MSM66587 Family

User's Manual

CMOS 16-bit microcontroller

SECOND EDITION

ISSUE DATE: Jan. 2000

FEUL66587-02

NOTICE

1. The information contained herein can change without notice owing to product and/or technical improvements. Before using the product, please make sure that the information being referred to is up-to-date.

2. The outline of action and examples for application circuits described herein have been chosen as an explanation for the standard action and performance of the product. When planning to use the product, please ensure that the external conditions are reflected in the actual circuit, assembly, and program designs.

3. When designing your product, please use our product below the specified maximum ratings and within the specified operating ranges including, but not limited to, operating voltage, power dissipation, and operating temperature.

4. Oki assumes no responsibility or liability whatsoever for any failure or unusual or unexpected operation resulting from misuse, neglect, improper installation, repair, alteration or accident, improper handling, or unusual physical or electrical stress including, but not limited to, exposure to parameters beyond the specified maximum ratings or operation outside the specified operating range.

5. Neither indemnity against nor license of a third party's industrial and intellectual property right, etc. is granted by us in connection with the use of the product and/or the information and drawings contained herein. No responsibility is assumed by us for any infringement of a third party's right which may result from the use thereof.

6. The products listed in this document are intended for use in general electronics equipment for commercial applications (e.g., office automation, communication equipment, measurement equipment, consumer electronics, etc.). These products are not authorized for use in any system or application that requires special or enhanced quality and reliability characteristics nor in any system or application where the failure of such system or application may result in the loss or damage of property, or death or injury to humans.
Such applications include, but are not limited to, traffic and automotive equipment, safety devices, aerospace equipment, nuclear power control, medical equipment, and life-support systems.

7. Certain products in this document may need government approval before they can be exported to particular countries. The purchaser assumes the responsibility of determining the legality of export of these products and will take appropriate and necessary steps at their own expense for these.

8. No part of the contents contained herein may be reprinted or reproduced without our prior permission.

9. MS-DOS is a registered trademark of Microsoft Corporation.

Printed in Japan

Preface

This user's manual describes the hardware of Oki-original CMOS 16-bit microcontrollers MSM66587 family. In addition to this manual, Oki also provides the following manuals which should be read with regard to the MSM66587 family.

nX-8/500S Core Instruction Manual

- nX-8/500S core instruction set
- Addressing modes

CC665S User's Manual

- Optimized compiler CC665S operation
- C-language specifications in CC665S

CL665S User's Manual

- Compiler loader CL665S operation

RTL665S Run Time Library Reference

- C run time library explanation

MAC66K Assembler Package User’s Manual

- Package overview
- RAS66K (relocatable assembler) operation
- RAS66K assembly language explanation
- RL66K (linker) operation
- LIB66K (librarian) operation
- OH66K (object converter) operation

Macroprocessor MP User’s Manual

- MP operation
- Macro language

Ultra-66K User’s Manual

- Ultra-66K (Emulator) explanation
- PathFinder-66K (Debugger) explanation

PW66K Flash Writer System User’s Manual

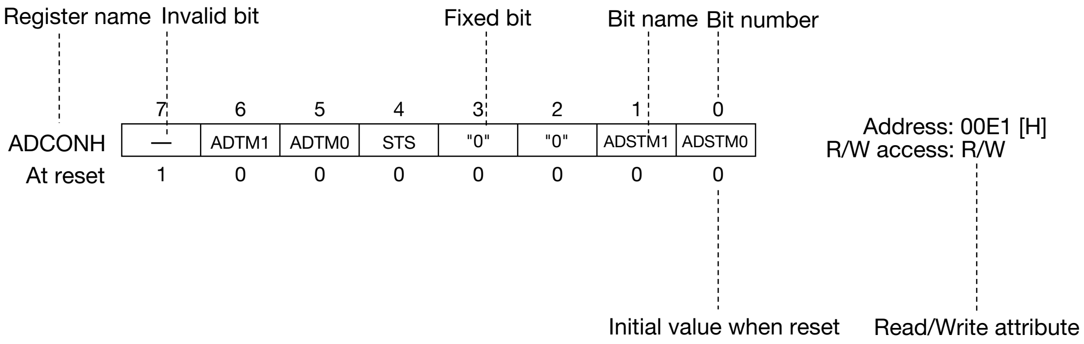
- PW66K Flash Writer System operation

This document is subject to change without notice.

Notation

Classification	Notation	Description
■ Numeric value	xxH	Represents a hexadecimal number
	xxb	Represents a binary number
■ Unit	Word, W	1 word = 16 bits
	byte, B	1 byte = 2 nibbles = 8 bits
	nibble, N	1 nibble = 4 bits
	mega-, M	10^6
	kilo-, K	$2^{10} = 1024$
	kilo-, k	$10^3 = 1000$
	mil-, m	10^{-3}
	micro-, µ	10^{-6}
	nano-, n	10^{-9}
	second, s	second
	KB	1KB = 1 kilobyte = 1024 bytes
	MB	1MB = 1 megabyte = 2^{20} bytes = 1,048,576 bytes
■ Terminology	"H" level	The signal level of the high side of the voltage; indicates the voltage level of V_{IH} and V_{OH} described in the electrical characteristics.
	"L" level	The signal level of the low side of the voltage; indicates voltage level of V_{IL} and V_{OL} described in the electrical characteristics.
	Opcode trap	Operation code trap. Occurs when an empty area that has not been assigned an instruction is fetched, or when an instruction code combination that does not contain an instruction is addressed.

■ Register description



Invalid bit : Indicates that the bit does not exist. Writing into this bit is invalid.

Fixed bit : When writing, always write the specified value. If read, the specified value will be read. Values of fixed bits are specified as "0" or "1."

Read/write attribute : R indicates that reading is possible and W indicates that writing is possible.

Table of Contents

Chapter 1

Overview

1. Overview

The MSM66587 family of devices are high-performance CMOS 16-bit microcontrollers that integrate a 16-bit CPU (nX-8/500S), ROM, RAM, 8-bit A/D converter, serial port, timers, and PWM. They also allow ROM and RAM to be expanded externally.
The MSM66P587 is of OTP (One-Time PROM) version, the MSM66Q587 is of Flash EEPROM version, and the MSM66X587A is of version with no ROM.

1.1 Features

- Powerful instruction set
  - Instruction set superior in orthogonal matrics
  - 8/16-bit arithmetic instructions
  - Multiply/divide instructions
  - Bit manipulation instructions
  - Bit logical operation instructions
  - ROM table reference instructions
- Abundant addressing modes
  - Register addressing
  - Page addressing
  - Pointer register indirect addressing
  - Stack addressing
  - Immediate addressing
- Minimum instruction cycle
  - 100 ns at 20 MHz (4.5 V to 5.5 V)
  - 200 ns at 10 MHz (2.7 V to 5.5 V)
- Program memory (ROM)
  - Internal: 64KB (The MSM66X587A includes no internal ROM)
  - External: 1MB, $\overline{\text{EA}}$ pin active
- Data memory (RAM)
  - Internal: 2KB
  - External: 1022KB
- I/O ports
  - Analog input-only port: 4 lines
  - Input/output port: 80 lines Maximum (40 lines with programmable pull-up)
- Timers
  - Free-run counter: One 16-bit
  - Realtime output: Two 16-bit
  - General autoreload timer: One 8-bit
- 16-bit PWM
  - Input clock divider: 1 divider
- 8-bit serial port
  - Synchronous with BRG: 1 port
- A/D converter
  - 8-bit resolution: 4 channels
- Interrupts
  - Non-maskable 1 interrupt
  - Maskable 9 internal, 4 external (12 vectors)
  - 3-level priority
- ROM window function
- Standby modes
  - HALT mode
  - STOP mode

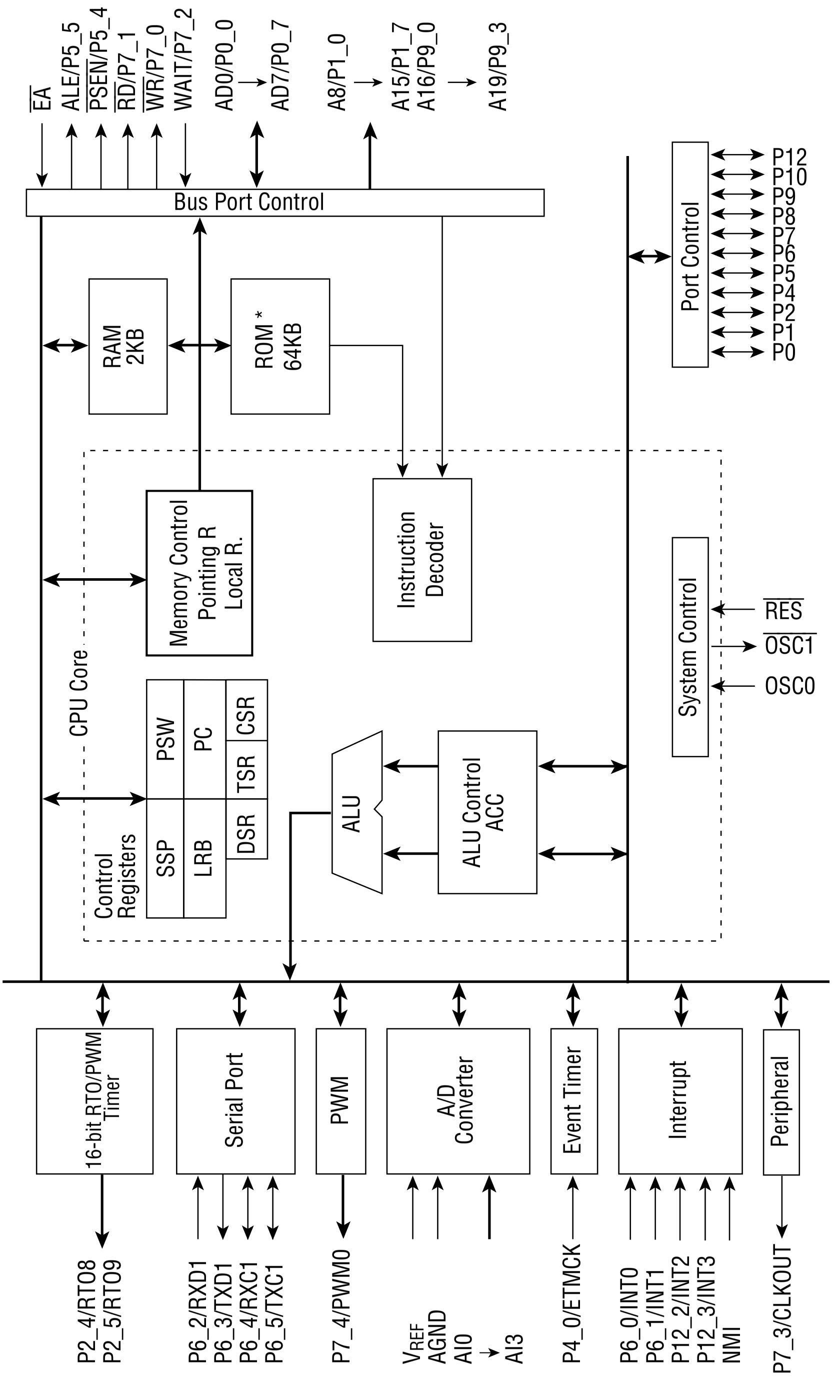
Package

100-pin plastic TQFP (TQFP100-P-1414-0.50-K)
(For external dimensions, refer to Chapter 22)

MSM66587 Family Products List

Type	MSM66587A	MSM66X587A
Operating Temperature	–30°C to +70°C	
Minimum Instruction Execution Time	100 nsec @20 MHz (4.5 to 5.5V) 200 nsec @10 MHz (2.7 to 5.5V)	
Internal ROM Size	64KB	Not included
(Maximum External ROM Size)	(1MB)	
Internal RAM Size (Maximum External RAM Size)	2KB (1MB)	
I/O Port	80 input/output lines (max) (includes 40 lines with programmable pull-up)	
Timer	16-bit free-run counter × 1ch 16-bit real-time-out × 2ch (RTO/toggle output RTO mode selectable) (Configure 2-ch RTO as 1-ch PWM)	
	16-bit PWM × 1ch	
	8-bit geneal auto-reload × 1ch (Capable of counting external events)	
	Used as 8-bit auto-reload × 1ch serial communication and baud-rate generator	
Serial Port	Synchronous × 1ch	
A/D Converter	8-bit × 4ch	
External Interrupt	Nonmaskable 1ch Maskable 4ch	
Other	OTP ROM version : MSM66P587 (2.7 to 5.5 V) FLASH ROM version : MSM66Q587 (4.5 to 5.5 V)	

1.2 Block Diagram



* The MSM66X587A includes no internal ROM

Figure 1-1 MSM66587 Family Block Diagram

1.3 Pin Configuration (Top View)

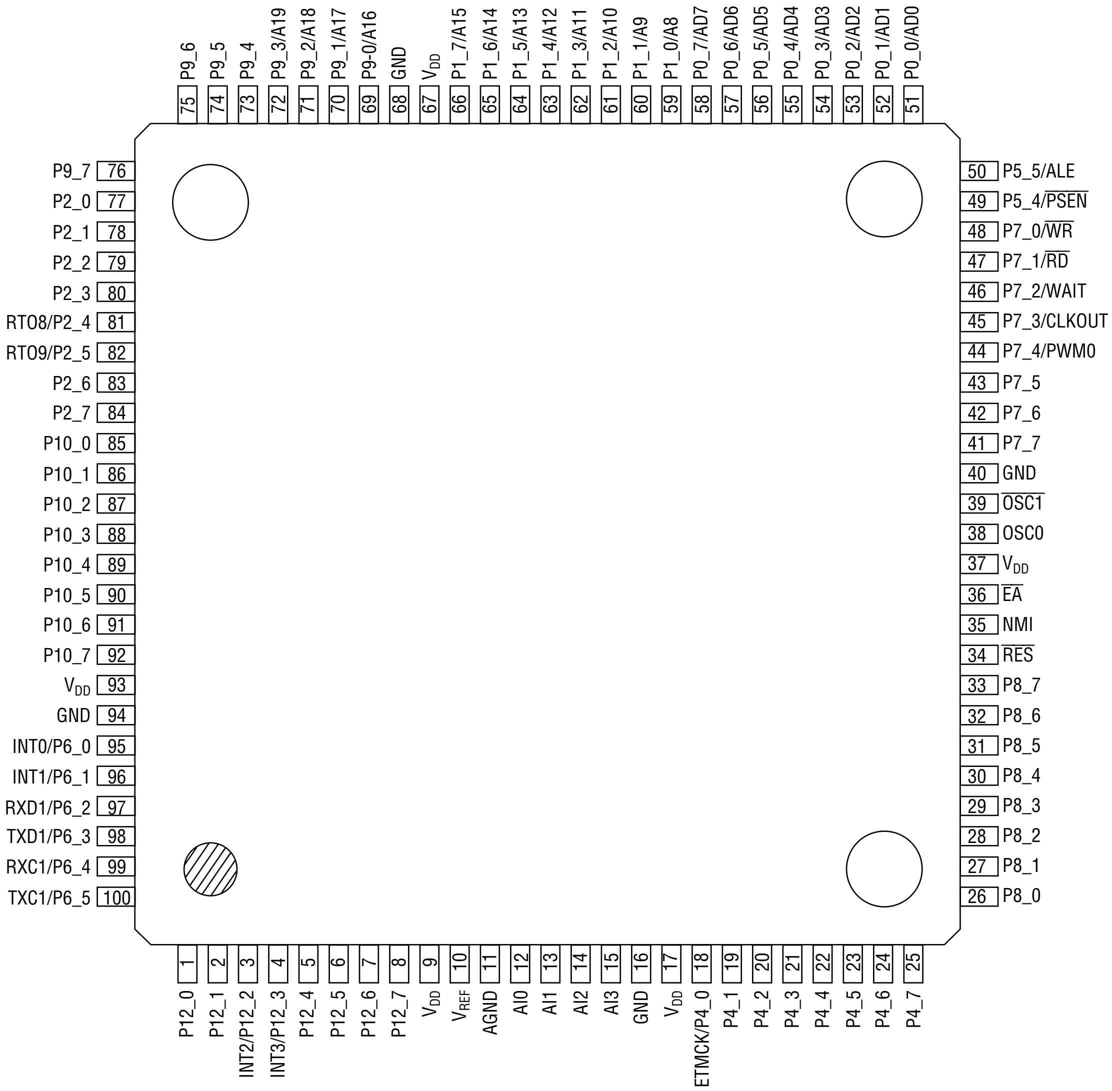


Figure 1-2 MSM66587 Family Pin Configuration (100-pin TQFP)

* For the external dimensions of the package, refer to Chapter 22, "Package Dimensions". For the connections of unused pins, refer to Section 2.22.

1.4 Basic Operation Timing

The MSM66587 family of devices integrate an OKI-original 16-bit CPU Core (nX-8/500S).
In the nX-8/500S, each instruction code consists of one byte to six bytes on the basis of 8 bits.
There are two types of instruction groups: one is NATIVE instructions and the other is COMPOSIT instruction.
The NATIVE instructions are used frequently and the COMPOSIT instructions are used to implement a wide range of addressing.
The NATIVE instructions consist of one to three bytes to achieve a high code utilization and high instruction execution efficiency.
The COMPOSIT instructions are composed of an address specification field (PREFIX: 1 to 3 bytes) and operation specification field (SUFFIX: 1 to 3 bytes).
The combination of PREFIX and SUFFIX can achieve a wide range of addressing.

The basic clock executes one state in a single pulse (CLK).
One state is 50 ns (at 20 kHz).
Execution of a single instruction takes multiple states (S2, S3,···Sn).

The number of states required for instruction execution differs by instruction. The minimum is 2 states;
The maximum is 48 states (For details refer to the nX-8/500S Core Instruction Manual).

In addition, when program memory is accessed externally, one cycle is inserted automatically for each one-byte read (fetch), and furthermore wait cycles (0 to 3) that are set by the ROM Ready Control Register (ROMRDY), are added. When data memory is accessed externally, three or four cycles (1 cycle=1 state) are inserted automatically for each one byte read or write, and moreover, wait cycles (0 to 7) that are set by the RAM Ready Control Register (RAMRDY) are added.
Figures 1-3 and 1-6 show basic timing examples.

[Note]
Some COMPOSIT instruction used to write data to external data memory outputs a meaningless $\overline{\text{RD}}$ signal (dummy RD signal) before actual access (PREFIX specifies external data memory).
One dummy RD signal is output for byte/bit access and two dummy RD signals are output for word access.
Table 1-1 and 1-2 list instructions with PREFIX and SUFFIX to output dummy RD signals when external data memory is accessed.
In the tables, * of SUFFIX describes an addressing of PREFIX.

Table 1-1 Instructions (Byte/Bit Manipulations) In Which a Dummy RD Occurs Once (PREFIX and SUFFIX Combinations)

SUFFIX

Instruction desciption	Instruction code
SB *	08+bit
RB *	00+bit
SBR *	B8
RBR *	B9
TBR *	CA
MB *.bit, C	18+bit
MBR *.bit, C	BB
MBR C, *.bit	BA
MOVB *,A	AA
MOVB *,#N8	AB
CLRB *	C7
FILLB *	D7

+

PREFIX

*	Instruction code
Rn	68+n
[X1]	B0
[DP]	B2
[DP–]	B1
[DP+]	B3
off	B5
dir	B7
N16[X1]	B8
N16[X2]	B9
n7[DP]	9B
n7[USP]	9B
[X1+A]	BA
[X1+R0]	BB

Table 1-2 Instructions (Word Manipulations) In Which a Dummy RD Occurs Twice (PREFIX and SUFFIX Combinations)

SUFFIX

Instruction desciption	Instruction code
MOVB *,A	AA
MOVB *,#N16	AB
CLR *	C7
FILL *	D7

+

PREFIX

*	Instruction code
ERn	64+n
[X1]	A0
[DP]	A2
[DP–]	A1
[DP+]	A3
off	A5
dir	A7
N16[X1]	A8
N16[X2]	A9
n7[DP]	8B
n7[USP]	8B
[X1+A]	AA
[X1+R0]	AB

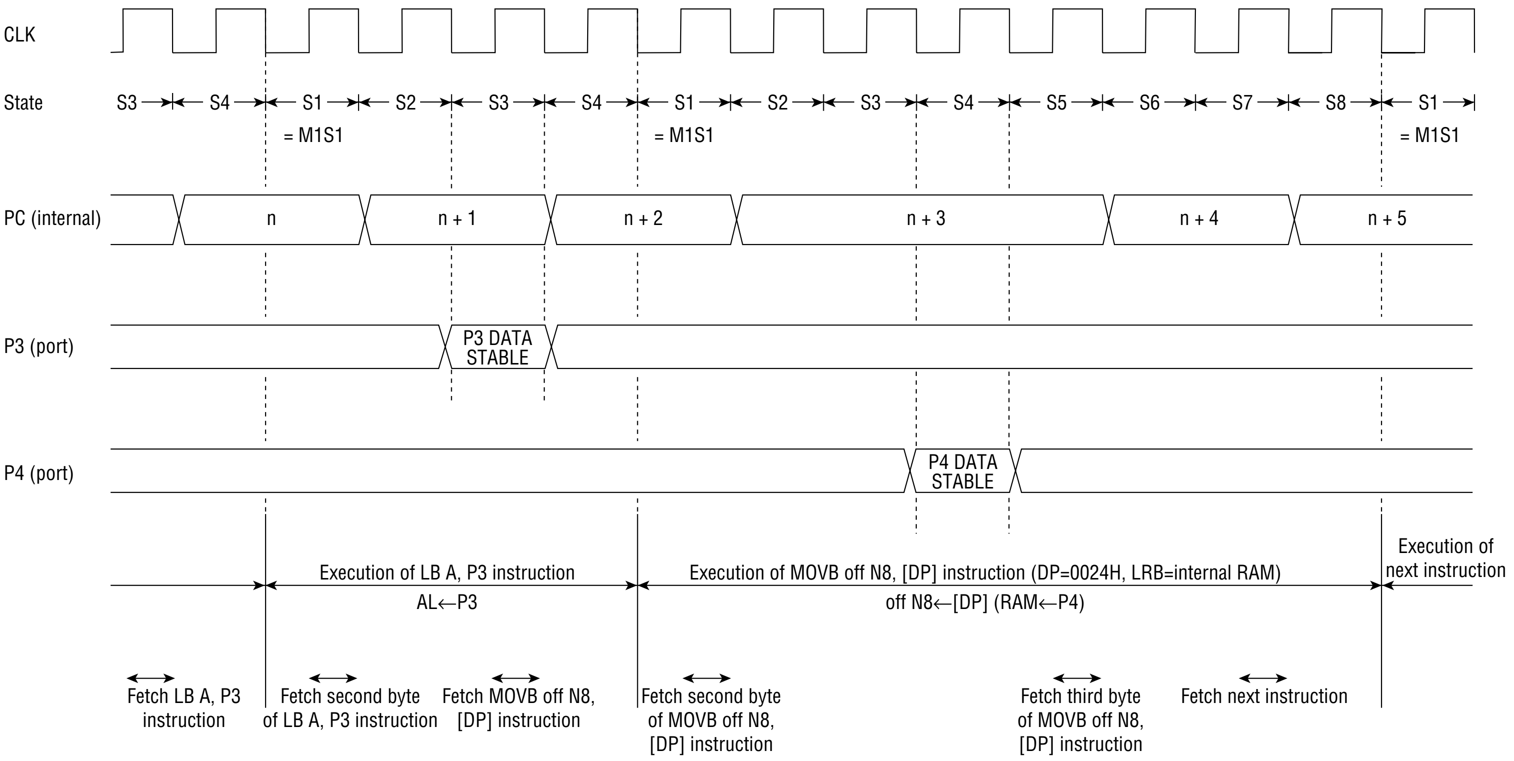

CLK
State
S3
S4
S1
S2
S3
S4
S1
S2
S3
S4
S5
S6
S7
S8
S1
= M1S1
= M1S1
= M1S1
PC (internal)
n
n + 1
n + 2
n + 3
n + 4
n + 5
P3 (port)
P3 DATA STABLE
P4 (port)
P4 DATA STABLE
Execution of LB A, P3 instruction
AL←P3
Execution of MOVB off N8, [DP] instruction (DP=0024H, LRB=internal RAM)
off N8←[DP] (RAM←P4)
Execution of next instruction
Fetch LB A, P3 instruction
Fetch second byte of LB A, P3 instruction
Fetch MOVB off N8, [DP] instruction
Fetch second byte of MOVB off N8, [DP] instruction
Fetch third byte of MOVB off N8, [DP] instruction
Fetch next instruction

Figure 1-3 Basic Operation Timing Example (Port Data Input)

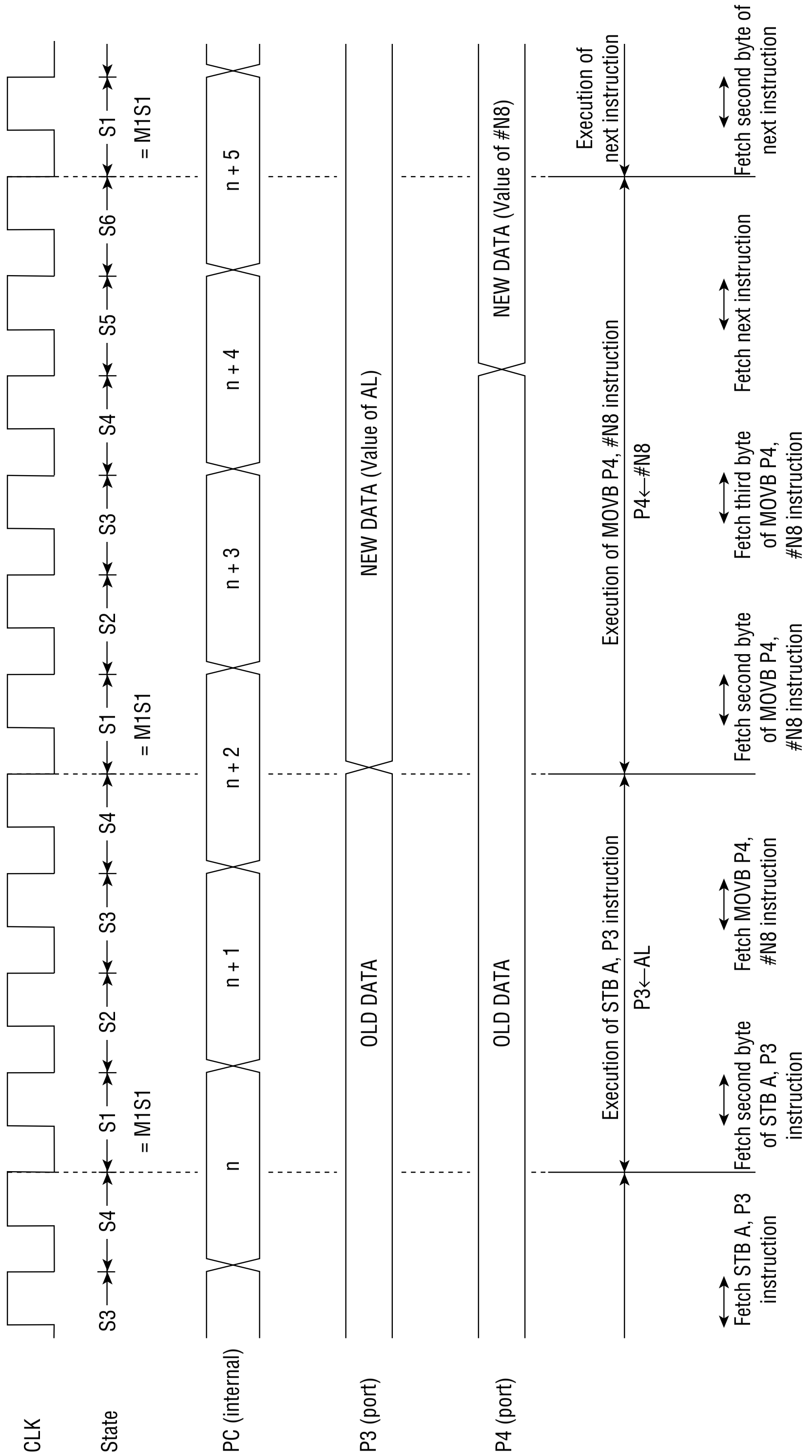

CLK
State
S3
S4
S1
S2
S3
S4
S1
S2
S3
S4
S5
S6
S1
= M1S1
= M1S1
= M1S1
PC (internal)
n
n + 1
n + 2
n + 3
n + 4
n + 5
P3 (port)
OLD DATA
NEW DATA (Value of AL)
P4 (port)
OLD DATA
NEW DATA (Value of #N8)
Execution of STB A, P3 instruction
P3←AL
Execution of MOVB P4, #N8 instruction
P4←#N8
Execution of next instruction
Fetch STB A, P3 instruction
Fetch second byte of STB A, P3 instruction
Fetch MOVB P4, #N8 instruction
Fetch second byte of MOVB P4, #N8 instruction
Fetch third byte of MOVB P4, #N8 instruction
Fetch next instruction
Fetch second byte of next instruction

Figure 1-4 Basic Operation Timing Example (Port Data Output)

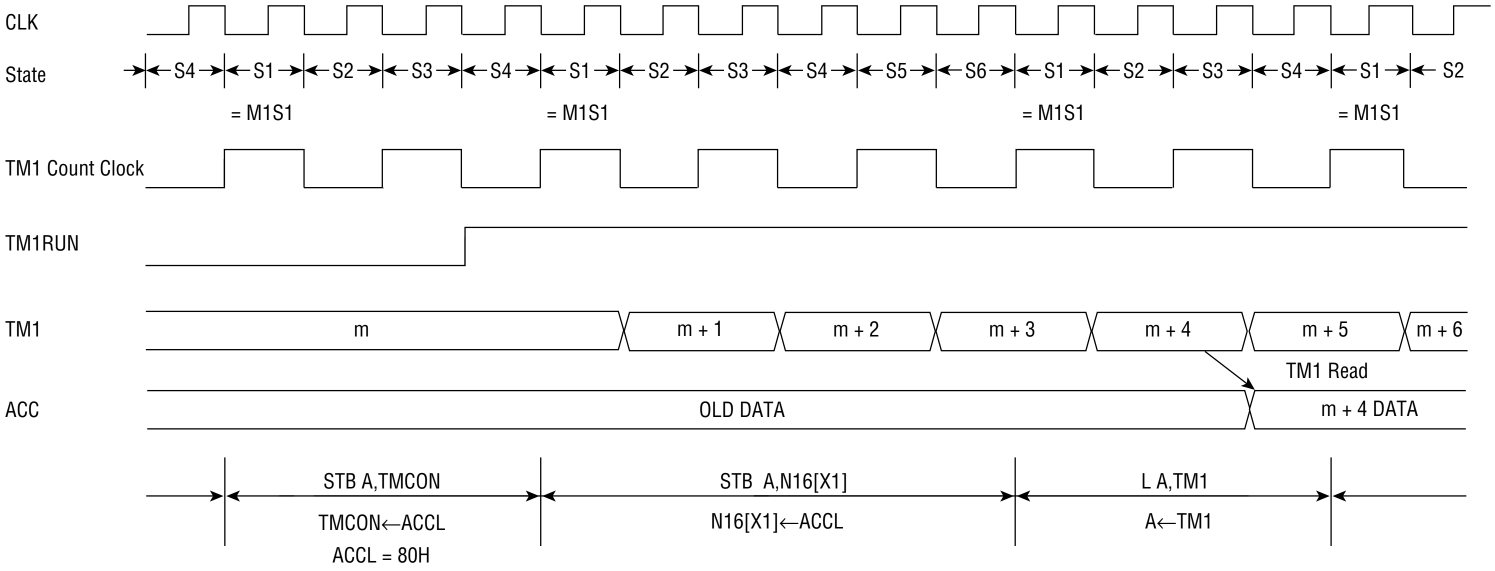


Figure 1-5 TM1 Operation Timing Example

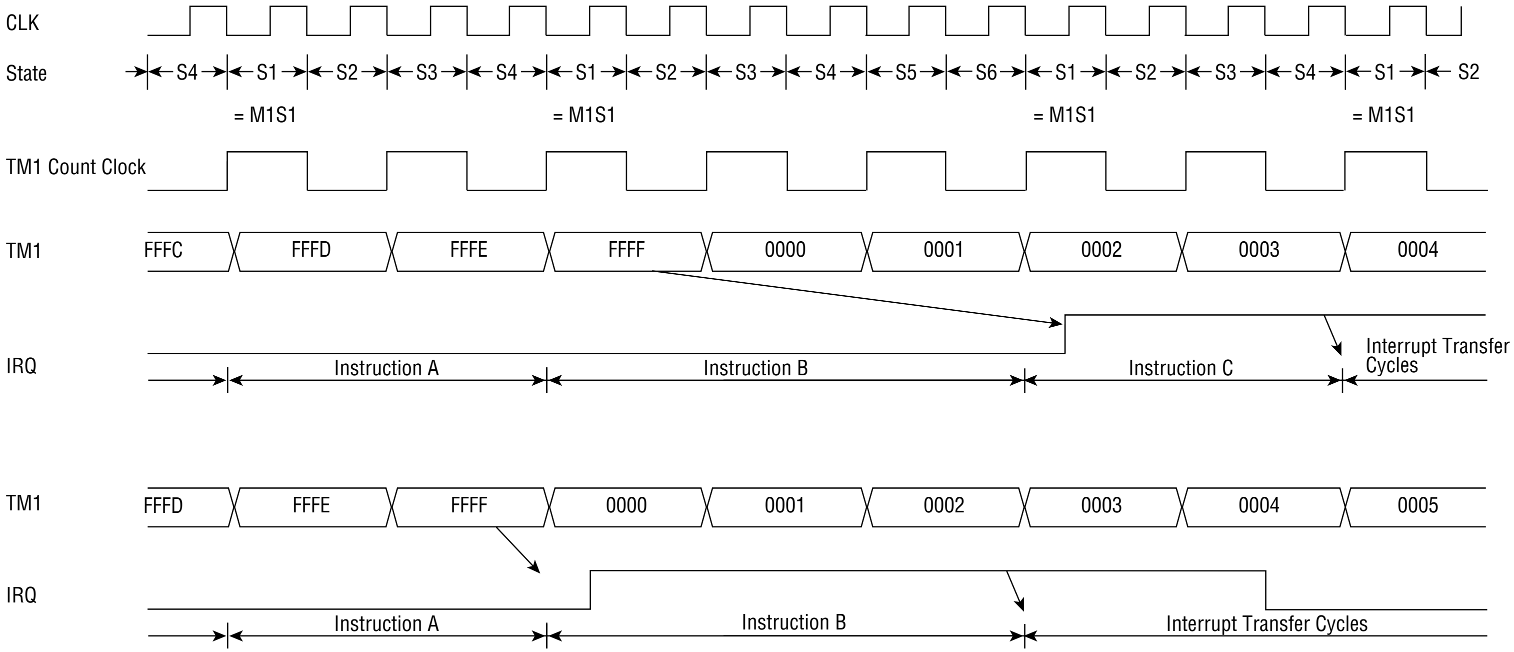


Figure 1-6 Interrupt Transfer Cycle Timing Example

Chapter 2

Pin Descriptions

2. Pin Descriptions

This chapter describes each pin.

2.1 P0_0-P0_7: Input/Output Pins

Port 0 is 8 input/output pins. Input or output can be specified for each bit with the Port 0 Mode Register (P0IO). Pull-up resistors can be specified for each bit with the Port 0 Pull-Up Register (P0PUP).

These pins also function as time-multiplexed address outputs and data input/output pins (AD0-AD7) when accessing memory that has been expanded externally (program or data memory).

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P0 will be high-impedance inputs.

2.2 P1_0-P1_7: Input/Output Pins

Port 1 is 8 input/output pins. Input or output can be specified for each bit with the Port 1 Mode Register (P1IO). Pull-up resistors can be specified for each bit with the Port 1 Pull-Up Register (P1PUP).

P1_0-P1_7 also have a secondary function as input/output pins for internal operation. Their secondary function can be set for each bit with the Port 1 Secondary Function Control Register (P1SF). The input/output settings by P1IO will be ignored for pins that have been set to the secondary function by P1SF.

< Secondary function of each pin >

A8-A15 (P1_0-P1_7)

These pins function as output pins for address A8-A15 when accessing program memory or data memory that has been expanded externally. When the $\overline{\text{EA}}$ pin is low, A8-A15 will be output regardless of P1SF settings.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P1 will be high-impedance inputs.

2.3 P2_0-P2_7: Input/Output Pins

Port 2 is 8 input/output pins. Input or output can be specified for each bit with the Port 2 Mode Register (P2IO).

P2_4 and P2_5 also have a secondary function as input/output pins for internal operation. Their secondary function can be set for each bit with the Port 2 Secondary Function Control Register (P2SF). The input/output settings of P2IO will be ignored for pins that have been set to the secondary function by P2SF.

< Secondary function of each pin >

RTO8 (P2_4), RTO9 (P2_5)

These pins output a previously set level when the value of Timer Registers 8 and 9 matches a selected counter value.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P2 will be high-impedance inputs.

2.4 P4_0-P4_7: Input/Output Pins

Port 4 is 8 input/output pins. Input or output can be specified for each bit with the Port 4 Mode Register (P4IO). Pull-up resistors can be specified for each bit with the Port 4 Pull-Up Register (P4PUP).

P4_0 also has a secondary function as an input pin for internal operation. Its secondary function can be set for the bit with the Port 4 Secondary Function Control Register (P4SF). The input/output settings by P4IO will be ignored for pins that have been set to the secondary function by P4SF.

< Secondary function of each pin >

ETMCK (P4_0)

This is the external clock input pin for the counter of a general 8-bit timer.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P4 will be high-impedance inputs.

2.5 P5_4, P5_5: Input/Output Pins

Port 5 is 2 input/output pins. Input or output can be specified for each bit with the Port 5 Mode Register (P5IO).

P5_4 and P5_5 also have a secondary function as output pins for internal operation. Their secondary function can be set for each bit with the Port 5 Secondary Function Control Register (P5SF). The input/output settings of P5IO will be ignored for pins that have been set to the secondary function by P5SF.

< Secondary function of each pin >

$\overline{\mathrm{PSEN}}$ (P5_4)

This pin outputs the strobe signal for read operations when external program memory is accessed. Operation will automatically switch to the secondary function when the $\overline{\mathrm{EA}}$ pin is low. This pin will be pulled up when both the $\overline{\mathrm{EA}}$ pin and $\overline{\mathrm{RES}}$ pin are low.

ALE (P5_5)

This pin outputs the strobe for externally latching the lower 8 address bits output from P0 when external memory is accessed. Operation will automatically switch to the secondary function when the $\overline{\mathrm{EA}}$ pin is low. This pin will be pulled up when both the $\overline{\mathrm{EA}}$ pin and $\overline{\mathrm{RES}}$ pin are low.

After reset (by $\overline{\mathrm{RES}}$ signal input, BRK instruction execution, or op code trap), P5 will be high-impedance inputs.

2.6 P6_0-P6_5

Port 6 is 6 input/output pins. Input or output can be specified for each bit with the Port 6 Mode Register (P6IO).

P6_0-P6_5 also have a secondary function as input/output pins for internal operation. Their secondary function can be set for each bit with the Port 6 Secondary Function Control Register (P6SF). The input/output settings of P6IO will be ignored for pins that have been set to the secondary function by P6SF.

< Secondary function of each pin >

INT0 (P6_0), INT1 (P6_1)

These pins input external interrupts 0 and 1.

RXD1 (P6_2)

This pin inputs receive data to the Serial Port 1 receive circuit.

TXD1 (P6_3)

This pin outputs transmit data to the Serial Port 1 transmit circuit.

RXC1 (P6_4)

This pin outputs the shift clock when the Serial Port 1 receive circuit is in master mode. It inputs the shift clock when the receive circuit is in slave mode.

TXC1 (P6_5)

This pin outputs the shift clock when the Serial Port 1 transmit circuit is in master mode. It inputs the shift clock when the transmit circuit is in slave mode.

After reset (by $\overline{\mathrm{RES}}$ signal input, BRK instruction execution, or op code trap), P6 will be high-impedance inputs.

2.7 P7_0-P7_7: Input/Output Pins

Port 7 is 8 input/output pins. Input or output can be specified for each bit with the Port 7 Mode Register (P7IO).

P7_0-P7_4 also have a secondary function as input/output pins for internal operation. Their secondary function can be set for each bit with the Port 7 Secondary Function Control Register (P7SF). The input/output settings of P7IO will be ignored for pins that have been set to the secondary function by P7SF.

< Secondary function of each pin >

$\overline{\text{WR}}$ (P7_0)

This pin outputs the strobe signal for write operations when external data memory is accessed.

$\overline{\text{RD}}$ (P7_1)

This pin outputs the strobe signal for read operations when external data memory is accessed.

WAIT (P7_2)

This pin inputs a wait to the internal CPU when external data memory with a slow access time is accessed. CPU is driven to "WAIT" state during WAIT pin high.

CLKOUT (P7_3)

This pin outputs the clock pulses set by the Peripheral Control Register (PRPHF).

PWM0 (P7_4)

This pin outputs PWM0.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P7 will be high-impedance inputs.

When P7_0 and P7_1 are used as their secondary functions ($\overline{\text{WR}}$, $\overline{\text{RD}}$), they need to be connected externally to pull-up resistors.

2.8 P8_0-P8_7: Input/Output Pins

Port 8 is 8 input/output pins. Input or output can be specified for each bit with the Port 8 Mode Register (P8IO).

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P8 will be high-impedance inputs.

2.9 P9_0-P9_7: Input/Output Pins

Port 9 is 8 input/output pins. Input or output can be specified for each bit with the Port 9 Mode Register (P9IO). Pull-up resistors can be specified for each bit with the Port 9 Pull-Up Register (P9PUP).

P9_0-P9_3 also have a secondary function as output pins for internal operation. Their secondary function can be set for each bit with the Port 9 Secondary Function Control Register (P9SF). The input/output settings of P9IO will be ignored for pins that have been set to the secondary function by P9SF.

< Secondary function of each pin >

A16-A19 (P9_0-P9_3)

These pins function as output pins for address A16-A19 when accessing program memory or data memory that has been expanded externally. Note that program memory address A16-A19 will be output even when accessing data memory that has been expanded externally. When the $\overline{\text{EA}}$ pin is low and program memory that has been expanded externally is accessed, A16-A19 will be output regardless of P9SF settings.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P9 will be high-impedance inputs.

2.10 P10_0-P10_7: Input/Output Pins

Port 10 is 8 input/output pins. Input or output can be specified for each bit with the Port 10 Mode Register (P10IO). Pull-up resistors can be specified for each bit with the Port 10 Pull-Up Register (P10PUP).

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P10 will be high-impedance inputs.

2.11 P12_0-P12_7: Input/Output Pins

Port 12 is 8 input/output pins. Input or output can be specified for each bit with the Port 12 Mode Register (P12IO).

P12_2 and P12_3 also have a secondary function as input pins for internal operation. Their secondary function can be set for each bit with the Port 12 Secondary Function Control Register (P12SF). The input/output settings of P12IO will be ignored for pins that have been set to the secondary function by P12SF.

< Secondary function of each pin >

INT2 (P12_2), INT3 (P12_3)

These pins input external interrupts 2 and 3.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P12 will be high-impedance inputs.

2.12 AI0-AI3: Input Pins

These are analog input pins for the A/D converter.

2.13 V_{REF}: Input Pin

This is the reference voltage pin for the A/D converter.

2.14 AGND: Input Pin

This is the ground input pin for the A/D converter.

2.15 OSC0, $\overline{\text{OSC1}}$: Input Pin, Output Pin

These pins connect to a crystal oscillator, ceramic oscillator, or capacitors for base clock oscillation. When the base clock is to be supplied externally, it should be input on the OSC0 pin with the $\overline{\text{OSC1}}$ pin left open.

2.16 NMI: Input Pin

This input pin requests a non-maskable interrupt.

2.17 $\overline{\text{RES}}$: Input Pin

This is an active-low reset input pin.

2.18 $\overline{\text{EA}}$: Input Pin

When this pin is high, program addresses 0H-FFFFH will access internal program memory and program addresses 10000H-FFFFFH will access external program memory. To access external program memory, P1, P5, and P9 must be set with their secondary function control registers.

When this pin is low, all program addresses will access external program memory.

2.19 V_{DD}: Input Pins

These are voltage pins. All V_{DD} pins (9, 17, 37, 67, 93) should be connected to the supply voltage.

2.20 GND: Input Pins

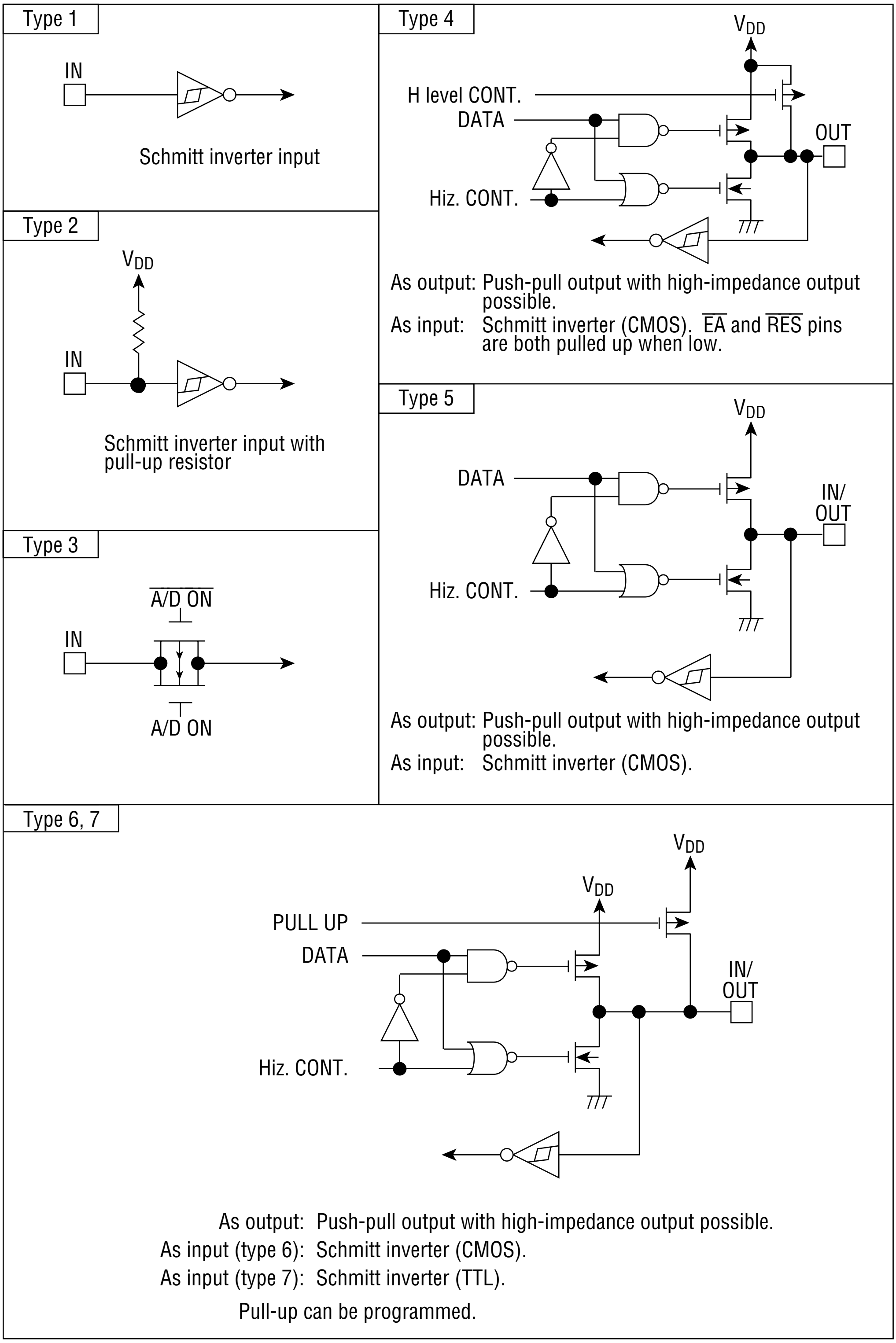
These are ground pins. All GND pins (16, 40, 68, 94) should be connected to ground.

2.21 Pin Configurations

Table 2-1 and Figure 2-1 show a simplified configuration for each pin.

Table 2-1 Pin Configurations

Pin Name	Type Number	Pin Name	Type Number
P0_0-P0_7	7	P9_0-P9_7	6
P1_0-P1_7	6	P10_0-P10_7	6
P2_0-P2_7	5	P12_0-P12_7	5
P4_0-P4_7	6	AI0-AI3	3
P5_4, P5_5	4	NMI	1
P6_0-P6_5	5	$\overline{\text{RES}}$	2
P7_0-P7_7	5	$\overline{\text{EA}}$	1
P8_0-P8_7	5		



Type 1

Schmitt inverter input

Type 2

Schmitt inverter input with pull-up resistor

Type 3

Type 4

As output: Push-pull output with high-impedance output possible.
As input: Schmitt inverter (CMOS). $\overline{EA}$ and $\overline{RES}$ pins are both pulled up when low.

Type 5

As output: Push-pull output with high-impedance output possible.
As input: Schmitt inverter (CMOS).

Type 6, 7

As output: Push-pull output with high-impedance output possible.
As input (type 6): Schmitt inverter (CMOS).
As input (type 7): Schmitt inverter (TTL).

Pull-up can be programmed.

Figure 2-1 Pin Configuration Types

2.22 Unused Pins

Table 2-2 shows how to connect unused pins.

Table 2-2 Unused Pins

Pin	Connection when unused
P0_0-P0_7	When set as programmable pull-up: open When set as input: high or low level When set as output: open
P1_0-P1_7	
P4_0-P4_7	
P9_0-P9_7	
P10_0-P10_7	
P2_0-P2_7	When set as input: high or low level When set as output: open
P5_4, P5_5	
P6_0-P6_5	
P7_0-P7_7	
P8_0-P8_7	
P12_0-P12_7	
AI0-AI3	V_{REF} or AGND
V_{REF}	V_{DD}
AGND	GND
NMI	High or Low level
$\overline{EA}$	High level

Chapter 3

CPU Architecture

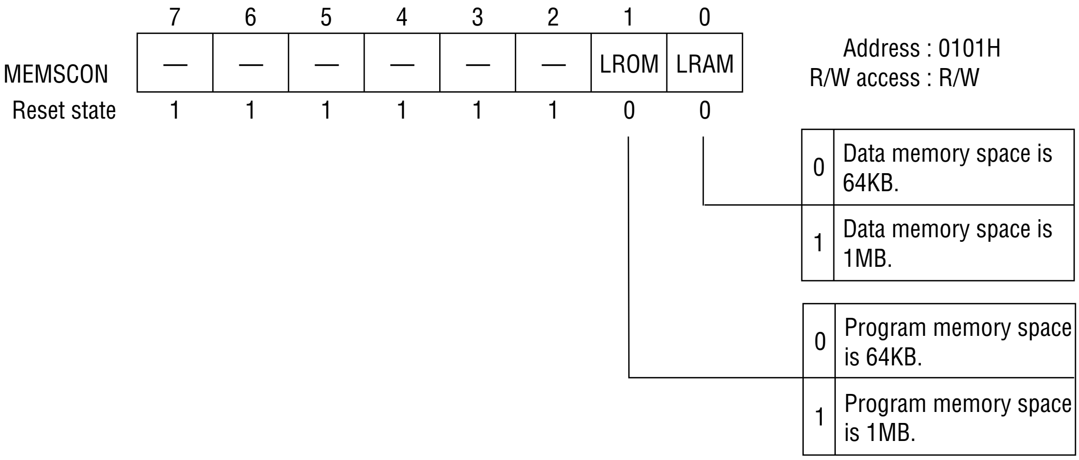
3. CPU Architecture

3.1 Memory Space

Two independent memory spaces, a program memory space and data memory space are provided. After reset both program memory space and data memory space can be accessed up to 64KB. By changing the Memory Size Control Register (allocated as an SFR), program memory space and data memory space can be expanded to a maximum of 1MB.

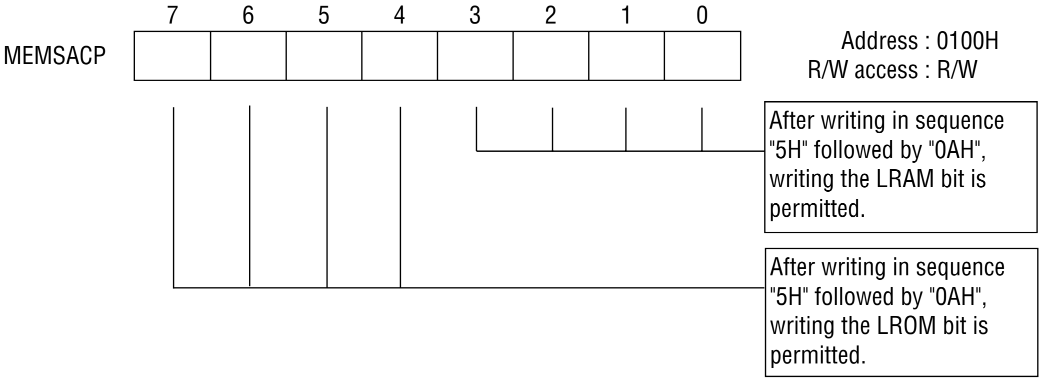
3.1.1 Memory Space Expansion

The Memory Size Control Register (MEMSCON) is allocated in the expanded SFR area to specify the size of the memory spaces. By setting bit 1 (LROM) to "1", program memory space will be expanded to 1MB. By setting bit 0 (LRAM) to "1", data memory space will be expanded to 1MB.


7
6
5
4
3
2
1
0
MEMSCON
—
—
—
—
—
—
LROM
LRAM
Reset state
1
1
1
1
1
1
0
0
Address : 0101H
R/W access : R/W
0 Data memory space is 64KB.
1 Data memory space is 1MB.
0 Program memory space is 64KB.
1 Program memory space is 1MB.

"—" indecates bits that do not exist.
When read, "1" is read.

Just before writing the LROM and LRAM bits of MEMSCON, you must write in sequence 5H followed by 0AH to the upper (for LROM) and lower (for LRAM) four bits of the Memory Size Acceptor Register (MEMSACP).


7
6
5
4
3
2
1
0
MEMSACP
Address : 0100H
R/W access : R/W
After writing in sequence "5H" followed by "0AH", writing the LRAM bit is permitted.
After writing in sequence "5H" followed by "0AH", writing the LROM bit is permitted.

When the LROM and LRAM bits are written (set), the actual change in memory space will be after execution of the instruction after the instruction that wrote the LROM and LRAM bits. Here is a program example showing memory space expansion to each individual model.

- COMPACT model (pragram memory space 64KB, data memory space 1MB)

```
MOVB    MEMSACP, #05H
MOVB    MEMSACP, #0AH
MOVB    MEMSCON, #01H
```

- MEDIUM model (program memory space 1MB, data memory space 64KB)

```
MOVB    MEMSACP, #50H
MOVB    MEMSACP, #0A0H
MOVB    MEMSCON, #02H
```

- LARGE model (program memory space 1MB, data memory space 1MB)

```
MOVB    MEMSACP, #55H
MOVB    MEMSACP, #0AAH
MOVB    MEMSCON, #03H
```

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, or OPTRP trap).
MEMSCON can be written one time only. Therefore, once the memory space model has been set, it can be changed to other models only after a reset ($\overline{\text{RES}}$ signal input, BRK instruction, or OPTRP trap).

3.1.2 Program Memory Space

Up to 1MB (1,048,576 bytes) of program memory can be accessed as segments 0-15, with one segment equal to 64KB (65,536 bytes). However, to access above 64KB (segments 1-15), the LROM bit of MEMSCON (Memory Size Control Register) allocated as an SFR must be set to 1.

The segment is specified by the Code Segment Register (CSR). The address within a segment is specified by the Program Counter (PC). However, the segment when executing ROM table reference instructions (LCA obj, etc.) and ROM window functions is specified by the Table Segment Register (TSR).

The 64KB in segment 0 are the internal ROM area, and the 64KB in segments 1-15 are the external ROM area.

The following are allocated in segment 0.

Vector table area (82 bytes)
VCAL table area (32 bytes)

The following are allocated in each segment.

ACAL area (2048 bytes)

Figure 3-1 shows a memory map of program memory space.

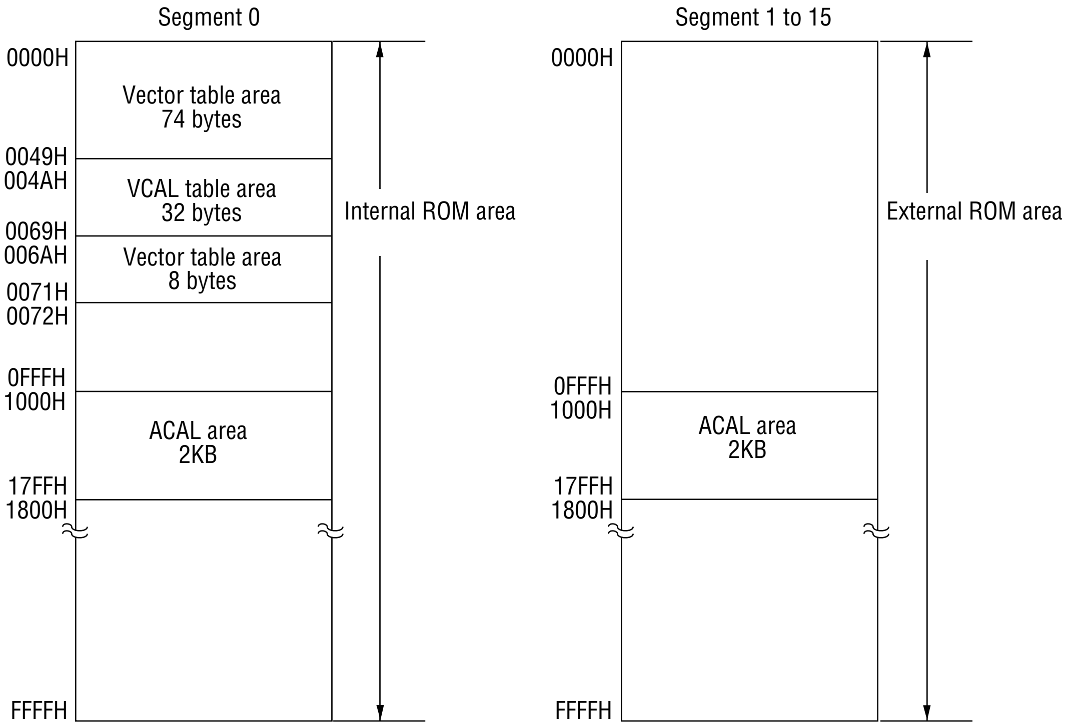


Figure 3-1 Program Memory Space Memory Map

(1) Accessing Program Memory Space

Program memory space is accessed using the Program Counter (PC) and Code Segment Register (CSR). However, when ROM table reference instructions (LC A, obj, etc.) and ROM window functions (refer to Section 5.1) are executed, program memory space will be accessed using the Table Segment Register (TSR) and a register specified by the instruction.

For program memory space, access of the internal ROM area and external memory area is switched automatically by the state of the $\overline{\text{EA}}$ pin (input: pin 36). When the $\overline{\text{EA}}$ pin is input high, program memory addresses 0000H-FFFFH will access the internal memory area, and 10000H-FFFFFH will access the external memory area. To access the external memory area, Port 1, 5, and 9 must be set by their secondary function control registers.

When the $\overline{\text{EA}}$ pin is input low, program addresses across the entire memory space will be accessed in the external memory area.

When program memory space is accessed in the external memory area, Port 0 (I/O pins 51-58: AD0-AD7 outputs and data inputs), Port 1 (output pins 59-66: A8-A15 outputs), and Port 9 (output pins 69-72: A16-A19 outputs) operate as a bus port synchronized to the P5_5/ALE pin (output pin 50). The P5_4/$\overline{\text{PSEN}}$ pin (output pin 49) will also be active.

The internal program area that can be fetched is 0000H-FFFDH. Be sure that the last address of instruction codes does not exceed FFFDH. The last address of table data is FFFFH.

(2) Vector Table Area

In program memory space segment 0, the 74 bytes at 0000H-0049H and the 8 bytes at 006AH-0071H form a vector table area (16 types) that stores branch addresses for each type of interrupt request and for reset by $\overline{\text{RES}}$ pin input (input pin 34), by break instruction, or by op code trap (OPTRP).

When a reset or interrupt occurs, the corresponding 2-byte branch address stored in the vector table is loaded into the PC (even address is low data, odd address is high data), and simultaneously 0 is loaded into the CSR. Program execution then starts from the loaded address in segment 0. If, for example, a reset or interrupt occurs during execution of an instruction in segment 1, then execution will branch to an address in segment 0.

If not used as a vector table area, this area can be used as a normal program area.

Table 3-1 shows the vector table for reset and interrupt requests.

[Example] Program start address after $\overline{\text{RES}}$ pin input from 0200H

Program Address	Data Code	
0000H	00H	Program start address low data
0001H	02H	Program start address high data

Table 3-1 Vector Table

Program Address Address [H]	Vector Source
0000	$\overline{\text{RES}}$ pin input reset
0002	BRK instruction execution reset
0006	OPTRP (op code trap) reset
0008	NMI pin input interrupt
000A	External interrupt pin INT0 interrupt
000E	TM1 overflow interrupt
0020	RT08 event generation interrupt
0022	RT09 event generation interrupt
0028	Serial port 1 transmit/receive interrupt
002A	S1TM overflow interrupt
002C	GTMC overflow interrupt
0030	A/D conversion completion interrupt
0032	PWC0 underflow/matching interrupt
0038	External interrupt pin INT1 interrupt
006E	External interrupt pin INT2 interrupt
0070	External interrupt pin INT3 interrupt

(3) VCAL Table Area

In program memory space segment 0, the 32 bytes at 004AH-0069H form the VCAL table area that stores branch addresses (VCAL: 16 types) for one-byte call instructions.

When a VCAL instruction is executed, the next address after the VCAL instruction is saved on the system stack, the System Stack Pointer (SSP) is decremented by two, the corresponding 2-byte branch address stored in the VCAL table is loaded into the PC (even address is low data, odd address is high data), and program execution then starts from the loaded address.

However, when program memory space has been expanded to 1MB, the value of CSR is saved at the same time the value of PC is saved, so SSP will be decremented by four. Also, at the same time the branch address is loaded into the PC, a 0 will be loaded into CSR. If, for example, a VCAL instruction in segment 1 is executed, then execution will branch to an address in segment 0.

When program memory space is 64KB with the LROM bit in MEMSCON set to 0, return from subroutines called by a VCAL instruction with an RT instruction. When program memory space is 1 MB with the LROM bit in MEMSCON set to 1, return from subroutines called by a VCAL instruction with an FRT instruction.

If not used as a VCAL table area, this area can be used as a normal program area.

Table 3-2 shows VCAL vector addresses.

[Example] Program start address after VCAL 4AH from 0400H

Program Address	Data Code	
004AH	00H	Program start address low data
004BH	04H	Program start address high data

Table 3-2 VCAL Vector Table Addresses

VCAL Table Address [H]	VCAL Instruction
004A	VCAL 4AH
004C	VCAL 4CH
004E	VCAL 4EH
0050	VCAL 50H
0052	VCAL 52H
0054	VCAL 54H
0056	VCAL 56H
0058	VCAL 58H
005A	VCAL 5AH
005C	VCAL 5CH
005E	VCAL 5EH
0060	VCAL 60H
0062	VCAL 62H
0064	VCAL 64H
0066	VCAL 66H
0068	VCAL 68H

(4) ACAL Area

In each segment of program memory space, the 2KB at 1000H-17FFH is the area for direct subroutine calls with the 2-byte call instruction (ACAL).

When an ACAL instruction is executed, the next address after the ACAL instruction is saved on the system stack, the System Stack Pointer (SSP) is decremented by two, the 11-bit data including in the ACAL instruction code is loaded into the PC, and program execution then starts from the loaded address. The CSR value is not modified.

3.1.3 Data Memory Space

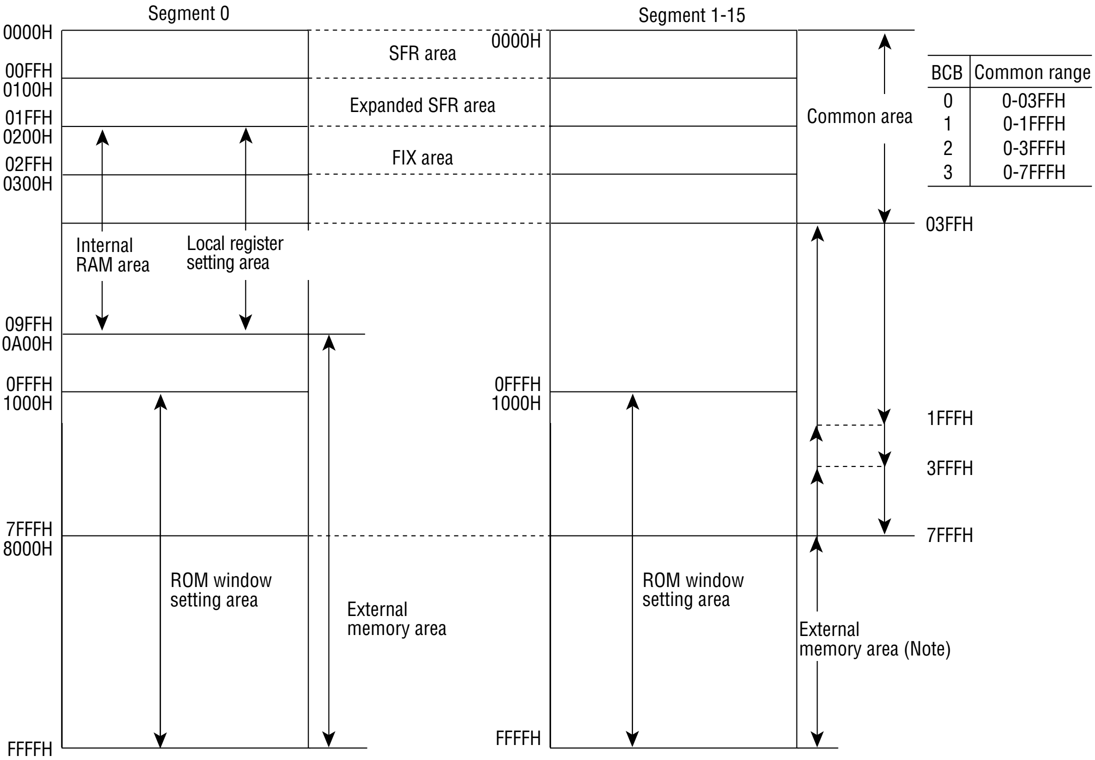
Up to 1MB (1,048,576 bytes) of data memory can be accessed.

The following areas are allocated to data memory space: special function register area (SFR: 256 bytes), expanded SFR area (256 bytes), fixed page area (FIX: 256 bytes), internal RAM area (2,048 bytes), local register setting area (2,048 bytes), and external memory area (1,046,016 bytes).

In addition, the pointing register area (PR: 64 bytes) and special bit addressing area (sbafix: 64 bytes) are allocated to the fixed page area. The ROM window setting area (1000H-FFFFH in every segment) is allocated to the external memory area.

To enable exchange of data between multiple data segments, there is a common area that starts in data memory from address 0H. The SFR area, expanded SFR area, and fixed page area always reside in the common area.

Figure 3-2 shows the memory map of data memory space.



Note: The range of the external memory area in segments 1 to 15 depends on that of the common area to be set.

Figure 3-2 Data Memory Space Memory Map

(1) Special Function Register (SFR) Area

In data memory space, the 256 bytes at 0000H-00FFH are allocated to a register group with special functions, like counters, control registers, and mode registers for internal peripheral hardware.

(2) Expanded Special Function Register (Expanded SFR) Area

In data memory space, the 256 bytes at 0100H-01FFH are allocated to a register group with special functions, just like the SFR area. However, note that the expanded SFR area is restricted in that it cannot use SFR addressing.

(3) Internal RAM Area

In data memory space, the 2KB (2,048 bytes) at 0200H-09FFH are allocated to internal RAM.

(4) Fixed Page (FIX) Area

In data memory space, the 256 bytes at 0200H-02FFH are allocated as a pointing register (PR) area and special bit address (sbafix) area.

The pointing register area is allocated to 0200H-023FH. It provides eight sets of the following four registers.

- Index Registers (X1, X2)
- Data Pointer (DP)
- User Stack Pointer (USP)

All of the pointing registers are 16-bit registers, with the even address as low data and the subsequent odd address as high data.

The special bit address area is allocated to 02C0H-02FF. This area enables efficient byte counts with SB, RB, JBR, and JBS instructions.

Figure 3-3 shows a map of the fixed page area.

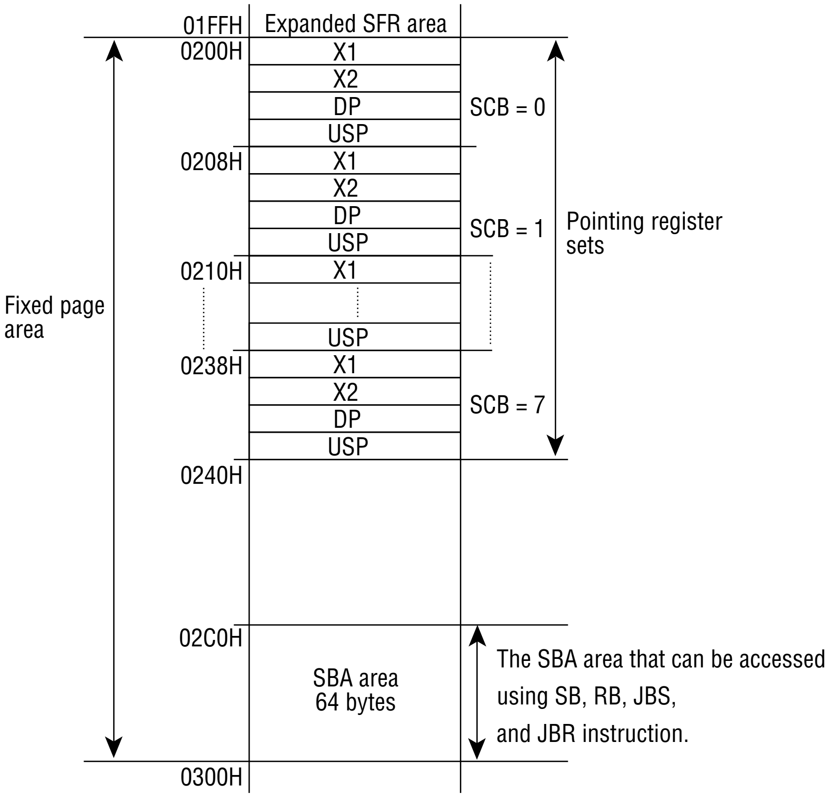


Figure 3-3 Fixed Page Area Map

(5) Area for Setting Local Registers

In data memory space, the 2KB at 0200H-09FFH can be set for local registers. Local registers are set in 8-byte blocks with the lower 8 bits of LRB (LRBL). Figure 3-4 shows the map of the area for setting local registers.

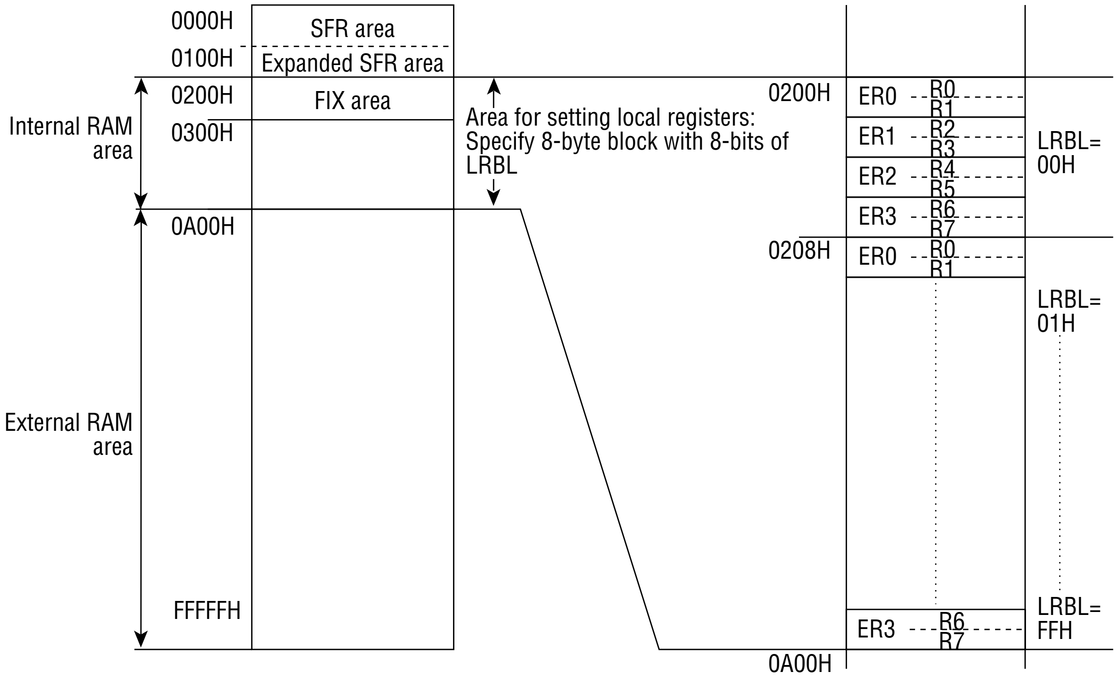


Figure 3-4 Map of Area for Setting Local Registers

(6) External Memory Area

In data memory space, the 1022KB (1,046,016 bytes) at 0A00H-FFFFFH are an external memory area. This area is accessed by the following signals; port 0 (address output AD0-AD7, data input /output), port 1 (address output A8-A15), port 9 (address output A16-A19), P5_5/ALE (port 5 P5_5 secondary function), P7_0/$\overline{\text{WR}}$ (write strobe output function), and P7_1/$\overline{\text{RD}}$ (read strobe output function). The ROM window function can also be set in this area with the ROM Window Setting Register. For a specified area in data memory space (1000H and above) the ROM window function makes instruction accesses (read operations) not on data in data memory space, but instead on the data in program memory space at the same address.

In the ROM window function, set the register for enabling it (ROMWIN) and access (read operations) addresses in external data memory.

(7) Common Area

Data memory space has a common area for exchanging data between segments. The area common to all segments is in the low end of data memory starting at offset address 0H of each segment. The range of the common area is determined by the values of the BCB bits in the PSW.

BCB		Common Area Range
1	0	
0	0	0H-03FFH
0	1	0H-1FFFH
1	0	0H-3FFFH
1	1	0H-7FFFH

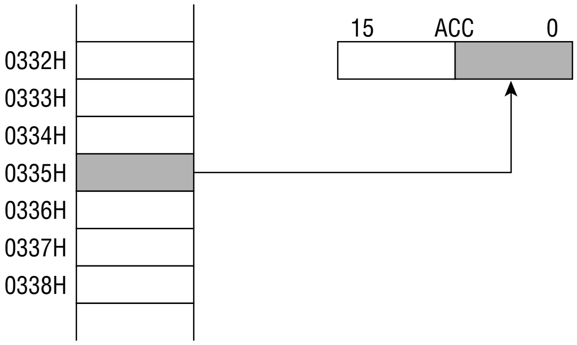
3.1.4 Accessing Data Memory

This section shows memory accesses in data memory space with byte and word operations performed by instructions.

(1) Byte Operation

For byte operations, 8-bit data at the address obtained by the instruction is accessed.

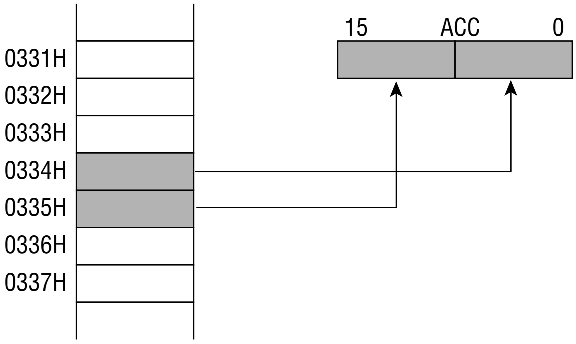
[Example] For LB A, [DP] assume that the value of DP is 0335H.



(2) Word Operation

For word operations, 8 bit-data at the address obtained by the instruction but with its least significant bit set to "0"(even address) is accessed as the low 8-bit data and 8bit-data at the obtained address but with its LSB set to "1" (odd address) is accessed as the high 8-bit data.

[Example] For L A, [DP] assume that the value of DP is 0334H (or 0335H).



3.2 Registers

Registers are classified by their function: accumulator registers, control registers, pointing registers, special function registers, local registers, and segment registers. Figure 3-5 shows the configuration of the registers.

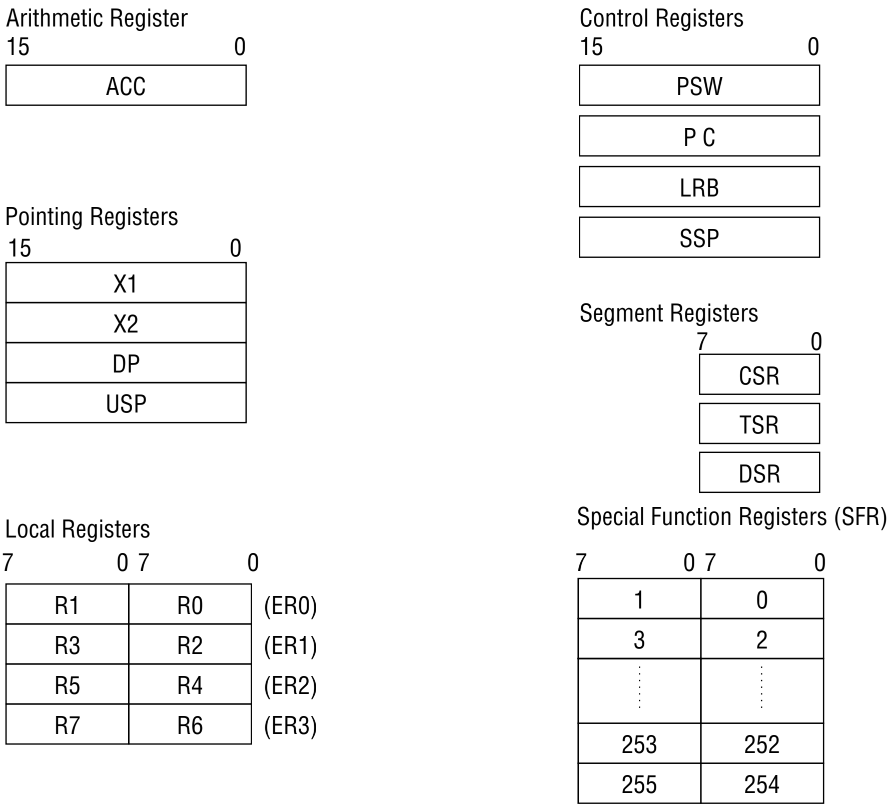


Figure 3-5 Register Configuration

3.2.1 Arithmetic Register (ACC)

The accumulator (ACC) is a 16-bit register for all types of calculations. The arithmetic operation, shifting, and so on are performed with as follows.

- Word type: all 16 bits (bits 15-0) are performed
- Byte type: lower 8 bits (bits 7-0) are performed
- Nibble type: lower 4 bits (bits 3-0) are performed

When the accumulator specifies the object bit of a bit operation instruction (SBR, RBR, etc.) the upper 5 bits of the lower byte (bits 7-3) specify the address offset and the lower 3 bits (bits 2-0) specify the bit position.

ACC is allocated as an SFR. Its value after reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) will be 0000H.

3.2.2 Control Registers

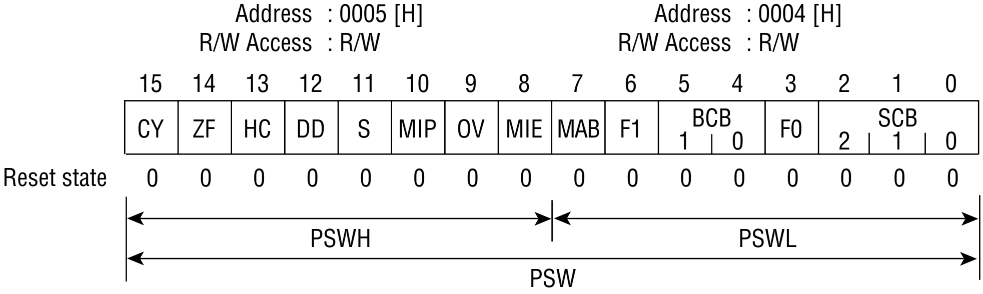
This is a group of registers with specific functions to control program status, program sequence, local registers, stacks, etc. There are four 16-bit control registers.

(1) Program Status Word (PSW)

The PSW is a 16-bit register configured as follows.

- Flag for instruction execution (DD)
- Flags set (1) or reset (0) by instruction execution results (CY, ZF, HC, S, OV)
- Flag to specify the pointing register set (SCB0-2)
- Flag to enable (1) or disable (0) all maskable interrupts (MIE)
- Flag to enable (1) or disable (0) priority of all maskable interrupts (MIP)
- Flag to set the segment 0 common area (BCB0, 1)
- Flags available for user use (F0, 1)
- Flag provided for future expansion of CPU core functions, but which is available for user use with the MSM66587 family (MAB)

PSW is divided into 8-bit registers, PSWH (bits 15-8) and PSWL (bits 7-0), so it can be operated on with 8-bit instruction operations in addition to 16-bit operations.



	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CY	ZF	HC	DD	S	MIP	OV	MIE	MAB	F1	BCB 1	BCB 0	F0	SCB 2	SCB 1	SCB 0
Reset state	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 3-6 PSW Configuration

The high byte of PSW (PSWH) contains the following.

- Flag for instruction execution (DD)
- Flags set (1) or reset (0) by instruction execution results (CY, ZF, HC, S, OV)

Please note that when the following instructions are executed, the actual operation of the PSW or PSWH flags may differ from expected.

① Instructions that load contents of PSW or PSWH into ACC.
(Value corresponding to ZF in ACC will be undefined.)

② Bit operation instructions on ZF.
(ZF will change based on its value immediately prior to execution of the bit operation instruction.)

③ Instructions that perform increment, decrement, arithmetic calculation, logical calculation, or comparison on PSW or PSWH
(Value of PSW or PSWH after instruction execution will be undefined.)

When an interrupt occurs, the PSW will be saved automatically during the interrupt process. It will be restored automatically by execution of an RTI instruction.

PSW is allocated as an SFR. Its value after reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) will be 0000H.

Below is an explanation of each bit in PSW.

Bit 15: Carry Flag (CY)

The Carry Flag will be set (1) if the result of an arithmetic calculation or comparison instruction generates any of the following. Otherwise it will be reset (0).

- Carry is generated from bit 7 for byte calculations.
- Borrow is generated to bit 7 for byte calculations.
- Carry is generated from bit 15 for word calculations.
- Borrow is generated to bit 15 for word calculations.

The Carry Flag can be directly set and reset by instructions and data can also be sent and received to the bits specified by the register. The state of the Carry Flag can also be tested by conditional branch instructions.

Bit 14: Zero Flag (ZF)

The Zero Flag will be set (1) when the following occurs. Otherwise it will be reset (0).

- Result value of an arithmetic calculation instruction is 0.
- Execution of a load instruction to ACC loads a value of 0.
- Object bit of a bit operation instruction is 0.

The state of the Zero Flag can also be tested by conditional branch instructions.

Bit 13: Half-Carry Flag (HC)

The Half-Carry Flag will be set ("1") when the result of executing an arithmetic calculation or comparison instruction (byte and word calculations are the same) carrys or borrows from bit 3. Otherwise it will be reset (0).

Bit 12: Data Descriptor (DD)

The Data Descriptor flag indicates the attribute of data stored in ACC.

- When DD is 1, all 16 bits of ACC data are considered valid.
- When DD is 0, the lower 8 bits of ACC data are considered valid.

When calculation and data transfer instructions with ACC are executed, DD is referred and execution proceeds as follows.

- When DD is 1, calculation and transfer will be executed with words.
- When DD is 0, calculation and transfer will be executed with bytes.

DD can be set ("1") or reset ("0") when a data transfer instruction to ACC is executed and when flag-specific set or reset instructions are executed.

- DD will be set ("1") when a word load instruction to ACC is executed or the SDD instruction is executed.
- DD will be reset ("0") when a byte load instruction to ACC is executed or the RDD instruction is executed.

When DD is modified by execution of a load instruction to ACC or a flag-specific set or reset instruction, the following instruction will refer and execute the modified DD if that instruction is reference DD.

DD is a part of PSW, so it can be overwritten by instructions other than the ones described above. In this case, the following instruction will refer the value of DD before it is modified if that instruction is reference DD. For this use, insert a NOP instruction after the instruction that directly modifies DD.

Bit 11: Sign Flag (S)

The Sign Flag will be set if the MSB of the execution result of an arithmetic or logical operation instruction is "1", and will be reset if MSB = "0".

Bit 10: Master Interrupt Priority Flag (MIP)

The Master Interrupt Priority Flag controls the priority function for maskable interrupts. If "1", the priority function is enabled. If "0", the priority function is disabled.

Bit 9: Overflow Flag (OV)

The Overflow Flag will be set ("1") if the execution result of an arithmetic operation instruction exceeds the range that can be expressed as 2's complement (–128 to +127 for byte arithmetic, –32768 to +32767 for word arithmetic). Otherwise it will be reset ("0").

Bit 8: Master Interrupt Enable Flag (MIE)

The Master Interrupt Enable Flag controls whether all maskable interrupts are enabled ("1") or disabled ("0").

This flag will be reset ("0") after the PSW is saved on the system stack during a maskable interrupt transfer cycle. It will be restored when an RTI instruction is executed. When MIE is set ("1"), all maskable interrupts are enabled to occur from the instruction following the one that set MIE. When MIE is reset ("0"), all maskable interrupts are disabled to occur from the instruction following the one that reset MIE.

Bit 7: Multiply-Accumulate Bank Flag (MAB)

This flag can be used as a user flag.

Bit 6: User Flag 1 (F1)
Bit 3: User Flag 0 (F0)

The User Flags can be set ("1") and reset ("0") by instructions.

Bit 5: Bank Common Base 1 (BCB1)
Bit 4: Bank Common Base 0 (BCB0)

The Bank Common Base specifies the last address of the common area between segments in data memory space. The relationship of BCB value and the common area is as follows.

BCB		Common Area Range
1	0	
0	0	0H-03FFH
0	1	0H-1FFFH
1	0	0H-3FFFH
1	1	0H-7FFFH

(2) Program Counter (PC)

The Program Counter is a 16-bit counter that stores the program address within a segment to execute next. It is generally incremented by the number of bytes in each instruction executed. When branch and conditional jump instructions are executed, the PC will be loaded with immediate data or the contents of a register. Even if the PC overflows when incremented, the value of CSR will not change.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap) or when an interrupt occurs, the PC will be loaded with a value from the vector table area.

(3) Local Register Base (LRB)

The Local Register Base is a 16-bit register. Its low-order 8 bits (LRBL) specify 2KB (0200H-09FFH) of data memory space (for local register addressing) in units of 8 bytes. Its high-order 8 bits (LRBH) specify 64KB of data memory space (for current page addressing) in units of 256 bytes. The 64 bytes at XXC0H-XXFFH in the current page are an area that can be accessed using the sba.bit of SB, RB, JBR, and JBS instructions.

LRBL (02H) and LRBH (03) are both allocated in the SFR area. After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), their contents are undefined.

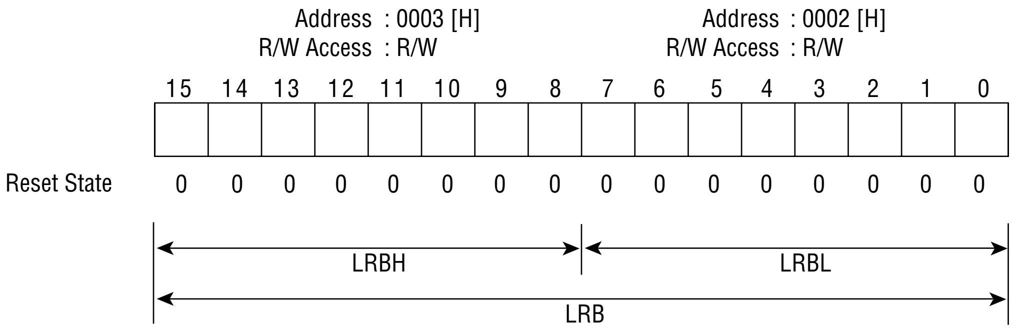
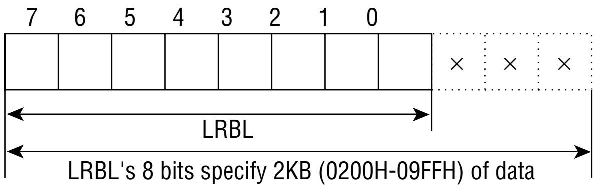


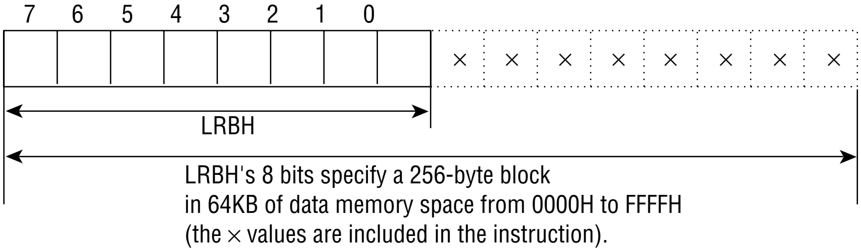
Figure 3-7 LRB Configuration

- LRBL’s 8 bits specify 2KB (0200H-09FFH) of data memory space (the X values are included in the instruction code) in units of 8 bytes.



LRBL's 8 bits specify 2KB (0200H-09FFH) of data memory space (the × values are included in the instruction code) in units of 8 bytes.

- LRBH’s 8 bits specify 64KB of data memory space in units of 256 bytes.


7 6 5 4 3 2 1 0
× × × × × × × ×
LRBH
LRBH's 8 bits specify a 256-byte block
in 64KB of data memory space from 0000H to FFFFH
(the × values are included in the instruction).

Bit 2: System Control Base 2 (SCB2)
Bit 1: System Control Base 1 (SCB1)
Bit 0: System Control Base 0 (SCB0)

The System Control Base specifies the Pointing Register (PR) set allocated in the fixed page area.

SCB			Pointing Register Set
2	**1**	**0**	
0	0	0	PR0 (0200H-0207H)
0	0	1	PR1 (0208H-020FH)
0	1	0	PR2 (0210H-0217H)
0	1	1	PR3 (0218H-021FH)
1	0	0	PR4 (0220H-0227H)
1	0	1	PR5 (0228H-022FH)
1	1	0	PR6 (0230H-0237H)
1	1	1	PR7 (0238H-023FH)

(4) System Stack Pointer (SSP)

The System Stack Pointer is a 16-bit register that points to the address of the stack for saving and restoring PC and register values when interrupts are processed and call, push, return, and pop instructions are executed. SSP is automatically incremented or decremented by instruction execution.

Saving and restoring to the stack address pointed to by SSP is performed with words, so the least significant bit (LSB) of SSP is "0".

The SFR area and expanded SFR area cannot be used as the stack.

The SSP (00H) is allocated in the SFR. After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), its value will be FFFFH.

3.2.3 Pointing Register (PR)

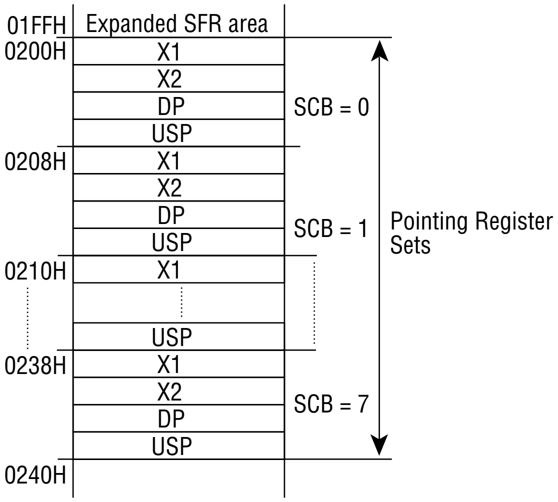
There are eight PR sets. Each set consists of four 16-bit registers.

- Index Register 1 (X1)
- Index Register 2 (X2)
- Data Pointer (DP)
- User Stack Pointer (USP)

PR sets are allocated to 0200H-023FH in the internal RAM area. One of the eight sets will be selected by the SCB0-2 bits in PSWL.

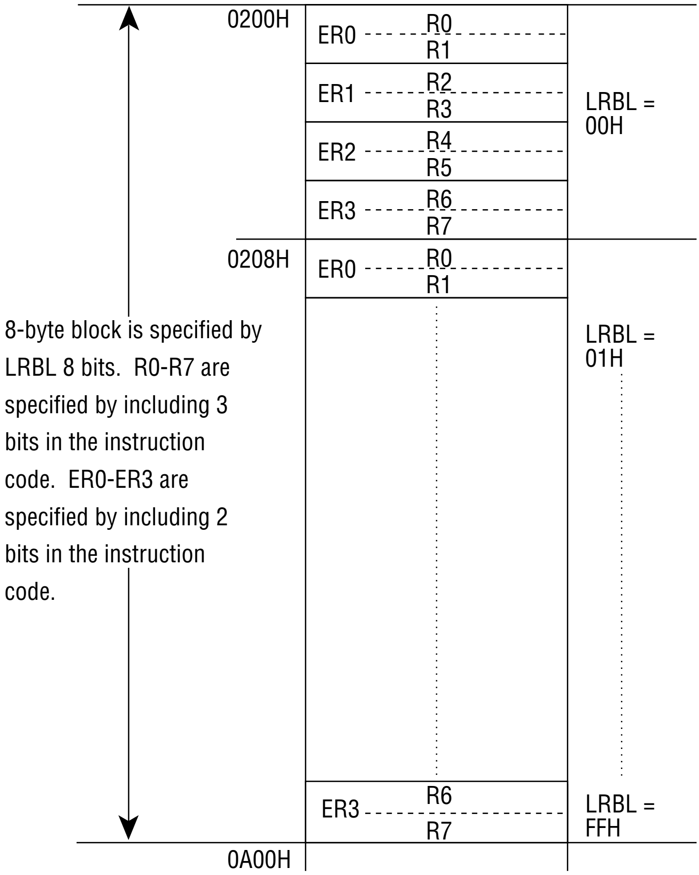
If the PR functions are not used, then area for PR sets can be used as normal internal RAM.

X1, X2, DP, and USP have their even address in the low-order 8 bits and odd address in the low-order 8 bits.



3.2.4 Local Registers (R, ER)

R is an 8-bit register, and ER is a 16-bit register. Local Registers R and ER specify 2KB of Data Memory Space from 0200H to 09FFH in 8-byte units by low 8-bit LRBL. The one of the 8 bytes can be accessed by specifying 3-bit data embedded in a local register operation instruction (for ER, two of the 8 bytes can be accessed by specifying 2-bit data embedded in a local register operation).


0200H
ER0
R0
R1
ER1
R2
R3
ER2
R4
R5
ER3
R6
R7
LRBL = 00H
0208H
ER0
R0
R1
LRBL = 01H
8-byte block is specified by LRBL 8 bits. R0-R7 are specified by including 3 bits in the instruction code. ER0-ER3 are specified by including 2 bits in the instruction code.
ER3
R6
R7
LRBL = FFH
0A00H

3.2.5 Segment Registers

The segments registers are three 8-bit registers: Code Segment Register (CSR), Table Segment Register (TSR), and Data Segment Register (DSR). They select segments in program memory space.

However, only segments 0-15 exist in program memory space, so only bits 3-0 are valid. Bits 7-4 are always 0.

(1) Code Segment Register (CSR)

	7	6	5	4	3	2	1	0
CSR	"0"	"0"	"0"	"0"				
Reset state	0	0	0	0	0	0	0	0

CSR specifies the segment in program memory space which contains the program code currently being executed. CSR is an independent 8-bit register; it is not allocated as an SFR. CSR may be written by FJ, FCAL, VCAL, FRT, and RTI instructions, as well as interrupts. There is no other way to write to CSR. Be sure that the destination address of an FJ or FCAL instruction is within segment 0-15.

Offset addresses 0-0FFFFH are assigned within each segment. Address calculations that determine an addressing object are performed on 16-bit offset addresses, with any overflows and underflows ignored. Similarly, if the PC overflows, CSR will not change. Therefore, unless CSR is written using one of the above methods, program execution will not proceed beyond code segment boundaries. After reset the value of CSR is 00H.

When program memory space has been expanded to 1MB and an interrupt occurs, the current CSR and PC values will both be saved automatically on the stack. The saved values will be restored by execution of an RTI instruction. (Refer to Section 3.1.1, “Memory Space Expansion.”)

(2) Table Segment Register (TSR)

	7	6	5	4	3	2	1	0
TSR	"0"	"0"	"0"	"0"				
Reset state	0	0	0	0	0	0	0	0

address : 0008[H]
R/W access : R/W

Be sure to write "0" in "0".
When read, "0" is output.

TSR specifies the segment in program memory space which contains table data. TSR is an 8-bit register allocated in SFR space. However, TSR is rewritable, only when MEMSCON bit 1 (LROM) is set to ("1"). TSR can be written by instructions that have SFR addressing. Data in the table segment can be accessed using ROM reference instructions (LC, LCB, CMPC, CMPCB). By using the ROM window function, RAM addressing can be used for the table segment.

Bits 3-0 of TSR are valid. When read, bits 7-4 will be "0". Always write 0 to bits 7-4.

Offset addresses 0-0FFFFH are assigned within each segment. Address calculations that determine an addressing object are performed on 16-bit offset addresses, with any overflows and underflows ignored. Therefore, TSR will not be changed by these operations. After reset the value of TSR is 00H.

(3) Data Segment Register (DSR)

	7	6	5	4	3	2	1	0
DSR	"0"	"0"	"0"	"0"				
Reset state	0	0	0	0	0	0	0	0

DSR specifies the segment in data memory space which contains data currently being used. DSR is an 8-bit register allocated in SFR space. DSR can be written by instructions that have SFR addressing. However, DSR is rewritable, only when MEMSCON bit 0 (LRAM) is set to ("1").

Bits 3-0 of DSR are valid. When read, bits 7-4 will be "0". Always write "0" to bits 7-4.

3.3 Addressing Modes

There are two independent memory spaces: data memory space and program memory space. Addressing modes are classified into two categories corresponding to each memory space.

Data memory space consists of memory that can be read and written (RAM), so it is also called RAM space. Addressing for this space is called RAM addressing.

Program memory space consists of read-only memory (ROM), so it is also called ROM space. Addressing for this space is called ROM addressing.

ROM addressing is further classified as immediate addressing for instruction codes, table data addressing for ROM space data, (normally read-only data) and program code addressing for programs in ROM space.

There is also a special addressing called ROM window addressing. It allows table data to be accessed in ROM space using RAM addressing. This is a mode for reading data in the table segment through a data segment window opened by program specification.

3.3.1 RAM Addressing

RAM addressing modes specify addressing of program variables in RAM space. Addressing modes provided are register addressing, page addressing, direct data addressing, pointing register indirect addressing, and special bit area addressing.

(1) Register Addressing

A. Accumulator Addressing	A
B. Control Register Addressing	PSW, LRB, SSP
C. Pointing Register Addressing	X1, X2, DP, USP
D. Local Register Addressing	ERn, Rn

A. Accumulator Addressing

Word instructions access the accumulator contents (A). Byte instructions access the low byte of the accumulator (AL).

```
[Word]
    L          A, #1234H
    ST         A, VAR

[Byte]
    LB         A, #12H
    STB        A, VAR

[Bit]
    MB         C, A.3
    JBS        A.3, LABEL
```

B. Control Register Addressing

Register contents are accessed.

SSP:	System Stack Pointer
LRB:	Local Register Base
PSW:	Program Status Word
PSWH:	Program Status Word High byte
PSWL:	Program Status Word Low byte
C:	Carry Flag

```
[Word]
    FILL       SSP
    MOV        LRB, #401H
    CLR        PSW

[Byte]
    CLRB       PSWH
    INCB       PSWL

[Bit]
    MB         C, BITVAR
```

C. Pointing Register Addressing

Pointing register contents are accessed. Pointing registers are provided with eight sets of registers (PR0-PR7: every 8-byte in memory 200H-23FH), but the set addressed in this mode is specified by the System Control Base (SCB) field in PSW.

X1:	Index Register 1
X2:	Index Register 2
DP:	Data Pointer
USP:	User Stack Pointer
X1L:	Index Register 1 Low byte
X2L:	Index Register 2 Low byte
DPL:	Data Pointer Low byte
DP*:	Data Pointer Low byte
USPL:	User Stack Pointer Low byte

* Used only for “JRNZ DP, radr” instruction provided for nX-8/100-400 compatibility.

```
[Word]
L       A, X1
ST      A, X2
MOV     DP, #2000H
CLR     USP

[Byte]
DJNZ    X1L, LOOP
DJNZ    X2L, LOOP
DJNZ    DPL, LOOP
DJNZ    USPL, LOOP
JRNZ    DP, LOOP
```

D. Local Register Addressing

Local register contents are accessed. Local registers are 256 8-byte sets of registers (in memory 200H-9FFH), but the set addressed in this mode is specified by the Local Register Base (LCB) low byte.

ER0-ER3:	Expanded Local Registers
R0-R7:	Local Registers

```
[Word]
L          A, ER0
MOV        ER2, ER1
CLR        ER3

[Byte]
LB         A, R0
ADDB       R1, A
CMPB       R2, #12H
INCB       R3
ROR        R4
MOVB       R5, R6

[Bit]
SB         R0.0
RB         R1.7
JBRS       R7.3, LABEL
```

(2) Page Addressing

A. SFR Page Addressing	sfr Dadr
B. FIXED Page Addressing	fix Dadr
C. Current Page Addressing	off Dadr

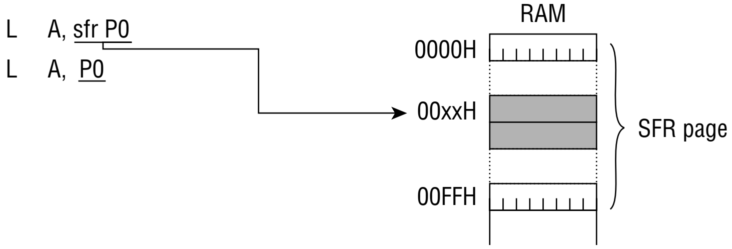
A. SFR Page Addressing

SFR page addressing specifies an offset in the SFR page (data memory 0-0FFH) with one byte of instruction code. Word, byte, or bit data can be accessed at the specified address.

The operand is coded with the “sfr” addressing specifier. The “sfr” can be omitted, but then SFR page addressing will only be used when the assembler recognizes that an address is in the SFR page.

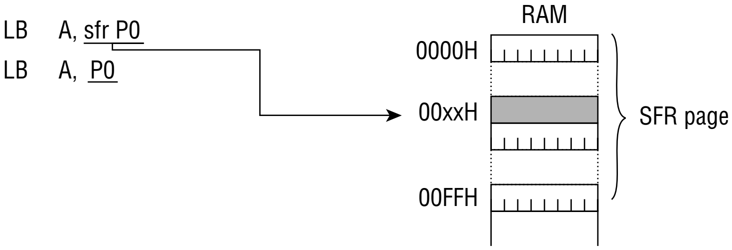
An address symbol is provided for every SFR. Normally these symbols are used for SFR accesses.

[Word]

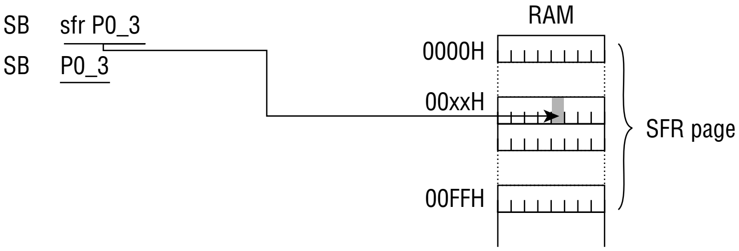


If an odd address is specified, then the word data starting at the next lower even address will be accessed (see "word boundary"). However, there are exceptions depending on the SFR.

[Byte]



[Bit]

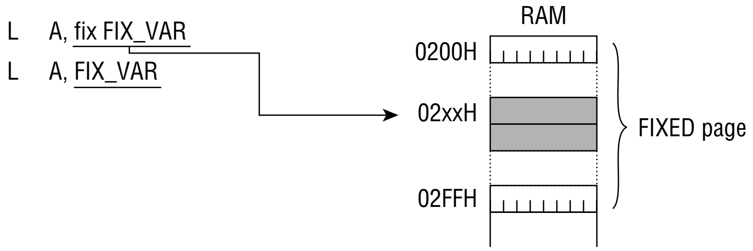


B. FIXED Page Addressing

FIXED page addressing specifies an offset in the FIXED page (data memory 200H-2FFH) with one byte of instruction code. Word, byte, or bit data can be accessed at the specified address.

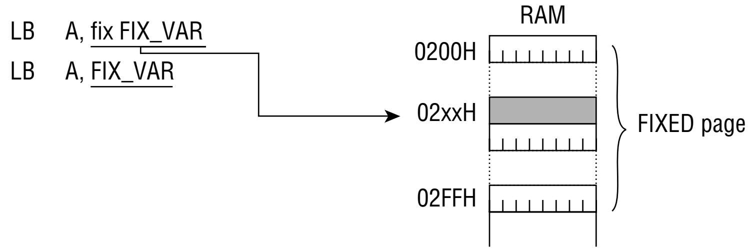
The operand is coded with the "fix" addressing specifier. The "fix" can be omitted, but then fixed page addressing will only be used when the assembler recognizes that an address is in the FIXED page.

[Word]

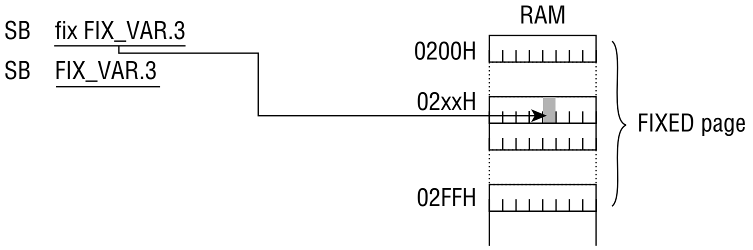


If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



[Bit]

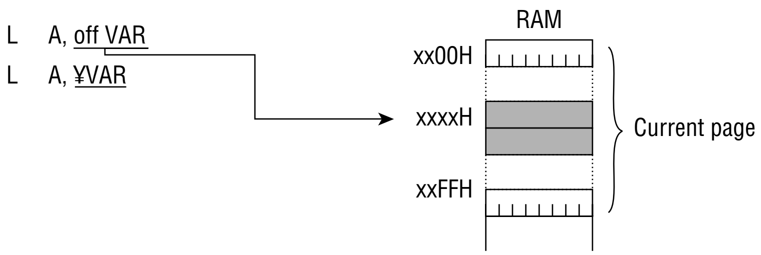


C. Current Page Addressing

Current page addressing specifies an offset in the current page (one of 256 pages in data memory specified by LRBH) with one byte of instruction code. Word, byte, or bit data can be accessed at the specified address.

The operand is coded with the “off” addressing specifier. The “off” can be replaced by "¥", but this will have a slightly different meaning when bit data in the SBA area is accessed (see “sbaoff Badr”).

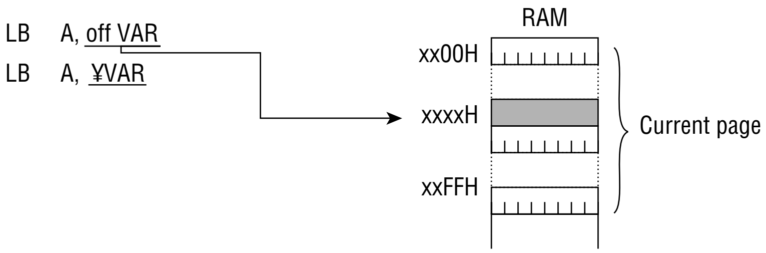
[Word]



```
L    A, off VAR
L    A, ¥VAR
```

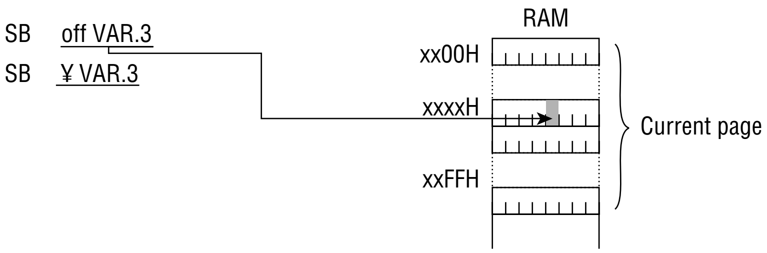
If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



```
LB    A, off VAR
LB    A,  ¥VAR
```

[Bit]



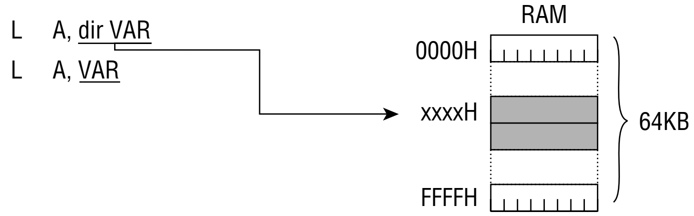
```
SB    off VAR.3
SB    ¥ VAR.3
```

(3) Direct Data Addressing

Direct page addressing specifies an address in the current physical segment of data memory(address 0-0FFFFH: 64KB) with two bytes of instruction code. Word, byte, or bit data can be accessed at the specified address.

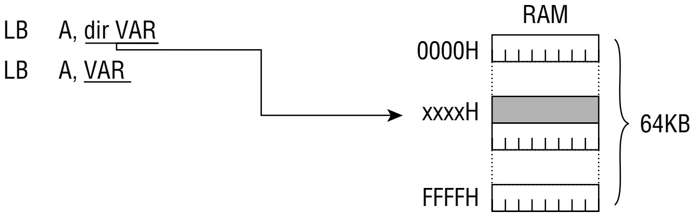
The operand is coded with the “dir” addressing specifier. The “dir” can be omitted, but then if an address in the SFR page or FIXED page is specified then the assembler may interpret it as SFR page addressing or FIXED page addressing.

[Word]

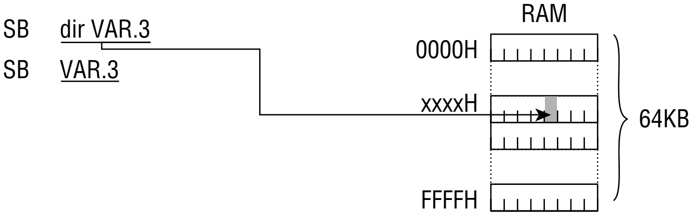


If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



[Bit]



(4) Pointing Register Indirect Addressing

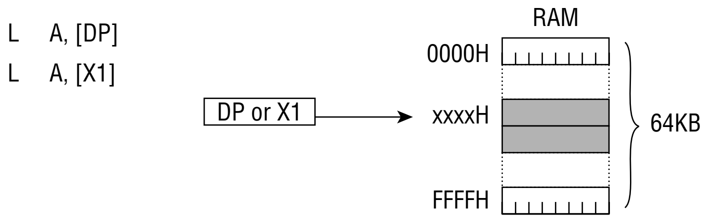
A. DP/X1 Indirect Addressing	[DP], [X1]
B. DP Indirect Addressing With Post-Increment	[DP+]
C. DP Indirect Addressing With Post-Decrement	[DP-]
D. DP/USP Indirect Addressing With 7-Bit Displacement	n7[DP], n7[USP]
E. X1/X2 Indirect Addressing With 16-Bit Base	D16[X1], D16[X2]
F. X1 Indirect Addressing With 8-Bit Register Displacement	[X1+R0], [X1+A]

A. DP/X1 Indirect Addressing

DP/X1 indirect addressing specifies an address in the current physical segment (address 0-0FFFFH: 64KB) by the contents of a pointing register. Word, byte, or bit data can be accessed at the specified address.

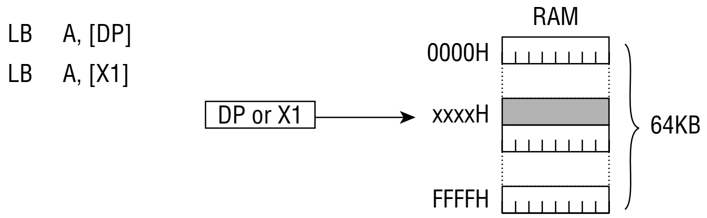
[DP] : DP indirect addressing
[X1] : X1 indirect addressing

[Word]

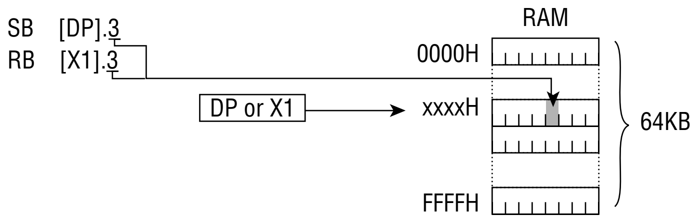


If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



[Bit]



B. DP Indirect Addressing With Post-Increment

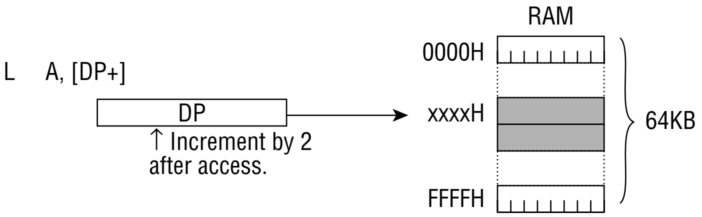
DP indirect addressing with post-increment specifies an address in the current physical segment (address 0-0FFFFH: 64KB) by the contents of a pointing register. Word, byte, or bit data can be accessed at the specified address.

After access, the pointing register contents will be incremented. The increment will be 2 for word instructions and 1 for byte and bit instructions.

This mode is primarily used to access consecutive array elements.

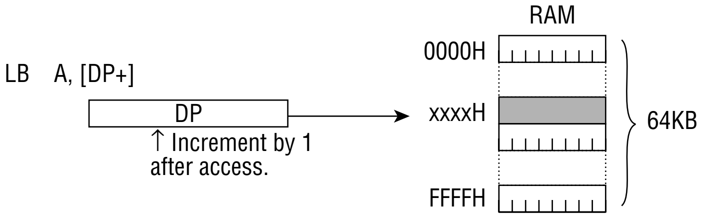
[DP+] : DP indirect addressing with post-increment

[Word]

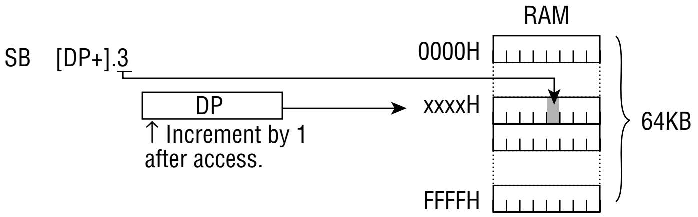


If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



[Bit]



C. DP Indirect Addressing With Post-Decrement

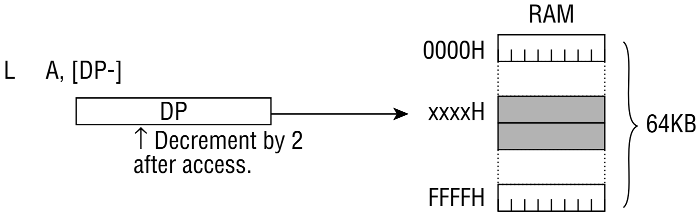
DP indirect addressing with post-decrement specifies an address in the current physical segment of data memory(address 0-0FFFFH: 64KB) by the contents of a pointing register. Word, byte, or bit data can be accessed at the specified address.

After access, the pointing register contents will be decremented. The decrement will be 2 for word instructions and 1 for byte and bit instructions.

This mode is primarily used to access consecutive array elements.

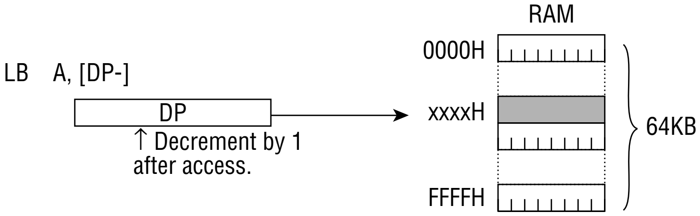
[DP-] : DP indirect addressing with post-decrement

[Word]

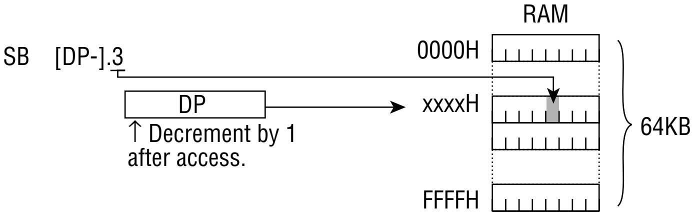


If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



[Bit]



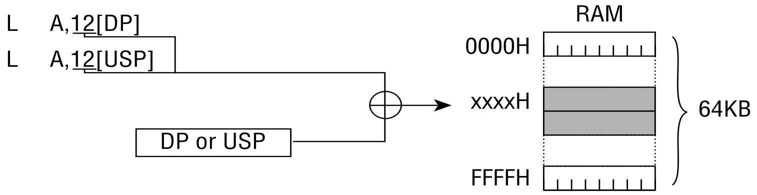
D. DP/USP Indirect Addressing With 7-Bit Displacement

DP/USP indirect addressing with 7-bit displacement specifies an address in the current physical segment (address 0-0FFFFH: 64KB) using the contents of a pointing register as a base and adding a 7-bit displacement with sign embedded in instruction code (bits 6-0). The range –64 to +63 bytes around the pointing register value can be accessed. Word, byte, or bit data can be accessed at the specified address.

numeric_expression[DP] :	DP indirect addressing with 7-bit displacement
numeric_expression[USP] :	USP indirect addressing with 7-bit displacement

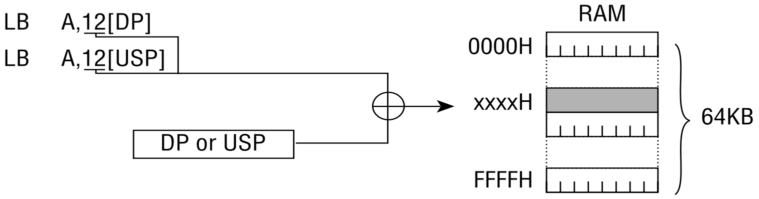
The *numeric_expression* is a value in the range –64 to +63. DP and USP can be used as the pointing register.

[Word]

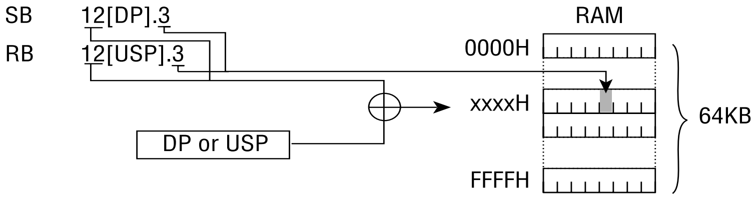


If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



[Bit]



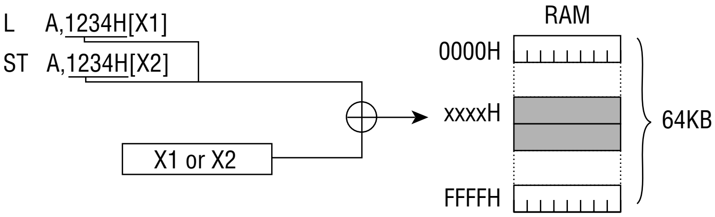
E. X1/X2 Indirect Addressing With 16-Bit Base

X1/X2 indirect addressing with 16-bit base specifies a 2-byte (D16) base embedded in instruction code and adds it to the contents of an index register (X1 or X2) to obtain an address in the current physical segment (address 0-0FFFFH: 64KB). Word (16-bit) calculations are used to generate the address, with overflows ignored. Therefore the generated address will be 0-0FFFFH. Word, byte, or bit data can be accessed at the specified address.

address_expression[X1] :	X1 indirect addressing with 16-bit base
address_expression[X2] :	X2 indirect addressing with 16-bit base

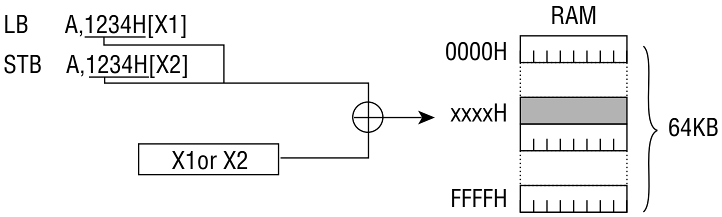
The *address_expression* is a value in the range 0-0FFFFH. However, the assembler allows a value in the range of –8000H to +0FFFFH. That is, D16 can also be thought of as a displacement instead of a base address.

[Word]

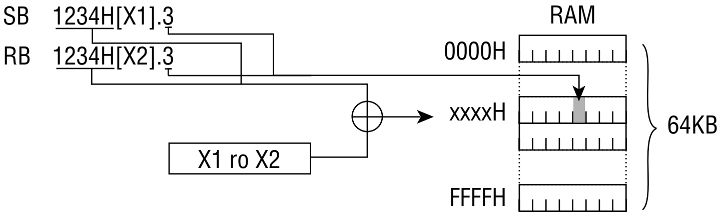


If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



[Bit]

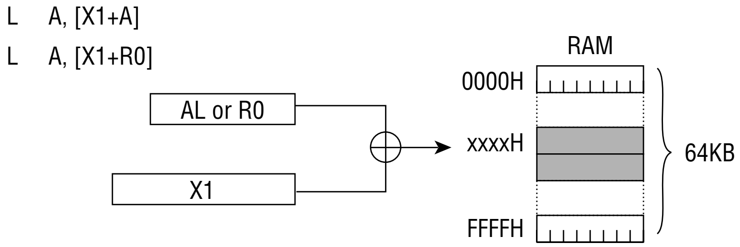


F. X1 Indirect Addressing With 8-Bit Register Displacement

X1 indirect addressing with 8-bit register displacement specifies an address in the current physical segment (address 0-0FFFFH: 64KB) using the contents of a pointing register as a base and the contents of the Accumulator low byte (AL) or Local Register 0 (R0). Word (16-bit) calculations are used to generate the address. The 8-bit displacement obtained from the register will be extended without sign, and overflow will be ignored so the generated address will be 0-0FFFFH. Word, byte, or bit data can be accessed at the specified address.

[X1+A] :	X1 indirect addressing with 8-bit register displacement (AL)
[X1+R0] :	X1 indirect addressing with 8-bit register displacement (R0)

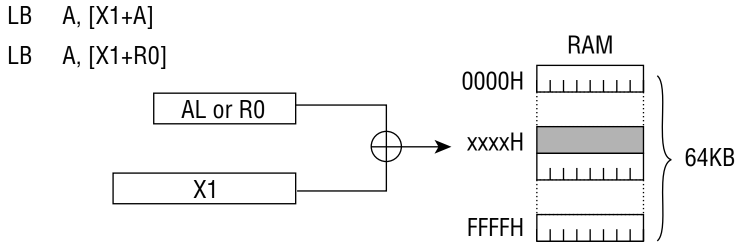
[Word]



```
L    A, [X1+A]
L    A, [X1+R0]
```

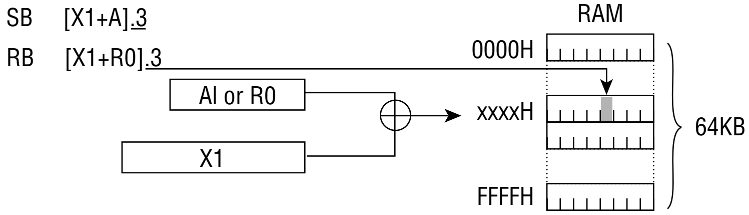
If an odd address is specified, then the word data starting at the next lower even address will be accessed.

[Byte]



```
LB    A, [X1+A]
LB    A, [X1+R0]
```

[Bit]



```
SB    [X1+A].3
RB    [X1+R0].3
```

(5) Special Bit Area Addressing

A. FIXED page SBA area addressing	sbafix Badr
B. Current page SBA area addressing	sbaoff Badr

A. FIXED page SBA area addressing

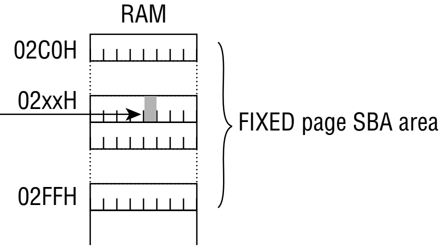
FIXED page SBA area addressing specifies a bit address in the FIXED page’s 512-bit SBA area (2C0H.0-2FFH.7).   Only bit data can be accessed at the specified address.

The instructions that can use this addressing are SB, RB, JBS, and JBR.

[Bit]

```
SB    sbafix 2C0H.0
RB    sbafix 1600H
JBS   sbafix VAR,LABEL
JBR   sbafix 2EFH.7


SB    2C0H.0
RB    1600H
JBS   VAR,LABEL
JBR   2EFH.3,LABEL
```



B. Current page SBA area addressing

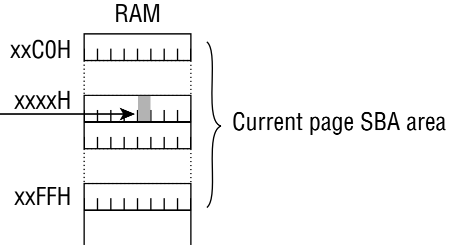
Current page SBA area addressing specifies a bit address in the current page’s 512-bit SBA area (xxC0H.0-xxFFH.7). Only bit data can be accessed at the specified address.

The instructions that can use this addressing are SB, RB, JBS, and JBR.

[Bit]

```
SB    sbaoff 4C0H.0
RB    sbaoff 2E80H
JBS   sbaoff VAR,LABEL
JBR   sbaoff 0FFFFH.3,LABEL

SB    2C0H.0
RB    2E80H
JBS   VAR,LABEL
JBR   0FFFFH.3,LABEL
```



3.3.2 ROM Addressing

ROM addressing specifies addressing of program variables in ROM space. The modes provided are immediate addressing, table data addressing, and program code addressing.

(1) Immediate Addressing

Immediate addressing specifies access of immediate data embedded in instruction code. For words, two bytes (N16) in instruction code will be accessed. For bytes, one byte (N8) in instruction code will be accessed.

Word values are expressions in the range 0-0FFFFH. Byte values are expressions in the range 0-0FFH. However, the assembler permits values in the range covered by both signed and unsigned expressions. For words that range is from –8000H to +0FFFFH, and for bytes it is from –80H to +0FFH.

```
[Word]
L        A, #1234H
MOV      X1, #WORD_ARRAY_BASE

[Byte]
LB       A, #12H
MOV      X1, #BYTE_ARRAY_BASE
```

(2) Table Data Addressing

Table data addressing specifies access for the 64KB in the table segment of ROM space as specified by TSR. This mode can be used with operands of LC, LCB, CMPC, and CMPCB instructions.

A. Direct Table Addressing	Tadr
B. RAM Indirect Table Addressing	[**]
C. RAM Indirect Addressing With 16-Bit Base	T16[**]

A. Direct Table Addressing

Direct table addressing specifies an address (0-0FFFFH: 64KB) in the table segment specified by TSR with two bytes of instruction code. Word or byte data can be accessed at the specified address.

This addressing can be used with the four instructions LC, LCB, CMPC, and CMPCB.

```
[Word]
LC        A, VAR
CMPC      A, VAR

[Byte]
LCB       A, VAR
CMPCB     A, VAR
```

B. RAM Indirect Table Addressing

RAM indirect table addressing uses the word data specified by RAM addressing as a pointer to the table segment specified by TSR. Table memory can be accessed by using a register or data memory as a pointer into the table memory.

This addressing can be used with the four instructions LC, LCB, CMPC, and CMPCB.

```
[Word]
LC        A, [A]
CMPC      A, [1234[X1]]

[Byte]
LCB       A, [ER0]
CMPCB     A, [VAR]
```

C. RAM Indirect Addressing With 16-Bit Base

RAM indirect addressing with 16-bit base specifies two bytes (D16) in instruction code as a base and adds it to the contents of word data specified by RAM addressing to obtain an address (0-0FFFFH: 64KB) in the table segment specified by TSR. Word (16-bit) calculations are used to generate the address, with overflows ignored. Therefore the generated address will be 0-0FFFFH. Word or byte data can be accessed at the specified address.

This mode can be used with operands of LC, LCB, CMPC, and CMPCB instructions.

```
[Word]
LC        A, 2000H [A]
CMPC      A, 2000H [1234[X1]]

[Byte]
LCB       A, 2000H [ER0]
CMPCB     A, 2000H [VAR]
```

(3) Program Code Addressing

Program code addressing specifies access of the current program code in ROM space. These modes are used as operands of branch instructions.

A. Near Code Addressing	Cadr
B. Far Code Addressing	Fadr
C. Relative Code Addressing	radr
D. ACAL Code Addressing	Cadr[]
E. VCAL Code Addressing	Vadr
F. RAM Indirect Code Addressing	[**]

A. Near Code Addressing

Near code addressing specifies an address (0-0FFFFH: 64KB) in the current code segment with two bytes of instruction code. This addressing can be used with J and CAL instructions.

[Usage Example]

```
J       3000H
CAL     LABEL
```

B. Far Code Addressing

Far code addressing specifies an address (0:0-F:0FFFFH: 1MB) in program memory space with three bytes of instruction code. This addressing can be used with FJ and FCAL instructions.

[Usage Example]

```
FJ      1:3000H
FCAL    FARLABEL
```

C. Relative Code Addressing

Relative code addressing takes the current program counter (PC) value as a base and adds it to an 8-bit or sign-extended 7-bit value in instruction code to obtain an address (0-0FFFFH: 64KB) in the current code segment. Word (16-bit) calculations are used to generate the address, with overflows ignored. Therefore the generated address will be 0-0FFFFH. This addressing can be used with SJ and conditional branch instructions.

[Usage Example]

```
SJ      LABEL
DJNZ    R0, LABEL
JC      LT, LABEL
```

D. ACAL Code Addressing

ACAL code addressing specifies an address in the ACAL area (1000H-17FFH: 2KB) in the current code segment with 11 bits of instruction code. This addressing can be used only with ACAL instructions.

[Usage Example]

```
ACAL    1000H
ACAL    ACALLABEL
```

E. VCAL Code Addressing

VCAL code addressing specifies an entry (word data) in the vector table for VCAL instructions with 4 bits of instruction code. The vector table is located at even addresses in the range 004AH-0069H. This addressing can be used only with VCAL instructions.

[Usage Example]

```
VCAL    4AH
VCAL    0:4AH
VCAL    VECTOR
```

F. RAM Indirect Code Addressing

RAM indirect code addressing uses word data specified by RAM addressing as a pointer to the code segment. It allows indirect jumps and calls using a register or data memory as a pointer to code memory. This addressing can be used with J and CAL instructions.

[Usage Example]

```
J       [A]
CAL     [1234[X1]]
```

(4) ROM Window Addressing

ROM window addressing accesses table data in ROM space using RAM addressing. This mode reads data in the table segment specified by TSR using data segment window opened by the program.

Data memory addressing is permitted in the ROM window area, but results are not guaranteed if an instruction that writes to the ROM window is executed.

Chapter 4

CPU Control Functions

4. CPU Control Functions

There are two CPU control functions.

- Standby function
- Reset function

4.1 Standby Function

There are two operating modes of the standby function.

- HALT mode: clock supplied to the CPU is stopped by software
- STOP mode: oscillation clock supply is stopped by software

The modes are set by the following hardware.

- Stop Code Acceptor (STPACP: for STOP mode)
- Standby Control Register (SBYCON: for HALT and STOP modes)

Table 4-1 shows the standby modes and the states of the output pins in each.

Table 4-1 Standby Modes

Standby Mode		HALT Mode	STOP Mode (note 1)	
Set condition		• SBYCON bit 1 (HLT) set.	• SBYCON bit 0 (STP) set when bit 2 (FLT) is in set state ("1").	• SBYCON bit 0 (STP) set when bit 2 (FLT) is in reset state ("0").
Release condition		• Interrupt • $\overline{\text{RES}}$ pin input	• Interrupt (note 6) • $\overline{\text{RES}}$ pin input	
Output Pin States	P0	Unchanged	(note 4) High impedance or Pull-up	(note 2)
	P1, P4, P9, P10	Unchanged		Unchanged
	P2, P6, P7_2-P7_7, P8, P12	Unchanged	High impedance	Unchanged
	P5_4, P5_5 (primary function)	Unchanged	High impedance	Unchanged
	P5_4 (secondary function)	High level	High impedance	High level
	P5_5 (secondary function)	Low level	High impedance	Low level
	P7_0, P7_1 (primary function)	Unchanged	High impedance	Unchanged
	P7_0, P7_1 (secondary function)	High level (pull-up)	High level (pull-up) (note 5)	High level (pull-up)
Internal Function Operation	TBC	Operating	Stopped	
	16-bit RTO/PWM timer	Operating	Stopped	
	GTMC	Operating	Operating if external clock selected (note 3)	
	S1TM	Operating	Stopped	
	SCI1	Operating	Stopped	
	ADC	Operating	Stopped	
	PWM	Operating	Stopped	

Notes:
1. The set condition for STOP mode is that the Stop Code Acceptor (STPACP) already be set ("1").
2. P0 will be high impedance in external ROM mode, or unchanged in internal ROM mode.
3. Edge specification is disabled, so the falling edge will be assumed.
4. Pins specified to pull-up will be kept in pull-up state regardless of the mode.
5. ICE output is high impedance, so it differs from the device operation(MASK version, OTP version, etc.).
6. The edge specification for external interrupts (INT0-INT3) is disabled and the "L" level is enabled.

4.1.1 Standby Control Register (SBYCON)

SBYCON is a 5-bit register that controls standby functions. Also, the Stop Code Acceptor (STPACP) is an acceptor used to set STOP mode.

Figure 4-1 shows the configuration of SBYCON.

	7	6	5	4	3	2	1	0
SBYCON	—	—	OST1	OST0	—	FLT	HLT	STP
Reset state	1	1	0	0	1	0	0	0

address : 000F[H]
R/W access : R/W

— indicates a non-existent bit. These will read as "1".

Figure 4-1 SBYCON Configuration

- STP (bit 0)

Writing n5H and nAH (n = 0-F) consecutively to STPACP will set ("1") the stop code acceptor. Setting ("1") the STP bit of SBYCON at that point will set the device in STOP mode. STP will be reset (0) when an interrupt or reset from $\overline{\text{RES}}$ pin input occurs, releasing STOP mode. For details refer to Section 4.1.2 (2), “STOP Mode.” STPACP will be reset ("0") at the same time STOP mode is entered.

- HLT (bit 1)

Setting ("1") the HLT (bit 1) of SBYCON will set the device in HALT mode. HLT will be reset ("0") when an interrupt or reset from $\overline{\text{RES}}$ pin input occurs, releasing halt mode. For details refer to Section 4.1.2 (1), “HALT Mode.”

- FLT (bit 2)

If STOP mode is entered when the FLT bit of SBYCON is set ("1"), then ports, control signals, and all other output pins will enter a high impedance state.
(However, some output pins may change their states by programmable pull-up settings/ secondary function setting. Refer to Table 4-1 Standby Modes.)

At this time, the internal circuits of the high-impedance pins will activate a circuit that prevents internal stray current even if the pins are left open. However, for the purpose of STOP mode release condition, the stray current prevention circuit will not operate for pins like external interrupts (INT0-INT3) and the event timer clock input pin (ETMCK). These pins need to be fixed to a high or low level when the FLT bit of SBYCON is set ("1") to enter STOP mode.

When the above pins are used as their primary function (I/O ports), the stray current prevention circuit will operate, so they have no need to be fixed at a logic level.

If STOP mode is entered when the FLT bit of SBYCON is reset ("0"), then ports, control signals, and other output pins will maintain their state at that time.

Pins in input mode should be fixed to a high or low level regardless of the FLT bit content.

- OST0 (bit 4), OST1 (bit 5)

  The OST0 and OST1 bits of SBYCON set the time after STOP mode is released by an interrupt and the oscillation clock starts oscillating until the clock is supplied to the CPU.

OST1	OST0	Selected Clock Cycles
0	0	32768
0	1	16384
1	0	8192
1	1	Setting prohibited

Note: Do not set OST1, 0 to "1" simultaneously.

4.1.2 Operation in Standby Modes

(1) HALT Mode

When HLT (bit 1) of SBYCON is set ("1"), the device will enter HALT mode. In HALT mode the oscillation clock operates, so TBC, timers, the serial ports, etc. will operate as well. However, the clock supplied to the CPU will stop, so instructions will not be executed. Execution will stop at the start of the instruction following the one that set ("1") HLT.

HALT mode will be released when either an interrupt or reset by $\overline{\text{RES}}$ input occurs.

When a non-maskable interrupt occurs in HALT mode, HALT mode will be released unconditionally and the CPU will execute the non-maskable interrupt process.

When a maskable interrupt occurs in HALT mode, HALT mode will be released when both its Interrupt Request Flag (IRQ bit) and Interrupt Enable Flag (IE) are set ("1").

If the Master Interrupt Enable Flag (MIE of the PSW) has been set ("1"), then after HALT mode is released the CPU will execute the maskable interrupt process for the interrupt. If it has been reset ("0"), then the CPU will start execution at the instruction after the instruction that set HALT mode (set the HLT bit of SBYCON).

If interrupt priority has been set (MIP = 1) and HALT mode is entered from within a high-priority interrupt routine, then HALT mode can be released by interrupt requests of lower priority but the lower priority interrupt process will not be executed even though MIE is 1. Instead the CPU will start execution at the instruction after the instruction that set HALT mode.

When HALT mode is released by $\overline{\text{RES}}$ pin input, the CPU will perform reset processing.

(2) STOP Mode

The device will enter STOP mode when n5H followed by nAH (n = 0-FH) are written to STPACP to set (1) the Stop Code Acceptor, and then STP (bit 0) of SBYCON is set (1). The Stop Code Acceptor will be reset (0) at the same time stop mode is entered.

STOP mode stops the oscillation clock, so TBC, timers, the serial port, etc. will also stop. However, the general 8-bit timer (GTMC) will operate if an external clock has been selected. The valid edge specification for the external clock will be disabled, so it will operate on the falling edge.

The clock supply to the CPU will also stop, so instructions will not be executed. Execution will stop at the instruction after the instruction that set ("1") STP (bit 0) in SBYCON.

STOP mode will be released when either an interrupt or reset by $\overline{\text{RES}}$ input occurs. The edge specification for external interrupts (INT0-INT3) is disabled and the "L" level is enabled.

When a non-maskable interrupt occurs in STOP mode, STOP mode will be released unconditionally and the CPU will execute the non-maskable interrupt process.

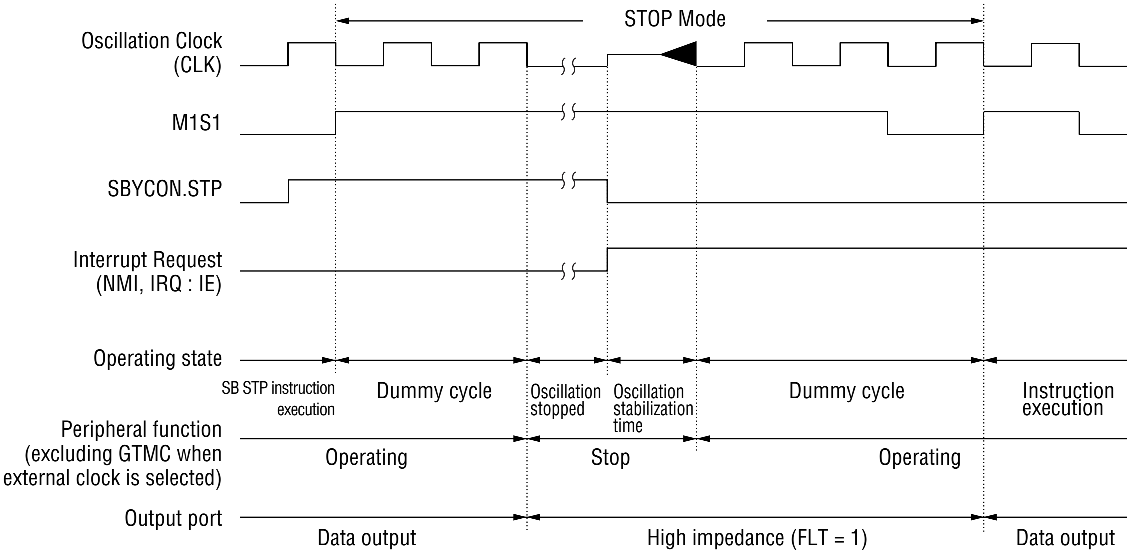
When a maskable interrupt occurs in STOP mode, STOP mode will be released when both its Interrupt Request Flag (IRQ bit) and Interrupt Enable Flag (IE) are set ("1").

If the Master Interrupt Enable Flag (MIE of the PSW) has been set ("1"), then after STOP mode is released the CPU will execute the maskable interrupt process for the interrupt. If it has been reset ("0"), then the CPU will start execution at the instruction after the instruction that set STOP mode (set the STP bit of SBYCON).

If STOP mode is set during an NMI routine and then released by an interrupt, execution after release will start at the instruction after the instruction that set STOP mode. Also, if interrupt priority has been set (MIP = 1) and STOP mode is entered from within a high-priority interrupt routine, then STOP mode can be released by interrupt requests of lower priority but the lower priority interrupt process will not be executed even though MIE is 1. Instead the CPU will start execution at the instruction after the instruction that set STOP mode.

Figure 4-2 shows STOP mode timing.

When STOP mode is released by $\overline{\text{RES}}$ pin input, the CPU will perform its reset processing. When releasing by $\overline{\text{RES}}$ pin input, apply a low level on the $\overline{\text{RES}}$ pin until the oscillation clock is stable.



* The oscillation stabilization time is the time until the oscillation clock begins oscillating, and the time set by OST0 and OST1 as a number of clock cycles.

Figure 4-2 STOP Mode Timing Diagram (Release By Interrupt)

4.2 Reset Function

There are three ways to reset the device.

- Input a low level on the $\overline{\text{RES}}$ pin
- Execute a BRK (break) instruction
- Generate an OPTRP (op code trap)

These three methods are identical in all ways except that the location of the vector address loaded into the Program Counter during reset processing will be different.

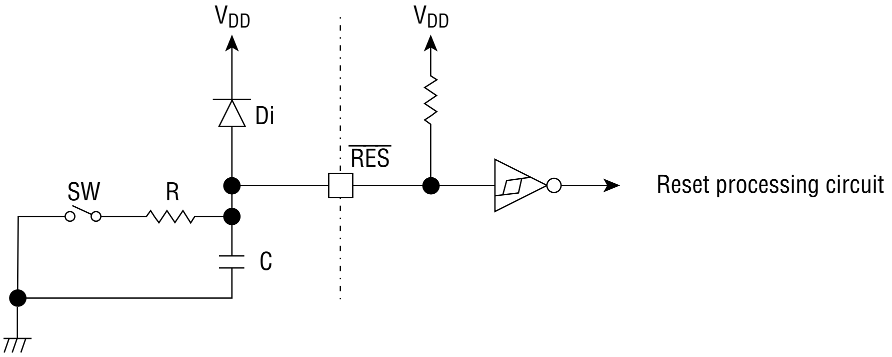
Reset processing initializes calculation registers, control registers, and mode registers. It also loads the contents of the location indicated by the vector address into the Program Counter.

See Chapter 21, “Special Function Registers (SFRs),” for the initial states of registers. See Table 3-1, “Vector Table,” for the vector address corresponding to each reset type.

When resetting by $\overline{\text{RES}}$ pin input, apply a low level on the $\overline{\text{RES}}$ pin until the oscillation clock is stable.

Reset processing has priority over all other processing (interrupt processes and instruction execution). It will abort all processes, and will not preserve register or RAM contents.

Figure 4-3 shows an example of the reset pin connection. Table 4-2 shows the reset state of output pins.



SW and R are for manual reset.

Figure 4-3 Reset Pin Connection Example

Table 4-2 Reset State of Output Pins

Name	P0	P1	P2, P4, P6-P10, P12	P5_4, P5_5 EA pin at high level
State	High impedance	High impedance	High impedance	High impedance

Note: If the $\overline{\text{EA}}$ pin is low and external program memory is selected, then P0, P1, P5_4, P5_5, and P9_0-P9_3 will automatically take their secondary functions and enter an operating state.

Chapter 5

Memory Control Functions

5. Memory Control Functions

There are two independent memory spaces, program memory space and data memory space. Several functions are provided to make using these spaces easier. One is the ROM window function, which allows the many instructions provided for data memory space to be used for program memory space.

Another is the ready function, which inserts wait cycles in external memory timing under software control so access times match the external memory being used in either memory space.

One more is the wait function, which can insert wait cycles under the control of an external device.

5.1 ROM Window Function

The ROM window function reads the contents of program memory specified by the ROM Window Control Register (ROMWIN) in the SFR area through a data memory space window at the same address.

In other words, when the ROM window function is enabled and an instruction that accesses (reads) data memory space is executed, the data in data memory space will not be accessed (read), but instead the data at the same address in program memory space in the segment specified by the TSR will be accessed (read).

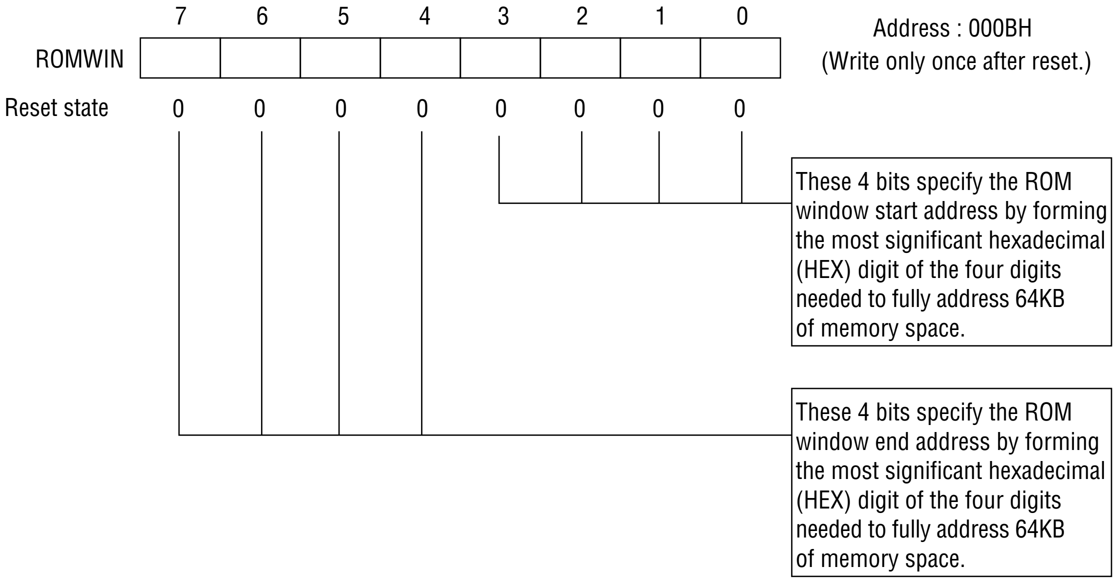
For each access (read), the number of instruction execution cycles will increase by three cycles for byte instructions and by six cycles for word instructions.

Results are not guaranteed if an instruction that performs a write operation is executed while the ROM window function is enabled. The number of cycles will not increase, though.

ROMWIN is an 8-bit register, the lower 4 bits specifying the ROM window start address, and the upper 4 bits specifying the ROM window end address. Both specify the most significant hexadecimal digit of the four digits needed to fully address 64KB of program memory space. However, if the lower 4 bits are all zeroes, then the ROM window function will not operate.

If internal RAM is located in the data memory area specified as the ROM window, then the data memory internal RAM will take precedence. (This does not apply to the MSM66587 family.)

External data memory cannot be allocated to the data memory area specified as the ROM window. Figure 5-1 shows the configuration of ROMWIN.



Address : 000BH
(Write only once after reset.)

These 4 bits specify the ROM window start address by forming the most significant hexadecimal (HEX) digit of the four digits needed to fully address 64KB of memory space.

These 4 bits specify the ROM window end address by forming the most significant hexadecimal (HEX) digit of the four digits needed to fully address 64KB of memory space.

Figure 5-1 ROMWIN Configuration

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), ROMWIN will be 00H, disabling the ROM window function.

ROMWIN may be written only one time after reset. Second and later writes will be ignored. Therefore after the ROM window function has been set once, it can be changed only after another reset. ROMWIN can be read any number of times.

Note: The device hardware does not compare the start address X and end address Y written to ROMWIN, so your program should ensure that $X \leq Y$.

5.2 READY Function

The READY function allows memory and general ICs with slow access speeds to be connected externally by specifying that wait cycles are to be inserted in external memory accesses (RAM: 0-7 cycles, ROM: 0-3 cycles). The number of wait cycles is specified by the RAM Ready Control Register (RAMRDY) and ROM Ready Control Register (ROMRDY).

RAMRDY specifies the wait cycles when data memory space has been expanded externally. The wait cycles can be set, equally for all segments, in two areas of the offset addresses 0H-7FFFH (0A00H-7FFFH for segment 0) and 8000H-FFFFH.

ROMRDY specifies the wait cycles when program memory space is used with external ROM mode.

RAMRDY and ROMRDY can be read and written by the program. However, RAMRDY bit 7 and bit 3 will not be affected by write operations. These bits will always read as "1".

The lower 2 bits of ROMRDY are valid for write operations, but the upper 6 bits will not be affected. The lower 2 bits will read correctly, but the upper 6 bits will always read as "1".

Figure 5-2 shows the configuration of RAMRDY, and Figure 5-3 shows that of ROMRDY.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), RAMRDY and ROMRDY will both be FFH, inserting 7 wait cycles when external data memory is accessed and 3 cycles when external program memory is accessed.

Notes: 1. Compared to internal program memory accesses, external program memory accesses have one cycle inserted automatically for each byte accessed. The ROMRDY register specifies the number of cycles inserted in addition to one cycle inserted automatically.

2. Compared to internal data memory accesses, external data memory accesses have 3 or 4 cycles inserted automatically for each byte accessed. The RAMRDY register specifies the number of cycles inserted in addition to 3 or 4 cycles inserted automatically.

	7	6	5	4	3	2	1	0
RAMRDY	—	ARDY12	ARDY11	ARDY10	—	ARDY02	ARDY01	ARDY00
Reset state	1	1	1	1	1	1	1	1

Address : 000DH

ARDY0			Additional cycles inserted when external data memory space of each segment (0000H-7FFFH) is accessed
2	1	0	
0	0	0	0 cycles
0	0	1	1 cycle
0	1	0	2 cycles
0	1	1	3 cycles
1	0	0	4 cycles
1	0	1	5 cycles
1	1	0	6 cycles
1	1	1	7 cycles

ARDY1			Additional cycles inserted when external data memory space of each segment (8000H-FFFFH) is accessed
2	1	0	
0	0	0	0 cycles
0	0	1	1 cycle
0	1	0	2 cycles
0	1	1	3 cycles
1	0	0	4 cycles
1	0	1	5 cycles
1	1	0	6 cycles
1	1	1	7 cycles

– indicates bits that are not provided.

It will read as "1".

1 cycle consists of 1 CLK.

Figure 5-2 RAMRDY Configuration

	7	6	5	4	3	2	1	0
ROMRDY	—	—	—	—	—	—	ORDY1	ORDY0
Reset state	1	1	1	1	1	1	1	1

Address : 000CH

ORDY		Additional cycles inserted when external program memory space is accessed
1	0	
0	0	0 cycles
0	1	1 cycle
1	0	2 cycles
1	1	3 cycles

– indicates bits that are not provided.

It will read as "1".

1 cycle consists of 1 CLK.

Figure 5-3 ROMRDY Configuration

5.3 WAIT Function

In addition to the READY function, which inserts wait cycles under CPU control, there is also a WAIT function, which can insert wait cycles under control of an external device.

More specifically, pin P7_2 (must be set to secondary function) is sampled on the falling edges of the internal clock, as shown in Figure 5-4. If its level is high, then wait cycles will be inserted into the $\overline{\text{RD}}$ and $\overline{\text{WR}}$ strobes.

As shown in the sample timing of Figure 5-4, the wait cycles are released when pin P7_2 has been sampled at a low level twice and 1 tøw.

When the WAIT function is used together with the READY function, the longer wait time will be effective (this only applies to data memory space accesses).

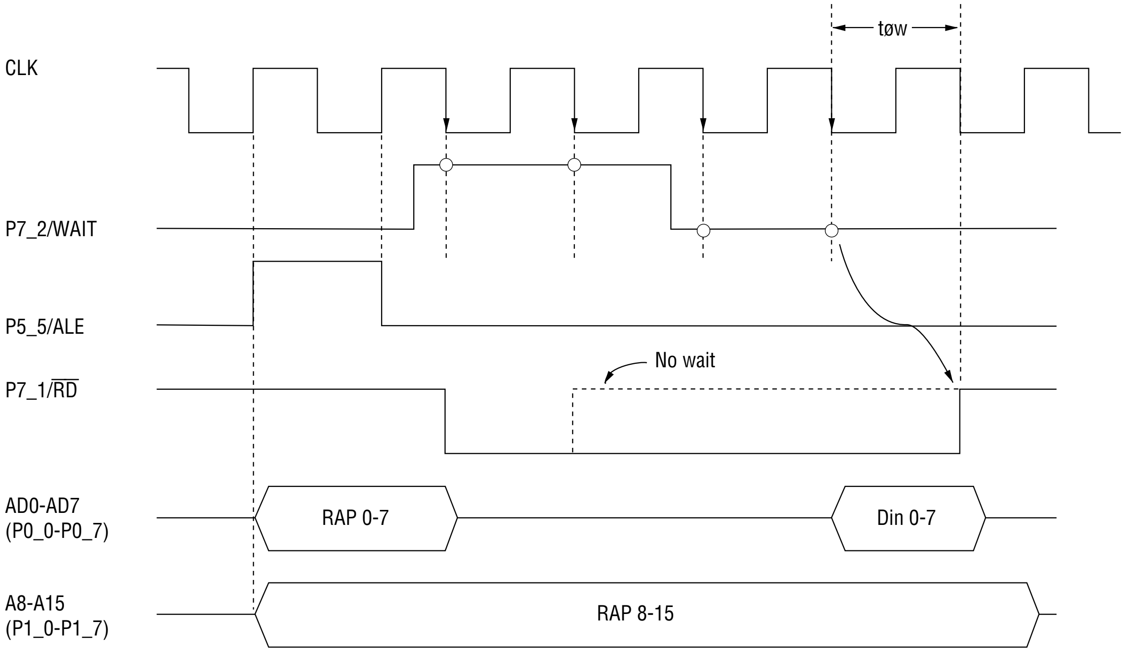


Figure 5-4 Sample Timing When WAIT Function is Used

Chapter 6

Port Functions

6. Port Functions

The MSM66587 family provides nine 8-bit, one 2-bit, and one 6-bit input/output port. Each bit of an input/output port can be specified individually.

Pins are high-impedance when inputs, and push-pull when outputs. Each port not only has port functions, but also is assigned a function for internal operation (secondary function). The bits of some ports can also be programmed individually for internal pull-up.

Table 6-1 lists the port functions.

Table 6-1 Port Functions

Port	Pin	Type	No.	Pull-up	Secondary Function
Port 0	P0_0-P0_7	A	8	Available	External memory access: AD0-AD7 (I/O)
Port 1	P1_0-P1_7	B	8	Available	External memory access: A8-A15 (O)
Port 2	P2_0-P2_3	E	4	None	None
	P2_4, P2_5	C	2	None	Flexible RTO output: RTO8, RTO9 (O)
	P2_6, P2_7	E	2	None	None
Port 4	P4_0	B	1	Available	Event timer external clock: ETMCK (I)
	P4_1-P4_7	D	7	Available	None
Port 5	P5_4	C	1	None	External memory access: $\overline{\text{PSEN}}$ (O)
	P5_5	C	1	None	External memory access: ALE (O)
Port 6	P6_0, P6_1	C	2	None	External interrupt signal inputs: INT0, INT1 (I)
	P6_2	C	1	None	Serial Port 1 data input: RXD1 (I)
	P6_3	C	1	None	Serial Port 1 data output: TXD1 (O)
	P6_4	C	1	None	Shift clock for serial receive: RXC1 (I/O)
	P6_5	C	1	None	Shift clock for serial transmit: TXC1 (I/O)
Port 7	P7_0	C	1	None	External memory access: $\overline{\text{WR}}$ (O)
	P7_1	C	1	None	External memory access: $\overline{\text{RD}}$ (O)
	P7_2	C	1	None	Wait signal input pin: WAIT (I)
	P7_3	C	1	None	Clock output: CLKOUT (O)
	P7_4	C	1	None	PWM output: PWM0 (O)
	P7_5-P7_7	E	3	None	None
Port 8	P8_0-P8_7	E	8	None	None
Port 9	P9_0-P9_3	B	4	Available	External memory access: A16-A19 (O)
	P9_4-P9_7	D	4	Available	None
Port 10	P10_0-P10_7	D	8	Available	None
Port 12	P12_0, P12_1	E	2	None	None
	P12_2, P12_3	C	2	None	External interrupt signal inputs: INT2, INT3 (I)
	P12_4-P12_7	E	4	None	None

Type A: The secondary function of these ports is to be an address/data bus. During external memory accesses, the ports automatically switch to the secondary function. They have programmable pull-ups.

Type B: These ports have a secondary function which is switched on by the state of each port's secondary function control register. They have programmable pull-ups.

Type C: These ports have a secondary function which is switched on by the state of each port's secondary function control register.

Type D: These ports do not have a secondary function. They have programmable pull-ups.

Type E: These ports do not have a secondary function.

6.1 Port Hardware Configurations

The functions of the ports (P0, P1, P2, P4, P5, P6, P7, P8, P9, P10, P12) are classified into five types.

6.1.1 Type A (P0_0-P0_7) Configuration

Type A ports automatically take their secondary function during external memory accesses. They function as low address output pins and data i/o pins. They also have programmable pull-ups.

Figure 6-1 shows the Type A configuration.

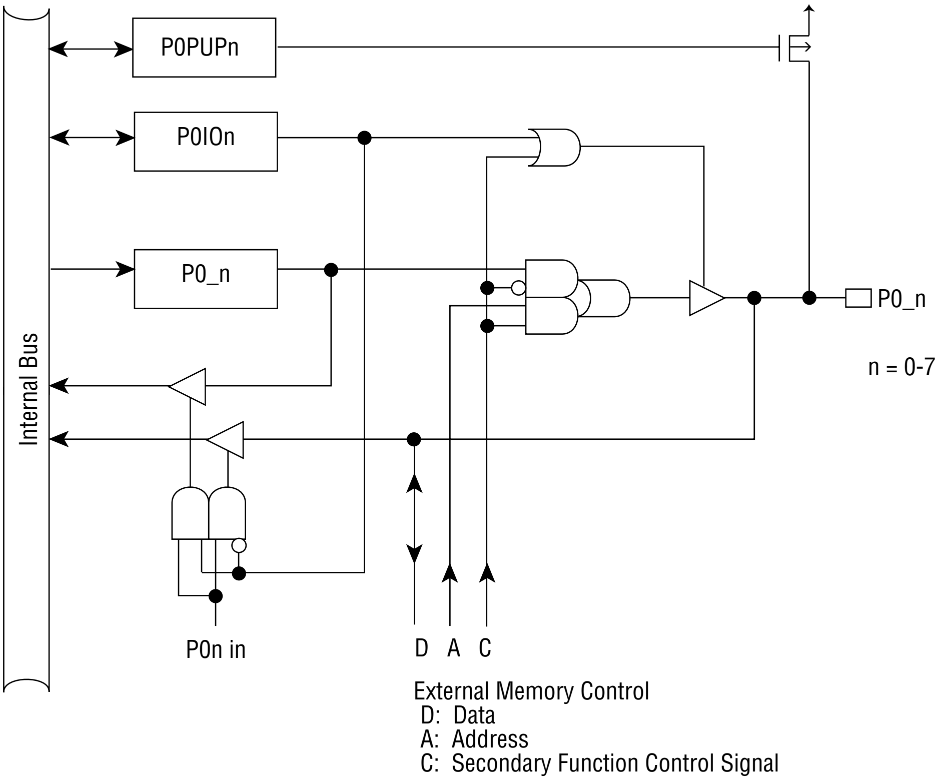


Figure 6-1 Type A Configuration

6.1.2 Type B (P1_0-P1_7, P4_0, P9_0-P9_3) Configuration

Type B ports take their secondary function as high address outputs during external data memory accesses if specified by the secondary function select register (PmSFn). They also have programmable pull-ups.

However, P1 and P9_0-P9_3 will switch to their secondary function as high address outputs automatically when the $\overline{EA}$ pin goes low.

Figure 6-2 shows the Type B configuration.

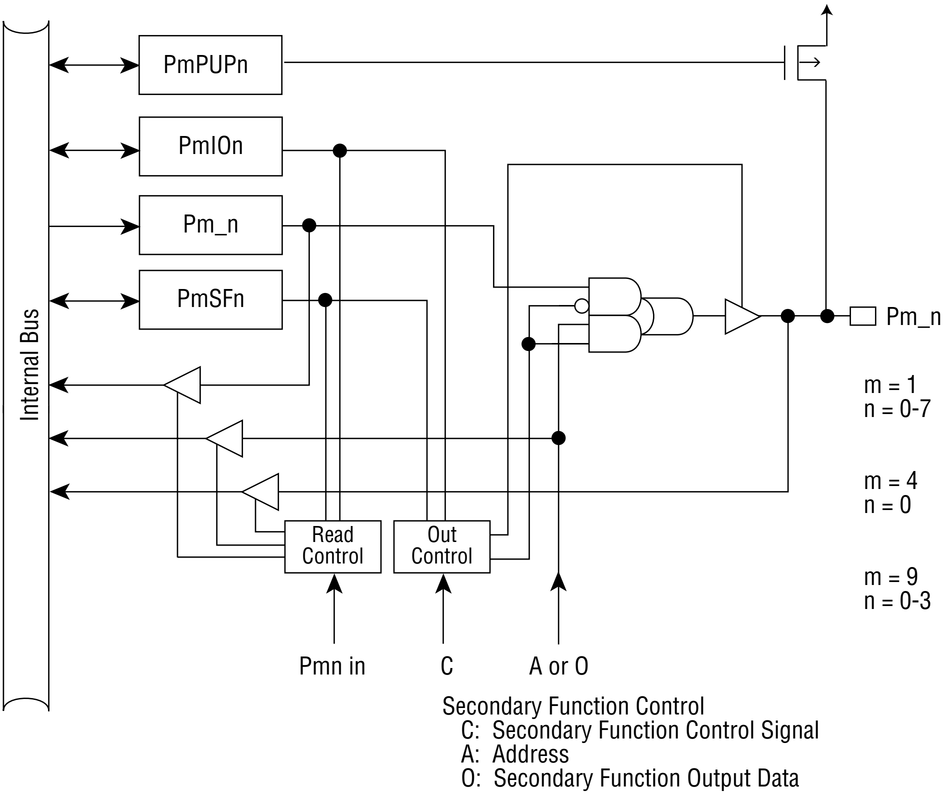


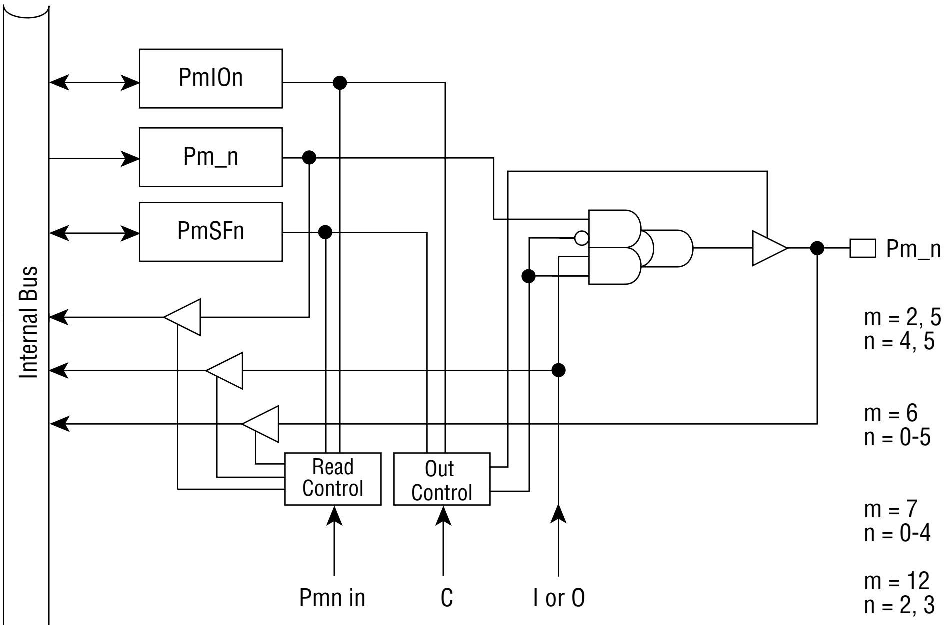
Figure 6-2 Type B Configuration

6.1.3 Type C (P2_4, P2_5, P5_4, P5_5, P6_0-P6-5, P7_0-P7_4, P12_2, P12_3) Configuration

Type C ports take their secondary function as output or input pins if specified by the secondary function select register (PmSFn).

However, P5_4 and P5_5 will switch to their secondary function automatically when the $\overline{\text{EA}}$ pin goes low. P5_4 outputs a strobe signal for read operations ($\overline{\text{PSEN}}$ pin). P5_5 outputs a strobe signal for externally latching the lower 8 address bits from P0 (ALE pin). P5_5 also functions as the ALE pin when external data memory is accessed.

Figure 6-3 shows the Type C configuration.



Secondary Function Control
C: Secondary Function Control Signal
I: Secondary Function Input
O: Secondary Function Output Data

Figure 6-3 Type C Configuration

6.1.4 Type D (P4_1-P4_7, P9_4-P9_7, P10_0-P10_7) Configuration

Type D ports do not have a secondary function. They have programmable pull-ups.

Figure 6-4 shows the Type D configuration.

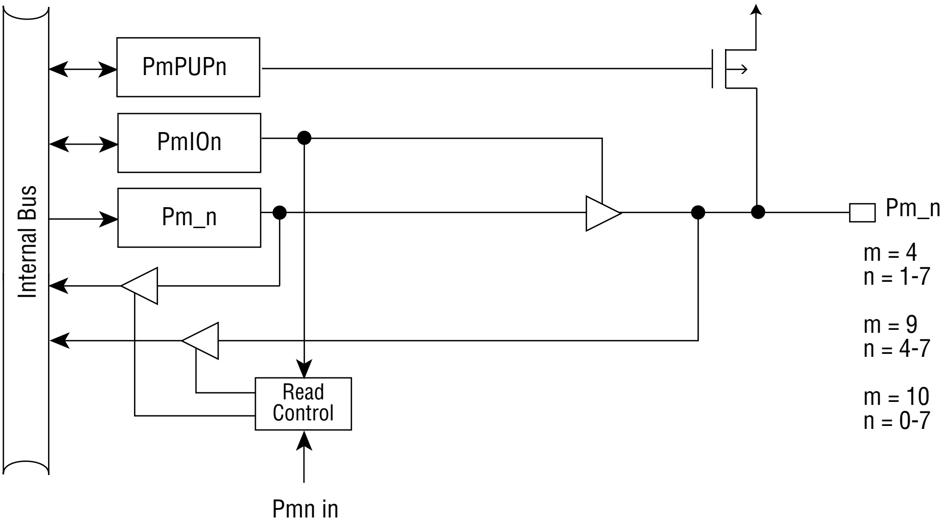


Figure 6-4 Type D Configuration

6.1.5 Type E (P2_0-P2_3, P2_6, P2_7, P7_5-P7_7, P8_0-P8_7, P12_0, P12_1, P12_4-P12_7) Configuration

Type E ports do not have a secondary function. They function as I/O pins.

Figure 6-5 shows the Type E configuration.

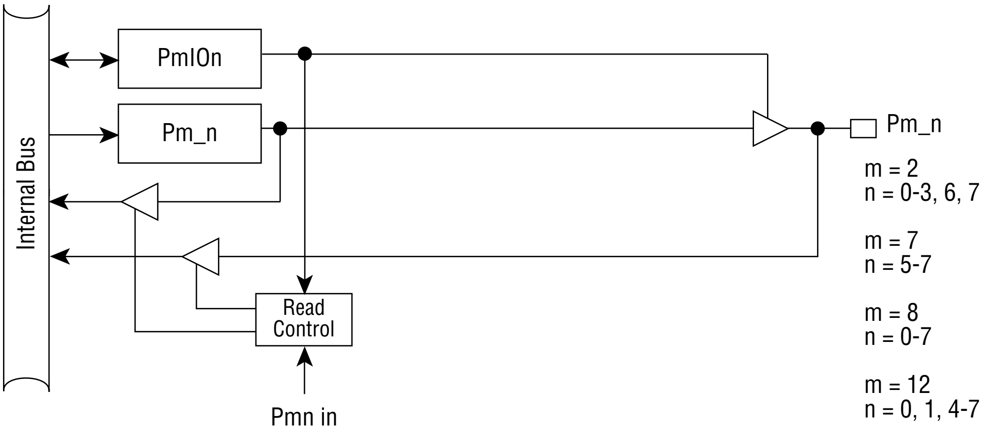


Figure 6-5 Type E Configuration

6.2 Port Control Registers

There are four types of port control registers.

- Port Data Registers (Pn: n = 0-2, 4-10, 12)
- Port Mode Registers (PnIO: n = 0-2, 4-10, 12)
- Port Secondary Control Registers (PnSF: n = 1, 2, 4-7, 9, 12)
- Port Pull-Up Control Registers (PnPUP: n = 0, 1, 4, 9, 10)

These registers are allocated as SFRs. Table 6-2 lists the port control registers.

6.2.1 Port Data Registers (Pn: n = 0-2, 4-10, 12)

Port Data Registers (Pn: n = 0-2, 4-10, 12) are 8-bit registers that store port output data. However, Port 5 has a 2-bit register, and Port 6 has a 6-bit register.

Pn are allocated as SFRs. After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) the contents of P0-P2, P4, P7-P10, and P12 will be 00H, the contents of P5, and P6 will be undefind.

When a read instruction is executed for a port, it will read the pin state ("0" or "1") if it has been specified as an input port, or it will read the contents of Pn ("0" or "1") if it has been specified as an output port. When a write instruction is executed for a port, data will be written to Pn regardless of the input/output specification.

6.2.2 Port Mode Registers (PnIO: n = 0-2, 4-10, 12)

Port Mode Registers (PnIO: n = 0-2, 4-10, 12) are 8-bit registers that specify whether ports are input or output. However, Port 5 has a 2-bit register, and Port 6 has a 6-bit register.

PnIO are allocated as SFRs. After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) the contents of P0IO-P2IO, P4IO, P7IO-P10IO, and P12IO will be 00H, and the contents of P5IO, and P6IO will be undefind.

When a PnIO bit is 0, the port will enter input mode. When 1, the port will enter output mode. However, if the Port Secondary Function Control Register (PnSF: n = 0-2 , 4-7, 9, 12) explained next has been set to 1 to select secondary function, then the PnIO specification will be disabled.

6.2.3 Port Secondary Function Control Registers (PnSF: n = 1, 2, 4-7, 9, 12)

Port Secondary Function Control Registers (PnSF: n = 1, 2, 4-7, 9, 12) are 8-bit registers that specify whether ports operate as their primary or secondary function. However, Port 2 has a 2-bit register, Port 5 has a 2-bit register, Port 6 has a 6-bit register, Port 7 has a 5-bit register, Port 9 has a 4-bit register, and Port 12 has a 2-bit register.

PnSF are allocated as SFRs. After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) the contents of P1SF, P4SF, P6SF, P7SF, P9SF, and P12SF will be 00H, the contents of P2SF will be 80H, and the contents of P5SF will be CFH. This selects the primary function for all ports.

When a PnSF bit is "0", the port's primary function will be selected. When "1", the port's secondary function will be selected. For Port 0 the primary/secondary function select is performed automatically by hardware, so it has not Port Secondary Function Control Register.

6.2.4 Port Pull-Up Control Registers (PnPUP: n = 0, 1, 4, 9, 10)

Port Pull-Up Control Registers (PnPUP: n = 0, 1, 4, 9, 10) are 8-bit registers that specify whether port pull-up resistors are enabled or disabled.

PnPUP are allocated as SFRs. After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) the contents of PnPUP registers will be 00H, disabling the pull-up resistors of all ports.

When a PnPUP bit is 0, the pull-up resistor is disabled. When 1, the pull-up resistor is enabled.

Table 6-2 Port Contorol SFR List

Address [H]	Name	Symbol (Byte)	Symbol (Word)	R/W	Reset State
0010	Port 0 Data Register	P0	—	R/W	00
0011	Port 1 Data Register	P1	—		00
0012	Port 2 Data Register	P2	—		00
0014	Port 4 Data Register	P4	—		00
0015☆	Port 5 Data Register	P5	—		undefind
0016☆	Port 6 Data Register	P6	—		undefind
0017	Port 7 Data Register	P7	—		00
0018	Port 0 Mode Register	P0IO	—		00
0019	Port 1 Mode Register	P1IO	—		00
001A	Port 2 Mode Register	P2IO	—		00
001C	Port 4 Mode Register	P4IO	—		00
001D☆	Port 5 Mode Register	P5IO	—		undefind
001E☆	Port 6 Mode Register	P6IO	—		undefind
001F	Port 7 Mode Register	P7IO	—		00
0020	Port 8 Data Register	P8	—		00
0021	Port 9 Data Register	P9	—		00
0022	Port 10 Data Register	P10	—		00
0028	Port 8 Mode Register	P8IO	—		00
0029	Port 9 Mode Register	P9IO	—		00
002A	Port 10 Mode Register	P10IO	—		00
0031	Port 1 Secondary Function Control Register	P1SF	—		00
0032☆	Port 2 Secondary Function Control Register	P2SF	—		80
0034☆	Port 4 Secondary Function Control Register	P4SF	—		00
0035☆	Port 5 Secondary Function Control Register	P5SF	—		CF
0036☆	Port 6 Secondary Function Control Register	P6SF	—		00
0037☆	Port 7 Secondary Function Control Register	P7SF	—		00
0039☆	Port 9 Secondary Function Control Register	P9SF	—		00
0120	Port 12 Data Register	P12	—		00
0122	Port 12 Mode Register	P12IO	—		00
0124☆	Port 12 Secondary Function Control Register	P12SF	—		00
0130	Port 0 Pull-Up Control Register	P0PUP	—		00
0131	Port 1 Pull-Up Control Register	P1PUP	—		00
0134	Port 4 Pull-Up Control Register	P4PUP	—		00
0139	Port 9 Pull-Up Control Register	P9PUP	—		00
013A	Port 10 Pull-Up Control Register	P10PUP	—		00

Notes: 1. Addresses do not run consecutively.
2. Addresses in the address column marked by stars indicate that the register has bits missing.

6.3 Port 0 (P0)

P0 (P0_0-P0_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 0 Mode Register (P0IO). Pull-up resistors can be individually enabled and disabled for each bit by the Port 0 Pull-Up Control Register (P0PUP).

When external memory is accessed, P0 automatically operates as its secondary function of address/data bus (low address output pins and data i/o pins) instead of as a port.

Figure 6-6 shows the configuration of the Port 0 Data Register (P0), Port 0 Mode Register (P0IO), and Port 0 Pull-Up Control Register (P0PUP).

	7	6	5	4	3	2	1	0
P0	P0_7	P0_6	P0_5	P0_4	P0_3	P0_2	P0_1	P0_0
Reset state	0	0	0	0	0	0	0	0

Address : 0010H
R/W access : R/W

	7	6	5	4	3	2	1	0
P0IO	P0IO7	P0IO6	P0IO5	P0IO4	P0IO3	P0IO2	P0IO1	P0IO0
Reset state	0	0	0	0	0	0	0	0

Address : 0018H
R/W access : R/W

0	P0_n = input
1	P0_n = output

n = 0-7

	7	6	5	4	3	2	1	0
P0PUP	P0PUP7	P0PUP6	P0PUP5	P0PUP4	P0PUP3	P0PUP2	P0PUP1	P0PUP0
Reset state	0	0	0	0	0	0	0	0

Address : 0130H
R/W access : R/W

0	P0PUP_n = pull-up disabled
1	P0PUP_n = pull-up enabled

n = 0-7

Figure 6-6 Configuration of P0, P0IO, P0PUP

When P0 has been specified as an input by P0IO (P0IOn = 0) and a read instruction is executed on it, the pin state will be read. When P0 has been specified as an output by P0IO (P0IOn = 1) and a read instruction is executed on it, the contents of the Port 0 Data Register will be read. When calculation, increment, and other read-modify-write instructions are executed on P0, the read will be of the pin state or Port 0 Data Register contents as specified by P0IO, while the write will be to the Port 0 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P0 will be a high-impedance input port (P0IO = 00H), and the contents of P0 will be 00H.

6.4 Port 1 (P1)

P1 (P1_0-P1_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 1 Mode Register (P1IO). Pull-up resistors can be individually enabled and disabled for each bit by the Port 1 Pull-Up Control Register (P1PUP).

In addition to its port function, P1 has been assigned a secondary function (external memory high-address output). The Port 1 Secondary Control Register (P1SF) selects whether the port function or secondary function will be enabled. However, when the $\overline{EA}$ pin is low, P1 will operate as the address bus (high address output pins) to external program memory and external data memory, regardless of the P1SF settings.

Figure 6-7 shows the configuration of the Port 1 Data Register (P1), Port 1 Mode Register (P1IO), Port 1 Secondary Function Control Register (P1SF), and Port 1 Pull-Up Control Register (P1PUP).

	7	6	5	4	3	2	1	0	
P1	P1_7	P1_6	P1_5	P1_4	P1_3	P1_2	P1_1	P1_0	Address : 0011H
Reset state	0	0	0	0	0	0	0	0	R/W access : R/W

	7	6	5	4	3	2	1	0	
P1IO	P1IO7	P1IO6	P1IO5	P1IO4	P1IO3	P1IO2	P1IO1	P1IO0	Address : 0019H
Reset state	0	0	0	0	0	0	0	0	R/W access : R/W

0	P1_n = input
1	P1_n = output

n = 0-7

	7	6	5	4	3	2	1	0	
P1SF	XDM15	XDM14	XDM13	XDM12	XDM11	XDM10	XDM9	XDM8	Address : 0031H
Reset state	0	0	0	0	0	0	0	0	R/W access : R/W

0	P1_n = port function
1	P1_n = XDM8-15 (data memory high address) output function

n = 0-7

	7	6	5	4	3	2	1	0	
P1PUP	P1PUP7	P1PUP6	P1PUP5	P1PUP4	P1PUP3	P1PUP2	P1PUP1	P1PUP0	Address : 0131H
Reset state	0	0	0	0	0	0	0	0	R/W access : R/W

0	P1PUP_n = pull-up disabled
1	P1PUP_n = pull-up enabled

n = 0-7

Figure 6-7 Configuration of P1, P1IO, P1SF, P1PUP

When P1 has been specified as an input by P1IO (P1IOn = 0) and a read instruction is executed on it, the pin state will be read. When P1 has been specified as an output by P1IO (P1IOn = 1) and a read instruction is executed on it, the contents of the Port 1 Data Register will be read. When calculation, increment, and other read-modify-write instructions are executed on P1, the read will be of the pin state or Port 1 Data Register contents as specified by P1IO, while the write will be to the Port 1 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P1 will be a high-impedance input port (P1IO,P1SF = 00H), and the contents of P1 will be 00H.

6.5 Port 2 (P2)

P2 (P2_0-P2_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 2 Mode Register (P2IO).

In addition to its port function, P2 has been assigned a secondary function (realtime output pins). The Port 2 Secondary Control Register (P2SF) selects whether the port function or secondary function will be enabled.

Figure 6-8 shows the configuration of the Port 2 Data Register (P2), Port 2 Mode Register (P2IO), and Port 2 Secondary Function Control Register (P2SF).

	7	6	5	4	3	2	1	0
P2	P2_7	P2_6	P2_5	P2_4	P2_3	P2_2	P2_1	P2_0
Reset state	0	0	0	0	0	0	0	0

Address : 0012H
R/W access : R/W

	7	6	5	4	3	2	1	0
P2IO	P2IO7	P2IO6	P2IO5	P2IO4	P2IO3	P2IO2	P2IO1	P2IO0
Reset state	0	0	0	0	0	0	0	0

Address : 001AH
R/W access : R/W

0	P2_n = input
1	P2_n = output

n = 0-7

	7	6	5	4	3	2	1	0
P2SF	—	"0"	RT09	RT08	"0"	"0"	"0"	"0"
Reset state	1	0	0	0	0	0	0	0

Address : 0032H
R/W access : R/W

0	P2_4
1	Realtime output 8

0	P2_5
1	Realtime output 9

— indicates a bit that is not provided. It will be read as "1".
Always write "0" to the bits shown as 0. They will always read as "0".

Figure 6-8 Configuration of P2, P2IO, P2SF

When a read instruction is executed on P2, the value read will depend on the contents of P2IO and P2SF, as shown in Table 6-3. When calculation, increment, and other read-modify-write instructions are executed on P2, the read will be of the pin state or Port 2 Data Register contents as specified by P2IO, while the write will be to the Port 2 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P2 will be a high-impedance input port (P2IO = 00H, P2SF = 80H), and the contents of P2 will be 00H.

Table 6-3 Port 2 Reads

	P2IO	**P2SF**	**Read Data**
P2_0-P2_3, P2_6, P2_7	0	—	Pin
	1	—	Output latch
P2_4, P2_5	0	0	Pin
	1	0	Output latch
	*	1	Output data

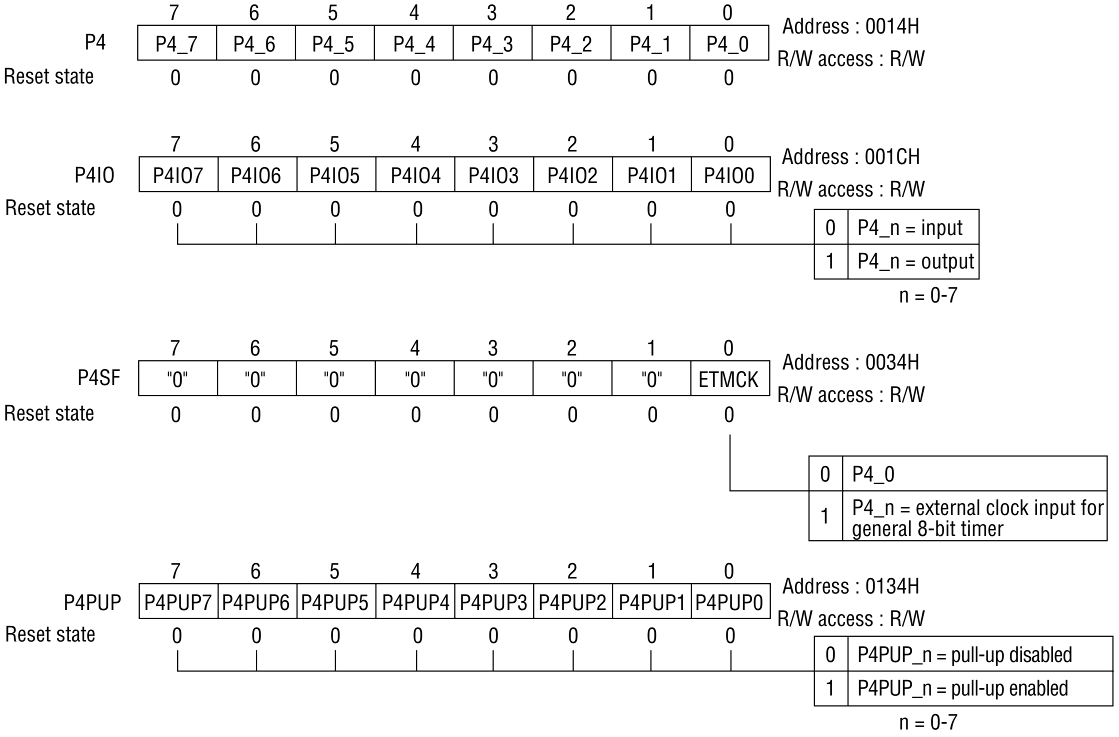
— indicates bits that are not provided.
* indicates 1 or 0.

6.6 Port 4 (P4)

P4 (P4_0-P4_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 4 Mode Register (P4IO). Pull-up resistors can be individually enabled and disabled for each bit by the Port 4 Pull-Up Control Register (P4PUP).

In addition to its port function, P4 has been assigned a secondary function (external clock input for general 8-bit timer). The Port 4 Secondary Control Register (P4SF) selects whether the port function or secondary function will be enabled.

Figure 6-9 shows the configuration of the Port 4 Data Register (P4), Port 4 Mode Register (P4IO), Port 4 Secondary Function Control Register (P4SF), and Port 4 Pull-Up Control Register (P4PUP).



	7	6	5	4	3	2	1	0	
P4	P4_7	P4_6	P4_5	P4_4	P4_3	P4_2	P4_1	P4_0	Address : 0014H R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

	7	6	5	4	3	2	1	0	
P4IO	P4IO7	P4IO6	P4IO5	P4IO4	P4IO3	P4IO2	P4IO1	P4IO0	Address : 001CH R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

0	P4_n = input
1	P4_n = output

n = 0-7

	7	6	5	4	3	2	1	0	
P4SF	"0"	"0"	"0"	"0"	"0"	"0"	"0"	ETMCK	Address : 0034H R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

0	P4_0
1	P4_n = external clock input for general 8-bit timer

	7	6	5	4	3	2	1	0	
P4PUP	P4PUP7	P4PUP6	P4PUP5	P4PUP4	P4PUP3	P4PUP2	P4PUP1	P4PUP0	Address : 0134H R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

0	P4PUP_n = pull-up disabled
1	P4PUP_n = pull-up enabled

n = 0-7

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 6-9 Configuration of P4, P4IO, P4SF, P4PUP

When a read instruction is executed on P4, the value read will depend on the contents of P4IO and P4SF, as shown in Table 6-4. When calculation, increment, and other read-modify-write instructions are executed on P4, the read will be of the pin state or Port 4 Data Register contents as specified by P4IO, while the write will be to the Port 4 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P4 will be a high-impedance input port (P4IO, P4SF = 00H), and the contents of P4 will be 00H.

Table 6-4 Port 4 Reads

	P4IO	P4SF	Read Data
P4_0	0	0	Pin
	1	0	Output latch
	*	1	Pin
P4_1-P4_7	0	—	Pin
	1	—	Output latch

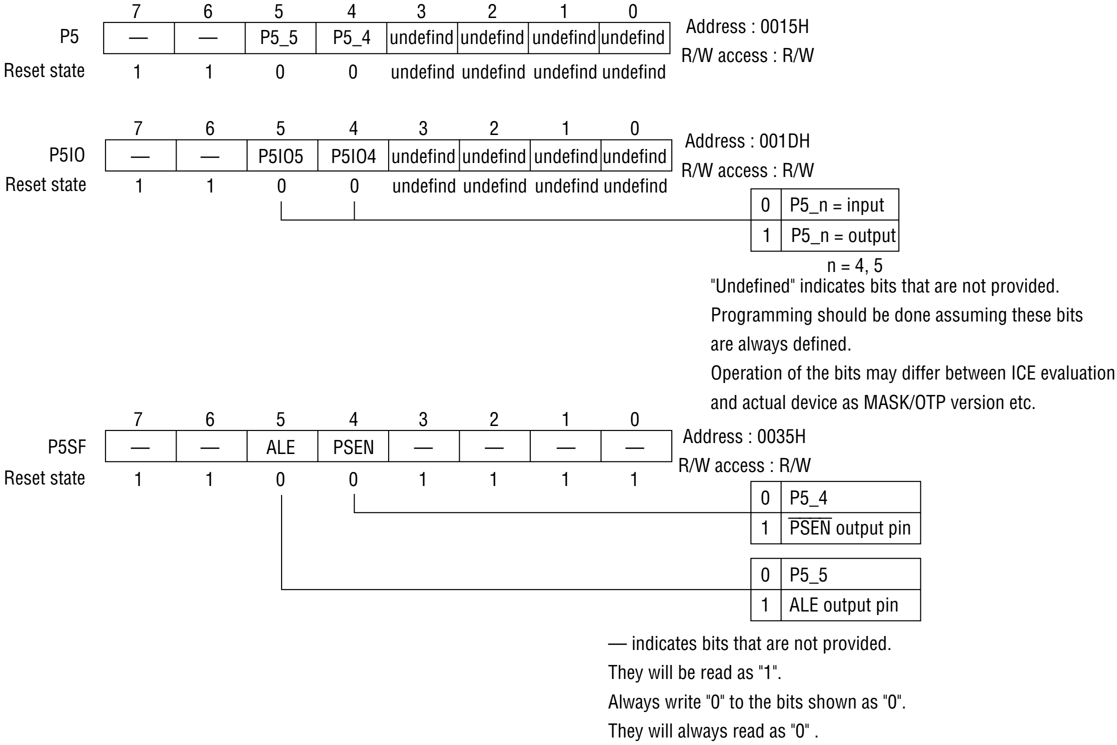
— indicates bits that are not provided.
* indicates 1 or 0.

6.7 Port 5 (P5)

P5 (P5_4, P5_5) is a 2-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 5 Mode Register (P5IO).

In addition to its port function, P5 has been assigned a secondary function (ALE and $\overline{\text{PSEN}}$ output pins). The Port 5 Secondary Control Register (P5SF) selects whether the port function or secondary function will be enabled. However, when the $\overline{\text{EA}}$ pin is low, P5_5 and P5_4 will be configured as ALE and $\overline{\text{PSEN}}$ pins regardless of P5SF settings.

Figure 6-10 shows the configuration of the Port 5 Data Register (P5), Port 5 Mode Register (P5IO), and Port 5 Secondary Function Control Register (P5SF).



	7	6	5	4	3	2	1	0
P5	—	—	P5_5	P5_4	undefind	undefind	undefind	undefind
Reset state	1	1	0	0	undefind	undefind	undefind	undefind

Address : 0015H
R/W access : R/W

	7	6	5	4	3	2	1	0
P5IO	—	—	P5IO5	P5IO4	undefind	undefind	undefind	undefind
Reset state	1	1	0	0	undefind	undefind	undefind	undefind

Address : 001DH
R/W access : R/W

0	P5_n = input
1	P5_n = output

n = 4, 5

"Undefined" indicates bits that are not provided.
Programming should be done assuming these bits are always defined.
Operation of the bits may differ between ICE evaluation and actual device as MASK/OTP version etc.

	7	6	5	4	3	2	1	0
P5SF	—	—	ALE	PSEN	—	—	—	—
Reset state	1	1	0	0	1	1	1	1

Address : 0035H
R/W access : R/W

0	P5_4
1	$\overline{\text{PSEN}}$ output pin

0	P5_5
1	ALE output pin

— indicates bits that are not provided.
They will be read as "1".
Always write "0" to the bits shown as "0".
They will always read as "0" .

Figure 6-10 Configuration of P5, P5IO, P5SF

When P5 has been specified as an input by P5IO (P5IOn = "0") and a read instruction is executed on it, the pin state will be read. When P5 has been specified as an output by P5IO (P5IOn = "1") and a read instruction is executed on it, the contents of the Port 5 Data Register will be read. When calculation, increment, and other read-modify-write instructions are executed on P5, the read will be of the pin state or Port 5 Data Register contents as specified by P5IO, while the write will be to the Port 5 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P5 will be a high-impedance input port (P5IO = undefined, P5SF = CFH), and the contents of P5 will be undefined.

6.8 Port 6 (P6)

P6 (P6_0-P6_5) is a 6-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 6 Mode Register (P6IO).

In addition to its port function, P6 has been assigned a secondary function (external interrupt pins, etc.). The Port 6 Secondary Control Register (P6SF) selects whether the port function or secondary function will be enabled.

Figure 6-11 shows the configuration of the Port 6 Data Register (P6), Port 6 Mode Register (P6IO), and Port 6 Secondary Function Control Register (P6SF).

	7	6	5	4	3	2	1	0
P6	undefind	undefind	P6_5	P6_4	P6_3	P6_2	P6_1	P6_0
Reset state	undefind	undefind	0	0	0	0	0	0

Address : 0016H
R/W access : R/W

	7	6	5	4	3	2	1	0
P6IO	undefind	undefind	P6IO5	P6IO4	P6IO3	P6IO2	P6IO1	P6IO0
Reset state	undefind	undefind	0	0	0	0	0	0

Address : 001EH
R/W access : R/W

0	P6_n = input
1	P6_n = output

n = 0-5

"Undefined" indicates bits that are not provided.
Programming should be done assuming these bits are always defined.
Operation of the bits may differ between ICE evaluation and actual device as MASK/OTP version etc.

	7	6	5	4	3	2	1	0
P6SF	"0"	"0"	TXC1	RXC1	TXD1	RXD1	INT1	INT0
Reset state	0	0	0	0	0	0	0	0

Address : 0036H
R/W access : R/W

Bit		
INT0	0	P6_0
	1	External interrupt 0 input
INT1	0	P6_1
	1	External interrupt 1 input
RXD1	0	P6_2
	1	Serial Port 1 receive data input
TXD1	0	P6_3
	1	Serial Port 1 transmit data output
RXC1	0	P6_4
	1	Serial Port 1 receive shift clock I/O
TXC1	0	P6_5
	1	Serial Port 1 transmit shift clock I/O

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 6-11 Configuration of P6, P6IO, P6SF

When a read instruction is executed on P6, the value read will depend on the contents of P6IO and P6SF, as shown in Table 6-5. When calculation, increment, and other read-modify-write instructions are executed on P6, the read will be of the pin state or Port 6 Data Register contents as specified by P6IO, while the write will be to the Port 6 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P6 will be a high-impedance input port (P6IO = undefined, P6SF = 00H), and the contents of P6 will be undefined.

Table 6-5 Port 6 Reads

	P6IO	P6SF	Read Data
P6_0-P6_2	0	0	Pin
	1	0	Output latch
	*	1	Pin
P6_3	0	0	Pin
	1	0	Output latch
	*	1	Output data
P6_4, P6_5	0	0	Pin
	1	0	Output latch
	*	1	Clock when shift clock is an output; pin when shift clock is an input.

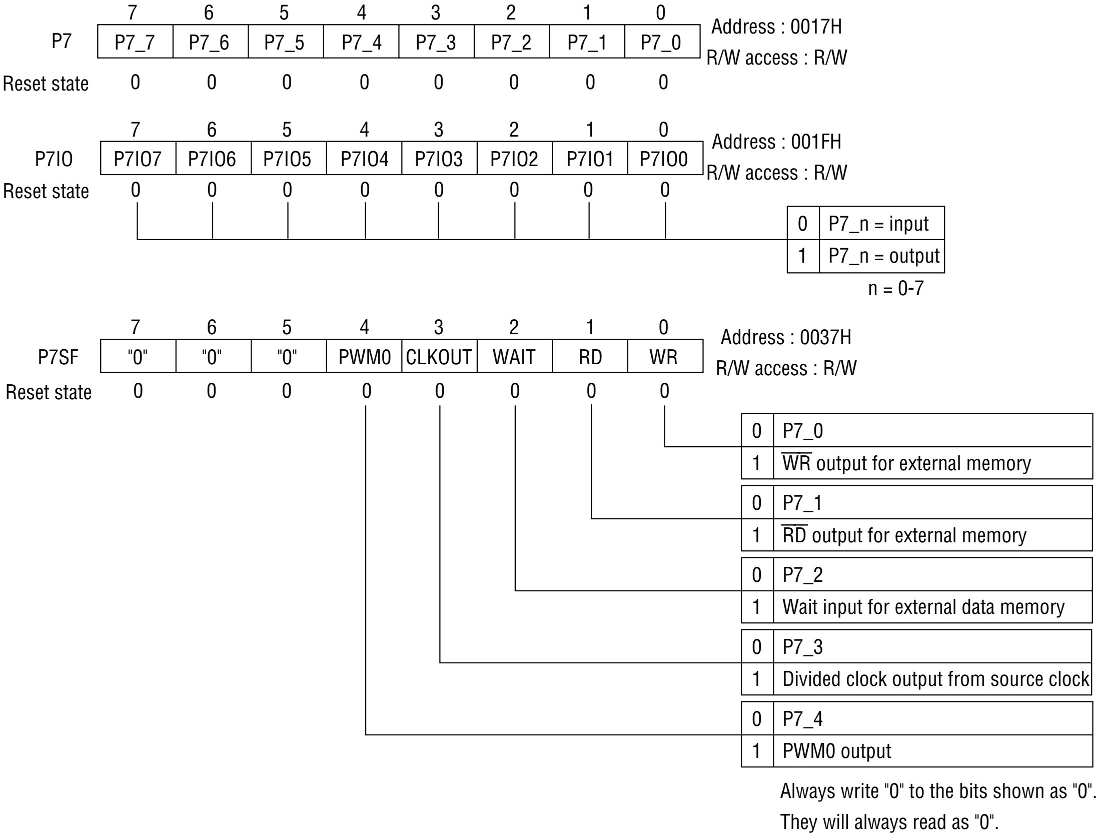
* indicates 1 or 0.

6.9 Port 7 (P7)

P7 (P7_0-P7_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 7 Mode Register (P7IO).

In addition to its port function, P7 has been assigned a secondary function (external memory strobe signal output pin, etc.). The Port 7 Secondary Control Register (P7SF) selects whether the port function or secondary function will be enabled.

Figure 6-12 shows the configuration of the Port 7 Data Register (P7), Port 7 Mode Register (P7IO), and Port 7 Secondary Function Control Register (P7SF).



	7	6	5	4	3	2	1	0	
P7	P7_7	P7_6	P7_5	P7_4	P7_3	P7_2	P7_1	P7_0	Address : 0017H R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

	7	6	5	4	3	2	1	0	
P7IO	P7IO7	P7IO6	P7IO5	P7IO4	P7IO3	P7IO2	P7IO1	P7IO0	Address : 001FH R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

0	P7_n = input
1	P7_n = output

n = 0-7

	7	6	5	4	3	2	1	0	
P7SF	"0"	"0"	"0"	PWM0	CLKOUT	WAIT	RD	WR	Address : 0037H R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

0	P7_0
1	$\overline{\text{WR}}$ output for external memory

0	P7_1
1	$\overline{\text{RD}}$ output for external memory

0	P7_2
1	Wait input for external data memory

0	P7_3
1	Divided clock output from source clock

0	P7_4
1	PWM0 output

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 6-12 Configuration of P7, P7IO, P7SF

When a read instruction is executed on P7, the value read will depend on the contents of P7IO and P7SF, as shown in Table 6-6. When calculation, increment, and other read-modify-write instructions are executed on P7, the read will be of the pin state or Port 7 Data Register contents as specified by P7IO, while the write will be to the Port 7 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P7 will be a high-impedance input port (P7IO, P7SF = 00H), and the contents of P7 will be 00H.

Table 6-6 Port 7 Reads

	P7IO	P7SF	Read Data
P7_0, P7_1	0	0	Pin
	1	0	Output latch
P7_2	0	0	Pin
	1	0	Output latch
	*	1	Pin
P7_4	0	0	Pin
	1	0	Output latch
	*	1	Output data

* indicates 1 or 0.

6.10 Port 8 (P8)

P8 (P8_0-P8_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 8 Mode Register (P8IO).

Figure 6-13 shows the configuration of the Port 8 Data Register (P8) and Port 8 Mode Register (P8IO).

	7	6	5	4	3	2	1	0	
P8	P8_7	P8_6	P8_5	P8_4	P8_3	P8_2	P8_1	P8_0	Address : 0020H R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

	7	6	5	4	3	2	1	0	
P8IO	P8IO7	P8IO6	P8IO5	P8IO4	P8IO3	P8IO2	P8IO1	P8IO0	Address : 0028H R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

0	P8_n = input
1	P8_n = output

n = 0-7

Figure 6-13 Configuration of P8, P8IO

When P8 has been specified as an input by P8IO (P8IOn = 0) and a read instruction is executed on it, the pin state will be read. When P8 has been specified as an output by P8IO (P8IOn = 1) and a read instruction is executed on it, the contents of the Port 8 Data Register will be read. When calculation, increment, and other read-modify-write instructions are executed on P8, the read will be of the pin state or Port 8 Data Register contents as specified by P8IO, while the write will be to the Port 8 Data Register.

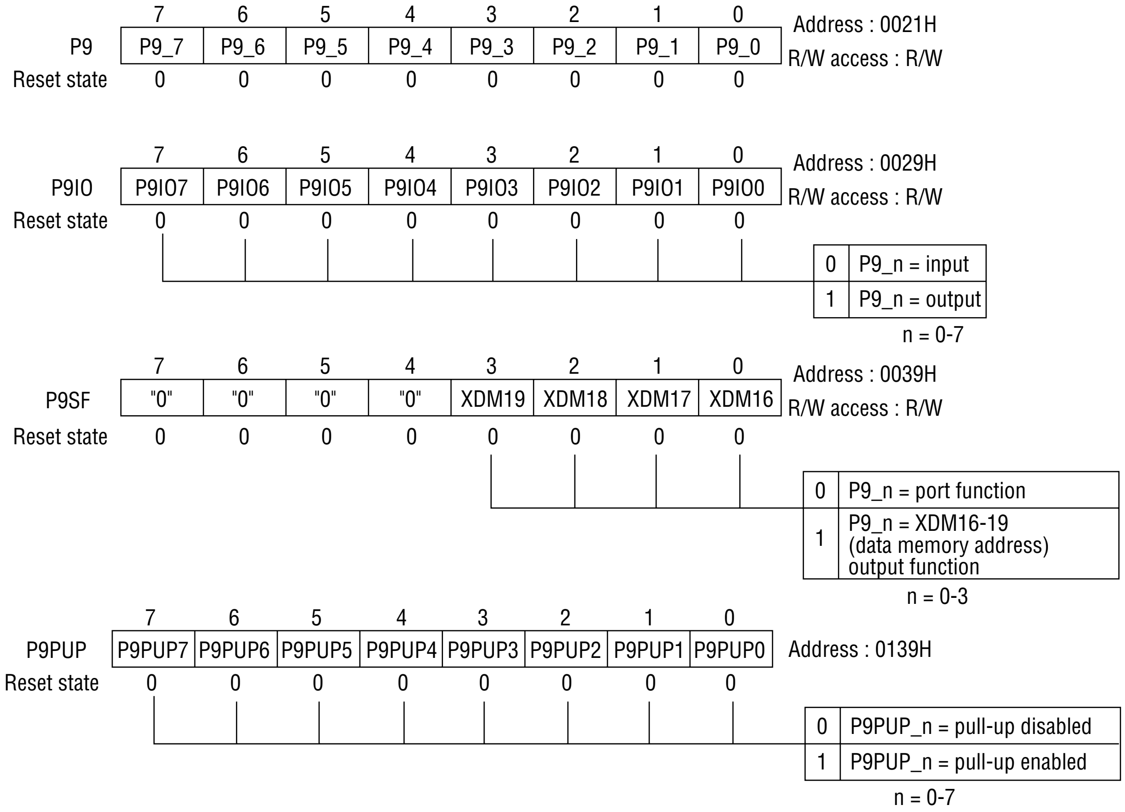
After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P8 will be a high-impedance input port (P8IO = 00H), and the contents of P8 will be 00H.

6.11 Port 9 (P9)

P9 (P9_0-P9_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 9 Mode Register (P9IO). Pull-up resistors can be individually enabled and disabled for each bit by the Port 9 Pull-Up Control Register (P9PUP).

In addition to its port function, P9 has been assigned a secondary function (external program memory address output). The Port 9 Secondary Control Register (P9SF) selects whether the port function or secondary function will be enabled. However, when the $\overline{EA}$ pin is low, P9 will operate as the program memory address bus (address output pins), regardless of the P9SF settings.

Figure 6-14 shows the configuration of the Port 9 Data Register (P9), Port 9 Mode Register (P9IO), Port 9 Secondary Function Control Register (P9SF), and Port 9 Pull-Up Control Register (P9PUP).



	7	6	5	4	3	2	1	0
P9	P9_7	P9_6	P9_5	P9_4	P9_3	P9_2	P9_1	P9_0
Reset state	0	0	0	0	0	0	0	0

Address : 0021H
R/W access : R/W

	7	6	5	4	3	2	1	0
P9IO	P9IO7	P9IO6	P9IO5	P9IO4	P9IO3	P9IO2	P9IO1	P9IO0
Reset state	0	0	0	0	0	0	0	0

Address : 0029H
R/W access : R/W

0	P9_n = input
1	P9_n = output

n = 0-7

	7	6	5	4	3	2	1	0
P9SF	"0"	"0"	"0"	"0"	XDM19	XDM18	XDM17	XDM16
Reset state	0	0	0	0	0	0	0	0

Address : 0039H
R/W access : R/W

0	P9_n = port function
1	P9_n = XDM16-19 (data memory address) output function

n = 0-3

	7	6	5	4	3	2	1	0
P9PUP	P9PUP7	P9PUP6	P9PUP5	P9PUP4	P9PUP3	P9PUP2	P9PUP1	P9PUP0
Reset state	0	0	0	0	0	0	0	0

Address : 0139H

0	P9PUP_n = pull-up disabled
1	P9PUP_n = pull-up enabled

n = 0-7

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 6-14 Configuration of P9, P9IO, P9SF, P9PUP

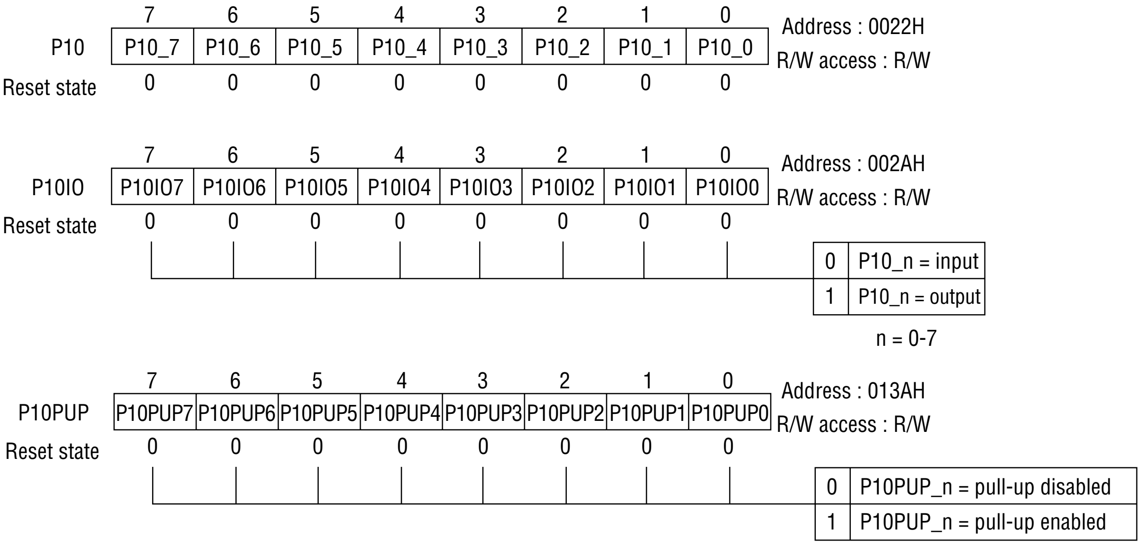
When P9 has been specified as an input by P9IO (P9IOn = 0) and a read instruction is executed on it, the pin state will be read. When P9 has been specified as an output by P9IO (P9IOn = 1) and a read instruction is executed on it, the contents of the Port 9 Data Register will be read. When calculation, increment, and other read-modify-write instructions are executed on P9, the read will be of the pin state or Port 9 Data Register contents as specified by P9IO, while the write will be to the Port 9 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P9 will be a high-impedance input port (P9IO, P9SF = 00H), and the contents of P9 will be 00H.

6.12 Port 10 (P10)

P10 (P10_0-P10_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 10 Mode Register (P10IO). Pull-up resistors can be individually enabled and disabled for each bit by the Port 10 Pull-Up Control Register (P10PUP).

Figure 6-15 shows the configuration of the Port 10 Data Register (P10), Port 10 Mode Register (P10IO), and Port 10 Pull-Up Control Register (P10PUP).



	7	6	5	4	3	2	1	0	
P10	P10_7	P10_6	P10_5	P10_4	P10_3	P10_2	P10_1	P10_0	Address : 0022H R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

	7	6	5	4	3	2	1	0	
P10IO	P10IO7	P10IO6	P10IO5	P10IO4	P10IO3	P10IO2	P10IO1	P10IO0	Address : 002AH R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

0	P10_n = input
1	P10_n = output

n = 0-7

	7	6	5	4	3	2	1	0	
P10PUP	P10PUP7	P10PUP6	P10PUP5	P10PUP4	P10PUP3	P10PUP2	P10PUP1	P10PUP0	Address : 013AH R/W access : R/W
Reset state	0	0	0	0	0	0	0	0	

0	P10PUP_n = pull-up disabled
1	P10PUP_n = pull-up enabled

Figure 6-15 Configuration of P10, P10IO, P10PUP

When P10 has been specified as an input by P10IO (P10IOn = 0) and a read instruction is executed on it, the pin state will be read. When P10 has been specified as an output by P10IO (P10IOn = 1) and a read instruction is executed on it, the contents of the Port 10 Data Register will be read. When calculation, increment, and other read-modify-write instructions are executed on P10, the read will be of the pin state or Port 10 Data Register contents as specified by P10IO, while the write will be to the Port 10 Data Register.

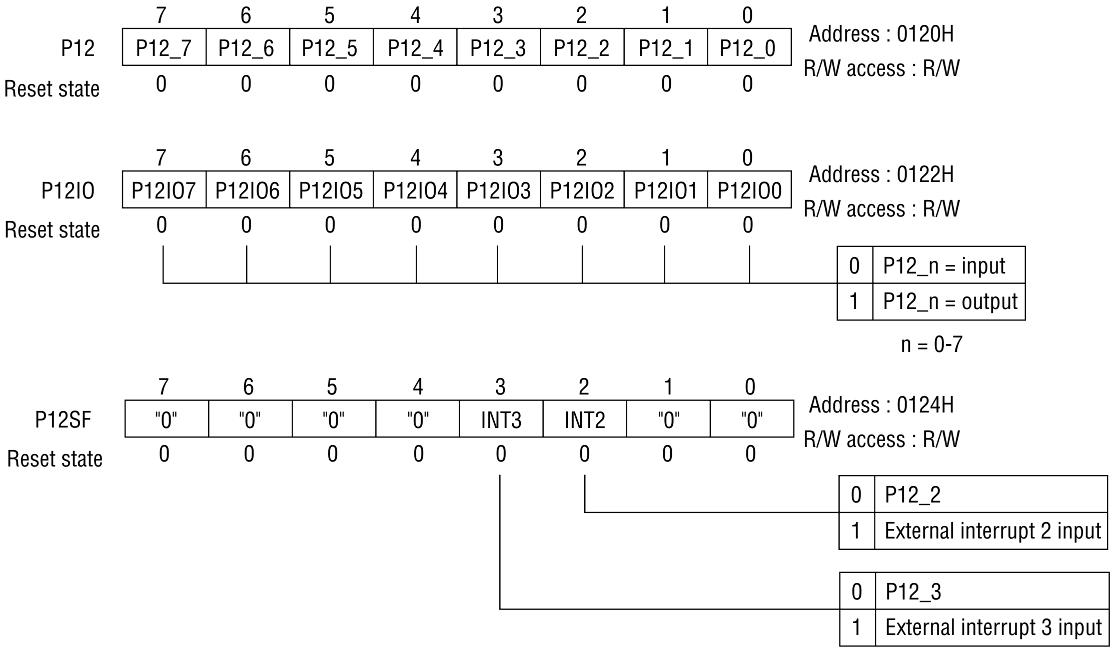
After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P10 will be a high-impedance input port (P10IO = 00H), and the contents of P10 will be 00H.

6.13 Port 12 (P12)

P12 (P12_0-P12_7) is an 8-bit input/output port. Its bits can be individually specified as inputs or outputs by the Port 12 Mode Register (P12IO).

In addition to its port function, P12 has been assigned a secondary function (external interrupt pins, etc.). The Port 12 Secondary Control Register (P12SF) selects whether the port function or secondary function will be enabled.

Figure 6-16 shows the configuration of the Port 12 Data Register (P12), Port 12 Mode Register (P12IO), and Port 12 Secondary Function Control Register (P12SF).



	7	6	5	4	3	2	1	0
P12	P12_7	P12_6	P12_5	P12_4	P12_3	P12_2	P12_1	P12_0
Reset state	0	0	0	0	0	0	0	0

Address : 0120H
R/W access : R/W

	7	6	5	4	3	2	1	0
P12IO	P12IO7	P12IO6	P12IO5	P12IO4	P12IO3	P12IO2	P12IO1	P12IO0
Reset state	0	0	0	0	0	0	0	0

Address : 0122H
R/W access : R/W

0	P12_n = input
1	P12_n = output

n = 0-7

	7	6	5	4	3	2	1	0
P12SF	"0"	"0"	"0"	"0"	INT3	INT2	"0"	"0"
Reset state	0	0	0	0	0	0	0	0

Address : 0124H
R/W access : R/W

0	P12_2
1	External interrupt 2 input

0	P12_3
1	External interrupt 3 input

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 6-16 Configuration of P12, P12IO, P12SF

When a read instruction is executed on P12, the value read will depend on the contents of P12IO and P12SF, as shown in Table 6-7. When calculation, increment, and other read-modify-write instructions are executed on P12, the read will be of the pin state or Port 12 Data Register contents as specified by P12IO, while the write will be to the Port 12 Data Register.

After reset (by $\overline{\text{RES}}$ signal input, BRK instruction execution, or op code trap), P12 will be a high-impedance input port (P12IO, P12SF = 00H), and the contents of P12 will be 00H.

Table 6-7 Port 12 Reads

	P12IO	P12SF	Read Data
P12_0, P12_1 P12_4-P12_7	0	0	Pin
	1	0	Output latch
P12_2, P12_3	0	0	Pin
	1	0	Output latch
	*	1	Pin

* indicates 1 or 0.

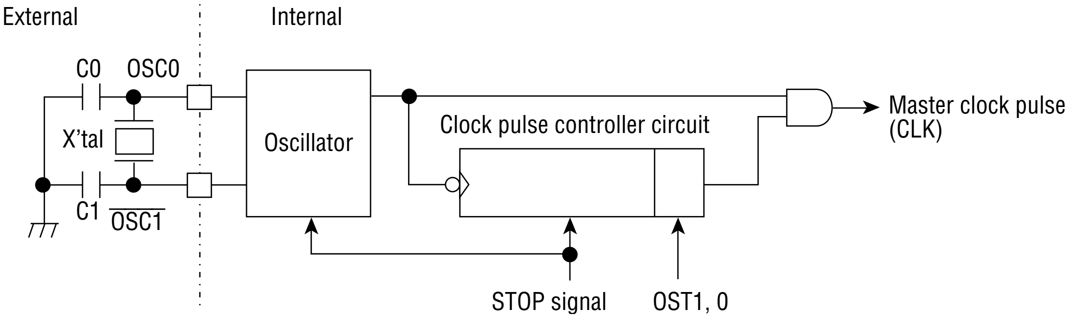
Chapter 7

Clock Generator Circuit

7. Clock Generator Circuit

The clock generator circuit is used for generating the master clock pulse (CLK). The crystal oscillator and necessary elements are connected between OSC0 pin and $\overline{\text{OSC1}}$ pin.

Figure 7-1 shows an example to connect the crystal generator circuit.



Notes: 1. The values of C0 and C1 have to be decided in accordance with the specifications of an external crystal oscillator.
2. The ceramic resonator can be used instead of the crystal.
3. Parts that are not illustrated may be required by the frequency band that is used.

Figure 7-1 Example for Connecting Crystal Oscillator Circuit

In addition, when the master clock pulse is supplied from external, input it at OSC0 pin and then open $\overline{\text{OSC1}}$ pin.

Figure 7-2 shows an example for external clock input.

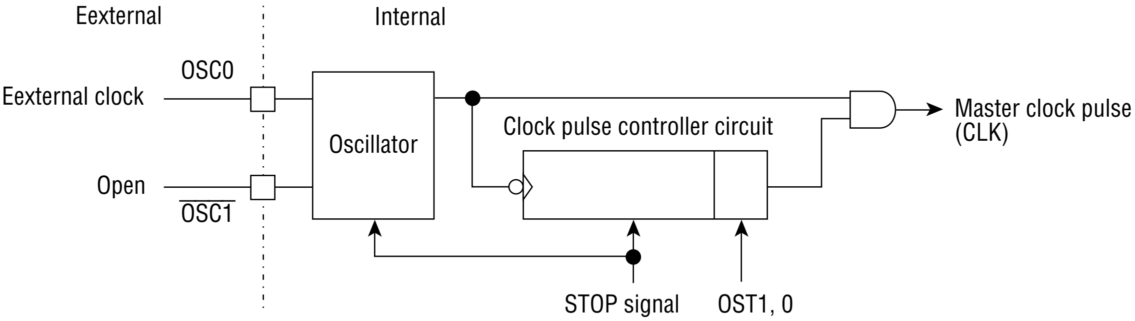


Figure 7-2 Example for External Clock Input

The clock generator circuit can be stopped by using stop mode, which can further reduce power consumption.

When stop mode is released by an interrupt, the basic clock will be passed on when the number of clock cycles specified by OST0 and OST1 of SBYCON (bit 4 and bit 5) elapse after the oscillator stabilizes. However, when STOP mode is released by the $\overline{\text{RES}}$ pin, the settings of OST0 and OST1 of SBYCON will be invalid, so apply a low level to the $\overline{\text{RES}}$ pin for longer than the oscillation stabilization time.

Chapter 8

Time-Base Counter (TBC)

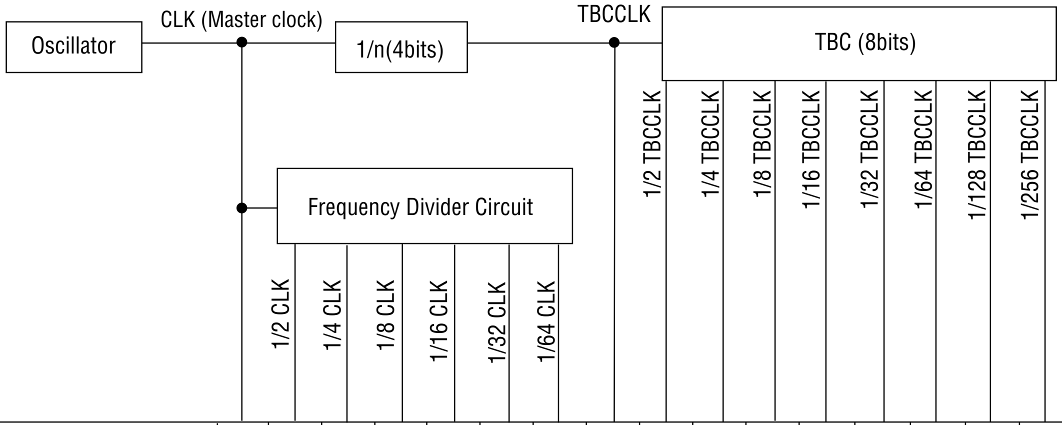
8. Time-Base Counter (TBC)

The Time-Base Counter (TBC) is an 8-bit counter that uses as its input clock the overflow of a 4-bit auto-reload timer that uses the master clock for its input.

TBC's frequency divided output is used as a basic clock for a 16-bit RTO/PWM timer or serial port timer.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) TBC will be cleared to 0, and will operate unless the master clock supply stops.

Figure 8-1 shows the configuration of TBC.



	CLK	1/2 CLK	1/4 CLK	1/8 CLK	1/16 CLK	1/32 CLK	1/64 CLK	TBCCLK	1/2 TBCCLK	1/4 TBCCLK	1/8 TBCCLK	1/16 TBCCLK	1/32 TBCCLK	1/64 TBCCLK	1/128 TBCCLK	1/256 TBCCLK
TM1								○	○	○	○	○	○	○	○	
General 8-bit timers									○	○	○	○		○		○
PWM	○	○	○	○	○	○	○									
SCI 8-bit timer	○	○							○	○	○	○		○		○

Notes: 1. When the 1/n (4-bit) counter is set with n = 1 and TBCCLK is selected for TM1, the 16-bit RTO/PWM timer does not operate normally. (The free-running counter TM1 will count normally.)
2. Refer to Chapter 11, "PWM Function," for the divider.

Figure 8-1 TBC Configuration

8.1 1/n Counter

A 4-bit auto-reload timer that uses CLK as its input clock is provided to allow the same clock pulse (TBCCLK) to be supplied to TBC even when oscillation is changed. The 1/n counter is configured from a 4-bit counter (TBCKDVC) and a 4-bit register (TBCKDVR) that stores the reload value.

	7	6	5	4	3	2	1	0
TBCKDVC	—	—	—	—				
Reset state	1	1	1	1	0	0	0	0

Address : 011EH
R/W access : R

— indicates bits that are not provided.
They will be read as 1.

TBCKDVC is a 4-bit counter (bit 0-bit 3) with CLK as its input clock. Writes to TBCKDVC are ignored. TBCKDVC can be read, but the upper 4 bits will read as 1.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) TBCKDVC will be F0H.

	7	6	5	4	3	2	1	0
TBCKDVR	—	—	—	—				
Reset state	1	1	1	1	0	0	0	0

Address : 011FH
R/W access : R/W

— indicates bits that are not provided.
They will be read as 1.

TBCKDVR is a 4-bit register (bit 0-bit 3) that stores the reload value to TBCKDVC.

TBCKDVR can be written, but the upper 4 bits will be ignored. TBCKDVR can be read, but the upper 4 bits will read as 1.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) TBCKDVR will be F0H. The 1/n counter will divide CLK by 16, so a 1/16 CLK will be supplied to TBC as TBCCLK.

After a reload value is written to TBCKDVR, it may take up to 16 CLK pulses before the frequency division by the written value affects operation.

Chapter 9

16-Bit RTO/PWM Timer

9. 16-Bit RTO/PWM Timer

The 16-bit RTO/PWM Timer is configured from a 16-bit counter, two 16-bit registers, and a control register. It provides two modes PWM/realtime output mode.

Figure 9-1 shows the 16-bit RTO/PWM timer configuration. Table 9-1 lists the 16-bit RTO/PWM timer control SFRs.

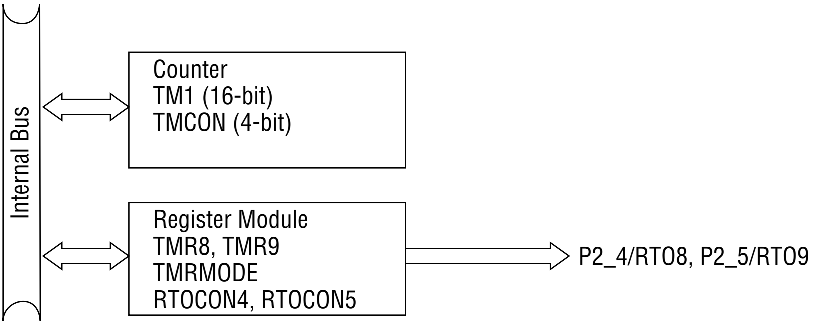


Figure 9-1 16-Bit RTO/PWM Timer Configuration

Table 9-1 16-Bit RTO/PWM Timer Control SFRs

Address [H]	Name	Symbol (Byte)	Symbol (Word)	R/W	Reset State
0090 0091	Timer Register 8	—	TMR8	R/W	0000
0092 0093	Timer Register 9	—	TMR9		0000
00B4☆	RTO Control Register 4	RTOCON4	—		FC
00B5☆	RTO Control Register 5	RTOCON5	—		FC
00C2 00C3	Timer Counter 1	—	TM1		0000
00E2☆	Timer Control Register	TMCON	—		00
00E4 00E5☆	TM Select Register	—	TMSEL		0300
00E6☆	TMR Mode Register	TMRMODE	—		00

Notes: 1. Addresses do not run consecutively.
2. Addresses in the address column marked by stars indicate that the register has bits missing.

9.1 Counter Configuration

The counter circuit is configured from a 16-bit free-running counter (TM1), an input clock selector, and a control register (TMCON) that controls TM1 operation. TM1 generates an interrupt request whenever an overflow occurs.

Figure 9-2 shows the counter configuration.

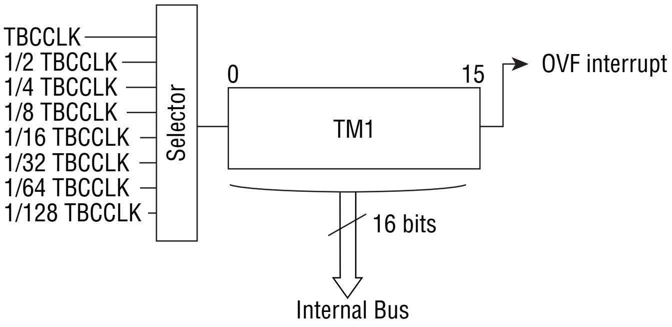


Figure 9-2 Counter Configuration

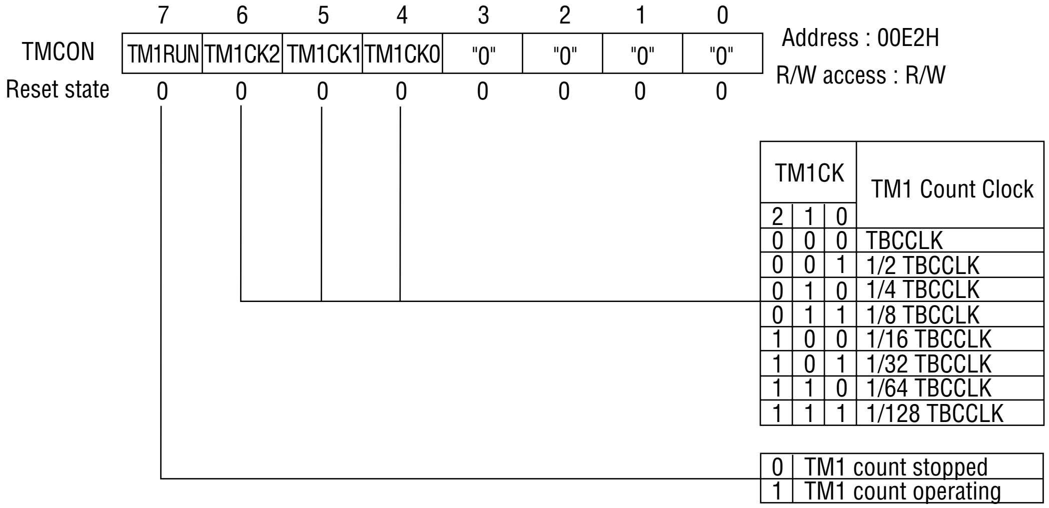
TM1 is a 16-bit counter that can be read or written by the program.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) TM1 will be 0000H, stopping operation.

The TM1 count clock is selected by the Timer Control Register (TMCON) from TBC’s frequency divided output 0-6 of the TBC count clock (TBCCLK).

TMCON is an 8-bit register that selects the TM1 count clock, and also selects whether the clock operates or stops. After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) TMCON will be 00H, selecting TBCCLK for the TM1 count clock and stopping operation.

Figure 9-3 shows the configuration of TMCON.



	7	6	5	4	3	2	1	0
TMCON	TM1RUN	TM1CK2	TM1CK1	TM1CK0	"0"	"0"	"0"	"0"
Reset state	0	0	0	0	0	0	0	0

Address : 00E2H
R/W access : R/W

TM1CK			TM1 Count Clock
2	1	0	
0	0	0	TBCCLK
0	0	1	1/2 TBCCLK
0	1	0	1/4 TBCCLK
0	1	1	1/8 TBCCLK
1	0	0	1/16 TBCCLK
1	0	1	1/32 TBCCLK
1	1	0	1/64 TBCCLK
1	1	1	1/128 TBCCLK

0	TM1 count stopped
1	TM1 count operating

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 9-3 TMCON Configuration

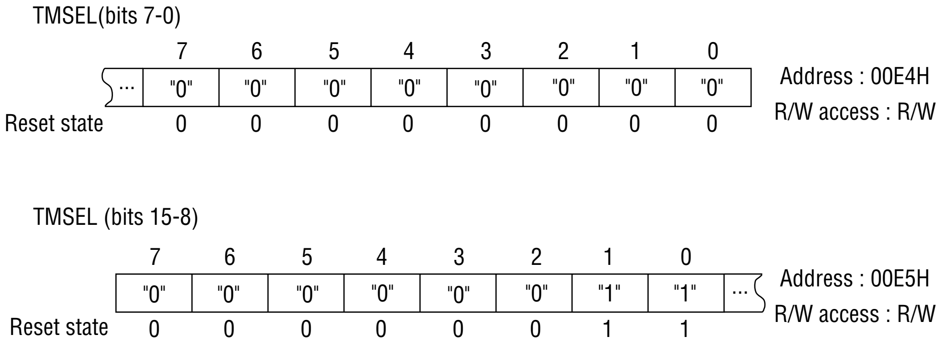
9.2 TM Select Register

The TM Select Register (TMSEL) is a 16-bit register.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) TMSEL will be 0300H. However, it will be 0000H with an ICE.

When the timer is to be used, always write 0300H after reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap).

Figure 9-4 shows the configuration of TMSEL.



TMSEL(bits 7-0)

		7	6	5	4	3	2	1	0
	···	"0"	"0"	"0"	"0"	"0"	"0"	"0"	"0"
Reset state		0	0	0	0	0	0	0	0

Address : 00E4H
R/W access : R/W

TMSEL (bits 15-8)

	7	6	5	4	3	2	1	0	
	"0"	"0"	"0"	"0"	"0"	"0"	"1"	"1"	···
Reset state	0	0	0	0	0	0	1	1	

Address : 00E5H
R/W access : R/W

Always write "0" to the bits shown as "0",
and "1" to the bits shown as "1".
Those shown as "0" will always read as "0",
and those shown as "1" will always read as "1".

Figure 9-4 TMSEL Configuration

9.3 Register Modules (TMR8, TMR9)

Two sets of register modules, TMR8 and TMR9, are provided. The two sets have the same configuration except that the SFR addresses assigned to the registers are different.

9.3.1 Register Module (TMR8, TMR9) Configuration

A register module has three types of realtime output: toggle RTO, RTO, and PWM mode. It is configured from a 16-bit timer register (TMR8, TMR9), output pins for the realtime output signal (RTO8, RTO9), timing controller, comparator to compare the timer counter value and timer register value, control register to specify realtime output operation (TMRMODE), and control register to control realtime output operation (RTOCON4, RTOCON5).

To use RTO8 and RTO9 as realtime output pins, be sure to set the corresponding bits in the Port 2 Secondary Control Register to "1".

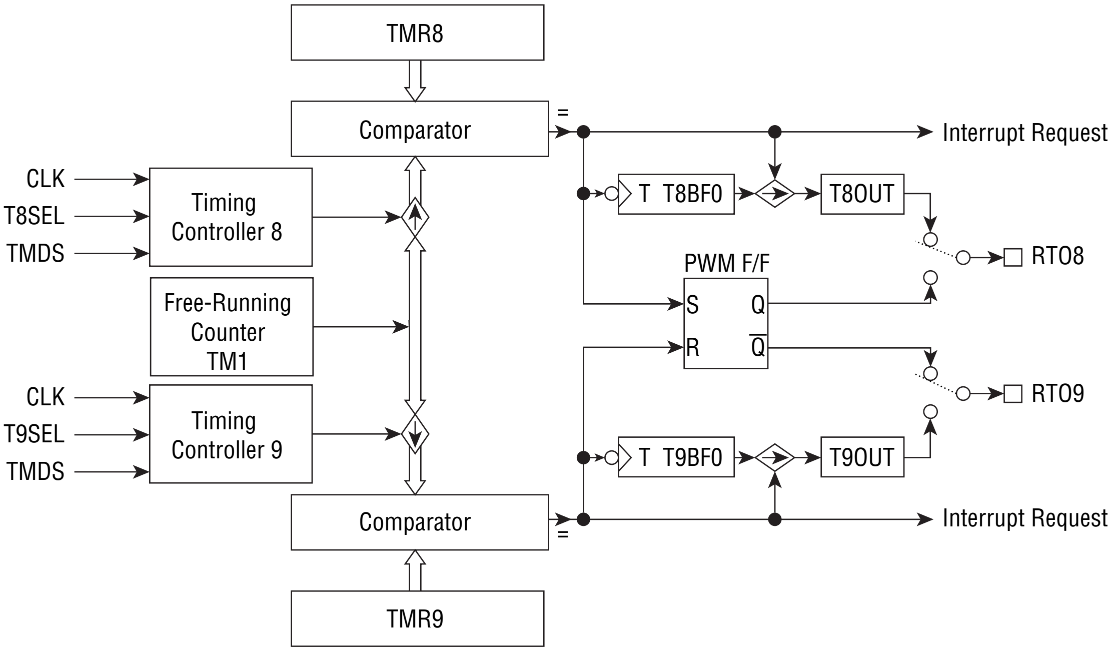


Figure 9-5 Register Module Configuration

(1) Timer Registers (TMR8, TMR9)

The timer registers (TMR8, TMR9) are 16-bit registers. TMR8 and TMR9 are always compared to the counter value of TM1. They can be read and written by the program.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) TMR8 and TMR9 will be 0000H.

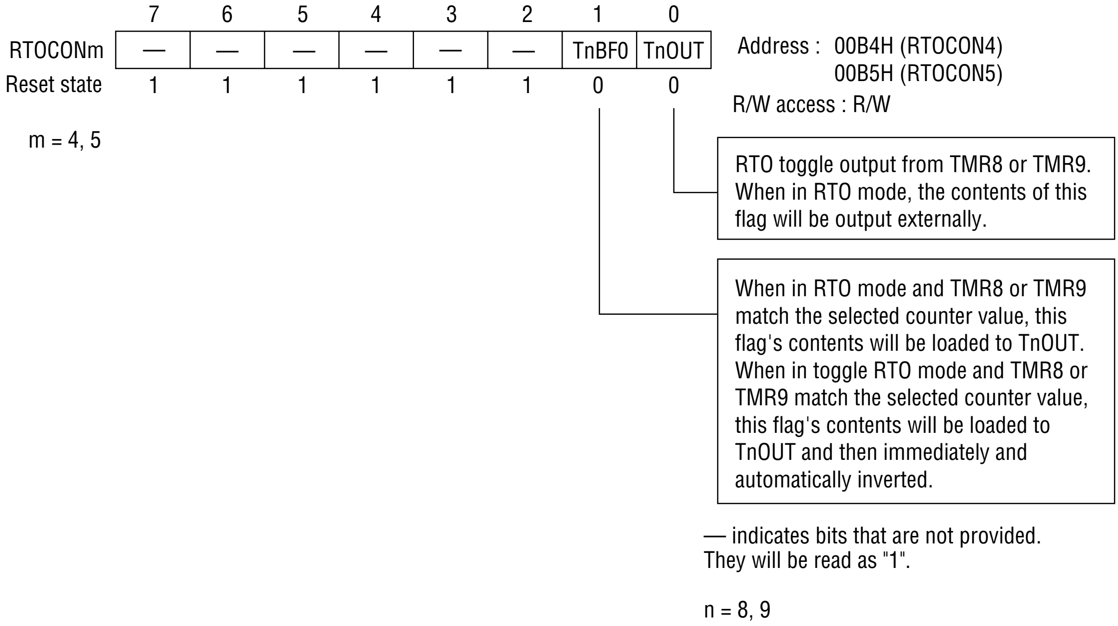
(2) Realtime Output Control Registers (RTOCON4, RTOCON5)

The realtime output control registers (RTOCON4, RTOCON5) are configured as two bits. When TMR8 or TMR9 match the counter value specified by TMSEL, the contents of TnBF0 (bit 1) will be loaded into TnOUT (bit 0). Set TnBF0 to the level desired after the next event change.

RTOCON4 and RTOCON5 can be read and written by the program. When written, the upper 6 bits will be ignored. When read, the upper 6 bits will read as "1". Note that TnBF0 and TnOUT might not operate correctly just before an RTO event when an SB, RB, XORB or other read-modify-write instruction is executed on RTOCON4 or RTOCON5.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) RTOCON4 and RTOCON5 will be FCH.

Figure 9-6 shows the configuration of RTOCON4 and RTOCON5.



	7	6	5	4	3	2	1	0
RTOCONm	—	—	—	—	—	—	TnBF0	TnOUT
Reset state	1	1	1	1	1	1	0	0

m = 4, 5

Address : 00B4H (RTOCON4)
00B5H (RTOCON5)
R/W access : R/W

RTO toggle output from TMR8 or TMR9. When in RTO mode, the contents of this flag will be output externally.

When in RTO mode and TMR8 or TMR9 match the selected counter value, this flag's contents will be loaded to TnOUT. When in toggle RTO mode and TMR8 or TMR9 match the selected counter value, this flag's contents will be loaded to TnOUT and then immediately and automatically inverted.

— indicates bits that are not provided.
They will be read as "1".

n = 8, 9

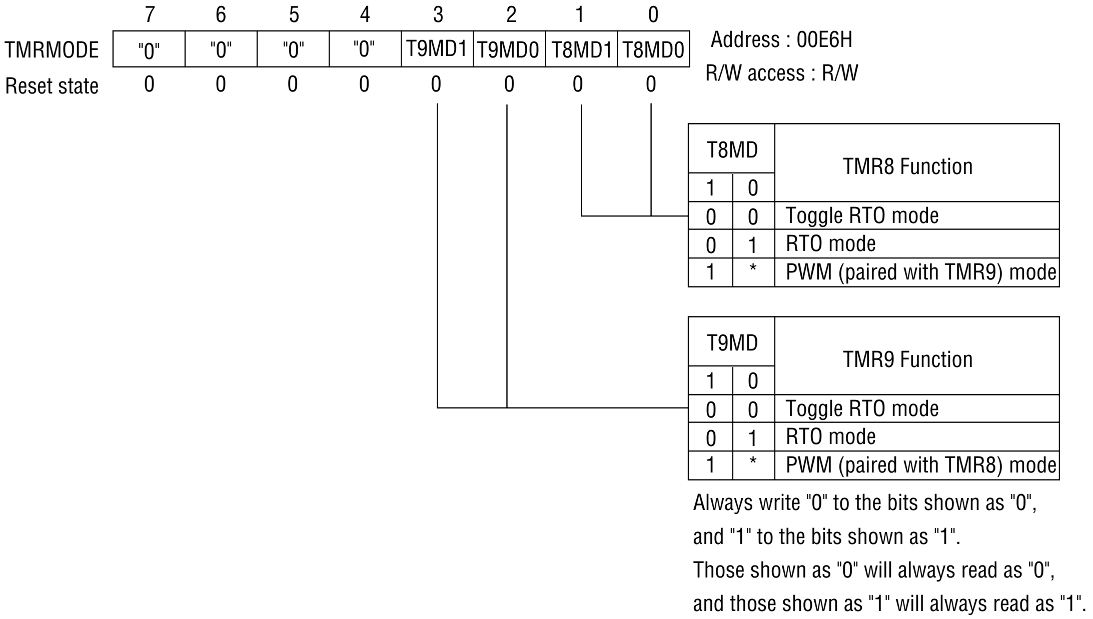
Figure 9-6 RTOCON4, RTOCON5 Configuration

(3) TMR Mode Register (TMRMODE)

The TMR Mode Register (TMRMODE) is configured as 4 bits. TMRMODE selects the operation of TMR8 and TMR9. It can be read and written by the program.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) TMRMODE will be 00H, specifying toggle RTO mode for TMR8 and TMR9.

Figure 9-7 shows the configuration of TMRMODE.



	7	6	5	4	3	2	1	0
TMRMODE	"0"	"0"	"0"	"0"	T9MD1	T9MD0	T8MD1	T8MD0
Reset state	0	0	0	0	0	0	0	0

Address : 00E6H
R/W access : R/W

T8MD		TMR8 Function
1	0	
0	0	Toggle RTO mode
0	1	RTO mode
1	*	PWM (paired with TMR9) mode

T9MD		TMR9 Function
1	0	
0	0	Toggle RTO mode
0	1	RTO mode
1	*	PWM (paired with TMR8) mode

Always write "0" to the bits shown as "0",
and "1" to the bits shown as "1".
Those shown as "0" will always read as "0",
and those shown as "1" will always read as "1".

Figure 9-7 TMRMODE Configuration

9.3.2 Register Module (TMR8, TMR9) Operation

TMRMODE can select one of three functions for a register module.

- Realtime output mode (RTO)
- Toggle realtime output mode (toggle RTO)
- PWM mode

(1) Realtime Output Mode (RTO) Operation

When the TM1 counter is running, its value will continuously be compared to TMR8 and TMR9. When the values match, a realtime output interrupt request will be generated and the contents of TnBF0 of RTOCON4 or RTOCON5 (bit 1) will be loaded in TnOUT (bit 0), where n = 8 or 9. Therefore, set TMR8 and TMR9 to the desired time interval until the next event change, and set TnFB0 to the desired level after the next event change.

Figure 9-8 shows an operation example of a register module in RTO mode.

To use RTO8 and RTO9 as realtime output pins, be sure to set the corresponding bits in the Port 2 Secondary Control Register to 1.

(2) Toggle Output Realtime Output Mode (Toggle RTO) Operation

When the TM1 counter is running, its value will continuously be compared to TMR8 and TMR9. When the values match, a realtime output interrupt request will be generated and the contents of TnBF0 of RTOCON4 or RTOCON5 (bit 1) will be loaded in TnOUT (bit 0), where n = 8 or 9. The contents of TnFB0 of RTOCON4 or RTOCON5 will then be immediately and automatically inverted.

Figure 9-9 shows an operation example of a register module in toggle RTO mode.

To use RTO8 and RTO9 as realtime output pins, be sure to set the corresponding bits in the Port 2 Secondary Control Register to 1.

(3) PWM Mode Operation

TMR8 and TMR9 can operated in PWM mode as a pair. When PWM mode is specified and the TM1 counter is running, its value will continuously be compared to TMR8. When the values match, an interrupt request will be generated, and the PWM flip-flop will be set (1). When the TM1 counter is running, its value will also continuously be compared to TMR9. When the values match, an interrupt request will be generated, and the PWM flip-flop will be reset (0). If a set and reset for the PWM flip-flop occur simultaneously, then the reset will take precedence.
The Q (true) output of the PWM flip-flop will be output from RTO8, and the $\overline{Q}$ (inverted) output will be output from RTO9.

In PWM mode, TnBF0 (bit 1) and TnOUT (bit 0) of RTOCON4 and RTOCON5 will be invalid (n = 8, 9).
Figure 9-10 shows an operation example of a register module in PWM mode.
To use RTO8 and RTO9 as PWM output pins, be sure to set the corresponding bits in the Port 2 Secondary Control Register to 1.

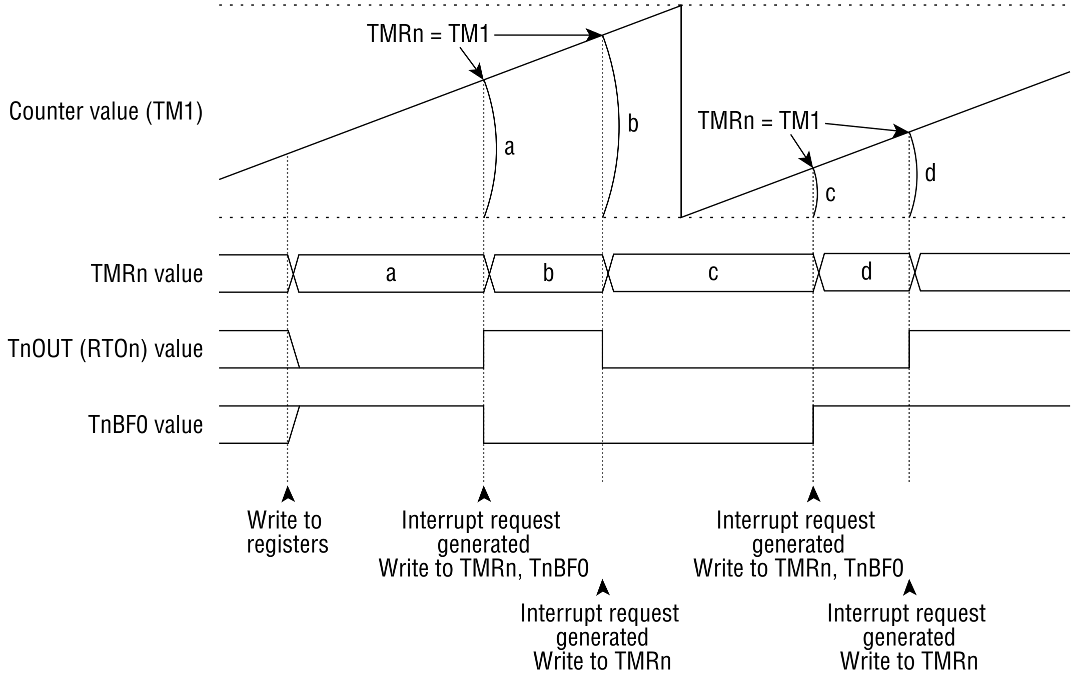

TMRn = TM1
Counter value (TM1)
a
b
TMRn = TM1
c
d
TMRn value
a
b
c
d
TnOUT (RTOn) value
TnBF0 value
Write to registers
Interrupt request generated Write to TMRn, TnBF0
Interrupt request generated Write to TMRn, TnBF0
Interrupt request generated Write to TMRn
Interrupt request generated Write to TMRn

Figure 9-8 Operation Example of Register Module in RTO Mode (n = 8, 9)

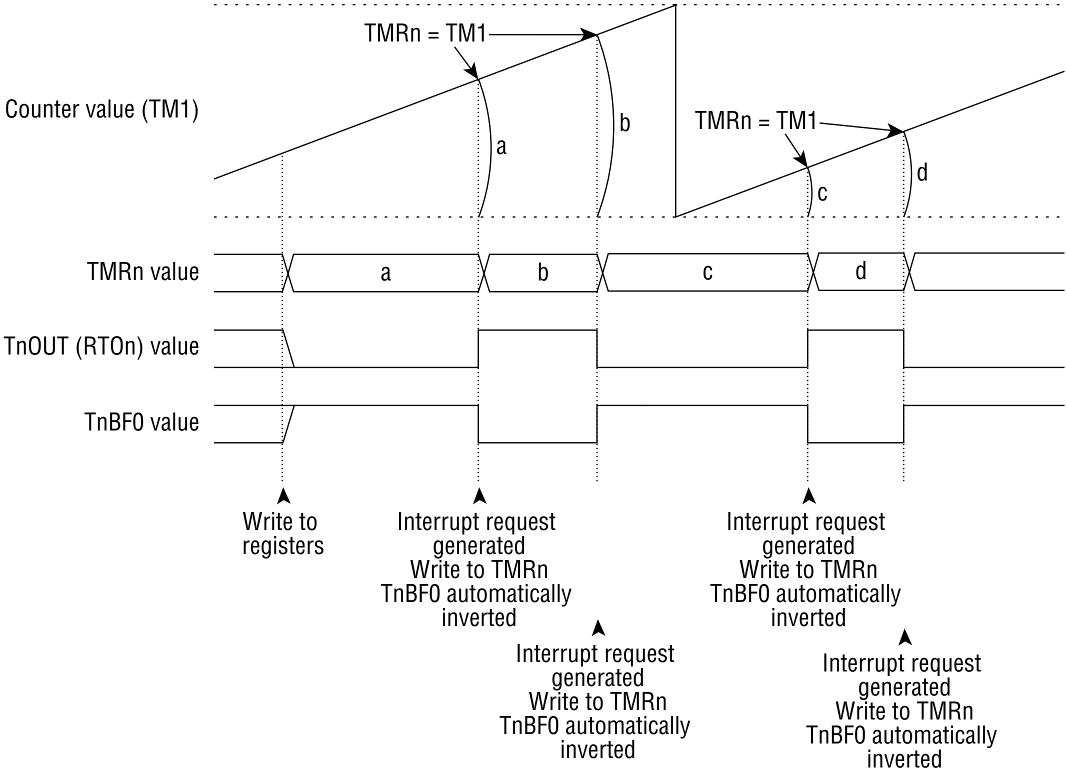

TMRn = TM1
Counter value (TM1)
a
b
TMRn = TM1
c
d
TMRn value
a
b
c
d
TnOUT (RTOn) value
TnBF0 value
Write to registers
Interrupt request generated Write to TMRn TnBF0 automatically inverted
Interrupt request generated Write to TMRn TnBF0 automatically inverted
Interrupt request generated Write to TMRn TnBF0 automatically inverted
Interrupt request generated Write to TMRn TnBF0 automatically inverted

Figure 9-9 Operation Example of Register Module in Toggle RTO Mode (n = 8, 9)

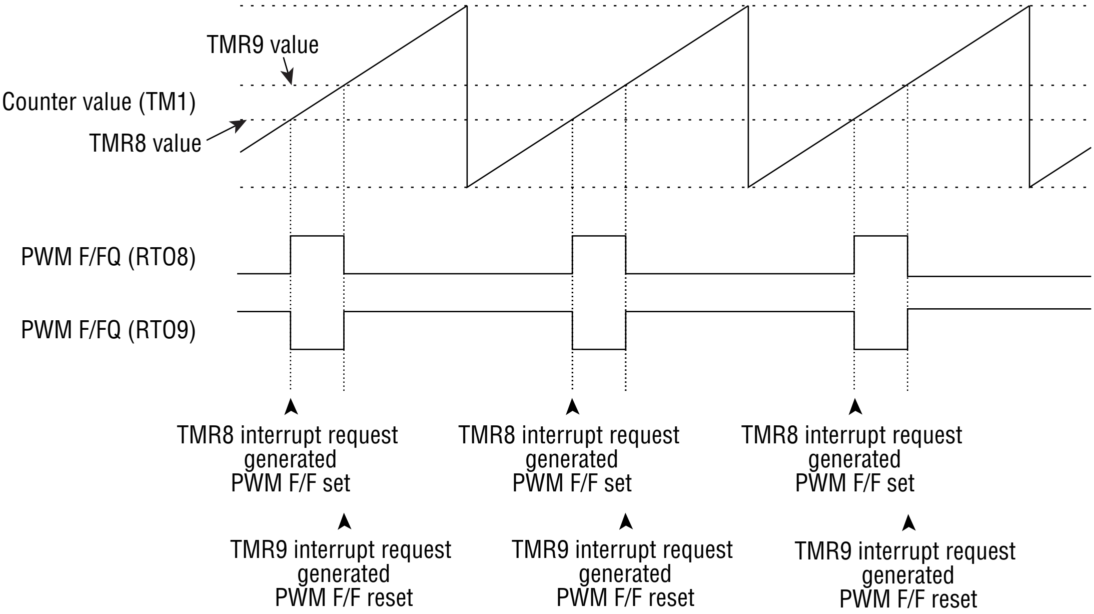

TMR9 value
Counter value (TM1)
TMR8 value
PWM F/FQ (RTO8)
PWM F/FQ (RTO9)
TMR8 interrupt request generated PWM F/F set
TMR8 interrupt request generated PWM F/F set
TMR8 interrupt request generated PWM F/F set
TMR9 interrupt request generated PWM F/F reset
TMR9 interrupt request generated PWM F/F reset
TMR9 interrupt request generated PWM F/F reset

Figure 9-10 Operation Example of Register Module in PWM Mode

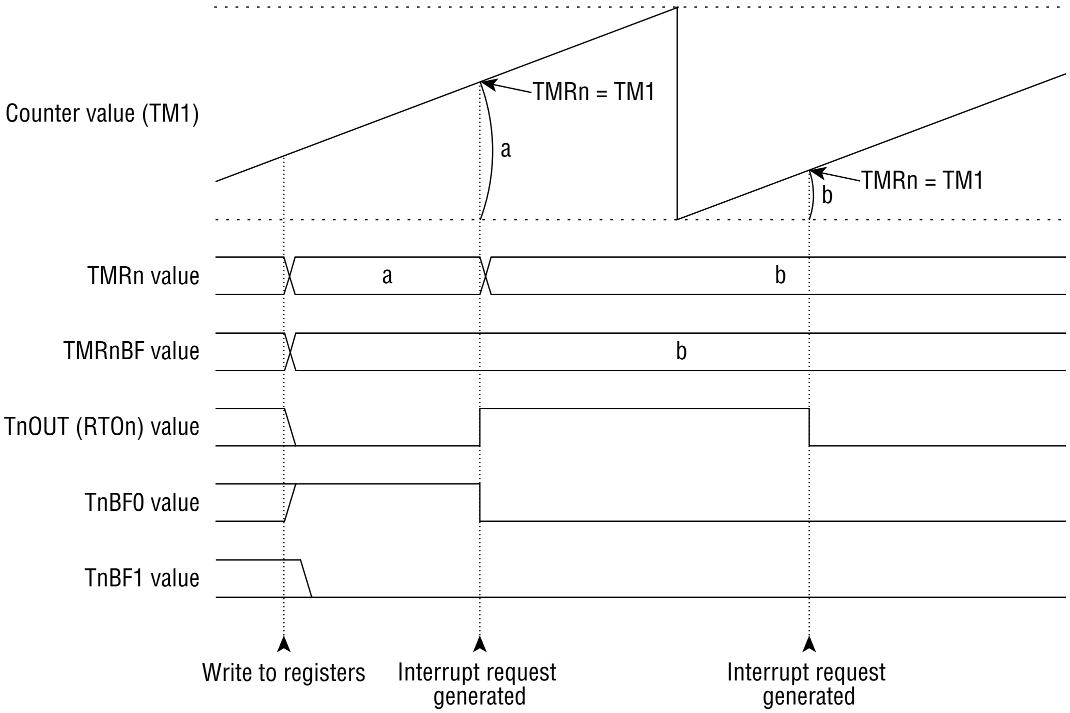


Figure 9-11 Operation Example of Register Module

Chapter 10

General 8-Bit Timer

10. General 8-Bit Timer

The MSM66587 family has one internal general 8-bit timer (GTM). Table 10-1 lists the SFRs for controlling GTM.

Table 10-1 SFRs for Controlling GTM

Address [H]	Name	Symbol (Byte)	Symbol (Word)	R/W	Reset State
00F8✰	General 8-Bit Timer Control Register	GTMCON	—	R/W	30
00FA	General 8-Bit Timer Counter	GTMC	—		00
00FB	General 8-Bit Timer Register	GTMR	—		00

Note: Address in the address column marked by a star indicates that the register has bits missing.

10.1 General 8-Bit Timer

The General 8-Bit Timer is configured from an 8-bit timer/counter (GTMC), an 8-bit timer register that stores the reload value of GTMC (GTMR), and an 8-bit timer control register for controlling operation (GTMCON).

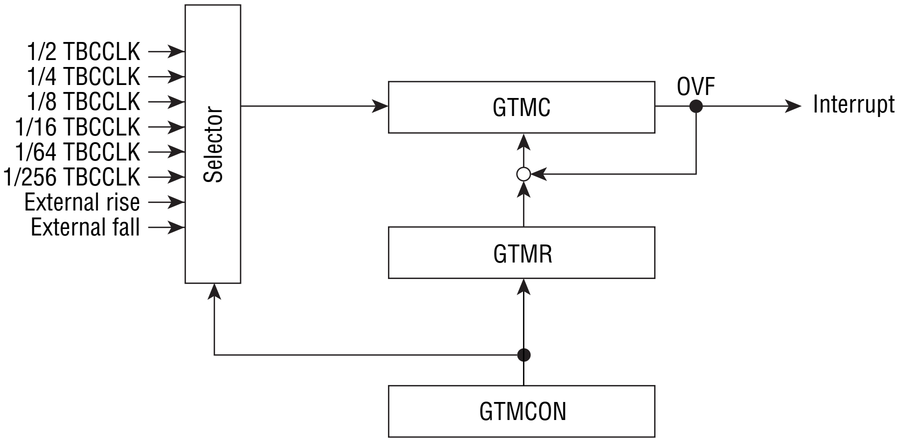


Figure 10-1 GTM Configuration

Note: If the General 8-Bit Timer is used in external clock mode, when stop mode is entered when released by an interrupt the General 8-Bit Timer Counter (GTMC) might increment by 1 depending on the input level of the external clock at the time of release.

(1) General 8-Bit Counter (GTMC)

GTMC is an 8-bit counter that generates an interrupt request and loads the General 8-Bit Timer Register (GTMR) when GTMC overflows. GTMC’s count clock is selected by the lower 3 bits of the General 8-Bit Timer Control Register (GTMCON).

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) GTMC will be 00H, with counting stopped. When an external count clock has been specified by GTMC, during stop mode, it will count on the clock’s falling edges.

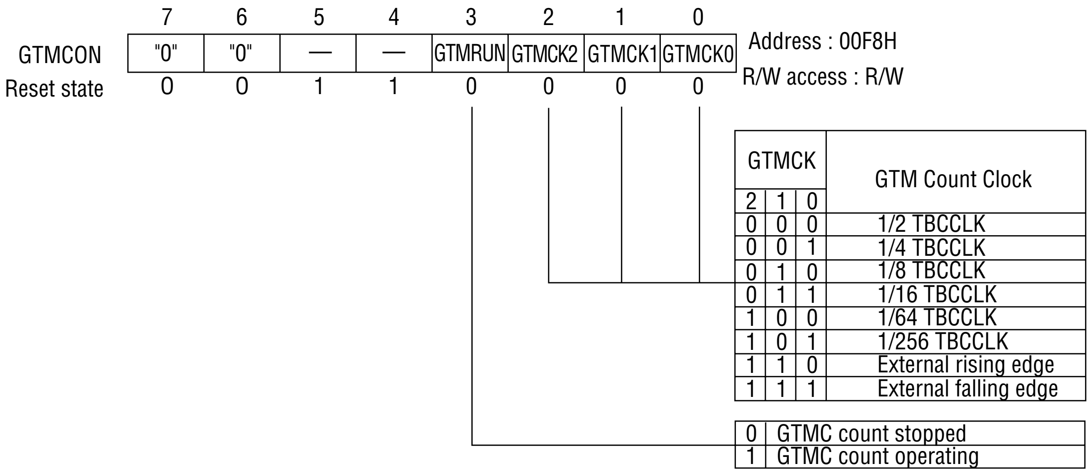
(2) General 8-Bit Register (GTMR)

GTMR is an 8-bit register that loads its contents to GTMC when GTMC overflows. After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) GTMR will be 00H.

(3) General 8-Bit Timer Control Register (GTMCON)

GTMCON is an 8-bit register, with its lower 4 bits selecting the GTMC count clock and controlling count start and stop. It upper 2 bits select the GEVC count clock.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) GTMCON will be 30H, selecting 1/2 TBCCLK as GTMC’s count clock and stopping count operation.



	7	6	5	4	3	2	1	0
GTMCON	"0"	"0"	—	—	GTMRUN	GTMCK2	GTMCK1	GTMCK0
Reset state	0	0	1	1	0	0	0	0

Address : 00F8H
R/W access : R/W

GTMCK			GTM Count Clock
2	1	0	
0	0	0	1/2 TBCCLK
0	0	1	1/4 TBCCLK
0	1	0	1/8 TBCCLK
0	1	1	1/16 TBCCLK
1	0	0	1/64 TBCCLK
1	0	1	1/256 TBCCLK
1	1	0	External rising edge
1	1	1	External falling edge

0	GTMC count stopped
1	GTMC count operating

— indicates bits that are not provided.
They will be read as "1".
Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 10-2 GTMCON Configuration

Chapter 11

PWM Function

11. PWM Function

The MSM66587 family has one internal PWM channel, configured as a divider circuit for the PWM count clock, a 16-bit counter, three 16-bit registers, and a control register.

If the master clock is 20 MHz, then a period of 100 ns to 209.7152 ms can be selected by the selected valued for PWM counter and input clock.

Figure 11-1 shows the configuration of PWM. Table 11-1 lists the SFRs for controlling PWM.

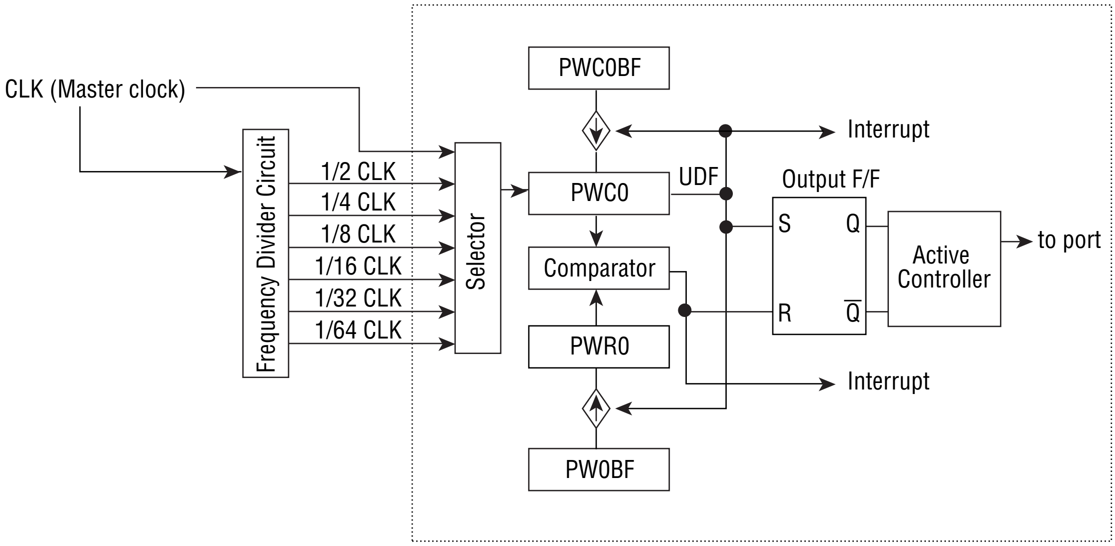


Figure 11-1 PWM Configuration

Table 11-1 PWM Control SFRs

Address [H]	Name	Symbol (Byte)	Symbol (Word)	R/W	Reset State
0050	PWM Counter 0	—	PWC0	R	FFFF
0051					
0060	PWC0 Buffer Register	—	PWC0BF	R/W	FFFF
0061					
0070	PWR0 Buffer Register	—	PW0BF		0000
0071					
00EB☆	PWMRUN Register	PWRUN	—		00
00EC☆	PWM Control Register 0	PWCON0	—		00
00F0☆	PWM Interrupt Register 0	PWINTQ0	PWINTQ		00
00F1☆	PWM Interrupt Register 1	PWINTQ1			00
00F2☆	PWM Interrupt Enable Register 0	PWINTE0	PWINTE		00
00F3☆	PWM Interrupt Enable Register 1	PWINTE1			00

Notes: 1. Addresses do not run consecutively.
2. Addresses in the adress column marked by stars indicates that the register has bits missing.

11.1 Clock Divider Circuit

The clock divider circuit creates the PWM count clock by dividing the master clock. It generates count clocks of 1/2 CLK, 1/4 CLK, 1/8 CLK, 1/16 CLK, 1/32 CLK, and 1/64 CLK.

11.2 PWM Configuration

One PWM set, PWM0, is provided.

PWM is configured from a 16-bit counter (PWC0), 16-bit counter buffer register (PWC0BF), 16-bit register (PWR0), 16-bit buffer register (PW0BF), PWC0/PWR0 comparator, output flip/flop, and registers for controlling operation (PWINTQ, PWINTE, PWCON0, PWRUN).

To use the PWM0 pin for the PWM function, set (1) the corresponding bits in the Port 7 Secondary Control Register.

(1) PWM Counter (PWC0)

PWC0 is a 16-bit down counter. Its input clock can be selected from the master clock CLK, 1/2 CLK, 1/4 CLK, 1/8 CLK, 1/16 CLK, 1/32 CLK, and 1/64 CLK. While the PW0RUN bit is "1", PWC0 will count down. When it underflows, the contents of the 16-bit PWM Buffer Register will be loaded into the PWM Register, and the output flip/flop will be set.

PWC0 can be read by the program, but it cannot be written. However, if the PW0RUN bit is 0 (stopped state) and a write to PWC0BF is performed, then those 16 bits will also be written to PWC0.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) PWC0 will be FFFFH.

(2) PWM Counter Buffer Register (PWC0BF)

PWC0BF is a 16-bit register for setting the period (stores the PWC0 reload value). PWC0BF contents will be loaded into the 16-bit PWM Counter when PWC0 underflows.

PWC0BF can be read and written by the program. If the PW0RUN bit is "0" (stopped state) and a write to PWC0BF is performed, then those 16 bits will also be written to PWC0.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) PWC0BF will be FFFFH.

(3) PWM Register (PWR0)

PWR0 is a 16-bit register that stores the duty value to be output. When PWC0 underflows, the contents of the 16-bit PWM Buffer Register will be loaded into PWR0.

PWR0 cannot be read or written by the program. However, if the PW0RUN bit is 0 (stopped state) and a write to PW0BF is performed, then those 16 bits will also be written to PWR0.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) PWR0 will be 0000H.

(4) PWM Buffer Register (PW0BF)

PW0BF is a 16-bit register. When PWC0 underflows, the contents of the 16-bit PWM Register will be loaded into PWM0BF.

PW0BF can be read and written by the program. If the PW0RUN bit is "0" (stopped state) and a write to PW0BF is performed, then those 16 bits will also be written to PWR0.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) PW0BF will be 0000H.

(5) Comparator Circuit

The comparator circuit continually compares the values of PWC0 and PWR0 while the PW0RUN bit is 1. When the values match, it generates an interrupt request and resets the output flip/flop.

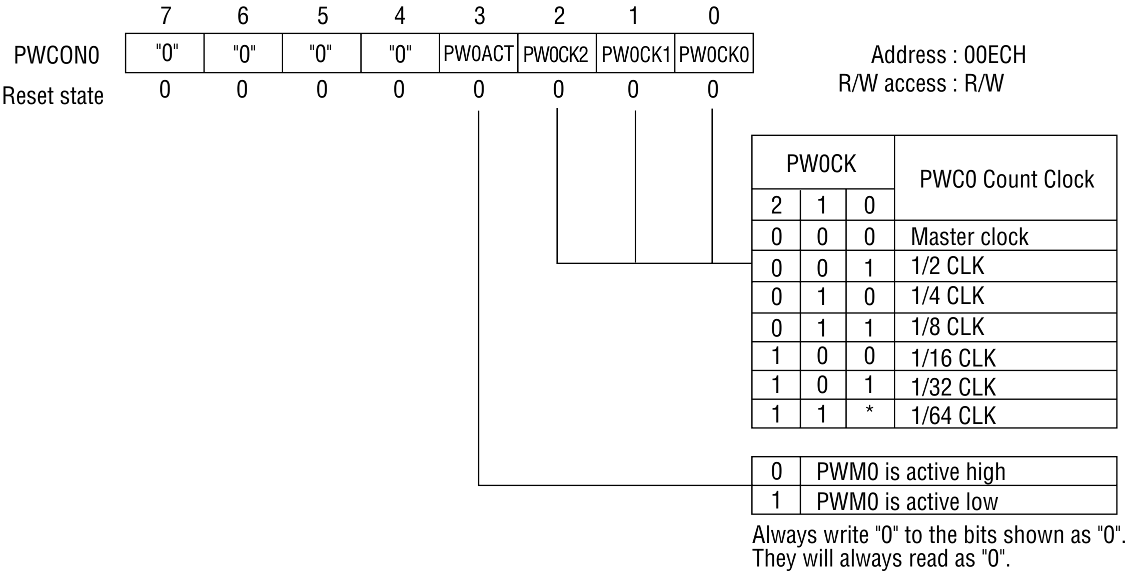
(6) Output Flip/Flop

The output flip/flop is set ("1") when the PW0RUN bit becomes "1" and when PWC0 overflows. It is reset ("0") when the values of PWC0 and PWR0 match.

(7) PWM Control Register (PWCON0)

PWCON0 is an 4-bit register that specifies the PWC0 count clock and PWM0 output logic. Figure 11-2 shows the configuration of PWCON0.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) PWCON0 will be 00H, setting the master clock as PWC0’s count clock and making PWM0 active high.



	7	6	5	4	3	2	1	0
PWCON0	"0"	"0"	"0"	"0"	PW0ACT	PW0CK2	PW0CK1	PW0CK0
Reset state	0	0	0	0	0	0	0	0

Address : 00ECH
R/W access : R/W

PW0CK			PWC0 Count Clock
2	1	0	
0	0	0	Master clock
0	0	1	1/2 CLK
0	1	0	1/4 CLK
0	1	1	1/8 CLK
1	0	0	1/16 CLK
1	0	1	1/32 CLK
1	1	*	1/64 CLK

0	PWM0 is active high
1	PWM0 is active low

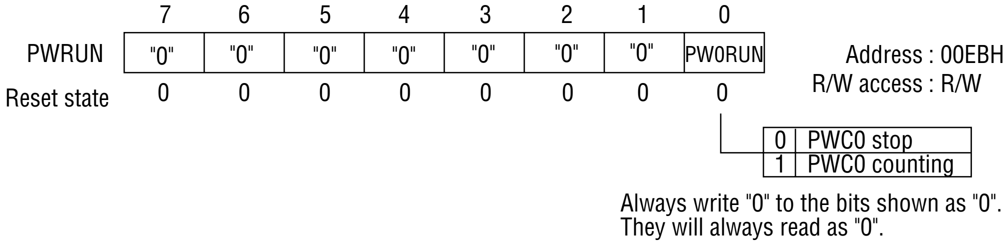
Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 11-2 PWCON0 Configuration

(8) PWM Run Register (PWRUN)

PWRUN is an 1-bit register that controls PWC0 counting operation to enable or to disable. Figure 11-3 shows the configuration of PWRUN.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) PWRUN will be 00H, stopping PWC0 counting.



	7	6	5	4	3	2	1	0
PWRUN	"0"	"0"	"0"	"0"	"0"	"0"	"0"	PW0RUN
Reset state	0	0	0	0	0	0	0	0

Address : 00EBH
R/W access : R/W

0	PWC0 stop
1	PWC0 counting

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 11-3 PWRUN Configuration

(9) PWM Interrupt Registers (PWINTQ0, PWINTQ1)

PWM0 issues two interrupt requests: when PWC0 underflows, and when the values of PWC0 and PWR0 match.

PWINTQ0 and PWINTQ1 are 1-bit registers. PWINTQ0 is a flag set (1) when PWC0 underflows. PWINTQ1 is a flag set (1) when the values of PWC0 and PWR0 match. Neither is automatically reset to 0, after their respective interrupts occur, so the program needs to reset them to 0.

If a PWC0 underflow and PWC0 and PWR0 match occur simultaneously (when PWR=0000H), then the PWC0 underflow interrupt request will take precedence. Only the PWINTQ0 bit will be set (1); the PWINTQ1 bit will not be set (1). Figure 11-4 shows the configuration of PWINTQ0, and Figure 11-5 shows the configuration of PWINTQ1.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, and op code trap), PWINTQ0 and PWINTQ1 will be 00H.

	7	6	5	4	3	2	1	0
PWINTQ0	"0"	"0"	"0"	"0"	"0"	"0"	"0"	QPW0
Reset state	0	0	0	0	0	0	0	0

Address : 00F0H
R/W access : R/W

0	PWC0 has not underflowed
1	PWC0 has underflowed

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 11-4 PWINTQ0 Configuration

	7	6	5	4	3	2	1	0
PWINTQ1	"0"	"0"	"0"	"0"	"0"	"0"	"0"	QPWR0
Reset state	0	0	0	0	0	0	0	0

Address : 00F1H
R/W access : R/W

0	PWC0 and PWR0 do not match.
1	PWC0 and PWR0 match.

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 11-5 PWINTQ1 Configuration

(10) PWM Interrupt Enable Registers (PWINTE0, PWINTE1)

PWINTE0 and PWINTE1 are 1-bit registers. PWINTE0 enables and disables interrupt requests generated by PWC0 underflow. PWINTE1 enables and disables interrupt requests generated when PWC0 and PWR0 values match. Figure 11-6 shows the configuration of PWINTE0, and Figure 11-7 shows the configuration of PWINTE1.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) PWINTE0 and PWINTE1 will be 00H, disabling PWM0 interrupt requests.

	7	6	5	4	3	2	1	0
PWINTE0	"0"	"0"	"0"	"0"	"0"	"0"	"0"	EPW0
Reset state	0	0	0	0	0	0	0	0

Address : 00F2H
R/W access : R/W

EPW0	
0	Disable PWC0 underflow interrupt requests.
1	Enable PWC0 underflow interrupt requests.

Always write "0" to the bits shown as "0".
They will always read as "0"

Figure 11-6 PWINTE0 Configuration

	7	6	5	4	3	2	1	0
PWINTE1	"0"	"0"	"0"	"0"	"0"	"0"	"0"	EPWR0
Reset state	0	0	0	0	0	0	0	0

Address : 00F3H
R/W access : R/W

EPWR0	
0	Disable PWC0/PWR0 match interrupt requests.
1	Enable PWC0/PWR0 match interrupt requests.

Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 11-7 PWINTE1 Configuration

11.3 PWM Operation

PWM0 control the duty of an output within a particular period (determined by the PWC0 count clock and the contents of PWC0 and PWC0BF).

PWM is started by setting ("1") the RUN bit. At the same time the RUN bit becomes "1", the output flip/flop will be set ("1") and the PWM output pin will output a high level (for active high operation). When the values of PWC0 and PWR0 match, the output flip/flop will be reset ("0"), the PWM output pin will output a low level, and an interrupt request flag will be set ("1"). When PWC0 underflows, the output flip/flop will be set ("1"), the PWM output pin will output a high level, and the interrupt request flag will be set ("1"). At the same time, the contents of PWC0BF will be loaded in PWC0 and the contents of PW0BF will be loaded into PWR0. This operation will repeat until the RUN bit is reset ("0"), with the duty-controlled waveform being output from the PWM output pin.

If a PWC0 underflow and PWC0 and PWR0 match occur simultaneously (when PWR = 0000H), then the PWC0 underflow interrupt request will take precedence. Only the PWINTQ0 bit will be set ("1"); the PWINTQ1 bit will not be set ("1").

Depending on the count clock selected for PWC0, the duty immediately after PWM is started (for just one period) might be shortened. The maximum deviations are listed below.

Count Clock	Maximum Deviation
Master clock	0
1/2 CLK	–1/2 CLK
1/4 CLK	–3/4 CLK
1/16 CLK	–15/16 CLK
1/64 CLK	–63/64 CLK

If PWC0 and PWC0BF are FFFFH, PWM is active high, and PWR is FFFFH, then the duty output will be 1/65536. If the value of PWR is decreased, then the output duty (time interval at high level) will increase. If PWR is 0000H, then the duty will become 65536/65536, or 100%. A duty of 0/65536, or 0%, cannot be realized by the PWM block.

The PWM period can be calculated with the following expression.

$$f_{(PWM)} : f_{(OSC)} \times \text{PWCLK} \times \frac{1}{N+1}$$

$f_{(PWM)}$: PWM frequency (Hz)
$f_{(OSC)}$: master clock frequency (Hz)
PWCLK : PWM input clock
N : value of PWC0 and PWC0BF

Example: Assume the master clock frequency is 20 MHz, the PWM input clock is 1/2 CLK, and the value of PWC0BF is 00FFH.

$$f_{(PWM)} = 20{,}000{,}000 \times \frac{1}{2} \times \frac{1}{(255+1)} = 39{,}062.5 \text{ Hz } (25.6\ \mu\text{s})$$

Figure 11-8 shows an example of PWM output operation. Figure 11-9 shows an example of the timing of PWM output changes.

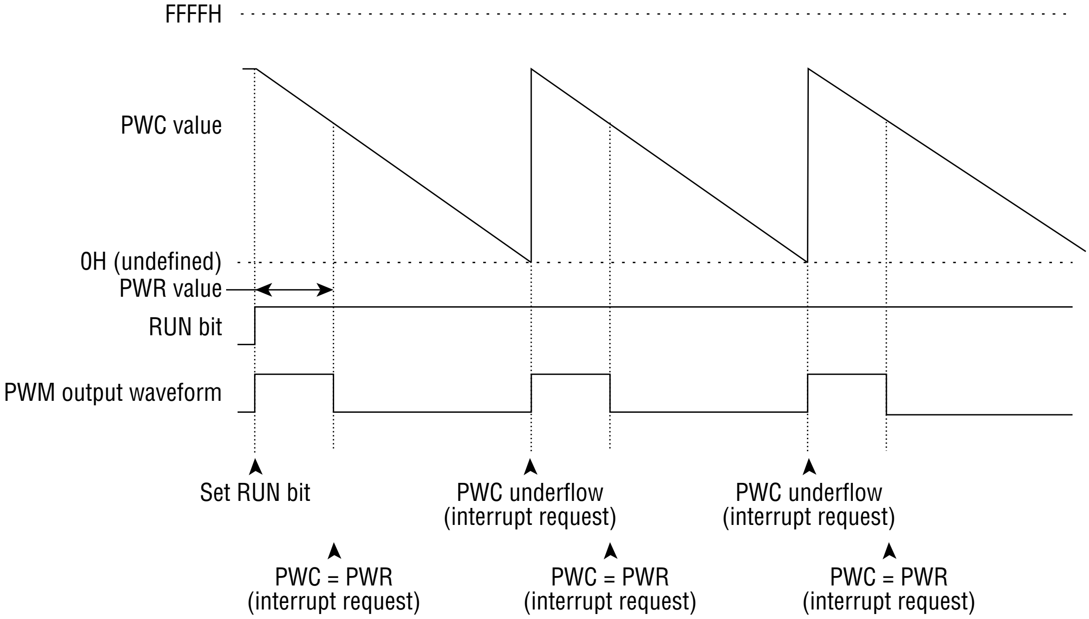


Figure 11-8 PWM Output Operation Example (Active High)

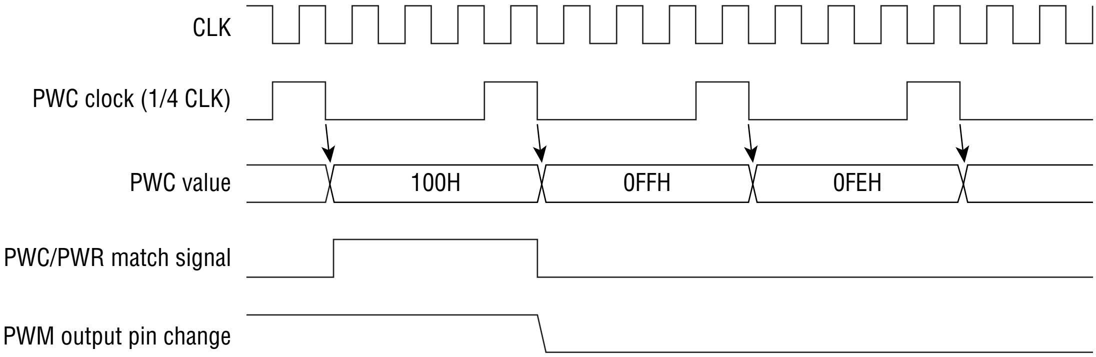


Figure 11-9 PWM Output Change Timing

Chapter 12

Baud Rate Generator Functions

12. Baud Rate Generator Functions

The MSM66587 family has a synchronous serial port for serial communication. It provides an 8-bit timer (S1TM) that can be used as a baud rate generator for that serial port. When this timer is not used as a baud rate generator, it can be used as an 8-bit auto-reload timer.

Figure 12-1 shows the configuration of S1TM. Table 12-1 lists the S1TM control SFRs.

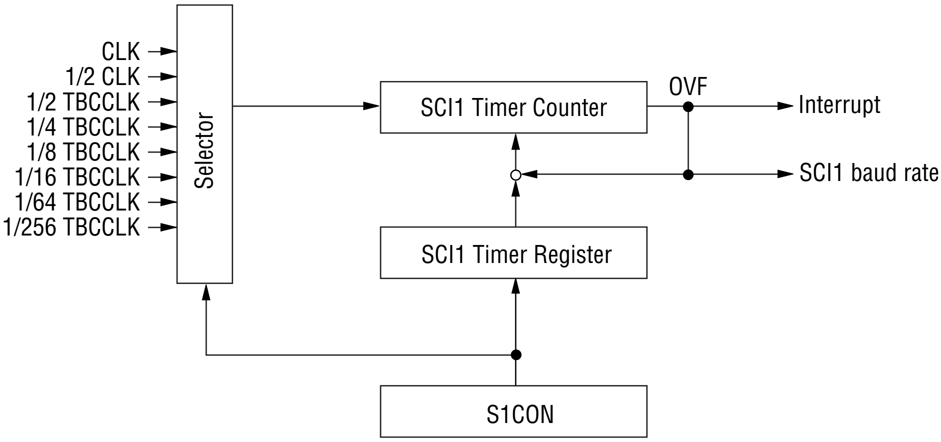


Figure 12-1 S1TM Configuration

Table 12-1 S1TM Control SFRs

Address [H]	Name	Symbol (Byte)	Symbol (Word)	R/W	Reset State
0118	SCI1 Timer Counter	—	S1TM	R/W	0000
0119	SCI1 Timer Register	—			
011A☆	SCI1 Timer Control Register	S1CON	—		02

Note: Address in the address column marked by a star indicates that the register has bits missing.

12.1 SCI1 Timer (S1TM) Configuration

The SCI1 Timer is configured from the 8-bit SCI1 Timer Counter, the 8-bit SCI1 Timer Register (which stores the reload value of the SCI1 Timer Counter), and the SCI1 Control Register (which specifies S1TM operation).

(1) SCI1 Timer Counter (lower 8 bits of 16-bit register S1TM)

The SCI1 Timer Counter is an 8-bit counter. When it overflows, an interrupt request will be generated and it will be loaded with the contents of the 8-bit SCI1 Timer Register. The S1TM count clock is selected by the upper 3 bits (bits 5-7) of the 8-bit SCI1 Timer Control Register (S1CON).

(2) SCI1 Timer Register (higher 8 bits of 16-bit register S1TM)

The SCI1 Timer Register is an 8-bit register. Its contents will be loaded into the SCI1 Timer Counter when the SCI1 Timer Counter overflows.

The SCI1 Timer Counter forms the lower 8 bits, and the SCI1 Timer Register forms the upper 8 bits, of the 16-bit S1TM, which can only be accessed as a word.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) S1TM will be 0000H, stopping the count operation.

(3) SCI1 Timer Control Register (S1CON)

S1CON is an 7-bit register. Its upper 4 bits (bits 4-7) select the S1TM count clock and control starting and stopping of count operations. Its next lower 2 bits (bits 2-3) control interrupts. The least significant bit (bit 0) specifies the S1TM mode.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) S1CON will be 02H, selecting CLK as the S1TM count clock, selecting auto-reload timer mode, and stopping the count operation.

Figure 12-2 shows the configuration for S1CON.

[Bit descriptions]

- S1MOD (bit 0)

  This bit specifies the S1TM operating mode.

  When this bit is "0", S1TM will operate in auto-reload timer mode. When "1", S1TM will operate as the SCI1 baud rate generator.

- QSTM1 (bit 2)

  This bit will become "1" when the SCI1 Timer Counter overflows. Interrupt processing will automatically reset this bit to "0", so the program must do so.

- ESTM1 (bit 3)

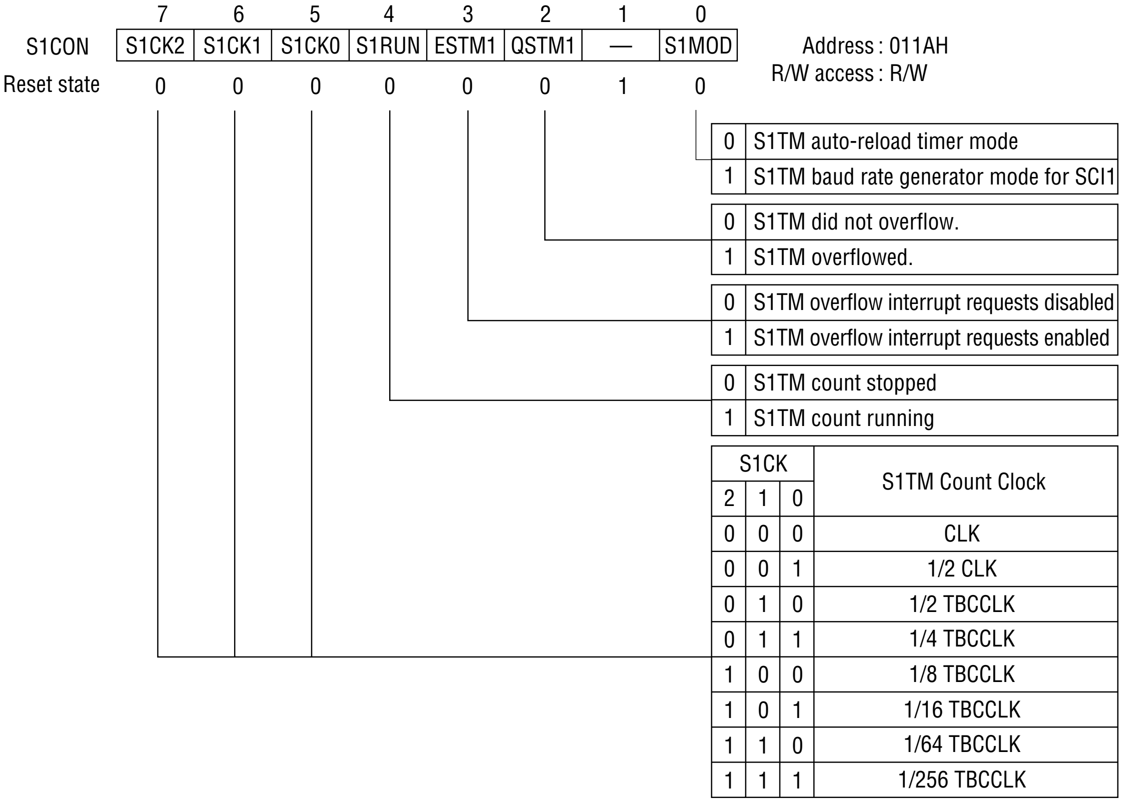
  This bit enables and disables interrupt requests from being generated when the SCI1 Timer Counter overflows. When this bit is "1", interrupt requests are enabled. When "0", they are disabled.

- S1RUN (bit 4)

  This bit specifies whether the SCI1 Timer Counter runs or stops. When this bit is "0", the SCI1 Timer Counter is disabled. When "1", it is enabled.

- S1CK0-S1CK2 (bits 5-7)

  These bits specify the SCI1 Timer Counter count clock.



	7	6	5	4	3	2	1	0
S1CON	S1CK2	S1CK1	S1CK0	S1RUN	ESTM1	QSTM1	—	S1MOD
Reset state	0	0	0	0	0	0	1	0

Address : 011AH
R/W access : R/W

0	S1TM auto-reload timer mode
1	S1TM baud rate generator mode for SCI1

0	S1TM did not overflow.
1	S1TM overflowed.

0	S1TM overflow interrupt requests disabled
1	S1TM overflow interrupt requests enabled

0	S1TM count stopped
1	S1TM count running

S1CK			S1TM Count Clock
2	1	0	
0	0	0	CLK
0	0	1	1/2 CLK
0	1	0	1/2 TBCCLK
0	1	1	1/4 TBCCLK
1	0	0	1/8 TBCCLK
1	0	1	1/16 TBCCLK
1	1	0	1/64 TBCCLK
1	1	1	1/256 TBCCLK

— indicates bits that are not provided.
They will be read as "1".

Figure 12-2 S1CON Configuration

12.2 SCI1 Timer Operation

The SCI1 Timer basically operates as an auto-reload timer. When the 8-bit SCI1 Timer Counter overflows, it will be loaded with the value of the 8-bit SCI1 Timer Register. At the same time the overflow generates an interrupt request. The baud rate when SCI1 is used as a baud rate generator can be calculated from the expression below.

$$\text{Synchronous mode : } B = f_{(BRG)} \times \frac{1}{256 - D} \times \frac{1}{4}$$

B : Baud rate
$f_{(BRG)}$: S1TM input clock frequency (Hz)
D : Reload value (0-255)

Writing a reload value to the SCI1 Timer Register will not change the contents of the SCI1 Timer Counter. When used as a fixed auto-reload counter (baud rate generator) from the start of counting, it is needed by program to write the same value to the SCI1 Timer Counter and SCI1 Timer Register.

Chapter 13

Serial Port Functions

13. Serial Port Functions

The MSM66587 family provides a 1-channel serial port (SCI1). SCI1 also provides a dedicated baud rate generator.

SCI1 has a transmit side and receive side. Its communications are synchronous, with serial data transfer synchronized to a controlled shift clock.

Synchronous mode includes a master mode, which generates the shift clock internally, and slave mode, which accepts an externally supplied shift clock. Table 13-1 lists the serial port modes.

Table 13-1 Serial Port Modes

Serial Port	Transmit Side	Synchronous Mode (SCI1)	Master Mode
			Slave Mode
	Receive Side	Synchronous Mode (SCI1)	Master Mode
			Slave Mode

13.1 Serial Port Configuration

The serial port is configured from the following registers and other elements.

- Baud rate generator, for generating the transfer speed (refer to Chapter 12: S1TM)
- SCI1 Control Registers (SR1CON, ST1CON), for controlling transmit and receive operations
- SCI1 Buffer Register (S1BUF), for setting transmit and receive data
- SCI1 Status Register (S1STAT)
- SCI1 Transmit/Receive Register

Figure 13-1 shows the configuration of SCI1. Table 13-2 lists the SFRs for controlling the serial port.

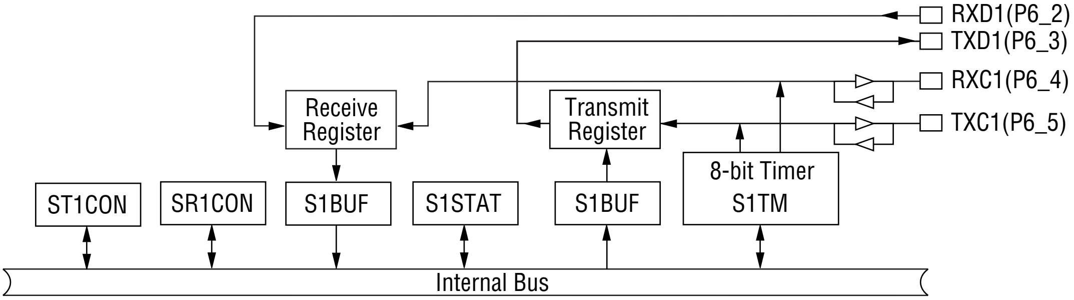


Figure 13-1 SCI1 Configuration

Table 13-2 Serial Port Control SFRs

Address [H]	Name	Symbol (Byte)	Symbol (Word)	R/W	Reset State
00F6	SCI1 Transmit/Receive Buffer Register	S1BUF	—	R/W	undefind
00F7☆	SCI1 Status Register	S1STAT	—		00
011B☆	SCI1 Transmit Control Register	ST1CON	—		8E
011C☆	SCI1 Transmit/Receive Control Register	SR1CON	—		0E

Notes: 1. Addresses do not run consecutively.
2. Addresses in the address column marked by stars indicate that the register has bits missing.

13.1.1 SCI1 Transmit/Receive Control Register (SR1CON)

SR1CON is a 2-bit register that controls transmit and receive operation.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) SR1CON will be 0EH, disabling SCI1 transmit/receive and placing both operations in slave mode.

Do not change SR1MST (bit 4) while data is being transmitted. Change it after the data transmit operation has finished. Also, do not change SR1MST while data is being received. Change it after the data receive operation has finished, or after SR1REN (bit 7) has been reset (0). If SR1CON is changed before SR1REN has been reset, the current or later transmit/receive operations will not execute correctly.

Figure 13-2 shows the configuration of SR1CON.

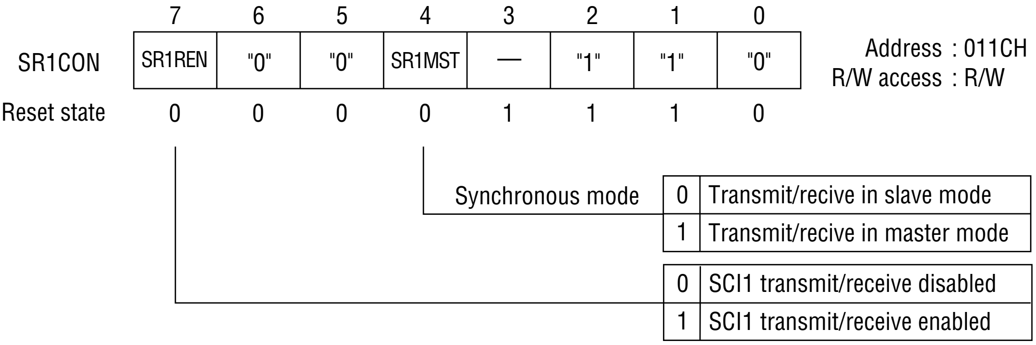
[Bit descriptions]

- SR1MST (bit 4)

  Setting this bit to "0" sets slave mode. Setting it to "1" sets master mode.

- SR1REN (bit 7)

  This bit enables and disables transmit/receive operation.
  Setting this bit to "0" disables SCI1 transmit/receive operation. Setting it to "1" enables SCI1 transmit/receive operation.



	7	6	5	4	3	2	1	0
SR1CON	SR1REN	"0"	"0"	SR1MST	—	"1"	"1"	"0"
Reset state	0	0	0	0	1	1	1	0

Address : 011CH
R/W access : R/W

Synchronous mode

0	Transmit/recive in slave mode
1	Transmit/recive in master mode

0	SCI1 transmit/receive disabled
1	SCI1 transmit/receive enabled

— indicates bits that are not provided.
They will be read as "1".
Always write "0" to the bits shown as "0".
They will always read as "0".
And also write "1" to the bits shown as "1".
They will read as "1".

Figure 13-2 SR1CON Configuration

Note

When using the serial port, be sure to set ST1MST (bit 4) of ST1CON and SR1MST (bit 4) of SR1CON to the same value.
(SR1MST = ST1MST = "0" or SR1MST = ST1MST = "1")

13.1.2 SCI1 Transmit Control Register (ST1CON)

ST1CON is a 1-bit register that controls SCI1 transmit operation.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) ST1CON will be 8EH, placing SCI1 transmit operation in slave mode.

Do not change ST1CON while data is being transmitted. Change it after the data transmit operation has finished. If ST1CON is changed before the data transmit operation has finished, the current or later transmit operations will not execute correctly.

Note

When using the serial port, be sure to set ST1MST (bit 4) of ST1CON and SR1MST (bit 4) of SR1CON to the same value.
(SR1MST = ST1MST = "0" or SR1MST = ST1MST = "1")

Figure 13-3 shows the configuration of ST1CON.

[Bit descriptions]

- ST1MST (bit 4)

  Setting this bit to 0 sets slave mode. Setting it to 1 sets master mode.

	7	6	5	4	3	2	1	0
ST1CON	—	"0"	"0"	ST1MST	—	"1"	"1"	"0"
Reset state	1	0	0	0	1	1	1	0

Address : 011BH
R/W access : R/W

0	Transmit in slave mode
1	Transmit in master mode

— indicates bits that are not provided.
They will be read as "1".
Always write "0" to the bits shown as "0".
They will always read as "0".
And also write "1" to the bits shown as "1".
They will read as "1".

Figure 13-3 ST1CON Configuration

13.1.3 SCI1 Transmit/Receive Buffer Register (S1BUF)

S1BUF is an 8-bit register that saves transmit and receive data for serial port transfer operations. S1BUF has dual functions different for reading and writing. In other words, it serves as a receive buffer when read, and as a transmit buffer when written.

At the time a receive operation finishes, the contents of the SCI1 Receive Register will be transferred to S1BUF’s receive buffer, and simultaneously a receive interrupt request will be generated. The contents of the S1BUF receive buffer will be saved until the next receive operation finishes.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) S1BUF will be undefined.

13.1.4 SCI1 Transmit/Receive Register

The SCI1 Transmit/Receive Register is an 8-bit shift register that performs the actual shift operations for data transmit and receive.

The SCI1 Transmit/Receive Register and the SCI1 Transmit/Receive Buffer Register (S1BUF) are arranged as two levels. When a receive operation finishes, the data latched in the SCI1 Transmit/Receive Register will be transferred to S1BUF, and a receive interrupt request will be generated.

The SCI1 Transmit/Receive Register cannot be read or written by the program.

13.1.5 SCI1 Status Register (S1STAT)

The upper 4 bits of S1STAT are the interrupt request control register for serial port transmit ready and receive ready. Bit 1 of S1STAT saves the status (normal or error) when a serial port receive operation finishes.

S1STAT bit 1 is updated at the time a receive operation finishes. When it has been set once (1 = error), then it will not be reset (0) even if the next receive operation finishes without error. After a receive operation finishes, the program should reset (0) bit 1 of S1STAT.

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) S1STAT will be 00H. Figure 13-4 shows the configuration of S1STAT.

[Bit Descriptions]

- OERR1 (bit 1)

  This bit will be set ("1") when an SCI1 receive operation finishes and the CPU cannot read the last receive data. However, even in this case S1BUF will be loaded with new receive data (overrun error).

- RV1IE (bit 4)

  This bit enables and disables interrupts generated by an SCI1 receive ready state. Setting this bit to "1" enables interrupts, while resetting to "0" disables interrupts.

- RV1IRQ (bit 5)

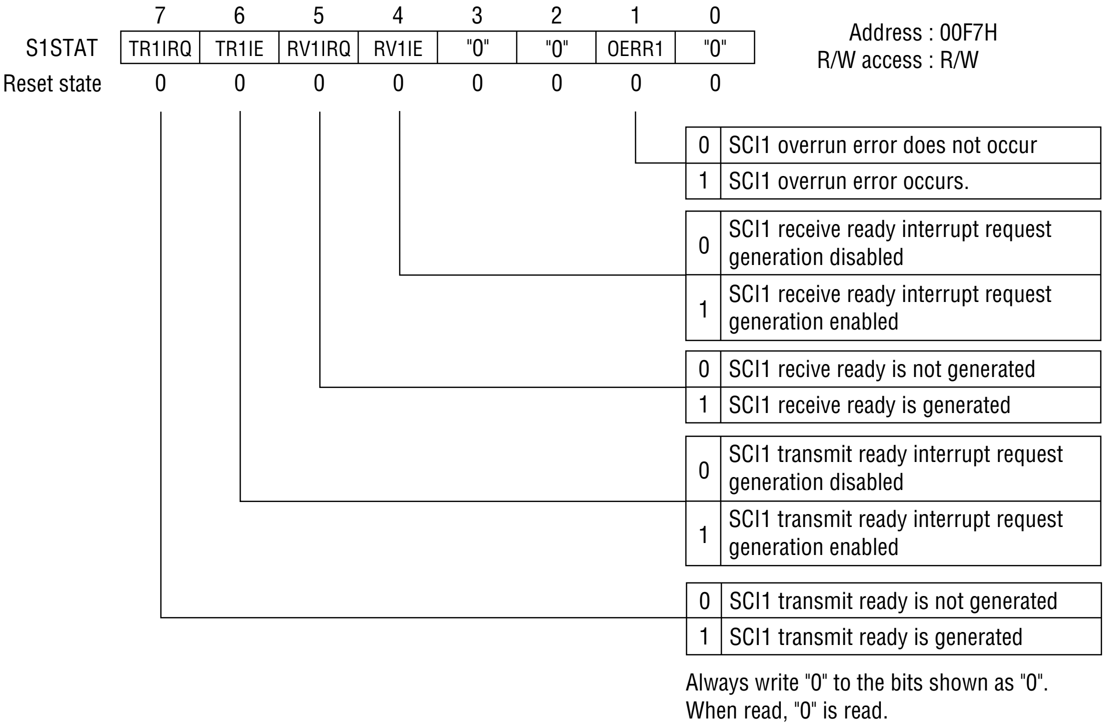
  This bit becomes "1" when an SCI1 receive ready state occurs. It will not automatically be reset to "0" by execution of the interrupt process, so the program should reset it.

- TR1IE (bit 6)

  This bit enables and disables interrupts generated by an SCI1 transmit ready state. Setting this bit to "1" enables interrupts, while resetting to "0" disables interrupts.

- TR1IRQ (bit 7)

  This bit becomes "1" when an SCI1 transmit ready state occurs. It will not automatically be reset to "0" by execution of the interrupt process, so the program should reset it.



	7	6	5	4	3	2	1	0
S1STAT	TR1IRQ	TR1IE	RV1IRQ	RV1IE	"0"	"0"	OERR1	"0"
Reset state	0	0	0	0	0	0	0	0

Address : 00F7H
R/W access : R/W

0	SCI1 overrun error does not occur
1	SCI1 overrun error occurs.

0	SCI1 receive ready interrupt request generation disabled
1	SCI1 receive ready interrupt request generation enabled

0	SCI1 recive ready is not generated
1	SCI1 receive ready is generated

0	SCI1 transmit ready interrupt request generation disabled
1	SCI1 transmit ready interrupt request generation enabled

0	SCI1 transmit ready is not generated
1	SCI1 transmit ready is generated

Always write "0" to the bits shown as "0".
When read, "0" is read.

Figure 13-4 Configuration of S1STAT

13.2 Serial Port Operation

13.2.1 Transmit Operation

<Synchronous Master Mode>

In synchronous master mode, the clock pulse obtained by the Baud Rate Generator (S1TM) is frequency divided by four, generating the transmit shift clock (ST1CLK). The transmit circuit synchronizes transmit data transfer to ST1CLK.

The start of a transmit operation is triggered by the signal that writes transmit data to S1BUF (WS1BUF: signal output when an instruction is executed that writes to S1BUF, like “STB A, S1BUF”). When WS1BUF is generated, after one clock (master clock) the signal to start the transmit operation (LST1SF) will be generated.

The falling edge of LST1SF transfers the contents of S1BUF to the SC1 Transmit Register.

The transmit operation will then start, setting the signal that indicates transmit operation in progress (ST1FREE) to a low level. When ST1FREE goes low, the transmit shift clock will be output from the TXC1 pin synchronized to the second falling edge of ST1CLK. After a half clock, bit 0 of the transmit data (LSB first) will be output from the TXD1 pin.

After the falling of LST1SF, synchronized to the signal that indicates the start of instruction execution (M1S1), a transmit interrupt request (TX1READY) will be generated, and the interrupt request flag (QSCI1) will be set ("1").

Then in accordance to the specification of SR1CON, transmit data and a parity bit will be output synchronized to TXC1 (that is, ST1CLK), completing transmission of one frame.

TXD1 will change a half clock after the falling edge of ST1CLK just before the falling edge of TXC1. Therefore, the receive side should latch TXD1 on the rising edge of TXC1.

<Synchronous Slave Mode>

In synchronous slave mode, the transmit clock is input externally, so ST1CLK is obtained by detecting edges synchronized to CLK. The transmit circuit synchronizes transmit data transfer to ST1CLK.

The start of a transmit operation is triggered by the signal that writes transmit data to S1BUF (WS1BUF: signal output when an instruction is executed that writes to S1BUF, like “STB A, S1BUF”). When WS1BUF is generated, after one clock (master clock) the signal to start the transmit operation (LST1SF) will be generated.

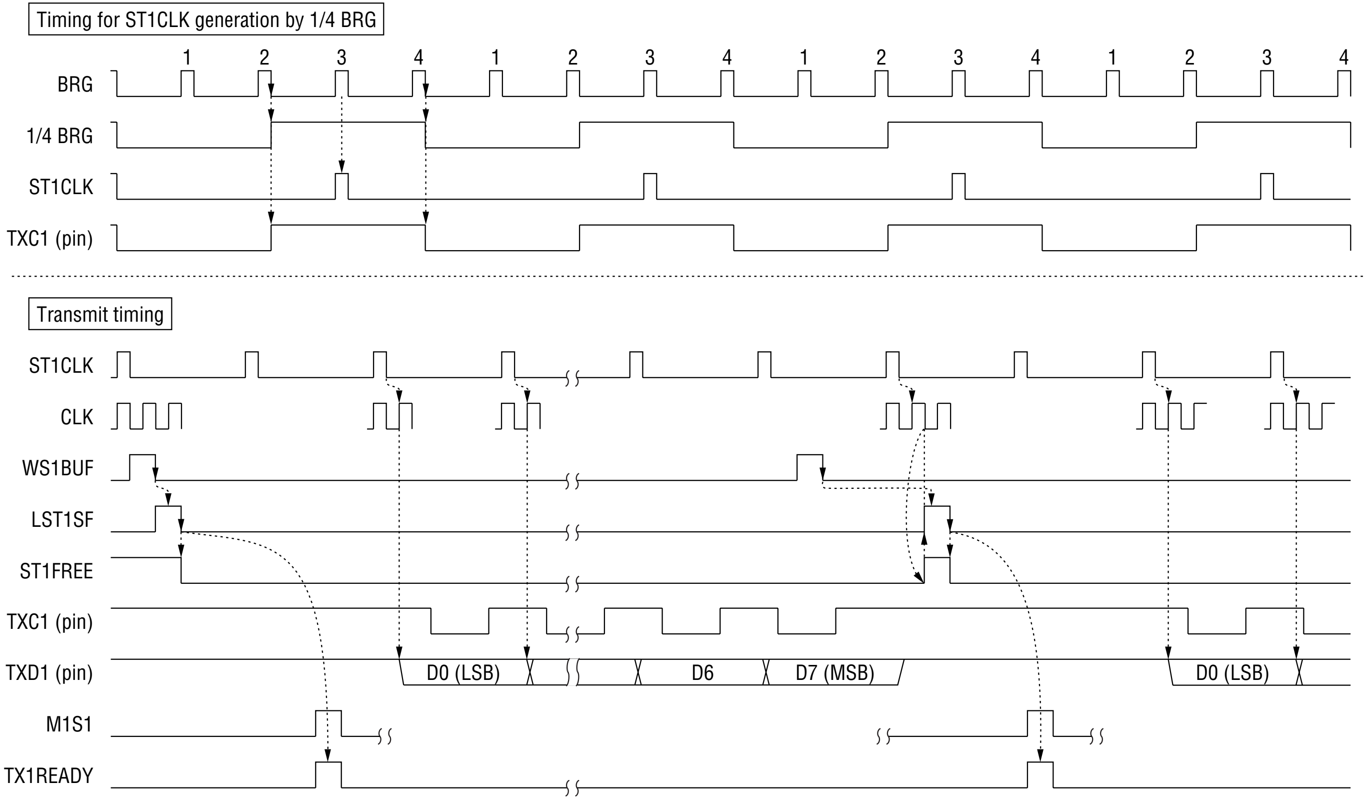
The falling edge of LST1SF transfers the contents of S1BUF to the SC1 Transmit Register.

The transmit operation will then start, setting the signal that indicates transmit operation in progress (ST1FREE) to a low level. When ST1FREE goes low, after a half clock bit 0 of the transmit data (LSB first) will be output from the TXD1 pin. Bit 1 of the transmit data is then output from the TXD1 pin after another half clock.

After the falling of LST1SF, synchronized to the signal that indicates the start of instruction execution (M1S1), a transmit interrupt request (TX1READY) will be generated, and the interrupt request flag (QSCI1) will be set ("1").

Then in accordance to the specification of SR1CON, transmit data and a parity bit will be output synchronized to TXC1 (that is, ST1CLK), completing transmission of one frame.

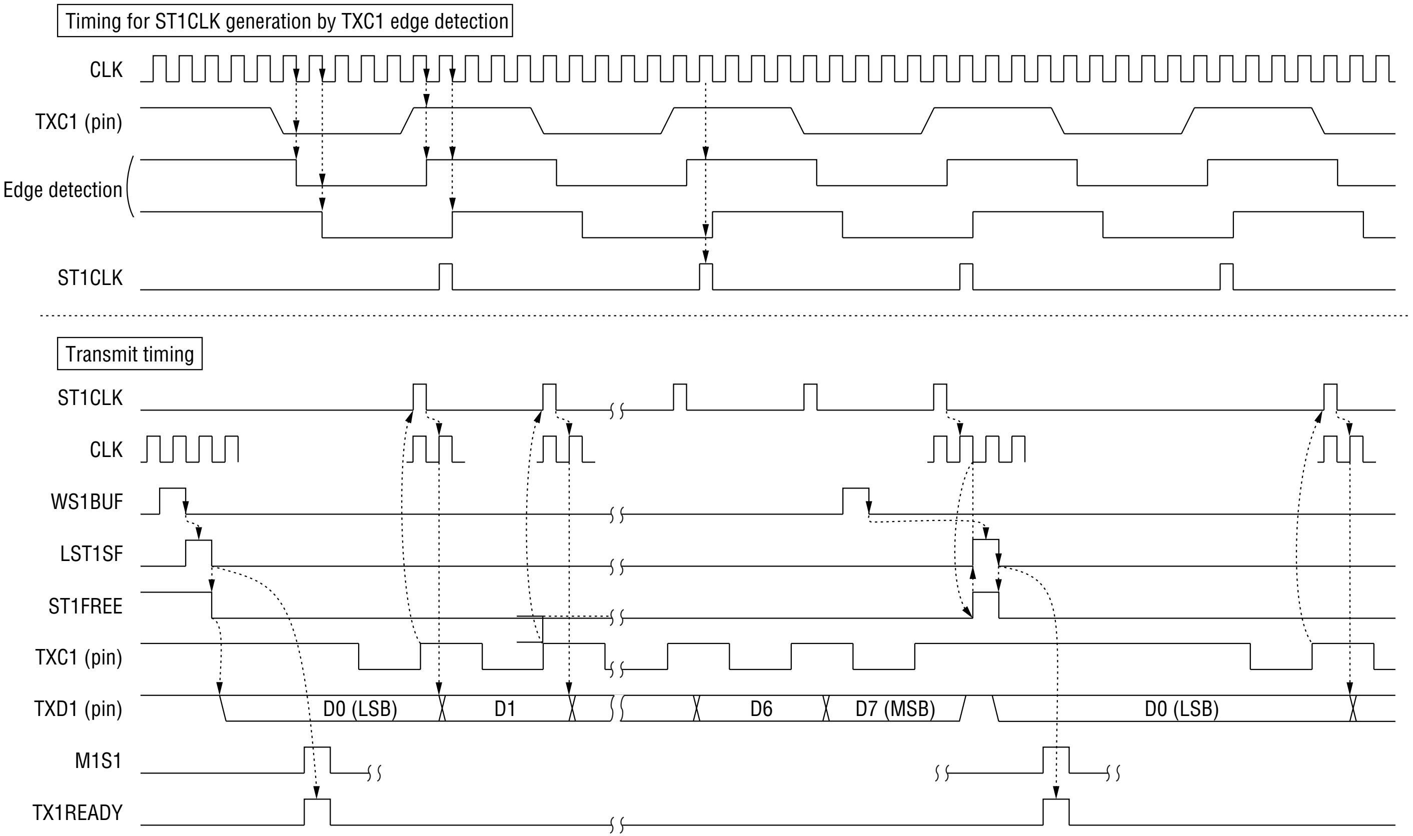
TXD1 will change a half clock after the falling edge of ST1CLK generated, after detecting the rising edge of TXC1 externally applied. Therefore, the receiving device should latch TXD1 on the rising edge of TXC1.



Explanation of symbols

BRG	Clock pulse obtained from baud rate generator (S1TM)
1/4 BRG	BRG divided by 4
ST1CLK	Transmit shift clock
TXC1(pin)	Transmit shift clock output from this pin (P6_5)
CLK	Master clock
WS1BUF	Write signal to S1BUF
LST1SF	Transmit start signal
ST1FREE	Signal indicating transmit in progress ("0")
TXD1 (pin)	Transmit data output from this pin (P6_3)
M1S1	Signal indicating start of instruction
TX1READY	Transmit interrupt request signal

Figure 13-5 Transmit Synchronous Master Mode Timing Chart Example



Explanation of symbols		
	CLK	Master clock
	TXC1(pin)	Write signal to S1BUF
	Edge detection	Transmit shift clock
	ST1CLK	Transmit start signal
	WS1BUF	Signal indicating transmit in progress (0)
	LST1SF	Signal indicating start of instruction
	ST1FREE	Transmit interrupt request signal
	TXD1 (pin)	Transmit shift clock input from this pin (P6_5)
	M1S1	Transmit shift clock detected from edges of CLK input from TXC1 pin
	TX1READY	Transmit data output from this pin (P6_3)

Figure 13-6 Transmit Synchronous Slave Mode Timing Chart Example

13.2.2 Receive Operation

<Synchronous Master Mode>

In synchronous master mode, the clock pulse obtained by the Baud Rate Generator (S1TM) is frequency divided by four, generating the receive shift clock (SR1CLK) and receive data sampling clock.

The second pulse of the 4 divided clock will become the RXD1 pin’s input sampling clock, and the third pulse will become SR1CLK. The receive circuit latches receive data synchronized to SR1CLK.

The start of a receive operation begins when the SR1REN bit of SR1CON is set to "1". This will set the signal that indicates receive operation in progress (SR1FREE) to a low level. When SR1FREE goes low, the receive shift clock will be output from the RXC1 pin synchronized to the next SR1CLK.

The next SR1CLK shifts the receive data that has been sampled just before into the SCI1 Receive Register. Receive data is sampled just once.

SR1CLK is generated after the rising edge of RXCT, and receive data is shifted into the SCI1 Receive Register. Therefore, the transmit side can send transmit data synchronized to the falling edge of RXC1. Then in accordance to the specification of SR1CON, receive data will be shifted in sequence into the SCI1 Receive Register synchronized to the receive clock output from RXC1.

When the last receive shift clock has been output, the receive end signal (LSR1BUF) will be generated synchronized to the next SR1CLK. When LSR1BUF is generated the contents of the SCI1 Receive Register (receive data) will be transferred to S1BUF. If there is an overrun error, the overrun error flag (OERR1) will be set "1", the receive interrupt request signal (RX1READY) will be generated synchronized to the signal that indicates the start of instruction execution (M1S1), and the interrupt request flag (QSCI1) will be set "1". Also, SR1FREE will go to a high level, and the SR1REN bit of SR1CON will be "0" to terminate the receive operation.

<Synchronous Slave Mode>

In synchronous slave mode, the start of a receive operation begins when the SR1REN bit of SR1CON is set high. This will set the signal that indicates receive operation in progress (SR1FREE) to a low level, which will accept the receive shift clock input from the RXC1 pin.

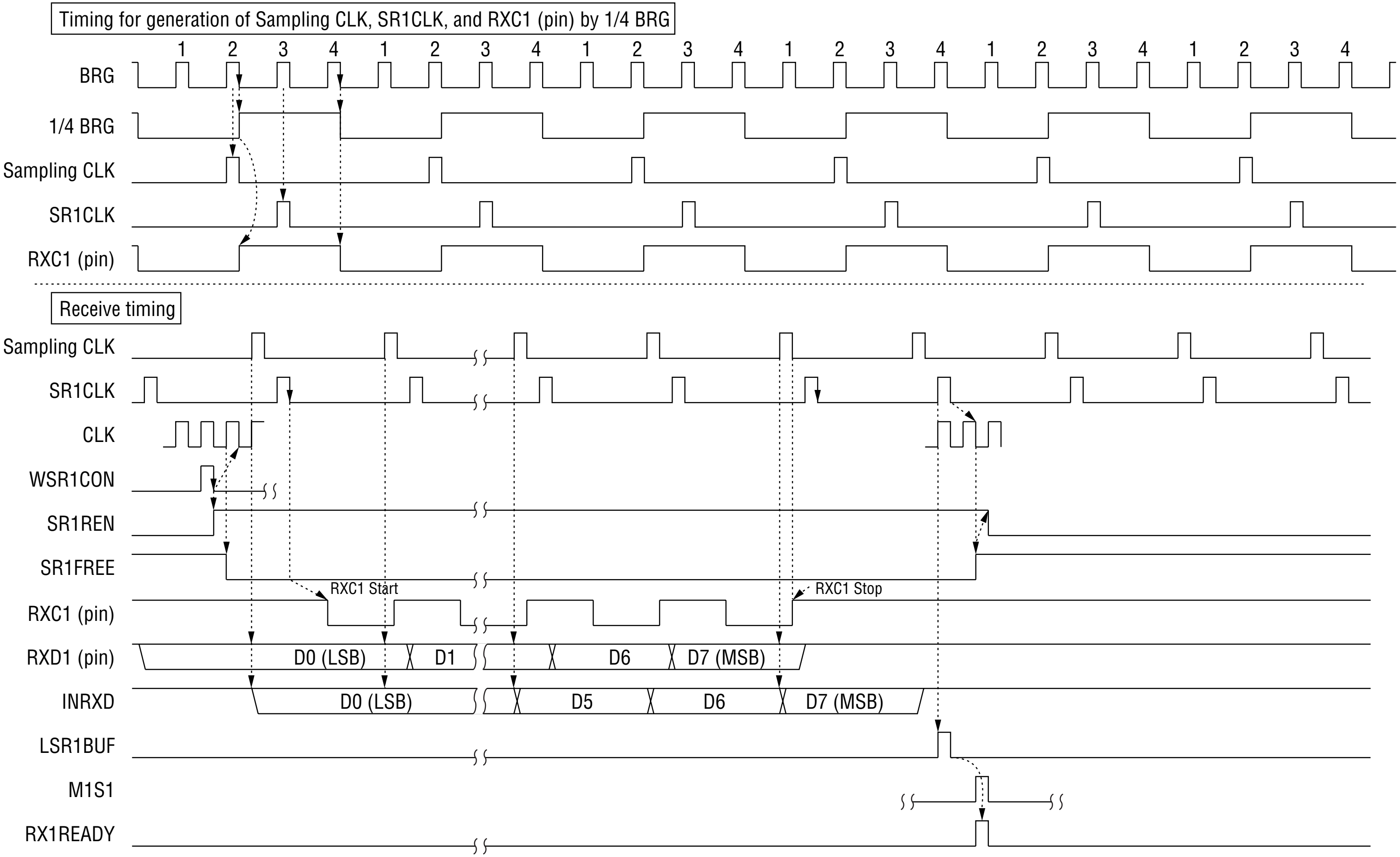
The receive shift clock SR1CLK is obtained by detecting the input signal’s edges synchronized to CLK. Receive data control is synchronized to SR1CLK.

When the RXC1 pin goes from low to high, the receive shift clock SR1CLK will be generated by the edge detection circuit. When SR1CLK is generated, the receive data just sampled on the RXD1 pin will be shifted in to the SCI1 Receive Register.

RXD1 pin sampling is performed while the RXC1 pin is low. The shift into the SCI1 Receive Register occurs after the RXC1 pin rises, so the transmit side should send data synchronized to the falling edges of the transmit shift clock on the RXC1 pin.

Then in accordance to the specification of SR1CON, receive data will be shifted in sequence into the SCI1 Receive Register synchronized to the input receive clock. Finally SR1CLK is obtained from edge detection of the receive shift clock, and the last receive data is input. After a half clock, the receive end signal (LSR1BUF) will be generated.

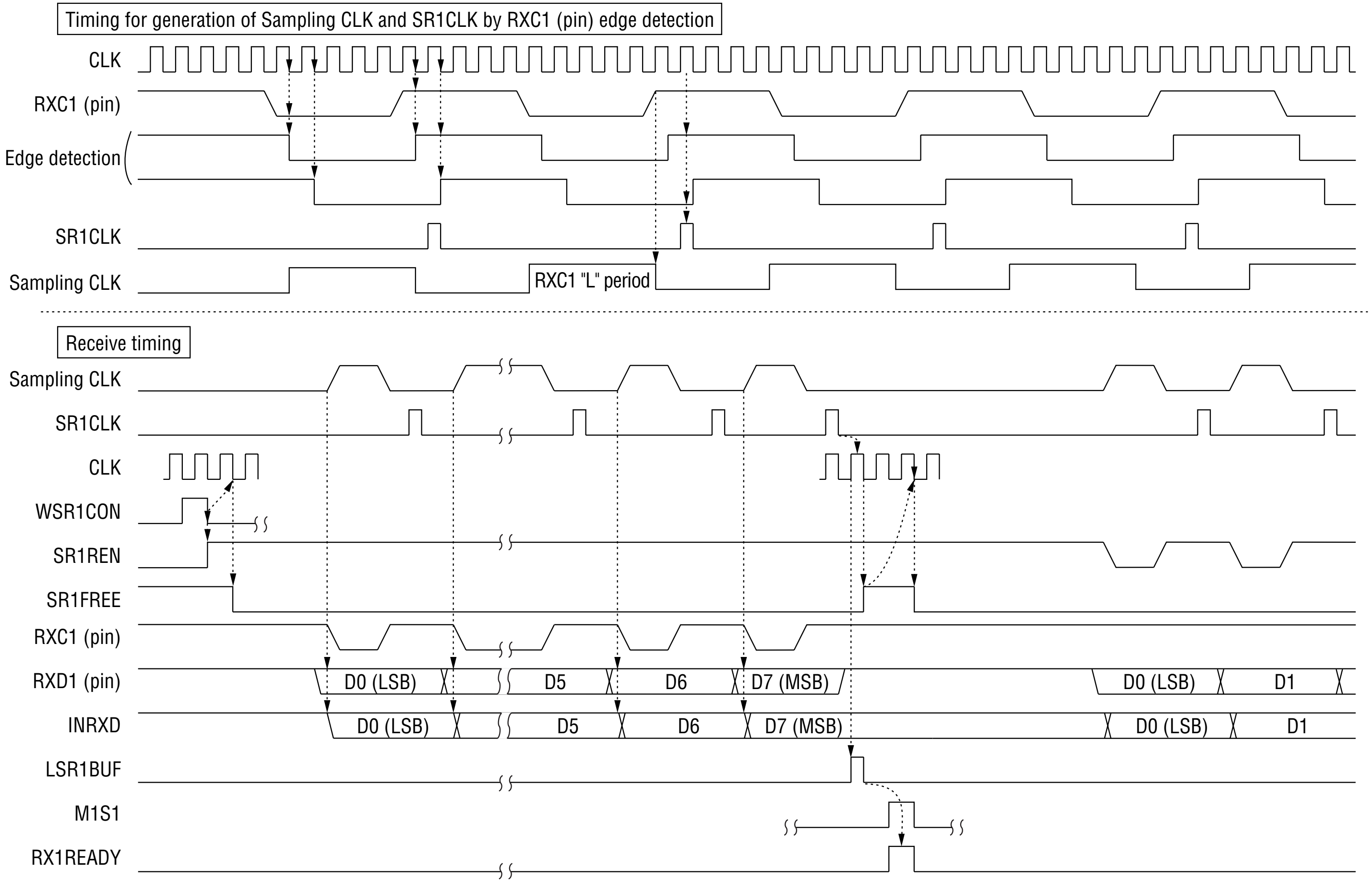
When LSR1BUF is generated the contents of the SCI1 Receive Register (receive data) will be transferred to S1BUF. If there is an overrun error, the overrun error flag (OERR1) will be set ("1"), the receive interrupt request signal (RX1READY) will be generated synchronized to the signal that indicates the start of instruction execution (M1S1), and the interrupt request flag (QSCI1) will be set ("1"). Also, SR1FREE will go to a high level. The SR1REN bit of SR1CON will not become low at this time, so if the next receive shift clock is input then the receive operation will start again.



Explanation of symbols

BRG	Clock pulse obtained from baud rate generator (S1TM)	1/4 BRG	BRG divided by 4
Sampling CLK	Receive data sampling clock	SR1CLK	Receive shift clock
RXC1 (pin)	Receive shift clock output from this pin (P6_4)	CLK	Master clock
WSR1CON	Write signal to SR1CON	SR1REN	Receive enable signal (bit 7 of SR1CON)
SR1FREE	Signal indicating receive in progress ("0")	RXD1 (pin)	Receive data input to this pin (P6_2)
INRXD	Receive data sampled from RXD1 by the RXD sampling clock	LSR1BUF	Receive end signal
M1S1	Signal indicating start of instruction	RX1READY	Receive interrupt request signal

Figure 13-7 Receive Synchronous Master Mode Timing Chart Example



Explanation of symbols

CLK	Master clock	RXC1 (pin)	Receive shift clock input to this pin (P6_4)
Edge detection	Receive shift clock detected by edges on CLK input to RXC1 pin	SR1CLK	Receive shift clock
		Sampling CLK	Receive data sampling clock
WSR1CON	Write signal to SR1CON	SR1REN	Receive enable signal (bit 7 of SR1CON)
SR1FREE	Signal indicating receive in progress (0)	RXD1 (pin)	Receive data input to this pin (P6_2)
INRXD	Receive data sampled from RXD1 by the RXD sampling clock	LSR1BUF	Receive end signal
M1S1	Signal indicating start of instruction	RX1READY	Receive interrupt request signal

Figure 13-8 Receive Synchronous Slave Mode Timing Chart Example

Chapter 14

A/D Converter Functions

14. A/D Converter Functions

The MSM66587 family provides an 8-bit resolution A/D converter with a four channel-analog input multiplexer.

The A/D converter uses a sample-and-hold successive comparison method for converting analog levels to digital data. The conversion result for the selected channel will be stored in the A/D Result Register.

Figure 14-1 shows the configuration of the A/D converter. Table 14-1 lists the SFRs for A/D converter control.

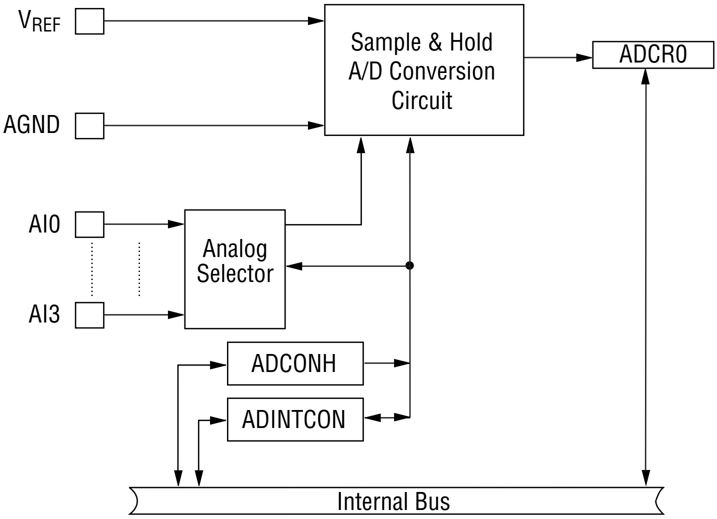


Figure 14-1 A/D Converter Configuration

Table 14-1 A/D Converter Control SFRs

Address [H]	Name	Symbol (Byte)	R/W	Reset State
002C✰	A/D Interrupt Control Register	ADINTCON	R/W	C0H
00D0	A/D Result Register 0	ADCR0	R	undefind
00E1✰	A/D Control Register High	ADCONH	R/W	80H

Notes: 1. Addresses do not run consecutively.
2. Addresses in the address column marked by stars indicate that the register has bits missing.

14.1 A/D Converter Configuration

(1) A/D Control Register High (ADCONH)

ADCONH is a 5-bit register that controls the A/D converter. Figure 14-2 shows the configuration of ADCONH. After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) ADCONH will be 80H.

[Bit Descriptions]

- ADSTM0, ADSTM1 (bit 0, bit1)

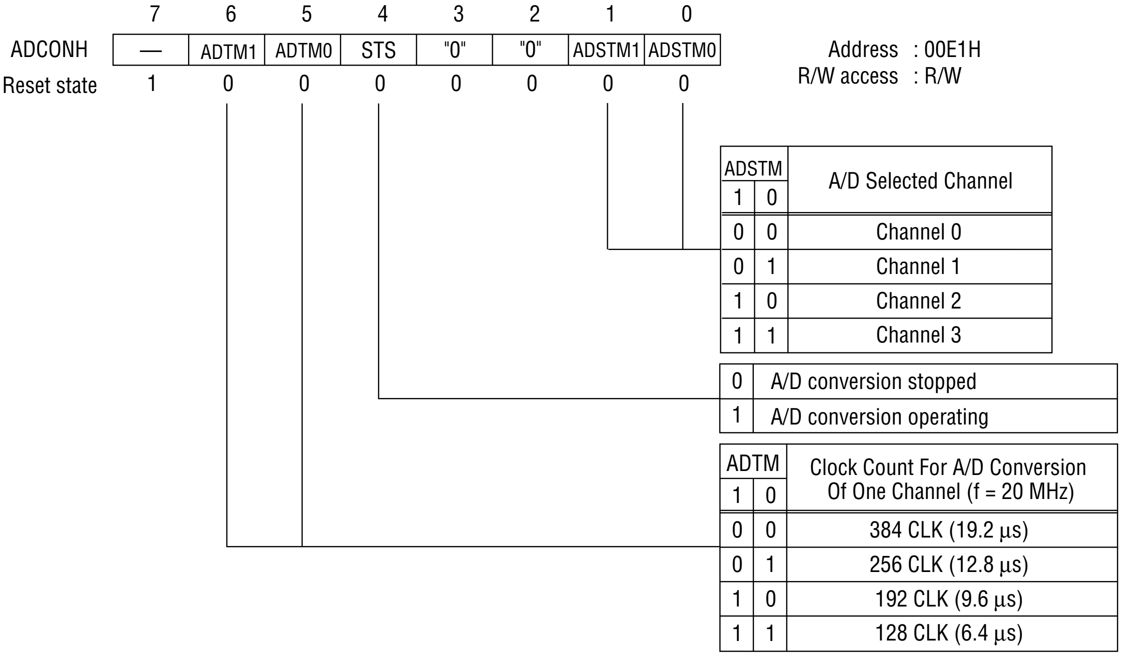
  These bits specify the A/D conversion channel. The channel change should be performed after STS (bit 4) has been reset to "0". While STS is "1" (A/D conversion in progress), changes to the selected channel will not be effective.

- STS (bit 4)

  This bit starts and stops A/D conversion. When this bit is set to 1, A/D conversion will be performed on the channel specified by ADSTM0-ADSTM1 (bit 0, bit 1). This bit will become "0" when A/D conversion finishes.

- ADTM0, ADTM1 (bit 5, bit 6)

  These bits specify the number of clocks needed to perform A/D conversion on one channel. Changes to the clock count specification will not be effective during A/D conversion.



	7	6	5	4	3	2	1	0
ADCONH	—	ADTM1	ADTM0	STS	"0"	"0"	ADSTM1	ADSTM0
Reset state	1	0	0	0	0	0	0	0

Address : 00E1H
R/W access : R/W

ADSTM 1	ADSTM 0	A/D Selected Channel
0	0	Channel 0
0	1	Channel 1
1	0	Channel 2
1	1	Channel 3

STS	
0	A/D conversion stopped
1	A/D conversion operating

ADTM 1	ADTM 0	Clock Count For A/D Conversion Of One Channel (f = 20 MHz)
0	0	384 CLK (19.2 µs)
0	1	256 CLK (12.8 µs)
1	0	192 CLK (9.6 µs)
1	1	128 CLK (6.4 µs)

— indicates bits that are not provided.
They will be read as "1".
Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 14-2 Configuration of ADCONH

(2) A/D Interrupt Control Register (ADINTCON)

ADINTCON is a 2-bit register that mainly controls interrupt requests generated by the A/D converter. Figure 14-3 shows the configuration of ADINTCON. After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) ADINTCON will be C0H.

[Bit Descriptions]

- INTST (bit 1)

  This bit indicates that A/D conversion has finished. When "1", A/D conversion has finished. This bit must be reset ("0") by the program.

- ADSTIE (bit 3)

  This bit enables and disables interrupt requests generated by A/D conversion finishing. When "0", interrupt requests are disabled. When 1, interrupt requests are enabled.

	7	6	5	4	3	2	1	0
ADINTCON	—	—	"0"	"0"	ADSTIE	"0"	INTST	"0"
Reset state	1	1	0	0	0	0	0	0

Address : 002CH
R/W access : R/W

0	A/D conversion not finished
1	A/D conversion finished

0	INTST interrupt generation disabled
1	INTST interrupt generation enabled

— indicates bits that are not provided.
They will be read as "1".
Always write "0" to the bits shown as "0".
They will always read as "0".

Figure 14-3 Configuration of ADINTCON

(3) A/D Result Register (ADCR0)

The A/D Result Register stores the result of A/D conversion on the selected channel. Figure 14-4 shows the configuration of ADCR0. After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) ADCR0 will be undefined.

	7	6	5	4	3	2	1	0
ADCR0								
Reset state	undefind							

Address : 00D0H
R/W access : R

Figure 14-4 Configuration of ADCR0

Peripheral Functions

15. Peripheral Functions

The MSM66587 family provides a clock out function for supplying a clock to peripheral devices.

15.1 Clock Out Function

The clock out function outputs a clock pulse obtained by dividing the master clock from the CLKOUT pin (secondary function of port P7_3). The frequency division of the master clock is specified by the Peripheral Control Register (PRPHF) lower 2 bits (bit 0 and bit 1). There are four master clock frequency division ratios: 1/4, 1/8, 1/16, and 1/32. The Port 7 Secondary Control Register bit 3 must be set ("1") to use the CLKOUT pin.

Figure 15-1 shows the configuration of PRPHF.

	7	6	5	4	3	2	1	0
PRPHF	Undefined	Undefined	—	—	—	—	CKOUT1	CKOUT0
Reset state	undefined	undefined	1	1	1	1	0	0

Address : 0107H
R/W access : R/W

CKOUT		CLKOUT Pin Output Clock
1	0	
0	0	1/4 CLK
0	1	1/8 CLK
1	0	1/16 CLK
1	1	1/32 CLK

— indicates bits that are not provided. They will be read as "1".
"undefined" indicates bits that are not provided.
Assuming these bits are always undefined, programming should be done.
Operation of the bits may differ between ICE evaluation and actual device as MASK/OTP version etc.

Figure 15-1 Configuration of PRPHF

Chapter 16

External Interrupt Requests

16. External Interrupt Requests

The MSM66587 family provides five external interrupt requests of two types. There are four maskable interrupts (INT0-INT3), and one non-maskable interrupt (NMI).

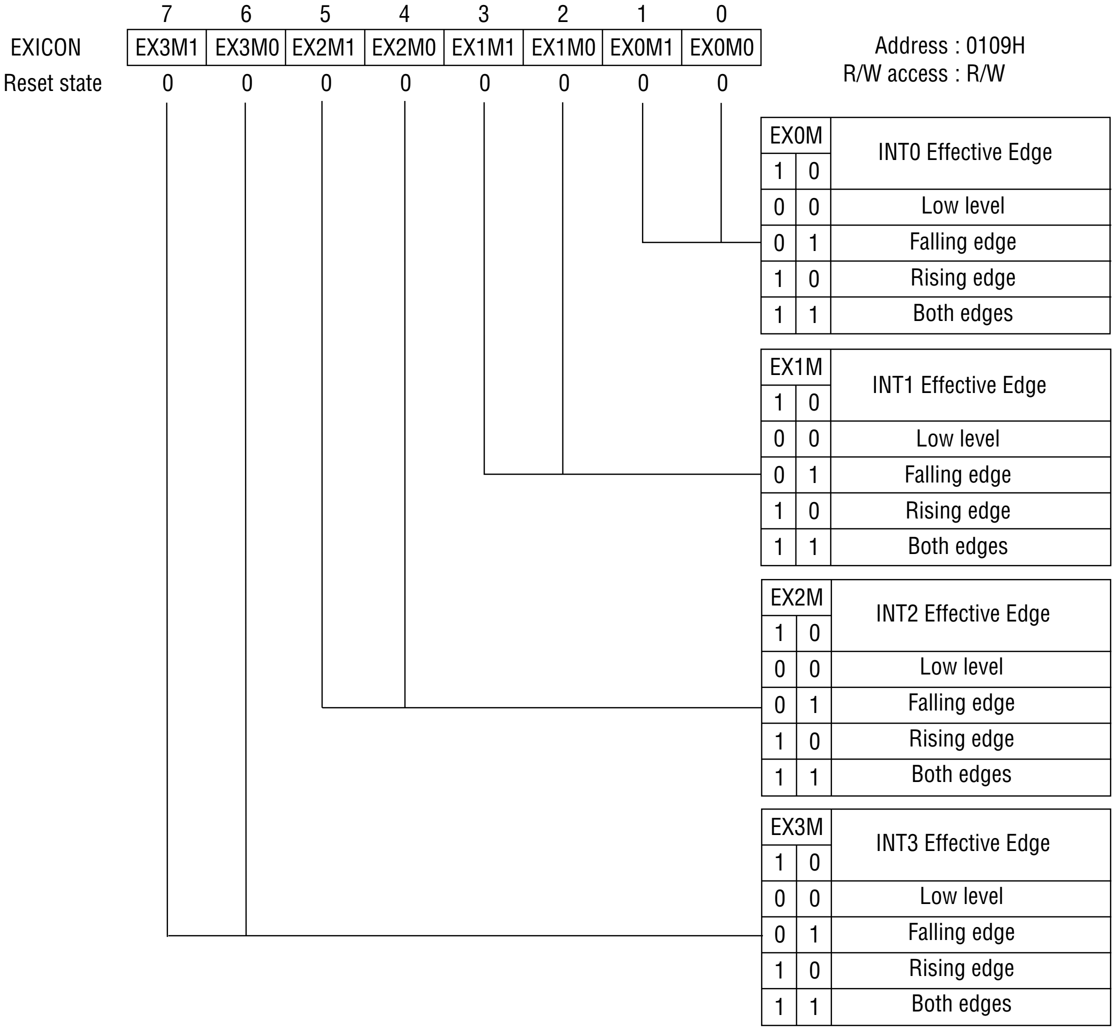
INT0-INT3 are allocated as the secondary functions of Port 6’s P6_0 and P6_1 and Port 12’s P12_2 and P12_3. Therefore the appropriate bits in the Port 6 Secondary Control Register and Port 12 Secondary Control Register must be set ("1") to use INT0-INT3.

A dedicated pin (pin 35) is provided for NMI.

The effective edge of INT0-INT3 can be specified by the External Interrupt Control Register (EXICON). The edge specification is disabled and the "L" level is enabled, during the stop mode. The effective edge of NMI can be specified by the NMI Interrupt Control Register (NMICON).

After reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) EXICON will be 00H, setting the effective edge of INT0-INT3 to a low level. NMICON will be FCH or 7CH, setting its effective edge to the falling edge.

Figure 16-1 shows the configuration of EXICON, and Figure 16-2 shows the configuration of NMICON. Bit 7 of NMICON is the bit that monitors the NMI pin.



	7	6	5	4	3	2	1	0
EXICON	EX3M1	EX3M0	EX2M1	EX2M0	EX1M1	EX1M0	EX0M1	EX0M0
Reset state	0	0	0	0	0	0	0	0

Address : 0109H
R/W access : R/W

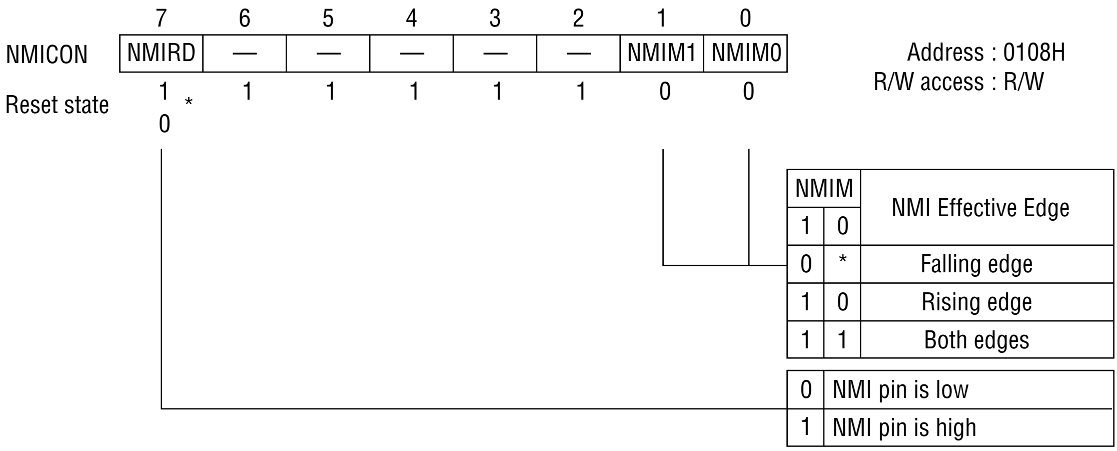
EX0M		INT0 Effective Edge
1	0	
0	0	Low level
0	1	Falling edge
1	0	Rising edge
1	1	Both edges

EX1M		INT1 Effective Edge
1	0	
0	0	Low level
0	1	Falling edge
1	0	Rising edge
1	1	Both edges

EX2M		INT2 Effective Edge
1	0	
0	0	Low level
0	1	Falling edge
1	0	Rising edge
1	1	Both edges

EX3M		INT3 Effective Edge
1	0	
0	0	Low level
0	1	Falling edge
1	0	Rising edge
1	1	Both edges

Figure 16-1 Configuration of EXICON



	7	6	5	4	3	2	1	0
NMICON	NMIRD	—	—	—	—	—	NMIM1	NMIM0
Reset state	1 * 0	1	1	1	1	1	0	0

Address : 0108H
R/W access : R/W

NMIM		NMI Effective Edge
1	0	
0	*	Falling edge
1	0	Rising edge
1	1	Both edges

0	NMI pin is low
1	NMI pin is high

— indicates bits that are not provided.
They will be read as "1".
Writes to bit 7 will be ignored.
* indicates "0" or "1".
The value after reset will be determined by the NMI pin level.

Figure 16-2 Configuration of NMICON

Chapter 17

Interrupt Request Processing

17. Interrupt Request Processing

The M66587 family provides 13 interrupt source types, 5 external and 9 internal, each of which is assigned a vector. One of the external interrupts is a non-maskable interrupt. Also, the 13 maskable interrupts can be set to three levels of interrupt priority.

Table 17-1 lists the interrupt control SFRs.

Table 17-1 Interrupt Control SFRs

Address [H]	Name	Symbol (Byte)	Symbol (Word)	R/W	Reset State
0040☆	Interrupt Request Register 0	IRQ0L	IRQ0	R/W	00
0041☆		IRQ0H			00
0042☆	Interrupt Request Register 1	IRQ1L	—		00
0044☆	Interrupt Enable Register 0	IE0L	IE0		00
0045☆		IE0H			00
0046☆	Interrupt Enable Register 1	IE1L	—		00
0048☆	Interrupt Priority Control Register 00	IP00L	IP00		00
0049☆		IP00H			00
004A☆	Interrupt Priority Control Register 01	IP01L	IP01		00
004B☆		IP01H			00
004C☆	Interrupt Priority Control Register 10	IP10L	—		00
004E☆	Interrupt Priority Control Register 11	IP11L	—		00
010C☆	Interrupt Request Register 2	IRQ2L	—		00
010D☆	Interrupt Enable Register 2	IE2L	—		00
010E☆	Interrupt Priority Control Register 20	IP20L	—		00
010F☆	Interrupt Priority Control Register 21	IP21L	—		00
0108☆	NMI Control Register	NMICON	—		FC or 7C
0109☆	External Interrupt Control Register	EXICON	—		00

Notes: 1. Addresses do not run consecutively.
2. Addresses in the address column marked by stars indicate that the register has bits missing.

17.1 Non-Maskable Interrupt (NMI)

The NMI interrupt cannot be masked. When the effective edge, specified by NMICON bit 0 and bit 1, is detected, the CPU will immediately shift to the NMI interrupt process.

However, there is a single exception. The NMI interrupt will be masked after reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap) until the first instruction has completely executed. This function is provided to remove the possibility that an NMI interrupt could be accepted before the System Stack Pointer (SSP) has an appropriate value after reset. The program can eliminate that possibility by using an instruction that sets SSP to an appropriate value as the first instruction after reset.

The following process will be performed in hardware automatically when an NMI interrupt occurs.

- Save Program Counter (PC).
- Save Accumulator (ACC).
- Save Local Register Base (LRB).
- Save Program Status Word (PSW).
- Reset NMI Interrupt Request Flag.
- Disable acceptance of maskable interrupts.
- Disable a multiple occurrence of the NMI interrupt itself.
- Load the value written in the vector table (0008H, 0009H) into the program counter.

Fourteen cycle are required to shift to each of these NMI process. However, if the program memory space is 1MB, 17 cycle are required because cycles required to save the code segment register (CSR) are added.
An RTI instruction is needed at the end of the NMI interrupt routine. The following process will be performed in hardware automatically when the RTI instruction is executed.

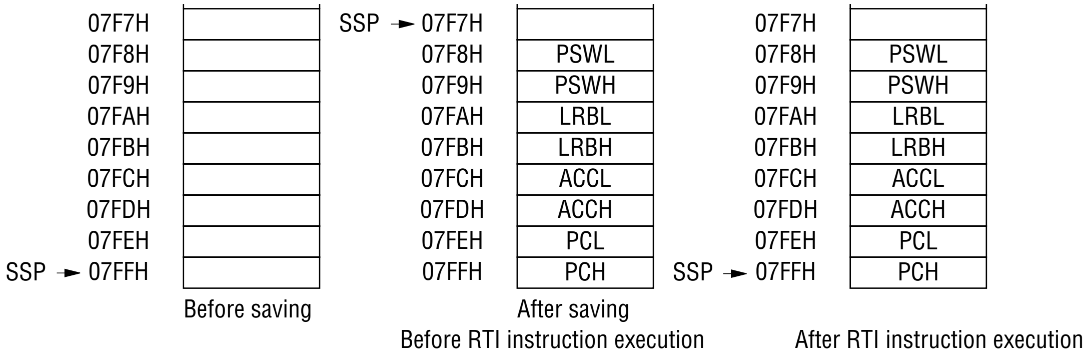
- Restore Program Status Word (PSW).
- Restore Local Register Base (LRB).
- Restore Accumulator (ACC).
- Restore Program Counter (PC).
- Enable acceptance of maskable interrupts.
- Enable further NMI interrupts.

This completes the NMI interrupt routine.

Note: When program memory space has been expanded to 1MB, the Code Segment Register (CSR) will be saved and restored in addition to the other registers listed above.

Figure 17-1 shows examples of saving and restoring PC, ACC, LRB, and PSW.

- Interrupt process (when program memory space is 64KB)



- Interrupt process (when program memory space is 1MB)

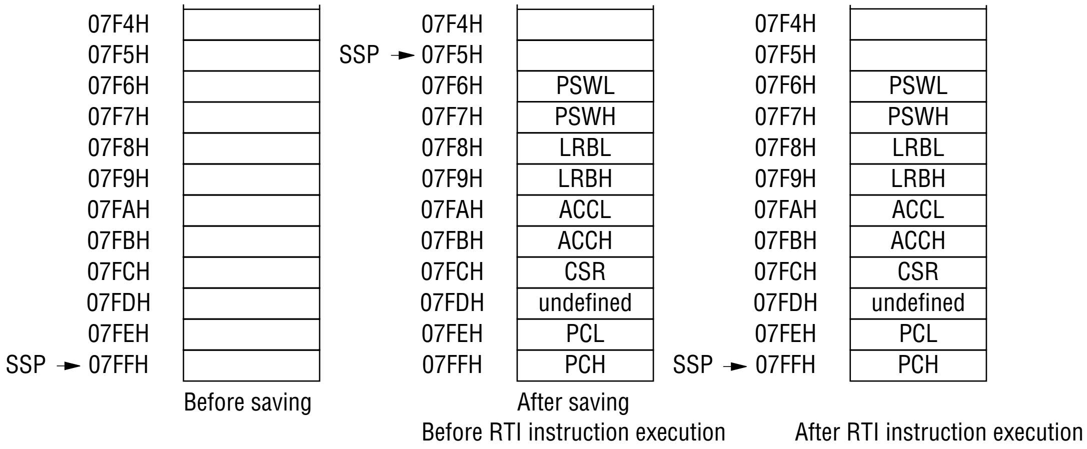


Figure 17-1 Examples of Saving and Restoring PC, ACC, LRB and PSW

17.2 Maskable Interrupts

Maskable interrupts are generated by internal peripheral hardware and external pin inputs. Maskable interrupt control is performed with the following.

- Registers set when an interrupt request is generated by each interrupt factor (IRQ)
- Registers that enable interrupts provided by each factor (IE)
- Flag that enables/disables interrupt generation from all maskable interrupts (MIE)
- Flag that enables/disables all priorities (MIP)
- Registers that set priority levels (IP)

Figure 17-2 shows a conceptual diagram of maskable interrupt control. Table 17-2 lists the maskable interrupt vector addresses and bit symbols.

[One interrupt vector for one interrupt factor]

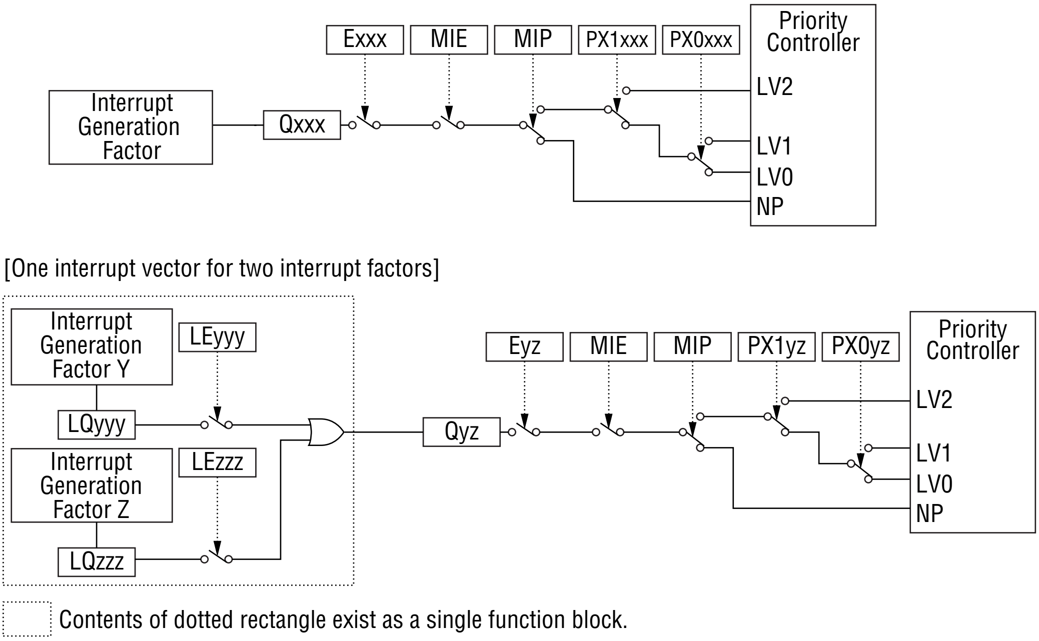

Exxx
MIE
MIP
PX1xxx
PX0xxx
Priority Controller
LV2
LV1
LV0
NP
Interrupt Generation Factor
Qxxx
[One interrupt vector for two interrupt factors]
Interrupt Generation Factor Y
LEyyy
LQyyy
Interrupt Generation Factor Z
LEzzz
LQzzz
Qyz
Eyz
MIE
MIP
PX1yz
PX0yz
Priority Controller
LV2
LV1
LV0
NP
Contents of dotted rectangle exist as a single function block.

Figure 17-2 Conceptual Diagram of Maskable Interrupts

Table 17-2 Maskable Interrupt Vector Addresses and Bit Symbols

bit	Interrupt Factor	Vector Addres	IRQ	IE	IPX0	IPX1
0	External interrupt 0 (INT0)	000A	QINT0	EINT0	P0INT0	P1INT0
2	TM1 overflow	000E	QTM1OV	ETM1OV	P0TM1OV	P1TM1OV
11	RTO8 event generation	0020	QRTO8	ERTO8	P0RTO8	P1RTO8
12	RTO9 event generation	0022	QRTO9	ERTO9	P0RTO9	P1RTO9
15	Serial port transmit/receive ready	0028	QSCI1	ESCI1	P0SCI1	P1SCI1
16	S1TM overflow	002A	QSTMOV	ESTMOV	P0STMOV	P1STMOV
17	GTMC overflow	002C	QGTMOV	EGTMOV	P0GTMOV	P1GTMOV
19	A/D conversion finished	0030	QAD	EAD	P0AD	P1AD
20	PWC0 underflow or match	0032	QPW01	EPW01	P0PW01	P1PW01
23	External interrupt 1 (INT1)	0038	QINT1	EINT1	P0INT1	P1INT1
34	External interrupt 2 (INT2)	006E	QINT2	EINT2	P0INT2	P1INT2
35	External interrupt 3 (INT3)	0070	QINT3	EINT3	P0INT3	P1INT3

(1) Interrupt Request Registers IRQ (IRQ0L, IRQ0H, IRQ1L)

IRQ are registers that save interrupt requests. Each IRQ becomes "1" when an interrupt signal is generated from an interrupt source, and becomes 0 during the interrupt transition cycle when the interrupt is accepted. IRQ bits can also be set to "0" or "1" by the program.

(2) Interrupt Enable Registers IE (IE0L, IE0H, IE1L)

IE are registers that enable/disable generation of individual interrupts. When an IE bit is "0", generation of the corresponding interrupt will be disabled. When an IE bit is "1", generation of all the interrupts independently enabled by IE will be enabled.

(3) Master Interrupt Enable Flag MIE

MIE is a 1-bit flag in PSW. MIE enables/disables generation of all maskable interrupts. When MIE is "0", generation of all maskable interrupts will be disabled. When MIE is "1", generation of all the interrupts independently enabled by IE will be enabled.

(4) Master Interrupt Priority Flag MIP

MIP is a 1-bit flag in PSW. MIP enables/disables all maskable interrupt priorities. When MIP is "0", priorities set by IPX0 and IPX1 will be disabled, so interrupt generation will be controlled by IE and MIE only. When MIP is "1", all maskable interrupts are given priority levels 0-3 by IPX0 and IPX1.

(5) Interrupt Priority Control Registers
IPX0 (IP00L, IP00H, IP10L)
IPX1 (IP01L, IP01H)

IPX0 and IPX1 specify maskable interrupt priorities. When a maskable interrupt's corresponding IPX0 and IPX1 bits are PX0xxx and PX1xxx, the priority given to that maskable interrupt will be as follows.

PX1xxx	PX0xxx	Priority	
0	0	Level 0	(low)
0	1	Level 1	↕
1	*	Level 2	(high)

* indicates "0" or "1"

17.3 Maskable Interrupt Operation

Generation of an interrupt signal causes an interrupt request to be passed to interrupt priority selection logic by the following operation.

- An interrupt request signal is output from the interrupt source.
- When the IRQ bit becomes "1", if either the corresponding IE bit or MIE in PSW is "1", then the interrupt will be held off. When the IRQ bit becomes "1", if both the corresponding IE bit and MIE in PSW is "1", then the interrupt priority selection logic will operate.

The interrupt priority selection logic operates as follows.

- If MIP in PSW is "0", then the interrupt priority selection logic will not operate, and the interrupt will be generated immediately. If MIP in PSW is "1", then the interrupt will be given a priority from one of three levels (level 0 to level 2) by the corresponding IPX0 and IPX1 bits.
- If no other interrupt is being executed at the time the interrupt request is sent to the interrupt priority selection logic, then the interrupt will be generated immediately.
- If one or more other interrupts are already being executed at the time the interrupt request is sent to the interrupt priority selection logic, and if the new interrupt has a higher priority level than any of the executing interrupts, then the interrupt will be generated immediately.
- If one or more other interrupts are already being executed at the time the interrupt request is sent to the interrupt priority selection logic, and if the new interrupt has a lower priority level than any of the executing interrupts, then the interrupt will be held off.

Generated Interrupt / Process Being Executed	Level 0 Interrupt Request	Level 1 Interrupt Request	Level 2 Interrupt Request	NMI Interrupt Request
No interrupt process	❍ : Interrupt generated	❍ : Interrupt generated	❍ : Interrupt generated	❍ : Interrupt generated
Level 0 interrupt process	× : Interrupt held	❍ : Interrupt generated	❍ : Interrupt generated	❍ : Interrupt generated
Level 1 interrupt process	× : Interrupt held	×: Interrupt held	❍:Interrupt generated	❍ : Interrupt generated
Level 2 interrupt process	×: Interrupt held	× : Interrupt held	× : Interrupt held	❍ : Interrupt generated
NMI interrupt process	× : Interrupt held	× : Interrupt held	× : Interrupt held	× : Interrupt held

Note: When interrupts are generated simultaneously, the one with higher priority will have precedence. When interrupts of the same priority are generated simultaneously, the one with the lower interrupt vector address will have precedence.

The following process will be performed in hardware automatically when a maskable interrupt occurs.

- Save Program Counter (PC).
- Save Accumulator (ACC).
- Save Local Register Base (LRB).
- Save Program Status Word (PSW).
- Reset the IRQ that generated the maskable interrupt process.
- Reset MIE in PSW (disable acceptance of maskable interrupts).
- Record interrupt priority level (disable acceptance of interrupts with equal or lower priority).
- Load the value written in the vector table into the program counter.

Fourteen cycle are required to shift to each maskable interrupt process. However, if the program memory space is 1MB, 17 cycles are required because cycles required to save the code segment register (CSR) are added.

An RTI instruction is needed at the end of the maskable interrupt routine. The following process will be performed in hardware automatically when the RTI instruction is executed.

- The highest priority level in the stored interrupt levels is deleted.
- Restore Program Status Word (PSW), also setting MIE to 1.
- Restore Local Register Base (LRB).
- Restore Accumulator (ACC).
- Restore Program Counter (PC).

This completes the maskable interrupt routine.

Note: When program memory space has been expanded to 1MB, the Code Segment Register (CSR) will be saved and restored in addition to the other registers listed above.

Figure 17-1 shows an example of saving and restoring PC, ACC, LRB, and PSW.

Chapter 18

Bus Port Functions

18. Bus Port Functions

External program memory (normally ROM) can be expanded up to 1MB. External data memory (normally RAM) can be expanded up to 1MB.

Port 0 (P0), Port 1 (P1), and Port 9 (P9_0-P9_3) are used as bus ports to access both external program memory and external data memory.

18.1 Bus Port (P0, P1, P9_0-P9_3) Functions

P0, P1, and P9_0-P9_3 serve as input/output ports for their primary function, and as bus ports for their secondary function.

Switching of P0 between its primary and secondary function is performed automatically depending on the object to be accessed (internal or external memory, program or data memory).

Switching of P1 and P9_0-P9_3 between their primary and secondary function is performed automatically when the $\overline{\text{EA}}$ pin is low, but their secondary registers must be set when the $\overline{\text{EA}}$ pin is high.

18.1.1 P0, P1, P9_0-P9_3 Operation During Program Memory Access

When internal program memory is accessed (when $\overline{\text{EA}}$ pin is high) P0, P1, and P9_0-P9_3 operate as input/output ports.

When external program memory is accessed (when $\overline{\text{EA}}$ pin is low) P0 operates as the port for low address outputs and program data input, and P1 and P9_0-P9_3 operate as the ports for high address outputs.

Table 18-1 lists the operations of P0, P1, and P9_0-P9_3 during program memory accesses.

Table 18-1 P0, P1, P9_0-P9_3 Operation During Program Memory Accesses

$\overline{\text{EA}}$ pin	Access Object		P0 Operation	P1, P9_0-P9_3 Operation
	Memory	Address		
High	Internal program	Note 1	Input/output port	
Low	External program	00000H-FFFFDH	Low address outputs and program data input	High address outputs

Note 1. The external program memory area is 10000H-FFFFDH. To access this area, the secondary registers of P1 and P9 (P1SF, P9SF) must be set.

18.1.2 P0, P1, P9_0-P9_3 Operation During Data Memory Access

When internal data memory is accessed (when access object is 0000H-9FFH) P0, P1 and P9_0-P9_3 operate as input/output ports.

When external data memory is accessed (access object is A00H-FFFFFH) P0 operates as the port for low address outputs and data input/output. P1 and P9_0-P9_3 operates as the port for high address outputs only for pins that have been set ("1") in the Port 1 and Port 9 Secondary Control Registers. P1 and P9_0-P9_3 operates as an input/output port for pins that have been reset ("0") in the Port 1 and Port 9 Secondary Control Registers. However, if the $\overline{EA}$ pin is low, then P1 and P9_0-P9_3 will operate as the port for high address outputs regardless of the Port 1 and Port 9 Secondary Control Registers.

Table 18-2 lists the operations of P0, P1 and P9_0-P9_3 during data memory accesses.

Table 18-2 P0, P1 , P9_0-P9_3 Operation During Data Memory Accesses

Access Object		**P0**	**P1, P9_0-P9_3**
Memory	Address	Operation	Operation When Secondary Function Set
Internal data	0000H-9FFH	Input/output port	
External data	A00H-FFFFFH	Low address outputs and data input/output	High address outputs

18.2 External Memory Access

18.2.1 External Program Memory Access

Program space up to 1MB (00000H-FFFFDH) can be accessed by the 16-bit Program Counter (PC) and 4-bit Code Segment Register (CSR). When the $\overline{\text{EA}}$ pin has been set to a high level, external program memory is accessed at program addresses 10000H-FFFFDH. When the $\overline{\text{EA}}$ pin has been set to a low level, external program memory is accessed at program addresses 00000H-FFFFDH.

When $\overline{\text{EA}}$ is set high and 10000H-FFFFDH (for MSM66586, C000-FFFFDH) is used for external program memory, the Port 1 Secondary Control Register and Port 9 Secondary Control Register (high address outputs) need to be set. The Port 5 Secondary Control Register (ALE and $\overline{\text{PSEN}}$ outputs) also needs to be set.

Figure 18-1 shows a external program memory (ROM) interface example

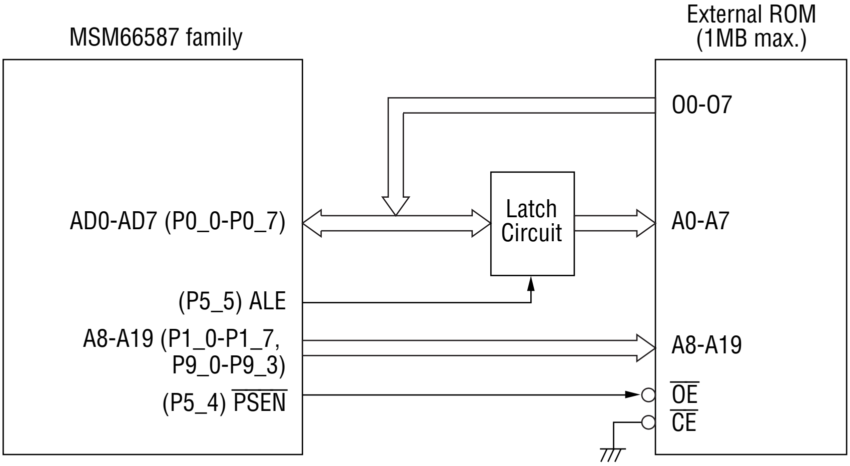


Figure 18-1 External ROM Interface Example

18.2.2 External Program Memory Access Timing

Figure 18-2 and Figure 18-3 show external program memory access timing diagrams.

A function for inserting wait cycles (see Section 5.2, "Ready Function") is provided for external memory with slow access time. It should be implemented to match the access time of the external memory used. However, relative to internal program memory accesses, one cycle is inserted automatically for each byte accessed during external program memory accesses. ROMRDY is a register that specifies further cycles to be inserted in addition to the one cycle inserted automatically.

Refer to Chapter 20, "Electrical Characteristics," for actual AC characteristics.

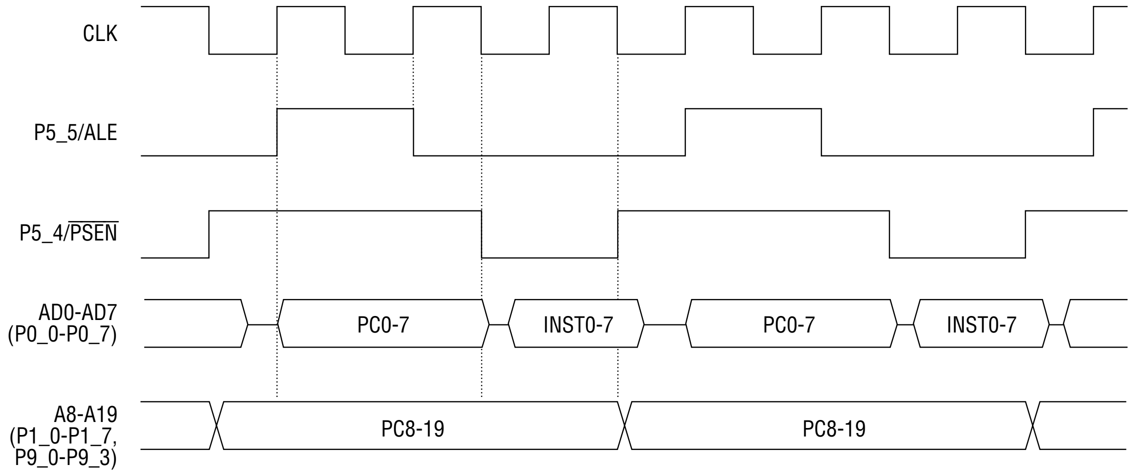


Figure 18-2 External Program Memory Access Timing Diagram (No Waits)

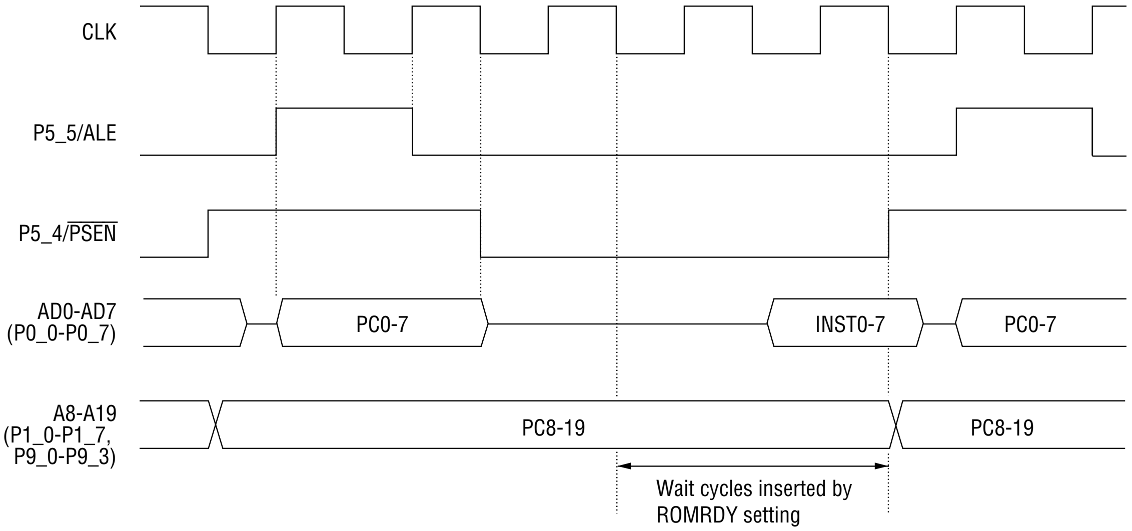


Figure 18-3 External Program Memory Access Timing Diagram (2 Wait Cycles)

18.2.3 External Data Memory Access

Data space up to 1MB (00000H-FFFFFH) can be accessed by the 16-bit Program Counter (PC) and 4-bit Data Segment Register (DSR). When data addresses 0000H-9FFH are accessed, internal data memory will be accessed. When data addresses A00H-FFFFFH are accessed, external data memory will be accessed. However, external data memory will be accessed in 8-bit units (bytes).

To use external data memory, the Port 1 and Port 9 Secondary Control Registers (high address outputs) must be set in accordance with external memory addresses. Also, the Port 5 Secondary Control Register bit 5 (ALE output) and Port 7 Secondary Control Register bit 0 and bit 1 ($\overline{\text{WR}}$ and $\overline{\text{RD}}$ outputs) must be set ("1"). However, when the $\overline{\text{EA}}$ pin is low, P1, P9 and P5_5 will switch to high address outputs and ALE output regardless of the Secondary Control Registers.

Add external pull-up resistors to the $\overline{\text{WR}}$ and $\overline{\text{RD}}$ pins as necessary.

Figure 18-4 shows a data memory (RAM) interface example.

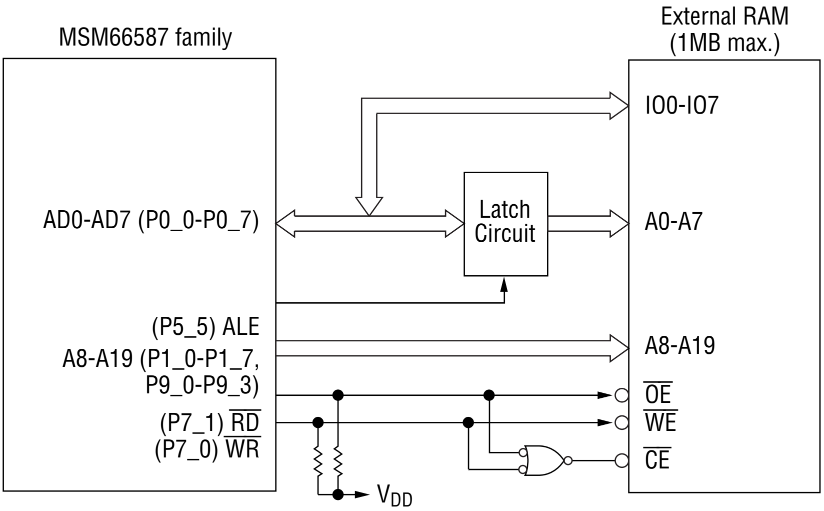


Figure 18-4 External RAM Interface Example

18.2.4 External Data Memory Access Timing

Figure 18-5 and Figure 18-6 show external data memory access timing diagrams.

A function for inserting wait cycles (see Section 5.2, “Ready Function”) is provided for external memory with slow access time. It should be implemented to match the access time of the external memory used. However, relative to internal data memory accesses, three or four cycles are inserted automatically for each byte accessed during external data memory accesses. RAMRDY is a register that specifies further cycles to be inserted in addition to the three or four cycles inserted automatically.

Refer to Chapter 20, “Electrical Characteristics,” for actual AC characteristics.

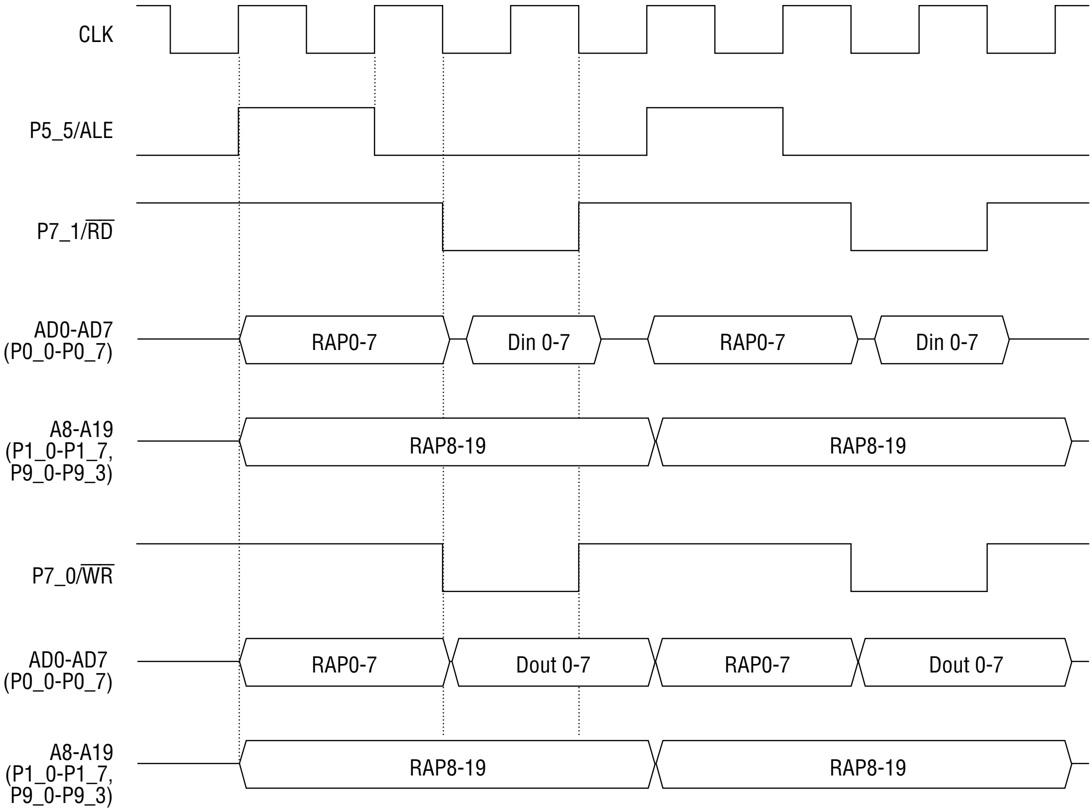


Figure 18-5 External Data Memory Access Timing Diagram (No Waits)

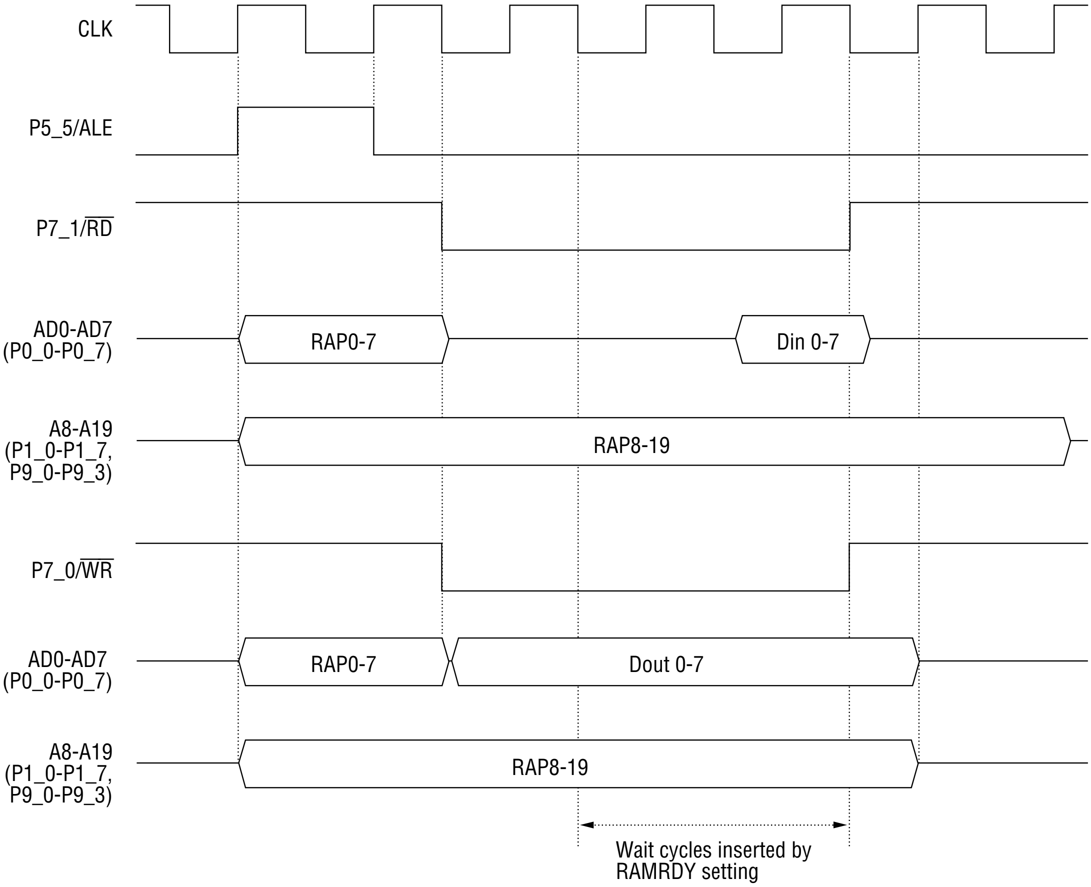


Figure 18-6 External Data Memory Access Timing Diagram (2 Wait Cycles)

Chapter 19

Flash Memory

19. Flash Memory

19.1 Summary

The MSM66Q587 replaces the internal program memory (64KB mask-ROM) of the MSM66587 with an electrically erasable and reprogrammable non-volatile memory (64KB flash memory). With two types of flash memory reprogramming modes, the program memory can be reprogrammed even after installation in the system.

19.2 Features

- Power supply voltage

  The flash memory is reprogrammable and read by a single 5 V power supply (4.5 to 5.5 V).

- Reprogramming modes

  There are two types of flash memory reprogramming modes.
  - Serial mode (reprogramming is possible before and after installation on the circuit board)
  - User mode (reprogramming is possible after installation on the circuit board)

- Erasure block

  The flash memory is erased in 1KB blocks (erasure of a sector).

- Reprogramming and erasing time

  The reprogramming and erasing time of the flash memory is as follows:
  - Page reprogramming (64-byte units) ... 3 ms
  - Sector erasure (1KB units) .... 10 ms

- Write protect function

  The flash memory contains a built-in supply voltage sense and reset function that automatically triggers a reset if the supply voltage drops to 2 V or lower, and a built-in power-on delay function that automatically inhibits reprogramming for approximately 20 ms when power is turned on.

  In addition, there is an acceptor function to prevent accidental programming or erasing if the program runs out of control when in the user mode.

19.3 Reprogramming Modes

The flash memory of the MSM66Q587 has the following two reprogramming modes:

(1) Serial Mode

In this mode, reprogramming is performed on the system or using a single microprocessor. The serial mode is used mainly for the development of user software.

In the serial mode, reprogramming is performed while the microprocessor is being reset or is in the stop mode. OKI flash memory writer (model no. PW66K) is used to perform the reprogramming.

(2) User Mode

In this mode, reprogramming is performed by the system. The user mode is used mainly in the development of user software. This mode is also used for final adjustments after product assembly, or for program maintenance after shipping.

In the user mode, reprogramming is performed by microprocessor operation, a program for reprogramming must be stored in advance in ROM using the serial mode.

Figure 19-1 shows a block diagram of the flash memory reprogramming modes.

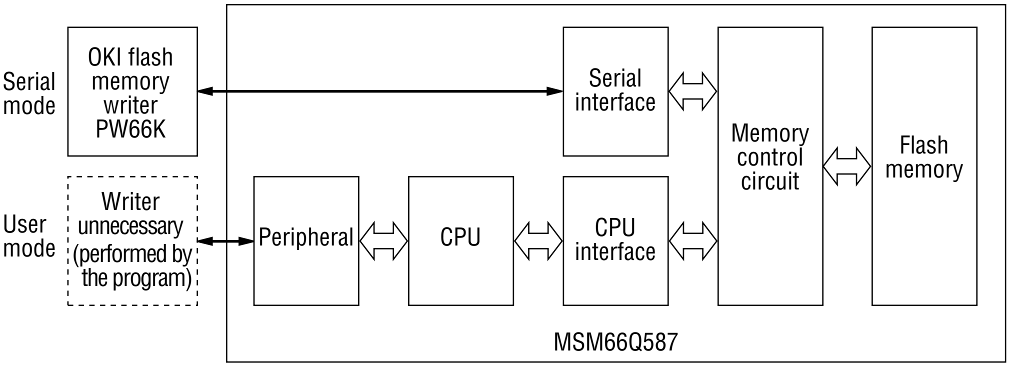


Figure 19-1 Reprogramming Modes Block Diagram

19.4 Serial Mode

19.4.1 Serial Mode Summary

In the serial mode, reprogramming is possible either after installation on a circuit board or using a single microprocessor.

A flash memory writer (PW66K) dedicated for use with the serial mode is used to perform the reprogramming.

Reprogramming is performed in the serial mode while the CPU is being reset or is in the STOP mode.

19.4.2 Setting the Serial Mode

Specified pins are connected to the flash memory writer. The serial mode is automatically set by performing a reprogramming or read operation. When the write or read operation is completed, the setting is canceled.

(1) Pins Used for Serial Mode

Table 19-1 lists the pins used in the serial mode.

The serial mode can only be set while the CPU is being reset or is in the STOP mode. A high voltage from the flash memory writer is applied to the $\overline{EA}$ pin to set the serial mode. Therefore, be careful of this pin. V_{DD} is connected so that the V_{DD} of the user's system can be monitored.

Table 19-1 Pins Used for Serial Mode

Pin Number	Primary Function Name	Flash Memory Function Name
32	P8_6	FLACLK (serial clock input)
33	P8_7	FLADAT (serial data I/O)
36	$\overline{EA}$	FLAMOD (high voltage input to set serial mode)
9, 17, 37, 67, 93	V_{DD}	Monitoring V_{DD} of user's system
16, 40, 68, 94	GND	Ground

Note: During the serial mode, a high voltage that is greater than the supply voltage (approximately 11 V) is applied to the $\overline{EA}$ pin.

(2) Serial Mode Connections

With the serial mode, it is necessary to connect the flash memory writer (PW66K) to pins P8_6, P8_7, $\overline{EA}$, V_{DD} and GND of the MSM66Q587 of the user's system. Install a switch on the user's system to isolate the flash memory when reprogramming and reading in the serial mode.

Figure 19-2 shows serial mode connection circuit example 1.

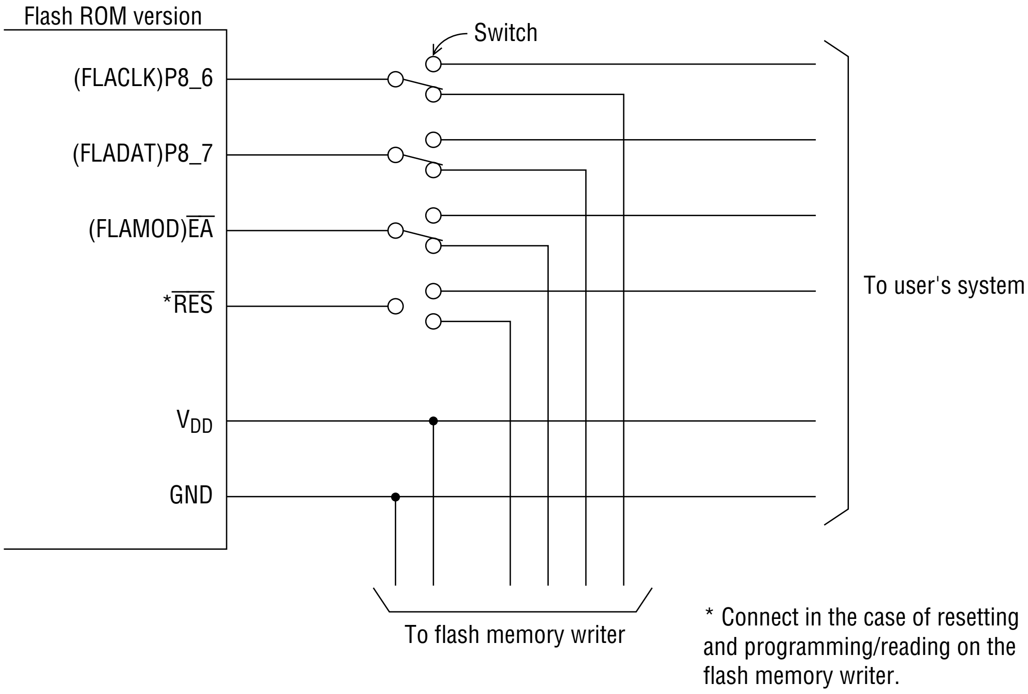


Figure 19-2 Serial Mode Connection Circuit Example 1

If a switch cannot be installed on the user’s system, instead of using the P8_6 and P8_7 pins with the user’s system, connect them only to the flash memory writer. Also, connect each of the P8_6, P8_7, and $\overline{EA}$ pins to V_{DD} through a resistor of approximately 100 kΩ.

Figure 19-3 shows the serial mode connection circuit example 2.

Note

The programming and reading in the serial mode are performed while the microcontroller is in the reset or STOP mode. To execute the programming/reading during reset, apply "L" level to the $\overline{RES}$ pin. In the case where the flash memory writer does not apply "L" level to the $\overline{RES}$ pin, the "L" level should be applied by the user application system.

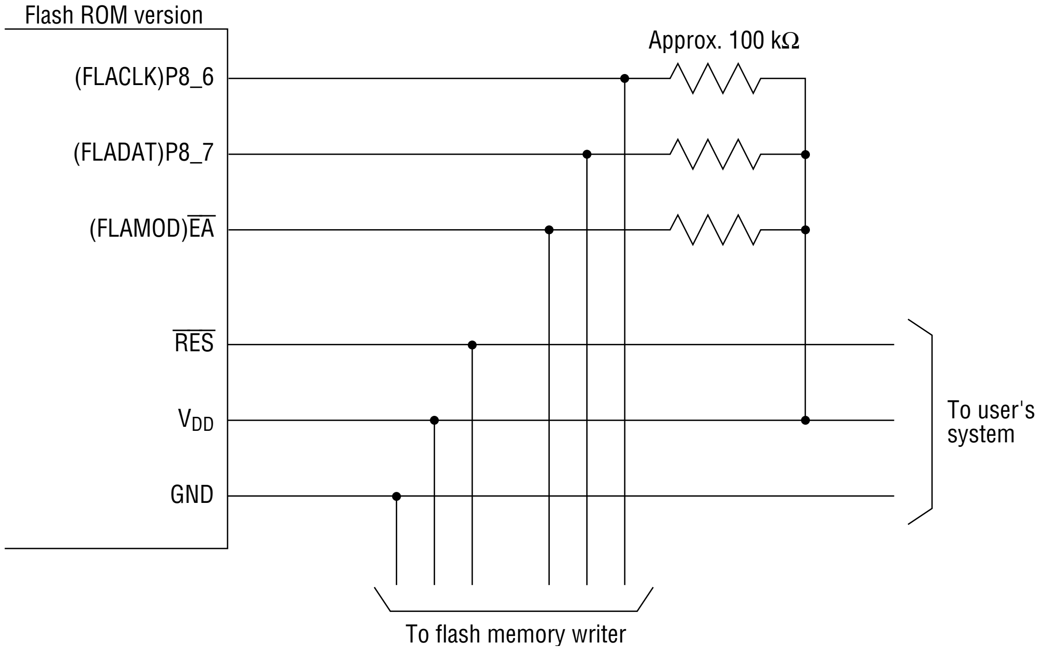


Figure 19-3 Serial Mode Connection Circuit Example 2

(3) Serial Mode Reprogramming Method

The serial mode reprograms flash memory with the flash memory writer (PW66K). If reprogramming is to be performed with only a single microprocessor, use the attached FWADP66587 write adapter.

The procedure for reprogramming with the flash memory writer is listed below.
Refer to the PW66K User’s Manual for details.

1) Connect the flash memory writer to pins P8_6, P8_7, $\overline{\text{EA}}$, V_{DD} and GND.

2) Set the microprocessor to RESET or STOP mode.
- A flash memory writer protocol error will occur if not in the reset or STOP modes.

3) Perform the reprogram or read operation with the flash memory writer.
- The serial mode will automatically be set.

4) Verify that normal operation of the flash memory writer has ended.
- The serial mode is automatically canceled.

5) Cancel RESET or STOP mode.
- The CPU executes the program reprogrammed into the flash memory.

(4) Notes Regarding Serial Mode Usage

If reprogramming while in the STOP mode, during the reprogramming operation, do not generate an interrupt or do not initiate a reset using $\overline{\text{RES}}$ pin input. (After the serial mode is canceled, the CPU may run out of control.)

If an interrupt or reset has been initiated, reprogram the program in flash memory.

19.5 User Mode

19.5.1 User Mode Summary

The user mode reprograms the flash memory from the user’s system. Reprogramming is possible after the microprocessor is mounted on the user’s system.

The user mode executes program to reprogram flash memory. The program is devised in advance to contain commands that execute reprogramming and a method of I/O for the reprogramming data.

It is necessary for the program to be written into the flash memory in advance in the serial mode or in the parallel mode.

Figure 19-4 shows a block diagram of the user mode.

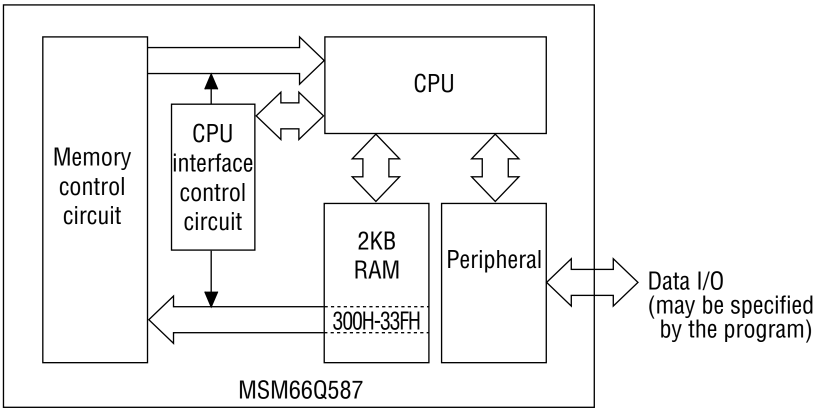

CPU
Memory control circuit
CPU interface control circuit
2KB RAM
300H-33FH
Peripheral
Data I/O (may be specified by the program)
MSM66Q587

Figure 19-4 User Mode Block Diagram

19.5.2 User Mode Reprogramming

The MSM66Q587 contains special function registers (SFRs) for reprogramming with the user mode.

The flash memory control register (FLACON), flash memory address register (FLAADRS), and flash memory acceptor (FLAACP) are controlled to perform reprogramming operations in the user mode.

Table 19-2 lists a summary of the user mode SFRs.

Table 19-2 User Mode SFRs

Address (H)	Name	Abbreviation (BYTE)	Abbreviation (WORD)	R/W	8/16 operation	Reset status
01F0☆	Flash memory acceptor	FLAACP	—	W	8	"0"
01F1☆	Flash memory control register	FLACON	—	R/W	8	0E4
01F2☆	Flash memory address register	—	FLAADRS	R/W	16	Undefined
01F3						

Note: The symbol ☆ in the above addresses indicates a missing bit.

(1) Flash Memory Address Register (FLAADRS)

Bits 6 to 15 (FA6 to 15) of the FLAADRS register set the address of reprogramming or sector erasure for the flash memory when performing reprogramming or sector erasure.

Figure 19-5 shows the FLAADRS configuration.

FLAADRS

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FA15	FA14	FA13	FA12	FA11	FA10	FA9	FA8	FA7	FA6	—	—	—	—	—	—

Address: 01F2H
R/W access: R/W

When Performing Reprogramming

FA15-6										Reprogramming address
15	14	13	12	11	10	9	8	7	6	
0	0	0	0	0	0	0	0	0	0	0000H-003FH
0	0	0	0	0	0	0	0	0	1	0040H-007FH
0	0	0	0	0	0	0	0	1	0	0080H-00BFH
1	1	1	1	1	1	1	1	1	0	0FF80H-0FFBFH
1	1	1	1	1	1	1	1	1	1	0FFC0H-0FFFFH

When Performing Sector erasure

FA15-6										Sector erasure address
15	14	13	12	11	10	9	8	7	6	
0	0	0	0	0	0	✱	✱	✱	✱	0000H-03FFH
0	0	0	0	0	1	✱	✱	✱	✱	0400H-07FFH
0	0	0	0	1	0	✱	✱	✱	✱	0800H-0BFFH
1	1	1	1	1	0	✱	✱	✱	✱	0F800H-0FBFFH
1	1	1	1	1	1	✱	✱	✱	✱	0FC00H-0FFFFH

Note: ✱ indicates "0" or "1".

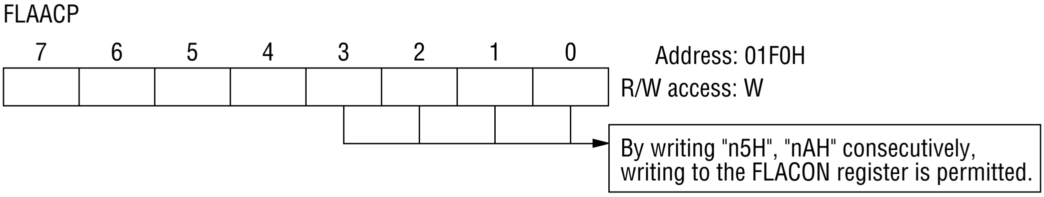
"—" indicates bits that do not exist. When read, read as "1".

Figure 19-5 FLAADRS Configuration

(2) Flash Memory Acceptor (FLAACP)

The FLAACP is an acceptor used when data is set in the flash memory control register. By writing the values n5H and nAH (n = 0 to F) consecutively, the acceptor is set to "1". It is reset to "0" when the flash memory is reprogrammed or erased.

The FLAACP configuration is shown in figure 19-6.



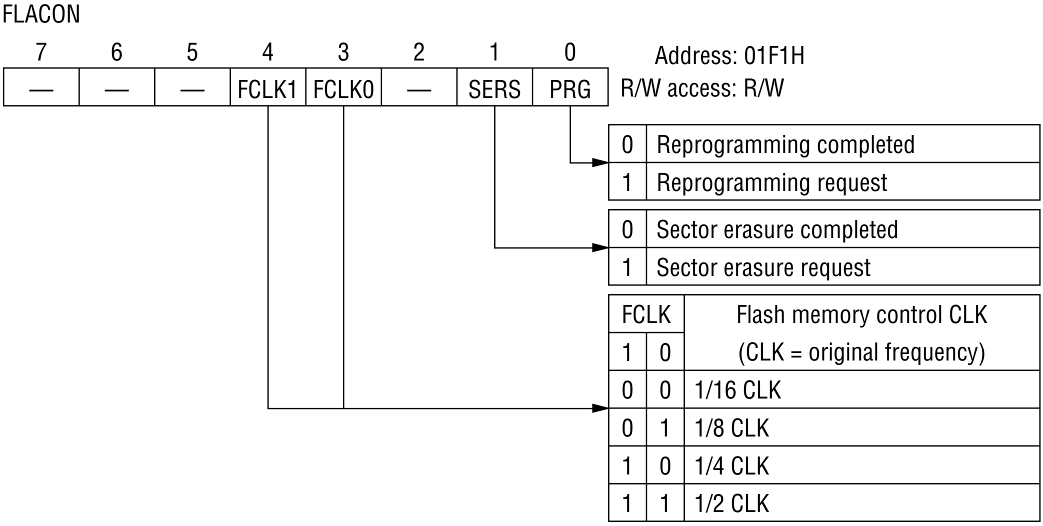
By writing "n5H", "nAH" consecutively, writing to the FLACON register is permitted.

Figure 19-6 FLAACP Configuration

(3) Flash Memory Control Register (FLACON)

FLACON is a 4-bit register used to control reprogramming and erasing of the flash memory.

The FLACON configuration is shown in figure 19-7.



7	6	5	4	3	2	1	0
—	—	—	FCLK1	FCLK0	—	SERS	PRG

PRG	
0	Reprogramming completed
1	Reprogramming request

SERS	
0	Sector erasure completed
1	Sector erasure request

FCLK		Flash memory control CLK
1	0	(CLK = original frequency)
0	0	1/16 CLK
0	1	1/8 CLK
1	0	1/4 CLK
1	1	1/2 CLK

"—" indicates bits that do not exist. When read, read as "1".

Figure 19-7 FLACON Configuration

Writing to the FLACON register is enabled after n5H and nAH (n = 0 to F) are written consecutively to the FLAACP and the flash memory acceptor is set to “1”. The FLAACP register is reset to “0” after reprogramming or erasing flash memory. When another reprogramming or erasing operation is performed on flash memory, the flash memory acceptor must be set to “1” again.

- PRG (bit 0)

After PRG (bit 0) of the FLACON register is set to “1” and one instruction is executed, the CPU enters a HOLD state, and the data in the RAM address area of 300H to 33FH is transferred to the flash memory and the reprogramming is executed.

When the reprogramming is complete, the HOLD state is canceled and this bit is reset to “0”. Previously set the reprogramming address and data to the FLAADRS register and the internal RAM address area of 300H to 33FH.

Notes: If an interrupt occurs during reprogramming the flash memory, the interrupt processing will be postponed until after the reprogramming is complete.
While reprogramming the flash memory, if the $\overline{\text{RES}}$ pin input asserts a reset, the reset processing will be postponed until after the reprogramming is complete.

Figure 19-8 shows the relation between internal RAM and flash memory.

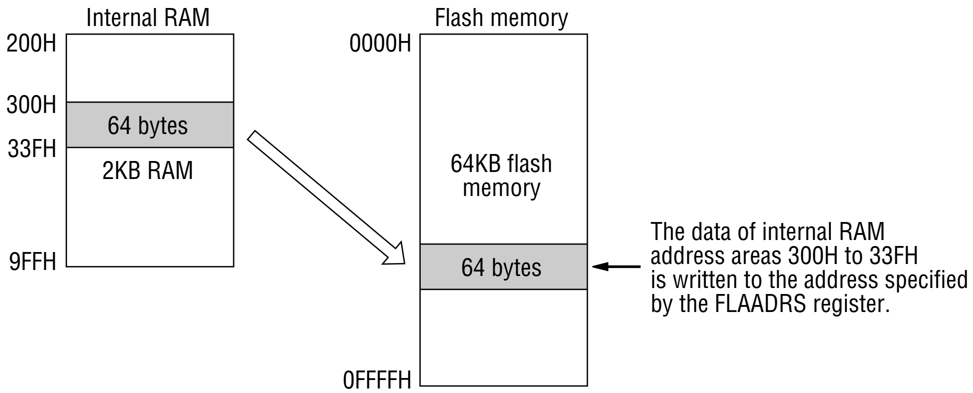

Internal RAM
200H
300H
64 bytes
33FH
2KB RAM
9FFH
Flash memory
0000H
64KB flash memory
64 bytes
0FFFFH
The data of internal RAM address areas 300H to 33FH is written to the address specified by the FLAADRS register.

Figure 19-8 Relation between Internal RAM and Flash Memory

- SERS (bit 1)

After SERS (bit 1) of the FLACON register is set to “1” and one instruction is executed, the CPU enters a HOLD state, and a sector of flash memory is erased (1KB).

When the erasing is complete, the HOLD state is canceled and this bit is reset to “0”. Set the address to be erased to the FLAADRS register in advance.

Notes: If an interrupt occurs while the flash memory is being erased, the interrupt processing is postponed until after the erase operation is complete.
While erasing the flash memory, if the $\overline{\text{RES}}$ pin input asserts a reset, the reset processing will be postponed until after the erase operation is complete.

- FCLK0, 1 (bits 3, 4)

FCLK0 (bit 3) and FCLK1 (bit 4) of the FLACON register are bits to set the clock division ratio (flash memory control CLK) to control the necessary wait time (CPU hold time) when erasing or reprogramming the flash memory. At reset, since FCLK0 and FCLK1 are cleared to “0”, 1/16 CLK is selected as the flash memory control CLK. Flash memory erasure and reprogramming times for each original frequency are shown in tables 19-3 and 19-4 respectively.

Table 19-3 Flash Memory Erasure Time

FCLK		Erasure time (msec)						
1	**0**	**CLK = 5 MHz**	**CLK = 10 MHz**	**CLK = 12 MHz**	**CLK = 14 MHz**	**CLK = 16 MHz**	**CLK = 18 MHz**	**CLK = 20 MHz**
0	0	25.0	17.0	15.7	14.7	14.0	13.4	13.0
0	1	17.0	13.0	12.3	11.9	11.5	11.2	11.0
1	0	13.0	11.0	10.7	10.4	10.3	10.1	10.0
1	1	11.0	10.0	9.8	9.7	9.6	9.6	9.3

Set FCLK0 and FCLK1 such that the flash memory erase time is 10 msec or greater (all of table 19-3 except for the shaded sections). The formula for calculating erasure time, t (sec), is shown below.

$$t = n \times \frac{1}{f\ \text{Hz}} \times 5000 + 0.009 \qquad \text{(n: specified by FCLK)}$$

Table 19-4 Flash Memory Reprogramming Time

FCLK		Reprogramming time (msec)						
1	**0**	**CLK = 5 MHz**	**CLK = 10 MHz**	**CLK = 12 MHz**	**CLK = 14 MHz**	**CLK = 16 MHz**	**CLK = 18 MHz**	**CLK = 20 MHz**
0	0	18.0	10.0	8.7	7.7	7.0	6.4	6.0
0	1	10.0	6.0	5.3	4.9	4.5	4.2	4.0
1	0	6.0	4.0	3.7	3.4	3.3	3.1	3.0
1	1	4.0	3.0	2.8	2.7	2.6	2.6	2.5

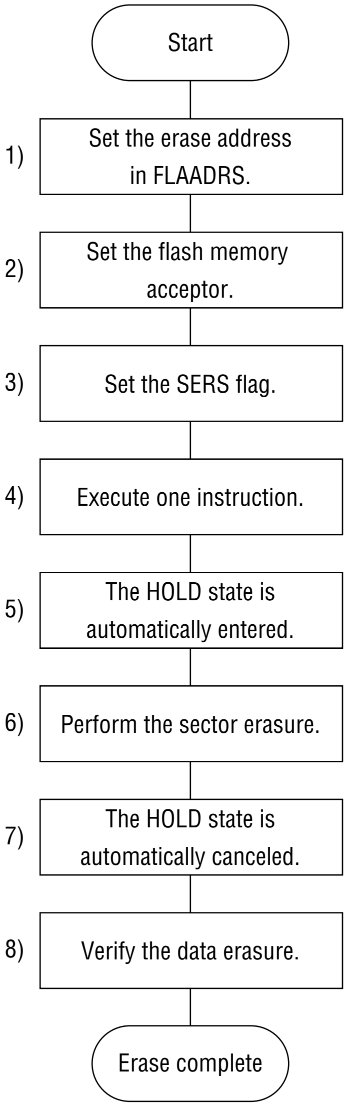
Set FCLK0 and FCLK1 such that the flash memory reprogramming time is 3 msec or greater (all of table 19-4 except for the shaded sections). The formula for calculating reprogramming time, t (sec), is shown below.

$$t = n \times \frac{1}{f\ \text{Hz}} \times 5000 + 0.002 \qquad \text{(n: specified by FCLK)}$$

(4) Flowchart Examples for User Mode

Reprogramming with the user mode first erases the flash memory with a sector erasure, and then performs the write operation in 64-byte units.

Figure 19-9 shows an example flowchart of sector erasure and figure 19-10 shows an example flowchart of reprogramming.


Start
1)
Set the erase address in FLAADRS.
2)
Set the flash memory acceptor.
3)
Set the SERS flag.
4)
Execute one instruction.
5)
The HOLD state is automatically entered.
6)
Perform the sector erasure.
7)
The HOLD state is automatically canceled.
8)
Verify the data erasure.
Erase complete

1) Set FLAADRS (FA10 to 15) with the upper 6 bits of the address to be erased in 1KB units. FA6 to 9 are invalid.

2) Write n5H and nAH (n = 0 to F) consecutively to FLAACP so that the flash memory acceptor is set to "1". After erasing or reprogramming the flash memory, the flash memory acceptor is reset to "0".

3) The sector erase (SERS) flag of the FLACON register is set to "1". After flash memory is erased, the SERS flag is reset to "0".

4) After the SERS flag is set, the CPU executes one instruction. (One instruction is to be executed immediately after the instruction that sets the SERS flag.)

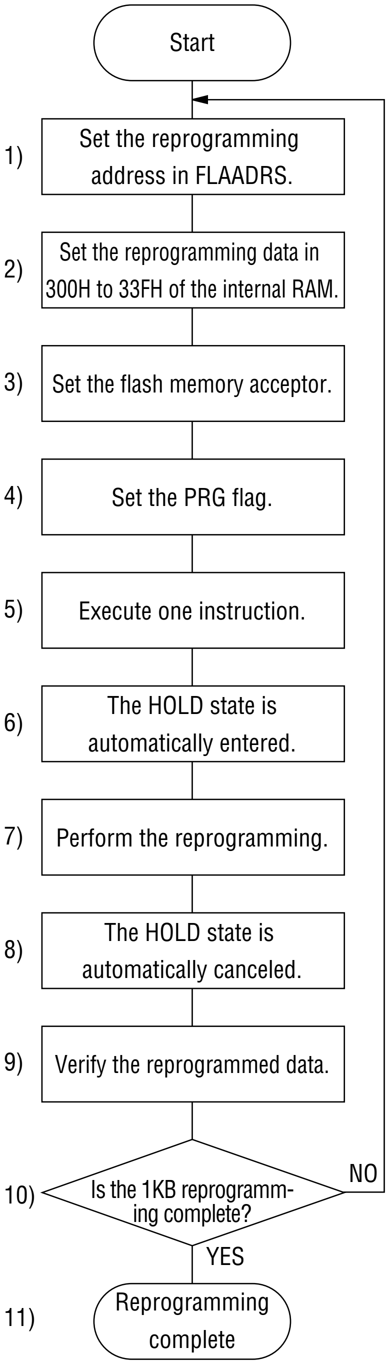
5) After executing one instruction, the CPU automatically enters the HOLD state.

6) When the CPU enters the HOLD state, flash memory sector erasing is performed.

7) When the flash memory sector erasure is complete, the HOLD state is automatically canceled and the CPU executes the next instruction.

8) Verify that the data at the flash memory address specified by FLAADRS is 0FFH. ROM table reference instructions can be used to verify the data.

Figure 19-9 Sector Erase Flowchart Example


Start
1)
Set the reprogramming address in FLAADRS.
2)
Set the reprogramming data in 300H to 33FH of the internal RAM.
3)
Set the flash memory acceptor.
4)
Set the PRG flag.
5)
Execute one instruction.
6)
The HOLD state is automatically entered.
7)
Perform the reprogramming.
8)
The HOLD state is automatically canceled.
9)
Verify the reprogrammed data.
10)
Is the 1KB reprogramm-ing complete?
NO
YES
11)
Reprogramming complete

1) Set FLAADRS (FA6 to 15) with the upper 10 bits of the address to be reprogrammed in 64-byte units.

2) Use serial I/O or an external memory access to set the reprogramming data in 300H to 33FH of the internal RAM.

3) Write n5H and nAH (n = 0 to F) consecutively to FLAACP so that the flash memory acceptor is set to "1". After erasing or reprogramming the flash memory, the flash memory acceptor is reset to "0".

4) The write (PRG) flag of the FLACON register is set to "1". Performing the reprogramming operation resets the PRG flag to "0".

5) After the PRG flag is set, the CPU executes one instruction. (One instruction is to be executed immediately after the instruction that sets the PRG flag.)

6) After executing one instuction, the CPU automatically enters the HOLD state.

7) When the CPU enters the HOLD state, the data in the RAM address area of 300H to 33FH is transferred to the flash memory and is reprogrammed.

8) When the flash memory reprogramming is complete, the HOLD state is automatically canceled and the CPU executes the next instruction.

9) Verify that the flash memory has been reprogrammed with the data. ROM table reference instructions can be used to verify the data.

10) Because reprogramming is performed in 64-byte units, 16 reprogramming cycles are required after the sector erasure.

Figure 19-10 Reprogramming Flowchart Example

(5) Example Data Verification Programs to Check Erased or Reprogrammed Data

• After the flash memory address range of 5000H to 53FFH (1KB) is erased, the following example program verifies that 1KB of data has been erased.

```
        MOV     FLAADRS, #5000H     Set the erase start address (5000H).
        MOVB    FLAACP, #05H
        MOVB    FLAACP, #0AH        Set the flash memory acceptor.
        MOVB    FLACON, #02H        Set the SERS flag.
        NOP                         (Erasing begins after executing one instruction.)
        L       A, #0FFFFH          Set "0FFFFH" in the accumulator.
        MOV     ER0, #5000H         Set the flash memory address to ER0.
LOOP:
        CMPC    A, [ER0]            Compare the accumulator and flash memory data.
        JC      NE, ERR             If different, jump to the error routine.
        ADD     ER0, #0002H         Increment the flash memory address by 2.
        JBR     R1.2, LOOP          If 1KB of data have not been checked, jump to LOOP.
                  ·
                  ·
ERR:            · · · · ·           Perform error processing.
```

• After reprogramming the flash memory address range of 5500H to 553FH (64 bytes), the following example program verifies that 64 bytes of data have been written.

It is assumed that the reprogramming data has been stored in advance in internal RAM addresses 300H to 33FH.

```
        MOV     FLAADRS, #5500H     Set the reprogramming start address (5500H).
        MOVB    FLAACP, #05H
        MOVB    FLAACP, #0AH        Set the flash memory acceptor.
        MOVB    FLACON, #11H        Set the PRG flag.
        NOP                         (Reprogramming begins after executing one instruction.)
        MOV     DP, #300H           Set DP with the internal RAM address.
        MOV     ER0, #5500H         Set ER0 with the flash memory address.
        SDD                         Set the data descriptor.
LOOP:
        LC      A, [ER0]            Load the accumulator with the flash memory data.
        CMP     A, [DP+]            After comparing the accumulator and internal RAM data,
                                    increment the internal RAM address by 2.
        JC      NE, ERR             If different, jump to the error routine.
        ADDB    R0, #02H            Increment the flash memory address by 2.
        JBR     R0.6, LOOP          If 64 bytes of data have not been checked, jump to LOOP.
                  ·
                  ·
ERR:            · · · · ·           Perform error processing.
```

(6) Notes on Usage Regarding User Mode

Be careful of the following items when developing software to be used in the user mode.

- If an interrupt is generated during reprogramming or erasing the flash memory, the interrupt processing is postponed until after the reprogramming or erasing operation is complete. (If timer or other interrupts are to be used, be sure to account for the flash memory reprogramming and erasing times.)

- While reprogramming or erasing the flash memory, when the $\overline{\text{RES}}$ pin input asserts a reset, the reset processing is postponed until after the reprogramming or erasing operation is complete.

- Since erasing is performed in 1KB units, to reprogram even 1 byte, it is necessary to perform a 64-byte write operation 16 times.

- Do not erase the area of flash memory that the reprogramming program is using.
  (After being erased, the CPU will run out of control.)

- Corresponding to one erasure, perform only one write operation to the same address, even if writing the same data.

- When flash memory is erased, the data is set to “0FFH”. Thereafter, writing “0FFH” data is considered as one reprogramming operation.

- The PRG and SERS flags of the FLACON register must be set only one at a time. If both the PRG and SERS flags are set to “1”, neither reprogramming nor erasing operations will be performed.

- Reprogramming and erasing operations initiated by ICE cannot be evaluated.

19.6 Programming Notes

(1) Reprogramming Immediately after Power is Supplied to Flash Memory

Flash memory reprogramming is automatically inhibited for approximately 20 msec after power is turned on (V_{DD} = 5.0 V ±10%). Therefore, when reprogramming is to be performed soon after power is turned on, be sure to wait for the above power-on reset time.

(2) Note when STOP Mode is Canceled

The flash memory requires at least 1 msec of standby time when the STOP mode is canceled. Therefore, set the standby control (SBYCON) register to ensure the wait time when the STOP mode is canceled.

Chapter 20

Electrical Characteristics

20. Electrical Characteristics

20.1 Absolute Maximum Ratings

Parameter	Symbol	Condition		Rating	Unit
Digital power supply voltage	V_{DD}	GND = AGND = 0 V Ta = 25°C		–0.3 to +7.0	V
Input voltage	V_I			–0.3 to V_{DD} + 0.3	V
Output voltage	V_O			–0.3 to V_{DD} + 0.3	V
Analog reference voltage	V_{REF}			–0.3 to V_{DD} + 0.3	V
Analog input voltage	V_{AI}			–0.3 to V_{REF}	V
Power dissipation	P_D	Ta = 70°C	Per package	650	mW
			Per output	8	mW
Storage temperature	T_{STG}	—		–50 to +150	°C

20.2 Recommended Operating Conditions

Parameter	Symbol	Condition		Range	Unit
Digital power supply voltage	V_{DD}	MSM66587A	$f_{OSC} \leq 20$ MHz	4.5 to 5.5	V
		MSM66P587			
		MSM66Q587			
		MSM66587A	$f_{OSC} \leq 10$ MHz	2.7 to 5.5	
		MSM66P587			
Analog reference voltage	V_{REF}	—		V_{DD} – 0.3 to V_{DD}	V
Analog input voltage	V_{AI}	—		AGND to V_{REF}	V
Memory hold voltage	V_{DDH}	f_{OSC} = 0 Hz		2.0 to 5.5	V
Operating frequency	f_{OSC}	V_{DD} = 5 V ±10%		2 to 20	MHz
		MSM66587A	V_{DD} = 2.7 to 5.5 V	2 to 10	MHz
		MSM66P587			
Temperature range	Ta	MSM66587A	—	–30 to +70	°C
		MSM66P587			
		MSM66Q587	V_{DD} = 4.5 to 5.5 V		
Fan-out	N	MOS loads		20	—
		TTL loads	P0, P5_4, P5_5, P7_0, P7_1	2	—
			P1, P2, P4, P6, P7_2-P7_7, P8-P10, P12	1	—

20.3 Allowable Output Current

MSM66587A/P587 (V_{DD} = 2.7 to 5.5 V, Ta = –30 to +70°C)
MSM66Q587 (V_{DD} = 4.5 to 5.5 V, Ta = –30 to +70°C)

Parameter	Pin	Symbol	Min.	Typ.	Max.	Unit
"H" output pin (1 pin)	All output pins	I_{OH}	—	—	–2	mA
"H" output pin (total)	Total of all output pins	ΣI_{OH}	—	—	–40	
"L" output pin (1 pin)	All output pins	I_{OL}	—	—	5	
"L" output pin (total)	Total of P0, P1, P5 and P7	ΣI_{OL}	—	—	50	
	Total of P2, P9 and P10					
	Total of P4 and P8					
	Total of P6 and P12					
	Total of all output pins				100	

Note: Power and ground connections must be made to all external V_{DD} and GND pins.

20.4 DC Characteristics

20.4.1 DC Characteristics (V_{DD} = 5 V ±10%)

MSM66587A/P587/Q587 (V_{DD} = 5 V ±10%, Ta = –30 to +70°C)

Parameter	Symbol	Condition		Min.	Typ.	Max.	Unit
Input high voltage 1	V_{IH}	—		$0.44V_{DD}$	—	V_{DD} + 0.3	V
Input high voltage 2, 4, 5, 6, 7				$0.80V_{DD}$	—	V_{DD} + 0.3	
Input low voltage 1	V_{IL}	—		–0.3	—	$0.16V_{DD}$	
Input low voltage 2, 4, 5, 6, 7				–0.3	—	$0.2V_{DD}$	
Output high voltage 1, 4	V_{OH}	I_O = –400 µA		V_{DD} – 0.4	—	—	
		I_O = –2.0 mA		V_{DD} – 0.6	—	—	
Output high voltage 2		I_O = –200 µA		V_{DD} – 0.4	—	—	
		I_O = –2.0 mA		V_{DD} – 0.6	—	—	
Output low voltage 1, 4	V_{OL}	I_O = 3.2 mA		—	—	0.4	
		I_O = 5.0 mA		—	—	0.8	
Output low voltage 2		I_O = 1.6 mA		—	—	0.4	
		I_O = 5.0 mA		—	—	0.8	
Input leakage current 3, 6	I_{IH}/I_{IL}	$V_I = V_{DD}$/0 V		—	—	1/–1	µA
Input current 5				—	—	1/–250	
Input current 7				—	—	15/–15	
Output leakage current 1, 2, 4	I_{LO}	$V_O = V_{DD}$/0 V		—	—	±10	µA
Pull-up resistor 2	R_{pull}	V_I = 0 V		25	50	100	kΩ
Input capacitance	C_I	f = 1 MHz, Ta = 25°C		—	5	—	pF
Output capacitance	C_O			—	7	—	
Analog reference power supply current	I_{REF}	A/D conversion operating		—	—	4	mA
		A/D conversion stopped		—	—	10	µA
Supply current (STOP mode)	I_{DDS}	V_{DD} = 2 V, Ta = 25°C*		—	0.2	10	µA
		*		—	1	100	
Supply Current (HALT mode)	I_{DDH}	f_{OSC} = 20 MHz, No Load	MSM66587A	—	8	15	mA
			MSM66P587	—	10	25	
			MSM66Q587	—	17	30	
Supply Current	I_{DD}	f_{OSC} = 20 MHz, No Load	MSM66587A	—	25	40	mA
			MSM66P587	—	45	70	
			MSM66Q587	—	45	70	

1. Applies to P0.
2. Applies to P1, P2, P4, P6, P7_2-P7_7, P8-P10, P12.
3. Applies to Ain.
4. Applies to P5_4, P5_5, P7_0, P7_1.
5. Applies to $\overline{\text{RES}}$.
6. Applies to $\overline{\text{EA}}$, NMI.
7. Applies to OSC0.

* For input ports, V_{DD} or 0 V. For other cases, unloaded.

20.4.2 DC Characteristics (V_{DD} = 2.7 to 5.5 V)

MSM66587A/P587 (V_{DD} = 2.7 to 5.5 V, Ta = –30 to +70°C)

Parameter		Symbol	Condition		Min.	Typ.	Max.	Unit
Input high voltage	1	V_{IH}	—		0.44V_{DD}	—	V_{DD} + 0.3	V
Input high voltage	2, 4, 5, 6, 7				0.80V_{DD}	—	V_{DD} + 0.3	
Input low voltage	1	V_{IL}	—		–0.3	—	0.16V_{DD}	
Input low voltage	2, 4, 5, 6, 7				–0.3	—	0.2V_{DD}	
Output high voltage	1, 4	V_{OH}	I_O = –400 µA		V_{DD} – 0.4	—	—	
			I_O = –2.0 mA		V_{DD} – 0.6	—	—	
Output high voltage	2		I_O = –200 µA		V_{DD} – 0.4	—	—	
			MSM66587A	I_O = –1.0 mA	V_{DD} – 0.8	—	—	
			MSM66P587	I_O = –2.0 mA				
Output low voltage	1, 4	V_{OL}	I_O = 3.2 mA		—	—	0.5	
			I_O = 5.0 mA		—	—	0.9	
Output low voltage	2		I_O = 1.6 mA		—	—	0.5	
			MSM66587A	I_O = 2.5 mA	—	—	0.9	
			MSM66P587	I_O = 5.0 mA	—	—	1.2	
Input leakage current	3, 6	I_{IH}/I_{IL}	V_I = V_{DD}/0 V		—	—	1/–1	µA
Input current	5				—	—	1/–250	
Input current	7				—	—	15/–15	
Output leakage current	1, 2, 4	I_{LO}	V_O = V_{DD}/0 V		—	—	±10	µA
Pull-up resistor		R_{pull}	V_I = 0 V, V_{DD} = 5 V ±10%		25	50	100	kΩ
			V_I = 0 V, V_{DD} = 3 V ±10%		40	100	200	
Input capacitance		C_I	f = 1 MHz, Ta = 25°C		—	5	—	pF
Output capacitance		C_O			—	7	—	
Analog reference power supply current		I_{REF}	A/D conversion operating	V_{DD} = 5.5 V	—	—	4	mA
				V_{DD} = 3.3 V	—	—	2	
			A/D conversion stopped	V_{DD} = 5.5 V	—	—	10	µA
				V_{DD} = 3.3 V	—	—	5	

1. Applies to P0.
2. Applies to P1, P2, P4, P6, P7_2-P7_7, P8-P10, P12.
3. Applies to Ain.
4. Applies to P5_4, P5_5, P7_0, P7_1.
5. Applies to $\overline{\text{RES}}$.
6. Applies to $\overline{\text{EA}}$, NMI.
7. Applies to OSC0.

MSM66587A/P587 (V_{DD} = 2.7 to 5.5 V, Ta = –30 to +70°C)

Parameter	Symbol	Condition			Min.	Typ.	Max.	Unit
Supply current (STOP mode)	I_{DDS}	V_{DD} = 2 V, Ta = 25°C*			—	0.2	10	µA
		*			—	1	100	
Supply current (HALT mode)	I_{DDH}	f_{OSC} = 10 MHz, No Load	V_{DD} = 5 V ±10%	MSM66587A	—	4	8	mA
				MSM66P587	—	5	15	
			V_{DD} = 3 V ±10%	MSM66587A	—	2	4	
				MSM66P587	—	3	10	
Supply current	I_{DD}	f_{OSC} = 10 MHz, No Load	V_{DD} = 5 V ±10%	MSM66587A	—	15	25	mA
				MSM66P587	—	30	50	
			V_{DD} = 3 V ±10%	MSM66587A	—	7	13	
				MSM66P587	—	13	25	

* For input ports, V_{DD} or 0 V. For other cases, unloaded.

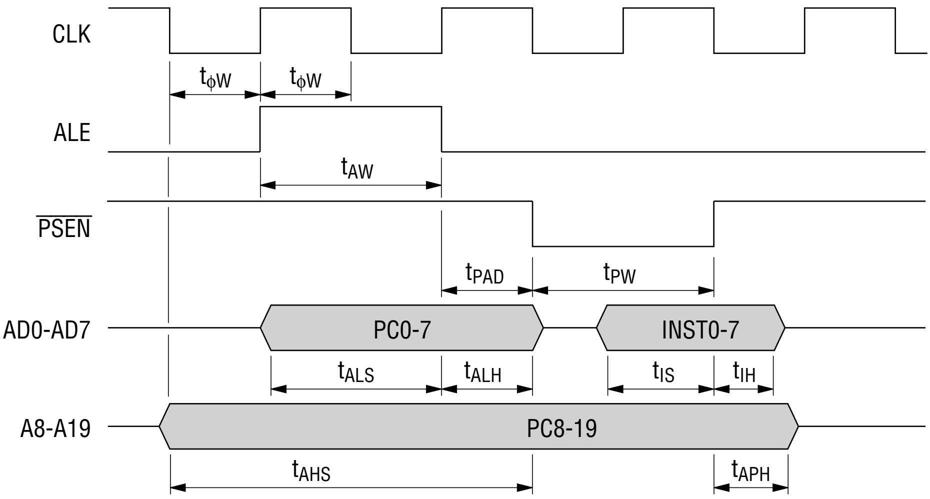
20.5 AC Characteristics

20.5.1 AC Characteristics (V_{DD} = 5 V ±10%)

(1) External Program Memory Control

(V_{DD} = 5 V ±10%, Ta = –30 to +70°C)

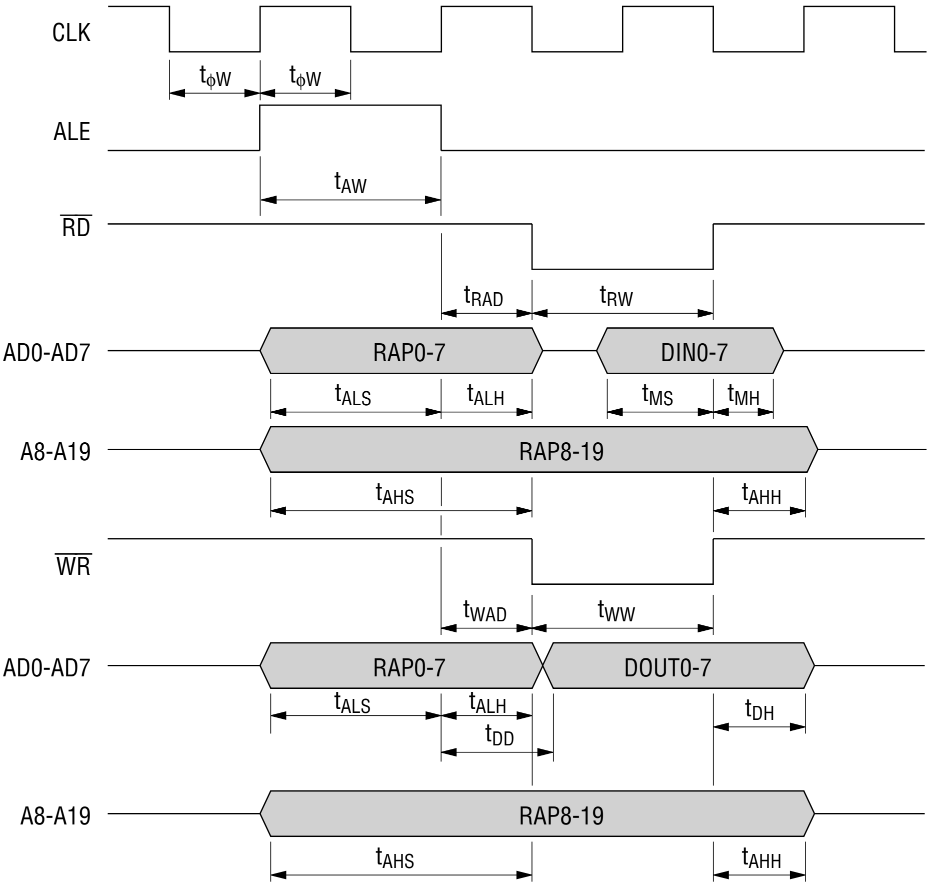
Parameter	Symbol	Condition	Min.	Max.	Unit
Clock (OSC) pulse width	$t_{\phi W}$	—	25	—	ns
ALE pulse width	t_{AW}	C_L = 50 pF	$2\ t_{\phi W} - 2$	—	
$\overline{PSEN}$ pulse width	t_{PW}		$2\ t_{\phi W} - 5$	—	
$\overline{PSEN}$ pulse delay time	t_{PAD}		$t_{\phi W} - 3$	$t_{\phi W} + 3$	
Low address setup time	t_{ALS}		$2\ t_{\phi W} - 3$	$2\ t_{\phi W} + 3$	
Low address hold time	t_{ALH}		$t_{\phi W} - 3$	$t_{\phi W} + 3$	
High address setup time	t_{AHS}		$4\ t_{\phi W} - 3$	$4\ t_{\phi W} + 3$	
High address hold time	t_{APH}		0	$t_{\phi W} + 3$	
Instruction setup time	t_{IS}		15	—	
Instruction hold time	t_{IH}		0	$t_{\phi W} - 3$	



(2) External Data Memory Control

(V_{DD} = 5 V ±10%, Ta = –30 to +70°C)

Parameter	Symbol	Condition	Min.	Max.	Unit
Clock (OSC) pulse width	$t_{\phi W}$	—	25	—	ns
ALE pulse width	t_{AW}	C_L = 50 pF	$2\ t_{\phi W} - 2$	—	
$\overline{RD}$ pulse width	t_{RW}		$2\ t_{\phi W} - 5$	—	
$\overline{WR}$ pulse width	t_{WW}		$2\ t_{\phi W} - 5$	—	
$\overline{RD}$ pulse delay time	t_{RAD}		$t_{\phi W} - 3$	$t_{\phi W} + 3$	
$\overline{WR}$ pulse delay time	t_{WAD}		$t_{\phi W} - 3$	$t_{\phi W} + 3$	
Low address setup time	t_{ALS}		$2\ t_{\phi W} - 3$	$2\ t_{\phi W} + 3$	
Low address hold time	t_{ALH}		$t_{\phi W} - 3$	$t_{\phi W} + 3$	
High address setup time	t_{AHS}		$3\ t_{\phi W} - 3$	$3\ t_{\phi W} + 3$	
High address hold time	t_{AHH}		$t_{\phi W} - 3$	$t_{\phi W} + 3$	
Memory data setup time	t_{MS}		15	—	
Memory data hold time	t_{MH}		0	$t_{\phi W} - 3$	
Data delay time	t_{DD}		$t_{ALH} - 0$	$t_{ALH} + 5$	
Data hold time	t_{DH}		$t_{\phi W} - 3$	$t_{\phi W} + 3$	

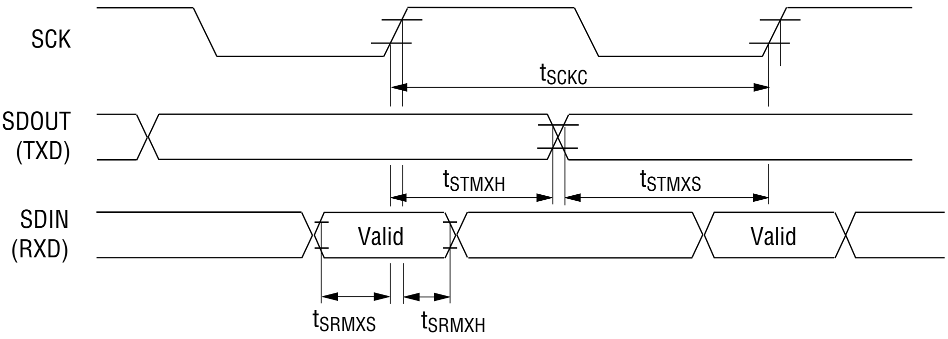


(3) Serial Port Contorl

- Master mode

(V_{DD} = 5 V ±10%, Ta = –30 to +70°C)

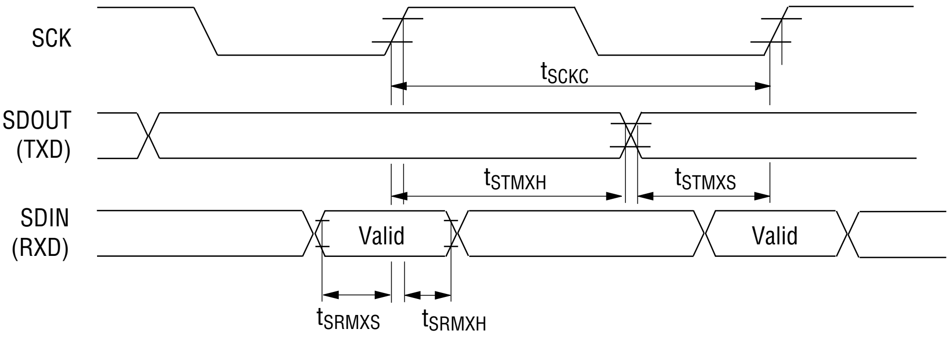
Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Clock (OSC) pulse width	$t_{\phi W}$	—	25	—	—	ns
Serial clock cycle time	t_{SCKC}	C_L = 50 pF	8 $t_{\phi W}$	—	—	
Output data setup time	t_{STMXS}		4 $t_{\phi W}$ – 5	—	—	
Output data hold time	t_{STMXH}		3 $t_{\phi W}$ – 10	—	—	
Input data setup time	t_{SRMXS}		20	—	—	
Input data hold time	t_{SRMXH}		0	—	—	



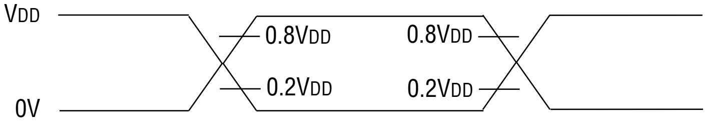
- Slave mode

(V_{DD} = 5 V ±10%, Ta = –30 to +70°C)

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Clock (OSC) pulse width	$t_{\phi W}$	—	25	—	—	ns
Serial clock cycle time	t_{SCKC}	C_L = 50 pF	8 $t_{\phi W}$	—	—	
Output data setup time	t_{STMXS}		2 $t_{\phi W}$ – 15	—	—	
Output data hold time	t_{STMXH}		4 $t_{\phi W}$ – 10	—	—	
Input data setup time	t_{SRMXS}		20	—	—	
Input data hold time	t_{SRMXH}		5	—	—	



AC timing mesurement point

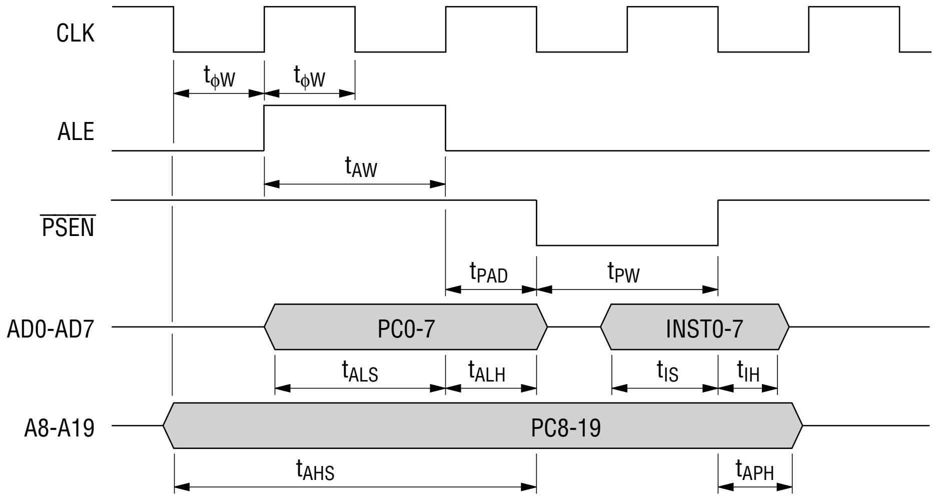


20.5.2 AC Characteristics (V_{DD} = 2.7 to 5.5 V)

(1) External Program Memory Control

MSM66587A/P587 (V_{DD} = 2.7 to 5.5 V, Ta = –30 to +70°C)

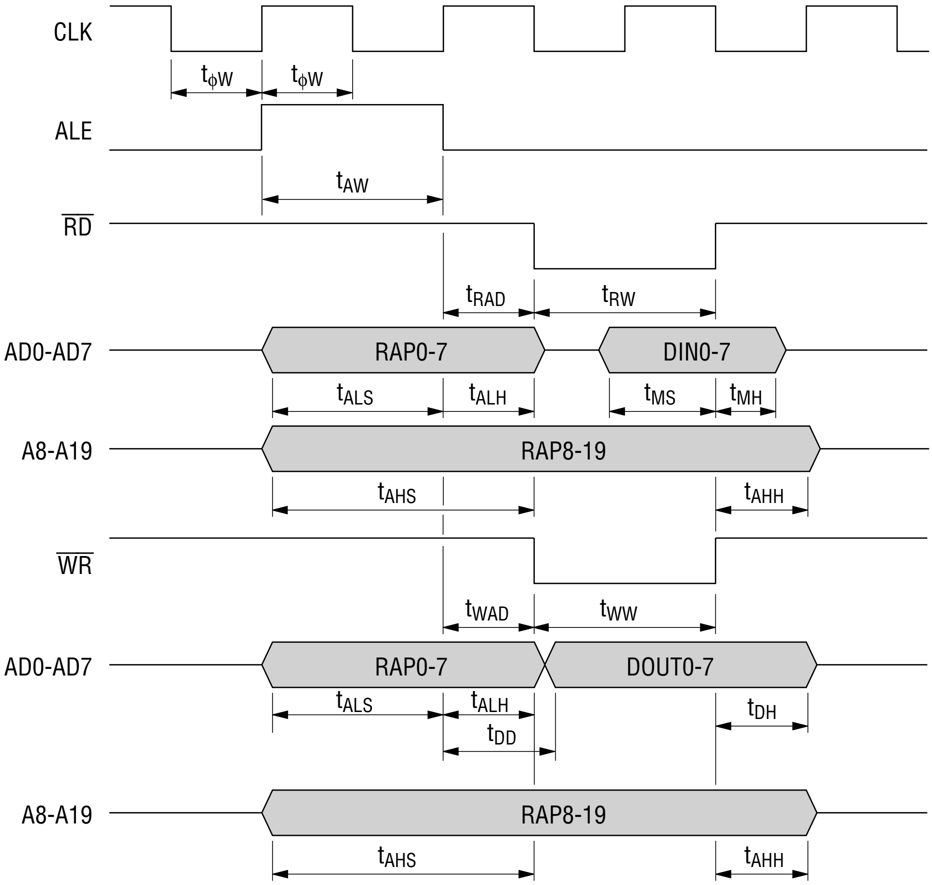
Parameter	Symbol	Condition	Min.	Max.	Unit
Clock (OSC) pulse width	$t_{\phi W}$	—	50	—	ns
ALE pulse width	t_{AW}	C_L = 50 pF	2 $t_{\phi W}$ – 4	—	
$\overline{\text{PSEN}}$ pulse width	t_{PW}		2 $t_{\phi W}$ – 10	—	
$\overline{\text{PSEN}}$ pulse delay time	t_{PAD}		$t_{\phi W}$ – 6	$t_{\phi W}$ + 6	
Low address setup time	t_{ALS}		2 $t_{\phi W}$ – 6	2 $t_{\phi W}$ + 6	
Low address hold time	t_{ALH}		$t_{\phi W}$ – 6	$t_{\phi W}$ + 6	
High address setup time	t_{AHS}		4 $t_{\phi W}$ – 6	4 $t_{\phi W}$ + 6	
High address hold time	t_{APH}		0	$t_{\phi W}$ + 6	
Instruction setup time	t_{IS}		30	—	
Instruction hold time	t_{IH}		0	$t_{\phi W}$ – 6	



(2) External Data Memory Control

MSM66587A/P587 (V_{DD} = 2.7 to 5.5 V, Ta = –30 to +70°C)

Parameter	Symbol	Condition	Min.	Max.	Unit
Clock (OSC) pulse width	$t_{\phi W}$	—	50	—	ns
ALE pulse width	t_{AW}	C_L = 50 pF	2 $t_{\phi W}$ – 4	—	
$\overline{RD}$ pulse width	t_{RW}		2 $t_{\phi W}$ – 10	—	
$\overline{WR}$ pulse width	t_{WW}		2 $t_{\phi W}$ – 10	—	
$\overline{RD}$ pulse delay time	t_{RAD}		$t_{\phi W}$ – 6	$t_{\phi W}$ + 6	
$\overline{WR}$ pulse delay time	t_{WAD}		$t_{\phi W}$ – 6	$t_{\phi W}$ + 6	
Low address setup time	t_{ALS}		2 $t_{\phi W}$ – 6	2 $t_{\phi W}$ + 6	
Low address hold time	t_{ALH}		$t_{\phi W}$ – 6	$t_{\phi W}$ + 6	
High address setup time	t_{AHS}		3 $t_{\phi W}$ – 6	3 $t_{\phi W}$ + 6	
High address hold time	t_{AHH}		$t_{\phi W}$ – 6	$t_{\phi W}$ + 6	
Memory data setup time	t_{MS}		30	—	
Memory data hold time	t_{MH}		0	$t_{\phi W}$ – 6	
Data delay time	t_{DD}		t_{ALH} – 0	t_{ALH} + 10	
Data hold time	t_{DH}		$t_{\phi W}$ – 6	$t_{\phi W}$ + 6	

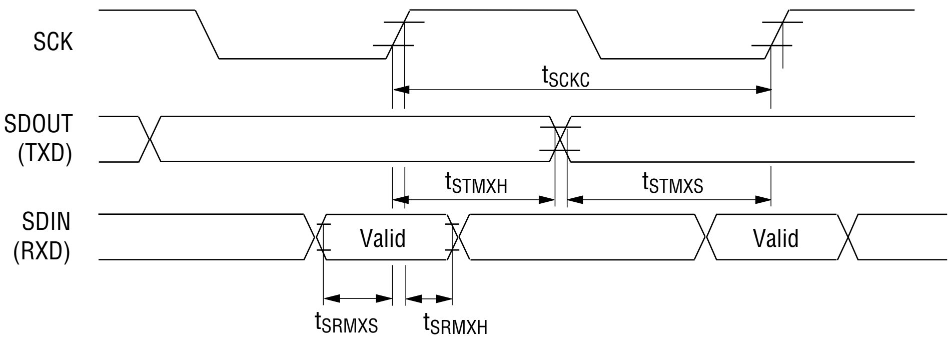


(3) Serial Port Control

- Master mode

MSM66587A/P587 (V_{DD} = 2.7 to 5.5 V, Ta = –30 to +70°C)

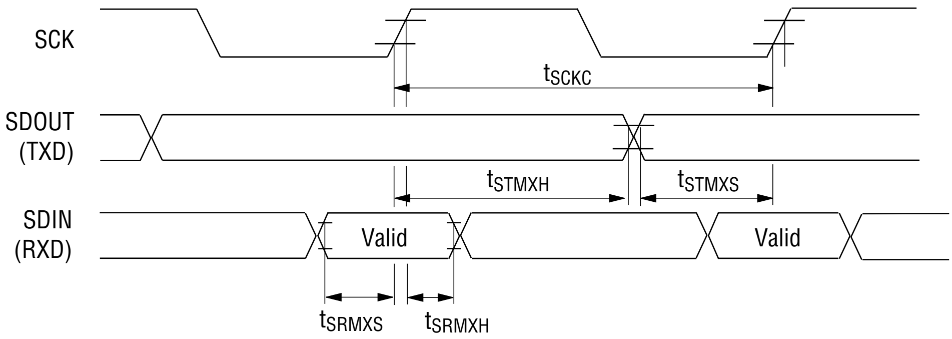
Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Clock (OSC) pulse width	$t_{\phi W}$	—	50	—	—	ns
Serial clock cycle time	t_{SCKC}	C_L = 50 pF	8 $t_{\phi W}$	—	—	
Output data setup time	t_{STMXS}		4 $t_{\phi W}$ – 10	—	—	
Output data hold time	t_{STMXH}		3 $t_{\phi W}$ – 20	—	—	
Input data setup time	t_{SRMXS}		30	—	—	
Input data hold time	t_{SRMXH}		0	—	—	



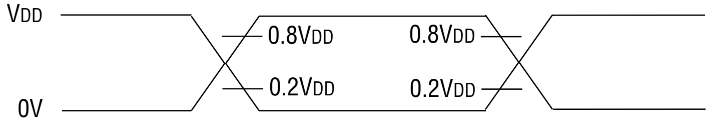
- Slave mode

MSM66587A/P587 (V_{DD} = 2.7 to 5.5 V, Ta = –30 to +70°C)

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Clock (OSC) pulse width	$t_{\phi W}$	—	50	—	—	ns
Serial clock cycle time	t_{SCKC}	C_L = 50 pF	8 $t_{\phi W}$	—	—	
Output data setup time	t_{STMXS}		2 $t_{\phi W}$ – 30	—	—	
Output data hold time	t_{STMXH}		4 $t_{\phi W}$ – 20	—	—	
Input data setup time	t_{SRMXS}		30	—	—	
Input data hold time	t_{SRMXH}		10	—	—	



AC timing mesurement point



20.6 A/D Converter Characteristics

20.6.1 A/D Converter Characteristics (V_{DD} = 5 V ±10%)

(Ta = –30 to +70°C, V_{DD} = V_{REF} = 5 V ±10%, AGND = GND = 0 V, f_{OSC} = 20 MHz)

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Resolution	n	Refer to the recommended circuit (Fig. 20-1) Analog input source impedance $R_I \leq 5$ kΩ t_{CONV} = 19.219.2 µsec	—	8	—	Bit
Linearity error	E_L		—	—	±2	LSB
Differential linearity error	E_D		—	—	±1	
Zero scale error	E_{ZS}		—	—	+2	
Full scale error	E_{FS}		—	—	–2	
Conversion time	t_{CONV}	by ADTM set data	6.4	—	19.2	µs/ch

20.6.2 A/D Converter Characteristics (V_{DD} = 3 V ±10%)

MSM66587A/P587 (Ta = –30 to +70°C, V_{DD} = V_{REF} = 3 V ±10%, AGND = GND = 0 V, f_{OSC} = 10 MHz)

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Resolution	n	Refer to the recommended circuit (Fig. 20-1) Analog input source impedance $R_I \leq 5$ kΩ t_{CONV} = 38.4 µsec	—	8	—	Bit
Linearity error	E_L		—	—	±1	LSB
Differential linearity error	E_D		—	—	±0.5	
Zero scale error	E_{ZS}		—	—	+1	
Full scale error	E_{FS}		—	—	–1	
Conversion time	t_{CONV}	ADTM = 00b (384CLK selection)	—	38.4	—	µs/ch

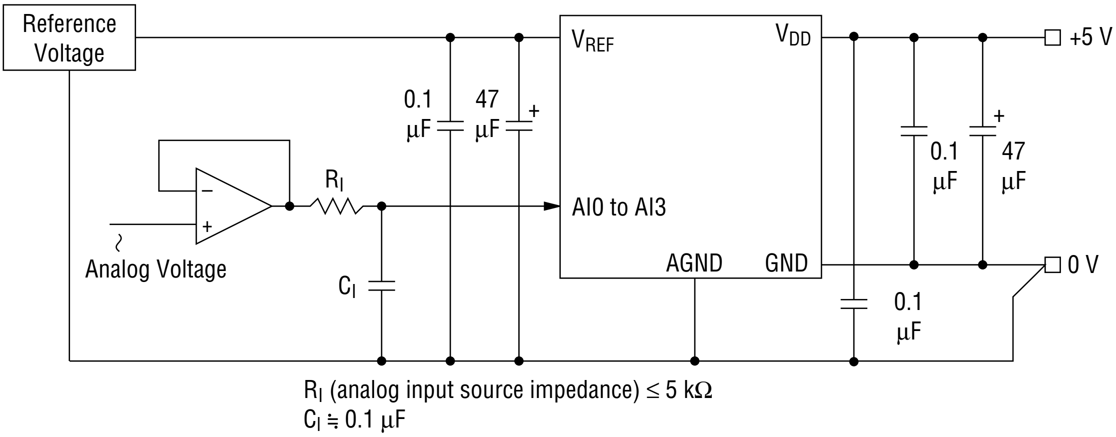


Figure 20-1 Recommended Circuit

Definition of terms

1. Resolution

   Resolution is t2he minimum input analog value that can be resolved. With 8 bits, $2^8 = 256$ so resolution can be to (V_{REF} – AGND) ÷ 256.

2. Linearity error

   Linearity error is the difference between actual conversion characteristics and ideal conversion characteristics of an 8-bit A/D converter (so this does not include quantization error).

   Ideal conversion characteristics would be to divide the voltage between V_{REF} and AGND into 256 equal steps.

3. Differential linearity error

   Differential linearity error indicates slope of conversion characteristics. The change in analog input voltage value that would change the digital output by one bit is ideally 1 LSB = (V_{REF} – AGND) ÷ 256, so differential linearity error is the difference between this ideal bit size and the actual bit size anywhere in the conversion range.

4. Zero scale error

   Zero scale error is the difference between actual conversion characteristics and ideal conversion characteristics at the point where digital output switches from 00H to 01H.

5. Full scale error

   Full scale error is the difference between actual conversion characteristics and ideal conversion characteristics at the point where digital output switches from FEH to FFH.

Chapter 21

Special Function Registers (SFRs)

21. Special Function Registers (SFRs)

The 256-byte area 0000H-00FFH in data memory space is the SFR area. The 256-byte area 0100H-01FFH is the expanded SFR area. There are restrictions that limit expanded SFR and SFR addressing.

The SFR and expanded SFR areas are register groups allocated special functions like the following.

- Peripheral hardware mode registers
- Arithmetic register (ACC)
- Control registers (PSW, LRBL, LRBH, SSP)

Table 21-1 lists the SFRs. Terms in the table have the following meanings.

- Address[H] Addresses are expressed in hexadecimal.

- Function This is what the SFR function is called.

- Byte, Word:
  - *symbol* This symbol indicates an I/O function register.
    The unit indicates whether it can be accessed by byte or by word.
  - — This indicates that the function cannot be accessed by this unit.

- Bit Symbol:
  - *symbol* This symbol indicates an I/O function register.
  - blank The I/O register is allocated to this bit as well.
    However, this bit has no need to be accessed on a bit basis and can not be accessed on a bit basis, so, the bit symbol is not needed.
  - — This symbol indicates a bit that represents that an I/O funciton register does not exist.
    Programming should be done assuming the bit is always undefined.
    Operation of this bit may differ with ICE evaluation and the actual chip (Mask version, OTP version, etc.).
  - 0 Always write "0" to this bit.
    Write 0 to this bit even when writing a byte or word that includes it.
    This bit will always read as "0".
  - 1 Always write "1" to this bit.
    Write 1 to this bit even when writing a byte or word that includes it.
    This bit will always read as "1".
  - (1) All writes to this bit will be ignored.
    This bit will always read as "1".

Undefined This indicates a bit that does not support the I/O function. Programming should be done on the assumption that the value of this bit is always undefined. Operation of this bit when an in-circuit emulator is used may differ from operation when an actual chip (such as MASK or OTP versions) is used.

- R/W This indicates whether the specified SFR can be read (R) or written (W).

R/W	Can be read and written.
R	Can only be read.
W	Can only be written.

- Reset state Reset state indicates SFR contents after reset ($\overline{\text{RES}}$ signal input, BRK instruction execution, op code trap).

- Reference page This indicates the page that discusses the configuration of the bit symbols.

Note: Stars indicate SFRs that exist only in flash-version devices. For details, please refer to Chapter 19, "Flash Memory".

Notes:

Do not perform the following operations on the SFRs listed below.

1. Write operations on read-only SFRs.
2. Read operations on write-only SFRs.
3. 16-bit operations on 8-bit-only SFRs.
4. 8-bit operations on 16-bit only SFRs.
5. Operations on addresses which have not been allocated as register and so on.
6. Operations in the area for emulator use.

Table 21-1 SFR List

Address [H]	Function	Byte	Word	Bit Symbol								R/W	Reset State	Refere-nce Page
				7	6	5	4	3	2	1	0			
0000	System Stack Pointer	—	SSP									R/W	FFFF	P3-20
0001		—												
0002	Local Register Base	LRBL	LRB										undefined	P3-19
0003		LRBH												
0004	Program Status Word	PSWL	PSW	MAB	F1	BCB1	BCB0	F0	SCB2	SCB1	SCB0		00	P3-15
0005		PSWH		CY	ZF	HC	DD	S	MIP	OV	MIE		00	
0006	Accumulator	ACCL	ACC										00	P3-14
0007		ACCH											00	
0008	Table Segment Register	TSR	DSTSR	0	0	0	0						00	P3-23
0009	Data Segment Register	DSR		0	0	0	0						00	P3-24
000B	ROM Window Register	ROMWIN	—										00	P5-2
000C	ROM Ready Control Register	ROMRDY	—	(1)	(1)	(1)	(1)	(1)	(1)	ORDY1	ORDY0		FF	P5-4
000D	RAM Ready Control Register	RAMRDY	—	(1)	ARDY12	ARDY11	ARDY10	(1)	ARDY02	ARDY01	ARDY00		FF	P5-3
000E	Stop Code Acceptor	STPACP	—	—	—	—	—					W	"0"	P4-5
000F	Standby Control Register	SBYCON	—	(1)	(1)	OST1	OST0	(1)	FLT	HLT	STP	R/W	C8	P4-3
0010	Port 0 Data Register	P0	—	P0_7	P0_6	P0_5	P0_4	P0_3	P0_2	P0_1	P0_0		00	P6-10
0011	Port 1 Data Register	P1		P1_7	P1_6	P1_5	P1_4	P1_3	P1_2	P1_1	P1_0		00	P6-11
0012	Port 2 Data Register	P2	—	P2_7	P2_6	P2_5	P2_4	P2_3	P2_2	P2_1	P2_0		00	P6-12
0014	Port 4 Data Register	P4	—	P4_7	P4_6	P4_5	P4_4	P4_3	P4_2	P4_1	P4_0		00	P6-14
0015	Port 5 Data Register	P5		(1)	(1)	P5_5	P5_4	undefined	undefined	undefined	undefined		undefined	P6-16
0016	Port 6 Data Register	P6	—	undefined	undefined	P6_5	P6_4	P6_3	P6_2	P6_1	P6_0		undefined	P6-17
0017	Port 7 Data Register	P7		P7_7	P7_6	P7_5	P7_4	P7_3	P7_2	P7_1	P7_0		00	P6-19
0018	Port 0 Mode Register	P0IO	—	P0IO7	P0IO6	P0IO5	P0IO4	P0IO3	P0IO2	P0IO1	P0IO0		00	P6-10
0019	Port 1 Mode Register	P1IO		P1IO7	P1IO6	P1IO5	P1IO4	P1IO3	P1IO2	P1IO1	P1IO0		00	P6-11
001A	Port 2 Mode Register	P2IO	—	P2IO7	P2IO6	P2IO5	P2IO4	P2IO3	P2IO2	P2IO1	P2IO0		00	P6-12
001C	Port 4 Mode Register	P4IO	—	P4IO7	P4IO6	P4IO5	P4IO4	P4IO3	P4IO2	P4IO1	P4IO0		00	P6-14
001D	Port 5 Mode Register	P5IO		(1)	(1)	P5IO5	P5IO4	undefined	undefined	undefined	undefined		undefined	P6-16
001E	Port 6 Mode Register	P6IO	—	undefined	undefined	P6IO5	P6IO4	P6IO3	P6IO2	P6IO1	P6IO0		undefined	P6-17
001F	Port 7 Mode Register	P7IO		P7IO7	P7IO6	P7IO5	P7IO4	P7IO3	P7IO2	P7IO1	P7IO0		00	P6-19

Address [H]	Function	Byte	Word	Bit Symbol 7	6	5	4	3	2	1	0	R/W	Reset State	Refere-nce Page
0020	Port 8 Data Register	P8	—	P8_7	P8_6	P8_5	P8_4	P8_3	P8_2	P8_1	P8_0	R/W	00	P6-21
0021	Port 9 Data Register	P9		P9_7	P9_6	P9_5	P9_4	P9_3	P9_2	P9_1	P9_0		00	P6-22
0022	Port 10 Data Register	P10	—	P10_7	P10_6	P10_5	P10_4	P10_3	P10_2	P10_1	P10_0		00	P6-24
0028	Port 8 Mode Register	P8IO	—	P8IO7	P8IO6	P8IO5	P8IO4	P8IO3	P8IO2	P8IO1	P8IO0		00	P6-21
0029	Port 9 Mode Register	P9IO		P9IO7	P9IO6	P9IO5	P9IO4	P9IO3	P9IO2	P9IO1	P9IO0		00	P6-22
002A	Port 10 Mode Register	P10IO	—	P10IO7	P10IO6	P10IO5	P10IO4	P10IO3	P10IO2	P10IO1	P10IO0		00	P6-24
002C	A/D Interrupt Control Register	ADINTCON	—	(1)	(1)	0	0	ADSTIE	0	INTST	0		C0	P14-4
0031	Port 1 Secondary Control Register	P1SF	—	XDM15	XDM14	XDM13	XDM12	XDM11	XDM10	XDM9	XDM8		00	P6-11
0032	Port 2 Secondary Control Register	P2SF	—	(1)	0	RTO9	RTO8	0	0	0	0		80	P6-12
0034	Port 4 Secondary Control Register	P4SF	—	0	0	0	0	0	0	0	ETMCK		00	P6-14
0035	Port 5 Secondary Control Register	P5SF	—	(1)	(1)	ALE	PSEN	(1)	(1)	(1)	(1)		CF	P6-16
0036	Port 6 Secondary Control Register	P6SF	—	0	0	TXC1	RXC1	TXD1	RXD1	INT1	INT0		00	P6-17
0037	Port 7 Secondary Control Register	P7SF	—	0	0	0	PWM0	CLKOUT	WAIT	RD	WR		00	P6-19
0039	Port 9 Secondary Control Register	P9SF	—	0	0	0	0	XDM19	XDM18	XDM17	XDM16		00	P6-22
0040	Interrupt Request Register 0	IRQ0L	IRQ0	0	0	0	0	0	QTM1OV	0	QINT0		00	P17-6
0041		IRQ0H		QSCI1	0	0	QRTO9	QRTO8	0	0	0		00	
0042	Interrupt Request Register 1	IRQ1L	—	QINT1	0	0	QPW01	QAD	0	QGTMOV	QSTMOV		00	
0044	Interrupt Enable Register 0	IE0L	IE0	0	0	0	0	0	ETM1OV	0	EINT0		00	
0045		IE0H		ESCI1	0	0	ERTO9	ERTO8	0	0	0		00	
0046	Interrupt Enable Register 1	IE1L	—	EINT1	0	0	EPW01	EAD	0	EGTMOV	ESTMOV		00	
0048	Interrupt Priority Control Register 00	IP00L	IP00	0	0	0	0	0	P0TM1OV	0	P0INT0		00	
0049		IP00H		P0SCI1	0	0	P0RTO9	P0RTO8	0	0	0		00	
004A	Interrupt Priority Control Register 01	IP01L	IP01	0	0	0	0	0	P1TM1OV	0	P1INT0		00	
004B		IP01H		P1SCI1	0	0	P1RTO9	P1RTO8	0	0	0		00	
004C	Interrupt Priority Control Register 10	IP10L	—	P0INT1	0	0	P0PW01	P0AD	0	P0GTMOV	P0STMOV		00	
004E	Interrupt Priority Control Register 11	IP11L	—	P1INT1	0	0	P1PW01	P1AD	0	P1GTMOV	P1STMOV		00	
0050	PWM Counter 0	—	PWC0									R	FFFF	P11-2
0051		—												

Address [H]	Function	Byte	Word	Bit Symbol 7	6	5	4	3	2	1	0	R/W	Reset State	Reference Page
0060	PWC0 Buffer Register	—	PWC0BF									R/W	FFFF	P11-3
0061		—												
0070	PWR0 Buffer Register	—	PW0BF										0000	
0071		—												
0090	Timer Register 8	—	TMR8										0000	P9-5
0091		—												
0092	Timer Register 9	—	TMR9										0000	
0093		—												
00B4	RTO Control Register 8	RTOCON4	—	(1)	(1)	(1)	(1)	(1)	(1)	T8BF0	T8OUT		FC	
00B5	RTO Control Register 9	RTOCON5		(1)	(1)	(1)	(1)	(1)	(1)	T9BF0	T9OUT		FC	
00C2	Timer Counter 1	—	TM1										0000	P9-2
00C3		—												
00D0	A/D Result Register 0	ADCR0	—									R	undefined	P14-4
00E1	A/D Control Register H	ADCONH	—	(1)	ADTM1	ADTM0	STS	0	0	ADSTM1	ADSTM0	R/W	80	P14-3
00E2	Timer Control Register	TMCON	—	TM1RUN	TM1CK2	TM1CK1	TM1CK0	0	0	0	0		00	P9-3
00E4	TM Setting Register	—	TMSEL	0	0	0	0	0	0	0	0		0300	
00E5		—		0	0	0	0	0	0	1	1			
00E6	TMR Mode Register	TMRMODE	—	0	0	0	0	T9MD1	T9MD0	T8MD1	T8MD0		00	P9-6
00EB	PWM Run Register	PWRUN	—	0	0	0	0	0	0	0	PW0RUN		00	P11-4
00EC	PWM Control Register 0	PWCON0		0	0	0	0	PW0ACT	PW0CK2	PW0CK1	PW0CK0		00	
00F0	PWM Interrupt Register 0	PWINTQ0	PWINTQ	0	0	0	0	0	0	0	QPW0		00	P11-5
00F1	PWM Interrupt Register 1	PWINTQ1		0	0	0	0	0	0	0	QPWR0		00	
00F2	PWM Interrupt Enable Register 0	PWINTE0	PWINTE	0	0	0	0	0	0	0	EPW0		00	P11-6
00F3	PWM Interrupt Enable Register 1	PWINTE1		0	0	0	0	0	0	0	EPWR0		00	
00F6	SCI1 Transmit/Receive Buffer Register	S1BUF	—										undefined	P13-5
00F7	SCI1 Status Register	S1STAT		TR1IRQ	TR1IE	RV1IRQ	RV1IE	0	0	OERR1	0		00	P13-6
00F8	General 8-Bit Timer Control Register	GTMCON	—	0	0	(1)	(1)	GTMRUN	GTMCK2	GTMCK1	GTMCK0		30	P10-2
00FA	General 8-Bit Timer Counter	GTMC	—										00	
00FB	General 8-Bit Timer Register	GTMR											00	

Expanded SFR Area

Address [H]	Function	Byte	Word	Bit Symbol 7	6	5	4	3	2	1	0	R/W	Reset State	Refere-nce Page
0100	Memory Size Acceptor	MEMSACP	—									W	"0"	P3-1
0101	Memory Size Control Register	MEMSCON		(1)	(1)	(1)	(1)	(1)	(1)	LROM	LRAM	R/W	FC	
0107	Peripheral Control Register	PRPHF	—	undefined	undefined	(1)	(1)	(1)	(1)	CKOUT1	CKOUT0		undefined	P15-1
0108	NMI Control Register	NMICON	—	NMIRD	(1)	(1)	(1)	(1)	(1)	NMIM1	NMIM0		FCor7C	P16-3
0109	External Interrupt Control Register	EXICON		EX3M1	EX3M0	EX2M1	EX2M0	EX1M1	EX1M0	EX0M1	EX0M0		00	P16-2
010C	Interrupt Request Register 2	IRQ2L	—	0	0	0	0	QINT3	QINT2	0	0		00	P17-4 P17-5
010D	Interrupt Enable Register 2	IE2L		0	0	0	0	EINT3	EINT2	0	0		00	
010E	Interrupt Priority Control Register 20	IP20L	—	0	0	0	0	P0INT3	P0INT2	0	0		00	
010F	Interrupt Priority Control Register 21	IP21L		0	0	0	0	P1INT3	P1INT2	0	0		00	
0118	SCI1Timer Counter	—	S1TM										0000	P12-2
0119	SCI1Timer Register	—												
011A	SCI1Timer Control Register	S1CON	—	S1CK2	S1CK1	S1CK0	S1RUN	ESTM1	QSTM1	(1)	S1MOD		02	P12-3
011B	SCI1Transmit Control Register	ST1CON	—	(1)	0	0	ST1MST	(1)	1	1	0	R	8E	P13-4
011C	SCI1Transmit/Receive Control Register	SR1CON	—	SR1REN	0	0	SR1MST	(1)	1	1	0	R/W	0E	P13-3
011E	TBC Clock Dividing Counter	TBCKDVC	—	(1)	(1)	(1)	(1)					R	F0	P8-2
011F	TBC Clock Division Register	TBCKDVR		(1)	(1)	(1)	(1)					R/W	F0	
0120	Port 12 Data Register	P12	—	P12_7	P12_6	P12_5	P12_4	P12_3	P12_2	P12_1	P12_0		00	P6-25
0122	Port 12 Mode Register	P12IO	—	P12IO7	P12IO6	P12IO5	P12IO4	P12IO3	P12IO2	P12IO1	P12IO0		00	
0124	Port 12 Secondary Control Register	P12SF	—	0	0	0	0	INT3	INT2	0	0		00	
0130	Port 0 Pull-Up Control Register	P0PUP	—	P0PUP7	P0PUP6	P0PUP5	P0PUP4	P0PUP3	P0PUP2	P0PUP1	P0PUP0		00	P6-10
0131	Port 1 Pull-Up Control Register	P1PUP		P1PUP7	P1PUP6	P1PUP5	P1PUP4	P1PUP3	P1PUP2	P1PUP1	P1PUP0		00	P6-11
0134	Port 4 Pull-Up Control Register	P4PUP	—	P4PUP7	P4PUP6	P4PUP5	P4PUP4	P4PUP3	P4PUP2	P4PUP1	P4PUP0		00	P6-14
0139	Port 9 Pull-Up Control Register	P9PUP		P9PUP7	P9PUP6	P9PUP5	P9PUP4	P9PUP3	P9PUP2	P9PUP1	P9PUP0		00	P6-22
013A	Port 10 Pull-Up Control Register	P10PUP	—	P10PUP7	P10PUP6	P10PUP5	P10PUP4	P10PUP3	P10PUP2	P10PUP1	P10PUP0		00	P6-24
☆01F0	Flash Memory Acceptor	FLAACP	—	—	—	—	—					W	"0"	P19-8
☆01F1	Flash Memory Control Register	FLACON		—	—	—	FCLK1	FCLK0	—	SERS	PRG	R/W	E4	
☆01F2	Flash Memory Address Register	—	FLAADRS			—	—	—	—	—	—		undefined	P19-7
☆01F3		—												

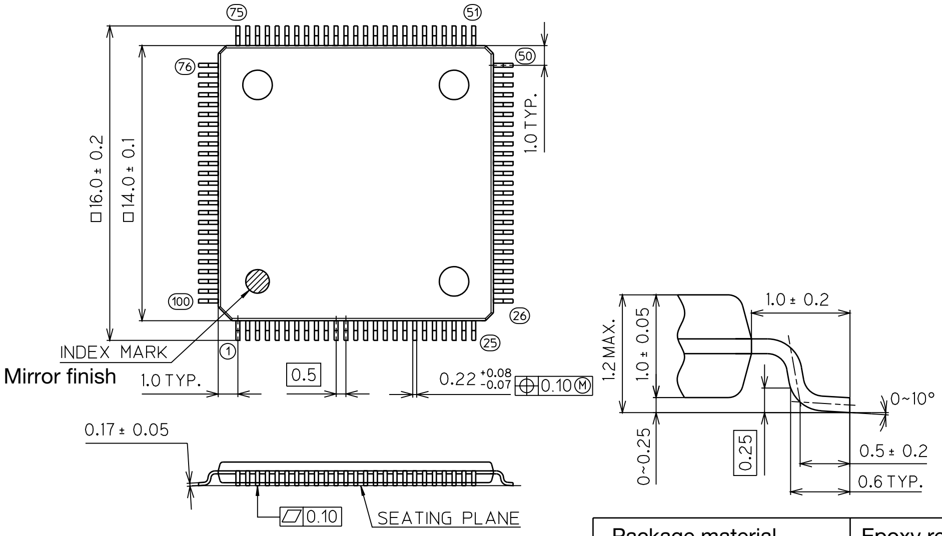
Chapter 22

Package Dimensions

22. Package Dimensions

(Unit : mm)

TQFP100-P-1414-0.50-K



Package material	Epoxy resin
Lead frame material	42 alloy
Pin treatment	Solder plating
Solder plate thickness	5 µm or more
Package weight (g)	0.55 TYP.

Notes for Mounting the Surface Mounting Type Package

The TQFP is a surface mount type package and is very susceptible to heat in reflow mounting and to humidity absorbed in storage. Therefore, before you do reflow mounting, contact Oki's responsible sales person on the product name, package name, pin number, package code and desired mounting conditions (reflow method, temperature and times).

MSM66587 Family

User's Manual

First Edition: September 1997
Second Edition: January 2000

FEUL66587-02