

ABOV SEMICONDUCTOR Co. Ltd.  
8-BIT SINGLE-CHIP MICROCONTROLLERS

# MC81F8816/8616

*User's Manual (Ver. 1.03)*

---



---

**Version 1.03**

**Published by  
FAE Team**

**©2008 ABOV Semiconductor Co., Ltd. All rights reserved.**

---

Additional information of this manual may be served by ABOV Semiconductor offices in Korea or Distributors.

ABOV Semiconductor reserves the right to make changes to any information here in at any time without notice.

The information, diagrams and other data in this manual are correct and reliable; however, ABOV Semiconductor is in no way responsible for any violations of patents or other rights of the third party generated by the use of this manual.

**REVISION HISTORY****VERSION 1.03 (December 3, 2012) This Book**

ABOV logo is renewed on this book.

Single and Gang writer are added in "1.3 Development Tools" on page 3.

VDD voltage for sub-active mode is changed to 3.0~5.5V in "7.2 Recommended Operating Conditions" on page 21.

The notice of STOP mode is added in "23.2 STOP Mode" on page 118.

Notice: If the STOP mode is used in the program, BOD function should be disabled in the initial routine of software.

Block diagram of BOD is updated in "Figure 27-1 Block Diagram of BOD (Brown-out Detector Reset)" on page 127.

**VERSION 1.02 (February 11, 2010)**

The caution for the  $\overline{\text{ALE}}$  pin at ISP mode is added in "31.3 Hardware Conditions to Enter the ISP Mode" on page 136.

**VERSION 1.01 (January 11, 2010)**

The block diagram of LCD Bias is modified in Figure 18-3 LCD Bias Control

The figures of flash writer were updated in "1. OVERVIEW" on page 1.

Config Read Voltage ( $V_{\text{CONFIG}}$ ), maximum VDD Start Voltage ( $V_{\text{START}}$ ) and description were added in "7.3 DC Electrical Characteristics" on page 22.

In case AVREF voltage was less than VDD voltage for ADC, the table-note and note were added in "7.5 A/D Converter



# Table of Contents

<b>1. OVERVIEW</b> .....	<b>1</b>	<b>16. BUZZER OUTPUT FUNCTION</b> .....	<b>79</b>
Description .....	1	<b>17. INTERRUPTS</b> .....	<b>81</b>
Features .....	1	Interrupt Sequence .....	84
Development Tools .....	3	BRK Interrupt .....	85
Ordering Information .....	5	Multi Interrupt .....	85
<b>2. BLOCK DIAGRAM</b> .....	<b>6</b>	External Interrupt .....	87
MC81F8816Q (80 pin package) .....	6	<b>18. LCD DRIVER</b> .....	<b>88</b>
MC81F8616Q (64 pin package) .....	7	Control of LCD Driver Circuit .....	89
<b>3. PIN ASSIGNMENT</b> .....	<b>8</b>	LCD BIAS Control .....	92
<b>4. PACKAGE DIAGRAM</b> .....	<b>10</b>	LCD Display Memory .....	94
<b>5. PIN FUNCTION</b> .....	<b>12</b>	Control Method of LCD Driver .....	95
<b>6. PORT STRUCTURES</b> .....	<b>16</b>	Duty and Bias Selection of LCD Driver .....	97
<b>7. ELECTRICAL CHARACTERISTICS</b> .....	<b>21</b>	<b>19. SERIAL PERIPHERAL INTERFACE (SPI)</b> ...	<b>98</b>
Absolute Maximum Ratings .....	21	Transmission/Receiving Timing .....	99
Recommended Operating Conditions .....	21	The usage of Serial I/O .....	100
DC Electrical Characteristics .....	22	The Method to Test Correct Transmission ....	101
LCD Characteristics .....	23	<b>20. INTER IC COMMUNICATION (I2C)</b> .....	<b>102</b>
A/D Converter Characteristics .....	23	Bit Transfer .....	104
AC Characteristics .....	25	Start/Stop Conditions .....	104
Serial I/O Characteristics .....	27	Data Transfer .....	105
Typical Characteristics .....	28	Acknowledge .....	106
<b>8. MEMORY ORGANIZATION</b> .....	<b>32</b>	Synchronization/Arbitration .....	107
Registers .....	32	<b>21. UNIVERSAL ASYNCHRONOUS SERIAL IN-</b>	<b>110</b>
Program Memory .....	35	<b>TERFACE (UART)</b> .....	<b>110</b>
Data Memory .....	38	Asynchronous Serial Interface Configuration	111
Addressing Mode .....	42	Relationship between main clock and baud rate .	114
<b>9. I/O PORTS</b> .....	<b>46</b>	<b>22. OPERATION MODE</b> .....	<b>115</b>
Registers for Ports .....	46	Operation Mode Switching .....	116
I/O Ports Configuration .....	47	<b>23. POWER DOWN OPERATION</b> .....	<b>117</b>
<b>10. CLOCK GENERATOR</b> .....	<b>51</b>	SLEEP Mode .....	117
<b>11. BASIC INTERVAL TIMER</b> .....	<b>53</b>	STOP Mode .....	118
<b>12. TIMER / COUNTER</b> .....	<b>55</b>	<b>24. OSCILLATOR CIRCUIT</b> .....	<b>122</b>
8-Bit Timer/Counter Mode .....	59	<b>25. PLL</b> .....	<b>123</b>
16 Bit Timer/Counter Mode .....	61	External PLL Circuit .....	124
8-Bit Capture Mode .....	63	<b>26. RESET</b> .....	<b>125</b>
16-bit Capture Mode .....	67	External Reset Input .....	125
8-Bit (16-Bit) Compare Output Mode .....	68	Power On Reset .....	126
PWM Mode .....	68	Brown-out Detector .....	126
<b>13. WATCH TIMER</b> .....	<b>72</b>	Watchdog Timer Reset .....	126
<b>14. WATCH DOG TIMER</b> .....	<b>74</b>	<b>27. Brown-out Detector (BOD)</b> .....	<b>127</b>
<b>15. ANALOG TO DIGITAL CONVERTER</b> .....	<b>76</b>		

<b>28. Oscillation Noise Protector .....</b>	<b>129</b>	Hardware Conditions to Enter the ISP Mode	136
<b>29. FLASH PROGRAMMING SPEC. ....</b>	<b>131</b>	Sequence to enter ISP mode/user mode .....	137
FLASH Configuration Byte .....	131	USB-SIO-ISP Board .....	138
FLASH Programming .....	131	<b>A. INSTRUCTION .....</b>	<b>ii</b>
<b>30. EMULATOR EVA. BOARD SETTING .....</b>	<b>133</b>	Terminology List .....	ii
<b>31. IN-SYSTEM PROGRAMMING .....</b>	<b>134</b>	Instruction Map .....	iii
Getting Started / Installation .....	134	Instruction Set .....	v
Basic ISP S/W Information .....	135	<b>B. MASK ORDER SHEET(MC81C8816) .....</b>	<b>xiii</b>
		<b>C. MASK ORDER SHEET(MC81C8616) .....</b>	<b>xiv</b>

# MC81F8816/8616

## CMOS SINGLE-CHIP 8-BIT MICROCONTROLLER WITH LCD CONTROLLER/DRIVER

### 1. OVERVIEW

#### 1.1 Description

The MC81F8816/8616 are an advanced CMOS 8-bit microcontroller with 16K bytes of FLASH ROM(MTP). This device is one of the MC800 family and a powerful microcontroller which provides a high flexibility and cost effective solution to many LCD applications. The MC81F8816/8616 provide the following standard features: 16K bytes of FLASH ROM, 512 bytes of RAM, 40 bytes of segment LCD display RAM, 8/16-bit timer/counter, 10-bit A/D converter, 7-bit watch dog timer, 21-bit watch timer with 7-bit auto reload counter, I2C, SPI, 8-bit UART, PLL, on-chip oscillator and clock circuitry. In addition, this device supports power saving modes to reduce power consumption. So the MC81F8816/8616 is the best controller solution in system which uses charatered LCD display and ADC.

FLASH MCU	MASK MCU	Memory (Bytes)		ADC	PWM	UART/ SPI/ I2C	I/O	LCD	Package
		ROM	RAM						
MC81F8816	MC81C8816	16K	512	8ch.	2ch.	1ch/ 1ch/ 1ch	56	36SEG x 8COM (40SEG x 4COM)	80MQFP
MC81F8616	MC81C8616	16K	512	5ch.	2ch.	1ch/ 1ch/ 1ch	48	28SEG x 8COM (32SEG x 4COM)	64MQFP 64LQFP

#### 1.2 Features

- **16K Bytes On-chip FLASH ROM (ISP)**
- **FLASH Memory**
  - Endurance : 1000 cycles
  - Data Retention : 10 years
- **512 Bytes On-chip Data RAM**
- **40 bytes Display RAM**
- **32 MHz PLL Oscillator**
- **Instruction Cycle Time**
  - 167ns at 12MHz (NOP instruction)
- **LCD display/controller**
  - 1/4 Duty Mode (40Seg x 4Com, 1/3 Bias)
  - 1/8 Duty Mode (36Seg x 8Com, 1/4 Bias)
- **Four 8-bit Timer/Counter**  
(They can be used as two 16-bit Timer/Counter)
- **One 7-bit Watch Dog Timer**
- **One 21-bit Watch Timer**
  - 1 minute interrupt available
- **One 8-bit Basic Interval Timer**
- **One 6-bit Buzzer Driving Port**
- **Dual Clock Operation**
  - Main Clock : 400kHz ~ 12MHz
  - Sub Clock : 32.768kHz
- **Main Clock Oscillation**
  - Crystal
  - Ceramic Resonator
  - Internal Oscillation : 8MHz/4MHz
- **Operating Temperature : -40~85 °C**
- **Built-in Noise Immunity Circuit**
  - Noise Filter
  - BOD(Brown-out Detector)
- **Power Down Mode**

- Main Clock : STOP, SLEEP, SUB-Active mode
- **400kHz to 12MHz Wide Operating Frequency**
- **On-Chip POR (Power On Reset) and BOR(Brown Out Reset)**
- **Internal Resistor for LCD Bias**
- **56/48 Programmable I/O Pins**

MC81F8816	I/O: 31 I : 1 I/O with SEG/COM:24
MC81F8616 MC81C8616	I/O: 23 I : 1 I/O with SEG/COM:24

- **8/5-channel 10-bit On-chip A/D Converter**

MC81F8816	8-channel ADC
MC81F8616 MC81C8616	5-channel ADC

- **Two 10-bit High Speed PWM Output**
- **16 Interrupt sources**
  - External Interrupt : 4
  - Timer : 4
  - UART : 2
  - I2C, SPI, ADC, WDT, WT, BIT
- **One Universal Asynchronous Receiver/Transmitter (UART)/ One Serial Peripheral Interface(SPI)/ One Inter IC Communication(I2C)**
- **Wide Operating Voltage & Frequency Range**
  - 2.2 ~ 5.5V @ (4.2Mhz)
  - 4.5 ~ 5.5V @ (12Mhz)
- **80MQFP, 64MQFP, 64LQFP Package Types**
  - Available Pb free package

MC81F8816	80MQFP
MC81F8616 MC81C8616	64MQFP, 64LQFP



### 1.3 Development Tools

The MC81F8816/8616 are supported by a full-featured macro assembler, an in-circuit emulator CHOICE-Dr.<sup>TM</sup> and OTP/FLASH programmers. There are two different type of programmers such as single type and gang type. For mode detail, Macro assembler operates under the MS-Windows 95 and upversioned Windows OS. And HMS800C compiler only operates under the MS-Windows 2000 and upversioned Windows OS.

Please contact sales part of ABOV semiconductor.

Software	<ul style="list-style-type: none"> <li>- MS-Windows based assembler</li> <li>- MS-Windows based Debugger</li> <li>- MC800 C compiler</li> </ul>
Hardware (Emulator)	<ul style="list-style-type: none"> <li>- CHOICE-Dr.</li> <li>- CHOICE-Dr. EVA81F88 B/D Rev2.0</li> </ul>
POD Name	<ul style="list-style-type: none"> <li>- POD80C73D-80MQ</li> <li>- POD80C74D-64MQ</li> </ul>
FLASH Writer	<ul style="list-style-type: none"> <li>- PGM Plus USB (Single writer)</li> <li>- Stand Alone PGM Plus(Single writer)</li> <li>- Standalone GANG4/8 USB (Gang writer)</li> <li>- USB-SIO-ISP Board</li> </ul>

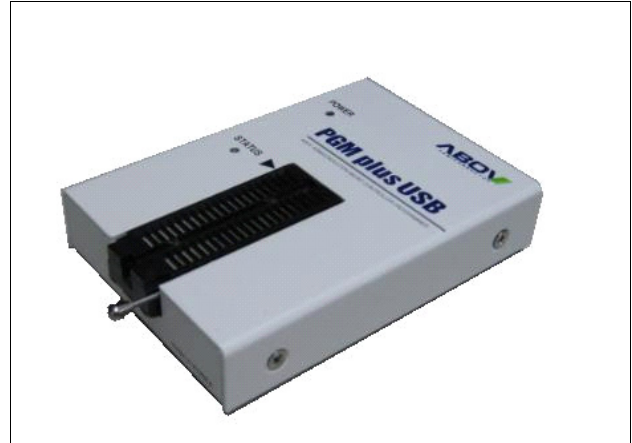


Figure 1-2 PGMplus USB ( Single Writer )



Figure 1-1 Choice-Dr. (Emulator)



Figure 1-3 Stand Alone PGM\_Plus(ISP)



Figure 1-4 Standalone Gang4 USB (Gang Writer)



Figure 1-6 USB-SIO-ISP Board



Figure 1-5 Standalone Gang8 (Gang Writer)

### 1.4 Ordering Information

	Device name	ROM Size	RAM size	Package
FLASH version	MC81F8816Q MC81F8616Q MC81F8616L	16K bytes	512 bytes	80MQFP 64MQFP 64LQFP
MASK version	MC81C8616Q MC81C8616L	16K bytes	512 bytes	64MQFP 64LQFP

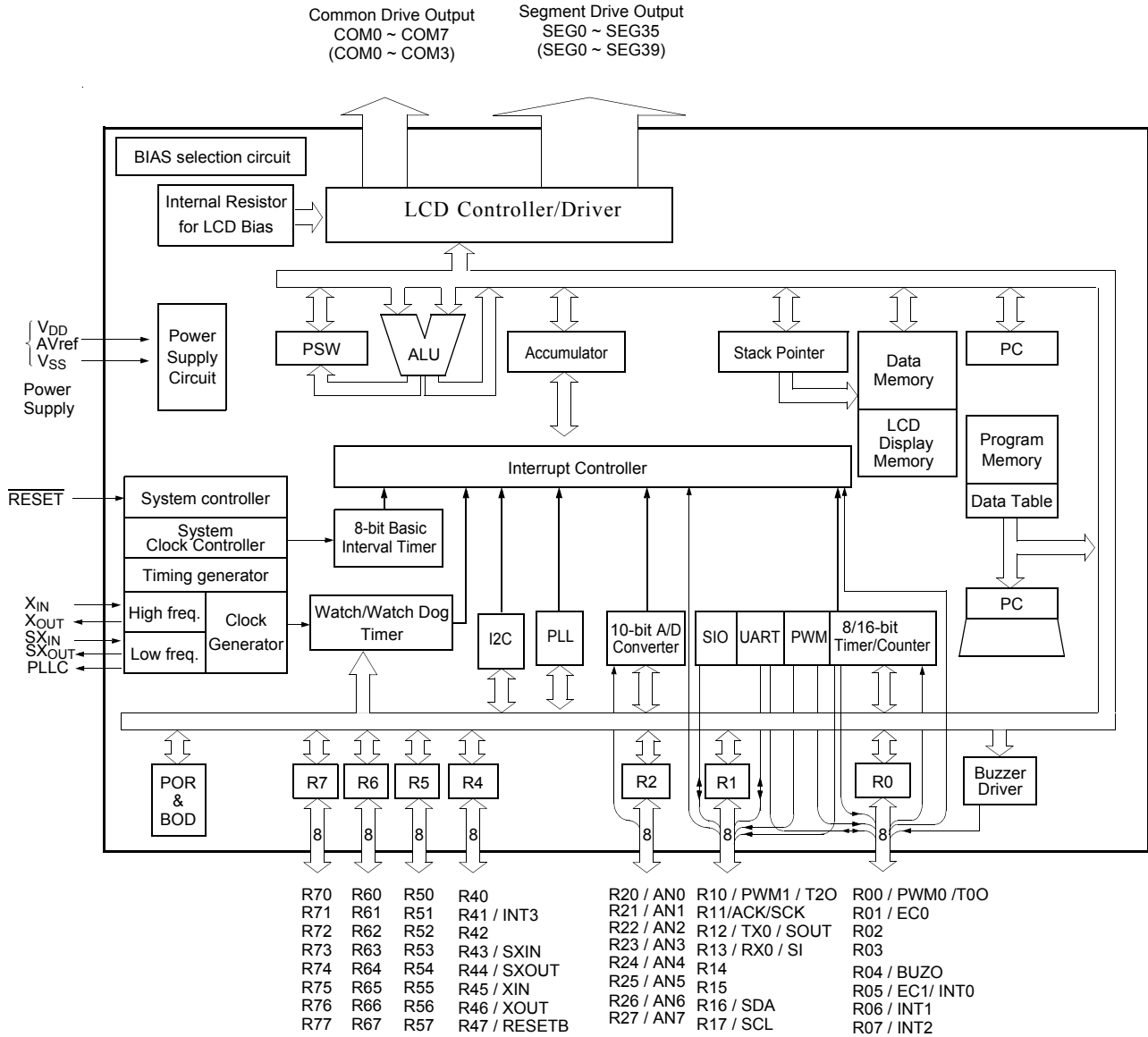
- Pb free package;

The “P” suffix will be added at original part number.

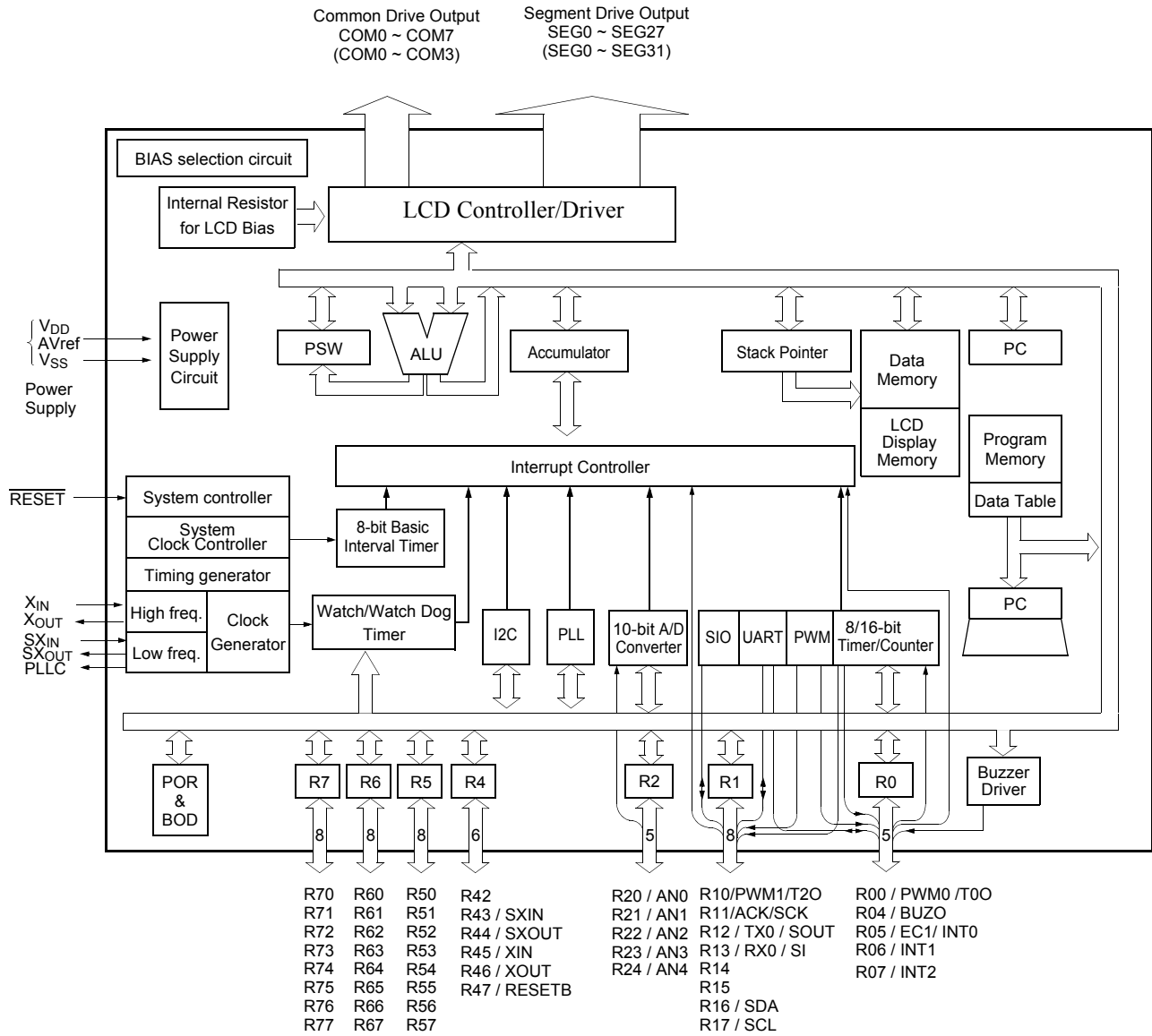
For example; MC81F8816Q(Normal package), MC81F8816Q P(Pb free package)

## 2. BLOCK DIAGRAM

### 2.1 MC81F8816Q (80 pin package)

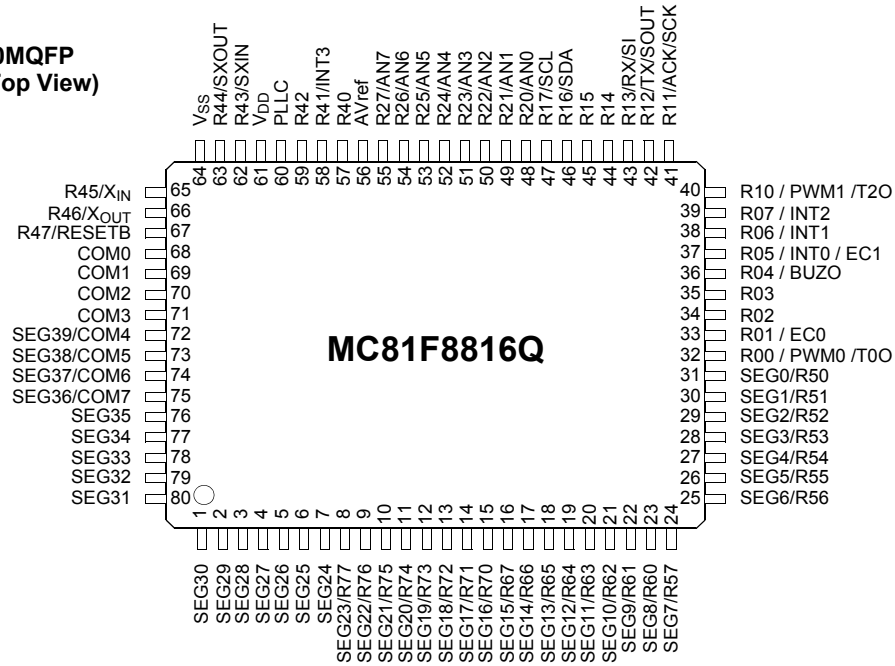


**2.2 MC81F8616Q (64 pin package)**

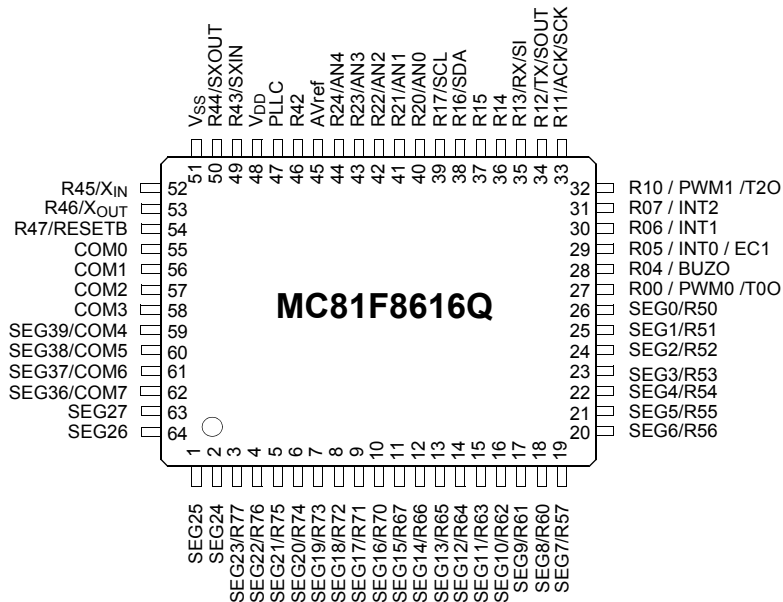


### 3. PIN ASSIGNMENT

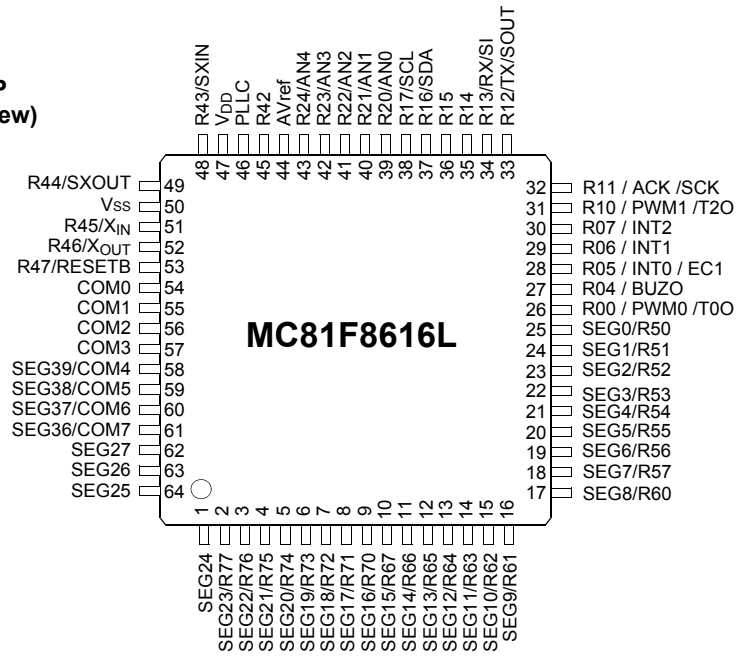
**80MQFP  
(Top View)**



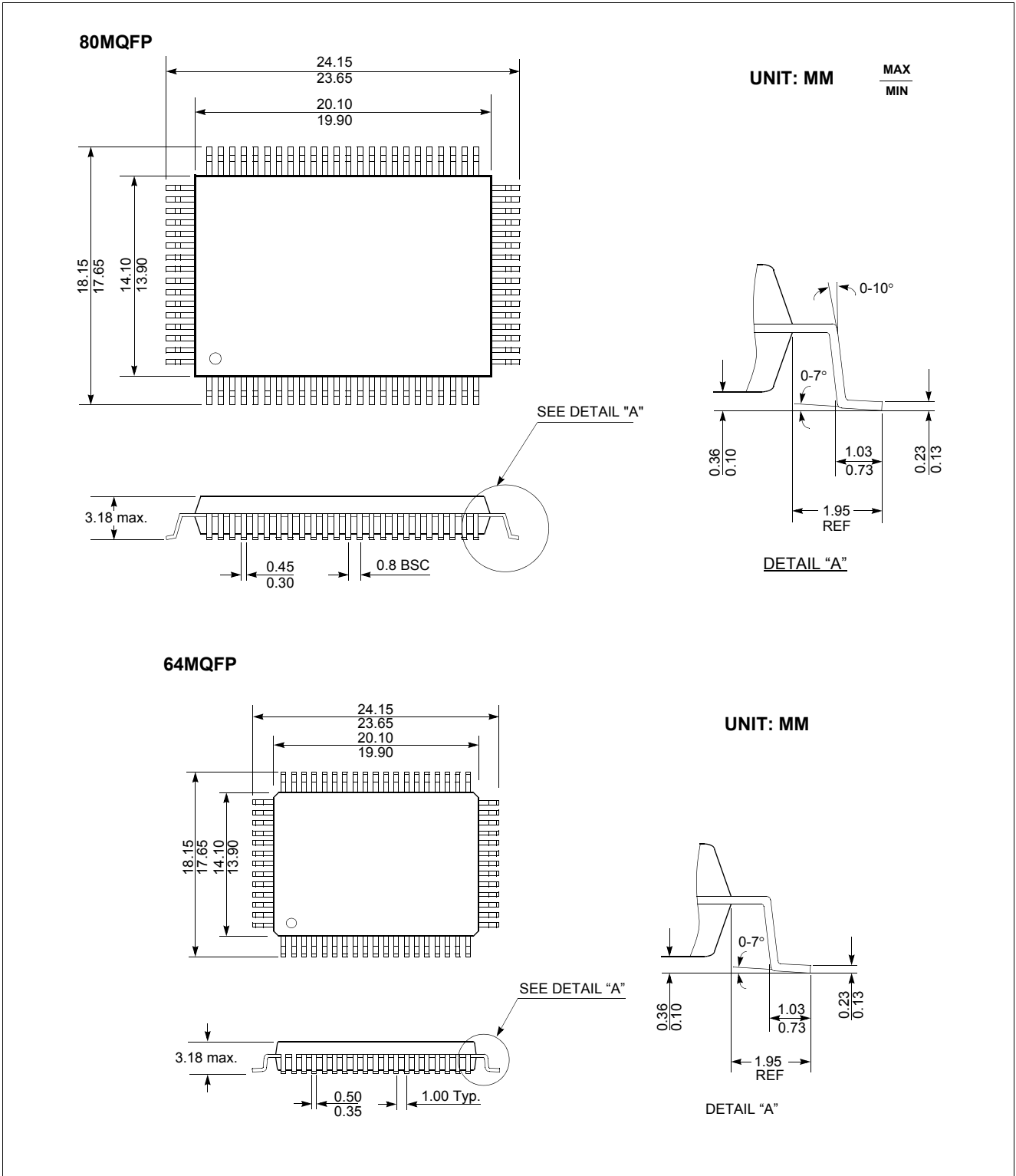
**64MQFP  
(Top View)**



**64LQFP  
(Top View)**

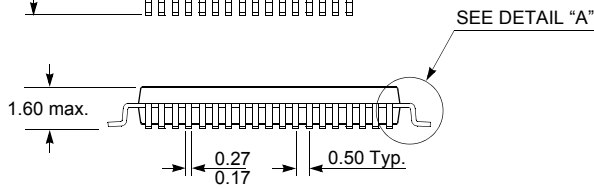
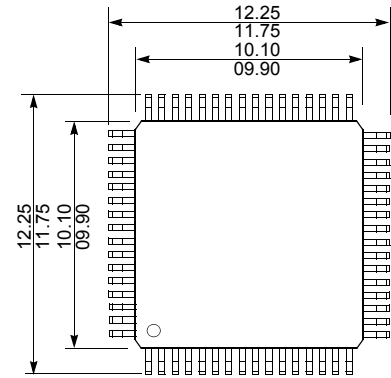


4. PACKAGE DIAGRAM

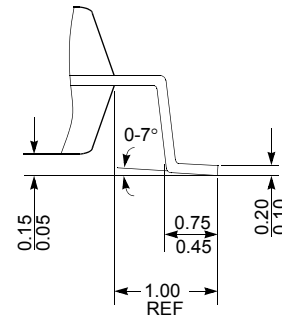




**64LQFP**



UNIT: MM



DETAIL "A"

## 5. PIN FUNCTION

**V<sub>DD</sub>**: Supply Voltage.

**V<sub>SS</sub>**: Circuit ground.

**RESET**: Reset the MCU Reset.

**X<sub>IN</sub>**: Input to the inverting oscillator amplifier and input to the internal main clock operating circuit.

**X<sub>OUT</sub>**: Output from the inverting oscillator amplifier.

**SX<sub>IN</sub>**: Input to the internal sub system clock operating circuit.

**SX<sub>OUT</sub>**: Output from the inverting subsystem oscillator amplifier.

**SEG0~SEG39**: Segment signal output pins for the LCD display. See "18. LCD DRIVER" on page 88 for details. Also SEG0~SEG23 are shared with normal I/O ports and SEG24~35 are only segment output port(SEG24~31 are shared with normal I/O port at EVA chip) and SEG36~39 are multiplexed with COM7~COM4.

**Note:** SEG28 ~ SEG35 are not supported in MC81F8616Q(64pin).

**COM0~COM7**: Common signal output pins for the LCD display. See "18. LCD DRIVER" on page 88 for details. Also COM0~COM3 are only common output ports and COM4~COM7 are multiplexed with SEG36~SEG39.

COM4~COM7 and SEG36~SEG39 are selected by LCDD0 of the LCR register.

LCDD0	COM4~COM7 / SEG39~SEG36
0	COM4 ~ COM7
1	SEG39 ~ SEG36

**R00~R07**: R0 is a 8-bit CMOS bidirectional I/O port(5-bit I/O port at MC81F8616Q). R0 pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. Also, pull-up resistors and open-drain outputs can be assigned by software.

In addition, R0 serves the functions of the various follow-

ing special features.

Port pin	Alternate function
R00	PWM0/T00 (Timer1 PWM Output / Timer0 Output)
R01	EC0 (Timer 0 Event Count Input)
R04	BUZO (Buzzer Output)
R05	EC1 / INT0 (Timer2 Event Count Input/External Interrupt 0 Request Input)
R06	INT1 (External Interrupt 1 Request Input)
R07	INT2 (External Interrupt 2 Request Input)

**Note:** R01/EC0~R03 are not supported in MC81F8616Q(64pin).

**R10~R17**: R1 is an 8-bit CMOS bidirectional I/O port. R1 pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs or schmitt trigger inputs. Also, pull-up resistors and open-drain outputs can be assigned by software.

In addition, R1 serves the function of the following special feature.

Port pin	Alternate function
R10	PWM1/T2O(Timer3 PWM / Timer2 Output)
R11	ACK/SCK
R12	TX/SOUT
R13	RX/SI
R14	-
R15	-
R16	SDA
R17	SCL

**R20~R27**: R2 is a 5/8-bit CMOS bidirectional I/O port(5-bit I/O port at MC81F8616Q). Each pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. Also, pull-up resistors and open-drain outputs can be assigned by software.

In addition, R2 serves the functions of the various follow-

ing special features.

Port pin	Alternate function
R20	AN0 (Analog Input Port0)
R21	AN1 (Analog Input Port1)
R22	AN2 (Analog Input Port2)
R23	AN3 (Analog Input Port3)
R24	AN4 (Analog Input Port4)
R25	AN5 (Analog Input Port5)
R26	AN6 (Analog Input Port6)
R27	AN7 (Analog Input Port7)

**Note:** R25/AN5 ~ R27/AN7 are not supported in MC81F8616Q(64pin).

**R40~R47:** R4 is a 6/8-bit CMOS bidirectional I/O port(6-bit I/O port at MC81F8616Q). Each pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. Also, pull-up resistors and open-drain outputs can be assigned by software.

In addition, R4 serves the functions of the various following special features.

Port pin	Alternate function
R40	-
R41	INT3 (External Interrupt 3 Request input)
R42	-
R43	SX <sub>IN</sub>
R44	SX <sub>OUT</sub>
R45	X <sub>IN</sub>
R46	X <sub>OUT</sub>
R47	RESET

**Note:** R40 ~ R41 are not supported in MC81F8616Q(64pin).

**R50~R57:** R5 is an 8-bit CMOS bidirectional I/O port or LCD segment output. Each pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. And each pins can also be set in segment output mode in 1-bit

units by R5PSR Register.

Port pin	Alternate function
R50	SEG0 (Segment Output 0)
R51	SEG1 (Segment Output 1)
R52	SEG2 (Segment Output 2)
R53	SEG3 (Segment Output 3)
R54	SEG4 (Segment Output 4)
R55	SEG5 (Segment Output 5)
R56	SEG6 (Segment Output 6)
R57	SEG7 (Segment Output 7)

**R60~R67:** R6 is an 8-bit CMOS bidirectional I/O port or LCD segment output. Each pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. And each pins can also be set in segment output mode in 1-bit units by R6PSR Register.

Port pin	Alternate function
R60	SEG8 (Segment Output 8)
R61	SEG9 (Segment Output 9)
R62	SEG10 (Segment Output 10)
R63	SEG11 (Segment Output 11)
R64	SEG12 (Segment Output 12)
R65	SEG13 (Segment Output 13)
R66	SEG14 (Segment Output 14)
R67	SEG15 (Segment Output 15)

**R70~R77:** R7 is a 8-bit CMOS input port or LCD segment output. Each pins can be set in digital input or segment output mode in 1-bit units by R7PSR Register.

Port pin	Alternate function
R70	SEG16 (Segment Output 16)
R71	SEG17 (Segment Output 17)
R72	SEG18 (Segment Output 18)
R73	SEG19 (Segment Output 19)
R74	SEG20 (Segment Output 20)
R75	SEG21 (Segment Output 21)
R76	SEG22 (Segment Output 22)
R77	SEG23 (Segment Output 23)

PIN NAME	Pin No.		Primary Function		Secondary Function		State @ Reset	State @ STOP
	MC81F8816Q	MC81F8616Q	I/O	Description	I/O	Description		
V <sub>DD</sub>	61	48	-	Supply Voltage	-	-	-	-
V <sub>SS</sub>	64	51	-	Circuit Ground	-	-	-	-
$\overline{\text{RESET}}$ / R47	67	54	I	General I/O port	I	Reset (low active)	'L' input	'H' input
X <sub>IN</sub> /R45, X <sub>OUT</sub> /R46	65,66	52,53	I,O	Main clock oscillator	-	-	Oscillation	'L', 'H'
SX <sub>IN</sub> /R43, SX <sub>OUT</sub> /R44	62,63	49,50	I,O	Sub clock oscillator	-	-	Oscillation	
R50/SEG0 ~ R77/SEG23	8~31	3~26	I/O	General I/O port	O	LCD segment output	Input port	State of before STOP
SEG24~SEG25	6,7	1,2	O	LCD segment output	-	-	-	
SEG26~SEG27	4,5	63,64	O	LCD segment output	-	-	-	
SEG28~SEG35	1~3,76~80	-	O	LCD segment output	-	-	-	
SEG36/COM7~ SEG39/COM4	72~75	62~59	O	LCD segment output	O	LCD common output	Output port	
COM3 ~ COM0	34~37	39~42	O	LCD common output	-	-	-	
AVref	56	45	I	Analog Power Voltage Input to A/D Converter	-	-	-	
PLLC	60	47	I	PLL input	-	-	-	
R40	57	-	I/O	General I/O port	-	-	Input port	
R41/INT3	58	-	I/O		I	Interrupt3 Input		
R42	49	46	I/O		-	-		

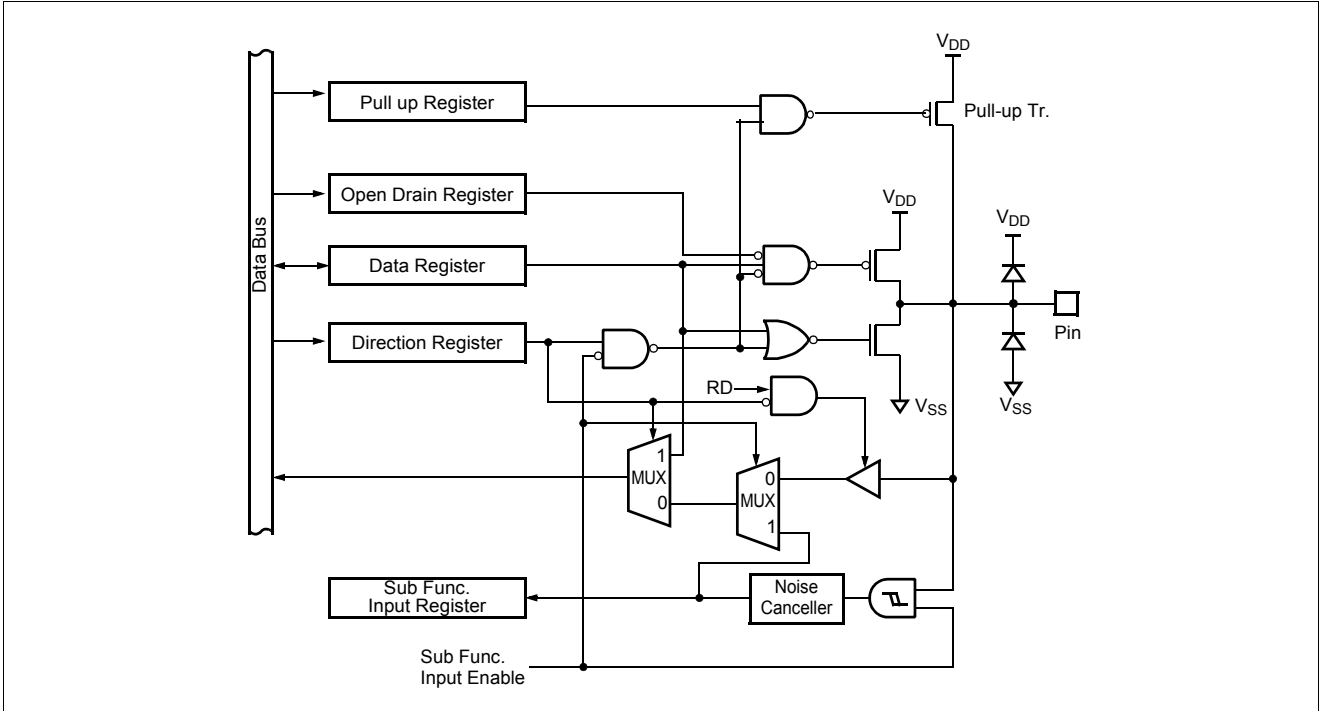
Table 5-1 Port Function Description

PIN NAME	Pin No.		Primary Function		Secondary Function		State @ Reset	State @ STOP
	MC81F8816Q	MC81F8616Q	I/O	Description	I/O	Description		
R00/PWM0/T00	32	27	I/O	General I/O port	O	Timer1 PWM Output Timer0 Output	Input port	State of before STOP
R01/EC0	33	-	I/O		I	Timer0 Event Counter Input		
R02	34	-	I/O		-	-		
R03	35	-	I/O		-	-		
R04/BUZO	36	28	I/O		O	Buzzer Output		
R05/EC1/INT0	37	29	I/O		I	Timer2 Event Counter Input Interrupt0 Input		
R06/INT1	38	30	I/O		I	Interrupt1 Input		
R07/INT2	39	31	I/O		I	Interrupt2 Input		
R10/PWM1/T20	40	32	I/O		O	Timer3 PWM Output Timer2 Output		
R11/ACK/SCK	41	33	I/O		I/O	Asynchronous Serial Interface Clock Input / SIO Serial Clock Input/Output		
R12/TX/SOUT	42	34	I/O		O	UART Serial Data Output / SIO Serial Data Output		
R13/RX/SI	43	35	I/O		I	UART Serial Data Input / SIO Serial Data Input		
R14	44	36	I/O		-	-		
R15	45	37	I/O		-	-		
R16/SDA	46	38	I/O		I/O	SIO Serial Data In/Out		
R17/SCL	47	39	I/O		I/O	I2C Serial Clock In/Out		
R20/AN0 ~ R24/AN4	48~52	40~44	I/O		I	A/D Converter Analog Input		
R25/AN5 ~ R27/AN7	53~55	-	I/O		I	A/D Converter Analog Input		

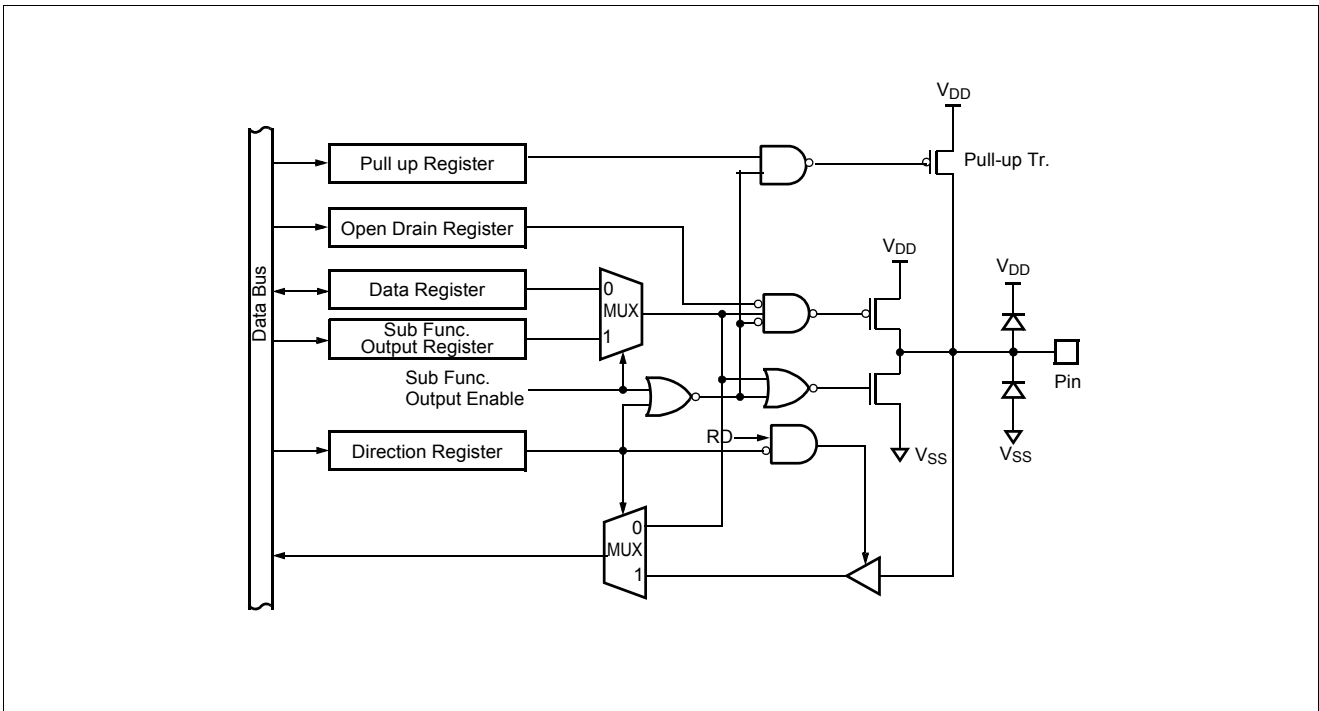
**Table 5-1 Port Function Description**

## 6. PORT STRUCTURES

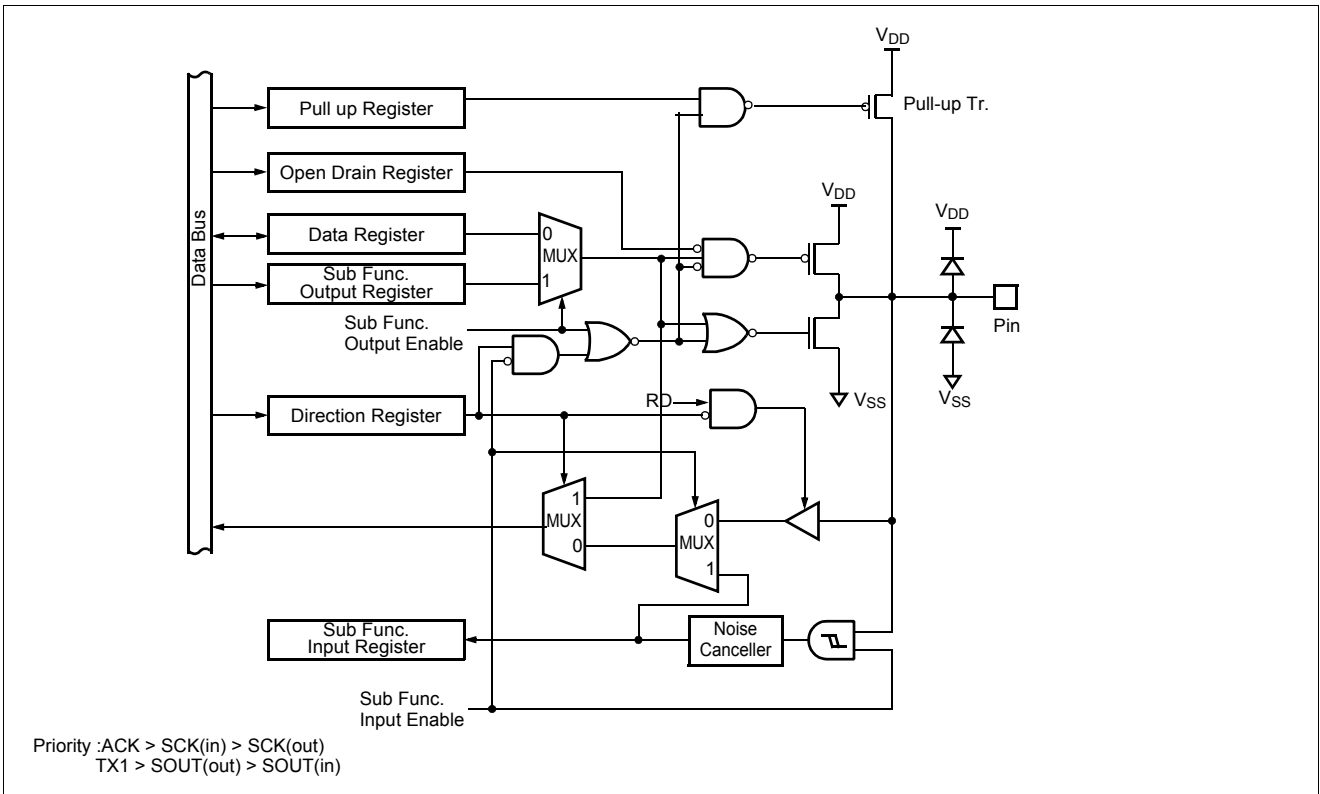
R01/EC0, R05/EC1/INT0, R06/INT1, R07/INT2, R13/  
RX/SI, R16/SDA, R41/IN3



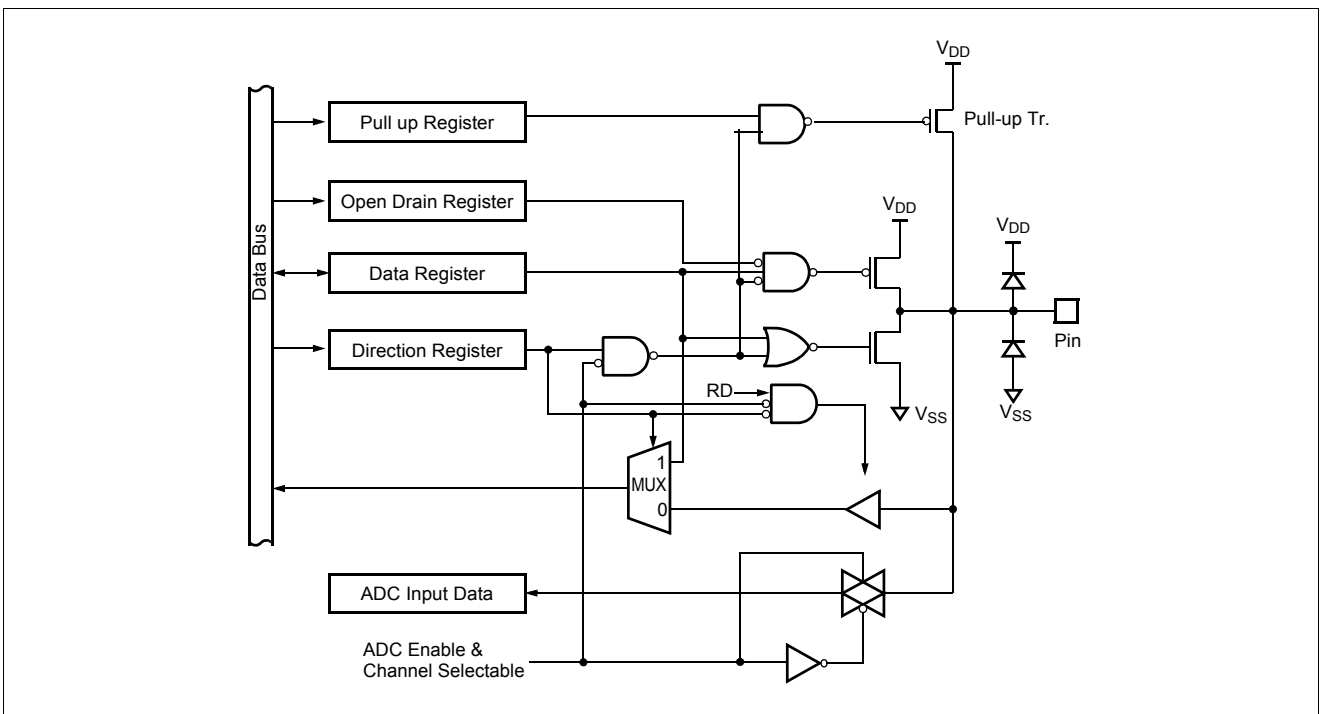
R00/PWM0/T00, R04/BUZO, R10/PWM1/T20, R17/  
SCL



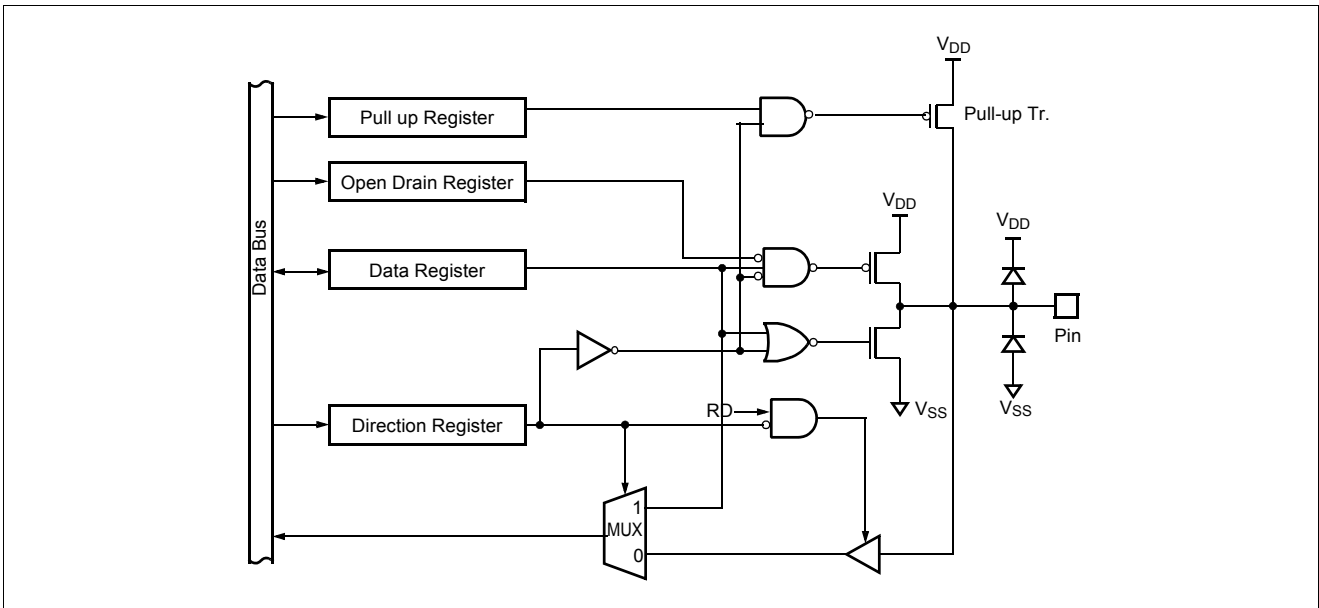
**R11/ACK/SCK, R12/TX/SOUT**



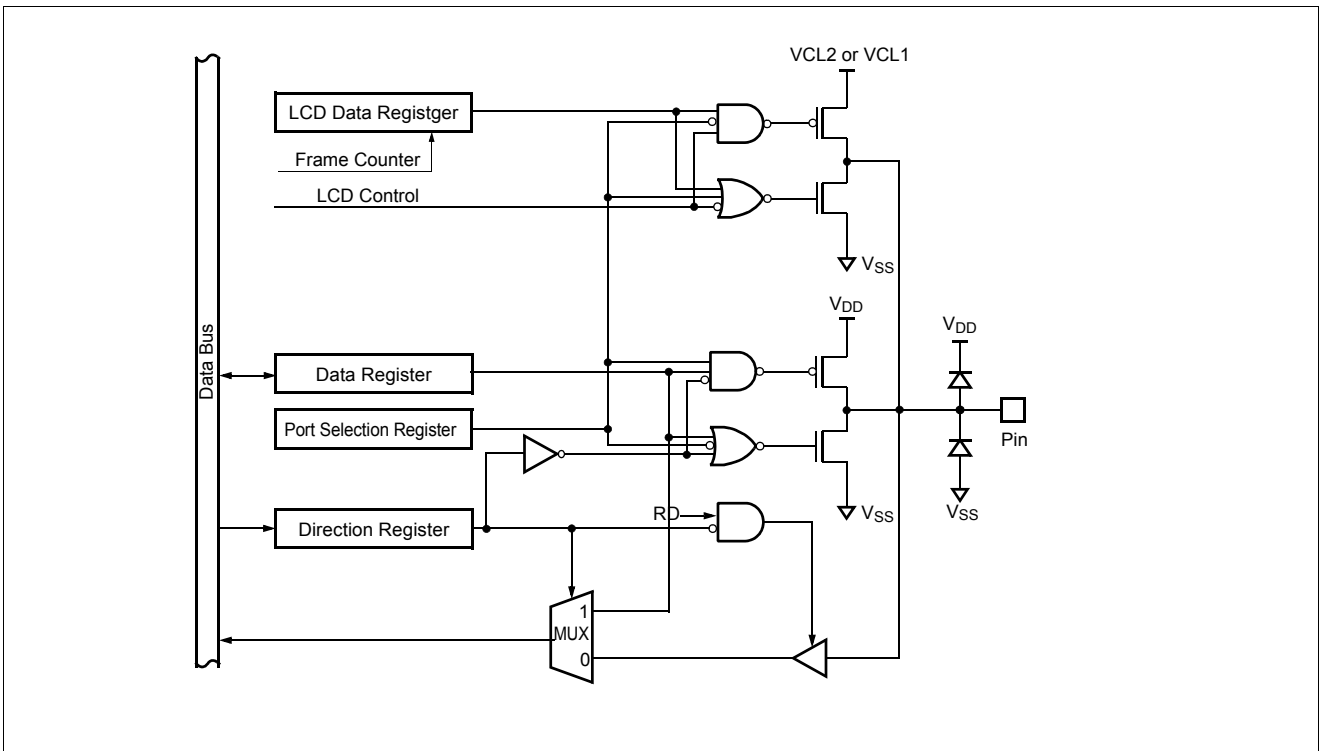
**R20/AN0~R27/AN7**



R02, R03, R14, R15, R40, R42

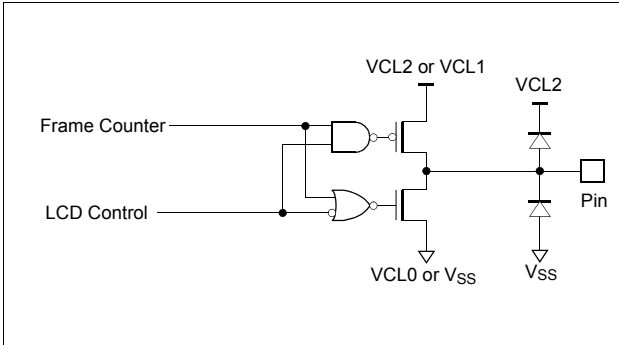


R50/SEG0~R77/SEG23

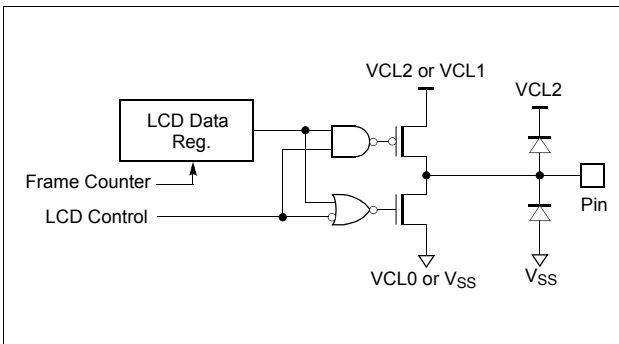




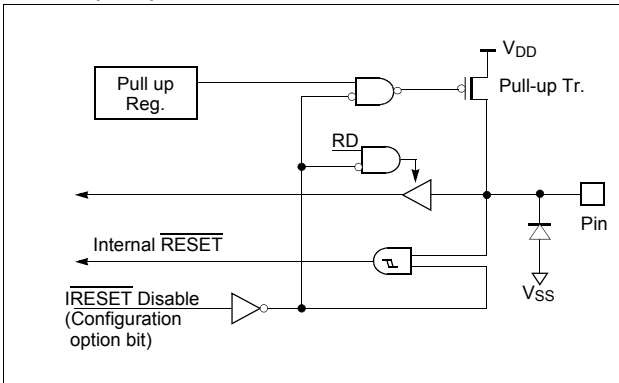
**COM0~COM3**



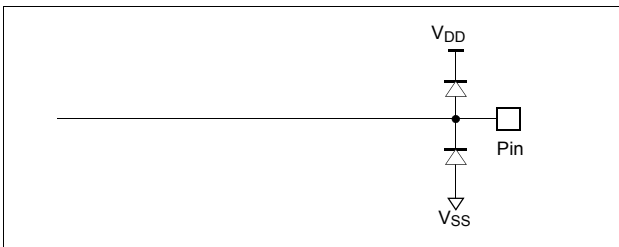
**COM0~COM3, SEG24~SEG35, COM4/  
SEG39~COM7/SEG36**



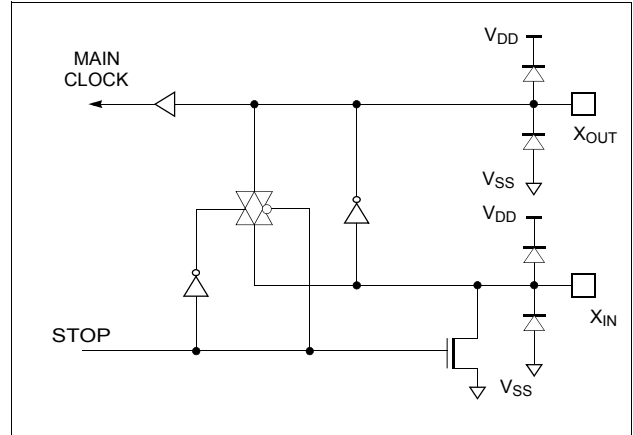
**RESET(R47)**



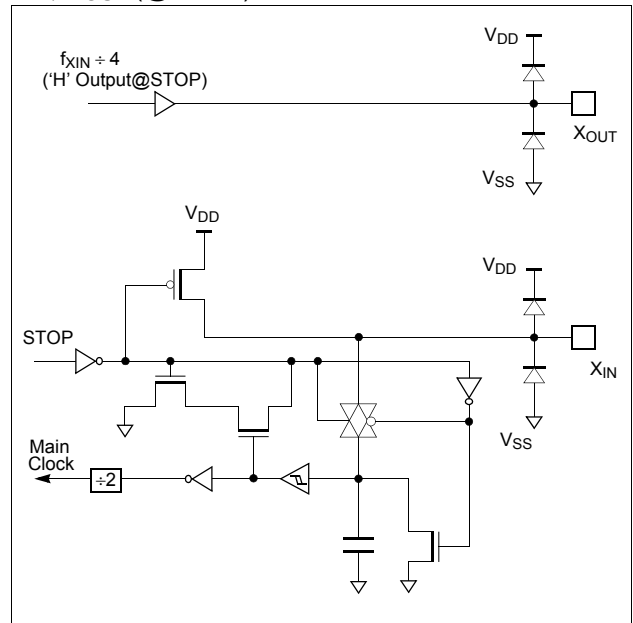
**PLL**



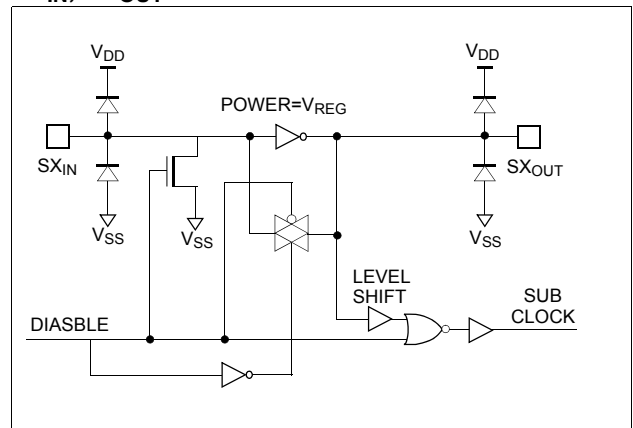
**XIN, XOUT (Crystal or Ceramic resonator)**



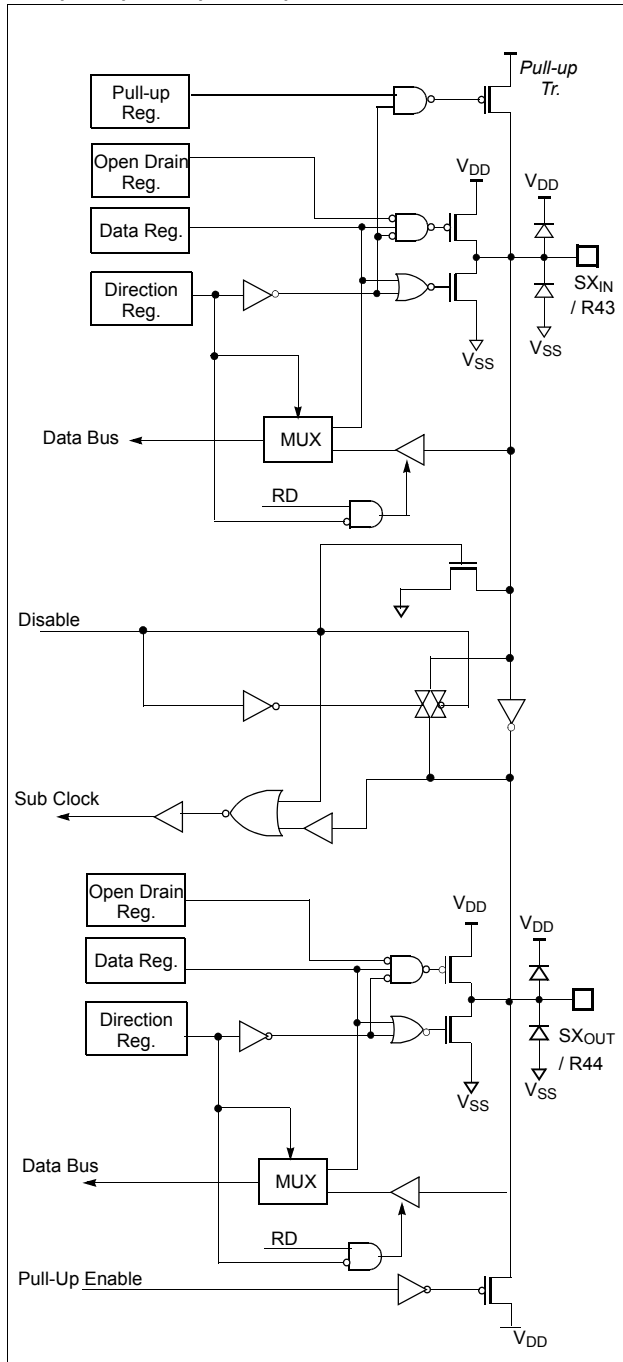
**XIN, XOUT (@RC, R)**



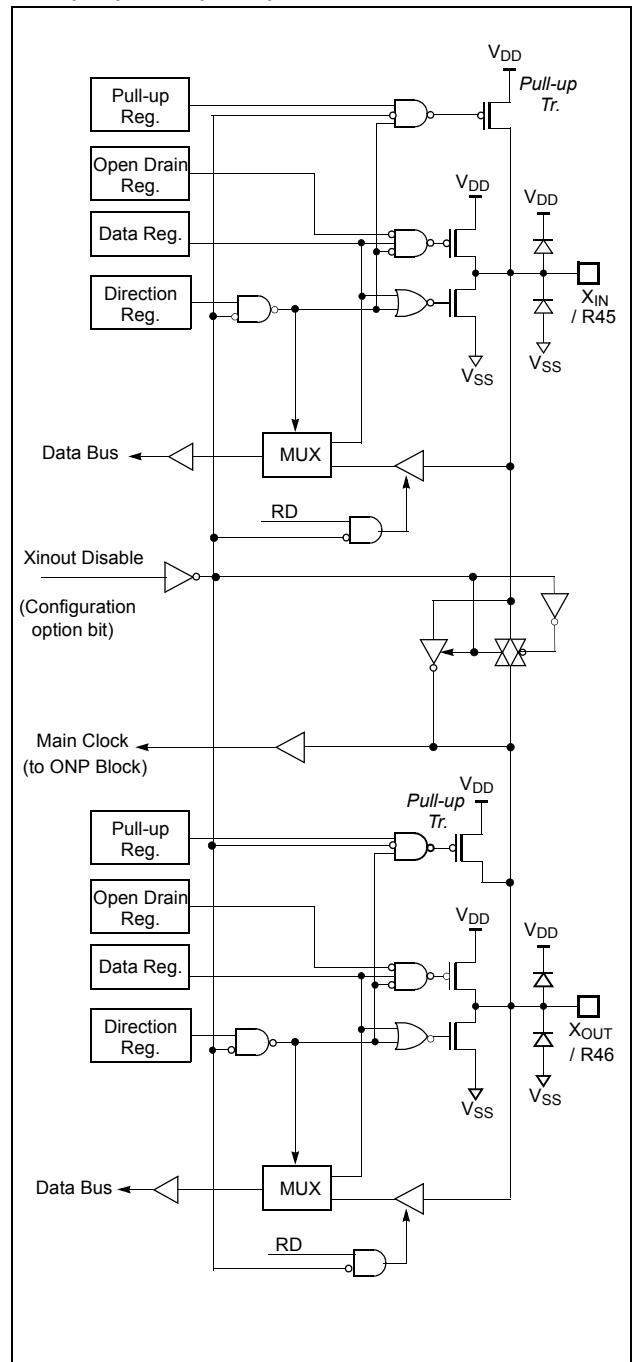
**SXIN, SXOUT**



R43(SX<sub>IN</sub>), R44(SX<sub>OUT</sub>)



R45 (X<sub>IN</sub>), R46 (X<sub>OUT</sub>)



## 7. ELECTRICAL CHARACTERISTICS

### 7.1 Absolute Maximum Ratings

Supply voltage ..... -0.3 to +6.0 V  
 Supply Voltage (AVref) .....  $V_{DD}-0.3$  to  $V_{DD}+0.3$  V  
 Storage Temperature ..... -45 to +125 °C  
 Voltage on any pin with respect to Ground ( $V_{SS}$ )  
 ..... -0.3 to  $V_{DD}+0.3$   
 Maximum current sunk by ( $I_{OL}$  per I/O Pin) ..... 20 mA  
 Maximum output current sourced by ( $I_{OH}$  per I/O Pin)  
 ..... 10 mA  
 Maximum current ( $\Sigma I_{OL}$ ) ..... 160 mA

Maximum current ( $\Sigma I_{OH}$ ) ..... 80 mA  
 Total Power Dissipation (PT) ..... 600 mW

---

**Note:** Stresses above 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 above 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

Parameter	Symbol	Condition	Specifications			Unit
			Min.	Typ.	Max.	
Supply Voltage	$V_{DD}$	$f_{MAIN}=12\text{MHz}$	4.5	-	5.5	V
		$f_{MAIN}=4\text{MHz}$	2.2	-	5.5	V
Main Operating Frequency	$f_{MAIN}$	$V_{DD}=2.2\sim 5.5\text{V}$	1	-	4.0	MHz
		$V_{DD}=4.5\sim 5.5\text{V}$	1	-	12.0	
	$f_{SUB}$	$V_{DD}=3.0\sim 5.5\text{V}$	-	32.768	-	kHz
Sub Operating Frequency	$f_{SUB}$	$V_{DD}=V_{DD}$	-	32.768	-	kHz
Operating Temperature	$T_{OPR}$	$V_{DD}=2.2\sim 5.5\text{V}$	-40	-	85	°C
		$V_{DD}=4.5\sim 5.5\text{V}$	-40	-	85	°C

### 7.3 DC Electrical Characteristics

( $T_A = -40 \sim 85^\circ\text{C}$ ,  $V_{DD} = 2.2 \sim 5.5\text{V}$ ,  $V_{SS} = 0\text{V}$ )

Parameter	Symbol	Pin / Condition	Specifications			Unit
			Min.	Typ.	Max.	
Input High Voltage	$V_{IH1}$	R0~R7	$0.7V_{DD}$	-	$V_{DD}+0.3$	V
	$V_{IH2}$	$\overline{\text{RESET}}$ , RX0, SI, SCK, ACK, X <sub>IN</sub> , SX <sub>IN</sub> , INT0~3, EC0~1	$0.8V_{DD}$	-	$V_{DD}+0.3$	
Input Low Voltage	$V_{IL1}$	R0~R7	-0.3	-	$0.3V_{DD}$	V
	$V_{IL2}$	$\overline{\text{RESET}}$ , RX0, SI, SCK, ACK, X <sub>IN</sub> , SX <sub>IN</sub> , INT0~3, EC0~1	-0.3	-	$0.2V_{DD}$	
Output High Voltage	$V_{OH1}$	R0~R4 ( $V_{DD}=4.5\text{V}$ , $I_{OH1}=-1.6\text{mA}$ )	$V_{DD}-0.3$	-	-	V
	$V_{OH2}$	R5~R7 ( $V_{DD}=4.5\text{V}$ , $I_{OH2}=-1.6\text{mA}$ )	$V_{DD}-1.0$	-	-	
	$V_{OH3}^1$	SEG0~39, COM0~3 ( $V_{DD}=4.5\text{V}$ , $V_{CL3}\sim 0=3\text{V}$ , $I_{OH3}=-15\mu\text{A}$ )	$V_{CL3}-0.4$	-	-	
Output Low Voltage	$V_{OL1}$	R0~R4 ( $V_{DD}=4.5\text{V}$ , $I_{OL1}=1.6\text{mA}$ )	-	-	0.35	V
	$V_{OL2}$	R5~R7 ( $V_{DD}=4.5\text{V}$ , $I_{OL2}=1.6\text{mA}$ )			0.4	
	$V_{OL3}^2$	SEG0~39, COM0~3 ( $V_{DD}=4.5\text{V}$ , $V_{CL3}\sim 0=3\text{V}$ , $I_{OL3}=15\mu\text{A}$ )			0.12	
Input High Leakage Current	$I_{IH}$	All input pins including R5~R7 ( $V_{IN}=V_{DD}$ )	-	-	1	$\mu\text{A}$
Input Low Leakage Current	$I_{IL}$	All input pins including R5~R7 ( $V_{IN}=V_{SS}$ )	-1	-	-	
Brown-out Detector	$V_{BOD}$	$V_{DD}$ (BODR<2:0>=000)	$2.0\pm 15\%$	2.0	$2.0\pm 15\%$	V
		$V_{DD}$ (BODR<2:0>=011)	$2.7\pm 15\%$	2.7	$2.7\pm 15\%$	V
		$V_{DD}$ (BODR<2:0>=110)	$3.6\pm 15\%$	3.6	$3.6\pm 15\%$	
POR(Power on Reset) Level	$V_{POR}$	$V_{DD}$ ( $T_A=25^\circ\text{C}$ )	2.0	2.4	2.8	V
VDD Start Voltage <sup>3</sup>	$V_{START}$	$V_{DD}$ ( $T_A=25^\circ\text{C}$ )	$V_{SS}$		0.3	V
Config Read Voltage <sup>3</sup>	$V_{config}$	$T_{VDD}=40\text{ms/V}$ , $V_{START}=V_{SS}$	1.8			V
VDD rising Time <sup>3</sup>	$T_{VDD}$	$V_{DD}$ ( $T_A=25^\circ\text{C}$ )			40	ms/V
Hysteresis	$V_{T+} \sim V_{T-}$	$\overline{\text{RESET}}$ , RX0, INT0~3, EC0~1 ( $V_{DD}=5\text{V}$ )	$0.2V_{DD}$	-	$0.8V_{DD}$	V
Pull-up Current	$I_{PU}$	R0~R4 ( $V_{DD}=3.0\text{V}$ , $V_{PIN}=0\text{V}$ )	20	-	60	$\mu\text{A}$
Current dissipation in active mode <sup>4</sup>	$I_{DD}$	$V_{DD}$ ( $f_{MAIN}=12\text{MHz}$ , $V_{DD}=5.5\text{V}$ , $f_{SUB}=0$ )	-	5	15	mA
Current dissipation in sleep mode <sup>5</sup>	$I_{SLEEP}$	$V_{DD}$ ( $f_{MAIN}=12\text{MHz}$ , $V_{DD}=5.5\text{V}$ , $f_{SUB}=0$ )	-	2	4	
Current dissipation in sub active mode <sup>6</sup>	$I_{subactive}$	$f_{MAIN}=\text{off}$ , $V_{DD}=5\text{V}$ , $f_{SUB}=32.768\text{kHz}$	-	67	-	$\mu\text{A}$
	$I_{subsleep}$	$f_{MAIN}=\text{off}$ , $V_{DD}=5\text{V}$ , $f_{SUB}=32.768\text{kHz}$	-	32	-	$\mu\text{A}$
Current dissipation in stop mode	$I_{STOP}$	$f_{MAIN}=\text{off}$ , $V_{DD}=5.5\text{V}$ , $f_{SUB}=0$	-	3	7	$\mu\text{A}$
	$I_{SUB}$	$f_{MAIN}=\text{off}$ , $V_{DD}=5.5\text{V}$ , $f_{SUB}=32.768\text{kHz}$		7	14	$\mu\text{A}$
Internal 8MHz Oscillation Frequency	$f_{IN8M}$	$V_{DD}=5\text{V}$ , $T_A=25^\circ\text{C}$	$8\pm 10\%$	8	$8\pm 10\%$	MHz

Parameter	Symbol	Pin / Condition	Specifications			Unit
			Min.	Typ.	Max.	
Internal 4MHz Oscillation Frequency	$f_{IN4M}$	VDD=5V, T <sub>A</sub> =25°C	±10%	4	±10%	MHz

1. V<sub>OH3</sub> is the voltage when VCL3, VCL2, VCL1 and VCL0 are supplied at pads.
2. V<sub>OL3</sub> is the voltage when V<sub>SS</sub> is supplied at pad.
3. These parameters are presented for design guidance only and not tested or guaranteed.
4. Current dissipation is proportioned according to operation voltage and frequency.
5. In sleep mode, oscillation continues and peripherals are operated normally but internal CPU clock stops.
6. In sub sleep mode, sub oscillation continues and peripherals are operated normally but internal CPU clock stops.

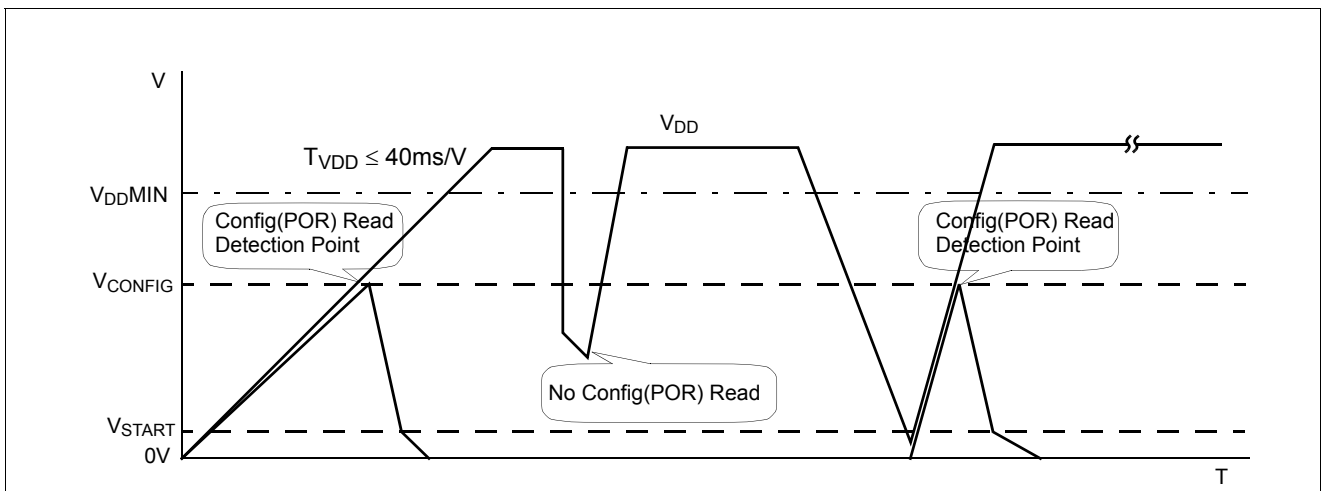


Figure 7-1 Config Read Voltage including POR vs Supply Voltage

### 7.4 LCD Characteristics

(T<sub>A</sub> = -40~85°C, V<sub>DD</sub> = 2.2~5.5V, V<sub>SS</sub> = 0V)

Parameter	Symbol	Condition	Specifications			Unit
			Min.	Typ.	Max.	
LCD Common Output Current	I <sub>COM</sub>	Output Voltage Deviation=0.2V	30	-	-	μA
LCD Segment Output Current	I <sub>SEG</sub>	Output Voltage Deviation=0.2V	5	-	-	

### 7.5 A/D Converter Characteristics

(T<sub>A</sub> = -40~85°C, V<sub>DD</sub> = AV<sub>ref</sub> = 5.12V/3.072V, V<sub>SS</sub> = 0V)

Parameter	Symbol	Pin/Condition	Specifications			Unit
			Min.	Typ.	Max.	
Resolution	N <sub>R</sub>	-	-	10	-	Bit
Analog Power Supply Input Voltage Range	AV <sub>REF</sub> <sup>1</sup>	-	AV <sub>SS</sub>	-	VDD	V
Analog Input Voltage Range	V <sub>AIN</sub>	-	AV <sub>SS</sub>	-	AV <sub>ref</sub>	
Conversion Current	I <sub>CON</sub>	V <sub>DD</sub> = 5.12V F <sub>XIN</sub> = 8MHz	-	80	200	μA

Parameter	Symbol	Pin/Condition	Specifications			Unit
			Min.	Typ.	Max.	
Overall Accuracy	N <sub>ACC</sub>	-	-	±1.0	±3.0	LSB
Non Linearity Error	N <sub>NLE</sub>	f <sub>XIN</sub> = 4MHz	-	-	±3.0	
Differential Non Linearity Error	N <sub>DNLE</sub>		-	-	±3.0	
Zero Offset Error	N <sub>ZOE</sub>		-	-	±3.0	
Full Scale Error	N <sub>FSE</sub>		-	-	±3.0	
Gain Error	N <sub>GE</sub>		-	-	±3.0	
Conversion Time (Clock)	T <sub>CONV</sub> (T <sub>ACLK</sub> )	Conversion Time = T <sub>ACLK</sub> × 13	1.0	-	-	μS
Analog Input Impedance	R <sub>AN</sub>	-	5	100	-	MΩ

1. If the AV<sub>REF</sub> voltage is less than V<sub>DD</sub> voltage and analog input pins(ANX), shared with various alternate function, are used bidirectional I/O port, the leakage current may flow V<sub>DD</sub> pin to AV<sub>REF</sub> pin in output high mode or analog input pins(ANX) to AV<sub>REF</sub> pin in input high mode.

## 7.6 AC Characteristics

(TA=25°C, V<sub>DD</sub>=4V, AV<sub>ref</sub>=4V, V<sub>SS</sub>=AV<sub>SS</sub>=0V)

Parameter	Symbol	Pins	Specifications			Unit
			Min.	Typ.	Max.	
Main Operating Frequency	f <sub>MCP</sub>	X <sub>IN</sub>	0.4	-	12	MHz
Sub Operating Frequency	f <sub>SCP</sub>	SX <sub>IN</sub>	30	32.768	35	kHz
System Clock Frequency <sup>1</sup>	t <sub>SYS</sub>	-	166	-	5000	nS
Main Oscillation Stabilization Time (4MHz)	t <sub>MST</sub>	X <sub>IN</sub> , X <sub>OUT</sub>	-	-	20	mS
Sub Oscillation Stabilization Time	t <sub>SST</sub>	SX <sub>IN</sub> , SX <sub>OUT</sub>	-	1	2	S
External Clock "H" or "L" Pulse Width	t <sub>MCPW</sub>	X <sub>IN</sub>	35	-	-	nS
	t <sub>SCPW</sub>	SX <sub>IN</sub>	5	-	-	μS
External Clock Transition Time	t <sub>RCP</sub> , t <sub>FCP</sub>	X <sub>IN</sub>	-	-	20	nS
Interrupt Pulse Width	t <sub>IW</sub>	INT0, INT1, INT2, IN3	2	-	-	t <sub>sys</sub>
RESET Input Pulse "L" Width	t <sub>RST</sub>	RESET	8	-	-	t <sub>sys</sub>
Event Counter Input "H" or "L" Pulse Width	t <sub>ECW</sub>	EC0~1	2	-	-	t <sub>sys</sub>
Event Counter Transition Time	t <sub>REC</sub> , t <sub>FEC</sub>	EC0~1	-	-	20	nS

1.SCMR=XXXX000X<sub>B</sub> that is f<sub>MAIN</sub>÷2

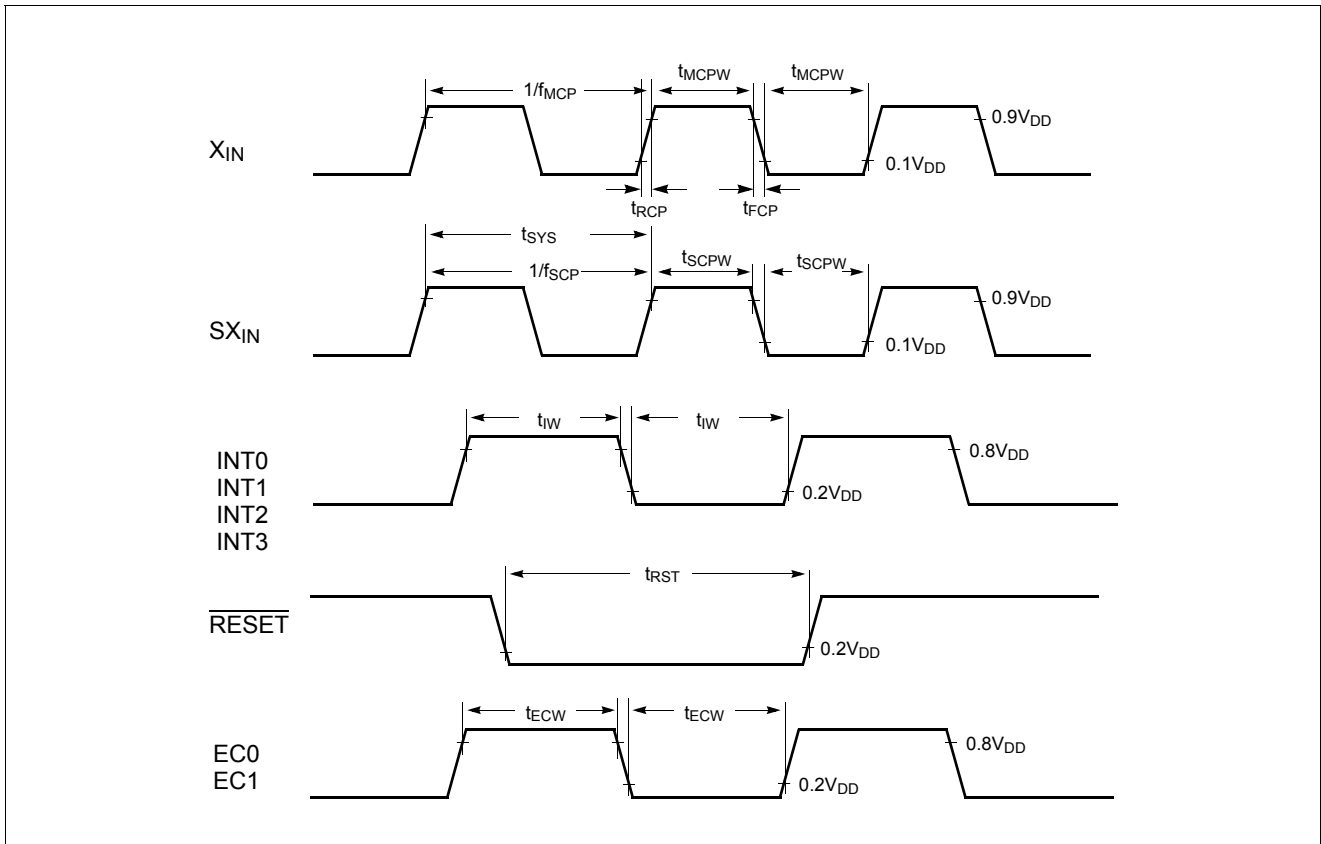


Figure 7-2 AC Timing Chart



**7.7 Serial I/O Characteristics**

( $T_A = -40 \sim 85^\circ\text{C}$ ,  $V_{DD} = 5.0\text{V} \pm 10\%$ ,  $V_{SS} = 0\text{V}$ )

Parameter	Symbol	Pins	Specifications			Unit
			Min.	Typ.	Max.	
Input Clock Pulse Period	$t_{SCYC}$	SCK	$2t_{SYS} + 200$	-	-	ns
Input Clock "H" or "L" Pulse Width	$t_{SCKW}$		$t_{SYS} + 70$	-	-	
Input Clock Pulse Transition Time	$t_{FSCK}, t_{RSCK}$		-	-	30	
Output Clock Cycle Time	$t_{SCYC}$		$4t_{SYS}$	-	$16t_{SYS}$	
Output Clock "H" or "L" Pulse Width	$t_{SCKW}$		$2t_{SYS} - 30$	-	-	
Output Clock Transition Time	$t_{FSCK}, t_{RSCK}$		-	-	30	
Output Clock Delay Time	$t_{DS}$		-	-	100	
Input Pulse Transition Time	$t_{FSIN}, t_{RSIN}$	SI	-	-	30	
Input Setup Time (External SCK)	$t_{ESUS}$		100	-	-	
Input Setup Time (Internal SCK)	$t_{ISUS}$		200	-	-	
Input Hold Time	$t_{HS}$		$t_{SYS} + 70$	-	-	

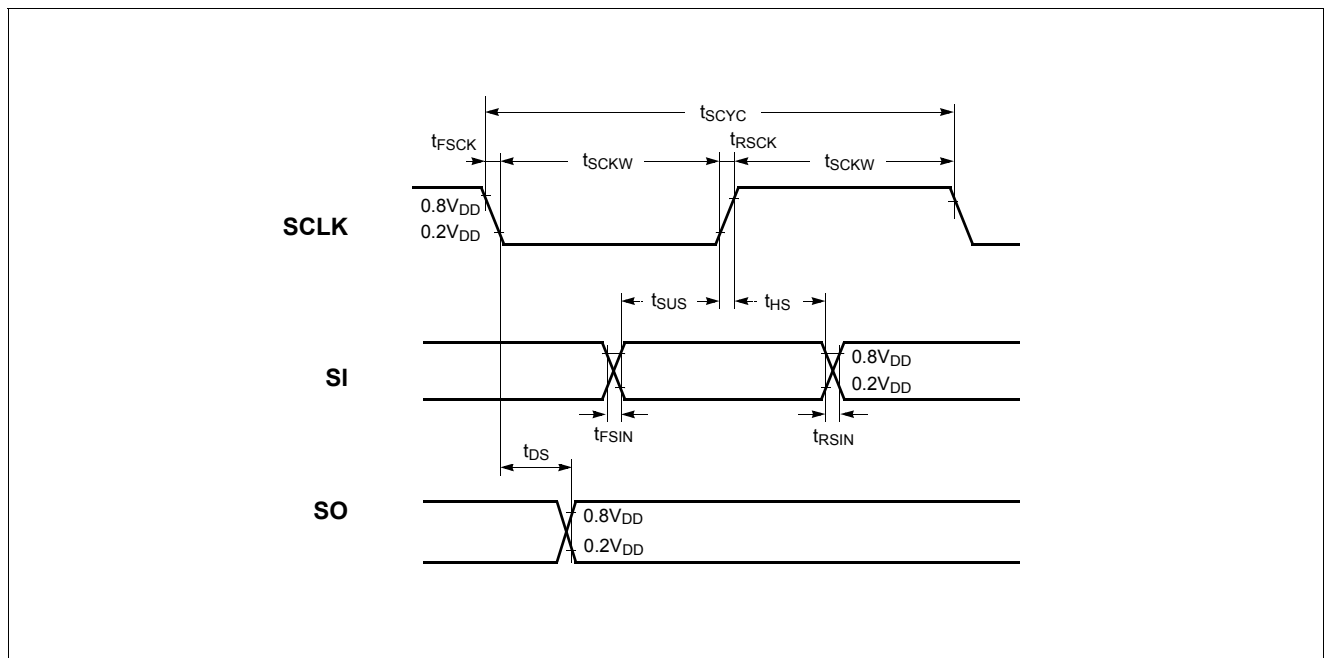


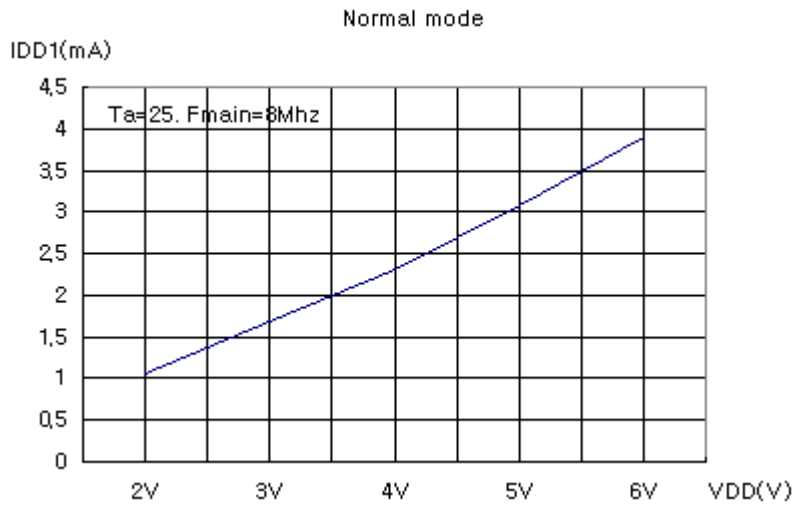
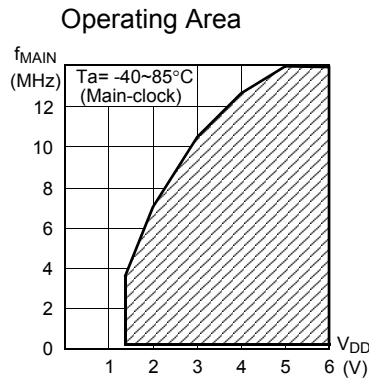
Figure 7-3 Serial I/O Timing Chart

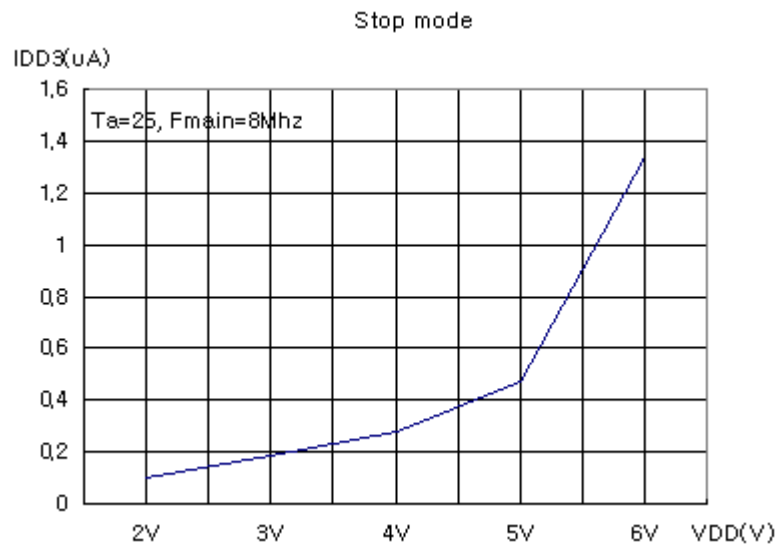
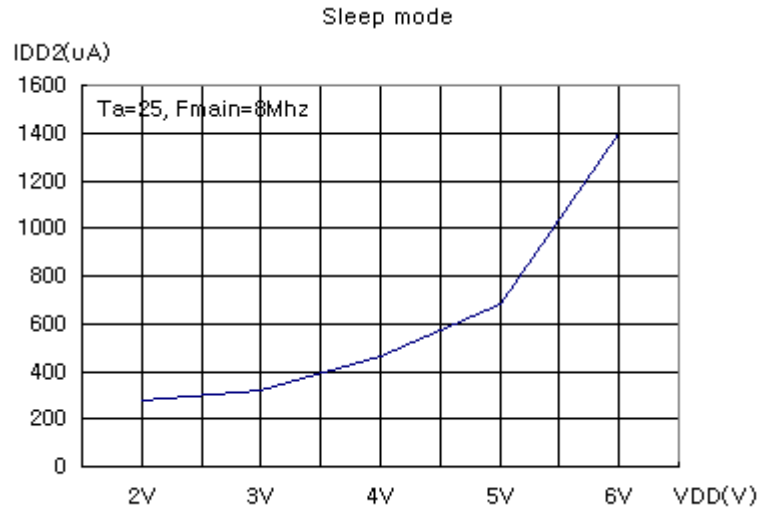
### 7.8 Typical Characteristics

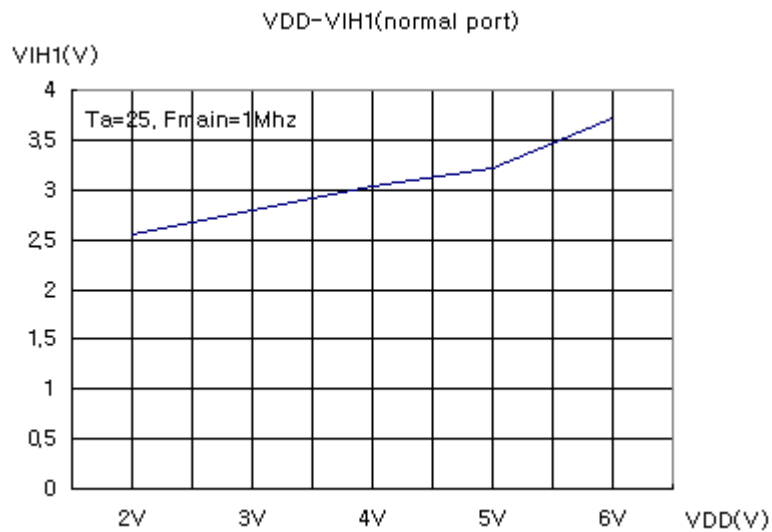
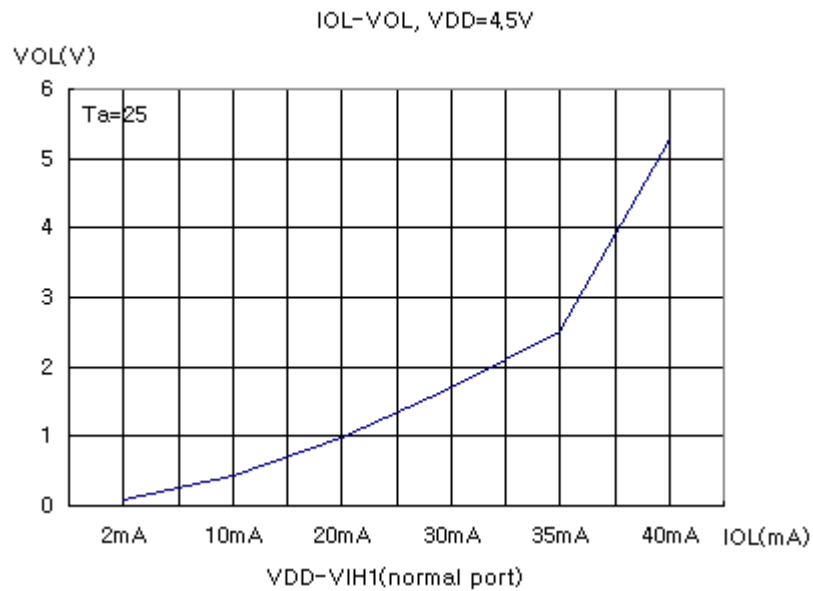
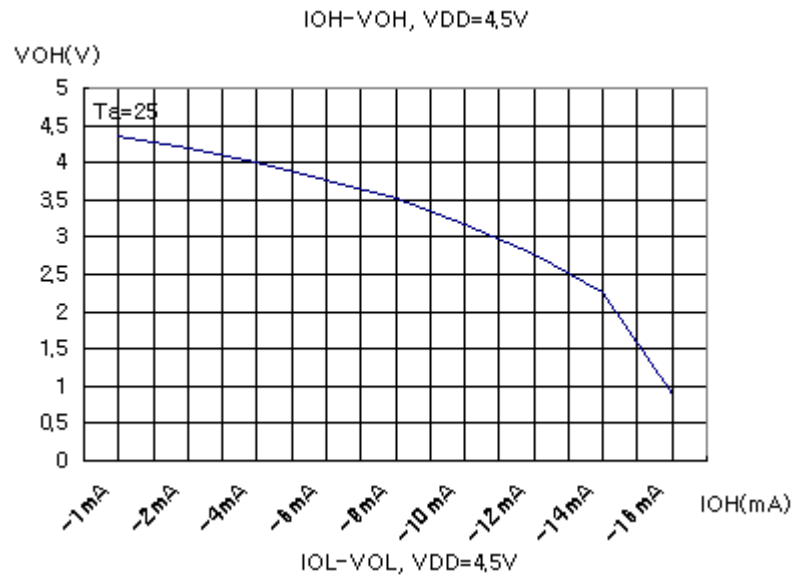
These graphs and tables are for design guidance only and are not tested or guaranteed.

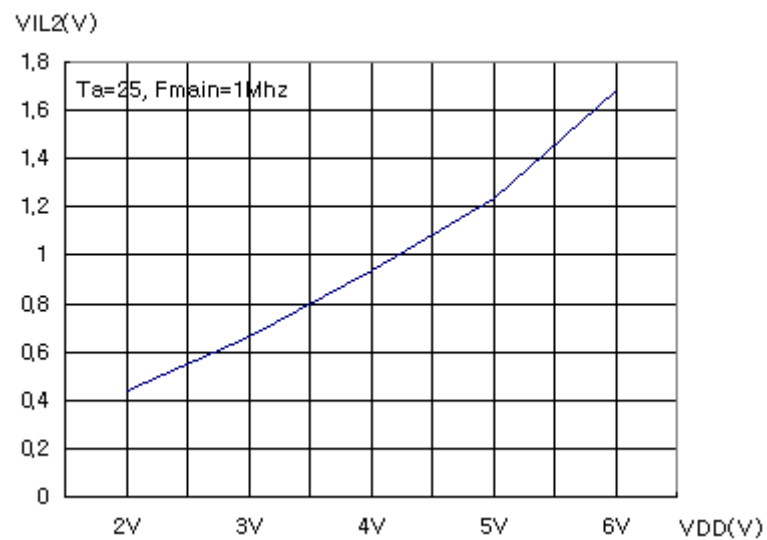
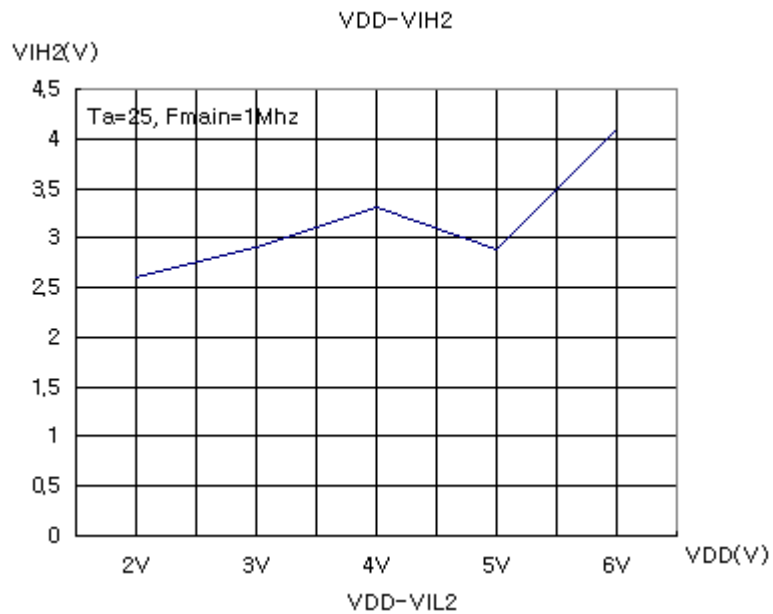
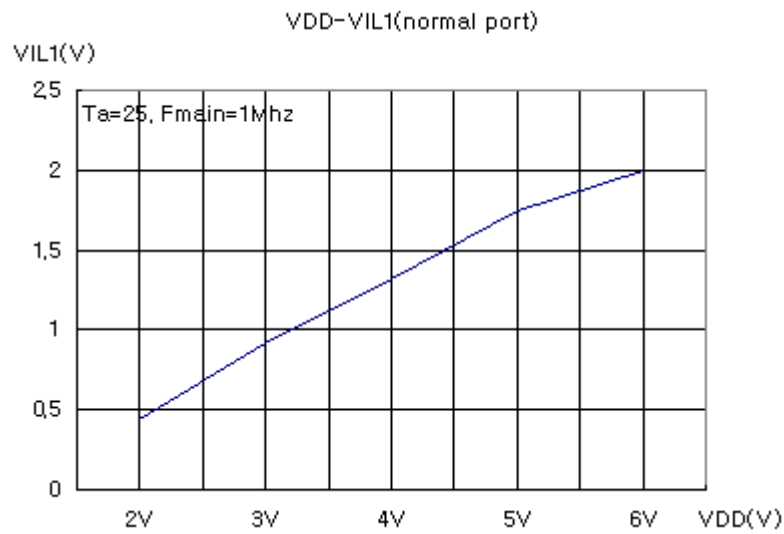
**In some graphs or tables, the data presented are outside specified operating range (e.g. outside specified V<sub>DD</sub> range). This is for information only and devices are guaranteed to operate properly only within the specified range.**

The data 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.









## 8. MEMORY ORGANIZATION

The have separate address spaces for Program memory, Data Memory and Display memory. Program memory can only be read, not written to. It can be up to 16K bytes of

### 8.1 Registers

This device has six registers that are the Program Counter (PC), a Accumulator (A), two index registers (X, Y), the Stack Pointer (SP), and the Program Status Word (PSW). The Program Counter consists of 16-bit register.

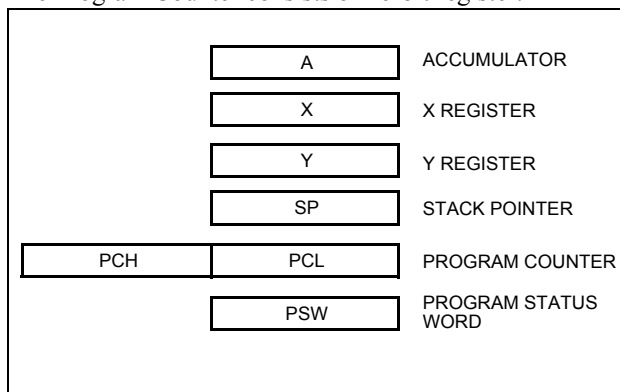


Figure 8-1 Configuration of Registers

**Accumulator:** The Accumulator is the 8-bit general purpose register, used for data operation such as transfer, temporary saving, and conditional judgement, etc.

The Accumulator can be used as a 16-bit register with Y Register as shown below.

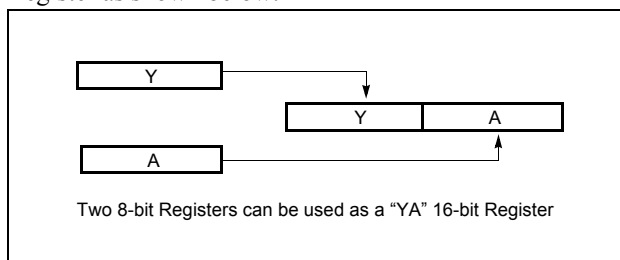


Figure 8-2 Configuration of YA 16-bit Register

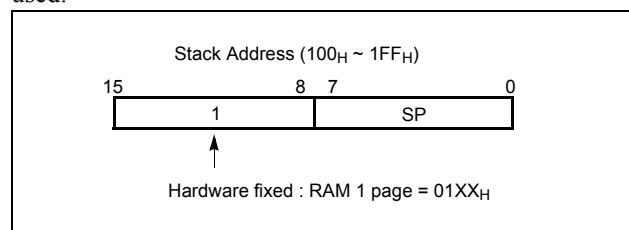
**X, Y Registers:** In the addressing mode which uses these index registers, the register contents are added to the specified address, which becomes the actual address. These modes are extremely effective for referencing subroutine tables and memory tables. The index registers also have increment, decrement, comparison and data transfer functions, and they can be used as simple accumulators.

**Stack Pointer:** The Stack Pointer is an 8-bit register used for occurrence interrupts and calling out subroutines. Stack Pointer identifies the location in the stack to be accessed (save or restore).

Program memory. Data memory can be read and written to up to 512 bytes including the stack area. Display memory has prepared 64bytes for LCD.

Generally, SP is automatically updated when a subroutine call is executed or an interrupt is accepted. However, if it is used in excess of the stack area permitted by the data memory allocating configuration, the user-processed data may be lost.

The stack can be located at any position within 100<sub>H</sub> to 1FF<sub>H</sub> of the internal data memory. The SP is not initialized by hardware, requiring to write the initial value (the location with which the use of the stack starts) by using the initialization routine. Normally, the initial value of "1FF<sub>H</sub>" is used.



#### Caution:

**The Stack Pointer must be initialized by software because its value is undefined after RESET.**

Example: To initialize the SP

```
LDX #0FFH ;
TXSP ; SP ← 0FFH
```

**Program Counter:** The Program Counter is a 16-bit wide which consists of two 8-bit registers, PCH and PCL. This counter indicates the address of the next instruction to be executed. In reset state, the program counter has reset routine address (PC<sub>H</sub>:0FF<sub>H</sub>, PC<sub>L</sub>:0FE<sub>H</sub>).

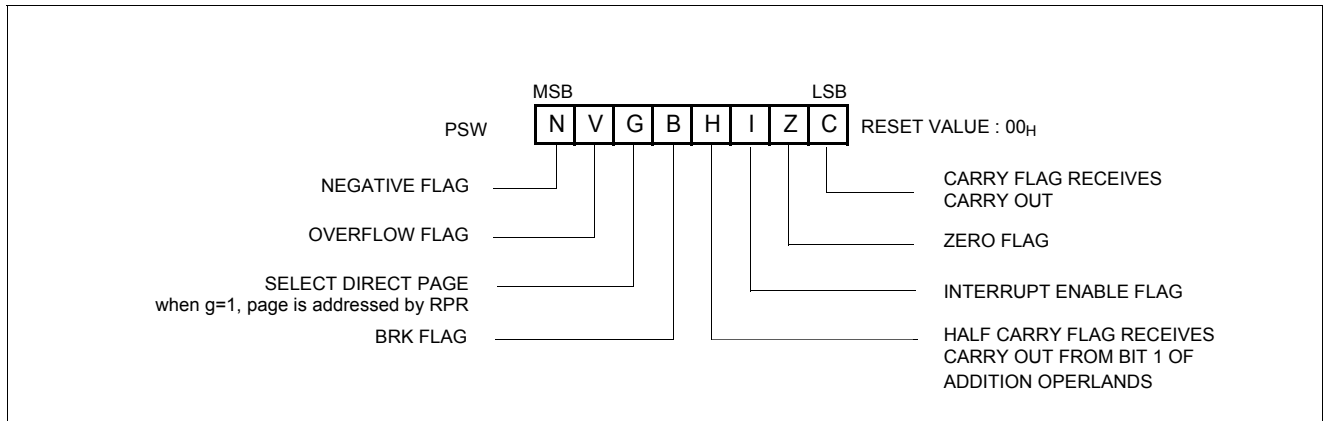
**Program Status Word:** The Program Status Word (PSW) contains several bits that reflect the current state of the CPU. The PSW is described in Figure 8-3. It contains the Negative flag, the Overflow flag, the Break flag the Half Carry (for BCD operation), the Interrupt enable flag, the Zero flag, and the Carry flag.

[Carry flag C]

This flag stores any carry or borrow from the ALU of CPU after an arithmetic operation and is also changed by the Shift Instruction or Rotate Instruction.

[Zero flag Z]

This flag is set when the result of an arithmetic operation or data transfer is “0” and is cleared by any other result.



**Figure 8-3 PSW (Program Status Word) Register**

[Interrupt disable flag I]

This flag enables/disables all interrupts except interrupt caused by Reset or software BRK instruction. All interrupts are disabled when cleared to “0”. This flag immediately becomes “0” when an interrupt is served. It is set by the EI instruction and cleared by the DI instruction.

[Half carry flag H]

After operation, this is set when there is a carry from bit 3 of ALU or there is no borrow from bit 4 of ALU. This bit can not be set or cleared except CLRV instruction with Overflow flag (V).

[Break flag B]

This flag is set by software BRK instruction to distinguish BRK from TCALL instruction with the same vector address.

[Direct page flag G]

This flag assigns RAM page for direct addressing mode. In

the direct addressing mode, addressing area is from zero page 00H to 0FFH when this flag is "0". If it is set to "1", addressing area is assigned by RPR register (address 0F3H). It is set by SETG instruction and cleared by CLRG.

[Overflow flag V]

This flag is set to “1” when an overflow occurs as the result of an arithmetic operation involving signs. An overflow occurs when the result of an addition or subtraction exceeds +127 (7FH) or -128 (80H). The CLRV instruction clears the overflow flag. There is no set instruction. When the BIT instruction is executed, bit 6 of memory is copied to this flag.

[Negative flag N]

This flag is set to match the sign bit (bit 7) status of the result of a data or arithmetic operation. When the BIT instruction is executed, bit 7 of memory is copied to this flag.

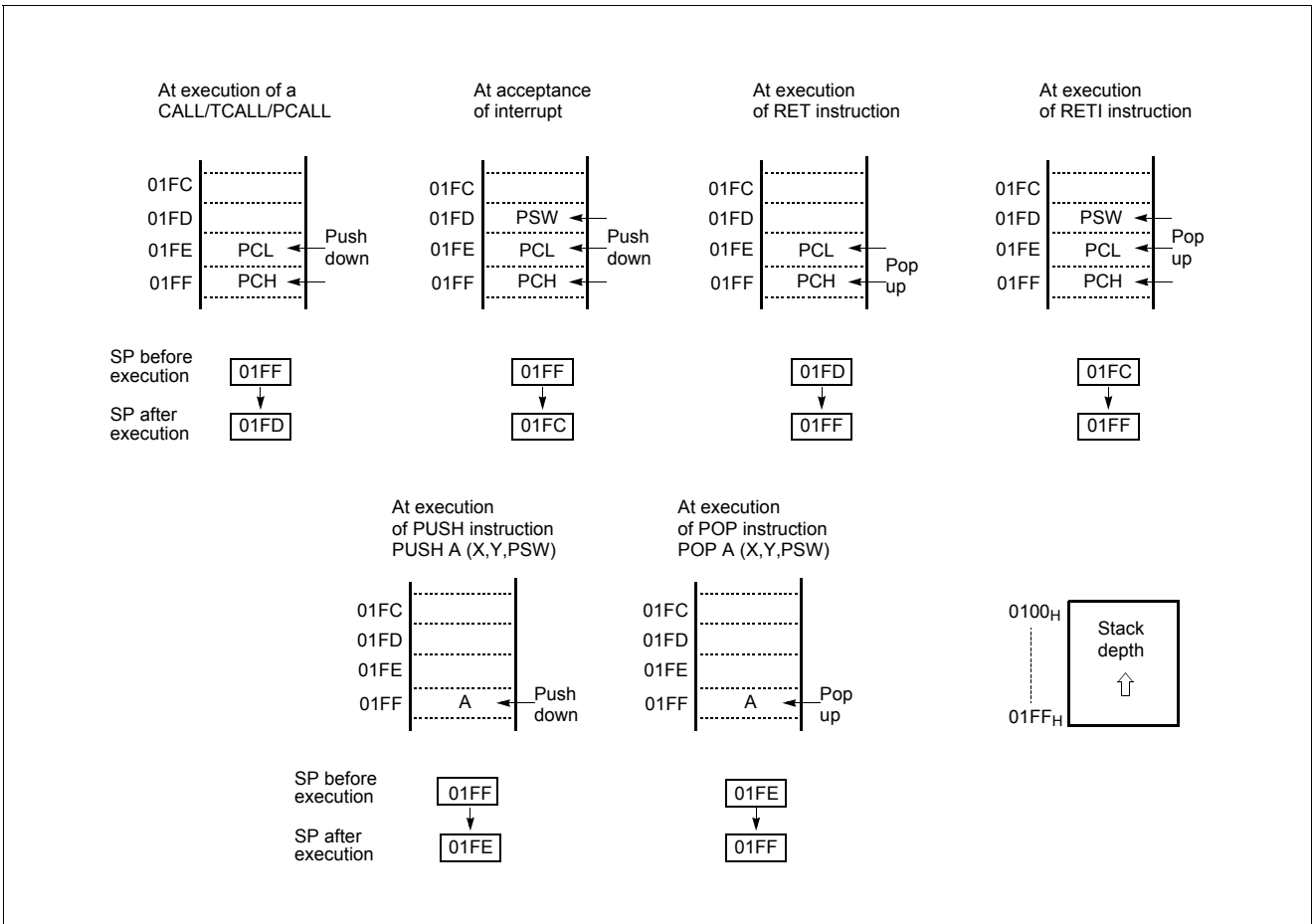


Figure 8-4 Stack Operation

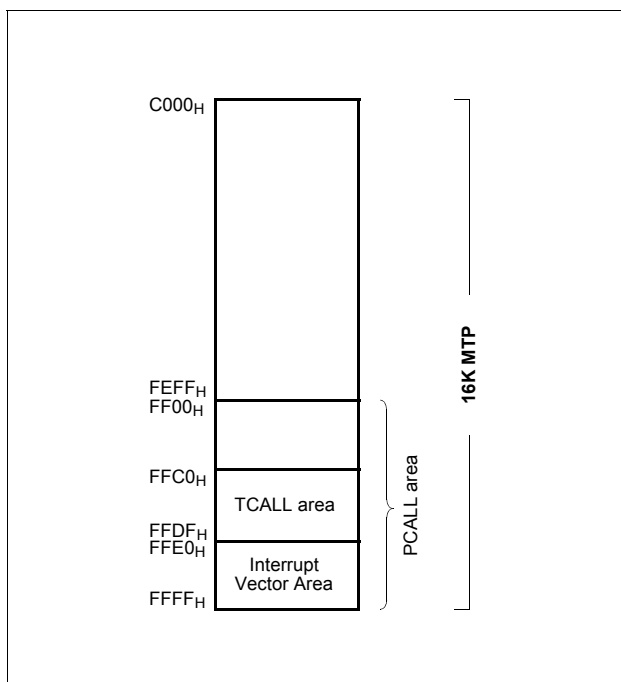


## 8.2 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has 16K bytes program memory space only physically implemented. Accessing a location above FFFF<sub>H</sub> will cause a wrap-around to 0000<sub>H</sub>.

Figure 8-5 shows a map of Program Memory. After reset, the CPU begins execution from reset vector which is stored in address FFFE<sub>H</sub> and FFFF<sub>H</sub> as shown in Figure 8-6.

As shown in Figure 8-5, each area is assigned a fixed location in Program Memory. Program Memory area contains the user program.



**Figure 8-5 Program Memory Map**

Page Call (PCALL) area contains subroutine program to reduce program byte length by using 2 bytes PCALL instead of 3 bytes CALL instruction. If it is frequently called, it is more useful to save program byte length.

Table Call (TCALL) causes the CPU to jump to each TCALL address, where it commences the execution of the service routine. The Table Call service area spaces 2-byte for every TCALL: 0FFC0<sub>H</sub> for TCALL15, 0FFC2<sub>H</sub> for TCALL14, etc., as shown in Figure 8-7.

Example: Usage of TCALL

```

LDA    #5
      TCALL 0FH           ; 1BYTE INSTRUCTION
      :               ; INSTEAD OF 2 BYTES
      :               ; NORMAL CALL
;
; TABLE CALL ROUTINE
;
FUNC_A: LDA    LRG0
      RET
;
FUNC_B: LDA    LRG1
      RET
;
; TABLE CALL ADD. AREA
;
      ORG    0FFC0H
      DW    FUNC_A
      DW    FUNC_B
    
```

The interrupt causes the CPU to jump to specific location, where it commences the execution of the service routine. The External interrupt 0, for example, is assigned to location 0FFFA<sub>H</sub>. The interrupt service locations spaces 2-byte interval: 0FF8<sub>H</sub> and 0FFF9<sub>H</sub> for External Interrupt 1, 0FFFA<sub>H</sub> and 0FFFB<sub>H</sub> for External Interrupt 0, etc.

Any area from 0FF00<sub>H</sub> to 0FFFF<sub>H</sub>, if it is not going to be used, its service location is available as general purpose Program Memory.

Address	Vector Area Memory
0FFE0 <sub>H</sub>	Watch Timer interrupt Vector Area
	Watch dog timer interrupt Vector
E2	Basic Interval Timer Interrupt Vector Area
E4	AD Converter Interrupt Vector Area
	I2C Interrupt Vector Area
E6	Timer/Counter 3 Interrupt Vector Area
E8	Timer/Counter 2 Interrupt Vector Area
EA	Timer/Counter 1 Interrupt Vector Area
EC	Timer/Counter 0 Interrupt Vector Area
EE	SPI Interrupt Vector Area
F2	UART TX0 Interrupt Vector Area
	UART RX0 Interrupt Vector Area
F4	External Interrupt 3 Vector Area
F6	External Interrupt 2 Vector Area
F8	External Interrupt 1 Vector Area
FA	External Interrupt 0 Vector Area
FE	Reset Interrupt Vector Area

**Figure 8-6 Interrupt Vector Area**

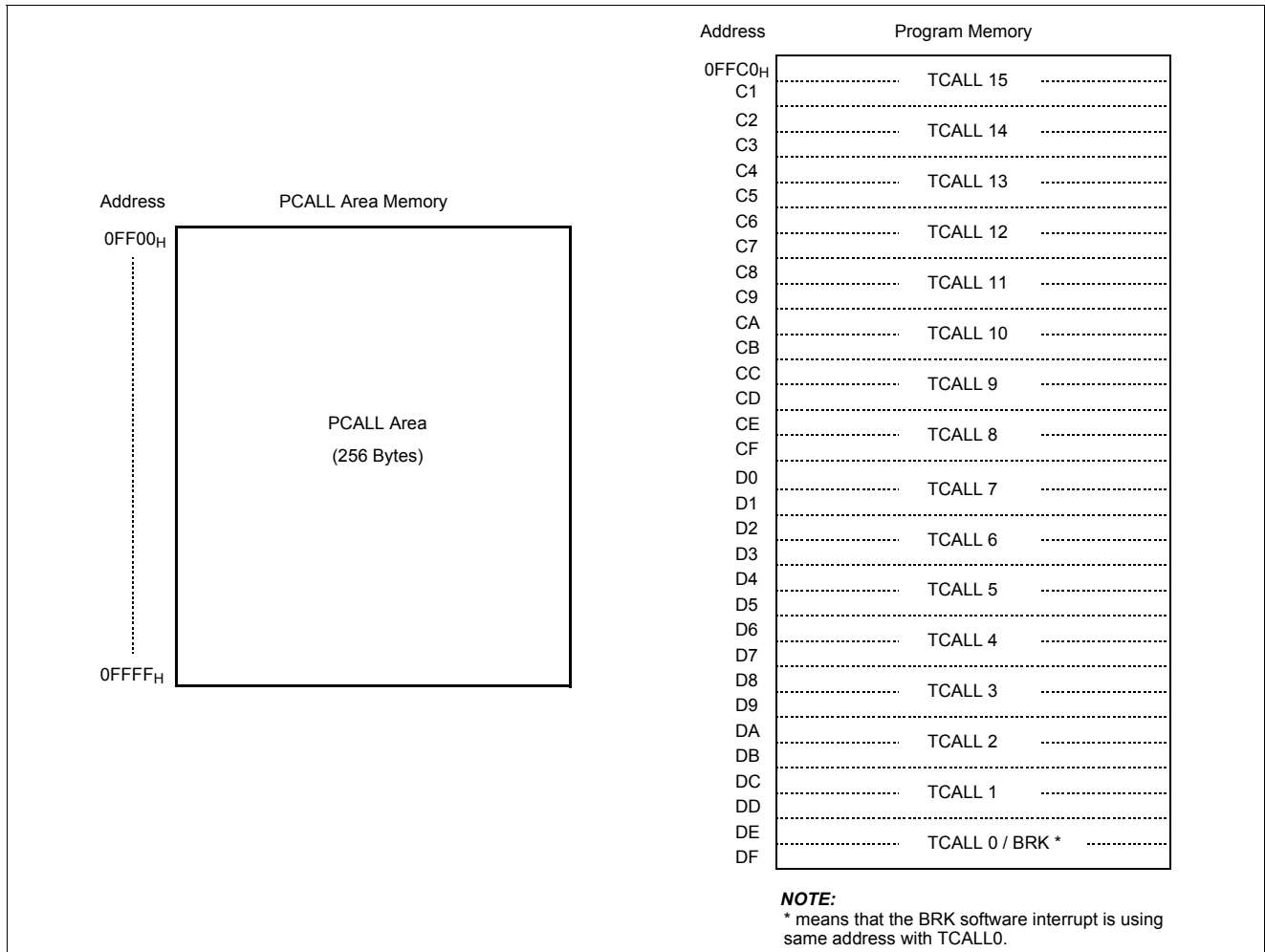
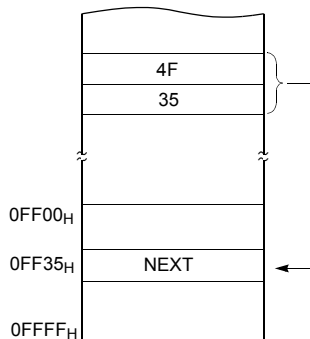


Figure 8-7 PCALL and TCALL Memory Area

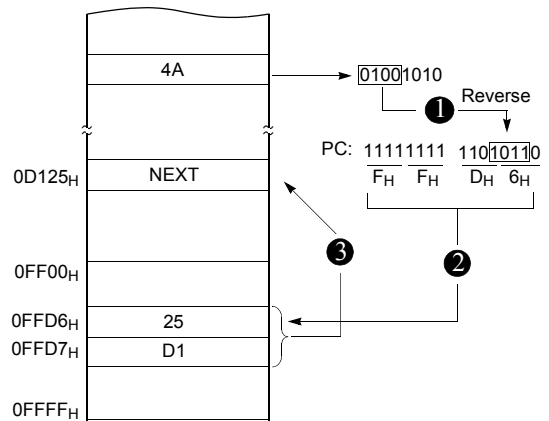
**PCALL → rel**

4F35 PCALL 35<sub>H</sub>



**TCALL → n**

4A TCALL 4



Example: The usage software example of Vector address and the initialize part.

```

ORG 0FFE0H          ; Device : MC81F8816

DW  WT_INT          ; Watch Timer / Watch Dog Timer
DW  BIT_INT         ; Basic Interval Timer
DW  ADC_I2C_INT     ; AD converter / I2C
DW  TMR3_INT       ; Timer-3
DW  TMR2_INT       ; Timer-2
DW  TMR1_INT       ; Timer-1
DW  TMR0_INT       ; Timer-0
DW  SPI            ; SPI
DW  NOT_USED       ; Not used
DW  UART0_INT      ; UART TX0, RX0
DW  EX3_INT        ; INT.3
DW  EX2_INT        ; INT.2
DW  EX1_INT        ; INT.1
DW  EX0_INT        ; INT.0
DW  NOT_USED       ; Not used
DW  RESET          ; Reset

;*****
;          MAIN          PROGRAM          *
;*****

ORG 0C000H

RESET:  DI          ;Disable All Interrupts
        CLRG
        LDX #0
        LDA #0
RAM_CLR1:STA {X}+;Page0 RAM Clear(!0000H->!009FH)
        CMPX #090H
        BNE RAM_CLR1

        LDM RPR,#0000_0001B;Page1 RAM Clear(!0100H->!00FFH)
        SETG
        LDX #0
        LDA #0
RAM_CLR2:STA {X}+
        CMPX #060H
        BNE RAM_CLR2
        CLRG

        LDX #0FFH;Stack Pointer Initialize
        TXSP

        LDM RPR,#0000_0000B;Page0 selection
        CALL LCD_CLR;Clear LCD display memory

        LDM R0, #0;Normal Port 0
        LDM R0IO,#1000_0010B;Normal Port Direction
        LDM R0PU,#1000_0010B;Pull Up Selection Set
        LDM R0OD,#0000_0001B;R0 port Open Drain control
        :
        :
        LDM SCMR,#1111_0000B;System clock control
        :
        :

```

### 8.3 Data Memory

Figure 8-8 shows the internal Data Memory space available. Data Memory is divided into four groups, a user RAM, control registers, Stack, and LCD memory.

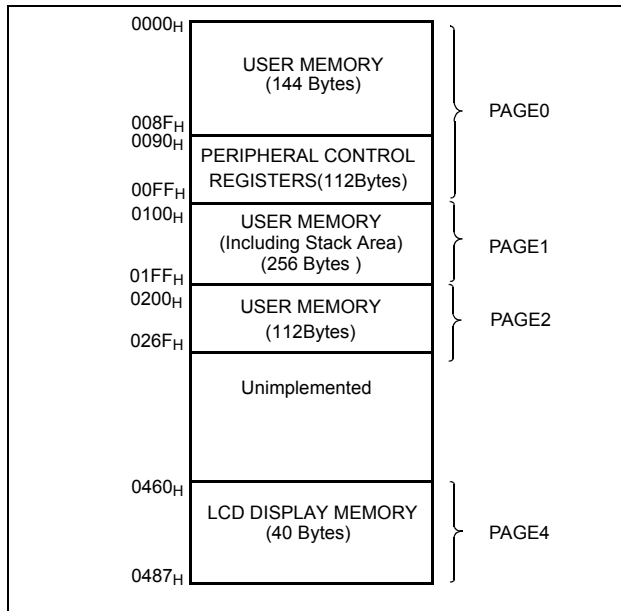


Figure 8-8 Data Memory Map

#### User Memory

The MC81F8816/8616 have  $256 \times 8$  bits for the user data memory (RAM). There are three pages internal RAM. Page is selected by G-flag and RAM page selection register RPR. When G-flag is cleared to “0”, always page 0 is selected regardless of RPR value. If G-flag is set to “1”, page will be selected according to RPR value.

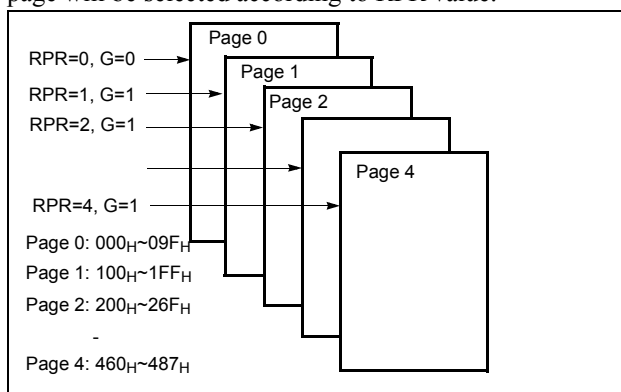


Figure 8-9 RAM page configuration

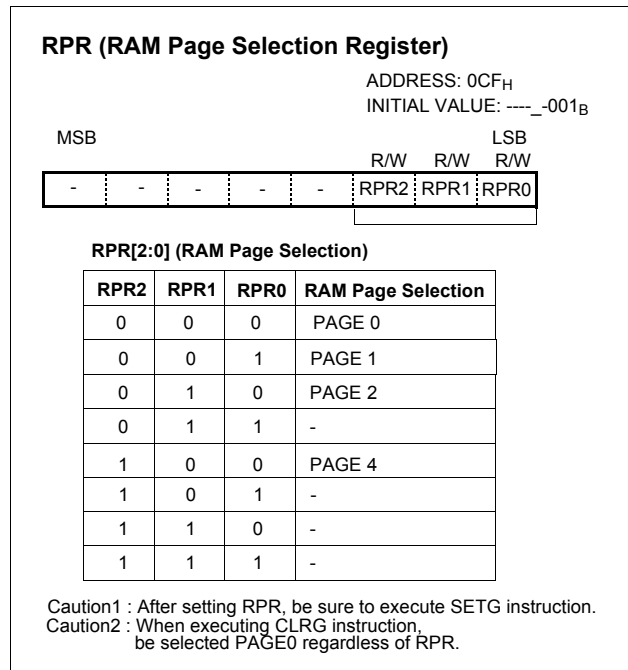


Figure 8-10 RAM Page Selection Register

#### Control Registers

The control registers are used by the CPU and Peripheral function blocks for controlling the desired operation of the device. Therefore these registers contain control and status bits for the interrupt system, the timer/counters, analog to digital converters and I/O ports. The control registers are in address range of 090H to 0FFH.

Note that unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

More detailed informations of each register are explained in each peripheral section.

**Note:** Write only registers can not be accessed by bit manipulation instruction. Do not use read-modify-write instruction. Use byte manipulation instruction.

Example; To write at CKCTRL

```
LDM CKCTRL, #05H ;Divide ratio +8
```

#### Stack Area

The stack provides the area where the return address is saved before a jump is performed during the processing

routine at the execution of a subroutine call instruction or the acceptance of an interrupt.

When returning from the processing routine, executing the subroutine return instruction [RET] restores the contents of the program counter from the stack; executing the interrupt return instruction [RETI] restores the contents of the program counter and flags.

The save/restore locations in the stack are determined by

the stack pointer (SP). The SP is automatically decreased after the saving, and increased before the restoring. This means the value of the SP indicates the stack location number for the next save. Refer to Figure 8-4 on page 34.

### LCD Display Memory

LCD display data area is handled in LCD section.

See "18.3 LCD Display Memory" on page 94.

Address	Register Name	Symbol	R/W	Initial Value								Addressing Mode
				7	6	5	4	3	2	1	0	
0090H	I2C Mode Control Register	I2CMR	R / W	0	0	0	0	1	0	0	0	byte
0091H	I2C Status Register	I2CSR	R	0	0	0	0	0	0	0	0	byte
0092H	I2C Clock Control Register	I2CCR	R / W	1	1	1	1	1	1	1	1	byte
0093H	I2C Pipe and Shift Register	I2CPR	R / W	1	1	1	1	1	1	1	1	byte
0094H	I2C Slave Address Register	I2CAR	R / W	0	0	0	0	0	0	0	0	byte
009AH	PLL Control Register	XPLLCR	R / W	0	0	0	0	0	0	0	0	byte
009BH	PLL Data Register	XPLLDAT	R / W	0	0	0	0	0	0	0	0	byte
009EH	WT Read Data Register	WTRH	R	-	0	0	0	0	0	0	0	byte
00A0H	R0 Open Drain Control Register	R0OD	W	0	0	0	0	0	0	0	0	byte <sup>1</sup>
00A1H	R1 Open Drain Control Register	R1OD	W	0	0	0	0	0	0	0	0	byte
00A2H	R2Open Drain Control Register	R2OD	W	0	0	0	0	0	0	0	0	byte
00A4H	R4Open Drain Control Register	R4OD	W	0	0	0	0	0	0	0	0	byte
00A5H	R0 Pull-up Register	R0PU	W	0	0	0	0	0	0	0	0	byte
00A6H	R1 Pull-up Register	R1PU	W	0	0	0	0	0	0	0	0	byte
00A7H	R2 Pull-up Register	R2PU	W	0	0	0	0	0	0	0	0	byte
00A9H	R4 Pull-up Register	R4PU	W	0	0	0	0	0	0	0	0	byte
00AAH	Port Selection Register 0	PSR0	W	0	0	0	0	-	-	0	0	byte
00ABH	Port Selection Register 1	PSR1	W	-	-	-	-	-	0	0	0	byte
00ACH	R5 Port Selection Register	R5PSR	R / W	1	1	1	1	1	1	1	1	byte, bit <sup>2</sup>
00ADH	R6 Port Selection Register	R6PSR	R / W	1	1	1	1	1	1	1	1	byte, bit
00AEH	R7 Port Selection Register	R7PSR	R / W	1	1	1	1	1	1	1	1	byte, bit
00AFH	R8 Port Selection Register	R8PSR	R / W	1	1	1	1	1	1	1	1	byte, bit(EVA only)
00B0H	R7 Data Register	R7	R / W	0	0	0	0	0	0	0	0	byte
00B2H	LCD Control Register	LCR	R / W	0	0	0	0	0	0	0	0	byte, bit
00B3H	LCD BIAS Control Register	LBCR	R / W	0	1	1	1	1	0	0	0	byte, bit
00B4H	R7 Direction Register	R7IO	W	0	0	0	0	0	0	0	0	byte
00B6H	SPI Mode Control Register	SPIM	R / W	0	0	0	0	0	0	0	1	byte
00B7H	SPI Data Shift Register	SPIR	R / W	-	-	-	-	-	-	-	-	byte

**Table 8-1 Control Registers**

Address	Register Name	Symbol	R/W	Initial Value								Addressing Mode
				7	6	5	4	3	2	1	0	
00B8H	Asynchronous Serial Mode Register	ASIMR	R / W	0	0	0	0	-	0	0	-	byte, bit
00B9H	Asynchronous Serial Status Register	ASISR	R	-	-	-	-	-	0	0	0	byte
00BAH	Baud Rate Generator Control Register	BRGCR	R / W	-	0	0	1	0	0	0	0	byte, bit
00BBH	Receive Buffer Register	RXBR	R	0	0	0	0	0	0	0	0	byte
	Transmit Shift Register	TXSR	W	1	1	1	1	1	1	1	1	byte
00C0H	R0 port data register	R0	R / W	0	0	0	0	0	0	0	0	byte, bit
00C1H	R0 Direction Register	R0IO	W	0	0	0	0	0	0	0	0	byte
00C2H	R1 port data register	R1	R / W	0	0	0	0	0	0	0	0	byte, bit
00C3H	R1 Direction Register	R1IO	W	0	0	0	0	0	0	0	0	byte
00C4H	R2 port data register	R2	R / W	0	0	0	0	0	0	0	0	byte, bit
00C5H	R2 Direction Register	R2IO	W	0	0	0	0	0	0	0	0	byte
00C8H	R4 port data register	R4	R / W	0	0	0	0	0	0	0	0	byte, bit
00C9H	R4 Direction Register	R4IO	W	0	0	0	0	0	0	0	0	byte
00CAH	R5 port data register	R5	R/W	0	0	0	0	0	0	0	0	byte, bit
00CBH	R5 Direction Register	R5IO	W	0	0	0	0	0	0	0	0	byte
00CCH	R6 port data register	R6	R/W	0	0	0	0	0	0	0	0	byte, bit
00CDH	R6 Direction Register	R6IO	W	0	0	0	0	0	0	0	0	byte
00CEH	Buzzer Driver Register	BUZR	W	1	1	1	1	1	1	1	1	byte
00CFH	Ram Page Selection Register	RPR	R / W	-	-	-	-	-	0	0	1	byte, bit
00D0H	Timer 0 Mode Control Register	TM0	R / W	-	-	0	0	0	0	0	0	byte, bit
00D1H	Timer 0 Register	T0	R	0	0	0	0	0	0	0	0	byte
	Timer 0 Data Register	TDR0	W	1	1	1	1	1	1	1	1	byte
	Timer 0 Capture Data Register	CDR0	R	0	0	0	0	0	0	0	0	byte
00D2	Timer 1 Mode Control Register	TM1	R / W	0	0	0	0	0	0	0	0	byte, bit
00D3H	Timer 1 Data Register	TDR1	W	1	1	1	1	1	1	1	1	byte
00D4H	Timer 1 Register	T1	R	0	0	0	0	0	0	0	0	byte
	Timer 1 PWM Duty Register	T1PDR	R/W	1	1	1	1	1	1	1	1	byte
	Timer 1 Capture Data Register	CDR1	R	0	0	0	0	0	0	0	0	byte
00D5H	Timer 1 PWM High Register	T1PWHR	R / W	-	-	-	-	0	0	0	0	byte
00D6H	Timer 2 Mode Control Register	TM2	R / W	-	-	0	0	0	0	0	0	byte, bit
00D7H	Timer 2 Register	T2	R	0	0	0	0	0	0	0	0	byte
	Timer 2 Data Register	TDR2	W	1	1	1	1	1	1	1	1	byte
	Timer 2 Capture data Register	CDR2	R	0	0	0	0	0	0	0	0	byte
00D8H	Timer 3 Mode Control Register	TM3	R / W	0	0	0	0	0	0	0	0	byte, bit
00D9H	Timer 3 Data Register	TDR3	W	1	1	1	1	1	1	1	1	byte
	Timer 3 PWM Period Register	T3PPR	W	1	1	1	1	1	1	1	1	byte

Table 8-1 Control Registers

Address	Register Name	Symbol	R/W	Initial Value								Addressing Mode	
				7	6	5	4	3	2	1	0		
00DAH	Timer 3 Register	T3	R	0	0	0	0	0	0	0	0	0	byte
	Timer 3 PWM Duty Register	T3PDR	R / W	0	0	0	0	0	0	0	0	0	byte, bit
	Timer 3 Capture Data Register	CDR3	R	0	0	0	0	0	0	0	0	0	byte
00DBH	Timer 3 PWM High Register	T3PWHR	W	-	-	-	-	0	0	0	0	0	byte
00E2H	10bit A/D Converter Mode Control Register	ADCM <sup>3</sup>	R / W	0	0	0	0	0	0	0	0	1	byte, bit
00E3H	10bit A/D Converter Result Register Low	ADCRL	R	Undefined								byte	
00E4H	10bit A/D Converter Result Register High	ADCRH	W, R	0	1	0	-	-	-	X	X	X	byte, bit
00E5H	BOD Control Register	BODR	R / W	0	0	0	0	0	0	0	0	0	byte, bit
00E6H	Basic Interval Timer Register	BITR	R	Undefined								byte	
	Clock Control Register	CKCTLR	W	-	-	0	1	0	1	1	1	1	byte
00E7H	System Clock Mode Register	SCMR	R / W	-	-	-	-	-	0	0	0	0	byte
00E8H	Watch Dog Timer Register	WDTR	W	0	1	1	1	1	1	1	1	1	byte
	Watch Dog Timer Data Register	WDTDR	R	Undefined								byte	
	Watch Timer Register	WTR	W	0	1	1	1	1	1	1	1	1	byte
00E9H	Stop & Sleep Mode Control Register	SSCR	W	0	0	0	0	0	0	0	0	0	byte
00EAH	Watch Timer Mode Register	WTMR	R / W	0	0	-	-	0	0	0	0	0	byte, bit
00F4H	Interrupt Generation Flag Register High	INTFH	R / W	-	-	-	0	0	0	-	-	-	byte, bit
00F5H	Interrupt Generation Flag Register Low	INTFL	R / W	0	0	-	-	0	0	0	0	0	byte, bit
00F6H	Interrupt Enable Register High	IENH	R / W	-	0	0	0	0	0	0	-	-	byte, bit
00F7H	Interrupt Enable Register Middle	IENM	R / W	0	0	0	0	-	-	-	0	0	byte, bit
00F8H	Interrupt Enable Register Low	IENL	R / W	0	0	0	0	0	0	0	-	-	byte, bit
00F9H	Interrupt Request Register High	IRQH	R / W	-	0	0	0	0	0	0	-	-	byte, bit
00FAH	Interrupt Request Register Middle	IRQM	R / W	0	0	0	0	-	-	-	0	0	byte, bit
00FBH	Interrupt Request Register Low	IRQL	R / W	0	0	0	0	0	0	0	-	-	byte, bit
00FCH	Interrupt Edge Selection Register	IEDS	R / W	0	0	0	0	0	0	0	0	0	byte, bit

**Table 8-1 Control Registers**

1. "byte", "bit" means that register can be addressed by not only bit but byte manipulation instruction.
2. "byte" means that register can be addressed by only byte manipulation instruction. On the other hand, do not use any read-modify-write instruction such as bit manipulation.
3. bit 0 of ADCM is read only.

### 8.4 Addressing Mode

The MC81F8816/8616 use six addressing modes;

- Register addressing
- Immediate addressing
- Direct page addressing
- Absolute addressing
- Indexed addressing
- Register-indirect addressing

#### (1) Register Addressing

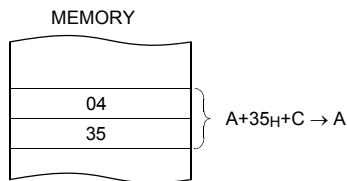
Register addressing accesses the A, X, Y, C and PSW.

#### (2) Immediate Addressing → #imm

In this mode, second byte (operand) is accessed as a data immediately.

Example:

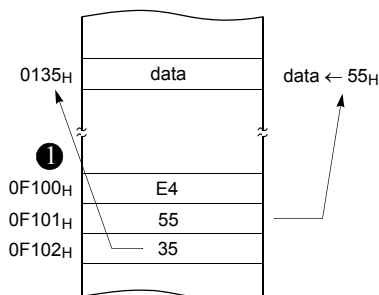
```
0435   ADC   #35H
```



When G-flag is 1, then RAM address is defined by 16-bit address which is composed of 8-bit RAM paging register (RPR) and 8-bit immediate data.

Example: G=1, RPR=01H

```
E45535  LDM   35H, #55H
```

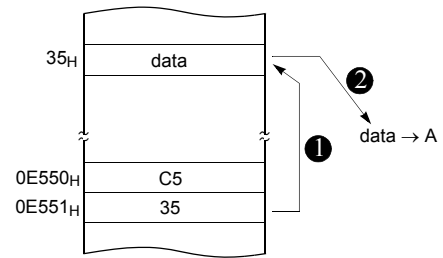


#### (3) Direct Page Addressing → dp

In this mode, a address is specified within direct page.

Example; G=0

```
C535   LDA   35H           ; A ← RAM[35H]
```



#### (4) Absolute Addressing → !abs

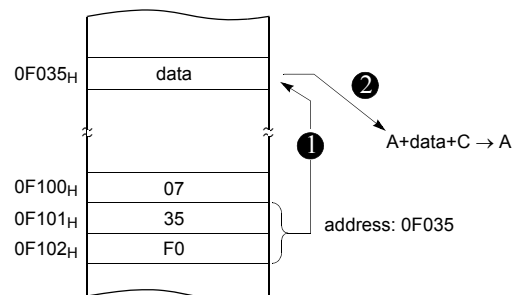
Absolute addressing sets corresponding memory data to Data, i.e. second byte (Operand I) of command becomes lower level address and third byte (Operand II) becomes upper level address.

With 3 bytes command, it is possible to access to whole memory area.

ADC, AND, CMP, CMPX, CMPY, EOR, LDA, LDX, LDY, OR, SBC, STA, STX, STY

Example;

```
0735F0  ADC   !0F035H     ; A ← ROM[0F035H]
```

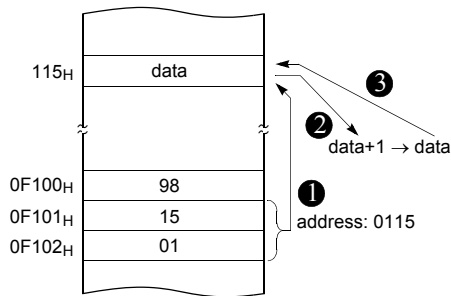




The operation within data memory (RAM)  
ASL, BIT, DEC, INC, LSR, ROL, ROR

Example; Addressing accesses the address 0135H regardless of G-flag and RPR.

```
981501 INC !0115H ; A ←ROM[115H]
```



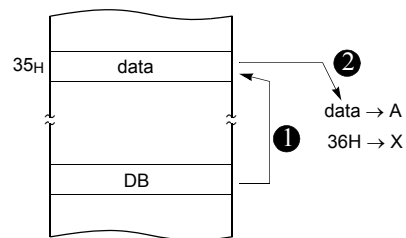
**X indexed direct page, auto increment → {X}+**

In this mode, a address is specified within direct page by the X register and the content of X is increased by 1.

LDA, STA

Example; G=0, X=35H

```
DB LDA {X}+
```



**(5) Indexed Addressing**

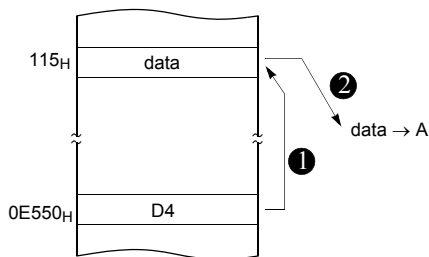
**X indexed direct page (no offset) → {X}**

In this mode, a address is specified by the X register.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA, XMA

Example; X=15H, G=1, RPR=01H

```
D4 LDA {X} ; ACC←RAM[X].
```



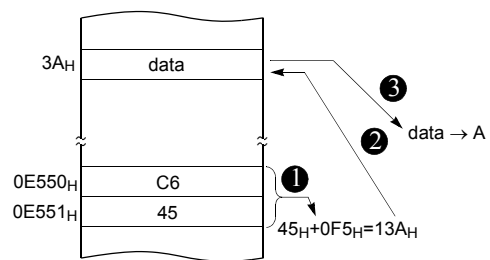
**X indexed direct page (8 bit offset) → dp+X**

This address value is the second byte (Operand) of command plus the data of X-register. And it assigns the memory in Direct page.

ADC, AND, CMP, EOR, LDA, LDY, OR, SBC, STA  
STY, XMA, ASL, DEC, INC, LSR, ROL, ROR

Example; G=0, X=0F5H

```
C645 LDA 45H+X
```



**Y indexed direct page (8 bit offset) → dp+Y**

This address value is the second byte (Operand) of command plus the data of Y-register, which assigns Memory in Direct page.

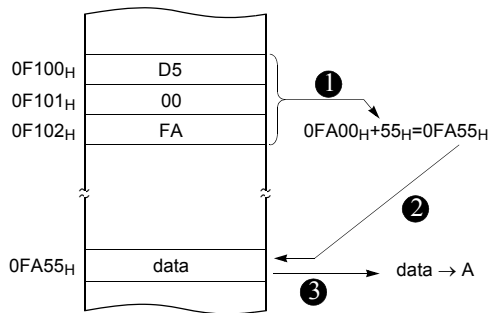
This is same with above. Use Y register instead of X.

**Y indexed absolute → !abs+Y**

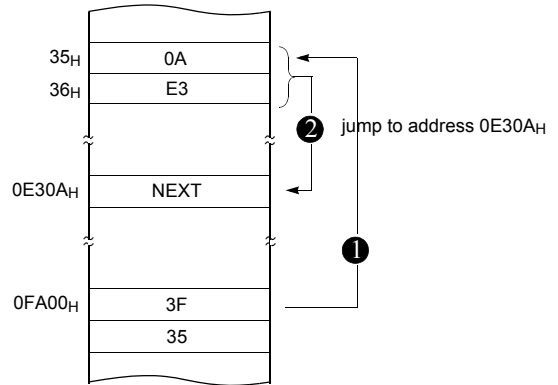
Sets the value of 16-bit absolute address plus Y-register data as Memory. This addressing mode can specify memory in whole area.

Example; Y=55H

```
D500FA LDA !0FA00H+Y
```



```
3F35 JMP [35H]
```



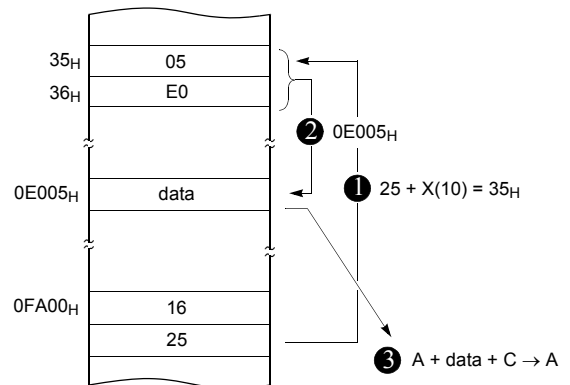
**X indexed indirect → [dp+X]**

Processes memory data as Data, assigned by 16-bit pair memory which is determined by pair data [dp+X+1][dp+X] Operand plus X-register data in Direct page.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA

Example; G=0, X=10H

```
1 625 ADC [25H+X]
```



**(6) Indirect Addressing**

**Direct page indirect → [dp]**

Assigns data address to use for accomplishing command which sets memory data (or pair memory) by Operand. Also index can be used with Index register X,Y.

JMP, CALL

Example; G=0

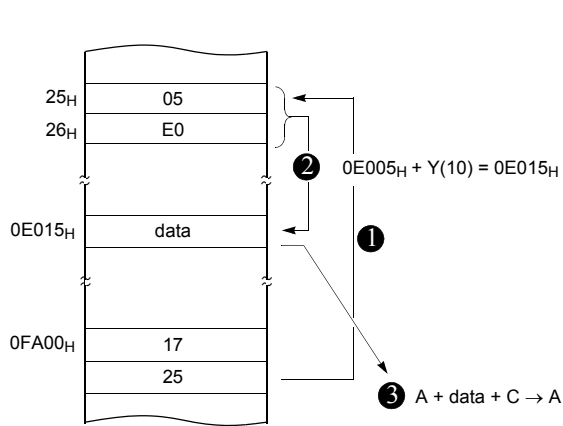
**Y indexed indirect → [dp]+Y**

Processes memory data as Data, assigned by the data [dp+1][dp] of 16-bit pair memory paired by Operand in Direct page plus Y-register data.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA

Example; G=0, Y=10H

1725    ADC    [25H]+Y



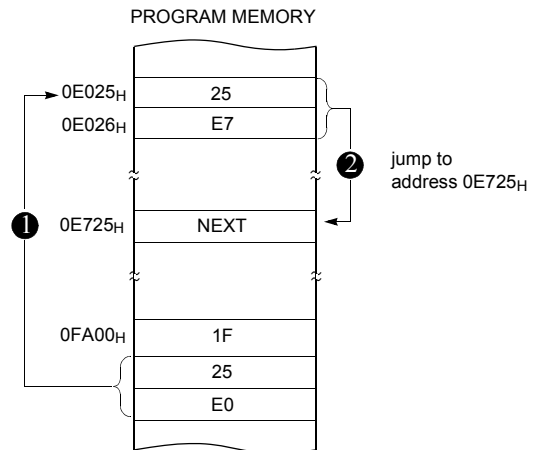
**Absolute indirect → [!abs]**

The program jumps to address specified by 16-bit absolute address.

JMP

Example; G=0

1F25E0    JMP    [!0E025H]



## 9. I/O PORTS

The MC81F8816/8616 have seven I/O ports, LCD segment ports (R0, R1, R2, R4, R50/SEG00 ~ R77/SEG23, SEG24 ~ SEG35) and LCD common ports (SEG39/COM4 ~ SEG36/COM7, COM0~COM3).

These ports pins may be multiplexed with an alternate

### 9.1 Registers for Ports

#### Port Data Registers

The Port Data Registers are represented as a D-Type flip-flop, which will clock in a value from the internal bus in response to a “write to data register” signal from the CPU. The Q output of the flip-flop is placed on the internal bus in response to a “read data register” signal from the CPU. The level of the port pin itself is placed on the internal bus in response to “read data register” signal from the CPU. Some instructions that read a port activating the “read register” signal, and others activating the “read pin” signal.

#### Port Direction Registers

All pins have data direction registers which can define these ports as output or input. A “1” in the port direction register configure the corresponding port pin as output. Conversely, write “0” to the corresponding bit to specify it as input pin. For example, to use the even numbered bit of R0 as output ports and the odd numbered bits as input ports, write “55H” to address 0C1H (R0 port direction register) during initial setting as shown in Figure 9-1.

All the port direction registers in the MC81F8816/8616 have 0 written to them by reset function. Therefore, its initial status is input.

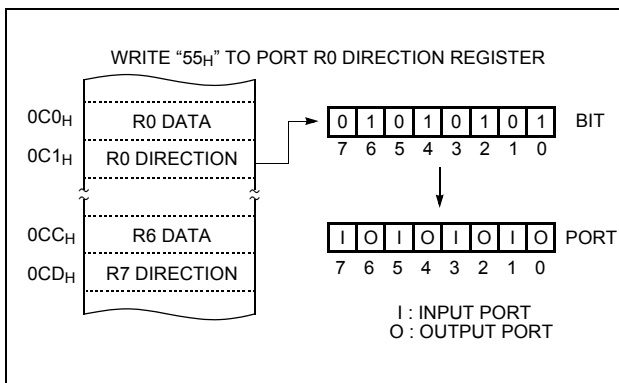


Figure 9-1 Example of port I/O assignment

#### Pull-up Control Registers

The R0, R1, R2 and R4 ports have internal pull-up resistors. Figure 9-2 shows a functional diagram of a typical

function for the peripheral features on the device.

**Note:** SEG28 ~ SEG35 are not supported in MC81F8616Q(64pin).

pull-up port. It is connected or disconnected by Pull-up Control register (RnPU). The value of that resistor is typically 100kΩ. Refer to DC characteristics for more details.

When a port is used as key input, input logic is firmly either low or high, therefore external pull-down or pull-up resistors are required practically. The MC81F8816/8616 have internal pull-up, it can be logic high by pull-up that can be able to configure either connect or disconnect individually by pull-up control registers RnPU.

When ports are configured as inputs and pull-up resistor is selected by software, they are pulled to high.

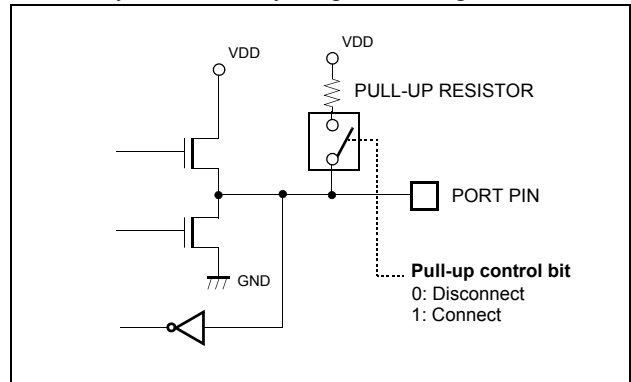


Figure 9-2 Pull-up Port Structure

#### Open drain port Registers

The R0, R1, R2 and R4 ports have open drain port resistors R0OD~R4OD.

Figure 9-3 shows an open drain port configuration by control register. It is selected as either push-pull port or open drain port by R0OD, R1OD, R2OD and R4OD.

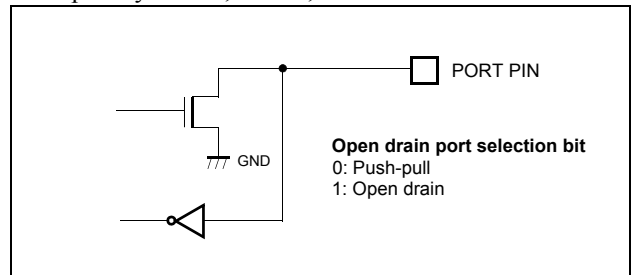


Figure 9-3 Open-drain Port Structure

## 9.2 I/O Ports Configuration

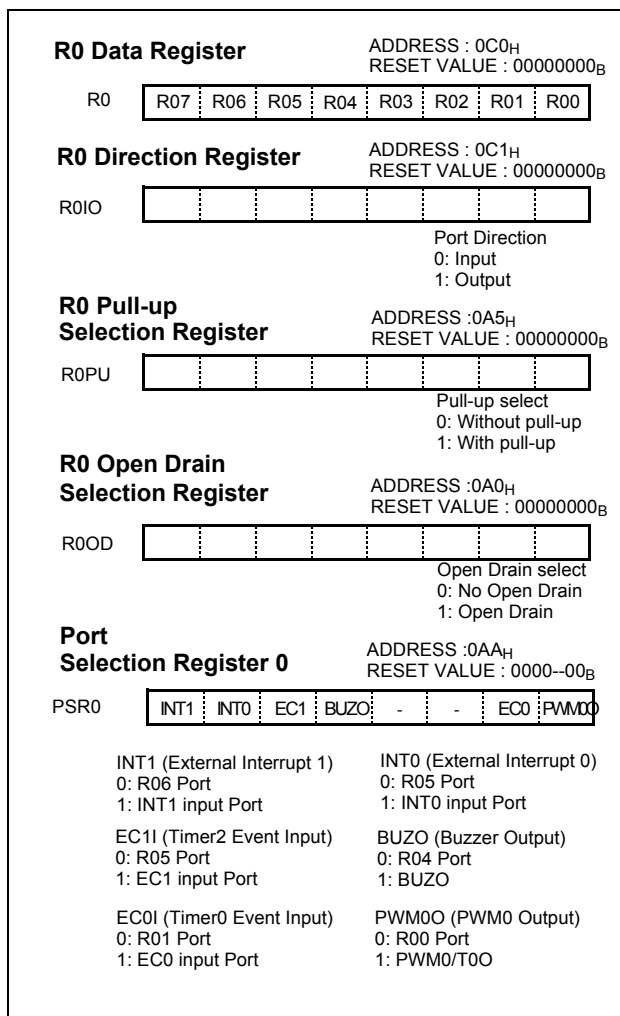
### R0 Port

R0 is a 8-bit CMOS bidirectional I/O port (address 0C0<sub>H</sub>). Each I/O pin can independently used as an input or an output through the R0IO register (address 0C1<sub>H</sub>).

R0 has internal pull-ups that is independently connected or disconnected by R0PU. The control registers for R0 are shown below.

In addition, Port R0 is multiplexed with various special features. The control register PSR0 (address 0AA<sub>H</sub>) controls the selection of alternate function. After reset, this value is “0”, port may be used as normal I/O port.

To use alternate function such as External Interrupt rather than normal I/O, write “1” in the corresponding bit of PSR0.



Port pin	Alternate function
R00	PWM0/T00 (Timer1 PWM Output / Timer0 Output)
R01	EC0 (Timer 0 Event Count Input)
R04	BUZO (Buzzer Output)
R05	EC1 / INT0 (Timer2 Event Count Input/External Interrupt 0 Request Input)
R06	INT1 (External Interrupt 1 Request Input)
R07	INT2 (External Interrupt 2 Request Input)

**Note:** R0IO, R0PU, P0OD and PSR0 are write-only registers. They can not be read and can not be accessed by bit manipulation instruction. Do not use read or read-modify-write instruction. Use byte manipulation instruction.

### R1 Ports

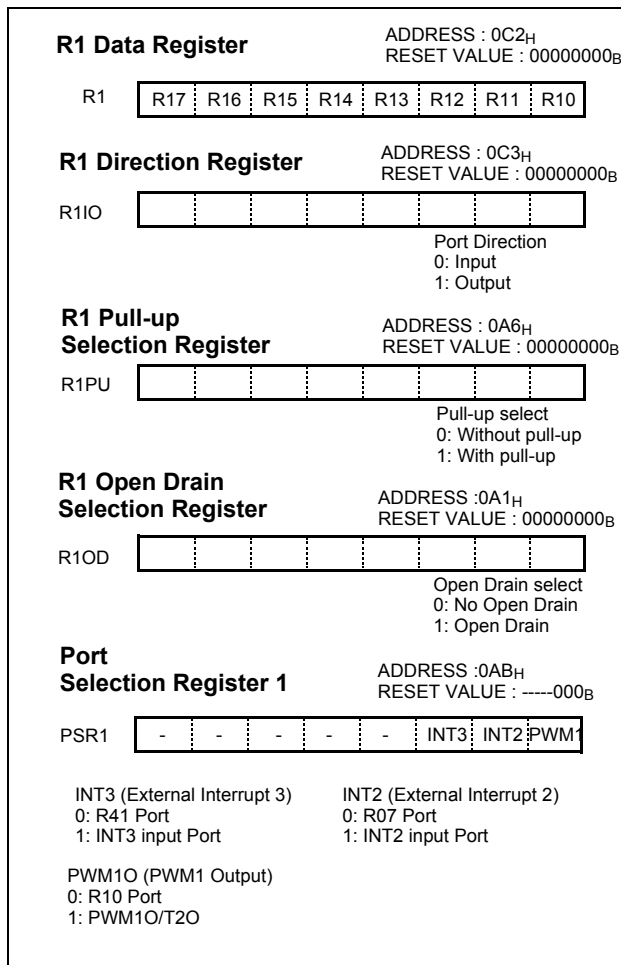
R1 is an 8-bit CMOS bidirectional I/O port (address 0C2<sub>H</sub>). Each I/O pin can independently used as an input or an output through the R1IO register (address 0C3<sub>H</sub>).

R1 has internal pull-up that is independently connected or disconnected by register R1PU. The control registers for R1 are shown below.

In addition, Port R1 is multiplexed with various special features. The control register PSR1 (address 0AB<sub>H</sub>) controls the selection of alternate function. After reset, this value is “0”, port may be used as normal I/O port.

To use alternate function such as External Interrupt rather than normal I/O, write “1” in the corresponding bit of PSR1.

**Note:** R1IO, R1PU, P1OD and PSR1 are write-only registers. They can not be read and can not be accessed by bit manipulation instruction. Do not use read or read-modify-write instruction. Use byte manipulation instruction.



## R2 Ports

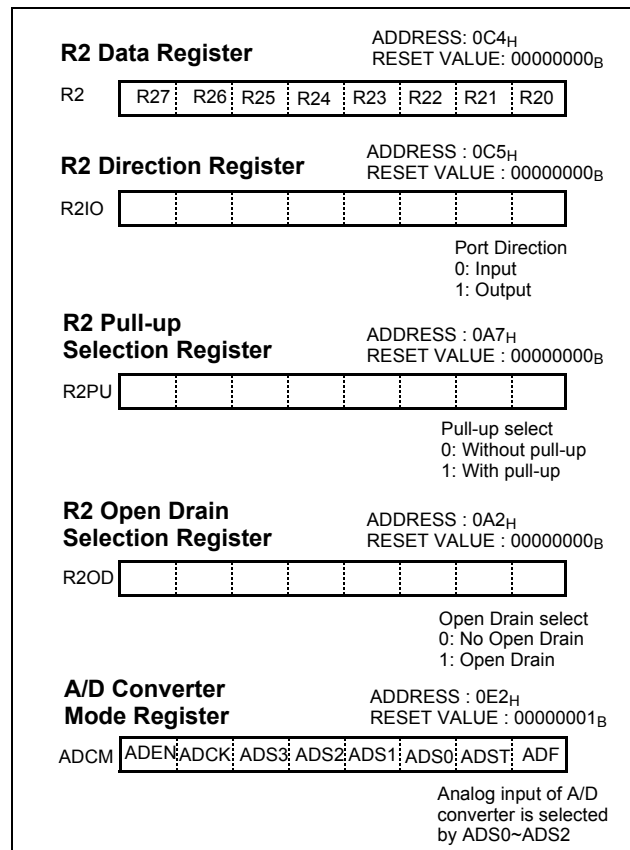
R2 is a 8/5-bit CMOS bidirectional I/O port (address 0C4<sub>H</sub>). Each I/O pin can independently used as an input or an output through the R2IO register (address 0C5<sub>H</sub>).

R2 has internal pull-ups that are independently connected or disconnected by R2PU (address 0A7<sub>H</sub>). The control registers for R2 are shown as below.

**Note:** R2IO, R2PU and P2OD are write-only registers. They can not be read and can not be accessed by bit ma-

nipulation instruction. Do not use read or read-modify-write instruction. Use byte manipulation instruction.

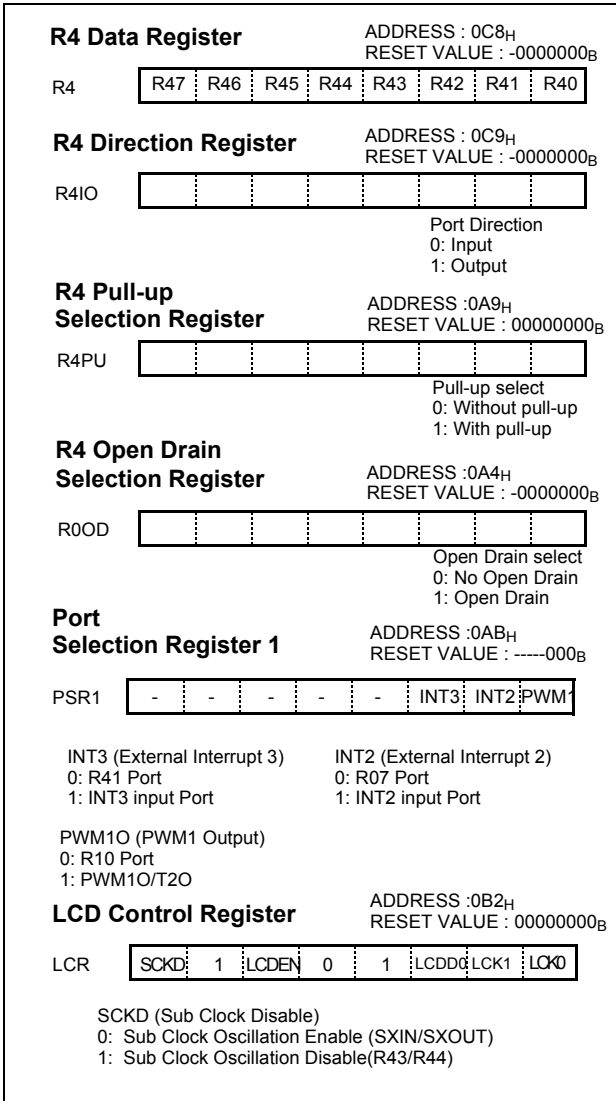
**Note:** The R25 and R27 are not supported in the MC81F8616Q.



## R4 Port

R4 is a 8-bit CMOS bidirectional I/O port (address 0C8<sub>H</sub>). Each I/O pin can independently used as an input or an output through the R4IO register (address 0C9<sub>H</sub>).

R4 has internal pull-ups that is independently connected or disconnected by R4PU. The control registers for R4 are shown below.



In addition, Port R4 is multiplexed with oscillation input/output, reset and interrupt input pins. The control register PSR1 (address 0AB<sub>H</sub>) controls the selection of alternate function. After reset, this value is “0”, port may be used as normal I/O port. To use alternate function such as External Interrupt rather than normal I/O, write “1” in the corresponding bit of PSR1.

Main oscillation input/output and reset pin can be used as normal I/O ports (R46/R45) and normal input port(R47) by selecting configuration options in flash writing. Sub oscillation input/output pin can be used as normal I/O ports by

writing “1” to the SCKD bit of the LCR register

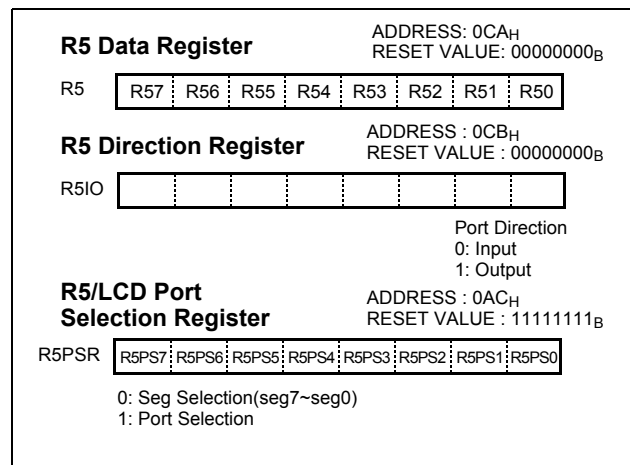
Port pin	Alternate function
R40	-
R41	INT3 (External Interrupt 3 Request input)
R42	-
R43	SX <sub>IN</sub>
R44	SX <sub>OUT</sub>
R45	X <sub>IN</sub>
R46	X <sub>OUT</sub>
R47	RESET

**Note:** R4IO, R4PU, P4OD and PSR1 are write-only registers. They can not be read and can not be accessed by bit manipulation instruction. Do not use read or read-modify-write instruction. Use byte manipulation instruction.

### R5 Ports

R5 is an 8-bit CMOS bidirectional I/O port (address 0CA<sub>H</sub>). Each I/O pin can independently used as an input or an output through the R5IO register (address 0CB<sub>H</sub>).

R5 is multiplexed with LCD segment output(SEG0 ~ SEG7), which can be selected by writing appropriate value into the R5PSR(address 0AC<sub>H</sub>).



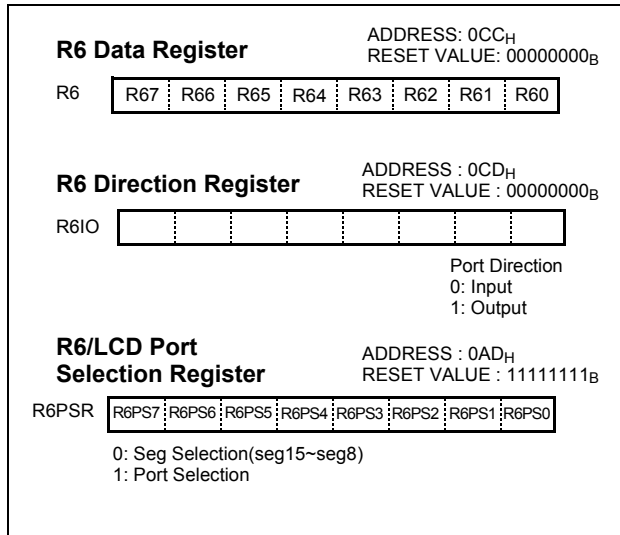
**Note:** R5IO is write-only register. It can not be read and can not be accessed by bit manipulation instruction. Do not use read or read-modify-write instruction. Use byte manipulation instruction.

### R6 Ports

R6 is an 8-bit CMOS bidirectional I/O port (address 0CC<sub>H</sub>). Each I/O pin can independently used as an input or

an output through the R6IO register (address 0CD<sub>H</sub>).

R6 is multiplexed with LCD segment output(SEG8 ~ SEG15), which can be selected by writing appropriate value into the R6PSR(address 0AD<sub>H</sub>).

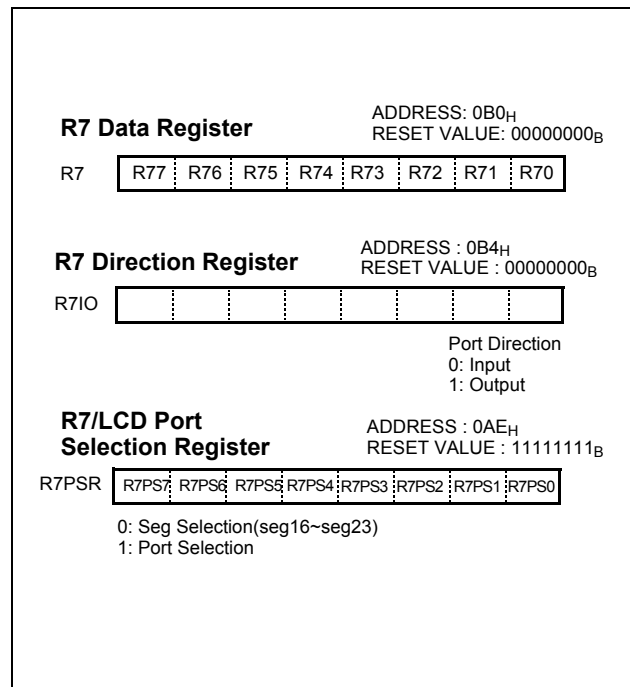


**Note:** R6IO is write-only register. It can not be read and can not be accessed by bit manipulation instruction. Do not use read or read-modify-write instruction. Use byte manipulation instruction.

## R7 Ports

R7 is a 8-bit CMOS bidirectional I/O port (address 0B0<sub>H</sub>). Each I/O pin can independently used as an input or an output through the R7IO register (address 0B4<sub>H</sub>).

R7 is multiplexed with LCD segment output(SEG16 ~ SEG23), which can be selected by writing appropriate value into the R7PSR(address 0AE<sub>H</sub>).



**Note:** R7IO is write-only register. It can not be read and can not be accessed by bit manipulation instruction. Do not use read or read-modify-write instruction. Use byte manipulation instruction.

## SEG0~SEG35

Segment signal output pins for the LCD display. See "18. LCD DRIVER" on page 88 for details.

SEG24 ~ SEG31 is multiplexed with normal I/O port(EVA chip), which can be selected by writing appropriate value into the R8PSR(address 0AF<sub>H</sub>).

## COM0~COM7

Common signal output pins for the LCD display. See "18. LCD DRIVER" on page 88 for details.

SEG36~SEG39 and COM7~COM4 are selected by LCDD of the LCR register.



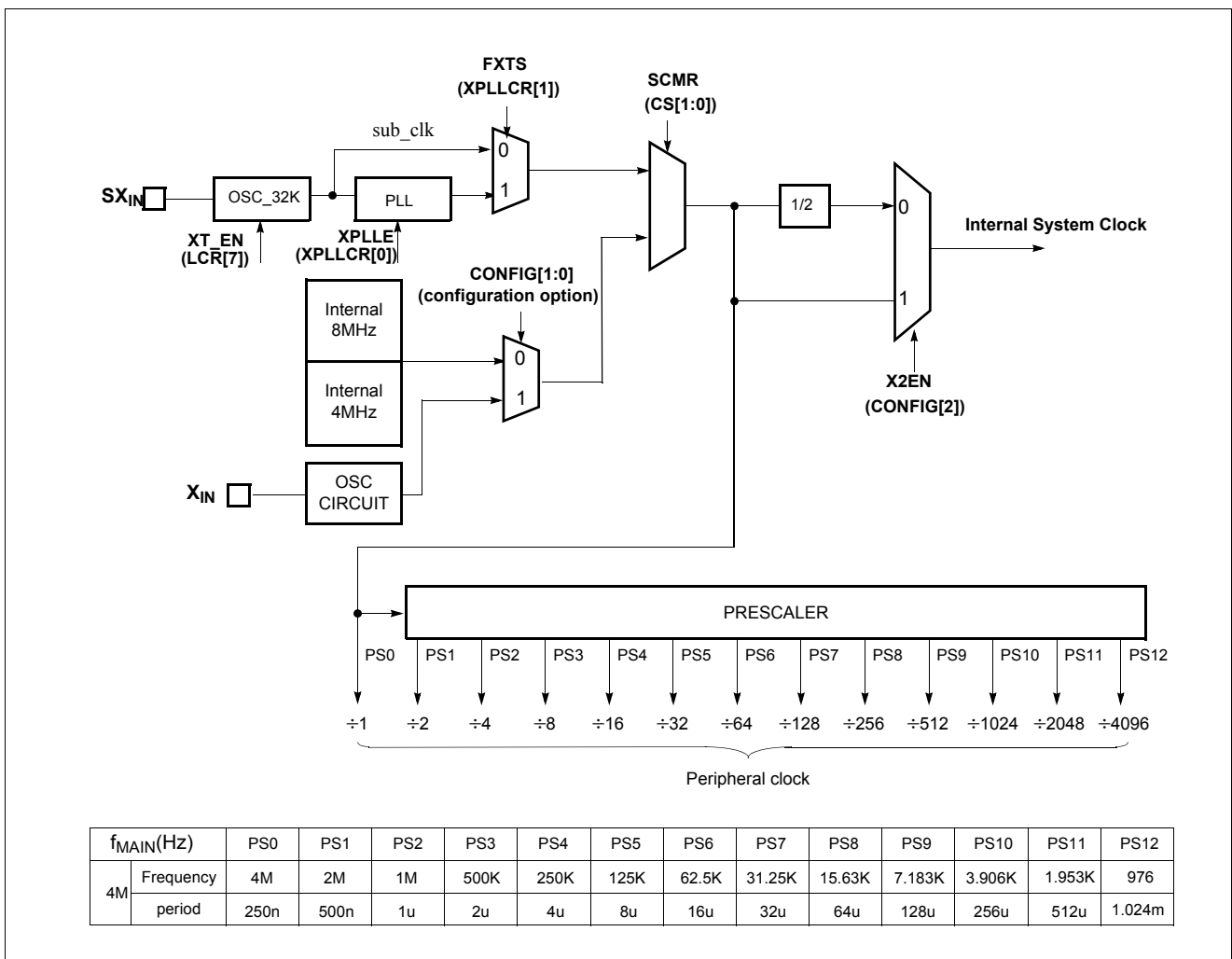
## 10. CLOCK GENERATOR

As shown in Figure 10-1, 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 two oscillators which are main-frequency clock oscillator and a sub-frequency clock oscillator. The system clock can also be obtained from the external oscillator. By setting configuration option, the internal 8MHz, 4MHz can also be selected for system clock source.

The clock generator produces the system clocks forming clock pulse, which are supplied to the CPU and the peripheral hardware.

The internal system clock should be selected to main oscillation by setting bit1 and bit0 of the system clock mode register (SCMR). The registers are shown in Figure 10-2.

To the peripheral block, the clock among the not-divided original clocks, divided by 2, 4,..., up to 4096 can be provided. Peripheral clock is enabled or disabled by STOP instruction. The peripheral clock is controlled by clock control register (CKCTRL). See "11. BASIC INTERVAL TIMER" on page 53 for details.



**Figure 10-1 Block Diagram of Clock Generator**

The SCMR should be set to operate by main oscillation. Bit2, bit1 and bit0 of the SCMR should be set to “000” or “001” to select main oscillation.

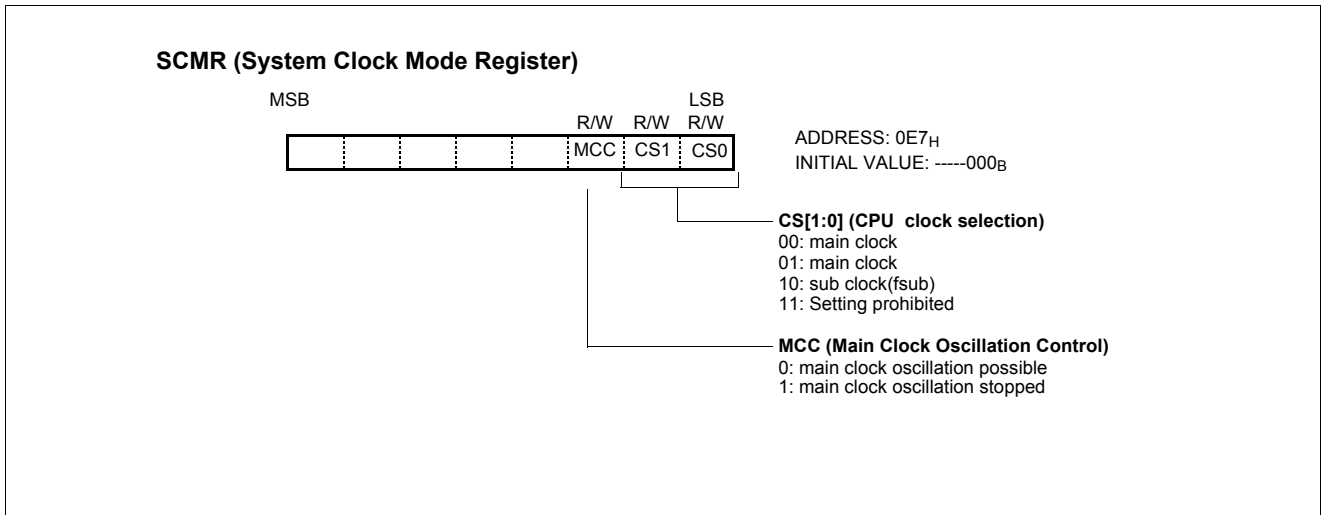


Figure 10-2 SCMR System Clock Mode Register

### 11. BASIC INTERVAL TIMER

The MC81F8816/8616 have one 8-bit Basic Interval Timer that is free-run and can not stop. Block diagram is shown in Figure 11-1.

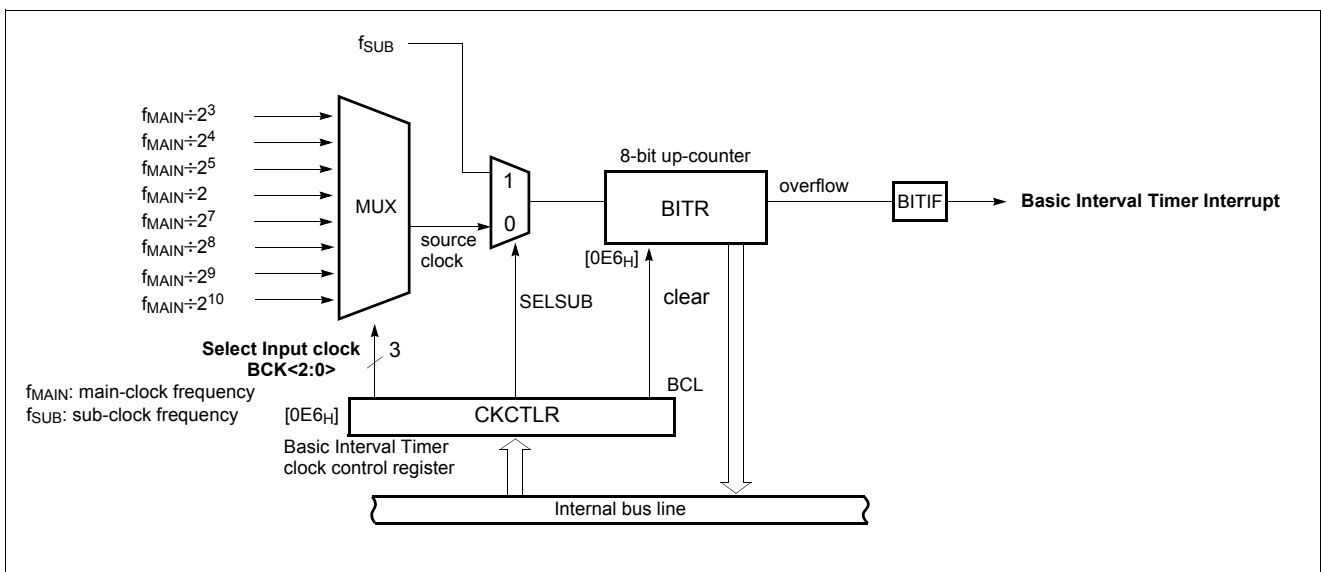
The Basic Interval Timer Register (BITR) is increased every internal count pulse which is divided by prescaler. Since prescaler has division ratio from 8 to 1024, the count rate is 1/8 to 1/1024 of the oscillator frequency. After reset, the BCK bits are all set, so the longest oscillation stabilization time is obtained.

It also provides a Basic interval timer interrupt (BITF). The count overflow of BITR from FF<sub>H</sub> to 00<sub>H</sub> causes the

interrupt to be generated. The Basic Interval Timer is controlled by the clock control register (CKCTLR) shown in Figure 11-2.

Source clock can be selected by lower 3 bits of CKCTLR. When write “1” to bit BCL of CKCTLR, BITR register is cleared to “0” and restart to count up. The bit BCL becomes “0” automatically after one machine cycle by hardware.

BITR and CKCTLR are located at same address, and address 0E6<sub>H</sub> is read as a BITR, and written to CKCTLR.



**Figure 11-1 Block Diagram of Basic Interval Timer**

BCK<2:0>	Source clock	Interrupt (overflow) Period
	SCMR[1:0] = 00 or 01	At f <sub>MAIN</sub> = 4MHz
000	f <sub>MAIN</sub> ÷2 <sup>3</sup>	0.512 ms
001	f <sub>MAIN</sub> ÷2 <sup>4</sup>	1.024
010	f <sub>MAIN</sub> ÷2 <sup>5</sup>	2.048
011	f <sub>MAIN</sub> ÷2 <sup>6</sup>	4.096
100	f <sub>MAIN</sub> ÷2 <sup>7</sup>	8.192
101	f <sub>MAIN</sub> ÷2 <sup>8</sup>	16.384
110	f <sub>MAIN</sub> ÷2 <sup>9</sup>	32.768
111	f <sub>MAIN</sub> ÷2 <sup>10</sup>	65.536

**Table 11-1 Basic Interval Timer Interrupt Time**

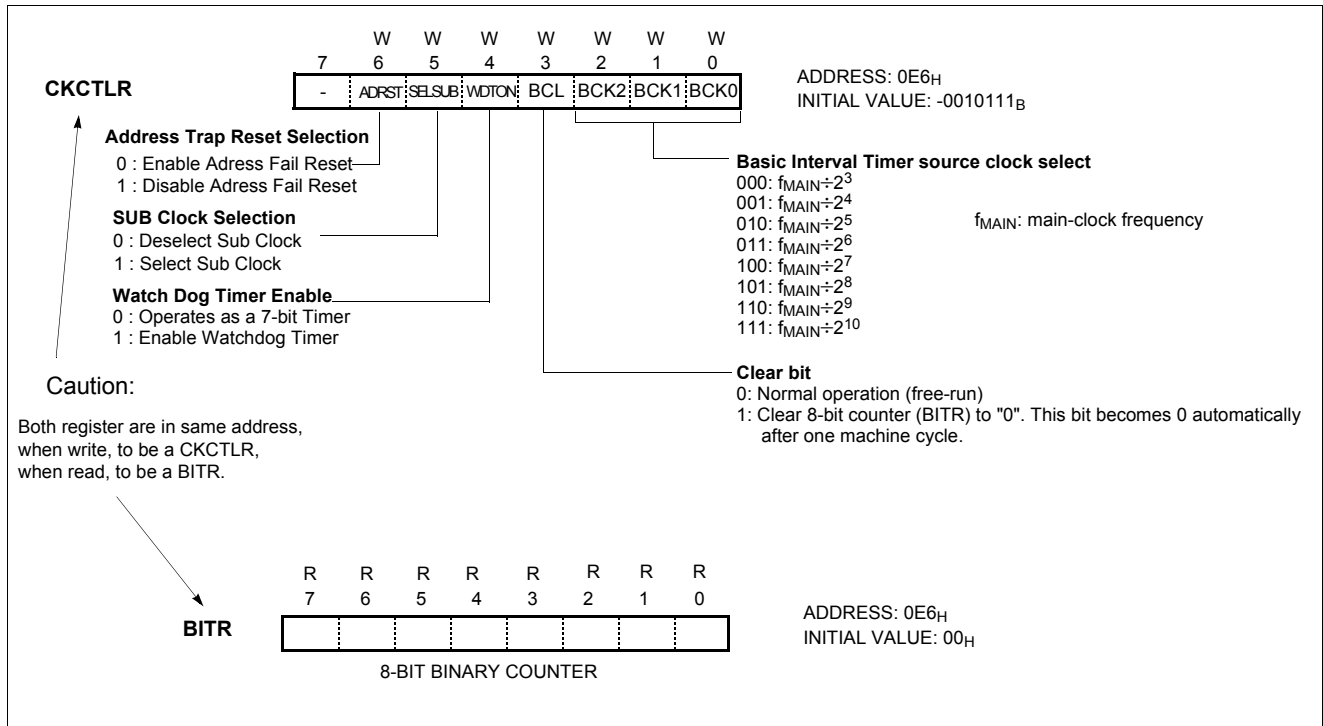


Figure 11-2 BITR: Basic Interval Timer Mode Register

**Example 1:**

Interrupt request flag is generated every 8.192ms at 4MHz.

```

:
LDM  CKCTLR, #0CH
SET1  BITE
EI
:
    
```

## 12. TIMER / COUNTER

Timer/Event Counter consists of prescaler, multiplexer, 8-bit timer data register, 8-bit counter register, mode register, input capture register and Comparator as shown in Figure 12-4. And the PWM high register for PWM is consisted separately.

The timer/counter 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

In the “timer” function, the register is increased every internal clock input. Thus, one can think of it as counting internal clock input. Since a least clock consists of 2 and

### Example 1:

Timer 0 = 8-bit timer mode, 8ms interval at 4MHz

Timer 1 = 8-bit timer mode, 4ms interval at 4MHz

```
LDM SCMR,#0 ;Main clock mode
LDM TDR0,#249
LDM TMO,#0001_0011B
LDM TDR1,#124
LDM TM1,#0000_1111B

SET1 TOE
SET1 T1E
EI
:
:
:
```

### Example 2:

Timer0 = 16-bit timer mode, 0.5s at 4MHz

```
LDM SCMR,#0 ;Main clock mode
LDM TDR0,#23H
LDM TDR1,#0F4H
LDM TMO,#0FH ;FMAIN/32, 8us
LDM TM1,#4CH

SET1 TOE
EI
:
:
:
```

most clock consists of 2048 oscillator periods, the count rate is 1/2 to 1/2048 of the oscillator frequency in Timer0. And Timer1 can use the same clock source too. In addition, Timer1 has more fast clock source (1/1 to 1/8).

In the “counter” function, the register is increased in response to a 0-to-1 (rising edge) transition at its corresponding external input pin EC0 (Timer 0).

In addition the “capture” function, the register is increased in response external or internal clock interrupt same with timer/counter function. When external interrupt edge input, the count register is captured into capture data register TMx.

Timer3 is shared with “PWM” function and Timer2 is shared with “Compare output” function.

### Example 3:

Timer0 = 8-bit event counter

Timer2 = 8-bit capture mode, 2us sampling count.

```
LDM TDR0,#0FFH ;don't care
LDM TMO,#1FH ;event counter
LDM ROIO,#1XXX_XX1XB ;R07, R01 input

LDM IEDS,#XXXX_01XXB ;FALLING
LDM PSR0,#1XXX_XX1XB ;INT1,EC0
LDM TDR2,#0FFH
LDM TM2,#0010_1011B ;2us

SET1 TOE ;ENABLE TIMER 0
SET1 T2E ;ENABLE TIMER 1
SET1 INT1E ;ENABLE INT1
EI
:
```

X: don't care.

### Example 4:

Timer0 = 16-bit capture mode, 8us sampling count. at 4MHz

```
LDM TDR0,#0FFH
LDM TDR1,#0FFH
LDM TMO,#2FH
LDM TM1,#5FH

LDM IEDS,#XXXX_XX01B
LDM PSR0,#X1XX_XXXXB ;AS INTO

SET1 TOE ;ENABLE TIMER 0
SET1 INTOE ;ENABLE EXT. INTO
EI
:
```

X: don't care.

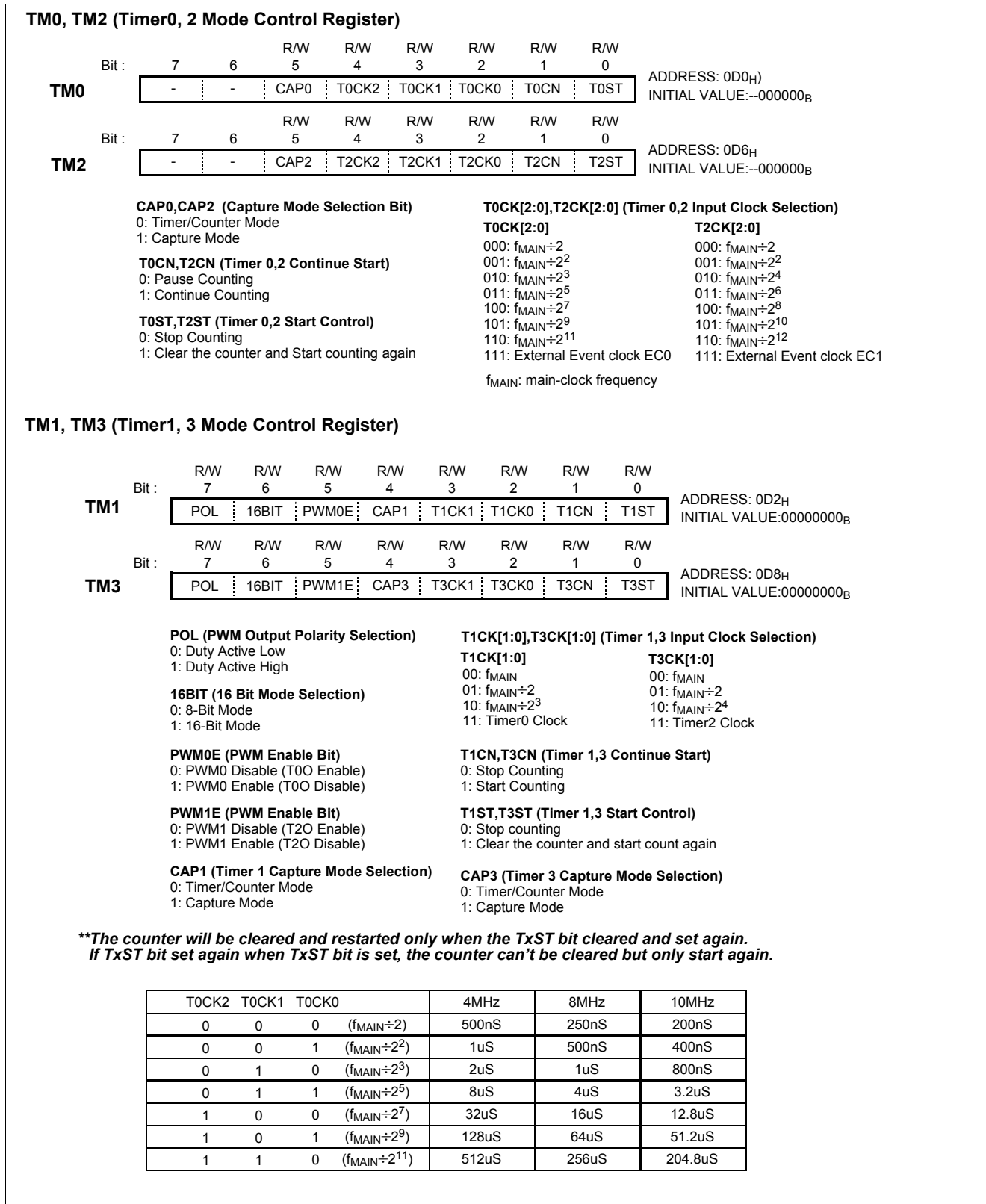
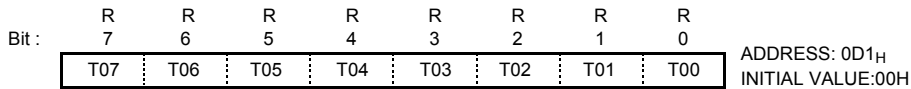
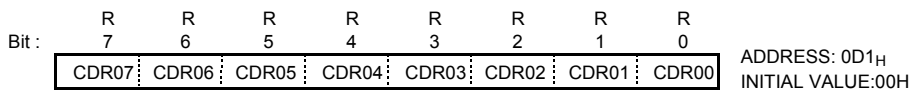


Figure 12-1 Timer0,1,2,3 Registers

**T0 (Timer0 Register)**

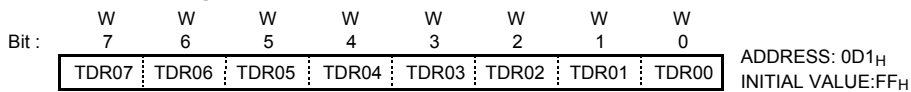


**CDR0 (Timer0 Input Capture Register)**



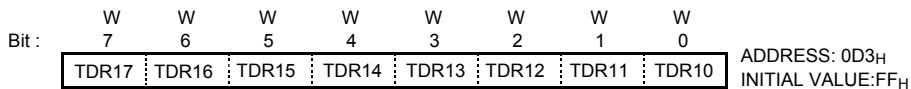
*In Timer mode, this register is the value of Timer 0 counter and in Capture mode, this register is the value of input capture.*

**TDR0 (Timer 0 Data Register)**



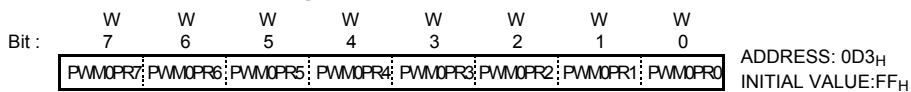
*If the counter of Timer 0 and the data of TDR0 is equal, interrupt is occurred.*

**TDR1 (Timer1 Data Register)**



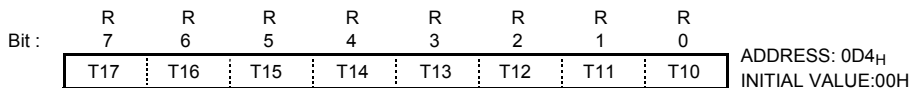
*If the counter of Timer 1 and the data of TDR1 is equal, interrupt is occurred.*

**T1PPR (Timer1 PWM Period Register)**

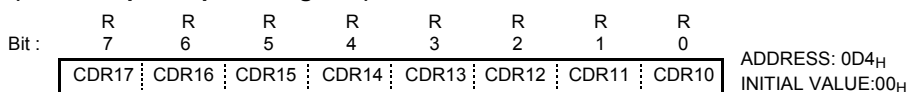


*The period is decided by PWM.*

**T1 (Timer1 Register)**

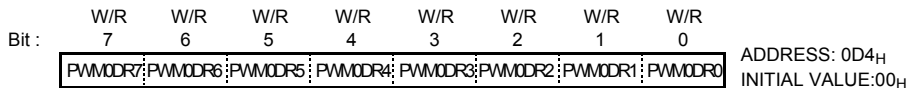


**CDR1 (Timer1 Input Capture Register)**



*In Timer mode, this register is the value of Timer 1 counter and in Capture mode, this register is the value of input capture.*

**T1PDR (Timer1 PWM0 Duty Register)**



*In PWM mode, decide the pulse duty.*

**T1PWHR (Timer1 PWM0 High Register)**

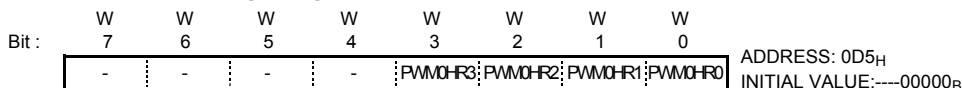


Figure 12-2 Related Registers with Timer/Counter0, 1

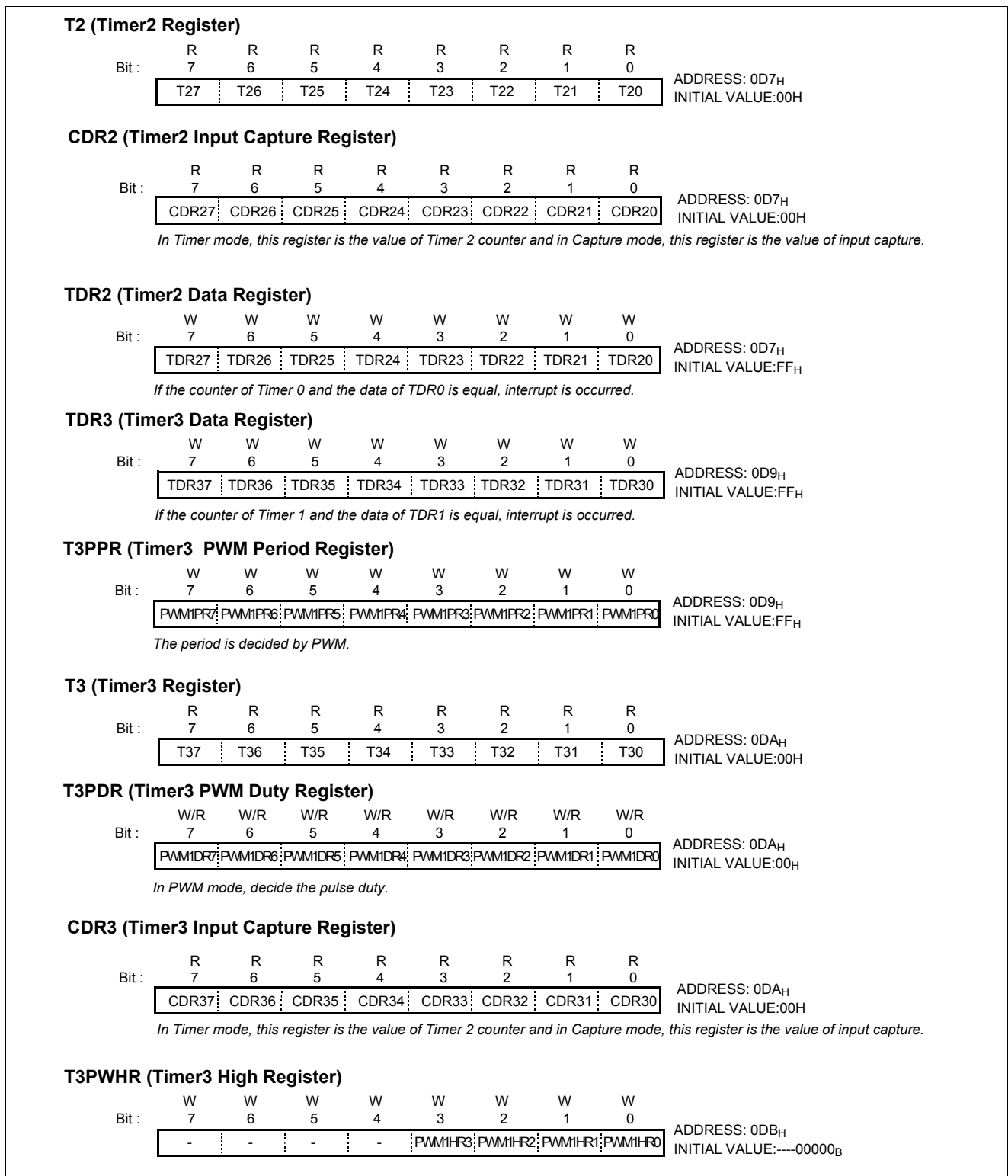


Figure 12-3 Related Registers with Timer/Counter2, 3



16BIT	CAP0	-	T0CK[2:0]	T1CK[1:0]	Timer 0	Timer 1
0	0	-	XXX	XX	8 Bit Timer	8 Bit Timer
0	0	-	111	XX	8 Bit Event Counter	8 Bit Timer
0	1	-	XXX	XX	8 Bit Capture	8 Bit Compare Output
1	0	-	XXX	11	16 Bit Timer	
1	0	-	111	11	16 Bit Event Counter	
1	1	-	XXX	11	16 Bit Capture	
1	0	-	XXX	11	16 Bit Compare Output	

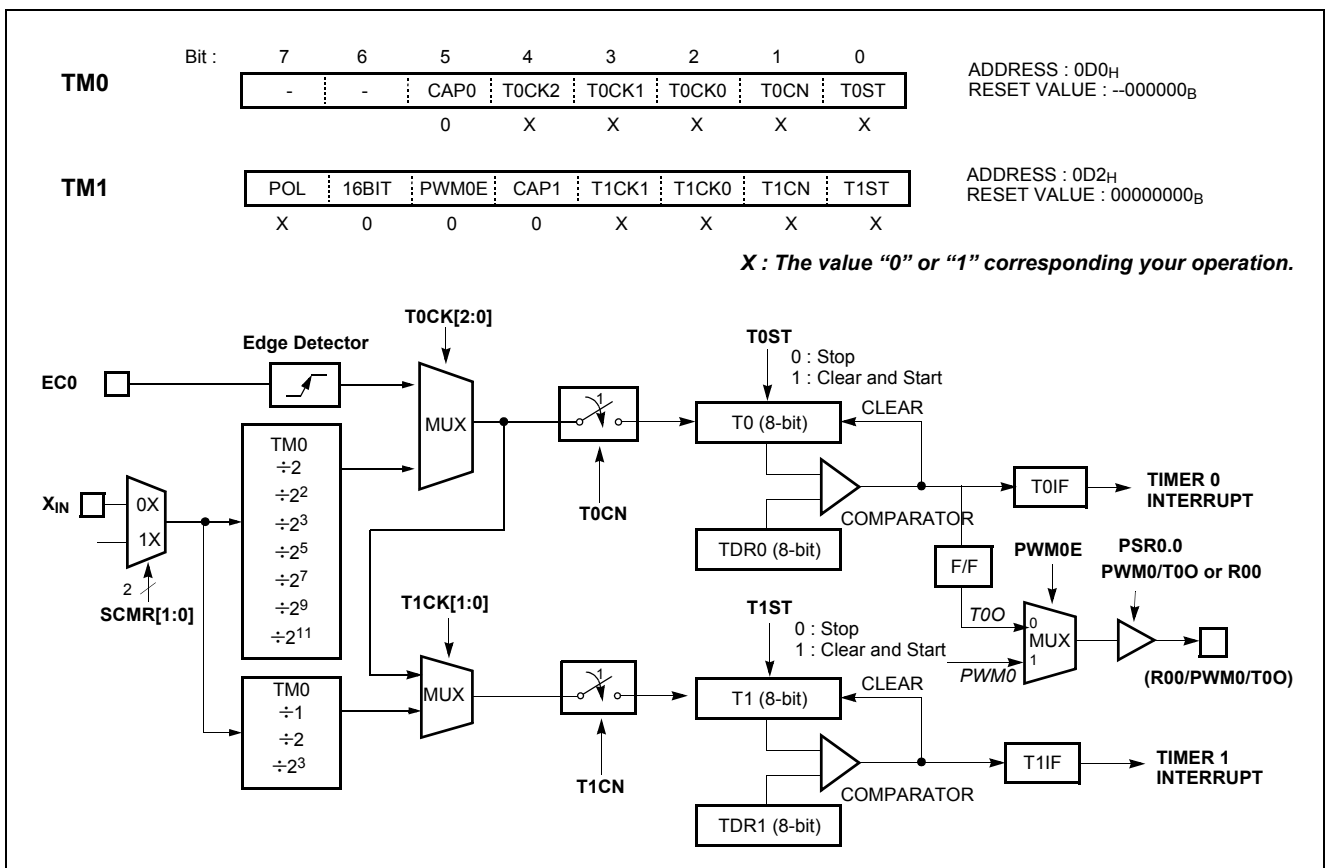
**Table 12-1 Operating Modes of Timer 0 and Timer 1**

**12.1 8-Bit Timer/Counter Mode**

The MC81F8816/8616 have four 8-bit Timer/Counters, Timer0, Timer1, Timer2 and Timer3 as shown in Figure 12-4.

The “timer” or “counter” function is selected by mode reg-

isters TMx (x=0,1,2,3) as shown in Figure 12-1 and Table 12-1. To use as an 8-bit timer/counter mode, bit CAPx of TMx is cleared to “0” and bits 16BIT of TM1(3) should be cleared to “0” (Table 12-1).



**Figure 12-4 Block Diagram of Timer/Event Counter0,1**

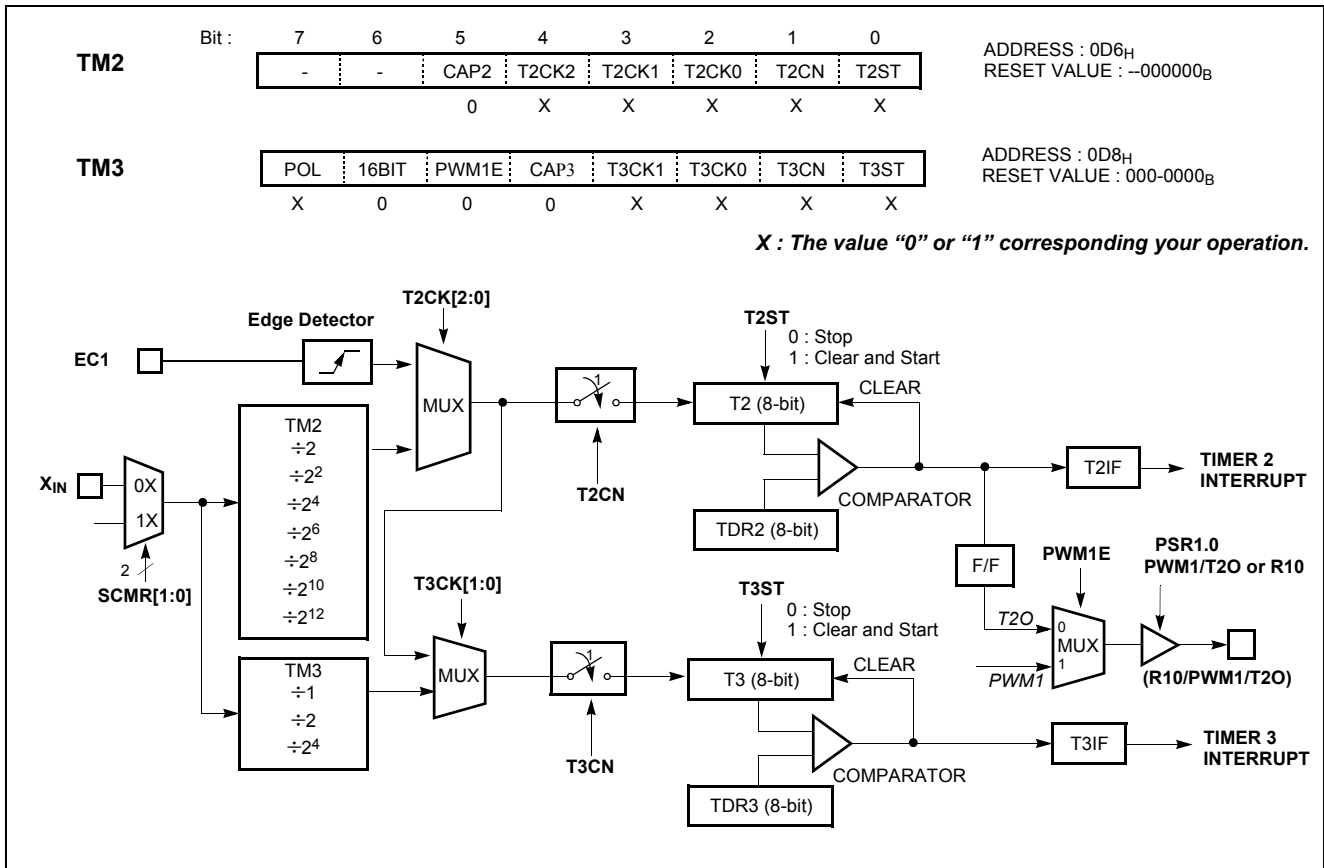


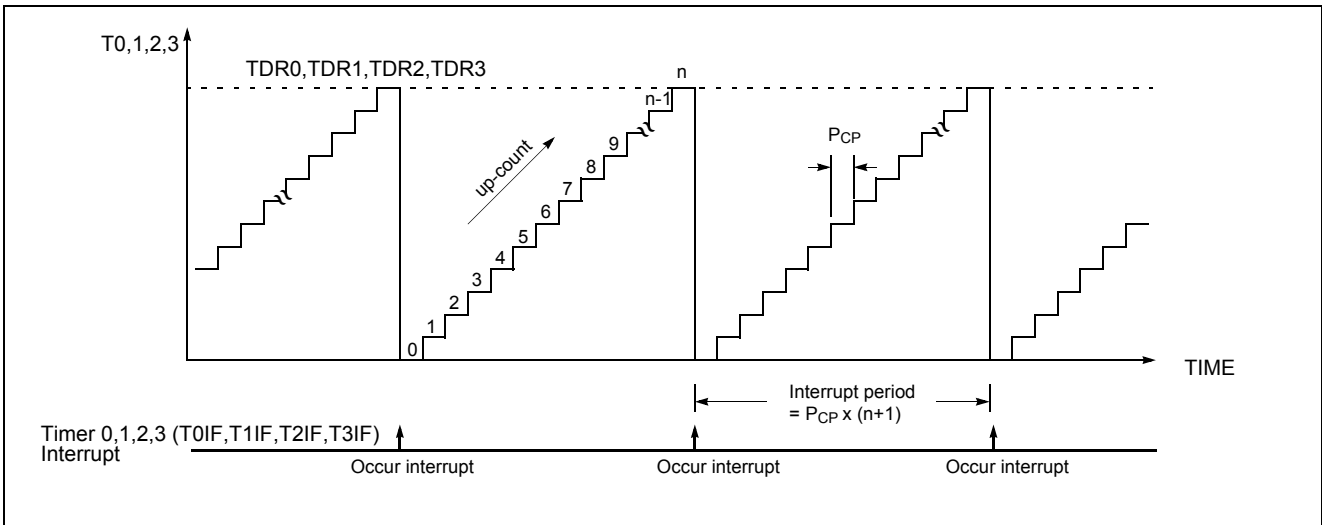
Figure 12-5 Block Diagram of Timer 2,3

These timers have each 8-bit count register and data register. The count register is increased by every internal or external clock input. The internal clock has a prescaler divide ratio option of 2, 4, 8, 32, 128, 512, 2048 (selected by control bits TxCK2, TxCK1 and TxCK0 of register TM0(2)) and 1, 2, 8 (selected by control bits TxCK1 and TxCK0 of register TM1(3)).

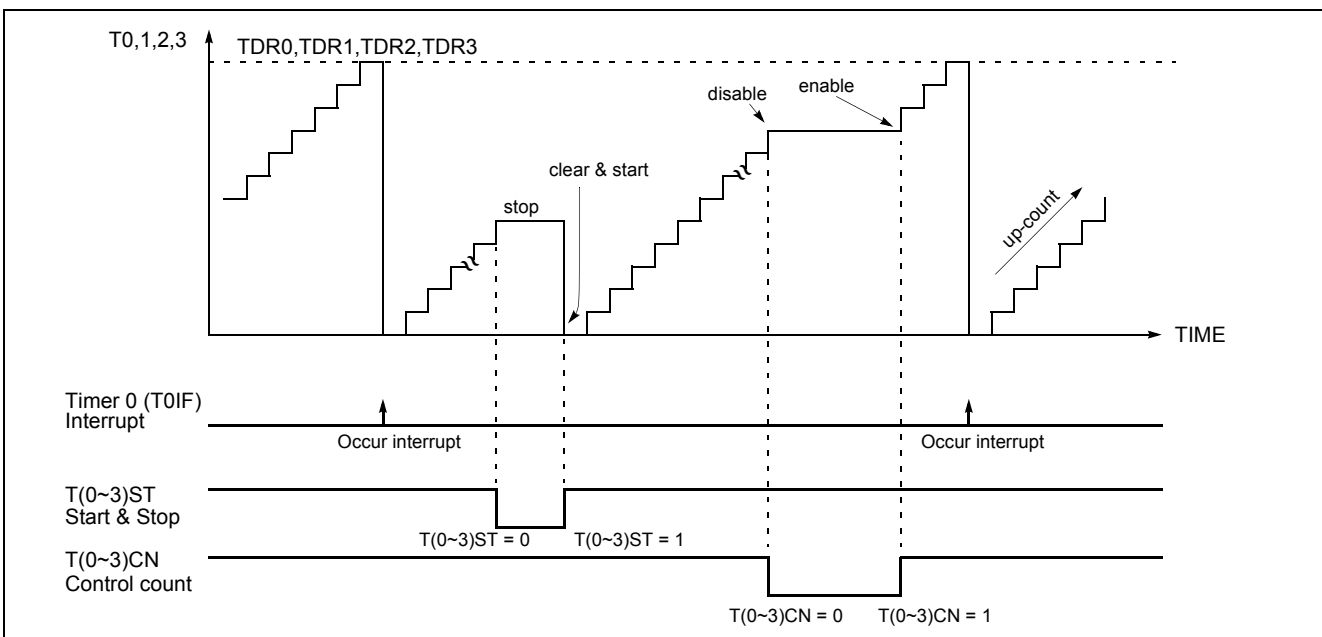
In the Timer, timer register Tx increases from 00<sub>H</sub> until it matches TxDR and then reset to 00<sub>H</sub>. If the value of Tx is equal with TxDR, Timer x interrupt is occurred (latched in TxIF bit). TxDR and T0 register are in same address, so this register is read from T0 and written to TDR0.

In counter function, the counter is increased every 0-to 1 (rising edge) transition of EC0 pin. In order to use counter function, the bit R01 of the R0 Direction Register (R0IO) should be set to "0" and the bit EC0E of Port Selection Register PSR0 should set to "1". The Timer 0 can be used as a counter by pin EC0 input, but other timers can not used as a event counter.

**Note:** The contents of TDR0, TDR1, TDR2 and TDR3 must be initialized (by software) with the value between 1<sub>H</sub> and 0FF<sub>H</sub>, not 0<sub>H</sub>.



**Figure 12-6 Counting Example of Timer Data Registers**



**Figure 12-7 Timer Count Operation**

### 12.2 16 Bit Timer/Counter Mode

The Timer register is running with 16 bits. A 16-bit timer/counter register T0, T1 are increased from 0000<sub>H</sub> until it matches TDR0, TDR1 and then resets to 0000<sub>H</sub>. The match output generates Timer 0 interrupt not Timer 1 interrupt.

The clock source of the Timer 0 is selected either internal or external clock by bit T0CK2, T0CK1 and T0CK0.

In 16-bit mode, the bits T1CK1, T1CK0 and 16BIT of TM1

should be set to “1” respectively.

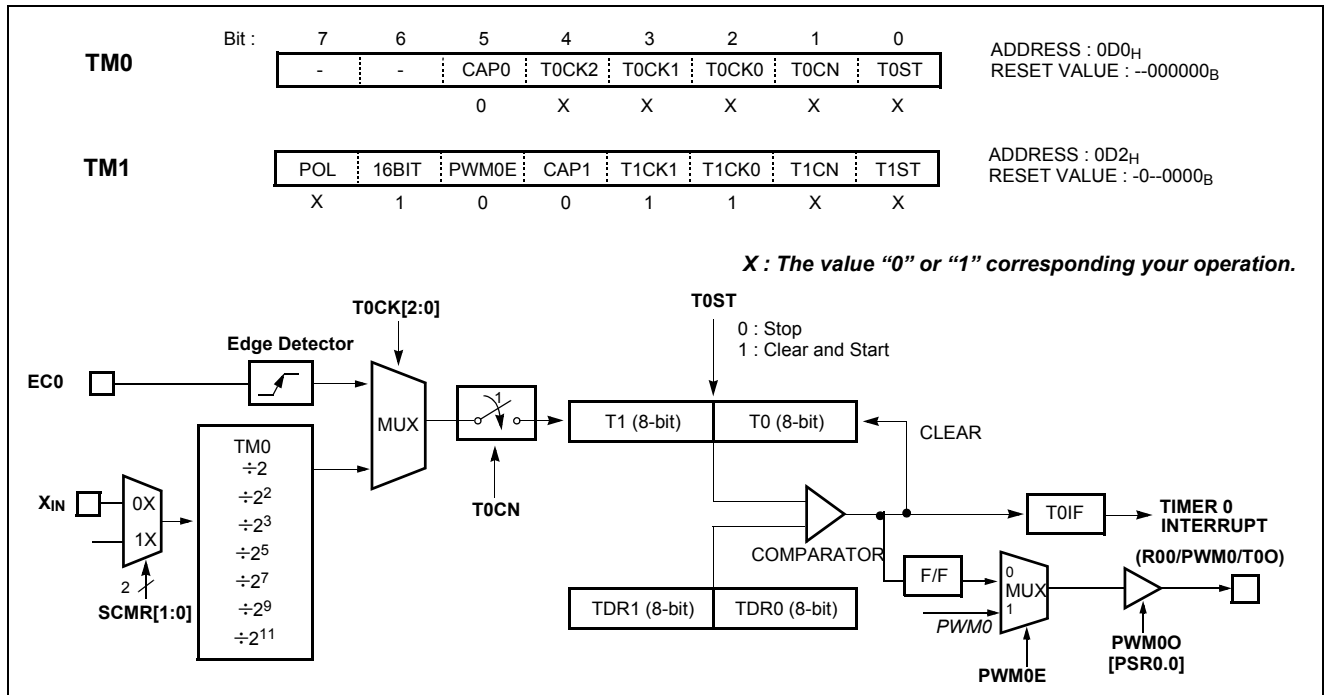


Figure 12-8 16-bit Timer / Counter Mode 0

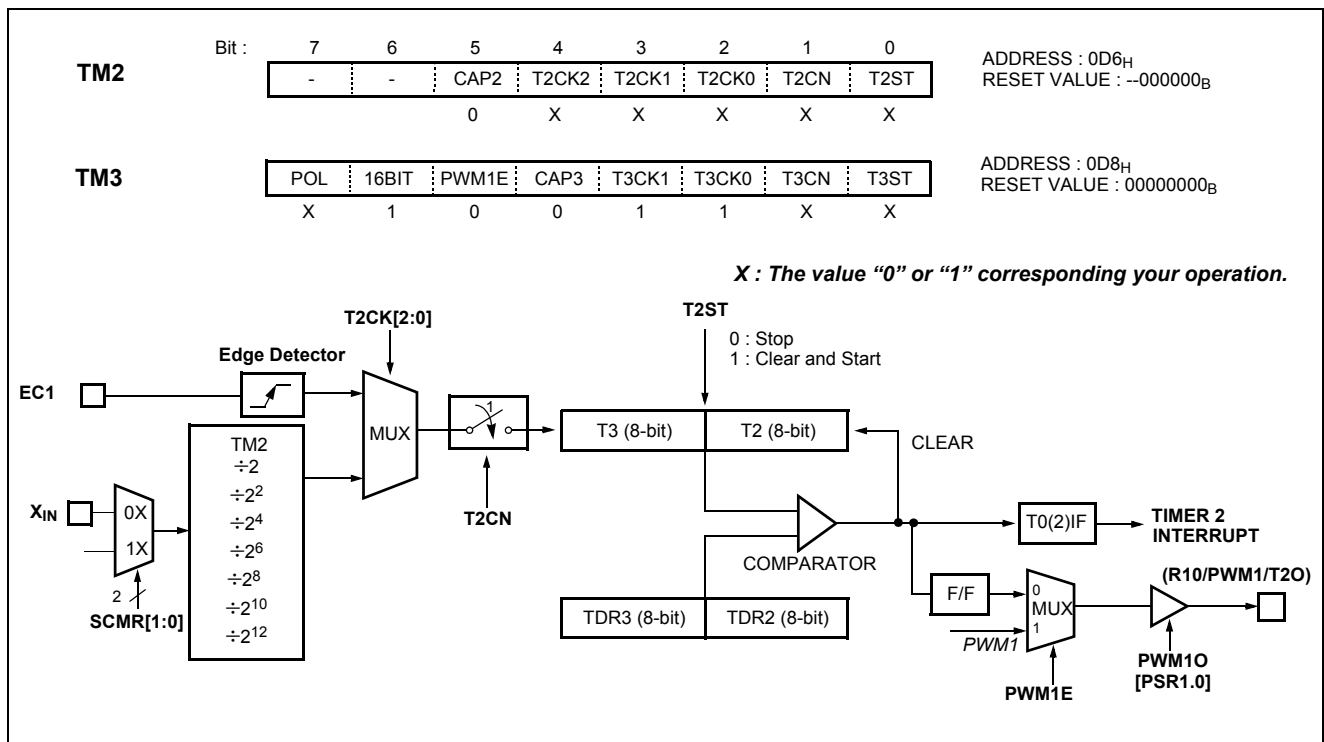


Figure 12-9 16-bit Timer / Counter Mode 2

### 12.3 8-Bit Capture Mode

The Timer 0 capture mode is set by bit CAP0 of timer mode register TM0 (bit CAPx of timer mode register TMx for Timer 1,2,3) as shown in Figure 12-10.

As mentioned above, not only Timer 0 but Timer 1,2,3 can also be used as a capture mode.

The Timer/Counter register is increased in response internal or external input. This counting function is same with normal timer mode, and Timer interrupt is generated when timer register T0 (T1,2,3) increases and matches TDR0 (TDR1,TDR2,TDR3).

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

For example, in Figure 12-13, the pulse width of captured signal is wider than the timer data value (FF<sub>H</sub>) over 2 times. When external interrupt is occurred, the captured value (13<sub>H</sub>) is more little than wanted value. It can be obtained correct value by counting the number of timer over-

flow occurrence.

Timer/Counter still does the above, but with the added feature that a edge transition at external input INTx pin causes the current value in the Timer x register (T0,T1,T2,T3), to be captured into registers CDRx (x=0,1,2,3), respectively. After captured, Timer x register is cleared and restarts by hardware.

It has three transition modes: “falling edge”, “rising edge”, “both edge” which are selected by interrupt edge selection register IEDS (Refer to External interrupt section). In addition, the transition at INTx pin generate an interrupt.

---

**Note:** *The CDR0, TDR0 and T0 are in same address. In the capture mode, reading operation is to read the CDR0 and in timer mode, reading operation is read the T0. TDR0 is only for writing operation. The CDR1, T1 are in same address, the TDR1 is located in different address. In the capture mode, reading operation is to read the CDR1*

---

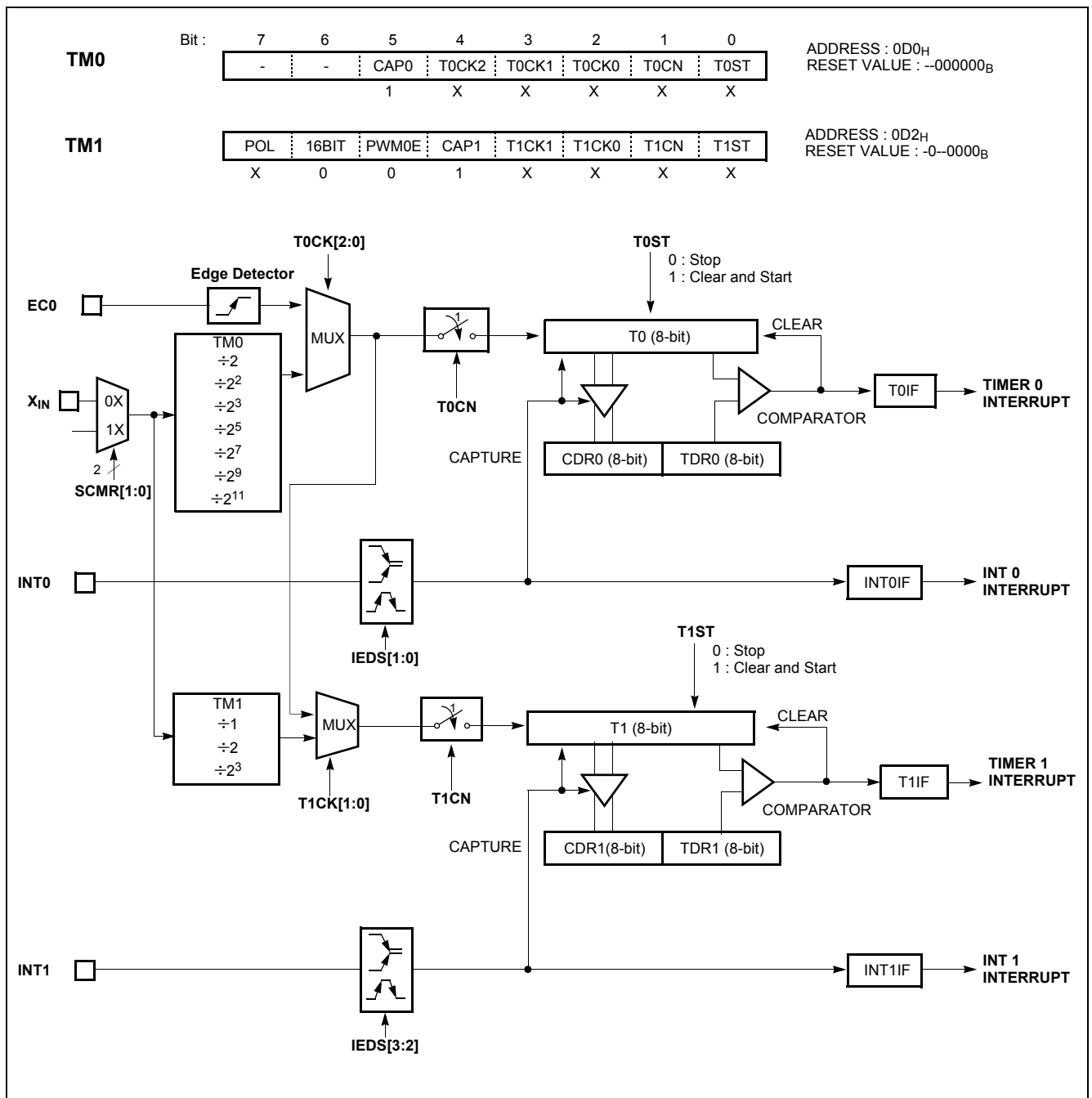


Figure 12-10 8-bit Capture Mode (Timer0, Timer1)

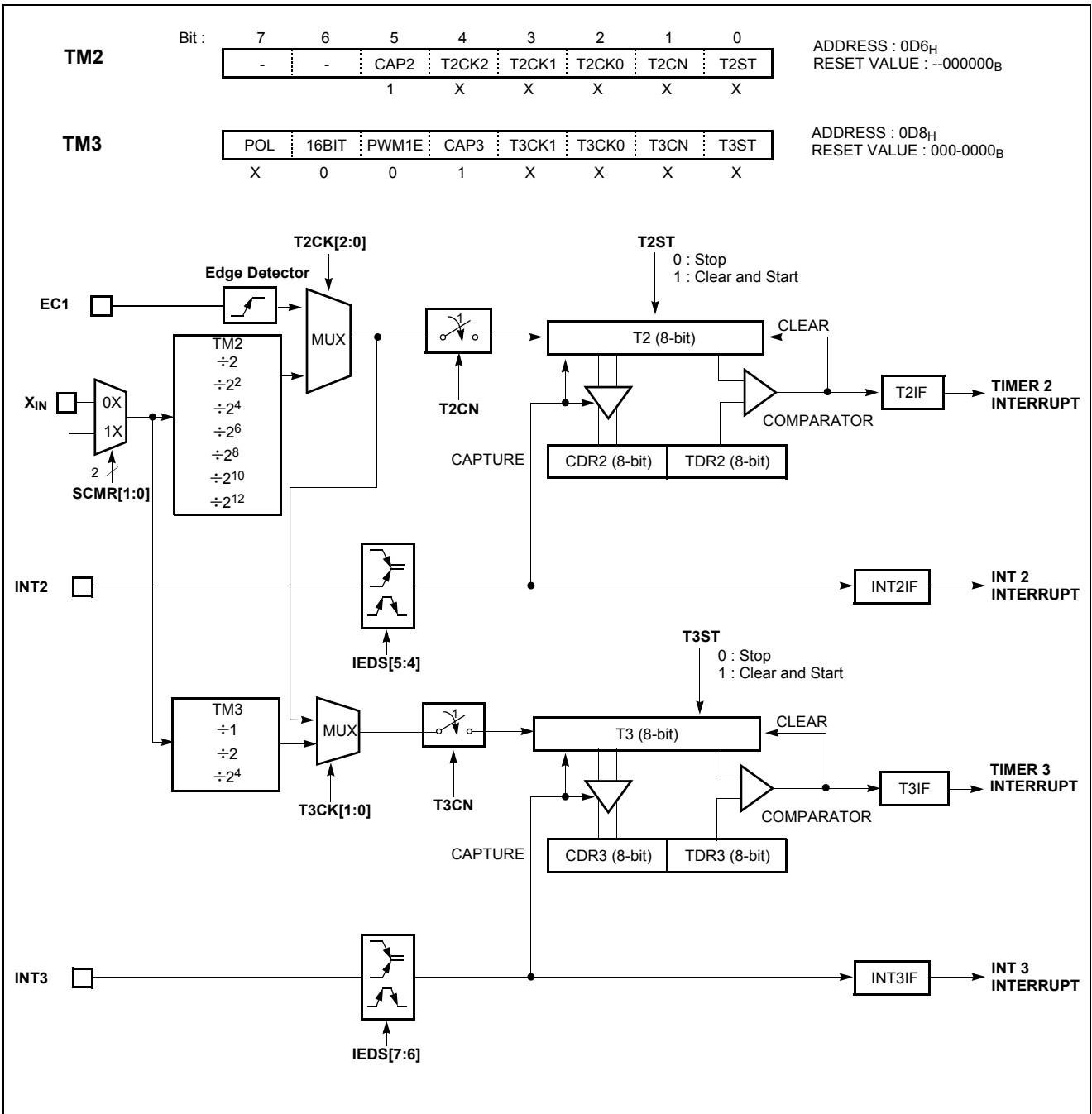


Figure 12-11 8-bit Capture Mode (Timer2, Timer3)

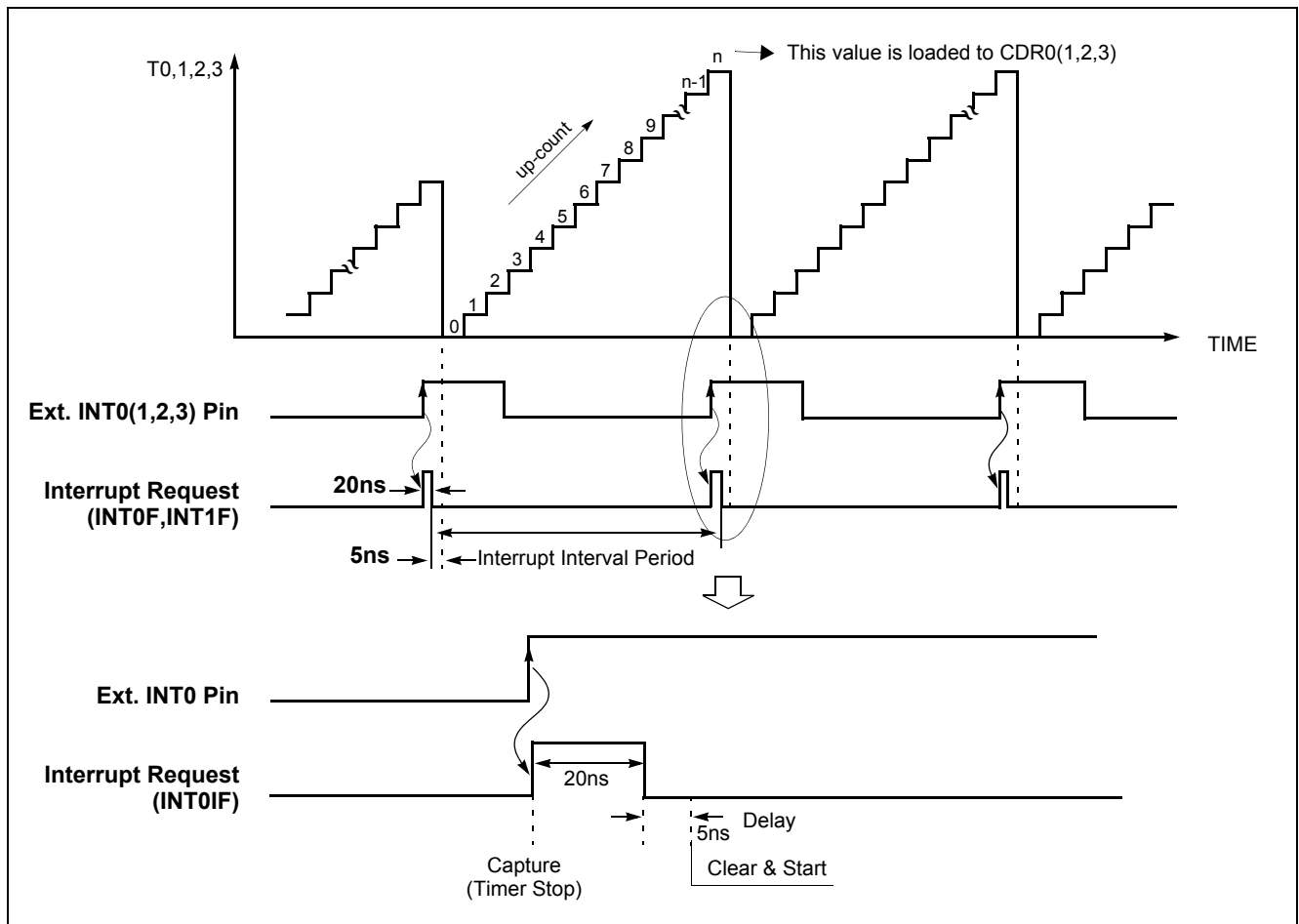


Figure 12-12 Input Capture Operation

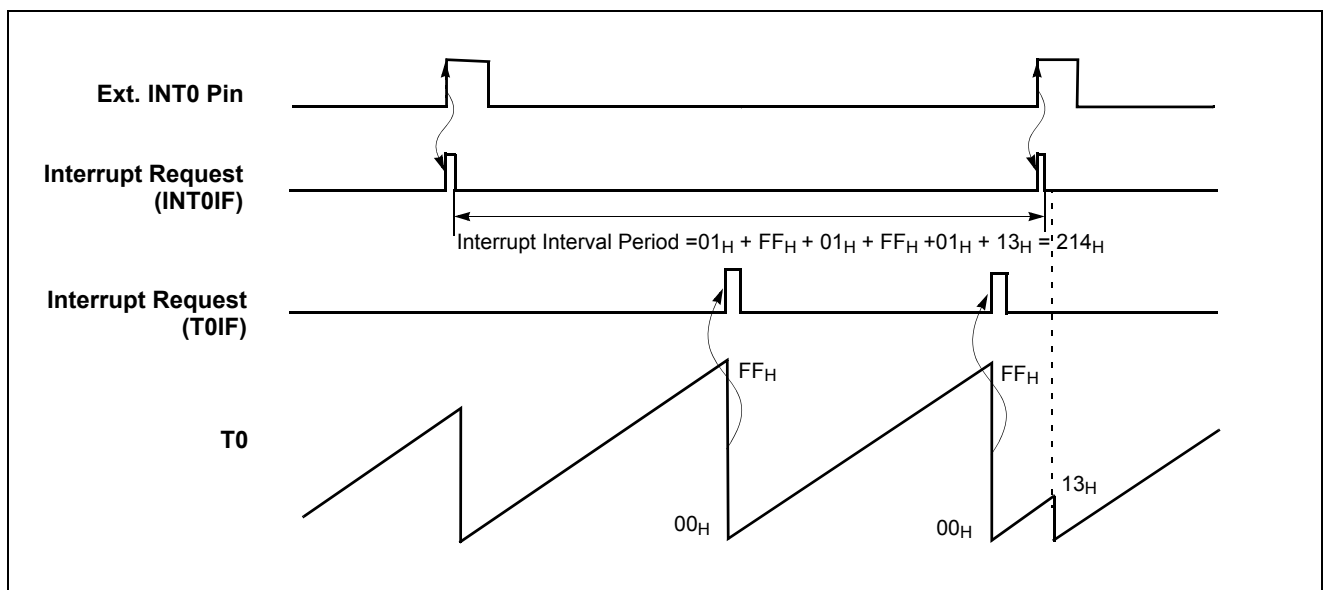


Figure 12-13 Excess Timer Overflow in Capture Mode

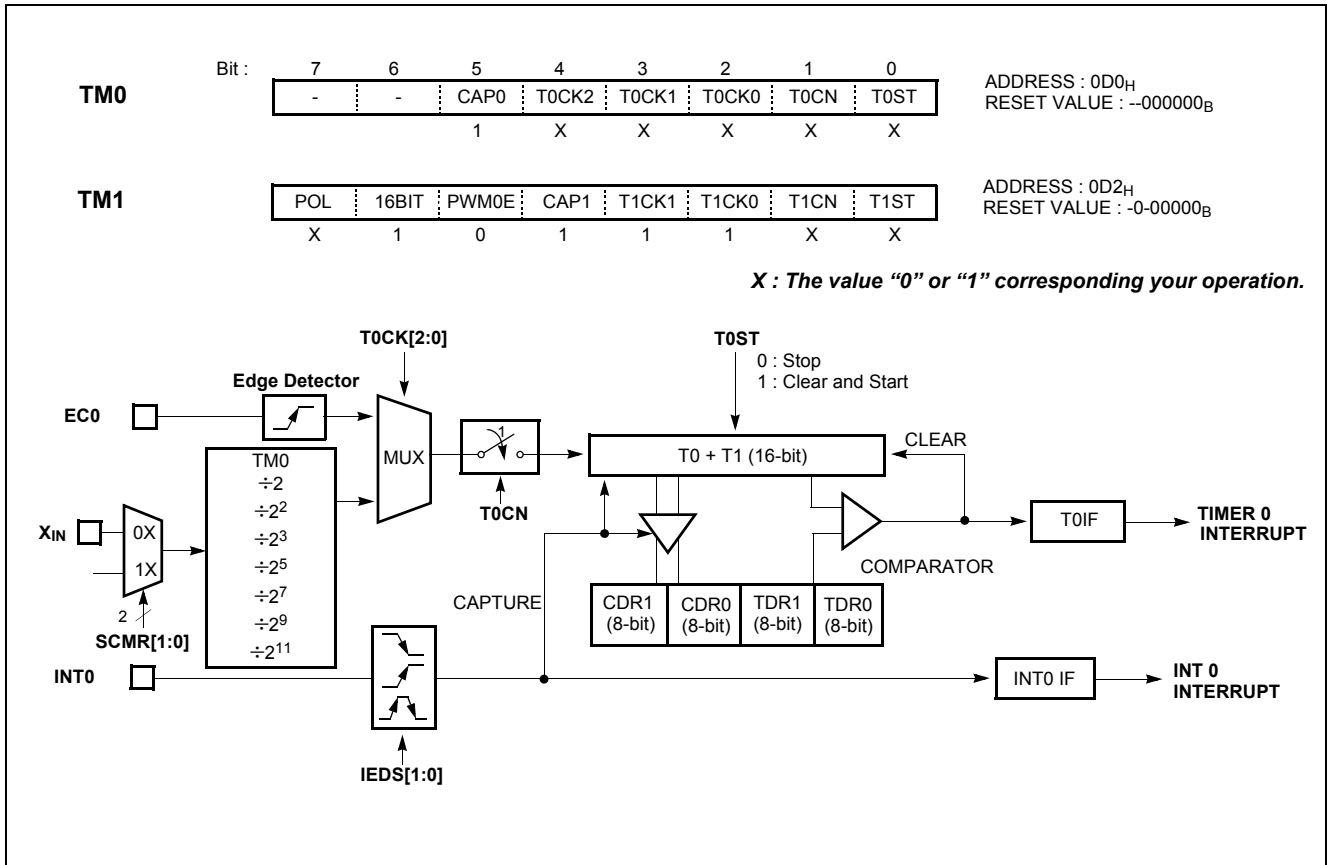


### 12.4 16-bit Capture Mode

16-bit capture mode is the same as 8-bit capture, except that the Timer register is running with 16 bits.

In 16-bit mode, the bits TxCK1, TxCK0 and 16BIT of TM1, TM3 should be set to "1" respectively.

The clock source of the Timer 0,2 is selected either internal or external clock by bit TxCK2, TxCK1 and TxCK0.



**Figure 12-14 16-bit Capture Mode (Timer0,1)**

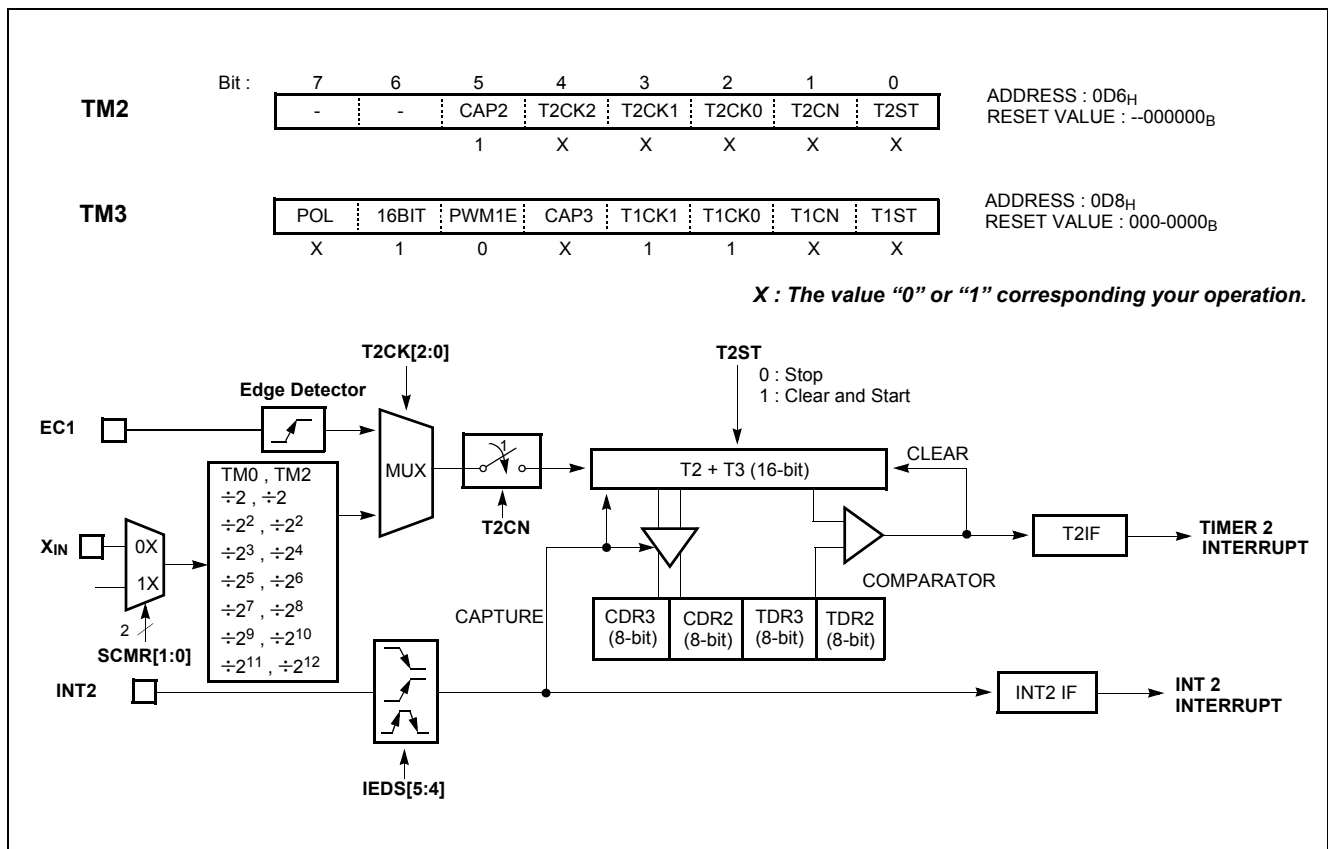


Figure 12-15 16-bit Capture Mode (Timer2,3)

### 12.5 8-Bit (16-Bit) Compare Output Mode

The MC81F8816/8616 have a function of Timer Compare Output. To pulse out, the timer match can go to port pin (R10) as shown in Figure 12-4 and Figure 12-8. Thus, pulse out is generated by the timer match. These operation is implemented to pin, R10/PWM1/T2O.

In this mode, the bit PWM1O of Port Mode Register R1FUNC should be set to "1", and the bit PWM1E of Timer3 Mode Register (TM3) should be cleared to "0".

In addition, 16-bit Compare output mode is available, also.

This pin output the signal having a 50: 50 duty square wave, and output frequency is same as below equation

$$f_{COMP} = \frac{f_{XIN}}{2 \times PrescalerValue \times (TDR + 1)}$$

### 12.6 PWM Mode

The MC81F8816/8616 has two high speed PWM (Pulse Width Modulation) function which shared with Timer1 and Timer3. In PWM mode, the R00/PWM0 and R10/PWM1 pins operate as a 10-bit resolution PWM output port. For this mode, the bit PWM0(1)O of Port Mode Register (R0(1)FUNC) and the bit PWM0(1)E of timer1(3) mode register (TM1) should be set to "1" respectively.

The period of the PWM output is determined by the T1(3)PPR (T1(3) PWM Period Register) and T1(3)PWHR[3:2] (bit3, 2 of T1(3) PWM High Register)

and the duty of the PWM output is determined by the T1(3)PDR (T1(3) PWM Duty Register) and T1(3)PWHR[1:0] (bit1, 0 of T1(3)PWM High Register).

The user can use PWM data by writing the lower 8-bit period value to the T1(3)PPR and the higher 2-bit period value to the T1(3)PWHR[3:2]. And the duty value can be used with the T1(3)PDR and the T1(3)PWHR[1:0] in the same way.

The T1(3)PDR is configured as a double buffering for

glitchless PWM output. In Figure 12-17, the duty data is transferred from the master to the slave when the period data matched to the counted value. (i.e. at the beginning of next duty cycle). The bit POL0(1) of TM1(3) decides the polarity of duty cycle.

The duty value can be changed when the PWM outputs. However the changed duty value is output after the current period is over. And it can be maintained the duty value at present output when changed only period value shown as Figure 12-19. As it were, the absolute duty time is not changed in varying frequency.

**Note:** If the user need to change mode from the Timer3 mode to the PWM mode, the Timer3 should be stopped firstly, and then set period and duty register

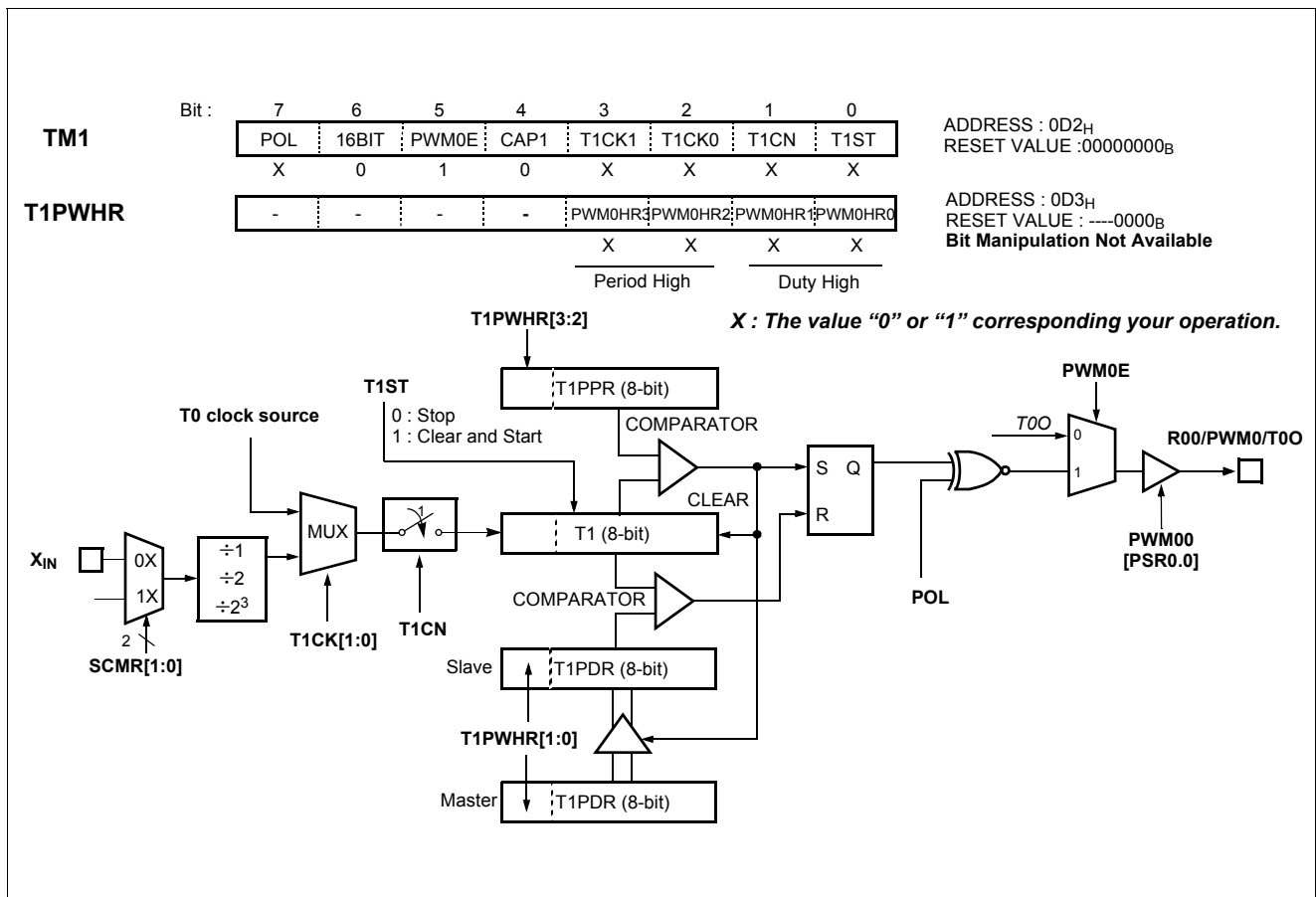
value. If user writes register values and changes mode to PWM mode while Timer3 is in operation, the PWM data would be different from expected data in the beginning.

The relation of frequency and resolution is in inverse proportion. Table 12-2 shows the relation of PWM frequency vs. resolution.

$$\text{PWM Period} = [T1(3)PWHR[3:2]T1(3)PPR+1] \times \text{Source Clock}$$

$$\text{PWM Duty} = [T1(3)PWHR[1:0]T1(3)PDR+1] \times \text{Source Clock}$$

If it needed more higher frequency of PWM, it should be reduced resolution.



**Figure 12-16 PWM0 Mode**

period value.

**Note:** If the duty value and the period value are same, the PWM output is determined by the bit POL1 (1: High, 0: Low). And if the duty value is set to "00<sub>H</sub>", the PWM output is determined by the bit POL1(1: Low, 0: High). The period value must be same or more than the duty value, and 00<sub>H</sub> cannot be used as the

Resolution	Frequency		
	T3CK[1:0] =00 (250nS)	T3CK[1:0] =01 (500nS)	T3CK[1:0] =10 (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	3.90kHz

Table 12-2 PWM Frequency vs. Resolution at 4MHz

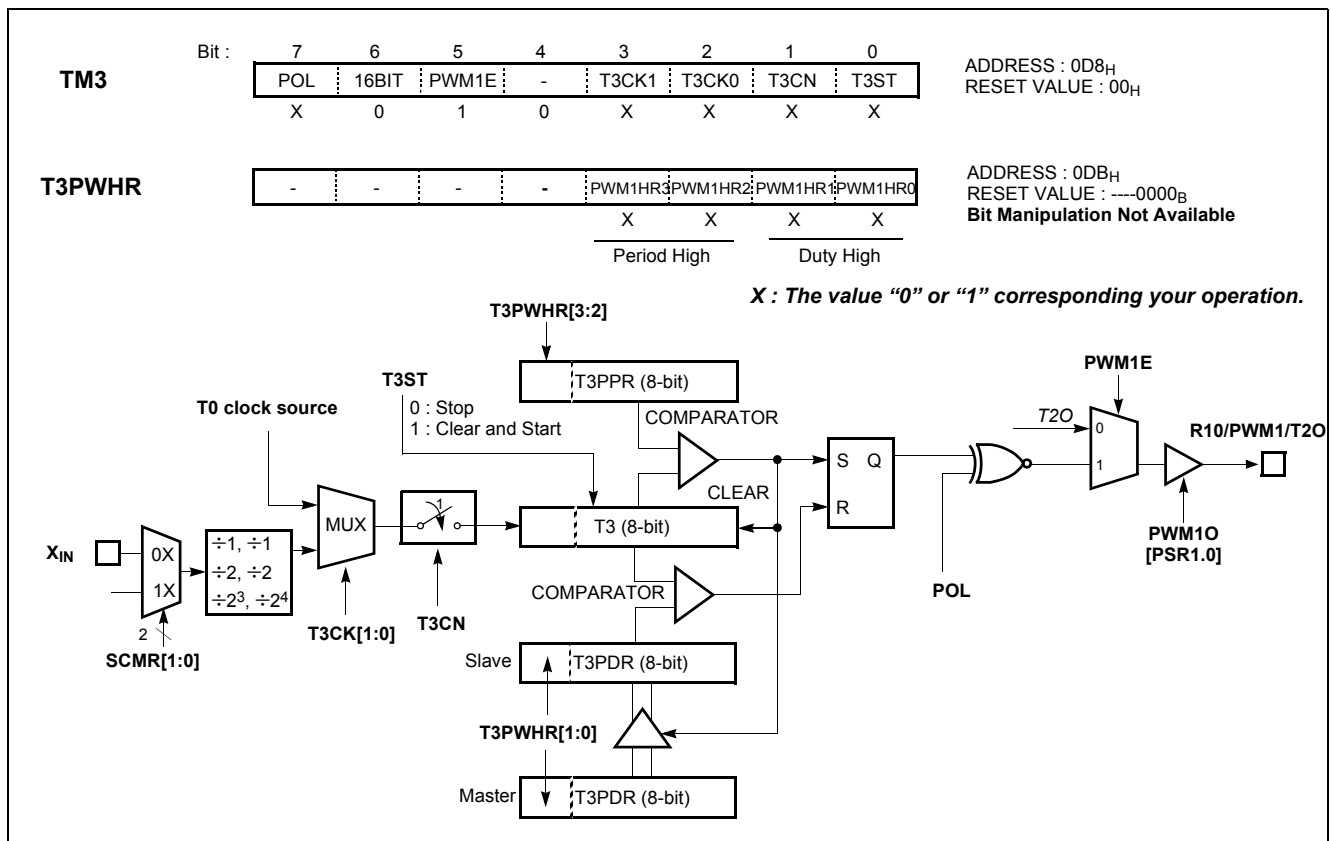


Figure 12-17 PWM1 Mode

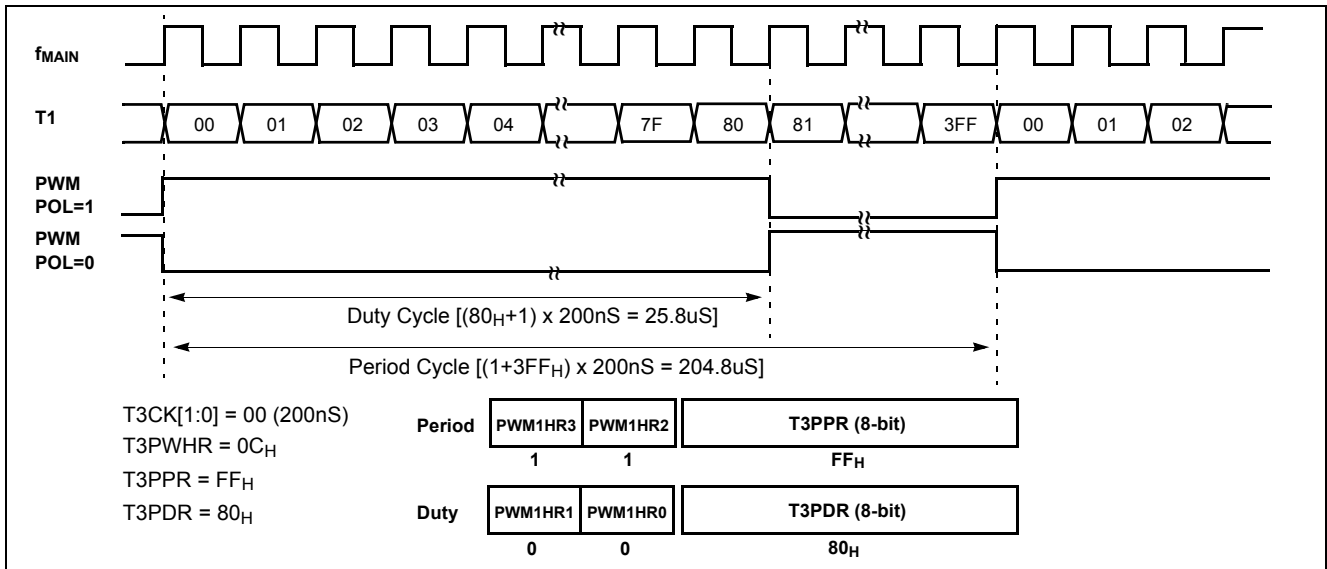


Figure 12-18 Example of PWM at 5MHz

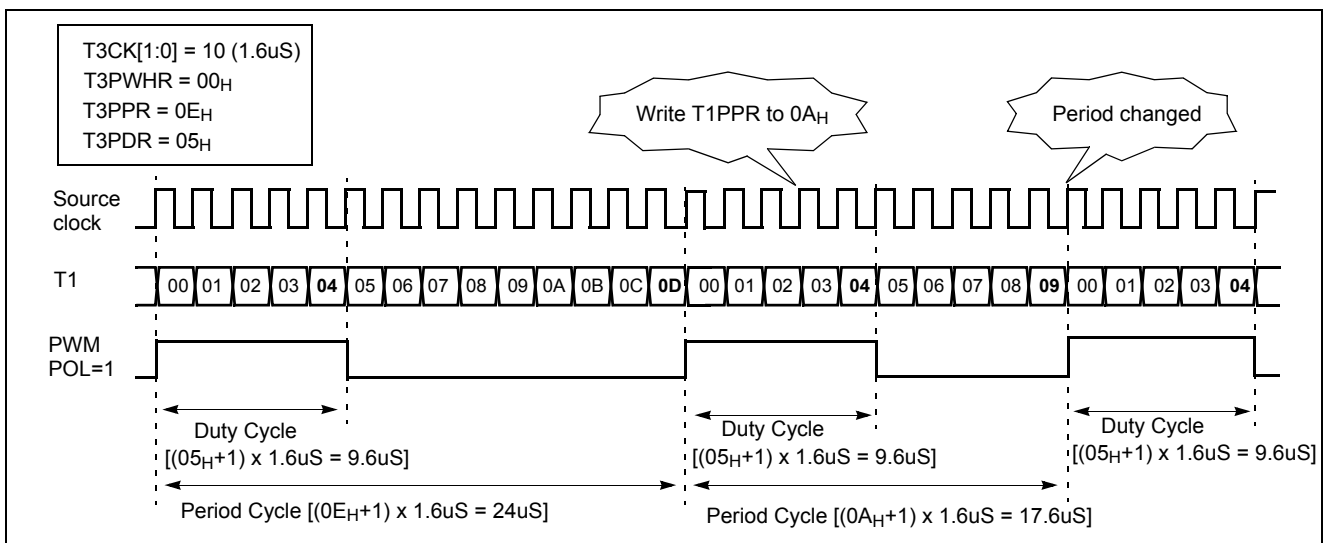


Figure 12-19 Example of Changing the Period in Absolute Duty Cycle (@5MHz)

**Example:**

Timer1 @4Mhz, 4kHz - 20% duty PWM mode

```

LDM R1IO, #0000_XXX1B ;R00 output
LDM TM3, #0010_0000B ;pwm enable
LDM T3PWHR, #0000_1100B ;20% duty
LDM T3PPR, #1110_0111B ;period 250uS
LDM T3PDR, #1100_0111B ;duty 50uS
LDM PSR1, #XXXX_XXX1B ;set pwm port
LDM TM3, #0010_0011B ;timer1 start
    
```

X means don't care

### 13. WATCH TIMER

The watch timer generates interrupt for watch operation. The watch timer consists of the clock selector, 21-bit binary counter and watch timer mode register. It is a multi-purpose timer. It is generally used for watch design.

The bit 0, 1, 2 of WTMR select the clock source of watch timer among sub-clock,  $f_{MAIN} \div 2^8$ ,  $f_{MAIN} \div 2^7$ ,  $f_{MAIN}$  or  $f_{MAIN} \div 2$  of main-clock and  $f_{MAIN}$  of main-clock. The  $f_{MAIN}$  of main-clock is used usually for watch timer test, so generally it is not used for the clock source of watch timer. The  $f_{MAIN} \div 2^7$  or  $f_{MAIN} \div 2^8$  clock is used when the single clock system is organized. If  $f_{MAIN} \div 2^8$  or  $f_{MAIN} \div 2^7$  clock is used as watch timer clock source, when the CPU enters into stop mode, the main clock is stopped and then watch

timer is also stopped. If the sub-clock is used as the watch timer source clock, the watch timer count cannot be stopped. Therefore, the sub-clock does not stop and continues to oscillate even when the CPU is in the STOP mode. The timer counter consists of 21-bit binary counter and it can count to max 60 seconds at sub-clock.

The bit 3, 4 of WTMR select the interrupt request interval of watch timer among 2Hz, 4Hz, 16Hz and 1/64Hz.

**Note:** The Clock source of watch timer is also applied to LCD driver clock source. When selecting LCD driver clock source, the WTCK[2:0] should be set to appropriate value.

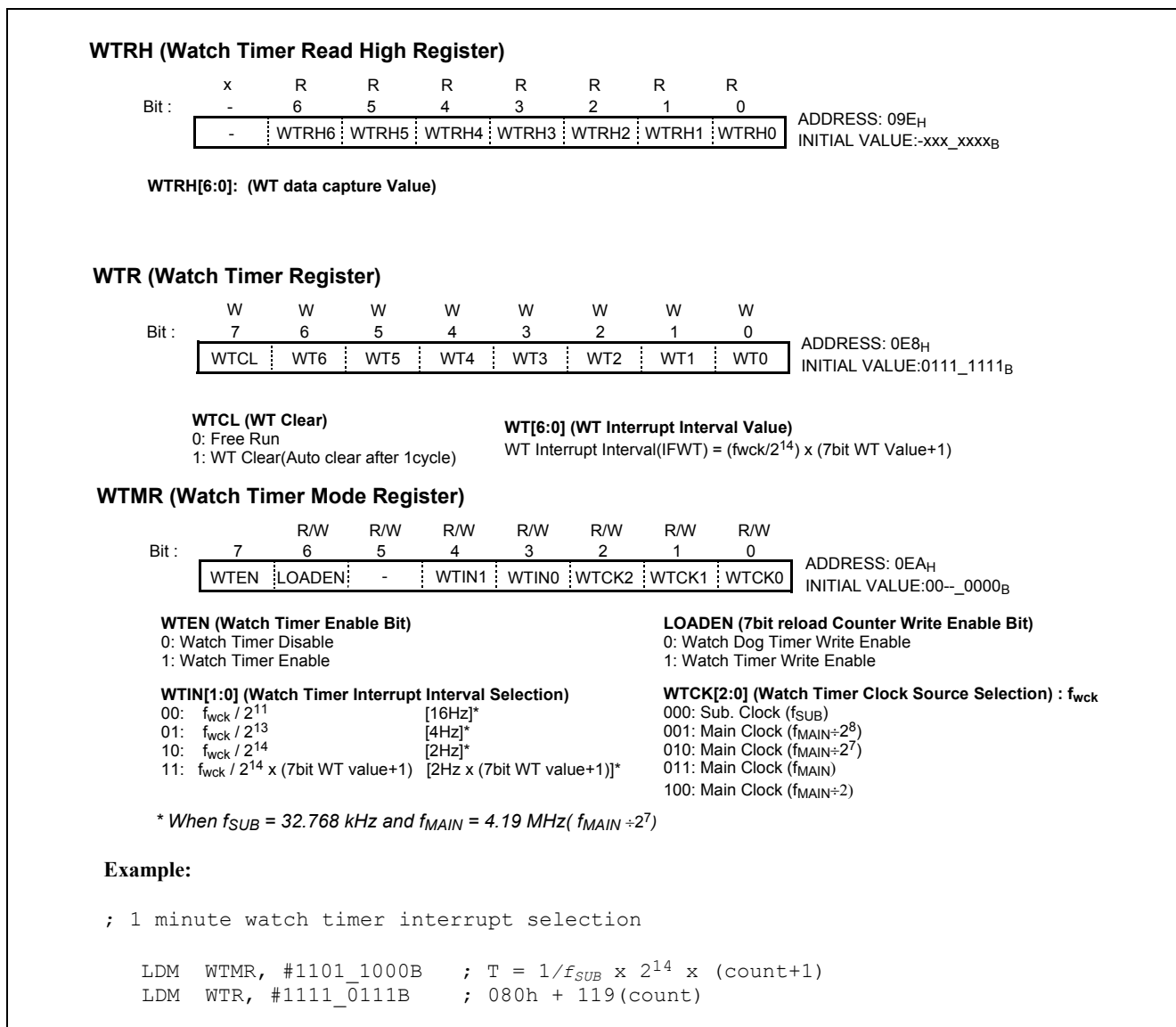


Figure 13-1 Watch Timer Mode Register

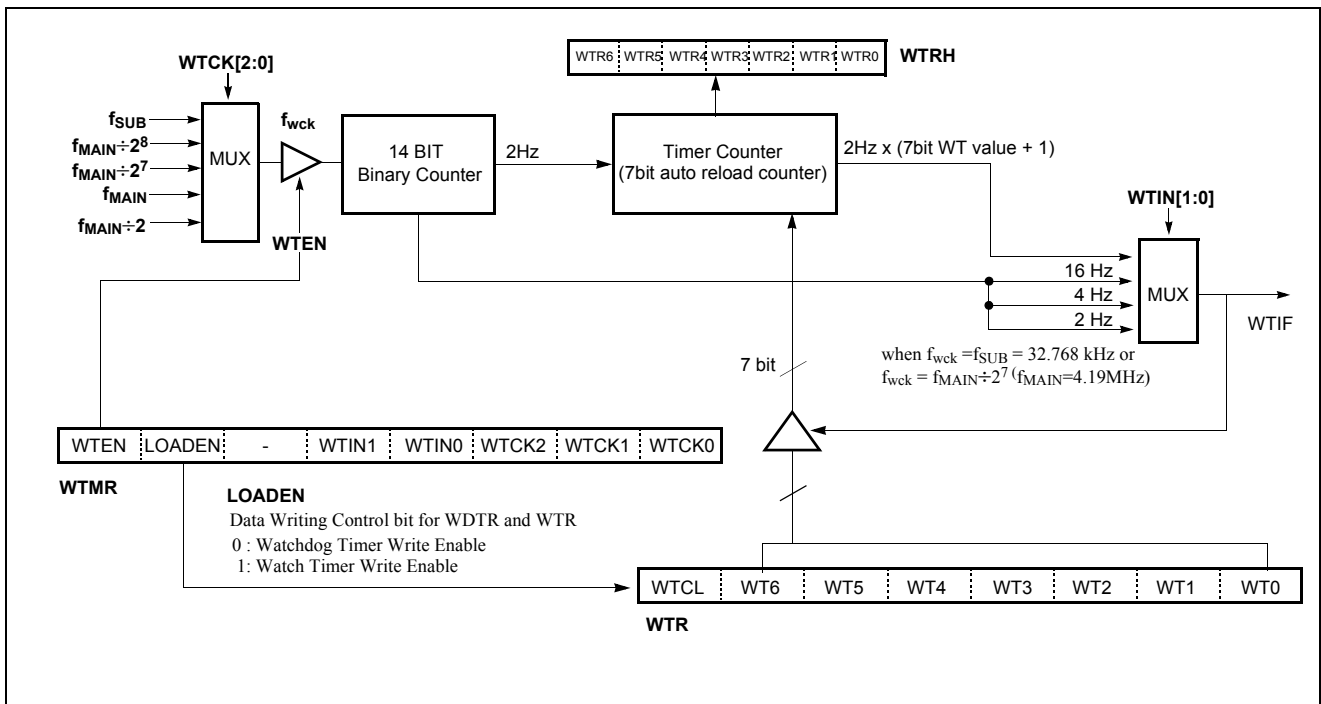


Figure 13-2 Watch Timer Block Diagram

**Usage of Watch Timer in STOP Mode**

When the system is off and the watch should be kept working, follow the steps below.

1. Set the clock source of watch timer to sub-clock.
2. Enters into STOP mode.
3. After released by watch timer interrupt, counts up timer and refreshes LCD Display. When performing count up and refresh the LCD, the CPU operates in main frequency mode.

4. Enters into STOP mode again.

5. Repeats 3 and 4.

When using STOP mode, if the watch timer interrupt interval is selected to 2Hz, the power consumption can be reduced considerably.

### 14. WATCH DOG TIMER

The watch dog timer (WDT) function is used for checking program malfunction due to external noise or other causes and return the operation to the normal contion.

The watchdog timer consists of 7-bit binary counter and the watchdog timer register(WDTR). The source clock of WDT is overflow of Basic Interval Timer. When the value of 7-bit binary counter is equal to the lower 7-bits of WDTR, the interrupt request flag is generated. This can be used as WDT interrupt or CPU reset signal in accordance with the bit WDTON. When WDTCL is set, 7-bit counter of WDT is reset. After one cycle, it is cleared by hardware.

When writing WDTR, the LOADEN bit of WTMR regis-

ter should be cleared to “0”.

**Note:** *WDTR and WTR has same address 0E8h. The LOADEN bit is used to select WDTR or WTR. When LOADEN of watch timer mode register(WTMR) is set to “1”, WDTR can not be wrote and WTR is wrote. The LOADEN bit should be cleared to “0” when writing any value to WDTR.*

**Note:** *When using watch dog timer, don't write WDT[6:0] to “000000”.*

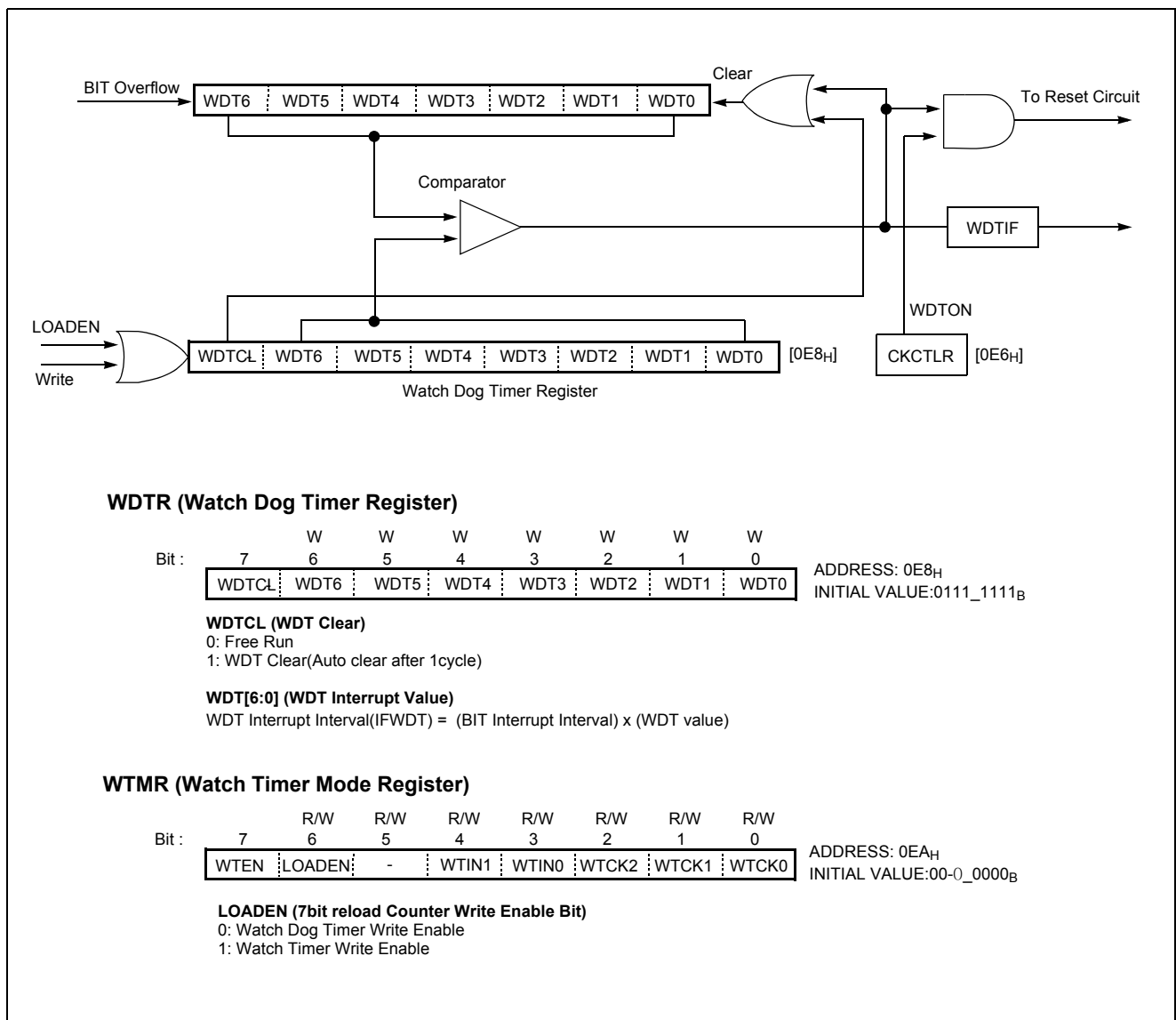


Figure 14-1 Block Diagram of Watch Dog Timer



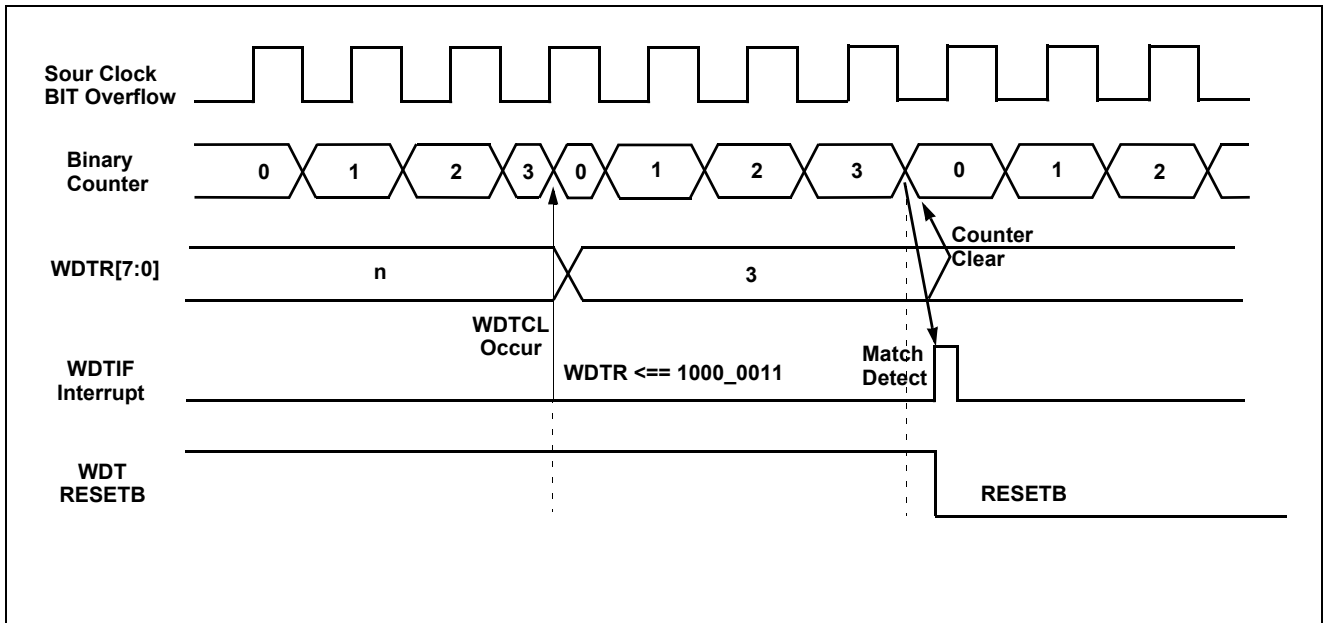


Figure 14-2 Watch Dog Timer Interrupt Time

### 15. ANALOG TO DIGITAL CONVERTER

The analog-to-digital(A/D) converter allows conversion of an analog input signal to an corresponding 10-bit digital value. The A/D module has six analog inputs, which are multiplexed into one sample and hold. The output of the sample and hold is the input of the converter, which generates the result via successive approximation. The analog supply voltage is connected to AVref of ladder resistance of A/D module.

The A/D module has three registers which are the control register ADCM and A/D result register ADCRH and ADCRL. The ADCRH[7:6] is also used as ADC clock source selection bits. The ADCM register, shown in Figure 15-2, controls the operation of the A/D converter module. The port pins can be configured as analog inputs or digital I/O. To use analog inputs, each port should be assigned analog input port by setting R2IO direction register as input mode and setting ADS[3:0] to select the corresponding channel.

The self bias check reference provides fixed voltage (typical 1.185V, tolerance to be defined), which can be the input of ADC when setting ADS[3:0] to "1111b". This feature can be used to check the voltage of VDD pin. The BOR\_ENB and AD\_REFB of BODR register should be set to "0" for using self bias check reference.

The processing of conversion is start when the start bit ADST is set to "1". After one cycle, it is cleared by hardware. The register ADCRH and ADCRL contain the result (10bit) of the A/D conversion. If the ADC is set to 8-bit mode (ADC8 bit of ADCRH is "1"), ADCRL contains the result of the A/D conversion. When the conversion is completed, the result is loaded into the ADCR, the A/D conversion status bit ADF is set to "1", and the A/D interrupt flag ADIF is set. The block diagram of the A/D module is shown in Figure 15-1. The A/D status bit ADF is automatically set when A/D conversion is completed, cleared when A/D conversion is in process. The conversion needs 13 clock period of ADC clock (f<sub>PS</sub>). It is recommended to use ADC clock of at least 1us period.

**Note:** The ADC value of self bias check reference(V<sub>bias\_ref</sub>) is can be used to check the VDD voltage. When V<sub>bias\_ref</sub> is 1.185V and VDD is 5.12V, the ADC value is "0EDh". If VDD is changed and ADC value is "13Ch", the VDD voltage is 3.84V. the ADC value is changed to "13Ch". The VDD voltage can be calculated by following formula.

$$VDD \text{ voltage} = V_{bias\_ref} \times 1024 \div \text{ADC Value}$$

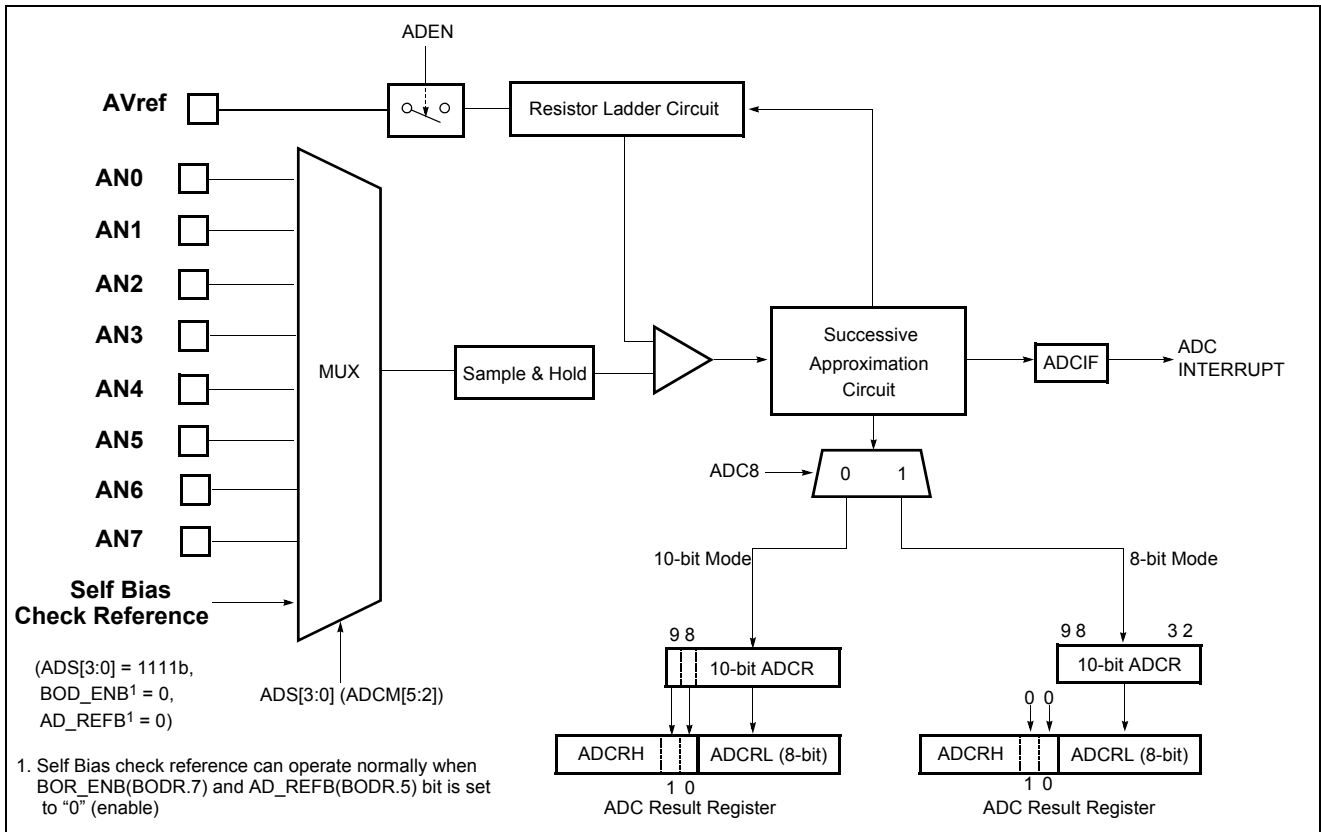


Figure 15-1 A/D Converter Block Diagram & Registers

**ADCM (A/D Converter Mode Register)**

	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R
Bit :	7	6	5	4	3	2	1	0
	ADEN	ADCK	ADS3	ADS2	ADS1	ADS0	ADST	ADF

ADDRESS : 0E2<sub>H</sub>  
RESET VALUE : 0000001<sub>B</sub>

- |   |   |
|---|---|
| <p><b>ADEN (A/D Converter Enable bit)</b><br/>1 : Enable<br/>0 : Disable</p> <p><b>ADST (A/D Start bit)</b><br/>1 : A/D Conversion is started<br/>After 1 cycle, cleared to "0"<br/>0 : Bit force to zero</p> <p><b>ADF (A/D Status bit)</b><br/>0 : A/D Conversion is in process<br/>1 : A/D Conversion is completed</p> <p><b>ADCK (A/D Converter Clock source bit)</b><br/>0 : A/D Converter Clock source = <math>f_{PS}/1</math><br/>1 : A/D Converter Clock source = <math>f_{PS}/2</math></p> | <p><b>ADS[3:0] (A/D Converter Input Selection)</b><br/>0000 : Channel 0 (R20/AN0)<br/>0001 : Channel 1 (R21/AN1)<br/>0010 : Channel 2 (R22/AN2)<br/>0011 : Channel 3 (R23/AN3)<br/>0100 : Channel 4 (R24/AN4)<br/>0101 : Channel 5 (R25/AN5)<br/>0110 : Channel 6 (R26/AN6)<br/>0111 : Channel 7 (R27/AN7)<br/>1000 : Reserved<br/>1001 : Reserved<br/>1010 : Reserved<br/>1011 : Reserved<br/>1100 : Reserved<br/>1101 : Reserved<br/>1110 : Reserved<br/>1111 : Self Bias Check Reference</p> |
|---|---|

Note : R25/AN5, R26/AN6 and R27/AN7 are not supported in MC81F8616Q.

**ADCRH (A/D Converter Result High Register)**

	W	W	W	R	R	R	R	R
Bit :	7	6	5	4	3	2	1	0
	PSSEL1	PSSEL0	ADC8	-	-	-	ADR9	ADR8

ADDRESS : 0E4<sub>H</sub>  
RESET VALUE : Undefined

- |   |  |
|---|--|
| <p><b>PSSEL[1:0] (A/D Converter PS Clock selection bit)</b><br/>00 : A/D Converter PS Clock (<math>f_{PS}</math>) = <math>f_{XIN}/4</math><br/>01 : A/D Converter PS Clock (<math>f_{PS}</math>) = <math>f_{XIN}/8</math><br/>10 : A/D Converter PS Clock (<math>f_{PS}</math>) = <math>f_{XIN}/16</math><br/>11 : A/D Converter PS Clock (<math>f_{PS}</math>) = <math>f_{XIN}/32</math></p> | <p><b>ADC8 (A/D Converter Mode bit)</b><br/>0 : 10-bit Mode<br/>1 : 8-bit Mode</p> |
|---|--|

**ADCRL (A/D Converter Result Low Register)**

	R	R	R	R	R	R	R	R
Bit :	7	6	5	4	3	2	1	0
	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0

ADDRESS : 0E3<sub>H</sub>  
RESET VALUE : Undefined

Figure 15-2 A/D Converter Mode & Result Registers

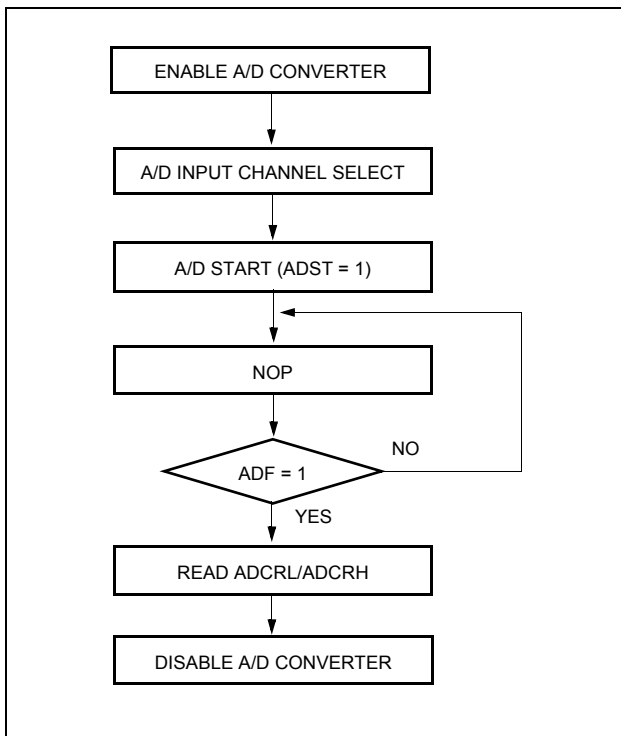


Figure 15-3 A/D Converter Operation Flow

### A/D Converter Cautions

#### (1) Input range of AN0 to AN7

The input voltages of AN0 to AN7 should be within the specification range. In particular, if a voltage above  $V_{REF}$  or below  $V_{SS}$  is input (even if within the absolute maximum rating range), the conversion value for that channel can not be determined. The conversion values of the other channels may also be affected.

#### (2) Noise counter measures

In order to maintain 8-bit resolution, any attention must be paid to noise on pins  $AV_{REF}$  and AN0 to AN7. Since the effect increases in proportion to the output impedance of the analog input source, it is recommended that a capacitor is

connected externally as shown below in order to reduce noise.

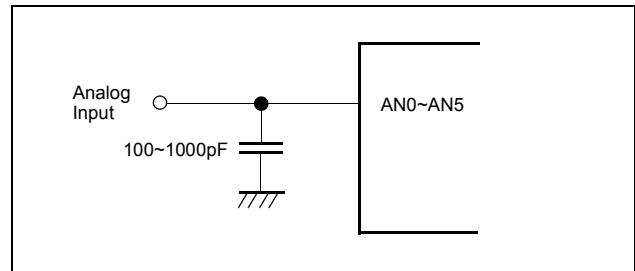


Figure 15-4 Analog Input Pin Connecting Capacitor

#### (3) Pins AN0/R20 to AN7/R27

The analog input pins AN0 to AN7 also function as input/output port (PORT R2) pins. When A/D conversion is performed with any of pins AN0 to AN7 selected, be sure not to execute a PORT input instruction while conversion is in progress, as this may reduce the conversion resolution.

Also, if digital pulses are applied to a pin adjacent to the pin in the process of A/D conversion, the expected A/D conversion value may not be obtainable due to coupling noise. Therefore, avoid applying pulses to pins adjacent to the pin undergoing A/D conversion.

#### (4) $AV_{REF}$ pin input impedance

A series resistor string of approximately  $10K\Omega$  is connected between the  $AV_{REF}$  pin and the  $V_{SS}$  pin.

Therefore, if the output impedance of the reference voltage source is high, this will result in parallel connection to the series resistor string between the  $AV_{REF}$  pin and the  $V_{SS}$  pin, and there will be a large reference voltage error.

---

**Note:** If the  $AV_{REF}$  voltage is less than  $V_{DD}$  voltage and analog input pins (ANX), shared with various alternate function, are used bidirectional I/O port, the leakage current may flow  $V_{DD}$  pin to  $AV_{REF}$  pin in output high mode or analog input pins (ANX) to  $AV_{REF}$  pin in input high mode.

---

## 16. BUZZER OUTPUT FUNCTION

The buzzer driver consists of 6-bit binary counter, the buzzer driver register BUZR and the clock selector. It generates square-wave which is very wide range frequency (500 Hz~125 kHz at  $f_{MAIN} = 4\text{MHz}$ ) by user programmable counter.

Pin R04/BUZO is assigned for output port of Buzzer driver by setting the bit BUZO of Port Selection Register0(PSR0) to "1".

The 6-bit buzzer counter is cleared and start the counting by writing signal to the register BUZR. It is increased from 00<sub>H</sub> until it matches with BUR[5:0].

Also, it is cleared by counter overflow and count up to output the square wave pulse of duty 50%.

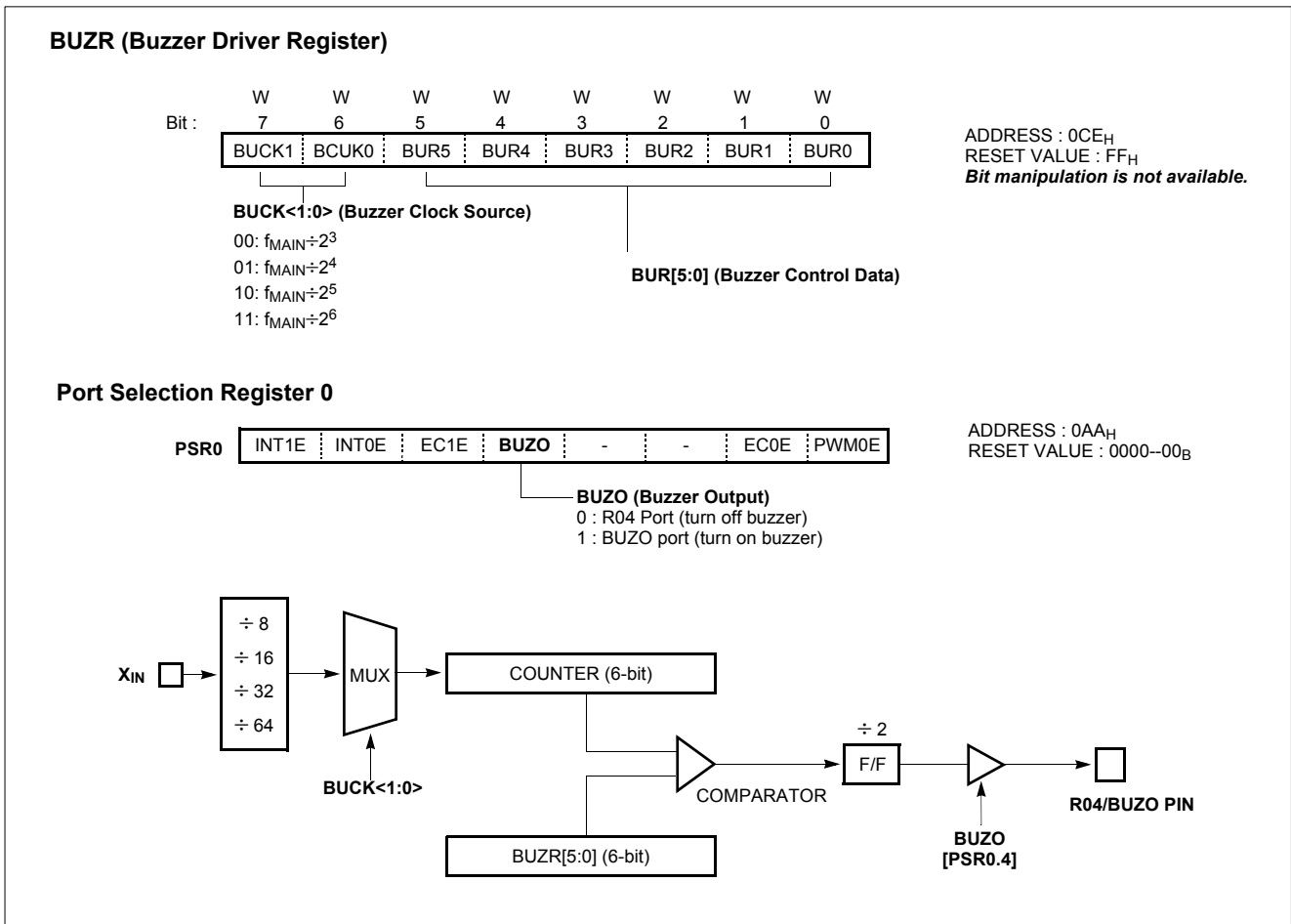
The bit 0 to 5 of BUZR determines output frequency for

buzzer driving. BUZR[5:0] is initialized to 3F<sub>H</sub> after reset. Note that BUZR is a write-only register. Frequency calculation is following as shown below.

$$f_{BUZ} = \frac{f_{XIN}}{2 \times DivideRatio \times (BUZR[5:0] + 1)}$$

The bits BUCK1, BUCK0 of BUZR select the source clock from prescaler output.

- $f_{BUZ}$ : Buzzer frequency
- $f_{XIN}$ : Oscillator frequency
- Divide Ratio: Prescaler divide ratio by BUCK[1:0]
- BUZR[5:0]: Lower 6-bit value of BUZR. Buzzer control data



**Figure 16-1 Buzzer Driver**

Example: 2.5kHz output at 4MHz.

```
LDM R0FUNC, #XXX1_XXXXB
LDM BUZR, #1001_1000B
```

X means don't care

### Buzzer Output Frequency

When main-frequency is 4MHz, buzzer frequency is shown as below.

BUZR [5:0]	Frequency Output (kHz) BUZR[7:6]				BUZR [5:0]	Frequency Output (kHz) BUZR[7:6]			
	00	01	10	11		00	01	10	11
00	250.000	125.000	62.500	31.250	20	7.576	3.788	1.894	0.947
01	125.000	62.500	31.250	15.625	21	7.353	3.676	1.838	0.919
02	83.333	41.667	20.833	10.417	22	7.143	3.571	1.786	0.893
03	62.500	31.250	15.625	7.813	23	6.944	3.472	1.736	0.868
04	50.000	25.000	12.500	6.250	24	6.757	3.378	1.689	0.845
05	41.667	20.833	10.417	5.208	25	6.579	3.289	1.645	0.822
06	35.714	17.857	8.929	4.464	26	6.410	3.205	1.603	0.801
07	31.250	15.625	7.813	3.906	27	6.250	3.125	1.563	0.781
08	27.778	13.889	6.944	3.472	28	6.098	3.049	1.524	0.762
09	25.000	12.500	6.250	3.125	29	5.952	2.976	1.488	0.744
0A	22.727	11.364	5.682	2.841	2A	5.814	2.907	1.453	0.727
0B	20.833	10.417	5.208	2.604	2B	5.682	2.841	1.420	0.710
0C	19.231	9.615	4.808	2.404	2C	5.556	2.778	1.389	0.694
0D	17.857	8.929	4.464	2.232	2D	5.435	2.717	1.359	0.679
0E	16.667	8.333	4.167	2.083	2E	5.319	2.660	1.330	0.665
0F	15.625	7.813	3.906	1.953	2F	5.208	2.604	1.302	0.651
10	14.706	7.353	3.676	1.838	30	5.102	2.551	1.276	0.638
11	13.889	6.944	3.472	1.736	31	5.000	2.500	1.250	0.625
12	13.158	6.579	3.289	1.645	32	4.902	2.451	1.225	0.613
13	12.500	6.250	3.125	1.563	33	4.808	2.404	1.202	0.601
14	11.905	5.952	2.976	1.488	34	4.717	2.358	1.179	0.590
15	11.364	5.682	2.841	1.420	35	4.630	2.315	1.157	0.579
16	10.870	5.435	2.717	1.359	36	4.545	2.273	1.136	0.568
17	10.417	5.208	2.604	1.302	37	4.464	2.232	1.116	0.558
18	10.000	5.000	2.500	1.250	38	4.386	2.193	1.096	0.548
19	9.615	4.808	2.404	1.202	39	4.310	2.155	1.078	0.539
1A	9.259	4.630	2.315	1.157	3A	4.237	2.119	1.059	0.530
1B	8.929	4.464	2.232	1.116	3B	4.167	2.083	1.042	0.521
1C	8.621	4.310	2.155	1.078	3C	4.098	2.049	1.025	0.512
1D	8.333	4.167	2.083	1.042	3D	4.032	2.016	1.008	0.504
1E	8.065	4.032	2.016	1.008	3E	3.968	1.984	0.992	0.496
1F	7.813	3.906	1.953	0.977	3F	3.906	1.953	0.977	0.488

Table 16-1 Buzzer Output Frequency

## 17. INTERRUPTS

The MC81F8816/8616 interrupt circuits consist of Interrupt enable register (IENH, IENM, IENL), Interrupt request flag register (IRQH, IRQM, IRQL), Interrupt flag register (INTFH, INTFL), Interrupt Edge Selection Register (IEDS), priority circuit and Master enable flag (“I” flag of PSW). The interrupts are controlled by the interrupt master enable flag I-flag (bit 2 of PSW), the interrupt enable register and the interrupt request flag register except Power-on reset and software BRK interrupt. The configuration of interrupt circuit is shown in Figure 17-1 and interrupt priority is shown in Table 17-1 .

**Table 17-1 Vector Table**

Reset/Interrupt	Symbol	Priority	Vector Addr.	800 C-complier
Hardware Reset	RESET	0	FFFEH	INT15
External Int. 0	INTR0	1	FFF8H	INT13
External Int. 1	INTR1	2	FFF8H	INT12
External Int. 2	INTR2	3	FFF6H	INT11
External Int. 3	INTR3	4	FFF4H	INT10
UART_RX0	RX0	5	FFF2H	INT9
UART_TX0	TX0	6	FFF2H	INT9
SPI	SPI	7	FFEEH	INT7
Timer 0 Int.	T0	8	FFEC	INT6
Timer 1 Int.	T1	9	FFEAH	INT5
Timer 2 Int.	T2	10	FFE8H	INT4
Timer 3 Int.	T3	11	FFE6H	INT3
I2C	I2C	12	FFE4H	INT2
A/D Int.	ADC	13	FFE4H	INT2
BIT Int.	BIT	14	FFE2H	INT1
Watch Dog timer int.	WDT	15	FFE0H	INT0
Watch timer int.	WT	16	FFE0H	INT0

Each bit of interrupt request flag registers (IRQH, IRQM, IRQL) in Figure 17-1 is set when corresponding interrupt condition is met. The interrupt request flags that actually generate external interrupts are bit INT0F, INT1F and INT2F in Register IRQH and INT3F in Register IRQL. The External Interrupts INT0, INT1, INT2 and INT3 can each be transition-activated (1-to-0, 0-to-1 and both transition). The RX0 and TX0 of UART0 Interrupts are generated by RX0IF and TX0IF which are set by finishing the reception and transmission of data.

The Timer 0,1,2 and Timer 3 Interrupts are generated by T0IF, T1IF, T2IF and T3IF, which are set by a match in their respective timer/counter register. The AD converter

Interrupt is generated by ADCIF which is set by finishing the analog to digital conversion.

The Basic Interval Timer Interrupt is generated by BITIF which is set by overflow of the Basic Interval Timer Register (BITR). The Watch dog Interrupt is generated by WDTIF which set by a match in Watch dog timer register (when the bit WDTON is set to “0”). The Watch Timer Interrupt is generated by WTIF which is set periodically according to the established time interval.

When an interrupt is generated, the bit of interrupt request flag register (IRQH, IRQM, IRQL) that generated it is cleared by the hardware when the service routine is vectored to only if the interrupt was transition-activated.

Each bit of Interrupt flag register (INTFH, INTFL) is set when corresponding interrupt flag bit as well as interrupt enable bit are set. The bits of interrupt flag register are never cleared by the hardware although the service routine is vectored to. Therefore, the interrupt flag register can be used to distinguish a right interrupt source from two available ones in a vector address. For example, RX0 and TX0 which have the same vector address (FFF2H) may be distinguished by INTFH register.

Interrupt enable registers are shown in Figure 17-2. These registers are composed of interrupt enable bits of each interrupt source, these bits determine whether an interrupt will be accepted or not. When enable bit is “0”, a corresponding interrupt source is prohibited. Note that PSW contains also a master enable bit, I-flag, which disables all interrupts at once. When an interrupt is occurred, the I-flag is cleared and disable any further interrupt, the return address and PSW are pushed into the stack and the PC is vectored to.

In an interrupt service routine, any other interrupt may be serviced. The source(s) of these interrupts can be determined by polling the interrupt request flag bits. Then, the interrupt request flag bit(s) must be cleared by software before re-enabling interrupts to avoid recursive interrupts. The Interrupt Request flags are able to be read and written.

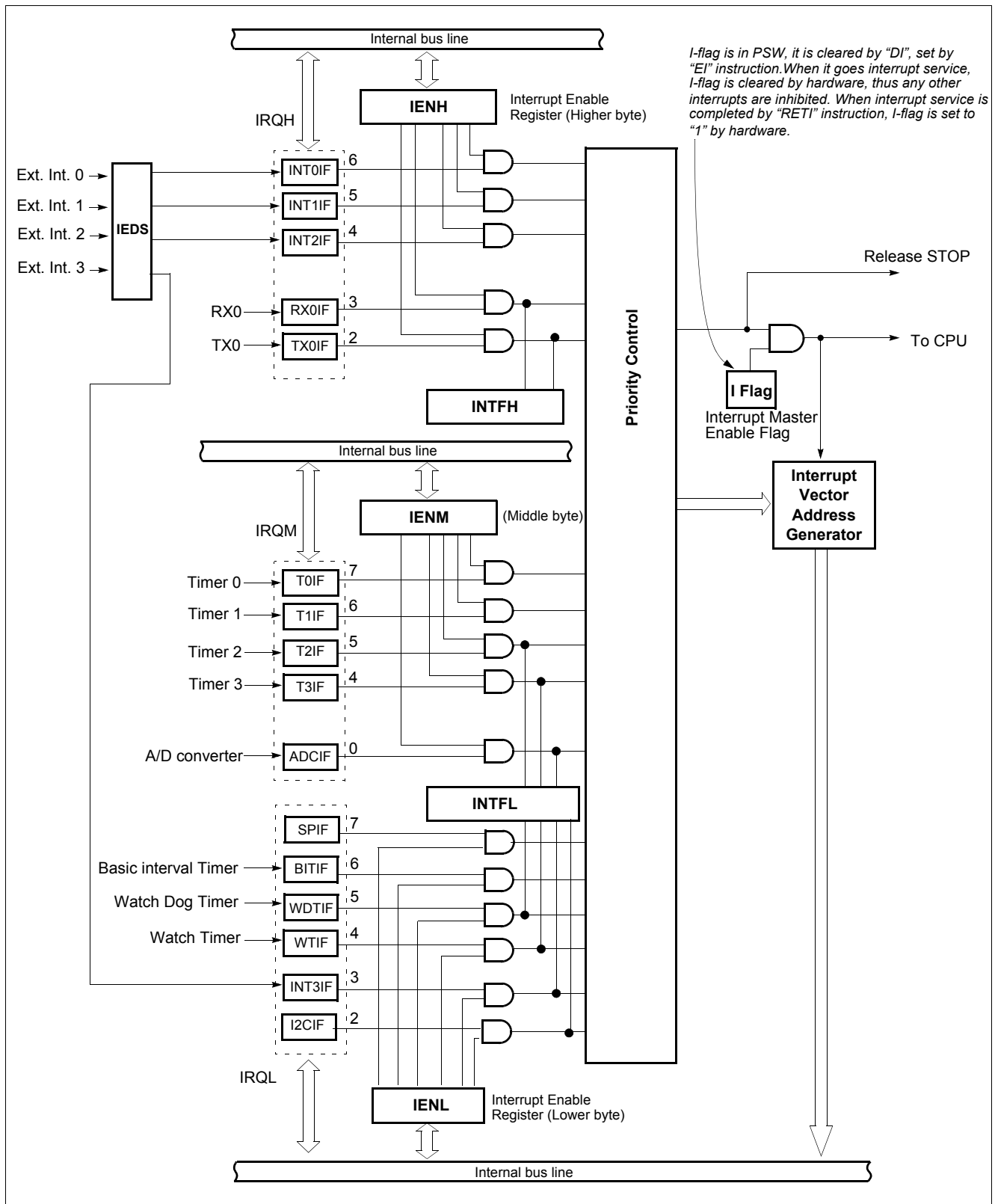


Figure 17-1 Block Diagram of Interrupt



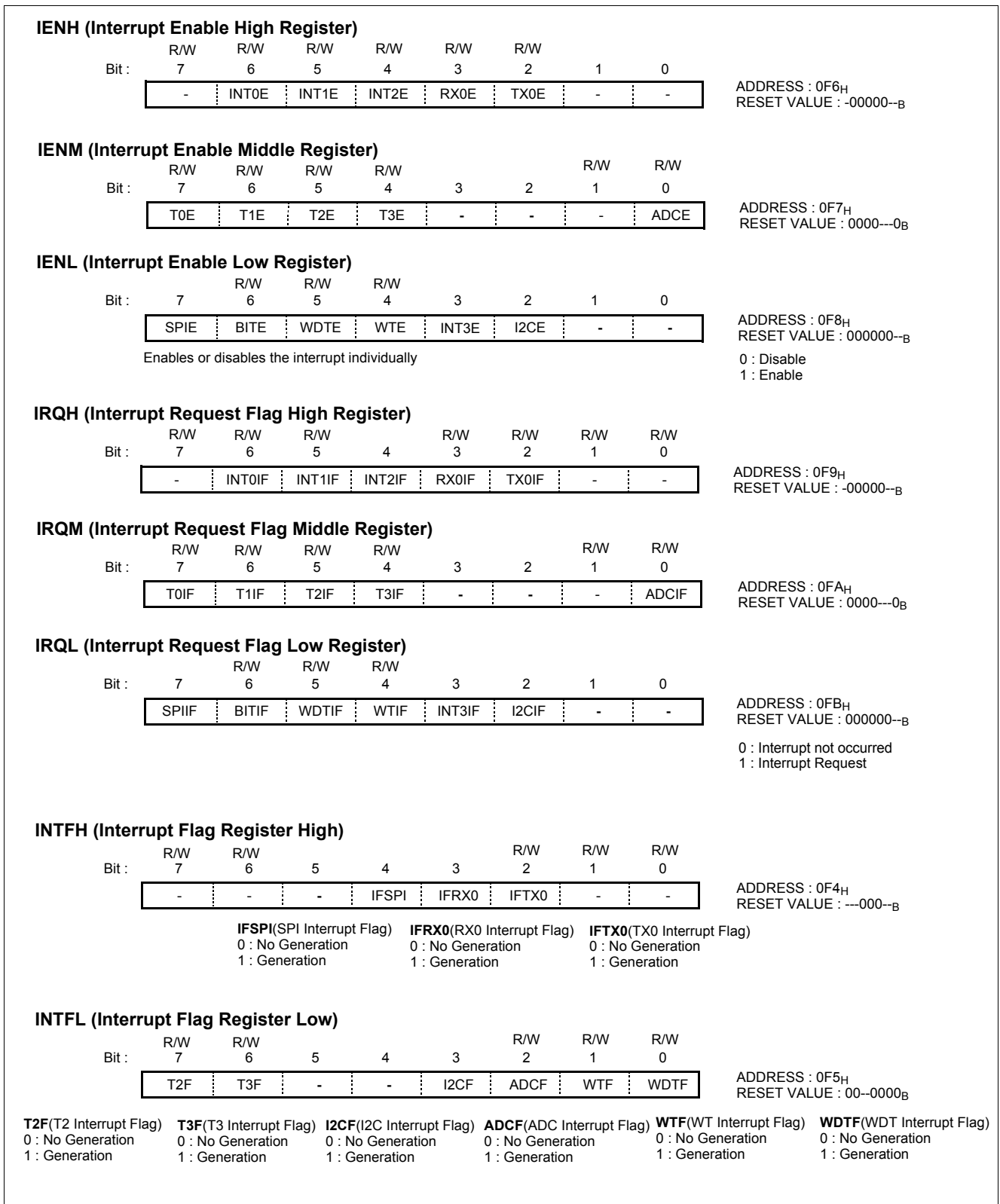


Figure 17-2 Interrupt Enable Registers and Interrupt Request Registers

### 17.1 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 sequence requires  $8 f_{OSC}$  ( $2 \mu s$  at  $f_{MAIN}=4MHz$ ) after the completion of the current instruction execution. The interrupt service task is terminated upon execution of an interrupt return instruction [RETI].

#### Interrupt acceptance

1. The interrupt master enable flag (I-flag) is cleared to “0” to temporarily disable the acceptance of any following maskable interrupts. When a non-maskable interrupt is accepted, the acceptance of any following interrupts is temporarily disabled.

2. Interrupt request flag for the interrupt source accepted is cleared to “0”.
3. The contents of the program counter (return address) and the program status word are saved (pushed) onto the stack area. The stack pointer decreases 3 times.
4. The entry address of the interrupt service program is read from the vector table address and the entry address is loaded to the program counter.
5. The instruction stored at the entry address of the interrupt service program is executed.

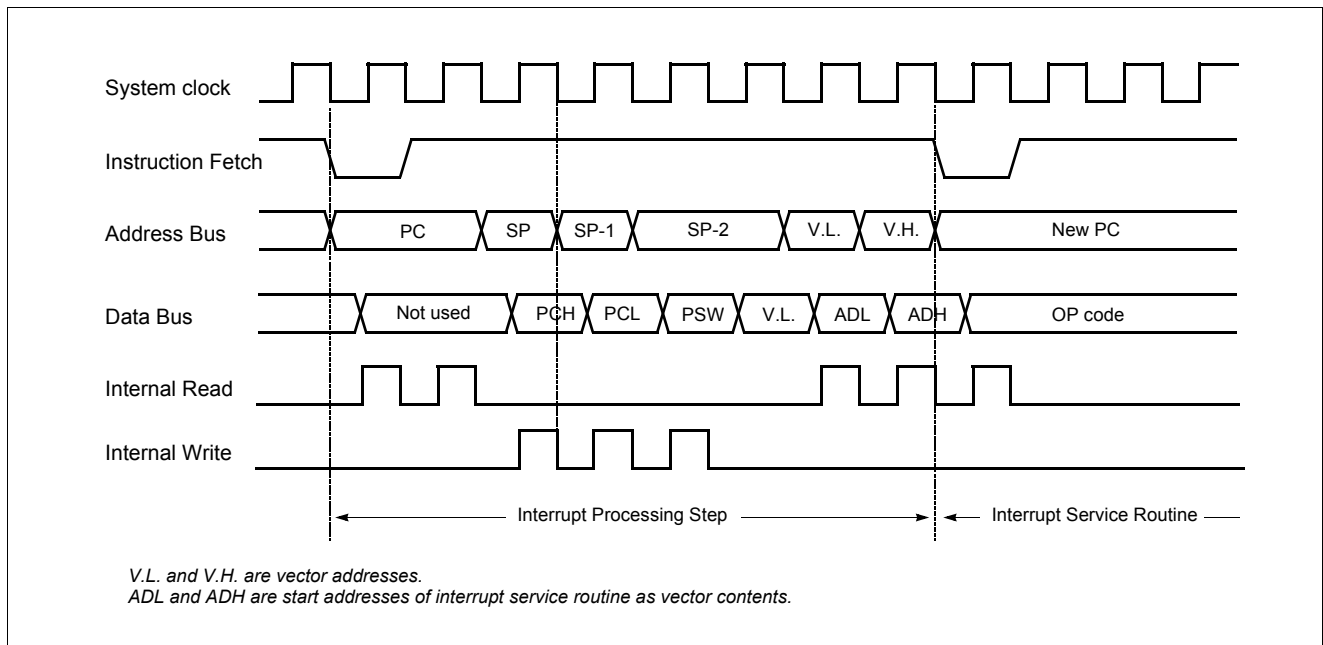
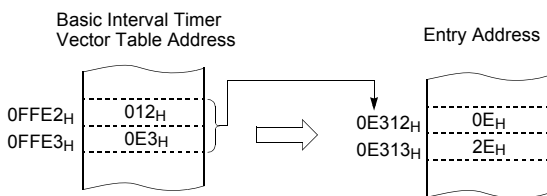


Figure 17-3 Timing chart of Interrupt Acceptance and Interrupt Return Instruction



Correspondence between vector table address for BIT interrupt and the entry address of the interrupt service program.

An interrupt request is not accepted until the I-flag is set to “1” even if a requested interrupt has higher priority than that of the current interrupt being serviced.

When nested interrupt service is required, the I-flag should be set to “1” by “EI” instruction in the interrupt service program. In this case, acceptable interrupt sources are selectively enabled by the individual interrupt enable flags.

#### Saving/Restoring General-purpose Register

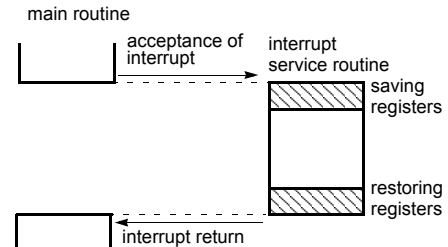
During interrupt acceptance processing, the program counter and the program status word are automatically saved on the stack, but accumulator and other registers are not saved itself. If necessary, these registers should be saved by the software. Also, when multiple interrupt services are nested, it is necessary to avoid using the same data memory area for saving registers.

The following method is used to save/restore the general-purpose registers.

Example: Register saving

```
INTxx:  PUSH   A      ;SAVE ACC.
        PUSH   X      ;SAVE X REG.
        PUSH   Y      ;SAVE Y REG.
        [interrupt processing]
        POP    Y      ;RESTORE Y REG.
        POP    X      ;RESTORE X REG.
        POP    A      ;RESTORE ACC.
        RETI          ;RETURN
```

General-purpose registers are saved or restored by using push and pop instructions.

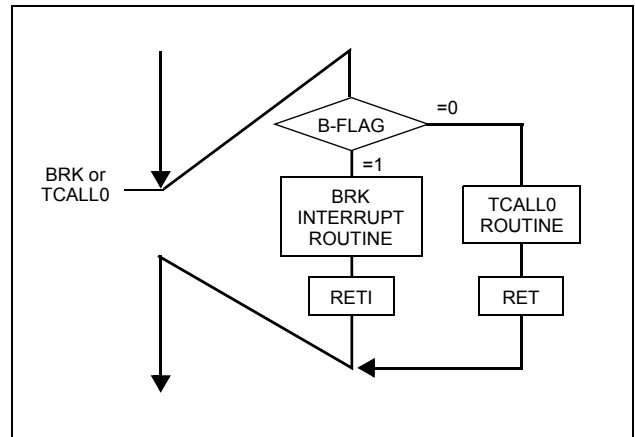


**17.2 BRK Interrupt**

Software interrupt can be invoked by BRK instruction, which has the lowest priority order.

Interrupt vector address of BRK is shared with the vector of TCALL 0 (Refer to Program Memory Section). When BRK interrupt is generated, B-flag of PSW is set to distinguish BRK from TCALL 0.

Each processing step is determined by B-flag as shown in Figure 17-4.



**Figure 17-4 Execution of BRK/TCALL0**

**17.3 Multi Interrupt**

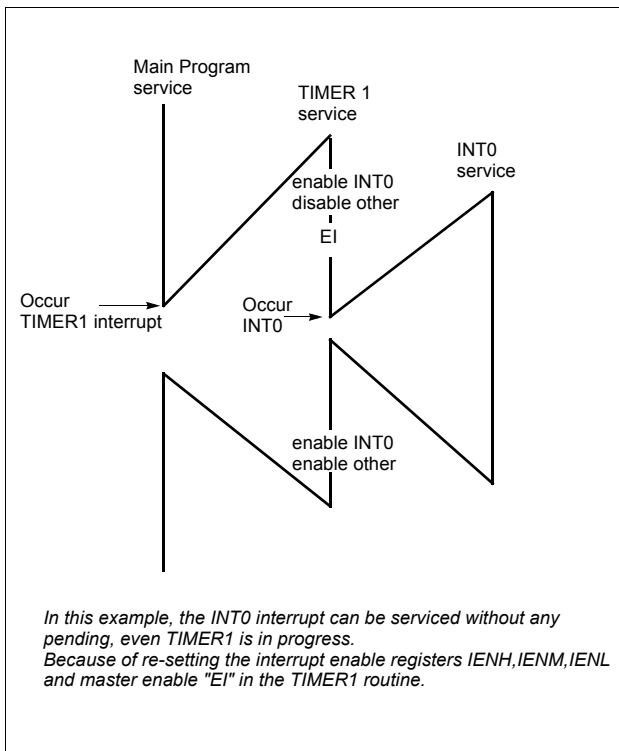
If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If requests of the interrupt are received at the same time simultaneously, an internal polling sequence determines by hardware which request is serviced.

However, multiple processing through software for special features is possible. Generally when an interrupt is accepted, the I-flag is cleared to disable any further interrupt. But as user sets I-flag in interrupt routine, some further interrupt can be serviced even if certain interrupt is in progress.

Example: Even though Timer1 interrupt is in progress, INTO interrupt serviced without any suspend.

```
TIMER1:  PUSH   A
         PUSH   X
         PUSH   Y
```

```
LDM   IENH, #40H ; Enable INTO only
LDM   IENM, #0   ; Disable other
LDM   IENL, #0   ; Disable other
EI    ; Enable Interrupt
:
:
:
:
LDM   IENH, #0FFH ; Enable all interrupts
LDM   IENM, #0FFH
LDM   IENL, #0F0H
POP   Y
POP   X
POP   A
RETI
```



**Figure 17-5 Execution of Multi Interrupt**

### 17.4 External Interrupt

The external interrupt on INT0, INT1, INT2 and INT3 pins are edge triggered depending on the edge selection register IEDS (address 0FC<sub>H</sub>) as shown in Figure 17-6.

The edge detection of external interrupt has three transition activated mode: rising edge, falling edge, and both edge.

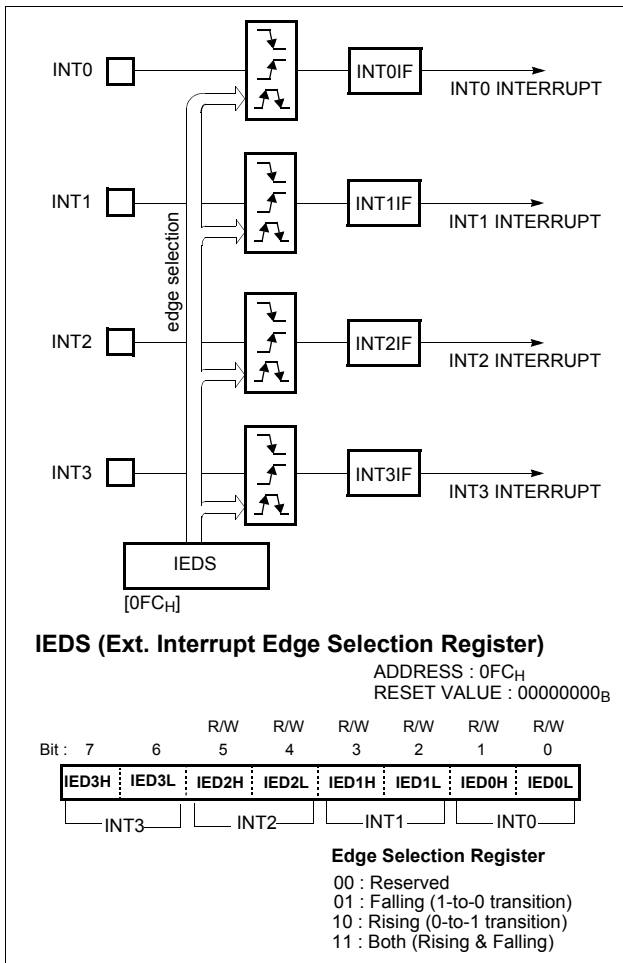


Figure 17-6 External Interrupt Block Diagram

Example: To use as an INT0

```

:
:
;**** Set port as an input port R0
LDM R0IO,#1101_1111B
;
;**** Set port as an interrupt port
LDM PSR0,#0100_0000B
;
;**** Set Falling-edge Detection
LDM IEDS,#0000_0001B
:
:
:

```

#### Response Time

The INT0, INT1, INT2 and INT3 edge are latched into INT0F, INT1F, INT2F and INT3F at every machine cycle. The values are not actually polled by the circuitry until the next machine cycle. If a request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The DIV itself takes twelve cycles. Thus, a maximum of twelve complete machine cycles elapse between activation of an external interrupt request and the beginning of execution of the first instruction of the service routine.

Interrupt response timings are shown in Figure 17-7.

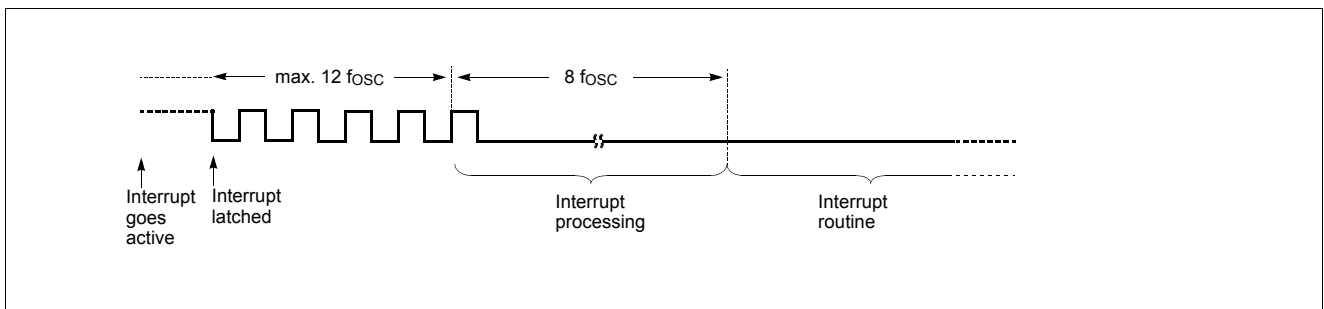


Figure 17-7 Interrupt Response Timing Diagram

### 18. LCD DRIVER

The MC81F8816/8616 has the circuit that directly drives the liquid crystal display (LCD) and its control circuit. The segment/common driver directly drives the LCD panel, and the LCD controller generates the segment/common signals according to the RAM which stores display data. VCL3 ~ VCL0 voltage are made by the internal bias resistor circuit.

The MC81F8816/8616 has the segment output port 36 pins (SEG0 ~ SEG35) and Common output port 8 pins (COM0 ~ COM7). If the LCDD0 bit of LCR is set to "1", COM4 ~ COM7 is used as SEG39 ~ SEG36.

The Figure 18-1 shows the configuration of the LCD driver.

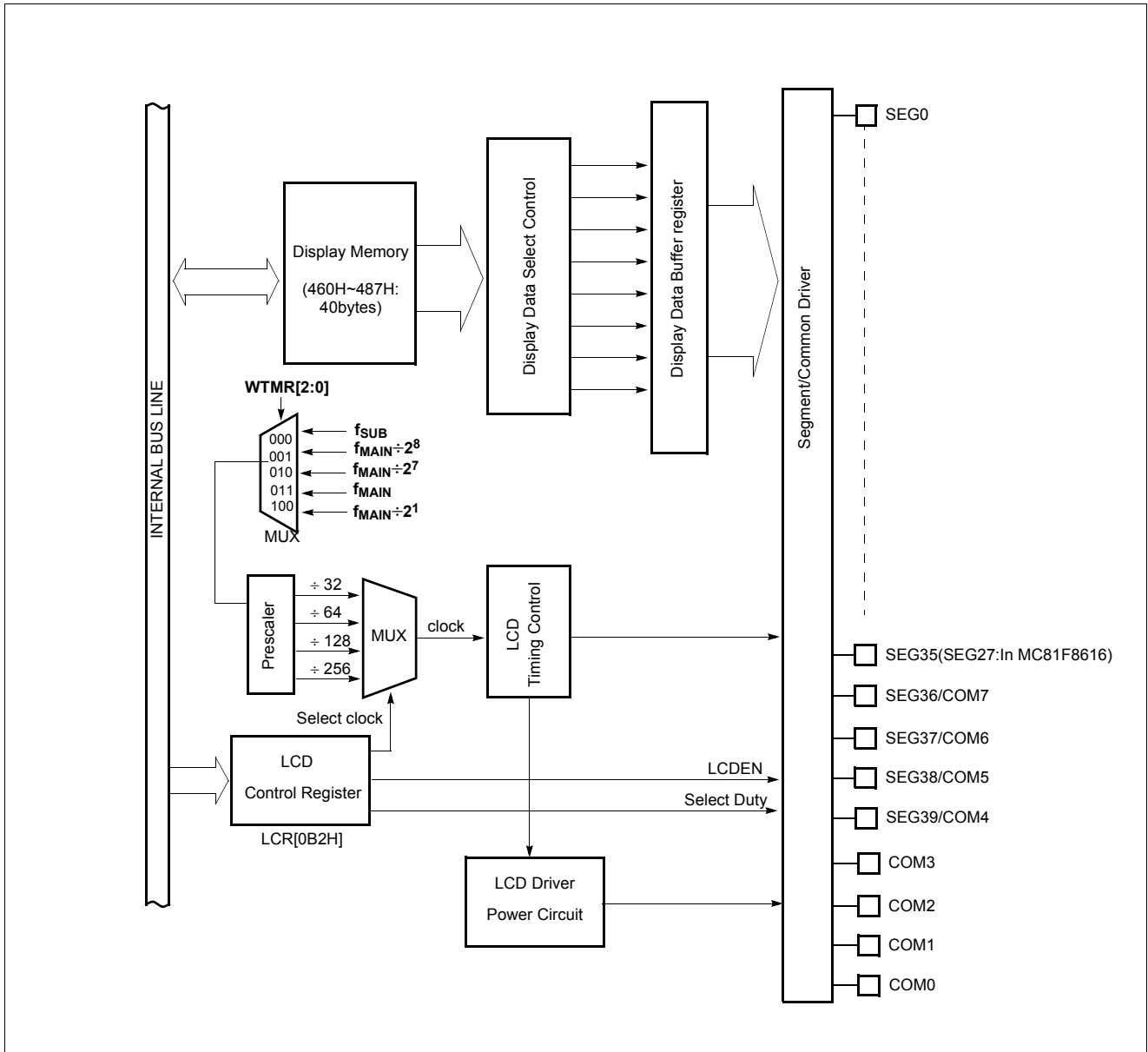


Figure 18-1 LCD Driver Block Diagram

### 18.1 Control of LCD Driver Circuit

The LCD driver is controlled by the LCD Control Register (LCR). The LCR[1:0] determines the frequency of COM signal scanning of each segment output. RESET clears the LCD control register LCR values to logic zero. The LCD SEG or COM ports are selected by setting corresponding

bits of R5PSR, R6PSR or R7PSR to “0”.

The LCD display can continue to operate during SLEEP and STOP modes if sub-frequency clock is used as LCD

clock source.

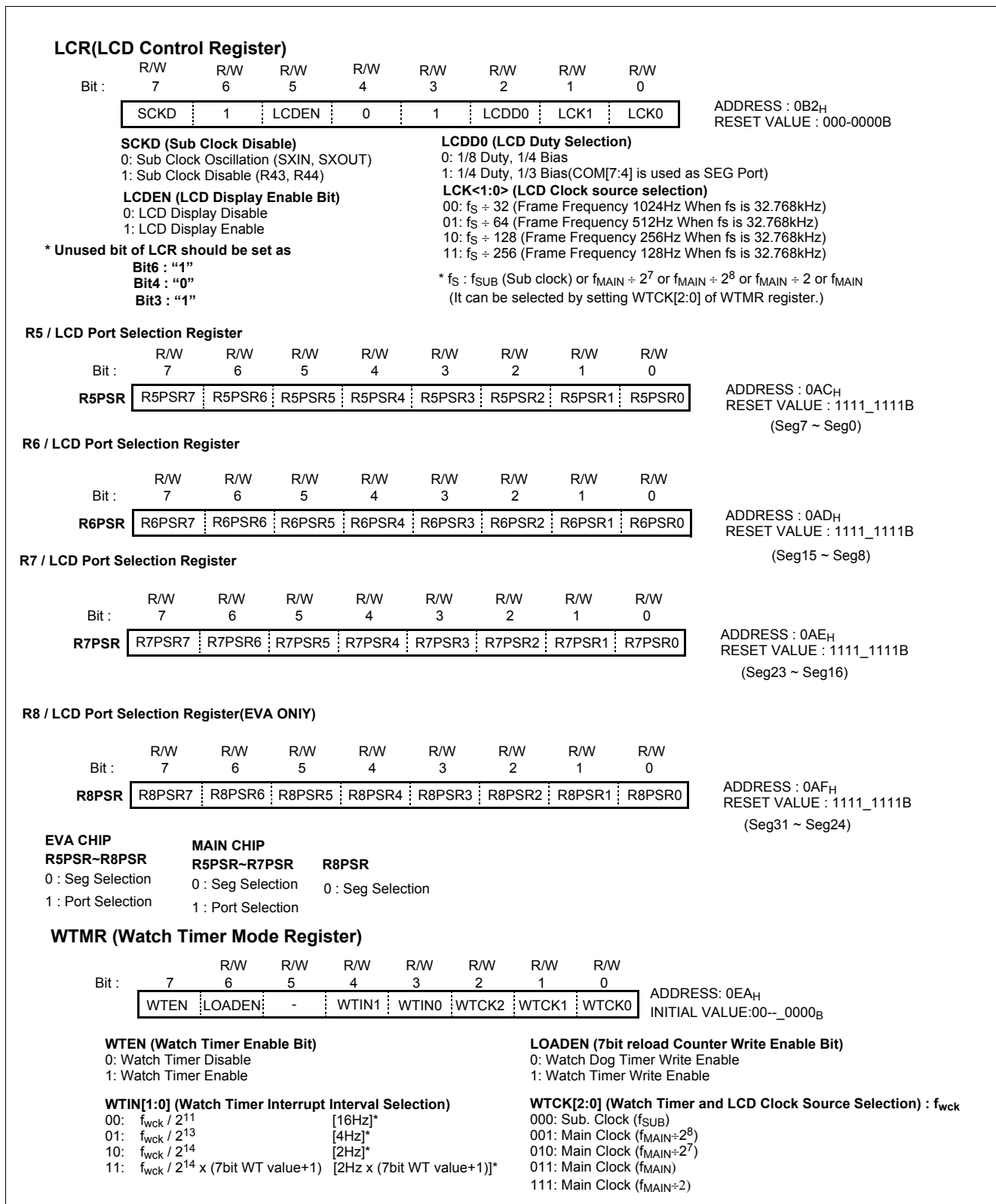


Figure 18-2 LCD Control Register



**Note:** If the SCKD is set to “1”, the SXIN and SXOUT pin is used as normal I/O pin R45, R46.

**Note:** When the Sub clock is used as internal bias source clock, stabilization time is needed. Normally, the stabilization time is need more than 500ms.

**Note:** When selecting Sub clock as the LCD clock source, the WTCK[2:0] bit of WTMR(Watch Timer Mode Register) should be set to “000” as well as SCKD bit of LCR be set to “0”.

**Note:** Bit 6, Bit 4, Bit 3 of LCR should be set to “1”, “0”, “1” respectively.

### Selecting Frame Frequency

Frame frequency is set to the base frequency as shown in the following Table 18-1. The  $f_S$  is selected to  $f_{SUB}$  (sub clock) which is 32.768kHz.

LCR[1:0]	LCD clock	Frame Frequency (Hz)	
		Duty = 1/4	Duty = 1/8
00	$f_{SUB} \div 32$	128	64
01	$f_{SUB} \div 64$	64	32
10	$f_{SUB} \div 128$	32	16
11	$f_{SUB} \div 256$	16	8

Table 18-1 Setting of LCD Frame Frequency

### The matters to be attended to use LCD driver

In reset state, LCD source clock is sub clock. So, when the power is supplied, the LCD display would be flickered be-

fore the oscillation of sub clock is stabilized. It is recommended to use LCD display on after the stabilization time of sub clock is considered enough.

## 18.2 LCD BIAS Control

The MC81F8816/8616 has internal Bias Circuit for driving LCD panel. It also has the contrast controller of 16 step.

The LCD Bias control register and internal Bias circuit is as shown in the Figure 18-3.

The SYS\_BOD[1:0] and BIF of LBCR register is used for controlling BOD. Refer to “27. Brown-out Detector (BOD)”

---

**Note:** *The self bias check reference can be applied to contrast adjustment with VDD voltage variation. Because the VDD voltage can be calculated by reading the ADC value of self bias check reference. Writing appropriate value to CTR[3:0] with VDD level, LCD contrast variation with VDD can be reduced.*

---

**LBCR(LCD Bias Control Register)**

Bit :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	CTR_S   CTR_DS3   CTR_DS2   CTR_DS1   CTR_DS0   SYS_BOD   SYS_BOD   BOD							

ADDRESS : 0B3<sub>H</sub>  
RESET VALUE : 01111000<sub>B</sub>

**CTR\_S (Voltage Source selection)**

- 0: Direct Voltage
- 1: Contrast Controller voltage

**SYS\_BOD<1:0> (Mode selection of BOD Result)**

- 00: Reset mode
- 10: Freeze mode

**BOD (BOD Flag)**

- 0: **BOD** No Detect
- 1: **BOD** Detect

\* **BOD** : Brown-out detector

**CTR\_DS<3:0> (Contrast Controller Level Selection)**

- 0000: VCL3 = VDD / 2
- 0001: VCL3 = VDD / 2 + VDD \* ( 1 / 30 )
- 0010: VCL3 = VDD / 2 + VDD \* ( 2 / 30 )
- 0011: VCL3 = VDD / 2 + VDD \* ( 3 / 30 )
- 0100: VCL3 = VDD / 2 + VDD \* ( 4 / 30 )
- 0101: VCL3 = VDD / 2 + VDD \* ( 5 / 30 )
- 0110: VCL3 = VDD / 2 + VDD \* ( 6 / 30 )
- 0111: VCL3 = VDD / 2 + VDD \* ( 7 / 30 )
- 1000: VCL3 = VDD / 2 + VDD \* ( 8 / 30 )
- 1001: VCL3 = VDD / 2 + VDD \* ( 9 / 30 )
- 1010: VCL3 = VDD / 2 + VDD \* ( 10 / 30 )
- 1011: VCL3 = VDD / 2 + VDD \* ( 11 / 30 )
- 1100: VCL3 = VDD / 2 + VDD \* ( 12 / 30 )
- 1101: VCL3 = VDD / 2 + VDD \* ( 13 / 30 )
- 1110: VCL3 = VDD / 2 + VDD \* ( 14 / 30 )
- 1111: VCL3 = VDD

**Block Diagram of LCD BIAS**

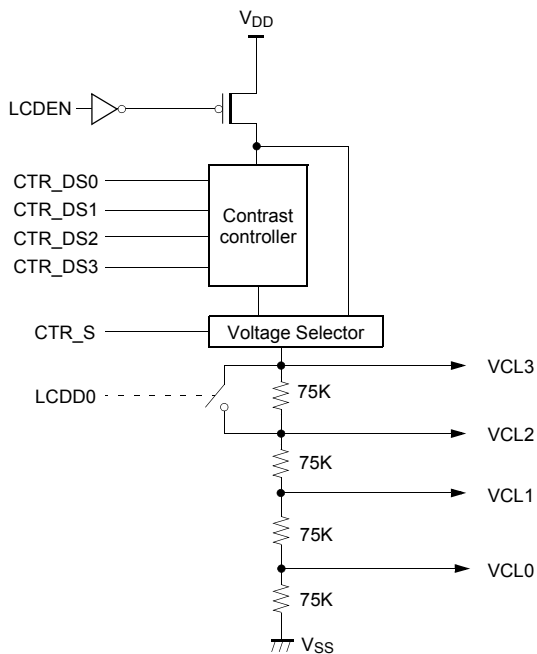


Figure 18-3 LCD Bias Control

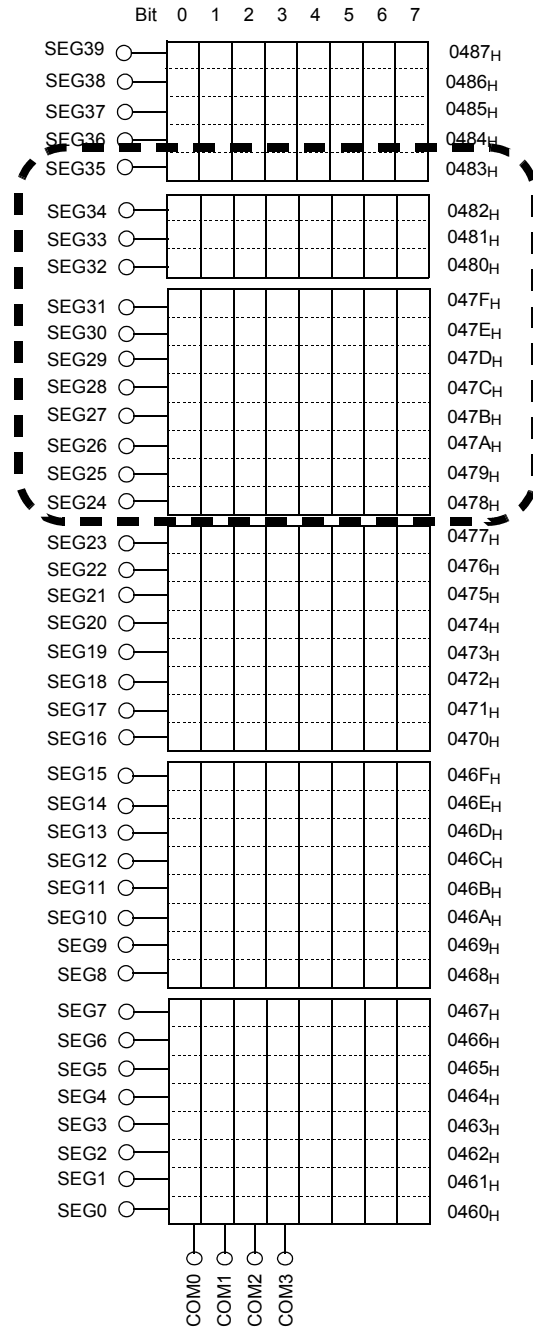
### 18.3 LCD Display Memory

Display data are stored to the display data area (page 4) in the data memory.

The display data which stored to the display data area (address 0460<sub>H</sub>-0487<sub>H</sub>) are read automatically and sent to the LCD driver by the hardware. The LCD driver generates the segment signals and common signals in accordance with the display data and drive method. Therefore, display patterns can be changed by only overwriting the contents of the display data area with a program. The table look up instruction is mainly used for this overwriting.

Figure 18-4 shows the correspondence between the display data area and the SEG/COM pins. The LCD lights when the display data is “1” and turn off when “0”.

The SEG data for display is controlled by RPR (RAM Paging Register).



Only supported in MC81F8616

Figure 18-4 LCD Display Memory

## 18.4 Control Method of LCD Driver

### Initial Setting

Flow chart of initial setting is shown in Figure 18-5.

Example: Driving of LCD

```

Select Frame Frequency      LDM   LCR, #4DH      ;fF=64Hz, 1/4 duty (fSUB= 32.768kHz)
                          :
                          LDM   RPR, #4      ;Select LCD Memory(4 page)
                          SETG
Clear LCD Display Memory   C_LCD1: LDX   #60H
                          LDA   #0          ;RAM Clear
                          ; (0460H->0487H)
                          STA   {X}+
                          CMPX  #088H
                          BNE   C_LCD1
                          CLRG
                          :
Turn on LCD                 SET1   LCR.5      ;Enable display
                          :
    
```

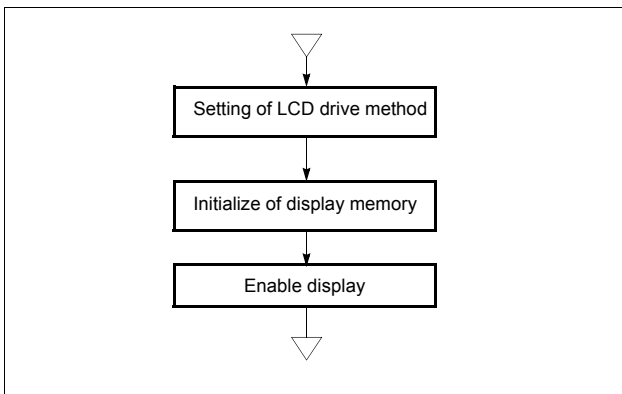


Figure 18-5 Initial Setting of LCD Driver

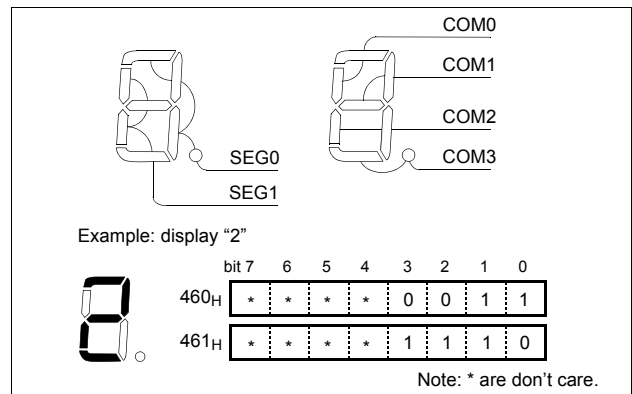


Figure 18-6 Example of Connection COM & SEG

### Display Data

Normally, display data are kept permanently in the program memory and then stored at the display data area by the table look-up instruction. This can be explained using character display with 1/4 duty LCD as an example as well as any LCD panel. The COM and SEG connections to the LCD and display data are the same as those shown in Figure 18-6. Following is showing the programming example for displaying character.

**Note:** When power on RESET, sub oscillation start up time is required. Enable LCD display after sub oscillation is stabilized, or LCD may occur flicker at power on time shortly.

Write into the LCD Memory	<pre> : : CLRG : LDX#&lt;DISPRAM ;Address included the data : ;to be displayed. GOLCD: LDA{X} : TAY : LDA!FONT+Y ;LOAD FONT DATA : LDMRPR,#4 ;Set RPR = 4 to access LCD : SETG ;Set Page 4 : LDX#60H : STA{X}+ ;LOWER 4 BITS OF ACC. seg0 : XCN : STA{X} ;UPPER 4 BITS OF ACC. seg1 : CLRG ;Set Page = 0 : : </pre>
Font data	<pre> FONT DB 1101_0111B; "0" DB 0000_0110B; "1" DB 1110_0011B; "2" DB 1010_0111B; "3" DB 0011_0110B; "4" DB 1011_0101B; "5" DB 1111_0101B; "6" DB 0000_0111B; "7" DB 1111_0111B; "8" DB 0011_0111B; "9" </pre>

**LCD Waveform**

The LCD duty(1/4, 1/8) can be selected by LCR register. The example of 1/4 duty, 1/3 bias are shown in shown Figure 18-7.

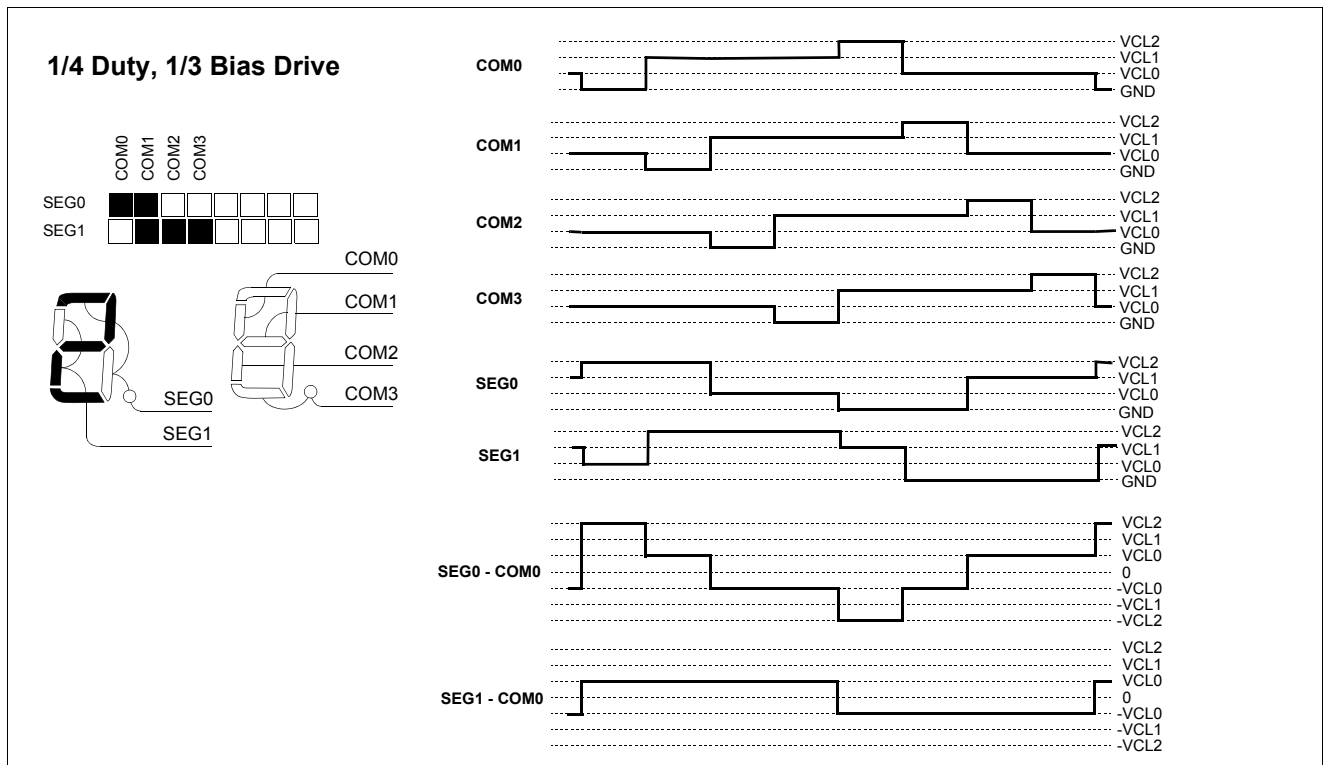


Figure 18-7 Example of LCD drive output

### 18.5 Duty and Bias Selection of LCD Driver

4 kinds of driving methods can be selected by LCDD[1:0] (bits 3 and 2 of LCD control register) and connection of BIAS pin exter-

nally. Figure 18-8 shows typical driving waveforms for LCD.).

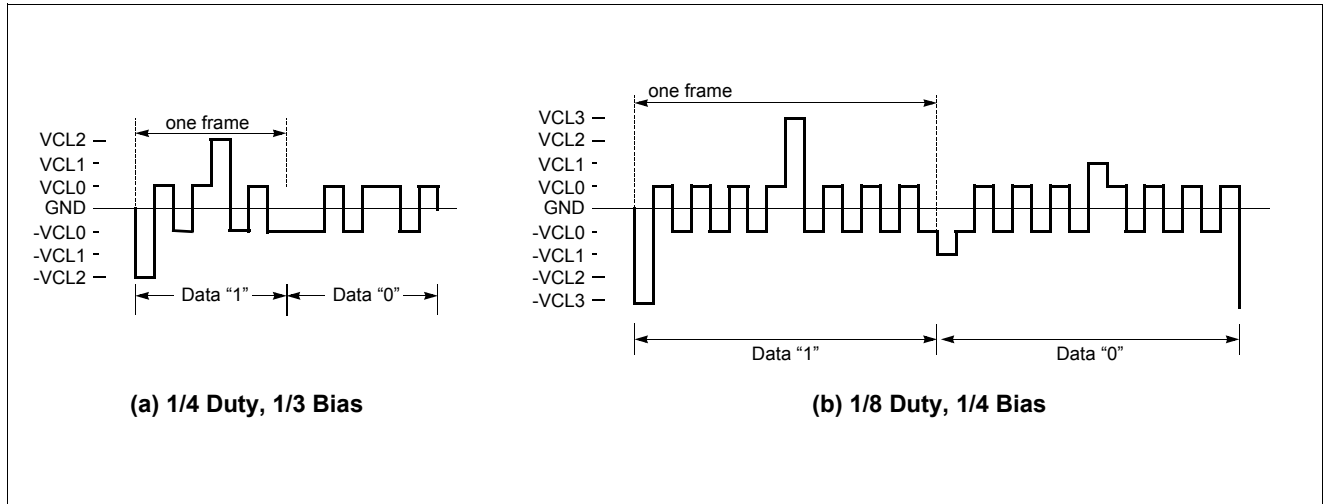


Figure 18-8 LCD Drive Waveform (Voltage COM-SEG Pins)

### 19. SERIAL PERIPHERAL INTERFACE (SPI)

The serial Input/Output is used to transmit/receive 8-bit data serially. The Serial Input/Output(SPI) module is a serial interface useful for communicating with other peripheral of microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. This SPI is 8-bit clock synchronous type and consists of serial I/O data register, serial I/O mode register, clock selection circuit, octal counter and control

circuit as illustrated in Figure 19-1. The SO pin is designed to input and output. So the Serial I/O(SPI) can be operated with minimum two pin. Pin R11/ACK/SCK, R13/RX0/SI, and R12/TX0/SO pins are controlled by the Serial Mode Register. The contents of the Serial I/O data register can be written into or read out by software. The data in the Serial Data Register can be shifted synchronously with the transfer clock signal.

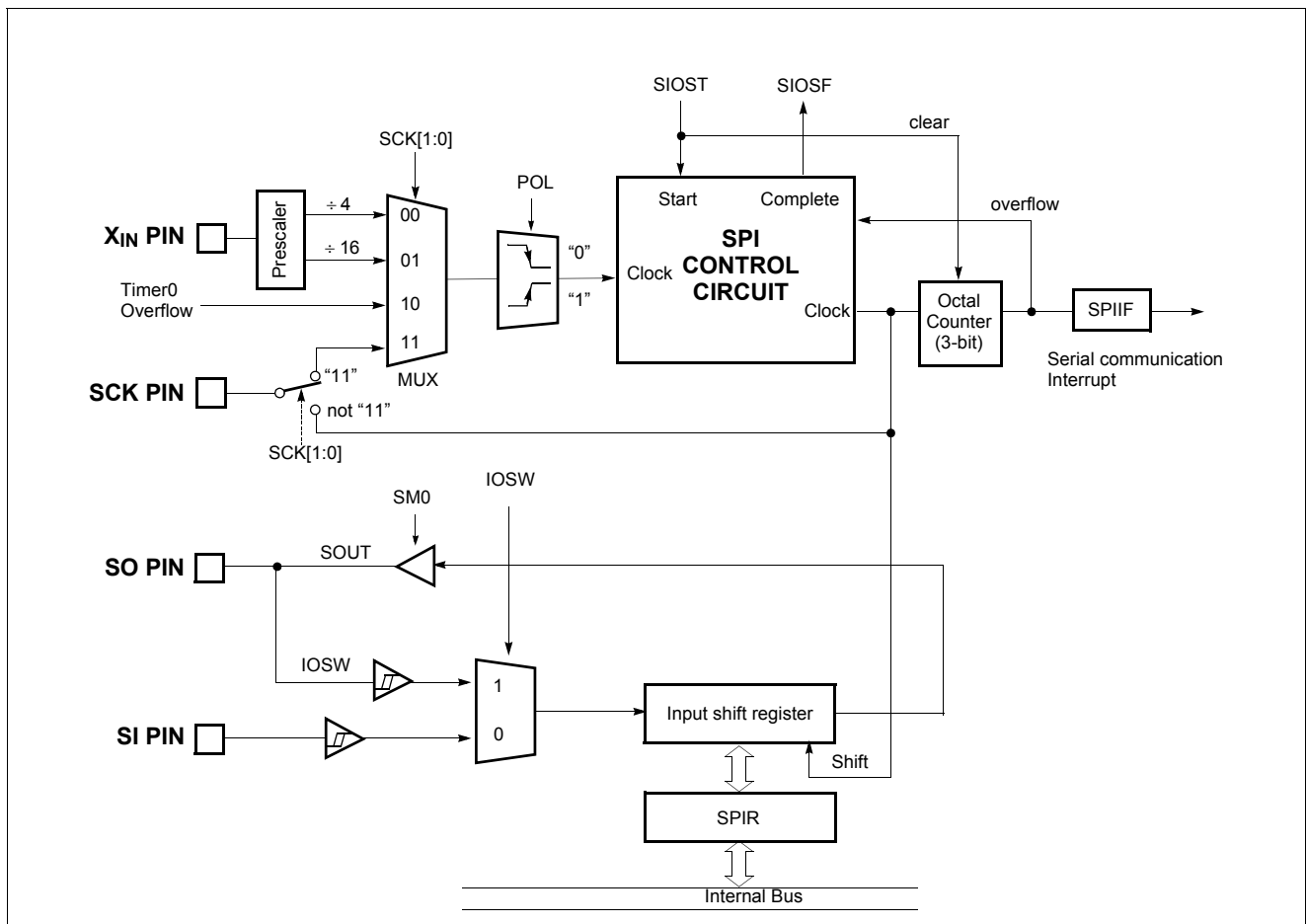
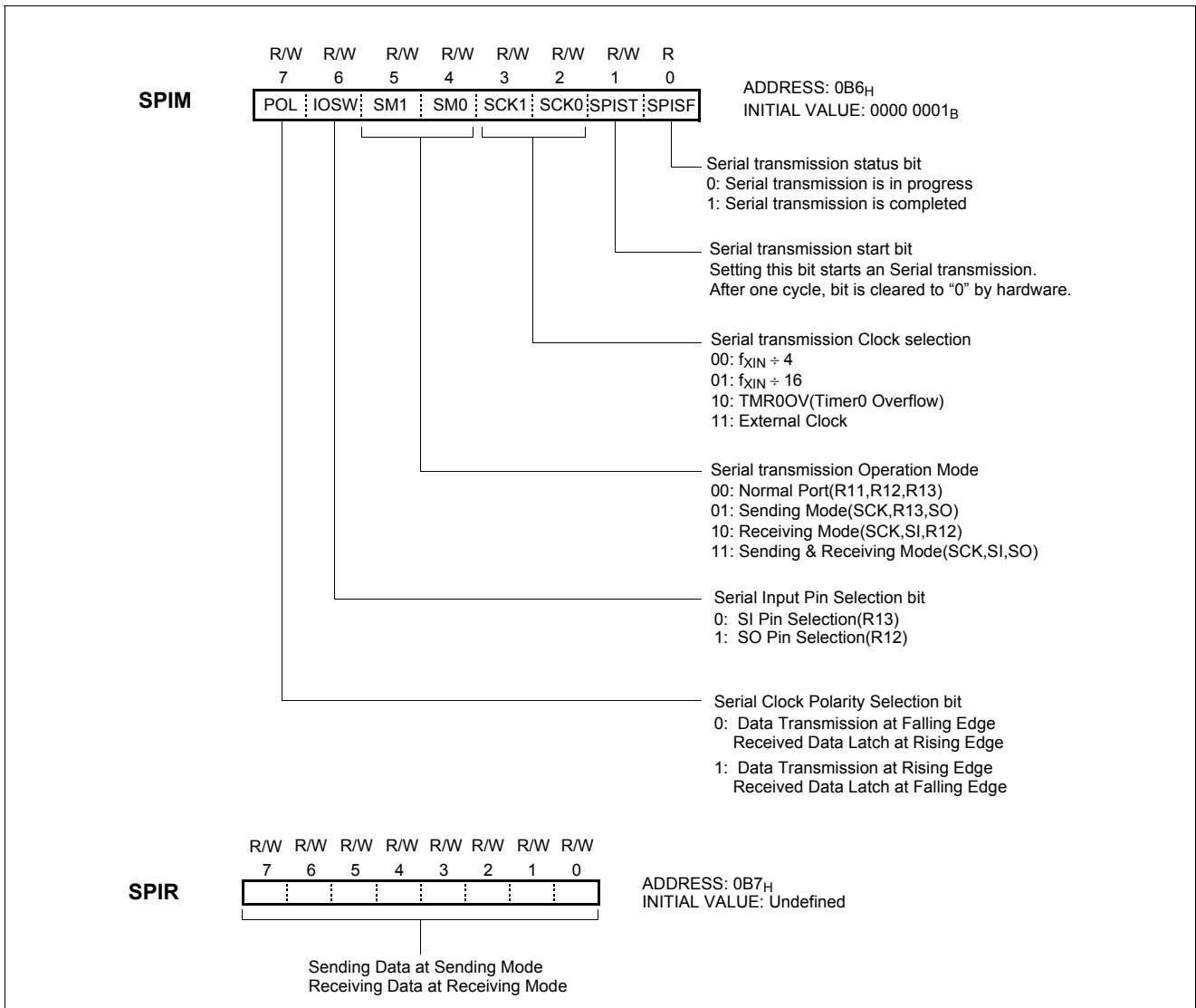


Figure 19-1 SPI Block Diagram

Serial I/O Mode Register(SPIM) controls serial I/O function. According to SCK1 and SCK0, the internal clock or external clock can be selected.

Serial I/O Data Register(SPIR) is an 8-bit shift register. First LSB is send or is received.





**Figure 19-2 SPI Control Register**

### 19.1 Transmission/Receiving Timing

The serial transmission is started by setting SPIST(bit1 of SPIM) to "1". After one cycle of SCK, SPIST is cleared automatically to "0". At the default state of POL bit clear, the serial output data from 8-bit shift register is output at falling edge of SCLK, and input data is latched at rising edge of SCLK pin (Refer to Figure 19-3). When transmission clock is counted 8 times, serial I/O counter is cleared as '0'. Transmission clock is halted in "H" state and serial I/O interrupt(SPIIF) occurred.

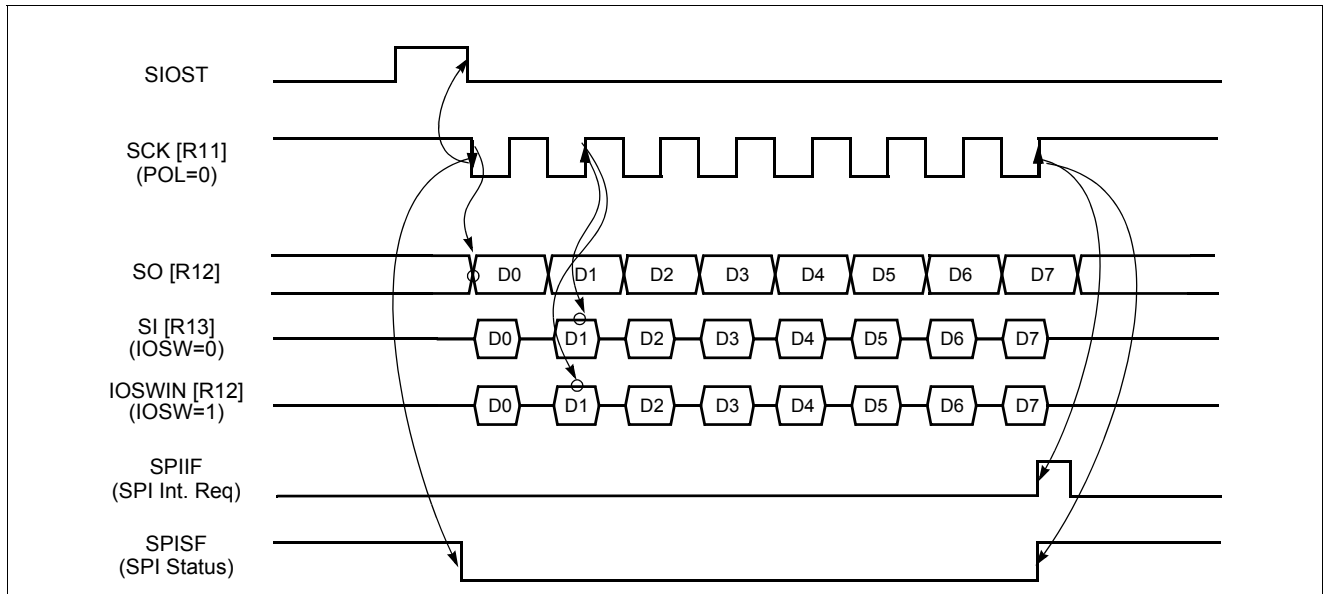


Figure 19-3 Serial I/O Timing Diagram at POL=0

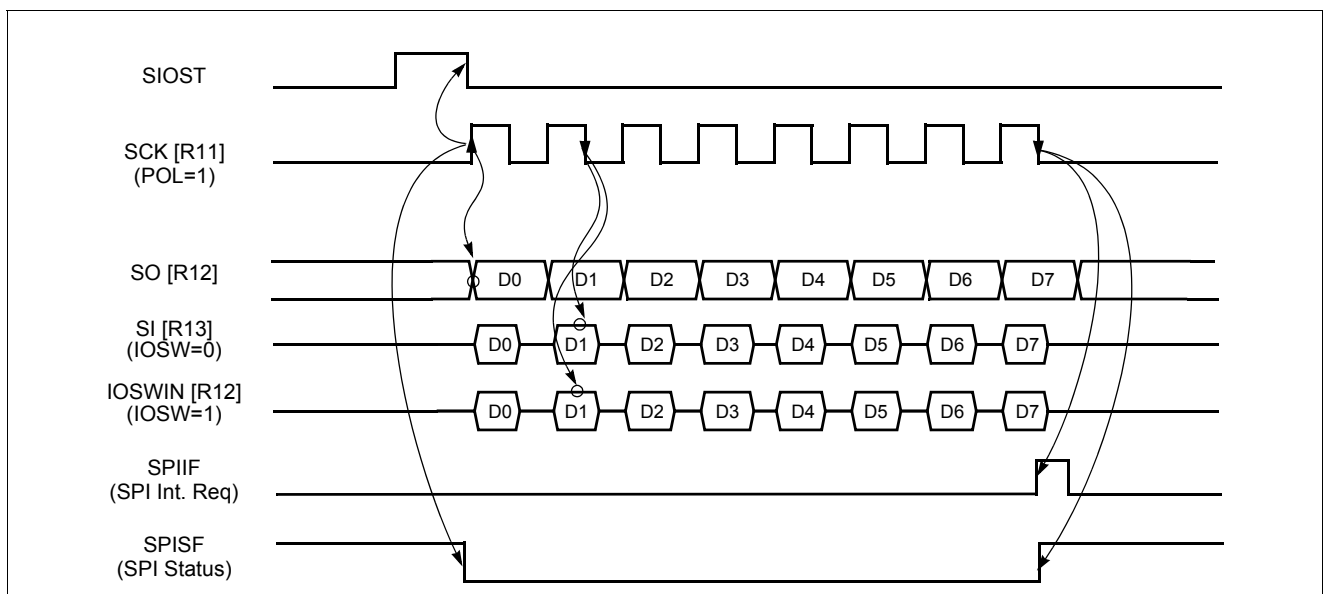


Figure 19-4 Serial I/O Timing Diagram at POL=1

## 19.2 The usage of Serial I/O

1. Select transmission/receiving mode.
2. In case of sending mode, write data to be send to SPIR.
3. Set SPIST to "1" to start serial transmission.
4. The SPI interrupt is generated at the completion of SPI and SPIIF is set to "1".
5. In case of receiving mode, the received data is acquired by reading the SPIR.
6. When using polling method, the completion of 1 byte serial communication can be checked by reading SPIST and SPISF. As shown in example code, wait until SPIST is changed to "0" and then wait the SPISF is

changed to “1” for completion check.

```

LDM SPIR,#0AAh ;set tx data
LDM SPIM,#0011_1100b;set SPI mode
NOP
LDM SPIM,#0011_1110b;SPI Start
SPI_WAIT:
NOP
BBS SPIST,SIO_WAIT ;wait first edge
BBC SPISF,SIO_WAIT ;wait complete
    
```

**Note:** When external clock is used, the frequency should be less than 1MHz and recommended duty is 50%. If both transmission mode is selected and transmission is performed simultaneously, error may occur.

### 19.3 The Method to Test Correct Transmission

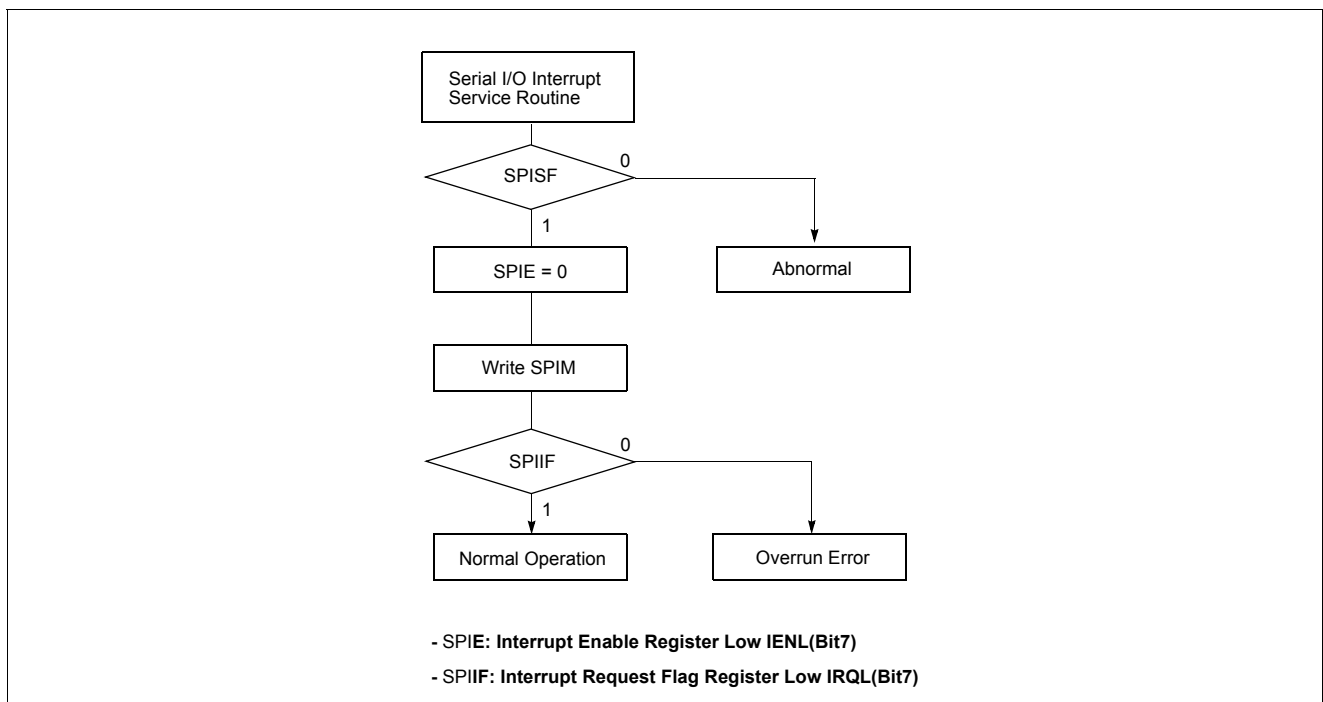


Figure 19-5 Serial IO Method to Test Transmission

## 20. INTER IC COMMUNICATION (I2C)

Generation of clock signals on the I2C-bus is always the responsibility of master devices; each master generates its own clock signals when transferring data on the bus. Bus clock signals from a master can only be altered when they are stretched by a slow-slave device holding-down the clock line, or by another master when arbitration occurs. Both SDA and SCL are bi-directional lines, connected to a positive supply voltage via a current-source or pull-up resistor. When the bus is free, both lines are HIGH. The output stages of devices connected to the bus must have an open-

drain or open-collector to perform the wired-AND function. Data on the I2C-bus can be transferred at rates of up to 100 kbit/s in the Standard-mode, up to 400 kbit/s in the Fast-mode, or up to 3.4 Mbit/s in the High-speed mode. The number of interfaces connected to the bus is solely dependent on the bus capacitance limit of 400 pF. For information on High-speed mode master devices,

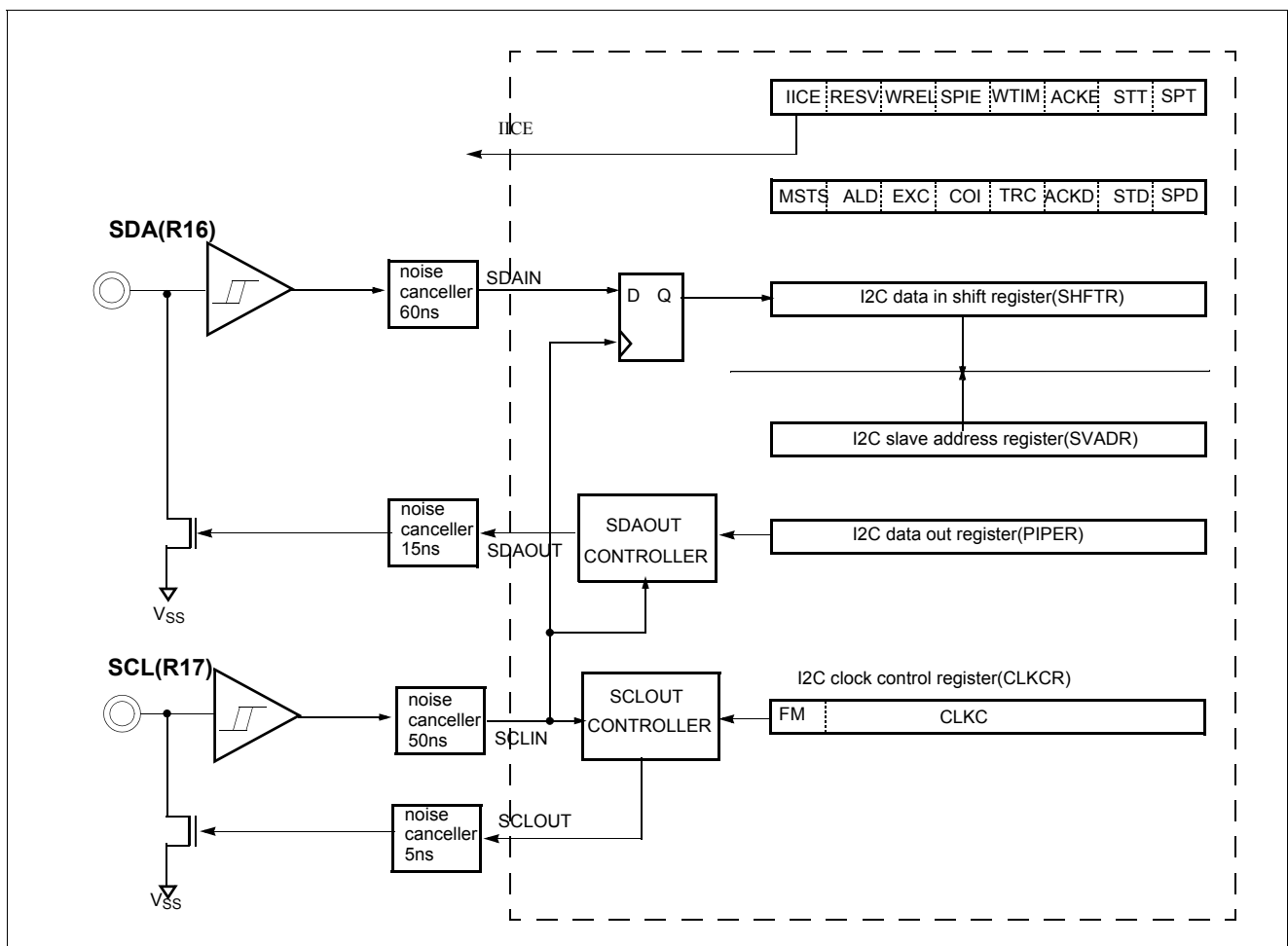


Figure 20-1 I2C Block Diagram

**I2CMR (I2C Mode Control Register)**

Bit :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	I2CE	RESV	WREL	SPIE	WTIM	ACKE	STT	SPT

ADDRESS : 090<sub>H</sub>  
RESET VALUE : 0-001000<sub>B</sub>

I2CE(I2C Enable)      WREL(cancel wait)      WTIM(when interrupt request occurs)      STT(start condition generation)  
 0 : disable            0 : no operation            0 : after 8th clock's falling edge      0 : disable  
 1 : enable            1 : release scl to high      1 : after ACK clock's falling edge      1 : enable

RESV                    SPIE(interrupt enable after stop detection)      ACKE(acknowledge enable)      SPT(stop condition generation)  
 0 : -                    0 : disable                    0 : no acknowledge            0 : disable  
 1 : -                    1 : enable                    1 : acknowledge                1 : enable

**I2CSR (I2C Status Register)**

Bit :	R	R	R	R	R	R	R	R
	7	6	5	4	3	2	1	0
	MSTS	ALD	EXC	COI	TRC	ACKD	STD	SPD

ADDRESS : 091<sub>H</sub>  
RESET VALUE : 00000000<sub>B</sub>

MSTS(master device status)      EXC(general call detection)      TRC(transmission status)      STD(start condition detection)  
 0 : no master                    0 : no detected                    0 : no transmitter                    0 : no detected  
 1 : master                        1 : detected                        1 : transmitter                        1 : detected

ALD(arbitration lost detection)      COI(selected as slave)      ACKD(acknowledge detection)      SPD(stop condition detection)  
 0 : no arbitration lost            0 : no selected                    0 : no detected                    0 : no detected  
 1 : arbitration lost                1 : selected                        1 : detected                        1 : detected

**I2CCR (I2C Clock Control Register)**

Bit :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	FM	CLK6	CLK5	CLK4	CLK3	CLK2	CLK1	CLK0

ADDRESS : 092<sub>H</sub>  
RESET VALUE : 11111111<sub>B</sub>

FM(protocol mode select)                    CLK6~CLK0(pre scale value)  
 0 : standard mode                    Fsc1 : Fsys/4N  
 1 : fast mode                        N: Pre scale value(I2CCR[6:0])

**I2CPR (I2C Pipe and Shift Register)**

Bit :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	PP7	PP6	PP5	PP4	PP3	PP2	PP1	PP0

ADDRESS : 093<sub>H</sub>  
RESET VALUE : 11111111<sub>B</sub>

note : shift and pipe register have the same address  
 shift register is only readable and pipe register is only writable

**I2CAR (I2C Slave Address Register)**

Bit :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	7	6	5	4	3	2	1	0
	SVAD6	SVAD5	SVAD4	SVAD3	SVAD2	SVAD1	SVAD0	RESA

ADDRESS : 094<sub>H</sub>  
RESET VALUE : 00000000<sub>B</sub>

note : SVAD is a 7bit slave address

**Table 20-1 I2C Enable Registers**

## 20.1 Bit Transfer

Due to the variety of different technology devices (CMOS, NMOS, bipolar) which can be connected to the I<sup>2</sup>C-bus, the levels of the logical '0' (LOW) and '1' (HIGH) are not fixed and depend on the associated level of V<sub>DD</sub> (see Section 15 for electrical specifications). One clock pulse is generated for each data bit transferred. The data on the SDA line must be

stable during the HIGH period of the clock. The HIGH or LOW state of the data line can only change when the clock signal on the SCL line is LOW (see Figure 20-2)

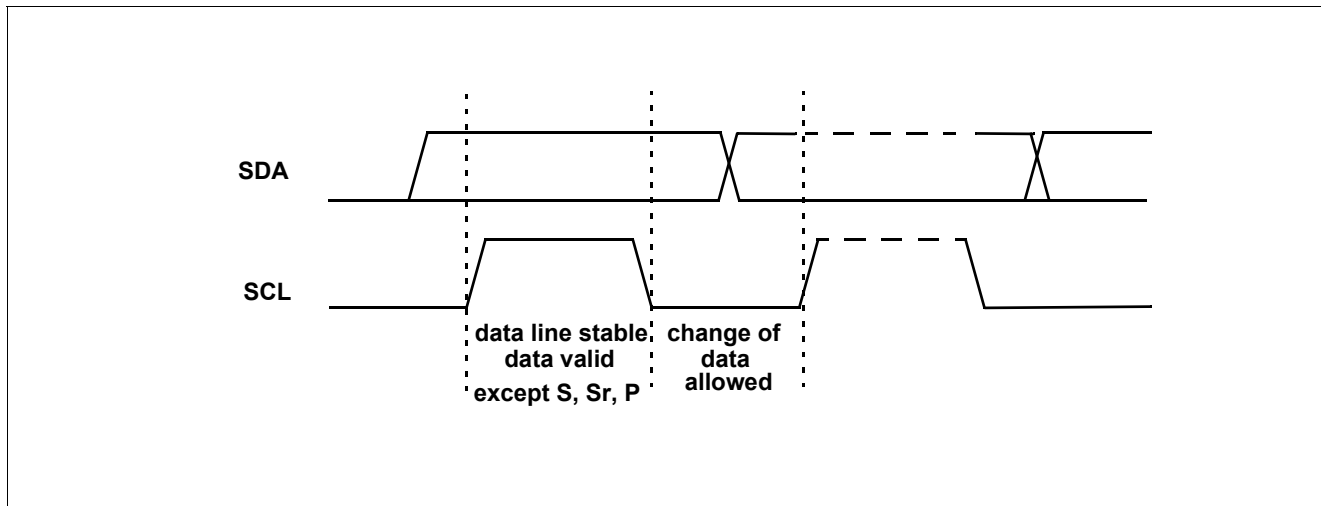


Figure 20-2 Bit transfer on I<sup>2</sup>C bus

## 20.2 Start/Stop Conditions

Within the procedure of the I<sup>2</sup>C-bus, unique situations arise which are defined as START (S) and STOP (P) conditions (see Figure 20-3). A HIGH to LOW transition on the SDA line while SCL is HIGH is one such unique case. This situation indicates a START condition. A LOW to HIGH transition on the SDA line while SCL is HIGH defines a STOP condition. START and STOP conditions are always generated by the master. The bus is considered to be busy after the START condition. The bus is considered to be free again a certain time after the STOP condition. The bus stays busy

if a repeated START (Sr) is generated instead of a STOP condition. In this respect, the START (S) and repeated START (Sr) conditions are functionally identical. For the remainder of this document, therefore, the S symbol will be used as a generic term to represent both the START and repeated START conditions, unless Sr is particularly relevant. Detection of START and STOP conditions by devices connected to the bus is easy if they incorporate the necessary interfacing hardware. However, microcontrollers with no such interface have to sample the SDA line at

least twice per clock period to sense the transition.

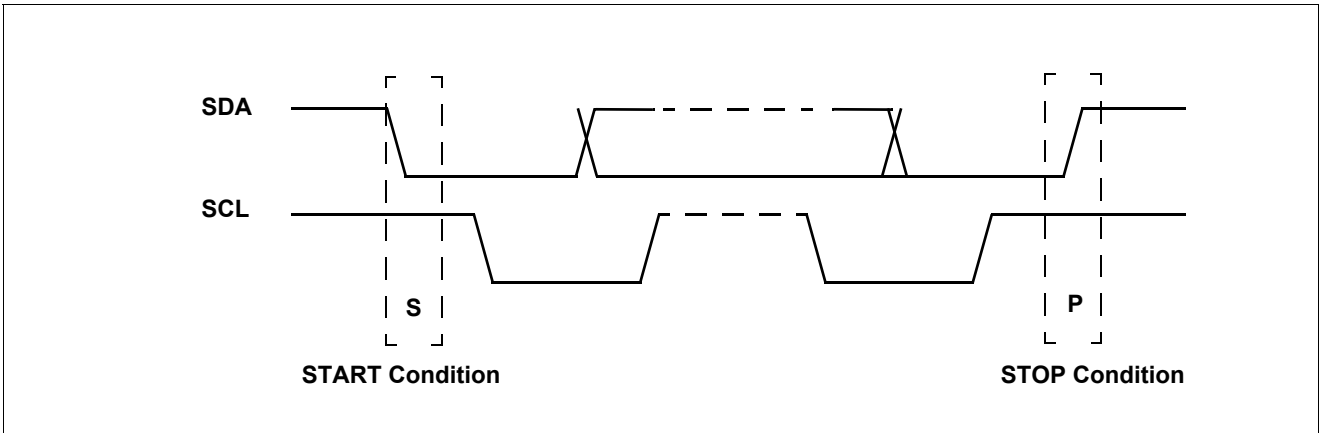


Figure 20-3 Start and Stop condition

### 20.3 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 unrestricted. Each byte has to be followed by an acknowledge bit. Data is transferred with the most significant bit (MSB) first (see Figure 20-4). If a slave can't receive or transmit another complete byte of data until it has performed some other function, for example servicing an internal interrupt, 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. In some cases, it's permitted to use a different format from the I2C-bus format (for CBUS compatible devices for example). A message which starts with such an address can be terminated by generation of a STOP condition, even during the transmission of a byte. In this case, no acknowledge is gen-

erated .

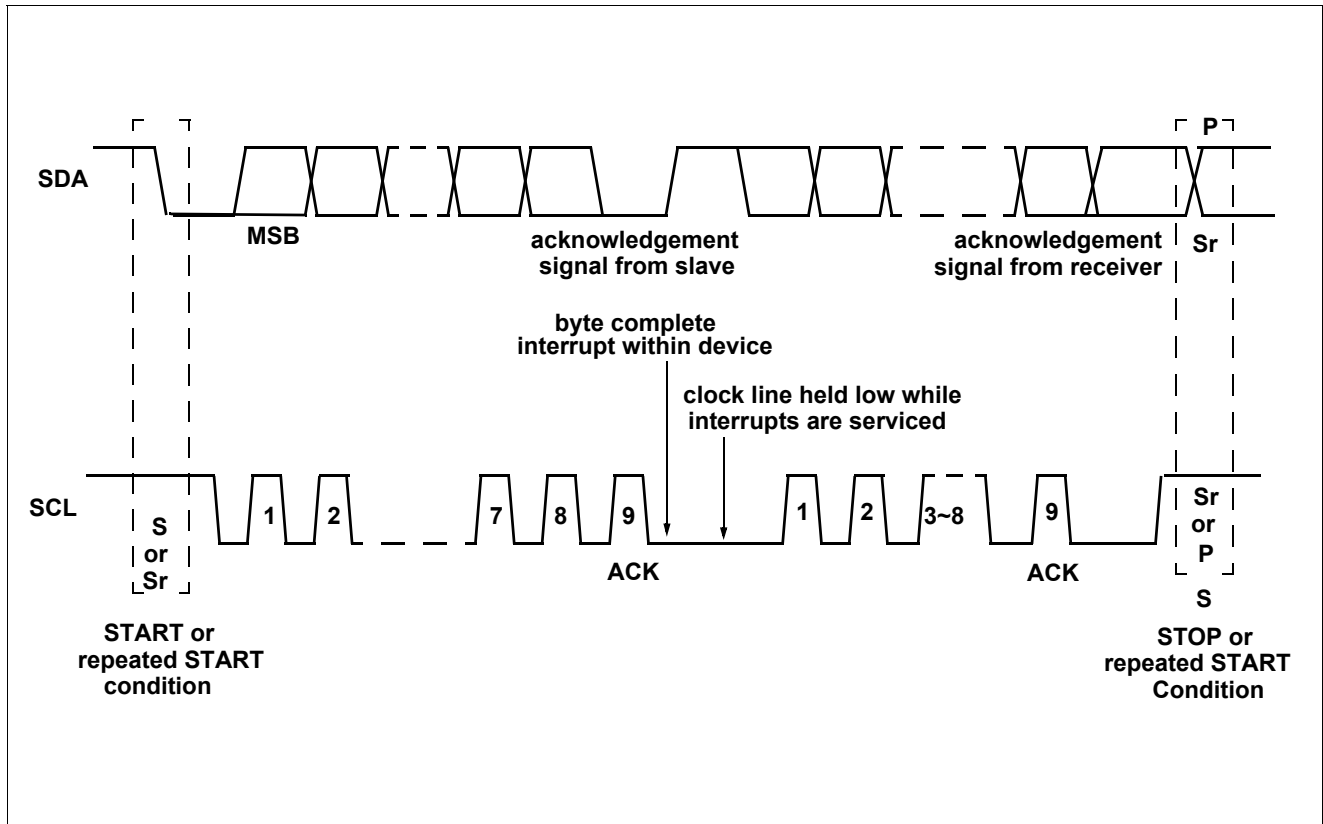


Figure 20-4 Data transfer on I2C bus

## 20.4 Acknowledge

Data transfer with acknowledge is obligatory. 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 (see Figure 20-5). Of course, set-up and hold times must also be taken into account. Usually, a receiver which has been addressed is obliged to generate an acknowledge after each byte has been received, except when the message starts with a CBUS address. When a slave doesn't acknowledge the slave address (for example, it's unable to receive or transmit because it's performing some real-time function), the data line must be left HIGH by the slave. 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 slave-receiver does acknowledge the slave address but, some time later in the transfer cannot receive any more data bytes, the master must again abort the transfer. This is indicated by the slave generating the not-acknowledge on the first byte to follow. The slave leaves the data line HIGH and the master generates a STOP or a repeated START condition. 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.



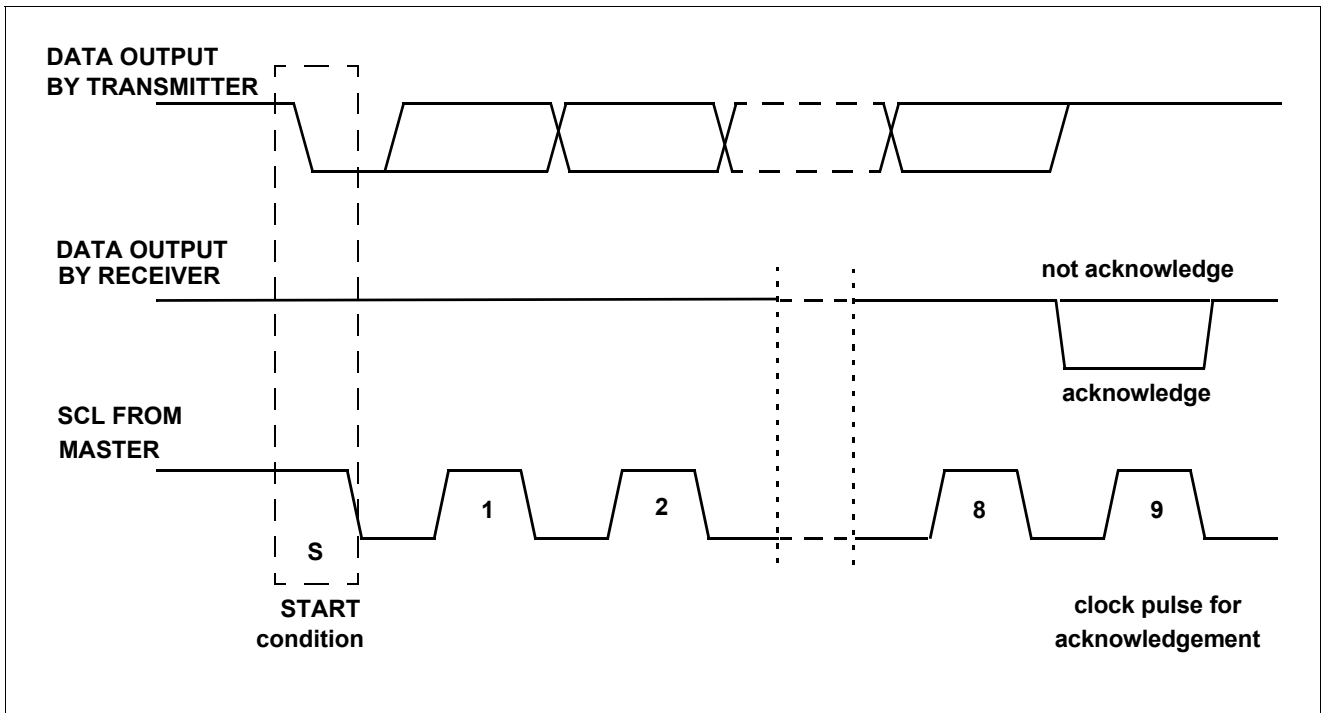


Figure 20-5 Acknowledge transfer on I2C bus

## 20.5 Synchronization/Arbitration

All masters generate their own clock on the SCL line to transfer messages on the I2C-bus. Data is only valid during the HIGH period of the clock. A defined clock is therefore needed for the bit-by-bit arbitration procedure to take place. Clock synchronization is performed using the wired-AND connection of I2C 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, once a device clock has gone LOW, it will hold the SCL line in that state until the clock HIGH state is reached (see Figure 20-6). 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. The SCL line will therefore be held LOW by the device with the longest LOW period. Devices with shorter LOW periods enter a HIGH wait-state during this time. When all devices concerned have counted off their LOW period, the clock line will be released and go HIGH. There will then be no difference between the device clocks and the state of the SCL line, and all the devices will start counting their HIGH periods. The first device to complete its HIGH period will again pull the SCL line LOW. 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 within the minimum hold time ( $t_{HD,STA}$ ) of the START condition which results in a defined START condition to the bus. 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 stage because the level on the bus doesn't correspond to its own level. Arbitration can continue for many bits. Its first stage is comparison of the address bits (addressing information is given in Sections 10 and 14). If the masters are each trying to address the same device, arbitration continues with comparison of the data-bits if they are master-transmitter, or acknowledge-bits if they are master-receiver. Because address and data information on the I2C-bus is determined by the winning master, no information is lost during the arbitration process. A master that loses the arbitration can generate clock pulses until the end of the byte in which it loses the arbitration. As an Hs-mode master has a unique 8-bit master code, it will always finish

the arbitration during the first byte. If a master also incorporates a slave function and it loses arbitration during the addressing stage, it's possible that the winning master is trying to address it. The losing master must therefore switch over immediately to its slave mode. Figure 20-7 shows the arbitration procedure for two masters. Of course, more may be involved (depending on how many masters are connected to the bus). The moment there is a difference between the internal data level of the master generating DATA 1 and the actual level on the SDA line, its data output is switched off, which means that a HIGH output level is then connected to the bus. This will not affect the data transfer initiated by the winning master. Since control of the I<sup>2</sup>C-bus is decided solely on the address or master code and data sent by competing masters, there is no central master, nor any order of priority on the bus. Special at-

tention must be paid if, during a serial transfer, the arbitration procedure is still in progress at the moment when a repeated START condition or a STOP condition is transmitted to the I<sup>2</sup>C-bus. If it's possible for such a situation to occur, the masters involved must send this repeated START condition or STOP condition at the same position in the format frame. In other words, arbitration isn't allowed between:

- A repeated START condition and a data bit
- A STOP condition and a data bit
- A repeated START condition and a STOP condition.

Slaves are not involved in the arbitration procedure.

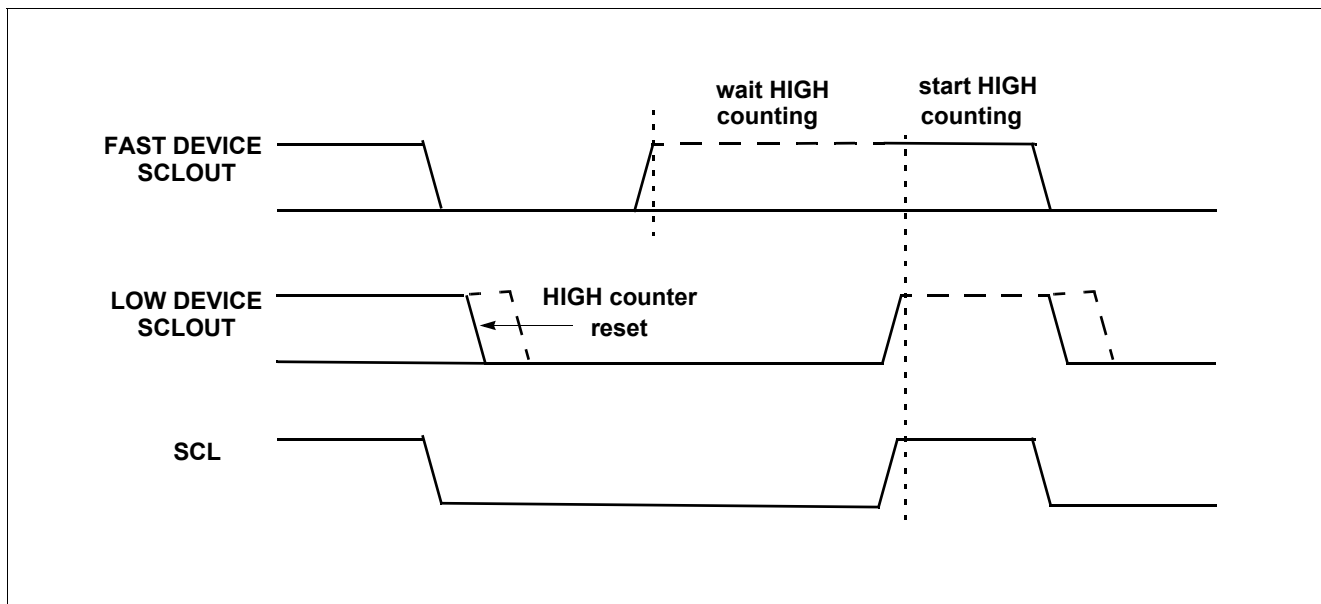


Figure 20-6 Clock synchronization during the arbitration procedure.

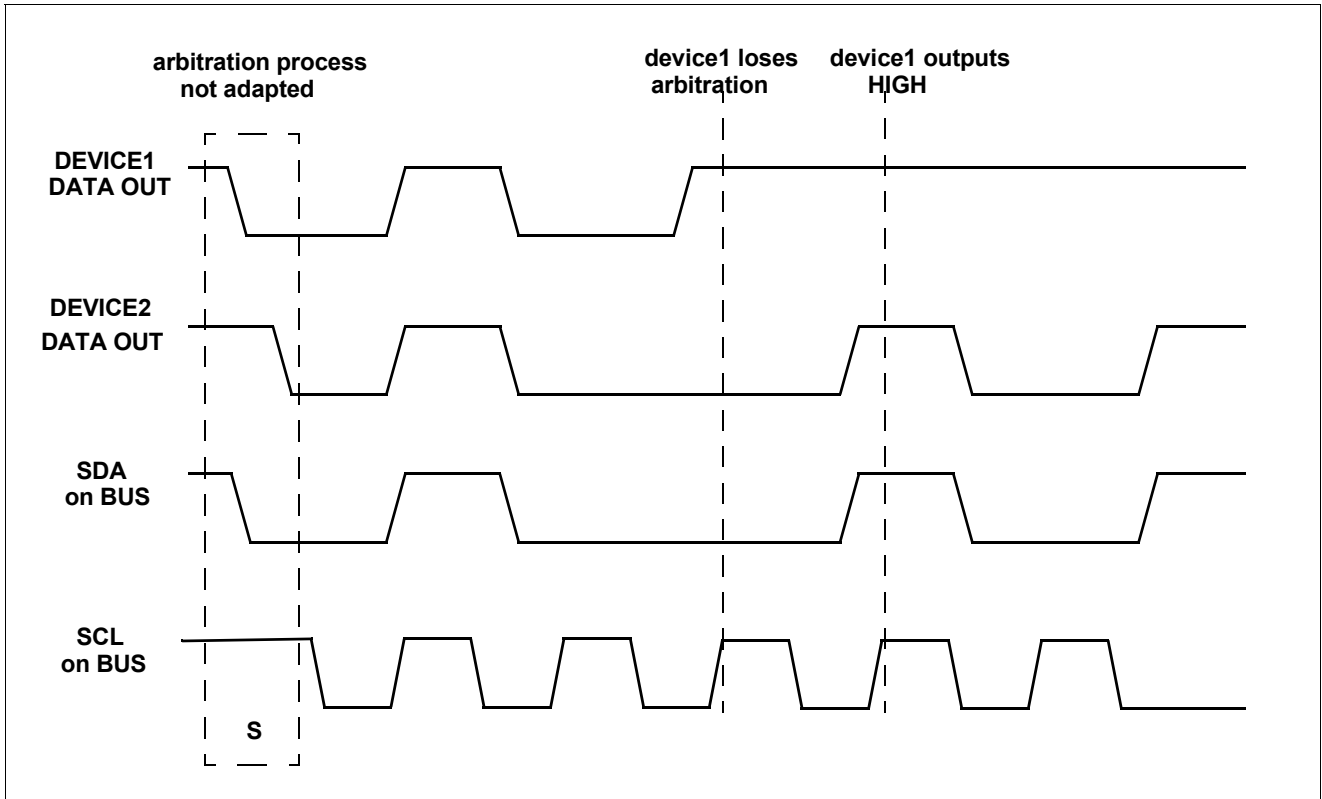


Figure 20-7 Arbitration procedure of two masters.

## 21. UNIVERSAL ASYNCHRONOUS SERIAL INTERFACE (UART)

The Asynchronous serial interface(UART) enables full-duplex operation wherein one byte of data after the start bit is transmitted and received. The on-chip baud rate generator dedicated to UART enables communications using a wide range of selectable baud rates.

The UART driver consists of TXSR, RXBR, ASIMR and BRGCR register. Clock asynchronous serial I/O mode (UART) can be selected by ASIMR register. Figure 21-1 shows a block diagram of the serial interface (UART).

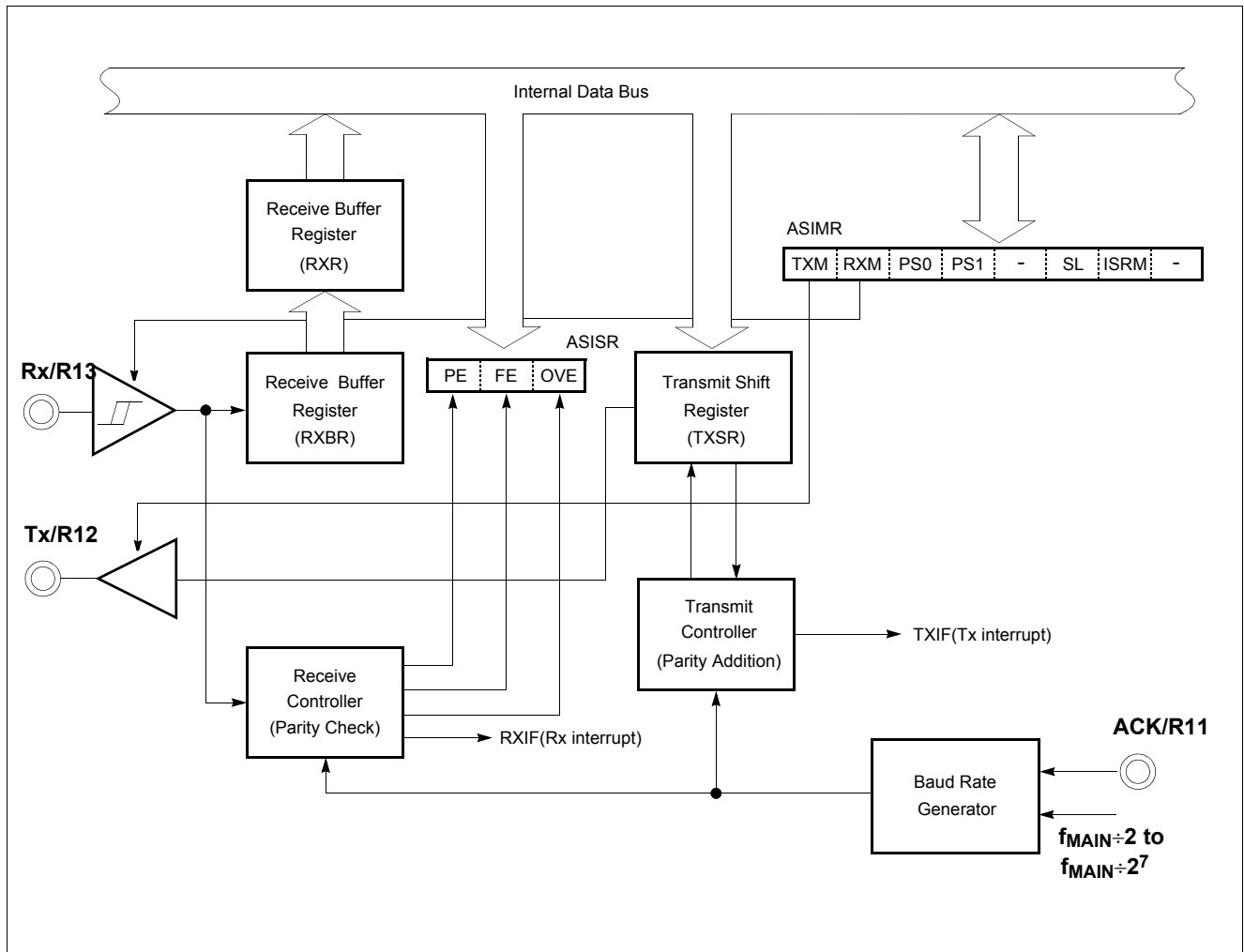


Figure 21-1 UART Block Diagram

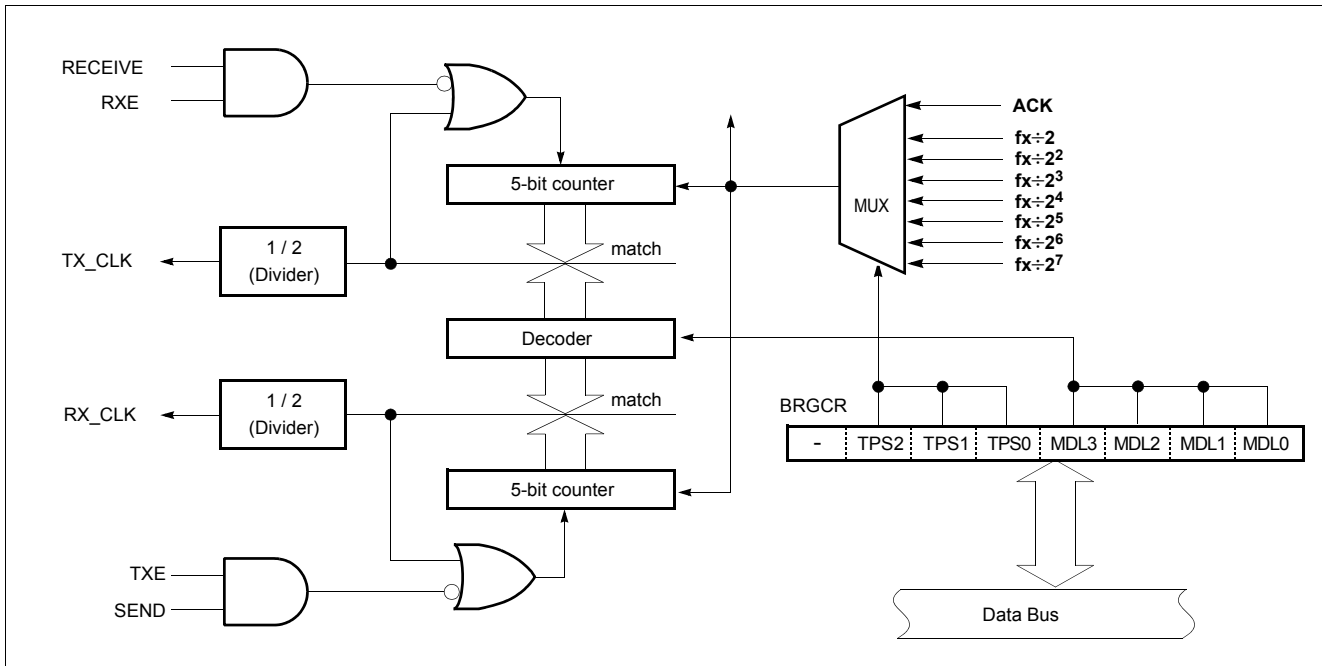


Figure 21-2 Baud Rate Generator Block Diagram

## 21.1 Asynchronous Serial Interface Configuration

The asynchronous serial interface (UART) consists of the following hardware.

Item	Configuration
Register	Transmit shift register (TXSR) Receive buffer register (RXBR)
Control register	Asynchronous serial interface mode register (ASIMR) Baudrate generator control register (BRGCR)

Table 21-1 Serial Interface Configuration

### Transmit Shift Register (TXSR)

This is the register for setting transmit data. Data written to TXSR is transmitted as serial data. When the data length is set as 7 bit, bit 0 to 6 of the data written to TXSR are transferred as transmit data. Writing data to TXSR starts the transmit operation. TXSR can be written by an 8 bit memory manipulation instruction. It cannot be read.

**Note:** Do not write to TXSR during a transmit operation. The same address is assigned to TXSR and the receive buffer register (RXBR). A read operation reads values from

RXBR.

### Receive Buffer Register (RXBR)

This register is used to hold received data. When one byte of data is received, one byte of new received data is transferred from the receive shift register. When the data length is set as 7 bits, received data is sent to bits 0 to 6 of RXBR. In this case, the MSB of RXBR always becomes 0. RXBR can be read by an 8 bit memory manipulation instruction. It cannot be written.

**Note:** The same address is assigned to RXBR and the transmit shift register (TXSR). During a write operation, values are written to TXSR.

### Asynchronous serial interface mode control register (ASIMR)

This is an 8 bit register that controls asynchronous serial interface (UART)'s serial transfer operation. ASIMR is set by a 1 bit or 8 bit memory manipulation instruction.

### Baud rate generator control register (BRGCR)

This register sets the serial clock for asynchronous serial interface. BRGCR is set by an 8 bit memory manipulation instruction.

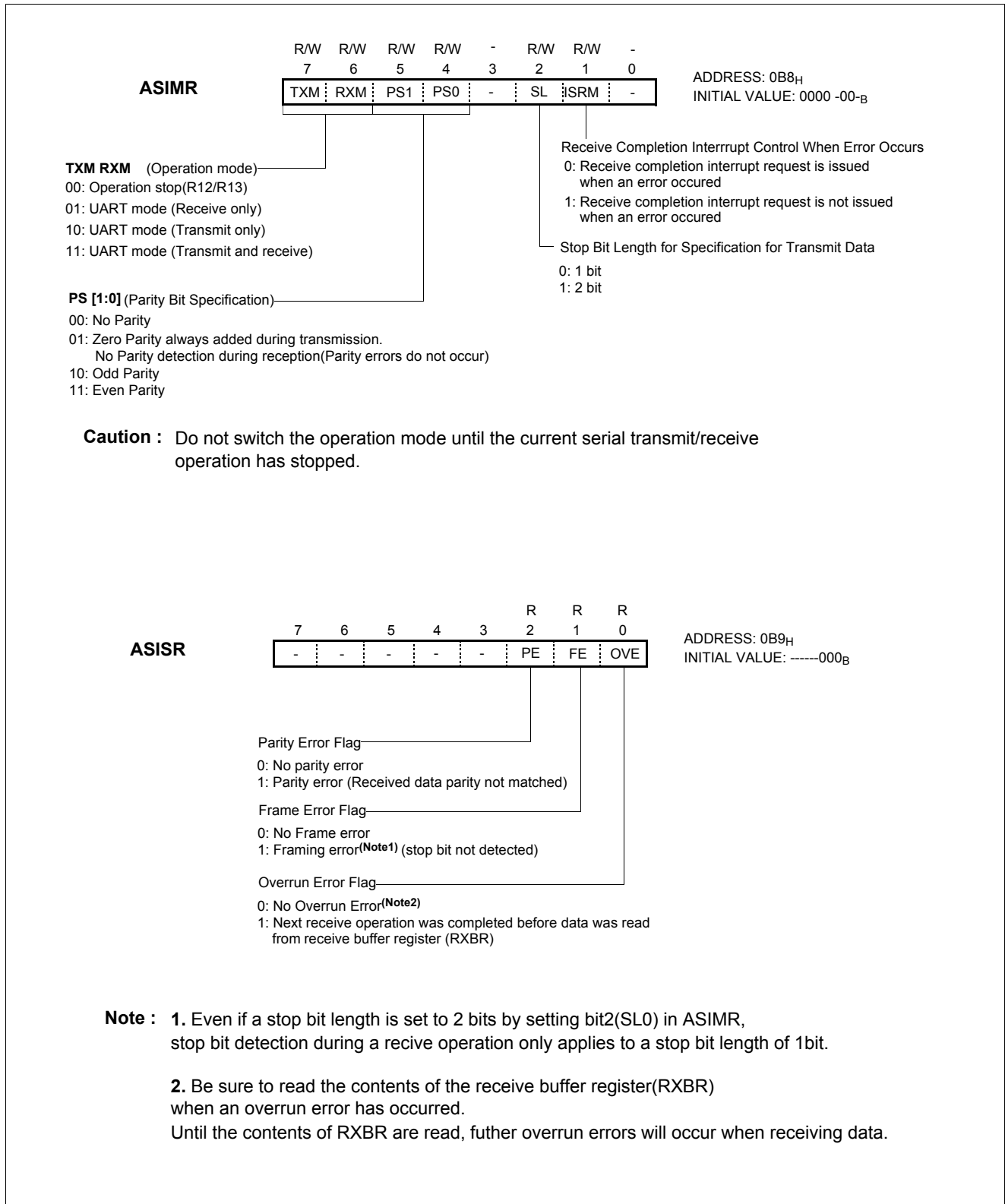


Figure 21-1 Asynchronous Serial Interface Mode & Status Register

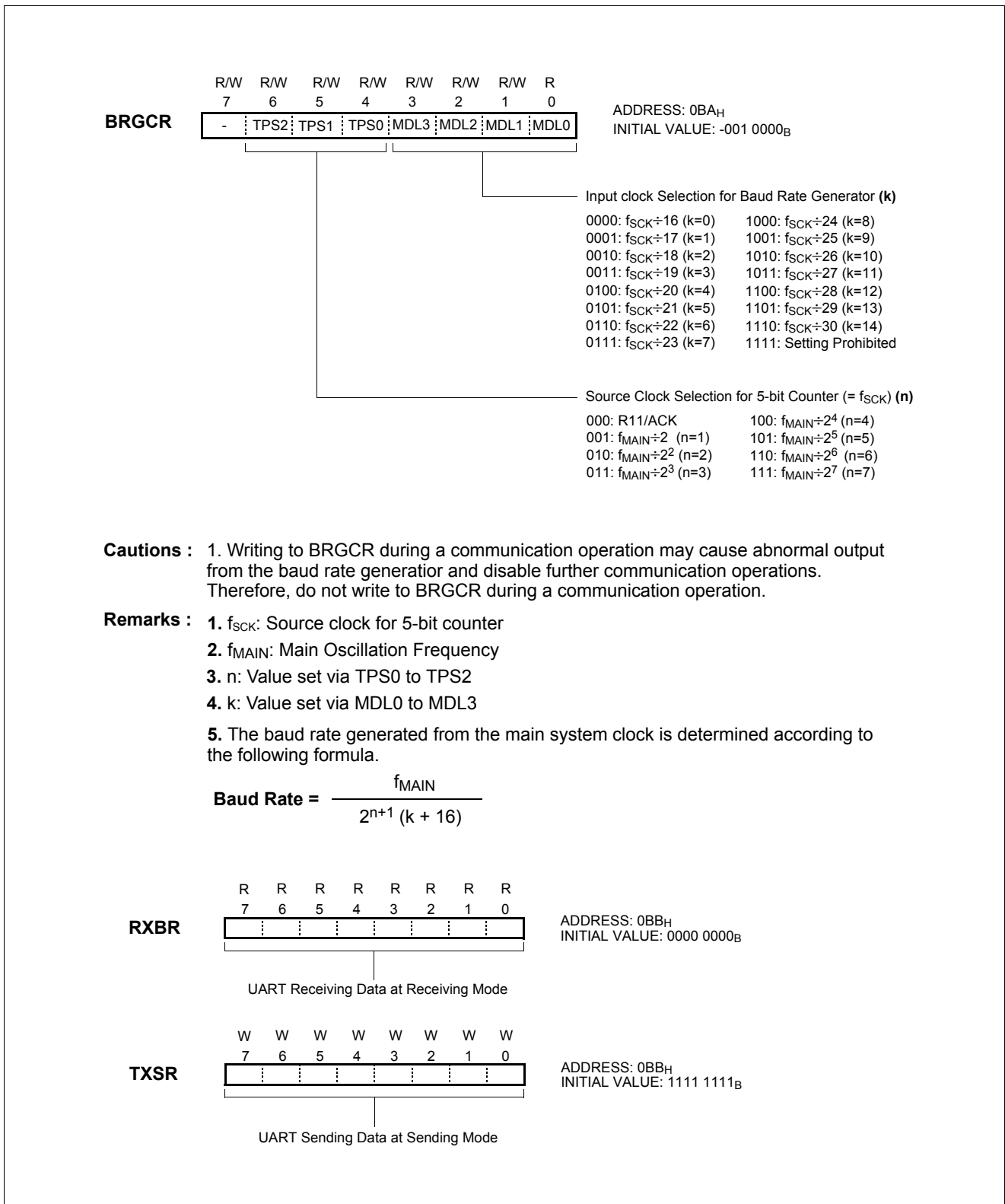


Figure 21-2 Baud Rate Generator Control Register, Receive Buffer Register, Transmit shift Register

## 21.2 Relationship between main clock and baud rate

The transmit/receive clock that is used to generate the baud rate is obtained by dividing the main system clock. Transmit/Receive clock generation for baud rate is made by using main system clock which is divided.

The baud rate generated from the main system clock is determined according to the following formula

$$BaudRate = \frac{fx}{2^{n+1}(K+16)}$$

- fx : main system clock oscillation frequency
- n : value set via TPS0 to TPS1 (1 ≤ n ≤ 7)
- k : value set via MDL0 to MDL3 (0 ≤ n ≤ 14)

Baud Rate (bps)	fx = 11.0592M		fx = 8.00M		fx = 7.3728M		fx = 6.00M		fx = 5.00M		fx = 4.1943	
	BRGCR	Err (%)	BRGCR	Err (%)	BRGCR	Err (%)	BRGCR	Err (%)	BRGCR	Err (%)	BRGCR	Err (%)
600	-	-	-	-	-	-	-	-	-	-	7BH	1.14
1,200	-	-	7AH	0.16	78H	0.00	73H	2.79	70H	1.73	6BH	1.14
2,400	72H	0.00	6AH	0.16	68H	0.00	63H	2.79	60H	1.73	5BH	1.14
4,800	62H	0.00	5AH	0.16	58H	0.00	53H	2.79	50H	1.73	4BH	1.14
9,600	52H	0.00	4AH	0.16	48H	0.00	43H	2.79	40H	1.73	3BH	1.14
19,200	42H	0.00	3AH	0.16	38H	0.00	33H	2.79	30H	1.73	2BH	1.14
31,250	36H	0.52	30H	0.00	2DH	1.70	28H	0.00	24H	0.00	21H	-1.30
38,400	32H	0.00	2AH	0.16	28H	0.00	23H	2.79	20H	1.73	1BH	1.14
76,800	22H	0.00	1AH	0.16	18H	0.00	13H	2.79	10H	1.73	-	-
115,200	18H	0.00	11H	2.12	10H	0.00	-	-	-	-	-	-

Table 21-2 Relationship Between Main Clock and Baud Rate



## 22. OPERATION MODE

The system clock controller starts or stops the main frequency clock oscillator, which is controlled by system clock mode register (SCMR). Figure 22-1 shows the operating mode transition diagram.

System clock control is performed by the system clock mode register (SCMR). During reset, this register is initialized to “0” so that the main-clock operating mode is selected.

### Main Active mode

This mode is fast-frequency operating mode. The CPU and

the peripheral hardware are operated on the high-frequency clock. At reset release, this mode is invoked.

### SLEEP mode

In this mode, the CPU clock stops while peripherals and the oscillation source continue to operate normally.

### STOP mode

In this mode, the system operations are all stopped, holding the internal states valid immediately before the stop at the low power consumption level.

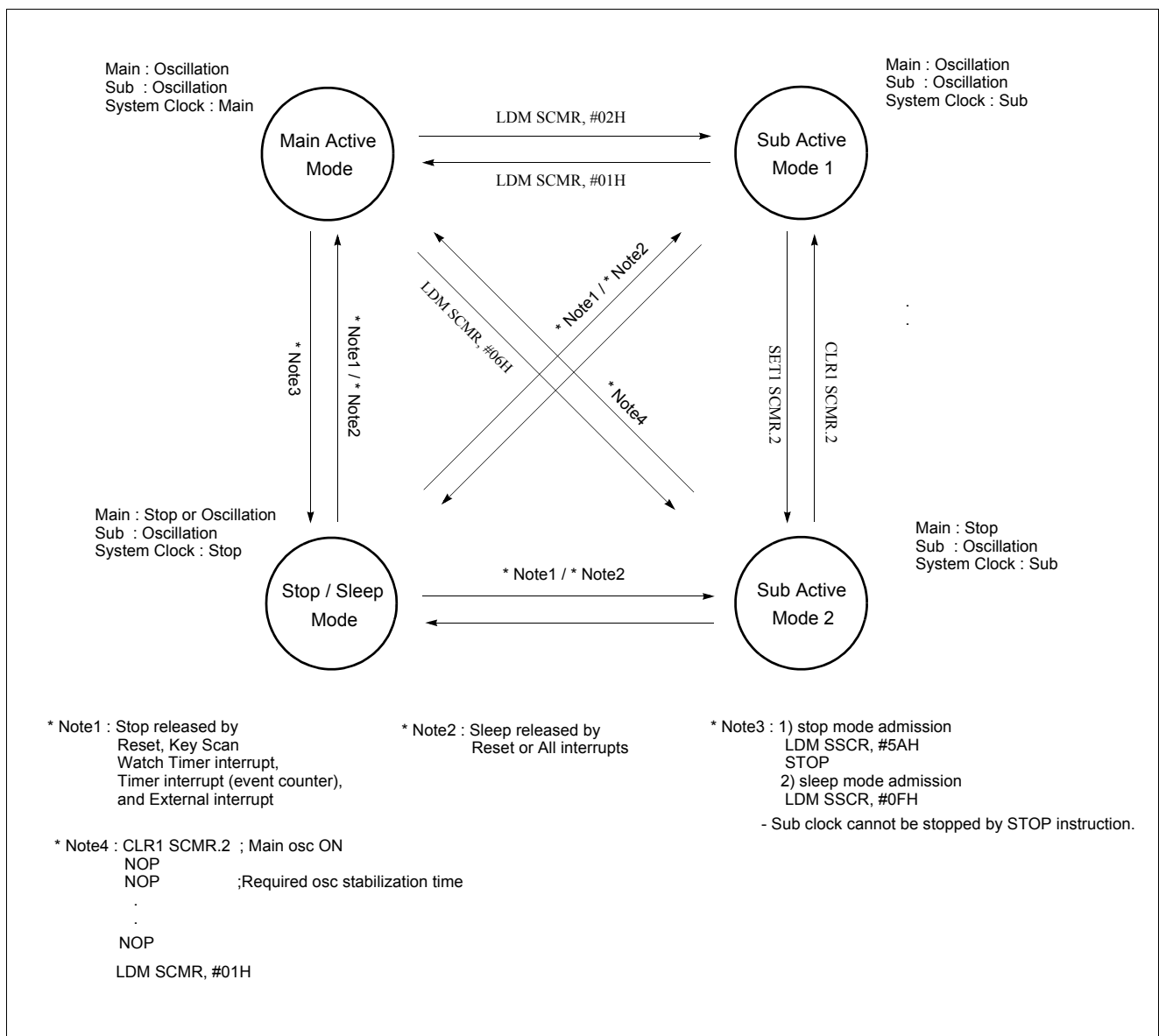


Figure 22-1 Operating Mode

## 22.1 Operation Mode Switching

### Shifting from the Normal operation to the SLEEP mode

By writing "0FH" into SSCR which will be explained in "23.1 SLEEP Mode" on page 117, the CPU clock stops and the SLEEP mode is invoked. The CPU stops while other peripherals are operate normally.

The way of release from this mode is RESET and all available interrupts.

For more detail, See "23.1 SLEEP Mode" on page 117

### Shifting from the Normal operation to the STOP mode

By writing "5AH" into SSCR and then executing STOP instruction, the main-frequency clock oscillation stops and the STOP mode is invoked. But sub-frequency clock oscillation is operated continuously.

After the STOP operation is released by reset, the operation mode is changed to Main active mode.

The methods of release are RESET, Key scan interrupt, Watch Timer interrupt, Timer/Event counter1 (EC0 pin) and External Interrupt.

For more details, see "23.2 STOP Mode" on page 118.

---

**Note:** *In the STOP and SLEEP operating modes, the power consumption by the oscillator and the internal hardware is reduced. However, the power for the pin interface (depending on external circuitry and program) is not directly associated with the low-power consumption operation. This must be considered in system design as well as interface circuit design.*

---

## 23. POWER DOWN OPERATION

MC81F8816/8616 have 2 power down mode. In power down mode, power consumption is reduced considerably in Battery operation that Battery life can be extended a lot.

Sleep mode is entered by writing “0F<sub>H</sub>” into Stop and Sleep Control Register(SSCR), and STOP mode is entered by writing “5A<sub>H</sub>” into SSCR and then executing STOP instruction.

### 23.1 SLEEP Mode

In this mode, the internal oscillation circuits remain active.

Oscillation continues and peripherals are operate normally but CPU stops. The status of all Peripherals in this mode is shown in Table 23-1. Sleep mode is entered by writing “0F<sub>H</sub>” into SSCR (address 0E9<sub>H</sub>).

It is released by RESET or all interrupt. To be released by interrupt, interrupt should be enabled before Sleep mode.

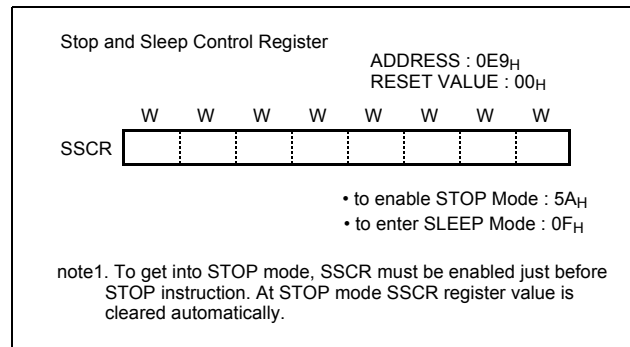


Figure 23-1 SLEEP Mode Register

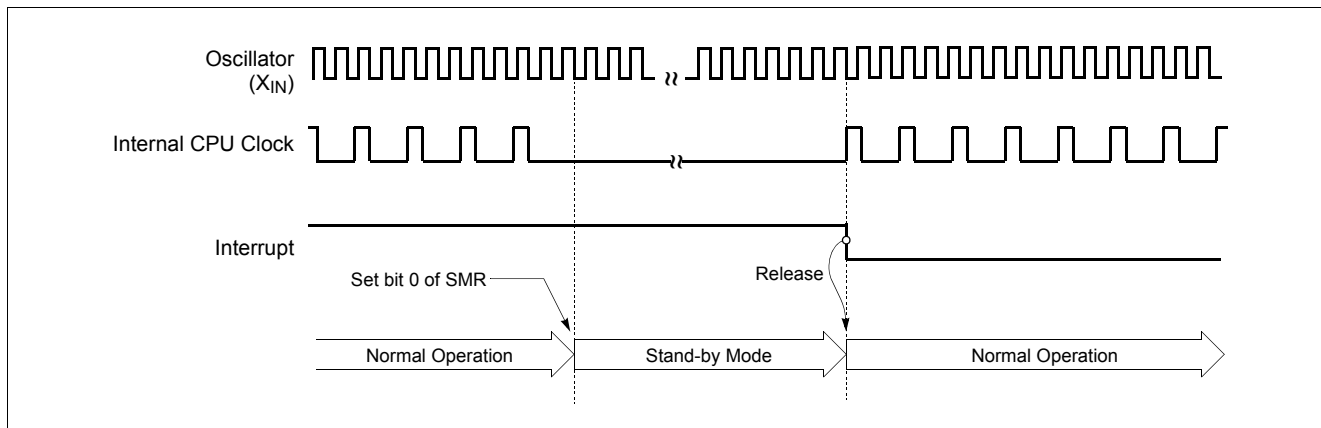


Figure 23-2 Sleep Mode Release Timing by External Interrupt

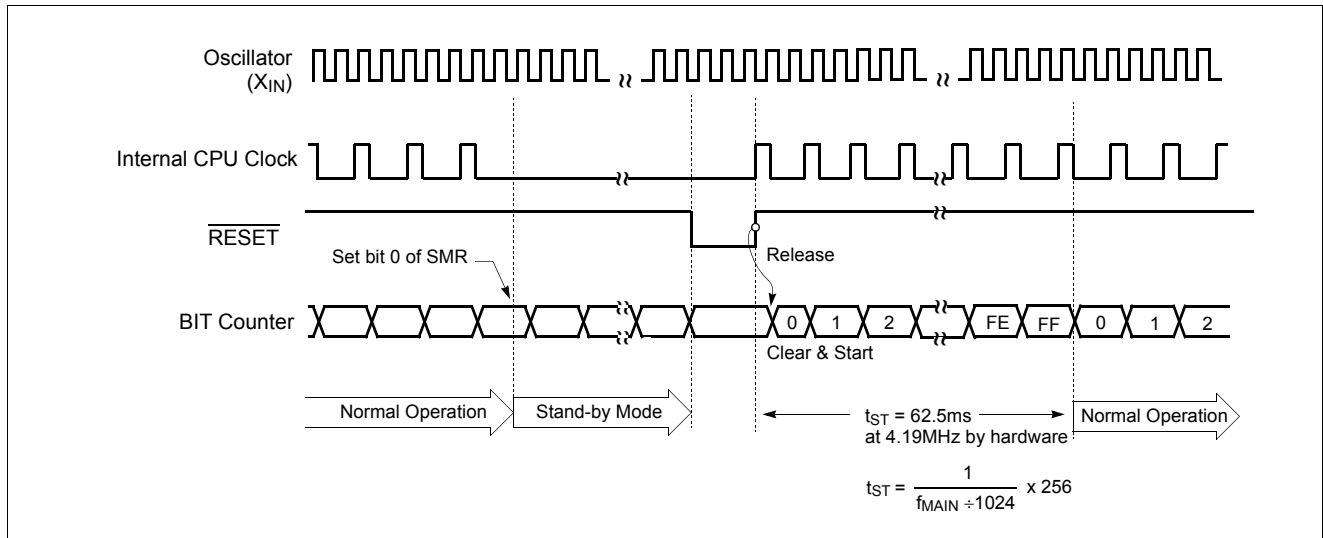


Figure 23-3 SLEEP Mode Release Timing by RESET pin

### 23.2 STOP Mode

**Note:** If the STOP mode is used in the program, BOD function should be disabled in the initial routine of software.

For applications where power consumption is a critical factor, this device provides STOP mode for reducing power consumption.

#### In case of starting the Stop Operation

The STOP mode can be entered by STOP instruction during program execution. In Stop mode, the on-chip main-frequency oscillator, system clock, and peripheral clock are stopped (Watch timer clock is oscillating continuously). With the clock frozen, all functions are stopped, but the on-chip RAM and Control registers are held. The port pins output the values held by their respective port data register and the port direction registers. The status of peripherals during Stop mode is shown below.

Peripheral	STOP Mode	Sleep Mode
CPU	All CPU operations are disabled	All CPU operations are disabled
RAM	Retain	Retain
LCD driver	Operates continuously	Operates continuously
Basic Interval Timer	Halted	Operates continuously
Timer/Event counter 0,1	Halted (Only when the Event counter mode is enabled, Timer 0, 1 operates normally)	Timer/Event counter 0,1 operates continuously
Watch Timer	Operates continuously	Operates continuously
Main-oscillation	Stop (X <sub>IN</sub> =L, X <sub>OUT</sub> =H)	Oscillation <sup>1</sup>
Sub-oscillation	Oscillation	Oscillation
I/O ports	Retain	Retain
Control Registers	Retain	Retain

Table 23-1 Peripheral Operation during Power Down Mode

Peripheral	STOP Mode	Sleep Mode
Release method	by RESET, Watch Timer interrupt, Timer interrupt (ECO), UART interrupt and External interrupt	by RESET, All interrupts

**Table 23-1 Peripheral Operation during Power Down Mode**

1. Refer to the Table 10-2

Operating Clock source	Main Operating Mode	Main Sleep Mode	Sub Active Operating Mode	Sub Sleep Operating Mode	Stop Mode 1	Stop Mode 2
Main Clock	Oscillation	Oscillation	SCMR[2] 0 ==>Oscillation 1 ==>Stop	SCMR[2] 0 ==>Oscillation 1 ==>Stop	Stop	Stop
Sub Clock	Oscillation	Oscillation	Oscillation	Oscillation	Oscillation	Stop
System Clock	Active	Stop	Active	Stop	Stop	Stop
Peri. Clock	Active	Active	Active	Active	Stop <sup>1</sup>	Stop

1. Except watch timer(sub clock) and LCD driver(sub clock)

**Table 23-2 Clock Operation of STOP and SLEEP mode**

**Note:** Since the  $X_{IN}$  pin is connected internally to GND to avoid current leakage due to the crystal oscillator in STOP mode, do not use STOP instruction when an external clock is used as the main system clock.

In the Stop mode of operation,  $V_{DD}$  can be reduced to minimize power consumption. Be careful, however, that  $V_{DD}$  is not reduced before the Stop mode is invoked, and that  $V_{DD}$  is restored to its normal operating level before the Stop mode is terminated.

The reset should not be activated before  $V_{DD}$  is restored to its normal operating level, and must be held active long enough to allow the oscillator to restart and stabilize. And after STOP instruction, at least two or more NOP instruction should be written as shown in example below.

Example)

```

Reset: :
      LDM   BODR, #1100_0000B
      :
Main:  :
      LDM   CKCLR, #0000_1111B
      STOP
    
```

```

NOP
NOP
:
    
```

The Interval Timer Register CKCLR should be initialized by software in order that oscillation stabilization time should be longer than 20ms before STOP mode.

**In case of releasing the STOP mode**

The exit from STOP mode is using hardware reset or external interrupt, watch timer or timer interrupt (ECO).

To release STOP mode, corresponding interrupt should be enabled before STOP mode. Specially as a clock source of Timer/Event counter, ECO pin can release it by Timer/Event counter Interrupt request.

Reset redefines all the control registers but does not change the on-chip RAM. External interrupts allow both on-chip RAM and Control registers to retain their values.

Start-up is performed to acquire the time for stabilizing oscillation. During the start-up, the internal operations are all stopped.

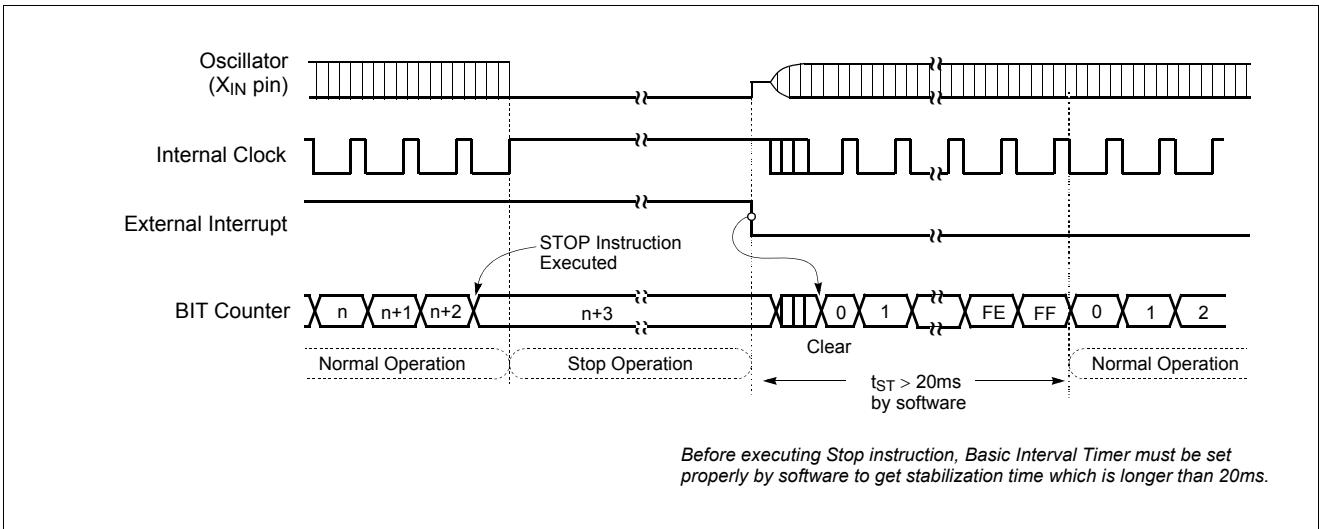


Figure 23-4 STOP Mode Release Timing by External Interrupt

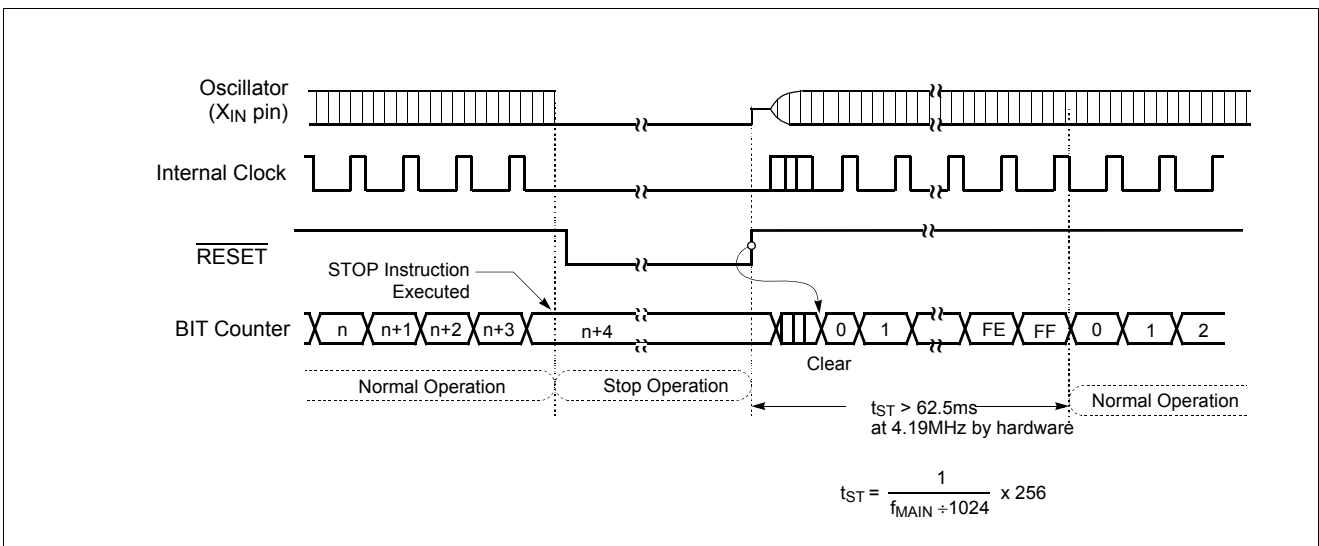


Figure 23-5 STOP Mode Release Timing by RESET

**Minimizing Current Consumption**

The Stop mode is designed to reduce power consumption. To minimize current drawn during Stop mode, the user should turn-off output drivers that are sourcing or sinking current, if it is practical.

**Note:** In the STOP operation, the power dissipation associated with the oscillator and the internal hardware is lowered; however, the power dissipation associated with the pin interface (depending on the external circuitry and program) is not directly determined by the hardware operation of the STOP feature. This point should be little current flows when the input level is stable at the power voltage level ( $V_{DD}/V_{SS}$ ); however, when the input level becomes higher than the power voltage level (by approximately 0.3V), a current begins to flow. Therefore, if cutting off the output transistor at an I/O port puts the pin signal into the high-impedance state, a current flow across the ports input transistor, requiring it to fix the level by pull-up or other means.

It should be set properly that current flow through port doesn't exist.

First consider the setting to input mode. Be sure that there is no current flow after considering its relationship with external circuit. In input mode, the pin impedance viewing from external MCU is very high that the current doesn't flow.

But input voltage level should be  $V_{SS}$  or  $V_{DD}$ . Be careful that if unspecified voltage, i.e. if uniformed voltage level (not  $V_{SS}$  or  $V_{DD}$ ) is applied to input pin, there can be little current (max. 1mA at around 2V) flow.

If it is not appropriate to set as an input mode, then set to output mode considering there is no current flow. Setting to High or Low is decided considering its relationship with external circuit. For example, if there is external pull-up resistor then it is set to output mode, i.e. to High, and if there is external pull-down resistor, it is set to low.

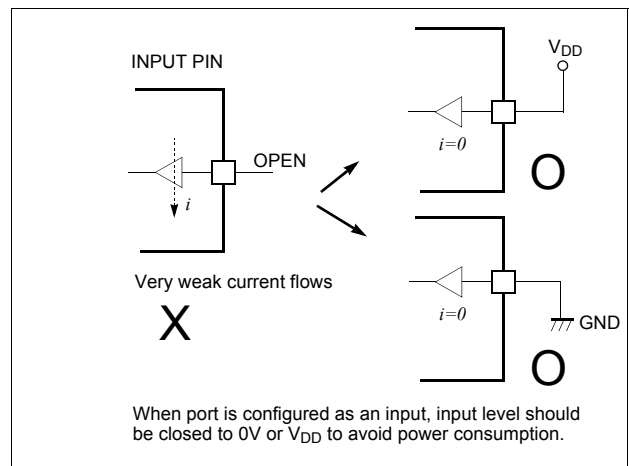
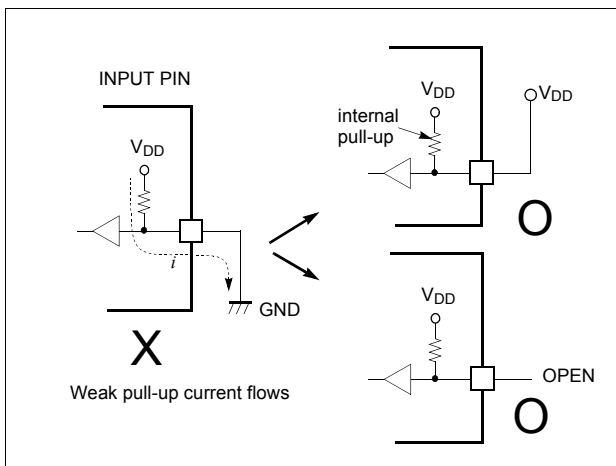


Figure 23-6 Application Example of Unused Input Port

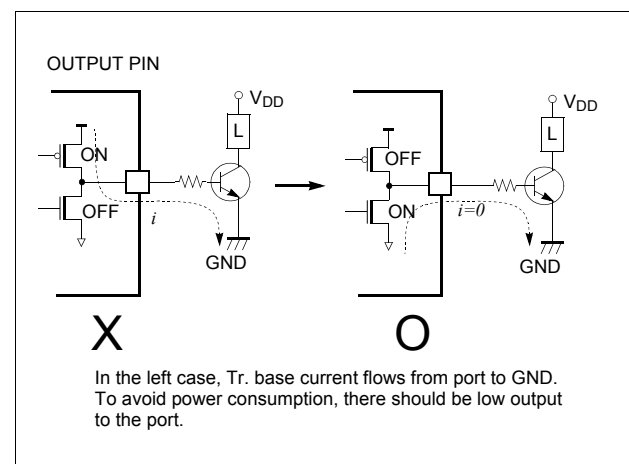
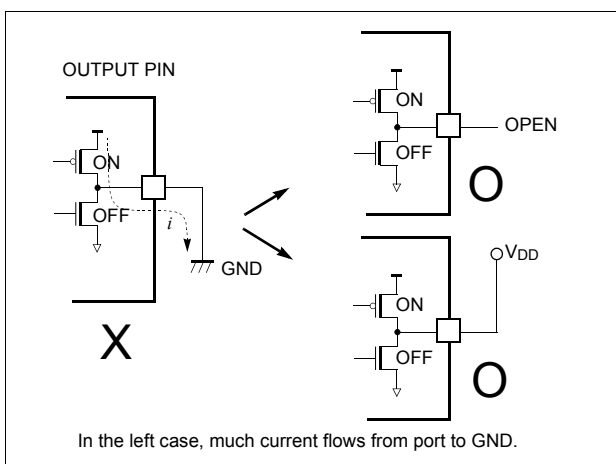


Figure 23-7 Application Example of Unused Output Port

## 24. OSCILLATOR CIRCUIT

The MC81F8816/8616 have three oscillation circuits internally.  $X_{IN}$  and  $X_{OUT}$  are input and output for main frequency and  $SX_{IN}$  and  $SX_{OUT}$  are input and output for sub

frequency, respectively, inverting amplifier which can be configured for being used as an on-chip oscillator, as shown in Figure 24-1.

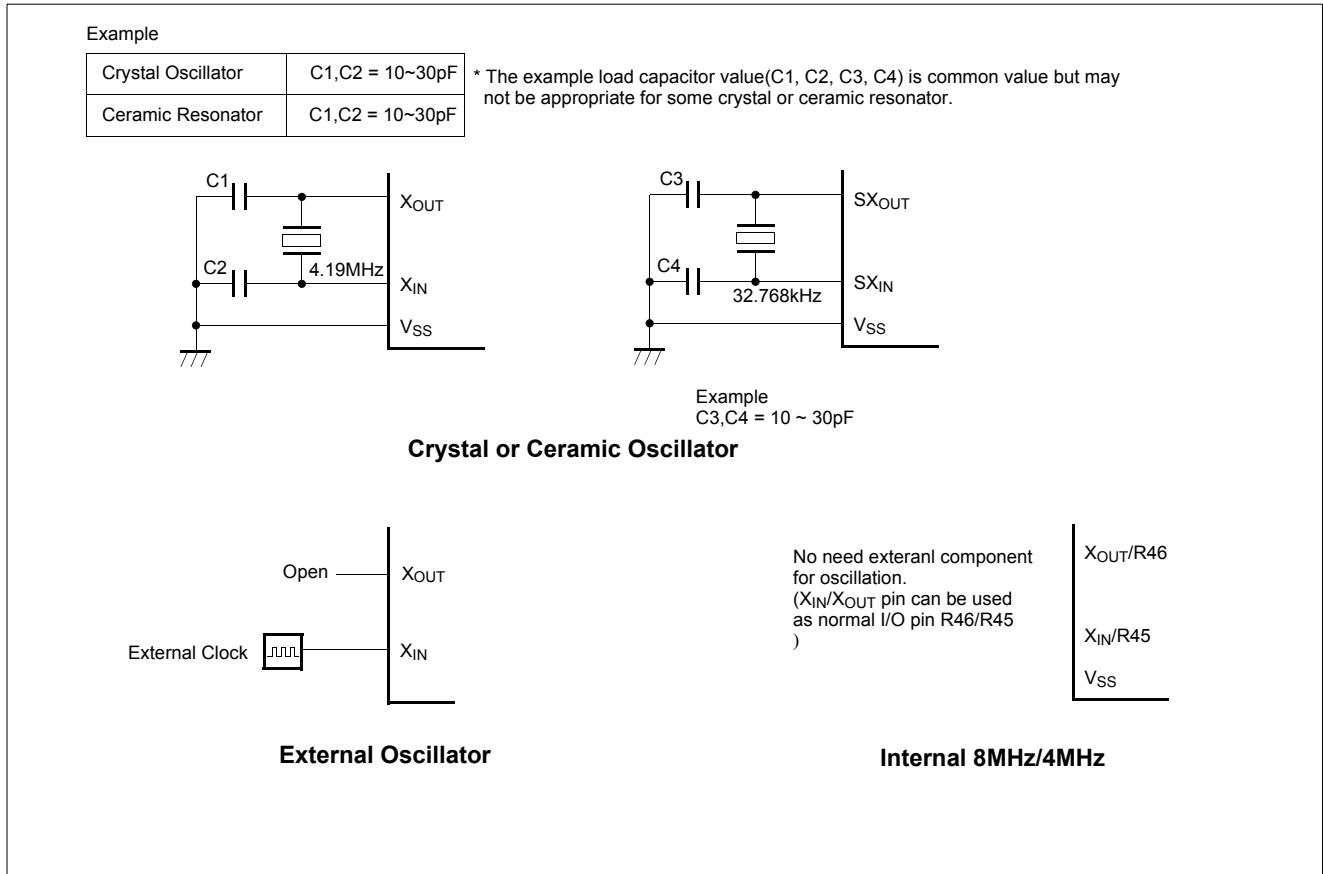


Figure 24-1 Oscillation Circuit

Oscillation circuit is designed to be used either with a ceramic resonator or crystal oscillator. Since each crystal and ceramic resonator have their own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.

In addition, see Figure 24-2 for the layout of the crystal.

**Note:** Minimize the wiring length. Do not allow the wiring to intersect with other signal conductors. Do not allow the wiring to come near changing high current. Set the potential of the grounding position of the oscillator capacitor to that of  $V_{SS}$ . Do not ground it to any ground pattern where high current is present. Do not fetch signals from the oscillator.

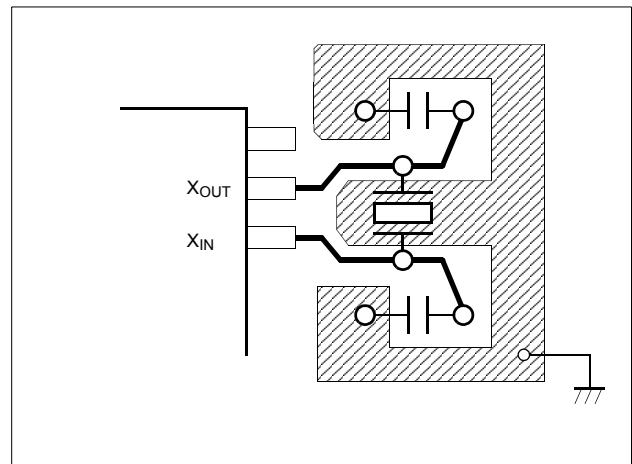


Figure 24-2 Layout of Oscillator PCB circuit



## 25. PLL

The phase locked loop (PLL) is used to a fixed frequency using a phase difference comparison system.

Figure 25-1 shows the PLL block diagram.

As shown in Figure 25-1, the PLL consists of an input se-

lection circuit, Feedback divider, phase comparator (Phase-Locked Loop), VCO and Post-scaler.

PLL consists of 8-bit XPLLCR register and XPLLDAT register shown in Figure 25-1 .

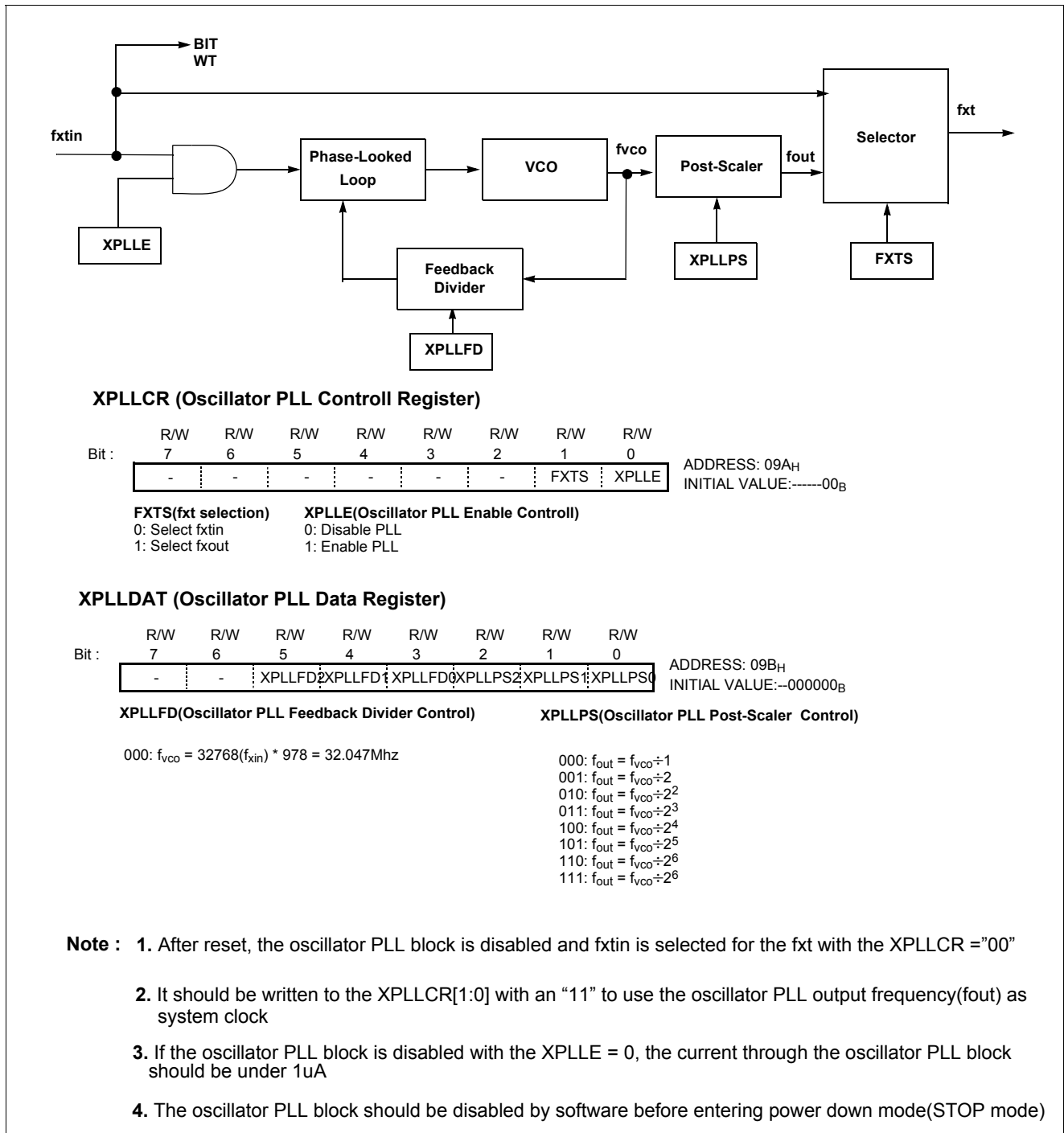
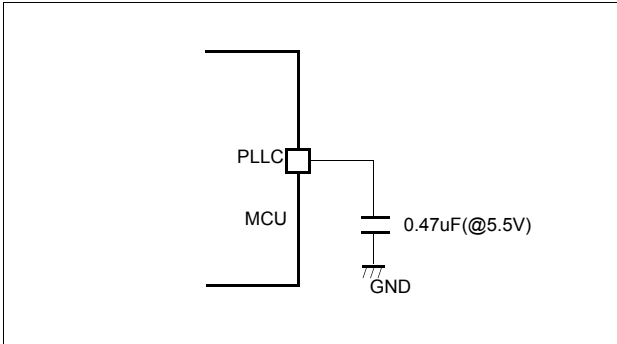


Figure 25-1 OSCILLATER PLL CIRCUIT Diagram

### 25.1 External PLL Circuit

A External connection for normal PLL is shown in Figure 25-2.



**Figure 25-2 External Circuit**

## 26. RESET

The MC81F8816/8616 have has four reset generation sources; external reset input, power on reset (POR), brown-out detector reset (BOD) and watch-dog timer re-

set. Table 26-1 shows on-chip hardware initialization by reset action.

On-chip Hardware	Initial Value
Program counter (PC)	(FFFF <sub>H</sub> ) - (FFFE <sub>H</sub> )
RAM page register (RPR)	0
G-flag (G)	0

On-chip Hardware	Initial Value
Operation mode	Main-frequency clock
Peripheral clock	On
Control registers	Refer to Table 8-1 on page 39

Table 26-1 Initializing Internal Status by Reset Action

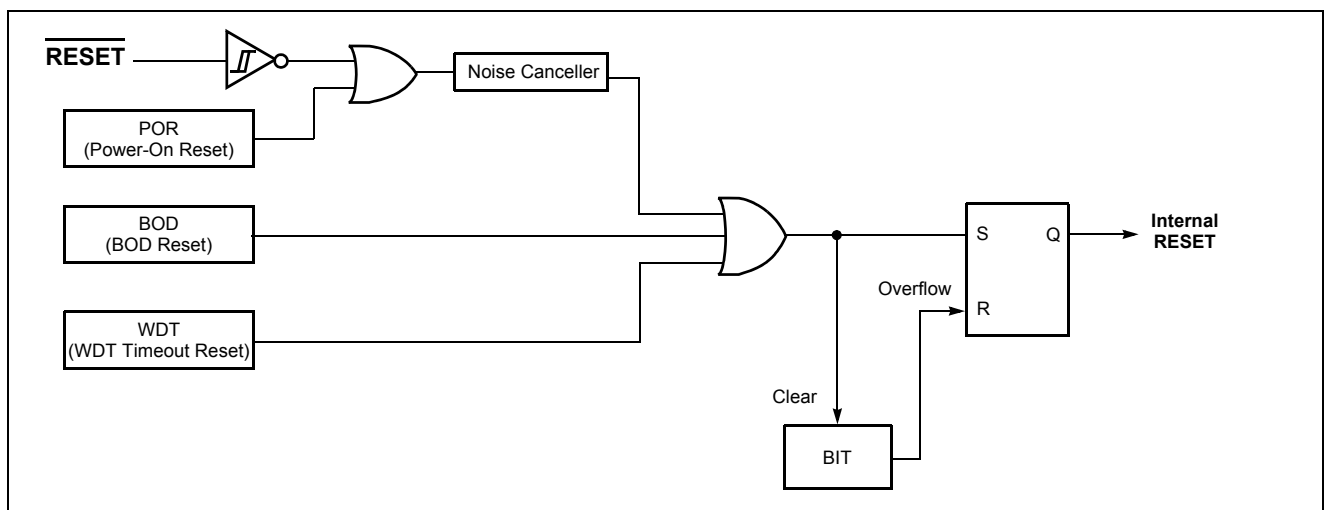


Figure 26-1 RESET Block Diagram

### 26.1 External Reset Input

The reset input is the  $\overline{\text{RESET}}$  pin, which is the input to a Schmitt Trigger. A reset accomplished by holding the  $\overline{\text{RESET}}$  pin to low for at least 8 oscillator periods, within the operating voltage range and oscillation stable, it is applied, and the internal state is initialized. After reset, 65.5ms (at 4MHz) and 7 oscillator periods are required to start execution as shown in Figure 26-3.

Internal User RAM is not affected by reset. When  $V_{DD}$  is turned on, the RAM content is indeterminate. Therefore, this RAM should be initialized before read or tested it.

When the  $\overline{\text{RESET}}$  pin input goes to high, the reset operation is released and the program execution starts at the vector address stored at FFFE<sub>H</sub> - FFFF<sub>H</sub>.

A connection for normal power-on-reset is shown in Figure 26-2.

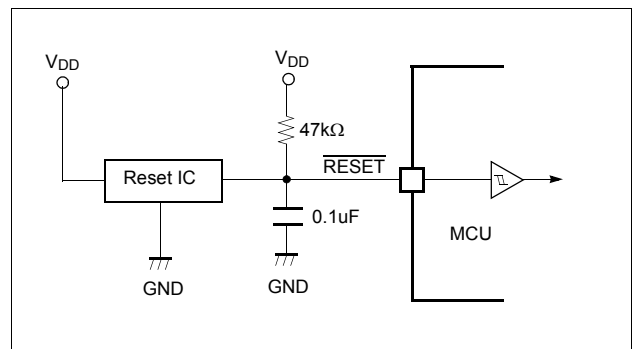


Figure 26-2 Normal Power-on-Reset Circuit

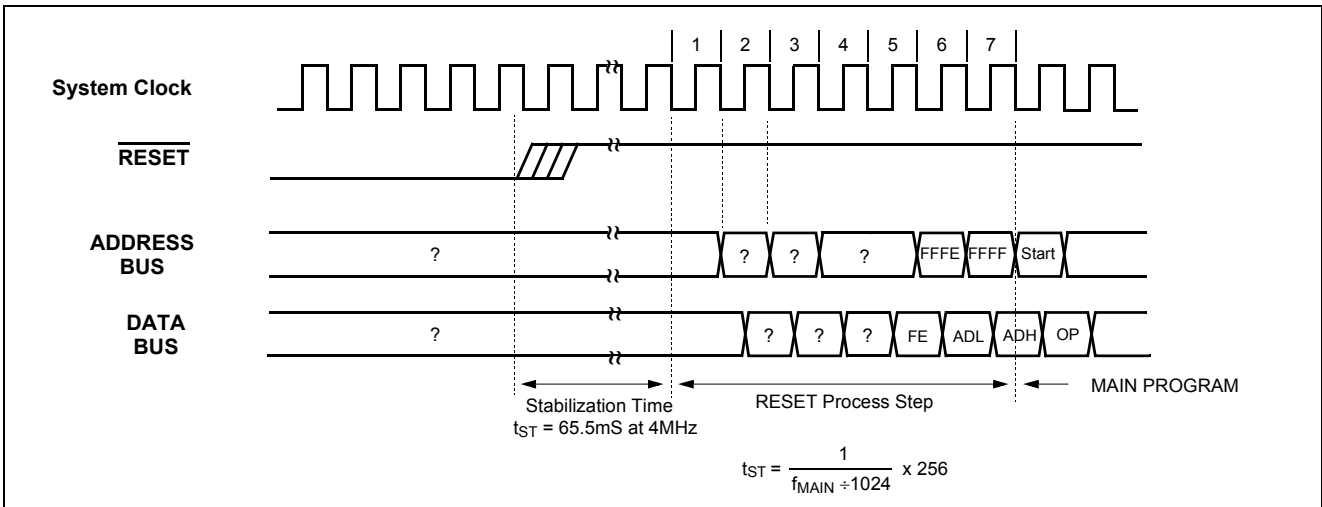


Figure 26-3 Timing Diagram of RESET

### 26.2 Power On Reset

The on-chip POR circuit holds down the device in RESET until V<sub>DD</sub> has reached a high enough level for proper operation. It will eliminate external components such as reset IC or external resistor and capacitor for external reset circuit. In addition that the RESET pin can be used to normal input port R47 by setting “POR” and “R47EN” bit of the Configuration option area(20FFH) in Flash programming. When the device starts normal operation, its operating

parameters (voltage, frequency, temperature...etc) must be met.

---

**Note:** When “POR” option is checked and “R47EN” option is not checked, RESET/R47 pin acts as external Reset input pin. In this case, the external reset circuit should be connected to RESET pin. If external reset is not needed, not only “POR”, but also “R47EN” option should be checked.

---

### 26.3 Brown-out Detector

Refer to “27. Brown-out Detector (BOD)”

### 26.4 Watchdog Timer Reset

Refer to “14. WATCH DOG TIMER”

## 27. Brown-out Detector (BOD)

**Note:** If the STOP mode is used in the program, BOD function should be disabled in the initial routine of software.

The MC81F8816/8616 has an on-chip BOD(Brown-out Detector) circuitry to immunize against power noise. The BOD control register BODR can enable or disable the built in reset circuitry. The Block diagram of BOD is shown in the Figure 27-1.

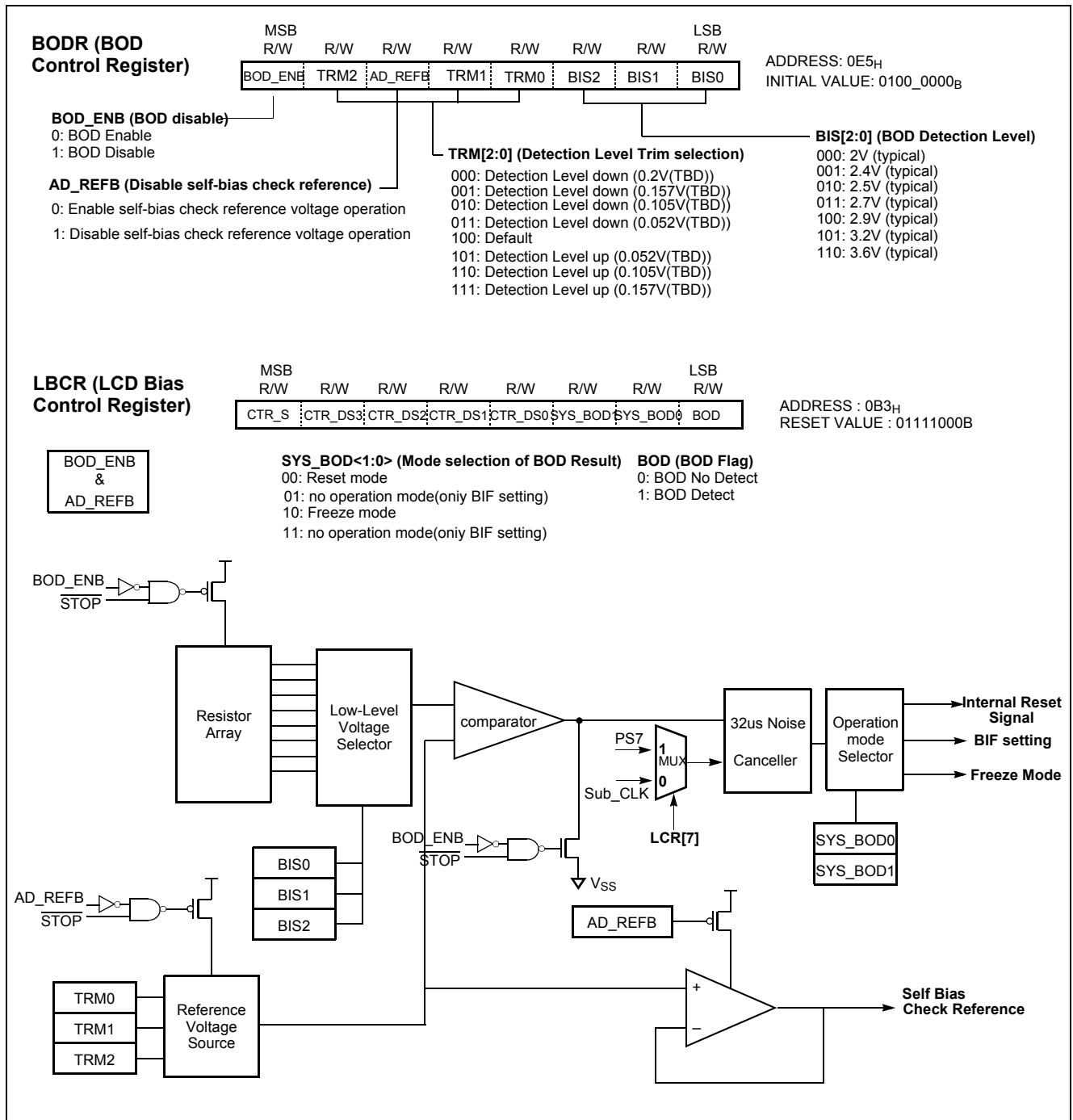


Figure 27-1 Block Diagram of BOD (Brown-out Detector Reset)

The BOD of MC81F8816/8616 has 8 detection level which can be selected by BIS[2:0] and each level can be trimmed by TRM[1:0].

The NC\_SEL bit of BODR is used for selecting BOD noise canceller. For example, if the NC\_SEL bit of BODR is set to “1” and VDD voltage falls below the BOD detection level during 20us, BOD does not generate internal reset signal or freeze mode signal because the 32us noise canceller eliminates low level detection signal less than 32us.

BOD result can be selected by SYS\_BOD[1:0] of LBCR register. When SYS\_BOD[1:0] is set to “00”, BOD generates reset signal. If SYS\_BOD[1:0] is set to “10”, it generates freeze mode signal and CPU freeze until the VDD voltage returns to regular level.

The self bias check reference, which can be used for calculating VDD voltage, can be activated by setting the AD\_REFB bit to “0” and BOD\_ENB bit to “0”. It is used for checking VDD voltage.

BIF is set to “1” when BOD occurs. It can be used to distinguish reset caused by BOD and other.

When the POR is used, the BOD detection level should be set to the level less than POR level. If the POR level is

2.4V, BOD level 2V and 2.4V can not operate.

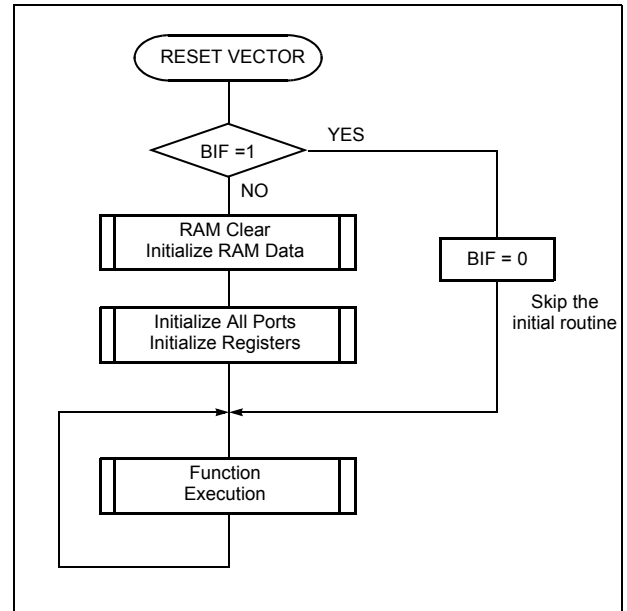


Figure 27-2 Example Flow of Reset flow by BOD

## 28. Oscillation Noise Protector

The Oscillation Noise Protector (ONP) is used to supply stable internal system clock by excluding the noise which could be entered into oscillator and recovery the oscillation fail. This function could be enabled or disabled by the “ONP” bit of the Device configuration area (20FFH) for the MC81F8816/8616.

The ONP function is like below.

- Recovery the oscillation wave crushed or loss caused

by high frequency noise.

- Change system clock to the internal oscillation clock when the high frequency noise is continuing.
- Change system clock to the internal oscillation clock when the X<sub>IN</sub>/X<sub>OUT</sub> is shorted or opened, the main oscillation is stopped except by stop instruction and the low frequency noise is entered.

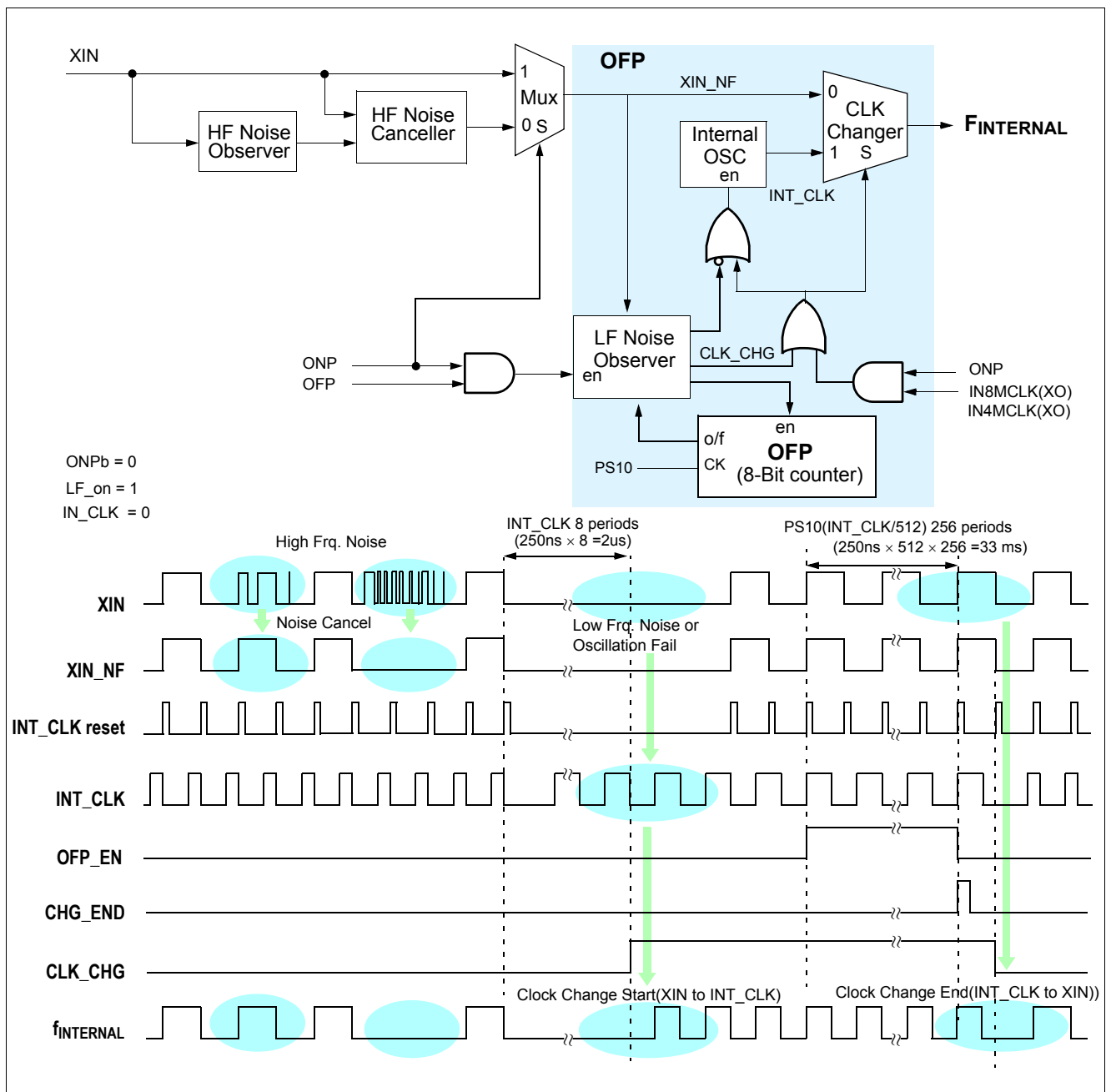


Figure 28-1 Block Diagram of ONP & OFP and Respective Wave Forms

The oscillation fail processor (OFP) can change the clock source from external to internal oscillator when the oscillation fail occurred. This function could be enabled or disabled by the “OFP” bit of the Device Configuration Area (MASK option for MC81F8816/8616).

And this function can recover the external clock source when the external clock is recovered to normal state.

The “IN8MCLK”, “IN4MCLK”, “IN8MCLKXO”,

“IN4MCLKXO”, option of the Device Configuration Area enables the function to operate the device by using the internal oscillator clock in ONP block as system clock. There is no need to connect the x-tal, resonator, RC and R externally. After selecting the this option, the period of internal oscillator clock could be checked by  $X_{OUT}$  outputting clock divided the internal oscillator clock by 4.



## 29. FLASH PROGRAMMING SPEC.

### 29.1 FLASH Configuration Byte

Except the user program memory, there is configuration byte(address 20FF<sub>H</sub>) for the selection of program lock, ONP, OPF, oscillation configuration and reset configuration. The configuration byte of FLASH is shown as Figure 29-1. It could be served when user use the FLASH programmer.

**Note:** The Configuration Option may not be read exactly when VDD rising time is very slow. It is recommended to adjust the VDD rising time faster than 40ms/V (200ms from 0V to 5V).

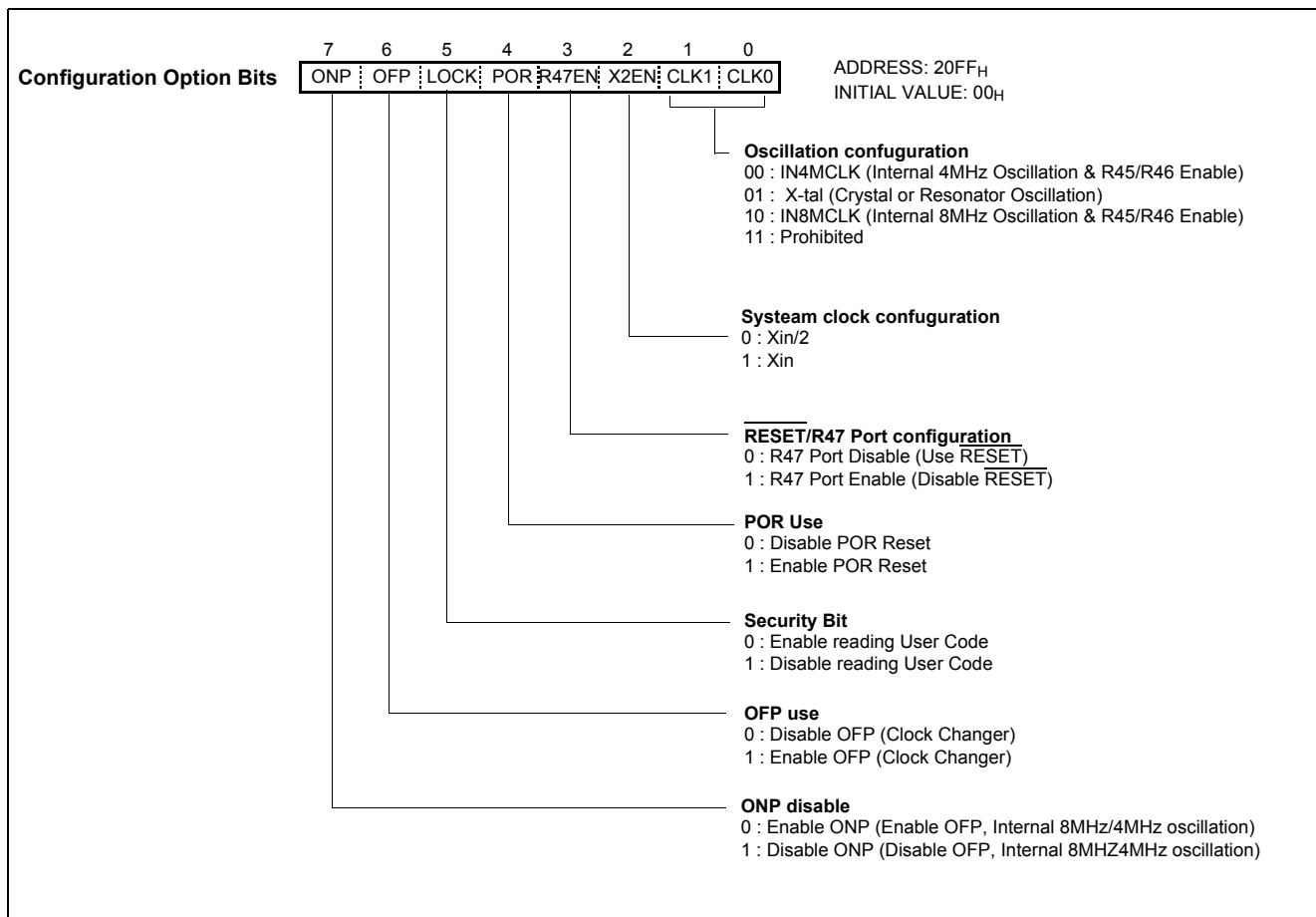


Figure 29-1 The FLASH Configuration Byte

### 29.2 FLASH Programming

The MC81F8816/8616 is a MTP microcontroller. Its internal user memory is constructed with FLASH ROM.

Blank FLASH's internal memory is filled by 00<sub>H</sub>, not FF<sub>H</sub>.

**Note:** In any case, you have to use the \*.OTP file for programming, not the \*.HEX file. After assemble, both OTP and HEX file are generated by automatically. The HEX file is used during program emulation on the emulator.

### How to Program

To program the FLASH or MTP devices, user can use ABOV own programmer.

### ABOV own programmer list

Manufacturer: ABOV Semiconductor Programmer:

**Choice-Sigma  
StandAlone-Gang4  
PGM-plus**

The Choice-Sigma is a ABOV Universal Single Programmer for

all of ABOV FLASH/OTP devices, also the StandAlone-Gang4 can program four FLASH/OTPs at once for ABOV device.

Ask to ABOV sales part for purchasing or more detail.

### Programming Procedure

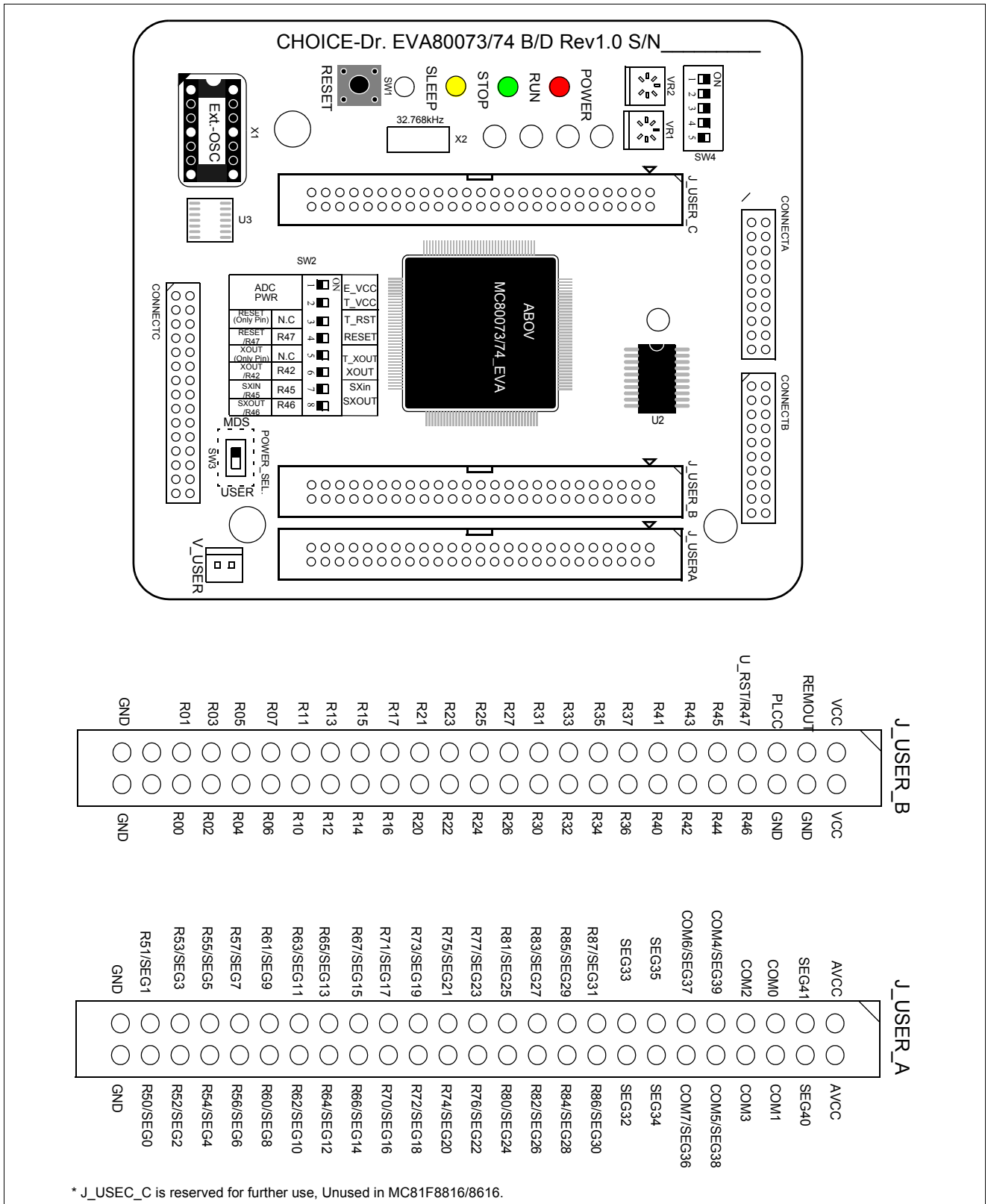
1. Select device MC81F8816/8616.
2. Load the \*.OTP file from the PC. The file is composed of Motorola-S1 format.

3. Set the programming address range as below table.

Address	Set Value
Buffer start address	E000 <sub>H</sub>
Buffer end address	FFFF <sub>H</sub>
Device start address	E000 <sub>H</sub>

4. Mount the socket adapter on the programmer.
5. Start program/verify.

### 30. EMULATOR EVA. BOARD SETTING



## 31. IN-SYSTEM PROGRAMMING

### 31.1 Getting Started / Installation

The In-System Programming (ISP) is performed without removing the microcontroller from the system. The In-System Programming (ISP) facility consists of a series of internal hardware resources coupled with internal firmware through the serial port. The In-System Programming (ISP) facility has made in-circuit programming in an embedded application possible with a minimum of additional expense in components and circuit board area. The following section details the procedure for accomplishing the installation procedure.

#### 1. Power off a target system.

#### 2. Configure a target system as ISP mode.

Refer to "31.3 Hardware Conditions to Enter the ISP Mode"

#### 3. Attach a USB-SIO-ISP B/D into a target system.

#### 4. Run the ABOV USB-SIO-ISP software.

- Down load the ISP S/W from <http://www.abov.co.kr>.
- Unzip the download file and run USB-SIO-ISP.exe

#### 5. Select a device in the USB-SIO-ISP S/W.

#### 6. Power on a target system.

#### 7. Execute ISP command such as read, program, auto... by pressing buttons on the USB-SIO-ISP S/W.

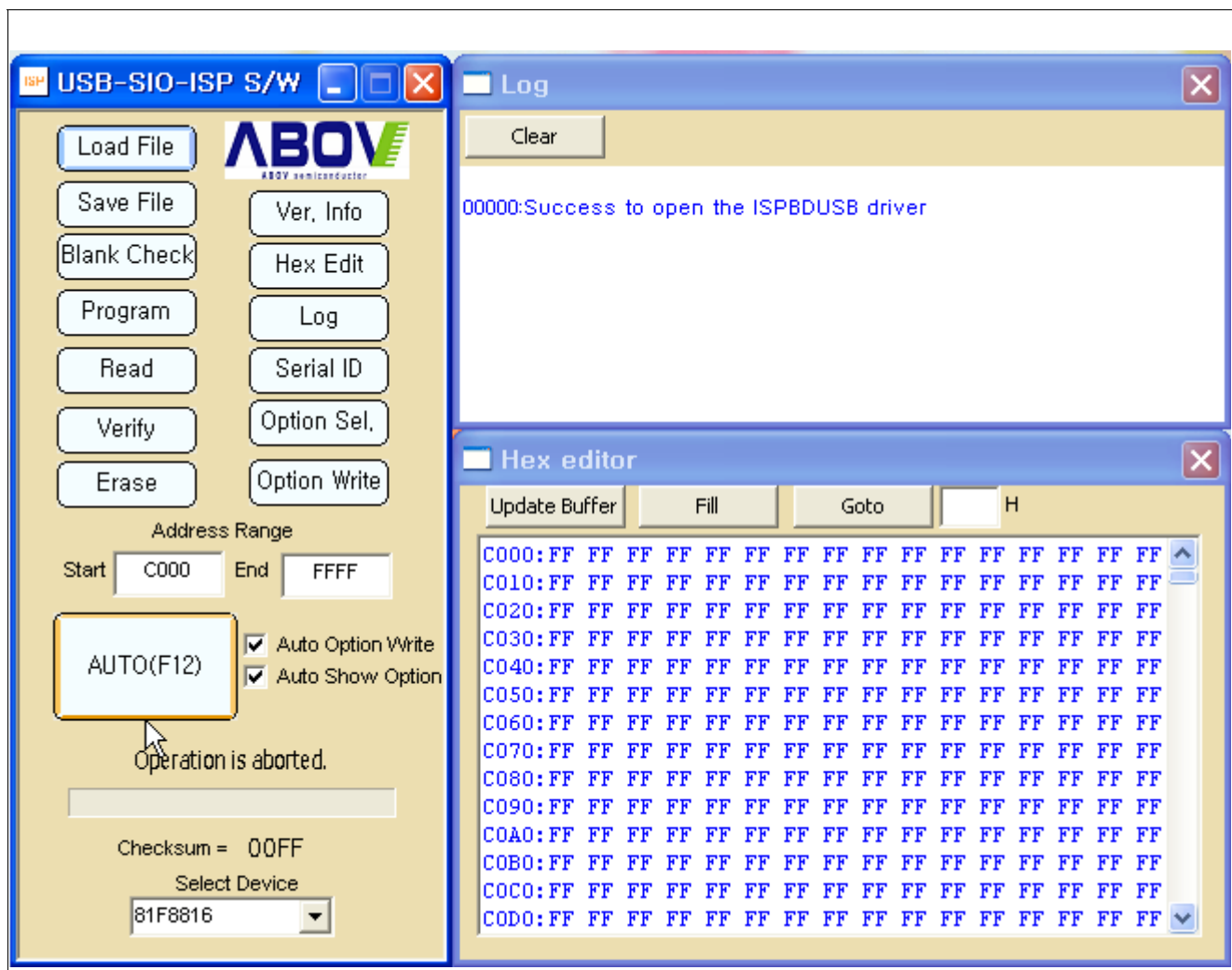


Figure 31-1 ISP software

### 31.2 Basic ISP S/W Information

The Figure 31-1 is the ISP software based on Windows™. This software is only supporting devices with SIO. In case

of not detecting its baudrates an user manually have to select specific baudrates.

Function	Description
Load File	Load the data from the selected file storage into the memory buffer.
Save File	Save the current data in your memory buffer to a disk storage by using the Intel Motorola HEX format.
Blank Check	Verify whether or not a device is in an erased or unprogrammed state.
Program	This button enables you to place new data from the memory buffer into the target device.
Read	Read the data in the target MCU into the buffer for examination. The checksum will be displayed on the checksum box.
Verify	Assures that data in the device matches data in the memory buffer. If your device is secured, a verification error is detected.
Erase	Erase the data in your target MCU before programming it.
Option Selection	Set the configuration data of target MCU. The security locking is set with this button.
Option Write	Program the configuration data of target MCU. The security locking is performed with this button.
Start _____	Starting address
End _____	End address
AUTO	Following sequence is performed ; 1.Erase 2.Program 3.Verify 4.Option Write
Auto Option Write	If you want to program the option(config) value after pressing the Auto Button, chek this button
Auto Show Option	If you check this button, the option(config) dialog is displayed whenever pressing the Auto button.
Checksum	Display the checksum(Hexdecimal) after reading the target device.
Select Device	Select target device. You need to select a device before turning on the target VDD
Update Buffer	Update buffer by pressing this button.
Fill	Fill the selected area with a data.
Goto	Display the selected page.
Serial ID	To program the serial ID.

**Table 31-1 ISP Function Description**

**Note:** MCU configuration value is erased after operation. It must be configured to match with user target board. Otherwise, it is failed to enter ISP mode, or its operation is not de-

scribable.

### 31.3 Hardware Conditions to Enter the ISP Mode

The boot loader can be executed by holding  $\overline{\text{ALE}}$  high,  $\overline{\text{RESET/V}_{\text{PP}}}$  as +9V.

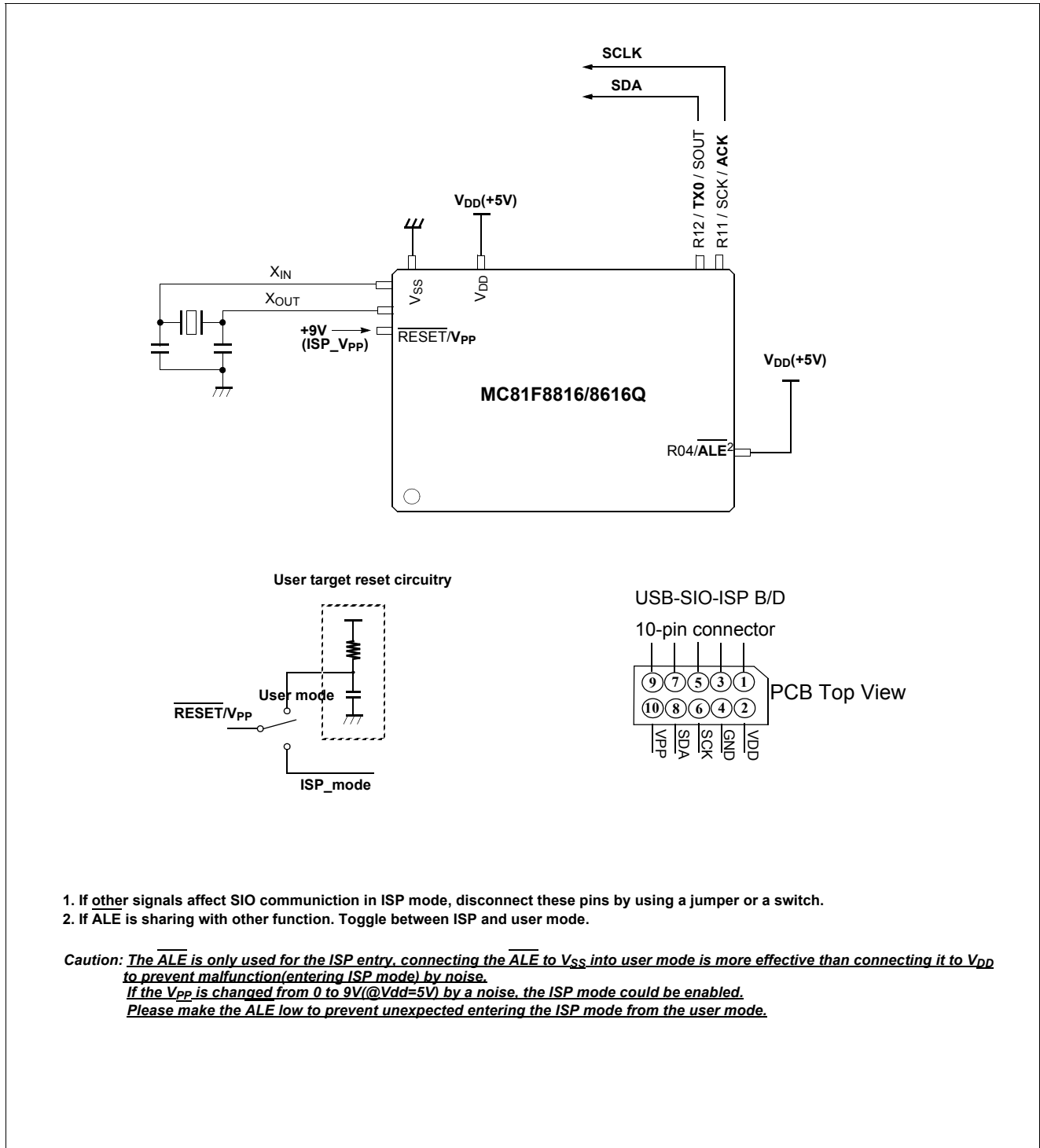


Figure 31-2 ISP Configuration

### 31.4 Sequence to enter ISP mode/user mode

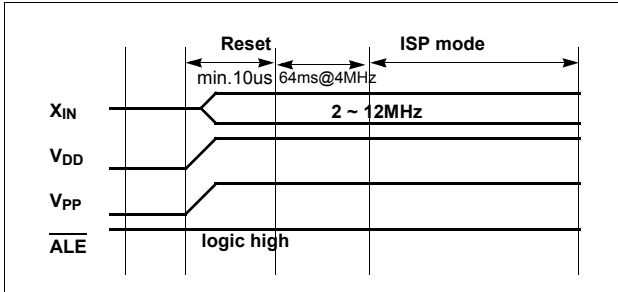


Figure 31-3 Timing diagram to enter the ISP mode

1. Power off a target system.
2. Configure a target system as ISP mode.
3. Attach a ISP B/D into a target system.
4. Run the ISP S/W and Select Device.
5. Power on a target system.

Sequence to enter user mode from ISP mode.

1. Close the ISP S/W..
2. Power off a target system.
3. Configure a target system as user mode
4. Detach a ISP B/D from a target system.
5. Power on.

Sequence to enter ISP mode from user mode.

### 31.5 USB-SIO-ISP Board

The ISP software and hardware circuit diagram are provided at [www.abov.co.kr](http://www.abov.co.kr).

To get a ISP B/D, contact to sales department. The following circuit diagram is for reference use.

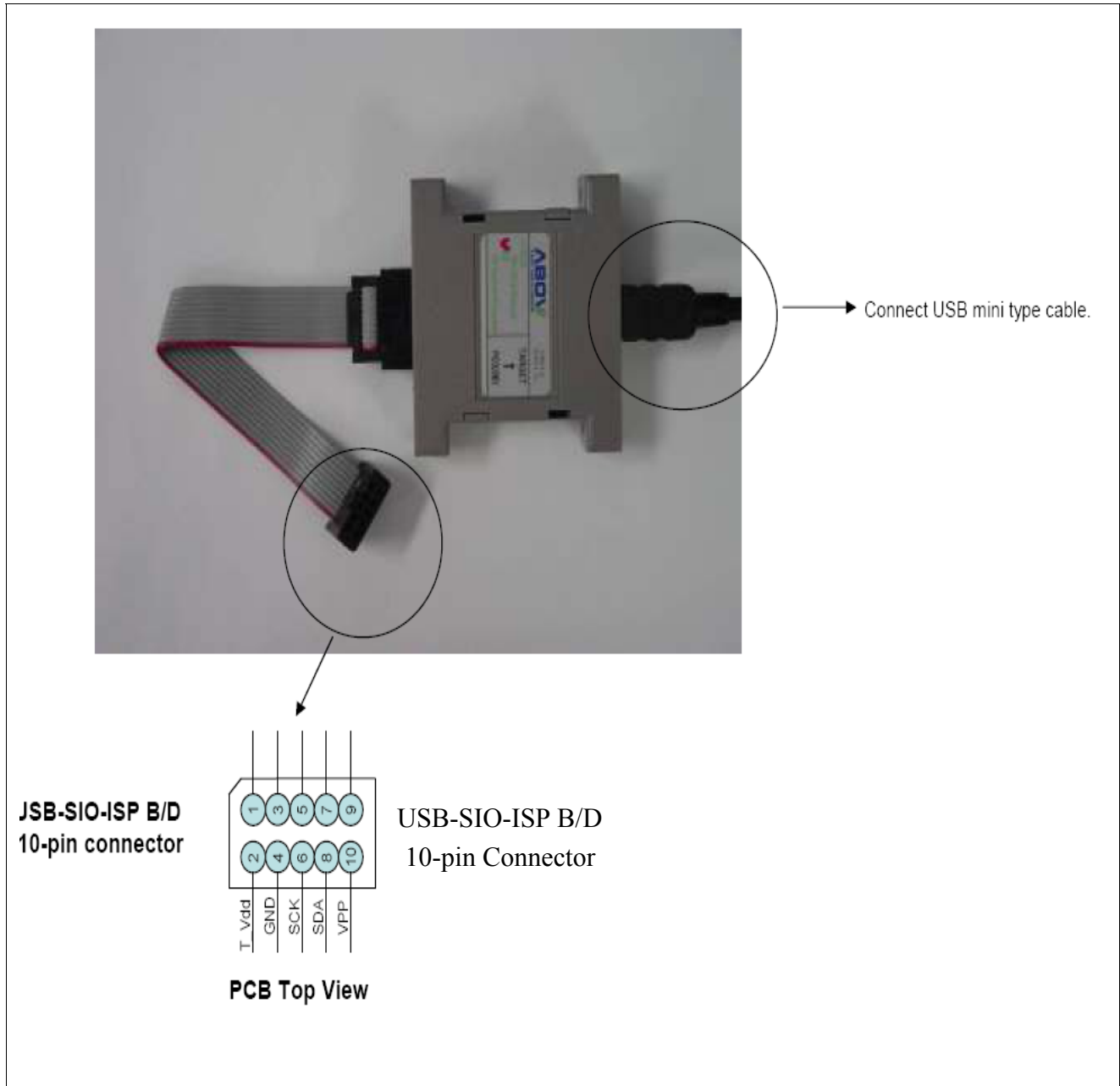


Figure 31-4 ISP board supplied by ABOV



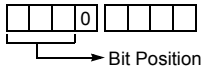
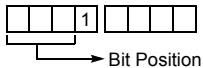
# APPENDIX

---

---

## A. INSTRUCTION

### A.1 Terminology List

Terminology	Description
A	Accumulator
X	X - register
Y	Y - register
PSW	Program Status Word
#imm	8-bit Immediate data
dp	Direct Page Offset Address
!abs	Absolute Address
[]	Indirect expression
{}	Register Indirect expression
{ }+	Register Indirect expression, after that, Register auto-increment
.bit	Bit Position
A.bit	Bit Position of Accumulator
dp.bit	Bit Position of Direct Page Memory
M.bit	Bit Position of Memory Data (000 <sub>H</sub> ~0FFF <sub>H</sub> )
rel	Relative Addressing Data
upage	U-page (0FF00 <sub>H</sub> ~0FFFF <sub>H</sub> ) Offset Address
n	Table CALL Number (0~15)
+	Addition
x	 Upper Nibble Expression in Opcode
y	 Upper Nibble Expression in Opcode
-	Subtraction
×	Multiplication
/	Division
()	Contents Expression
^	AND
∨	OR
⊕	Exclusive OR
~	NOT
←	Assignment / Transfer / Shift Left
→	Shift Right
↔	Exchange
=	Equal
≠	Not Equal

## A.2 Instruction Map

LOW HIGH	0000 00	00001 01	00010 02	00011 03	00100 04	00101 05	00110 06	00111 07	01000 08	01001 09	01010 0A	01011 0B	01100 0C	01101 0D	01110 0E	01111 0F
000	-	SET1 dp.bit	BBS A.bit,rel	BBS dp.bit,rel	ADC #imm	ADC dp	ADC dp+X	ADC !abs	ASL A	ASL dp	TCAL L 0	SETA 1 .bit	BIT dp	POP A	PUSH A	BRK
001	CLRC				SBC #imm	SBC dp	SBC dp+X	SBC !abs	ROL A	ROL dp	TCAL L 2	CLRA 1 .bit	COM dp	POP X	PUSH X	BRA rel
010	CLRG				CMP #imm	CMP dp	CMP dp+X	CMP !abs	LSR A	LSR dp	TCAL L 4	NOT1 M.bit	TST dp	POP Y	PUSH Y	PCAL L Upage
011	DI				OR #imm	OR dp	OR dp+X	OR !abs	ROR A	ROR dp	TCAL L 6	OR1 OR1B	CMPX dp	POP PSW	PUSH PSW	RET
100	CLR V				AND #imm	AND dp	AND dp+X	AND !abs	INC A	INC dp	TCAL L 8	AND1 AND1 B	CMPY dp	CBNE dp+X	TXSP	INC X
101	SETC				EOR #imm	EOR dp	EOR dp+X	EOR !abs	DEC A	DEC dp	TCAL L 10	EOR1 EOR1 B	DBNE dp	XMA dp+X	TSPX	DEC X
110	SETG				LDA #imm	LDA dp	LDA dp+X	LDA !abs	TXA	LDY dp	TCAL L 12	LDC LDCB	LDX dp	LDX dp+Y	XCN	DAS (N/A)
111	EI				LDM dp,#imm	STA dp	STA dp+X	STA !abs	TAX	STY dp	TCAL L 14	STC M.bit	STX dp	STX dp+Y	XAX	STOP

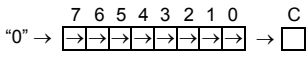
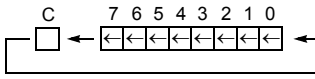
LOW HIGH	10000 10	10001 11	10010 12	10011 13	10100 14	10101 15	10110 16	10111 17	11000 18	11001 19	11010 1A	11011 1B	11100 1C	11101 1D	11110 1E	11111 1F
000	BPL rel	CLR1 dp.bit	BBC A.bit,rel	BBC dp.bit,rel	ADC {X}	ADC !abs+Y	ADC [dp+X]	ADC [dp]+Y	ASL !abs	ASL dp+X	TCAL L 1	JMP !abs	BIT !abs	ADD W dp	LDX #imm	JMP [!abs]
001	BVC rel				SBC {X}	SBC !abs+Y	SBC [dp+X]	SBC [dp]+Y	ROL !abs	ROL dp+X	TCAL L 3	CALL !abs	TEST !abs	SUB W dp	LDY #imm	JMP [dp]
010	BCC rel				CMP {X}	CMP !abs+Y	CMP [dp+X]	CMP [dp]+Y	LSR !abs	LSR dp+X	TCAL L 5	MUL	TCLR 1 !abs	CMP W dp	CMPX #imm	CALL [dp]
011	BNE rel				OR {X}	OR !abs+Y	OR [dp+X]	OR [dp]+Y	ROR !abs	ROR dp+X	TCAL L 7	DBNE Y	CMPX !abs	LDYA dp	CMPY #imm	RETI
100	BMI rel				AND {X}	AND !abs+Y	AND [dp+X]	AND [dp]+Y	INC !abs	INC dp+X	TCAL L 9	DIV	CMPY !abs	INCW dp	INC Y	TAY
101	BVS rel				EOR {X}	EOR !abs+Y	EOR [dp+X]	EOR [dp]+Y	DEC !abs	DEC dp+X	TCAL L 11	XMA {X}	XMA dp	DEC W dp	DEC Y	TYA

110	BCS rel				LDA {X}	LDA !abs+ Y	LDA [dp+X]	LDA [dp]+Y	LDY !abs	LDY dp+X	TCAL L 13	LDA {X}+	LDX !abs	STYA dp	XAY	DAA (N/A)
111	BEQ rel				STA {X}	STA !abs+ Y	STA [dp+X]	STA [dp]+Y	STY !abs	STY dp+X	TCAL L 15	STA {X}+	STX !abs	CBNE dp	XYX	NOP

### A.3 Instruction Set

#### Arithmetic / Logic Operation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	ADC #imm	04	2	2	Add with carry. $A \leftarrow (A) + (M) + C$	NV--H-ZC
2	ADC dp	05	2	3		
3	ADC dp + X	06	2	4		
4	ADC !abs	07	3	4		
5	ADC !abs + Y	15	3	5		
6	ADC [ dp + X ]	16	2	6		
7	ADC [ dp ] + Y	17	2	6		
8	ADC { X }	14	1	3		
9	AND #imm	84	2	2	Logical AND $A \leftarrow (A) \wedge (M)$	N-----Z-
10	AND dp	85	2	3		
11	AND dp + X	86	2	4		
12	AND !abs	87	3	4		
13	AND !abs + Y	95	3	5		
14	AND [ dp + X ]	96	2	6		
15	AND [ dp ] + Y	97	2	6		
16	AND { X }	94	1	3		
17	ASL A	08	1	2	Arithmetic shift left C    7 6 5 4 3 2 1 0 □ ← [←←←←←←←←] ← "0"	N-----ZC
18	ASL dp	09	2	4		
19	ASL dp + X	19	2	5		
20	ASL !abs	18	3	5		
21	CMP #imm	44	2	2	Compare accumulator contents with memory contents $(A) - (M)$	N-----ZC
22	CMP dp	45	2	3		
23	CMP dp + X	46	2	4		
24	CMP !abs	47	3	4		
25	CMP !abs + Y	55	3	5		
26	CMP [ dp + X ]	56	2	6		
27	CMP [ dp ] + Y	57	2	6		
28	CMP { X }	54	1	3		
29	CMPX #imm	5E	2	2	Compare X contents with memory contents $(X) - (M)$	N-----ZC
30	CMPX dp	6C	2	3		
31	CMPX !abs	7C	3	4		
32	CMPY #imm	7E	2	2	Compare Y contents with memory contents $(Y) - (M)$	N-----ZC
33	CMPY dp	8C	2	3		
34	CMPY !abs	9C	3	4		
35	COM dp	2C	2	4	1'S Complement : $(dp) \leftarrow \sim(dp)$	N-----Z-
36	DAA	-	-	-	Not supported	
37	DAS	-	-	-	Not supported	

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
38	DEC A	A8	1	2	Decrement $M \leftarrow (M) - 1$	N-----Z-
39	DEC dp	A9	2	4		
40	DEC dp + X	B9	2	5		
41	DEC !abs	B8	3	5		
42	DEC X	AF	1	2		
43	DEC Y	BE	1	2		
44	DIV	9B	1	12	Divide : YA / X Q: A, R: Y	NV--H-Z-
45	EOR #imm	A4	2	2	Exclusive OR $A \leftarrow (A) \oplus (M)$	N-----Z-
46	EOR dp	A5	2	3		
47	EOR dp + X	A6	2	4		
48	EOR !abs	A7	3	4		
49	EOR !abs + Y	B5	3	5		
50	EOR [ dp + X ]	B6	2	6		
51	EOR [ dp ] + Y	B7	2	6		
52	EOR { X }	B4	1	3		
53	INC A	88	1	2	Increment $M \leftarrow (M) + 1$	N-----Z-
54	INC dp	89	2	4		
55	INC dp + X	99	2	5		
56	INC !abs	98	3	5		
57	INC X	8F	1	2		
58	INC Y	9E	1	2	Logical shift right "0" →  → C	N-----ZC
59	LSR A	48	1	2		
60	LSR dp	49	2	4		
61	LSR dp + X	59	2	5		
62	LSR !abs	58	3	5	Multiply : $YA \leftarrow Y \times A$	N-----Z-
63	MUL	5B	1	9		
64	OR #imm	64	2	2	Logical OR $A \leftarrow (A) \vee (M)$	N-----Z-
65	OR dp	65	2	3		
66	OR dp + X	66	2	4		
67	OR !abs	67	3	4		
68	OR !abs + Y	75	3	5		
69	OR [ dp + X ]	76	2	6		
70	OR [ dp ] + Y	77	2	6		
71	OR { X }	74	1	3	Rotate left through Carry 	N-----ZC
72	ROL A	28	1	2		
73	ROL dp	29	2	4		
74	ROL dp + X	39	2	5		
75	ROL !abs	38	3	5		

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
76	ROR A	68	1	2	Rotate right through Carry 	N-----ZC
77	ROR dp	69	2	4		
78	ROR dp + X	79	2	5		
79	ROR !abs	78	3	5		
80	SBC #imm	24	2	2	Subtract with Carry $A \leftarrow (A) - (M) - \sim(C)$	NV--HZC
81	SBC dp	25	2	3		
82	SBC dp + X	26	2	4		
83	SBC !abs	27	3	4		
84	SBC !abs + Y	35	3	5		
85	SBC [ dp + X ]	36	2	6		
86	SBC [ dp ] + Y	37	2	6		
87	SBC { X }	34	1	3		
88	TST dp	4C	2	3	Test memory contents for negative or zero, ( dp ) - 00 <sub>H</sub>	N-----Z-
89	XCN	CE	1	5	Exchange nibbles within the accumulator $A_7 \sim A_4 \leftrightarrow A_3 \sim A_0$	N-----Z-

## Register / Memory Operation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	LDA #imm	C4	2	2	Load accumulator	N-----Z-
2	LDA dp	C5	2	3	$A \leftarrow (M)$	
3	LDA dp + X	C6	2	4		
4	LDA !abs	C7	3	4		
5	LDA !abs + Y	D5	3	5		
6	LDA [ dp + X ]	D6	2	6		
7	LDA [ dp ] + Y	D7	2	6		
8	LDA { X }	D4	1	3		
9	LDA { X }+	DB	1	4	X- register auto-increment : $A \leftarrow (M), X \leftarrow X + 1$	
10	LDM dp,#imm	E4	3	5	Load memory with immediate data : $(M) \leftarrow \text{imm}$	-----
11	LDX #imm	1E	2	2	Load X-register	N-----Z-
12	LDX dp	CC	2	3	$X \leftarrow (M)$	
13	LDX dp + Y	CD	2	4		
14	LDX !abs	DC	3	4		
15	LDY #imm	3E	2	2	Load Y-register	N-----Z-
16	LDY dp	C9	2	3	$Y \leftarrow (M)$	
17	LDY dp + X	D9	2	4		
18	LDY !abs	D8	3	4		
19	STA dp	E5	2	4	Store accumulator contents in memory	-----
20	STA dp + X	E6	2	5	$(M) \leftarrow A$	
21	STA !abs	E7	3	5		
22	STA !abs + Y	F5	3	6		
23	STA [ dp + X ]	F6	2	7		
24	STA [ dp ] + Y	F7	2	7		
25	STA { X }	F4	1	4		
26	STA { X }+	FB	1	4	X- register auto-increment : $(M) \leftarrow A, X \leftarrow X + 1$	
27	STX dp	EC	2	4	Store X-register contents in memory	-----
28	STX dp + Y	ED	2	5	$(M) \leftarrow X$	
29	STX !abs	FC	3	5		
30	STY dp	E9	2	4	Store Y-register contents in memory	-----
31	STY dp + X	F9	2	5	$(M) \leftarrow Y$	
32	STY !abs	F8	3	5		
33	TAX	E8	1	2	Transfer accumulator contents to X-register : $X \leftarrow A$	N-----Z-
34	TAY	9F	1	2	Transfer accumulator contents to Y-register : $Y \leftarrow A$	N-----Z-
35	TSPX	AE	1	2	Transfer stack-pointer contents to X-register : $X \leftarrow \text{sp}$	N-----Z-
36	TXA	C8	1	2	Transfer X-register contents to accumulator: $A \leftarrow X$	N-----Z-
37	TXSP	8E	1	2	Transfer X-register contents to stack-pointer: $\text{sp} \leftarrow X$	N-----Z-
38	TYA	BF	1	2	Transfer Y-register contents to accumulator: $A \leftarrow Y$	N-----Z-



39	XAX	EE	1	4	Exchange X-register contents with accumulator :X ↔ A	-----
40	XAY	DE	1	4	Exchange Y-register contents with accumulator :Y ↔ A	-----
41	XMA dp	BC	2	5	Exchange memory contents with accumulator ( M ) ↔ A	N-----Z-
42	XMA dp+X	AD	2	6		
43	XMA {X}	BB	1	5		
44	XYX	FE	1	4	Exchange X-register contents with Y-register : X ↔ Y	-----

### 16-BIT operation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	ADDW dp	1D	2	5	16-Bits add without Carry $YA \leftarrow (YA) + (dp + 1)(dp)$	NV--H-ZC
2	CMPW dp	5D	2	4	Compare YA contents with memory pair contents : $(YA) - (dp+1)(dp)$	N-----ZC
3	DECW dp	BD	2	6	Decrement memory pair $(dp+1)(dp) \leftarrow (dp+1)(dp) - 1$	N-----Z-
4	INCW dp	9D	2	6	Increment memory pair $(dp+1)(dp) \leftarrow (dp+1)(dp) + 1$	N-----Z-
5	LDYA dp	7D	2	5	Load YA $YA \leftarrow (dp + 1)(dp)$	N-----Z-
6	STYA dp	DD	2	5	Store YA $(dp + 1)(dp) \leftarrow YA$	-----
7	SUBW dp	3D	2	5	16-Bits subtract without carry $YA \leftarrow (YA) - (dp + 1)(dp)$	NV--H-ZC

### Bit Manipulation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	AND1 M.bit	8B	3	4	Bit AND C-flag : $C \leftarrow (C) \wedge (M.bit)$	-----C
2	AND1B M.bit	8B	3	4	Bit AND C-flag and NOT : $C \leftarrow (C) \wedge \sim(M.bit)$	-----C
3	BIT dp	0C	2	4	Bit test A with memory : $Z \leftarrow (A) \wedge (M), N \leftarrow (M_7), V \leftarrow (M_6)$	MM----Z-
4	BIT labs	1C	3	5		
5	CLR1 dp.bit	y1	2	4	Clear bit : $(M.bit) \leftarrow "0"$	-----
6	CLRA1 A.bit	2B	2	2	Clear A bit : $(A.bit) \leftarrow "0"$	-----
7	CLRC	20	1	2	Clear C-flag : $C \leftarrow "0"$	-----0
8	CLRG	40	1	2	Clear G-flag : $G \leftarrow "0"$	--0-----
9	CLRV	80	1	2	Clear V-flag : $V \leftarrow "0"$	-0--0---
10	EOR1 M.bit	AB	3	5	Bit exclusive-OR C-flag : $C \leftarrow (C) \oplus (M.bit)$	-----C
11	EOR1B M.bit	AB	3	5	Bit exclusive-OR C-flag and NOT : $C \leftarrow (C) \oplus \sim(M.bit)$	-----C
12	LDC M.bit	CB	3	4	Load C-flag : $C \leftarrow (M.bit)$	-----C
13	LDCB M.bit	CB	3	4	Load C-flag with NOT : $C \leftarrow \sim(M.bit)$	-----C
14	NOT1 M.bit	4B	3	5	Bit complement : $(M.bit) \leftarrow \sim(M.bit)$	-----
15	OR1 M.bit	6B	3	5	Bit OR C-flag : $C \leftarrow (C) \vee (M.bit)$	-----C

16	OR1B M.bit	6B	3	5	Bit OR C-flag and NOT : $C \leftarrow (C) \vee \sim(M.bit)$	-----C
17	SET1 dp.bit	x1	2	4	Set bit : $(M.bit) \leftarrow "1"$	-----
18	SETA1 A.bit	0B	2	2	Set A bit : $(A.bit) \leftarrow "1"$	-----
19	SETC	A0	1	2	Set C-flag : $C \leftarrow "1"$	-----1
20	SETG	C0	1	2	Set G-flag : $G \leftarrow "1"$	--1-----
21	STC M.bit	EB	3	6	Store C-flag : $(M.bit) \leftarrow C$	-----
22	TCLR1 !abs	5C	3	6	Test and clear bits with A : $A - (M), (M) \leftarrow (M) \wedge \sim(A)$	N-----Z-
23	TSET1 !abs	3C	3	6	Test and set bits with A : $A - (M), (M) \leftarrow (M) \vee (A)$	N-----Z-

**Branch / Jump Operation**

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	BBC A.bit,rel	y2	2	4/6	Branch if bit clear :	-----
2	BBC dp.bit,rel	y3	3	5/7	if ( bit ) = 0 , then $pc \leftarrow ( pc ) + rel$	
3	BBS A.bit,rel	x2	2	4/6	Branch if bit set :	-----
4	BBS dp.bit,rel	x3	3	5/7	if ( bit ) = 1 , then $pc \leftarrow ( pc ) + rel$	
5	BCC rel	50	2	2/4	Branch if carry bit clear if ( C ) = 0 , then $pc \leftarrow ( pc ) + rel$	-----
6	BCS rel	D0	2	2/4	Branch if carry bit set if ( C ) = 1 , then $pc \leftarrow ( pc ) + rel$	-----
7	BEQ rel	F0	2	2/4	Branch if equal if ( Z ) = 1 , then $pc \leftarrow ( pc ) + rel$	-----
8	BMI rel	90	2	2/4	Branch if minus if ( N ) = 1 , then $pc \leftarrow ( pc ) + rel$	-----
9	BNE rel	70	2	2/4	Branch if not equal if ( Z ) = 0 , then $pc \leftarrow ( pc ) + rel$	-----
10	BPL rel	10	2	2/4	Branch if minus if ( N ) = 0 , then $pc \leftarrow ( pc ) + rel$	-----
11	BRA rel	2F	2	4	Branch always $pc \leftarrow ( pc ) + rel$	-----
12	BVC rel	30	2	2/4	Branch if overflow bit clear if ( V ) = 0 , then $pc \leftarrow ( pc ) + rel$	-----
13	BVS rel	B0	2	2/4	Branch if overflow bit set if ( V ) = 1 , then $pc \leftarrow ( pc ) + rel$	-----
14	CALL !abs	3B	3	8	Subroutine call	
15	CALL [dp]	5F	2	8	$M(sp) \leftarrow ( pc_H )$ , $sp \leftarrow sp - 1$ , $M(sp) \leftarrow ( pc_L )$ , $sp \leftarrow sp - 1$ , if !abs, $pc \leftarrow abs$ ; if [dp], $pc_L \leftarrow ( dp )$ , $pc_H \leftarrow ( dp + 1 )$ .	-----
16	CBNE dp,rel	FD	3	5/7	Compare and branch if not equal :	-----
17	CBNE dp+X,rel	8D	3	6/8	if ( A ) $\neq$ ( M ) , then $pc \leftarrow ( pc ) + rel$ .	
18	DBNE dp,rel	AC	3	5/7	Decrement and branch if not equal :	-----
19	DBNE Y,rel	7B	2	4/6	if ( M ) $\neq$ 0 , then $pc \leftarrow ( pc ) + rel$ .	
20	JMP !abs	1B	3	3	Unconditional jump	
21	JMP [!abs]	1F	3	5	$pc \leftarrow$ jump address	-----
22	JMP [dp]	3F	2	4		
23	PCALL upage	4F	2	6	U-page call $M(sp) \leftarrow ( pc_H )$ , $sp \leftarrow sp - 1$ , $M(sp) \leftarrow ( pc_L )$ , $sp \leftarrow sp - 1$ , $pc_L \leftarrow ( upage )$ , $pc_H \leftarrow "0FFH"$ .	-----
24	TCALL n	nA	1	8	Table call : $(sp) \leftarrow ( pc_H )$ , $sp \leftarrow sp - 1$ , $M(sp) \leftarrow ( pc_L )$ , $sp \leftarrow sp - 1$ , $pc_L \leftarrow ( Table \ vector \ L )$ , $pc_H \leftarrow ( Table \ vector \ H )$	-----

## Control Operation &amp; Etc.

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	BRK	0F	1	8	Software interrupt : $B \leftarrow "1"$ , $M(sp) \leftarrow (pc_H)$ , $sp \leftarrow sp-1$ , $M(s) \leftarrow (pc_L)$ , $sp \leftarrow sp - 1$ , $M(sp) \leftarrow (PSW)$ , $sp \leftarrow sp - 1$ , $pc_L \leftarrow (0FFDE_H)$ , $pc_H \leftarrow (0FFDF_H)$ .	---1-0--
2	DI	60	1	3	Disable all interrupts : $I \leftarrow "0"$	-----0--
3	EI	E0	1	3	Enable all interrupt : $I \leftarrow "1"$	-----1--
4	NOP	FF	1	2	No operation	-----
5	POP A	0D	1	4	$sp \leftarrow sp + 1$ , $A \leftarrow M(sp)$	restored
6	POP X	2D	1	4	$sp \leftarrow sp + 1$ , $X \leftarrow M(sp)$	
7	POP Y	4D	1	4	$sp \leftarrow sp + 1$ , $Y \leftarrow M(sp)$	
8	POP PSW	6D	1	4	$sp \leftarrow sp + 1$ , $PSW \leftarrow M(sp)$	
9	PUSH A	0E	1	4	$M(sp) \leftarrow A$ , $sp \leftarrow sp - 1$	-----
10	PUSH X	2E	1	4	$M(sp) \leftarrow X$ , $sp \leftarrow sp - 1$	
11	PUSH Y	4E	1	4	$M(sp) \leftarrow Y$ , $sp \leftarrow sp - 1$	
12	PUSH PSW	6E	1	4	$M(sp) \leftarrow PSW$ , $sp \leftarrow sp - 1$	
13	RET	6F	1	5	Return from subroutine $sp \leftarrow sp + 1$ , $pc_L \leftarrow M(sp)$ , $sp \leftarrow sp + 1$ , $pc_H \leftarrow M(sp)$	-----
14	RETI	7F	1	6	Return from interrupt $sp \leftarrow sp + 1$ , $PSW \leftarrow M(sp)$ , $sp \leftarrow sp + 1$ , $pc_L \leftarrow M(sp)$ , $sp \leftarrow sp + 1$ , $pc_H \leftarrow M(sp)$	restored
15	STOP	EF	1	3	Stop mode ( halt CPU, stop oscillator )	-----

**B. MASK ORDER SHEET(MC81C8816)**

**MASK ORDER & VERIFICATION SHEET**

**MC81C88**

**-LE**

16 : 16K  
60 : 60K

Blank:Pb Free PKG  
B:Pb/Halogen Free PKG

Q:MQFP  
L:LQFP

Customer should write inside thick line box.

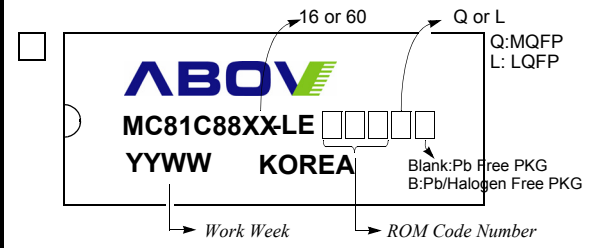
**1. Customer Information**

Company Name			
Application			
Order Date	YYYY	MM	DD
Tel:	Fax:		
E-mail:			
Name & Signature:			

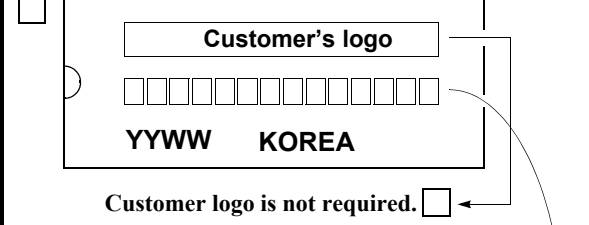
**2. Device Information**

Package	<input type="checkbox"/> 80MQFP	
ROM Size	<input type="checkbox"/> 16K	
Unused ROM	<input type="checkbox"/> 00H	<input type="checkbox"/> FFH
POR/R47 Use	<input type="checkbox"/> Yes	<input type="checkbox"/> No
X2EN	<input type="checkbox"/> Yes	<input type="checkbox"/> No
ONP Use	<input type="checkbox"/> Yes	<input type="checkbox"/> No
If ONP is "Yes",	CLK Use	<input type="checkbox"/> Crystal <input type="checkbox"/> IN4M <input type="checkbox"/> IN8M
	CLK Use	<input type="checkbox"/> Crystal
If ONP is "No",	CLK Use	<input type="checkbox"/> Crystal
Mask Data	File Name: ( .OTP )	
	Check Sum: ( )	
	C000H	.OTP file data
	FFFFH	

**3. Marking Specification**



*Work Week*      *ROM Code Number*



Customer logo is not required.

If the customer logo must be used in the special mark, please submit a clean original of the logo.

Customer's part number

(Please check mark into  )

**4. Delivery Schedule**

	Date	Quantity	ABOV Confirmation
Customer Sample	YYYY MM DD	pcs	
Risk Order		pcs	

**5. ROM Code Verification**

Verification Date:	YYYY MM DD	Tel:	Fax:
Check Sum:		Name & Signature:	
E-mail:			



C. MASK ORDER SHEET(MC81C8616)

MASK ORDER & VERIFICATION SHEET

MC81C86

-LE

16 : 16K  
60 : 60K

Blank:Pb Free PKG  
B:Pb/Halogen Free PKG

Q:MQFP  
L:LQFP

Customer should write inside thick line box.

1. Customer Information

Company Name			
Application			
Order Date	YYYY	MM	DD
Tel:	Fax:		
E-mail:			
Name & Signature:			

2. Device Information

Package	<input type="checkbox"/> 64MQFP	<input type="checkbox"/> 64LQFP
ROM Size	<input type="checkbox"/> 16K	
Unused ROM	<input type="checkbox"/> 00H	<input type="checkbox"/> FFH
POR/R47 Use	<input type="checkbox"/> Yes	<input type="checkbox"/> No
X2EN	<input type="checkbox"/> Yes	<input type="checkbox"/> No
ONP Use	<input type="checkbox"/> Yes	<input type="checkbox"/> No
If ONP is "Yes",	CLK Use	<input type="checkbox"/> Crystal <input type="checkbox"/> IN4M <input type="checkbox"/> IN8M
	CLK Use	<input type="checkbox"/> Crystal
If ONP is "No",	CLK Use	<input type="checkbox"/> Crystal
Mask Data	File Name: (                      ).OTP Check Sum: (                      )	
	C000H	.OTP file data
	FFFFH	

3. Marking Specification

16 or 60 Q or L Q:MQFP L:LQFP Blank:Pb Free PKG B:Pb/Halogen Free PKG Work Week ROM Code Number

Customer's logo YYWW KOREA

Customer logo is not required.

If the customer logo must be used in the special mark, please submit a clean original of the logo.

Customer's part number

(Please check mark into  )

4. Delivery Schedule

	Date	Quantity	ABOV Confirmation
Customer Sample	YYYY MM DD	pcs	
Risk Order		pcs	

5. ROM Code Verification

Verification Date:	MM DD	Tel:	Fax:
Check Sum:		Name & Signature:	
E-mail:			





