Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

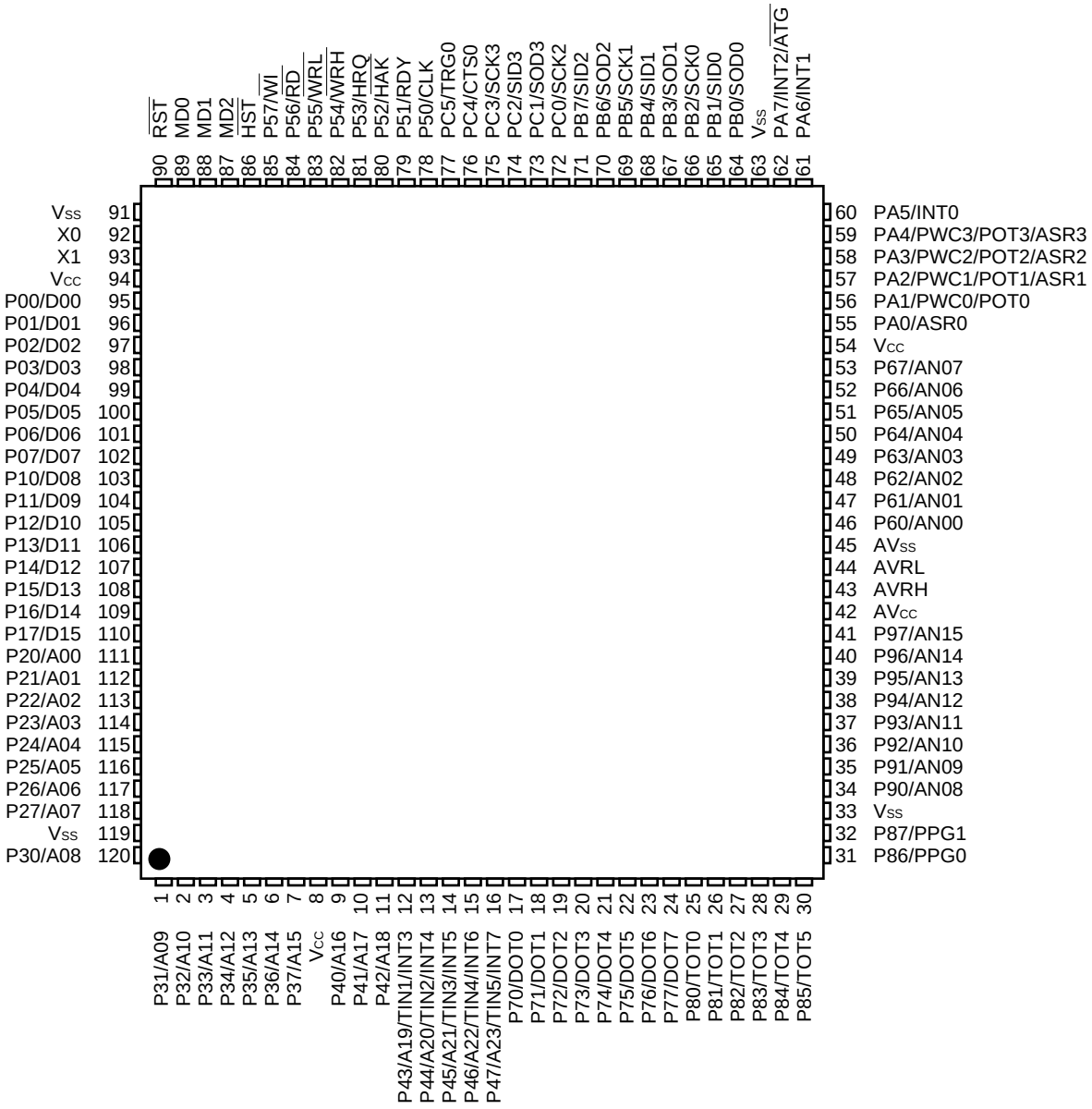
"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	F²MC-16F
Core Size	16-Bit
Speed	16MHz
Connectivity	EBI/EMI, UART/USART
Peripherals	POR, PWM, WDT
Number of I/O	102
Program Memory Size	96KB (96K x 8)
Program Memory Type	Mask ROM
EEPROM Size	-
RAM Size	4.5K x 8
Voltage - Supply (Vcc/Vdd)	3V ~ 5.5V
Data Converters	A/D 16x10b
Oscillator Type	External
Operating Temperature	-40°C ~ 105°C (TA)
Mounting Type	Surface Mount
Package / Case	120-BQFP
Supplier Device Package	120-QFP (28x28)
Purchase URL	https://www.e-xfl.com/product-detail/infineon-technologies/mb90224pf-gt-368

■ PIN ASSIGNMENT

(Top view)



(FPT-120P-M03)
(FPT-120C-C02)

■ PIN DESCRIPTION

Pin no. QFP*	Pin name	Circuit type	Function
92, 93	X0, X1	A	Crystal oscillation pins (16 MHz)
89 to 87	MD0 to MD2	D	Operation mode specification input pins Connect directly to V_{CC} or V_{SS}.
90	$\overline{RST}$	G	External reset request input
86	$\overline{HST}$	E	Hardware standby input pin
95 to 102	P00 to P07	C	General-purpose I/O ports This function is valid only in single-chip mode.
	D00 to D07		Output pins for low-order 8 bits of the external address bus. This function is valid only in modes where the external bus is enabled.
103 to 110	P10 to P17	C	General-purpose I/O ports This function is valid only in single-chip mode or when the external bus is enabled and the 8-bit data bus specification has been made.
	D08 to D15		I/O pins for higher-order 8 bits of the external data bus This function is valid only when the external bus is enabled and the 16-bit bus specification has been made.
111 to 118	P20 to P27	C	General-purpose I/O ports This function is valid only in single-chip mode.
	A00 to A07		Output pins for lower-order 8 bits of the external address bus This function is valid only in modes where the external bus is enabled.
120, 1 to 7	P30, P31 to P37	C	General-purpose I/O ports This function is valid either in single-chip mode or when the address mid-order control register specification is “port”.
	A08, A09 to A15		Output pins for mid-order 8 bits of the external address bus This function is valid in modes where the external bus is enabled and the address mid-order control register specification is “address”.
9 to 11	P40 to P42	C	General-purpose I/O ports This function is valid either in single-chip mode or when the address high-order control register specification is “port”.
	A16 to A18		Output pins for higher-order 8 bits of the external address bus This function is valid in modes where the external bus is enabled and the address high-order control register specification is “address”.
12 to 16	P43 to P47	C	General-purpose I/O ports This function is valid when either single-chip mode is enabled or the address higher-order control register specification is “port”.
	A19 to A23		Output pins for higher-order 8 bits of the external address bus This function is valid in modes where the external bus is enabled and the address higher-order control register specification is “address”.
	TIN1 to TIN5		16-bit reload timer input pins This function is valid when the timer input specification is “enabled”. The data on the pins is read as timer input (TIN1 to TIN5).

* : FPT-120P-M03, FPT-120C-C02

(Continued)

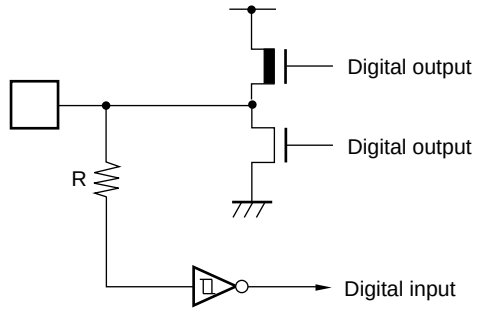
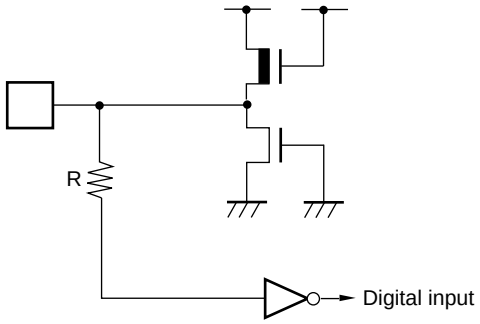
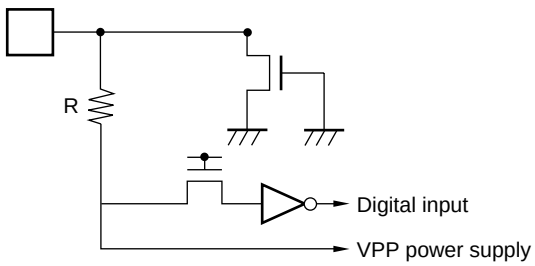
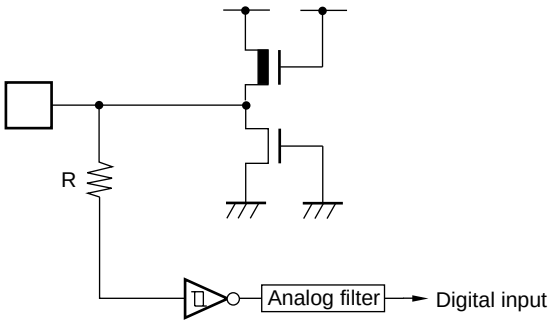
Pin no. QFP*	Pin name	Circuit type	Function
12 to 16	INT3 to INT7	C	External interrupt request input pins When external interrupts are enabled, these inputs may be used suddenly at any time; therefore, it is necessary to stop output by other functions on these pins, except when using them for output deliberately.
78	P50	C	General-purpose I/O port This function is valid in single-chip mode and when the CLK output specification is disabled.
	CLK		CLK output pin This function is valid in modes where the external bus is enabled and the CLK output specification is enabled.
79	P51	C	General-purpose I/O port This function is valid in single-chip mode or when the ready function is disabled.
	RDY		Ready input pin This function is valid in modes where the external bus is enabled and the ready function is enabled.
80	P52	C	General-purpose I/O port This function is valid in single-chip mode or when the hold function is disabled.
	$\overline{\text{HAK}}$		Hold acknowledge output pin This function is valid in modes where the external bus is enabled and the hold function is enabled.
81	P53	C	General-purpose I/O port This function is valid in single-chip mode or external bus mode and when the hold function is disabled.
	HRQ		Hold request input pin This function is valid in modes where the external bus is enabled and the hold function is enabled. During this operation, the input may be used suddenly at any time; therefore, it is necessary to stop output by other fuctions on this pin, except when using it for output deliberately.
82	P54	C	General-purpose I/O port This function is valid in single-chip mode, when the external bus is in 8-bit mode, or when $\overline{\text{WRH}}$ pin output is disabled.
	$\overline{\text{WRH}}$		Write strobe output pin for the high-order 8 bits of the data bus This function is valid in modes where the external bus is enabled, the external bus is in 16-bit mode, and $\overline{\text{WRH}}$ pin output is enabled.
83	P55	C	General-purpose I/O port This function is valid in single-chip mode or when $\overline{\text{WRL}}$ pin output is disabled.
	$\overline{\text{WRL}}$		Write strobe output pin for the low-order 8 bits of the data bus This function is valid in modes where the external bus is enabled and $\overline{\text{WRL}}$ pin output is enabled.

* : FPT-120P-M03, FPT-120C-C02 *(Continued)*

Pin no. QFP*	Pin name	Circuit type	Function
62	INT2	B	DTP/External interrupt request input pin When DTP/external interrupts are enabled, these inputs may be used suddenly at any time; therefore, it is necessary to stop output by other functions on these pins, except when using them for output deliberately. When these pins are open in input mode, through current may leak in stop mode/reset mode, be sure to fix these pins to V_{CC}/V_{SS} level to use these pins in input mode.
	$\overline{ATG}$		10-bit A/D converter external trigger input pin When these pins are open in input mode, through current may leak in stop mode/reset mode, be sure to fix these pins to V_{CC}/V_{SS} level to use these pins in input mode.
64	PB0	C	General-purpose I/O port This function is valid when the UART0 (ch.0) serial data output specification is “disabled”.
	SOD0		UART0 (ch.0) serial data output This function is valid when the UART0 (ch.0) serial data output specification is “enabled”.
65	PB1	C	General-purpose I/O port This function is always valid.
	SID0		UART0 (ch.0) serial data input pin During UART0 (ch.0) input operations, this input may be used suddenly at any time; therefore, it is necessary to stop output by other functions on this pin, except when using it for output deliberately.
66	PB2	C	General-purpose output port This function is valid when the UART0 (ch.0) clock output specification is “disabled”.
	SCK0		UART0 (ch.0) clock output pin The clock output function is valid when the UART0 (ch.0) clock output specification is “enabled”. UART0 (ch.0) external clock input pin. This function is valid when the port is in input mode and the UART0 (ch.0) specification is external clock mode.
67	PB3	C	General-purpose I/O port This function is valid when the UART0 (ch.1) serial data output specification is “disabled”.
	SOD1		UART0 (ch.1) serial data output pin This function is valid when the UART0 (ch.1) serial data output specification is “enabled”.
68	PB4	C	General-purpose I/O port This function is always valid.
	SID1		UART0 (ch.1) serial data input pin During UART0 (ch.1) input operations, this input may be used suddenly at any time; therefore, it is necessary to stop output by other functions on this pin, except when using it for output deliberately.

* : FPT-120P-M03, FPT-120C-C02

(Continued)

Type	Circuit	Remarks
C		• CMOS-level output • CMOS-level hysteresis input with standby control
D		• CMOS-level input with no standby control Mask ROM products only: MD2: with pull-down resistor MD1: with pull-up resistor MD0: with pull-down resistor
		• CMOS-level input with no standby control MD2 of OTPROM products/EPROM products only
E		• CMOS-level hysteresis input with no standby control • With input analog filter (40 ns Typ.)

Note: The pull-up and pull-down resistors are always connected, regardless of the state.

(Continued)

■ HANDLING DEVICES

1. Preventing Latchup

CMOS ICs may cause latchup when a voltage higher than V_{CC} or lower than V_{SS} is applied to input or output pins other than medium-and high-voltage pins, or when a voltage exceeding the rating is applied between V_{CC} and V_{SS}.

If latch-up occurs, the power supply current increases rapidly, sometimes resulting in thermal breakdown of the device. Use meticulous care not to let any voltage exceed the maximum rating.
Also, take care to prevent the analog power supply (AV_{CC} and AVRH) and analog input from exceeding the digital power supply (V_{CC}) when the analog system power supply is turned on and off.

2. Treatment of Unused Input Pins

Leaving unused input pins open could cause malfunctions. They should be connected to a pull-up or pull-down resistor.

3. Treatment of Pins when A/D is not Used

Connect to be AV_{CC} = AVRH = V_{CC} and AV_{SS} = AVRL = V_{SS} even if the A/D converter is not in use.

4. Precautions when Using an External Input

To reset the internal circuit properly by the “L” level input to the $\overline{RST}$ pin, the “L” level input to the $\overline{RST}$ pin must be maintained for at least five machine cycles. Pay attention to it if the chip uses external clock input.

5. V_{CC} and V_{SS} Pins

Apply equal potential to the V_{CC} and V_{SS} pins.

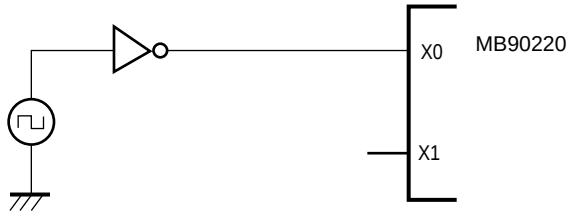
6. Supply Voltage Variation

The operation assurance range for the V_{CC} supply voltage is as given in the ratings. However, sudden changes in the supply voltage can cause misoperation, even if the voltage remains within the rated range. Therefore, it is important to supply a stable voltage to the IC. The recommended power supply control guidelines are that the commercial frequency (50 to 60 Hz) ripple variation (P-P value) on V_{CC} should be less than 10% of the standard V_{CC} value and that the transient rate of change during sudden changes, such as during power supply switching, should be less than 0.1 V/ms.

7. Notes on Using an External Clock

When using an external clock, drive the X0 pin as illustrated below. When an external clock is used, oscillation stabilization time is required even for power-on reset and wake-up from stop mode.

- **Use of External Clock**



Note: When using an external clock, be sure to input external clock more than 6 machine cycles after setting the $\overline{HST}$ pin to “L” to transfer to the hardware standby mode.

■ PERIPHERAL RESOURCES

1. Parallel Ports

The MB90220 series has 86 I/O pins and 16 open-drain I/O pins.

(1) Register Configuration

- **Port 0 to C Data Register (PDR0 to PDRC)**

Register name	Address
PDR1	000001H
PDR3	000003H
PDR5	000005H
PDR7	000007H
PDR9	000009H
PDRB	00000BH

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
	PD x 7	PD x 6	PD x 5	PD x 4	PD x 3	PD x 2	PD x 1	PD x 0	XXXXXXXXB
	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(PDR9 only: 11111111)
PDR7 only: (R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	

Register name	Address
PDR0	000000H
PDR2	000002H
PDR4	000004H
PDR6	000006H
PDR8	000008H
PDRA	00000AH
PDRC	00000CH

	bit7	bit6	bit5	bit 4	bit3	bit2	bit1	bit0	Initial value
	PD x 7	PD x 6	PD x 5	PD x 4	PD x 3	PD x 2	PD x 1	PD x 0	XXXXXXXXB
	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(PDR6 only: 11111111)

Note: There are no register bits for bits 7 and 6 of port C.

- **Port 0 to C Data Register (PDR0 to PDRC)**

Register name	Address
DDR1	000011H
DDR3	000013H
DDR5	000015H
DDR7	000017H
DDRB	00001BH

	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
	DD x 7	DD x 6	DD x 5	DD x 4	DD x 3	DD x 2	DD x 1	DD x 0	00000000B
	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(PDR7 only: 11111111)

Register name	Address
DDR0	000010H
DDR2	000012H
DDR4	000014H
DDR8	000018H
DDRA	00001AH
DDRC	00001CH

	bit7	bit6	bit5	bit 4	bit3	bit2	bit1	bit0	Initial value
	DD x 7	DD x 6	DD x 5	DD x 4	DD x 3	DD x 2	DD x 1	DD x 0	00000000B
	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	

Note: There are no register bits for bits 7 and 6 of port C.

- **Port 6, 9 Analog Input Enable Register (ADER0, ADER1)**

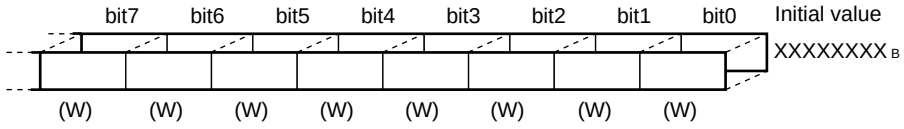
Register name	Address
ADER0	000016H

	bit7	bit6	bit5	bit 4	bit3	bit2	bit1	bit0	Initial value
	AE07	AE06	AE05	AE04	AE03	AE02	AE01	AE00	11111111B
	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	

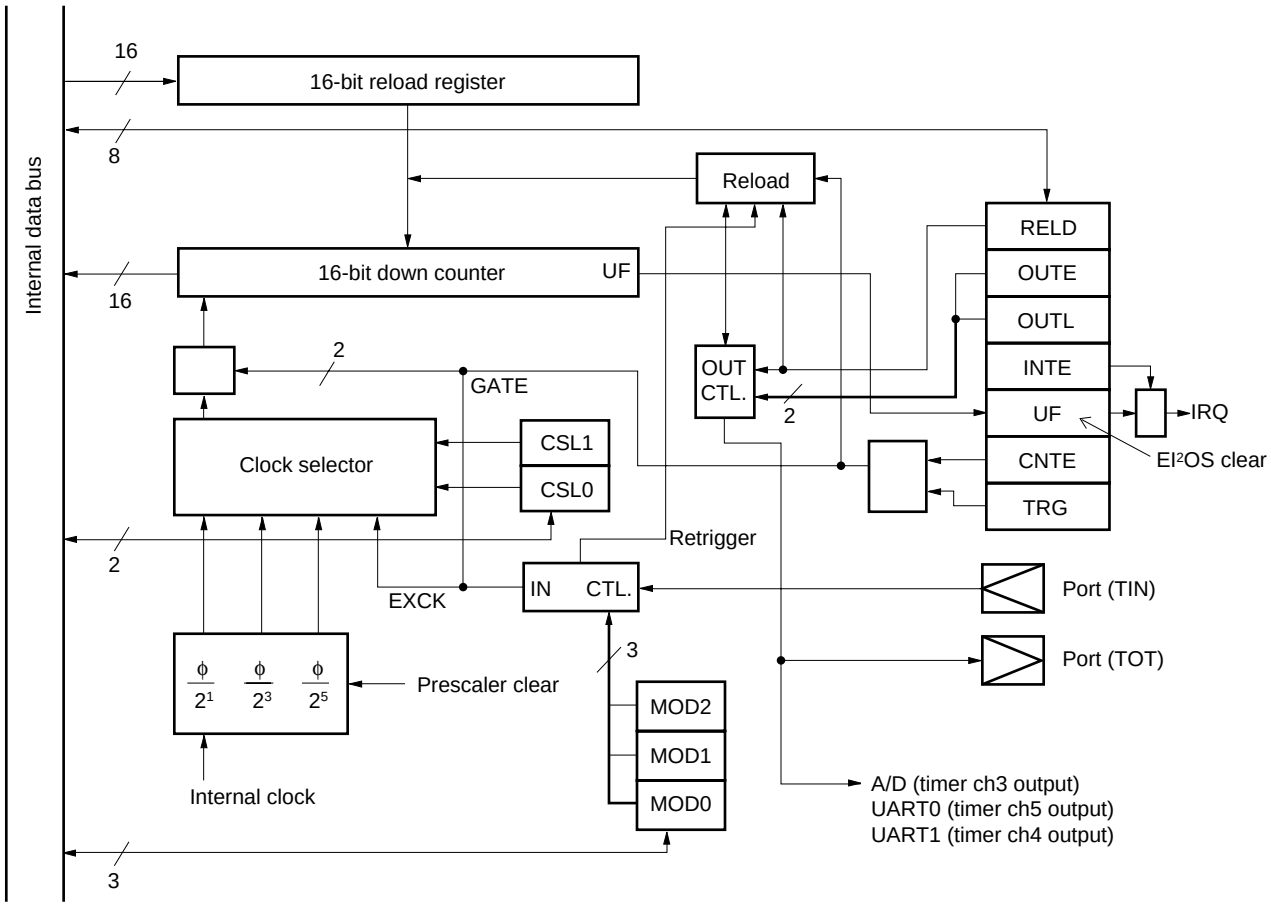
Register name	Address
ADER1	000019H

	bit7	bit6	bit5	bit 4	bit3	bit2	bit1	bit0	Initial value
	AE15	AE14	AE13	AE12	AE11	AE10	AE09	AE08	11111111B
	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	

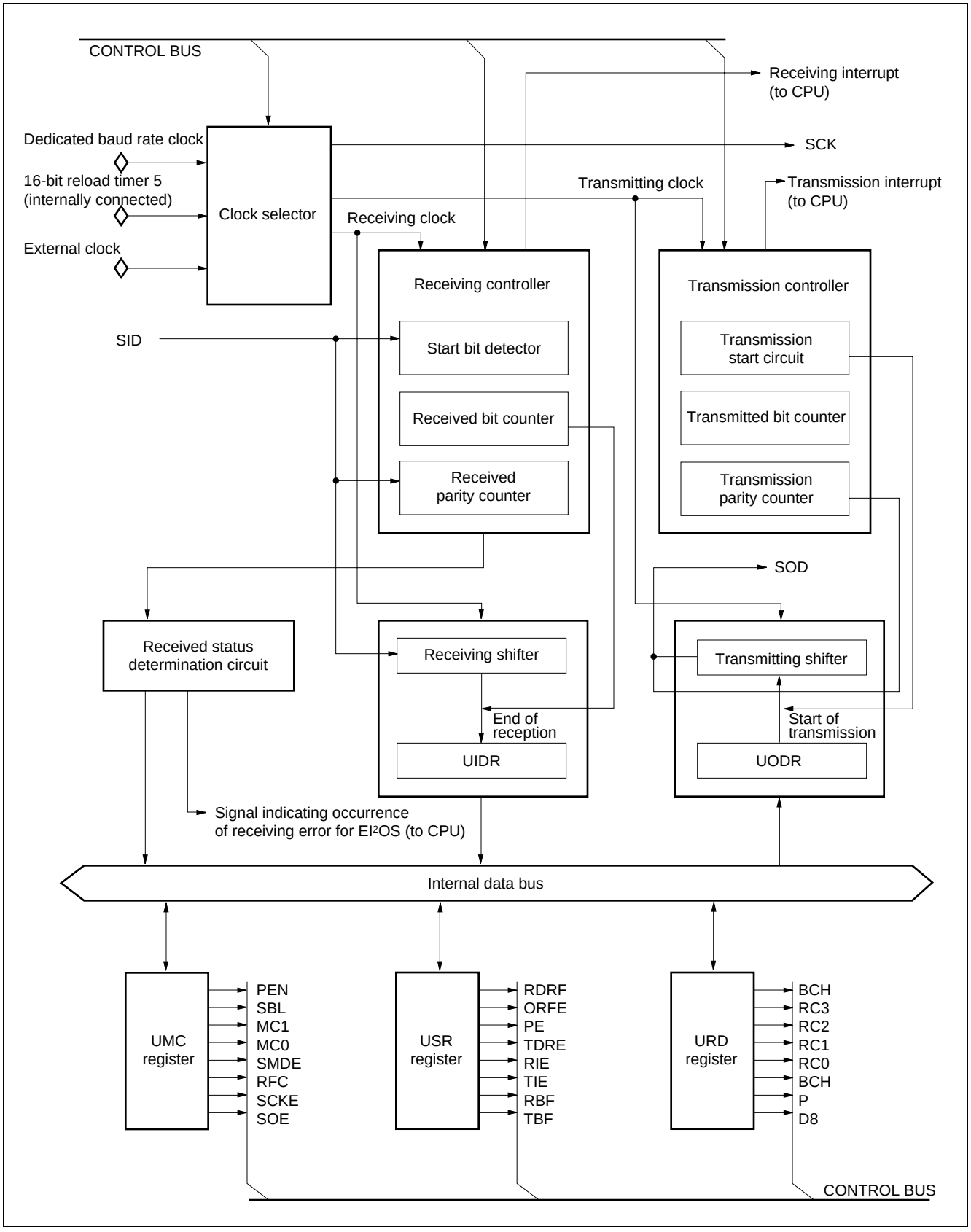
Register name	Address
TMRLR0	001F32 H
TMRLR1	001F36 H
TMRLR2	001F3A H
TMRLR3	001F3E H
TMRLR4	001F42 H
TMRLR5	001F46 H



(2) Block Diagram



(2) Block Diagram



5. 10-bit A/D Converter

The 10-bit A/D converter converts analog input voltage into a digital value. The features of this module are described below:

- Conversion time: 6.125 µs/channel (min.) (with machine clock running at 16 MHz)
- Uses RC-type sequential comparison and conversion method with built-in sample and hold circuit
- 10-bit resolution
- Analog input can be selected by software from among 16 channels
  Single-conversion mode: Selects and converts one channel.
  Scan conversion mode: Converts several consecutive channels (up to 16 can be programmed).
  One-shot mode: Converts the specified channel once and terminates.
  Continuous conversion mode: Repeatedly converts the specified channel.
  Stop conversion mode: Pauses after converting one channel and waits until the next startup (permits synchronization of start of conversion).
- When A/D conversion is completed, an "A/D conversion complete" interrupt request can be issued to the CPU. Because the generation of this interrupt can be used to start up the EI^{2}OS and transfer the A/D conversion results to memory, this function is suitable for continuous processing.
- Startup triggers can be selected from among software, an external trigger (falling edge), and a timer (rising edge).

(1) Register Configuration

- **A/D Channel Setting Register (ADCH)**

This register specfies the A/D converter conversion channel.

Register name	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
ADCH	000032$_H$	ANS3	ANS2	ANS1	ANS0	ANE3	ANE2	ANE1	ANE0	00000000$_B$
		(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	(R/W)	

- **A/D Mode Register (ADMD)**

This register specfies the A/D converter operation mode and the startup source.

Register name	Address	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8	Initial value
ADMD	000033$_H$	—	—	—	Reserved	MOD1	MOD0	STS1	STS0	- - - X0000$_B$
		(—)	(—)	(—)	(W)	(R/W)	(R/W)	(R/W)	(R/W)	

Note: Program "0" to bit 12 when write. Read value is indeterminated.

- **A/D Control Status Register (ADCS)**

This register is the A/D converter control and status register.

Register name	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
ADCS	000034$_H$	BUSY	INT	INTE	PAUS	—	—	STRT	Reserved	0000 - - 00$_B$
		(R/W)	(R/W)	(R/W)	(R/W)	(—)	(—)	(W)	(R/W)	

- **A/D Data Register (ADCD)**

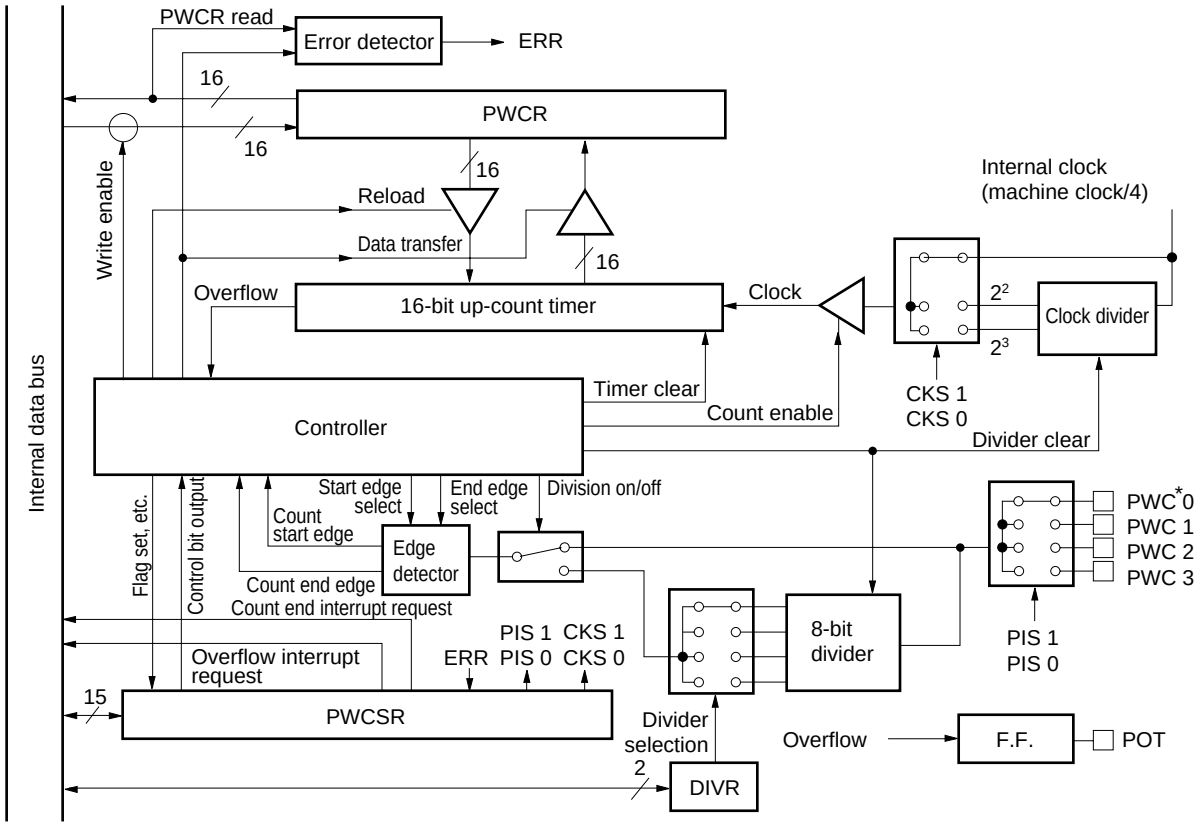
This register stores the A/D converter conversion data.

Register name	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
ADCD	000036$_H$	D7	D6	D5	D4	D3	D2	D1	D0	XXXXXXXX$_B$
		(R)	(R)	(R)	(R)	(R)	(R)	(R)	(R)	

- **PWC Division Ratio Control Register 0 to 3 (DIVR0 to DIVR3)**

Register name	Address	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Initial value
DIVR0	00007A H	—	—	—	—	—	—	MOD1	MOD0	------ 00B
DIVR1	00007C H									
DIVR2	00007E H									
DIVR3	000080 H	(—)	(—)	(—)	(—)	(—)	(—)	(R/W)	(R/W)	

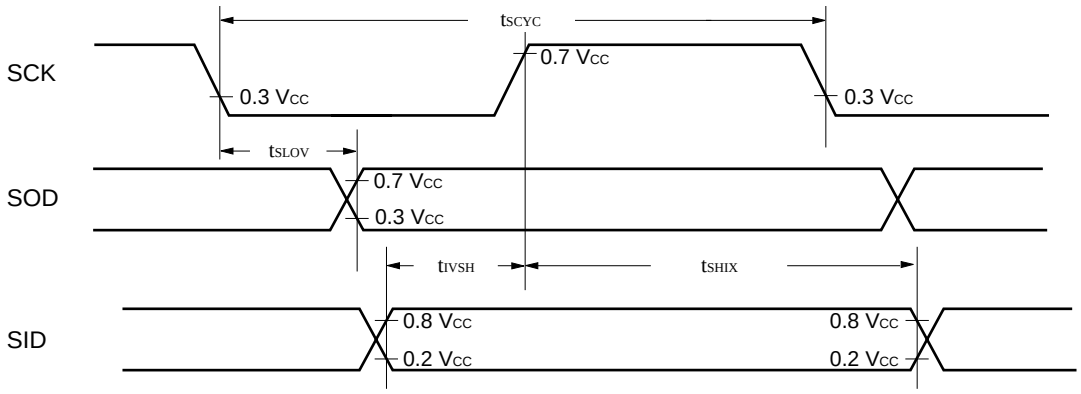
(2) Block Diagram



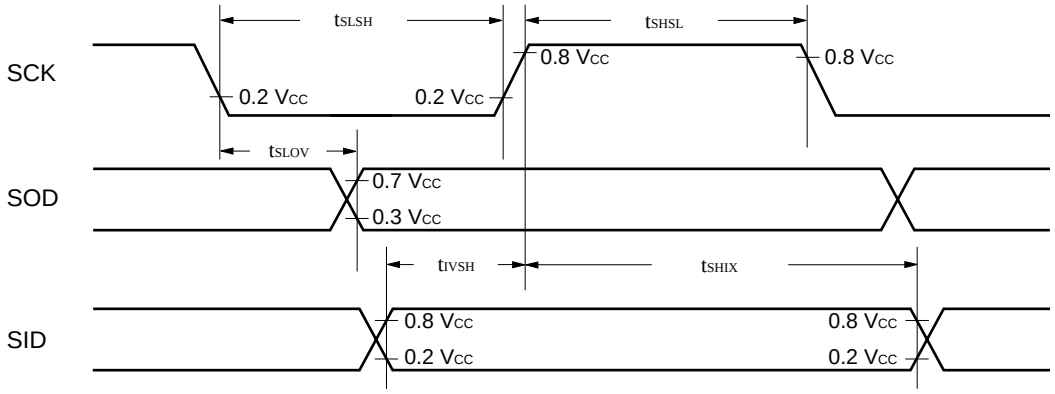
*: In the MB90220 series, only the module input PWC 0 of each channel is connected to the respective external pins.

Channel	POT pin
PWC ch. 0	PA 1/PWC 0/POT 0
PWC ch. 1	PA 2/PWC 1/POT 1/ASR 1
PWC ch. 2	PA 3/PWC 2/POT 2/ASR 2
PWC ch. 3	PA 4/PWC 3POT 3/ASR 3

- **Internal Shift Clock Mode**



- **External Shift Clock Input Mode**



■ INSTRUCTION SET (412 INSTRUCTIONS)

Table 1 Explanation of Items in Table of Instructions

Item	Explanation
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.
#	Indicates the number of bytes.
~	Indicates the number of cycles. See Table 4 for details about meanings of letters in items.
B	Indicates the correction value for calculating the number of actual cycles during execution of instruction. The number of actual cycles during execution of instruction is summed with the value in the “cycles” column.
Operation	Indicates operation of instruction.
LH	Indicates special operations involving the bits 15 through 08 of the accumulator. Z: Transfers “0”. X: Extends before transferring. —: Transfers nothing.
AH	Indicates special operations involving the high-order 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 extending AL.
I	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.
S	
T	
N	
Z	
V	
C	
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: Cannot be used for addresses that have different meanings depending on whether they are read or written.

(Continued)

Mnemonic	#	cycles	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
XCH A, ear	2	3	0	byte (A) ↔ (ear)	Z	–	–	–	–	–	–	–	–	–
XCH A, eam	2+	3+ (a)	2× (b)	byte (A) ↔ (eam)	Z	–	–	–	–	–	–	–	–	–
XCH Ri, ear	2	4	0	byte (Ri) ↔ (ear)	–	–	–	–	–	–	–	–	–	–
XCH Ri, eam	2+	5+ (a)	2× (b)	byte (Ri) ↔ (eam)	–	–	–	–	–	–	–	–	–	–

For an explanation of “(a)” and “(b)”, refer to Table 4, “Number of Execution Cycles for Each Form 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 (Word/Long Word) [11 Insturctions]

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

For an explanation of “(b)” and “(c)”, refer to Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

*1: 3 when dividing into zero, 8 or 18 when an overflow occurs, and 18 normally.
*2: 3 when dividing into zero, 10 or 21 when an overflow occurs, and 22 normally.
*3: 4 + (a) when dividing into zero, 11 + (a) or 22 + (a) when an overflow occurs, and 23 + (a) normally.
*4: When the dividend is positive: 4 when dividing into zero, 10 or 29 when an overflow occurs, and 30 normally.
When the dividend is negative: 4 when dividing into zero, 11 or 30 when an overflow occurs, and 31 normally.
*5: When the dividend is positive: 4 + (a) when dividing into zero, 11 + (a) or 30 + (a) when an overflow occurs, and 31 + (a) normally.
When the dividend is negative: 4 + (a) when dividing into zero, 12 + (a) or 31 + (a) when an overflow occurs, and 32 + (a) normally.
*6: (b) when dividing into zero or when an overflow occurs, and 2 × (b) normally.
*7: (c) when dividing into zero or when an overflow occurs, and 2 × (c) normally.
*8: 3 when byte (AH) is zero, 12 when the result is positive, and 13 when the result is negative.
*9: 3 when byte (ear) is zero, 12 when the result is positive, and 13 when the result is negative.
*10: 4 + (a) when byte (eam) is zero, 13 + (a) when the result is positive, and 14 + (a) when the result is negative.
*11: 3 when word (AH) is zero, 12 when the result is positive, and 13 when the result is negative.
*12: 3 when word (ear) is zero, 16 when the result is positive, and 19 when the result is negative.
*13: 4 + (a) when word (eam) is zero, 17 + (a) when the result is positive, and 20 + (a) when the result is negative.

Note: Which of the two values given for the number of execution cycles applies when an overflow error occurs in a DIV or DIVW instruction depends on whether the overflow was detected before or after the operation.

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

Mnemonic	#	cycles	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
ANDL A, ear	2	5	0	long (A) ← (A) and (ear)	–	–	–	–	–	*	*	R	–	–
ANDL A, eam	2+	6+ (a)	(d)	long (A) ← (A) and (eam)	–	–	–	–	–	*	*	R	–	–
ORL A, ear	2	5	0	long (A) ← (A) or (ear)	–	–	–	–	–	*	*	R	–	–
ORL A, eam	2+	6+ (a)	(d)	long (A) ← (A) or (eam)	–	–	–	–	–	*	*	R	–	–
XORL A, ear	2	5	0	long (A) ← (A) xor (ear)	–	–	–	–	–	*	*	R	–	–
XORL A, eam	2+	6+ (a)	(d)	long (A) ← (A) xor (eam)	–	–	–	–	–	*	*	R	–	–

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

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

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

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

Table 17 Absolute Value Instructions (Byte/Word/Long Word) [3 Insturctions]

Mnemonic	#	cycles	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
ABS A	2	2	0	byte (A) ← absolute value (A)	Z	–	–	–	–	*	*	*	–	–
ABSW A	2	2	0	word (A) ← absolute value (A)	–	–	–	–	–	*	*	*	–	–
ABSL A	2	4	0	long (A) ← absolute value (A)	–	–	–	–	–	*	*	*	–	–

Table 18 Normalize Instructions (Long Word) [1 Instruction]

Mnemonic	#	cycles	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
NRML A, R0	2	*	0	long (A) ← Shifts to the position at which “1” was set first byte (R0) ← current shift count	–	–	–	–	*	–	–	–	–	–

* : 5 when the contents of the accumulator are all zeroes, 5 + (R0) in all other cases.

Table 19 Shift Instructions (Byte/Word/Long Word) [27 Instructions]

Mnemonic	#	cycles	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
RORC A	2	2	0	byte (A) ← Right rotation with carry	–	–	–	–	–	*	*	–	*	–
ROLC A	2	2	0	byte (A) ← Left rotation with carry	–	–	–	–	–	*	*	–	*	–
RORC ear	2	2	0	byte (ear) ← Right rotation with carry	–	–	–	–	–	*	*	–	*	*
RORC eam	2+	3+ (a)	2× (b)	byte (eam) ← Right rotation with carry	–	–	–	–	–	*	*	–	*	*
ROLC ear	2	2	0	byte (ear) ← Left rotation with carry	–	–	–	–	–	*	*	–	*	*
ROLC eam	2+	3+ (a)	2× (b)	byte (eam) ← Left rotation with carry	–	–	–	–	–	*	*	–	*	*
ASR A, R0	2	*1	0	byte (A) ← Arithmetic right barrel shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSR A, R0	2	*1	0	byte (A) ← Logical right barrel shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSL A, R0	2	*1	0	byte (A) ← Logical left barrel shift (A, R0)	–	–	–	–	–	*	*	–	*	–
ASR A, #imm8	3	*3	0	byte (A) ← Arithmetic right barrel shift (A, imm8)	–	–	–	–	*	*	*	–	*	–
LSR A, #imm8	3	*3	0	byte (A) ← Logical right barrel shift (A, imm8)	–	–	–	–	*	*	*	–	*	–
LSL A, #imm8	3	*3	0	byte (A) ← Logical left barrel shift (A, imm8)	–	–	–	–	–	*	*	–	*	–
ASRW A	1	2	0	word (A) ← Arithmetic right shift (A, 1 bit)	–	–	–	–	*	*	*	–	*	–
LSRW A/SHRW A	1	2	0	word (A) ← Logical right shift (A, 1 bit)	–	–	–	–	*	R	*	–	*	–
LSLW A/SHLW A	1	2	0	word (A) ← Logical left shift (A, 1 bit)	–	–	–	–	–	*	*	–	*	–
ASRW A, R0	2	*1	0	word (A) ← Arithmetic right barrel shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSRW A, R0	2	*1	0	word (A) ← Logical right barrel shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSLW A, R0	2	*1	0	word (A) ← Logical left barrel shift (A, R0)	–	–	–	–	–	*	*	–	*	–
ASRW A, #imm8	3	*3	0	word (A) ← Arithmetic right barrel shift (A, imm8)	–	–	–	–	*	*	*	–	*	–
LSRW A, #imm8	3	*3	0	word (A) ← Logical right barrel shift (A, imm8)	–	–	–	–	*	*	*	–	*	–
LSLW A, #imm8	3	*3	0	word (A) ← Logical left barrel shift (A, imm8)	–	–	–	–	–	*	*	–	*	–
ASRL A, R0	2	*2	0	long (A) ← Arithmetic right shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSRL A, R0	2	*2	0	long (A) ← Logical right barrel shift (A, R0)	–	–	–	–	*	*	*	–	*	–
LSLL A, R0	2	*2	0	long (A) ← Logical left barrel shift (A, R0)	–	–	–	–	–	*	*	–	*	–
ASRL A, #imm8	3	*4	0	long (A) ← Arithmetic right shift (A, imm8)	–	–	–	–	*	*	*	–	*	–
LSRL A, #imm8	3	*4	0	long (A) ← Logical right barrel shift (A, imm8)	–	–	–	–	*	*	*	–	*	–
LSLL A, #imm8	3	*4	0	long (A) ← Logical left barrel shift (A, imm8)	–	–	–	–	–	*	*	–	*	–

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

*1: 3 when R0 is 0, 3 + (R0) in all other cases.
*2: 3 when R0 is 0, 4 + (R0) in all other cases.
*3: 3 when imm8 is 0, 3 + (imm8) in all other cases.
*4: 3 when imm8 is 0, 4 + (imm8) in all other cases.

Table 20 Branch 1 Instructions [31 Instructions]

Mnemonic	#	cycles	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
BZ/BEQ rel	2	*1	0	Branch when (Z) = 1	–	–	–	–	–	–	–	–	–	–
BNZ/BNE rel	2	*1	0	Branch when (Z) = 0	–	–	–	–	–	–	–	–	–	–
BC/BLO rel	2	*1	0	Branch when (C) = 1	–	–	–	–	–	–	–	–	–	–
BNC/BHS rel	2	*1	0	Branch when (C) = 0	–	–	–	–	–	–	–	–	–	–
BN rel	2	*1	0	Branch when (N) = 1	–	–	–	–	–	–	–	–	–	–
BP rel	2	*1	0	Branch when (N) = 0	–	–	–	–	–	–	–	–	–	–
BV rel	2	*1	0	Branch when (V) = 1	–	–	–	–	–	–	–	–	–	–
BNV rel	2	*1	0	Branch when (V) = 0	–	–	–	–	–	–	–	–	–	–
BT rel	2	*1	0	Branch when (T) = 1	–	–	–	–	–	–	–	–	–	–
BNT rel	2	*1	0	Branch when (T) = 0	–	–	–	–	–	–	–	–	–	–
BLT rel	2	*1	0	Branch when (V) xor (N) = 1	–	–	–	–	–	–	–	–	–	–
BGE rel	2	*1	0	Branch when (V) xor (N) = 0	–	–	–	–	–	–	–	–	–	–
BLE rel	2	*1	0	((V) xor (N)) or (Z) = 1	–	–	–	–	–	–	–	–	–	–
BGT rel	2	*1	0	((V) xor (N)) or (Z) = 0	–	–	–	–	–	–	–	–	–	–
BLS rel	2	*1	0	Branch when (C) or (Z) = 1	–	–	–	–	–	–	–	–	–	–
BHI rel	2	*1	0	Branch when (C) or (Z) = 0	–	–	–	–	–	–	–	–	–	–
BRA rel	2	*1	0	Branch unconditionally	–	–	–	–	–	–	–	–	–	–
JMP @A	1	2	0	word (PC) ← (A)	–	–	–	–	–	–	–	–	–	–
JMP addr16	3	2	0	word (PC) ← addr16	–	–	–	–	–	–	–	–	–	–
JMP @ear	2	3	0	word (PC) ← (ear)	–	–	–	–	–	–	–	–	–	–
JMP @eam	2+	4+ (a)	(c)	word (PC) ← (eam)	–	–	–	–	–	–	–	–	–	–
JMPP @ear *3	2	3	0	word (PC) ← (ear), (PCB) ← (ear +2)	–	–	–	–	–	–	–	–	–	–
JMPP @eam *3	2+	4+ (a)	(d)	word (PC) ← (eam), (PCB) ← (eam +2)	–	–	–	–	–	–	–	–	–	–
JMPP addr24	4	3	0	word (PC) ← ad24 0 to 15 (PCB) ← ad24 16 to 23	–	–	–	–	–	–	–	–	–	–
CALL @ear *4	2	4	(c)	word (PC) ← (ear)	–	–	–	–	–	–	–	–	–	–
CALL @eam *4	2+	5+ (a)	2× (c)	word (PC) ← (eam)	–	–	–	–	–	–	–	–	–	–
CALL addr16 *5	3	5	(c)	word (PC) ← addr16	–	–	–	–	–	–	–	–	–	–
CALLV #vct4 *5	1	5	2× (c)	Vector call linstruction	–	–	–	–	–	–	–	–	–	–
CALLP @ear *6	2	7	2× (c)	word (PC) ← (ear) 0 to 15, (PCB) ← (ear) 16 to 23	–	–	–	–	–	–	–	–	–	–
CALLP @eam *6	2+	8+ (a)	*2	word (PC) ← (eam) 0 to 15, (PCB) ← (eam) 16 to 23	–	–	–	–	–	–	–	–	–	–
CALLP addr24 *7	4	7	2× (c)	word (PC) ← addr 0 to 15, (PCB) ← addr 16 to 23	–	–	–	–	–	–	–	–	–	–

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

*1: 3 when branching, 2 when not branching.
*2: 3 × (c) + (b)
*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) branch address.
*7: Save (long word) to stack.

Table 21 Branch 2 Instructions [20 Instructions]

Mnemonic	#	cycles	B	Operation	LH	AH	I	S	T	N	Z	V	C	RMW
CBNE A, #imm8, rel	3	*1	0	Branch when byte (A) ≠ imm8	–	–	–	–	–	*	*	*	*	–
CWBNE A, #imm16, rel	4	*1	0	Branch when byte (A) ≠ imm16	–	–	–	–	–	*	*	*	*	–
CBNE ear, #imm8, rel	4	*1 *3	0	Branch when byte (ear) ≠ imm8	–	–	–	–	–	*	*	*	*	–
CBNE eam, #imm8, rel	4+	*1	(b)	Branch when byte (eam) ≠ imm8	–	–	–	–	–	*	*	*	*	–
CWBNE ear, #imm16, rel	5	*3	0	Branch when word (ear) ≠ imm16	–	–	–	–	–	*	*	*	*	–
CWBNE eam, #imm16, rel	5+	*2	(c)	Branch when word (eam) ≠ imm16	–	–	–	–	–	*	*	*	*	–
DBNZ ear, rel	3	*4	0	Branch when byte (ear) = (ear) – 1, and (ear) ≠ 0	–	–	–	–	–	*	*	*	–	–
DBNZ eam, rel	3+	*2	2× (b)	Branch when byte (ear) = (eam) – 1, and (eam) ≠ 0	–	–	–	–	–	*	*	*	–	*
DWBNZ ear, rel	3	*4	0	Branch when word (ear) = (ear) – 1, and (ear) ≠ 0	–	–	–	–	–	*	*	*	–	–
DWBNZ eam, rel	3+	14 12	2× (c)	Branch when word (eam) = (eam) – 1, and (eam) ≠ 0	–	–	–	–	–	*	*	*	–	*
INT #vct8	2	13	8× (c)	Software interrupt	–	–	R	S	–	–	–	–	–	–
INT addr16	3	14	6× (c)	Software interrupt	–	–	R	S	–	–	–	–	–	–
INTP addr24	4	9	6× (c)	Software interrupt	–	–	R	S	–	–	–	–	–	–
INT9	1	11	8× (c)	Software interrupt	–	–	R	S	–	–	–	–	–	–
RETI	1		6× (c)	Return from interrupt	–	–	*	*	*	*	*	*	*	–
RETIQ *6	2	6	*5	Return from interrupt	–	–	*	*	*	*	*	*	*	–
LINK #imm8	2	5	(c)	At constant entry, save old frame pointer to stack, set new frame pointer, and allocate local pointer area	–	–	–	–	–	–	–	–	–	–
UNLINK	1	4 5	(c)	At constant entry, retrieve old frame pointer from stack.	–	–	–	–	–	–	–	–	–	–
RET *7	1		(c)	Return from subroutine	–	–	–	–	–	–	–	–	–	–
RETP *8	1		(d)	Return from subroutine	–	–	–	–	–	–	–	–	–	–

For an explanation of “(b)”, “(c)” and “(d)”, refer to Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

*1: 4 when branching, 3 when not branching
*2: 5 when branching, 4 when not branching
*3: 5 + (a) when branching, 4 + (a) when not branching
*4: 6 + (a) when branching, 5 + (a) when not branching
*5: 3 × (b) + 2 × (c) when an interrupt request is generated, 6 × (c) when returning from the interrupt.
*6: High-speed interrupt return instruction. When an interrupt request is detected during this instruction, the instruction branches to the interrupt vector without performing stack operations when the interrupt is generated.
*7: Return from stack (word)
*8: Return from stack (long word)

Table 23 Bit Manipulation Instructions [21 Instructions]

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

For an explanation of “(b)”, refer to Table 5, “Correction Values for Number of Cycles Used to Calculate Number of Actual Cycles.”

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