[26]	**CMPDIF2**	**PWM Channel 2 DOWN Interrupt Flag** Flag is set by hardware when a channel 2 PWM DOWN counter reaches EPWM_CMPDAT2. Software can write 1 to clear this bit.
[25]	**CMPDIF1**	**PWM Channel 1 DOWN Interrupt Flag** Flag is set by hardware when a channel 1 PWM DOWN counter reaches EPWM_CMPDAT1. Software can write 1 to clear this bit.
[24]	**CMPDIF0**	**PWM Channel 0 DOWN Interrupt Flag** Flag is set by hardware when a channel 0 PWM DOWN counter reaches EPWM_CMPDAT0. Software can write 1 to clear this bit.
[23:19]	**Reserved**	Reserved.
[18]	**CIF**	**PWM Channel 0 Central Interrupt Flag** Flag is set by hardware when PWM down counter reaches zero point. Software can write 1 to clear this bit.

Bits	Description	
[17]	**BRK1IF**	**PWM Brake1 Flag** 0 = PWM Brake does not recognize a falling signal at BKP1. 1 = When PWM Brake detects a falling signal at pin BKP1, this flag will be set to high. **Note:** Software can write 1 to clear this bit.
[16]	**BRK0IF**	**PWM Brake0 Flag** 0 = PWM Brake does not recognize a falling signal at BKP0. 1 = When PWM Brake detects a falling signal at pin BKP0, this flag will be set to high. **Note:** Software can write 1 to clear this bit.
[15:14]	**Reserved**	Reserved.
[13]	**CMPUIF5**	**PWM Channel 5 UP Interrupt Flag** Flag is set by hardware when a channel 5 PWM UP counter reaches PWM_CMPDAT5. Software can write 1 to clear this bit.
[12]	**CMPUIF4**	**PWM Channel 4 UP Interrupt Flag** Flag is set by hardware when a channel 4 PWM UP counter reaches PWM_CMPDAT4. Software can write 1 to clear this bit.
[11]	**CMPUIF3**	**PWM Channel 3 UP Interrupt Flag** Flag is set by hardware when a channel 3 PWMUP counter reaches PWM_CMPDAT3. Software can write 1 to clear this bit.
[10]	**CMPUIF2**	**PWM Channel 2 UP Interrupt Flag** Flag is set by hardware when a channel 2 PWM UP counter reaches PWM_CMPDAT2. Software can write 1 to clear this bit.
[9]	**CMPUIF1**	**PWM Channel 1 UP Interrupt Flag** Flag is set by hardware when a channel 1 PWM UP counter reaches PWM_CMPDAT1. Software can write 1 to clear this bit.
[8]	**CMPUIF0**	**PWM Channel 0 UP Interrupt Flag** Flag is set by hardware when a channel 0 PWM UP counter reaches PWM_CMPDAT0. Software can write 1 to clear this bit.
[7:1]	**Reserved**	Reserved.
[0]	**PIF**	**PWM Channel 0 Period Interrupt Flag** Edge-aligned mode: Flag is set by hardware when PWM down counter reaches zero point. Center-aligned mode: Flag is set by hardware when PWM down counter reaches zero point and then up counter reaches EPWM_PERIOD. Software can write 1 to clear this bit.

EPWM BRK Low Voltage Detect Resume Delay (EPWM_RESDLY)

Register	Offset	R/W	Description	Reset Value
EPWM_RESDLY	EPWM_BA+0x5C	R/W	EPWM BRK Low Voltage Detect Resume Delay	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				**DELAY**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DELAY							

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:0]	**DELAY**	**PWM BRK Low Voltage Detect Resume Delay** 12 bits Down-Counter

EPWM Fault Brake Control Register (EPWM_BRKCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_BRKCTL	EPWM_BA+0x60	R/W	EPWM Fault Brake Control Register	0x0000_0000

31	30	29	28	27	26	25	24
NFPEN	**Reserved**	**BKOD5**	**BKOD4**	**BKOD3**	**BKOD2**	**BKOD1**	**BKOD0**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
LVDTYPE	**LVDBKEN**	**BRK1PEN**	**BK1ADCEN**	**BRK1A1EN**	**BRK1A0EN**	**SWBRK**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**BRK0PEN**	**BK0ADCEN**	**BRK0A1EN**	**BRK0A0EN**	**BRK1EN**	**BRK0EN**

Bits	Description	
[31]	**NFPEN**	**Noise Filter for External Brake Input Pin (BRAKE) Enable Bit** 0 = Noise Filter for External Brake Input Pin (BRAKE) Disabled. 1 = Noise Filter for External Brake Input Pin (BRAKE) Enabled.
[30]	**Reserved**	Reserved.
[29]	**BKOD5**	**PWM Channel 5 Brake Output Selection** 0 = PWM output low when fault brake conditions asserted. 1 = PWM output high when fault brake conditions asserted.
[28]	**BKOD4**	**PWM Channel 4 Brake Output Selection** 0 = PWM output low when fault brake conditions asserted. 1 = PWM output high when fault brake conditions asserted.
[27]	**BKOD3**	**PWM Channel 3 Brake Output Selection** 0 = PWM output low when fault brake conditions asserted. 1 = PWM output high when fault brake conditions asserted.
[26]	**BKOD2**	**PWM Channel 2 Brake Output Selection** 0 = PWM output low when fault brake conditions asserted. 1 = PWM output high when fault brake conditions asserted.
[25]	**BKOD1**	**PWM Channel 1 Brake Output Selection** 0 = PWM output low when fault brake conditions asserted. 1 = PWM output high when fault brake conditions asserted.
[24]	**BKOD0**	**PWM Channel 0 Brake Output Selection** 0 = PWM output low when fault brake conditions asserted. 1 = PWM output high when fault brake conditions asserted.
[23:16]	**Reserved**	Reserved.

Bits	Description	
[15]	**LVDTYPE**	**Low-level Detection Resume Type** 0 = Brake resume at BRK resume delay counter counting to 0. 1 = Brake resume at period edge.
[14]	**LVDBKEN**	**Low-level Detection Trigger PWM Brake Function 1 Enable Bit** 0 = Brake Function 1 triggered by Low-level detection Disabled. 1 = Brake Function 1 triggered by Low-level detection Enabled.
[13]	**BRK1PEN**	**BRK1 Source From External Pin Enable Bit** 0 = BRK1 Source From External Pin Disabled. 1 = BRK1 Source From External Pin Enabled.
[12]	**BK1ADCEN**	**BRK1 Source From ADC Enable Bit** 0 = BRK1 Source From ADC Disabled. 1 = BRK1 Source From ADC Enabled.
[11]	**BRK1A1EN**	**BRK1 Source From ACMP1 Enable Bit** 0 = BRK1 Source From ACMP1 Disabled. 1 = BRK1 Source From ACMP1 Enabled.
[10]	**BRK1A0EN**	**BRK1 Source From ACMP0 Enable Bit** 0 = BRK1 Source From ACMP0 Disabled. 1 = BRK1 Source From ACMP0 Enabled.
[9]	**SWBRK**	**Software Break** 0 = Software break and back to normal PWM function Disabled. 1 = Issue Software break Enabled.
[8:6]	**Reserved**	Reserved.
[5]	**BRK0PEN**	**BRK0 Source From External Pin Enable Bit** 0 = BRK0 Source From External Pin Disabled. 1 = BRK0 Source From External Pin Enabled.
[4]	**BK0ADCEN**	**BRK0 Source From ADC Enable Bit** 0 = BRK0 Source From ADC Disabled. 1 = BRK0 Source From ADC Enabled.
[3]	**BRK0A1EN**	**BRK0 Source From ACMP1 Enable Bit** 0 = BRK0 Source From ACMP1 Disabled. 1 = BRK0 Source From ACMP1 Enabled.
[2]	**BRK0A0EN**	**BRK0 Source From ACMP0 Enable Bit** 0 = BRK0 Source From ACMP0 Disabled. 1 = BRK0 Source From ACMP0 Enabled.
[1]	**BRK1EN**	**Brake1 Function Enable Bit** 0 = Brake1 detect function Disabled. 1 = Brake1 detect function Enabled.
[0]	**BRK0EN**	**Brake0 Function Enable Bit** 0 = Brake0 detect function Disabled. 1 = Brake0 detect function Enabled.

EPWM Dead-zone Interval Register (EPWM_DTCTL)

Register	Offset	R/W	Description	Reset Value
EPWM_DTCTL	EPWM_BA+0x64	R/W	EPWM Dead-zone Interval Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DTCNT45							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
DTCNT23							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DTCNT01							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**DTCNT45**	**Dead-zone Interval Register for Pair of Channel4 and Channel5 (PWM4 and PWM5 Pair)** These 8 bits determine dead-zone length. The unit time of dead-zone length is received from corresponding EPWM_CLKDIV bits.
[15:8]	**DTCNT23**	**Dead-zone Interval Register for Pair of Channel2 and Channel3 (PWM2 and PWM3 Pair)** These 8 bits determine dead-zone length. The unit time of dead-zone length is received from corresponding EPWM_CLKDIV bits.
[7:0]	**DTCNT01**	**Dead-zone Interval Register for Pair of Channel0 and Channel1 (PWM0 and PWM1 Pair)** These 8 bits determine dead-zone length. The unit time of dead-zone length is received from corresponding EPWM_CLKDIV bits.

EPWM Phase Change Register (EPWM_PHCHG)

Register	Offset	R/W	Description	Reset Value
EPWM_PHCHG	EPWM_BA+0x78	R/W	EPWM Phase Changed Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		ACMP1TEN	ACMP0TEN	A1POSSEL		A0POSSEL	
23	22	21	20	19	18	17	16
Reserved	TRGSEL			Reserved			
15	14	13	12	11	10	9	8
Reserved		MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0
7	6	5	4	3	2	1	0
Reserved		MSKDAT5	MSKDAT4	MSKDAT3	MSKDAT2	MSKDAT1	MSKDAT0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**ACMP1TEN**	**ACMP1 Trigger Function Enable Bit** 0 = ACMP1 trigger PWM function Disabled. 1 = ACMP1 trigger PWM function Enabled.
[28]	**ACMP0TEN**	**ACMP0 Trigger Function Enable Bit** 0 = ACMP0 trigger PWM function Disabled. 1 = ACMP0 trigger PWM function Enabled.
[27:26]	**A1POSSEL**	**Alternative Comparator 1 Positive Input Selection** Select the positive input source of ACMP1. 00 = Select ACMP1_P0 (PC.0) as the input of ACMP1. 01 = Select ACMP1_P1 (PC.1) as the input of ACMP1. 10 = Select ACMP1_P2 (PD.1) as the input of ACMP1. 11 = Reserved.
[25:24]	**A0POSSEL**	**Alternative Comparator 0 Positive Input Selection** Select the positive input source of ACMP0. 00 = Select ACMP0_P0 (PB.0) as the input of ACMP0. 01 = Select ACMP0_P1 (PB.1) as the input of ACMP0. 10 = Select ACMP0_P2 (PB.2) as the input of ACMP0. 11 = Reserved.
[23]	**Reserved**	Reserved.

Bits	Description	
[22:20]	**TRGSEL**	**Phase Change Trigger Selection** Select the trigger condition to load PHCHG from PHCHG_NXT. When the trigger condition occurs it will load PHCHG_NOW with PHCHG_NXT. Phase Change: PWM outputs are masked according with the definition of MSKENn and MSKDATn in PHCHG_NOW. 000 = Triggered by Timer0 event. 001 = Triggered by Timer1 event. 010 = Triggered by Timer2 event. 011 = Triggered by HALLSTS (EPWM_PHCHGNXT[18:16]) matched hall sensor state. 100 = Triggered by ACMP0 event. 101 = Triggered by ACMP1 event. 110 = Reserved. 111 = Auto Phase Change Function Disabled.
[19:14]	**Reserved**	Reserved.
[13]	**MSKEN5**	**Enable PWM5 Mask Function** 0 = PWM5 Mask Function Disabled. 1 = PWM5 Mask Function Enabled.
[12]	**MSKEN4**	**Enable PWM4 Mask Function** 0 = PWM4 Mask Function Disabled. 1 = PWM4 Mask Function Enabled.
[11]	**MSKEN3**	**Enable PWM3 Mask Function** 0 = PWM3 Mask Function Disabled. 1 = PWM3 Mask Function Enabled.
[10]	**MSKEN2**	**Enable PWM2 Mask Function** 0 = PWM2 Mask Function Disabled. 1 = PWM2 Mask Function Enabled.
[9]	**MSKEN1**	**Enable PWM1 Mask Function** 0 = PWM1 Mask Function Disabled. 1 = PWM1 Mask Function Enabled.
[8]	**MSKEN0**	**Enable PWM0 Mask Function** 0 = PWM0 Mask Function Disabled. 1 = PWM0 Mask Function Enabled.
[7:6]	**Reserved**	Reserved.
[5]	**MSKDAT5**	**Enable PWM5 Mask Data** 0 = PWM5 state is masked with zero. 1 = PWM5 state is masked with one.
[4]	**MSKDAT4**	**Enable PWM4 Mask Data** 0 = PWM4 state is masked with zero. 1 = PWM4 state is masked with one.
[3]	**MSKDAT3**	**Enable PWM3 Mask Data** 0 = PWM3 state is masked with zero. 1 = PWM3 state is masked with one.

Bits	Description	
[2]	**MSKDAT2**	**Enable PWM2 Mask Data** 0 = PWM2 state is masked with zero. 1 = PWM2 state is masked with one.
[1]	**MSKDAT1**	**Enable PWM1 Mask Data** 0 = PWM1 state is masked with zero. 1 = PWM1 state is masked with one.
[0]	**MSKDAT0**	**Enable PWM0 Mask Data** 0 = PWM0 state is masked with zero. 1 = PWM0 state is masked with one.

EPWM Next Phase Change Register (EPWM_PHCHGNXT)

Register	Offset	R/W	Description	Reset Value
EPWM_PHCHGNXT	EPWM_BA+0x7C	R/W	EPWM Next Phase Change Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved		ACMP1TEN	ACMP0TEN	A1POSSEL		A0POSSEL	
23	22	21	20	19	18	17	16
Reserved	TRGSEL			Reserved	HALLSTS		
15	14	13	12	11	10	9	8
Reserved		MSKEN5	MSKEN4	MSKEN3	MSKEN2	MSKEN1	MSKEN0
7	6	5	4	3	2	1	0
Reserved		MSKDAT5	MSKDAT4	MSKDAT3	MSKDAT2	MSKDAT1	MSKDAT0

Bits	Description	
[31:30]	**Reserved**	Reserved.
[29]	**ACMP1TEN**	**ACMP1 Trigger Function Control Preset Bit** This bit will be load to bit ACMP1TEN in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[28]	**ACMP0TEN**	**ACMP0 Trigger Function Control Preset Bit** This bit will be load to bit ACMP0TEN in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[27:26]	**A1POSSEL**	**Alternative Comparator 1 Positive Input Selection Preset Bits** This bit field will be load to bit field A1POSSEL in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[25:24]	**A0POSSEL**	**Alternative Comparator 0 Positive Input Selection Preset Bits** This bit field will be load to bit field A0POSSEL in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[23]	**Reserved**	Reserved.

Bits	Description	
[22:20]	**TRGSEL**	**Phase Change Trigger Selection Preset Bits** This bit field will be load to bit field TRGSEL in PHCHG_NOW when load trigger condition occurs. 000 = Triggered by Timer0 event. 001 = Triggered by Timer1 event. 010 = Triggered by Timer2 event. 011 = Triggered by HALLSTS (EPWM_PHCHGNXT[18:16]) matched hall sensor state. 100 = Triggered by ACMP0 event. 101 = Triggered by ACMP1 event. 110 = Reserved. 111 = Auto Phase Change Function Disabled. Refer to register EPWM_PHCHG for detailed definition.
[19]	**Reserved**	Reserved.
[18:16]	**HALLSTS**	**Predicted Next HALL State** This bit field indicates the predicted hall state at next commutation. If TRGSEL (EPWM_PHCHG[22:20]) = 0x3,. the hardware will compare bits (CAP2, CAP1, CAP0) in timer 2 with HALLSTS [2:0] when any hall state change occurs. If the comparison is matched it will trigger phase change function.
[15:14]	**Reserved**	Reserved.
[13]	**MSKEN5**	**Enable PWM5 Mask Function Preset Bit** This bit will be load to bit MSKEN5 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[12]	**MSKEN4**	**Enable PWM4 Mask Function Preset Bit** This bit will be load to bit MSKEN4 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[11]	**MSKEN3**	**Enable PWM3 Mask Function Preset Bit** This bit will be load to bit MSKEN3 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[10]	**MSKEN2**	**Enable PWM2 Mask Function Preset Bit** This bit will be load to bit MSKEN2 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[9]	**MSKEN1**	**Enable PWM1 Mask Function Preset Bit** This bit will be load to bit MSKEN1 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[8]	**MSKEN0**	**Enable PWM0 Mask Function Preset Bit** This bit will be load to bit MSKEN0 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[7:6]	**Reserved**	Reserved.
[5]	**MSKDAT5**	**Enable PWM5 Mask Data Preset Bit** This bit will be load to bit MSKDAT5 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.

Bits	Description	
[4]	**MSKDAT4**	**Enable PWM4 Mask Data Preset Bit** This bit will be load to bit MSKDAT4 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[3]	**MSKDAT3**	**Enable PWM3 Mask Data Preset Bit** This bit will be load to bit MSKDAT3 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[2]	**MSKDAT2**	**Enable PWM2 Mask Data Preset Bit** This bit will be load to bit MSKDAT2 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[1]	**MSKDAT1**	**Enable PWM1 Mask Data Preset Bit** This bit will be load to bit MSKDAT1 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.
[0]	**MSKDAT0**	**Enable PWM0 Mask Data Preset Bit** This bit will be load to bit MSKDAT0 in PHCHG_NOW when load trigger condition occurs. Refer to register PHCHG_NOW for detailed definition.

EPWM Phase Change Alternative Control Register (EPWM_PHCHGALT)

Register	Offset	R/W	Description	Reset Value
EPWM_PHCHGALT	EPWM_BA+0x80	R/W	EPWM Phase Change 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						**POSCTL1**	**POSCTL0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**POSCTL1**	**Positive Input Control for ACMP1** 0 = The input of ACMP1 is controlled by ACMP_CTL1. 1 = The input of ACMP1 is controlled by A1POSSEL in PHCHG_NOW register. **Note:** Register ACMP_CTL1 is describe in Comparator Controller chapter
[0]	**POSCTL0**	**Positive Input Control for ACMP0** 0 = The input of ACMP0 is controlled by ACMP_CTL0. 1 = The input of ACMP0 is controlled by A0POSSEL in PHCHG_NOW register. **Note:** Register ACMP_CTL0 is describe in Comparator Controller chapter

EPWM Period Interrupt Accumulation Control Register (EPWM_IFA)

Register	Offset	R/W	Description	Reset Value
EPWM_IFA	EPWM_BA+0x84	R/W	EPWM Period Interrupt Accumulation Control Register	0x0000_00F0

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**
IFDAT				**Reserved**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
IFCNT				**Reserved**			**IFAEN**

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:12]	**IFDAT**	**Period Interrupt Down-counter Data Register (Read Only)** When IFAEN is set, IFDAT will decrease when every PWM Interrupt flag is set, and when IFDAT reaches 0, the PWM interrupt will occurred and IFCNT will reload to IFDAT.
[11:8]	**Reserved**	Reserved.
[7:4]	**IFCNT**	**Period Interrupt Accumulation Counter Value Setting Register (Write Only)** 16 step Down-Counter value setting register. When IFAEN is set, IFCNT value will load into IFDAT and decrase gradually.
[3:1]	**Reserved**	Reserved.
[0]	**IFAEN**	**Enable Period Interrupt Accumulation Function** 0 = Period Interrupt Accumulation Disabled. 1 = Period Interrupt Accumulation Enabled.

6.9 Basic PWM Generator (BPWM)

6.9.1 Overview

The NDA102 series has one set of BPWM group supporting one set of PWM generator that can be configured as 2 independent PWM outputs, BPWM CH0~BPWM CH1, or as 1 complementary PWM pairs, (BPWM CH0, BPWM CH1) 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 DTCNT01(BPWM_CTL[4]) is set, BPWM CH0 and BPWM CH1 perform complementary; the paired PWM timing, period, duty and dead-time are determined by PWM0 timer and Dead-zone generator 0. Refer to Figure 6.9-1 PWM Clock Source Control

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 BPWM Counter Register(BPWM_PERIODx, x=0,1) 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.

6.9.2 Features

- 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

6.9.3 Block Diagram

Figure 6.9-1 shows PWM clock source control and Figure 6.9-2 illustrates the PWM architecture.

Figure 6.9-1 PWM Clock Source Control

Figure 6.9-2 PWM Architecture Diagram

6.9.4 PWM-Timer Operation

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

6.9.4.1 Edge-aligned PWM (down-counter)

In Edge-aligned PWM Output mode, the 16 bits PWM counter will starts down-counting from PERIOD (BPWM_PERIOD0-1[15:0]) to match with the value of the duty cycle CMP (BPWM_CMPDAT0-1[15:0]), 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 CMP and PERIOD are updated with CNTMODEn=1 and request the BPWM

interrupt if BPWM interrupt is enabled BPWM_INTEN(PWM_INTEN.n=1).

The PWM period and duty control are configured by BPWM counter register (BPWM_PERIOD0-1) and BPWM comparator register (BPWM_CMPDAT0-1). The PWM-timer timing operation is shown in Figure 6.9-4. The pulse width modulation follows the formula below and the legend of PWM-Timer Comparator is shown as Figure 6.9-3. Note that the corresponding GPIO pins must be configured as BPWM function when enable BPWM_POEN for the corresponding BPWM channel.

- PWM frequency = BPWM_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)].
- Duty ratio = (CMP+1)/(PERIOD+1)
- CMP >= PERIOD: PWM output is always high
- CMP < PERIOD: PWM low width= (PERIOD-CMP) unit[1]; PWM high width = (CMP+1) unit
- CMP = 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit

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

Note: x= 0~1.

Figure 6.9-3 Legend of Internal Comparator Output of PWM-Timer

Note: x = 0,1

Figure 6.9-4 PWM-Timer Operation Timing

Figure 6.9-5 PWM Edge-aligned Interrupt Generate Timing Waveform

6.9.4.2 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 CMP (BPWM_CMPDAT0-1[15:0]); this will cause the toggling of the PWMn generator output to low. The counter will continue counting to match with the PERIOD (BPWM_PERIOD0-1[15:0]) . Upon reaching this states counter is configured automatically to down counting, when PWM counter matches the CMP value again the PWMn generator output toggles to high. Once the PWM counter underflows it will update the PERIOD of PWM counter register and CMP of BPWM comparator register0-1 with CNTMODEn = 1, n= 0, 1.

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

- PWM frequency = BPWM_CLK/[(prescale+1)*(clock divider)*2(PERIOD+1)].
- Duty ratio = [(2 x CMP) + 1]/[2 x (PERIOD+1)]
- CMP > PERIOD: PWM output is always high
- CMP <= PERIOD: PWM low width= 2 x (PERIOD-CMP) + 1 unit[1]; PWM high width = (2 x CMP) + 1 unit

- CMP = 0: PWM low width = 2 x PERIOD + 1 unit; PWM high width = 1 unit

Figure 6.9-6 Center-aligned Type Output Waveform

Figure 6.9-7 PWM Center-aligned Interrupt Generate Timing Waveform

6.9.4.3 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 PERIOD (BPWM_PERIOD0-1) and current PWM counter value can be read from CNTx (BPWM_CNT0-1[15:0]).

PWM0 will operate in One-shot mode if CNTMODE0 bit is set to 0, and operate in Auto-reload mode if CNTMODE0 bit is set to 1. It is recommend that switch PWM0 operating mode before set CNTEN0 bit to 1 to enable PWM0 counter start running because the content of BPWM_PERIOD0 and BPWM_CMPDAT0 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, BPWM_CMPDAT0 and BPWM_PERIOD0 should be written first and then set CNTEN0 bit to 1 to enable PWM0 counter start running. After PWM0 counter down count from BPWM_PERIOD0 value to 0, BPWM_PERIOD0 and BPWM_CMPDAT0 will be cleared to 0 by hardware and PWM counter will be held. Software need to write new BPWM_CMPDAT0 and BPWM_PERIOD0 value to set next one-shot period and duty. When re-start next one-shot operation, the BPWM_CMPDAT0 should be written first because PWM0 counter will auto re-start counting when BPWM_PERIOD0 is written a non-zero value. As PWM0 operates at auto-reload mode, BPWM_CMPDAT0 and BPWM_PERIOD0 should be written first and then set CNTEN0 bit to 1 to enable PWM0 counter start running. The value of BPWM_PERIOD0 will reload to PWM0 counter when it down count reaches 0. If BPWM_PERIOD0 is set to 0, PWM0 counter will be held. PWM1 performs the same function as PWM0.

Figure 6.9-8 PWM Double Buffering Illustration

6.9.4.4 Modulate Duty Ratio

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

Figure 6.9-9 PWM Controller Output Duty Ratio

6.9.4.5 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 DTI01 (BPWM_ CLKPSC [23:16]) to determine the Dead-zone interval.

Figure 6.9-10 Paired-PWM Output with Dead-zone Generation Operation

6.9.4.6 PWM-Timer Interrupt Architecture

There are two PWM interrupts, BPWM0_INT and BPWM1_INT. Figure 6.9-11 demonstrates the architecture of PWM Timer interrupts.

Figure 6.9-11 PWM Interrupt Architecture Diagram

6.9.4.7 PWM-Timer Start Procedure

The following procedure is recommended for starting a PWM drive.

1. Set clock source divider select register (BPWM_CLKDIV)
2. Set prescaler (BPWM_CLKPSC)
3. Set inverter on/off, Dead-zone generator on/off, Auto-reload/One-shot mode and Stop PWM-timer (BPWM_CTL)
4. Set comparator register (BPWM_CMPDAT) for setting PWM duty.
5. Set PWM down-counter register (BPWM_PERIOD) for setting PWM period.
6. Set interrupt enable register (BPWM_INTEN) (optional)
7. Set corresponding GPIO pins as PWM function (enable BPWM_POEN) for the corresponding PWM channel.
8. Enable PWM timer start running (Set CNTENx = 1 in BPWM_CTL, x= 0 or 1)

6.9.4.8 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 (BPWM_CMPDAT) for setting PWM duty.
- Set PWM down-counter register (BPWM_PERIOD) for setting PWM period. After setting PERIOD, PWM wave will be generated.

6.9.4.9 PWM-Timer Stop Procedure

Method 1:

Set 16-bit counter (PERIOD) as 0, and monitor CNT (current value of 16-bit down-counter). When CNT reaches to 0, disable PWM-Timer (CNTENx in BPWM_CTL, x= 0 or 1). ***(Recommended)***

Method 2:

Set 16-bit counter (PERIOD) as 0. When interrupt request happened, disable PWM-Timer (CNTENx in BPWM_CTL, x= 0 or 1). ***(Recommended)***

Method 3:

Disable PWM-Timer directly ((CNTENx in BPWM_CTL, x= 0 or 1). ***(Not recommended)***

The reason why method 3 is not recommended is that disable CNTENx 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

6.9.5 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
BPWM Base Address: **BPWM_BA = 0x4014_0000**				
BPWM_CLKPSC	BPWM_BA+0x00	R/W	Basic PWM Pre-scalar Register	0x0000_0000
BPWM_CLKDIV	BPWM_BA+0x04	R/W	Basic PWM Clock Source Divider Select Register	0x0000_0000
BPWM_CTL	BPWM_BA+0x08	R/W	Basic PWM Control Register	0x0000_0000
BPWM_PERIOD0	BPWM_BA+0x0C	R/W	Basic PWM Period Counter Register 0	0x0000_0000
BPWM_CMPDAT0	BPWM_BA+0x10	R/W	Basic PWM Comparator Register 0	0x0000_0000
BPWM_CNT0	BPWM_BA+0x14	R	Basic PWM Data Register 0	0x0000_0000
BPWM_PERIOD1	BPWM_BA+0x18	R/W	Basic PWM Period Counter Register 1	0x0000_0000
BPWM_CMPDAT1	BPWM_BA+0x1C	R/W	Basic PWM Comparator Register 1	0x0000_0000
BPWM_CNT1	BPWM_BA+0x20	R	Basic PWM Data Register 1	0x0000_0000
BPWM_INTEN	BPWM_BA+0x40	R/W	Basic PWM Interrupt Enable Register	0x0000_0000
BPWM_INTSTS	BPWM_BA+0x44	R/W	Basic PWM Interrupt Indication Register	0x0000_0000
BPWM_POEN	BPWM_BA+0x7C	R/W	Basic PWM Output Enable	0x0000_0000

6.9.6 Register Description

BPWM Pre-scale Register (BPWM_CLKPSC)

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

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
DTI01							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CLKPSC01							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**DTI01**	**Dead-zone Interval for Pair of Channel 0 and Channel 1** These 8-bit determine the Dead-zone length. The unit time of Dead-zone length = [(prescale+1)*(clock source divider)] / BPWM_CLK.
[15:8]	**Reserved**	Reserved.
[7:0]	**CLKPSC01**	**Clock Prescaler** Clock input is divided by (CLKPSC01 + 1) before it is fed to the corresponding PWM-timer If CLKPSC01=0, then the clock prescaler 0 output clock will be stopped. So corresponding PWM-timer will also be stopped.

BPWM Clock Source Divider Select Register (BPWM_CLKDIV)

Register	Offset	R/W	Description	Reset Value
BPWM_CLKDIV	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	**CLKDIV1**			**Reserved**	**CLKDIV0**		

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**CLKDIV1**	**PWM Timer 1 Clock Source Divider Selection** Select clock source divider for PWM timer 1. 000 = 1/2. 001 = 1/4. 010 = 1/8. 011 = 1/16. 100 = 1.
[3]	**Reserved**	Reserved.
[2:0]	**CLKDIV0**	**PWM Timer 0 Clock Source Divider Selection** Select clock source divider for PWM timer 0. (Table is the same as CLKDIV1)

BPWM Control Register (BPWM_CTL)

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

31	30	29	28	27	26	25	24
Reserved	**CNTTYPE01**	**Reserved**					
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**CNTMODE1**	**CMPINV1**	**PINV1**	**CNTEN1**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**DTCNT01**	**CNTMODE0**	**CMPINV0**	**PINV0**	**CNTEN0**

Bits	Description	
[31]	**Reserved**	Reserved.
[30]	**CNTTYPE01**	**PWM01 Aligned Type Selection** 0 = Edge-aligned type. 1 = Center-aligned type.
[29:12]	**Reserved**	Reserved.
[11]	**CNTMODE1**	**PWM-timer 1 Auto-reload/One-shot Mode** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause BPWM_PERIOD1 and BPWM_CMPDAT1 be cleared.
[10]	**CMPINV1**	**PWM-timer 1 Output Inverter Enable Bit** 0 = Inverter Disabled. 1 = Inverter Enabled.
[9]	**PINV1**	**PWM-timer 1 Output Polar Inverse Enable Bit** 0 = PWM1 output polar inverse Disabled. 1 = PWM1 output polar inverse Enabled.
[8]	**CNTEN1**	**PWM-timer 1 Enable Bit** 0 = Corresponding PWM-Timer Stopped. 1 = Corresponding PWM-Timer Start Running.
[7:5]	**Reserved**	Reserved.
[4]	**DTCNT01**	**Dead-zone 0 Generator Enable Bit** 0 = Dead-zone 0 Generator Disabled. 1 = Dead-zone 0 Generator Enabled. **Note:** When Dead-zone generator is enabled, the pair of PWM0 and PWM1 becomes a complementary pair.

[3]	**CNTMODE0**	**PWM-timer 0 Auto-reload/One-shot Mode** 0 = One-shot mode. 1 = Auto-reload mode. **Note:** If there is a transition at this bit, it will cause BPWM_PERIOD0 and BPWM_CMPDAT0 be cleared.
[2]	**CMPINV0**	**PWM-timer 0 Output Inverter Enable Bit** 0 = Inverter Disabled. 1 = Inverter Enabled.
[1]	**PINV0**	**PWM-timer 0 Output Polar Inverse Enable Bit** 0 = PWM0 output polar inverse Disabled. 1 = PWM0 output polar inverse Enabled.
[0]	**CNTEN0**	**PWM-timer 0 Enable Bit** 0 = The corresponding PWM-Timer stops running. 1 = The corresponding PWM-Timer starts running.

BPWM Counter Register 0-1 (BPWM_PERIOD0-1)

Register	Offset	R/W	Description	Reset Value
BPWM_PERIOD0	BPWM_BA+0x0C	R/W	Basic PWM Period Counter Register 0	0x0000_0000
BPWM_PERIOD1	BPWM_BA+0x18	R/W	Basic PWM Period 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**
PERIOD							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
PERIOD							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**PERIOD**	**Basic PWM Period Counter Register** PERIOD data determines the PWM period. For Edge-aligned type: PWM frequency = BPWM_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)]. ● Duty ratio = (CMP+1)/(PERIOD+1). ● CMP >= PERIOD: PWM output is always high. ● CMP < PERIOD: PWM low width = (PERIOD-CMP) unit; PWM high width = (CMP+1) unit. ● CMP = 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit. For Center-aligned type: PWM frequency = BPWM_CLK/[(prescale+1)*(clock divider)*2(PERIOD+1)]. ● Duty ratio = [(2 x CMP) + 1]/[2 x (PERIOD+1)]. ● CMP > PERIOD: PWM output is always high. ● CMP <= PERIOD: PWM low width = 2 x (PERIOD-CMP) + 1 unit; PWM high width = (2 x CMP) + 1 unit. ● CMP = 0: PWM low width = 2 x PERIOD + 1 unit; PWM high width = 1 unit. (Unit = one PWM clock cycle). **Note:** Any write to PERIOD will take effect in next PWM cycle. **Note:** When PWM operating at Center-aligned type, PERIOD value should be set between 0x0000 to 0xFFFE. If PERIOD equal to 0xFFFF, the PWM will work unpredictable. **Note:** When PERIOD value is set to 0, PWM output is always high.

BPWM Comparator Register0-1 (BPWM_CMPDAT0-1)

Register	Offset	R/W	Description	Reset Value
BPWM_CMPDAT0	BPWM_BA+0x10	R/W	Basic PWM Comparator Register 0	0x0000_0000
BPWM_CMPDAT1	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**
CMP							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CMP							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CMP**	**PWM Comparator Register** CMP determines the PWM duty. PWM frequency = BPWM_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)]. For Edge-aligned type: PWM frequency = BPWM_CLK/[(prescale+1)*(clock divider)*(PERIOD+1)]. ● Duty ratio = (CMP+1)/(PERIOD+1). ● CMP >= PERIOD: PWM output is always high. ● CMP < PERIOD: PWM low width = (PERIOD-CMP) unit; PWM high width = (CMP+1) unit. ● CMP = 0: PWM low width = (PERIOD) unit; PWM high width = 1 unit. For Center-aligned type: PWM frequency = BPWM_CLK/[(prescale+1)*(clock divider)*2(PERIOD+1)]. ● Duty ratio = [(2 x CMP) + 1]/[2 x (PERIOD+1)]. ● CMP > PERIOD: PWM output is always high. ● CMP <= PERIOD: PWM low width = 2 x (PERIOD-CMP) + 1 unit; PWM high width = (2 x CMP) + 1 unit. ● CMP = 0: PWM low width = 2 x PERIOD + 1 unit; PWM high width = 1 unit. (Unit = one PWM clock cycle). **Note:** Any write to PERIOD will take effect in next PWM cycle.

BPWM Data Register 0-1 (BPWM_CNT0-1)

Register	Offset	R/W	Description	Reset Value
BPWM_CNT0	BPWM_BA+0x14	R	Basic PWM Data Register 0	0x0000_0000
BPWM_CNT1	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**
CNT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CNT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**CNT**	**PWM Data Register** User can monitor CNT to know the current value in 16-bit counter.

BPWM Interrupt Enable Register (BPWM_INTEN)

Register	Offset	R/W	Description	Reset Value
BPWM_INTEN	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							**PINTTYPE**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**DIEN1**	**DIEN0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**PIEN1**	**PIEN0**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16]	**PINTTYPE**	**BPWM Interrupt Period Type Selection** 0 = PIFn will be set if BPWM counter underflow. 1 = PIFn will be set if BPWM counter matches PERIODn register. **Note:** This bit is effective when BPWM in Center-aligned type only.
[15:10]	**Reserved**	Reserved.
[9]	**DIEN1**	**BPWM Channel 1 Duty Interrupt Enable Bit** 0 = BPWM Channel 1 Duty Interrupt Disabled. 1 = BPWM Channel 1 Duty Interrupt Enabled.
[8]	**DIEN0**	**BPWM Channel 0 Duty Interrupt Enable Bit** 0 = BPWM Channel 0 Duty Interrupt Disabled. 1 = BPWM Channel 0 Duty Interrupt Enabled.
[7:2]	**Reserved**	Reserved.
[1]	**PIEN1**	**BPWM Channel 1 Period Interrupt Enable Bit** 0 = BPWM Channel 1 Period Interrupt Disabled. 1 = BPWM Channel 1 Period Interrupt Enabled.
[0]	**PIEN0**	**BPWM Channel 0 Period Interrupt Enable Bit** 0 = BPWM Channel 0 Period Interrupt Disabled. 1 = BPWM Channel 0 Period Interrupt Enabled.

BPWM Interrupt Indication Register (BPWM_INTSTS)

Register	Offset	R/W	Description	Reset Value
BPWM_INTSTS	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						**DIF1**	**DIF0**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved						**PIF1**	**PIF0**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9]	**DIF1**	**BPWM Channel 1 Duty Interrupt Flag** Flag is set by hardware when channel 1 BPWM counter down count and reaches BPWM_CMPDAT 1, software can clear this bit by writing a one to it. **Note:** If CMP equal to PERIOD, this flag is not working in Edge-aligned type selection
[8]	**DIF0**	**BPWM Channel 0 Duty Interrupt Flag** Flag is set by hardware when channel 0 BPWM counter down count and reaches BPWM_CMPDAT 0, software can clear this bit by writing a one to it. **Note:** If CMP equal to PERIOD, this flag is not working in Edge-aligned type selection
[7:2]	**Reserved**	Reserved.
[1]	**PIF1**	**BPWM Channel 1 Period Interrupt Status** This bit is set by hardware when BPWM1 counter reaches the requirement of interrupt (depend on PINTTYPE bit of PWM_INTEN register), software can write 1 to clear this bit to 0.
[0]	**PIF0**	**BPWM Channel 0 Period Interrupt Status** This bit is set by hardware when BPWM0 counter reaches the requirement of interrupt (depend on PINTTYPE bit of PWM_INTEN 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 BPWM_INTSTS.

BPWM Output Enable Register (BPWM_POEN)

Register	Offset	R/W	Description	Reset Value
BPWM_POEN	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						**POEN1**	**POEN0**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**POEN1**	**Channel 1 Output Enable Register** 0 = BPWM channel 1 output to pin Disabled. 1 = BPWM channel 1 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to BPWM function
[0]	**POEN0**	**Channel 0 Output Enable Register** 0 = BPWM channel 0 output to pin Disabled. 1 = BPWM channel 0 output to pin Enabled. **Note:** The corresponding GPIO pin must also be switched to BPWM function

6.10 Watchdog Timer (WDT)

6.10.1 Overview

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

6.10.2 Features

- 18-bit free running up counter for Watchdog Timer time-out interval
- Selectable time-out interval (24 ~ 218) WDT_CLK cycle and the time-out interval period is 104 ms ~ 26.3168 s if WDT_CLK = 10 kHz
- System kept in reset state for a period of (1 / WDT_CLK) * 63
- Supports Watchdog Timer time-out wake-up function only if WDT clock source is selected as 10 kHz

6.10.3 Block Diagram

Note1: WDT resets CPU and lasts 63 WDT_CLK.

Note2: If user intends to use WDT to wake-up Idle/Power-down mode, it is recommended that CPU clock source is set the same as WDT clock source before CPU enters Power-down mode.

Note3: The WDT reset delay period is fixed to 1024 WDT_CLK

Figure 6.10-1 Watchdog Timer Block Diagram

6.10.4 Clock Control

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

Figure 6.10-2 Watchdog Timer Clock Control Diagram

6.10.5 Basic Configuration

The WDT peripheral clock is enabled in WDTKEN (CLK_APBCLK[0]) and clock source can be selected in WDTSEL (CLK_CLKSEL1[1:0]).

6.10.6 Functional Description

The Watchdog Timer (WDT) includes an 18-bit free running up counter with programmable time-out intervals. Figure 6.10-3 shows the WDT time-out interval and reset period timing.

6.10.6.1 WDT Time-out Interrupt

Setting WDTEN (WDTCR[7]) bit to 1 will enable the WDT function and the WDT counter to start counting up. There are eight time-out interval period can be selected by setting TOUTSEL (WDT_CTL[10:9]). When the WDT up counter reaches the TOUTSEL ((WDT_CTL[10:9]) settings, WDT time-out interrupt will occur then IF (WDT_CTL[3]) flag will be set to 1 immediately.

6.10.6.2 WDT Reset Delay Period and Reset System

There is a specified T_{RSTD} delay period follows the IF (WDT_CTL[3]) flag is setting to 1. User must enabled RSTCNT (WDT_CTL[0]) bit to reset the 18-bit WDT up counter value to avoid generate WDT time-out reset signal before the T_{RSTD} delay period expires. If the WDT up counter value has not been cleared after the specific T_{RSTD} delay period expires, the WDT control will set RSTF (WDT_CTL[2]) flag to 1 if RSTEN(WDT_CTL[1]) bit is enabled, then chip enters to reset state immediately. Refer to Figure 6.10-3, the T_{RST} reset period will keep last 63 WDT clocks then chip restart executing program from reset vector (0x0000_0000). The RSTF (WDT_CTL[2]) flag will keep 1 after WDT time-out reset the chip, user can check RSTF(WDT_CTL[2]) flag by software to recognize the system has been reset by WDT time-out reset or not.

6.10.6.3 WDT Wake-up

If WDT clock source is selected to 10 kHz, system can be woken-up from Power-down mode while WDT time-out interrupt signal is generated and WKEN (WDT_CTL[4]) bit enabled. In the meanwhile, the WKF (WDT_CTL[5]) flag will set to 1 automatically, and user can check WKF (WDT_CTL[5]) flag by software to recognize if the system has been woken-up by WDT time-out interrupt or not.

TOUTSEL	Time-Out Interval Period T_{TIS}	Reset Delay Period T_{RSTD}
000	$2^4 * T_{WDT}$	$1024 * T_{WDT}$
001	$2^6 * T_{WDT}$	$1024 * T_{WDT}$
010	$2^8 * T_{WDT}$	$1024 * T_{WDT}$
011	$2^{10} * T_{WDT}$	$1024 * T_{WDT}$
100	$2^{12} * T_{WDT}$	$1024 * T_{WDT}$
101	$2^{14} * T_{WDT}$	$1024 * T_{WDT}$
110	$2^{16} * T_{WDT}$	$1024 * T_{WDT}$
111	$2^{18} * T_{WDT}$	$1024 * T_{WDT}$

Table 6.10-1 Watchdog Timer Time-out Interval Period Selection

- T_{WDT} : *Watchdog Clock Time Period*
- T_{TIS} : *Watchdog Time out Interval Period* ($(2^4 \sim 2^{18}) * T_{WDT}$)
- T_{RSTD} : *Watchdog Reset Delay Period - Fixed 1024* T_{WDT} delay period*
- T_{RST} : *Watchdog Reset Period* ($63 * T_{WDT}$)

Figure 6.10-3 Watchdog Timer Time-out Interval and Reset Period Timing

6.10.7 Registers Map

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

Register	Offset	R/W	Description	Reset Value
WDT Base Address: **WDT_BA = 0x4000_4000**				
WDT_CTL	WDT_BA+0x00	R/W	Watchdog Timer Control Register	0x0000_0700

6.10.8 Register Description

Watchdog Timer Control Register (WDT_CTL)

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

Note: All bits in this register are write-protected. To program it, an open lock sequence is needed, by sequentially writing 0x59, 0x16, and 0x88 to register REGWRPROT at address SYS_BA + 0x100.

31	30	29	28	27	26	25	24
ICEDEBUG	**Reserved**						
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved					**TOUTSEL**		
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
WDTEN	**INTEN**	**WKF**	**WKEN**	**IF**	**RSTF**	**RSTEN**	**RSTCNT**

Bits	Description	
[31]	**ICEDEBUG**	**ICE Debug Mode Acknowledge Disabl Bit (Write Protect)** 0 = ICE debug mode acknowledgement effects WDT counting. WDT up counter will be kept while CPU is hanging by ICE. 1 = ICE debug mode acknowledgement Disabled. WDT up counter will keep going no matter CPU is hanging by ICE or not.
[30:11]	**Reserved**	Reserved.
[10:8]	**TOUTSEL**	**Watchdog Timer Interval Selection** These three bits select the time-out interval for the Watchdog Timer. 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]	**WDTEN**	**Watchdog Timer Enable Bit (Write Protect)** 0 = WDT Disabled. (This action will reset the internal up counter value.) 1 = WDT Enabled.
[6]	**INTEN**	**Watchdog Timer Time-out Interrupt Enable Bit (Write Protect)** If this bit is enabled, the WDT time-out interrupt signal is generated and inform to CPU. 0 = WDT time-out interrupt Disabled. 1 = WDT time-out interrupt Enabled.

Bits	Description	
[5]	**WKF**	**Watchdog Timer Time-out Wake-up Flag** This bit indicates the interrupt wake-up flag status of WDT. 0 = WDT does not cause chip wake-up. 1 = Chip wake-up from Idle or Power-down mode if WDT time-out interrupt signal generated. **Note:** This bit is cleared by writing 1 to it.
[4]	**WKEN**	**Watchdog Timer Time-out Wake-up Function Control (Write Protect)** If this bit is set to 1, while IF is generated to 1 and INTEN enabled, the WDT time-out interrupt signal will generate a wake-up trigger event to chip. 0 = Wake-up trigger event Disabled if WDT time-out interrupt signal generated. 1 = Wake-up trigger event Enabled if WDT time-out interrupt signal generated. **Note:** Chip can be woken-up by WDT time-out interrupt signal generated only if WDT clock source is selected to 10 kHz oscillator.
[3]	**IF**	**Watchdog Timer Time-out Interrupt Flag** This bit will be set to 1 while WDT up counter value reaches the selected WDT time-out interval. 0 = WDT time-out interrupt did not occur. 1 = WDT time-out interrupt occurred. **Note:** This bit is cleared by writing 1 to it.
[2]	**RSTF**	**Watchdog Timer Time-out Reset Flag** This bit indicates the system has been reset by WDT time-out reset or not. 0 = WDT time-out reset did not occur. 1 = WDT time-out reset occurred. **Note:** This bit is cleared by writing 1 to it.
[1]	**RSTEN**	**Watchdog Timer Time-out Reset Enable Bit (Write Protect)** Setting this bit will enable the WDT time-out reset function if the WDT up counter value has not been cleared after the specific WDT reset delay period (1024 * T_{WDT}) expires. 0 = WDT time-out reset function Disabled. 1 = WDT time-out reset function Enabled.
[0]	**RSTCNT**	**Reset Watchdog Timer Up Counter (Write Protect)** 0 = No effect. 1 = Reset the internal 18-bit WDT up counter value. **Note:** This bit will be automatically cleared by hardware.

6.11 USCI – Universal Serial Control Interface Controller

6.11.1 Overview

The Universal Serial Control Interface (USCI) is a flexible interface module covering several serial communication protocols. The user can configure this controller as UART, SPI, or I^2C functional protocol.

Note: For detailed USCI UART, I^2C and SPI information, please refer to section 6.12, 6.13 and 6.14.

6.11.2 Features

The controller can be individually configured to match the application needs. The following protocols are supported:

- UART
- SPI
- I^2C

To increase readability, the registers of USCI have different alias names that depending on the selected protocol. For example, register USCI_CTL has alias name UUART_CTL for protocol UART, has alias name USPI_CTL for protocol SPI, and has alias name UI2C_CTL for protocol I^2C.

6.11.3 Block Diagram

The basic configurations of USCI are as Figure 6.11-1.

Figure 6.11-1 USCI Block Diagram

6.11.4 Functional Description

The structure of the Universal Serial Control Interface (USCI) controller is shown in Figure 6.11-1. The input signal is implemented in an input processor. The data buffers and the data shift unit support the data transfers. Each protocol-specific function is handled by the protocol processor unit. The timing and time event control signals of the specific protocol are handled by the protocol-relative clock generator. All the protocol-specific events are processed in the interrupt generation unit. The wake-up function of the specific protocol is implemented in the wake-up control unit.

The USCI is equipped with three protocols including UART, SPI, and I^2C. They can be selected by FUNMODE (USCI_CTL [2:0]). Note that the FUNMODE must be set to 0 before changing protocol.

6.11.4.1 I/O Processer

Input Signal

All input stages offer the similar feature set. They are used for all protocols.

Table 6.11-1 lists the relative input signals for each selected protocol. Each input signal is handled by an input processor for signal conditioning, such as signal inverse selection control, or a digital input filter.

Selected Protocol		**UART**	**SPI**	**I^2C**
Serial Bus Clock Input	USCIx_CLK	-	SPIx_CLK	I2Cx_SCL
Control Input	USCIx_CTL0	-	SPIx_SS	-
	USCIx_CTL1	-	-	-
Data Input	USCIx_DAT0	UARTx_RX	SPIx_MOSI	I2Cx_SDA
	USCIx_DAT1	-	SPxI_MISO	-

Table 6.11-1 Input Signals for Different Protocols

Note1: x = 0, 1

Note2: The description of protocol-specific items are given in the related protocol chapters.

General Input Structure

The input structures of data and control signals include inverter, digital filter and edge detection (data signal only).

Figure 6.11-2 Input Conditioning for USCIx_DAT[1:0] and USCIx_CTL[1:0]

The input structure of USCIx_CLK is similar to USCIx_CTL[1:0] input structure, except it does not support inverse function.

Figure 6.11-3 Input Conditioning for USCIx_CLK

All configurations of control, clock and data input structures are in USCI_CTLIN0, USCI_CLKIN and USCI_DATIN0 registers respectively. EDGEDET (USCI_DATIN0[4:3]) is used to select the edge detection condition. Note that the EDGEDET for USCI_DATIN0 must be set to 2'b10 in UART mode. The programmable edge detection indicates that the desired event has occurred by activating the trigger signal.

ININV (USCI_DATIN0[2] / USCI_CTLIN0[2]) allows a polarity inversion of the selected input signal to adapt the input signal polarity to the internal polarity of the data shift unit and the protocol state machine.

If the SYNCSEL (USCI_DATIN0[0] / USCI_CTLIN0[0] / USCI_CLKIN[0]) is set to 0, the paths of input signals do not contain any delay due to synchronization or filtering. If there is noise on the input signals, there is the possibility to synchronize the input signal (signal IN_SYNC is

synchronized to f_{PCLK}). The synchronized input signal is taken into account by SYNCSEL = 1. The synchronization leads to a delay in the signal path of 2-3 times the period of f_{PCLK}.

Output Signals

Table 6.11-2 shows the relative output signals for each protocol. The number of actually used outputs depends on the selected protocol and they can be classified according to their meaning for the protocols.

Selected Protocol		UART	SPI	I^2C
Serial Bus Clock Output	USCIx_CLK	-	SPIx_CLK	I2Cx_SCL
Control Output	USCIx_CTL0	-	SPIx_SS	-
	USCIx_CTL1	-	-	-
Data Output	USCIx_DAT0	-	SPIx_MOSI	I2Cx_SDA
	USCIx_DAT1	UARTx_TX	SPIx_MISO	-

Table 6.11-2 Output Signals for Different Protocols

Note1: x = 0, 1

Note2: The description of protocol-specific items are given in the related protocol chapters.

6.11.4.2 Data Buffering

The data handling of the USCI controller is based on a Data Shift Unit (DSU) and a buffer structure. Both of the data shift and buffer registers are 16-bit wide. The inputs of Data Shift Unit include the shift data, the serial bus clock, and the shift control. The output pin of transmission can be USCIx_DAT0 pin or USCIx_DAT1 pin depends on what protocol is selected.

Figure 6.11-4 Block Diagram of Data Buffering

The operation of data handling includes:

- The peripheral device user interface (APB) is used to handle data, interrupts, status and control information.
- A transmitter includes transmit shift register (TX_SFTR) and a transmit data buffer (TX_BUF). The TXFULL (USCI_BUFSTS[9], TXEMPTY (USCI_BUFSTS[8]) and TXENDIF (USCI_PROTSTS[2]) can indicate the status of transmitter.
- A receiver includes receive shift register (RX_SFTR) and a double receive buffer structure (RX_BUF0, RX_BUF1). In double buffer structure, user does not need to consider the reception sequence and two received data can be hold if user does not read the data of USCI_RXDAT register in time.

Data Access Structure

The Data Access Structure includes read access to received data and write access of data to be transmitted. The received data is stored in the receiver buffers including RX_BUF0 and RX_BUF1. User need no care about the reception sequence. The receive buffer can be accessed by reading USCI_RXDAT register. The first received data is read out first and the next received data becomes visible in USCI_RXDAT and can be read out next.

Transmitted data can be loaded to TX_BUF by writing to the transmit register USCI_TXDAT.

Figure 6.11-5 Data Access Structure

Transmit Data Path

The transmit data path is based on 16-bit wide transmit shift register (TX_SFTR) and transmit buffer TX_BUF. The data transfer parameters like data word length is controlled commonly for transmission and reception by the line control register USCI_LINECTL.

Transmit Buffering

The transmit shift register cannot be directly accessed by user. It is updated automatically with the value stored in the transmit buffer (TX_BUF) if a currently transmitted data is finished and new

data is valid for transmission.

Figure 6.11-6 Transmit Data Path

Transmit Data Validation

The status of TXEMPTY (USCI_BUFSTS[8]) indicates the transmission data is valid or not in the transmit buffer (TX_BUF) and the TXSTIF (USCI_PROTSTS[1]) labels the start conditions for each data.

- If the USCI controller is a Master, the data transfer can only be started with valid data in the transmit buffer (TX_BUF). In this case, the transmit shift register is loaded with the content of transmit buffer.

  Note: Master defines the start of data transfer.

- If the USCI controller is a Slave, a data transfer requested by Master and it has to be started independently of the status in transmit buffer (TX_BUF). If a data transfer is requested and started by the Master, the transmit shift register is loaded from specific protocol control signal if it is valid for transmission.

  Note: Slave cannot define the start itself, but has to react.

- The timing of loading data from transmit buffer to data shift unit depends on protocol configurations.

  UART: A transmission of the data word in transmit buffer can be started if TXEMPTY = 0 in normal operation.

  SPI: In Master mode, data transmission will be started when TXEMPTY (USCI_BUFSTS[8]) is 0. In Slave mode, the data transmission can be started only when slave selection signal is at active state and clock is presented on USCIx_CLK pin.

  I^2C: A transmission of the data byte in transmit buffer can be started if TXEMPTY = 0.

- A transmission data which is located in transmit buffer can be started if the TXEMPTY (USCI_BUFSTS [8]) = 0. The content of the transmit buffer (in TX_BUF condition) should not be overwritten with new data while it is valid for transmission and a new transmission can start. If the content of TX_BUF has to be changed, user can set TXRST (USCI_BUFCTL [16]) to 1 to clear the content of TX_BUF before updating the data. Moreover, TXEMPTY (USCI_BUFSTS [8]) will be cleared automatically when transmit buffer (TX_BUF) is updated with new data. While a transmission is in progress, TX_BUF can be loaded with new data. User has to update the TX_BUF before a new transmission.

Receive Data Path

The receive data path is based on 16-bit wide receive shift register RX_SFTR and receive buffers RX_BUF0 and RX_BUF1. The data transfer parameters like data word length, or the shift direction are controlled commonly for transmission and reception by the line control register USCI_LINECTL. Register USCI_BUFSTS monitors the data validation of USCI_RXDAT.

Receive Buffering

The receive shift register cannot be directly accessed by user, but its content is automatically loaded into the receive buffer if a complete data word has been received or the frame is finished. The received data words in Receive Buffer can be read out automatically from register USCI_RXDAT.

Figure 6.11-7 Receive Data Path

6.11.4.3 Protocol Control and Status

The protocol-related control and status information are located in the protocol control register USCI_PROTCTL and in the protocol status register USCI_PROTSTS. These registers are shared between the available protocols. As a consequence, the meaning of the bit positions in these registers is different within the protocols. Refer to each protocol's relative register for detail information.

6.11.4.4 Protocol-Relative Clock Generator

USCI controller contains a protocol-relative clock generator and it is controlled by register USCI_BRGEN. It is reset when the USCI_BRGEN register is written. The structured of protocol-relative clock generator is shown in Figure 6.11-8.

Note: Refer the **Basic Clock Divider Counter** section to get the f_{SAMP_CLK}

Figure 6.11-8 Protocol-Relative Clock Generator

The protocol related counter contains basic clock divider counter and timing measurement counter. It is based on a divider stages, providing the frequencies needed for the different protocols. It contains:

- The external clock input to generate the input frequency f_{REF_CLK} = f_{ECLK} for baud rate generation based on an external signal. Note that the external clock is half of system clock frequency because the external clock is sampled by system clock.
- The basic clock divider counter provides the protocol relative clock signal and other protocol-related signals (f_{SAMP_CLK} and f_{DS_CLK}).
- The timing measurement counter for time interval measurement, e.g. baud rate detection on UART protocol.
- The output signals of protocol relative clock generator can be made available on pins (e.g USCIx_CLK for SPI).

Basic Clock Divider Counter

The basic clock divider counter is used for an integer division delivering f_{REF_CLK2}, f_{REF_CLK}, f_{DIV_CLK}, f_{SCLK},and f_{SAMP_CLK} . The frequencies of this divider are controlled by PTCLKSEL (USCI_BRGEN [1]), CLKDIV (USCI_BRGEN [25:16]), SPCLKSEL (USCI_BRGEN [3:2]).

The basic clock divider counter is used to generate the relative protocol timing signals.

$$f_{\mathrm{DIV_CLK}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV} + 1} \text{ if } \mathrm{PTCLKSEL} = 0$$

$$f_{\mathrm{DIV_CLK}} = f_{REF_CLK} \times \frac{1}{(\mathrm{CLKDIV} + 1) \times 2} \text{ if } \mathrm{PTCLKSEL} = 1$$

Figure 6.11-9 Basic Clock Divider Counter

Timing Measurement Counter

The timing measurement counter is used for time interval measurement and is enabled by TMCNTEN (USCI_BRGEN [4]) = 1. When TMCNTSRC (USCI_BRGEN [5]) is set to 1, the timer works on f_{DIV_CLK}, otherwise, the timer works independently from f_{PROT_CLK}. Therefore, any serial data reception or transmission can continue while the timer is performing timing measurements. The timer counts the length of protocol-related signals with f_{PROT_CLK} or f_{DIV_CLK}. It stops counting when it reaches the user-specified value.

Figure 6.11-10 Block of Timing Measurement Counter

The timing measurement counter is used to perform time-out function or auto-baud rate mechanism. Its functionality depends on the selected protocol as shown below.

- **UART:** The timing measurement counter is used in auto baud rate detection.
- **SPI:** The timing measurement counter is used for counting the slave time-out period.

- **I²C:** The timing measurement counter indicates time-out clock cycle.

Sample Time Counter

A sample time counter associated to the protocol related counter defining protocol specific timings, such shift control signals or bit timings, based on the input frequency f_{SAMP_CLK}. The sample time counter allows generating time intervals for protocol-specific purposes. The period of a sample frequency f_{PDS_CNT} is given by the selected input frequency f_{SAMP_CLK} and the programmed pre-divider value (PDSCNT (USCI_BRGEN [9:8])). The meaning of the sample time depends on the selected protocol. Please refer to the corresponding chapters for more protocol-specific information.

Figure 6.11-11 Sample Time Counter

6.11.4.5 Data Transfer Events and Interrupts

The data transfer events are based on the transmission or reception of a data word. The related indication flags are located in register USCI_PROTSTS. All events can be individually enabled for interrupt generation. If the FUNMODE (USCI_CTL [2:0]) is set to 0, the USCI is disabled. When FUNMODE (USCI_CTL [2:0]) is set for a protocol port, the internal states will be controlled by logic hardware of the selected protocol.

- Transmit start interrupt event to indicate that a data word has been started:

  A transmit start interrupt event occurs when the data is loaded into transmitted shift register. It is indicated by flag TXSTIF (USCI_PROTSTS [1]) and, if enabled, leads to transmit start interrupt.

- Transmit end interrupt event to indicate that a data word transmission has been done:

  A transmit end interrupt event occurs when the current transmit data in shift register had finished. It is indicated by flag TXENDIF (USCI_PROTSTS [2]) and, if enabled, leads to transmit end interrupt. This event also indicates when the shift control settings (word length, shift direction, etc.) are internally "frozen" for the current data word transmission. In UART and I²C mode, the transmit data valid is according to TXEMPTY (USCI_BUFSTS [8]) and protocol relative internal signal with the transmit end interrupt event.

- Receiver start event to indicate that a data word reception has started:

  When the receive clock edge that shifts in the first bit of a new data word is detected and reception is enabled, a receiver start event occurs. It is indicated by flag RXSTIF (USCI_PROTSTS [3]) and, if enabled, leads to receiver start interrupt.

- Receive event to indicate that a data word has been received:

  If a new received word becomes available in the receive buffer, a receive event occurs. It is indicated by flag RXENDIF (USCI_PROTSTS [4]) and, if enabled, leads to receive interrupt.

- Data lost event to indicate a loss of the newest received data word:

  If the data word available in register USCI_RXDAT (oldest data word from RX_BUF0 or RX_BUF1) has not been read out and the receive buffer is FULL, the new incoming data will lose and this event occurs. It is indicated by flag RXOVIF (USCI_BUFSTS[3]) and, if enabled, leads to a protocol interrupt.

  The general event and interrupt structure is shown in Figure 6.11-12

Figure 6.11-12 Event and Interrupt Structure

Each general interrupt enable can set by RXENDIEN, RXSTIEN, TXENDIEN, and TXSTIEN of USCI_INTEN [4:1]. The events are including receive end interrupt event, receive start interrupt event, transmit end interrupt event, and transmit start interrupt event. For protocol-specific interrupt, it is specified in each protocol interrupt enable register.

If a defined condition is met, an event is detected and an event indication flag becomes automatically set. The flag stays set until it is cleared by software. If enabled, an interrupt can be generated if an event is detected.

The registers, bits and bit fields indicate the data transfer events and control the general interrupts of a USCI are shown in Table 6.11-3

Event	Indication Flag	Indication Cleared By	Interrupt Enabled By
Transmit start interrupt event	TXSTIF (USCI_PROTSTS [1])	It is cleared by software writes 1 to corresponding interrupt bit of USCI_PROTSTS.	TXSTIEN (USCI_INTEN [1])
Transmit end interrupt event	TXENDIF (USCI_PROTSTS [2])		TXENDIEN (USCI_INTEN [2])
Receive start interrupt event	RXSTIF (USCI_PROTSTS [3])		RXSTIEN (USCI_INTEN [3])
Receive end interrupt event	RXENDIF (USCI_PROTSTS [4])		RXENDIEN (USCI_INTEN [4])

Table 6.11-3 Data Transfer Events and Interrupt Handling

6.11.4.6 Protocol-specific Events and Interrupts

These events are related to protocol-specific actions that are described in the corresponding protocol chapters. The related indication flags are located in register USCI_PROTSTS. All events can be individually enabled for the generation of the common protocol interrupt.

Event	Indication Flag	Indication Cleared By	Interrupt Enabled By
Protocol-specific events in UART mode	USCI_PROTSTS [17:16] and USCI_PROTSTS [11:5]	It is cleared by software writes 1 to corresponding interrupt bit of USCI_PROTSTS.	USCI_PROTIEN[2:1]
Protocol-specific events in SPI mode	USCI_PROTSTS [9:8], USCI_PROTSTS [6:5]		USCI_PROTIEN [3:0]
Protocol-specific events in I^2C mode	USCI_PROTSTS [13:8], USCI_PROTSTS [5]		USCI_PROTIEN [6:0]

Table 6.11-4 Protocol-specific Events and Interrupt Handling

6.11.4.7 Wake-up

The protocol-related wake-up functional information is located in the Wake-up Control Register (USCI_WKCTL) and in the Wake-up Status Register (USCI_WKSTS). These registers are shared between the available protocols. As a consequence, the meaning of the bit positions in these registers is different within the protocols.

6.12 USCI – UART Mode

6.12.1 Overview

The asynchronous serial channel UART covers the reception and the transmission of asynchronous data frames. It performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the controller. The receiver and transmitter are independent, frames can start at different points in time for transmission and reception.

The UART controller also provides the LIN function. There is incoming data to wake up the system.

6.12.2 Features

- Supports one transmit buffer and two receive buffer for data payload
- Supports programmable baud-rate generator
- Supports 9-Bit Data Transfer
- Supports LIN function
- Supports baud rate detection by built-in capture event of baud rate generator
- Supports Wake-up function

6.12.3 Block Diagram

Figure 6.12-1 USCI - UART Mode Block Diagram

Note: nCTS and nRTS function are not supported.

6.12.4 Basic Configuration

The basic configurations of USCI0 for UART mode are as follows.

- USCI0 pins are configured in SYS_GPA_MFP, SYS_GPC_MFP, SYS_GPD_MFP registers.
- Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK[24]).
- Reset USCI0 controller in USCI0RST (SYS_IPRST1[24]).

The basic configurations of USCI1 for UART mode are as follows.

- USCI1 pins are configured in SYS_GPA_MFP, SYS_GPC_MFP, SYS_GPD_MFP registers.
- Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK[25]).
- Reset USCI1 controller in USCI1RST (SYS_IPRST1[25]).

6.12.5 Functional Description

6.12.5.1 USCI Common Function Description

Please refer to section 6.11.4 for detailed information.

6.12.5.2 Signal Description

An UART connection is characterized by the use of a single connection line between a transmitter and a receiver. The receiver input signal (RXD) is handled by the input stage USCIx_DAT0 and the transmit output (TXD) signal is handled by the output stage of USCIx_DAT1.

For full-duplex communication, an independent communication line is needed for each transfer direction. Figure 6.12-2 shows an example with a point-to-point full-duplex connection between two communication partners UART module A and UART module B.

Figure 6.12-2 UART Signal Connection for Full-Duplex Communication

Input Signals

For UART protocol, the number of input signals is demonstratedTable 6.12-1 Each input signal is handled by an input processor for signal conditioning, such as signal inverse selection control, or a digital input filter. The input signals can be classified according to their meaning for the protocols, as shown in Table 6.12-1.

Selected Protocol		UART
Control Input	USCIx_CTL0	X
	USCIx_CTL1	X
Data Input(s)	USCIx_DAT0	RX
	USCIx_DAT1	X

Table 6.12-1 Input Signals for UART Protocols

Output Signals

For UART protocol, up to each protocol-related output signals are available. The number of actually used outputs depends on the selected protocol. They can be classified according to their meaning for the protocols.

Selected Protocol		UART
Control Output	USCI_CTL0	X
	USCI_CTL1	X
Data Output (s)	USCI_DAT0	X
	USCI_DAT1	TX

Table 6.12-2 Output Signals for Different Protocols

6.12.5.3 Frame Format

A standard UART frame is shown in Figure 6.12-3. It consists of:

- An idle time with the signal level 1.
- One start of frame bit (SOF) with the signal level 0.
- 6~13 bit data
- A parity bit (P), programmable for either even or odd parity. It is optionally possible to handle frames without parity bit.
- One or two stop bits with the signal level 1.

Figure 6.12-3 UART Standard Frame Format

The protocol specific bits (SOF, P, STOP) are automatically handled by the UART protocol state machine and do not appear in the data flow via the receive and transmit buffers.

Start Bit

The receiver input signal USCIx_DAT0 is checked for a falling edge. An SOF bit is detected when a falling edge occurs while the receiver is idle or after the sampling point of the last stop bit. To increase noise immunity, the SOF bit timing starts with the first falling edge that is detected. If the sampled bit value of the SOF is 1, the previous falling edge is considered to be due to noise and the receiver is considered to be idle again.

Data Field

The length of the data field (number of data bits) can be programmed by the bit field of DWIDTH (UUART_LINECTL[11:8]). It can vary between 6 to 13 data bits.

Note: In UART protocol, the data transmission order is LSB first by setting LSB (UUART_LINECTL[0]) to 1.

Parity Bit

The UART allows parity generation for transmission and parity check for reception on frame base. The type of parity can be selected by bit field PARITYEN (UUART_PROTCTL[1]) and EVENPARITY (UUART_PROTCTL[2]), common for transmission and reception (no parity, even or odd parity). If the parity handling is disabled, the UART frame does not contain any parity bit. For consistency reasons, all communication partners have to be programmed to the same parity mode.

After the last data bit of the data field, the transmitter automatically sends out its calculated parity bit if parity generation has been enabled. The receiver interprets this bit as received parity and compares it to its internally calculated one. The result of the parity check and frame check (STOP bit) are monitored in the protocol status registers (UUART_PROTSTS). The register contains bits to monitor a protocol-related status and protocol-related error indication (FRMERR, PARITYERR).

Stop Bit

Each UART frame is completed by 1 or 2 of stop bits with the signal level 1 (same level as the idle level). The number of stop bits is programmable by bit STOPB (UUART_PROTCTL[0]). A new start bit can be transferred directly after the last stop bit.

Transfer Status Indication

RXBUSY (UUART_PROTSTS[10]) indicates the receiver status.

The receiver status can be monitored by RXBUSY bit. In this case, bit RXBUSY is set during a complete frame reception from the beginning of the start of frame bit to the end of the last stop bit.

6.12.5.4 Operating Mode

In order to operate the UART protocol, the following issues have to be considered:

Select UART Mode

The UART protocol can be selected by setting FUNMODOE (UUART_CTL[2:0]) to 0x2 and the UART protocol can be enabled by setting PROTEN (UUART_PROTCTL [31]) to 1. Note that the FUNMODE must be set 0 before protocol changing and it is recommended to configure all parameters of the UART before UART protocol is enabled.

Pin Connections

The USCI_DAT0 pin is used for UART receive data input signal (RX) in UART protocol. The property of input data signal can be configured in UUART_DATIN0. It is suggested to set EDGEDET (UUART_DATIN0[4:3]) as 0x2 for start bit detection.

The USCI_DAT1 pin is used for UART transmit data output signal (TX) in UART protocol. The property of output data signal can be configured in UUART_LINECTL.

Bit Timing Configuration

The desired baud rate setting has to be selected, including the baud rate generator and the bit timing. Please refer to section 6.12.5 for detailed description.

Frame format configuration

The word length, the stop bit number, and the parity mode has to be set up according to the application requirements by programming UUART_LINECTL and the UUART_PROTCTL register. It is required by the application, the data input and output signals can be inverted. The data transmission order is LSB first by setting LSB (UUART_LINECTL[0]) to 1.

6.12.5.5 Bit Timing

In UART mode, each frame bit is divided into data sample time in order to provide granularity in the sub-bit range to adjust the sample point to the application requirements. The number of data sample time per bit is defined by bit fields DSCNT (UUART_BRGEN[14:10]) and the length of a data sample time is given by PDSCNT (UUART_BRGEN[9:8]).

In the example given in Figure 6.12-4, one bit time is composed of 16 data sample time DSCNT(UUART_BRGEN[14:10]) = 15. It is not recommended to program less and equal than 4 data sample time per bit time.

The position of the sampling point for the bit value is fixed in 1/2 samples time. It is possible to sample the bit value to take the average of samples.

The bit timing setup (number of data sample time) is common for the transmitter and the receiver because they use the same hardware circuit.

Figure 6.12-4 UART Bit Timing (Data Sample Time)

6.12.5.6 Baud Rate Generation

The baud rate f_{UART} in UART mode depends on the number of data sample time per bit time and their timing. The baud rate setting should only be changed while the transmitter and the receiver are idle. The bits RCLKSEL, SPCLKSEL, PDSCNT, and DSCNT define the baud rate setting:

RCLKSEL (UUART_BRGEN [0])

to define the input frequency f_{REF_CLK}

SPCLKSEL (UUART_BRGEN[3:2])

to define the multiple source of the sample clock f_{SAMP_CLK}

PDSCNT (UUART_BRGEN [9:8])

to define the length of a data sample time (division of f_{REF_CLK} by 1, 2, 3, or 4)

DSCNT (UUART_BRGEN [14:10])

to define the number of data sample time per bit time

The standard setting is given by RCLKSEL = 0 ($f_{REF_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT_CLK} = f_{REF_CLK}$) and SPCLKSEL = 0x0 ($f_{SAMP_CLK} = f_{DIV_CLK}$). Under these conditions, the baud rate is given by:

$$f_{\mathrm{UART}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

To generate slower frequencies, additional divide-by-2 stages can be selected by PTCLKSEL = 1 ($f_{PROT_CLK} = f_{REF_CLK2}$), leading to:

$$f_{\mathrm{UART}} = \frac{f_{REF_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

If SPCLKSEL = 0x2 (fSAMP_CLK = f_{SCLK}), and RCLKSEL = 0 (fREF_CLK = f_{PCLK}), PTCLKSEL = 0 ($f_{PROT_CLK} = f_{REF_CLK}$). The baud rate is given by:

$$f_{\mathrm{UART}} = f_{REF_CLK} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{2} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

There is error tolerance for the UART baud rate after setting the baud rate parameter. Table 6.12-3 lists the relative error percentage examples for user to calculate his relative baud rate setting.

HCLK Source	PCLK Source	Expect Baud Rate	CLKDIV (UUART_BRGEN[25:16])	DSCNT (UUART_BRGEN[14:10])	PDSCNT	Active Baud Rate	Error Percentage
HXT12M	12M (HCLK)	115200	0xC	0x7	0x0	115384	1.6%
HIRC/2, 24M	24M (HCLK)	9600	0xF9	0x9	0x0	9600	0%
HIRC48M	48M (HCLK)	115200	0xC	0xF	0x1	115384	1.6%

Table 6.12-3 Baud rate Relationship

Note: SPCLKSEL = 0x0, PTCLKSEL = 0, RCLKSEL = 0

6.12.5.7 Auto Baud Rate Detection

The UART controller supports auto baud rate detection function. It is used to identify the input baud rate from the receiver signal (USCIx_DAT0) and then revised the baud rate clock divider CLKDIV (UUART_BRGEN[25:16]) after the baud rate function done to meet the detected baud rate information. According to the section of Timing Measurement Counter, the timing measurement counter is used for time interval measurement of the input signal (USCIx_DAT0) and the actual timer value is captured into bit field BRDETITV (UUART_PROTCTL [24:16]) in each falling edge of the detected signal.

When the ABREN (PROTOCOL[6]) bit is enabled, the 0x55 data patterns is necessary for auto baud rate detection. The falling edge of input signal starts the baud rate counter and it loads the timing measurement counter value into the BRDETITV (UUART_PROTCTL [24:16]) in the next falling edge. It is suggested to use the f_{DIV_CLK} (TMCNTSRC (UUART_BRGENC[5]) =1) as the counter source.

The CLKDIV (UUART_BRGEN[25:16]) will be revised by BRDETITV (UUART_PROTCTL [25:16]) after the auto baud rate function done (the time of 4th falling edge of input signal). If the user want to receive the next successive frame correctly, it is better to set the value of CLKDIV (UUART_BRGEN[25:16]) and DSCNT (UUART_BRGEN[14:10]) as the same value (the value shall be among the rang of 0xF and 0x5 because the DSCNT is used to define the sample counter of each bit and the PDSCNT (UUART_BRGEN[9:8]) is 0x0.

During the auto baud rate detection, the ABRDETIF (UUART_PROTSTS[9]) and the BRDETITV (UUART_PROTCTL [24:16]) will be updated after each falling edge of input signal and the auto baud rate pattern, 0x55, won't be received into the receiver buffer after the frame done. The bit of ABREN will be cleared by hardware after the 4th falling edge of input signal is detected thus the user can read the status of ABREN to know the auto baud rate function is done or not.

If the CLKDIV and DSCNT are not set as the same value in calculation the auto baud rate function, the user shall calculate the proper average baud rate by the value of BRDETITV and CLKDIV after the auto baud rate function done.

If the baud rate of input signal is very slow and the bit time of timing measurement counter can't calculate the correct period of the input bit time, there is a ABERRSTS bit (UUART_PROTSTS[11]) to indicate the error information of the auto baud rate detection. At this time, the user shall revise the value of CLKDIV and require the Host device to send the 0x55 pattern again.

According to the limitation of timing measurement counter, the maximum auto baud rate detection

is 0x1FE for BRDETITV.

Figure 6.12-5 UART Auto Baud Rate Control

6.12.5.8 Hardware LIN Support

To support the LIN (Local Interconnect Network) protocol, the bit LINBRKEN (UUART_PROTCTL[7]) should be set for the master. For slave devices, it can be cleared and the fixed number of 8 data bits has to be set (DWIDTH = 8). For both, LIN Master and LIN Slave devices, the parity generation has to be switched off (PARITYEN (UUART_PROTCTL[1]) = 0) and transfers take place with LSB first (LSB (UUART_LINECTL[0]) = 1) and 1 stop bit (STOPB (UUART_PROTCTL[0]) = 0).

The LIN data exchange protocol contains several symbols that can be all handled in UART mode. Each single LIN symbol represents a complete UART frame. The LIN bus is a master-slave bus system with a single master and multiple slaves (for the exact definition please refer to the official LIN specification).

A complete LIN frame contains the following symbols:

Synchronization Break

The master sends a synchronization break to signal at the beginning of a new frame. It contains 13 consecutive bit times at 0 level, followed by at least one bit time at 1 level (corresponding to 1 stop bit). Therefore, LINBRKEN (UUART_PROTCTL[7]) has to be written with 1 (leading to a frame with SOF followed by 12 data bits at 0 level).

A slave device shall detect 11 consecutive bit times at 0 level, which is done by the synchronization break detection. The bit BRKDETIF (UUART_PROTSTS[8]) is set if such an event is detected and a protocol interrupt can be generated. Additionally, the received data value 0 appears in the receive buffer and a format error is signaled.

If the baud rate of the slave has to be adapted to the master, the baud rate measurement has to be enabled for falling edges by setting ABREN (UUART_PROTCTL[6])= 1 before the next symbol starts.

Synchronization Byte

The master sends this symbol after writing the data value 0x55 to UUART_TXDAT. A slave

device can either receive this symbol without any further action (and can discard it) or use the falling edges for baud rate measurement. ABREN is used to capture a timer counter value for the receiver synchronization byte. The valid captured values can be read out after the frame is transmitted done. After this symbol, the baud rate detection can be disabled (ABREN = 0) and then revise the baud rate clock divider CLKDIV (UUART_BRGEN[25:16]). Please refer to section 6.12.5.6 for detailed description.

Other Symbols

The other symbols of a LIN frame can be handled with UART data frames without specific actions.

Please note that during the baud rate measurement of the UART receiver, the transmitter still performs a transmission.

6.12.5.9 Wake-up Function

The USCI controller in UART mode supports wake-up system function. The wake-up source includes incoming data. Incoming data wake-up source description as follows:

Incoming data wake-up

When system is in power-down and both of the WKEN (UUART_WKCTL [0]) and DATWKEN (UUART_PROTCTL[9]) are set, the toggle of incoming data pin can wake-up the system. In order to receive the incoming data after the system wake-up, the WAKECNT (UUART_PROTCTL[14:11]) shall be set. These bits field of WAKECNT (UUART_PROTIEN [14:11]) indicate how many clock cycle selected by f_{PDS_CLK} do the controller can get the 1st bit (start bit) when the device is woken up from Power-down mode.

Note 1: By the WAKECNT is loaded into the hardware counter at the time of WKF (UUART_WKSTS[0]) is cleared the user shall clear the wake-up flag first to make sure the time period of WAKECNT is about the wake time of system.

Note 2: To receive the incoming data, the relation between the selected clock stable and the baud rate shall be take care (for example, the stable time of HXT is 4096 clock period)..

Note: Stable count means HCLK source recovery stable count.

Figure 6.12-6 Incoming Data Wake-Up

6.12.5.10 Interrupt Events

Protocol Interrupt Events

The following protocol-related events are generated in UART mode and can lead to a protocol interrupt.

Please note that the bits in register UUART_PROTSTS are not automatically cleared by hardware and have to be cleared by software in order to monitor new incoming events.

Receiver line status

The protocol-related error FRMERR (UUART_PROTSTS[6]) or PARITYERR (UUART_PROTSTS[5]) are two flags that are assigned to each received data word in the corresponding receiver buffer status registers.

In UART mode, the result of the parity check by the protocol-related error indication (0 = received parity bit equal to calculated parity value), and the result of frame check by the protocol-related error indication (0 = received stop bit equal to the format value '1'). This information is elaborated for each data frame.

The break error flag (BREAK (UUART_PROTSTS[7])) is assigned when the receive data is 0, the received parity and the stop bit are also 0.

The interrupt indicates that there are parity error, frame error or the break data detection in the BREAK, FRMERR, PARITYERR (UUART_PROTSTS[7:5]) bits. The controller wil issue an interrupt if RLSIEN (UUART_PROTIEN[2]) is also set to 1.

Auto baud rate detection

The auto baud rate interrupt, ABRDETIF (UUART_PROTSTS [9]), indicates that the timing measurement counter has getting 2-bit duration for auto baud rate capture function. The controller wil issue an interrupt if ABRIEN (UUART_PROTIEN[1]) is also set to 1.

The auto baud rate detection function will be enabled in the first falling edge of receiver signal. The auto baud rate detection function is measurement after the next following falling is detected and it is finished when the frame transfer done. After the transfer done, the timing measurement counter value divided by twice is equal to the number of sample time per bit. The user can read the value of BRDETITV (UUART_PROTCTL[24:16]) and write into the baud rate generator register CLKDIV (UUART_BRGEN[25:16]).

Synchronization break detection

This interrupt can be used in LIN networks to indicate the reception of the synchronization break symbol (at the beginning of a LIN frame). The controller wil issue an interrupt if BRKIEN (UUART_PROTIEN[0]) is also set to 1.

Data Transfer Interrupt Handling

The data transfer interrupts indicate events related to UART frame handling.

Transmit start interrupt

Bit TXSTIF (UUART_PROTSTS [1]) is set after the start bit of a data word. In buffer mode, this is the earliest point in time when a new data word can be written to UUART_TXDAT. The controller wil issue an interrupt if TXSTIEN (UUART_INTEN[1]) is also set to 1.

Transmitter finished

This interrupt indicates that the transmitter has completely finished all data in the buffer. Bit

TXENDIF (UUART_PROTSTS [2]) becomes set at the end of the last stop bit. The controller wil issue an interrupt if TXENDIEN (UUART_INTEN[2]) is also set to 1.

Receiver starts interrupt

Bit RXSTIF (UUART_PROTSTS [3]) is set after the sample point of the start bit. The controller wil issue an interrupt if RXSTIEN (UUART_INTEN[3]) is also set to 1.

Receiver frame finished

This interrupt indicates that the receiver has completely finished a frame. Bit RXENDIF (UUART_PROTSTS [4]) becomes set at the end of the last receive bit. The controller wil issue an interrupt if RXENDIEN (UUART_INTEN[4]) is also set to 1.

6.12.5.11 Programming Example

The following steps are used to configure the UART protocol setting and the data transmission.

1. Set FUNMODE (UUART_CTL[2:0]) to 0x2 to select UART protocol.
2. Write baud rate generator register UUART_BRGEN to select desired baud rate.
    1. Set SPCLKSEL (UUART_BRGEN[3:2]), PTCLKSEL (UUART_BRGEN[1]) and RCLKSEL (UUART_BRGEN[0]) to select the clock source.
    2. Configure CLKDIV (UUART_BRGEN[25:16]), DSCNT (UUART_BRGEN[14:10]) and PDSCNT (UUART_BRGEN[9:8]) to determine the baud rate divider.
3. Write line control register UUART_LINECTL and protocol control register UUART_PROTCTL to configure the transmission data format and UART protocol setting.
    1. Program data field length in DWIDTH (UUART_LINECTL[11:8]).
    2. Enable parity bit and determine the parity bit type by setting EVENPARITY (UUART_PROTCTL[2]) and PARITYEN (UUART_PROTCTL[1]).
    3. Configure stop bit length by setting STOPB (UUART_PROTCTL[0]).
    4. Enable LSB (UUART_LINECTL[0]) to select LSB first transmission for UART protocol.
    5. Set EDGEDET (UUART_DATIN0[4:3]) to 0x2 to select the detected edge as falling edge for receiver start bit detection.
4. Set PROTEN (UUART_PROTCTL[31]) to 1 to enable UART protocol.
5. Transmit and receive data.
    - Write transmit data register UUART_TXDAT to transmit data.
    - Wait until TXSTIF(UUART_PROTSTS[1]) is set and then user can write the next data in UUART_TXDAT.
    - Enable parity bit and determine the parity bit type by setting EVENPARITY (UUART_PROTCTL[2]) and PARITYEN (UUART_PROTCTL[1]).
    - When TXENDIF(UUART_PROTSTS[2]) is set, the transmit buffer is empty and the stop bit of stop bit of the last data has been transmitted.
    - If RXENDIF(UUART_PROTSTS[4]) is set, the receiver has finished a data frame completely. User can get the data by reading receive data register UUART_RXDAT.

6.12.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UUART_CTL **x = 0, 1**	UUARTx_BA+0x00	R/W	USCI Control Register	0x0000_0000
UUART_INTEN **x = 0, 1**	UUARTx_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
UUART_BRGEN **x = 0, 1**	UUARTx_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UUART_DATIN0 **x = 0, 1**	UUARTx_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
UUART_CTLIN0 **x = 0, 1**	UUARTx_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
UUART_CLKIN **x = 0, 1**	UUARTx_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
UUART_LINECTL **x = 0, 1**	UUARTx_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UUART_TXDAT **x = 0, 1**	UUARTx_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UUART_RXDAT **x = 0, 1**	UUARTx_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UUART_BUFCTL **x = 0, 1**	UUARTx_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
UUART_BUFSTS **x = 0, 1**	UUARTx_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101
UUART_WKCTL **x = 0, 1**	UUARTx_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UUART_WKSTS **x = 0, 1**	UUARTx_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UUART_PROTCTL **x = 0, 1**	UUARTx_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UUART_PROTIEN **x = 0, 1**	UUARTx_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UUART_PROTSTS **x = 0, 1**	UUARTx_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.12.7 Register Description

USCI Control Register (UUART_CTL)

Register	Offset	R/W	Description	Reset Value
UUART_CTL **x = 0, 1**	UUARTx_BA+0x00	R/W	USCI 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					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 0x0 = The USCI is disabled. All protocol related state machines are set to idle state. 0x1 = The SPI protocol is selected. 0x2 = The UART protocol is selected. 0x4 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Interrupt Enable Register (UUART_INTEN)

Register	Offset	R/W	Description	Reset Value
UUART_INTEN x = 0, 1	UUARTx_BA+0x04	R/W	USCI 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**RXENDIEN**	**RXSTIEN**	**TXENDIEN**	**TXSTIEN**	**Reserved**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RXENDIEN**	**Receive End Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive finish event. 0 = The receive end interrupt Disabled. 1 = The receive end interrupt Enabled.
[3]	**RXSTIEN**	**Receive Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive start event. 0 = The receive start interrupt Disabled. 1 = The receive start interrupt Enabled.
[2]	**TXENDIEN**	**Transmit End Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit finish event. 0 = The transmit finish interrupt Disabled. 1 = The transmit finish interrupt Enabled.
[1]	**TXSTIEN**	**Transmit Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit start event. 0 = The transmit start interrupt Disabled. 1 = The transmit start interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (UUART_BRGEN)

Register	Offset	R/W	Description	Reset Value
UUART_BRGEN x = 0, 1	UUARTx_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						**CLKDIV**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CLKDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**DSCNT**					**PDSCNT**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TMCNTSRC**	**TMCNTEN**	**SPCLKSEL**		**PTCLKSEL**	**RCLKSEL**

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} ($f_{DIV_CLK} = f_{PROT_CLK}$ / (CLKDIV+1)). **Note:** In UART function, it can be updated by hardware in the 4th falling edge of the input data 0x55 when the auto baud rate function (ABREN(UUART_PROTCTL[6])) is enabled. The revised value is the average bit time between bit 5 and bit 6. The user can use revised CLKDIV and new BRDETITV (UUART_PROTCTL[24:16]) to calculate the precise baud rate.
[15]	**Reserved**	Reserved.
[14:10]	**DSCNT**	**Denominator for Sample Counter** This bit field defines the divide ratio of the sample clock f_{SAMP_CLK}. The divided frequency $f_{DS_CNT} = f_{PDS_CNT}$ / (DSCNT+1). **Note:** The maximum value of DSCNT is 0xF on UART mode and suggest to set over 4 to confirm the receiver data is sampled in right value.
[9:8]	**PDSCNT**	**Pre-divider for Sample Counter** This bit field defines the divide ratio of the clock division from sample clock f_{SAMP_CLK}. The divided frequency $f_{PDS_CNT} = f_{SAMP_CLK}$ / (PDSCNT+1).
[7:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Timing Measurement Counter Clock Source Selection** 0 = Timing measurement counter with f_{PROT_CLK}. 1 = Timing measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Timing Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Timing measurement counter Disabled. 1 = Timing measurement counter Enabled.

[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of a sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = f_{SAMP_CLK} = f_{DIV_CLK}. 01 = f_{SAMP_CLK} = f_{PROT_CLK}. 10 = f_{SAMP_CLK} = f_{SCLK}. 11 = f_{SAMP_CLK} = f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source signal of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source signal of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = HXT/LXT.

USCI Input Data Signal Configuration (UUART_DATIN0)

Register	Offset	R/W	Description	Reset Value
UUART_DATIN0 **x = 0, 1**	UUARTx_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	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			**EDGEDET**		**ININV**	**Reserved**	**SYNCSEL**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4:3]	**EDGEDET**	**Input Signal Edge Detection Mode** This bit field selects which edge actives the trigger event of input data signal. 00 = The trigger event activation is disabled. 01 = A rising edge activates the trigger event of input data signal. 10 = A falling edge activates the trigger event of input data signal. 11 = Both edges activate the trigger event of input data signal. **Note:** In UART function mode, it is suggested to set this bit field as 10.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Signal Synchronization Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Input Control Signal Configuration (UUART_CTLIN0)

Register	Offset	R/W	Description	Reset Value
UUART_CTLIN0 **x = 0, 1**	UUARTx_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	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					**ININV**	**Reserved**	**SYNCSEL**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Input Clock Signal Configuration (UUART_CLKIN)

Register	Offset	R/W	Description	Reset Value
UUART_CLKIN x = 0, 1	UUARTx_BA+0x28	R/W	USCI Input Clock Signal Configuration 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							**SYNCSEL**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit.

USCI Line Control Register (UUART_LINECTL)

Register	Offset	R/W	Description	Reset Value
UUART_LINECTL **x = 0, 1**	UUARTx_BA+0x2C	R/W	USCI 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				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTLOINV	**Reserved**	**DATOINV**	**Reserved**				**LSB**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0]. **Note:** In UART protocol, the length can be configured as 6~13 bits.
[7]	**CTLOINV**	**Control Signal Output Inverse Selection** This bit defines the relation between the internal control signal and the output control signal. 0 = No effect. 1 = The control signal will be inverted before its output. **Note:** In UART protocol, the control signal means nRTS signal.
[6]	**Reserved**	Reserved.
[5]	**DATOINV**	**Data Output Inverse Selection** This bit defines the relation between the internal shift data value and the output data signal of USCIx_DAT1 pin. 0 = The value of USCIx_DAT1 is equal to the data shift register. 1 = The value of USCIx_DAT1 is the inversion of data shift register.

[4:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (UUART_TXDAT)

Register	Offset	R/W	Description	Reset Value
UUART_TXDAT **x = 0, 1**	UUARTx_BA+0x30	W	USCI Transmit 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**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission.

USCI Receive Data Register (UUART_RXDAT)

Register	Offset	R/W	Description	Reset Value
UUART_RXDAT **x = 0, 1**	UUARTx_BA+0x34	R	USCI Receive 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**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer. **Note:** RXDAT[15:13] indicate the same frame status of BREAK, FRMERR and PARITYERR (UUART_PROTSTS[7:5]).

USCI Transmitter/Receive Buffer Control Register (UUART_BUFCTL)

Register	Offset	R/W	Description	Reset Value
UUART_BUFCTL **x = 0, 1**	UUARTx_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**RXRST**	**TXRST**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXCLR	**RXOVIEN**	**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXCLR	**Reserved**						

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset the receive-related counters, state machine, and the content of receive shift register and data buffer. **Note 1:** It is cleared automatically after one PCLK cycle. **Note 2:** It is suggest to check the RXBUSY (UUART_PROTSTS[10]) before this bit will be set to 1.
[16]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset the transmit-related counters, state machine, and the content of transmit shift register and data buffer**.** **Note:** It is cleared automatically after one PCLK cycle.
[15]	**RXCLR**	**Clear Receive Buffer** 0 = No effect. 1 = The receive buffer is cleared (filling level is cleared and output pointer is set to input pointer value). Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[14]	**RXOVIEN**	**Receive Buffer Overrun Error Interrupt Enable Bit** 0 = Receive overrun interrupt Disabled. 1 = Receive overrun interrupt Enabled.
[13:8]	**Reserved**	Reserved.
[7]	**TXCLR**	**Clear Transmit Buffer** 0 = No effect. 1 = The transmit buffer is cleared (filling level is cleared and output pointer is set to input

		pointer value). Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[6:0]	**Reserved**	Reserved.

USCI Transmit/Receive Buffer Status Register (UUART_BUFSTS)

Register	Offset	R/W	Description	Reset Value
UUART_BUFSTS x = 0, 1	UUARTx_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101

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						**TXFULL**	**TXEMPTY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**RXOVIF**	**Reserved**	**RXFULL**	**RXEMPTY**

Bits	Description	
[31:10]	**Reserved**	Reserved.
[10]	**Reserved**	Reserved.
[9]	**TXFULL**	**Transmit Buffer Full Indicator** 0 = Transmit buffer is not full. 1 = Transmit buffer is full.
[8]	**TXEMPTY**	**Transmit Buffer Empty Indicator** 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty.
[7:4]	**Reserved**	Reserved.
[3]	**RXOVIF**	**Receive Buffer Over-run Error Interrupt Status** This bit indicates that a receive buffer overrun error event has been detected. If RXOVIEN (UUART_BUFCTL[14]) is enabled, the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit. 0 = A receive buffer overrun error event has not been detected. 1 = A receive buffer overrun error event has been detected.
[2]	**Reserved**	Reserved.
[1]	**RXFULL**	**Receive Buffer Full Indicator** 0 = Receive buffer is not full. 1 = Receive buffer is full.

USCI Wake-up Control Register (UUART_WKCTL)

Register	Offset	R/W	Description	Reset Value
UUART_WKCTL x = 0, 1	UUARTx_BA+0x54	R/W	USCI 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					**PDBOPT**	**Reserved**	**WKEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDBOPT**	**Power Down Blocking Option** 0 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, MCU will stop the transfer and enter Power-down mode immediately. 1 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, the on-going transfer will not be stopped and MCU will enter idle mode immediately.
[1]	**Reserved**	Reserved.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (UUART_WKSTS)

Register	Offset	R/W	Description	Reset Value
UUART_WKSTS x = 0, 1	UUARTx_BA+0x58	R/W	USCI 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							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – UART (UUART_PROTCTL)

Register	Offset	R/W	Description	Reset Value
UUART_PROTCTL **x = 0, 1**	UUARTx_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	**Reserved**	**BCEN**	**Reserved**	**Reserved**	**STICKEN**	**Reserved**	**BRDETITV**
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
BRDETITV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**WAKECNT**				**Reserved**	**DATWKEN**	**LINRXEN**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
LINBRKEN	**ABREN**	**Reserved**			**EVENPARITY**	**PARITYEN**	**STOPB**

Bits	Description	
[31]	**PROTEN**	**UART Protocol Enable Bit** 0 = UART Protocol Disabled. 1 = UART Protocol Enabled.
[30]	**Reserved**	Reserved.
[29]	**BCEN**	**Transmit Break Control Enable Bit** 0 = Transmit Break Control Disabled. 1 = Transmit Break Control Enabled. **Note:** 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 line and has no effect on the transmitter logic.
[27]	**Reserved**	Reserved.
[26]	**STICKEN**	**Stick Parity Enable Bit** 0 = Stick parity Disabled. 1 = Stick parity Enabled.
[25]	**Reserved**	Reserved.
[24:16]	**BRDETITV**	**Baud Rate Detection Interval** This bit fields indicate how many clock cycle selected by TMCNTSRC (UUART_BRGEN [5]) does the slave calculates the baud rate in one bits. The order of the bus shall be 1 and 0 step by step (e.g. the input data pattern shall be 0x55). The user can read the value to know the current input baud rate of the bus whenever the ABRDETIF (UUART_PROTCTL[9]) is set. **Note:** This bit can be cleared to 0 by software writing '0' to the **BRDETITV**.
[15]	**Reserved**	Reserved.
[14:11]	**WAKECNT**	**Wake-up Counter** These bits field indicate how many clock cycle selected by f_{PDS_CNT} do the slave can get the 1st bit (start bit) when the device is woken up from Power-down mode.

[10]	**Reserved**	Reserved.
[9]	**DATWKEN**	**Data Wake-up Mode Enable Bit** 0 = Data wake-up mode Disabled. 1 = Data wake-up mode Enabled.
[8]	**LINRXEN**	**LIN RX Duplex Mode Enable Bit** 0 = LIN RX Duplex mode Disabled. 1 = LIN RX Duplex mode Enabled. The LIN can be play as Slave to receive the LIN frame. **Note:** This bit is used to check the break duration for incoming data when the LIN operation is active.
[7]	**LINBRKEN**	**LIN TX Break Mode Enable Bit** 0 = LIN TX Break mode Disabled. 1 = LIN TX Break mode Enabled. **Note1:** When TX break field transfer operation is finished, this bit will be cleared automatically. **Note2:** 13-bit level 0 and 1-bit level 1 were sent out before the 1st data be transmitted.
[6]	**ABREN**	**Auto-baud Rate Detect Enable Bit** 0 = Auto-baud rate detect function Disabled. 1 = Auto-baud rate detect function Enabled. **Note:** When the auto - baud rate detect operation finishes, hardware will clear this bit. The associated interrupt ABRDETIF (USCI_PROTST[9]) will be generated (If ARBIEN (UUART_PROTIEN [1]) is enabled).
[5:4]	**Reserved**	Reserved.
[3]	**Reserved**	Reserved.
[2]	**EVENPARITY**	**Even Parity Enable Bit** 0 = Odd number of logic 1's is transmitted and checked in each word. 1 = Even number of logic 1's is transmitted and checked in each word. **Note:** This bit has effect only when PARITYEN is set.
[1]	**PARITYEN**	**Parity Enable Bit** This bit defines the parity bit is enabled in an UART frame. 0 = The parity bit Disabled. 1 = The parity bit Enabled.
[0]	**STOPB**	**Stop Bits** This bit defines the number of stop bits in an UART frame. 0 = The number of stop bits is 1. 1 = The number of stop bits is 2.

USCI Protocol Interrupt Enable Register – UART (UUART_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UUART_PROTIEN **x = 0, 1**	UUARTx_BA+0x60	R/W	USCI Protocol 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved					**RLSIEN**	**ABRIEN**	**BRKIEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**RLSIEN**	**Receive Line Status Interrupt Enable Bit** 0 = Receive line status interrupt Disabled. 1 = Receive line status interrupt Enabled. **Note:** UUART_PROTSTS[7:5] indicates the current interrupt event for receive line status interrupt.
[1]	**ABRIEN**	**Auto-baud Rate Interrupt Enable Bit** 0 = Auto-baud rate interrupt Disabled. 1 = Auto-baud rate interrupt Enabled.
[0]	**BRKIEN**	**LIN Break Detected Interrupt Enable Bit** 0 = The LIN break detected interrupt generation Disabled. 1 = The LIN break detected interrupt generation Enabled.

USCI Protocol Status Register – UART (UUART_PROTSTS)

Register	Offset	R/W	Description	Reset Value
UUART_PROTSTS **x = 0, 1**	UUARTx_BA+0x64	R/W	USCI Protocol 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				**ABERRSTS**	**RXBUSY**	**ABRDETIF**	**BRKDETIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
BREAK	**FRMERR**	**PARITYERR**	**RXENDIF**	**RXSTIF**	**TXENDIF**	**TXSTIF**	**Reserved**

Bits	Description	
[31:17]	**Reserved**	Reserved.
[16:12]	**Reserved**	Reserved.
[11]	**ABERRSTS**	**Auto-baud Rate Error Status** This bit is set when auto-baud rate detection counter overrun. When the auto-baud rate counter overrun, the user shall revise the CLKDIV (UUART_BRGEN[25:16]) value and enable ABREN (UUART_PROTCTL[6]) to detect the correct baud rate again. 0 = Auto-baud rate detect counter is not overrun. 1 = Auto-baud rate detect counter is overrun. **Note 1:** This bit is set at the same time of ABRDETIF. **Note 2:** This bit can be cleared by writing “1” to ABRDETIF or ABERRSTS.
[10]	**RXBUSY**	**RX Bus Status Flag (Read Only)** This bit indicates the busy status of the receiver. 0 = The receiver is Idle. 1 = The receiver is BUSY.
[9]	**ABRDETIF**	**Auto-baud Rate Interrupt Flag** This bit is set when auto-baud rate detection is done among the falling edge of the input data. If the ABRIEN (UUART_PROTCTL[6]) is set, the auto-baud rate interrupt will be generated. This bit can be set 4 times when the input data pattern is 0x55 and it is cleared before the next falling edge of the input bus. 0 = Auto-baud rate detect function is not done. 1 = One Bit auto-baud rate detect function is done. **Note:** This bit can be cleared by writing “1” to it.
[8]	**BRKDETIF**	**LIN Break Detected Interrupt Flag (Read Only)** This bit is set to logic 1 whenever the received data input (RX) is held in the “spacing state” (logic 0) for longer than 12-bit transmission time in LIN mode function. 0 = LIN Break is no detected.

		1 = LIN Break is detected. **Note:** This bit is read only, but can be cleared by writing '1' to it.
[7]	**BREAK**	**Break Flag** This bit is set to 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). 0 = No Break is generated. 1 = Break is generated in the receiver bus. **Note:** This bit can be cleared by write "1" among the BREAK, FRMERR and PARITYERR bits.
[6]	**FRMERR**	**Framing Error Flag** 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 logic 0). 0 = No framing error is generated. 1 = Framing error is generated. **Note:** This bit can be cleared by write "1" among the BREAK, FRMERR and PARITYERR bits.
[5]	**PARITYERR**	**Parity Error Flag** This bit is set to logic 1 whenever the received character does not have a valid "parity bit". 0 = No parity error is generated. 1 = Parity error is generated. **Note:** This bit can be cleared by write "1" among the BREAK, FRMERR and PARITYERR bits.
[4]	**RXENDIF**	**Receive End Interrupt Flag** 0 = A receive finish interrupt status has not occurred. 1 = A receive finish interrupt status has occurred. **Note:** It is cleared by software writing one into this bit.
[3]	**RXSTIF**	**Receive Start Interrupt Flag** 0 = A receive start interrupt status has not occurred. 1 = A receive start interrupt status has occurred. **Note:** It is cleared by software writing one into this bit.
[2]	**TXENDIF**	**Transmit End Interrupt Flag** 0 = A transmit end interrupt status has not occurred. 1 = A transmit end interrupt status has occurred. **Note:** It is cleared by software writing one into this bit.
[1]	**TXSTIF**	**Transmit Start Interrupt Flag** 0 = A transmit start interrupt status has not occurred. 1 = A transmit start interrupt status has occurred. **Note 1:** It is cleared by software writing one into this bit. **Note 2:** Used for user to load next transmit data when there is no data in transmit buffer.
[0]	**Reserved**	Reserved.

6.13 USCI – SPI Mode

6.13.1 Overview

The SPI protocol of USCI controller applies to synchronous serial data communication and allows full duplex transfer. It supports both master and Slave operation mode with the 4-wire bi-direction interface. SPI mode of USCI controller performs 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. The SPI mode is selected by FUNMODE (USPI_CTL[2:0]) = 0x1.

The SPI protocol can operate as master or Slave mode by setting the SLAVE (USPI_PROTCTL[0]) to communicate with the off-chip SPI Slave or master device. The application block diagrams in master and Slave mode are shown as Figure 6.13-1 and Figure 6.13-2.

Figure 6.13-1 SPI Master Mode Application Block Diagram (x=0, 1)

Figure 6.13-2 SPI Slave Mode Application Block Diagram (x=0, 1)

6.13.2 Features

- Supports master or slave mode operation (the maximum frequency for Master = f_{PCLK} / 2, for Slave < f_{PCLK} / 5)
- Configurable bit length of a transfer word from 4 to 16-bit
- Supports one transmit buffer and two receive buffers for data payload
- Supports MSB first or LSB first transfer sequence
- Supports Word Suspend function
- Supports 3-wire, no slave select signal, bi-direction interface
- Supports wake-up function by slave select signal in Slave mode

6.13.3 Block Diagram

Figure 6.13-3 USCI - SPI Mode Block Diagram

6.13.4 Basic Configuration

The basic configurations of USCI0 for SPI mode are as follows.

- USCI0 pins are configured in SYS_GPA_MFP, SYS_GPC_MFP, and SYS_GPD_MFP registers.
- Enable USCI0 peripheral clock in USCI0CKEN (CLK_APBCLK[24]).
- Reset USCI0 controller in USCI0RST (SYS_IPRST1[24]).

The basic configurations of USCI1 for SPI mode are as follows:

- USCI1 pins are configured in SYS_GPA_MFP, SYS_GPC_MFP, and SYS_GPD_MFP registers.
- Enable USCI1 peripheral clock in USCI1CKEN (CLK_APBCLK[25]).
- Reset USCI1 controller in USCI1RST (SYS_IPRST1[25]).

6.13.5 Functional Description

6.13.5.1 USCI Common Function Description

Please refer to section 6.11.4 for detailed information.

6.13.5.2 Signal Description

A device operating in master mode controls the start and end of a data transfer, as well as the generation of the SPI bus clock and slave select signal. The slave select signal indicates the start and the end of a data transfer, and the master device can use it to enable the transmitting or receiving operations of slave device. Slave device receives the SPI bus clock and optionally a slave select signal for data transaction. The signals for SPI communication are shown as Table 6.13-1.

SPI Mode	Receive Data	Transmit Data	Serial Bus Clock	Slave Select
Full-duplex SPI Master	SPI_MISO (USCIx_DAT1)	SPI_MOSI (USCIx_DAT0)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)
Full-duplex SPI Slave	SPI_MOSI (USCIx_DAT0)	SPI_MISO (USCIx_DAT1)	SPI_CLK (USCIx_CLK)	SPI_SS (USCIx_CTL0)

Table 6.13-1 SPI Communication Signals (x=0, 1)

Figure 6.13-4 4-Wire Full-Duplex SPI Communication Signals (Master Mode)

Figure 6.13-5 4-Wire Full-Duplex SPI Communication Signals (Slave Mode)

6.13.5.3 Serial Bus Clock Configuration

The USCI controller needs the peripheral clock to drive the USCI logic unit to perform the data transfer. The peripheral clock frequency is equal to PCLK frequency.

In master mode, the frequency of the SPI bus clock is determined by protocol-relative clock generator. In general, the SPI bus clock is denoted as SPI clock. The frequency of SPI clock is half of f_{SAMP_CLK}, which can be selected by SPCLKSEL (USPI_BRGEN[3:2]). Refer to section 6.11.4.4 for details of protocol-relative clock generator.

In slave mode, the SPI bus clock is provided by an off-chip Master device. The peripheral clock frequency, f_{PCLK}, of SPI Slave device must be 5-times faster than the serial bus clock rate of the SPI Master device connected together (i.e. the clock rate of serial bus clock < 1/5 peripheral clock f_{PCLK} in Slave mode).

In SPI protocol, SCLKMODE (USPI_PROTCTL[7:6]) defines not only the idle state of serial bus clock but also the serial clock edge used for transmit and receive data. Both Master and Slave devices on the same communication bus should have the same SCLKMODE configuration. The four kinds of serial bus clock configuration are shown as Table 6.13-2, and timing diagrams are shown in Figure 6.13-6, Figure 6.13-7, Figure 6.13-8 and Figure 6.13-9.

SCLKMODE [1:0]	SPI Clock Idle State	Transmit Timing	Receive Timing
0x0	Low	Falling edge	Rising edge
0x1	Low	Rising edge	Falling edge
0x2	High	Rising edge	Falling edge
0x3	High	Falling edge	Rising edge

Table 6.13-2 Serial Bus Clock Configuration

Note: x = 0, 1

Figure 6.13-6 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x0)

Note: x = 0, 1

Figure 6.13-7 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x1)

Figure 6.13-8 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x2)

Figure 6.13-9 SPI Communication with Different SPI Clock Configuration (SCLKMODE=0x3)

6.13.5.4 Slave Select Signal

The slave selection signal of SPI protocol is active high by default. In SPI Master mode, the USCI controller can drive the control signal to off-chip SPI slave device through slave select pin SPI_SS (USCIx_CTL0). In SPI Slave mode, the received slave select signal can be inverted by ININV (USPI_CTLIN0[2]).

If the slave select signal of external SPI Master device is low active, the ININV (USPI_CTLIN0[2]) setting of slave device should be set to 1 for the inversion of input control signal. If USCI operates as SPI Master mode, the output slave select inversion CTLOINV (USPI_LINECTL[7]) is also

needed to set as 1 for the external SPI Slave device whose slave select signal is active low.

The duration between the slave select active edge and the first SPI clock input edge shall over 2 USCI peripheral clock cycles.

The input slave select signal of SPI Slave has to be keep inactive for at least 2 USCI peripheral clock cycles between two consecutive frames in order to correctly detect the end of a frame.

6.13.5.5 Transmit and Receive Data

The bit length of a transmit/receive data word in SPI protocol of USCI controller is defined in DWIDTH (USPI_LINECTL[11:8]), and it can be configured up to 16-bit length and not less than 4-bit length for transmitting for transmitting and receiving data in SPI communication.

The LSB bit (USPI_LINECTL[0]) defines the order of transfer data bit. If the LSB bit is set to 1, the transmission data sequence is LSB first. If the LSB bit is cleared to 0, the transmission data sequence is MSB first.

Figure 6.13-10 16-bit data Length in One Word Transaction with MSB First Format

6.13.5.6 Word Suspend

SUSPITV (USPI_PROTCTL[11:8]) provides 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 SUSPITV (USPI_PROTCTL[11:8]) is 0x3 (3.5 SPI clock cycles).

Figure 6.13-11 Word Suspend Interval between Two Transaction Words

6.13.5.7 Automatic Slave Select Function

AUTOSS (USPI_PROTCTL[3]) is used for SPI Master mode to enable the automatic slave select function. If the bit AUTOSS (USPI_PROTCTL[3]) is set, the slave select signal will be generated automatically and the setting value of SS (USPI_PROTCTL[2]) will not affect the output slave select (through USCIx_CTL0 line). This means that the slave select signal will be asserted by the USCI controller when the SPI data transfer is started by writing to the transmit buffer. And, it will be de-asserted after either all transaction is finished or one word transaction done if the value of SUSPITV (USPI_PROTCTL[11:8]) is equal to or great than 3.

If the AUTOSS bit (USPI_PROTCTL[3]) is cleared, the slave selected on USCIx_CTL0 pin will be asserted/de-asserted by setting/clearing the SS (USPI_PROTCTL[2]). The internal slave select signal is active high and the CTLOINV (USPI_LINECTL[7]) can be used for the inversion of the slave select signal.

In SPI Master mode, if the value of SUSPITV (USPI_PROTCTL[11:8]) is less than 3 and the AUTOSS (USPI_PROTCTL[3]) is set as 1, the slave select signal will be kept at active state between two successive word transactions.

In SPI Slave mode, to recognize the inactive state of the slave select signal, the inactive period of the received slave select signal must be larger than 2 peripheral clock cycles between two successive transactions.

Figure 6.13-12 Auto Slave Select (SUSPITV ≧ 0x3)

Figure 6.13-13 Auto Slave Select (SUSPITV < 0x3)

6.13.5.8 Slave 3-wire Mode

When the SLV3WIRE (USPI_PROTCTL[1]) is set by software to enable the Slave 3-wire mode, the USCI SPI communication can work with no slave select signal in Slave mode. The SLV3WIRE (USPI_PROTCTL[1]) only takes effect in SPI Slave mode. Only three pins, SPI_CLK (through USCIx_CLK line), SPI_MOSI (through USCIx_DAT0 line), and SPI_MISO (through USCIx_DAT1 line), are required to communicate with a SPI Master. When the SLV3WIRE (USPI_PROTCTL[1])

is set to 1, the SPI Slave will be ready to transmit/receive data after the SPI protocol is enabled by setting FUNMODE(USPI_CTL [2:0]) to 0x1.

6.13.5.9 Data Transfer Mode

The USCI controller supports full-duplex SPI transfer.

In full-duplex SPI transfer, there are two data pins. One is used for transmitting data and the other is used for receiving data. Thus, data transmission and data reception can be performed simultaneously.

SCLKMODE (USPI_PROTCTL[7:6]) defines the transition timing of the data shift output signal on USCIx_DAT0 pin. The transition may happen at the corresponding edge of SPI bus clock or active edge of slave select signal. The level of the last data bit of a data word is held on USCIx_DAT0 pin until the next data word begins with the next corresponding edge of the serial bus clock.

6.13.5.10 Interrupt

Data Transfer Interrupts

Transmit start interrupt

The interrupt event TXSTIF (USPI_PROTSTS[1]) is set after the start of the first data bit of a transmit data word. It can be cleared only by writing 1 to it. The controller wil issue an interrupt if TXSTIEN (USPI_INTEN[1]) is also set to 1.

Transmit end interrupt

The interrupt event TXENDIF (USPI_PROTSTS[2]) is set after the start of the last data bit of the last transmit data which has been stored in transmit buffer. It can be cleared only by writing 1 to it. The controller wil issue an interrupt if TENDIEN (USPI_INTEN[2]) is also set to 1.

Receive start interrupt

The interrupt event RXSTIF (USPI_PROTSTS[3]) is set after the start of the first data bit of a receive data word. It can be cleared only by writing 1 to it. The controller wil issue an interrupt if RXSTIEN (USPI_INTEN[3]) is also set to 1.

Receive end interrupt

The interrupt event RXENDIF (USPI_PROTSTS[4]) is set after the start of the last data bit of a receive data word. It can be cleared only by writing 1 to it. The controller wil issue an interrupt if RXENDIEN (USPI_INTEN[4]) is also set to 1.

Protocol-Related Interrupts

SPI slave select interrupt

In SPI slave mode, there are slave select active and in-active interrupt flags, SSACTIF (USPI_PROTSTS[9]) and SSINAIF (USPI_PROTSTS[8]), will be set to 1 when SLAVE (USPI_PROTCTL [0]) is set to 1 and slave senses the slave select signal active or inactive. The SPI controller will issue an interrupt if SSINAIEN (USPI_PROTIEN[0]) or SSACTIEN (USPI_PROTIEN[1]), are set to 1. Because the internal slave select signal in SPI function is active high, the ININV (USPI_CTLIN0[2]) can be used for inverting the slave select signal comes from an active low device.

Slave time-out interrupt

In SPI slave mode, there is Slave time-out function for user to know that there is no serial clock input during the period of one word transaction. The Slave time-out function uses the timing measurement counter for the calculation of Slave time-out period which is defined by SLVTOCNT (USPI_PROTCTL[25:16]). TMCNTSRC (USPI_BRGEN[5]) can be used for clock frequency selection of timing measurement counter to calculate the Slave time-out period.

When the timing measurement counter is enabled by TMCNTEN (USPI_BRGEN[4]) and the setting value of SLVTOCNT (USPI_PROTCTL[25:16]) is not 0 in SPI Slave mode, the timing measurement counter will start counting after the first input serial clock of each received word data. This counter will be reset while receiving the following input serial clock and then keep counting. Finally, the timing measurement counter will be cleared and stopped after the finish of the current word transaction. If the value of the time-out counter is equal to or greater than the value of SLVTOCNT (USPI_PROTCTL[25:16]) before one word transaction is done, the Slave time-out interrupt event occurs and the SLVTOIF (USPI_PROTSTS[5]) will be set to 1.

Buffer-Related Interrupts

The buffer-related interrupts are available if there is transmit/receive buffer in USCI controller.

Receive buffer overrun interrupt

If there is receive buffer overrun event, RXOVIF (USPI_BUFSTS[3]) will be set as 1. It can be cleared by write 1 into it. The controller wil issue an interrupt if RXOVIEN (USPI_ BUFCTL[14]) is also set to 1.

Transmit buffer under-run interrupt

If there is transmit buffer under-run event, TXUDRIF (USPI_BUFSTS[11]) will be set as 1. It can be cleared by write 1 into it. The controller wil issue an interrupt if TXUDRIEN (USPI_ BUFCTL[6]) is also set to 1.

6.13.5.11 Timing Diagram

The slave select signal of USCI SPI protocol is active high by default, and it can be inverted by CTLOINV (USPI_LINECTL[7]) setting.

The idle state of serial bus clock and the serial bus clock edge used for transmit/receive data can be configured by setting SCLKMODE (USPI_PROTCTL[7:6]). The bit length of a transaction word data is determined by DWIDTH (USPI_LINECTL[11:8]), and data bit transfer sequence is determined by LSB (USPI_LINECTL[0]). Four SPI timing diagrams for Master/Slave operations and the related settings are shown as Figure 6.13-14, Figure 6.13-15, Figure 6.13-16 and Figure 6.13-17.

Figure 6.13-14 SPI Timing in Master Mode

Figure 6.13-15 SPI Timing in Master Mode (Alternate Phase of Serial Bus Clock)

Figure 6.13-16 SPI Timing in Slave Mode

Figure 6.13-17 SPI Timing in Slave Mode (Alternate Phase of Serial Bus Clock)

6.13.5.12 Programming flow

This section describes the programming flow for USCI SPI data transfer.

For Master mode:

1. Enable USCI peripheral clock by setting CLK_APBCLK register.
2. Configure user-specified pins as USCI function pins by setting corresponding multiple function control registers.

3. Set FUNMODE (USPI_CTL[2:0]) to 0x1 to select SPI mode.
4. Set USPI_BRGEN register to determine the SPI bus clock frequency.
5. According to the requirements of user's application, configured the settings as follows.
   - CTLOINV (USPI_LINECTL[7]): If the slave selection signal is active low, set this bit to 1; otherwise, set it to 0.
   - DWIDTH (USPI_LINECTL[11:8]): Data width setting.
   - LSB (USPI_LINECTL[0]): LSB first or MSB first.
   - TSMSEL (USPI_PROTCTL[14:12]): Full-duplex SPI transfer.
   - SCLKMODE (USPI_PROTCTL[7:6]): Determine the clock timing.
   - AUTOSS (USPI_PROTCTL[3]): Enable automatic slave select function or not.
   - SLAVE (USPI_PROTCTL[0]): Set to 0 for Master mode.
6. Set PROTEN (USPI_PROTCTL[31]) to 1 to enable SPI protocol.
7. If automatic slave select function is disabled (AUTOSS=0), set SS (USPI_PROTCTL[2]) to 1 before data transfer; set SS to 0 to inactivate the slave selection signal by user's application.
8. Write USPI_TXDAT register to trigger SPI transfer.
9. User can get the received data by reading USPI_RXDAT register as long as RXEMPTY (USPI_BUFSTS[0]) is 0. The SPI data transfer can be triggered by writing USPI_TXDAT register as long as TXFULL (USPI_BUFSTS[9]) is 0.

For Slave mode:

1. Enable USCI peripheral clock by setting CLK_APBCLK register.
2. Configure user-specified pins as USCI function pins by setting corresponding multiple function control registers.
3. Set FUNMODE (USPI_CTL[2:0]) to 0x1 to select SPI mode.
4. According to the requirements of user's application, configured the settings as follows.
   - ININV (USPI_CTLIN0[2]): If the slave selection signal is active low, set this bit to 1; otherwise, set it to 0.
   - DWIDTH (USPI_LINECTL[11:8]): Data width setting.
   - LSB (USPI_LINECTL[0]): LSB first or MSB first.
   - TSMSEL (USPI_PROTCTL[14:12]): Full-duplex SPI transfer
   - SCLKMODE (USPI_PROTCTL[7:6]): Determine the clock timing.
   - SLAVE (USPI_PROTCTL[0]): Set to 1 for Slave mode.
5. Set PROTEN (USPI_PROTCTL[31]) to 1 to enable SPI protocol.
6. Write USPI_TXDAT register for transmission.
7. User can get the received data by reading USPI_RXDAT register as long as RXEMPTY (USPI_BUFSTS[0]) is 0. The next datum for transmission can be written to USPI_TXDAT register as long as TXFULL (USPI_BUFSTS[9]) is 0.

6.13.5.13 Wake-up Function

The USCI Controller in SPI mode supports wake-up system function. The wake-up source in SPI protocol is the transition of input slave select signal.

6.13.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
USPI Base Address: **USPIx_BA = 0x4007_0000 + (0x10_0000 * x)** **x = 0, 1**				
USPI_CTL **x = 0, 1**	USPIx_BA+0x00	R/W	USCI Control Register	0x0000_0000
USPI_INTEN **x = 0, 1**	USPIx_BA+0x04	R/W	USCI Interrupt Enable Register	0x0000_0000
USPI_BRGEN **x = 0, 1**	USPIx_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
USPI_DATIN0 **x = 0, 1**	USPIx_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	0x0000_0000
USPI_CTLIN0 **x = 0, 1**	USPIx_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	0x0000_0000
USPI_CLKIN **x = 0, 1**	USPIx_BA+0x28	R/W	USCI Input Clock Signal Configuration Register	0x0000_0000
USPI_LINECTL **x = 0, 1**	USPIx_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
USPI_TXDAT **x = 0, 1**	USPIx_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
USPI_RXDAT **x = 0, 1**	USPIx_BA+0x34	R	USCI Receive Data Register	0x0000_0000
USPI_BUFCTL **x = 0, 1**	USPIx_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000
USPI_BUFSTS **x = 0, 1**	USPIx_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101
USPI_WKCTL **x = 0, 1**	USPIx_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
USPI_WKSTS **x = 0, 1**	USPIx_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
USPI_PROTCTL **x = 0, 1**	USPIx_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300
USPI_PROTIEN **x = 0, 1**	USPIx_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
USPI_PROTSTS **x = 0, 1**	USPIx_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

6.13.7 Register Description

USCI Control Register (USPI_CTL)

Register	Offset	R/W	Description	Reset Value
USPI_CTL **x = 0, 1**	USPIx_BA+0x00	R/W	USCI 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					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 0x0 = The USCI is disabled. All protocol related state machines are set to idle state. 0x1 = The SPI protocol is selected. 0x2 = The UART protocol is selected. 0x4 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Interrupt Enable Register (USPI_INTEN)

Register	Offset	R/W	Description	Reset Value
USPI_INTEN **x = 0, 1**	USPIx_BA+0x04	R/W	USCI 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved			**RXENDIEN**	**RXSTIEN**	**TXENDIEN**	**TXSTIEN**	**Reserved**

Bits	Description	
[31:5]	**Reserved**	Reserved.
[4]	**RXENDIEN**	**Receive End Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive finish event. 0 = The receive end interrupt Disabled. 1 = The receive end interrupt Enabled.
[3]	**RXSTIEN**	**Receive Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a receive start event. 0 = The receive start interrupt Disabled. 1 = The receive start interrupt Enabled.
[2]	**TXENDIEN**	**Transmit End Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit finish event. 0 = The transmit finish interrupt Disabled. 1 = The transmit finish interrupt Enabled.
[1]	**TXSTIEN**	**Transmit Start Interrupt Enable Bit** This bit enables the interrupt generation in case of a transmit start event. 0 = The transmit start interrupt Disabled. 1 = The transmit start interrupt Enabled.
[0]	**Reserved**	Reserved.

USCI Baud Rate Generator Register (USPI_BRGEN)

Register	Offset	R/W	Description	Reset Value
USPI_BRGEN x = 0, 1	USPIx_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						**CLKDIV**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CLKDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TMCNTSRC**	**TMCNTEN**	**SPCLKSEL**		**PTCLKSEL**	**RCLKSEL**

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} ($f_{DIV_CLK} = f_{PROT_CLK}$ / (CLKDIV+1)).
[15:10]	**Reserved**	Reserved.
[9:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Time Measurement Counter Clock Source Selection** 0 = Time measurement counter with $f_{PROT_CLK.}$ 1 = Time measurement counter with $f_{DIV_CLK.}$
[4]	**TMCNTEN**	**Time Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Time measurement counter Disabled. 1 = Time measurement counter Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection** This bit field used for the clock source selection of sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = $f_{DIV_CLK.}$ 01 = $f_{PROT_CLK.}$ 10 = $f_{SCLK.}$ 11 = $f_{REF_CLK.}$
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source of protocol clock (f_{PROT_CLK}). 0 = Reference clock $f_{REF_CLK.}$ 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).

[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source of reference clock (f_{REF_CLK}). 0 = Peripheral device clock $f_{PCLK.}$ 1 = HXT/LXT.

USCI Input Data Signal Configuration (USPI_DATIN0)

Register	Offset	R/W	Description	Reset Value
USPI_DATIN0 **x = 0, 1**	USPIx_BA+0x10	R/W	USCI Input Data Signal Configuration Register 0	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					**ININV**	**Reserved**	**SYNCSEL**

Bits	Description	
[31:0]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted. **Note:** In SPI protocol, we suggested this bit should be set as 0.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Signal Synchronization Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested that the bit should be set as 0.

USCI Input Control Signal Configuration (USPI_CTLIN0)

Register	Offset	R/W	Description	Reset Value
USPI_CTLIN0 **x = 0, 1**	USPIx_BA+0x20	R/W	USCI Input Control Signal Configuration Register 0	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					**ININV**	**Reserved**	**SYNCSEL**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**ININV**	**Input Signal Inverse Selection** This bit defines the inverter enable of the input asynchronous signal. 0 = The un-synchronized input signal will not be inverted. 1 = The un-synchronized input signal will be inverted.
[1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal (with optionally inverted) or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested that the bit should be set as 0.

USCI Input Clock Signal Configuration (USPI_CLKIN)

Register	Offset	R/W	Description	Reset Value
USPI_CLKIN x = 0, 1	USPIx_BA+0x28	R/W	USCI Input Clock Signal Configuration 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							**SYNCSEL**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**SYNCSEL**	**Input Synchronization Signal Selection** This bit selects if the un-synchronized input signal or the synchronized (and optionally filtered) signal can be used as input for the data shift unit. 0 = The un-synchronized signal can be taken as input for the data shift unit. 1 = The synchronized signal can be taken as input for the data shift unit. **Note:** In SPI protocol, it is suggested that the bit should be set as 0.

USCI Line Control Register (USPI_LINECTL)

Register	Offset	R/W	Description	Reset Value
USPI_LINECTL **x = 0, 1**	USPIx_BA+0x2C	R/W	USCI 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				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
CTLOINV	**Reserved**	**DATOINV**	**Reserved**				**LSB**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0].
[7]	**CTLOINV**	**Control Signal Output Inverse Selection** This bit defines the relation between the internal control signal and the output control signal. 0 = No effect. 1 = The control signal will be inverted before its output. **Note:** The control signal has different definitions in different protocol. In SPI protocol, the control signal means slave select signal.
[6]	**Reserved**	Reserved.
[5]	**DATOINV**	**Data Output Inverse Selection** This bit defines the relation between the internal shift data value and the output data signal of USCIx_DAT0/1 pin. 0 = Data output level is not inverted. 1 = Data output level is inverted.

[4:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (USPI_TXDAT)

Register	Offset	R/W	Description	Reset Value
USPI_TXDAT **x = 0, 1**	USPIx_BA+0x30	W	USCI Transmit 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**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission. In order to avoid overwriting the transmit data, user have to check TXEMPTY (USPI_BUFSTS[8]) status before writing transmit data into this bit field.

USCI Receive Data Register (USPI_RXDAT)

Register	Offset	R/W	Description	Reset Value
USPI_RXDAT **x = 0, 1**	USPIx_BA+0x34	R	USCI Receive 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**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer.

USCI Transmit/Receive Buffer Control Register (USPI_BUFCTL)

Register	Offset	R/W	Description	Reset Value
USPI_BUFCTL **x = 0, 1**	USPIx_BA+0x38	R/W	USCI Transmit/Receive Buffer Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved						**RXRST**	**TXRST**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
RXCLR	**RXOVIEN**	**Reserved**					
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXCLR	**TXUDRIEN**	**Reserved**					

Bits	Description	
[31:18]	**Reserved**	Reserved.
[17]	**RXRST**	**Receive Reset** 0 = No effect. 1 = Reset the receive-related counters, state machine, and the content of receive shift register and data buffer. **Note:** It is cleared automatically after one PCLK cycle.
[16]	**TXRST**	**Transmit Reset** 0 = No effect. 1 = Reset the transmit-related counters, state machine, and the content of transmit shift register and data buffer**.** **Note:** It is cleared automatically after one PCLK cycle.
[15]	**RXCLR**	**Clear Receive Buffer** 0 = No effect. 1 = The receive buffer is cleared. Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.
[14]	**RXOVIEN**	**Receive Buffer Overrun Interrupt Enable Bit** 0 = Receive overrun interrupt Disabled. 1 = Receive overrun interrupt Enabled.
[13:8]	**Reserved**	Reserved.
[7]	**TXCLR**	**Clear Transmit Buffer** 0 = No effect. 1 = The transmit buffer is cleared. Should only be used while the buffer is not taking part in data traffic. **Note:** It is cleared automatically after one PCLK cycle.

[6]	**TXUDRIEN**	**Slave Transmit Under-run Interrupt Enable Bit** 0 = Transmit under-run interrupt Disabled. 1 = Transmit under-run interrupt Enabled.
[5:0]	**Reserved**	Reserved.

USCI Transmit/Receive Buffer Status Register (USPI_BUFSTS)

Register	Offset	R/W	Description	Reset Value
USPI_BUFSTS x = 0, 1	USPIx_BA+0x3C	R	USCI Transmit/Receive Buffer Status Register	0x0000_0101

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				**TXUDRIF**	**Reserved**	**TXFULL**	**TXEMPTY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**RXOVIF**	**Reserved**	**RXFULL**	**RXEMPTY**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11]	**TXUDRIF**	**Transmit Buffer Under-run Interrupt Status** This bit indicates that a transmit buffer under-run event has been detected. If enabled by TXUDRIEN (USPI_BUFCTL[6]), the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit 0 = A transmit buffer under-run event has not been detected. 1 = A transmit buffer under-run event has been detected.
[10]	**Reserved**	Reserved.
[9]	**TXFULL**	**Transmit Buffer Full Indicator** 0 = Transmit buffer is not full. 1 = Transmit buffer is full.
[8]	**TXEMPTY**	**Transmit Buffer Empty Indicator** 0 = Transmit buffer is not empty. 1 = Transmit buffer is empty and available for the next transmission datum.
[7:4]	**Reserved**	Reserved.
[3]	**RXOVIF**	**Receive Buffer Overrun Interrupt Status** This bit indicates that a receive buffer overrun event has been detected. If RXOVIEN (USPI_BUFCTL[14]) is enabled, the corresponding interrupt request is activated. It is cleared by software writes 1 to this bit. 0 = A receive buffer overrun event has not been detected. 1 = A receive buffer overrun event has been detected.
[2]	**Reserved**	Reserved.
[1]	**RXFULL**	**Receive Buffer Full Indicator** 0 = Receive buffer is not full.

		1 = Receive buffer is full.

USCI Wake-up Control Register (USPI_WKCTL)

Register	Offset	R/W	Description	Reset Value
USPI_WKCTL **x = 0, 1**	USPIx_BA+0x54	R/W	USCI 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					**PDBOPT**	**Reserved**	**WKEN**

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2]	**PDBOPT**	**Power Down Blocking Option** 0 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, MCU will stop the transfer and enter Power-down mode immediately. 1 = If user attempts to enter Power-down mode by executing WFI while the protocol is in transferring, the on-going transfer will not be stopped and MCU will enter idle mode immediately.
[1]	**Reserved**	Reserved.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (USPI_WKSTS)

Register	Offset	R/W	Description	Reset Value
USPI_WKSTS **x = 0, 1**	USPIx_BA+0x58	R/W	USCI 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							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – SPI (USPI_PROTCTL)

Register	Offset	R/W	Description	Reset Value
USPI_PROTCTL **x = 0, 1**	USPIx_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0300

31	30	29	28	27	26	25	24
PROTEN	**Reserved**		**TXUDRPOL**	**Reserved**		**SLVTOCNT**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
SLVTOCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**TSMSEL**			**SUSPITV**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SCLKMODE		**Reserved**		**AUTOSS**	**SS**	**SLV3WIRE**	**SLAVE**

Bits	Description	
[31]	**PROTEN**	**SPI Protocol Enable Bit** 0 = SPI Protocol Disabled. 1 = SPI Protocol Enabled.
[30:29]	**Reserved**	Reserved.
[28]	**TXUDRPOL**	**Transmit Under-run Data Polarity (for Slave)** This bit defines the transmitting data level when no data is available for transferring. 0 = The output data level is 0 if TX under-run event occurs. 1 = The output data level is 1 if TX under-run event occurs.
[27:26]	**Reserved**	Reserved.
[25:16]	**SLVTOCNT**	**Slave Mode Time-out Period (Slave Only)** In Slave mode, this bit field is used for Slave time-out period. This bit field indicates how many clock periods (selected by TMCNTSRC, USPI_BRGEN[5]) between the two edges of input SCLK will assert the Slave time-out event. Writing 0x0 into this bit field will disable the Slave time-out function. Example: Assume SLVTOCNT is 0x0A and TMCNTSRC (USPI_BRGEN[5]) is 1, it means the time-out event will occur if the state of SPI bus clock pin is not changed more than (10+1) periods of f_{DIV_CLK}.
[15]	**Reserved**	Reserved.
[14:12]	**TSMSEL**	**Transmit Data Mode Selection** This bit field describes how receive and transmit data is shifted in and out. 0x0 = Full-duplex SPI. Other values are reserved. **Note:** Changing the value of this bit field will produce the TXRST and RXRST to clear the TX/RX data buffer automatically.
[11:8]	**SUSPITV**	**Suspend Interval (Master Only)** This bit field provides the 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. (SUSPITV[3:0] + 0.5) * period of SPI_CLK clock cycle Example: SUSPITV = 0x0 …. 0.5 SPI_CLK clock cycle. SUSPITV = 0x1 …. 1.5 SPI_CLK clock cycle. …… SUSPITV = 0xE …. 14.5 SPI_CLK clock cycle. SUSPITV = 0xF …. 15.5 SPI_CLK clock cycle.
[7:6]	**SCLKMODE**	**Serial Bus Clock Mode** This bit field defines the SCLK idle status, data transmit, and data receive edge. MODE0 = The idle state of SPI clock is low level. Data is transmitted with falling edge and received with rising edge. MODE1 = The idle state of SPI clock is low level. Data is transmitted with rising edge and received with falling edge. MODE2 = The idle state of SPI clock is high level. Data is transmitted with rising edge and received with falling edge. MODE3 = The idle state of SPI clock is high level. Data is transmitted with falling edge and received with rising edge.
[5:4]	**Reserved**	Reserved.
[3]	**AUTOSS**	**Automatic Slave Select Function Enable (Master Only)** 0 = Slave select signal will be controlled by the setting value of SS (USPI_PROTCTL[2]) bit. 1 = Slave select signal will be generated automatically. The slave select signal will be asserted by the SPI controller when transmit/receive is started, and will be de-asserted after each transmit/receive is finished.
[2]	**SS**	**Slave Select Control (Master Only)** If AUTOSS bit is cleared, setting this bit to 1 will set the slave select signal to active state, and setting this bit to 0 will set the slave select signal back to inactive state. If the AUTOSS function is enabled (AUTOSS = 1), the setting value of this bit will not affect the current state of slave select signal. **Note:** In SPI protocol, the internal slave select signal is active high.
[1]	**SLV3WIRE**	**Slave 3-wire Mode Selection (Slave Only)** The SPI protocol can work with 3-wire interface (without slave select signal) in Slave mode. 0 = 4-wire bi-direction interface. 1 = 3-wire bi-direction interface.
[0]	**SLAVE**	**Slave Mode Selection** 0 = Master mode. 1 = Slave mode.

USCI Protocol Interrupt Enable Register – SPI (USPI_PROTIEN)

Register	Offset	R/W	Description	Reset Value
USPI_PROTIEN x = 0, 1	USPIx_BA+0x60	R/W	USCI Protocol 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved				**SLVBEIEN**	**SLVTOIEN**	**SSACTIEN**	**SSINAIEN**

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3]	**SLVBEIEN**	**Slave Mode Bit Count Error Interrupt Enable Bit** If data transfer is terminated by slave time-out or slave select inactive event in Slave mode, so that the transmit/receive data bit count does not match the setting of DWIDTH (USPI_LINECTL[11:8]). Bit count error event occurs. 0 = Slave mode bit count error interrupt Disabled. 1 = Slave mode bit count error interrupt Enabled.
[2]	**SLVTOIEN**	**Slave Time-out Interrupt Enable Bit** In SPI protocol, this bit enables the interrupt generation in case of a Slave time-out event. 0 = Slave time-out interrupt Disabled. 1 = Slave time-out interrupt Enabled.
[1]	**SSACTIEN**	**Slave Select Active Interrupt Enable Bit** This bit enables/disables the generation of a slave select interrupt if the slave select changes to active. 0 = Slave select active interrupt generation Disabled. 1 = Slave select active interrupt generation Enabled.
[0]	**SSINAIEN**	**Slave Select Inactive Interrupt Enable Bit** This bit enables/disables the generation of a slave select interrupt if the slave select changes to inactive. 0 = Slave select inactive interrupt generation Disabled. 1 = Slave select inactive interrupt generation Enabled.

USCI Protocol Status Register – SPI (USPI_PROTSTS)

Register	Offset	R/W	Description	Reset Value
USPI_PROTSTS x = 0, 1	USPIx_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**SLVUDR**	**BUSY**	**SSLINE**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved						**SSACTIF**	**SSINAIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**SLVBEIF**	**SLVTOIF**	**RXENDIF**	**RXSTIF**	**TXENDIF**	**TXSTIF**	**Reserved**

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**SLVUDR**	**Slave Mode Transmit Under-run Status (Read Only)** In Slave mode, if there is no available transmit data in buffer while transmit data shift out caused by input serial bus clock, this status flag will be set to 1. This bit indicates whether the current shift-out data of word transmission is switched to TXUDRPOL (USPI_PROTCTL[28]) or not. 0 = Slave transmit under-run event does not occur. 1 = Slave transmit under-run event occurs.
[17]	**BUSY**	**Busy Status (Read Only)** 0 = SPI is in idle state. 1 = SPI is in busy state. The following listing are the bus busy conditions: a. USPI_PROTCTL[31] = 1 and the TXEMPTY = 0. b. For SPI Master mode, the TXEMPTY = 1 but the current transaction is not finished yet. c. For SPI Slave mode, the USPI_PROTCTL[31] = 1 and there is serial clock input into the SPI core logic when slave select is active. **d.** For SPI Slave mode, the USPI_PROTCTL[31] = 1 and the transmit buffer or transmit shift register is not empty even if the slave select is inactive.
[16]	**SSLINE**	**Slave Select Line Bus Status (Read Only)** This bit is only available in Slave mode. It used to monitor the current status of the input slave select signal on the bus. 0 = The slave select line status is 0. 1 = The slave select line status is 1.
[15:10]	**Reserved**	Reserved.
[9]	**SSACTIF**	**Slave Select Active Interrupt Flag (for Slave Only)** This bit indicates that the internal slave select signal has changed to active. It is cleared by

		software writes one to this bit 0 = The slave select signal has not changed to active. 1 = The slave select signal has changed to active. **Note:** The internal slave select signal is active high.
[8]	**SSINAIF**	**Slave Select Inactive Interrupt Flag (for Slave Only)** This bit indicates that the internal slave select signal has changed to inactive. It is cleared by software writes 1 to this bit 0 = The slave select signal has not changed to inactive. 1 = The slave select signal has changed to inactive. **Note:** The internal slave select signal is active high.
[7]	**Reserved**	Reserved.
[6]	**SLVBEIF**	**Slave Bit Count Error Interrupt Flag (for Slave Only)** 0 = Slave bit count error event does not occur. 1 = Slave bit count error event occurs. **Note:** It is cleared by software writes 1 to this bit.
[5]	**SLVTOIF**	**Slave Time-out Interrupt Flag (for Slave Only)** 0 = Slave time-out event does not occur. 1 = Slave time-out event occurs. **Note:** It is cleared by software writes 1 to this bit
[4]	**RXENDIF**	**Receive End Interrupt Flag** 0 = Receive end event does not occur. 1 = Receive end event occurs. **Note:** It is cleared by software writes 1 to this bit
[3]	**RXSTIF**	**Receive Start Interrupt Flag** 0 = Receive start event does not occur. 1 = Receive start event occurs. **Note:** It is cleared by software writes 1 to this bit
[2]	**TXENDIF**	**Transmit End Interrupt Flag** 0 = Transmit end event does not occur. 1 = Transmit end event occurs. **Note:** It is cleared by software writes 1 to this bit
[1]	**TXSTIF**	**Transmit Start Interrupt Flag** 0 = Transmit start event does not occur. 1 = Transmit start event occurs. **Note:** It is cleared by software writes 1 to this bit
[0]	**Reserved**	Reserved.

6.14 USCI – I²C Mode

6.14.1 Overview

On I²C bus, 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. 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 Figure 6.14-1 for more detailed I²C BUS Timing.

Figure 6.14-1 I²C Bus Timing

The device on-chip I²C provides the serial interface that meets the I²C bus standard mode specification. The I²C port handles byte transfers autonomously. The I²C mode is selected by FUNMODE (UI2C_CTL [2:0]) = 0100B. When this port is enabled, the USCI interfaces to the I²C bus via two pins: SDA and SCL. When I/O pins are used as I²C ports, user must set the pins function to I²C in advance.

Note: A pull-up resistor is needed for I²C operation because the SDA and SCL are set to open-drain pins when USCI is selected to I²C operation mode.

6.14.2 Features

- Full master and slave device capability
- Supports of 7-bit addressing, as well as 10-bit addressing
- Communication in standard mode (100 kBit/s) or in fast mode (up to 400 kBit/s)
- Supports multi-master bus
- Supports one transmit buffer and two receive buffer for data payload
- Supports 10-bit bus time-out capability
- Supports Power down wake-up by data toggle or address match
- Supports setup/hold time programmable

6.14.3 Block Diagram

Figure 6.14-2 USCI - I²C Mode Block Diagram

6.14.4 Basic Configuration

The basic configurations of USCI0 for I2C are as follows:

- USCI0 pins are configured on SYS_GPA_MFP, SYS_GPC_MFP, SYS_GPD_MFP registers.
- Enable USCI0 clock (USCI0CKEN) on CLK_APBCLK[24] register.
- Reset USCI0 controller (USCI0RST) on SYS_IPRST1[24] register.
- Enable I2C function (FUNMODE=100) on (UI2C_CTL[2:0]) register.

The basic configurations of USCI1 for I^2C are as follows:

- USCI1 pins are configured on SYS_GPA_MFP, SYS_GPC_MFP, SYS_GPD_MFP registers.
- Enable USCI1 clock (USCI0CKEN) on CLK_APBCLK[25] register.
- Reset USCI1 controller (USCI0RST) on SYS_IPRST1[25] register.
- Enable I2C function (FUNMODE=100) on (UI2C_CTL[2:0]) register.

6.14.5 Functional Description

6.14.5.1 USCI Common Function Description

Please refer to section 6.13.4 for detailed information.

6.14.5.2 START or Repeated START Signal

Figure 6.14-3 shows the typical I^{2}C protocol. Normally, a standard communication consists of four parts:

- START or Repeated START signal generation
- Slave address and R/W bit transfer
- Data transfer
- STOP signal generation

Figure 6.14-3 I^{2}C Protocol

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 is not a STOP signal between two START signals and usually referred to as the “Sr” bit. 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 idle flag.

6.14.5.3 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. The section between STOP and START is called bus free. Figure 6.14-4 shows the waveform of START, Repeat START and STOP.

Figure 6.14-4 START and STOP Conditions

6.14.5.4 Slave Address Transfer

After a (repeated) start condition, the master sends a slave address to identify the target device of the communication. The start address can comprise one or two address bytes (for 7-bit or for 10-bit addressing schemes). After an address byte, a slave sensitive to the transmitted address has to acknowledge the reception.

Therefore, the slave's address can be programmed in the device, where it is compared to the received address. In case of a match, the slave answers with an acknowledge (SDA = 0). Slaves that are not targeted answer with a non-acknowledge (SDA = 1). In addition to the match of the programmed address, another address byte value has to be answered with an acknowledge if the slave is capable to handle the corresponding requests. The address byte 00H indicates a general call address that can be acknowledged.

To allow selective acknowledges for the different values of the address byte(s), the following control mechanism is implemented:

- If the GCFUNC bit (UI2C_PROTCTL [0]) is set the I^2C port hardware will respond to General Call address (00H). Clear GC bit to disable general call function.

- The I^2C port is equipped with one device address registers, UI2C_DEVADDR0. In 7-bit address mode, the first 7 bits of a received first address byte are compared to the programmed slave address (UI2C_DEVADDR0 [6:0]). If these bits match, the slave sends an acknowledge.

- In addition, if the slave address is programmed to 1111 0XXB, the XX bits are compared to the bits UI2C_DEVADDR0 [9:8] to check for address match and also sends an acknowledge when ADDR10EN (UI2C_PROTCTL [4]) is set. The slave waits for a second address byte compares it with UI2C_DEVADDR0 [7:0] and sends an acknowledge accordingly to cover the 10 bit addressing mode. The user has to consider about reserved addresses (refer to I^2C specification for more detailed description). Only the address 1111 0XXB is supported. Under each of these conditions, bit SLASEL (UI2C_PROTSTS [14]) will be set when the addressing delivered a match. This SLASEL (UI2C_PROTSTS [14]) bit is cleared automatically by a (repeated) start or stop condition.

- The I^2C port is equipped multiple address recognition with one address mask registers I2C_ADDRMSKn (n = 0). When the bit in the address mask register is set to 1, it means the received corresponding address bit is "Don't care". If the bit is set to 0, it means the received corresponding register bit should be exactly the same as address register.

6.14.5.5 Data Transfer

When a slave receives a correct address with an R/W bit, the data will follow R/W 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.

Figure 6.14-5 Bit Transfer on I^2C Bus

If the master received data, does Not Acknowledge (NACK) the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal.

Figure 6.14-6 Acknowledge on I^2C Bus

6.14.5.6 Clock Baud Rate Bits

For this section, please refer to Figure 6.11-9. The data baud rate of I^2C is determines by UI2C_BRGEN register when I^2C is in Master Mode, and it is not necessary in a Slave mode. In the Slave mode, I^2C will automatically synchronize it with any clock frequency from master I^2C device. The bits RCLKSEL, SPCLKSEL, PDSCNT, and DSCNT define the baud rate setting:

• RCLKSEL (UI2C_BRGEN [0])

to define the input frequency f_{REF_CLK}

•SPCLKSEL (UI2C_BRGEN[3:2])

to define the multiple source of the sample clock f_{SAMP_CLK}

•PDSCNT (UI2C_BRGEN [9:8])

to define the length of a data sample time (division of f_{REF_CLK} by 1, 2, 3, or 4)

• DSCNT (UI2C_BRGEN [14:10])

to define the number of data sample time per bit time

The standard setting is given by RCLKSEL = 0 ($f_{REF_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT_CLK} = f_{REF_CLK}$) and SPCLKSEL = 2'b00 ($f_{SAMP_CLK} = f_{DIV_CLK}$). Under these conditions, the baud rate is given by:

$$f_{\mathrm{I2C}} = \frac{f_{REF_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

To generate slower frequencies, additional divide-by-2 stages can be selected by PTCLKSEL = 1 ($f_{PROT_CLK} = f_{REF_CLK2}$), leading to:

$$f_{\mathrm{I2C}} = \frac{f_{REF_CLK}}{4} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

If SPCLKSEL = 2'b10 ($f_{SAMP_CLK} = f_{SCLK}$), and RCLKSEL = 0 ($f_{REF_CLK} = f_{PCLK}$), PTCLKSEL = 0 ($f_{PROT_CLK} = f_{REF_CLK}$). The baud rate is given by:

$$f_{\mathrm{I2C}} = \frac{f_{REF_CLK}}{2} \times \frac{1}{\mathrm{CLKDIV}+1} \times \frac{1}{2} \times \frac{1}{\mathrm{PDSCNT}+1} \times \frac{1}{\mathrm{DSCNT}+1}$$

6.14.5.7 Byte Stretching

If a device is selected as transceiver and should transmit a data byte but the transmit buffer TXDAT does not contain valid data to be transmitted, the device ties down SCL = 0 at the end of the previous acknowledge bit. The waiting period is finished if software writes 1 to PTRG (UI2C_PROTCTL [5]).

6.14.5.8 Master Arbitration

In some applications, there are two or more masters on the same I^2C bus to access slaves, and the masters may transmit data simultaneously. The I^2C supports multi-master by including collision detection and arbitration to prevent data corruption.

If two masters sometimes initiate I^2C command at the same time, the arbitration procedure determines which master wins and can continue with the command. Arbitration is performed on the SDA signal while the SCL signal is high. Each master checks if the SDA signal on the bus corresponds to the generated SDA signal. If the SDA signal on the bus is low but it should be high, then this master has lost arbitration. Master I^2C device that has lost arbitration can generate SCL pulses until the byte ends and must then release the bus and go into slave mode. The arbitration procedure can continue until all the data is transferred. This means that in multi-master system each I^2C master must monitor the I^2C bus for collisions and act accordingly. Figure 6.14-7 describes DATA1 and DATA2 are compete arbitration.

Figure 6.14-7 Arbitration Lost

In this case, during the address and data transmission, the master transmitter checks at the rising edge of SCL for each data bit if the value it is sending is equal to the value read on the SDA line. If yes, the next data bit values can be 0. If this is not the case (transmitted value = 1, value read = 0), the master has lost the transmit arbitration. This is indicated by interrupt flag ARBLOIF (UI2C_PROTSTS [11]) and can generate a protocol interrupt if enabled by ARBLOIEN (UI2C_PROTIEN [4]).

When the transmit arbitration has been lost, the software has to initialize the complete frame again, starting with the first address byte together with the start condition for a new master transmit attempt. Arbitration also takes place for the ACK bit. If master arbitration lost and match the device address, then master will turn to slave.

6.14.5.9 Transmission Chain

The I^2C bus protocol requiring a kind of in-bit-response during the arbitration phase and while a slave is transmitting, the resulting loop delay of the transmission chain can limit the reachable maximal baud rate, strongly depending on the bus characteristics (bus load, module frequency, etc.).

The shift clock SCL is generated by the master device, output on the wire, then it passes through the input stage and the input filter. Now, the edges can be detected and the SDA data signal can be generated accordingly. The SDA signal passes through the output stage and the wire to the master receiver part. There, it passes through the input stage and the input filter before it is sampled.

This complete loop has to be finished (including all settling times to obtain stable signal levels) before the SCL signal changes again. The delays in this path have to be taken into account for the calculation of the baud rate as a function of f_{PCLK} and f_{PROT_CLK}.It is suggested that user adopts f_{PCLK}.

6.14.5.10 Non-Acknowledge and Error Conditions

In case of a non-acknowledge (NACKIF (UI2C_PROTSTS [10])) or an error (ERRIF(UI2C_PROTSTS [12])), no further transmission will take place. User software doesn't invalidate the transmit buffer and disable transmissions, before configuring the transmission (by writing TXDAT) again with appropriate values to react on the previous event.

I²C Protocol Interrupt Events

The following protocol-related events are generated in I²C mode and can lead to a protocol interrupt.

Please note that the bits in register UI2C_PROTSTS are not all automatically cleared by hardware and have to be cleared by software to monitor new incoming events.

- Start condition received at a correct position in a frame (STARIF (UI2C_PROTSTS [8]))
- Stop condition transferred at a correct position in a frame (STORIF (UI2C_PROTSTS [9]))
- Master arbitration lost (ARBLOIF (UI2C_PROTSTS [11]))
- Slave read requested (SLAREAD (UI2C_PROTSTS [15]))
- Acknowledge received (ACKIF (UI2C_PROTSTS [13]))
- Non-acknowledge received (NACKIF (UI2C_PROTSTS [10]))
- Start condition not at the expected position in a frame (ERRIF (UI2C_PROTSTS [12]))
- Stop condition not at the expected position in a frame (ERRIF (UI2C_PROTSTS [12]))

6.14.5.11 Operating the I²C

To operate the I²C protocol, the following issues have to be considered:

Select I²C mode

It is recommended to configure all parameters of the I²C that do not change during run time while FUNMODE (UI2C_CTL [2:0]) = 000B. The I²C control flow has to be done while FUNMODE (UI2C_CTL [2:0]) = 000B to avoid unintended edges of the input signals and the I²C mode can be enabled by FUNMODE (UI2C_CTL [2:0]) = 100B afterwards.

-Step 1. Set FUNMODE (UI2C_CTL [2:0]) = 000B

-Step 2. Set FUNMODE (UI2C_CTL [2:0]) = 100B

Pin connections

The pins used for SDA and SCL have to be set to open-drain mode by USCI controller to support the wired-AND structure of the I²C bus lines.

Note: The step to enable the alternate output port functions should only be done after the I²C mode is enabled, to avoided unintended spikes on the output.

Bit timing configuration

In standard mode (100 kBit/s) a minimum module frequency of 2 MHz is necessary, whereas in fast mode (400 kBit/s) a minimum of 10 MHz is required. Additionally, if the digital filter stage should be used to eliminate spikes up to 50 ns, a filter frequency of 20 MHz is necessary. There could be an uncertainty in the SCL high phase timing of maximum $1/f_{PROT_CLK}$ if another I²C participant lengthens the SCL low phase on the bus. Note that the SCL maximum frequency is SAMP_CLK/2 and the SPCLKSEL (UI2C_BRGEN [3:2]) must be set 0 for selecting $f_{SAMP_CLK} = f_{DIV_CLK}$.

Data format configuration

The data format has to be configured for 8 data bits (DWIDTH (UI2C_LINECTL [11:8]) = 8), and MSB shifted first (LSB (UI2C_LINECTL [0]) = 0). As a result, UI2C_LINECTL has to be set to 0x800.

Control flow

The on-chip I^{2}C ports support three operation modes, Master, Slave, and General Call Mode.

In a given application, I^{2}C port may operate as a master or as a slave. In Slave mode, the I^{2}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 pulse 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 address arbitration is lost in Master mode, I^{2}C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer.

To control the I^{2}C bus transfer in each mode, user needs to set UI2C_PROTCTL, UI2C_PROTIEN, TXDAT registers according to current status of UI2C_PROTSTS register. In other words, for each I^{2}C bus action, user needs to check current status by UI2C_PROTSTS register, and then set UI2C_PROTCTL, UI2C_PROTIEN, TXDAT registers to take bus action. Finally, check the response status by UI2C_PROTSTS.

The bits, STA, STO and AA in UI2C_PROTCTL register are used to control the next state of the I^{2}C hardware after interrupt signal is cleared. Upon completion of the new action, a new status will be updated in UI2C_PROTSTS register will be set. If the I^{2}C interrupt control bit of UI2C_PROTIEN is set, appropriate action or software branch of the new status can be performed in the Interrupt service routine.

Figure 6.14-8 shows the current I^{2}C STARIF (UI2C_PROTSTS [8]) is set to 1 by hardware, and then set TXDAT = SLA+W (Slave address + Write bit), (PTRG, STA, STO, AA) = (1, 0, 0, x) to send the address to I^{2}C bus, and write 1 to STARIF (UI2C_PROTSTS [8]) to clear flag. If a slave on the bus matches the address and response ACK, the UI2C_PROTSTS will be updated by ACKIF (UI2C_PROTSTS [13]) setting.

Figure 6.14-8 Control I^{2}C Bus according to Current I^{2}C Status

Data Transfer on the I²C Bus

Figure 6.14-9 shows a master transmits data to slave. A master addresses a slave with a 7-bit address and 1-bit write index to denote that the master wants to transmit data to the slave. The master keeps transmitting data after the slave returns acknowledge to the master.

Figure 6.14-9 Master Transmits Data to Slave with a 7-bit Address

Figure 6.14-10 shows a master read data from slave. A master addresses a slave with a 7-bit address and 1-bit read index to denote that the master wants to read data from the slave. The slave will start transmitting data after the slave returns acknowledge to the master.

Figure 6.14-10 Master Reads Data from Slave with a 7-bit Address

Figure 6.14-11 shows a master transmits data to slave by 10-bit address. A master addresses a slave with a 10-bit address. First byte contains 10-bit address indicator (5'b11110) and 2-bit address with write index, second byte contains 8-bit address. The master keeps transmitting data after the second byte end. Note that 7-bit and 10-bit address device can work on the same bus.

Figure 6.14-11 Master Transmits Data to Slave by 10-bit Address

Figure 6.14-12 shows a master read data from slave by 10-bit address. A master addresses a slave with a 10-bit address. First mater transmits 10-bit address to slave, after that master transmits first byte with read index. The slave will start transmitting data after the first byte with read index.

Figure 6.14-12 Master Reads Data from Slave by 10-bit Address

Master Mode

In Figure 6.14-13, all possible protocols for I^{2}C master are shown. User needs to follow proper path of the flow to implement required I^{2}C protocol.

In other words, user can send a START signal to bus and I^{2}C will be in Master Transmitter mode (Figure 6.14-13) or Master receiver mode (Figure 6.14-14) after START signal has been sent successfully and new status register would be set STARIF (UI2C_PROTSTS [8]). Followed by START signal, user can send slave address, read/write bit, data and Repeat START, STOP to perform I^{2}C protocol.

Figure 6.14-13 Master Transmitter Mode Control Flow with 7-bit Address

Figure 6.14-14 Master Receiver Mode Control Flow with 7-bit Address

If the I^{2}C is in Master mode and gets arbitration lost, the bit of ARBLOIF (UI2C_PROTSTS [11]) will be set. User may writing 1 to ARBLOIF (UI2C_PROTSTS [11]) and set (PTRG, STA, STO, AA) = (1, 1, 0, X) to send START to re-start Master operation when bus become free. Otherwise, user may writing 1 to ARBLOIF (UI2C_PROTSTS [11]) and set (PTRG, STA, STO, AA) = (1, 0, 0, X) to release I^{2}C bus and enter not addressed Slave mode.

Slave Mode

When reset, I^{2}C is not addressed and will not recognize the address on I^{2}C bus. User can set device address by UI2C_DEVADDR0 and set (PTRG, STA, STO, AA) = (1, 0, 0, 1) to let I^{2}C recognize the address sent by master. Figure 6.14-15 shows all the possible flow for I^{2}C in Slave mode. Users need to follow a proper flow (as shown in Figure 6.14-15 to implement their own I^{2}C protocol.

If bus arbitration is lost in Master mode, I^{2}C port switches to Slave mode immediately and can detect its own slave address in the same serial transfer. If the detected address is SLA+W (Master want to write data to Slave) or SLA+R (Master want to read data from Slave) after

arbitration lost, the ARBLOIF will be set to 1.

Note: During I^{2}C communication, the SCL clock will be released when writing '1' to PTRG (UI2C_PROTCTL [5]) in Slave mode.

Figure 6.14-15 Save Mode Control Flow with 7-bit Address

If I^{2}C is still transmitting and receiving data in addressed Slave mode but got a STOP or Repeat

START, the register STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) will be set. User could follow the action for NACKIF (UI2C_PROTSTS [10]) as shown in Figure 6.14-15 when got STARIF (UI2C_PROTSTS [8]) is set.

Note: After slave gets interrupt flag of NACKIF (UI2C_PROTSTS [10]) and start/stop symbol including STARIF (UI2C_PROTSTS [8]) and STORIF (UI2C_PROTSTS [9]), slave can switch to not address mode and own SLA will not be recognized. If setting this interrupt flag, slave will not receive any I^2C signal or address from master. At this status, I^2C should be reset by setting FUNMODE (UI2C_CTL [2:0]) = 000B to leave this status.

General Call (GC) Mode

If the GCFUNC bit (UI2C_PROTCTL [0]) is set, the I^2C port hardware will respond to General Call address (00H). User can clear GC bit to disable general call function. When the GC bit is set and the I^2C in slave mode, it can receive the general call address by 0x00 after master send general call address to I^2C bus, and then it also will follow protocol status register.

Figure 6.14-16 GC Mode with 7-bit Address

If I^2C is still receiving data in GC mode but got a STOP or Repeat START, the STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) will be set. User could follow the action for NACKIF (UI2C_PROTSTS [10]) in Figure 6.14-16when got STORIF (UI2C_PROTSTS [9]) or STARIF (UI2C_PROTSTS [8]) is set.

Note: After slave gets interrupt flag of NACKIF (UI2C_PROTSTS [10]) and start/stop symbol including STARIF (UI2C_PROTSTS [8]) and STORIF (UI2C_PROTSTS [9]), slave can switch to not address mode and own SLA will not be recognized. If setting this interrupt flag, slave will not receive any I^2C signal or address from master. At this time, I^2C controller should be reset by setting FUNMODE (UI2C_CTL [2:0]) = 000B to leave this status.

Protocol Functional Description

Programmable Setup and Hold Time

To guarantee a correct data setup and hold time, the timing must be configured. By programming HTCTL [5:0] (UI2C_TMCTL[11:6]) to configure hold time and STCTL [5:0] (UI2C_TMCTL[5:0]) to configure setup time.

The delay timing refer peripheral clock (PCLK). When device stretch master clock, the setup and hold time configuration value will not affected by stretched.

User should focus the limitation of setup and hold time configuration, the timing setting must follow I^2C protocol. Once setup time configuration greater than design limitation, that means if setup time setting make SCL output less than three PCLKs, I^2C controller can't work normally due to SCL must sample three times. And once hold time configuration greater than I^2C clock limitation, I^2C will occur bus error. Suggest that user calculate suitable timing with baud rate and protocol before setting timing. Table 6.14-1 Relationship between I2C Baud Rate and PCLKshows the relationship between I^2C baud rate and PCLK, the number of table represent one clock duty contain how many PCLKs. Setup and hold time configuration even can program some extreme values in our design, but user should follow I^2C protocol standard.

I^2C Baud Rate PCLK	100k	200k	400k
12MHz	120	60	30
24MHz	240	120	60
48MHz	480	240	120

Table 6.14-1 Relationship between I^2C Baud Rate and PCLK

For setup time wrong adjustment example, assume one SCL cycle contains ten PCLKs and set STCTL [5:0] (UI2C_TMCTL[5:0]) to 3 that stretches three PCLKs for setup time setting. The setup time setting limitation is ST_{limit} = (UI2C_BRGEN[25:16]+1) - 6. For example, if user decides PCLK = 12MHz and baud rate =100k, the UI2C_BRGEN[25:16] must be set to 59, and the STCTL [5:0] maximum value is 54.

Figure 6.14-17 Setup Time Wrong Adjustment

For hold time wrong adjustment example, use I^2C Baud Rate = 400k and PCLK = 48MHz, the SCL high/low duty = 60 PCLK. When HTCTL [5:0] (UI2C_TMCTL[11:6]) is set to 63 and STCTL [5:0] (UI2C_TMCTL[5:0]) is set to 0, the SDA output delay will over SCL high duty and cause bus error. The hold time setting limitation: HT_{limit} = (UI2C_BRGEN[25:16]+1) - 9. For example, if user

decides PCLK = 12MHz and baud rate =100k, the UI2C_BRGEN[25:16] must set 59, and the HTCTL [5:0] maximum value is 51.

Figure 6.14-18 Hold Time Wrong Adjustment

I²C Time-out Function

There is a time-out counter TOCNT (UI2C_PROTCTL [25:16]) which can be used to deal with the I²C bus hang-up. If the time-out counter is enabled, the counter starts up counting until it equals TOCNT (UI2C_PROTCTL [25:16]) and generates I²C interrupt to CPU or stops counting by clearing TOIEN (UI2C_PROTIEN [0]) to 0 or setting all I²C interrupt signal (ACKIF, ERRIF, ARBLOIF, NACKIF, STORIF, STARIF). User may write 1 to clear TOIF(UI2C_PROTSTS[5]) to 0. When time-out counter is enabled, writing 1 to the TOIF will reset counter and re-start up counting after TOIF is cleared. Refer to Figure 6.14-19 for the time-out counter TOCNT (UI2C_PROTCTL [25:16]). T_{TOCNT} = (TOCNT (UI2C_PROTCTL [25:16]) +1) x32 (5-bit) x T_{PCLK}. Note that the time counter clock source TMCNTSRC (USCI_ BRGEN [5]) must be set zero.

Figure 6.14-19 I²C Time-out Count Block Diagram

Wake-up Function

When chip enters Power-down mode and set WKEN (WKCTL[0]) to 1, other I²C master can wake-up our chip by addressing our I²C device, user must configure the related setting before entering sleep mode. The ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched the device's address and the ACK cycle done. The SCL is stretched until the bit is clear by user. If the frequency of SCL is low speed and the system has wake-up from address match frame, the user shall check this bit to

confirm this frame has transaction done and then to do the wake-up procedure. Therefore, when the chip is woken-up by address match with one of the device address register (UI2C_DEVADDR0), the user shall check the WKAKDONE (UI2C_PROTSTS [16]) bit is set to 1 to confirm the address wake-up frame has done. The WKAKDONE bit indicates that the ACK bit cycle of address match frame is done in power-down. The controller will stretch the SCL to low when the address is matched the device's slave address and the ACK cycle done. The SCL is stretched until the WKAKDONE bit is clear by user. If the frequency of SCL is low speed and the system has wake-up from address match frame, the user shall check this bit to confirm this frame has transaction done and then to do the wake-up procedure. Note that user must clear WKUPIF after clearing the WKAKDONE bit to 0.

The WR_STATUS (UI2C_PROTSTS [17]) bit records the Read/Write command on the address match wake-up frame. The user can use read this bit's status to prepare the next transmitted data (WR_STATUS = 0) or to wait the incoming data (WR_STATUS = 1) can be stored in time after the system is woken up by the address match frame.

When system is woken up by other I^2C master device, WKUPIF is set to indicate this event. User needs write "1" to clear this bit.

I^2C also support data toggle mode. When system is in power-down and the WKEN (UI2C_WKCTL [0]) set to 1 and WKADDREN (UI2C_WKCTL[1]) set to 0, the toggle of incoming data pin can wake up the system.

Example for Random Read on EEPROM

The following steps are used to configure the USCI0_I^2C related registers when using I^2C protocol to read data from EEPROM.

1. Set USCI0_I^2C the multi-function pin in the SYS_GPA_MFP, SYS_GPC_MFP, SYS_GPD_MFP registers as SCL and SDA pins.
2. Enable USCI0 APB clock, USCI0CKEN =1 in the "CLK_APBCLK[24]" register.
3. Set USCI0RST=1 to reset USCI controller then set USCI0RST=0 let USCI controller to normal operation, in the "SYS_IPRST1[24]" register.
4. Set FUNMODE =100 to enable USCI0_I^2C controller in the "UI2C_CTL" register.
5. Give USCI0_I^2C clock a divided register value for USCI0_I^2C clock rate in the "UI2C_BRGEN".
6. Set SETENA =0x00400000 in the "NVIC_ISR" register to set USCI_IRQ.
7. Set ACKIEN, ERRIEN, ARBLOIEN, NACKIEN, STORIEN, STARIEN, and TOIEN to enable I^2C Interrupt in the "UI2C_PROTIEN" register.
8. Set USCI address registers "USCI_DEVADDR0".

Random read operation is one of the methods of access EEPROM. The method allows the master to access any address of EEPROM space. Figure 6.14-20 shows the EEPROM random read operation.

Figure 6.14-20 EEPROM Random Read

Figure 6.14-21 shows how to use the I^{2}C controller to implement the protocol of EEPROM random read.

Figure 6.14-21 Protocol of EEPROM Random Read

The I^{2}C controller, which is a master, sends START to bus. Then, it sends a SLA+W (Slave address + Write bit) to EERPOM followed by two bytes data address to set the EEPROM address to read. Finally, a Repeat START followed by SLA+R is sent to read the data from EEPROM.

6.14.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
UI2C Base Address: **UI2Cx_BA = 0x4007_0000 + (0x10_0000 * x)** **x = 0, 1**				
UI2C_CTL	UI2Cx_BA+0x00	R/W	USCI Control Register	0x0000_0000
UI2C_BRGEN	UI2Cx_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00
UI2C_LINECTL	UI2Cx_BA+0x2C	R/W	USCI Line Control Register	0x0000_0000
UI2C_TXDAT	UI2Cx_BA+0x30	W	USCI Transmit Data Register	0x0000_0000
UI2C_RXDAT	UI2Cx_BA+0x34	R	USCI Receive Data Register	0x0000_0000
UI2C_DEVADDR0	UI2Cx_BA+0x44	R/W	USCI Device Address Register 0	0x0000_0000
UI2C_ADDRMSK0	UI2Cx_BA+0x4C	R/W	USCI Device Address Mask Register 0	0x0000_0000
UI2C_WKCTL	UI2Cx_BA+0x54	R/W	USCI Wake-up Control Register	0x0000_0000
UI2C_WKSTS	UI2Cx_BA+0x58	R/W	USCI Wake-up Status Register	0x0000_0000
UI2C_PROTCTL	UI2Cx_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000
UI2C_PROTIEN	UI2Cx_BA+0x60	R/W	USCI Protocol Interrupt Enable Register	0x0000_0000
UI2C_PROTSTS	UI2Cx_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000
UI2C_TMCTL	UI2Cx_BA+0x8C	R/W	I^2C Timing Configure Control Register	0x0000_0000

6.14.7 Register Description

USCI Control Register (UI2C_CTL)

Register	Offset	R/W	Description	Reset Value
UI2C_CTL	UI2Cx_BA+0x00	R/W	USCI 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					**FUNMODE**		

Bits	Description	
[31:3]	**Reserved**	Reserved.
[2:0]	**FUNMODE**	**Function Mode** This bit field selects the protocol for this USCI controller. Selecting a protocol that is not available or a reserved combination disables the USCI. When switching between two protocols, the USCI has to be disabled before selecting a new protocol. Simultaneously, the USCI will be reset when user write 000 to FUNMODE. 000 = The USCI is disabled. All protocol related state machines are set to idle state. 001 = The SPI protocol is selected. 010 = The UART protocol is selected. 100 = The I^2C protocol is selected. **Note:** Other bit combinations are reserved.

USCI Baud Rate Generator Register (UI2C_BRGEN)

Register	Offset	R/W	Description	Reset Value
UI2C_BRGEN	UI2Cx_BA+0x08	R/W	USCI Baud Rate Generator Register	0x0000_3C00

31	30	29	28	27	26	25	24
Reserved						**CLKDIV**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
CLKDIV							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved	**DSCNT**					**PDSCNT**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**TMCNTSRC**	**TMCNTEN**	**SPCLKSEL**		**PTCLKSEL**	**RCLKSEL**

Bits	Description	
[31:26]	**Reserved**	Reserved.
[25:16]	**CLKDIV**	**Clock Divider** This bit field defines the ratio between the protocol clock frequency f_{PROT_CLK} and the clock divider frequency f_{DIV_CLK} (fDIV_CLK = fPROT_CLK / (CLKDIV+1)). **Note:** In UART function, it can be updated by hardware in the 4th falling edge of the input data 0x55 when the auto baud rate function (ABREN(UI2C_PROTCTL[6])) is enabled. The revised value is the average bit time between bit 5 and bit 6. The user can use revised CLKDIV and new BRDETITV (UI2C_PROTCTL[24:16]) to calculate the precise baud rate.
[15]	**Reserved**	Reserved.
[14:10]	**DSCNT**	**Denominator for Sample Counter** This bit field defines the divide ratio of the sample clock f_{SAMP_CLK}. The divided frequency $f_{DS_CNT} = f_{PDS_CNT}$ / (DSCNT+1). **Note:** The maximum value of DSCNT is 0xF on UART mode and suggest to set over 4 to confirm the receiver data is sampled in right value.
[9:8]	**PDSCNT**	**Pre-divider for Sample Counter** This bit field defines the divide ratio of the clock division from sample clock f_{SAMP_CLK}. The divided frequency $f_{PDS_CNT} = f_{SAMP_CLK}$ / (PDSCNT+1).
[7:6]	**Reserved**	Reserved.
[5]	**TMCNTSRC**	**Time Measurement Counter Clock Source Selection** 0 = Time measurement counter with f_{PROT_CLK}. 1 = Time measurement counter with f_{DIV_CLK}.
[4]	**TMCNTEN**	**Time Measurement Counter Enable Bit** This bit enables the 10-bit timing measurement counter. 0 = Time measurement counter Disabled. 1 = Time measurement counter Enabled.
[3:2]	**SPCLKSEL**	**Sample Clock Source Selection**

		This bit field used for the clock source selection of a sample clock (f_{SAMP_CLK}) for the protocol processor. 00 = f_{SAMP_CLK} = f_{DIV_CLK}. 01 = f_{SAMP_CLK} = f_{PROT_CLK}. 10 = f_{SAMP_CLK} = f_{SCLK}. 11 = f_{SAMP_CLK} = f_{REF_CLK}.
[1]	**PTCLKSEL**	**Protocol Clock Source Selection** This bit selects the source signal of protocol clock (f_{PROT_CLK}). 0 = Reference clock f_{REF_CLK}. 1 = f_{REF_CLK2} (its frequency is half of f_{REF_CLK}).
[0]	**RCLKSEL**	**Reference Clock Source Selection** This bit selects the source signal of reference clock (f_{REF_CLK}). 0 = Peripheral device clock f_{PCLK}. 1 = HXT/LXT.

USCI Line Control Register (UI2C_LINECTL)

Register	Offset	R/W	Description	Reset Value
UI2C_LINECTL	UI2Cx_BA+0x2C	R/W	USCI 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				**DWIDTH**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved							**LSB**

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**DWIDTH**	**Word Length of Transmission** This bit field defines the data word length (amount of bits) for reception and transmission. The data word is always right-aligned in the data buffer. USCI support word length from 4 to 16 bits. 0x0: The data word contains 16 bits located at bit positions [15:0]. 0x1: Reserved. 0x2: Reserved. 0x3: Reserved. 0x4: The data word contains 4 bits located at bit positions [3:0]. 0x5: The data word contains 5 bits located at bit positions [4:0]. ... 0xF: The data word contains 15 bits located at bit positions [14:0]. **Note:** In I^2C protocol, the length must be configured as 8 bits.
[7:1]	**Reserved**	Reserved.
[4:1]	**Reserved**	Reserved.
[0]	**LSB**	**LSB First Transmission Selection** 0 = The MSB, which bit of transmit/receive data buffer depends on the setting of DWIDTH, is transmitted/received first. 1 = The LSB, the bit 0 of data buffer, will be transmitted/received first.

USCI Transmit Data Register (UI2C_TXDAT)

Register	Offset	R/W	Description	Reset Value
UI2C_TXDAT	UI2Cx_BA+0x30	W	USCI Transmit 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**
TXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
TXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**TXDAT**	**Transmit Data** Software can use this bit field to write 16-bit transmit data for transmission.

USCI Receive Data Register (UI2C_RXDAT)

Register	Offset	R/W	Description	Reset Value
UI2C_RXDAT	UI2Cx_BA+0x34	R	USCI Receive 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**
RXDAT							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RXDAT							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**RXDAT**	**Received Data** This bit field monitors the received data which stored in receive data buffer. **Note 1:** In I^2C protocol, only use RXDAT[7:0].

USCI Device Address Register (UI2C_DEVADDR)

Register	Offset	R/W	Description	Reset Value
UI2C_DEVADDR0	UI2Cx_BA+0x44	R/W	USCI Device Address Register 0	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						**DEVADDR**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DEVADDR							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**DEVADDR**	**Device Address** In I^2C protocol, this bit field contains the programmed slave address. If the first received address byte is b1111 0AAX, the AA bits are compared to the bits DEVADDR[9:8] to check for address match, where the X is R/W bit. Then the second address byte is also compared to DEVADDR[7:0]. **Note:** When I^2C operating in 7-bit address mode, only use DEVADDR[6:0]

USCI Device Address Mask Register (UI2C_ADDRMSK) – for I^{2}C Only

Register	Offset	R/W	Description	Reset Value
UI2C_ADDRMSK0	UI2Cx_BA+0x4C	R/W	USCI Device Address Mask Register 0	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						**ADDRMSK**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADDRMSK							

Bits	Description	
[31:10]	**Reserved**	Reserved.
[9:0]	**ADDRMSK**	**USCI Device Address Mask** 0 = Mask Disabled (the received corresponding register bit should be exact the same as address register.). 1 = Mask Enabled (the received corresponding address bit is don't care.). USCI support multiple address recognition with two address mask register. 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.

USCI Wake-up Control Register (UI2C_WKCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_WKCTL	UI2Cx_BA+0x54	R/W	USCI 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						**WKADDREN**	**WKEN**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**WKADDREN**	**Wake-up Address Match Enable Bit** 0 = The chip is woken up according to data toggle. 1 = The chip is woken up according to address match.
[0]	**WKEN**	**Wake-up Enable Bit** 0 = Wake-up function Disabled. 1 = Wake-up function Enabled.

USCI Wake-up Status Register (UI2C_WKSTS)

Register	Offset	R/W	Description	Reset Value
UI2C_WKSTS	UI2Cx_BA+0x58	R/W	USCI 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							**WKF**

Bits	Description	
[31:1]	**Reserved**	Reserved.
[0]	**WKF**	**Wake-up Flag** When chip is woken up from Power-down mode, this bit is set to 1. Software can write 1 to clear this bit.

USCI Protocol Control Register – I²C (UI2C_PROTCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTCTL	UI2Cx_BA+0x5C	R/W	USCI Protocol Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PROTEN	**Reserved**					**TOCNT**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
TOCNT							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		**PTRG**	**ADDR10EN**	**STA**	**STO**	**AA**	**GCFUNC**

Bits	Description	
[31]	**PROTEN**	**I²C Protocol Enable Bit** 0 = I²C Protocol disable. 1 = I²C Protocol enable.
[30:26]	**Reserved**	Reserved.
[25:16]	**TOCNT**	**Time-out Clock Cycle** This bit field indicates how many clock cycle selected by TMCNTSRC (UI2C_BRGEN [5]) when each interrupt flags are clear. The time-out is enable when TOCNT bigger than 0. **Note:** The TMCNTSRC (UI2C_BRGEN [5]) must be set zero on I²C mode.
[15:6]	**Reserved**	Reserved.
[5]	**PTRG**	**I²C Protocol Trigger** When a new state is present in the UI2C_PROTSTS register, if the related interrupt enable bits are set, the I²C interrupt is requested. It must write one by software to this bit after the related interrupt flags are set to 1 and the I²C protocol function will go ahead until the STOP is active or the PROTEN is disabled. 0 = I²C's stretch disabled and the I²C protocol function will go ahead. 1 = I²C's stretch active.
[4]	**ADDR10EN**	**Address 10-bit Function Enable Bit** 0 = Address match 10 bit function Disabled. 1 = Address match 10 bit function Enabled.
[3]	**STA**	**I²C START Control** Setting STA to logic 1 to enter Master mode, and the I²C hardware sends a START or repeat START condition to bus when the bus is free.
[2]	**STO**	**I²C STOP Control** In Master mode, setting STO to transmit a STOP condition to bus then I²C hardware will check the bus condition if a STOP condition is detected this bit will be cleared by hardware automatically. In a slave mode, setting STO resets I²C hardware to the defined "not addressed" slave mode when bus error (UI2C_PROTSTS.ERRIF = 1).

[1]	**AA**	**Assert Acknowledge Control** 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.
[0]	**GCFUNC**	**General Call Function** 0 = General Call Function Disabled. 1 = General Call Function Enabled.

USCI Protocol Interrupt Enable Register – I^{2}C (UI2C_PROTIEN)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTIEN	UI2Cx_BA+0x60	R/W	USCI Protocol 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							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**ACKIEN**	**ERRIEN**	**ARBLOIEN**	**NACKIEN**	**STORIEN**	**STARIEN**	**TOIEN**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6]	**ACKIEN**	**Acknowledge Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an acknowledge is detected by a master. 0 = The acknowledge interrupt Disabled. 1 = The acknowledge interrupt Enabled.
[5]	**ERRIEN**	**Error Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an I^{2}C error condition is detected (indicated by ERR (UI2C_PROTSTS [16])). 0 = The error interrupt Disabled. 1 = The error interrupt Enabled.
[4]	**ARBLOIEN**	**Arbitration Lost Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if an arbitration lost event is detected. 0 = The arbitration lost interrupt Disabled. 1 = The arbitration lost interrupt Enabled.
[3]	**NACKIEN**	**Non - Acknowledge Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a non - acknowledge is detected by a master. 0 = The non - acknowledge interrupt Disabled. 1 = The non - acknowledge interrupt Enabled.
[2]	**STORIEN**	**Stop Condition Received Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a stop condition is detected. 0 = The stop condition interrupt Disabled. 1 = The stop condition interrupt Enabled.

[1]	**STARIEN**	**Start Condition Received Interrupt Enable Bit** This bit enables the generation of a protocol interrupt if a start condition is detected. 0 = The start condition interrupt Disabled. 1 = The start condition interrupt Enabled.
[0]	**TOIEN**	**Time-out Interrupt Enable Bit** In I^2C protocol, this bit enables the interrupt generation in case of a time-out event. 0 = The time-out interrupt Disabled. 1 = The time-out interrupt Enabled.

USCI Protocol Status Register – I^{2}C (UI2C_PROTSTS)

Register	Offset	R/W	Description	Reset Value
UI2C_PROTSTS	UI2Cx_BA+0x64	R/W	USCI Protocol Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved					**BUSHANG**	**WRSTSWK**	**WKAKDONE**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
SLAREAD	**SLASEL**	**ACKIF**	**ERRIF**	**ARBLOIF**	**NACKIF**	**STORIF**	**STARIF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**ONBUSY**	**TOIF**	**Reserved**				

Bits	Description	
[31:19]	**Reserved**	Reserved.
[18]	**BUSHANG**	**Bus Hang-up** This bit indicates bus hang-up status. There is 4-bit counter count when SCL hold high and refer f_{SAMP_CLK}. The hang-up counter will count to overflow and set this bit when SDA is low. The counter will be reset by falling edge of SCL signal. 0 = The bus is normal status for transmission. 1 = The bus is hang-up status for transmission. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware.
[17]	**WRSTSWK**	**Read/Write Status Bit in Address Wake-up Frame** 0 = Write command be record on the address match wake-up frame. 1 = Read command be record on the address match wake-up frame.
[16]	**WKAKDONE**	**Wake-up Address Frame Acknowledge Bit Done** 0 = The ACK bit cycle of address match frame isn't done. 1 = The ACK bit cycle of address match frame is done in power-down. **Note:** This bit can't release when WKUPIF is set.
[15]	**SLAREAD**	**Slave Read Request Status** This bit indicates that a slave read request has been detected. 0 = A slave read request has not been detected. 1 = A slave read request has been detected. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware.
[14]	**SLASEL**	**Slave Select Status** This bit indicates that this device has been selected as slave. 0 = The device is not selected as slave. 1 = The device is selected as slave. **Note:** This bit has no interrupt signal, and it will be cleared automatically by hardware.

[13]	**ACKIF**	**Acknowledge Received Interrupt Flag** This bit indicates that an acknowledge has been received in master mode. This bit is not set in slave mode. A protocol interrupt can be generated if UI2C_PROTCTL.ACKIEN = 1. 0 = An acknowledge has not been received. 1 = An acknowledge has been received. It is cleared by software writing one into this bit
[12]	**ERRIF**	**Error Interrupt Flag** This bit indicates that 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. A protocol interrupt can be generated if UI2C_PROTCTL.ERRIEN = 1. 0 = An I^2C error has not been detected. 1 = An I^2C error has been detected. It is cleared by software writing one into this bit **Note:** This bit is set when slave mode, user must write one into STO register to the defined “not addressed” slave mode.
[11]	**ARBLOIF**	**Arbitration Lost Interrupt Flag** This bit indicates that an arbitration has been lost. A protocol interrupt can be generated if UI2C_PROTCTL.ARBLOIEN = 1. 0 = An arbitration has not been lost. 1 = An arbitration has been lost. It is cleared by software writing one into this bit
[10]	**NACKIF**	**Non - Acknowledge Received Interrupt Flag** This bit indicates that a non - acknowledge has been received in master mode. This bit is not set in slave mode. A protocol interrupt can be generated if UI2C_PROTCTL.NACKIEN = 1. 0 = A non - acknowledge has not been received. 1 = A non - acknowledge has been received. It is cleared by software writing one into this bit
[9]	**STORIF**	**Stop Condition Received Interrupt Flag** This bit indicates that a stop condition has been detected on the I^2C bus lines. A protocol interrupt can be generated if UI2C_PROTCTL.STORIEN = 1. 0 = A stop condition has not yet been detected. 1 = A stop condition has been detected. It is cleared by software writing one into this bit
[8]	**STARIF**	**Start Condition Received Interrupt Flag** This bit indicates that a start condition or repeated start condition has been detected on master mode. However, this bit also indicates that a repeated start condition has been detected on slave mode. A protocol interrupt can be generated if UI2C_PROTCTL.STARIEN = 1. 0 = A start condition has not yet been detected. 1 = A start condition has been detected. It is cleared by software writing one into this bit
[7]	**Reserved**	Reserved.
[6]	**ONBUSY**	**On Bus Busy** Indicates that a communication is in progress on the bus. It is set by hardware when a START condition is detected. It is cleared by hardware when a STOP condition is detected 0 = The bus is IDLE (both SCLK and SDA High).

		1 = The bus is busy.
[5]	**TOIF**	**Time-out Interrupt Flag** 0 = A time-out interrupt status has not occurred. 1 = A time-out interrupt status has occurred. **Note:** It is cleared by software writing one into this bit
[4:3]	**Reserved**	Reserved.
[2:0]	**Reserved**	Reserved.
[0]	**Reserved**	Reserved.

USCI Timing Configure Control Register (UI2C_TMCTL)

Register	Offset	R/W	Description	Reset Value
UI2C_TMCTL	UI2Cx_BA+0x8C	R/W	I^2C Timing Configure 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				**HTCTL**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
HTCTL		**STCTL**					

Bits	Description	
[31:8]	**Reserved**	Reserved.
[11:6]	**HTCTL**	**Hold Time Configure Control Register** This field is used to generate the delay timing between SCL falling edge SDA edge in transmission mode. The delay hold time is numbers of peripheral clock = HTCTL x f_{PCLK}.
[5:0]	**STCTL**	**Setup Time Configure Control Register** This field is used to generate a delay timing between SDA edge and SCL rising edge in transmission mode. The delay setup time is numbers of peripheral clock = STCTL x f_{PCLK}.

6.15 Hardware Divider (HDIV)

6.15.1 Overview

The hardware divider (HDIV) is useful to the high performance application. The hardware divider is a signed, integer divider with both quotient and remainder outputs.

6.15.2 Features

- Signed (two's complement) integer calculation
- 32-bit dividend with 16-bit divisor calculation capacity
- 32-bit quotient and 32-bit remainder outputs (16-bit remainder with sign extends to 32-bit)
- Divided by zero warning flag
- 6 HCLK clocks taken for one cycle calculation
- Write divisor to trigger calculation
- Waiting for calculation ready automatically when reading quotient and remainder

6.15.3 Basic Configuration

Before using the hardware divider, the clock of hardware divider must be enabled. To enable hardware divider, it needs to set HDIVCKEN (CLK_AHBCLK[4]) to 1.

6.15.4 Functional Description

To use hardware divider, it needs to set dividend first. Then set divisor and the hardware divider will trigger calculation automatically after divisor written. The calculation results including the quotient and remainder could be got by reading QUOTIENT (HDIV_QUOTIENT[31:0]) and REM (HDIV_REM[31:0]) register. If CPU reads QUOTIENT or REM before hardware divider calculation finishing, CPU will be held until hardware divider finishing the calculation. Therefore, CPU can always get valid results after trigger one hardware divider calculation without software delay.

DIVBYZERO(HDIV_STATUS[1]) will be set if divisor is 0.

The dividend is 32-bit signed integer and divisor is 16-bit signed integer. The quotient is 32-bit signed integer and the remainder is 16-bit signed integer. It is noted that the case of dividing the minimum dividend by -1, the quotient is set to be the minimum negative value since overflow and the remainder is set to 0. This is the only case the quotient is not represented in a positive number when a negative number by a negative number.

Figure 6.15-1 shows the operation flow of hardware divider. To calculate X / Y, CPU needs to write X to DIVIDEND (HDIV_DIVIDEND[31:0]) register, and then write Y to DIVISOR (HDIV_DIVISOR[15:0]). CPU can read QUOTIENT and REM registers to get calculation results after DIVISOR has been written.

Figure 6.15-1 Hardware Divider Operation Flow

6.15.5 Register Map

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

Register	Offset	R/W	Description	Reset Value
HDIV Base Address: **HDIV_BA = 0x5001_4000**				
HDIV_DIVIDEND	HDIV_BA+0x00	R/W	Dividend Source Register	0x0000_0000
HDIV_DIVISOR	HDIV_BA+0x04	R/W	Divisor Source Resister	0x0000_FFFF
HDIV_QUOTIENT	HDIV_BA+0x08	R/W	Quotient Result Resister	0x0000_0000
HDIV_REM	HDIV_BA+0x0C	R/W	Remainder Result Register	0x0000_0000
HDIV_STATUS	HDIV_BA+0x10	R	Divider Status Register	0x0000_0001

6.15.6 Register Description

Dividend Source Register (HDIV_DIVIDEND)

Register	Offset	R/W	Description	Reset Value
HDIV_DIVIDEND	HDIV_BA+0x00	R/W	Dividend Source Register	0x0000_0000

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

Bits	Description	
[31:0]	**DIVIDEND**	**Dividend Source** This register is given the dividend of divider before calculation is started.

Divisor Source Register (HDIV_DIVISOR)

Register	Offset	R/W	Description	Reset Value
HDIV_DIVISOR	HDIV_BA+0x04	R/W	Divisor Source Resister	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**
DIVISOR							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DIVISOR							

Bits	Description	
[31:16]	**Reserved**	Reserved.
[15:0]	**DIVISOR**	**Divisor Source** This register is given the divisor of divider before calculation starts. **Note:** When this register is written, hardware divider will start calculation.

Quotient Result Register (HDIV_QUOTIENT)

Register	Offset	R/W	Description	Reset Value
HDIV_QUOTIENT	HDIV_BA+0x08	R/W	Quotient Result Resister	0x0000_0000

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

Bits	Description	
[31:0]	**QUOTIENT**	**Quotient Result** This register holds the quotient result of divider after calculation is completed.

Remainder Result Register (HDIV_REM)

Register	Offset	R/W	Description	Reset Value
HDIV_REM	HDIV_BA+0x0C	R/W	Remainder Result Register	0x0000_0000

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

Bits	Description	
[31:0]	**REM**	**Remainder Result** The remainder of hardware divider is 16-bit sign integer (REM[15:0]) with sign extension (REM[31:16]) to 32-bit integer.

Divider Status Register (HDIV_STATUS)

Register	Offset	R/W	Description	Reset Value
HDIV_STATUS	HDIV_BA+0x10	R	Divider Status 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**
Reserved						**DIVBYZERO**	**Reserved**

Bits	Description	
[31:2]	**Reserved**	Reserved.
[1]	**DIVBYZERO**	**Divisor Zero Warning (Read Only)** 0 = The divisor is not 0. 1 = The divisor is 0. **Note:** The DIVBYZERO flag is used to indicate divide-by-zero situation and updated whenever HDIV_DIVISOR is written. This bit is read only.
[0]	**Reserved**	Reserved.

6.16 Analog to Digital Converter (ADC)

1.1.1 Overview

The NDA102 series contains one 12-bit successive approximation analog-to-digital converter (SAR A/D converter) with 8 single-end external input channels. The A/D converters can be started by software, external pin (STADC/PC.1) or PWM trigger.

1.1.2 Features

- Analog input voltage range: 0~V_{DD}.
- 12-bit resolution and 10-bit accuracy guaranteed.
- Up to 8 single-end analog input channels.
- ADC clock frequency up to 16MHz.
- Configurable ADC internal sampling time.

1.1.3 Block Diagram

Figure 6.16-1 ADC Control Block Diagram

6.16.1 Basic Configuration

The NDA102 series has two sample and hold (S/H) to sampling two input ADC channel simultaneously. And support four type operating mode for BLDC motor used.

The ADC pin functions are configured in SYS_PB_MFP, SYS_PC_MFP and SYS_PD_MFP register. It is recommended to disable the digital input path of the analog input pins to avoid the leakage current. User can disable the digital input path by configuring PB_DINOFF, PC_DINOFF and PD_DINOFF register.

The ADC peripheral clock can be enabled in ADCCKEN (CLK_APBCLK[28]). The ADC peripheral clock source is selected by ADCSEL (CLK_CLKSEL1[5:4]). The clock pre-scalar is determined by ADCDIV (CLK_CLKDIV[23:16]).

6.16.2 Functional Description

The A/D converter operates by successive approximation with 12-bit resolution. When changing the analog input channel is enabled, in order to prevent incorrect operation, software must clear ADCnSWTRG bit to 0 in the ADC_CTL register. The A/D converter discards the current conversion immediately and enters idle state while ADCnSWTRG bit is cleared.

6.16.2.1 ADC Peripheral Clock Generator

The ADC engine has four clock sources selected by ADCSEL (CLK_CLKSEL1), and selected between HXT and LXT by CLK_PWRCTL. The ADC clock peripheral frequency is divided by an 8-bit pre-scalar with the following formula:

ADC peripheral clock frequency = (ADC peripheral clock source frequency) / (ADCDIV+1); where the 8-bit ADCDIV is located in register CLK_CLKDIV.

In general, software can set ADCSEL and ADCDIV to get 16 MHz or slightly less.

Figure 6.16-2 ADC Peripheral Clock Control

6.16.2.2 ADC Operation

A/D conversion is performed only once on the specified single channel. The operation is as follows:

1. A/D conversion will be started when the ADCnSWTRG bit of ADC_CTL is set to 1 by software or hardware trigger input.
2. When A/D conversion is finished, the result is stored in the A/D data register.
3. The ADCnIF bit of ADC_STATUS register will be set to 1. If the ADCnIEN bit of ADC_CTL register is set to 1, the ADC interrupt will be asserted.
4. The ADCnSWTRG bit remains 1 during A/D conversion. When A/D conversion ends, the ADCnSWTRG bit is automatically cleared to 0 and the A/D converter enters idle state.

An example timing diagram for Single mode is shown in Figure 6.16-3.

Figure 6.16-3 Single Mode Conversion Timing Diagram

6.16.2.3 Hardware Trigger Input Sampling and A/D Conversion Time

A/D conversion can be triggered by hardware trigger request. When the ADCnHWTRGEN bit of ADC_CTL register is set to 1 to enable ADC hardware trigger function, setting the ADCnTRGSOR bits to 0000b is to select external trigger input from the STADC pin. Software can set ADCnSTADCSEL to select trigger condition between falling or rising edge. If edge trigger condition is selected, the high and low state must be kept at least 4 PLCKs. Pulse that is shorter than this specification will be ignored.

6.16.2.4 PWM trigger

A/D conversion can also be triggered by PWM request. When the ADCnHWTRGEN is set to high to enable ADC hardware trigger function, setting the ADCnTRGSOR can be select hardware trigger input source from PWM trigger. When PWM trigger is enabled, setting ADC_TRGDLY can insert a delay time between PWM trigger condition and ADC start conversion.

Figure 6.16-4 ADC Hardware Trigger Source

6.16.2.5 Conversion Result Monitor by Window Compare Mode Function

The ADC controller in the NDA102 series provides a window comparator function, Software can write ADC_WCMPDAT register to set low and high bound range and to monitor three step ADC value.

When the compare result meets the setting, compare match counter will increase 1, otherwise, the compare match counter will be clear to 0. When the match counter reaches the setting of (WCMPMCNT+1) then WCMPIF bit will be set to 1, if WCMPIEN and WCMPEN bit is set then an ADCINT interrupt request is generated. Software can use it to monitor the external analog input pin voltage transition.

6.16.2.6 Interrupt Sources

There are three interrupt sources of ADC interrupt. ADC0 interrupt, ADC1 interrupt and ADC windows comparator interrupt.

6.16.2.7 Independent Simple Mode

The NDA102 has two Sample & Hold (S/H) and one A/D conversion block. It can set to independent sampling, conversion and independent general interrupt when ADCMODE is set as 00b (ADC_CTL[7:6]). It can be seen as two ADC function in used.

Figure 6.16-5 Independent Sample Mode Conversion Timing Diagram

6.16.2.8 Independent 2SH Mode

The NDA102 can also be set to independent, but it needs to convert S/H twice continuously (S/H0 and S/H1) and only generates ADC0IF interrupt when ADCMODE is set as 01b (ADC_CTL[7:6]).

For example, ADC0 trigger source is set as PWM0 and ADC1 trigger source is set as PWM2. If PWM0 triggers ADC0 first, when ADC0 conversion is finished, it does not generate interrupts, and needs to wait for ADC1 conversion. When ADC1 conversion is finished, then into ADC0 interrupt handler. The mean is continuous convert ADC0 and ADC1 then generate interrupt.

6.16.2.9 Simultaneous Simple Mode

In this mode, the ADC in the NDA102 can be set to perform two S/H conversions triggered once by the ADC0 trigger source, and generate ADC0IF interrupt when ADCMODE is set as 10b (ADC_CTL[7:6]).

Figure 6.16-6 shows one time and simultaneous sample-hold in S/H0 and S/H1, and then sequential conversion by the A/D converter.

Figure 6.16-6 Simultaneous Simple Mode Conversion Timing Diagram

6.16.2.10 Simultaneous Sequential 4R Mode

In this mode, the ADC in the NDA102 can be set to perform two S/H conversions triggered by the ADC0 trigger source, then perform continuous conversion four times by ADC, and generate ADC0IF interrupt when ADCMODE is set as 11b (ADC_CTL[7:6]).

Figure 6.16-7 shows simultaneous sample-hold in S/H0 and S/H1, converting ADC0 first, and then sequentially converting ADC1, ADC0, and ADC1 four times. The ADC0 S/H will be sampled again when ADC0 conversion is finished. Also, the ADC1 S/H will be sampled again when ADC1 conversion is finished.

Figure 6.16-7 Simultaneous Sequential 4R Mode Conversion Timing Diagram

6.16.3 Register Map

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

Register	Offset	R/W	Description	Reset Value
ADC Base Address: **ADC_BA = 0x400E_0000**				
ADC_DAT0	ADC_BA+0x00	R	ADC data register 0	0x0000_0000
ADC_DAT1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000
ADC_CTL	ADC_BA+0x20	R/W	ADC Control Register	0x0000_0000
ADC_TRGSOR	ADC_BA+0x24	R/W	ADC Hardware Trigger Source Control Register	0x0000_0000
ADC_TRGDLY	ADC_BA+0x28	R/W	ADC Trigger Delay Control Register	0x0000_0000
ADC_SMPCNT	ADC_BA+0x2C	R/W	ADC Sampling Time Counter Register	0x0000_0005
ADC_STATUS	ADC_BA+0x30	R/W	ADC Status Register	0x0000_0000
ADC_WCMPCTL	ADC_BA+0x34	R/W	ADC Window Comparator Control Register	0x0000_0000
ADC_WCMPDAT	ADC_BA+0x38	R/W	ADC Window Comparator Data Register	0x0000_0000

6.16.4 Register Description

ADC Data Register 0 (ADC_DAT0)

Register	Offset	R/W	Description	Reset Value
ADC_DAT0	ADC_BA+0x00	R	ADC data register 0	0x0000_0000

31	30	29	28	27	26	25	24
ADC1VALID	**ADC1OV**	**Reserved**		**ADC1DAT0**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ADC1DAT0							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ADC0VALID	**ADC0OV**	**Reserved**		**ADC0DAT0**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADC0DAT0							

Bits	Description	
[31]	**ADC1VALID**	**ADC1 Valid Flag** 0 = Data in ADC1DAT0[27:16] bits not valid. 1 = Data in ADC1DAT0[27:16] bits valid. **Note:** This bit is set to "1" when A/D conversion is completed and cleared by hardware after the ADC_DAT0 register is read.
[30]	**ADC1OV**	**ADC1 over Run Flag** 0 = Data in ADC1DAT0[27:16] is recent conversion result. 1 = Data in ADC1DAT0[27:16] overwritten. **Note1:** If converted data in ADC1DAT0[27:16] has not been read before, the new conversion result is loaded to this register, OV is set to "1". **Note2:**It is cleared by hardware after the ADC_DAT0 register is read.
[29:28]	**Reserved**	Reserved.
[27:16]	**ADC1DAT0**	**ADC1 Conversion Result** This field contains conversion result of ADC.
[15]	**ADC0VALID**	**ADC0 Valid Flag** 0 = Data in ADC0DAT0[11:0] bits not valid. 1 = Data in ADC0DAT0[11:0] bits valid. **Note:** This bit is set to "1" when A/D conversion is completed and cleared by hardware after the ADC_DAT0 register is read.

Bits	Description	
[14]	**ADC0OV**	**ADC0 over Run Flag** 0 = Data in ADC0DAT0[11:0] is recent conversion result. 1 = Data in ADC0DAT0[11:0] overwritten. **Note1:** If converted data in ADC0DAT0[11:0] has not been read before the new conversion result is loaded to this register, OV is set to "1". **Note2:** It is cleared by hardware after the ADC_DAT0 register is read.
[13:12]	**Reserved**	Reserved.
[11:0]	**ADC0DAT0**	**ADC0 Conversion Result** This field contains conversion result of ADC.

ADC Data Register 1 (ADC_DAT1)

Register	Offset	R/W	Description	Reset Value
ADC_DAT1	ADC_BA+0x04	R	ADC Data Register 1	0x0000_0000

31	30	29	28	27	26	25	24
ADC1VALID	**ADC1OV**	**Reserved**		**ADC1DAT1**			
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ADC1DAT1							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ADC0VALID	**ADC0OV**	**Reserved**		**ADC0DAT1**			
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADC0DAT1							

Bits	Description	
[31]	**ADC1VALID**	**ADC1 Valid Flag** 0 = Data in ADC1DAT1[27:16] bits not valid. 1 = Data in ADC1DAT1[27:16] bits valid. **Note:** This bit is set to "1" when A/D conversion is completed and cleared by hardware after the ADC_DAT1 register is read.
[30]	**ADC1OV**	**ADC1 over Run Flag** 0 = Data in ADC1DAT1[27:16] is recent conversion result. 1 = Data in ADC1DAT1[27:16] overwritten. **Note1:** If converted data in ADC1DAT1[27:16] has not been read before the new conversion result is loaded to this register, OV is set to "1". **Note2:** It is cleared by hardware after the ADC_DAT1 register is read.
[29:28]	**Reserved**	Reserved.
[27:16]	**ADC1DAT1**	**ADC1 Conversion Result for FIFO1** This field contains conversion result of ADC.
[15]	**ADC0VALID**	**ADC0 Valid Flag** 0 = Data in ADC0DAT1[11:0] bits not valid. 1 = Data in ADC0DAT1[11:0] bits valid. **Note:** This bit is set to "1" when A/D conversion is completed and cleared by hardware after the ADC_DAT1 register is read.
[14]	**ADC0OV**	**ADC0Over Run Flag** 0 = Data in ADC0DAT1[11:0] is recent conversion result. 1 = Data in ADC0DAT1[11:0]] overwritten. **Note1:** If converted data in ADC0DAT1[11:0] has not been read before the new conversion result is loaded to this register, OV is set to "1". **Note2:** It is cleared by hardware after the ADC_DAT1 register is read.
[13:12]	**Reserved**	Reserved.

Bits	Description	
[11:0]	**ADC0DAT1**	**ADC0 Conversion Result for FIFO1** This field contains conversion result of ADC.

ADC Control Register (ADC_CTL)

Register	Offset	R/W	Description	Reset Value
ADC_CTL	ADC_BA+0x20	R/W	ADC Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved	**ADC1SEQSEL**			**Reserved**	**ADC1CHSEL**		
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved	**ADC0SEQSEL**			**Reserved**	**ADC0CHSEL**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved				**ADC1SWTRG**	**ADC1HWTRGEN**	**ADC1IEN**	**Reserved**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADCMODE		**ADCSS3R**	**Reserved**	**ADC0SWTRG**	**ADC0HWTRGEN**	**ADC0IEN**	**ADCEN**

Bits	Description	
[31]	**Reserved**	Reserved.
[30:28]	**ADC1SEQSEL**	**ADC1 Sequential Input Pin Selection (Second Input)** 000 = ADC1_CH0. 001 = ADC1_CH1. 010 = ADC1_CH2. 011 = ADC0_CH0. 100 = ADC0_CH4. 101 = PGA_ADC. 110 = Temp Sensor. 111 = VSS.
[27]	**Reserved**	Reserved.
[26:24]	**ADC1CHSEL**	**ADC1 Channel Select** 000 = ADC1_CH0. 001 = ADC1_CH1. 010 = ADC1_CH2. 011 = ADC0_CH0. 100 = ADC0_CH4. 101 = PGA_ADC. 110 = Temp Sensor. 111 = VSS.
[23]	**Reserved**	Reserved.

Bits	Description	
[22:20]	**ADC0SEQSEL**	**ADC0 Sequential Input Pin Selection** 000 = ADC0_CH0. 001 = ADC0_CH1. 010 = ADC0_CH2. 011 = ADC0_CH3. 100 = ADC0_CH4. 101 = PGA_ADC. 110 = BAND_GAP. 111 = VSS.
[19]	**Reserved**	Reserved.
[18:16]	**ADC0CHSEL**	**ADC1 Channel Select** 000 = ADC0_CH0. 001 = ADC0_CH1. 010 = ADC0_CH2. 011 = ADC0_CH3. 100 = ADC0_CH4. 101 = PGA_ADC. 110 = BAND_GAP. 111 = VSS.
[15:12]	**Reserved**	Reserved.
[11]	**ADC1SWTRG**	**ADC1 Conversion Start** 0 = Conversion stopped and A/D converter entered idle state. 1 = Conversion start. **Note:** ADC1SWTRG can be set to "1" from two sources: software and external pin STADC. This bit will be cleared to "0" by hardware automatically.
[10]	**ADC1HWTRGEN**	**Hardware Trigger ADC Convertion Enable Bit** Enable or disable triggering of A/D conversion by Hardware (PWM, Timer, ADC self) 0= Hardware Trigger ADC Convertion Disabled. 1= Hardware Trigger ADC Convertion Enabled.
[9]	**ADC1IEN**	**ADC1 Interrupt Enable Bit** 0 = ADC1 interrupt function Disabled. 1 = ADC1 interrupt function Enabled. **Note:** A/D conversion end interrupt request is generated if ADC1IEN bit is set to "1".
[7:6]	**ADCMODE**	**A/D Conversion Mode** 00 = Independent simple; independent function and independent interrupt by themselves. 01 = Independent 2SH; independent trigger function, ADC0 with ADC1 both convert finish then only generate interrupt ADC0IF. 10 = Simultaneous Simple; simultaneous trigger function by ADC0, ADC0 with ADC1 both convert finish then generate interrupt ADC0IF. 11 = Simultaneous Sequential; simultaneous trigger function by ADC0, this mode converts sequential is ADC0 -> ADC1 ->ADC0 -> ADC1 4 times, then generate interrupt ADC0IF.
[5]	**ADCSS3R**	**ADC Simultaneous Sequential 3 data at ADCMODE = 11** 0 = convert sequential is ADC0 -> ADC1 -> ADC0 -> ADC1, four datas at ADCMODE=11. 1 = convert sequential is ADC0 -> ADC1 -> ADC0, three datas at ADCMODE=11.

Bits	Description	
[4]	**Reserved**	Reserved.
[3]	**ADC0SWTRG**	**ADC0 Conversion Start** 0 = Conversion stopped and A/D converter entered idle state. 1 = Conversion start. **Note:** ADC0SWTRG can be set to "1" from two sources: software and external pin STADC. This bit will be cleared to "0" by hardware automatically.
[2]	**ADC0HWTRGEN**	**Hardware Trigger ADC Convertion Enable** Enable or disable triggering of A/D conversion by Hardware (PWM, Timer, ADC self) 0= Disabled. 1= Enabled.
[1]	**ADC0IEN**	**ADC0 Interrupt Enable** 0 = ADC0 interrupt function Disabled. 1 = ADC0 interrupt function Enabled. **Note:** A/D conversion end interrupt request is generated if ADC0IEN bit is set to "1".
[0]	**ADCEN**	**ADC Converter Enable** 0 = ADC Converter Disabled. 1 = ADC Converter Enabled. **Note:** 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.

ADC Hardware Trigger Source Control Register (ADC_TRGSOR)

Register	Offset	R/W	Description	Reset Value
ADC_TRGSOR	ADC_BA+0x24	R/W	ADC Hardware Trigger Source Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ADC1STADCSEL		**ADC1PWMTRGSEL**		**ADC1TRGSOR**			
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADC0STADCSEL		**ADC0PWMTRGSEL**		**ADC0TRGSOR**			

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:22]	**ADC1STADCSEL**	**ADC1 External Trigger Pin (STADC) Trigger Selection** 00 = Rising. 01 = Falling. 10 = Rising or Falling. 11 = Reserved.
[21:20]	**ADC1PWMTRGSEL**	**PWM Trigger Selection for ADC1** 00 = EPWM Signal Falling. 01 = EPWM Counter Central. 10 = EPWM signal Rising. 11 = Period.
[19:16]	**ADC1TRGSOR**	**ADC1 Trigger Source** 0000 = STADC. 0001 = PWM0. 0010 = PWM1. 0011 = PWM2. 0100 = PWM3. 0101 = PWM4. 0110 = PWM5. 0111 = TMR0. 1000 = TMR1. 1001 = TMR2. 1010 = ADC0IF. 1011 = ADC1IF. 1100~1111 = Reserved.
[15:8]	**Reserved**	Reserved.

Bits	Description	
[7:6]	**ADC0STADCSEL**	**ADC0 External Trigger Pin (STADC) Trigger Selection** 00 = Rising. 01 = Falling. 10 = Rising or Falling. 11 = Reserved.
[5:4]	**ADC0PWMTRGSEL**	**PWM Trigger Selection for ADC0** 00 = EPWM Signal Falling. 01 = EPWM Counter Central. 10 = EPWM signal Rising. 11 = Period.
[3:0]	**ADC0TRGSOR**	**ADC0 Trigger Source** 0000 = STADC. 0001 = PWM0. 0010 = PWM1. 0011 = PWM2. 0100 = PWM3. 0101 = PWM4. 0110 = PWM5. 0111 = TMR0. 1000 = TMR1. 1001 = TMR2. 1010 = ADC0IF. 1011 = ADC1IF. 1100~1111 = Reserved.

ADC Trigger Delay Control Register (ADC_TRGDLY)

Register	Offset	R/W	Description	Reset Value
ADC_TRGDLY	ADC_BA+0x28	R/W	ADC Trigger Delay Control Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
ADC1DELAY							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADC0DELAY							

Bits	Description	
[31:24]	**Reserved**	Reserved.
[23:16]	**ADC1DELAY**	**ADC1 Trigger Delay Timer** Setting this field will delay ADC start conversion time after ADCxTRGCTL trigger is coming. (x:0/1) Delay time is (4 * **ADC1DELAY**) * system clock
[15:8]	**Reserved**	Reserved.
[7:0]	**ADC0DELAY**	**ADC0 Trigger Delay Timer** Setting this field will delay ADC start conversion time after ADCxTRGCTL trigger is coming. (x:0/1) Delay time is (4 * **ADC0DELAY**) * system clock

ADC Sampling Time Counter Register (ADC_SMPCNT)

Register	Offset	R/W	Description	Reset Value
ADC_SMPCNT	ADC_BA+0x2C	R/W	ADC Sampling Time Counter Register	0x0000_0005

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				**ADCSMPCNT**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**ADCSMPCNT**	**ADC Sampling Counter** **ADC sampling counters are 6 ADC clock is suggestion** **0 = 1 * ADC Clock.** **1 = 2 * ADC Clock.** **2 = 3 * ADC Clock.** **3 = 4 * ADC Clock.** **4 = 5 * ADC Clock.** **5 = 6 * ADC Clock.** **6 = 7 * ADC Clock.** **7 = 8 * ADC Clock.** **8 = 16 * ADC Clock.** **9 = 32 * ADC Clock.** **10 = 64 * ADC Clock.** **11 = 128 * ADC Clock.** **12 = 256 * ADC Clock.** **13 = 512 * ADC Clock.** **14 = 1024 * ADC Clock.** **15 = 1024 * ADC Clock.**

ADC Status Register (ADC_STATUS)

Register	Offset	R/W	Description	Reset Value
ADC_STATUS	ADC_BA+0x30	R/W	ADC Status Register	0x0000_0000

31	30	29	28	27	26	25	24
Reserved							
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Reserved				**HIGHFG**	**MIDFG**	**LOWFG**	**WCMPIF**
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
ADC1CH				**ADC1BUSY**	**Reserved**	**ADC1OV**	**ADC1IF**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
ADC0CH				**ADC0BUSY**	**Reserved**	**ADC0OV**	**ADC0IF**

Bits	Description	
[31:20]	**Reserved**	Reserved.
[19]	**HIGHFG**	**Window Comparator High Bound Flag** When A/D conversion result higher than the setting condition in High Bound (WCMPHIGHDAT), this bit is set to “1”. Then it is cleared by writing “1” to ifself. 0 = Conversion result in ADC_DAT1 does not meet the WCMPHIGHDAT setting. 1 = Conversion result in ADC_DAT1 meets the WCMPHIGHDAT setting.
[18]	**MIDFG**	**Window Comparator Middle Bound Flag** When A/D conversion result is between High Bound (WCMPHIGHDAT) and Low Bound (WCMPLOWDAT), this bit is set to “1”. Then it is cleared by writing “1” to ifself. 0 = Conversion result in ADC_DAT1 isn't between High Bound (WCMPHIGHDAT) and Low Bound (WCMPLOWDAT). 1 = Conversion result in ADC_DAT1 is between High Bound (WCMPHIGHDAT) and Low Bound (WCMPLOWDAT).
[17]	**LOWFG**	**Window Comparator Low Bound Flag** When A/D conversion result lower than the setting condition in Low Bound (WCMPLOWDAT), this bit is set to “1”. Then it is cleared by writing “1” to ifself. 0 = Conversion result in ADC_DAT1 does not meet the WCMPLOWDAT setting. 1 = Conversion result in ADC_DAT1 meets the WCMPLOWDAT setting.
[16]	**WCMPIF**	**Window Comparator Interrupt Flag** When Windows Comparator has generat a result output, this bit is set to “1”. Then it is cleared by writing “1” to ifself. 0 = Conversion result in ADC_DAT1 does not meets the WCMPLOWDAT setting. 1 = Conversion result in ADC_DAT1 meets the WCMPLOWDAT setting.

Bits	Description	
[15:12]	**ADC1CH**	**Current Conversion Channel** This filed reflects the current conversion channel when ADC1BUSY =1. When ADC1BUSY =0, it shows the number of the next converted channel. It is read only.
[11]	**ADC1BUSY**	**BUSY/IDLE** 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion. This bit is mirror of as ADST bit in ADCR.
[10]	**Reserved**	Reserved.
[9]	**ADC1OV**	**Over Run Flag** It is a mirror to OV bit in ADDR.
[8]	**ADC1IF**	**ADC1 Conversion End Flag** A status flag that indicates the end of A/D conversion. ADF is set to “1” When A/D conversion ends. This flag can be cleared by writing “1” to itself.
[7:4]	**ADC0CH**	**Current Conversion Channel** This filed reflects the current conversion channel when ADC0BUSY =1. When ADC0BUSY =0, it shows the number of the next converted channel. It is read only.
[3]	**ADC0BUSY**	**BUSY/IDLE** 0 = A/D converter is in idle state. 1 = A/D converter is busy at conversion. This bit is mirror of as ADST bit in ADCR.
[2]	**Reserved**	Reserved.
[1]	**ADC0OV**	**Over Run Flag** It is a mirror to OV bit in ADDR.
[0]	**ADC0IF**	**A/D Conversion End Flag** A status flag that indicates the end of A/D conversion. ADF is set to “1” When A/D conversion ends. This flag can be cleared by writing “1” to itself.

ADC Window Comparator Control Register (ADC_WCMPCTL)

Register	Offset	R/W	Description	Reset Value
ADC_WCMPCTL	ADC_BA+0x34	R/W	ADC Window Comparator 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				WCMPMCNT			
7	6	5	4	3	2	1	0
WFLAGCTL	WCMPHIGHEN	WCMPMIDEN	WCMPLOWEN	Reserved		WCMPIEN	WCMPEN

Bits	Description	
[31:12]	**Reserved**	Reserved.
[11:8]	**WCMPMCNT**	**Window Compare Match Count** When the A/D conversion result matches the compare condition defined by CMP Flag setting (CMPUPEN, CMPEQUEN, CMPLOWEN and WCFLAGCTL), the internal match counter will increase 1. When the internal counter reaches the value to (WCMPMCNT), the CMPIF bit will be set. **NOTE:**If WCMPMCNT = 0,the counter would do 16 times.
[7]	**WFLAGCTL**	**Window Comparator Flag Control** When the A/D conversion result matches the compare condition 0 = Auto-update. 1 = none.
[6]	**WCMPHIGHEN**	**Window Comparator High Flag Enable Bit** Set A/D conversion result higher than compare condition High bound range 0 = Window Comparator High Flag Disabled. 1 = Window Comparator High Flag Enabled.
[5]	**WCMPMIDEN**	**Window Comparator Middle Flag Enable Bit** Set A/D conversion result equal to compare condition at Low and High bound range 0 = Window Comparator Middle Flag Disabled. 1 = Window Comparator Middle Flag Enabled.
[4]	**WCMPLOWEN**	**Window Comparator Low Flag Enable Bit** Set A/D conversion result lower than compare condition Low bound range 0 = Window Comparator Low Flag Disabled. 1 = Window Comparator Low Flag Enabled.
[3:2]	**Reserved**	Reserved.

Bits	Description	
[1]	**WCMPIEN**	**Window Comparator Interrupt Enable Bit** 0 = Window Comparator Interrupt Disabled. 1 = Window Comparator Interrupt Enabled.
[0]	**WCMPEN**	**Window Comparator Enable Bit** 0 = Window Comparator Disabled. 1 = Window Comparator Enabled.

ADC Window Comparator Data Register (ADC_WCMPDAT)

Register	Offset	R/W	Description	Reset Value
ADC_WCMPDAT	ADC_BA+0x38	R/W	ADC Window Comparator Data Register	0x0000_0000

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

Bits	Description	
[31:28]	**Reserved**	Reserved.
[27:16]	**WCMPHIGHDAT**	Window Comparator High Bound Data
[15:12]	**Reserved**	Reserved.
[11:0]	**WCMPLOWDAT**	Window Comparator Low Bound Data

6.17 Analog Comparator (ACMP)

6.17.1 Overview

The NDA102 series contains two comparators which can be used in a number of different configurations. The comparator output is logic 1 when positive input greater than negative input, otherwise the output is 0. Each comparator can be configured to generate interrupt when the comparator output value changes.

6.17.2 Features

- Analog input voltage range: 0 ~ V_{DD}
- Supports Hysteresis function
- Optional internal reference voltage source for each comparator negative input

6.17.3 Block Diagram

Figure 6.17-1 Analog Comparator Block Diagram

6.17.4 Basic Configuration

The ACMP pin functions are configured in SYS_GPB_MFP, SYS_GPC_MFP and SYS_GPD_MFP registers. It is recommended to disable the digital input path of the analog input pins to avoid the leakage current. The digital input path can be disabled by configuring PB_DINOFF, PC_DINOFF and PD_DINOFF registers. If a GPIO pin is configured as an ACMP input pin, this pin should not be set as Push-pull Output mode in the Px_MODE register. Input mode is the safest configuration. If Open-drain, Output mode or Quasi-bidirectional mode is selected, do not output 0 on this GPIO pin. The default GPIO output value is 1. The default PB_MODE, PC_MODE and PD_MODE setting is determined by user configuration. It could be configured as Input mode or Quasi-bidirectional mode in user configuration.

The ACMP peripheral clocks can be enabled by setting ACMPCKEN (CLK_APBCLK [30]) to 1.

6.17.5 Functional Description

6.17.5.1 Interrupt Sources

The output of comparators are sampled by PCLK and reflected at ACMPOx(ACMP_STATUS[3] and ACMP_STATUS[2]). If ACMPIE(ACMP_CTLx[1]) 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, ACMPFx(ACMP_STATUS[1] and ACMP_STATUS[0]), will be set. Software can clear the flag to 0 by writing 1 to it.

Figure 6.17-2 Analog Comparator Controller Interrupt Sources

6.17.5.2 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 the negative input voltage by a negative hysteresis voltage.

Figure 6.17-3 Comparator Hysteresis Function

6.17.6 Comparator Reference Voltage (CRV)

6.17.6.1 Introduction

The comparator reference voltage (CRV) module is responsible for generating reference voltage for comparators. The CRV module consists of resisters ladder and analog switch, and user can set the CRV output voltage using CRVCTL(ACMP_VREF[3:0]) and select the reference voltage to ACMP by setting CPNSEL (ACMP_CTL[25:24]).

6.17.6.2 Features

- User selectable references voltage by setting CRVCTL(ACMP_ VREF [3:0])
- Automatic disable resisters ladder for reducing power consumption when setting CPNSEL (ACMP_CTL[25:24]) = 01b (selecting Band-gap source voltage)

The block diagram of the CRV module is shown in Figure 6.17-4.

Figure 6.17-4 Comparator Reference Voltage Block Diagram

6.17.7 Register Map

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

Register	Offset	R/W	Description	Reset Value
ACMP Base Address: **ACMP_BA = 0x400D_0000**				
ACMP_CTL0	ACMP_BA+0x00	R/W	Analog Comparator0 Control Register	0x0000_0000
ACMP_CTL1	ACMP_BA+0x04	R/W	Analog Comparator1 Control Register	0x0000_0000
ACMP_STATUS	ACMP_BA+0x08	R/W	Analog Comparator Status Register	0x0000_0000
ACMP_VREF	ACMP_BA+0x0C	R/W	Analog Comparator Reference Voltage Control Register	0x0000_0000
ACMP_TRGDLY	ACMP_BA+0x10	R/W	Analog Comparator Delay Trigger Mode Dleay Register	0x0000_0000

6.17.8 Register Description

Analog Comparator0 Control Register (ACMP_CTL0)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL0	ACMP_BA+0x00	R/W	Analog Comparator0 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRESET	**CPPSEL**			**Reserved**		**CPNSEL**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
NFDIS	**Reserved**	**NFCLKS**		**POLARITY**	**Reserved**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**DLYTRGIE**	**DLYTRGEN**	**DLYTRGSOR**		**DLYTRGSEL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**PBRKSEL**	**EDGESEL**		**ACMPHYSEN**		**ACMPIE**	**ACMPEN**

Bits	Description	
[31]	**PRESET**	**Comparator Result Preset Value** 0 = 0 for preset value. 1 = 1 for preset value.
[30:28]	**CPPSEL**	**Comparator Positive Input Select** 000 = ACMP0_P0 (PB.0). 001 = ACMP0_P1 (PB.1). 010 = ACMP0_P2 (PB.2). 011 = ACMP0_P3 (PC.1). 100 = PGA_CMP.
[27:26]	**Reserved**	Reserved.
[25:24]	**CPNSEL**	**Comparator Negative Input Select** 00 = ACMP0_N (PB.4). 01 = Band_Gap. 10 = CRV. 11 = Reserved.
[23]	**NFDIS**	**Disable Comparator Noise Filter** 0 = Noise filter Enabled. 1 = Noise filter Disabled.
[22]	**Reserved**	Reserved.

Bits	Description	
[21:20]	**NFCLKS**	**Noise Filter Clock Pre-divided Selection** To determine the sampling frequency of the Noise Filter clock 00 = PCLK. 01 = PCLK / 2. 10 = PCLK / 4. 11 = PCLK / 16.
[19]	**POLARITY**	**Analog Comparator Polarity Control** 0 = Analog Comparator normal output. 1 = Analog Comparator invert output.
[18:14]	**Reserved**	Reserved.
[13]	**DLYTRGIE**	**Analog Comparator Delay Trigger Mode Interrupt Enable Bit** 0 = Analog Comparator Delay Trigger Mode Interrupt Disabled. 1 = Analog Comparator Delay Trigger Mode Interrupt Enabled.
[12]	**DLYTRGEN**	**Analog Comparator Delay Trigger Mode Enable Bit** 0 = Analog Comparator Delay Trigger Mode Disabled. 1 = Analog Comparator Delay Trigger Mode Enabled.
[11:10]	**DLYTRGSOR**	**Analog Comparator Delay Trigger Mode Trigger Source Selection** 00 = PWM0. 01 = PWM2. 10 = PWM4. 11 = Reserved.
[9:8]	**DLYTRGSEL**	**Analog Comparator Delay Trigger Mode Trigger Level Selection** 00 = Analog Comparator Delay Trigger Mode Trigger Disabled. 01 = Rising. 10 = Falling. 11 = Rising/Falling.
[7]	**Reserved**	Reserved.
[6]	**PBRKSEL**	**ACMP to EPWM Brake Selection** 0 = ACMP Result direct output. 1 = ACMP Delay Trigger Result output.
[5:4]	**EDGESEL**	**Interrupt Flag Trigger Edge Detection** 00 = Interrupt Flag Trigger Edge Disabled. 01 = Rising. 10 = Falling. 11 = Rising/Falling.
[3:2]	**ACMPHYSEN**	**Comparator0 Hysteresis Enable Bit (Only 20mV)** 00 = ACMP0 Hysteresis function Disabled (Default). 01 = ACMP0 Hysteresis function at comparator 0 Enabled that the typical range is 10mV. 10 = ACMP0 Hysteresis function at comparator 0 Enabled that the typical range is 90mV. 11 = ACMP0 Hysteresis function Disabled.

Bits	Description	
[1]	**ACMPIE**	**Comparator Interrupt Enable Bit** 0 = ACMP interrupt function Disabled. 1 = ACMP interrupt function Enabled. **Note1:** Interrupt is generated if ACMPIE bit is set to “1” after ACMP conversion is finished. **Note2:** ACMP interrupt will wake CPU up in Power-down mode.
[0]	**ACMPEN**	**Comparator Enable Bit** 0 = Comparator Disabled. 1 = Comparator Enabled. **Note:** Comparator output needs to wait 2 us stable time after ACMPEN is set.

Analog Comparator1 Control Register (ACMP_CTL1)

Register	Offset	R/W	Description	Reset Value
ACMP_CTL1	ACMP_BA+0x04	R/W	Analog Comparator1 Control Register	0x0000_0000

31	30	29	28	27	26	25	24
PRESET	**CPPSEL**			**Reserved**		**CPNSEL**	
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
NFDIS	**Reserved**	**NFCLKS**		**POLARITY**	**Reserved**		
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved		**DLYTRGIE**	**DLYTRGEN**	**DLYTRGSOR**		**DLYTRGSEL**	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved	**PBRKSEL**	**EDGESEL**		**ACMPHYSEN**		**ACMPIE**	**ACMPEN**

Bits	Description	
[31]	**PRESET**	**Comparator Result Preset Value** 0 = 0 for preset value. 1 = 1 for preset value.
[30:28]	**CPPSEL**	**Comparator Positive Input Selection** 000 = ACMP1_P0 (PC.0). 001 = ACMP1_P1 (PC.1). 010 = ACMP1_P2 (PD.1). 011 = PGA_CMP.
[27:26]	**Reserved**	Reserved.
[25:24]	**CPNSEL**	**Comparator Negative Input Selection** 00 = ACMP1_N (PB.3). 01 = Band_Gap. 10 = CRV. 11 = Reserved.
[23]	**NFDIS**	**Disable Comparator Noise Filter** 0 = Noise filter Enable. 1 = Noise filter Disable.
[22]	**Reserved**	Reserved.
[21:20]	**NFCLKS**	**Noise Filter Clock Pre-divided Selection** To determine the sampling frequency of the Noise Filter clock 00 = PCLK. 01 = PCLK / 2. 10 = PCLK / 4. 11 = PCLK / 16.

Bits	Description	
[19]	**POLARITY**	**Analog Comparator Polarity Control** 0 = Analog Comparator normal output. 1 = Analog Comparator invert output.
[18:14]	**Reserved**	Reserved.
[13]	**DLYTRGIE**	**Analog Comparator Delay Trigger Mode Interrupt Enable** 0 = Analog Comparator Delay Trigger Mode Interrupt Disabled. 1 = Analog Comparator Delay Trigger Mode Interrupt Enabled.
[12]	**DLYTRGEN**	**Analog Comparator Delay Trigger Mode Enable** 0 = Analog Comparator Delay Trigger Mode Disabled. 1 = Analog Comparator Delay Trigger Mode Enabled.
[11:10]	**DLYTRGSOR**	**Analog Comparator Delay Trigger Mode Trigger Source Selection** 00 = PWM0. 01 = PWM2. 10 = PWM4. 11 = Reserved.
[9:8]	**DLYTRGSEL**	**Analog Comparator Delay Trigger Mode Trigger Level Selection** 00 = Analog Comparator Delay Trigger Mode Trigger Disabled. 01 = Rising. 10 = Falling. 11 = Rising/Falling.
[7]	**Reserved**	Reserved.
[6]	**PBRKSEL**	**ACMP to EPWM Brake Selection** 0 = ACMP Result direct output. 1 = ACMP Delay Trigger Result output.
[5:4]	**EDGESEL**	**Interrupt Flag Trigger Edge Detection** 00 = Interrupt Flag Trigger Edge Detection Disable. 01 = Rising. 10 = Falling. 11 = Rising/Falling.
[3:2]	**ACMPHYSEN**	**Comparator1 Hysteresis Enable Bit (Only 20mV)** 00 = ACMP0 Hysteresis function Disabled (Default). 01 = ACMP0 Hysteresis function at comparator 0 Enabled that the typical range is 10mV. 10 = ACMP0 Hysteresis function at comparator 0 Enabled that the typical range is 90mV. 11 = ACMP0 Hysteresis function Disabled.
[1]	**ACMPIE**	**Comparator Interrupt Enable Bit** 0 = ACMP interrupt function Disabled. 1 = ACMP interrupt function Enabled. **Note1:** Interrupt is generated if ACMPIE bit is set to “1” after ACMP conversion is finished. **Note2:** ACMP interrupt will wake CPU up in Power-down mode.

Bits	Description	
[0]	**ACMPEN**	**Comparator Enable Bit** 0 = Comparator Disabled. 1 = Comparator Enabled. **Note:** Comparator output needs to wait 2 us stable time after ACMPEN is set.

Analog Comparator Status Register (ACMP_STATUS)

Register	Offset	R/W	Description	Reset Value
ACMP_STATUS	ACMP_BA+0x08	R/W	Analog Comparator 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**
DLYTRGO1	**DLYTRGO0**	**DLYTRGF1**	**DLYTRGF0**	**ACMPO1**	**ACMPO0**	**ACMPF1**	**ACMPF0**

Bits	Description	
[31:8]	**Reserved**	Reserved.
[7]	**DLYTRGO1**	**Analog Comparator1 Delay Trigger Mode Comparator Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (DLYTRGEN = 0).
[6]	**DLYTRGO0**	**Analog Comparator0 Delay Trigger Mode Comparator Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (DLYTRGEN = 0).
[5]	**DLYTRGF1**	**Comparator1 Flag** This bit is set by hardware whenever the comparator1 output changes state. This will cause an interrupt if DLYTRGIEN set. **Note:** Write “1” to clear this bit to 0.
[4]	**DLYTRGF0**	**Comparator0 Flag** This bit is set by hardware whenever the comparator0 output changes state. This will cause an interrupt if DLYTRGIEN set. **Note:** Write “1” to clear this bit to 0.
[3]	**ACMPO1**	**Comparator1 Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (ACMPEN = 0).
[2]	**ACMPO0**	**Comparator0 Output** Synchronized to the APB clock to allow reading by software. Cleared when the comparator is disabled (ACMPEN = 0).
[1]	**ACMPF1**	**Comparator1 Flag** This bit is set by hardware whenever the comparator1 output changes state. This will cause an interrupt if ACMPIE set. **Note:** Write “1” to clear this bit to 0.

Bits	Description	
[0]	**ACMPF0**	**Comparator0 Flag** This bit is set by hardware whenever the comparator0 output changes state. This will cause an interrupt if ACMPIE set. **Note:** Write “1” to clear this bit to 0.

Analog Comparator Reference Voltage Control Register (ACMP_VREF)

Register	Offset	R/W	Description	Reset Value
ACMP_VREF	ACMP_BA+0x0C	R/W	Analog Comparator Reference Voltage 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				**CRVCTL**			

Bits	Description	
[31:4]	**Reserved**	Reserved.
[3:0]	**CRVCTL**	**Comparator Reference Voltage Setting** CRVS = AV_{DD} x (1/6+CRV[3:0]/24).

Analog Comparator Delay Trigger Mode Dleay Register (ACMP_TRGDLY)

Register	Offset	R/W	Description	Reset Value
ACMP_TRGDLY	ACMP_BA+0x10	R/W	Analog Comparator Delay Trigger Mode Dleay 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							**DELAY**
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DELAY							

Bits	Description	
[31:9]	**Reserved**	Reserved.
[8:0]	**DELAY**	**Analog Comparator Delay Trigger Mode Dleay cycle**

6.18 Programmable Gain Amplifier (PGA)

6.18.1 Overview

The NDA102 series contains a programmable gain amplifier (PGA) which can be enabled through the PGAEN bit. User can measure the outputs of the programmable gain amplifier as the programmable gain amplifier output to the integrated A/D converter channel, where digital results can be taken. Furthermore, user can adjust gain to 1, 2, 3, 5, 7, 9, 11, and 13.

Note: The analog input port pins must be configured as input type before the PGA function is enabled.

1.1.4 Features

- Supports analog input voltage range: 0~ V_{DD}.
- Supports programmable gain: 1,2, 3,5,7,9.11,13
- Supports PGA output as input of ADC and ACMP

1.1.5 Block Diagram

Figure 6.18-1 OP Amplifier Block Diagram

1.1.6 Register Map

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

Register	Offset	R/W	Description	Reset Value
PGA Base Address: **PGA_BA = 0x400F_0000**				
PGA_CTL	PGA_BA+0x00	R/W	Programmable Gain Amplifier Control Register	0x0000_0000

6.18.2 Register Description

PGA Control Register (PGA_CTL)

Register	Offset	R/W	Description	Reset Value
PGA_CTL	PGA_BA+0x00	R/W	Programmable Gain Amplifier 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	**GAIN**			**Reserved**			**PGAEN**

Bits	Description	
[31:7]	**Reserved**	Reserved.
[6:4]	**GAIN**	**PGA Gain Selection** 000 = 2. 001 = 3. 010 = 5. 011 = 7. 100 = 9. 101 = 11. 110 = 13. 111 = 1 . (*See Note)
[3:1]	**Reserved**	Reserved.
[0]	**PGAEN**	**Programmable Gain Amplifier Enable Bit** 0 = Programmable Gain Amplifier Disabled. 1 = Programmable Gain Amplifier Enabled. **Note:** The PGA output needs to wait stable 20μs after PGAEN is first set.

Note:: GAIN =111; PGA uses bypass path; Software must write PGA_EN to disable.

7 APPLICATION CIRCUIT

8 ELECTRICAL CHARACTERISTICS

For information on the NDA102 series electrical characteristics, please refer to NuMicro® NDA102 Series Datasheet.

9 PACKAGE DIMENSIONS

1.2 28-Pin TSSOP (4.4x9.7x1.0 mm)

VARIATIONS (ALL DIMENSIONS SHOWN IN MM)

SYMBOLS	MIN.	NOM.	MAX.
A	–	–	1.20
A1	0.00	–	0.15
A2	0.80	1.00	1.05
b	0.19	–	0.30
c	0.09	–	0.20
D	9.60	9.70	9.80
E1	4.30	4.40	4.50
E	6.40 BSC		
e	0.65 BSC		
L1	1.00 REF		
L	0.45	0.60	0.75
S	0.20	–	–
θ	0°	–	8°
Y	0.10		

10 REVISION HISTORY

Date	Revision	Description
2018.03.06	1.00	Preliminary version.

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.