FUJITSU SEMICONDUCTOR
DATA SHEET

DS07-13704-1E

16-bit Proprietary Microcontroller

CMOS

F²MC-16LX MB90590 Series

MB90594/591/F594A/F591/V590A

■ DESCRIPTION

The MB90590-series with two FULL-CAN interfaces and FLASH ROM is especially designed for automotive and industrial applications. Its main feature are two on board CAN Interfaces, which conform to V2.0 Part A and Part B, while supporting a very flexible message buffer scheme and so offering more functions than a normal full CAN approach. With the new 0.5µm CMOS technology, Fujitsu now also offers on-chip FLASH-ROM program memory. An internal voltage booster removes the necessity for a second programming voltage.
An on board voltage regulator provides 3V to the internal MCU core. This creates a major advantage in terms of EMI and power consumption.
The internal PLL clock frequency multiplier provides an internal 62.5 nsec instruction cycle time from an external 4 MHz clock.
The unit features 4 Stepper Motor Controllers with high current outputs.
Furthermore it features a 6 channel Output Compare Unit and a 6 channel Input Capture Unit with a 16-bit free running timer. Three UARTs constitute additional functionality for communication purposes.

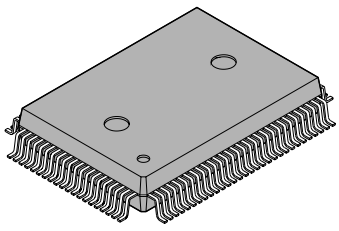
■ FEATURES

- 16-bit core CPU:4MHz external clock (16 MHz internal, 62.5 nsec instr. cycle time)
- New 0.5 µm CMOS Process Technology
- Internal voltage regulator supports 3V MCU core, offering low EMI and low power consumption figures
- Two FULL-CAN interfaces; conforming to Version 2.0 Part A and Part B, flexible message buffering (mailbox and FIFO buffering can be mixed)
- Powerful interrupt functions (8 progr. priority levels; 8 external interrupts)

(Continued)

■ PACKAGE

100-pin Plastic QFP



(FPT-100P-M06)

MB90590 Series

(Continued)

- EI^{2}OS - Automatic transfer function indep.of CPU; 16 ch. of intelligent I/O Services
- 18-bit Time-base counter
- Watchdog Timer
- 3 full duplex UARTs; support 10.4 KBaud (USA standard)
- Serial I/O: 1ch for synchronous data transfer
- A/D Converter: 8 ch. analog inputs (Resolution 10 bits or 8 bits)
- 16-bit reload timer 2ch
- ICU (Input capture) 16bit * 6ch
- OCU (Output capture) 16bit * 6ch
- 16-bit Programmable Pulse Generator 6ch
- Stepping Motor Controller 4ch
- Optimized instruction set for controller applications (bit, byte, word and long-word data types; 23 different addressing modes; barrel shift; variety of pointers)
- 4-byte instruction execution queue
- signed multiply (16bit*16bit) and divide (32bit/16bit) instructions available
- Program Patch Function
- Fast Interrupt processing
- Low Power Consumption - 7 different power saving modes: (Sleep, Stop, CPU intermittent mode, Hardware standby,...)
- Sound Generator
- Real Time Watch Timer
- Package: 100-pin plastic QFP

Controller Area Network (CAN) - License of Robert Bosch GmbH

■ PRODUCT LINEUP

The following table provides a quick outlook of the MB90590 Series

Features	MB90V590A	MB90F594A/MB90F591	MB90594/MB90591
CPU	F²MC-16LX CPU		
System clock	On-chip PLL clock multiplier (× 1, × 2, × 3, × 4, 1/2 when PLL stop) Minimum instruction execution time: 62.5 ns (4 MHz osc. PLL × 4)		
ROM	External	Boot-block Flash memory 256/384 Kbytes Hard-wired reset vector	Mask ROM 256/384 Kbytes
RAM	6 Kbytes	6/8 Kbytes	6/8 Kbytes
Technology	0.5 µm CMOS with on-chip voltage regulator for internal power supply	0.5 µm CMOS with on-chip voltage regulator for internal power supply + Flash memory with On-chip charge pump for programming voltage	0.5 µm CMOS with on-chip voltage regulator for internal power supply
Operating voltage range	5 V ± 10% (Target for MB90F591 and MB90591)		
Temperature range	– 40 to 85 °C		
Package	PGA-256	QFP-100	
UART (3 channels)	Full duplex double buffer Supports asynchronous/synchronous (with start/stop bit) transfer Baud rate: 4808/5208/9615/10417/19230/38460/62500/500000bps (asynchronous) 500K/1M/2Mbps (synchronous) at System clock = 16MHz		
Serial IO	Transfer can be started from MSB or LSB Supports internal clock synchronized transfer and external clock synchronized transfer Supports positive-edge and negative-edge clock synchronization Baud rate : 31.25K/62.5K/125K/500K/1Mbps at System clock = 16MHz		
A/D Converter	10-bit or 8-bit resolution 8 input channels Conversion time: 26.3µs (per one channel)		
16-bit Reload Timer (2 channels)	Operation clock frequency: fsys/2^1, fsys/2^3, fsys/2^5 (fsys = System clock frequency) Supports External Event Count function		
Watch Timer	Directly operates with the oscillation clock Facility to correct oscillation deviation Read/Write accessible Second/Minute/Hour registers Signals interrupts		
16-bit IO Timer	Signals an interrupt when overflow Supports Timer Clear when a match with Output Compare(Channel 0) Operation clock freq.: fsys/2^2, fsys/2^4, fsys/2^6, fsys/2^8(fsys = System clock freq.)		
16-bit Output Compare (6 channels)	Signals an interrupt when a match with 16-bit IO Timer Six 16-bit compare registers A pair of compare registers can be used to generate an output signal		

(Continued)

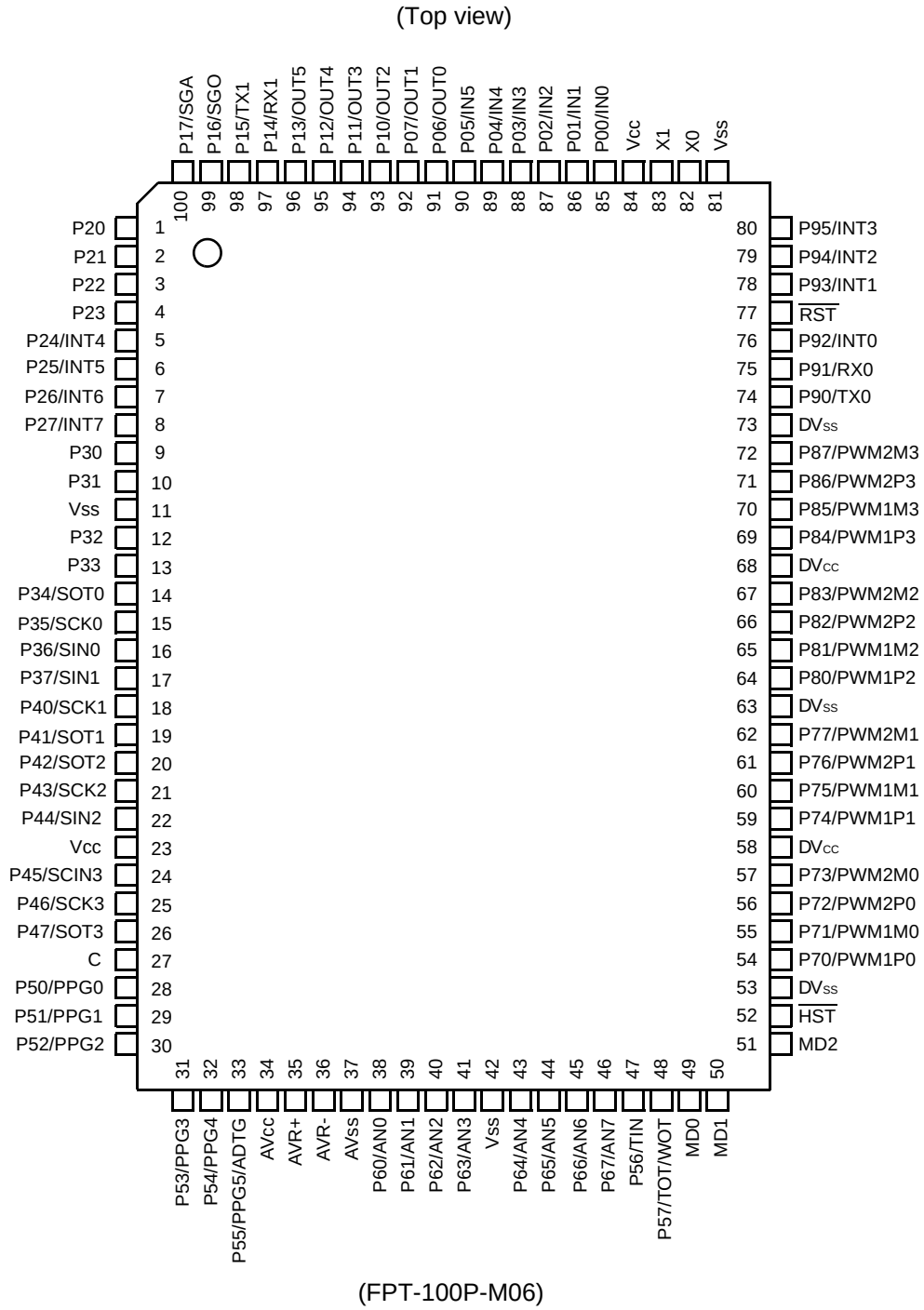
MB90590 Series

(Continued)

Features	MB90V590A	MB90F594A/MB90F591	MB90594/MB90591
16-bit Input Capture (6 channels)	Rising edge, falling edge or rising & falling edge sensitive Six 16-bit Capture registers Signals an interrupt upon external event		
8/16-bit Programma-ble Pulse Gen-erator (6channels)	Supports 8-bit and 16-bit operation modes Twelve 8-bit reload counters Twelve 8-bit reload registers for L pulse width Twelve 8-bit reload registers for H pulse width A pair of 8-bit reload counters can be configured as one 16-bit reload counter or as 8-bit prescaler plus 8-bit reload counter 6 output pins Operation clock freq.: fsys, fsys/2 1 , fsys/2 2 , fsys/2 3 , fsys/2 4 or 128µs@fosc=4MHz (fsys = System clock frequency, fosc = Oscillation clock frequency)		
CAN Interface (2 channels)	Conforms to CAN Specification Version 2.0 Part A and B Automatic re-transmission in case of error Automatic transmission responding to Remote Frame Prioritized 16 message buffers for data and ID's Supports multiple messages Flexible configuration of acceptance filtering: Full bit compare / Full bit mask / Two partial bit masks Supports up to 1Mbps		
Stepper Motor Controller (4 channels)	Four high current outputs for each channel Synchronized two 8-bit PWM's for each channel Succeeds to MB89940 design resource		
External Inter-rupt (8 channels)	Can be programmed edge sensitive or level sensitive		
Sound Gener-ator	8-bit PWM signal is mixed with tone frequency from 8-bit reload counter PWM frequency : 62.5K, 31.2K, 15.6K, 7.8KHz at System clock = 16MHz Tone frequency : PWM frequency / 2 / (reload value + 1)		
IO Ports	Virtually all external pins can be used as general purpose IO All push-pull outputs and schmitt trigger inputs Bit-wise programmable as input/output or peripheral signal		
Flash Memory	—	Supports automatic program-ming,Embedded Algorithm TM * Write/Erase/Erase-Suspend/Re-sume commands A flag indicating completion of the algorithm Number of erase cycles: 10,000 times Data retention time: 10 years Hard-wired reset vector available in order to point to a fixed boot sector in Flash Memory Flash Writer from Minato Electron-ics Inc. Boot block configuration Erase can be performed on each block Block protection with external pro-gramming voltage	—

*: Embedded Algorithm is a trade mark of Advanced Micro Devices Inc.

■ PIN ASSIGNMENT



(FPT-100P-M06)

■ PIN DESCRIPTION

No.	Pin name	Circuit type	Function
82	X0	A	Oscillation input
83	X1		Oscillation output
77	$\overline{\text{RST}}$	B	Reset input
52	$\overline{\text{HST}}$	C	Hardware standby input
85 to 90	P00 to P05	D	General purpose IO
	IN0 to IN5		Inputs for the Input Captures
91 to 96	P06 to P07 P10 to P13	D	General purpose IO
	OUT0 to OUT5		Outputs for the Output Compares. To enable the signal outputs, the corresponding bits of the Port Direction registers should be set to “1”.
97	P14	D	General purpose IO
	RX1		RX input for CAN Interface 1
98	P15	D	General purpose IO
	TX1		TX output for CAN Interface 1. To enable the signal output, the corresponding bit of the Port Direction register should be set to “1”.
99	P16	D	General purpose IO
	SGO		SGO output for the Sound Generator. To enable the signal output, the corresponding bit of the Port Direction register should be set to “1”.
100	P17	D	General purpose IO
	SGA		SGA output for the Sound Generator. To enable the signal output, the corresponding bit of the Port Direction register should be set to “1”.
1 to 4	P20 to P23	D	General purpose IO
5 to 8	P24 to P27	D	General purpose IO
	INT4 to INT7		External interrupt input for INT4 to INT7
9 to 10	P30 to P31	D	General purpose IO
12 to 13	P32 to P33	D	General purpose IO
14	P34	D	General purpose IO
	SOT0		SOT output for UART 0. To enable the signal output, the corresponding bit of the Port Direction register should be set to “1”.
15	P35	D	General purpose IO
	SCK0		SCK input/output for UART 0. To enable the signal output, the corresponding bit of the Port Direction register should be set to “1”.

(Continued)

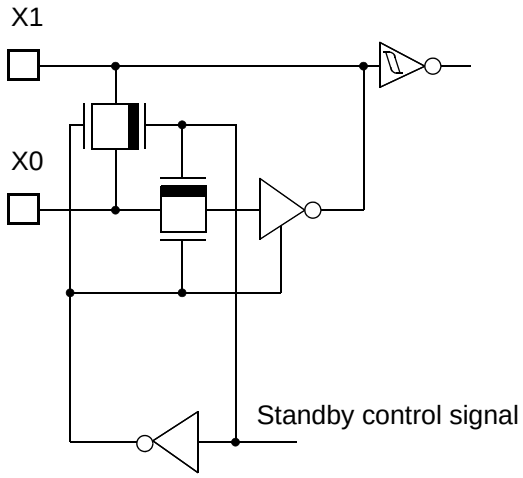
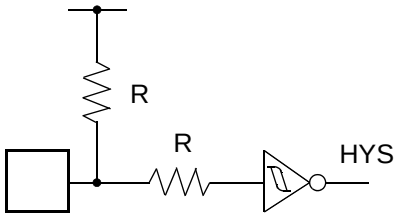
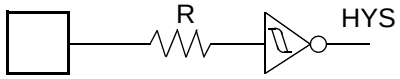
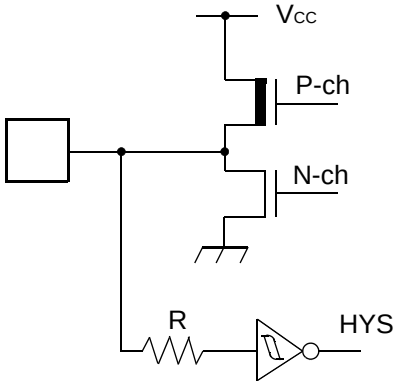
No.	Pin name	Circuit type	Function
16	P36	D	General purpose IO
	SIN0		SIN input for UART 0
17	P37	D	General purpose IO
	SIN1		SIN input for UART 1
18	P40	D	General purpose IO
	SCK1		SCK input/output for UART 1
19	P41	D	General purpose IO
	SOT1		SOT output for UART 1
20	P42	D	General purpose IO
	SOT2		SOT output for UART 2
21	P43	D	General purpose IO
	SCK2		SCK input/output for UART 2
22	P44	D	General purpose IO
	SIN2		SIN input for UART 2
24	P45	D	General purpose IO
	SIN3		SIN input for the Serial IO
25	P46	D	General purpose IO
	SCK3		SCK input/output for the Serial IO
26	P47	D	General purpose IO
	SOT3		SOT output for the Serial IO
28 to 33	P50 to P55	D	General purpose IO
	PPG0 to PPG5, ADTG		Outputs for the Programmable Pulse Generators. Pin number 33 is also shared with ADTG input for the external trigger of the A/D Converter.
38 to 41	P60 to P63	E	General purpose IO
	AN0 to AN3		Inputs for the A/D Converter
43 to 46	P64 to P67	E	General purpose IO
	AN4 to AN7		Inputs for the A/D Converter
47	P56	D	General purpose IO
	TIN		TIN input for the 16-bit Reload Timers
48	P57	D	General purpose IO
	TOT/WOT		TOT output for the 16-bit Reload Timers and WOT output for the Watch Timer. Only one of three output enable flags in these pheriph-eral blocks can be set at a time. Otherwise the output signal has no meaning.

(Continued)

(Continued)

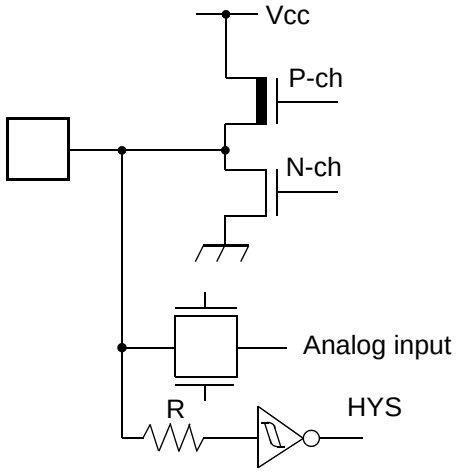
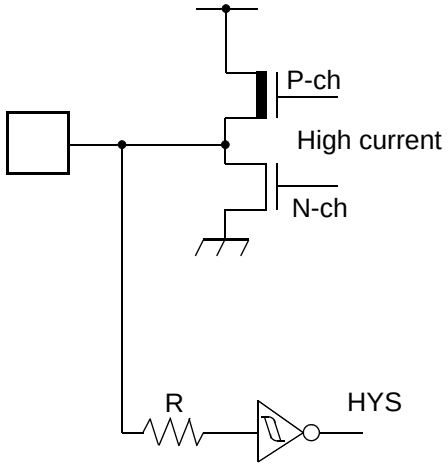
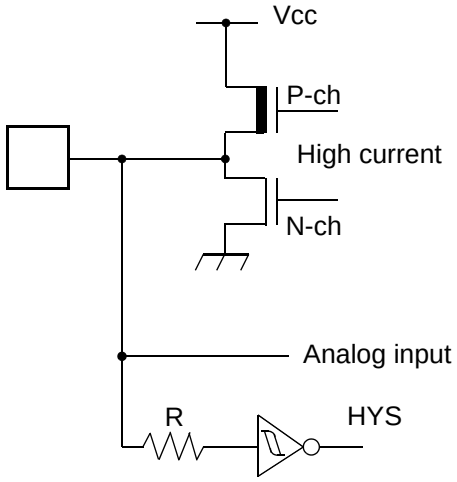
No.	Pin name	Circuit type	Function
54 to 57	P70 to P73	F	General purpose IO
	PWM1P0 PWM1M0 PWM2P0 PWM2M0		Output for Stepping Motor Controller channel 0.
59 to 62	P74 to P77	F	General purpose IO
	PWM1P1 PWM1M1 PWM2P1 PWM2M1		Output for Stepping Motor Controller channel 1.
64 to 67	P80 to P83	F	General purpose IO
	PWM1P2 PWM1M2 PWM2P2 PWM2M2		Output for Stepping Motor Controller channel 2.
69 to 72	P84 to P87	F	General purpose IO
	PWM1P3 PWM1M3 PWM2P3 PWM2M3		Output for Stepping Motor Controller channel 3.
74	P90	D	General purpose IO
	TX0		TX output for CAN Interface 0
75	P91	D	General purpose IO
	RX0		RX input for CAN Interface 0
76	P92	D	General purpose IO
	INT0		External interrupt input for INT0
78	P93	D	General purpose IO
	INT1		External interrupt input for INT1
79	P94	D	General purpose IO
	INT2		External interrupt input for INT2
80	P95	D	General purpose IO
	INT3		External interrupt input for INT3
58 68	DV_{CC}		Dedicated power supply pins for the high current output buffers (Pin No. 54 to 72)
53 63 73	DV_{SS}		Dedicated ground pins for the high current output buffers (Pin No. 54 to 72)

■ I/O CIRCUIT TYPE

Circuit Type	Circuit	Remarks
A		• Oscillation feedback resistor: 1 MΩ approx.
B		• Hysteresis input with pull-up Resistor: 50 kΩ approx.
C		• Hysteresis input
D		• CMOS output • Hysteresis input

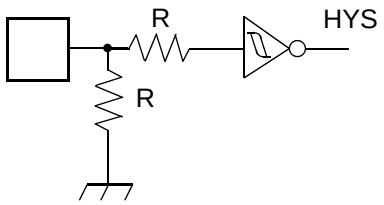
(Continued)

MB90590 Series

Circuit Type	Circuit	Remarks
E		• CMOS output • Hysteresis input • Analog input
F		• CMOS high current output • Hysteresis input
G		• CMOS high current output • Hysteresis input • Analog input

(Continued)

(Continued)

Circuit Type	Circuit	Remarks
H		• Hysteresis input with pull-down Resistor: 50 Kohm approx. • Flash version does not have pull-down register.

MB90590 Series

■ HANDLING DEVICES

(1)Preventing latch-up

CMOS IC chips may suffer latch-up under the following conditions:

- A voltage higher than Vcc or lower than Vss is applied to an input or output pin.
- A voltage higher than the rated voltage is applied between Vcc and Vss.
- The AVcc power supply is applied before the Vcc voltage.

Latch-up may increase the power supply current drastically, causing thermal damage to the device.

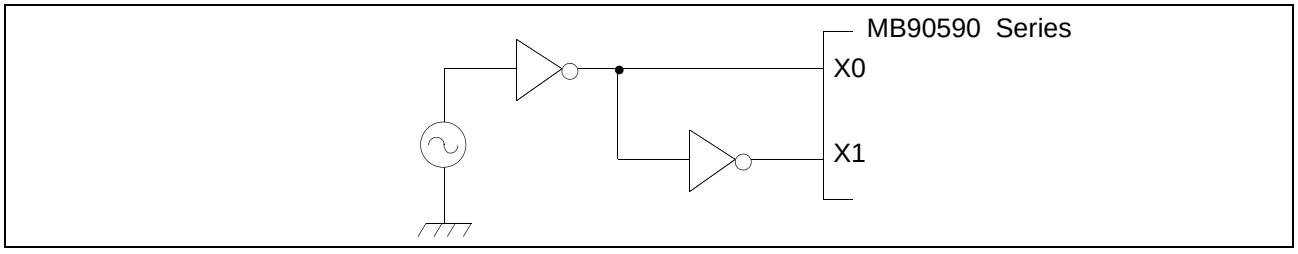
(2)Handling unused input pins

Do not leave unused input pins open, as doing so may cause misoperation of the device. Use a pull-up or pull-down resistor.

(3)Using external clock

To use external clock, drive the X0 and X1 pins in reverse phase.

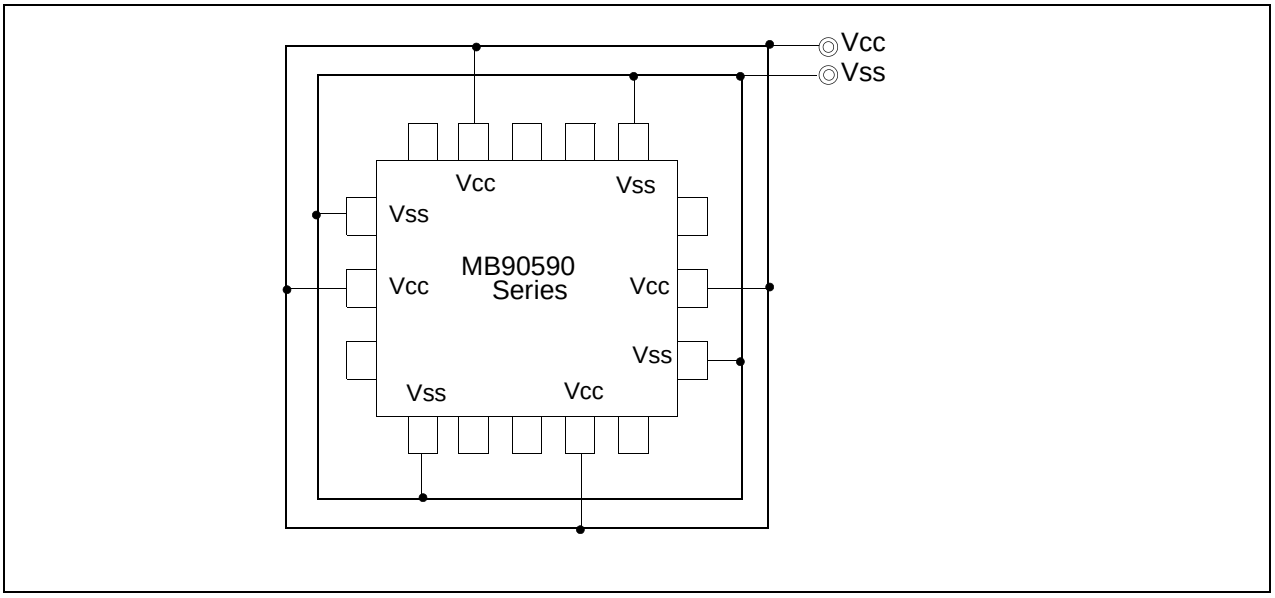
Below is a diagram of how to use external clock.



Using external clock

(4)Power supply pins (Vcc/Vss)

Ensure that all Vcc-level power supply pins are at the same potential. In addition, ensure the same for all Vss-level power supply pins. (See the figure below.) If there are more than one Vcc or Vss system, the device may operate incorrectly even within the guaranteed operating range.



(5) Pull-up/down resistors

The MB90590 Series does not support internal pull-up/down resistors. Use external components where needed.

(6) Crystal Oscillator Circuit

Noises around X0 or X1 pins may be possible causes of abnormal operations. Make sure to provide bypass capacitors via shortest distance from X0, X1 pins, crystal oscillator (or ceramic resonator) and ground lines, and make sure, to the utmost effort, that lines of oscillation circuit not cross the lines of other circuits.

It is highly recommended to provide a printed circuit board art work surrounding X0 and X1 pins with an grand area for stabilizing the operation.

(7) Turning-on Sequence of Power Supply to A/D Converter and Analog Inputs

Make sure to turn on the A/D converter power supply(AV_{CC}, AVR +, AVR –) and analog inputs (AN0 to AN7) after turning-on the digital power supply (V_{CC}).

Turn-off the digital power after turning off the A/D converter supply and analog inputs. In this case, make sure that the voltage not exceed AVR + or AV_{CC} (turning on/off the analog and digital power supplies simultaneously is acceptable).

(8) Connection of Unused Pins of A/D Converter

Connect unused pins of A/D converter to $AV_{CC} = V_{CC}$, AV_{SS} = AVR + = V_{SS}.

(9) N.C. Pin

The N.C. (internally connected) pin must be opened for use.

(10) Notes on Energization

To prevent the internal regulator circuit from malfunctioning, set the voltage rise time during energization at 50 or more ms (0.2 V to 2.7 V).

(11) Initialization

In the device, there are internal registers which is initialized only by a power-on reset. To initialize these registers turning on the power again.

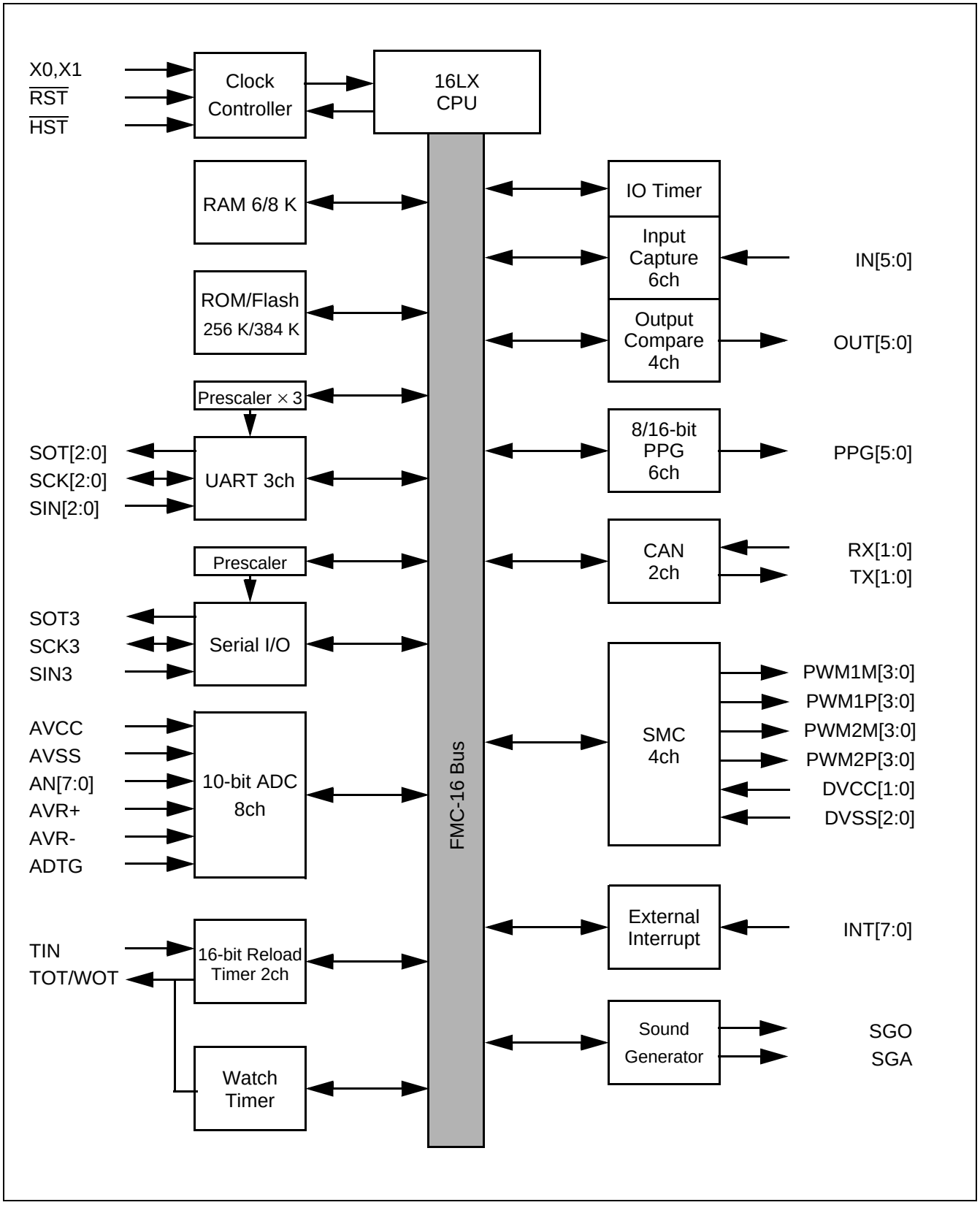
(12) Directions of “DIV A, Ri” and “DIVW A, RWi” instructions

In the Signed multiplication and division instructions (“DIV A, Ri” and “DIVW A, RWi”), the value of the corresponding bank register (DTB, ADB, USB, SSB) is set in “00h”.

If the values of the corresponding bank register (DTB,ADB,USB,SSB) are setting other than “00h”, the remainder by the execution result of the instruction is not stored in the register of the instruction operand.

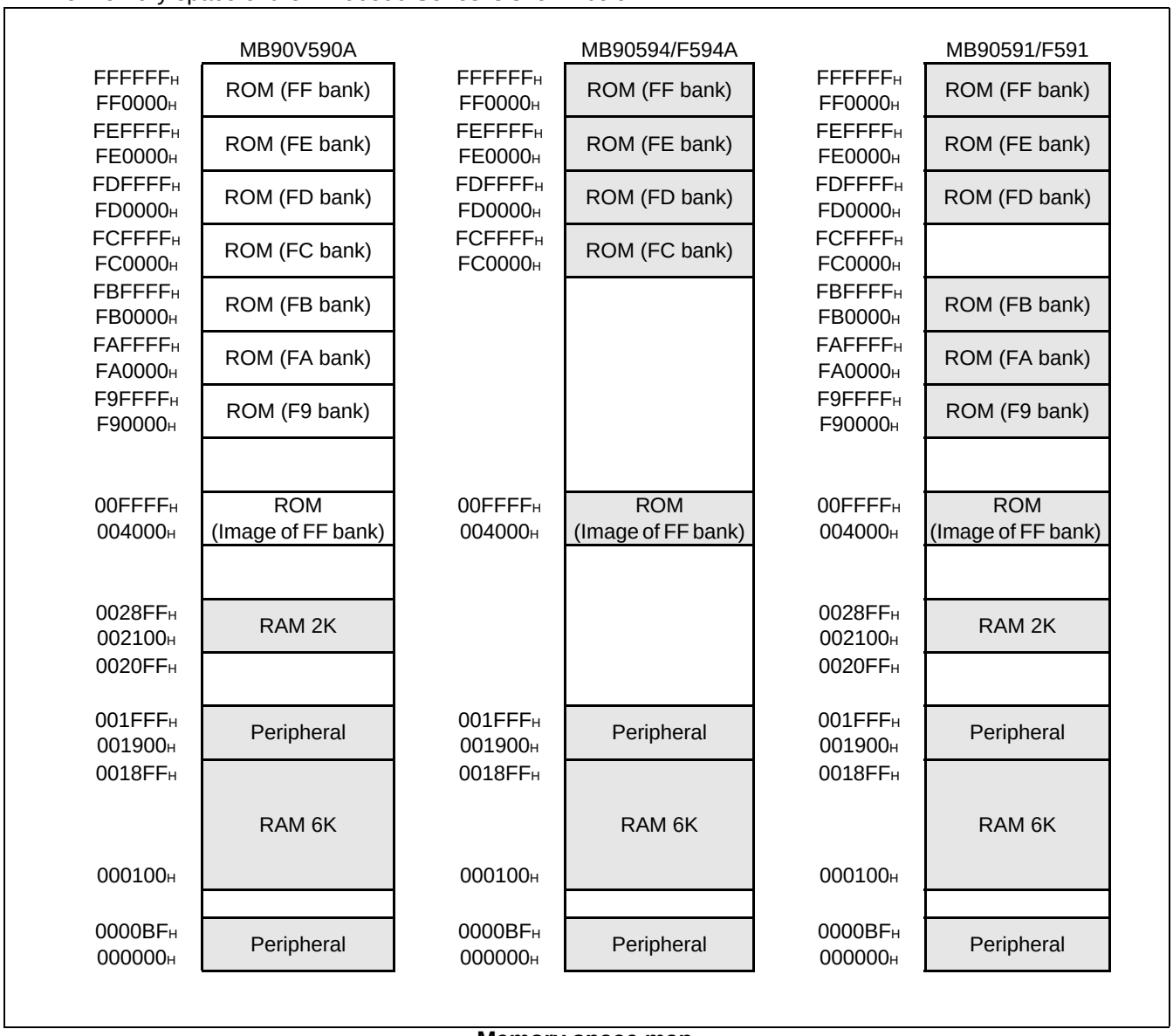
MB90590 Series

■ BLOCK DIAGRAM



■ MEMORY SPACE

The memory space of the MB90590 Series is shown below



Memory space map

The high-order portion of bank 00 gives the image of the FF bank ROM to make the small model of the C compiler effective. Since the low-order 16 bits are the same, the table in ROM can be referenced without using the far specification in the pointer declaration.
For example, an attempt to access 00C000H accesses the value at FFC000H in ROM.
The ROM area in bank FF exceeds 48 Kbytes, and its entire image cannot be shown in bank 00.
The image between FF4000H and FFFFFFH is visible in bank 00, while the image between FF0000H and FF3FFFH is visible only in bank FF.

■ I/O MAP

Address	Register	Abbreviation	Access	Pripheral	Initial value
00 H	Port 0 data register	PDR0	R/W	Port 0	XXXXXXXXB
01 H	Port 1 data register	PDR1	R/W	Port 1	XXXXXXXXB
02 H	Port 2 data register	PDR2	R/W	Port 2	XXXXXXXXB
03 H	Port 3 data register	PDR3	R/W	Port 3	XXXXXXXXB
04 H	Port 4 data register	PDR4	R/W	Port 4	XXXXXXXXB
05 H	Port 5 data register	PDR5	R/W	Port 5	XXXXXXXXB
06 H	Port 6 data register	PDR6	R/W	Port 6	XXXXXXXXB
07 H	Port 7 data register	PDR7	R/W	Port 7	XXXXXXXXB
08 H	Port 8 data register	PDR8	R/W	Port 8	XXXXXXXXB
09 H	Port 9 data register	PDR9	R/W	Port 9	_ _ XXXXXXB
0A to 0F H	Reserved				
10 H	Port 0 direction register	DDR0	R/W	Port 0	0 0 0 0 0 0 0 0B
11 H	Port 1 direction register	DDR1	R/W	Port 1	0 0 0 0 0 0 0 0B
12 H	Port 2 direction register	DDR2	R/W	Port 2	0 0 0 0 0 0 0 0B
13 H	Port 3 direction register	DDR3	R/W	Port 3	0 0 0 0 0 0 0 0B
14 H	Port 4 direction register	DDR4	R/W	Port 4	0 0 0 0 0 0 0 0B
15 H	Port 5 direction register	DDR5	R/W	Port 5	0 0 0 0 0 0 0 0B
16 H	Port 6 direction register	DDR6	R/W	Port 6	0 0 0 0 0 0 0 0B
17 H	Port 7 direction register	DDR7	R/W	Port 7	0 0 0 0 0 0 0 0B
18 H	Port 8 direction register	DDR8	R/W	Port 8	0 0 0 0 0 0 0 0B
19 H	Port 9 direction register	DDR9	R/W	Port 9	_ _ 0 0 0 0 0 0B
1A H	Reserved				
1B H	Analog Input Enable	ADER	R/W	Port 6, A/D	1 1 1 1 1 1 1 1B
1C to 1F H	Reserved				
20 H	Serial Mode Control 0	UMC0	R/W	UART0	0 0 0 0 0 1 0 0B
21 H	Status 0	USR0	R/W		0 0 0 1 0 0 0 0B
22 H	Input/Output Data 0	UIDR0/ UODR0	R/W		XXXXXXXXB
23 H	Rate and Datar 0	URD0	R/W		0 0 0 0 0 0 0XB
24 H	Serial Mode Control 1	UMC1	R/W	UART1	0 0 0 0 0 1 0 0B
25 H	Status 1	USR1	R/W		0 0 0 1 0 0 0 0B
26 H	Input/Output Data 1	UIDR1/ UODR1	R/W		XXXXXXXXB
27 H	Rate and Datar 1	URD1	R/W		0 0 0 0 0 0 0XB

(Continued)

Address	Register	Abbreviation	Access	Peripheral	Initial value
28 H	Serial Mode Control 2	UMC2	R/W	UART2	0 0 0 0 0 1 0 0B
29 H	Status 2	USR2	R/W		0 0 0 1 0 0 0 0B
2A H	Input/Output Data 2	UIDR2/ UODR2	R/W		XXXXXXXXB
2B H	Rate and Datar 2	URD2	R/W		0 0 0 0 0 0 0XB
2C H	Serial Mode Control	SMCS	R/W	Serial IO	_ _ _ _0 0 0 0B
2D H	Serial Mode Control	SMCS	R/W		0 0 0 0 0 0 1 0B
2E H	Serial Data	SDR	R/W		XXXXXXXXB
2F H	Edge Selector	SES	R/W		_ _ _ _ _ _ _ 0B
30 H	External Interrupt Enable	ENIR	R/W	External Interrupt	0 0 0 0 0 0 0 0B
31 H	External Interrupt Request	EIRR	R/W		XXXXXXXXB
32 H	External Interrupt Level	ELVR	R/W		0 0 0 0 0 0 0 0B
33 H	External Interrupt Level	ELVR	R/W		0 0 0 0 0 0 0 0B
34 H	A/D Control Status 0	ADCS0	R/W	A/D Converter	0 0 0 0 0 0 0 0B
35 H	A/D Control Status 1	ADCS1	R/W		0 0 0 0 0 0 0 0B
36 H	A/D Data 0	ADCR0	R		XXXXXXXXB
37 H	A/D Data 1	ADCR1	R/W		0 0 0 0 1 0 XXB
38 H	PPG0 operation mode control register	PPGC0	R/W	16-bit Programable Pulse Generator 0/1	0 _ 0 0 0 _ _ 1B
39 H	PPG1 operation mode control register	PPGC1	R/W		0 _ 0 0 0 0 0 1B
3A H	PPG0 and PPG1 clock select register	PPG01	R/W		0 0 0 0 0 0 0 0B
3B H	Reserved				
3C H	PPG2 operation mode control register	PPGC2	R/W	16-bit Programable Pulse Generator 2/3	0 _ 0 0 0 _ _ 1B
3D H	PPG3 operation mode control register	PPGC3	R/W		0 _ 0 0 0 0 0 1B
3E H	PPG2 and PPG3 clock select register	PPG23	R/W		0 0 0 0 0 0 0 0B
3F H	Reserved				
40 H	PPG4 operation mode control register	PPGC4	R/W	16-bit Programable Pulse Generator 4/5	0 _ 0 0 0 _ _ 1B
41 H	PPG5 operation mode control register	PPGC5	R/W		0 _ 0 0 0 0 0 1B
42 H	PPG4 and PPG5 clock select register	PPG45	R/W		0 0 0 0 0 0 0 0B
43 H	Reserved				
44 H	PPG6 operation mode control register	PPGC6	R/W	16-bit Programable Pulse Generator 6/7	0 _ 0 0 0 _ _ 1B
45 H	PPG7 operation mode control register	PPGC7	R/W		0 _ 0 0 0 0 0 1B
46 H	PPG6 and PPG7 clock select register	PPG67	R/W		0 0 0 0 0 0 0 0B
47 H	Reserved				

(Continued)

Address	Register	Abbreviation	Access	Peripheral	Initial value
48 H	PPG8 operation mode control register	PPGC8	R/W	16-bit Programable Pulse Generator 8/9	0 _ 0 0 0 _ _ 1B
49 H	PPG9 operation mode controlregister	PPGC9	R/W		0 _ 0 0 0 0 0 1B
4A H	PPG8 and PPG9 clock select register	PPG89	R/W		0 0 0 0 0 0 0 0B
4B H	Reserved				
4C H	PPGA operation mode control register	PPGCA	R/W	16-bit Programable Pulse Generator A/B	0 _ 0 0 0 _ _ 1B
4D H	PPGB operation mode control register	PPGCB	R/W		0 _ 0 0 0 0 0 1B
4E H	PPGA and PPGB clock select register	PPGAB	R/W		0 0 0 0 0 0 0 0B
4F H	Reserved				
50 H	Timer Control Status 0	TMCSR0	R/W	16-bit Reload Timer 0	0 0 0 0 0 0 0 0B
51 H	Timer Control Status 0	TMCSR0	R/W		_ _ _ _ 0 0 0 0B
52 H	Timer Control Status 1	TMCSR1	R/W	16-bit Reload Timer 1	0 0 0 0 0 0 0 0B
53 H	Timer Control Status 1	TMCSR1	R/W		_ _ _ _ 0 0 0 0B
54 H	Input Captue Control Status 0/1	ICS01	R/W	Input Capture 0/1	0 0 0 0 0 0 0 0B
55 H	Input Captue Control Status 2/3	ICS23	R/W	Input Capture 2/3	0 0 0 0 0 0 0 0B
56 H	Input Captue Control Status 4/5	ICS45	R/W	Input Capture 4/5	0 0 0 0 0 0 0 0B
57 H	Reserved				
58 H	Output Compare Control Status 0	OCS0	R/W	Output Compare 0/1	0 0 0 0 _ _ 0 0B
59 H	Output Compare Control Status 1	OCS1	R/W		_ _ _0 0 0 0 0B
5A H	Output Compare Control Status 2	OCS2	R/W	Output Compare 2/3	0 0 0 0 _ _ 0 0B
5B H	Output Compare Control Status 3	OCS3	R/W		_ _ _ 0 0 0 0 0B
5C H	Output Compare Control Status 4	OCS4	R/W	Output Compare 4/5	0 0 0 0 _ _ 0 0B
5D H	Output Compare Control Status 5	OCS5	R/W		_ _ _ 0 0 0 0 0B
5E H	Sound Control	SGCR	R/W	Sound Generator	0 0 0 0 0 0 0 0B
5F H	Sound Control	SGCR	R/W		0 _ _ _ _ _ _ 0B
60 H	Watch Timer Control	WTCR	R/W	Watch Timer	0 0 0 _ _ 0 0 0B
61 H	Watch Timer Control	WTCR	R/W		0 0 0 0 0 0 0 0B
62 H	PWM Control 0	PWC0	R/W	Stepping Motor Controller 0	0 0 0 0 0 _ _ 0B
63 H	Reserved				
64 H	PWM Control 1	PWC1	R/W	Stepping Motor Controller 1	0 0 0 0 0 _ _ 0B
65 H	Reserved				
66 H	PWM Control 2	PWC2	R/W	Stepping Motor Controller 2	0 0 0 0 0 _ _ 0B
67 H	Reserved				
68 H	PWM Control 3	PWC3	R/W	Stepping Motor Controller 3	0 0 0 0 0 _ _ 0B

(Continued)

Address	Register	Abbreviation	Access	Peripheral	Initial value
69 to 6C H	Reserved				
6D H	Serial IO Prescaler	CDCR	R/W	Prescaler (Serial IO)	0 XXX 1 1 1 1B
6E H	Timer Control	TCCS	R/W	I/O Timer	0 0 0 0 0 0 0 0B
6F H	ROM Mirror	ROMM	W	ROM Mirror	XXXXXXX1B
70 to 8F H	Reserved for CAN Interface 0/1. Refer to section about CAN Controller				
90 to 9D H	Reserved				
9E H	ROM Correction Control Status	PACSR	R/W	ROM Correction	0 0 0 0 0 0 0 0B
9F H	Delayed Interrupt/release	DIRR	R/W	Delayed Interrupt	_ _ _ _ _ _ _ 0B
A0 H	Low-power Mode	LPMCR	R/W	Low Power Controller	0 0 0 1 1 0 0 0B
A1 H	Clock Selector	CKSCR	R/W	Low Power Controller	1 1 1 1 1 1 0 0B
A2 to A7 H	Reserved				
A8 H	Watchdog Control	WDTC	R/W	Watchdog Timer	XXXXX 1 1 1B
A9 H	Time Base Timer Control	TBTC	R/W	Time Base Timer	1 - - 0 0 1 0 0B
AA to AD H	Reserved				
AE H	Flash Control Status (MB90F594 only. Otherwise reserved)	FMCS	R/W	Flash Memory	0 0 0 X 0 _ _ 0B
AF H	Reserved				
B0 H	Interrupt control register 00	ICR00	R/W	Interrupt controller	0 0 0 0 0 1 1 1B
B1 H	Interrupt control register 01	ICR01	R/W		0 0 0 0 0 1 1 1B
B2 H	Interrupt control register 02	ICR02	R/W		0 0 0 0 0 1 1 1B
B3 H	Interrupt control register 03	ICR03	R/W		0 0 0 0 0 1 1 1B
B4 H	Interrupt control register 04	ICR04	R/W		0 0 0 0 0 1 1 1B
B5 H	Interrupt control register 05	ICR05	R/W		0 0 0 0 0 1 1 1B
B6 H	Interrupt control register 06	ICR06	R/W		0 0 0 0 0 1 1 1B
B7 H	Interrupt control register 07	ICR07	R/W		0 0 0 0 0 1 1 1B
B8 H	Interrupt control register 08	ICR08	R/W		0 0 0 0 0 1 1 1B
B9 H	Interrupt control register 09	ICR09	R/W		0 0 0 0 0 1 1 1B
BA H	Interrupt control register 10	ICR10	R/W		0 0 0 0 0 1 1 1B
BB H	Interrupt control register 11	ICR11	R/W		0 0 0 0 0 1 1 1B
BC H	Interrupt control register 12	ICR12	R/W		0 0 0 0 0 1 1 1B
BD H	Interrupt control register 13	ICR13	R/W		0 0 0 0 0 1 1 1B
BE H	Interrupt control register 14	ICR14	R/W		0 0 0 0 0 1 1 1B
BF H	Interrupt control register 15	ICR15	R/W		0 0 0 0 0 1 1 1B

(Continued)

Address	Register	Abbreviation	Access	Peripheral	Initial value
1900 H	Reload L	PRLL0	R/W	16-bit Programable Pulse Generator 0/1	XXXXXXXXB
1901 H	Reload H	PRLH0	R/W		XXXXXXXXB
1902 H	Reload L	PRLL1	R/W		XXXXXXXXB
1903 H	Reload H	PRLH1	R/W		XXXXXXXXB
1904 H	Reload L	PRLL2	R/W	16-bit Programable Pulse Generator 2/3	XXXXXXXXB
1905 H	Reload H	PRLH2	R/W		XXXXXXXXB
1906 H	Reload L	PRLL3	R/W		XXXXXXXXB
1907 H	Reload H	PRLH3	R/W		XXXXXXXXB
1908 H	Reload L	PRLL4	R/W	16-bit Programable Pulse Generator 4/5	XXXXXXXXB
1909 H	Reload H	PRLH4	R/W		XXXXXXXXB
190A H	Reload L	PRLL5	R/W		XXXXXXXXB
190B H	Reload H	PRLH5	R/W		XXXXXXXXB
190C H	Reload L	PRLL6	R/W	16-bit Programable Pulse Generator 6/7	XXXXXXXXB
190D H	Reload H	PRLH6	R/W		XXXXXXXXB
190E H	Reload L	PRLL7	R/W		XXXXXXXXB
190F H	Reload H	PRLH7	R/W		XXXXXXXXB
1910 H	Reload L	PRLL8	R/W	16-bit Programable Pulse Generator 8/9	XXXXXXXXB
1911 H	Reload H	PRLH8	R/W		XXXXXXXXB
1912 H	Reload L	PRLL9	R/W		XXXXXXXXB
1913 H	Reload H	PRLH9	R/W		XXXXXXXXB
1914 H	Reload L	PRLLA	R/W	16-bit Programable Pulse Generator A/B	XXXXXXXXB
1915 H	Reload H	PRLHA	R/W		XXXXXXXXB
1916 H	Reload L	PRLLB	R/W		XXXXXXXXB
1917 H	Reload H	PRLHB	R/W		XXXXXXXXB
1918 to 191F H	Reserved				
1920 H	Input Capture 0	IPCP0	R	Input Capture 0/1	XXXXXXXXB
1921 H	Input Capture 0	IPCP0	R		XXXXXXXXB
1922 H	Input Capture 1	IPCP1	R		XXXXXXXXB
1923 H	Input Capture 1	IPCP1	R		XXXXXXXXB
1924 H	Input Capture 2	IPCP2	R	Input Capture 2/3	XXXXXXXXB
1925 H	Input Capture 2	IPCP2	R		XXXXXXXXB
1926 H	Input Capture 3	IPCP3	R		XXXXXXXXB
1927 H	Input Capture 3	IPCP3	R		XXXXXXXXB

(Continued)

Address	Register	Abbreviation	Access	Peripheral	Initial value
1928 H	Input Capture 4	IPCP4	R	Input Capture 4/5	XXXXXXXXB
1929 H	Input Capture 4	IPCP4	R		XXXXXXXXB
192A H	Input Capture 5	IPCP5	R		XXXXXXXXB
192B H	Input Capture 5	IPCP5	R		XXXXXXXXB
192C to 192F H	Reserved				
1930 H	Output Compare 0	OCCP0	R/W	Output Compare 0/1	XXXXXXXXB
1931 H	Output Compare 0	OCCP0	R/W		XXXXXXXXB
1932 H	Output Compare 1	OCCP1	R/W		XXXXXXXXB
1933 H	Output Compare 1	OCCP1	R/W		XXXXXXXXB
1934 H	Output Compare 2	OCCP2	R/W	Output Compare 2/3	XXXXXXXXB
1935 H	Output Compare 2	OCCP2	R/W		XXXXXXXXB
1936 H	Output Compare 3	OCCP3	R/W		XXXXXXXXB
1937 H	Output Compare 3	OCCP3	R/W		XXXXXXXXB
1938 H	Output Compare 4	OCCP4	R/W	Output Compare 4/5	XXXXXXXXB
1939 H	Output Compare 4	OCCP4	R/W		XXXXXXXXB
193A H	Output Compare 5	OCCP5	R/W		XXXXXXXXB
193B H	Output Compare 5	OCCP5	R/W		XXXXXXXXB
193C to 193F H	Reserved				
1940 H	Timer 0/Reload 0	TMR0/ TMRLR0	R/W	16-bit Reload Timer 0	XXXXXXXXB
1941 H	Timer 0/Reload 0	TMR0/ TMRLR0	R/W		XXXXXXXXB
1942 H	Timer 1/Reload 1	TMR1/ TMRLR1	R/W	16-bit Reload Timer 1	XXXXXXXXB
1943 H	Timer 1/Reload 1	TMR1/ TMRLR1	R/W		XXXXXXXXB
1944 H	Timer Data	TCDT	R/W	IO Timer	00000000B
1945 H	Timer Data	TCDT	R/W		00000000B
1946 H	Frequency Dtata	SGFR	R/W	Sound Generator	XXXXXXXXB
1947 H	Amplitude Data	SGAR	R/W		XXXXXXXXB
1948 H	Decrement Grade	SGDR	R/W		XXXXXXXXB
1949 H	Tone Count	SGTR	R/W		XXXXXXXXB
194A H	Sub-second Data	WTBR	R/W	Watch Timer	XXXXXXXXB
194B H	Sub-second Data	WTBR	R/W		XXXXXXXXB
194C H	Sub-second Data	WTBR	R/W		___XXXXXB
194D H	Second Data	WTSR	R/W		__000000B

(Continued)

(Continued)

Address	Register	Abbreviation	Access	Peripheral	Initial value
194E H	Minute Data	WTMR	R/W	Watch Timer	__000000B
194F H	Hour Data	WTHR	R/W		___00000B
1950 H	PWM1 Compare 0	PWC10	R/W	Stepping Motor Controller 0	XXXXXXXXB
1951 H	PWM2 Compare 0	PWC20	R/W		XXXXXXXXB
1952 H	PWM1 Select 0	PWS10	R/W		__000000B
1953 H	PWM2 Select 0	PWS20	R/W		_0000000B
1954 H	PWM1 Compare 1	PWC11	R/W	Stepping Motor Controller 1	XXXXXXXXB
1955 H	PWM2 Compare 1	PWC21	R/W		XXXXXXXXB
1956 H	PWM1 Select 1	PWS11	R/W		__000000B
1957 H	PWM2 Select 1	PWS21	R/W		_0000000B
1958 H	PWM1 Compare 2	PWC12	R/W	Stepping Motor Controller 2	XXXXXXXXB
1959 H	PWM2 Compare 2	PWC22	R/W		XXXXXXXXB
195A H	PWM1 Select 2	PWS12	R/W		__000000B
195B H	PWM2 Select 2	PWS22	R/W		_0000000B
195C H	PWM1 Compare 3	PWC13	R/W	Stepping Motor Controller 3	XXXXXXXXB
195D H	PWM2 Compare 3	PWC23	R/W		XXXXXXXXB
195E H	PWM1 Select 3	PWS13	R/W		__000000B
195F H	PWM2 Select 3	PWS23	R/W		_0000000B
1960 to 19FF H	Reserved				
1A00 to 1AFF H	Reserved for CAN Interface 0. Refer to section about CAN Controller				
1B00 to 1BFF H	Reserved for CAN Interface 1. Refer to section about CAN Controller				
1C00 to 1CFF H	Reserved for CAN Interface 0. Refer to section about CAN Controller				
1D00 to 1DFF H	Reserved for CAN Interface 1. Refer to section about CAN Controller				
1E00 to 1EFF H	Reserved				
1FF0 H	ROM Correction Address 0	PADR0	R/W	ROM Correction	XXXXXXXX B
1FF1 H	ROM Correction Address 1	PADR0	R/W		XXXXXXXX B
1FF2 H	ROM Correction Address 2	PADR0	R/W		XXXXXXXX B
1FF3 H	ROM Correction Address 3	PADR1	R/W		XXXXXXXX B
1FF4 H	ROM Correction Address 4	PADR1	R/W		XXXXXXXX B
1FF5 H	ROM Correction Address 5	PADR1	R/W		XXXXXXXX B
1FF6 to 1FFF H	Reserved				

Note Initial value of "_" represents unused bit, "X" represents unknown valu
Addresses in the rage 0000H to 00FFH, which are not listed in the table, are reserved for the primary functions of the MCU. A read access to these reserved addresses results reading "X" and any write access should not be performed.

■ CAN CONTROLLERS

The CAN controller has the following features:

- Conforms to CAN Specification Version 2.0 Part A and B
  - Supports transmission/reception in standard frame and extended frame formats
- Supports transmitting of data frames by receiving remote frames
- 16 transmitting/receiving message buffers
  - 29-bit ID and 8-byte data
  - Multi-level message buffer configuration
- Provides full-bit comparison, full-bit mask, acceptance register 0/acceptance register 1 for each message buffer as 1D acceptance mask
  - Two acceptance mask registers in either standard frame format or extended frame formats
- Bit rate programmable from 10 Kbits/s to 2 Mbits/s (when input clock is at 16 MHz)

List of Control Registers

Address		Register	Abbreviation	Access	Initial Value
CAN0	CAN1				
000070_H	000080_H	Message buffer valid register	BVALR	R/W	$00000000\ 00000000_B$
000071_H	000081_H				
000072_H	000082_H	Transmit request register	TREQR	R/W	$00000000\ 00000000_B$
000073_H	000083_H				
000074_H	000084_H	Transmit cancel register	TCANR	W	$00000000\ 00000000_B$
000075_H	000085_H				
000076_H	000086_H	Transmit complete register	TCR	R/W	$00000000\ 00000000_B$
000077_H	000087_H				
000078_H	000088_H	Receive complete register	RCR	R/W	$00000000\ 00000000_B$
000079_H	000089_H				
$00007A_H$	$00008A_H$	Remote request receiving register	RRTRR	R/W	$00000000\ 00000000_B$
$00007B_H$	$00008B_H$				
$00007C_H$	$00008C_H$	Receive overrun register	ROVRR	R/W	$00000000\ 00000000_B$
$00007D_H$	$00008D_H$				
$00007E_H$	$00008E_H$	Receive interrupt enable register	RIER	R/W	$00000000\ 00000000_B$
$00007F_H$	$00008F_H$				

List of Control Registers

Address		Register	Abbreviation	Access	Initial Value
CAN0	CAN1				
001C00H	001D00H	Control status register	CSR	R/W, R	00---000 0----0-1B
001C01H	001D01H				
001C02H	001D02H	Last event indicator register	LEIR	R/W	-------- 000-0000B
001C03H	001D03H				
001C04H	001D04H	Receive/transmit error counter	RTEC	R	00000000 00000000B
001C05H	001D05H				
001C06H	001D06H	Bit timing register	BTR	R/W	-1111111 11111111B
001C07H	001D07H				
001C08H	001D08H	IDE register	IDER	R/W	XXXXXXXX XXXXXXXXB
001C09H	001D09H				
001C0AH	001D0AH	Transmit RTR register	TRTRR	R/W	00000000 00000000B
001C0BH	001D0BH				
001C0CH	001D0CH	Remote frame receive waiting register	RFWTR	R/W	XXXXXXXX XXXXXXXXB
001C0DH	001D0DH				
001C0EH	001D0EH	Transmit interrupt enable register	TIER	R/W	00000000 00000000B
001C0FH	001D0FH				
001C10H	001D10H	Acceptance mask select register	AMSR	R/W	XXXXXXXX XXXXXXXXB
001C11H	001D11H				
001C12H	001D12H				XXXXXXXX XXXXXXXXB
001C13H	001D13H				
001C14H	001D14H	Acceptance mask register 0	AMR0	R/W	XXXXXXXX XXXXXXXXB
001C15H	001D15H				
001C16H	001D16H				XXXXX--- XXXXXXXXB
001C17H	001D17H				
001C18H	001D18H	Acceptance mask register 1	AMR1	R/W	XXXXXXXX XXXXXXXXB
001C19H	001D19H				
001C1AH	001D1AH				XXXXX--- XXXXXXXXB
001C1BH	001D1BH				

List of Message Buffers (ID Registers)

Address		Register	Abbreviation	Access	Initial Value
CAN0	CAN1				
001A20H	001B20H	ID register 0	IDR0	R/W	XXXXXXXX XXXXXXXXB
001A21H	001B21H				
001A22H	001B22H				XXXXX--- XXXXXXXXB
001A23H	001B23H				
001A24H	001B24H	ID register 1	IDR1	R/W	XXXXXXXX XXXXXXXXB
001A25H	001B25H				
001A26H	001B26H				XXXXX--- XXXXXXXXB
001A27H	001B27H				
001A28H	001B28H	ID register 2	IDR2	R/W	XXXXXXXX XXXXXXXXB
001A29H	001B29H				
001A2AH	001B2AH				XXXXX--- XXXXXXXXB
001A2BH	001B2BH				
001A2CH	001B2CH	ID register 3	IDR3	R/W	XXXXXXXX XXXXXXXXB
001A2DH	001B2DH				
001A2EH	001B2EH				XXXXX--- XXXXXXXXB
001A2FH	001B2FH				
001A30H	001B30H	ID register 4	IDR4	R/W	XXXXXXXX XXXXXXXXB
001A31H	001B31H				
001A32H	001B32H				XXXXX--- XXXXXXXXB
001A33H	001B33H				
001A34H	001B34H	ID register 5	IDR5	R/W	XXXXXXXX XXXXXXXXB
001A35H	001B35H				
001A36H	001B36H				XXXXX--- XXXXXXXXB
001A37H	001B37H				
001A38H	001B38H	ID register 6	IDR6	R/W	XXXXXXXX XXXXXXXXB
001A39H	001B39H				
001A3AH	001B3AH				XXXXX--- XXXXXXXXB
001A3BH	001B3BH				
001A3CH	001B3CH	ID register 7	IDR7	R/W	XXXXXXXX XXXXXXXXB
001A3DH	001B3DH				
001A3EH	001B3EH				XXXXX--- XXXXXXXXB
001A3FH	001B3FH				

(Continued)

(Continued)

Address		Register	Abbreviation	Access	Initial Value
CAN0	CAN1				
001A40H	001B40H	ID register 8	IDR8	R/W	XXXXXXXX XXXXXXXXB
001A41H	001B41H				
001A42H	001B42H				XXXXX--- XXXXXXXXB
001A43FH	001B43H				
001A44H	001B44H	ID register 9	IDR9	R/W	XXXXXXXX XXXXXXXXB
001A45H	001B45H				
001A46H	001B46H				XXXXX--- XXXXXXXXB
001A47H	001B47H				
001A48H	001B48H	ID register 10	IDR10	R/W	XXXXXXXX XXXXXXXXB
001A49H	001B49H				
001A4AH	001B4AH				XXXXX--- XXXXXXXXB
001A4BH	001B4BH				
001A4CH	001B4CH	ID register 11	IDR11	R/W	XXXXXXXX XXXXXXXXB
001A4DH	001B4DH				
001A4EH	001B4EH				XXXXX--- XXXXXXXXB
001A4FH	001B4FH				
001A50H	001B50H	ID register 12	IDR12	R/W	XXXXXXXX XXXXXXXXB
001A51H	001B51H				
001A52H	001B52H				XXXXX--- XXXXXXXXB
001A53H	001B53H				
001A54H	001B54H	ID register 13	IDR13	R/W	XXXXXXXX XXXXXXXXB
001A55H	001B55H				
001A56H	001B56H				XXXXX--- XXXXXXXXB
001A57H	001B57H				
001A58H	001B58H	ID register 14	IDR14	R/W	XXXXXXXX XXXXXXXXB
001A59H	001B59H				
001A5AH	001B5AH				XXXXX--- XXXXXXXXB
001A5BH	001B5BH				
001A5CH	001B5CH	ID register 15	IDR15	R/W	XXXXXXXX XXXXXXXXB
001A5DH	001B5DH				
001A5EH	001B5EH				XXXXX--- XXXXXXXXB
001A5FH	001B5FH				

List of Message Buffers (DLC Registers and Data Registers)

Address		Register	Abbreviation	Access	Initial Value
CAN0	CAN1				
001A60H	001B60H	DLC register 0	DLCR0	R/W	----XXXXB
001A61H	001B61H				
001A62H	001B62H	DLC register 1	DLCR1	R/W	----XXXXB
001A63H	001B63H				
001A64H	001B64H	DLC register 2	DLCR2	R/W	----XXXXB
001A65H	001B65H				
001A66H	001B66H	DLC register 3	DLCR3	R/W	----XXXXB
001A67H	001B67H				
001A68H	001B68H	DLC register 4	DLCR4	R/W	----XXXXB
001A69H	001B69H				
001A6AH	001B6AH	DLC register 5	DLCR5	R/W	----XXXXB
001A6BH	001B6BH				
001A6CH	001B6CH	DLC register 6	DLCR6	R/W	----XXXXB
001A6DH	001B6DH				
001A6EH	001B6EH	DLC register 7	DLCR7	R/W	----XXXXB
001A6FH	001B6FH				
001A70H	001B70H	DLC register 8	DLCR8	R/W	----XXXX
001A71H	001B71H				
001A72H	001B72H	DLC register 9	DLCR9	R/W	----XXXXB
001A73H	001B73H				
001A74H	001B74H	DLC register 10	DLCR10	R/W	----XXXXB
001A75H	001B75H				
001A76H	001B76H	DLC register 11	DLCR11	R/W	----XXXXB
001A77H	001B77H				
001A78H	001B78H	DLC register 12	DLCR12	R/W	----XXXXB
001A79H	001B79H				
001A7AH	001B7AH	DLC register 13	DLCR13	R/W	----XXXXB
001A7BH	001B7BH				
001A7CH	001B7CH	DLC register 14	DLCR14	R/W	----XXXXB
001A7DH	001B7DH				
001A7EH	001B7EH	DLC register 15	DLCR15	R/W	----XXXXB
001A7FH	001B7FH				
001A80H to 001A87H	001B80H to 001B87H	Data register 0 (8 bytes)	DTR0	R/W	XXXXXXXXB to XXXXXXXXB

(Continued)

(Continued)

Address		Register	Abbreviation	Access	Initial Value
CAN0	CAN1				
001A88H to 001A8FH	001B88H to 001B8FH	Data register 1 (8 bytes)	DTR1	R/W	XXXXXXXXB to XXXXXXXXB
001A90H to 001A97H	001B90H to 001B97H	Data register 2 (8 bytes)	DTR2	R/W	XXXXXXXXB to XXXXXXXXB
001A98H to 001A9FH	001B98H to 001B9FH	Data register 3 (8 bytes)	DTR3	R/W	XXXXXXXXB to XXXXXXXXB
001AA0H to 001AA7H	001BA0H to 001BA7H	Data register 4 (8 bytes)	DTR4	R/W	XXXXXXXXB to XXXXXXXXB
001AA8H to 001AAFH	001BA8H to 001BAFH	Data register 5 (8 bytes)	DTR5	R/W	XXXXXXXXB to XXXXXXXXB
001AB0H to 001AB7H	001BB0H to 001BB7H	Data register 6 (8 bytes)	DTR6	R/W	XXXXXXXXB to XXXXXXXXB
001AB8H to 001ABFH	001BB8H to 001BBFH	Data register 7 (8 bytes)	DTR7	R/W	XXXXXXXXB to XXXXXXXXB
001AC0H to 001AC7H	001BC0H to 001BC7H	Data register 8 (8 bytes)	DTR8	R/W	XXXXXXXXB to XXXXXXXXB
001AC8H to 001ACFH	001BC8H to 001BCFH	Data register 9 (8 bytes)	DTR9	R/W	XXXXXXXXB to XXXXXXXXB
001AD0H to 001AD7H	001BD0H to 001BD7H	Data register 10 (8 bytes)	DTR10	R/W	XXXXXXXXB to XXXXXXXXB
001AD8H to 001ADFH	001BD8H to 001BDFH	Data register 11 (8 bytes)	DTR11	R/W	XXXXXXXXB to XXXXXXXXB
001AE0H to 001AE7H	001BE0H to 001BE7H	Data register 12 (8 bytes)	DTR12	R/W	XXXXXXXXB to XXXXXXXXB
001AE8H to 001AEFH	001BE8H to 001BEFH	Data register 13 (8 bytes)	DTR13	R/W	XXXXXXXXB to XXXXXXXXB
001AF0H to 001AF7H	001BF0H to 001BF7H	Data register 14 (8 bytes)	DTR14	R/W	XXXXXXXXB to XXXXXXXXB
001AF8H to 001AFFH	001BF8H to 001BFFH	Data register 15 (8 bytes)	DTR15	R/W	XXXXXXXXB to XXXXXXXXB

■ INTERRUPT MAP

Interrupt cause	I²OS clear	Interrupt vector		Interrupt control register	
		Number	Address	Number	Address
Reset	N/A	# 08	FFFFDC$_H$	—	—
INT9 instruction	N/A	# 09	FFFFD8$_H$	—	—
Exception	N/A	# 10	FFFFD4$_H$	—	—
Time Base Timer	N/A	# 11	FFFFD0$_H$	ICR00	0000B0$_H$
External Interrupt (INT0 to INT7)	*1	# 12	FFFFCC$_H$		
CAN 0 RX	N/A	# 13	FFFFC8$_H$	ICR01	0000B1$_H$
CAN 0 TX/NS	N/A	# 14	FFFFC4$_H$		
CAN 1 RX	N/A	# 15	FFFFC0$_H$	ICR02	0000B2$_H$
CAN 1 TX/NS	N/A	# 16	FFFFBC$_H$		
PPG 0/1	N/A	# 17	FFFFB8$_H$	ICR03	0000B3$_H$
PPG 2/3	N/A	# 18	FFFFB4$_H$		
PPG 4/5	N/A	# 19	FFFFB0$_H$	ICR04	0000B4$_H$
PPG 6/7	N/A	# 20	FFFFAC$_H$		
PPG 8/9	N/A	# 21	FFFFA8$_H$	ICR05	0000B5$_H$
PPG A/B	N/A	# 22	FFFFA4$_H$		
16-bit Reload Timer 0	*1	# 23	FFFFA0$_H$	ICR06	0000B6$_H$
16-bit Reload Timer 1	*1	# 24	FFFF9C$_H$		
Input Capture 0/1	*1	# 25	FFFF98$_H$	ICR07	0000B7$_H$
Output compare 0/1	*1	# 26	FFFF94$_H$		
Input Capture 2/3	*1	# 27	FFFF90$_H$	ICR08	0000B8$_H$
Output Compare 2/3	*1	# 28	FFFF8C$_H$		
Input Capture 4/5	*1	# 29	FFFF88$_H$	ICR09	0000B9$_H$
Output Compare 4/5	*1	# 30	FFFF84$_H$		
A/D Converter	*1	# 31	FFFF80$_H$	ICR10	0000BA$_H$
I/O Timer/Watch Timer	N/A	# 32	FFFF7C$_H$		
Serial I/O	*1	# 33	FFFF78$_H$	ICR11	0000BB$_H$
Sound Generator	N/A	# 34	FFFF74$_H$		
UART 0 RX	*2	# 35	FFFF70$_H$	ICR12	0000BC$_H$
UART 0 TX	*1	# 36	FFFF6C$_H$		
UART 1 RX	*2	# 37	FFFF68$_H$	ICR13	0000BD$_H$
UART 1 TX	*1	# 38	FFFF64$_H$		
UART 2 RX	*2	# 39	FFFF60$_H$	ICR14	0000BE$_H$
UART 2 TX	*1	# 40	FFFF5C$_H$		
Flash Memory	N/A	# 41	FFFF58$_H$	ICR15	0000BF$_H$
Delayed interrupt	N/A	# 42	FFFF54$_H$		

*1: The interrupt request flag is cleared by the I²OS interrupt clear signal.

*2: The interrupt request flag is cleared by the I²OS interrupt clear signal. A stop request is available.

N/A:The interrupt request flag is not cleared by the I²OS interrupt clear signal.

Note: For a peripheral module with two interrupt causes for a single interrupt number, both interrupt request flags are cleared by the I²OS interrupt clear signal.

Note: At the end of IIOS, the IIOS clear signal will be asserted for all the interrupt flags assigned to the same interrupt number. If one interrupt flag starts the IIOS and in the meantime another interrupt flag is set by hardware event, the later event is lost because the flag is cleared by the IIOS clear signal caused by the first event. So it is recommended not to use the IIOS for this interrupt number.

Note: If IIOS is enabled, IIOS is initiated when one of the two interrupt signals in the same interrupt control register (ICR) is asserted. This means that different interrupt sources share the same IIOS Descriptor which should be unique for each interrupt source.. For this reason, when one interrupt source uses the IIOS, the other interrupt should be disabled.

■ ELECTRICAL CHARACTERISTICS

1. Absolute Maximum Ratings

(V_{SS} = AV_{SS} = 0 V)

Parameter	Symbol	Value		Units	Remarks
		Min.	Max.		
Power supply voltage	V_{CC}	V_{SS} – 0.3	V_{SS} + 6.0	V	
	AV_{CC}	V_{SS} – 0.3	V_{SS} + 6.0	V	V_{CC} = AV_{CC} *1
	AVR±	V_{SS} – 0.3	V_{SS} + 6.0	V	AV_{CC} ≥ AVR±, AVR+ ≥ AVR –
	DV_{CC}	V_{SS} – 0.3	V_{SS} + 6.0	V	V_{CC} ≥ DV_{CC}
Input voltage	V_I	V_{SS} – 0.3	V_{SS} + 6.0	V	*2
Output voltage	V_O	V_{SS} – 0.3	V_{SS} + 6.0	V	*2
Clamp Current	I_{CLAMP}	–2.0	2.0	mA	
"L" level max. output current	I_{OL1}	—	15	mA	Normal outputs
"L" level avg. output current	I_{OLAV1}	—	4	mA	Normal outputs, average value
"L" level max. output current	I_{OL2}	—	40	mA	High current outputs
"L" level avg. output current	I_{OLAV2}	—	30	mA	High current outputs, average value
"L" level max. overall output current	$\sum I_{OL1}$	—	100	mA	Sum of all normal outputs
"L" level max. overall output current	$\sum I_{OL2}$		330	mA	Sum of all high current outputs
"L" level avg. overall output current	$\sum I_{OLAV1}$	—	50	mA	Sum of all normal outputs, average value
"L" level avg. overall output current	$\sum I_{OLAV2}$		250	mA	Sum of all high current outputs, average value
"H" level max. output current	I_{OH1}	—	–15	mA	Normal outputs
"H" level avg. output current	I_{OHAV1}	—	–4	mA	Normal outputs, average value
"H" level max. output current	I_{OH2}	—	–40	mA	High current outputs
"H" level avg. output current	I_{OHAV2}	—	–30	mA	High current outputs, average value
"H" level max. overall output current	$\sum I_{OH1}$	—	-100	mA	Sum of all normal outputs
"H" level max. overall output current	$\sum I_{OH2}$	—	-330	mA	Sum of all high current outputs
"H" level avg. overall output current	$\sum I_{OHAV1}$	—	-50	mA	Sum of all normal outputs, average value
"H" level avg. overall output current	$\sum I_{OHAV2}$	—	-250	mA	Sum of all high current outputs, average value
Power consumption	P_D	—	500	mW	MB90F594A, MB90F591
		—	400	mW	MB90594, MB90591
Operating temperature	T_A	–40	+85	°C	
Storage temperature	T_{STG}	–55	+150	°C	

*1: Set AV_{CC} and V_{CC} to the same voltage. Make sure that AV_{CC} does not exceed V_{CC} and that the voltage at the analog inputs does not exceed AV_{CC} when the power is switched on.

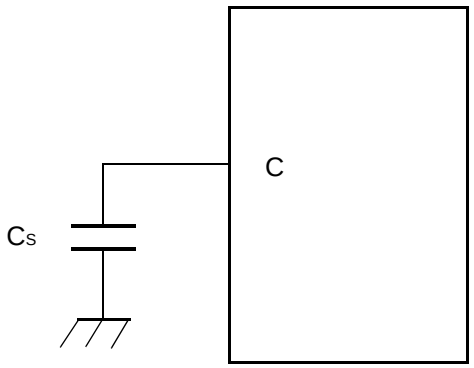
*2: V_I and V_O should not exceed V_{CC} + 0.3 V. V_I should not exceed the specified ratings. However if the maximun current to/from a input is limited by some means with external components, the I_{CLAMP} rating supercedes the V_I rating.

2. Recommended Conditions

(V_{SS} = AV_{SS} = 0 V)

Parameter	Symbol	Rated Value			Units	Remarks
		Min.	Typ.	Max.		
Power supply voltage	V_{CC} AV_{CC}	4.5	5.0	5.5	V	Under normal operation
		3V		5.5	V	Maintains RAM data in stop mode
Input H voltage	V_{IHS}	0.8 V_{CC}		V_{CC} +0.3	V	CMOS hysteresis input pin
	V_{IHM}	V_{CC} – 0.3		V_{CC} +0.3	V	MD input pin
Input L voltage	V_{ILS}	V_{SS} – 0.3		0.6V_{CC}	V	CMOS hysteresis input pin
	V_{ILM}	V_{SS} – 0.3		V_{SS} + 0.3	V	MD input pin
Smooth capacitor	C_S	0.022	0.1	1.0	µF	Use a ceramic capacitor or capacitor of better AC characteristics. Capacitor at the VCC should be greater than this capacitor.
Operating temperature	T_A	–40		+85	°C	

- **C Pin Connection Diagram**



3. DC Characteristics

(V_{CC} = 5.0 V±10%, V_{SS} = AV_{SS} = 0V, T_A = –40 °C to +85 °C)

Parameter	Symbol	Pin	Test Condition	Rated Value			Units	Remarks
				Min.	Typ.	Max.		
Output H voltage	V_{OH1}	Normal outputs	V_{CC} = 4.5 V, I_{OH1} = –4.0 mA	V_{CC} – 0.5	—	—	V	
Output H voltage	V_{OH2}	High current outputs	V_{CC} = 4.5 V, I_{OH2} = –30.0 mA	V_{CC} – 0.5	—	—	V	
Output L voltage	V_{OL1}	Normal outputs	V_{CC} = 4.5 V, I_{OL1} = 4.0 mA	—	—	0.4	V	
Output L voltage	V_{OL2}	High current outputs	V_{CC} = 4.5 V, I_{OL2} = 30.0 mA	—	—	0.5	V	
Input leak current	I_{IL}		V_{CC} = 5.5 V, V_{SS} < V_I < V_{CC}	–5	—	5	µA	
Analog input leak current	I_{IAL}	AN0 to AN7	V_{CC} = 5.5 V, AV_{SS} < V_I < AV_{CC}	–1	—	1	µA	
Power supply current *	I_{CC}	V_{CC}	V_{CC} = 5.0 V±10%, Internal frequency: 16 MHz, At normal operation.	—	37	60	mA	MB90594
				—	50	80	mA	MB90F594A
				—	TBD	TBD	mA	MB90F591
				—	TBD	TBD	mA	MB90591
	I_{CCS}		V_{CC} = 5.0 V±10%, Internal frequency: 16 MHz, At Sleep mode.	—	13	20	mA	MB90594
				—	15	23	mA	MB90F594A
				—	TBD	TBD	mA	MB90F591
				—	TBD	TBD	mA	MB90591
	I_{CTS}		V_{CC} = 5.0 V±1%, Internal frequency: 2 MHz, At Timer mode	—	0.3	0.6	mA	MB90594
				—	0.35	0.6	mA	MB90F594A
				—	TBD	TBD	mA	MB90F591
				—	TBD	TBD	mA	MB90591
	I_{CCH}		V_{CC} = 5.0 V±10%, At Stop mode, T_A = 25°C	—	5	20	µA	MB90594
				—	5	20	µA	MB90F594A
				—	TBD	TBD	µA	MB90F591
				—	TBD	TBD	µA	MB90591
Input capacity	C_{IN}	Other than C, AV_{CC}, AV_{SS}, AVR+, AVR-, V_{CC}, V_{SS}, DV_{CC}, DV_{SS}	—	—	10	80	pF	

*: Current values are tentative. They are subject to change without notice according to improvements in the characteristics. The power supply current testing conditions are when using the external clock.

4. AC Characteristics

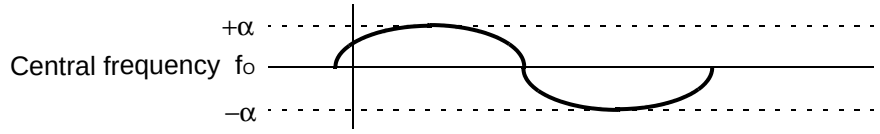
(1) Clock Timing

(V_{CC} = 5.0 V±10%, V_{SS} = AV_{SS} = 0V, T_A = –40 °C to +85 °C)

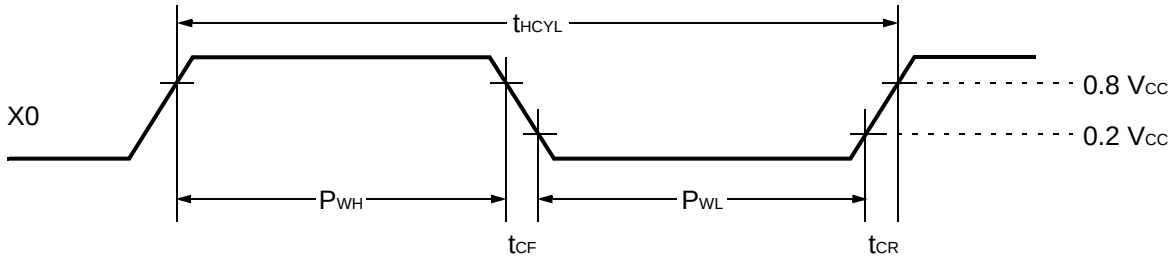
Parameter	Symbol	Pin	Value			Units	Remarks
			Min.	Typ.	Max.		
Oscillation frequency	f_C	X0, X1	3	—	16	MHz	
Oscillation cycle time	t_{CYL}	X0, X1	62.5	—	333	ns	
Frequency deviation with PLL *	Δf	—	—	—	5	%	
Input clock pulse width	P_{WH}, P_{WL}	X0	10	—	—	ns	Duty ratio is about 30 to 70%.
Input clock rise and fall time	t_{CR}, t_{CF}	X0	—	—	5	ns	When using external clock
Machine clock frequency	f_{CP}	—	1.5	—	16	MHz	
Machine clock cycle time	t_{CP}	—	62.5	—	666	ns	

*: Frequency deviation indicates the maximum frequency difference from the target frequency when using a multiplied clock.

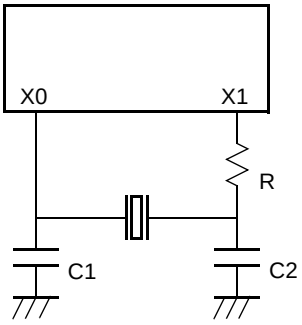
$$\Delta f = \frac{|\alpha|}{fo} \times 100\%$$



- **Clock Timing**

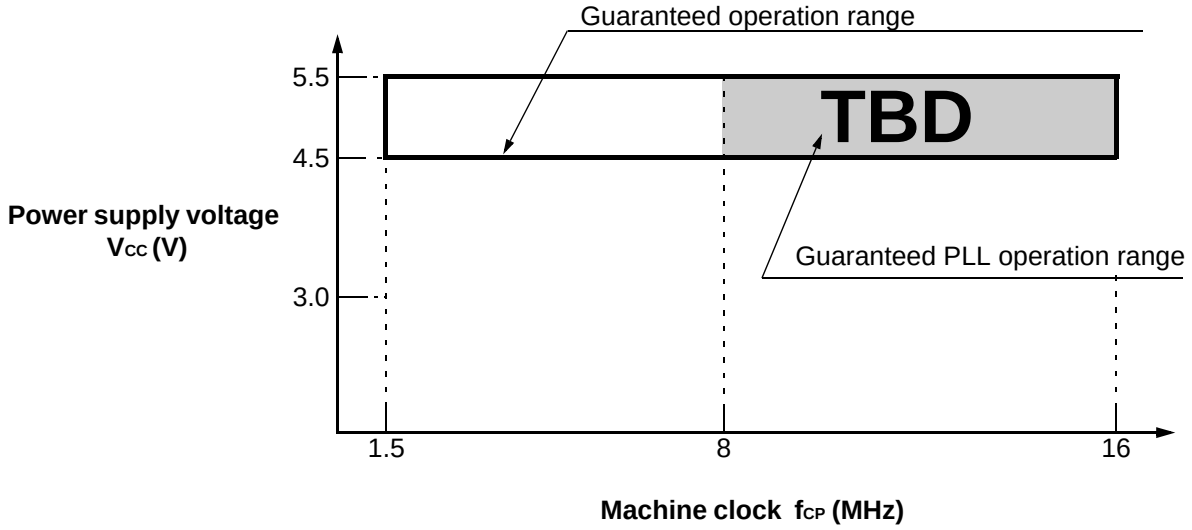


Example of Oscillation circuit

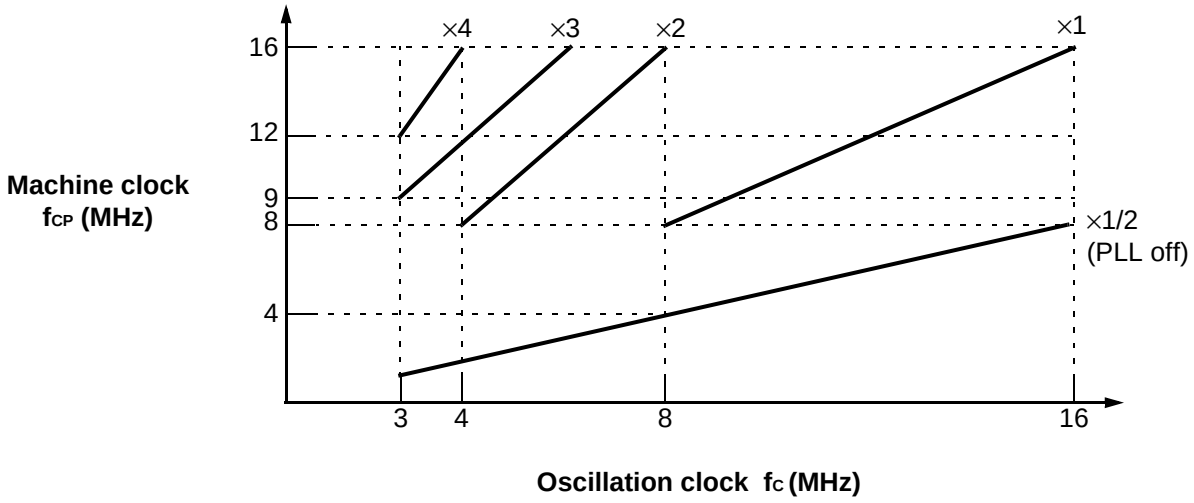


Make	Oscillator	Frequency (MHz)	C1 (pF)	C2 (pF)	R (Ω)
TBD	TBD	4MHz	TBD	TBD	TBD

• **Guaranteed operation range**



• **Ocsillation clock frequency and Machine clock frequency**

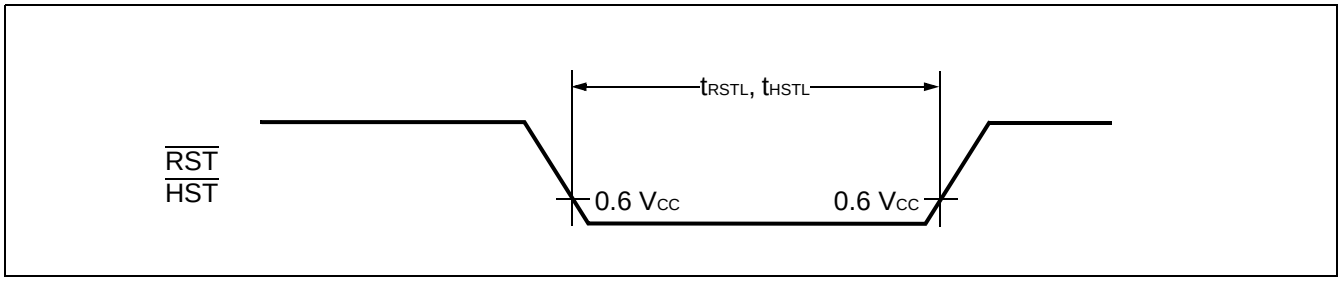


(2) Reset and Hardware Standby Input

(V_{CC} = 5.0 V±10%, V_{SS} = AV_{SS} = 0V, T_A = –40 °C to +85 °C)

Parameter	Symbol	Pin	Rated Value		Units	Remarks
			Min.	Max.		
Reset input time	t_{RSTL}	$\overline{RST}$	16 t_{CP}	—	ns	
Hardware standby input time	t_{HSTL}	$\overline{HST}$	16 t_{CP}	—	ns	

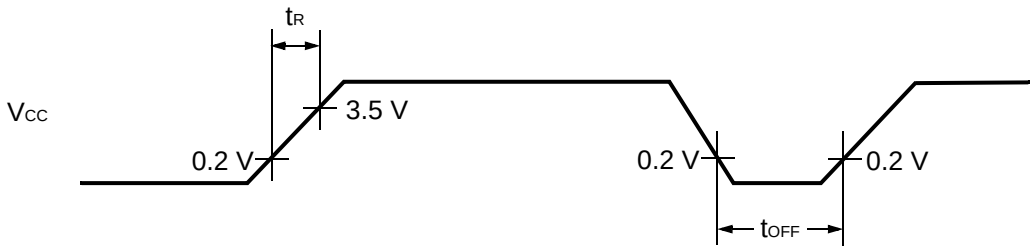
"t_{cp}" represents one cycle time of the machine clock.
Any reset can not fully initialize the Flash Memory if it is performing the automatic algorithm.



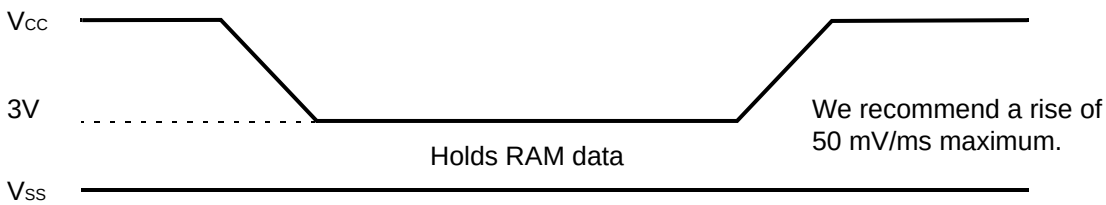
(3) Power On Reset

(V_{CC} = 5.0 V±10%, V_{SS} = AV_{SS} = 0V, T_A = –40 °C to +85)

Parameter	Symbol	Pin	Test Condition	Rated Value		Units	Remarks
				Min.	Max.		
Power on rise time	t_R	V_{CC}	—	0.05	30	ms	
Power off time	t_{OFF}	V_{CC}		50	—	ms	Due to repetitive operation



If you change the power supply voltage too rapidly, a power on reset may occur. We recommend that you startup smoothly by restraining voltages when changing the power supply voltage during operation, as shown in the figure below. Perform while not using the PLL clock. However, if voltage drops are within 1 mV/sec, you can operate while using the PLL clock.



(4) UART0/1/2, Serial I/O Timing

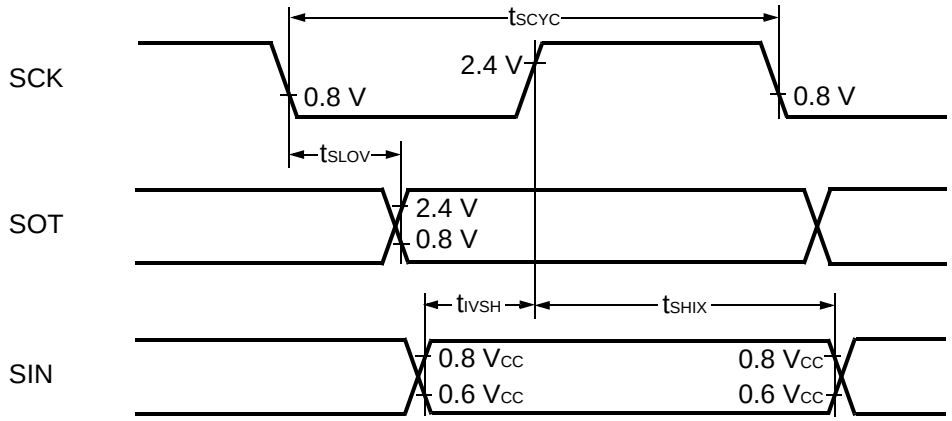
(V_{CC} = 5.0 V±10%, V_{SS} = AV_{SS} = 0V, T_A = –40 °C to +85 °C)

Parameter	Symbol	Pin Symbol	Condition	Value		Units	Remarks
				Min.	Max.		
Serial clock cycle time	t_{SCYC}	SCK0 to SCK3	Internal clock operation output pins are C_L = 80 pF + 1 TTL.	8 t_{CP}	—	ns	
SCK ↓ ⇒ SOT delay time	t_{SLOV}	SCK0 to SCK3, SOT0 to SOT3		–80	80	ns	
Valid SIN ⇒ SCK ↑	t_{IVSH}	SCK0 to SCK3, SIN0 to SIN3		100	—	ns	
SCK ↑ ⇒ Valid SIN hold time	t_{SHIX}	SCK0 to SCK3, SIN0 to SIN3		60	—	ns	
Serial clock "H" pulse width	t_{SHSL}	SCK0 to SCK3	External clock operation output pins are C_L = 80 pF + 1 TTL.	4 t_{CP}	—	ns	
Serial clock "L" pulse width	t_{SLSH}	SCK0 to SCK3		4 t_{CP}	—	ns	
SCK ↓ ⇒ SOT delay time	t_{SLOV}	SCK0 to SCK3, SOT0 to SOT3		—	150	ns	
Valid SIN ⇒ SCK ↑	t_{IVSH}	SCK0 to SCK3, SIN0 to SIN3		60	—	ns	
SCK ↑ ⇒ Valid SIN hold time	t_{SHIX}	SCK0 to SCK3, SIN0 to SIN3		60	—	ns	

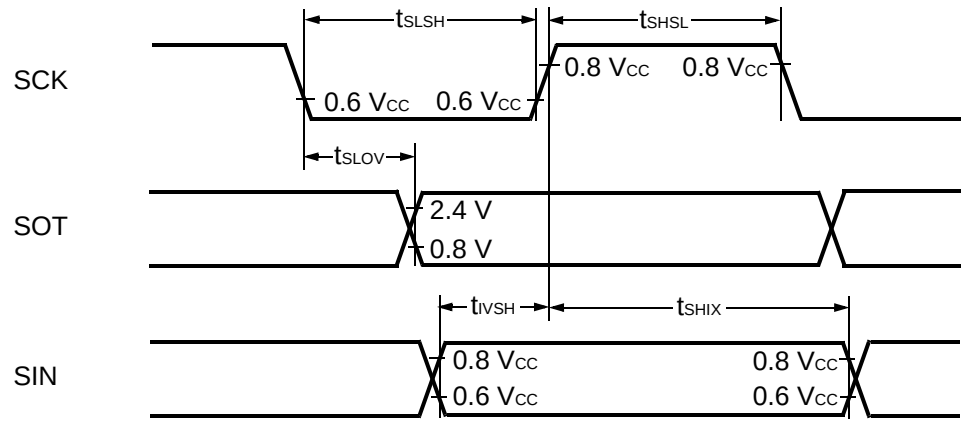
Note:

1. AC characteristic in CLK synchronized mode.
2. C_L is load capacity value of pins when testing.
3. t_{CP} is the machine cycle (Unit: ns).

- **Internal Shift Clock Mode**



• **External Shift Clock Mode**

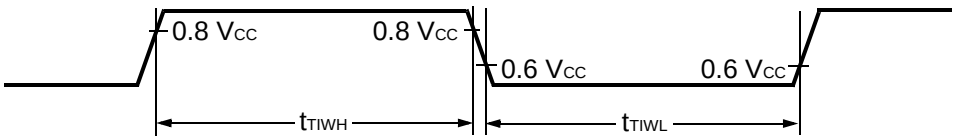


(5)Timer Related Resource Input Timing

(V_{CC} = 5.0 V±10%, V_{SS} = AV_{SS} = 0V, T_A = –40 °C to +85 °C)

Parameter	Symbol	Pin	Condition	Value		Units	Remarks
				Min.	Max.		
Input pulse width	t_{TIWH}	TIN0	—	4 t_{CP}	—	ns	
	t_{TIWL}	IN0 to IN5					

• **Timer Input Timing**

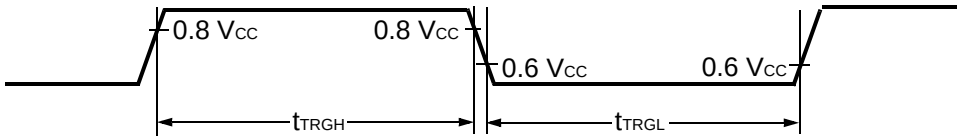


(6)Trigger Input Timing

(V_{CC} = 5.0 V±10%, V_{SS} = AV_{SS} = 0V, T_A = –40 °C to +85 °C)

Parameter	Symbol	Pin	Condition	Value		Units	Remarks
				Min.	Max.		
Input pulse width	t_{TRGH} t_{TRGL}	INT0 to INT7, ADTG	—	5 t_{CP}	—	ns	

- **Trigger Input Timing**



5. A/D Converter

(V_{CC} = AV_{CC} = 5.0 V±10%, V_{SS} = AV_{SS} = 0 V,3.0 V ≤ $AVR_+ - AVR_-$, T_A = –40 °C to +85 °C)

Parameter	Symbol	Pin	Value			Units	Remarks
			Min.	Typ.	Max.		
Resolution	—	—	—		10	bit	
Conversion error	—	—	—	—	±5.0	LSB	
Nonlinearity error	—	—	—	—	±2.5	LSB	
Differential nonlinearity error	—	—	—	—	±1.9	LSB	
Zero reading voltage	V_{OT}	AN0 to AN7	AVR- – 3.5	AVR- +0.5	AVR- + 4.5	mV	
Full scale reading voltage	V_{FST}	AN0 to AN7	AVR+ – 6.5	AVR+ –1.5	AVR+ + 1.5	mV	
Conversion time	—	—	—	352t_{CP}	—	ns	
Sampling time	—	—	—	64t_{CP}	—	ns	
Analog port input current	I_{AIN}	AN0 to AN7	-1	—	+1	µA	
Analog input voltage range	V_{AIN}	AN0 to AN7	AVR-	—	AVR+	V	
Reference voltage range	—	AVR+	AVR- + 2.7	—	AV_{CC}	V	
	—	AVR-	0	—	AVR+ – 2.7	V	
Power supply current	I_A	AV_{CC}	—	5	—	mA	
	I_{AH}	AV_{CC}	—	—	5	µA	*1
Reference voltage current	I_R	AVR+	200	400	600	µA	
	I_{RH}	AVR+	—	—	5	µA	*1
Offset between input channels	—	AN0 to AN7	—	—	4	LSB	

*1: When not operating A/D converter, this is the current (V_{CC} = AV_{CC} = AVR_+ = 5.0 V) when the CPU is stopped.

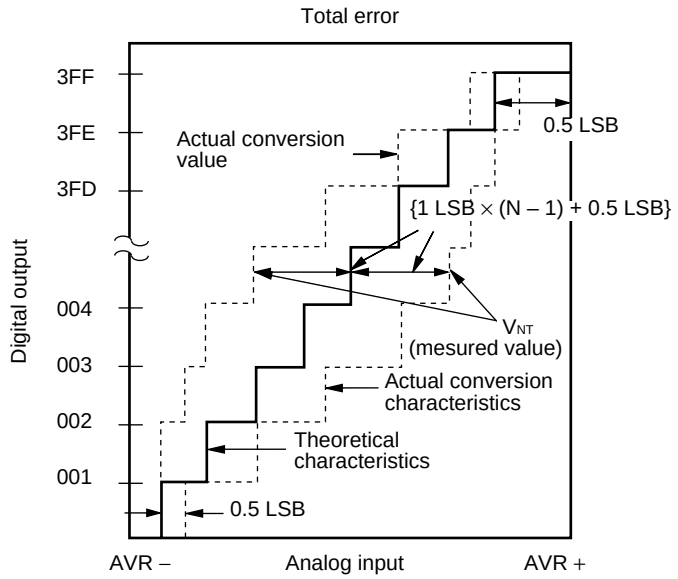
6. A/D Converter Glossary

Resolution: Analog changes that are identifiable with the A/D converter

Linearity error: The deviation of the straight line connecting the zero transition point (“00 0000 0000” ↔ “00 0000 0001”) with the full-scale transition point (“11 1111 1110” ↔ “11 1111 1111”) from actual conversion characteristics

Differential linearity error: The deviation of input voltage needed to change the output code by 1 LSB from the theoretical value

Total error: The total error is defined as a difference between the actual value and the theoretical value, which includes zero-transition error/full-scale transition error and linearity error.



1 LSB = (Theoretical value) $\frac{\text{AVR+} - \text{AVR–}}{1024}$ [V]

Total error for digital output N = $\frac{V_{NT} - \{1\text{ LSB} \times (N - 1) + 0.5\text{ LSB}\}}{1\text{ LSB}}$ [LSB]

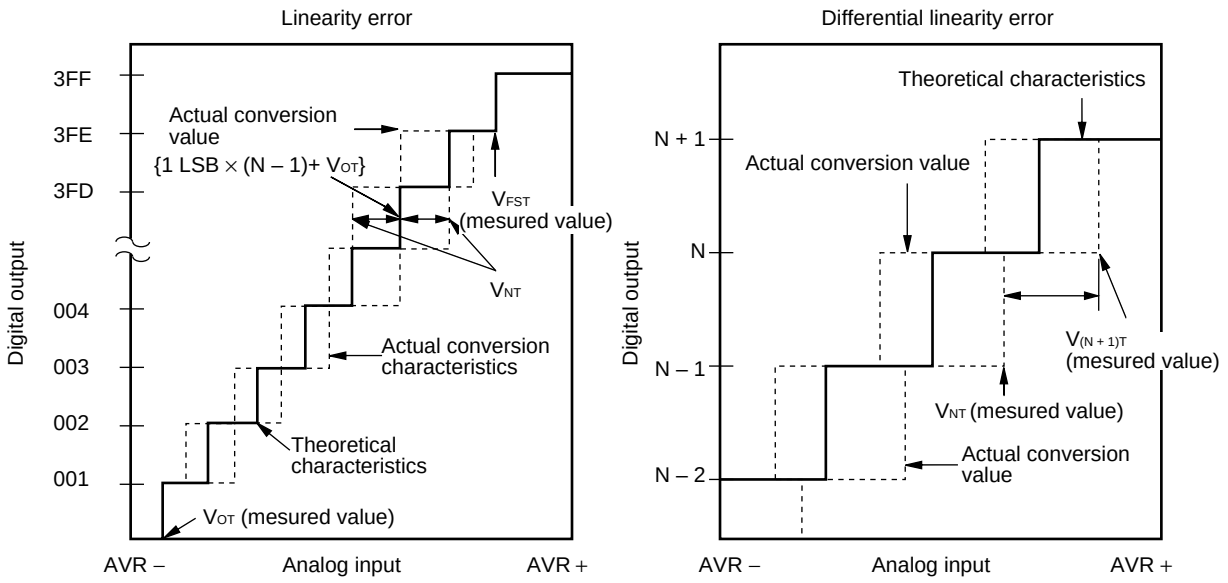
V_{OT} (Theoretical value) = AVR – + 0.5 LSB[V]

V_{NT}: Voltage at a transition of digital output from (N – 1) to N

V_{FST} (Theoretical value) = AVR + – 1.5 LSB[V]

(Continued)

(Continued)



$$\text{Linearity error of digital output N} = \frac{V_{NT} - \{1\ \text{LSB} \times (N - 1) + V_{OT}\}}{1\ \text{LSB}}\ [\text{LSB}]$$

$$\text{Differential linearity error of digital N} = \frac{V_{(N+1)T} - V_{NT}}{1\ \text{LSB}} - 1\ \text{LSB}\ [\text{LSB}]$$

$$1\ \text{LSB} = \frac{V_{FST} - V_{OT}}{1022}\ [\text{V}]$$

V_{OT}: Voltage at transition of digital output from "000_H" to "001_H"
V_{FST}: Voltage at transition of digital output from "$3FE_H$" to "$3FF_H$"

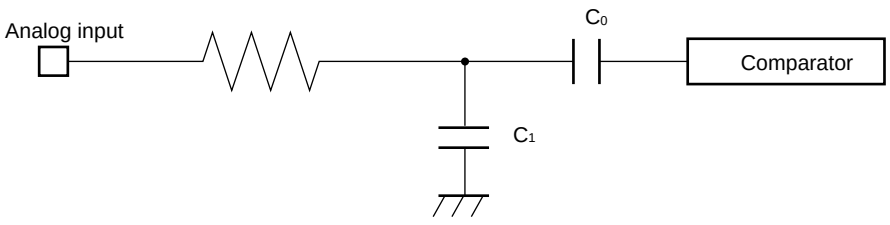
7. Notes on Using A/D Converter

Select the output impedance value for the external circuit of analog input according to the following conditions. Output impedance values of the external circuit of 15 kΩ or lower are recommended.

When capacitors are connected to external pins, the capacitance of several thousand times the internal capacitor value is recommended to minimized the effect of voltage distribution between the external capacitor and internal capacitor.

When the output impedance of the external circuit is too high, the sampling period for analog voltages may not be sufficient (sampling period = 4.00 µs @machine clock of 16 MHz).

- **Equipment of analog input circuit model**



Note: Listed values must be considered as standards.

- **Error**

The smaller the | AVR + – AVR – |, the greater the error would become relatively.

■ INSTRUCTIONS (340 INSTRUCTIONS)

Table 1 Explanation of Items in Tables of Instructions

Item	Meaning
Mnemonic	Upper-case letters and symbols: Represented as they appear in assembler. Lower-case letters: Replaced when described in assembler. Numbers after lower-case letters: Indicate the bit width within the instruction code.
#	Indicates the number of bytes.
~	Indicates the number of cycles. m : When branching n : When not branching See Table 4 for details about meanings of other letters in items.
RG	Indicates the number of accesses to the register during execution of the instruction. It is used calculate a correction value for intermittent operation of CPU.
B	Indicates the correction value for calculating the number of actual cycles during execution of the instruction. (Table 5) The number of actual cycles during execution of the instruction is the correction value summed with the value in the “~” column.
Operation	Indicates the operation of instruction.
LH	Indicates special operations involving the upper 8 bits of the lower 16 bits of the accumulator. Z : Transfers “0”. X : Extends with a sign before transferring. – : Transfers nothing.
AH	Indicates special operations involving the upper 16 bits in the accumulator. * : Transfers from AL to AH. – : No transfer. Z : Transfers 00_H to AH. X : Transfers 00_H or FF_H to AH by signing and extending AL.
I S T N Z V C	Indicates the status of each of the following flags: I (interrupt enable), S (stack), T (sticky bit), N (negative), Z (zero), V (overflow), and C (carry). * : Changes due to execution of instruction. – : No change. S : Set by execution of instruction. R : Reset by execution of instruction.
RMW	Indicates whether the instruction is a read-modify-write instruction. (a single instruction that reads data from memory, etc., processes the data, and then writes the result to memory.) * : Instruction is a read-modify-write instruction. – : Instruction is not a read-modify-write instruction. Note: A read-modify-write instruction cannot be used on addresses that have different meanings depending on whether they are read or written.

- **Number of execution cycles**

  The number of cycles required for instruction execution is acquired by adding the number of cycles for each instruction, a corrective value depending on the condition, and the number of cycles required for program fetch. Whenever the instruction being executed exceeds the two-byte (word) boundary, a program on an internal ROM connected to a 16-bit bus is fetched. If data access is interfered with, therefore, the number of execution cycles is increased.

  For each byte of the instruction being executed, a program on a memory connected to an 8-bit external data bus is fetched. If data access in interfered with, therefore, the number of execution cycles is increased. When a general-purpose register, an internal ROM, an internal RAM, an internal I/O device, or an external bus is accessed during intermittent CPU operation, the CPU clock is suspended by the number of cycles specified by the CG1/0 bit of the low-power consumption mode control register. When determining the number of cycles required for instruction execution during intermittent CPU operation, therefore, add the value of the number of times access is done × the number of cycles suspended as the corrective value to the number of ordinary execution cycles.

Table 2 Explanation of Symbols in Tables of Instructions

Symbol	Meaning
A	32-bit accumulator The bit length varies according to the instruction. Byte : Lower 8 bits of AL Word : 16 bits of AL Long : 32 bits of AL and AH
AH AL	Upper 16 bits of A Lower 16 bits of A
SP	Stack pointer (USP or SSP)
PC	Program counter
PCB	Program bank register
DTB	Data bank register
ADB	Additional data bank register
SSB	System stack bank register
USB	User stack bank register
SPB	Current stack bank register (SSB or USB)
DPR	Direct page register
brg1	DTB, ADB, SSB, USB, DPR, PCB, SPB
brg2	DTB, ADB, SSB, USB, DPR, SPB
Ri	R0, R1, R2, R3, R4, R5, R6, R7
RWi	RW0, RW1, RW2, RW3, RW4, RW5, RW6, RW7
RWj	RW0, RW1, RW2, RW3
RLi	RL0, RL1, RL2, RL3
dir	Compact direct addressing
addr16 addr24 ad24 0 to 15 ad24 16 to 23	Direct addressing Physical direct addressing Bit 0 to bit 15 of addr24 Bit 16 to bit 23 of addr24
io	I/O area (000000_H to $0000FF_H$)
imm4 imm8 imm16 imm32 ext (imm8)	4-bit immediate data 8-bit immediate data 16-bit immediate data 32-bit immediate data 16-bit data signed and extended from 8-bit immediate data
disp8 disp16	8-bit displacement 16-bit displacement
bp	Bit offset
vct4 vct8	Vector number (0 to 15) Vector number (0 to 255)
( )b	Bit address
rel	PC relative addressing
ear eam	Effective addressing (codes 00 to 07) Effective addressing (codes 08 to 1F)
rlst	Register list

Table 3 Effective Address Fields

Code	Notation			Address format	Number of bytes in address extension *
00 01 02 03 04 05 06 07	R0 R1 R2 R3 R4 R5 R6 R7	RW0 RW1 RW2 RW3 RW4 RW5 RW6 RW7	RL0 (RL0) RL1 (RL1) RL2 (RL2) RL3 (RL3)	Register direct “ea” corresponds to byte, word, and long-word types, starting from the left	—
08 09 0A 0B	@RW0 @RW1 @RW2 @RW3			Register indirect	0
0C 0D 0E 0F	@RW0 + @RW1 + @RW2 + @RW3 +			Register indirect with post-increment	0
10 11 12 13 14 15 16 17	@RW0 + disp8 @RW1 + disp8 @RW2 + disp8 @RW3 + disp8 @RW4 + disp8 @RW5 + disp8 @RW6 + disp8 @RW7 + disp8			Register indirect with 8-bit displacement	1
18 19 1A 1B	@RW0 + disp16 @RW1 + disp16 @RW2 + disp16 @RW3 + disp16			Register indirect with 16-bit displacement	2
1C 1D 1E 1F	@RW0 + RW7 @RW1 + RW7 @PC + disp16 addr16			Register indirect with index Register indirect with index PC indirect with 16-bit displacement Direct address	0 0 2 2

Note : The number of bytes in the address extension is indicated by the “+” symbol in the “#” (number of bytes) column in the tables of instructions.

Table 4 Number of Execution Cycles for Each Type of Addressing

Code	Operand	(a) Number of execution cycles for each type of addressing	Number of register accesses for each type of addressing
00 to 07	Ri RWi RLi	Listed in tables of instructions	Listed in tables of instructions
08 to 0B	@RWj	2	1
0C to 0F	@RWj +	4	2
10 to 17	@RWi + disp8	2	1
18 to 1B	@RWj + disp16	2	1
1C 1D 1E 1F	@RW0 + RW7 @RW1 + RW7 @PC + disp16 addr16	4 4 2 1	2 2 0 0

Note : “(a)” is used in the “~” (number of states) column and column B (correction value) in the tables of instructions.

Table 5 Compensation Values for Number of Cycles Used to Calculate Number of Actual Cycles

Operand	(b) byte		(c) word		(d) long	
	Cycles	Access	Cycles	Access	Cycles	Access
Internal register	+0	1	+0	1	+0	2
Internal memory even address Internal memory odd address	+0 +0	1 1	+0 +2	1 2	+0 +4	2 4
Even address on external data bus (16 bits) Odd address on external data bus (16 bits)	+1 +1	1 1	+1 +4	1 2	+2 +8	2 4
External data bus (8 bits)	+1	1	+4	2	+8	4

Notes: • “(b)”, “(c)”, and “(d)” are used in the “~” (number of states) column and column B (correction value) in the tables of instructions.
• When the external data bus is used, it is necessary to add in the number of wait cycles used for ready input and automatic ready.

Table 6 Correction Values for Number of Cycles Used to Calculate Number of Program Fetch Cycles

Instruction	Byte boundary	Word boundary
Internal memory	—	+2
External data bus (16 bits)	—	+3
External data bus (8 bits)	+3	—

Notes: • When the external data bus is used, it is necessary to add in the number of wait cycles used for ready input and automatic ready.
• Because instruction execution is not slowed down by all program fetches in actuality, these correction values should be used for “worst case” calculations.

Table 7 Transfer Instructions (Byte) [41 Instructions]

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOV	A, dir	2	3	0	(b)	byte (A) ← (dir)	Z	*	–	–	–	*	*	–	–	–
MOV	A, addr16	3	4	0	(b)	byte (A) ← (addr16)	Z	*	–	–	–	*	*	–	–	–
MOV	A, Ri	1	2	1	0	byte (A) ← (Ri)	Z	*	–	–	–	*	*	–	–	–
MOV	A, ear	2	2	1	0	byte (A) ← (ear)	Z	*	–	–	–	*	*	–	–	–
MOV	A, eam	2+	3+ (a)	0	(b)	byte (A) ← (eam)	Z	*	–	–	–	*	*	–	–	–
MOV	A, io	2	3	0	(b)	byte (A) ← (io)	Z	*	–	–	–	*	*	–	–	–
MOV	A, #imm8	2	2	0	0	byte (A) ← imm8	Z	*	–	–	–	*	*	–	–	–
MOV	A, @A	2	3	0	(b)	byte (A) ← ((A))	Z	–	–	–	–	*	*	–	–	–
MOV	A, @RLi+disp8	3	10	2	(b)	byte (A) ← ((RLi)+disp8)	Z	*	–	–	–	*	*	–	–	–
MOVN	A, #imm4	1	1	0	0	byte (A) ← imm4	Z	*	–	–	–	R	*	–	–	–
MOVX	A, dir	2	3	0	(b)	byte (A) ← (dir)	X	*	–	–	–	*	*	–	–	–
MOVX	A, addr16	3	4	0	(b)	byte (A) ← (addr16)	X	*	–	–	–	*	*	–	–	–
MOVX	A, Ri	2	2	1	0	byte (A) ← (Ri)	X	*	–	–	–	*	*	–	–	–
MOVX	A, ear	2	2	1	0	byte (A) ← (ear)	X	*	–	–	–	*	*	–	–	–
MOVX	A, eam	2+	3+ (a)	0	(b)	byte (A) ← (eam)	X	*	–	–	–	*	*	–	–	–
MOVX	A, io	2	3	0	(b)	byte (A) ← (io)	X	*	–	–	–	*	*	–	–	–
MOVX	A, #imm8	2	2	0	0	byte (A) ← imm8	X	*	–	–	–	*	*	–	–	–
MOVX	A, @A	2	3	0	(b)	byte (A) ← ((A))	X	–	–	–	–	*	*	–	–	–
MOVX	A,@RWi+disp8	2	5	1	(b)	byte (A) ← ((RWi)+disp8)	X	*	–	–	–	*	*	–	–	–
MOVX	A, @RLi+disp8	3	10	2	(b)	byte (A) ← ((RLi)+disp8)	X	*	–	–	–	*	*	–	–	–
MOV	dir, A	2	3	0	(b)	byte (dir) ← (A)	–	–	–	–	–	*	*	–	–	–
MOV	addr16, A	3	4	0	(b)	byte (addr16) ← (A)	–	–	–	–	–	*	*	–	–	–
MOV	Ri, A	1	2	1	0	byte (Ri) ← (A)	–	–	–	–	–	*	*	–	–	–
MOV	ear, A	2	2	1	0	byte (ear) ← (A)	–	–	–	–	–	*	*	–	–	–
MOV	eam, A	2+	3+ (a)	0	(b)	byte (eam) ← (A)	–	–	–	–	–	*	*	–	–	–
MOV	io, A	2	3	0	(b)	byte (io) ← (A)	–	–	–	–	–	*	*	–	–	–
MOV	@RLi+disp8, A	3	10	2	(b)	byte ((RLi) +disp8) ← (A)	–	–	–	–	–	*	*	–	–	–
MOV	Ri, ear	2	3	2	0	byte (Ri) ← (ear)	–	–	–	–	–	*	*	–	–	–
MOV	Ri, eam	2+	4+ (a)	1	(b)	byte (Ri) ← (eam)	–	–	–	–	–	*	*	–	–	–
MOV	ear, Ri	2	4	2	0	byte (ear) ← (Ri)	–	–	–	–	–	*	*	–	–	–
MOV	eam, Ri	2+	5+ (a)	1	(b)	byte (eam) ← (Ri)	–	–	–	–	–	*	*	–	–	–
MOV	Ri, #imm8	2	2	1	0	byte (Ri) ← imm8	–	–	–	–	–	*	*	–	–	–
MOV	io, #imm8	3	5	0	(b)	byte (io) ← imm8	–	–	–	–	–	–	–	–	–	–
MOV	dir, #imm8	3	5	0	(b)	byte (dir) ← imm8	–	–	–	–	–	–	–	–	–	–
MOV	ear, #imm8	3	2	1	0	byte (ear) ← imm8	–	–	–	–	–	*	*	–	–	–
MOV	eam, #imm8	3+	4+ (a)	0	(b)	byte (eam) ← imm8	–	–	–	–	–	–	–	–	–	–
MOV	@AL, AH															
/MOV	@A, T	2	3	0	(b)	byte ((A)) ← (AH)	–	–	–	–	–	*	*	–	–	–
XCH	A, ear	2	4	2	0	byte (A) ↔ (ear)	Z	–	–	–	–	–	–	–	–	–
XCH	A, eam	2+	5+ (a)	0	2× (b)	byte (A) ↔ (eam)	Z	–	–	–	–	–	–	–	–	–
XCH	Ri, ear	2	7	4	0	byte (Ri) ↔ (ear)	–	–	–	–	–	–	–	–	–	–
XCH	Ri, eam	2+	9+ (a)	2	2× (b)	byte (Ri) ↔ (eam)	–	–	–	–	–	–	–	–	–	–

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 8 Transfer Instructions (Word/Long Word) [38 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVW A, dir	2	3	0	(c)	word (A) ← (dir)	–	*	–	–	–	*	*	–	–	–
MOVW A, addr16	3	4	0	(c)	word (A) ← (addr16)	–	*	–	–	–	*	*	–	–	–
MOVW A, SP	1	1	0	0	word (A) ← (SP)	–	*	–	–	–	*	*	–	–	–
MOVW A, RWi	1	2	1	0	word (A) ← (RWi)	–	*	–	–	–	*	*	–	–	–
MOVW A, ear	2	2	1	0	word (A) ← (ear)	–	*	–	–	–	*	*	–	–	–
MOVW A, eam	2+	3+ (a)	0	(c)	word (A) ← (eam)	–	*	–	–	–	*	*	–	–	–
MOVW A, io	2	3	0	(c)	word (A) ← (io)	–	*	–	–	–	*	*	–	–	–
MOVW A, @A	2	3	0	(c)	word (A) ← ((A))	–	–	–	–	–	*	*	–	–	–
MOVW A, #imm16	3	2	0	0	word (A) ← imm16	–	*	–	–	–	*	*	–	–	–
MOVW A, @RWi+disp8	2	5	1	(c)	word (A) ← ((RWi) +disp8)	–	*	–	–	–	*	*	–	–	–
MOVW A, @RLi+disp8	3	10	2	(c)	word (A) ← ((RLi) +disp8)	–	*	–	–	–	*	*	–	–	–
MOVW dir, A	2	3	0	(c)	word (dir) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW addr16, A	3	4	0	(c)	word (addr16) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW SP, A	1	1	0	0	word (SP) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW RWi, A	1	2	1	0	word (RWi) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW ear, A	2	2	1	0	word (ear) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW eam, A	2+	3+ (a)	0	(c)	word (eam) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW io, A	2	3	0	(c)	word (io) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW @RWi+disp8, A	2	5	1	(c)	word ((RWi) +disp8) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW @RLi+disp8, A	3	10	2	(c)	word ((RLi) +disp8) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVW RWi, ear	2	3	2	(0)	word (RWi) ← (ear)	–	–	–	–	–	*	*	–	–	–
MOVW RWi, eam	2+	4+ (a)	1	(c)	word (RWi) ← (eam)	–	–	–	–	–	*	*	–	–	–
MOVW ear, RWi	2	4	2	0	word (ear) ← (RWi)	–	–	–	–	–	*	*	–	–	–
MOVW eam, RWi	2+	5+ (a)	1	(c)	word (eam) ← (RWi)	–	–	–	–	–	*	*	–	–	–
MOVW RWi, #imm16	3	2	1	0	word (RWi) ← imm16	–	–	–	–	–	*	*	–	–	–
MOVW io, #imm16	4	5	0	(c)	word (io) ← imm16	–	–	–	–	–	–	–	–	–	–
MOVW ear, #imm16	4	2	1	0	word (ear) ← imm16	–	–	–	–	–	*	*	–	–	–
MOVW eam, #imm16	4+	4+ (a)	0	(c)	word (eam) ← imm16	–	–	–	–	–	–	–	–	–	–
MOVW @AL, AH /MOVW@A, T	2	3	0	(c)	word ((A)) ← (AH)	–	–	–	–	–	*	*	–	–	–
XCHW A, ear	2	4	2	0	word (A) ↔ (ear)	–	–	–	–	–	–	–	–	–	–
XCHW A, eam	2+	5+ (a)	0	2× (c)	word (A) ↔ (eam)	–	–	–	–	–	–	–	–	–	–
XCHW RWi, ear	2	7	4	0	word (RWi) ↔ (ear)	–	–	–	–	–	–	–	–	–	–
XCHW RWi, eam	2+	9+ (a)	2	2× (c)	word (RWi) ↔ (eam)	–	–	–	–	–	–	–	–	–	–
MOVL A, ear	2	4	2	0	long (A) ← (ear)	–	–	–	–	–	*	*	–	–	–
MOVL A, eam	2+	5+ (a)	0	(d)	long (A) ← (eam)	–	–	–	–	–	*	*	–	–	–
MOVL A, #imm32	5	3	0	0	long (A) ← imm32	–	–	–	–	–	*	*	–	–	–
MOVL ear, A	2	4	2	0	long (ear) ← (A)	–	–	–	–	–	*	*	–	–	–
MOVL eam, A	2+	5+ (a)	0	(d)	long (eam) ← (A)	–	–	–	–	–	*	*	–	–	–

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 9 Addition and Subtraction Instructions (Byte/Word/Long Word) [42 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
ADD A,#imm8	2	2	0	0	byte (A) ← (A) +imm8	Z	–	–	–	–	*	*	*	*	–
ADD A, dir	2	5	0	(b)	byte (A) ← (A) +(dir)	Z	–	–	–	–	*	*	*	*	–
ADD A, ear	2	3	1	0	byte (A) ← (A) +(ear)	Z	–	–	–	–	*	*	*	*	–
ADD A, eam	2+	4+ (a)	0	(b)	byte (A) ← (A) +(eam)	Z	–	–	–	–	*	*	*	*	–
ADD ear, A	2	3	2	0	byte (ear) ← (ear) + (A)	–	–	–	–	–	*	*	*	*	–
ADD eam, A	2+	5+ (a)	0	2× (b)	byte (eam) ← (eam) + (A)	Z	–	–	–	–	*	*	*	*	*
ADDC A	1	2	0	0	byte (A) ← (AH) + (AL) + (C)	Z	–	–	–	–	*	*	*	*	–
ADDC A, ear	2	3	1	0	byte (A) ← (A) + (ear) + (C)	Z	–	–	–	–	*	*	*	*	–
ADDC A, eam	2+	4+ (a)	0	(b)	byte (A) ← (A) + (eam) + (C)	Z	–	–	–	–	*	*	*	*	–
ADDDC A	1	3	0	0	byte (A) ← (AH) + (AL) + (C) (decimal)	Z	–	–	–	–	*	*	*	*	–
SUB A, #imm8	2	2	0	0	byte (A) ← (A) –imm8	Z	–	–	–	–	*	*	*	*	–
SUB A, dir	2	5	0	(b)	byte (A) ← (A) – (dir)	Z	–	–	–	–	*	*	*	*	–
SUB A, ear	2	3	1	0	byte (A) ← (A) – (ear)	Z	–	–	–	–	*	*	*	*	–
SUB A, eam	2+	4+ (a)	0	(b)	byte (A) ← (A) – (eam)	Z	–	–	–	–	*	*	*	*	–
SUB ear, A	2	3	2	0	byte (ear) ← (ear) – (A)	–	–	–	–	–	*	*	*	*	–
SUB eam, A	2+	5+ (a)	0	2× (b)	byte (eam) ← (eam) – (A)	–	–	–	–	–	*	*	*	*	*
SUBC A	1	2	0	0	byte (A) ← (AH) – (AL) – (C)	Z	–	–	–	–	*	*	*	*	–
SUBC A, ear	2	3	1	0	byte (A) ← (A) – (ear) – (C)	Z	–	–	–	–	*	*	*	*	–
SUBC A, eam	2+	4+ (a)	0	(b)	byte (A) ← (A) – (eam) – (C)	Z	–	–	–	–	*	*	*	*	–
SUBDC A	1	3	0	0	byte (A) ← (AH) – (AL) – (C) (decimal)	Z	–	–	–	–	*	*	*	*	–
ADDW A	1	2	0	0	word (A) ← (AH) + (AL)	–	–	–	–	–	*	*	*	*	–
ADDW A, ear	2	3	1	0	word (A) ← (A) +(ear)	–	–	–	–	–	*	*	*	*	–
ADDW A, eam	2+	4+ (a)	0	(c)	word (A) ← (A) +(eam)	–	–	–	–	–	*	*	*	*	–
ADDW A, #imm16	3	2	0	0	word (A) ← (A) +imm16	–	–	–	–	–	*	*	*	*	–
ADDW ear, A	2	3	2	0	word (ear) ← (ear) + (A)	–	–	–	–	–	*	*	*	*	–
ADDW eam, A	2+	5+ (a)	0	2× (c)	word (eam) ← (eam) + (A)	–	–	–	–	–	*	*	*	*	*
ADDCW A, ear	2	3	1	0	word (A) ← (A) + (ear) + (C)	–	–	–	–	–	*	*	*	*	–
ADDCW A, eam	2+	4+ (a)	0	(c)	word (A) ← (A) + (eam) + (C)	–	–	–	–	–	*	*	*	*	–
SUBW A	1	2	0	0	word (A) ← (AH) – (AL)	–	–	–	–	–	*	*	*	*	–
SUBW A, ear	2	3	1	0	word (A) ← (A) – (ear)	–	–	–	–	–	*	*	*	*	–
SUBW A, eam	2+	4+ (a)	0	(c)	word (A) ← (A) – (eam)	–	–	–	–	–	*	*	*	*	–
SUBW A, #imm16	3	2	0	0	word (A) ← (A) –imm16	–	–	–	–	–	*	*	*	*	–
SUBW ear, A	2	3	2	0	word (ear) ← (ear) – (A)	–	–	–	–	–	*	*	*	*	–
SUBW eam, A	2+	5+ (a)	0	2× (c)	word (eam) ← (eam) – (A)	–	–	–	–	–	*	*	*	*	*
SUBCW A, ear	2	3	1	0	word (A) ← (A) – (ear) – (C)	–	–	–	–	–	*	*	*	*	–
SUBCW A, eam	2+	4+ (a)	0	(c)	word (A) ← (A) – (eam) – (C)	–	–	–	–	–	*	*	*	*	–
ADDL A, ear	2	6	2	0	long (A) ← (A) + (ear)	–	–	–	–	–	*	*	*	*	–
ADDL A, eam	2+	7+ (a)	0	(d)	long (A) ← (A) + (eam)	–	–	–	–	–	*	*	*	*	–
ADDL A, #imm32	5	4	0	0	long (A) ← (A) +imm32	–	–	–	–	–	*	*	*	*	–
SUBL A, ear	2	6	2	0	long (A) ← (A) – (ear)	–	–	–	–	–	*	*	*	*	–
SUBL A, eam	2+	7+ (a)	0	(d)	long (A) ← (A) – (eam)	–	–	–	–	–	*	*	*	*	–
SUBL A, #imm32	5	4	0	0	long (A) ← (A) –imm32	–	–	–	–	–	*	*	*	*	–

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 10 Increment and Decrement Instructions (Byte/Word/Long Word) [12 Instructions]

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
INC	ear	2	2	2	0	byte (ear) ← (ear) +1	–	–	–	–	–	*	*	*	–	–
INC	eam	2+	5+ (a)	0	2× (b)	byte (eam) ← (eam) +1	–	–	–	–	–	*	*	*	–	*
DEC	ear	2	3	2	0	byte (ear) ← (ear) –1	–	–	–	–	–	*	*	*	–	–
DEC	eam	2+	5+ (a)	0	2× (b)	byte (eam) ← (eam) –1	–	–	–	–	–	*	*	*	–	*
INCW	ear	2	3	2	0	word (ear) ← (ear) +1	–	–	–	–	–	*	*	*	–	–
INCW	eam	2+	5+ (a)	0	2× (c)	word (eam) ← (eam) +1	–	–	–	–	–	*	*	*	–	*
DECW	ear	2	3	2	0	word (ear) ← (ear) –1	–	–	–	–	–	*	*	*	–	–
DECW	eam	2+	5+ (a)	0	2× (c)	word (eam) ← (eam) –1	–	–	–	–	–	*	*	*	–	*
INCL	ear	2	7	4	0	long (ear) ← (ear) +1	–	–	–	–	–	*	*	*	–	–
INCL	eam	2+	9+ (a)	0	2× (d)	long (eam) ← (eam) +1	–	–	–	–	–	*	*	*	–	*
DECL	ear	2	7	4	0	long (ear) ← (ear) –1	–	–	–	–	–	*	*	*	–	–
DECL	eam	2+	9+ (a)	0	2× (d)	long (eam) ← (eam) –1	–	–	–	–	–	*	*	*	–	*

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 11 Compare Instructions (Byte/Word/Long Word) [11 Instructions]

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
CMP	A	1	1	0	0	byte (AH) – (AL)	–	–	–	–	–	*	*	*	*	–
CMP	A, ear	2	2	1	0	byte (A) ← (ear)	–	–	–	–	–	*	*	*	*	–
CMP	A, eam	2+	3+ (a)	0	(b)	byte (A) ← (eam)	–	–	–	–	–	*	*	*	*	–
CMP	A, #imm8	2	2	0	0	byte (A) ← imm8	–	–	–	–	–	*	*	*	*	–
CMPW	A	1	1	0	0	word (AH) – (AL)	–	–	–	–	–	*	*	*	*	–
CMPW	A, ear	2	2	1	0	word (A) ← (ear)	–	–	–	–	–	*	*	*	*	–
CMPW	A, eam	2+	3+ (a)	0	(c)	word (A) ← (eam)	–	–	–	–	–	*	*	*	*	–
CMPW	A, #imm16	3	2	0	0	word (A) ← imm16	–	–	–	–	–	*	*	*	*	–
CMPL	A, ear	2	6	2	0	word (A) ← (ear)	–	–	–	–	–	*	*	*	*	–
CMPL	A, eam	2+	7+ (a)	0	(d)	word (A) ← (eam)	–	–	–	–	–	*	*	*	*	–
CMPL	A, #imm32	5	3	0	0	word (A) ← imm32	–	–	–	–	–	*	*	*	*	–

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 12 Multiplication and Division Instructions (Byte/Word/Long Word) [11 Instructions]

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
DIVU	A	1	*1	0	0	word (AH) /byte (AL) Quotient → byte (AL) Remainder → byte (AH)	–	–	–	–	–	–	–	*	*	–
DIVU	A, ear	2	*2	1	0	word (A)/byte (ear) Quotient → byte (A) Remainder → byte (ear)	–	–	–	–	–	–	–	*	*	–
DIVU	A, eam	2+	*3	0	*6	word (A)/byte (eam) Quotient → byte (A) Remainder → byte (eam)	–	–	–	–	–	–	–	*	*	–
DIVUW	A, ear	2	*4	1	0	long (A)/word (ear) Quotient → word (A) Remainder → word (ear)	–	–	–	–	–	–	–	*	*	–
DIVUW	A, eam	2+	*5	0	*7	long (A)/word (eam) Quotient → word (A) Remainder → word (ear)	–	–	–	–	–	–	–	*	*	–
MULU	A	1	*8	0	0	byte (AH) *byte (AL) → word (A)	–	–	–	–	–	–	–	–	–	–
MULU	A, ear	2	*9	1	0	byte (A) *byte (ear) → word (A)	–	–	–	–	–	–	–	–	–	–
MULU	A, eam	2+	*10	0	(b)	byte (A) *byte (eam) → word (A)	–	–	–	–	–	–	–	–	–	–
MULUW	A	1	*11	0	0	word (AH) *word (AL) → long (A)	–	–	–	–	–	–	–	–	–	–
MULUW	A, ear	2	*12	1	0	word (A) *word (ear) → long (A)	–	–	–	–	–	–	–	–	–	–
MULUW	A, eam	2+	*13	0	(c)	word (A) *word (eam) → long (A)	–	–	–	–	–	–	–	–	–	–

*1: 3 when the result is zero, 7 when an overflow occurs, and 15 normally.
*2: 4 when the result is zero, 8 when an overflow occurs, and 16 normally.
*3: 6 + (a) when the result is zero, 9 + (a) when an overflow occurs, and 19 + (a) normally.
*4: 4 when the result is zero, 7 when an overflow occurs, and 22 normally.
*5: 6 + (a) when the result is zero, 8 + (a) when an overflow occurs, and 26 + (a) normally.
*6: (b) when the result is zero or when an overflow occurs, and 2 × (b) normally.
*7: (c) when the result is zero or when an overflow occurs, and 2 × (c) normally.
*8: 3 when byte (AH) is zero, and 7 when byte (AH) is not zero.
*9: 4 when byte (ear) is zero, and 8 when byte (ear) is not zero.
*10: 5 + (a) when byte (eam) is zero, and 9 + (a) when byte (eam) is not 0.
*11: 3 when word (AH) is zero, and 11 when word (AH) is not zero.
*12: 4 when word (ear) is zero, and 12 when word (ear) is not zero.
*13: 5 + (a) when word (eam) is zero, and 13 + (a) when word (eam) is not zero.

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 13 Signed Multiplication and Division Instructions (Byte/Word/Long Word) [11 Instructions]

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
DIV	A	2	*1	0	0	word (AH) /byte (AL) Quotient → byte (AL) Remainder → byte (AH)	Z	–	–	–	–	–	–	*	*	–
DIV	A, ear	2	*2	1	0	word (A)/byte (ear) Quotient → byte (A) Remainder → byte (ear)	Z	–	–	–	–	–	–	*	*	–
DIV	A, eam	2 +	*3	0	*6	word (A)/byte (eam) Quotient → byte (A) Remainder → byte (eam)	Z	–	–	–	–	–	–	*	*	–
DIVW	A, ear	2	*4	1	0	long (A)/word (ear) Quotient → word (A) Remainder → word (ear)	–	–	–	–	–	–	–	*	*	–
DIVW	A, eam	2+	*5	0	*7	long (A)/word (eam) Quotient → word (A) Remainder → word (eam)	–	–	–	–	–	–	–	*	*	–
MULU	A	2	*8	0	0	byte (AH) *byte (AL) → word (A)	–	–	–	–	–	–	–	–	–	–
MULU	A, ear	2	*9	1	0	byte (A) *byte (ear) → word (A)	–	–	–	–	–	–	–	–	–	–
MULU	A, eam	2 +	*10	0	(b)	byte (A) *byte (eam) → word (A)	–	–	–	–	–	–	–	–	–	–
MULUW	A	2	*11	0	0	word (AH) *word (AL) → long (A)	–	–	–	–	–	–	–	–	–	–
MULUW	A, ear	2	*12	1	0	word (A) *word (ear) → long (A)	–	–	–	–	–	–	–	–	–	–
MULUW	A, eam	2 +	*13	0	(c)	word (A) *word (eam) → long (A)	–	–	–	–	–	–	–	–	–	–

*1: Set to 3 when the division-by-0, 8 or 18 for an overflow, and 18 for normal operation.
*2: Set to 3 when the division-by-0, 10 or 21 for an overflow, and 22 for normal operation.
*3: Set to 4 + (a) when the division-by-0, 11 + (a) or 22 + (a) for an overflow, and 23 + (a) for normal operation.
*4: Positive dividend: Set to 4 when the division-by-0, 10 or 29 for an overflow, and 30 for normal operation.
Negative dividend: Set to 4 when the division-by-0, 11 or 30 for an overflow and 31 for normal operation.
*5: Positive dividend: Set to 4 + (a) when the division-by-0, 11 + (a) or 30 + (a) for an overflow, and 31 + (a) for normal operation.
Negative dividend: Set to 4 + (a) when the division-by-0, 12 + (a) or 31 + (a) for an overflow, and 32 + (a) for normal operation.
*6: When the division-by-0, (b) for an overflow, and 2 × (b) for normal operation.
*7: When the division-by-0, (c) for an overflow, and 2 × (c) for normal operation.
*8: Set to 3 when byte (AH) is zero, 12 when the result is positive, and 13 when the result is negative.
*9: Set to 3 when byte (ear) is zero, 12 when the result is positive, and 13 when the result is negative.
*10: Set to 4 + (a) when byte (eam) is zero, 13 + (a) when the result is positive, and 14 + (a) when the result is negative.
*11: Set to 3 when word (AH) is zero, 12 when the result is positive, and 13 when the result is negative.
*12: Set to 3 when word (ear) is zero, 16 when the result is positive, and 19 when the result is negative.
*13: Set to 4 + (a) when word (eam) is zero, 17 + (a) when the result is positive, and 20 + (a) when the result is negative.

Notes:
- When overflow occurs during DIV or DIVW instruction execution, the number of execution cycles takes two values because of detection before and after an operation.
- When overflow occurs during DIV or DIVW instruction execution, the contents of AL are destroyed.
- For (a) to (d), refer to “Table 4 Number of Execution Cycles for Effective Address in Addressing Modes” and “Table 5 Correction Values for Number of Cycles for Calculating Actual Number of Cycles.”

Table 14 Logical 1 Instructions (Byte/Word) [39 Instructions]

Mnemonic		#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
AND	A, #imm8	2	2	0	0	byte (A) ← (A) and imm8	–	–	–	–	–	*	*	R	–	–
AND	A, ear	2	3	1	0	byte (A) ← (A) and (ear)	–	–	–	–	–	*	*	R	–	–
AND	A, eam	2+	4+ (a)	0	(b)	byte (A) ← (A) and (eam)	–	–	–	–	–	*	*	R	–	–
AND	ear, A	2	3	2	0	byte (ear) ← (ear) and (A)	–	–	–	–	–	*	*	R	–	–
AND	eam, A	2+	5+ (a)	0	2× (b)	byte (eam) ← (eam) and (A)	–	–	–	–	–	*	*	R	–	*
OR	A, #imm8	2	2	0	0	byte (A) ← (A) or imm8	–	–	–	–	–	*	*	R	–	–
OR	A, ear	2	3	1	0	byte (A) ← (A) or (ear)	–	–	–	–	–	*	*	R	–	–
OR	A, eam	2+	4+ (a)	0	(b)	byte (A) ← (A) or (eam)	–	–	–	–	–	*	*	R	–	–
OR	ear, A	2	3	2	0	byte (ear) ← (ear) or (A)	–	–	–	–	–	*	*	R	–	–
OR	eam, A	2+	5+ (a)	0	2× (b)	byte (eam) ← (eam) or (A)	–	–	–	–	–	*	*	R	–	*
XOR	A, #imm8	2	2	0	0	byte (A) ← (A) xor imm8	–	–	–	–	–	*	*	R	–	–
XOR	A, ear	2	3	1	0	byte (A) ← (A) xor (ear)	–	–	–	–	–	*	*	R	–	–
XOR	A, eam	2+	4+ (a)	0	(b)	byte (A) ← (A) xor (eam)	–	–	–	–	–	*	*	R	–	–
XOR	ear, A	2	3	2	0	byte (ear) ← (ear) xor (A)	–	–	–	–	–	*	*	R	–	–
XOR	eam, A	2+	5+ (a)	0	2× (b)	byte (eam) ← (eam) xor (A)	–	–	–	–	–	*	*	R	–	*
NOT	A	1	2	0	0	byte (A) ← not (A)	–	–	–	–	–	*	*	R	–	–
NOT	ear	2	3	2	0	byte (ear) ← not (ear)	–	–	–	–	–	*	*	R	–	–
NOT	eam	2+	5+ (a)	0	2× (b)	byte (eam) ← not (eam)	–	–	–	–	–	*	*	R	–	*
ANDW	A	1	2	0	0	word (A) ← (AH) and (A)	–	–	–	–	–	*	*	R	–	–
ANDW	A, #imm16	3	2	0	0	word (A) ← (A) and imm16	–	–	–	–	–	*	*	R	–	–
ANDW	A, ear	2	3	1	0	word (A) ← (A) and (ear)	–	–	–	–	–	*	*	R	–	–
ANDW	A, eam	2+	4+ (a)	0	(c)	word (A) ← (A) and (eam)	–	–	–	–	–	*	*	R	–	–
ANDW	ear, A	2	3	2	0	word (ear) ← (ear) and (A)	–	–	–	–	–	*	*	R	–	–
ANDW	eam, A	2+	5+ (a)	0	2× (c)	word (eam) ← (eam) and (A)	–	–	–	–	–	*	*	R	–	*
ORW	A	1	2	0	0	word (A) ← (AH) or (A)	–	–	–	–	–	*	*	R	–	–
ORW	A, #imm16	3	2	0	0	word (A) ← (A) or imm16	–	–	–	–	–	*	*	R	–	–
ORW	A, ear	2	3	1	0	word (A) ← (A) or (ear)	–	–	–	–	–	*	*	R	–	–
ORW	A, eam	2+	4+ (a)	0	(c)	word (A) ← (A) or (eam)	–	–	–	–	–	*	*	R	–	–
ORW	ear, A	2	3	2	0	word (ear) ← (ear) or (A)	–	–	–	–	–	*	*	R	–	–
ORW	eam, A	2+	5+ (a)	0	2× (c)	word (eam) ← (eam) or (A)	–	–	–	–	–	*	*	R	–	*
XORW	A	1	2	0	0	word (A) ← (AH) xor (A)	–	–	–	–	–	*	*	R	–	–
XORW	A, #imm16	3	2	0	0	word (A) ← (A) xor imm16	–	–	–	–	–	*	*	R	–	–
XORW	A, ear	2	3	1	0	word (A) ← (A) xor (ear)	–	–	–	–	–	*	*	R	–	–
XORW	A, eam	2+	4+ (a)	0	(c)	word (A) ← (A) xor (eam)	–	–	–	–	–	*	*	R	–	–
XORW	ear, A	2	3	2	0	word (ear) ← (ear) xor (A)	–	–	–	–	–	*	*	R	–	–
XORW	eam, A	2+	5+ (a)	0	2× (c)	word (eam) ← (eam) xor (A)	–	–	–	–	–	*	*	R	–	*
NOTW	A	1	2	0	0	word (A) ← not (A)	–	–	–	–	–	*	*	R	–	–
NOTW	ear	2	3	2	0	word (ear) ← not (ear)	–	–	–	–	–	*	*	R	–	–
NOTW	eam	2+	5+ (a)	0	2× (c)	word (eam) ← not (eam)	–	–	–	–	–	*	*	R	–	*

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 15 Logical 2 Instructions (Long Word) [6 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
ANDL A, ear	2	6	2	0	long (A) ← (A) and (ear)	–	–	–	–	–	*	*	R	–	–
ANDL A, eam	2+	7+ (a)	0	(d)	long (A) ← (A) and (eam)	–	–	–	–	–	*	*	R	–	–
ORL A, ear	2	6	2	0	long (A) ← (A) or (ear)	–	–	–	–	–	*	*	R	–	–
ORL A, eam	2+	7+ (a)	0	(d)	long (A) ← (A) or (eam)	–	–	–	–	–	*	*	R	–	–
XORL A, ea	2	6	2	0	long (A) ← (A) xor (ear)	–	–	–	–	–	*	*	R	–	–
XORL A, eam	2+	7+ (a)	0	(d)	long (A) ← (A) xor (eam)	–	–	–	–	–	*	*	R	–	–

Table 16 Sign Inversion Instructions (Byte/Word) [6 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
NEG A	1	2	0	0	byte (A) ← 0 – (A)	X	–	–	–	–	*	*	*	*	–
NEG ear	2	3	2	0	byte (ear) ← 0 – (ear)	–	–	–	–	–	*	*	*	*	–
NEG eam	2+	5+ (a)	0	2× (b)	byte (eam) ← 0 – (eam)	–	–	–	–	–	*	*	*	*	*
NEGW A	1	2	0	0	word (A) ← 0 – (A)	–	–	–	–	–	*	*	*	*	–
NEGW ear	2	3	2	0	word (ear) ← 0 – (ear)	–	–	–	–	–	*	*	*	*	–
NEGW eam	2+	5+ (a)	0	2× (c)	word (eam) ← 0 – (eam)	–	–	–	–	–	*	*	*	*	*

Table 17 Normalize Instruction (Long Word) [1 Instruction]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
NRML A, R0	2	*1	1	0	long (A) ← Shift until first digit is “1” byte (R0) ← Current shift count	–	–	–	–	–	–	*	–	–	–

*1: 4 when the contents of the accumulator are all zeroes, 6 + (R0) in all other cases (shift count).

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 18 Shift Instructions (Byte/Word/Long Word) [18 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
RORC A	2	2	0	0	byte (A) ← Right rotation with carry	–	–	–	–	–	*	*	–	*	–
ROLC A	2	2	0	0	byte (A) ← Left rotation with carry	–	–	–	–	–	*	*	–	*	–
RORC ear	2	3	2	0	byte (ear) ← Right rotation with carry	–	–	–	–	–	*	*	–	*	–
RORC eam	2+	5+ (a)	0	2× (b)	byte (eam) ← Right rotation with carry	–	–	–	–	–	*	*	–	*	*
ROLC ear	2	3	2	0	byte (ear) ← Left rotation with carry	–	–	–	–	–	*	*	–	*	–
ROLC eam	2+	5+ (a)	0	2× (b)	byte (eam) ← Left rotation with carry	–	–	–	–	–	*	*	–	*	*
ASR A, R0	2	*1	1	0	byte (A) ← Arithmetic right barrel shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSR A, R0	2	*1	1	0	byte (A) ← Logical right barrel shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSL A, R0	2	*1	1	0	byte (A) ← Logical left barrel shift (A, R0)	–	–	–	–	–	*	*	–	*	–
ASRW A	1	2	0	0	word (A) ← Arithmetic right shift (A, 1 bit)	–	–	–	–	*	*	*	–	*	–
LSRW A/SHRW A	1	2	0	0	word (A) ← Logical right shift (A, 1 bit)	–	–	–	–	*	R	*	–	*	–
LSLW A/SHLW A	1	2	0	0	word (A) ← Logical left shift (A, 1 bit)	–	–	–	–	–	*	*	–	*	–
ASRW A, R0	2	*1	1	0	word (A) ← Arithmetic right barrel shift (A,	–	–	–	–	*	*	*	–	*	–
LSRW A, R0	2	*1	1	0	R0)	–	–	–	–	*	*	*	–	*	–
LSLW A, R0	2	*1	1	0	word (A) ← Logical right barrel shift (A, R0) word (A) ← Logical left barrel shift (A, R0)	–	–	–	–	–	*	*	–	*	–
ASRL A, R0	2	*2	1	0	long (A) ← Arithmetic right shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSRL A, R0	2	*2	1	0	long (A) ← Logical right barrel shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSLL A, R0	2	*2	1	0	long (A) ← Logical left barrel shift (A, R0)	–	–	–	–	–	*	*	–	*	–

*1: 6 when R0 is 0, 5 + (R0) in all other cases.

*2: 6 when R0 is 0, 6 + (R0) in all other cases.

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 19 Branch 1 Instructions [31 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
BZ/BEQ rel	2	*1	0	0	Branch when (Z) = 1	–	–	–	–	–	–	–	–	–	–
BNZ/BNE rel	2	*1	0	0	Branch when (Z) = 0	–	–	–	–	–	–	–	–	–	–
BC/BLO rel	2	*1	0	0	Branch when (C) = 1	–	–	–	–	–	–	–	–	–	–
BNC/BHS rel	2	*1	0	0	Branch when (C) = 0	–	–	–	–	–	–	–	–	–	–
BN rel	2	*1	0	0	Branch when (N) = 1	–	–	–	–	–	–	–	–	–	–
BP rel	2	*1	0	0	Branch when (N) = 0	–	–	–	–	–	–	–	–	–	–
BV rel	2	*1	0	0	Branch when (V) = 1	–	–	–	–	–	–	–	–	–	–
BNV rel	2	*1	0	0	Branch when (V) = 0	–	–	–	–	–	–	–	–	–	–
BT rel	2	*1	0	0	Branch when (T) = 1	–	–	–	–	–	–	–	–	–	–
BNT rel	2	*1	0	0	Branch when (T) = 0	–	–	–	–	–	–	–	–	–	–
BLT rel	2	*1	0	0	Branch when (V) xor (N) = 1	–	–	–	–	–	–	–	–	–	–
BGE rel	2	*1	0	0	Branch when (V) xor (N) = 0	–	–	–	–	–	–	–	–	–	–
BLE rel	2	*1	0	0	Branch when ((V) xor (N)) or (Z) = 1	–	–	–	–	–	–	–	–	–	–
BGT rel	2	*1	0	0	Branch when ((V) xor (N)) or (Z) = 0	–	–	–	–	–	–	–	–	–	–
BLS rel	2	*1	0	0	Branch when (C) or (Z) = 1	–	–	–	–	–	–	–	–	–	–
BHI rel	2	*1	0	0	Branch when (C) or (Z) = 0	–	–	–	–	–	–	–	–	–	–
BRA rel	2	*1	0	0	Branch unconditionally	–	–	–	–	–	–	–	–	–	–
JMP @A	1	2	0	0	word (PC) ← (A)	–	–	–	–	–	–	–	–	–	–
JMP addr16	3	3	0	0	word (PC) ← addr16	–	–	–	–	–	–	–	–	–	–
JMP @ear	2	3	1	0	word (PC) ← (ear)	–	–	–	–	–	–	–	–	–	–
JMP @eam	2+	4+ (a)	0	(c)	word (PC) ← (eam)	–	–	–	–	–	–	–	–	–	–
JMPP @ear *3	2	5	2	0	word (PC) ← (ear), (PCB) ← (ear +2)	–	–	–	–	–	–	–	–	–	–
JMPP @eam *3	2+	6+ (a)	0	(d)	word (PC) ← (eam), (PCB) ← (eam +2)	–	–	–	–	–	–	–	–	–	–
JMPP addr24	4	4	0	0	word (PC) ← ad24 0 to 15, (PCB) ← ad24 16 to 23	–	–	–	–	–	–	–	–	–	–
CALL @ear *4	2	6	1	(c)	word (PC) ← (ear)	–	–	–	–	–	–	–	–	–	–
CALL @eam *4	2+	7+ (a)	0	2× (c)	word (PC) ← (eam)	–	–	–	–	–	–	–	–	–	–
CALL addr16 *5	3	6	0	(c)	word (PC) ← addr16	–	–	–	–	–	–	–	–	–	–
CALLV #vct4 *5	1	7	0	2× (c)	Vector call instruction	–	–	–	–	–	–	–	–	–	–
CALLP @ear *6	2	10	2	2× (c)	word (PC) ← (ear) 0 to 15, (PCB) ← (ear) 16 to 23	–	–	–	–	–	–	–	–	–	–
CALLP @eam *6	2+	11+ (a)	0	*2	word (PC) ← (eam) 0 to 15, (PCB) ← (eam) 16 to 23	–	–	–	–	–	–	–	–	–	–
CALLP addr24 *7	4	10	0	2× (c)	word (PC) ← addr0 to 15, (PCB) ← addr16 to 23	–	–	–	–	–	–	–	–	–	–

*1: 4 when branching, 3 when not branching.
*2: (b) + 3 × (c)
*3: Read (word) branch address.
*4: W: Save (word) to stack; R: read (word) branch address.
*5: Save (word) to stack.
*6: W: Save (long word) to W stack; R: read (long word) R branch address.
*7: Save (long word) to stack.

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 20 Branch 2 Instructions [19 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
CBNE A, #imm8, rel	3	*1	0	0	Branch when byte (A) ≠ imm8	–	–	–	–	–	*	*	*	*	–
CWBNE A, #imm16, rel	4	*1	0	0	Branch when word (A) ≠ imm16	–	–	–	–	–	*	*	*	*	–
CBNE ear, #imm8, rel	4	*2	1	0	Branch when byte (ear) ≠ imm8	–	–	–	–	–	*	*	*	*	–
CBNE eam, #imm8, rel*10	4+	*3	0	(b)	Branch when byte (eam) ≠ imm8	–	–	–	–	–	*	*	*	*	–
CWBNE ear, #imm16, rel	5	*4	1	0	Branch when word (ear) ≠ imm16	–	–	–	–	–	*	*	*	*	–
CWBNE eam, #imm16, rel*10	5+	*3	0	(c)	Branch when word (eam) ≠ imm16	–	–	–	–	–	*	*	*	*	–
DBNZ ear, rel	3	*5	2	0	Branch when byte (ear) = (ear) – 1, and (ear) ≠ 0	–	–	–	–	–	*	*	*	–	–
DBNZ eam, rel	3+	*6	2	2× (b)	Branch when byte (eam) = (eam) – 1, and (eam) ≠ 0	–	–	–	–	–	*	*	*	–	*
DWBNZ ear, rel	3	*5	2	0	Branch when word (ear) = (ear) – 1, and (ear) ≠ 0	–	–	–	–	–	*	*	*	–	–
DWBNZ eam, rel	3+	*6	2	2× (c)	Branch when word (eam) = (eam) – 1, and (eam) ≠ 0	–	–	–	–	–	*	*	*	–	*
INT #vct8	2	20	0	8× (c)	Software interrupt	–	–	R	S	–	–	–	–	–	–
INT addr16	3	16	0	6× (c)	Software interrupt	–	–	R	S	–	–	–	–	–	–
INTP addr24	4	17	0	6× (c)	Software interrupt	–	–	R	S	–	–	–	–	–	–
INT9	1	20	0	8× (c)	Software interrupt	–	–	R	S	–	–	–	–	–	–
RETI	1	15	0	*7	Return from interrupt	–	–	*	*	*	*	*	*	*	–
LINK #local8	2	6	0	(c)	At constant entry, save old frame pointer to stack, set new frame pointer, and allocate local pointer area	–	–	–	–	–	–	–	–	–	–
UNLINK	1	5	0	(c)	At constant entry, retrieve old frame pointer from stack.	–	–	–	–	–	–	–	–	–	–
RET *8	1	4	0	(c)	Return from subroutine	–	–	–	–	–	–	–	–	–	–
RETP *9	1	6	0	(d)	Return from subroutine	–	–	–	–	–	–	–	–	–	–

*1: 5 when branching, 4 when not branching
*2: 13 when branching, 12 when not branching
*3: 7 + (a) when branching, 6 + (a) when not branching
*4: 8 when branching, 7 when not branching
*5: 7 when branching, 6 when not branching
*6: 8 + (a) when branching, 7 + (a) when not branching
*7: Set to 3 × (b) + 2 × (c) when an interrupt request occurs, and 6 × (c) for return.
*8: Retrieve (word) from stack
*9: Retrieve (long word) from stack
*10: In the CBNE/CWBNE instruction, do not use the RWj+ addressing mode.

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 21 Other Control Instructions (Byte/Word/Long Word) [36 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
PUSHW A	1	4	0	(c)	word (SP) ← (SP) –2, ((SP)) ← (A)	–	–	–	–	–	–	–	–	–	–
PUSHW AH	1	4	0	(c)	word (SP) ← (SP) –2, ((SP)) ← (AH)	–	–	–	–	–	–	–	–	–	–
PUSHW PS	1	4	0	(c)	word (SP) ← (SP) –2, ((SP)) ← (PS)	–	–	–	–	–	–	–	–	–	–
PUSHW rlst	2	*3	*5	*4	(SP) ← (SP) –2n, ((SP)) ← (rlst)	–	–	–	–	–	–	–	–	–	–
POPW A	1	3	0	(c)	word (A) ← ((SP)), (SP) ← (SP) +2	–	*	–	–	–	–	–	–	–	–
POPW AH	1	3	0	(c)	word (AH) ← ((SP)), (SP) ← (SP) +2	–	–	–	–	–	–	–	–	–	–
POPW PS	1	4	0	(c)	word (PS) ← ((SP)), (SP) ← (SP) +2	–	–	*	*	*	*	*	*	*	–
POPW rlst	2	*2	*5	*4	(rlst) ← ((SP)), (SP) ← (SP) +2n	–	–	–	–	–	–	–	–	–	–
JCTX @A	1	14	0	6× (c)	Context switch instruction	–	–	*	*	*	*	*	*	*	–
AND CCR, #imm8	2	3	0	0	byte (CCR) ← (CCR) and imm8	–	–	*	*	*	*	*	*	*	–
OR CCR, #imm8	2	3	0	0	byte (CCR) ← (CCR) or imm8	–	–	*	*	*	*	*	*	*	–
MOV RP, #imm8	2	2	0	0	byte (RP) ←imm8	–	–	–	–	–	–	–	–	–	–
MOV ILM, #imm8	2	2	0	0	byte (ILM) ←imm8	–	–	–	–	–	–	–	–	–	–
MOVEA RWi, ear	2	3	1	0	word (RWi) ←ear	–	–	–	–	–	–	–	–	–	–
MOVEA RWi, eam	2+	2+ (a)	1	0	word (RWi) ←eam	–	–	–	–	–	–	–	–	–	–
MOVEA A, ear	2	1	0	0	word(A) ←ear	–	*	–	–	–	–	–	–	–	–
MOVEA A, eam	2+	1+ (a)	0	0	word (A) ←eam	–	*	–	–	–	–	–	–	–	–
ADDSP #imm8	2	3	0	0	word (SP) ← (SP) +ext (imm8)	–	–	–	–	–	–	–	–	–	–
ADDSP #imm16	3	3	0	0	word (SP) ← (SP) +imm16	–	–	–	–	–	–	–	–	–	–
MOV A, brgl	2	*1	0	0	byte (A) ← (brgl)	Z	*	–	–	–	*	*	–	–	–
MOV brg2, A	2	1	0	0	byte (brg2) ← (A)	–	–	–	–	–	*	*	–	–	–
NOP	1	1	0	0	No operation	–	–	–	–	–	–	–	–	–	–
ADB	1	1	0	0	Prefix code for accessing AD space	–	–	–	–	–	–	–	–	–	–
DTB	1	1	0	0	Prefix code for accessing DT space	–	–	–	–	–	–	–	–	–	–
PCB	1	1	0	0	Prefix code for accessing PC space	–	–	–	–	–	–	–	–	–	–
SPB	1	1	0	0	Prefix code for accessing SP space	–	–	–	–	–	–	–	–	–	–
NCC	1	1	0	0	Prefix code for no flag change	–	–	–	–	–	–	–	–	–	–
CMR	1	1	0	0	Prefix code for common register bank	–	–	–	–	–	–	–	–	–	–

*1: PCB, ADB, SSB, USB, and SPB : 1 state
DTB, DPR : 2 states

*2: 7 + 3 × (pop count) + 2 × (last register number to be popped), 7 when rlst = 0 (no transfer register)

*3: 29 + (push count) – 3 × (last register number to be pushed), 8 when rlst = 0 (no transfer register)

*4: Pop count × (c), or push count × (c)

*5: Pop count or push count.

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 22 Bit Manipulation Instructions [21 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVB A, dir:bp	3	5	0	(b)	byte (A) ← (dir:bp) b	Z	*	–	–	–	*	*	–	–	–
MOVB A, addr16:bp	4	5	0	(b)	byte (A) ← (addr16:bp) b	Z	*	–	–	–	*	*	–	–	–
MOVB A, io:bp	3	4	0	(b)	byte (A) ← (io:bp) b	Z	*	–	–	–	*	*	–	–	–
MOVB dir:bp, A	3	7	0	2× (b)	bit (dir:bp) b ← (A)	–	–	–	–	–	*	*	–	–	*
MOVB addr16:bp, A	4	7	0	2× (b)	bit (addr16:bp) b ← (A)	–	–	–	–	–	*	*	–	–	*
MOVB io:bp, A	3	6	0	2× (b)	bit (io:bp) b ← (A)	–	–	–	–	–	*	*	–	–	*
SETB dir:bp	3	7	0	2× (b)	bit (dir:bp) b ← 1	–	–	–	–	–	–	–	–	–	*
SETB addr16:bp	4	7	0	2× (b)	bit (addr16:bp) b ← 1	–	–	–	–	–	–	–	–	–	*
SETB io:bp	3	7	0	2× (b)	bit (io:bp) b ← 1	–	–	–	–	–	–	–	–	–	*
CLRB dir:bp	3	7	0	2× (b)	bit (dir:bp) b ← 0	–	–	–	–	–	–	–	–	–	*
CLRB addr16:bp	4	7	0	2× (b)	bit (addr16:bp) b ← 0	–	–	–	–	–	–	–	–	–	*
CLRB io:bp	3	7	0	2× (b)	bit (io:bp) b ← 0	–	–	–	–	–	–	–	–	–	*
BBC dir:bp, rel	4	*1	0	(b)	Branch when (dir:bp) b = 0	–	–	–	–	–	–	*	–	–	–
BBC addr16:bp, rel	5	*1	0	(b)	Branch when (addr16:bp) b = 0	–	–	–	–	–	–	*	–	–	–
BBC io:bp, rel	4	*2	0	(b)	Branch when (io:bp) b = 0	–	–	–	–	–	–	*	–	–	–
BBS dir:bp, rel	4	*1	0	(b)	Branch when (dir:bp) b = 1	–	–	–	–	–	–	*	–	–	–
BBS addr16:bp, rel	5	*1	0	(b)	Branch when (addr16:bp) b = 1	–	–	–	–	–	–	*	–	–	–
BBS io:bp, rel	4	*2	0	(b)	Branch when (io:bp) b = 1	–	–	–	–	–	–	*	–	–	–
SBBS addr16:bp, rel	5	*3	0	2× (b)	Branch when (addr16:bp) b = 1, bit = 1	–	–	–	–	–	–	*	–	–	*
WBTS io:bp	3	*4	0	*5	Wait until (io:bp) b = 1	–	–	–	–	–	–	–	–	–	–
WBTC io:bp	3	*4	0	*5	Wait until (io:bp) b = 0	–	–	–	–	–	–	–	–	–	–

*1: 8 when branching, 7 when not branching
*2: 7 when branching, 6 when not branching
*3: 10 when condition is satisfied, 9 when not satisfied
*4: Undefined count
*5: Until condition is satisfied

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

Table 23 Accumulator Manipulation Instructions (Byte/Word) [6 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
SWAP	1	3	0	0	byte (A) 0 to 7 ↔ (A) 8 to 15	–	–	–	–	–	–	–	–	–	–
SWAPW	1	2	0	0	word (AH) ↔ (AL)	–	*	–	–	–	–	–	–	–	–
EXT	1	1	0	0	byte sign extension	X	–	–	–	–	*	*	–	–	–
EXTW	1	2	0	0	word sign extension	–	X	–	–	–	*	*	–	–	–
ZEXT	1	1	0	0	byte zero extension	Z	–	–	–	–	R	*	–	–	–
ZEXTW	1	1	0	0	word zero extension	–	Z	–	–	–	R	*	–	–	–

Table 24 String Instructions [10 Instructions]

Mnemonic	#	~	RG	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
MOVS/MOVSI	2	*2	*5	*3	Byte transfer @AH+ ← @AL+, counter = RW0	–	–	–	–	–	–	–	–	–	–
MOVSD	2	*2	*5	*3	Byte transfer @AH– ← @AL–, counter = RW0	–	–	–	–	–	–	–	–	–	–
SCEQ/SCEQI	2	*1	*5	*4	Byte retrieval (@AH+) – AL, counter = RW0	–	–	–	–	–	*	*	*	*	–
SCEQD	2	*1	*5	*4	Byte retrieval (@AH–) – AL, counter = RW0	–	–	–	–	–	*	*	*	*	–
FISL/FILSI	2	6m +6	*5	*3	Byte filling @AH+ ← AL, counter = RW0	–	–	–	–	–	*	*	–	–	–
MOVSW/MOVSWI	2	*2	*8	*6	Word transfer @AH+ ← @AL+, counter = RW0	–	–	–	–	–	–	–	–	–	–
MOVSWD	2	*2	*8	*6	Word transfer @AH– ← @AL–, counter = RW0	–	–	–	–	–	–	–	–	–	–
SCWEQ/SCWEQI	2	*1	*8	*7	Word retrieval (@AH+) – AL, counter = RW0	–	–	–	–	–	*	*	*	*	–
SCWEQD	2	*1	*8	*7	Word retrieval (@AH–) – AL, counter = RW0	–	–	–	–	–	*	*	*	*	–
FILSW/FILSWI	2	6m +6	*8	*6	Word filling @AH+ ← AL, counter = RW0	–	–	–	–	–	*	*	–	–	–

m: RW0 value (counter value)
n: Loop count
*1: 5 when RW0 is 0, 4 + 7 × (RW0) for count out, and 7 × n + 5 when match occurs
*2: 5 when RW0 is 0, 4 + 8 × (RW0) in any other case
*3: (b) × (RW0) + (b) × (RW0) when accessing different areas for the source and destination, calculate (b) separately for each.
*4: (b) × n
*5: 2 × (RW0)
*6: (c) × (RW0) + (c) × (RW0) when accessing different areas for the source and destination, calculate (c) separately for each.
*7: (c) × n
*8: 2 × (RW0)

Note : For an explanation of “(a)” to “(d)”, refer to Table 4, “Number of Execution Cycles for Each Type of Addressing,” and Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

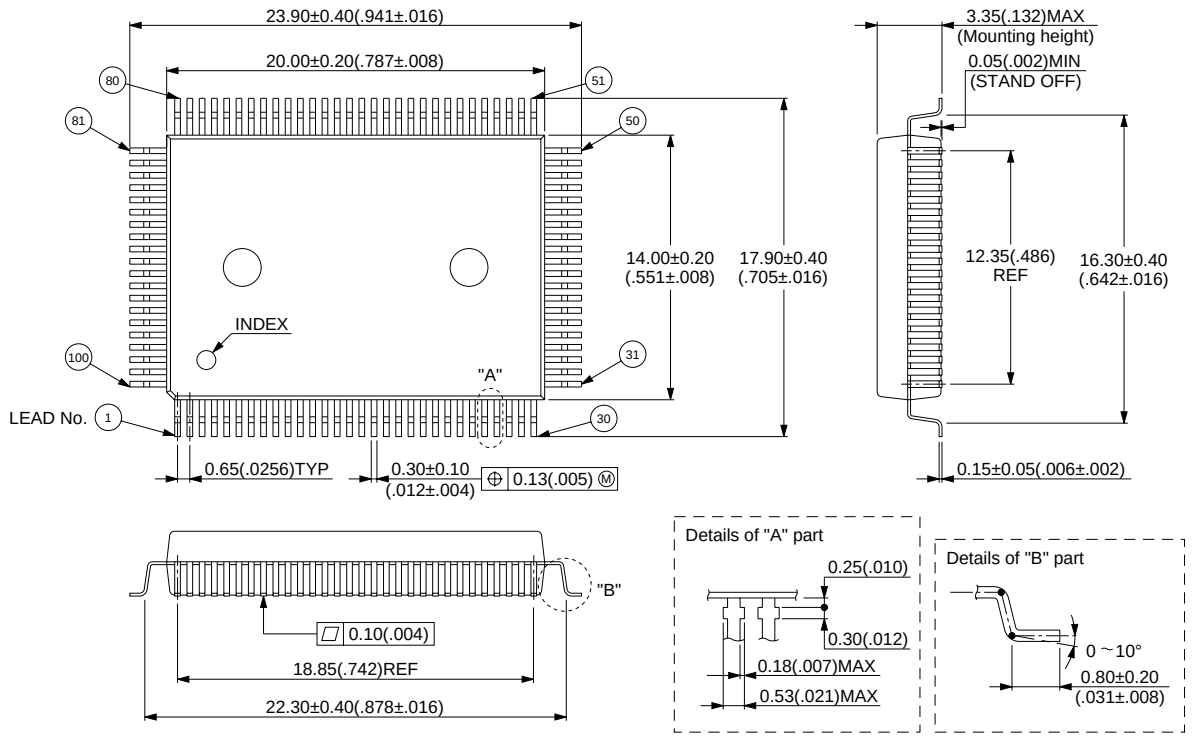
■ ORDERING INFORMATION

Part number	Package	Remarks
MB90549PF MB90591PF MB90F594APF MB90591PF	100-pin Plastic QFP (FPT-100P-M06)	
MB90V590ACR	256-pin Ceramic PGA (PGA-256C-A01)	For evaluation

MB90590 Series

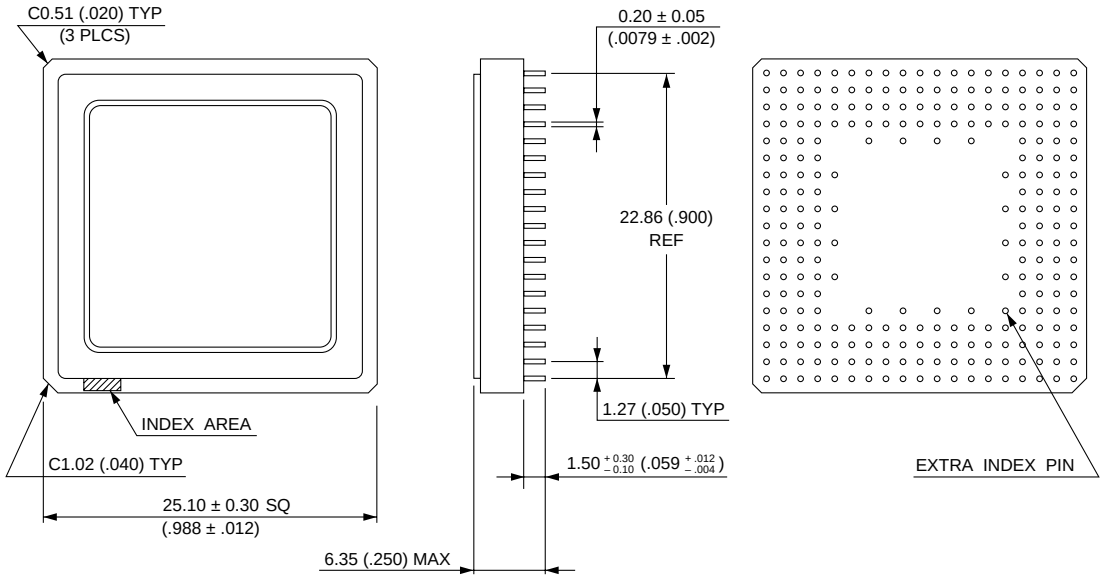
■ PACKAGE DIMENSION

100-pin plastic QFP
(FPT-100P-M06)



Dimensions in mm (inches)

250-pin ceramic PGA
(PGA-256-A01)



Dimensions in mm (inches)

MB90590 Series

FUJITSU LIMITED

For further information please contact:

Japan
FUJITSU LIMITED
Corporate Global Business Support Division
Electronic Devices
KAWASAKI PLANT, 4-1-1, Kamikodanaka
Nakahara-ku, Kawasaki-shi
Kanagawa 211-8588, Japan
Tel: 81(44) 754-3763
Fax: 81(44) 754-3329

http://www.fujitsu.co.jp/

North and South America
FUJITSU MICROELECTRONICS, INC.
Semiconductor Division
3545 North First Street
San Jose, CA 95134-1804, USA
Tel: (408) 922-9000
Fax: (408) 922-9179

Customer Response Center
Mon. - Fri.: 7 am - 5 pm (PST)
Tel: (800) 866-8608
Fax: (408) 922-9179

http://www.fujitsumicro.com/

Europe
FUJITSU MICROELECTRONICS EUROPE GmbH
Am Siebenstein 6-10
D-63303 Dreieich-Buchschlag
Germany
Tel: (06103) 690-0
Fax: (06103) 690-122

http://www.fujitsu-ede.com/

Asia Pacific
FUJITSU MICROELECTRONICS ASIA PTE LTD
#05-08, 151 Lorong Chuan
New Tech Park
Singapore 556741
Tel: (65) 281-0770
Fax: (65) 281-0220

http://www.fmap.com.sg/

F9909

The contents of this document are subject to change without notice. Customers are advised to consult with FUJITSU sales representatives before ordering.

The information and circuit diagrams in this document are presented as examples of semiconductor device applications, and are not intended to be incorporated in devices for actual use. Also, FUJITSU is unable to assume responsibility for infringement of any patent rights or other rights of third parties arising from the use of this information or circuit diagrams.

FUJITSU semiconductor devices are intended for use in standard applications (computers, office automation and other office equipment, industrial, communications, and measurement equipment, personal or household devices, etc.).

CAUTION:
Customers considering the use of our products in special applications where failure or abnormal operation may directly affect human lives or cause physical injury or property damage, or where extremely high levels of reliability are demanded (such as aerospace systems, atomic energy controls, sea floor repeaters, vehicle operating controls, medical devices for life support, etc.) are requested to consult with FUJITSU sales representatives before such use. The company will not be responsible for damages arising from such use without prior approval.

Any semiconductor devices have an inherent chance of failure. You must protect against injury, damage or loss from such failures by incorporating safety design measures into your facility and equipment such as redundancy, fire protection, and prevention of over-current levels and other abnormal operating conditions.

If any products described in this document represent goods or technologies subject to certain restrictions on export under the Foreign Exchange and Foreign Trade Law of Japan, the prior authorization by Japanese government will be required for export of those products from Japan.