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	Obsolete
Core Processor	H8/300H
Core Size	16-Bit
Speed	4MHz
Connectivity	I²C, IrDA, SCI, SSU
Peripherals	POR, PWM, WDT
Number of I/O	13
Program Memory Size	16KB (16K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	1K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 3.6V
Data Converters	A/D 6x10b
Oscillator Type	Internal
Operating Temperature	-20°C ~ 75°C (TA)
Mounting Type	Surface Mount
Package / Case	32-VFQFN
Supplier Device Package	32-VQFN (5x6)
Purchase URL	https://www.e-xfl.com/product-detail/renesas-electronics-america/df38602rft4v

2.2 Register Configuration

The H8/300H CPU has the internal registers shown in figure 2.2. There are two types of registers; general registers and control registers. The control registers are a 24-bit program counter (PC), and an 8-bit condition-code register (CCR).

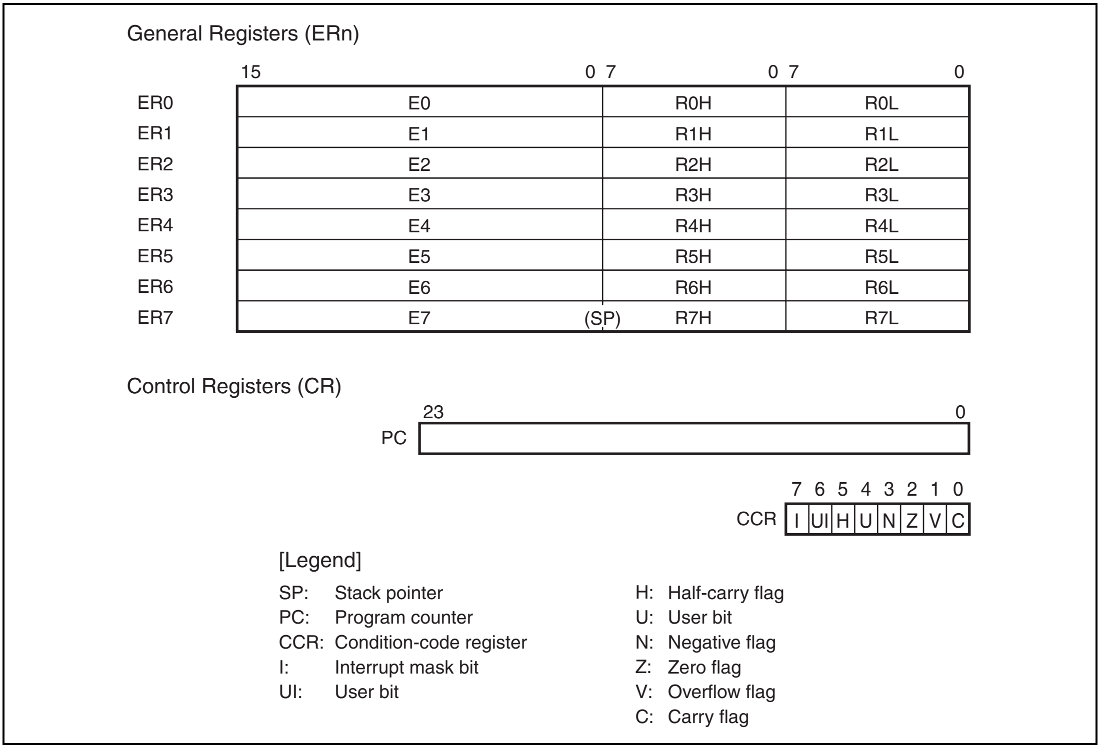


Figure 2.2 CPU Registers

2.3.2 Memory Data Formats

Figure 2.6 shows the data formats in memory. The H8/300H CPU can access word data and longword data in memory, however word or longword data must begin at an even address. If an attempt is made to access word or longword data at an odd address, an address error does not occur, however the least significant bit of the address is regarded as 0, so access begins the preceding address. This also applies to instruction fetches.

When ER7 (SP) is used as an address register to access the stack area, the operand size should be word or longword.

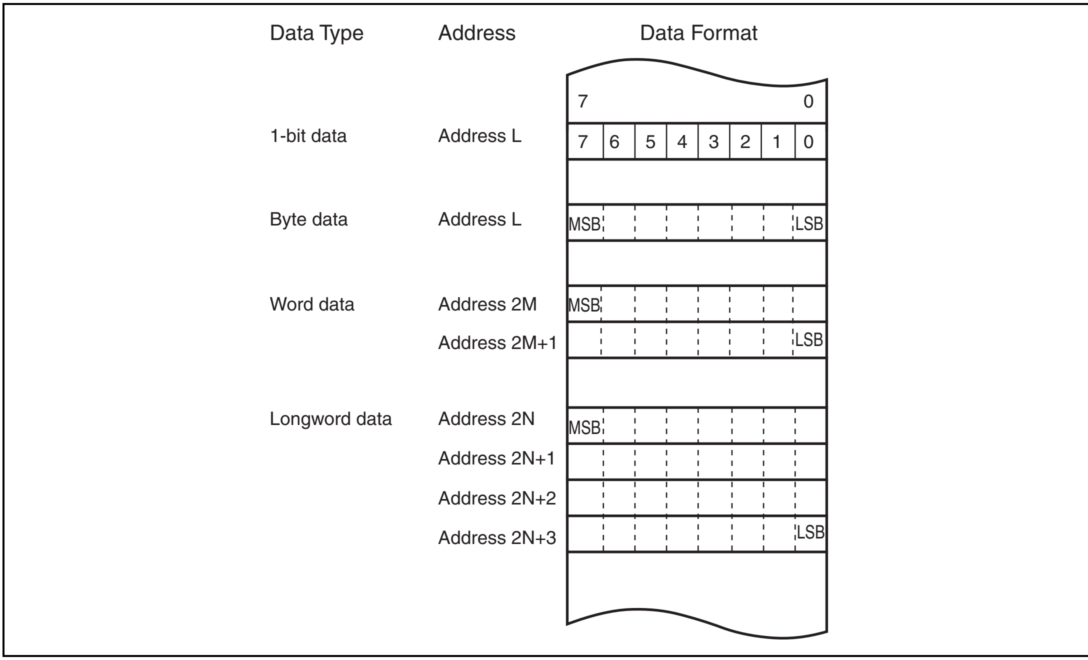


Figure 2.6 Memory Data Formats

Source Origin	Exception Sources	Vector Number	Vector Address	Priority
WDT	WDT overflow (interval timer)	31	H'003E to H'003F	High
Asynchronous event counter	Asynchronous event counter overflow	32	H'0040 to H'0041	↑
Timer B1	Overflow	33	H'0042 to H'0043	
Synchronous serial communication unit (SSU)/ IIC2*	Overrun error (SSU) Transmit data empty (SSU) Transmit end (SSU) Receive data full (SSU) Conflict error (SSU)/ Transmit data empty (IIC2) Transmit end (IIC2) Receive data full (IIC2) NACK detection (IIC2) Arbitration (IIC2) Overrun error (IIC2)	34	H'0044 to H'0045	
Timer W	Input capture A/compare match A Input capture B/compare match B Input capture C/compare match C Input capture D/compare match D Overflow	35	H'0046 to H'0047	
—	Reserved for system use	36	H'0048 to H'0049	
SCI3	Transmit end Transmit data empty Receive data full Overrun error Framing error Parity error	37	H'004A to H'004B	
A/D converter	A/D conversion end	38	H'004C to H'004D	↓
—	Reserved for system use	39	H'004E to H'004F	Low

Note: * The SSU and IIC share the same vector address. When using the IIC, shift the SSU to standby mode using CKSTPR2.

3.4.3 Interrupt Enable Register 2 (IENR2)

IENR2 enables the A/D converter, timer B1, and asynchronous event counter interrupts.

Bit	Bit Name	Initial Value	R/W	Description
7	—	0	—	Reserved The write value should always be 0.
6	IENAD	0	R/W	A/D Converter Interrupt Request Enable The A/D converter interrupt request is enabled when this bit is set to 1.
5 to 3	—	All 0	—	Reserved The write value should always be 0.
2	IENTB1	0	R/W	Timer B1 Interrupt Request Enable The timer B1 interrupt request is enabled when this bit is set to 1.
1	—	0	—	Reserved The write value should always be 0.
0	IENEC	0	R/W	Asynchronous Event Counter Interrupt Request Enable The asynchronous event counter interrupt request is enabled when this bit is set to 1.

5.2 Mode Transitions and States of LSI

Figure 5.1 shows the possible transitions among these operating modes. A transition is made from the program execution state to the program halt state of the program by executing a SLEEP instruction. Interrupts allow for returning from the program halt state to the program execution state of the program. A direct transition between active mode and subactive mode, which are both program execution states, can be made without halting the program. The operating frequency can also be changed in the same modes by making a transition directly from active mode to active mode, and from subactive mode to subactive mode. $\overline{\text{RES}}$ input enables transitions from a mode to the reset state. Table 5.2 shows the transition conditions of each mode after the SLEEP instruction is executed and a mode to return by an interrupt. Table 5.3 shows the internal states of the LSI in each mode.

6.4 Flash Memory Programming/Erasure

A software method using the CPU is employed to program and erase flash memory in the on-board programming modes. Depending on the FLMCR1 setting, the flash memory operates in one of the following four modes: Programming mode, programming-verifying mode, erasing mode, and erasing-verifying mode. The programming control program in boot mode and the user programming/erasing control program in user program mode use these operating modes in combination to perform programming/erasure. Flash memory programming and erasing should be performed in accordance with the descriptions in section 6.4.1, Programming/Programming-Verifying and section 6.4.2, Erasing/Erasing-Verifying, respectively.

6.4.1 Programming/Programming-Verifying

When writing data or programs to the flash memory, the programming/programming-verifying flowchart shown in figure 6.3 should be followed. Performing programming operations according to this flowchart will enable data or programs to be written to the flash memory without subjecting the chip to voltage stress or sacrificing program data reliability.

1. Programming must be performed on an erased area. Do not reprogram an address to which data has already been programmed.
2. Programming should be carried out 128 bytes at a time. A 128-byte data transfer must be performed even if programming fewer than 128 bytes. In this case, the remaining area must be filled with H'FF.
3. Prepare the following data storage areas in RAM: A 128-byte programming data area, a 128-byte reprogramming data area, and a 128-byte additional-programming data area. Perform reprogramming data computation according to table 6.4, and additional programming data computation according to table 6.5.
4. Consecutively transfer 128 bytes of data in bytes from the reprogramming data area or additional-programming data area to the flash memory. The programming address and 128-byte data are latched in the flash memory. The lower eight bits of the start address in the flash memory destination area must be H'00 or H'80.
5. The time during which the P bit is set to 1 is the programming time. Table 6.6 shows the allowable programming times.
6. The watchdog timer (WDT) is set to prevent overprogramming due to program runaway, etc. An overflow cycle of approximately 6.6 ms is allowed.
7. For a dummy write to a verifying address, write 1-byte of data H'FF to an address whose lower two bits are B'00. Verifying data can be read in words or in longwords from the address to which a dummy write was performed.

Section 7 RAM

This LSI has an on-chip high-speed static RAM. The RAM is connected to the CPU by a 16-bit data bus, enabling two-state access by the CPU to both byte data and word data.

Product Classification		RAM Size	RAM Address
Flash memory version	H8/38602RF	1 Kbyte	H'FB80 to H'FF7F
Masked ROM version	H8/38602R	1 Kbyte	H'FB80 to H'FF7F
	H8/38600R	512 bytes	H'FD80 to H'FF7F

8.2.6 Input Pull-Up MOS

Port 3 has an on-chip input pull-up MOS function that can be controlled by software. When a PCR3 bit is cleared to 0, setting the corresponding PUCR3 bit to 1 turns on the input pull-up MOS for that pin. The input pull-up MOS function is in the off state after a reset.

(n = 2 to 0)

PCR3n	0		1
PUCR3n	0	1	x
Input Pull-Up MOS	Off	On	Off

[Legend] x: Don't care.

8.3 Port 8

Port 8 is an I/O port also functioning as a timer W I/O pin. Figure 8.3 shows its pin configuration.

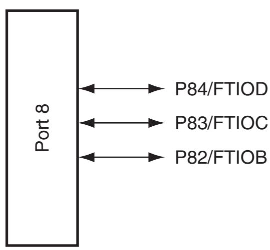


Figure 8.3 Port 8 Pin Configuration

Port 8 has the following registers.

- Port data register 8 (PDR8)
- Port control register 8 (PCR8)
- Port pull-up control register 8 (PUCR8)

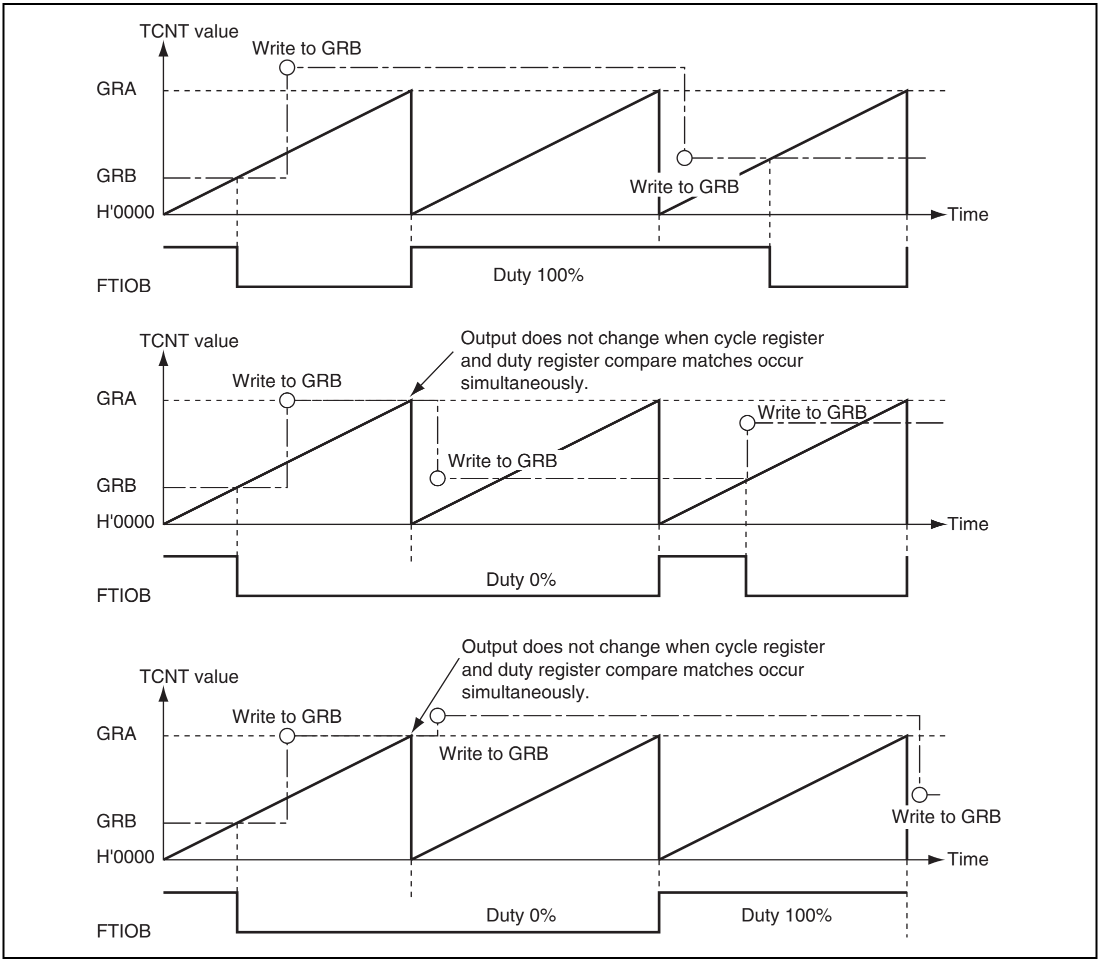


Figure 10.13 PWM Mode Example
(TOB, TOC, and TOD = 1: initial output values are set to 1)

11.4.3 Data Reading Procedure

When the seconds, minutes, hours, or day-of-week datum is updated while time data is being read, the data obtained may not be correct, and so the time data must be read again. Figure 11.4 shows an example in which correct data is not obtained. In this example, since only RSECDR is read after data update, about 1-minute inconsistency occurs.

To avoid reading in this timing, the following processing must be performed.

1. Check the setting of the BSY bit, and when the BSY bit changes from 1 to 0, read from the second, minute, hour, and day-of-week registers. When about 62.5 ms is passed after the BSY bit is set to 1, the registers are updated, and the BSY bit is cleared to 0.
2. When INT in RTCCR1 is cleared to 0 and an interrupt is used, read from the second, minute, hour, and day-of-week registers after the relevant flag in RTCFLG is set to 1 and the BSY bit is confirmed to be 0.

   When INT in RTCCR1 is set to 1 and an interrupt is used, read from the second, minute, hour, and day-of-week registers after the relevant flag in RTCFLG is set to 1.
3. Read from the second, minute, hour, and day-of-week registers twice in a row, and if there is no change in the read data, the read data is used.

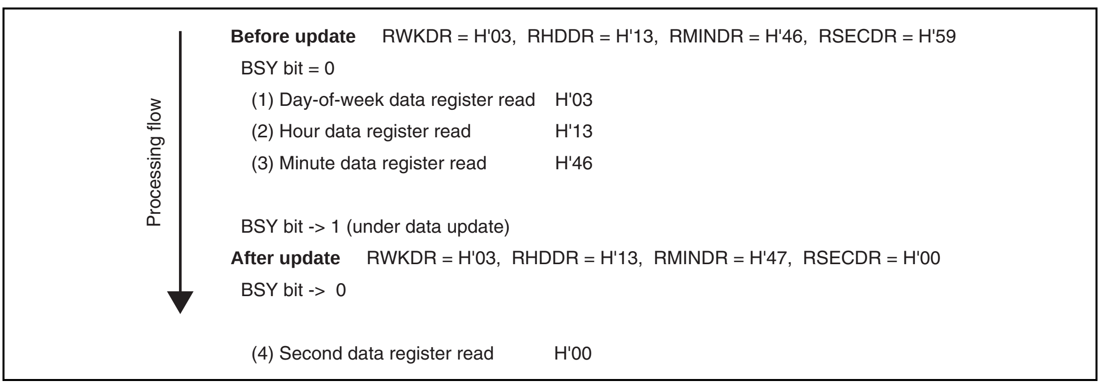


Figure 11.4 Example: Reading of Inaccurate Time Data

Section 14 Serial Communication Interface 3 (SCI3, IrDA)

The serial communication interface 3 (SCI3) can handle both asynchronous and clock synchronous serial communication. In the asynchronous method, serial data communication can be carried out using standard asynchronous communication chips such as a Universal Asynchronous Receiver/Transmitter (UART) or an Asynchronous Communication Interface Adapter (ACIA).

The SCI3 can transmit and receive IrDA communication waveforms based on the Infrared Data Association (IrDA) standard version 1.0.

14.1 Features

- Choice of asynchronous or clock synchronous serial communication mode
- Full-duplex communication capability

  The transmitter and receiver are mutually independent, enabling transmission and reception to be executed simultaneously.

  Double-buffering is used in both the transmitter and the receiver, enabling continuous transmission and continuous reception of serial data.
- On-chip baud rate generator allows any bit rate to be selected
- On-chip baud rate generator, internal clock, or external clock can be selected as a transfer clock source.
- Six interrupt sources

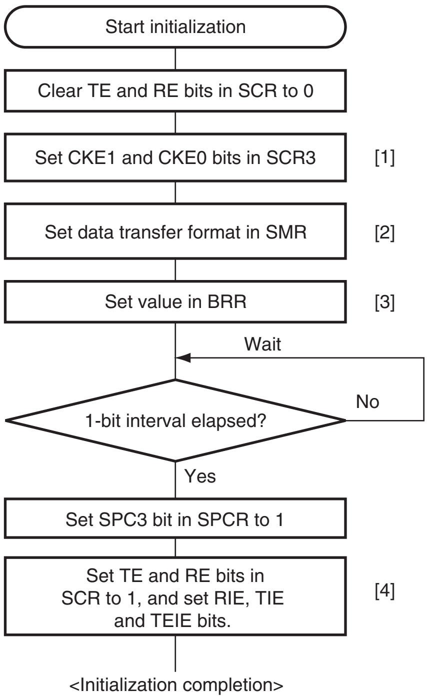
  Transmit-end, transmit-data-empty, receive-data-full, overrun error, framing error, and parity error.
- Use of module standby mode enables this module to be placed in standby mode independently when not used. (The SCI3 is halted as the initial value. For details, refer to section 5.4, Module Standby Function.)

Asynchronous mode

- Data length: 7, 8, or 5 bits
- Stop bit length: 1 or 2 bits
- Parity: Even, odd, or none
- Receive error detection: Parity, overrun, and framing errors
- Break detection: Break can be detected by reading the RXD3 pin level directly in the case of a framing error

14.4.2 SCI3 Initialization

Follow the flowchart as shown in figure 14.4 to initialize the SCI3. When the TE bit is cleared to 0, the TDRE flag is set to 1. Note that clearing the RE bit to 0 does not initialize the contents of the RDRF, PER, FER, and OER flags, or the contents of RDR. When the external clock is used in asynchronous mode, the clock must be supplied even during initialization. When the external clock is used in clock synchronous mode, the clock must not be supplied during initialization.



[1] Set the clock selection in SCR. Be sure to clear bits RIE, TIE, TEIE, and MPIE, and bits TE and RE, to 0.

When the clock output is selected in asynchronous mode, clock is output immediately after CKE1 and CKE0 settings are made. When the clock output is selected at reception in clock synchronous mode, clock is output immediately after CKE1, CKE0, and RE are set to 1.

[2] Set the data transfer format in SMR.

[3] Write a value corresponding to the bit rate to BRR. Not necessary if an external clock is used.

[4] Wait at least one bit interval, then set the TE bit or RE bit in SCR to 1. Setting bits TE and RE enables the TXD3 and RXD3 pins to be used. Also set the RIE, TIE, and TEIE bits, depending on whether interrupts are required. In asynchronous mode, the bits are marked at transmission and idled at reception to wait for the start bit.

Figure 14.4 Sample SCI3 Initialization Flowchart

15.4.2 Relationship between Clock Polarity and Phase, and Data

Relationship between clock polarity and phase, and transfer data changes according to a combination of the SSUMS bit in SSCRL and the CPOS and CPHS bits in SSMR. Figure 15.2 shows the relationship.

MSB-first transfer or LSB first transfer can be selected by the setting of the MLS bit in SSMR. When the MLS bit is 0, transfer is started from LSB to MSB. When the MLS bit is 1, transfer is started from MSB to LSB.

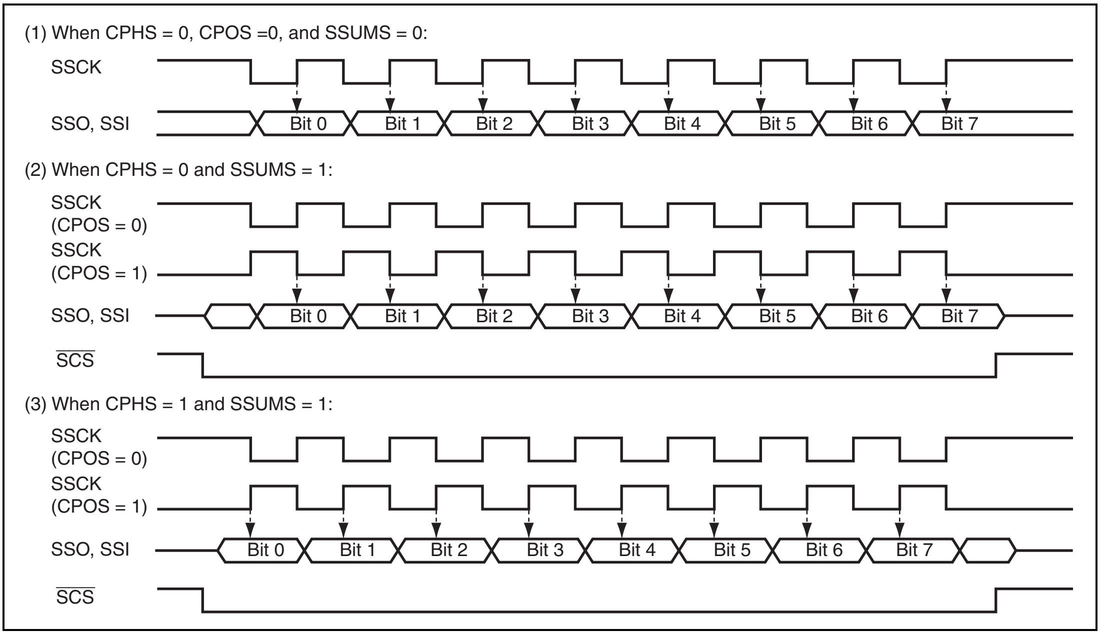


Figure 15.2 Relationship between Clock Polarity and Phase, and Data

(2) Transmit Operation

In transmit mode, transmit data is output from SDA, in synchronization with the fall of the transfer clock. The transfer clock is output when MST in ICCR1 is 1, and is input when MST is 0. For transmit mode operation timing, refer to figure 16.14. The transmission procedure and operations in transmit mode are described below.

1. Set the ICE bit in ICCR1 to 1. Set the MST and CKS3 to CKS0 bits in ICCR1 to 1. (Initial setting)
2. Set the TRS bit in ICCR1 to select the transmit mode. Then, TDRE in ICSR is set.
3. Confirm that TDRE has been set. Then, write the transmit data to ICDRT. The data is transferred from ICDRT to ICDRS, and TDRE is set automatically. The continuous transmission is performed by writing data to ICDRT every time TDRE is set. When changing from transmit mode to receive mode, clear TRS while TDRE is 1.

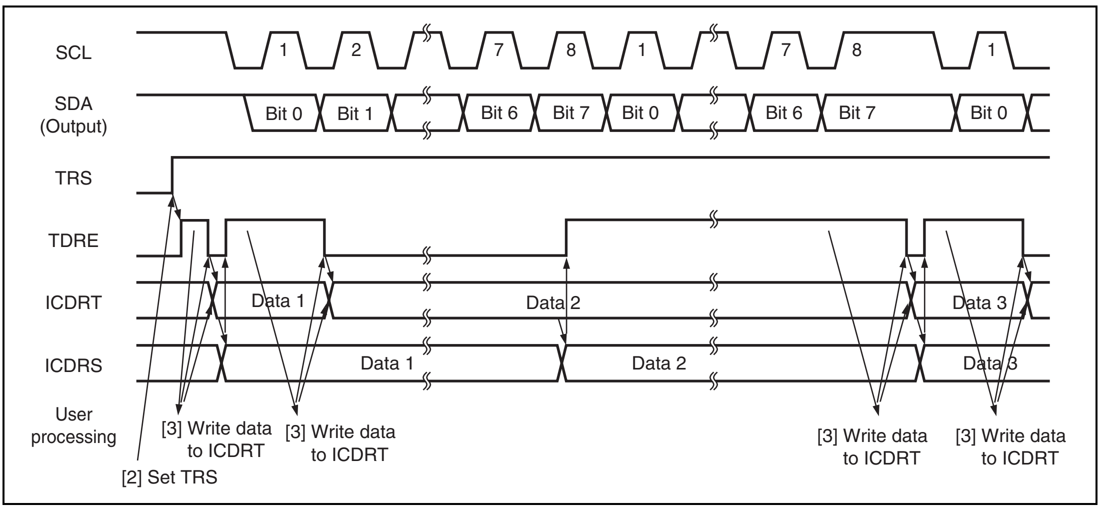


Figure 16.14 Transmit Mode Operation Timing

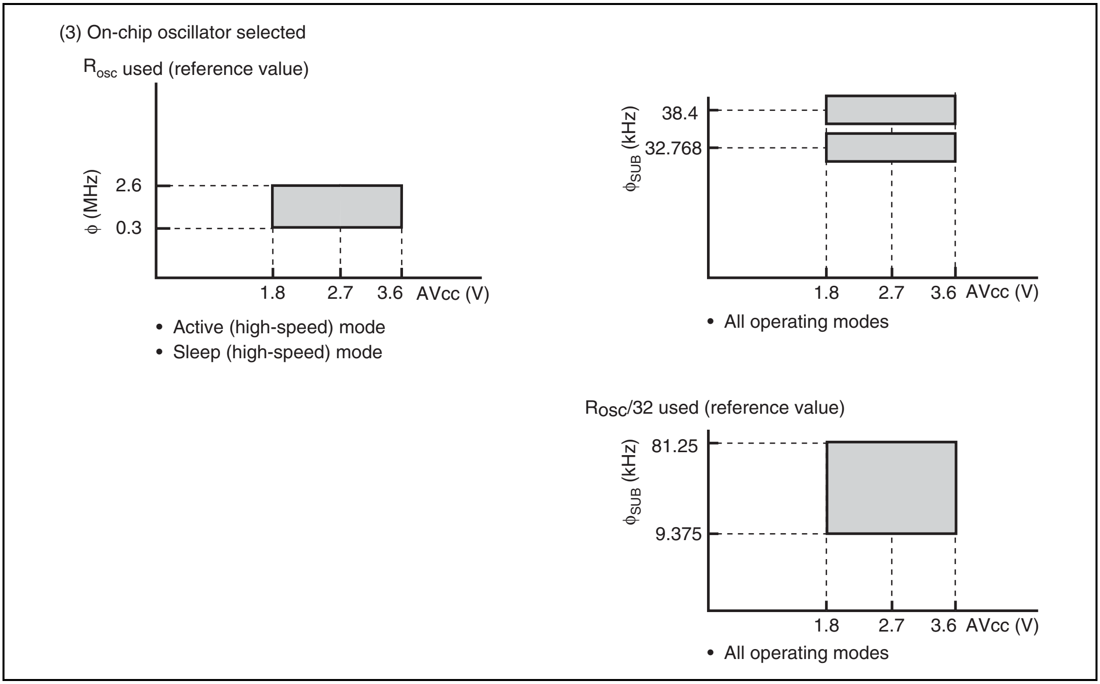


Figure 21.7 Analog Power Supply Voltage and Operating Frequency Range of A/D Converter (2)

Item	Symbol	Applicable Pins	Test Condition	Values Min.	Values Typ.	Values Max.	Unit	Notes
Permissible output high current (per pin)	$-I_{OH}$	All output pins	V_{CC} = 2.7 V to 3.6 V	—	—	2.0	mA	
			Other than above	—	—	0.2		
Permissible output high current (total)	$\sum - I_{OH}$	All output pins		—	—	10.0	mA	

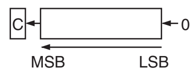
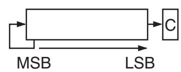
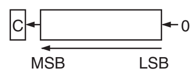
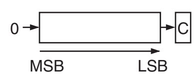
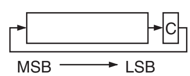
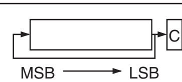
Notes: 1. Pin states during current measurement.

Mode	$\overline{RES}$ Pin	Internal State	Other Pins	Oscillator Pins
Active (high-speed) mode (I_{OPE1})	V_{CC}	Only CPU operates	V_{CC}	System clock oscillator: Crystal resonator Subclock oscillator: Pin X1 = GND
Active (medium-speed) mode (I_{OPE2})				
Sleep mode	V_{CC}	Only on-chip timers operate	V_{CC}	
Subactive mode	V_{CC}	Only CPU operates	V_{CC}	System clock oscillator: Crystal resonator Subclock oscillator: Crystal resonator
Subsleep mode	V_{CC}	Only on-chip timers operate, CPU stops	V_{CC}	
Watch mode	V_{CC}	Only timer base operates, CPU stops	V_{CC}	
Standby mode	V_{CC}	CPU and timers both stop, SUBSTP = 1	V_{CC}	System clock oscillator: Crystal resonator Subclock oscillator: Pin X1 = Crystal resonator

2. Excludes current in pull-up MOS transistors and output buffers.
3. Used for the determination of user mode or boot mode when the reset is released.
4. When bits IRQ0S1 and IRQ0S0 are set to B'01 or B'10, and bits IRQ1S1 and IRQ1S0 are set to B'01 or B'10, the maximum value is given V_{CC} + 0.3 (V).

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@–ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
NEG	NEG.B Rd	B		2								0–Rd8 → Rd8	—	↕	↕	↕	↕	↕	2	
	NEG.W Rd	W		2								0–Rd16 → Rd16	—	↕	↕	↕	↕	↕	2	
	NEG.L ERd	L		2								0–ERd32 → ERd32	—	↕	↕	↕	↕	↕	2	
EXTU	EXTU.W Rd	W		2								0 → (<bits 15 to 8> of Rd16)	—	—	0	↕	0	—	2	
	EXTU.L ERd	L		2								0 → (<bits 31 to 16> of ERd32)	—	—	0	↕	0	—	2	
EXTS	EXTS.W Rd	W		2								(<bit 7> of Rd16) → (<bits 15 to 8> of Rd16)	—	—	↕	↕	0	—	2	
	EXTS.L ERd	L		2								(<bit 15> of ERd32) → (<bits 31 to 16> of ERd32)	—	—	↕	↕	0	—	2	

4. Shift Instructions

Mnemonic		Operand Size	Addressing Mode and Instruction Length (bytes)									Operation	Condition Code						No. of States*1	
			#xx	Rn	@ERn	@(d, ERn)	@−ERn/@ERn+	@aa	@(d, PC)	@@aa	—		I	H	N	Z	V	C	Normal	Advanced
SHAL	SHAL.B Rd	B		2									—	—	↕	↕	↕	↕	2	
	SHAL.W Rd	W		2									—	—	↕	↕	↕	↕	2	
	SHAL.L ERd	L		2									—	—	↕	↕	↕	↕	2	
SHAR	SHAR.B Rd	B		2									—	—	↕	↕	0	↕	2	
	SHAR.W Rd	W		2									—	—	↕	↕	0	↕	2	
	SHAR.L ERd	L		2									—	—	↕	↕	0	↕	2	
SHLL	SHLL.B Rd	B		2									—	—	↕	↕	0	↕	2	
	SHLL.W Rd	W		2									—	—	↕	↕	0	↕	2	
	SHLL.L ERd	L		2									—	—	↕	↕	0	↕	2	
SHLR	SHLR.B Rd	B		2									—	—	↕	↕	0	↕	2	
	SHLR.W Rd	W		2									—	—	↕	↕	0	↕	2	
	SHLR.L ERd	L		2									—	—	↕	↕	0	↕	2	
ROTXL	ROTXL.B Rd	B		2									—	—	↕	↕	0	↕	2	
	ROTXL.W Rd	W		2									—	—	↕	↕	0	↕	2	
	ROTXL.L ERd	L		2									—	—	↕	↕	0	↕	2	
ROTXR	ROTXR.B Rd	B		2									—	—	↕	↕	0	↕	2	
	ROTXR.W Rd	W		2									—	—	↕	↕	0	↕	2	
	ROTXR.L ERd	L		2									—	—	↕	↕	0	↕	2	
ROTL	ROTL.B Rd	B		2									—	—	↕	↕	0	↕	2	
	ROTL.W Rd	W		2									—	—	↕	↕	0	↕	2	
	ROTL.L ERd	L		2									—	—	↕	↕	0	↕	2	
ROTR	ROTR.B Rd	B		2									—	—	↕	↕	0	↕	2	
	ROTR.W Rd	W		2									—	—	↕	↕	0	↕	2	
	ROTR.L ERd	L		2									—	—	↕	↕	0	↕	2	

A.2 Operation Code Map

Table A.2 Operation Code Map (1)

Instruction code:

1st byte		2nd byte	
AH	AL	BH	BL

Split cells (shown as "X / Y" below): X = Instruction when most significant bit of BH is 0. Y = Instruction when most significant bit of BH is 1.

AH \ AL	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NOP	Table A-2 (2)	STC	LDC	ORC	XORC	ANDC	LDC	ADD		Table A-2 (2)	Table A-2 (2)	MOV		ADDX	Table A-2 (2)
1	Table A-2 (2)	Table A-2 (2)	Table A-2 (2)	Table A-2 (2)	OR.B	XOR.B	AND.B	Table A-2 (2)	SUB		Table A-2 (2)	Table A-2 (2)	CMP		SUBX	Table A-2 (2)
2	MOV.B															
3																
4	BRA	BRN	BHI	BLS	BCC	BCS	BNE	BEQ	BVC	BVS	BPL	BMI	BGE	BLT	BGT	BLE
5	MULXU	DIVXU	MULXU	DIVXU	RTS	BSR	RTE	TRAPA	Table A-2 (2)	JMP			BSR	JSR		
6	BSET	BNOT	BCLR	BTST	OR	XOR	AND	BST / BIST	MOV							
7					BOR / BIOR	BXOR / BIXOR	BAND / BIAND	BLD / BILD	MOV	Table A-2 (2)	Table A-2 (2)	EEPMOV	Table A-2 (3)			
8	ADD															
9	ADDX															
A	CMP															
B	SUBX															
C	OR															
D	XOR															
E	AND															
F	MOV															

A.3 Number of Execution States

The status of execution for each instruction of the H8/300H CPU and the method of calculating the number of states required for instruction execution are shown below. Table A.4 shows the number of cycles of each type occurring in each instruction, such as instruction fetch and data read/write. Table A.3 shows the number of states required for each cycle. The total number of states required for execution of an instruction can be calculated by the following expression:

$$\text{Execution states} = I \times S_I + J \times S_J + K \times S_K + L \times S_L + M \times S_M + N \times S_N$$

Examples: When instruction is fetched from on-chip ROM, and an on-chip RAM is accessed.

BSET #0, @FF00

From table A.4:
$I = L = 2, \quad J = K = M = N = 0$

From table A.3:
$S_I = 2, \quad S_L = 2$

Number of states required for execution = 2 × 2 + 2 × 2 = 8

When instruction is fetched from on-chip ROM, branch address is read from on-chip ROM, and on-chip RAM is used for stack area.

JSR @@ 30

From table A.4:
$I = 2, \quad J = K = 1, \quad L = M = N = 0$

From table A.3:
$S_I = S_J = S_K = 2$

Number of states required for execution = 2 × 2 + 1 × 2+ 1 × 2 = 8