

ABOV SEMICONDUCTOR CO., LTD.  
8-BIT SINGLE-CHIP MICROCONTROLLERS

# MC80F0504/0604

*User's Manual (Ver. 1.47)*



---

**Version 1.47**

**Published by  
FAE Team**

**©2006 ABOV semiconductor Ltd. All right reserved.**

---

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

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 1.47 (NOV.2011) This book**

Logo is changed.

**VERSION 1.46 (MAY.2008)**

Change POR and Internal OSC characteristics on chapter '7.Electrical Characteristics'.

**VERSION 1.45 (MAR.2008)**

Add POR characteristic on chapter '7.Electrical Characteristics'.

**VERSION 1.44 (FEB.2008)**

Repalce 'TBD' with real data in '7.Electrical Characteristics'.

Amend the contents of '21.Device Configuration Area' chapter.

**VERSION 1.43 (DEC.2007)**

Add 20 SSOP package info

**VERSION 1.42 (NOV.2007)**

Add '16 SOP( 153 mil)' package info

**VERSION 1.41 (APL.2007)**

Add T<sub>VDD</sub> parameter specification and change T<sub>POR</sub> in DC Electrical Characteristics.

Note for configuration option is added and fix some errata.

**VERSION 1.4 (MAR.2007)**

Add T<sub>VDD</sub> parameter specification and change T<sub>POR</sub> in DC Electrical Characteristics.

Note for configuration option is added and fix some errata.

**VERSION 1.3 (JUL.2006)**

Correct Interrupts Sequence and example codes in Chapter 19.

**VERSION 1.2 (JUN.2006)**

Correct the description of TM1 in Figure 13-1.  $f_{XIN}/2$ ,  $f_{XIN}/8$  and timer0 instead of  $f_{XIN}/4$ ,  $f_{XIN}/16$  and timer2 are selected when TICK[1..0] is "01b", "10b", "11b" respectively.

**VERSION 1.1 (MAY.2006)**

Correct direction symble in Chapter 3. Pin Assignment.(RESET pin as Input, XOUT pin as Input/Output)

Fix some font error in chapter 22. Emulator EVA. Board Setting.

**VERSION 1.0 (APR. 2006)**

Update Electrical Characteristics

**VERSION 0.4 (APR. 2006)**

Change Flash Endurance 1000 times to 100 times.

**VERSION 0.3 (MAR. 2006)**

The company name, MagnaChip Semiconductor Ltd. changed to ABOV Semiconductor Co.,Ltd..

Fix some errata.

**VERSION 0.2 (NOV. 2005)**

Fix some errata.

**VERSION 0.1 (OCT. 2005)**

First Edition



# Table of Contents

<b>1. OVERVIEW</b> .....	<b>1</b>	<b>13. TIMER/EVENT COUNTER</b> .....	<b>47</b>
Description .....	1	8-bit Timer / Counter Mode .....	49
Features .....	1	16-bit Timer / Counter Mode .....	53
Development Tools .....	2	8-bit (16-bit) Compare Output .....	53
Ordering Information .....	3	8-bit Capture Mode .....	54
<b>2. BLOCK DIAGRAM</b> .....	<b>4</b>	16-bit Capture Mode .....	57
<b>3. PIN ASSIGNMENT</b> .....	<b>5</b>	PWM Mode .....	58
<b>4. PACKAGE DRAWING</b> .....	<b>6</b>	<b>14. ANALOG TO DIGITAL CONVERTER</b> .....	<b>61</b>
<b>5. PIN FUNCTION</b> .....	<b>10</b>	<b>15. BUZZER FUNCTION</b> .....	<b>64</b>
<b>6. PORT STRUCTURES</b> .....	<b>12</b>	<b>16. INTERRUPTS</b> .....	<b>66</b>
<b>7. ELECTRICAL CHARACTERISTICS</b> .....	<b>15</b>	Interrupt Sequence .....	68
Absolute Maximum Ratings .....	15	BRK Interrupt .....	70
Recommended Operating Conditions .....	15	Multi Interrupt .....	70
A/D Converter Characteristics .....	15	External Interrupt .....	72
DC Electrical Characteristics .....	16	<b>17. POWER SAVING OPERATION</b> .....	<b>74</b>
AC Characteristics .....	17	Sleep Mode .....	74
Typical Characteristics .....	18	Stop Mode .....	75
<b>8. MEMORY ORGANIZATION</b> .....	<b>21</b>	Stop Mode at Internal RC-Oscillated Watchdog .....	78
Registers .....	21	Timer Mode .....	78
Program Memory .....	24	Minimizing Current Consumption .....	80
Data Memory .....	27	<b>18. RESET</b> .....	<b>82</b>
Addressing Mode .....	32	<b>19. POWER FAIL PROCESSOR</b> .....	<b>84</b>
<b>9. I/O PORTS</b> .....	<b>36</b>	<b>20. COUNTERMEASURE OF NOISE</b> .....	<b>86</b>
R0 and R0IO register .....	36	Oscillation Noise Protector .....	86
R1 and R1IO register .....	37	Oscillation Fail Processor .....	87
R3 and R3IO register .....	39	<b>21. DEVICE CONFIGURATION AREA</b> .....	<b>88</b>
<b>10. CLOCK GENERATOR</b> .....	<b>40</b>	<b>22. EMULATOR EVA. BOARD SETTING</b> .....	<b>89</b>
Oscillation Circuit .....	40	<b>A. INSTRUCTION</b> .....	<b>ii</b>
<b>11. BASIC INTERVAL TIMER</b> .....	<b>42</b>	Terminology List .....	ii
<b>12. WATCHDOG TIMER</b> .....	<b>44</b>	Instruction Map .....	iii
		Instruction Set .....	iv



# MC80F0504/0604

## CMOS SINGLE-CHIP 8-BIT MICROCONTROLLER WITH 10-BIT A/D CONVERTER

### 1. OVERVIEW

#### 1.1 Description

The MC80F0504/0604 is advanced CMOS 8-bit microcontroller with 4K bytes of FLASH. This is a powerful microcontroller which provides a highly flexible and cost effective solution to many embedded control applications. This provides the following features : 4K bytes of FLASH (MTP), 256 bytes of RAM, 8/16-bit timer/counter, watchdog timer, on-chip POR, 10-bit A/D converter, buzzer driving port, 10-bit PWM output and on-chip oscillator and clock circuitry. It also has ONP, noise filter, PFD for improving noise immunity. In addition, the MC80F0504/0604 supports power saving modes to reduce power consumption.

This document **explains the base MC80F0604**, the other's eliminated functions are same as below table.

Device Name	FLASH (ROM) Size	RAM	ADC	I/O PORT	Package
MC80F0604	4KB	256B	10 channel	18 port	20 PDIP, 20SOP, 20 SSOP
MC80F0504			8 channel	14 port	16 PDIP, 16 SOP, 16 SOP(153 mil), 16 TSSOP

Note : The DAA, DAS decimal adjust instructions are not provided in these devices.

#### 1.2 Features

- **4K Bytes On-chip FLASH (MTP)**
  - Endurance : 100 times
  - Retention time : 10 years
- **256 Bytes On-chip Data RAM (Included stack memory)**
- **Minimum Instruction Execution Time:**
  - 333ns at 12MHz (NOP instruction)
- **Programmable I/O pins (LED direct driving can be a source and sink)**
  - MC80F0604 : 18(17)
  - MC80F0504 : 14(13)
- **One 8-bit Basic Interval Timer**
- **Two 8-bit Timer/counters (or one 16-bit Timer/counter)**
- **One Watchdog timer**
- **One 10-bit High Speed PWM Outputs**
- **10-bit A/D converter**
  - MC80F0604 : 10 channels
  - MC80F0504 : 8 channels
- **One Buzzer Driving port**
  - 488Hz ~ 250kHz@4MHz
- **Two External Interrupt input ports**
- **On-chip POR (Power on Reset)**
- **Seven Interrupt sources**
  - External input : 2
  - Timer : 4
  - A/D Conversion : 1
- **Built in Noise Immunity Circuit**
  - Noise Canceller
  - PFD (Power fail detector)
  - ONP (Oscillation Noise Protector)
- **Operating Voltage & Frequency**
  - 2.2V ~ 5.5V (at 1 ~ 4MHz)
  - 2.7V ~ 5.5V (at 1 ~ 8MHz)
  - 4.5V ~ 5.5V (at 1 ~ 12MHz)
- **Operating Temperature : -40°C ~ 85°C**
- **Power Saving Modes**
  - STOP mode
  - SLEEP mode
  - RC-WDT mode
- **Oscillator Type**
  - Crystal
  - Ceramic resonator

- External RC Oscillator (C can be omitted)
- Internal Oscillator (4MHz/2MHz)

• **Package**

- 20 PDIP, SOP or SSOP
- 16 PDIP, SOP, SOP(153 mil) or TSSOP
- Available Pb free package

**1.3 Development Tools**

The MC80F0504/0604 is supported by a full-featured macro assembler, an in-circuit emulator CHOICE-Dr.™ and 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. Please contact sales part of abov semiconductor.

Software	- MS-Windows based assembler - MS-Windows based Debugger - HMS800 C compiler
Hardware (Emulator)	- CHOICE-Dr. - CHOICE-Dr. EVA80C0x B/D
Pod Name	- CHPOD80C01D-16PD - CHPOD80C02D-20PD
FLASH Writer	- CHOICE - SIGMA I/II (Single writer) - PGM Plus III (Single writer) - Standalone GANG4 I/II (Gang writer)



PGMplus III ( Single Writer )



Choice-Dr. (Emulator)



Standalone Gang4 II ( Gang Writer )



### 1.4 Ordering Information

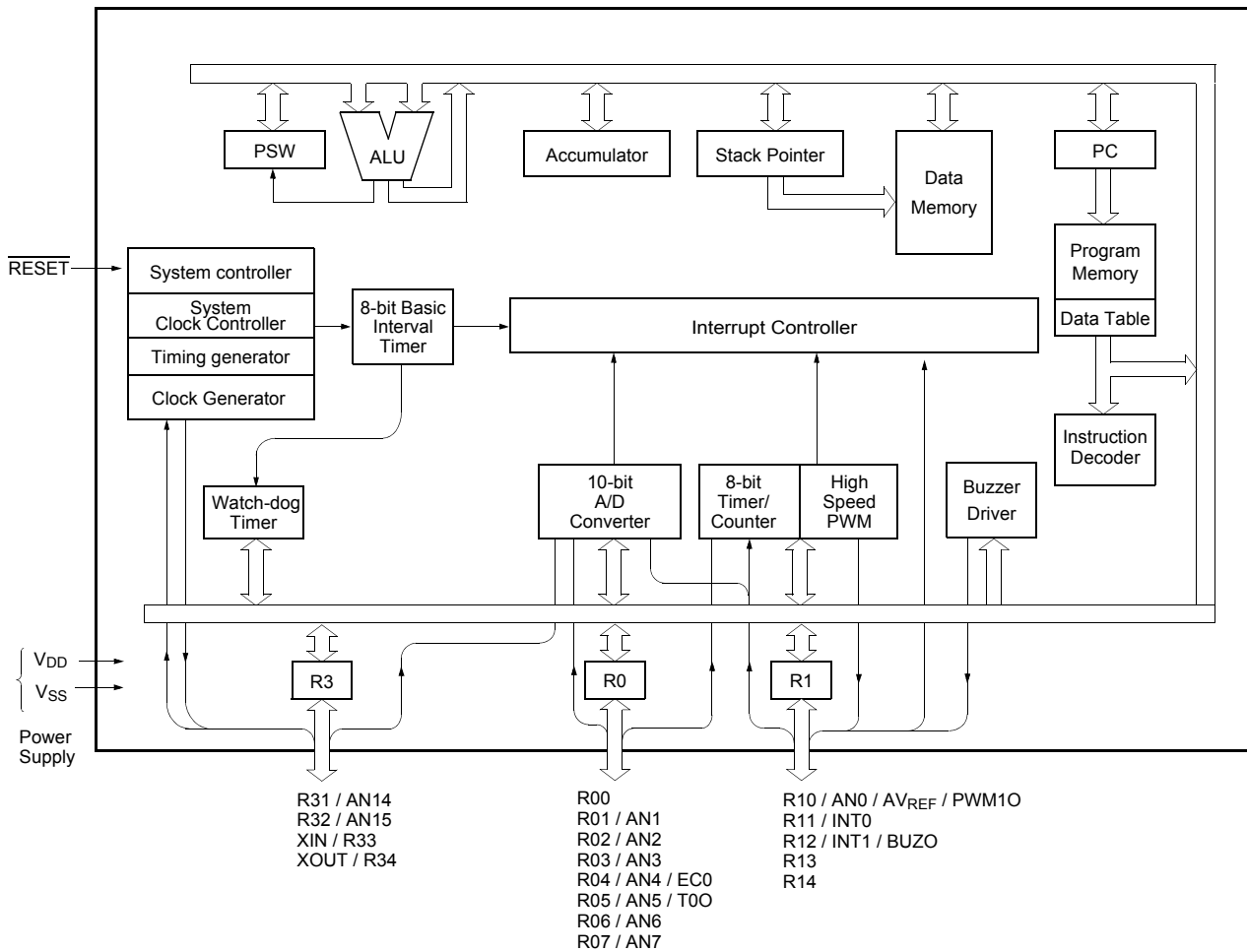
Device name	ROM Size	RAM size	Package
MC80F0604B	4K bytes FLASH	256 bytes	20PDIP
MC80F0604D			20SOP
MC80F0604S			20SSOP
MC80F0504B			16PDIP
MC80F0504D			16SOP
MC80F0504M			16SOP(153 mil)
MC80F0504R			16TSSOP

Pb free package :

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

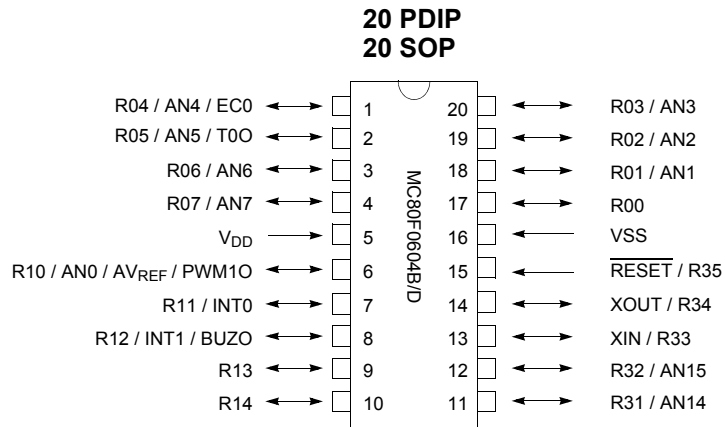
For example; MC80F0604B (Normal package), MC80F0604B P (Pb free package)

## 2. BLOCK DIAGRAM

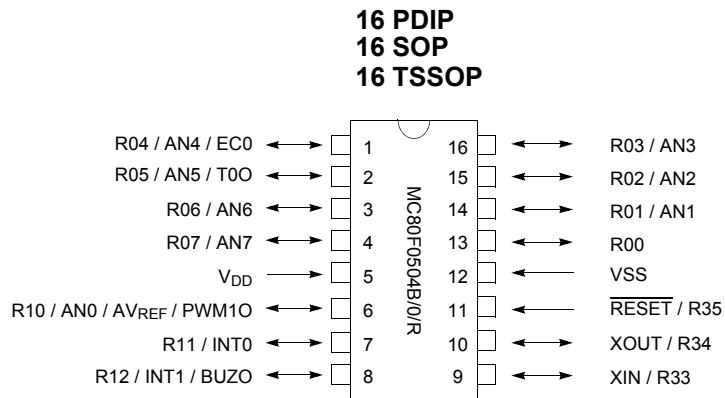


### 3. PIN ASSIGNMENT

#### MC80F0604B/0604D



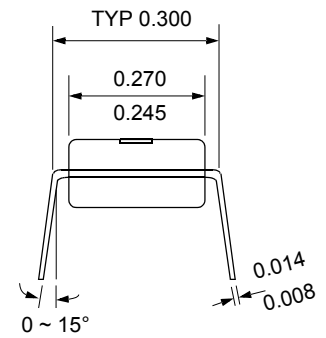
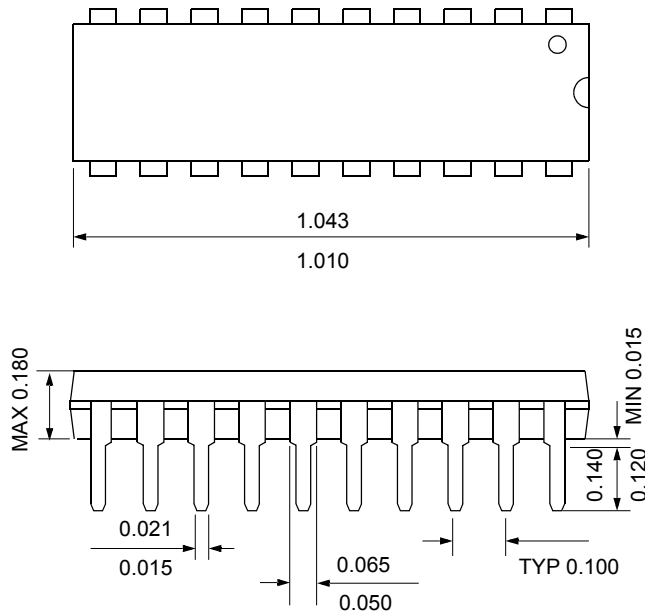
#### MC80F0504B/0504D/0504R



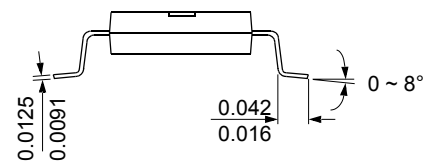
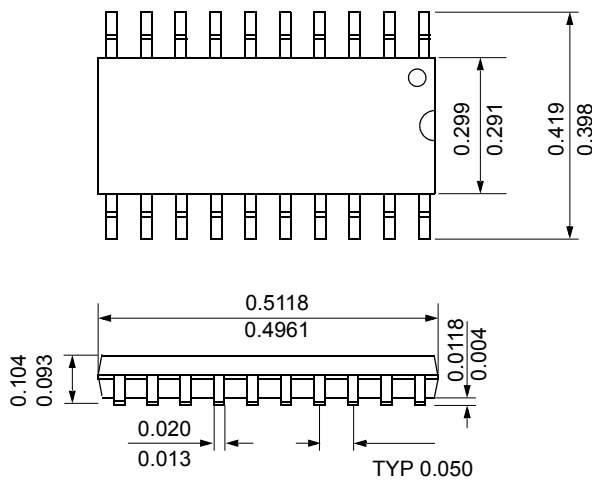
4. PACKAGE DRAWING

20 PDIP

unit: inch  
MAX  
MIN

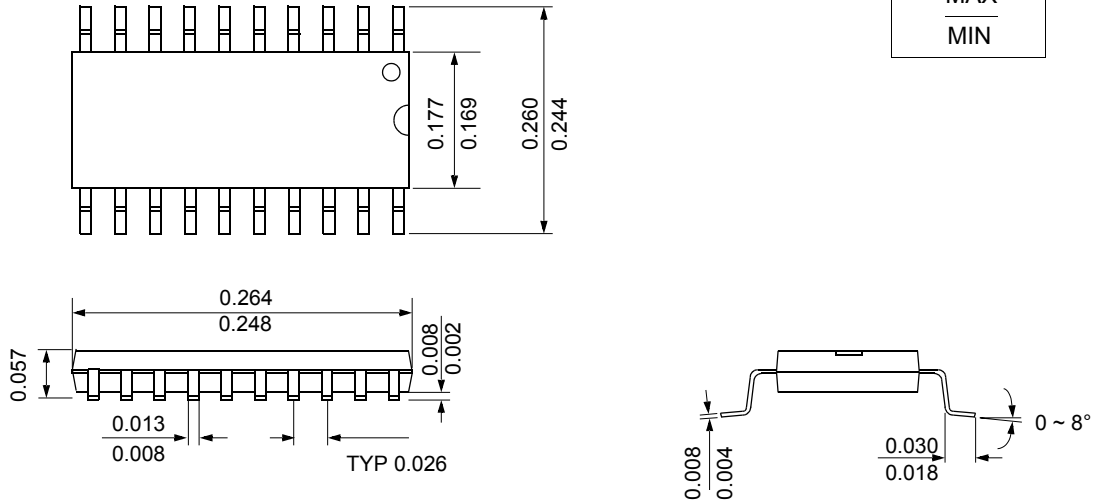


20 SOP



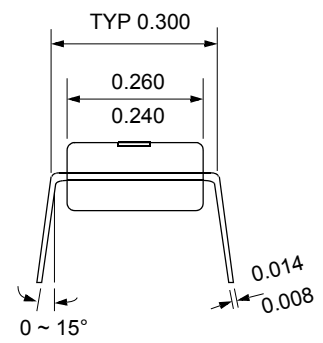
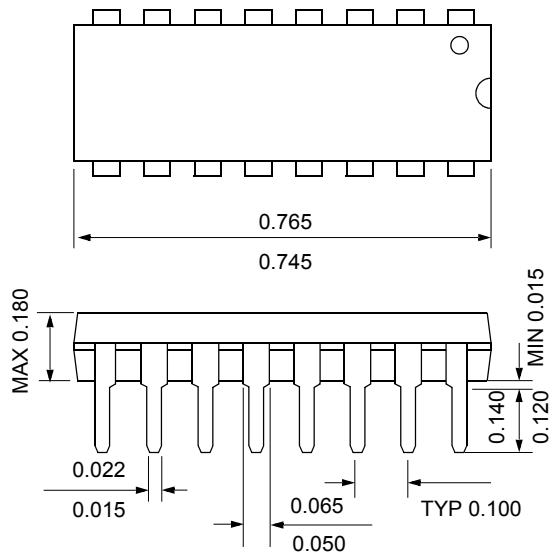
**20 SSOP**

unit: inch
MAX
MIN

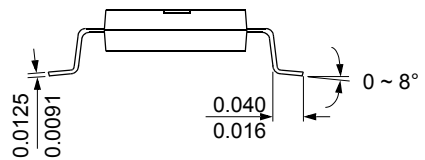
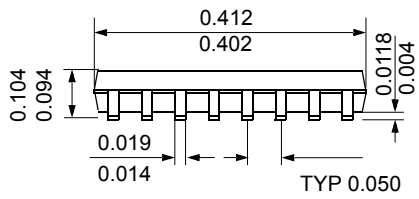
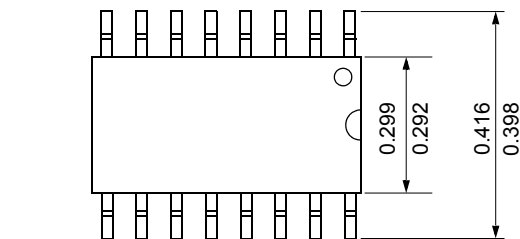


**16 PDIP**

unit: inch
MAX
MIN

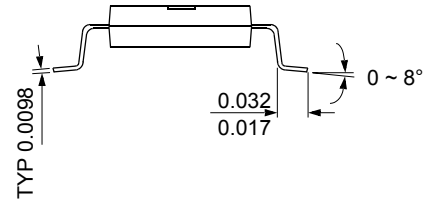
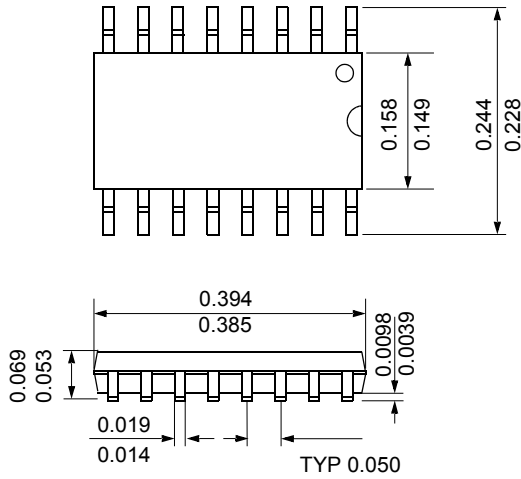


**16 SOP**

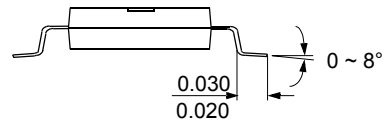
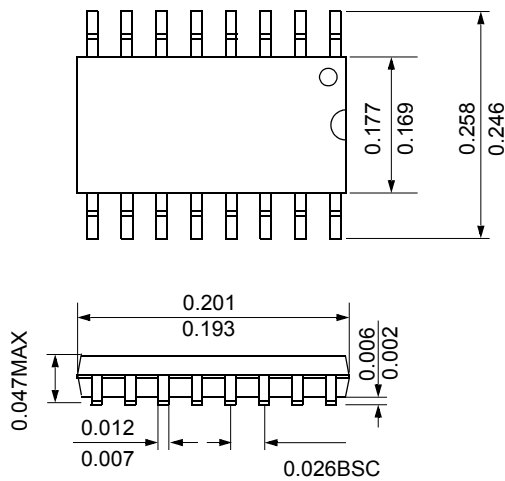


**16 SOP (153 mil)**

unit: inch  
MAX  
MIN



**16 TSSOP**



## 5. PIN FUNCTION

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

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

**RESET**: Reset the MCU.

**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.

**R00~R07**: R0 is an 8-bit, CMOS, bidirectional I/O port. RA pins can be used as outputs or inputs according to “1” or “0” written the their Port Direction Register(R0IO).

Port pin	Alternate function
R00	
R01	AN1 ( Analog Input Port 1 )
R02	AN2 ( Analog Input Port 2 )
R03	AN3 ( Analog Input Port 3 )
R04	AN4 ( Analog Input Port 4 )
	EC0 ( Event Counter Input Source 0 )
R05	AN5 ( Analog Input Port 5 )
	T00 (Timer0 Clock Output )
R06	AN6 ( Analog Input Port 6 )
R07	AN7 ( Analog Input Port 7 )

**Table 5-1 R0 Port**

In addition, R0 serves the functions of the various special features in Table 5-1 .

**R10~R14**: R1 is a 5-bit, CMOS, bidirectional I/O port. R1 pins can be used as outputs or inputs according to “1” or “0” written the their Port Direction Register (R1IO).

R1 serves the functions of the various following special features in Table 5-2

Port pin	Alternate function
R10	AN0 ( Analog Input Port 0 ) AVref ( External Analog Reference Pin ) PWM1O ( PWM1 Output )
R11	INT0 ( External Interrupt Input Port 0 )
R12	INT1 ( External Interrupt Input Port 1 ) BUZ ( Buzzer Driving Output Port )
R13	
R14	

**Table 5-2 R1 Port**

**R31~R35**: R3 is a 5-bit, CMOS, bidirectional I/O port. R3 pins can be used as outputs or inputs according to “1” or “0” written the their Port Direction Register (R3IO). In R3 pins, R35 pin can be used as input port only.

R3 serves the functions of the following special features in Table 5-3 .

Port pin	Alternate function
R31	AN14 ( Analog Input Port 14 )
R32	AN15 ( Analog Input Port 15 )
R33	X <sub>IN</sub> ( Oscillation Input )
R34	X <sub>OUT</sub> ( Oscillation Output )
R35	RESET ( Reset input port )

**Table 5-3 R3 Port**

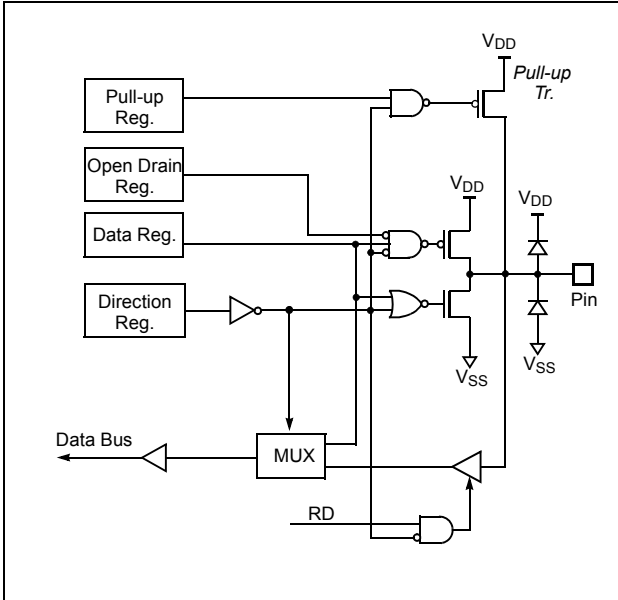


PIN NAME	Pin No. (20PDIP)	In/Out	Function	
V <sub>DD</sub>	5	-	Supply voltage	
V <sub>SS</sub>	16	-	Circuit ground	
RESET (R35)	15	I (I)	Reset signal input	Input only port
X <sub>IN</sub> (R33)	13	I (I/O)	Oscillation Input	Normal I/O Port
X <sub>OUT</sub> (R34)	14	O (I/O)	Oscillation Output	Normal I/O Port
R00	17	I/O	Normal I/O Ports	-
R01 (AN1)	18	I/O (Input)		Analog Input Port 1
R02 (AN2)	19	I/O (Input)		Analog Input Port 2
R03 (AN3)	20	I/O (Input)		Analog Input Port 3
R04 (AN4 / EC0)	1	I/O (Input/Input/Input)		Analog Input Port 4 / Event Counter Input 0
R05 (AN5 / T00)	2	I/O (Input/Output)		Analog Input Port 5 / Timer0 Output
R06 (AN6)	3	I/O (Input)		Analog Input Port 6
R07 (AN7)	4	I/O (Input)		Analog Input Port 7
R10 (AN0 / AV <sub>REF</sub> / PWM10)	6	I/O (Input/Input/Output)		Analog Input Port 0 / Analog Reference / PWM 1 output
R11 (INT0)	7	I/O (Input)		External Interrupt Input 0
R12 (INT1 / BUZO)	8	I/O (Input/Output)		External Interrupt Input 1 / Buzzer Driving Output
R13	9	I/O		-
R14	10	I/O		-
R31 (AN14)	11	I/O (Input)		Analog Input Port 14
R32 (AN15)	12	I/O (Input)		Analog Input Port 15

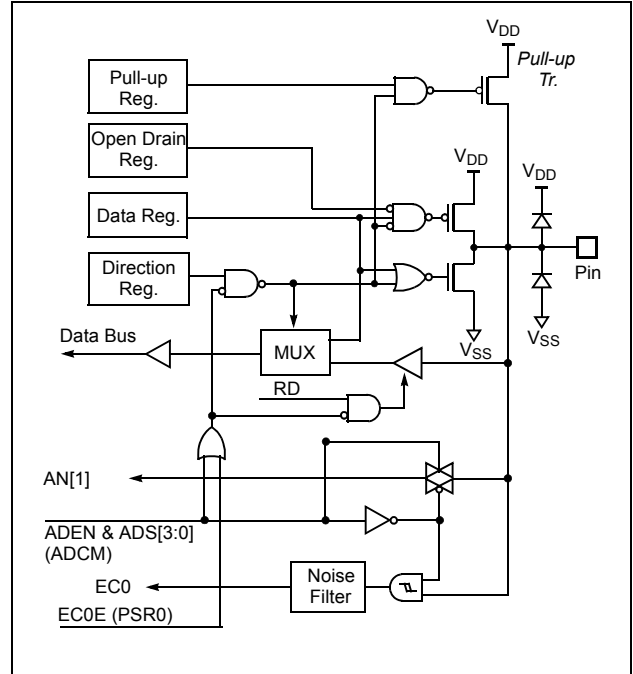
**Table 5-4 Pin Description**

## 6. PORT STRUCTURES

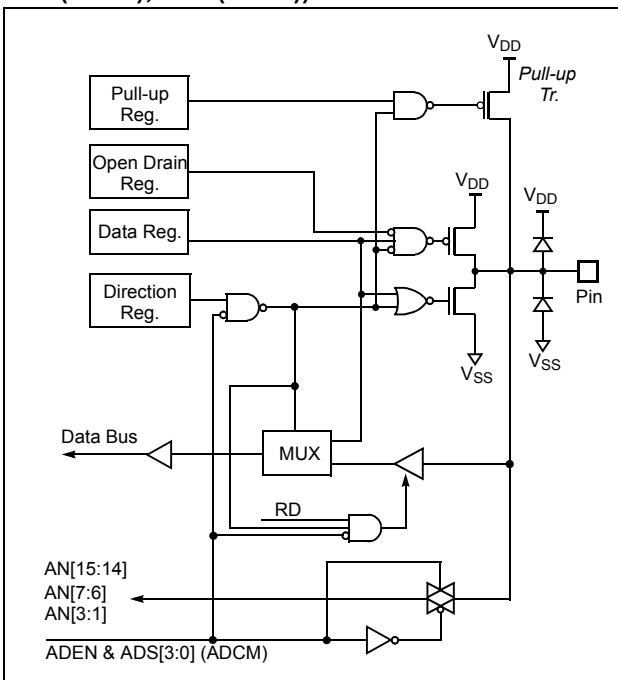
**R00, R13, R14**



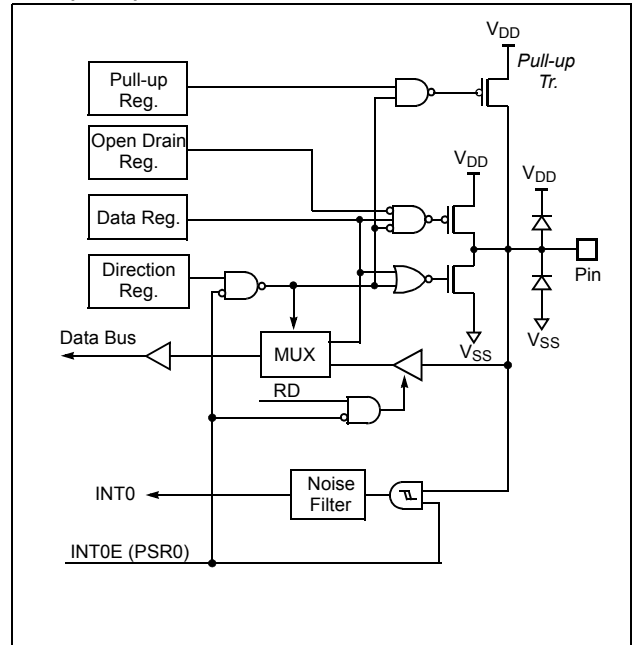
**R04 (AN4 / EC0)**



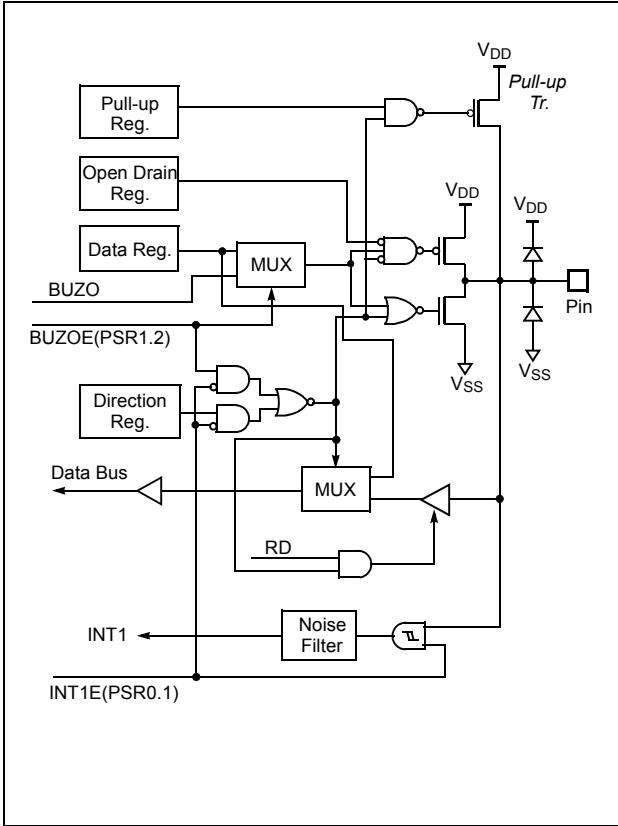
**R01(AN1)~R03(AN3), R06(AN6), R07(AN7),  
R31 (AN14), R32 (AN15))**



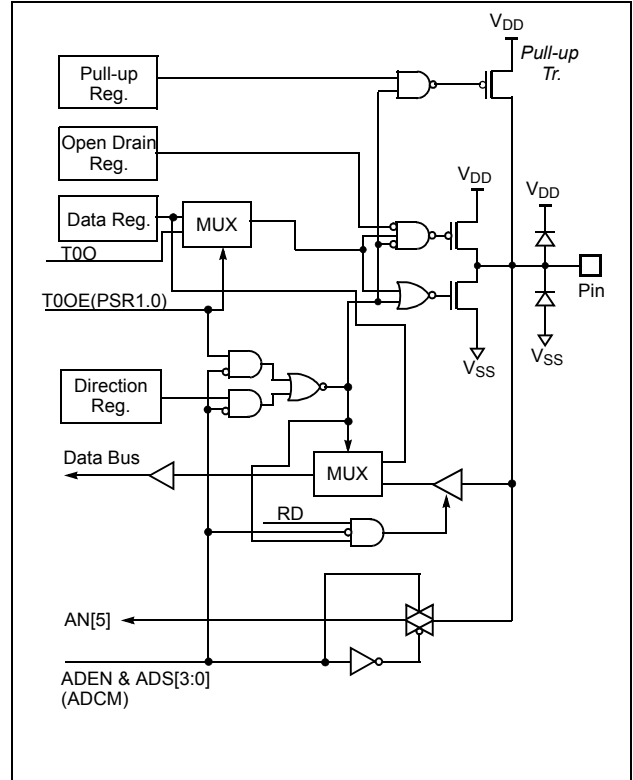
**R11 (INT0)**



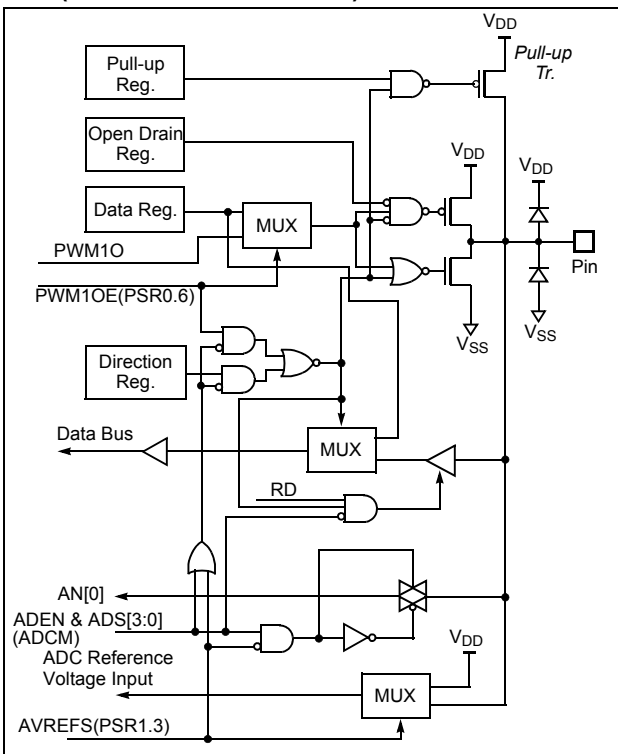
**R12 (INT1 / BUZO)**



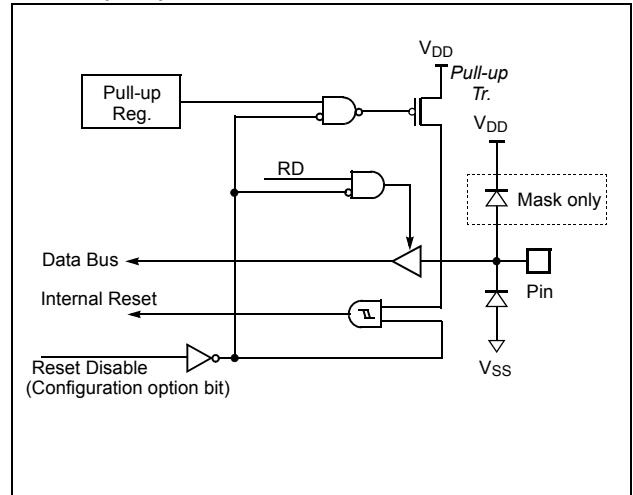
**R05 (AN5 / T00)**



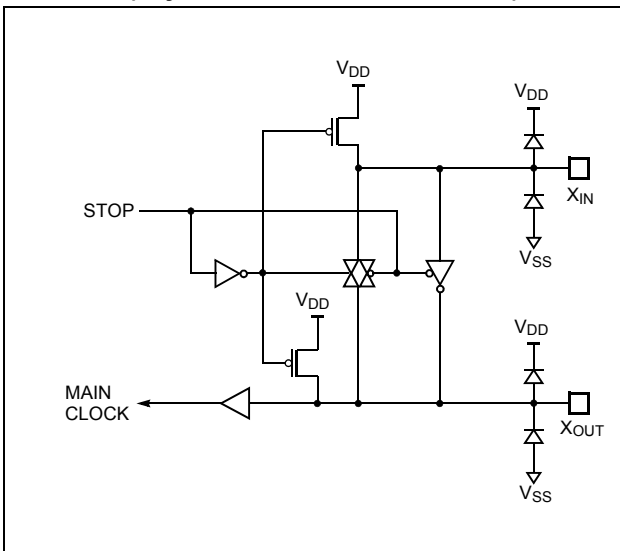
**R10 (AN0 / AVREF / PWM10)**



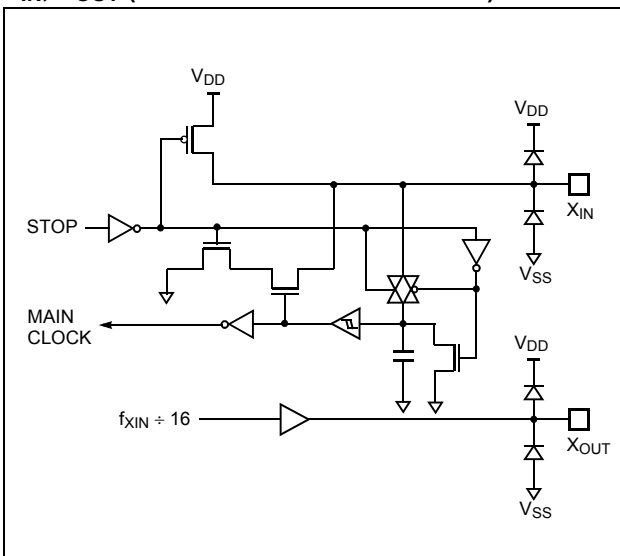
**RESET(R35)**



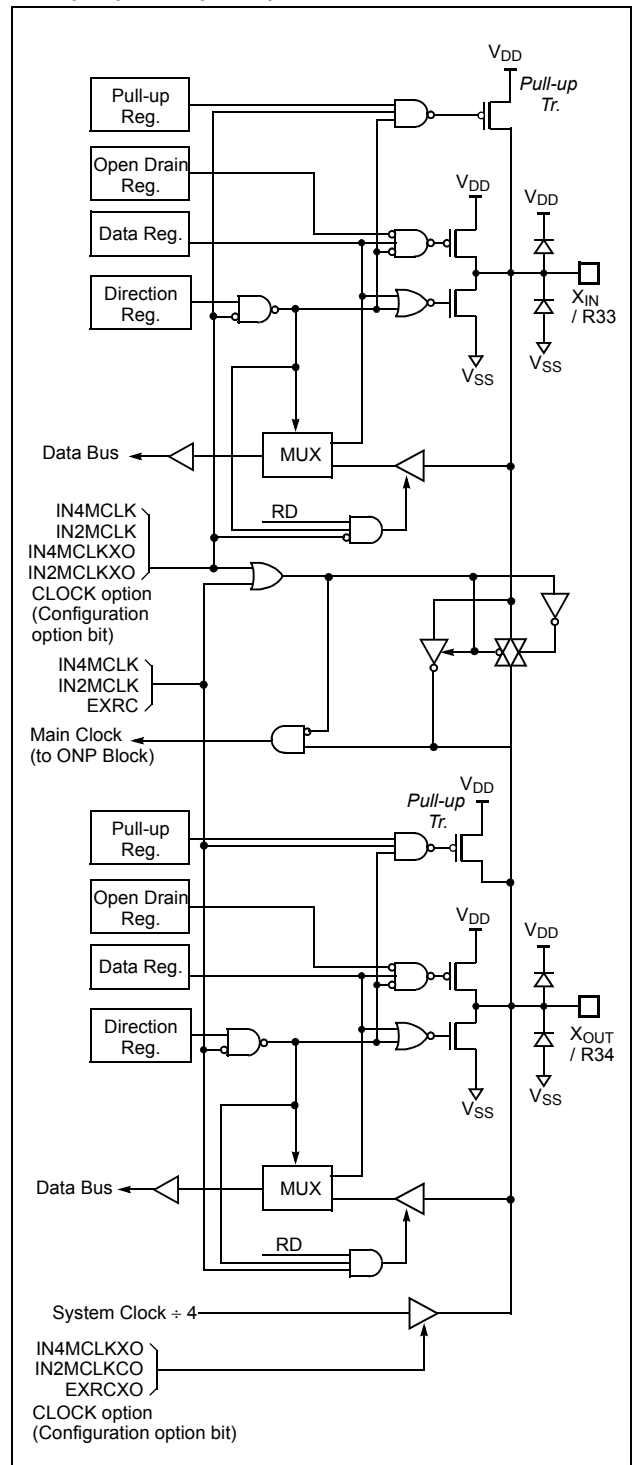
**X<sub>IN</sub>, X<sub>OUT</sub> (Crystal or Ceramic Resonator)**



**X<sub>IN</sub>, X<sub>OUT</sub> (External RC or R oscillation)**



**R33 (X<sub>IN</sub>), R34 (X<sub>OUT</sub>)**



## 7. ELECTRICAL CHARACTERISTICS

### 7.1 Absolute Maximum Ratings

Supply voltage .....	-0.3 to +6.5 V	.....	10 mA
Storage Temperature .....	-65 to +150 °C	Maximum current ( $\Sigma I_{OL}$ ) .....	160 mA
Voltage on any pin with respect to Ground ( $V_{SS}$ ) .....	-0.3 to $V_{DD}+0.3V$	Maximum current ( $\Sigma I_{OH}$ ) .....	80 mA
Maximum current out of $V_{SS}$ pin .....	200 mA	<hr/>	
Maximum current into $V_{DD}$ pin .....	100 mA	<b>Note:</b> 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.	
Maximum current sunk by ( $I_{OL}$ per I/O Pin) .....	20 mA	<hr/>	
Maximum output current sourced by ( $I_{OH}$ per I/O Pin)			

### 7.2 Recommended Operating Conditions

Parameter	Symbol	Condition	Min.	Max.	Unit
Supply Voltage	$V_{DD}$	$f_{XIN}=1\sim 12MHz$ $f_{XIN}=1\sim 8MHz$ $f_{XIN}=1\sim 8MHz$	4.5 2.7 2.2	5.5 5.5 5.5	V
Operating Frequency	$f_{XIN}$	$V_{DD}=4.5\sim 5.5V$ $V_{DD}=2.7\sim 5.5V$ $V_{DD}=2.2\sim 5.5V$	1 1 1	12 8 4	MHz
Operating Temperature	$T_{OPR}$	$V_{DD}=2.2\sim 5.5V$	-40	85	°C

### 7.3 A/D Converter Characteristics

( $T_a=-40\sim 85^\circ C$ ,  $V_{SS}=0V$ ,  $V_{DD}=2.7\sim 5.5V$  @ $f_{XIN}=8MHz$ )

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Resolution		-	-	10	-	BIT
Overall Accuracy	$CAIN$	-	-	-	$\pm 3$	LSB
Non Linearity Error	$NLE$	$V_{DD} = AV_{REF} = 5V$ CPU Clock = 4MHz $V_{SS} = 0V$	-	-	$\pm 3$	LSB
Differential Non Linearity Error	$DLE$		-	-	$\pm 3$	LSB
Zero Offset Error	$NZOE$		-	$\pm 1$	$\pm 3$	LSB
Full Scale Error	$NFSE$		-	$\pm 0.5$	$\pm 3$	LSB
Conversion Time	$T_{CONV}$	-	13	-	-	$\mu S$
Analog Input Voltage Range	$V_{AN}$	-	$V_{SS}$	-	$V_{DD}(AV_{REF})$	V
Analog Reference Voltage	$AV_{REF}$	-	2.7	-	$V_{DD}$	V
Analog Input Impedance	$R_{AIN}$	$V_{DD} = AV_{REF} = 5V$	5	100	-	$M\Omega$
Analog Block Current	$I_{AVDD}$	$V_{DD} = AV_{REF} = 5V$	-	1	3	mA
		$V_{DD} = AV_{REF} = 3V$	-	0.5	1.5	
		$V_{DD} = AV_{REF} = 5V$ power down mode	-	100	500	nA

## 7.4 DC Electrical Characteristics

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

Parameter	Symbol	Pin	Condition	Specifications			Unit
				Min.	Typ.	Max.	
Input High Voltage	$V_{IH1}$	$X_{IN}, \overline{\text{RESET}}$		$0.8 V_{DD}$	-	$V_{DD}$	V
	$V_{IH2}$	Hysteresis Input <sup>1</sup>		$0.8 V_{DD}$	-	$V_{DD}$	
	$V_{IH3}$	Normal Input		$0.7 V_{DD}$	-	$V_{DD}$	
Input Low Voltage	$V_{IL1}$	$X_{IN}, \overline{\text{RESET}}$		0	-	$0.2 V_{DD}$	V
	$V_{IL2}$	Hysteresis Input <sup>1</sup>		0	-	$0.2 V_{DD}$	
	$V_{IL3}$	Normal Input		0	-	$0.3 V_{DD}$	
Output High Voltage	$V_{OH}$	All Output Port	$V_{DD} = 5\text{V}$ , $I_{OH} = -5\text{mA}$	$V_{DD} - 1$	-	-	V
Output Low Voltage	$V_{OL}$	All Output Port	$V_{DD} = 5\text{V}$ , $I_{OL} = 10\text{mA}$	-	-	1	V
Input Pull-up Current	$I_P$	Normal Input	$V_{DD} = 5\text{V}$	-60	-	-150	$\mu\text{A}$
Input High Leakage Current	$I_{IH1}$	All Pins (except $X_{IN}$ )	$V_{DD} = 5\text{V}$	-	-	5	$\mu\text{A}$
	$I_{IH2}$	$X_{IN}$	$V_{DD} = 5\text{V}$	-	12	20	$\mu\text{A}$
Input Low Leakage Current	$I_{IL1}$	All Pins (except $X_{IN}$ )	$V_{DD} = 5\text{V}$	-5	-	-	$\mu\text{A}$
	$I_{IL2}$	$X_{IN}$	$V_{DD} = 5\text{V}$	-20	-12	-	$\mu\text{A}$
Hysteresis	$ V_T $	Hysteresis Input <sup>1</sup>	$V_{DD} = 5\text{V}$	0.5	-	-	V
PFD Voltage	$V_{PFD}$	$V_{DD}$		2.0	-	3.0	V
POR Voltage	$V_{POR}$	$V_{DD}$		2.1	2.6	3.0	V
POR Start Voltage <sup>2</sup>	$V_{START}$	$V_{DD}$		0		1.9	V
POR Rising Time <sup>2</sup>	$T_{POR}$	$V_{DD}$				40	ms/V
$V_{DD}$ Rising Time <sup>2</sup>	$T_{VDD}$	$V_{DD}$		-	-	40	ms/V
Internal RC WDT Period	$T_{RCWDT}$	$X_{OUT}$	$V_{DD} = 5.5\text{V}$	36	-	90	$\mu\text{s}$
Operating Current	$I_{DD}$	$V_{DD}$	$V_{DD} = 5.5\text{V}$ , $f_{XIN} = 12\text{MHz}$	-	7	15	mA
Sleep Mode Current	$I_{SLEEP}$	$V_{DD}$	$V_{DD} = 5.5\text{V}$ , $f_{XIN} = 12\text{MHz}$	-	2	4.5	mA
RCWDT Mode Current at STOP Mode	$I_{RCWDT}$	$V_{DD}$	$V_{DD} = 5.5\text{V}$ , $f_{XIN} = 12\text{MHz}$	-	20	55	$\mu\text{A}$
Stop Mode Current	$I_{STOP}$	$V_{DD}$	$V_{DD} = 5.5\text{V}$ , $f_{XIN} = 12\text{MHz}$	-	1	5	$\mu\text{A}$
Internal Oscillation Frequency( 4MHz )	$f_{IN\_CLK}$	$X_{OUT}$	$V_{DD} = 5\text{V}$ , Temp = $25^\circ\text{C}$	3.75	4.25	4.65	MHz
$\overline{\text{RESET}}$ Input Noise Cancel Time	$T_{RST\_NC}$	$\overline{\text{RESET}}$	$V_{DD} = 5\text{V}$	1.5		1.8	$\mu\text{s}$
External RC Oscillator Frequency	$f_{RC\_OSC}$	$f_{XOUT} = f_{RC\_OSC} \div 4$	$V_{DD} = 5.5\text{V}$ $R = 30\text{k}\Omega$ , $C = 10\text{pF}$	0.5	1.5	2.5	MHz
	$f_{R\_OSC}$	$f_{XOUT} = f_{R\_OSC} \div 4$	$V_{DD} = 5.5\text{V}$ , $R = 30\text{k}\Omega$	1	2	3	MHz

1. Hysteresis Input: INT0(R11), INT1(R12), EC0(R04)

2. These parameters are presented for design guidance only and not tested or guaranteed.

### 7.5 AC Characteristics

( $T_A = -40 \sim 85^\circ\text{C}$ ,  $V_{DD} = 5\text{V} \pm 10\%$ ,  $V_{SS} = 0\text{V}$ )

Parameter	Symbol	Pins	Specifications			Unit
			Min.	Typ.	Max.	
Operating Frequency	$f_{XIN}$	$X_{IN}$	1	-	12	MHz
System Clock Cycle Time	$t_{SYS}$	-	166	-	5000	nS
Oscillation Stabilizing Time (4MHz)	$t_{ST}$	$X_{IN}$ , $X_{OUT}$	-	-	20	mS
External Clock Pulse Width	$t_{CPW}$	$X_{IN}$	35	-	-	nS
External Clock Transition Time	$t_{RCP}, t_{FCP}$	$X_{IN}$	-	-	20	nS
Interrupt Pulse Width	$t_{IW}$	$\overline{INT0}$ , $\overline{INT1}$	2	-	-	$t_{SYS}$
RESET Input Width	$t_{RST}$	$\overline{RESET}$	8	-	-	$t_{SYS}$
Event Counter Input Pulse Width	$t_{ECW}$	EC0	2	-	-	$t_{SYS}$
Event Counter Transition Time	$t_{REC}, t_{FEC}$	EC0	-	-	20	nS

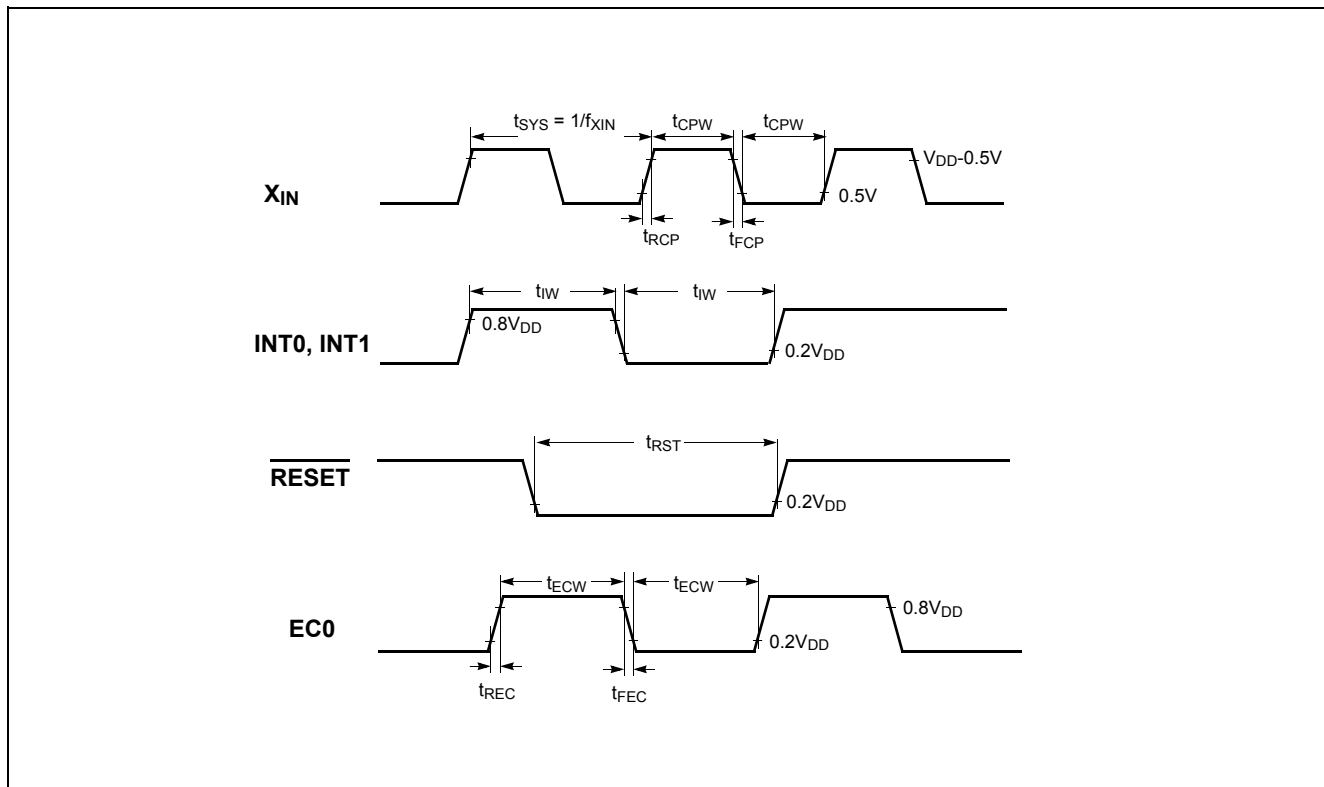


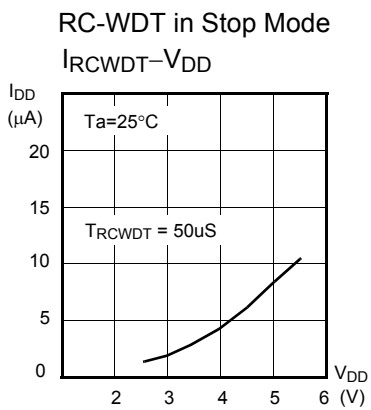
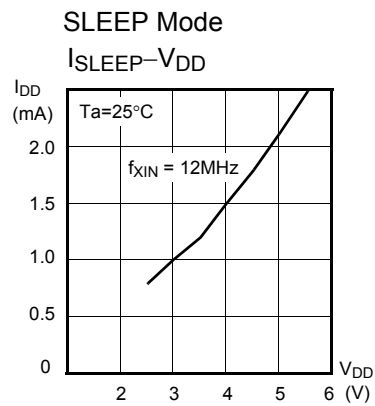
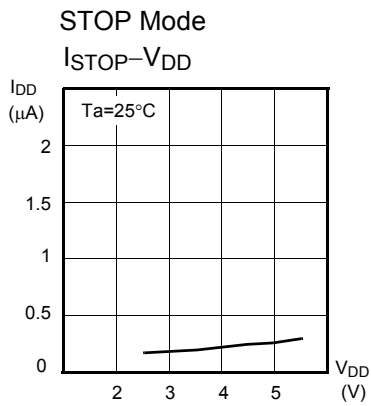
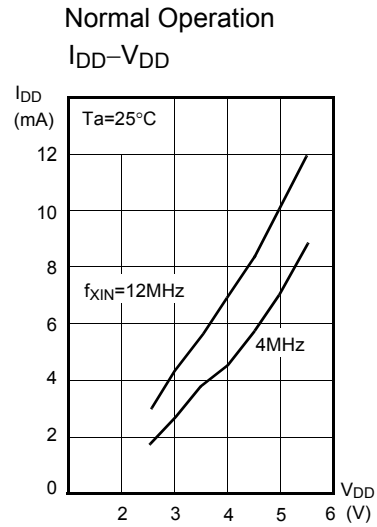
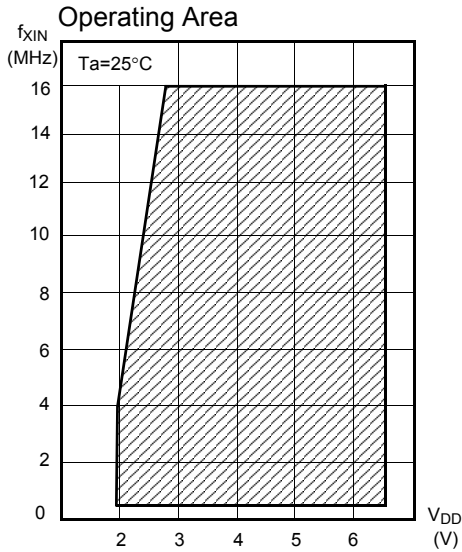
Figure 7-1 Timing Chart

### 7.6 Typical Characteristics

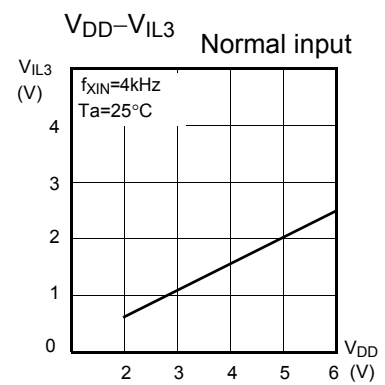
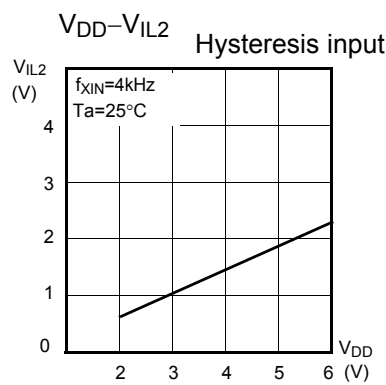
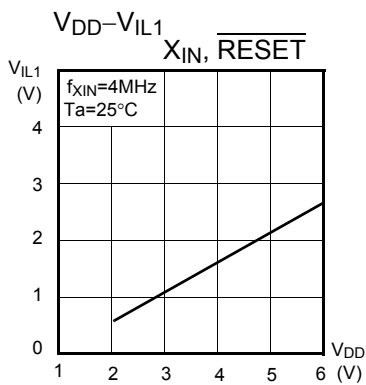
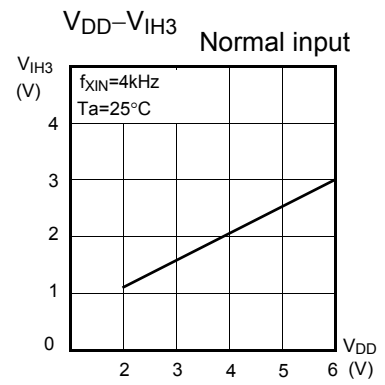
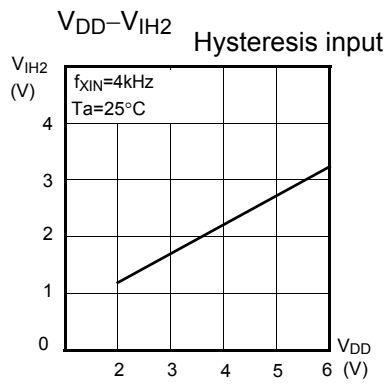
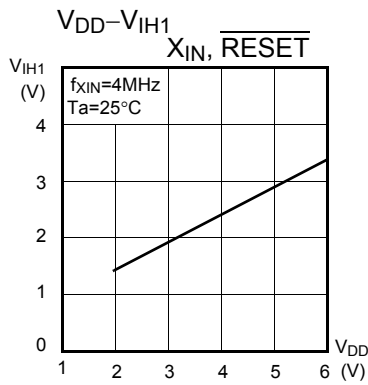
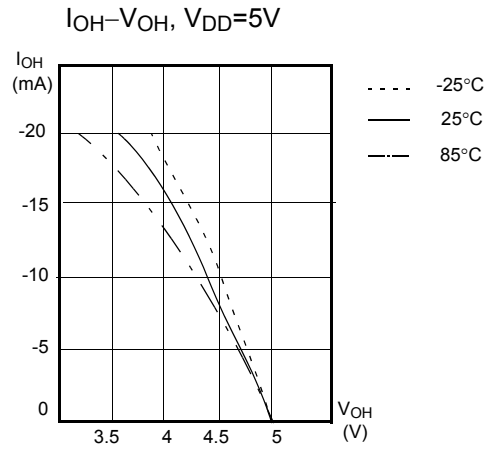
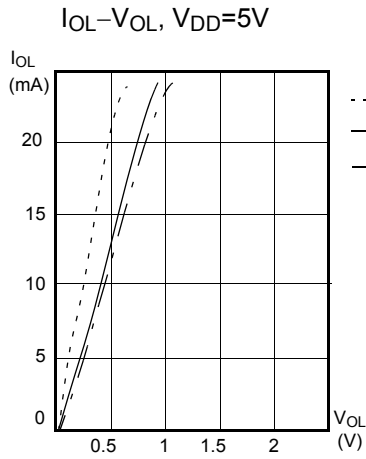
These graphs and tables provided in this section are for design guidance only and are not tested or guaranteed.

**In some graphs or tables the data presented are outside specified operating range (e.g. outside specified  $V_{DD}$  range). This is for information only and devices are guaranteed to operate properly only within the specified range.**

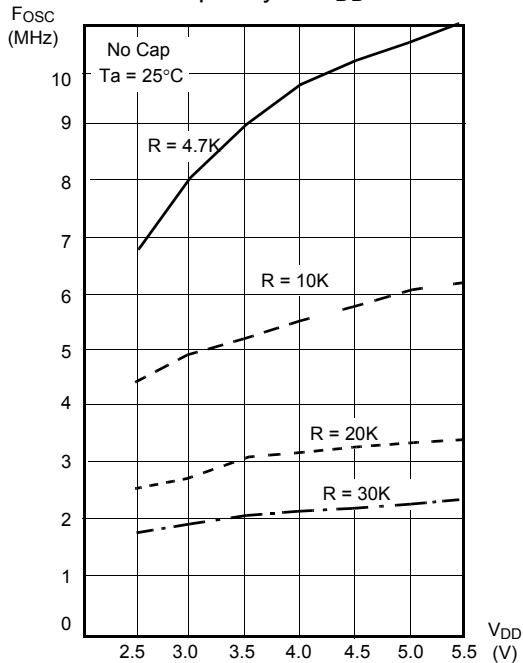
The data presented in this section is a statistical summary of data collected on units from different lots over a period of time. “Typical” represents the mean of the distribution while “max” or “min” represents (mean +  $3\sigma$ ) and (mean -  $3\sigma$ ) respectively where  $\sigma$  is standard deviation



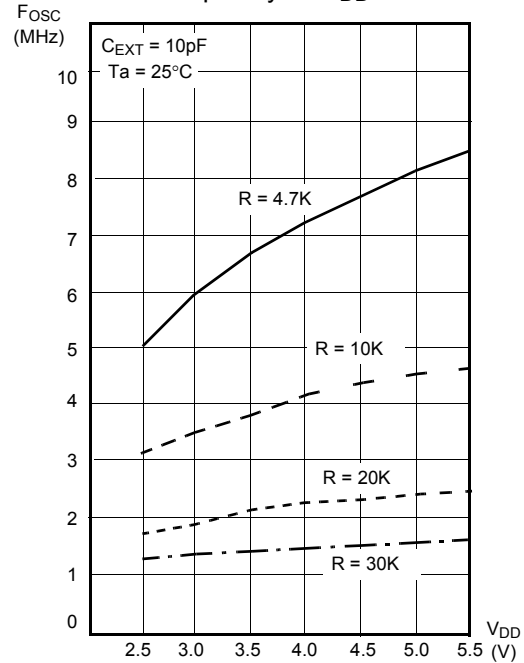




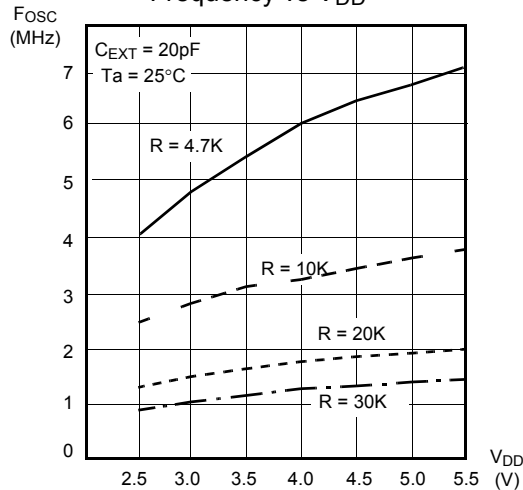
Typical RC Oscillator Frequency vs  $V_{DD}$



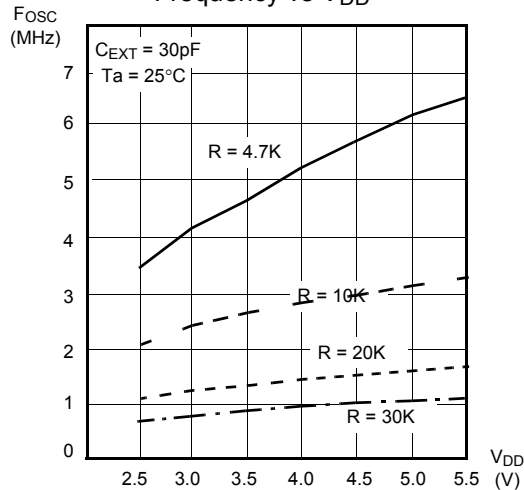
Typical RC Oscillator Frequency vs  $V_{DD}$



Typical RC Oscillator Frequency vs  $V_{DD}$



Typical RC Oscillator Frequency vs  $V_{DD}$



**Note:** The external RC oscillation frequencies shown in above are provided for design guidance only and not tested or guaranteed. The user needs to take into account that the external RC oscillation frequencies generated by the same circuit design may be not the same. Because there are variations in the resistance and capacitance due to the tolerance of external R and C components. The parasitic capacitance difference due to the different wiring length and layout may change the external RC oscillation frequencies.

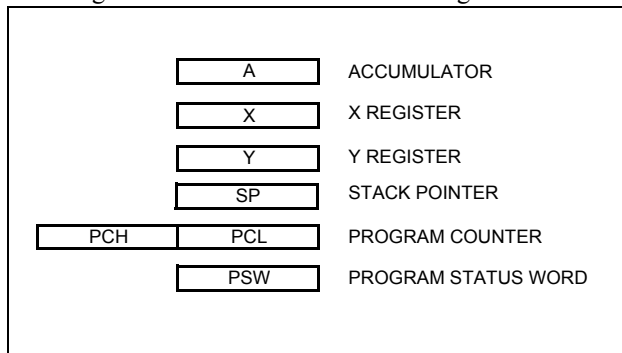
**Note:** There may be the difference between package types (PDIP, SOP, TSSOP). The user should modify the value of R and C components to get the proper frequency in exchanging MC80F0104/0204 to MC80F0504/0604 or one package type to another package type.

## 8. MEMORY ORGANIZATION

The MC80F0504/0604 has separate address spaces for Program memory and Data Memory. 4K bytes program memory can only be read, not written to.

### 8.1 Registers

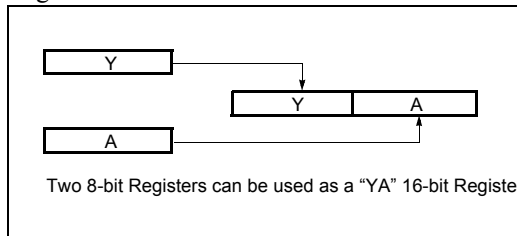
This device has six registers that are the Program Counter (PC), a Accumulator (A), two index registers (X, Y), the Stack Pointer (SP), and the Program Status Word (PSW). The Program Counter consists of 16-bit register.



**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**

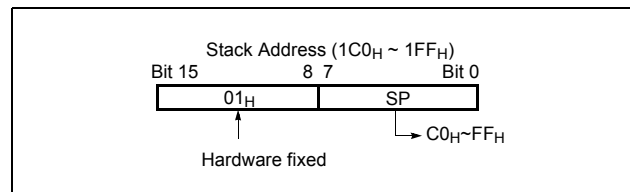
**X, Y Registers:** In the addressing mode which uses these index registers, the register contents are added to the specified address, which becomes the actual address. These modes are extremely effective for referencing subroutine tables and memory tables. The index registers also have increment, decrement, 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).

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 1C0H to 1FFH 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 "FFH" is used.



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

**Example:** To initialize the SP

```
LDX    #0FFH
TXSP                      ; SP ← FFH
```

**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 (PCH:0FFH, PCL:0FEH).

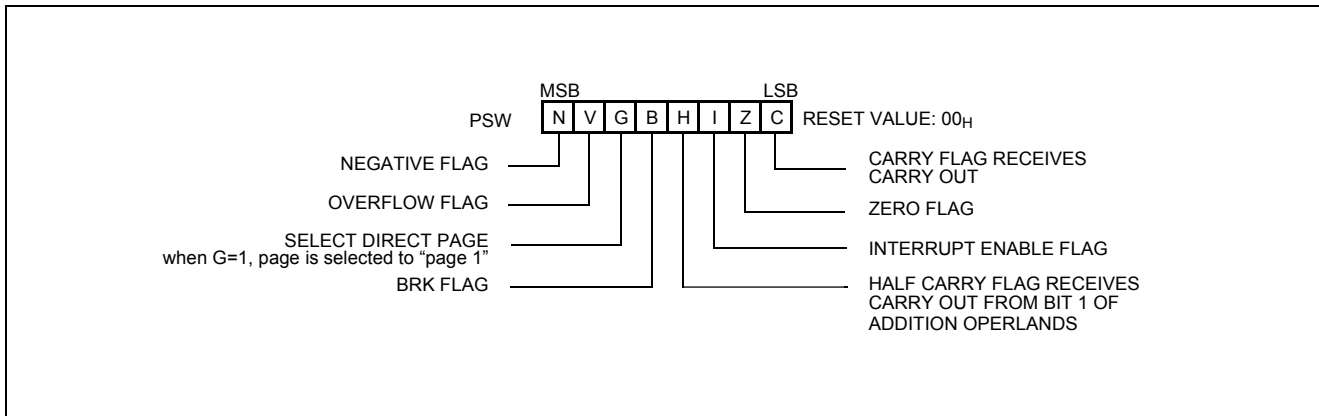
**Program Status Word:** The Program Status Word (PSW) contains several bits that reflect the current state of the CPU. The PSW is described in Figure 8-3 . It contains the Negative flag, the Overflow flag, the Break flag the Half Carry (for BCD operation), the Interrupt enable flag, the Zero flag, and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

**[Break flag B]**

This flag is set by software BRK instruction to distinguish BRK from T<sub>CALL</sub> instruction with the same vector address.

**[Direct page flag G]**

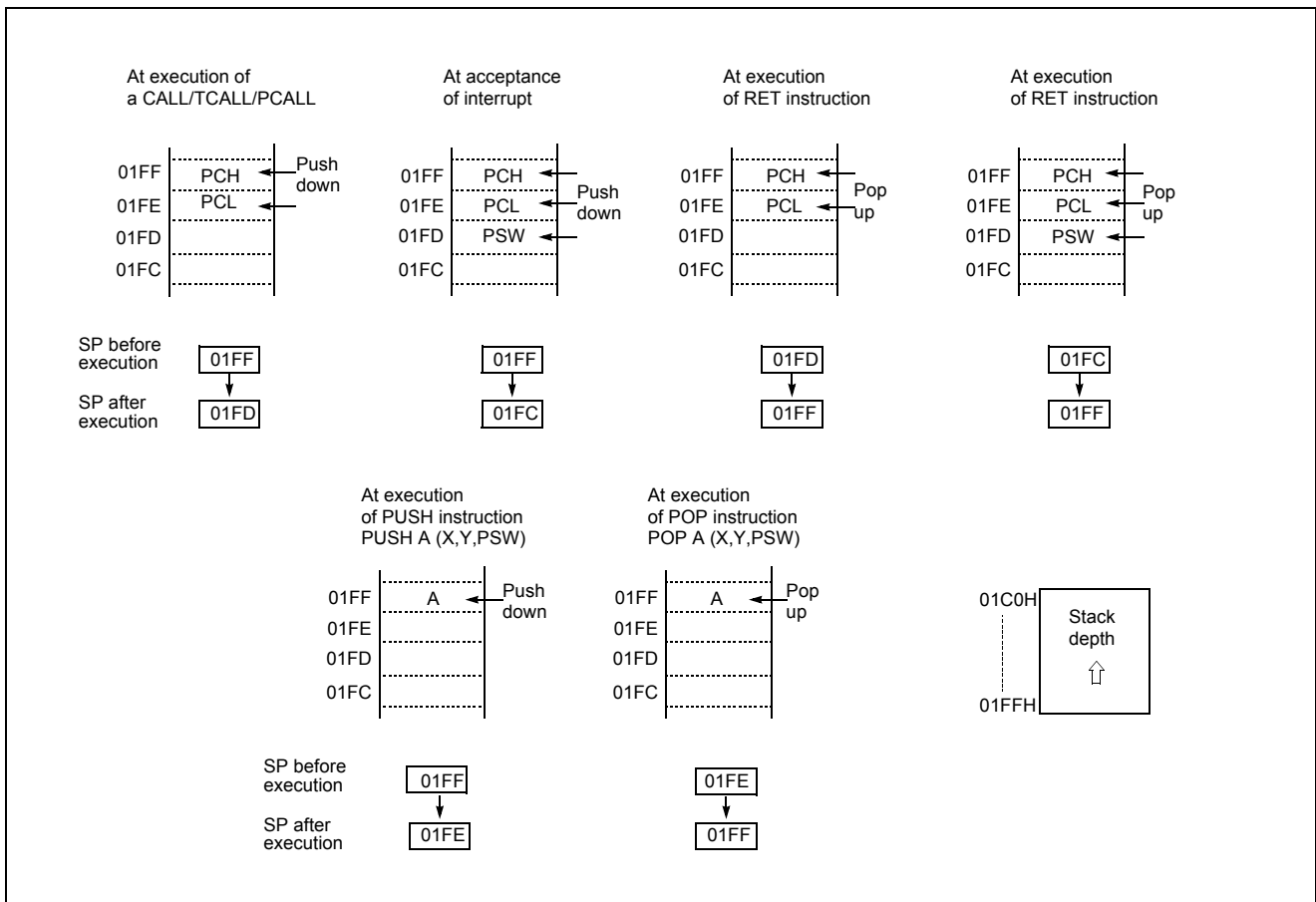
This flag assigns RAM page for direct addressing mode. In the direct addressing mode, addressing area is from zero page 00<sub>H</sub> to 0FF<sub>H</sub> when this flag is "0". If it is set to "1", addressing area is assigned 100<sub>H</sub> to 1FF<sub>H</sub>. It is set by SET<sub>G</sub> instruction and cleared by CLR<sub>G</sub>.

**[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(7F<sub>H</sub>) or -128(80<sub>H</sub>). The CLR<sub>V</sub> 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 4K 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

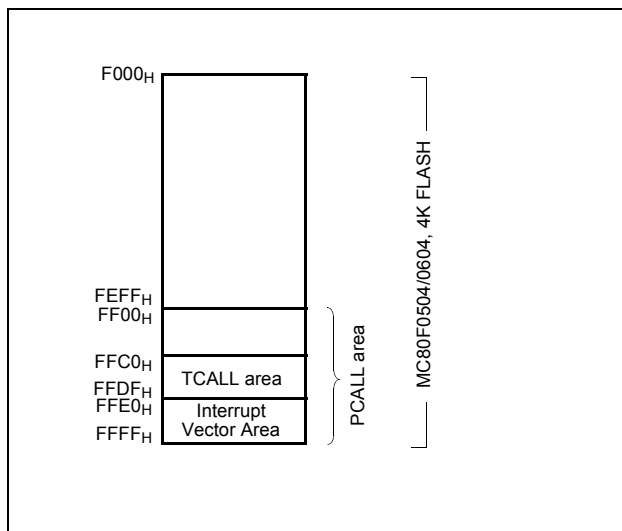


Figure 8-5 Program Memory Map

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

```

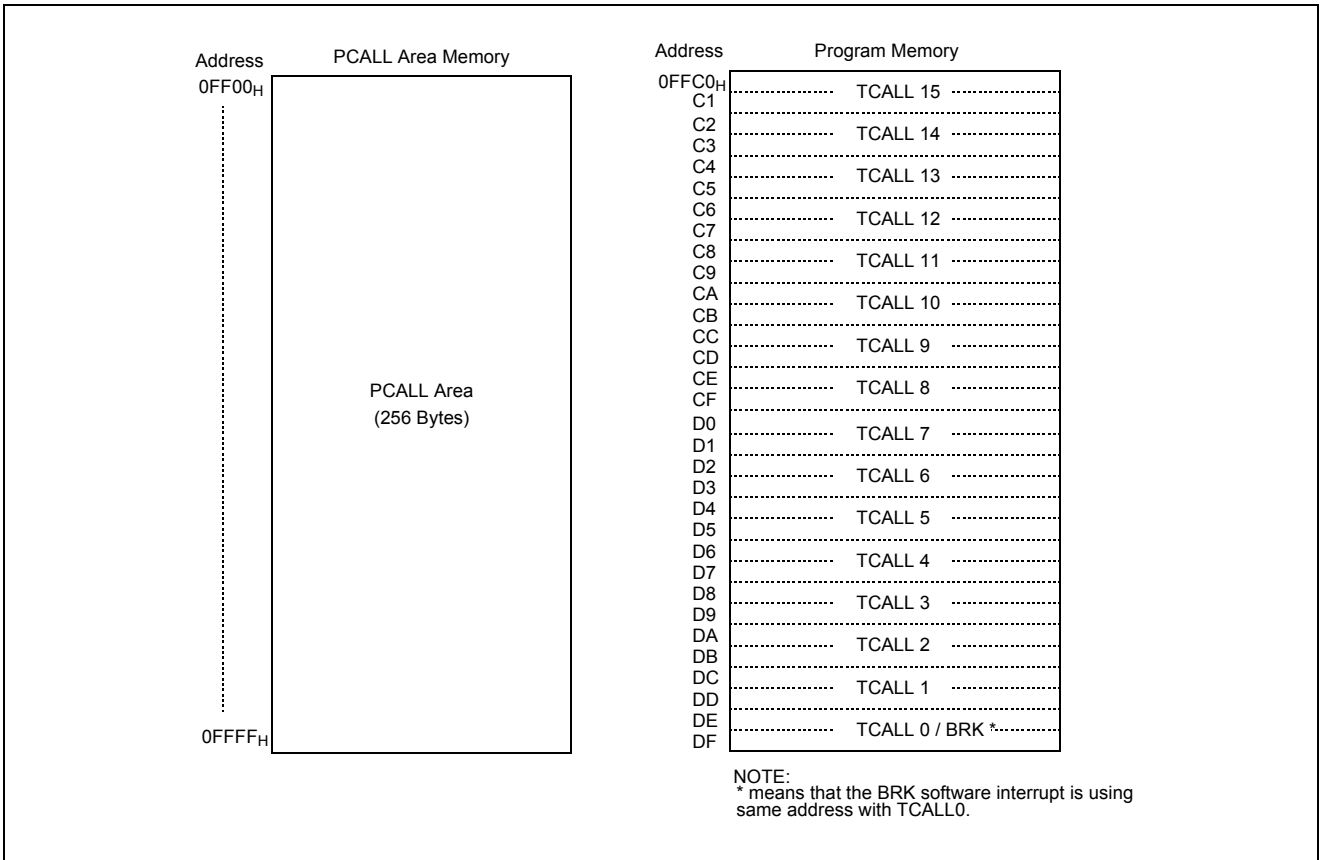
LDA    #5
TCALL  0FH      ; 1BYTE INSTRUCTION
           ; INSTEAD OF 3 BYTES
           ; NORMAL CALL
;
; TABLE CALL ROUTINE
;
FUNC_A: LDA    LRG0
        RET
;
FUNC_B: LDA    LRG1  ②
        RET          ①
;
; TABLE CALL ADD. AREA
;
           ORG    0FFC0H
           DW    FUNC_A
           DW    FUNC_B
           ; TCALL ADDRESS AREA
    
```

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 0FFFC<sub>H</sub>. The interrupt service locations spaces 2-byte interval: 0FFFA<sub>H</sub> and 0FFFB<sub>H</sub> for External Interrupt 1, 0FFFC<sub>H</sub> and 0FFFD<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
0FFE0H	Basic Interval Timer
E2	Watchdog Timer Interrupt
E4	A/D Converter
E6	-
E8	-
EA	-
EC	Timer/Counter 1 Interrupt
EE	Timer/Counter 0 Interrupt
F0	-
F2	-
F4	-
F6	-
F8	-
FA	External Interrupt 1
FC	External Interrupt 0
FE	RESET

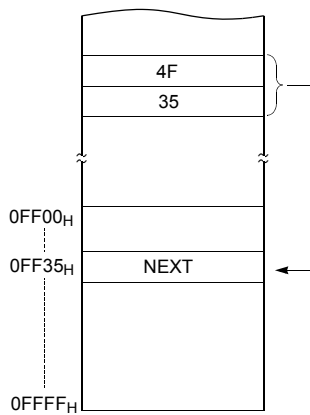
Figure 8-6 Interrupt Vector Area



**Figure 8-7 PCALL and TCALL Memory Area**

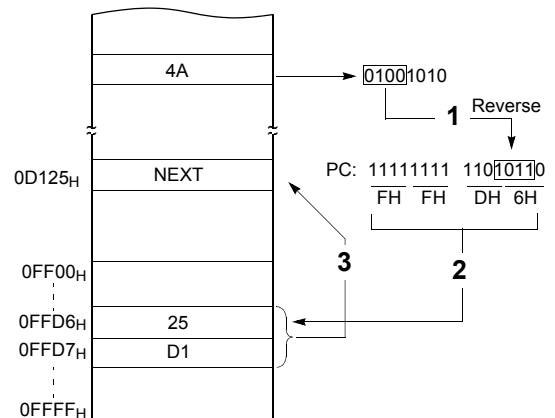
**PCALL → rel**

4F35 PCALL 35H



**TCALL → n**

4A TCALL 4



Example: The usage software example of Vector address for MC80F0604.

```

;Interrupt Vector Table
    ORG    0FFE0H
    DW    BIT_TIMER    ; BIT
    DW    WDT          ; WDT
    DW    ADC          ; AD Converter
    DW    Noticed      ;
    DW    Noticed      ;
    DW    Noticed      ;
    DW    TIMER1       ; Timer-1
    DW    TIMER0       ; Timer-0
    DW    Noticed      ;
    DW    Noticed      ;
    DW    Noticed      ;
    DW    Noticed      ;
    DW    Noticed      ;
    DW    INT1         ; Ext. Int.1
    DW    INT0         ; Ext. Int.0
    DW    RESET        ; Reset

    ORG    0F000H      ; 4K bytes ROM Start address
;*****
;          MAIN      PROGRAM      *
;*****
RESET:    DI          ;Disable All Interrupt
;RAM Clear Routine
    LDX    #0
RAM_Clear0:
    LDA    #0          ;Page0 RAM Clear(0000h ~ 00BFh)
    STA    {X}+
    CMPX   #0C0h
    BNE    RAM_Clear0
    LDM    RPR,#1     ;Page Select
    SETG

    LDX    #0C0h
RAM_Clear1:
    LDA    #0
    STA    {X}+
    CMPX   #00h
    BNE    RAM_Clear1

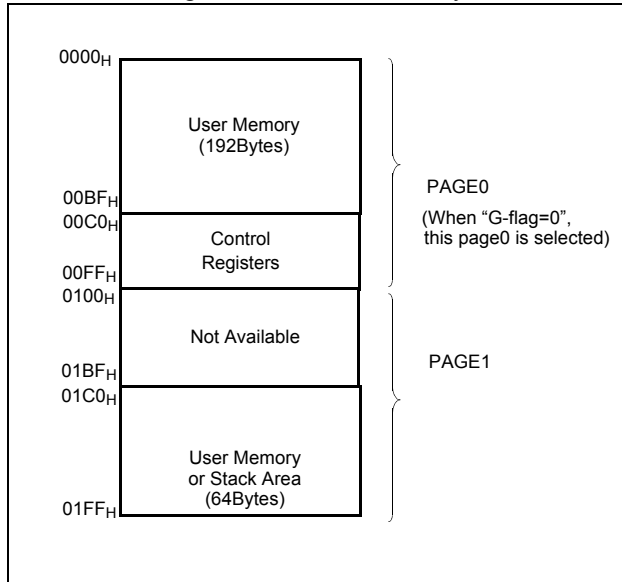
RAM_Clear_Finish:
    CLRG    ;Page0 Select
    LDX    #0FFh     ;Initial Stack Pointer
    TXSP
    :
    :
;Initialize IO
    LDM    R0, #0     ;Normal Port R0
    LDM    R0IO,#0FFH ;Normal Port R0 Direction
    :
    :

```



### 8.3 Data Memory

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



**Figure 8-8 Data Memory Map**

#### User Memory

The MC80F0504/0604 has  $256 \times 8$  bits for the user memory (RAM). RAM pages are selected by RPR (See Figure 8-9).

**Note:** After setting RPR(RAM Page Select Register), be sure to execute SETG instruction. When executing CLRG instruction, be selected PAGE0 regardless of RPR.

#### 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 0C0H 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, for example "LDM".

Example; To write at CKCTLR

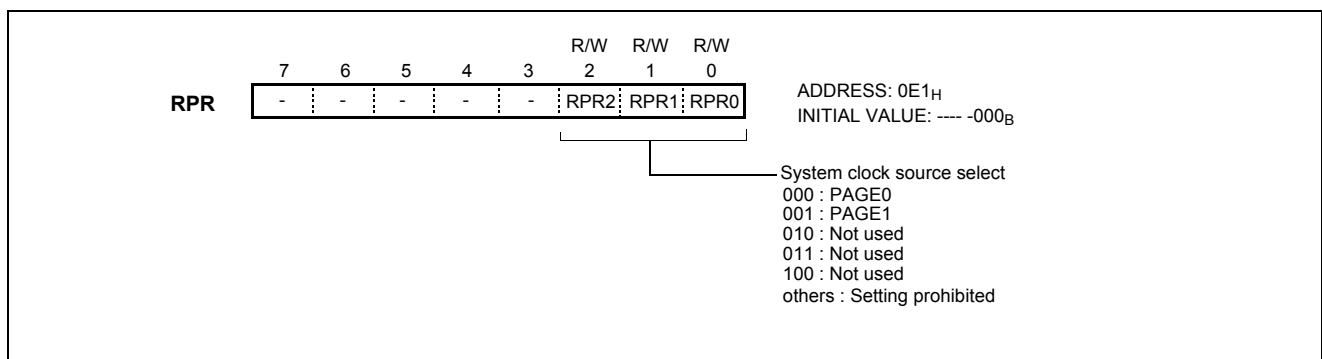
```
LDM CKCTLR, #0AH ;Divide ratio(+32)
```

#### 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 pointed (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.



**Figure 8-9 RPR(RAM Page Select Register)**

Address	Register Name	Symbol	R/W	Initial Value								Addressing Mode	
				7	6	5	4	3	2	1	0		
00C0	R0 port data register	R0	R/W	0	0	0	0	0	0	0	0	0	byte, bit <sup>1</sup>
00C1	R0 port I/O direction register	R0IO	W	0	0	0	0	0	0	0	0	0	byte <sup>2</sup>
00C2	R1 port data register	R1	R/W	-	-	-	0	0	0	0	0	0	byte, bit
00C3	R1 port I/O direction register	R1IO	W	-	-	-	0	0	0	0	0	0	byte
00C6	R3 port data register	R3	R/W	-	-	0	0	0	0	0	0	-	byte, bit
00C7	R3 port I/O direction register	R3IO	W	0	0	0	0	0	0	0	0	-	byte
00C8	Port 0 Open Drain Selection Register	R0OD	W	0	0	0	0	0	0	0	0	0	byte
00C9	Port 1 Open Drain Selection Register	R1OD	W	-	-	-	0	0	0	0	0	0	byte
00CB	Port 3 Open Drain Selection Register	R3OD	W	-	-	-	0	0	0	0	0	-	byte
00D0	Timer 0 mode control register	TM0	R/W	-	-	0	0	0	0	0	0	0	byte, bit
00D1	Timer 0 register	T0	R	0	0	0	0	0	0	0	0	0	byte
	Timer 0 data register	TDR0	W	1	1	1	1	1	1	1	1	1	
	Timer 0 capture data register	CDR0	R	0	0	0	0	0	0	0	0	0	
00D2	Timer 1 mode control register	TM1	R/W	0	0	0	0	0	0	0	0	0	byte, bit
00D3	Timer 1 data register	TDR1	W	1	1	1	1	1	1	1	1	1	byte
	Timer 1 PWM period register	T1PPR	W	1	1	1	1	1	1	1	1	1	byte
00D4	Timer 1 register	T1	R	0	0	0	0	0	0	0	0	0	byte
	Timer 1 capture data register	CDR1	R	0	0	0	0	0	0	0	0	0	
	Timer 1 PWM duty register	T1PDR	R/W	0	0	0	0	0	0	0	0	0	
00D5	Timer 1 PWM high register	T1PWHR	W	-	-	-	-	0	0	0	0	0	byte
00E0	Buzzer driver register	BUZR	W	1	1	1	1	1	1	1	1	1	byte
00E1	RAM page selection register	RPR	R/W	-	-	-	-	-	0	0	0	0	byte, bit
00EA	Interrupt enable register high	IENH	R/W	0	0	-	-	-	-	-	-	0	byte, bit
00EB	Interrupt enable register low	IENL	R/W	0	-	-	-	0	0	-	0	0	byte, bit
00EC	Interrupt request register high	IRQH	R/W	0	0	-	-	-	-	-	-	0	byte, bit
00ED	Interrupt request register low	IRQL	R/W	0	-	-	-	0	0	-	0	0	byte, bit
00EE	Interrupt edge selection register	IEDS	R/W	-	-	-	-	0	0	0	0	0	byte, bit
00EF	A/D converter mode control register	ADCM	R/W	0	0	0	0	0	0	0	0	1	byte, bit
00F0	A/D converter result high register	ADCRH	R(W)	0	1	0	Undefined					byte	
00F1	A/D converter result low register	ADCRL	R	Undefined								byte	
00F2	Basic interval timer register	BITR	R	Undefined								byte	
	Clock control register	CKCTLR	W	0	-	0	1	0	1	1	1		1

Table 8-1 Control Registers

Address	Register Name	Symbol	R/W	Initial Value								Addressing Mode	
				7	6	5	4	3	2	1	0		
00F4	Watch dog timer register	WDTR	W	0	1	1	1	1	1	1	1	1	byte
	Watch dog timer data register	WDTDR	R	Undefined									
00F5	Stop & sleep mode control register	SSCR	W	0	0	0	0	0	0	0	0	0	byte
00F7	PFD control register	PFDR	R/W	-	-	-	-	-	0	0	0	0	byte, bit
00F8	Port selection register 0	PSR0	W	0	0	0	0	0	0	0	0	0	byte
00F9	Port selection register 1	PSR1	W	-	-	-	-	0	0	0	0	0	byte
00FC	Pull-up selection register 0	PU0	W	0	0	0	0	0	0	0	0	0	byte
00FD	Pull-up selection register 1	PU1	W	-	-	-	0	0	0	0	0	0	byte
00FF	Pull-up selection register 3	PU3	W	-	-	0	0	0	0	0	0	-	byte

**Table 8-1 Control Registers**

1. The 'byte, bit' means registers are controlled by both bit and byte manipulation instruction. Caution) The R/W registers except T1PDR are both can be byte and bit manipulated.
2. The 'byte' means registers are controlled by only byte manipulation instruction. Do not use bit manipulation instruction such as SET1, CLR1 etc. If bit manipulation instruction is used on these registers, content of other seven bits are may varied to unwanted value.

\*The mark of '-' means this bit location is reserved.

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0C0H	R0	R0 Port Data Register							
0C1H	R0IO	R0 Port Direction Register							
0C2H	R1	R1 Port Data Register							
0C3H	R1IO	R1 Port Direction Register							
0C6H	R3	R3 Port Data Register							
0C7H	R3IO	R3 Port Direction Register							
0C8H	R0OD	R0 Open Drain Selection Register							
0C9H	R1OD	R1 Open Drain Selection Register							
0CBH	R3OD	R3 Open Drain Selection Register							
0D0H	TM0	-	-	CAP0	T0CK2	T0CK1	T0CK0	T0CN	T0ST
0D1H	T0/TDR0/ CDR0	Timer0 Register / Timer0 Data Register / Timer0 Capture Data Register							
0D2H	TM1	POL	16BIT	PWM1E	CAP1	T1CK1	T1CK0	T1CN	T1ST
0D3H	TDR1/ T1PPR	Timer1 Data Register / Timer1 PWM Period Register							
0D4H	T1/CDR1/ T1PDR	Timer1 Register / Timer1 Capture Data Register / Timer1 PWM Duty Register							
0D5H	PWM1HR	-	-	-	-	Timer1 PWM High Register			
0E0H	BUZR	BUCK1	BUCK0	BUR5	BUR4	BUR3	BUR2	BUR1	BUR0
0E1H	RPR	-	-	-	-	-	RPR2	RPR1	RPR0
0EAH	IENH	INT0E	INT1E	-	-	-	-	-	T0E
0EBH	IENL	T1E	-	-	-	ADCE	WDTE	-	BITE
0ECH	IRQH	INT0IF	INT1IF	-	-	-	-	-	T0IF
0EDH	IRQL	T1IF	-	-	-	ADCIF	WDTIF	-	BITIF
0EEH	IEDS	-	-	-	-	IED1H	IED1L	IED0H	IED0L
0EFH	ADCM	ADEN	ADCK	ADS3	ADS2	ADS1	ADS0	ADST	ADSF
0F0H	ADCRH	PSSEL1	PSSEL0	ADC8	-	-	-	ADC Result Reg. High	
0F1H	ADCRL	ADC Result Register Low							
0F2H	BITR <sup>1</sup>	Basic Interval Timer Data Register							
	CKCTLR <sup>1</sup>	ADRST	-	RCWDT	WDTON	BTCL	BTS2	BTS1	BTS0
0F4H	WDTR	WDTCL	7-bit Watchdog Timer Register						
	WDTDR	Watchdog Timer Data Register (Counter Register)							
0F5H	SSCR	Stop & Sleep Mode Control Register							
0F7H	PFDR	-	-	-	-	-	PFDEN	PFDM	PFDS
0F8H	PSR0	-	PWM1OE	-	EC0E	-	-	INT1E	INT0E
0F9H	PSR1	-	-	-	-	AVREFS	BUZO	-	T0O
0FCH	PU0	R0 Pull-up Selection Register							

Table 8-2 Control Register Function Description

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0FDH	PU1	R1 Pull-up Selection Register							
0FFH	PU3	R3 Pull-up Selection Register							

**Table 8-2 Control Register Function Description**

1. The register *BITR* and *CKCTRL* are located at same address. Address *ECH* is read as *BITR*, written to *CKCTRL*.

Caution) The registers of dark-shaded area can not be accessed by bit manipulation instruction such as "SET1, CLR1", but should be accessed by register operation instruction such as "LDM dp,#imm".

### 8.4 Addressing Mode

The MC80 series MCU uses six addressing modes;

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

#### 8.4.1 Register Addressing

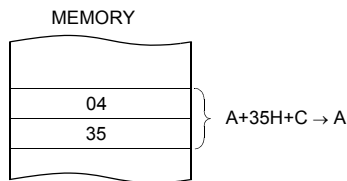
Register addressing accesses the A, X, Y, C and PSW.

#### 8.4.2 Immediate Addressing → #imm

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

Example:

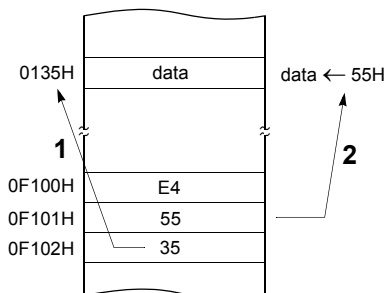
```
0435   ADC   #35H
```



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

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

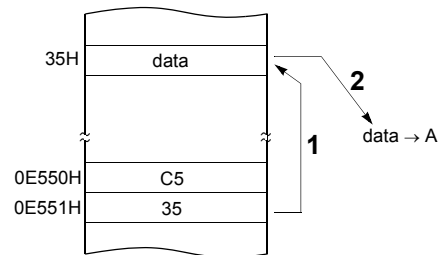


#### 8.4.3 Direct Page Addressing → dp

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

Example; G=0

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



#### 8.4.4 Absolute Addressing → !abs

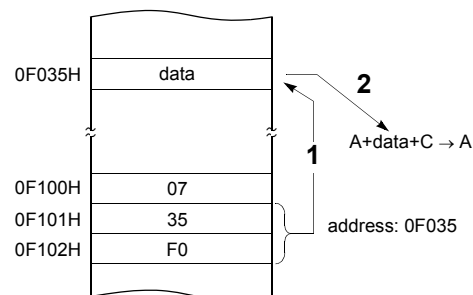
Absolute addressing sets corresponding memory data to Data, i.e. second byte (Operand I) of command becomes lower level address and third byte (Operand II) becomes upper level address.

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;

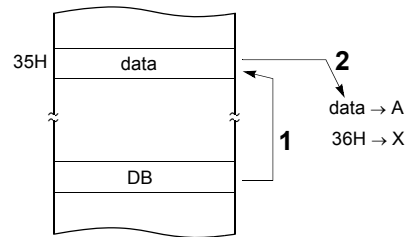
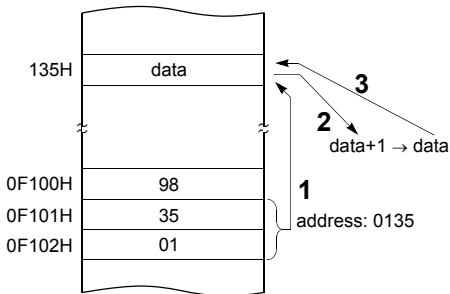
```
0735F0  ADC   !0F035H     ;A ←ROM[0F035H]
```



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

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

```
983501 INC !0135H ;A ←ROM[135H]
```

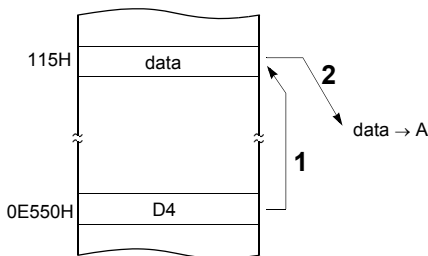


**8.4.5 Indexed Addressing**

**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

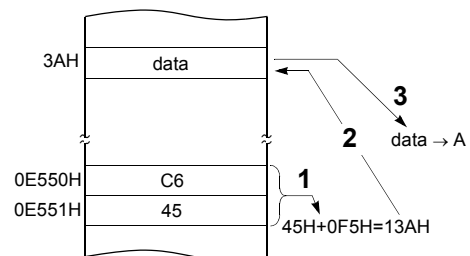
```
D4 LDA {X} ;ACC←RAM[X].
```



**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
```



**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}+

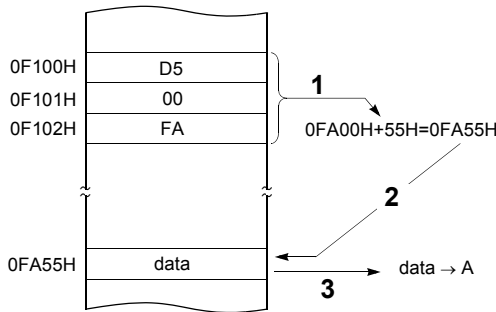
**Y indexed direct page (8 bit offset) → dp+Y**

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 (2). Use Y register instead of X.

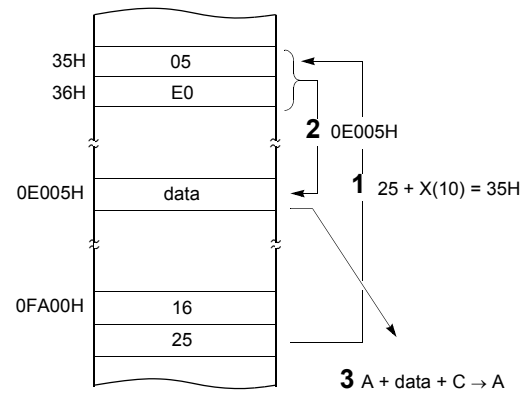
**Y indexed absolute → !abs+Y**

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
```



```
1625 ADC [25H+X]
```



### 8.4.6 Indirect Addressing

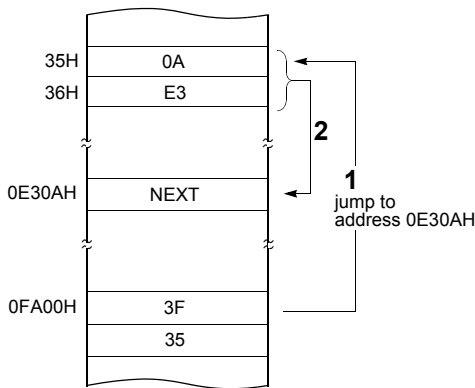
#### Direct page indirect → [dp]

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

```
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

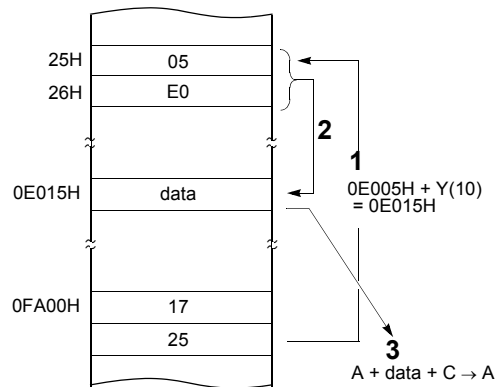
#### Y indexed indirect → [dp]+Y

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]

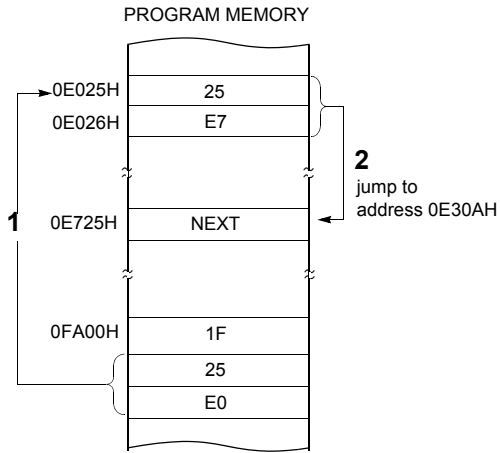
The program jumps to address specified by 16-bit absolute address.

JMP

Example; G=0



1F25E0 JMP [!0C025H]



### 9. I/O PORTS

The MC80F0504/0604 has three ports (R0, R1 and R3). These ports pins may be multiplexed with an alternate function for the peripheral features on the device. All port can drive maximum 20mA of high current in output low state, so it can directly drive LED device.

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 MC80F0504/0604 have 0 written to them by reset function. On the other hand,

its initial status is input.

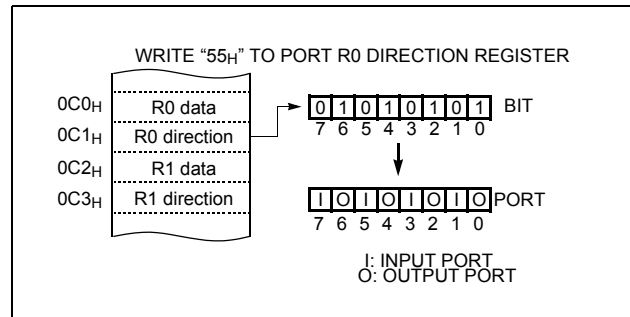
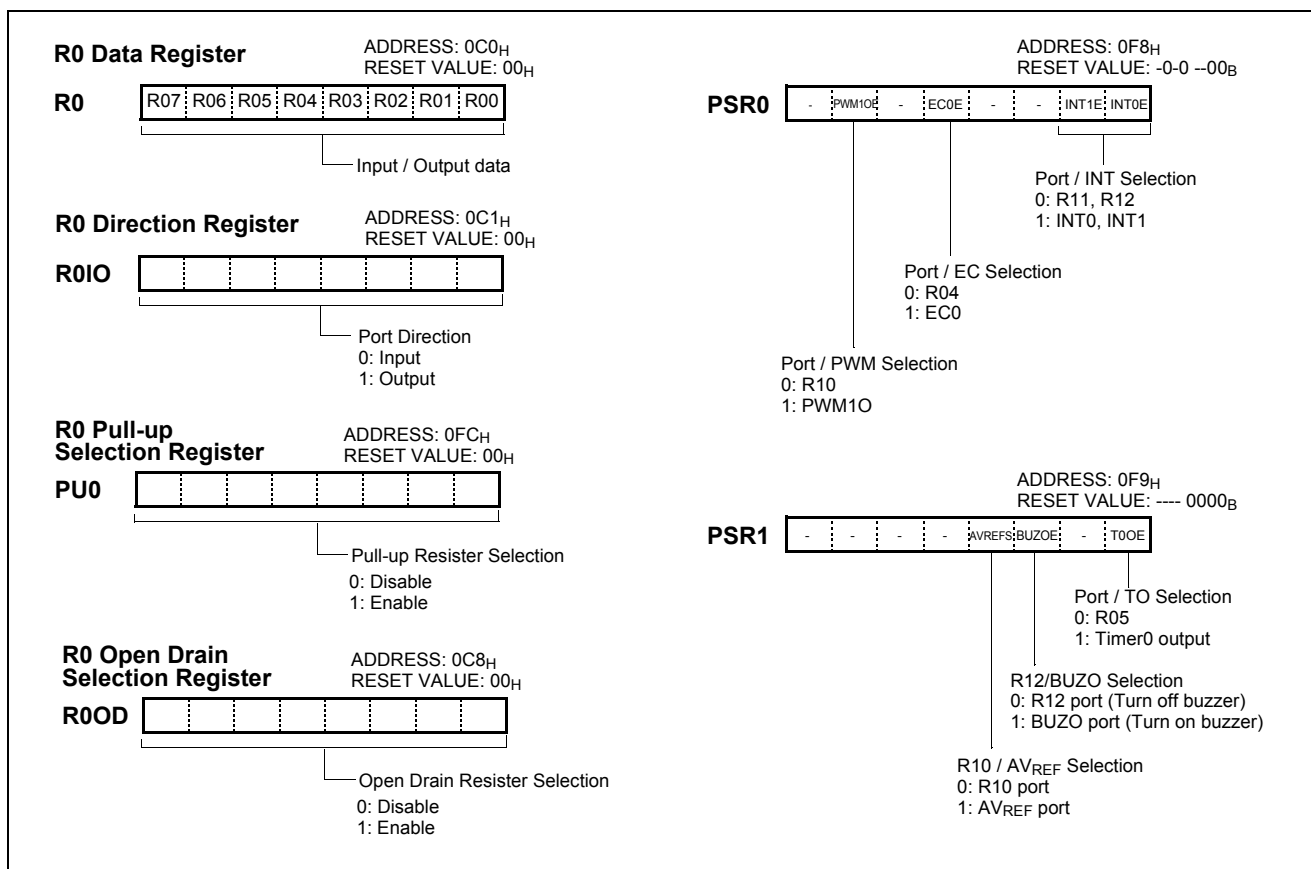


Figure 9-1 Example of port I/O assignment

#### 9.1 R0 and R0IO register

R0 is an 8-bit CMOS bidirectional I/O port (address 0C0H). Each I/O pin can independently used as an input or an output through the R0IO register (address 0C1H). When R00 through R07 pins are used as input ports, an on-chip pull-up resistor can be connected to them in 1-bit units

with a pull-up selection register 0 (PU0). Each I/O pin of R0 port can be used to open drain output port by setting the corresponding bit of the open drain selection register 0 (R0OD).



**Figure 9-2 R0 Port Register**

In addition, Port R0 is multiplexed with various alternate functions. The port selection register PSR0 (address 0F8H) and PSR1 (address 0F9H) control the selection of alternate functions such as event counter input 0 (EC0) and timer 0 output (T00). When the alternate function is selected by writing “1” in the corresponding bit of PSR0 or PSR1, port pin can be used as a corresponding alternate features regardless of the direction register R0IO.

The ADC input channel 1~7 (AN1~AN7) can be selected by setting ADCM(00EFH) register to enable the corresponding peripheral operation and select operation mode.

Port Pin	Alternate Function
R00	
R01	AN1 (ADC Input channel 1)
R02	AN2 (ADC Input channel 2)
R03	AN3 (ADC Input channel 3)
R04	AN4 (ADC Input channel 4)
	EC0 (Event counter input 0)
R05	AN5 (ADC Input channel 5)
	T00 (Timer output 0)
R06	AN6 (ADC Input channel 6)
R07	AN7 (ADC Input channel 7)

### 9.2 R1 and R1IO register

R1 is a 5-bit CMOS bidirectional I/O port (address 0C2H). Each I/O pin can independently used as an input or an output through the R1IO register (address 0C3H). When R10 through R14 pins are used as input ports, an on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up selection register 1 (PU1). Each I/O pin of R0 port can be used to open drain output port by setting the corresponding bit of the open drain selection register 1 (R1OD).

In addition, Port R1 is multiplexed with various alternate functions. The port selection register PSR0 (address 0F8H) and PSR1 (address 0F9H) control the selection of alternate functions such as Analog reference voltage input (AVREF), external interrupt 0 (INT0), external interrupt 1 (INT1), PWM 1 output (PWM1O) and buzzer output (BUZO). When the alternate function is selected by writing “1” in the corresponding bit of PSR0 or PSR1, port pin can be used as a corresponding alternate features regardless of the direction register R1IO.

The ADC input channel 0 (AN0) can be selected by setting ADCM(00EFH) register to enable ADC and select channel 0 .

Port Pin	Alternate Function
R10	AN0 (ADC input channel 0) AVREF (Analog reference voltage) PWM1O (PWM 1 output)
R11	INT0 (External Interrupt 0)
R12	INT1 (External Interrupt 1) BUZO (Buzzer output)

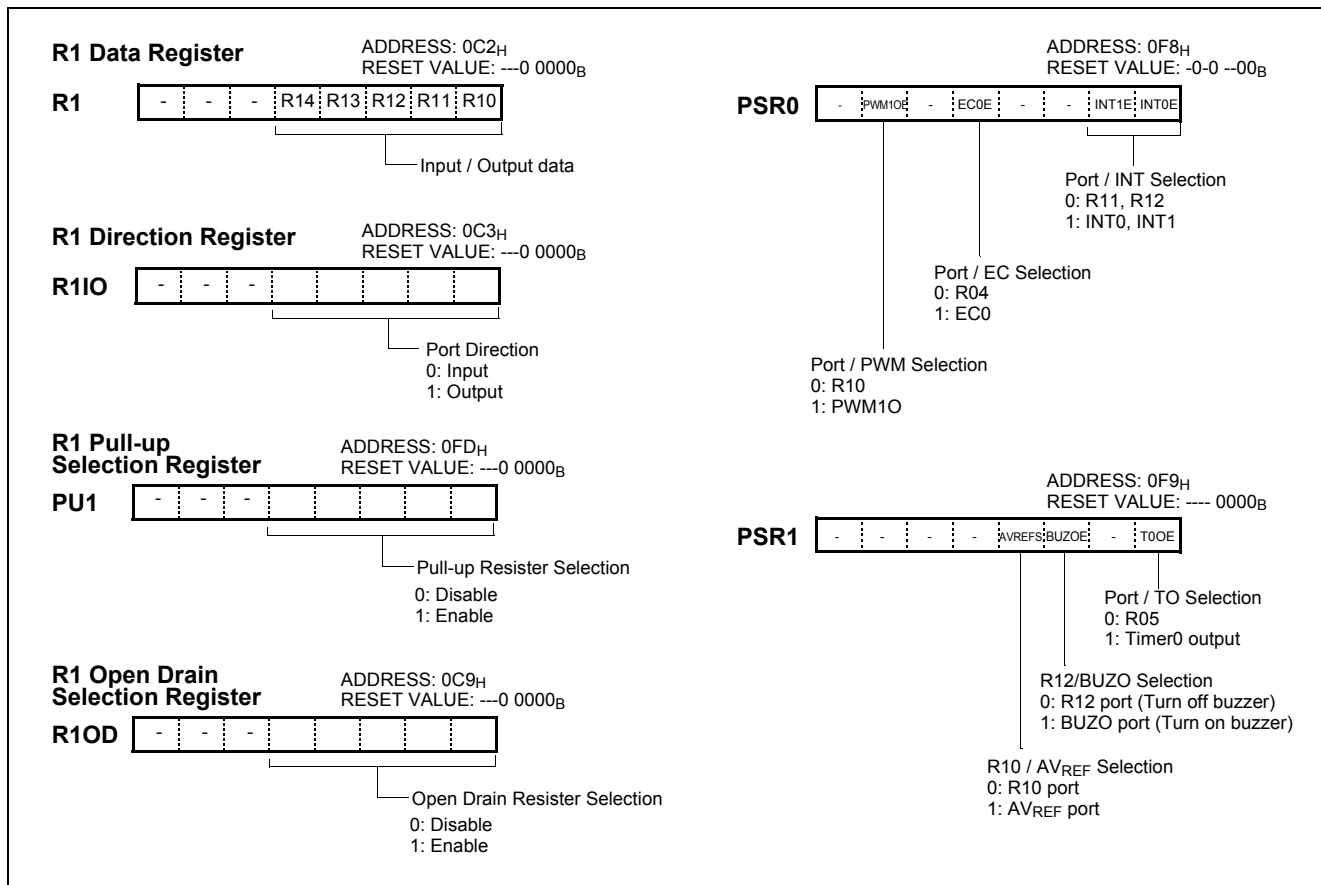


Figure 9-3 R1 Port Register

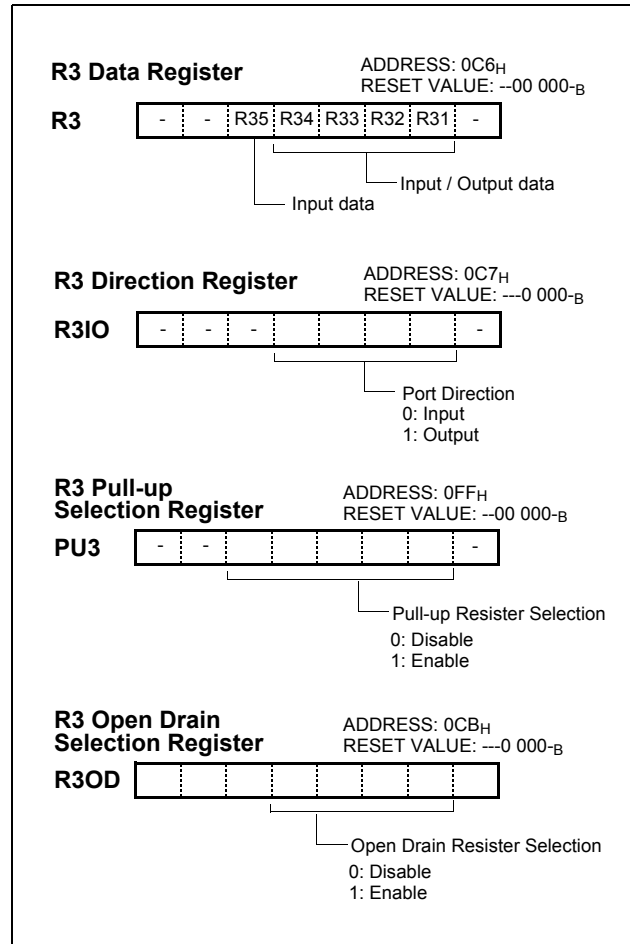
### 9.3 R3 and R3IO register

R3 is a 5-bit CMOS bidirectional I/O port (address 0C6H). Each I/O pin (except R35) can independently used as an input or an output through the R3IO register (address 0C7H). R35 is an input only port. When R31 through R35 pins are used as input ports, an on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up selection register 3 (PU3). R31 through R34 pins can be used to open drain output port by setting the corresponding bit of the open drain selection register 3 (R3OD).

In addition, Port R3 is multiplexed with alternate functions. R31 and R32 can be used as ADC input channel 14 and 15 by setting ADCM to enable ADC and select channel 14 and 15.

Port Pin	Alternate Function
R31	AN14 (ADC input channel 14)
R32	AN15 (ADC input channel 15)

R33, R34 and R35 is multiplexed with  $X_{IN}$ ,  $X_{OUT}$ , and  $\overline{RESET}$  pin. These pins can be used as general I/O pins by setting writing option described in "21. DEVICE CONFIGURATION AREA".



## 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 main-frequency clock oscillator. The system clock operation can be easily obtained by attaching a crystal or a ceramic resonator between the X<sub>IN</sub> and X<sub>OUT</sub> pin, respectively. The system clock can also be obtained from the external oscillator. In this case, it is necessary to input an external clock signal to the X<sub>IN</sub> pin and open the X<sub>OUT</sub> pin. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuit-

ry is through a divide-by-two flip-flop, but minimum and maximum high and low times specified on the data sheet must be observed.

To the peripheral block, the clock among the not-divided original clock, clocks divided by 1, 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 (CKCTLR). See "11. BASIC INTERVAL TIMER" on page 42 for details.

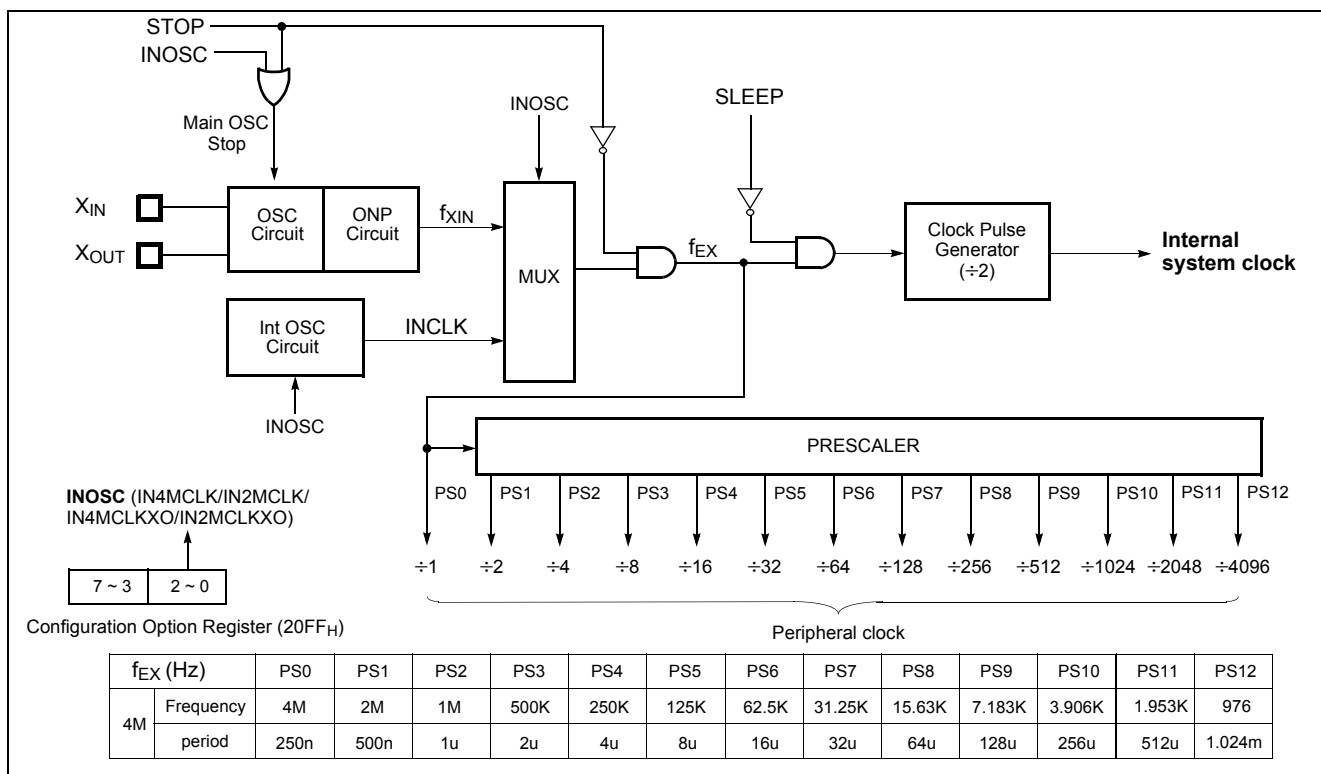


Figure 10-1 Block Diagram of Clock Generator

### 10.1 Oscillation Circuit

X<sub>IN</sub> and X<sub>OUT</sub> are the input and output, respectively, a inverting amplifier which can be set for use as an on-chip oscillator, as shown in Figure 10-2.

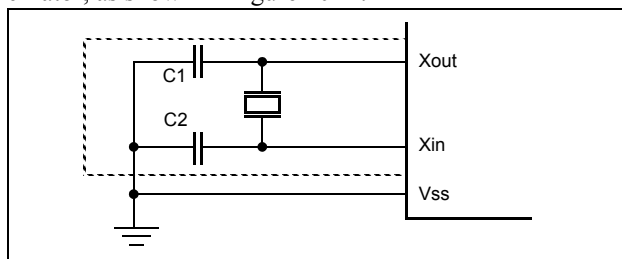
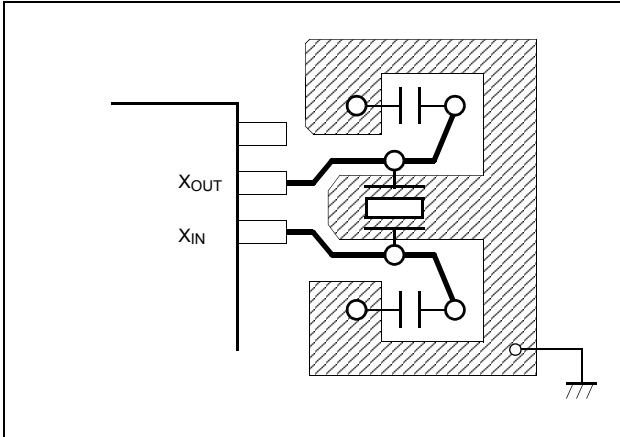


Figure 10-2 Oscillator Connections

**Note:** When using a system clock oscillator, carry out wiring in the broken line area in Figure 10-2 to prevent any effects from wiring capacities.

- Minimize the wiring length.
- Do not allow wiring to intersect with other signal conductors.
- Do not allow wiring to come near changing high current.
- Set the potential of the grounding position of the oscillator capacitor to that of V<sub>SS</sub>. Do not ground to any ground pattern where high current is present.
- Do not fetch signals from the oscillator.

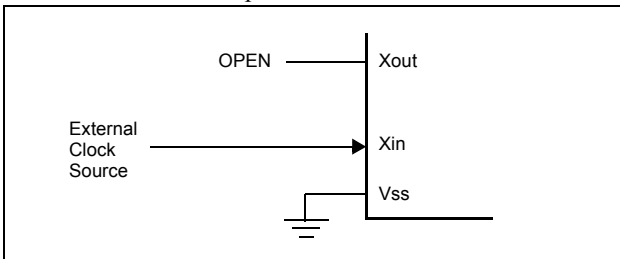
In addition, see Figure 10-3 for the layout of the crystal.



**Figure 10-3 Layout of Oscillator PCB circuit**

To drive the device from an external clock source, Xout should be left unconnected while Xin is driven as shown in Figure 10-4. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum high and low times specified on the data sheet must be observed.

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.



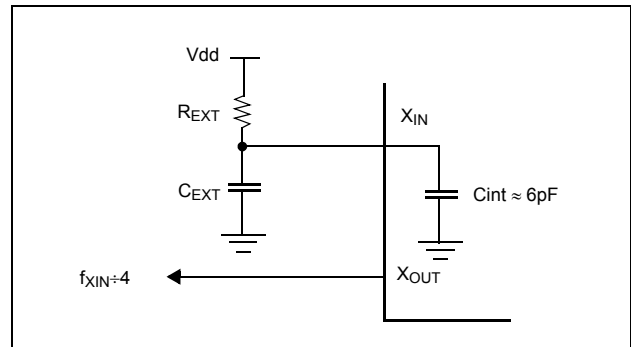
**Figure 10-4 External Clock Connections**

In addition, the MC80F0504/0604 has an ability for the external RC oscillated operation. It offers additional cost savings for **timing insensitive applications**. The RC oscillator frequency is a function of the supply voltage, the external resistor ( $R_{EXT}$ ) and capacitor ( $C_{EXT}$ ) values, and the operating temperature.

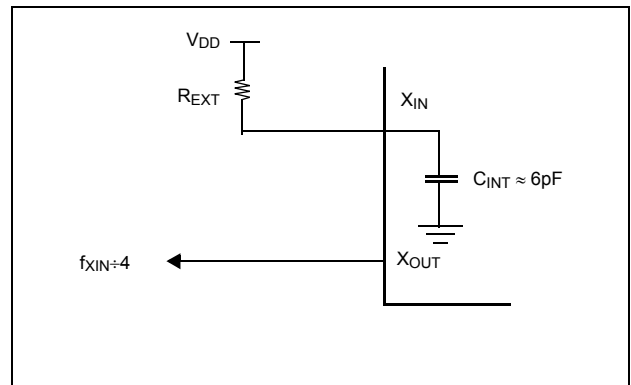
The user needs to take into account variation due to tolerance of external R and C components used.

Figure 10-5 shows how the RC combination is connected to the MC80F0504/0604. External capacitor ( $C_{EXT}$ ) can be

omitted for more cost saving. However, the characteristics of external R only oscillation are more variable than external RC oscillation.



**Figure 10-5 RC Oscillator Connections**



**Figure 10-6 R Oscillator Connections**

To use the RC oscillation, the CLK option of the configuration bits (20FFH) should be set to "EXRC or EXRCXO".

The oscillator frequency, divided by 4, is output from the Xout pin, and can be used for test purpose or to synchronize other logic.

In addition to external crystal/resonator and external RC/R oscillation, the MC80F0504/0604 provides the internal 4MHz or 2MHz oscillation. The internal 4MHz/2MHz oscillation needs no external parts.

To use the internal 4MHz/2MHz oscillation, the CLK option of the configuration bits should be set to "IN4MCLK", "IN2MCLK", "IN4MCLKXO" or "IN2MCLKXO". For detail description on the configuration bits, refer to "21. DEVICE CONFIGURATION AREA"

## 11. BASIC INTERVAL TIMER

The MC80F0504/0604 has one 8-bit Basic Interval Timer that is free-run and can not stop. Block diagram is shown in Figure 11-1. In addition, the Basic Interval Timer generates the time base for watchdog timer counting. It also provides a Basic interval timer interrupt (BITIF).

The 8-bit Basic interval timer register (BITR) is increased every internal count pulse which is divided by prescaler. Since prescaler has divided ratio by 8 to 1024, the count rate is 1/8 to 1/1024 of the oscillator frequency. As the count overflow from FFH to 00H, this overflow causes the interrupt to be generated.

The Basic Interval Timer is controlled by the clock control register (CKCTRL) shown in Figure 11-2. If the RCWDT bit is set to "1", the clock source of the BITR is changed to the internal RC oscillation.

When write "1" to bit BTCL of CKCTRL, BITR register is cleared to "0" and restart to count-up. The bit BTCL becomes "0" after one machine cycle by hardware.

If the STOP instruction executed after writing "1" to bit RCWDT of CKCTRL, it goes into the internal RC oscillated watchdog timer mode. In this mode, all of the block is halted except the internal RC oscillator, Basic Interval Timer and Watchdog Timer. More detail informations are explained in Power Saving Function. The bit WDTON decides Watchdog Timer or the normal 7-bit timer. Source clock can be selected by lower 3 bits of CKCTRL.

BITR and CKCTRL are located at same address, and address 0F2H is read as a BITR, and written to CKCTRL.

**Note:** All control bits of Basic interval timer are in CKCTRL register which is located at same address of BITR (address EC<sub>H</sub>). Address EC<sub>H</sub> is read as BITR, written to CKCTRL. Therefore, the CKCTRL can not be accessed by bit manipulation instruction.

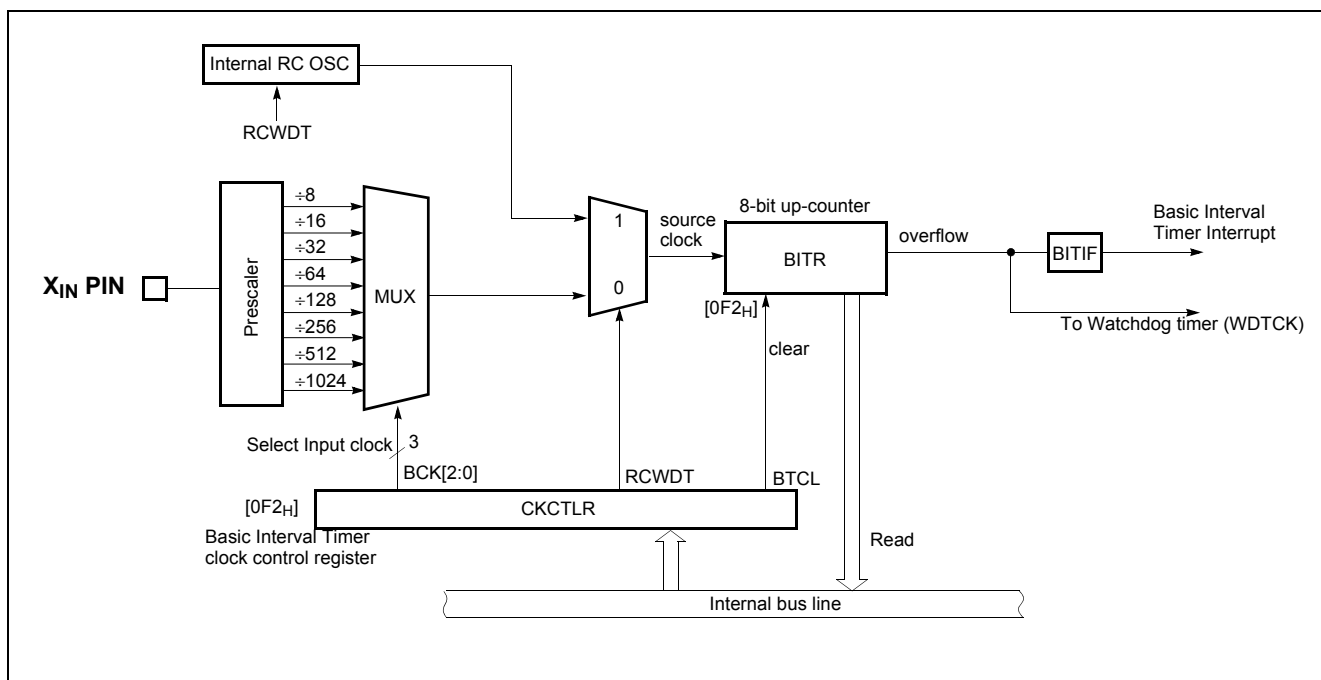
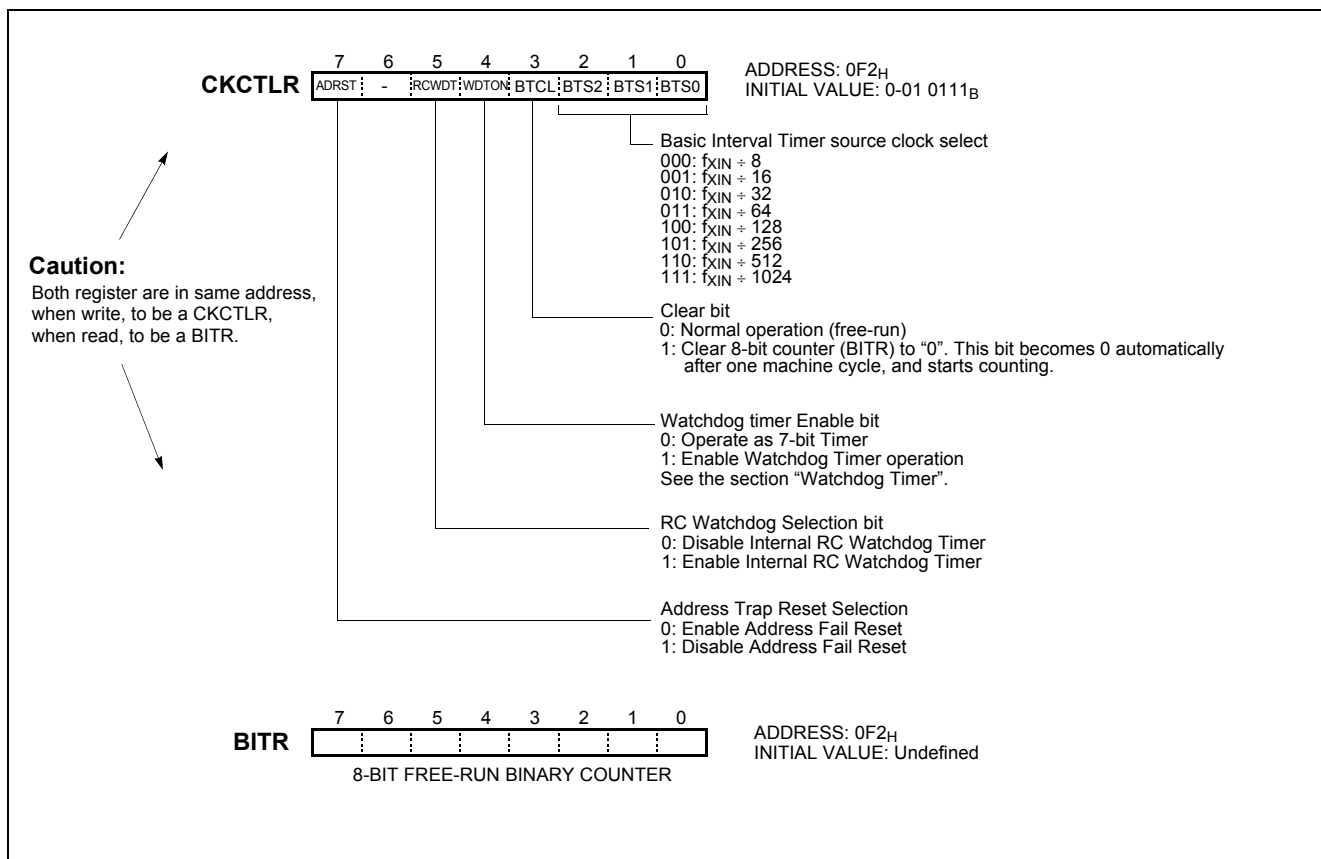


Figure 11-1 Block Diagram of Basic Interval Timer



CKCTLR [2:0]	Source clock	Interrupt (overflow) Period (ms) @ f <sub>XIN</sub> = 8MHz
000	f <sub>XIN</sub> ÷8	0.256
001	f <sub>XIN</sub> ÷16	0.512
010	f <sub>XIN</sub> ÷32	1.024
011	f <sub>XIN</sub> ÷64	2.048
100	f <sub>XIN</sub> ÷128	4.096
101	f <sub>XIN</sub> ÷256	8.192
110	f <sub>XIN</sub> ÷512	16.384
111	f <sub>XIN</sub> ÷1024	32.768

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



**Figure 11-2 BITR: Basic Interval Timer Mode Register**

Example 1:

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

```

:
LDM   CKCTLR, #1BH
SET1  BITE
EI
:
    
```

Example 2:

Interrupt request flag is generated every 8.192ms at 8MHz.

```

:
LDM   CKCTLR, #1CH
SET1  BITE
EI
:
    
```

## 12. WATCHDOG TIMER

The watchdog timer rapidly detects the CPU malfunction such as endless looping caused by noise or the like, and resumes the CPU to the normal state. The watchdog timer signal for detecting malfunction can be selected either a reset CPU or a interrupt request.

When the watchdog timer is not being used for malfunction detection, it can be used as a timer to generate an interrupt at fixed intervals.

The watchdog timer has two types of clock source. The first type is an on-chip RC oscillator which does not require any external components. This RC oscillator is separate from the external oscillator of the X<sub>IN</sub> pin. It means that the watchdog timer will run, even if the clock on the X<sub>IN</sub> pin of the device has been stopped, for example, by entering the STOP mode. The other type is a prescaled system clock.

The watchdog timer consists of 7-bit binary counter and the watchdog timer data register. 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 Watchdog timer interrupt or reset the CPU in accordance with the bit WDTON.

**Note:** Because the watchdog timer counter is enabled after clearing Basic Interval Timer, after the bit WDTON set to "1", maximum error of timer is depend on prescaler ratio of Basic Interval Timer. The 7-bit binary counter is cleared by setting WDTCL(bit7 of WDTR) and the WDTCL is cleared automatically after 1 machine cycle.

The RC oscillated watchdog timer is activated by setting the bit RCWDT as shown below.

```
LDM      CKCTLR, #3FH; enable the RC-OSC WDT
LDM      WDTR, #0FFH; set the WDT period
LDM      SSCR, #5AH; ready for STOP mode
STOP     ; enter the STOP mode
NOP
NOP
NOP     ; RC-OSC WDT running
:
```

The RC-WDT oscillation period is vary with temperature, V<sub>DD</sub> and process variations from part to part (approximately, 33~100uS). The following equation shows the RCWDT oscillated watchdog timer time-out.

$$T_{RCWDT} = CLK_{RCWDT} \times 2^8 \times WDTR + (CLK_{RCWDT} \times 2^8) / 2$$

where,  $CLK_{RCWDT} = 33 \sim 100 \mu S$

In addition, this watchdog timer can be used as a simple 7-bit timer by interrupt WDTIF. The interval of watchdog timer interrupt is decided by Basic Interval Timer. Interval equation is as below.

$$T_{WDT} = (WDTR + 1) \times \text{Interval of BIT}$$

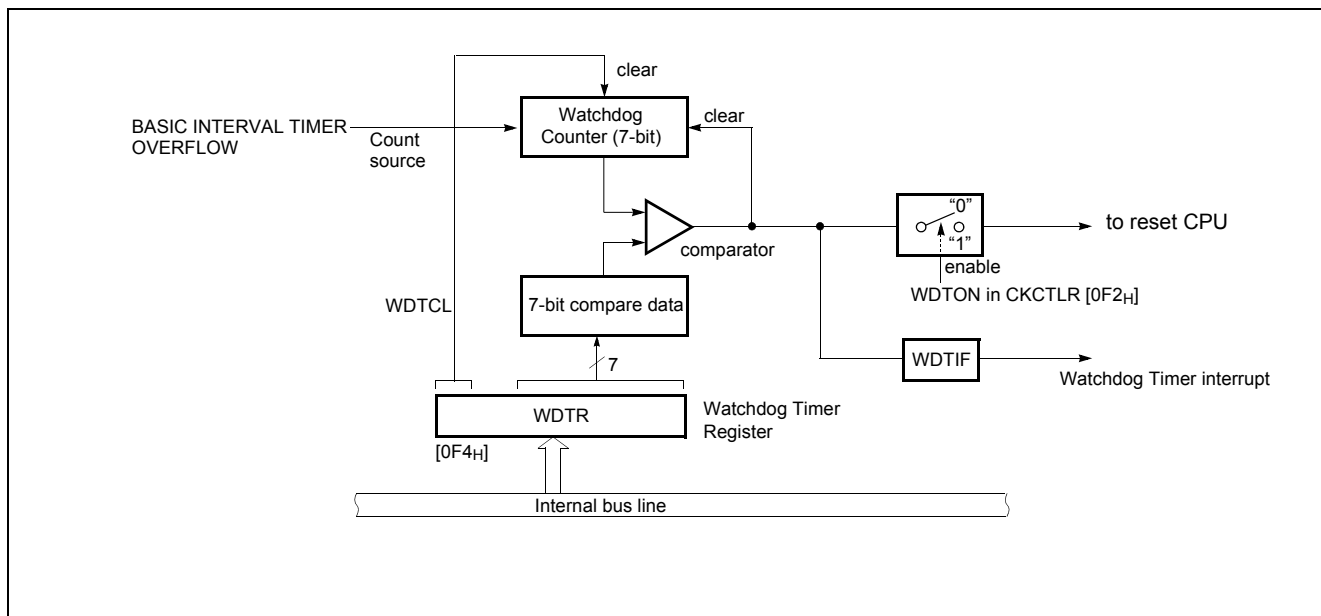


Figure 12-1 Block Diagram of Watchdog Timer

**Watchdog Timer Control**

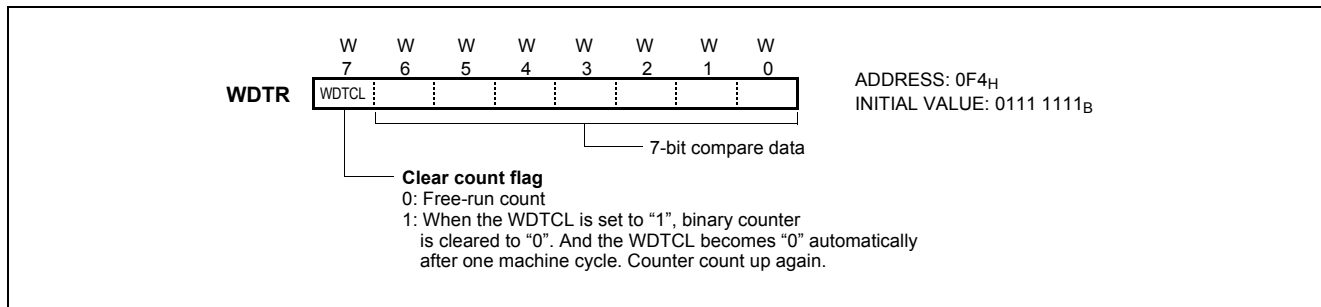
Figure 12-2 shows the watchdog timer control register. The watchdog timer is automatically disabled after reset.

The CPU malfunction is detected during setting of the detection time, selecting of output, and clearing of the binary counter. Clearing the binary counter is repeated within the detection time.

If the malfunction occurs for any cause, the watchdog timer

output will become active at the rising overflow from the binary counters unless the binary counter is cleared. At this time, when  $WDTON=1$ , a reset is generated, which drives the  $\overline{RESET}$  pin to low to reset the internal hardware. When  $WDTON=0$ , a watchdog timer interrupt (WDTIF) is generated. The  $WDTON$  bit is in register  $CLKCTLR$ .

The watchdog timer temporarily stops counting in the STOP mode, and when the STOP mode is released, it automatically restarts (continues counting).



**Figure 12-2 WDTR: Watchdog Timer Control Register**

Example: Sets the watchdog timer detection time to 1 sec. at 4.194304MHz

```

LDM    CKCTLR, #3FH           ; Select 1/1024 clock source, WDTON ← 1, Clear Counter
LDM    WDTR, #08FH
:
:
:
LDM    WDTR, #08FH           ; Clear counter
:
:
:
LDM    WDTR, #08FH           ; Clear counter
:
:
:
LDM    WDTR, #08FH           ; Clear counter

```

**Enable and Disable Watchdog**

Watchdog timer is enabled by setting  $WDTON$  (bit 4 in  $CKCTLR$ ) to “1”.  $WDTON$  is initialized to “0” during reset and it should be set to “1” to operate after reset is released.

Example: Enables watchdog timer for Reset

```

:
LDM    CKCTLR, #xxx1_xxxxB; WDTON ← 1
:
:

```

The watchdog timer is disabled by clearing bit 4 ( $WDTON$ ) of  $CKCTLR$ . The watchdog timer is halted in STOP mode and restarts automatically after STOP mode is released.

**Watchdog Timer Interrupt**

The watchdog timer can be also used as a simple 7-bit timer by clearing bit4 of  $CKCTLR$  to “0”. The interval of watchdog timer interrupt is decided by Basic Interval Timer. Interval equation is shown as below.

$$T_{WDT} = (WDTR+1) \times Interval\ of\ BIT$$

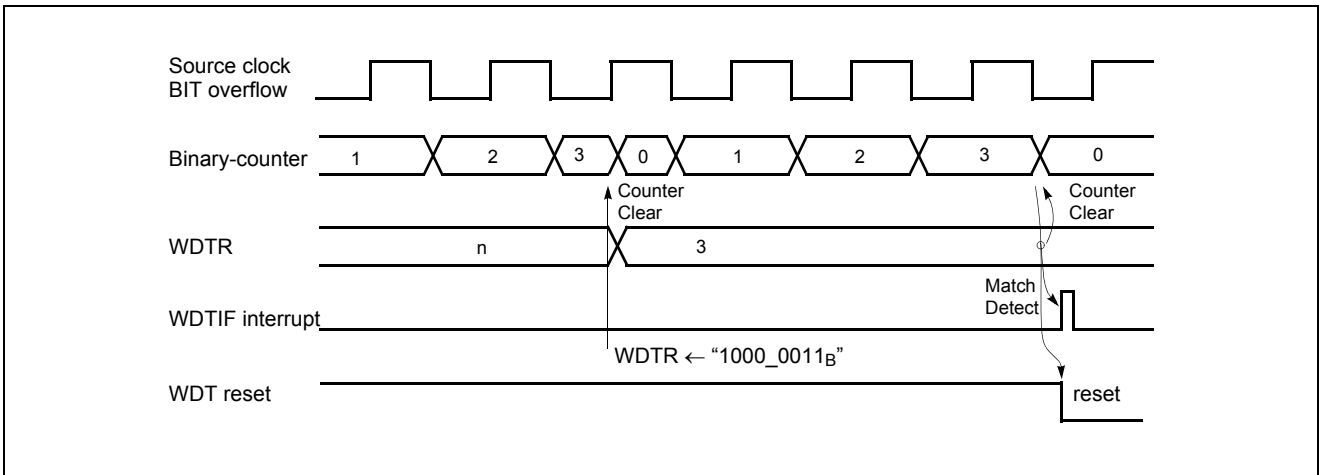
The stack pointer (SP) should be initialized before using the watchdog timer output as an interrupt source.

Example: 7-bit timer interrupt set up.

```

LDM    CKCTLR, #xxx0_xxxxB; WDTON ← 0
LDM    WDTR, #8FH      ; WDTCF ← 1

```



**Figure 12-3 Watchdog timer Timing**

If the watchdog timer output becomes active, a reset is generated, which drives the  $\overline{RESET}$  pin low to reset the internal hardware.

The main clock oscillator also turns on when a watchdog timer reset is generated in sub clock mode.

### 13. TIMER/EVENT COUNTER

The MC80F0504/0604 has two Timer/Counter. Each module can generate an interrupt to indicate that an event has occurred (i.e. timer match).

Timer 0 and Timer 1 are can be used either two 8-bit Timer/Counter or one 16-bit Timer/Counter with combine them. Also Timer 2 and Timer 3 are same.

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 most clock consists of 2048 oscillator periods, the count rate is 1/2 to 1/2048 of the oscillator frequency.

In the “counter” function, the register is increased in response to a 0-to-1 (rising edge) transition at its correspond-

ing external input pin, EC0.

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

Timer 0 and Timer 1 is shared with "PWM" function and "Compare output" function. It has six operating modes: "8-bit timer/counter", "16-bit timer/counter", "8-bit capture", "16-bit capture", "8-bit compare output", and "10-bit PWM" which are selected by bit in Timer mode register TM0 and TM1 as shown in Table 13-1, Figure 13-1 .

16BIT	CAP0	CAP1	PWM1E	T0CK [2:0]	T1CK [1:0]	PWM1O	TIMER 0	TIMER 1
0	0	0	0	XXX	XX	0	8-bit Timer	8-bit Timer
0	0	1	0	111	XX	0	8-bit Event counter	8-bit Capture
0	1	0	0	XXX	XX	1	8-bit Capture (internal clock)	8-bit Compare Output
0	X	0	1	XXX	XX	1	8-bit Timer/Counter	10-bit PWM
1	0	0	0	XXX	11	0	16-bit Timer	
1	0	0	0	111	11	0	16-bit Event counter	
1	1	1	0	XXX	11	0	16-bit Capture (internal clock)	

**Table 13-1 Operation Modes of Timer 0, 1**

1. X means the value of “0” or “1” corresponds to user operation.

		R/W	R/W	R/W	R/W	R/W	R/W			
		5	4	3	2	1	0			
<b>TM0</b>	-	-	CAP0	T0CK2	T0CK1	T0CK0	T0CN	T0ST	ADDRESS: 0D0 <sub>H</sub>	INITIAL VALUE: --00 0000 <sub>B</sub>

Bit Name	Bit Position	Description
CAP0	TM0.5	0: Timer/Counter mode 1: Capture mode selection flag
T0CK2	TM0.4	000: 8-bit Timer, Clock source is $f_{XIN} \div 2$
T0CK1	TM0.3	001: 8-bit Timer, Clock source is $f_{XIN} \div 4$
T0CK0	TM0.2	010: 8-bit Timer, Clock source is $f_{XIN} \div 8$ 011: 8-bit Timer, Clock source is $f_{XIN} \div 32$ 100: 8-bit Timer, Clock source is $f_{XIN} \div 128$ 101: 8-bit Timer, Clock source is $f_{XIN} \div 512$ 110: 8-bit Timer, Clock source is $f_{XIN} \div 2048$ 111: EC0 (External clock)
T0CN	TM0.1	0: Timer count pause 1: Timer count start
T0ST	TM0.0	0: When cleared, stop the counting. 1: When set, Timer 0 Count Register is cleared and start again.

		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
		7	6	5	4	3	2	1	0	
<b>TM1</b>	POL	16BIT	PWM1E	CAP1	T1CK1	T1CK0	T1CN	T1ST	ADDRESS: 0D2 <sub>H</sub>	INITIAL VALUE: 00 <sub>H</sub>

Bit Name	Bit Position	Description
POL	TM1.7	0: PWM Duty Active Low 1: PWM Duty Active High
16BIT	TM1.6	0: 8-bit Mode 1: 16-bit Mode
PWM1E	TM1.5	0: Disable PWM 1: Enable PWM
CAP1	TM1.4	0: Timer/Counter mode 1: Capture mode selection flag
T1CK1	TM1.3	00: 8-bit Timer, Clock source is $f_{XIN}$
T1CK0	TM1.2	01: 8-bit Timer, Clock source is $f_{XIN} \div 2$ 10: 8-bit Timer, Clock source is $f_{XIN} \div 8$ 11: 8-bit Timer, Clock source is Using the Timer 0 Clock
T1CN	TM1.1	0: Timer count pause 1: Timer count start
T1ST	TM1.0	0: When cleared, stop the counting. 1: When set, Timer 0 Count Register is cleared and start again.

		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
		7	6	5	4	3	2	1	0	
<b>TDR0</b>										ADDRESS: 0D1 <sub>H</sub> INITIAL VALUE: 0FF <sub>H</sub>

		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
		7	6	5	4	3	2	1	0	
<b>TDR1</b>										ADDRESS: 0D3 <sub>H</sub> INITIAL VALUE: 0FF <sub>H</sub>

Read: Count value read  
Write: Compare data write

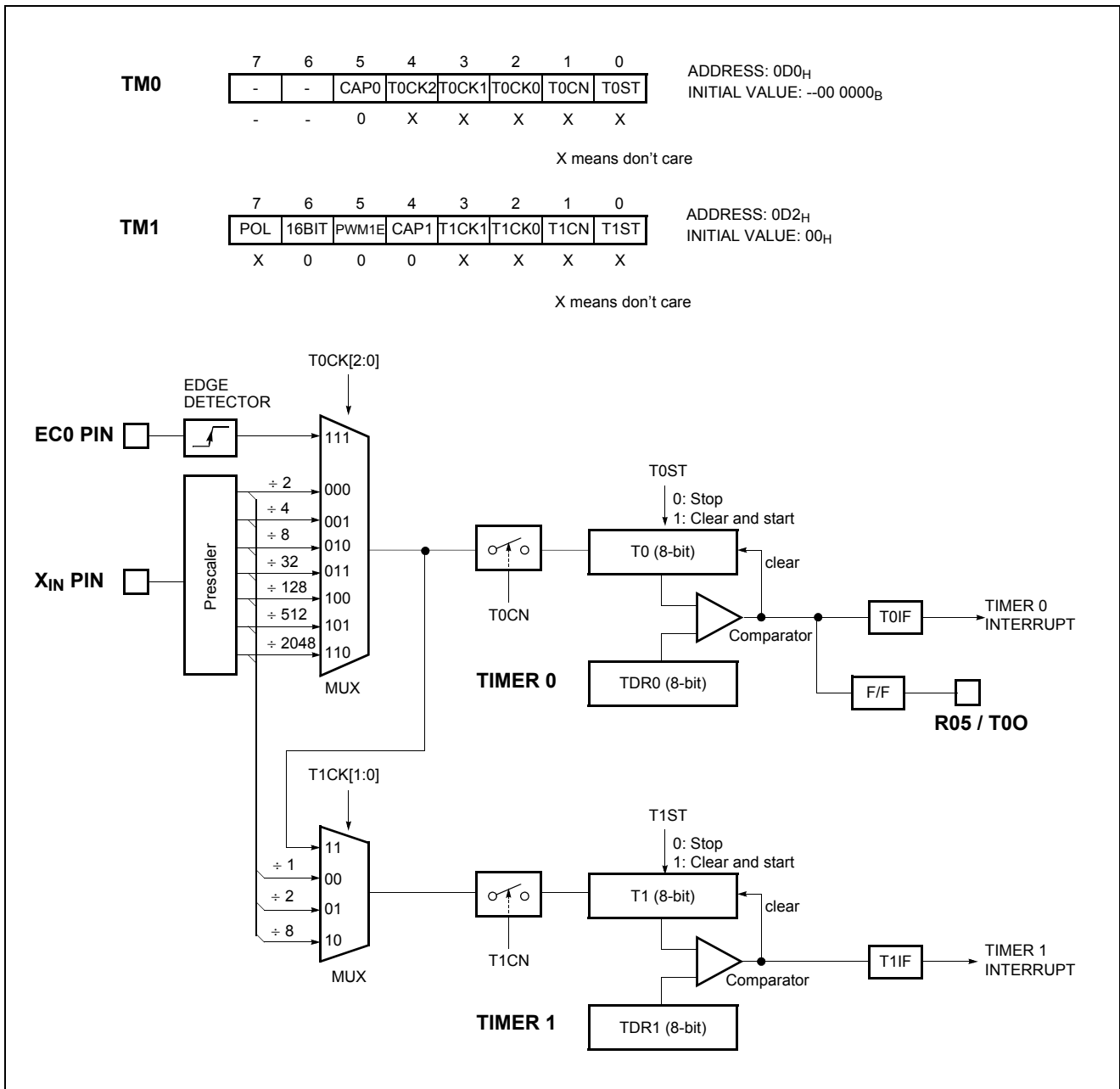
Figure 13-1 TM0, TM1 Registers

### 13.1 8-bit Timer / Counter Mode

The MC80F0504/0604 has two 8-bit Timer/Counters, Timer 0 and Timer 1, which are shown in Figure 13-2 .

The “timer” or “counter” function is selected by control registers TM0, TM1 as shown in Figure 13-1 . To use as an 8-bit timer/counter mode, bit CAP0 or CAP1 of TMx should be cleared to “0” and 16BIT and PWM1E of TM1 should be cleared to “0” (Figure 13-2 ). 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 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048 or external clock (selected by control bits TxCK0, TxCK1, TxCK2 of register TMx).



**Figure 13-2 8-bit Timer/Counter 0, 1**

**Example 1:**

Timer0 = 2ms 8-bit timer mode at 4MHz  
 Timer1 = 0.5ms 8-bit timer mode at 4MHz

```
LDM   TDR0, #249
LDM   TDR1, #249
LDM   TM0, #0000_1111B
LDM   TM1, #0000_1011B
SET1  T0E
SET1  T1E
EI
```

**Example 2:**

Timer0 = 8-bit event counter mode  
 Timer1 = 0.5ms 8-bit timer mode at 4MHz

```
LDM   TDR0, #249
LDM   TDR1, #249
LDM   TM0, #0001_1111B
LDM   TM1, #0000_1011B
SET1  T0E
SET1  T1E
EI
```

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 T0CK[2:0] of register TM0 or 1, 2, 8 selected by control bits T1CK[1:0] of register TM1. In the Timer 0, timer register T0 increases from 00<sub>H</sub> until it matches TDR0 and then reset to 00<sub>H</sub>. The match output of Timer 0 generates Timer 0 interrupt (latched in T0IF bit).

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 EC0 of the Port Selection Register (PSR0.4) is set to "1". The Timer 0 can be used as a counter by pin EC0 input, but Timer 1 can not.

**13.1.1 8-bit Timer Mode**

In the timer mode, the internal clock is used for counting up. Thus, you can think of it as counting internal clock input. The contents of TDR<sub>n</sub> are compared with the contents of up-counter, T<sub>n</sub>. If match is found, a timer *n* interrupt (T<sub>n</sub>IF) is generated and the up-counter is cleared to 0. Counting up is resumed after the up-counter is cleared.

As the value of TDR<sub>n</sub> is changeable by software, time interval is set as you want.

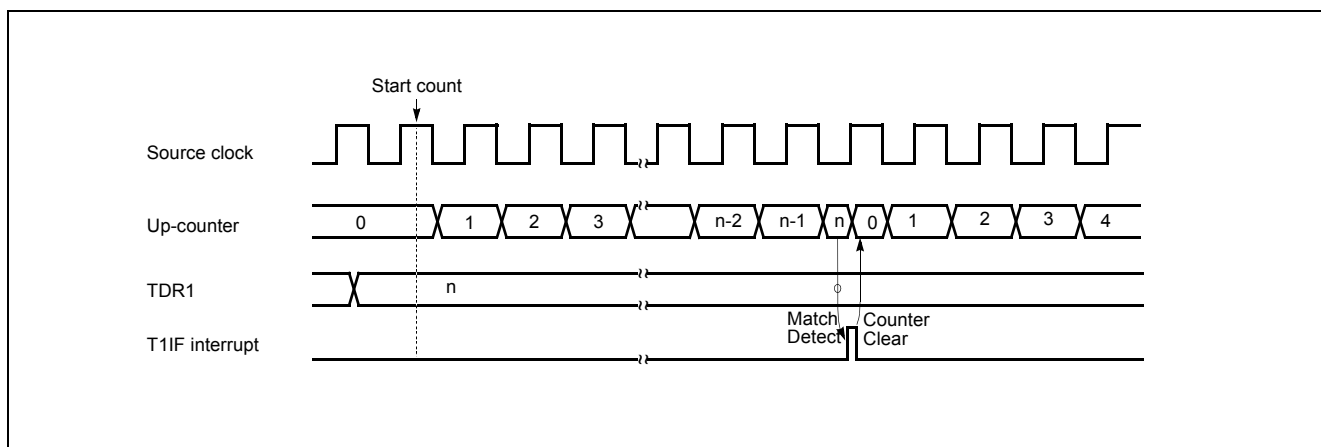


Figure 13-3 Timer Mode Timing Chart



**Example:** Make 1ms interrupt using by Timer0 at 4MHz

```
LDM    TM0,#0FH    ; divide by 32
LDM    TDR0,#124  ; 8us x (124+1)= 1ms
SET1   T0E        ; Enable Timer 0 Interrupt
EI     ; Enable Master Interrupt
```

When  $\left\{ \begin{array}{l} \text{TM0} = 0000\ 1111_B \text{ (8-bit Timer mode, Prescaler divide ratio} = 32) \\ \text{TDR0} = 124_D = 7C_H \\ f_{XIN} = 4 \text{ MHz} \end{array} \right.$

$$\text{INTERRUPT PERIOD} = \frac{1}{4 \times 10^6 \text{ Hz}} \times 32 \times (124+1) = 1 \text{ ms}$$

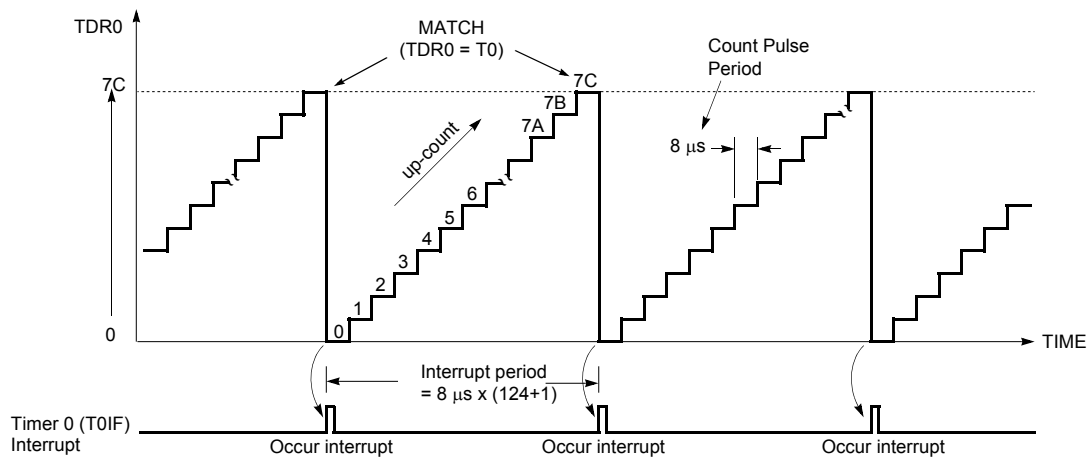


Figure 13-4 Timer Count Example

### 13.1.2 8-bit Event Counter Mode

In this mode, counting up is started by an external trigger. This trigger means rising edge of the EC0 pin input. Source clock is used as an internal clock selected with timer mode register TM0. The contents of timer data register TDR0 are compared with the contents of the up-counter T1. If a match is found, a timer interrupt request flag T0IF is generated, and the counter is cleared to "0". The counter is re-start and count up continuously by every rising edge of the EC0 pin input. The maximum frequency applied to the EC0 pin is  $f_{XIN}/2$  [Hz].

In order to use event counter function, the bit 4 of the Port Selection Register PSR0(address 0F8H) is required to be set to "1".

After reset, the value of timer data register TDRn is initialized to "0". The interval period of Timer is calculated as below equation.

$$\text{Period (sec)} = \frac{1}{f_{XIN}} \times 2 \times \text{Divide Ratio} \times (\text{TDRn} + 1)$$

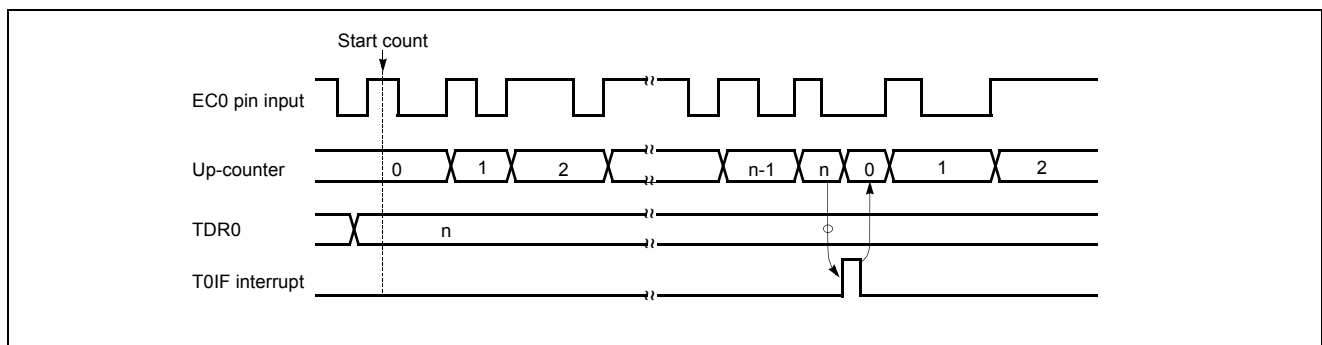


Figure 13-5 Event Counter Mode Timing Chart

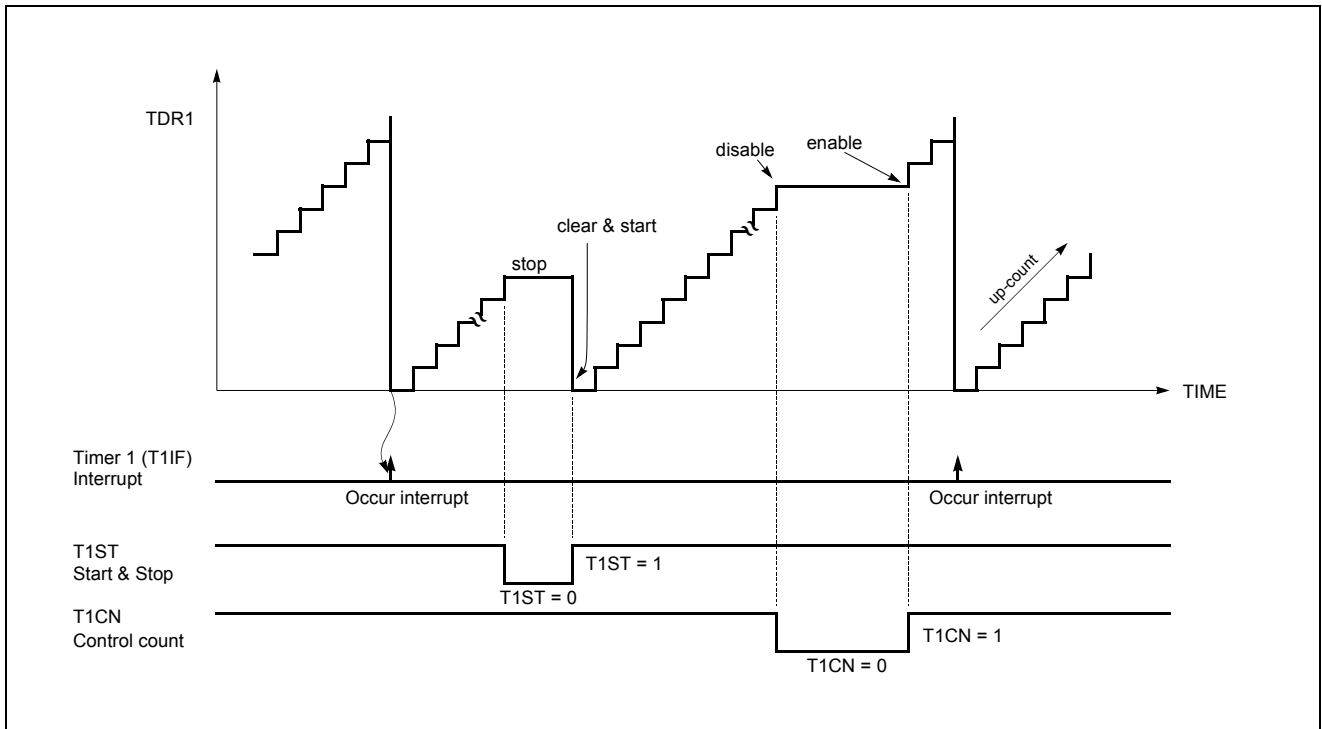
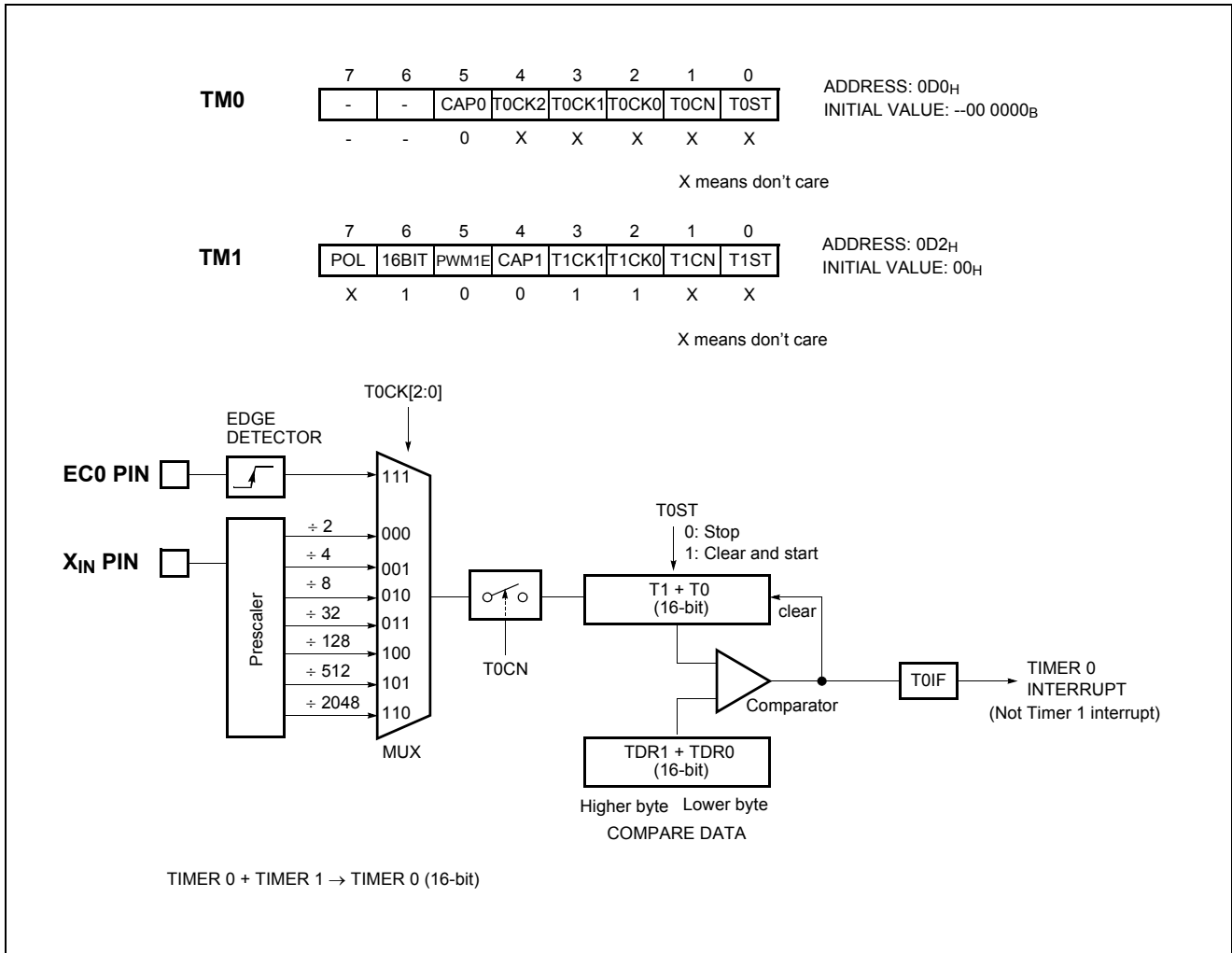


Figure 13-6 Count Operation of Timer / Event counter

### 13.2 16-bit Timer / Counter Mode

The Timer register is being run with all 16 bits. A 16-bit timer/counter register T0, T1 increments 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.

The clock source of the Timer 0 is selected either internal or external clock by bit T0CK[2:0]. In 16-bit mode, the bits T1CK[1:0] and 16BIT of TM1 should be set to "1" respectively as shown in Figure 13-7 .



**Figure 13-7 16-bit Timer/Counter for Timer 0, 1**

### 13.3 8-bit (16-bit) Compare Output

The MC80F0504/0604 has Timer Compare Output function. To pulse out, the timer match can go to port pin (T00) as shown in Figure 13-2 . Thus, pulse out is generated by the timer match. These operation is implemented to pin, R05/AN5//T00.

In this mode, the bit T0OE bit of Port Selection register1 (PSR1.0) should be set to "1". This pin output the signal

having a 50 : 50 duty square wave, and output frequency is same as below equation.

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

### 13.4 8-bit Capture Mode

The Timer 0 capture mode is set by bit CAP0 of timer mode register TM0 (bit CAP1 of timer mode register TM1 for Timer 1) as shown in Figure 13-8 .

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) increases and matches TDR0 (TDR1).

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 13-10 , 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 ob-

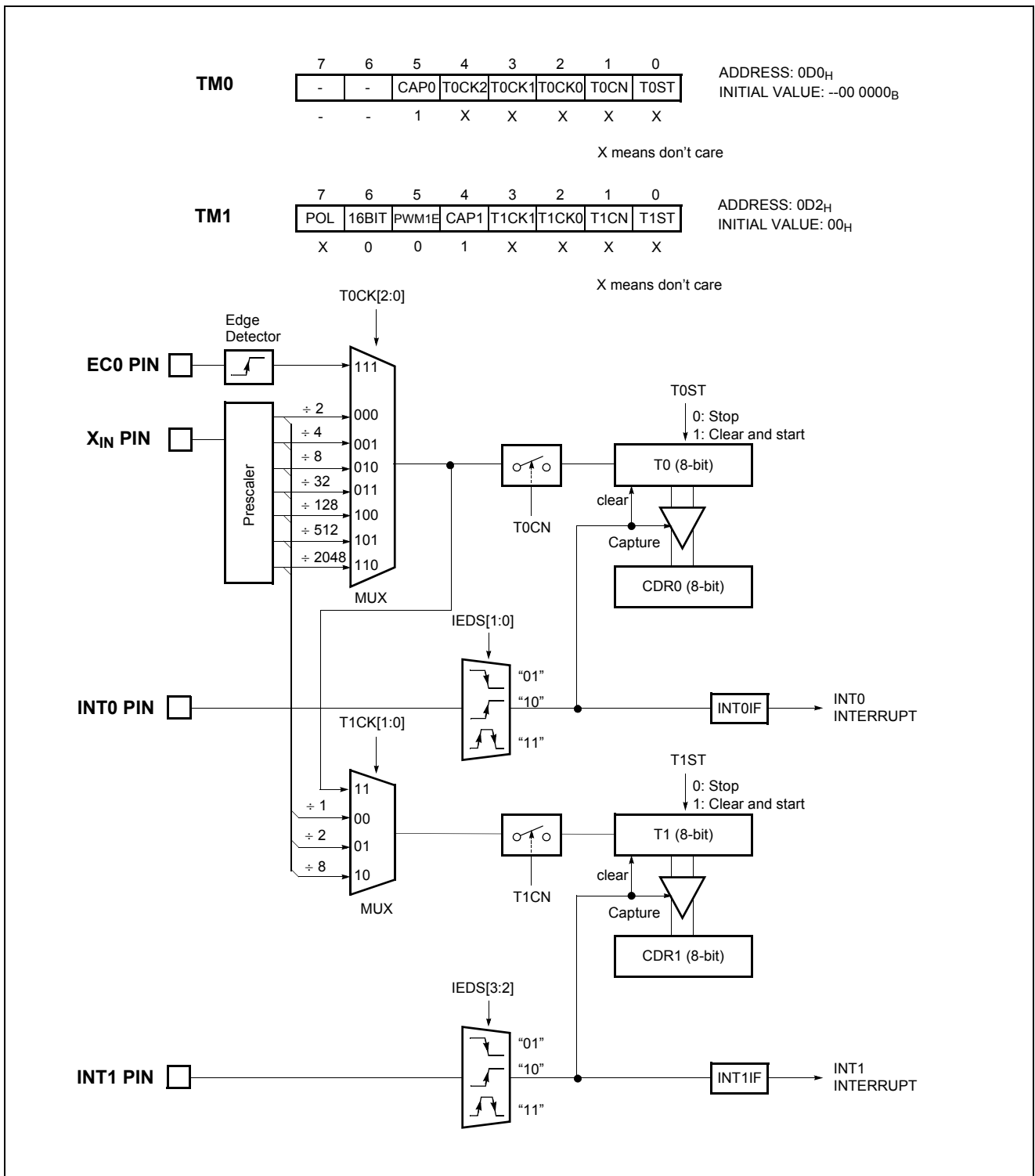
tained correct value by counting the number of timer overflow occurrence.

Timer/Counter still does the above, but with the added feature that a edge transition at external input INT<sub>x</sub> pin causes the current value in the Timer x register (T0,T1), to be captured into registers CDR<sub>x</sub> (CDR0, CDR1), 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 "16.4 External Interrupt" on page 72. In addition, the transition at INT<sub>n</sub> pin generate an interrupt.

---

**Note:** The CDR<sub>n</sub> and TDR<sub>n</sub> are in same address. In the capture mode, reading operation is read the CDR<sub>n</sub>, not TDR<sub>n</sub> because path is opened to the CDR<sub>n</sub>.

---



**Figure 13-8 8-bit Capture Mode for Timer 0, 1**

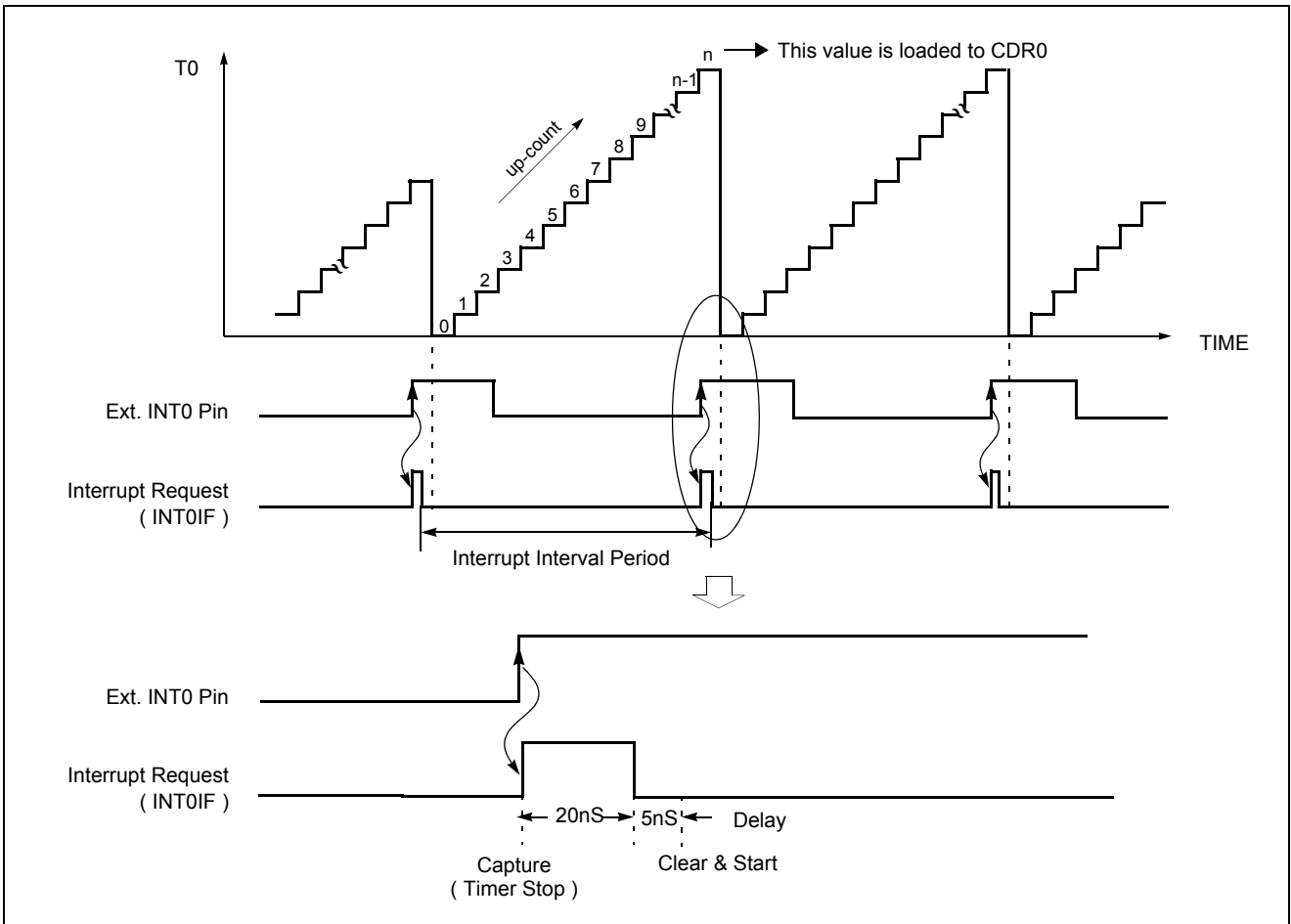


Figure 13-9 Input Capture Operation of Timer 0 Capture mode

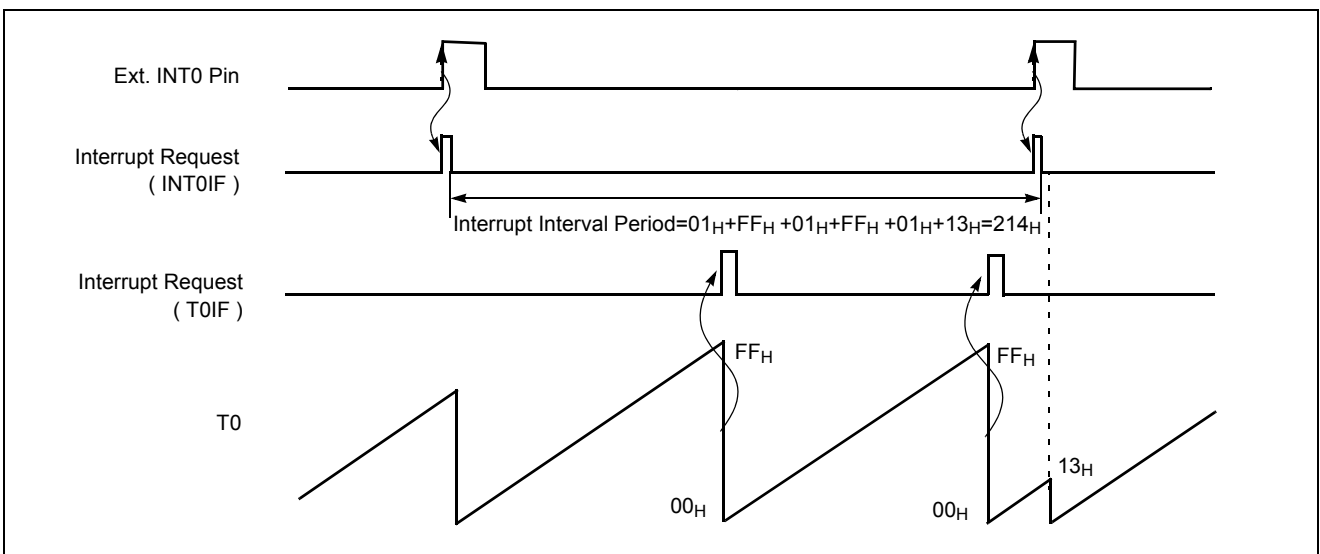
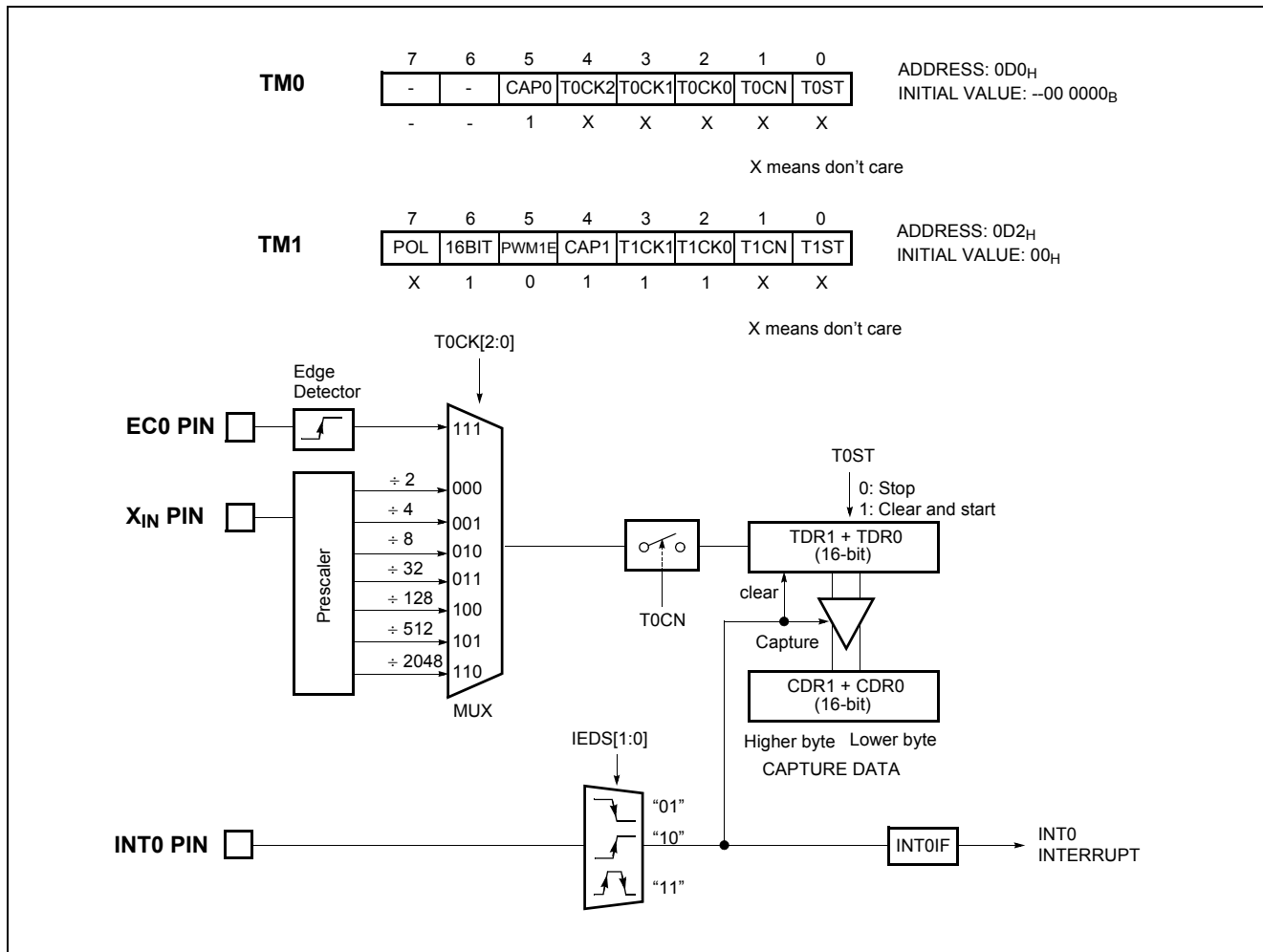


Figure 13-10 Excess Timer Overflow in Capture Mode

### 13.5 16-bit Capture Mode

16-bit capture mode is the same as 8-bit capture, except that the Timer register is being run will 16 bits. The clock source of the Timer 0 is selected either internal or external

clock by bit TOCK[2:0]. In 16-bit mode, the bits T1CK1, T1CK0, CAP1 and 16BIT of TM1 should be set to "1" respectively as shown in Figure 13-11 .



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

**Example 1:**

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

```
LDM TM0,#0000_1111B;8uS
LDM TM1,#0100_1100B;16bit Mode
LDM TDR0,#<62499 ;8uS X 62500
LDM TDR1,#>62499 ;=0.5s
SET1 TOE
EI
:
:
```

**Example 2:**

Timer0 = 16-bit event counter mode

```
LDM PSR0,#0001_0000B;EC0 Set
LDM TM0,#0001_1111B;Counter Mode
LDM TM1,#0100_1100B;16bit Mode
LDM TDR0,#<0FFH ;
```

```
LDM TDR1,#>0FFH ;
SET1 TOE
EI
:
:
```

**Example 3:**

Timer0 = 16-bit capture mode

```
LDM PSR0,#0000_0001B;INT0 set
LDM TM0,#0010_1111B;Capture Mode
LDM TM1,#0100_1100B;16bit Mode
LDM TDR0,#<0FFH ;
LDM TDR1,#>0FFH ;
LDM IEDS,#01H;Falling Edge
SET1 TOE
EI
:
:
```

### 13.6 PWM Mode

The MC80F0504/0604 has high speed PWM (Pulse Width Modulation) functions which shared with Timer1.

In PWM mode, R10 / PWM1O pin output up to a 10-bit resolution PWM output. This pin should be configured as a PWM output by setting "1" bit PWM1OE in PSR0 register.

The period of the PWM1 output is determined by the T1PPR (T1 PWM Period Register) and T1PWHR[3:2] (bit3,2 of T1 PWM High Register) and the duty of the PWM output is determined by the T1PDR (T1 PWM Duty Register) and T3PWHR[1:0] (bit1,0 of T1 PWM High Register).

The user writes the lower 8-bit period value to the T1PPR and the higher 2-bit period value to the T1PWHR[3:2]. And writes duty value to the T1PDR and the T1PWHR[1:0] same way.

The T1PDR is configured as a double buffering for glitchless PWM output. In Figure 13-12, 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)

$$\text{PWM1 Period} = [\text{PWM1HR}[3:2]\text{T1PPR} + 1] \times \text{Source Clock}$$

$$\text{PWM1 Duty} = [\text{PWM1HR}[1:0]\text{T1PDR} + 1] \times \text{Source Clock}$$

The relation of frequency and resolution is in inverse proportion. Table 13-2 shows the relation of PWM frequency vs. resolution.

If it needed more higher frequency of PWM, it should be

reduced resolution.

Resolution	Frequency		
	T1CK[1:0] = 00(250nS)	T1CK[1:0] = 01(500nS)	T1CK[1:0] = 10(2uS)
10-bit	3.9kHz	0.98kHz	0.49kHz
9-bit	7.8kHz	1.95kHz	0.97kHz
8-bit	15.6kHz	3.90kHz	1.95kHz
7-bit	31.2kHz	7.81kHz	3.90kHz

**Table 13-2 PWM Frequency vs. Resolution at 4MHz**

The bit POL of TM1 decides the polarity of duty cycle.

If the duty value is set same to the period value, the PWM output is determined by the bit POL (1: High, 0: Low). And if the duty value is set to "00H", the PWM output is determined by the bit POL (1: Low, 0: High).

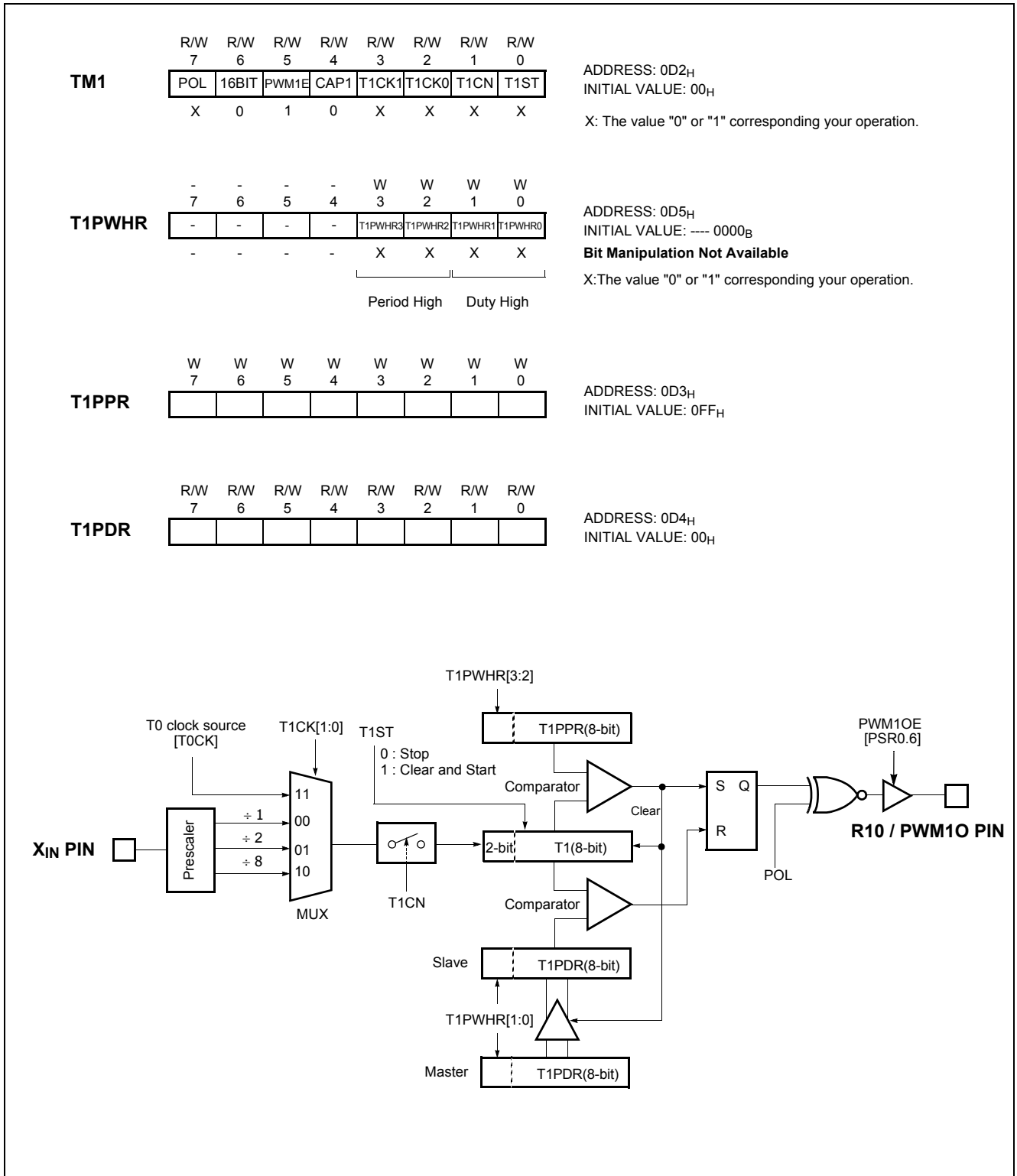
It can be changed duty value when the PWM output. However the changed duty value is output after the current period is over. And it can be maintained the duty value at present output when changed only period value shown as Figure 13-14. As it were, the absolute duty time is not changed in varying frequency. But the changed period value must greater than the duty value.

**Note:** If changing the Timer1 to PWM function, it should be stop the timer clock firstly, and then set period and duty register value. If user writes register values while timer is in operation, these register could be set with certain values.

Ex) Sample Program @4MHz 2uS

```
LDM TM1,#1010_1000b ; Set Clock & PWM1E
LDM T1PPR,#199      ; Period :400uS=2uSX(199+1)
LDM T1PDR,#99       ; Duty:200uS=2uSX(99+1)
LDM PWM1HR,00H
LDM TM1,#1010_1011b ; Start timer1
```





**Figure 13-12 PWM1 Mode**

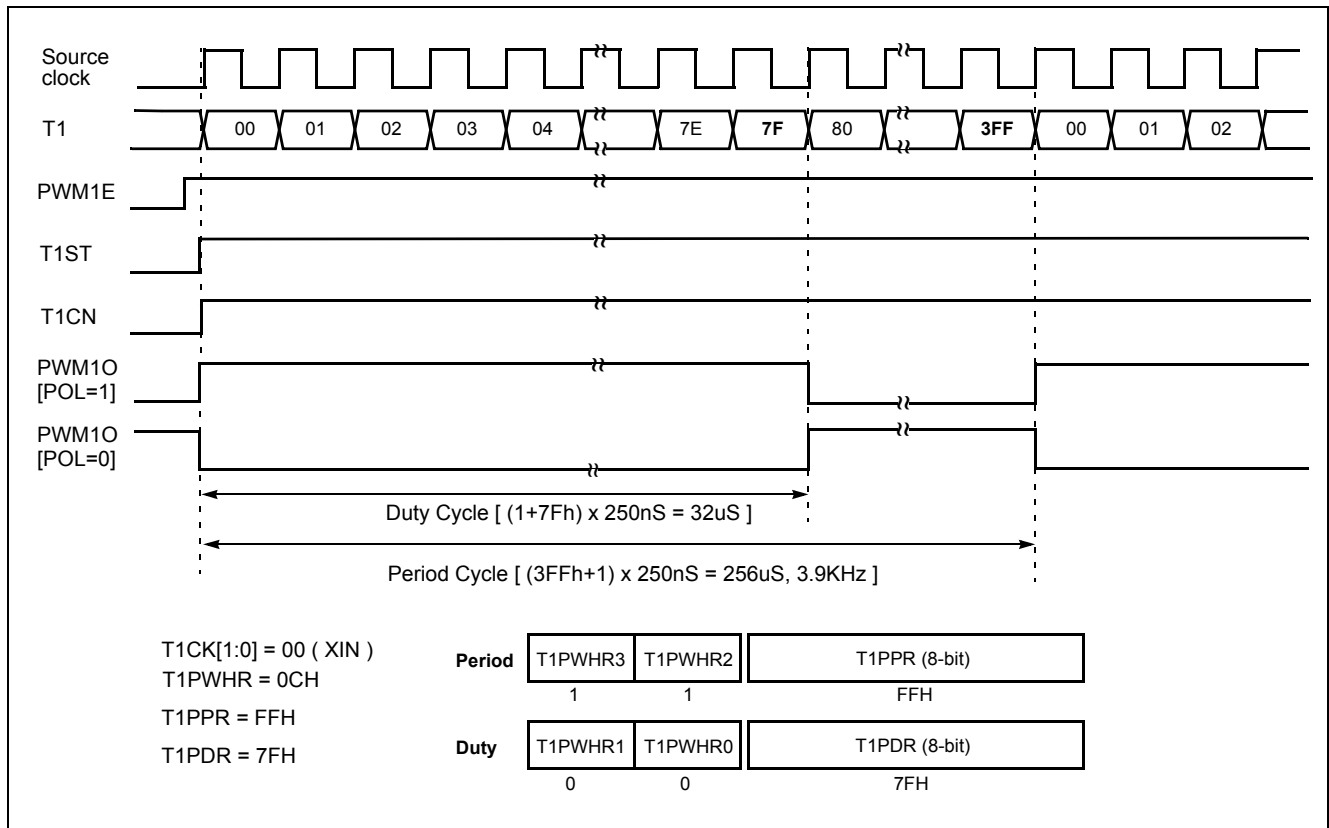


Figure 13-13 Example of PWM1 at 4MHz

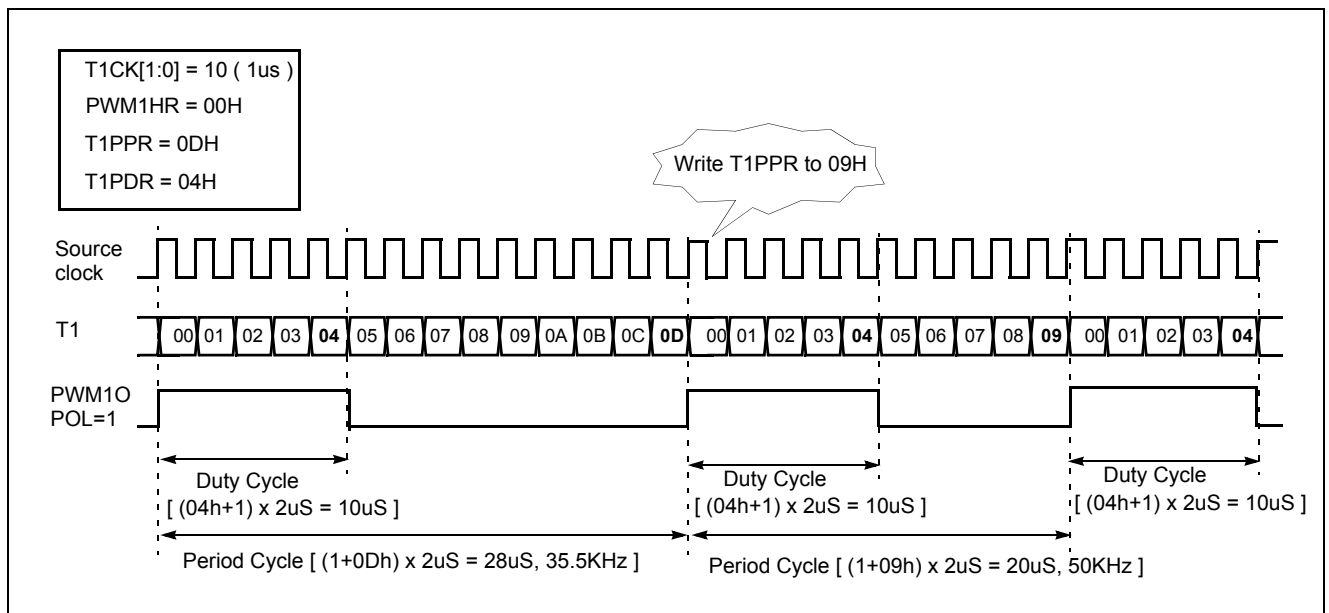


Figure 13-14 Example of Changing the PWM1 Period in Absolute Duty Cycle (@4MHz)

## 14. ANALOG TO DIGITAL CONVERTER

The analog-to-digital converter (A/D) allows conversion of an analog input signal to a corresponding 10-bit digital value. The A/D module has ten (eight for MC80F0504) analog inputs, which are multiplexed into one sample and hold. The output of the sample and hold is the input into the converter, which generates the result via successive approximation.

The analog reference voltage is selected to  $V_{DD}$  or  $AV_{ref}$  by setting of the bit  $AVREFS$  in  $PSR1$  register. If external analog reference  $AV_{ref}$  is selected, the analog input channel 0 ( $AN0$ ) should not be selected to use. Because this pin is used to an analog reference of A/D converter.

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 used as ADC clock source selection bits too. The register  $ADCM$ , shown in Figure 14-4, controls the operation of the A/D converter module. The port pins can be configured as analog inputs or digital I/O.

It is selected for the corresponding channel to be converted by setting  $ADS[3:0]$ . The A/D port is set to analog input port by  $ADEN$  and  $ADS[3:0]$  regardless of port I/O direction register. The port unselected by  $ADS[3:0]$  operates as normal port.

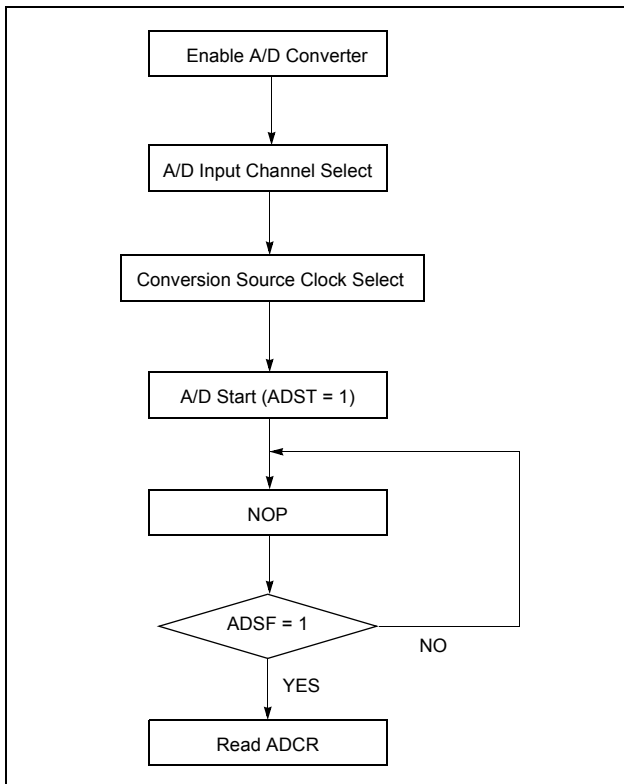


Figure 14-1 A/D Converter Operation Flow

### How to Use A/D Converter

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  $ADCRL$  contains the results of the A/D conversion. When the conversion is completed, the result is loaded into the  $ADCRH$  and  $ADCRL$ , the A/D conversion status bit  $ADSF$  is set to “1”, and the A/D interrupt flag  $ADCIF$  is set. See Figure 14-1 for operation flow.

The block diagram of the A/D module is shown in Figure 14-3. The A/D status bit  $ADSF$  is set automatically when A/D conversion is completed, cleared when A/D conversion is in process. The conversion time takes 13 times of conversion source clock. The conversion source clock should selected for the conversion time being more than  $25\mu s$ .

### A/D Converter Cautions

#### (1) Input range of $AN0 \sim AN7$ , $AN14$ and $AN15$

The input voltage of A/D input pins should be within the specification range. In particular, if a voltage above  $V_{DD}$  (or  $AV_{ref}$ ) or below  $V_{SS}$  is input (even if within the absolute maximum rating range), the conversion value for that channel can not be determinate. The conversion values of the other channels may also be affected.

#### (2) Noise countermeasures

In order to maintain 10-bit resolution, attention must be paid to noise on pins  $V_{DD}$  (or  $AV_{ref}$ ) and analog input pins ( $AN0 \sim AN7$ ,  $AN14$ ,  $AN15$ ). Since the effect increases in proportion to the output impedance of the analog input source, it is recommended in some cases that a capacitor be connected externally as shown in Figure 14-2 in order to reduce noise. The capacitance is user-selectable and appropriately determined according to the target system.

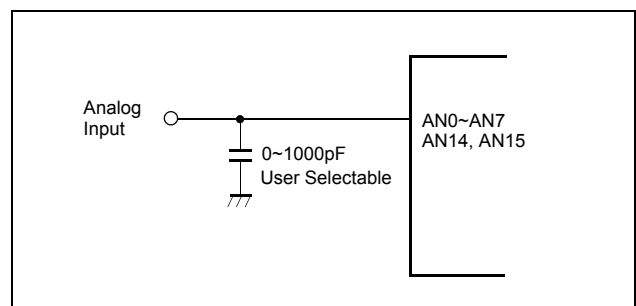


Figure 14-2 Analog Input Pin Connecting Capacitor

(3) I/O operation

The analog input pins AN0 ~ AN7, AN14 and AN15 also have function as input/output port pins. When A/D conversion is performed with any pin, be sure not to execute a PORT input instruction with the selected pin 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<sub>DD</sub> pin input impedance

A series resistor string of approximately 5KΩ is connected between the AV<sub>REF</sub> pin and the V<sub>SS</sub> pin. Therefore, if the output impedance of the analog power source is high, this will result in parallel connection to the series resistor string between the AV<sub>REF</sub> pin and the V<sub>SS</sub> pin, and there will be a large analog supply voltage error

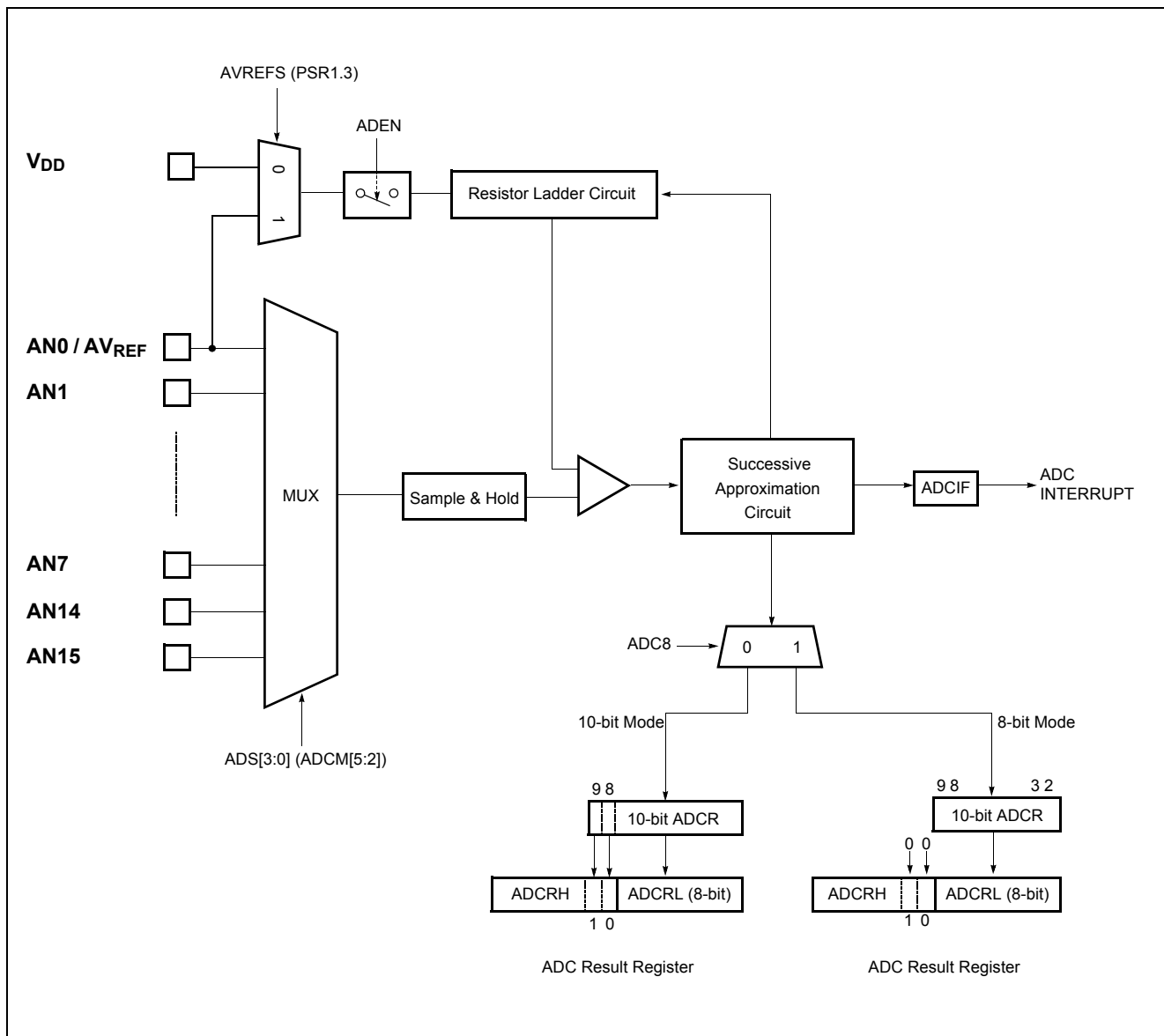
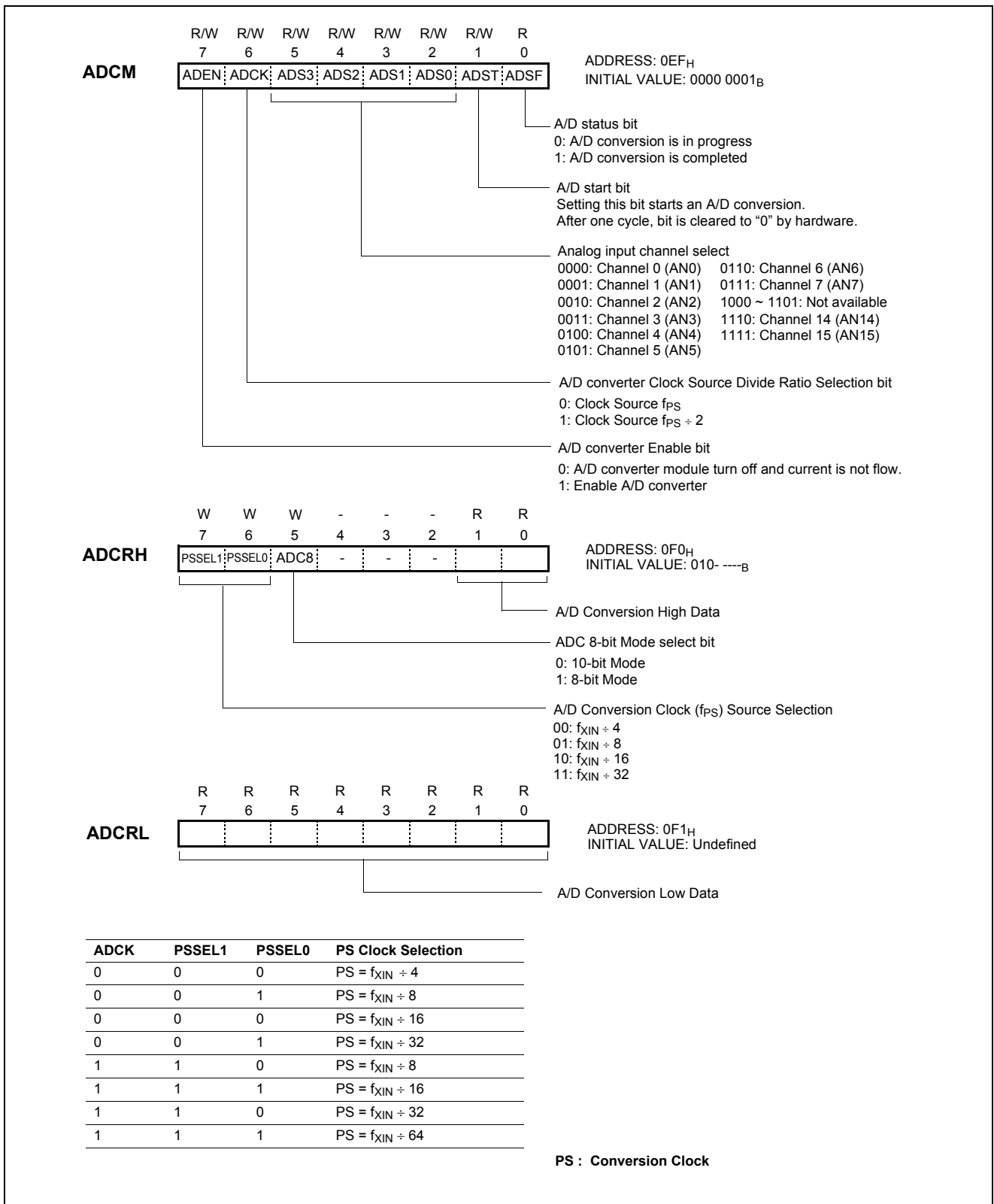


Figure 14-3 A/D Block Diagram



**Figure 14-4 A/D Converter Control & Result Register**

### 15. BUZZER FUNCTION

The buzzer driver block consists of 6-bit binary counter, buzzer register BUZR, and clock source selector. It generates square-wave which has very wide range frequency (488Hz ~ 250kHz at  $f_{XIN}=4\text{MHz}$ ) by user software.

A 50% duty pulse can be output to R12 / BUZO pin to use for piezo-electric buzzer drive. Pin R12 is assigned for output port of Buzzer driver by setting the bit 2 of PSR1(address 0F9H) to "1". For PSR1 register, refer to Figure 15-2 .

Example: 5kHz output at 4MHz.

```
LDM BUZR, #0011_0001B
LDM PSR1, #XXXX_X1XXB
```

X means don't care

The bit 0 to 5 of BUZR determines output frequency for buzzer driving.

Equation of frequency calculation is shown below.

$$f_{BUZ} = \frac{f_{XIN}}{2 \times DivideRatio \times (BUR + 1)}$$

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

The frequency of output signal is controlled by the buzzer control register BUZR. The bit 0 to bit 5 of BUZR determine output frequency for buzzer driving.

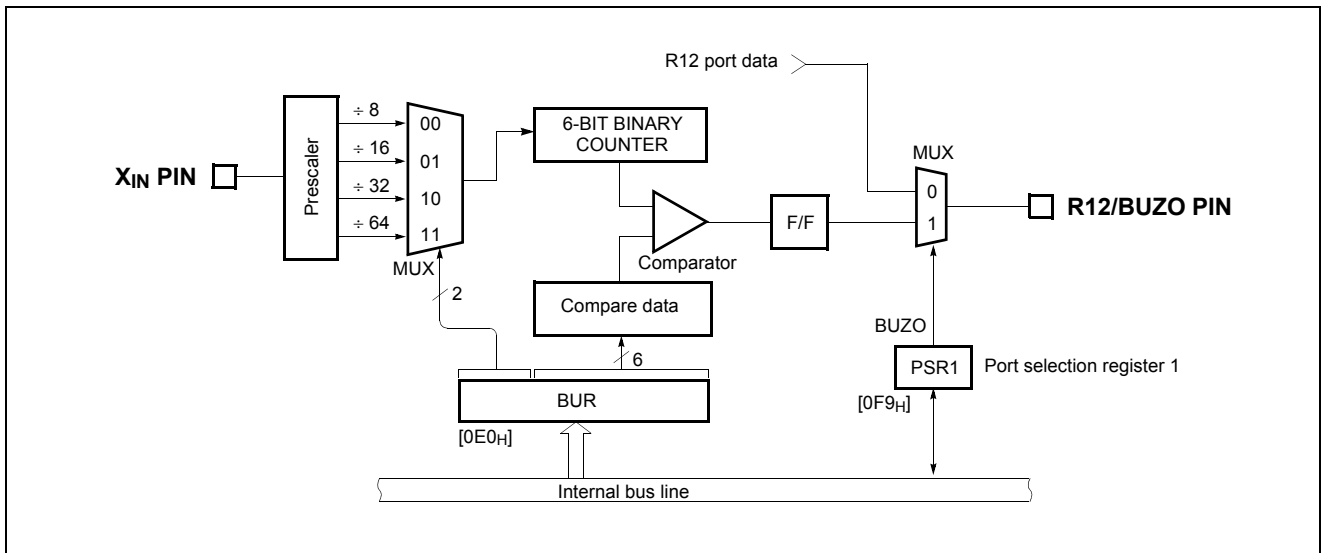


Figure 15-1 Block Diagram of Buzzer Driver

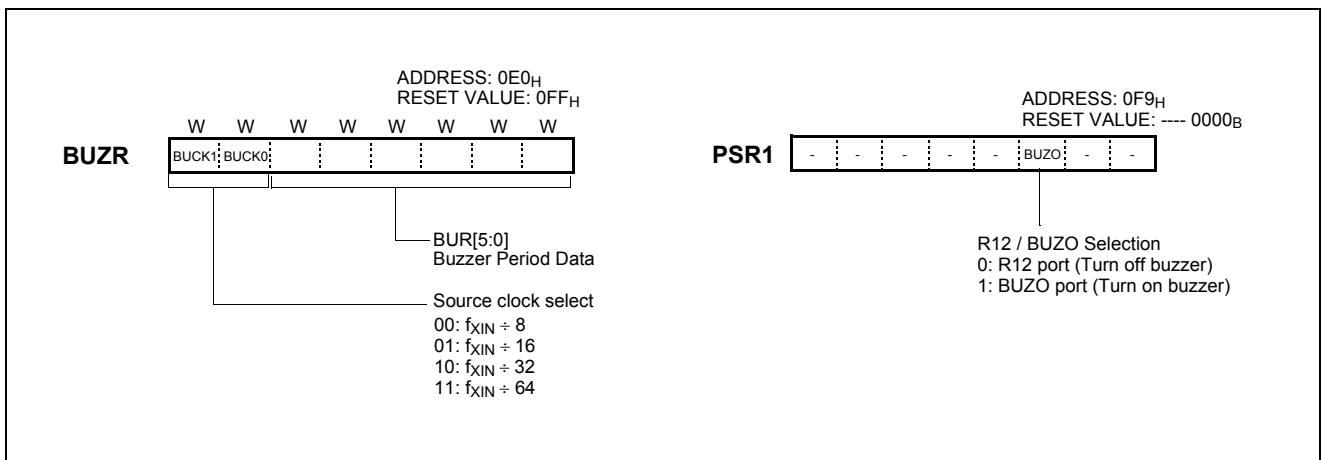


Figure 15-2 Buzzer Register & PSR1

The 6-bit counter is cleared and starts the counting by writing signal at BUZR register. It is incremental from 00<sub>H</sub> until it matches 6-bit BUR value.

When main-frequency is 4MHz, buzzer frequency is shown as below Table 15-1.

BUR [5:0]	BUR[7:6]				BUR [5:0]	BUR[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	3.907	1.953	0.977	0.488

Table 15-1 buzzer frequency (kHz unit)

## 16. INTERRUPTS

The MC80F0504/0604 interrupt circuits consist of Interrupt enable register (IENH, IENL), Interrupt request flags of IRQH, IRQL, Priority circuit, and Master enable flag ("I" flag of PSW). Fifteen interrupt sources are provided. The configuration of interrupt circuit is shown in Figure 16-1 and interrupt priority is shown in Table 16-1.

The External Interrupts INT0 and INT1 each can be transition-activated (1-to-0 or 0-to-1 transition) by selection IEDS register.

The flags that actually generate these interrupts are bit INT0IF and INT1IF in register IRQH. When an external interrupt is generated, the generated flag is cleared by the hardware when the service routine is vectored to only if the

interrupt was transition-activated.

The Timer 0 and Timer 1 Interrupts are generated by T0IF, T1IF and T1IF which is set by a match in their respective timer/counter register.

The Basic Interval Timer Interrupt is generated by BITIF which is set by an overflow in the timer register.

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

The Watchdog timer is generated by WDTIF and WTIF which is set by a match in Watchdog timer register.

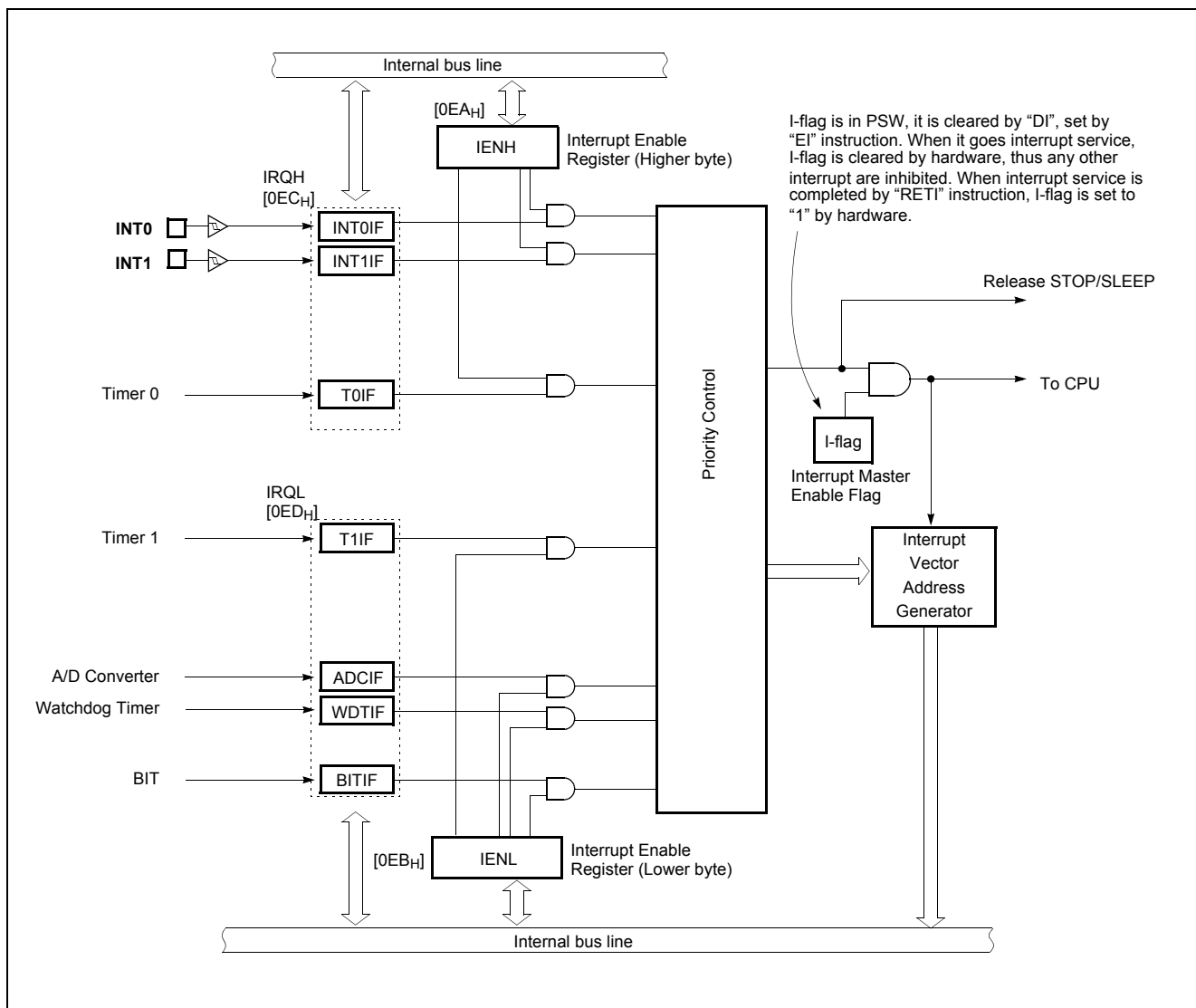


Figure 16-1 Block Diagram of Interrupt



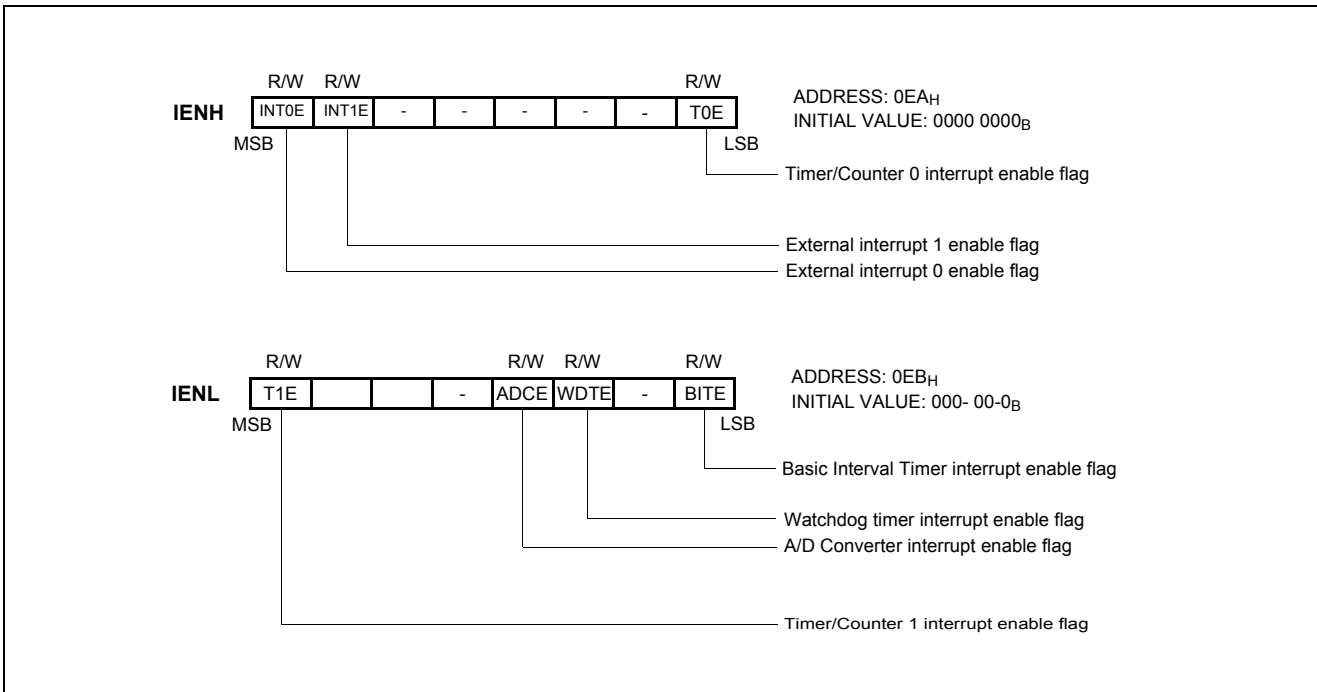
The Basic Interval Timer Interrupt is generated by BITIF which is set by an overflow in the timer counter register.

The interrupts are controlled by the interrupt master enable flag I-flag (bit 2 of PSW on Figure 8-3 ), the interrupt enable register (IENH, IENL), and the interrupt request flags (in IRQH and IRQL) except Power-on reset and software BRK interrupt. The Table 16-1 shows the Interrupt priority.

Vector addresses are shown in Figure 8-6 . Interrupt enable registers are shown in Figure 16-2 . These registers are composed of interrupt enable flags of each interrupt source and these flags determines whether an interrupt will be accepted or not. When enable flag 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.

Reset/Interrupt	Symbol	Priority
Hardware Reset	RESET	1
External Interrupt 0	INT0	2
External Interrupt 1	INT1	3
Timer/Counter 0	Timer 0	4
Timer/Counter 1	Timer 1	5
ADC Interrupt	ADC	6
Watchdog Timer	WDT	7
Basic Interval Timer	BIT	8

**Table 16-1 Interrupt Priority**



**Figure 16-2 Interrupt Enable Flag Register**

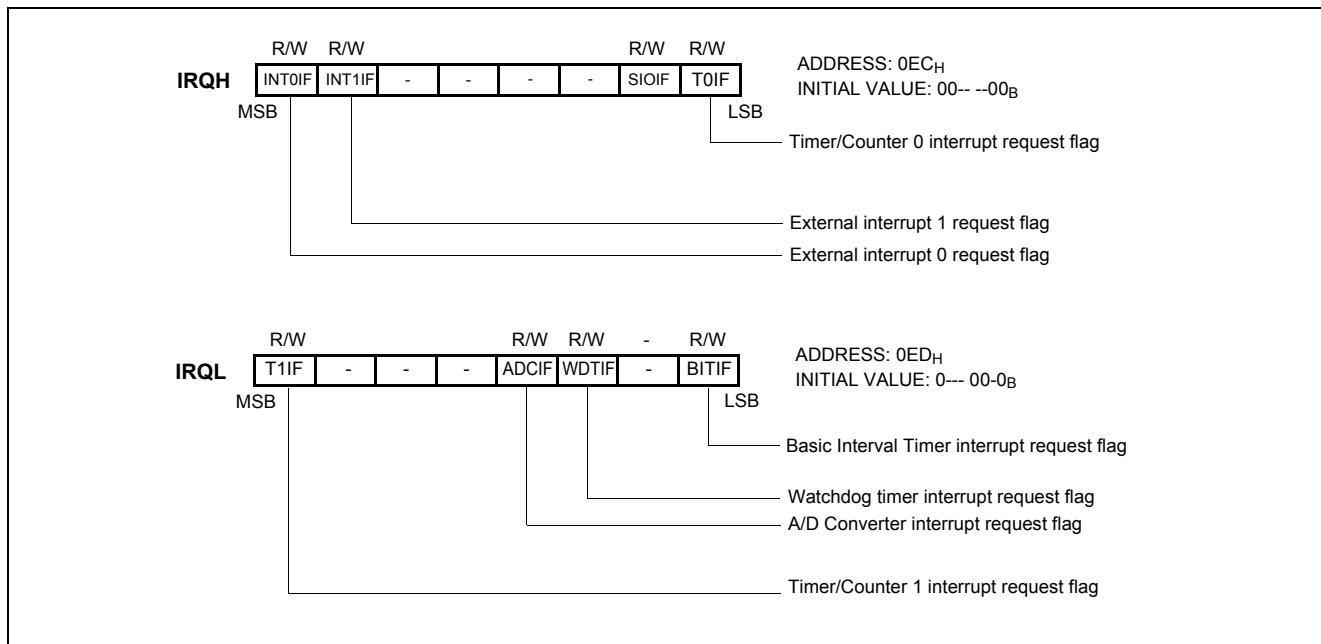


Figure 16-3 Interrupt Request Flag Register

## 16.1 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 cycles of  $f_{XIN}$  ( $2\mu\text{s}$  at  $f_{XIN}=4\text{MHz}$ ) after the completion of the

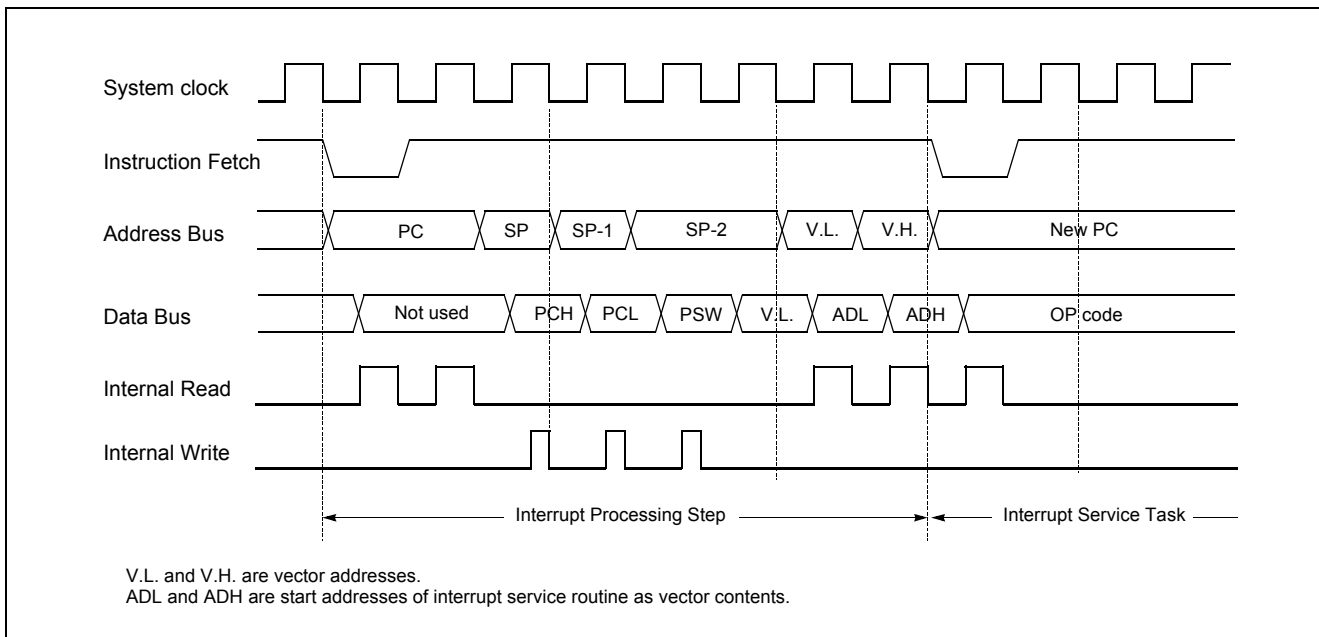
current instruction execution. The interrupt service task is terminated upon execution of an interrupt return instruction [RETI].

### 16.1.1 Interrupt acceptance

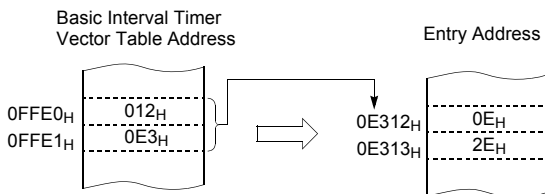
1. The interrupt master enable flag (I-flag) is cleared to “0” to temporarily disable the acceptance of any following maskable interrupts. When a non-maskable interrupt is accepted, the acceptance of any following interrupts is temporarily disabled.
2. The contents of the program counter (return address) and the program status word are saved (pushed) onto the

stack area. The stack pointer decreases 3 times.

3. 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.
4. The instruction stored at the entry address of the interrupt service program is executed.



**Figure 16-4 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.

A 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.

### 16.1.2 Clearing Interrupt Request Flag

The Interrupt Request flag may not cleared itself during interrupt acceptance processing. After interrupt acceptance, it should be cleared as shown in interrupt service routine.

**Note:** The MC80F0504 and HMS87C1102A is very similar in function, but the interrupt processing method is different. When replacing the HMS87C1102A to MC80F0504, clearing interrupt request flag should be added.

#### Example: Clearing Interrupt Request Flag

```
T1_INT:   CLR1   T1IF  ;CLEAR T1 REQUEST
          interrupt processing
          RETI    ;RETURN
```

### 16.1.3 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. These registers are saved by the software if necessary. 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.

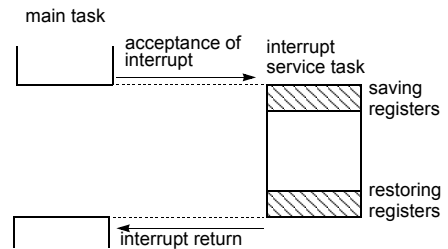
**Example: Register save using push and pop instructions**

```

INTxx: CLR1    INTxxIF  ;CLEAR REQUEST.
        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 register save/restore using push and pop instructions;

**16.2 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 16-5 .

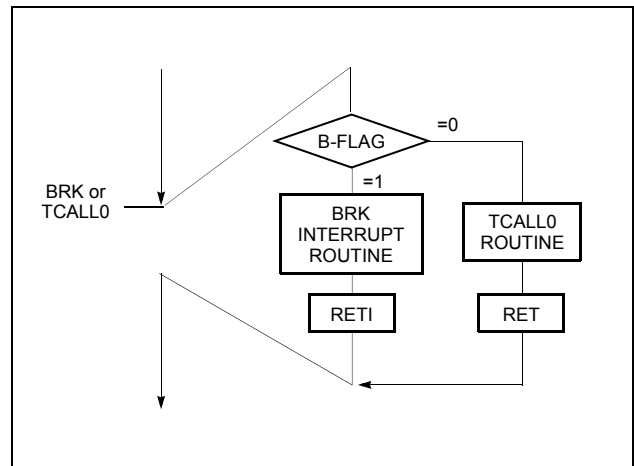
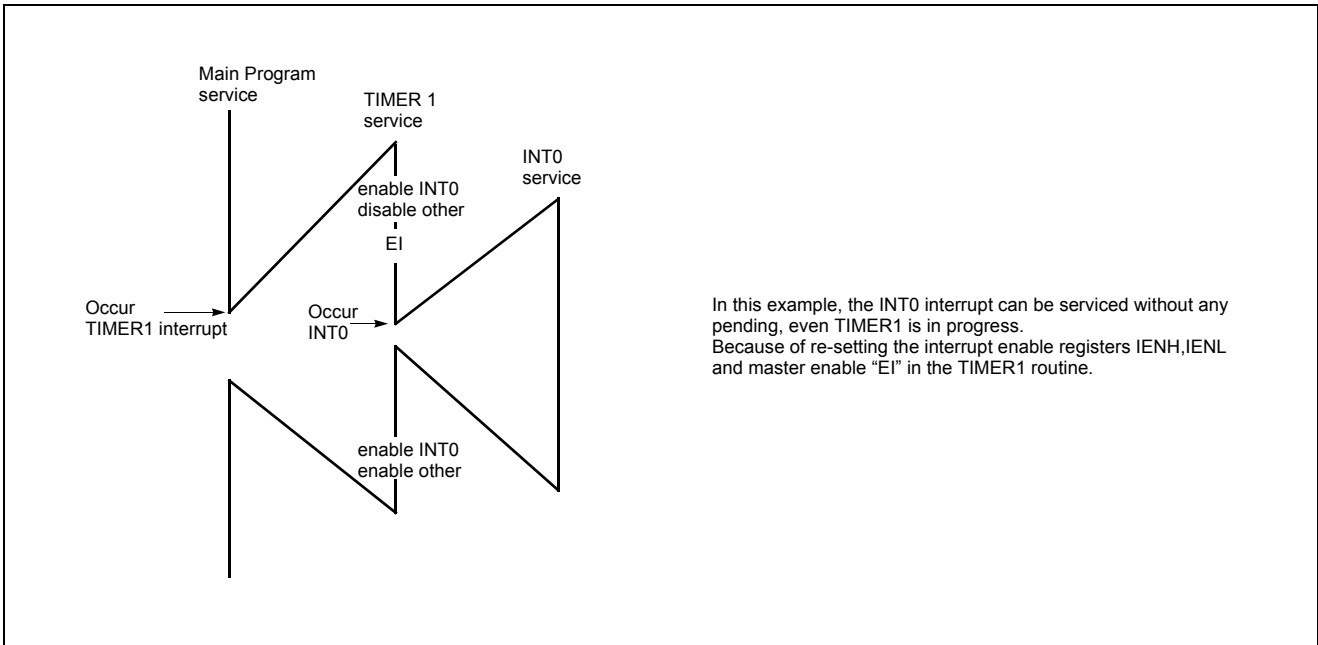


Figure 16-5 Execution of BRK/TCALL0

**16.3 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.



**Figure 16-6 Execution of Multi Interrupt**

**Example:** During Timer1 interrupt is in progress, INTO interrupt serviced without any suspend.

```

TIMER1:  PUSH  A
         PUSH  X
         PUSH  Y
         LDM   IENH, #80H ; Enable INTO only
         LDM   IENL, #0   ; Disable other int.
         EI    ; Enable Interrupt
         :
         :
         :
         :
         LDM   IENH, #0FFH ; Enable all interrupts
         LDM   IENL, #0FFH
         POP   Y
         POP   X
         POP   A
         RETI
    
```

### 16.4 External Interrupt

The external interrupt on INT0 and INT1 pins are edge triggered depending on the edge selection register IEDS (address 0EEH) as shown in Figure 16-7 .

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

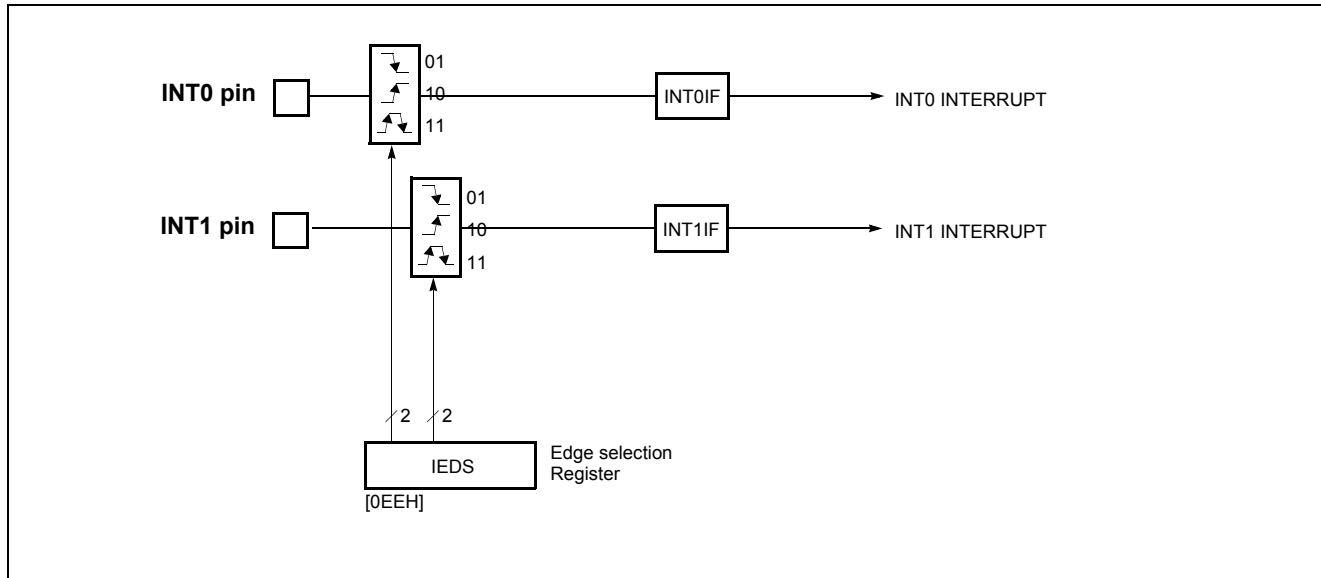


Figure 16-7 External Interrupt Block Diagram

INT0 and INT1 are multiplexed with general I/O ports (R11, R12). To use as an external interrupt pin, the bit of port selection register PSR0 should be set to “1” correspondingly.

**Example:** To use as an INT0 and INT1

```

:
;**** Set external interrupt port as pull-up state.
LDM PU1, #0000_0110B
;
;**** Set port as an external interrupt port
LDM PSR0, #0000_0011B
;
;**** Set Falling-edge Detection
LDM IEDS, #0000_0101B
:
    
```

#### Response Time

The INT0 and INT1 edge are latched into INT0IF and INT1IF 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 minimum 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.

Figure 16-8 shows interrupt response timings.

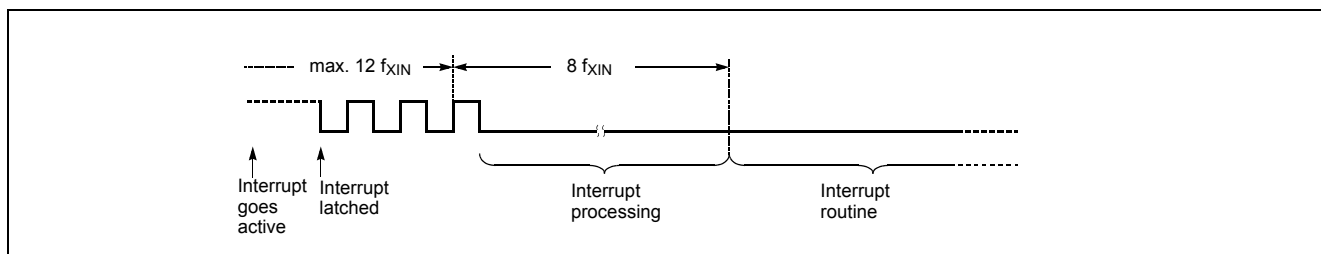


Figure 16-8 Interrupt Response Timing Diagram

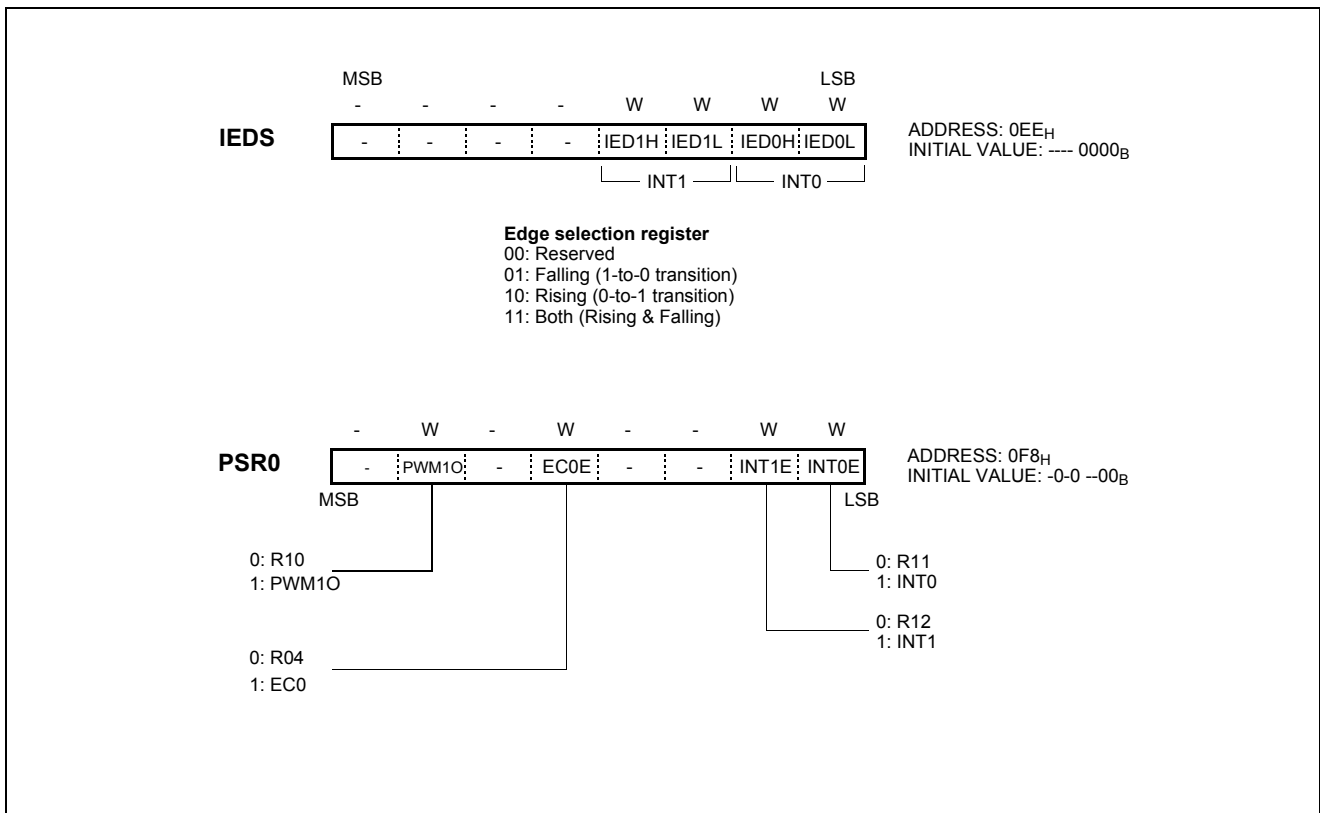


Figure 16-9 IEDS register and Port Selection Register PSR0

## 17. POWER SAVING OPERATION

The MC80F0504/0604 has two power-down modes. In power-down mode, power consumption is reduced considerably. For applications where power consumption is a critical factor, device provides two kinds of power saving functions, STOP mode and SLEEP mode. Table 17-1

shows the status of each Power Saving Mode. SLEEP mode is entered by the SSCR register to “0Fh”, and STOP mode is entered by STOP instruction after the SSCR register to “5Ah”.

### 17.1 Sleep Mode

In this mode, the internal oscillation circuits remain active. Oscillation continues and peripherals are operate normally but CPU stops. Movement of all peripherals is shown in Table 17-1. SLEEP mode is entered by setting the SSCR register to “0Fh”. It is released by Reset or interrupt. To be

released by interrupt, interrupt should be enabled before SLEEP mode.

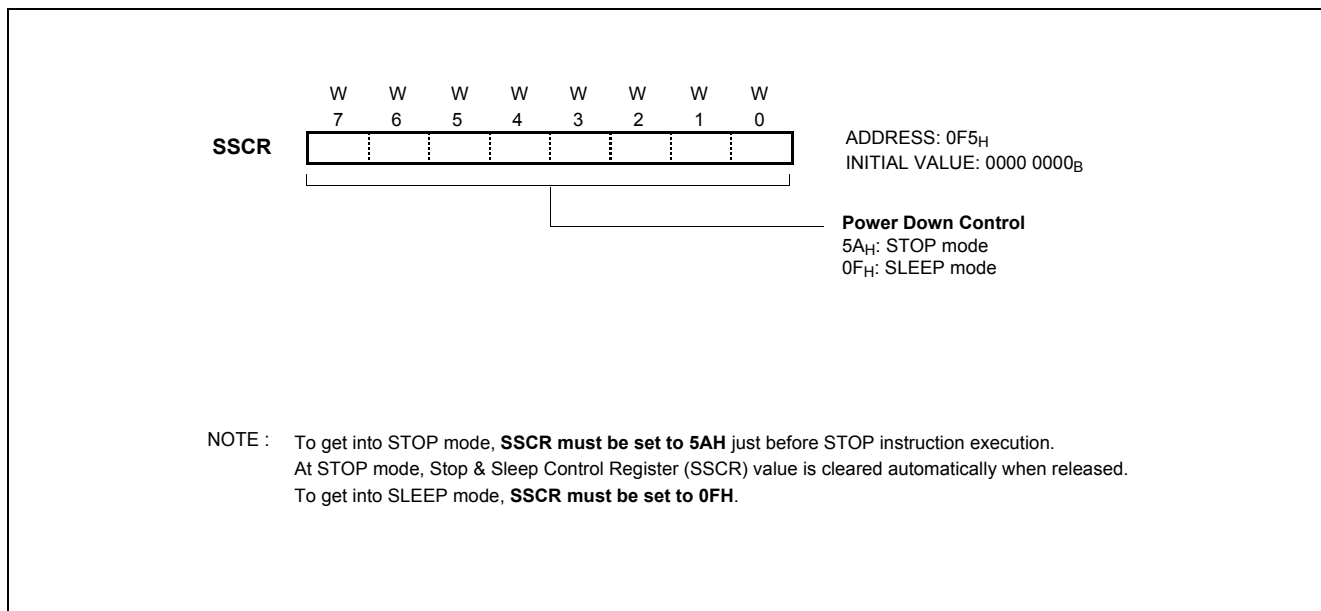


Figure 17-1 STOP and SLEEP Control Register

#### Release the SLEEP mode

The exit from SLEEP mode is hardware reset or all interrupts. Reset re-defines all the Control registers but does not change the on-chip RAM. Interrupts allow both on-chip RAM and Control registers to retain their values.

If I-flag = 1, the normal interrupt response takes place. If I-flag = 0, the chip will resume execution starting with the instruction following the SLEEP instruction. It will not vector to interrupt service routine. (refer to Figure 17-4 )

When exit from SLEEP mode by reset, enough oscillation

stabilizing time is required to normal operation. Figure 17-3 shows the timing diagram. When released from the SLEEP mode, the Basic interval timer is activated on wake-up. It is increased from 00H until FFH. The count overflow is set to start normal operation. Therefore, before SLEEP instruction, user must be set its relevant prescaler divide ratio to have long enough time (more than 20msec). This guarantees that oscillator has started and stabilized. By interrupts, exit from SLEEP mode is shown in Figure 17-2 . By reset, exit from SLEEP mode is shown in Figure 17-3 .



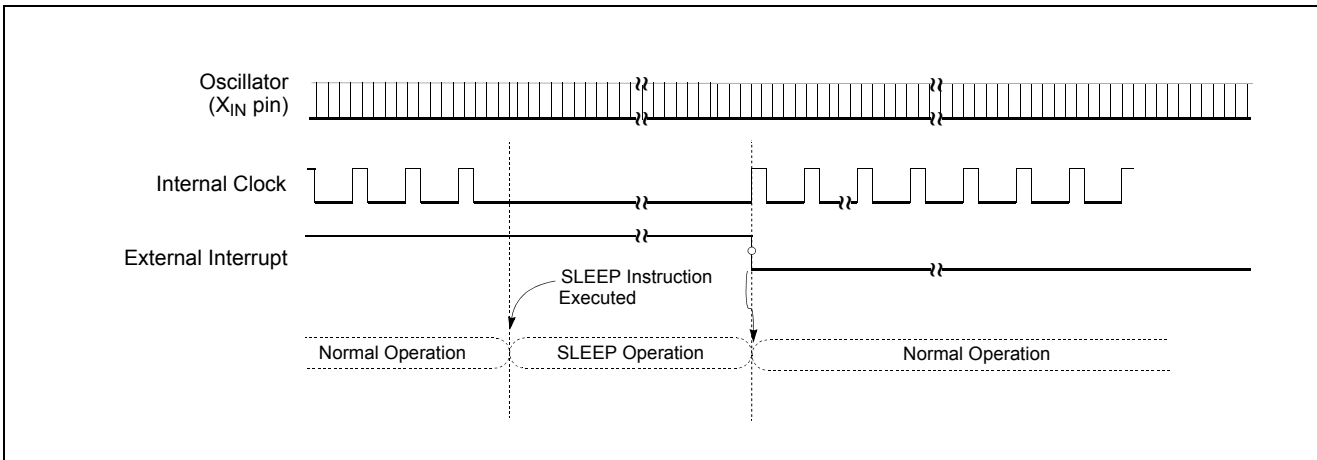


Figure 17-2 SLEEP Mode Release Timing by External Interrupt

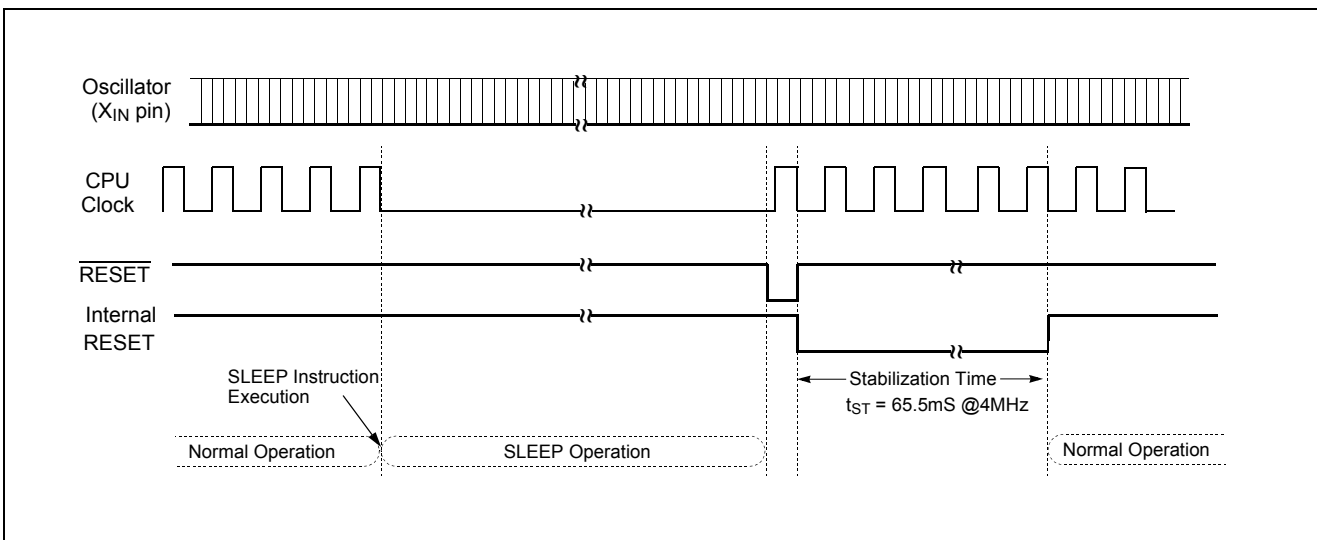


Figure 17-3 Timing of SLEEP Mode Release by Reset

## 17.2 Stop Mode

In the Stop mode, the main oscillator, system clock and peripheral clock is stopped, but RC-oscillated watchdog timer continue to operate. With the clock frozen, all functions are stopped, but the on-chip RAM and Control registers are held. The port pins out the values held by their respective port data register, port direction registers. Oscillator stops and the systems internal operations are all held up.

- The states of the RAM, registers, and latches valid immediately before the system is put in the STOP state are all held.
- The program counter stop the address of the instruction to be executed after the instruction

"STOP" which starts the STOP operating mode.

**Note:** The Stop mode is activated by execution of STOP instruction after setting the SSCR to "5AH". (This register should be written by byte operation. If this register is set by bit manipulation instruction, for example "set1" or "clr1" instruction, it may occur undesired operation)

In the Stop mode of operation,  $V_{DD}$  can be reduced to minimize power consumption. Care must be taken, however, to ensure 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.

**Note:** After STOP instruction, at least two or more NOP instruction should be written.

```
Ex)  LDM CKCTLR,#0FH ;more than 20ms
      LDM SSCR,#5AH
      STOP
      NOP ;for stabilization time
      NOP ;for stabilization time
```

In the STOP operation, the dissipation of the power 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 gets higher than the power voltage level (by approximately 0.3 to 0.5V), 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 to fix the level by pull-up or other means.

Peripheral	STOP Mode	SLEEP Mode
CPU	Stop	Stop
RAM	Retain	Retain
Basic Interval Timer	Halted	Operates Continuously
Watchdog Timer	Stop (Only operates in RC-WDT mode)	Operates Continuously
Timer/Counter	Halted (Only when the event counter mode is enabled, timer operates normally)	Operates Continuously
ADC	Stop	Stop
Buzzer	Stop	Operates Continuously
Oscillator	Stop ( $X_{IN}=L$ , $X_{OUT}=H$ )	Oscillation
I/O Ports	Retain	Retain
Control Registers	Retain	Retain
Internal Circuit	Stop mode	Sleep mode
Prescaler	Retain	Active
Address Data Bus	Retain	Retain
Release Source	Reset, Timer(EC0), Watchdog Timer (RC-WDT mode), External Interrupt	Reset, All Interrupts

**Table 17-1 Peripheral Operation During Power Saving Mode**

### Release the STOP mode

The source for exit from STOP mode is hardware reset, external interrupt, Timer(EC0), Watch Timer, WDT. Reset re-defines 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.

If I-flag = 1, the normal interrupt response takes place. If I-flag = 0, the chip will resume execution starting with the instruction following the STOP instruction. It will not vector to interrupt service routine. (refer to Figure 17-4)

When exit from Stop mode by external interrupt, enough oscillation stabilizing time is required to normal operation. Figure 17-5 shows the timing diagram. When released from the Stop mode, the Basic interval timer is activated on wake-up. It is increased from 00<sub>H</sub> until FF<sub>H</sub>. The count overflow is set to start normal operation. Therefore, before STOP instruction, user must be set its relevant prescaler divide ratio to have long enough time (more than 20msec). This guarantees that oscillator has started and stabilized.

By reset, exit from Stop mode is shown in Figure 17-6 .

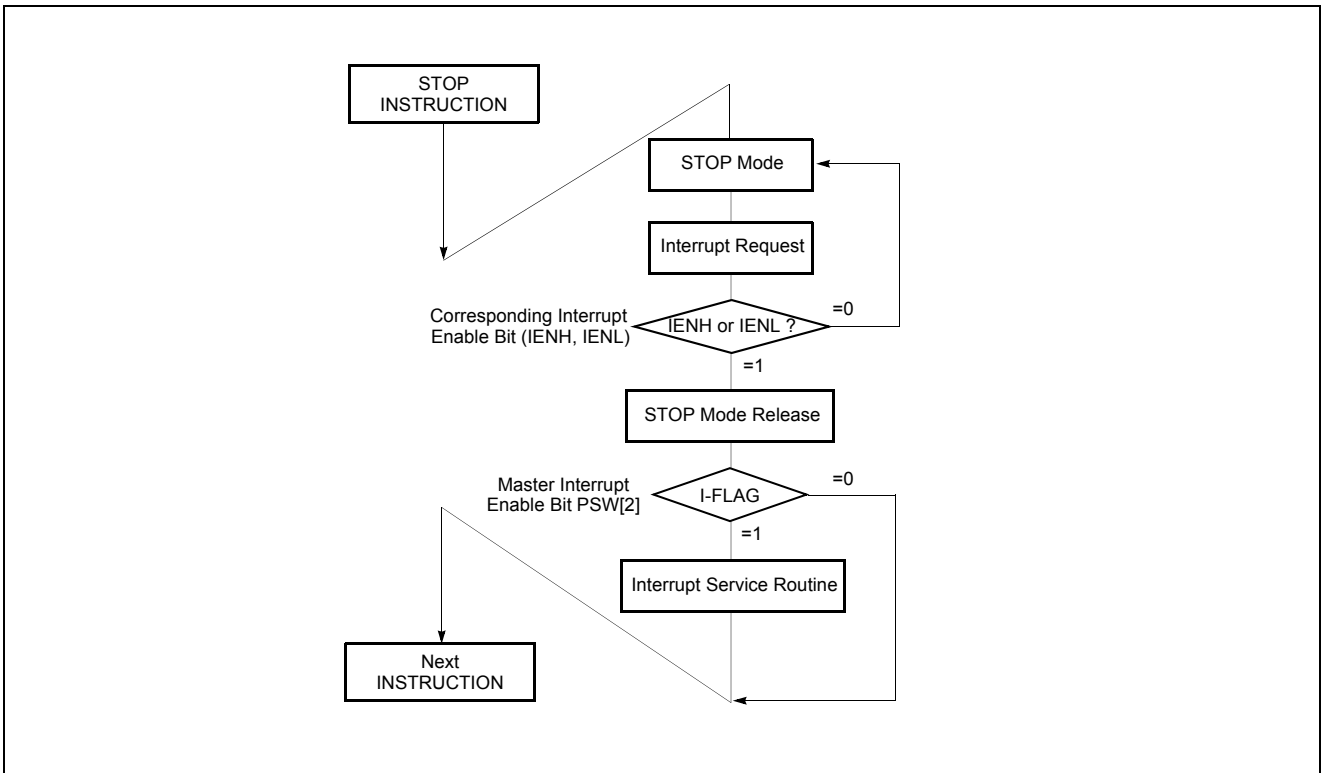


Figure 17-4 STOP Releasing Flow by Interrupts

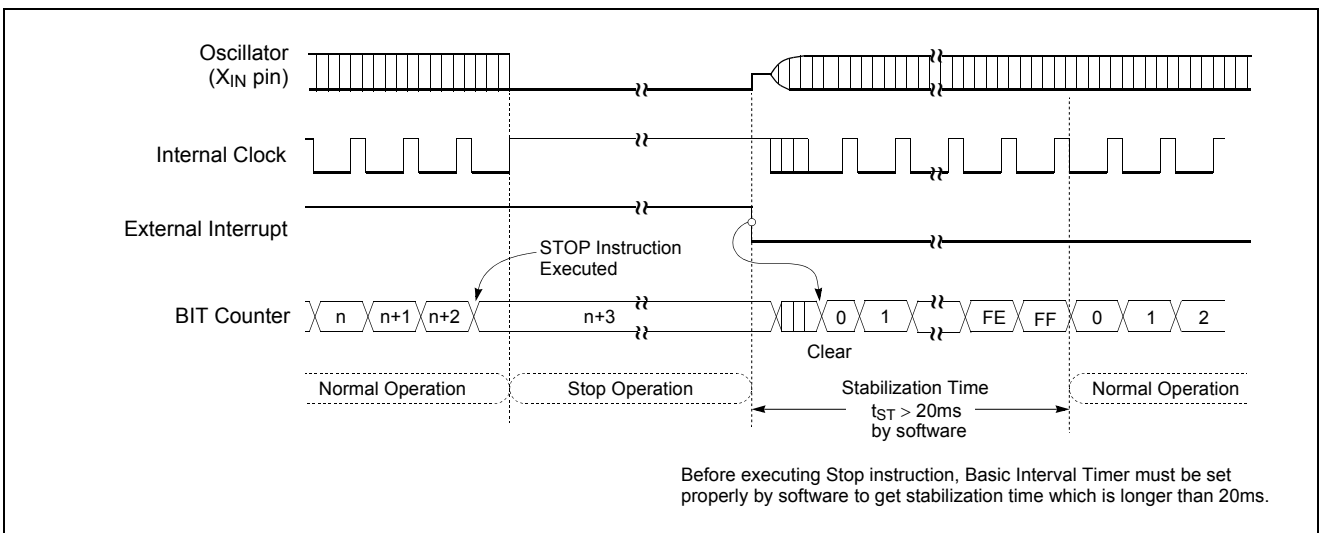


Figure 17-5 STOP Mode Release Timing by External Interrupt

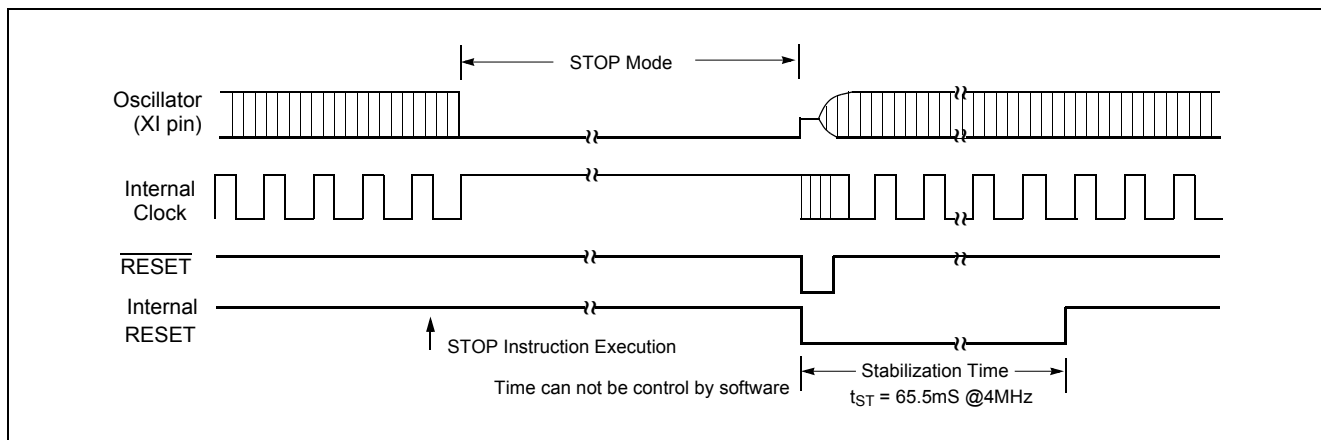


Figure 17-6 Timing of STOP Mode Release by Reset

### 17.3 Stop Mode at Internal RC-Oscillated Watchdog Timer Mode

In the Internal RC-Oscillated Watchdog Timer mode, the on-chip oscillator is stopped. But internal RC oscillation circuit is oscillated in this mode. The on-chip RAM and Control registers are held. The port pins out the values held by their respective port data register, port direction registers.

The Internal RC-Oscillated Watchdog Timer mode is activated by execution of STOP instruction after setting the bit RCWDT of CKCTLR to "1". (This register should be written by byte operation. If this register is set by bit manipulation instruction, for example "set1" or "clr1" instruction, it may be undesired operation)

**Note:** Caution: After STOP instruction, at least two or more NOP instruction should be written

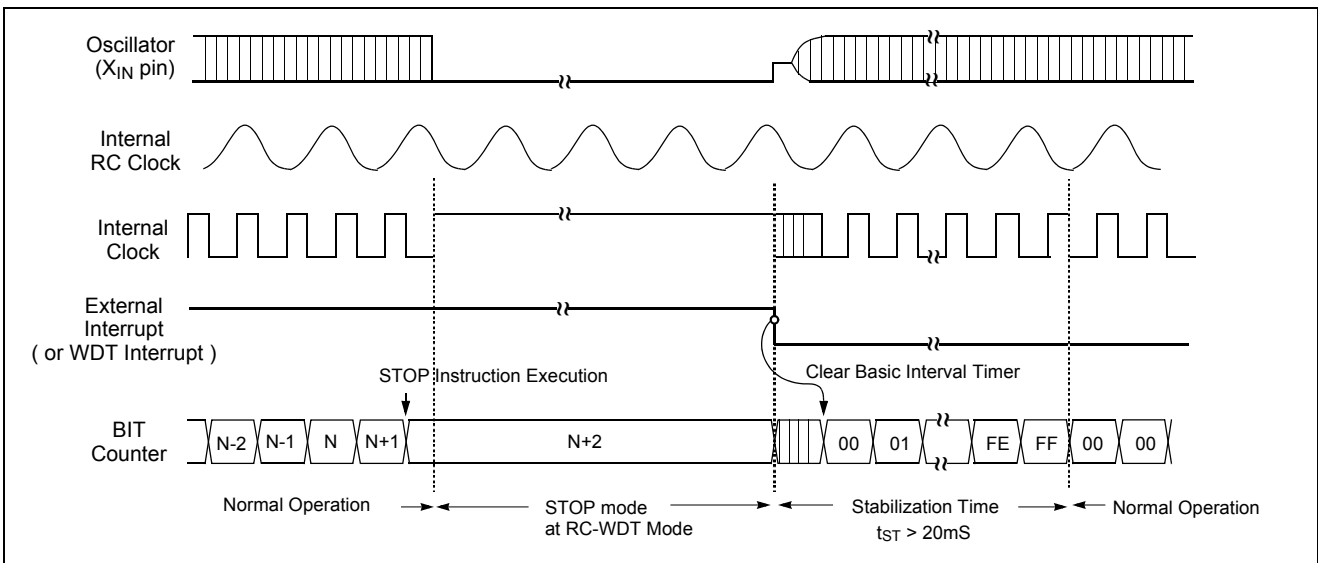
```
Ex) LDM WDTR,#1111_1111B
     LDM CKCTLR,#0010_1110B
     LDM SSCR,#0101_1010B
     STOP
     NOP ;for stabilization time
     NOP ;for stabilization time
```

The exit from Internal RC-Oscillated Watchdog Timer mode is hardware reset or external interrupt or watchdog timer interrupt (at RC-watchdog timer mode). Reset re-de-

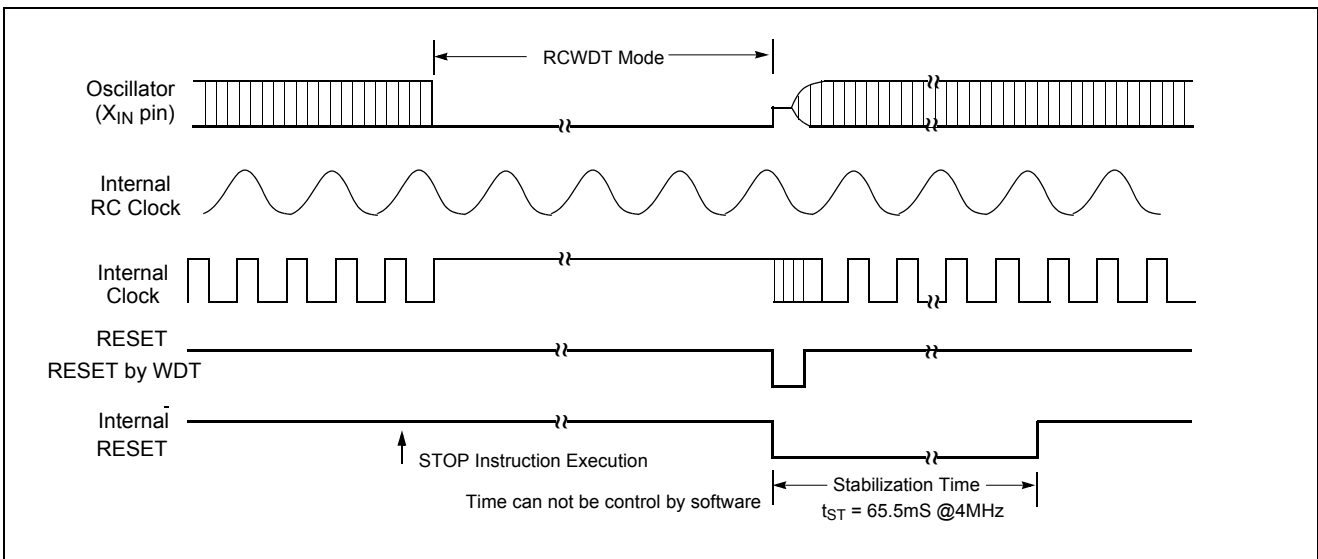
fines all the Control registers but does not change the on-chip RAM. External interrupts allow both on-chip RAM and Control registers to retain their values.

If I-flag = 1, the normal interrupt response takes place. In this case, if the bit WDTON of CKCTLR is set to "0" and the bit WDTE of IENH is set to "1", the device will execute the watchdog timer interrupt service routine (Figure 8-6). However, if the bit WDTON of CKCTLR is set to "1", the device will generate the internal Reset signal and execute the reset processing (Figure 17-8). If I-flag = 0, the chip will resume execution starting with the instruction following the STOP instruction. It will not vector to interrupt service routine. (refer to Figure 17-4)

When exit from Stop mode at Internal RC-Oscillated Watchdog Timer mode by external interrupt, the oscillation stabilization time is required to normal operation. Figure 17-7 shows the timing diagram. When release the Internal RC-Oscillated Watchdog Timer mode, the basic interval timer is activated on wake-up. It is increased from 00<sub>H</sub> until FF<sub>H</sub>. The count overflow is set to start normal operation. Therefore, before STOP instruction, user must be set its relevant prescaler divide ratio to have long enough time (more than 20msec). This guarantees that oscillator has started and stabilized. By reset, exit from internal RC-Oscillated Watchdog Timer mode is shown in Figure 17-8



**Figure 17-7 Stop Mode Release at Internal RC-WDT Mode by External Interrupt or WDT Interrupt**



**Figure 17-8 Internal RC-WDT Mode Releasing by Reset**

### 17.4 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.

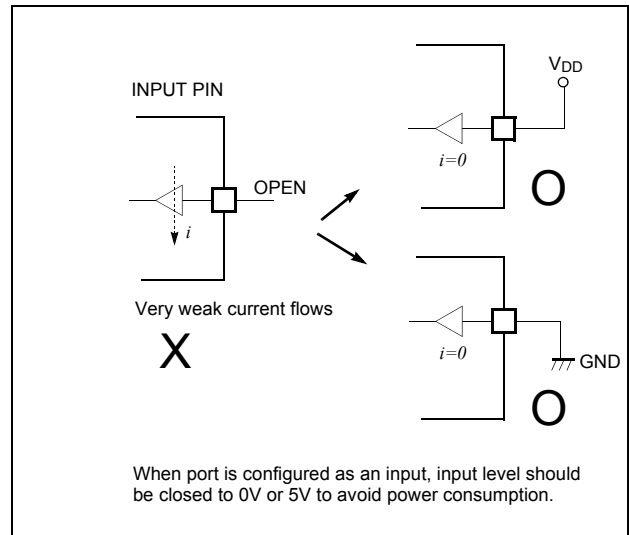
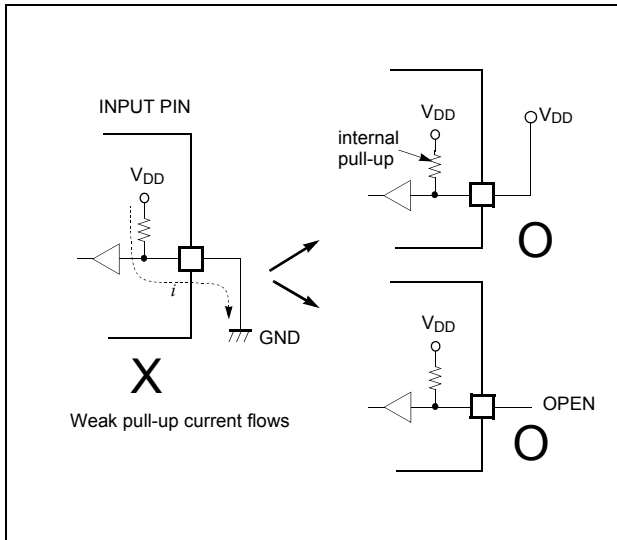


Figure 17-9 Application Example of Unused Input Port

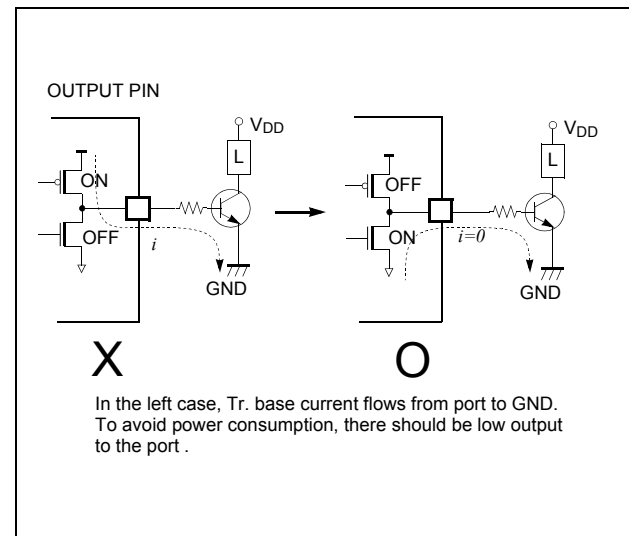
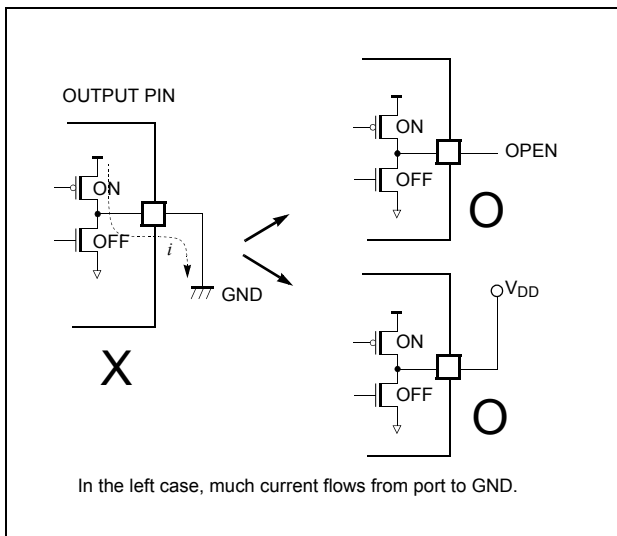


Figure 17-10 Application Example of Unused Output Port

**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 in order that current flow through port doesn't exist.

First consider the port 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 uncertain 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. The port setting to High or Low is decided by 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 register, it is set to low.

### 18. RESET

The MC80F0504/0604 supports various kinds of reset as below.

- On-Chip Power-On Reset (POR)
- RESET (external reset circuitry)
- Watchdog Timer Timeout Reset
- Power-Fail Detection (PFD) Reset
- Address Fail Reset

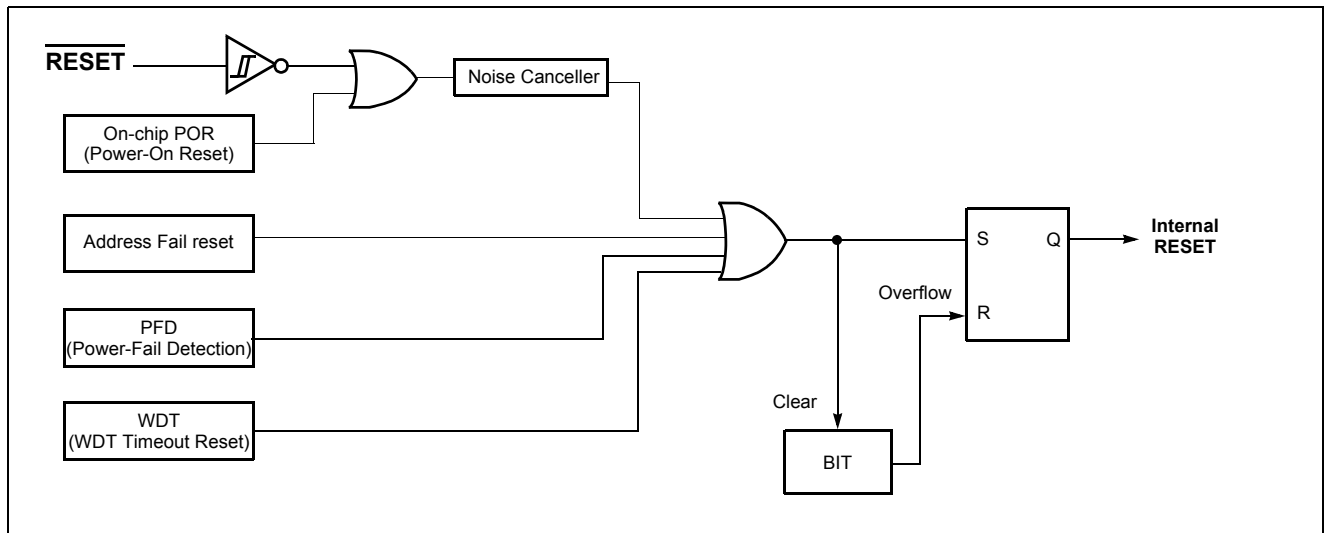


Figure 18-1 RESET Block Diagram

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 R35 by setting “POR” and “R35EN” bit Config-

uration Area(20FFH) in the Flash programming. When the device starts normal operation, its operating parameters (voltage, frequency, temperature...etc) must be met.

Table 18-1 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
Operation mode	Main-frequency clock

On-chip Hardware	Initial Value
Peripheral clock	Off
Watchdog timer	Disable
Control registers	Refer to Table 8-1 on page 28
Power fail detector	Disable

Table 18-1 Initializing Internal Status by Reset Action

The reset input is the RESET pin, which is the input to a Schmitt Trigger. A reset is accomplished by holding the RESET pin 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 4 MHz) add with 7 oscillator periods are required to start execution as shown in Figure 18-2 .

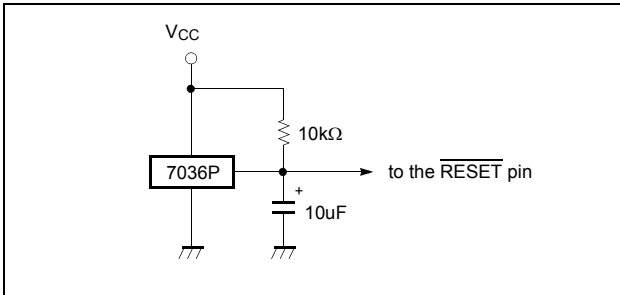
Internal RAM is not affected by reset. When V<sub>DD</sub> is turned

on, the RAM content is indeterminate. Therefore, this RAM should be initialized before read or tested it.

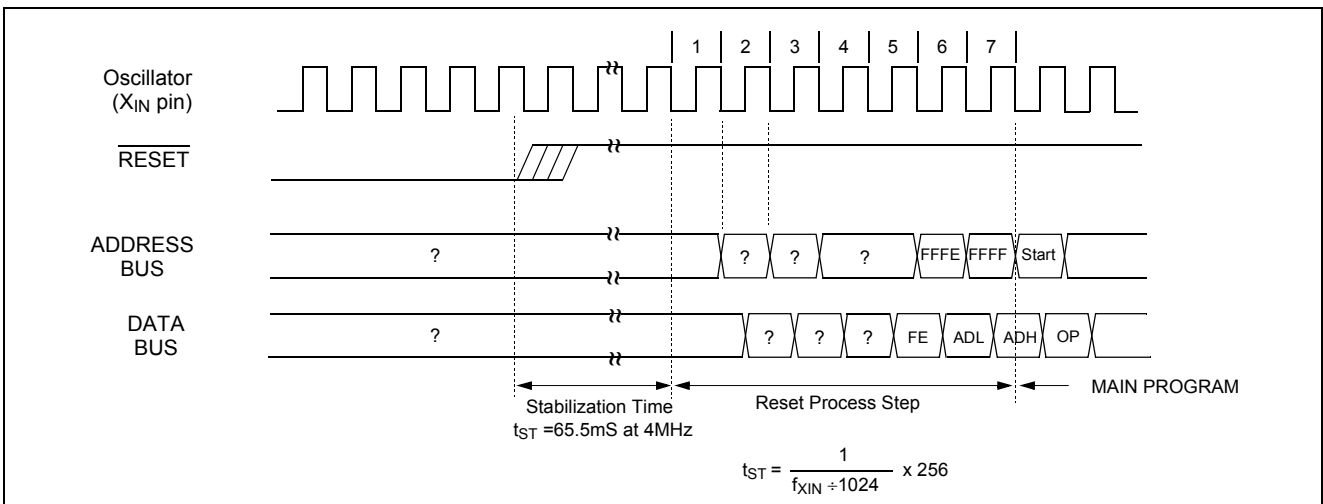
When the RESET pin input goes to high, the reset operation is released and the program execution starts at the vector address stored at addresses FFFE<sub>H</sub> - FFFF<sub>H</sub>.

A connection for simple external reset circuit is shown in Figure 18-1 .





**Figure 18-1 Simple External Reset Circuit**



**Figure 18-2 Timing Diagram after Reset**

The Address Fail Reset is the function to reset the system by checking code access of abnormal and unwished address caused by erroneous program code itself or external noise, which could not be returned to normal operation and would become malfunction state. If the CPU tries to fetch

the instruction from ineffective code area or RAM area, the address fail reset is occurred. Please refer to Figure 11-2 for setting address fail option.

### 19. POWER FAIL PROCESSOR

The MC80F0504/0604 has an on-chip power fail detection circuitry to immunize against power noise. A configuration register, PFDR, can enable or disable the power fail detect circuitry. Whenever  $V_{DD}$  falls close to or below power fail voltage for 100ns, the power fail situation may reset or freeze MCU according to PFDM bit of PFDR as

shown in Figure 19-1

In the in-circuit emulator, power fail function is not implemented and user can not experiment with it. Therefore, after final development of user program, this function may be experimented or evaluated.

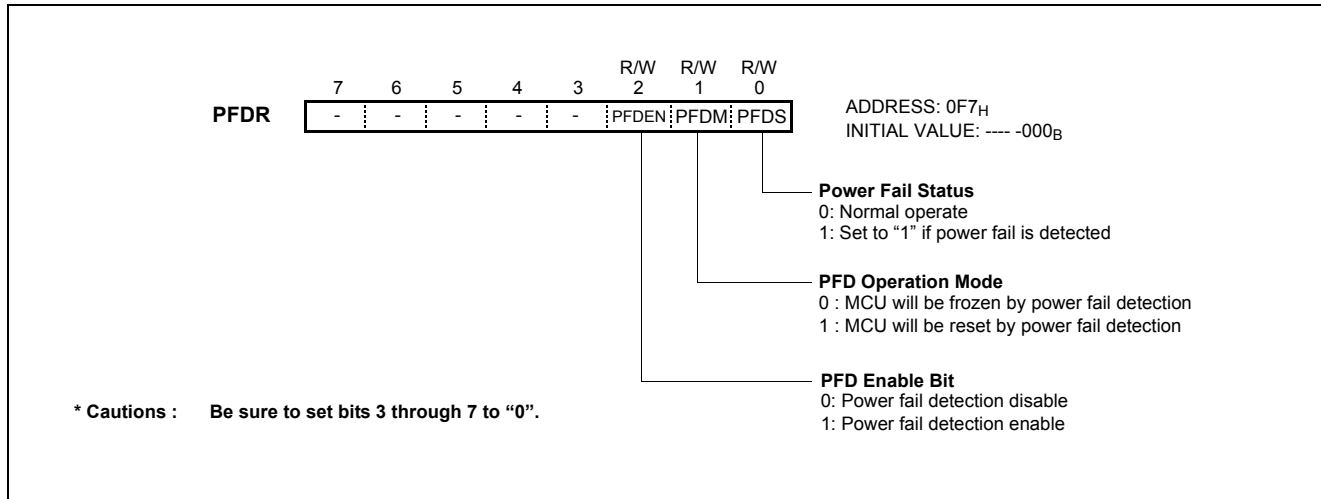


Figure 19-1 Power Fail Voltage Detector Register

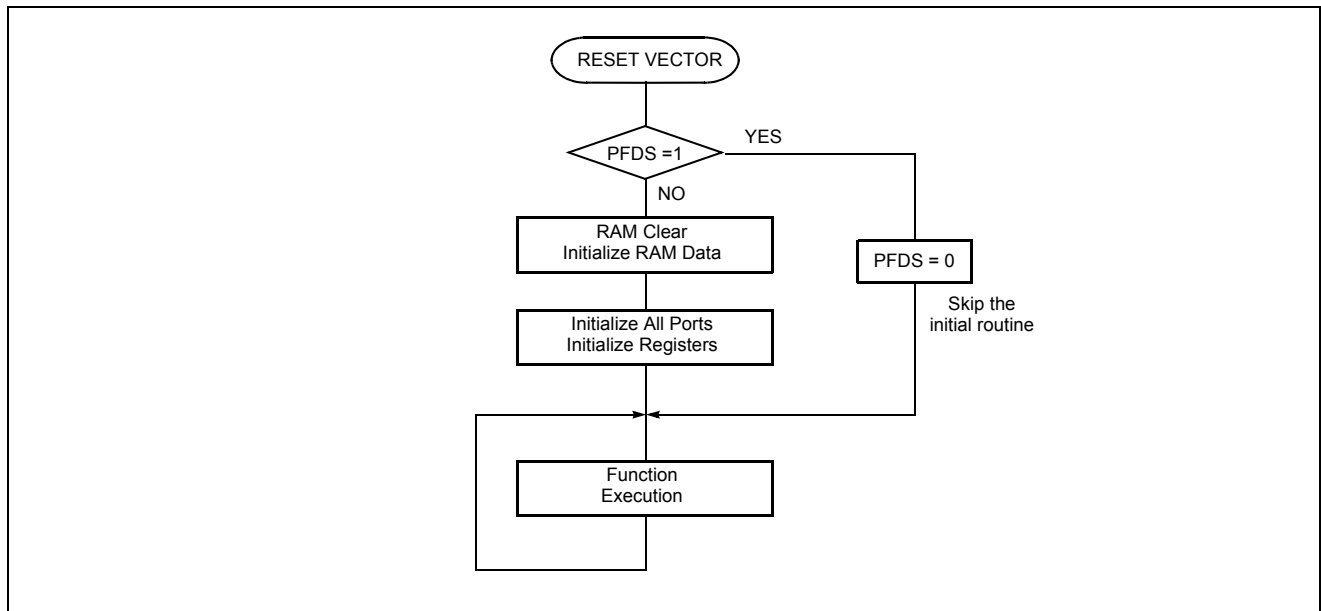


Figure 19-2 Example S/W of Reset flow by Power fail

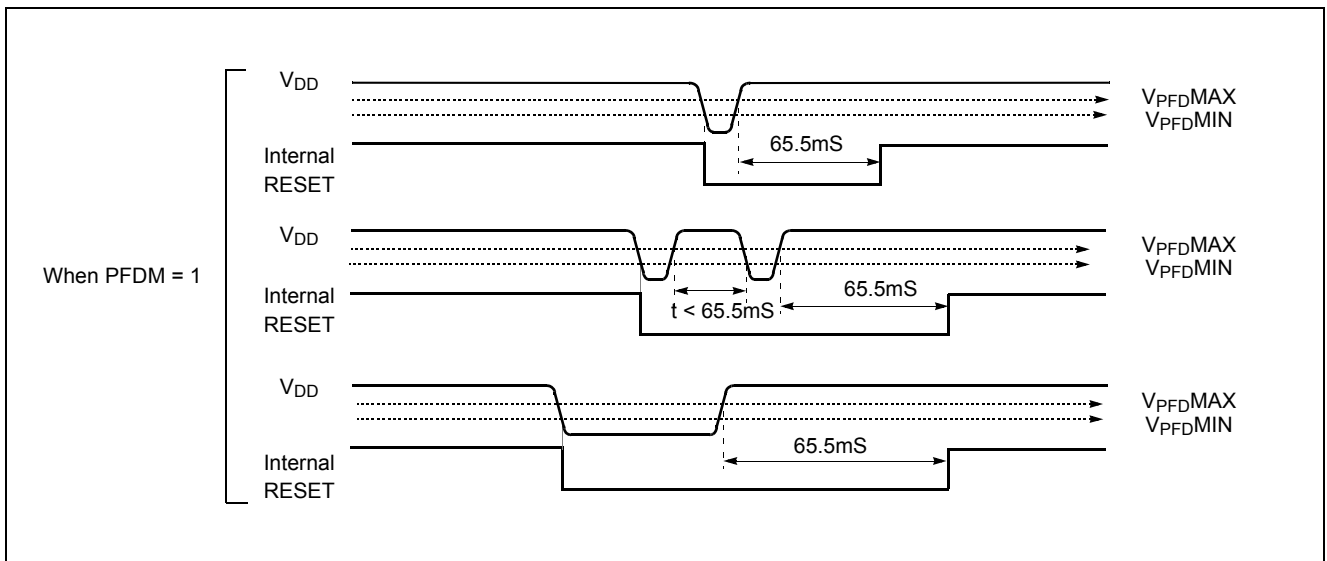


Figure 19-3 Power Fail Processor Situations (at 4MHz operation)

## 20. COUNTERMEASURE OF NOISE

### 20.1 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 MC80F0604.

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.

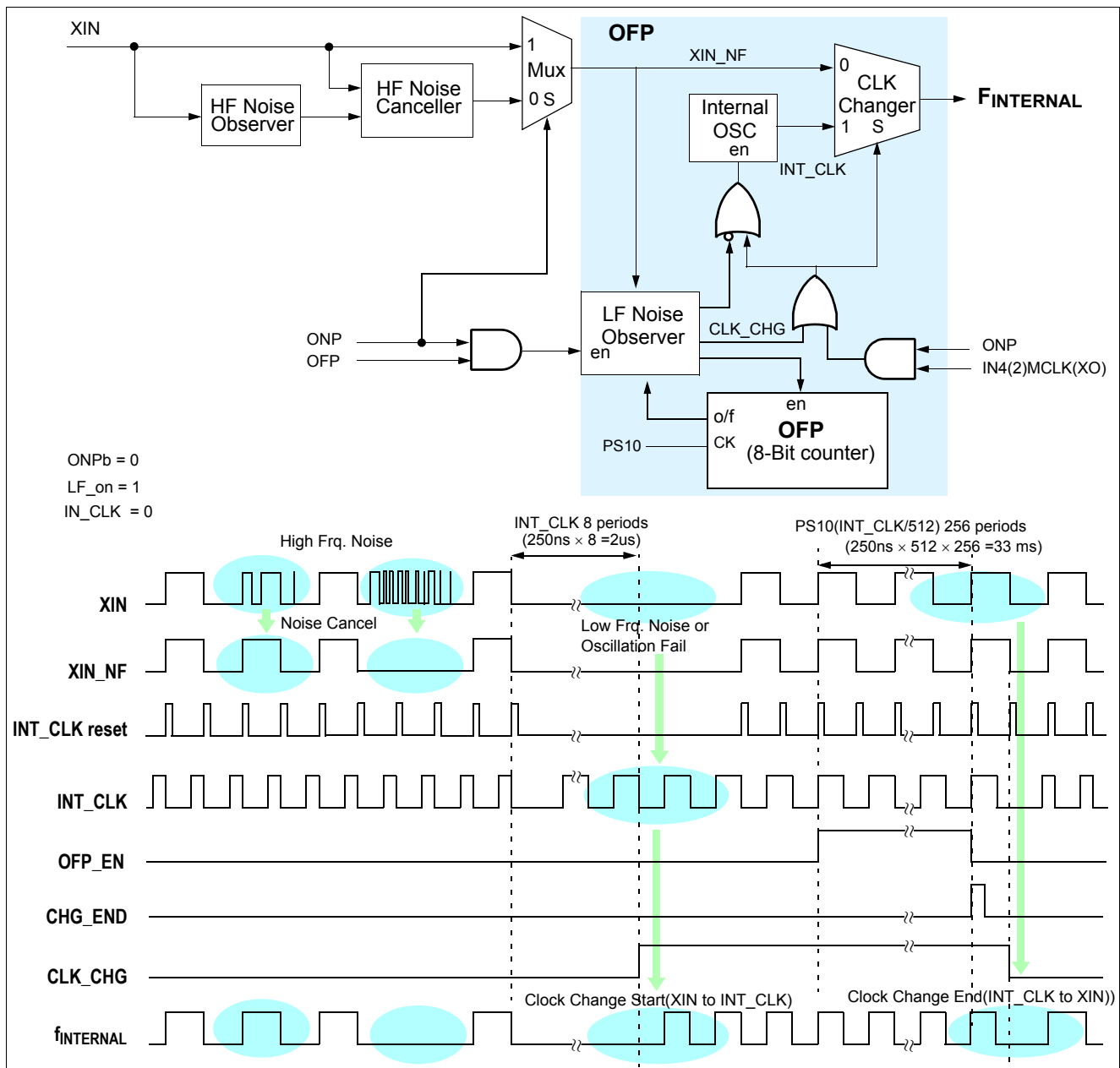


Figure 20-1 Block Diagram of ONP & OFP and Respective Wave Forms

## 20.2 Oscillation Fail Processor

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

### IN4(2)MCLK/CLK(XO) Option

The internal 4MHz or 4MHz oscillation can be used as system clock source in timing insensitive applications. The “IN4MCLK(XO)”, “IN2MCLK(XO)” bit of the Device Configuration Area enables the function to operate the de-

vice 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.

When using internal oscillation, the  $X_{IN}$ ,  $X_{OUT}$  pin can be used as normal I/O pin. In case of selecting IN4MCLK or IN2MCLK, the  $X_{IN}$  pin can be used to R33 and the period of internal oscillator clock could be checked by  $X_{OUT}$  outputting clock divided the internal oscillator clock by 16. If IN4MCLKXO or IN2MCLKCO is selected, the XIN and XOUT pin can be used as R33 and R34 I/O ports.

## 21. DEVICE CONFIGURATION AREA

The Device Configuration Area can be programmed or left unprogrammed to select device configuration such as POR, ONP, CLK option and security bit. This area is not accessible during normal execution but is readable and writable during FLASH program / verify mode.

**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).

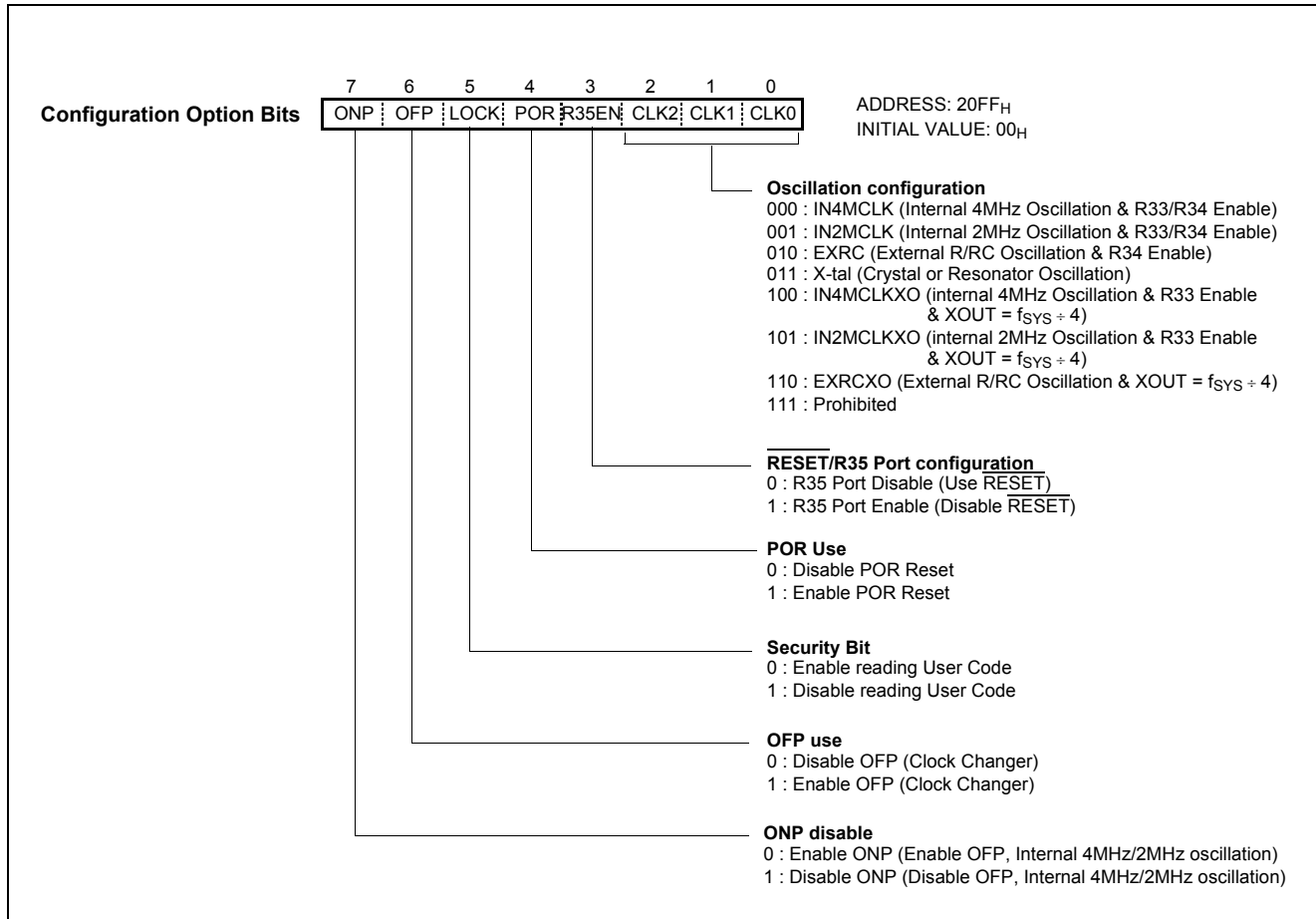
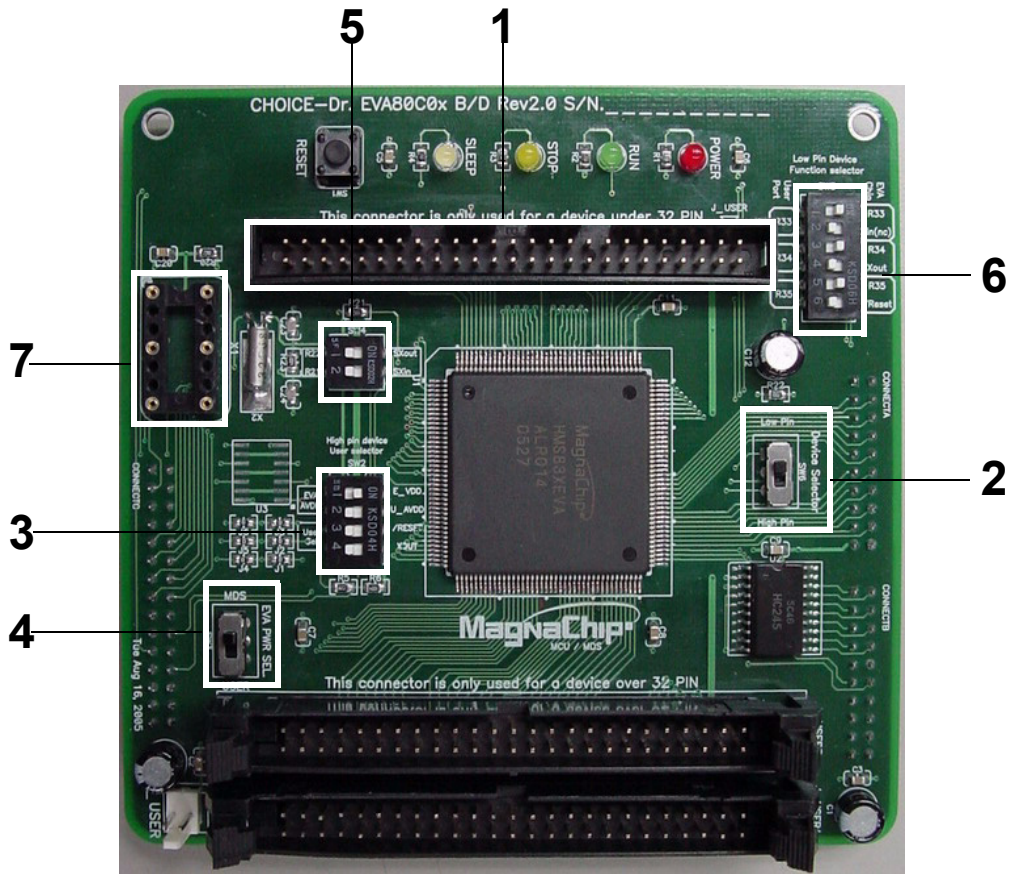


Figure 21-1 Device Configuration Area

These various options shown in Figure 21-1 can be selected by checking option field listed in writer (PGM Plus,

SIGMA or GANG4) software after selecting device name.

22. EMULATOR EVA. BOARD SETTING

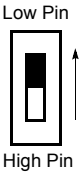
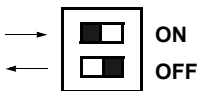
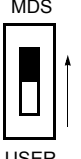
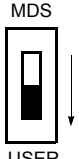


		J_USER			
NC	1	<input type="checkbox"/>	2	NC	
VDD	3	<input type="checkbox"/>	4	VDD	
GND	5	<input type="checkbox"/>	6	GND	
R00	7	<input type="checkbox"/>	8	R10	
R01	9	<input type="checkbox"/>	10	R11	
R02	11	<input type="checkbox"/>	12	R12	
R03	13	<input type="checkbox"/>	14	R13	
R04	15	<input type="checkbox"/>	16	R14	
R05	17	<input type="checkbox"/>	18	R15	
R06	19	<input type="checkbox"/>	20	R16	
R07	21	<input type="checkbox"/>	22	R17	
GND	23	<input type="checkbox"/>	24	GND	
R20	25	<input type="checkbox"/>	26	R30	
R21	27	<input type="checkbox"/>	28	R31	
R22	29	<input type="checkbox"/>	30	R32	
R23	31	<input type="checkbox"/>	32	R33/XIN	
R24	33	<input type="checkbox"/>	34	R34/XOUT	
R25	35	<input type="checkbox"/>	36	R35/RST	
R26	37	<input type="checkbox"/>	38	R36	
R27	39	<input type="checkbox"/>	40	R37	
GND	41	<input type="checkbox"/>	42	GND	
NC	43	<input type="checkbox"/>	44	NC	
NC	45	<input type="checkbox"/>	46	NC	
NC	47	<input type="checkbox"/>	48	NC	
NC	49	<input type="checkbox"/>	50	NC	

## DIP Switch and VR Setting

low configuration

Before execute the user program, keep in your mind the be-

DIP S/W		Description	ON/OFF Setting
<b>1</b>	-	This connector is only used for a device under 32 PIN.	For the MC80F0504/0504
<b>2</b> SW6	-	Device select switch Low pin . 	Must be <b>Low Pin</b> position. <b>High Pin</b> : For the MC80F0208/16/24. <b>Low Pin</b> : For the MC80F0504/0604.
<b>3</b> SW2	1 2	 Use Eva. V <sub>DD</sub> AV <sub>DD</sub> select switch to Eva. V <sub>DD</sub> .	These switches select the AV <sub>DD</sub> source for high pin devices and should be set to use Eva. V <sub>DD</sub> . <b>ON &amp; OFF</b> : Use Eva. V <sub>DD</sub>
	3	This switch select the /Reset source.	Normally <b>OFF</b> . EVA. chip can be reset by external user target board. <b>ON</b> : Reset is available by either user target system board or Emulator RESET switch. <b>OFF</b> : Reset the MCU by Emulator RESET switch. Does not work from user target board.
	4	This switch select the Xout signal on/off.	Normally <b>OFF</b> . MCU XOUT pin is disconnected internally in the Emulator. User may connect this circuit with this switch. <b>ON</b> : Output XOUT signal <b>OFF</b> : Disconnect circuit
<b>4</b> SW3	1	This switch select Eva. B/D Power supply source.  Use MDS Power  Use User's Power	Normally <b>MDS</b> . This switch select Eva. B/D Power supply source.
<b>5</b> SW4	1 2	This switch select the R22 or SX <sub>OUT</sub> . This switch select the R21 or SX <sub>IN</sub> .	These switches select the Normal I/O port (off) or Sub-Clock (on). It is reserved for the MC80F0448. <b>ON</b> : SX <sub>OUT</sub> , SX <sub>IN</sub> <b>OFF</b> : R22, R21 OFF (MC80F0504/0604).



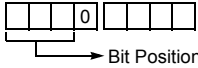
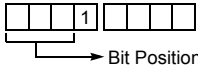
DIP S/W	Description	ON/OFF Setting
<p><b>6</b> SW5</p>	<p>These switches select the R33 or X<sub>IN</sub></p> <p>Select R33 port                      Select X<sub>IN</sub> (NC)</p>	<p>This switch select the Normal I/O port (off) or X<sub>IN</sub>(NC) select (on).  <b>ON &amp; OFF</b> : R33 Port selected.  <b>OFF &amp; ON</b> : X<sub>IN</sub>(NC) selected.</p>
	<p>These switches select the R34 or X<sub>OUT</sub></p> <p>Select R34 port                      Select X<sub>OUT</sub></p>	<p>This switch select the Normal I/O port (off) or X<sub>OUT</sub> select (on).  <b>ON &amp; OFF</b> : R34 Port selected.  <b>OFF &amp; ON</b> : X<sub>OUT</sub> selected.</p>
	<p>These switches select the R35 or X<sub>OUT</sub></p> <p>Select R35 port                      Select /Reset</p>	<p>This switch select the Normal I/O port (off) or /Reset select (on).  <b>ON &amp; OFF</b> : R35 Port selected.  <b>OFF &amp; ON</b> : /Reset selected.</p>
<p><b>7</b></p>	<p>- This is External oscillation socket (CAN Type. OSC)</p>	<p>This is for External Clock (CAN Type. OSC).</p>



# **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 (000H~0FFFH)
rel	Relative Addressing Data
upage	U-page (0FF00H~0FFFFH) Offset Address
n	Table CALL Number (0~15)
+	Addition
x	 Upper Nibble Expression in Opcode
y	 Upper Nibble Expression in Opcode
-	Subtraction
x	Multiplication
/	Division
()	Contents Expression
^	AND
∨	OR
⊕	Exclusive OR
~	NOT
←	Assignment / Transfer / Shift Left
→	Shift Right
↔	Exchange
=	Equal
≠	Not Equal

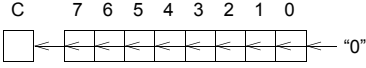
## A.2 Instruction Map

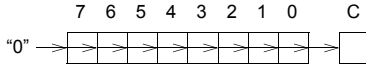
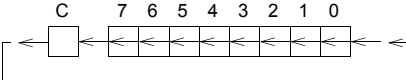
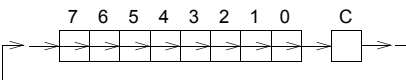
LOW HIGH	0000 00	00001 01	00010 02	00011 03	00100 04	00101 05	00110 06	00111 07	01000 08	01001 09	01010 0A	01011 0B	01100 0C	01101 0D	01110 0E	01111 0F
000	-	SET1 dp.bit	BBS A.bit,rel	BBS dp.bit,rel	ADC #imm dp	ADC dp	ADC dp+X	ADC !abs	ASL A	ASL dp	TCALL 0	SETA1 .bit	BIT dp	POP A	PUSH A	BRK
001	CLRC	"	"	"	SBC #imm dp	SBC dp	SBC dp+X	SBC !abs	ROL A	ROL dp	TCALL 2	CLRA1 .bit	COM dp	POP X	PUSH X	BRA rel
010	CLRG	"	"	"	CMP #imm dp	CMP dp	CMP dp+X	CMP !abs	LSR A	LSR dp	TCALL 4	NOT1 M.bit	TST dp	POP Y	PUSH Y	PCALL Upage
011	DI	"	"	"	OR #imm dp	OR dp	OR dp+X	OR !abs	ROR A	ROR dp	TCALL 6	OR1 OR1B	CMPX dp	POP PSW	PUSH PSW	RET
100	CLRV	"	"	"	AND #imm dp	AND dp	AND dp+X	AND !abs	INC A	INC dp	TCALL 8	AND1 AND1B	CMPY dp	CBNE dp+X	TXSP	INC X
101	SETC	"	"	"	EOR #imm dp	EOR dp	EOR dp+X	EOR !abs	DEC A	DEC dp	TCALL 10	EOR1 EOR1B	DBNE dp	XMA dp+X	TSPX	DEC X
110	SETG	"	"	"	LDA #imm dp	LDA dp	LDA dp+X	LDA !abs	TXA	LDY dp	TCALL 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	TCALL 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	TCALL 1	JMP !abs	BIT !abs	ADDW dp	LDX #imm	JMP [!abs]
001	BVC rel	"	"	"	SBC {X}	SBC !abs+Y	SBC [dp+X]	SBC [dp]+Y	ROL !abs	ROL dp+X	TCALL 3	CALL !abs	TEST !abs	SUBW dp	LDY #imm	JMP [dp]
010	BCC rel	"	"	"	CMP {X}	CMP !abs+Y	CMP [dp+X]	CMP [dp]+Y	LSR !abs	LSR dp+X	TCALL 5	MUL	TCLR1 !abs	CMPW dp	CMPX #imm	CALL [dp]
011	BNE rel	"	"	"	OR {X}	OR !abs+Y	OR [dp+X]	OR [dp]+Y	ROR !abs	ROR dp+X	TCALL 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	TCALL 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	TCALL 11	XMA {X}	XMA dp	DECW dp	DEC Y	TYA
110	BCS rel	"	"	"	LDA {X}	LDA !abs+Y	LDA [dp+X]	LDA [dp]+Y	LDY !abs	LDY dp+X	TCALL 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	TCALL 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 	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	-	-	-	Unsupported	-
37	DAS	-	-	-	Unsupported	-
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-

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
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		
59	LSR A	48	1	2	Logical shift right 	N-----ZC
60	LSR dp	49	2	4		
61	LSR dp + X	59	2	5		
62	LSR !abs	58	3	5		
63	MUL	5B	1	9	Multiply : $YA \leftarrow Y \times A$	N-----Z-
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		
72	ROL A	28	1	2	Rotate left through carry 	N-----ZC
73	ROL dp	29	2	4		
74	ROL dp + X	39	2	5		
75	ROL !abs	38	3	5		
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 $A \leftarrow (M)$	N-----Z-
2	LDA dp	C5	2	3		
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		
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 $X \leftarrow (M)$	N-----Z-
12	LDX dp	CC	2	3		
13	LDX dp + Y	CD	2	4		
14	LDX !abs	DC	3	4		
15	LDY #imm	3E	2	2	Load Y-register $Y \leftarrow (M)$	N-----Z-
16	LDY dp	C9	2	3		
17	LDY dp + X	D9	2	4		
18	LDY !abs	D8	3	4		
19	STA dp	E5	2	4	Store accumulator contents in memory $(M) \leftarrow A$	-----
20	STA dp + X	E6	2	5		
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 $(M) \leftarrow X$	-----
28	STX dp + Y	ED	2	5		
29	STX !abs	FC	3	5		
30	STY dp	E9	2	4	Store Y-register contents in memory $(M) \leftarrow Y$	-----
31	STY dp + X	F9	2	5		
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 : $X \leftrightarrow A$	-----
40	XAY	DE	1	4	Exchange Y-register contents with accumulator : $Y \leftrightarrow A$	-----
41	XMA dp	BC	2	5	Exchange memory contents with accumulator $(M) \leftrightarrow A$	N-----Z-
42	XMA dp+X	AD	2	6		
43	XMA {X}	BB	1	5		
44	XYX	FE	1	4		



**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 :	MM----Z-
4	BIT labs	1C	3	5	$Z \leftarrow (A) \wedge (M), N \leftarrow (M_7), V \leftarrow (M_6)$	
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 labs	5C	3	6	Test and clear bits with A : $A - (M), (M) \leftarrow (M) \wedge \sim(A)$	N-----Z-
23	TSET1 labs	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 ← ( 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 ← ( pc ) + rel	-----
5	BCC rel	50	2	2/4	Branch if carry bit clear if ( C ) = 0 , then pc ← ( pc ) + rel	-----
6	BCS rel	D0	2	2/4	Branch if carry bit set if ( C ) = 1 , then pc ← ( pc ) + rel	-----
7	BEQ rel	F0	2	2/4	Branch if equal if ( Z ) = 1 , then pc ← ( pc ) + rel	-----
8	BMI rel	90	2	2/4	Branch if minus if ( N ) = 1 , then pc ← ( pc ) + rel	-----
9	BNE rel	70	2	2/4	Branch if not equal if ( Z ) = 0 , then pc ← ( pc ) + rel	-----
10	BPL rel	10	2	2/4	Branch if minus if ( N ) = 0 , then pc ← ( pc ) + rel	-----
11	BRA rel	2F	2	4	Branch always pc ← ( pc ) + rel	-----
12	BVC rel	30	2	2/4	Branch if overflow bit clear if ( V ) = 0 , then pc ← ( pc ) + rel	-----
13	BVS rel	B0	2	2/4	Branch if overflow bit set if ( V ) = 1 , then pc ← ( pc ) + rel	-----
14	CALL !abs	3B	3	8	Subroutine call	-----
15	CALL [dp]	5F	2	8	M( sp)←( pc <sub>H</sub> ), sp←sp - 1, M(sp)←( pc <sub>L</sub> ), sp ←sp - 1, if !abs, pc← abs ; if [dp], pc <sub>L</sub> ← ( dp ), pc <sub>H</sub> ← ( 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 ) ≠ ( M ) , then pc ← ( 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 ) ≠ 0 , then pc ← ( pc ) + rel.	-----
20	JMP !abs	1B	3	3	Unconditional jump pc ← jump address	-----
21	JMP [!abs]	1F	3	5		
22	JMP [dp]	3F	2	4		
23	PCALL upage	4F	2	6	U-page call M(sp)←( pc <sub>H</sub> ), sp ←sp - 1, M(sp)←( pc <sub>L</sub> ), sp ← sp - 1, pc <sub>L</sub> ← ( upage ), pc <sub>H</sub> ← "OFFH" .	-----
24	TCALL n	nA	1	8	Table call : ( sp ) ←( pc <sub>H</sub> ), sp ← sp - 1, M(sp)←( pc <sub>L</sub> ), sp ← sp - 1, pc <sub>L</sub> ← ( Table vector L ), pc <sub>H</sub> ← ( Table vector H )	-----

**Control Operation & 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 interrupts : $I \leftarrow "0"$	-----0--
3	EI	E0	1	3	Enable interrupts : $I \leftarrow "1"$	-----1--
4	NOP	FF	1	2	No operation	-----
5	POP A	0D	1	4	$sp \leftarrow sp + 1$ , $A \leftarrow M(sp)$ $sp \leftarrow sp + 1$ , $X \leftarrow M(sp)$ $sp \leftarrow sp + 1$ , $Y \leftarrow M(sp)$ $sp \leftarrow sp + 1$ , $PSW \leftarrow M(sp)$	-----
6	POP X	2D	1	4		
7	POP Y	4D	1	4		
8	POP PSW	6D	1	4		
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 )	-----

