

MAGNACHIP SEMICONDUCTOR LTD.
8-BIT SINGLE-CHIP MICROCONTROLLERS

MC80F0208/16/24

MC80C0208/16/24

User's Manual (Ver. 0.2)

Preliminary



REVISION HISTORY

VERSION 0.2 (MAR. 2005) This book

Version 0.2

**Published by
MCU Application Team**

©2005 MagnaChip semiconductor Inc. All right reserved.

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.

| | |
|---|-----------|
| 1. OVERVIEW | 1 |
| Description | 1 |
| Features | 1 |
| Ordering Information | 2 |
| Development Tools | 3 |
| 2. BLOCK DIAGRAM | 4 |
| 3. PIN ASSIGNMENT | 5 |
| 4. PACKAGE DIAGRAM | 6 |
| 5. PIN FUNCTION | 7 |
| Pin Description | 8 |
| 6. PORT STRUCTURES | 10 |
| 7. ELECTRICAL CHARACTERISTICS | 13 |
| Absolute Maximum Ratings | 13 |
| Recommended Operating Conditions | 13 |
| A/D Converter Characteristics | 13 |
| DC Electrical Characteristics | 14 |
| AC Characteristics | 15 |
| Serial Interface Timing Characteristics | 16 |
| Typical Characteristic Curves | 17 |
| 8. MEMORY ORGANIZATION | 19 |
| Registers | 19 |
| Program Memory | 21 |
| Data Memory | 25 |
| Addressing Mode | 31 |
| 9. I/O PORTS | 35 |
| 10. CLOCK GENERATOR | 39 |
| 11. BASIC INTERVAL TIMER | 40 |
| 12. WATCHDOG TIMER | 42 |
| 13. WATCH TIMER | 45 |
| 14. TIMER/EVENT COUNTER | 46 |
| 8-bit Timer / Counter Mode | 50 |
| 16-bit Timer / Counter Mode | 56 |
| 8-bit Compare Output (16-bit) | 57 |
| 8-bit Capture Mode | 58 |
| 16-bit Capture Mode | 62 |
| PWM Mode | 65 |
| 15. ANALOG TO DIGITAL CONVERTER | 68 |
| 16. SERIAL INPUT/OUTPUT (SIO) | 71 |
| Transmission/Receiving Timing | 72 |
| The method of Serial I/O | 74 |
| The Method to Test Correct Transmission | 74 |
| 17. UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART) | 75 |
| UART Serial Interface Functions | 75 |

| | |
|---|------------|
| Serial Interface Configuration | 77 |
| Communication operation | 80 |
| Relationship between main clock and baud rate | 82 |
| 18. BUZZER FUNCTION | 83 |
| 19. INTERRUPTS | 85 |
| Interrupt Sequence | 87 |
| BRK Interrupt | 89 |
| Shared Interrupt Vector | 89 |
| Multi Interrupt | 90 |
| External Interrupt | 91 |
| 20. OPERATION MODE | 93 |
| Operation Mode Switching | 93 |
| 21. POWER SAVING OPERATION | 94 |
| Sleep Mode | 94 |
| Stop Mode | 95 |
| Stop Mode at Internal RC-Oscillated Watchdog Timer Mode | 98 |
| Minimizing Current Consumption | 100 |
| 22. OSCILLATOR CIRCUIT | 102 |
| 23. RESET | 103 |
| 24. POWER FAIL PROCESSOR | 104 |
| 25. FLASH PROGRAMMING | 106 |
| Device Configuration Area | 106 |
| 26. Emulator EVA. Board Setting | 107 |
| 27. IN-SYSTEM PROGRAMMING (ISP) | 110 |
| Getting Started / Installation | 110 |
| Basic ISP S/W Information | 110 |
| Hardware Conditions to Enter the ISP Mode | 111 |
| Reference ISP Circuit diagram | 113 |
| A. INSTRUCTION | i |
| Terminology List | i |
| Instruction Map | ii |
| Instruction Set | iii |
| B. MASK ORDER SHEET | ix |

MC80F0208/16/24

MC80C0208/16/24

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

1. OVERVIEW

1.1 Description

The MC80F0208/16/24 is advanced CMOS 8-bit microcontroller with 8K/16K/24K bytes of FLASH(ROM). This is a powerful microcontroller which provides a highly flexible and cost effective solution to many embedded control applications. This provides the following standard features : 8K/16K/24K bytes of FLASH, 1K bytes of RAM, 8/16-bit timer/counter, watchdog timer, watch 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 8 high current I/O pins with typical 20mA. In addition, the MC80F0208/16/24 supports power saving modes to reduce power consumption.

| Device Name | | FLASH(ROM) Size | RAM | ADC | PWM | I/O PORT | Package |
|-------------|------------|--------------------|--------------|-----------|-----------|----------|---------|
| FLASH | MASK ROM | | | | | | |
| MC80F0208Q | MC80C0208Q | 8KByte | 1024 Byte | 8 channel | 1 channel | 36 port | 44MQFP |
| MC80F0208K | MC80C0208K | | | | | | 42SDIP |
| MC80F0216Q | MC80C0216Q | 16KByte | 1024 Byte | 8 channel | 1 channel | 36 port | 44MQFP |
| MC80F0216K | MC80C0216K | | | | | | 42SDIP |
| MC80F0224Q | MC80C0224Q | 24KByte | 1024 Byte | 8 channel | 1 channel | 36 port | 44MQFP |
| MC80F0224K | MC80C0224K | | | | | | 42SDIP |

1.2 Features

- **8K/16K/24K Bytes On-chip ROM**
- **FLASH Mermory**
 - Endurance : 100 cycles
 - Data Retention : 10 years
- **1024 Bytes On-chip Data RAM (Included stack memory)**
- **Minimum Instruction Execution Time**
 - 333ns at 12MHz (NOP instruction)
- **36 I/O Ports**
- **One 8-bit Basic Interval Timer**
- **Four 8-bit and one 16-bit Timer/Event counter (or three 16-bit Timer/Event counter)**
- **One Watchdog timer**
- **One Watch timer**
- **One 10-bit PWM**
- **8 channel 10-bit A/D converter**
- **Three 8-bit Serial Communication Interface**
 - One Serial I/O and two UART
- **One Buzzer Driving port**
 - 488Hz ~ 250kHz@4MHz
- **Four External Interrupt input ports**
- **Fifteen Interrupt sources**
 - Basic Interval Timer(1)
 - External input(4)
 - Timer/Event counter(5)
 - ADC(1)
 - Serial Interface(1), UART(2)
 - WDT and Watch Timer(1)
- **Built in Noise Immunity Circuit**
 - Noise filter
 - 3-level Power fail detector [3.0V, 2.7V, 2.4V]
- **Power Down Mode**
 - Stop, Sleep mode
- **Operating Voltage Range**
 - 2.7V ~ 5.5V (@ 8MHz)
 - 4.5V ~ 5.5V (@ 12MHz)

- **Operating Frequency Range**
- 0.4 ~ 12MHz
- **44MQFP, 42SDIP type**
- **Operating Temperature : -40°C ~ 85°C**
- **Oscillator Type**
- Crystal, Ceramic resonator, External clock

1.3 Ordering Information

| ROM Type | Device name | ROM Size | RAM size | Package |
|---------------|--------------------------|------------------------------------|------------|------------------|
| Mask version | MC80C0208Q MC80C0208K | 8K bytes 8K bytes | 1024 bytes | 44MQFP 42SDIP |
| | MC80C0216Q MC80C0216K | 16K bytes 16K bytes | 1024 bytes | 44MQFP 42SDIP |
| | MC80C0224Q MC80C0224K | 24K bytes 24K bytes | 1024 bytes | 44MQFP 42SDIP |
| FLASH version | MC80F0208Q MC80F0208K | 8K bytes FLASH 8K bytes FLASH | 1024 bytes | 44MQFP 42SDIP |
| | MC80F0216Q MC80F0216K | 16K bytes FLASH 16K bytes FLASH | 1024 bytes | 44MQFP 42SDIP |
| | MC80F0224Q MC80F0224K | 24K bytes FLASH 24K bytes FLASH | 1024 bytes | 44MQFP 42SDIP |

Table 1-1 Ordering Information of MC80F0208/16/24 & MC80C0208/16/24

1.4 Development Tools

The MC80F0208/16/24 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 | <ul style="list-style-type: none"> - MS-Windows based assembler - MS-Windows based Debugger - MC800 C compiler |
| Hardware (Emulator) | <ul style="list-style-type: none"> - CHOICE-Dr. - CHOICE-Dr. EVA80C0x B/D |
| FLASH Writer | <ul style="list-style-type: none"> - CHOICE - SIGMA I/II(Single writer) - PGM Plus I/II/III(Single writer) - Standalone GANG4 I/II(Gang writer) |



Choice-Dr. (Emulator)

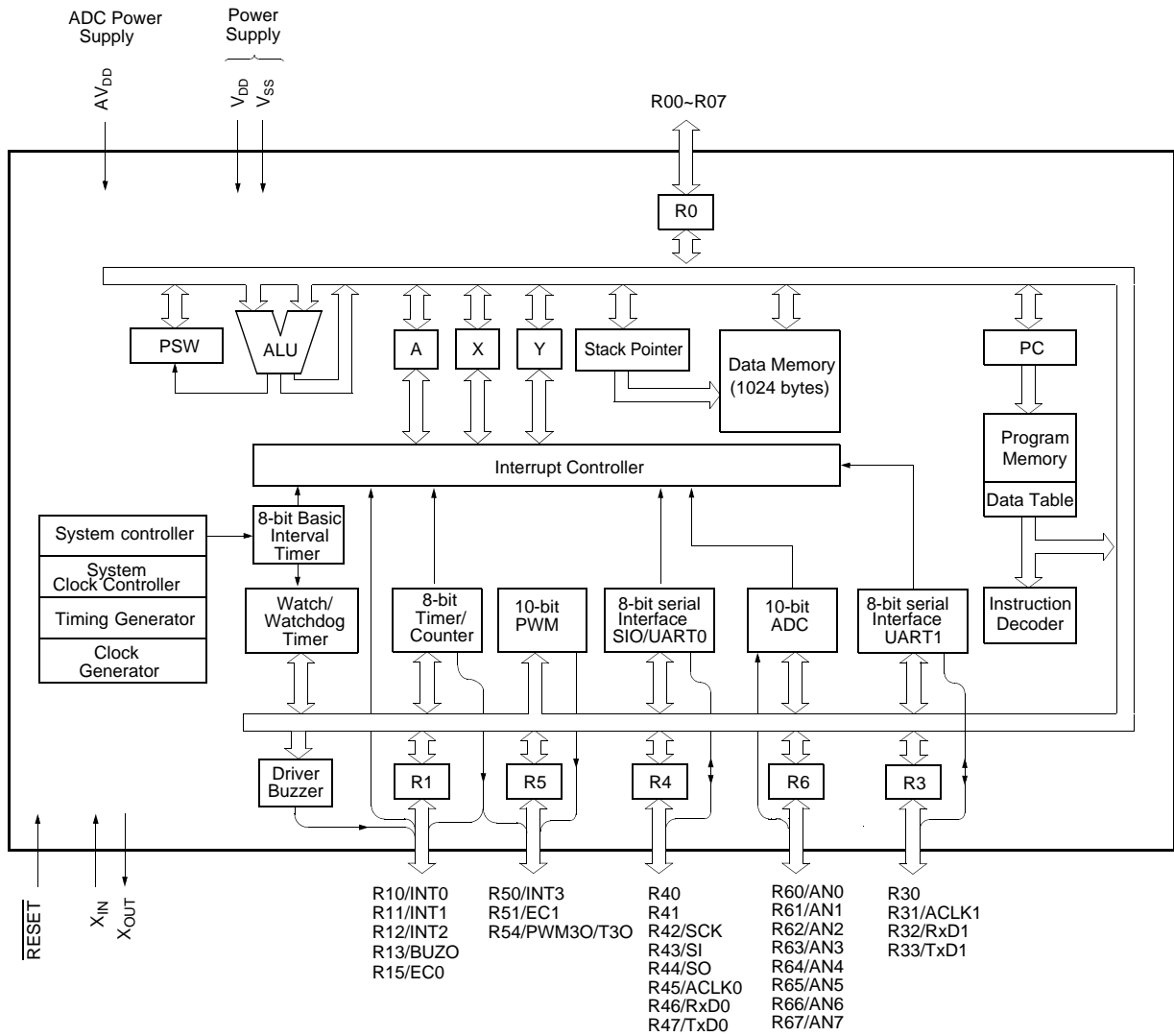


PGMplus III (Single Writer)



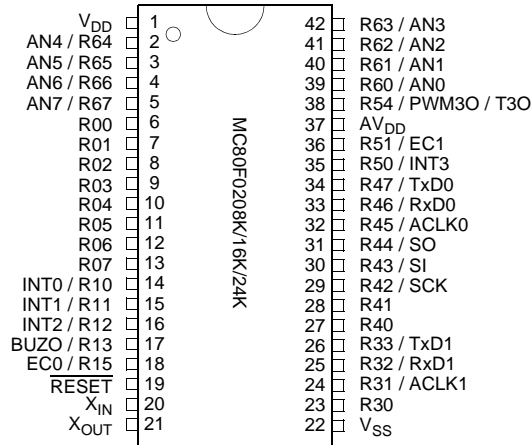
Standalone Gang4 II (Gang Writer)

2. BLOCK DIAGRAM

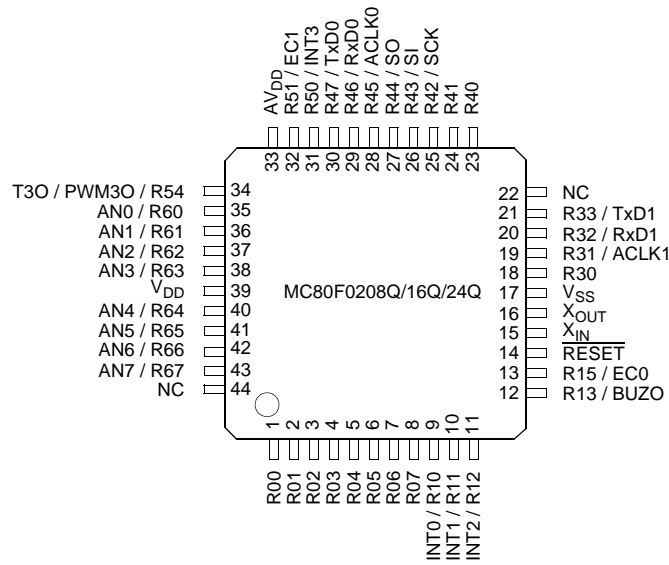


3. PIN ASSIGNMENT

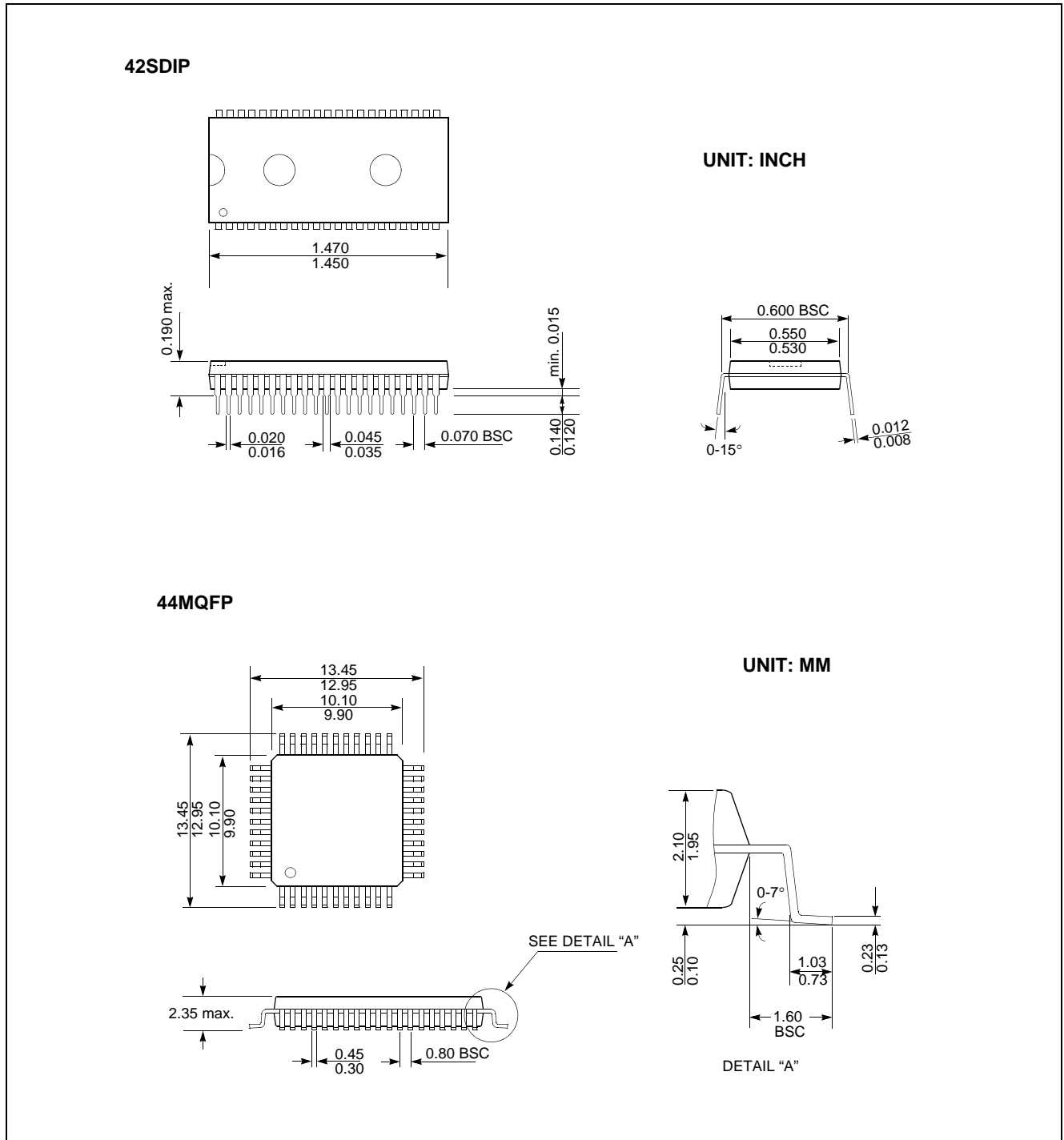
42SDIP
(Top View)



44MQFP
(Top View)



4. PACKAGE DIAGRAM



5. PIN FUNCTION

V_{DD}: Supply voltage.

V_{SS}: Circuit ground.

AV_{DD}: Supply voltage to the ladder resistor of ADC circuit.

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. R0 pins with 1 or 0 written to the R0 Port Direction Register R0IO can be used as outputs or inputs. The internal pull-up resistor can be connected by using the pull-up selection register 0 (PU0).

R10~R13, R15: R1 is an 5-bit CMOS bidirectional I/O port. R1 pins with 1 or 0 written to the R1 Port Direction Register R1IO can be used as outputs or inputs. The internal pull-up resistor can be connected by using the pull-up selection register 1 (PU1). In addition, R1 serves the functions of the various following special features such as INT0 (External interrupt 0), INT1 (External interrupt 1), INT2 (External interrupt 2), BUZO (Buzzer driver output), EC0 (Event counter input 0).

R30~R33: R3 is an 4-bit CMOS bidirectional I/O port. R3 pins with 1 or 0 written to the R3 Port Direction Register R3IO can be used as outputs or inputs. R3 operates as the high current output

port with typical 20mA at low level output.

In addition, R3 serves the functions of the following special features such as ACLK1 (UART1 Asynchronous serial clock input), RxD1 (UART1 data input), TxD1 (UART1 data output).

R40~R47: R4 is an 8-bit CMOS bidirectional I/O port. R4 pins with 1 or 0 written to the R4 Port Direction Register R4IO can be used as outputs or inputs. The internal pull-up resistor can be connected by using the pull-up selection register 4 (PU4).

In addition, R4 serves the functions of the various following special features such as SCK (Serial clock), SI (Serial data input), SO (Serial data output), ACLK0 (UART1 Asynchronous serial clock input), RxD0 (UART0 data input), TxD0 (UART0 data output).

R50, R51, R54: R5 is an 3-bit CMOS bidirectional I/O port. R5 pins with 1 or 0 written to the R5 Port Direction Register R5IO can be used as outputs or inputs.

In addition, R5 serves the functions of the various following special features such as INT3 (External interrupt 3), EC1 (Event counter input 1), PWM3O (PWM output 3)/T3O (Timer3 Compare output).

R60~R67: R6 is an 8-bit CMOS bidirectional I/O port. R6 pins with 1 or 0 written to the R6 Port Direction Register R6IO can be used as outputs or inputs.

In addition, R6 serves the functions of the ADC analog input port AN[7:0].

5.1 Pin Description

5.1.1 Normal Function Pin Description

| PIN NAME | In/Out | Function | Initial state | Alternate Function |
|------------------|--------|---|---------------|--------------------|
| R00~R07 | I/O | Port0 8-bit I/O port. Can be set in input or output mode in 1-bit units. Internal pull-up resistor PU0 can be used via software. | Input | - |
| R10 | I/O | Port 1. 5-bit I/O port. Can be set in input or output mode in 1-bit units. Internal pull-up resistor PU1 can be used via software. | Input | INT0 |
| R11 | | | | INT1 |
| R12 | | | | INT2 |
| R13 | | | | BUZO |
| R15 | | | | EC0 |
| R30 | I/O | Port 3. 4-bit I/O port. Can be set in input or output mode in 1-bit units. | Input | - |
| R31 | | | | ACLK1 |
| R32 | | | | RxD1 |
| R33 | | | | TxD1 |
| R40 | I/O | Port 4. 8-bit I/O port. Can be set in input or output mode in 1-bit units. Internal pull-up resistor PU4 can be used via software. | Input | - |
| R41 | | | | - |
| R42 | | | | SCK |
| R43 | | | | SI |
| R44 | | | | SO |
| R45 | | | | ACLK0 |
| R46 | | | | RxD0 |
| R47 | | | | TxD0 |
| R50 | I/O | Port 5. 3-bit I/O port. Can be set in input or output mode in 1-bit units. | Input | INT3 |
| R51 | | | | EC1 |
| R54 | | | | PWM3O/T3O |
| R60~R67 | I/O | Port 6. 8-bit I/O port. Can be set in input or output mode in 1-bit units. | Input | AN0~AN7 |
| RESET | I | System reset input. | Input | - |
| X _{IN} | I | Crystal connection for main system clock oscillation. | Input | - |
| X _{OUT} | O | | Output | - |
| AV _{DD} | - | Analog power/reference voltage input to A/D converter. Set the same potential as VDD. | - | - |
| V _{DD} | - | Positive power supply. | - | - |
| V _{SS} | - | Ground potential. | - | - |

Table 5-1 Normal Function Pin Description

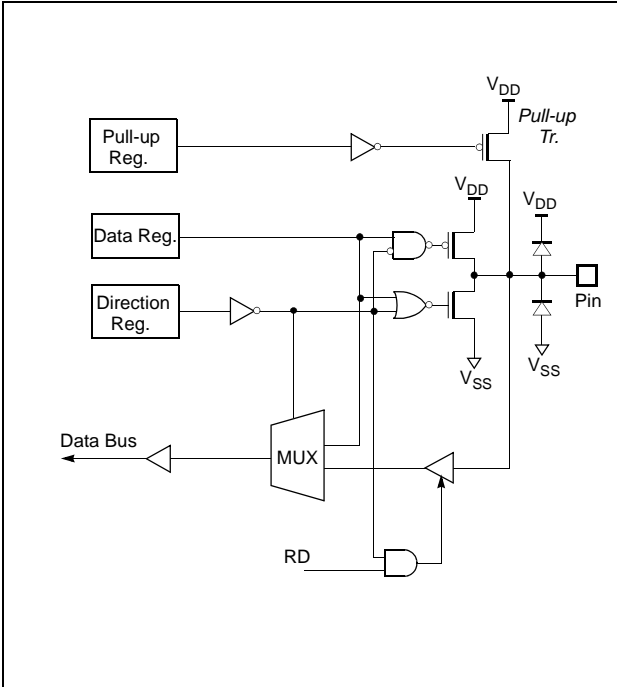
5.1.2 Alternate Function Pin Description

| PIN NAME | In/Out | Function | Initial state | Shared Pin |
|----------|--------|---|---------------|------------|
| INT0 | I | Valid edges(rising, falling, or both rising and falling) can be specified. External Interrupt request Input. | Input | R10 |
| INT1 | | | | R11 |
| INT2 | | | | R12 |
| INT3 | | | | R50 |
| BUZO | O | Buzzer Output | Input | R13 |
| EC0 | I | Timer0 Event Counter Input | Input | R15 |
| EC1 | I | Timer2 Event Counter Input | Input | R51 |
| SCK | I/O | Serial clock input/output of serial interface. | Input | R42 |
| SI | I | Serial data input of serial interface. | Input | R43 |
| SO | O | Serial data output of serial interface. | Input | R44 |
| ACLK0 | I | Asynchronous serial interface serial clock input. | Input | R45 |
| RxD0 | I | Asynchronous serial interface serial data input. | Input | R46 |
| TxD0 | O | Asynchronous serial interface serial data output. | Input | R47 |
| ACLK1 | I | Asynchronous serial interface serial clock input2. | Input | R31 |
| RxD1 | I | Asynchronous serial interface serial data input2. | Input | R32 |
| TxD1 | O | Asynchronous serial interface serial data output2. | Input | R33 |
| PWM3O | O | Timer3 PWM Output | Output | R54 |
| T3O | | Timer3 Compare Output | | |
| AN0~AN7 | I | Analog input Channel 0 ~ 7 for A/D converter. | Input | R60~R67 |

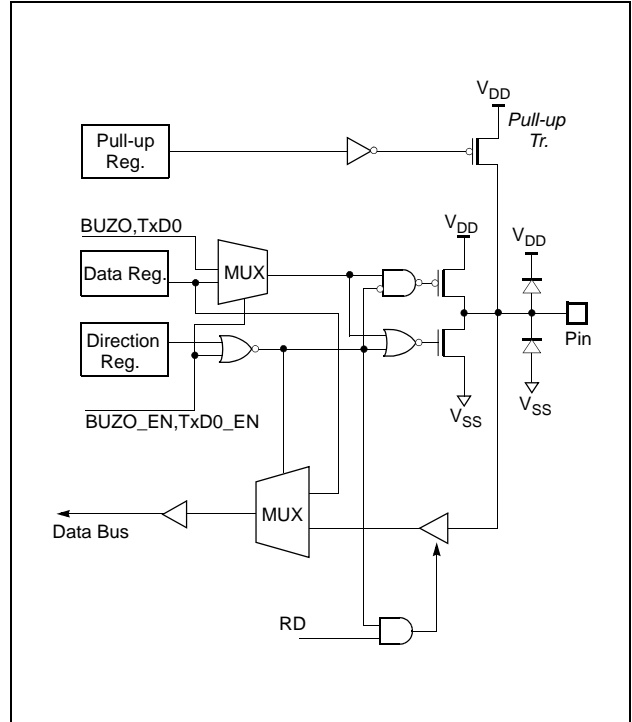
Table 5-2 Alternate Function Pin Description

6. PORT STRUCTURES

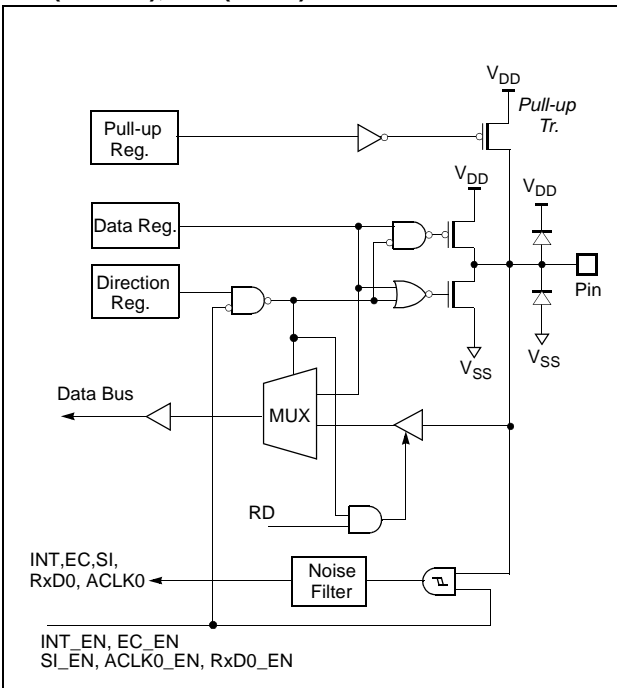
R00~R07, R40, R41



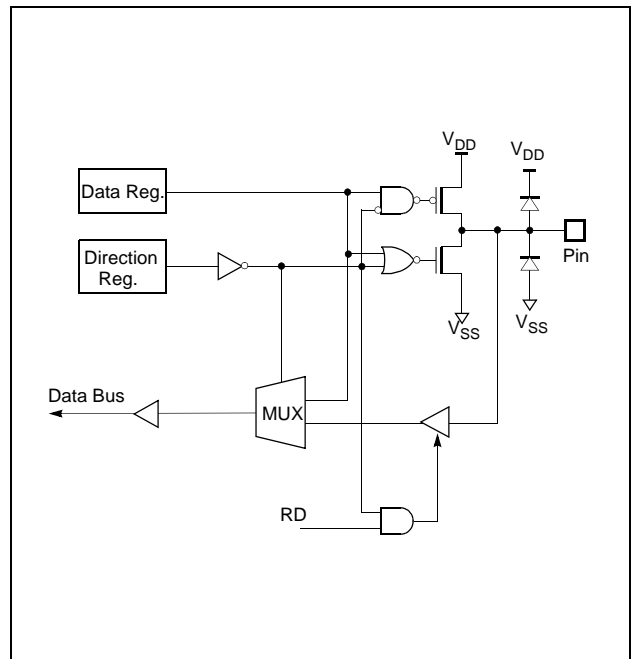
R13(BUZO), R47(TxD0)



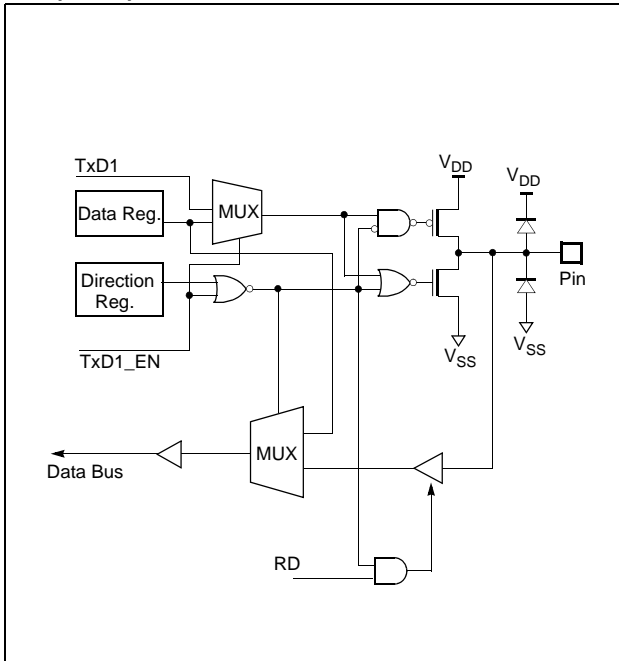
R10(INT0)~ R12(INT2), R15(EC0), R43(SI), R45(ACLK0), R46(RxD0)



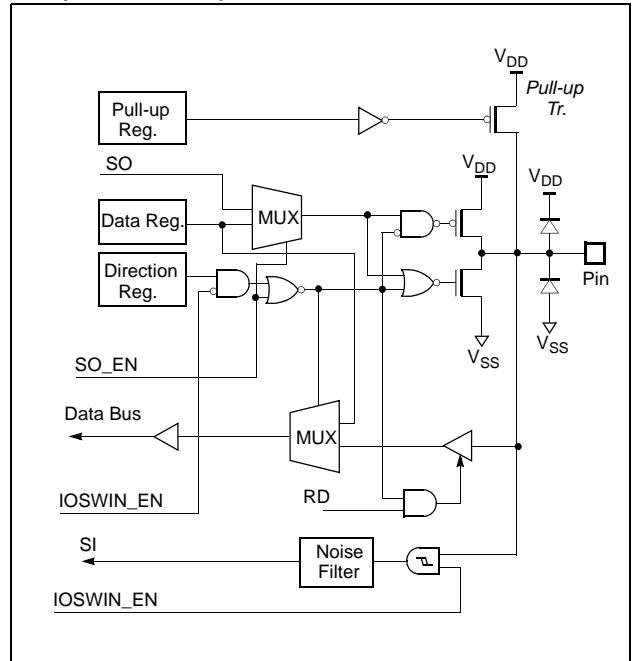
R30



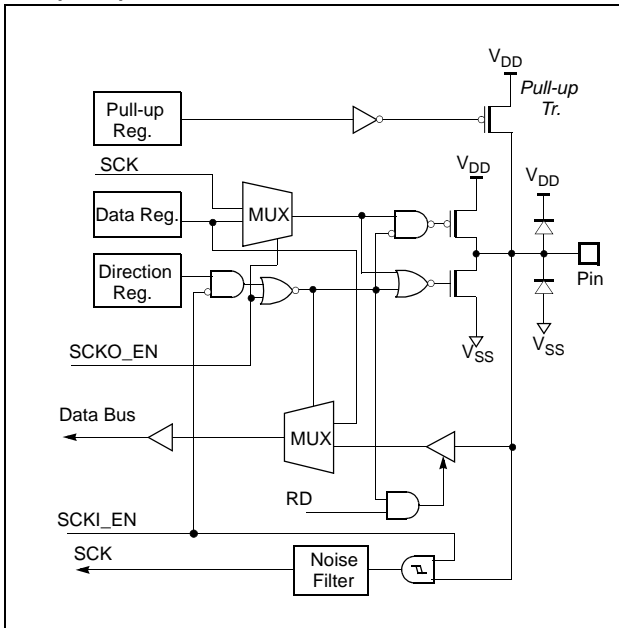
R33(TxD1)



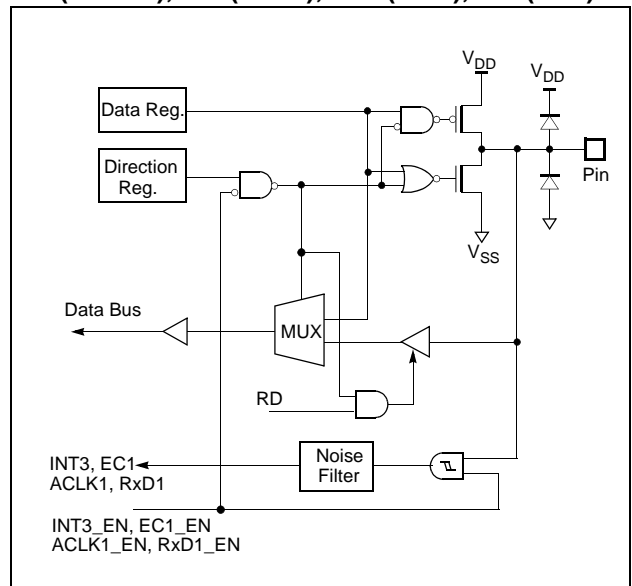
R44(SO, IOSWIN)



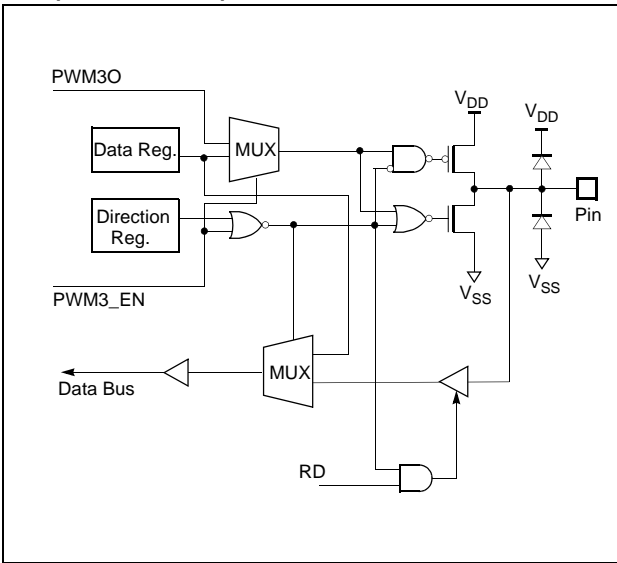
R42(SCK)



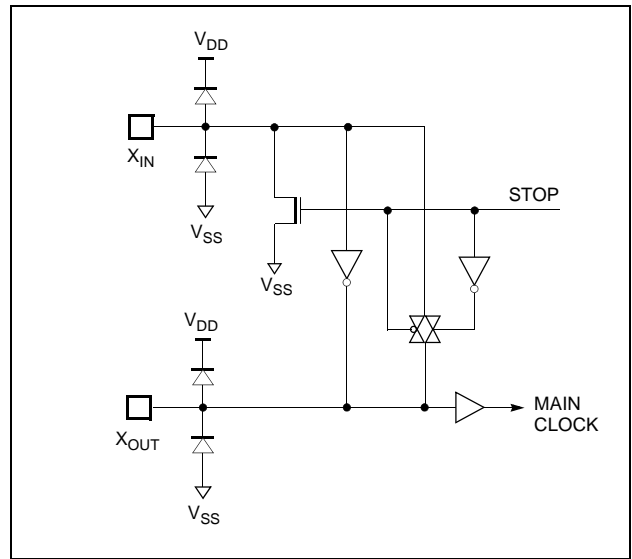
R31(ACLK1), R32(RxD1), R50(INT3), R51(EC1)



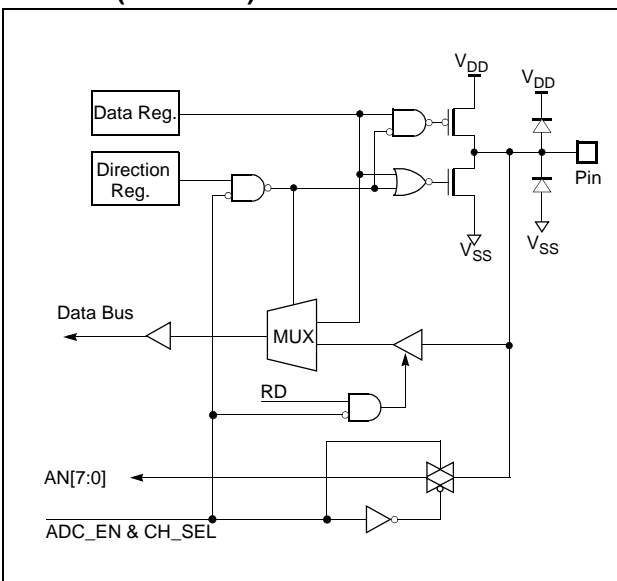
R54(PWM3O/T3O)



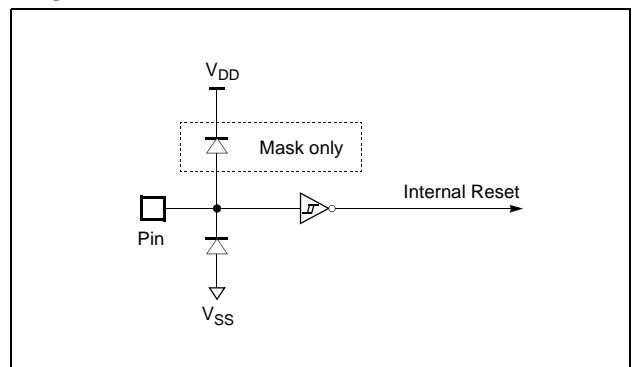
X_{IN}, X_{OUT}



R60~R67(AN0~AN7)



RESET



7. ELECTRICAL CHARACTERISTICS

7.1 Absolute Maximum Ratings

| Parameter | Symbol | Rating | Unit | Note |
|-------------------------|-----------------|----------------------------------|------|---|
| Supply Voltage | V_{DD} | -0.3 ~ +6.5 | V | - |
| | AV_{DD} | $V_{DD} - 0.3 \sim V_{DD} + 0.3$ | V | - |
| Normal Voltage Pin | V_I | -0.3 ~ $V_{DD} + 0.3$ | V | Voltage on any pin with respect to Ground (V_{SS}) |
| | V_O | -0.3 ~ $V_{DD} + 0.3$ | V | |
| | I_{OH} | 10 | mA | Maximum output current sourced by (I_{OH} per I/O Pin) |
| | ΣI_{OH} | 80 | mA | Maximum current (ΣI_{OH}) |
| | I_{OL} | 20 | mA | Maximum current sunk by (I_{OL} per I/O Pin) |
| | ΣI_{OL} | 160 | mA | Maximum current (ΣI_{OL}) |
| Total Power Dissipation | f_{XIN} | 600 | mW | - |
| Storage Temperature | T_{STG} | -65 ~ +150 | °C | °C |

Note: Stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at any other conditions above those indicated in

the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

7.2 Recommended Operating Conditions

| Parameter | Symbol | Condition | Specifications | | Unit |
|-----------------------|-----------|-----------------------------------|----------------|------|------|
| | | | Min. | Max. | |
| Supply Voltage | V_{DD} | $f_{XIN} = 0.4 \sim 12\text{MHz}$ | 4.5 | 5.5 | V |
| | | $f_{XIN} = 0.4 \sim 8\text{MHz}$ | 2.7 | 5.5 | V |
| Operating Temperature | T_{OPR} | $V_{DD} = 4.5 \sim 5.5\text{V}$ | -40 | 85 | °C |

7.3 A/D Converter Characteristics

($T_a = -40 \sim 85^\circ\text{C}$, $V_{SS} = 0\text{V}$, $V_{DD} = AV_{DD} = 2.7 \sim 5.5\text{V}$ @ $f_{XIN} = 4\text{MHz}$)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|------------------------------|-----------|--|---|------|---------|------|
| Resolution | - | - | - | 10 | - | BIT |
| Total Accuracy | - | $AV_{DD} = V_{DD} = 5.12\text{V}$ $f_{XIN} = 4\text{MHz}$ | - | - | ± 3 | LSB |
| Integral Linearity Error | ILE | | - | - | ± 2 | LSB |
| Differential Linearity Error | DLE | | - | - | ± 2 | LSB |
| Zero Offset Error | ZOE | | - | - | ± 2 | LSB |
| Full Scale Error | FSE | | - | - | ± 2 | LSB |
| Conversion Time | t_{CON} | | 10bit conversion $f_{XIN} = 4\text{MHz}$ | 13* | - | - |

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|----------------------|------------|------------------------|----------|------|--------------|---------|
| Analog Input Voltage | V_{AN} | - | V_{SS} | - | AV_{DD} | V |
| Analog Power Supply | AV_{DD} | - | - | - | V_{DD} | V |
| Analog Ground | V_{SS} | - | V_{SS} | - | $V_{SS}+0.3$ | V |
| Analog Input Current | I_{ADIN} | $AV_{DD}=V_{DD}=5.12V$ | - | - | 10 | μA |
| Analog Block Current | I_{ADC} | $AV_{DD}=V_{DD}=5.12V$ | - | 200 | 300 | μA |

Note : $4MHz(f_{XIN}) / 2^2 \times 13Cycle = 13\mu S$

7.4 DC Electrical Characteristics

($T_A=-40\sim 85^\circ C$, $V_{DD}=5.0V\pm 10\%$, $V_{SS}=0V$, $f_{XIN}=8MHz$)

| Parameter | Symbol | Pin/Condition | Min. | Typ. | Max. | Unit |
|--------------------------------|-----------|---|--------------|------|--------------|-----------|
| Input High Voltage | V_{IH1} | INT0, INT1, INT2, INT3, EC0, EC1, SI, SCK, ACLK0, ACLK1, RxD0, RxD1, \overline{RESET} | $0.8V_{DD}$ | - | $V_{DD}+0.3$ | V |
| | V_{IH2} | R0, R1, R3, R4, R5, R6 | $0.7V_{DD}$ | - | $V_{DD}+0.3$ | V |
| | V_{IH3} | X_{IN} | $0.8V_{DD}$ | - | $V_{DD}+0.3$ | V |
| Input Low Voltage | V_{IL1} | INT0, INT1, INT2, INT3, EC0, EC1, SI, SCK, ACLK0, ACLK1, RxD0, RxD1, \overline{RESET} | -0.3 | - | $0.2V_{DD}$ | V |
| | V_{IL2} | R0, R1, R3, R4, R5, R6 | -0.3 | - | $0.3V_{DD}$ | V |
| | V_{IL3} | X_{IN} | -0.3 | - | $0.2V_{DD}$ | V |
| Output High Voltage | V_{OH1} | R0, R1, R3, R4, R5, R6 ($I_{OH}=-0.7mA$) | $V_{DD}-0.4$ | - | - | V |
| | V_{OH2} | X_{OUT} ($I_{OH}=-50\mu A$) | $V_{DD}-0.5$ | - | - | V |
| Output Low Voltage | V_{OL1} | R0, R1, R3, R4, R5, R6 ($I_{OL}=1.6mA$) | - | - | 0.4 | V |
| | V_{OL2} | X_{OUT} ($I_{OL}=50\mu A$) | - | - | 0.5 | V |
| High Current | I_{OL} | R3 ($V_{OL}=1V$) | - | - | 20 | mA |
| Input High Leakage Current | I_{IH} | R0, R1, R3, R4, R5, R6 | - | - | 1 | μA |
| Input Low Leakage Current | I_{IL} | R0, R1, R3, R4, R5, R6 | -1 | - | - | μA |
| Pull-up Resistor | RPU | R0, R1, R4 | 10 | - | 100 | $k\Omega$ |
| OSC Feedback Resistor | R_X | X_{IN} , X_{OUT} | 0.45 | - | 4.5 | $M\Omega$ |
| Internal RC WDT Period (RCWDT) | I_{IL} | $V_{DD}=4.5V$ | 33 | - | 100 | μS |
| Hysteresis | V_T | INT0, INT1, INT2, INT3, EC0, EC1, SI, SCK, ACLK0, ACLK1, RxD0, RxD1 | 0.3 | - | 0.8 | V |
| Power Fail Detect Voltage | V_{PFD} | | 2.2 | 2.7 | 3.2 | V |
| | | | 2.5 | 3.0 | 3.5 | V |
| | | | 1.9 | 2.4 | 2.9 | V |

| Parameter | Symbol | Pin/Condition | Min. | Typ. | Max. | Unit |
|----------------------|--------------------|---|------|------|------|------|
| Power Supply Current | I _{DD1} | Active Mode, X _{IN} =8MHz | - | - | 15 | mA |
| | I _{SLEEP} | Sleep Mode, X _{IN} =8MHz | - | - | 6 | mA |
| | I _{STOP} | Stop Mode, Oscillator Stop, X _{IN} =4MHz | - | - | 5 | μA |
| | I _{RCWDT} | Stop Mode, Oscillator Stop, X _{IN} =8MHz | - | - | 40 | μA |

7.5 AC Characteristics

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

| Parameter | Symbol | Pins | Specifications | | | Unit |
|-------------------------------------|--------------------|------------------------|----------------|------|------|-----------|
| | | | Min. | Typ. | Max. | |
| Operating Frequency | f_{XIN} | X_{IN} | 0.4 | - | 12 | MHz |
| Oscillation Stabilizing Time (4MHz) | t_{ST} | X_{IN}, X_{OUT} | - | - | 20 | mS |
| External Clock Pulse Width | t_{CPW} | X_{IN} | 35 | - | - | nS |
| External Clock Transition Time | t_{RCP}, t_{FCP} | X_{IN} | - | - | 20 | nS |
| Interrupt Pulse Width | t_{IW} | INT0, INT1, INT2, INT3 | 2 | - | - | t_{SYS} |
| RESET Input Width | t_{RST} | RESET | 8 | - | - | t_{SYS} |
| Event Counter Input Pulse Width | t_{ECW} | EC0, EC1 | 2 | - | - | t_{SYS} |
| Event Counter Transition Time | t_{REC}, t_{FEC} | EC0, EC1 | - | - | 20 | nS |

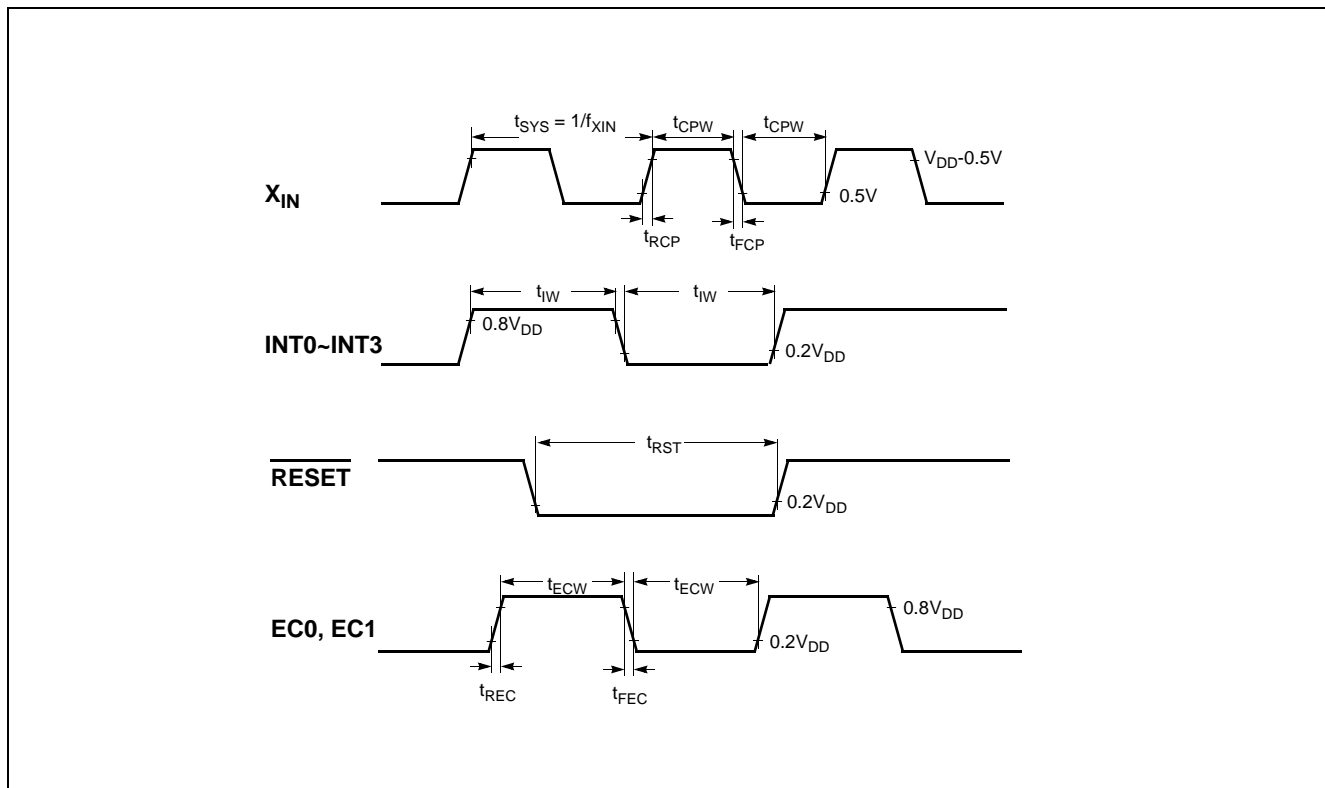


Figure 7-1 Timing Chart

7.6 Serial Interface Timing Characteristics

($T_A = -40 \sim +85^\circ\text{C}$, $V_{DD} = 5V \pm 10\%$, $V_{SS} = 0V$, $f_{XIN} = 8\text{MHz}$)

| Parameter | Symbol | Pins | Specifications | | | Unit |
|---|--------------------------|------|------------------|------|-------------|------|
| | | | Min. | Typ. | Max. | |
| Serial Input Clock Pulse | t_{SCYC} | SCK | $2t_{SYS} + 200$ | - | - | nS |
| Serial Input Clock Pulse Width | t_{SCKW} | SCK | $t_{SYS} + 70$ | - | - | nS |
| Serial Input Clock Pulse Transition Time | t_{FSCK} t_{RSCK} | SCK | - | - | 30 | nS |
| Serial Input Pulse Transition Time | t_{FSIN} t_{RSIN} | SI | - | - | 30 | nS |
| Serial Input Setup Time (External SCK) | t_{SUS} | SI | 100 | - | - | nS |
| Serial Input Setup Time (Internal SCK) | t_{SUS} | SI | 200 | - | - | nS |
| Serial Input Hold Time | t_{HS} | SI | $t_{SYS} + 70$ | - | - | nS |
| Serial Output Clock Cycle Time | t_{SCYC} | SCK | $4t_{SYS}$ | - | $16t_{SYS}$ | nS |
| Serial Output Clock Pulse Width | t_{SCKW} | SCK | $t_{SYS} - 30$ | - | - | nS |
| Serial Output Clock Pulse Transition Time | t_{FSCK} t_{RSCK} | SCK | - | - | 30 | nS |
| Serial Output Delay Time | s_{OUT} | SO | - | - | 100 | nS |

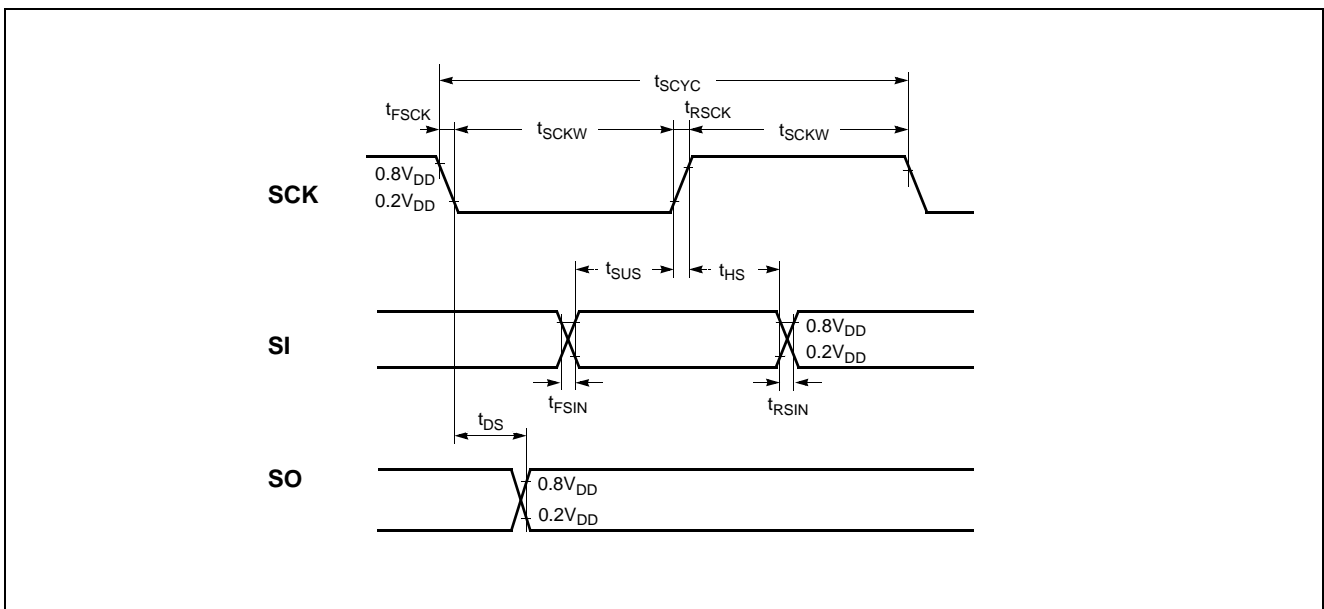


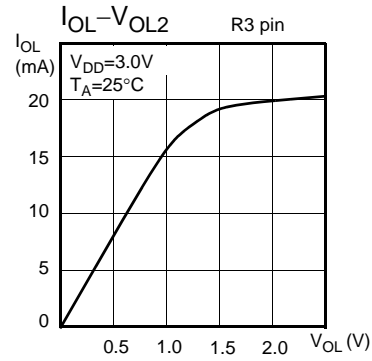
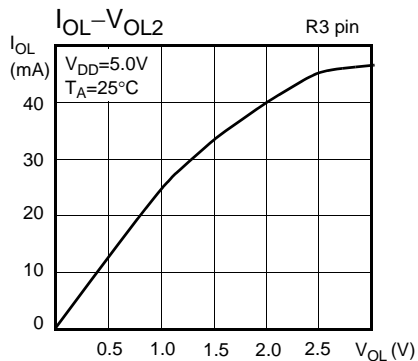
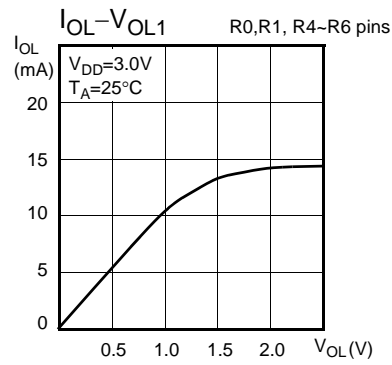
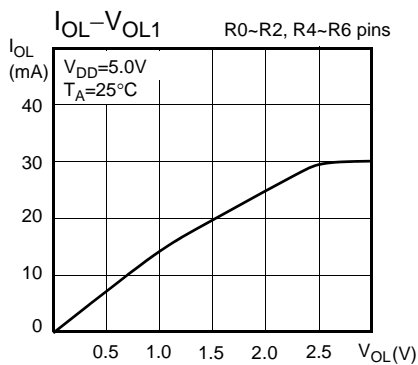
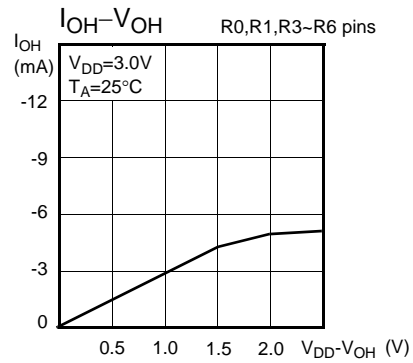
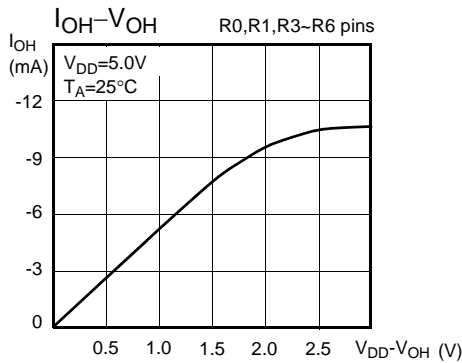
Figure 7-2 Serial I/O Timing Chart

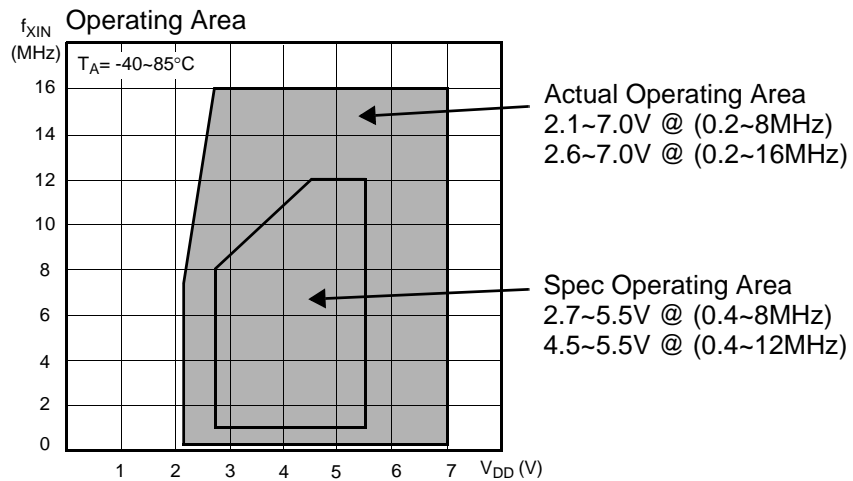
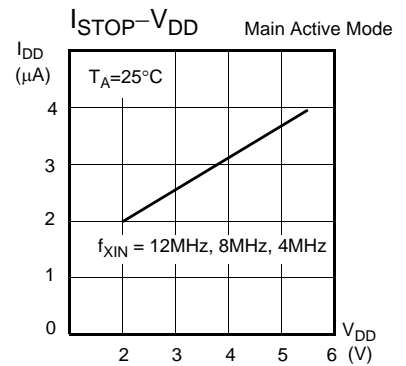
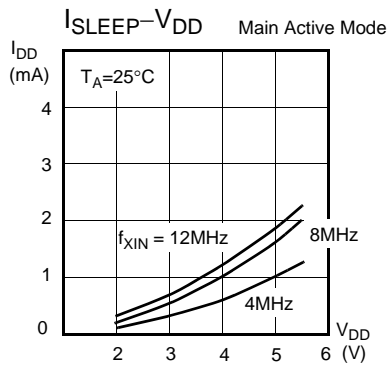
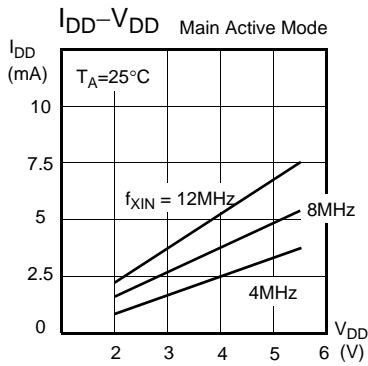
7.7 Typical Characteristic Curves

This 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





8. MEMORY ORGANIZATION

The MC80F0208/16/24 has separate address spaces for Program memory and Data Memory. Program memory can only be read, not written to. It can be up to 48K bytes of Program memory.

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.

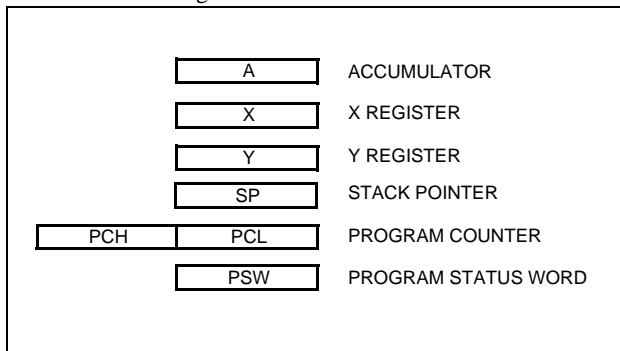


Figure 8-1 Configuration of Registers

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.

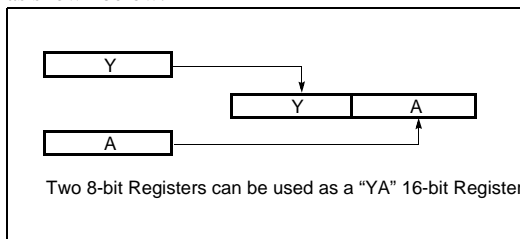


Figure 8-2 Configuration of YA 16-bit Register

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.

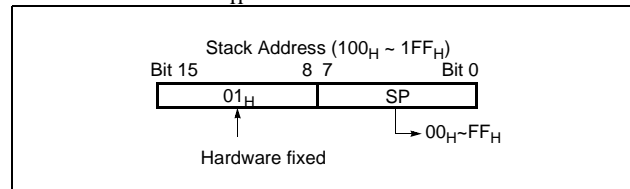
Stack Pointer: The Stack Pointer is an 8-bit register used for occurrence interrupts and calling out subroutines. Stack Pointer identifies the location in the stack to be accessed (save or restore).

Generally, SP is automatically updated when a subroutine call is

Data memory can be read and written to up to 1024 bytes including the stack area.

executed or an interrupt is accepted. However, if it is used in excess of the stack area permitted by the data memory allocating configuration, the user-processed data may be lost.

The stack can be located at any position within 100_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.

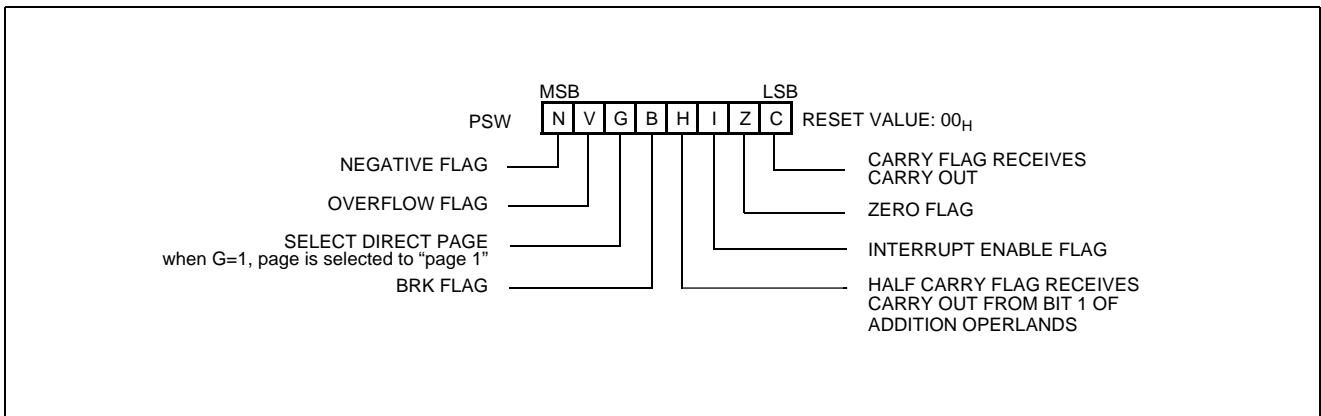


Figure 8-3 PSW (Program Status Word) Register

[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 00H to 0FH when this flag is "0". If it is set to "1", addressing area is assigned 100H to 1FFH. It is set by SETG instruction and cleared by CLRG.

[Overflow flag V]

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

[Negative flag N]

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

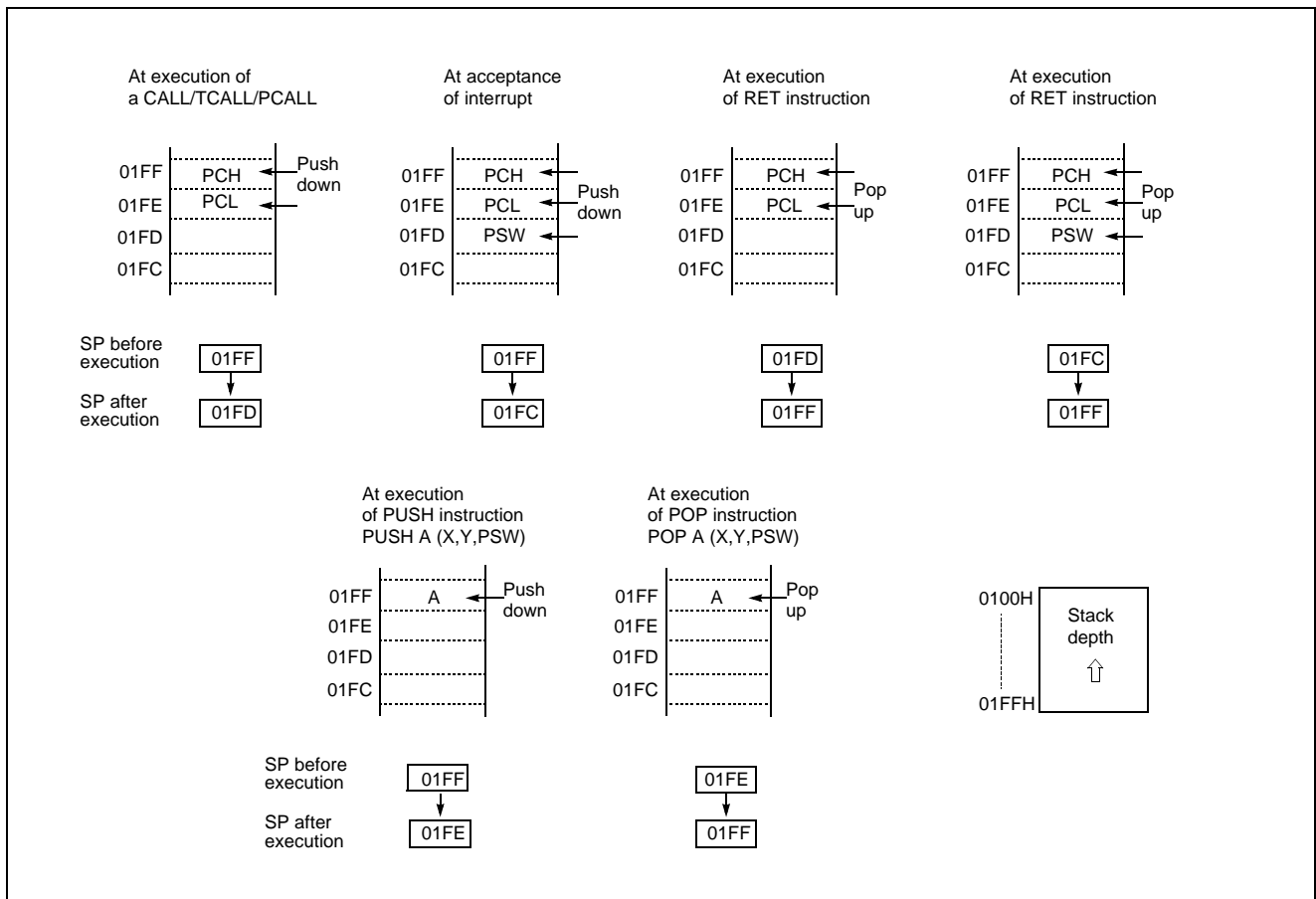


Figure 8-4 Stack Operation

8.2 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has 32/48K 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

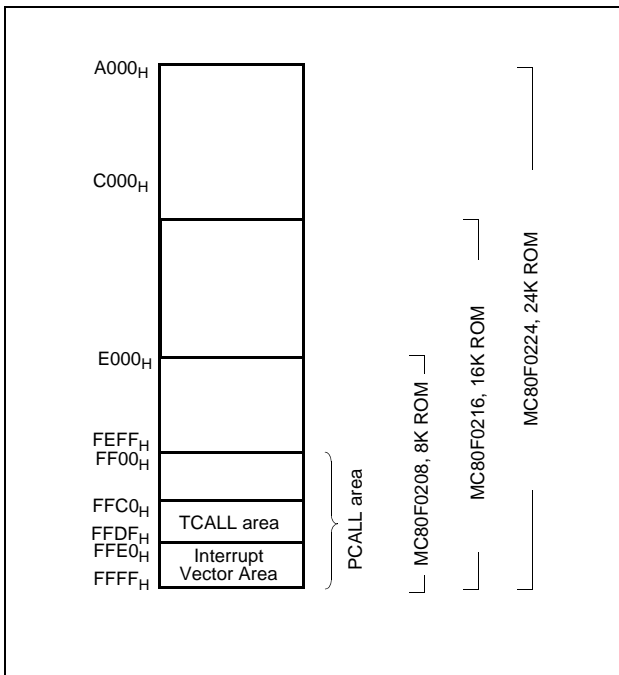


Figure 8-5 Program Memory Map

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: 0FFC0H for TCALL15, 0FFC2H for TCALL14, etc., as shown in Figure 8-7.

Example: Usage of TCALL

```

LDA    #5
      TCALL 0FH      ; 1BYTE INSTRUCTION
      :           ; INSTEAD OF 3 BYTES
      :           ; NORMAL CALL
;
;TABLE CALL ROUTINE
;
FUNC_A: LDA  LRG0
      RET
;
FUNC_B: LDA  LRG1  ②
      RET
;
;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 | Watch / Watchdog Timer Interrupt |
| E4 | A/D Converter |
| E6 | Timer/Counter 4 Interrupt |
| 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 | UART1 Rx/Tx interrupt |
| F4 | UART0 Rx/Tx 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

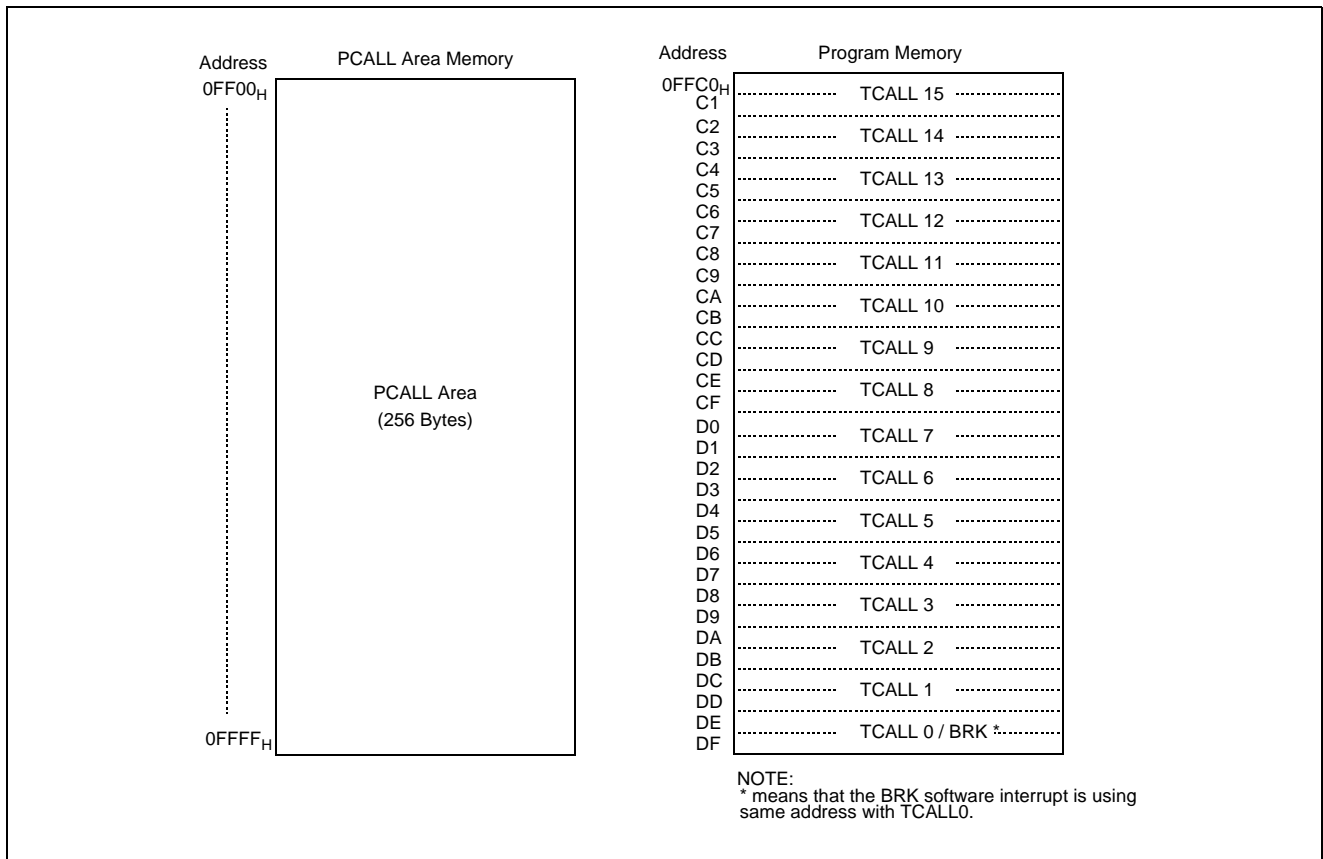
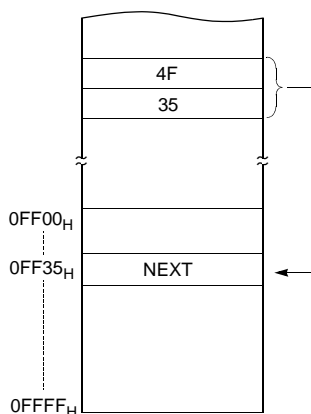


Figure 8-7 PCALL and TCALL Memory Area

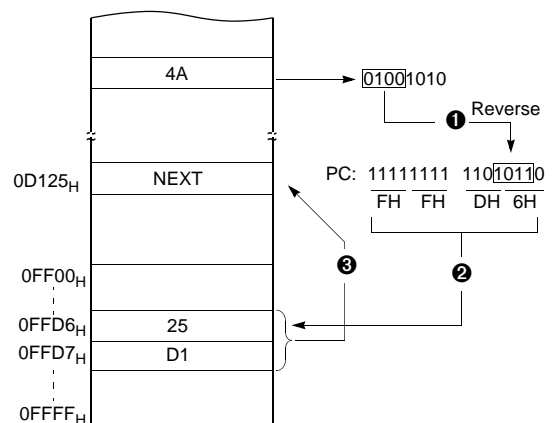
PCALL → rel

```
4F35 PCALL 35H
```



TCALL → n

```
4A TCALL 4
```



Example: The usage software example of Vector address for MC80F0208/16/24 .

```

;Interrupt Vector Table
    ORG    0FFE0H
    DW    BIT_TIMER    ; BIT
    DW    WATCH_WDT    ; WDT & WT
    DW    ADC           ; AD Converter
    DW    TIMER4       ; Timer-4
    DW    TIMER3       ; Timer-3
    DW    TIMER2       ; Timer-2
    DW    TIMER1       ; Timer-1
    DW    TIMER0       ; Timer-0
    DW    SIO          ; Serial Interface
    DW    UART1        ; UART1 Rx/Tx
    DW    UART0        ; UART0 Rx/Tx
    DW    INT3         ; Ext Int.3
    DW    INT2         ; Ext Int.2
    DW    INT1         ; Ext Int.1
    DW    INT0         ; Ext Int.0
    DW    RESET        ; Reset

    ORG    0A000H      ; 24K bytes ROM Start address
;*****
;          MAIN      PROGRAM      *
;*****
RESET:    DI          ;Disable All Interrupt
RAMCLEAR:
    LDX    #00H      ;USER RAM START ADDRESS LOAD !
    LDY    #0
RAMCLR1:
    LDA    #00H      ;Page0 Ram Clear(0000h ~ 00BFh)
    STA    {X}+      ;
    CMPX  #0C0H      ;
    BNE    RAMCLR1   ;

    INC    Y          ;
    STY    !RPR      ;Page1 Ram Select
    SETG   ;G-FLAG SET !

    LDX    #00H
RAMCLR2:
    LDA    #00H
    STA    {X}+
    CMPX  #00H
    BNE    RAMCLR2

    INC    Y
    CMPY  #4
    BCS   RAMCLR3    ;Page1 ~ Page3 Clear(0100h ~ 03FFh)

    STY    !RPR
    SETG

    BRA    RAMCLR2

RAMCLR3:
    STY    !RPR      ;Page4 Clear(0400h ~ 043Fh)
    SETG
    LDA    #00H      ;A <-- #0
    STA    {X}+
    CMPX  #40H      ;
    BNE    RAMCLR3

    CLRG          ;G-FLAG CLEAR !

    LDX    #0FFH
    TXSP          ;Initial Stack Point (01FFh)

```

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.

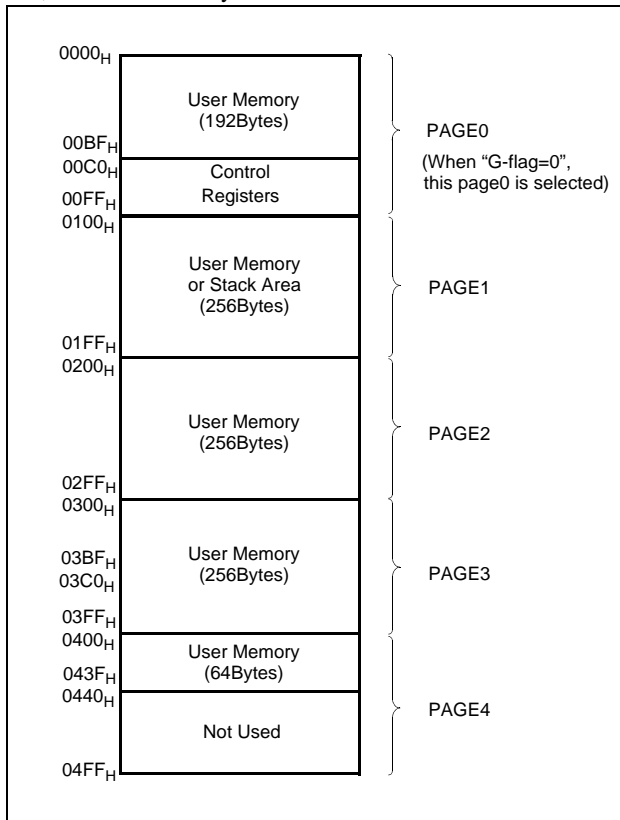


Figure 8-8 Data Memory Map

User Memory

The MC80F0208/16/24 has 1024 × 8 bits for the user memory (RAM). RAM pages are selected by RPR (See Figure 8-9).

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

Control Registers

The control registers are used by the CPU and Peripheral function blocks for controlling the desired operation of the device. Therefore these registers contain control and status bits for the interrupt system, the timer/ counters, analog to digital converters and I/O ports. The control registers are in address range of 0C0H to 0FFH.

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

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

Note: Write only registers can not be accessed by bit manipulation instruction. Do not use read-modify-write instruction. Use byte manipulation instruction, for example "LDM".

Example; To write at CKCTLR

```
LDM CLCTLR, #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 22.

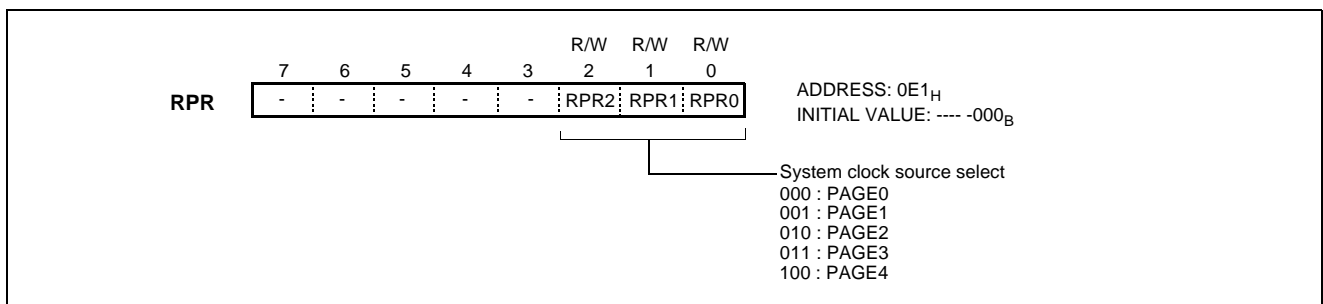


Figure 8-9 RPR(RAM Page Select Register)

| Address | Register Name | Symbol | R/W | Initial Value | | | | | | | | Addressing Mode | |
|---------|--------------------------------|--------|-----|---------------|---|---|---|---|---|---|---|-----------------|------------------------|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| 00C0 | R0 port data register | R0 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit ¹ |
| 00C1 | R0 port I/O direction register | R0IO | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte ² |
| 00C2 | R1 port data register | R1 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00C3 | R1 port I/O direction register | R1IO | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00C4 | Reserved | | | | | | | | | | | | |
| 00C5 | Reserved | | | | | | | | | | | | |
| 00C6 | R3 port data register | R3 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00C7 | R3 port I/O direction register | R3IO | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00C8 | R4 port data register | R4 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00C9 | R4 port I/O direction register | R4IO | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00CA | R5 port data register | R5 | R/W | - | - | - | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00CB | R5 port I/O direction register | R5IO | W | - | - | - | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00CC | R6 port data register | R6 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00CD | R6 port I/O direction register | R6IO | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00CE | Reserved | | | | | | | | | | | | |
| 00CF | Reserved | | | | | | | | | | | | |
| 00D0 | Timer 0 mode control register | TM0 | R/W | - | - | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00D1 | Timer 0 register | T0 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| | Timer 0 data register | TDR0 | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | Timer 0 capture data register | CDR0 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 00D2 | Timer 1 mode control register | TM1 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00D3 | Timer 1 data register | TDR1 | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | byte |
| 00D4 | Timer 1 register | T1 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| | Timer 1 capture data register | CDR1 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 00D5 | Reserved | | | | | | | | | | | | |
| 00D6 | Timer 2 mode control register | TM2 | R/W | - | - | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00D7 | Timer 2 register | T2 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| | Timer 2 data register | TDR2 | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | Timer 2 capture data register | CDR2 | R | 0 | 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 | 0 | byte, bit |
| 00D9 | Timer 3 data register | TDR3 | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | byte |
| | Timer 3 PWM period register | T3PPR | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |

Table 8-1 Control Registers

| Address | Register Name | Symbol | R/W | Initial Value | | | | | | | | Addressing Mode | |
|---------|--|--------|------|---------------|---|-----------|---|---|---|---|------|-----------------|------|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| 00DA | Timer 3 register | T3 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| | Timer 3 PWM duty register | T3PDR | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| | Timer 3 capture data register | CDR3 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |
| 00DB | Timer 3 PWM high register | T3PWHR | W | - | - | - | - | 0 | 0 | 0 | 0 | byte | |
| 00DC | Timer 4 mode control register | TM4 | R/W | - | - | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit | |
| 00DD | Timer 4 low register | T4L | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte | |
| | Timer 4 low data register | TDR4L | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | |
| | Timer 4 capture low data register | CDR4L | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |
| 00DE | Timer 4 high register | T4H | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte | |
| | Timer 4 high data register | TDR4H | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | | |
| | Timer 4 capture high data register | CDR4H | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | |
| 00DF | Interrupt flag register | IFR | R/W | - | - | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit | |
| 00E0 | Buzzer driver register | BUZR | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | byte | |
| 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 | |
| 00E4 | Reserved | | | | | | | | | | | | |
| 00E5 | Reserved | | | | | | | | | | | | |
| 00E6 | UART0 mode register | ASIMR0 | R/W | 0 | 0 | 0 | 0 | - | 0 | 0 | - | byte, bit | |
| 00E7 | UART0 status register | ASISR0 | R | - | - | - | - | - | 0 | 0 | 0 | byte | |
| 00E8 | UART0 Baud rate generator control register | BRGCR0 | R/W | - | 0 | 0 | 1 | 0 | 0 | 0 | 0 | byte, bit | |
| 00E9 | UART0 Receive buffer register | RXR0 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte | |
| | UART0 Transmit shift register | TXR0 | 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 | 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 | | |
| 00F3 | Reserved | | | | | | | | | | | | |

Table 8-1 Control Registers

| Address | Register Name | Symbol | R/W | Initial Value | | | | | | | | Addressing Mode | |
|---------|--|--------|-----|---------------|---|---|---|---|---|---|---|-----------------|-----------|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| 00F4 | Watch dog timer register | WDTR | W | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | byte |
| | Watch dog timer data register | WDTDR | R | Undefined | | | | | | | | | |
| 00F5 | Stop & sleep mode control register | SSCR | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00F6 | Watch timer mode register | WTMR | R/W | 0 | - | - | 0 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 00F7 | PFD control register | PFDR | R/W | - | - | - | - | - | 0 | 0 | 0 | 0 | byte, bit |
| 00F8 | Port selection register 0 | PSR0 | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00F9 | Port selection register 1 | PSR1 | W | - | - | - | - | 0 | 0 | 0 | 0 | 0 | byte |
| 00FA | Reserved | | | | | | | | | | | | |
| 00FB | Reserved | | | | | | | | | | | | |
| 00FC | Pull-up selection register 0 | PU0 | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00FD | Pull-up selection register 1 | PU1 | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00FE | Pull-up selection register 4 | PU4 | W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| 00FF | Reserved | | | | | | | | | | | | |
| 0EE6 | UART1 mode register | ASIMR1 | R/W | 0 | 0 | 0 | 0 | - | 0 | 0 | - | - | byte, bit |
| 0EE7 | UART1 status register | ASISR1 | R | - | - | - | - | - | 0 | 0 | 0 | 0 | byte |
| 0EE8 | UART1 Baud rate generator control register | BRGCR1 | R/W | - | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | byte, bit |
| 0EE9 | UART1 Receive buffer register | RXR1 | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | byte |
| | UART1 Transmit shift register | TXR1 | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |

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.
3. The UART1 control register ASIMR1, ASISR1, BRGCR1, RXR1 and TXR1 are located at EE6H ~ EE9H address. These address must be accessed(read and written) by absolute addressing manipulation instruction.

*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 | | | | | | | |
| 0C4H | | Reserved | | | | | | | |
| 0C5H | | Reserved | | | | | | | |
| 0C6H | R3 | R3 Port Data Register | | | | | | | |
| 0C7H | R3IO | R3 Port Direction Register | | | | | | | |
| 0C8H | R4 | R4 Port Data Register | | | | | | | |
| 0C9H | R4IO | R4 Port Direction Register | | | | | | | |
| 0CAH | R5 | - | - | - | R5 Port Data Register | | | | |
| 0CBH | R5IO | - | - | - | R5 Port Direction Register | | | | |
| 0CCH | R6 | R6 Port Data Register | | | | | | | |
| 0CDH | R6IO | R6 Port Direction Register | | | | | | | |
| 0CEH | | Reserved | | | | | | | |
| 0CFH | | Reserved | | | | | | | |
| 0D0H | TM0 | - | - | CAP0 | T0CK2 | T0CK1 | T0CK0 | T0CN | T0ST |
| 0D1H | T0/TDR0/ CDR0 | Timer0 Register / Timer0 Data Register / Timer0 Capture Data Register | | | | | | | |
| 0D2H | TM1 | - | 16BIT | - | CAP1 | T1CK1 | T1CK0 | T1CN | T1ST |
| 0D3H | TDR1 | Timer1 Data 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 | | | |
| 0DCH | TM4 | - | - | CAP4 | T4CK2 | T4CK1 | T4CK0 | T4CN | T4ST |
| 0DDH | T4L/ TDR4L/ CDR4L | Timer4 Register Low / Timer4 Data Register Low / Timer4 Capture Data Register Low | | | | | | | |

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 |
|---------|-------------------------|--|-------------------------------|--------|--------|--------|--------|----------------------|--------|
| 0DEH | T4H/ TDR4H/ CDR4H | Timer4 Register High / Timer4 Data Register High / Timer4 Capture Data Register High | | | | | | | |
| 0DFH | IFR | - | - | RX0IOF | TX0IOF | RX1IOF | TX1IOF | WTIOF | WDTIOF |
| 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 | | | | | | | |
| 0E4H | | Reserved | | | | | | | |
| 0E5H | | Reserved | | | | | | | |
| 0E6H | ASIMR0 | TXE0 | RXE0 | PS01 | PS00 | - | SL0 | ISRM0 | - |
| 0E7H | ASISR0 | - | - | - | - | - | PE0 | FE0 | OVE0 |
| 0E8H | BRGCR0 | - | TPS02 | TPS01 | TPS00 | MLD03 | MLD02 | MLD01 | MLD00 |
| 0E9H | RXR0 | UART0 Receive Buffer Register | | | | | | | |
| | TXR0 | UART0 Transmit Shift Register | | | | | | | |
| 0EAH | IENH | INT0E | INT1E | INT2E | INT3E | RXE | TXE | SIOE | T0E |
| 0EBH | IENL | T1E | T2E | T3E | T4E | ADCE | WDTE | WTE | BITE |
| 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 |
| 0F3H | | Reserved | | | | | | | |
| 0F4H | WDTR | WDTCL | 7-bit Watchdog Timer Register | | | | | | |
| | WDTDR | Watchdog Timer Data Register (Counter Register) | | | | | | | |
| 0F5H | SSCR | Stop & Sleep Mode Control Register | | | | | | | |
| 0F6H | WTMR | WTEN | - | - | WTIN2 | WTIN1 | WTIN0 | WTCK1 | WTCK0 |
| 0F7H | PFDR | - | - | - | - | - | PFDEN | PFDM | PFDS |
| 0F8H | PSR0 | PWM30 | - | EC1E | EC0E | INT3E | INT2E | INT1E | INT0E |
| 0F9H | PSR1 | - | - | - | - | XTEN | BUZO | - | - |
| 0FAH | | Reserved | | | | | | | |
| 0FBH | | Reserved | | | | | | | |
| 0FCH | PU0 | R0 Pull-up Selection Register | | | | | | | |
| 0FDH | PU1 | R1 Pull-up Selection Register | | | | | | | |
| 0FEH | PU4 | R4 Pull-up Selection Register | | | | | | | |

Table 8-2 Control Register Function Description

| Address | Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|-------------------|--------|-------------------------------|-------|-------|-------|-------|-------|-------|-------|
| 0FFH | | Reserved | | | | | | | |
| EE6H ² | ASIMR1 | TXE1 | RXE1 | PS11 | PS10 | - | SL1 | ISRM1 | - |
| EE7H ² | ASISR1 | - | - | - | - | - | PE1 | FE1 | OVE1 |
| EE8H ² | BRGCR1 | - | TPS12 | TPS11 | TPS10 | MLD13 | MLD12 | MLD11 | MLD10 |
| EE9H ² | RXR1 | UART1 Receive Buffer Register | | | | | | | |
| | TXR1 | UART1 Transmit Shift Register | | | | | | | |

Table 8-2 Control Register Function Description

- The register *BITR* and *CKCTLR* are located at same address. Address *F2H* 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".
- The *UART1* control register *ASIMR1*, *ASISR1*, *BRGCR1*, *RXR1* and *TXR1* are located at *EE6H ~ EE9H* address. These address must be accessed (read and written) by absolute addressing manipulation instruction.

8.4 Addressing Mode

The MC800 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

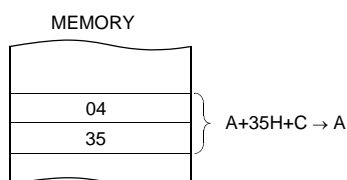
Register addressing accesses the A, X, Y, C and PSW.

8.4.2 Immediate Addressing → #imm

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

Example:

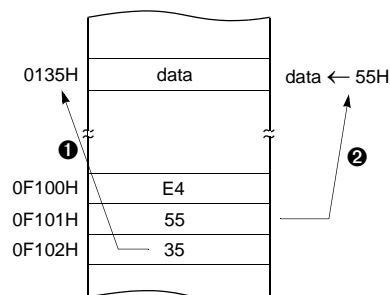
```
0435 ADC #35H
```



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
```

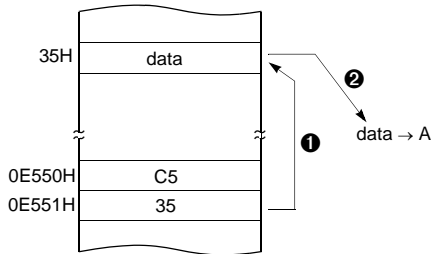


8.4.3 Direct Page Addressing → dp

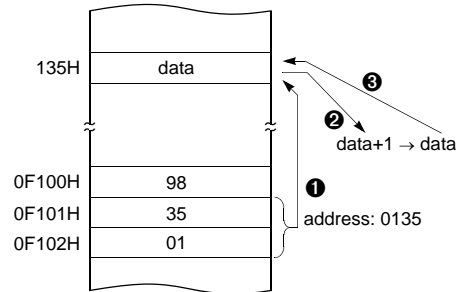
In this mode, a address is specified within direct page.

Example; G=0

```
C535 LDA 35H ;A ←RAM[35H]
```



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



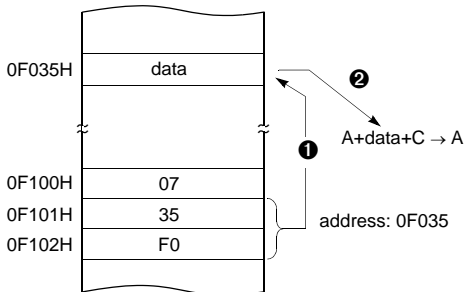
8.4.4 Absolute Addressing → !abs

Absolute addressing sets corresponding memory data to Data, i.e. second byte (Operand I) of command becomes lower level address and third byte (Operand II) becomes upper level address. With 3 bytes command, it is possible to access to whole memory area.

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

Example;

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



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

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

8.4.5 Indexed Addressing

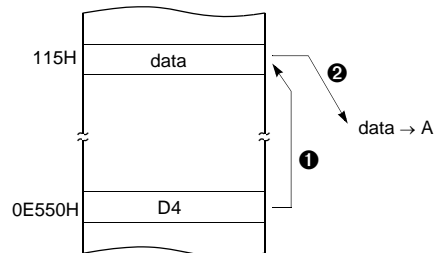
X indexed direct page (no offset) → {X}

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

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

Example; X=15H, G=1

```
D4 LDA {X} ;ACC←RAM[X].
```



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

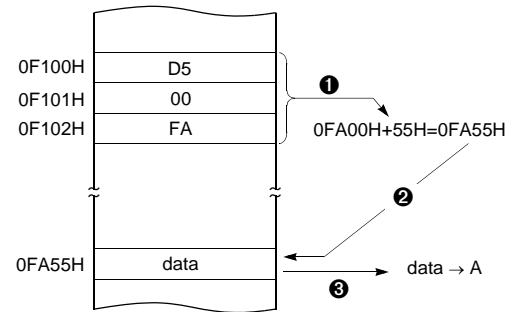
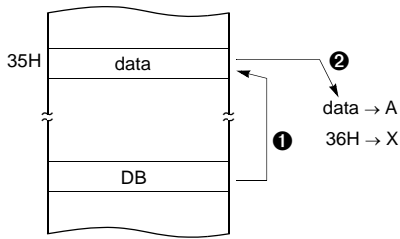
In this mode, a address is specified within direct page by the X register and the content of X is increased by 1.

LDA, STA

Example; G=0, X=35H

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

D500FA LDA !0FA00H+Y



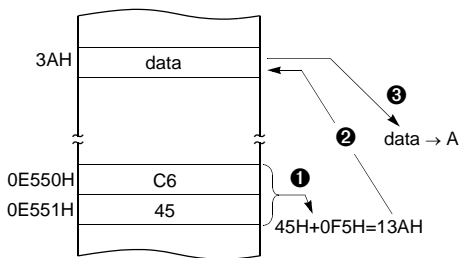
X indexed direct page (8 bit offset) → dp+X

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

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

Example; G=0, X=0F5H

C645 LDA 45H+X



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

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

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

Y indexed absolute → !abs+Y

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

Example; Y=55H

8.4.6 Indirect Addressing

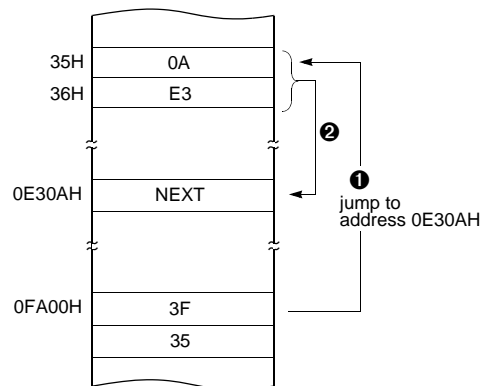
Direct page indirect → [dp]

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

JMP, CALL

Example; G=0

3F35 JMP [35H]



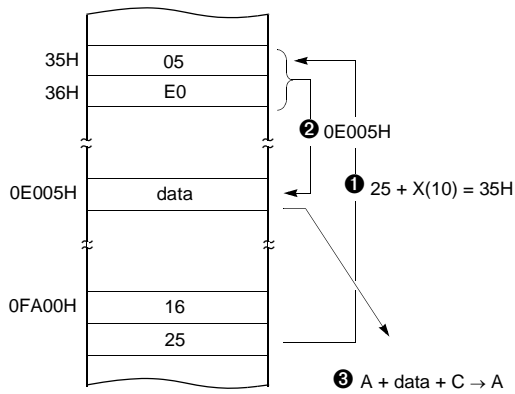
X indexed indirect → [dp+X]

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

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

Example; G=0, X=10H

1625 ADC [25H+X]



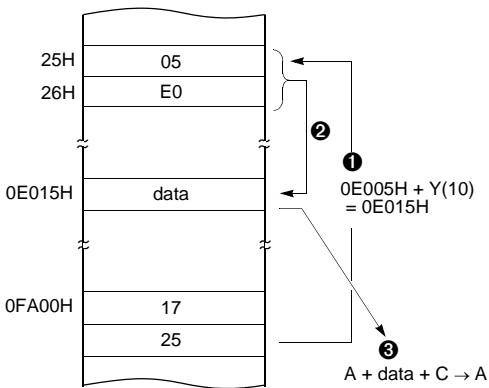
Y indexed indirect → [dp]+Y

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



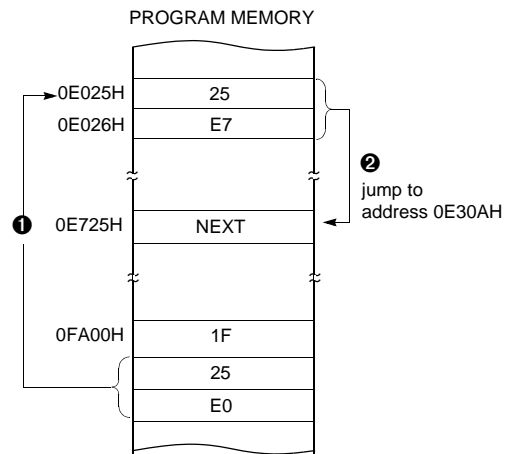
Absolute indirect → [!abs]

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

JMP

Example; G=0

1F25E0 JMP [!0C025H]



9. I/O PORTS

The MC80F0208/16/24 has six ports (R0, R1, R3, R4, R5 and R6). These ports pins may be multiplexed with an alternate function for the peripheral features on the device. R3 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 MC80F0208/16/24 have 0 written to them by reset function. On the other hand, its initial status is input.

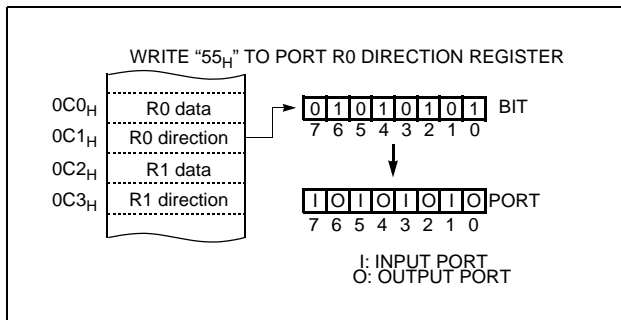
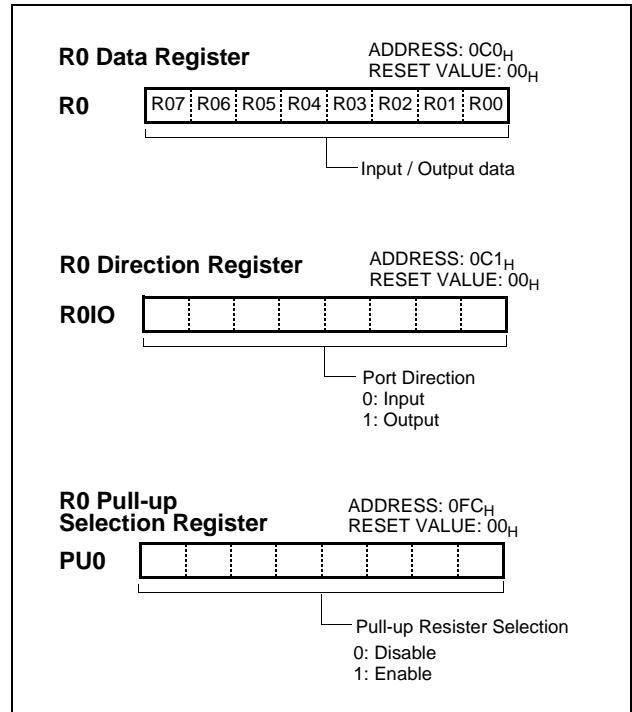


Figure 9-1 Example of port I/O assignment

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). The on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up selection register 0 (PU0).

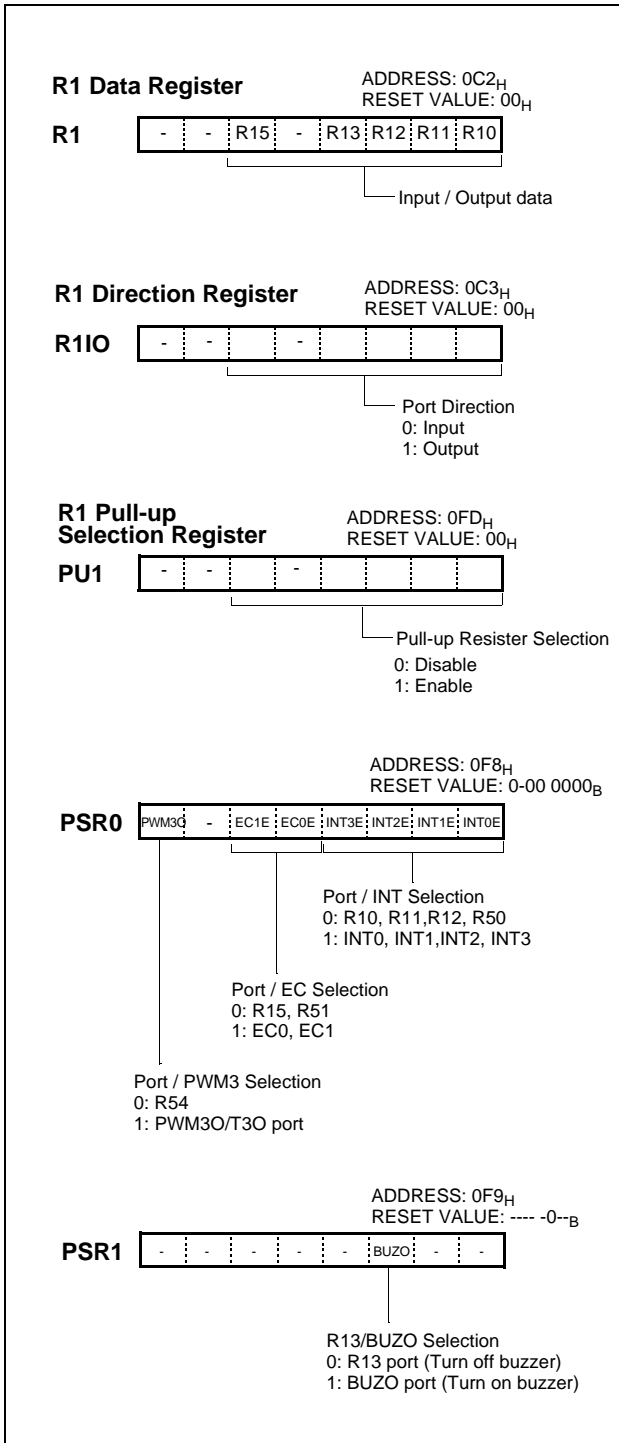


R1 and R1IO register: R1 is an 5-bit CMOS bidirectional I/O port (address 0C2_H). Each I/O pin can independently used as an input or an output through the R1IO register (address 0C3_H). The on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up selection register 1 (PU1).

In addition, Port R1 is multiplexed with various special features. The control register PSR0 (address 0F8_H) and PSR1 (address 0F9_H) controls the selection of alternate function. After reset, this value is “0”, port may be used as normal I/O port.

To use alternate function such as external interrupt, event counter input or timer clock output, write “1” in the corresponding bit of PSR0 or PSR1. Regardless of the direction register R1IO, PSR0 or PSR1 is selected to use as alternate functions, port pin can be used as a corresponding alternate features.

| Port Pin | Alternate Function |
|----------|--|
| R10 | INT0 (External Interrupt 0) |
| R11 | INT1 (External Interrupt 1) |
| R12 | INT2 (External Interrupt 2) |
| R13 | BUZO (Square-wave output for buzzer) |
| R15 | EC0 (Event counter input to Counter 0) |

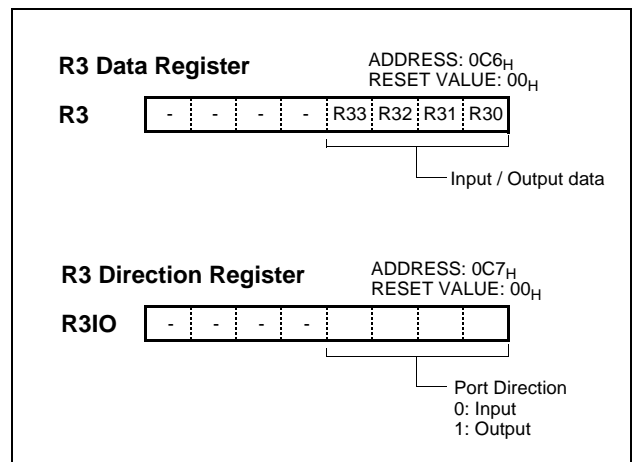


R3 and R3IO register: R3 is an 4-bit CMOS bidirectional I/O

port (address 0C6_H). Each I/O pin can independently used as an input or an output through the R3IO register (address 0C7_H).

In addition, Port R3 is multiplexed with various special features. After reset, this value is “0”, port may be used as normal I/O port.

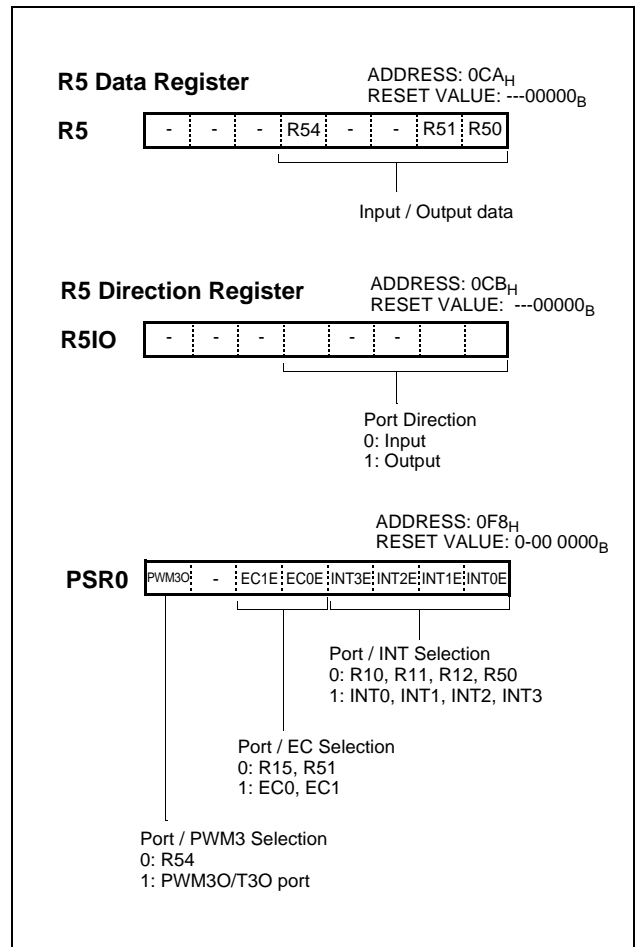
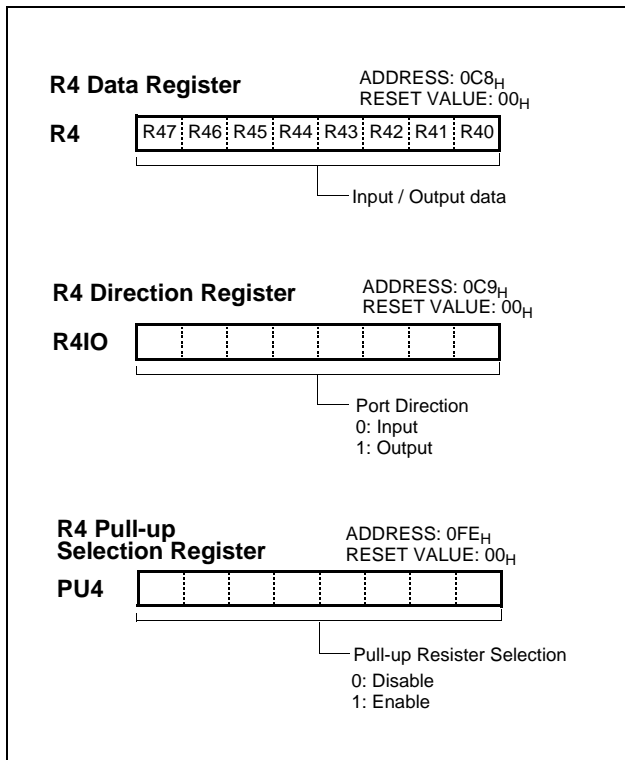
| Port Pin | Alternate Function |
|----------|---------------------------|
| R30 | - |
| R31 | ACLK1 (UART1 clock input) |
| R32 | RxD1 (UART1 data input) |
| R33 | TxD1 (UART1 data output) |



R4 and R4IO register: R4 is an 8-bit CMOS bidirectional I/O port (address 0C8_H). Each I/O pin can independently used as an input or an output through the R4IO register (address 0C9_H). The on-chip pull-up resistor can be connected to them in 1-bit units with a pull-up selection register 4 (PU4).

In addition, Port R4 is multiplexed with various special features. After reset, this value is “0”, port may be used as normal I/O port.

| Port Pin | Alternate Function |
|----------|------------------------------|
| R40 | - |
| R41 | - |
| R42 | SCK (SIO clock input/output) |
| R43 | SI (SIO data input) |
| R44 | SO (Serial1 data output) |
| R45 | ACLK0 (UART0 clock input) |
| R46 | RxD0 (UART0 data input) |
| R47 | TxD0 (UART0 data output) |



R5 and R5IO register: R5 is a 3-bit CMOS bidirectional I/O port (address 0CA_H). Each I/O pin can independently used as an input or an output through the R5IO register (address 0CB_H).

In addition, Port R5 is multiplexed with various special features. The control register PSR0 (address 0F8_H) and PSR1 (address 0F9_H) controls the selection of alternate function. After reset, this value is “0”, port may be used as normal I/O port.

To use alternate function such as external interrupt, event counter input, timer clock output or PWM output, write “1” in the corresponding bit of PSR0 or PSR1. Regardless of the direction register R5IO, PSR0 or PSR1 is selected to use as alternate functions, port pin can be used as a corresponding alternate features.

| Port Pin | Alternate Function |
|----------|--|
| R50 | INT3 (External Interrupt 3) |
| R51 | EC1 (Event counter input to Counter 2) |
| R54 | PWM30 (PWM3/T30 output) |

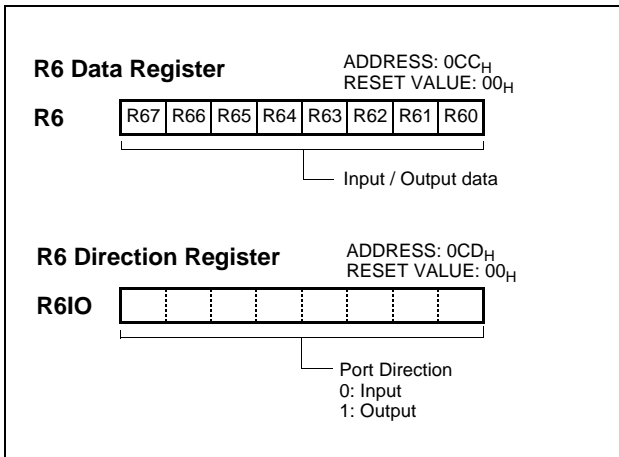
R6 and R6IO register: R6 is an 8-bit CMOS bidirectional I/O port (address 0CC_H). Each I/O pin can independently used as an input or an output through the R6IO register (address 0CD_H).

In addition, Port R6 is multiplexed with AD converter analog input AN0~AN7.

| Port Pin | Alternate Function |
|----------|---------------------------|
| R60 | AN0 (ADC input channel 0) |
| R61 | AN1 (ADC input channel 1) |
| R62 | AN2 (ADC input channel 2) |
| R63 | AN3 (ADC input channel 3) |
| R64 | AN4 (ADC input channel 4) |
| R65 | AN5 (ADC input channel 5) |
| R66 | AN6 (ADC input channel 6) |
| R67 | AN7 (ADC input channel 7) |

R6IO (address CD_H) controls the direction of the R6 pins, except when they are being used as analog input channels. The user don't have to keep the pins configured as inputs when using them as analog input channels, because the analog input mode is activated by the setting of ADC enable bit of ADCM register and ADC

channel selection



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 (CKCTRL). See "11. BASIC INTERVAL TIMER" on page 41 for details.

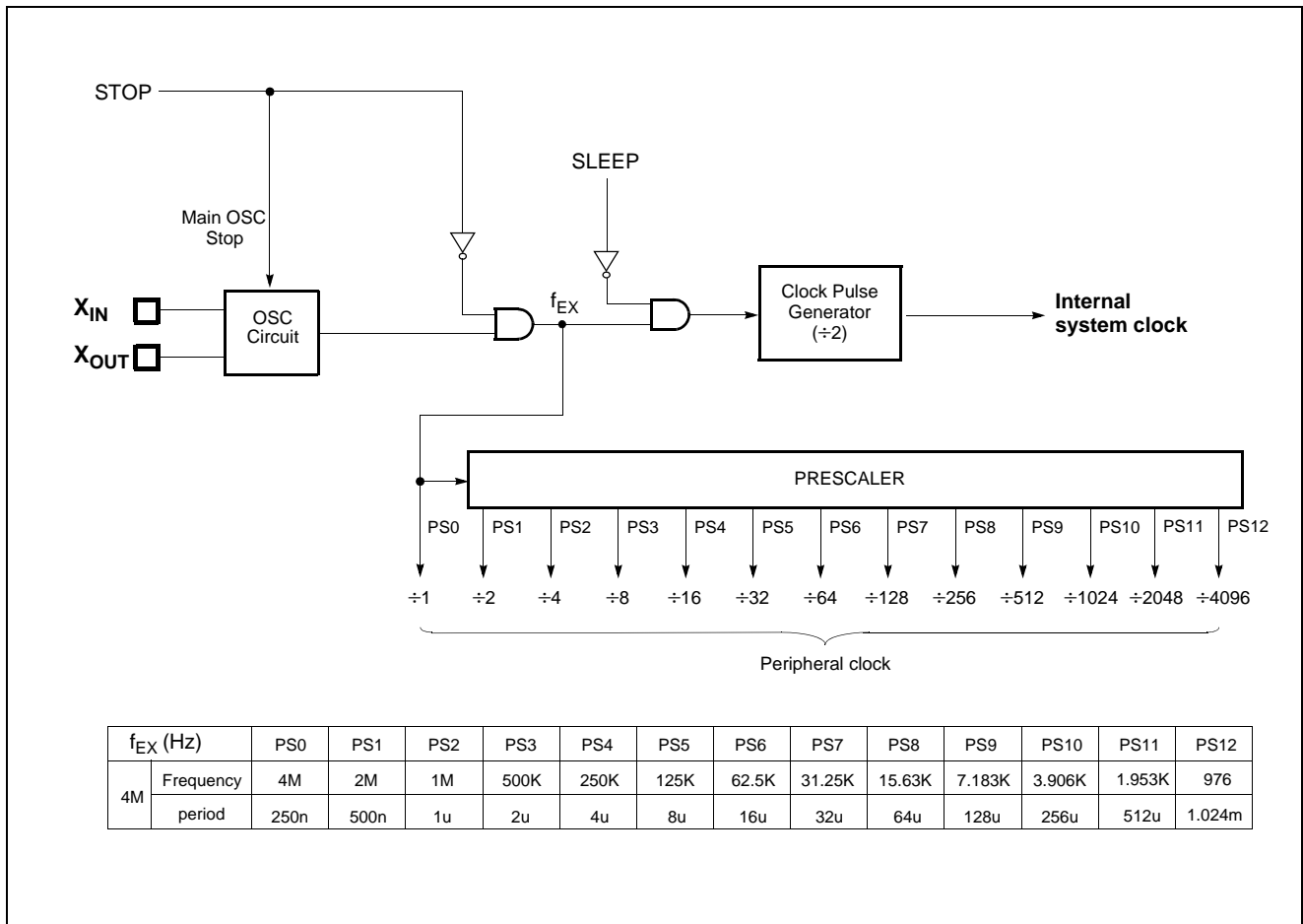


Figure 10-1 Block Diagram of Clock Generator

11. BASIC INTERVAL TIMER

The MC80F0208/16/24 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, this overflow causes the interrupt to be generated. The Basic Interval Timer is controlled by the clock control register (CKCTLR) shown in Figure 10-2.

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.

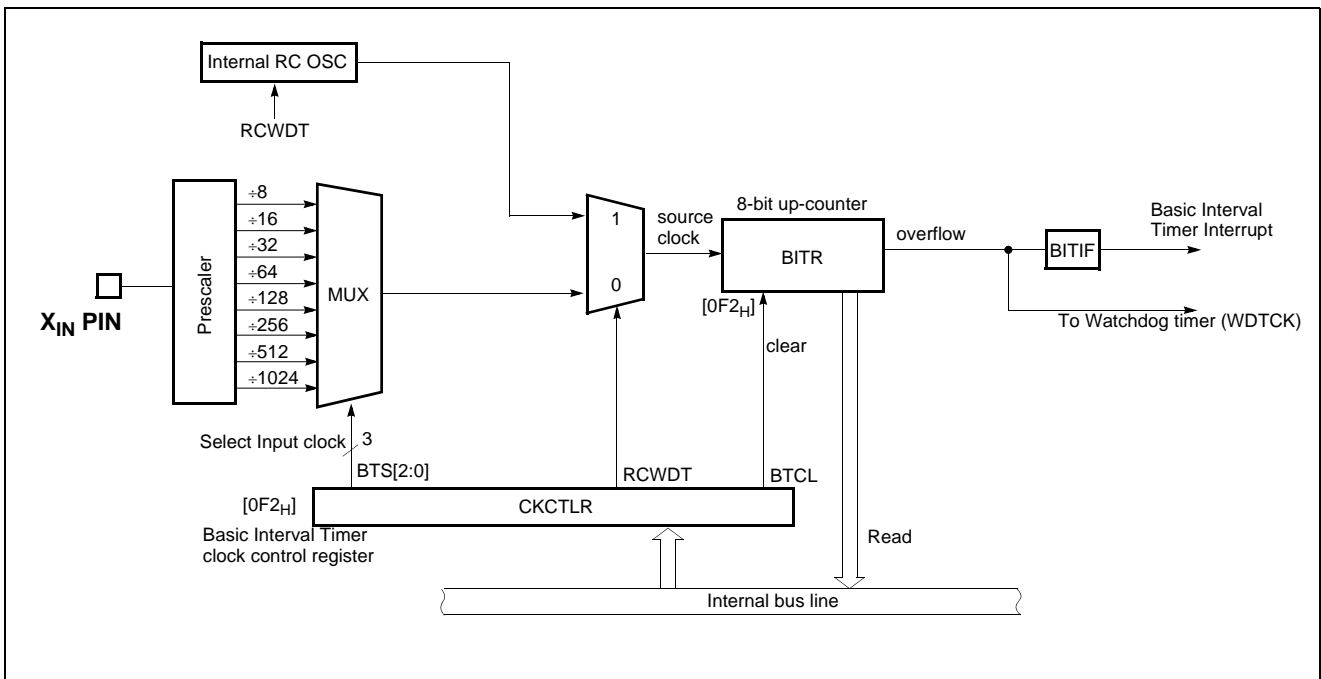


Figure 11-1 Block Diagram of Basic Interval Timer

| 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

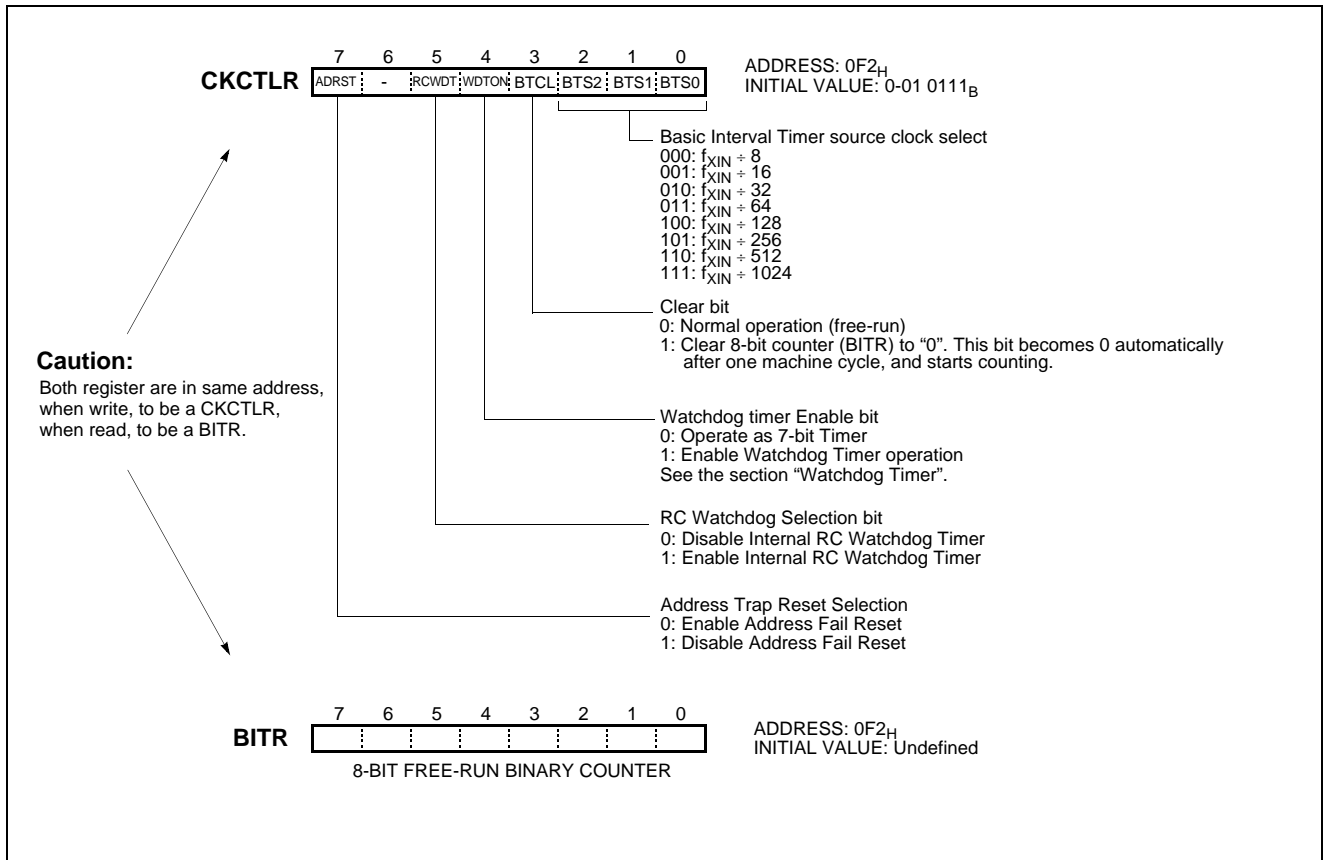


Figure 11-2 BITR: Basic Interval Timer Mode Register

Example 1:

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

```

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

Example 2:

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

```

:
LDM  CKCTRL, #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    CKCTRLR,#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, $CLK_{RCWDT} = 33 \sim 100 \mu S$

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

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

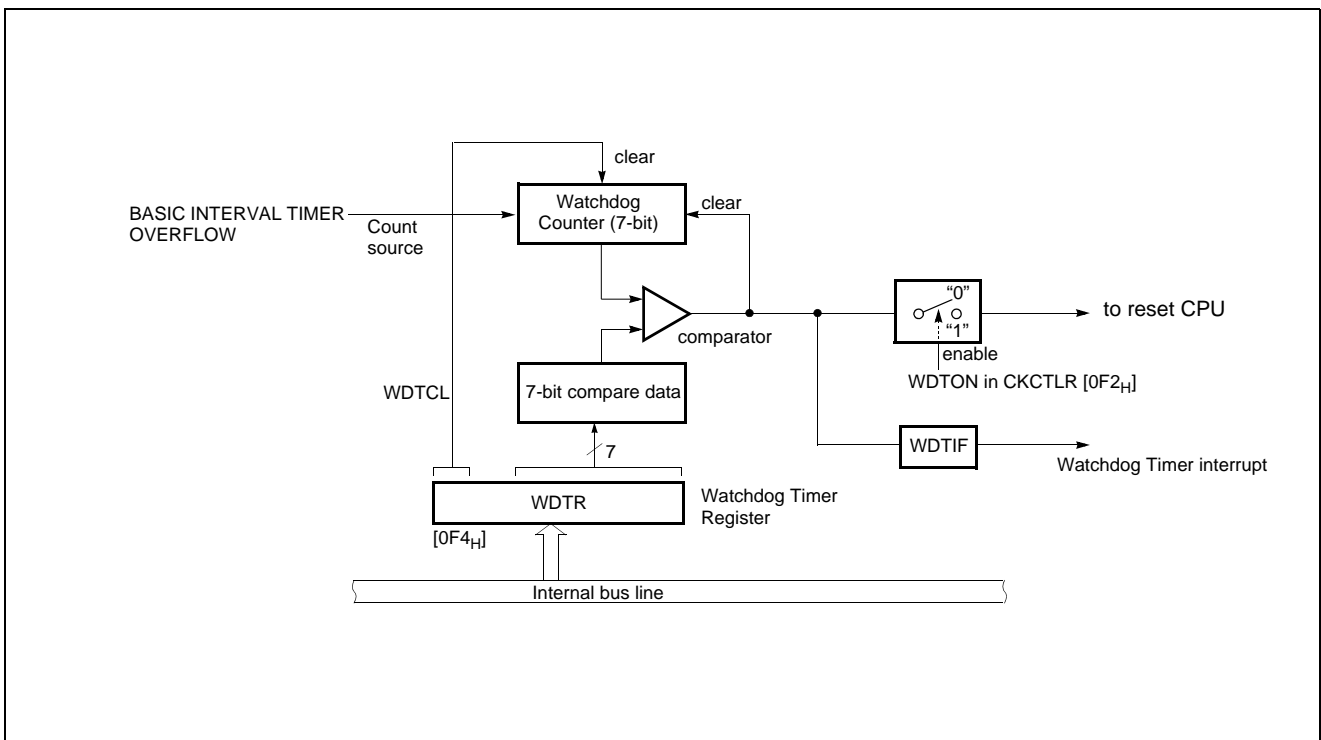


Figure 12-1 Block Diagram of Watchdog Timer

Watchdog Timer Control

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

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

If the malfunction occurs for any cause, the watchdog timer output will become active at the rising overflow from the binary

counters unless the binary counter is cleared. At this time, when WDTON=1, a reset is generated, which drives the 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).

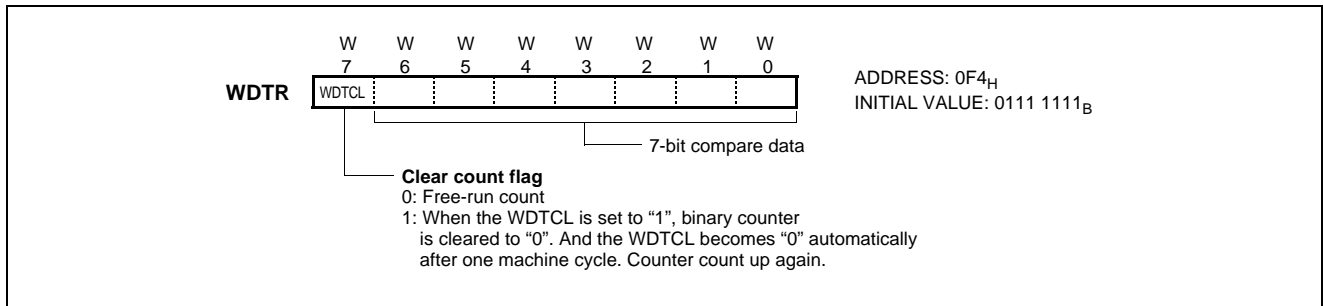


Figure 12-2 WDTCL: Watchdog Timer Control Register

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    WDTCL, #08FH
:
:
:
:
LDM    WDTCL, #08FH         ; Clear counter
:
:
:
:
LDM    WDTCL, #08FH         ; Clear counter
:
:
:
:
LDM    WDTCL, #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.

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

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    WDTCL, #8FH         ; WDTCL ← 1
:
    
```

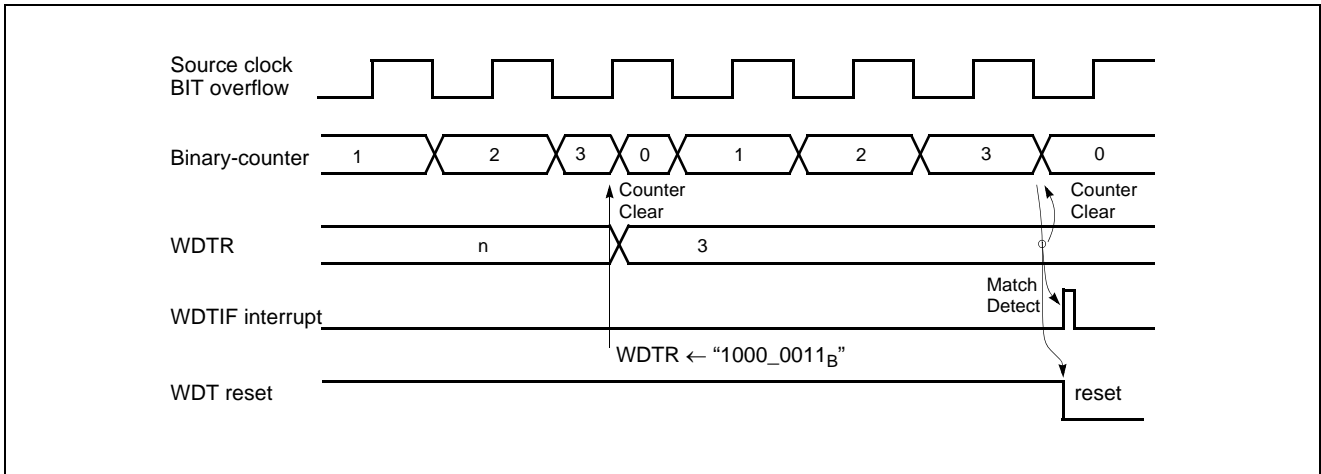



Figure 12-3 Watchdog timer Timing

If the watchdog timer output becomes active, a reset is generated, which drives the **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. The WDTIF bit of IFR register is set when watchdog timer interrupt is generated. (Refer to Figure 12-4)

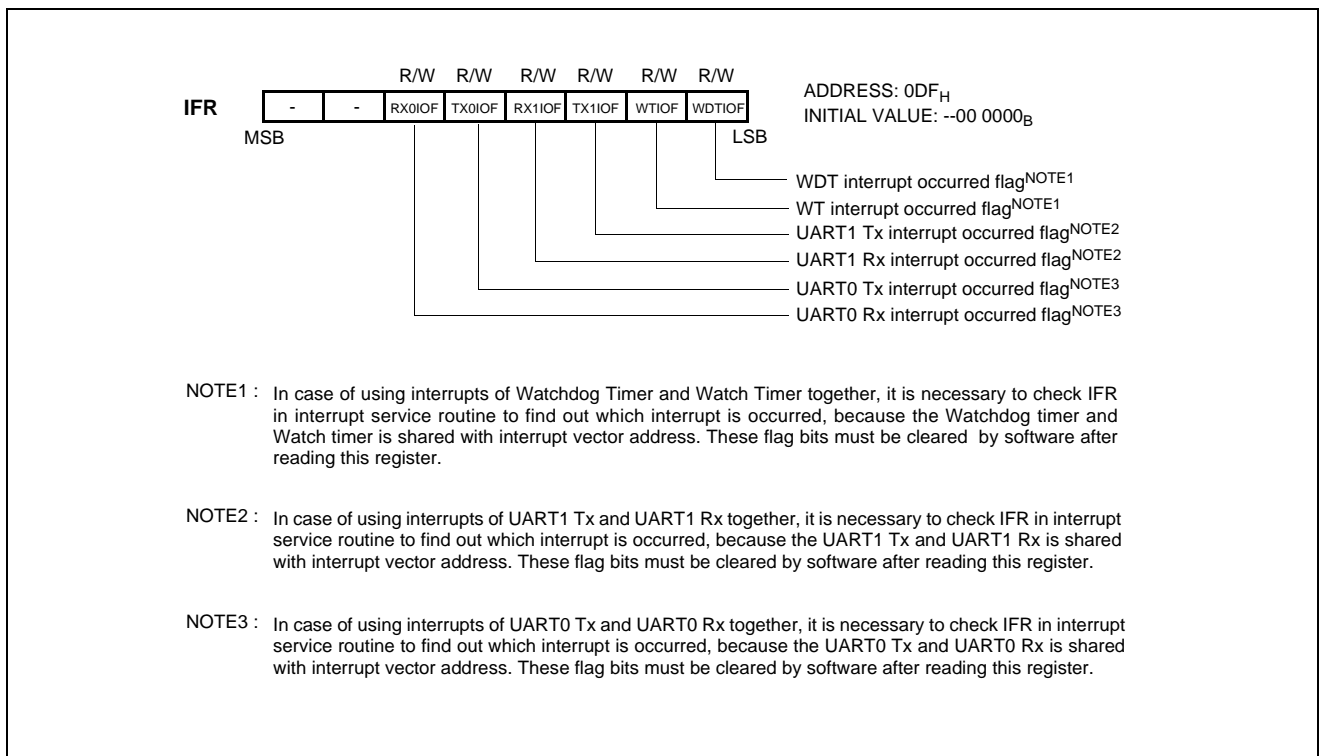


Figure 12-4 IFR(Interrupt Flag Register)

13. WATCH TIMER

The watch timer generates interrupt for watch operation. The watch timer consists of the clock selector, 15-bit binary counter, interval selector and watch timer mode register. It is a multi-purpose timer. It is generally used for watch design.

The bit 0,1 of WTMR select the clock source of watch timer among $f_{XIN} \div 2$, $f_{XIN} \div 2^7$ and main-clock(f_{XIN}). The f_{XIN} of main-clock is used usually for watch timer test, so generally it is not used for the clock source of watch timer. The $f_{XIN} \div 2^7$ of main-clock(4.194MHz) is used when the single clock system is orga-

nized. In $f_{XIN} \div 2^7$ clock source, if the CPU enters into stop mode, the main-clock is stopped and then watch timer is also stopped. The watch timer counter can output with period of max 1 seconds at sub-clock. The bit 2, 3, 4 of WTMR select the interrupt interval divide ratio selection of watch timer among 16, 64, 256, 1024, 4096, 8192, 16384 or 32768.

The WTIF bit of IFR register is set when watch timer interrupt is generated. (Refer to Figure 12-4)

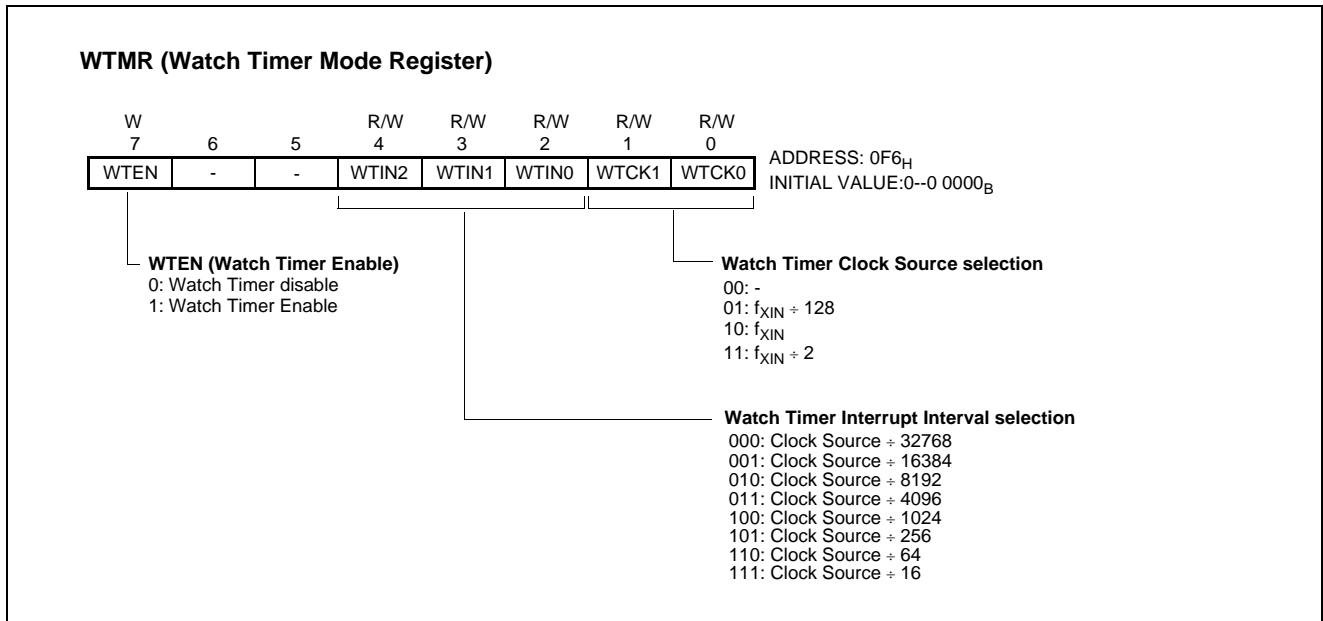


Figure 13-1 Watch Timer Mode Register

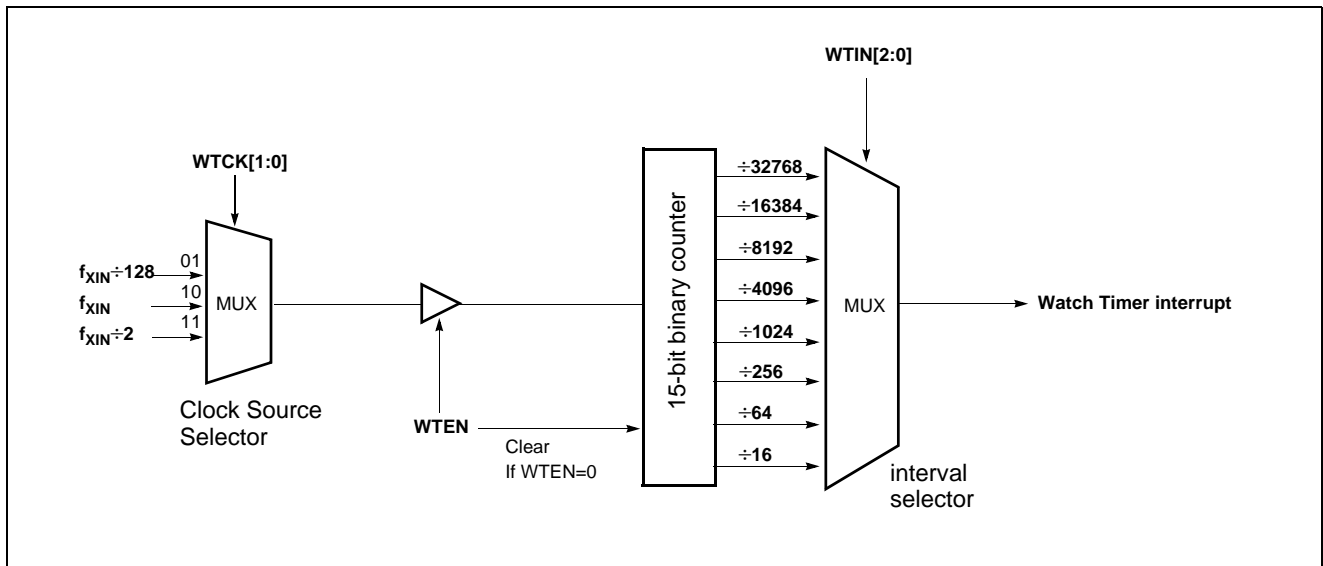


Figure 13-2 Watch Timer Block Diagram

14. TIMER/EVENT COUNTER

The MC80F0208/16/24 has five 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 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 has four operating modes: "8-bit timer/counter", "16-bit timer/counter", "8-bit capture" and "16-bit capture" which are selected by bit in Timer mode register TM0 and TM1 as shown in Table 14-1, Figure 14-1.

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 14-2, Figure 14-2.

Timer 4 has two operating modes: "16-bit timer/counter" and "16-bit capture" which are selected by bit in Timer mode register TM4 as shown in Table 14-3, and Figure 14-3.

| 16BIT | CAP0 | CAP1 | T0CK [2:0] | T1CK [1:0] | TIMER 0 | TIMER 1 |
|-------|------|------|------------|------------|---------------------------------|---------------|
| 0 | 0 | 0 | XXX | XX | 8-bit Timer | 8-bit Timer |
| 0 | 0 | 1 | 111 | XX | 8-bit Event counter | 8-bit Capture |
| 0 | 1 | 0 | XXX | XX | 8-bit Capture (internal clock) | 8-bit Timer |
| 1 | 0 | 0 | XXX | 11 | 16-bit Timer | |
| 1 | 0 | 0 | 111 | 11 | 16-bit Event counter | |
| 1 | 1 | 1 | XXX | 11 | 16-bit Capture (internal clock) | |

Table 14-1 Operating 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 14-2 Operating Modes of Timer 2, 3

| CAP4 | T4CK[2:0] | TIMER 4 |
|-------------|------------------|---------------------------------|
| 0 | XXX | 16-bit Timer |
| 1 | XXX | 16-bit Capture (internal clock) |

Table 14-3 Operating Modes of Timer 4

TM0

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

TM1

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

ADDRESS: 0D2_H
INITIAL VALUE: -0-0 0000_B

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

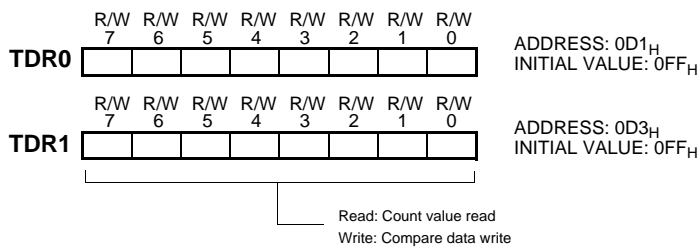


Figure 14-1 TM0, TM1 Registers

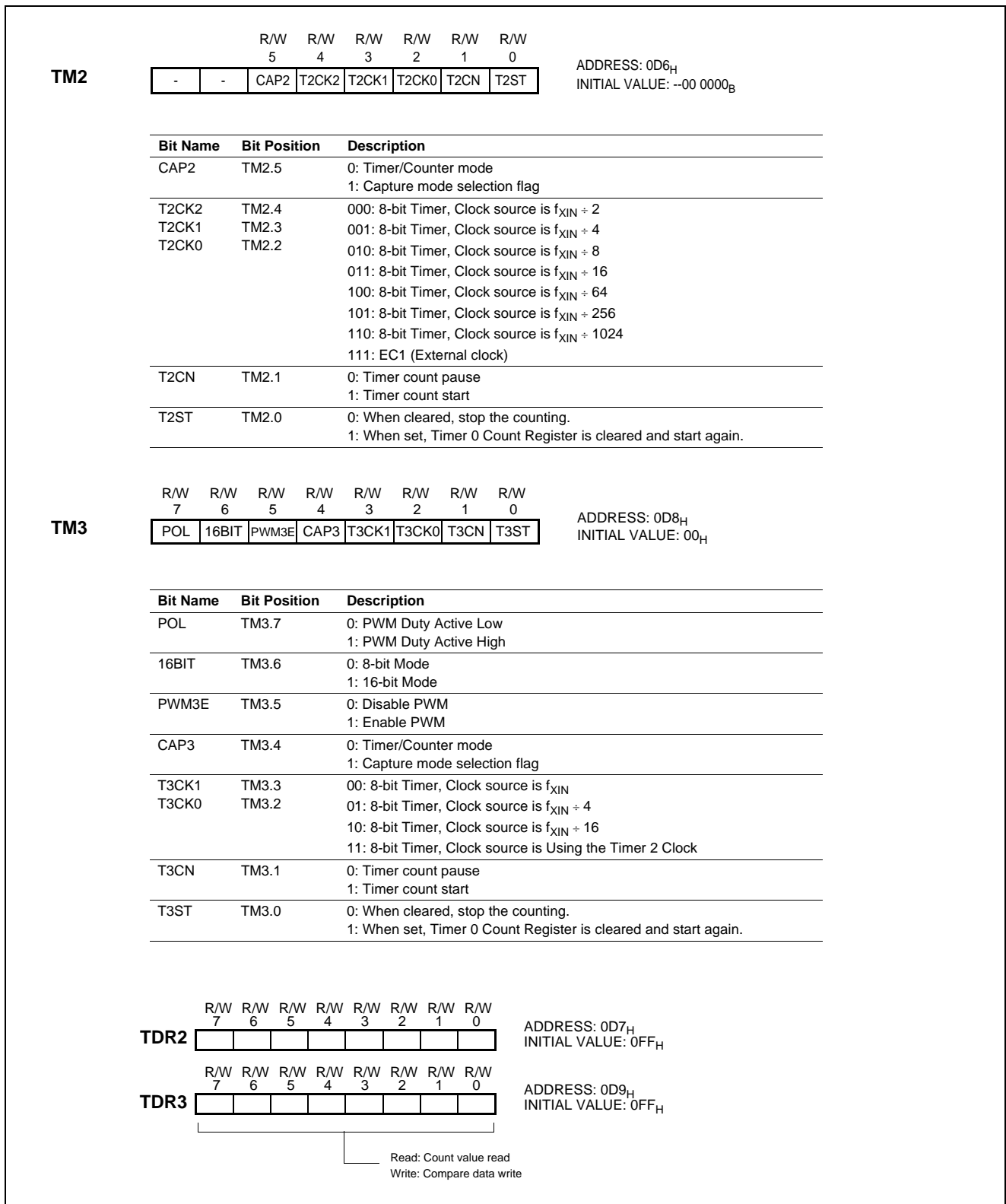


Figure 14-2 TM2, TM3 Registers

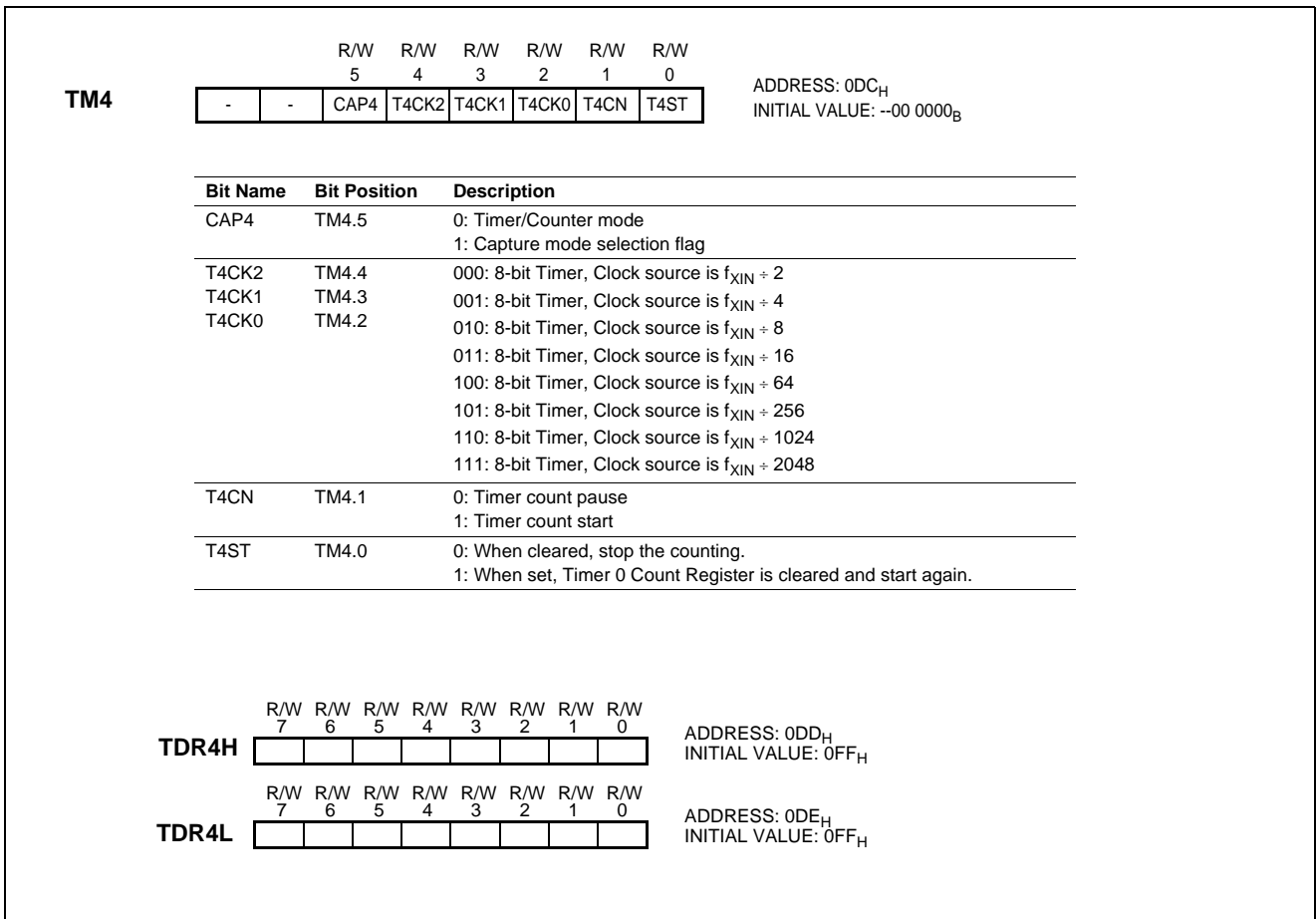


Figure 14-3 TM4 Register

14.1 8-bit Timer / Counter Mode

The MC80F0208/16/24 has four 8-bit Timer/Counters, Timer 0, Timer 1, Timer 2, Timer 3. The Timer 0, Timer 1 are shown in Figure 14-4 and Timer 2, Timer 3 are shown in Figure 14-5.

The “timer” or “counter” function is selected by control registers TM0, TM1, TM2, TM3 as shown in Figure 14-1. To use as an 8-bit timer/counter mode, bit CAP0, CAP1, CAP2, or CAP3 of TMx should be cleared to “0” and 16BIT of TM1 or TM3 should

be cleared to "0"(Figure 14-4). 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).

| | | | | | | | | | |
|------------|---|---|------|-------|-------|-------|------|------|--|
| TM0 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | ADDRESS: 0D0 _H INITIAL VALUE: --00 0000 _B |
| | - | - | CAP0 | T0CK2 | T0CK1 | T0CK0 | T0CN | T0ST | |
| | - | - | 0 | X | X | X | X | X | X means don't care |

| | | | | | | | | | |
|------------|---|-------|---|------|-------|-------|------|------|--|
| TM1 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | ADDRESS: 0D2 _H INITIAL VALUE: -0-0 0000 _B |
| | - | 16BIT | - | CAP1 | T1CK1 | T1CK0 | T1CN | T1ST | |
| | - | 0 | - | 0 | X | X | X | X | X means don't care |

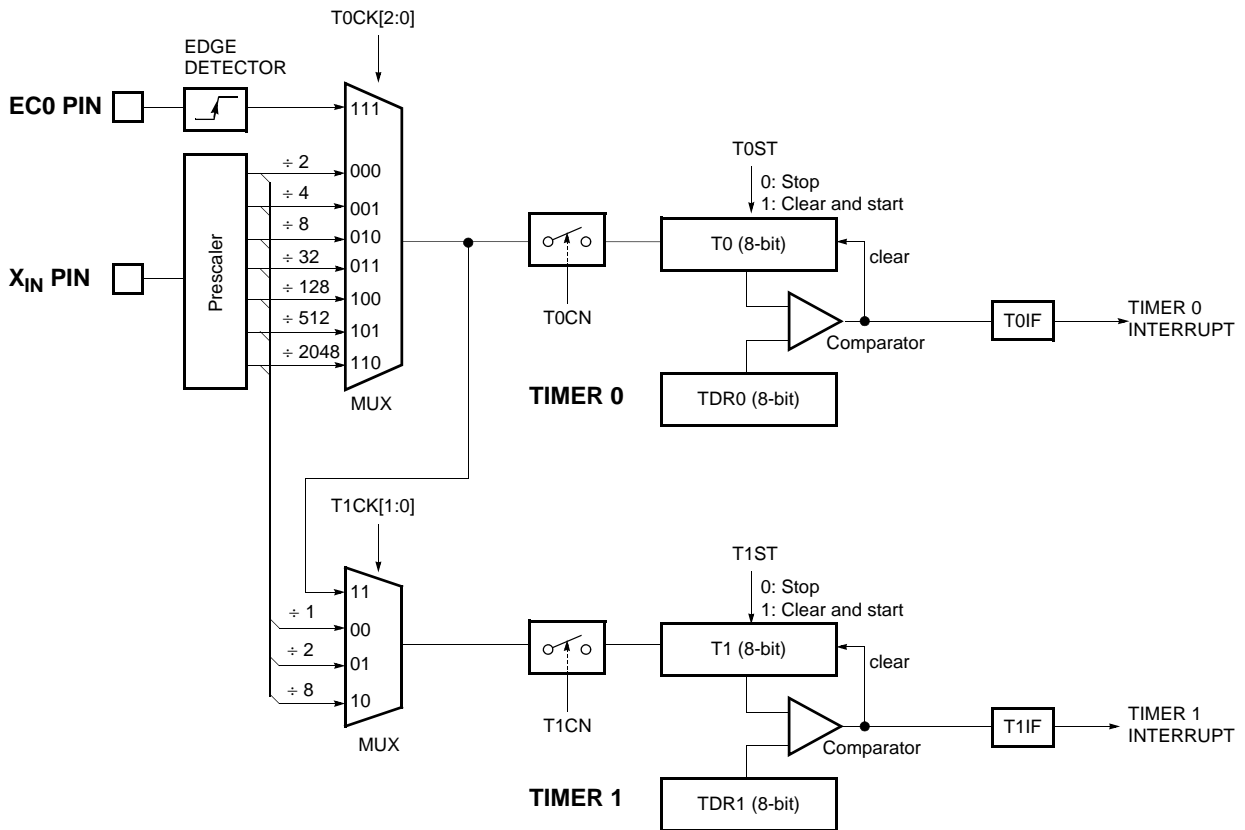


Figure 14-4 8-bit Timer/Counter 0, 1

| | | | | | | | | | |
|------------|---|---|------|-------|-------|-------|------|------|---|
| TM2 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | ADDRESS: 0D6 _H INITIAL VALUE: --000000 _B |
| | - | - | CAP2 | T2CK2 | T2CK1 | T2CK0 | T2CN | T2ST | |
| | - | - | 0 | X | X | X | X | X | |

X means don't care

| | | | | | | | | | |
|------------|-----|-------|-------|------|-------|-------|------|------|---|
| TM3 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | ADDRESS: 0D8 _H INITIAL VALUE: 00 _H |
| | POL | 16BIT | PWM3E | CAP3 | T3CK1 | T3CK0 | T3CN | T3ST | |
| | X | 0 | 0 | 0 | X | X | X | X | |

X means don't care

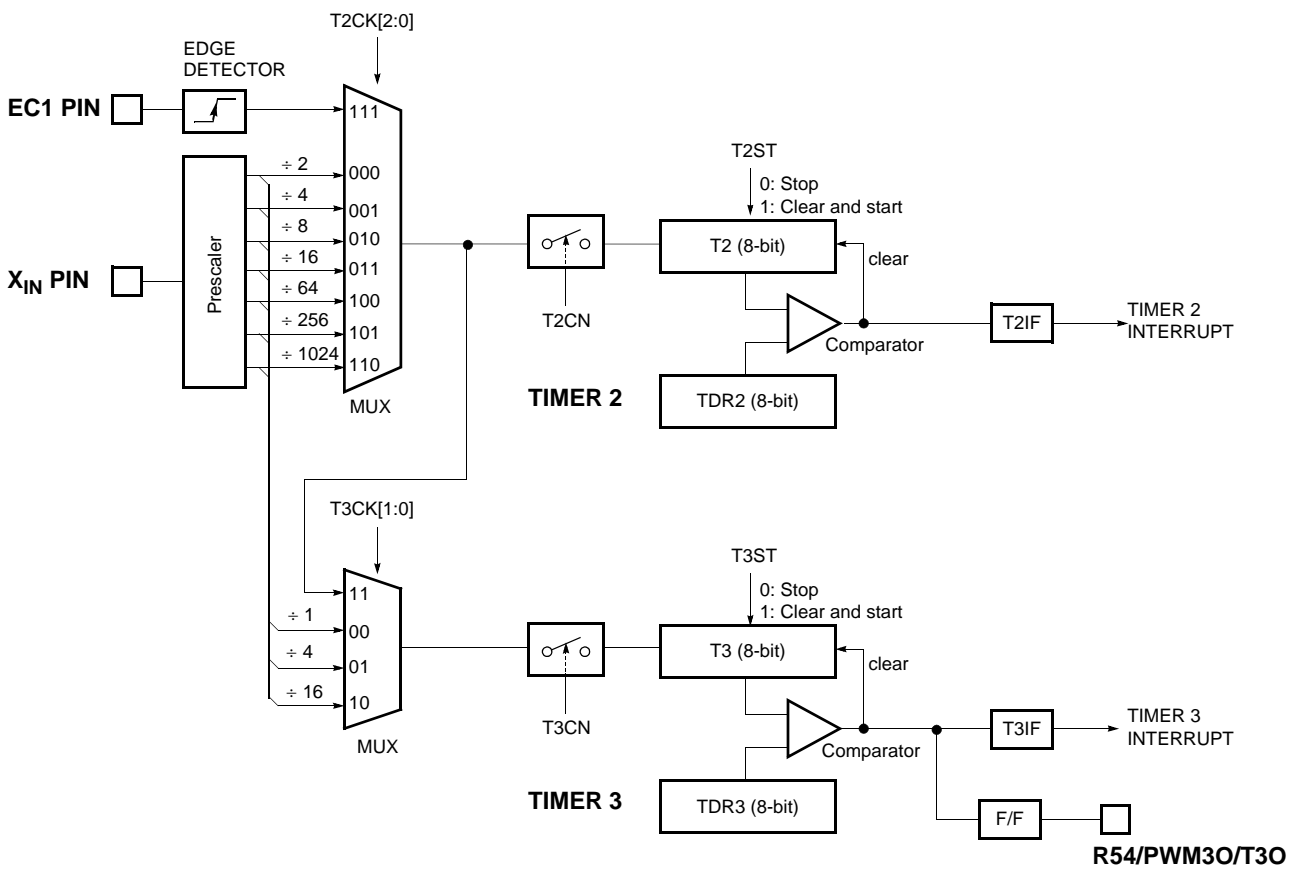


Figure 14-5 8-bit Timer/Counter 2, 3

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(1-to-0) (rising & falling 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.

14.1.1 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_nIF) 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.

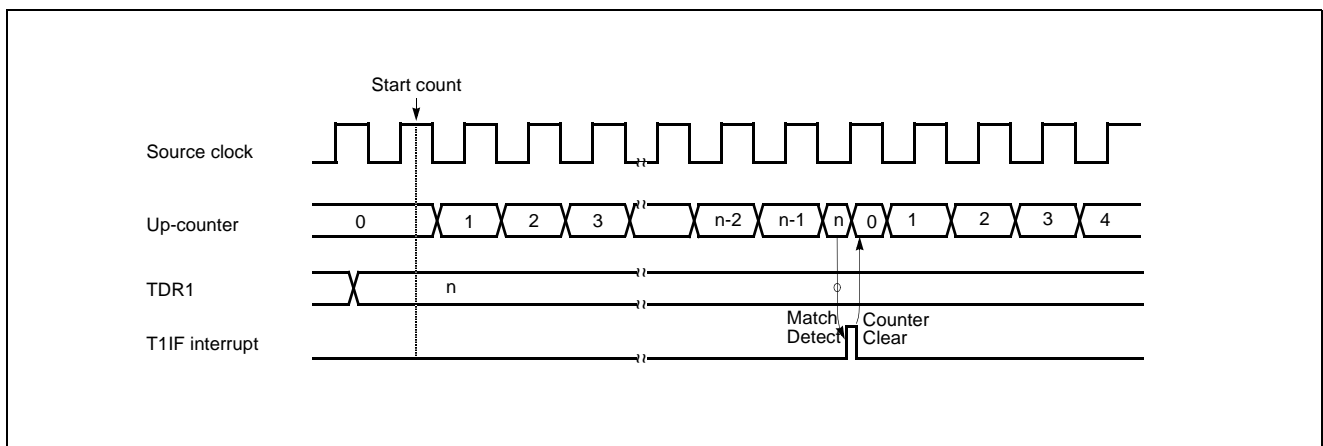


Figure 14-6 Timer Mode Timing Chart

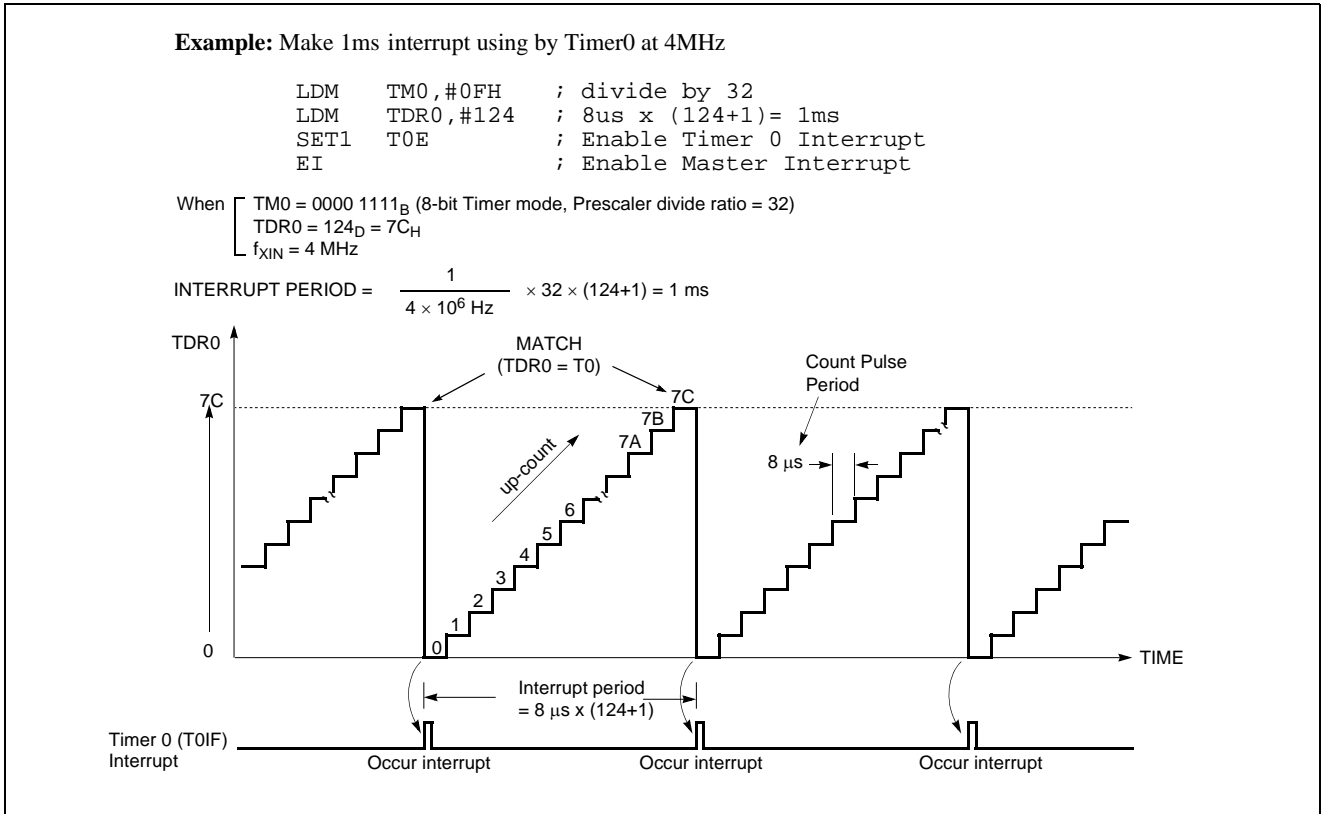


Figure 14-7 Timer Count Example

14.1.2 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 TDRn (n = 0,1,2,3) are compared with the contents of the up-counter Tn. If a match is found, a timer interrupt request flag TnIF 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 0F8H) is required to be set to “1”.

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

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

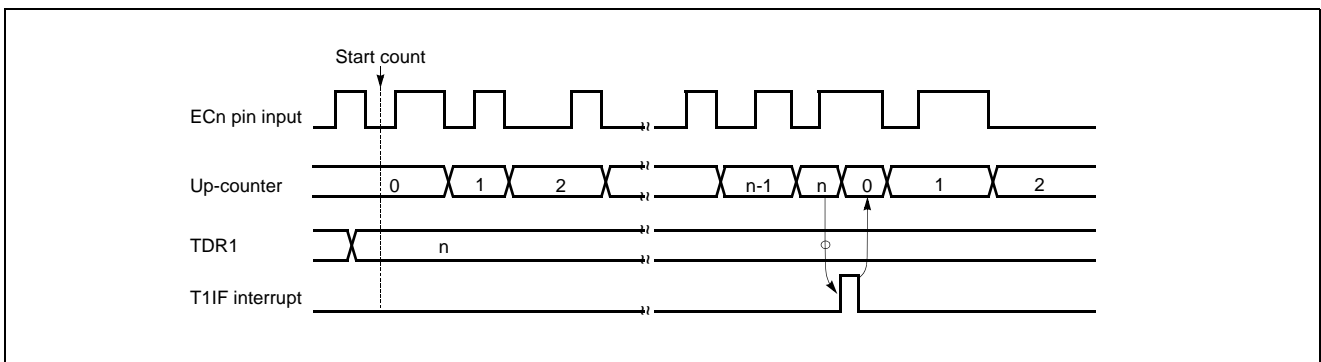


Figure 14-8 Event Counter Mode Timing Chart

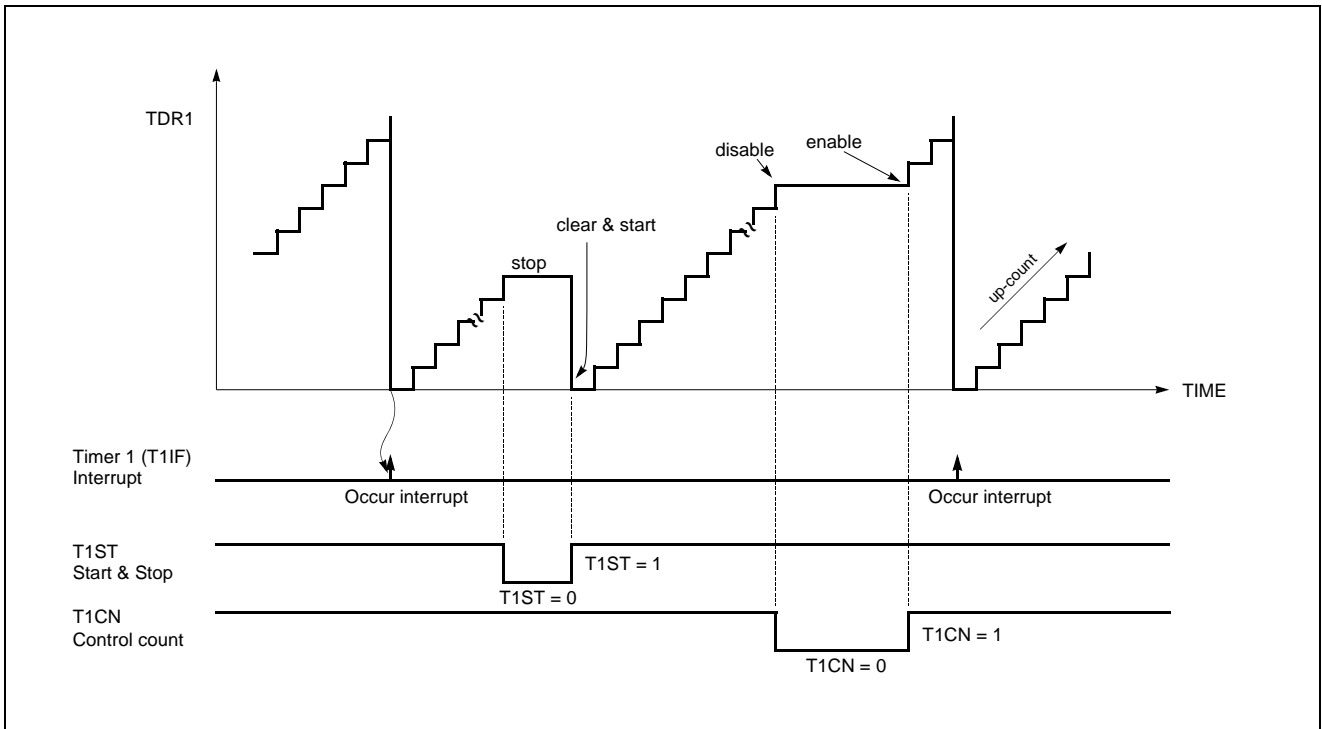


Figure 14-9 Count Operation of Timer / Event counter

14.2 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 14-10.

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 14-11.

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 TM2. 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.

A 16-bit timer/counter 4 register T4H, T4L are increased from 0000_H until it matches TDR4H, TDR4L and then resets to 0000_H. The match output generates Timer 4 interrupt. Timer/Counter 4 is 16 bit mode as shown in Figure 14-12.

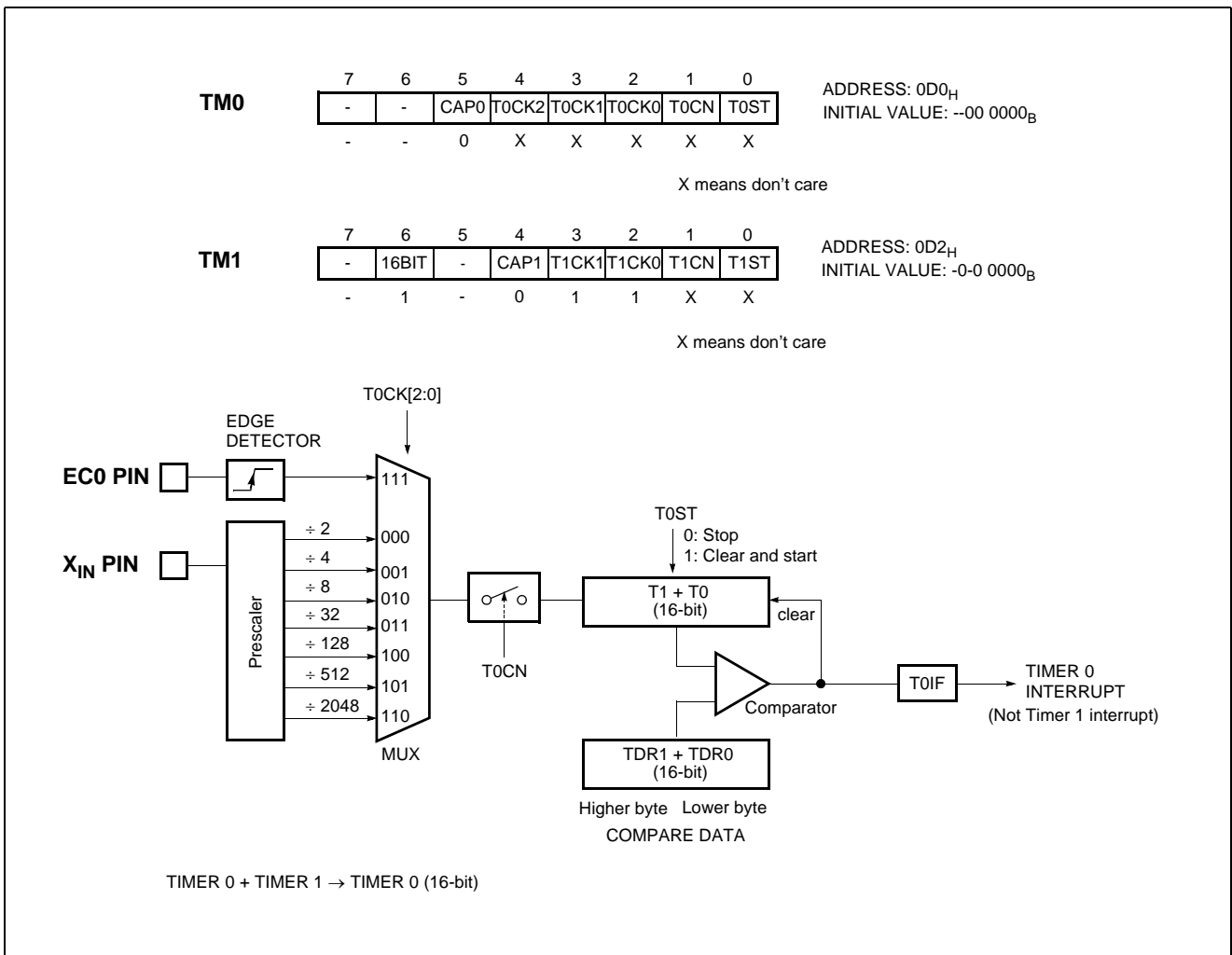


Figure 14-10 16-bit Timer/Counter for Timer 0, 1

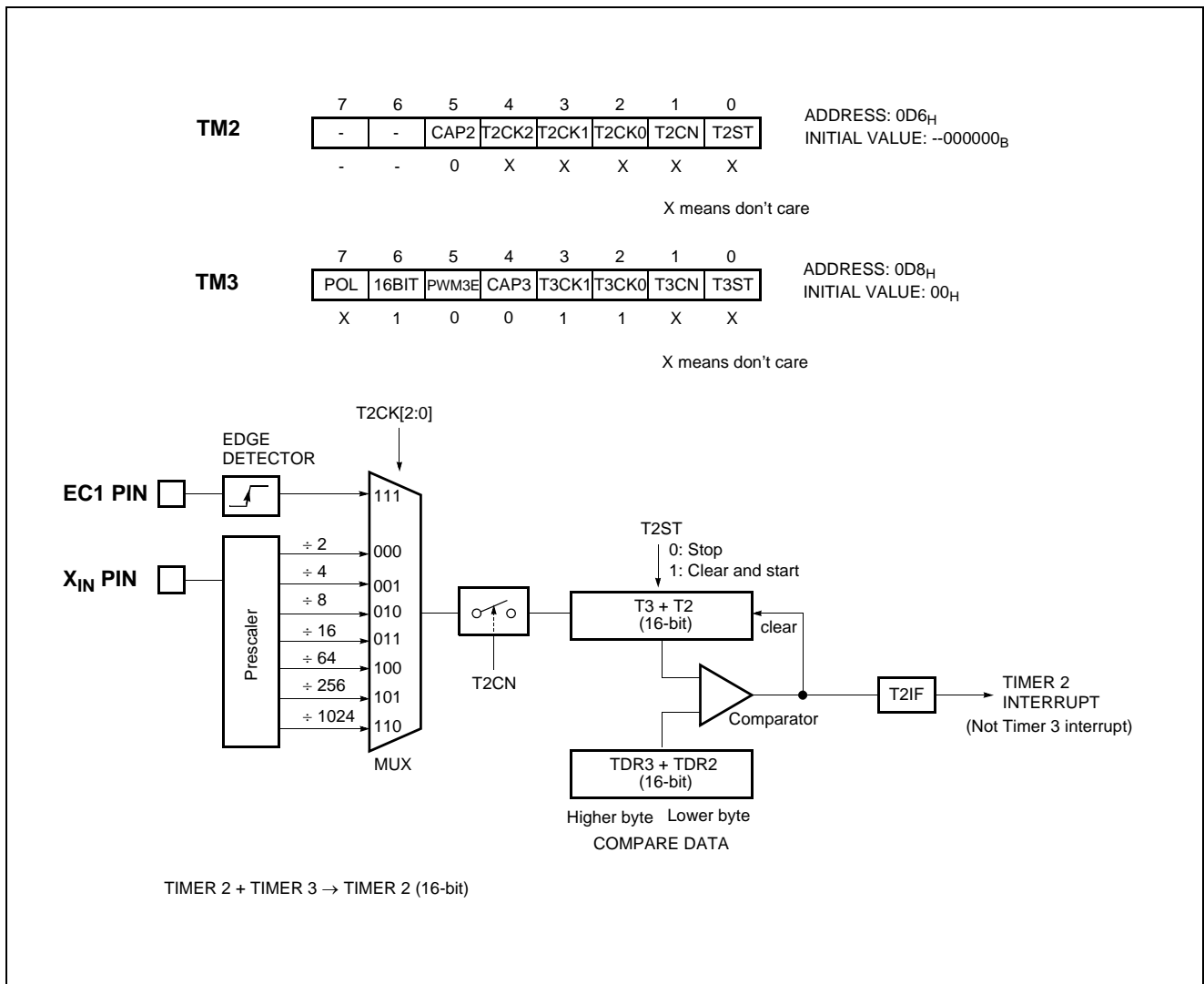


Figure 14-11 16-bit Timer/Counter for Timer 2, 3

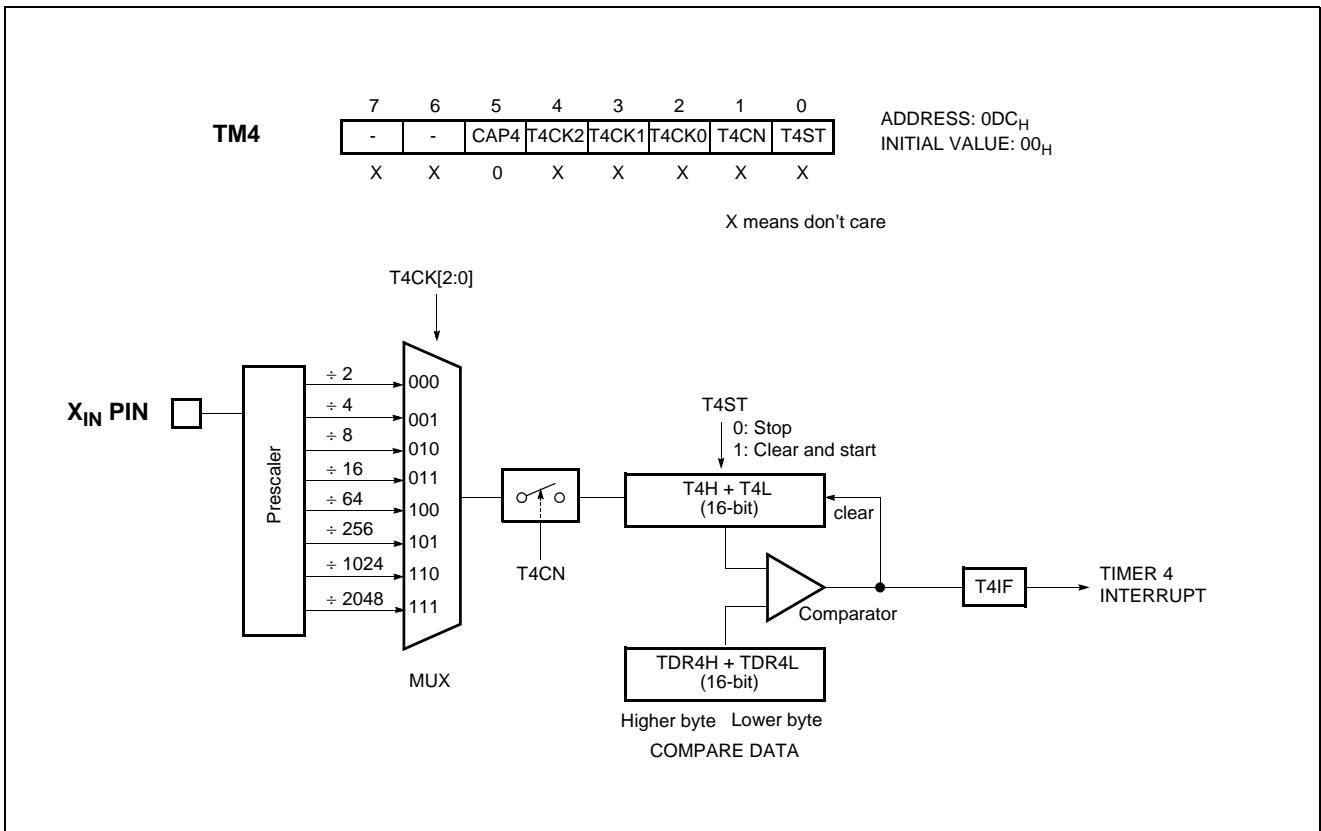
14.3 8-bit Compare Output (16-bit)

The MC80F0208/16/24 has a function of Timer Compare Output. To pulse out, the timer match can go to port pin (T3O) as shown in Figure 14-5. Thus, pulse out is generated by the timer match. These operations are implemented to pin, PWM3O/T3O.

In this mode, the bit PWM3O/T3O of R5 Port Selection register0 (PSR0.7) should be set to "1", and the bit PWM3E of timer3 mode register (TM3) should be set to "0". This pin outputs the sig-

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

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



14.4 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 14-13. 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 14-14.

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 14-16, 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 INT_x pin causes the current value in the Timer x register (T0,T1,T2,T3), to be captured into registers CDR_x (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 "19.5 External Interrupt" on page 92. In addition, the transition at INT_n pin generate an interrupt.

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

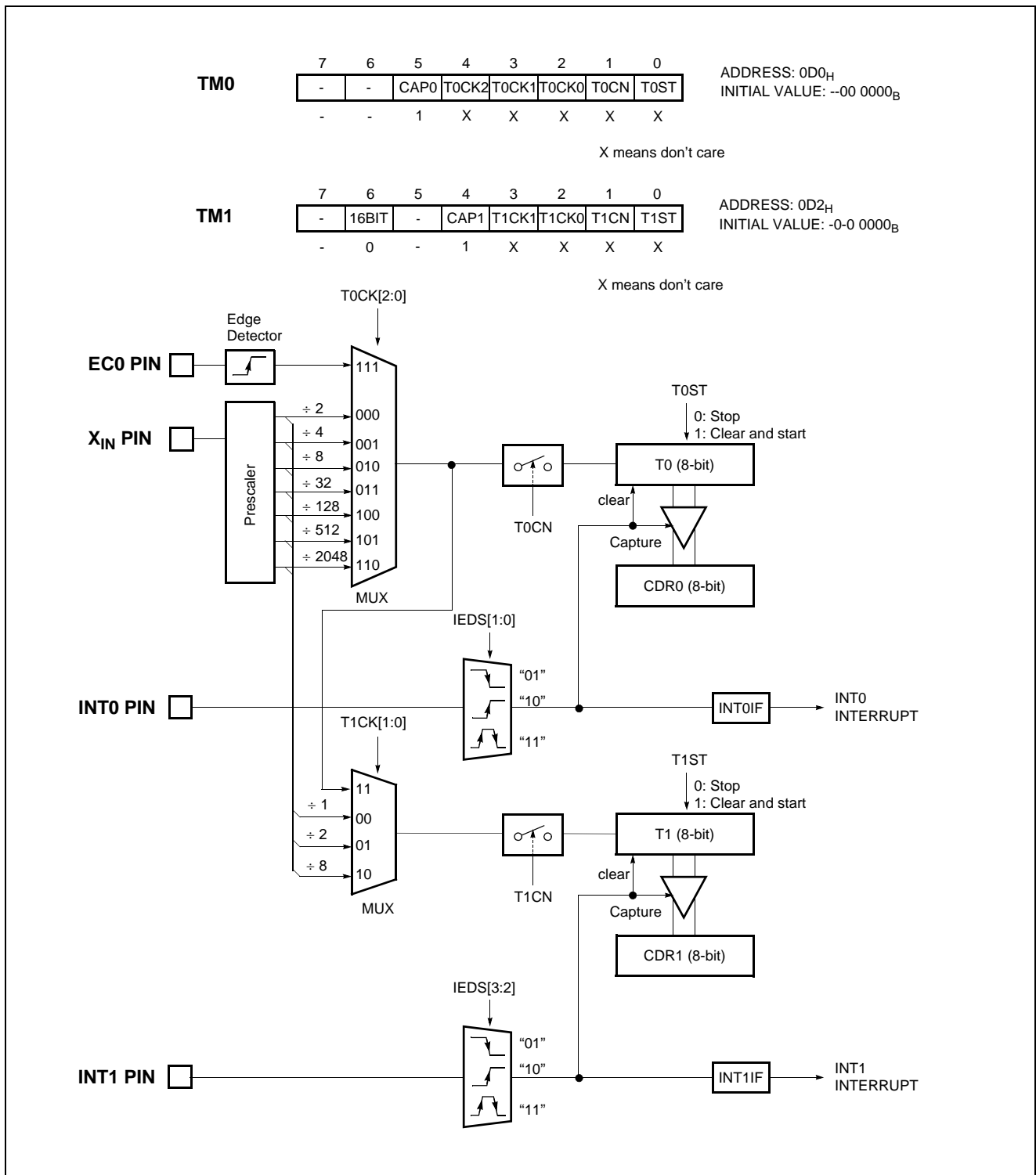


Figure 14-13 8-bit Capture Mode for Timer 0, 1

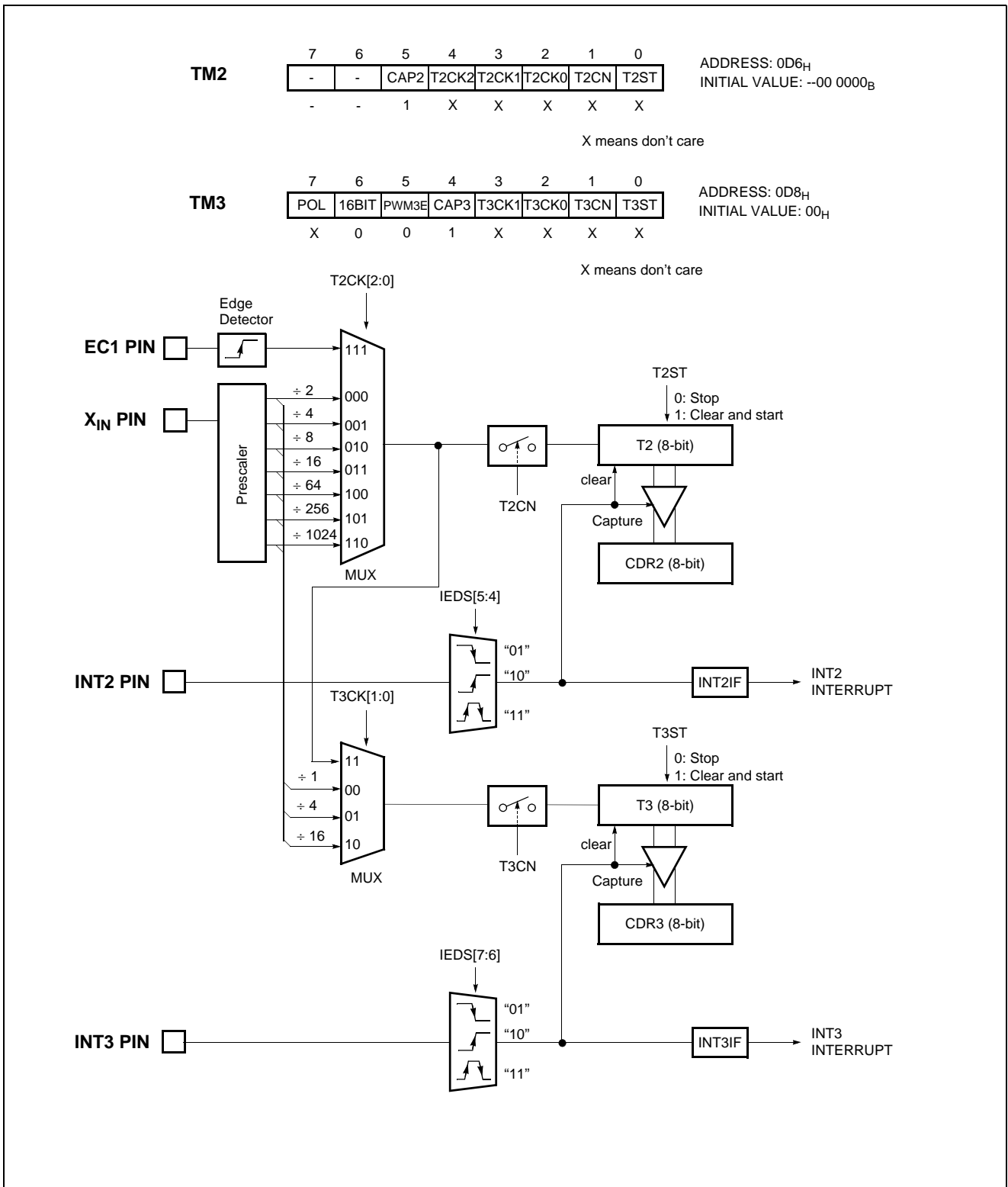


Figure 14-14 8-bit Capture Mode for Timer 2, 3

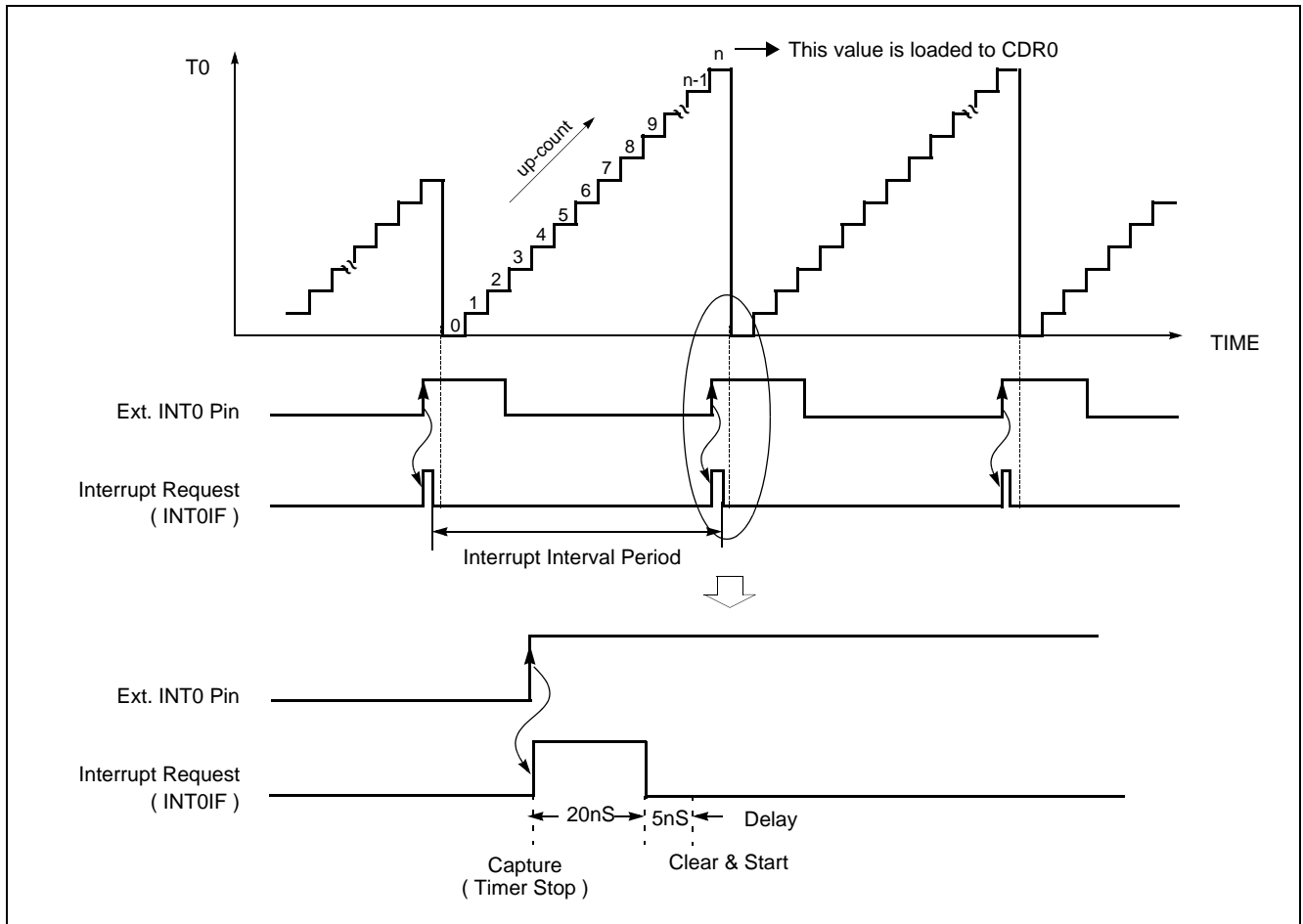


Figure 14-15 Input Capture Operation of Timer 0 Capture mode

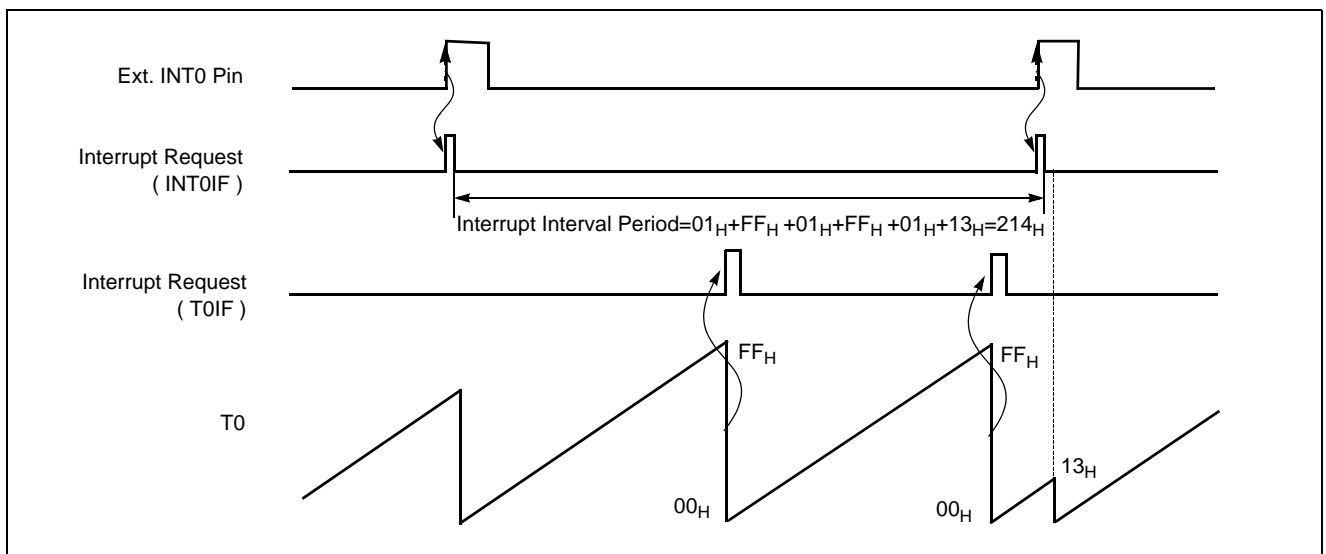


Figure 14-16 Excess Timer Overflow in Capture Mode

14.5 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 14-17.

ternal 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 14-18.

The clock source of the Timer 4 is selected either internal or external clock by bit T4CK[2:0] as shown in Figure 14-18.

The clock source of the Timer 2 is selected either internal or ex-

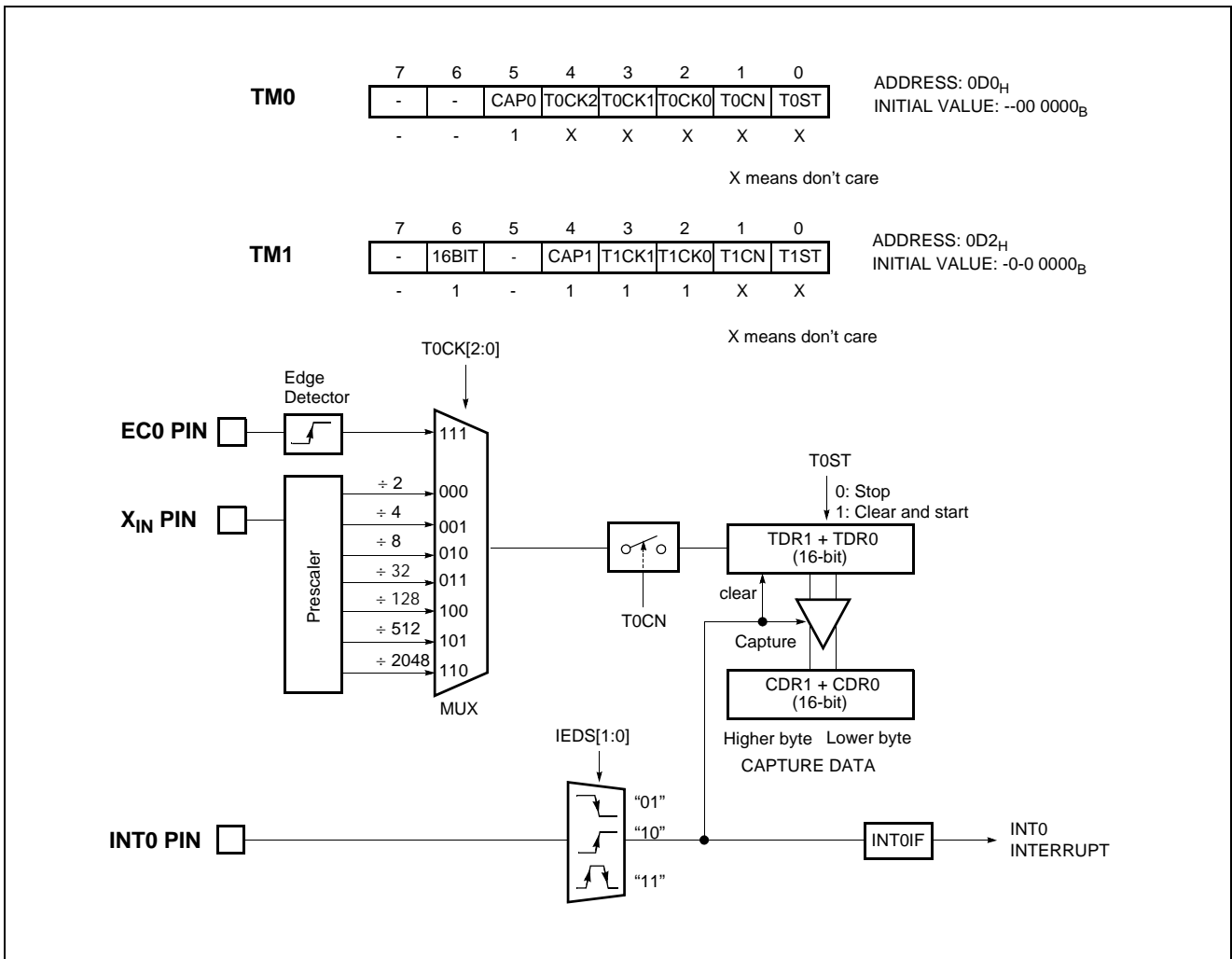


Figure 14-17 16-bit Capture Mode of Timer 0, 1

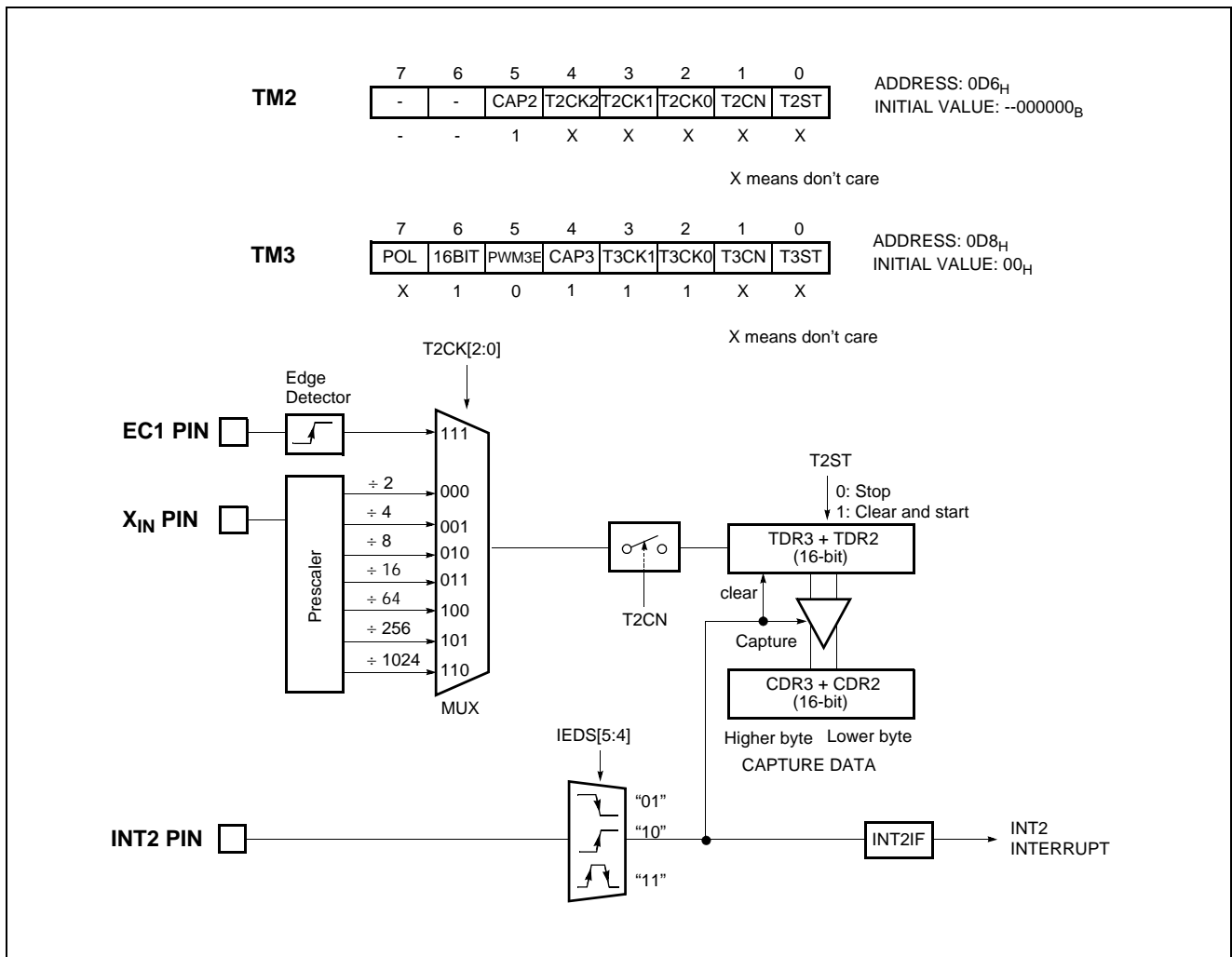


Figure 14-18 16-bit Capture Mode of Timer 2, 3

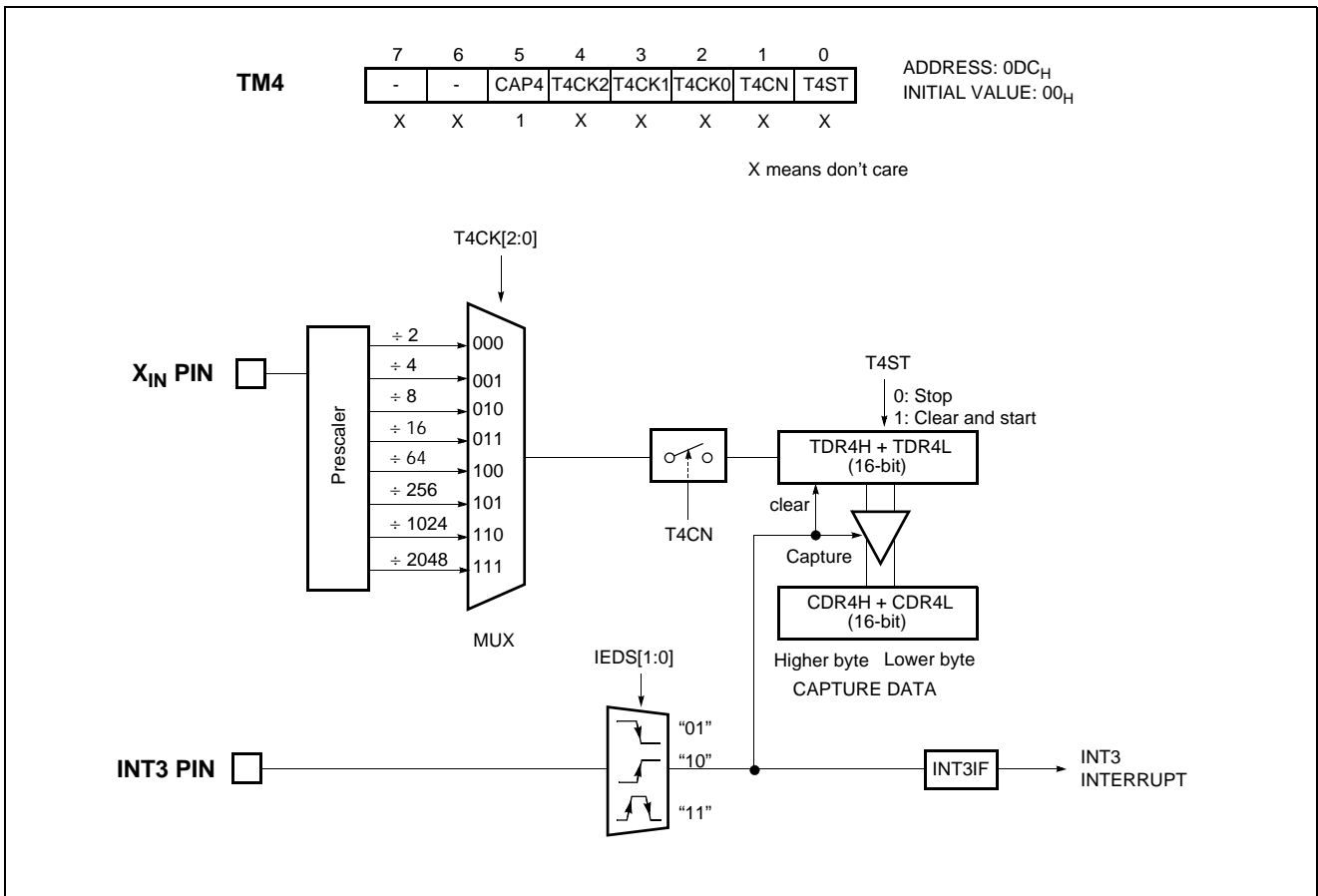


Figure 14-19 16-bit Capture Mode of Timer 4

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
:
:
```

14.6 PWM Mode

The MC80F0208/16/24 has a high speed PWM (Pulse Width Modulation) functions which shared with Timer3.

In PWM mode, pin R54/PWM3O outputs up to a 10-bit resolution PWM output. This pin should be configured as a PWM output by setting "1" bit PWM3O in PSR0 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 T3PPR and the higher 2-bit period value to the T3PWHR[3:2]. And writes duty value to the T3PDR and the T3PWHR[1:0] same way.

The T3PDR is configured as a double buffering for glitchless PWM output. In Figure 14-20, the duty data is transferred from the master to the slave when the period data matched to the count value. (i.e. at the beginning of next duty cycle)

PWM3 Period = [PWM3HR[3:2]T3PPR] X Source Clock

PWM3 Duty = [PWM3HR[1:0]T3PDR] X Source Clock

The relation of frequency and resolution is in inverse proportion. Table 14-4 shows the relation of PWM frequency vs. resolution.

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

| Resolution | Frequency | | |
|------------|--------------------------|------------------------|------------------------|
| | T3CK[1:0] = 00(250nS) | T3CK[1:0] = 01(1uS) | T3CK[1:0] = 10(4uS) |
| 10-bit | 3.9kHz | 1.95kHz | 0.97kHz |
| 9-bit | 7.8kHz | 3.90kHz | 1.95kHz |
| 8-bit | 15.6kHz | 7.81kHz | 3.90kHz |
| 7-bit | 31.2kHz | 15.6kHz | 7.8kHz |

Table 14-4 PWM Frequency vs. Resolution at 4MHz

The bit POL of 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 14-22. 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 Timer3 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 4uS

```
LDM TM3,#1010_1000b ; Set Clock & PWM3E
LDM T3PPR,#199      ; Period :800uS=4uSX(199+1)
LDM T3PDR,#99      ; Duty:400uS=4uSX(99+1)
LDM PWM3HR,00H
LDM TM3,#1010_1011b ; Start timer3
```

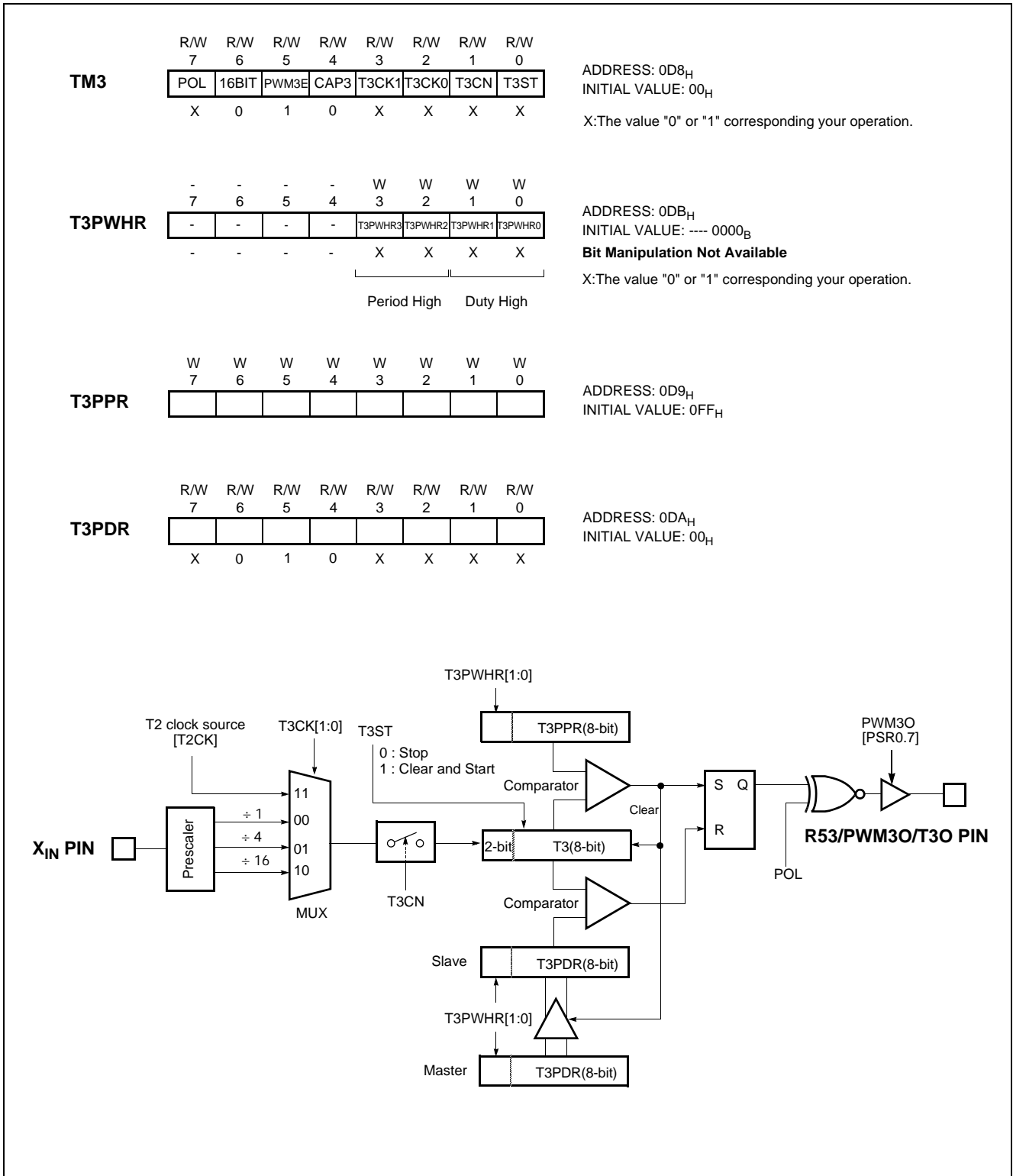


Figure 14-20 PWM3 Mode

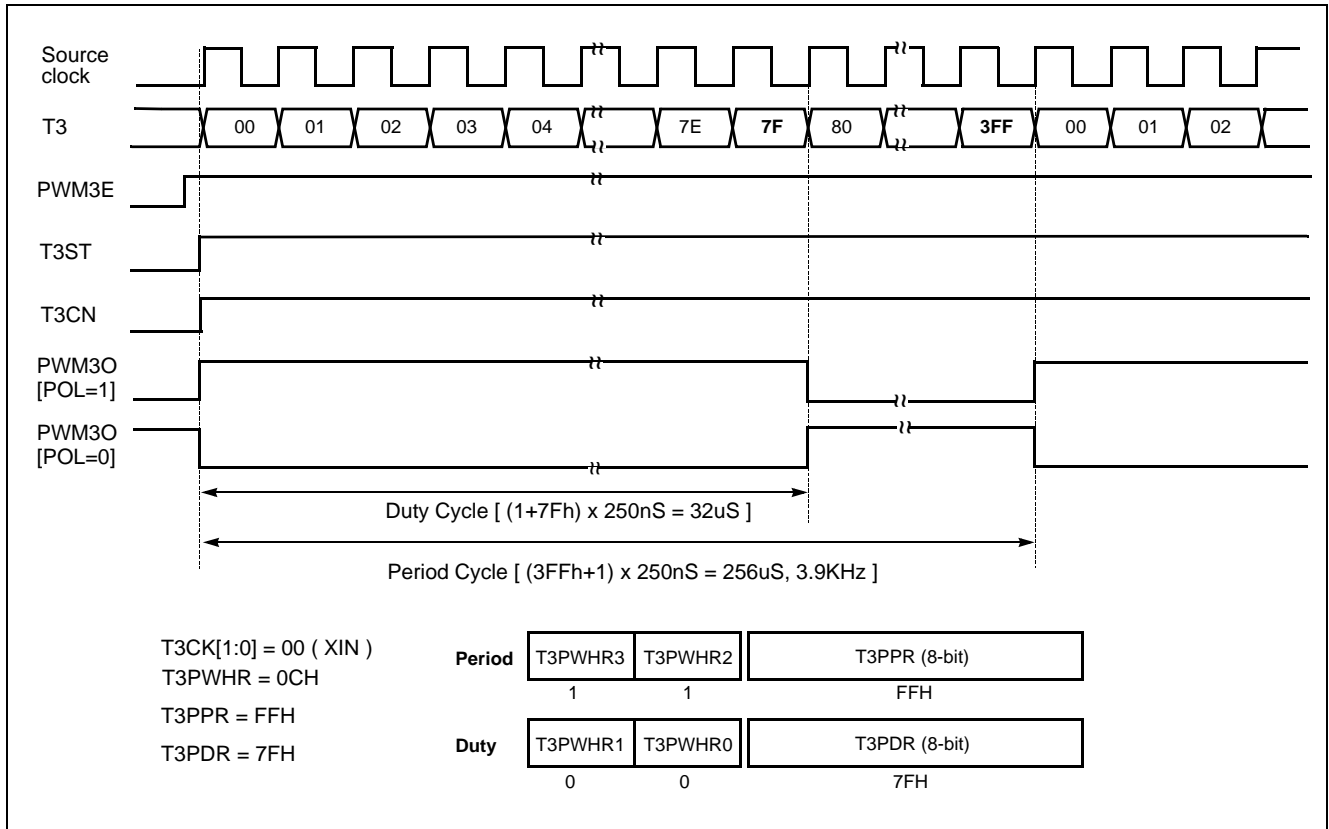


Figure 14-21 Example of PWM at 4MHz

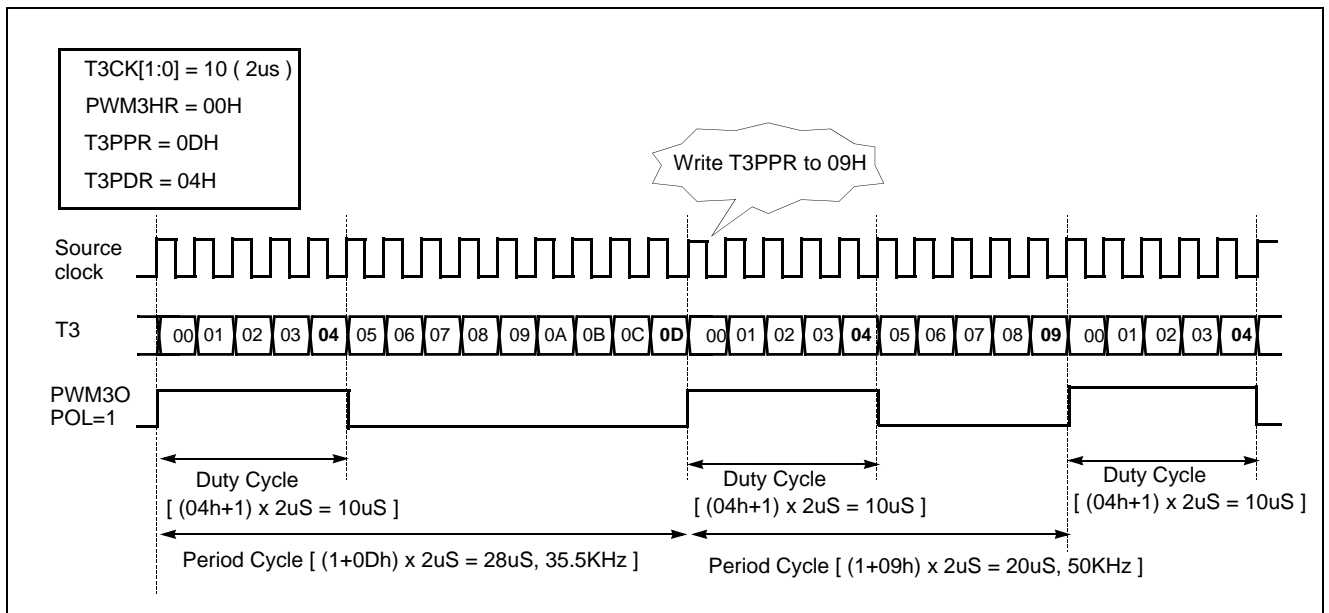


Figure 14-22 Example of Changing the Period in Absolute Duty Cycle (@8MHz)

15. 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 sixteen 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 supply voltage is connected to AV_{DD} of Sample & Hold logic of A/D module. The AV_{DD} was separated with V_{DD} in order to minimize the degradation of operation characteristic by power supply noise.

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 15-4, 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.

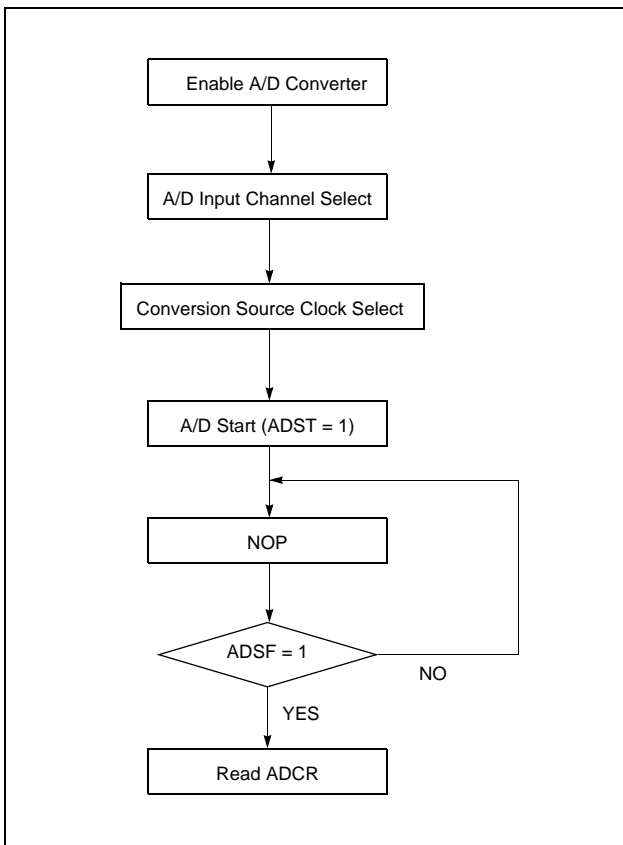


Figure 15-1 A/D Converter Operation Flow

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 15-1 for operation flow.

The block diagram of the A/D module is shown in Figure 15-3. 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 7 times of conversion source clock. The period of actual A/D conversion clock should be minimally 1 μ s

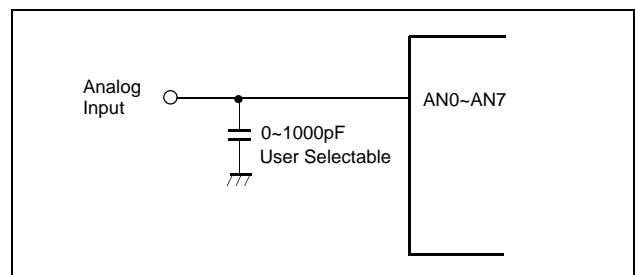


Figure 15-2 Analog Input Pin Connecting Capacitor

A/D Converter Cautions

(1) Input range of AN0 to AN7

The input voltage of AN0 to AN7 should be within the specification range. In particular, if a voltage above AV_{DD} or below AV_{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 AV_{DD} and AN0 to AN7. 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 15-2 in order to reduce noise. The capacitance is user-selectable and appropriately determined according to the target system.

(3) Pins AN0/R60 to AN7/R67

The analog input pins AN0 to AN7 also function as input/output port (PORT R6) pins. When A/D conversion is performed with any of pins AN0 to AN15 selected, be sure not to execute a PORT input instruction while conversion is in progress, as this may reduce the conversion resolution.

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

(4) AV_{DD} pin input impedance

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

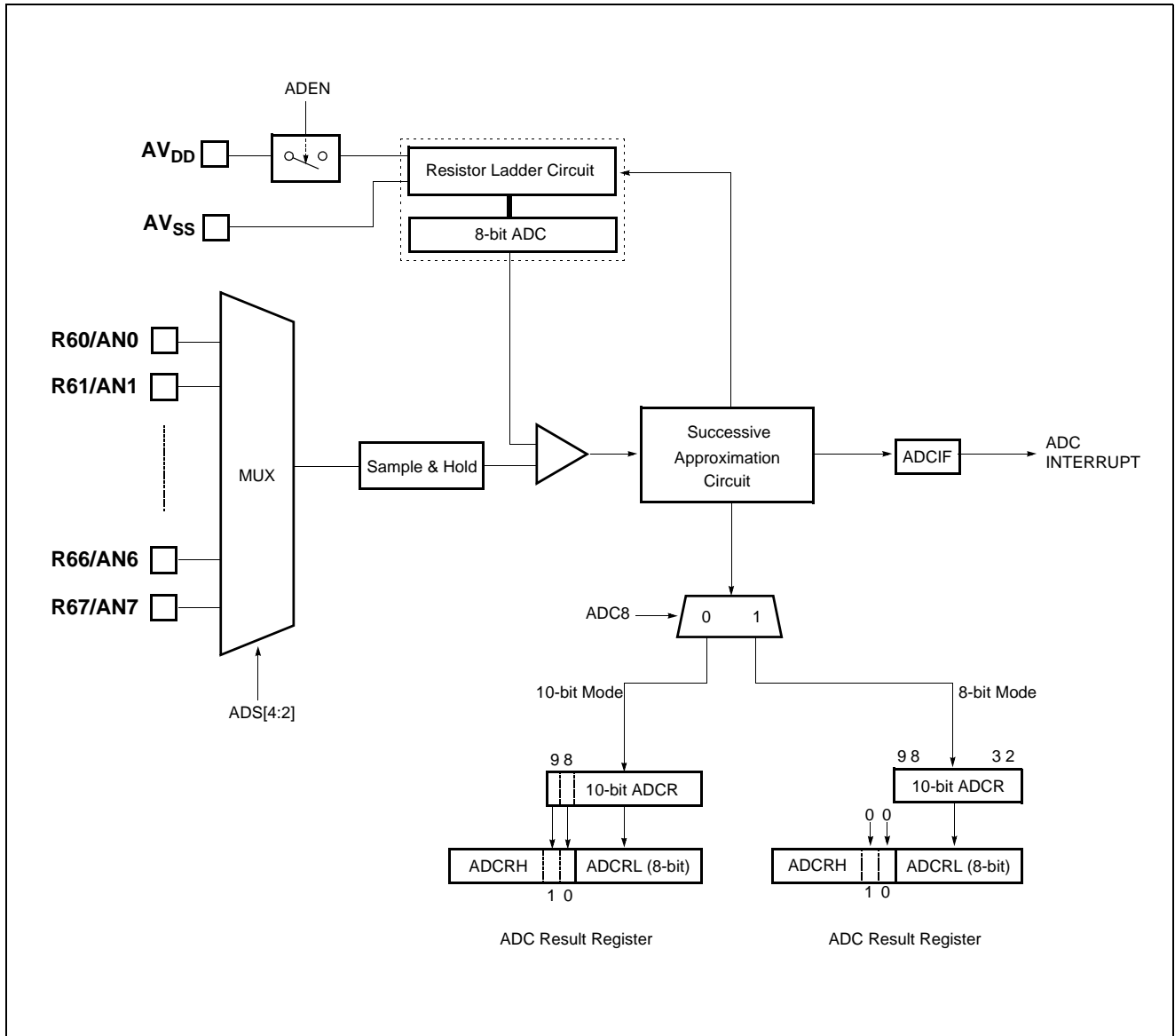


Figure 15-3 A/D Block Diagram

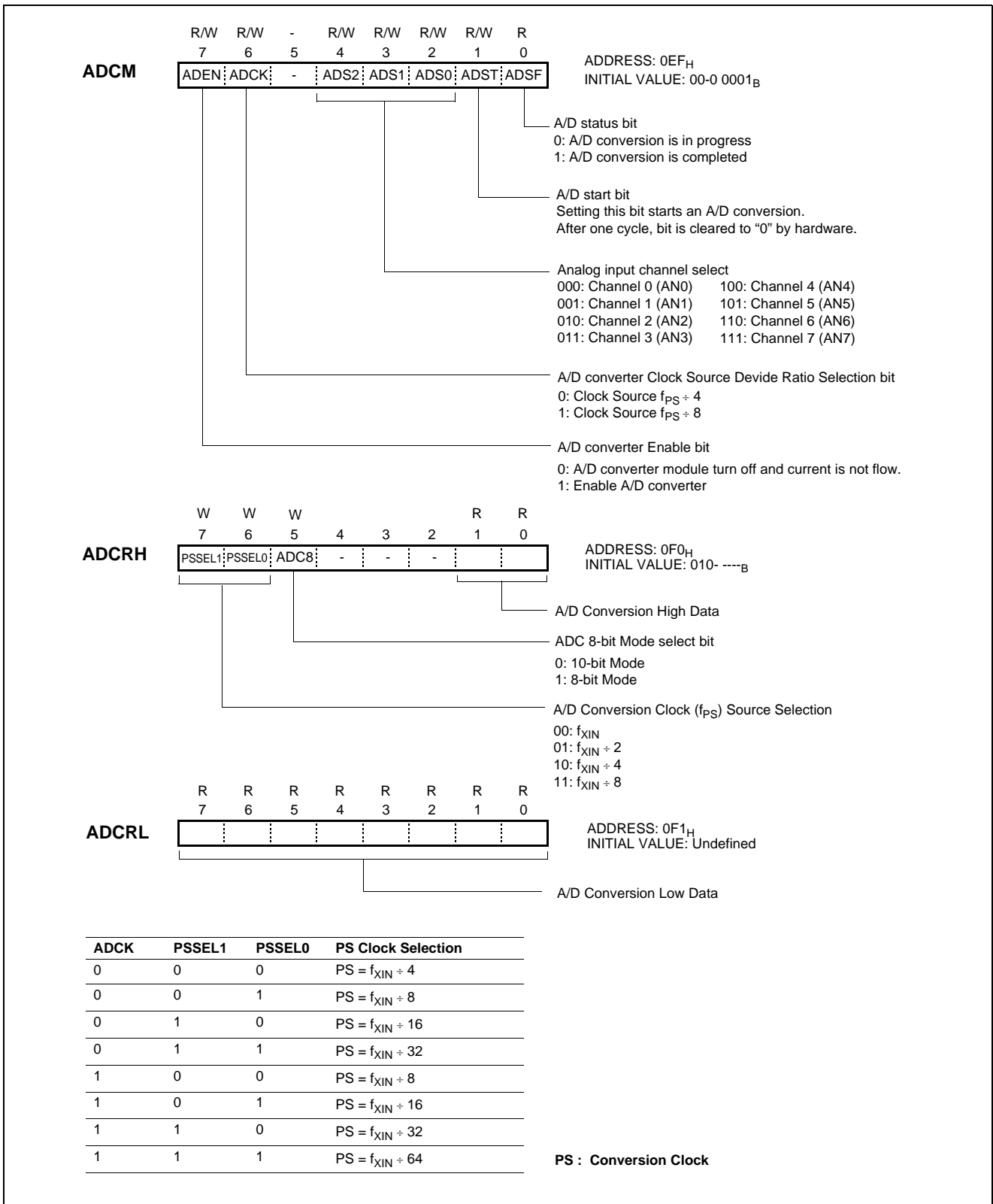


Figure 15-4 A/D Converter Control & Result Register

16. 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 16-1. The SO pin is designed to input and output. So the Serial I/O(SIO) can be operated with minimum two pin. Pin R42/SCK, R43/SI, and R44/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.

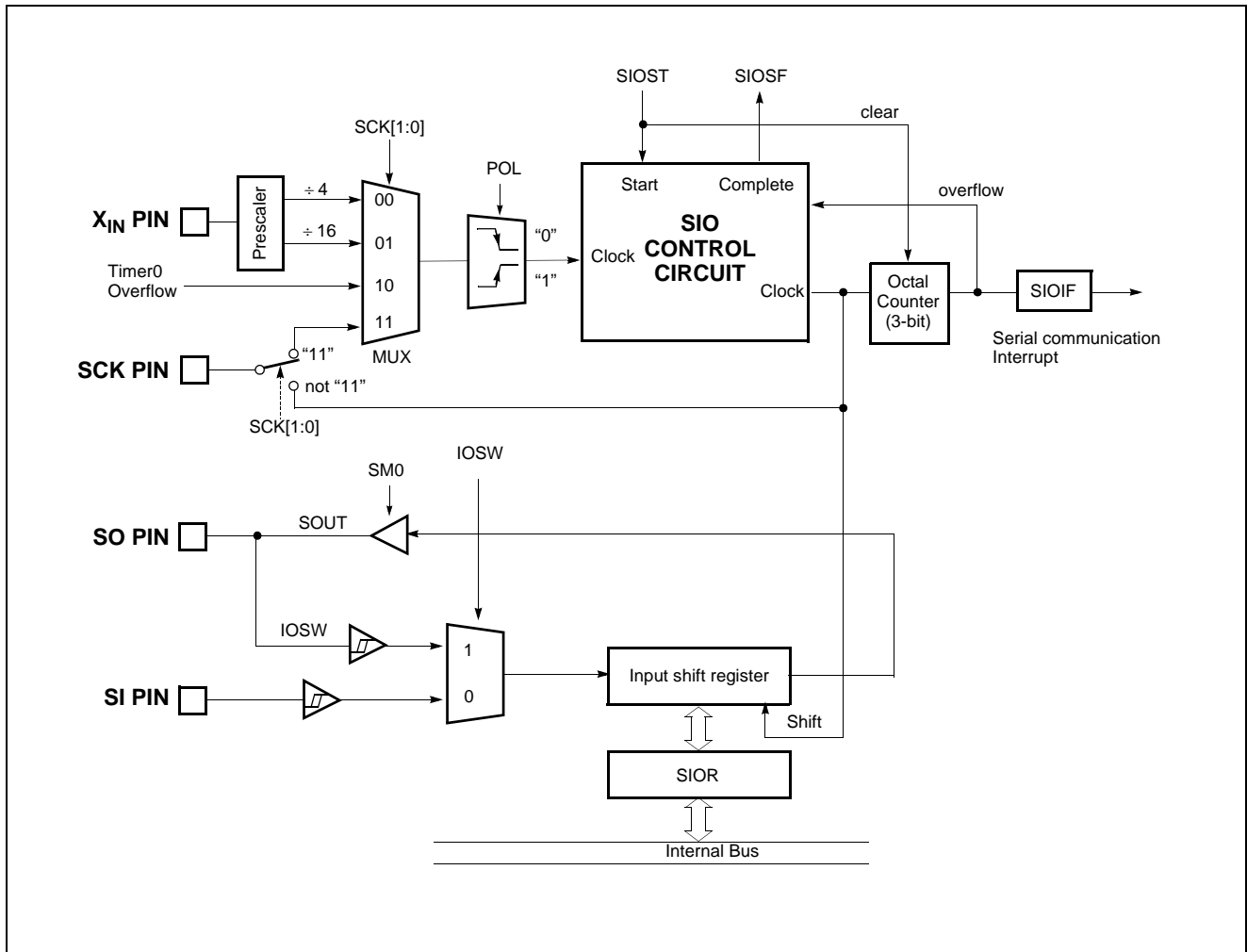


Figure 16-1 SIO Block Diagram

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.

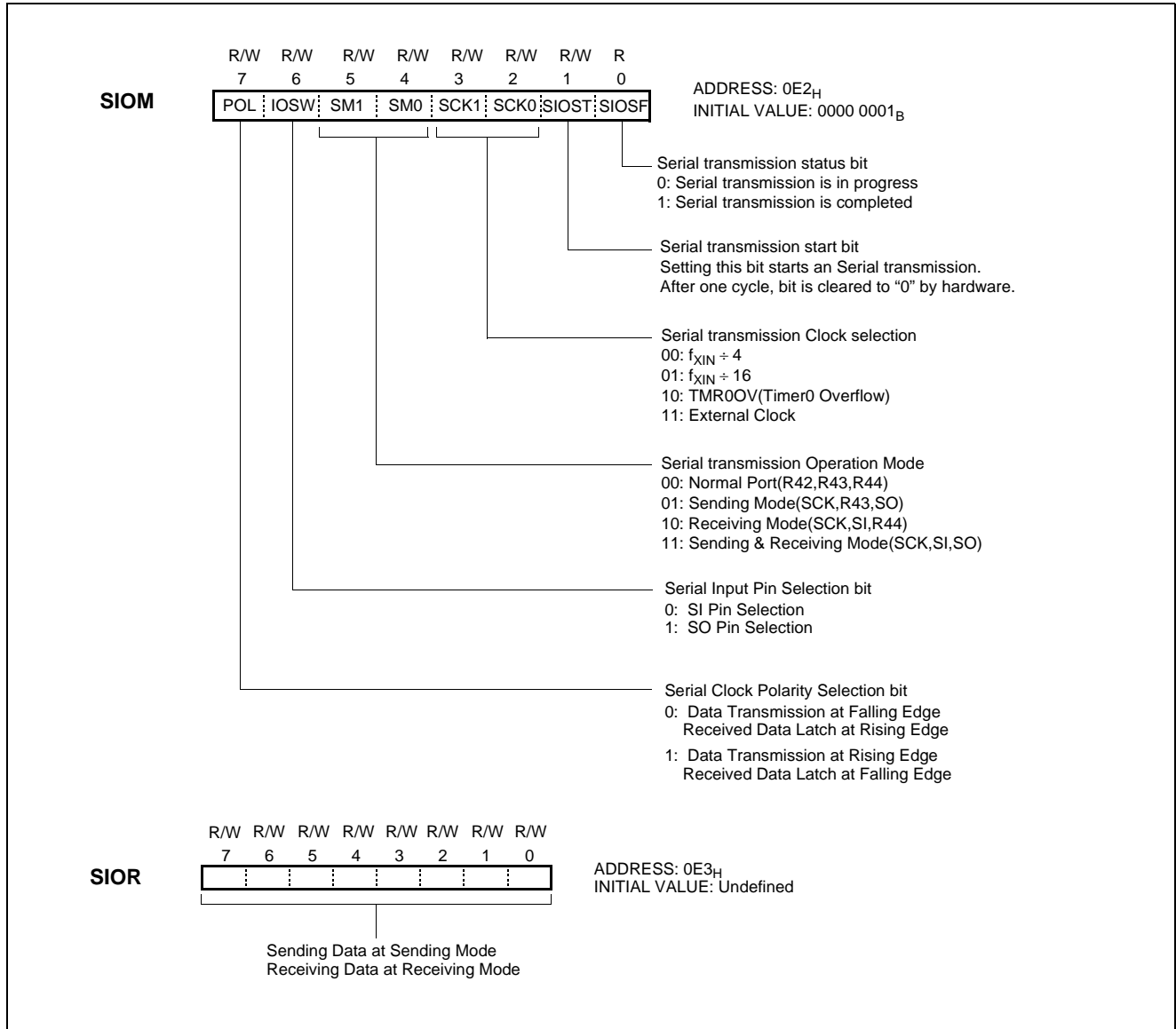


Figure 16-2 SIO Control Register

16.1 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 in-

put data is latched at rising edge of SCLK pin (Refer to Figure 16-3). 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.

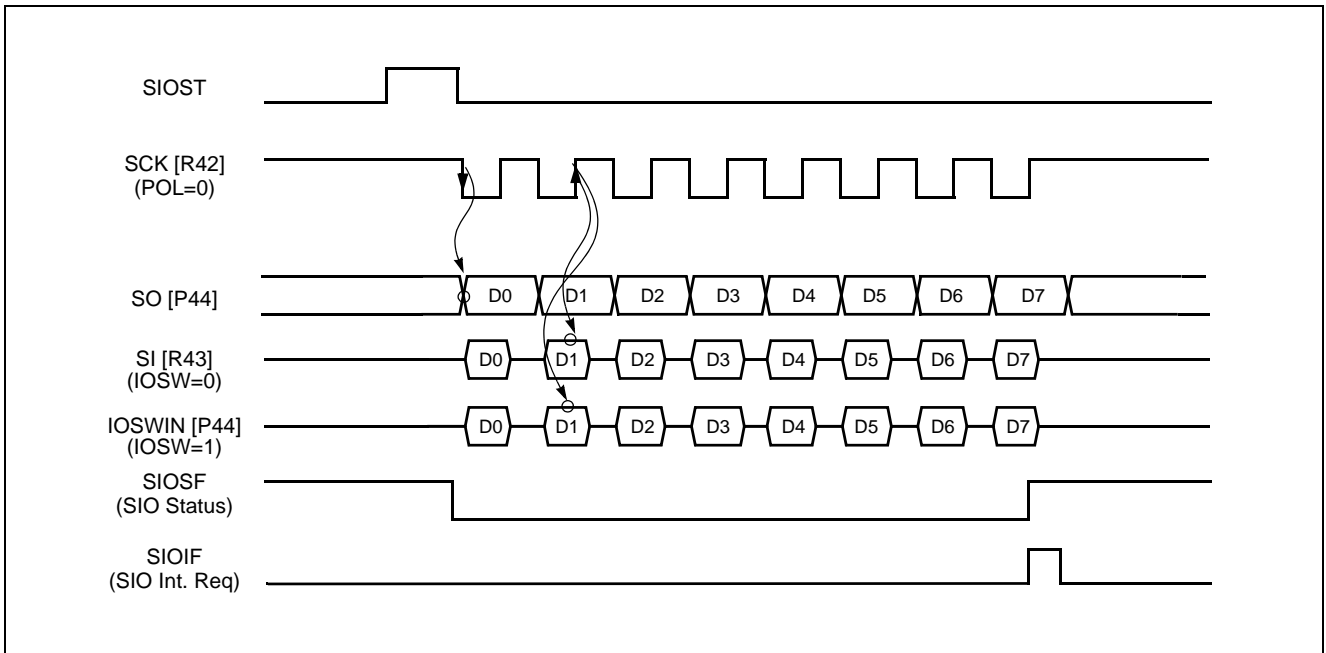


Figure 16-3 Serial I/O Timing Diagram at POL=0

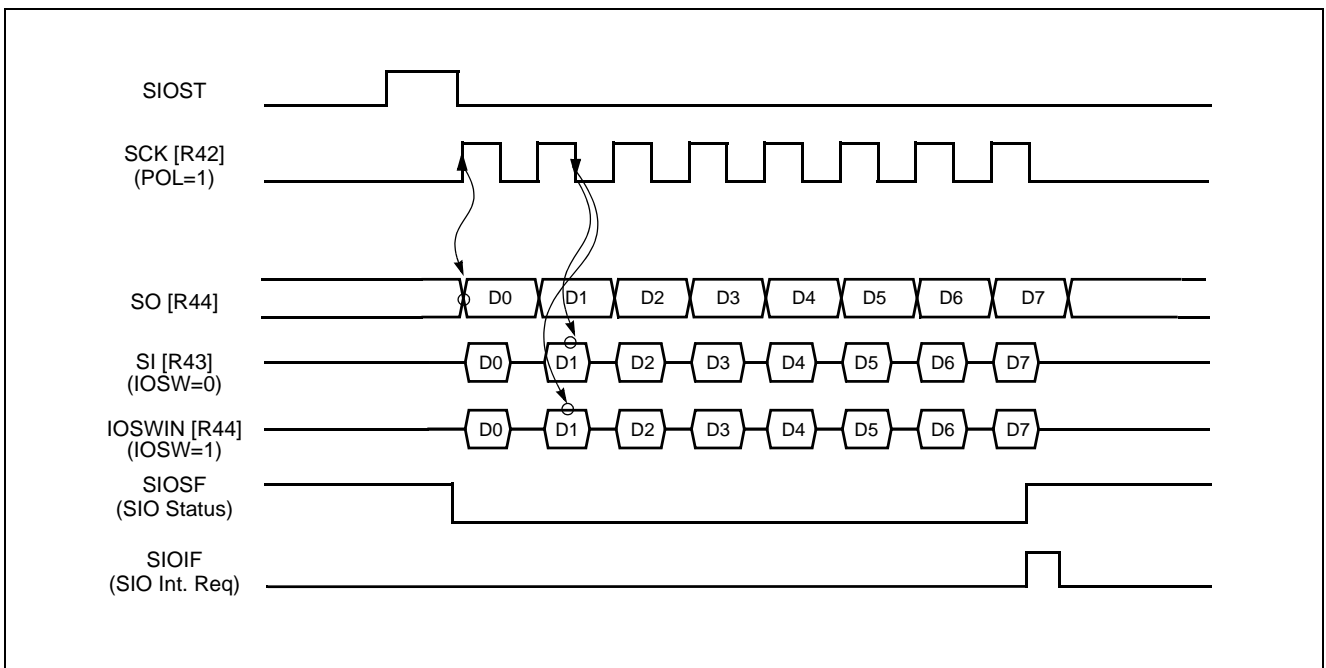


Figure 16-4 Serial I/O Timing Diagram at POL=1

16.2 The method 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 will be made.

16.3 The Method to Test Correct Transmission

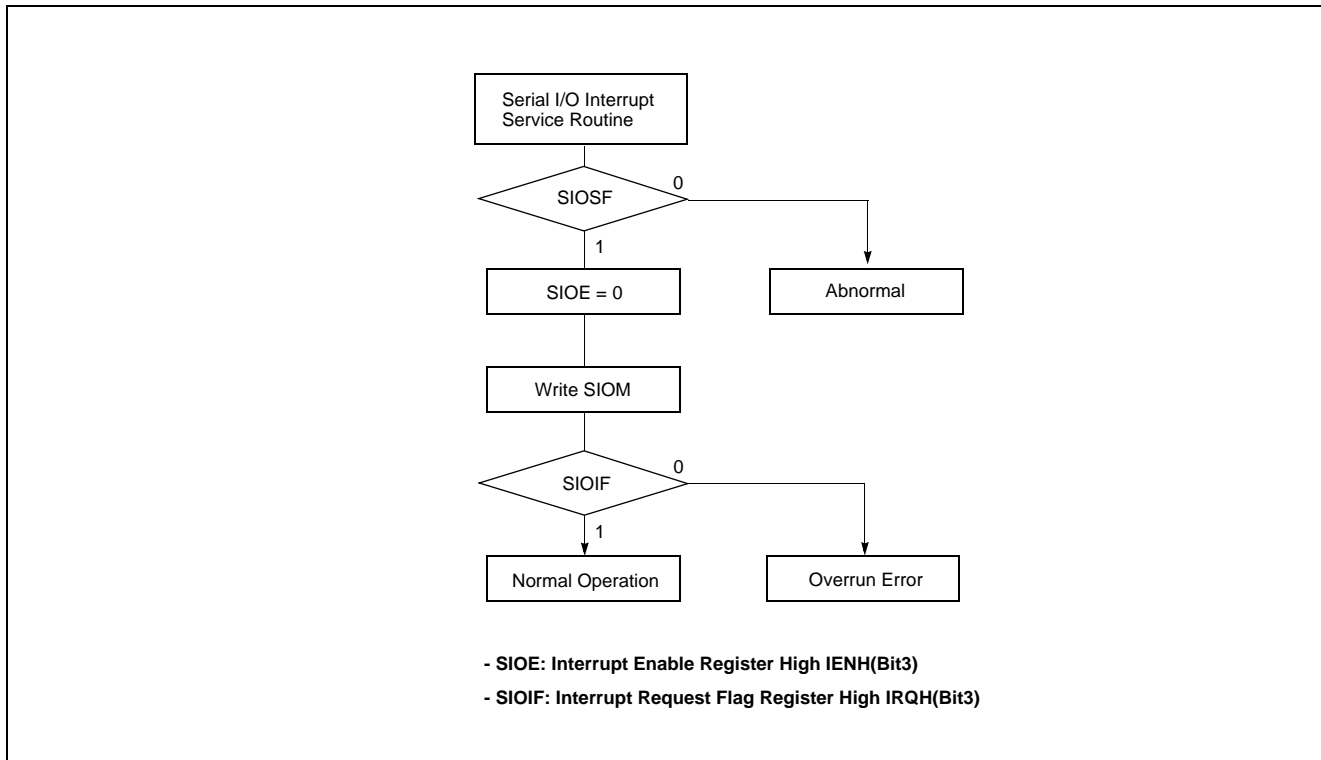


Figure 16-5 Serial IO Method to Test Transmission

17. UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART)

17.1 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. Clock asynchronous serial I/O mode (UART) can be selected by ASIMR register. Figure 17-1 shows a block diagram of the UART driver.

Note: The UART1 control register ASIMR1, ASISR1, BRGCR1, RXR1 and TXR1 are located at EE6H ~ EE9H address. These address must be accessed(read and written) by absolute addressing manipulation instruction.

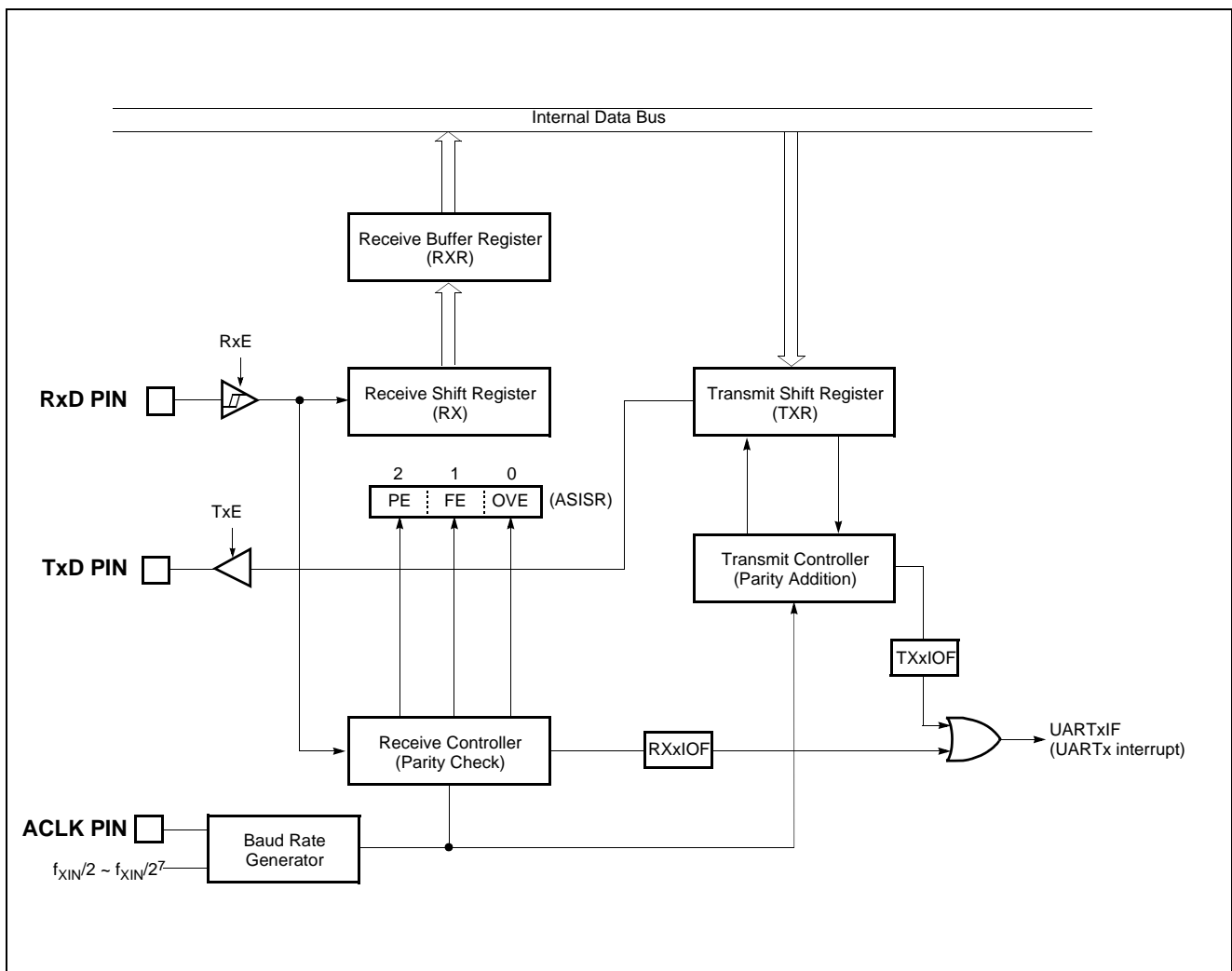


Figure 17-1 UART Block Diagram

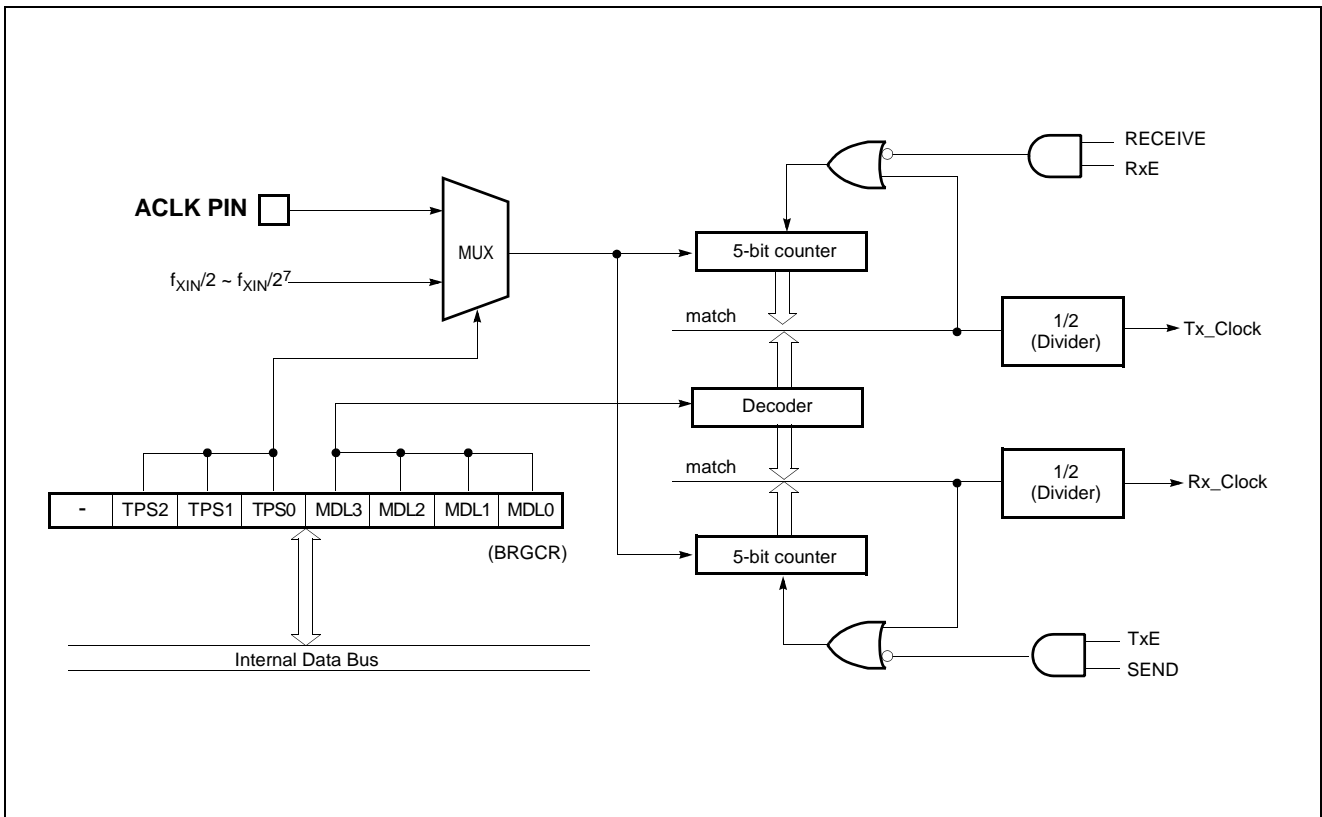


Figure 17-2 Baud Rate Generator Block Diagram

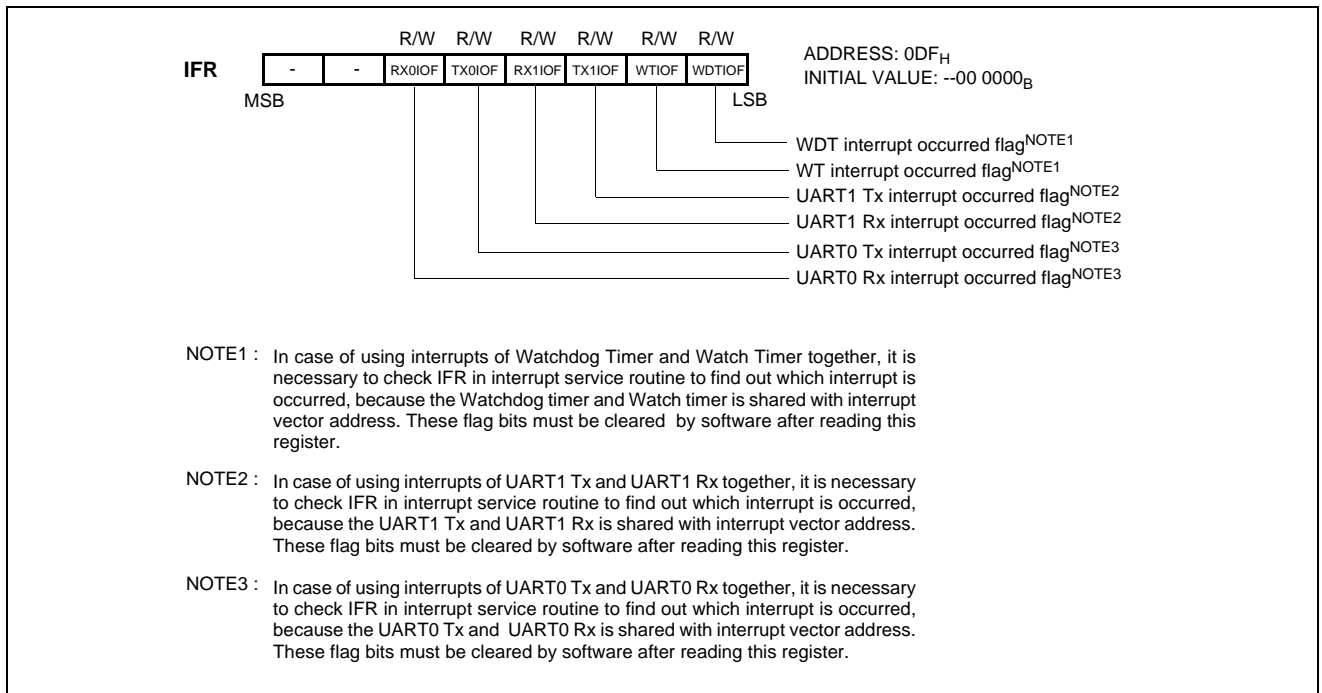


Figure 17-3 IFR : Interrupt Flag Register

17.2 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) Baudrate generator control register (BRGCR) |

Table 17-1 Serial Interface Configuration

Transmit shift register (TXR)

This is the register for setting transmit data. Data written to TXR0 is transmitted as serial data. When the data length is set as 7 bit, bit 0 to 6 of the data written to TX0 are transferred as transmit data. Writing data to TXR0 starts the transmit operation.

TXR0 can be written by an 8 bit memory manipulation instruction. It cannot be read. The $\overline{\text{RESET}}$ input sets TXR0 to 0FF_H.

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

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 RXR0. 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 RXR0 to 00_H.

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

Receive shift register

This register converts serial data input via the RxD pin to parallel 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 instruction. The $\overline{\text{RESET}}$ input sets ASIMR to 0000_00_0_B. Table 17-4 shows the format of ASIMR.

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

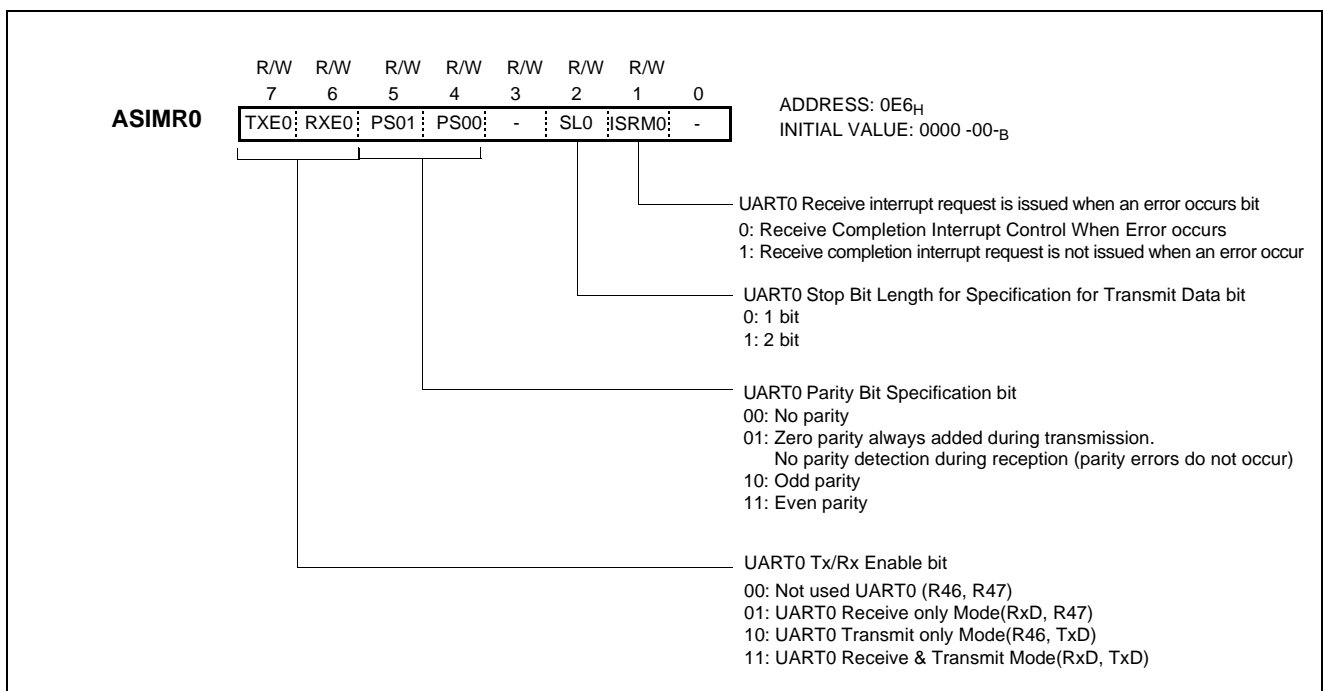


Figure 17-4 Asynchronous Serial Interface Mode register (ASIMR0) Format

Asynchronous serial interface status register0 (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 RESET input sets ASISR0 to -----

000B. Figure 17-5 shows the format of ASISR.

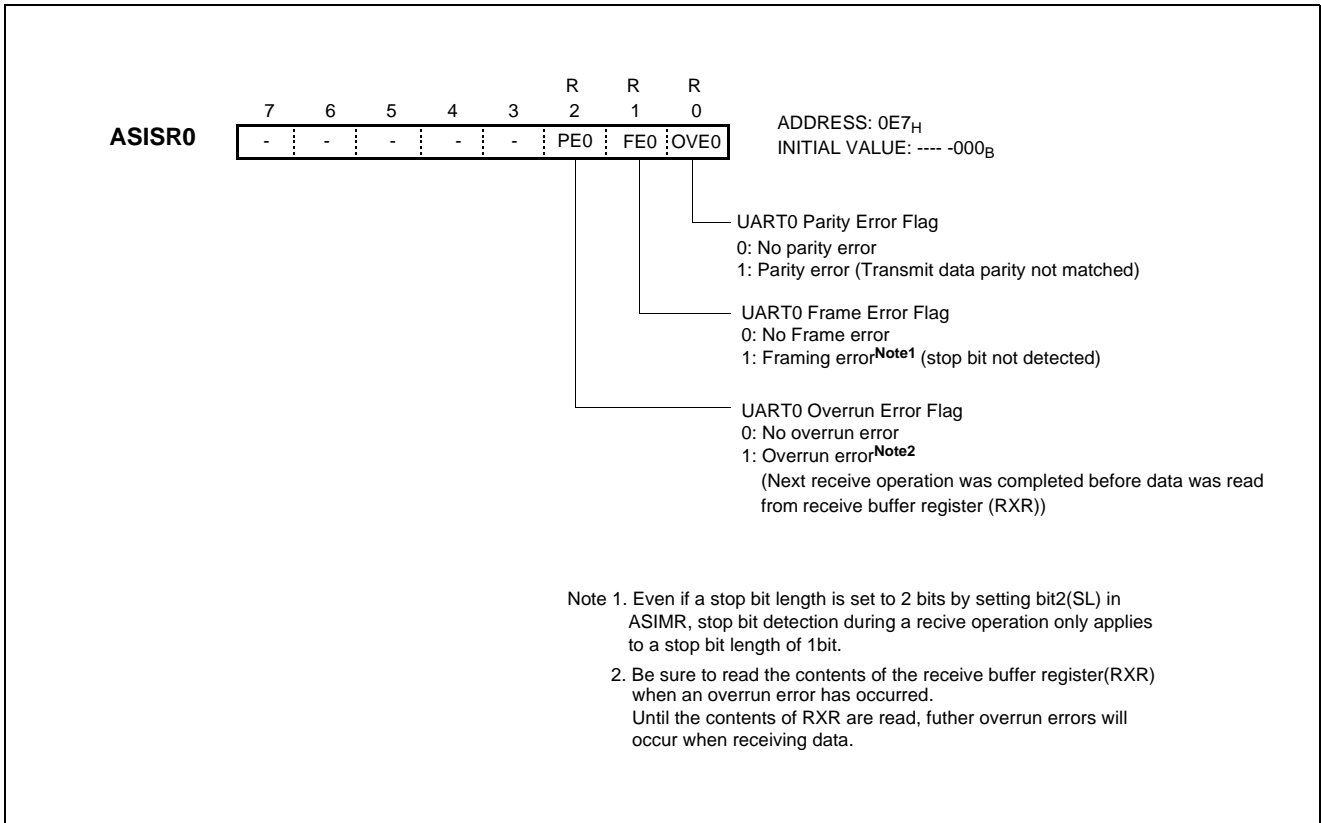


Figure 17-5 Asynchronous Serial Interface Status Register (ASISR) Format

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 **RESET** input sets BRGCR to -001_0000B.

Figure 17-6 shows the format of BRGCR.

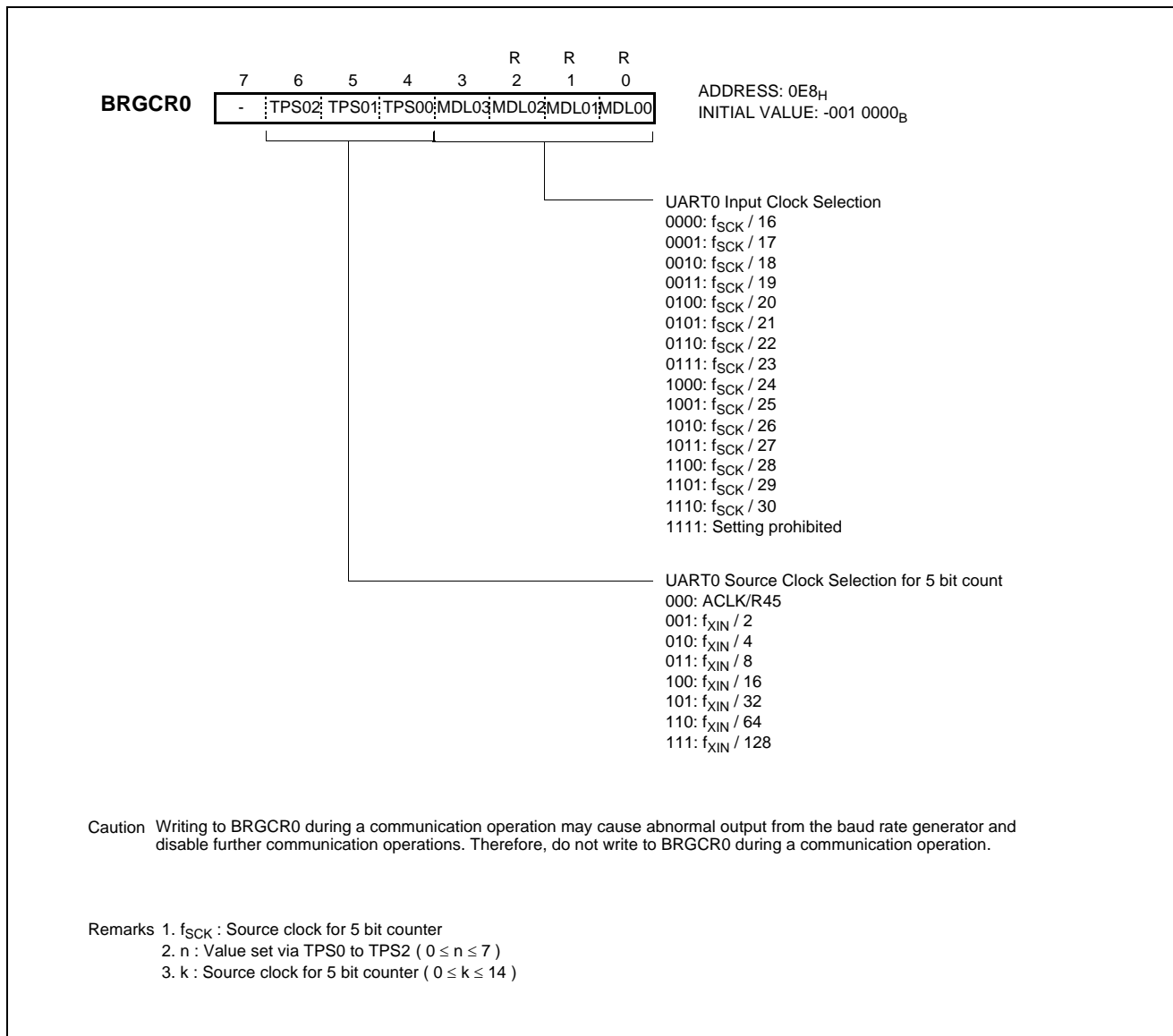


Figure 17-6 Baud Rate Generator Control Register0(BRGCR) Format

17.3 Communication operation

The transmit operation is enabled when bit 7 (TXE0) 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 17-8.

The receive operation is enabled when bit 6 (RXE0) 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_RX0) occurs. Even if an error has occurred, the receive data in which the error occurred is still transferred to RXR. When ASIMR bit 1 (ISRM0) is cleared to 0 upon occurrence of an error, and INT_RX0 occurs. When ISRM bit is set to 1, INT_RX0 does not occur in case of error occurrence. Figure 17-8 shows the timing of the asynchronous serial interface receive completion interrupt request.

In case of using interrupts of UART0 Tx and UART0 Rx together, it is necessary to check IFR in interrupt service routine to find out which interrupt is occurred, because the UART0 Tx and UART0 Rx is shared with interrupt vector address. These flag bits must be cleared by software after reading this register.

In case of using interrupts of UART1 Tx and UART1 Rx together, it is necessary to check IFR in interrupt service routine to find out which interrupt is occurred, because the UART1 Tx and UART1 Rx is shared with interrupt vector address. These flag bits must be cleared by software after reading this register.

Each processing step is determined by IFR as shown in Figure 17-7.

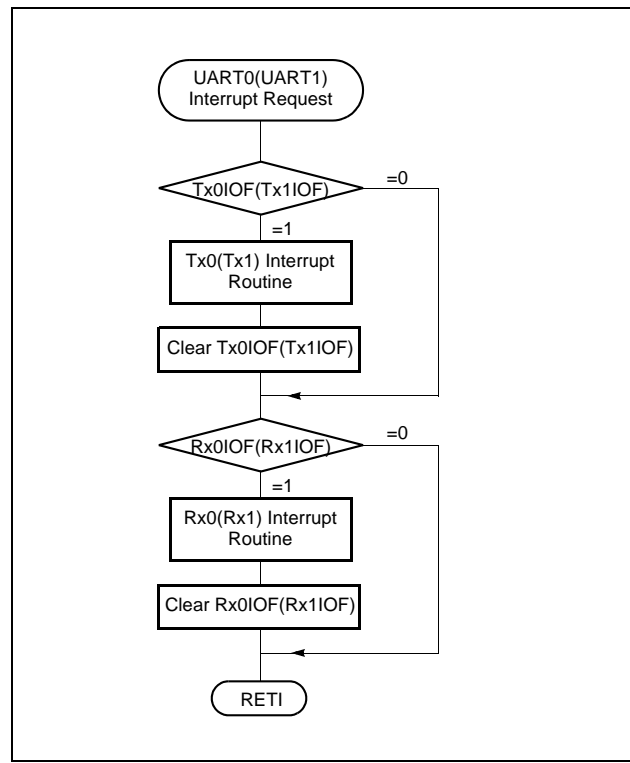


Figure 17-7 Shared Interrupt Vector of UART

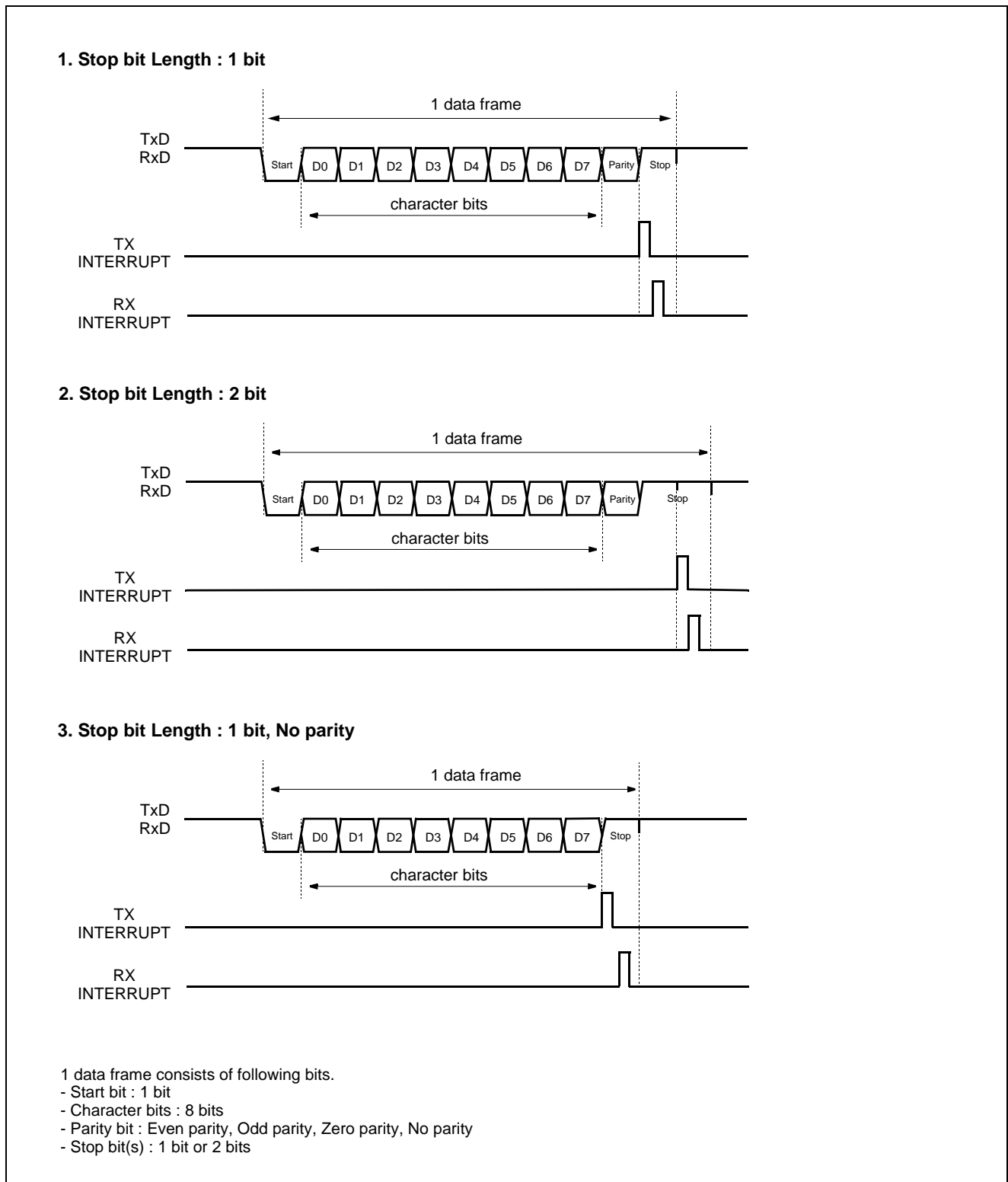


Figure 17-8 UART data format and interrupt timing diagram

17.4 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} =6.0M | | f _{XIN} =4.0M | | f _{XIN} =2.0M | |
|-----------------|----------------------------|---------|-------------------------|---------|------------------------|---------|------------------------|---------|------------------------|---------|------------------------|---------|
| | BRGCR | ERR (%) | BRGCR | ERR (%) | BRGCR | ERR (%) | BRGCR | ERR (%) | BRGCR | ERR (%) | BRGCR | ERR (%) |
| 600 | - | - | - | - | - | - | - | - | 7AH | 0.16 | 6AH | 0.16 |
| 1200 | - | - | - | - | 7AH | 0.16 | 74H | 2.34 | 6AH | 0.16 | 5AH | 0.16 |
| 2400 | 72H | 0.00 | 70H | 1.73 | 6AH | 0.16 | 64H | 2.34 | 5AH | 0.16 | 4AH | 0.16 |
| 4800 | 62H | 0.00 | 60H | 1.73 | 5AH | 0.16 | 54H | 2.34 | 4AH | 0.16 | 3AH | 0.16 |
| 9600 | 52H | 0.00 | 50H | 1.73 | 4AH | 0.16 | 44H | 2.34 | 3AH | 0.16 | 2AH | 0.16 |
| 19200 | 42H | 0.00 | 40H | 1.73 | 3AH | 0.16 | 34H | 2.34 | 2AH | 0.16 | 1AH | 0.16 |
| 31250 | 36H | 0.53 | 34H | 0.00 | 30H | 0.00 | 28H | 0.00 | 20H | 0.00 | 10H | 0.00 |
| 38400 | 32H | 0.00 | 30H | 1.73 | 2AH | 0.16 | 24H | 2.34 | 1AH | 0.16 | - | - |
| 57600 | 28H | 0.00 | 26H | 1.35 | 21H | 2.11 | 1AH | 0.16 | 11H | 2.12 | - | - |
| 76800 | 22H | 0.00 | 20H | 1.73 | 1AH | 0.16 | 14H | 2.34 | - | - | - | - |
| 115200 | 18H | 0.00 | 16H | 1.36 | 11H | 2.12 | - | - | - | - | - | - |

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

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 TPS00 to TPS02 (0 ≤ n ≤ 7)

4. k : Source clock for 5 bit counter (0 ≤ k ≤ 14)

Figure 17-9 Relationship between main clock and Baud Rate

18. 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 0F9H) to "1". For PSR1 register, refer to Figure 18-2.

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.

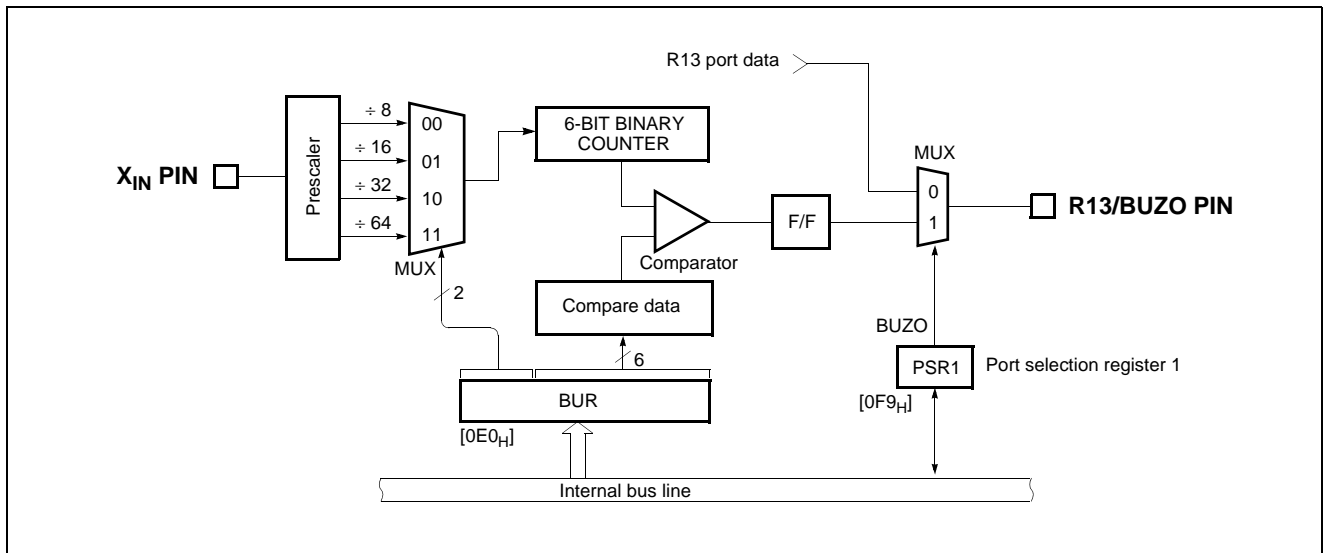


Figure 18-1 Block Diagram of Buzzer Driver

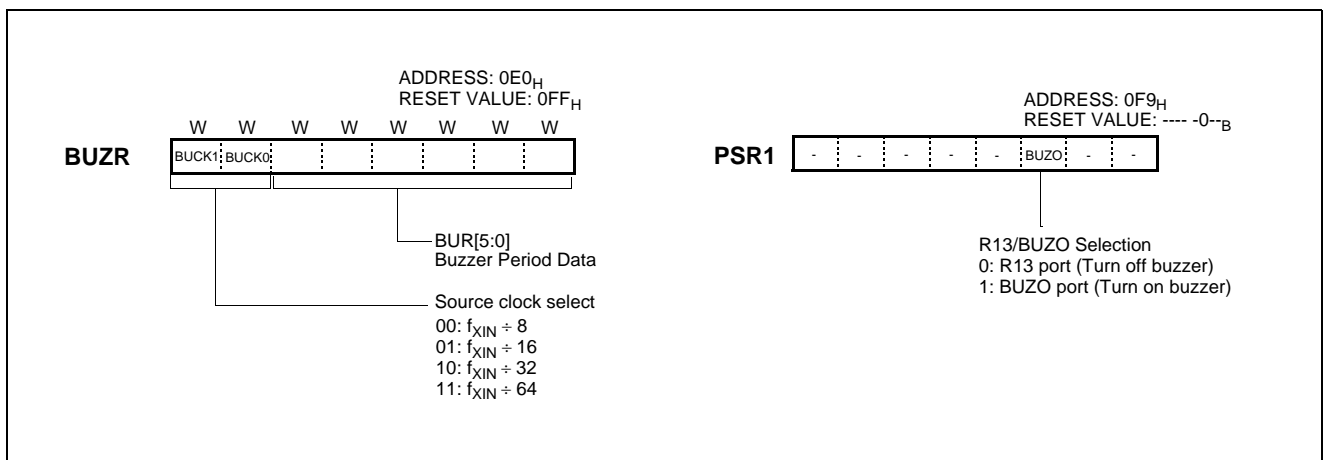


Figure 18-2 Buzzer Register & PSR1

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 18-1.

| 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 18-1 buzzer frequency (kHz unit)

19. INTERRUPTS

The MC80F0208/16/24 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 19-1 and interrupt priority is shown in Table 19-1.

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 4 Interrupts are generated by T0IF, T1IF, T2IF, T3IF and T4IF 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 and Watch Timer Interrupt is generated by WDTIF and WTIF which is set by a match in Watchdog timer register or Watch timer register. The IFR(Interrupt Flag Register) is used for discrimination of the interrupt source among these two Watchdog timer and Watch Timer Interrupt.

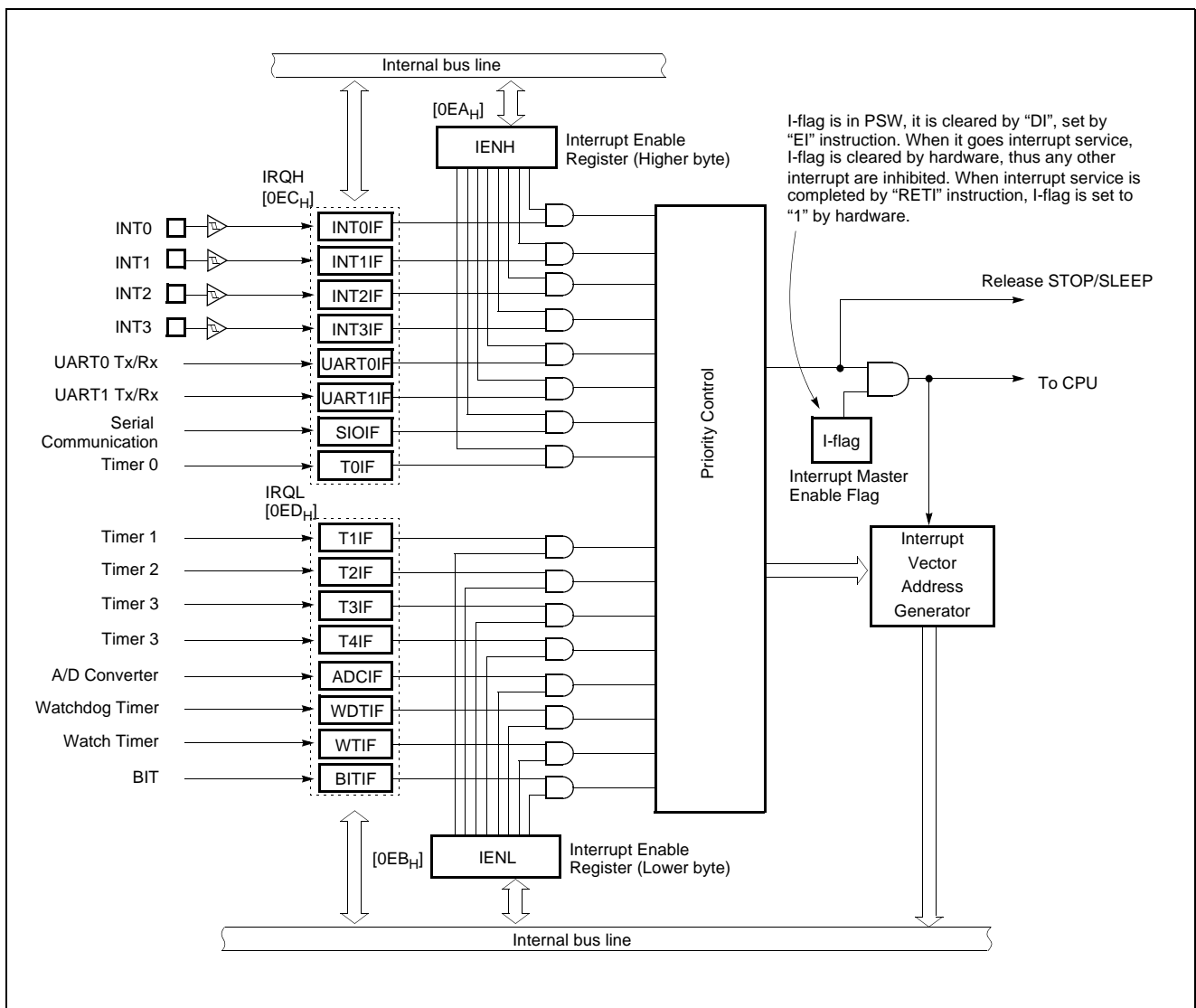


Figure 19-1 Block Diagram of Interrupt

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

The UART0 receive/transmit interrupt is generated by UART0IF is set by completion of UART0 data reception or transmission. The IFR(Interrupt Flag Register) is used for discrimination of the interrupt source among these two UART0 receive and UART0 transmit Interrupt.

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 19-1 shows the Interrupt priority.

Vector addresses are shown in Figure 8-6. Interrupt enable registers are shown in Figure 19-2. 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 |
| UART0 Rx/Tx Interrupt | UART0 | 6 |
| UART1 Rx/Tx Interrupt | UART1 | 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 |
| Timer/Counter 4 | Timer 4 | 13 |
| ADC Interrupt | ADC | 14 |
| Watchdog/Watch Timer | WDT_WT | 15 |
| Basic Interval Timer | BIT | 16 |

Table 19-1 Interrupt Priority

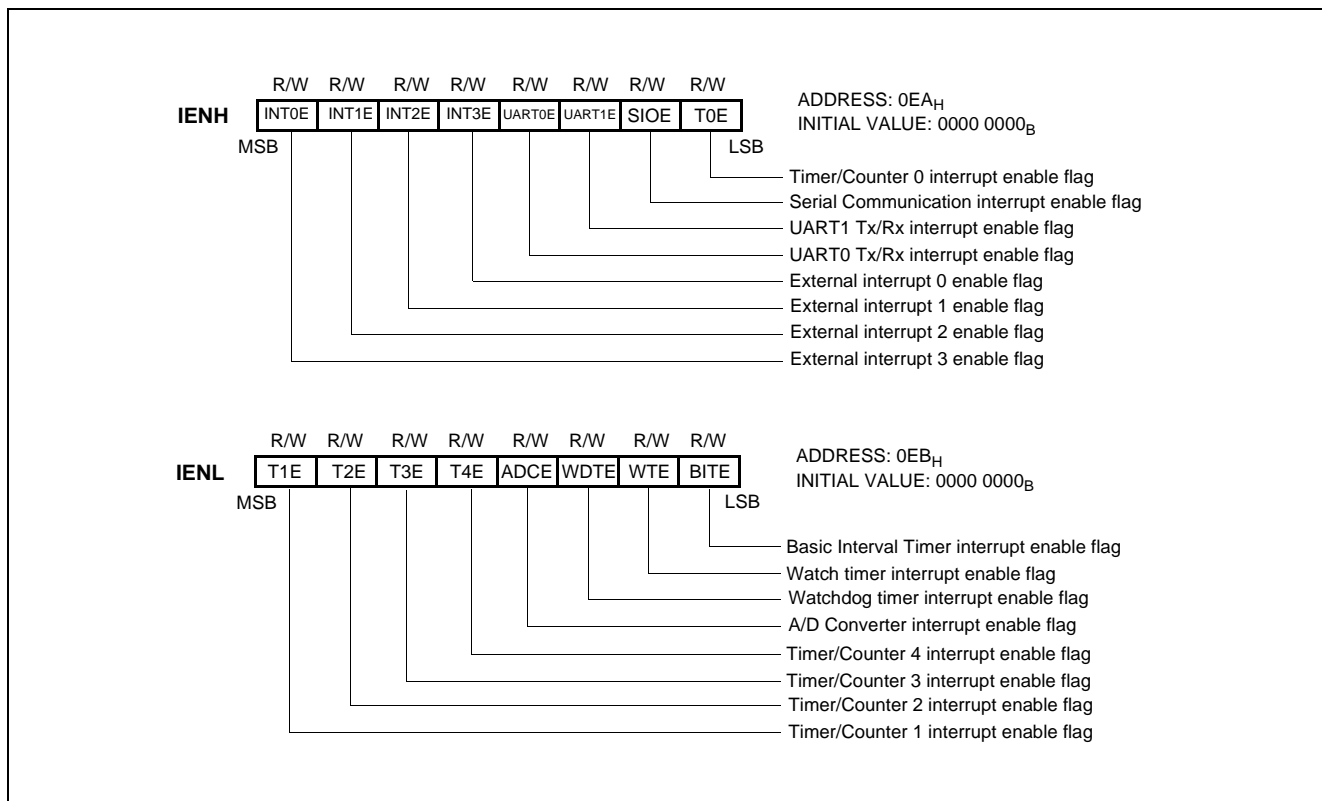


Figure 19-2 Interrupt Enable Flag Register

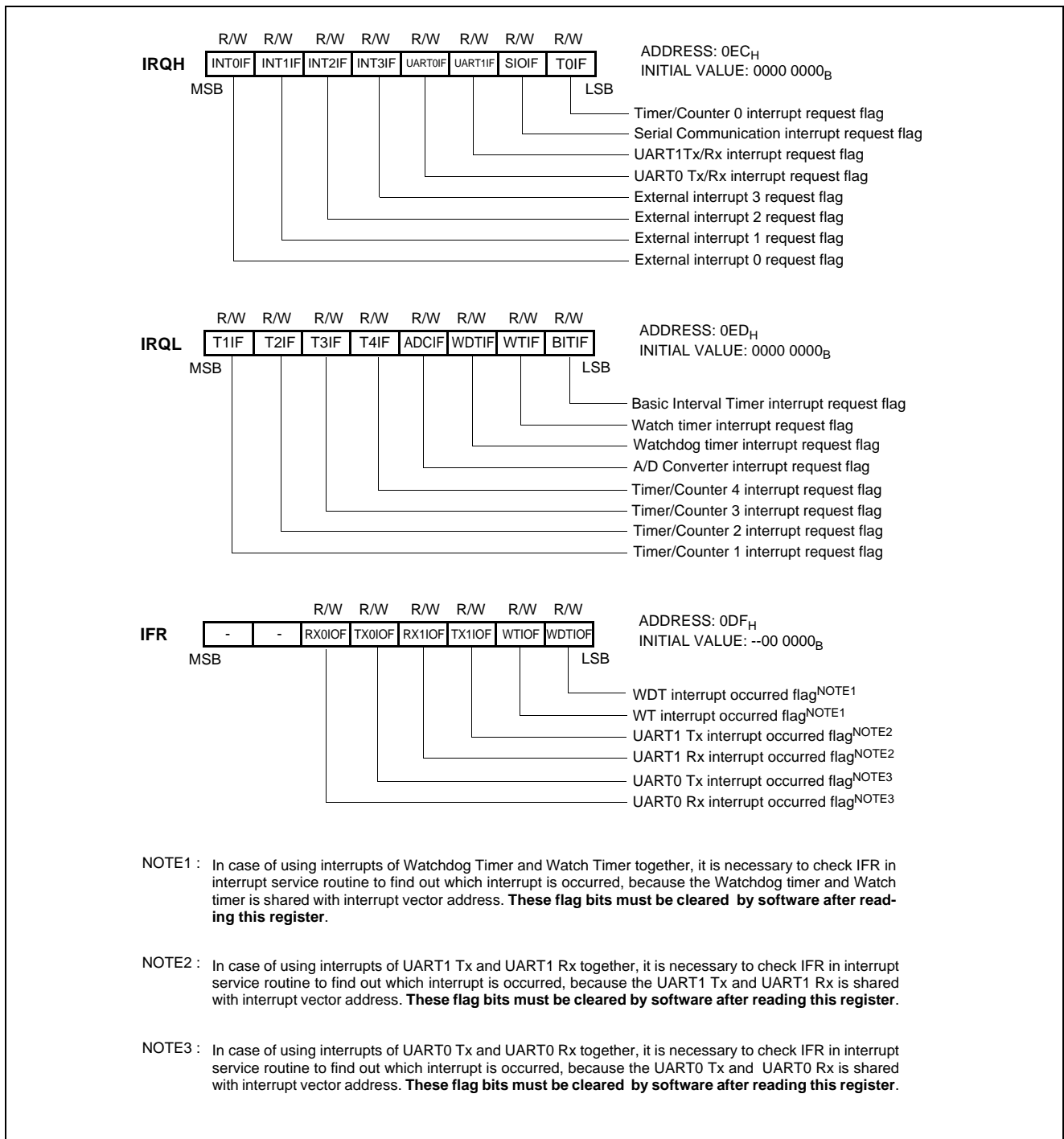


Figure 19-3 Interrupt Request Flag Register & Interrupt Flag Register

19.1 Interrupt Sequence

An interrupt request is held until the interrupt is accepted or the interrupt latch is cleared to “0” by a reset or an instruction. Interrupt acceptance sequence requires 8 cycles of f_{XIN} ($2\mu s$ at f_X).

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

19.1.1 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.

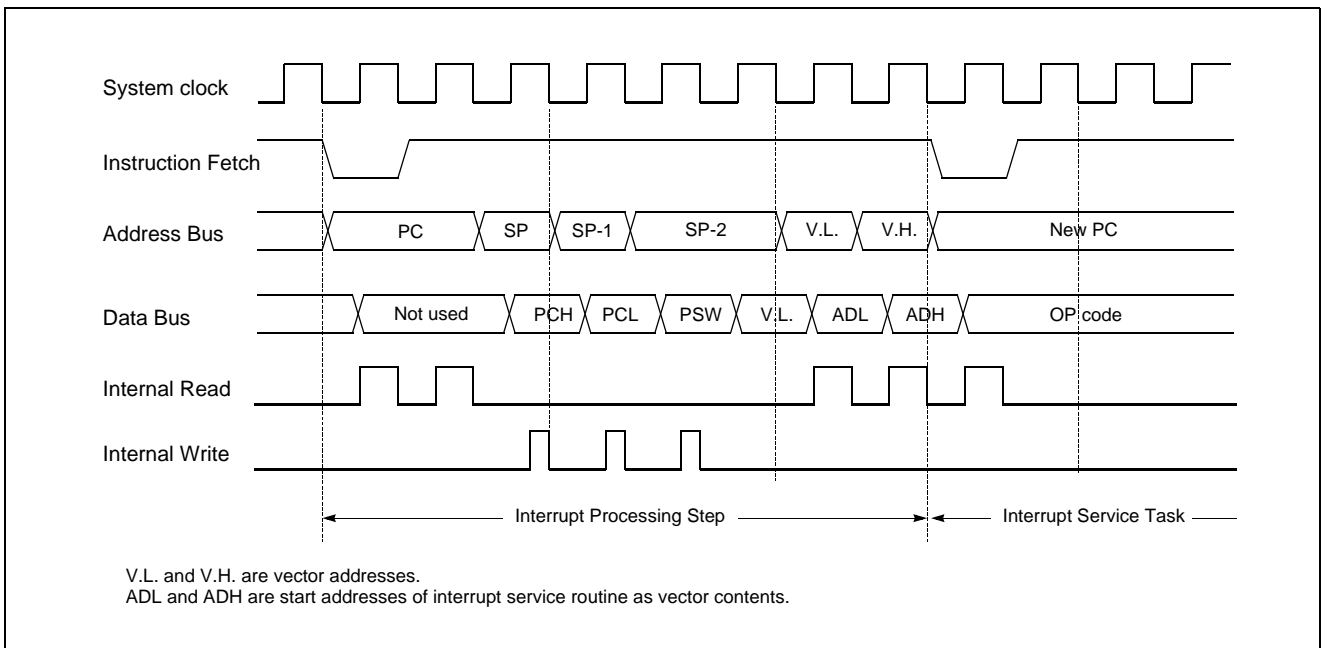
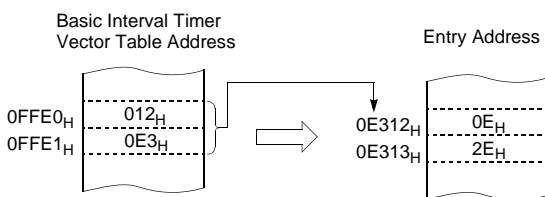


Figure 19-4 Timing chart of Interrupt Acceptance and Interrupt Return Instruction



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.

19.1.2 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.

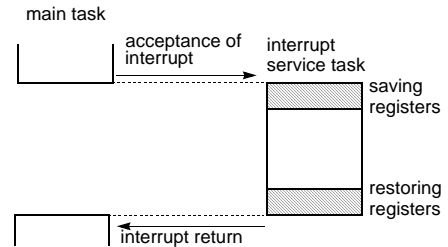
Example: Register save using push and pop instructions

```
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;



19.2 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 19-5.

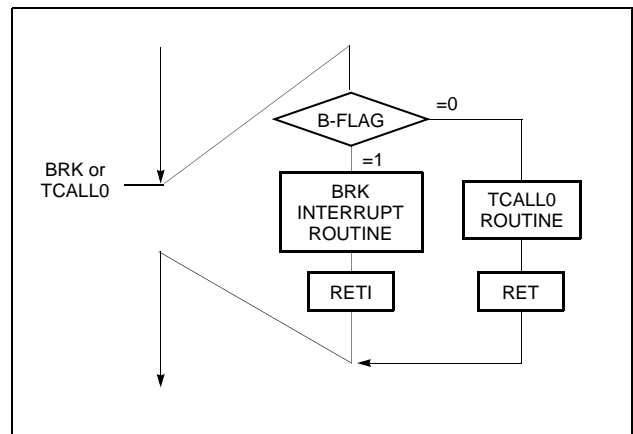


Figure 19-5 Execution of BRK/TCALL0

19.3 Shared Interrupt Vector

In case of using interrupts of Watchdog Timer and Watch Timer together, it is necessary to check IFR in interrupt service routine to find out which interrupt is occurred, because the Watchdog timer and Watch timer is shared with interrupt vector address. These flag bits must be cleared by software after reading this register.

In case of using interrupts of UART0 Tx and UART0 Rx together, it is necessary to check IFR in interrupt service routine to find

out which interrupt is occurred, because the UART0 Tx and UART0 Rx is shared with interrupt vector address. These flag bits must be cleared by software after reading this register.

In case of using interrupts of UART1 Tx and UART1 Rx together, it is necessary to check IFR in interrupt service routine to find out which interrupt is occurred, because the UART1 Tx and UART1 Rx is shared with interrupt vector address. These flag bits must be cleared by software after reading this register. Each

processing step is determined by IFR as shown in Figure 19-6.

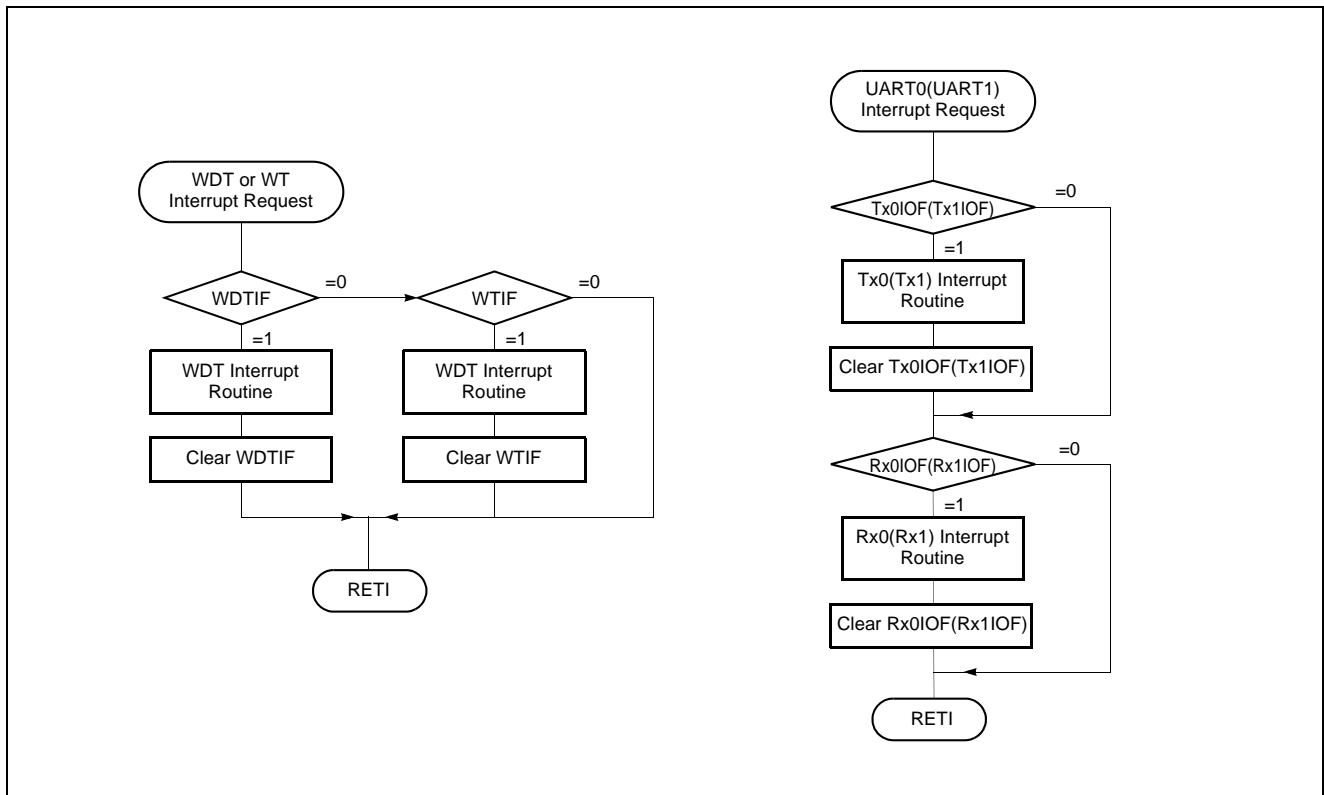
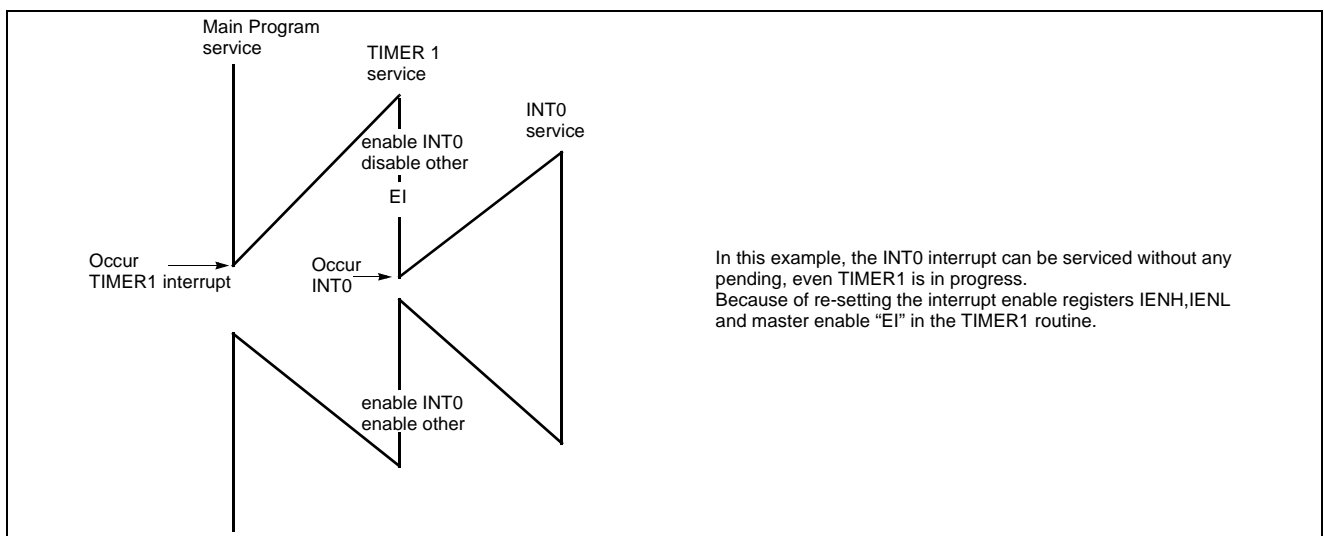


Figure 19-6 Software Flowchart of Shared Interrupt Vector

19.4 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.



In this example, the INT0 interrupt can be serviced without any pending, even TIMER1 is in progress. Because of re-setting the interrupt enable registers IENH, IENL and master enable "EI" in the TIMER1 routine.

Figure 19-7 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
    
```

19.5 External Interrupt

The external interrupt on INT0, INT1, INT2 and INT3 pins are edge triggered depending on the edge selection register IEDS (address 0EEH) as shown in Figure 19-8.

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

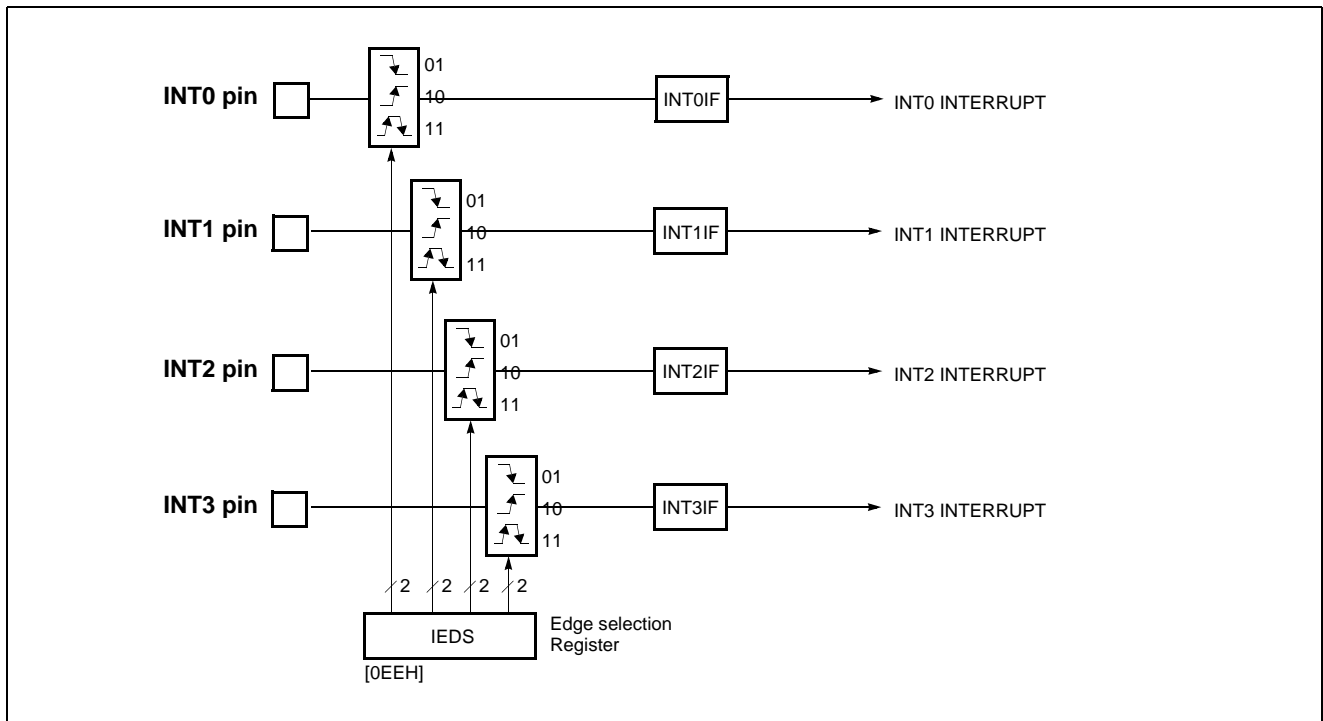


Figure 19-8 External Interrupt Block Diagram

INT0 ~ INT3 are multiplexed with general I/O ports (R10, R11, R12, R50). 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 19-9 shows interrupt response timings.

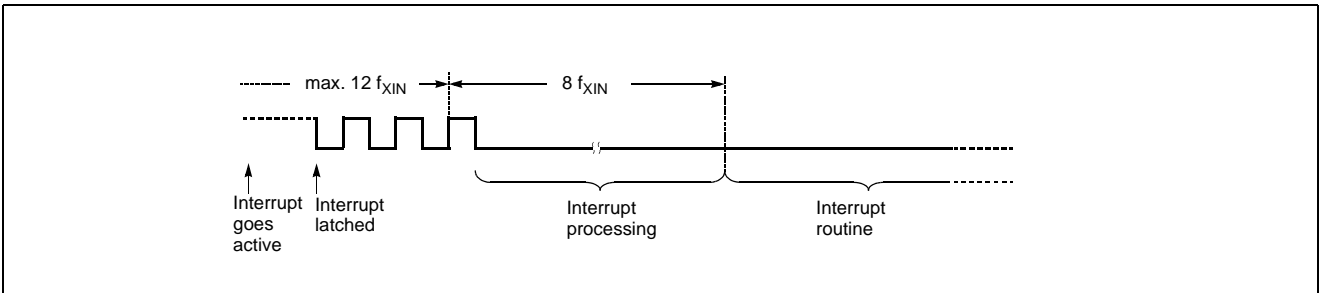


Figure 19-9 Interrupt Response Timing Diagram

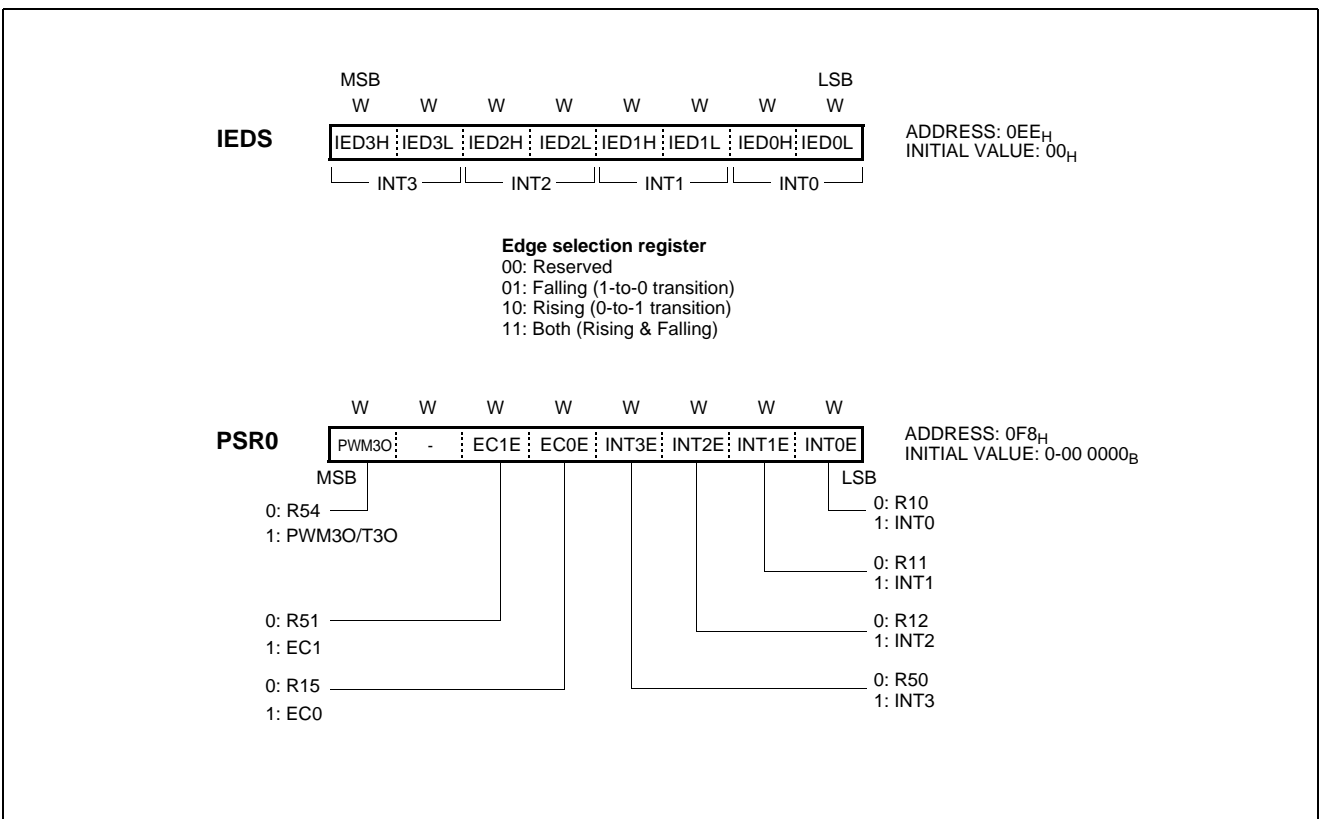


Figure 19-10 IEDS register and Port Selection Register PSR0

20. OPERATION MODE

The system clock controller starts or stops the main-frequency clock oscillator. The operating mode is generally divided into the main active mode. Figure 20-1 shows the operating mode transition diagram.

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

Main Active Mode

This mode is fast-frequency operating mode. The CPU and the peripheral hardware are operated on the high-frequency clock. At reset release, this mode is invoked.

SLEEP Mode

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

STOP Mode

In this mode, the system operations are all stopped, holding the internal states valid immediately before the stop at the low power consumption level. The main oscillation source stops, but the sub clock oscillation and watch timer by sub clock and RC-oscillated watchdog timer don't stop.

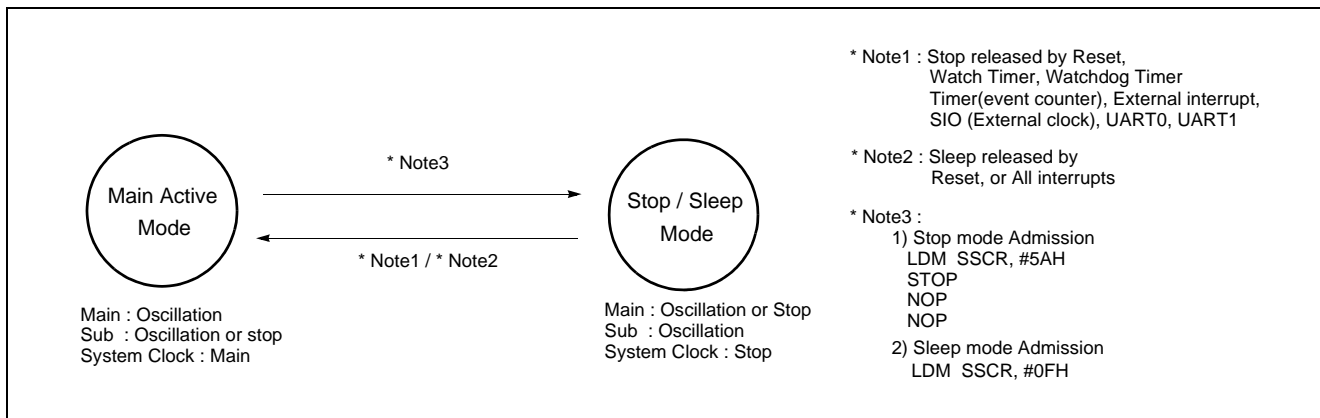


Figure 20-1 Operating Mode

20.1 Operation Mode Switching

In the Main active mode, only the high-frequency clock oscillator is used. In the Sub active mode, the low-frequency clock oscillation is used, so the low power voltage operation or the low power consumption operation can be enabled. Instruction execution does not stop during the change of operation mode. In this case, some peripheral hardware capabilities may be affected. For de-

tails, refer to the description of the relevant operation.

The following describes the switching between the Main active mode and the Sub active mode. During reset, the system clock mode register is initialized at the Main active mode. It must be set to the Sub active mode for reducing the power consumption.

Shifting from the Normal operation to the SLEEP mode

If the CPU clock stops and the SLEEP mode is invoked, the CPU stops while other peripherals are operate normally. The ways of release from this mode are by setting the RESET pin

to low and all available interrupts. For more detail, See "21. POWER SAVING OPERATION" on page 95.

Shifting from the Normal operation to the STOP mode

If the main-frequency clock oscillation stops and the STOP mode is invoked, the CPU stops and other peripherals are stop too. But sub-frequency clock oscillation operate continuously if enabled previously. After the STOP operation is released by reset, the operation mode is changed to Main active mode.

The methods of release from this mode are Reset, Watch Timer, Timer/Event counter, SIO(External clock), UART, and External Interrupt.

For more details, see "21. POWER SAVING OPERATION" on page 95.

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

21. POWER SAVING OPERATION

The MC80F0208/16/24 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 21-1 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”.

21.1 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 21-1. 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.

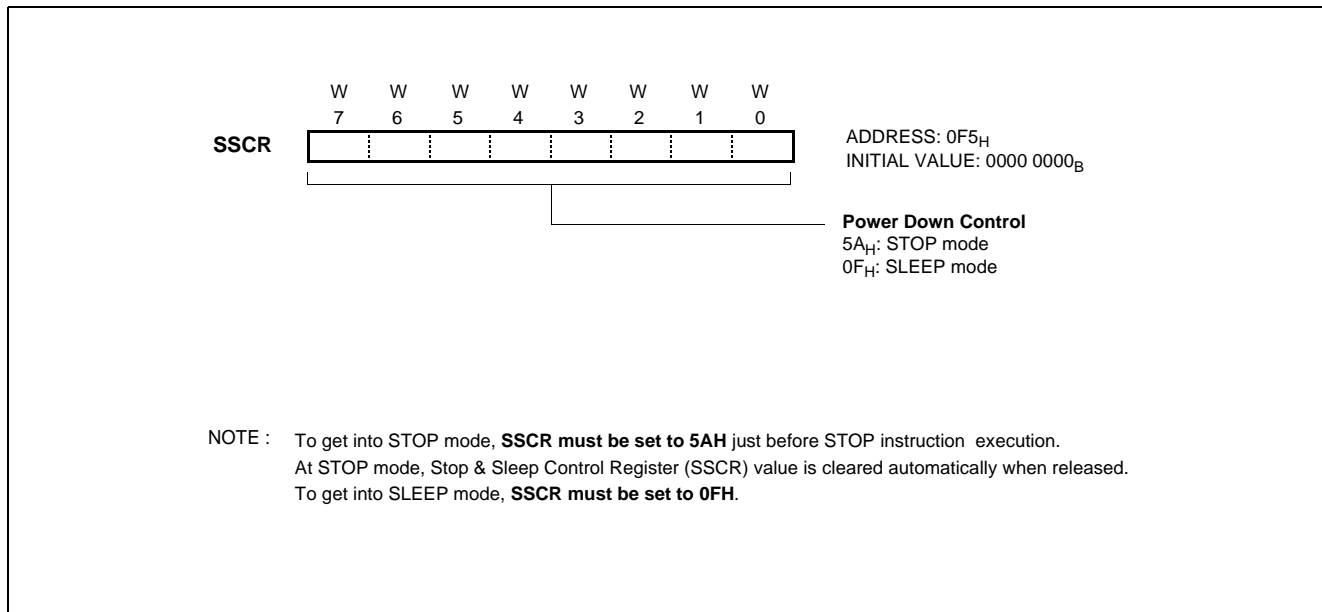


Figure 21-1 STOP and SLEEP Control Register

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 21-4)

When exit from SLEEP mode by reset, enough oscillation stabilization time is required to normal operation. Figure 21-3 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 21-2. By reset, exit from SLEEP mode is shown in Figure 21-3.

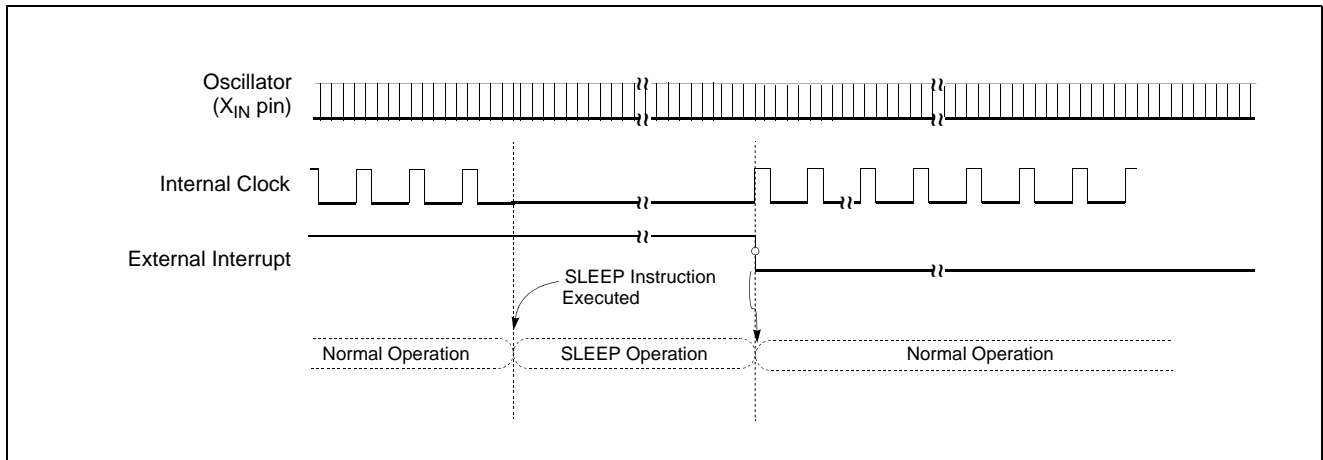


Figure 21-2 SLEEP Mode Release Timing by External Interrupt

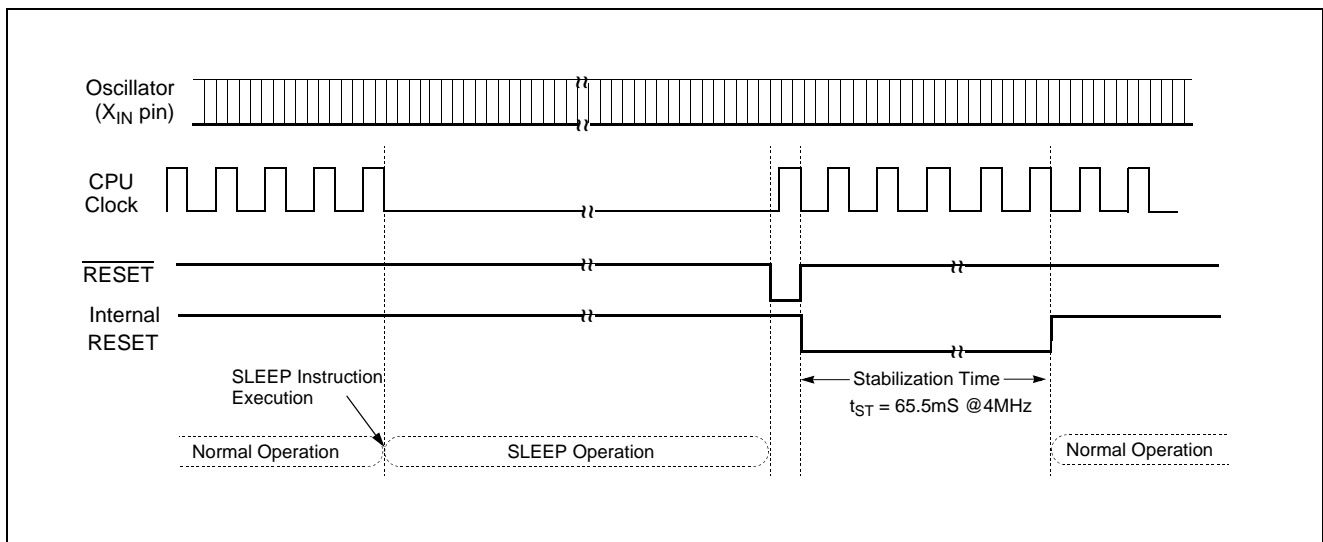


Figure 21-3 Timing of SLEEP Mode Release by Reset

21.2 Stop Mode

In the Stop mode, the main oscillator, system clock and peripheral clock is stopped, but the sub clock oscillation and Watch Timer by sub clock and 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
```

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.

In the STOP operation, the dissipation of the power associated

| 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 |
| Watch Timer | Stop | 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 |
| Sub Oscillator | Oscillation | 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, UART0(using ACLK0), UART1(using ACLK1) Watch Timer(RC-WDT mode), Watchdog Timer(RC-WDT mode), External Interrupt | Reset, All Interrupts |

Table 21-1 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 21-4)

When exit from Stop mode by external interrupt, enough oscillation stabilization time is required to normal operation. Figure 21-5 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 op-

eration. 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 21-6.

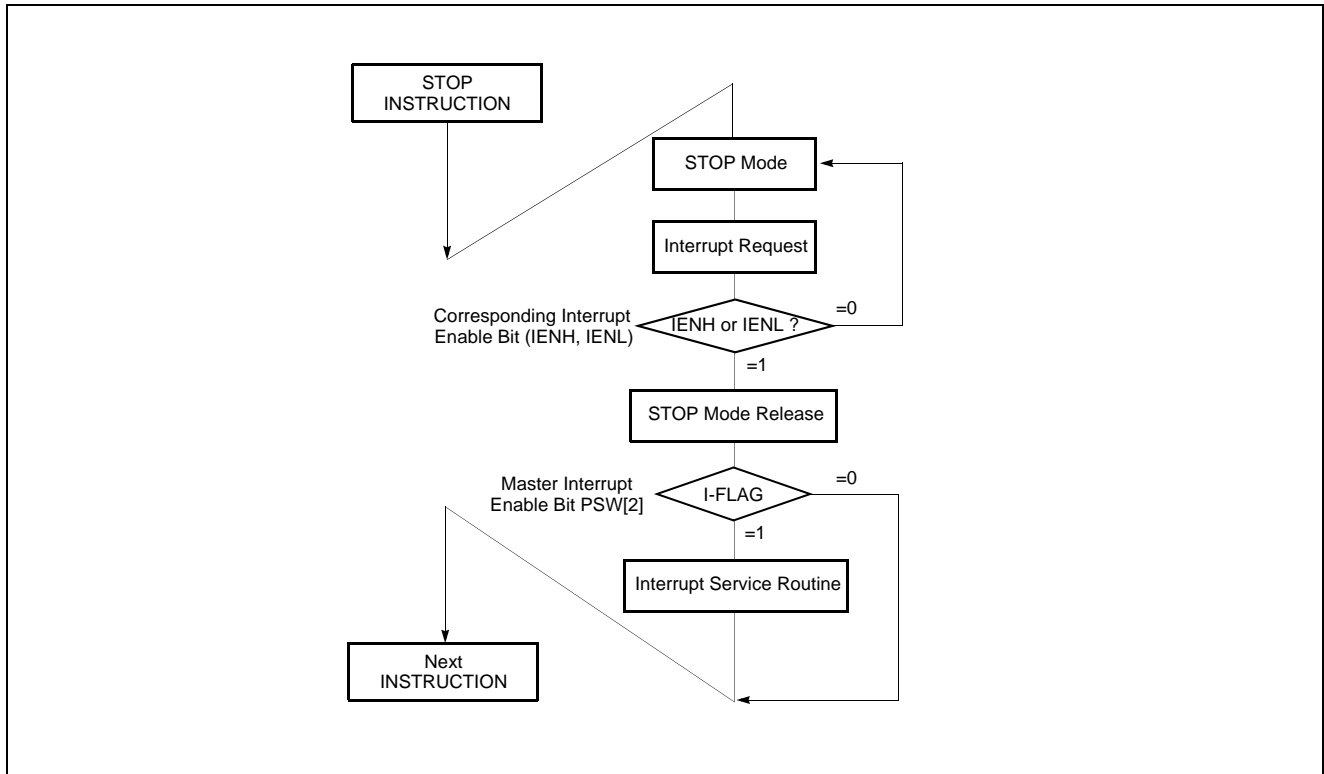


Figure 21-4 STOP Releasing Flow by Interrupts

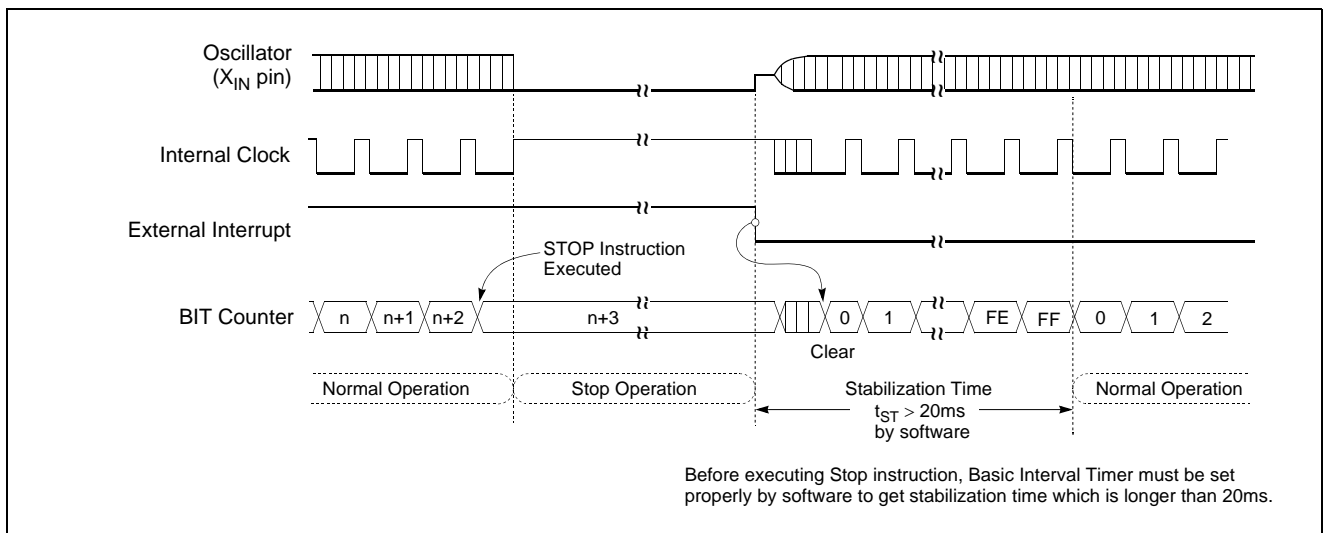


Figure 21-5 STOP Mode Release Timing by External Interrupt

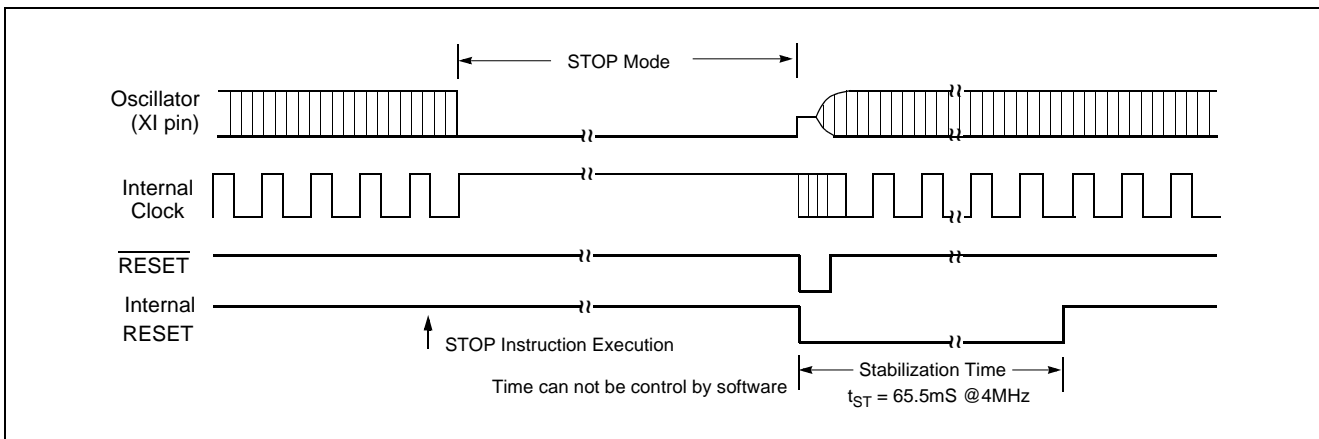


Figure 21-6 Timing of STOP Mode Release by Reset

21.3 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 21-8). 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 21-4)

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 21-7 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 21-8.

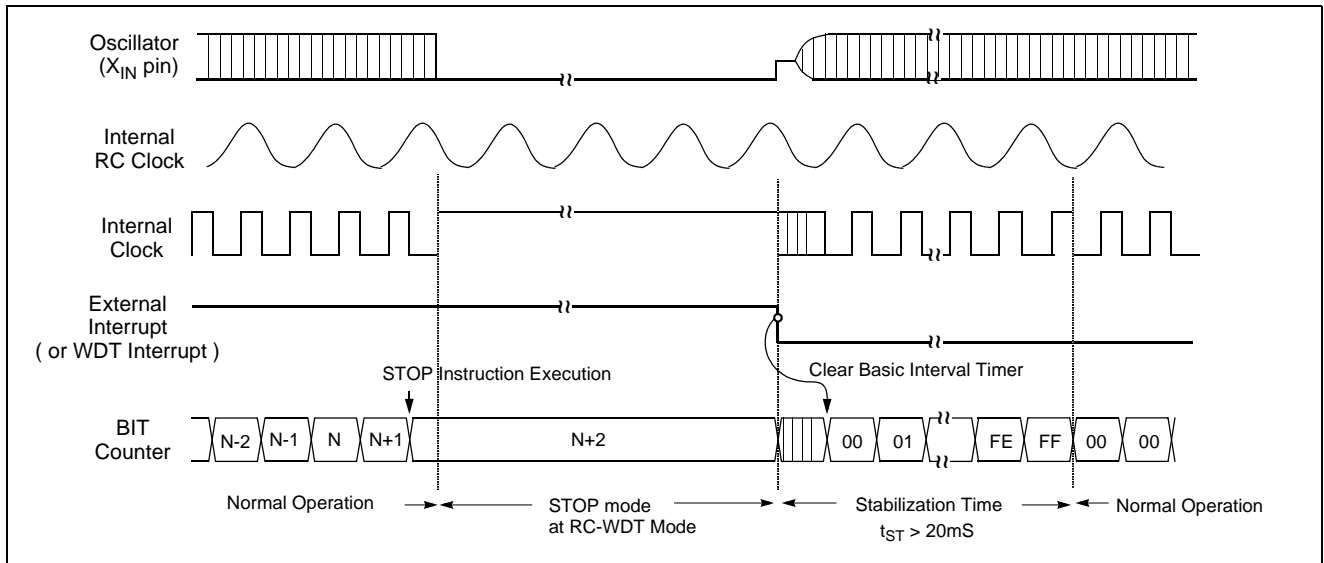


Figure 21-7 Stop Mode Release at Internal RC-WDT Mode by External Interrupt or WDT Interrupt

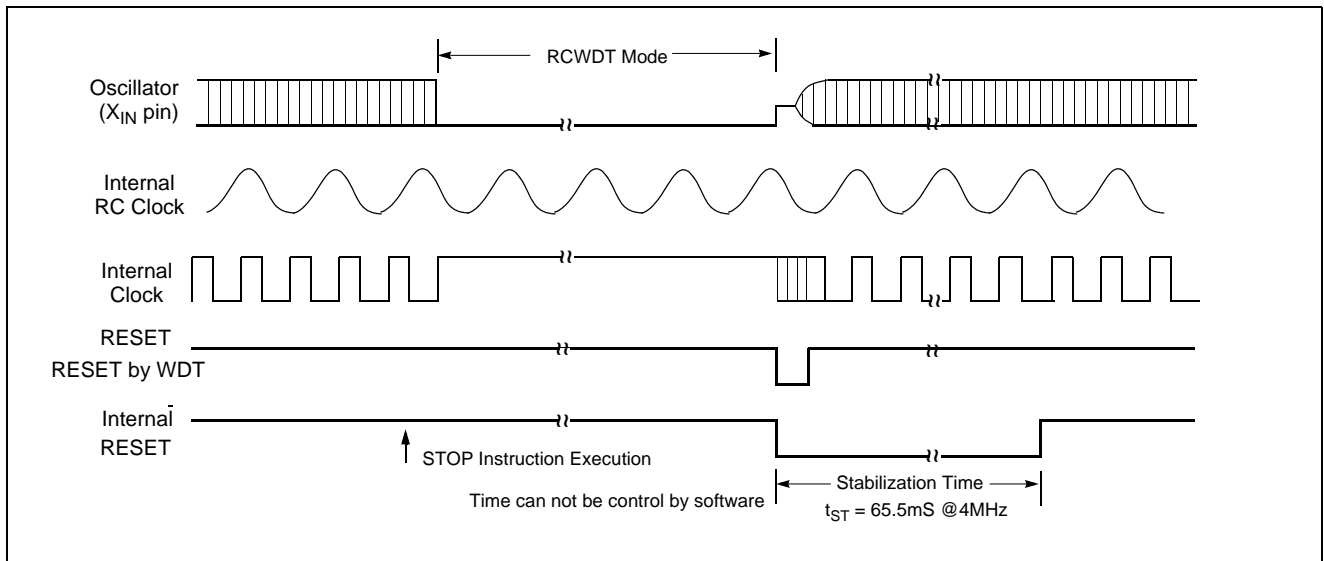


Figure 21-8 Internal RC-WDT Mode Releasing by Reset

21.4 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.

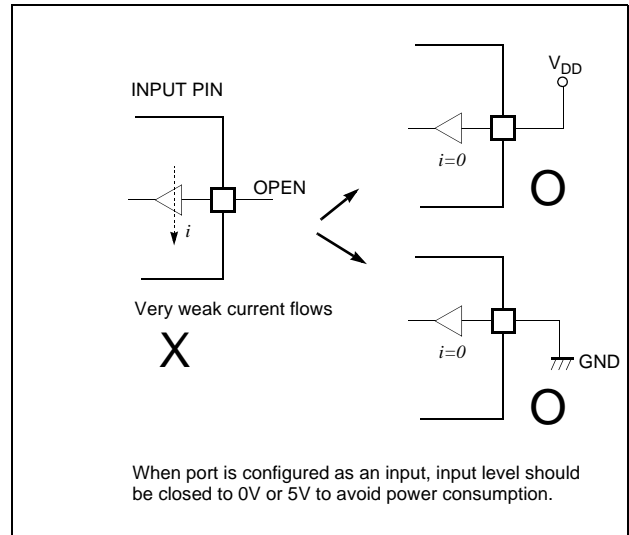
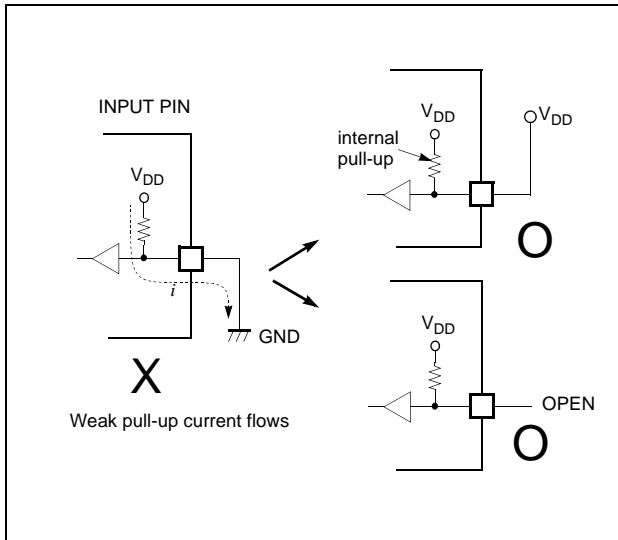


Figure 21-9 Application Example of Unused Input Port

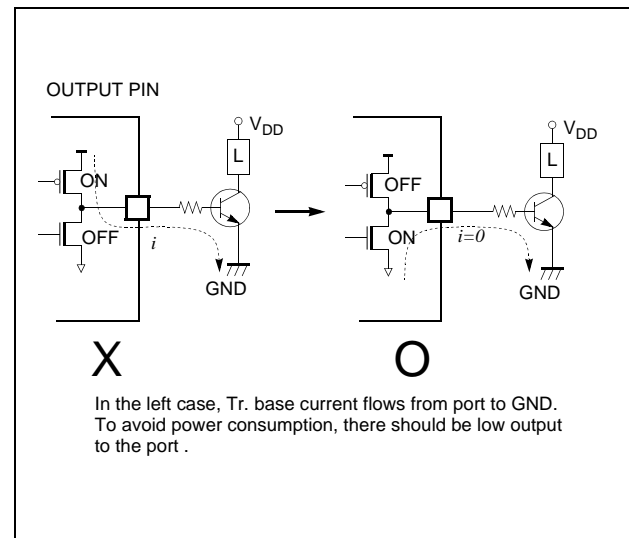
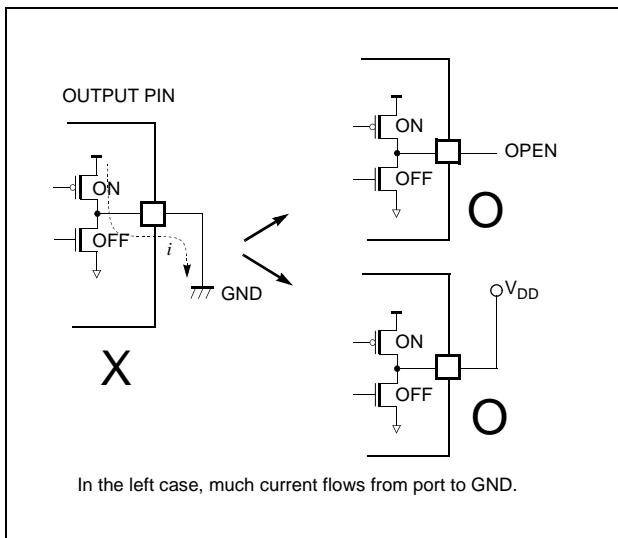


Figure 21-10 Application Example of Unused Output Port

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.

22. OSCILLATOR CIRCUIT

The MC80F0208/16/24 have oscillation circuits internally. X_{IN} and X_{OUT} are input and output for frequency. Respectively, in-

verting amplifier which can be configured for being used as an on-chip oscillator, as shown in Figure 22-1.

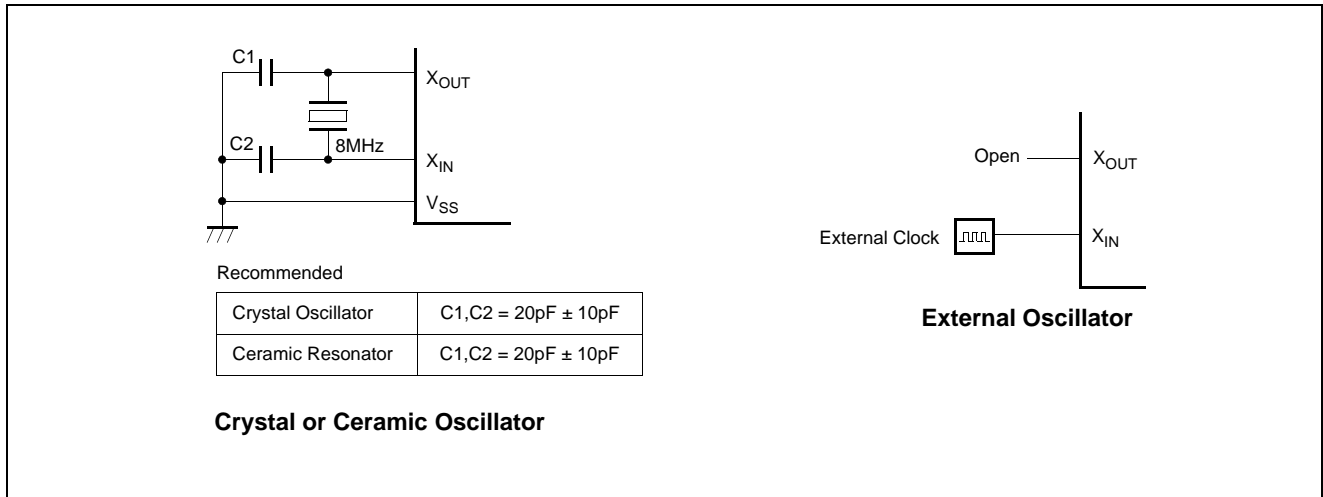


Figure 22-1 Oscillation Circuit

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

In addition, see Figure 22-2 for the layout of the crystal.

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

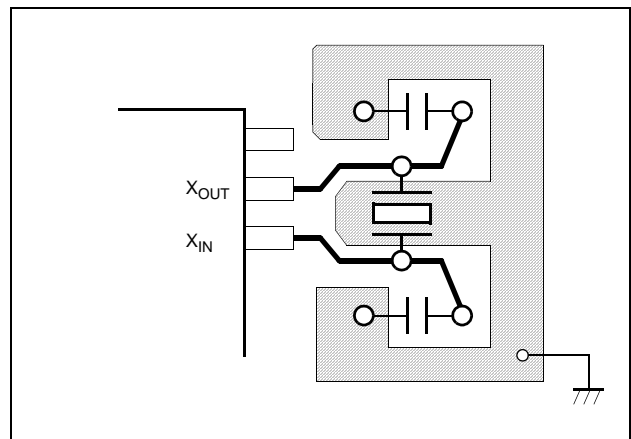


Figure 22-2 Layout of Oscillator PCB circuit

23. RESET

The MC80F0208/16/24 have four types of reset generation procedures; they are an external reset input, a watch-dog timer reset,

power fail processor reset, and address fail reset. Table 23-1 shows on-chip hardware initialization by reset action.

| On-chip Hardware | Initial Value |
|-------------------------|---|
| Program counter (PC) | (FFFF _H) - (FFFE _H) |
| RAM page register (RPR) | 0 |
| G-flag (G) | 0 |
| Operation mode | Main-frequency clock |

| On-chip Hardware | Initial Value |
|---------------------|-------------------------------|
| Peripheral clock | Off |
| Watchdog timer | Disable |
| Control registers | Refer to Table 8-1 on page 27 |
| Power fail detector | Disable |

Table 23-1 Initializing Internal Status by Reset Action

External Reset Input

The reset input is the $\overline{\text{RESET}}$ pin, which is the input to a Schmitt Trigger. A reset is 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 23-2.

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 23-1.

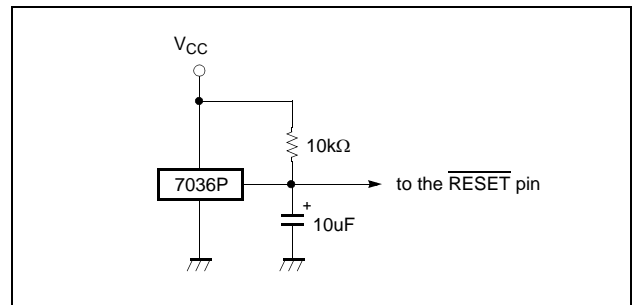


Figure 23-1 Simple Power-on-Reset Circuit

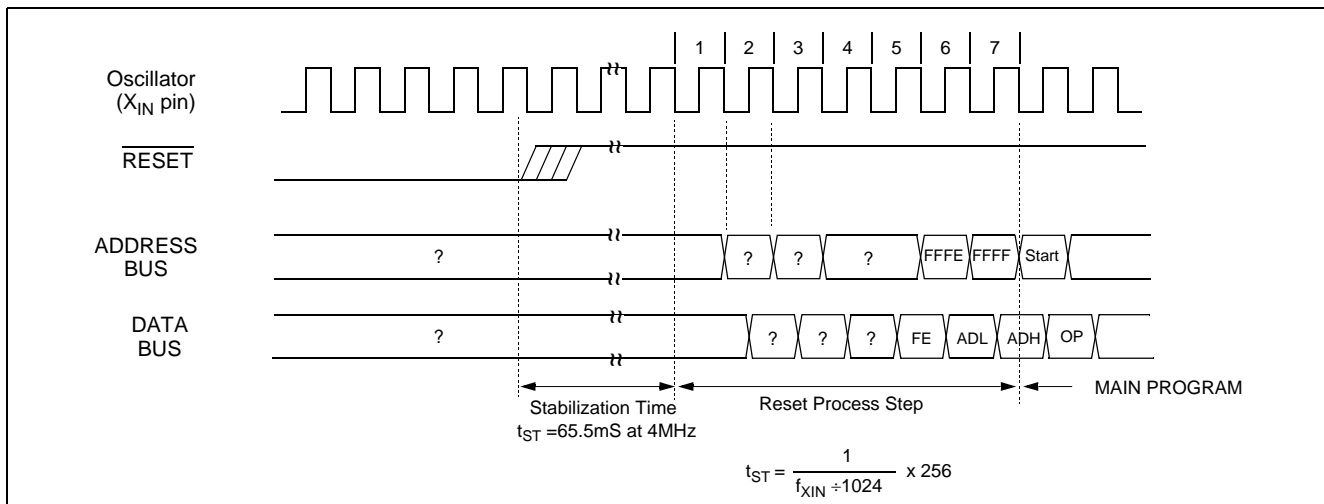


Figure 23-2 Timing Diagram after Reset

Address Fail 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.

24. POWER FAIL PROCESSOR

The MC80F0208/16/24 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 24-1 Power Fail Voltage Detector Register" on page 105.

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.

Note: User can select power fail voltage level according to PFS0, PFS1 bit of CONFIG register(703FH) at the FLASH (MC80F0208/16/24) but must select the power fail voltage level to define PFD option of "Mask Order & Verification Sheet" at the mask chip(MC80C0208/16/24), because the power fail voltage level of mask chip (MC80C0208/16/24) is determined according to mask option.

Note: If power fail voltage is selected to 2.4V or 2.7V on below 3V operation, MCU is freezed at all the times.

| Power Fail Function | FLASH | MASK |
|---------------------|----------------------|-------------|
| Enable/Disable | PFDEN flag | PFDEN flag |
| Level Selection | PFS0 bit PFS1 bit | Mask option |

Table 24-1 Power fail processor

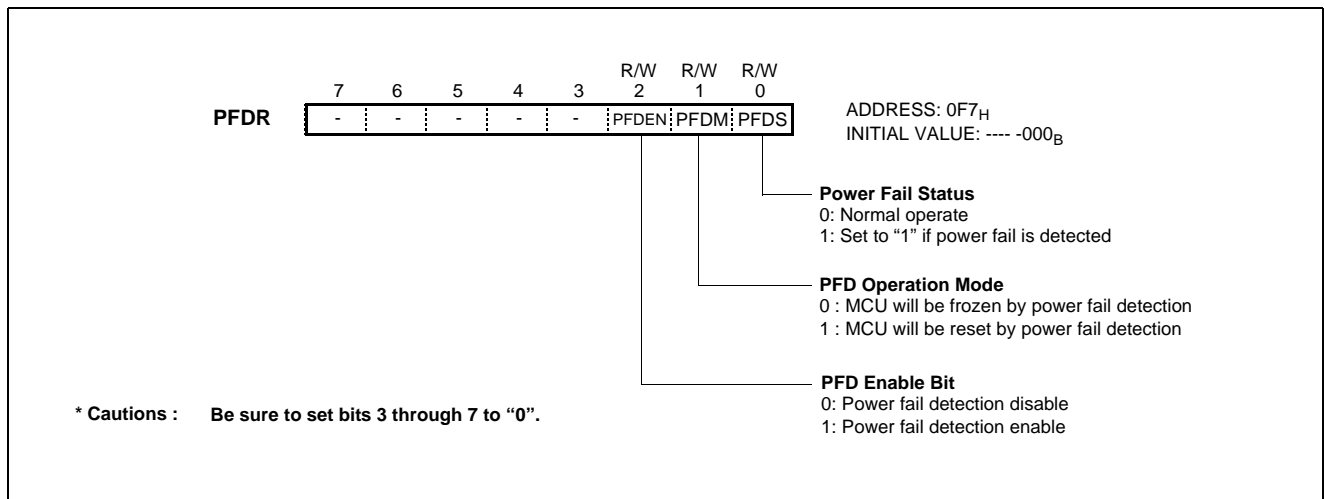


Figure 24-1 Power Fail Voltage Detector Register

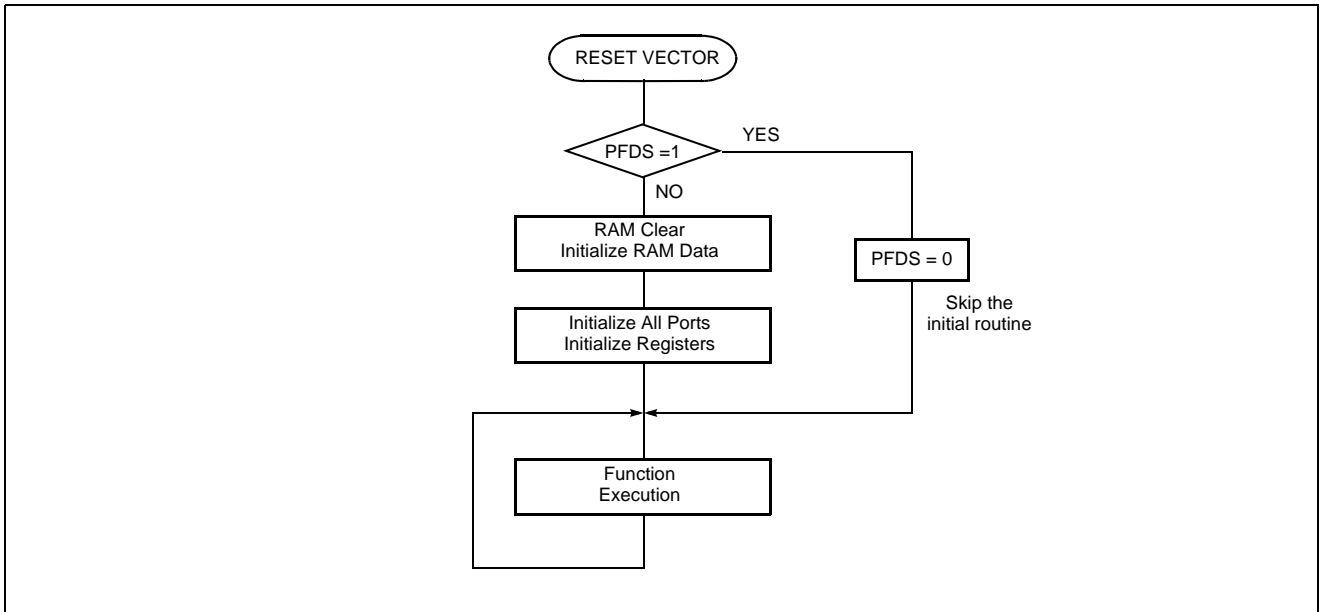


Figure 24-2 Example S/W of Reset flow by Power fail

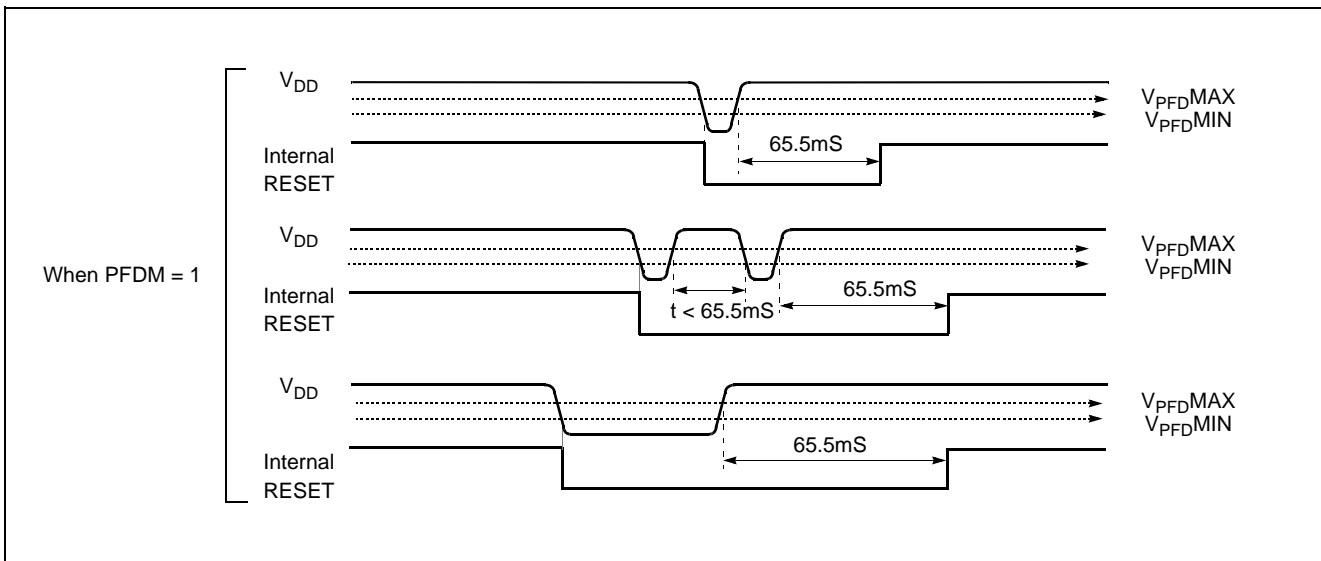


Figure 24-3 Power Fail Processor Situations (at 4MHz operation)

25. FLASH PROGRAMMING

The Device Configuration Area can be programmed or left un-programmed to select device configuration such as security bit. This area is not accessible during normal execution but is read-

able and writable during FLASH program / verify mode. The Device Configuration Area register is located at the address 20FF_H.

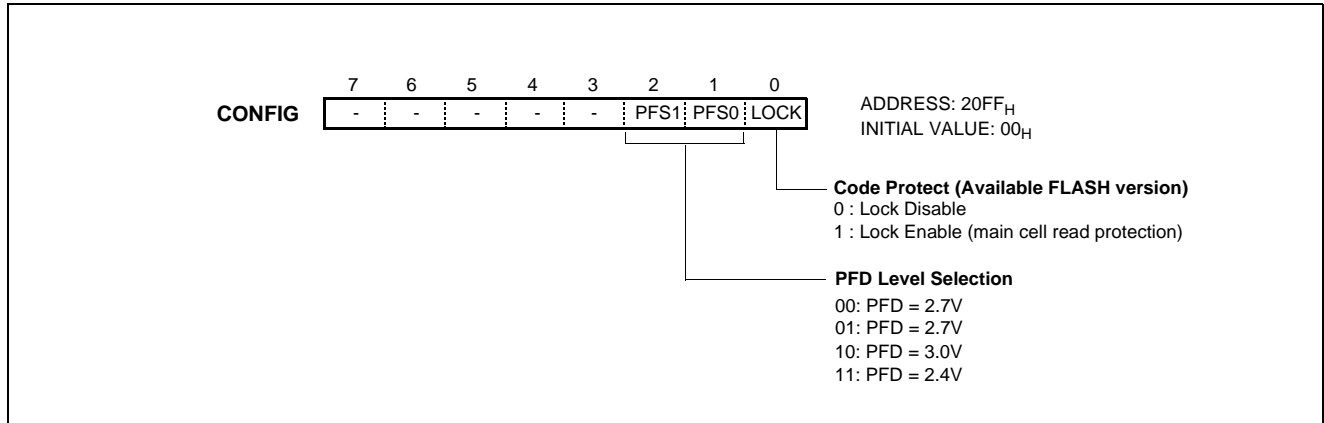


Figure 25-1 Device Configuration Area

25.1 Lock bit

The lock bit exists in Device Configuration Area register. If lock bit is programmed and user tries to read FLASH memory cell, the output data from the data port is 5AH that means the normal pro-

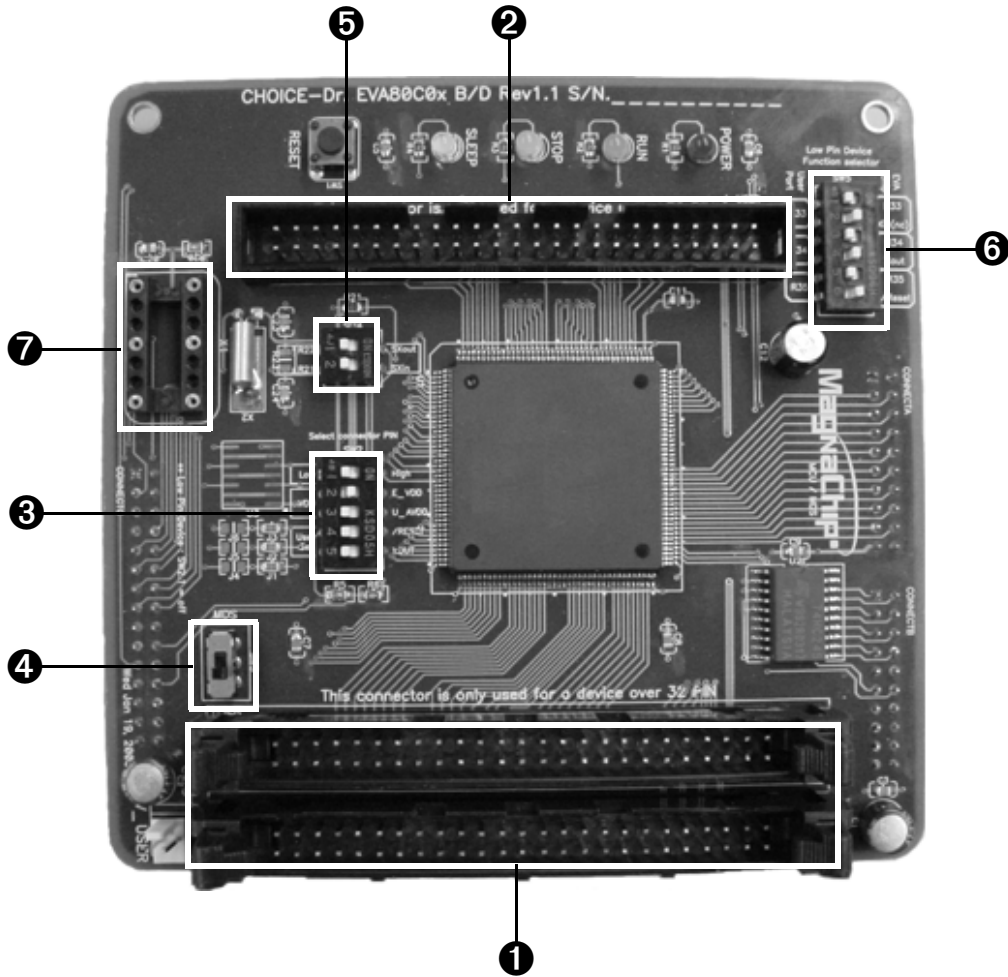
tection operation of user program data. Once the lock bit is programmed, the user can't modify and read the data of user program area.

25.2 Power Fail Detector

The power fail detection provides 3 level of detection, 2.4V, 2.7V and 3.0V. The default level of detection is 2.7V and this level is applied if user does not select the specific level in FLASH pro-

gramming S/W tools. For more information, Refer to "24. POWER FAIL PROCESSOR" on page 105.

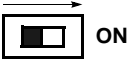


26. Emulator EVA. Board Setting



| J_USERB | | J_USERA | |
|---------|----|---------|----|
| VDD | 1 | VDD | 1 |
| AVDD | 3 | R70 | 3 |
| GND | 5 | R72 | 5 |
| R67 | 7 | R74 | 7 |
| R65 | 9 | R76 | 9 |
| R63 | 11 | GND | 11 |
| R61 | 13 | R80 | 13 |
| GND | 15 | R82 | 15 |
| R57 | 17 | R84 | 17 |
| R55 | 19 | R86 | 19 |
| R53 | 21 | GND | 21 |
| R51 | 23 | R00 | 23 |
| GND | 25 | R02 | 25 |
| R47 | 27 | R04 | 27 |
| R45 | 29 | R06 | 29 |
| R43 | 31 | GND | 31 |
| R41 | 33 | R10 | 33 |
| GND | 35 | R12 | 35 |
| R37 | 37 | R14 | 37 |
| R35 | 39 | R16 | 39 |
| R33 | 41 | GND | 41 |
| R31 | 43 | R20 | 43 |
| GND | 45 | R22 | 45 |
| U_Reset | 47 | R24 | 47 |
| GND | 49 | R26 | 49 |
| | | VDD | 2 |
| | | R71 | 4 |
| | | R73 | 6 |
| | | R75 | 8 |
| | | R77 | 10 |
| | | GND | 12 |
| | | R81 | 14 |
| | | R83 | 16 |
| | | R85 | 18 |
| | | R87 | 20 |
| | | GND | 22 |
| | | R01 | 24 |
| | | R03 | 26 |
| | | R05 | 28 |
| | | R07 | 30 |
| | | GND | 32 |
| | | R11 | 34 |
| | | R13 | 36 |
| | | R15 | 38 |
| | | R17 | 40 |
| | | GND | 42 |
| | | R21 | 44 |
| | | R23 | 46 |
| | | R25 | 48 |
| | | R27 | 50 |

DIP Switch and VR Setting

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

| DIP S/W | Description | ON/OFF Setting |
|----------|---|---|
| ① | - This connector is only used for a device over 32 PIN. | For the MC80F0208/16/24. |
| ② | - This connector is only used for a device under 32 PIN. | For the MC80F0204. |
| ③ SW2 | 1  ON Eva. select switch | Must be ON position. ON : For the MC80F0208/16/24. OFF : For the MC80F0204. |
| | 2 3  ON OFF Use Eva. V _{DD} OFF ON Use User's AV _{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 |
| | ④ SW3 | 1  MDS ↑ USER Use MDS Power MDS ↓ USER Use User's Power |
| ⑤ SW4 | 1 This switch select the R22 or SX _{OUT} . 2 This switch select the R21 or SX _{IN} . | These switches 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 (MC80F0208/16/24). |

| DIP S/W | | Description | ON/OFF Setting |
|----------|--------|--|---|
| ⑥ SW5 | 1 2 | These switches select the R33 or X _{IN} | This switch select the Normal I/O port(on&off) or special function select(off&on). It is reserved for the MC80F0204. ON & OFF : R33,R34,R35 Port selected. OFF & ON : X _{OUT} , X _{IN} , /Reset selected. Don't care (MC80F0208/16/24). |
| | 3 4 | These switches select the R34 or X _{OUT} | |
| | 5 6 | These switches select the R35 or /Reset | |
| ⑦ | - | This is External oscillation socket(CAN Type. OSC) | This is for External Clock(CAN Type. OSC). |

27. IN-SYSTEM PROGRAMMING (ISP)

27.1 Getting Started / Installation

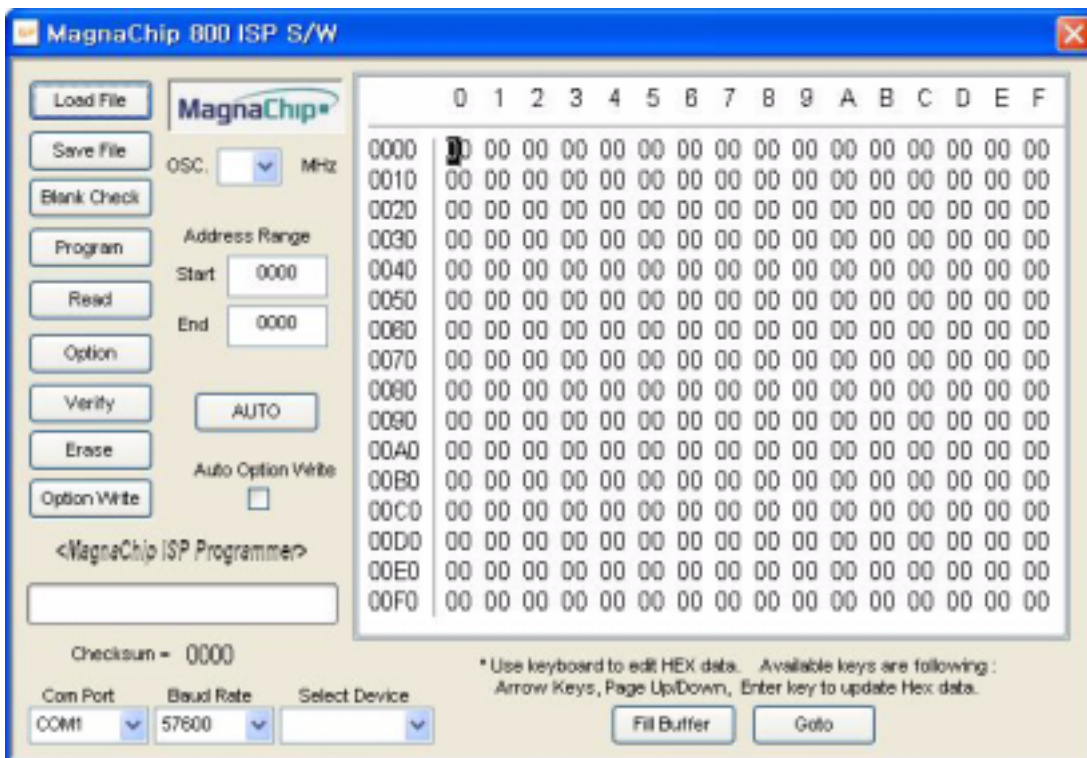
The following section details the procedure for accomplishing the installation procedure.

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.

27.2 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 Motorola 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 | Program 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 1. ISP Function Description

27.3 Hardware Conditions to Enter the ISP Mode

The In-System Programming (ISP) is performed without removing the microcontroller from the target 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 ALEB high, RST/V_{PP} as +9V, and ACLK0 with the OSC. 1.8432MHz. The ISP function uses five pins: TxD0, RxD0, ALEB, ACLK0 and RST/V_{PP}.

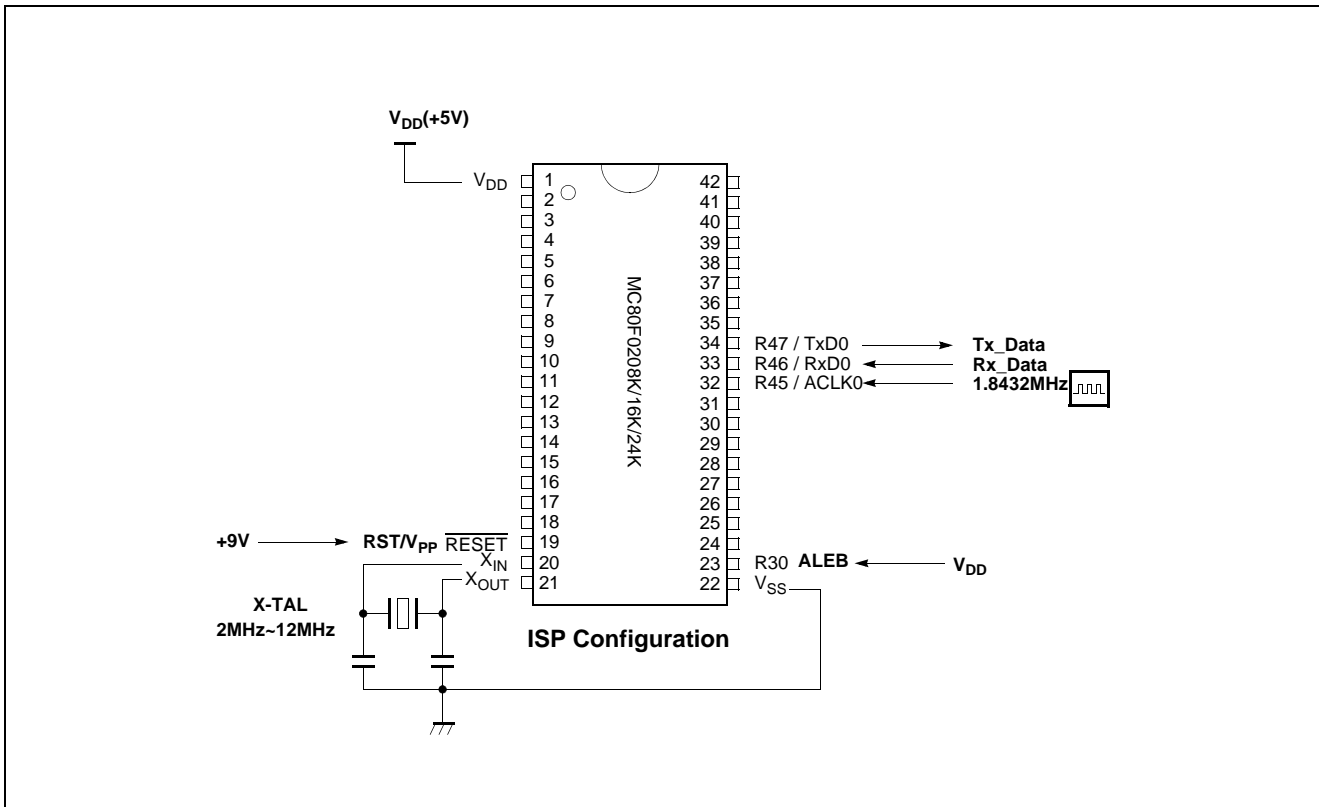


Figure 27-1 ISP Configuration

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

- The ACLK0 must be connected to the specified oscillator.
- Connect the +9V to RST/V_{pp} pin directly.
- The ALEB pin must be pulled high.
- The main clk must be higher than 2MHz.

27.4 Reference ISP Circuit diagram

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

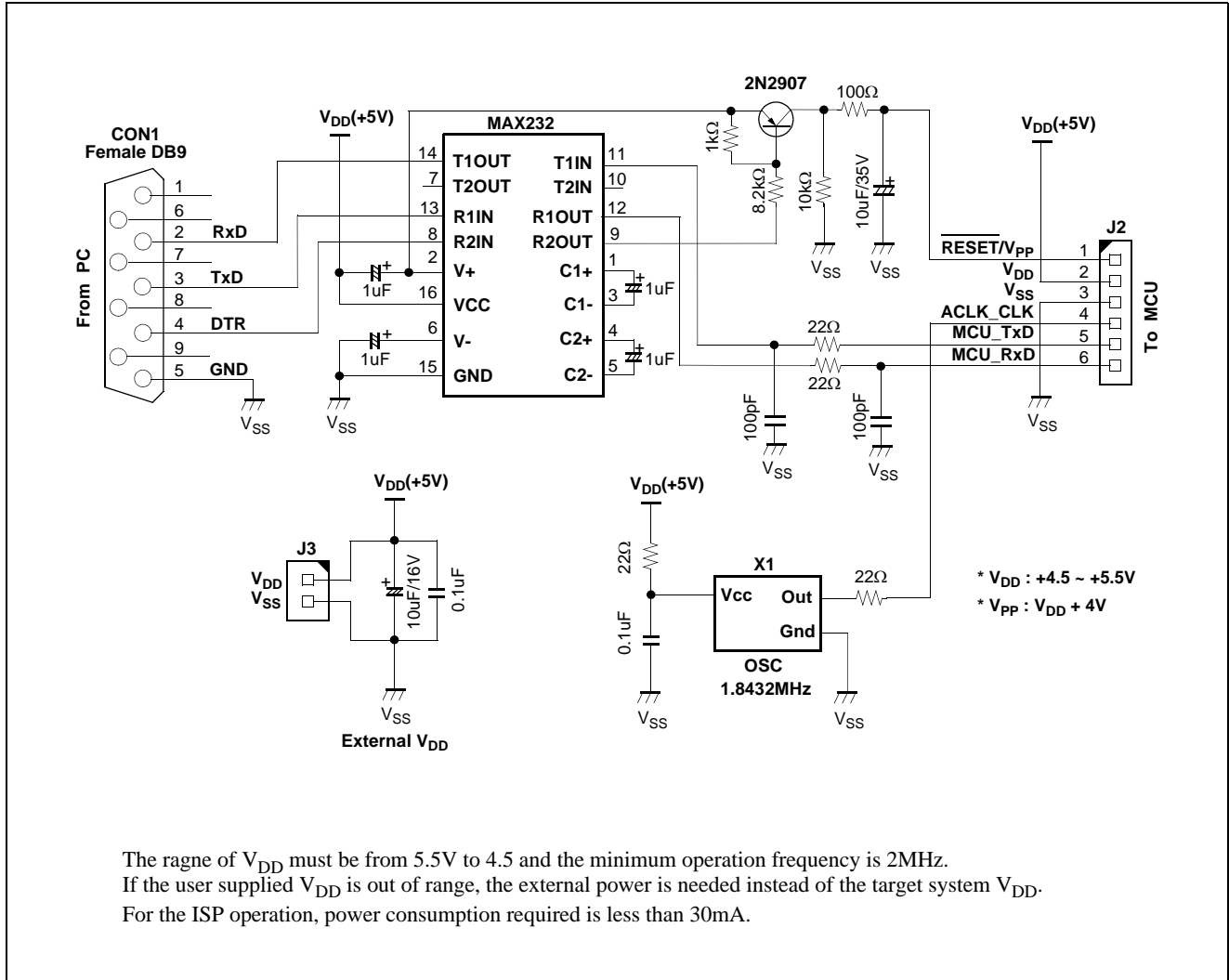


Figure 27-2 Reference ISP Circuit Diagram

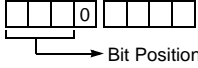
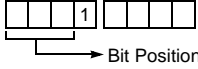


Figure 27-3 MagnaChip supplied ISP Board

APPENDIX

A. INSTRUCTION

A.1 Terminology List

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

A.2 Instruction Map

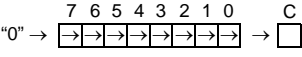

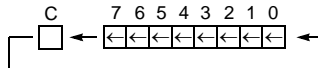
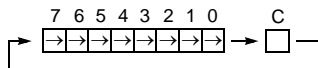
| LOW HIGH | 0000 00 | 00001 01 | 00010 02 | 00011 03 | 00100 04 | 00101 05 | 00110 06 | 00111 07 | 01000 08 | 01001 09 | 01010 0A | 01011 0B | 01100 0C | 01101 0D | 01110 0E | 01111 0F |
|-------------|------------|----------------|------------------|-------------------|----------------|-------------|-------------|-------------|-------------|-------------|-------------|---------------|-------------|--------------|-------------|----------------|
| 000 | - | SET1 dp.bit | BBS A.bit,rel | BBS dp.bit,rel | ADC #imm | ADC dp | ADC dp+X | ADC !abs | ASL A | ASL dp | 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 |

A.3 Instruction Set

Arithmetic / Logic Operation

| No. | Mnemonic | Op Code | Byte No | Cycle No | Operation | Flag NVGBHIZC |
|-----|----------------|---------|---------|----------|---|------------------|
| 1 | ADC #imm | 04 | 2 | 2 | Add with carry. | NVGBHIZC |
| 2 | ADC dp | 05 | 2 | 3 | $A \leftarrow (A) + (M) + C$ | NV--H-ZC |
| 3 | ADC dp + X | 06 | 2 | 4 | | |
| 4 | ADC !abs | 07 | 3 | 4 | | |
| 5 | ADC !abs + Y | 15 | 3 | 5 | | |
| 6 | ADC [dp + X] | 16 | 2 | 6 | | |
| 7 | ADC [dp] + Y | 17 | 2 | 6 | | |
| 8 | ADC { X } | 14 | 1 | 3 | | |
| 9 | AND #imm | 84 | 2 | 2 | | |
| 10 | AND dp | 85 | 2 | 3 | | |
| 11 | AND dp + X | 86 | 2 | 4 | | |
| 12 | AND !abs | 87 | 3 | 4 | | |
| 13 | AND !abs + Y | 95 | 3 | 5 | | |
| 14 | AND [dp + X] | 96 | 2 | 6 | | |
| 15 | AND [dp] + Y | 97 | 2 | 6 | | |
| 16 | AND { X } | 94 | 1 | 3 | | |
| 17 | ASL A | 08 | 1 | 2 | Arithmetic shift left C 7 6 5 4 3 2 1 0 $\square \leftarrow \leftarrow \leftarrow \leftarrow \leftarrow \leftarrow \leftarrow \leftarrow \leftarrow$ ← "0" | N-----ZC |
| 18 | ASL dp | 09 | 2 | 4 | | |
| 19 | ASL dp + X | 19 | 2 | 5 | | |
| 20 | ASL !abs | 18 | 3 | 5 | | |
| 21 | CMP #imm | 44 | 2 | 2 | Compare accumulator contents with memory contents $(A) - (M)$ | N-----ZC |
| 22 | CMP dp | 45 | 2 | 3 | | |
| 23 | CMP dp + X | 46 | 2 | 4 | | |
| 24 | CMP !abs | 47 | 3 | 4 | | |
| 25 | CMP !abs + Y | 55 | 3 | 5 | | |
| 26 | CMP [dp + X] | 56 | 2 | 6 | | |
| 27 | CMP [dp] + Y | 57 | 2 | 6 | | |
| 28 | CMP { X } | 54 | 1 | 3 | | |
| 29 | CMPX #imm | 5E | 2 | 2 | Compare X contents with memory contents $(X) - (M)$ | N-----ZC |
| 30 | CMPX dp | 6C | 2 | 3 | | |
| 31 | CMPX !abs | 7C | 3 | 4 | | |
| 32 | CMPY #imm | 7E | 2 | 2 | Compare Y contents with memory contents $(Y) - (M)$ | N-----ZC |
| 33 | CMPY dp | 8C | 2 | 3 | | |
| 34 | CMPY !abs | 9C | 3 | 4 | | |
| 35 | COM dp | 2C | 2 | 4 | 1'S Complement : $(dp) \leftarrow \sim(dp)$ | N-----Z- |
| 36 | DAA | 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 $M \leftarrow (M) - 1$ | N-----Z- |
| 39 | DEC dp | A9 | 2 | 4 | | N-----Z- |
| 40 | DEC dp + X | B9 | 2 | 5 | | N-----Z- |
| 41 | DEC !abs | B8 | 3 | 5 | | N-----Z- |
| 42 | DEC X | AF | 1 | 2 | | N-----Z- |
| 43 | DEC Y | BE | 1 | 2 | | N-----Z- |

| No. | Mnemonic | Op Code | Byte No | Cycle No | Operation | Flag NVGBHIZC |
|-----|----------------|---------|---------|----------|---|------------------|
| 44 | DIV | 9B | 1 | 12 | Divide : YA / X Q: A, R: Y | NV--H-Z- |
| 45 | EOR #imm | A4 | 2 | 2 | Exclusive OR $A \leftarrow (A) \oplus (M)$ | N-----Z- |
| 46 | EOR dp | A5 | 2 | 3 | | |
| 47 | EOR dp + X | A6 | 2 | 4 | | |
| 48 | EOR !abs | A7 | 3 | 4 | | |
| 49 | EOR !abs + Y | B5 | 3 | 5 | | |
| 50 | EOR [dp + X] | B6 | 2 | 6 | | |
| 51 | EOR [dp] + Y | B7 | 2 | 6 | | |
| 52 | EOR { X } | B4 | 1 | 3 | | |
| 53 | INC A | 88 | 1 | 2 | Increment $M \leftarrow (M) + 1$ | N-----ZC |
| 54 | INC dp | 89 | 2 | 4 | | N-----Z- |
| 55 | INC dp + X | 99 | 2 | 5 | | N-----Z- |
| 56 | INC !abs | 98 | 3 | 5 | | N-----Z- |
| 57 | INC X | 8F | 1 | 2 | | N-----Z- |
| 58 | INC Y | 9E | 1 | 2 | | N-----Z- |
| 59 | LSR A | 48 | 1 | 2 | Logical shift right "0" \rightarrow  \rightarrow  | N-----ZC |
| 60 | LSR dp | 49 | 2 | 4 | | |
| 61 | LSR dp + X | 59 | 2 | 5 | | |
| 62 | LSR !abs | 58 | 3 | 5 | | |
| 63 | MUL | 5B | 1 | 9 | Multiply : YA \leftarrow Y \times A | N-----Z- |
| 64 | OR #imm | 64 | 2 | 2 | Logical OR $A \leftarrow (A) \vee (M)$ | N-----Z- |
| 65 | OR dp | 65 | 2 | 3 | | |
| 66 | OR dp + X | 66 | 2 | 4 | | |
| 67 | OR !abs | 67 | 3 | 4 | | |
| 68 | OR !abs + Y | 75 | 3 | 5 | | |
| 69 | OR [dp + X] | 76 | 2 | 6 | | |
| 70 | OR [dp] + Y | 77 | 2 | 6 | | |
| 71 | OR { X } | 74 | 1 | 3 | | |
| 72 | ROL A | 28 | 1 | 2 | Rotate left through Carry  | N-----ZC |
| 73 | ROL dp | 29 | 2 | 4 | | |
| 74 | ROL dp + X | 39 | 2 | 5 | | |
| 75 | ROL !abs | 38 | 3 | 5 | | |
| 76 | ROR A | 68 | 1 | 2 | Rotate right through Carry  | N-----ZC |
| 77 | ROR dp | 69 | 2 | 4 | | |
| 78 | ROR dp + X | 79 | 2 | 5 | | |
| 79 | ROR !abs | 78 | 3 | 5 | | |
| 80 | SBC #imm | 24 | 2 | 2 | Subtract with Carry $A \leftarrow (A) - (M) - \sim(C)$ | NV--HZC |
| 81 | SBC dp | 25 | 2 | 3 | | |
| 82 | SBC dp + X | 26 | 2 | 4 | | |
| 83 | SBC !abs | 27 | 3 | 4 | | |
| 84 | SBC !abs + Y | 35 | 3 | 5 | | |
| 85 | SBC [dp + X] | 36 | 2 | 6 | | |
| 86 | SBC [dp] + Y | 37 | 2 | 6 | | |
| 87 | SBC { X } | 34 | 1 | 3 | | |
| 88 | TST dp | 4C | 2 | 3 | Test memory contents for negative or zero, (dp) - 00 _H | N-----Z- |
| 89 | XCN | CE | 1 | 5 | Exchange nibbles within the accumulator $A_7 \sim A_4 \leftrightarrow A_3 \sim A_0$ | N-----Z- |

Register / Memory Operation

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

16-BIT operation

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

Bit Manipulation

| No. | Mnemonic | Op Code | Byte No | Cycle No | Operation | Flag NVGBHIZC |
|-----|-------------|---------|---------|----------|--|------------------|
| 1 | AND1 M.bit | 8B | 3 | 4 | Bit AND C-flag : $C \leftarrow (C) \wedge (M.bit)$ | -----C |
| 2 | AND1B M.bit | 8B | 3 | 4 | Bit AND C-flag and NOT : $C \leftarrow (C) \wedge \sim(M.bit)$ | -----C |
| 3 | BIT dp | 0C | 2 | 4 | Bit test A with memory : | MM----Z- |
| 4 | BIT !abs | 1C | 3 | 5 | $Z \leftarrow (A) \wedge (M), N \leftarrow (M_7), V \leftarrow (M_6)$ | |
| 5 | CLR1 dp.bit | y1 | 2 | 4 | Clear bit : $(M.bit) \leftarrow "0"$ | ----- |
| 6 | CLRA1 A.bit | 2B | 2 | 2 | Clear A bit : $(A.bit) \leftarrow "0"$ | ----- |
| 7 | CLRC | 20 | 1 | 2 | Clear C-flag : $C \leftarrow "0"$ | -----0 |
| 8 | CLRG | 40 | 1 | 2 | Clear G-flag : $G \leftarrow "0"$ | --0----- |
| 9 | CLRV | 80 | 1 | 2 | Clear V-flag : $V \leftarrow "0"$ | -0--0--- |
| 10 | EOR1 M.bit | AB | 3 | 5 | Bit exclusive-OR C-flag : $C \leftarrow (C) \oplus (M.bit)$ | -----C |
| 11 | EOR1B M.bit | AB | 3 | 5 | Bit exclusive-OR C-flag and NOT : $C \leftarrow (C) \oplus \sim(M.bit)$ | -----C |
| 12 | LDC M.bit | CB | 3 | 4 | Load C-flag : $C \leftarrow (M.bit)$ | -----C |
| 13 | LDCB M.bit | CB | 3 | 4 | Load C-flag with NOT : $C \leftarrow \sim(M.bit)$ | -----C |
| 14 | NOT1 M.bit | 4B | 3 | 5 | Bit complement : $(M.bit) \leftarrow \sim(M.bit)$ | ----- |
| 15 | OR1 M.bit | 6B | 3 | 5 | Bit OR C-flag : $C \leftarrow (C) \vee (M.bit)$ | -----C |
| 16 | OR1B M.bit | 6B | 3 | 5 | Bit OR C-flag and NOT : $C \leftarrow (C) \vee \sim(M.bit)$ | -----C |
| 17 | SET1 dp.bit | x1 | 2 | 4 | Set bit : $(M.bit) \leftarrow "1"$ | ----- |
| 18 | SETA1 A.bit | 0B | 2 | 2 | Set A bit : $(A.bit) \leftarrow "1"$ | ----- |
| 19 | SETC | A0 | 1 | 2 | Set C-flag : $C \leftarrow "1"$ | -----1 |
| 20 | SETG | C0 | 1 | 2 | Set G-flag : $G \leftarrow "1"$ | --1----- |
| 21 | STC M.bit | EB | 3 | 6 | Store C-flag : $(M.bit) \leftarrow C$ | ----- |
| 22 | TCLR1 !abs | 5C | 3 | 6 | Test and clear bits with A : $A - (M), (M) \leftarrow (M) \wedge \sim(A)$ | N-----Z- |
| 23 | TSET1 !abs | 3C | 3 | 6 | Test and set bits with A : $A - (M), (M) \leftarrow (M) \vee (A)$ | N-----Z- |

Branch / Jump Operation

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

Control Operation & Etc.

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

B. MASK ORDER SHEET

Mask Order & Verification Sheet

MC80C02 - MC

Customer should write inside thick line box.

1. Customer Information

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

2. Device Information

| | | |
|-----------|---|--|
| Package | <input type="checkbox"/> 44MQFP <input type="checkbox"/> 42SDIP | |
| Mask Data | File Name | () .OTP |
| | ROM Size (bytes) | <input type="checkbox"/> 8K <input type="checkbox"/> 16K <input type="checkbox"/> 24K |
| | Check Sum | () |
| | * PFD Option <input type="checkbox"/> 3.0V <input type="checkbox"/> 2.7V <input type="checkbox"/> 2.4V <input type="checkbox"/> Not use | Set "00H" in blanked area (24K) A000 H (16K) C000 H (8K) E000 H .OTP file FFFFH |

3. Marking Specification

(Please check mark √ into)

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

Customer's part number

4. Delivery Schedule

| | Date | Quantity | MagnaChip Confirmation |
|-----------------|--|----------|------------------------|
| Customer sample | YYYY MM DD . . | pcs | |
| Risk order | YYYY MM DD . . | pcs | |

5. ROM Code Verification

Please confirm out verification data.

| | |
|--------------------|--|
| Verification date: | YYYY MM DD . . |
| Check sum: | |
| Tel: | Fax: |
| E-mail address: | |
| Name & Signature: | |

| | |
|--|--|
| Approval date: | YYYY MM DD . . |
| <i>I agree with your verification data and confirm you to make mask set.</i> | |
| Tel: | Fax: |
| E-mail address: | |
| Name & Signature: | |

