MAGNACHIP SEMICONDUCTOR LTD.
8-BIT SINGLE-CHIP MICROCONTROLLERS

MC80F0104/0204
MC80C0104/0204

Preliminary User's Manual (Ver. 0.2)

MagnaChip

REVISION HISTORY

VERSION 0.2 (MAR. 2005) This book

Fix some errata.

VERSION 0.1 (MAR. 2005) This book

First Edition

Version 0.2

Published by
MCU Application Team

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

MagnaChip 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, MagnaChip 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.

MC80F0104/0204
MC80C0104/0204

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

1. OVERVIEW

1.1 Description

The MC80F0104/0204 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, 256 bytes of RAM, 8/16-bit timer/counter, watchdog timer, 10-bit A/D converter, 8-bit Serial Input/Output, UART, 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 MC80F0104/0204 supports power saving modes to reduce power consumption.

The MC80C0104/0204 is the MASK ROM version of the MC80F0104/0204. It is fully compatible to the MC80F0104/0204 in function.

This document **explains the base MC80F0204**, the other's eliminated functions are same as below table.

Device Name		FLASH (ROM) Size	RAM	ADC	I/O PORT	Package
FLASH	MASK ROM					
MC80F0204	MC80C0204	4KB	256B	10 channel	18 port	20 PDIP, 20SOP
MC80F0104	MC80C0104			8 channel	14 port	16 PDIP, 16 SOP

1.2 Features

- **4K Bytes On-chip FLASH**
  - 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)**
  - MC80F0204 : 18(17)
  - MC80F0104 : 14(13)
- **One 8-bit Basic Interval Timer**
- **Four 8-bit Timer/counters (or two 16-bit Timer/counter)**
- **One Watchdog timer**
- **Two 10-bit High Speed PWM Outputs**
- **10-bit A/D converter**
  - MC80F0204 : 10 channels
  - MC80F0104 : 8 channels
- **Two 8-bit Serial Communication Interface**
  - One Serial I/O and one UART
- **One Buzzer Driving port**
  - 488Hz ~ 250kHz@4MHz
- **Four External Interrupt input ports**
- **On-chip POR (Power on Reset)**
- **Fourteen Interrupt sources**
  - External input : 4
  - Timer : 6
  - A/D Conversion : 1
  - Serial Interface : 1
  - UART : 2
- **Built in Noise Immunity Circuit**
  - Noise Canceller
  - PFD (Power fail detector)
  - ONP (Oscillation Noise Protector)
- **Operating Voltage & Frequency (MC80F0104/0204)**
  - 2.7V ~ 5.5V (at 0.4 ~ 8MHz)
  - 4.5V ~ 5.5V (at 0.4 ~ 12MHz)

- **Operating Voltage & Frequency(MC80C0104/0204)**
  - 2.0V ~ 5.5V (at 0.4 ~ 4.2MHz)
  - 2.7V ~ 5.5V (at 0.4 ~ 8MHz)
  - 4.5V ~ 5.5V (at 0.4 ~ 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)

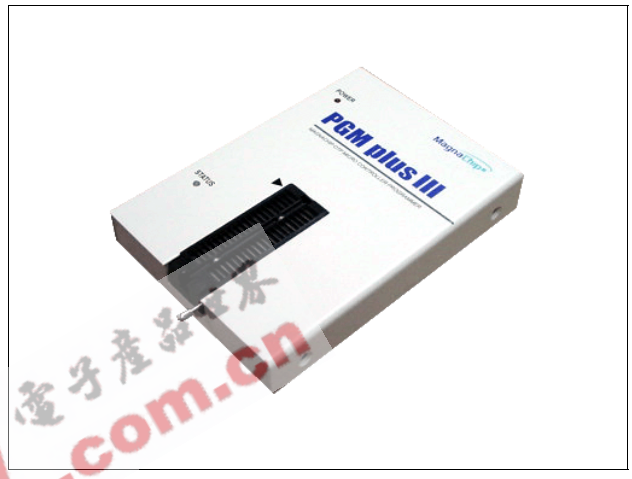
1.3 Development Tools

The MC80F0104/0204 is supported by a full-featured macro assembler, an in-circuit emulator CHOICE-Dr.TM and OTP 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 MagnaChip semiconductor.

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

Choice-Dr. (Emulator)



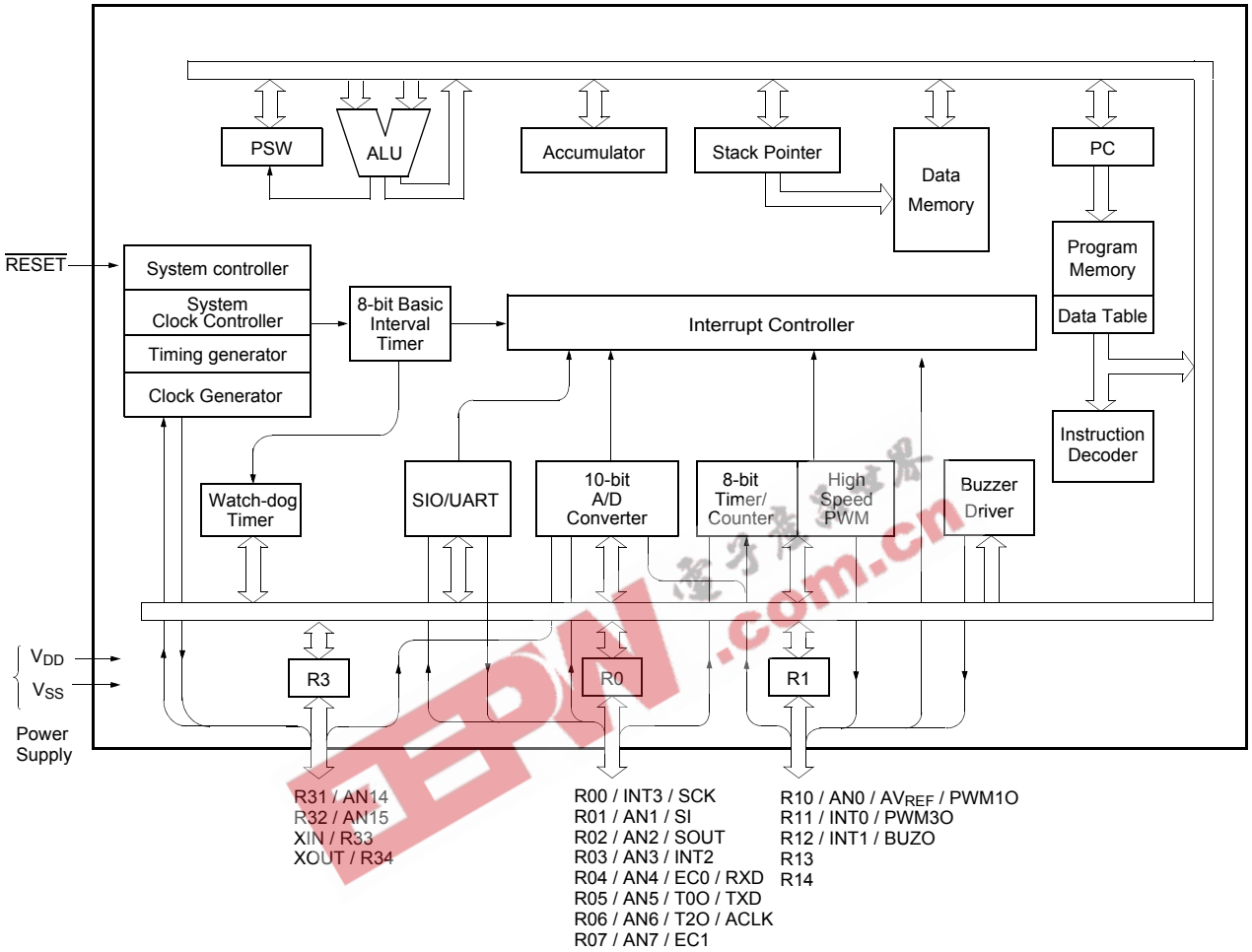
PGMplus III (Single Writer)

Standalone Gang4 II (Gang Writer)

1.4 Ordering Information

	Device name	ROM Size	RAM size	Package
Mask version	MC80C0204B MC80C0204D MC80C0104B MC80C0104D	4K bytes 4K bytes 4K bytes 4K bytes	256 bytes	20PDIP 20SOP 16PDIP 16SOP
FLASH version	MC80F0204B MC80F0204D MC80F0104B MC80F0104D	4K bytes FLASH 4K bytes FLASH 4K bytes FLASH 4K bytes FLASH	256 bytes	20PDIP 20SOP 16PDIP 16SOP

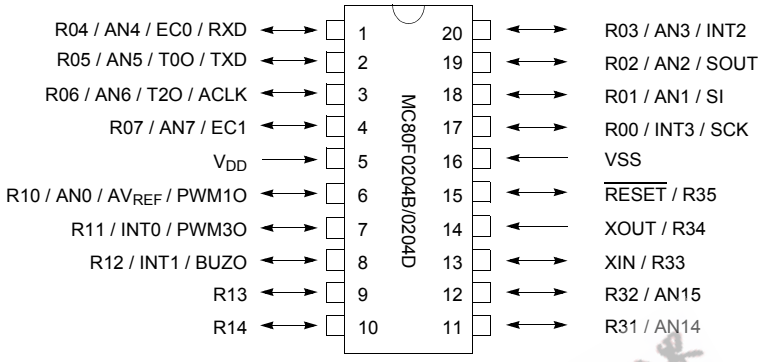
2. BLOCK DIAGRAM



3. PIN ASSIGNMENT

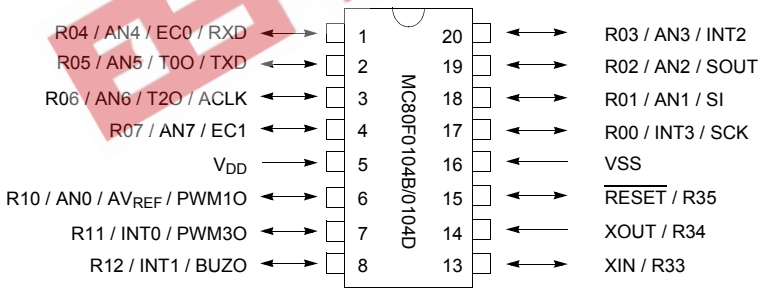
MC80F0204B/0204D

20 PDIP
20 SOP



MC80F0104B/0104D

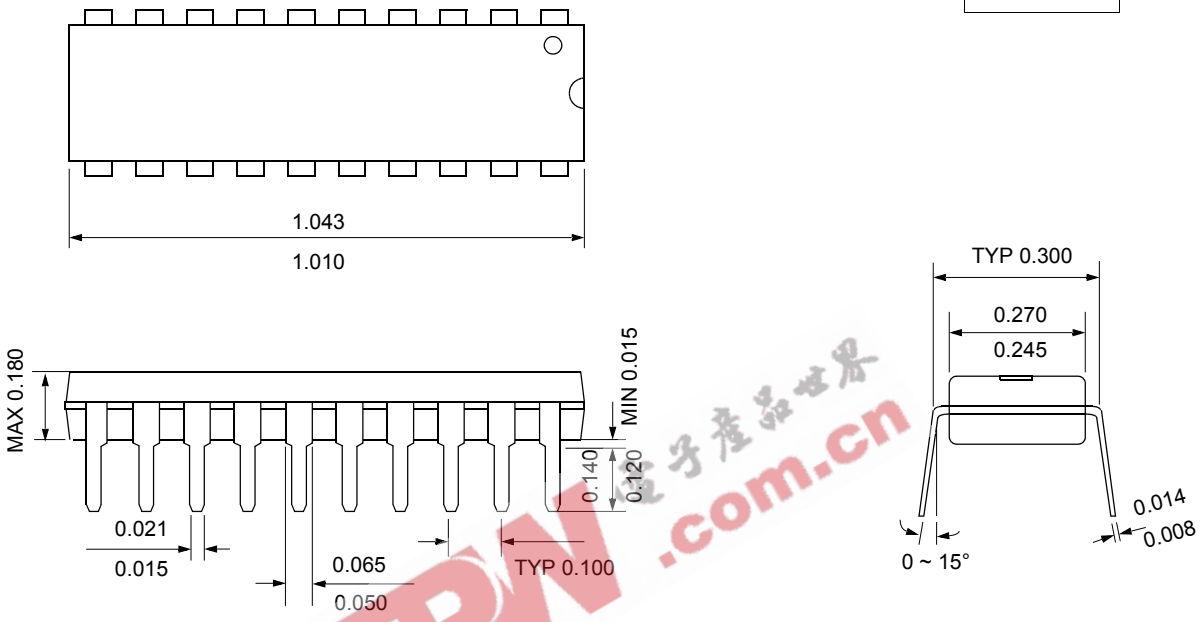
16 PDIP
16 SOP



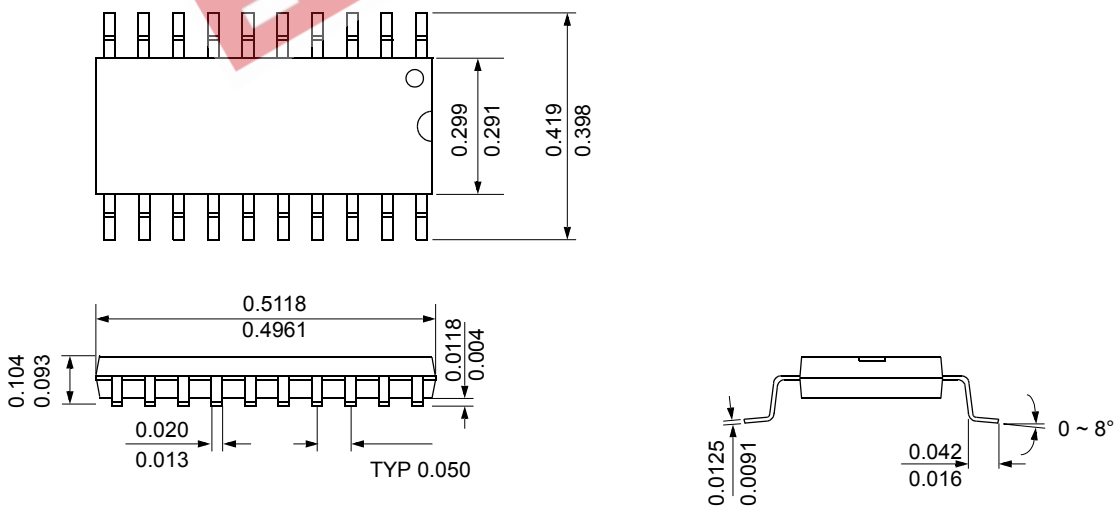
4. PACKAGE DRAWING

20 PDIP

unit: inch
MAX
MIN

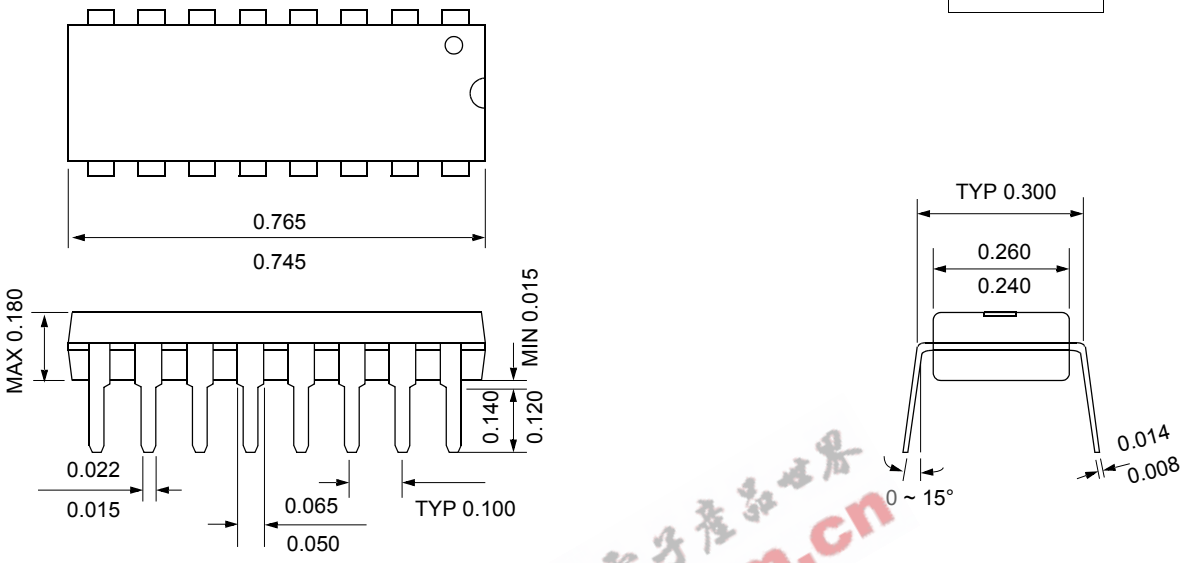


20 SOP

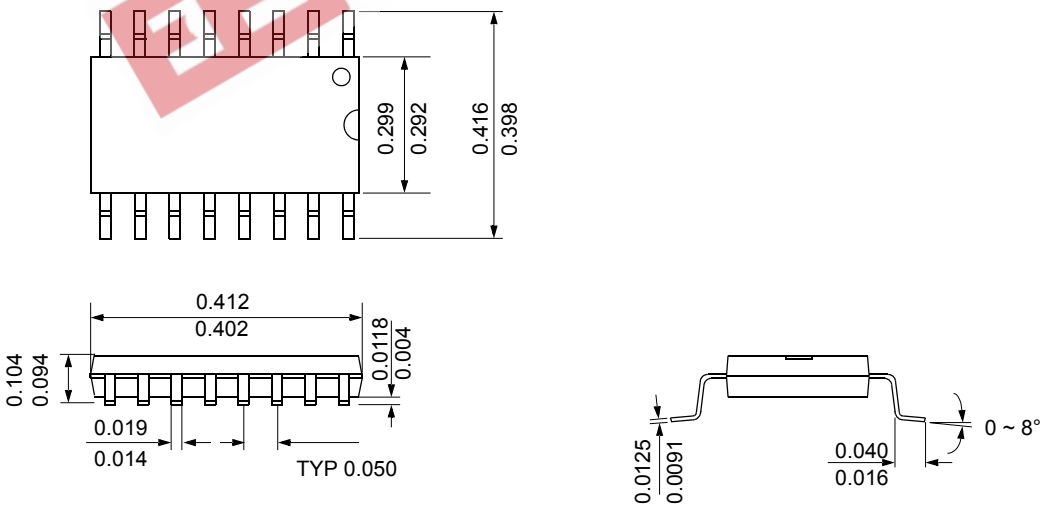


16 PDIP

unit: inch
MAX
MIN



16 SOP



5. PIN FUNCTION

V_{DD}: Supply voltage.

V_{SS}: Circuit ground.

$\overline{RESET}$: Reset the MCU.

X_{IN}: Input to the inverting oscillator amplifier and input to the internal main clock operating circuit.

X_{OUT}: 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	INT3 (External Interrupt Input Port3) SCK (SPI CLK)
R01	AN1 (Analog Input Port 1) SI (SPI Serial Data Input)
R02	AN2 (Analog Input Port 2) SOUT (SPI Serial Data Output)
R03	AN3 (Analog Input Port 3) INT2 (External Interrupt Input Port2)
R04	AN4 (Analog Input Port 4) EC0 (Event Counter Input Source 0) RXD (UART Data Input)
R05	AN5 (Analog Input Port 5) T0O (Timer0 Clock Output) TXD (UART Data Output)
R06	AN6 (Analog Input Port 6) T2O (Timer2 Clock Output) ACLK (UART Clock Input)
R07	AN7 (Analog Input Port 7) EC1 (Event Counter Input Source 1)

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) PWM3O (PWM3 Output)
R12	INT1 (External Interrupt Input Port 1) BUZ (Buzzer Driving Output Port)
R13	
R14	

Table 5-2 R1 Port

R31~R34: R3 is an 4-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).

R3 serves the functions of the serial interface 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_{IN} (Oscillation Input)
R34	X_{OUT} (Oscillation Output)
R35	RESET (Reset input port)

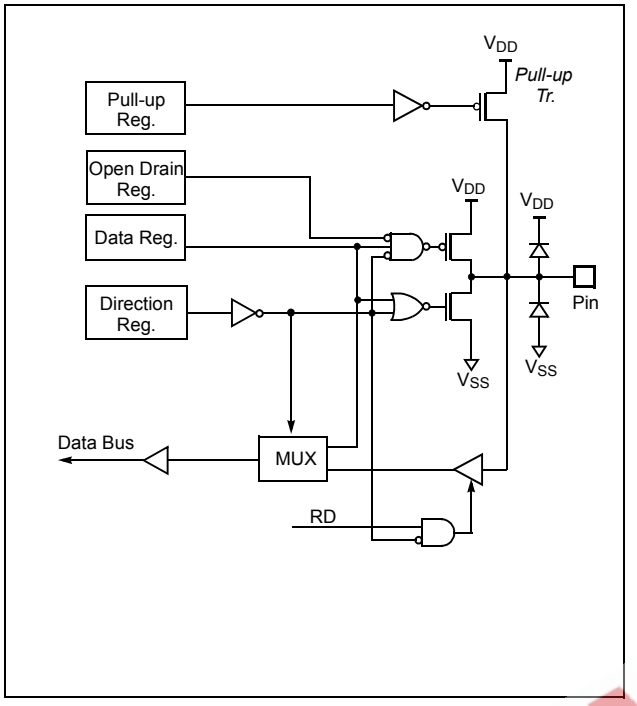
Table 5-3 R3 Port

PIN NAME	Pin No. (20PDIP)	In/Out	Function	
V_{DD}	5	-	Supply voltage	
V_{SS}	16	-	Circuit ground	
$\overline{RESET}$ (R35)	15	I (I)	Reset signal input	Input only port
X_{IN} (R33)	13	I (I/O)	Oscillation Input	Normal I/O Port
X_{OUT} (R34)	14	O (I/O)	Oscillation Output	Normal I/O Port
R00 (INT3 / SCK)	17	I/O (Input / I/O)	Normal I/O Ports	External Interrupt Input3 / SPI clock Input/Output
R01 (AN1 / SI)	18	I/O (Input/Input)		Analog Input Port 1 / SPI Data Input
R02 (AN2 / SOUT)	19	I/O (Input/Output)		Analog Input Port 2 / SPI Data Output
R03 (AN3 / INT2)	20	I/O (Input/Input)		Analog Input Port 3 / External Interrupt Input2
R04 (AN4 / EC0 / RXD)	1	I/O (Input/Input/Input)		Analog Input Port 4 / Event Counter Input 0 / UART Data Input
R05 (AN5 / T0O / TXD)	2	I/O (Input/Output/Output)		Analog Input Port 5 / Timer0 Output / UART Data Output
R06 (AN6 / T2O / ACLK)	3	I/O (Input/Output/Input)		Analog Input Port 6 / Timer2 Output / UART Clock Input
R07 (AN7 / EC1)	4	I/O (Input/Input)		Analog Input Port 7 / Event Counter Input 1
R10 (AN0 / AV_{REF} / PWM1O)	6	I/O (Input/Input/Output)		Analog Input Port 0 / Analog Reference / PWM 1 output
R11 (INT0 / PWM3O)	7	I/O (Input/Output)		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

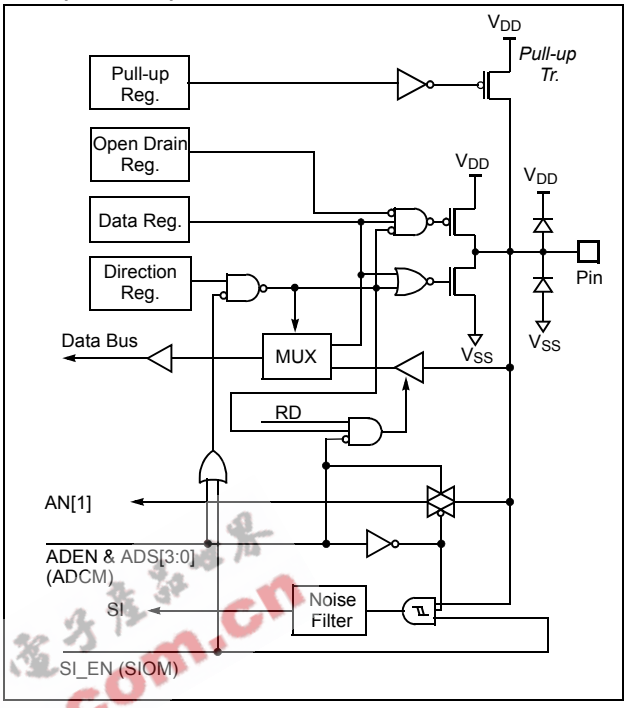
Table 5-4 Pin Description

6. PORT STRUCTURES

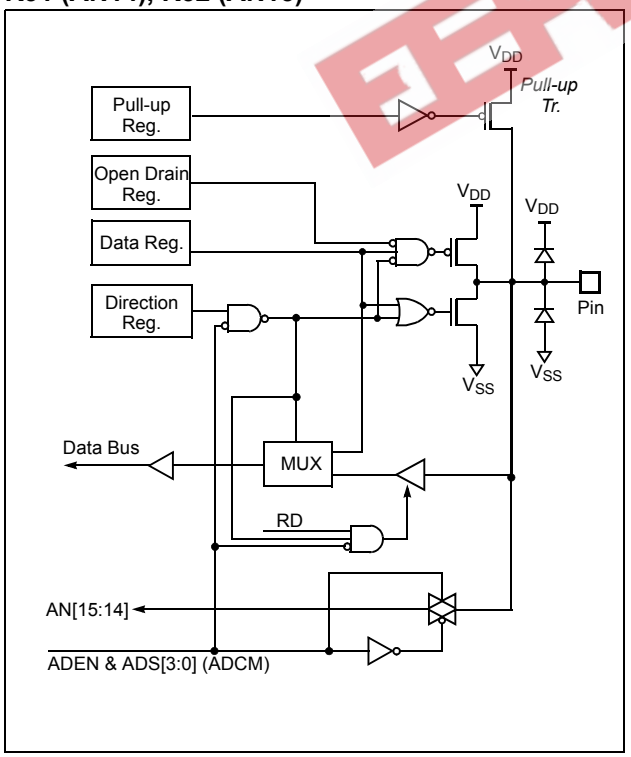
R13~R14



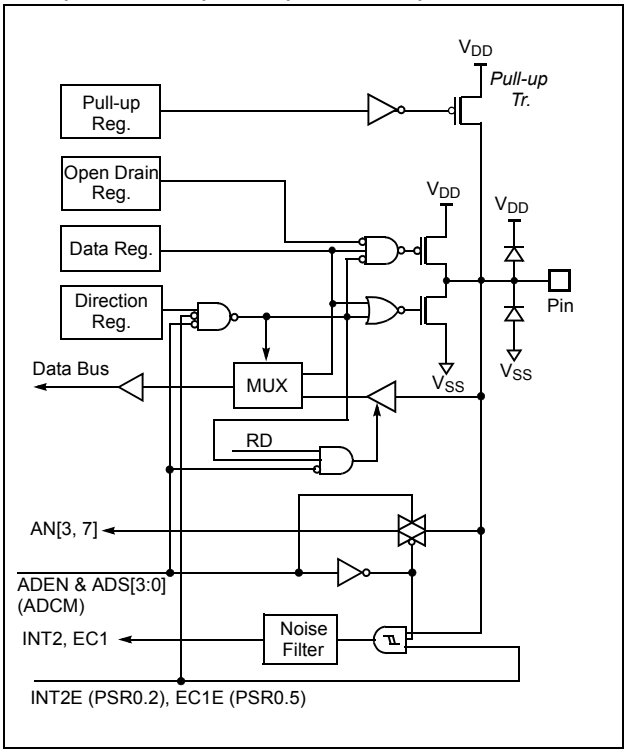
R01 (AN1 / SI)



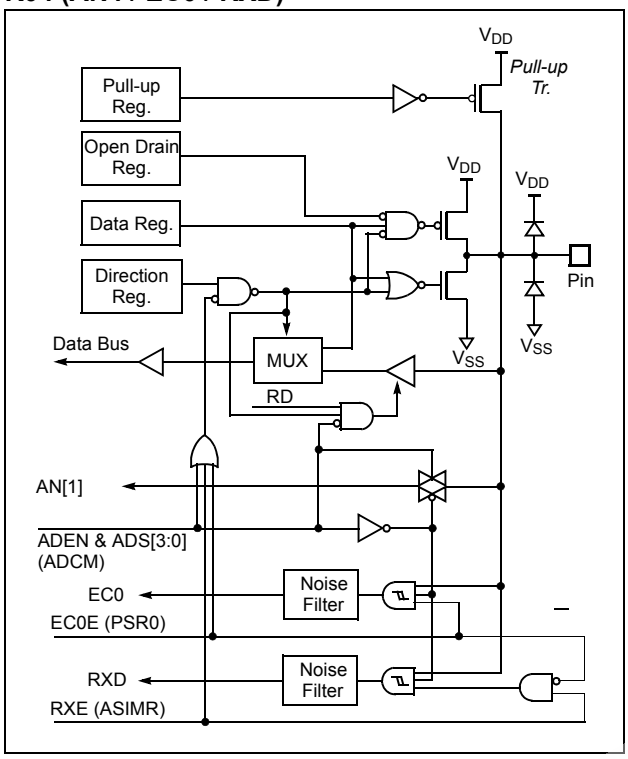
R31 (AN14), R32 (AN15)



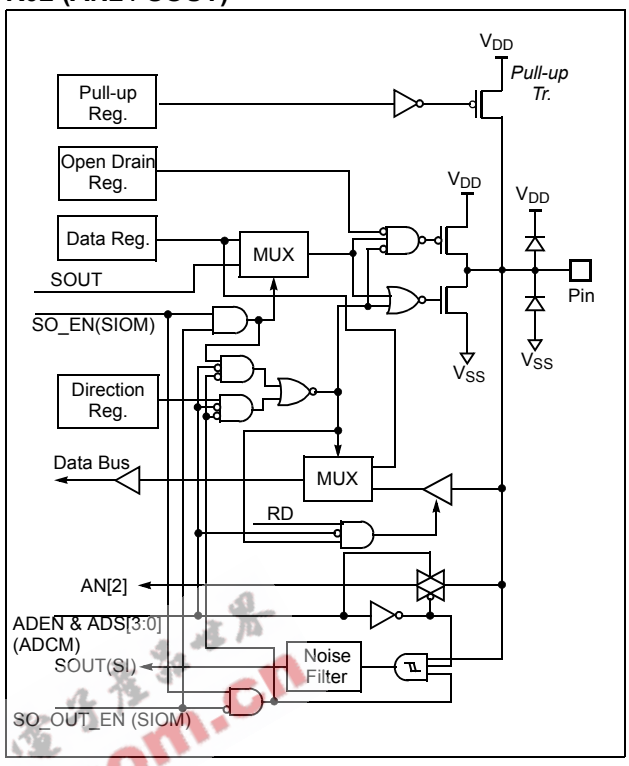
R03 (AN3 / INT2), R07 (AN7 / EC1)



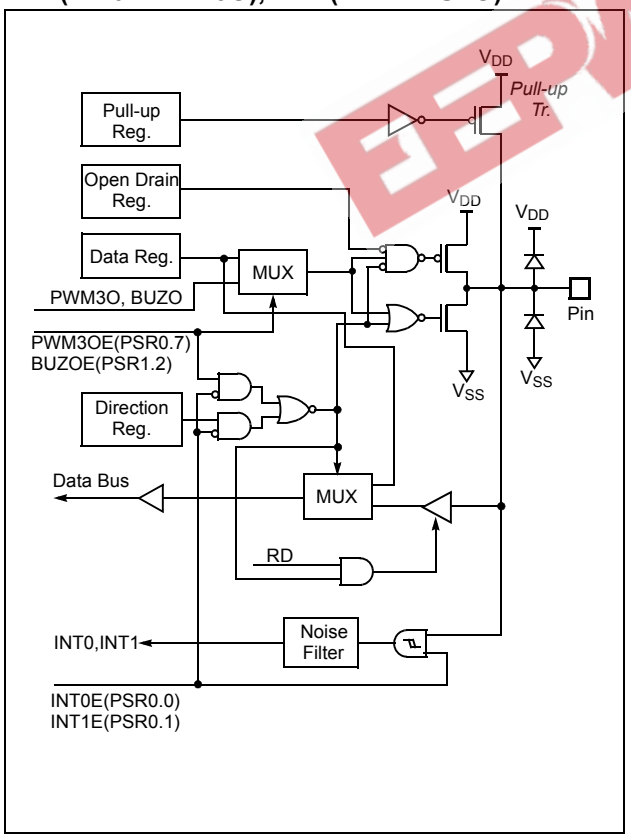
R04 (AN4 / EC0 / RXD)



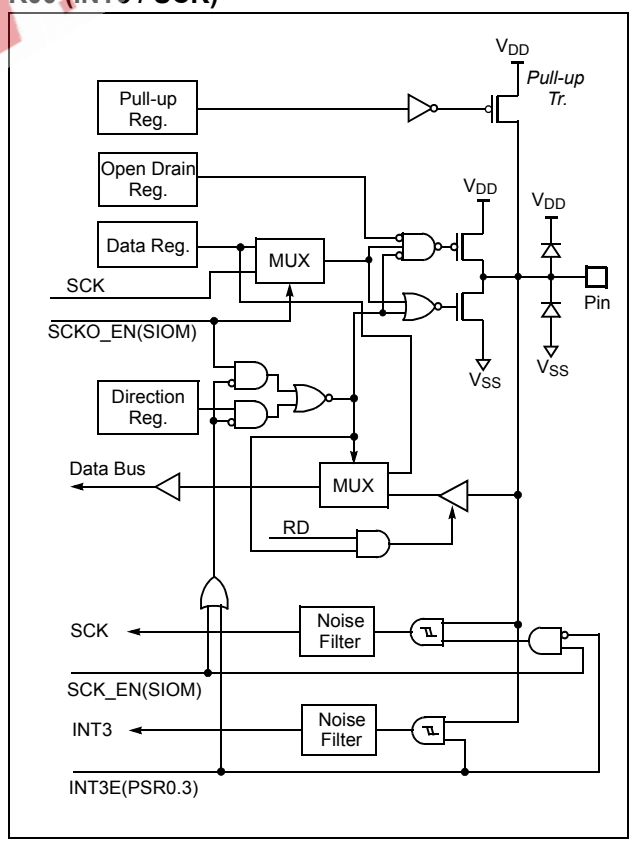
R02 (AN2 / SOUT)



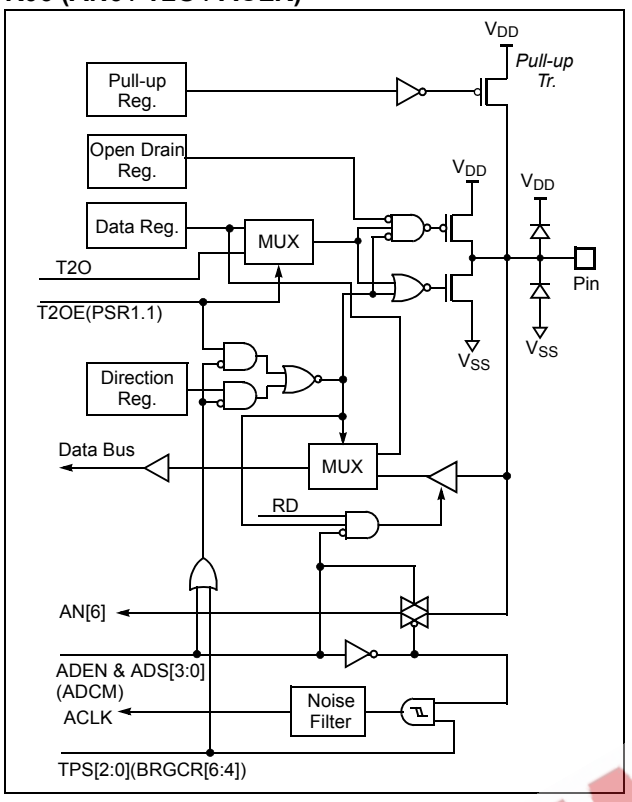
R11 (INT0 / PWM3O), R12 (INT1 / BUZO)



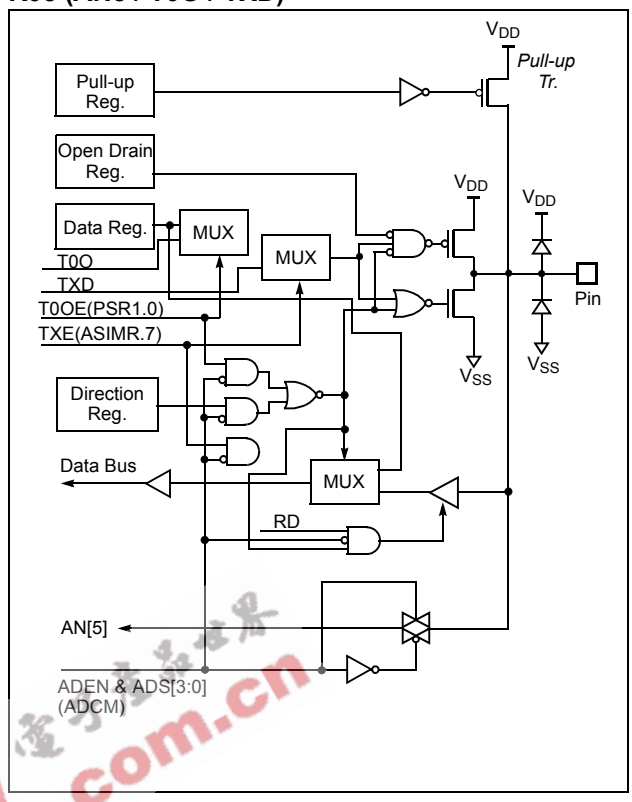
R00 (INT3 / SCK)



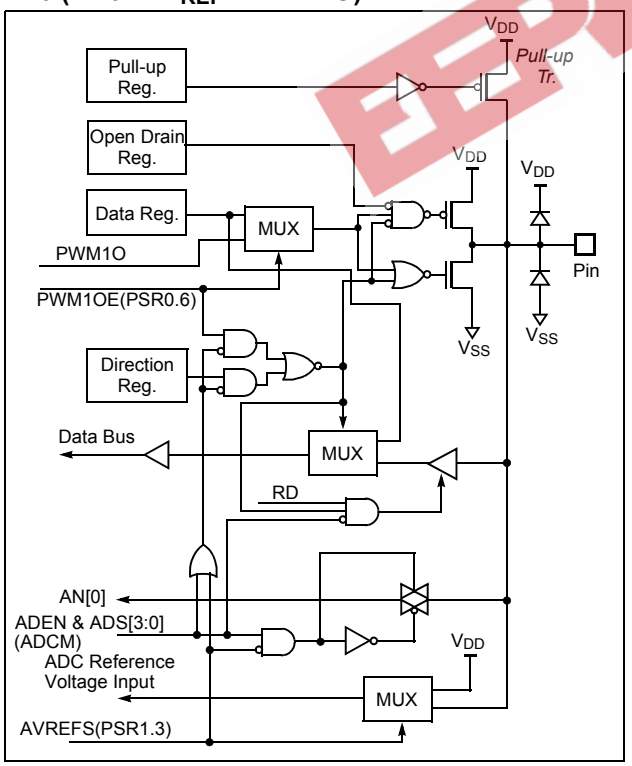
R06 (AN6 / T2O / ACLK)



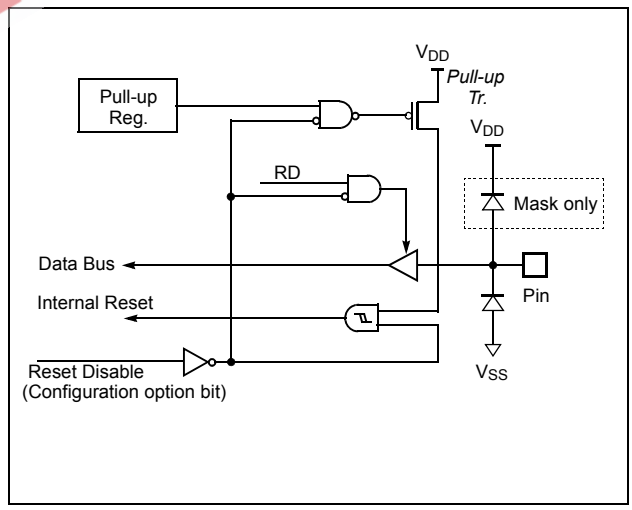
R05 (AN5 / T0O / TXD)



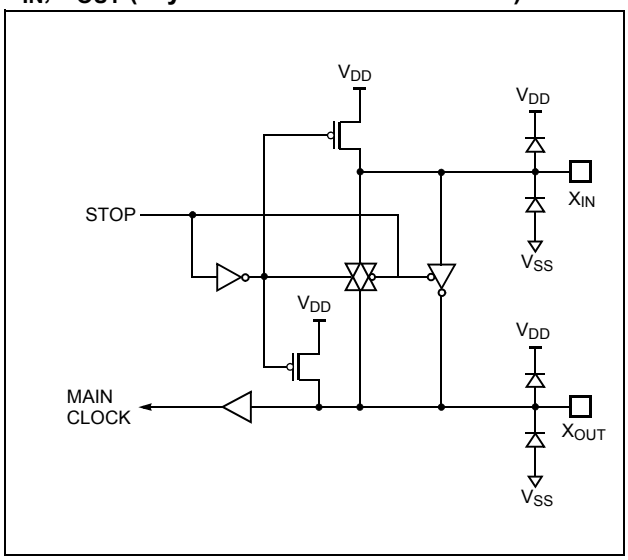
R10 (AN0 / AV_{REF} / PWM1O)



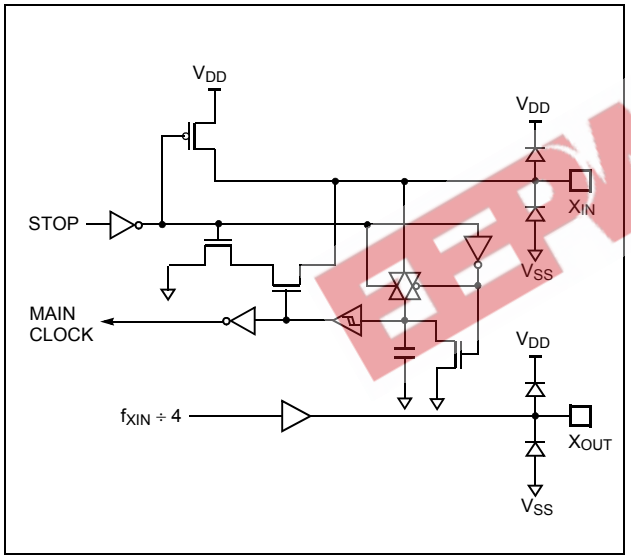
RESET



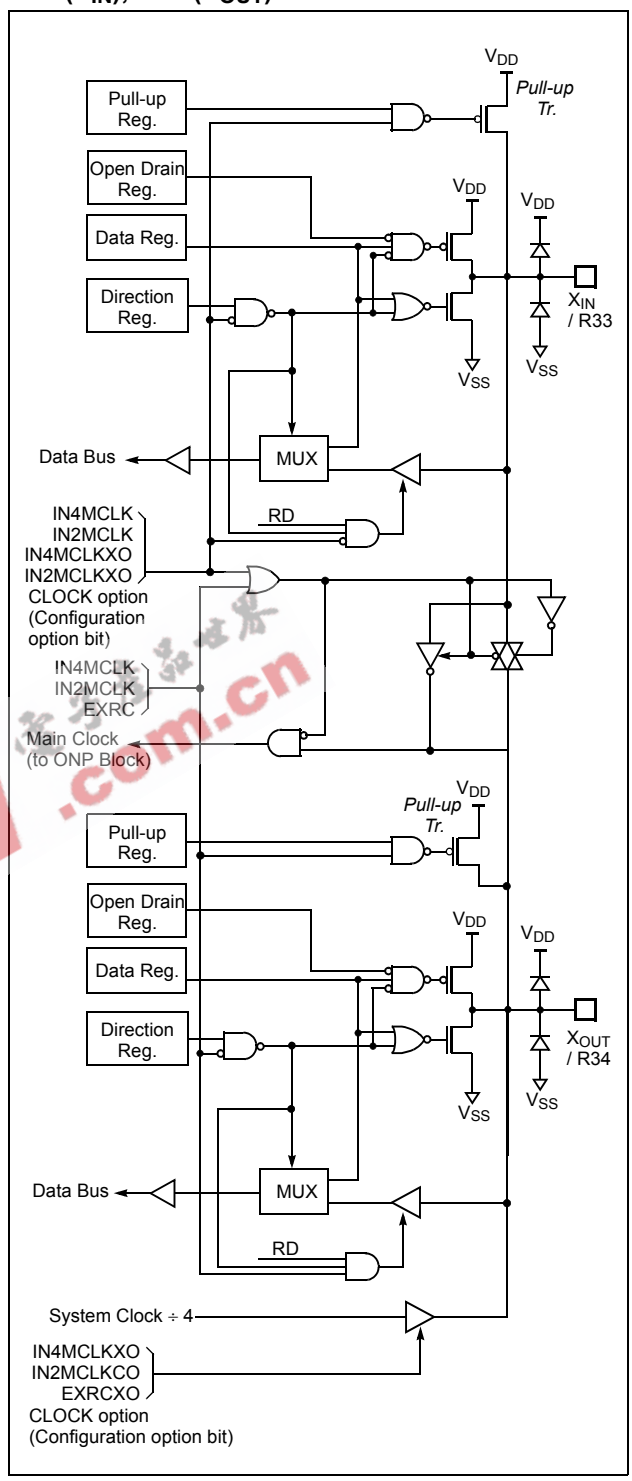
X_{IN}, X_{OUT} (Crystal or Ceramic Resonator)



X_{IN}, X_{OUT} (External RC or R oscillation)



R33 (X_{IN}), R34 (X_{OUT})



7. ELECTRICAL CHARACTERISTICS

7.1 Absolute Maximum Ratings

Supply voltage ..... -0.3 to +6.5 V

Storage Temperature ..... -65 to +150 °C

Voltage on any pin with respect to Ground (V_{SS}) ..... -0.3 to V_{DD}+0.3V

Maximum current out of V_{SS} pin ..... 200 mA

Maximum current into V_{DD} pin ..... 100 mA

Maximum current sunk by (I_{OL} per I/O Pin) ..... 20 mA

Maximum output current sourced by (I_{OH} per I/O Pin) ..... 10 mA

Maximum current (ΣI_{OL}) ..... 160 mA

Maximum current (ΣI_{OH}) ..... 80 mA

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

7.2 Recommended Operating Conditions

Parameter	Symbol	Condition	Min.	Max.	Unit
Supply Voltage	V_{DD}	f_{XIN}=0.4~12MHz f_{XIN}=0.4~8MHz	4.5 2.5	5.5 5.5	V
Operating Frequency	f_{XIN}	V_{DD}=4.5~5.5V V_{DD}=2.7~5.5V(MC80F0X04) V_{DD}=2.0~5.5V(MC80C0X04)	0.4 0.4 0.4	12 8 4.2	MHz
Operating Temperature	T_{OPR}	V_{DD}=2.7~5.5V(MC80F0X04) V_{DD}=2.0~5.5V(MC80C0X04)	-40	85	°C

7.3 A/D Converter Characteristics

(T_a=-40~85°C, V_{SS}=0V, V_{DD}=2.7~5.5V @f_{XIN}=8MHz)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Resolution		-	-	10	-	BIT
Overall Accuracy	-	-	-	-	±3	LSB
Integral Linearity Error	ILE	V_{DD} = AV_{REF} = 5V CPU Clock = 10MHz V_{SS} = 0V	-	–	±3	LSB
Differential Linearity Error	DLE		-	–	±3	LSB
Offset Error of Top	EOT		-	±1	±3	LSB
Offset Error of Bottom	EOB		-	±0.5	±3	LSB
Conversion Time	T_{CONV}	-	13	-	-	μS
Analog Input Voltage	V_{AIN}	-	V_{SS}	-	V_{DD} (AV_{REF})	V
Analog Reference Voltage	AV_{REF}	-	TBD	-	V_{DD}	V
Analog Input Current	I_{AIN}	V_{DD} = AV_{REF} = 5V	-	-	10	μA
Analog Block Current	I_{AVDD}	V_{DD} = AV_{REF} = 5V V_{DD} = AV_{REF} = 3V	- -	1 0.5	3 1.5	mA
		V_{DD} = AV_{REF} = 5V power down mode	-	100	500	nA

7.4 DC Electrical Characteristics

(T_A=-40~85°C, V_{DD}=5.0V, V_{SS}=0V),

Parameter	Symbol	Pin	Condition	Specifications			Unit
				Min.	Typ.	Max.	
Input High Voltage	V_{IH1}	X_{IN}, $\overline{RESET}$		0.8 V_{DD}	-	V_{DD}	V
	V_{IH2}	Hysteresis Input[1]		0.8 V_{DD}	-	V_{DD}	
	V_{IH3}	Normal Input		0.7 V_{DD}	-	V_{DD}	
Input Low Voltage	V_{IL1}	X_{IN}, $\overline{RESET}$		0	-	0.2 V_{DD}	V
	V_{IL2}	Hysteresis Input[1]		0	-	0.2 V_{DD}	
	V_{IL3}	Normal Input		0	-	0.3 V_{DD}	
Output High Voltage	V_{OH}	All Output Port	V_{DD}=5V, I_{OH}=-5mA	V_{DD} -1	-	-	V
Output Low Voltage	V_{OL}	All Output Port	V_{DD}=5V, I_{OL}=10mA	-	-	1	V
Input Pull-up Current	I_P	Normal Input	V_{DD}=5V	-70	-	-130	μA
Input High Leakage Current	I_{IH1}	All Pins (except X_{IN})	V_{DD}=5V	-	-	5	μA
	I_{IH2}	X_{IN}	V_{DD}=5V	-	-	15	μA
Input Low Leakage Current	I_{IL1}	All Pins (except X_{IN})	V_{DD}=5V	-5	-	-	μA
	I_{IL2}	X_{IN}	V_{DD}=5V	-15	-	-	μA
Hysteresis	| V_T |	Hysteresis Input[1]	V_{DD}=5V	0.5	-	-	V
PFD Voltage	V_{PFD}	V_{DD}		2.0	-	3.0	V
POR Voltage	V_{POR}	V_{DD}			2.4		V
POR Start Voltage[2]	V_{START}	V_{DD}		0		TBD	V
POR Rising Time[2]	T_{RISE}	V_{DD}				TBD	V/ms
Internal RC WDT Period	T_{RCWDT}	X_{OUT}	V_{DD}=5.5V	36	-	90	μS
Operating Current	I_{DD}	V_{DD}	V_{DD}=5.5V, f_{XIN}=12MHz	-	6	9	mA
Wake-up Timer Mode Current	I_{WKUP}	V_{DD}	V_{DD}=5.5V, f_{XIN}=12MHz	-	1	2	mA
RCWDT Mode Current at STOP Mode	I_{RCWDT}	V_{DD}	V_{DD}=5.5V	-	20	50	μA
Stop Mode Current	I_{STOP}	V_{DD}	V_{DD}=5.5V, f_{XIN}=12MHz	-	0.7	1.6	μA
Internal Oscillation Frequency	f_{IN_CLK}	X_{OUT}	V_{DD}=5V	3	4	5	MHz
$\overline{RESET}$ Input Noise Cancel Time	T_{RST_NC}	$\overline{RESET}$	V_{DD}=5V	1.5		1.8	μs
External RC Oscillator Frequency	$f_{RC\text{-}OSC}$	f_{XOUT} = $f_{RC\text{-}OSC}$ ÷ 4	V_{DD}=5.5V R=30kΩ, C=10pF		TBD		MHz
	$f_{R\text{-}OSC}$	f_{XOUT} = $f_{R\text{-}OSC}$ ÷ 4	V_{DD}=5.5V, R=30kΩ		TBD		MHz

1. Hysteresis Input: INT0 ~INT3(R11,R12,R03,R00),SIO(R00,R01,R02),UART(R04,R06),EC0,EC1
2. V_{START} and T_{RISE} parameter is presented for design guidance only and not tested or guaranteed.

7.5 AC Characteristics

(T_A=-40~+85°C, V_{DD}=5V±10%, V_{SS}=0V)

Parameter	Symbol	Pins	Specifications			Unit
			Min.	Typ.	Max.	
Operating Frequency	f_{CP}	X_{IN}	1	-	8	MHz
External Clock Pulse Width	t_{CPW}	X_{IN}	50	-	-	nS
External Clock Transition Time	t_{RCP},t_{FCP}	X_{IN}	-	-	20	nS
Oscillation Stabilizing Time	t_{ST}	X_{IN}, X_{OUT}	-	-	20	mS
External Input Pulse Width	t_{EPW}	INT0, INT1, INT2, INT3 EC0, EC1	2	-	-	t_{SYS}
$\overline{\text{RESET}}$ Input Width	t_{RST}	$\overline{\text{RESET}}$	8	-	-	t_{SYS}

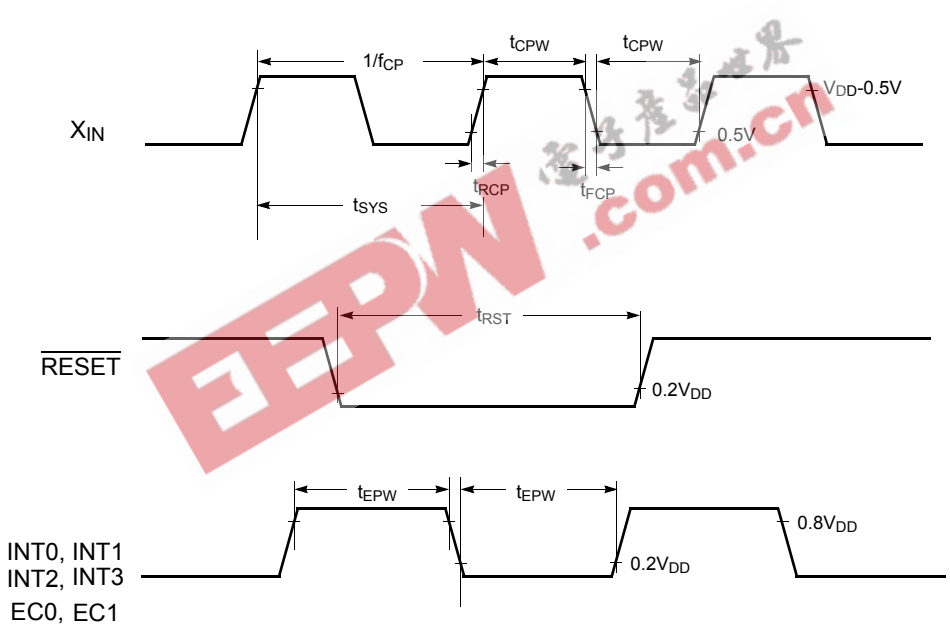


Figure 7-1 Timing Chart

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 + 3σ) and (mean – 3σ) respectively where σ is standard deviation

TBD

8. MEMORY ORGANIZATION

The MC80F0104/0204 has separate address spaces for Program memory and Data Memory. 4K bytes program memory can only be read, not written to.

Data memory can be read and written to up to 256 bytes including the stack area.

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.

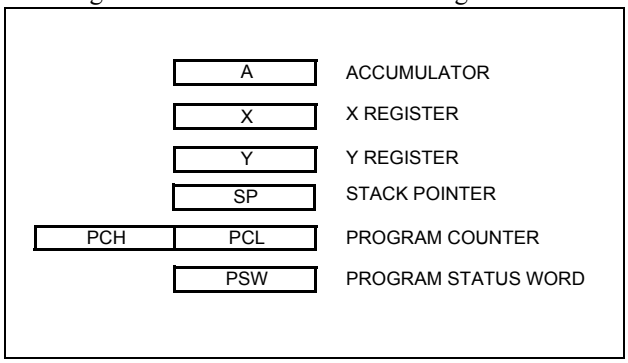


Figure 8-1 Configuration of Registers

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.

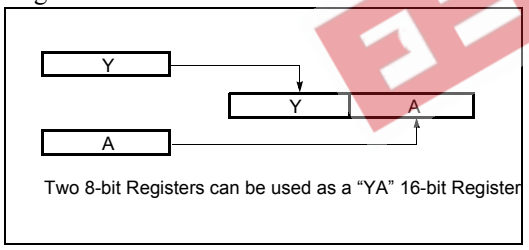


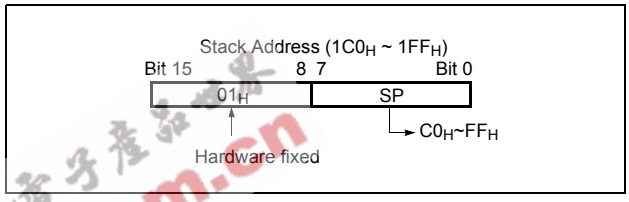
Figure 8-2 Configuration of YA 16-bit Register

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 $1C0_H$ to $1FF_H$ 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 "FF_H" 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 (PC_H:$0FF_H$, PC_L:$0FE_H$).

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

[Carry flag C]

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

[Zero flag Z]

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

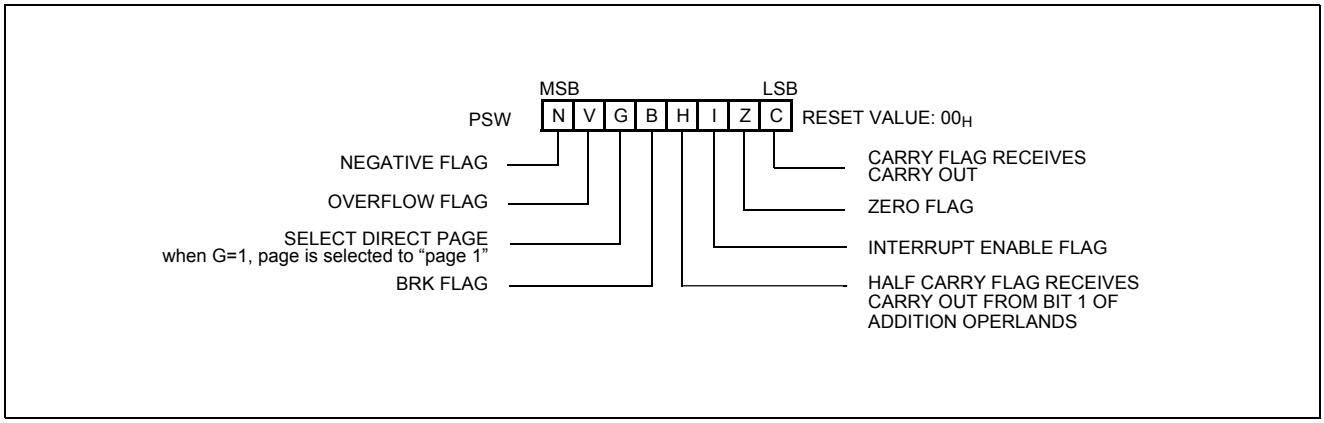


Figure 8-3 PSW (Program Status Word) Register

[Interrupt disable flag I]

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

[Half carry flag H]

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

[Break flag B]

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

[Direct page flag G]

This flag assigns RAM page for direct addressing mode. In the direct addressing mode, addressing area is from zero page 00_H to $0FF_H$ when this flag is "0". If it is set to "1", addressing area is assigned 100_H to $1FF_H$. It is set by SETG instruction and cleared by CLRG.

[Overflow flag V]

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

[Negative flag N]

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

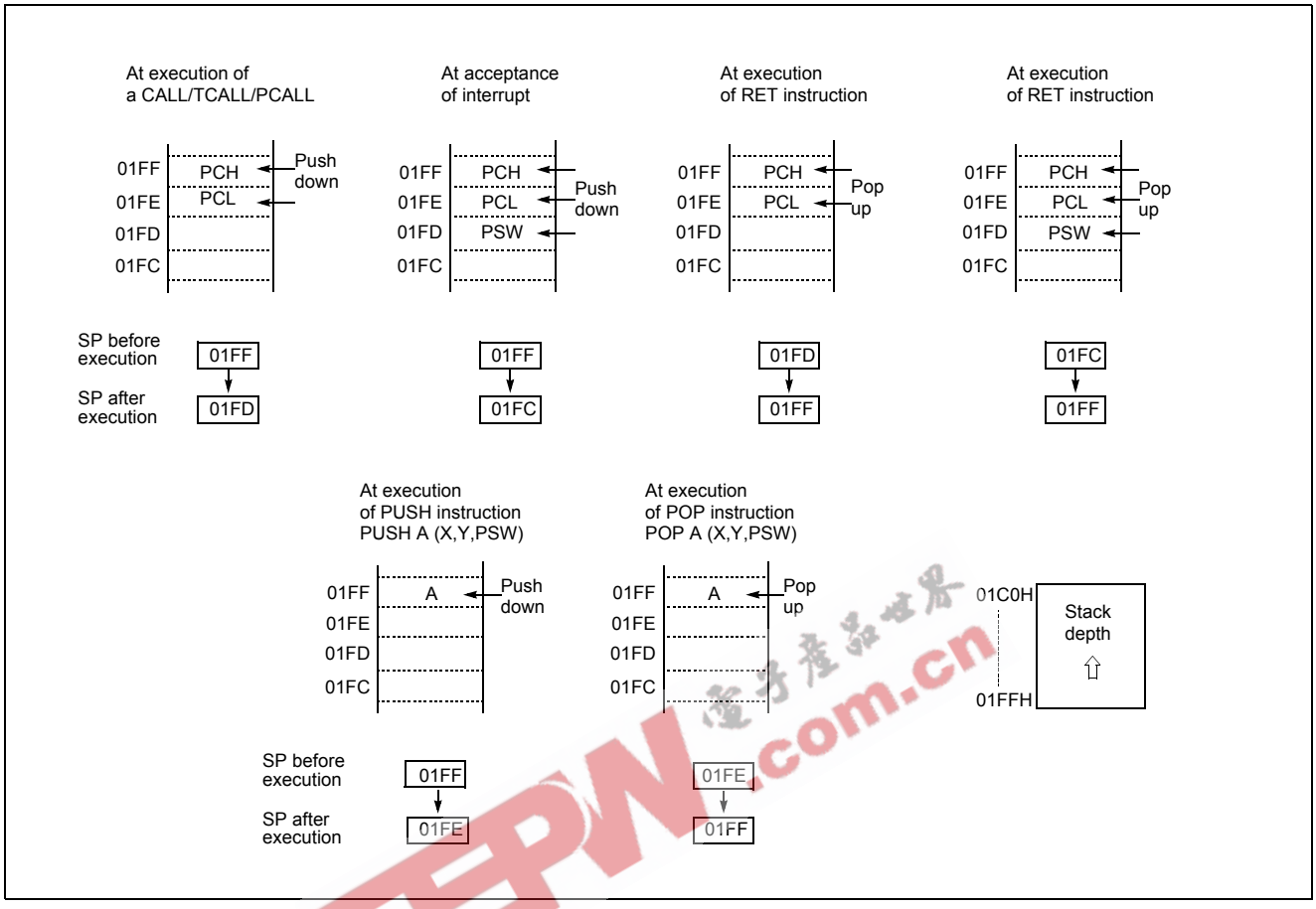


Figure 8-4 Stack Operation

8.2 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has 4K bytes program memory space only physically implemented. Accessing a location above $FFFF_H$ will cause a wrap-around to 0000_H.

Figure 8-5 , shows a map of Program Memory. After reset, the CPU begins execution from reset vector which is stored in address $FFFE_H$ and $FFFF_H$ 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

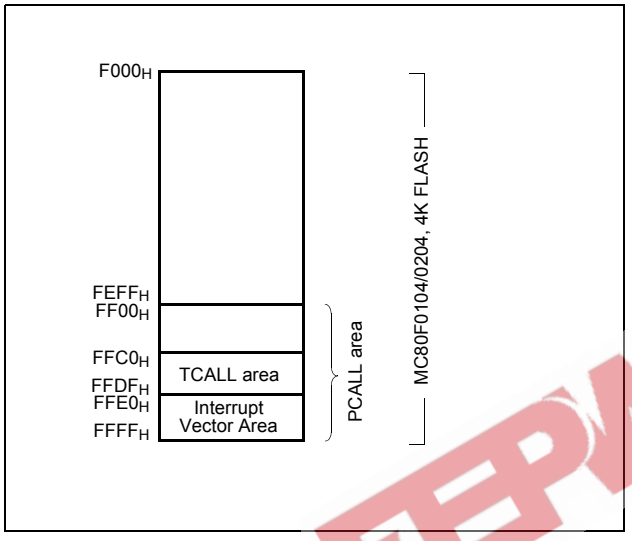


Figure 8-5 Program Memory Map

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: $0FFC0_H$ for TCALL15, $0FFC2_H$ 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)  (1)
          RET
;
;TABLE CALL ADD. AREA
;
          ORG    0FFC0H       ;TCALL ADDRESS AREA
          DW     FUNC_A
          DW     FUNC_B
```

The interrupt causes the CPU to jump to specific location, where it commences the execution of the service routine. The External interrupt 0, for example, is assigned to location $0FFFC_H$. The interrupt service locations spaces 2-byte interval: $0FFFA_H$ and $0FFFB_H$ for External Interrupt 1, $0FFFC_H$ and $0FFFD_H$ for External Interrupt 0, etc.

Any area from $0FF00_H$ to $0FFFF_H$, if it is not going to be used, its service location is available as general purpose Program Memory.

Address	Vector Area Memory
$0FFE0_H$	Basic Interval Timer
E2	Watchdog Timer Interrupt
E4	A/D Converter
E6	-
E8	Timer/Counter 3 Interrupt
EA	Timer/Counter 2 Interrupt
EC	Timer/Counter 1 Interrupt
EE	Timer/Counter 0 Interrupt
F0	Serial Input/Output (SIO)
F2	UART Tx interrupt
F4	UART Rx interrupt
F6	External Interrupt 3
F8	External Interrupt 2
FA	External Interrupt 1
FC	External Interrupt 0
FE	RESET

Figure 8-6 Interrupt Vector Area

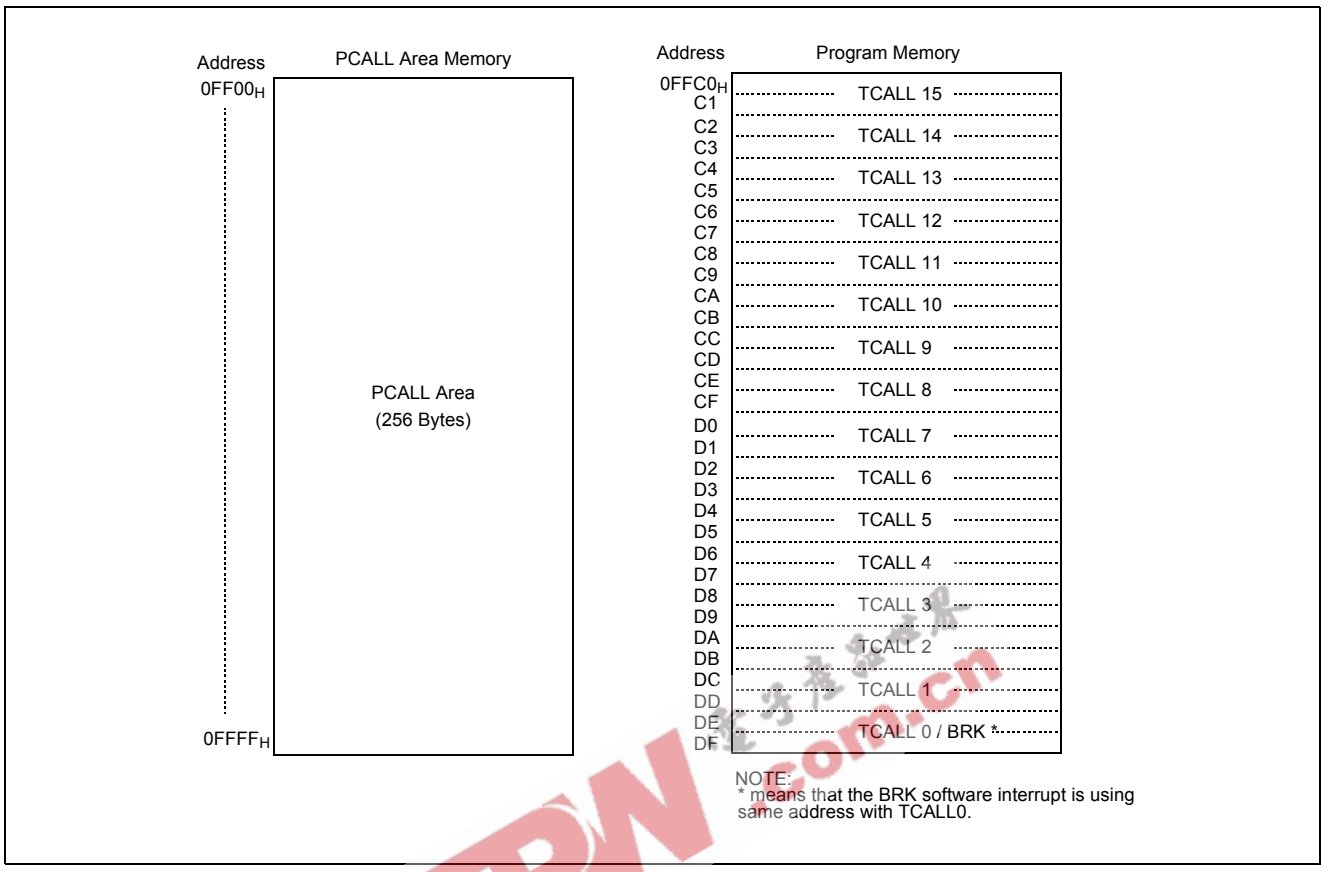
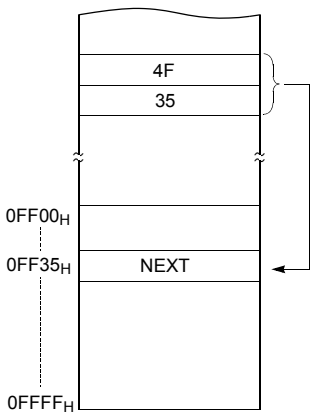


Figure 8-7 PCALL and TCALL Memory Area

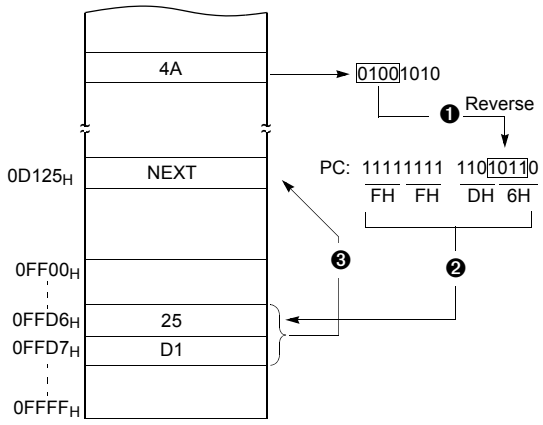
PCALL→ rel

```
4F35      PCALL  35H
```



TCALL→ n

```
4A        TCALL  4
```



Example: The usage software example of Vector address for MC80F0204.

```
;Interrupt Vector Table
        ORG    0FFE0H
        DW     BIT_TIMER   ; BIT
        DW     WDT         ; WDT
        DW     ADC         ; AD Converter
        DW     Not_used    ;
        DW     TIMER3      ; Timer-3
        DW     TIMER2      ; Timer-2
        DW     TIMER1      ; Timer-1
        DW     TIMER0      ; Timer-0
        DW     SIO         ; Serial Interface
        DW     TX          ; UART Tx
        DW     RX          ; UART Rx
        DW     INT3        ; Ext Int.3
        DW     INT2        ; Ext Int.2
        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.

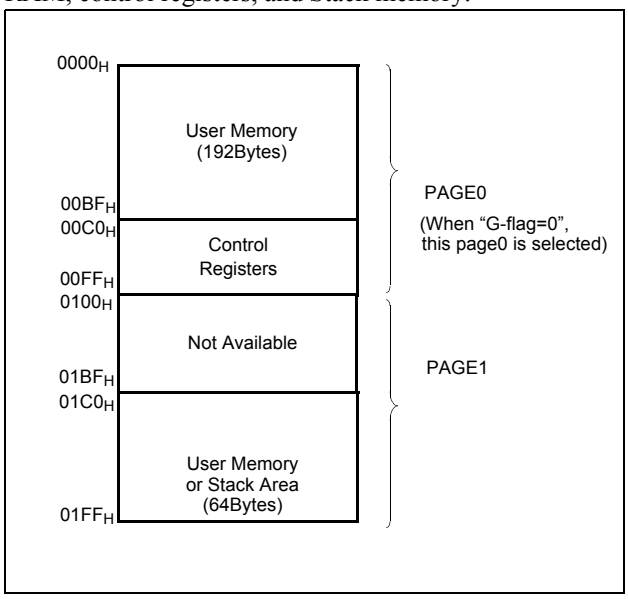


Figure 8-8 Data Memory Map

User Memory

The MC80F0104/0204 has 256 × 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 $0C0_H$ to $0FF_H$.

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 on page 20.

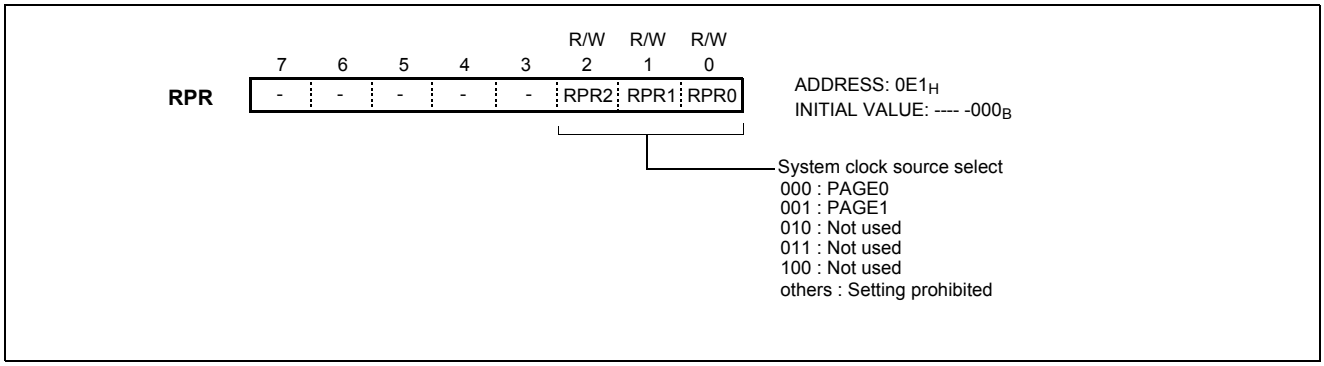


Figure 8-9 RPR(RAM Page Select Register)

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	byte, bit[1]
00C1	R0 port I/O direction register	R0IO	W	0	0	0	0	0	0	0	0	byte[2]
00C2	R1 port data register	R1	R/W	-	-	-	0	0	0	0	0	byte, bit
00C3	R1 port I/O direction register	R1IO	W	-	-	-	0	0	0	0	0	byte
00C6	R3 port data register	R3	R/W	-	-	0	0	0	0	0	-	byte, bit
00C7	R3 port I/O direction register	R3IO	W	0	0	0	0	0	0	0	-	byte
00C8	Port 0 Open Drain Selection Register	R0OD	R/W	0	0	0	0	0	0	0	0	byte
00C9	Port 1 Open Drain Selection Register	R1OD	W	-	-	-	0	0	0	0	0	byte
00CB	Port 3 Open Drain Selection Register	R3OD	W	-	-	-	0	0	0	0	-	byte
00D0	Timer 0 mode control register	TM0	R/W	-	-	0	0	0	0	0	0	byte, bit
00D1	Timer 0 register	T0	R	0	0	0	0	0	0	0	0	byte
	Timer 0 data register	TDR0	W	1	1	1	1	1	1	1	1	
	Timer 0 capture data register	CDR0	R	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	byte, bit
00D3	Timer 1 data register	TDR1	W	1	1	1	1	1	1	1	1	byte
	Timer 1 PWM period register	T1PPR	W	1	1	1	1	1	1	1	1	byte
00D4	Timer 1 register	T1	R	0	0	0	0	0	0	0	0	byte
	Timer 1 capture data register	CDR1	R	0	0	0	0	0	0	0	0	
	Timer 1 PWM duty register	T1PDR	R/W	0	0	0	0	0	0	0	0	byte
00D5	Timer 1 PWM high register	T1PWHR	W	-	-	-	-	0	0	0	0	byte
00D6	Timer 2 mode control register	TM2	R/W	-	-	0	0	0	0	0	0	byte, bit
00D7	Timer 2 register	T2	R	0	0	0	0	0	0	0	0	byte
	Timer 2 data register	TDR2	W	1	1	1	1	1	1	1	1	
	Timer 2 capture data register	CDR2	R	0	0	0	0	0	0	0	0	
00D8	Timer 3 mode control register	TM3	R/W	0	0	0	0	0	0	0	0	byte, bit
00D9	Timer 3 data register	TDR3	W	1	1	1	1	1	1	1	1	byte
	Timer 3 PWM period register	T3PPR	W	1	1	1	1	1	1	1	1	
00DA	Timer 3 register	T3	R	0	0	0	0	0	0	0	0	byte
	Timer 3 PWM duty register	T3PDR	R/W	0	0	0	0	0	0	0	0	
	Timer 3 capture data register	CDR3	R	0	0	0	0	0	0	0	0	
00DB	Timer 3 PWM high register	T3PWHR	W	-	-	-	-	0	0	0	0	byte
00E0	Buzzer driver register	BUZR	W	1	1	1	1	1	1	1	1	byte

Table 8-1 Control Registers

Address	Register Name	Symbol	R/W	Initial Value								Addressing Mode
				7	6	5	4	3	2	1	0	
00E1	RAM page selection register	RPR	R/W	-	-	-	-	-	0	0	0	byte, bit
00E2	SIO mode control register	SIOM	R/W	0	0	0	0	0	0	0	1	byte, bit
00E3	SIO data shift register	SIOR	R/W	Undefined								byte, bit
00E6	UART mode register	ASIMR	R/W	0	0	0	0	-	0	0	-	byte, bit
00E7	UART status register	ASISR	R	-	-	-	-	-	0	0	0	byte
00E8	UART Baud rate generator control register	BRGCR	R/W	-	0	0	1	0	0	0	0	byte, bit
00E9	UART Receive buffer register	RXBR	R	0	0	0	0	0	0	0	0	byte
	UART Transmit shift register	TXSR	W	1	1	1	1	1	1	1	1	
00EA	Interrupt enable register high	IENH	R/W	0	0	0	0	0	0	0	0	byte, bit
00EB	Interrupt enable register low	IENL	R/W	0	0	0	0	0	0	0	0	byte, bit
00EC	Interrupt request register high	IRQH	R/W	0	0	0	0	0	0	0	0	byte, bit
00ED	Interrupt request register low	IRQL	R/W	0	0	0	0	0	0	0	0	byte, bit
00EE	Interrupt edge selection register	IEDS	R/W	0	0	0	0	0	0	0	0	byte, bit
00EF	A/D converter mode control register	ADCM	R/W	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	
00F4	Watch dog timer register	WDTR	W	0	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	byte
00F7	PFD control register	PFDR	R/W	-	-	-	-	-	0	0	0	byte, bit
00F8	Port selection register 0	PSR0	W	0	0	0	0	0	0	0	0	byte
00F9	Port selection register 1	PSR1	W	-	-	-	-	0	0	0	0	byte
00FC	Pull-up selection register 0	PU0	W	0	0	0	0	0	0	0	0	byte
00FD	Pull-up selection register 1	PU1	W	-	-	-	0	0	0	0	0	byte
00FF	Pull-up selection register 3	PU3	W	-	-	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 register except T1PDR and T3PDR 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	Timer1 Register / Timer1 Capture Data Register							
0D5H	PWM1HR	-	-	-	-	Timer1 PWM High Register			
0D6H	TM2	-	-	CAP2	T2CK2	T2CK1	T2CK0	T2CN	T2ST
0D7H	T2/TDR2/ CDR2	Timer2 Register / Timer2 Data Register / Timer2 Capture Data Register							
0D8H	TM3	POL	16BIT	PWM3E	CAP3	T3CK1	T3CK0	T3CN	T3ST
0D9H	TDR3/ T3PPR	Timer3 Data Register / Timer3 PWM Period Register							
0DAH	T3/CDR3/ T3PDR	Timer3 Register / Timer3 Capture Data Register / Timer3 PWM Duty Register							
0DBH	PWM3HR	-	-	-	-	Timer3 PWM High Register			
0E0H	BUZR	BUCK1	BUCK0	BUR5	BUR4	BUR3	BUR2	BUR1	BUR0
0E1H	RPR	-	-	-	-	-	RPR2	RPR1	RPR0
0E2H	SIOM	POL	IOSW	SM1	SM0	SCK1	SCK0	SIOST	SIOSF
0E3H	SIOR	SIO Data Shift Register							
0E6H	ASIMR	TXE	RXE	PS1	PS0	-	SL	ISRM	-
0E7H	ASISR	-	-	-	-	-	PE	FE	OVE
0E8H	BRGCR	-	TPS2	TPS1	TPS0	MLD3	MLD2	MLD1	MLD0
0E9H	RXR	UART Receive Buffer Register							
	TXR	UART Transmit Shift Register							
0EAH	IENH	INT0E	INT1E	INT2E	INT3E	RXE	TXE	SIOE	T0E
0EBH	IENL	T1E	T2E	T3E	T4E	ADCE	WDTE	WTE	BITE

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
0ECH	IRQH	INT0IF	INT1IF	INT2IF	INT3IF	RXIF	TXIF	SIOIF	T0IF
0EDH	IRQL	T1IF	T2IF	T3IF	T4IF	ADCIF	WDTIF	WTIF	BITIF
0EEH	IEDS	IED3H	IED3L	IED2H	IED2L	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[1]	Basic Interval Timer Data Register							
	CKCTLR[1]	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	PWM3O	PWM1O	EC1E	EC0E	INT3E	INT2E	INT1E	INT0E
0F9H	PSR1	-	-	-	-	AVREFS	BUZO	T2O	T0O
0FCH	PU0	R0 Pull-up Selection Register							
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 CKCTLR are located at same address. Address ECH is read as BITR, written to CKCTLR.

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 HMS800 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

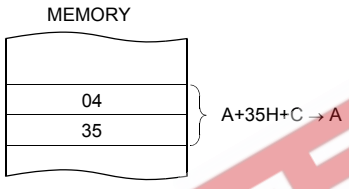
Register addressing accesses the A, X, Y, C and PSW.

8.4.2 Immediate Addressing → #imm

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

Example:

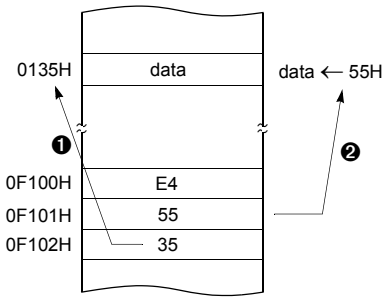
```
0435    ADC    #35H
```



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
```

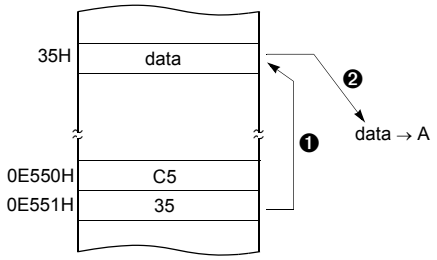


8.4.3 Direct Page Addressing→ dp

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

Example; G=0

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



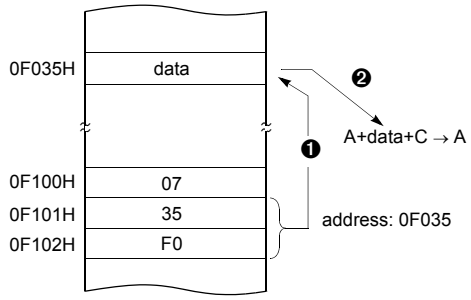
8.4.4 Absolute Addressing → !abs

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;

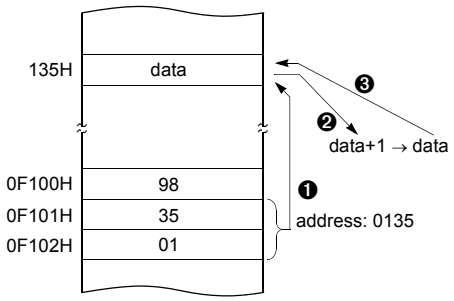
```
0735F0  ADC    !0F035H      ;A ←ROM[0F035H]
```

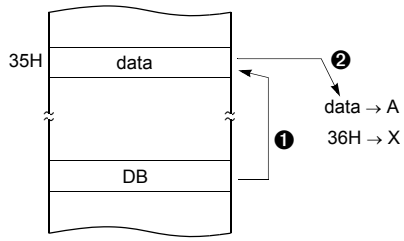


The operation within data memory (RAM)
ASL, BIT, DEC, INC, LSR, ROL, ROR

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

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





8.4.5 Indexed Addressing

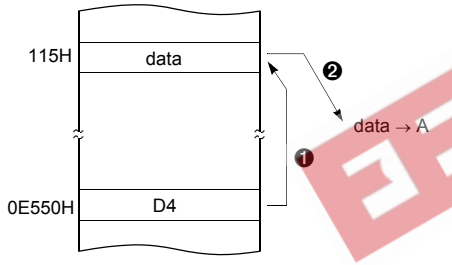
X indexed direct page (no offset) → {X}

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

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

Example; X=15$_H$, G=1

```
D4        LDA    {X}          ;ACC←RAM[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=35$_H$

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

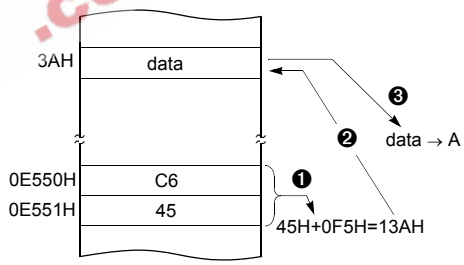
X indexed direct page (8 bit offset) → dp+X

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=0F5$_H$

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



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

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

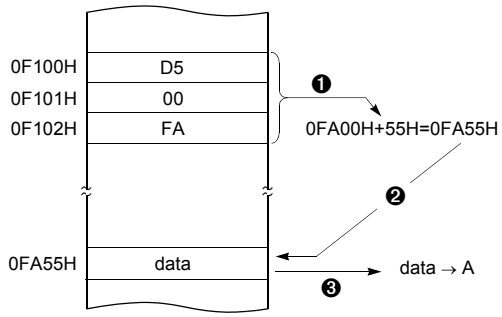
This is same with above (2). Use Y register instead of X.

Y indexed absolute → !abs+Y

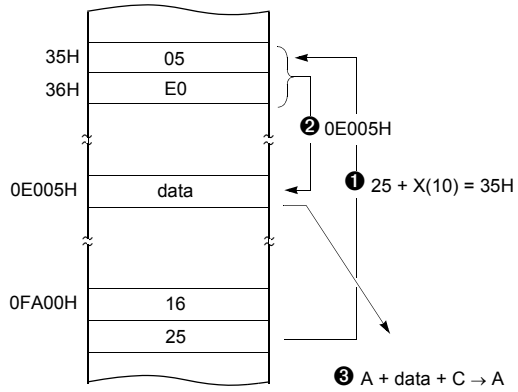
Sets the value of 16-bit absolute address plus Y-register data as Memory.This addressing mode can specify memory in whole area.

Example; Y=55$_H$

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




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



8.4.6 Indirect Addressing

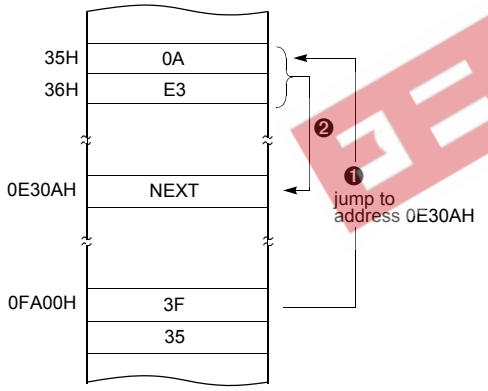
Direct page indirect → [dp]

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=10_H

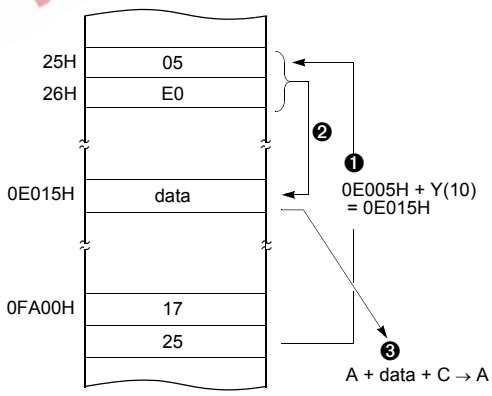
Y indexed indirect → [dp]+Y

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=10_H

```
1725     ADC    [25H]+Y
```



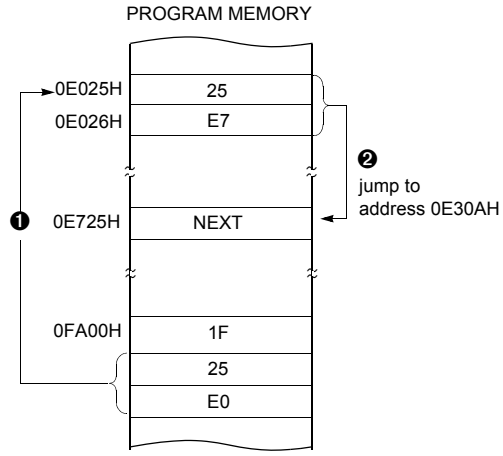
Absolute indirect → [!abs]

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

JMP

Example; G=0

```
1F25E0   JMP    [!0C025H]
```



9. I/O PORTS

The MC80F0104/0204 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 "55_H" to address $0C1_H$ (R0 port direction register) during initial setting as shown in Figure 9-1 .

All the port direction registers in the MC80F0104/0204 have 0 written to them by reset function. On the other hand, its initial status is input.

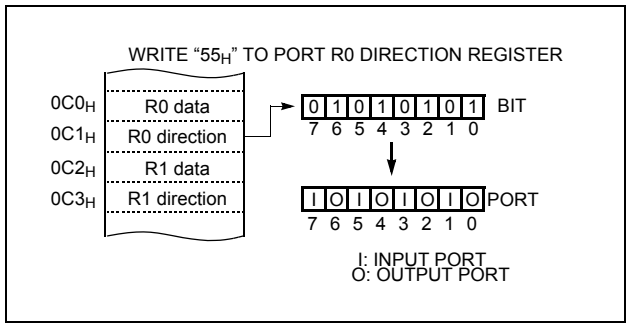


Figure 9-1 Example of port I/O assignment

9.1 R0 and R0IO register

R0 is an 8-bit CMOS bidirectional I/O port (address $0C0_H$). Each I/O pin can independently used as an input or an output through the R0IO register (address $0C1_H$). 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).

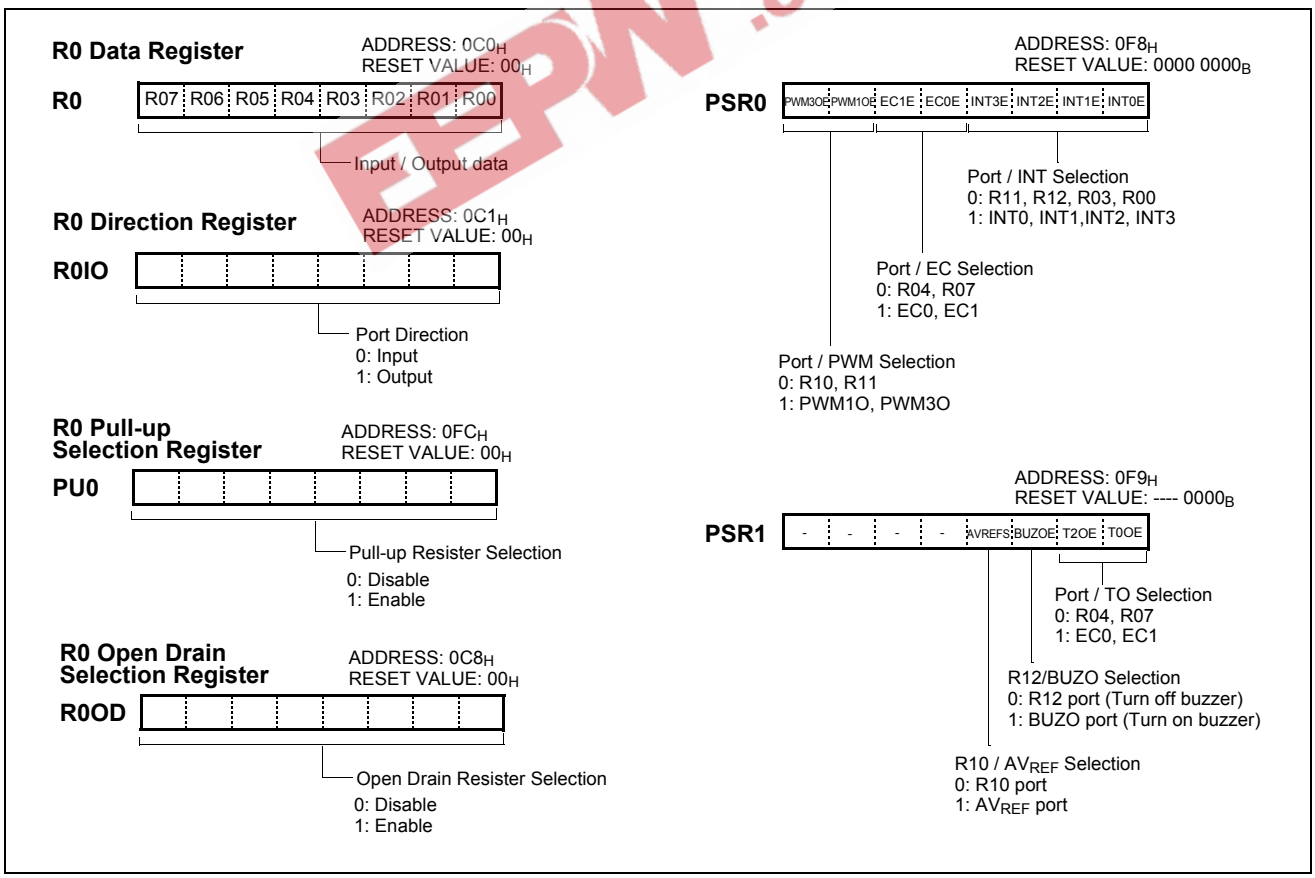


Figure 9-2 R0 Port Register

In addition, Port R0 is multiplexed with various alternate functions. The port selection register PSR0 (address $0F8_H$) and PSR1 (address $0F9_H$) control the selection of alternate functions such as external interrupt 3 (INT3), external interrupt 2 (INT2), event counter input 0 (EC0), timer 0 output (T0O), timer 2 output (T2O) and event counter input 1 (EC1). 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), SIO data input (SI), SIO data output (SOUT) and UART data input (RXD), UART data output (TXD) and UART clock input (ACLK) can be selected by setting ADCM($00EF_H$), SIOM($00E2_H$) and ASIMR($00E6_H$) register to enable the corresponding peripheral operation and select operation mode.

Port Pin	Alternate Function
R00	INT3 (External interrupt 3) SCK (SIO clock input/output)
R01	AN1(ADC Input channel 1) SI (SIO data input)
R02	AN2 (ADC Input channel 2) SOUT (SIO data output)
R03	AN3 (ADC Input channel 3) INT2 (External interrupt 2)
R04	AN4 (ADC Input channel 4) EC0 (Event counter input 0) RXD (UART data input)
R05	AN5 (ADC Input channel 5) T0O (Timer output 0) TXD (UART data output)
R06	AN6 (ADC Input channel 6) T2O (Timer output 2) ACLK (UART clock input)
R07	AN7 (ADC Input channel 7) EC1 (Event counter input 1)

9.2 R1 and R1IO register

R1 is a 5-bit CMOS bidirectional I/O port (address $0C2_H$). Each I/O pin can independently used as an input or an output through the R1IO register (address $0C3_H$). 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 $0F8_H$) and PSR1 (address $0F9_H$) control the selection of alternate functions such as Analog reference voltage input (AV_{REF}), external interrupt 0 (INT0), external interrupt 1 (INT1), PWM 1 output (PWM1O), PWM 3 output (PWM3O) 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($00EF_H$) register to enable ADC and select channel 0 .

Port Pin	Alternate Function
R10	AN0 (ADC input channel 0) AV_{REF} (Analog reference voltage) PWM1O (PWM 1 output)
R11	INT0 (External Interrupt 0) PWM3O (PWM 3 output)
R12	INT1 (External Interrupt 1) BUZO (Buzzer output)

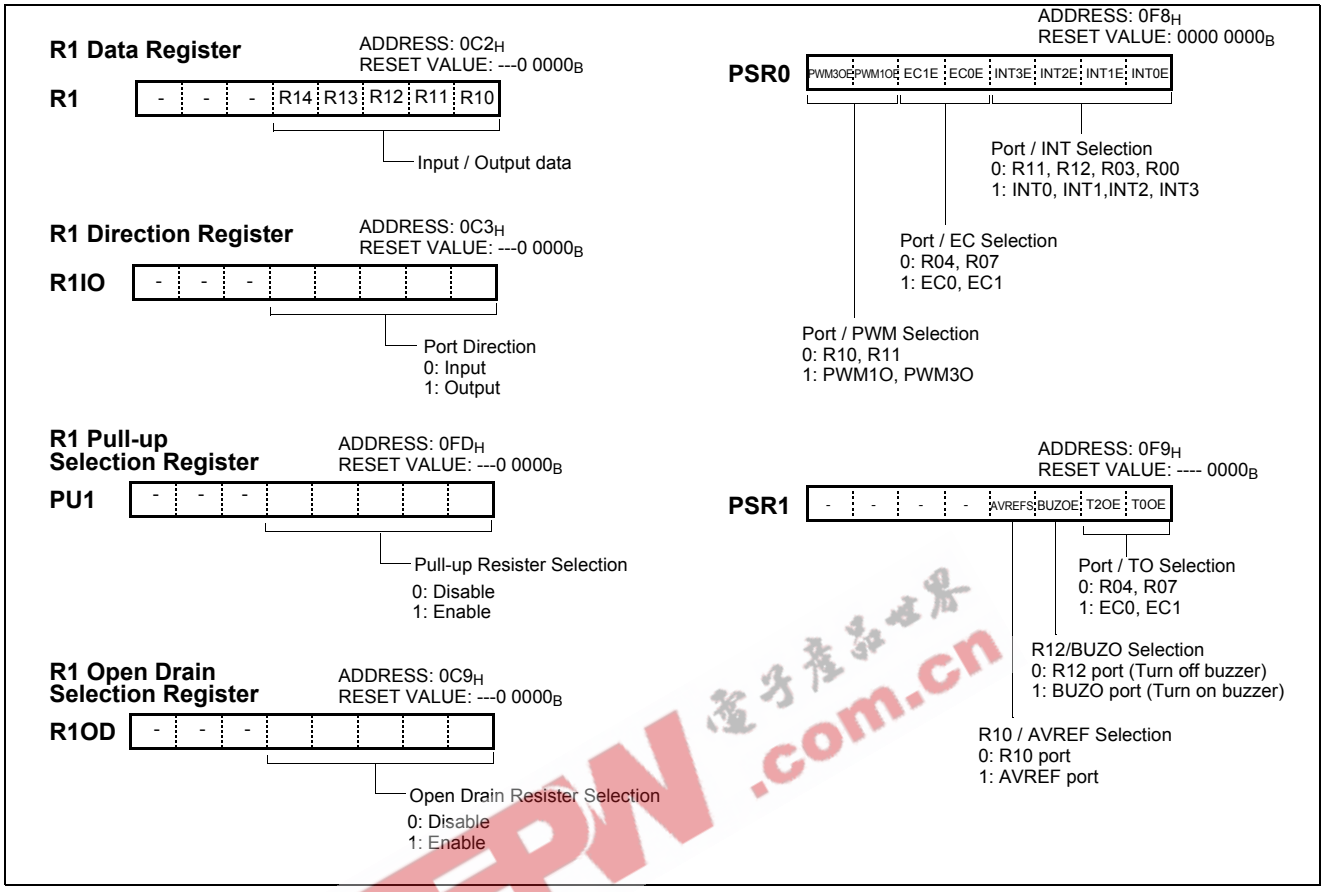


Figure 9-3 R1 Port Register

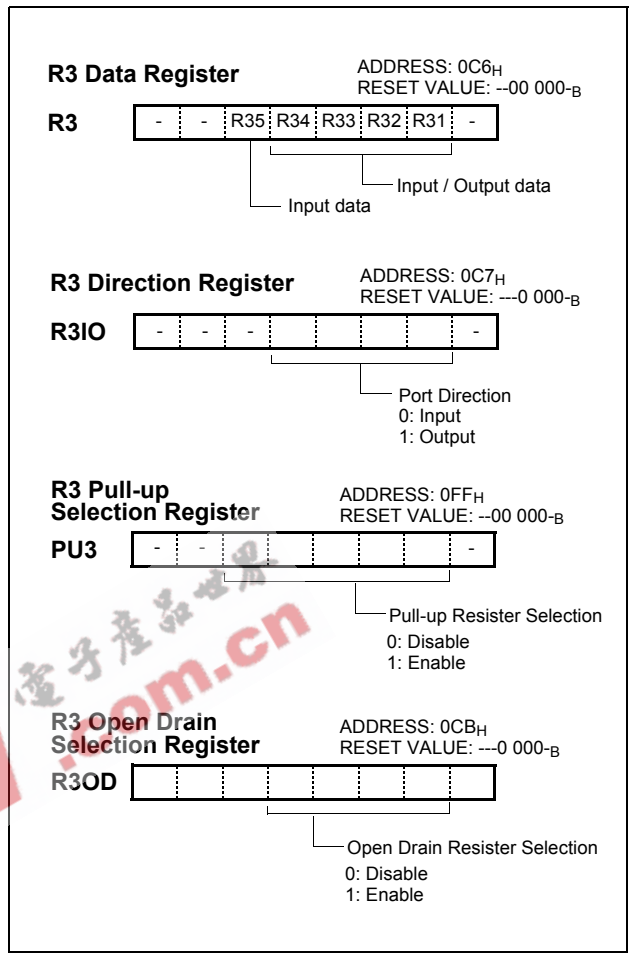
9.3 R3 and R3IO register

R3 is a 5-bit CMOS bidirectional I/O port (address $0C6_H$). Each I/O pin (except R35) can independently used as an input or an output through the R3IO register (address $0C7_H$). 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 R32	AN14 (ADC input channel 14) AN15 (ADC input channel 15)

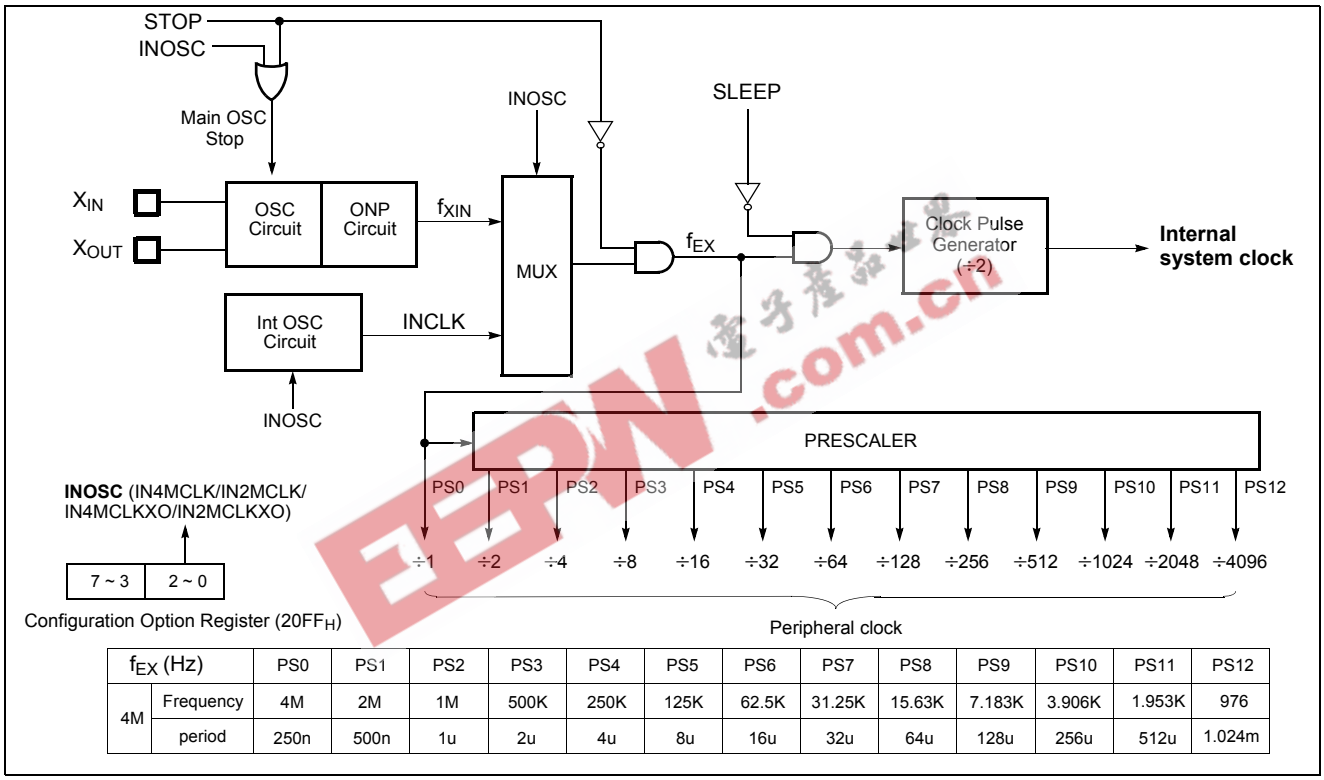
R33, R34 and R35 is multiplexed with X_{IN}, X_{OUT}, and $\overline{RESET}$ pin. These pins can be used as general I/O pins by setting writing option described in "23. Device Configuration Area" on page 102.



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_{IN} and X_{OUT} pin, respectively. The system clock can also be obtained from the external oscillator. In this case, it is necessary to input a external clock signal to the X_{IN} pin and open the X_{OUT} pin. 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.

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 39 for details.



fEX (Hz)		PS0	PS1	PS2	PS3	PS4	PS5	PS6	PS7	PS8	PS9	PS10	PS11	PS12
4M	Frequency	4M	2M	1M	500K	250K	125K	62.5K	31.25K	15.63K	7.183K	3.906K	1.953K	976
	period	250n	500n	1u	2u	4u	8u	16u	32u	64u	128u	256u	512u	1.024m

Figure 10-1 Block Diagram of Clock Generator

10.1 Oscillation Circuit

X_{IN} and X_{OUT} 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 .

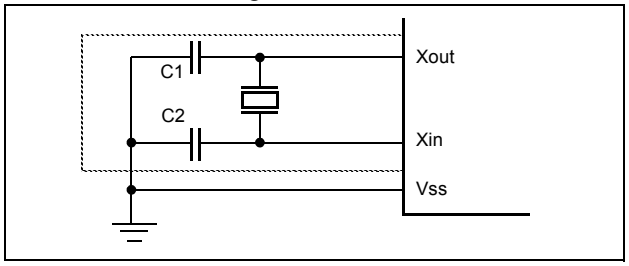


Figure 10-2 Oscillator Connections

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 Vss. Do not ground to any ground pattern where high current is present.
- Do not fetch signals from the oscillator.

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

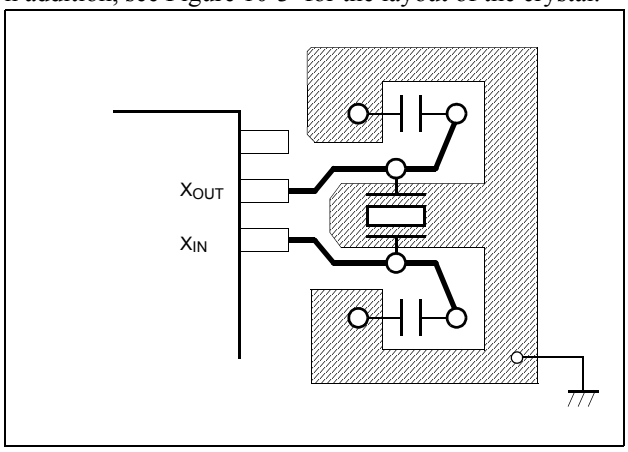


Figure 10-3 Layout of Oscillator PCB circuit

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.

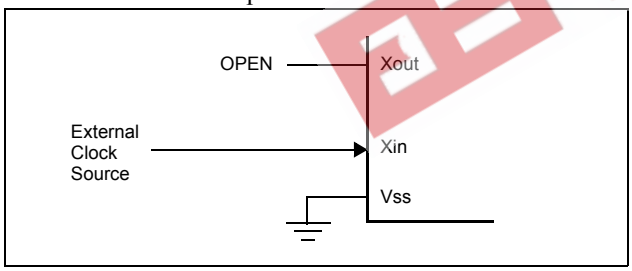


Figure 10-4 External Clock Connections

In addition, the MC80F0104/0204 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 MC80F0104/0204. 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.

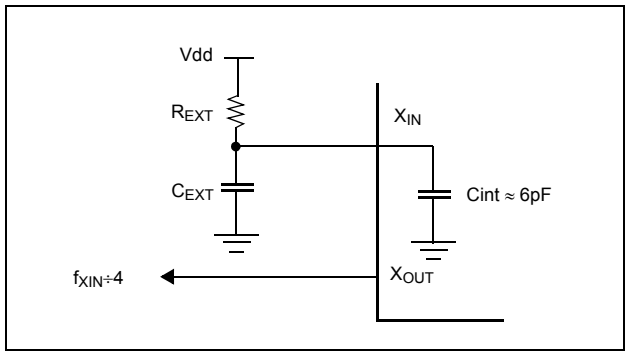


Figure 10-5 RC Oscillator Connections

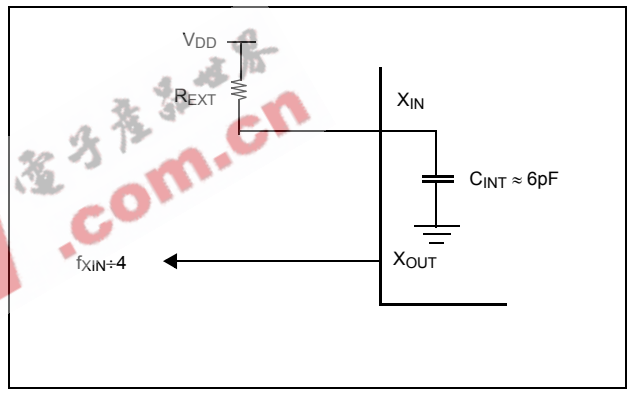


Figure 10-6 R Oscillator Connections

To use the RC oscillation , the CLK option of the configuration bits ($20FF_H$) 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 MC80F0104/0204 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 "23. Device Configuration Area" on page 102

11. BASIC INTERVAL TIMER

The MC80F0104/0204 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 (CKCTLR) 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 CKCTLR, 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 CKCTLR, 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 CKCTLR.

BITR and CKCTLR are located at same address, and address $0F2_H$ is read as a BITR, and written to CKCTLR.

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

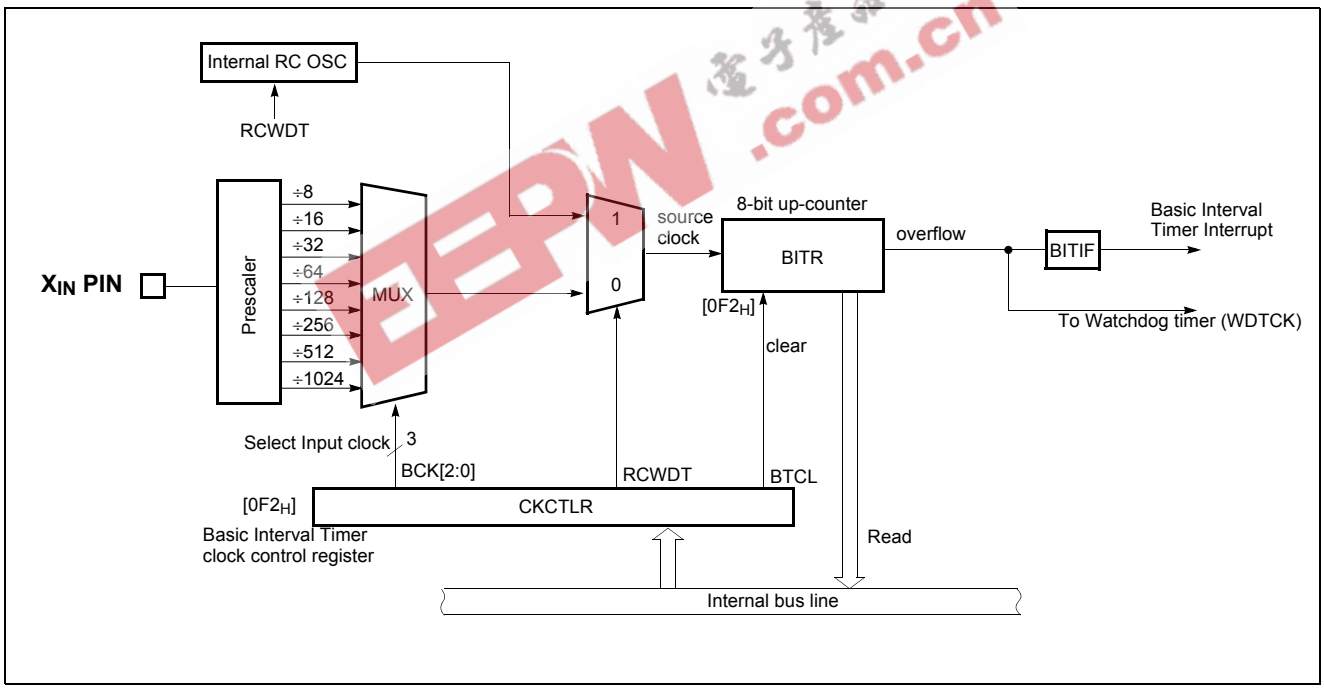


Figure 11-1 Block Diagram of Basic Interval Timer

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

Table 11-1 Basic Interval Timer Interrupt Period

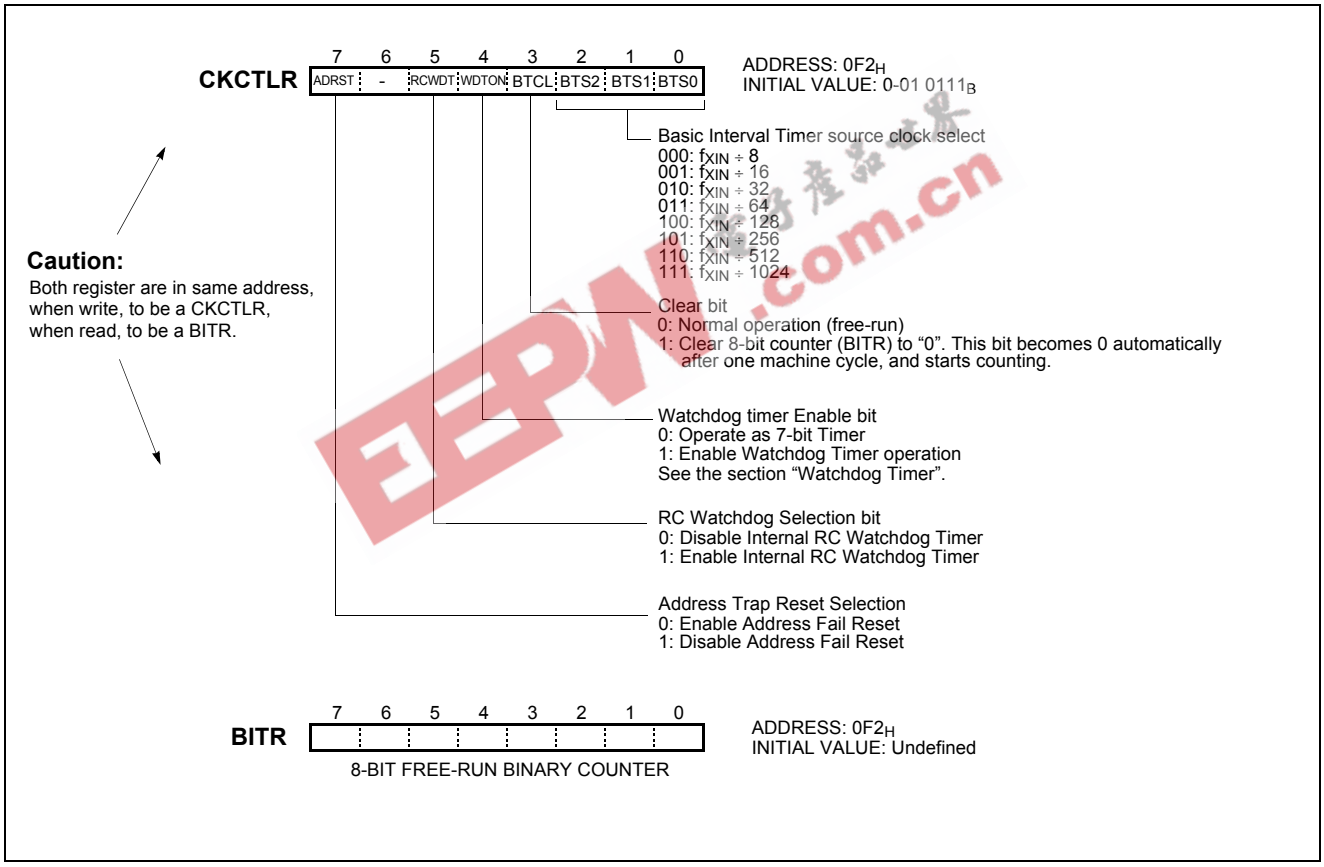


Figure 11-2 BITR: Basic Interval Timer Mode Register

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_{IN} pin. It means that the watchdog timer will run, even if the clock on the X_{IN} 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           ; RC-OSC WDT running
  :
```

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

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

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

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.

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

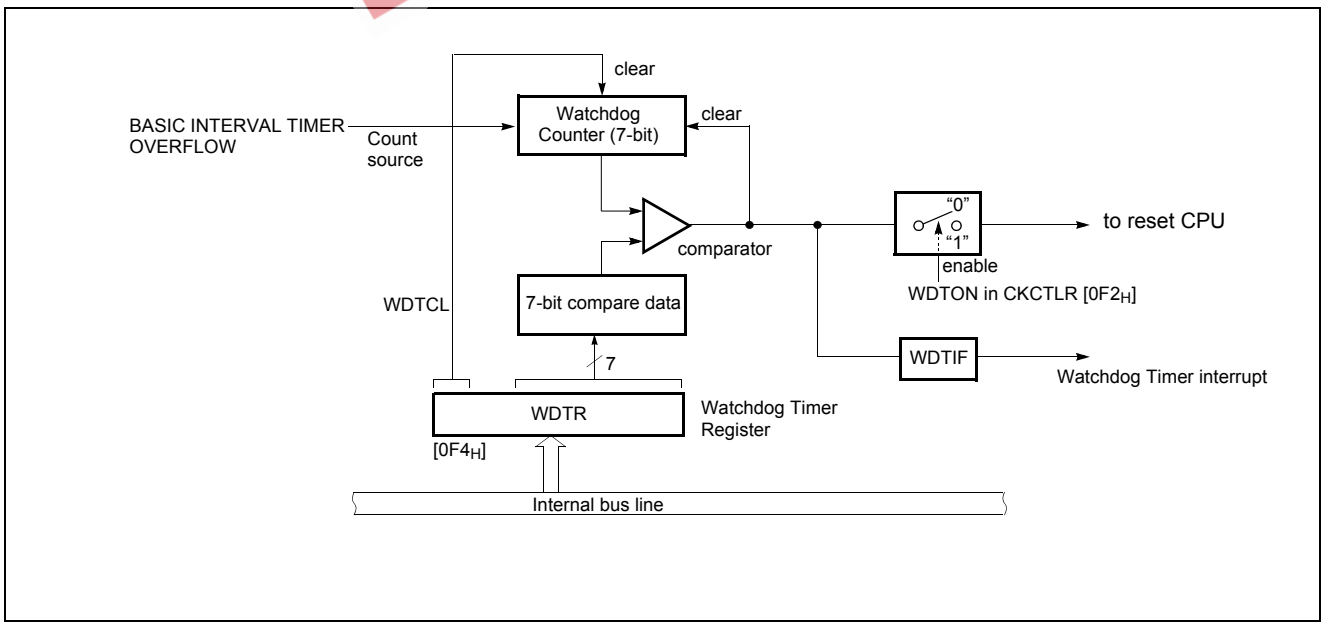


Figure 12-1 Block Diagram of Watchdog Timer

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 $\overline{\text{RESET}}$ 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).

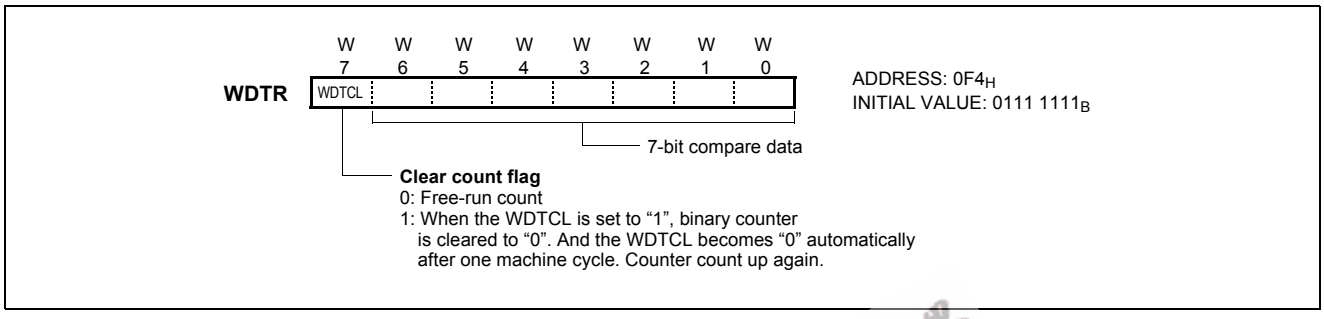


Figure 12-2 WDTR: Watchdog Timer Control Register

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
```

Within WDT detection time

Within WDT detection time

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    ;WDTCL ←1
```

:

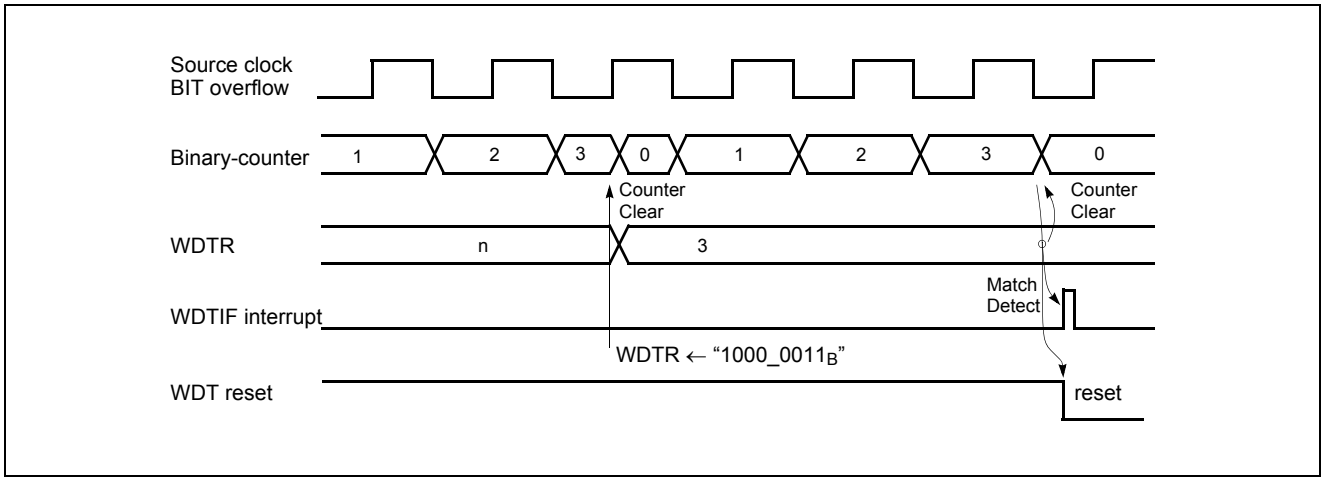


Figure 12-3 Watchdog timer Timing

If the watchdog timer output becomes active, a reset is generated, which drives the $\overline{\text{RESET}}$ 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

TheMC80F0104/0204 has Four Timer/Counter registers. 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. Timer 4 is 16-bit Timer/Counter.

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 corresponding external input pin, EC0 or EC1.

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 .

Timer 2 and Timer 3 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 TM2 and TM3 as shown in Table 13-2, Figure 13-2 .

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.

16BIT	CAP2	CAP3	PWM3E	T2CK [2:0]	T3CK [1:0]	PWM3O	TIMER 2	TIMER 3
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 13-2 Operating Modes of Timer 2, 3

TM0

R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0
-	-	CAP0	T0CK2	T0CK1	T0CK0	T0CN	T0ST

ADDRESS: $0D0_H$
INITIAL VALUE: --00 0000_B

Bit Name	Bit Position	Description
CAP0	TM0.5	0: Timer/Counter mode 1: Capture mode selection flag
T0CK2 T0CK1 T0CK0	TM0.4 TM0.3 TM0.2	000: 8-bit Timer, Clock source is $f_{XIN} \div 2$ 001: 8-bit Timer, Clock source is $f_{XIN} \div 4$ 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.

TM1

R/W 7	R/W 6	R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0
POL	16BIT	PWM1E	CAP1	T1CK1	T1CK0	T1CN	T1ST

ADDRESS: $0D2_H$
INITIAL VALUE: 00_H

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 T1CK0	TM1.3 TM1.2	00: 8-bit Timer, Clock source is f_{XIN} 01: 8-bit Timer, Clock source is $f_{XIN} \div 4$ 10: 8-bit Timer, Clock source is $f_{XIN} \div 16$ 11: 8-bit Timer, Clock source is Using the Timer 2 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.

TDR0

R/W 7	R/W 6	R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0

ADDRESS: $0D1_H$
INITIAL VALUE: $0FF_H$

TDR1

R/W 7	R/W 6	R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0

ADDRESS: $0D3_H$
INITIAL VALUE: $0FF_H$

Read: Count value read
Write: Compare data write

Figure 13-1 TM0, TM1 Registers

TM2

		R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0
-	-	CAP2	T2CK2	T2CK1	T2CK0	T2CN	T2ST

ADDRESS: $0D6_H$
INITIAL VALUE: --00 0000_B

Bit Name	Bit Position	Description
CAP2	TM2.5	0: Timer/Counter mode 1: Capture mode selection flag
T2CK2 T2CK1 T2CK0	TM2.4 TM2.3 TM2.2	000: 8-bit Timer, Clock source is $f_{XIN} \div 2$ 001: 8-bit Timer, Clock source is $f_{XIN} \div 4$ 010: 8-bit Timer, Clock source is $f_{XIN} \div 8$ 011: 8-bit Timer, Clock source is $f_{XIN} \div 16$ 100: 8-bit Timer, Clock source is $f_{XIN} \div 64$ 101: 8-bit Timer, Clock source is $f_{XIN} \div 256$ 110: 8-bit Timer, Clock source is $f_{XIN} \div 1024$ 111: EC1 (External clock)
T2CN	TM2.1	0: Timer count pause 1: Timer count start
T2ST	TM2.0	0: When cleared, stop the counting. 1: When set, Timer 0 Count Register is cleared and start again.

TM3

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

ADDRESS: $0D8_H$
INITIAL VALUE: 00_H

Bit Name	Bit Position	Description
POL	TM3.7	0: PWM Duty Active Low 1: PWM Duty Active High
16BIT	TM3.6	0: 8-bit Mode 1: 16-bit Mode
PWM3E	TM3.5	0: Disable PWM 1: Enable PWM
CAP3	TM3.4	0: Timer/Counter mode 1: Capture mode selection flag
T3CK1 T3CK0	TM3.3 TM3.2	00: 8-bit Timer, Clock source is f_{XIN} 01: 8-bit Timer, Clock source is $f_{XIN} \div 4$ 10: 8-bit Timer, Clock source is $f_{XIN} \div 16$ 11: 8-bit Timer, Clock source is Using the Timer 2 Clock
T3CN	TM3.1	0: Timer count pause 1: Timer count start
T3ST	TM3.0	0: When cleared, stop the counting. 1: When set, Timer 0 Count Register is cleared and start again.

TDR2

R/W 7	R/W 6	R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0

ADDRESS: $0D7_H$
INITIAL VALUE: $0FF_H$

TDR3

R/W 7	R/W 6	R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0

ADDRESS: $0D9_H$
INITIAL VALUE: $0FF_H$

Read: Count value read
Write: Compare data write

Figure 13-2 TM2, TM3 Registers

13.1 8-bit Timer / Counter Mode

The MC80F0104/0204 has four 8-bit Timer/Counters, Timer 0, Timer 1, Timer 2, Timer 3. The Timer 0, Timer 1 are shown in Figure 13-3 and Timer 2, Timer 3 are shown in Figure 13-4 .

The “timer” or “counter” function is selected by control registers TM0, TM1, TM2, TM3 as shown in Figure 13-1 . To use as an 8-bit timer/counter mode, bit CAP0, CAP1, CAP2, or CAP3 of TMx should be cleared to “0” and 16BIT and PWM1E or PWM3E of TM1 or TM3 should be cleared to "0" (Figure 13-3). These timers have each 8-bit count register and data register. The count register is increased by every internal or external clock input. The internal clock has a prescaler divide ratio option of 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).

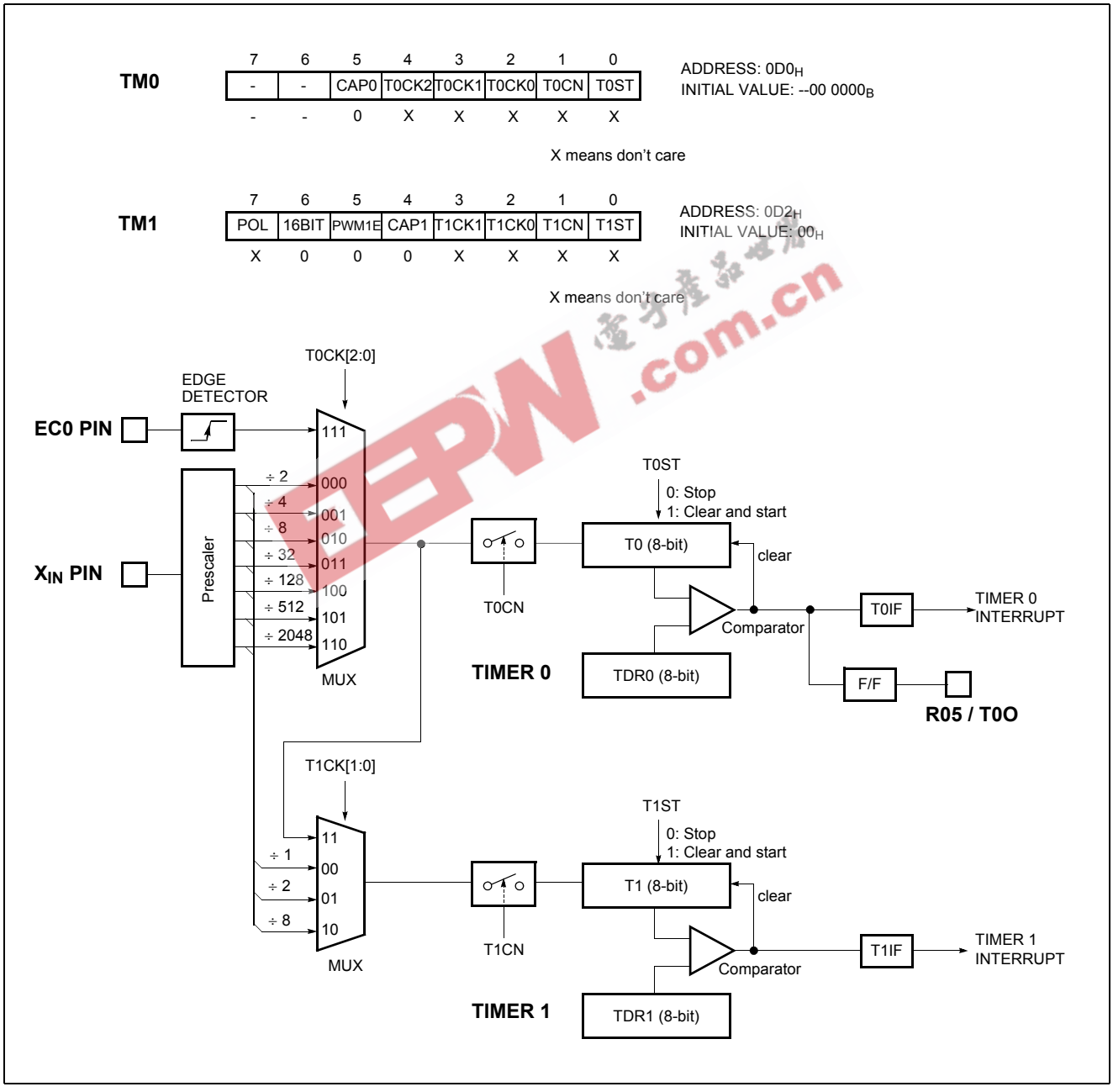


Figure 13-3 8-bit Timer/Counter 0, 1

TM2

7	6	5	4	3	2	1	0
-	-	CAP2	T2CK2	T2CK1	T2CK0	T2CN	T2ST
-	-	0	X	X	X	X	X

ADDRESS: $0D6_H$
INITIAL VALUE: $--000000_B$

X means don't care

TM3

7	6	5	4	3	2	1	0
POL	16BIT	PWM3E	CAP3	T3CK1	T3CK0	T3CN	T3ST
X	0	0	0	X	X	X	X

ADDRESS: $0D8_H$
INITIAL VALUE: 00_H

X means don't care

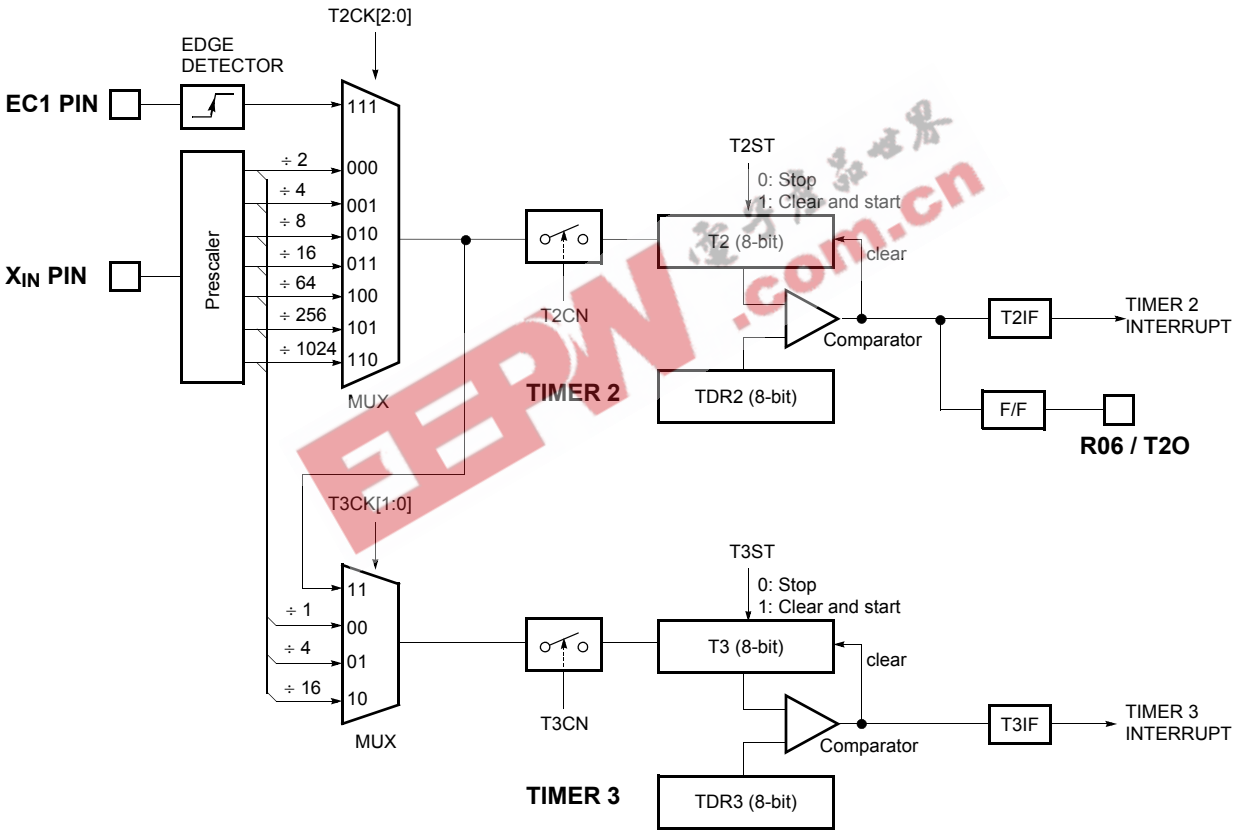


Figure 13-4 8-bit Timer/Counter 2, 3

Example 1:

Timer0 = 2ms 8-bit timer mode at 4MHz
Timer1 = 0.5ms 8-bit timer mode at 4MHz
Timer2 = 1ms 8-bit timer mode at 4MHz
Timer3 = 1ms 8-bit timer mode at 4MHz

```
LDM    TDR0,#249
LDM    TDR1,#249
LDM    TDR2,#249
LDM    TDR3,#249
LDM    TM0,#0000_1111B
LDM    TM1,#0000_1011B
LDM    TM2,#0000_1111B
LDM    TM3,#0000_1011B
SET1   T0E
SET1   T1E
SET1   T2E
SET1   T3E
EI
```

Example 2:

Timer0 = 8-bit event counter mode
Timer1 = 0.5ms 8-bit timer mode at 4MHz
Timer2 = 8-bit event counter mode
Timer3 = 1ms 8-bit timer mode at 4MHz

```
LDM    TDR0,#249
LDM    TDR1,#249
LDM    TDR2,#249
LDM    TDR3,#249
LDM    TM0,#0001_1111B
LDM    TM1,#0000_1011B
LDM    TM2,#0001_1111B
LDM    TM3,#0000_1011B
SET1   T0E
SET1   T1E
SET1   T2E
SET1   T3E
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, or 2, 4, 8, 16, 64, 256, 1024 selected by control bits T2CK[2:0] of register TM2, or 1, 4, 16 selected by control bits T3CK[1:0] of register TM3. In the Timer 0, timer register T0 increases from 00_H until it matches TDR0 and then reset to 00_H. 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. Likewise, In order to use Timer2 as counter function, the bit EC1 of the Port Selection Register (PSR0.5) is set to "1". The Timer 2 can be used as a counter by pin EC1 input, but Timer 3 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*n* are compared with the contents of up-counter, T*n*. If match is found, a timer *n* interrupt (T*n*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*n* is changeable by software, time interval is set as you want.

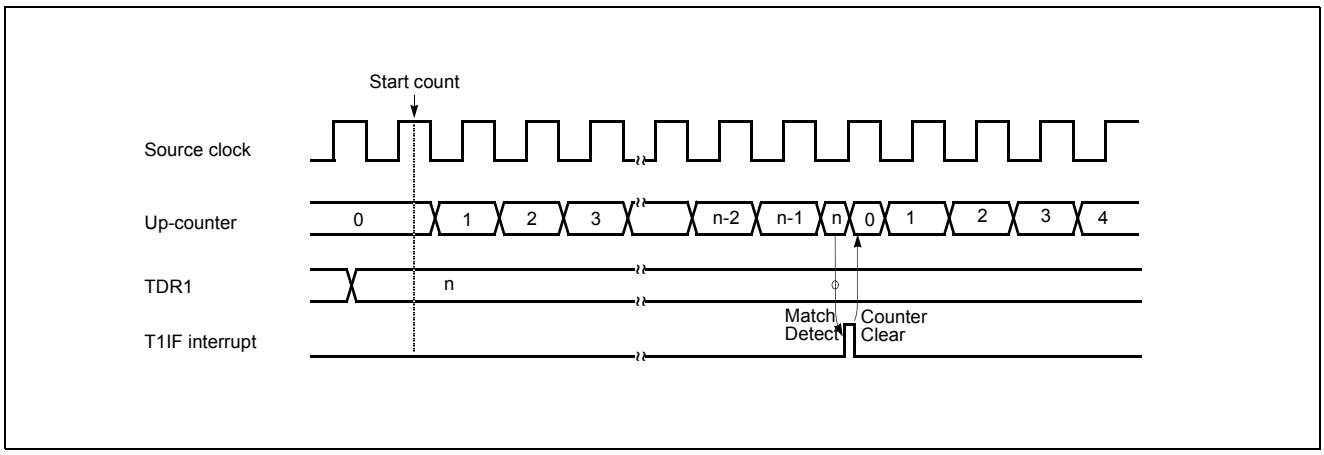


Figure 13-5 Timer Mode Timing Chart

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 TM0 = 0000 1111_B (8-bit Timer mode, Prescaler divide ratio = 32)
TDR0 = $124_D = 7C_H$
f_{XIN} = 4 MHz

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

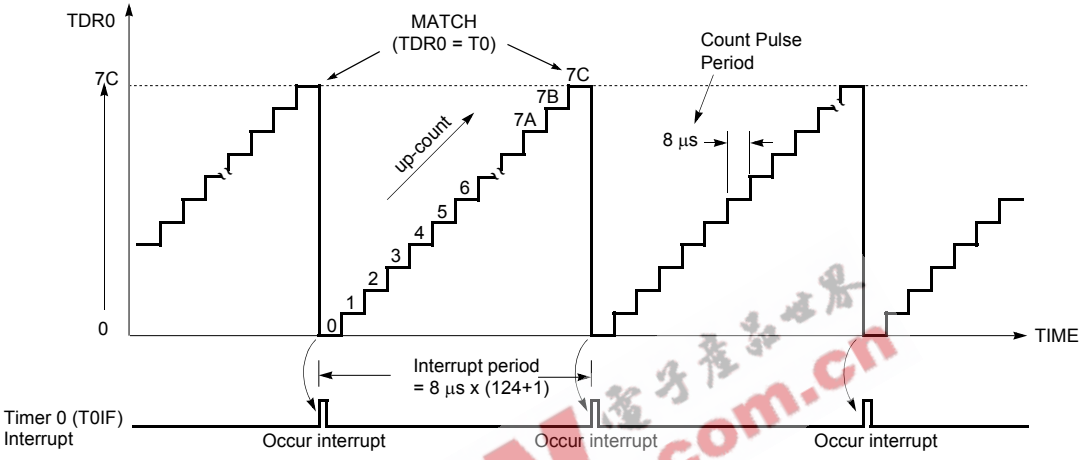


Figure 13-6 Timer Count Example

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 or EC1 pin input. Source clock is used as an internal clock selected with timer mode register TM0 or TM2. The contents of timer data register TDR*n* (n = 0,1,2,3) are compared with the contents of the up-counter T*n*. If a match is found, an timer interrupt request flag T*n*IF is generated, and the counter is cleared to "0". The counter is restart and count up continuously by every falling edge of the EC0 or EC1 pin input. The maximum frequency applied to the EC0 or EC1 pin is $f_{XIN}/2$ [Hz].

In order to use event counter function, the bit 4, 5 of the Port Selection Register PSR0(address $0F8_H$) is required to be set to "1".

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

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

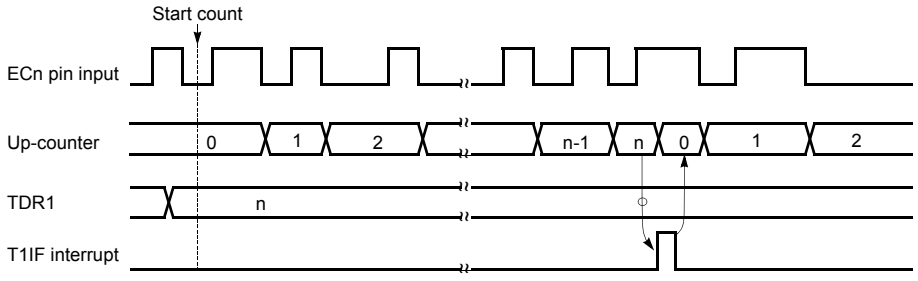


Figure 13-7 Event Counter Mode Timing Chart

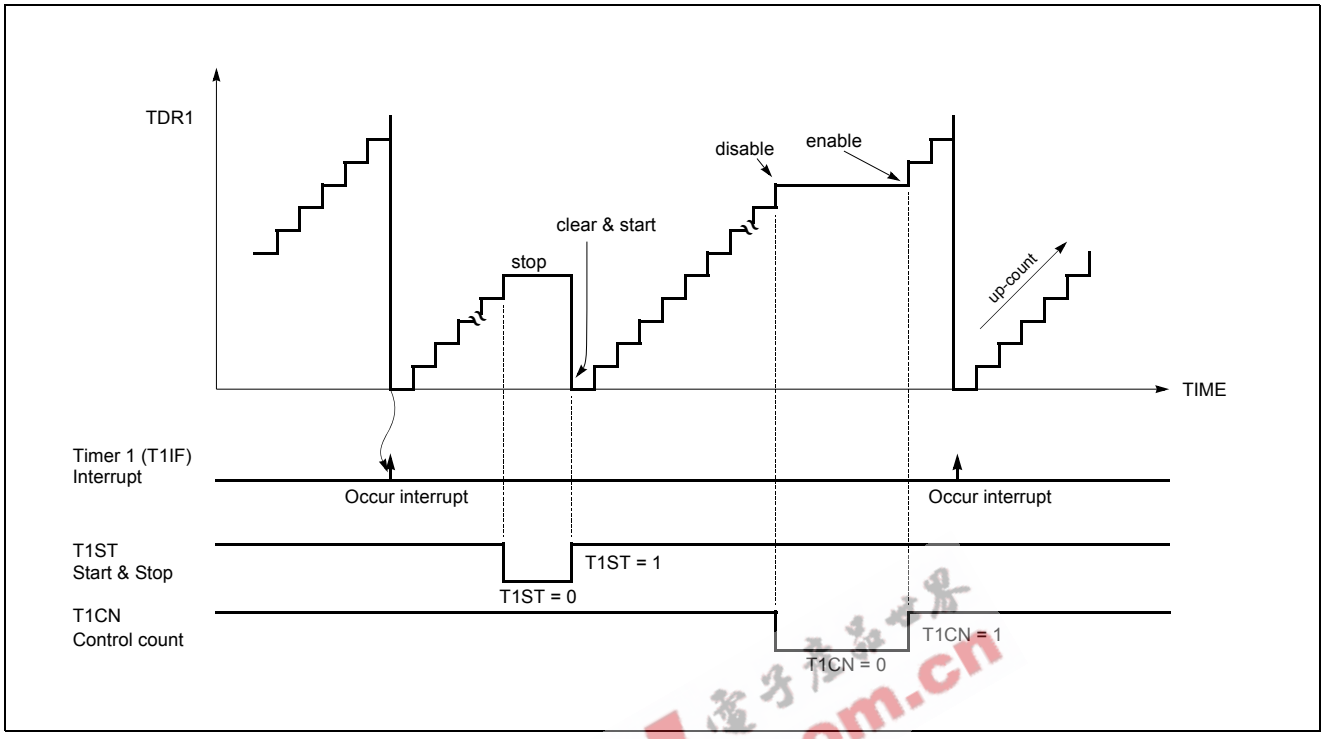


Figure 13-8 Count Operation of Timer / Event counter

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 are incremented from 0000_H until it matches TDR0, TDR1 and then resets to 0000_H. 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 13-9 .

Likewise, A 16-bit timer/counter register T2, T3 are incremented from 0000_H until it matches TDR2, TDR3 and then resets to 0000_H. The match output generates Timer 2 interrupt.

The clock source of the Timer 2 is selected either internal or external clock by bit T2CK[2:0]. In 16-bit mode, the bits T3CK[1:0] and 16BIT of TM3 should be set to "1" respectively as shown in Figure 13-10 .

Even if the Timer 0 (including Timer 1) is used as a 16-bit timer, the Timer 2 and Timer 3 can still be used as either two 8-bit timer or one 16-bit timer by setting the TM3. Reversely, even if the Timer 2 (including Timer 3) is used as a 16-bit timer, the Timer 0 and Timer 1 can still be used as 8-bit timer independently.

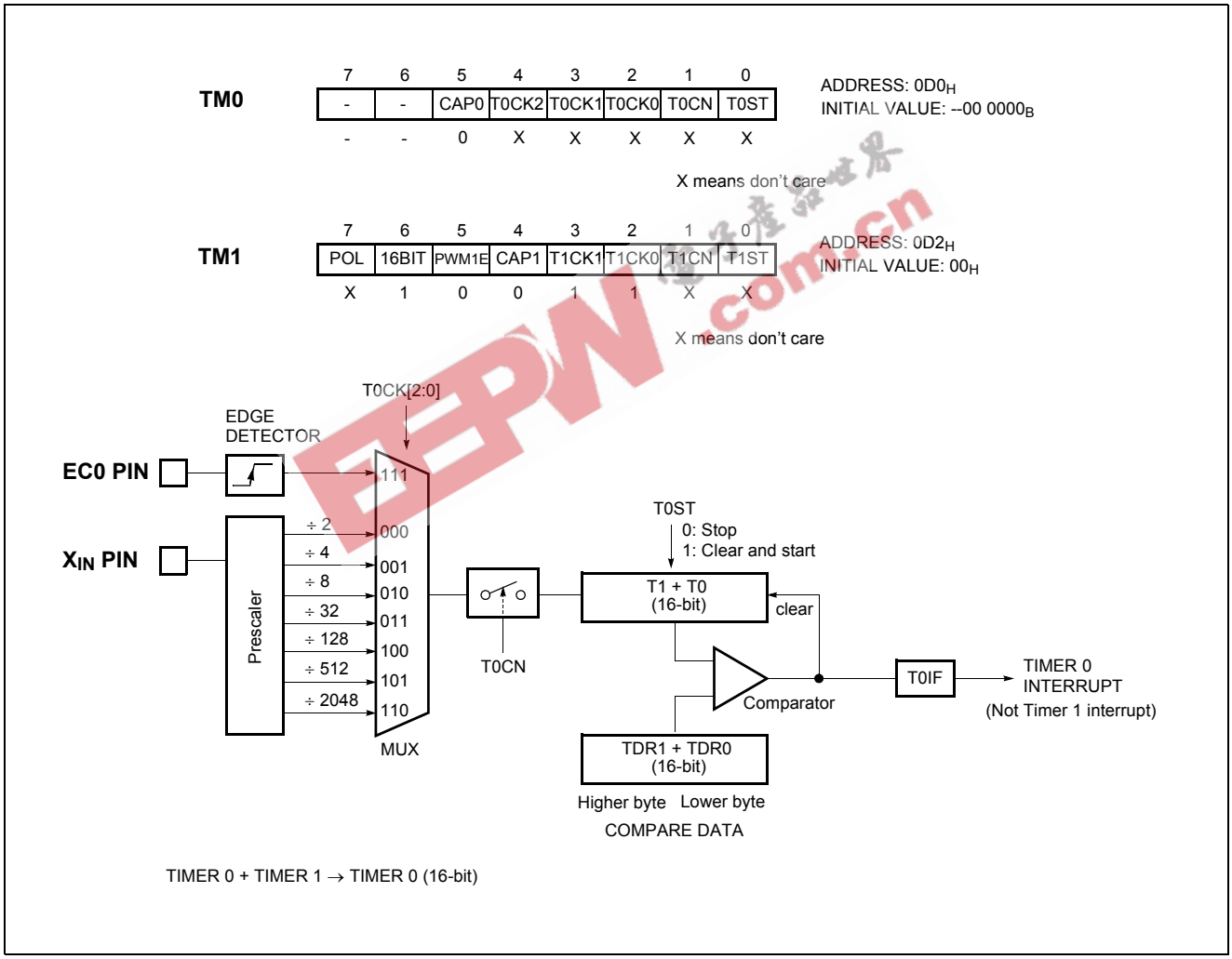


Figure 13-9 16-bit Timer/Counter for Timer 0, 1

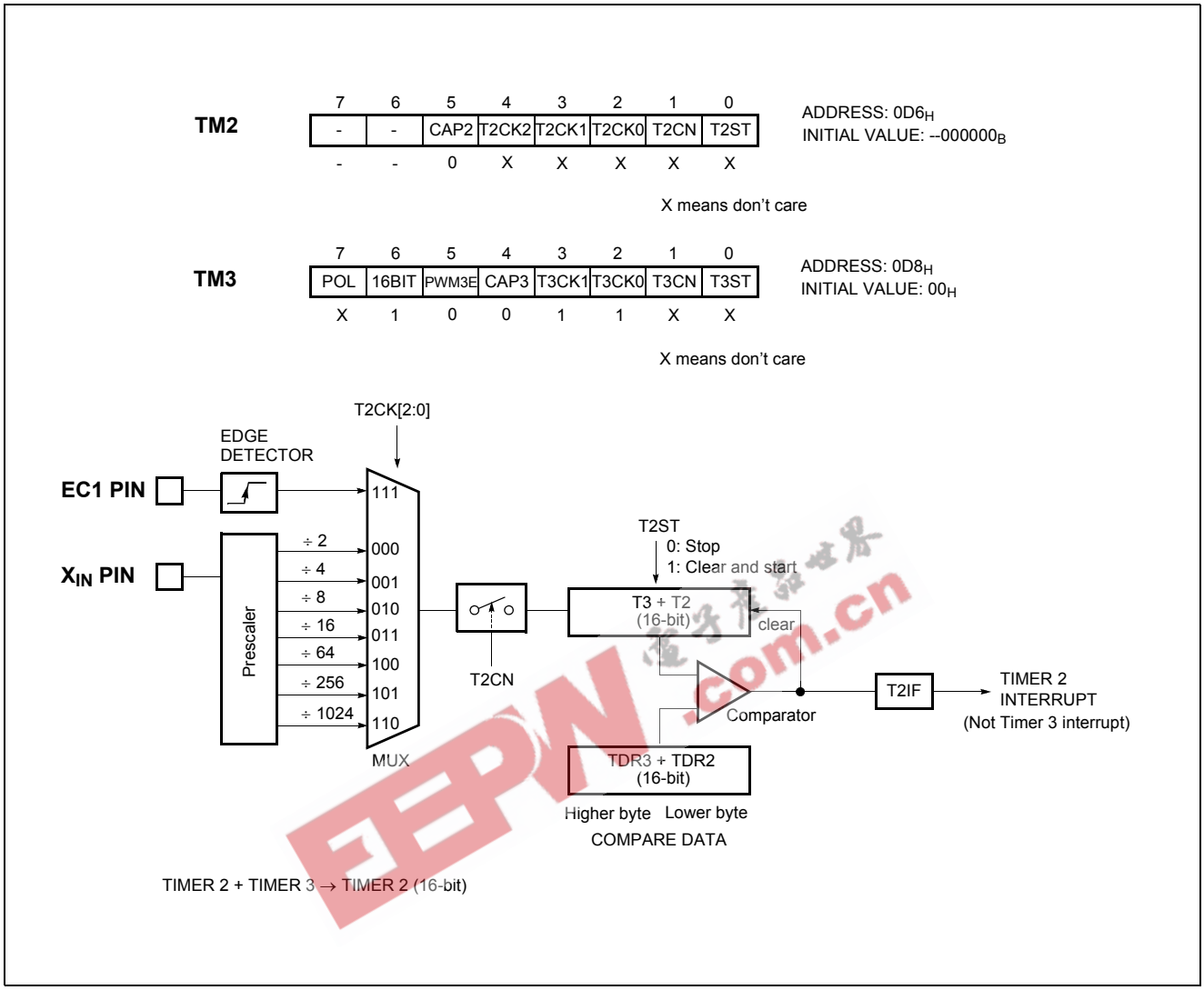


Figure 13-10 16-bit Timer/Counter for Timer 2, 3

13.3 8-bit Compare Output (16-bit)

TheMC80F0104/0204 has Timer Compare Output function. To pulse out, the timer match can goes to port pin(T0O or T2O) as shown in Figure 13-3 or Figure 13-4 . Thus, pulse out is generated by the timer match. These operation is implemented to pin, R05/AN5//T0O/TXD or R06/AN6/T2O/ACK.

In this mode, the bit T0OE or T2OE bit of Port Selection register1 (PSR1.0 or PSR1.1) 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 13-11 . Likewise, the Timer 2 capture mode is set by bit CAP2 of timer mode register TM2 (bit CAP3 of timer mode register TM3 for Timer 3) as shown in Figure 13-12 .

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, T2, T3) increases and matches TDR0 (TDR1, TDR2, TDR3).

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

For example, in Figure 13-14 , the pulse width of captured signal is wider than the timer data value (FF_H) over 2 times. When external interrupt is occurred, the captured value (13_H) is more little than wanted value. It can be obtained 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 INTx pin causes the current value in the Timer x register (T0,T1,T2,T3), to be captured into registers CDRx (CDR0, CDR1, CDR2, CDR3), 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 “18.4 External Interrupt” on page 86. In addition, the transition at INT*n* pin generate an interrupt.

Note: *The CDRn and TDRn are in same address.In the capture mode, reading operation is read the CDRn, not TDRn because path is opened to the CDRn.*

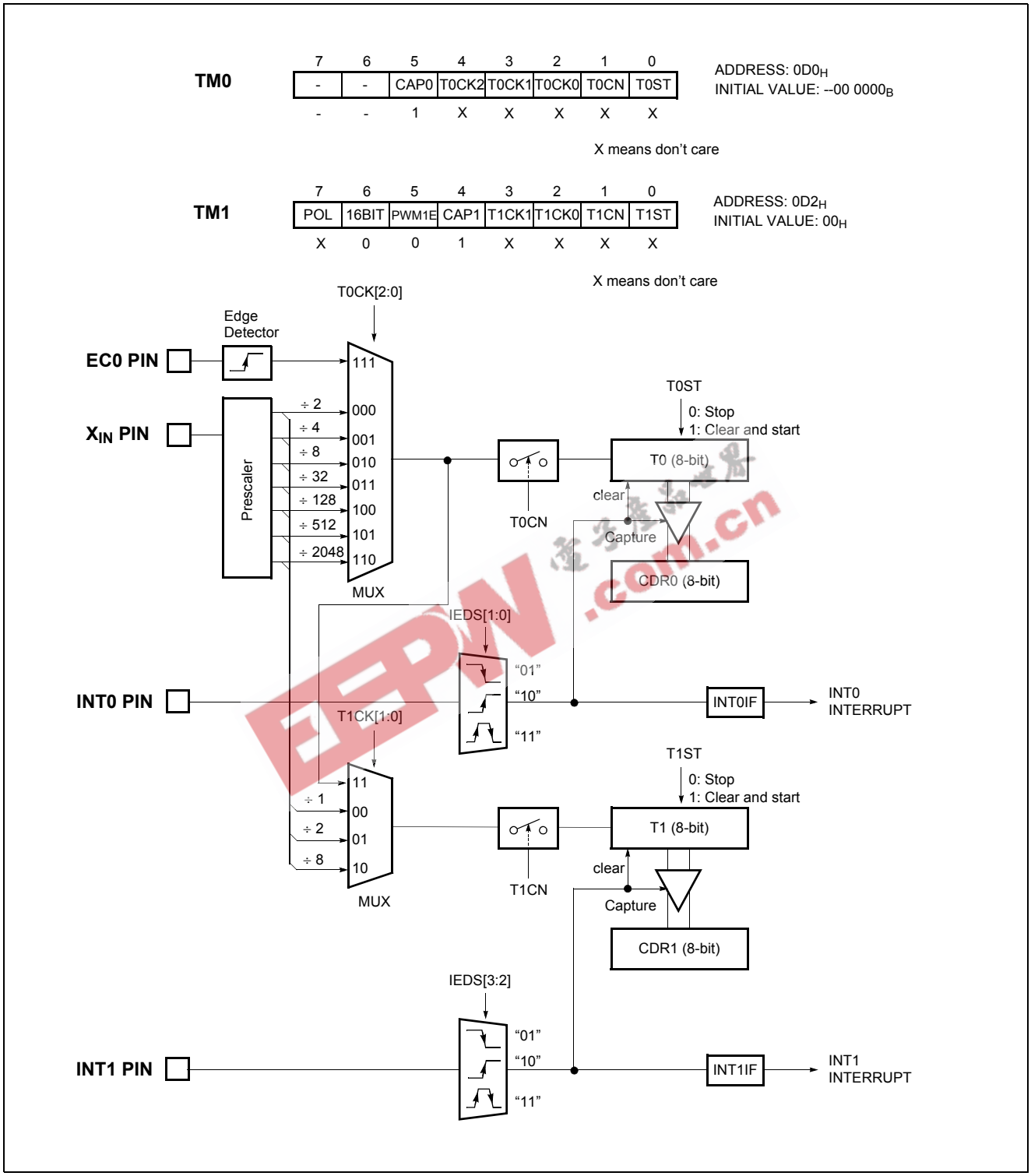


Figure 13-11 8-bit Capture Mode for Timer 0, 1

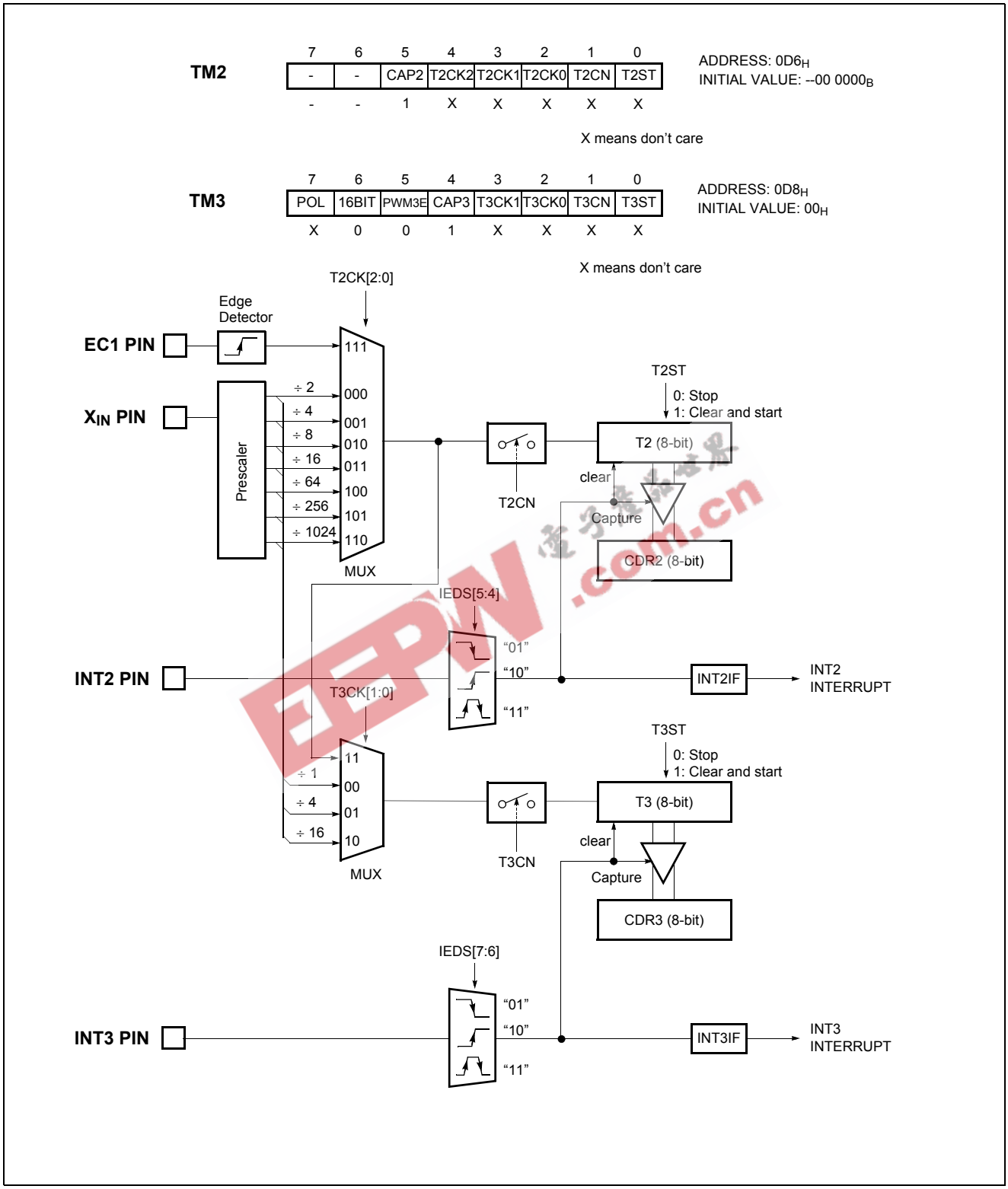


Figure 13-12 8-bit Capture Mode for Timer 2, 3

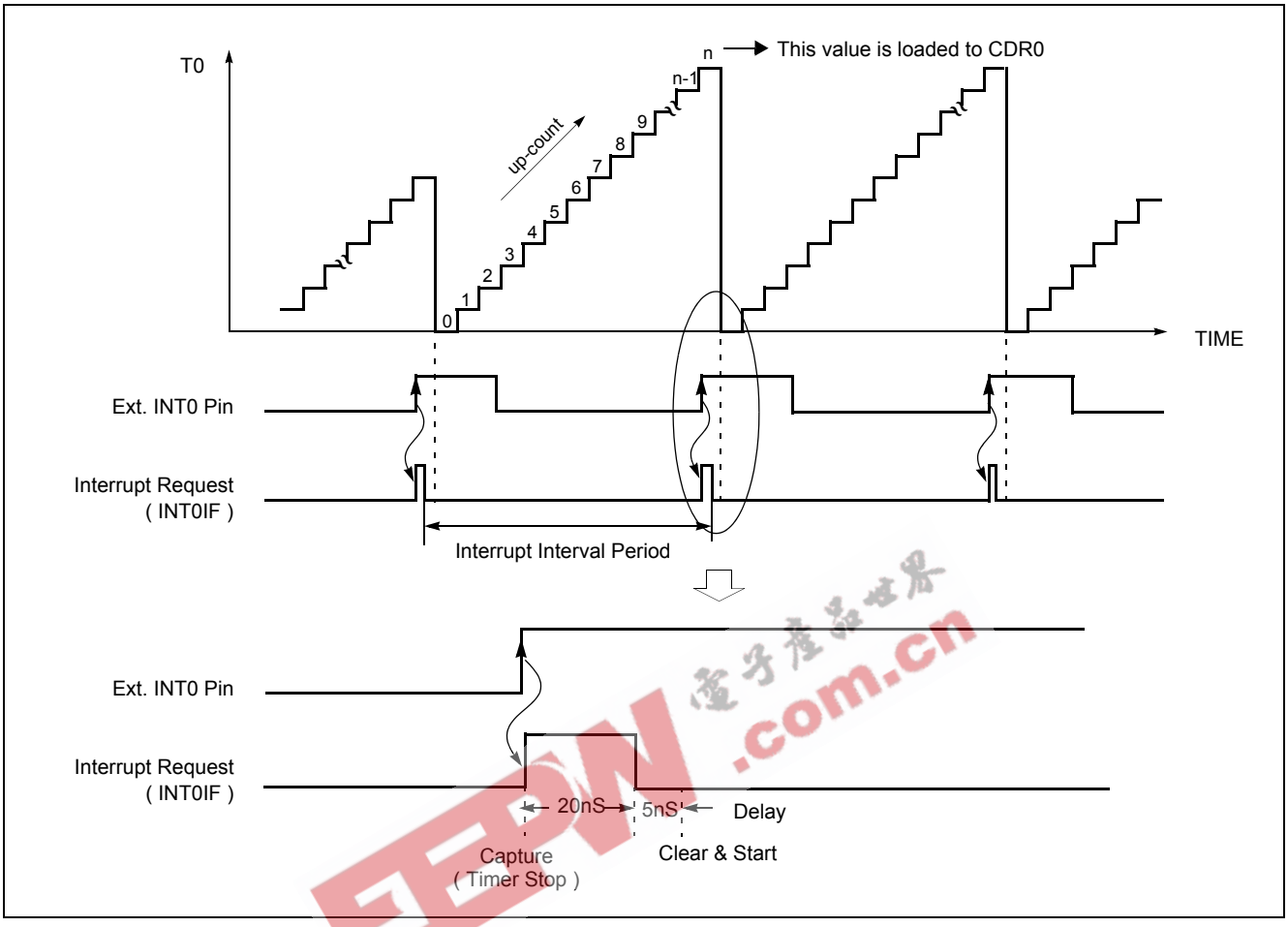


Figure 13-13 Input Capture Operation of Timer 0 Capture mode

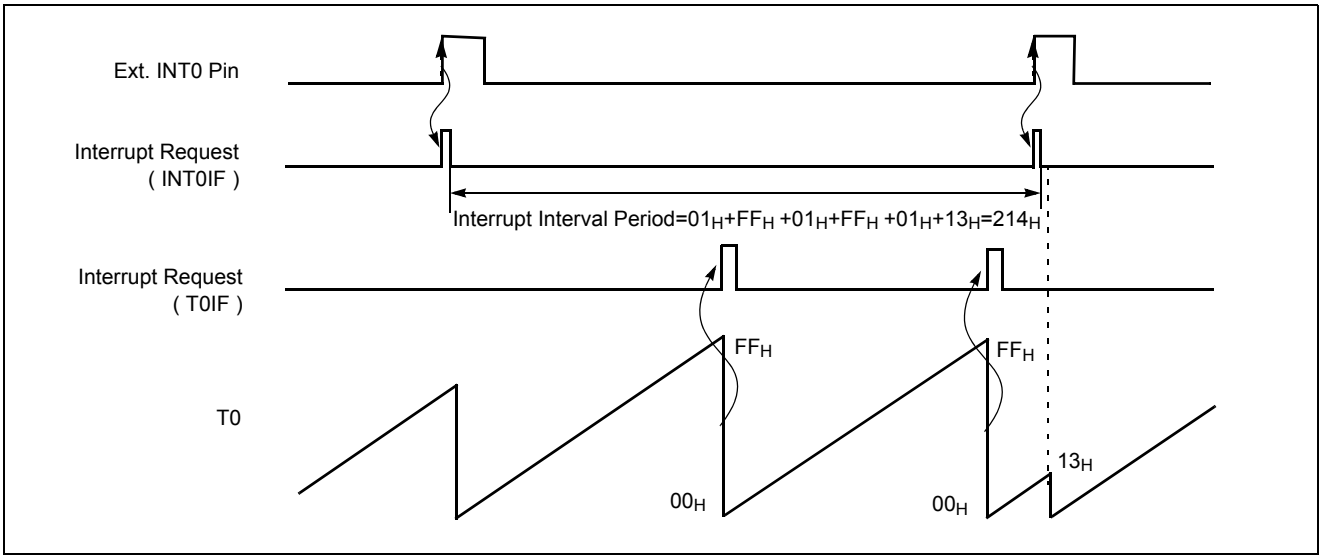


Figure 13-14 Excess Timer Overflow in Capture Mode

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 T0CK[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 13-15 .

The clock source of the Timer 2 is selected either internal or external clock by bit T2CK[2:0]. In 16-bit mode, the bits T3CK1,T3CK0, CAP3 and 16BIT of TM3 should be set to "1" respectively as shown in Figure 13-16 .

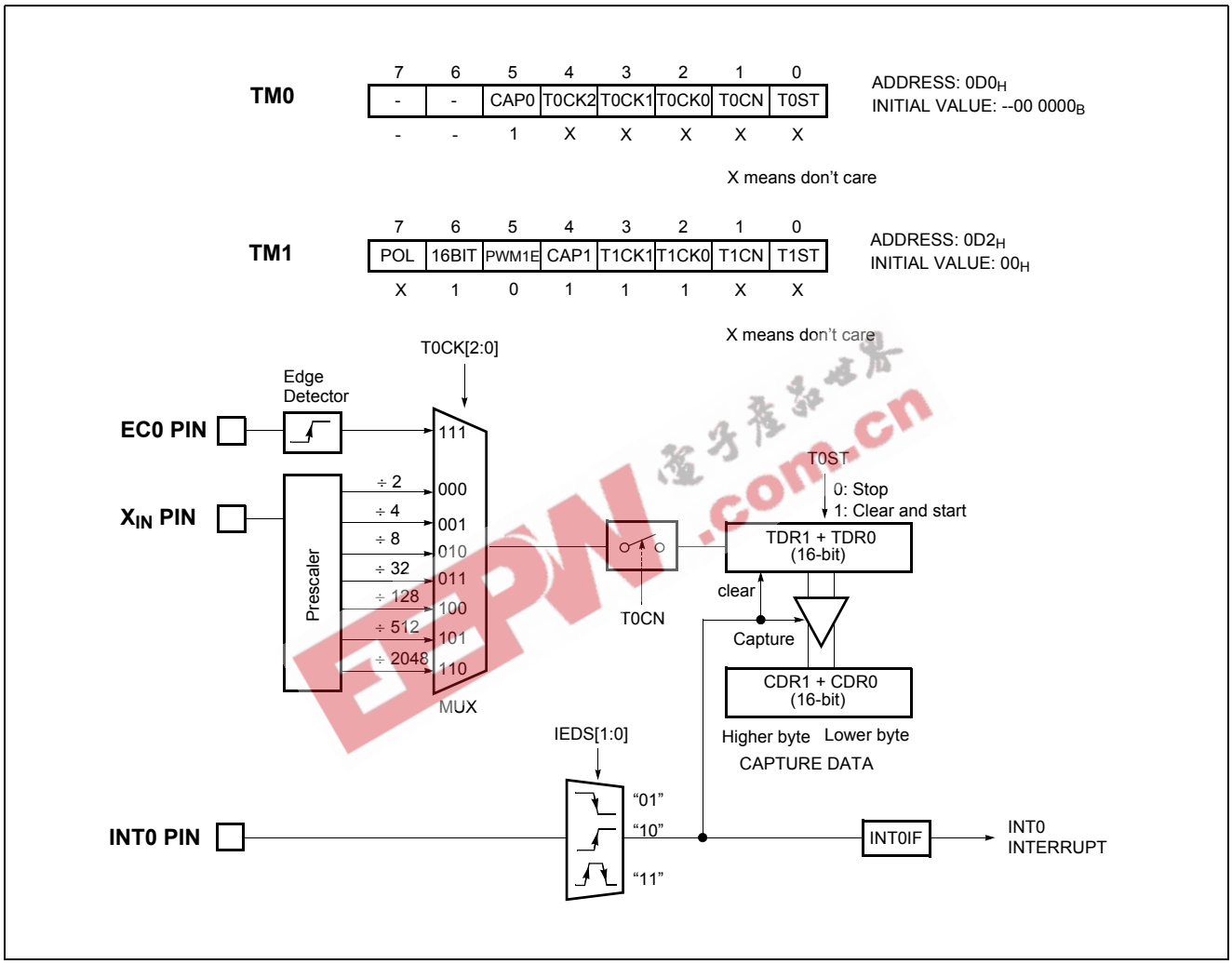


Figure 13-15 16-bit Capture Mode of Timer 0, 1

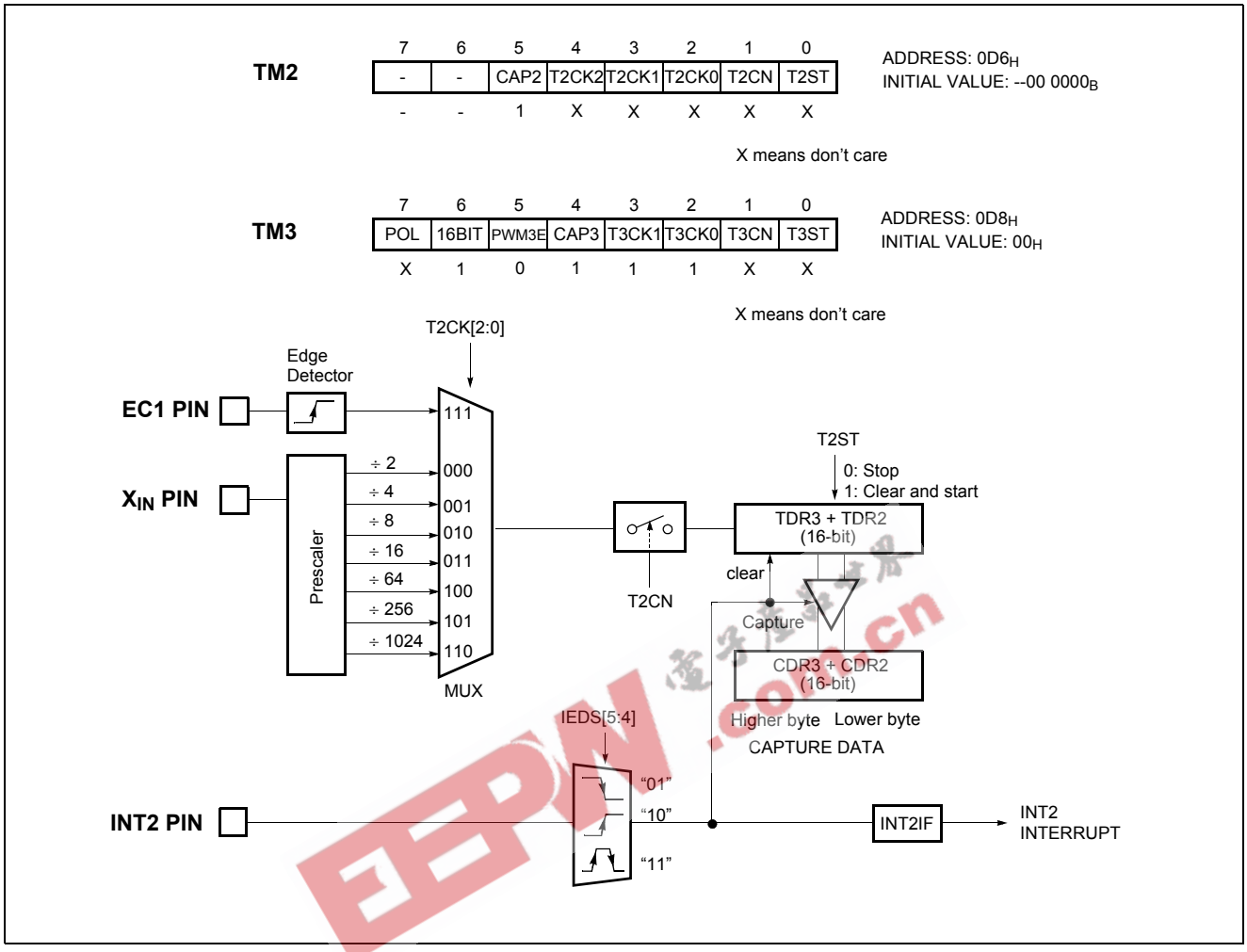


Figure 13-16 16-bit Capture Mode of Timer 2, 3

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   T0E
EI
:
:
```

Example 2:

Timer0 = 16-bit event counter mode

```
LDM    PSR0,#0001_0000B;EC0 Set
LDM    TM0,#0001_1111B;CounterMode
LDM    TM1,#0100_1100B;16bit Mode
LDM    TDR0,#<0FFH     ;
LDM    TDR1,#>0FFH     ;
SET1   T0E
EI
:
:
```

Example 3:

Timer0 = 16-bit capture mode

```
LDM    PSR0,#0000_0001B;INT0 set
LDM    TM0,#0010_1111B;CaptureMode
LDM    TM1,#0100_1100B;16bit Mode
LDM    TDR0,#<0FFH     ;
LDM    TDR1,#>0FFH     ;
LDM    IEDS,#01H;Falling Edge
SET1   T0E
EI
:
:
```

13.6 PWM Mode

TheMC80F0104/0204 has high speed PWM (Pulse Width Modulation) functions which shared with Timer1 or Timer3.

In PWM mode, R10 / PWM1O or R11 / PWM3O pin output up to a 10-bit resolution PWM output. These pins should be configured as a PWM output by setting "1" bit PWM1OE and PWM3OE 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 period of the PWM3 output is determined by the T3PPR (T3 PWM Period Register) and T3PWHR[3:2] (bit3,2 of T3 PWM High Register) and the duty of the PWM output is determined by the T3PDR (T3 PWM Duty Register) and T3PWHR[1:0] (bit1,0 of T3 PWM High Register).

The user writes the lower 8-bit period value to the T1(3)PPR(and the higher 2-bit period value to the T1(3)PWHR[3:2]. And writes duty value to the T1(3)PDR and the T1(3)PWHR[1:0] same way.

The T1(3)PDR is configured as a double buffering for glitchless PWM output. In Figure 13-18 , 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)

PWM1(3) Period = [PWM1(3)HR[3:2]T(2)3PPR] X Source Clock

PWM1(3) Duty = [PWM3HR[1:0]T3PDR] X Source Clock

The relation of frequency and resolution is in inverse proportion. Table 13-3 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 13-3 PWM Frequency vs. Resolution at 4MHz

The bit POL of TM1 or TM3 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 "00_H", 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 13-20 . 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 & PWM3E
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
```

TM1

R/W 7	R/W 6	R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0
POL	16BIT	PWM1E	CAP3	T3CK1	T3CK0	T3CN	T3ST
X	0	1	0	X	X	X	X

ADDRESS: $0D2_H$
INITIAL VALUE: 00_H

X:The value "0" or "1" corresponding your operation.

T1PWHR

- 7	- 6	- 5	- 4	W 3	W 2	W 1	W 0
-	-	-	-	T3PWHR3	T3PWHR2	T3PWHR1	T3PWHR0
-	-	-	-	X	X	X	X
				Period High		Duty High	

ADDRESS: $0D5_H$
INITIAL VALUE: ---- 0000_B
Bit Manipulation Not Available

X:The value "0" or "1" corresponding your operation.

T1PPR

W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0

ADDRESS: $0D3_H$
INITIAL VALUE: $0FF_H$

T1PDR

R/W 7	R/W 6	R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0

ADDRESS: $0D4_H$
INITIAL VALUE: 00_H

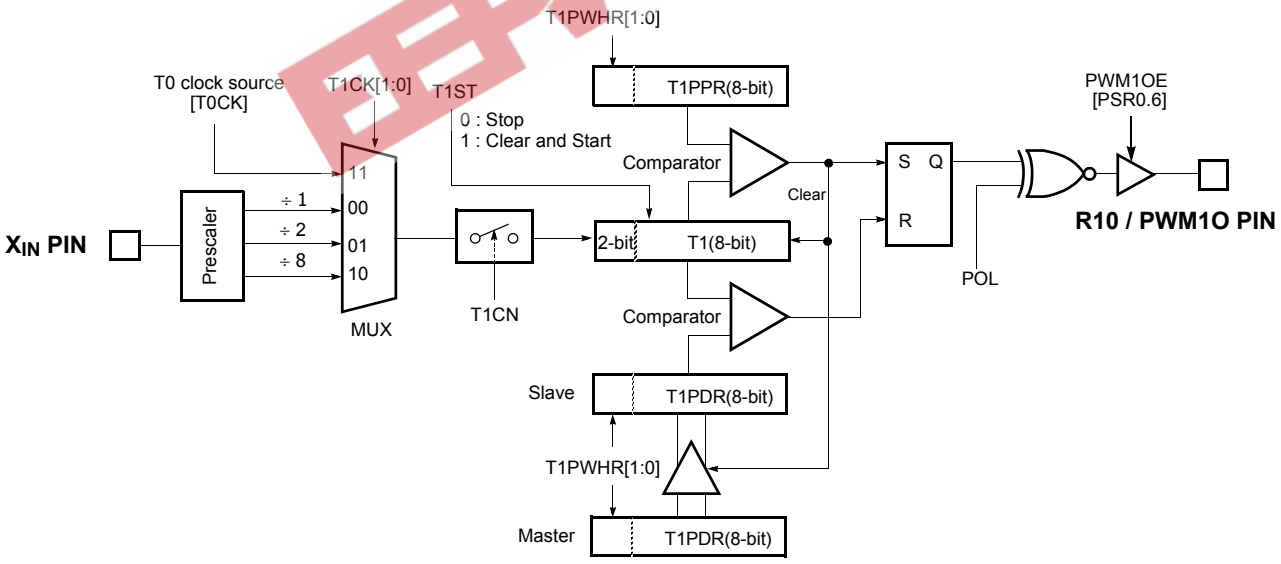


Figure 13-17 PWM1 Mode

TM3

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

ADDRESS: $0D8_H$
INITIAL VALUE: 00_H

X:The value "0" or "1" corresponding your operation.

T3PWHR

- 7	- 6	- 5	- 4	W 3	W 2	W 1	W 0
-	-	-	-	T3PWHR3	T3PWHR2	T3PWHR1	T3PWHR0
-	-	-	-	X	X	X	X
				Period High		Duty High	

ADDRESS: $0DB_H$
INITIAL VALUE: ---- 0000_B
Bit Manipulation Not Available

X:The value "0" or "1" corresponding your operation.

T3PPR

W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0

ADDRESS: $0D9_H$
INITIAL VALUE: $0FF_H$

T3PDR

R/W 7	R/W 6	R/W 5	R/W 4	R/W 3	R/W 2	R/W 1	R/W 0

ADDRESS: $0DA_H$
INITIAL VALUE: 00_H

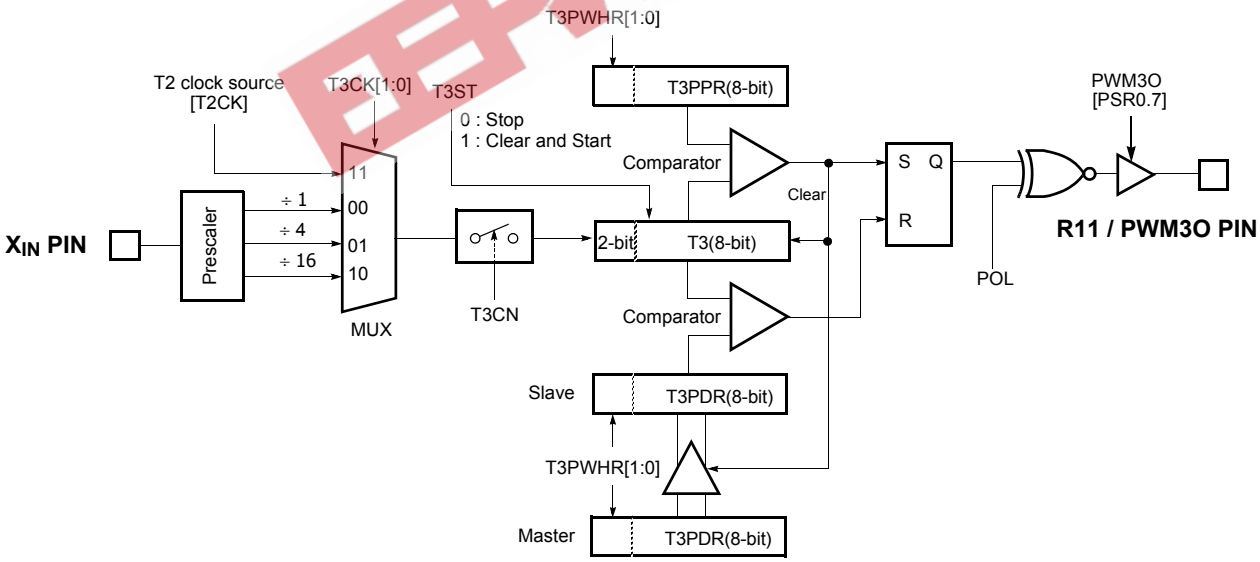


Figure 13-18 PWM3 Mode

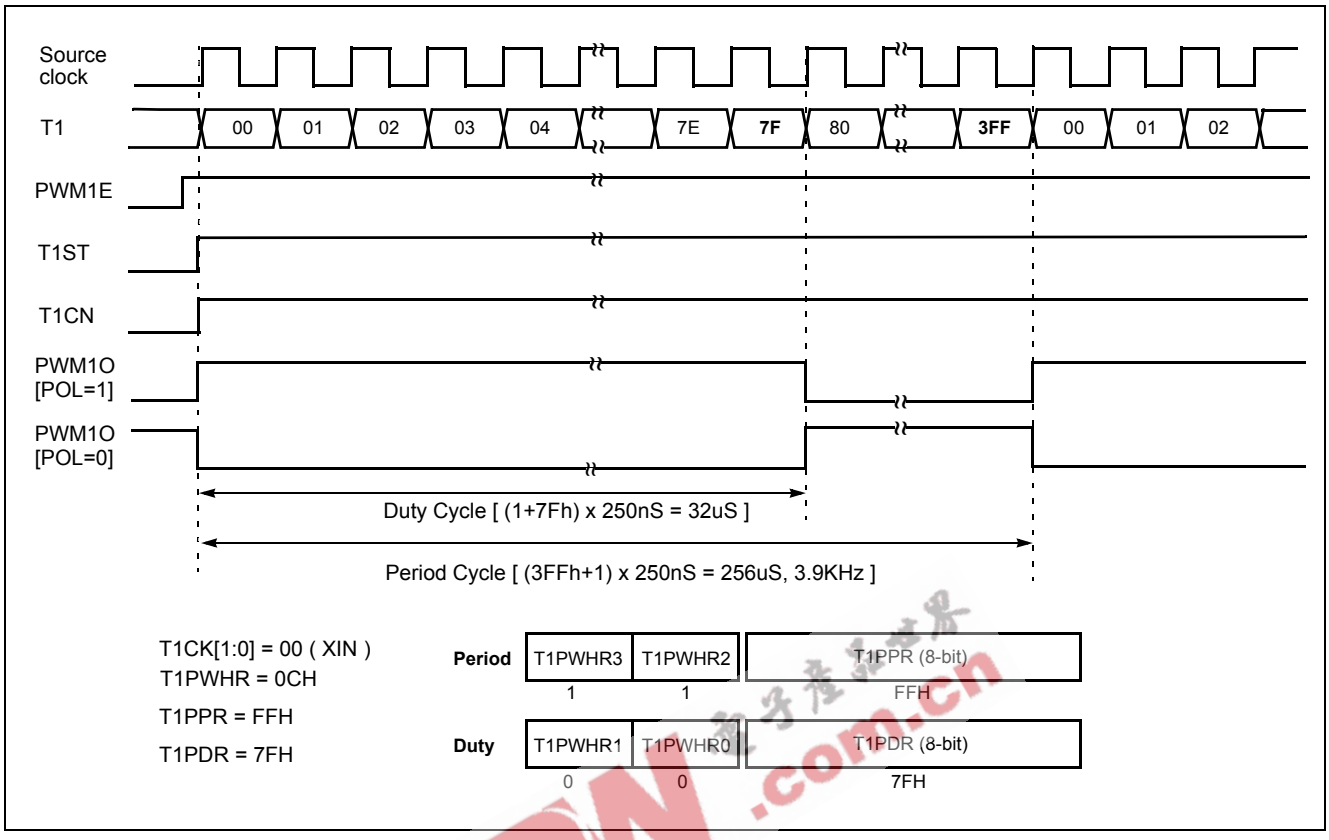


Figure 13-19 Example of PWM1 at 4MHz

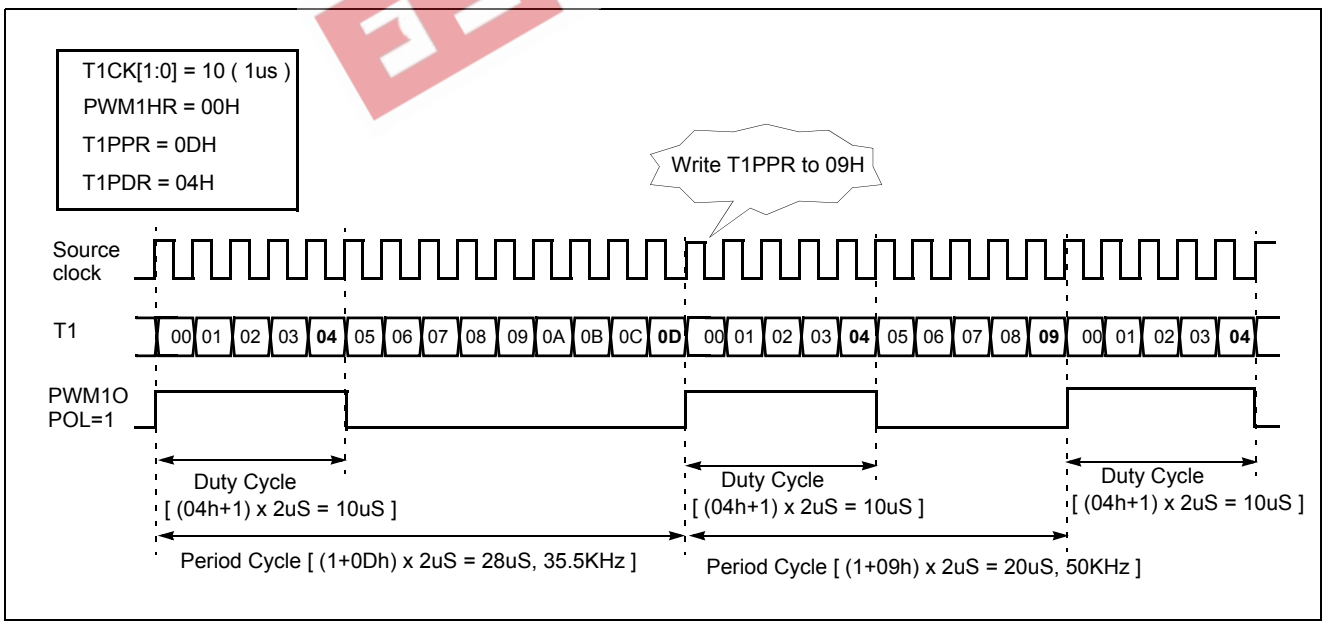


Figure 13-20 Example of Changing the PWM1 Period in Absolute Duty Cycle (@4MHz)

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 MC80F0104) 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 AVref by setting of the bit AVREFS in PSR1 register. If external analog reference AVref 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.

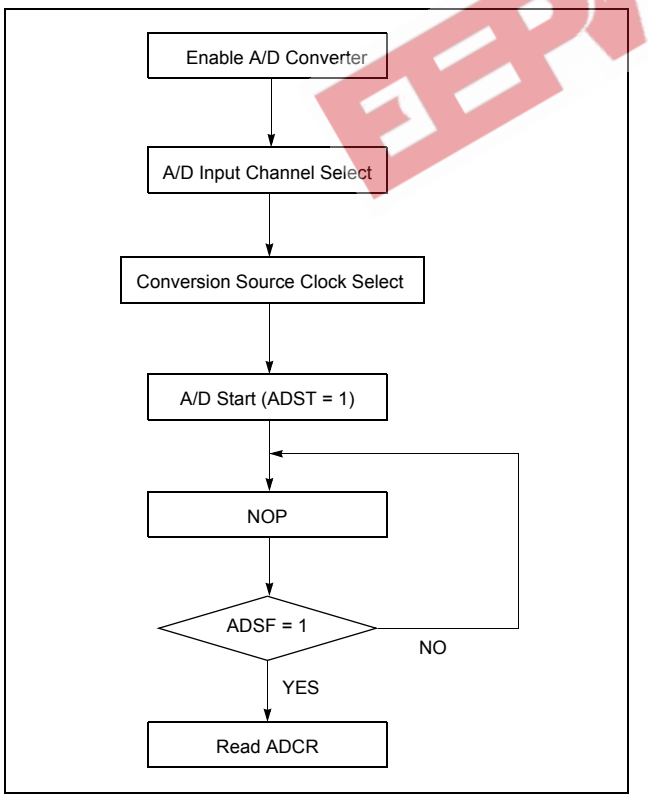


Figure 14-1 A/D Converter Operation Flow

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 25μs.

A/D Converter Cautions

(1) Input range of AN0 ~ 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 AVref) or below V_{SS} is input (even if within the absolute maximum rating range), the conversion value for that channel can not be indeterminate. 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 AVref) and analog input pins (AN0 ~ 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.

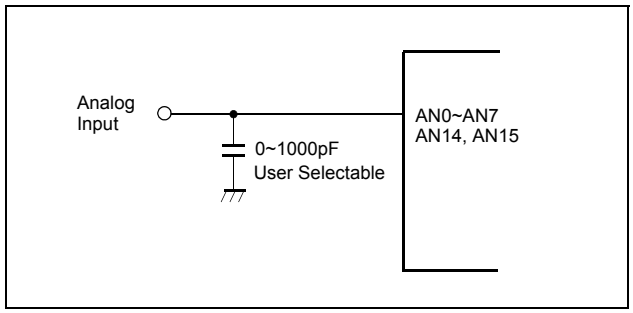


Figure 14-2 Analog Input Pin Connecting Capacitor

(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_{DD} pin input impedance

A series resistor string of approximately 5KΩ is connected between the AV_{REF} pin and the V_{SS} 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_{REF} pin and the V_{SS} pin, and there will be a large analog supply voltage error

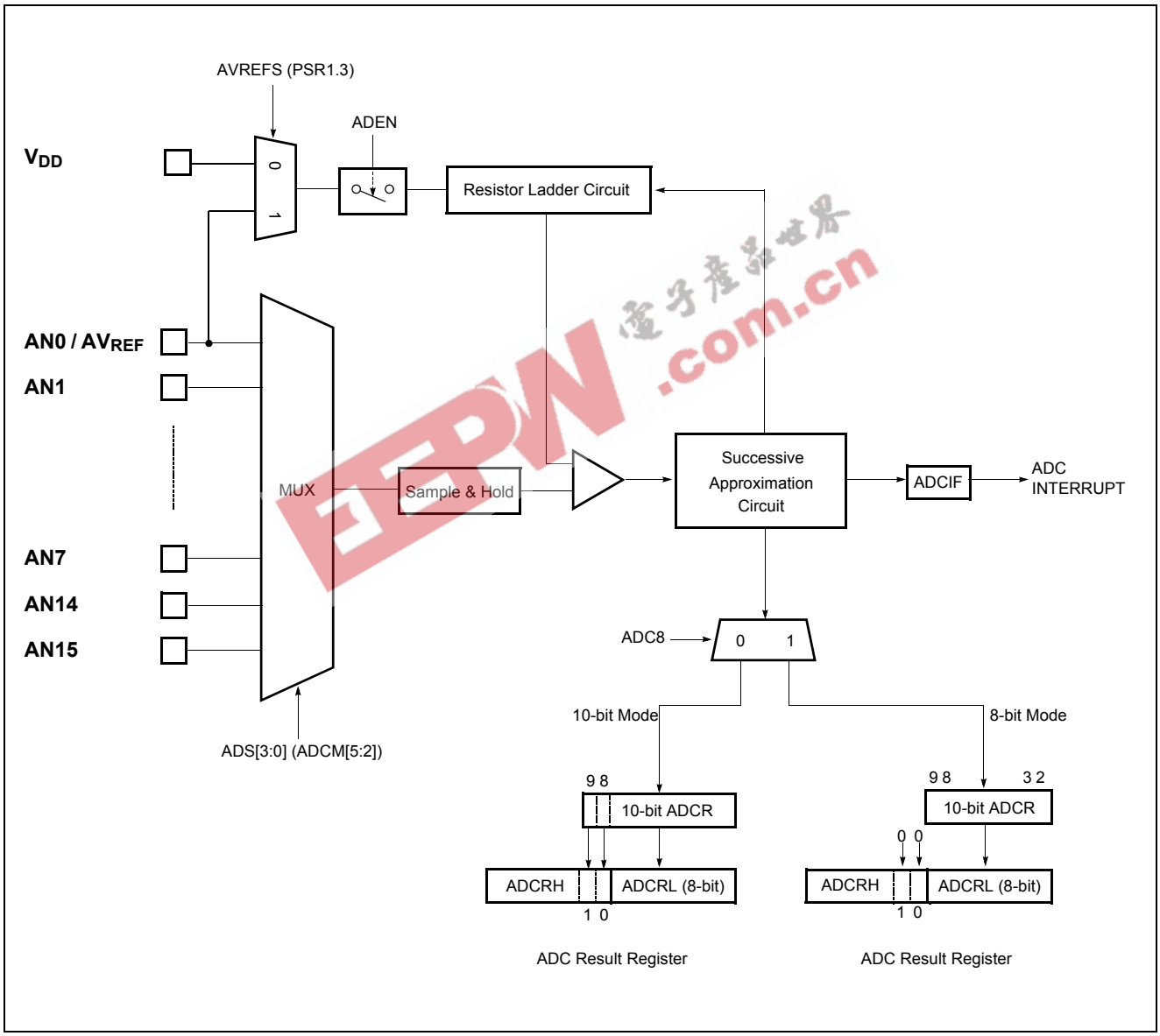
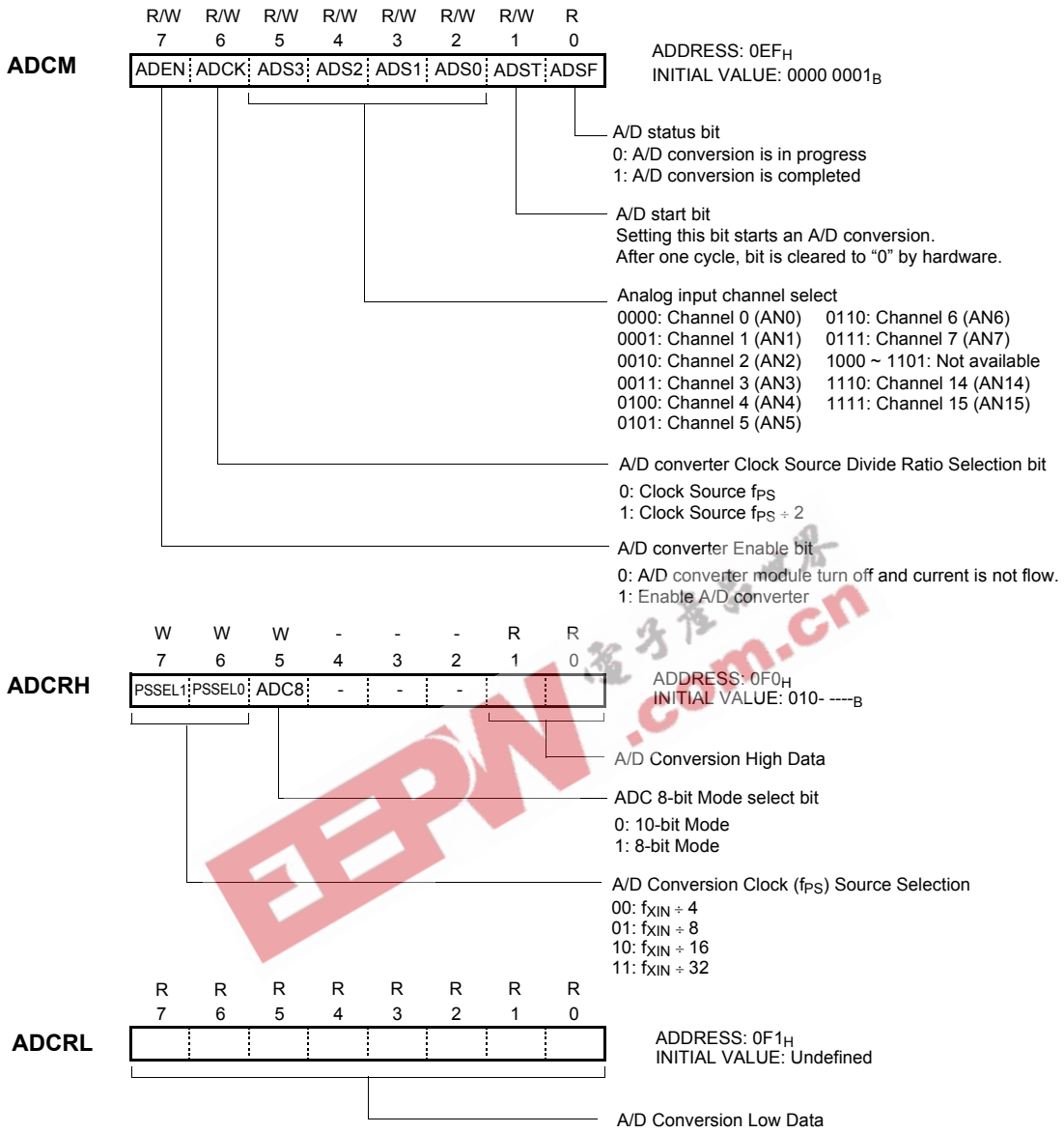


Figure 14-3 A/D Block Diagram



ADCK	PSSEL1	PSSEL0	PS Clock Selection
0	0	0	PS = $f_{XIN} \div 4$
0	0	1	PS = $f_{XIN} \div 8$
0	0	0	PS = $f_{XIN} \div 16$
0	0	1	PS = $f_{XIN} \div 32$
1	1	0	PS = $f_{XIN} \div 8$
1	1	1	PS = $f_{XIN} \div 16$
1	1	0	PS = $f_{XIN} \div 32$
1	1	1	PS = $f_{XIN} \div 64$

PS : Conversion Clock

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

15. SERIAL INPUT/OUTPUT (SIO)

The serial Input/Output is used to transmit/receive 8-bit data serially. The Serial Input/Output (SIO) module is a serial interface useful for communicating with other peripheral of microcontroller devices. These peripheral devices may be serial EEPROMs, shift registers, display drivers, A/D converters, etc. This SIO is 8-bit clock synchronous type and consists of serial I/O data register, serial I/O mode register, clock selection circuit, octal counter and control circuit as illustrated in Figure 15-1 . The SO pin is designed to input and output. So the Serial I/O(SIO) can be operated with minimum two pin. Pin R00/SCK, R01/SI, and R02/SO pins are controlled by the Serial Mode Register. The contents of the Serial I/O data register can be written into or read out by software. The data in the Serial Data Register can be shifted synchronously with the transfer clock signal.

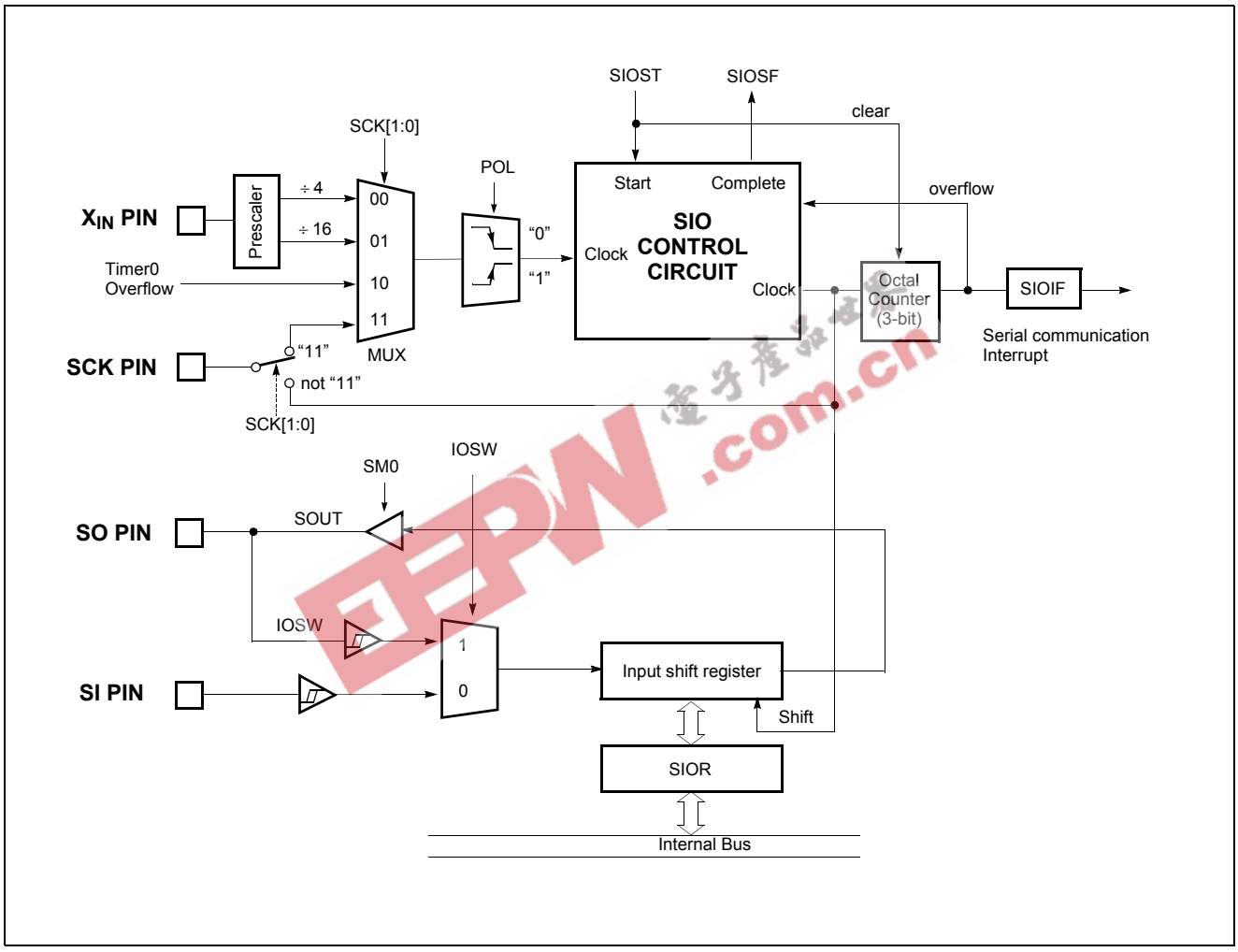


Figure 15-1 SIO Block Diagram

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

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

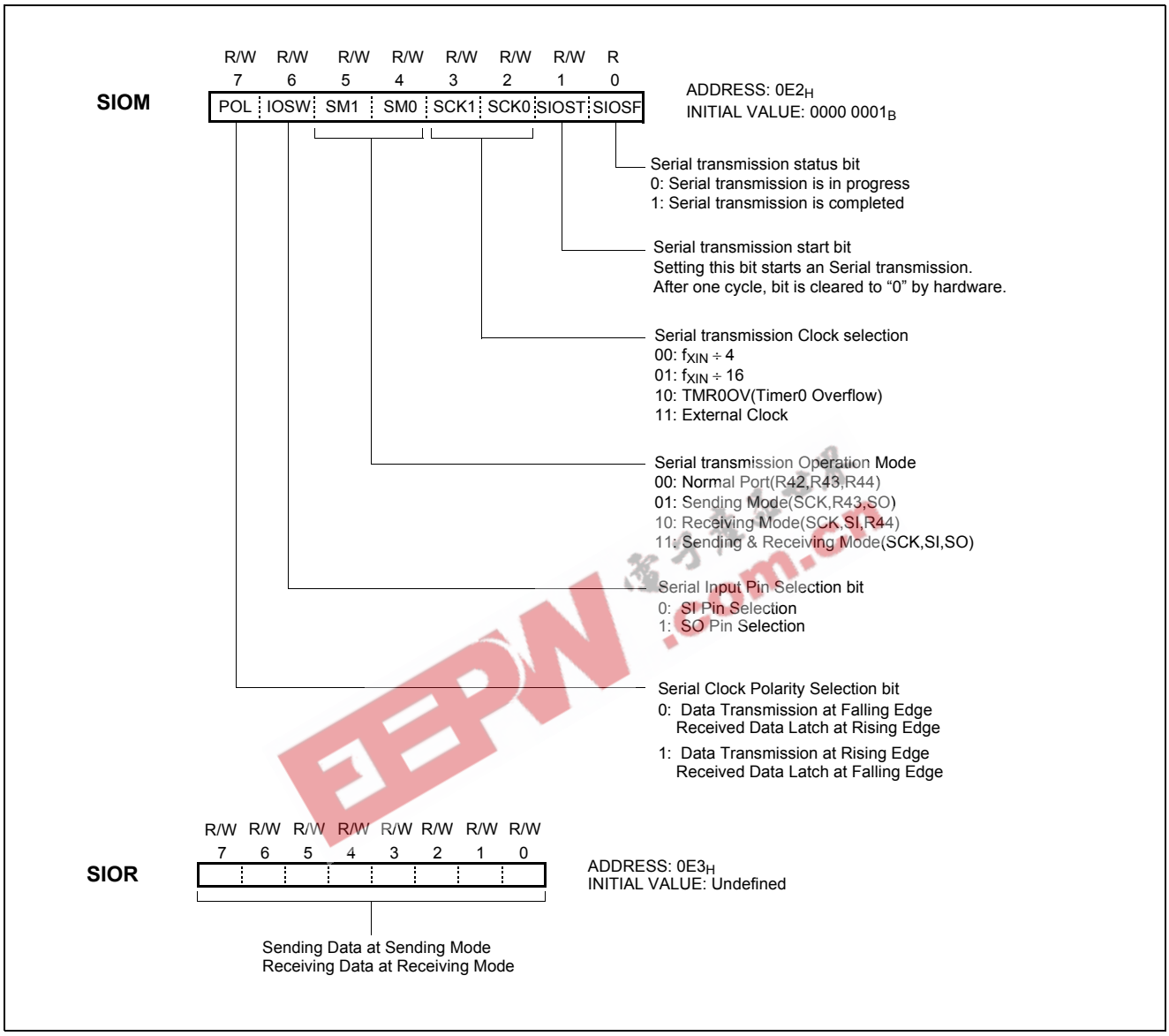


Figure 15-2 SIO Control Register

15.1 Transmission/Receiving Timing

The serial transmission is started by setting SIOST(bit1 of SIOM) to "1". After one cycle of SCK, SIOST is cleared automatically to "0". At the default state of POL bit clear, the serial output data from 8-bit shift register is output at falling edge of SCLK, and input data is latched at rising edge of SCLK pin (Refer to Figure 15-3). When transmission clock is counted 8 times, serial I/O counter is cleared as '0". Transmission clock is halted in "H" state and serial I/O interrupt (SIOIF) occurred.

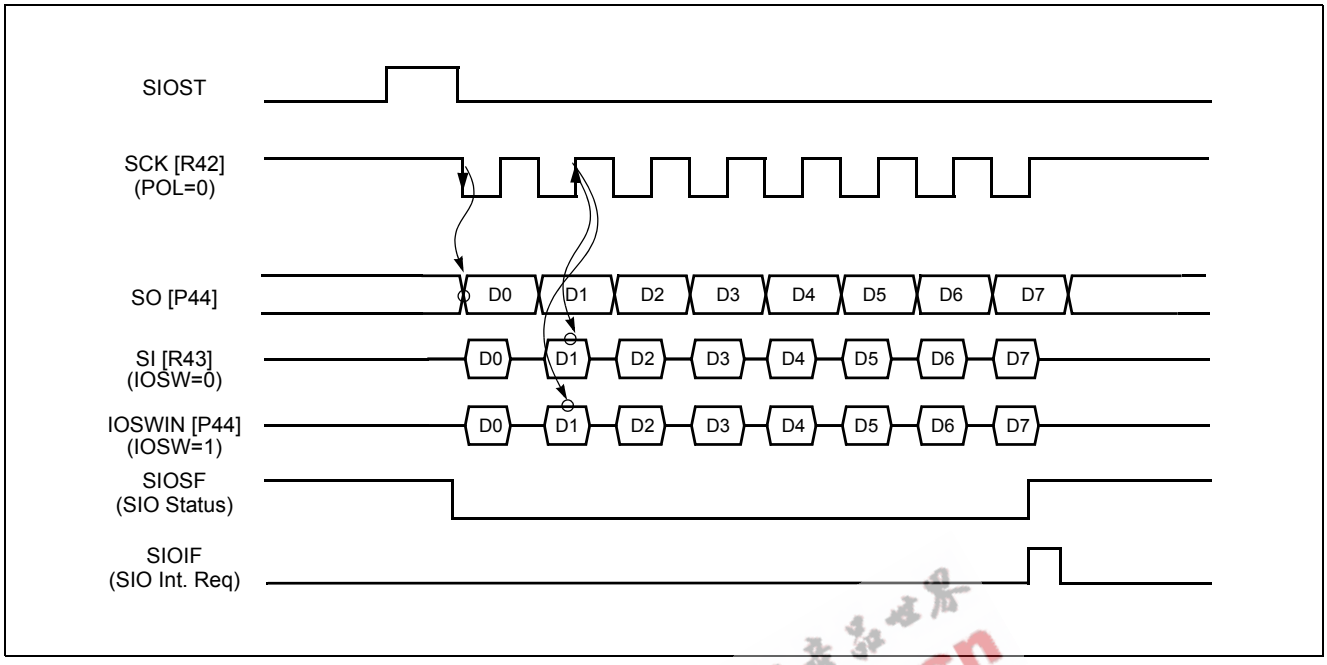


Figure 15-3 Serial I/O Timing Diagram at POL=0

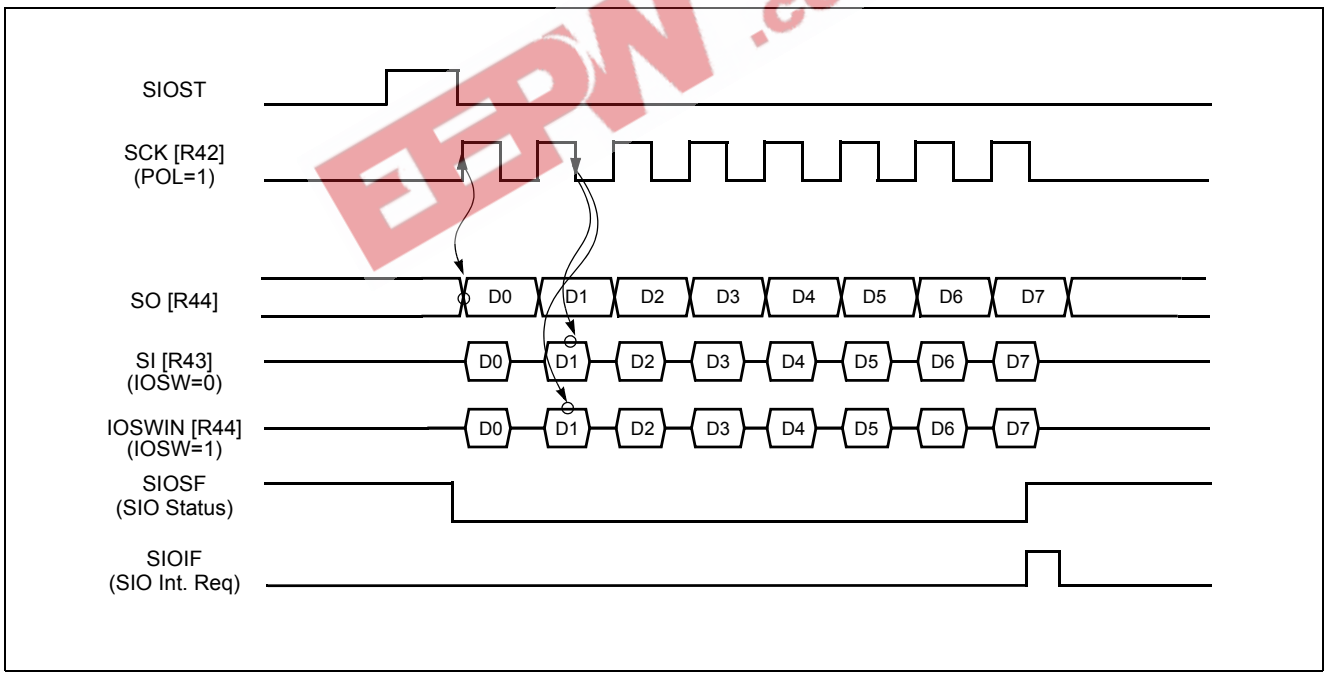


Figure 15-4 Serial I/O Timing Diagram at POL=1

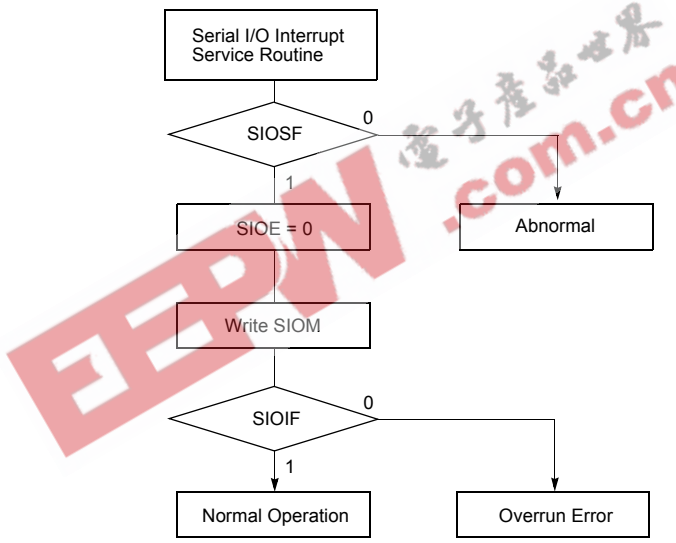
15.2 The usage of Serial I/O

1. Select transmission/receiving mode.
2. In case of sending mode, write data to be send to SIOR.
3. Set SIOST to “1” to start serial transmission.
4. The SIO interrupt is generated at the completion of SIO and SIOIF is set to “1”. In SIO interrupt service routine, correct transmission should be tested.
5. In case of receiving mode, the received data is acquired by reading the SIOR.

```
LDM     SIOR,#0AAh          ;set tx data
LDM     SIOM,#0011_1100b    ;set SIO mode
NOP
LDM     SIOM,#0011_1110b    ;SIO Start
```

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

15.3 The Method to Test Correct Transmission



- SIOE: Interrupt Enable Register High IENH(Bit1)

- SIOIF: Interrupt Request Flag Register High IRQH(Bit1)

Figure 15-5 Serial IO Method to Test Transmission

16. UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART)

16.1 UART Serial Interface Functions

The Universal Asynchronous Receiver / Transmitter (UART) enables full-duplex operation wherein one byte of data after the start bit is transmitted and received. The on-chip baud rate generator dedicated to UART enables communications using a wide range of selectable baud rates. In addition, a baud rate can also be defined by dividing clocks input to the ACLK pin.

The UART driver consists of RXR, TXR, ASIMR, ASISR and BRGCR register. Universal asynchronous serial I/O mode (UART) can be selected by ASIMR register. Figure 16-1 shows a block diagram of the UART driver.

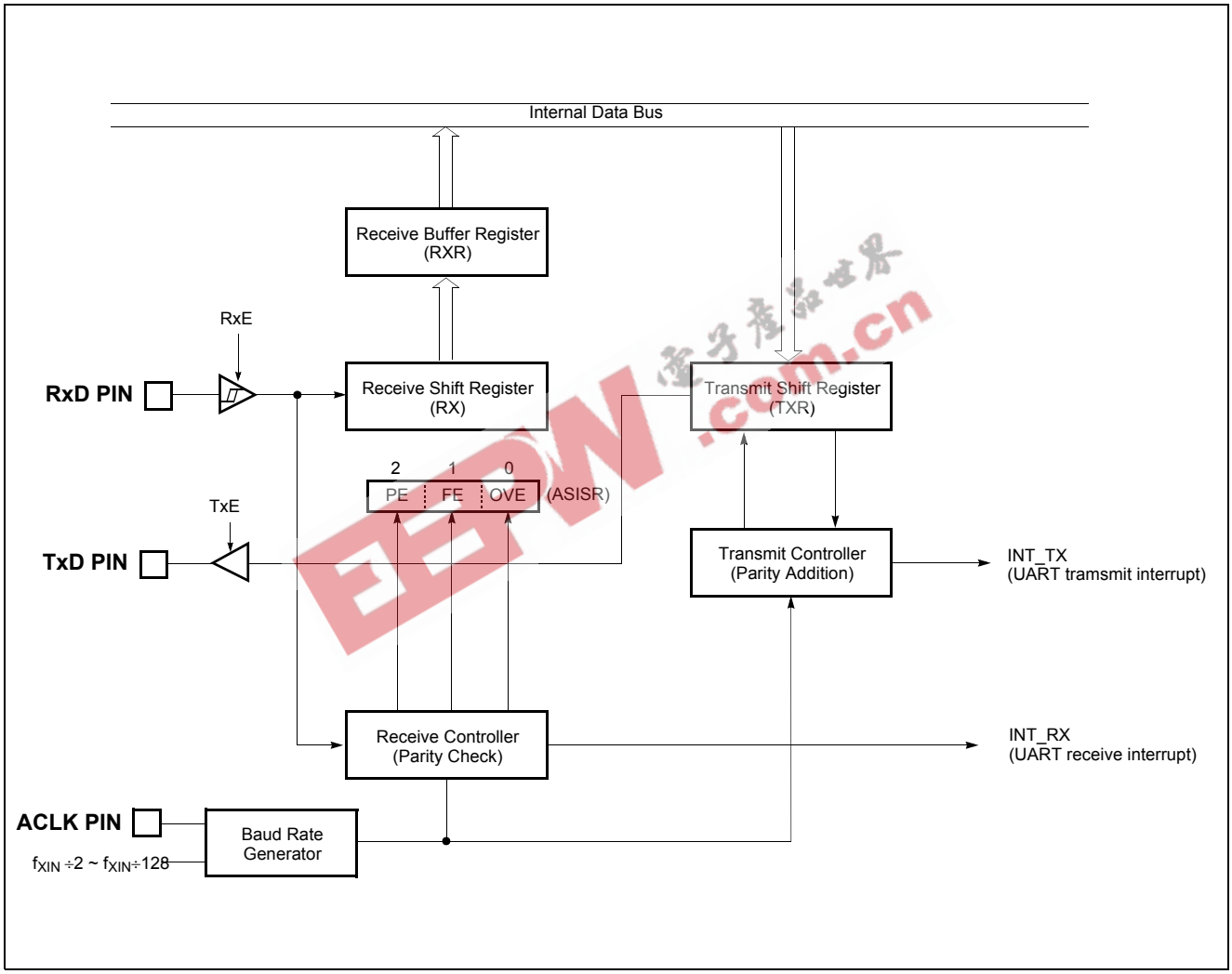


Figure 16-1 UART Block Diagram

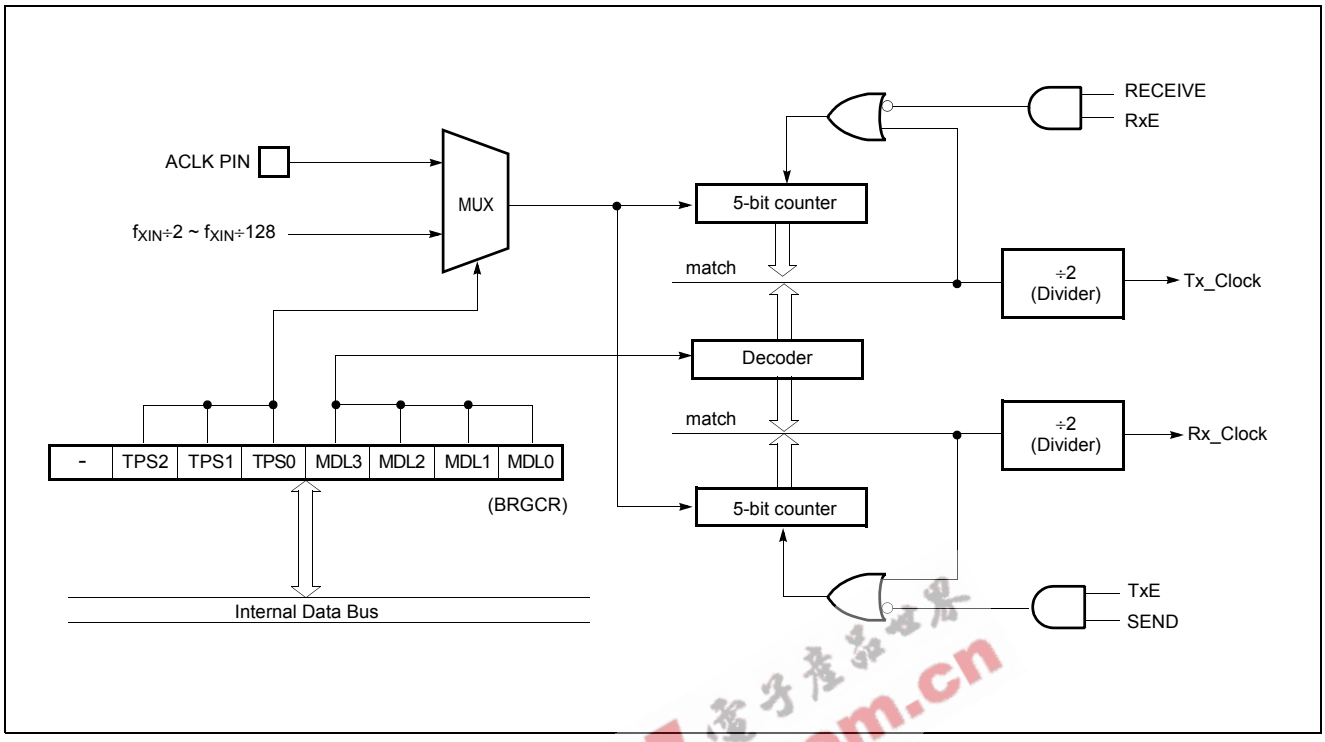


Figure 16-2 Baud Rate Generator Block Diagram

16.2 Serial Interface Configuration

The UART interface consists of the following hardware.

Item	Configuration
Register	Transmit shift register (TXR) Receive buffer register (RXR) Receive shift register
Control register	Serial interface mode register (ASIMR) Serial interface status register (ASISR) Baud rate generator control register (BRGCR)

Table 16-1 Serial Interface Configuration

Transmit shift register (TXR)

This is the register for setting transmit data. Data written to TXR is transmitted as serial data. When the data length is set as 7 bit, bit 0 to 6 of the data written to TXR are transferred as transmit data. Writing data to TXR starts the transmit operation.

TXR can be written by an 8 bit memory manipulation instruction. It cannot be read. The $\overline{\text{RESET}}$ input sets TXR to $0FF_H$.

Receive buffer register (RXR)

This register is used to hold receive data. When one byte of data is received, one byte of new receive data is transferred from the receive shift register (RXSR). When the data length is set as 7 bits, receive data is sent to bits 0 to 6 of RXR. In this case, the MSB of RXR always becomes 0.

RXR can be read by an 8 bit memory manipulation instruction. It cannot be written. The $\overline{\text{RESET}}$ input sets RXR to 00_H.

Receive shift register

This register converts serial data input via the RXD pin to paralleled data. When one byte of data is received at this register cannot be manipulated directly by a program.

Asynchronous serial interface mode register (ASIMR)

This is an 8 bit register that controls UART serial transfer operation. ASIMR is set by a 1 bit or 8 bit memory manipulation intruction. The $\overline{\text{RESET}}$ input sets ASIMR to 0000_-00-$_B$. Figure 16-3 shows the format of ASIMR The RXD / R04 and TXD / R05 pin function selection is shown in Table 16-2.

Note: *Do not switch the operation mode until the current serial transmit/receive operation has stopped.*

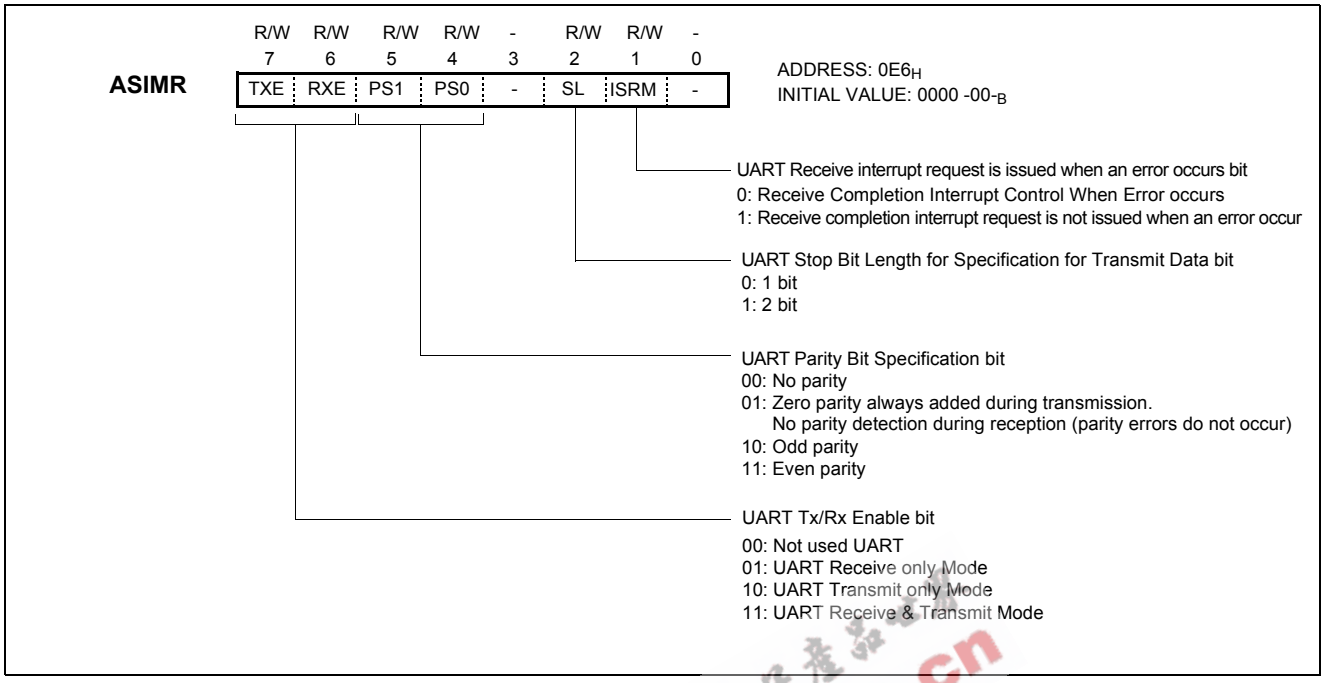


Figure 16-3 Asynchronous Serial Interface Mode register (ASIMR) Format

TXE (ASIMR.7)	RXE(ASIMR.6)	EC0(PSR0.4)	Operation Mode	RXD/R04	TXD/R05
0	0	X[1]	Operation Stop	R04	R05
0	1	0	UART mode (Receive only)	RXD	R05
1	0	X	UART mode (Transmit only)	R04	TXD
1	1	0	UART mode (Transmit and receive)	RXD	TXD

Table 16-2 UART mode and RXD/TXD pin function

1. X:The value "0" or "1" corresponding your operation

Asynchronous serial interface status register (ASISR)

When a receive error occurs during UART mode, this register indicates the type of error. ASISR can be read by an 8 bit memory manipulation instruction. The $\overline{\text{RESET}}$ input sets ASISR to ----_-000B. Figure 16-4 shows the format of ASISR..

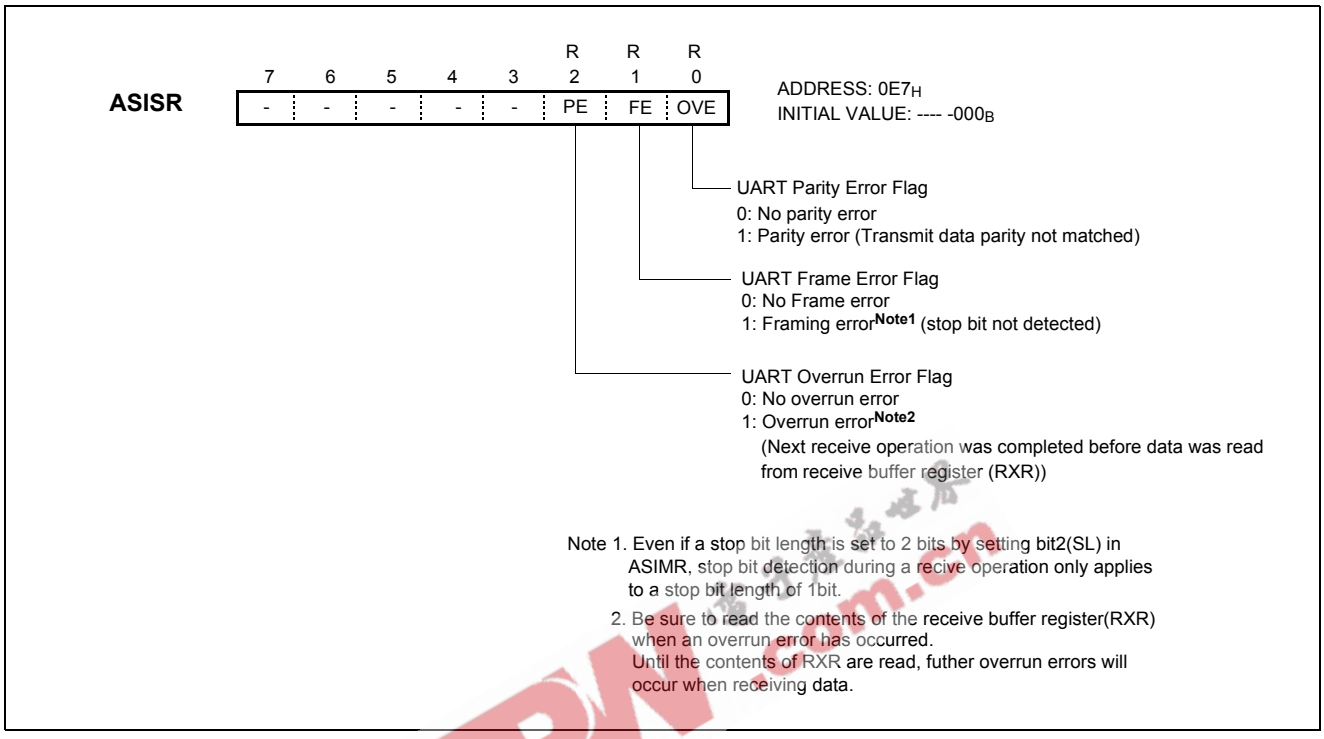
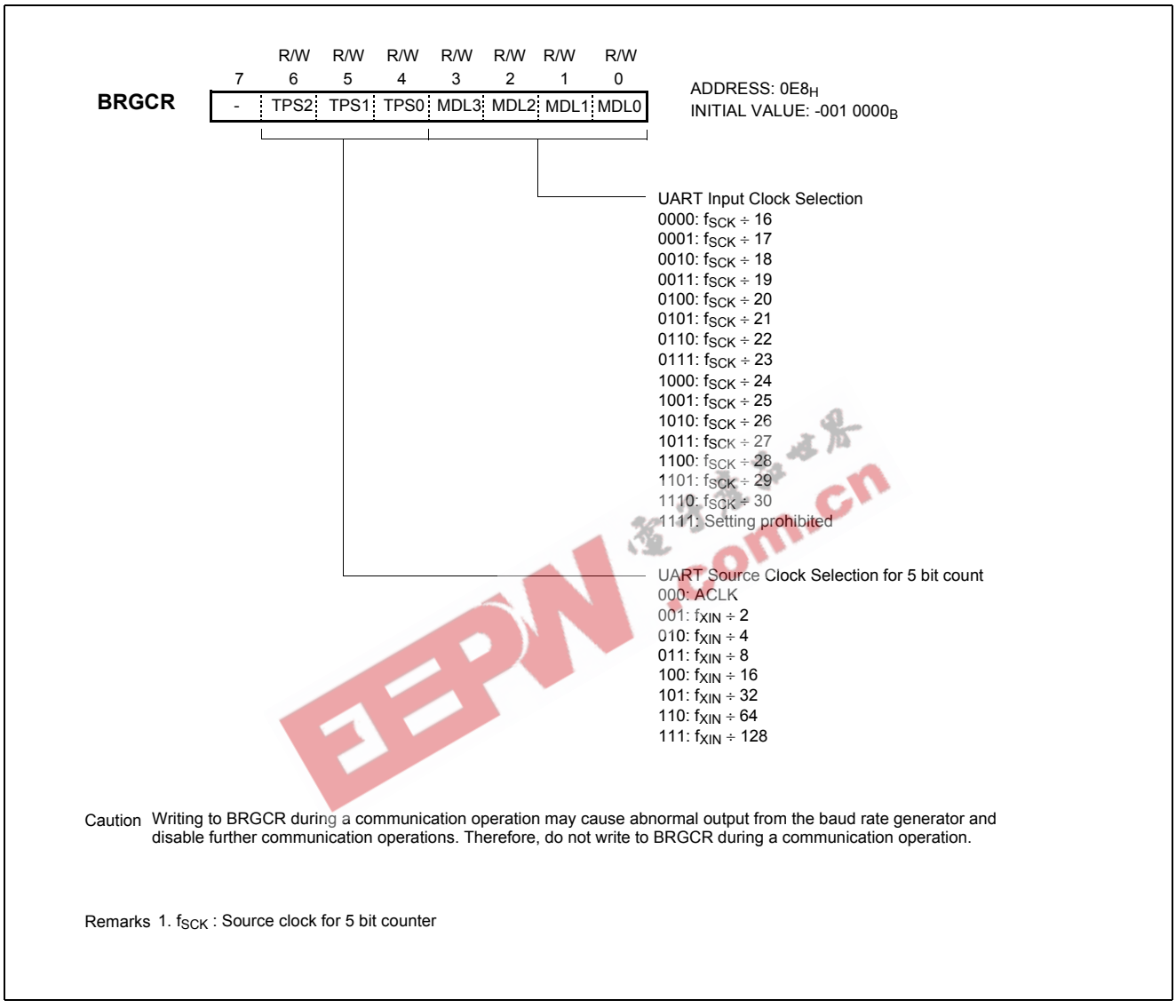


Figure 16-4 Asynchronous Serial Interface Status Register (ASISR) Format

Baud rate generator control register (BRGCR)

This register sets the serial clock for serial interface. BRGCR is set by an 8 bit memory manipulation instruction. The $\overline{\text{RESET}}$ input sets BRGCR to -001_0000B.

Figure 16-5 shows the format of BRGCR.



Caution Writing to BRGCR during a communication operation may cause abnormal output from the baud rate generator and disable further communication operations. Therefore, do not write to BRGCR during a communication operation.

Remarks 1. f_{SCK} : Source clock for 5 bit counter

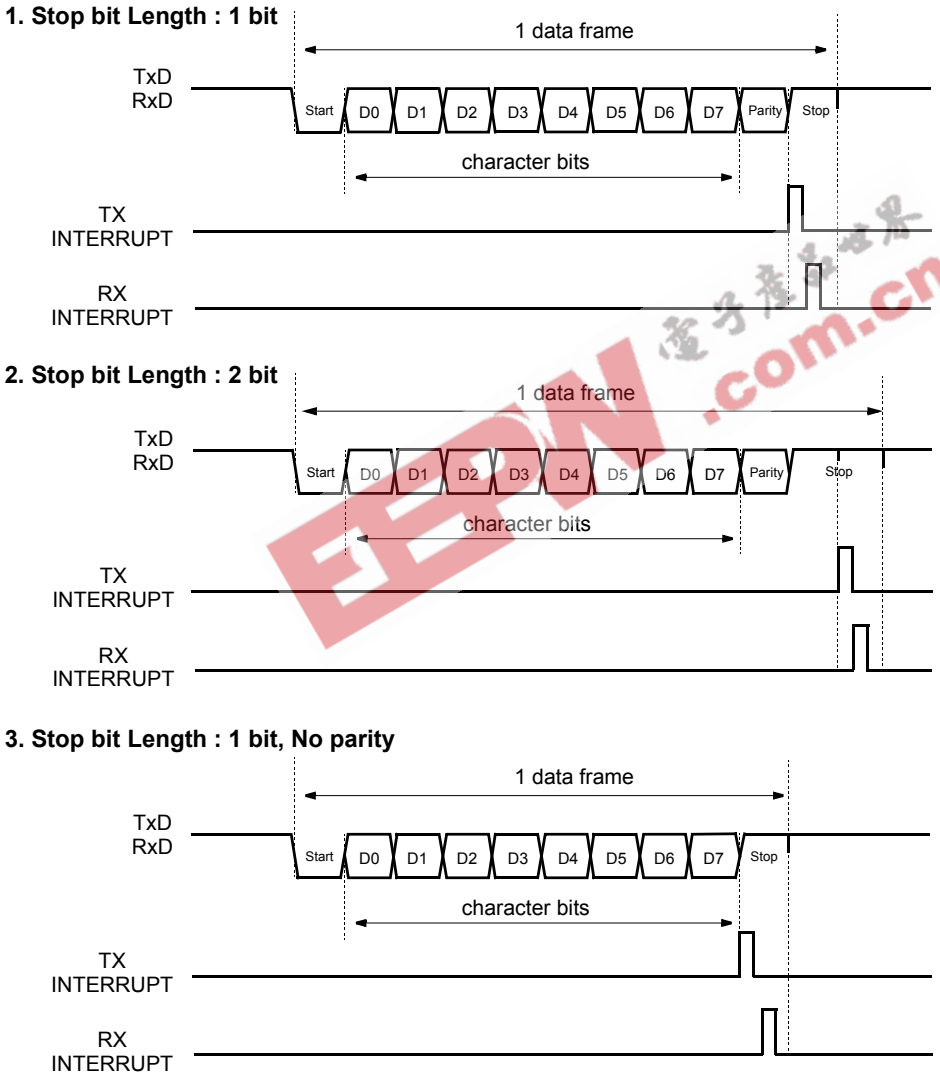
Figure 16-5 Baud Rate Generator Control Register (BRGCR) Format

16.3 Communication operation

The transmit operation is enabled when bit 7 (TXE) of the asynchronous serial interface mode register (ASIMR) is set to 1. The transmit operation is started when transmit data is written to the transmit shift register (TXR). The timing of the transmit completion interrupt request is shown in Figure 16-6 .

The receive operation is enabled when bit 6 (RXE) of the asynchronous serial interface mode register (ASIMR) is set to 1, and input via the RxD pin is sampled. The serial clock specified by ASIMR is used to sample the RxD pin.

Once reception of one data frame is completed, a receive completion interrupt request (INT_RX) occurs. Even if an error has occurred, the receive data in which the error occurred is still transferred to RXR. When ASIMR bit 1 (ISRM) is cleared to 0 upon occurrence of an error, and INT_RX occurs. When ISRM bit is set to 1, INT_RX does not occur in case of error occurrence. Figure 16-6 shows the timing of the asynchronous serial interface receive completion interrupt request.



1 data frame consists of following bits.
- Start bit : 1 bit
- Character bits : 8 bits
- Parity bit : Even parity, Odd parity, Zero parity, No parity
- Stop bit(s) : 1 bit or 2 bits

Figure 16-6 UART data format and interrupt timing diagram

16.4 Relationship between main clock and baud rate

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

Baud Rate (bps)	f_{XIN}=11.0592M		f_{XIN}=10.0M		f_{XIN}=8.0M		f_{XIN}=4.0M		f_{XIN}=2.0M	
	BRGCR	ERR(%)	BRGCR	ERR(%)	BRGCR	ERR(%)	BRGCR	ERR (%)	BRGCR	ERR(%)
600	-	-	-	-	-	-	7AH	0.16	6AH	0.16
1200	-	-	-	-	7AH	0.16	6AH	0.16	5AH	0.16
2400	72H	0.00	70H	1.73	6AH	0.16	5AH	0.16	4AH	0.16
4800	62H	0.00	60H	1.73	5AH	0.16	4AH	0.16	3AH	0.16
9600	52H	0.00	50H	1.73	4AH	0.16	3AH	0.16	2AH	0.16
19200	42H	0.00	40H	1.73	3AH	0.16	2AH	0.16	1AH	0.16
31250	36H	0.53	34H	0.00	30H	0.00	20H	0.00	10H	0.00
38400	32H	0.00	30H	1.73	2AH	0.16	1AH	0.16	-	-
57600	28H	0.00	26H	1.35	21H	2.11	11H	2.12	-	-
76800	22H	0.00	20H	1.73	1AH	0.16	-	-	-	-
115200	18H	0.00	16H	1.36	11H	2.12	-	-	-	-

$$\text{Baud Rate} = f_{XIN} \div (2^{n+1}(k+16))$$

Remarks 1. f_{XIN} : Main system clock oscillation frequency
When ACLK is selected as the source clock of the 5-bit counter, substitute the input clock frequency to ACLK pin for in the above expression.
2. f_{SCK} : Source clock for 5 bit counter
3. n : Value set via TPS0 to TPS2 ($0 \le n \le 7$)
4. k : Source clock for 5 bit counter ($0 \le k \le 14$)

Figure 16-7 Relationship between main clock and Baud Rate

17. 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}= 4MHz) by user software.

A 50% duty pulse can be output to R13 / BUZO pin to use for piezo-electric buzzer drive. Pin R13 is assigned for output port of Buzzer driver by setting the bit 2 of PSR1(address $0F9_H$) to "1". For PSR1 register, refer to Figure 17-2 .

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.

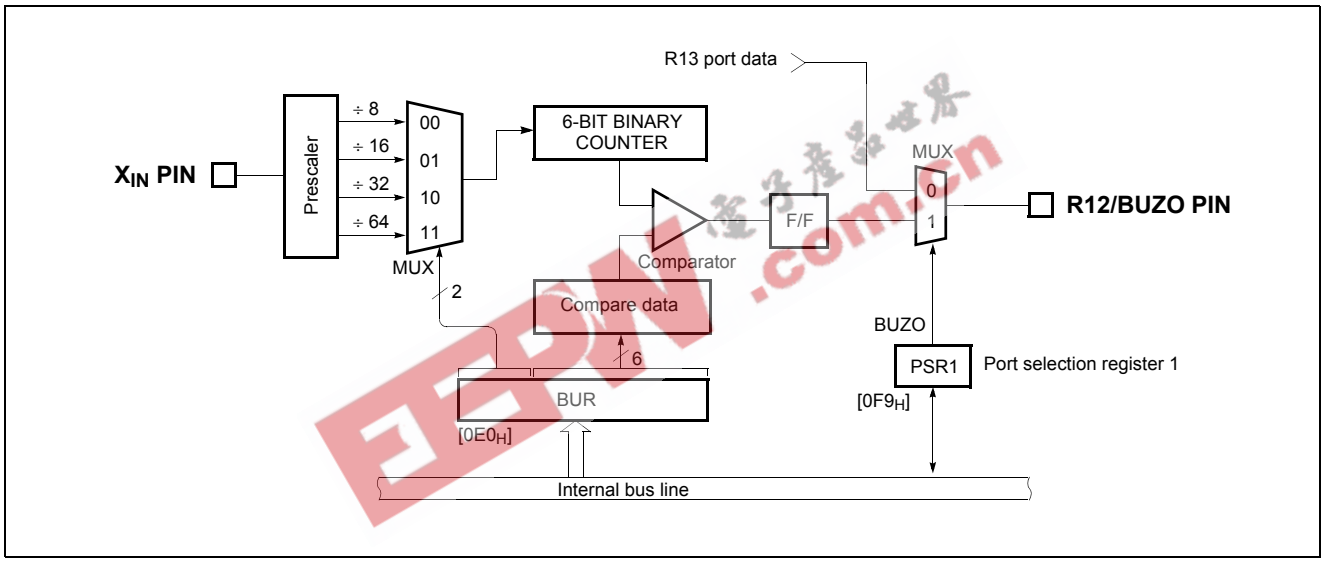


Figure 17-1 Block Diagram of Buzzer Driver

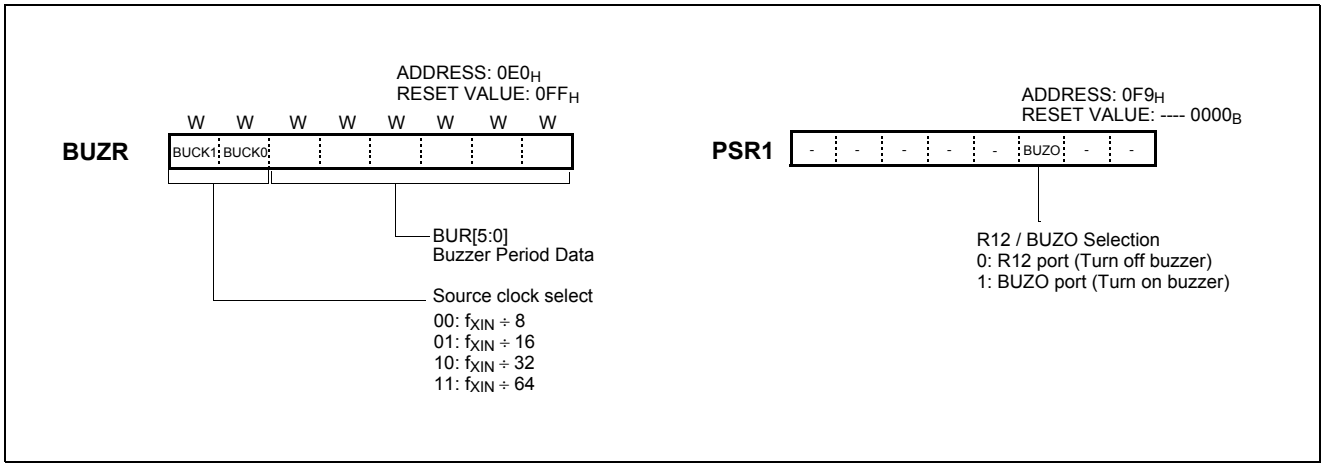


Figure 17-2 Buzzer Register & PSR1

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

When main-frequency is 4MHz, buzzer frequency is shown as below Table 17-1.

BUR [5:0]	BUR[7:6]			
	00	01	10	11
00	250.000	125.000	62.500	31.250
01	125.000	62.500	31.250	15.625
02	83.333	41.667	20.833	10.417
03	62.500	31.250	15.625	7.813
04	50.000	25.000	12.500	6.250
05	41.667	20.833	10.417	5.208
06	35.714	17.857	8.929	4.464
07	31.250	15.625	7.813	3.906
08	27.778	13.889	6.944	3.472
09	25.000	12.500	6.250	3.125
0A	22.727	11.364	5.682	2.841
0B	20.833	10.417	5.208	2.604
0C	19.231	9.615	4.808	2.404
0D	17.857	8.929	4.464	2.232
0E	16.667	8.333	4.167	2.083
0F	15.625	7.813	3.906	1.953
10	14.706	7.353	3.676	1.838
11	13.889	6.944	3.472	1.736
12	13.158	6.579	3.289	1.645
13	12.500	6.250	3.125	1.563
14	11.905	5.952	2.976	1.488
15	11.364	5.682	2.841	1.420
16	10.870	5.435	2.717	1.359
17	10.417	5.208	2.604	1.302
18	10.000	5.000	2.500	1.250
19	9.615	4.808	2.404	1.202
1A	9.259	4.630	2.315	1.157
1B	8.929	4.464	2.232	1.116
1C	8.621	4.310	2.155	1.078
1D	8.333	4.167	2.083	1.042
1E	8.065	4.032	2.016	1.008
1F	7.813	3.906	1.953	0.977

BUR [5:0]	BUR[7:6]			
	00	01	10	11
20	7.576	3.788	1.894	0.947
21	7.353	3.676	1.838	0.919
22	7.143	3.571	1.786	0.893
23	6.944	3.472	1.736	0.868
24	6.757	3.378	1.689	0.845
25	6.579	3.289	1.645	0.822
26	6.410	3.205	1.603	0.801
27	6.250	3.125	1.563	0.781
28	6.098	3.049	1.524	0.762
29	5.952	2.976	1.488	0.744
2A	5.814	2.907	1.453	0.727
2B	5.682	2.841	1.420	0.710
2C	5.556	2.778	1.389	0.694
2D	5.435	2.717	1.359	0.679
2E	5.319	2.660	1.330	0.665
2F	5.208	2.604	1.302	0.651
30	5.102	2.551	1.276	0.638
31	5.000	2.500	1.250	0.625
32	4.902	2.451	1.225	0.613
33	4.808	2.404	1.202	0.601
34	4.717	2.358	1.179	0.590
35	4.630	2.315	1.157	0.579
36	4.545	2.273	1.136	0.568
37	4.464	2.232	1.116	0.558
38	4.386	2.193	1.096	0.548
39	4.310	2.155	1.078	0.539
3A	4.237	2.119	1.059	0.530
3B	4.167	2.083	1.042	0.521
3C	4.098	2.049	1.025	0.512
3D	4.032	2.016	1.008	0.504
3E	3.968	1.984	0.992	0.496
3F	3.907	1.953	0.977	0.488

Table 17-1 buzzer frequency (kHz unit)

18. INTERRUPTS

TheMC80F0104/0204 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 18-1 and interrupt priority is shown in Table 18-1.

The External Interrupts INT0 ~ INT3 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, INT1IF, INT2IF and INT3IF 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 ~ Timer 3 Interrupts are generated by T0IF, T1IF, T2IF and T3IF 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.

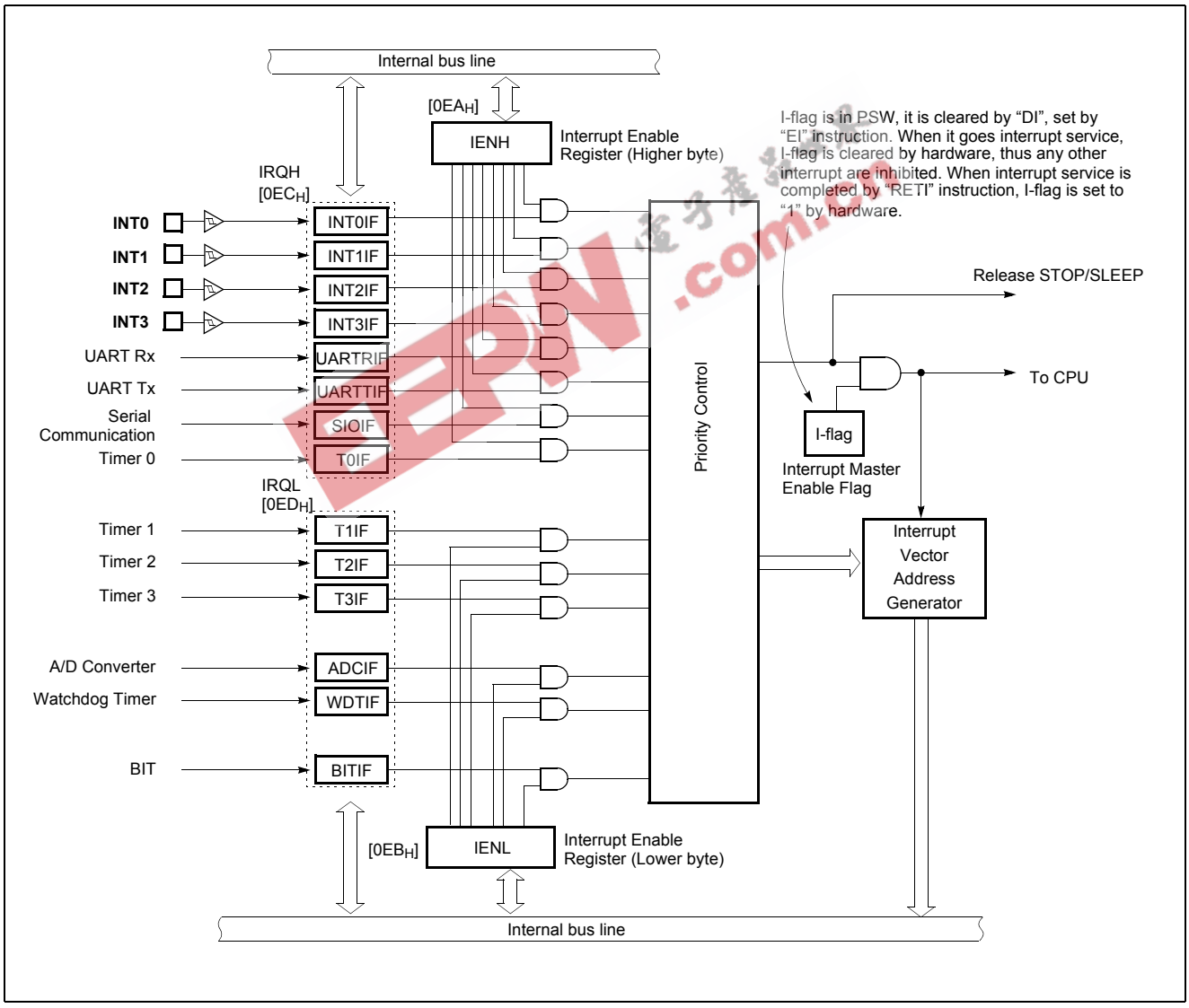


Figure 18-1 Block Diagram of Interrupt

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

The UART receive or transmit interrupts are generated by UARTRIF or UARTTIF are set by completion of UART data reception or transmission.

The SIO interrupt is generated by SIOIF which is set by completion of SIO data reception or transmission.

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 18-1 shows the Interrupt priority.

Vector addresses are shown in Figure 8-6 . Interrupt enable registers are shown in Figure 18-2 . 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
External Interrupt 2	INT2	4
External Interrupt 3	INT3	5
UART Rx Interrupt	INT_RX	6
UART Tx Interrupt	INT_TX	7
Serial Input/Output	SIO	8
Timer/Counter 0	Timer 0	9
Timer/Counter 1	Timer 1	10
Timer/Counter 2	Timer 2	11
Timer/Counter 3	Timer 3	12
ADC Interrupt	ADC	13
Watchdog Timer	WDT	14
Basic Interval Timer	BIT	15

Table 18-1 Interrupt Priority

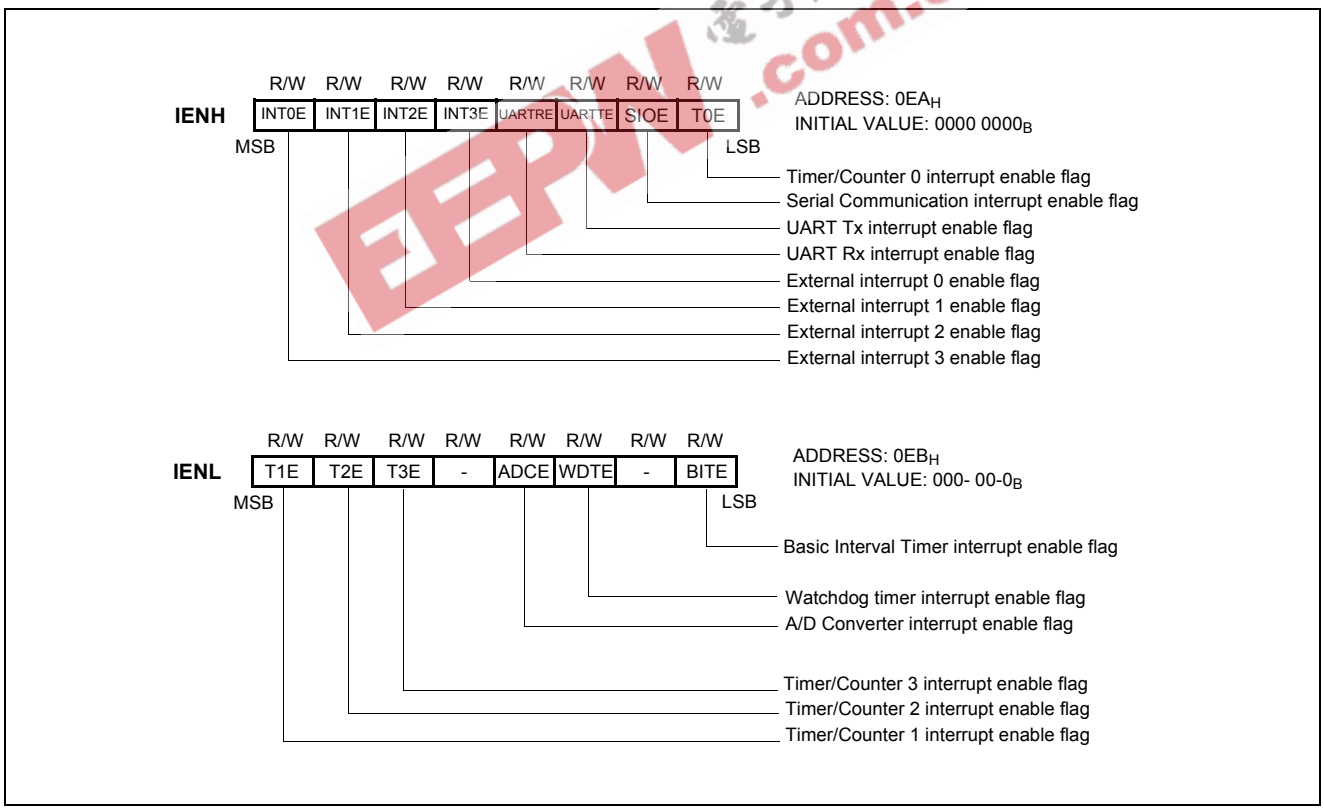


Figure 18-2 Interrupt Enable Flag Register

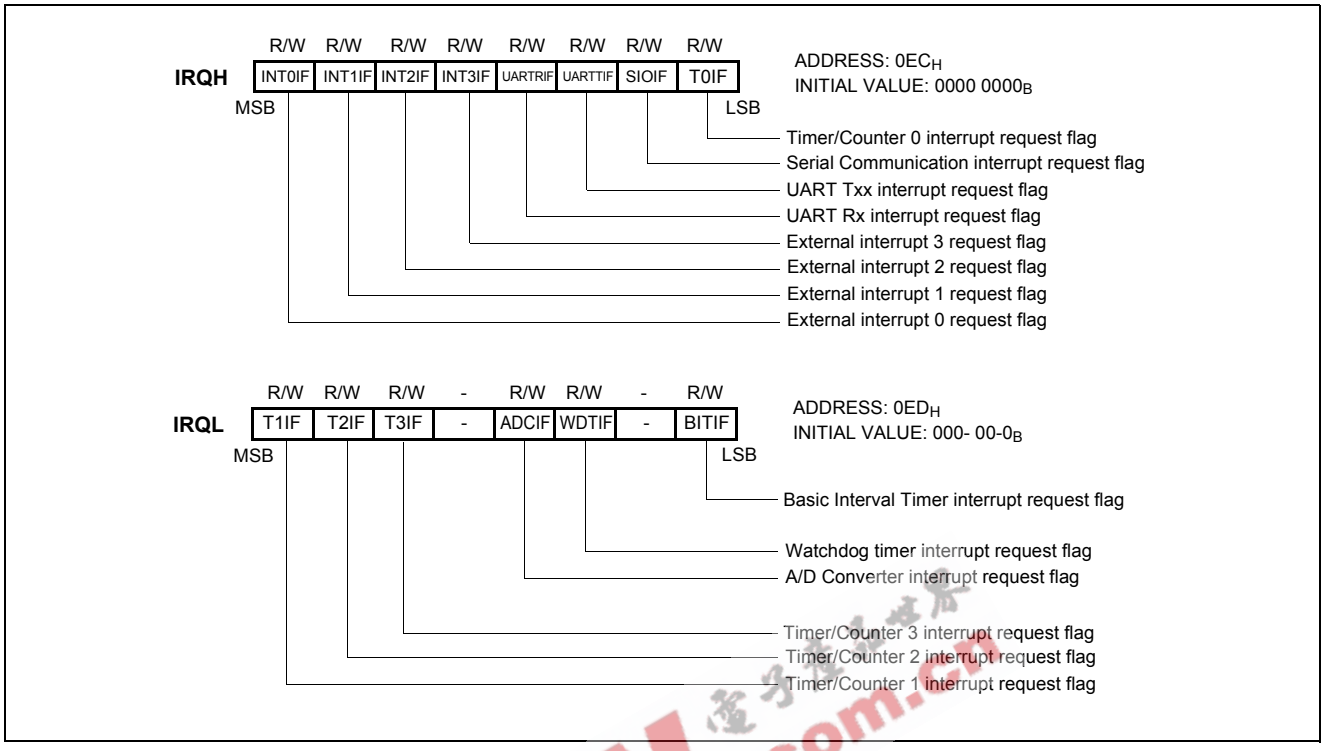


Figure 18-3 Interrupt Request Flag Register

18.1 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} (2μs at f_{XIN}=4MHz) after the completion of the current instruction execution. The interrupt service task is terminated upon execution of an interrupt return instruction [RETI].

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

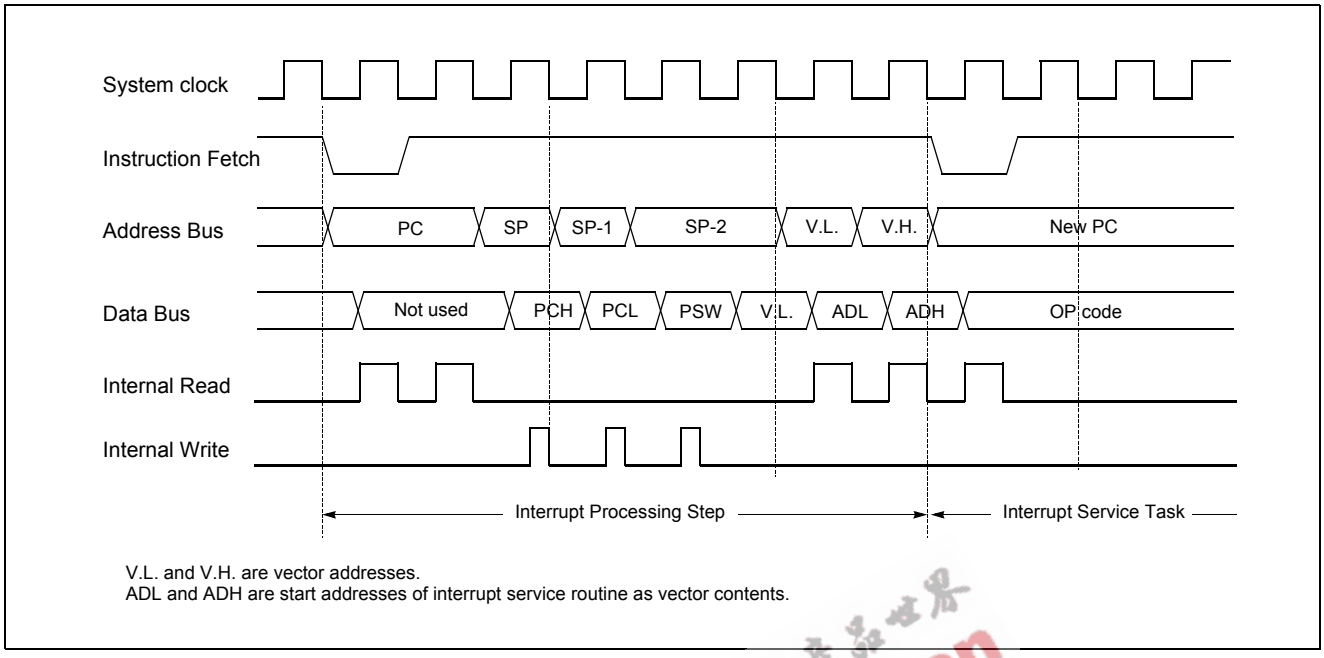
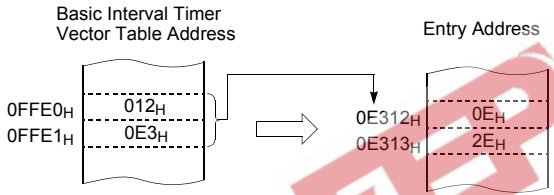


Figure 18-4 Timing chart of Interrupt Acceptance and Interrupt Return Instruction



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.

18.1.2 Saving/Restoring General-purpose Register

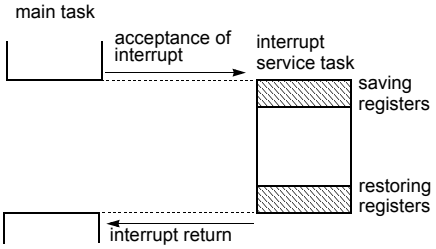
During interrupt acceptance processing, the program counter and the program status word are automatically saved on the stack, but accumulator and other registers are not saved itself. These registers are saved by the software if necessary. Also, when multiple interrupt services are nested, it is necessary 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:  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;



18.2 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 18-5 .

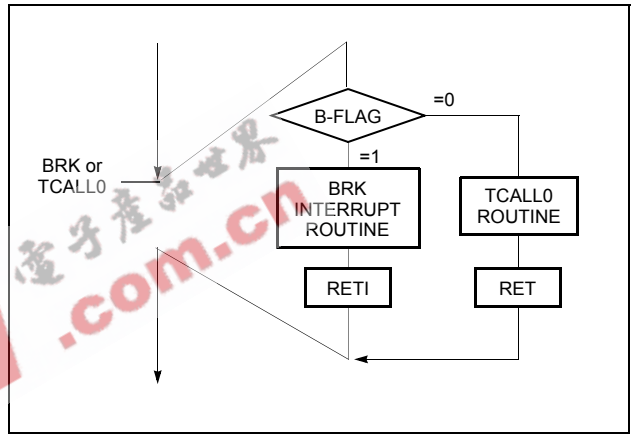


Figure 18-5 Execution of BRK/TCALL0

18.3 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.

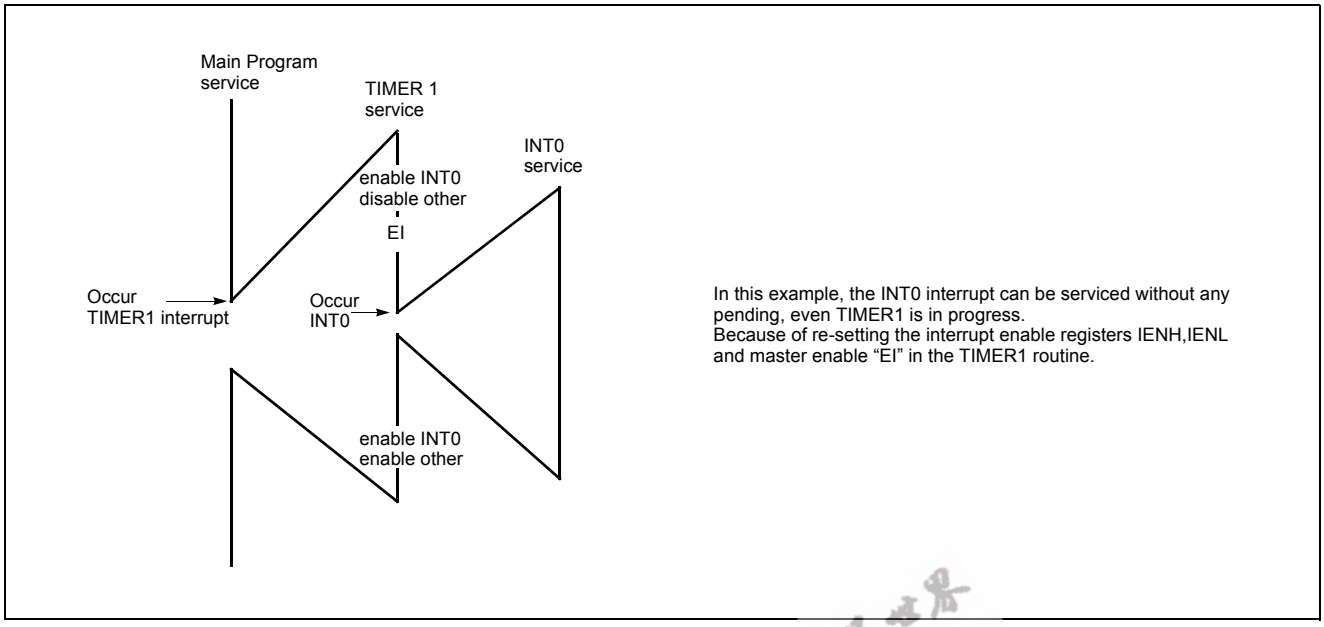


Figure 18-6 Execution of Multi Interrupt

Example: During Timer1 interrupt is in progress, INT0 interrupt serviced without any suspend.

```
TIMER1:  PUSH   A
         PUSH   X
         PUSH   Y
         LDM    IENH,#80H    ;Enable INT0 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
```

18.4 External Interrupt

The external interrupt on INT0, INT1, INT2 and INT3 pins are edge triggered depending on the edge selection register IEDS (address $0EE_H$) as shown in Figure 18-7 .

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

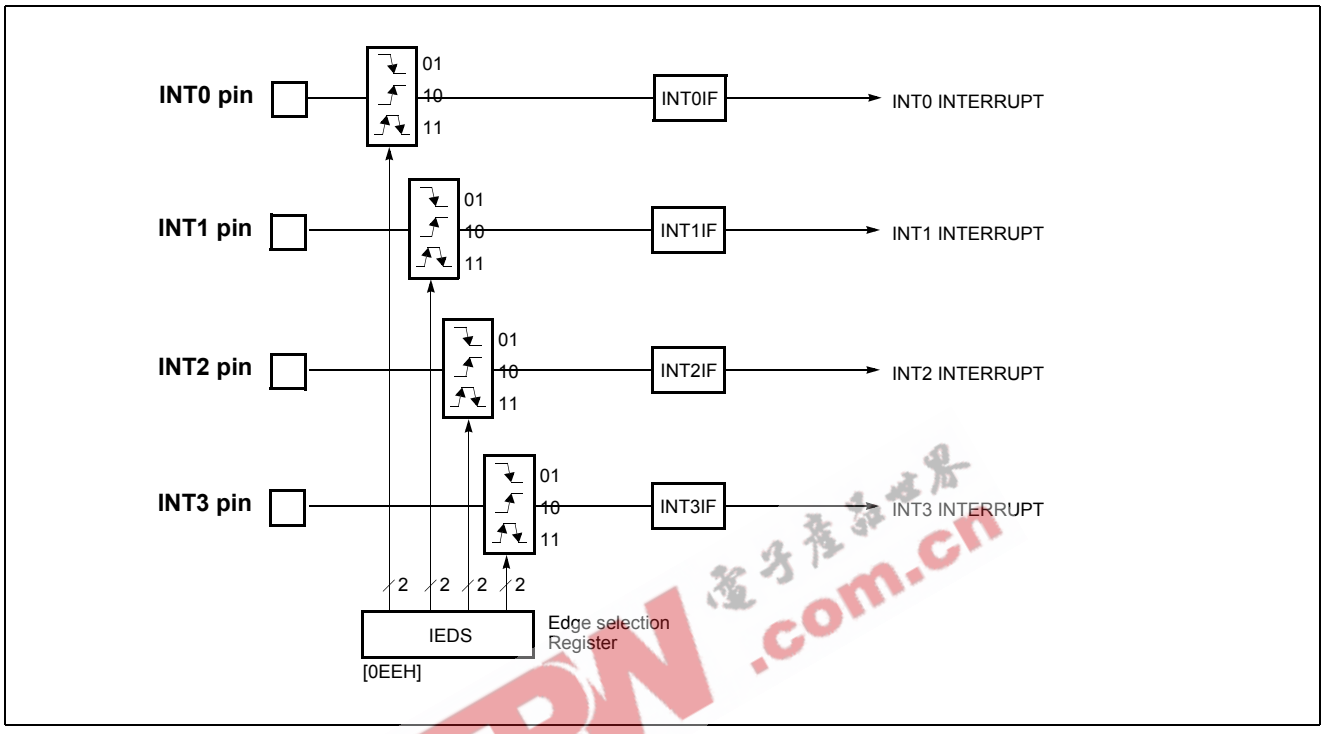


Figure 18-7 External Interrupt Block Diagram

INT0 ~ INT3 are multiplexed with general I/O ports (R11, R12, R03, R00). 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 INT2

```
        :
;**** Set external interrupt port as pull-up state.
        LDM     PU1,#0000_0101B
        ;
;**** Set port as an external interrupt port
        LDM     PSR0,#0000_0101B
        ;
;**** Set Falling-edge Detection
        LDM     IEDS,#0001_0001B
        :
```

Response Time

The INT0 ~ INT3 edge are latched into INT0IF ~ INT3IF 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 18-8 shows interrupt response timings.

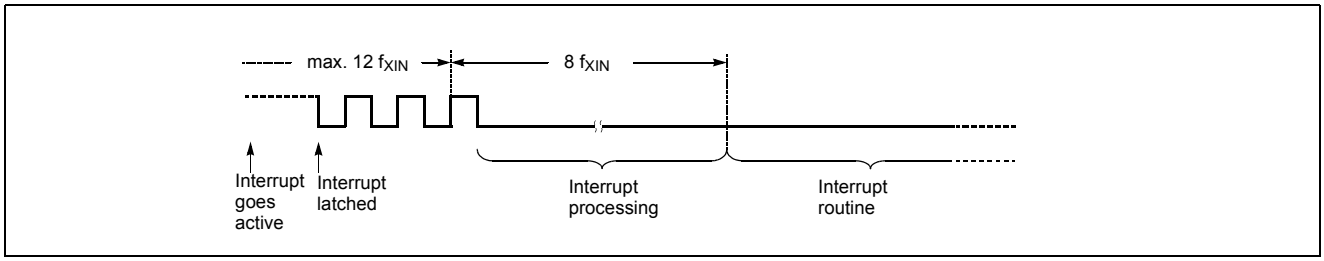


Figure 18-8 Interrupt Response Timing Diagram

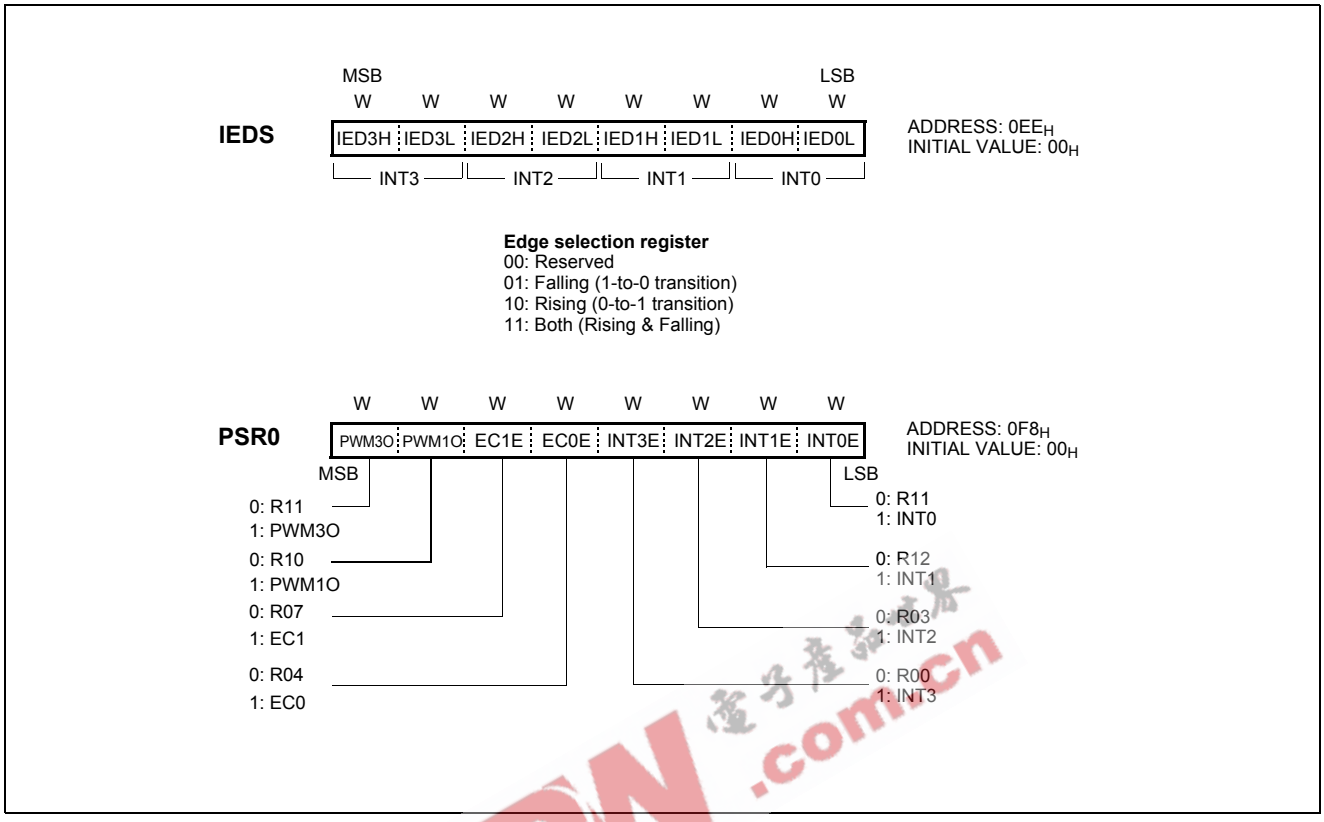


Figure 18-9 IEDS register and Port Selection Register PSR0

19. POWER SAVING OPERATION

TheMC80F0104/0204 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 19-1 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”.

19.1 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 19-1. 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.

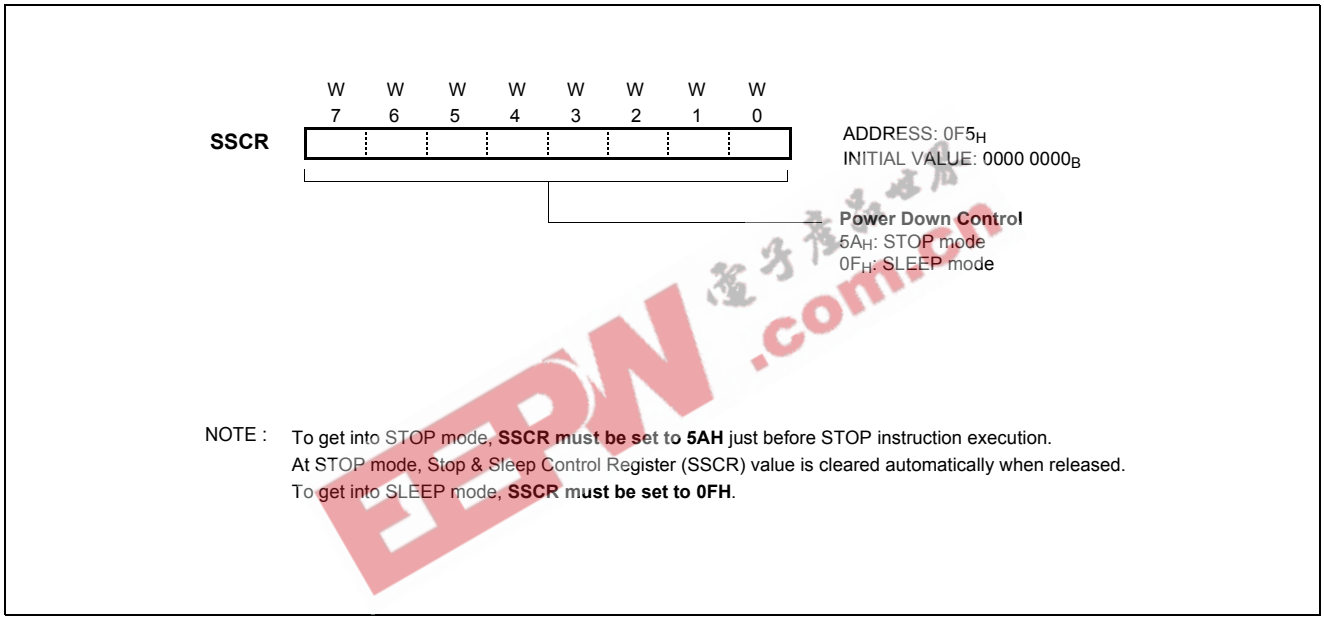


Figure 19-1 STOP and SLEEP Control Register

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 19-4)

When exit from SLEEP mode by reset, enough oscillation stabilizing time is required to normal operation. Figure 19-3 shows the timing diagram. When released from the SLEEP mode, the Basic interval timer is activated on wake-up. It is increased from 00_H until FF_H. 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 19-2 . By reset, exit from SLEEP mode is shown in Figure 19-3 .

.

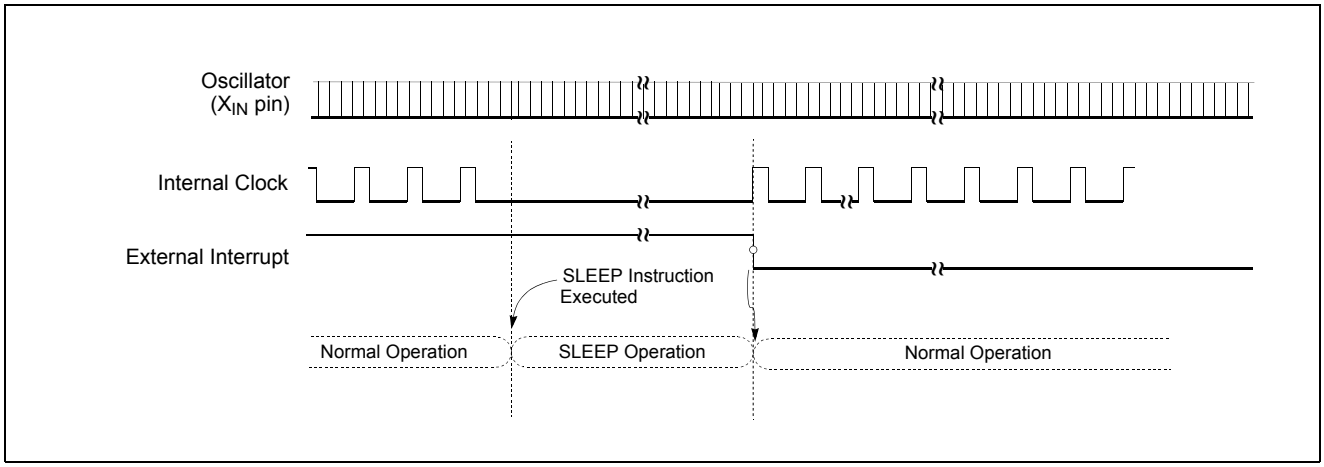


Figure 19-2 SLEEP Mode Release Timing by External Interrupt

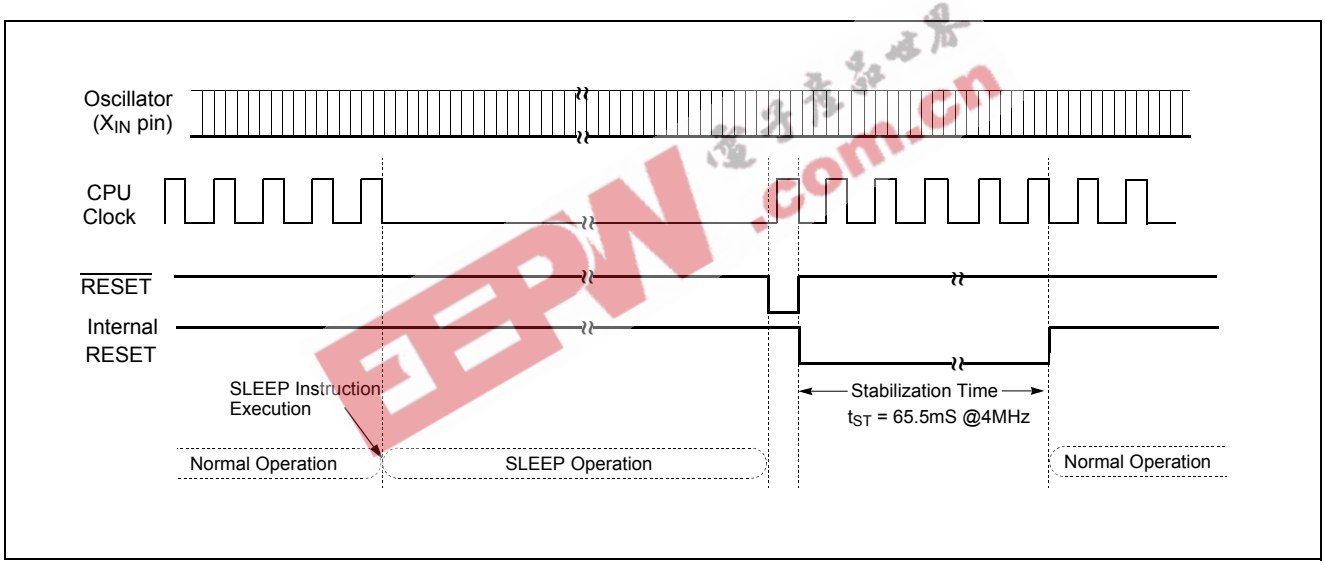


Figure 19-3 Timing of SLEEP Mode Release by Reset

19.2 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 "5A$_H$". (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)*

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)	Stop
Timer/Counter	Halted (Only when the event counter mode is enabled, timer operates normally)	Operates Continuously
Buzzer, ADC	Stop	Stop
SIO	Only operate with external clock	Only operate with external clock
UART	Only operate with external clock	Only operate with external clock
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(EC0,1), SIO, UART(using ACLK), Watchdog Timer (RC-WDT mode), External Interrupt	Reset, All Interrupts

Table 19-1 Peripheral Operation During Power Saving Mode

Release the STOP mode

The source for exit from STOP mode is hardware reset, external interrupt, Timer(EC0,1), Watch Timer, WDT, SIO or UART. 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 19-4)

When exit from Stop mode by external interrupt, enough oscillation stabilizing time is required to normal operation. Figure 19-5 shows the timing diagram. When released from the Stop mode, the Basic interval timer is activated on wake-up. It is increased from 00_H until FF_H. 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 19-6 .

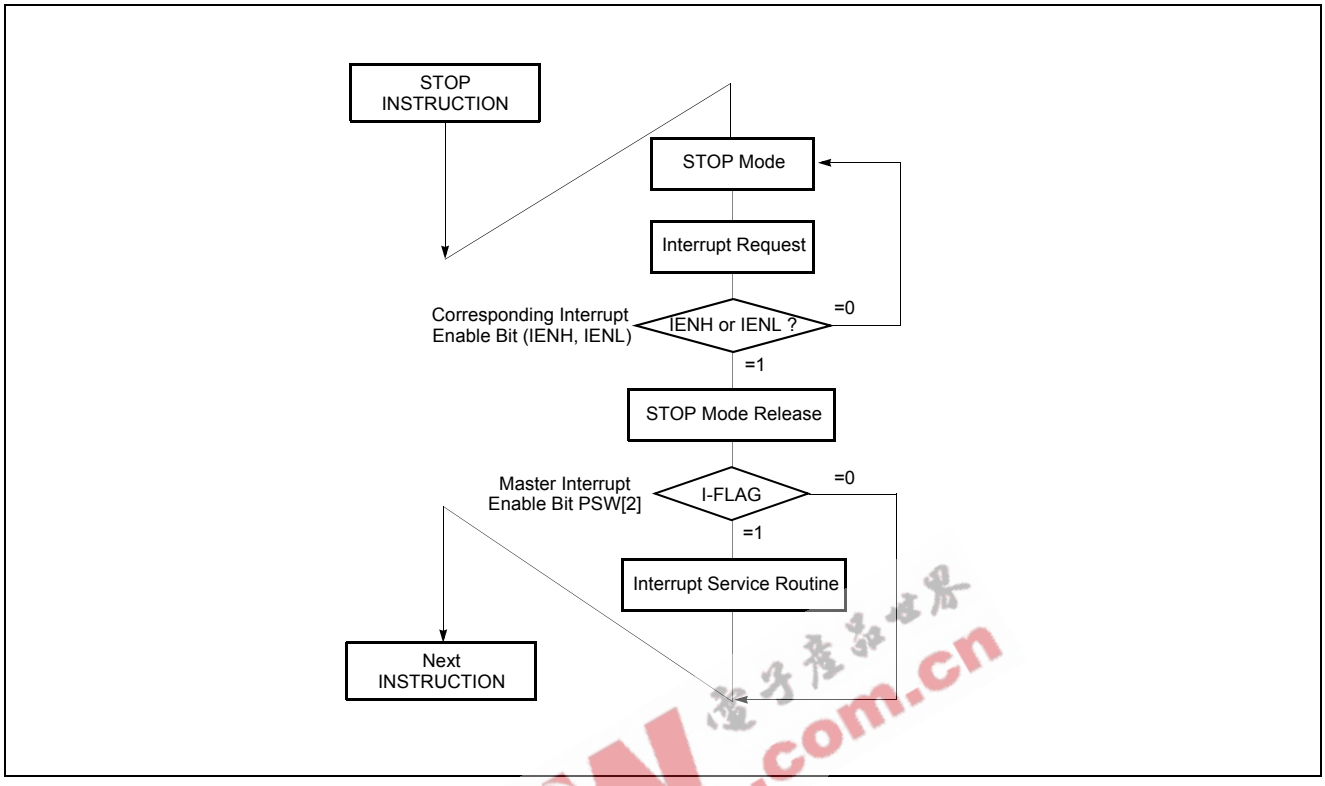


Figure 19-4 STOP Releasing Flow by Interrupts

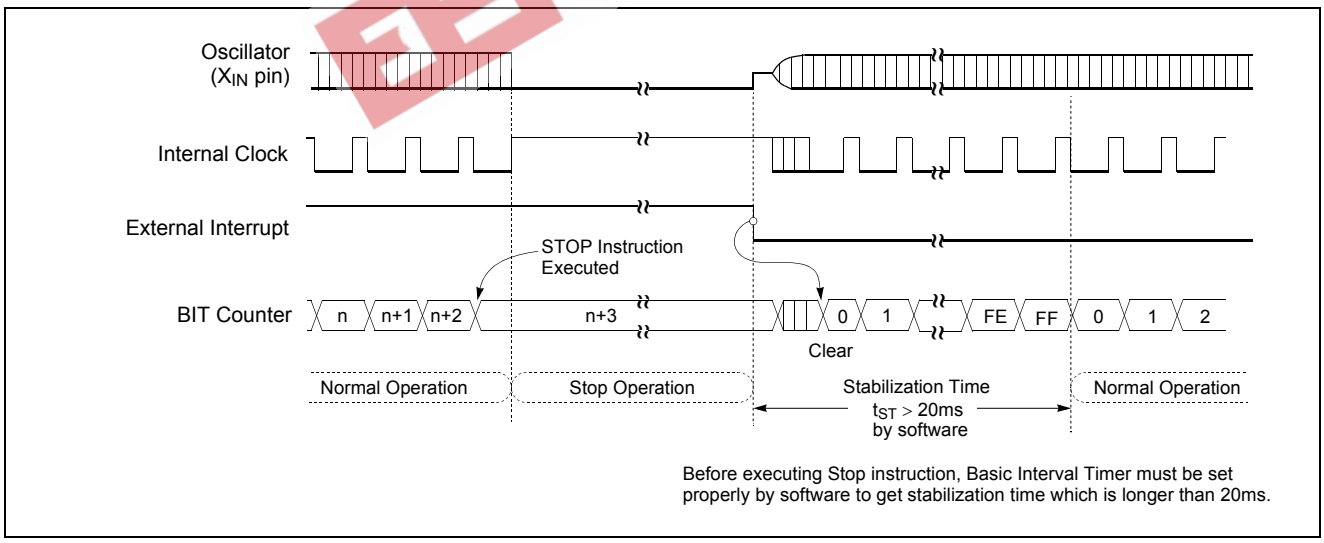


Figure 19-5 STOP Mode Release Timing by External Interrupt

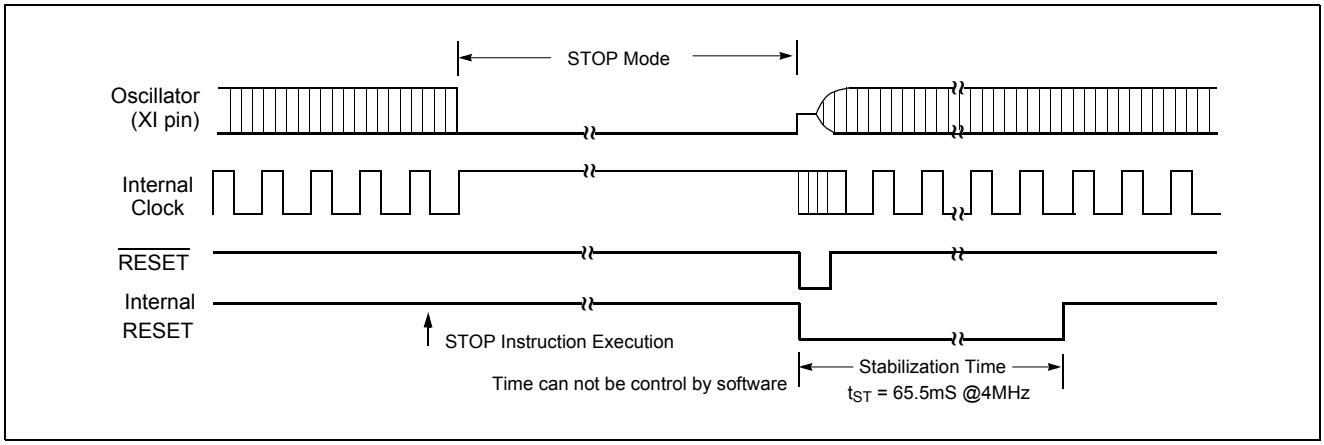


Figure 19-6 Timing of STOP Mode Release by Reset

19.3 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-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. 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 19-8). 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 19-4)

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 19-7 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_H until FF_H. 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 19-8 .

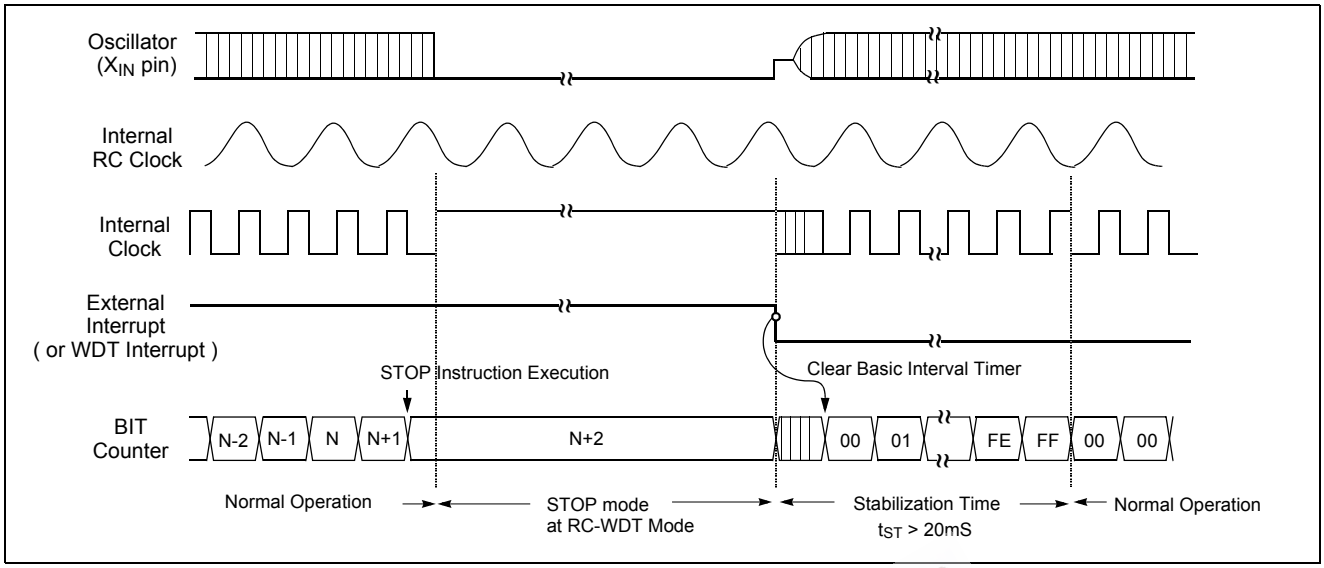


Figure 19-7 Stop Mode Release at Internal RC-WDT Mode by External Interrupt or WDT Interrupt

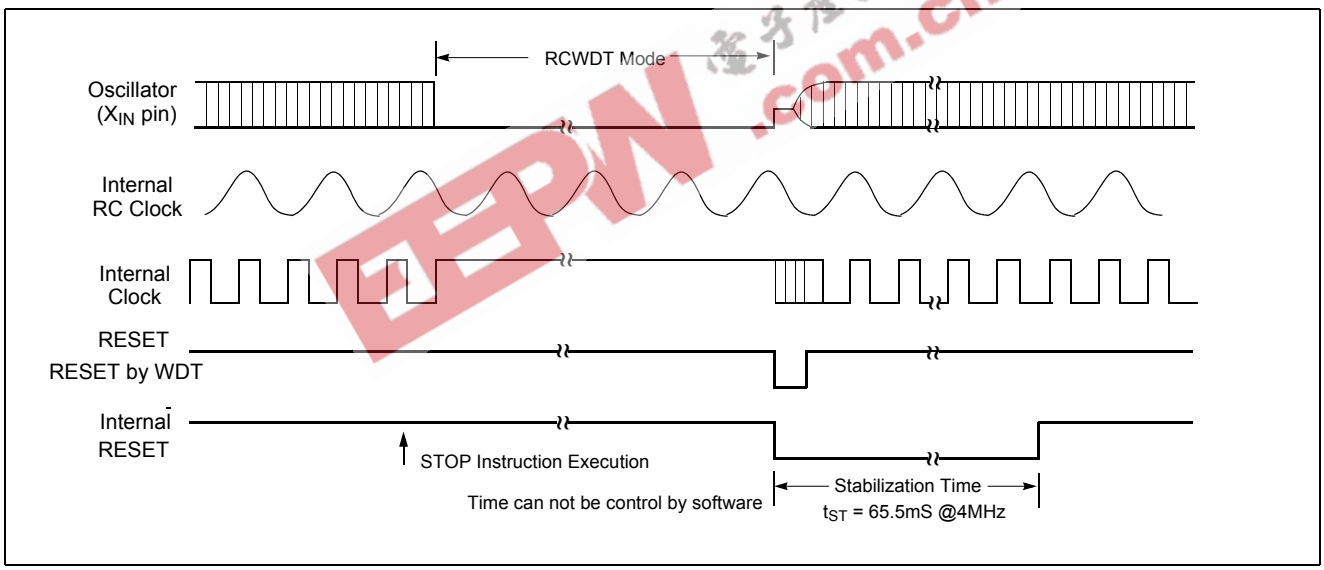


Figure 19-8 Internal RC-WDT Mode Releasing by Reset

19.4 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.

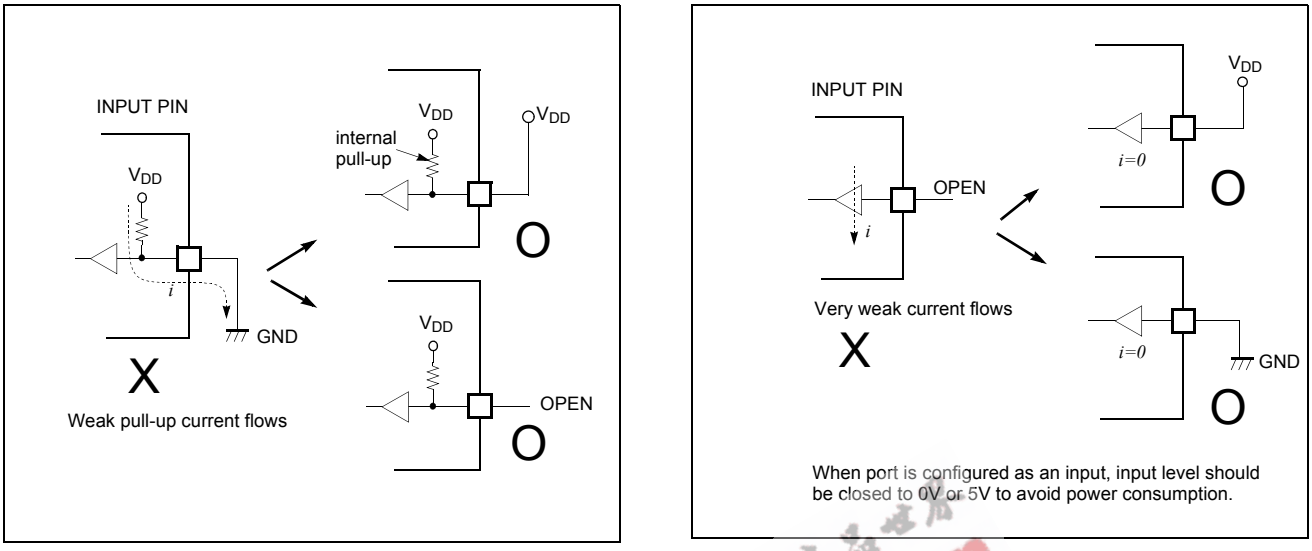


Figure 19-9 Application Example of Unused Input Port

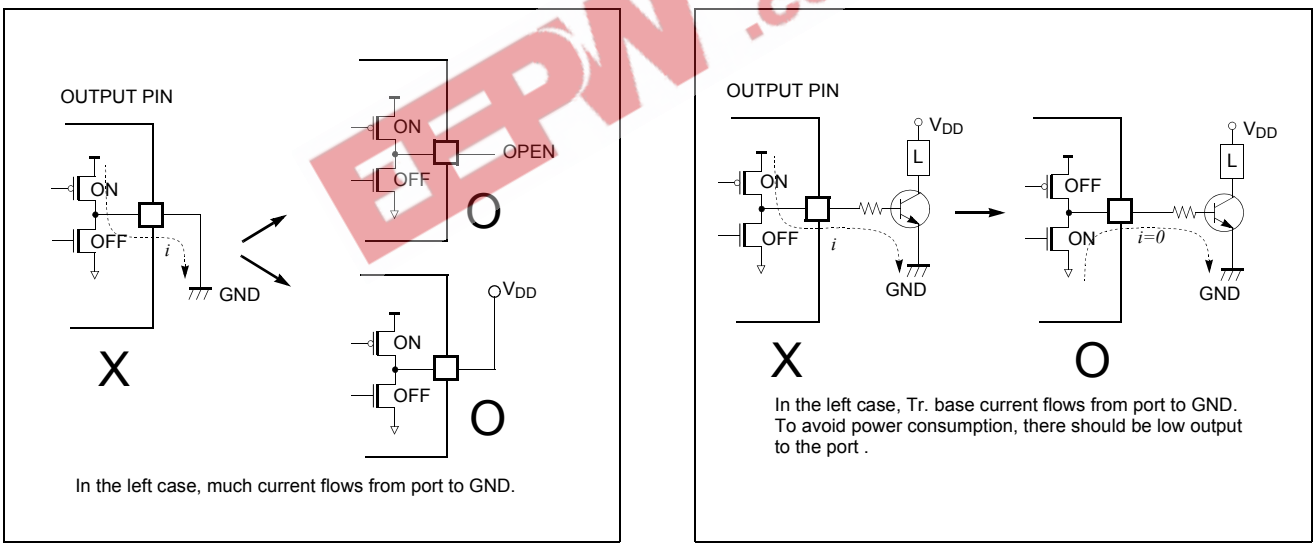


Figure 19-10 Application Example of Unused Output Port

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.

20. RESET

The MC80F0104/0204 supports various kinds of reset as below.

- Power-On Reset (POR)
- $\overline{\text{RESET}}$ (external reset circuitry)
- Watchdog Timer Timeout Reset
- Power-Fail Detection (PFD) Reset
- Address Fail Reset

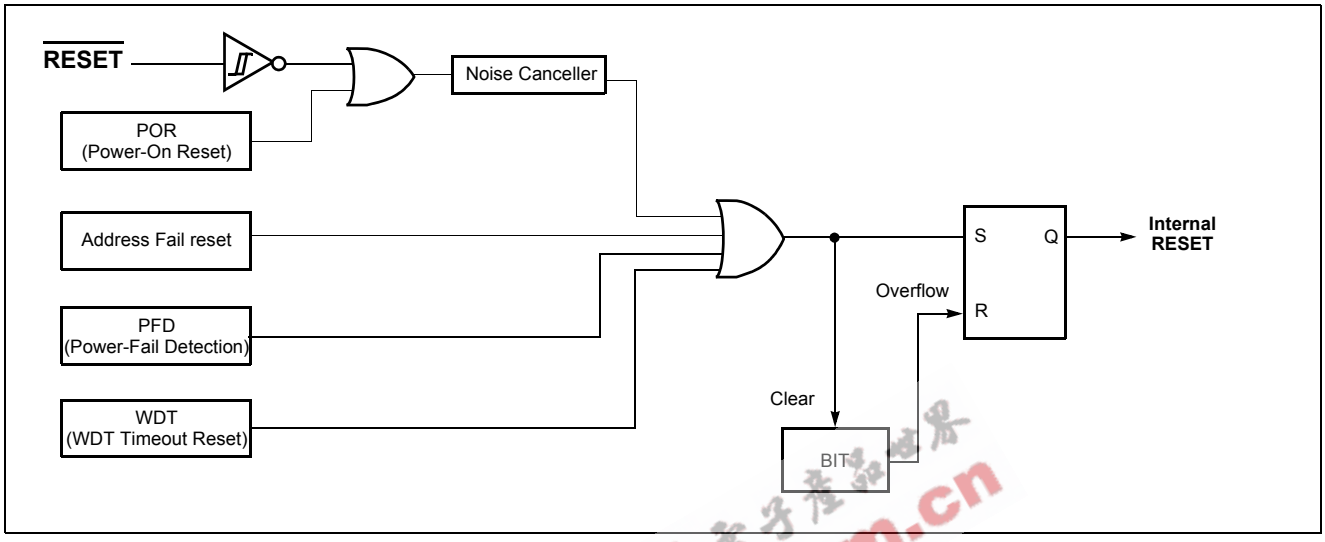


Figure 20-1 RESET Block Diagram

The on-chip POR circuit holds down the device in RESET until V_{DD} 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 $\overline{\text{RESET}}$ pin can be used to normal input port R35 by setting "POR" and "R35EN" bit Configuration Area(20FFH) in the Flash programming. When the device starts normal operation, its operating parmeters (voltage, frequency, temperature...etc) must be met.

.Table 20-1 shows on-chip hardware initialization by reset action.

On-chip Hardware	Initial Value	On-chip Hardware	Initial Value
Program counter (PC)	($FFFF_H$) - ($FFFE_H$)	Peripheral clock	Off
RAM page register (RPR)	0	Watchdog timer	Disable
G-flag (G)	0	Control registers	Refer to Table 8-1 on page 25
Operation mode	Main-frequency clock	Power fail detector	Disable

Table 20-1 Initializing Internal Status by Reset Action

The reset input is the $\overline{\text{RESET}}$ pin, which is the input to a Schmitt Trigger. A reset in accomplished by holding the $\overline{\text{RESET}}$ 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 20-2 .

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

When the $\overline{\text{RESET}}$ pin input goes to high, the reset operation is released and the program execution starts at the vector address stored at addresses $FFFE_H$ - $FFFF_H$.

A connection for simple power-on-reset is shown in Figure 20-1 .

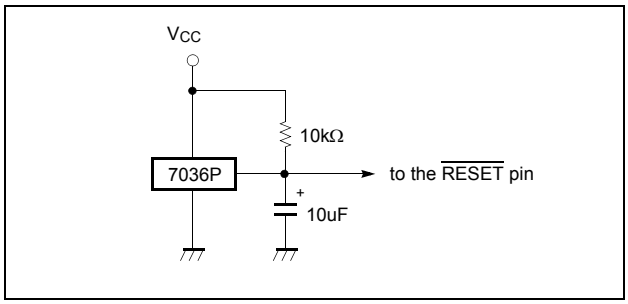


Figure 20-1 Simple Power-on-Reset Circuit

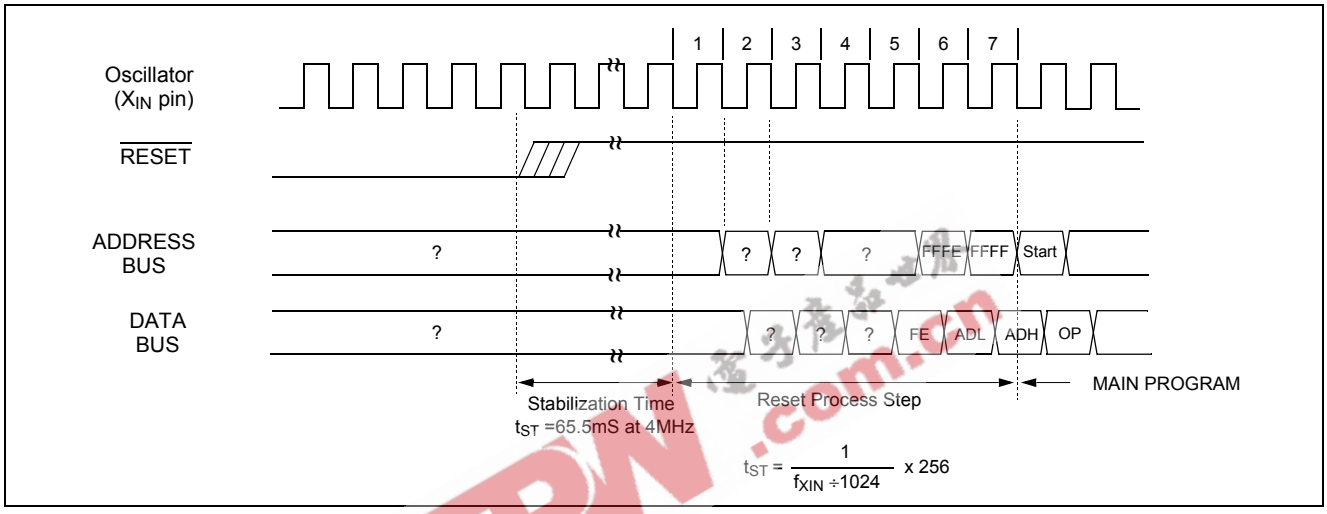


Figure 20-2 Timing Diagram after Reset

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.

21. POWER FAIL PROCESSOR

TheMC80F0104/0204 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. Refer to "Figure 21-1 Power Fail Voltage Detector Register" on page 98.

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.

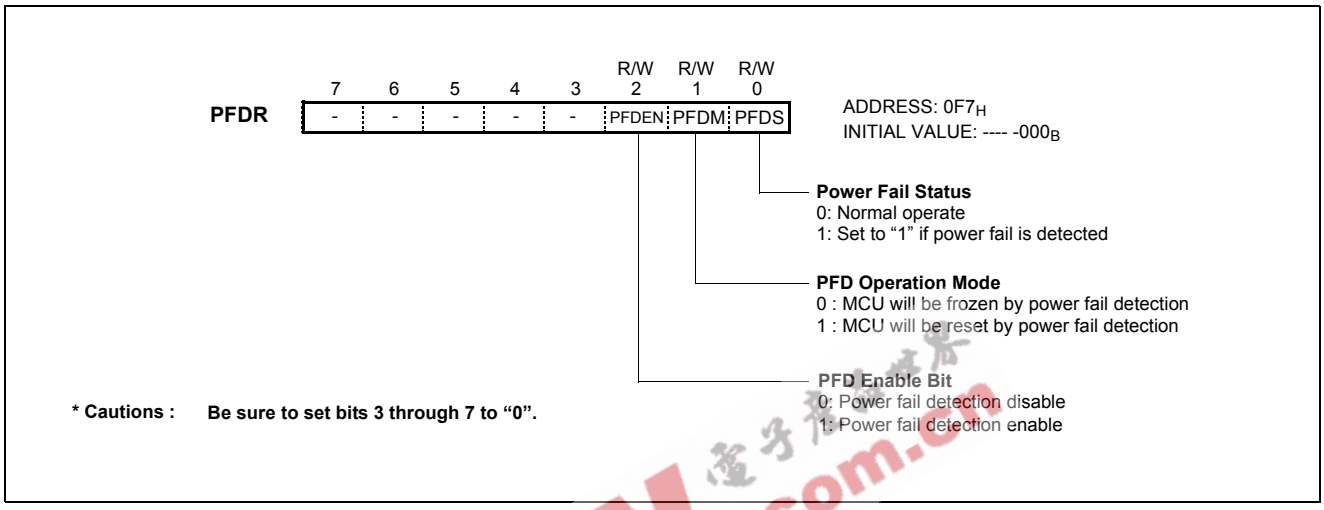


Figure 21-1 Power Fail Voltage Detector Register

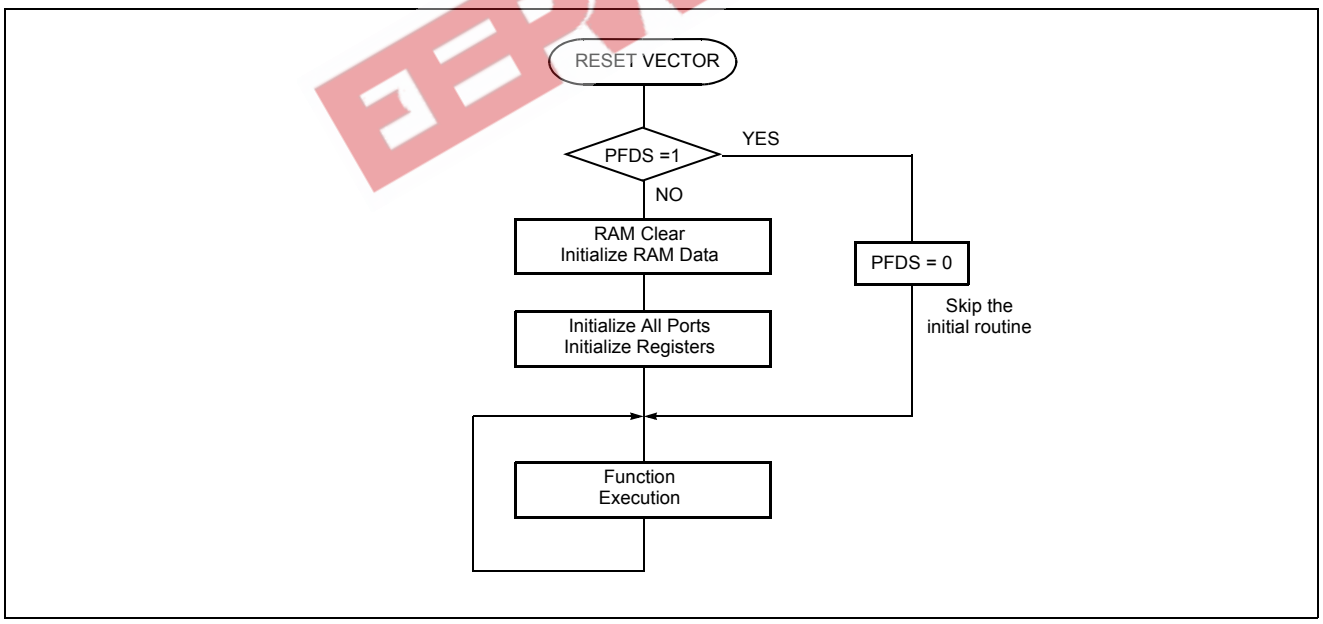


Figure 21-2 Example S/W of Reset flow by Power fail

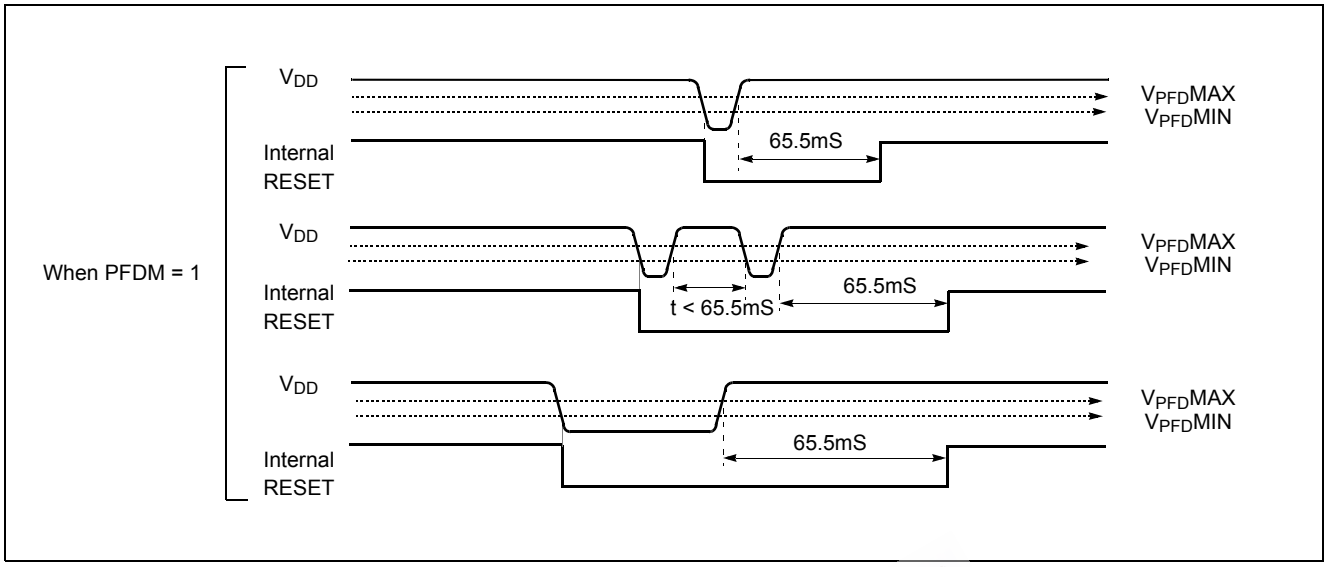


Figure 21-3 Power Fail Processor Situations (at 4MHz operation)

22. COUNTERMEASURE OF NOISE

22.1 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 ($20FF_H$) for the MC80F0204, "ONP" option bits MASK option.

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_{IN}/X_{OUT} is shorted or opened, the main oscillation is stopped except by stop instruction and the low frequency noise is entered.

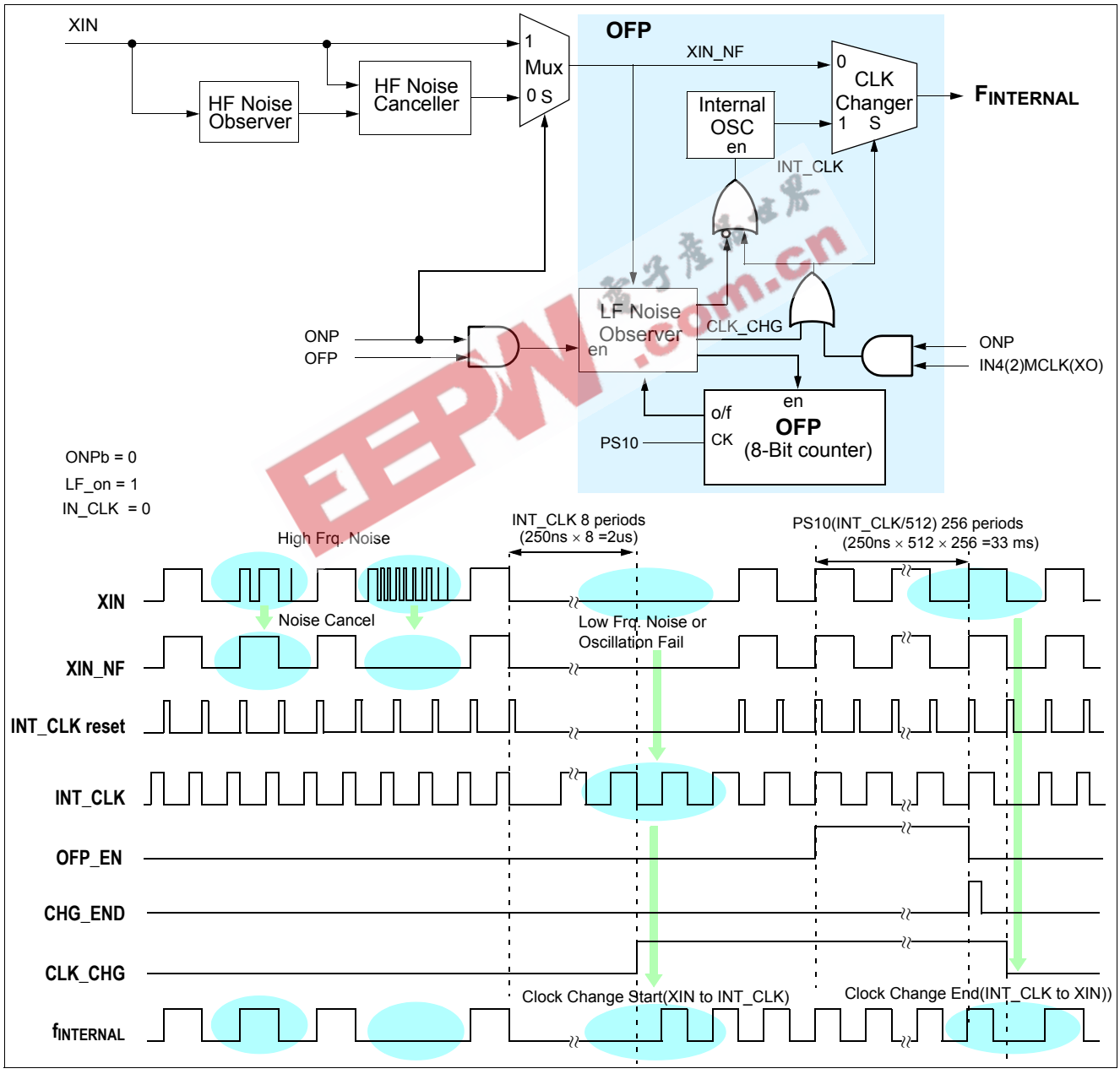


Figure 22-1 Block Diagram of ONP & OFP and Respective Wave Forms

22.2 Oscillation Fail Processor

The oscillation fail processor (OFP) can change the clock source from external to internal oscillator when the oscillation fail occured. This function could be enabled or disabled by the “OFP” bit of the Device Configuration Area (MASK option for MC80C0104/0204).
And this function can recover the external clock source when the external clock is recovered to normal state.

IN4(2)MCLK/CLKXO(XO) Option

The “IN4MCLK(XO)”, “IN2MCLK(XO)” bit of the Device Configuration Area (MASK option for MC80C0104/0204) enables the function to operate the device by using the internal oscillator clock in ONP block as system clock. There is no need to connect the x-tal, resonator, RC and R externally. The user only to connect the X_{IN} pin to V_{DD}. After selecting the this option, the period of internal oscillator clock could be checked by X_{OUT} outputting clock divided the internal oscillator clock by 4.

23. 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.

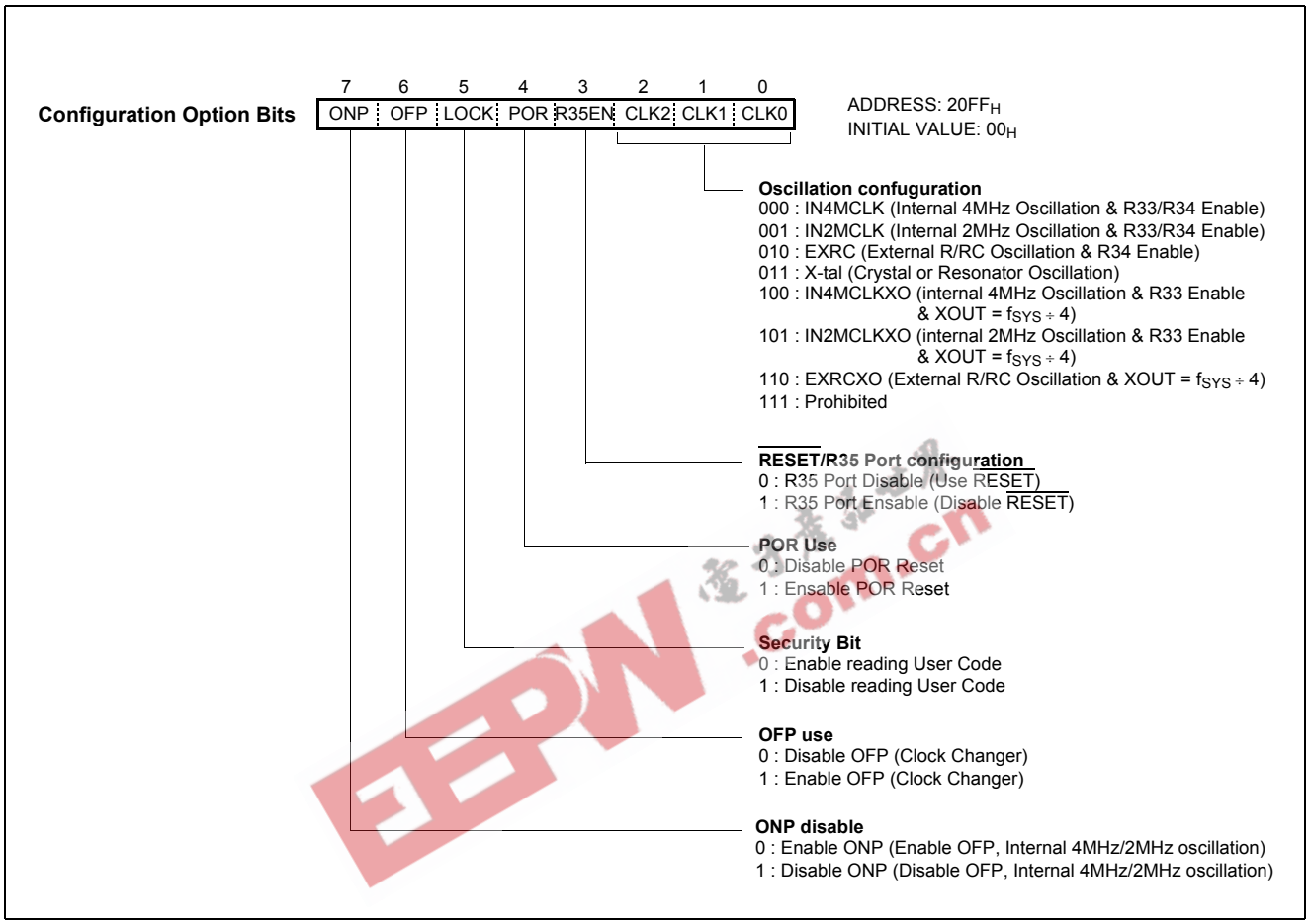


Figure 23-1 Device Configuration Area

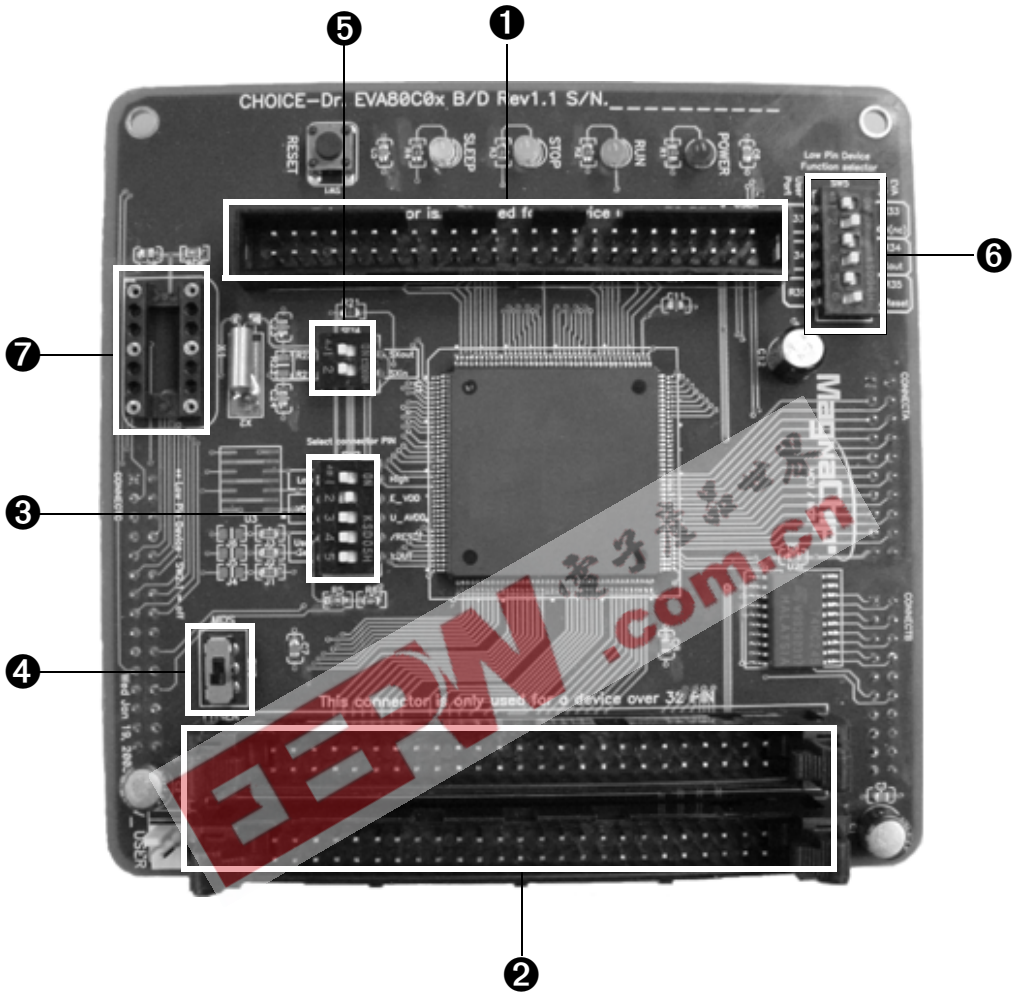
24. MASK Option (MC80C0104/0204)

The MC80C0104/0204 has several MASK option which configures the package type or use of some special features of the device. The Mask option of the MASK order sheet should be checked to select device configuration such as package type, Oscillation selection, oscillation noise protector, oscillation fail protector, internal 4MHz, amount of noise to be cancelled.

	Option	Check	Operation	Remark
MASK Option	Package	16 PDIP	16PDIP type package select	This option is valid only for the MC80C0104
		16SOP	16SOP type package select	
	ONP	Yes	ONP Enable	OSC Noise Protector (ONP) Operation En/Disable Bit
		No	ONP Disable	
	OFP	Yes	Enables Oscillation Fail Processor (ONP clock changer)	Change the Inter clock when oscillation failed
		No	Disables Oscillation Fail Processor (ONP clock changer)	
	POR	Yes	Enables POR	To select Power-on Reset
		No	Disables POR	
	R35EN	Yes	R35 port Enable (Disable $\overline{\text{RESET}}$)	To use R35 port as normal input port
		No	R35 port Disable (Use $\overline{\text{RESET}}$)	
	CLK option	Crystal	Crystal Oscillation	To select Oscillation Type
		EXRC	External R/RC oscillation & R33 Enable	
		IN4MCLK	Internal 4MHz Oscillation & R33/R34 Enable	
		IN2MCLK	Internal 2MHz Oscillation & R33/R34 Enable	
		EXRCXO	External R/RC oscillation & R33 Enable X_{OUT} Pin : System clock ÷ 4	
		IN4MCLKXO	Internal 4MHz Oscillation & R33 Enable X_{OUT} Pin : System clock ÷ 4	
		IN2MCLKXO	Internal 2MHz Oscillation & R33 Enable X_{OUT} Pin : System clock ÷ 4	

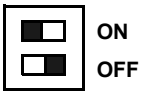
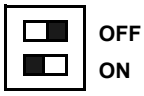
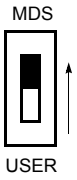
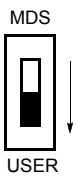
Table 24-1 MASK options

25. Emulator EVA. Board Setting



DIP Switch and VR Setting

Before execute the user program, keep in your mind the below configuration

DIP S/W, VR		Description	ON/OFF Setting
❶	-	This connector is only used for a device over 32 PIN.	For the MC80F0104/0204.
❷	-	This connector is only used for a device under 32 PIN.	For the MC80F0208/0216/0224.
❸ SW2	1	 Eva. select switch	Must be **OFF** position. **ON** : For the MC80F0208/0216/0224. **OFF** : For the MC80F0104/0204.
	2 3	 Use Eva. V_{DD}  Use User's V_{DD} AV_{DD} pin select switch	These switches select the AV_{DD} source. **ON & OFF** : Use Eva. V_{DD} **OFF & ON** : Use User AV_{DD}
	4	This switch select the /Reset source.	Normally **OFF**. EVA. chip can be reset by external user target board. **ON** : Reset is available by either user target system board or Emulator RESET switch. **OFF** : Reset the MCU by Emulator RESET switch. Does not work from user target board.
	5	This switch select the Xout signal on/off.	Normally **OFF**. MCU XOUT pin is disconnected internally in the Emulator. Some circumstance user may connect this circuit. **ON** : Output XOUT signal **OFF** : Disconnect circuit
❹ SW3	1	This switch select Eva. B/D Power supply source.  Use MDS Power  Use User's Power	Normally **MDS**. This switch select Eva. B/D Power supply source.
❺ SW4	1 2	This switch select the R22 or SX_{OUT}. This switch select the R21 or SX_{IN}.	These switchs select the Normal I/O port (off) or Sub-Clock (on). It is reserved for the MC80F0448. **ON** : SX_{OUT}, SX_{IN} **OFF** : R22, R21 Don't care.

DIP S/W, VR		Description	ON/OFF Setting
❻ SW5	1 2 3 4 5 6	(switch 1 & 2) These switches select the R33 or X_{IN} (switch 3 & 4) These switches select the R34 or X_{OUT} (switch 5 & 6) These switches select the R35 or /Reset	This switch select the Normal I/O port (off) or special function select (on). **ON & OFF** : R33,R34,R35 Port selected. **OFF & ON** : X_{IN}, X_{OUT}, /Reset selected.
❼	-	This is External oscillation socket (CAN Type. OSC)	This is for External Clock (CAN Type. OSC).

26. IN-SYSTEM PROGRAMMING (ISP)

26.1 Getting Started / Installation

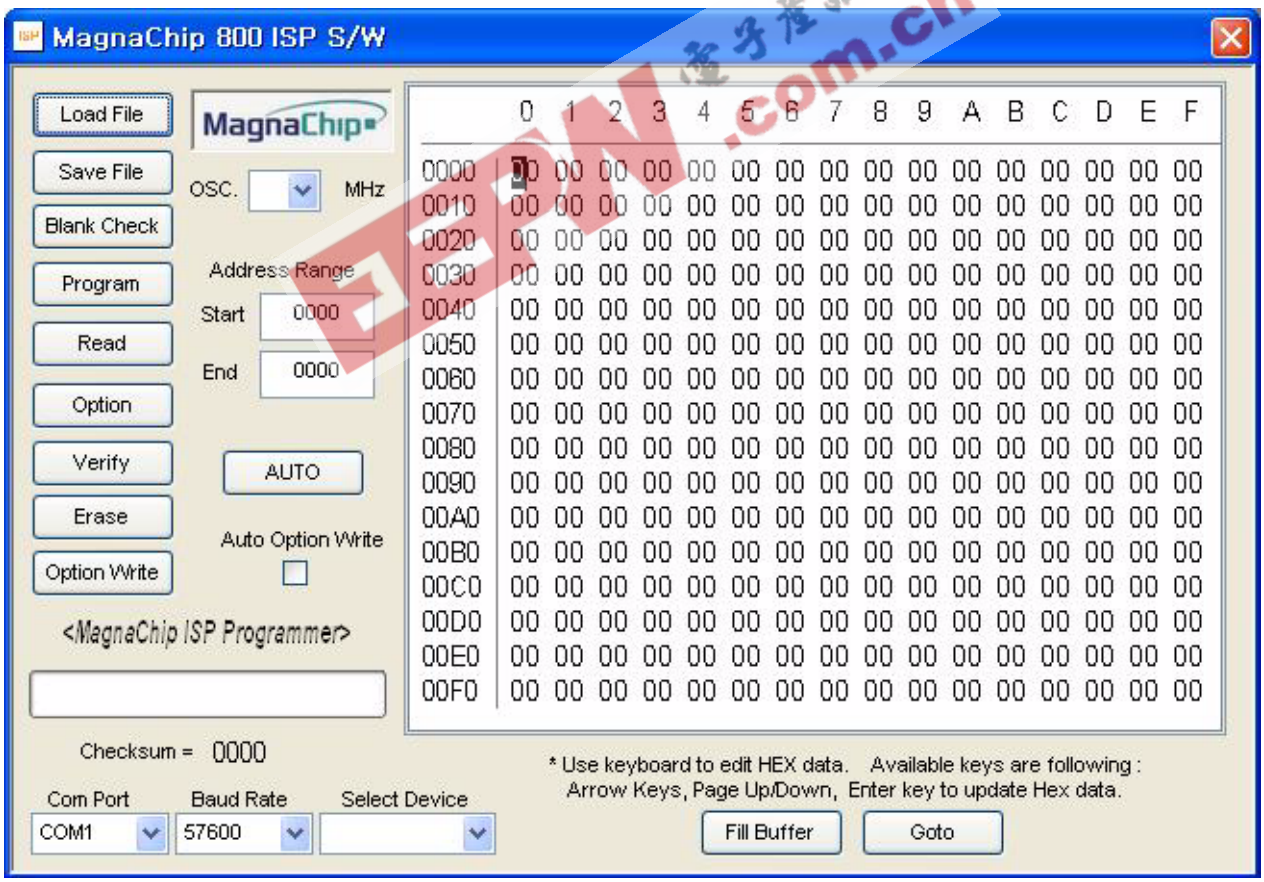
The following section details the procedure for accomplishing the installation procedure.

1. Connect the serial(RS-232C) cable between a target board and the COM port of your PC.
2. Configure the COM port of your PC as following.

Baudrate	115,200
Data bit	8
Parity	No
Stop bit	1
Flow control	No

3. Turn your target B/D power switch ON. Your target B/D must be configured to enter the ISP mode.
4. Run the MagnaChip ISP software.
5. Press the Reset Button in the ISP S/W. If the status windows shows a message as "Connected", all the conditions for ISP are provided.

26.2 Basic ISP S/W Information

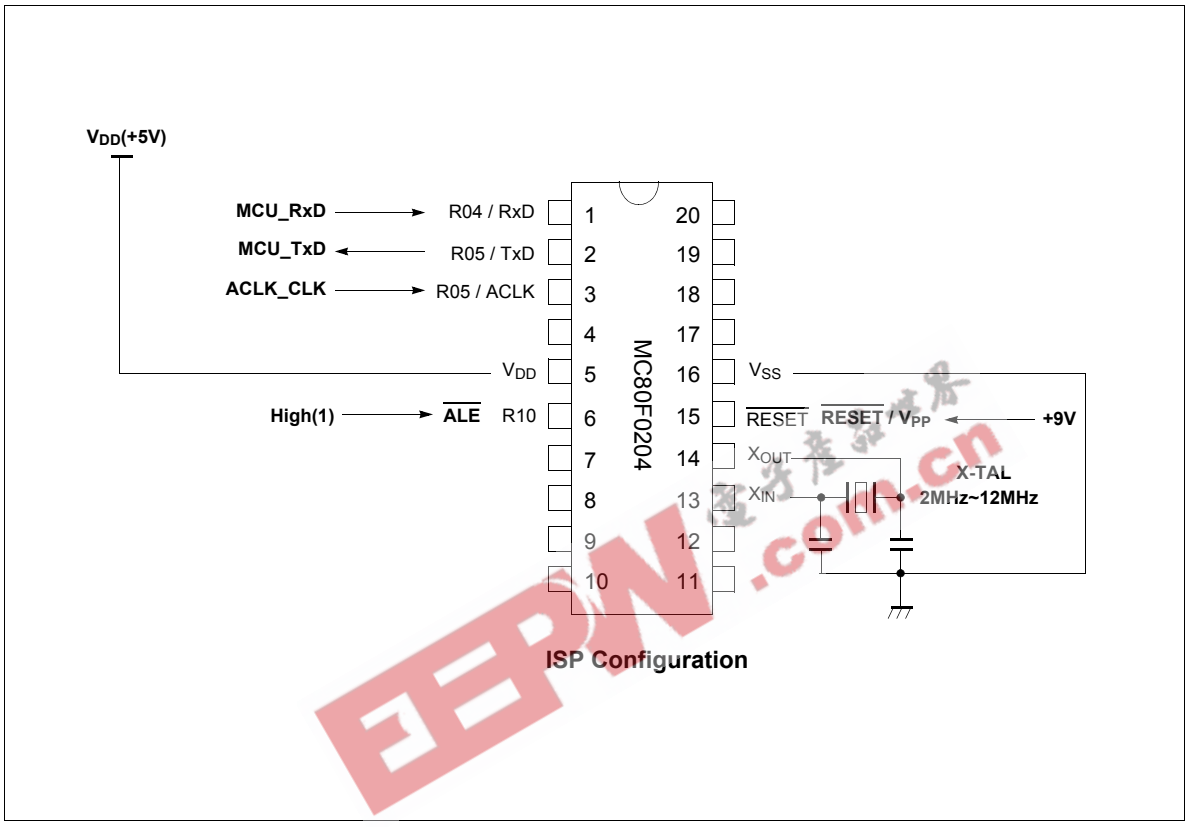


Function	Description
Load HEX File	Load the data from the selected file storage into the memory buffer.
Save HEX File	Save the current data in your memory buffer to a disk storage by using the Intel Motorolla HEX format.
Erase	Erase the data in your target MCU before programming it.
Blank Check	Verify whether or not a device is in an erased or unprogrammed state.
Program	This button enables you to place new data from the memory buffer into the target device.
Read	Read the data in the target MCU into the buffer for examination. The checksum will be displayed on the checksum box.
Verify	Assures that data in the device matches data in the memory buffer. If your device is secured, a verification error is detected.
Option Write	Progam the configuration data of target MCU. The security locking is performed with this button.
Option	Set the configuration data of target MCU. The security locking is set with this button.
AUTO	Erase & Program & Verify.
Auto Option Write	If selected with check mark, the option write is performed after erasure and write.
Edit Buffer	Modify the data in the selected address in your buffer memory
Fill Buffer	Fill the selected area with a data.
Goto	Display the selected page.
OSC. _____ MHz	Enter your target system's oscillator value with discarding below point.
Start _____	Starting address
End _____	End address
Checksum	Display the checksum(Hexdecimal) after reading the target device.
Com Port	Select serial port.
Baud Rate	Select UART baud rate.
Select Device	Select target device.
Page Up Key	Display the previous page of your memory buffer.
Page Down Key	Display the higher page than the current location.

Table 26-1 ISP Function Description

26.3 Hardware Conditions to Enter the ISP Mode

The In-System Programming (ISP) is performed without removing the microcontroller from the system. The In-System Programming (ISP) facility consists of a series of internal hardware resources coupled with internal firmware through the serial port. The In-System Programming (ISP) facility has made in-circuit programming in an embedded application possible with a minimum of additional expense in components and circuit board area. The boot loader can be executed by holding $\overline{ALE}$ high, $\overline{RESET}$ /V_{PP} as +9V, and ACLK with the OSC. 1.8432MHz. The ISP function uses five pins: TxD, RxD, $\overline{ALE}$, ACLK and $\overline{RESET}$/V_{PP}.



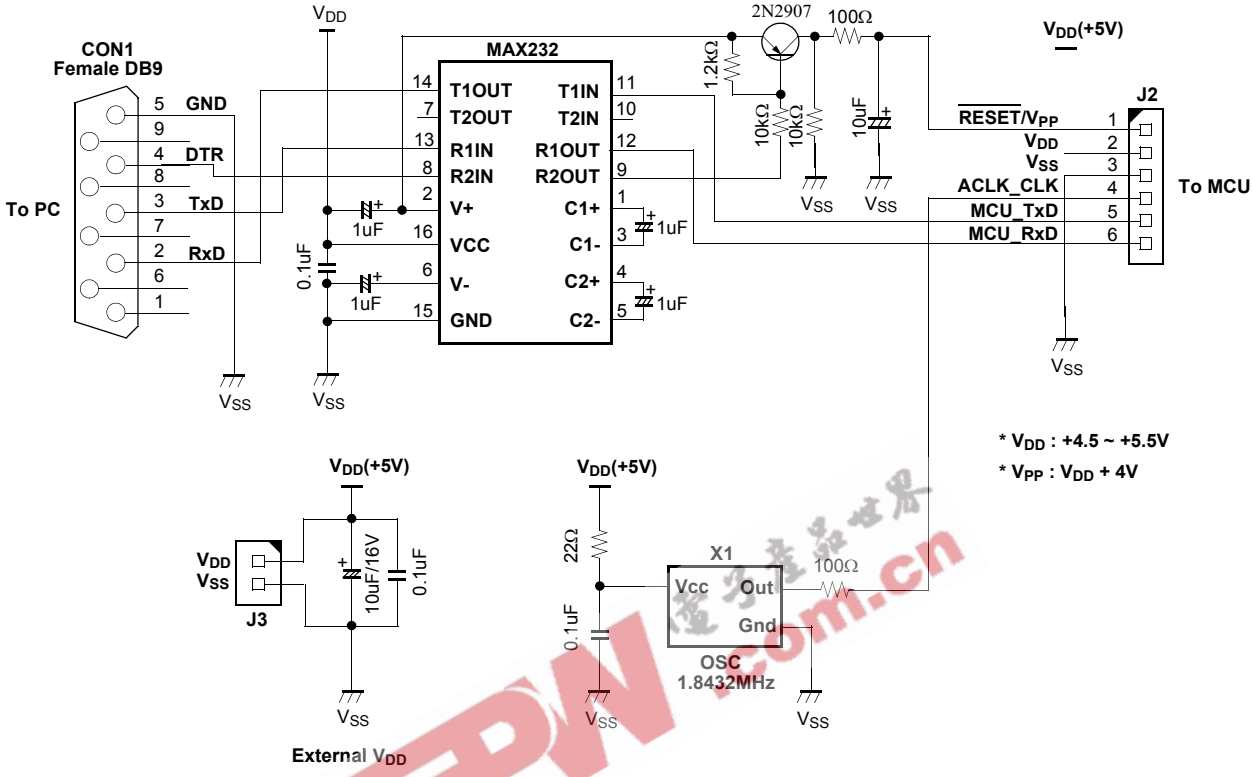
ISP Configuration

Note: *Considerations to implement ISP function in a user target board*

- **The ACLK must be connected to the specifed oscillator.**
- **Connect the +9V to $\overline{RESET}$/Vpp pin directly.**
- **The $\overline{ALE}$ pin must be pulled high.**
- **The main clk must be higher than 2MHz.**

26.4 Reference ISP Circuit Diagram and MagnaChip Supplied ISP Board

The ISP software and hardware circuit diagram are provided at *www.magnachipmcu.com* . To get a ISP B/D, contact to sales department. The following circuit diagram is for reference use..



The ragne of V_{DD} must be from 4.5 to 5.5V and ISP function is not supported under 2MHz system clock. If the user supplied V_{DD} is out of range, the external power is needed instead of the target system V_{DD}. For the ISP operation, power consumption required is minimum 30mA.

Figure 26-1 Reference ISP Circuit Diagram

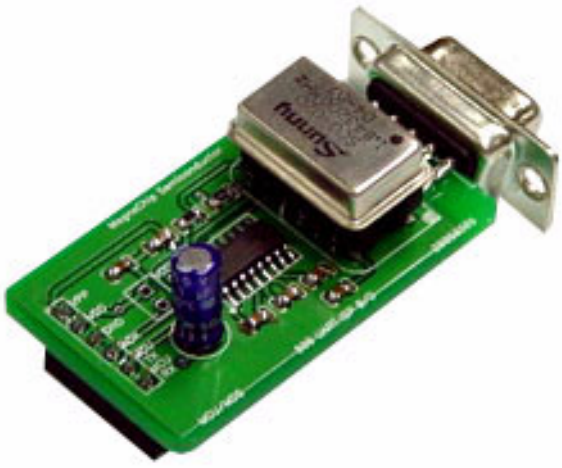


Figure 26-2 MagnaChip supplied ISP Board

APPENDIX

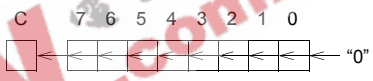
A. INSTRUCTION MAP

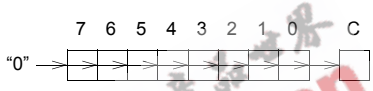
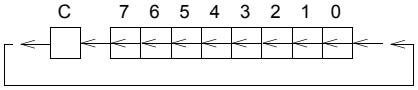
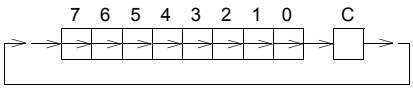
LOW / HIGH	00000 00	00001 01	00010 02	00011 03	00100 04	00101 05	00110 06	00111 07	01000 08	01001 09	01010 0A	01011 0B	01100 0C	01101 0D	01110 0E	01111 0F
000	-	SET1 dp.bit	BBS A.bit,rel	BBS dp.bit,rel	ADC #imm	ADC dp	ADC dp+X	ADC !abs	ASL A	ASL dp	TCALL 0	SETA1 .bit	BIT dp	POP A	PUSH A	BRK
001	CLRC				SBC #imm	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	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	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	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	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	LDA dp	LDA dp+X	LDA !abs	TXA	LDY dp	TCALL 12	LDC LDCB	LDX dp	LDX dp+Y	XCN	DAS
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
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

B. INSTRUCTION SET

1. ARITHMETIC/ LOGIC OPERATION

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	ADC #imm	04	2	2	Add with carry.	
2	ADC dp	05	2	3	A ← (A) + (M) + C	
3	ADC dp + X	06	2	4		
4	ADC !abs	07	3	4		NV--H-ZC
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	
10	AND dp	85	2	3	A ← (A) ∧ (M)	
11	AND dp + X	86	2	4		
12	AND !abs	87	3	4		N-----Z-
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	
18	ASL dp	09	2	4		N-----ZC
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	
22	CMP dp	45	2	3	(A) - (M)	
23	CMP dp + X	46	2	4		
24	CMP !abs	47	3	4		N-----ZC
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	
30	CMPX dp	6C	2	3	(X) - (M)	N-----ZC
31	CMPX !abs	7C	3	4		
32	CMPY #imm	7E	2	2	Compare Y contents with memory contents	
33	CMPY dp	8C	2	3	(Y) - (M)	N-----ZC
34	CMPY !abs	9C	3	4		
35	COM dp	2C	2	4	1'S Complement : (dp) ← ~(dp)	N-----Z-
36	DAA	DF	1	3	Decimal adjust for addition	N-----ZC
37	DAS	CF	1	3	Decimal adjust for subtraction	N-----ZC
38	DEC A	A8	1	2	Decrement	N-----Z-
39	DEC dp	A9	2	4	M ← (M) - 1	
40	DEC dp + X	B9	2	5		N-----Z-
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 ← (A) ⊕ (M)	
46	EOR dp	A5	2	3		
47	EOR dp + X	A6	2	4		
48	EOR !abs	A7	3	4		N-----Z-
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 ← (M) + 1	N-----Z-
54	INC dp	89	2	4		
55	INC dp + X	99	2	5		N-----Z-
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 ← Y × A	N-----Z-
64	OR #imm	64	2	2	Logical OR A ← (A) ∨ (M)	
65	OR dp	65	2	3		
66	OR dp + X	66	2	4		
67	OR !abs	67	3	4		N-----Z-
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 ← (A) - (M) - ~(C)	
81	SBC dp	25	2	3		
82	SBC dp + X	26	2	4		
83	SBC !abs	27	3	4		NV--HZC
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_H	N-----Z-
89	XCN	CE	1	5	Exchange nibbles within the accumulator A_7~A_4 ↔ A_3~A_0	N-----Z-

2. REGISTER / MEMORY OPERATION

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	LDA #imm	C4	2	2	Load accumulator	
2	LDA dp	C5	2	3	A ← (M)	
3	LDA dp + X	C6	2	4		
4	LDA !abs	C7	3	4		
5	LDA !abs + Y	D5	3	5		N-----Z-
6	LDA [dp + X]	D6	2	6		
7	LDA [dp] + Y	D7	2	6		
8	LDA { X }	D4	1	3		
9	LDA { X }+	DB	1	4	X- register auto-increment : A ← (M) , X ← X + 1	
10	LDM dp,#imm	E4	3	5	Load memory with immediate data : (M) ← imm	--------
11	LDX #imm	1E	2	2	Load X-register	
12	LDX dp	CC	2	3	X ← (M)	N-----Z-
13	LDX dp + Y	CD	2	4		
14	LDX !abs	DC	3	4		
15	LDY #imm	3E	2	2	Load Y-register	
16	LDY dp	C9	2	3	Y ← (M)	N-----Z-
17	LDY dp + X	D9	2	4		
18	LDY !abs	D8	3	4		
19	STA dp	E5	2	4	Store accumulator contents in memory	
20	STA dp + X	E6	2	5	(M) ← A	
21	STA !abs	E7	3	5		
22	STA !abs + Y	F5	3	6		--------
23	STA [dp + X]	F6	2	7		
24	STA [dp] + Y	F7	2	7		
25	STA { X }	F4	1	4		
26	STA { X }+	FB	1	4	X- register auto-increment : (M) ← A, X ← X + 1	
27	STX dp	EC	2	4	Store X-register contents in memory	
28	STX dp + Y	ED	2	5	(M) ← X	--------
29	STX !abs	FC	3	5		
30	STY dp	E9	2	4	Store Y-register contents in memory	
31	STY dp + X	F9	2	5	(M) ← Y	--------
32	STY !abs	F8	3	5		
33	TAX	E8	1	2	Transfer accumulator contents to X-register : X ← A	N-----Z-
34	TAY	9F	1	2	Transfer accumulator contents to Y-register : Y ← A	N-----Z-
35	TSPX	AE	1	2	Transfer stack-pointer contents to X-register : X ← sp	N-----Z-
36	TXA	C8	1	2	Transfer X-register contents to accumulator: A ← X	N-----Z-
37	TXSP	8E	1	2	Transfer X-register contents to stack-pointer: sp ← X	N-----Z-
38	TYA	BF	1	2	Transfer Y-register contents to accumulator: A ← Y	N-----Z-
39	XAX	EE	1	4	Exchange X-register contents with accumulator :X ↔ A	--------
40	XAY	DE	1	4	Exchange Y-register contents with accumulator :Y ↔ A	--------
41	XMA dp	BC	2	5	Exchange memory contents with accumulator	
42	XMA dp+X	AD	2	6	(M) ↔ A	N-----Z-
43	XMA {X}	BB	1	5		
44	XYX	FE	1	4	Exchange X-register contents with Y-register : X ↔ Y	--------

3. 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 ← (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) ← (dp+1) (dp) - 1	N-----Z-
4	INCW dp	9D	2	6	Increment memory pair (dp+1) (dp) ← (dp+1) (dp) + 1	N-----Z-
5	LDYA dp	7D	2	5	Load YA YA ← (dp +1) (dp)	N-----Z-
6	STYA dp	DD	2	5	Store YA (dp +1) (dp) ← YA	--------
7	SUBW dp	3D	2	5	16-Bits substact without carry YA ← (YA) - (dp +1) (dp)	NV--H-ZC

4. 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 ← (C) ∧ (M .bit)	-------C
2	AND1B M.bit	8B	3	4	Bit AND C-flag and NOT : C ← (C) ∧ ~(M .bit)	-------C
3	BIT dp	0C	2	4	Bit test A with memory : Z ← (A) ∧ (M) , N ← (M_7) , V ← (M_6)	MM----Z-
4	BIT !abs	1C	3	5		
5	CLR1 dp.bit	y1	2	4	Clear bit : (M.bit) ← “0”	--------
6	CLRA1 A.bit	2B	2	2	Clear A bit : (A.bit)← “0”	--------
7	CLRC	20	1	2	Clear C-flag : C ← “0”	-------0
8	CLRG	40	1	2	Clear G-flag : G ← “0”	--0-----
9	CLRV	80	1	2	Clear V-flag : V ← “0”	-0--0---
10	EOR1 M.bit	AB	3	5	Bit exclusive-OR C-flag : C ← (C) ⊕ (M .bit)	-------C
11	EOR1B M.bit	AB	3	5	Bit exclusive-OR C-flag and NOT : C ← (C) ⊕ ~(M .bit)	-------C
12	LDC M.bit	CB	3	4	Load C-flag : C ← (M .bit)	-------C
13	LDCB M.bit	CB	3	4	Load C-flag with NOT : C ← ~(M .bit)	-------C
14	NOT1 M.bit	4B	3	5	Bit complement : (M .bit) ← ~(M .bit)	--------
15	OR1 M.bit	6B	3	5	Bit OR C-flag : C ← (C) ∨ (M .bit)	-------C
16	OR1B M.bit	6B	3	5	Bit OR C-flag and NOT : C ← (C) ∨ ~(M .bit)	-------C
17	SET1 dp.bit	x1	2	4	Set bit : (M.bit) ← “1”	--------
18	SETA1 A.bit	0B	2	2	Set A bit : (A.bit) ← “1”	--------
19	SETC	A0	1	2	Set C-flag : C ← “1”	-------1
20	SETG	C0	1	2	Set G-flag : G ← “1”	--1-----
21	STC M.bit	EB	3	6	Store C-flag : (M .bit) ← C	--------
22	TCLR1 !abs	5C	3	6	Test and clear bits with A : A - (M) , (M) ← (M) ∧ ~(A)	N-----Z-
23	TSET1 !abs	3C	3	6	Test and set bits with A : A - (M) , (M) ← (M) ∨ (A)	N-----Z-

5. 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_H), sp←sp - 1, M(sp)← (pc_L), sp ←sp - 1, if !abs, pc← abs ; if [dp], pc_L← (dp), pc_H← (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	--------
21	JMP [!abs]	1F	3	5	pc ← jump address	
22	JMP [dp]	3F	2	4		
23	PCALL upage	4F	2	6	U-page call M(sp) ←(pc_H), sp ←sp - 1, M(sp) ← (pc_L), sp ← sp - 1, pc_L ← (upage), pc_H ← ”0FF_H” .	--------
24	TCALL n	nA	1	8	Table call : (sp) ←(pc_H), sp ← sp - 1, M(sp) ← (pc_L),sp ← sp - 1, pc_L ← (Table vector L), pc_H ← (Table vector H)	--------

6. CONTROL OPERATION & etc.

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NVGBHIZC
1	BRK	0F	1	8	Software interrupt : B ← "1", M(sp) ← (pc_H), sp ←sp-1, M(s) ← (pc_L), sp ← sp - 1, M(sp) ← (PSW), sp ← sp -1, pc_L ← ($0FFDE_H$) , pc_H ← ($0FFDF_H$) .	---1-0--
2	DI	60	1	3	Disable interrupts : I ← "0"	-----0--
3	EI	E0	1	3	Enable interrupts : I ← "1"	-----1--
4	NOP	FF	1	2	No operation	--------
5	POP A	0D	1	4	sp ← sp + 1, A ← M(sp)	--------
6	POP X	2D	1	4	sp ← sp + 1, X ← M(sp)	
7	POP Y	4D	1	4	sp ← sp + 1, Y ← M(sp)	
8	POP PSW	6D	1	4	sp ← sp + 1, PSW ← M(sp)	restored
9	PUSH A	0E	1	4	M(sp) ← A , sp ← sp - 1	--------
10	PUSH X	2E	1	4	M(sp) ← X , sp ← sp - 1	
11	PUSH Y	4E	1	4	M(sp) ← Y , sp ← sp - 1	
12	PUSH PSW	6E	1	4	M(sp) ← PSW , sp ← sp - 1	
13	RET	6F	1	5	Return from subroutine sp ← sp +1, pc_L ← M(sp), sp ← sp +1, pc_H ← M(sp)	--------
14	RETI	7F	1	6	Return from interrupt sp ← sp +1, PSW ← M(sp), sp ← sp + 1, pc_L ← M(sp), sp ← sp + 1, pc_H ← M(sp)	restored
15	STOP	EF	1	3	Stop mode (halt CPU, stop oscillator)	--------

MASK ORDER SHEET

MASK ORDER & VERIFICATION SHEET

MC80C0104 [ ] - [ ] [ ] [ ]

Customer should write inside thick line box.

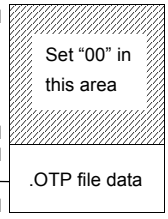
1. Customer Information

Company Name	
Application	
Order Date	YYYY MM DD • •
Tel: Fax: Name & Signature:	

2. Device Information

Package	☐ 16PDIP	☐ 16SOP
POR	☐ Yes	☐ No
R35 Use	☐ Yes	☐ No
ONP Use	☐ Yes	☐ No
OFP Use	☐ Yes	☐ No
CLK	☐ Crystal	
	☐ IN4M	☐ IN4MXO
	☐ IN2M	☐ IN2MXO
	☐ EXRC	☐ EXRCXO
Mask Data	File Name: (.OTP) Check Sum: ( )	

Notice : Unused user ROM area should be filled with "00H"



(Please check mark into ☐)

3. Marking Specification

MagnaChip
MC80C0104x-xxx
YYWW KOREA
O

#1 index mark

4. Delivery Schedule

	Date	Quantity	MagnaChip Confirmation
Customer Sample	YYYY MM DD • •	pcs	
Risk Order	YYYY MM DD • •	pcs	

5. ROM Code Verification

Verification Date: YYYY MM DD • •
Please confirm our verification data.
Check Sum:
Tel: Fax: Name & Signature:

This box is written after "5.Verification".

Approval Date: YYYY MM DD • •
I agree with your verification data and confirm you to make mask set.
Tel: Fax: Name & Signature:

MagnaChip

MASK ORDER & VERIFICATION SHEET

MC80C0204 [ ] - [ ] [ ] [ ]

Customer should write inside thick line box.

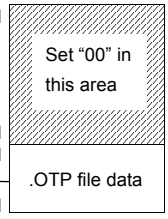
1. Customer Information

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

2. Device Information

Package	☐ 20PDIP	☐ 20SOP
POR	☐ Yes	☐ No
R35 Use	☐ Yes	☐ No
ONP Use	☐ Yes	☐ No
OFP Use	☐ Yes	☐ No
CLK	☐ Crystal	
	☐ IN4M	☐ IN4MXO
	☐ IN2M	☐ IN2MXO
	☐ EXRC	☐ EXRCXO
Mask Data	File Name: (.OTP)	
	Check Sum: ( )	

Notice : Unused user ROM area should be filled with "00H"



(Please check mark into ☐)

3. Marking Specification

MagnaChip
MC80C0204x-xxx
YYWW KOREA
O

#1 index mark

4. Delivery Schedule

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

5. ROM Code Verification

Verification Date:	YYYY MM DD
Please confirm our verification data.	
Check Sum:	
Tel: Fax:	
Name & Signature:	

This box is written after "5.Verification".

Approval Date:	YYYY MM DD
I agree with your verification data and confirm you to make mask set.	
Tel: Fax:	
Name & Signature:	

MagnaChip