zilog

Embedded in Life

An IXYS Company

## Z8051 Series 8-Bit Microcontrollers

# Z51F0811

## Product Specification

PS029602-0212

PRELIMINARY

Copyright ©2012 Zilog®, Inc. All rights reserved.

www.zilog.com

**Warning:** DO NOT USE THIS PRODUCT IN LIFE SUPPORT SYSTEMS.

## LIFE SUPPORT POLICY

ZILOG'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS PRIOR WRITTEN APPROVAL OF THE PRESIDENT AND GENERAL COUNSEL OF ZILOG CORPORATION.

### As used herein

Life support devices or systems are devices which (a) are intended for surgical implant into the body, or (b) support or sustain life and whose failure to perform when properly used in accordance with instructions for use provided in the labeling can be reasonably expected to result in a significant injury to the user. A critical component is any component in a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system or to affect its safety or effectiveness.

### Document Disclaimer

©2012 Zilog, Inc. All rights reserved. Information in this publication concerning the devices, applications, or technology described is intended to suggest possible uses and may be superseded. ZILOG, INC. DOES NOT ASSUME LIABILITY FOR OR PROVIDE A REPRESENTATION OF ACCURACY OF THE INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED IN THIS DOCUMENT. ZILOG ALSO DOES NOT ASSUME LIABILITY FOR INTELLECTUAL PROPERTY INFRINGEMENT RELATED IN ANY MANNER TO USE OF INFORMATION, DEVICES, OR TECHNOLOGY DESCRIBED HEREIN OR OTHERWISE. The information contained within this document has been verified according to the general principles of electrical and mechanical engineering.

Z8051 is a trademark or registered trademark of Zilog, Inc. All other product or service names are the property of their respective owners.

# Revision History

Each instance in this document's revision history reflects a change from its previous edition. For more details, refer to the corresponding page(s) or appropriate links furnished in the table below.

| Date | Revision Level | Description | Page |
|---|---|---|---|
| Feb 2012 | 02 | Removed references to SOP, LQFP packages. | All |
| Jan 2012 | 01 | Original Zilog issue. | All |

# Table of Contents

Z51F0811 ..... 12
1. Overview ..... 12
1.1 Description ..... 12
1.2 Features ..... 13
1.3 Ordering Information ..... 14
1.3.1 Part Number Suffix Designation ..... 14
1.4 Development Tools ..... 15
1.4.1 Compiler ..... 15
1.4.2 OCD emulator and debugger ..... 15
1.4.3 Programmer ..... 15
2. Block Diagram ..... 16
3. Pin Assignment ..... 17
4. Package Diagram ..... 21
5. Pin Description ..... 25
6. Port Structures ..... 26
6.1 General Purpose I/O Port ..... 26
6.2 External Interrupt I/O Port ..... 27
7. Electrical Characteristics ..... 28
7.1 Absolute Maximum Ratings ..... 28
7.2 Recommended Operating Conditions ..... 28
7.3 A/D Converter Characteristics ..... 29
7.4 Analog Comparator Characteristics ..... 29
7.5 Voltage Dropout Converter Characteristics ..... 30
7.6 Power-On Reset Characteristics ..... 31
7.7 Brown Out Detector Characteristics ..... 31
7.8 Internal RC Oscillator Characteristics ..... 32
7.9 Ring-Oscillator Characteristics ..... 32
7.10 PLL Characteristics ..... 32
7.11 DC Characteristics ..... 33
7.12 AC Characteristics ..... 34
7.13 SPI Characteristics ..... 35
7.14 Typical Characteristics ..... 36
8. Memory ..... 37
8.1 Program Memory ..... 37
8.2 Data Memory ..... 39
8.3 EEPROM Data Memory ..... 41
8.4 SFR Map ..... 42
8.4.1 SFR Map Summary ..... 42
8.4.2 Compiler Compatible SFR ..... 42
9. I/O Ports ..... 45
9.1 I/O Ports ..... 45
9.2 Port Register ..... 45
9.2.1 Data Register (Px) ..... 45
9.2.2 Direction Register (PxIO) ..... 45

9.2.3 Pull-up Resistor Selection Register (PxPU) ..... 45
9.2.4 Open-drain Selection Register (PxOD) ..... 45
9.2.5 Debounce Enable Register (PxDB) ..... 45
9.2.6 Pin Change Interrupt Enable Register (PCI0) ..... 45
9.2.7 Port Selection Register (PSRx) ..... 46
9.2.8 Register Map ..... 46
9.3 Px Port ..... 46
9.3.1 Px Port Description ..... 46
9.3.2 Register description for Px ..... 46
9.4 Port RESET Noise Canceller ..... 48
10. Interrupt Controller ..... 49
10.1 Overview ..... 49
10.2 External Interrupt ..... 49
10.3 Block Diagram ..... 51
10.4 Interrupt Vector Table ..... 52
10.5 Interrupt Sequence ..... 52
10.6 Effective Timing after Controlling Interrupt bit ..... 54
10.7 Multi Interrupt ..... 55
10.8 Interrupt Enable Accept Timing ..... 56
10.9 Interrupt Service Routine Address ..... 56
10.10 Saving/Restore General-Purpose Registers ..... 56
10.11 Interrupt Timing ..... 57
10.12 Interrupt Register Overview ..... 58
10.12.1 Interrupt Enable Register (IE, IE1, IE2, IE3, IE4, IE5) ..... 58
10.12.2 Interrupt Priority Register (IP, IP1) ..... 58
10.12.3 External Interrupt Flag Register (EIFLAG) ..... 58
10.12.4 External Interrupt Edge Register (EIEDGE) ..... 58
10.12.5 External Interrupt Polarity Register (EIPOLA) ..... 58
10.12.6 External Interrupt Enable Register (EIENAB) ..... 58
10.12.7 External Interrupt Both Edge Enable Register (EIBOTH) ..... 58
10.12.8 Register Map ..... 59
10.13 Interrupt Register Description ..... 59
10.13.1 Register description for Interrupt ..... 59
11. Peripheral Hardware ..... 65
11.1 Clock Generator ..... 65
11.1.1 Overview ..... 65
11.1.2 Block Diagram ..... 65
11.1.3 Register Map ..... 66
11.1.4 Clock Generator Register description ..... 66
11.1.5 Register description for Clock Generator ..... 66
11.2 BIT ..... 68
11.2.1 Overview ..... 68
11.2.2 Block Diagram ..... 68
11.2.3 Register Map ..... 68
11.2.4 Bit Interval Timer Register description ..... 69
11.2.5 Register description for Bit Interval Timer ..... 69
11.3 WDT ..... 70
11.3.1 Overview ..... 70

11.3.2 Block Diagram ..... 70
11.3.3 Register Map ..... 70
11.3.4 Watch Dog Timer Register description ..... 71
11.3.5 Register description for Watch Dog Timer ..... 71
11.3.6 WDT Interrupt Timing Waveform ..... 72
11.4 WT ..... 73
11.4.1 Overview ..... 73
11.4.2 Block Diagram ..... 73
11.4.3 Register Map ..... 73
11.4.4 Watch Timer Register description ..... 74
11.4.5 Register description for Watch Timer ..... 74
11.5 Timer/PWM ..... 76
11.5.1 8-bit Timer/Event Counter 0, 1 ..... 76
11.5.2 8-bit Timer/Event Counter 2, 3 ..... 103
11.5.3 16-Bit Timer 4 ..... 117
11.5.4 Timer Interrupt Status Register (TMISR) ..... 121
11.6 Buzzer Driver ..... 122
11.6.1 Overview ..... 122
11.6.2 Block Diagram ..... 122
11.6.3 Register Map ..... 123
11.6.4 Buzzer Driver Register description ..... 123
11.6.5 Register description for Buzzer Driver ..... 123
11.7 USART ..... 124
11.7.1 Overview ..... 124
11.7.2 Block Diagram ..... 125
11.7.3 Clock Generation ..... 126
11.7.4 External Clock (XCK) ..... 127
11.7.5 Synchronous mode Operation ..... 127
11.7.6 Data format ..... 128
11.7.7 Parity bit ..... 129
11.7.8 USART Transmitter ..... 129
11.7.9 USART Receiver ..... 130
11.7.10 SPI Mode ..... 133
11.7.11 Register Map ..... 136
11.7.12 USART Register description ..... 136
11.7.13 Register description for USART ..... 137
11.7.14 Baud Rate setting (example) ..... 141
11.8 SPI ..... 142
11.8.1 Overview ..... 142
11.8.2 Block Diagram ..... 142
11.8.3 Data Transmit / Receive Operation ..... 143
11.8.4 SS pin function ..... 143
11.8.5 Timing Waveform ..... 144
11.8.6 Register Map ..... 144
11.8.7 SPI Register description ..... 144
11.8.8 Register description for SPI ..... 145
11.9 $I^2C$ ..... 147
11.9.1 Overview ..... 147

11.9.2 Block Diagram ..... 147
11.9.3 I²C Bit Transfer ..... 148
11.9.4 Start / Repeated Start / Stop ..... 148
11.9.5 Data Transfer ..... 148
11.9.6 Acknowledge ..... 149
11.9.7 Synchronization / Arbitration ..... 150
11.9.8 Operation ..... 151
11.9.9 Register Map ..... 159
11.9.10 I²C Register description ..... 160
11.9.11 Register description for I²C ..... 160
11.10 12-Bit A/D Converter ..... 164
11.10.1 Overview ..... 164
11.10.2 Block Diagram ..... 164
11.10.3 ADC Operation ..... 165
11.10.4 Register Map ..... 166
11.10.5 ADC Register description ..... 166
11.10.6 Register description for ADC ..... 167
11.11 Analog Comparator ..... 170
11.11.1 Overview ..... 170
11.11.2 Block Diagram ..... 170
11.11.3 IN/OUT signal description ..... 170
11.11.4 Register Map ..... 171
11.11.5 Analog Comparator Register description ..... 171
11.11.6 Register description for Analog Comparator ..... 172
12. Power Down Operation ..... 173
12.1 Overview ..... 173
12.2 Peripheral Operation in IDLE/STOP Mode ..... 173
12.3 IDLE mode ..... 174
12.4 STOP mode ..... 174
12.5 Release Operation of STOP1, 2 Mode ..... 175
12.5.1 Register Map ..... 177
12.5.2 Power Down Operation Register description ..... 177
12.5.3 Register description for Power Down Operation ..... 177
13. RESET ..... 178
13.1 Overview ..... 178
13.2 Reset source ..... 178
13.3 Block Diagram ..... 178
13.4 RESET Noise Canceller ..... 179
13.5 Power ON RESET ..... 179
13.6 External RESETB Input ..... 182
13.7 Brown Out Detector Processor ..... 183
13.7.1 Register Map ..... 184
13.7.2 Reset Operation Register description ..... 184
13.7.3 Register description for Reset Operation ..... 184
14. On-chip Debug System ..... 186
14.1 Overview ..... 186
14.1.1 Description ..... 186
14.1.2 Feature ..... 186

14.2 Two-pin external interface ..... 187
14.2.1 Basic transmission packet ..... 187
14.2.2 Packet transmission timing ..... 188
14.2.3 Connection of transmission ..... 190
15. Memory Programming ..... 191
15.1 Overview ..... 191
15.1.1 Description ..... 191
15.1.2 Features ..... 191
15.2 Flash and EEPROM Control and status register ..... 191
15.2.1 Register Map ..... 191
15.2.2 Register description for Flash and EEPROM ..... 192
15.3 Memory map ..... 196
15.3.1 Flash Memory Map ..... 196
15.3.2 Data EEPROM Memory Map ..... 197
15.4 Serial In-System Program Mode ..... 198
15.4.1 Flash operation ..... 198
15.4.2 Data EEPROM operation ..... 202
15.4.3 Summary of Flash and Data EEPROM Program/Erase Mode ..... 205
15.5 Parallel Mode ..... 206
15.5.1 Overview ..... 206
15.5.2 Parallel Mode instruction format ..... 207
15.5.3 Parallel Mode timing diagram ..... 208
15.6 Mode entrance method of ISP and byte-parallel mode ..... 208
15.6.1 Mode entrance method for ISP ..... 208
15.6.2 Mode entrance of Byte-parallel ..... 209
15.7 Security ..... 210
16. Configure option ..... 211
16.1 Configure option Control Register ..... 211
17. APPENDIX ..... 212

# List of Figures

Figure 1-1 OCD Debugger and Pin description ..... **Error! Bookmark not defined.**
Figure 1-2 Single Programmer ..... 15
Figure 1-3 Gang Programmer ..... **Error! Bookmark not defined.**
Figure 2-1 Z51F0811 block diagram ..... 16
Figure 3-2 Z51F0811 32 QFN Pin assignment ..... 17
Figure 3-4 Z51F0811 28 TSSOP Pin assignment ..... 18
Figure 3-5 Z51F0811 20 TSSOP Pin assignment ..... 19
Figure 3-6 Z51F0811 16 TSSOP Pin assignment ..... 20
Figure 4-2 32 pin QFN package ..... 21
Figure 4-5 28 pin TSSOP package ..... 22
Figure 4-7 20 pin TSSOP package ..... 23
Figure 4-9 16 pin TSSOP package ..... 24
Figure 6-1 General Purpose I/O Port ..... 26
Figure 6-2 External Interrupt I/O Port ..... 27
Figure 7-1 AC Timing ..... 34
Figure 7-2 SPI Timing ..... 35
Figure 7-3 8MHz Internal OSC Freq.(OSC_HOT: 85℃) ..... 36
Figure 8-1 Program memory ..... 38
Figure 8-2 Data memory map ..... 39
Figure 8-3 Lower 128 bytes RAM ..... 40
Figure 8-4 XDATA memory area ..... 41
Figure 9-1 Port Reset noise canceller time diagram ..... 48
Figure 10-1 External Interrupt Description ..... 50
Figure 10-2 Block Diagram of Interrupt ..... 51
Figure 10-3 Interrupt Vector Address Table ..... 53
Figure 10-4 Interrupt Enable Register effective Timing ..... 54
Figure 10-5 Execution of Multi Interrupt ..... 55
Figure 10-6 Interrupt Response Timing Diagram ..... 56
Figure 10-7 Correspondence between vector Table address and the entry address of ISP . 56
Figure 10-8 Saving/Restore Process Diagram & Sample Source ..... 56
Figure 10-9 Timing chart of Interrupt Acceptance and Interrupt Return Instruction ..... 57
Figure 11-1 Clock Generator Block Diagram ..... 65
Figure 11-2 BIT Block Diagram ..... 68
Figure 11-3 WDT Block Diagram ..... 70
Figure 11-4 WDT Interrupt Timing Waveform ..... 72
Figure 11-5 Watch Timer Block Diagram ..... 73
Figure 11-6 8 Bit Timer/Event Counter 0, 1 Block Diagram ..... 77
Figure 11-7 Timer/Event Counter 0, 1 Example ..... 78
Figure 11-8 Timer/Event Counter0, 1 Count Operation ..... 78
Figure 11-9 16-bit Timer/Counter for Time 0, 1 ..... 79
Figure 11-10 8-bit Capture Mode for Timer 0, 1 ..... 80

Figure 11-11 Input Capture Mode Operation of Timer 0, 1 ..... 81
Figure 11-12 Express Timer Overflow in Capture Mode ..... 81
Figure 11-13 16-bit Capture Mode of Timer 0, 1 ..... 82
Figure 11-14 PWM Mode (Force 6-ch) ..... 84
Figure 11-15 PWM Mode (Force All-Ch) ..... 85
Figure 11-16 Example of PWM at 4MHz ..... 86
Figure 11-17 Example of Changing the Period in Absolute Duty Cycle at 4Mhz ..... 86
Figure 11-18 Example of PWM Output Waveform ..... 87
Figure 11-19 Example of PWM waveform in Back-to-Back mode at 4Mhz ..... 87
Figure 11-20 Example of Phase Correction and Frequency correction of PWM ..... 88
Figure 11-21 Example of PWM External Synchronization with BLNKB Input ..... 89
Figure 11-22 Example of Force Drive All-ch with A-ch ..... 89
Figure 11-23 Example of Force Drive 6-ch Mode ..... 90
Figure 11-24 PWM Port control ..... 91
Figure 11-25 Example of PWM Delay ..... 92
Figure 11-26 8 Bit Timer/Event Counter2, 3 Block Diagram ..... 104
Figure 11-27 Timer/Event Counter2, 3 Example ..... 105
Figure 11-28 Operation Example of Timer/Event Counter2, 3 ..... 105
Figure 11-29 16 Bit Timer/Event Counter2, 3 Block Diagram ..... 106
Figure 11-30 8-bit Capture Mode for Timer2, 3 ..... 107
Figure 11-31 Input Capture Mode Operation of Timer 2, 3 ..... 108
Figure 11-32 Express Timer Overflow in Capture Mode ..... 108
Figure 11-33 16-bit Capture Mode of Timer 2, 3 ..... 109
Figure 11-34 PWM Mode ..... 110
Figure 11-35 Example of PWM at 4MHz ..... 111
Figure 11-36 Example of Changing the Period in Absolute Duty Cycle at 4Mhz ..... 111
Figure 11-37 Timer4 16-bit Mode Block Diagram ..... 117
Figure 11-38 Buzzer Driver Block Diagram ..... 122
Figure 11-39 USART Block Diagram ..... 125
Figure 11-40 Clock Generation Block Diagram ..... 126
Figure 11-41 Synchronous Mode XCKn Timing ..... 127
Figure 11-42 frame format ..... 128
Figure 11-43 Start Bit Sampling ..... 132
Figure 11-44 Sampling of Data and Parity Bit ..... 132
Figure 11-45 Stop Bit Sampling and Next Start Bit Sampling ..... 132
Figure 11-46 SPI Clock Formats when UCPHA=0 ..... 134
Figure 11-47 SPI Clock Formats when UCPHA=1 ..... 135
Figure 11-48 SPI Block Diagram ..... 142
Figure 11-49 SPI Transmit/Receive Timing Diagram at CPHA = 0 ..... 144
Figure 11-50 SPI Transmit/Receive Timing Diagram at CPHA = 1 ..... 144
Figure 11-51 $I^2C$ Block Diagram ..... 147
Figure 11-52 Bit Transfer on the $I^2C$-Bus ..... 148
Figure 11-53 START and STOP Condition ..... 148
Figure 11-54 STOP or Repeated START Condition ..... 149
Figure 11-55 Acknowledge on the $I^2C$-Bus ..... 149

Figure 11-56 Clock Synchronization during Arbitration Procedure ..... 150
Figure 11-57 Arbitration Procedure of Two Masters ..... 150
Figure 11-58 Formats and States in the Master Transmitter Mode ..... 153
Figure 11-59 Formats and States in the Master Receiver Mode ..... 155
Figure 11-60 Formats and States in the Slave Transmitter Mode ..... 157
Figure 11-61 Formats and States in the Slave Receiver Mode ..... 159
Figure 11-62 ADC Block Diagram ..... 164
Figure 11-63 A/D Analog Input Pin Connecting Capacitor ..... 164
Figure 11-64 A/D Power(AVDD) Pin Connecting Capacitor ..... 165
Figure 11-65 ADC Operation for Align bit ..... 165
Figure 11-66 Converter Operation Flow ..... 166
Figure 11-67 Analog Comparator Block Diagram ..... 170
Figure 12-1 IDLE Mode Release Timing by External Interrupt ..... 174
Figure 12-2 IDLE Mode Release Timing by /RESET ..... 174
Figure 12-3 STOP Mode Release Timing by External Interrupt ..... 175
Figure 12-4 STOP Mode Release Timing by /RESET ..... 175
Figure 12-5 STOP1, 2 Mode Release Flow ..... 176
Figure 13-1 RESET Block Diagram ..... 178
Figure 13-2 Reset noise canceller time diagram ..... 179
Figure 13-3 Fast VDD rising time ..... 179
Figure 13-4 Internal RESET Release Timing On Power-Up ..... 180
Figure 13-5 Configuration timing when Power-on ..... 180
Figure 13-6 Boot Process Waveform ..... 181
Figure 13-7 Timing Diagram after RESET ..... 182
Figure 13-8 Oscillator generating waveform example ..... 182
Figure 13-9 Block Diagram of BOD ..... 183
Figure 13-10 Internal Reset at the power fail situation ..... 183
Figure 13-11 Configuration timing when BOD RESET ..... 184
Figure 14-1 Block Diagram of On-chip Debug System ..... 187
Figure 14-2 10-bit transmission packet ..... 188
Figure 14-3 Data transfer on the twin bus ..... 188
Figure 14-4 Bit transfer on the serial bus ..... 189
Figure 14-5 Start and stop condition ..... 189
Figure 14-6 Acknowledge on the serial bus ..... 189
Figure 14-7 Clock synchronization during wait procedure ..... 190
Figure 14-8 Connection of transmission ..... 190
Figure 15-1 Flash Memory Map ..... 196
Figure 15-2 Address configuration of Flash memory ..... 196
Figure 15-3 Data EEPROM memory map ..... 197
Figure 15-4 Address configuration of data EEPROM ..... 197
Figure 15-5 The sequence of page program and erase of Flash memory ..... 198
Figure 15-6 The sequence of bulk erase of Flash memory ..... 199
Figure 15-7 Pin diagram for parallel programming ..... 206
Figure 15-8 Parallel Byte Read Timing of Program Memory ..... 208
Figure 15-9 Parallel Byte Write Timing of Program Memory ..... 208

Figure 15-10 ISP mode ..... 209
Figure 15-11 Byte-parallel mode ..... 209

# List of Tables

Table 1-1 Ordering Information for Z51F0811 parts ..... 14
Table 5-1 Normal pin description ..... 25
Table 7-1 Absolute Maximum Rationgs ..... 28
Table 7-2 Recommended Operation Conditions ..... 28
Table 7-3 A/D Converter Characteristics ..... 29
Table 7-4 Analog Comparator Characteristics ..... 29
Table 7-5 Voltage Dropout Converter Characteristics ..... 30
Table 7-6 Power-On Reset Characteristics ..... 31
Table 7-7 Brown Out Detector Characteristics ..... 31
Table 7-8 Internal RC Oscillator Characteristics ..... 32
Table 7-9 Ring-Oscillator Characteristics ..... 32
Table 7-10 PLL Characteristics ..... 32
Table 7-11 DC Characteristics ..... 33
Table 7-12 AC Characteristics ..... 34
Table 7-13 SPI Characteristics ..... 35
Table 8-1 SFR Map Summary ..... 42
Table 9-1 Register Map ..... 46
Table 10-1 Interrupt Group Priority Level ..... 49
Table 10-2 Interrupt Vector Address Table ..... 52
Table 10-3 Register Map ..... 59
Table 11-1 Register Map ..... 66
Table 11-2 Register Map ..... 68
Table 11-3 Register Map ..... 70
Table 11-4 Register Map ..... 73
Table 11-5 Timer 0,1 operating modes ..... 76
Table 11-6 PWM Frequency vs. Resolution at 8 Mhz ..... 83
Table 11-7 Register Map ..... 93
Table 11-8 Operating Modes of Timer ..... 103
Table 11-9 PWM Frequency vs. Resolution at 8 Mhz ..... 109
Table 11-10 Register Map ..... 112
Table 11-11 Register Map ..... 118
Table 11-12 Buzzer Frequency at 8MHz ..... 122
Table 11-13 Register Map ..... 123
Table 11-14 Equations for Calculating Baud Rate Register Setting ..... 126
Table 11-15 CPOL Funtionality ..... 134
Table 11-16 Register Map ..... 136
Table 11-17 Examples of UBAUD Settings for Commonly Used Oscillator Frequencies ..... 141
Table 11-18 Register Map ..... 144
Table 11-19 Register Map ..... 171
Table 12-1 Peripheral Operation during Power Down Mode. ..... 173
Table 12-2 Register Map ..... 177
Table 13-1 Reset state ..... 178
Table 13-2 Boot Process Description ..... 181

Table 13-3 Register Map ..... 184
Table 15-1 Register Map ..... 191
Table 15-2 Program/erase Time ..... 195
Table 15-3 Operation Mode ..... 205
Table 15-4 The selection of memory type by ADDRH[7:4] ..... 206
Table 15-5 Parallel mode instruction format ..... 207
Table 15-6 Security policy using lock-bits ..... 210

# Z51F0811

## CMOS SINGLE-CHIP 8-BIT MICROCONTROLLER WITH 12-BIT A/D CONVERTER

## 1. Overview

### 1.1 Description

The Z51F0811 is advanced CMOS 8-bit microcontroller with 8K bytes of Flash. This is powerful microcontroller which provides a highly flexible and cost effective solution to many embedded control applications. This provides the following features: 8K bytes of Flash, 256 bytes of internal SRAM, 256 bytes of external SRAM, 512 bytes of Data EEPROM, general purpose I/O, 8/16-bit timer/counter, watchdog timer, watch timer, SPI, USART, I2C, on-chip POR, BOD, 12-bit A/D converter, analog comparator, buzzer driving port, 10-bit high speed PWM output, on-chip oscillator and clock circuitry. The Z51F0811 also supports power saving modes to reduce power consumption.

| Device Name | Flash | EEPROM | RAM | ADC | I/O Port | Package |
|---|---|---|---|---|---|---|
| Z51F0811 | 8K bytes | 512 bytes | 256 bytes | 15ch | 30 | 32-pin QFN |
| | | | | 12ch | 26 | 28-pin TSSOP |
| | | | | 9ch | 18 | 20-pin TSSOP |
| | | | | 8ch | 14 | 16-pin TSSOP |

## 1.2 Features

- **CPU**
  - 8 Bit CISC Core (8051 Compatible,2 clock per cycle)
- **8K Bytes On-chip Flash**
  - Optional boot code section with protection
  - Endurance : 10,000 times at room temp
  - Retention : 10 years
- **256 Bytes SRAM**
- **256 Bytes XRAM**
- **512 Bytes Data EEPROM**
  - Endurance : 300,000 times at room temp
  - Retention : 10 years
- **General Purpose I/O**
  - 30 Ports (P0[7:0], P1[6:0], P2[6:0], P3[7:0]): 32 Pin
  - 26 Ports (P0[7:0], P1[6:0], P2[2:0], P3[7:0]): 28 Pin
  - 18 Ports (P0[7:0], P1[6:0], P2[2:0]): 20 Pin
  - 14 Ports (P0[7:0], P1[2:0], P2[2:0]): 16 Pin
- **One Basic Interval Timer**
- **Timer/ Counter**
  - 8Bit×4ch(16Bit×2ch) + 16Bitx1ch
- **3 High Frequency 10-bit PWM (Using Timer1)**
- **10-bit PWM (Using Timer3)**
- **Watch Dog Timer**
- **Watch Timer**
- **SPI**
- **USART (2ch)**
- **I2C**
- **Buzzer Driving Port**
- **12 Bit A/D Converter**
  - 15 Input channels
- **Analog Comparator**
  - On Chip Analog Comparator
- **Interrupt Sources**
  - External (8)
  - Pin Change Interrupt(P0) (1)
  - USART0,1 (4)
  - SPI (1)
  - Timer (5)
  - I2C (1)
  - Data EEPROM (1)
  - ADC (1)
  - Analog Comparator(1)
  - WDT (1)
  - WT (1)
  - BIT (1)
- **On-Chip RC-Oscillator**
  - 8MHz(±3%)
- **Power On Reset**
- **Programmable Brown-Out Detector**
- **Minimum Instruction Execution Time**
  - 200ns (@10MHz, NOP Instruction)
- **Power down mode**
  - IDLE, STOP1, STOP2 mode
- **Sub-Active mode**
  - System used external 32.768KHz crystal or system used internal 125KHz Ring oscillator
- **Operating Frequency**
  - 1MHz ~ 12MHz
- **Operating Voltage**
  - 1.8V ~ 5.5V
- **Operating Temperature : -40 ~ +85℃**
- **Package Type**
  - 32-pin QFN
  - 28-pin TSSOP
  - 20-pin TSSOP
  - 16-pin TSSOP
  - Pb free package

## 1.3 Ordering Information

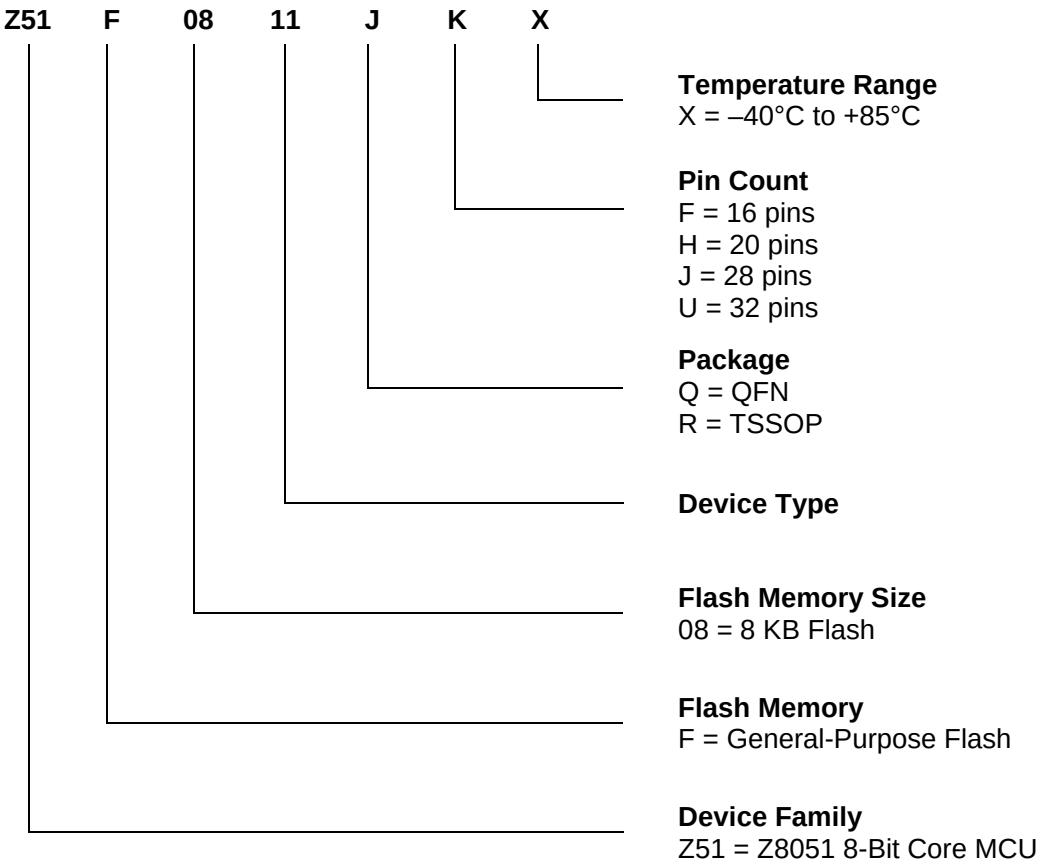
Table 1-1 Ordering Information for Z51F0811 parts

| Device name | ROM size | RAM size | EEPROM size | Package |
|---|---|---|---|---|
| Z51F0811QUX | 8K bytes Flash | 256 bytes | 512 bytes | 32 QFN |
| Z51F0811RJX | | | | 28 TSSOP |
| Z51F0811RHX | | | | 20 TSSOP |
| Z51F0811RFX | | | | 16 TSSOP |

### 1.3.1 Part Number Suffix Designation

Zilog part numbers consist of a number of components, as indicated in the following example.

**Example:** Part number Z51F0811RFX is an 8-bit MCU with 8 KB of Flash memory and 512 bytes of RAM in a 16-pin TSSOP package and operating within a –40°C to +85°C temperature range. In accordance with RoHS standards, this device has been built using lead-free solder.



## 1.4 Development Tools

### 1.4.1 Compiler

We do not provide the compiler. Please contact third parties.

The Z51F0811 core is Mentor 8051. Anyway, device ROM size is smaller than 64KB. Developer can use all kinds of third party's standard 8051 compiler.

### 1.4.2 OCD emulator and debugger

The OCD (On Chip Debug) emulator supports Zilogs 8051 series MCU emulation.

The OCD interface uses two wires interfacing between PC and MCU which is attached to user's system. The OCD can read or change the value of MCU internal memory and I/O peripherals. And also the OCD controls MCU internal debugging logic, it means OCD controls emulation, step run, monitoring, etc.

The OCD Debugger program works on Microsoft-Windows NT, 2000, XP, Vista (32bit) operating system.

If you want to see more details, please refer OCD debugger manual. You can download debugger S/W and manual from our web-site.

Connection:

- SCLK (Z51F0811 P06 port)
- SDATA (Z51F0811 P07 port)

### 1.4.3 Programmer

Single programmer:

PGMplus USB: It programs MCU device directly.

Figure 1-1 Single Programmer

OCD emulator: It can write code in MCU device too.

Because of, OCD debugging supports ISP (In System Programming).

It does not require additional H/W, except developer's target system.

Note) If you produce semiconductor and measure the stop current, use OCD ISP

Gang programmer:

It programs 8 MCU devices at once. So, it is mainly used in mass production line.

Gang programmer is standalone type, it means it does not require host PC.

## 2. Block Diagram

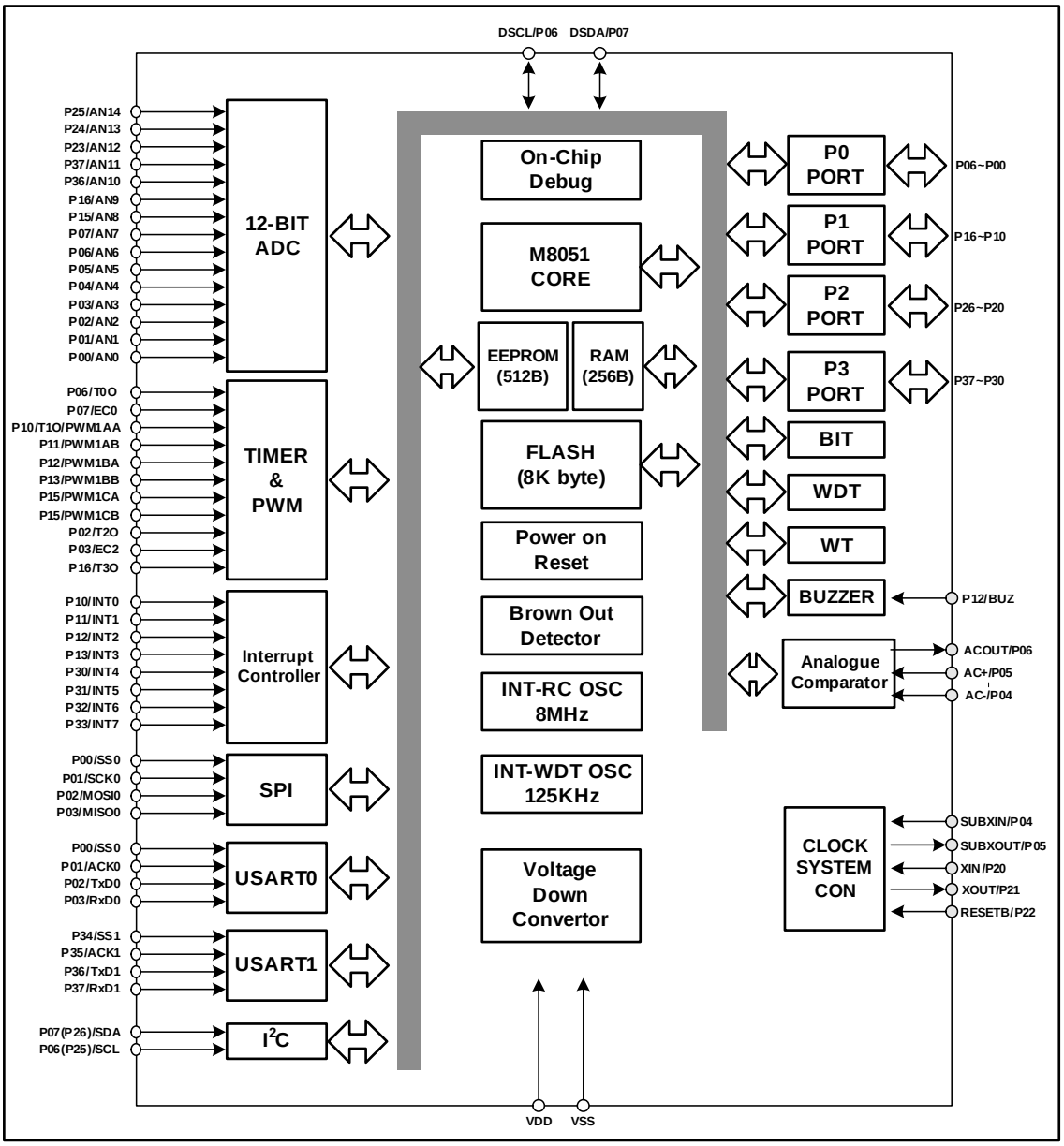



Figure 2-1 Z51F0811 block diagram

## 3. Pin Assignment

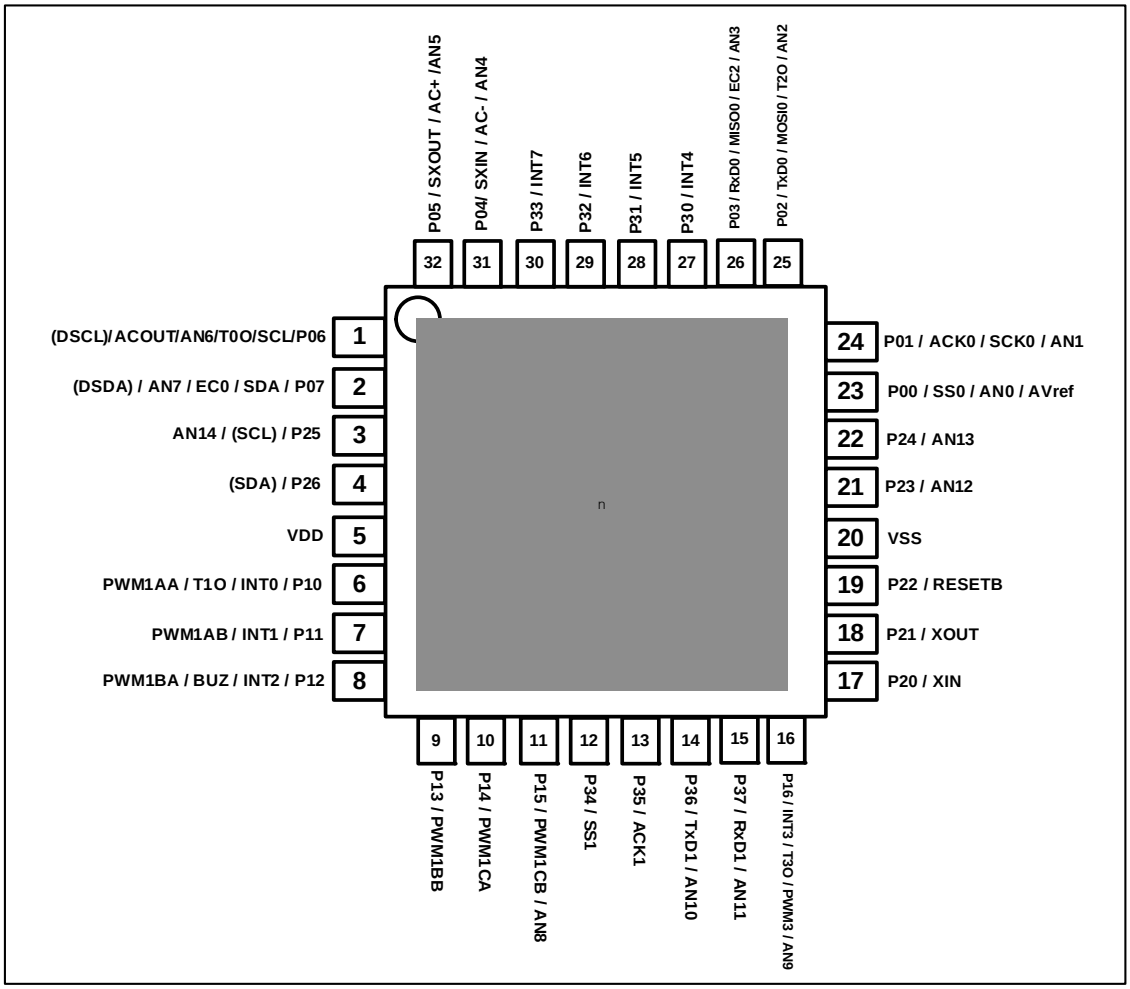



Figure 3-1 Z51F0811 32 QFN Pin assignment

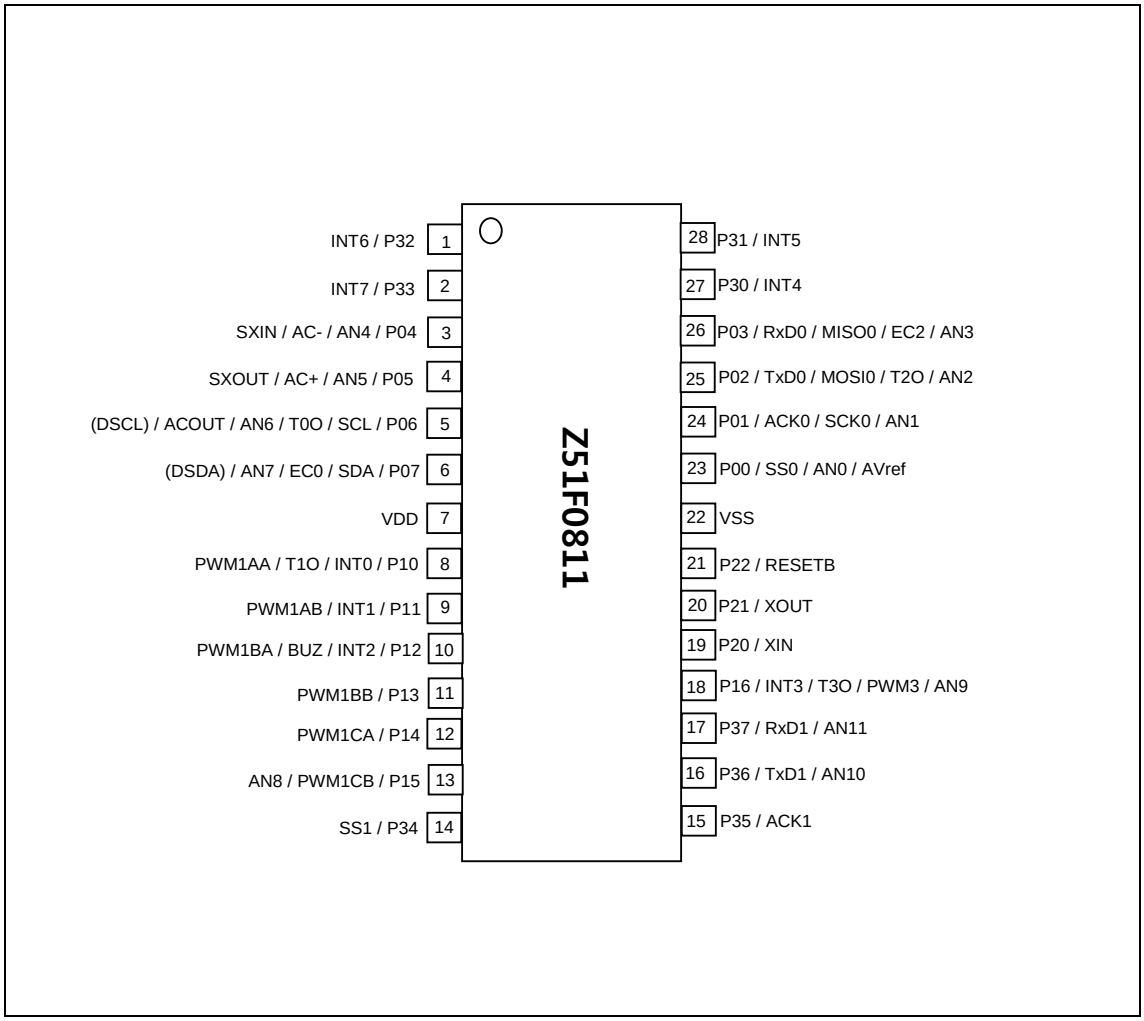



Figure 3-2 Z51F0811 28 TSSOP Pin assignment

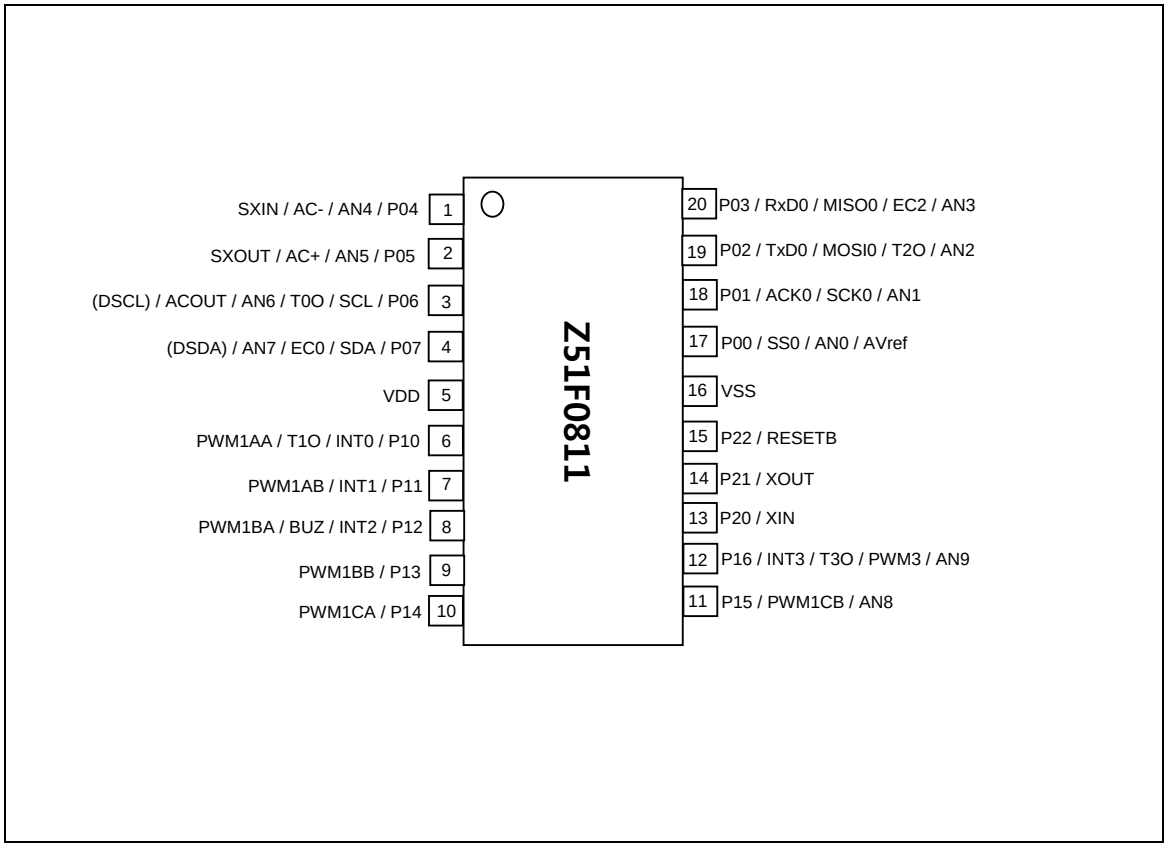



Figure 3-3 Z51F0811 20 TSSOP Pin assignment

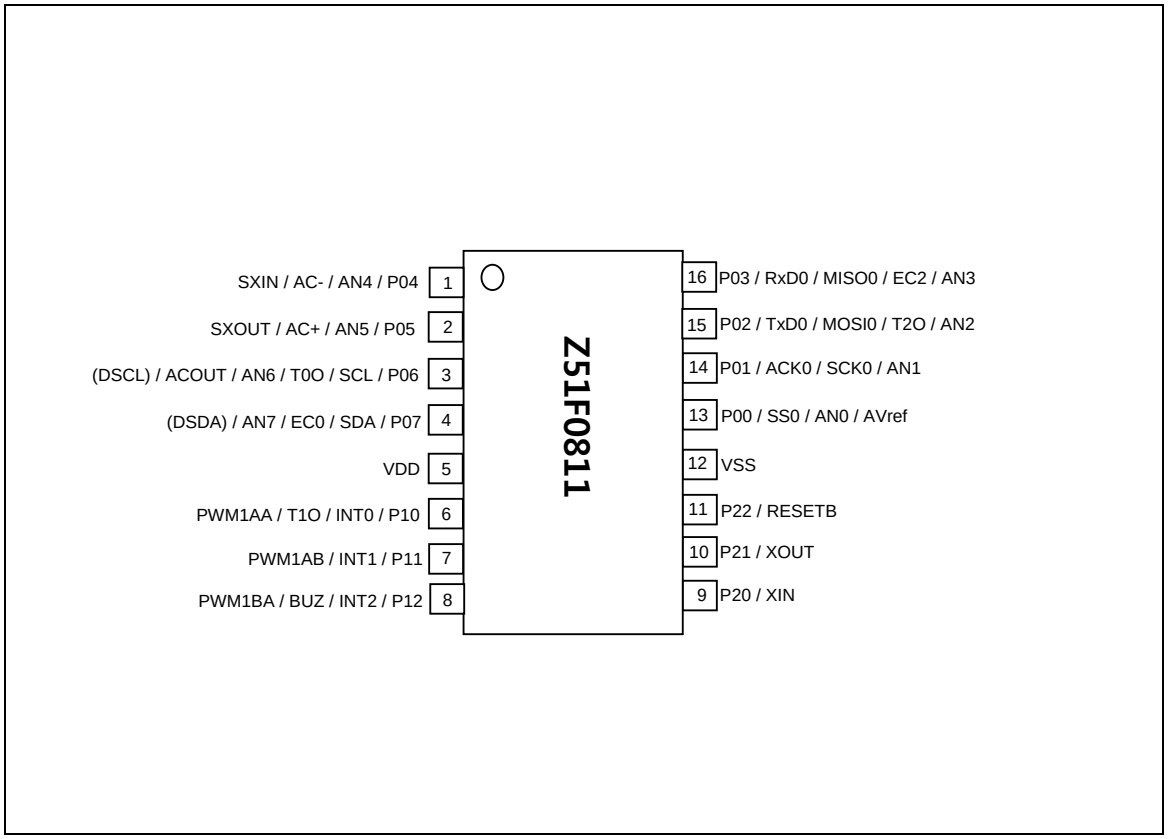



Figure 3-4 Z51F0811 16 TSSOP Pin assignment

## 4. Package Diagram

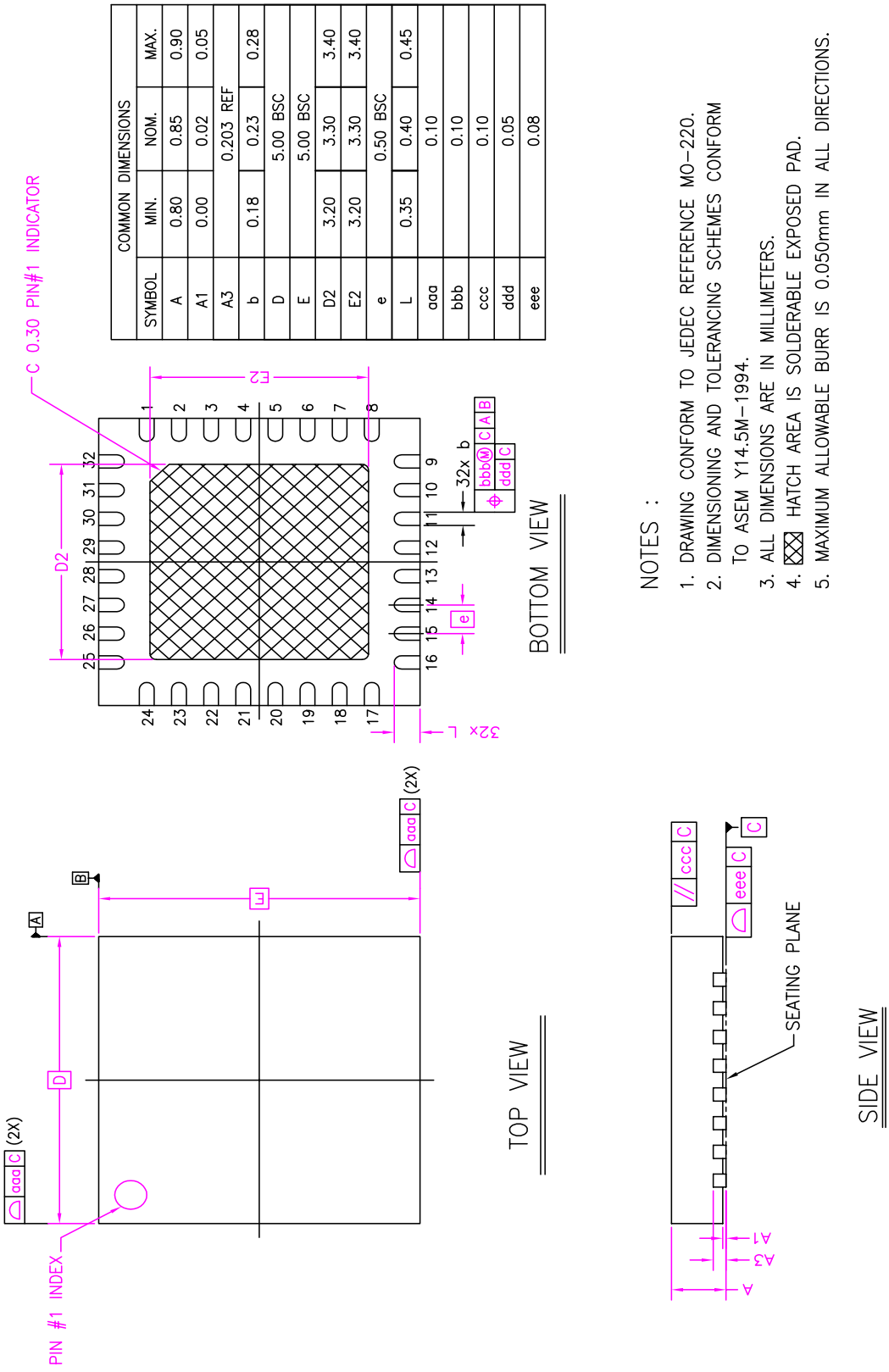
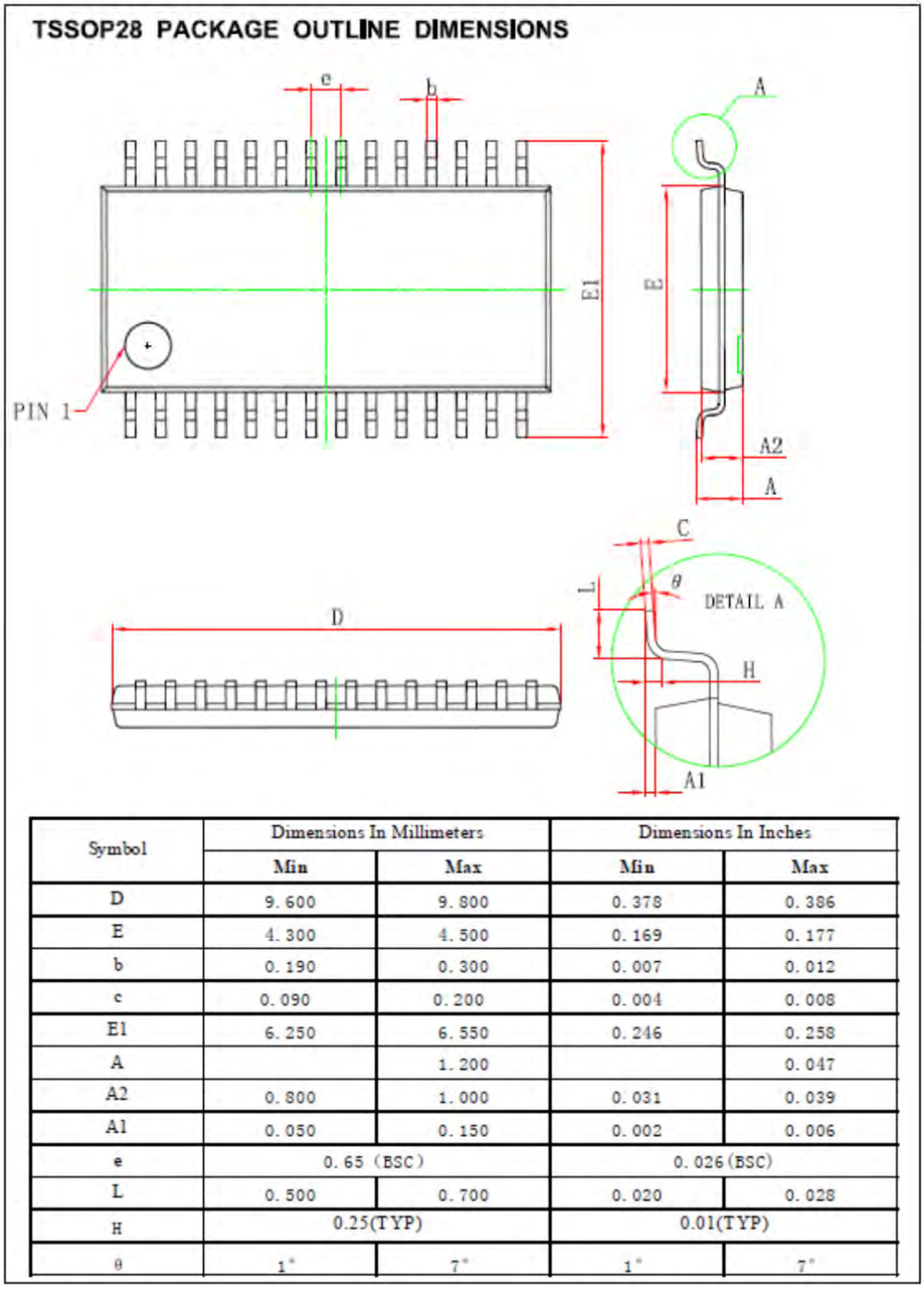
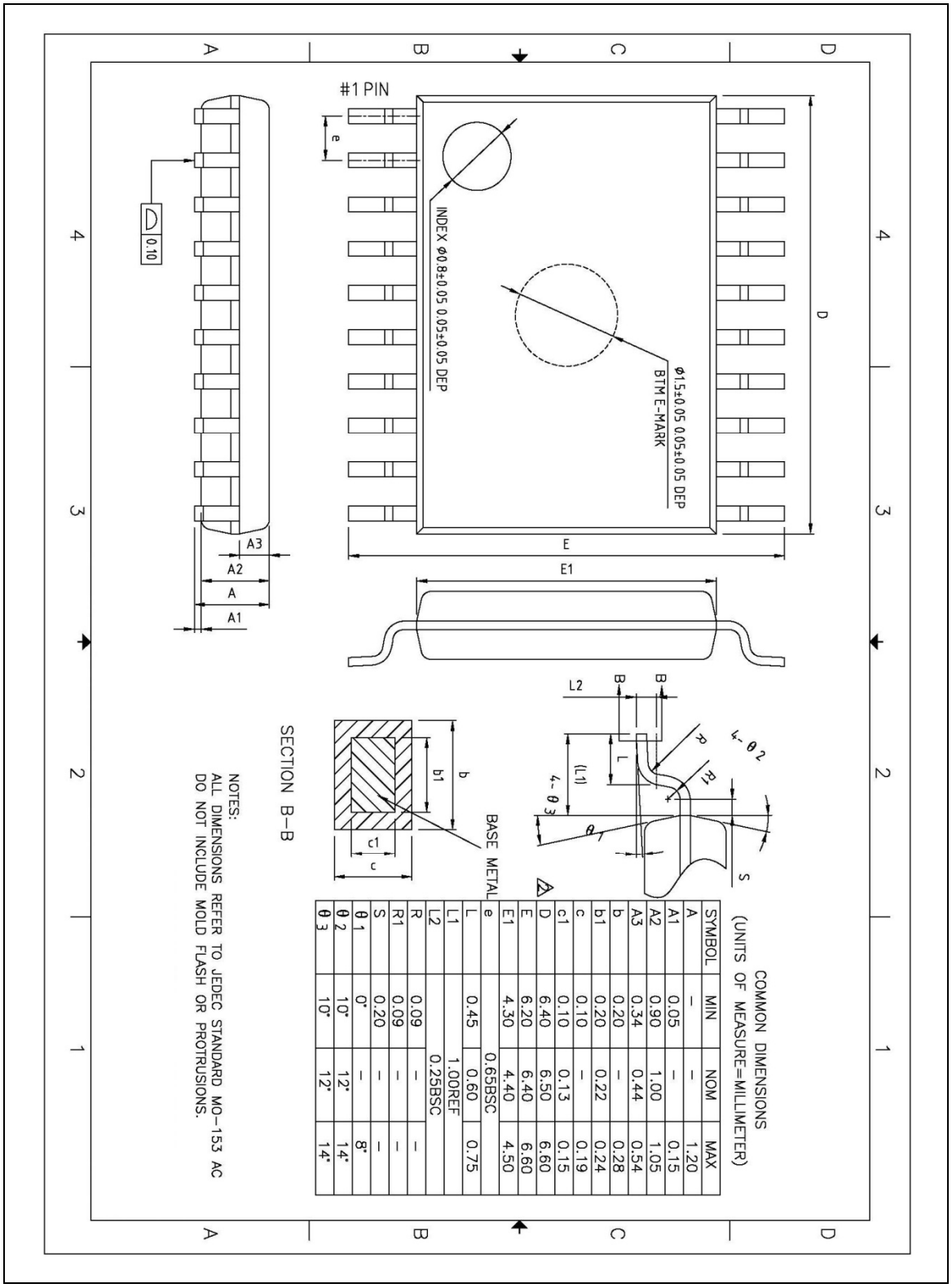



Figure 4-1 32 pin QFN package




| Symbol | Dimensions In Millimeters | | Dimensions In Inches | |
|---|---|---|---|---|
| | Min | Max | Min | Max |
| D | 9.600 | 9.800 | 0.378 | 0.386 |
| E | 4.300 | 4.500 | 0.169 | 0.177 |
| b | 0.190 | 0.300 | 0.007 | 0.012 |
| c | 0.090 | 0.200 | 0.004 | 0.008 |
| E1 | 6.250 | 6.550 | 0.246 | 0.258 |
| A | | 1.200 | | 0.047 |
| A2 | 0.800 | 1.000 | 0.031 | 0.039 |
| A1 | 0.050 | 0.150 | 0.002 | 0.006 |
| e | 0.65 (BSC) | | 0.026(BSC) | |
| L | 0.500 | 0.700 | 0.020 | 0.028 |
| H | 0.25(TYP) | | 0.01(TYP) | |
| θ | 1° | 7° | 1° | 7° |

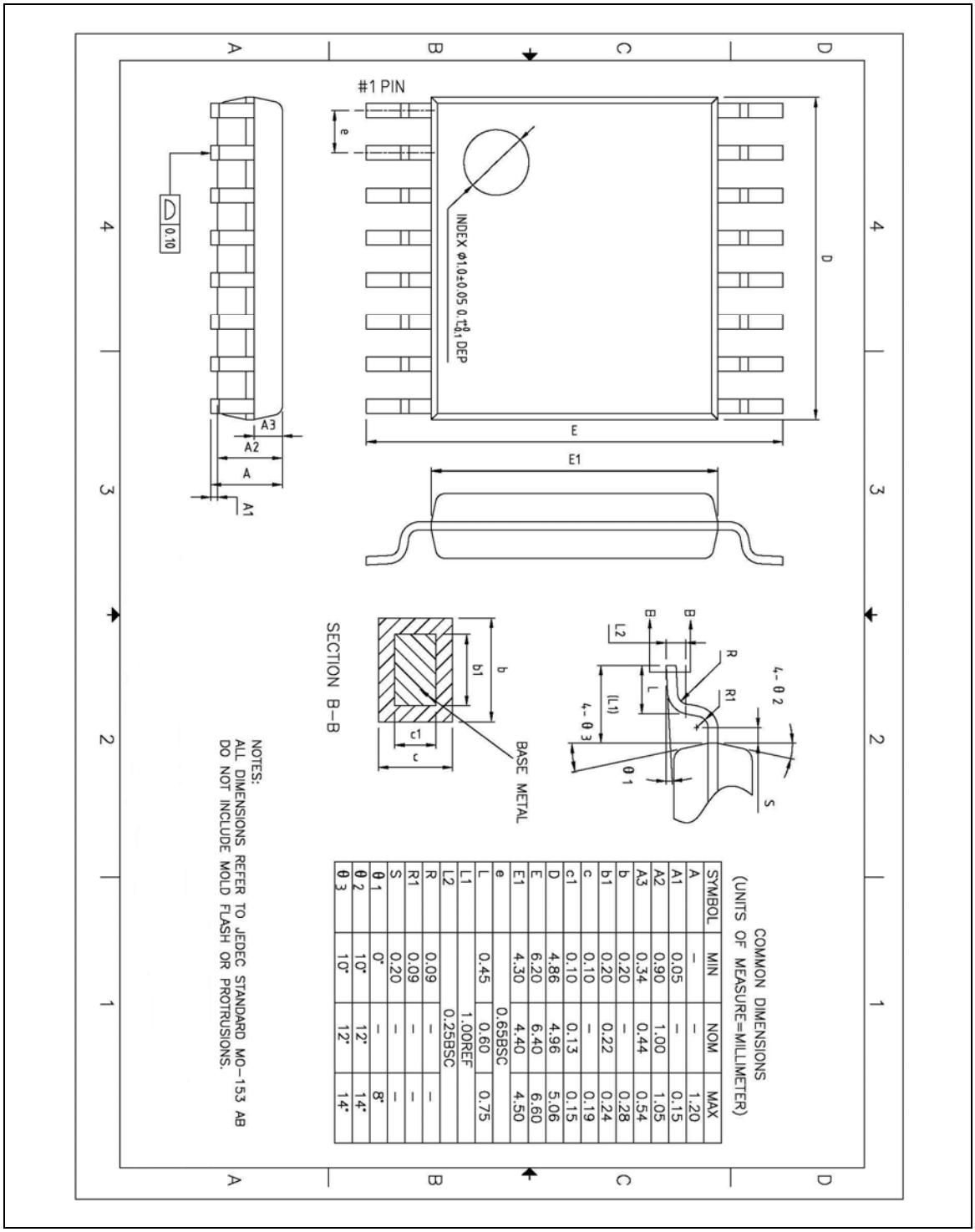
Figure 4-2 28 pin TSSOP package




COMMON DIMENSIONS
(UNITS OF MEASURE=MILLIMETER)

| SYMBOL | MIN | NOM | MAX |
|---|---|---|---|
| A | – | – | 1.20 |
| A1 | 0.05 | – | 0.15 |
| A2 | 0.90 | 1.00 | 1.05 |
| A3 | 0.34 | 0.44 | 0.54 |
| b | 0.20 | – | 0.28 |
| b1 | 0.20 | 0.22 | 0.24 |
| c | 0.10 | – | 0.19 |
| c1 | 0.10 | 0.13 | 0.15 |
| D | 6.40 | 6.50 | 6.60 |
| E | 6.20 | 6.40 | 6.60 |
| E1 | 4.30 | 4.40 | 4.50 |
| e | | 0.65BSC | |
| L | 0.45 | 0.60 | 0.75 |
| L1 | | 1.00REF | |
| L2 | | 0.25BSC | |
| R | 0.09 | – | – |
| R1 | 0.09 | – | – |
| S | 0.20 | – | – |
| θ1 | 0° | – | 8° |
| θ2 | 10° | 12° | 14° |
| θ3 | 10° | 12° | 14° |

Figure 4-3 20 pin TSSOP package




COMMON DIMENSIONS
(UNITS OF MEASURE=MILLIMETER)

| SYMBOL | MIN | NOM | MAX |
|---|---|---|---|
| A | – | – | 1.20 |
| A1 | 0.05 | – | 0.15 |
| A2 | 0.90 | 1.00 | 1.05 |
| A3 | 0.34 | 0.44 | 0.54 |
| b | 0.20 | – | 0.28 |
| b1 | 0.20 | 0.22 | 0.24 |
| c | 0.10 | – | 0.19 |
| c1 | 0.10 | 0.13 | 0.15 |
| D | 4.86 | 4.96 | 5.06 |
| E | 6.20 | 6.40 | 6.60 |
| E1 | 4.30 | 4.40 | 4.50 |
| e | | 0.65BSC | |
| L | 0.45 | 0.60 | 0.75 |
| L1 | | 1.00REF | |
| L2 | | 0.25BSC | |
| R | 0.09 | – | – |
| R1 | 0.09 | – | – |
| S | 0.20 | – | – |
| θ 1 | 0° | – | 8° |
| θ 2 | 10° | 12° | 14° |
| θ 3 | 10° | 12° | 14° |

Figure 4-4 16 pin TSSOP package

## 5. Pin Description

Table 5-1 Normal pin description

| PIN Name | I/O | Function | @RESET | Shared with |
|---|---|---|---|---|
| P00 | I/O | Port P0<br>8-Bit I/O Port<br>Can be set in input or output mode in 1-bit units<br>Internal pull-up register can be used via software when this port is used as input port<br>Open Drain enable register can be used via software when this port is used as output port<br>AN0~AN7 can be selected by ADCM register | Input | Avref / AN0 / SS0 |
| P01 | | | | AN1 / SCK0 / ACK0 |
| P02 | | | | AN2 / T2O / MOSI0 / TxD0 |
| P03 | | | | AN3 / EC2 / MISO0 / RxD0 |
| P04 | | | | SXIN / AC- / AN4 |
| P05 | | | | SXOUT / AC+ / AN5 |
| P06 | | | | DSCL / ACOUT/ AN6/ T0O / SCL |
| P07 | | | | DSDA / AN7/ EC0 / SDA |
| P10 | I/O | Port P1<br>7-Bit I/O Port<br>Can be set in input or output mode in 1-bit units<br>Internal pull-up register can be used via software when this port is used as input port<br>Open Drain enable register can be used via software when this port is used as output port<br>AN8, AN9 can be selected by ADCM register | Input | PWM1AA / T1O / INT0 |
| P11 | | | | PWM1AB / INT1 |
| P12 | | | | PWM1BA / BUZ / INT2 |
| P13 | | | | PWM1BB |
| P14 | | | | PWM1CA |
| P15 | | | | AN8 / PWM1CB |
| P16 | | | | AN9 / PWM3 / T3O / INT3 |
| P20 | I/O | Port P2<br>7-Bit I/O Port<br>Can be set in input or output mode in 1-bit units<br>Internal pull-up register can be used via software when this port is used as input port<br>Open Drain enable register can be used via software when this port is used as output port<br>AN12, AN13, AN14 can be selected by ADCM register | Input | XIN |
| P21 | | | | XOUT |
| P22 | | | | RESETB |
| P23 | | | | AN12 |
| P24 | | | | AN13 |
| P25 | | | | AN14 / (SCL) |
| P26 | | | | (SDA) |
| P30 | I/O | Port P3<br>8-Bit I/O Port<br>Can be set in input or output mode in 1-bit units<br>Internal pull-up register can be used via software when this port is used as input port<br>Open Drain enable register can be used via software when this port is used as output port<br>AN10, AN11 can be selected by ADCM register | Input | INT4 |
| P31 | | | | INT5 |
| P32 | | | | INT6 |
| P33 | | | | INT7 |
| P34 | | | | SS1 |
| P35 | | | | ACK1 |
| P36 | | | | AN10 / TxD1 |
| P37 | | | | AN11 / RxD1 |

# 6. Port Structures

## 6.1 General Purpose I/O Port

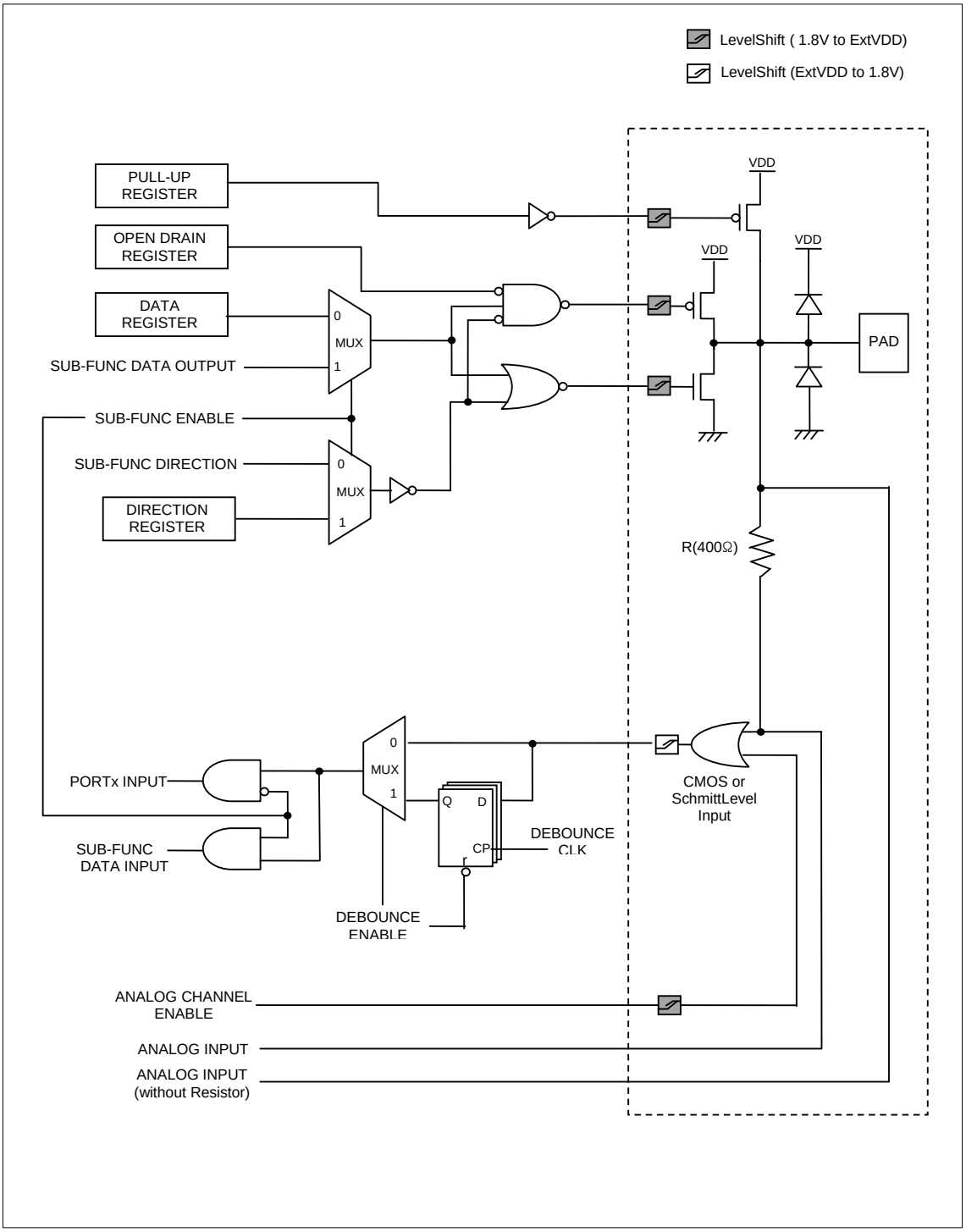



Figure 6-1 General Purpose I/O Port

## 6.2 External Interrupt I/O Port

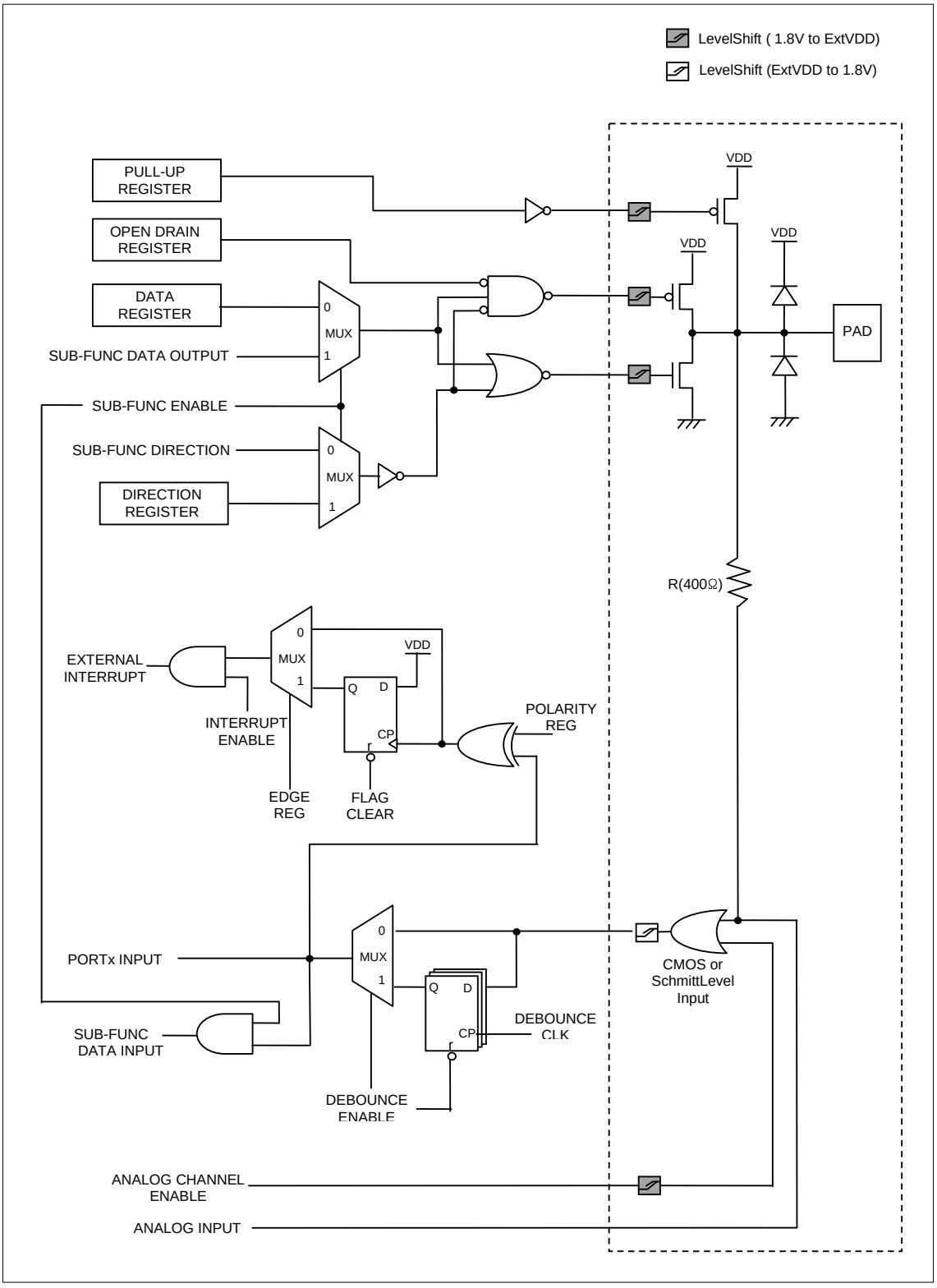



Figure 6-2 External Interrupt I/O Port

# 7. Electrical Characteristics

## 7.1 Absolute Maximum Ratings

Table 7-1 Absolute Maximum Rationgs

| Parameter | Symbol | Rating | Unit |
|---|---|---|---|
| Supply Voltage | VDD | -0.3~+6.5 | V |
| | VSS | -0.3~+0.3 | V |
| Normal Voltage Pin | VI | -0.3~VDD+0.3 | V |
| | VO | -0.3~VDD+0.3 | V |
| | IOH | 10 | mA |
| | ∑IOH | 80 | mA |
| | IOL | 20 | mA |
| | ∑IOL | 160 | mA |
| Total Power Dissipation | PT | 600 | mW |
| Storage Temperature | TSTG | -45~+125 | ℃ |

Note) Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## 7.2 Recommended Operating Conditions

Table 7-2 Recommended Operation Conditions

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Supply Voltage | VDD | fXIN=1~12MHz | 4.5 | - | 5.5 | V |
| | | fXIN=1~8MHz | 1.8 | | | |
| | | fSUB=32.768KHz | | | | |
| Operating Temperature | TOPR | VDD=1.8~5.5V | -40 | - | 85 | ℃ |
| Operating Frequency | FOPR | fXIN | 1 | - | 12 | MHz |
| | | fSUB | - | 32.768 | - | KHz |
| | | Internal RC-OSC | 7.76 | | 8.24 | MHz |
| | | Internal Ring-OSC | - | 1 | | MHz |

## 7.3 A/D Converter Characteristics

Table 7-3 A/D Converter Characteristics (TA=-40℃ ~ +85℃, VDD=AVDD=2.7V ~ 5.5V, VSS=0V)

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Resolution | | - | - | 12 | - | bits |
| Total Accuracy | | AVDD=VDD=5.12V fXIN=4MHz | | - | ±3 | lsb |
| Integral Linear Error | INL | | - | - | ±2 | lsb |
| Differential Linearity Error | DLE | | - | - | ±2 | lsb |
| Zero Offset Error | ZOE | | - | | ±3 | lsb |
| Full Scale Error | FSE | | - | | ±3 | lsb |
| Conversion Time | tCON | 12bit conversion max 3MHz | - | 60 | - | cycle |
| Analog Input Voltage | VAN | - | VSS | - | AVDD=VDD | V |
| Analog Power Voltage | AVDD | - | - | *AVDD=VDD | - | V |
| Analog Reference Voltage | AVREF | - | 2.7 | - | 5.5 | V |
| Analog Ground Voltage | AVSS | - | - | VSS | - | V |
| Analog Input Leakage Current | | AVDD=VDD=5.12V | - | - | 10 | uA |
| ADC Operating Current | IDD | AVDD=VDD=5.12V | - | 1 | 3 | mA |
| | SIDD | | - | - | 1 | uA |

## 7.4 Analog Comparator Characteristics

Table 7-4 Analog Comparator Characteristics

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Input Leakage Current | $I_L$ | VDDEXT=5V, Vin=1/2VDDEXT | -50 | - | 50 | nA |
| Input Offset Voltage | $V_{offset}$ | VDDEXT=5V, Vin=1/2VDD | 10 | - | 40 | ±mV |
| Operating Current | IOP | COMP_EN=H | - | 1 | - | mA |
| Power Down Current | IPD | COMP_EN=L | - | 1 | - | uA |
| Response Time | $V_{RT}$ | CL= 50pF, VDDEXT=5V | - | - | 500 | nS |

## 7.5 Voltage Dropout Converter Characteristics

Table 7-5 Voltage Dropout Converter Characteristics

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Operating Voltage | | - | 1.8 | - | 5.5 | V |
| Operating Temperature | | - | -40 | - | +85 | ℃ |
| Regulation Voltage | | - | 1.62 | 1.8 | 1.98 | V |
| Drop-out Voltage | | - | - | - | 0.02 | V |
| Current Drivability | | RUN/IDLE | - | 20 | - | mA |
| | | SUB-ACTIVE | - | 1 | - | mA |
| | | STOP1 | - | 50 | - | uA |
| | | STOP2 | - | 10 | - | uA |
| Operating Current | IDD1 | RUN/IDLE | - | - | 1 | mA |
| | IDD2 | SUB-ACTIVE | - | - | 0.1 | mA |
| | SIDD1 | STOP1 | - | - | 5 | uA |
| | SIDD2 | STOP2 | - | - | 0.1 | uA |
| Drivability Transition Time | TRAN1 | SUB to RUN | - | - | 1 | uS |
| | TRAN2 | STOP to RUN | - | - | 200 | uS |

Note) -STOP1: WDT running - STOP2: WDT disable

## 7.6 Power-On Reset Characteristics

Table 7-6 Power-On Reset Characteristics

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Operating Voltage | | - | VSS | - | 5.5 | V |
| Operating Temperature | | - | -40 | - | +85 | ℃ |
| RESET Release Level | | - | 1.3 | 1.4 | 1.5 | V |
| Operating Current | IDD | - | - | - | 10 | uA |
| | SIDD | - | - | - | 1 | uA |

## 7.7 Brown Out Detector Characteristics

Table 7-7 Brown Out Detector Characteristics

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Operating Voltage | | - | VSS | - | 5.5 | V |
| Operating Temperature | | - | -40 | - | +85 | ℃ |
| Detection Level | 4.2V | - | 4.0 | - | 4.4 | V |
| | 3.6V | - | 3.4 | - | 3.8 | V |
| | 2.5V | - | 2.3 | - | 2.7 | V |
| | 1.6V | - | 1.4 | - | 1.8 | V |
| Hysteresis | | - | - | - | - | mV |
| Operating Current | IDD | - | - | - | 50 | uA |
| | SIDD | - | - | - | 1 | uA |

## 7.8 Internal RC Oscillator Characteristics

Table 7-8 Internal RC Oscillator Characteristics

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Operating Voltage | | - | 1.8 | - | 5.5 | V |
| Operating Temperature | | - | -40 | - | +85 | ℃ |
| Frequency | | - | 7.76 | | 8.24 | MHz |
| Stabilization Time | | - | - | - | 10 | mS |
| Operating Current | IDD | - | - | - | - | uA |
| | SIDD | - | - | - | 1 | uA |

## 7.9 Ring-Oscillator Characteristics

Table 7-9 Ring-Oscillator Characteristics

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Operating Voltage | | - | 1.8 | - | 5.5 | V |
| Operating Temperature | | - | -40 | - | +85 | ℃ |
| Frequency | | - | - | 1 | - | MHz |
| Stabilization Time | | - | - | - | - | mS |
| Operating Current | IDD | - | - | - | - | uA |
| | SIDD | - | - | - | 1 | uA |

## 7.10 PLL Characteristics

Table 7-10 PLL Characteristics

| Parameter | Symbol | Condition | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| PLL current | $I_{PLL}$ | - | - | 1.5 | TBD | mA |
| Input clock frequency | fxin | - | 2 | - | 16 | MHz |
| Output clock frequency | fout | - | 6.25 | - | 128 | MHz |
| Output clock duty | - | - | 40 | - | 60 | % |
| Setting time | $t_D$ | - | - | 1 | - | mS |
| Accuracy | - | - | - | 2 | - | % |

( TA = 0℃ ~ +70℃, VDD18 = 1.6V ~ 2.0V, VSS = 0V )

## 7.11 DC Characteristics

Table 7-11 DC Characteristics

| Parameter | Symbol | Condition | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Input Low Voltage | VIL1 | P2[2] | -0.5 | - | 0.2VDD | V |
| | VIL2 | All others PAD | -0.5 | - | 0.2VDD | V |
| Input High Voltage | VIH1 | P2[2] | 0.8VDD | - | VDD | V |
| | VIH2 | All others PAD | 0.7VDD | - | VDD | V |
| Output Low Voltage | VOL1 | ALL I/O (IOL=20mA, VDD=4.5V) | - | - | 1 | V |
| Output High Voltage | VOH1 | ALL I/O (IOH=-8.57mA, VDD=4.5V) | 3.5 | - | - | V |
| Input High Leakage Current | IIH | ALL PAD | - | - | 1 | uA |
| Input Low Leakage Current | IIL | ALL PAD | -1 | - | - | uA |
| Pull-Up Resister | RPU | ALL PAD | 20 | - | 50 | kΩ |
| Power Supply Current | IDD1 | Run Mode, fXIN=12MHz @5V | - | *2.6 | 10 | mA |
| | IDD2 | Sleep Mode, fXIN=12MHz @5V | - | *1.5 | 5 | mA |
| | IDD3 | Sub Active Mode, fSUBXIN=32.768KHz @5V | - | *71 | 500 | uA |
| | IDD4 | STOP1 Mode, WDT Active @5V (BOD enable) | - | *45 | 200 | uA |
| | IDD5 | STOP1 Mode, WDT Active @5V (BOD disable) | - | *20 | 100 | uA |
| | IDD6 | STOP2 Mode, WDT Disable @5V (BOD enable) | - | *27 | 100 | uA |
| | IDD7 | STOP2 Mode, WDT Disable @5V (BOD disable) | - | *1 | 7 (room temp) | uA |

(VDD =2.7~5.5V, VSS =0V, fXIN=10.0MHz, TA=-40~+85℃)

Note) - STOP1: WDT running, STOP2: WDT disable.

- (*) typical test condition : VDD=5V, Internal RC-OSC=8MHz, ROOM TEMP, all PORT output LOW, Timer0 Active, 1PORT toggling.

## 7.12 AC Characteristics

Table 7-12 AC Characteristics (VDD=5.0V±10%, VSS=0V, TA=-40~+85℃)

| Parameter | Symbol | PIN | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Operating Frequency | fMCP | XIN | 1 | - | 10 | MHz |
| System Clock Cycle Time | tSYS | - | 100 | - | 1000 | ns |
| Oscillation Stabilization Time (8MHz) | tMST1 | XIN, XOUT | - | - | 10 | ms |
| External Clock “H” or “L” Pulse Width | tCPW | XIN | 90 | - | - | ns |
| External Clock Transition Time | tRCP,tFCP | XIN | - | - | 10 | ns |
| External Interrupt Input Width | tIW | INT0~INTx | 2 | - | - | tSYS |
| External Interrupt Transition Time | tFI,tRI | INT0~INTx | | | 1 | us |
| nRESET Input Pulse “L” Width | tRST | nRESET | 8 | - | - | tSYS |
| External Counter Input “H” or “L” Pulse Width | tECW | EC0~ECx | 2 | - | - | tSYS |
| Event Counter Transition Time | tREC,tFEC | EC0~ECx | - | - | 20 | ns |

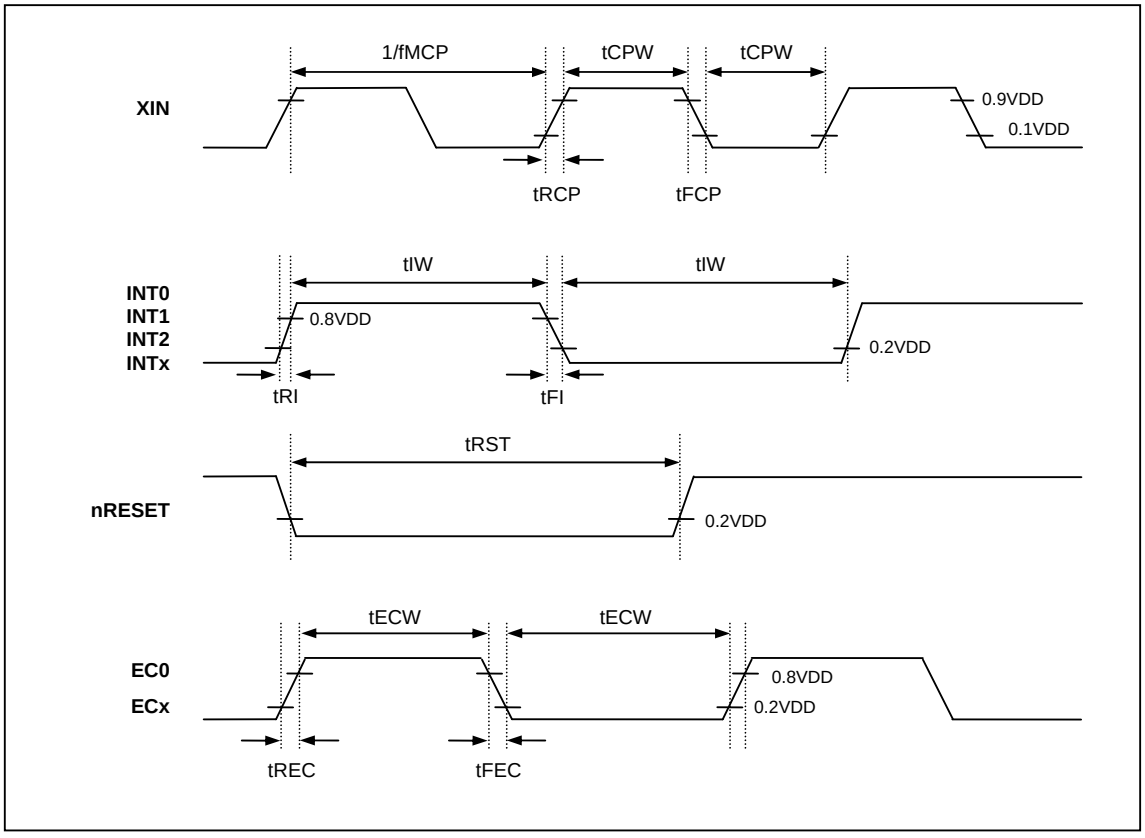



Figure 7-1 AC Timing

## 7.13 SPI Characteristics

Table 7-13 SPI Characteristics

| Parameter | Symbol | PIN | MIN | TYP | MAX | Unit |
|---|---|---|---|---|---|---|
| Output Clock Pulse Period | tSCK | SCK | - | SPI clock mode | - | ns |
| Input Clock Pulse Period | tSCK | SCK | 2• tSYS | - | - | ns |
| Input Clock “H” or “L” Pulse Width | tSCKL, tSCKH | SCK | | 50% duty | - | ns |
| Input Clock Pulse Transition Time | tFSCK,tRSCK | SCK | - | - | 30 | ns |
| Output Clock “H” or “L” Pulse Width | tSCKL, tSCKH | SCK | tSYS-30 | - | - | ns |
| Output Clock Pulse Transition Time | tFSCK,tRSCK | SCK | - | - | 30 | ns |
| First Output Clock Delays Time | tFOD | OUTPUT | | | | |
| Output Clock Delay Time | tDS | OUTPUT | - | - | 100 | ns |
| Input Pulse Transition Time | tFSIN,tRSIN | INPUT | - | - | 30 | ns |
| Input Setup Time | tDIS | INPUT | 100 | | - | ns |
| Input Hold Time | tDIH | INPUT | tSYS+70 | - | - | ns |

(VDD=5.0V±10%, VSS=0V, TA=-40~+85℃)

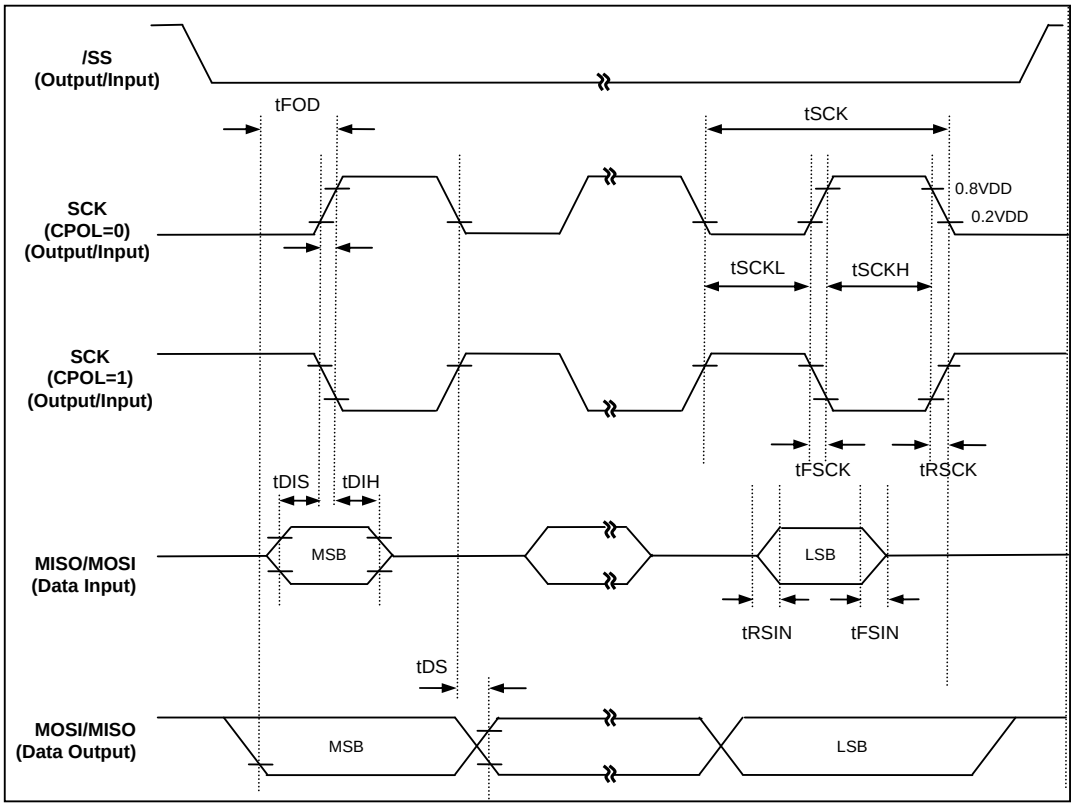



Figure 7-2 SPI Timing

## 7.14 Typical Characteristics

These graphs and tables provided in this section are for design guidance only and are not tested or guaranteed. In some graphs or tables the data presented are outside specified operating range (e.g. outside specified VDD range). This is for information only and devices are guaranteed to operate properly only within the specified range.

The data presented in this section is a statistical summary of data collected on units from different lots over a period of time. “Typical” represents the mean of the distribution while “max” or “min” represents (mean + 3σ) and (mean - 3σ) respectively where σ is standard deviation.

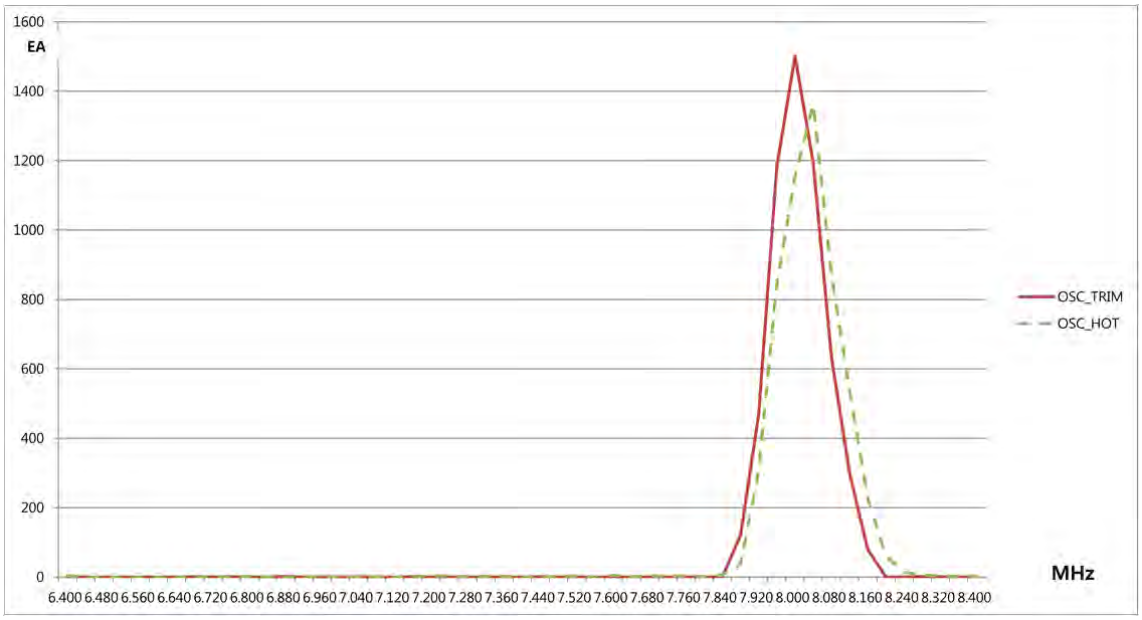



Figure 7-3 8MHz Internal OSC Freq.(OSC_HOT: 85℃)

# 8. Memory

The Z51F0811 addresses two separate address memory stores: Program memory and Data memory. The logical separation of Program and Data memory allows Data memory to be assessed by 8-bit addresses, which can be more quickly stored and manipulated by 8-bit CPU. Nevertheless, 16-bit Data memory addresses can also be generated through the DPTR register.

Program memory can only be read, not written to. There can be up to 64K bytes of Program memory. In the Z51F0811 Flash version of these devices the 8K bytes of Program memory are provided on-chip. Data memory can be read and written to up to 256 bytes internal memory (DATA) including the stack area.

## 8.1 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has just 8K bytes program memory space.

Figure 8-1 shows a map of the lower part of the program memory. After reset, the CPU begins execution from location 0000H. Each interrupt is assigned a fixed location in program memory. The interrupt causes the CPU to jump to that location, where it commences execution of the service routine. External interrupt 0, for example, is assigned to location 0003H. If external interrupt 0 is going to be used, its service routine must begin at location 0003H. If the interrupt is not going to be used, its service location is available as general purpose program memory. If an interrupt service routine is short enough (as is often the case in control applications), it can reside entirely within that 8 byte interval. Longer service routines can use a jump instruction to skip over subsequent interrupt locations, if other interrupts are in use.

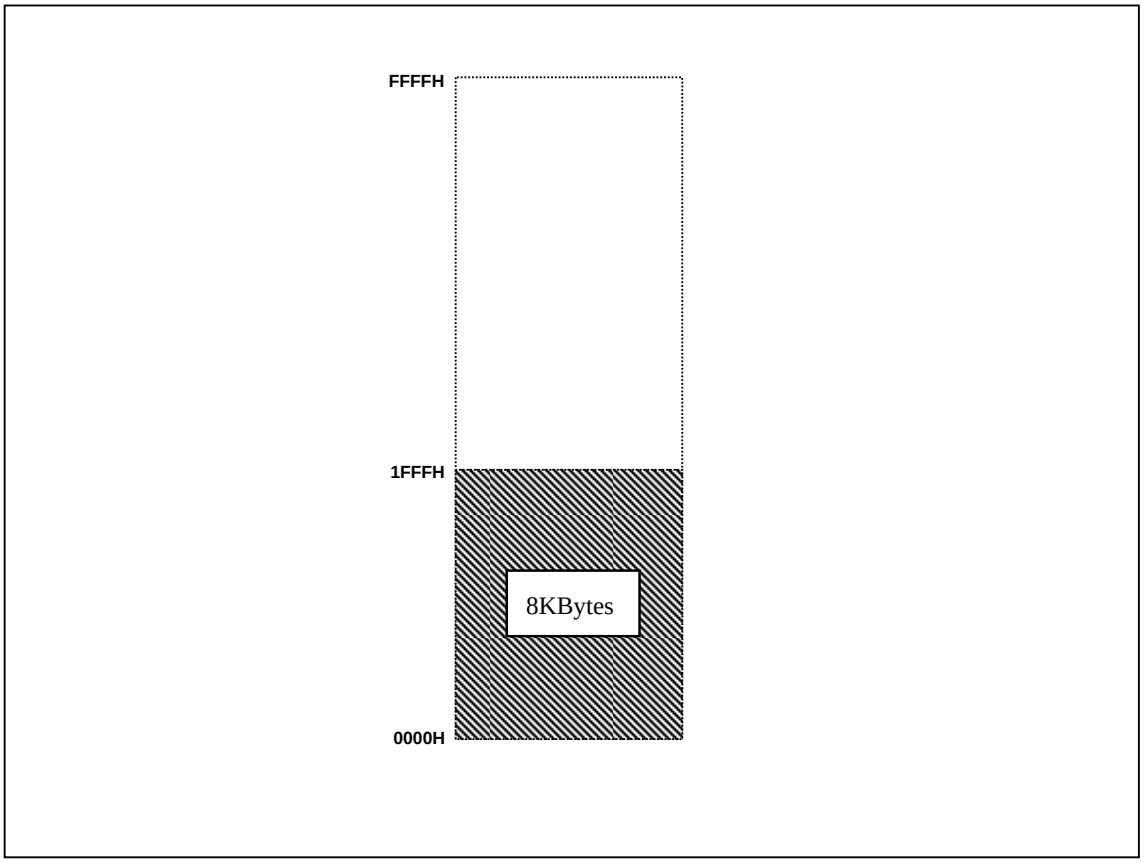



Figure 8-1 Program memory

- User Function Mode: 8KBytes Included Interrupt Vector Region
- Non-volatile and reprogramming memory: Flash memory based on EEPROM cell

## 8.2 Data Memory

Figure 8-2 shows the internal Data memory space available.

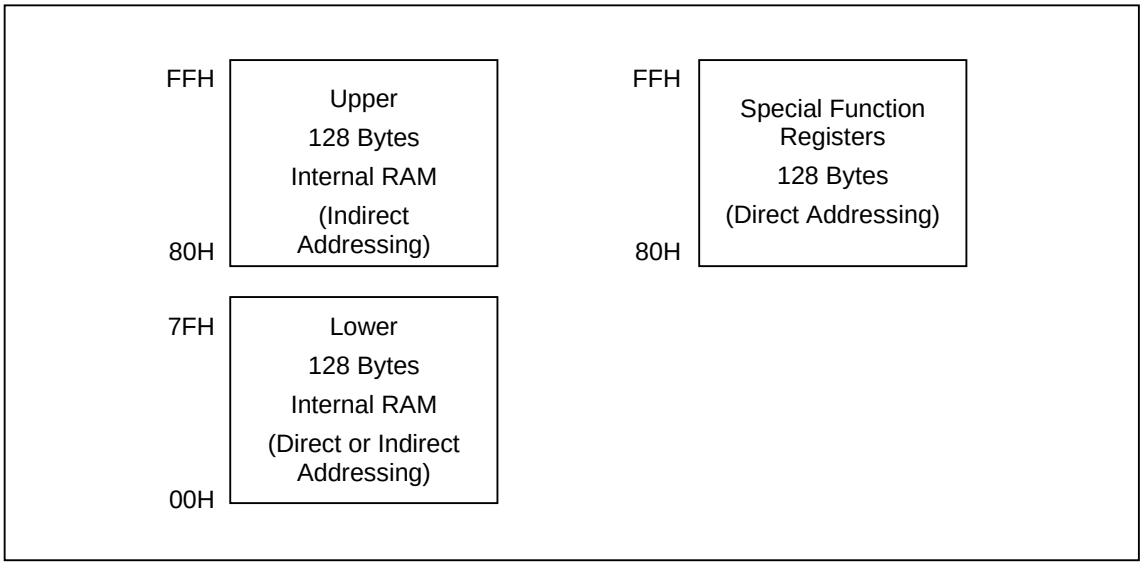



Figure 8-2 Data memory map

The internal memory space is divided into three blocks, which are generally referred to as the lower 128, upper 128, and SFR space.

Internal Data memory addresses are always one byte wide, which implies an address space of only 256 bytes. However, the addressing modes for internal RAM can in fact accommodate 384 bytes, using a simple trick. Direct addresses higher than 7FH access one memory space and indirect addresses higher than 7FH access a different memory space. Thus Figure 8-2 shows the upper 128 and SFR space occupying the same block of addresses, 80H through FFH, although they are physically separate entities.

The lower 128 bytes of RAM are present in all 8051 devices as mapped in Figure 8-3. The lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word select which register bank is in use. This allows more efficient used of code space, since register instructions are shorter than instructions that use direct addressing.

The next 16 bytes above the register banks form a block of bit-addressable memory space. The 8051 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00H through 7FH.

All of the bytes in the lower 128 can be accessed by either direct or indirect addressing. The upper 128 bytes RAM can only be accessed by indirect addressing. These spaces are used for user RAM and stack pointer.

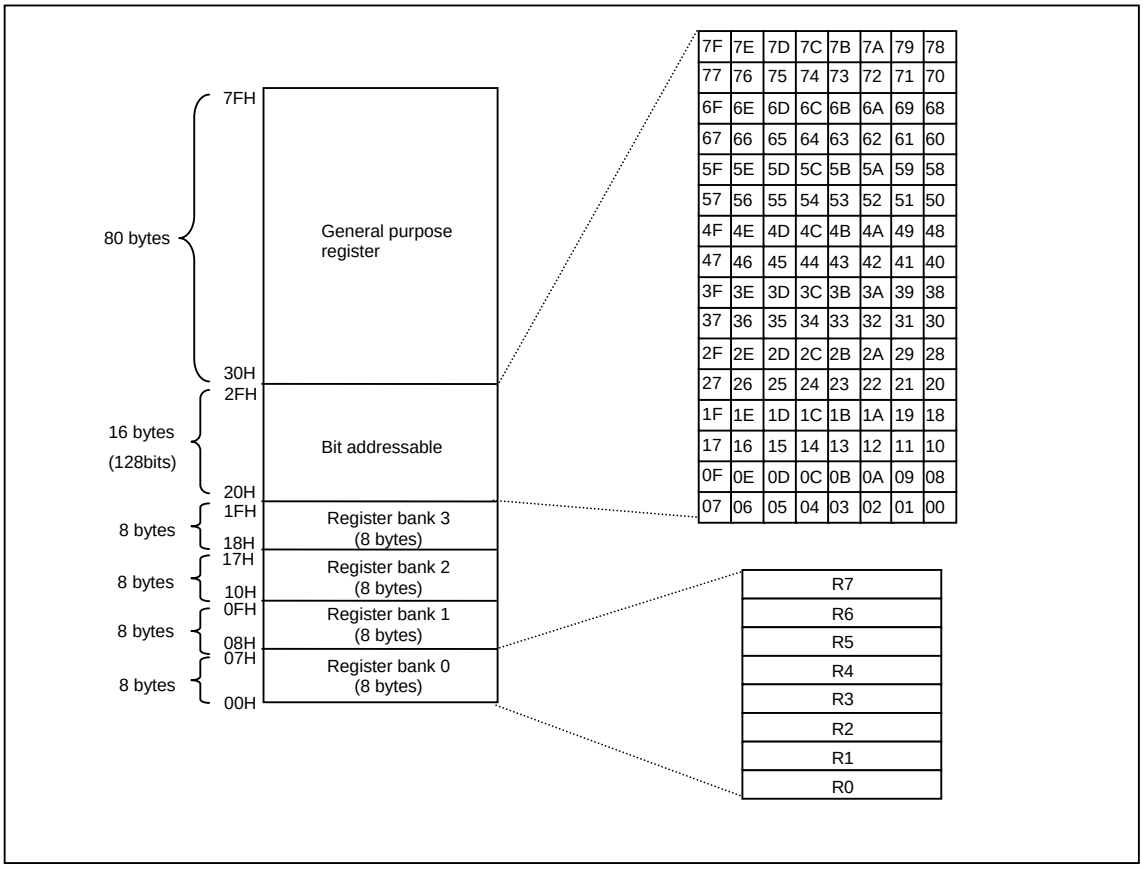



Figure 8-3 Lower 128 bytes RAM

## 8.3 EEPROM Data Memory

Z51F0811 has 512 bytes EEPROM Data memory. This area has no relation with RAM/Flash. It can read and write through SFR with 8-bit unit.

For more information about EEPROM Data memory, see chapter 15.

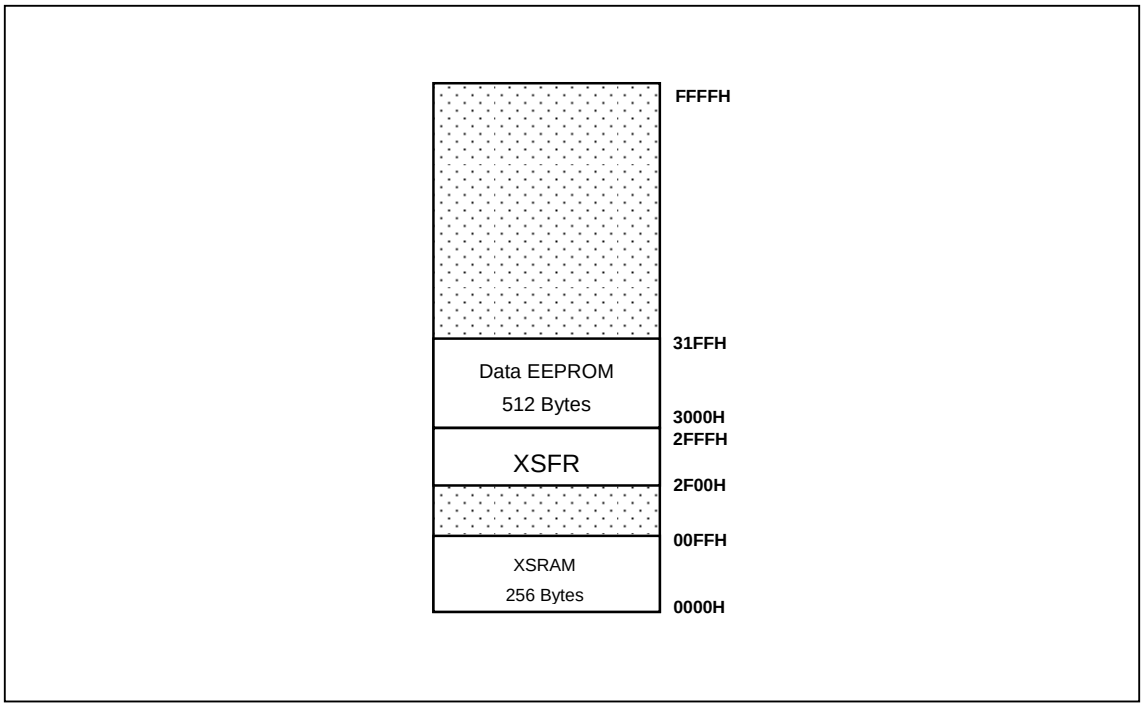



Figure 8-4 XDATA memory area

## 8.4 SFR Map

### 8.4.1 SFR Map Summary

Table 8-1 SFR Map Summary

| | 0H/8H | 1H/9H | 2H/AH | 3H/BH | 4H/CH | 5H/DH | 6H/EH | 7H/FH |
|---|---|---|---|---|---|---|---|---|
| 2F58H | - | FUSE_PKG | FUSE_CAL2 | FUSE_CAL1 | FUSE_CAL0 | FUSE_CONF | TEST_REGB | TEST_REGA |
| 2F50H | PSR0 | PSR1 | - | - | - | - | - | - |
| 2F48H | - | - | - | - | - | - | - | - |
| 2F40H | - | - | - | - | - | - | - | - |
| 2F38H | - | - | - | - | - | - | - | - |
| 2F30H | - | - | - | - | - | - | - | - |
| 2F28H | - | - | - | - | - | - | - | - |
| 2F20H | - | - | - | - | - | - | - | - |
| 2F18H | P0DB | P1DB | P2DB | P3DB | - | - | - | - |
| 2F10H | - | - | - | - | - | - | - | - |
| 2F08H | - | - | - | - | P0OD | P1OD | P2OD | P3OD |
| 2F00H | P0PU | P1PU | P2PU | P3PU | - | - | - | - |
| | | | | | | | | |
| F8H | IP1 | ACCSR | UCTRL11 | UCTRL12 | UCTRL13 | USTAT1 | UBAUD1 | UDATA1 |
| F0H | B | - | FEARL | FEARM | FEARH | FEDR | FETR | - |
| E8H | - | - | FEMR | FECR | FESR | FETCR | - | - |
| E0H | ACC | - | UCTRL1 | UCTRL2 | UCTRL3 | USTAT | UBAUD | UDATA |
| D8H | - | - | I2CMR | I2CSR | I2CSCLLR | I2CSCLHR | I2CSDHR | I2CDR |
| D0H | PSW | - | SPICR | SPIDR | SPISR | T4H | I2CSAR1 | I2CSAR |
| C8H | - | - | T3CR | T3DR / PWM3PR | T3 / PWM3DR / CDR3 | PWM3HR | T4CR | T4L |
| C0H | - | - | T2DLYB | T1DLYC | T1ISR | T1IMSK | T2CR | T2 / T2DR / CDR2 |
| B8H | IP | - | T1BDR | T1CDR | T1PHR | T1PCR2 | T1PCR3 | T1DLYA |
| B0H | - | - | T0CR | T0 / T0DR / CDR0 | T1CR | T1DR /T1PPR | T1 / T1ADR / CDR1 | T1PCR |
| A8H | IE | IE1 | IE2 | IE3 | IE4 | IE5 | PCI0 | TMISR |
| A0H | - | P3IO | EO | EIENAB | EIFLAG | EIEDGE | EIPOLA | EIBOTH |
| 98H | P3 | P2IO | ADCM | ADCM2 / ADCRH | ADCRL | WTMR | WTR / WTCR | BUZCR |
| 90H | P2 | P1IO | - | - | - | - | - | - |
| 88H | P1 | P0IO | SCCR | BCCR | BITR | WDTMR | WDTR / WDTCR | BUZDR |
| 80H | P0 | SP | DPL | DPH | - | PLLCR | BODR | PCON |

Note: 1) The registers of which lower 3-bit address are 000 are bit-addressable (except for XSFR)

### 8.4.2 Compiler Compatible SFR

**ACC (Accumulator) : E0H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ACC | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**ACC** Accumulator

**B (B Register) : F0H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| B | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**B** B Register

**SP (Stack Pointer) : 81H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SP | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 07H

**SP** Stack Pointer

**DPL (Data Pointer Low Byte) : 82H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DPL | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**DPL** Data Pointer Low Byte

**DPH (Data Pointer High Byte) : 83H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DPH | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**DPH** Data Pointer High Byte

### PSW (Program Status Word) : D0H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CY | AC | F0 | RS1 | RS0 | OV | F1 | P |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

| | |
|---|---|
| **CY** | Carry Flag |
| **AC** | Auxiliary Carry Flag |
| **F0** | General Purpose User-Definable Flag |
| **RS1** | Register Bank Select bit 1 |
| **RS0** | Register Bank Select bit 0 |
| **OV** | Overflow Flag |
| **F1** | User-Definable Flag |
| **P** | Parity Flag. Set/cleared by hardware each instruction cycle to indicate an odd/even number of '1' bits in the accumulator |

### EO (Extended Operation Register) : A2H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | TRAP_EN | - | DPSEL.2 | DPSEL.1 | DPSEL.0 |
| R | R | R | R/W | R | R/W | R/W | R/W |

Initial value : 00H

**TRAP_EN** Select the instruction

0 Select MOVC @(DPTR++), A

1 Select Software TRAP instruction

**DPSEL[2:0]** Select Banked Data Point Register

| DPSEL2 | DPSEL1 | DPSEL0 | |
|---|---|---|---|
| 0 | 0 | 0 | DPTR0 |
| 0 | 0 | 1 | DPTR1 |
| 0 | 1 | 0 | DPTR2 |
| 0 | 1 | 1 | DPTR3 |
| 1 | 0 | 0 | DPTR4 |
| 1 | 0 | 1 | DPTR5 |
| 1 | 1 | 0 | DPTR6 |
| 1 | 1 | 1 | DPTR7 |

# 9. I/O Ports

## 9.1 I/O Ports

The Z51F0811 has four I/O ports (P0, P1, P2 and P3). Each port can be easily configured by software as I/O pin, internal pull up and open drain pin to meet various system configurations and design requirements. Also P0 includes function that can generate interrupt according to change of state of the pin.

## 9.2 Port Register

### 9.2.1 Data Register (Px)

Data Register is a bidirectional I/O port. If ports are configured as output ports, data can be written to the corresponding bit of the Px. If ports are configured as input ports, the data can be read from the corresponding bit of the Px.

### 9.2.2 Direction Register (PxIO)

Each I/O pin can independently used as an input or an output through the PxIO register. Bits cleared in this read/write register will select the corresponding pin in Px to become an input, setting a bit sets the pin to output. All bits are cleared by a system reset.

### 9.2.3 Pull-up Resistor Selection Register (PxPU)

The on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up resistor selection register (PxPU). The pull-up register selection controls the pull-up resister enable/disable of each port. When the corresponding bit is 1, the pull-up resister of the pin is enabled. When 0, the pull-up resister is disabled. All bits are cleared by a system reset. (Only port pull-up resistor selection have default ON state for unused pins in 32-pin package for 16, 20, 28-pin package).

### 9.2.4 Open-drain Selection Register (PxOD)

There is internally open-drain selection register (PxOD) in P0, P1, P2 and P3. The open-drain selection register controls the open-drain enable/disable of each port. Ports become push-pull by a system reset.

### 9.2.5 Debounce Enable Register (PxDB)

P0, P1, P2 and P3 support debounce function. Debounce time of each ports has about 5us, but if P2[2] uses external reset function, it has about 7us debounce time. (except P2[2], other port initialization state is OFF)

### 9.2.6 Pin Change Interrupt Enable Register (PCI0)

The P0 can support Pin Change Interrupt function. Pin Change Interrupts PCI will trigger if any enabled P0[7:0] pin toggles. The PCI0 Register control which pins contribute to the pin change interrupts.

### 9.2.7 Port Selection Register (PSRx)

PSRx registers prevent the input leakage current when ports are connected to analog inputs. If the bit of PSRx is '1', the dynamic current path of the schmitt OR gate of the port is cut off and the digital input of the corresponding port is always '1'.

### 9.2.8 Register Map

Table 9-1 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| P0 | 80H | R/W | 00H | P0 Data Register |
| P0IO | 89H | R/W | 00H | P0 Direction Register |
| P0PU | 2F00H | R/W | 00H | P0 Pull-up Resistor Selection Register |
| P0OD | 2F0CH | R/W | 00H | P0 Open-drain Selection Register |
| P0DB | 2F18H | R/W | 00H | P0 Debounce Enable Register |
| PCI0 | AEH | R/W | 00H | P0 Pin Change Interrupt Enable Register |
| P1 | 88H | R/W | 00H | P1 Data Register |
| P1IO | 91H | R/W | 00H | P1 Direction Register |
| P1PU | 2F01H | R/W | 00H | P1 Pull-up Resistor Selection Register |
| P1OD | 2F0DH | R/W | 00H | P1 Open-drain Selection Register |
| P1DB | 2F19H | R/W | 00H | P1 Debounce Enable Register |
| P2 | 90H | R/W | 00H | P2 Data Register |
| P2IO | 99H | R/W | 00H | P2 Direction Register |
| P2PU | 2F02H | R/W | 00H | P2 Pull-up Resistor Selection Register |
| P2OD | 2F0EH | R/W | 00H | P2 Open-drain Selection Register |
| P2DB | 2F1AH | R/W | 00H | P2 Debounce Enable Register |
| P3 | 98H | R/W | 00H | P3 Data Register |
| P3IO | A1H | R/W | 00H | P3 Direction Register |
| P3PU | 2F03H | R/W | 00H | P3 Pull-up Resistor Selection Register |
| P3OD | 2F0FH | R/W | 00H | P3 Open-drain Selection Register |
| P3DB | 2F1BH | R/W | 00H | P3 Debounce Enable Register |
| PSR0 | 2F50H | R/W | 00H | Port Selection Register 0 |
| PSR1 | 2F51H | R/W | 00H | Port Selection Register 1,2,3 |

## 9.3 Px Port

### 9.3.1 Px Port Description

Px is 8-bit I/O port. Px control registers consist of Data register (Px), direction register (PxIO), debounce enable register (PxDB), pull-up register selection register (PxPU), open-drain selection register (PxOD), pin change interrupt register (PCI0)

### 9.3.2 Register description for Px

**Px (Px Data Register) : 80H, 88H, 90H, 98H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px7 | Px6 | Px5 | Px4 | Px3 | Px2 | Px1 | Px0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**Px[7:0]** I/O Data

### PxIO (Px Direction Register) : 89H, 91H, 99H, A1H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px7IO | Px6IO | Px5IO | Px4IO | Px3IO | Px2IO | Px1IO | Px0IO |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**PxIO[7:0]** Px data I/O direction.

0 Input

1 Output

### PxPU (Px Pull-up Resistor Selection Register) : 2F00H ~ 2F03H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px7PU | Px6PU | Px5PU | Px4PU | Px3PU | Px2PU | Px1PU | Px0PU |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**PxPU[7:0]** Configure pull-up resistor of Px port

0 Disable

1 Enable

### PxOD (Px Open-drain Selection Register) : 2F0CH ~ 2F0FH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px7OD | Px6OD | Px5OD | Px4OD | Px3OD | Px2OD | Px1OD | Px0OD |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**PxOD[7:0]** Configure open-drain of Px port

0 Disable

1 Enable

### PxDB (Px Debounce Enable Register) : 2F18H ~ 2F1BH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| Px7DB | Px6DB | Px5DB | Px4DB | Px3DB | Px2DB | Px1DB | Px0DB |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**PxDB[7:0]** Configure debounce of Px port

0 Disable

1 Enable

### PCI0 (P0 Pin Change Interrupt Enable Register) : AEH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PCI07 | PCI06 | PCI05 | PCI04 | PCI03 | PCI02 | PCI01 | PCI00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**PCI0[7:0]** Configure Pin Change Interrupt of P0 port

0 Disable

1 Enable

**PSR0 (P0 Port Selection Register) : 2F50H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PSR07 | PSR06 | PSR05 | PSR04 | PSR03 | PSR02 | PSR01 | PSR00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**PSR0[7:0]** P07~P00 port selection register

0 Disable analog channel AN[7:0] (default)

1 Enable analog channel AN[7:0]

**PSR1 (Port Selection Register 1, 2, 3) : 2F51H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PSR17 | PSR16 | PSR15 | PSR14 | PSR13 | PSR12 | PSR11 | PSR10 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**PSR1[7]** I2C ports selection register

0 P0[7:6] for I2C (default)

1 P2[6:5] for I2C

**PSR1[6:0]** P25,P24,P23,P37,P36,P16,P15 port selection register

0 Disable analog channel AN[14:8] (default)

1 Enable analog channel AN[14:8]

## 9.4 Port RESET Noise Canceller

The Figure 13-21 is the Noise canceller diagram for Noise cancel of Pore RESET. It has the Noise cancel value of about 5us (@$V_{DD}$=5V) to input of Port Reset.

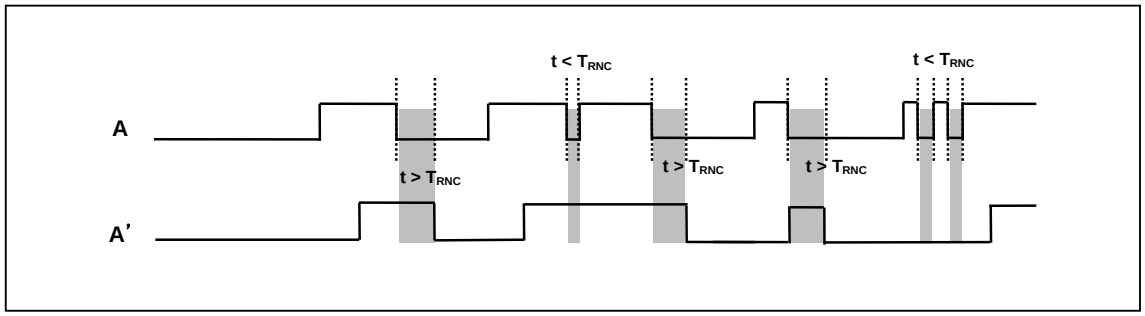



Figure 9-1 Port Reset noise canceller time diagram

# 10. Interrupt Controller

## 10.1 Overview

The Z51F0811 supports up to 32 interrupt sources. The interrupts have separate enable register bits associated with them, allowing software control. They can also have four levels of priority assigned to them. The interrupt controller has following features:

- receive the request from 32 interrupt source
- 8 group priority
- 4 priority levels
- Multi Interrupt possibility
- If the requests of different priority levels are received simultaneously, the request of higher priority level is serviced
- Each interrupt source can control by EA bit and each IEx bit
- Interrupt latency: 5~8 machine cycles in single interrupt system

The maskable interrupts are enabled through six of interrupt enable registers (IE, IE1, IE2, IE3, IE4, IE5). Bits of IE, IE1, IE2, IE3, IE4, IE5 register each individually enable/disable a particular interrupt source. Overall control is provided by bit 7 of IE (EA). When EA is set to '0', all interrupts are disabled: when EA is set to '1', interrupts are individually enabled or disabled through the other bits of the interrupt enable registers. The Z51F0811 supports a four-level priority scheme. Each maskable interrupt is individually assigned to one of four priority levels by writing to IP or IP1.

Interrupt default mode is level-trigger basically but if needed, it is able to change edge-trigger mode. Table 10-1 shows the Interrupt Group Priority Level that is available for sharing interrupt priority. Priority sets two bit which is to IP and IP1 register about group. Interrupt service routine services higher priority. If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If the request of same or lower priority level is received, that request is not serviced.

Table 10-1 Interrupt Group Priority Level

| Interrupt Group | Highest → Lowest | | | | |
|---|---|---|---|---|---|
| 0 (Bit0) | Interrupt0 | Interrupt8 | Interrupt16 | Interrupt24 | Highest |
| 1 (Bit1) | Interrupt1 | Interrupt9 | Interrupt17 | Interrupt25 | |
| 2 (Bit2) | Interrupt2 | Interrupt10 | Interrupt18 | Interrupt26 | |
| 3 (Bit3) | Interrupt3 | Interrupt11 | Interrupt19 | Interrupt27 | |
| 4 (Bit4) | Interrupt4 | Interrupt12 | Interrupt20 | Interrupt28 | |
| 5 (Bit5) | Interrupt5 | Interrupt13 | Interrupt21 | Interrupt29 | |
| 6 (Bit6) | Interrupt6 | Interrupt14 | Interrupt22 | Interrupt30 | |
| 7 (Bit7) | Interrupt7 | Interrupt15 | Interrupt23 | Interrupt31 | Lowest |

## 10.2 External Interrupt

The external interrupt on INT0, INT1, INT2, INT3, INT4, INT5, INT6 and INT7 pins receive various interrupt request depending on the EIEDGE (External Interrupt Edge register) and EIPOLA (External

Interrupt Polarity register) and EIBOTH(External Interrupt Both Edge register) as shown in Figure 10-1. Also each external interrupt source has control setting bits. The EIFLAG (External interrupt flag register) register provides the status of external interrupts.

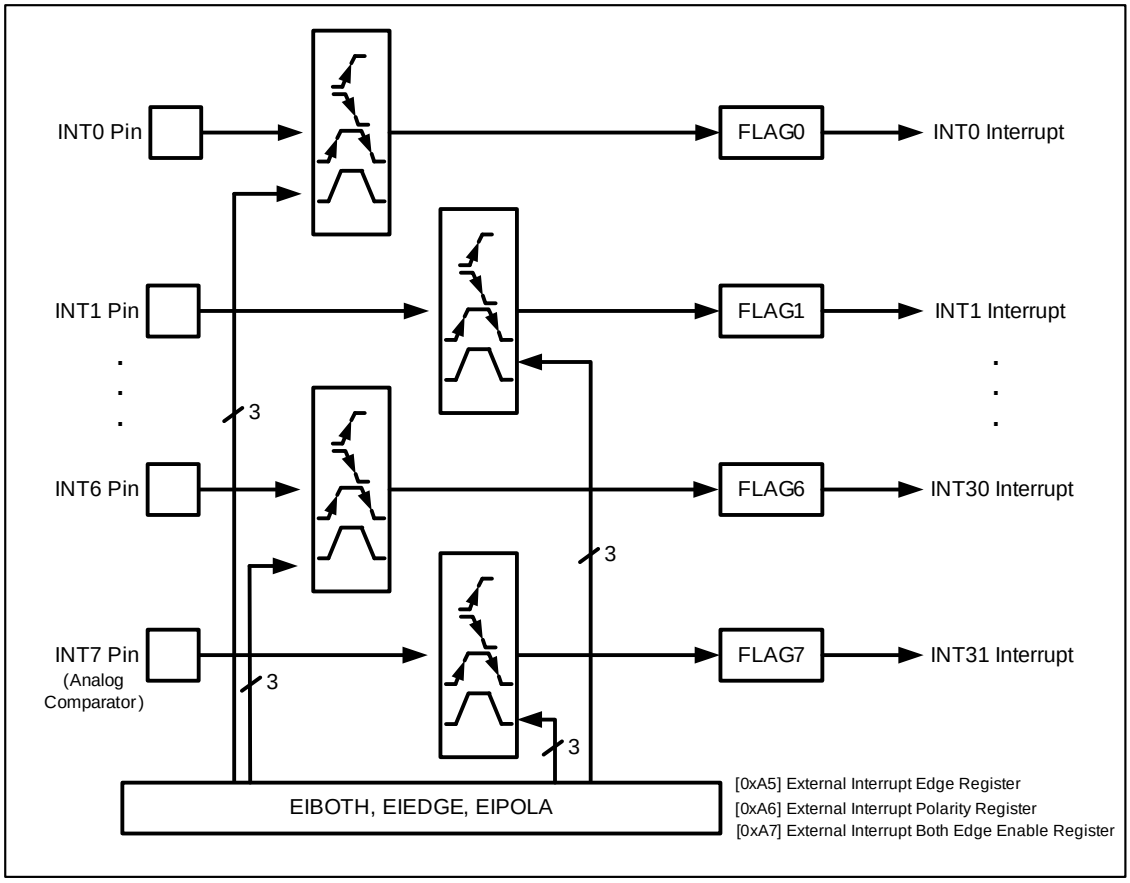



Figure 10-1 External Interrupt Description

## 10.3 Block Diagram

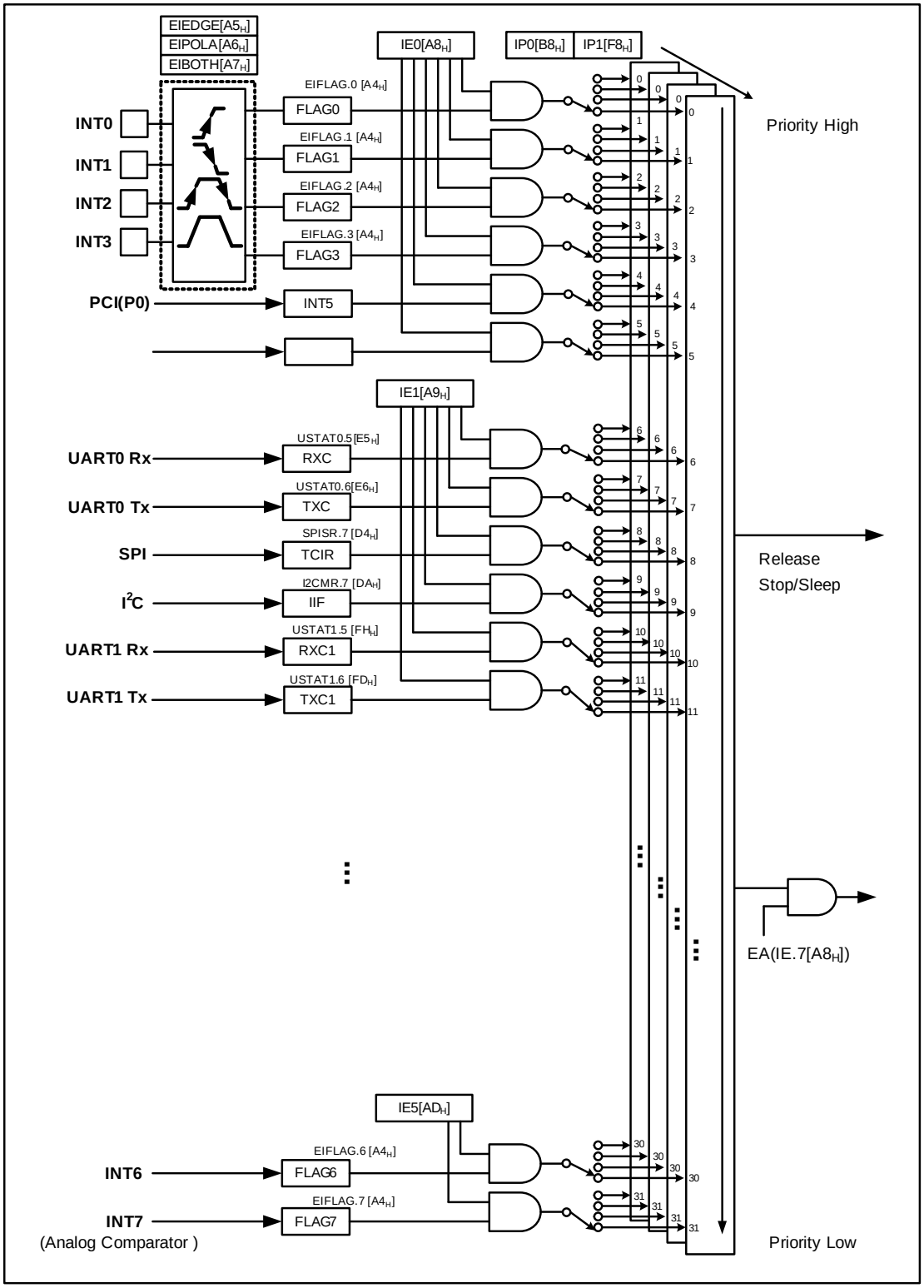



Figure 10-2 Block Diagram of Interrupt

## 10.4 Interrupt Vector Table

The interrupt controller supports 32 interrupt sources as shown in the Table 10-2 below. When interrupt becomes service, long call instruction (LCALL) is executed in the vector address. Interrupt request 32 has a decided priority order.

Table 10-2 Interrupt Vector Address Table

| Interrupt Source | Symbol | Interrupt Enable Bit | Priority | Mask | Vector Address |
|---|---|---|---|---|---|
| Hardware Reset | RESETB | 0 | 0 | Non-Maskable | 0000H |
| External Interrupt 0 | INT0 | IE0.0 | 1 | Maskable | 0003H |
| External Interrupt 1 | INT1 | IE0.1 | 2 | Maskable | 000BH |
| External Interrupt 2 | INT2 | IE0.2 | 3 | Maskable | 0013H |
| External Interrupt 3 | INT3 | IE0.3 | 4 | Maskable | 001BH |
| Pin Change Interrupt (P0) | INT4 | IE0.4 | 5 | Maskable | 0023H |
| Reserved | INT5 | - | 6 | - | 002BH |
| USART0 Rx | INT6 | IE1.0 | 7 | Maskable | 0033H |
| USART0Tx | INT7 | IE1.1 | 8 | Maskable | 003BH |
| SPI0 | INT8 | IE1.2 | 9 | Maskable | 0043H |
| $I^2C$ | INT9 | IE1.3 | 10 | Maskable | 004BH |
| USART1 Rx | INT10 | IE1.4 | 11 | Maskable | 0053H |
| USART1 Tx | INT11 | IE1.5 | 12 | Maskable | 005BH |
| T0 | INT12 | IE2.0 | 13 | Maskable | 0063H |
| T1 | INT13 | IE2.1 | 14 | Maskable | 006BH |
| T2 | INT14 | IE2.2 | 15 | Maskable | 0073H |
| T3 | INT15 | IE2.3 | 16 | Maskable | 007BH |
| T4 | INT16 | IE2.4 | 17 | Maskable | 0083H |
| EEPROM | INT17 | IE2.5 | 18 | Maskable | 008BH |
| ADC | INT18 | IE3.0 | 19 | Maskable | 0093H |
| Comparator | INT19 | IE3.1 | 20 | Maskable | 009BH |
| WT | INT20 | IE3.2 | 21 | Maskable | 00A3H |
| WDT | INT21 | IE3.3 | 22 | Maskable | 00ABH |
| BIT | INT22 | IE3.4 | 23 | Maskable | 00B3H |
| Reserved | INT23 | - | 24 | - | 00BBH |
| Reserved | INT24 | - | 25 | - | 00C3H |
| Reserved | INT25 | - | 26 | - | 00CBH |
| Reserved | INT26 | - | 27 | - | 00D3H |
| Reserved | INT27 | - | 28 | - | 00DBH |
| External Interrupt 4 | INT28 | IE4.4 | 29 | Maskable | 00E3H |
| External Interrupt 5 | INT29 | IE4.5 | 30 | Maskable | 00EBH |
| External Interrupt 6 | INT30 | IE5.0 | 31 | Maskable | 00F3H |
| External Interrupt 7 | INT31 | IE5.1 | 32 | Maskable | 00FBH |

For maskable interrupt execution, first EA bit must set ‘1’ and specific interrupt source must set ‘1’ by writing a ‘1’ to associated bit in the IEx. If interrupt request is received, specific interrupt request flag set ‘1’. And it remains ‘1’ until CPU accepts interrupt. After that, interrupt request flag will be cleared automatically.

## 10.5 Interrupt Sequence

An interrupt request is held until the interrupt is accepted or the interrupt latch is cleared to ‘0’ by a reset or an instruction. Interrupt acceptance always generates at last cycle of the instruction. So instead of fetching the current instruction, CPU executes internally LCALL instruction and saves the PC stack. For the interrupt service routine, the interrupt controller gives the address of LJMP instruction to CPU. After finishing the current instruction, at the next instruction to go interrupt service routine needs 5~8 machine cycle and the interrupt service task is terminated upon execution of an interrupt return instruction [RETI]. After generating interrupt, to go to interrupt service routine, the following process is progressed

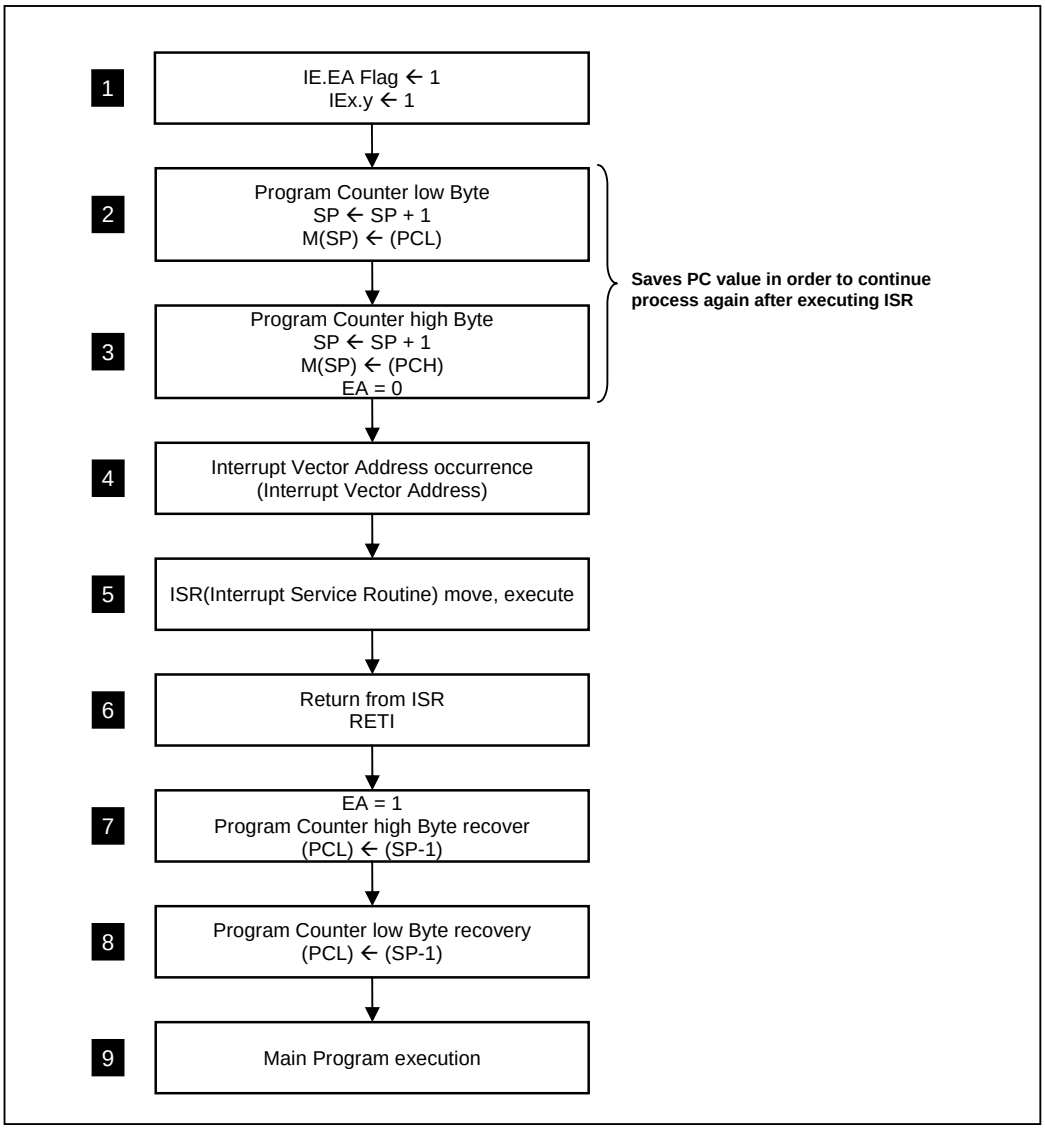



Figure 10-3 Interrupt Vector Address Table

## 10.6 Effective Timing after Controlling Interrupt bit

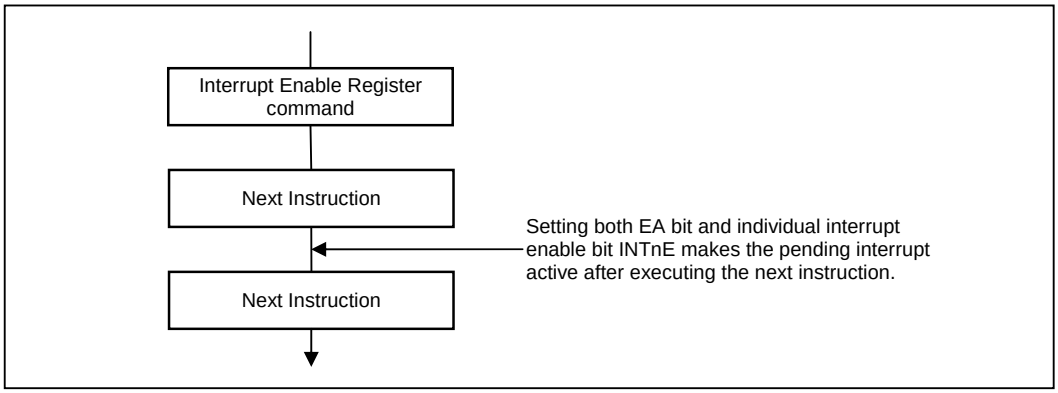



Figure 10-4 Interrupt Enable Register effective Timing

## 10.7 Multi Interrupt

If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If requests of the interrupt are received at the same time simultaneously, an interrupt polling sequence determines by hardware which request is serviced. However, multiple processing through software for special features is possible.

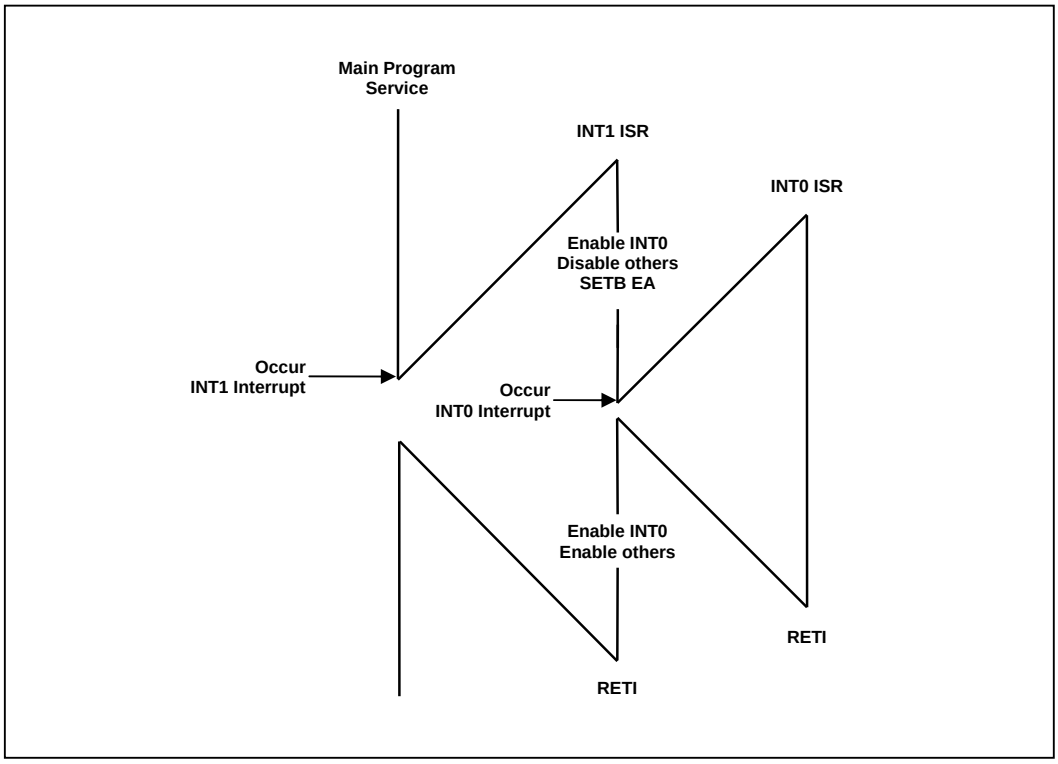



Figure 10-5 Execution of Multi Interrupt

Following example is shown to service INT0 routine during INT1 routine in Figure 10-6. In this example, INT0 interrupt priority is higher than INT1 interrupt priority. If some interrupt is lower than INT1 priority, it can't service its interrupt routine.

Example) Software Multi Interrupt:

```
INT1:   MOV     IE, #01H      ; Enable INT0 only
        MOV     IE1, #00H     ; Disable others
        SETB    EA            ; Enable global interrupt (necessary for multi interrupt)
        :
        MOV     IE, #0FFH     ; Enable all Interrupts
        MOV     IE1, #0FFH
        RETI
```

## 10.8 Interrupt Enable Accept Timing

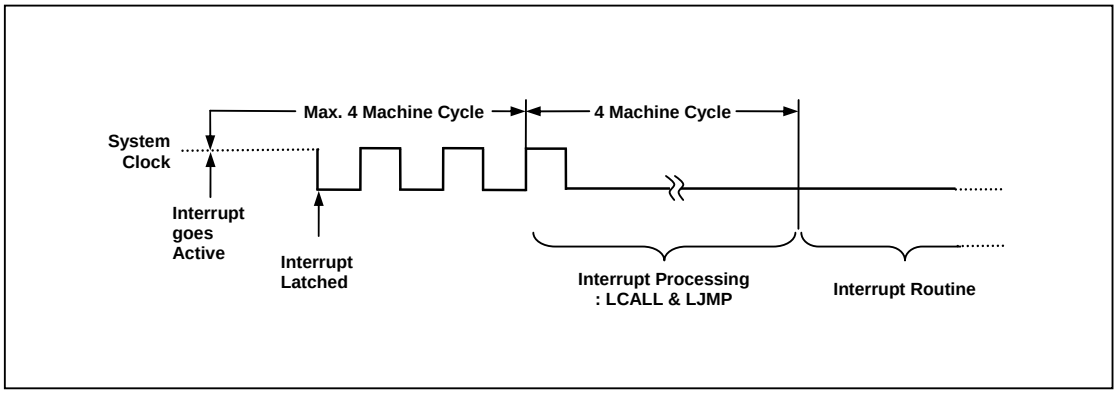



Figure 10-6 Interrupt Response Timing Diagram

## 10.9 Interrupt Service Routine Address

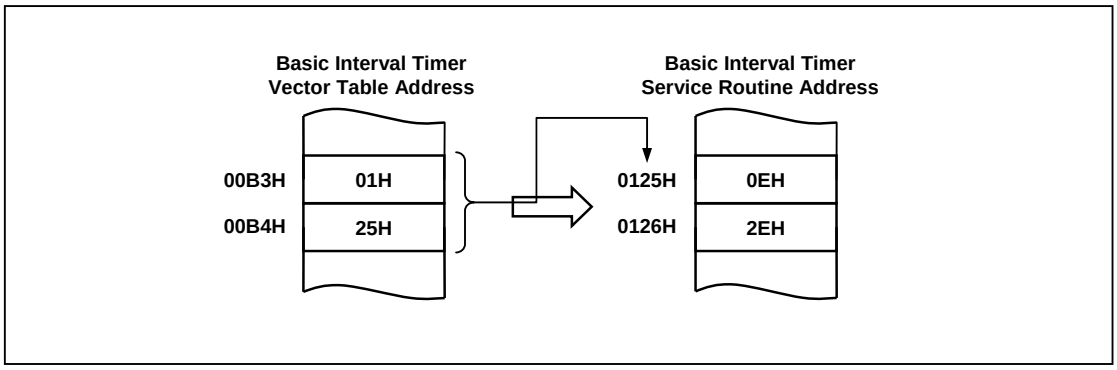



Figure 10-7 Correspondence between vector Table address and the entry address of ISP

## 10.10 Saving/Restore General-Purpose Registers

```
INTxx : PUSH   PSW
        PUSH   DPL
        PUSH   DPH
        PUSH   B
        PUSH   ACC
        :

Interrupt_Processing:
        :

       POP   ACC
       POP   B
       POP   DPH
       POP   DPL
       POP   PSW
       RETI
```

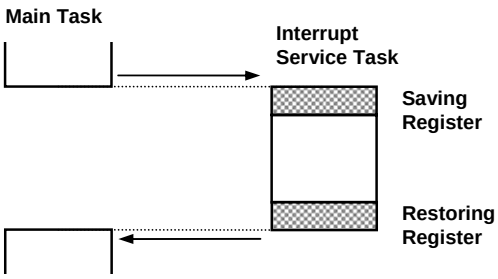



Figure 10-8 Saving/Restore Process Diagram & Sample Source

## 10.11 Interrupt Timing

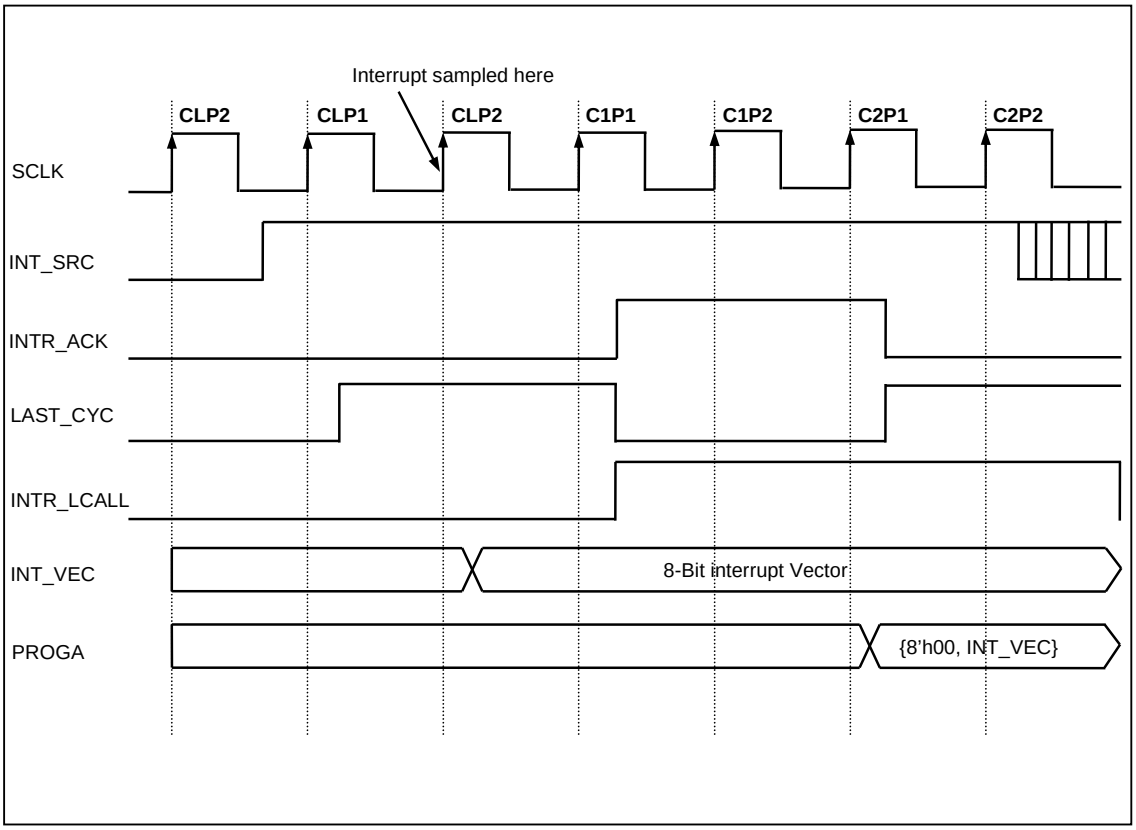



Figure 10-9 Timing chart of Interrupt Acceptance and Interrupt Return Instruction

Interrupt source sampled at last cycle of the command. When sampling interrupt source, it is decided to low 8-bit of interrupt vector. M8051W core makes interrupt acknowledge at first cycle of command, executes long call to jump interrupt routine as INT_VEC.

Note) command cycle C?P?: L=Last cycle, 1=1$^{st}$ cycle or 1$^{st}$ phase, 2=2$^{nd}$ cycle or 2$^{nd}$ phase

## 10.12 Interrupt Register Overview

### 10.12.1 Interrupt Enable Register (IE, IE1, IE2, IE3, IE4, IE5)

Interrupt enable register consists of Global interrupt control bit (EA) and peripheral interrupt control bits. Totally 32 peripheral are able to control interrupt.

### 10.12.2 Interrupt Priority Register (IP, IP1)

The 32 interrupt divides 8 groups which have each 4 interrupt sources. A group can decide 4 levels interrupt priority using interrupt priority register. Level 3 is the high priority, while level 0 is the low priority. Initially, IP, IP1 reset value is ‘0’. At that initialization, low interrupt number has a higher priority than high interrupt number. If decided the priority, low interrupt number has a higher priority than high interrupt number in that group.

### 10.12.3 External Interrupt Flag Register (EIFLAG)

The external interrupt flag register is set to ‘1’ when the external interrupt generating condition is satisfied. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a ‘0’ to it.

### 10.12.4 External Interrupt Edge Register (EIEDGE)

The External interrupt edge register determines which type of edge or level sensitive interrupt. Initially, default value is level. For level, write ‘0’ to related bit. For edge, write ‘1’ to related bit.

### 10.12.5 External Interrupt Polarity Register (EIPOLA)

According to EIEDGE register, the external interrupt polarity (EIPOLA) register has a different meaning. If EIEDGE is level type, EIPOLA is able to have Low/High level value. If EIEGDE is edge type, EIPOLA is able to have rising/falling edge value.

### 10.12.6 External Interrupt Enable Register (EIENAB)

When the external interrupt enable register is written to ‘1’, the corresponding external pin interrupt is enabled. The EIEDGE and EIPOLA register defines whether the external interrupt is activated on rising or falling edge or level sensed.

### 10.12.7 External Interrupt Both Edge Enable Register (EIBOTH)

When the external interrupt both edge enable register is written to ‘1’, the corresponding external pin interrupt is enabled by both edges. Initially, default value is disabled.

### 10.12.8 Register Map

Table 10-3 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| IE | A8H | R/W | 00H | Interrupt Enable Register |
| IE1 | A9H | R/W | 00H | Interrupt Enable Register 1 |
| IE2 | AAH | R/W | 00H | Interrupt Enable Register 2 |
| IE3 | ABH | R/W | 00H | Interrupt Enable Register 3 |
| IE4 | ACH | R/W | 00H | Interrupt Enable Register 4 |
| IE5 | ADH | R/W | 00H | Interrupt Enable Register 5 |
| IP | B8H | R/W | 00H | Interrupt Priority Register |
| IP1 | F8H | R/W | 00H | Interrupt Priority Register 1 |
| EIENAB | A3H | R/W | 00H | External Interrupt Enable Register |
| EIFLAG | A4H | R/W | 00H | External Interrupt Flag Register |
| EIEDGE | A5H | R/W | 00H | External Interrupt Edge Register |
| EIPOLA | A6H | R/W | 00H | External Interrupt Polarity Register |
| EIBOTH | A7H | R/W | 00H | External Interrupt Both Edge Enable Register |

## 10.13 Interrupt Register Description

The Interrupt Register is used for controlling interrupt functions. Also it has External interrupt control registers. The interrupt register consists of Interrupt Enable Register (IE), Interrupt Enable Register 1 (IE1), Interrupt Enable Register 2 (IE2), Interrupt Enable Register 3 (IE3), Interrupt Enable Register 4 (IE4) and Interrupt Enable Register 5 (IE5). For external interrupt, it consists of External Interrupt Flag Register (EIFLAG), External Interrupt Edge Register (EIEDGE), External Interrupt Polarity Register (EIPOLA), External Interrupt Enable Register (EIENAB) and External Interrupt Both Edge Enable Register(EIBOTH).

### 10.13.1 Register description for Interrupt

**IE (Interrupt Enable Register) : A8H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EA | - | INT5E | INT4E | INT3E | INT2E | INT1E | INT0E |
| R/W | - | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**EA** Enable or disable all interrupt bits
- 0 All Interrupt disable
- 1 All Interrupt enable

**INT5E** Reserved
- 0 Disable
- 1 Enable

**INT4E** Enable or disable Pin Change Interrupt 0 (Port 0)
- 0 Disable
- 1 Enable

**INT3E** Enable or disable External Interrupt 3
- 0 Disable

1 Enable

**INT2E** Enable or disable External Interrupt 2

0 Disable

1 Enable

**INT1E** Enable or disable External Interrupt 1

0 Disable

1 Enable

**INT0E** Enable or disable External Interrupt 0

0 Disable

1 Enable

### IE1 (Interrupt Enable Register 1) : A9H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | INT11E | INT10E | INT9E | INT8E | INT7E | INT6E |
| - | - | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**INT11E** Enable or disable USART1 Tx Interrupt

0 Disable

1 Enable

**INT10E** Enable or disable USART1 Rx Interrupt

0 Disable

1 Enable

**INT9E** Enable or disable $I^2C$ Interrupt

0 Disable

1 Enable

**INT8E** Enable or disable SPI0 Interrupt

0 Disable

1 Enable

**INT7E** Enable or disable USART0 Tx Interrupt

0 Disable

1 Enable

**INT6E** Enable or disable USART0 Rx Interrupt

0 Disable

1 Enable

### IE2 (Interrupt Enable Register 2) : AAH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | INT17E | INT16E | INT15E | INT14E | INT13E | INT12E |
| - | - | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**INT17E** Enable or disable EEPROM Interrupt

0 Disable

1 Enable

**INT16E** Enable or disable Timer 4 Interrupt

0 Disable

1 Enable

**INT15E** Enable or disable Timer 3 Interrupt

0 Disable
1 Enable

**INT14E** Enable or disable Timer 2 Interrupt
0 Disable
1 Enable

**INT13E** Enable or disable Timer 1 Interrupt
0 Disable
1 Enable

**INT12E** Enable or disable Timer 0 Interrupt
0 Disable
1 Enable

### IE3 (Interrupt Enable Register 3) : ABH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | INT23E | INT22E | INT21E | INT20E | INT19E | INT18E |
| R | R | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**INT23E** Reserved
0 Disable
1 Enable

**INT22E** Enable or disable BIT Interrupt
0 Disable
1 Enable

**INT21E** Enable or disable WDT Interrupt
0 Disable
1 Enable

**INT20E** Enable or disable WT Interrupt
0 Disable
1 Enable

**INT19E** Enable or disable Analog Comparator Interrupt
0 Disable
1 Enable

**INT18E** Enable or disable ADC Interrupt
0 Disable
1 Enable

### IE4 (Interrupt Enable Register 4) : ACH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | INT29E | INT28E | INT27E | INT26E | INT25E | INT24E |
| R | R | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**INT29E** Enable or disable External Interrupt 5
0 Disable
1 Enable

**INT28E** Enable or disable External Interrupt 4
0 Disable
1 Enable

**INT27E** Reserved
0 Disable
1 Enable

**INT26E** Reserved
0 Disable
1 Enable

**INT25E** Reserved
0 Disable
1 Enable

**INT24E** Reserved
0 Disable
1 Enable

### IE5 (Interrupt Enable Register 5) : ADH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | INT35E | INT34E | INT33E | INT32E | INT31E | INT30E |
| R | R | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**INT35E** Reserved
0 Disable
1 Enable

**INT34E** Reserved
0 Disable
1 Enable

**INT33E** Reserved
0 Disable
1 Enable

**INT32E** Reserved
0 Disable
1 Enable

**INT31E** Enable or disable External Interrupt 7
0 Disable
1 Enable

**INT30E** Enable or disable External Interrupt 6
0 Disable
1 enable

### IP (Interrupt Priority Register) : B8H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IP7 | IP6 | IP5 | IP4 | IP3 | IP2 | IP1 | IP0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

### IP1 (Interrupt Priority Register 1) : F8H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IP17 | IP16 | IP15 | IP14 | IP13 | IP12 | IP11 | IP10 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**IP[7:0], IP1[7:0]** Select Interrupt Group Priority

| IP1x | IPx | Description |
|---|---|---|
| 0 | 0 | level 0 (lowest) |
| 0 | 1 | level 1 |
| 1 | 0 | level 2 |
| 1 | 1 | level 3 (highest) |

### EIFLAG (External Interrupt Flag Register) : A4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FLAG7 | FLAG6 | FLAG5 | FLAG4 | FLAG3 | FLAG2 | FLAG1 | FLAG0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**FLAG[7:0]** If External Interrupt is occurred, the flag becomes '1'. The flag can be cleared by writing a '0' to bit

0 External Interrupt not occurred

1 External Interrupt occurred

### EIEDGE (External Interrupt Edge Register) : A5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EDGE7 | EDGE6 | EDGE5 | EDGE4 | EDGE3 | EDGE2 | EDGE1 | EDGE0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**EDGE[7:0]** Determines which type of edge or level sensitive interrupt may occ ur.

0 Level (default)

1 Edge

### EIPOLA (External Interrupt Polarity Register) : A6H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| POLA7 | POLA6 | POLA5 | POLA4 | POLA3 | POLA2 | POLA1 | POLA0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**POLA[7:0]** According to EIEDGE, External interrupt polarity register has a different means. If EIEDGE is level type, external interrupt polarity is able to have Low/High level value. If EIEGDE is edge type, external interrupt polarity is able to have rising/ falling edge value.

Level case:

0 When High level, Interrupt occurred (default)

1 When Low level, Interrupt occurred

Edge case:

0 When Rising edge, Interrupt occurred (default)

1 When Falling edge, Interrupt occurred

### EIENAB (External Interrupt Enable Register) : A3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ENAB7 | ENAB6 | ENAB5 | ENAB4 | ENAB3 | ENAB2 | ENAB1 | ENAB0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**ENAB[7:0]** Control External Interrupt

0 Disable (default)

1 Enable

### EIBOTH (External Interrupt Both Edge Enable Register) : A7H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BOTH7 | BOTH6 | BOTH5 | BOTH4 | BOTH3 | BOTH2 | BOTH1 | BOTH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**BOTH[7:0]** Determines which type of interrupt may occur, EIBOTH or EIEDGE+EIPOLA. if EIBOTH is enable, EIEDGE and EIPOLA register value don't matter.

0 Disable (default)

1 Enable

# 11. Peripheral Hardware

## 11.1 Clock Generator

### 11.1.1 Overview

As shown in Figure 11-1, the clock generator produces the basic clock pulses which provide the system clock to be supplied to the CPU and the peripheral hardware. It contains main-frequency clock oscillator. The system clock operation can be easily obtained by attaching a crystal between the XIN and XOUT pin, respectively. The system clock can also be obtained from the external oscillator. In this case, it is necessary to put the external clock signal into the XIN pin and open the XOUT pin. The default system clock is INT-RC Oscillator and the default division rate is one. In order to stabilize system internally, use 1MHz RING oscillator for BIT, WDT and ports de-bounce.

- Calibrated Internal RC Oscillator (8 MHz)
  - . INT-RC OSC/1 (Default system clock)
  - . INT-RC OSC/2 (4 MHz)
  - . INT-RC OSC/4 (2 MHz)
  - . INT-RC OSC/8 (1 MHz)
- Crystal Oscillator (1~16 MHz)
- Sub-Clock Crystal Oscillator (32.768 KHz)

### 11.1.2 Block Diagram

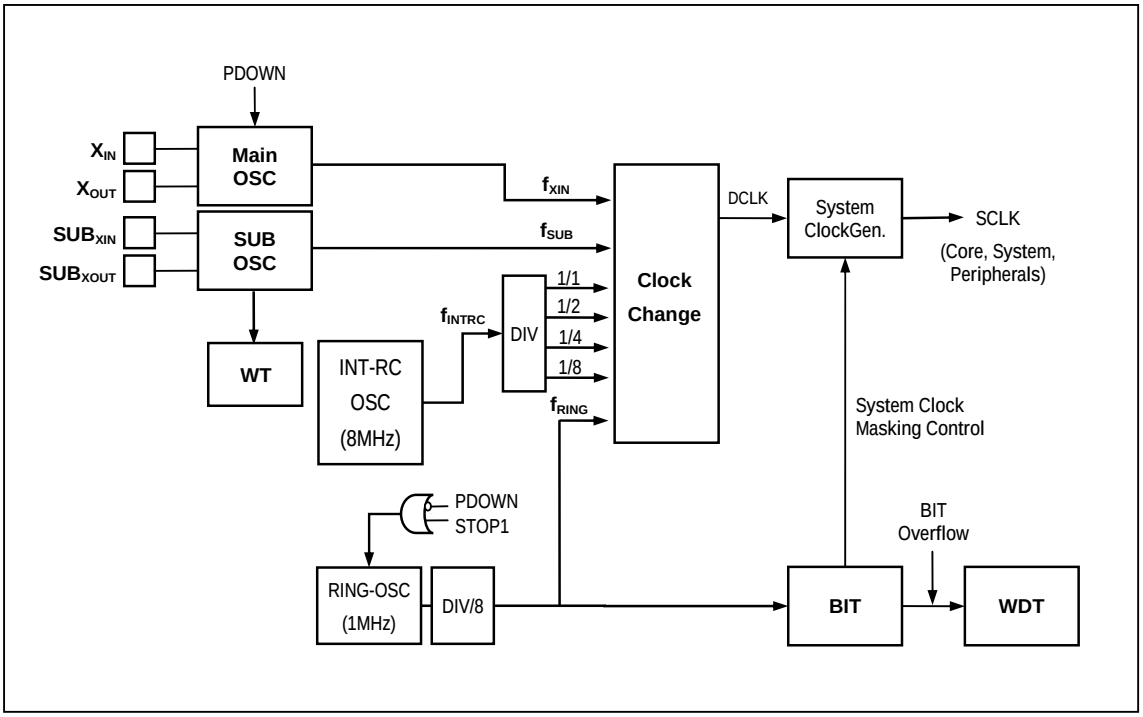



Figure 11-1 Clock Generator Block Diagram

### 11.1.3 Register Map

Table 11-1 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| SCCR | 8AH | R/W | 04H | System and Clock Control Register |

### 11.1.4 Clock Generator Register description

The Clock Generation Register uses clock control for system operation. The clock generation consists of System and Clock register.

### 11.1.5 Register description for Clock Generator

**SCCR (System and Clock Control Register) : 8AH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| STOP1 | DIV1 | DIV0 | CBYS | ISTOP | XSTOP | CS1 | CS0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 04H

**STOP1** Control the STOP Mode

Note) when PCON=0x03, It is applied. But when PCON=0x01, don't set this bit.

0 STOP2 Mode (at PCON=0x03) (default)

1 STOP1 Mode (at PCON=0x03)

**DIV[1:0]** When using fINTRC as system clock, determine division rate. Note) when using fINTRC as system clock, only division rate come into effect.

Note) To change by software, CBYS set to '1'

| DIV1 | DIV0 | description |
|---|---|---|
| 0 | 0 | fINTRC/1 (8MHz) |
| 0 | 1 | fINTRC/2 (4MHz) |
| 1 | 0 | fINTRC/4 (2MHz) |
| 1 | 1 | fINTRC/8 (1MHz) |

**CBYS** Control the scheme of clock change. If this bit set to '0', clock change is controlled by hardware. But if this set to '1', clock change is controlled by software. Ex) when setting CS[1:0], if CBYS bit set to '0', it is not changed right now, CPU goes to STOP mode and then when wake-up, it applies to clock change.

Note) when clear this bit, keep other bits in SCCR.

0 Clock changed by hardware during stop mode (default)

1 Clock changed by software

**ISTOP** Control the operation of INT-RC Oscillation

Note) when CBYS='1', It is applied

0 RC-Oscillation enable (default)

1 RC-Oscillation disable

**XSTOP** Control the operation of X-Tal Oscillation

Note1) when CBYS='1', It is applied

Note2) if XINENA bit in FUSE_CONF to '0', XSTOP is fixed to '1'

0 X-Tal Oscillation enable

1 X-Tal Oscillation disable (default)

**CS[1:0]** Determine System Clock
Note) by CBYS bit, reflection point is decided

| CS1 | CS0 | Description |
|---|---|---|
| 0 | 0 | fINTRC INTRC (8 MHz) |
| 0 | 1 | fXIN Main Clock (1~16 MHz) |
| 1 | 0 | fSUB (32.768 KHz) |
| 1 | 1 | fRING (125 KHz) |

## 11.2 BIT

### 11.2.1 Overview

The Z51F0811 has one 8-bit Basic Interval Timer that is free-run and can't stop. Block diagram is shown in Figure 11-2. In addition, the Basic Interval Timer generates the time base for watchdog timer counting. It also provides a basic interval timer interrupt (BITF).

The Z51F0811 has these Basic Interval Timer (BIT) features:

- During Power On, BIT gives a stable clock generation time
- On exiting Stop mode, BIT gives a stable clock generation time
- As clock function, time interrupt occurrence

### 11.2.2 Block Diagram

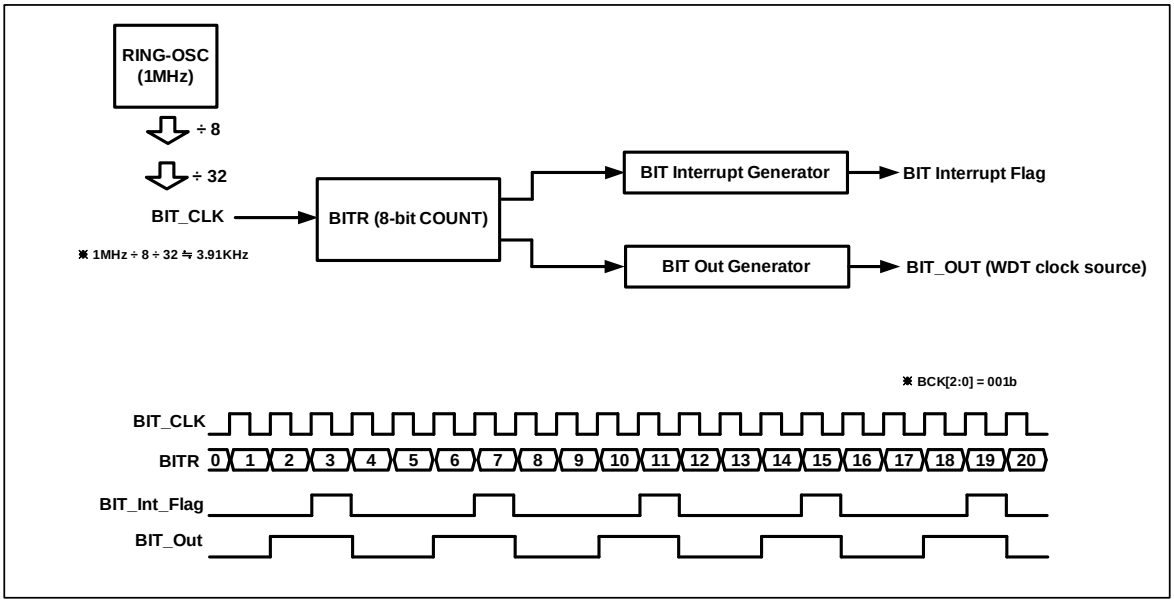



Figure 11-2 BIT Block Diagram

### 11.2.3 Register Map

Table 11-2 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| BCCR | 8BH | R/W | 05H | BIT Clock Control Register |
| BITR | 8CH | R | 00H | Basic Interval Timer Register |

### 11.2.4 Bit Interval Timer Register description

The Bit Interval Timer Register consists of BIT Clock control register (BCCR) and Basic Interval Timer register (BITR). If BCLR bit set to '1', BITR becomes '0' and then counts up. After 1 machine cycle, BCLR bit is cleared as '0' automatically.

### 11.2.5 Register description for Bit Interval Timer

**BCCR (BIT Clock Control Register) : 8BH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BITF | - | - | - | BCLR | BCK2 | BCK1 | BCK0 |
| R/W | R | R | R | R/W | R/W | R/W | R/W |

Initial value : 05H

**BITF** When BIT Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit.

0 no generation

1 generation

**BCLR** If BCLR Bit is written to '1', BIT Counter is cleared as '0'

0 Free Running

1 Clear Counter

**BCK[2:0]** Select BIT overflow period (BIT Clock ≒3.9 KHz)

| BCK2 | BCK1 | BCK0 | |
|---|---|---|---|
| 0 | 0 | 0 | 0.512msec (BIT Clock * 2) |
| 0 | 0 | 1 | 1.024msec |
| 0 | 1 | 0 | 2.048msec |
| 0 | 1 | 1 | 4.096msec |
| 1 | 0 | 0 | 8.192msec |
| 1 | 0 | 1 | 16.384msec (default) |
| 1 | 1 | 0 | 32.768msec |
| 1 | 1 | 1 | 65.536msec |

**BITR (Basic Interval Timer Register) : 8CH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BIT7 | BIT6 | BIT5 | BIT4 | BIT3 | BIT2 | BIT1 | BIT0 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**BIT[7:0]** BIT Counter

## 11.3 WDT

### 11.3.1 Overview

The watchdog timer rapidly detects the CPU malfunction such as endless looping caused by noise or the like, and resumes the CPU to the normal state. The watchdog timer signal for detecting malfunction can be selected either a reset CPU or an interrupt request. When the watchdog timer is not being used for malfunction detection, it can be used as a timer to generate an interrupt at fixed intervals. It is possible to use free running 8-bit timer mode (WDTRSON='0') or watch dog timer mode (WDTRSON='1') as setting WDTMR[6] bit. If writing WDTMR[5] to '1', WDT counter value is cleared and counts up. After 1 machine cycle, this bit has '0' automatically. The watchdog timer consists of 8-bit binary counter and the watchdog timer data register. When the value of 8-bit binary counter is equal to the 8 bits of WDTR, the interrupt request flag is generated. This can be used as Watchdog timer interrupt or reset the CPU in accordance with the bit WDTRSON.

The clock source of Watch Dog Timer is BIT overflow output. The interval of watchdog timer interrupt is decided by BIT overflow period and WDTR set value. The equation is as below

WDT Interrupt Interval = (BIT Interrupt Interval) X (WDTR Value+1)

### 11.3.2 Block Diagram

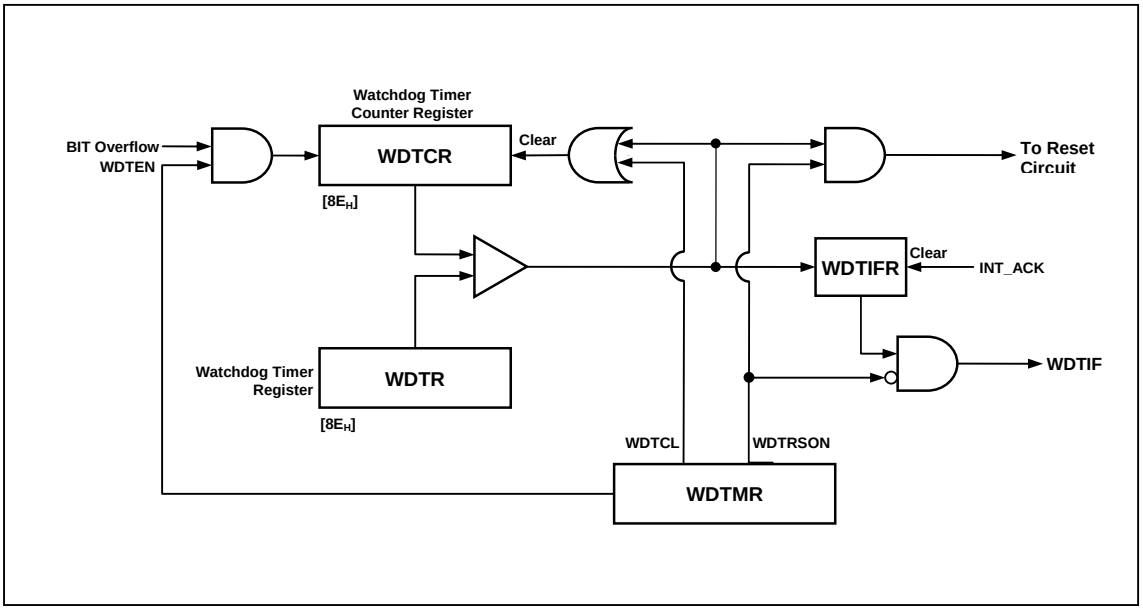



Figure 11-3 WDT Block Diagram

### 11.3.3 Register Map

Table 11-3 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| WDTR | 8EH | W | FFH | Watch Dog Timer Register |
| WDTCR | 8EH | R | 00H | Watch Dog Timer Counter Register |
| WDTMR | 8DH | R/W | 00H | Watch Dog Timer Mode Register |

### 11.3.4 Watch Dog Timer Register description

The Watch dog timer (WDT) Register consists of Watch Dog Timer Register (WDTR), Watch Dog Timer Counter Register (WDTCR) and Watch Dog Timer Mode Register (WDTMR).

### 11.3.5 Register description for Watch Dog Timer

**WDTR (Watch Dog Timer Register: Write Case) : 8EH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTR7 | WDTR6 | WDTR5 | WDTR4 | WDTR3 | WDTR2 | WDTR1 | WDTR0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**WDTR[7:0]** Set a period
WDT Interrupt Interval=(BIT Interrupt Interval) x(WDTR Value+1)

Note) To guarantee proper operation, the data should be greater than 01H.

**WDTCR (Watch Dog Timer Counter Register: Read Case) : 8EH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTCR7 | WDTCR6 | WDTCR5 | WDTCR4 | WDTCR3 | WDTCR2 | WDTCR1 | WDTCR0 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**WDTCR[7:0]** WDT Counter

**WDTMR (Watch Dog Timer Mode Register) : 8DH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTEN | WDTRSON | WDTCL | - | - | - | - | WDTIFR |
| R/W | R/W | R/W | - | - | - | - | R/W |

Initial value : 00H

**WDTEN** Control WDT operation
0 disable
1 enable

**WDTRSON** Control WDT Reset operation
0 Free Running 8-bit timer
1 Watch Dog Timer Reset ON

**WDTCL** Clear WDT Counter
0 Free Run
1 Clear WDT Counter (auto clear after 1 Cycle)

**WDTIFR** When WDT Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal.
0 WDT Interrupt no generation
1 WDT Interrupt generation

### 11.3.6 WDT Interrupt Timing Waveform

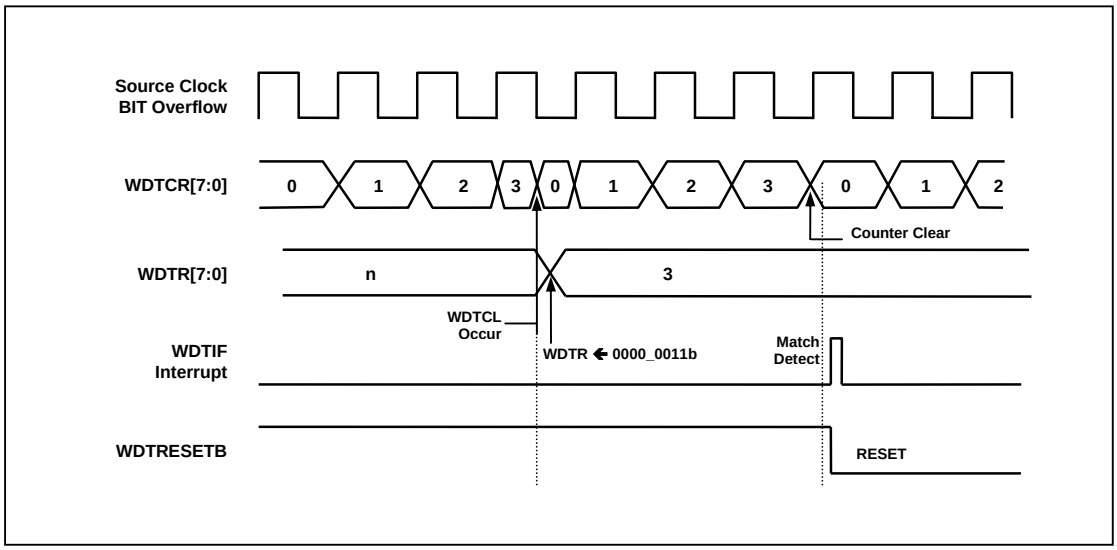



Figure 11-4 WDT Interrupt Timing Waveform

## 11.4 WT

### 11.4.1 Overview

The watch timer has the function for RTC (Real Time Clock) operation. It is generally used for RTC design. The internal structure of the watch timer consists of the clock source select circuit, timer counter circuit, output select circuit and watch timer mode register. To operate the watch timer, determine the input clock source, output interval and set WTEN to '1' in watch timer mode register (WTMR). It is able to execute simultaneously or individually. To stop or reset WT, clear the WTEN bit in WTMR register. Even if CPU is STOP mode, sub clock is able to be alive so WT can continue the operation. The watch timer counter circuits may be composed of 21-bit counter which is low 14-bit with binary counter and high 7-bit with auto reload counter in order to raise resolution. In WTR, it can control WT clear and set Interval value at write time, and it can read 7-bit WT counter value at read time.

### 11.4.2 Block Diagram

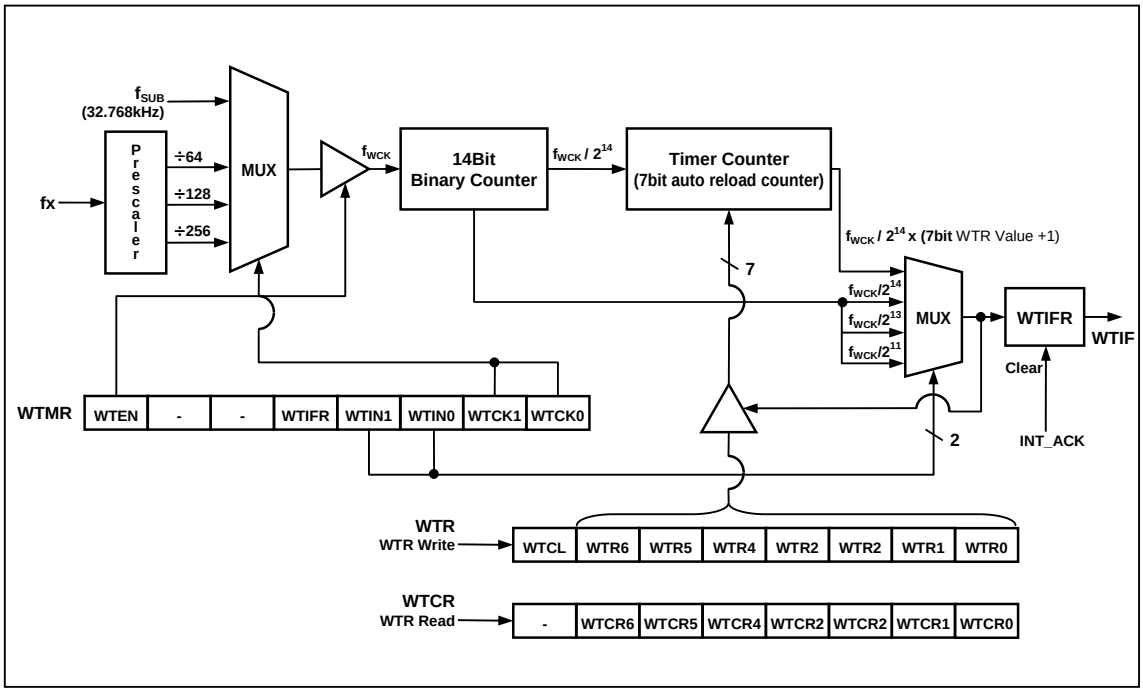



Figure 11-5 Watch Timer Block Diagram

### 11.4.3 Register Map

Table 11-4 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| WTMR | 9DH | R/W | 00H | Watch Timer Mode Register |
| WTR | 9EH | W | 7FH | Watch Timer Register |
| WTCR | 9EH | R | 00H | Watch Timer Counter Register |

### 11.4.4 Watch Timer Register description

The watch timer register (WT) consists of Watch Timer Mode Register (WTMR), Watch Timer Counter Register (WTCR) and Watch Timer Register (WTR). As WTMR is 6-bit writable/readable register, WTMR can control the clock source (WTCK), interrupt interval (WTIN) and function enable/disable (WTEN). Also there is WT interrupt flag bit (WTIFR).

### 11.4.5 Register description for Watch Timer

**WTMR (Watch Timer Mode Register) : 9DH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WTEN | - | - | WTIFR | WTIN1 | WTIN0 | WTCK1 | WTCK0 |
| R/W | - | - | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**WTEN** Control Watch Timer

0 disable

1 enable

**WTIFR** When WT Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal.

0 WT Interrupt no generation

1 WT Interrupt generation

**WTIN[1:0]** Determine interrupt interval

| WTIN1 | WTIN0 | description |
|---|---|---|
| 0 | 0 | fwck/2048 |
| 0 | 1 | fwck/8192 |
| 1 | 0 | fwck/16384 |
| 1 | 1 | fwck/16384 x (7bit WT Value) |

**WTCK[1:0]** Determine Source Clock

| WTCK1 | WTCK0 | description |
|---|---|---|
| 0 | 0 | fsub |
| 0 | 1 | fx/256 |
| 1 | 0 | fx/128 |
| 1 | 1 | fx/64 |

Remark: fx– Main system clock oscillation frequency
fsub- Sub clock oscillation frequency
fwck- selected Watch Timer clock

### WTR (Watch Timer Register: Write Case) : 9EH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WTCL | WTR6 | WTR5 | WTR4 | WTR3 | WTR2 | WTR1 | WTR0 |
| W | W | W | W | W | W | W | W |

Initial value : 7FH

**WTCL** Clear WT Counter

0 Free Run

1 Clear WT Counter (auto clear after 1 Cycle)

**WTR[6:0]** Set WT period

WT Interrupt Interval=(fwck/2^14) x(7bit WT Value+1)

Note) To guarantee proper operation, it is greater than 01H to write WTR.

### WTCR (Watch Timer Counter Register: Read Case) : 9EH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | WTCR6 | WTCR5 | WTCR4 | WTCR3 | WTCR2 | WTCR1 | WTCR0 |
| - | R | R | R | R | R | R | R |

Initial value : 00H

**WTCR[6:0]** WT Counter

## 11.5 Timer/PWM

### 11.5.1 8-bit Timer/Event Counter 0, 1

#### 11.5.1.1 Overview

Timer 0 and timer 1 can be used either two 8-bit timer/counter or one 16-bit timer/counter with combine them. Each 8-bit timer/event counter module has multiplexer, 8-bit timer data register, 8-bit counter register, mode register, input capture register, comparator. For PWM, it has PWM register (T1PPR, T1ADR, T1BDR, T1CDR, T1PCR, T1PCR2, T1PCR3, T1PHR, T1DLYA, T1DLYB, T1DLYC, T1ISR, T1IMSK).

It has seven operating modes:

- 8-bit timer/counter mode
- 8-bit capture mode
- 8-bit compare output mode
- 16-bit timer/counter mode
- 16-bit capture mode
- 16-bit compare output mode
- PWM mode

Note> TxDR must be set to higher than 0x03 for guaranteeing operation.

The timer/counter can be clocked by an internal or an external clock source (external EC0). The clock source is selected by clock select logic which is controlled by the clock select (T0CK[2:0], T1CK[3:0]). Also the timer/PWM/event counter 1 can use more clock sources than timer/event counter 0.

- TIMER0 clock source: fX/2, 4, 8, 32, 128, 512, 2048, EC0
- TIMER1 clock source: fX/1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192, 16384, T0CK

In the capture mode, by INT0, INT1, the data is captured into Input Capture Register. The timer 0 outputs the compare result to T0 port in 8/16-bit mode. Also the timer 1 outputs the result to T1 port in the timer mode and the PWM wave form to PWMA, PWMAB(bar), PWMB, PWMBB, PWMC, PWMCB Port(6-channel) in the PWM mode.

Table 11-5 Timer 0,1 operating modes

| 16 Bit | CAP0 | CAP1 | PWM1E | T0CK[2:0] | T1CK[3:0] | T0/1_PE | Timer 0 | Timer 1 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | XXX | XXXX | 00 | 8 Bit Timer | 8 Bit Timer |
| 0 | 0 | 1 | 0 | 111 | XXXX | 00 | 8 Bit Event Counter | 8 Bit Capture |
| 0 | 1 | 0 | 0 | XXX | XXXX | 01 | 8 Bit Capture | 8 Bit Compare Output |
| 0 | 0 | 0 | 1 | XXX | XXXX | 11 | 8 Bit Timer/Counter | 10 Bit PWM |
| 1 | 0 | 0 | 0 | XXX | 1111 | 00 | 16 Bit Timer | |
| 1 | 0 | 0 | 0 | 111 | 1111 | 00 | 16 Bit Event Counter | |
| 1 | 1 | 1 | 0 | XXX | 1111 | 00 | 16 Bit Capture | |
| 1 | 0 | 0 | 0 | XXX | 1111 | 01 | 16 Bit Compare Output | |

### 11.5.1.2 8 Bit Timer/Counter Mode

The 8-bit Timer/Counter Mode is selected by control registers as shown in Figure 11-6.

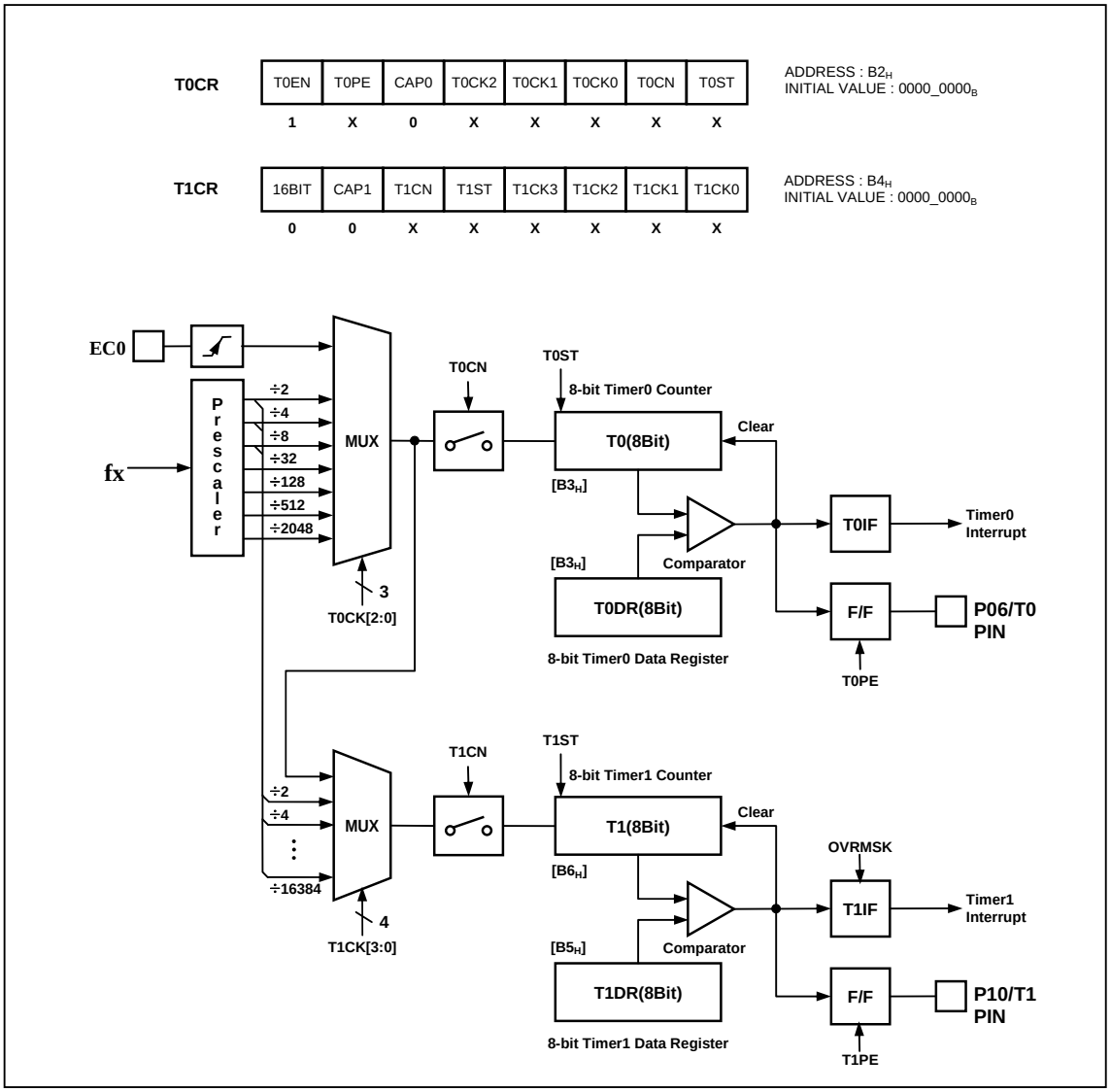



Figure 11-6 8 Bit Timer/Event Counter 0, 1 Block Diagram

The two 8-bit timers have each counter and data register. The counter register is increased by internal or external clock input. The timer 0 can use the input clock with one of 2, 4, 8, 32, 128, 512, 2048 prescaler division rates (T0CK[2:0]). The timer 1 can use the input clock with one of 1, 2, 8 ~ 16384 and timer 0 overflow clock (T1CK[3:0]). When the value of T0,1 value and the value of T0DR, T1DR are respectively identical in Timer 0, 1, the interrupt of TimerP 0, 1 occurs. The external clock (EC0) counts up the timer at the rising edge. If EC0 is selected from T0CK[2:0], EC0 port becomes input port. The timer 1 can't use the external EC0 clock.

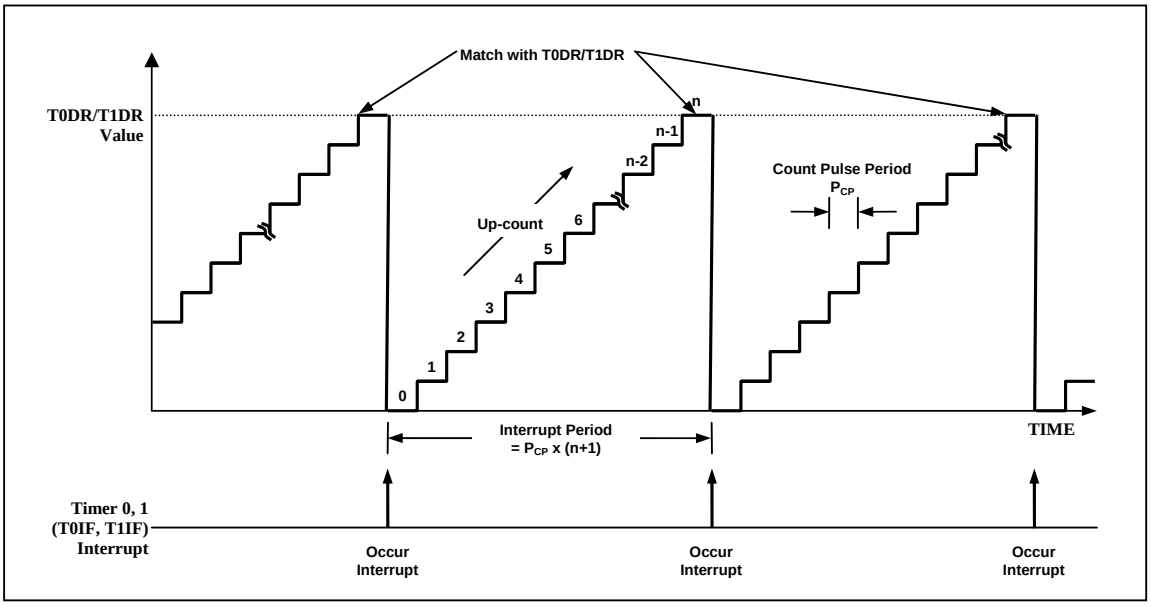



Figure 11-7 Timer/Event Counter 0, 1 Example

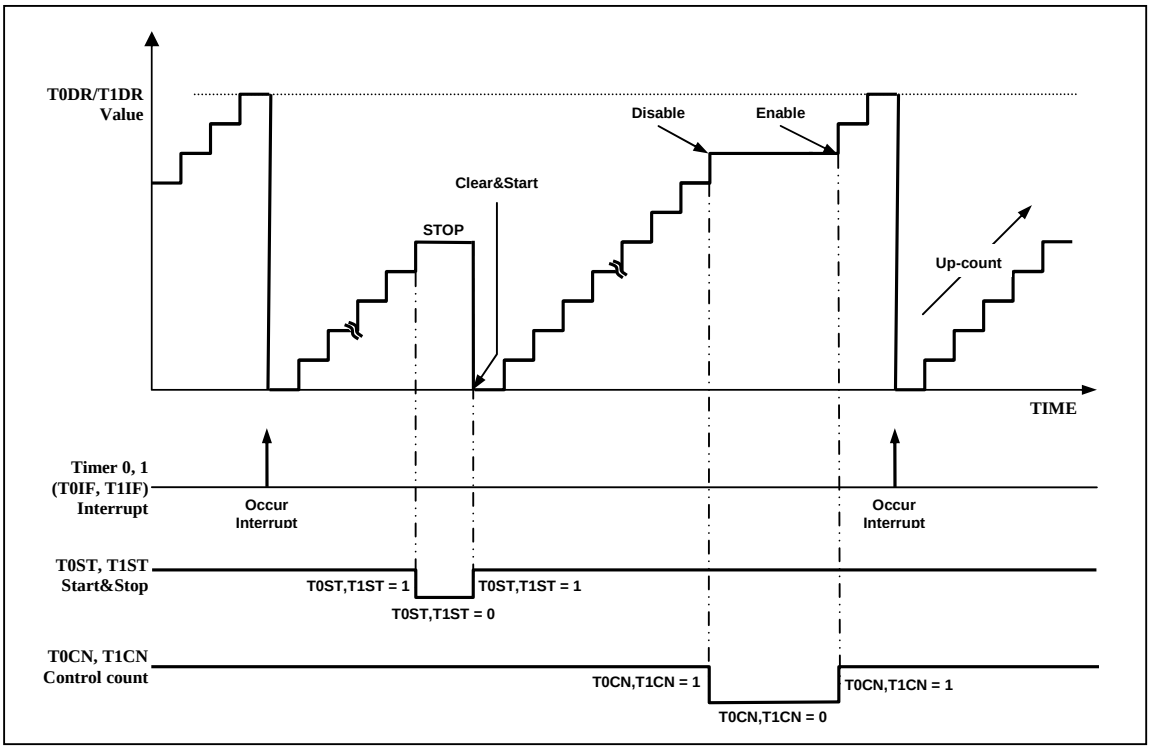



Figure 11-8 Timer/Event Counter0, 1 Count Operation

#### 11.5.1.3 16 Bit Timer/Counter Mode

The timer register is being run with all 16 bits. A 16-bit timer/counter register T0, T1 are incremented from 0003H to FFFFH until it matches T0DR, T1DR and then resets to 0000H. The match output

generates the Timer 0 interrupt (No timer 1 interrupt). The clock source is selected from T0CK[2:0] and T1CK[3:0] must set 1111b and 16BIT bit must set to '1'. The timer 0 is LSB 8-bit, the timer 1 is MSB 8-bit. T0DR must not be 0x00 ( 0x01~0xFF). The 16-bit mode selection is shown as Figure 11-9.

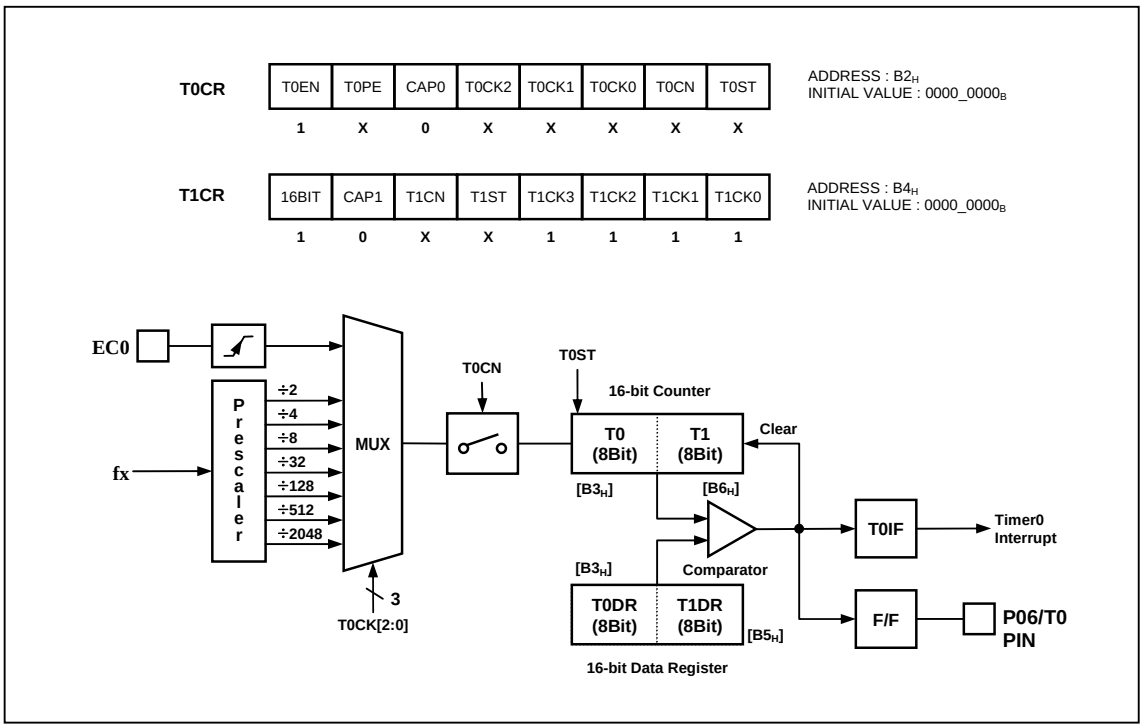



Figure 11-9 16-bit Timer/Counter for Time 0, 1

### 11.5.1.4 8-Bit Capture Mode

The timer 0, 1 capture mode is set by CAP0, CAP1 as '1'. The clock source can use the internal/external clock. Basically, it has the same function of the 8-bit timer/counter mode and the interrupt occurs at T0, T1 and T0DR, T1DR matching time, respectively. The capture result is loaded into CDR0, CDR1. The T0, T1 value is automatically cleared by hardware and restarts counter.

This timer interrupt in capture mode is very useful when the pulse width of captured signal is wider than the maximum period of timer.

As the EIEDGE and EIPOLA register setting, the external interrupt INT1, INT0 function is chosen.

The CDR0, T0 and T0DR are in same address. In the capture mode, reading operation reads the CDR0, not T0DR because path is opened to the CDR0. The CDR1 has the same function.

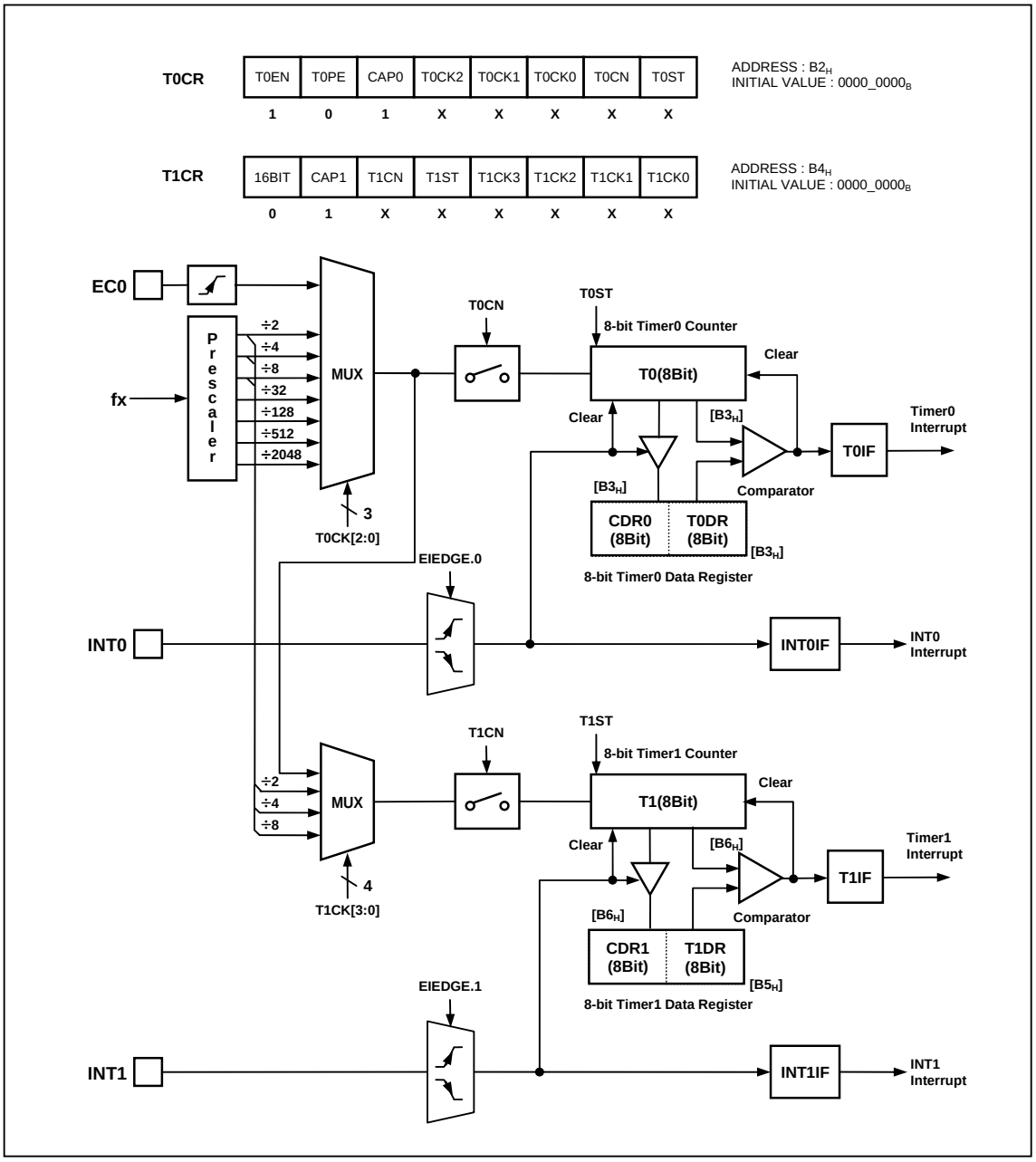



Figure 11-10 8-bit Capture Mode for Timer 0, 1

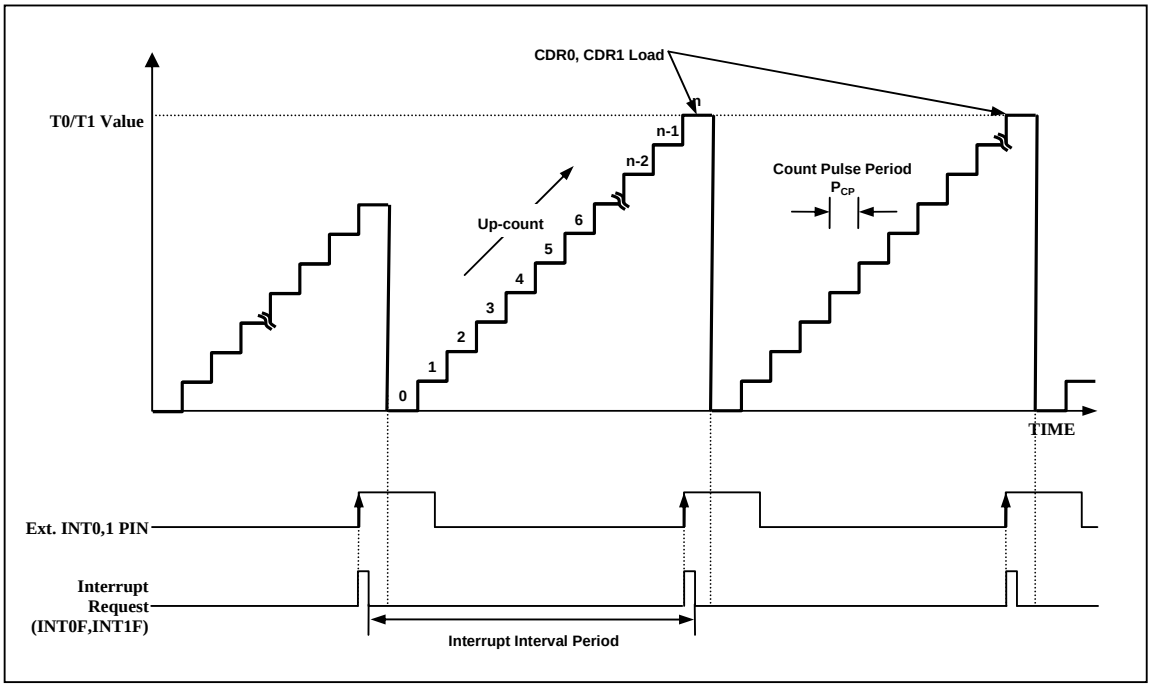



Figure 11-11 Input Capture Mode Operation of Timer 0, 1

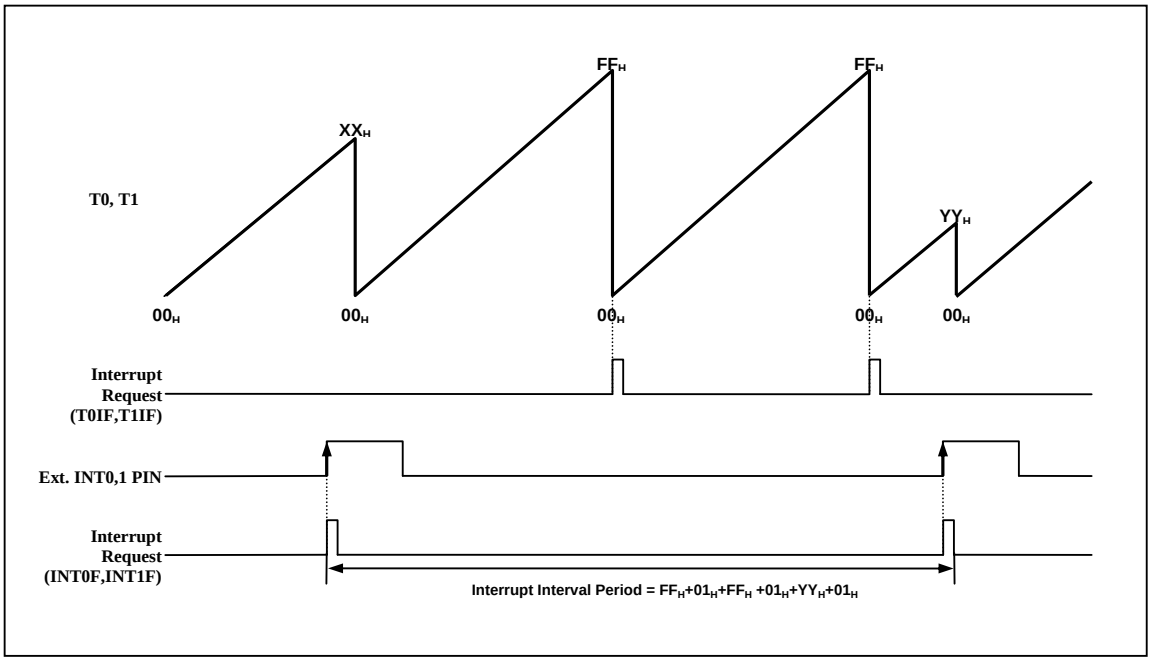



Figure 11-12 Express Timer Overflow in Capture Mode

#### 11.5.1.5 16 Bit Capture Mode

The 16-bit capture mode is the same operation as 8-bit capture mode, except that the timer register uses 16 bits.

The clock source is selected from T0CK[2:0] and T1CK[3:0] must set 1111b and 16BIT bit must set to '1'. The 16-bit mode setting is shown as Figure 11-13.

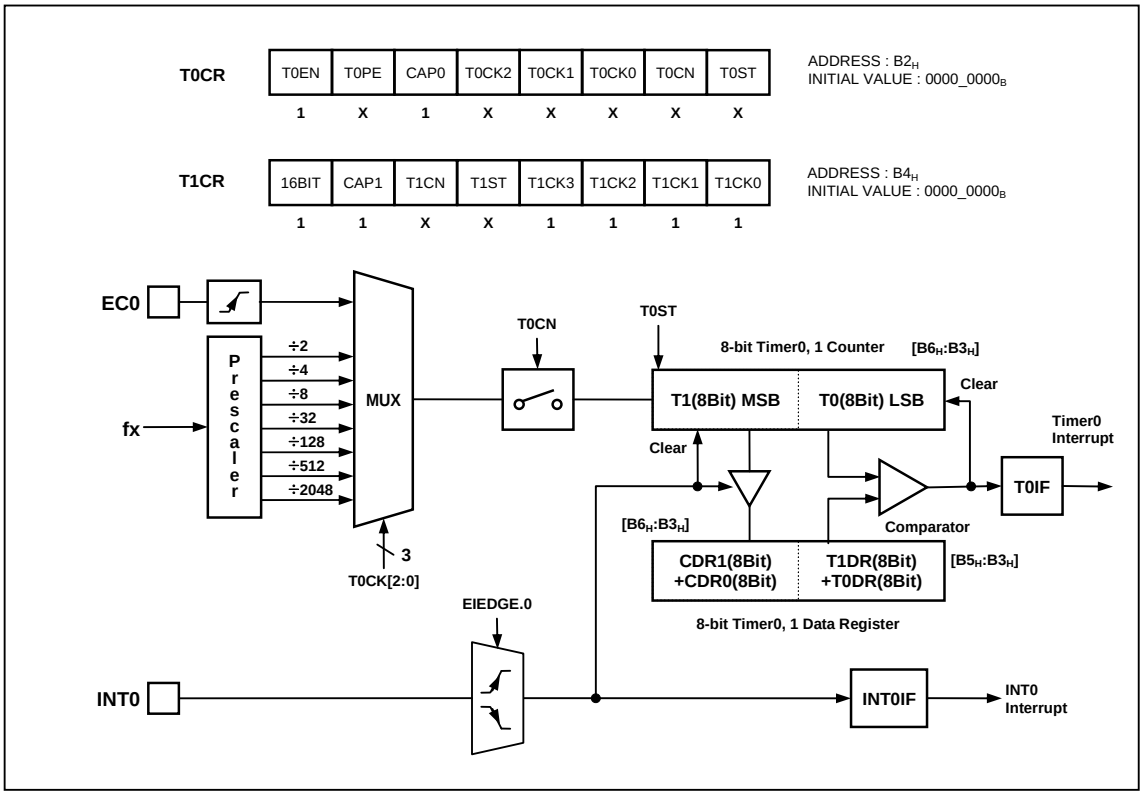



Figure 11-13 16-bit Capture Mode of Timer 0, 1

#### 11.5.1.6 PWM Mode

The timer 1 has a high speed PWM (pulse Width Modulation) function. In PWM mode, the 6-channel pins output up to 10-bit resolution PWM output. This pin should be configured as a PWM output by set PWM1E to '1'. The period of the PWM output is determined by the T1PPR (PWM period register) + T1PHR[1:0], T1xDR (each channel PWM duty register) + T1PHR[7:2].

PWM Period = [ T1PHR[1:0]T1PPR ] X Source Clock
PWM Duty(A-ch) = [ T1PHR[7:6] T1ADR ] X Source Clock

Note> T1PPR must be set to higher than T1PDR for guaranteeing operation.

Table 11-6 PWM Frequency vs. Resolution at 8 Mhz

| Resolution | Frequency | | |
|---|---|---|---|
| | T1CK[3:0]=0001 (250ns) | T1CK[3:0]=0010 (500ns) | T1CK[3:0]=0100 (2us) |
| 10 Bit | 3.9KHz | 1.95KHz | 0.49KHz |
| 9 Bit | 7.8KHz | 3.9KHz | 0.98KHz |
| 8 Bit | 15.6KHz | 7.8KHz | 1.95KHz |
| 7 Bit | 31.2KHz | 15.6KHz | 3.91KHz |

The POLx bit of T1PCR3 register decides the polarity of duty cycle. If the duty value is set same to the period value, the PWM output is determined by the bit POLx (1: High, 0: Low). And if the duty value is set to "00H", the PWM output is determined by the bit POLx (1: Low, 0: High).

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| T1CR | 16BIT | CAP1 | T1CN | T1ST | T1CK3 | T1CK2 | T1CK1 | T1CK0 | ADDRESS : $B4_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | 0 | 0 | X | X | X | X | X | X | |
| T1PCR | PWM1E | ESYNC | BMOD | PHLT | UPDT | UALL | NOPS1 | NOPS0 | ADDRESS : $B7_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | 1 | X | X | X | X | X | X | X | |
| T1PCR2 | FORCA | FORC6 | PAOE | PABOE | PBOE | PBBOE | PCOE | PCOE | ADDRESS : $BD_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | X | 1 | X | X | X | X | X | X | |
| T1PCR3 | T1_PE | POLA | POLB | POLC | POCON | HCKE | - | PLLPDB | ADDRESS : $BE_H$<br>INITIAL VALUE : $0000_00\text{-}0_B$ |
| | X | X | X | X | X | X | X | X | |
| T1PHR | ADR9 | ADR8 | BDR9 | BDR8 | CDR9 | CDR8 | PPR9 | PPR8 | ADDRESS : $BC_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | X | X | X | X | X | X | X | X | |

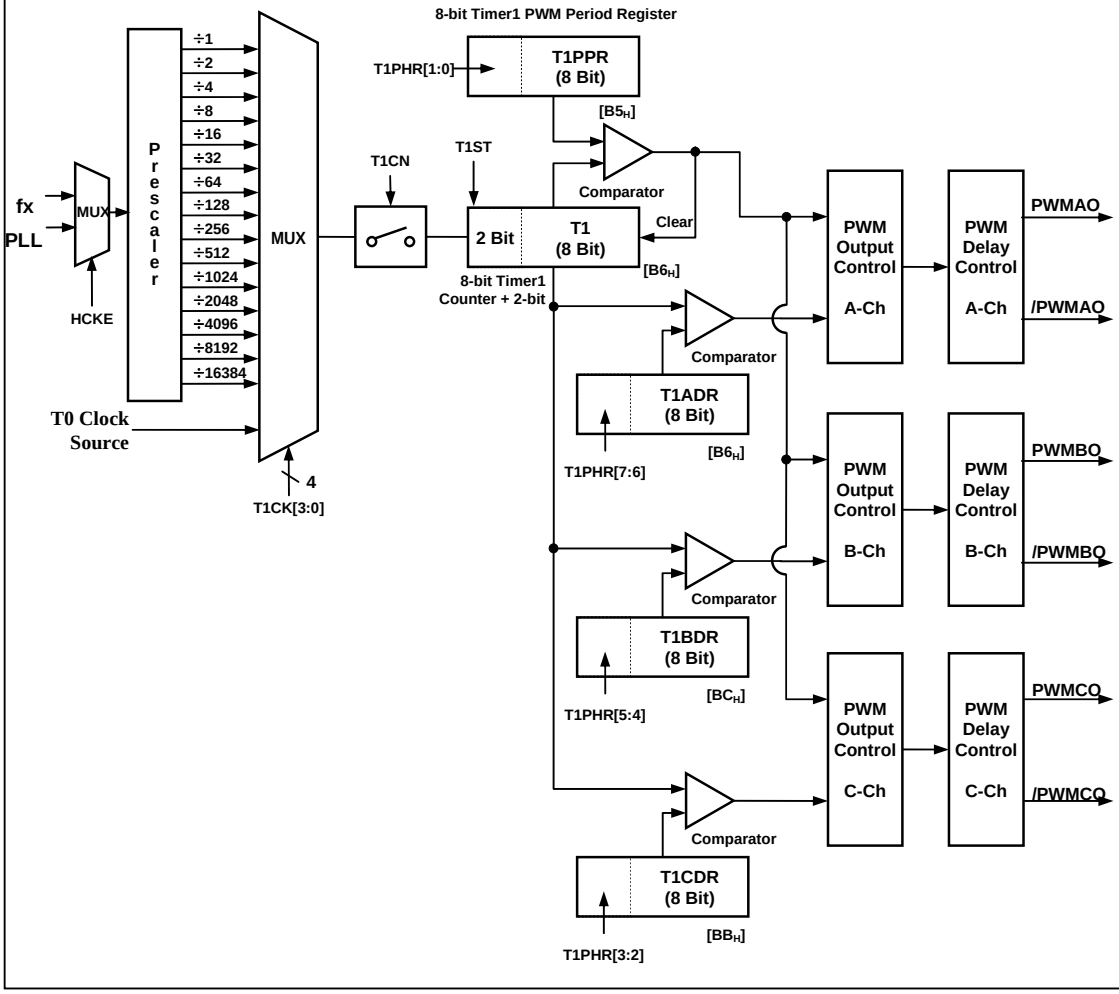



Figure 11-14 PWM Mode (Force 6-ch)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| T1CR | 16BIT | CAP1 | T1CN | T1ST | T1CK3 | T1CK2 | T1CK1 | T1CK0 | ADDRESS : $B4_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | 0 | 0 | X | X | X | X | X | X | |
| T1PCR | PWM1E | ESYNC | BMOD | PHLT | UPDT | UALL | NOPS1 | NOPS0 | ADDRESS : $B7_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | 1 | X | X | X | X | X | X | X | |
| T1PCR2 | FORCA | FORC6 | PAOE | PABOE | PBOE | PBBOE | PCOE | PCOE | ADDRESS : $BD_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | 1 | X | X | X | X | X | X | X | |
| T1PCR3 | T1_PE | POLA | POLB | POLC | POCON | HCKE | - | PLLPDB | ADDRESS : $BE_H$<br>INITIAL VALUE : $0000_00\text{-}0_B$ |
| | X | X | X | X | X | X | X | X | |
| T1PHR | ADR9 | ADR8 | BDR9 | BDR8 | CDR9 | CDR8 | PPR9 | PPR8 | ADDRESS : $BC_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | X | X | X | X | X | X | X | X | |

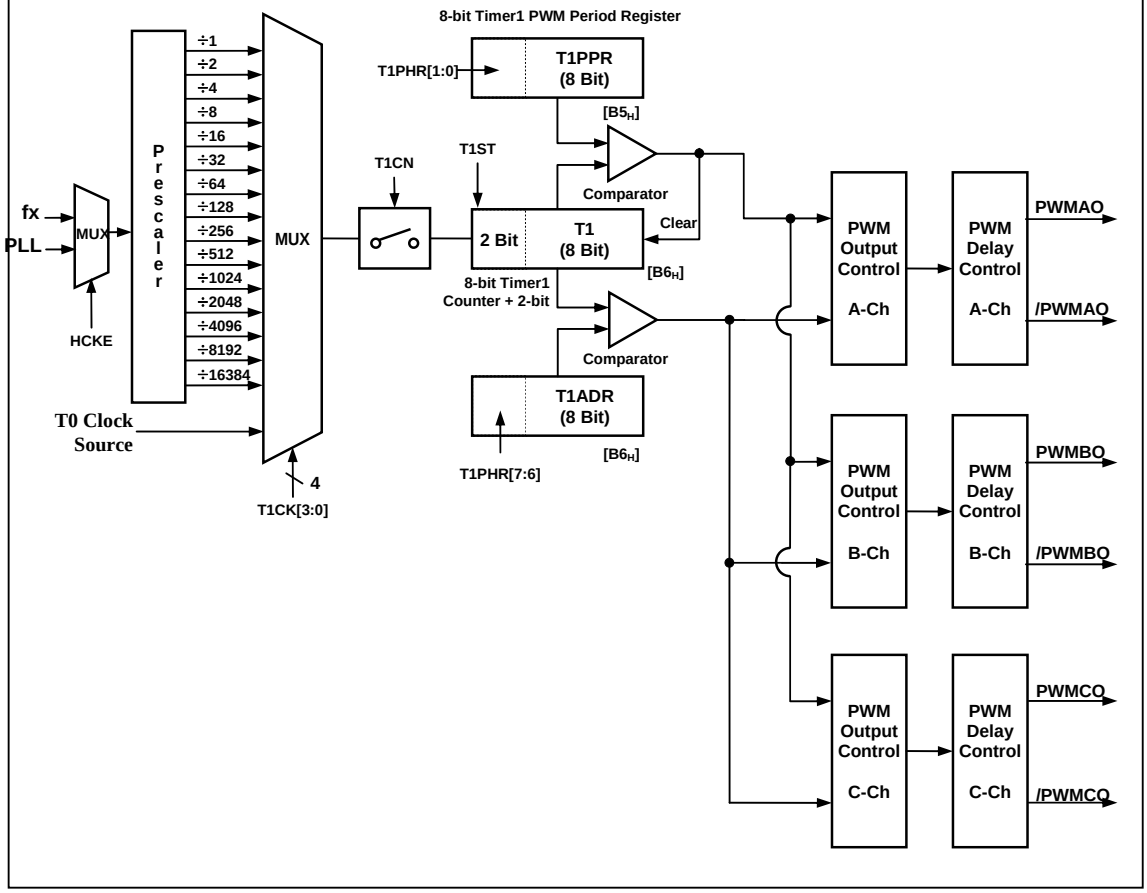



Figure 11-15 PWM Mode (Force All-Ch)

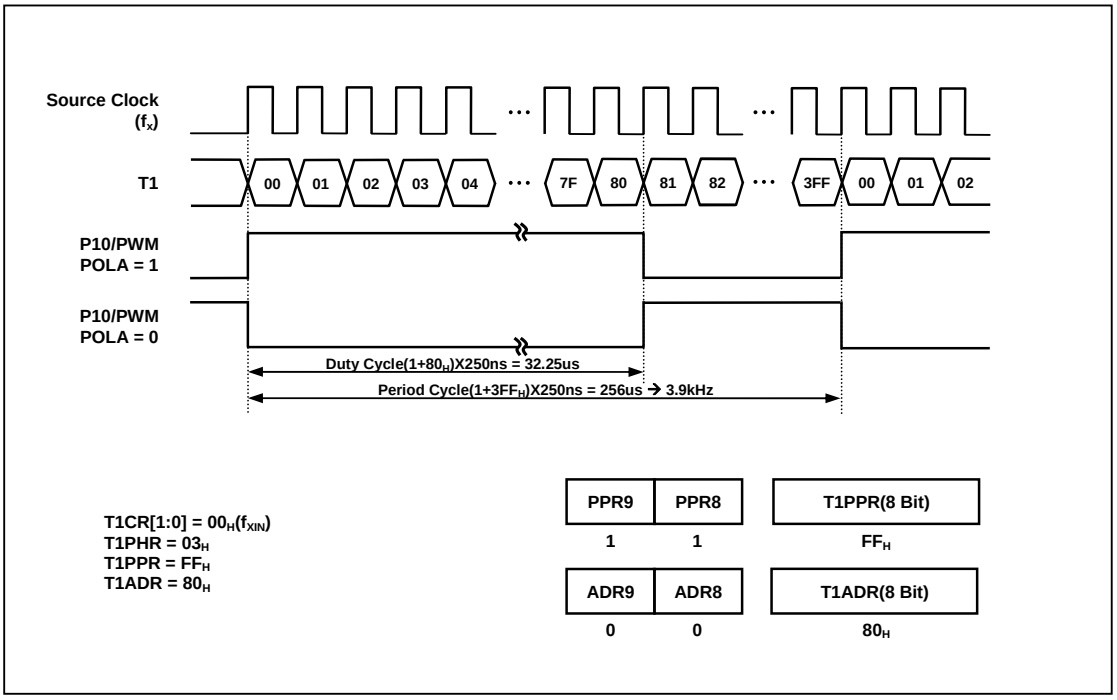



Figure 11-16 Example of PWM at 4MHz

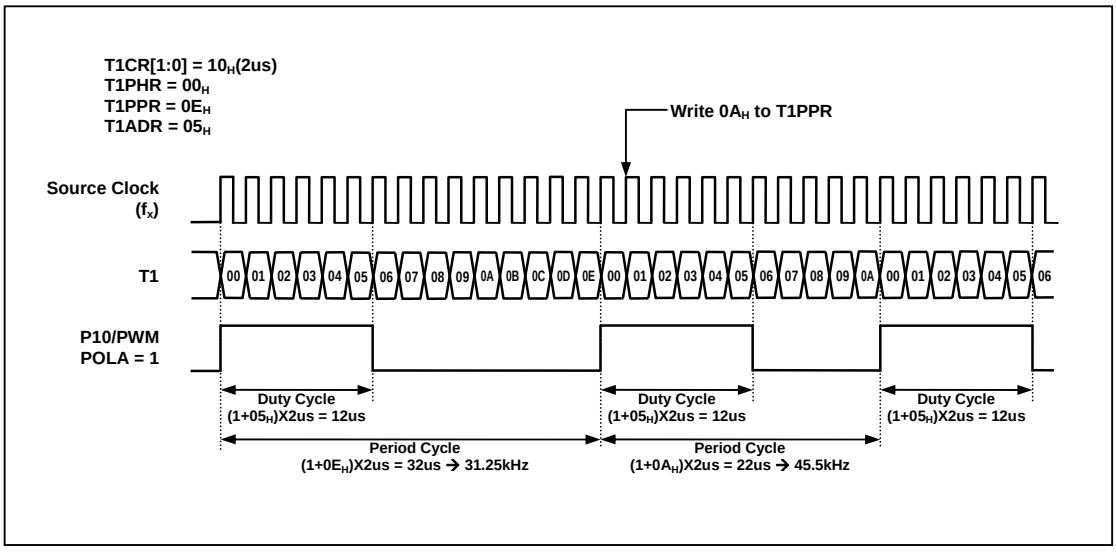



Figure 11-17 Example of Changing the Period in Absolute Duty Cycle at 4Mhz

**Update period & duty register value at once**

The period and duty of PWM comes to move from temporary registers to PPR (PWM Period Register) and PDR (PWM Duty Register) when always period match occurs. If you want that the period and duty is immediately changed, the UPDT bit in the T1PCR register must set to '1'. It should be noted that it needs the 3 cycle of timer clock for data transfer in the internal clock synchronization circuit. So the update data is written before 3 cycle of timer clock to get the right output waveform.

**Phase correction & Frequency correction**

On operating PWM, it is possible that it is changed the phase and the frequency by using BMOD bit (back-to-back mode) in T1PCR register. (Figure 11-18, Figure 11-19, Figure 11-20 referred)

In the back-to-back mode, the counter of PWM repeats up/down count. In fact, the effective duty and period becomes twofold of the register set values. (Figure 11-18, Figure 11-19 referred)

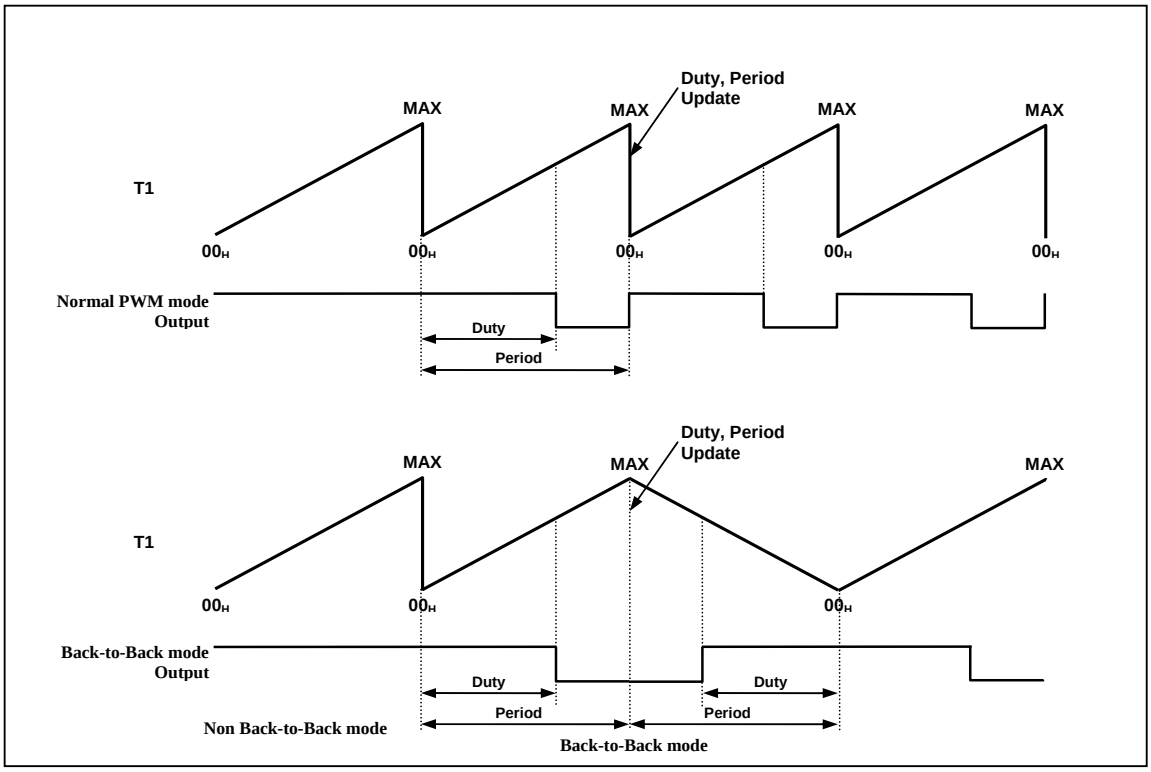



Figure 11-18 Example of PWM Output Waveform

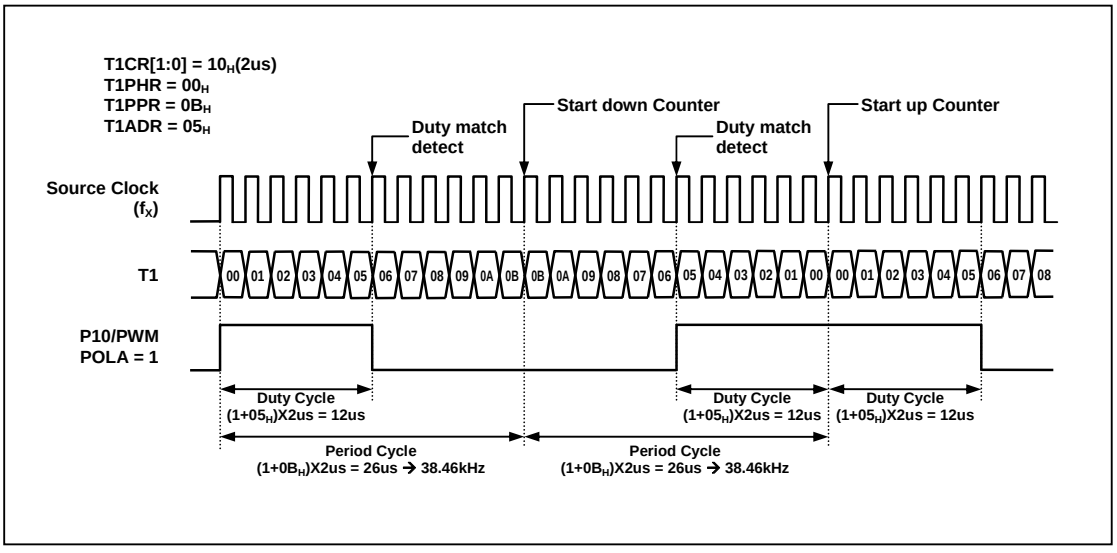



Figure 11-19 Example of PWM waveform in Back-to-Back mode at 4Mhz

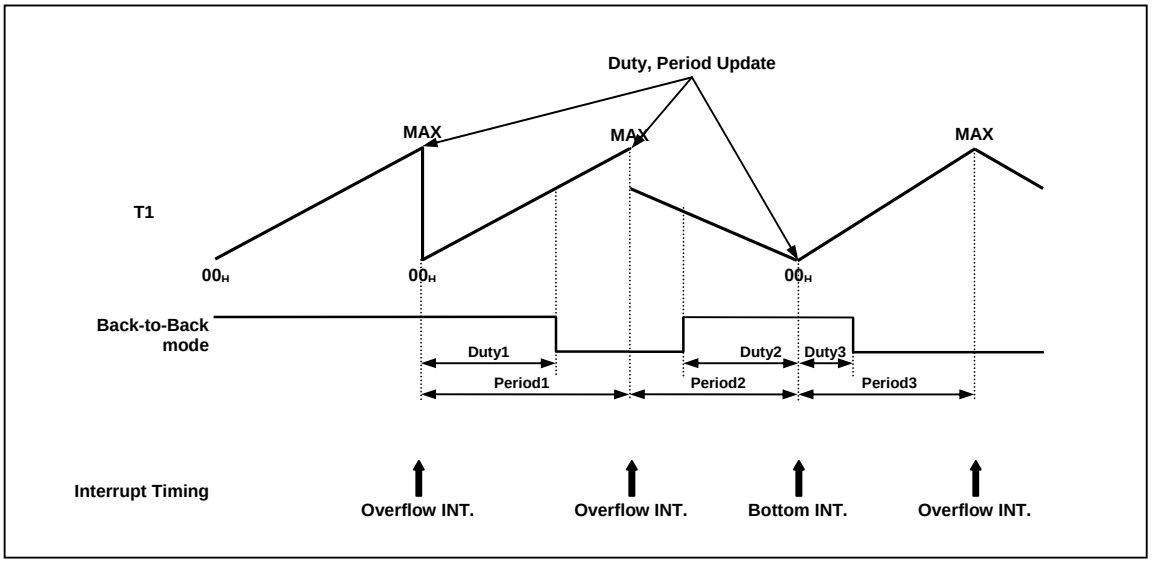



Figure 11-20 Example of Phase Correction and Frequency correction of PWM

### External Sync

If using ESYNC bit of T1PCR register, it is possible to synchronize the output of PWM from external signal.

If ESYNC bit sets to '1', the external signal moves to PWM module through the BLNKB pin (P16). If BLNKB signal is low, immediately PWM output becomes a reset value, and internal counter becomes reset. If BLNKB signal returns to '1', the counter is started again and PWM output is normally generated. (Figure 11-21 referred)

### PWM Halt

If using PHLT bit of T1PCR register, it is possible to stop PWM operation by the software. During PHLT bit being '1', PWM output becomes a reset value, and internal counter becomes reset as 0. Without changing PWM setting, temporarily it is able to stop PWM. In case of T1CN, when stopping counter, PWM output pin remains before states. But if PHLT bit sets to '1', PWM output pin has reset value.

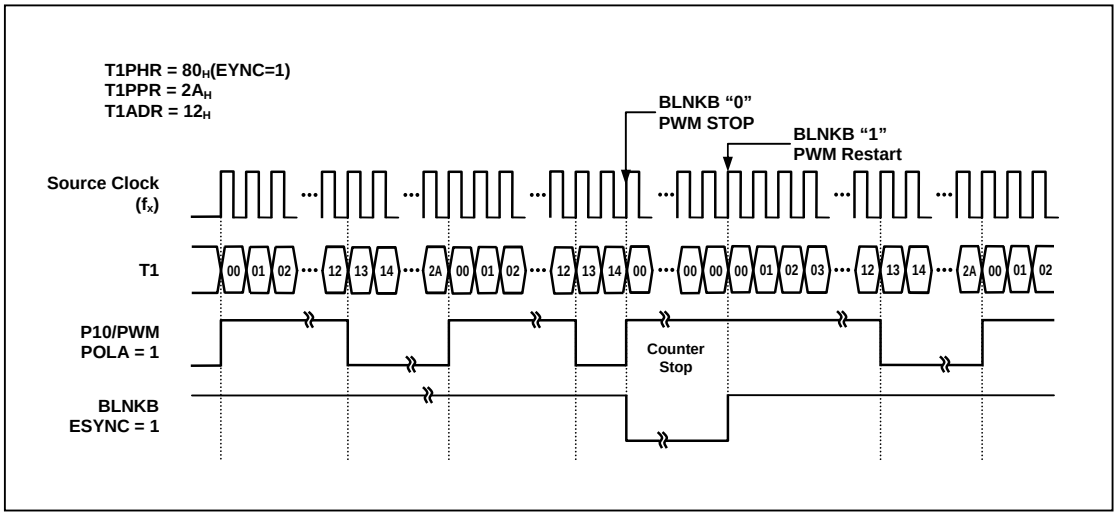


Figure 11-21 Example of PWM External Synchronization with BLNKB Input

**FORCE Drive ALL ch with A-ch mode**

If FORCA bit sets to '1', it is possible to enable or disable all PWM output pins through PWM outputs which occur from A-ch duty counter. It is noted that the inversion outputs of A, B, C channel have the same A-ch output waveform. According to POLA/B/C, it is able to control the inversion of outputs.

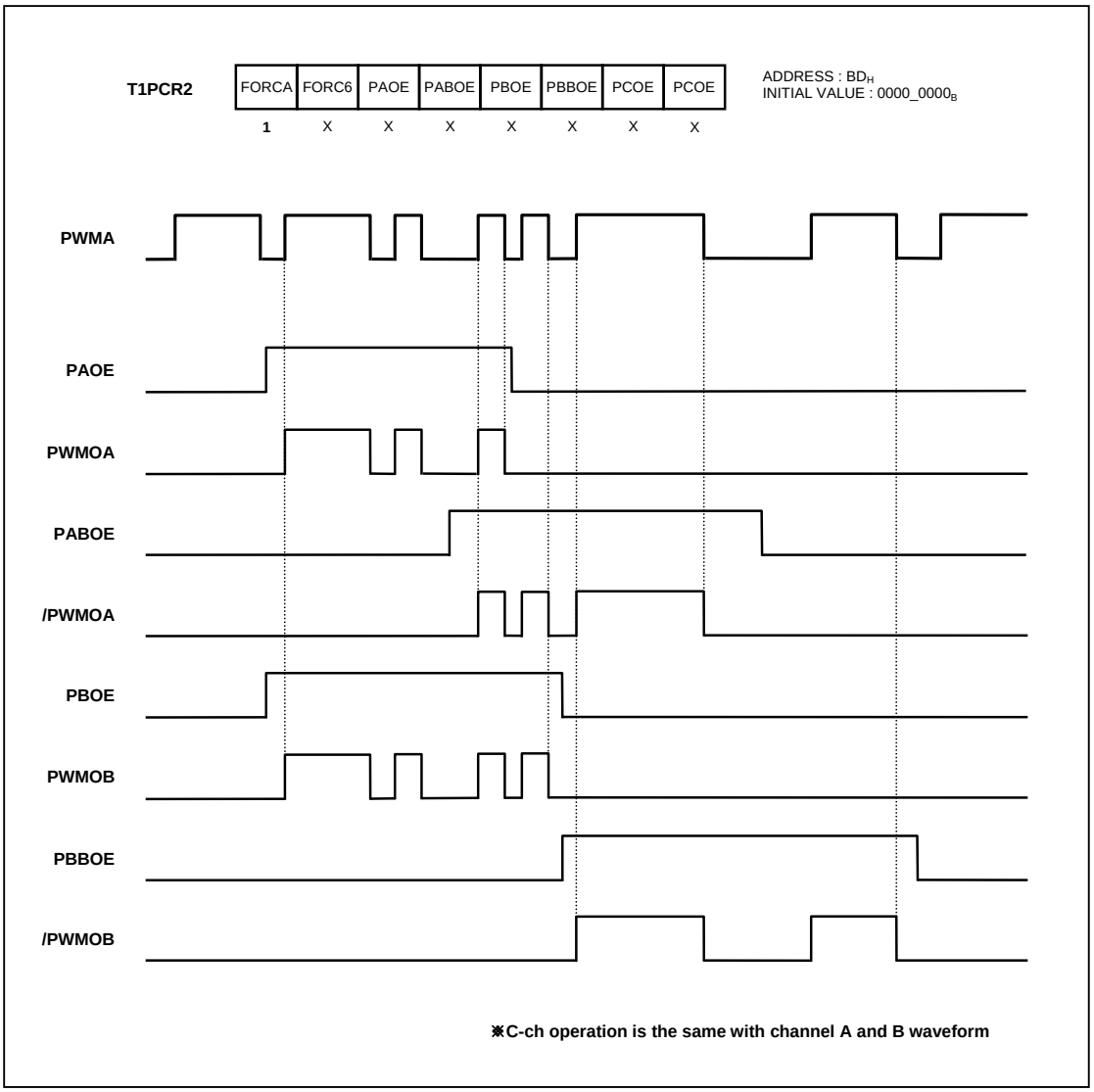



Figure 11-22 Example of Force Drive All-ch with A-ch

### FORCE 6-Ch Drive

If FORC6 bit sets to ‘1’, it is possible to enable or disable PWM output pin and inversion output pin generated through the duty counter of each channel. The inversion output is the reverse phase of the PWM output. A A/AB output of the A-channel duty register, a B/BB output of the B-channel duty register, a C/CB output of the C-channel duty register are controlled respectively. If the UALL bit is set to ‘1’, it is updated B/C channel duty at the same time, when it is written by a A-channel duty register.

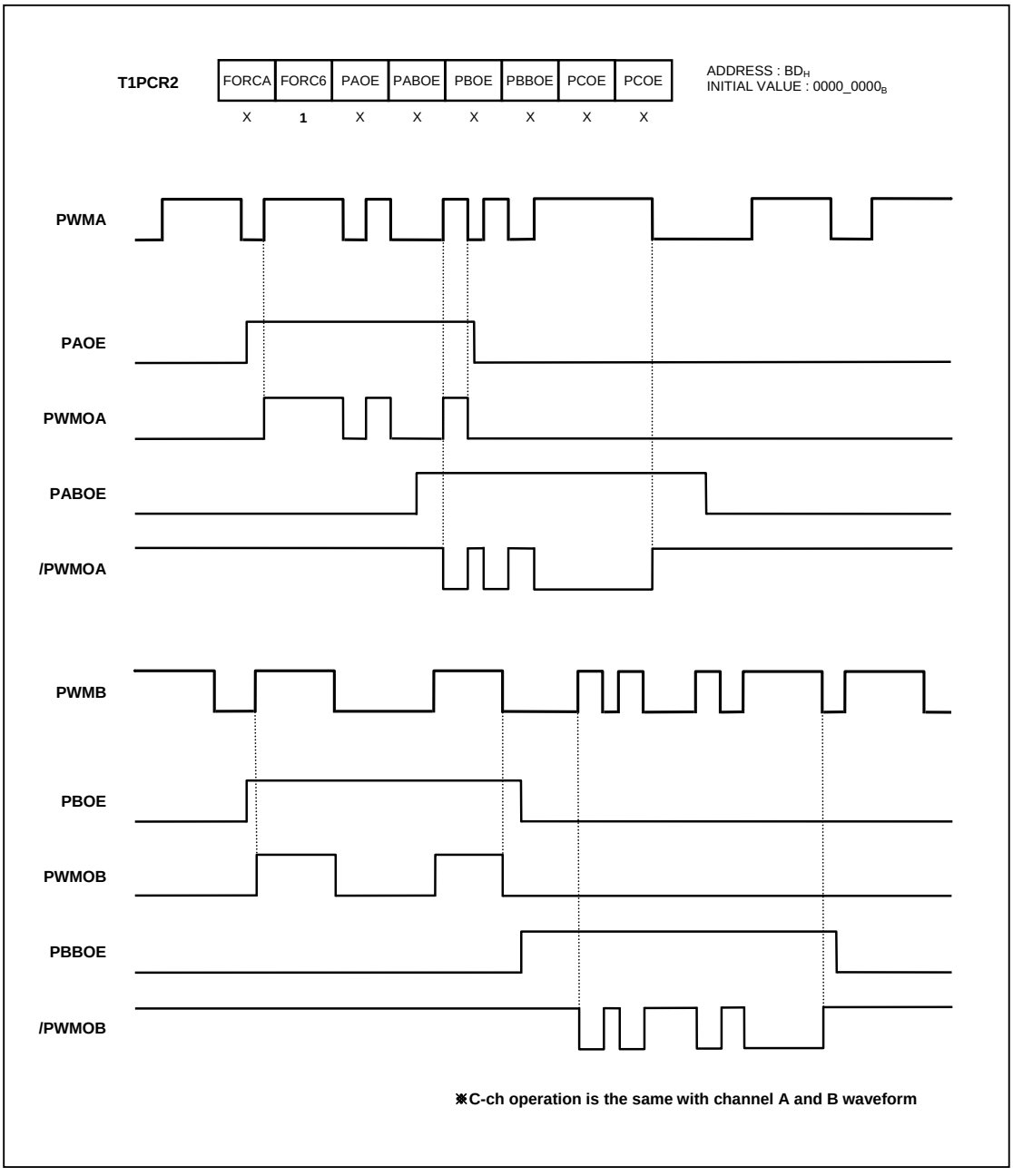



Figure 11-23 Example of Force Drive 6-ch Mode

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **T1CR** | 16BIT | CAP1 | T1CN | T1ST | T1CK3 | T1CK2 | T1CK1 | T1CK0 | ADDRESS : $B4_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | **0** | **0** | **X** | **X** | **X** | **X** | **X** | **X** | |
| **T1PCR** | PWM1E | ESYNC | BMOD | PHLT | UPDT | UALL | NOPS1 | NOPS0 | ADDRESS : $B7_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | |
| **T1PCR2** | FORCA | FORC6 | PAOE | PABOE | PBOE | PBBOE | PCOE | PCOE | ADDRESS : $BD_H$<br>INITIAL VALUE : $0000_0000_B$ |
| | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | |
| **T1PCR3** | T1_PE | POLA | POLB | POLC | POCON | HCKE | - | PLLPDB | ADDRESS : $BE_H$<br>INITIAL VALUE : $0000_00\text{-}0_B$ |
| | **X** | **X** | **X** | **X** | **X** | **X** | **X** | **X** | |

| PWM1E | T1_PE | FORCA<br>FORC6 | POCON | P1xE<br>P1xBE | Port Control |
|---|---|---|---|---|---|
| 0 | 1 | X | X | X | T1 (timer1) output enable |
| 1 | 1 | 0 | 0 | X | All 6 PWM outputs enable |
| **1** | 1 | 0 | **1** | X | **GPIO get control, when PWM stop*** |
| 1 | X | 1 | 0 | 0 | Corresponding PWM output “0” |
| 1 | X | 1 | 0 | 1 | Corresponding PWM outputs enable |
| **1** | X | 1 | **1** | 0 | **GPIO get control for disabled channels** |
| **1** | X | 1 | **1** | 1 | Corresponding PWM outputs enable |

* PWM stop condition : when ESYNC = “1”, BLANKB can stop PWM1 or PHLT bit was set(“1”)

When POCON was set “1”, ports direction of the disabled PWM channels would be controlled by GPIO registers. So if user set GPIO direction register as input mode for corresponding pins, disabled PWM pins would be input mode.

The port is fixed as the output at the Timer mode. But at the PWM mode, using POCON bit , when PWM output port does not work, it can change the state of output port to input (high-Z) state. It is determined from the port control register to select in/out signal. If using FORCE mode, it can only change the wanted channel direction of the 6-channel outputs. In the FORCE mode, the channel direction of the disabled output is determined by each port control register bit, regardless of the PWM stop.

Figure 11-24 PWM Port control

### PWM output Delay

If using the PDLYA, PDLYB, PDLYC register, it can delay PWM output based on the rising edge. At that time, it does not change the falling edge, so the duty is reduced as the time delay. In POLA/B/C setting to ‘0’, the delay is applied to the falling edge. In POLA/B/C setting to ‘1’, the delay is applied to the rising edge. It can produce a pair of Non-overlapping clock. The each channel is able to have 4-bit delay. As it can select the clock up to 1/8 divided clock using NOPS1, NOPS0, the delay of its maximum 128 timer clock cycle is produced.

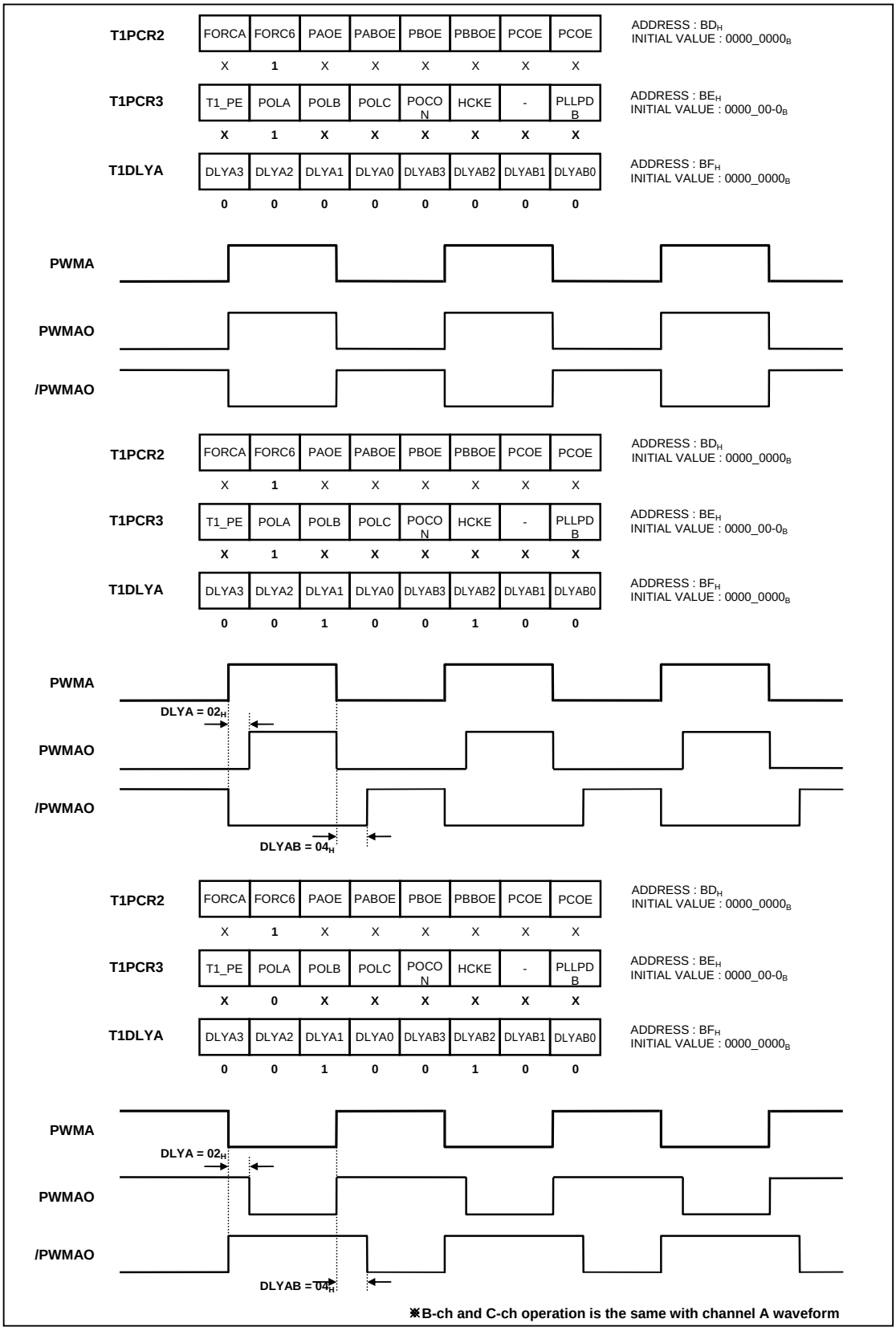



Figure 11-25 Example of PWM Delay

#### 11.5.1.7 8-Bit (16 Bit) Compare Output Mode

If the T1 (T0+T1) value and the T1DR (T0DR+T1DR) value are matched, T1/PWM1A port outputs. The output is 50:50 of duty square wave, the frequency is following

$$f_{COMP} = \frac{\text{Oscillator Frequency}}{2 \times \text{Prescaler Value} \times (TDR + 1)}$$

To export the compare output as T1/PWM1A, the T1_PE bit in the T1PCR3 register must set to '1'.

#### 11.5.1.8 Register Map

Table 11-7 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| T0CR | B2H | R/W | 00H | Timer 0 Mode Control Register |
| T0 | B3H | R | 00H | Timer 0 Register |
| T0DR | B3H | W | FFH | Timer 0 Data Register |
| CDR0 | B3H | R | 00H | Capture 0 Data Register |
| T1CR | B4H | R/W | 00H | Timer 1 Mode Control Register |
| T1DR | B5H | W | FFH | Timer 1 Data Register |
| T1PPR | B5H | W | FFH | Timer 1 PWM Period Register |
| T1 | B6H | R | 00H | Timer 1 Register |
| T1ADR | B6H | R/W | 7FH | Timer 1 PWM 1A Duty Register |
| CDR1 | B6H | R | 00H | Capture 1 Data Register |
| T1PCR | B7H | R/W | 00H | Timer 1 PWM Control Register |
| T1BDR | BAH | R/W | 7FH | Timer 1 PWM 1B Duty Register |
| T1CDR | BBH | R/W | 7FH | Timer 1 PWM 1C Duty Register |
| T1PHR | BCH | R/W | 00H | Timer 1 PWM High Register |
| T1PCR2 | BDH | R/W | 00H | Timer 1 PWM Control Register 2 |
| T1PCR3 | BEH | R/W | 00H | Timer 1 PWM Control Register 3 |
| T1DLYA | BFH | R/W | 00H | PWM1 Non-Overlap Delay Register ch. A/AB |
| T1DLYB | C2H | R/W | 00H | PWM1 Non-Overlap Delay Register ch. B/BB |
| T1DLYC | C3H | R/W | 00H | PWM1 Non-Overlap Delay Register ch. C/CB |
| T1ISR | C4H | R/W | 00H | Timer 1 Interrupt Status Register |
| T1IMSK | C5H | R/W | 00H | Timer 1 Interrupt Mask Register |
| PLLCR | 85H | R/W | 42H | Timer1 PLL Control Register |

#### 11.5.1.9 Timer/Counter 0, 1 Register description

The Timer/Counter 0,1 register consists of Timer 0 Mode Control Register (T0CR), Timer 0 Register (T0), Timer 0 Data Register (T0DR), Capture 0 Data Register (CDR0), Timer 1 Mode Control Register (T1CR), Timer 1 Data Register (T1DR), Timer 1 PWM Period Register (T1PPR), Timer 1 Register (T1), Timer 1 PWM 1A Duty Register (T1ADR), Capture 1 Data Register (CDR1), Timer 1 PWM Control Register (T1PCR), Timer 1 PWM 1B Duty Register (T1BDR), Timer 1 PWM 1C Duty Register

(T1CDR), Timer 1 PWM High Register (T1PHR), Timer 1 PWM Control Register 2 (T1PCR2), Timer 1 PWM Control Register 3 (T1PCR3), PWM1 Non-Overlap Delay Register ch. A/AB (T1DLYA), PWM1 Non-Overlap Delay Register ch. B/BB (T1DLYB), PWM1 Non-Overlap Delay Register ch. C/CB (T1DLYC), Timer 1 Interrupt Status Register (T1ISR), Timer 1 Interrupt Mask Register (T1IMSK) and PLL Control Register (PLLCR)

#### 11.5.1.10 Register description for Timer/Counter 0, 1

**T0CR (Timer 0 Mode Control Register) : B2H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T0EN | T0_PE | CAP0 | T0CK2 | T0CK1 | T0CK0 | T0CN | T0ST |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**T0EN** Control Timer 0
- 0 Timer 0 disable
- 1 Timer 0 enable

**T0_PE** Control Timer 0 Output port
- 0 Timer 0 Output disable
- 1 Timer 0 Output enable

**CAP0** Control Timer 0 operation mode
- 0 Timer/Counter mode
- 1 Capture mode

**T0CK[2:0]** Select Timer 0 clock source. Fx is main system clock frequency

| T0CK2 | T0CK1 | T0CK0 | description |
|---|---|---|---|
| 0 | 0 | 0 | fx/2 |
| 0 | 0 | 1 | fx/4 |
| 0 | 1 | 0 | fx/8 |
| 0 | 1 | 1 | fx/32 |
| 1 | 0 | 0 | fx/128 |
| 1 | 0 | 1 | fx/512 |
| 1 | 1 | 0 | fx/2048 |
| 1 | 1 | 1 | External Clock (EC0) |

**T0CN** Control Timer 0 Count pause/continue
- 0 Temporary count stop
- 1 Continue count

**T0ST** Control Timer 0 start/stop
- 0 Counter stop
- 1 Clear counter and start

**T0 (Timer 0 Register: Read Case) : B3H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T07 | T06 | T05 | T04 | T03 | T02 | T01 | T00 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**T0[7:0]** T0 Counter

### T0DR (Timer 0 Data Register: Write Case) : B3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T0D7 | T0D6 | T0D5 | T0D4 | T0D3 | T0D2 | T0D1 | T0D0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**T0D[7:0]** T0 Compare

### CDR0 (Capture 0 Data Register: Read Case, Capture mode only) : B3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CDR07 | CDR06 | CDR05 | CDR04 | CDR03 | CDR02 | CDR01 | CDR00 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**CDR0[7:0]** T0 Capture

### T1CR (Timer 1 Mode Count Register) : B4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 16BIT | CAP1 | T1CN | T1ST | T1CK3 | T1CK2 | T1CK1 | T1CK0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**16BIT** Select Timer 1 8/16Bit
- 0 8 Bit
- 1 16 Bit

**CAP1** Control Timer 1 operation mode
- 0 Timer/Counter mode
- 1 Capture mode

**T1CN** Control Timer 1 Count pause/continue
- 0 Temporary count stop
- 1 Continue count

**T1ST** Control Timer 1 start/stop
- 0 Counter stop
- 1 Clear counter and start

**T1CK[3:0]** Select Timer 1 clock source. Fx is main system clock frequency

| T1CK3 | T1CK2 | T1CK1 | T1CK0 | description |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | fx |
| 0 | 0 | 0 | 1 | fx/2 |
| 0 | 0 | 1 | 0 | fx/4 |
| 0 | 0 | 1 | 1 | fx/8 |
| 0 | 1 | 0 | 0 | fx/16 |
| 0 | 1 | 0 | 1 | fx/32 |
| 0 | 1 | 1 | 0 | fx/64 |
| 0 | 1 | 1 | 1 | fx/128 |
| 1 | 0 | 0 | 0 | fx/256 |
| 1 | 0 | 0 | 1 | fx/512 |
| 1 | 0 | 1 | 0 | fx/1024 |
| 1 | 0 | 1 | 1 | fx/2048 |
| 1 | 1 | 0 | 0 | fx/4096 |

| | | | | |
|---|---|---|---|---|
| 1 | 1 | 0 | 1 | fx/8192 |
| 1 | 1 | 1 | 0 | fx/16384 |
| 1 | 1 | 1 | 1 | Using Timer 0 Clock |

Note) If you want to use “Using Timer 0 Clock”, you can set T0EN bit in T0CR

**T1DR (Timer 1 Data Register: Write Case) : B5H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1D7 | T1D6 | T1D5 | T1D4 | T1D3 | T1D2 | T1D1 | T1D0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**T1D[7:0]** T1 Compare

**T1PPR (Timer 1 PWM Period Register: Write Case PWM mode only) : B5H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1PP7 | T1PP6 | T1PP5 | T1PP4 | T1PP3 | T1PP2 | T1PP1 | T1PP0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**T1PP[7:0]** T1 PWM period

**T1 (Timer 1 Register: Read Case) : B6H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T17 | T16 | T15 | T14 | T13 | T12 | T11 | T10 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**T1[7:0]** T1 Counter period

**T1ADR (Timer 1 PWM 1A Duty Register PWM mode only) : B6H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PAD7 | PAD6 | PAD5 | PAD4 | PAD3 | PAD2 | PAD1 | PAD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 7FH

**T1ADR[7:0]** T1 PWM Duty
Note) only write, when PWM1E ‘1’

**CDR1 (Capture 1 Data Register: Read Case, Capture mode only) : B6H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CDR17 | CDR16 | CDR15 | CDR14 | CDR13 | CDR12 | CDR11 | CDR10 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**CDR1[7:0]** T1 Capture

**T1PCR (Timer 1 PWM Control Register) : B7H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PWM1E | ESYNC | BMOD | PHLT | UPDT | UALL | NOPS1 | NOPS0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**PWM1E** Control PWM
- 0 PWM disable
- 1 PWM enable

**ESYNC** Select the operation of External Sync Mode
- 0 External Sync Mode disable
- 1 External Sync Mode enable (using with BLNKB(P16))

**BMOD** Control Back-To-Back Mode operation
- 0 BtB mode disable (only up count)
- 1 BtB mode enable (Up/Down count)

**PHLT** Control PWM
- 0 PWM running
- 1 PWM stop

**UPDT** Determine the update time of PPR, PDR
- 0 Update at period match
- 1 Update at any time (after 3 timer clock, update)

**UALL** Control update all duty register
- 0 Write duty register separately
- 1 Write all duty registers (via A duty)

**NOPS1[1:0]** Select on-Overlap prescaler
Note) fpwm: PWM operation clock frequency

| NOPS1 | NOPS0 | description |
|---|---|---|
| 0 | 0 | fpwm |
| 0 | 1 | fpwm/2 |
| 1 | 0 | fpwm/4 |
| 1 | 1 | fpwm/8 |

### T1BDR (Timer 1 PWM 1B Duty Register) : BAH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PBD7 | PBD6 | PBD5 | PBD4 | PBD3 | PBD2 | PBD1 | PBD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 7FH

**T1BDR[7:0]** PWM 1B ch Duty
Note) only write, when PWM1E '1'

### T1CDR (Timer 1 PWM 1C Duty Register) : BBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PCD7 | PCD6 | PCD5 | PCD4 | PCD3 | PCD2 | PCD1 | PCD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 7FH

**T1CDR[7:0]** PWM 1C ch Duty
Note) only write, when PWM1E '1'

### T1PHR (Timer 1 PWM High Register) : BCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ADR9 | ADR8 | BDR9 | BDR8 | CDR9 | CDR8 | PPR9 | PPR8 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**ADR[9:8]** PWM 1A High (Bit [9:8])
**BDR[9:8]** PWM 1B High (Bit [9:8])
**CDR[9:8]** PWM 1C High (Bit [9:8])
**PPR[9:8]** PERIOD High (Bit [9:8])

| | | | |
|---|---|---|---|
| PERIOD: | PPR9 | PPR8 | T1PPR[7:0] |
| DUTY A: | ADR9 | ADR8 | P1ADR[7:0] |
| DUTY B: | BDR9 | BDR8 | P1BDR[7:0] |
| DUTY C: | CDR9 | CDR8 | P1CDR[7:0] |

### T1PCR2 (Timer 1 PWM Control Register 2) : BDH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FORCA | FORC6 | PAOE | PABOE | PBOE | PBBOE | PCOE | PCBOE |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**FORCA** Control Force Drive A Channel mode
0 Force Drive A Channel mode disable
1 Force Drive A Channel mode enable

**FORC6** Control Force 6 Channel mode
Note) PAOE~PCBOE is effective when FORC6 sets to '1'
0 Force 6 Channel mode disable
1 Force 6 Channel mode enable

**PAOE/ PABOE** Select Channel A/AB operation
0 P1A (or P1AB) output disable
1 P1A (or P1AB) output enable

**PBOE/ PBBOE** Select Channel B/BB operation
0 P1B (or P1BB) output disable
1 P1B (or P1BB) output enable

| | | |
|---|---|---|
| **PCOE/ PCBOE** | Select Channel C/CB operation | |
| | 0 | P1C (or P1CB) output disable |
| | 1 | P1C (or P1CB) output enable |

### T1PCR3 (Timer 1 PWM Control Register 3) : BEH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1_PE | POLA | POLB | POLC | POCON | HCKE | - | PLLPDB |
| R/W | R/W | R/W | R/W | R/W | R/W | - | R/W |

Initial value : 00H

| | | |
|---|---|---|
| **T1_PE** | Control Timer1/PWM1 Output port | |
| | 0 | T1, PWM1 Output operation disable |
| | 1 | T1, PWM1 Output operation enable |
| **POLA** | Configure PWM A-ch polarity | |
| | 0 | Negative (Duty Match time, Clear) |
| | 1 | Positive (Duty Match time, Set) |
| **POLB** | Configure PWM B-ch polarity | |
| | 0 | Negative (Duty Match time, Clear) |
| | 1 | Positive (Duty Match time, Set) |
| **POLC** | Configure PWM C-ch polarity | |
| | 0 | Negative (Duty Match time, Clear) |
| | 1 | Positive (Duty Match time, Set) |
| **POCON** | Control PWM output operation | |
| | 0 | PWM output control disable |
| | 1 | PWM output control enable |
| **HCKE** | Select High frequency<br>Note) fCK is system frequency, Fout is PLL output frequency | |
| | 0 | High frequency disable |
| | 1 | High frequency enable (Fout > 3* fCK) |
| **PLLPDB** | Control PLL power Down mode | |
| | 0 | PLL disable (power down mode) |
| | 1 | PLL enable (for stable, needs 1ms wait) |

### T1DLYA (PWM1 Non-Overlap Delay Register for channel A/AB) : BFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DLYA3 | DLYA2 | DLYA1 | DLYA0 | DLYAB3 | DLYAB2 | DLYAB1 | DLYAB0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

| | |
|---|---|
| **DLYA[3:0]** | PWM A channel Output Delay (Rising edge only) |
| **DLYAB[3:0]** | PWM AB channel Output Delay (Rising edge only) |

### T1DLYB (PWM1 Non-Overlap Delay Register for channel B/BB) : C2H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DLYB3 | DLYB2 | DLYB1 | DLYB0 | DLYBB3 | DLYBB2 | DLYBB1 | DLYBB0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**DLYB[3:0]** PWM B channel Output Delay (Rising edge only)

**DLYBB[3:0]** PWM BB channel Output Delay (Rising edge only)

### T1DLYC (PWM1 Non-Overlap Delay Register for channel C/CB) : C3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DLYC3 | DLYC2 | DLYC1 | DLYC0 | DLYCB3 | DLYCB2 | DLYCB1 | DLYCB0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**DLYC[3:0]** PWM C channel Output Delay (Rising edge only)

**DLYCB[3:0]** PWM CB channel Output Delay (Rising edge only)

### T1ISR (Timer 1 Interrupt Status Register) : C4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IOVR | IBTM | ICMA | ICMB | ICMC | ICAP | - | - |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**IOVR** Overflow (match with T1DR in Timer mode or T1PPR in PWM mode) interrupt status
Note) for clear, write '1' to this bit
0 Overflow no occurrence
1 Overflow occurrence

**IBTM** Timer Bottom (goto zero) interrupt status in PWM Back-to-Back mode
Note) for clear, write '1' to this bit
0 Timer Bottom no occurrence
1 Timer Bottom occurrence

**ICMA** PWM A-ch Duty Match interrupt status
Note) for clear, write '1' to this bit
0 PWM A-ch Duty Match no occurrence
1 PWM A-ch Duty Match occurrence

**ICMB** PWM B-ch Duty Match interrupt status
Note) for clear, write '1' to this bit
0 PWM B-ch Duty Match no occurrence
1 PWM B-ch Duty Match occurrence

**ICMC** PWM C-ch Duty Match interrupt status
Note) for clear, write '1' to this bit
0 PWM C-ch Duty Match no occurrence
1 PWM C-ch Duty Match occurrence

**ICAP** Timer Capture event interrupt status
Note) for clear, write '1' to this bit
0 Timer Capture event no occurrence
1 Timer Capture event occurrence

### T1MSK (Timer 1 Interrupt Mask Register) : C5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OVRMSK | BTMMSK | CMAMSK | CMBMSK | CMCMSK | CAPMSK | - | - |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**OVRMSK** Control Overflow interrupt
- 0 Overflow interrupt disable
- 1 Overflow interrupt enable

**BTMMSK** Control Timer Bottom interrupt
- 0 Timer Bottom interrupt disable
- 1 Timer Bottom interrupt enable

**CMAMSK** Control Timer Compare Match (or PWM A-ch Match) interrupt
- 0 Timer Compare Match (or PWM A-ch Match) interrupt disable
- 1 Timer Compare Match (or PWM A-ch Match) interrupt enable

**CMBMSK** Control PWM B-ch Match interrupt
- 0 PWM B-ch Match interrupt disable
- 1 PWM B-ch Match interrupt enable

**CMCMSK** Control PWM C-ch Match interrupt
- 0 PWM C-ch Match interrupt disable
- 1 PWM C-ch Match interrupt enable

**CAPMSK** Control Timer Capture event interrupt
- 0 Timer Capture event interrupt disable
- 1 Timer Capture event interrupt enable

### PLLCR (PLL Control Register) : 85H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | PLLCT6 | PLLCT5 | PLLCT4 | PLLCT3 | PLLCT2 | PLLCT1 | PLLCT0 |
| - | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 42H

**PLLCT[6:5]** Pre Scaler (divider) Control

| PLLCT6 | PLLCT5 | description |
|---|---|---|
| 0 | 0 | Div 1 |
| 0 | 1 | Div 2 |
| 1 | 0 | Div 4 |
| 1 | 1 | Div 8 |

**PLLCT[4]** Feedback Control

0 Div 64

1 Div 50

**PLLCT[3:1]** Post Scaler (divider) Control (FOUT=fvco/M)

| PLLCT3 | PLLCT2 | PLLCT1 | description |
|---|---|---|---|
| 0 | 0 | 0 | M=1 |
| 0 | 0 | 1 | M=2 |
| 0 | 1 | 0 | M=4 |
| 0 | 1 | 1 | M=5 |
| 1 | 0 | 0 | M=6 |
| 1 | 0 | 1 | M=8 |
| 1 | 1 | 0 | M=10 |
| 1 | 1 | 1 | M=16 |

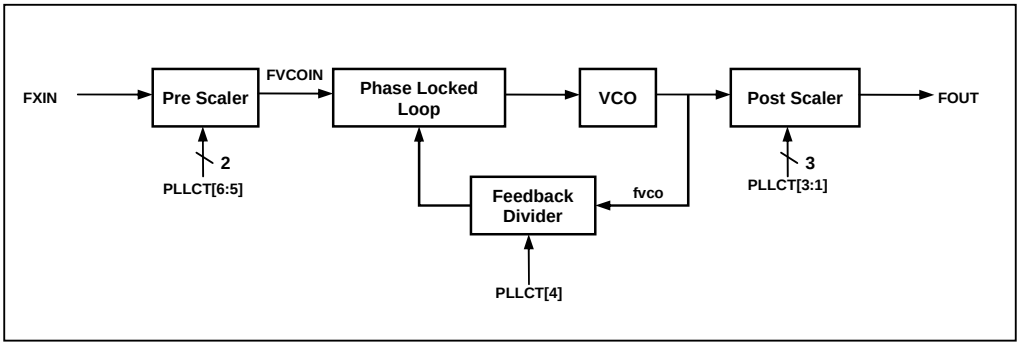
**PLLCT[0]** PLL enable (this bit should enable before 1ms for using PLL)

0 PLL disable

1 PLL enable

Note) FVCOIN value must be 2 MHz for desire FOUT.

To change PLL frequency during the operation, PLL must be disabled before XPLLCT change




FVCOIN = 2MHz (to be fixed) = FXIN / Pre-Divide

FVCO = FVCOIN * Feedback-Divider = 100 MHz or 128 MHz

FOUT = FVCO / Post-Divider

### 11.5.2 8-bit Timer/Event Counter 2, 3

#### 11.5.2.1 Overview

Timer 2 and timer 3 can be used either two 8-bit timer/counter or one 16-bit timer/counter with combine them. Each 8-bit timer/event counter module has multiplexer, 8-bit timer data register, 8-bit counter register, mode register, input capture register, comparator. For PWM, it has PWM register (T3PPR, T3PDR, T3PWHR).

It has seven operating modes:

- 8 Bit Timer/Counter Mode
- 8 Bit Capture Mode
- 8 Bit Compare Output Mode
- 16 Bit Timer/Counter Mode
- 16 Bit Capture Mode
- 16 Bit Compare Output Mode
- PWM Mode

The timer/counter can be clocked by an internal or external clock source (external EC2). The clock source is selected by clock select logic which is controlled by the clock select (T2CK[2:0], T3CK[1:0]).

- TIMER2 clock source : fX/1, 2, 4, 64, 256, 1024, 4096, EC2

- TIMER3 clock source : fX/1, 2, 16, T2CK

In the capture mode, by INT2, INT3, the data is captured into Input Capture Register. The Timer 2 outputs the compare result to T2 port in 8/16-bit mode. Also the timer 3 outputs the result T3 port in the timer mode and the PWM waveform to PWM3 in the PWM mode.

Table 11-8 Operating Modes of Timer

| 16 Bit | CAP2 | CAP3 | PWM3E | T2CK[2:0] | T3CK[1:0] | T2/3_PE | Timer 2 | Timer 3 |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | XXX | XX | 00 | 8 Bit Timer | 8 Bit Timer |
| 0 | 0 | 1 | 0 | 111 | XX | 00 | 8 Bit Event Counter | 8 Bit Capture |
| 0 | 1 | 0 | 0 | XXX | XX | 01 | 8 Bit Capture | 8 Bit Compare Output |
| 0 | 0 | 0 | 1 | XXX | XX | 11 | 8 Bit Timer/Counter | 10 Bit PWM |
| 1 | 0 | 0 | 0 | XXX | 11 | 00 | 16 Bit Timer | |
| 1 | 0 | 0 | 0 | 111 | 11 | 00 | 16 Bit Event Counter | |
| 1 | 1 | 1 | 0 | XXX | 11 | 00 | 16 Bit Capture | |
| 1 | 0 | 0 | 0 | XXX | 11 | 01 | 16 Bit Compare Output | |

### 11.5.2.2 8-Bit Timer/Counter Mode

The 8-bit Timer/Counter Mode is selected by control registers as shown in Figure 11-26.

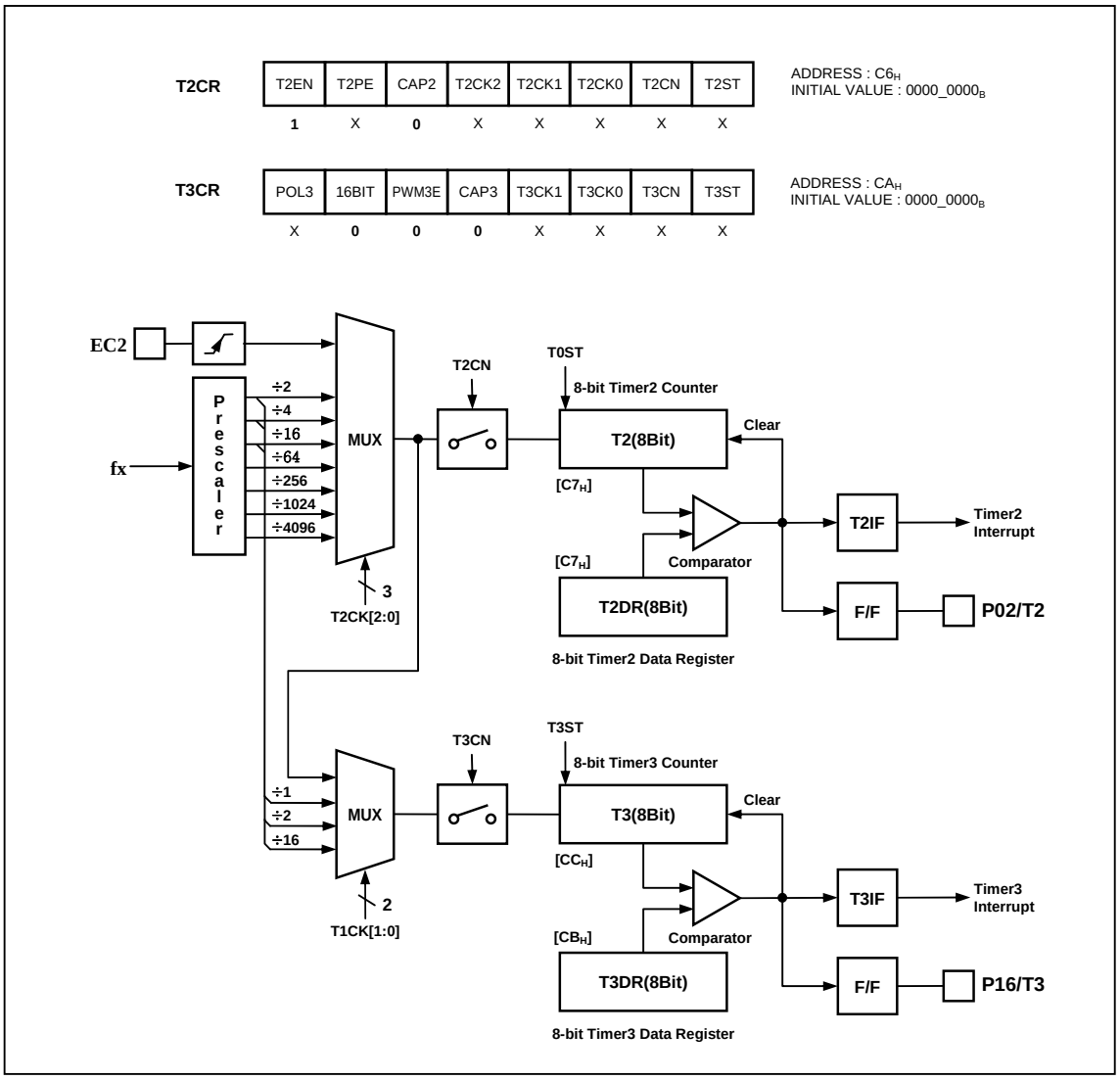



Figure 11-26 8 Bit Timer/Event Counter2, 3 Block Diagram

The two 8-bit timers have each counter and data register. The counter register is increased by internal or external clock input. The timer 2 can use the input clock with 2, 4, 16, 64, 256, 1024, 4096 prescaler division rates (T2CK[2:0]). The timer 3 can use the input clock with 1, 2, 16 and timer 2 overflow clock (T3CK[1:0]). When the value of T2, 3value and the value of T2DR, T3DR are respectively identical in Timer 2, 3, the interrupt of timer T2, 3 occurs. The external clock (EC2) counts up the timer at the rising edge. If EC2 is selected from T2CK[2:0], EC2 port becomes input port. The timer 3 can't use the external EC clock.

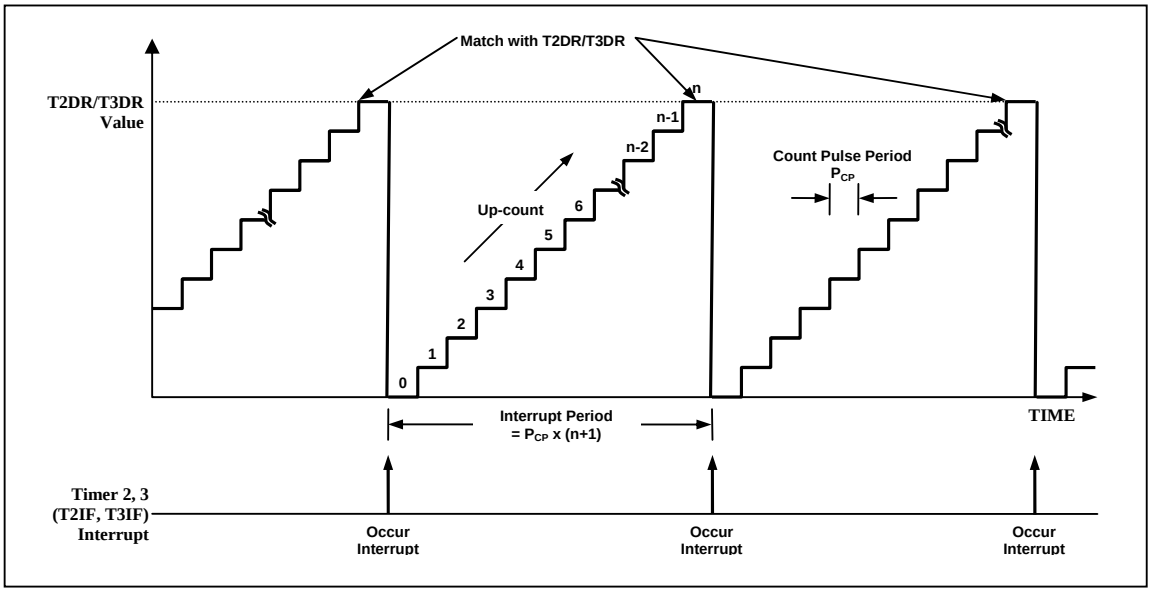



Figure 11-27 Timer/Event Counter2, 3 Example

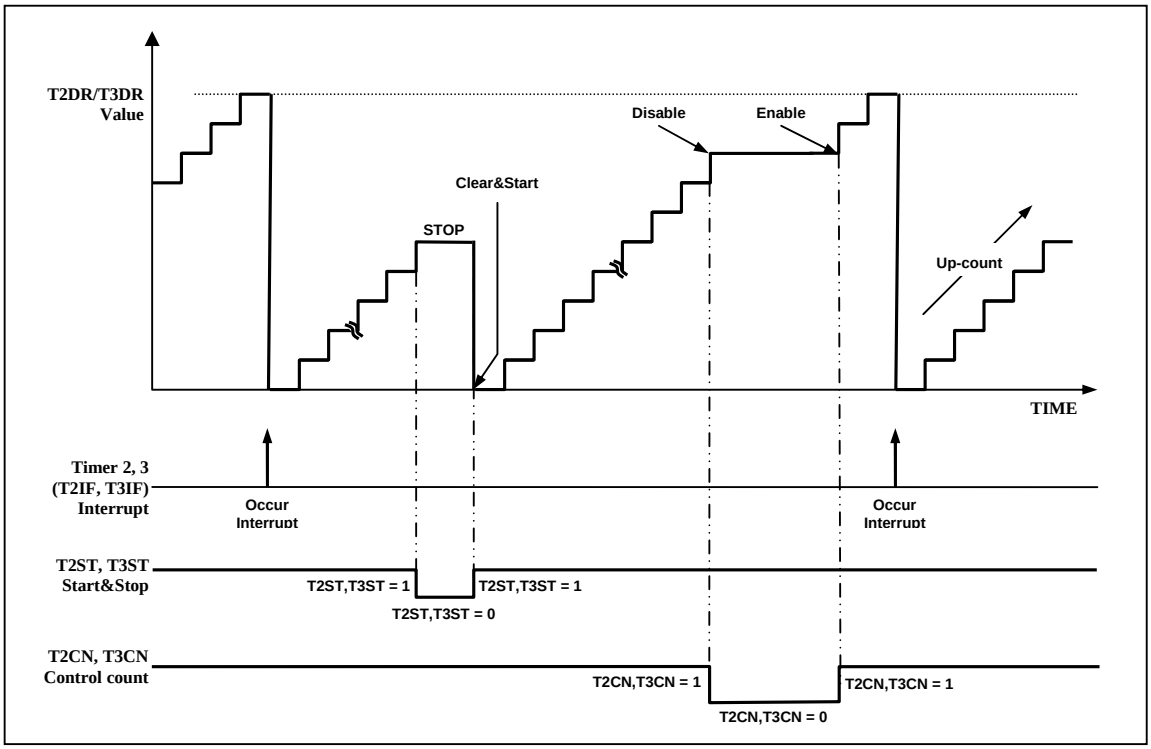



Figure 11-28 Operation Example of Timer/Event Counter2, 3

### 11.5.2.3 16-Bit Timer/Counter Mode

The timer register is being run with all 16bits. A 16-bit timer/counter register T2, T3 are incremented from 0003H to FFFFH until it matches T2DR, T3DR and then resets to 0000H. the match output

generates the Timer 2 interrupt ( no timer 3 interrupt). The clock source is selected from T2CK[2:0] and T3CK[1:0] must set 11b and 16BIT bit must set to '1'. The timer 2 is LSB 8-bit, the timer 3 is MSB 8-bit. T2DR must not be 0x00(0x01~0xFF). The 16-bit mode setting is shown as Figure 11-29.

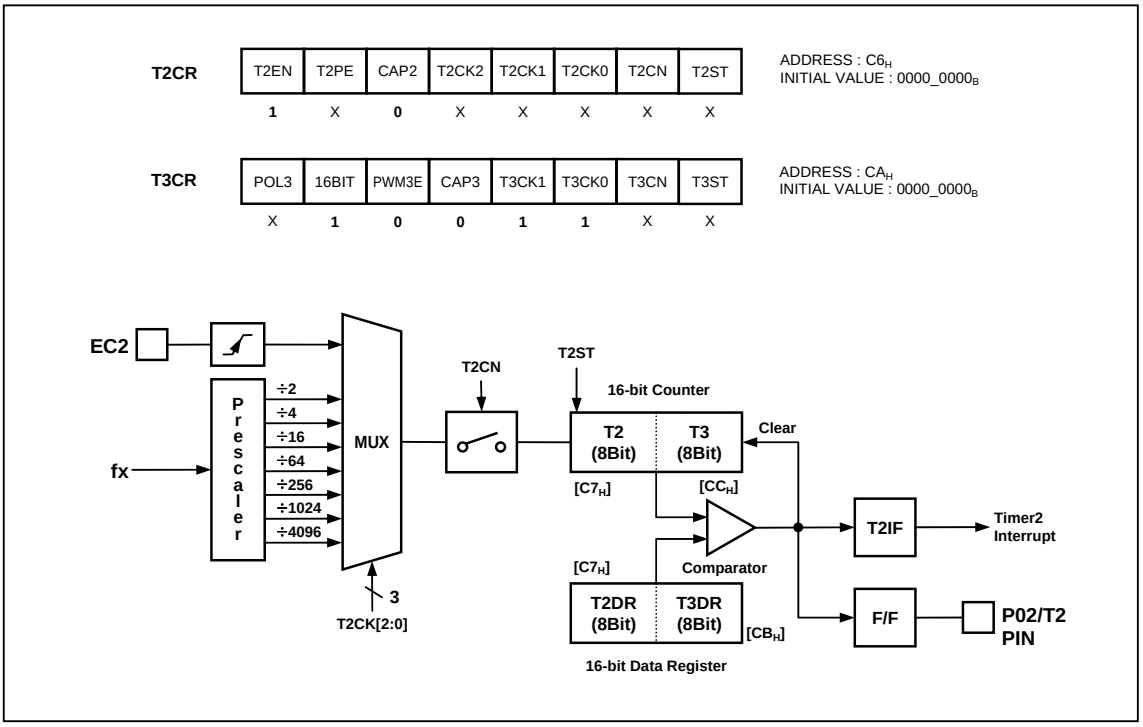



Figure 11-29 16 Bit Timer/Event Counter2, 3 Block Diagram

### 11.5.2.4 8-Bit Capture Mode

The timer 2, 3 capture mode is set by CAP2, CAP3 as '1'. The clock source can use the internal/external clock. Basically, it has the same function of the 8-bit timer/counter mode and the interrupt occurs at T 2, 3 and T2DR, T3DR matching time, respectively. The capture result is loaded into CDR2, CDR3. The T2, T3 value is automatically cleared by hardware and restarts counter.

This timer interrupt in capture mode is very useful when the pulse width of captured signal is wider than the maximum period of timer.

As the EIEDGE and EIPOLA register setting, the external interrupt INT2, INT3 function is chosen.

The CDR2, T2 and T2DR are in same address. In the capture mode, reading operation is read the CDR2, not T2DR because path is opened to the CDR2. The CDR3 has the same function.

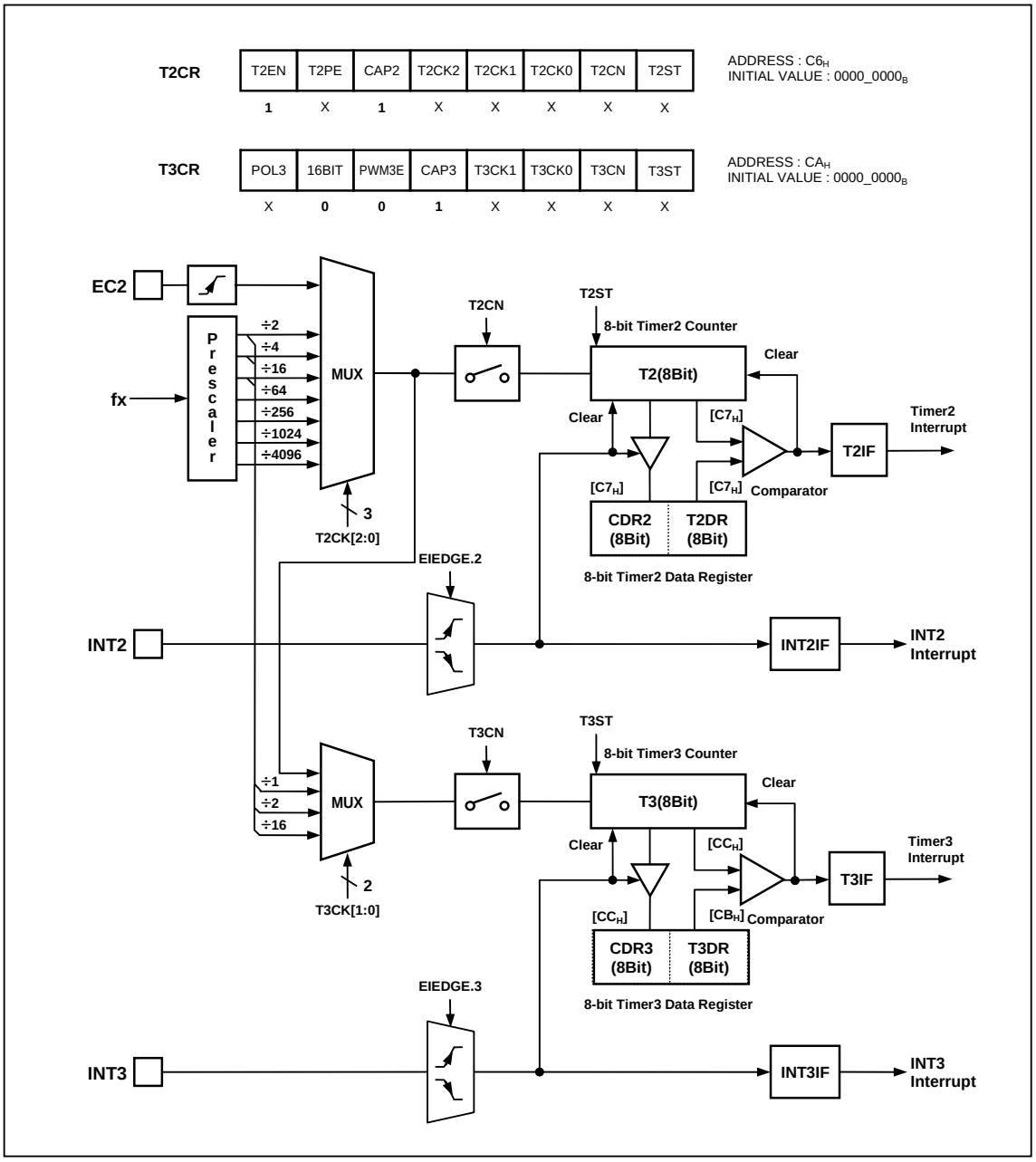



Figure 11-30 8-bit Capture Mode for Timer2, 3

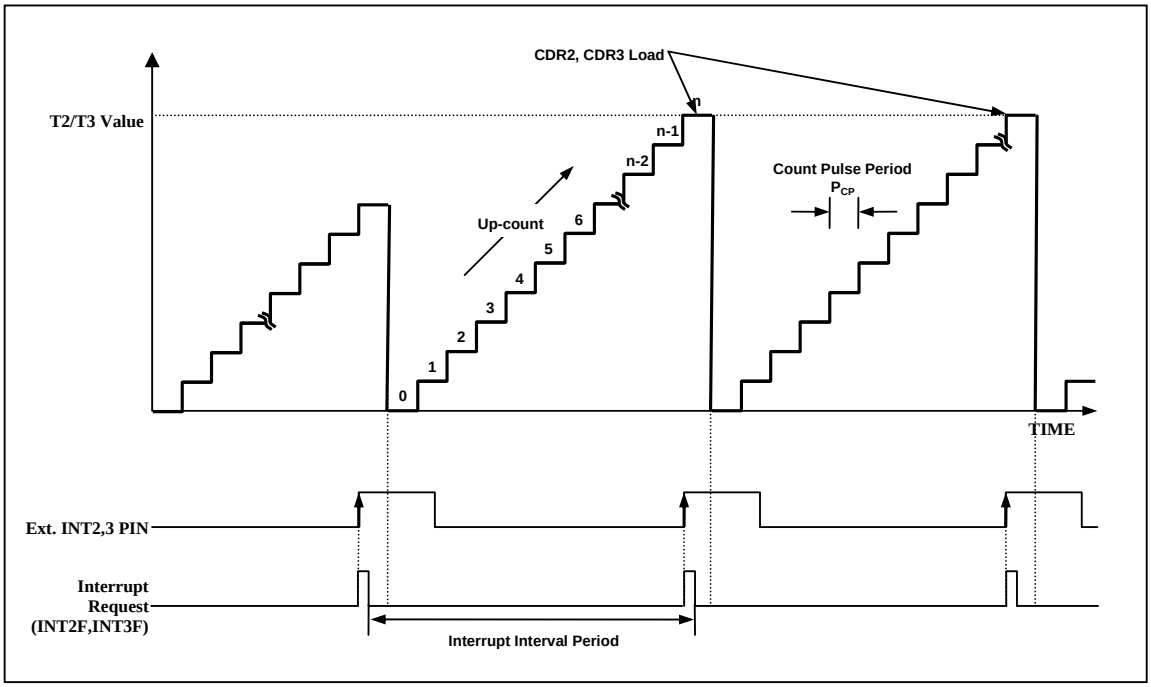



Figure 11-31 Input Capture Mode Operation of Timer 2, 3

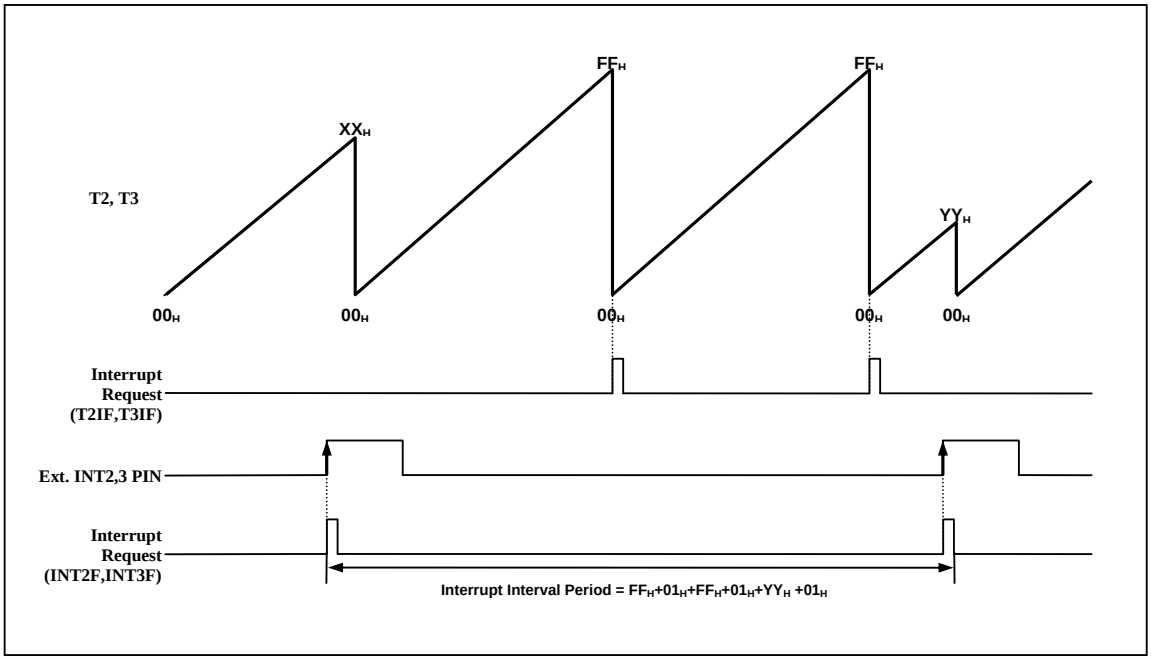



Figure 11-32 Express Timer Overflow in Capture Mode

### 11.5.2.5 16-Bit Capture Mode

The 16-bit capture mode is the same operation as 8-bit capture mode, except that the timer register uses 16 bits.

The clock source is selected from T2CK[2:0] and T3CK[1:0] must set 11b and 16BIT2 bit must set to '1'. The 16-bit mode setting is shown as Figure 11-33.

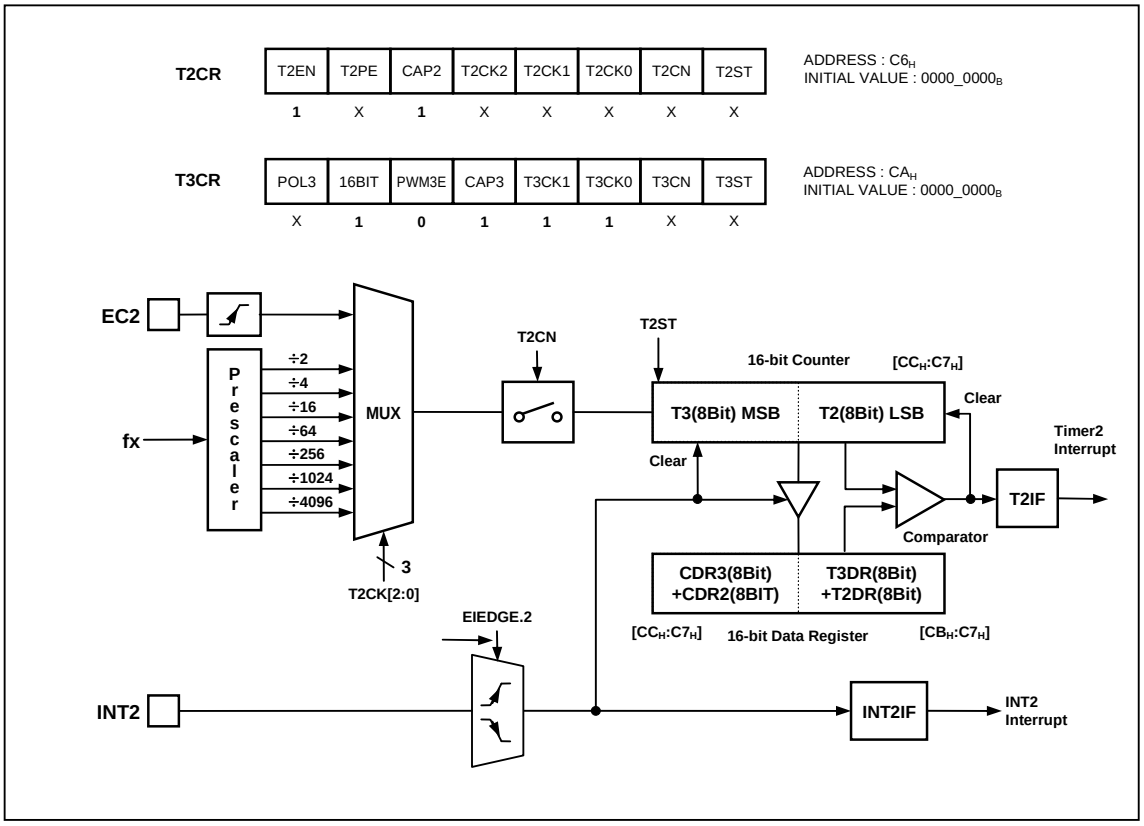



Figure 11-33 16-bit Capture Mode of Timer 2, 3

### 11.5.2.6 PWM Mode

The timer 3 has a PWM (pulse Width Modulation) function. In PWM mode, the T3/PWM3 output pin outputs up to 10-bit resolution PWM output. This pin should be configured as a PWM output by set T3_PE to '1'. The period of the PWM output is determined by the T3PPR (PWM period register) + T3PWHR[3:2] + T3PWHR[1:0]

PWM Period = [ T3PWHR[3:2]T3PPR ] X Source Clock
PWM Duty = [ T3PWHR[1:0] T3PDR ] X Source Clock

Table 11-9 PWM Frequency vs. Resolution at 8 Mhz

| Resolution | Frequency | | |
|---|---|---|---|
| | T3CK[1:0]=00 (125ns) | T3CK[1:0]=01 (250ns) | T3CK[1:0]=10 (2us) |
| 10 Bit | 7.8KHz | 3.9KHz | 0.49KHz |
| 9 Bit | 15.6KHz | 7.8KHz | 0.98KHz |
| 8 Bit | 31.2KHz | 15.6KHz | 1.95KHz |

| | | | |
|---|---|---|---|
| 7 Bit | 62.4KHz | 31.2KHz | 3.91KHz |

The POL bit of T3CR register decides the polarity of duty cycle. If the duty value is set same to the period value, the PWM output is determined by the bit POL (1: High, 0: Low). And if the duty value is set to "00H", the PWM output is determined by the bit POL (1: Low, 0: High). If duty value and period value are equal, PWM output is not retain high or low but toggle.

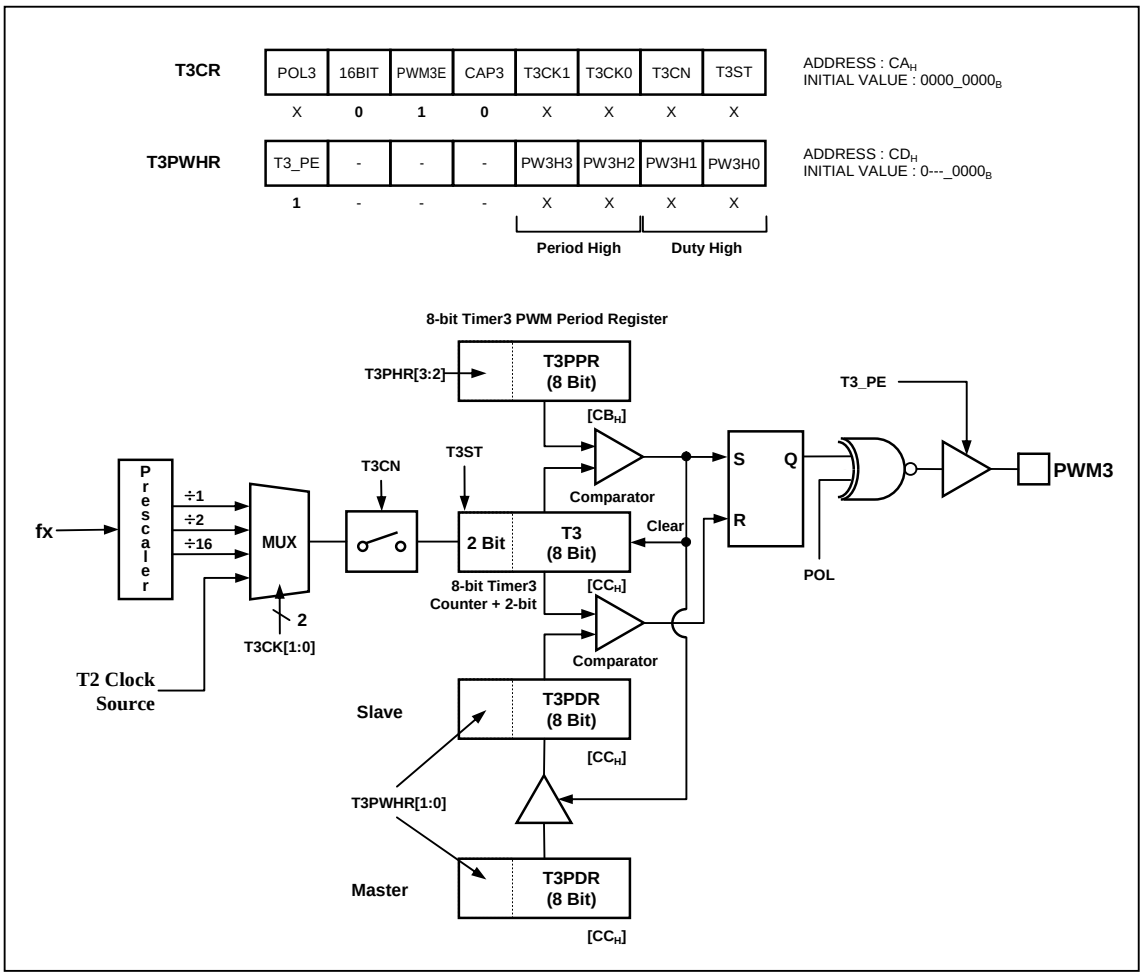



Figure 11-34 PWM Mode

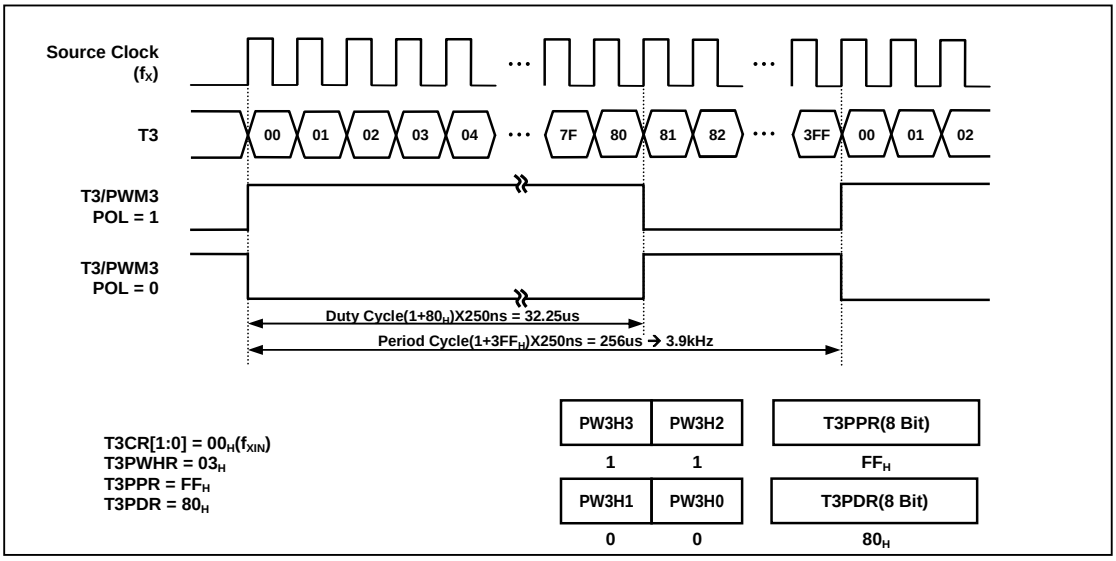



Figure 11-35 Example of PWM at 4MHz

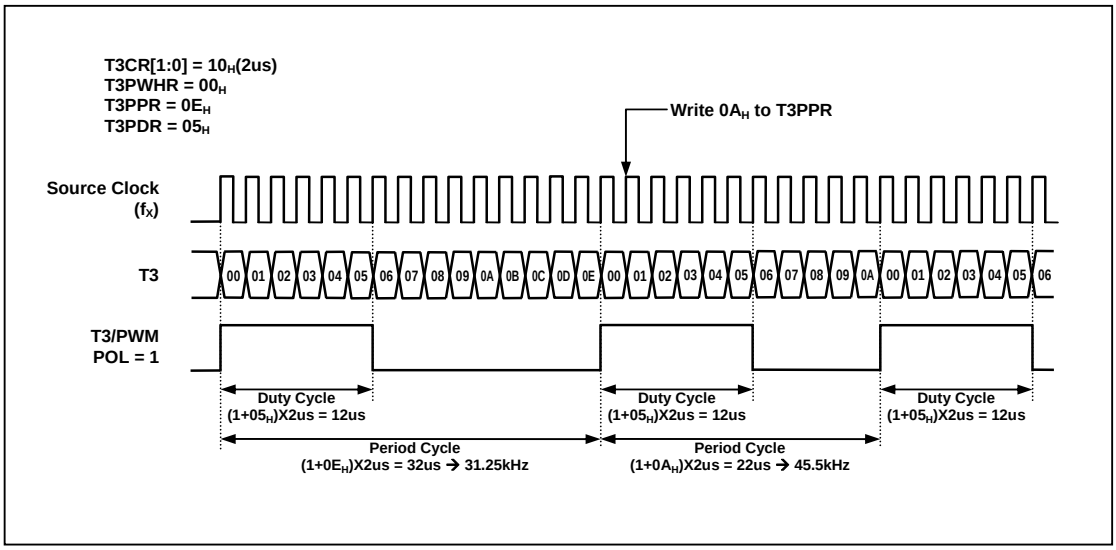



Figure 11-36 Example of Changing the Period in Absolute Duty Cycle at 4Mhz

### 11.5.2.7 8-Bit (16-Bit) Compare Output Mode

If the T3 (T2+T3) value and the T3DR (T2DR+T3DR) value are matched, T3/PWM3 port outputs. The output is 50:50 of duty square wave, the frequency is following

$$f_{COMP} = \frac{\text{Oscillator Frequency}}{2 \times \text{Prescaler Value} \times (TDR + 1)}$$

To export the compare output as T3/PWM3, the T3_PE bit in the T3PWHR register must set to '1'.

#### 11.5.2.8 Register Map

Table 11-10 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| T2CR | C6 | R/W | 00H | Timer 2 Mode Control Register |
| T2 | C7 | R | 00H | Timer 2 Register |
| T2DR | C7 | W | FFH | Timer 2 Data Register |
| CDR2 | C7 | R | 00H | Capture 2 Data Register |
| T3CR | CA | R/W | 00H | Timer 3 Mode Control Register |
| T3DR | CB | W | FFH | Timer 3 Data Register |
| T3PPR | CB | W | FFH | Timer 3 PWM Period Register |
| T3 | CC | R | 00H | Timer 3 Register |
| T3PDR | CC | R/W | 00H | Timer 3 PWM Duty Register |
| CDR3 | CC | R | 00H | Capture 3 Data Register |
| T3PWHR | CD | W | 00H | Timer 3 PWM High Register |

#### 11.5.2.9 Timer/Counter 2, 3 Register description

The Timer/Counter 2, 3 Register consists of Timer 2 Mode Control Register (T2CR), Timer 2 Register (T2), Timer 2 Data Register (T2DR), Capture 2 Data Register (CDR2), Timer 3 Mode Control Register (T3CR), Timer 3 Data Register (T3DR), Timer 3 PWM Period Register (T3PPR), Timer 3 Register (T3), Timer 3 PWM Duty Register (T3PPR), Capture 3 Data Register (CDR3) and Timer 3 PWM High Register (T3PWHR).

### 11.5.2.10 Register description for Timer/Counter 2, 3

**T2CR (Timer 2 Mode Control Register) : C6H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T2EN | T2_PE | CAP2 | T2CK2 | T2CK1 | T2CK0 | T2CN | T2ST |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**T2EN** Control Timer 2
- 0 Timer 2 disable
- 1 Timer 2 enable

**T2_PE** Control Timer 2 Output port
- 0 Timer 2 Output disable
- 1 Timer 2 Output enable

**CAP2** Control Timer 2 operation mode
- 0 Timer/Counter mode
- 1 Capture mode

**T2CK[2:0]** Select Timer 2 clock source. Fx is main system clock frequency

| T2CK2 | T2CK1 | T2CK0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | fx/2 |
| 0 | 0 | 1 | fx/4 |
| 0 | 1 | 0 | fx/16 |
| 0 | 1 | 1 | fx/64 |
| 1 | 0 | 0 | fx/256 |
| 1 | 0 | 1 | fx/1024 |
| 1 | 1 | 0 | fx/4096 |
| 1 | 1 | 1 | External Clock (EC2) |

**T2CN** Control Timer 2 Count pause/continue
- 0 Temporary count stop
- 1 Continue count

**T2ST** Control Timer 2 start/stop
- 0 Counter stop
- 1 Clear counter and start

**T2 (Timer 2 Register: Read Case) : C7H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T27 | T26 | T25 | T24 | T23 | T22 | T21 | T20 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**T2[7:0]** T2 Counter data

### T2DR (Timer 2 Data Register: Write Case) : C7H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T2D7 | T2D6 | T2D5 | T2D4 | T2D3 | T2D2 | T2D1 | T2D0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**T2D[7:0]** T2 Compare data

### CDR2 (Capture 2 Data Register: Read Case) : C7H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CDR27 | CDR26 | CDR25 | CDR24 | CDR23 | CDR22 | CDR21 | CDR20 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**CDR2[7:0]** T2 Capture data

### T3CR (Timer 3 Mode Count Register) : CAH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| POL | 16BIT | PWM3E | CAP3 | T3CK1 | T3CK0 | T3CN | T3ST |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**POL** Configure PWM polarity
- 0 Negative (Duty Match: Clear)
- 1 Positive (Duty Match: Set)

**16BIT** Select Timer 1 8/16Bit
- 0 8 Bit
- 1 16 Bit

**PWM3E** Control PWM enable
- 0 PWM disable
- 1 PWM enable

**CAP3** Control Timer 3 mode
- 0 Timer/Counter mode
- 1 Capture mode

**T3CK[1:0]** Select clock source of Timer 3. Fx is the frequency of main system.

| T3CK1 | T3CK0 | Description |
|---|---|---|
| 0 | 0 | fx |
| 0 | 1 | fx/2 |
| 1 | 0 | fx/16 |
| 1 | 1 | Use Timer 2 Clock |

Note) If you want to use “Using Timer 2 Clock”, you can set T2EN bit in T2CR

**T3CN** Control Timer 3 Count pause/continue
- 0 Temporary count stop
- 1 Continue count

**T3ST** Control Timer 3 start/stop
- 0 Counter stop
- 1 Clear counter and start

### T3DR (Timer 3 Data Register: Write Case) : CBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T3D7 | T3D6 | T3D5 | T3D4 | T3D3 | T3D2 | T3D1 | T3D0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**T3D[7:0]** T3 Compare data

### T3PPR (Timer 3 PWM Period Register: Write Case PWM mode only) : CBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T3PP7 | T3PP6 | T3PP5 | T3PP4 | T3PP3 | T3PP2 | T3PP1 | T3PP0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**T3PP[7:0]** T3 PWM Period data

### T3 (Timer 3 Register: Read Case) : CCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T37 | T36 | T35 | T34 | T33 | T32 | T31 | T30 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**T3[7:0]** T3 Counter Period data

### T3PDR (Timer 3 PWM Duty Register) : CCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T3PD7 | T3PD6 | T3PD5 | T3PD4 | T3PD3 | T3PD2 | T3PD1 | T3PD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**T3PD[7:0]** T3 PWM Duty data
Note) only write, when PWM3E '1'

### CDR3 (Capture 3 Data Register: Read Case) : CCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CDR37 | CDR36 | CDR35 | CDR34 | CDR33 | CDR32 | CDR31 | CDR30 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**CDR3[7:0]** T3 Capture data

### T3PWHR (Timer 3 PWM High Register) : CDH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T3_PE | - | - | - | PW3H3 | PW3H2 | PW3H1 | PW3H0 |
| W | - | - | - | W | W | W | W |

Initial value : 00H

**T3_PE** Control Timer 3 Output port operation
Note) only writable Bit. Be careful
0 Timer 3 Output disable
1 Timer 3 Output enable

**PW3H[3:2]** PWM period High value (Bit [9:8])

**PW3H[1:0]** PWM duty High value (Bit [9:8])

| | | | |
|---|---|---|---|
| PERIOD: | PW3H3 | PW3H2 | T3PPR[7:0] |
| DUTY: | PW3H1 | PW3H0 | T3PDR[7:0] |

### 11.5.3 16-Bit Timer 4

#### 11.5.3.1 Overview

The 16-bit timer 4 consists of Multiplexer, Timer Data Register High/Low, Timer Register High/Low, Timer Mode Control Register. It is able to use internal 16-bit timer/ counter without a port output function.

The 16-bit timer 4 is able to use the divided clock of the main clock selected from pre-scalar output.

#### 11.5.3.2 16 Bit Timer/Counter Mode

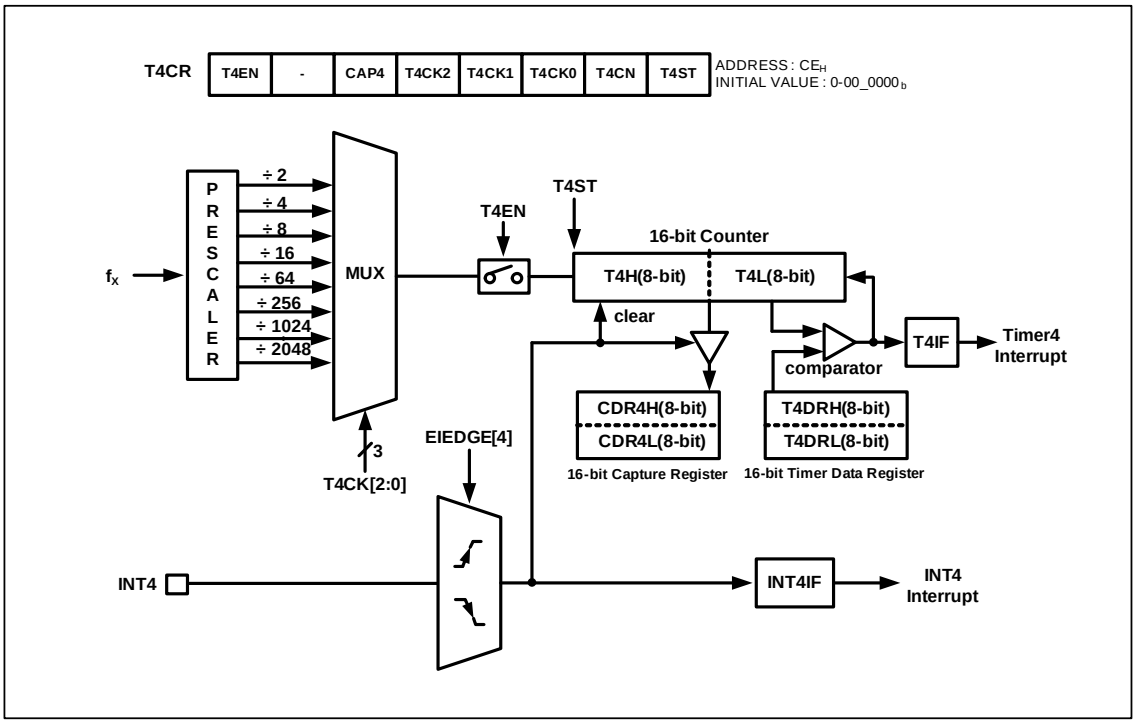



Figure 11-37 Timer4 16-bit Mode Block Diagram

### 11.5.3.3 Register Map

Table 11-11 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| T4CR | 0xCE | R/W | 00H | Timer 4 Mode Control Register |
| T4L | 0xCF | R | 00H | Timer 4 Low Register |
| T4LDR | 0xCF | W | FFH | Timer 4 Low Data Register |
| LCDR4 | 0xCF | R | 00H | Low Capture 4 Data Register |
| T4H | 0xD5 | R | 00H | Timer 4 High Register |
| T4HDR | 0xD5 | R/W | 00H | Timer 4 High Data Register |
| HCDR4 | 0xD5 | R | 00H | High Capture 4 Data Register |

### 11.5.3.4 Timer 4 Register description

The timer 4 register consists of Timer 4 Mode Control Register (T4CR), Timer 4 Low Register (T4L), Timer 4 Low Data Register (T4LDR), Low Capture 4 Data Register (LCDR4), Timer 4 High Register (T4H), Timer 4 High Data Register (T4HDR), High Capture 4 Data Register (HCDR4).

### 11.5.3.5 Register description for Timer 4

**T4CR (Timer 4 Mode Control Register) : CEH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T4EN | - | CAP4 | T4CK2 | T4CK1 | T4CK0 | T4CN | T4ST |
| R/W | - | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**T4EN** Control Timer 4 operation
- 0 Timer 4 disable
- 1 Timer 4 enable

**CAP4** Control Timer 4 mode
- 0 Timer/Counter mode
- 1 Capture mode

**T4CK[2:0]** Select Timer 4 clock source. fx is main system clock frequency

| T4CK2 | T4CK1 | T4CK0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | fx/2 |
| 0 | 0 | 1 | fx/4 |
| 0 | 1 | 0 | fx/8 |
| 0 | 1 | 1 | fx/16 |
| 1 | 0 | 0 | fx/64 |
| 1 | 0 | 1 | fx/256 |
| 1 | 1 | 0 | fx/1024 |
| 1 | 1 | 1 | fx/2048 |

**T4CN** Control Timer 4 Count pause/continue
- 0 Temporary count stop
- 1 Continue count

**T4ST** Control Timer 4 start/stop

| | |
|---|---|
| 0 | Counter stop |
| 1 | Clear Counter and start |

### T4L (Timer 4 Low Register: Read Case) : CFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T4L7 | T4L6 | T4L5 | T4L4 | T4L3 | T4L2 | T4L1 | T4L0 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**T4L[7:0]** T4L Counter

### T4LDR (Timer 4 Low Data Register: Write Case) : CFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T4LD7 | T4LD6 | T4LD5 | T4LD4 | T4LD3 | T4LD2 | T4LD1 | T4LD0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**T4LD[7:0]** T4L Compare

### LCDR4 (Low Capture 4 Data Register: Read Case) : CFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| LCDR47 | LCDR46 | LCDR45 | LCDR44 | LCDR43 | LCDR42 | LCDR41 | LCDR40 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**LCDR4[7:0]** T4L Capture data

### T4H (Timer 4 High Register: Read Case) : D5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T4H7 | T4H6 | T4H5 | T4H4 | T4H3 | T4H2 | T4H1 | T4H0 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**T4H[7:0]** T4H Counter Period

### T4HDR (Timer 4 High Data Register: Write Case) : D5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T4HD7 | T4HD6 | T4HD5 | T4HD4 | T4HD3 | T4HD2 | T4HD1 | T4HD0 |
| W | W | W | W | W | W | W | W |

Initial value : FFH

**T4HD[7:0]** T4H Compare

### HDR4 (High Capture 4 Data Register: Read Case) : D5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| HCDR47 | HCDR46 | HCDR45 | HCDR44 | HCDR43 | HCDR42 | HCDR41 | HCDR40 |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**HCDR4[7:0]** T4H Capture data

### 11.5.4 Timer Interrupt Status Register (TMISR)

#### 11.5.4.1 Register description for TMISR

**TMISR (Timer Interrupt Status Register) : D5H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | TMIF5 | TMIF4 | TMIF3 | TMIF2 | TMIF1 | TMIF0 |
| - | - | R | R | R | R | R | R |

Initial value : 00H

**TMIF5** Timer 5 Interrupt Flag
- 0 No Timer 5 interrupt
- 1 Timer 5 interrupt occurred, write “1” to clear interrupt flag

**TMIF4** Timer 4 Interrupt Flag
- 0 No Timer 4 interrupt
- 1 Timer 4 interrupt occurred, write “1” to clear interrupt flag

**TMIF3** Timer 3 Interrupt Flag
- 0 No Timer 3 interrupt
- 1 Timer 3 interrupt occurred, write “1” to clear interrupt flag

**TMIF2** Timer 2 Interrupt Flag
- 0 No Timer 2 interrupt
- 1 Timer 2 interrupt occurred, write “1” to clear interrupt flag

**TMIF1** Timer 1 Interrupt Flag
- 0 No Timer 1 interrupt
- 1 Timer 1 interrupt occurred, write “1” to clear interrupt flag

**TMIF0** Timer 0 Interrupt Flag
- 0 No Timer 0 interrupt
- 1 Timer 0 interrupt occurred, write “1” to clear interrupt flag

Note) The Timer Interrupt Status Register contains interrupt information of each timers. Even if user disabled timer interrupt at IE2, user could check timer interrupt condition from this register.

## 11.6 Buzzer Driver

### 11.6.1 Overview

The Buzzer consists of 8 Bit Counter and BUZDR (Buzzer Data Register), BUZCR (Buzzer Control Register). The Square Wave (61.035Hz~125 KHz, @8MHz) gets out of P12/BUZ pin. BUZDR (Buzzer Data Register) controls the Buzzer frequency (look at the following expression). In the BUZCR (Buzzer Control Register), BUCK[1:0] selects source clock divided from prescaler.

$$f_{BUZ}(Hz) = \frac{\text{Oscillator Frequency}}{2 \times \text{Prescaler Ratio} \times (\text{BUZDR} + 1)}$$

Table 11-12 Buzzer Frequency at 8MHz

| BUZDR[7:0] | Buzzer Frequency (kHz) | | | |
|---|---|---|---|---|
| | BUZCR[2:1]=00 | BUZCR[2:1]=01 | BUZCR[2:1]=10 | BUZCR[2:1]=11 |
| 0000_0000 | 125kHz | 62.5kHz | 31.25kHz | 15.625kHz |
| 0000_0001 | 62.5kHz | 31.25kHz | 15.625kHz | 7.812kHz |
| … | … | … | … | … |
| 1111_1101 | 492.126Hz | 246.063Hz | 123.031Hz | 61.515Hz |
| 1111_1110 | 490.196Hz | 245.098Hz | 122.549Hz | 61.274Hz |
| 1111_1111 | 488.281Hz | 244.141Hz | 122.07Hz | 61.035Hz |

### 11.6.2 Block Diagram

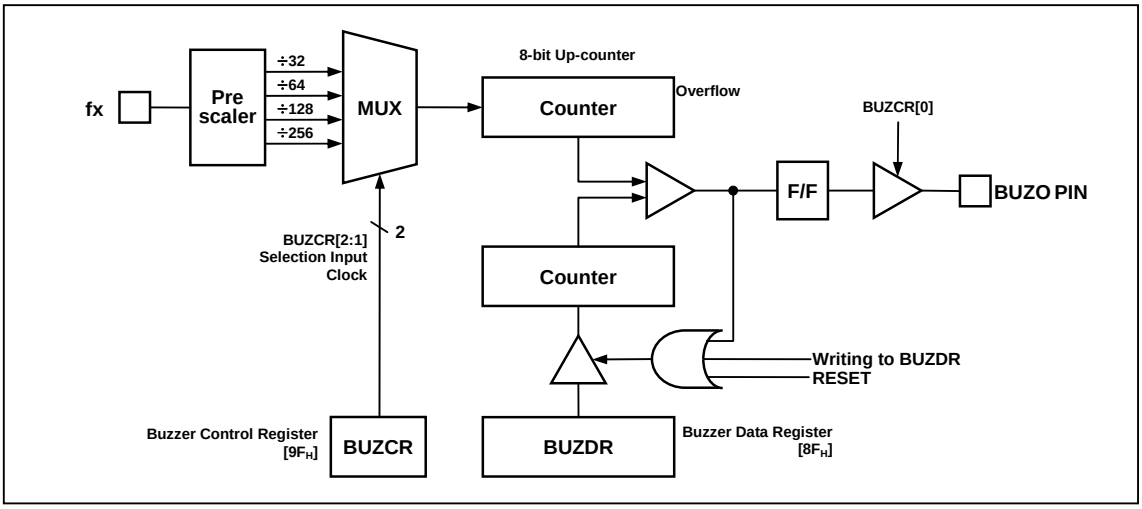



Figure 11-38 Buzzer Driver Block Diagram

### 11.6.3 Register Map

Table 11-13 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| BUZDR | 8FH | R/W | FFH | Buzzer Data Register |
| BUZCR | 9FH | R/W | 00H | Buzzer Control Register |

### 11.6.4 Buzzer Driver Register description

Buzzer Driver consists of Buzzer Data Register (BUZDR), Buzzer Control Register (BUZCR).

### 11.6.5 Register description for Buzzer Driver

**BUZDR (Buzzer Data Register) : 8FH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BUZDR7 | BUZDR6 | BUZDR5 | BUZDR4 | BUZDR3 | BUZDR2 | BUZDR1 | BUZDR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : FFH

**BUZDR[7:0]** This bits control the Buzzer frequency
Its resolution is 00H ~ FFH

**BUZCR (Buzzer Control Register) : 9FH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | - | - | - | - | BUCK1 | BUCK0 | BUZEN |
| - | - | - | - | - | R/W | R/W | R/W |

Initial value : 00H

**BUCK[1:0]** Buzzer Driver Source Clock Selection

| BUCK1 | BUCK0 | Source Clock |
|---|---|---|
| 0 | 0 | fx/32 |
| 0 | 1 | fx/64 |
| 1 | 0 | fx/128 |
| 1 | 1 | fx/256 |

**BUZEN** Buzzer Driver Operation Control

| | |
|---|---|
| 0 | Buzzer Driver disable |
| 1 | Buzzer Driver enable |

Note) fx: Main system clock oscillation frequency

## 11.7 USART

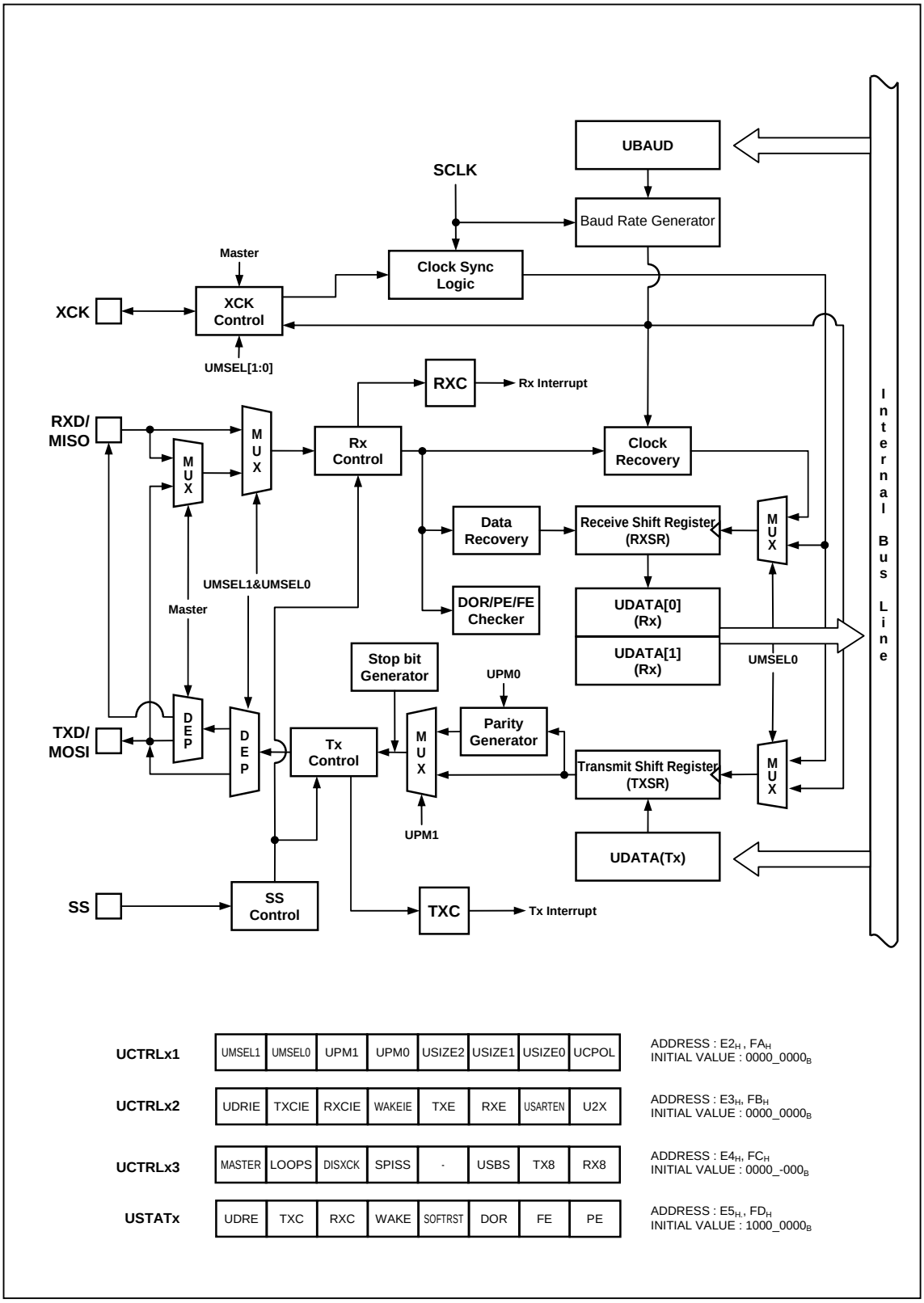
### 11.7.1 Overview

The Universal Synchronous and Asynchronous serial Receiver and Transmitter (USART) is a highly flexible serial communication device. The main features are listed below.

- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous Operation
- Master or Slave Clocked Synchronous and SPI Operation
- Supports all four SPI Modes of Operation (Mode 0, 1, 2, 3)
- LSB First or MSB First Data Transfer @SPI mode
- High Resolution Baud Rate Generator
- Supports Serial Frames with 5,6,7,8, or 9 Data Bits and 1 or 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data OverRun Detection
- Framing Error Detection
- Digital Low Pass Filter
- Three Separate Interrupts on TX Complete, TX Data Register Empty and RX Complete
- Double Speed Asynchronous Communication Mode

USART has three main parts of Clock Generator, Transmitter and Receiver. The Clock Generation logic consists of synchronization logic for external clock input used by synchronous or SPI slave operation, and the baud rate generator for asynchronous or master (synchronous or SPI) operation. The Transmitter consists of a single write buffer, a serial shift register, parity generator and control logic for handling different serial frame formats. The write buffer allows a continuous transfer of data without any delay between frames. The receiver is the most complex part of the USART module due to its clock and data recovery units. The recovery unit is used for asynchronous data reception. In addition to the recovery unit, the Receiver includes a parity checker, a shift register, a two level receive FIFO (UDATAx) and control logic. The Receiver supports the same frame formats as the Transmitter and can detect Frame Error, Data OverRun and Parity Errors.

### 11.7.2 Block Diagram




| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **UCTRLx1** | UMSEL1 | UMSEL0 | UPM1 | UPM0 | USIZE2 | USIZE1 | USIZE0 | UCPOL |
| **UCTRLx2** | UDRIE | TXCIE | RXCIE | WAKEIE | TXE | RXE | USARTEN | U2X |
| **UCTRLx3** | MASTER | LOOPS | DISXCK | SPISS | - | USBS | TX8 | RX8 |
| **USTATx** | UDRE | TXC | RXC | WAKE | SOFTRST | DOR | FE | PE |

ADDRESS : $E2_H$ , $FA_H$
INITIAL VALUE : $0000_0000_B$

ADDRESS : $E3_H$, $FB_H$
INITIAL VALUE : $0000_0000_B$

ADDRESS : $E4_H$, $FC_H$
INITIAL VALUE : $0000_-000_B$

ADDRESS : $E5_H$, $FD_H$
INITIAL VALUE : $1000_0000_B$

Figure 11-39 USART Block Diagram

### 11.7.3 Clock Generation

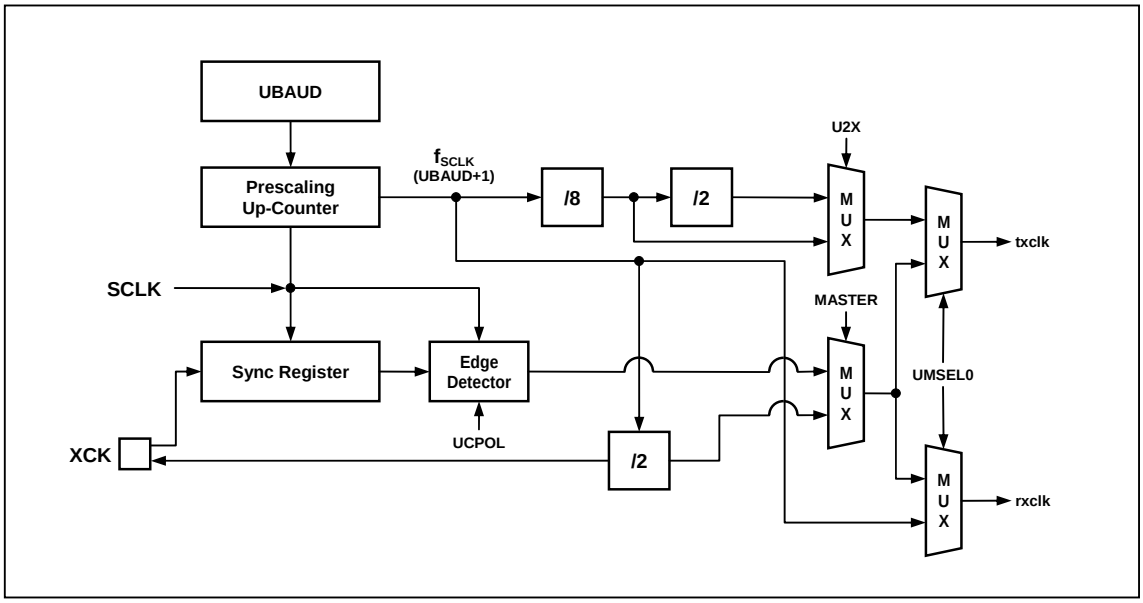



Figure 11-40 Clock Generation Block Diagram

The Clock generation logic generates the base clock for the Transmitter and Receiver. The USART supports four modes of clock operation and those are Normal Asynchronous, Double Speed Asynchronous, Master Synchronous and Slave Synchronous. The clock generation scheme for Master SPI and Slave SPI mode is the same as Master Synchronous and Slave Synchronous operation mode. The UMSELn bit in UCTRLx1 register selects between asynchronous and synchronous operation. Asynchronous Double Speed mode is controlled by the U2X bit in the UCTRLx2 register. The MASTER bit in UCTRLx2 register controls whether the clock source is internal (Master mode, output port) or external (Slave mode, input port). The XCK pin is only active when the USART operates in Synchronous or SPI mode.

Table below contains equations for calculating the baud rate (in bps).

Table 11-14 Equations for Calculating Baud Rate Register Setting

| Operating Mode | Equation for Calculating Baud Rate |
|---|---|
| Asynchronous Normal Mode (U2X=0) | $\text{Baud Rate} = \frac{\text{fSCLK}}{16(\text{UBAUDx} + 1)}$ |
| Asynchronous Double Speed Mode (U2X=1) | $\text{Baud Rate} = \frac{\text{fSCLK}}{8(\text{UBAUDx} + 1)}$ |
| Synchronous or SPI Master Mode | $\text{Baud Rate} = \frac{\text{fSCLK}}{2(\text{UBAUDx} + 1)}$ |

### 11.7.4 External Clock (XCK)

External clocking is used by the synchronous or spi slave modes of operation.

External clock input from the XCK pin is sampled by a synchronization logic to remove meta-stability. The output from the synchronization logic must then pass through an edge detector before it can be used by the Transmitter and Receiver. This process introduces a two CPU clock period delay and therefore the maximum frequency of the external XCK pin is limited by the following equation.

$$fXCK = \frac{fSCLK}{4}$$

where fXCK is the frequency of XCK and fSCLK is the frequency of main system clock (SCLK).

### 11.7.5 Synchronous mode Operation

When synchronous or spi mode is used, the XCK pin will be used as either clock input (slave) or clock output (master). The dependency between the clock edges and data sampling or data change is the same. The basic principle is that data input on RXD (MISO in spi mode) pin is sampled at the opposite XCK clock edge of the edge in the data output on TXD (MOSI in spi mode) pin is changed.

The UCPOL bit in UCTRLx1 register selects which XCK clock edge is used for data sampling and which is used for data change. As shown in the figure below, when UCPOL is zero the data will be changed at rising XCK edge and sampled at falling XCK edge.

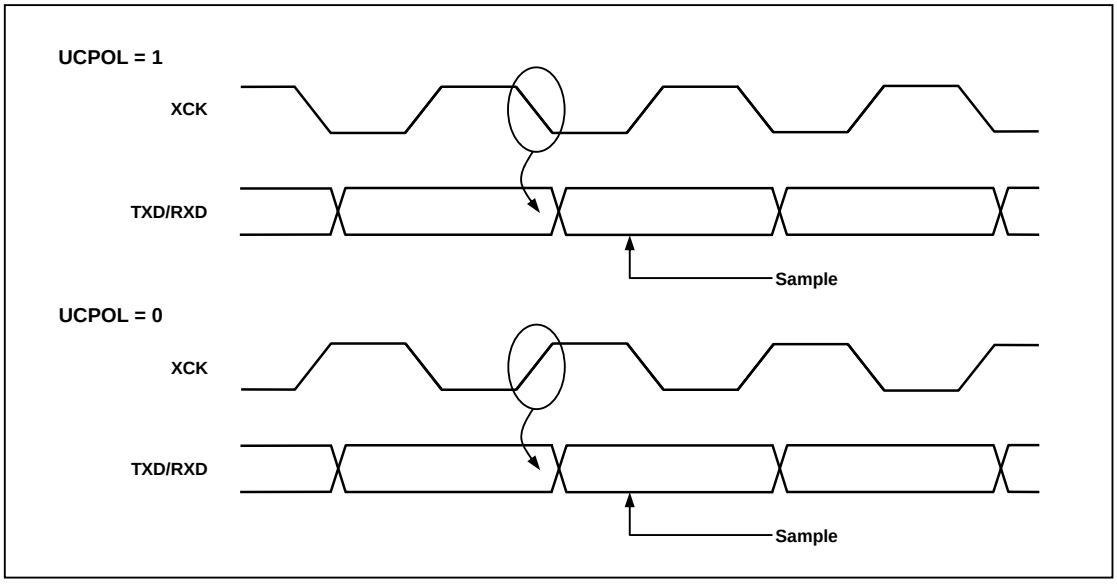



Figure 11-41 Synchronous Mode XCKn Timing

### 11.7.6 Data format

A serial frame is defined to be one character of data bits with synchronization bits (start and stop bits), and optionally a parity bit for error checking.

The USART supports all 30 combinations of the following as valid frame formats.

- 1 start bit
- 5, 6, 7, 8 or 9 data bits
- no, even or odd parity bit
- 1 or 2 stop bits

A frame starts with the start bit followed by the least significant data bit (LSB). Then the next data bits, up to a total of nine, are succeeding, ending with the most significant bit (MSB). If enabled the parity bit is inserted after the data bits, before the stop bits. A high to low transition on data pin is considered as start bit. When a complete frame is transmitted, it can be directly followed by a new frame, or the communication line can be set to an idle state. The idle means high state of data pin. The next figure shows the possible combinations of the frame formats. Bits inside brackets are optional.

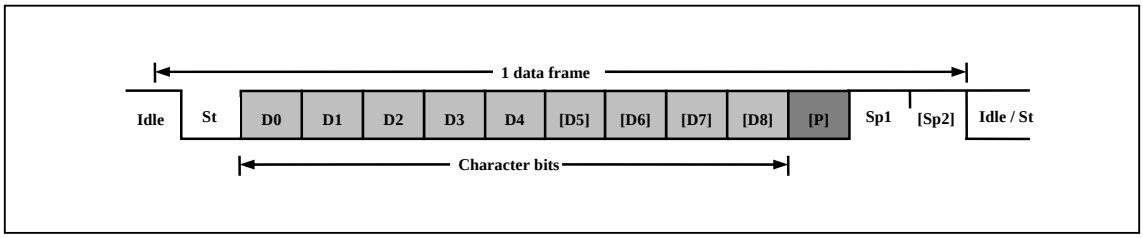



Figure 11-42 frame format

1 data frame consists of the following bits

- Idle No communication on communication line (TxD/RxD)
- St Start bit (Low)
- Dn Data bits (0~8)
- Parity bit ---------- Even parity, Odd parity, No parity
- Stop bit(s) ---------- 1 bit or 2 bits

The frame format used by the USART is set by the USIZE[2:0], UPM[1:0] and USBS bits in UCTRLx1 register. The Transmitter and Receiver use the same setting.

### 11.7.7 Parity bit

The parity bit is calculated by doing an exclusive-or of all the data bits. If odd parity is used, the result of the exclusive-or is inverted. The parity bit is located between the MSB and first stop bit of a serial frame.

$P_{even} = D_{n-1} \wedge \ldots \wedge D_3 \wedge D_2 \wedge D_1 \wedge D_0 \wedge 0$
$P_{odd} = D_{n-1} \wedge \ldots \wedge D_3 \wedge D_2 \wedge D_1 \wedge D_0 \wedge 1$
$P_{even}$ : Parity bit using even parity
$P_{odd}$ : Parity bit using odd parity
$D_n$ : Data bit n of the character

### 11.7.8 USART Transmitter

The USART Transmitter is enabled by setting the TXE bit in UCTRLx1 register. When the Transmitter is enabled, the normal port operation of the TXD pin is overridden by the serial output pin of USART. The baud-rate, operation mode and frame format must be setup once before doing any transmissions. If synchronous or spi operation is used, the clock on the XCK pin will be overridden and used as transmission clock. If USART operates in spi mode, SS pin is used as SS input pin in slave mode or can be configured as SS output pin in master mode. This can be done by setting SPISS bit in UCTRLx3 register.

#### 11.7.8.1 Sending Tx data

A data transmission is initiated by loading the transmit buffer (UDATAx register I/O location) with the data to be transmitted. The data written in transmit buffer is moved to the shift register when the shift register is ready to send a new frame. The shift register is loaded with the new data if it is in idle state or immediately after the last stop bit of the previous frame is transmitted. When the shift register is loaded with new data, it will transfer one complete frame at the settings of control registers. If the 9-bit characters are used in asynchronous or synchronous operation mode (USIZE[2:0]=7), the ninth bit must be written to the TX8 bit in UCTRLx3 register before loading transmit buffer (UDATA register).

#### 11.7.8.2 Transmitter flag and interrupt

The USART Transmitter has 2 flags which indicate its state. One is USART Data Register Empty (UDRE) and the other is Transmit Complete (TXC). Both flags can be interrupt sources.

UDRE flag indicates whether the transmit buffer is ready to be loaded with new data. This bit is set when the transmit buffer is empty and cleared when the transmit buffer contains data to be transmitted that has not yet been moved into the shift register. And also this flag can be cleared by writing ‘0’ to this bit position. Writing ‘1’ to this bit position is prevented.

When the Data Register Empty Interrupt Enable (UDRIE) bit in UCTRL2 register is set and the Global Interrupt is enabled, USART Data Register Empty Interrupt is generated while UDRE flag is set.

The Transmit Complete (TXC) flag bit is set when the entire frame in the transmit shift register has been shifted out and there are no more data in the transmit buffer. The TXC flag is automatically cleared when the Transmit Complete Interrupt service routine is executed, or it can be cleared by writing ‘0’ to TXC bit in USTAT register.

When the Transmit Complete Interrupt Enable (TXCIE) bit in UCTRL2 register is set and the Global Interrupt is enabled, USART Transmit Complete Interrupt is generated while TXC flag is set.

#### 11.7.8.3 Parity Generator

The Parity Generator calculates the parity bit for the sending serial frame data. When parity bit is enabled (UPM[1]=1), the transmitter control logic inserts the parity bit between the MSB and the first stop bit of the sending frame.

#### 11.7.8.4 Disabling Transmitter

Disabling the Transmitter by clearing the TXE bit will not become effective until ongoing transmission is completed. When the Transmitter is disabled, the TXD pin is used as normal General Purpose I/O (GPIO) or primary function pin.

### 11.7.9 USART Receiver

The USART Receiver is enabled by setting the RXE bit in the UCTRLx1 register. When the Receiver is enabled, the normal pin operation of the RXD pin is overridden by the USART as the serial input pin of the Receiver. The baud-rate, mode of operation and frame format must be set before serial reception. If synchronous or spi operation is used, the clock on the XCK pin will be used as transfer clock. If USART operates in spi mode, SS pin is used as SS input pin in slave mode or can be configured as SS output pin in master mode. This can be done by setting SPISS bit in UCTRLx3 register.

#### 11.7.9.1 Receiving Rx data

When USART is in synchronous or asynchronous operation mode, the Receiver starts data reception when it detects a valid start bit (LOW) on RXD pin. Each bit after start bit is sampled at pre-defined baud-rate (asynchronous) or sampling edge of XCK (synchronous), and shifted into the receive shift register until the first stop bit of a frame is received. Even if there's $2^{nd}$ stop bit in the frame, the $2^{nd}$ stop bit is ignored by the Receiver. That is, receiving the first stop bit means that a complete serial frame is present in the receiver shift register and contents of the shift register are to be moved into the receive buffer. The receive buffer is read by reading the UDATAx register.

If 9-bit characters are used (USIZE[2:0] = 7) the ninth bit is stored in the RX8 bit position in the UCTRLx3 register. The $9^{th}$ bit must be read from the RX8 bit before reading the low 8 bits from the UDATAx register. Likewise, the error flags FE, DOR, PE must be read before reading the data from UDATAx register. This is because the error flags are stored in the same FIFO position of the receive buffer.

#### 11.7.9.2 Receiver flag and interrupt

The USART Receiver has one flag that indicates the Receiver state.

The Receive Complete (RXC) flag indicates whether there are unread data present in the receive buffer. This flag is set when there are unread data in the receive buffer and cleared when the receive

buffer is empty. If the Receiver is disabled (RXE=0), the receiver buffer is flushed and the RXC flag is cleared.

When the Receive Complete Interrupt Enable (RXCIE) bit in the UCTRLx2 register is set and Global Interrupt is enabled, the USART Receiver Complete Interrupt is generated while RXC flag is set.

The USART Receiver has three error flags which are Frame Error (FE), Data OverRun (DOR) and Parity Error (PE). These error flags can be read from the USTATx register. As data received are stored in the 2-level receive buffer, these error flags are also stored in the same position of receive buffer. So, before reading received data from UDATAx register, read the USTATx register first which contains error flags.

The Frame Error (FE) flag indicates the state of the first stop bit. The FE flag is zero when the stop bit was correctly detected as one, and the FE flag is one when the stop bit was incorrect, ie detected as zero. This flag can be used for detecting out-of-sync conditions between data frames.

The Data OverRun (DOR) flag indicates data loss due to a receive buffer full condition. A DOR occurs when the receive buffer is full, and another new data is present in the receive shift register which are to be stored into the receive buffer. After the DOR flag is set, all the incoming data are lost. To prevent data loss or clear this flag, read the receive buffer.

The Parity Error (PE) flag indicates that the frame in the receive buffer had a Parity Error when received. If Parity Check function is not enabled (UPM[1]=0), the PE bit is always read zero.

Note) The error flags related to receive operation are not used when USART is in SPI mode.

#### 11.7.9.3 Parity Checker

If Parity Bit is enabled (UPM[1]=1), the Parity Checker calculates the parity of the data bits in incoming frame and compares the result with the parity bit from the received serial frame.

#### 11.7.9.4 Disabling Receiver

In contrast to Transmitter, disabling the Receiver by clearing RXE bit makes the Receiver inactive immediately. When the Receiver is disabled the Receiver flushes the receive buffer and the remaining data in the buffer is all reset. The RXD pin is not overridden the function of USART, so RXD pin becomes normal GPIO or primary function pin.

#### 11.7.9.5 Asynchronous Data Reception

To receive asynchronous data frame, the USART includes a clock and data recovery unit. The Clock Recovery logic is used for synchronizing the internally generated baud-rate clock to the incoming asynchronous serial frame on the RXD pin.

The Data recovery logic samples and low pass filters the incoming bits, and this removes the noise of RXD pin.

The next figure illustrates the sampling process of the start bit of an incoming frame. The sampling rate is 16 times the baud-rate for normal mode, and 8 times the baud rate for Double Speed mode (U2X=1). The horizontal arrows show the synchronization variation due to the asynchronous sampling process. Note that larger time variation is shown when using the Double Speed mode.

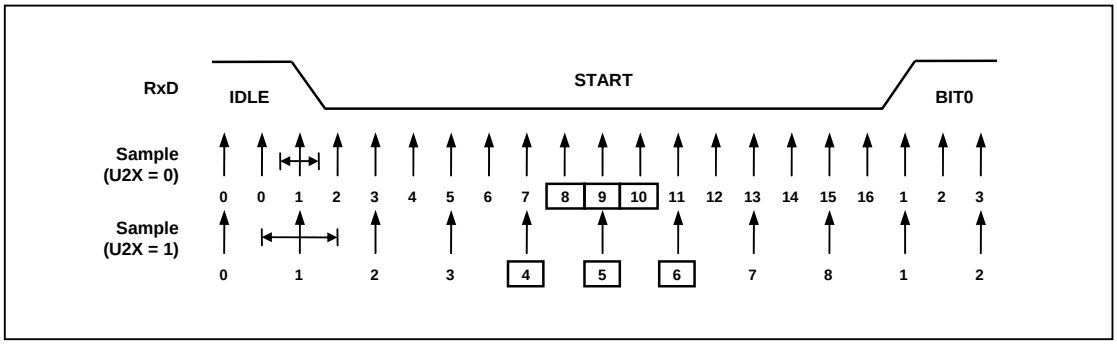



Figure 11-43 Start Bit Sampling

When the Receiver is enabled (RXE=1), the clock recovery logic tries to find a high to low transition on the RXD line, the start bit condition. After detecting high to low transition on RXD line, the clock recovery logic uses samples 8,9, and 10 for Normal mode, and samples 4, 5, and 6 for Double Speed mode to decide if a valid start bit is received. If more than 2 samples have logical low level, it is considered that a valid start bit is detected and the internally generated clock is synchronized to the incoming data frame. And the data recovery can begin. The synchronization process is repeated for each start bit.

As described above, when the Receiver clock is synchronized to the start bit, the data recovery can begin. Data recovery process is almost similar to the clock recovery process. The data recovery logic samples 16 times for each incoming bits for Normal mode and 8 times for Double Speed mode. And uses sample 8, 9, and 10 to decide data value for Normal mode, samples 4, 5, and 6 for Double Speed mode. If more than 2 samples have low levels, the received bit is considered to a logic 0 and more than 2 samples have high levels, the received bit is considered to a logic 1. The data recovery process is then repeated until a complete frame is received including the first stop bit. The decided bit value is stored in the receive shift register in order. Note that the Receiver only uses the first stop bit of a frame. Internally, after receiving the first stop bit, the Receiver is in idle state and waiting to find start bit.

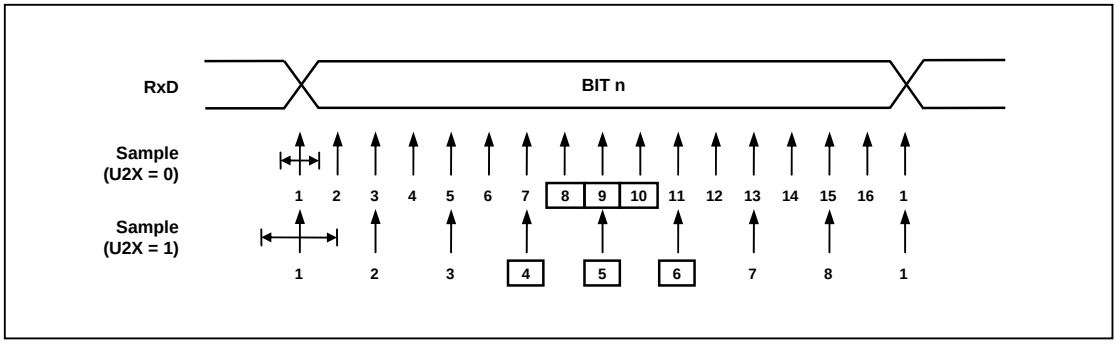
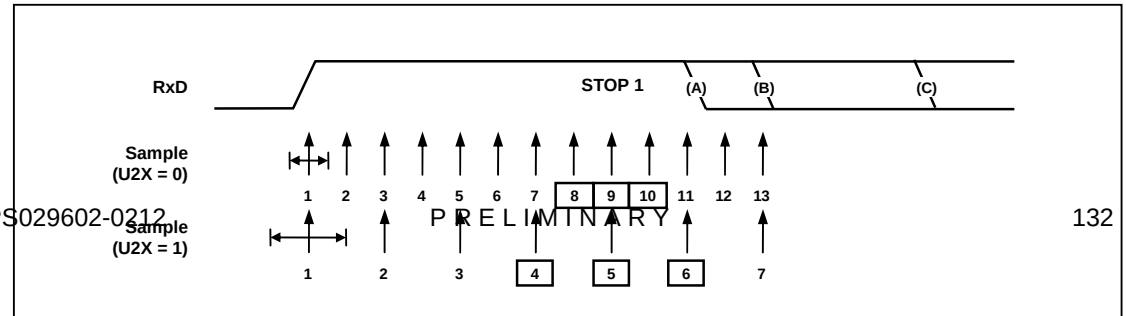



Figure 11-44 Sampling of Data and Parity Bit

The process for detecting stop bit is like clock and data recovery process. That is, if 2 or more samples of 3 center values have high level, correct stop bit is detected, else a Frame Error flag is set. After deciding first stop bit whether a valid stop bit is received or not, the Receiver goes idle state and monitors the RXD line to check a valid high to low transition is detected (start bit detection).



### 11.7.10 SPI Mode

The USART can be set to operate in industrial standard SPI compliant mode. The SPI mode has the following features.

- Full duplex, three-wire synchronous data transfer
- Master or Slave operation
- Supports all four SPI modes of operation (mode0, 1, 2, and 3)
- Selectable LSB first or MSB first data transfer
- Double buffered transmit and receive
- Programmable transmit bit rate

When SPI mode is enabled (UMSEL[1:0]=3), the Slave Select (SS) pin becomes active low input in slave mode operation, or can be output in master mode operation if SPISS bit is set.

Note that during SPI mode of operation, the pin RXD is renamed as MISO and TXD is renamed as MOSI for compatibility to other SPI devices.

#### 11.7.10.1 SPI Clock formats and timing

To accommodate a wide variety of synchronous serial peripherals from different manufacturers, the USART has a clock polarity bit (UCPOL) and a clock phase control bit (UCPHA) to select one of four clock formats for data transfers. UCPOL selectively insert an inverter in series with the clock. UCPHA chooses between two different clock phase relationships between the clock and data. Note that UCPHA and UCPOL bits in UCTRLx1 register have different meanings according to the UMSEL[1:0] bits which decides the operating mode of USART.

Table below shows four combinations of UCPOL and UCPHA for SPI mode 0, 1, 2, and 3.

Table 11-15 CPOL Funtionality

| SPI Mode | UCPOL | UCPHA | Leading Edge | Trailing Edge |
|---|---|---|---|---|
| 0 | 0 | 0 | Sample (Rising) | Setup (Falling) |
| 1 | 0 | 1 | Setup (Rising) | Sample (Falling) |
| 2 | 1 | 0 | Sample (Falling) | Setup (Rising) |
| 3 | 1 | 1 | Setup (Falling) | Sample (Rising) |

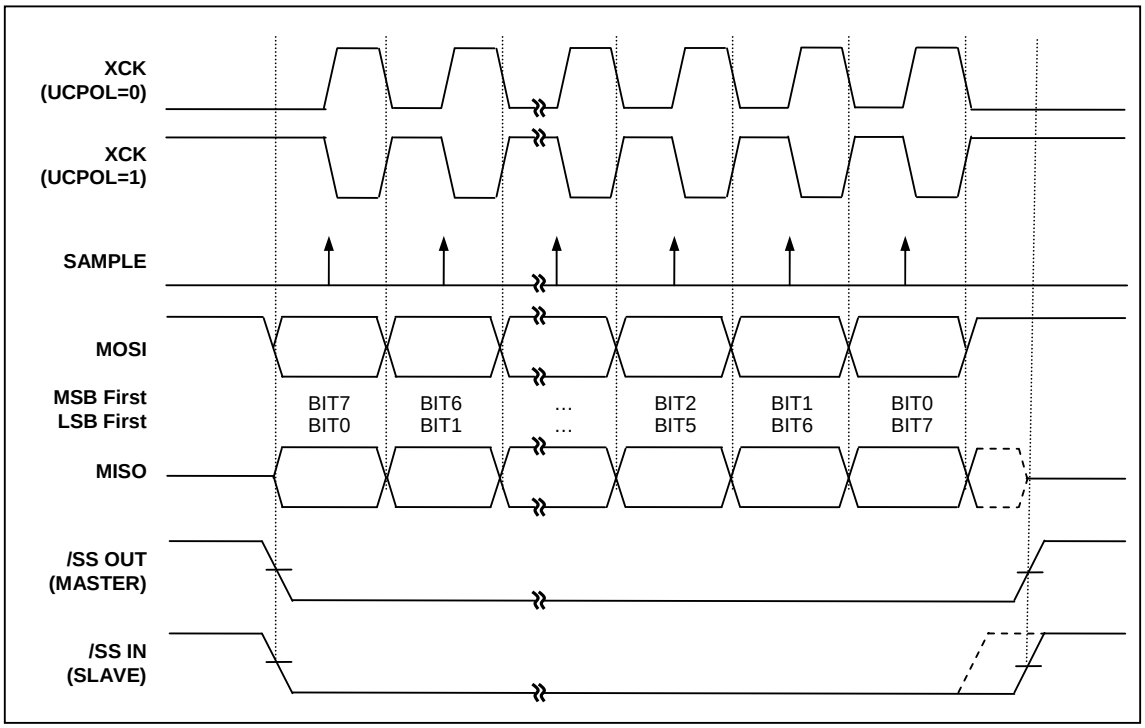



Figure 11-46 SPI Clock Formats when UCPHA=0

When UCPHA=0, the slave begins to drive its MISO output with the first data bit value when SS goes to active low. The first XCK edge causes both the master and the slave to sample the data bit value on their MISO and MOSI inputs, respectively. At the second XCK edge, the USART shifts the second data bit value out to the MOSI and MISO outputs of the master and slave, respectively. Unlike the case of UCPHA=1, when UCPHA=0, the slave's SS input must go to its inactive high level between transfers. This is because the slave can prepare the first data bit when it detects falling edge of SS input.

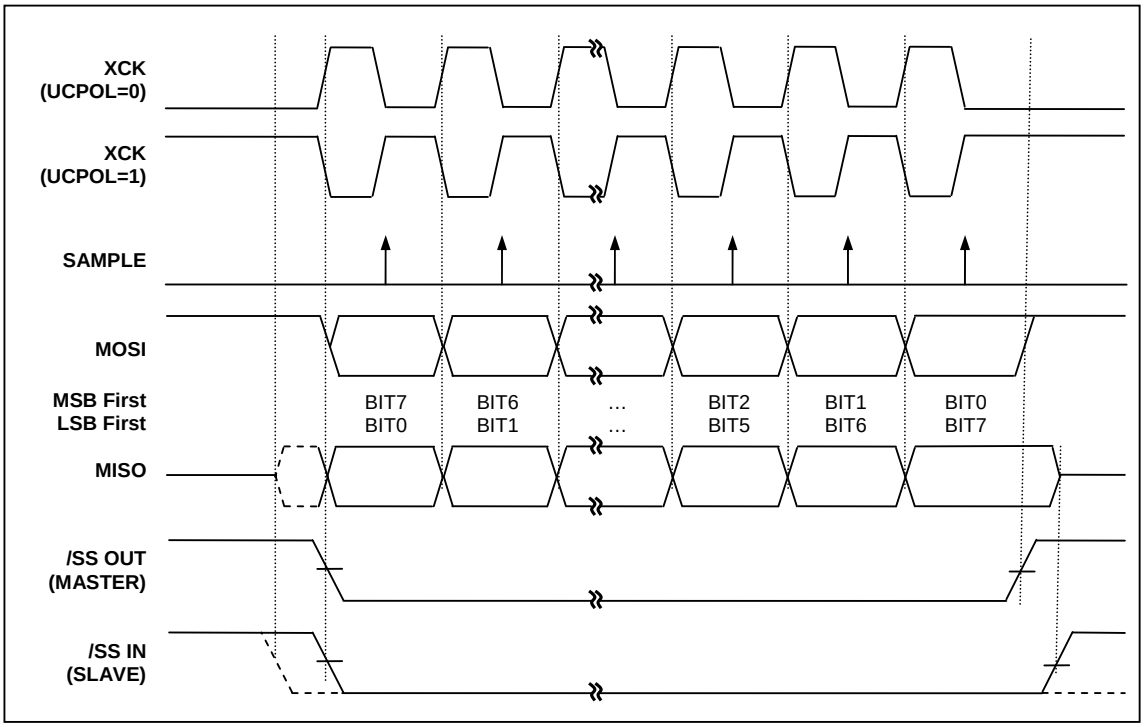



Figure 11-47 SPI Clock Formats when UCPHA=1

When UCPHA=1, the slave begins to drive its MISO output when SS goes active low, but the data is not defined until the first XCK edge. The first XCK edge shifts the first bit of data from the shifter onto the MOSI output of the master and the MISO output of the slave. The next XCK edge causes both the master and slave to sample the data bit value on their MISO and MOSI inputs, respectively. At the third XCK edge, the USART shifts the second data bit value out to the MOSI and MISO output of the master and slave respectively. When UCPHA=1, the slave's SS input is not required to go to its inactive high level between transfers.

Because the SPI logic reuses the USART resources, SPI mode of operation is similar to that of synchronous or asynchronous operation. An SPI transfer is initiated by checking for the USART Data Register Empty flag (UDRE=1) and then writing a byte of data to the UDATA Register. In master mode of operation, even if transmission is not enabled (TXE=0), writing data to the UDATA register is necessary because the clock XCK is generated from transmitter block.

### 11.7.11 Register Map

Table 11-16 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| UCTRL01 | E2H | R/W | 00H | USART Control 1 Register 0 |
| UCTRL02 | E3H | R/W | 00H | USART Control 2 Register 0 |
| UCTRL03 | E4H | R/W | 00H | USART Control 3 Register 0 |
| USTAT0 | E5H | R | 80H | USART Status Register 0 |
| UBAUD0 | E6H | R/W | FFH | USART Baud Rate Generation Register 0 |
| UDATA0 | E7H | R/W | FFH | USART Data Register 0 |
| UCTRL11 | FAH | R/W | 00H | USART Control 1 Register 1 |
| UCTRL12 | FBH | R/W | 00H | USART Control 2 Register 1 |
| UCTRL13 | FCH | R/W | 00H | USART Control 3 Register 1 |
| USTAT1 | FDH | R | 80H | USART Status Register 1 |
| UBAUD1 | FEH | R/W | FFH | USART Baud Rate Generation Register 1 |
| UDATA1 | FFH | R/W | FFH | USART Data Register 2 |

### 11.7.12 USART Register description

USART module consists of USART Control 1 Register (UCTRLx1), USART Control 2 Register (UCTRLx2), USART Control 3 Register (UCTRLx3), USART Status Register (USTATx), USART Data Register (UDATAx), and USART Baud Rate Generation Register (UBAUDx).

### 11.7.13 Register description for USART

**UCTRLx1 (USART Control 1 Register) E2H, FAH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UMSEL1 | UMSEL0 | UPM1 | UPM0 | USIZE2 | USIZE1<br>UDORD | USIZE0<br>UCPHA | UCPOL |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : $00_H$

**UMSEL[1:0]** Selects operation mode of USART

| UMSEL1 | UMSEL0 | Operating Mode |
|---|---|---|
| 0 | 0 | Asynchronous Mode (Normal Uart) |
| 0 | 1 | Synchronous Mode (Synchronous Uart) |
| 1 | 0 | Reserved |
| 1 | 1 | SPI Mode |

**UPM[1:0]** Selects Parity Generation and Check methods

| UPM1 | UPM0 | Parity mode |
|---|---|---|
| 0 | 0 | No Parity |
| 0 | 1 | Reserved |
| 1 | 0 | Even Parity |
| 1 | 1 | Odd Parity |

**USIZE[2:0]** When in asynchronous or synchronous mode of operation, selects the length of data bits in frame.

| USIZE2 | USIZE1 | USIZE0 | Data length |
|---|---|---|---|
| 0 | 0 | 0 | 5 bit |
| 0 | 0 | 1 | 6 bit |
| 0 | 1 | 0 | 7 bit |
| 0 | 1 | 1 | 8 bit |
| 1 | 0 | 0 | Reserved |
| 1 | 0 | 1 | Reserved |
| 1 | 1 | 0 | Reserved |
| 1 | 1 | 1 | 9 bit |

**UDORD** This bit is in the same bit position with USIZE1. In SPI mode, when set to one the MSB of the data byte is transmitted first. When set to zero the LSB of the data byte is transmitted first.

| | |
|---|---|
| 0 | LSB First |
| 1 | MSB First |

**UCPOL** Selects polarity of XCK in synchronous or spi mode

| | |
|---|---|
| 0 | TXD change @Rising Edge, RXD change @Falling Edge |
| 1 | TXD change @ Falling Edge, RXD change @ Rising Edge |

**UCPHA** This bit is in the same bit position with USIZE0. In SPI mode, along with UCPOL bit, selects one of two clock formats for different kinds of synchronous serial peripherals. Leading edge means first XCK edge and trailing edge means 2nd or last clock edge of XCK in one XCK pulse. And Sample means detecting of incoming receive bit, Setup means preparing transmit data.

| UCPOL | UCPHA | Leading Edge | Trailing Edge |
|---|---|---|---|
| 0 | 0 | Sample (Rising) | Setup (Falling) |
| 0 | 1 | Setup (Rising) | Sample (Falling) |
| 1 | 0 | Sample (Falling) | Setup (Rising) |
| 1 | 1 | Setup (Falling) | Sample (Rising) |

### UCTRL2 (USART Control 2 Register) E3H, FBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UDRIE | TXCIE | RXCIE | WAKEIE | TXE | RXE | USARTEN | U2X |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : $00_H$

**UDRIE** Interrupt enable bit for USART Data Register Empty.
- 0 Interrupt from UDRE is inhibited (use polling)
- 1 When UDRE is set, request an interrupt

**TXCIE** Interrupt enable bit for Transmit Complete.
- 0 Interrupt from TXC is inhibited (use polling)
- 1 When TXC is set, request an interrupt

**RXCIE** Interrupt enable bit for Receive Complete
- 0 Interrupt from RXC is inhibited (use polling)
- 1 When RXC is set, request an interrupt

**WAKEIE** Interrupt enable bit for Asynchronous Wake in STOP mode. When device is in stop mode, if RXD goes to LOW level an interrupt can be requested to wake-up system.
- 0 Interrupt from Wake is inhibited
- 1 When WAKE is set, request an interrupt

**TXE** Enables the transmitter unit.
- 0 Transmitter is disabled
- 1 Transmitter is enabled

**RXE** Enables the receiver unit.
- 0 Receiver is disabled
- 1 Receiver is enabled

**USARTEN** Activate USART module by supplying clock.
- 0 USART is disabled (clock is halted)
- 1 USART is enabled

**U2X** This bit only has effect for the asynchronous operation and selects receiver sampling rate.
- 0 Normal asynchronous operation
- 1 Double Speed asynchronous operation

### UCTRL3 (USART Control 3 Register) E4H, FCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MASTER | LOOPS | DISXCK | SPISS | - | USBS | TX8 | RX8 |
| R/W | R/W | R/W | R/W | - | R/W | R/W | R/W |

Initial value : $00_H$

**MASTER** Selects master or slave in SPI or Synchronous mode operation and controls the direction of XCK pin.
- 0 Slave mode operation and XCK is input pin.
- 1 Master mode operation and XCK is output pin

**LOOPS** Controls the Loop Back mode of USART, for test mode
- 0 Normal operation
- 1 Loop Back mode

**DISXCK** In Synchronous mode of operation, selects the waveform of XCK output.
- 0 XCK is free-running while USART is enabled in synchronous master mode.

1 XCK is active while any frame is on transferring.

**SPISS** Controls the functionality of SS pin in master SPI mode.

0 SS pin is normal GPIO or other primary function

1 SS output to other slave device

**USBS** Selects the length of stop bit in Asynchronous or Synchronous mode of operation.

0 1 Stop Bit

1 2 Stop Bit

**TX8** The ninth bit of data frame in Asynchronous or Synchronous mode of operation. Write this bit first before loading the UDATA register.

0 MSB ($9^{th}$ bit) to be transmitted is '0'

1 MSB ($9^{th}$ bit) to be transmitted is '1'

**RX8** The ninth bit of data frame in Asynchronous or Synchronous mode of operation. Read this bit first before reading the receive buffer.

0 MSB ($9^{th}$ bit) received is '0'

1 MSB ($9^{th}$ bit) received is '1'

### USTAT (USART Status Register) E5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UDRE | TXC | RXC | WAKE | SOFTRST | DOR | FE | PE |
| R/W | R/W | R/W | R/W | R/W | R | R | R |

Initial value : $80_H$

**UDRE** The UDRE flag indicates if the transmit buffer (UDATA) is ready to be loaded with new data. If UDRE is '1', it means the transmit buffer is empty and can hold one or two new data. This flag can generate an UDRE interrupt. Writing '0' to this bit position will clear UDRE flag.

0 Transmit buffer is not empty.

1 Transmit buffer is empty.

**TXC** This flag is set when the entire frame in the transmit shift register has been shifted out and there is no new data currently present in the transmit buffer. This flag is automatically cleared when the interrupt service routine of a TXC interrupt is executed. It is also cleared by writing '0' to this bit position. This flag can generate a TXC interrupt.

0 Transmission is ongoing.

1 Transmit buffer is empty and the data in transmit shift register are shifted out completely.

**RXC** This flag is set when there are unread data in the receive buffer and cleared when all the data in the receive buffer are read. The RXC flag can be used to generate a RXC interrupt.

0 There is no data unread in the receive buffer

1 There are more than 1 data in the receive buffer

**WAKE** This flag is set when the RX pin is detected low while the CPU is in stop mode. This flag can be used to generate a WAKE interrupt. This bit is set only when in asynchronous mode of operation. NOTE

0 No WAKE interrupt is generated.

1 WAKE interrupt is generated.

**SOFTRST** This is an internal reset and only has effect on USART. Writing '1' to this bit initializes the internal logic of USART and is auto cleared.

0 No operation

1 Reset USART

**DOR** This bit is set if a Data OverRun occurs. While this bit is set, the incoming data frame is ignored. This flag is valid until the receive buffer

is read.

0 No Data OverRun

1 Data OverRun detected

**FE** This bit is set if the first stop bit of next character in the receive buffer is detected as ‘0’. This bit is valid until the receive buffer is read.

0 No Frame Error

1 Frame Error detected

**PE** This bit is set if the next character in the receive buffer has a Parity Error when received while Parity Checking is enabled. This bit is valid until the receive buffer is read.

0 No Parity Error

1 Parity Error detected

NOTE When the WAKE function of USART is used as a release source from STOP mode, it is required to clear this bit in the RX interrupt service routine. Else the device will not wake-up from STOP mode again by the change of RX pin.

### UBAUD (USART Baud-Rate Generation Register) E6H, FEH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UBAUD7 | UBAUD6 | UBAUD5 | UBAUD4 | UBAUD3 | UBAUD2 | UBAUD1 | UBAUD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : $FF_H$

**UBAUD [7:0]** The value in this register is used to generate internal baud rate in asynchronous mode or to generate XCK clock in synchronous or spi mode. To prevent malfunction, do not write ‘0’ in asynchronous mode, and do not write ‘0’ or ‘1’ in synchronous or spi mode.

### UDATA (USART Data Register) E7H, FFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UDATA7 | UDATA6 | UDATA5 | UDATA4 | UDATA3 | UDATA2 | UDATA1 | UDATA0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : $FF_H$

**UDATA [7:0]** The USART Transmit Buffer and Receive Buffer share the same I/O address with this DATA register. The Transmit Data Buffer is the destination for data written to the UDATA register. Reading the UDATA register returns the contents of the Receive Buffer.

Write this register only when the UDRE flag is set. In spi or synchronous master mode, write this register even if TX is not enabled to generate clock, XCK.

### 11.7.14 Baud Rate setting (example)

Table 11-17 Examples of UBAUD Settings for Commonly Used Oscillator Frequencies

| Baud Rate | fOSC=1.00MHz | | | | fOSC=1.8432MHz | | | | fOSC=2.00MHz | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | U2X=0 | | U2X=1 | | U2X=0 | | U2X=1 | | U2X=0 | | U2X=1 | |
| | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR |
| 2400 | 25 | 0.2% | 51 | 0.2% | 47 | 0.0% | 95 | 0.0% | 51 | 0.2% | 103 | 0.2% |
| 4800 | 12 | 0.2% | 25 | 0.2% | 23 | 0.0% | 47 | 0.0% | 25 | 0.2% | 51 | 0.2% |
| 9600 | 6 | -7.0% | 12 | 0.2% | 11 | 0.0% | 23 | 0.0% | 12 | 0.2% | 25 | 0.2% |
| 14.4K | 3 | 8.5% | 8 | -3.5% | 7 | 0.0% | 15 | 0.0% | 8 | -3.5% | 16 | 2.1% |
| 19.2K | 2 | 8.5% | 6 | -7.0% | 5 | 0.0% | 11 | 0.0% | 6 | -7.0% | 12 | 0.2% |
| 28.8K | 1 | 8.5% | 3 | 8.5% | 3 | 0.0% | 7 | 0.0% | 3 | 8.5% | 8 | -3.5% |
| 38.4K | 1 | -18.6% | 2 | 8.5% | 2 | 0.0% | 5 | 0.0% | 2 | 8.5% | 6 | -7.0% |
| 57.6K | - | - | 1 | 8.5% | 1 | -25.0% | 3 | 0.0% | 1 | 8.5% | 3 | 8.5% |
| 76.8K | - | - | 1 | -18.6% | 1 | 0.0% | 2 | 0.0% | 1 | -18.6% | 2 | 8.5% |
| 115.2 K | - | - | - | - | - | - | 1 | 0.0% | - | - | 1 | 8.5% |
| 230.4 K | - | - | - | - | - | - | - | - | - | - | - | - |

| Baud Rate | fOSC=3.6864MHz | | | | fOSC=4.00MHz | | | | fOSC=7.3728MHz | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | U2X=0 | | U2X=1 | | U2X=0 | | U2X=1 | | U2X=0 | | U2X=1 | |
| | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR |
| 2400 | 95 | 0.0% | 191 | 0.0% | 103 | 0.2% | 207 | 0.2% | 191 | 0.0% | - | - |
| 4800 | 47 | 0.0% | 95 | 0.0% | 51 | 0.2% | 103 | 0.2% | 95 | 0.0% | 191 | 0.0% |
| 9600 | 23 | 0.0% | 47 | 0.0% | 25 | 0.2% | 51 | 0.2% | 47 | 0.0% | 95 | 0.0% |
| 14.4K | 15 | 0.0% | 31 | 0.0% | 16 | 2.1% | 34 | -0.8% | 31 | 0.0% | 63 | 0.0% |
| 19.2K | 11 | 0.0% | 23 | 0.0% | 12 | 0.2% | 25 | 0.2% | 23 | 0.0% | 47 | 0.0% |
| 28.8K | 7 | 0.0% | 15 | 0.0% | 8 | -3.5% | 16 | 2.1% | 15 | 0.0% | 31 | 0.0% |
| 38.4K | 5 | 0.0% | 11 | 0.0% | 6 | -7.0% | 12 | 0.2% | 11 | 0.0% | 23 | 0.0% |
| 57.6K | 3 | 0.0% | 7 | 0.0% | 3 | 8.5% | 8 | -3.5% | 7 | 0.0% | 15 | 0.0% |
| 76.8K | 2 | 0.0% | 5 | 0.0% | 2 | 8.5% | 6 | -7.0% | 5 | 0.0% | 11 | 0.0% |
| 115.2K | 1 | 0.0% | 3 | 0.0% | 1 | 8.5% | 3 | 8.5% | 3 | 0.0% | 7 | 0.0% |
| 230.4K | - | - | 1 | 0.0% | - | - | 1 | 8.5% | 1 | 0.0% | 3 | 0.0% |
| 250K | - | - | 1 | -7.8% | - | - | 1 | 0.0% | 1 | -7.8% | 3 | -7.8% |
| 0.5M | - | - | - | - | - | - | - | - | - | - | 1 | -7.8% |

| Baud Rate | fOSC=8.00MHz | | | | fOSC=11.0592MHz | | | | fOSC=14.7456MHz | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | U2X=0 | | U2X=1 | | U2X=0 | | U2X=1 | | U2X=0 | | U2X=1 | |
| | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR | UBAUD | ERROR |
| 2400 | 207 | 0.2% | - | - | - | - | - | - | - | - | - | - |
| 4800 | 103 | 0.2% | 207 | 0.2% | 143 | 0.0% | - | - | 191 | 0.0% | - | - |
| 9600 | 51 | 0.2% | 103 | 0.2% | 71 | 0.0% | 143 | 0.0% | 95 | 0.0% | 191 | 0.0% |
| 14.4K | 34 | -0.8% | 68 | 0.6% | 47 | 0.0% | 95 | 0.0% | 63 | 0.0% | 127 | 0.0% |
| 19.2K | 25 | 0.2% | 51 | 0.2% | 35 | 0.0% | 71 | 0.0% | 47 | 0.0% | 95 | 0.0% |
| 28.8K | 16 | 2.1% | 34 | -0.8% | 23 | 0.0% | 47 | 0.0% | 31 | 0.0% | 63 | 0.0% |
| 38.4K | 12 | 0.2% | 25 | 0.2% | 17 | 0.0% | 35 | 0.0% | 23 | 0.0% | 47 | 0.0% |
| 57.6K | 8 | -3.5% | 16 | 2.1% | 11 | 0.0% | 23 | 0.0% | 15 | 0.0% | 31 | 0.0% |
| 76.8K | 6 | -7.0% | 12 | 0.2% | 8 | 0.0% | 17 | 0.0% | 11 | 0.0% | 23 | 0.0% |
| 115.2K | 3 | 8.5% | 8 | -3.5% | 5 | 0.0% | 11 | 0.0% | 7 | 0.0% | 15 | 0.0% |
| 230.4K | 1 | 8.5% | 3 | 8.5% | 2 | 0.0% | 5 | 0.0% | 3 | 0.0% | 7 | 0.0% |
| 250K | 1 | 0.0% | 3 | 0.0% | 2 | -7.8% | 5 | -7.8% | 3 | -7.8% | 6 | 5.3% |
| 0.5M | - | - | 1 | 0.0% | - | - | 2 | -7.8% | 1 | -7.8% | 3 | -7.8% |
| 1M | - | - | - | - | - | - | - | - | - | - | 1 | -7.8% |

## 11.8 SPI

### 11.8.1 Overview

There is Serial Peripheral Interface (SPI) one channel in Z51F0811. The SPI allows synchronous serial data transfer between the external serial devices. It can do Full-duplex communication by 4-wire (MOSI, MISO, SCK, SS), support Master/Slave mode, can select serial clock (SCK) polarity, phase and whether LSB first data transfer or MSB first data transfer.

### 11.8.2 Block Diagram

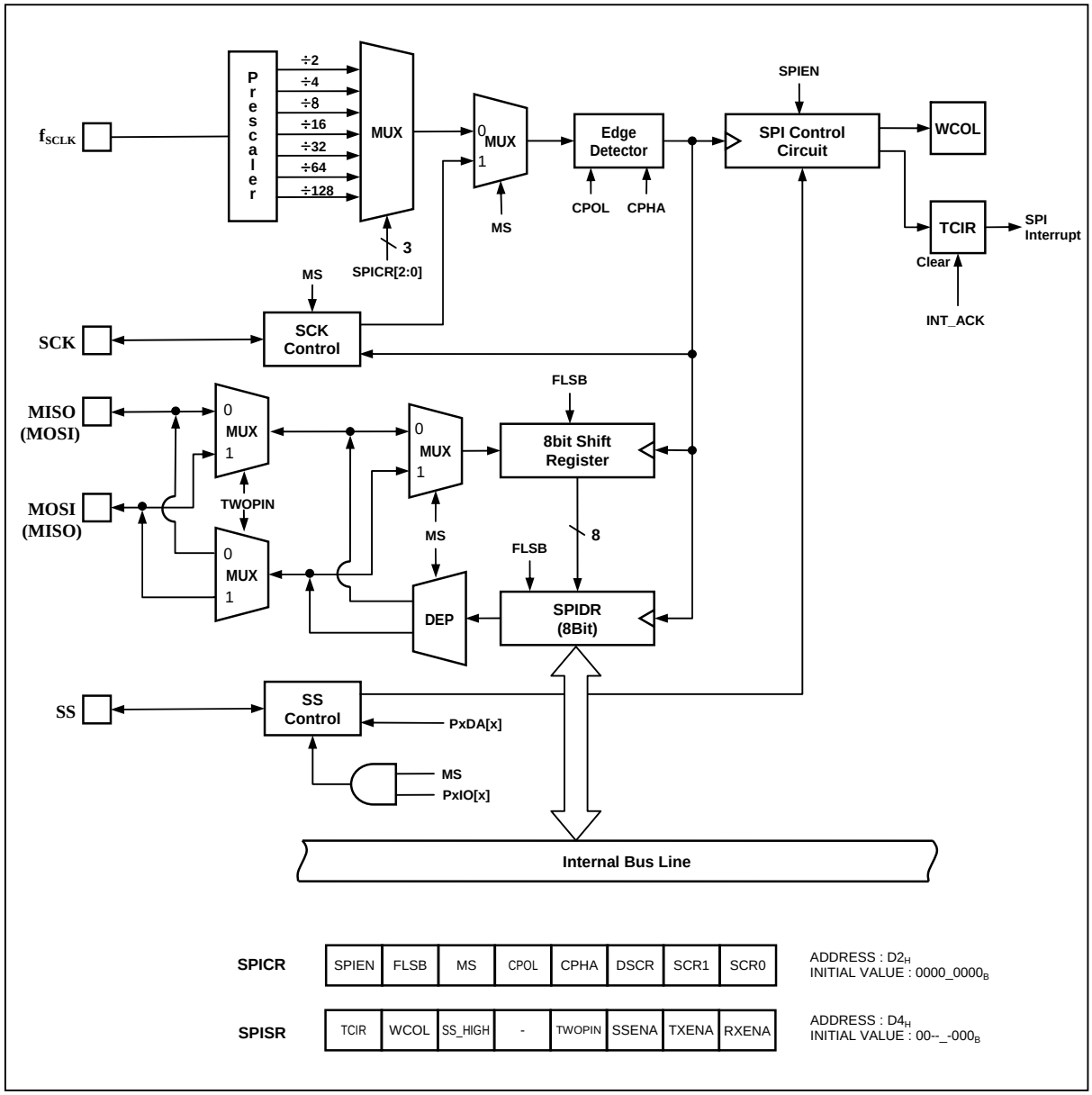



Figure 11-48 SPI Block Diagram

### 11.8.3 Data Transmit / Receive Operation

User can use SPI for serial data communication by following step

1. Select SPI operation mode(master/slave, polarity, phase) by control register SPICR.
2. When the SPI is configured as a Master, it selects a Slave by SS signal (active low).

   When the SPI is configured as a Slave, it is selected by SS signal incoming from Master
3. When the user writes a byte to the data register SPIDR, SPI will start an operation.
4. In this time, if the SPI is configured as a Master, serial clock will come out of SCK pin. And Master shifts the eight bits into the Slave (transmit), Slave shifts the eight bits into the Master at the same time (receive). If the SPI is configured as a Slave, serial clock will come into SCK pin. And Slave shifts the eight bits into the Master (transmit), Master shifts the eight bits into the Slave at the same time (receive).
5. When transmit/receive is done, TCIR (Transmit Complete or Interrupt Request) bit will be set. If the SPI interrupt is enabled, an interrupt is requested. And TCIR bit is cleared by hardware when executing the corresponding interrupt. If SPI interrupt is disable, TCIR bit is cleared when user read the status register SPISR, and then access (read/write) the data register SPIDR.

Note) If you want to use both transmit and receive, set the TXENA, RXENA bit of SPISR, and if user want to use only either transmit or receive, clear the TXENA or RXENA. In this case, user can use disabled pin by GPIO freely.

### 11.8.4 SS pin function

1. When the SPI is configured as a Slave, the SS pin is always input. If LOW signal come into SS pin, the SPI logic is active. And if ‘HIGH’ signal come into SS pin, the SPI logic is stop. In this time, SPI logic will be reset, and invalidated any received data.
2. When the SPI is configured as a Master, the user can select the direction of the SS pin by port direction register (PxIO[x]). If the SS pin is configured as an output, user can use general GPIO output mode. If the SS pin is configured as an input, ‘HIGH’ signal must come into SS pin to guarantee Master operation. If ‘LOW’ signal come into SS pin, the SPI logic interprets this as another master selecting the SPI as a slave and starting to send data to it. To avoid bus contention, MS bit of SPICR will be cleared and the SPI becomes a Slave and then, TCIR bit of SPISR will be set, and if the SPI interrupt is enabled, an interrupt is requested.

Note)

- When the SS pin is configured as an output at Master mode, SS pin’s output value is defined by user’s software (PxDA[x]). Before SPICR setting, the direction of SS pin must be defined
- If you don’t need to use SS pin, clear the SSENA bit of SPISR. So, you can use disabled pin by GPIO freely. In this case, SS signal is driven by ‘HIGH’ or ‘LOW’ internally. In other words, master is ‘HIGH’, salve is ‘LOW’
- When SS pin is configured as input(master or slave), if ‘HIGH’ signal come into SS pin, this flag bit will be set at the SS rising time. And you can clear it by writing ‘0’.

### 11.8.5 Timing Waveform

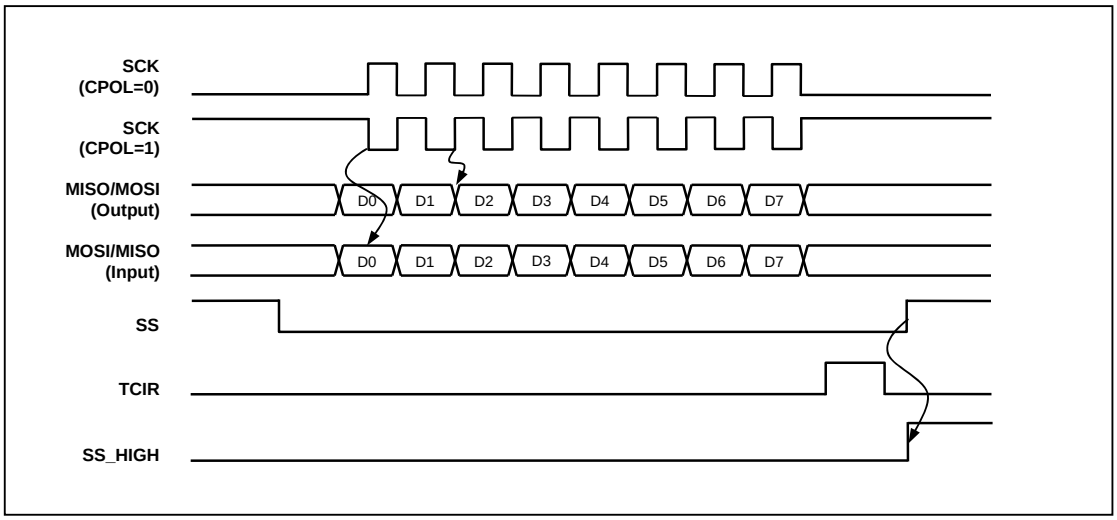



Figure 11-49 SPI Transmit/Receive Timing Diagram at CPHA = 0

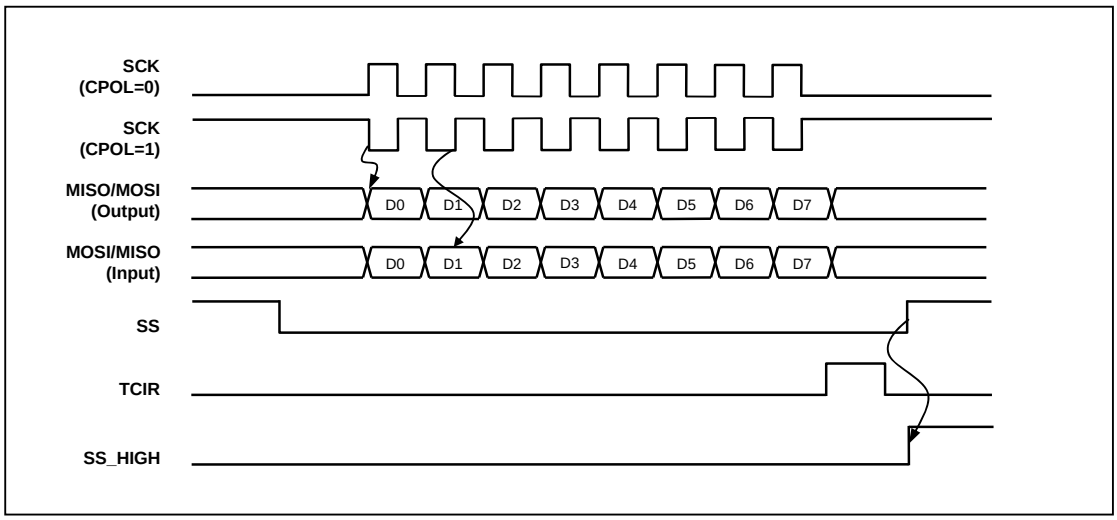



Figure 11-50 SPI Transmit/Receive Timing Diagram at CPHA = 1

### 11.8.6 Register Map

Table 11-18 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| SPICR | D2H | R/W | 0H | SPI Control Register |
| SPIDR | D3H | R/W | 0H | SPI Data Register |
| SPISR | D4H | R/W | 0H | SPI Status Register |

### 11.8.7 SPI Register description

The SPI Register consists of SPI Control Register (SPICR), SPI Status Register (SPISR) and SPI Data Register (SPIDR)

### 11.8.8 Register description for SPI

**SPICR (SPI Control Register) : D2H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SPIEN | FLSB | MS | CPOL | CPHA | DSCR | SCR1 | SCR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**SPIEN** This bit controls the SPI operation

0 SPI Disable

1 SPI Enable

**FLSB** This bit selects the data transmission sequence

0 MSB First

1 LSB First

**MS** This bit selects whether Master or Slave mode

0 Slave mode

1 Master mode

**CPOL**
**CPHA** These two bits control the serial clock (SCK) mode

Clock Polarity (CPOL) bit determine SCK's value at idle mode

Clock Phase (CPHA) bit determine if data is sampled on the leading or trailing edge of SCK. Refer to Figure 11-49, Figure 11-50

| CPOL | CPHA | Leading Edge | Trailing Edge |
|---|---|---|---|
| 0 | 0 | Sample (Rising) | Setup (Falling) |
| 0 | 1 | Setup (Rising) | Sample (Falling) |
| 1 | 0 | Sample (Falling) | Setup (Rising) |
| 1 | 1 | Setup (Falling) | Sample (Rising) |

**DSCR**
**SCR[2:0]** These three bits select the SCK rate of the device configured as a Master. When DSCR bit is written one, SCK will be doubled in Master mode.

fx– Main system clock oscillation frequency.

| DSCR | SCR1 | SCR0 | SCK frequency |
|---|---|---|---|
| 0 | 0 | 0 | fx/4 |
| 0 | 0 | 1 | fx/16 |
| 0 | 1 | 0 | fx/64 |
| 0 | 1 | 1 | fx/128 |
| 1 | 0 | 0 | fx/2 |
| 1 | 0 | 1 | fx/8 |
| 1 | 1 | 0 | fx/32 |
| 1 | 1 | 1 | fx/64 |

### SPIDR (SPI Data Register) : D3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SPIDR7 | SPIDR6 | SPIDR5 | SPIDR4 | SPIDR3 | SPIDR2 | SPIDR1 | SPIDR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**SPIDR [7:0]** SPI data register.
Although you only use reception, user must write any data in here to start the SPI operation.

### SPISR (SPI Status Register) : D4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TCIR | WCOL | SS_HIGH | - | TWOPIN | SSENA | TXENA | RXENA |
| R | R | R/W | - | R/W | R/W | R/W | R/W |

Initial value : 00H

**TCIR** When a serial data transmission is complete, the TCIR bit is set. If the SPI interrupt is enabled, an interrupt is requested. And TCIR bit is cleared by hardware when executing the corresponding interrupt. If SPI interrupt is disable, TCIR bit is cleared when user read the status register SPISR, and then access (read/write) the data register SPIDR.

0 Interrupt cleared

1 Transmission Complete and Interrupt Requested

**WCOL** This bit is set if the data register SPIDR is written during a data transfer. This bit is cleared when user read the status register SPISR, and then access (read/write) the data register SPIDR.

0 No collision

1 Write Collision

**SS_HIGH** When SS pin is configured as input(master or slave), if ‘HIGH’ signal come into SS pin, this flag bit will be set at the SS rising time. And you can clear it by writing ‘0’.
You can write only zero.

0 Flag is cleared

1 Flag is set

**TWOPIN** This bit controls the 2 pin operation.
In master mode,

0 Disable

1 Enable

**SSENA** This bit controls the SS pin operation

0 Disable

1 Enable

**TXENA** This bit controls a data transfer operation

0 Disable

1 Enable

**RXENA** This bit controls a data reception operation

0 Disable

1 Enable

Note that if the MS is set to ‘0’, when TWOPIN is set to ‘0’, port 03 is set to MISO and if the MS is set to ‘0’, when TWOPIN is set to ‘1’, port 02 is set to MOSI. But if the MS is set to ‘1’, when TWOPIN is set to ‘0’, port 03 is set to MOSI and if the MS is set to ‘1’, when TWOPIN is set to ‘1’, port 02 is set to MISO.

## 11.9 I²C

### 11.9.1 Overview

The I²C is one of industrial standard serial communication protocols, and which uses 2 bus lines Serial Data Line (SDA) and Serial Clock Line (SCL) to exchange data. Because both SDA and SCL lines are open-drain output, each line needs pull-up resistor. The features are as shown below.

- Compatible with I²C bus standard
- Multi-master operation
- Up to 400 KHz data transfer speed
- 7 bit address
- Support  two slave addresses
- Both master and slave operation
- Bus busy detection

### 11.9.2 Block Diagram

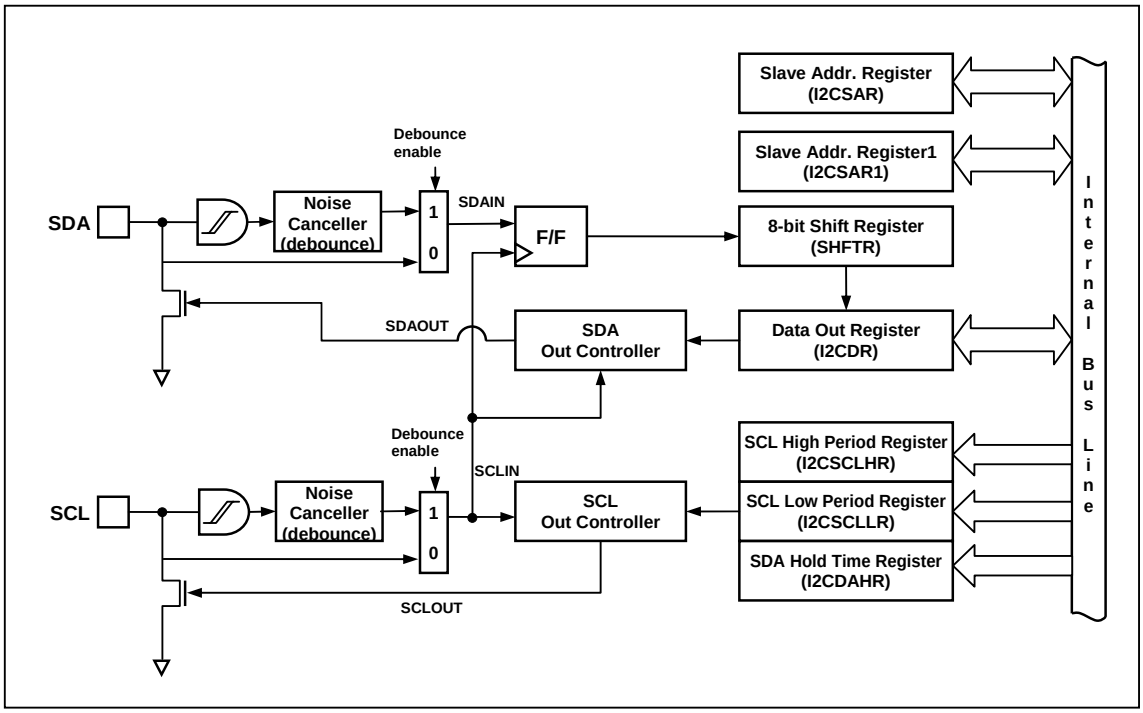



Figure 11-51 I²C Block Diagram

### 11.9.3 I²C Bit Transfer

The data on the SDA line must be stable during HIGH period of the clock, SCL. The HIGH or LOW state of the data line can only change when the clock signal on the SCL line is LOW. The exceptions are START(S), repeated START(Sr) and STOP(P) condition where data line changes when clock line is high.

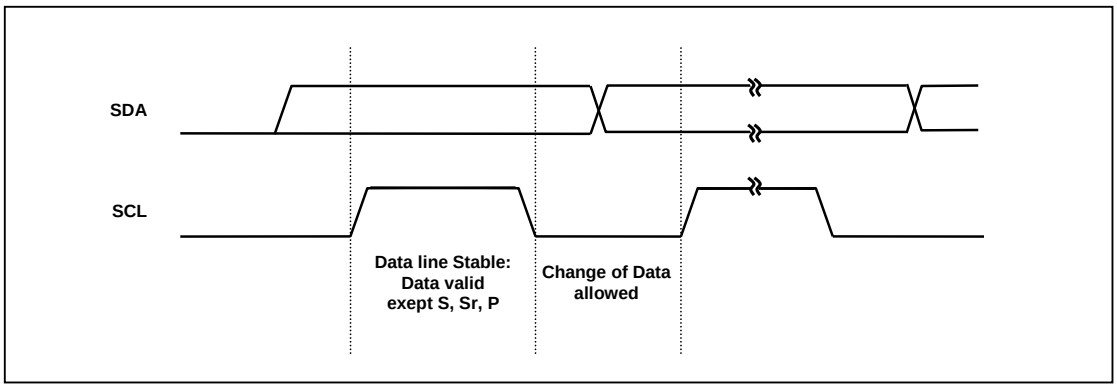



Figure 11-52 Bit Transfer on the I²C-Bus

### 11.9.4 Start / Repeated Start / Stop

One master can issue a START (S) condition to notice other devices connected to the SCL, SDA lines that it will use the bus. A STOP (P) condition is generated by the master to release the bus lines so that other devices can use it.

A high to low transition on the SDA line while SCL is high defines a START (S) condition.

A low to high transition on the SDA line while SCL is high defines a STOP (P) condition.

START and STOP conditions are always generated by the master. The bus is considered to be busy after START condition. The bus is considered to be free again after STOP condition, ie, the bus is busy between START and STOP condition. If a repeated START condition (Sr) is generated instead of STOP condition, the bus stays busy. So, the START and repeated START conditions are functionally identical.

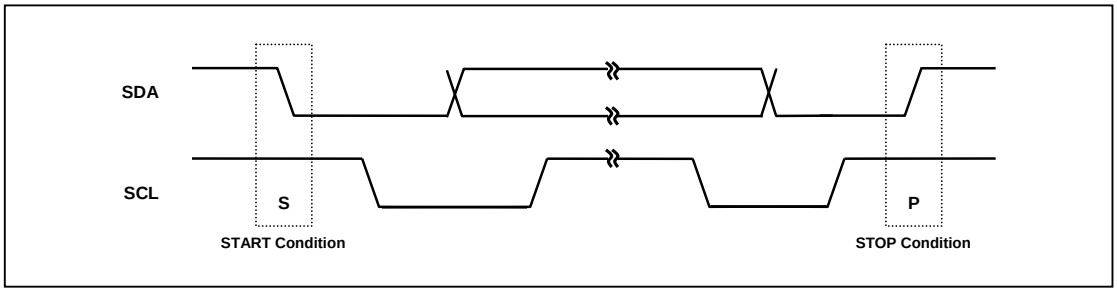



Figure 11-53 START and STOP Condition

### 11.9.5 Data Transfer

Every byte put on the SDA line must be 8-bits long. The number of bytes that can be transmitted per transfer is unlimited. Each byte has to be followed by an acknowledge bit. Data is transferred with

the most significant bit (MSB) first. If a slave can't receive or transmit another complete byte of data until it has performed some other function, it can hold the clock line SCL LOW to force the master into a wait state. Data transfer then continues when the slave is ready for another byte of data and releases clock line SCL.

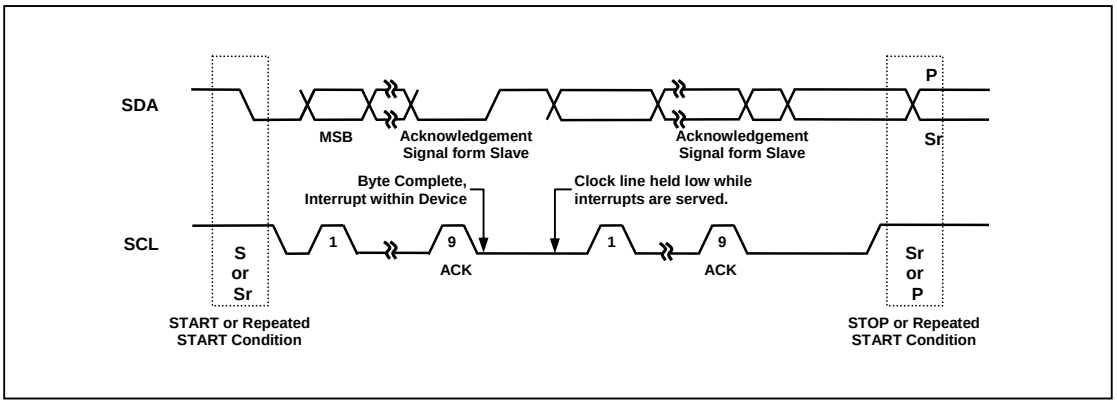



Figure 11-54 STOP or Repeated START Condition

### 11.9.6 Acknowledge

The acknowledge related clock pulse is generated by the master. The transmitter releases the SDA line (HIGH) during the acknowledge clock pulse. The receiver must pull down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse. When a slave is addressed by a master (Address Packet), and if it is unable to receive or transmit because it's performing some real time function, the data line must be left HIGH by the slave. And also, when a slave addressed by a master is unable to receive more data bits, the slave receiver must release the SDA line (Data Packet). The master can then generate either a STOP condition to abort the transfer, or a repeated START condition to start a new transfer.

If a master receiver is involved in a transfer, it must signal the end of data to the slave transmitter by not generating an acknowledge on the last byte that was clocked out of the slave. The slave transmitter must release the data line to allow the master to generate a STOP or repeated START condition.

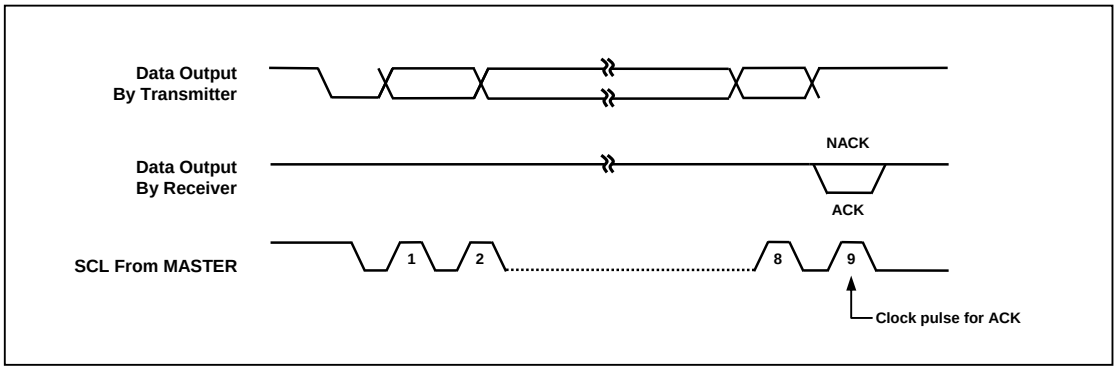



Figure 11-55 Acknowledge on the $I^2C$-Bus

### 11.9.7 Synchronization / Arbitration

Clock synchronization is performed using the wired-AND connection of $I^2C$ interfaces to the SCL line. This means that a HIGH to LOW transition on the SCL line will cause the devices concerned to start counting off their LOW period and it will hold the SCL line in that state until the clock HIGH state is reached. However the LOW to HIGH transition of this clock may not change the state of the SCL line if another clock is still within its LOW period. In this way, a synchronized SCL clock is generated with its LOW period determined by the device with the longest clock LOW period, and its HIGH period determined by the one with the shortest clock HIGH period.

A master may start a transfer only if the bus is free. Two or more masters may generate a START condition. Arbitration takes place on the SDA line, while the SCL line is at the HIGH level, in such a way that the master which transmits a HIGH level, while another master is transmitting a LOW level will switch off its DATA output state because the level on the bus doesn't correspond to its own level. Arbitration continues for many bits until a winning master gets the ownership of $I^2C$ bus. Its first stage is comparison of the address bits.

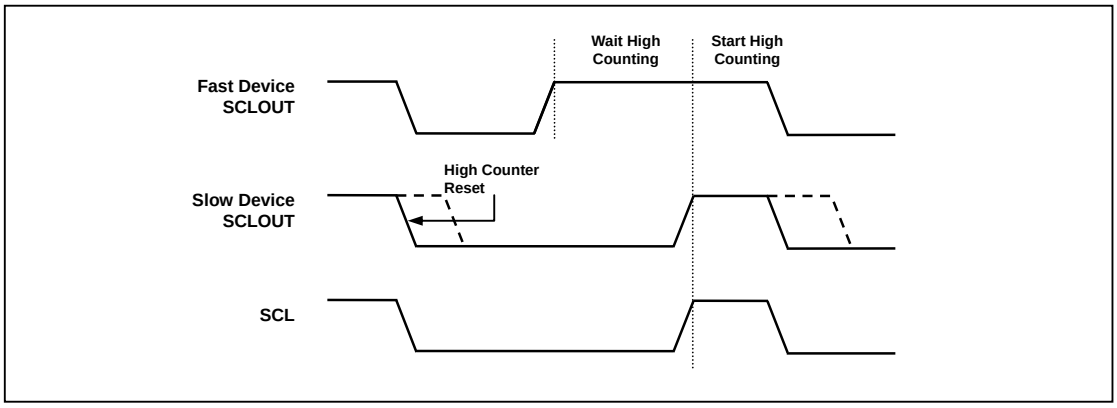



Figure 11-56 Clock Synchronization during Arbitration Procedure

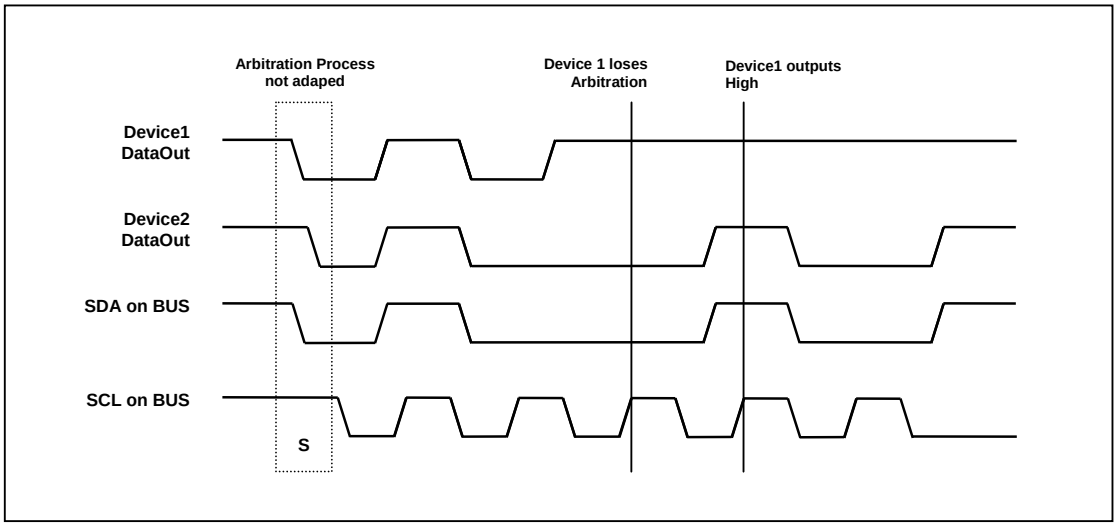



Figure 11-57 Arbitration Procedure of Two Masters

### 11.9.8 Operation

The I²C is byte-oriented and interrupt based. Interrupts are issued after all bus events except for a transmission of a START condition. Because the I²C is interrupt based, the application software is free to carry on other operations during a I²C byte transfer.

Note that when a I²C interrupt is generated, IIF flag in I2CMR register is set, it is cleared by writing an arbitrary value to I2CSR. When I²C interrupt occurs, the SCL line is hold LOW until writing any value to I2CSR. When the IIF flag is set, the I2CSR contains a value indicating the current state of the I²C bus. According to the value in I2CSR, software can decide what to do next.

I²C can operate in 4 modes by configuring master/slave, transmitter/receiver. The operating mode is configured by a winning master. A more detailed explanation follows below.

#### 11.9.8.1 Master Transmitter

To operate I²C in master transmitter, follow the recommended steps below.

1. Enable I²C by setting IICEN bit in I2CMR. This provides main clock to the peripheral.
2. Load SLA+W into the I2CDR where SLA is address of slave device and W is transfer direction from the viewpoint of the master. For master transmitter, W is ‘0’. Note that I2CDR is used for both address and data.
3. Configure baud rate by writing desired value to both I2CSCLLR and I2CSCLHR for the Low and High period of SCL line.
4. Configure the I2CSDAHR to decide when SDA changes value from falling edge of SCL. If SDA should change in the middle of SCL LOW period, load half the value of I2CSCLLR to the I2CSDAHR.
5. Set the START bit in I2CMR. This transmits a START condition. And also configure how to handle interrupt and ACK signal. When the START bit is set, 8-bit data in I2CDR is transmitted out according to the baud-rate.
6. This is ACK signal processing stage for address packet transmitted by master. When 7-bit address and 1-bit transfer direction is transmitted to target slave device, the master can know whether the slave acknowledged or not in the 9th high period of SCL. If the master gains bus mastership, I²C generates GCALL interrupt regardless of the reception of ACK from the slave device. When I²C loses bus mastership during arbitration process, the MLOST bit in I2CSR is set, and I²C waits in idle state or can be operate as an addressed slave. To operate as a slave when the MLSOT bit in I2CSR is set, the ACKEN bit in I2CMR must be set and the received 7-bit address must equal to the SLA bits in I2CSAR. In this case I²C operates as a slave transmitter or a slave receiver (go to appropriate section). In this stage, I²C holds the SCL LOW. This is because to decide whether I²C continues serial transfer or stops communication. The following steps continue assuming that I²C does not lose mastership during first data transfer.

   I²C (Master) can choose one of the following cases regardless of the reception of ACK signal from slave.

   1) Master receives ACK signal from slave, so continues data transfer because slave can receive more data from master. In this case, load data to transmit to I2CDR.
   2) Master stops data transfer even if it receives ACK signal from slave. In this case, set the STOP bit in I2CMR.
   3) Master transmits repeated START condition with not checking ACK signal. In this case, load SLA+R/W into the I2CDR and set START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CDR and if transfer direction bit is ‘1’ go to master receiver section.

7. 1-Byte of data is being transmitted. During data transfer, bus arbitration continues.
8. This is ACK signal processing stage for data packet transmitted by master. $I^2C$ holds the SCL LOW. When $I^2C$ loses bus mastership while transmitting data arbitrating other masters, the MLOST bit in I2CSR is set. If then, $I^2C$ waits in idle state. When the data in I2CDR is transmitted completely, $I^2C$ generates TEND interrupt.

   $I^2C$ can choose one of the following cases regardless of the reception of ACK signal from slave.

   1) Master receives ACK signal from slave, so continues data transfer because slave can receive more data from master. In this case, load data to transmit to I2CDR.
   2) Master stops data transfer even if it receives ACK signal from slave. In this case, set the STOP bit in I2CMR.
   3) Master transmits repeated START condition with not checking ACK signal. In this case, load SLA+R/W into the I2CDR and set the START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CDR, and if transfer direction bit is '1' go to master receiver section.
9. This is the final step for master transmitter function of $I^2C$, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, $I^2C$ enters idle state.

The next figure depicts above process for master transmitter operation of $I^2C$.

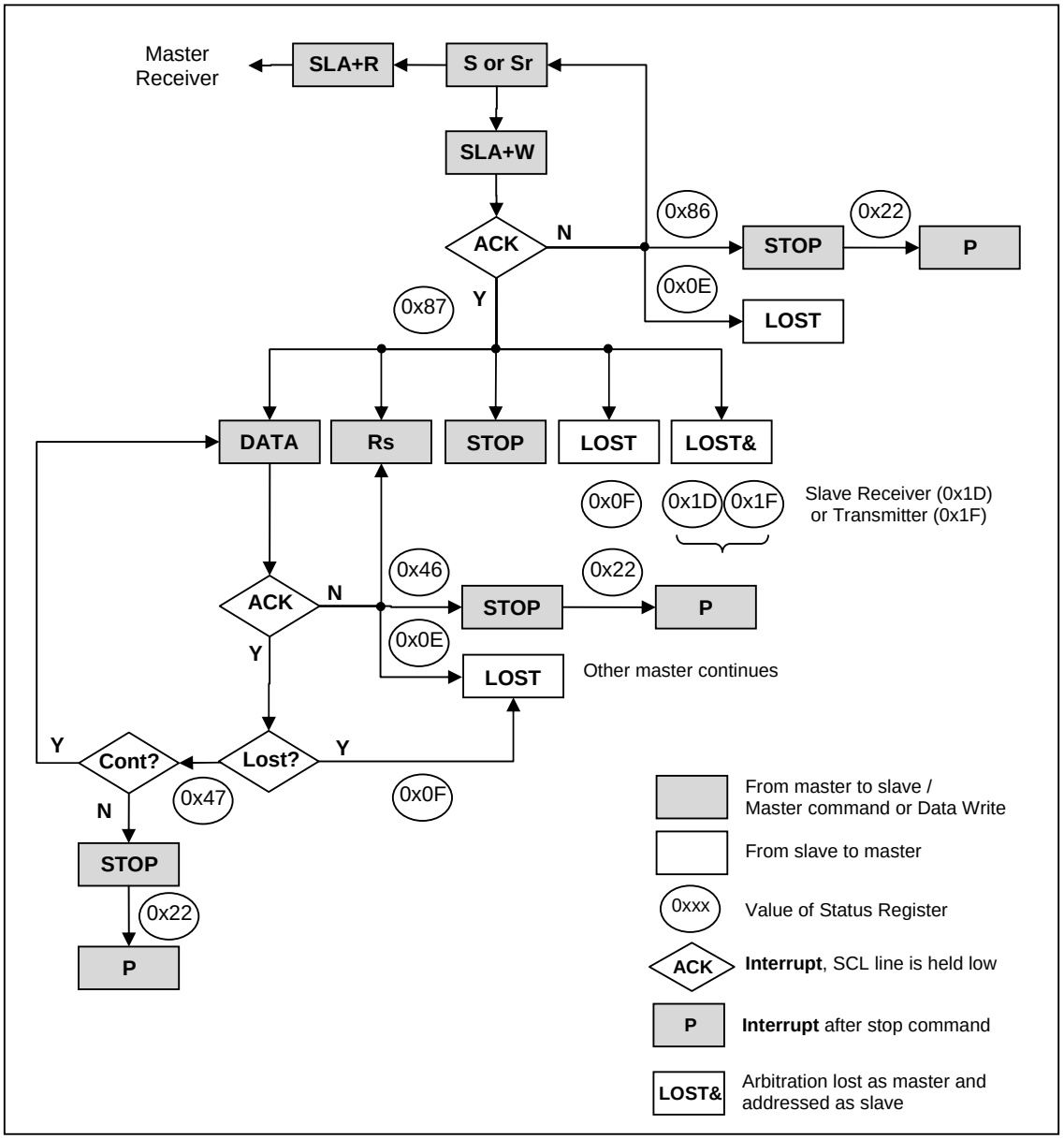



Figure 11-58 Formats and States in the Master Transmitter Mode

#### 11.9.8.2 Master Receiver

To operate $I^2C$ in master receiver, follow the recommended steps below.

1. Enable $I^2C$ by setting IICEN bit in I2CMR. This provides main clock to the peripheral.
2. Load SLA+R into the I2CDR where SLA is address of slave device and R is transfer direction from the viewpoint of the master. For master receiver, R is '1'. Note that I2CDR is used for both address and data.
3. Configure baud rate by writing desired value to both I2CSCLLR and I2CSCLHR for the Low and High period of SCL line.
4. Configure the I2CSDAHR to decide when SDA changes value from falling edge of SCL. If SDA should change in the middle of SCL LOW period, load half the value of I2CSCLLR to the I2CSDAHR.
5. Set the START bit in I2CMR. This transmits a START condition. And also configure how to handle interrupt and ACK signal. When the START bit is set, 8-bit data in I2CDR is transmitted out according to the baud-rate.
6. This is ACK signal processing stage for address packet transmitted by master. When 7-bit address and 1-bit transfer direction is transmitted to target slave device, the master can know whether the slave acknowledged or not in the 9$^{th}$ high period of SCL. If the master gains bus mastership, $I^2C$ generates GCALL interrupt regardless of the reception of ACK from the slave device. When $I^2C$ loses bus mastership during arbitration process, the MLOST bit in I2CSR is set, and $I^2C$ waits in idle state or can be operate as an addressed slave. To operate as a slave when the MLSOT bit in I2CSR is set, the ACKEN bit in I2CMR must be set and the received 7-bit address must equal to the SLA bits in I2CSAR. In this case $I^2C$ operates as a slave transmitter or a slave receiver (go to appropriate section). In this stage, $I^2C$ holds the SCL LOW. This is because to decide whether $I^2C$ continues serial transfer or stops communication. The following steps continue assuming that $I^2C$ does not lose mastership during first data transfer.

   $I^2C$ (Master) can choose one of the following cases according to the reception of ACK signal from slave.

   1) Master receives ACK signal from slave, so continues data transfer because slave can prepare and transmit more data to master. Configure ACKEN bit in I2CMR to decide whether $I^2C$ ACKnowledges the next data to be received or not.
   2) Master stops data transfer because it receives no ACK signal from slave. In this case, set the STOP bit in I2CMR.
   3) Master transmits repeated START condition due to no ACK signal from slave. In this case, load SLA+R/W into the I2CDR and set START bit in I2CMR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1), move to step 7. In case of 2), move to step 9 to handle STOP interrupt. In case of 3), move to step 6 after transmitting the data in I2CDR and if transfer direction bit is '0' go to master transmitter section.

7. 1-Byte of data is being received.
8. This is ACK signal processing stage for data packet transmitted by slave. $I^2C$ holds the SCL LOW. When 1-Byte of data is received completely, $I^2C$ generates TEND interrupt.

   $I^2C$ can choose one of the following cases according to the RXACK flag in I2CSR.

   1) Master continues receiving data from slave. To do this, set ACKEN bit in I2CMR to ACKnowledge the next data to be received.
   2) Master wants to terminate data transfer when it receives next data by not generating ACK signal. This can be done by clearing ACKEN bit in I2CMR.
   3) Because no ACK signal is detected, master terminates data transfer. In this case, set the STOP bit in I2CMR.
   4) No ACK signal is detected, and master transmits repeated START condition. In this case,

load SLA+R/W into the I2CDR and set the START bit in I2CMR.

After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1) and 2), move to step 7. In case of 3), move to step 9 to handle STOP interrupt. In case of 4), move to step 6 after transmitting the data in I2CDR, and if transfer direction bit is '0' go to master transmitter section.

9. This is the final step for master receiver function of $I^2C$, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, $I^2C$ enters idle state.

The processes described above for master receiver operation of $I^2C$ can be depicted as the following figure.

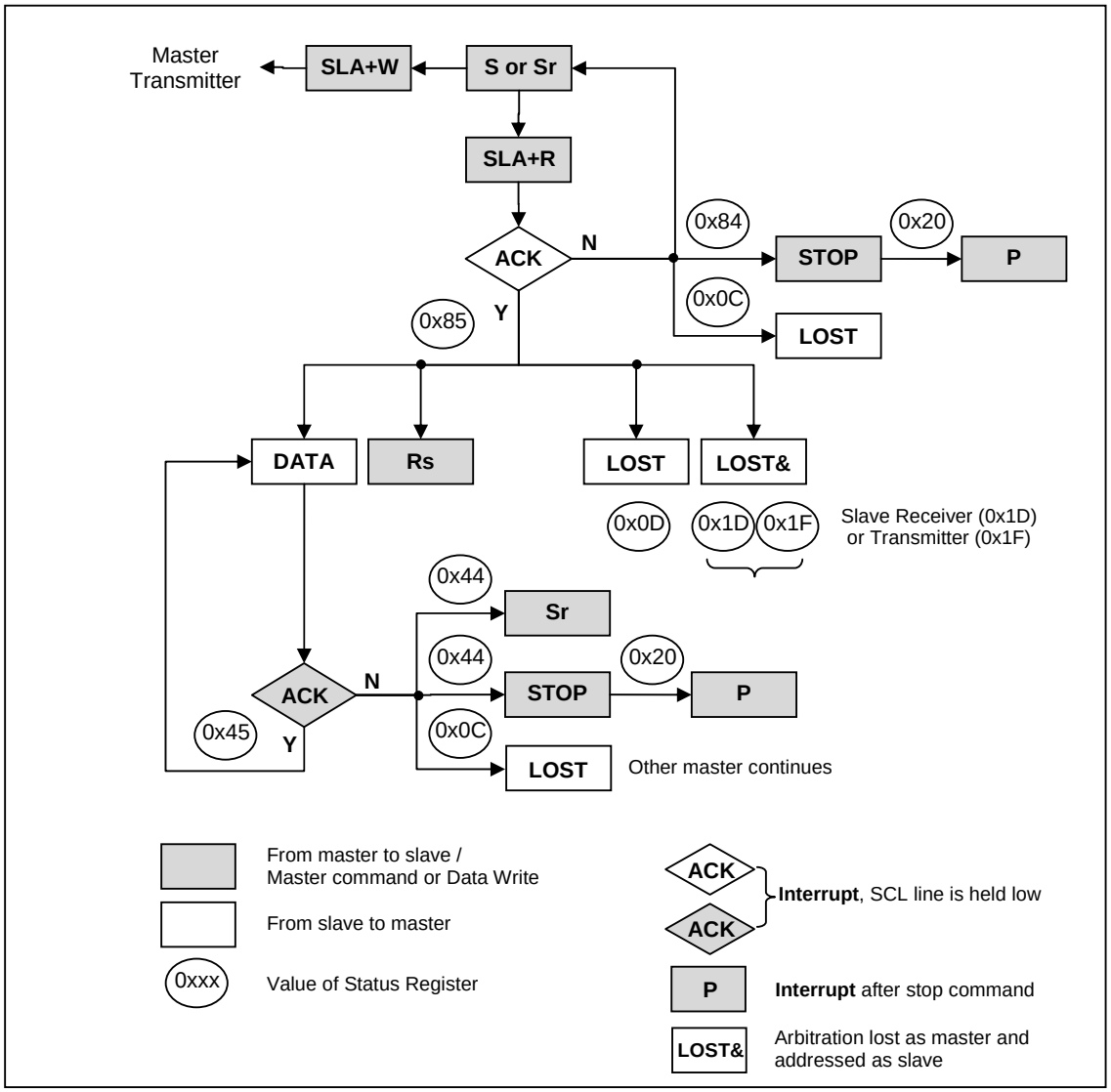



Figure 11-59 Formats and States in the Master Receiver Mode

#### 11.9.8.3 Slave Transmitter

To operate $I^2C$ in slave transmitter, follow the recommended steps below.

1. If the main operating clock (SCLK) of the system is slower than that of SCL, load value 0x00 into I2CSDAHR to make SDA change within one system clock period from the falling edge of SCL. Note that the hold time of SDA is calculated by SDAH x period of SCLK where SDAH is multiple of number of SCLK coming from I2CSDAHR. When the hold time of SDA is longer than the period of SCLK, $I^2C$ (slave) cannot transmit serial data properly.
2. Enable $I^2C$ by setting IICEN bit and INTEN bit in I2CMR. This provides main clock to the peripheral.
3. When a START condition is detected, $I^2C$ receives one byte of data and compares it with SLA bits in I2CSAR. If the GCALLEN bit in I2CSAR is enabled, $I^2C$ compares the received data with value 0x00, the general call address.
4. If the received address does not equal to SLA bits in I2CSAR, $I^2C$ enters idle state ie, waits for another START condition. Else if the address equals to SLA bits and the ACKEN bit is enabled, $I^2C$ generates SSEL interrupt and the SCL line is held LOW. Note that even if the address equals to SLA bits, when the ACKEN bit is disabled, $I^2C$ enters idle state. When SSEL interrupt occurs, load transmit data to I2CDR and write arbitrary value to I2CSR to release SCL line.
5. 1-Byte of data is being transmitted.
6. In this step, $I^2C$ generates TEND interrupt and holds the SCL line LOW regardless of the reception of ACK signal from master. Slave can select one of the following cases.

   1) No ACK signal is detected and $I^2C$ waits STOP or repeated START condition.
   2) ACK signal from master is detected. Load data to transmit into I2CDR.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1) move to step 7 to terminate communication. In case of 2) move to step 5. In either case, a repeated START condition can be detected. For that case, move step 4.
7. This is the final step for slave transmitter function of $I^2C$, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, $I^2C$ enters idle state.

The next figure shows flow chart for handling slave transmitter function of $I^2C$.

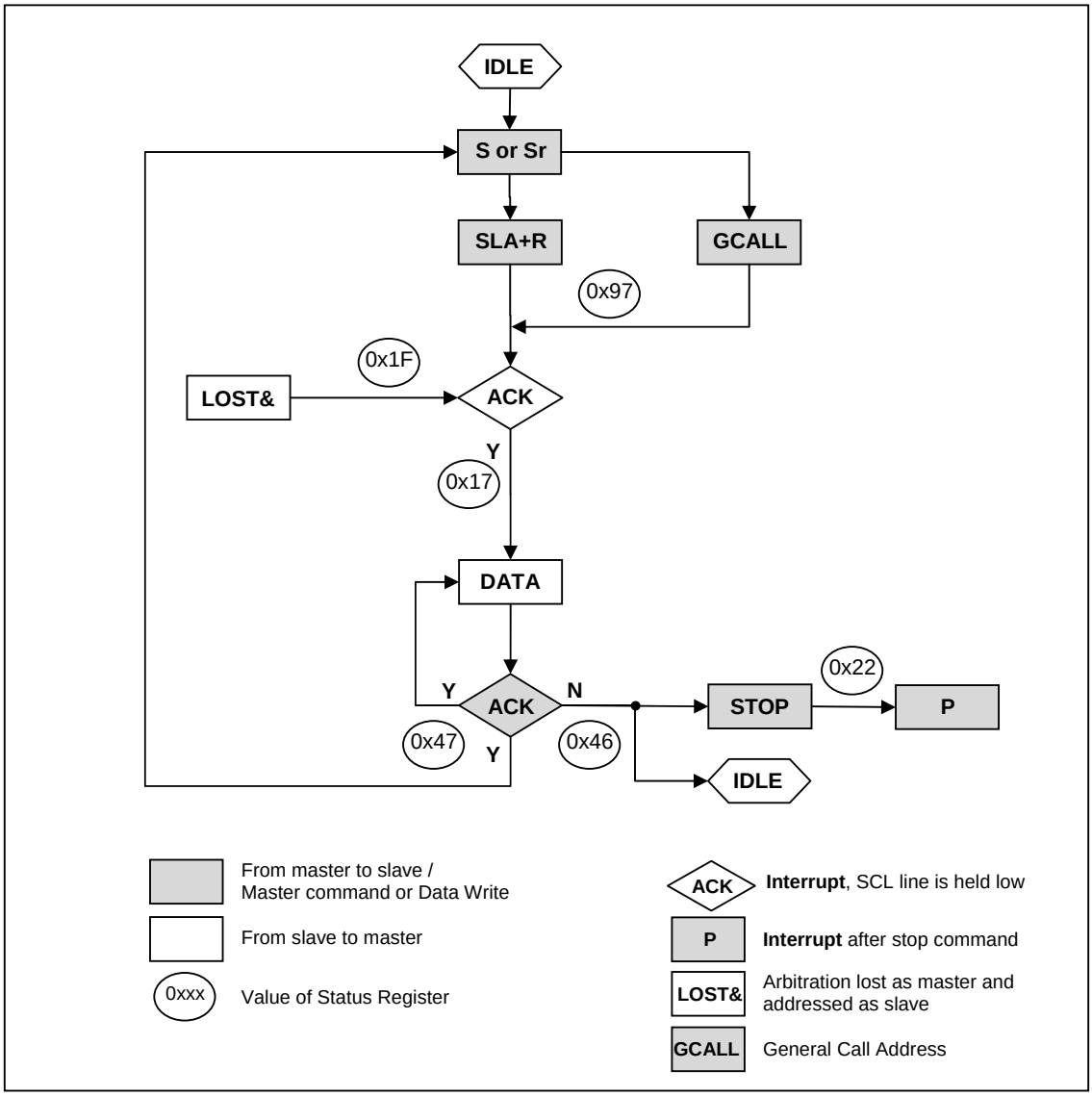



Figure 11-60 Formats and States in the Slave Transmitter Mode

#### 11.9.8.4 Slave Receiver

To operate $I^2C$ in slave receiver, follow the recommended steps below.

1. If the main operating clock (SCLK) of the system is slower than that of SCL, load value 0x00 into I2CSDAHR to make SDA change within one system clock period from the falling edge of SCL. Note that the hold time of SDA is calculated by SDAH x period of SCLK where SDAH is multiple of number of SCLK coming from I2CSDAHR. When the hold time of SDA is longer than the period of SCLK, $I^2C$ (slave) cannot transmit serial data properly.
2. Enable $I^2C$ by setting IICEN bit and INTEN bit in I2CMR. This provides main clock to the peripheral.
3. When a START condition is detected, $I^2C$ receives one byte of data and compares it with SLA bits in I2CSAR. If the GCALLEN bit in I2CSAR is enabled, $I^2C$ compares the received data with value 0x00, the general call address.
4. If the received address does not equal to SLA bits in I2CSAR, $I^2C$ enters idle state ie, waits for another START condition. Else if the address equals to SLA bits and the ACKEN bit is enabled, $I^2C$ generates SSEL interrupt and the SCL line is held LOW. Note that even if the address equals to SLA bits, when the ACKEN bit is disabled, $I^2C$ enters idle state. When SSEL interrupt occurs and $I^2C$ is ready to receive data, write arbitrary value to I2CSR to release SCL line.
5. 1-Byte of data is being received.
6. In this step, $I^2C$ generates TEND interrupt and holds the SCL line LOW regardless of the reception of ACK signal from master. Slave can select one of the following cases.

   1) No ACK signal is detected (ACKEN=0) and $I^2C$ waits STOP or repeated START condition.
   2) ACK signal is detected (ACKEN=1) and $I^2C$ can continue to receive data from master.

   After doing one of the actions above, write arbitrary value to I2CSR to release SCL line. In case of 1) move to step 7 to terminate communication. In case of 2) move to step 5. In either case, a repeated START condition can be detected. For that case, move step 4.
7. This is the final step for slave receiver function of $I^2C$, handling STOP interrupt. The STOP bit indicates that data transfer between master and slave is over. To clear I2CSR, write arbitrary value to I2CSR. After this, $I^2C$ enters idle state.

The process can be depicted as following figure when $I^2C$ operates in slave receiver mode.

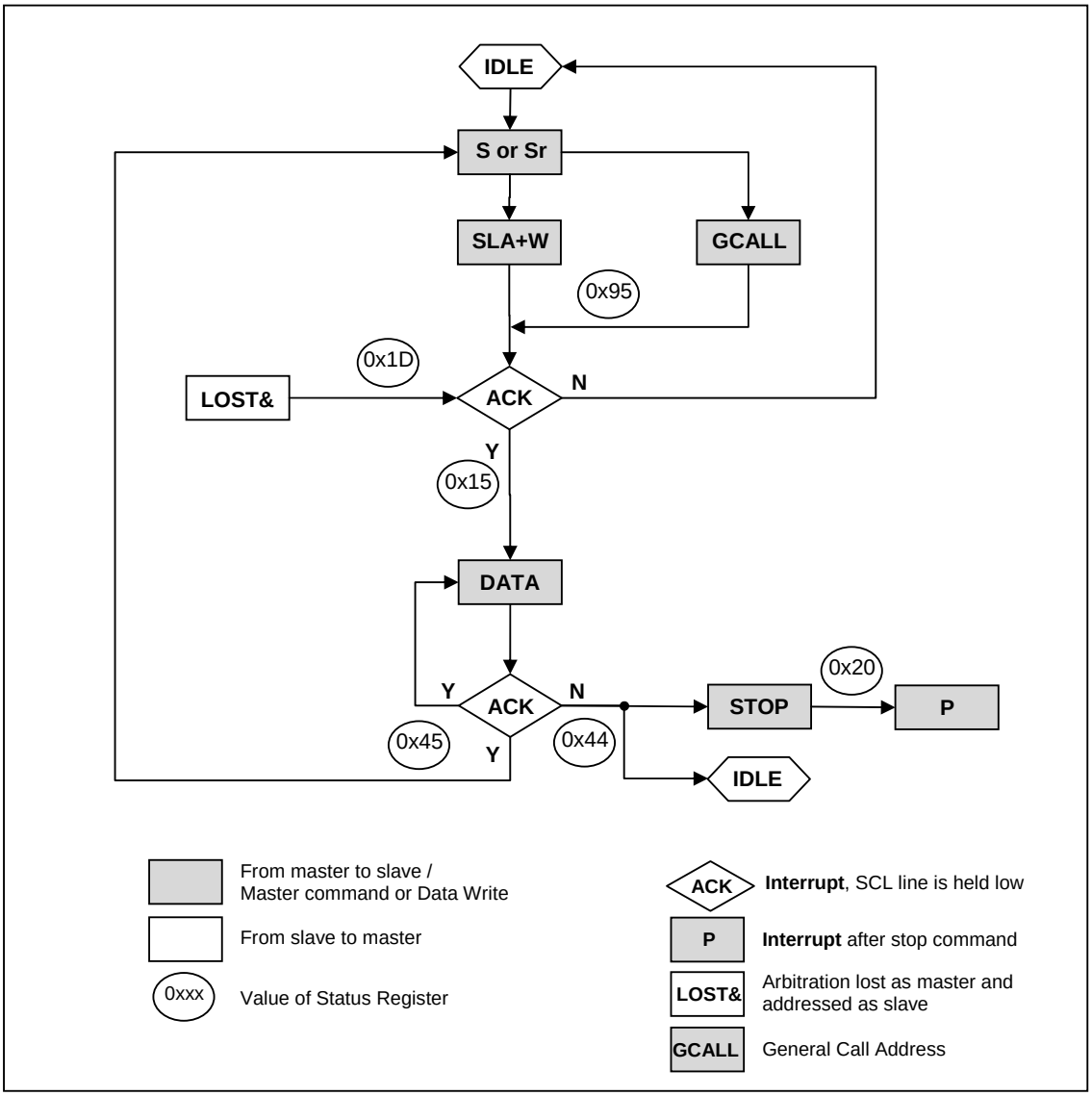



Figure 11-61 Formats and States in the Slave Receiver Mode

### 11.9.9 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| I2CMR | DAH | R/W | 00H | $I^2C$ Mode Control Register |
| I2CSR | DBH | R | 00H | $I^2C$ Status Register |
| I2CSCLLR | DCH | R/W | 3FH | SCL Low Period Register |
| I2CSCLHR | DDH | R/W | 3FH | SCL High Period Register |
| I2CSDAHR | DEH | R/W | 01H | SDA Hold Time Register |
| I2CDR | DFH | R/W | FFH | $I^2C$ Data Register |
| I2CSAR | D7H | R/W | 00H | $I^2C$ Slave Address Register |
| I2CSAR1 | D6H | R/W | 00H | $I^2C$ Slave Address Register 1 |

### 11.9.10 I²C Register description

I²C Registers are composed of I²C Mode Control Register (I2CMR), I²C Status Register (I2CSR), SCL Low Period Register (I2CSCLLR), SCL High Period Register (I2CSCLHR), SDA Hold Time Register (I2CSDAHR), I²C Data Register (I2CDR), and I²C Slave Address Register (I2CSAR).

### 11.9.11 Register description for I²C

**I2CMR (I²C Mode Control Register) : DAH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IIF | IICEN | RESET | INTEN | ACKEN | MASTER | STOP | START |
| R/W | R/W | R/W | R/W | R/W | R | R/W | R/W |

Initial value : 00H

**IIF** This is interrupt flag bit.
- 0 No interrupt is generated or interrupt is cleared
- 1 An interrupt is generated

**IICEN** Enable I²C Function Block (by providing clock)
- 0 I²C is inactive
- 1 I²C is active

**RESET** Initialize internal registers of I²C.
- 0 No operation
- 1 Initialize I²C, auto cleared

**INTEN** Enable interrupt generation of I²C.
- 0 Disable interrupt, operates in polling mode
- 1 Enable interrupt

**ACKEN** Controls ACK signal generation at ninth SCL period.
Note) ACK signal is output (SDA=0) for the following 3 cases.
When received address packet equals to SLA bits in I2CSAR
When received address packet equals to value 0x00 with GCALL enabled
When I²C operates as a receiver (master or slave)
- 0 No ACK signal is generated (SDA=1)
- 1 ACK signal is generated (SDA=0)

**MASTER** Represent operating mode of I²C
- 0 I²C is in slave mode
- 1 I²C is in master mode

**STOP** When I²C is master, generates STOP condition.
- 0 No operation
- 1 STOP condition is to be generated

**START** When I²C is master, generates START condition.
- 0 No operation
- 1 START or repeated START condition is to be generated

### I2CSR ($I^2C$ Status Register) : DBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| GCALL | TEND | STOP | SSEL | MLOST | BUSY | TMODE | RXACK |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**GCALL** This bit has different meaning depending on whether $I^2C$ is master or slave. Note 1)
When $I^2C$ is a master, this bit represents whether it received AACK (Address ACK) from slave.
When $I^2C$ is a slave, this bit is used to indicate general call.
0 No AACK is received (Master mode)
1 AACK is received (Master mode)
0 Received address is not general call address (Slave mode)
1 General call address is detected (Slave mode)

**TEND** This bit is set when 1-Byte of data is transferred completely. Note 1)
0 1 byte of data is not completely transferred
1 1 byte of data is completely transferred

**STOP** This bit is set when STOP condition is detected. Note 1)
0 No STOP condition is detected
1 STOP condition is detected

**SSEL** This bit is set when $I^2C$ is addressed by other master. Note 1)
0 $I^2C$ is not selected as slave
1 $I^2C$ is addressed by other master and acts as a slave

**MLOST** This bit represents the result of bus arbitration in master mode. Note 1)
0 $I^2C$ maintains bus mastership
1 $I^2C$ has lost bus mastership during arbitration process

**BUSY** This bit reflects bus status.
0 $I^2C$ bus is idle, so any master can issue a START condition
1 $I^2C$ bus is busy

**TMODE** This bit is used to indicate whether $I^2C$ is transmitter or receiver.
0 $I^2C$ is a receiver
1 $I^2C$ is a transmitter

**RXACK** This bit shows the state of ACK signal.
0 No ACK is received
1 ACK is generated at ninth SCL period

Note 1) These bits can be source of interrupt.

When an $I^2C$ interrupt occurs except for STOP interrupt, the SCL line is hold LOW. To release SCL, write arbitrary value to I2CSR. When I2CSR is written, the TEND, STOP, SSEL, LOST, RXACK bits are cleared.

### I2CSCLLR (SCL Low Period Register) : DCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SCLL7 | SCLL6 | SCLL5 | SCLL4 | SCLL3 | SCLL2 | SCLL1 | SCLL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 3FH

**SCLL[7:0]** This register defines the LOW period of SCL when $I^2C$ operates in master mode. The base clock is SCLK, the system clock, and the period is calculated by the formula : $t_{SCLK} \times (4 \times SCLL + 1)$ where $t_{SCLK}$ is the period of SCLK.

### I2CSCLHR (SCL High Period Register) : DDH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SCLH7 | SCLH6 | SCLH5 | SCLH4 | SCLH3 | SCLH2 | SCLH1 | SCLH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 3FH

**SCLH[7:0]** This register defines the HIGH period of SCL when $I^2C$ operates in master mode. The base clock is SCLK, the system clock, and the period is calculated by the formula : $t_{SCLK} \times (4 \times SCLH + 3)$ where $t_{SCLK}$ is the period of SCLK.

So, the operating frequency of $I^2C$ in master mode (fI2C) is calculated by the following equation.

$$fI2C = \frac{1}{tSCLK \times (4\ (SCLL + SCLH) + 4)}$$

### I2CSDAHR (SDA Hold Time Register) : DEH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SDAH7 | SDAH6 | SDAH5 | SDAH4 | SDAH3 | SDAH2 | SDAH1 | SDAH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 01H

**SDAH[7:0]** This register is used to control SDA output timing from the falling edge of SCL. Note that SDA is changed after $t_{SCLK} \times SDAH$. In master mode, load half the value of SCLL to this register to make SDA change in the middle of SCL. In slave mode, configure this register regarding the frequency of SCL from master. The SDA is changed after $t_{SCLK} \times (SDAH + 1)$. So, to insure normal operation in slave mode, the value $t_{SCLK} \times (SDAH + 1)$ must be smaller than the period of SCL.

### I2CDR ($I^2C$ Data Register) : DFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ICD7 | ICD6 | ICD5 | ICD4 | ICD3 | ICD2 | ICD1 | ICD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : FFH

**ICD[7:0]** When $I^2C$ is configured as a transmitter, load this register with data to be transmitted. When $I^2C$ is a receiver, the received data is stored into this register.

### I2CSAR ($I^2C$ Slave Address Register) : D7H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SLA7 | SLA6 | SLA5 | SLA4 | SLA3 | SLA2 | SLA1 | GCALLEN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**SLA[7:1]** These bits configure the slave address of this $I^2C$ module when $I^2C$ operates in slave mode.

**GCALLEN** This bit decides whether $I^2C$ allows general call address or not when $I^2C$ operates in slave mode.

0 Ignore general call address

1 Allow general call address

### I2CSAR1 ($I^2C$ Slave Address Register 1) : D6H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SLA7 | SLA6 | SLA5 | SLA4 | SLA3 | SLA2 | SLA1 | GCALLEN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**SLA[7:1]** These bits configure the slave address of this $I^2C$ module when $I^2C$ operates in slave mode.

**GCALLEN** This bit decides whether $I^2C$ allows general call address or not when $I^2C$ operates in slave mode.

0 Ignore general call address

1 Allow general call address

## 11.10 12-Bit A/D Converter

### 11.10.1 Overview

The analog-to-digital converter (A/D) allows conversion of an analog input signal to a corresponding 12-bit digital value. The A/D module has tenth analog inputs. The output of the multiplex is the input into the converter, which generates the result via successive approximation. The A/D module has four registers which are the control register ADCM (A/D Converter Mode Register), ADCM2 (A/D Converter Mode Register 2) and A/D result register ADCHR (A/D Converter Result High Register) and ADCLR (A/D Converter Result Low Register). It is selected for the corresponding channel to be converted by setting ADSEL[3:0]. To executing A/D conversion, ADST bit sets to '1'. The register ADCHR and ADCLR contains the results of the A/D conversion. When the conversion is completed, the result is loaded into the ADCHR and ADCLR, the A/D conversion status bit AFLAG is set to '1', and the A/D interrupt is set. For processing A/D conversion, AFLAG bit is read as '0'. If using STBY (power down) bit, the ADC is disabled. Also internal timer, external generating event, comparator, the trigger of timer1pwm and etc. can start ADC regardless of interrupt occurrence.

ADC Conversion Time = ADCLK * 60 cycles

After STBY bit is reset (ADC power enable) and it is restarted, during some cycle, ADC conversion value may have an inaccurate value.

### 11.10.2 Block Diagram

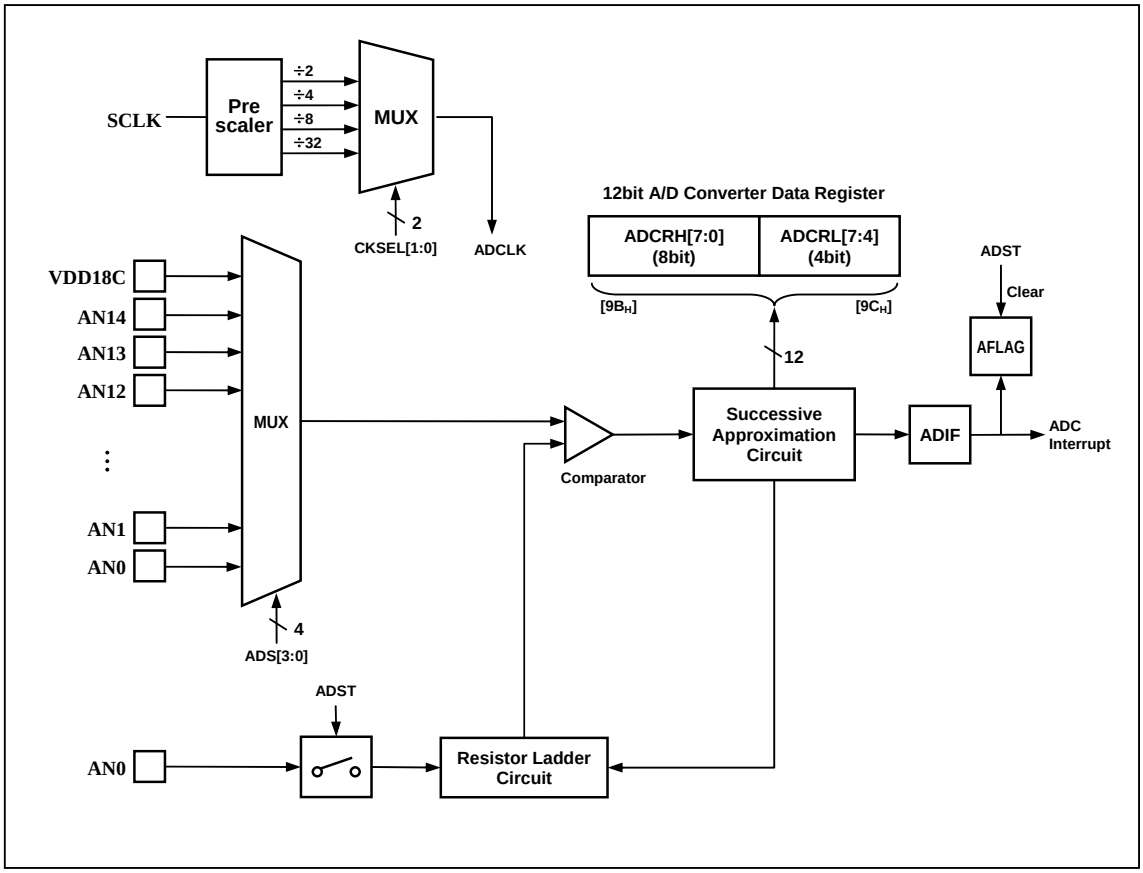
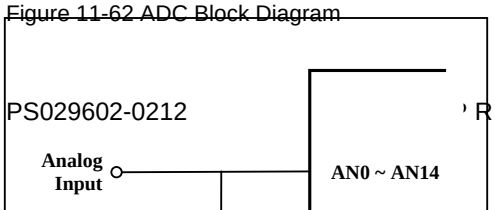



Figure 11-62 ADC Block Diagram



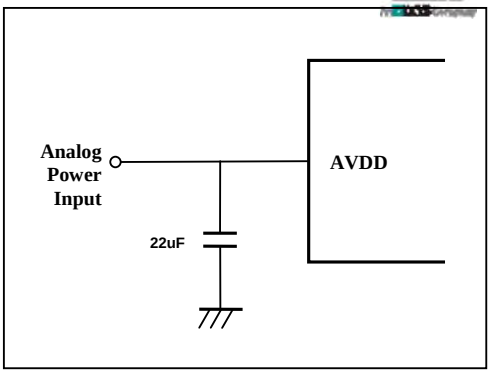



Figure 11-64 A/D Power(AVDD) Pin Connecting Capacitor

### 11.10.3 ADC Operation

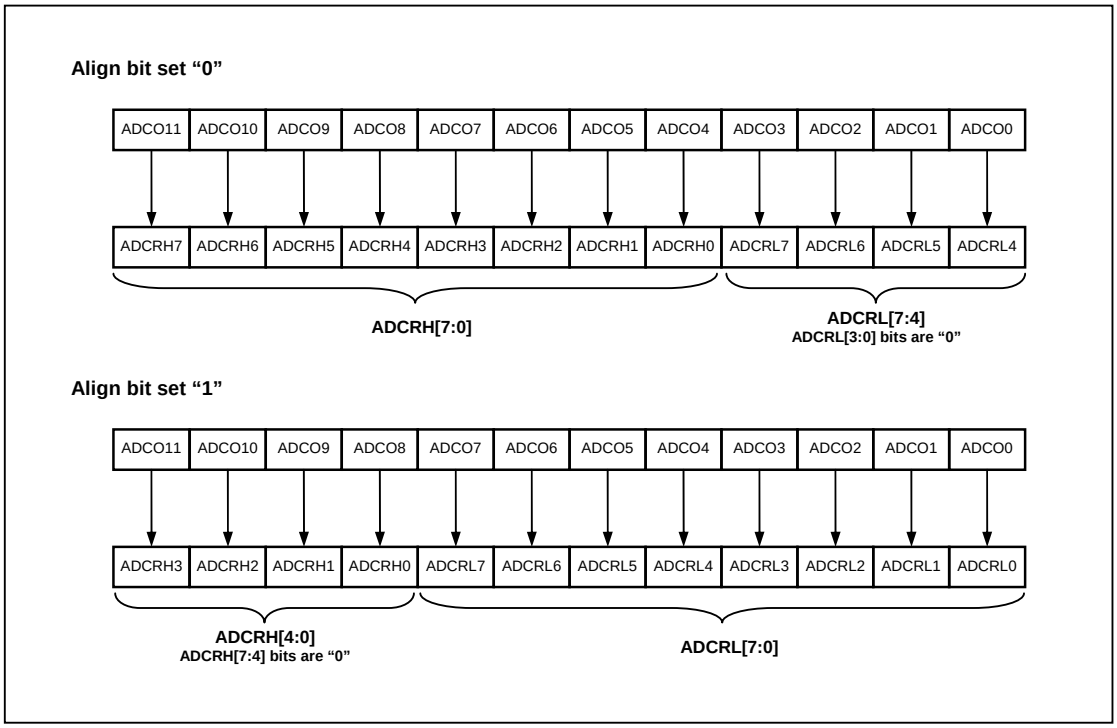



Figure 11-65 ADC Operation for Align bit

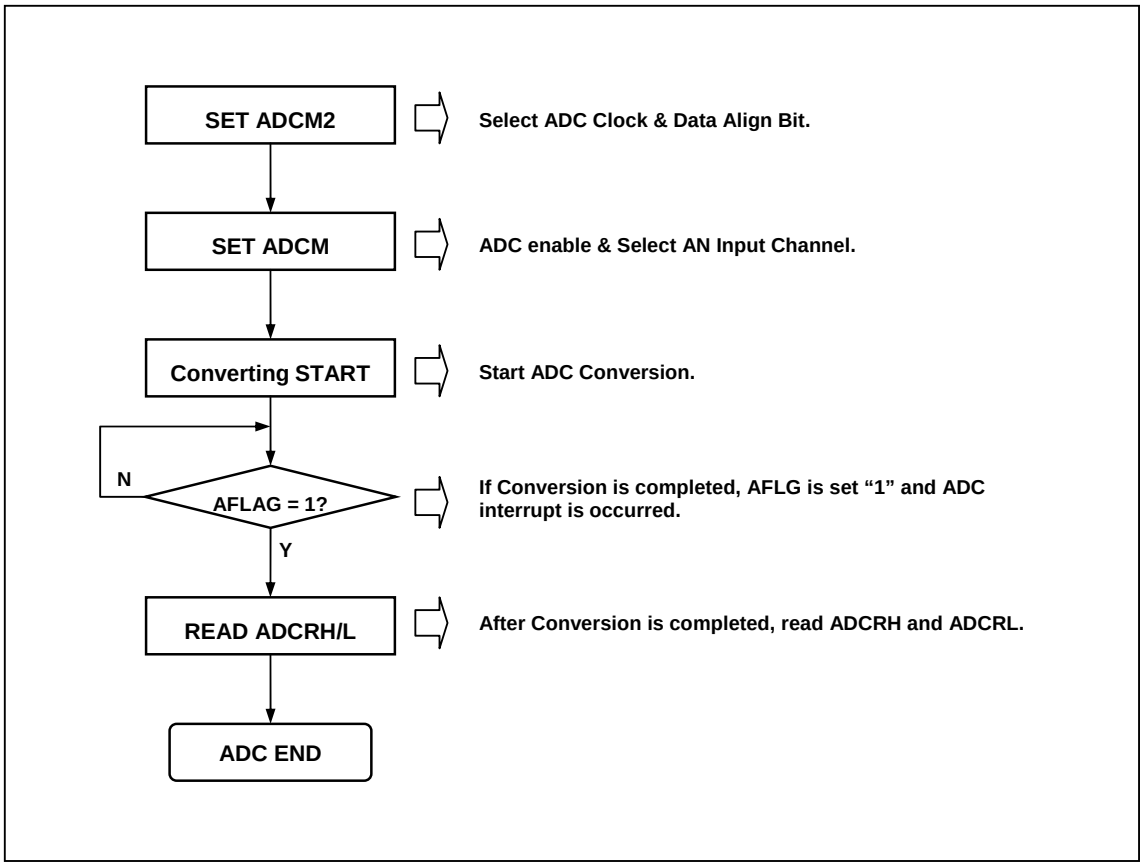



Figure 11-66 Converter Operation Flow

### 11.10.4 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| ADCM | 9AH | R/W | 8FH | A/D Converter Mode Register |
| ADCRH | 9BH | R | - | A/D Converter Result High Register |
| ADCRL | 9CH | R | - | A/D Converter Result Low Register |
| ADCM2 | 9BH | R/W | 8FH | A/D Converter Mode 2 Register |

### 11.10.5 ADC Register description

The ADC Register consists of A/D Converter Mode Register (ADCM), A/D Converter Result High Register (ADCRH), A/D Converter Result Low Register (ADCRL), A/D Converter Mode 2 Register (ADCM2).

Note) when STBY bit is set to '1', ADCM2 can be read. If ADC enables, it is possible only to write ADCM2.When reading, ADCRH is read.

### 11.10.6 Register description for ADC

**ADCM (A/D Converter Mode Register) : 9AH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| STBY | ADST | REFSEL | AFLAG | ADSEL3 | ADSEL2 | ADSEL1 | ADSEL0 |
| R/W | R/W | R/W | R | R/W | R/W | R/W | R/W |

Initial value : 8FH

**STBY** Control operation of A/D standby (power down)

0 ADC module enable

1 ADC module disable (power down)

**ADST** Control A/D Conversion stop/start.

0 ADC Conversion Stop

1 ADC Conversion Start

**REFSEL** A/D Converter reference selection

0 Internal Reference (VDD)

1 External Reference(AVREF, AN0 disable)

**AFLAG** A/D Converter operation state

0 During A/D Conversion

1 A/D Conversion finished

**ADSEL[3:0]** A/D Converter input selection

| ADSEL3 | ADSEL2 | ADSEL1 | ADSEL0 | Description |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Channel0(AN0) |
| 0 | 0 | 0 | 1 | Channel1(AN1) |
| 0 | 0 | 1 | 0 | Channel2(AN2) |
| 0 | 0 | 1 | 1 | Channel3(AN3) |
| 0 | 1 | 0 | 0 | Channel4(AN4) |
| 0 | 1 | 0 | 1 | Channel5(AN5) |
| 0 | 1 | 1 | 0 | Channel6(AN6) |
| 0 | 1 | 1 | 1 | Channel7(AN7) |
| 1 | 0 | 0 | 0 | Channel8(AN8) |
| 1 | 0 | 0 | 1 | Channel9(AN9) |
| 1 | 0 | 1 | 0 | Channel10(AN10) |
| 1 | 0 | 1 | 1 | Channel11(AN11) |
| 1 | 1 | 0 | 0 | Channel12(AN12) |
| 1 | 1 | 0 | 1 | Channel13(AN13) |
| 1 | 1 | 1 | 0 | Channel14(AN14) |
| 1 | 1 | 1 | 1 | Channel15(VDD18) |

### ADCRH (A/D Converter Result High Register) : 9BH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ADDM11 | ADDM10 | ADDM9 | ADDM8 | ADDM7<br>ADDL11 | ADDM6<br>ADDL10 | ADDM5<br>ADDL9 | ADDM4<br>ADDL8 |
| R | R | R | R | R | R | R | R |

Initial value : xxH

**ADDM[11:4]** MSB align, A/D Converter High result (8-bit)
**ADDL[11:8]** LSB align, A/D Converter High result (4-bit)

### ADCRL (A/D Converter Result Low Register) : 9CH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ADDM3<br>ADDL7 | ADDM2<br>ADDL6 | ADDM1<br>ADDL5 | ADDM0<br>ADDL4 | ADDL3 | ADDL2 | ADDL1 | ADDL0 |
| R | R | R | R | R | R | R | R |

Initial value : xxH

**ADDM[3:0]** MSB align, A/D Converter Low result (4-bit)
**ADDL[7:0]** LSB align, A/D Converter Low result (8-bit)

### ADCM2 (A/D Converter Mode Register) : 9BH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EXTRG | TSEL2 | TSEL1 | TSEL0 | - | ALIGN | CKSEL1 | CKSEL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 01H

**EXTRG** A/D external Trigger

0 External Trigger disable

1 External Trigger enable

**TSEL[2:0]** A/D Trigger Source selection

| TSEL2 | TSEL1 | TSEL0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | Ext. Interrupt 0 |
| 0 | 0 | 1 | Analog Comparator Low to High Transition |
| 0 | 1 | 0 | Analog Comparator High to Low Transition |
| 0 | 1 | 1 | Timer1PWM overflow event |
| 1 | 0 | 0 | Timer1PWM A-ch event compare match |
| 1 | 0 | 1 | Timer1PWM B-ch event compare match |
| 1 | 1 | 0 | Timer1PWM C-ch event compare match |
| 1 | 1 | 1 | Timer3(PWM) interrupt |

**ALIGN** A/D Converter data align selection.

0 MSB align (ADCRH[7:0], ADCRL[7:4])

1 LSB align (ADCRH[3:0], ADCRL[7:0])

**CKSEL[1:0]** A/D Converter Clock selection

| CKSEL1 | CKSEL0 | ADC Clock | ADC VDD |
|---|---|---|---|
| 0 | 0 | fx/2 | Test Only |
| 0 | 1 | fx/4 | 3V~5V |
| 1 | 0 | fx/8 | 2.7V~3V |
| 1 | 1 | fx/32 | 2.4V~2.7V |

Note) 1. fx : system clock

2. ADC clock have to be used 3MHz under

## 11.11 Analog Comparator

### 11.11.1 Overview

The Analog Comparator compares the input values on the positive pin AC+ and the negative pin AC-. When the voltage on the positive pin AC+ is higher than the voltage on the negative pin AC-, the Analog Comparator output, ACOUT, is set.

### 11.11.2 Block Diagram

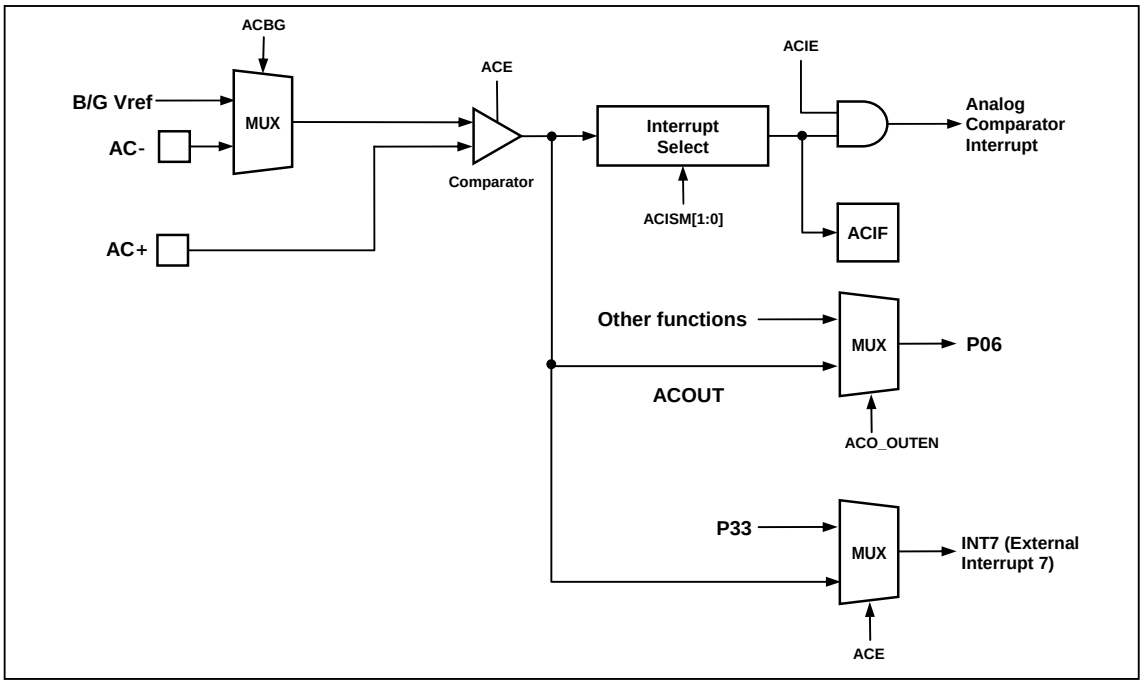



Figure 11-67 Analog Comparator Block Diagram

### 11.11.3 IN/OUT signal description

ACE : This enables Analog Comparator. When ACE is ‘0’, the output of Comparator goes LOW.

BGR : Band Gap Reference Voltage

ACBG : This selects (-) input source between BGR and AC-. When ACBG is ‘1’, the (-) input to AC is BGR.

AC- : This can be (-) input to the AC, and comes directly from external analog pad.

AC+ : This can be (+) input to the AC, and comes directly from external analog pad.

AMUXENB : This selects (+) input source between multiplexed output of ADC and AN5. AMUXENB is the inverted signal of AMUXEN bit in ADCM2 register. When AMUXENB is ‘0’, the (+) input to AC comes from ADC module which is selected by ADSEL[3:0], the channel selection bits in ADCM register.

ACOUT : This is the output of Comparator.

ACO_OUTEN : Analog Comparator output port Enable.

### 11.11.4 Register Map

Table 11-19 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| ACCSR | E9 | R/W | 00H | Analog Comparator Control & Status Register |

### 11.11.5 Analog Comparator Register description

Analog Comparator Register has one control register, Analog Comparator Control & Status Register (ACCSR). Note that AMUXENB is the inverted signal of AMUXEN bit which comes from ADC's ADCM2 register

### 11.11.6 Register description for Analog Comparator

**ACCSR (Analog Comparator Control & Status Register) : F9H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ACE | ACBG | ACO | ACIF | ACIE | ACO_OUTEN | ACISM1 | ACISM0 |
| R/W | R/W | R | R | R/W | R/W | R/W | R/W |

Initial value : 00H

**ACE** Enable Analog Comparator (AC).

0 Disable AC (power down)

1 Enable AC

**ACBG** Select (-) input source of AC, Band Gap Reference Voltage or AN4.

0 (-) input is from AN4

1 (-) input is from Band Gap Reference Voltage

**ACO** This bit represents the value of ACOUT (Output of Analog Comparator). ACO bit is sampled by SCLK, system clock, twice. When ACE is '0', this bit is also cleared.

0 Comparator output is LOW

1 Comparator output is HIGH

**ACIF** This bit is set when an Analog Comparator Interrupt is generated according to the ACISM[1:0] bits. This bit is cleared when Analog Comparator Interrupt is executed or '0' is written to this bit field.

0 No interrupt generated or cleared

1 Interrupt generated

**ACIE** Enable Analog Comparator Interrupt.

0 Disable Interrupt, Polling mode operation

1 Enable Interrupt

**ACO_OUTEN** Analog Comparator output port Enable

0 Disable

1 Enable

**ACISM[1:0]** Select Interrupt Mode of Analog Comparator.

| ACISM1 | ACISM0 | Description |
|---|---|---|
| 0 | 0 | Reserved |
| 0 | 1 | Interrupt on falling edge of ACOUT |
| 1 | 0 | Interrupt on rising edge of ACOUT |
| 1 | 1 | Interrupt on both edge of ACOUT |

# 12. Power Down Operation

## 12.1 Overview

The Z51F0811 has three power-down modes to minimize the power consumption of the device. In power down mode, power consumption is reduced considerably. The device provides three kinds of power saving functions, IDLE, STOP1 and STOP2 mode. In three modes, program is stopped.

## 12.2 Peripheral Operation in IDLE/STOP Mode

Table 12-1 Peripheral Operation during Power Down Mode.

| Peripheral | IDLE Mode | STOP1 Mode | STOP2 Mode |
|---|---|---|---|
| CPU | ALL CPU Operation are Disable | ALL CPU Operation are Disable | ALL CPU Operation are Disable |
| RAM | Retain | Retain | Retain |
| Basic Interval Timer | Operates Continuously | Operates Continuously | Stop |
| Watch Dog Timer | Operates Continuously | Operates Continuously | Stop |
| Watch Timer | Operates Continuously | Stop (Only operate in sub clock mode) | Stop (Only operate in sub clock mode) |
| TimerP0~4 | Operates Continuously | Halted (Only when the Event Counter Mode is Enable, Timer operates Normally) | Halted (Only when the Event Counter Mode is Enable, Timer operates Normally) |
| ADC | Operates Continuously | Stop | Stop |
| BUZ | Operates Continuously | Stop | Stop |
| SPI/SCI | Operates Continuously | Only operate with external clock | Only operate with external clock |
| $I^2C$ | Operates Continuously | Stop | Stop |
| Internal OSC (8MHz) | Oscillation | Stop | Stop |
| Main OSC (1~12MHz) | Oscillation | Stop | Stop |
| Sub OSC (32.768kHz) | Oscillation | Oscillation | Oscillation |
| Internal RCOSC (125kHz) | Oscillation | Oscillation | Stop |
| I/O Port | Retain | Retain | Retain |
| Control Register | Retain | Retain | Retain |
| Address Data Bus | Retain | Retain | Retain |
| Release Method | By RESET, all Interrupts | By RESET, Timer Interrupt (EC0, EC2), SIO (External clock), External Interrupt, UART by ACK PCI, $I^2C$ (slave mode), WT (sub clock),WDT, BIT | By RESET, Timer Interrupt (EC0, EC2), SIO (External clock), External Interrupt, UART by ACK PCI, $I^2C$ (slave mode), WT (sub clock) |

## 12.3 IDLE mode

The power control register is set to '01h' to enter the IDLE Mode. In this mode, the internal oscillation circuits remain active. Oscillation continues and peripherals are operated normally but CPU stops. It is released by reset or interrupt. To be released by interrupt, interrupt should be enabled before IDLE mode. If using reset, because the device becomes initialized state, the registers have reset value.

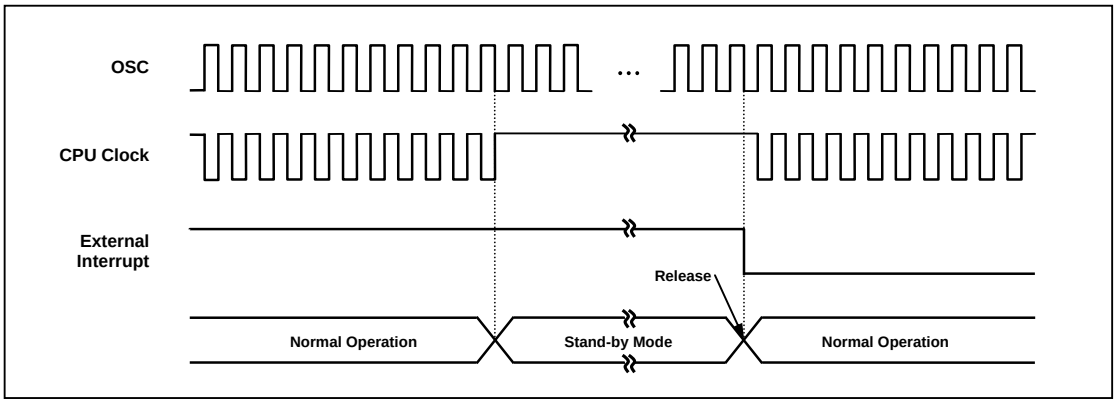



Figure 12-1 IDLE Mode Release Timing by External Interrupt

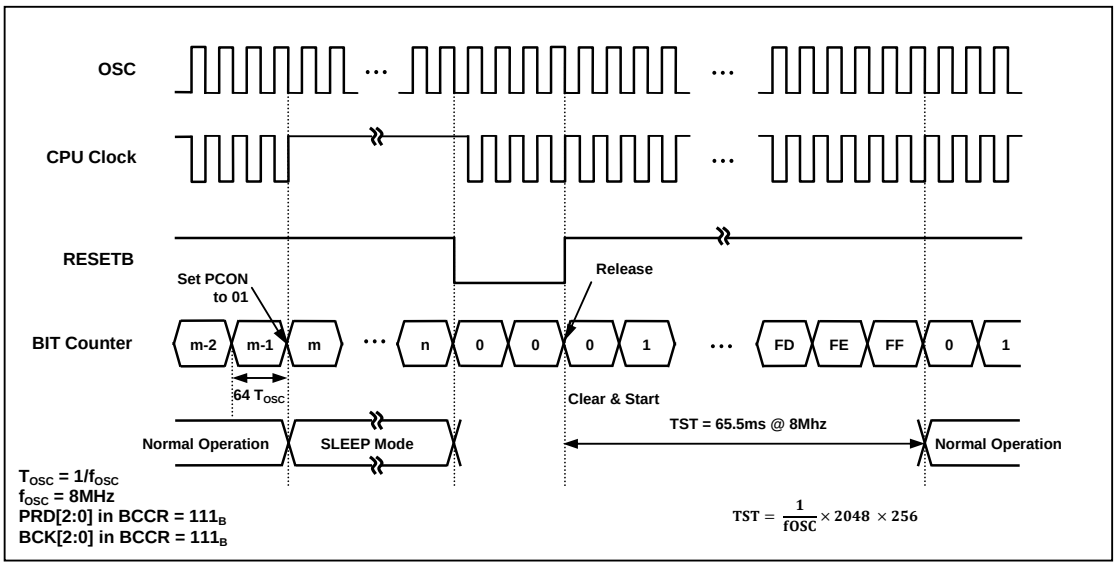



Figure 12-2 IDLE Mode Release Timing by /RESET

(Ex) MOV PCON, #0000_0001b ; setting of IDLE mode : set the bit of STOP and IDLE Control register (PCON)

## 12.4 STOP mode

The power control register is set to '03h' to enter the STOP Mode. In the stop mode, the main oscillator, system clock and peripheral clock is stopped, but watch timer continue to operate. With the clock frozen, all functions are stopped, but the on-chip RAM and control registers are held.

The source for exit from STOP mode is hardware reset and interrupts. The reset re-defines all the control registers.

When exit from STOP mode, enough oscillation stabilization time is required to normal operation. Figure 12-3 shows the timing diagram. When released from STOP mode, the Basic interval timer is activated on wake-up. Therefore, before STOP instruction, user must be set its relevant prescaler divide ratio to have long enough time (more than 20msec). this guarantees that oscillator has started and stabilized.

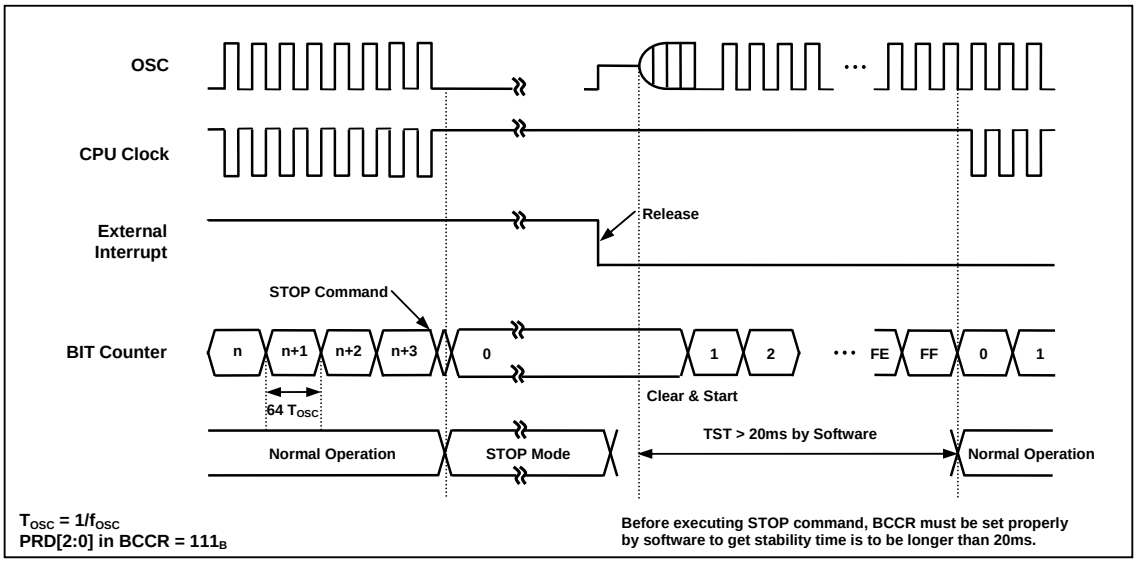



Figure 12-3 STOP Mode Release Timing by External Interrupt

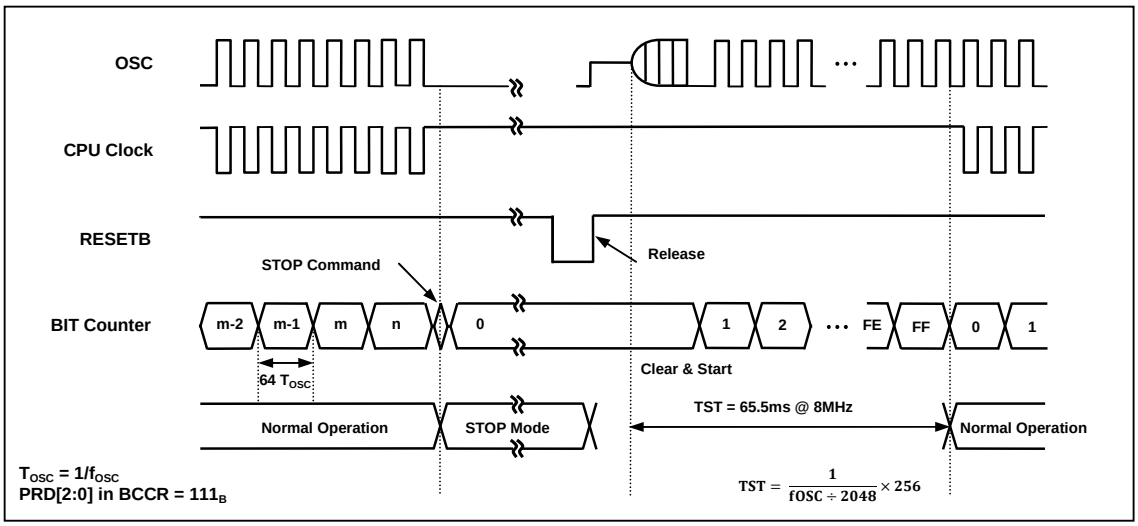



Figure 12-4 STOP Mode Release Timing by /RESET

## 12.5 Release Operation of STOP1, 2 Mode

After STOP1, 2 mode is released, the operation begins according to content of related interrupt register just before STOP1, 2 mode start (Figure 12-5). Interrupt Enable Flag of All (EA) of IE should be set to `1`. Released by only interrupt which each interrupt enable flag = `1`, and jump to the relevant interrupt service routine.

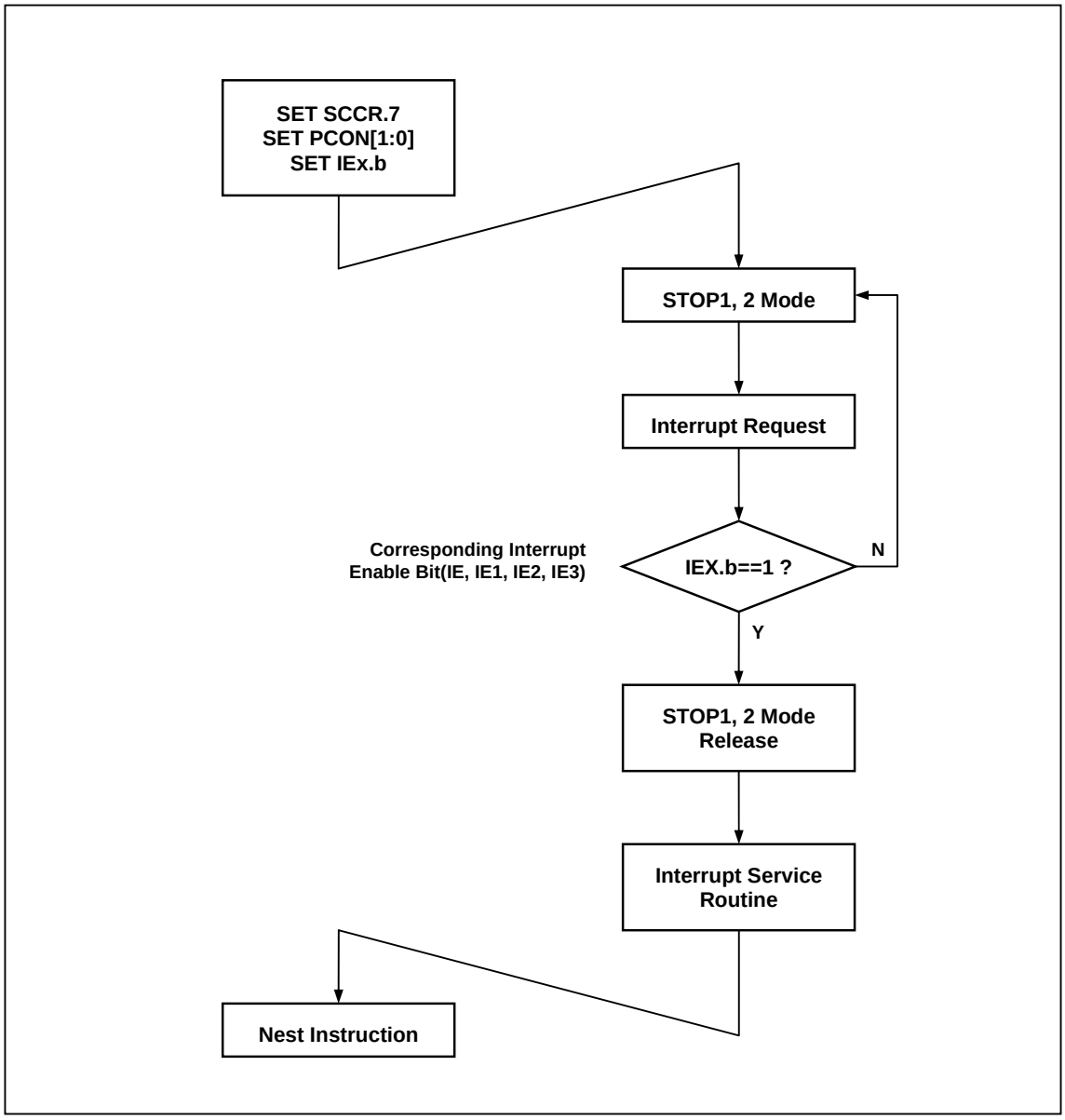



Figure 12-5 STOP1, 2 Mode Release Flow

### 12.5.1 Register Map

Table 12-2 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| PCON | 87H | R/W | 00H | Power Control Register |

### 12.5.2 Power Down Operation Register description

The Power Down Operation Register consists of the Power Control Register (PCON).

### 12.5.3 Register description for Power Down Operation

**PCON (Power Control Register) : 87H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

IDLE Mode

01H IDLE mode enable

STOP1, 2 Mode

03H STOP1, 2 mode enable

Note)

1. To enter IDLE mode, PCON must be set to ‘01H’.
2. To STOP1,2 mode, PCON must be set to ‘03H’.
   (In STOP1,2 mode, PCON register is cleared automatically by interrupt or reset)
3. When PCON is set to ‘03H’, if SCCR[7] is set to ‘1’, it enters the STOP1 mode. if SCCR[7] is cleared to ‘0’, it enters the STOP2 mode
4. The different thing in STOP 1,2 is only clock operation of internal 125kHz-OSC during STOP mode operating.

# 13. RESET

## 13.1 Overview

The Z51F0811 has reset by external RESETB pin. The following is the hardware setting value.

Table 13-1 Reset state

| On Chip Hardware | Initial Value |
|---|---|
| Program Counter (PC) | 0000h |
| Accumulator | 00h |
| Stack Pointer (SP) | 07h |
| Peripheral Clock | On |
| Control Register | Peripheral Registers refer |
| Brown-Out Detector | Enable |

## 13.2 Reset source

The Z51F0811 has five types of reset generation procedures. The following is the reset sources.

- External RESETB
- Power ON RESET (POR)
- WDT Overflow Reset (In the case of WDTEN = `1`)
- BOD Reset (In the case of BODEN = `1 `)
- OCD Reset

## 13.3 Block Diagram

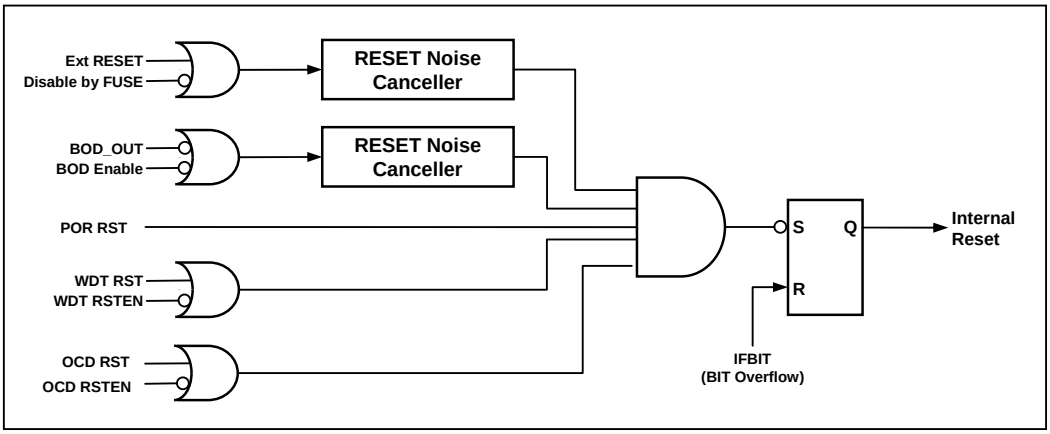



Figure 13-1 RESET Block Diagram

## 13.4 RESET Noise Canceller

The Figure 13-2 is the Noise canceller diagram for Noise cancel of RESET. It has the Noise cancel value of about 7us (@$V_{DD}$=5V) to the low input of System Reset.

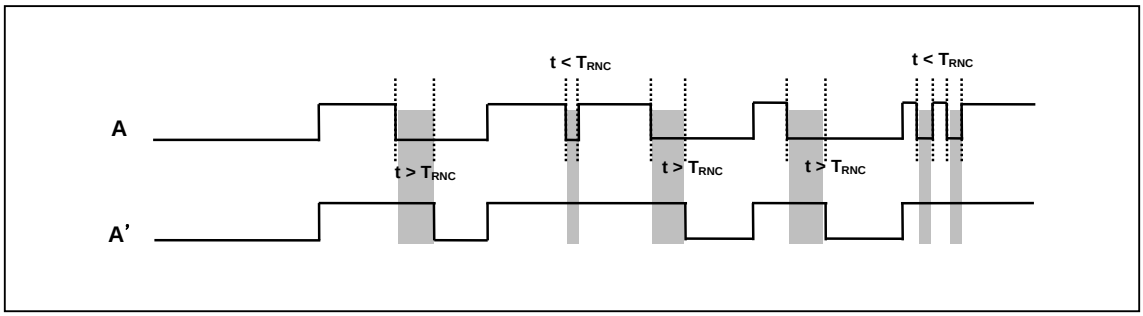



Figure 13-2 Reset noise canceller time diagram

## 13.5 Power ON RESET

When rising device power, the POR (Power ON Reset) have a function to reset the device. If using POR, it executes the device RESET function instead of the RESET IC or the RESET circuits. And External RESET PIN is able to use as Normal I/O pin.

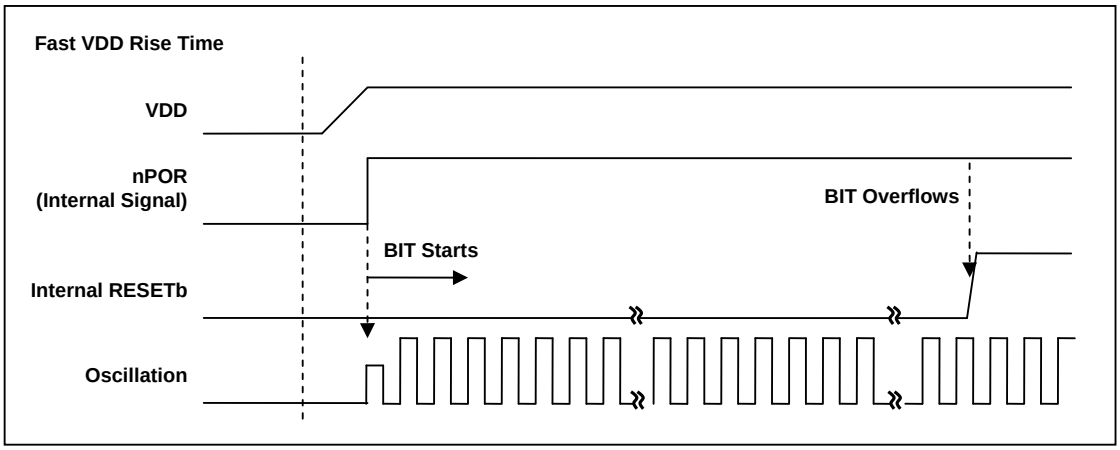



Figure 13-3 Fast VDD rising time

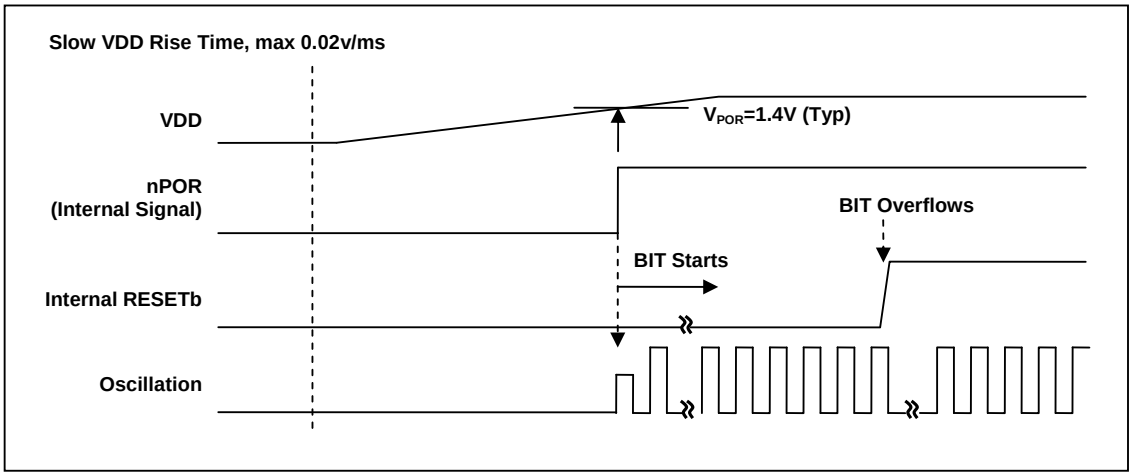



Figure 13-4 Internal RESET Release Timing On Power-Up

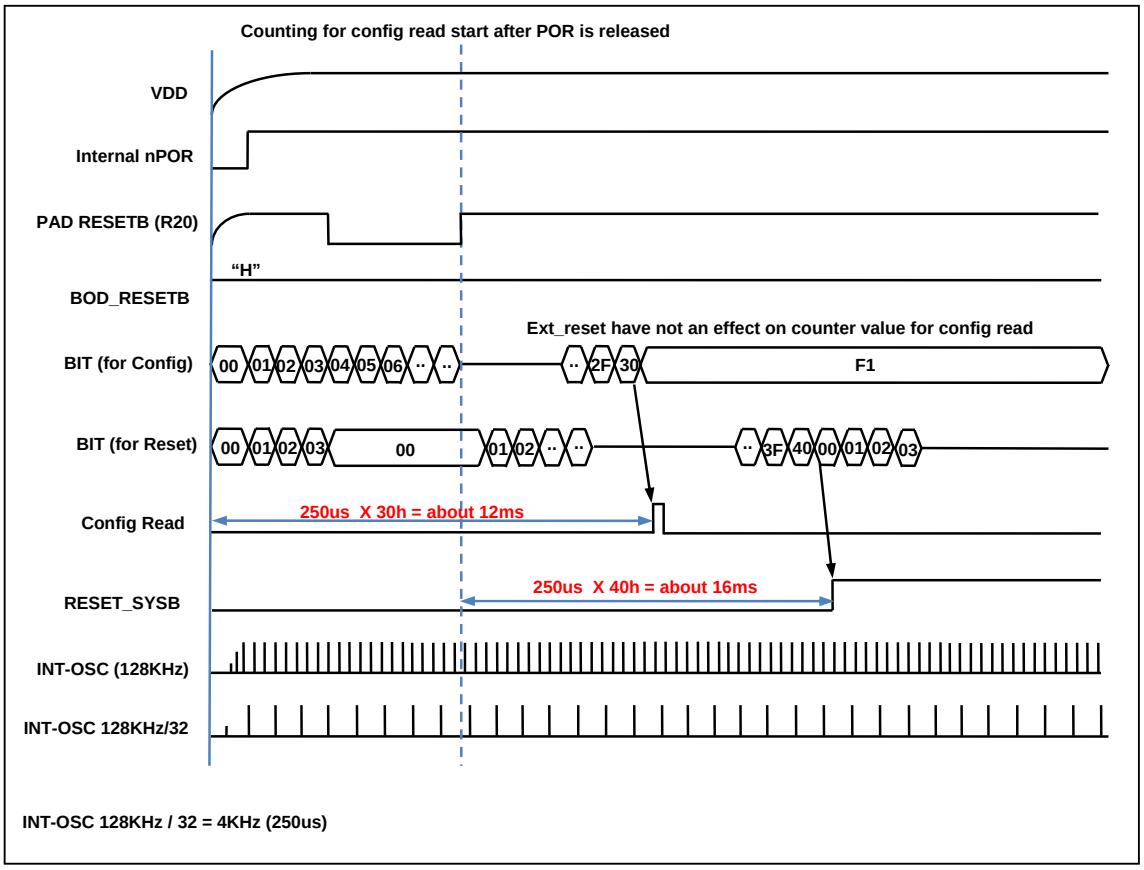



Figure 13-5 Configuration timing when Power-on

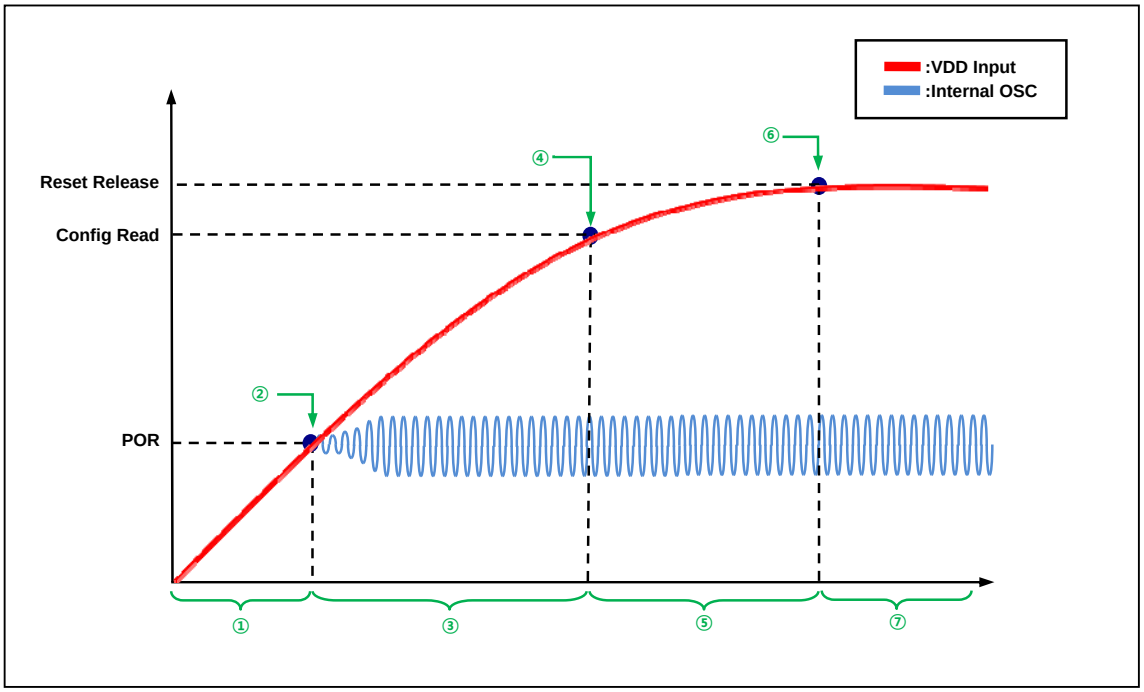



Figure 13-6 Boot Process Waveform

Table 13-2 Boot Process Description

| Process | Description | Remarks |
|---|---|---|
| ① | -No Operation | |
| ② | -1st POR level Detection<br>-Internal OSC (125KHz) ON | -about 1.4V ~ 1.5V |
| ③ | - (INT-OSC125KHz/32)×30h Delay section (=12ms)<br>-VDD input voltage must rise over than flash operating voltage for Config read | -Slew Rate >= 0.025V/ms |
| ④ | - Config read point | -about 1.5V ~ 1.6V<br>-Config Value is determined by Writing Option |
| ⑤ | - Rising section to Reset Release Level | -16ms point after POR or Ext_reset release |
| ⑥ | - Reset Release section (BIT overflow)<br>i) after16ms, after External Reset Release (External reset)<br>ii) 16ms point after POR (POR only) | - BIT is used for Peripheral stability |
| ⑦ | -Normal operation | |

## 13.6 External RESETB Input

The External RESETB is the input to a Schmitt trigger. A reset in accomplished by holding the reset pin low for at least 7us over, within the operating voltage range and oscillation stable, it is applied, and the internal state is initialized. After reset state becomes ‘1’, it needs the stabilization time with 16ms and after the stable state, the internal RESET becomes ‘1’. The Reset process step needs 5 oscillator clocks. And the program execution starts at the vector address stored at address 0000H.

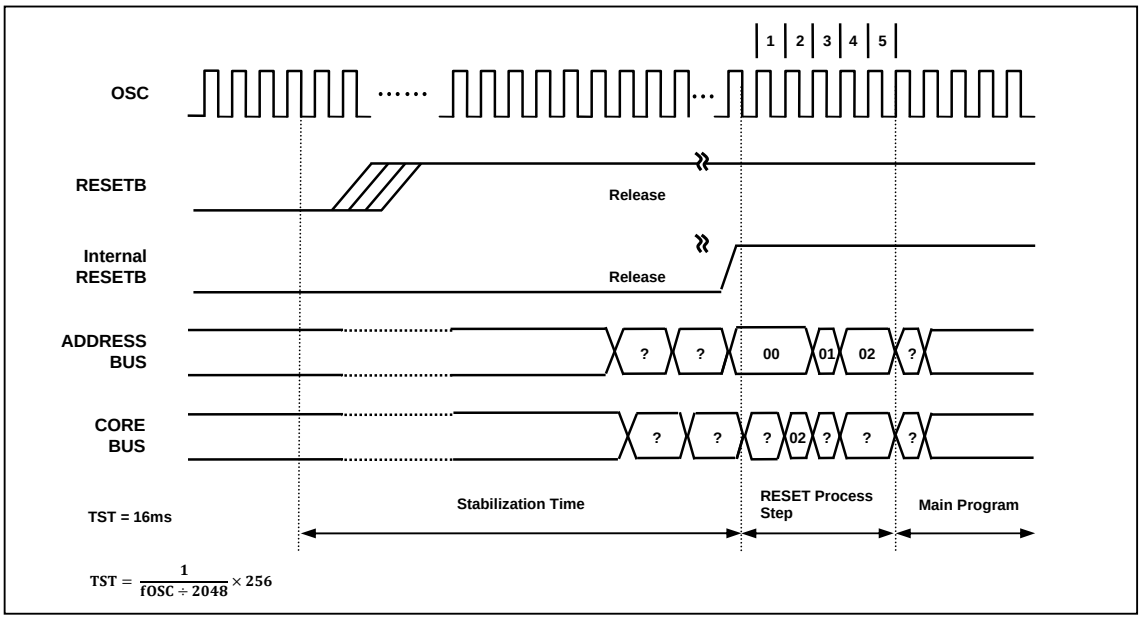



Figure 13-7 Timing Diagram after RESET

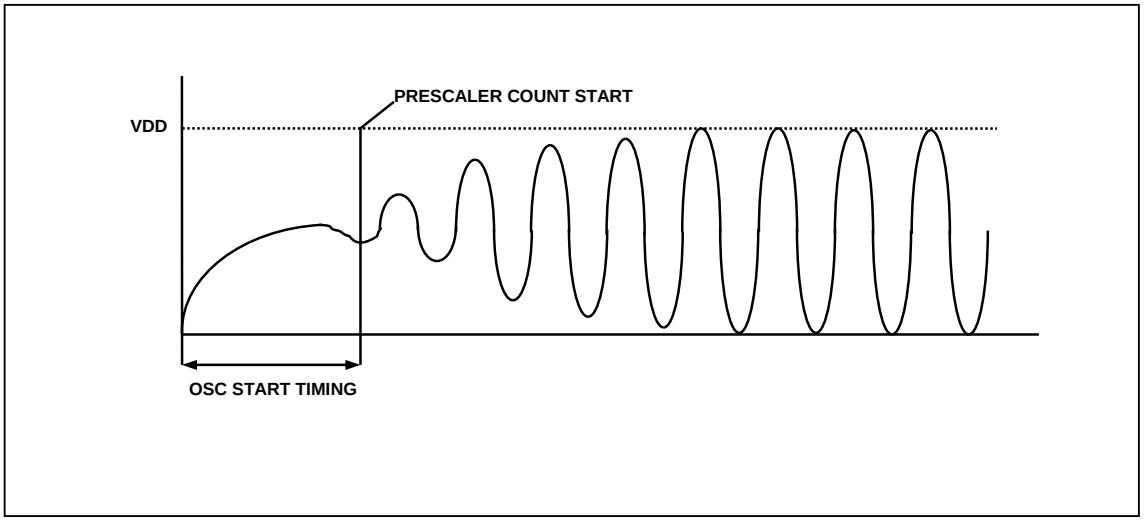



Figure 13-8 Oscillator generating waveform example

Note) as shown Figure 13-8, the stable generating time is not included in the start-up time.

## 13.7 Brown Out Detector Processor

The Z51F0811 has an On-chip Brown-out detection circuit for monitoring the VDD level during operation by comparing it to a fixed trigger level. The trigger level for the BOD can be selected by BODLS[1:0] bit to be 1.6V, 2.5V, 3.6V or 4.2V. In the STOP mode, this will contribute significantly to the total current consumption. So to minimize the current consumption, the BODEN bit is set to off by software.

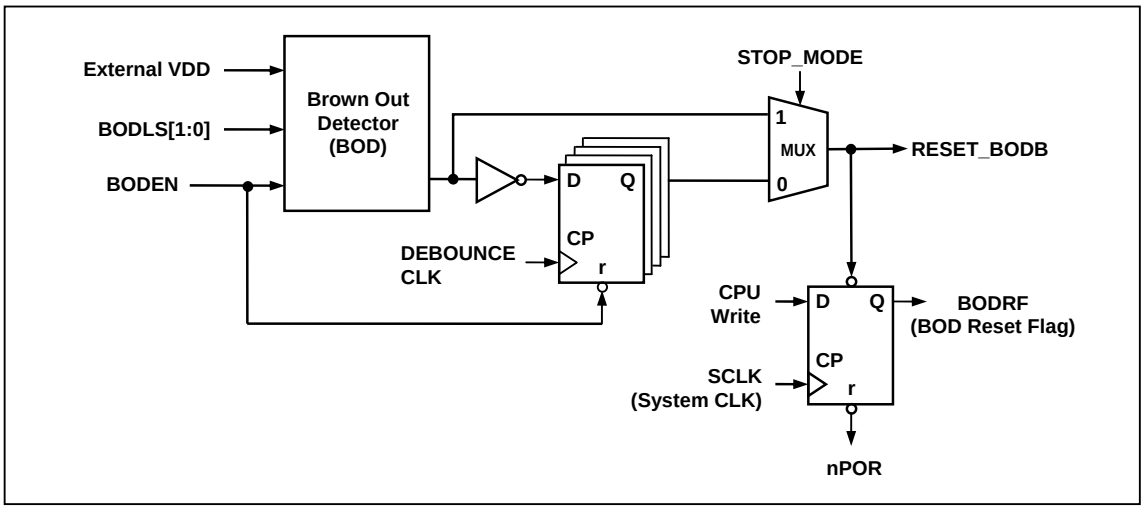



Figure 13-9 Block Diagram of BOD

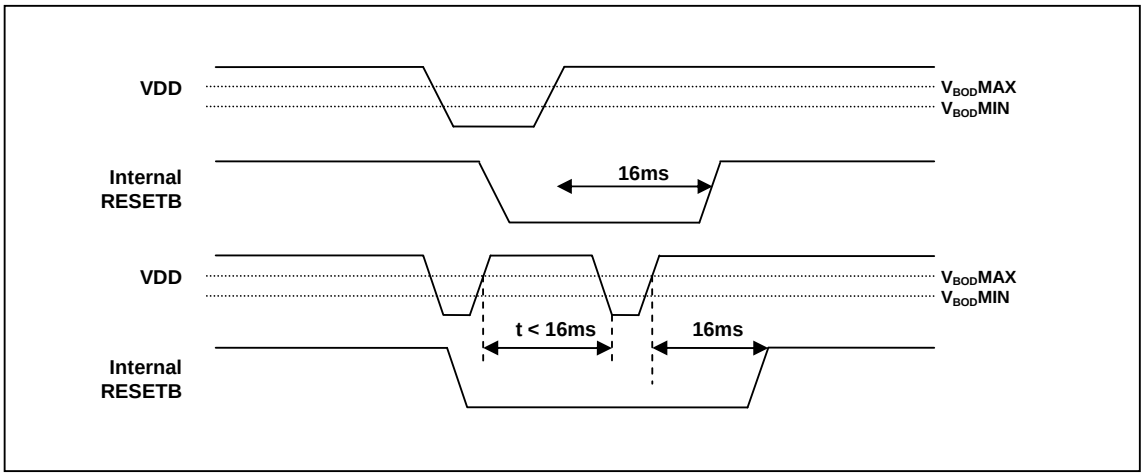



Figure 13-10 Internal Reset at the power fail situation

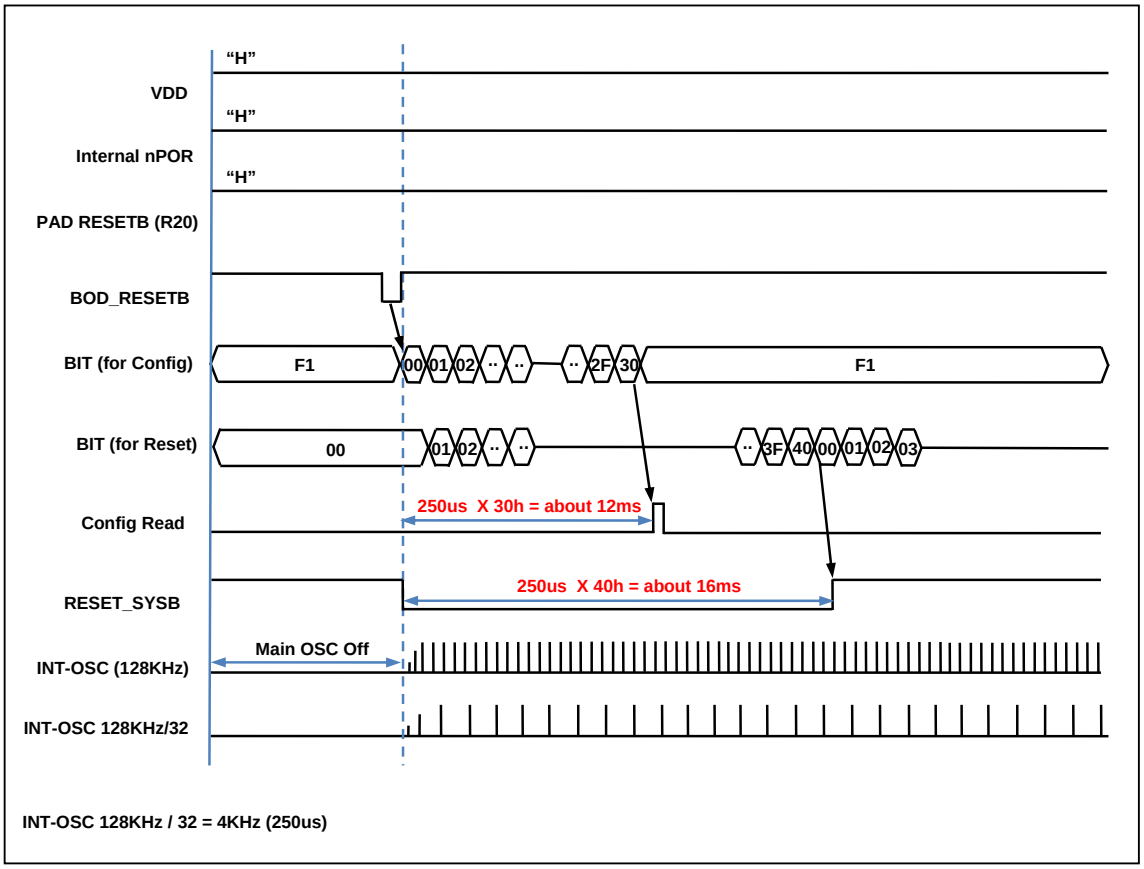



Figure 13-11 Configuration timing when BOD RESET

### 13.7.1 Register Map

Table 13-3 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| BODR | 86H | R/W | 81H | BOD Control Register |

### 13.7.2 Reset Operation Register description

Reset control Register consists of the BOD Control Register (BODR).

### 13.7.3 Register description for Reset Operation

**BODR (BOD Control Register) : 86H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PORF | EXTRF | WDTRF | OCDRF | BODRF | BODLS[1] | BODLS[0] | BODEN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 81H

**PORF** Power-On Reset flag bit. The bit is reset by writing '0' to this bit.

0 No detection

1 Detection

**EXTRF** External Reset flag bit. The bit is reset by writing ‘0’ to this bit or by Power ON reset.

0 No detection

1 Detection

**WDTRF** Watch Dog Reset flag bit. The bit is reset by writing ‘0’ to this bit or by Power ON reset.

0 No detection

1 Detection

**OCDRF** On-Chip Debug Reset flag bit. The bit is reset by writing ‘0’ to this bit or by Power ON reset.

0 No detection

1 Detection

**BODRF** Brown-Out Reset flag bit. The bit is reset by writing ‘0’ to this bit or by Power ON reset.

0 No detection

1 Detection

**BODLS[1:0]** BOD level Voltage

| BODLS1 | BODLS0 | Description |
|---|---|---|
| 0 | 0 | 1.6V |
| 0 | 1 | 2.5V |
| 1 | 0 | 3.6V |
| 1 | 1 | 4.2V |

**BODEN** BOD operation

0 BOD disable

1 BOD enable

# 14. On-chip Debug System

## 14.1 Overview

### 14.1.1 Description

On-chip debug System (OCD) of Z51F0811 can be used for programming the non-volatile memories and on-chip debugging. Detailed descriptions for programming via the OCD interface can be found in the following chapter.

Figure 14-1 shows a block diagram of the OCD interface and the On-chip Debug system.

### 14.1.2 Feature

- Two-wire external interface: 1-wire serial clock input, 1-wire bi-directional serial data bus
- Debugger Access to:
  - All Internal Peripheral Units
  - Internal data RAM
  - Program Counter
  - Flash and Data EEPROM Memories
- Extensive On-chip Debug Support for Break Conditions, Including
  - Break Instruction
  - Single Step Break
  - Program Memory Break Points on Single Address
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the two-wire Interface
  - On-chip Debugging Supported by Dr.Choice®
- Operating frequency

  Supports the maximum frequency of the target MCU

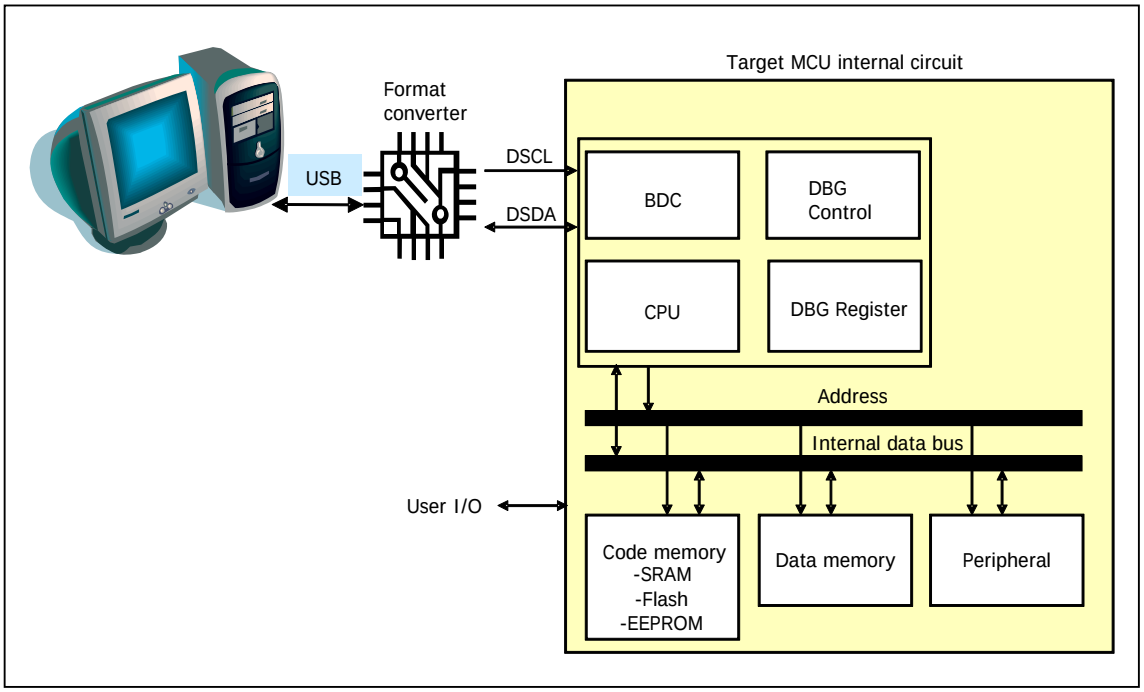



Figure 14-1 Block Diagram of On-chip Debug System

## 14.2 Two-pin external interface

### 14.2.1 Basic transmission packet

- 10-bit packet transmission using two-pin interface.
- 1-packet consists of 8-bit data, 1-bit parity and 1-bit acknowledge.
- Parity is even of '1' for 8-bit data in transmitter.
- Receiver generates acknowledge bit as '0' when transmission for 8-bit data and its parity has no error.
- When transmitter has no acknowledge (Acknowledge bit is '1' at tenth clock), error process is executed in transmitter.
- When acknowledge error is generated, host PC makes stop condition and transmits command which has error again.
- Background debugger command is composed of a bundle of packet.
- Star condition and stop condition notify the start and the stop of background debugger command respectively.

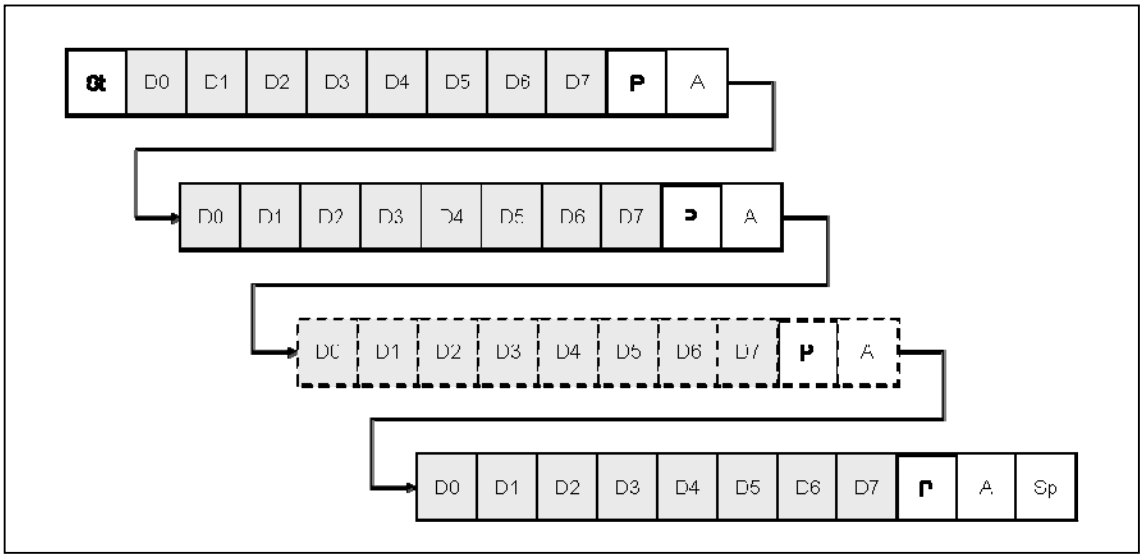



Figure 14-2 10-bit transmission packet

### 14.2.2 Packet transmission timing

#### 14.2.2.1 Data transfer

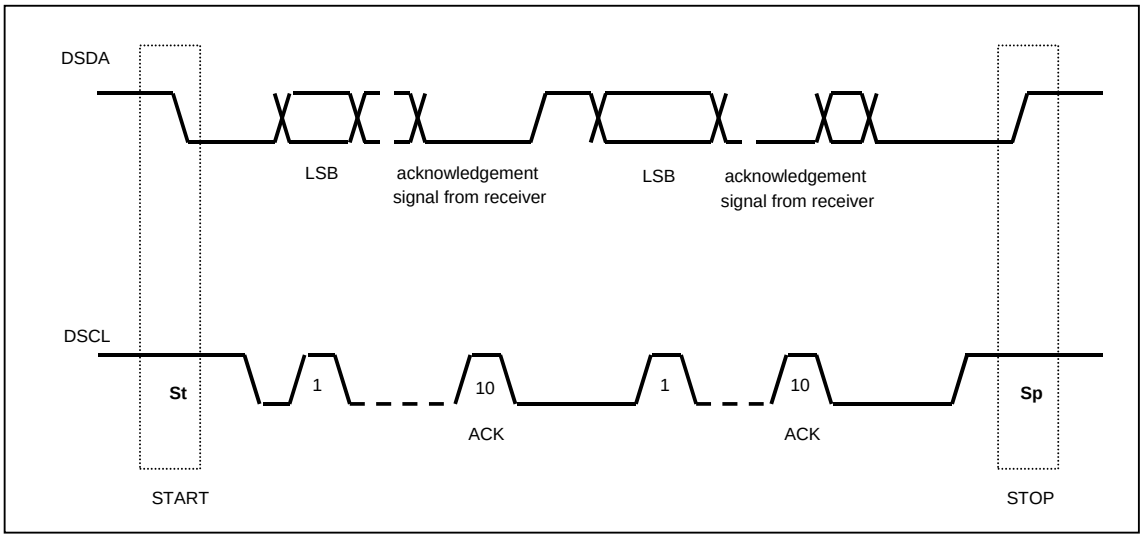



Figure 14-3 Data transfer on the twin bus

#### 14.2.2.2 Bit transfer

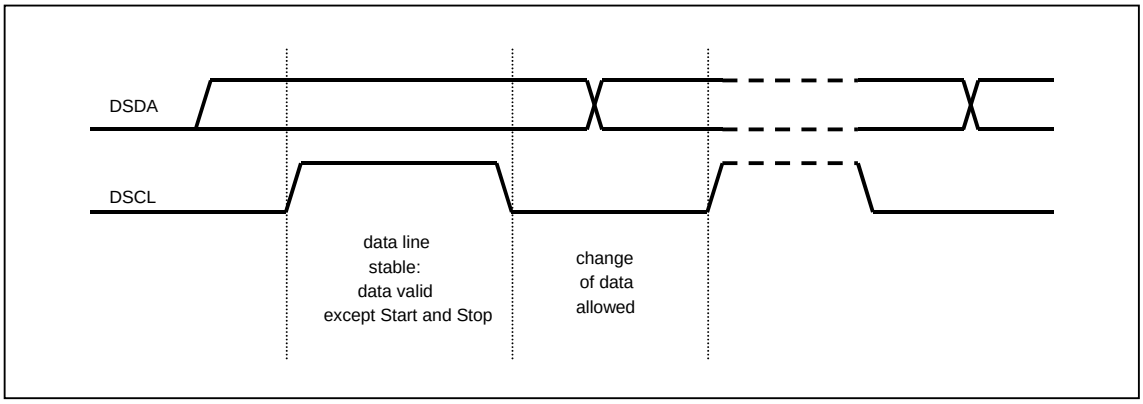



Figure 14-4 Bit transfer on the serial bus

#### 14.2.2.3 Start and stop condition

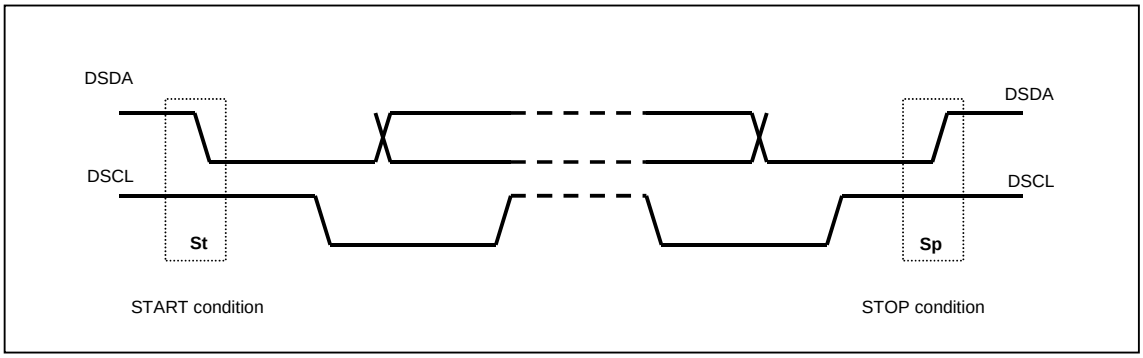



Figure 14-5 Start and stop condition

#### 14.2.2.4 Acknowledge bit

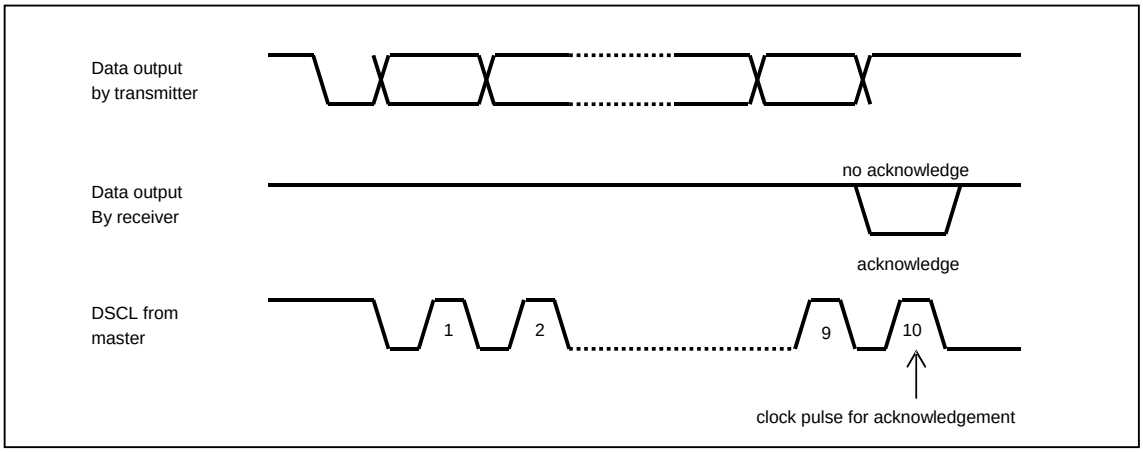



Figure 14-6 Acknowledge on the serial bus

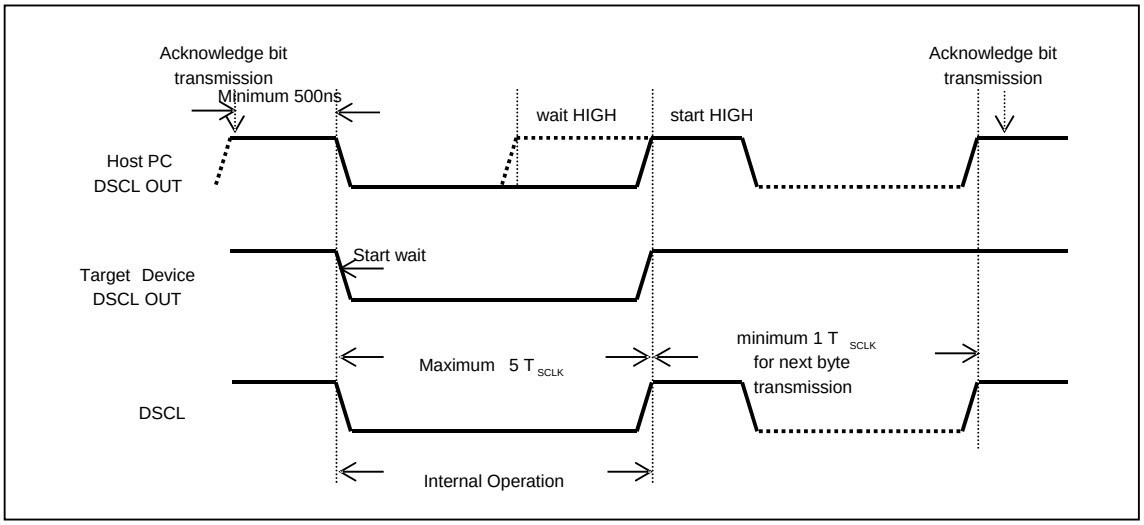



Figure 14-7 Clock synchronization during wait procedure

### 14.2.3 Connection of transmission

Two-pin interface connection uses open-drain (wire-AND bidirectional I/O).

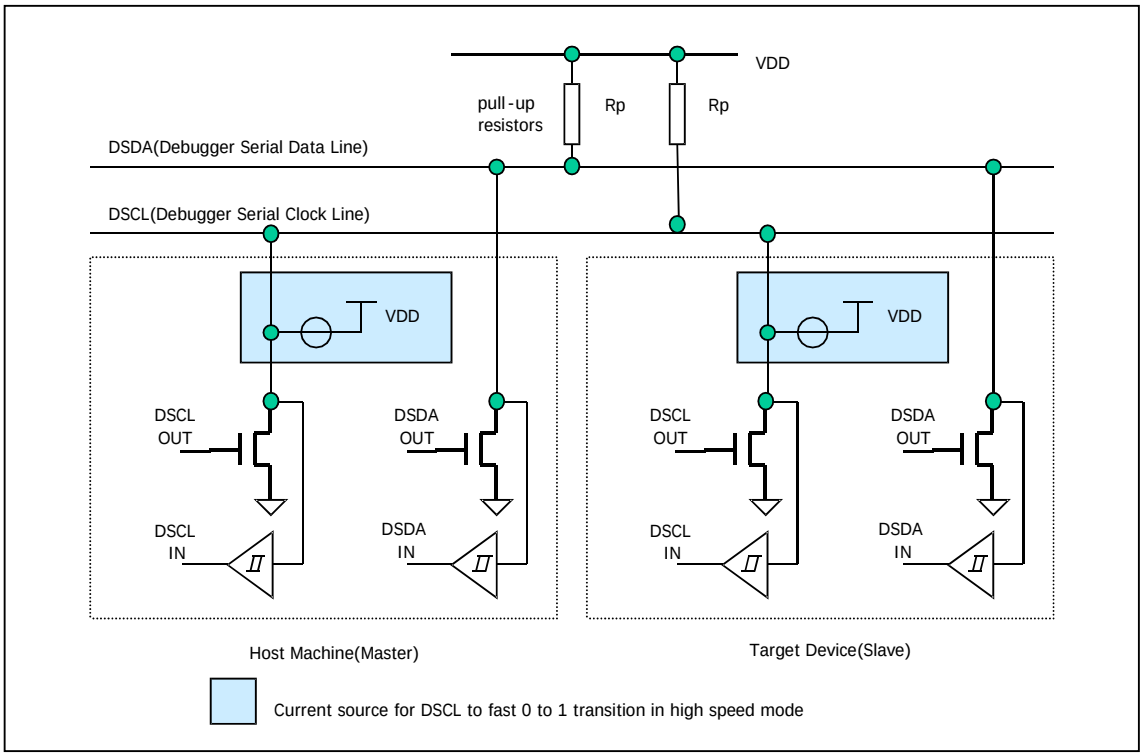



Figure 14-8 Connection of transmission

# 15. Memory Programming

## 15.1 Overview

### 15.1.1 Description

Z51F0811 incorporates flash and data EEPROM memory to which a program can be written, erased, and overwritten while mounted on the board. Also, data EEPROM can be programmed or erased in user program. Flash area can be programmed in only OCD or parallel ROM mode.

Serial ISP modes and byte-parallel ROM writer mode are supported.

### 15.1.2 Features

- Flash Size : 8Kbytes
- Single power supply program and erase
- Command interface for fast program and erase operation
- Up to 10,000 program/erase cycles at typical voltage and temperature for flash memory
- Up to 100,000 program/erase cycles at typical voltage and temperature for data EEPROM memory
- Security feature

## 15.2 Flash and EEPROM Control and status register

Registers to control Flash and Data EEPROM are Mode Register (FEMR), Control Register (FECR), Status Register (FESR), Time Control Register (FETCR), Address Low Register (FEARL), Address Middle Register (FEARM), address High Register (FEARH) and Data Register (FEDR). They are mapped to SFR area and can be accessed only in programming mode.

### 15.2.1 Register Map

Table 15-1 Register Map

| Name | Address | Dir | Default | Description |
|---|---|---|---|---|
| FEMR | EAH | R/W | 00H | Flash and EEPROM Mode Register |
| FECR | EBH | R/W | 03H | Flash and EEPROM Control Register |
| FESR | ECH | R/W | 80H | Flash and EEPROM Status Register |
| FETCR | EDH | R/W | 00H | Flash and EEPROM Time Control Register |
| FEARL | F2H | R/W | 00H | Flash and EEPROM Address Low Register |
| FEARM | F3H | R/W | 00H | Flash and EEPROM Address Middle Register |
| FEARH | F4H | R/W | 00H | Flash and EEPROM Address High Register |
| FEDR | F5H | R/W | 00H | Flash and EEPROM Data Register |

### 15.2.2 Register description for Flash and EEPROM

**FEMR (Flash and EEPROM Mode Register) : EAH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FSEL | ESEL | PGM | ERASE | PBUFF | OTPE | VFY | FEEN |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**FSEL** Select flash memory.
- 0 Deselect flash memory
- 1 Select flash memory

**ESEL** Select data EEPROM
- 0 Deselect data EEPROM
- 1 Select data EEPROM

**PGM** Enable program or program verify mode with VFY
- 0 Disable program or program verify mode
- 1 Enable program or program verify mode

**ERASE** Enable erase or erase verify mode with VFY
- 0 Disable erase or erase verify mode
- 1 Enable erase or erase verify mode

**PBUFF** Select page buffer
- 0 Deselect page buffer
- 1 Select page buffer

**OTPE** Select OTP area instead of program memory
- 0 Deselect OTP area
- 1 Select OTP area

**VFY** Set program or erase verify mode with PGM or ERASE

Program Verify: PGM=1, VFY=1

Erase Verify: ERASE=1, VFY=1

**FEEN** Enable program and erase of Flash and data EEPROM. When inactive, it is possible to read as normal mode
- 0 Disable program and erase
- 1 Enable program and erase

### FECR (Flash and EEPROM Control Register) : EBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AEF | AEE | EXIT1 | EXIT0 | WRITE | READ | nFERST | nPBRST |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 03H

**AEF** Enable flash bulk erase mode

0 Disable bulk erase mode of Flash memory

1 Enable bulk erase mode of Flash memory

**AEE** Enable data EEPROM bulk erase mode

0 Disable bulk erase mode of data EEPROM

1 Enable bulk erase mode of data EEPROM

**EXIT[1:0]** Exit from program mode. It is cleared automatically after 1 clock

| EXIT1 | EXIT0 | Description |
|---|---|---|
| 0 | 0 | Don't exit from program mode |
| 0 | 1 | Don't exit from program mode |
| 1 | 0 | Don't exit from program mode |
| 1 | 1 | Exit from program mode |

**WRITE** Start to program or erase of Flash and data EEPROM. It is cleared automatically after 1 clock

0 No operation

1 Start to program or erase of Flash and data EEPROM

**READ** Start auto-verify of Flash or data EEPROM. It is cleared automatically after 1 clock

0 No operation

1 Start auto-verify of Flash or data EEPROM

**nFERST** Reset Flash or data EEPROM control logic. It is cleared automatically after 1 clock

0 No operation

1 Reset Flash or data EEPROM control logic.

**nPBRST** Reset page buffer with PBUFF. It is cleared automatically after 1 clock

| PBUFF | nPBRST | Description |
|---|---|---|
| 0 | 0 | Page buffer reset |
| 1 | 0 | Write checksum reset |

WRITE and READ bits can be used in program, erase and verify mode with FEAR registers. Read or writes for memory cell or page buffer uses read and write enable signals from memory controller. Indirect address mode with FEAR is only allowed to program, erase and verify

### FESR (Flash and EEPROM Status Register) : ECH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PEVBSY | VFYGOOD | - | - | ROMINT | WMODE | EMODE | VMODE |
| R | R/W | R | R | R/W | R | R | R |

Initial value : 80H

**PEVBSY** Operation status flag. It is cleared automatically when operation starts. Operations are program, erase or verification

0 Busy (Operation processing)

1 Complete Operation

**VFYGOOD** Auto-verification result flag.

0 Auto-verification fails

1 Auto-verification successes

**ROMINT** Flash and Data EEPROM interrupt request flag. Auto-cleared when program/erase/verify starts. Active in program/erase/verify completion

0 No interrupt request.

1 Interrupt request.

**WMODE** Write mode flag

**EMODE** Erase mode flag

**VMODE** Verify mode flag

### FEARL (Flash and EEPROM address low Register) : F2H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ARL7 | ARL6 | ARL5 | ARL4 | ARL3 | ARL2 | ARL1 | ARL0 |
| W | W | W | W | W | W | W | W |

Initial value : 00H

**ARL[7:0]** Flash and EEPROM address low

### FEARM (Flash and EEPROM address middle Register) : F3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ARM7 | ARM6 | ARM5 | ARM4 | ARM3 | ARM2 | ARM1 | ARM0 |
| W | W | W | W | W | W | W | W |

Initial value : 00H

**ARM[7:0]** Flash and EEPROM address middle

### FEARH (Flash and EEPROM address high Register) : F4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ARH7 | ARH6 | ARH5 | ARH4 | ARH3 | ARH2 | ARH1 | ARH0 |
| W | W | W | W | W | W | W | W |

Initial value : 00H

**ARH[7:0]** Flash and EEPROM address high

FEAR registers are used for program, erase and auto-verify. In program and erase mode, it is page address and ignored the same least significant bits as the number of bits of page address. In auto-verify mode, address increases automatically by one.

FEARs are write-only register. Reading these registers returns 24-bit checksum result

### FEDR (Flash and EEPROM data control Register) : F5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FEDR7 | FEDR6 | FEDR5 | FEDR4 | FEDR3 | FEDR2 | FEDR1 | FEDR0 |
| W | W | W | W | W | W | W | W |

Initial value : 00H

**FEDR[7:0]** Flash and EEPROM data

Data register. In no program/erase/verify mode, READ/WRITE of FECR read or write data from EEPROM or Flash to this register or from this register to Flash or EEPROM.

The sequence of writing data to this register is used for EEPROM program entry. The mode entrance sequence is to write 0xA5 and 0x5A to it in order.

### FETCR (Flash and EEPROM Time control Register) : EDH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TCR7 | TCR6 | TCR5 | TCR4 | TCR3 | TCR2 | TCR1 | TCR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value : 00H

**TCR[7:0]** Flash and EEPROM Time control

Program and erase time is controlled by setting FETCR register. Program and erase timer uses 10-bit counter. It increases by one at each divided system clock frequency(=SCLK/128). It is cleared when program or erase starts. Timer stops when 10-bit counter is same to FETCR. PEVBSY is cleared when program, erase or verify starts and set when program, erase or verify stops.

Max program/erase time at 12Mhz system clock : (255+1) * 2 * (83.3ns * 128) = 5.459ms

In the case of 10% of error rate of counter source clock, program or erase time is 5.0~5.9ms

* * Program/erase time calculation

for page write or erase, Tpe = (TCON+1) * 2 * (SCLK * 128)

for bulk erase, Tbe = (TCON+1) * 4 * (SCLK * 128)

Table 15-2 Program/erase Time

| | Min | Typ | Max | Unit |
|---|---|---|---|---|
| program/erase Time | 2.4 | 2.5 | 2.6 | ms |

※ Recommended program/erase time at 12Mhz (FETCR = 75h)

## 15.3 Memory map

### 15.3.1 Flash Memory Map

Program memory uses 8-Kbyte of Flash memory. It is read by byte and written by byte or page. One page is 32-byte

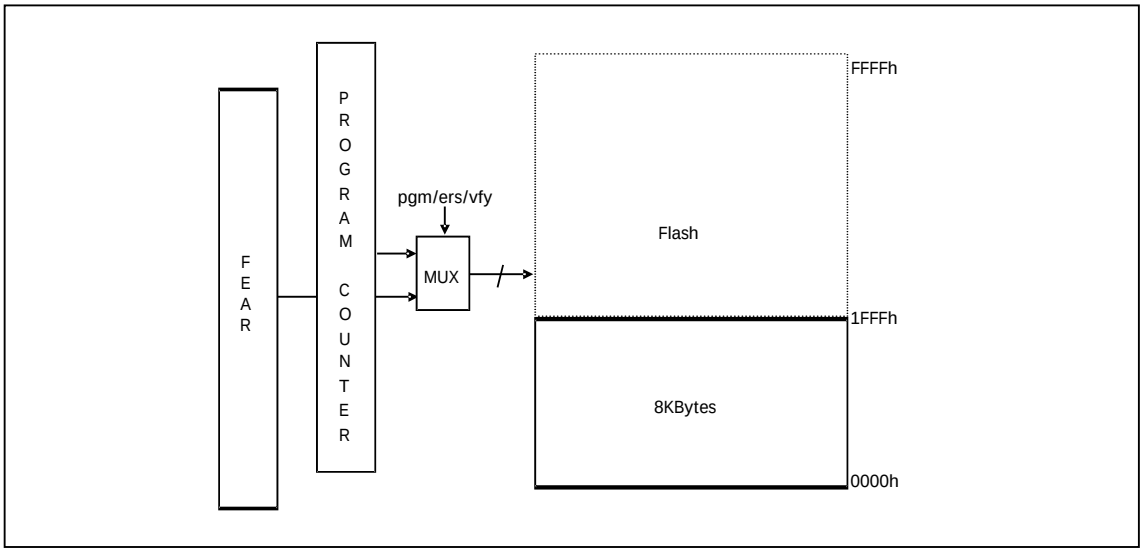



Figure 15-1 Flash Memory Map

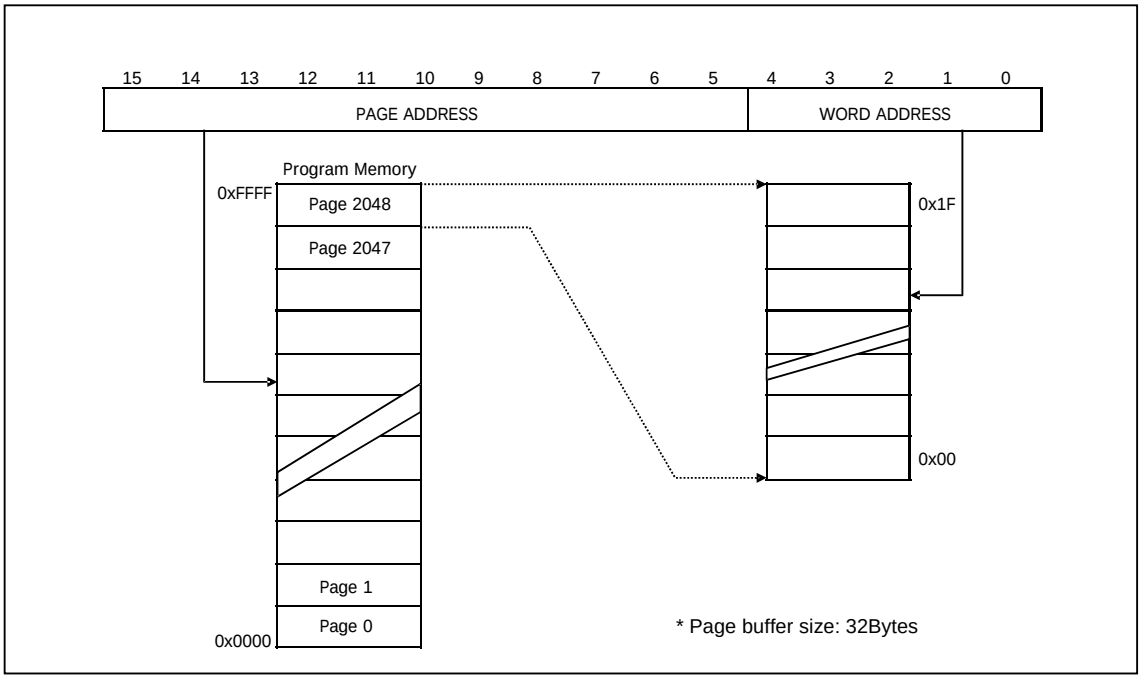



Figure 15-2 Address configuration of Flash memory

### 15.3.2 Data EEPROM Memory Map

Data EEPROM memory uses 512-byte of EEPROM. It is read by byte and written by byte or page. One page is 16-byte. It is mapped to external data memory of 8051

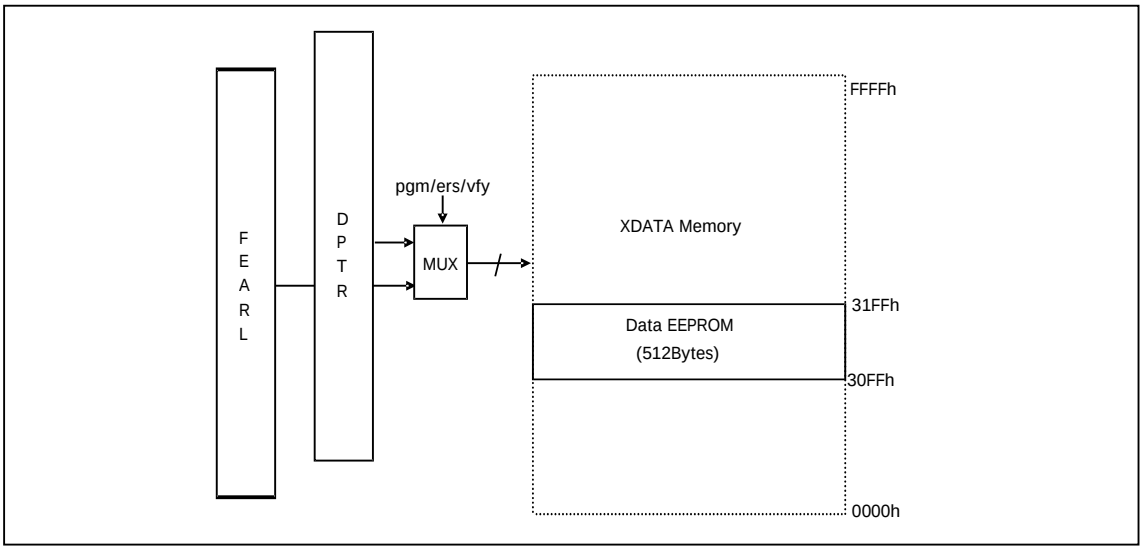



Figure 15-3 Data EEPROM memory map

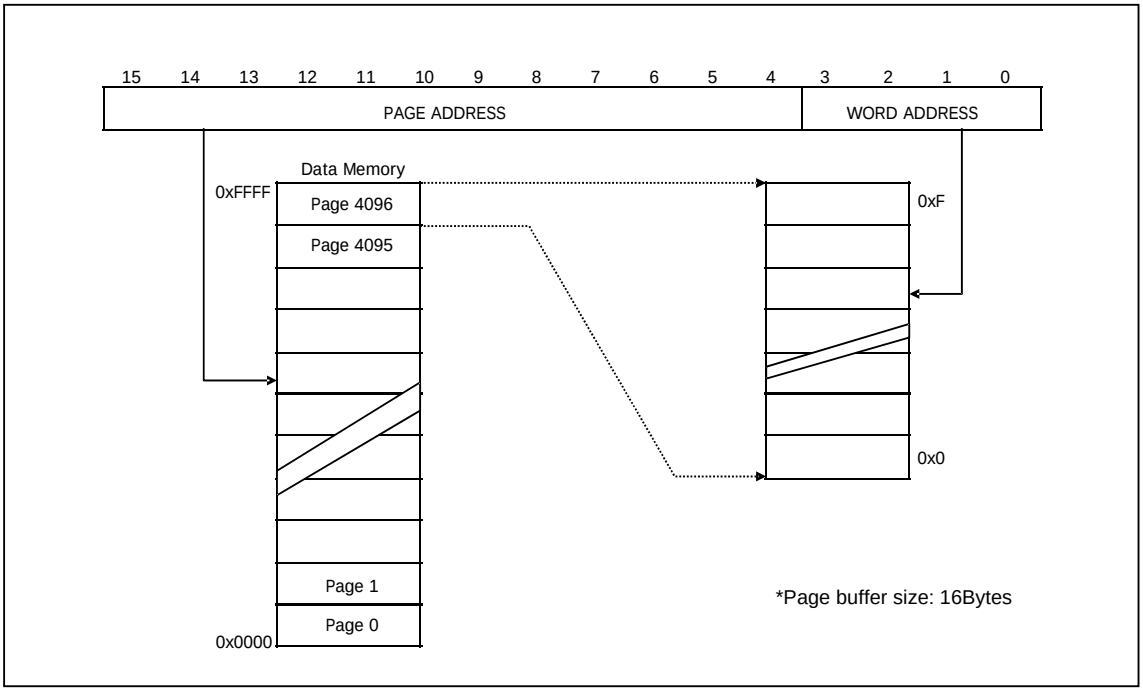



Figure 15-4 Address configuration of data EEPROM

## 15.4 Serial In-System Program Mode

Serial in-system program uses the interface of debugger which uses two wires. Refer to chapter 14 in details about debugger

### 15.4.1 Flash operation

**Configuration** (This Configuration is just used for follow description)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| - | FEMR[4] &[1] | FEMR[5] &[1] | - | - | FEMR[2] | FECR[6] | FECR[7] |
| - | ERASE&VFY | PGM&VFY | - | - | OTPE | AEE | AEF |

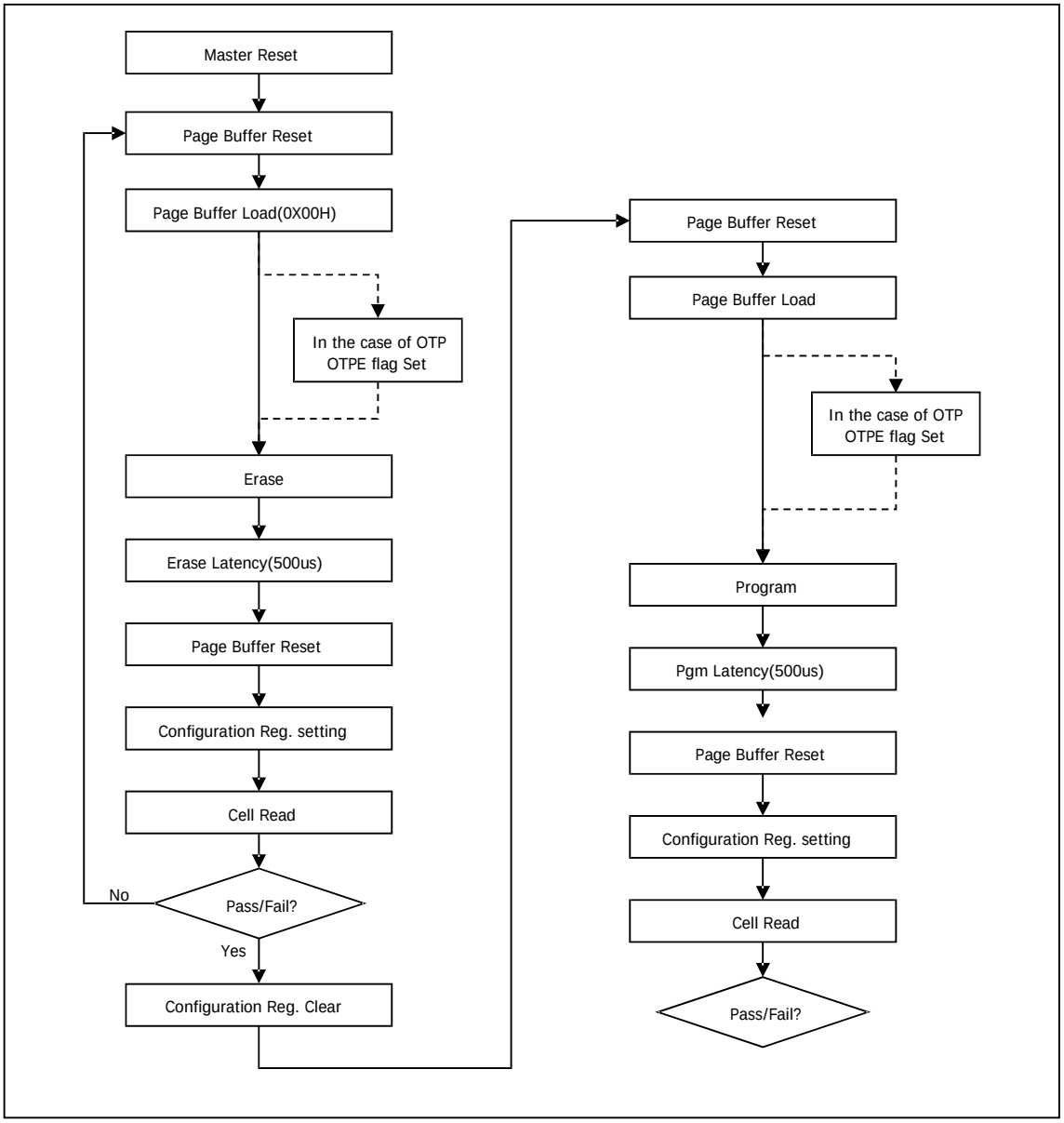



Figure 15-5 The sequence of page program and erase of Flash memory

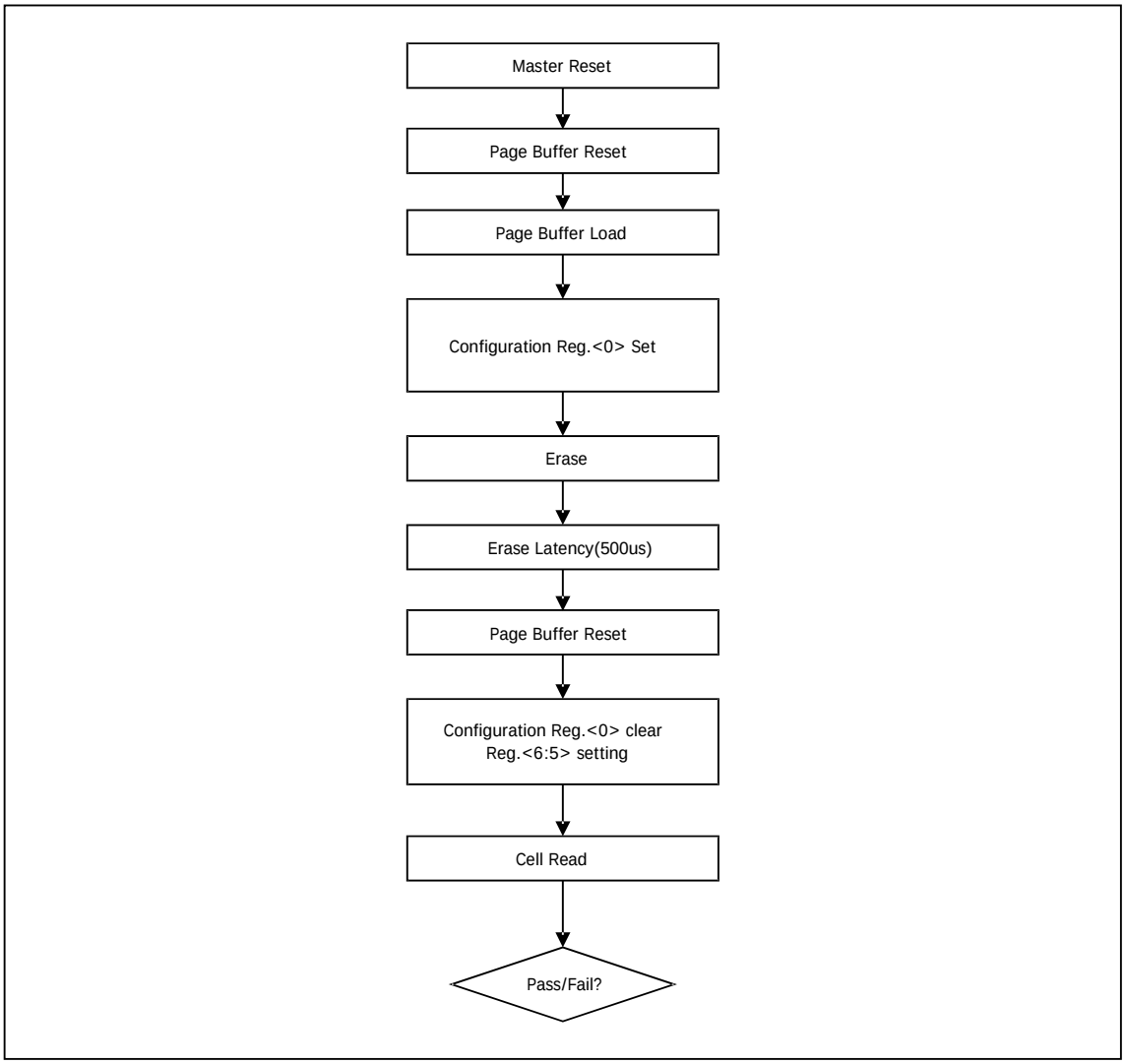



Figure 15-6 The sequence of bulk erase of Flash memory

#### 15.4.1.1 Flash Read

Step 1. Enter OCD(=ISP) mode.
Step 2. Set ENBDM bit of BCR.
Step 3. Enable debug and Request debug mode.
Step 4. Read data from Flash.

#### 15.4.1.2 Enable program mode

Step 1. Enter OCD(=ISP) mode.[1]
Step 2. Set ENBDM bit of BCR.
Step 3. Enable debug and Request debug mode.
Step 4. Enter program/erase mode sequence.[2]
(1) Write 0xAA to 0xF555.

(2) Write 0x55 to 0xFAAA.

(3) Write 0xA5 to 0xF555.

[1] Refer to how to enter ISP mode..

[2] Command sequence to activate Flash write/erase mode. It is composed of sequentially writing data of Flash memory.

#### 15.4.1.3 Flash write mode

Step 1. Enable program mode.

Step 2. Reset page buffer. FEMR: 1000_0001 FECR:0000_0010

Step 3. Select page buffer. FEMR:1000_1001

Step 4. Write data to page buffer.(Address automatically increases by twin.)

Step 5. Set write mode. FEMR:1010_0001

Step 6. Set page address. FEARH:FEARM:FEARL=20'hx_xxxx

Step 7. Set FETCR.

Step 8. Start program. FECR:0000_1011

Step 9. Insert one NOP operation

Step 10. Read FESR until PEVBSY is 1.

Step 11. Repeat step2 to step 8 until all pages are written.

#### 15.4.1.4 Flash page erase mode

Step 1. Enable program mode.

Step 2. Reset page buffer. FEMR: 1000_0001 FECR:0000_0010

Step 3. Select page buffer. FEMR:1000_1001

Step 4. Write 'h00 to page buffer. (Data value is not important.)

Step 5. Set erase mode. FEMR:1001_0001

Step 6. Set page address. FEARH:FEARM:FEARL=20'hx_xxxx

Step 7. Set FETCR.

Step 8. Start erase. FECR:0000_1011

Step 9. Insert one NOP operation

Step 10. Read FESR until PEVBSY is 1.

Step 11. Repeat step2 to step 8 until all pages are erased.

#### 15.4.1.5 Flash bulk erase mode

Step 1. Enable program mode.

Step 2. Reset page buffer. FEMR: 1000_0001 FECR:0000_0010

Step 3. Select page buffer. FEMR:1000_1001

Step 4. Write 'h00 to page buffer. (Data value is not important.)

Step 5. Set erase mode. FEMR:1001_0001.

(Only main cell area is erased. For bulk erase including OTP area, select OTP area.(set FEMR to 1000_1101.)

Step 6. Set FETCR

Step 7. Start bulk erase. FECR:1000_1011
Step 8. Insert one NOP operation
Step 9. Read FESR until PEVBSY is 1.

#### 15.4.1.6 Flash OTP area read mode

Step 1. Enter OCD(=ISP) mode.
Step 2. Set ENBDM bit of BCR.
Step 3. Enable debug and Request debug mode.
Step 4. Select OTP area. FEMR:1000_0101
Step 5. Read data from Flash.

#### 15.4.1.7 Flash OTP area write mode

Step 1. Enable program mode.
Step 2. Reset page buffer. FEMR: 1000_0001 FECR:0000_0010
Step 3. Select page buffer. FEMR:1000_1001
Step 4. Write data to page buffer.(Address automatically increases by twin.)
Step 5. Set write mode and select OTP area. FEMR:1010_0101
Step 6. Set page address. FEARH:FEARM:FEARL=20'hx_xxxx
Step 7. Set FETCR.
Step 8. Start program. FECR:0000_1011
Step 9. Insert one NOP operation
Step 10. Read FESR until PEVBSY is 1.

#### 15.4.1.8 Flash OTP area erase mode

Step 1. Enable program mode.
Step 2. Reset page buffer. FEMR: 1000_0001 FECR:0000_0010
Step 3. Select page buffer. FEMR:1000_1001
Step 4. Write 'h00 to page buffer. (Data value is not important.)
Step 5. Set erase mode and select OTP area. FEMR:1001_0101
Step 6. Set page address. FEARH:FEARM:FEARL=20'hx_xxxx
Step 7. Set FETCR.
Step 8. Start erase. FECR:0000_1011
Step 9. Insert one NOP operation
Step 10. Read FESR until PEVBSY is 1.

#### 15.4.1.9 Flash program verify mode

Step 1. Enable program mode.
Step 2. Set program verify mode. FEMR:1010_0011
Step 3. Read data from Flash.

#### 15.4.1.10 OTP program verify mode

Step 1. Enable program mode.
Step 2. Set program verify mode. FEMR:1010_0111
Step 3. Read data from Flash.

#### 15.4.1.11 Flash erase verify mode

Step 1. Enable program mode.
Step 2. Set erase verify mode. FEMR:1001_0011
Step 3. Read data from Flash.

#### 15.4.1.12 Flash page buffer read

Step 1. Enable program mode.
Step 2. Select page buffer. FEMR:1000_1001
Step 3. Read data from Flash.

### 15.4.2 Data EEPROM operation

Program and erase operation of Data EEPROM are executed by direct and indirect address mode.
Direct address mode uses external data area of 8051. Indirect address mode uses address register of SFR area..

#### 15.4.2.1 Data EEPROM Read

Step 1. Enter OCD(=ISP) mode.
Step 2. Set ENBDM bit of BCR.
Step 3. Enable debug and Request debug mode.
Step 4. Read data from Data EEPROM.

#### 15.4.2.2 Enable program mode

Step 1. Enter OCD(=ISP) mode.[1]
Step 2. Set ENBDM bit of BCR.
Step 3. Enable debug and Request debug mode.
Step 4. Enter program/erase mode sequence.[2]
(1) Write 0xA5 to FEDR.
(2) Write 0x5A to FEDR.

[1] Refer to how to enter ISP mode..
[2] Command sequence to activate data EEPROM write/erase mode. It is composed of sequentially writing to data register(FEDR)

#### 15.4.2.3 EEPROM write mode

Step 1. Enable program mode.
Step 2. Reset page buffer. FEMR: 0100_0001 FECR:0000_0010
Step 3. Select page buffer. FEMR:0100_1001
Step 4. Write data to page buffer.(Address automatically increases by twin.)
Step 5. Set write mode. FEMR:0110_0001
Step 6. Set page address. FEARH:FEARM:FEARL=20'hx_xxxx
Step 7. Set FETCR.
Step 8. Start program. FECR:0000_1011
Step 9. Insert one NOP operation
Step 10. Read FESR until PEVBSY is 1.
Step 11. Repeat step2 to step 8 until all pages are written.

#### 15.4.2.4 EEPROM page erase mode

Step 1. Enable program mode.
Step 2. Reset page buffer. FEMR: 0100_0001 FECR:0000_0010
Step 3. Select page buffer. FEMR:0100_1001
Step 4. Write 'h00 to page buffer. (Data value is not important.)
Step 5. Set erase mode. FEMR:0101_0001
Step 6. Set page address. FEARH:FEARM:FEARL=20'hx_xxxx
Step 7. Set FETCR.
Step 8. Start erase. FECR:0000_1011
Step 9. Insert one NOP operation
Step 10. Read FESR until PEVBSY is 1.
Step 11. Repeat step2 to step 8 until all pages are erased.

#### 15.4.2.5 EEPROM bulk erase mode

Step 1. Enable program mode.
Step 2. Reset page buffer. FEMR: 0100_0001 FECR:0000_0010
Step 3. Select page buffer. FEMR:0100_1001
Step 4. Write 'h00 to page buffer. (Data value is not important.)
Step 5. Set erase mode. FEMR:0101_0001.
Step 6. Set FETCR
Step 7. Start bulk erase. FECR:0100_1011
Step 8. Insert one NOP operation
Step 9. Read FESR until PEVBSY is 1.

#### 15.4.2.6 Data EEPROM program verify mode

Step 1. Enable program mode.
Step 2. Set program verify mode. FEMR:0110_0011
Step 3. Read data from Flash.

#### 15.4.2.7 Data EEPROM erase verify mode

Step 1. Enable program mode.
Step 2. Set erase verify mode. FEMR:0101_0011
Step 3. Read data from Flash.

#### 15.4.2.8 Data EEPROM page buffer read

Step 1. Enable program mode.
Step 2. Select page buffer. FEMR:0100_1001
Step 3. Read data from Flash.

### 15.4.3 Summary of Flash and Data EEPROM Program/Erase Mode

Table 15-3 Operation Mode

| Operation mode | | Description |
|---|---|---|
| FLASH | Flash read | Read cell by byte. |
| | Flash write | Write cell by bytes or page. |
| | Flash page erase | Erase cell by page. |
| | Flash bulk erase | Erase the whole cells. |
| | Flash program verify | Read cell in verify mode after programming. |
| | Flash erase verify | Read cell in verify mode after erase. |
| | Flash page buffer load | Load data to page buffer. |
| EEPROM | Data EEPROM read | Read cell by byte. |
| | Data EEPROM write | Write cell by bytes or page. |
| | Data EEPROM page erase | Erase cell by page. |
| | Data EEPROM bulk erase | Erase the whole cells. |
| | Data EEPROM program verify | Read cell in verify mode after programming. |
| | Data EEPROM erase verify | Read cell in verify mode after erase. |
| | Data EEPROM page buffer load | Load data to page buffer. |

## 15.5 Parallel Mode

### 15.5.1 Overview

Parallel program mode transfers address and data by byte. 3-byte address can be entered by one from the lease significant byte of address. If only LSB is changed, only one byte can be transferred. And if the second byte is changed, the first and second byte can be transferred. Upper 4-bit of the most significant byte selects memory to be accessed. Table 15-4 shows memory type to be accessible by parallel mode. Address auto-increment is supported when read or write data without address.

The erase and program sequence of Flash and data EEPROM is identical to that of ISP mode except the entrance of parallel mode . Refer to Table 15-5 for the entrance method for parallel mode.

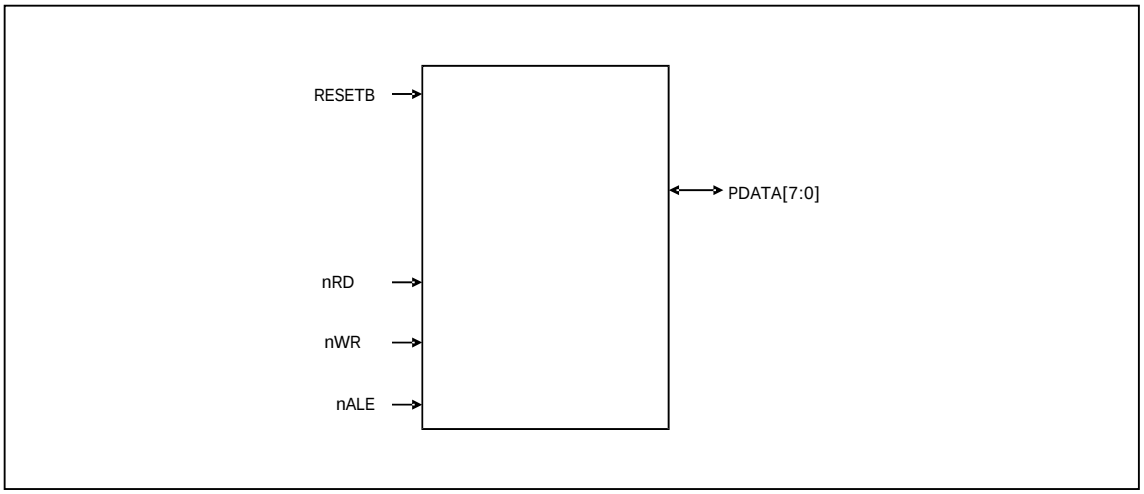



Figure 15-7 Pin diagram for parallel programming

Table 15-4 The selection of memory type by ADDRH[7:4]

| ADDRH[7:4] | | | | Memory Type |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Program Memory |
| 0 | 0 | 0 | 1 | External Memory |
| 0 | 0 | 1 | 0 | SFR |

### 15.5.2 Parallel Mode instruction format

Table 15-5 Parallel mode instruction format

| Instruction | Signal | Instruction Sequence | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| n-byte data read with 3-byte address | nALE | L | | L | | L | | H | | H | | H | | H | |
| | nWR | L | H | L | H | L | H | H | H | H | H | H | H | H | H |
| | nRD | H | H | H | H | H | H | L | H | L | H | L | H | L | H |
| | PDATA | ADDRL | | ADDRM | | ADDRH | | DATA0 | | DATA1 | | --- | | DATAn | |
| n-byte data write with 3-byte address | nALE | L | | L | | L | | H | | H | | H | | H | |
| | nWR | L | H | L | H | L | H | L | H | L | H | L | H | L | H |
| | nRD | H | H | H | H | H | H | H | H | H | H | H | H | H | H |
| | PDATA | ADDRL | | ADDRM | | ADDRH | | DATA0 | | DATA1 | | --- | | DATAn | |
| n-byte data read with 2-byte address | nALE | L | | L | | H | | H | | H | | H | | H | |
| | nWR | L | H | L | H | H | H | H | H | H | H | H | H | H | H |
| | nRD | H | H | H | H | L | H | L | H | L | H | L | H | L | H |
| | PDATA | ADDRL | | ADDRM | | DATA0 | | DATA1 | | DATA2 | | --- | | DATAn | |
| n-byte data write with 2-byte address | nALE | L | | L | | H | | H | | H | | H | | H | |
| | nWR | L | H | L | H | L | H | L | H | L | H | L | H | L | H |
| | nRD | H | H | H | H | H | H | H | H | H | H | H | H | H | H |
| | PDATA | ADDRL | | ADDRM | | DATA0 | | DATA1 | | DATA2 | | --- | | DATAn | |
| n-byte data read with 1-byte address | nALE | L | | H | | H | | H | | H | | H | | H | |
| | nWR | L | H | H | H | L | H | L | H | L | H | L | H | L | H |
| | nRD | H | H | L | H | H | H | H | H | H | H | H | H | H | H |
| | PDATA | ADDRL | | DATA0 | | DATA1 | | DATA2 | | DATA3 | | --- | | DATAn | |
| n-byte data write with 1-byte address | nALE | L | | H | | H | | H | | H | | H | | H | |
| | nWR | L | H | L | H | L | H | L | H | L | H | L | H | L | H |
| | nRD | H | H | H | H | H | H | H | H | H | H | H | H | H | H |
| | PDATA | ADDRL | | DATA0 | | DATA1 | | DATA2 | | DATA3 | | --- | | DATAn | |

### 15.5.3 Parallel Mode timing diagram

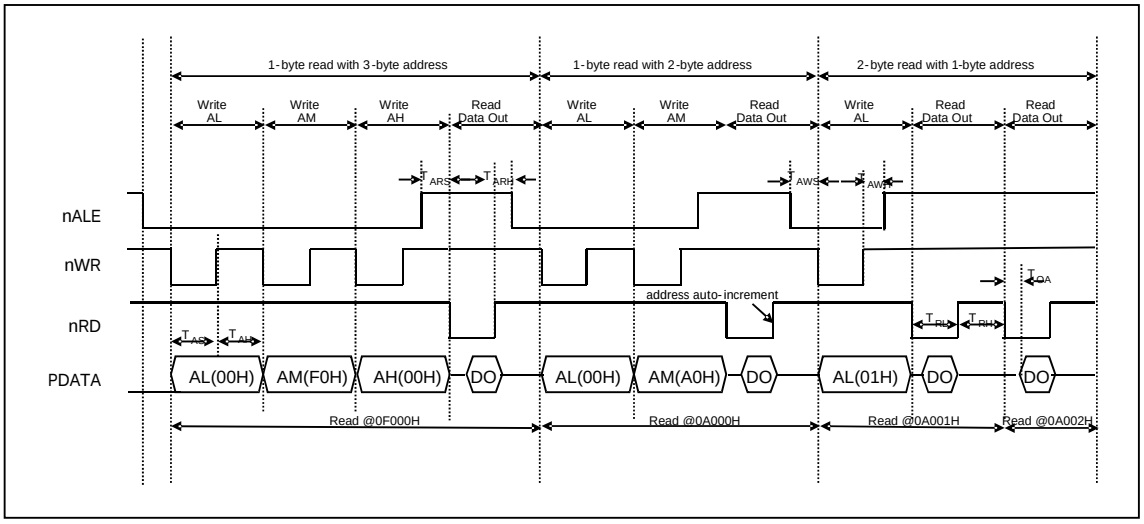



Figure 15-8 Parallel Byte Read Timing of Program Memory

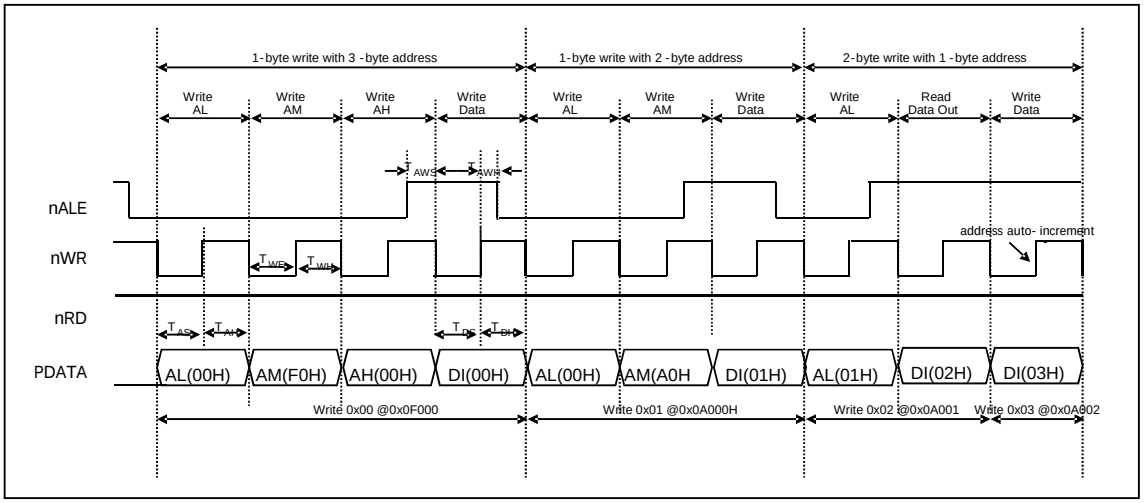



Figure 15-9 Parallel Byte Write Timing of Program Memory

## 15.6 Mode entrance method of ISP and byte-parallel mode

### 15.6.1 Mode entrance method for ISP

| TARGET MODE | DSDA | DSCL | DSDA |
|---|---|---|---|
| OCD(ISP) | ‘hC | ‘hC | ‘hC |

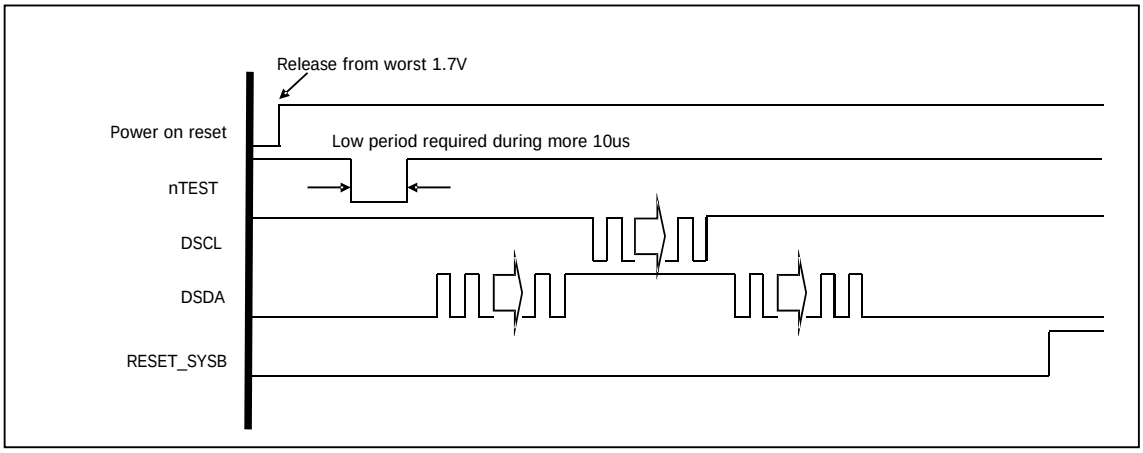



Figure 15-10 ISP mode

### 15.6.2 Mode entrance of Byte-parallel

| TARGET MODE | P0[3:0] | P0[3:0] | P0[3:0] |
|---|---|---|---|
| Byte-Parallel Mode | 4'h5 | 4'hA | 4'h5 |

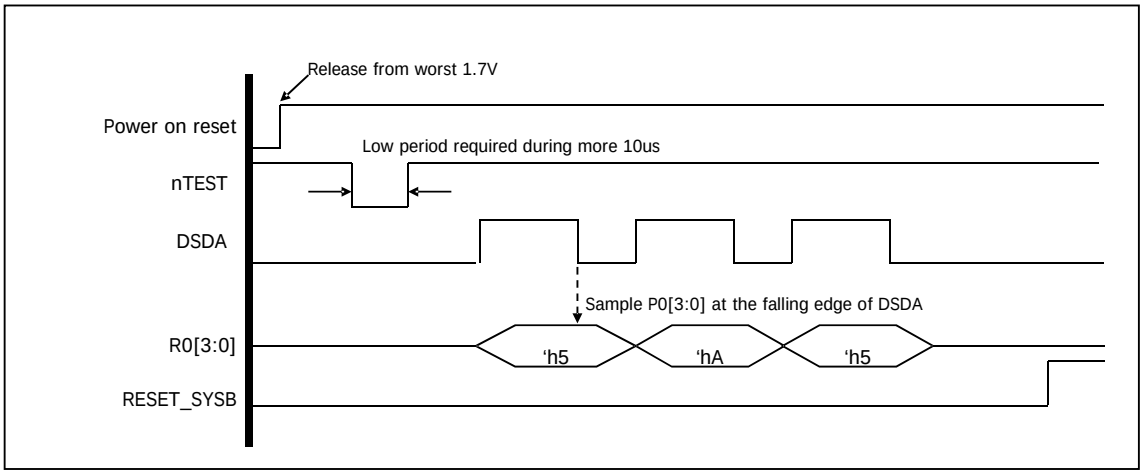



Figure 15-11 Byte-parallel mode

## 15.7 Security

Z51F0811 provides Lock bits which can be left unprogrammed (“0”) or can be programmed (“1”) to obtain the additional features listed in Table 15-6. The Lock bits can be erased to “0” with only the bulk erase command and a value of more than 0x80 at FETCR.

Table 15-6 Security policy using lock-bits

| LOCK MODE | | USER MODE | | | | | | | | | | | | ISP/PMODE | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Flash | | | | DATA EEPROM | | | | OTP | | | | Flash | | | | DATA EEPROM | | | | OTP | | | |
| LOCKE | LOCKF | R | W | PE | BE | R | W | PE | BE | R | W | PE | BE | R | W | PE | BE | R | W | PE | BE | R | W | PE | BE |
| 0 | 0 | O | O | O | X | O | O | O | O | X | X | X | X | O | O | O | O | O | O | O | O | O | O | O | O |
| 0 | 1 | O | O | O | X | O | O | O | O | X | X | X | X | X | X | X | O | O | O | O | O | O | X | X | O |
| 1 | 0 | O | O | O | X | O | O | O | O | X | X | X | X | O | ◇ | ◇ | ◇ | X | X | X | O | O | ◇ | ◇ | ◇ |
| 1 | 1 | O | O | O | X | O | O | O | O | X | X | X | X | X | X | X | ◇ | X | X | X | O | O | X | X | ◇ |

- LOCKF: Lock bit of Flash memory
- LOCKE: Lock bit of data EEPROM
- R: Read
- W: Write
- PE: Page erase
- BE: Bulk Erase
- O: Operation is possible.
- X: Operation is impossible.
- ◇: When LOCKE is programmed, each operation can be done after data EEPROM is erased with the bulk erase command.

# 16. Configure option

## 16.1 Configure option Control Register

### FUSE_CONF (Pseudo-Configure Data) : 2F5DH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BSIZE1 | BSIZE0 | SXINEN | XINENA | RSTDIS | LOCKB | LOCKE | LOCKF |
| R | R | R | R | R | R | R | R |

Initial value : 00H

**BSIZE** Boot Area Hard Lock Size Bit
- 00 0100H~01FFH (default)
- 01 0100H~03FFH
- 10 0100H~07FFH
- 11 0100H~0FFFH

**SXINEN** External Sub Oscillator Enable Bit
- 0 Sub OSC disable (default)
- 1 Sub OSC Enable

**XINENA** External Main Oscillator Enable Bit
- 0 Main OSC disable (default)
- 1 Main OSC Enable

**RSTDIS** External RESETB disable Bit
- 0 External RESET enable
- 1 External RESET disable

**LOCKB** Boot Area Hard LOCK(protection) Bit
- 0 Hard LOCK Disable
- 1 Hard LOCK Enable

**LOCKE** DATA memory LOCK bit
- 0 LOCK Disable
- 1 LOCK Enable

**LOCKF** CODE memory LOCK bit
- 0 LOCK Disable
- 1 LOCK Enable

# 17. APPENDIX

## A. Instruction Table

Instructions are either 1, 2 or 3 bytes long as listed in the 'Bytes' column below.
Each instruction takes either 1, 2 or 4 machine cycles to execute as listed in the following table. 1 machine cycle comprises 2 system clock cycles.

| ARITHMETIC | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| ADD A,Rn | Add register to A | 1 | 1 | 28-2F |
| ADD A,dir | Add direct byte to A | 2 | 1 | 25 |
| ADD A,@Ri | Add indirect memory to A | 1 | 1 | 26-27 |
| ADD A,#data | Add immediate to A | 2 | 1 | 24 |
| ADDC A,Rn | Add register to A with carry | 1 | 1 | 38-3F |
| ADDC A,dir | Add direct byte to A with carry | 2 | 1 | 35 |
| ADDC A,@Ri | Add indirect memory to A with carry | 1 | 1 | 36-37 |
| ADDC A,#data | Add immediate to A with carry | 2 | 1 | 34 |
| SUBB A,Rn | Subtract register from A with borrow | 1 | 1 | 98-9F |
| SUBB A,dir | Subtract direct byte from A with borrow | 2 | 1 | 95 |
| SUBB A,@Ri | Subtract indirect memory from A with borrow | 1 | 1 | 96-97 |
| SUBB A,#data | Subtract immediate from A with borrow | 2 | 1 | 94 |
| INC A | Increment A | 1 | 1 | 04 |
| INC Rn | Increment register | 1 | 1 | 08-0F |
| INC dir | Increment direct byte | 2 | 1 | 05 |
| INC @Ri | Increment indirect memory | 1 | 1 | 06-07 |
| DEC A | Decrement A | 1 | 1 | 14 |
| DEC Rn | Decrement register | 1 | 1 | 18-1F |
| DEC dir | Decrement direct byte | 2 | 1 | 15 |
| DEC @Ri | Decrement indirect memory | 1 | 1 | 16-17 |
| INC DPTR | Increment data pointer | 1 | 2 | A3 |
| MUL AB | Multiply A by B | 1 | 4 | A4 |
| DIV AB | Divide A by B | 1 | 4 | 84 |
| DA A | Decimal Adjust A | 1 | 1 | D4 |

| LOGICAL | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| ANL A,Rn | AND register to A | 1 | 1 | 58-5F |
| ANL A,dir | AND direct byte to A | 2 | 1 | 55 |
| ANL A,@Ri | AND indirect memory to A | 1 | 1 | 56-57 |
| ANL A,#data | AND immediate to A | 2 | 1 | 54 |
| ANL dir,A | AND A to direct byte | 2 | 1 | 52 |
| ANL dir,#data | AND immediate to direct byte | 3 | 2 | 53 |
| ORL A,Rn | OR register to A | 1 | 1 | 48-4F |
| ORL A,dir | OR direct byte to A | 2 | 1 | 45 |
| ORL A,@Ri | OR indirect memory to A | 1 | 1 | 46-47 |
| ORL A,#data | OR immediate to A | 2 | 1 | 44 |
| ORL dir,A | OR A to direct byte | 2 | 1 | 42 |
| ORL dir,#data | OR immediate to direct byte | 3 | 2 | 43 |
| XRL A,Rn | Exclusive-OR register to A | 1 | 1 | 68-6F |
| XRL A,dir | Exclusive-OR direct byte to A | 2 | 1 | 65 |

| | | | | |
|---|---|---|---|---|
| XRL A, @Ri | Exclusive-OR indirect memory to A | 1 | 1 | 66-67 |
| XRL A,#data | Exclusive-OR immediate to A | 2 | 1 | 64 |
| XRL dir,A | Exclusive-OR A to direct byte | 2 | 1 | 62 |
| XRL dir,#data | Exclusive-OR immediate to direct byte | 3 | 2 | 63 |
| CLR A | Clear A | 1 | 1 | E4 |
| CPL A | Complement A | 1 | 1 | F4 |
| SWAP A | Swap Nibbles of A | 1 | 1 | C4 |
| RL A | Rotate A left | 1 | 1 | 23 |
| RLC A | Rotate A left through carry | 1 | 1 | 33 |
| RR A | Rotate A right | 1 | 1 | 03 |
| RRC A | Rotate A right through carry | 1 | 1 | 13 |

| DATA TRANSFER | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| MOV A,Rn | Move register to A | 1 | 1 | E8-EF |
| MOV A,dir | Move direct byte to A | 2 | 1 | E5 |
| MOV A,@Ri | Move indirect memory to A | 1 | 1 | E6-E7 |
| MOV A,#data | Move immediate to A | 2 | 1 | 74 |
| MOV Rn,A | Move A to register | 1 | 1 | F8-FF |
| MOV Rn,dir | Move direct byte to register | 2 | 2 | A8-AF |
| MOV Rn,#data | Move immediate to register | 2 | 1 | 78-7F |
| MOV dir,A | Move A to direct byte | 2 | 1 | F5 |
| MOV dir,Rn | Move register to direct byte | 2 | 2 | 88-8F |
| MOV dir,dir | Move direct byte to direct byte | 3 | 2 | 85 |
| MOV dir,@Ri | Move indirect memory to direct byte | 2 | 2 | 86-87 |
| MOV dir,#data | Move immediate to direct byte | 3 | 2 | 75 |
| MOV @Ri,A | Move A to indirect memory | 1 | 1 | F6-F7 |
| MOV @Ri,dir | Move direct byte to indirect memory | 2 | 2 | A6-A7 |
| MOV @Ri,#data | Move immediate to indirect memory | 2 | 1 | 76-77 |
| MOV DPTR,#data | Move immediate to data pointer | 3 | 2 | 90 |
| MOVC A,@A+DPTR | Move code byte relative DPTR to A | 1 | 2 | 93 |
| MOVC A,@A+PC | Move code byte relative PC to A | 1 | 2 | 83 |
| MOVX A,@Ri | Move external data(A8) to A | 1 | 2 | E2-E3 |
| MOVX A,@DPTR | Move external data(A16) to A | 1 | 2 | E0 |
| MOVX @Ri,A | Move A to external data(A8) | 1 | 2 | F2-F3 |
| MOVX @DPTR,A | Move A to external data(A16) | 1 | 2 | F0 |
| PUSH dir | Push direct byte onto stack | 2 | 2 | C0 |
| POP dir | Pop direct byte from stack | 2 | 2 | D0 |
| XCH A,Rn | Exchange A and register | 1 | 1 | C8-CF |
| XCH A,dir | Exchange A and direct byte | 2 | 1 | C5 |
| XCH A,@Ri | Exchange A and indirect memory | 1 | 1 | C6-C7 |
| XCHD A,@Ri | Exchange A and indirect memory nibble | 1 | 1 | D6-D7 |

| BOOLEAN | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| CLR C | Clear carry | 1 | 1 | C3 |
| CLR bit | Clear direct bit | 2 | 1 | C2 |
| SETB C | Set carry | 1 | 1 | D3 |
| SETB bit | Set direct bit | 2 | 1 | D2 |
| CPL C | Complement carry | 1 | 1 | B3 |
| CPL bit | Complement direct bit | 2 | 1 | B2 |
| ANL C,bit | AND direct bit to carry | 2 | 2 | 82 |

| | | | | |
|---|---|---|---|---|
| ANL C,/bit | AND direct bit inverse to carry | 2 | 2 | B0 |
| ORL C,bit | OR direct bit to carry | 2 | 2 | 72 |
| ORL C,/bit | OR direct bit inverse to carry | 2 | 2 | A0 |
| MOV C,bit | Move direct bit to carry | 2 | 1 | A2 |
| MOV bit,C | Move carry to direct bit | 2 | 2 | 92 |

| BRANCHING | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| ACALL addr 11 | Absolute jump to subroutine | 2 | 2 | 11→F1 |
| LCALL addr 16 | Long jump to subroutine | 3 | 2 | 12 |
| RET | Return from subroutine | 1 | 2 | 22 |
| RETI | Return from interrupt | 1 | 2 | 32 |
| AJMP addr 11 | Absolute jump unconditional | 2 | 2 | 01→E1 |
| LJMP addr 16 | Long jump unconditional | 3 | 2 | 02 |
| SJMP rel | Short jump (relative address) | 2 | 2 | 80 |
| JC rel | Jump on carry = 1 | 2 | 2 | 40 |
| JNC rel | Jump on carry = 0 | 2 | 2 | 50 |
| JB bit,rel | Jump on direct bit = 1 | 3 | 2 | 20 |
| JNB bit,rel | Jump on direct bit = 0 | 3 | 2 | 30 |
| JBC bit,rel | Jump on direct bit = 1 and clear | 3 | 2 | 10 |
| JMP @A+DPTR | Jump indirect relative DPTR | 1 | 2 | 73 |
| JZ rel | Jump on accumulator = 0 | 2 | 2 | 60 |
| JNZ rel | Jump on accumulator ≠ 0 | 2 | 2 | 70 |
| CJNE A,dir,rel | Compare A,direct jne relative | 3 | 2 | B5 |
| CJNE A,#d,rel | Compare A,immediate jne relative | 3 | 2 | B4 |
| CJNE Rn,#d,rel | Compare register, immediate jne relative | 3 | 2 | B8-BF |
| CJNE @Ri,#d,rel | Compare indirect, immediate jne relative | 3 | 2 | B6-B7 |
| DJNZ Rn,rel | Decrement register, jnz relative | 3 | 2 | D8-DF |
| DJNZ dir,rel | Decrement direct byte, jnz relative | 3 | 2 | D5 |

| MISCELLANEOUS | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| NOP | No operation | 1 | 1 | 00 |

| ADDITIONAL INSTRUCTIONS (selected through EO[7:4]) | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| MOVC @(DPTR++),A | M8051W/M8051EW-specific instruction supporting software download into program memory | 1 | 2 | A5 |
| TRAP | Software break command | 1 | 1 | A5 |

In the above table, an entry such as E8-EF indicates a continuous block of hex opcodes used for 8 different registers, the register numbers of which are defined by the lowest three bits of the corresponding code. Non-continuous blocks of codes, shown as 11→F1 (for example), are used for absolute jumps and calls, with the top 3 bits of the code being used to store the top three bits of the destination address.

The CJNE instructions use the abbreviation #d for immediate data; other instructions use #data.s

### B. Instructions on how to use the input port.

- Error occur status

- Using compare jump instructions with input port, it could cause error due to the timing conflict inside the MCU.
- Compare jump Instructions which cause potential error used with input port condition:

```
JB      bit, rel   ; jump on direct bit=1
JNB     bit, rel   ; jump on direct bit=0
JBC     bit, rel   ; jump on direct bit=1 and clear
CJNE    A, dir, rel        ; compare A, direct jne relative
DJNZ    dir, rel   ; decrement direct byte, jnz relative
```

- It is only related with Input port. Internal parameters, SFRs and output bit ports don't cause an y error by using compare jump instructions.
- If input signal is fixed, there is no error in using compare jump instructions.

- Error status example

```
while(1){
  if (P00==1){ P10=1; }
  else { P10=0; }
  P11^=1;
}
```

```
zzz:   JNB     080.0, xxx  ; it possible to be error
       SETB    088.0
       SJMP    yyy
xxx:   CLR     088.0
yyy:   MOV     C,088.1
       CPL     C
       MOV     088.1,C
       SJMP    zzz
```

```
unsigned char ret_bit_err(void)
{
  return !P00;
}
```

```
       MOV     R7, #000
       JB      080.0, xxx  ; it possible to be error
       MOV     R7, #001
xxx:   RET
```

- Preventative measures (2 cases)
  - Do not use input bit port for bit operation but for byte operation. Using byte operation instead of bit operation will not cause any error in using compare jump instructions for input port.

```
while(1){
 if ((P0&0x01)==0x01){ P10=1; }
  else { P10=0; }
   P11^=1;
}
```

```
zzz:   MOV     A, 080      ; read as byte
       JNB     0E0.0, xxx  ; compare
       SETB    088.0
       SJMP    yyy
xxx:   CLR     088.0
yyy:   MOV     C,088.1
       CPL     C
       MOV     088.1,C
       SJMP    zzz
```

```
bit tt;
while(1){
  tt=P00;
  if (tt==0){ P10=1;}
  else {  P10=0;}
  P11^=1;
}
```

```
zzz:  MOV    C,080.0  ; input port use internal parameter
      MOV    020.0, C      ; move
      JB     020.0, xxx    ; compare
      SETB   088.0
      SJMP   yyy
xxx:  CLR    088.0
yyy:  MOV    C,088.1
      CPL    C
      MOV    088.1,C
      SJMP   zzz
```

- If you use input bit port for compare jump instruction, you have to copy the input port as intern al parameter or carry bit and then use compare jump instruction.

# Customer Support

To share comments, get your technical questions answered, or report issues you may be experiencing with our products, please visit Zilog's Technical Support page at http://support.zilog.com.

To learn more about this product, find additional documentation, or to discover other facets about Zilog product offerings, please visit the Zilog Knowledge Base or consider participating in the Zilog Forum.

This publication is subject to replacement by a later edition. To determine whether a later edition exists, please visit the Zilog website at http://www.zilog.com.

LifeElectronics
Живое партнерство

**ООО «ЛайфЭлектроникс»** **“LifeElectronics” LLC**

*ИНН 7805602321 КПП 780501001 Р/С 40702810122510004610 ФАКБ "АБСОЛЮТ БАНК" (ЗАО) в г.Санкт-Петербурге К/С 30101810900000000703 БИК 044030703*

*Компания «Life Electronics» занимается поставками электронных компонентов импортного и отечественного производства от производителей и со складов крупных дистрибьюторов Европы, Америки и Азии.*

*С конца 2013 года компания активно расширяет линейку поставок компонентов по направлению коаксиальный кабель, кварцевые генераторы и конденсаторы (керамические, пленочные, электролитические), за счёт заключения дистрибьюторских договоров*

*Мы предлагаем:*

- *Конкурентоспособные цены и скидки постоянным клиентам.*
- *Специальные условия для постоянных клиентов.*
- *Подбор аналогов.*
- *Поставку компонентов в любых объемах, удовлетворяющих вашим потребностям.*

- *Приемлемые сроки поставки, возможна ускоренная поставка.*
- *Доставку товара в любую точку России и стран СНГ.*
- *Комплексную поставку.*
- *Работу по проектам и поставку образцов.*
- *Формирование склада под заказчика.*

- *Сертификаты соответствия на поставляемую продукцию (по желанию клиента).*
- *Тестирование поставляемой продукции.*
- *Поставку компонентов, требующих военную и космическую приемку.*
- *Входной контроль качества.*
- *Наличие сертификата ISO.*

*В составе нашей компании организован Конструкторский отдел, призванный помогать разработчикам, и инженерам.*

*Конструкторский отдел помогает осуществить:*

- *Регистрацию проекта у производителя компонентов.*
- *Техническую поддержку проекта.*
- *Защиту от снятия компонента с производства.*
- *Оценку стоимости проекта по компонентам.*
- *Изготовление тестовой платы монтаж и пусконаладочные работы.*

*Тел: +7 (812) 336 43 04 (многоканальный)*
*Email: org@lifeelectronics.ru*

***www.lifeelectronics.ru***