Hitachi Single-Chip Microcomputer
H8/330

ADE-802-030A(O)
(21-9A)

HITACHI

When using this document, keep the following in mind:

1. This document may, wholly or partially, be subject to change without notice.
2. All rights are reserved: No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without Hitachi's permission.
3. Hitachi will not be held responsible for any damage to the user that may result from accidents or any other reasons during operation of the user's unit according to this document.
4. Circuitry and other examples described herein are meant merely to indicate the characteristics and performance of Hitachi's semiconductor products. Hitachi assumes no responsibility for any intellectual property claims or other problems that may result from applications based on the examples described herein.
5. No license is granted by implication or otherwise under any patents or other rights of any third party or Hitachi, Ltd.
6. MEDICAL APPLICATIONS: Hitachi's products are not authorized for use in MEDICAL APPLICATIONS without the written consent of the appropriate officer of Hitachi's sales company. Such use includes, but is not limited to, use in life support systems. Buyers of Hitachi's products are requested to notify the relevant Hitachi sales offices when planning to use the products in MEDICAL APPLICATIONS.

Copyright ©Hitachi, Ltd. 1989. All rights reserved.

Contents

Preface

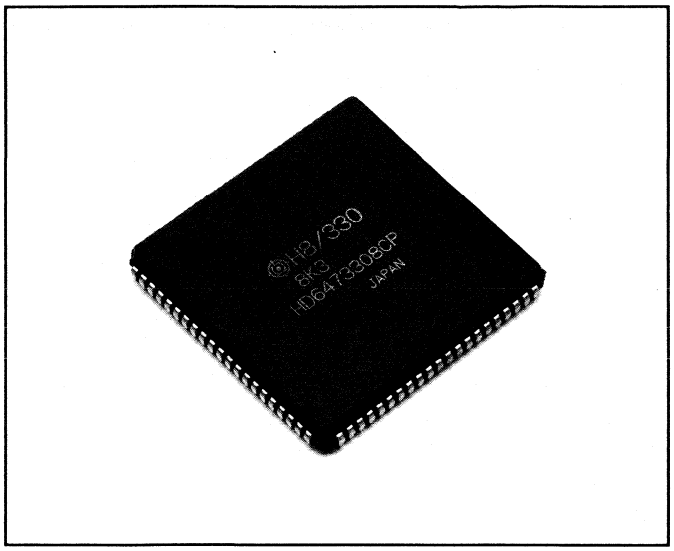
This document describes the H8/330, the first product in the H8/300 lineup.

Supplementing the H8/300 CPU core in the H8/330 are many of the supporting functions required in application systems, including a 16K-byte ROM, a 512-byte static RAM, three types of timers, a serial I/O interface, a dual-port RAM (DPRAM), an A/D converter, and general-purpose I/O ports, enabling high-performance systems to be designed with a low chip count.

The H8/330 is available in both a masked ROM version and a user-programmable PROM version. The PROM version, also known as the ZTAT (Zero Turn-Around Time) version, is suitable for products with frequently-changing specifications, or for the early stages of volume production.

Hitachi is working to provide a full, efficient development environment for microcomputer application systems. In addition to support software, the environment will include an ASE (Adaptive System Evaluator): a stand-alone emulator that can also be connected to a general-purpose computer.

Section 1. H8/330 Features



- **H8/300 CPU: a Hitachi-original architecture with high cost-performance**

- General-register machine
  - — Sixteen 8-bit general registers (or eight 16-bit general registers)
- High speed
  - — Maximum clock rate—10MHz (ø clock)
  - — High-speed instructions
    - 8- or 16-bit register-register add .....0.2µs (at 10MHz)
    - 8 × 8-bit multiply .....1.4µs (at 10MHz)
    - 16 ÷ 8-bit divide .....1.4µs (at 10MHz)
- RISC-like speed-oriented instruction set
  - — Fifty-seven types of instructions
  - — Instruction length: 2 or 4 bytes
  - — Fast multiply and divide instructions; versatile bit-manipulation instructions
- Maximum 64K-byte address space
- Three operating modes
  - — Mode 1 (up to 64K-byte address space, on-chip ROM disabled)
  - — Mode 2 (up to 64K-byte address space, on-chip ROM enabled)
  - — Mode 3 (single-chip mode: on-chip ROM, RAM and registers only)

- **16K bytes of on-chip PROM (or masked ROM)**

- **512 bytes of on-chip RAM**

- **On-chip dual-port RAM**
  - — Control register (1 byte) + 15-byte data register
  - — Simple protocol for easy implementation of master-slave systems

- **16-Bit free-running timer on-chip (1 channel) with two output compare waveforms and four input capture inputs**

- **8-Bit multifunction timer on-chip (2 channels)**

- **PWM timer on-chip (2 channels)**

- **Serial communication interface on-chip (1 channel)**
  - — Asynchronous or synchronous mode (selectable)
  - — On-chip baud rate generator
  - — Separate pins for asynchronous and synchronous modes

- **8-Bit A/D converter on-chip (8 channels)**
  - — Single mode or scan mode (selectable)
  - — Sample-and-hold function
  - — A/D conversion can be externally triggered

- **Nine I/O ports**
  - — 58 Input/output pins (of which 16 can drive large current loads)
  - — 8 Input-only pins
  - — All input pins (except port 7) have software-programmable pull-up resistors

- **Interrupts**
  - — Nine external interrupt pins ($\overline{\text{NMI}}$, $\overline{\text{IRQ}_0}$ to $\overline{\text{IRQ}_7}$)
  - — Nineteen internal interrupt sources
  - — Eight priority levels

- **Power-down state**
  - — Sleep mode, software standby mode, hardware standby mode

- **E clock output available**

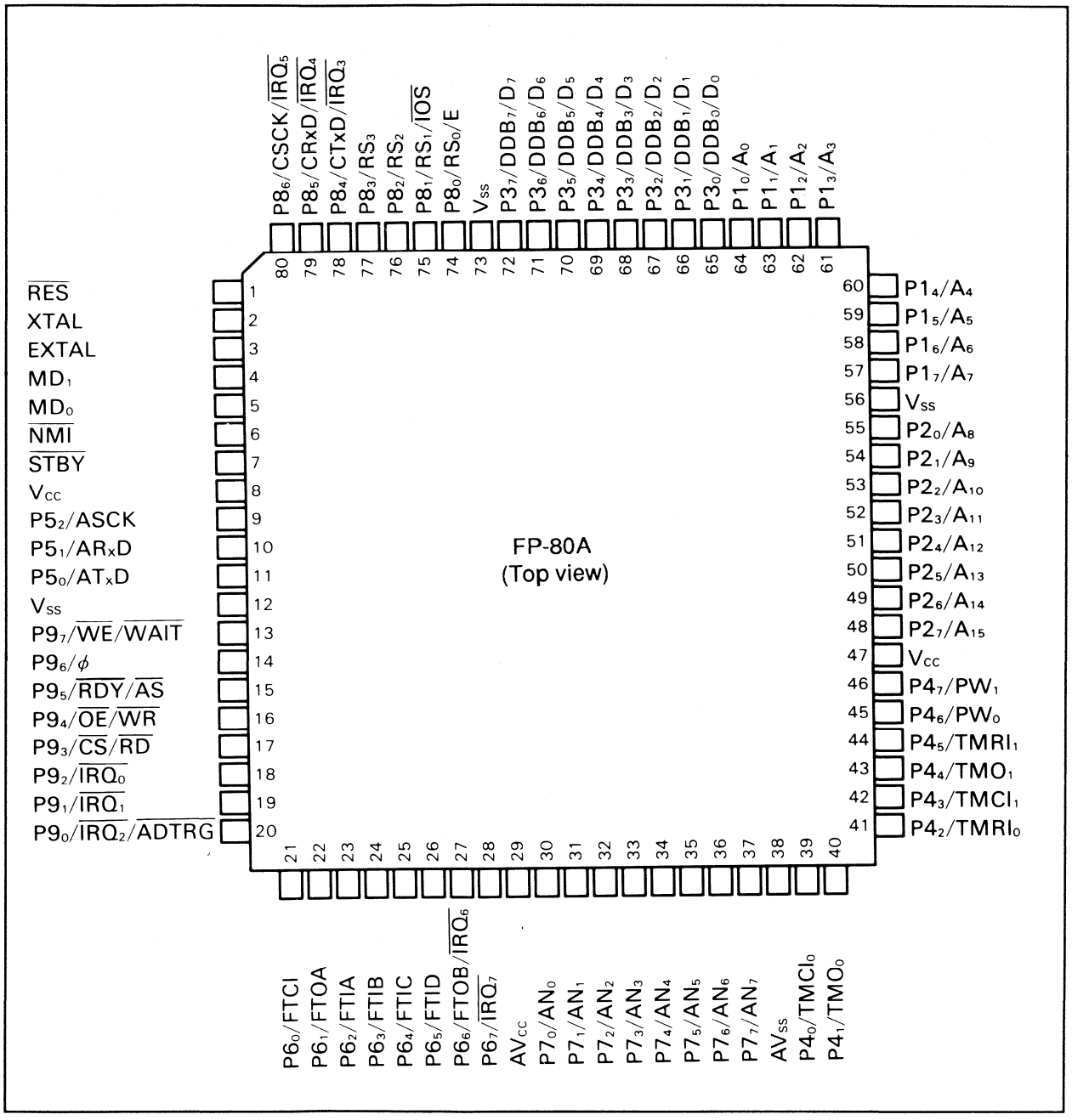
- **Clock pulse generator on-chip**

- **Package options**
  - — 80-Pin Plastic QFP (FP-80A)
  - — 84-Pin PLCC (CP-84)
  - — 84-Pin Windowed LCC (CG-84)

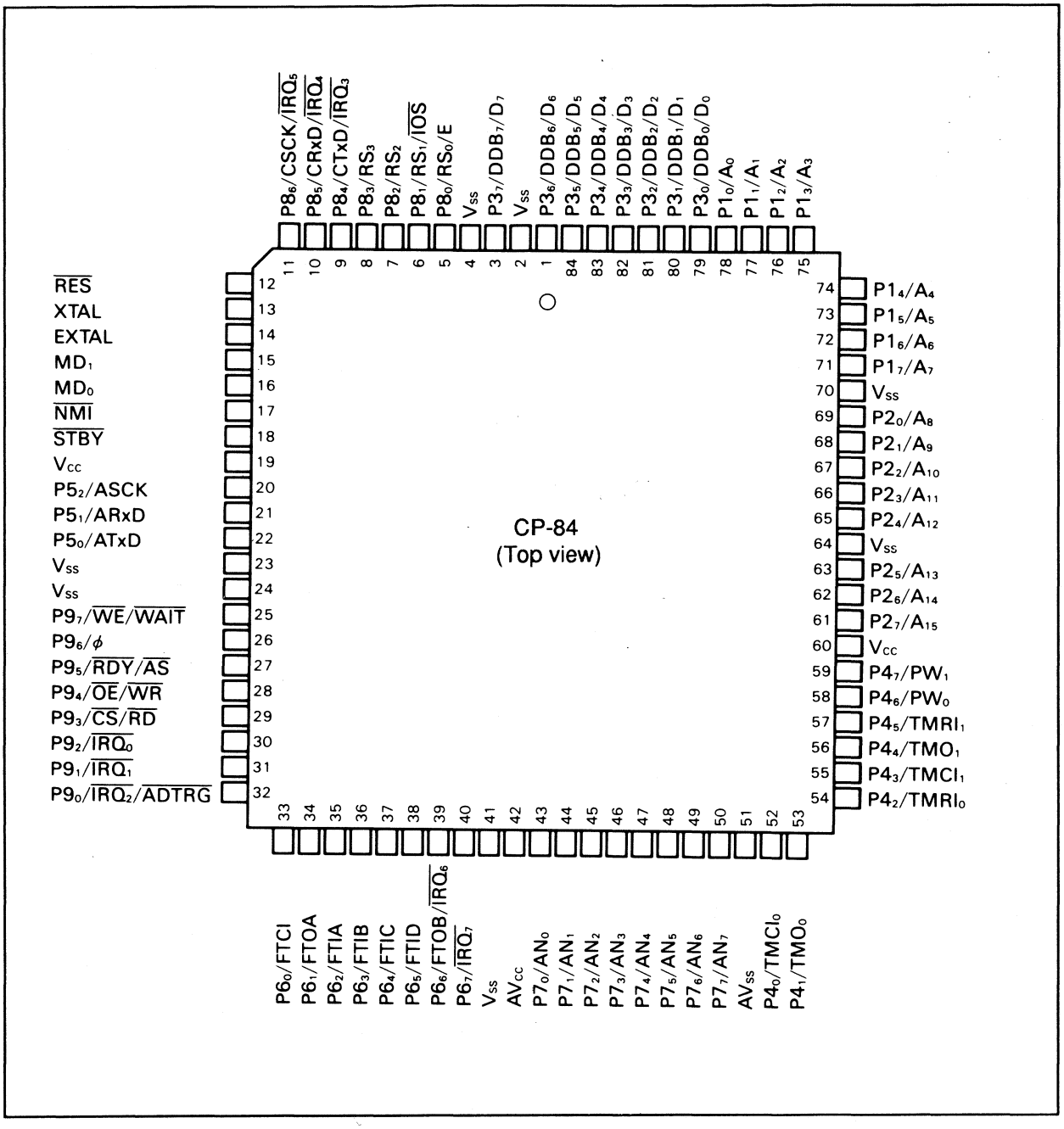
Section 2. Pin Assignments and Functions

2.1 Pin Assignments

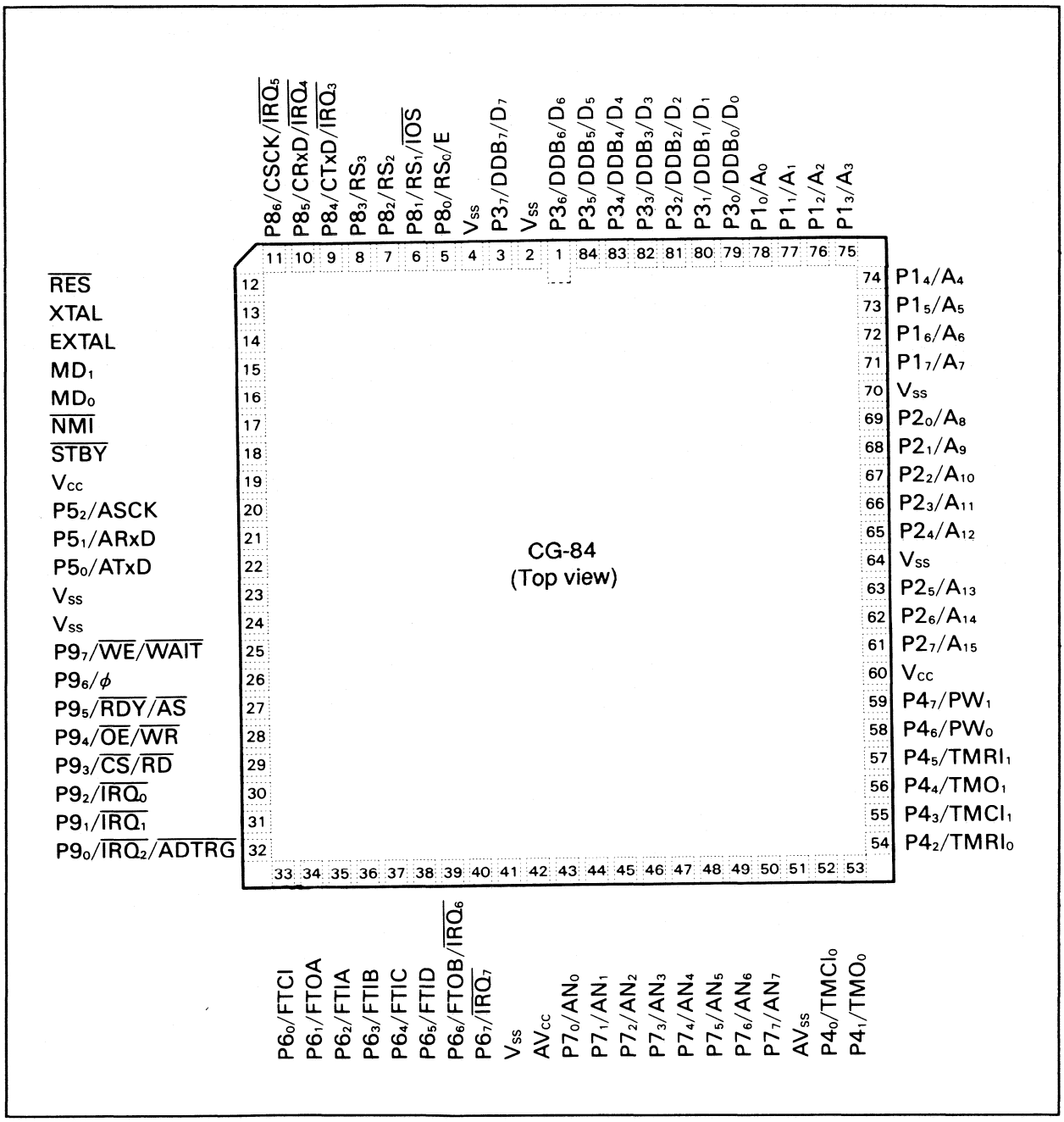
- **80-Pin Plastic QFP**



- **84-Pin PLCC**


CP-84
(Top view)
P8₆/CSCK/IRQ₅
P8₅/CRxD/IRQ₄
P8₄/CTxD/IRQ₃
P8₃/RS₃
P8₂/RS₂
P8₁/RS₁/IOS
P8₀/RS₀/E
Vss
P3₇/DDB₇/D₇
Vss
P3₆/DDB₆/D₆
P3₅/DDB₅/D₅
P3₄/DDB₄/D₄
P3₃/DDB₃/D₃
P3₂/DDB₂/D₂
P3₁/DDB₁/D₁
P3₀/DDB₀/D₀
P1₀/A₀
P1₁/A₁
P1₂/A₂
P1₃/A₃
RES
XTAL
EXTAL
MD₁
MD₀
NMI
STBY
Vcc
P5₂/ASCK
P5₁/ARxD
P5₀/ATxD
Vss
Vss
P9₇/WE/WAIT
P9₆/φ
P9₅/RDY/AS
P9₄/OE/WR
P9₃/CS/RD
P9₂/IRQ₀
P9₁/IRQ₁
P9₀/IRQ₂/ADTRG
P1₄/A₄
P1₅/A₅
P1₆/A₆
P1₇/A₇
Vss
P2₀/A₈
P2₁/A₉
P2₂/A₁₀
P2₃/A₁₁
P2₄/A₁₂
Vss
P2₅/A₁₃
P2₆/A₁₄
P2₇/A₁₅
Vcc
P4₇/PW₁
P4₆/PW₀
P4₅/TMRI₁
P4₄/TMO₁
P4₃/TMCI₁
P4₂/TMRI₀
P6₀/FTCI
P6₁/FTOA
P6₂/FTIA
P6₃/FTIB
P6₄/FTIC
P6₅/FTID
P6₆/FTOB/IRQ₆
P6₇/IRQ₇
Vss
AVcc
P7₀/AN₀
P7₁/AN₁
P7₂/AN₂
P7₃/AN₃
P7₄/AN₄
P7₅/AN₅
P7₆/AN₆
P7₇/AN₇
AVss
P4₀/TMCI₀
P4₁/TMO₀

- **84-Pin Windowed LCC**



2.2 Pin Assignments in Each Mode Operating

Pin No.		Pin name				Pin No.		Pin name			
CP-84 CG-84	FP-80A	Single-chip mode		Expanded modes		CP-84 CG-84	FP-80A	Single-chip mode		Expanded modes	
		Non-slave*	Slave*	Mode 1	Mode 2			Non-slave*	Slave*	Mode 1	Mode 2
1	71	P36	OBB6	D6	D6	43	30	P70/AN0		P70/AN0	P70/AN0
2	—	Vss		Vss	Vss	44	31	P71/AN1		P71/AN1	P71/AN1
3	72	P37	OBB7	D7	D7	45	32	P72/AN2		P72/AN2	P72/AN2
4	73	Vss		Vss	Vss	46	33	P73/AN3		P73/AN3	P73/AN3
5	74	P80	RS0	P80/E	P80/E	47	34	P74/AN4		P74/AN4	P74/AN4
6	75	P81	RS1	P81/$\overline{IOS}$	P81/$\overline{IOS}$	48	35	P75/AN5		P75/AN5	P75/AN5
7	76	P82	RS2	P82	P82	49	36	P76/AN6		P76/AN6	P76/AN6
8	77	P83	RS3	P83	P83	50	37	P77/AN7		P77/AN7	P77/AN7
9	78	P84/CTxD/$\overline{IRQ3}$		P84/CTxD/$\overline{IRQ3}$	P84/CTxD/$\overline{IRQ3}$	51	38	AVss		AVss	AVss
10	79	P85/CRxD/$\overline{IRQ4}$		P85/CRxD/$\overline{IRQ4}$	P85/CRxD/$\overline{IRQ4}$	52	39	P40/TMCI0		P40/TMCI0	P40/TMCI0
11	80	P86/CSCK/$\overline{IRQ5}$		P86/CSCK/$\overline{IRQ5}$	P86/CSCK/$\overline{IRQ5}$	53	40	P41/TMO0		P41/TMO0	P41/TMO0
12	1	$\overline{RES}$		$\overline{RES}$	$\overline{RES}$	54	41	P42/TMRI0		P42/TMRI0	P42/TMRI0
13	2	XTAL		XTAL	XTAL	55	42	P43/TMCI1		P43/TMCI1	P43/TMCI1
14	3	EXTAL		EXTAL	EXTAL	56	43	P44/TMO1		P44/TMO1	P44/TMO1
15	4	MD1		MD1	MD1	57	44	P45/TMRI1		P45/TMRI1	P45/TMRI1
16	5	MD0		MD0	MD0	58	45	P46/PW0		P46/PW0	P46/PW0
17	6	$\overline{NMI}$		$\overline{NMI}$	$\overline{NMI}$	59	46	P47/PW1		P47/PW1	P47/PW1
18	7	$\overline{STBY}$		$\overline{STBY}$	$\overline{STBY}$	60	47	Vcc		Vcc	Vcc
19	8	Vcc		Vcc	Vcc	61	48	P27		A15	P27/A15
20	9	P52/ASCK		P52/ASCK	P52/ASCK	62	49	P26		A14	P26/A14
21	10	P51/ARxD		P51/ARxD	P51/ARxD	63	50	P25		A13	P25/A13
22	11	P50/ATxD		P50/ATxD	P50/ATxD	64	—	Vss		Vss	Vss
23	12	Vss		Vss	Vss	65	51	P24		A12	P24/A12
24	—	Vss		Vss	Vss	66	52	P23		A11	P23/A11
25	13	P97	$\overline{WE}$	$\overline{WAIT}$	$\overline{WAIT}$	67	53	P22		A10	P22/A10
26	14	P96/ø		ø	ø	68	54	P21		A9	P21/A9
27	15	P95	$\overline{RDY}$	$\overline{AS}$	$\overline{AS}$	69	55	P20		A8	P20/A8
28	16	P94	$\overline{OE}$	$\overline{WR}$	$\overline{WR}$	70	56	Vss		Vss	Vss
29	17	P93	$\overline{CS}$	$\overline{RD}$	$\overline{RD}$	71	57	P17		A7	P17/A7
30	18	P92/$\overline{IRQ0}$		P92/$\overline{IRQ0}$	P92/$\overline{IRQ0}$	72	58	P16		A6	P16/A6
31	19	P91/$\overline{IRQ1}$		P91/$\overline{IRQ1}$	P91/$\overline{IRQ1}$	73	59	P15		A5	P15/A5
32	20	P90/$\overline{IRQ2}$/$\overline{ADTRG}$		P90/$\overline{IRQ2}$/$\overline{ADTRG}$	P90/$\overline{IRQ2}$/$\overline{ADTRG}$	74	60	P14		A4	P14/A4
33	21	P60/FTCI		P60/FTCI	P60/FTCI	75	61	P13		A3	P13/A3
34	22	P61/FTOA		P61/FTOA	P61/FTOA	76	62	P12		A2	P12/A2
35	23	P62/FTIA		P62/FTIA	P62/FTIA	77	63	P11		A1	P11/A1
36	24	P63/FTIB		P63/FTIB	P63/FTIB	78	64	P10		A0	P10/A0
37	25	P64/FTIC		P64/FTIC	P64/FTIC	79	65	P30	DDB0	D0	D0
38	26	P65/FTID		P65/FTID	P65/FTID	80	66	P31	DDB1	D1	D1
39	27	P66/FTOB/$\overline{IRQ6}$		P66/FTOB/$\overline{IRQ6}$	P66/FTOB/$\overline{IRQ6}$	81	67	P32	DDB2	D2	D2
40	28	P67/$\overline{IRQ7}$		P67/$\overline{IRQ7}$	P67/$\overline{IRQ7}$	82	68	P33	DDB3	D3	D3
41	—	Vss		Vss	Vss	83	69	P34	DDB4	D4	D4
42	25	AVcc		AVcc	AVcc	84	70	P35	DDB5	D5	D5

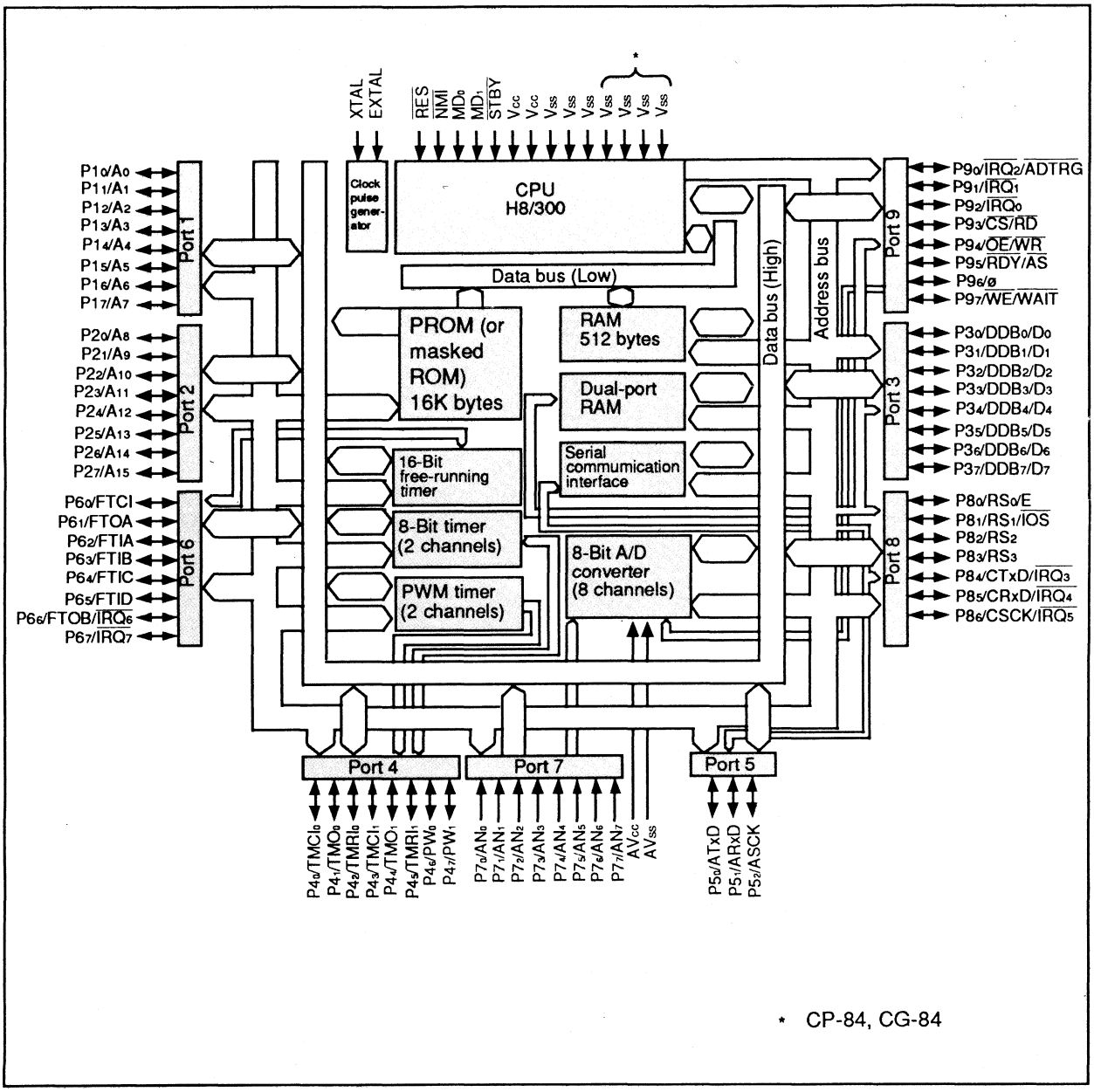
* Non-slave: when the dual-port RAM is disabled (DPME=0)
Slave: when the dual-port RAM is enabled (DPME=1)

2.3 Pin Functions

Type	Symbol	I/O	Function
Power	V_{CC}	I	Power supply
	V_{SS}	I	Ground
Clock	XTAL	I	Crystal
	EXTAL	I	External clock
	ø	O	System clock
	E	O	Enable clock
System control	$\overline{RES}$	I	Reset
	$\overline{STBY}$	I	Standby
Address bus	A_0-A_{15}	O	Address bus
Data bus	D_0-D_7	I/O	Data bus
Bus control	$\overline{WAIT}$	I	Wait
	$\overline{RD}$	O	Read
	$\overline{WR}$	O	Write
	$\overline{AS}$	O	Address strobe
	$\overline{IOS}$	O	I/O select
Interrupts	$\overline{NMI}$	I	Nonmaskable interrupt
	$\overline{IRQ_0}$-$\overline{IRQ_7}$	I	Interrupt request 0 to 7
Operating mode control	MD_0, MD_1	I	Mode control
16-Bit free-running timer (FRT)	FTCI	I	FRT counter clock input
	FTOA	O	FRT output compare A
	FTOB	O	FRT output compare B
	FTIA	I	FRT input capture A
	FTIB	I	FRT input capture B
	FTIC	I	FRT input capture C
	FTID	I	FRT input capture D
8-Bit timer (TMR)	TMO_0, TMO_1	O	8-Bit timer clock output (channels 0 and 1)
	$TMCI_0$, TMC_1	I	8-Bit timer clock input (channels 0 and 1)
	$TMRI_0$, $TMRI_1$	I	8-Bit timer counter reset input (channels 0 and 1)

Type		Symbol	I/O	Function
PWM timer		PW_0, PW_1	O	PWM timer output (channels 0 and 1)
Serial communication interface (SCI)	Asynchronous	ATxD	O	Transmit data output
		ARxD	I	Receive data input
		ASCK	I/O	Serial clock input/output
	Synchronous	CTxD	O	Transmit data output
		CRxD	I	Receive data input
		CSCK	I/O	Serial clock input/output
A/D converter		AN_0-AN_7	I	Analog input
		$\overline{ADTRG}$	I	A/D conversion external trigger input
		AVcc	I	Analog power supply
		AVss	I	Analog ground
Dual-port RAM (DPRAM)		DDB_0-DDB_7	I	DPRAM data bus
		$\overline{CS}$	I	Chip select
		RS_0-RS_3	I	DPRAM register select
		$\overline{OE}$	I	Output enable
		$\overline{WE}$	I	Write enable
		$\overline{RDY}$	O	Ready
I/O ports		$P1_0$-$P1_7$	I/O	Port 1
		$P2_0$-$P2_7$	I/O	Port 2
		$P3_0$-$P3_7$	I/O	Port 3
		$P4_0$-$P4_7$	I/O	Port 4
		$P5_0$-$P5_2$	I/O	Port 5
		$P6_0$-$P6_7$	I/O	Port 6
		$P7_0$-$P7_7$	I	Port 7
		$P8_0$-$P8_6$	I/O	Port 8
		$P9_0$-$P9_7$	I/O	Port 9

Section 3. Block Diagram



Section 4. CPU

The H8/330 has the generic H8/300 CPU: a central processing unit with a Hitachi-original architecture featuring sixteen 8-bit general registers and a concise, optimized instruction set. The general registers can also be paired as eight 16-bit registers. Arithmetic and logic operations and data transfers are carried out at high speed with a maximum system clock rate of 10MHz.

4.1 Address Space

The CPU supports a maximum address space of 64K bytes. The address space configuration depends on the operating mode of the chip, which is selected by the inputs at the mode pins (MD_0 and MD_1) when the chip comes out of a reset. For details, see section 5, "Operating Modes."

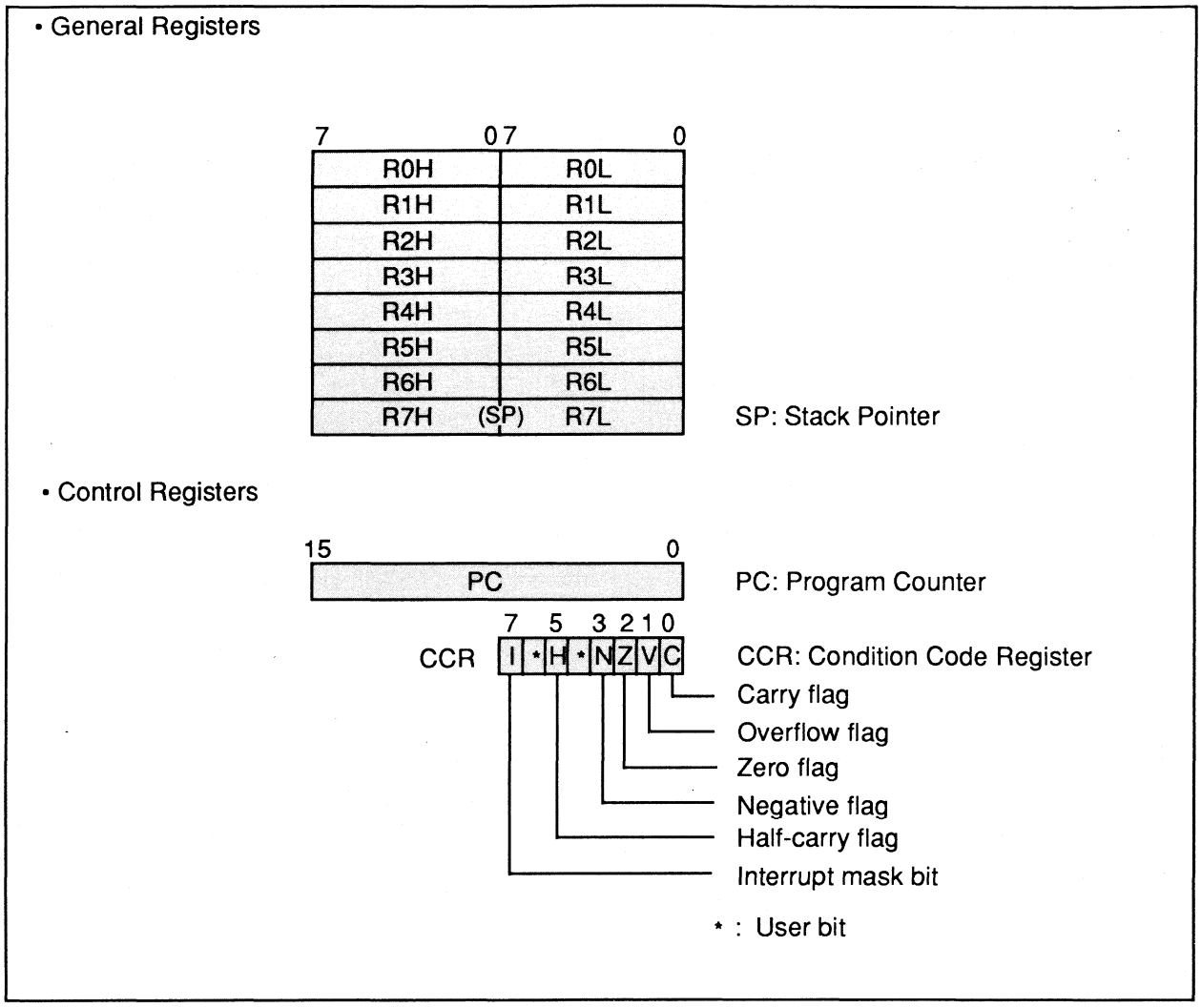
Mode	MD_1	MD_0	Description	ROM	RAM	Interrupt vector table	Register field
Mode 1	0	1	On-chip ROM disabled	External	On-chip*	External	On-chip
Mode 2	1	0	On-chip ROM enabled	On-chip	On-chip*	On-chip	On-chip
Mode 3	1	1	Single-chip mode	On-chip	On-chip	On-chip	On-chip

0: Low 1: High

* External address space when the RAM enable (RAME) bit in the system control register is cleared to "0."

4.2 Register Configuration

The register structure of the CPU is shown below. Besides the 8-bit general registers (R0H/ R0L to R7H /R7L) there is a 16-bit program counter (PC) and an 8-bit condition code register (CCR).



CPU Registers

• **General Registers:** All the general registers are functionally alike; there is no distinction between data registers and address registers. When used as address registers, the general registers are accessed as 16-bit registers (R0 to R7). When used as data registers, they can be accessed as 16-bit registers, or the high and low bytes can be accessed separately as 8-bit registers. The register length is determined by the instruction.

R7 also functions as the stack pointer (SP), µsed implicitly in exception handling and subroutine calls.

- **Control Registers**

Program Counter (PC): This 16-bit register indicates the address of the next instruction the CPU will execute.

Condition Code Register (CCR): This 8-bit register contains internal status information, including carry (C), overflow (V), zero (Z), negative (N), and half-carry (H) flags and the interrupt mask bit (I). Special instructions are provided for loading and storing the CCR, or setting and clearing selected bits by logic operations.

Bit 7—Interrupt Mask Bit (I): This bit masks interrupts (other than NMI) when set to "1." It is set to "1" at the beginning of interrupt handling.

Bit 6—User Bit: This bit can be written and read by software for its own purposes, using the CCR instructions mentioned above.

Bit 5—Half-Carry (H): This bit is set to "1" when the ADD.B, ADDX.B, SUB.B, SUBX.B, or CMP.B instruction causes a carry or borrow out of bit 3, and is cleared to "0" when one of these instructions is executed without causing a carry or borrow out of bit 3. Similarly, it is set to "1" when the ADD.W, SUB.W, or CMP.W instruction causes a carry or borrow out of bit 11, and cleared to "0" when one of these instructions is executed without causing a carry or borrow out of bit 11. It is used implicitly by the DAA and DAS instructions.

Bit 4—User Bit: This bit can be written and read by software for its own purposes, using the CCR instructions mentioned above.

Bit 3—Negative (N): This bit indicates the most significant bit (sign bit) of the result of an instruction.

Bit 2—Zero (Z): This bit is set to "1" to indicate a zero result and cleared to "0" to indicate a nonzero result.

Bit 1—Overflow (V): This bit is set to "1" when an arithmetic overflow occurs, and cleared to "0" at other times.

Bit 0—Carry (C): This bit is used by:

- Add and subtract instructions, to indicate a carry or borrow at the most significant bit

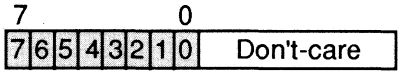
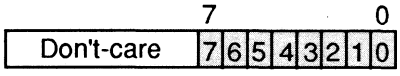
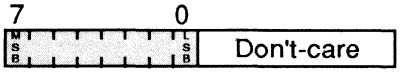
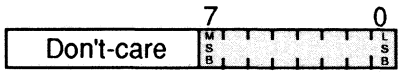
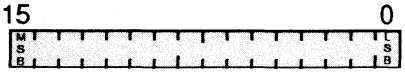
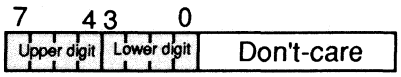
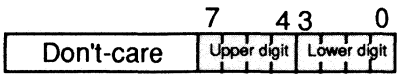
- Shift and rotate instructions, to store the value shifted out of the most significant or least significant bit
- Bit instructions, as a bit accumulator

4.3 Data Formats

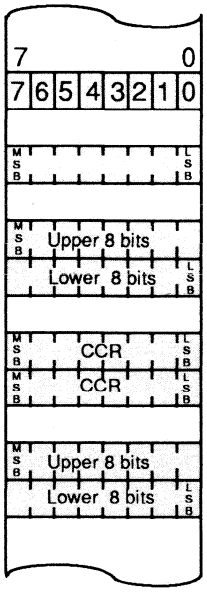
The H8/300 CPU can process 1-bit data, 4-bit (BCD) data, 8-bit (byte) data, and 16-bit (word) data. Essentially all instructions can process byte data. The bit manipulation instructions process 1-bit data. Data transfer instructions and some arithmetic instructions handle word data. The decimal adjust instructions permit BCD data to be used.

The formats in which data are stored in general registers and memory are shown next.

- **Register Data Formats**

Data type	Register No.	Data format
1-Bit data	RnH	
1-Bit data	RnL	
Byte data	RnH	
Byte data	RnL	
Word data	Rn	
4-Bit BCD data	RnH	
4-Bit BCD data	RnL	

• **Memory Data Formats**

Data type	Address	Data format
1-Bit data	Address n	
Byte data	Address n	
Word data	Even address Odd address	
Byte data (CCR) on stack	Even address Odd address	
Word data on stack	Even address Odd address	

Note: Word data must begin at an even address.

4.4 Addressing Modes

The CPU supports the following eight addressing modes.

• **Addressing Modes**

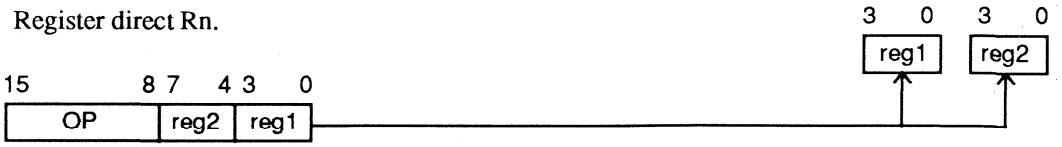
No.	Addressing mode	Mnemonic
1	Register direct	Rn
2	Register indirect	@Rn
3	Register indirect with 16-bit displacement	@(d:16, Rn)
4	Register indirect with pre-decrement or post-increment	@–Rn @Rn+
5	Immediate (8-, or 16-bit data)	#xx:8, #xx:16
6	Absolute address (8 or 16 bits)	@aa:8, @aa:16
7	PC-relative (8-bit displacement)	@(d:8, PC)
8	Memory indirect	@@aa:8

Note: Data transfer instructions can use modes 1 through 6.

• Effective Address Calculation

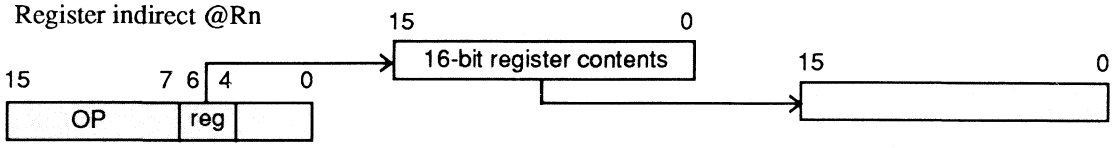
No.	Addressing mode, instruction format	Effective address calculation	Effective address

1 Register direct Rn.



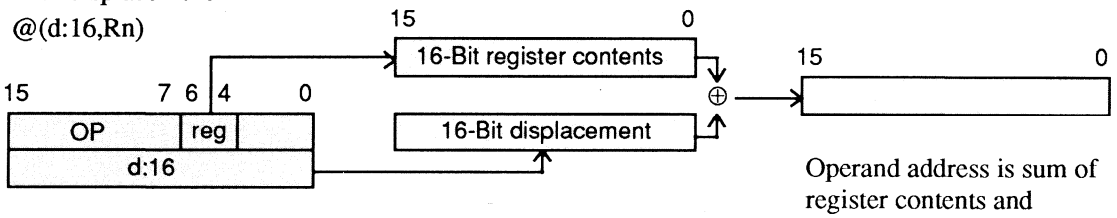
Operands are contained in registers 1 and 2

2 Register indirect @Rn



Operand is at address indicated by register

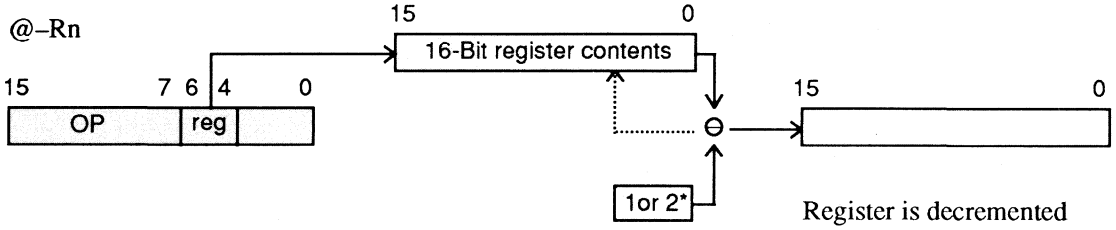
3 Register indirect with displacement @(d:16,Rn)



Operand address is sum of register contents and displacement

4 Register indirect with pre-decrement

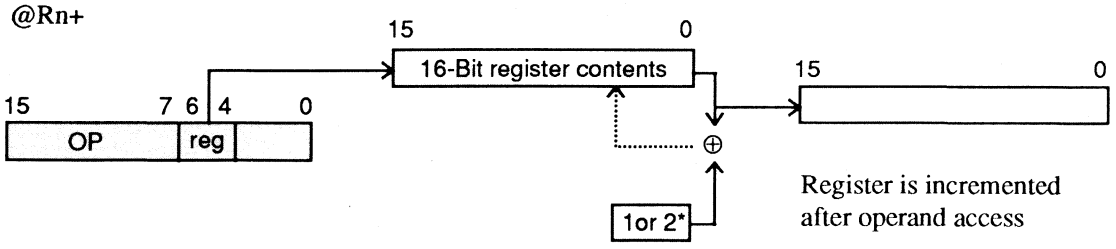
@–Rn



Register is decremented before operand access

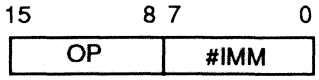
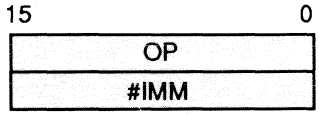
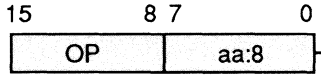
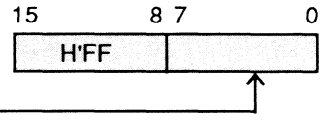
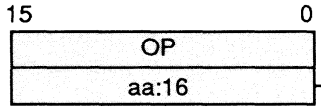
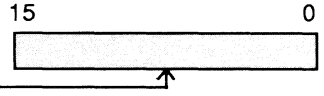
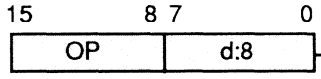
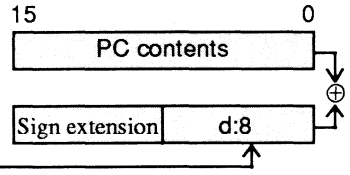
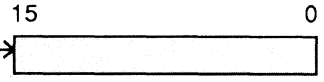
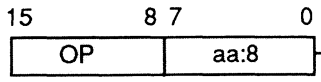
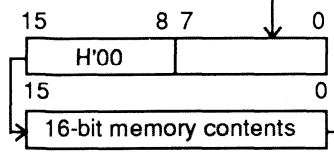
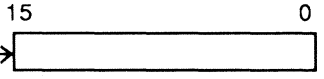
Register indirect with post-increment

@Rn+



Register is incremented after operand access

* 1 for a byte operand,
2 for a word operand

No.	Addressing mode, instruction format	Effective address calculation	Effective address
5	Immediate #xx:8		
			Operand is 1-byte immediate data
	Immediate #xx:16		
			Operand is 2-byte immediate data
6	Absolute address @aa:8		
			Operand address is in range from H'FF00 to H'FFFF
	Absolute address @aa:16		
			Arbitrary address
7	PC-relative @(d:8, PC)		
			 Destination address
8	Memory indirect @@aa:8		
			 Destination address

reg: General register
#IMM: Immediate data
d: Displacement
aa: Absolute address

4.5 Instruction Set

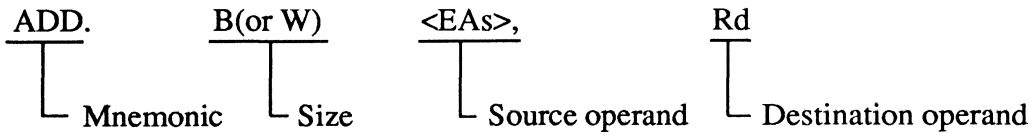
The H8/300 CPU supports 57 instructions with the following features.

- **Features**

- Concise, RISC-like instruction set; all instructions are 2 or 4 bytes long
- High speed
  - — All frequently-used instructions execute in 2 to 4 states
  - — A 16-bit register-register add takes 200ns (with 10MHz system clock)
- General-register architecture
- Powerful bit-manipulation instructions
- Standard H-Series mnemonics

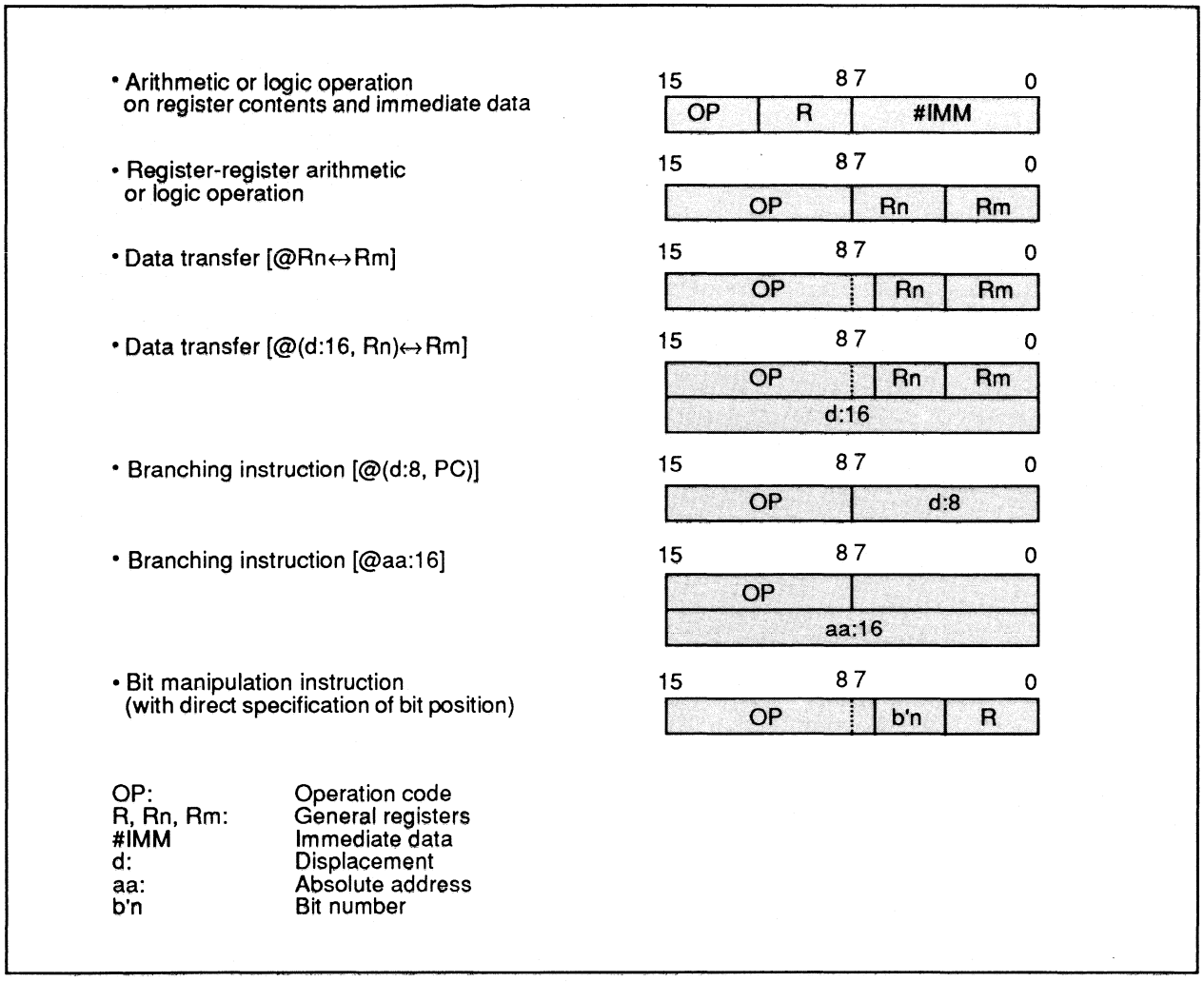
- **Assembly-Language Format**

The ADD instruction below gives an example of the assembly-language format. The letter B (byte) or W (word) designates the operand size. For some instructions only one of these sizes is available.



• Main Instruction Formats

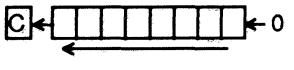
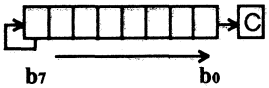
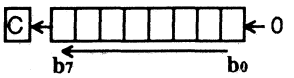
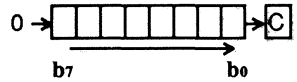
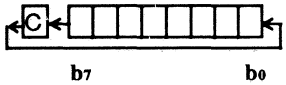
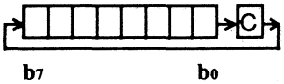
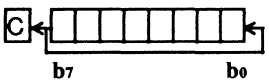
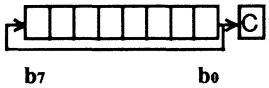
The main instruction formats of the H8/300 CPU are shown below.



• Instruction Set List

	Mnemonic	Operand size	Operation	Addressing mode/ instruction length									Condition code						No. of states
				#xx:	Rn	@Rn	@(d:16,Rn)	@–Rn/@Rn+	@aa:	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
Data transfer instructions	MOV.B Rs,Rd	B	Rs8 → Rd8		2								–	–	↕	↕	0	–	2
	MOV.B #xx:8,Rd	B	#xx:8 → Rd8	2									–	–	↕	↕	0	–	2
	MOV.B @Rs,Rd	B	@Rs16 → Rd8			2							–	–	↕	↕	0	–	4
	MOV.B @(d:16,Rs),Rd	B	@(d:16,Rs16)→ Rd8				4						–	–	↕	↕	0	–	6
	MOV.B @Rs+,Rd	B	@Rs16 → Rd8 Rs16+1 → Rs16					2					–	–	↕	↕	0	–	6
	MOV.B @aa:8,Rd	B	@aa:8 → Rd8						2				–	–	↕	↕	0	–	4
	MOV.B @aa:16,Rd	B	@aa:16 → Rd8						4				–	–	↕	↕	0	–	6
	MOV.B Rs,@Rd	B	Rs8 → @Rd16			2							–	–	↕	↕	0	–	4
	MOV.B Rs,@(d:16,Rd)	B	Rs8 → @(d:16,Rd16)				4						–	–	↕	↕	0	–	6
	MOV.B Rs,@–Rd	B	Rd16–1 → Rd16 Rs8 → @Rd16					2					–	–	↕	↕	0	–	6
	MOV.B Rs,@aa:8	B	Rs8 → @aa:8						2				–	–	↕	↕	0	–	4
	MOV.B Rs,@aa:16	B	Rs8 → @aa:16						4				–	–	↕	↕	0	–	6
	MOV.W Rs,Rd	W	Rs16 → Rd16		2								–	–	↕	↕	0	–	2
	MOV.W @Rs,Rd	W	@Rs16 → Rd16			2							–	–	↕	↕	0	–	4
	MOV.W @(d:16,Rs),Rd	W	@(d:16,Rs16) → Rd16				4						–	–	↕	↕	0	–	6
	MOV.W @Rs+,Rd	W	@Rs16 → Rd16 Rs16+2 → Rs16					2					–	–	↕	↕	0	–	6
	MOV.W @aa:16,Rd	W	@aa:16 → Rd16						4				–	–	↕	↕	0	–	6
	MOV.W Rs,@Rd	W	Rs16 → @Rd16			2							–	–	↕	↕	0	–	4
	MOV.W Rs,@(d:16,Rd)	W	Rs16 → @(d:16,Rd16)				4						–	–	↕	↕	0	–	6
	MOV.W Rs,@–Rd	W	Rd16–2 → Rd16 Rs16 → @Rd16					2					–	–	↕	↕	0	–	6
	MOV.W Rs, @aa:16	W	Rs16 → @aa:16						4				–	–	↕	↕	0	–	6
	MOV.W #xx:16,Rd	W	#xx:16 → Rd	4									–	–	↕	↕	0	–	4
	POP Rd	W	@SP+ → Rd					2					–	–	↕	↕	0	–	6
	PUSH Rs	W	Rs → @–SP					2					–	–	↕	↕	0	–	6
	MOVFPE @aa:16,Rd	B	@aa:16 → Rd						4				–	–	↕	↕	0	–	⑤
	MOVTPE Rs,@aa:16	B	Rs → @aa:16						4				–	–	↕	↕	0	–	⑤
	EEPMOV	B	if R4L≠0 then Repeat @R5*→ @R6 R5+1 → R5 R6+1 → R6 R4L–1 → R4L Until R4L=0 else next									4	–	–	–	–	–	–	④

	Mnemonic	Operand size	Operation	Addressing mode/ instruction length								Condition code						No. of states
				#xx:	Rn	@Rn	@(d:16,Rn)	@–Rn/@Rn+	@aa:	@(d:8, PC)	@@aa	I	H	N	Z	V	C	
Arithmetic instructions	ADD.B #xx:8,Rd	B	Rd8+#xx:8 → Rd8	2								–	↕	↕	↕	↕	↕	2
	ADD.B Rs,Rd	B	Rs8+Rd8 → Rd8		2							–	↕	↕	↕	↕	↕	2
	ADD.W Rs,Rd	W	Rs16+Rd16 → Rd16		2							–	①	↕	↕	↕	↕	2
	ADDX.B #xx:8,Rd	B	Rd8+#xx:8 +C → Rd8	2								–	↕	↕	②	↕	↕	2
	ADDX.B Rs,Rd	B	Rd8+Rs8 +C → Rd8		2							–	↕	↕	②	↕	↕	2
	ADDS.W #1,Rd	W	Rd16+1 → Rd16		2							–	–	–	–	–	–	2
	ADDS.W #2,Rd	W	Rd16+2 → Rd16		2							–	–	–	–	–	–	2
	INC.B Rd	B	Rd8+1 → Rd8		2							–	–	↕	↕	↕	–	2
	DAA.B Rd	B	Rd8 decimal adjust → Rd8		2							–	*	↕	↕	*	③	2
	NEG.B Rd	B	0–Rd → Rd		2							–	↕	↕	↕	↕	↕	2
	SUB.B Rs,Rd	B	Rd8–Rs8 → Rd8		2							–	↕	↕	↕	↕	↕	2
	SUB.W Rs,Rd	W	Rd16–Rs16 → Rd16		2							–	①	↕	↕	↕	↕	2
	SUBX.B #xx:8,Rd	B	Rd8–#xx:8 –C → Rd8	2								–	↕	↕	②	↕	↕	2
	SUBX.B Rs,Rd	B	Rd8–Rs8 → Rd8		2							–	↕	↕	②	↕	↕	2
	SUBS.W #1,Rd	W	Rd16–1 → Rd16		2							–	–	–	–	–	–	2
	SUBS.W #2,Rd	W	Rd16–2 → Rd16		2							–	–	–	–	–	–	2
	DEC.B Rd	B	Rd8–1 → Rd8		2							–	–	↕	↕	↕	–	2
	DAS.B Rd	B	Rd8 decimal adjust → Rd8		2							–	*	↕	↕	*	–	2
	CMP.B #xx:8,Rd	B	Rd8–#xx:8	2								–	↕	↕	↕	↕	↕	2
	CMP.B Rs,Rd	B	Rd8–Rs8		2							–	↕	↕	↕	↕	↕	2
	CMP.W Rs,Rd	W	Rd16–Rs16		2							–	①	↕	↕	↕	↕	2
	MULXU.B Rs,Rd	B	Rd8×Rs8 → Rd16		2							–	–	–	–	–	–	14
	DIVXU.B Rs,Rd	B	Rd16÷Rs8 → Rd16 (RdH:remainder,RdL:quotient)		2							–	–	↕	↕	–	–	14
Logic instructions	AND.B #xx:8,Rd	B	Rd8∧#xx:8 → Rd8	2								–	–	↕	↕	0	–	2
	AND.B Rs,Rd	B	Rd8∧Rs8 → Rd8		2							–	–	↕	↕	0	–	2
	OR.B #xx:8,Rd	B	Rd8∨#xx:8 → Rd8	2								–	–	↕	↕	0	–	2
	OR.B Rs,Rd	B	Rd8∨Rs8 → Rd8		2							–	–	↕	↕	0	–	2
	XOR.B #xx:8,Rd	B	Rd8⊕#xx:8 → Rd8	2								–	–	↕	↕	0	–	2
	XOR.B Rs,Rd	B	Rd8⊕Rs8 → Rd8		2							–	–	↕	↕	0	–	2
	NOT.B Rd	B	$\overline{Rd}$ → Rd		2							–	–	↕	↕	0	–	2

	Mnemonic	Operand size	Operation	Addressing mode/ instruction length								Condition code						No. of states
				#xx:	Rn	@Rn	@(d:16,Rn)	@–Rn/@Rn+	@aa:	@(d:8, PC)	@@aa	I	H	N	Z	V	C	
Shift instructions	SHAL.B Rd	B			2							–	–	↕	↕	↕	↕	2
	SHAR.B Rd	B			2							–	–	↕	↕	0	↕	2
	SHLL.B Rd	B			2							–	–	↕	↕	0	↕	2
	SHLR.B Rd	B			2							–	–	0	↕	0	↕	2
	ROTXL.B Rd	B			2							–	–	↕	↕	0	↕	2
	ROTXR.B Rd	B			2							–	–	↕	↕	0	↕	2
	ROTL.B Rd	B			2								–	↕	↕	0	↕	2
	ROTR.B Rd	B			2							–	–	↕	↕	0	↕	2
Bit manipulation instructions	BSET #xx:3,Rd	B	(#xx:3 of Rd8) ← 1		2							–	–	–	–	–	–	2
	BSET #xx:3,@Rd	B	(#xx:3 of @Rd16) ← 1			4						–	–	–	–	–	–	8
	BSET #xx:3,@aa:8	B	(#xx:3 of @aa:8) ← 1						4			–	–	–	–	–	–	8
	BSET Rn,Rd	B	(Rn8 of Rd8) ← 1		2							–	–	–	–	–	–	2
	BSET Rn,@Rd	B	(Rn8 of @Rd16) ← 1			4						–	–	–	–	–	–	8
	BSET Rn,@aa:8	B	(Rn8 of @aa:8) ← 1						4			–	–	–	–	–	–	8
	BCLR #xx:3,Rd	B	(#xx:3 of Rd8) ← 0		2							–	–	–	–	–	–	2
	BCLR #xx:3,@Rd	B	(#xx:3 of @Rd16) ← 0			4						–	–	–	–	–	–	8
	BCLR #xx:3,@aa:8	B	(#xx:3 of @aa:8) ← 0						4			–	–	–	–	–	–	8
	BCLR Rn,Rd	B	(Rn8 of Rd8) ← 0		2							–	–	–	–	–	–	2
	BCLR Rn,@Rd	B	(Rn8 of @Rd16) ← 0			4						–	–	–	–	–	–	8
	BCLR Rn,@aa:8	B	(Rn8 of @aa:8) ← 0						4			–	–	–	–	–	–	8
	BNOT #xx:3,Rd	B	(#xx:3 of Rd8) ← ($\overline{\text{#xx:3 of Rd8}}$)		2							–	–	–	–	–	–	2
	BNOT #xx:3,@Rd	B	(#xx:3 of @Rd16) ← ($\overline{\text{#xx:3 of @Rd16}}$)			4						–	–	–	–	–	–	8
	BNOT #xx:3,@aa:8	B	(#xx:3 of @aa:8) ← ($\overline{\text{#xx:3 of @aa:8}}$)						4			–	–	–	–	–	–	8

	Mnemonic	Operand size	Operation	Addressing mode/ instruction length								Condition code						No. of states
				#xx:	Rn	@Rn	@(d:16,Rn)	@–Rn/@Rn+	@aa:	@(d:8, PC)	@@aa	I	H	N	Z	V	C	
Bit manipulation instructions	BNOT Rn,Rd	B	(Rn8 of Rd8) ← $\overline{\text{(Rn8 of Rd8)}}$		2							–	–	–	–	–	–	2
	BNOT Rn,@Rd	B	(Rn8 of @Rd16) ← $\overline{\text{(Rn8 of @Rd16)}}$			4						–	–	–	–	–	–	8
	BNOT Rn,@aa:8	B	(Rn8 of @aa:8) ← $\overline{\text{(Rn8 of @aa:8)}}$						4			–	–	–	–	–	–	8
	BTST #xx:3,Rd	B	$\overline{\text{(\#xx:3 of Rd8)}}$ → Z		2							–	–	–	↕	–	–	2
	BTST #xx:3,@Rd	B	$\overline{\text{(\#xx:3 of @Rd16)}}$ → Z			4						–	–	–	↕	–	–	6
	BTST #xx:3,@aa:8	B	$\overline{\text{(\#xx:3 of @aa:8)}}$ → Z						4			–	–	–	↕	–	–	6
	BTST Rn,Rd	B	$\overline{\text{(Rn8 of Rd8)}}$ → Z		2							–	–	–	↕	–	–	2
	BTST Rn,@Rd	B	$\overline{\text{(Rn8 of @Rd16)}}$ → Z			4						–	–	–	↕	–	–	6
	BTST Rn,@aa:8	B	$\overline{\text{(Rn8 of @aa:8)}}$ → Z						4			–	–	–	↕	–	–	6
	BLD #xx:3,Rd	B	(#xx:3 of Rd8) → C		2							–	–	–	–	–	↕	2
	BLD #xx:3,@Rd	B	(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
	BLD #xx:3,@aa:8	B	(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
	BILD #xx:3,Rd	B	$\overline{\text{(\#xx:3 of Rd8)}}$ → C		2							–	–	–	–	–	↕	2
	BILD #xx:3,@Rd	B	$\overline{\text{(\#xx:3 of @Rd16)}}$ → C			4						–	–	–	–	–	↕	6
	BILD #xx:3,@aa:8	B	$\overline{\text{(\#xx:3 of @aa:8)}}$ → C						4			–	–	–	–	–	↕	6
	BST #xx:3,Rd	B	C → (#xx:3 of Rd8)		2							–	–	–	–	–	–	2
	BST #xx:3,@Rd	B	C → (#xx:3 of Rd16)			4						–	–	–	–	–	–	8
	BST #xx:3,@aa:8	B	C → (#xx:3 of @aa:8)						4			–	–	–	–	–	–	8
	BIST #xx:3,Rd	B	$\overline{C}$ → (#xx:3 of Rd8)		2							–	–	–	–	–	–	2
	BIST #xx:3,@Rd	B	$\overline{C}$ → (#xx:3 of Rd16)			4						–	–	–	–	–	–	8
	BIST #xx:3,@aa:8	B	$\overline{C}$ → (#xx:3 of @aa:8)						4			–	–	–	–	–	–	8
	BAND #xx:3,Rd	B	C∧(#xx:3 of Rd8) → C		2							–	–	–	–	–	↕	2
	BAND #xx:3,@Rd	B	C∧(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
	BAND #xx:3,@aa:8	B	C∧(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
	BIAND #xx:3,Rd	B	C∧$\overline{\text{(\#xx:3 of Rd8)}}$ → C		2							–	–	–	–	–	↕	2
	BIAND #xx:3,@Rd	B	C∧$\overline{\text{(\#xx:3 of @Rd16)}}$ → C			4						–	–	–	–	–	↕	6
	BIAND #xx:3, @aa:8	B	C∧$\overline{\text{(\#xx:3 of @aa:8)}}$ → C						4			–	–	–	–	–	↕	6
	BOR #xx:3,Rd	B	C∨(#xx:3 of Rd8) → C		2							–	–	–	–	–	↕	2
	BOR #xx:3,@Rd	B	C∨(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
	BOR #xx:3, @aa:8	B	C∨(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
	BIOR #xx:3,@Rd	B	C∨$\overline{\text{(\#xx:3 of Rd8)}}$ → C		2							–	–	–	–	–	↕	2

	Mnemonic	Operand size	Operation	Addressing mode/ instruction length								Condition code						No. of states
				#xx:	Rn	@Rn	@(d:16,Rn)	@–Rn/@Rn+	@aa:	@(d:8, PC)	@@aa	I	H	N	Z	V	C	
Bit manipulation instructions	BIOR #xx:3,@Rd	B	C∨(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
	BIOR #xx:3, @aa:8	B	C∨(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
	BXOR #xx:3,Rd	B	C⊕(#xx:3 of Rd8) → C		2							–	–	–	–	–	↕	2
	BXOR #xx:3,@Rd	B	C⊕(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
	BXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
	BIXOR #xx:3,Rd	B	C⊕(#xx:3 of Rd8) → C		2							–	–	–	–	–	↕	2
	BIXOR #xx:3,@Rd	B	C⊕(#xx:3 of @Rd16) → C			4						–	–	–	–	–	↕	6
	BIXOR #xx:3, @aa:8	B	C⊕(#xx:3 of @aa:8) → C						4			–	–	–	–	–	↕	6
Branching instructions	BRA(BT)	–	PC ← PC+d:8							2		–	–	–	–	–	–	4
	BRN(BF)	–	No operation							2		–	–	–	–	–	–	4
	BHI	–	if true then							2		–	–	–	–	–	–	4
	BLS	–	PC ← PC+d:8							2		–	–	–	–	–	–	4
	BCC(BHS)	–	else next							2		–	–	–	–	–	–	4
	BCS(BLO)	–								2		–	–	–	–	–	–	4
	BNE	–								2		–	–	–	–	–	–	4
	BEQ	–								2		–	–	–	–	–	–	4
	BVC	–								2		–	–	–	–	–	–	4
	BVS	–								2		–	–	–	–	–	–	4
	BPL	–								2		–	–	–	–	–	–	4
	BMI	–								2		–	–	–	–	–	–	4
	BGE	–								2		–	–	–	–	–	–	4
	BLT	–								2		–	–	–	–	–	–	4
	BGT	–								2		–	–	–	–	–	–	4
	BLE	–								2		–	–	–	–	–	–	4
	JMP Rn.	–	PC ← Rn16		2							–	–	–	–	–	–	4
	JMP @aa:16	–	PC ← aa:16						4			–	–	–	–	–	–	6
	JMP @@aa:8	–	PC ← @aa:8								2	–	–	–	–	–	–	8
	BSR	–	SP–2 → SP PC → @SP PC ← PC+d:8							2		–	–	–	–	–	–	6

	Mnemonic	Operand size	Operation	Addressing mode/ instruction length									Condition code						No. of states
				#xx:	Rn	@Rn	@(d:16,Rn)	@–Rn/@Rn+	@aa:	@(d:8, PC)	@@aa	Implied	I	H	N	Z	V	C	
Branching instructions	JSR @Rn	–	SP–2 → SP PC → @SP PC ← Rn16			2							–	–	–	–	–	–	6
	JSR @aa:16	–	SP–2 → SP PC → @SP PC ← @aa:16						4				–	–	–	–	–	–	8
	JSR @@aa:8		SP–2 → SP PC → @SP PC ← @aa:8								2		–	–	–	–	–	–	8
	RTS	–	PC ← @SP SP+2 → SP									2	–	–	–	–	–	–	8
System control instructions	RTE	–	CCR ← @SP SP+2 → SP PC ← @SP SP+2 → SP									2	↕	↕	↕	↕	↕	↕	10
	SLEEP	–	Transit to sleep mode.									2	–	–	–	–	–	–	2
	LDC #xx:8,CCR	B	#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
	LDC Rs,CCR	B	Rs8 → CCR		2								↕	↕	↕	↕	↕	↕	2
	STC CCR,Rd	B	CCR → Rd8		2								–	–	–	–	–	–	2
	ANDC #xx:8,CCR	B	CCR∧#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
	ORC #xx:8,CCR	B	CCR∨#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
	XORC #xx:8,CCR	B	CCR⊕#xx:8 → CCR	2									↕	↕	↕	↕	↕	↕	2
	NOP	–	No operation									2	–	–	–	–	–	–	2

Notes: The number of states is the number of states required for execution when the instruction and its operands are located in on-chip memory.

① Set to "1" when there is a carry or borrow from bit 11; otherwise cleared to "0."

② If the result is zero, the previous value of the flag is retained; otherwise the flag is cleared to "0."

③ Set to "1" if decimal adjustment produces a carry; otherwise cleared to "0."

④ The number of states required for execution is 4n+8 (n = value of R4L)

⑤ These instructions transfer data in synchronization with the E clock. The number of states varies depending on the synchronization delay.

Number of States Required for Instruction Execution: The number of states indicated in the instruction set list applies when the instruction and its operands are located in on-chip memory. If the instruction or an operand must be accessed at an off-chip address or in the on-chip register field, the number of states increases as indicated in the following table.

Access type		Location	Size	Additional states
Operand		Register field	Byte data	1
			Word data	4
		External memory	Byte data	1 + m
			Word data	4 + 2m
Instruction	Non-branching instruction	External memory	2-byte instruction	4 + 2m
			4-byte instruction	8 + 4m
	Branching instruction		2-byte instruction	8 + 4m
			4-byte instruction	8 + 4m

Note: m—number of wait states inserted in access to external device.

Operation Notation

Symbol	Meaning
PC	Program counter
SP	Stack pointer (R7)
CCR	Condition code register
Z	Zero flag in CCR
C	Carry flag in CCR
Rs, Rd, Rn	General registers (8-bit—R0H/R0L to R7H/R7L; 16-bit—R0 to R7)
d:8, d:16	8-Bit or 16-bit displacement
#xx:3, #xx:8, #xx:16	3-Bit, 8-bit or 16-bit immediate data
→	The result of the operation on the left is assigned to the operand on the right.
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	AND logical
∨	OR logical
⊕	Exclusive OR logical
¬	Not
( ) and < >	Contents of effective address

Condition Code Notation

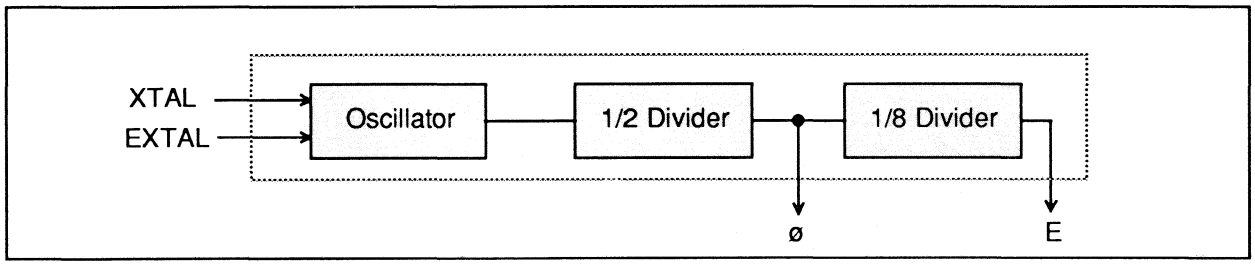
Symbol	Meaning
↕	The flag is altered according to the result of the instruction.
*	Undetermined; the flag is left in an unpredictable state.
0	The flag is cleared to "0."
—	The flag is not changed.

4.6 Bus Timing

- **System Clock Timing**

The system clock (ø) is generated from an external clock signal input at the EXTAL pin, or by connecting a crystal oscillator across the XTAL and EXTAL pins. A divider divides the input frequency by 2, so the input frequency should be twice the desired system clock rate.

The E (Enable) clock is created by dividing the system clock (ø) by 8.

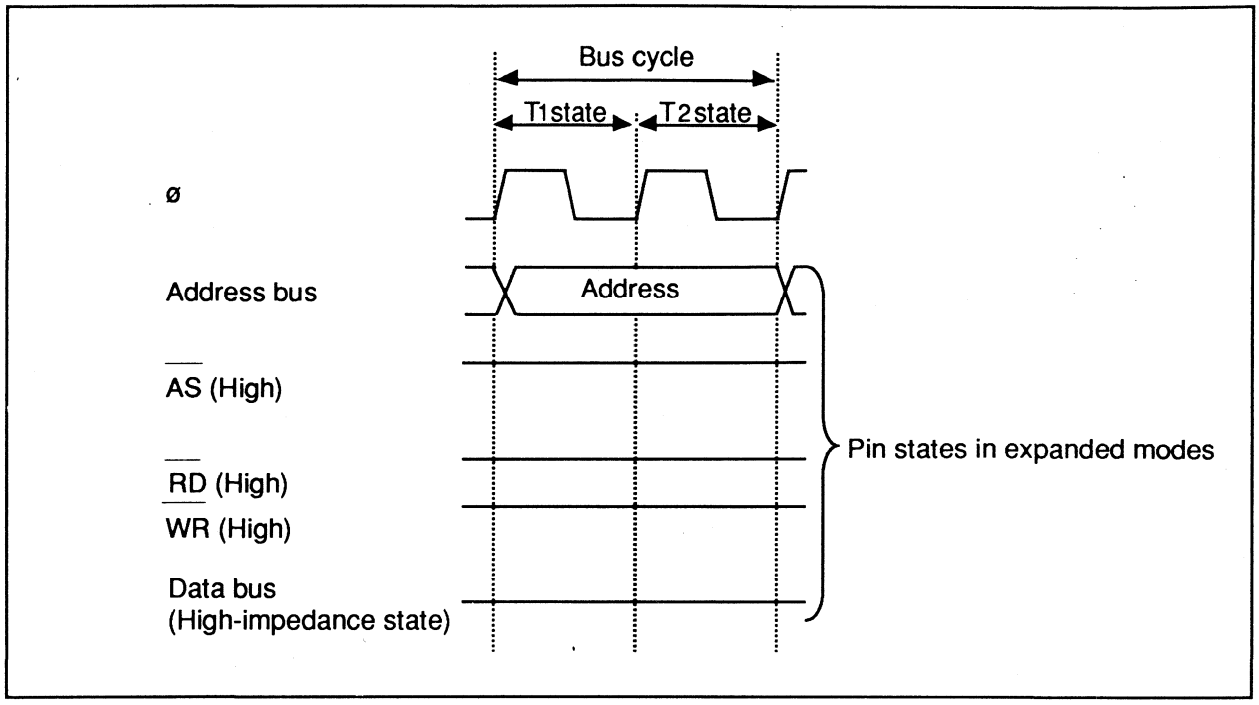


Clock Diagram of Clock Pulse Generator

- **CPU Read/Write Cycle**

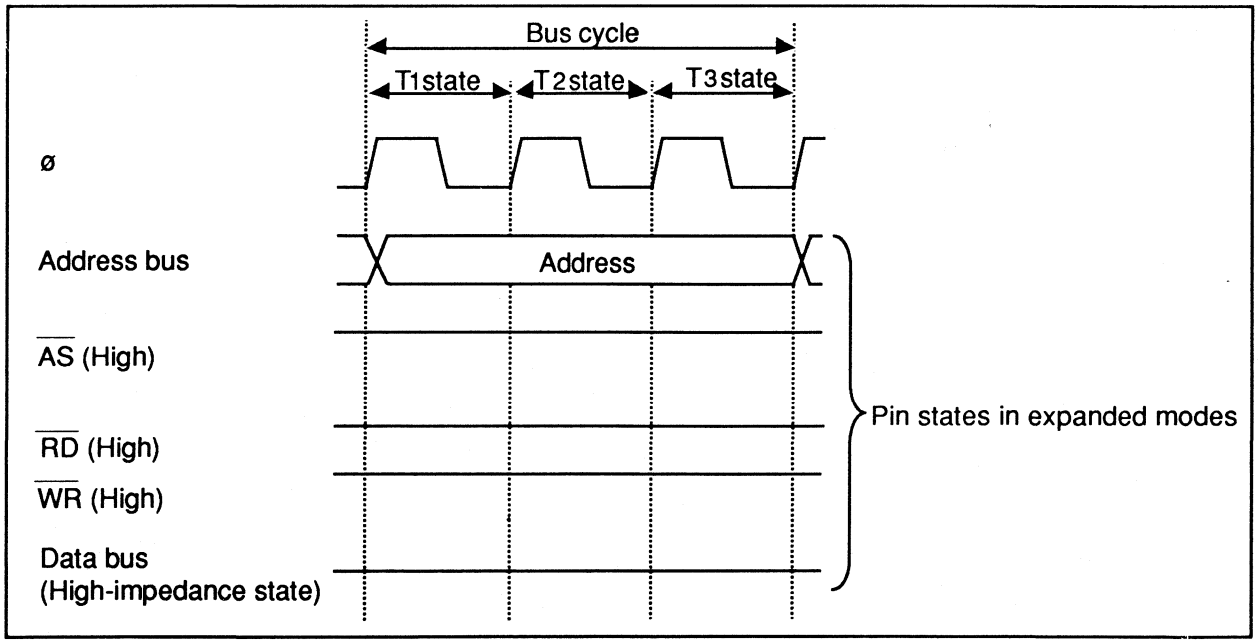
The CPU operates on the ø clock. One period of the ø clock is called a "state." The structure of the bus cycle depends on whether the access is to on-chip memory, the on-chip register field, or an external address. The basic bus cycle length is two or three states.

- **On-Chip Memory Access Timing (RAM, ROM):** On-chip memory is accessed in two states. The data bus is 16 bits wide, permitting either byte access or word access.

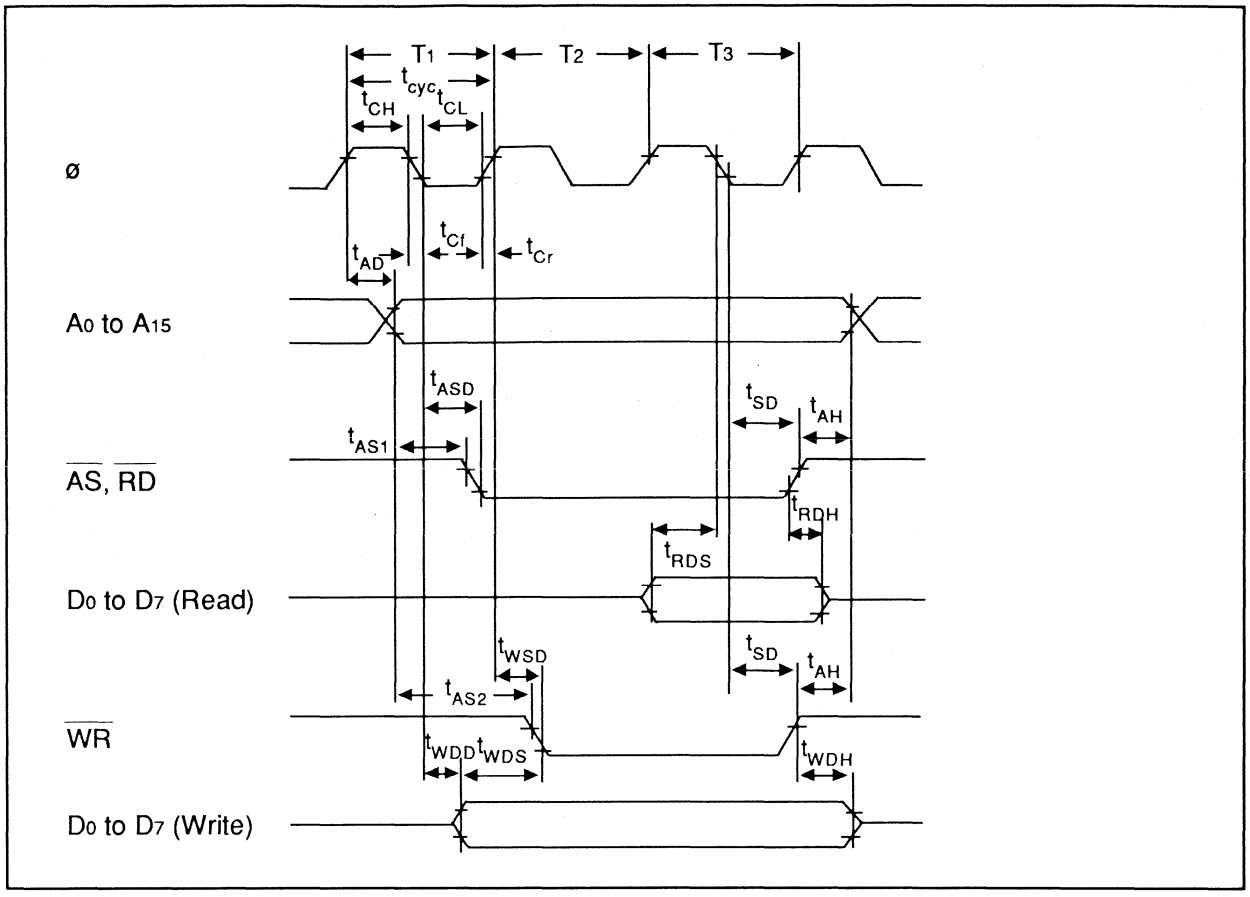


Access Timing for On-Chip Memory

• **Access Timing for On-Chip Register Field and External Addresses:** In both cases the access is performed in three states and the data bus is 8 bits wide. Word data and instruction codes are accessed by two consecutive byte accesses.



Access Timing for On-Chip Register Field



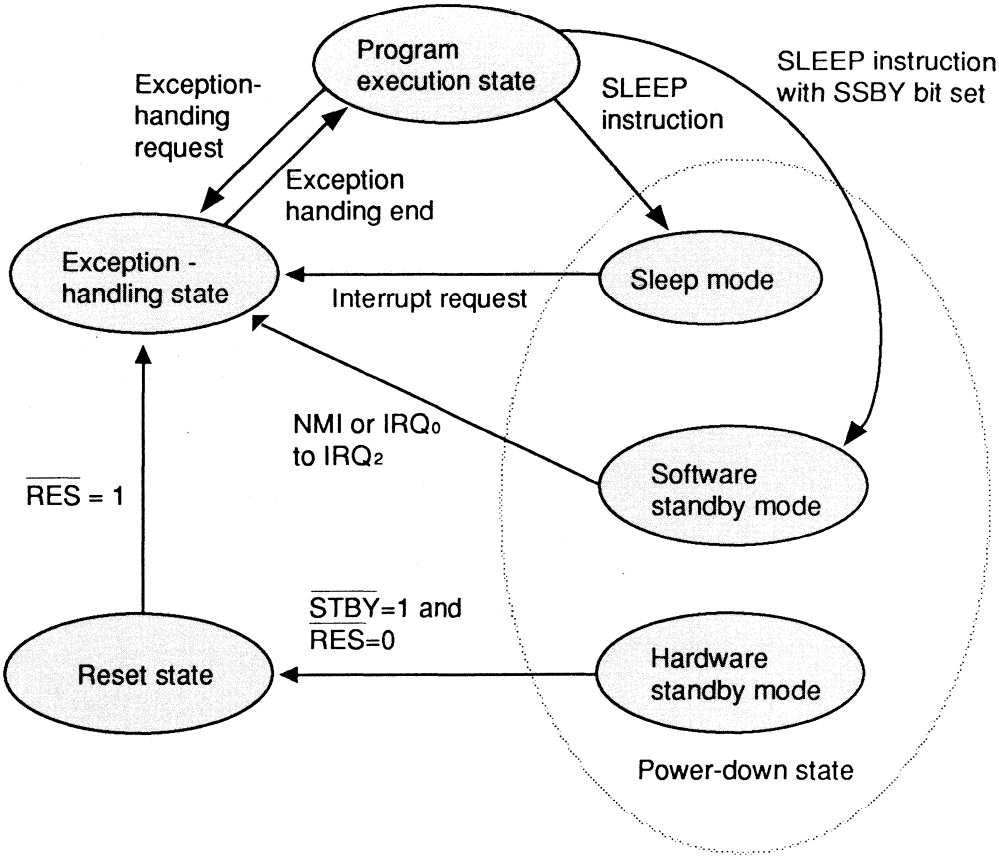
External Device Access Cycle

— Preliminary specifications —

Item	Symbol	6MHz min.	6MHz max.	8MHz min.	8MHz max.	10MHz min.	10MHz max.	Unit
Clock cycle time	t_{cyc}	166.7	2000	125	2000	100	2000	ns
Clock pulse width Low	t_{CL}	65	—	45	—	35	—	ns
Clock pulse width High	t_{CH}	65	—	45	—	35	—	ns
Clock rise time	t_{Cr}	—	15	—	15	—	15	ns
Clock fall time	t_{Cf}	—	15	—	15	—	15	ns
Address delay time	t_{AD}	—	70	—	60	—	55	ns
Address hold time	t_{AH}	30	—	25	—	20	—	ns
Address strobe delay time	t_{ASD}	—	70	—	60	—	50	ns
Write strobe delay time	t_{WSD}	—	70	—	60	—	50	ns
Strobe delay time	t_{SD}	—	70	—	60	—	50	ns
Address setup time 1	t_{AS1}	25	—	20	—	15	—	ns
Address setup time 2	t_{AS2}	105	—	80	—	65	—	ns
Read data setup time	t_{RDS}	60	—	50	—	50	—	ns
Read data hold time	t_{RDH}	0	—	0	—	0	—	ns
Write data delay time	t_{WDD}	—	70	—	60	—	60	ns
Write data setup time	t_{WDS}	30	—	15	—	10	—	ns
Write data hold time	t_{WDH}	30	—	25	—	20	—	ns

4.7 Operating States

The CPU operates in three states: the program execution state, exception-handling state, and power-down state. The following diagram shows the state transitions.



Notes:

1. A transition to the reset state occurs when $\overline{RES}$ goes Low, except when the chip is in the hardware standby mode.
2. A transition from any state to the hardware standby mode occurs when $\overline{STBY}$ goes Low.

State Transitions

- **Reset State:** In this state the CPU and all on-chip supporting modules are reset.

- **Program Execution State:** In this state the CPU executes instructions in normal sequence.

- **Exception-Handling State:** This is a transient state in which the CPU executes a hardware sequence preparatory to handling a reset or interrupt. For an interrupt, the program counter and condition code register are saved to the location indicated by the stack pointer.

• **Power-Down State:** This state comprises three modes: the sleep mode, software standby mode, and hardware standby mode. The CPU halts to conserve power. In the standby modes the clock also stops. See section 7, "Power-Down Modes" for details.

4.8 Exception Handling

There are two types of exceptions: interrupts and the reset. When an exception occurs, the CPU pushes the program counter (PC) and condition code register (CCR) onto the stack (in the case of an interrupt), then sets the interrupt mask bit in the CCR to "1," fetches the address of the reset or interrupt-handling routine from the vector table, and branches to that address.

The reset exception has the highest priority. The interrupts have fixed priorities (see the interrupt vector table) which determine the order in which they are handled when two or more interrupts are requested simultaneously.

Priority	Exception type	When detected	How requested
1	Reset	Each clock period	When $\overline{RES}$ changes from Low to High
2	Interrupt	At end of instruction execution*	Requested by 9 external and 19 on-chip interrupt sources

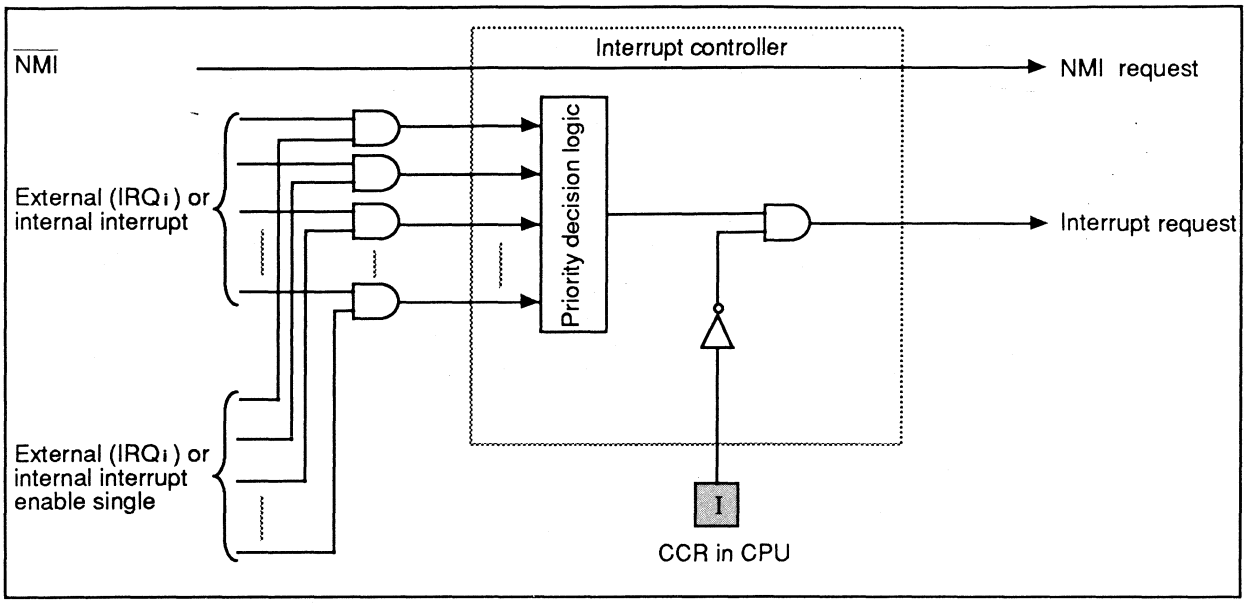
* Not detected after the ANDC, ORC, XORC, and LDC instructions.

4.9 Interrupts

There are 28 types of interrupts: 9 requested by inputs at the external interrupt pins ($\overline{NMI}$, $\overline{IRQ_0}$ to $\overline{IRQ_7}$), and 19 requested from the on-chip supporting modules. NMI has the highest priority and is always accepted. The other external interrupts and the internal interrupts can be masked by the I bit in the CCR. In other words, the I bit masks all interrupts except NMI. All interrupts are individually vectored.

Interrupts are controlled by an on-chip interrupt controller. When two or more interrupts are requested at the same time, the interrupt controller selects the one with the highest priority and leaves the others pending. When interrupted, the CPU stores the PC and CCR contents at the location indicated by the stack pointer, then fetches the address of the interrupt-handling routine from the vector table and begins executing the interrupt-handling routine.

- **Interrupt Controller**



Interrupt Controller Block Diagram

• **Interrupt Vector Table**

Priority	Interrupt	Source	Vector No.	Address of vector
High	NMI	External	3	H'0006
↑	IRQ_0	interrupts	4	H'0008
	IRQ_1		5	H'000A
	IRQ_2		6	H'000C
	IRQ_3		7	H'000E
	IRQ_4		8	H'0010
	IRQ_5		9	H'0012
	IRQ_6		10	H'0014
	IRQ_7		11	H'0016
	ICIA (Input capture A)	16-Bit	12	H'0018
	ICIB (Input capture B)	Free-running	13	H'001A
	ICIC (Input capture C)	timer	14	H'001C
	ICID (Input capture D)		15	H'001E
	OCIA (Output compare A)		16	H'0020
	OCIB (Output compare B)		17	H'0022
	FOVI (Overflow)		18	H'0024
	CMIA0 (Compare-match A)	8-Bit timer 0	19	H'0026
	CMIB0 (Compare-match B)		20	H'0028
	OVI0 (Overflow)		21	H'002A
	CMIA1 (Compare-match A)	8-Bit timer 1	22	H'002C
	CMIB1 (Compare-match B)		23	H'002E
	OVI1 (Overflow)		24	H'0030
	MREI (Master read end)	Dual-port	25	H'0032
	MWEI (Master write end)	RAM	26	H'0034
	ERI (Receive error)	Serial	27	H'0036
	RxI (Receive end)	communication	28	H'0038
	TxI (Transmit end)	interface	29	H'003A
Low	ADI (A/D end)	A/D converter	30	H'003C

Notes:

1. H'0000 is the reset vector.
2. H'0002 to H'0005 are reserved for system use and are not available to the user.

Section 5. Operating Modes

The H8/330 has three operating modes. The mode is selected by the input at the mode pins (MD_1 and MD_0) at the instant the chip comes out of the reset state.

5.1 Mode 1 (Expanded Mode with On-Chip ROM Disabled)

This mode permits access to external memory and peripheral devices. Ports 1 and 2 are used as the address bus and port 3 as the data bus. Ports 8 and 9 are used in part for control signals.

The on-chip ROM is disabled. The corresponding addresses are mapped onto external memory.

The total address space, including on-chip and external addresses, is 64K bytes.

5.2 Mode 2 (Expanded Mode with On-Chip ROM Enabled)

Like mode 1, this mode permits access to external memory and peripheral devices, but when the chip comes out of reset, ports 1 and 2 are assigned as input ports. Software can select which pins of these ports to use for address output by setting bits in their data direction registers. Port 3 is used as the data bus, and ports 8 and 9 are used in part for control signals. The on-chip ROM is enabled.

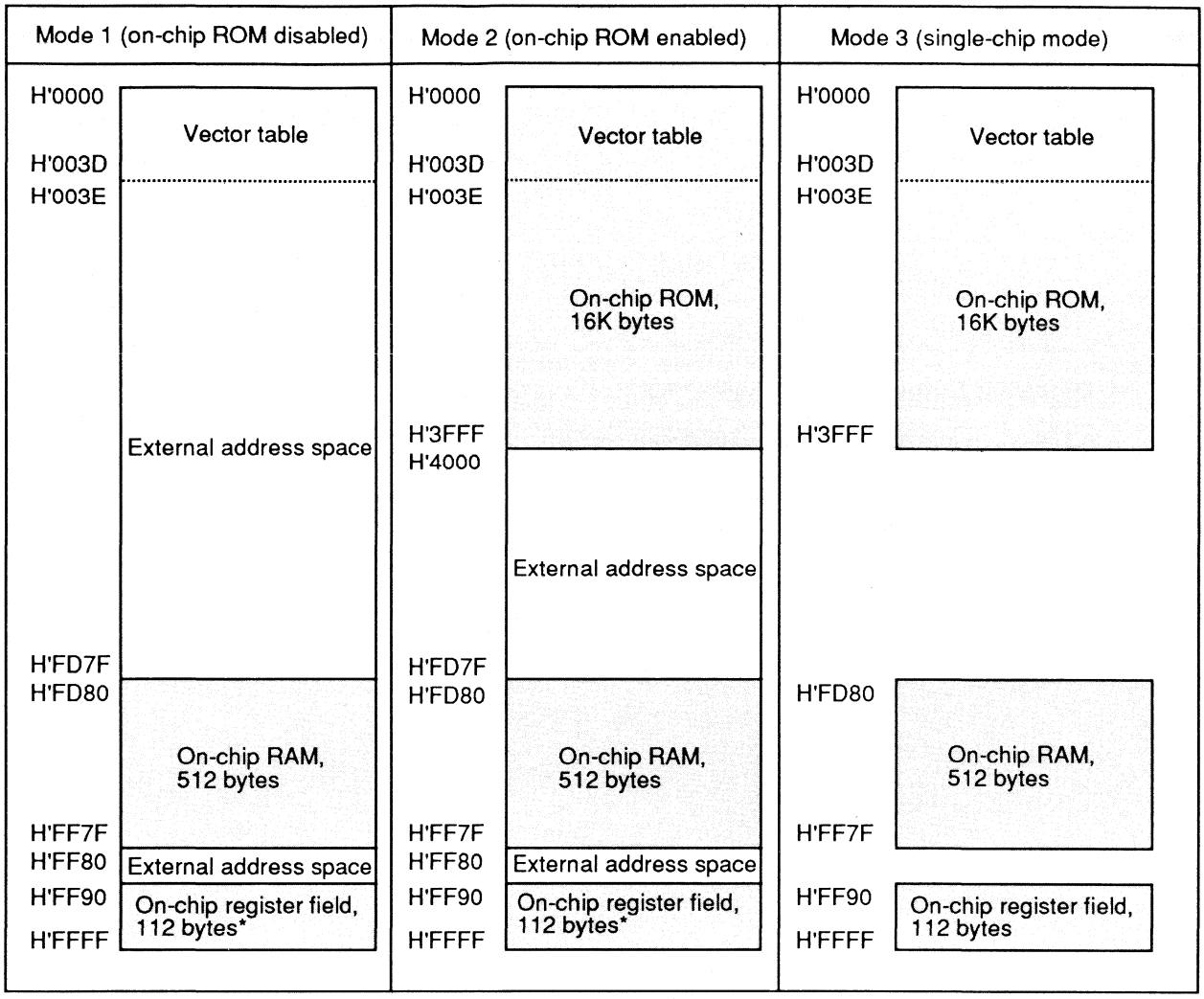
The total address space, including on-chip and external addresses, is 64K bytes.

5.3 Mode 3 (Single-Chip Mode)

In this mode the external address space cannot be used. The H8/330 operates with only its on-chip ROM and RAM and on-chip register field (including the dual-port RAM, timer registers, I/O port registers, etc.). All I/O ports are available for general-purpose input and output.

When the dual-port RAM is enabled, it can be accessed by an external CPU.

- **Memory Map in Each Mode**



* H'FFA8 to H'FFAF are external addresses

Section 6. On-Chip Supporting Functions

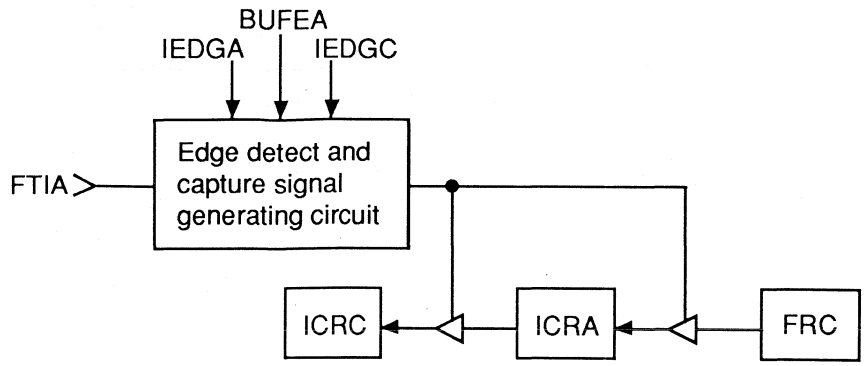
6.1 16-Bit Free-Running Timer

The H8/330 chip includes a single-channel free-running timer module (FRT). Using a 16-bit counter as a time base, the FRT can provide two independent waveform outputs, and can measure the width of input pulse signals.

- **Features**

- Selection of four counter clock sources
  - — Three internal clock sources (ø/2, ø/8, ø/32)
  - — External clock source
- Two independent comparators, for output of two independent waveforms
- Input capture function
  - — Selection of rising or falling edge
  - — Four independent input capture registers, with buffer mode*
- Free-running counter (FRC) can be cleared by compare-match A
- Seven independent interrupts
  - — Two compare-match interrupts
  - — Four input capture interrupts
  - — One overflow interrupt

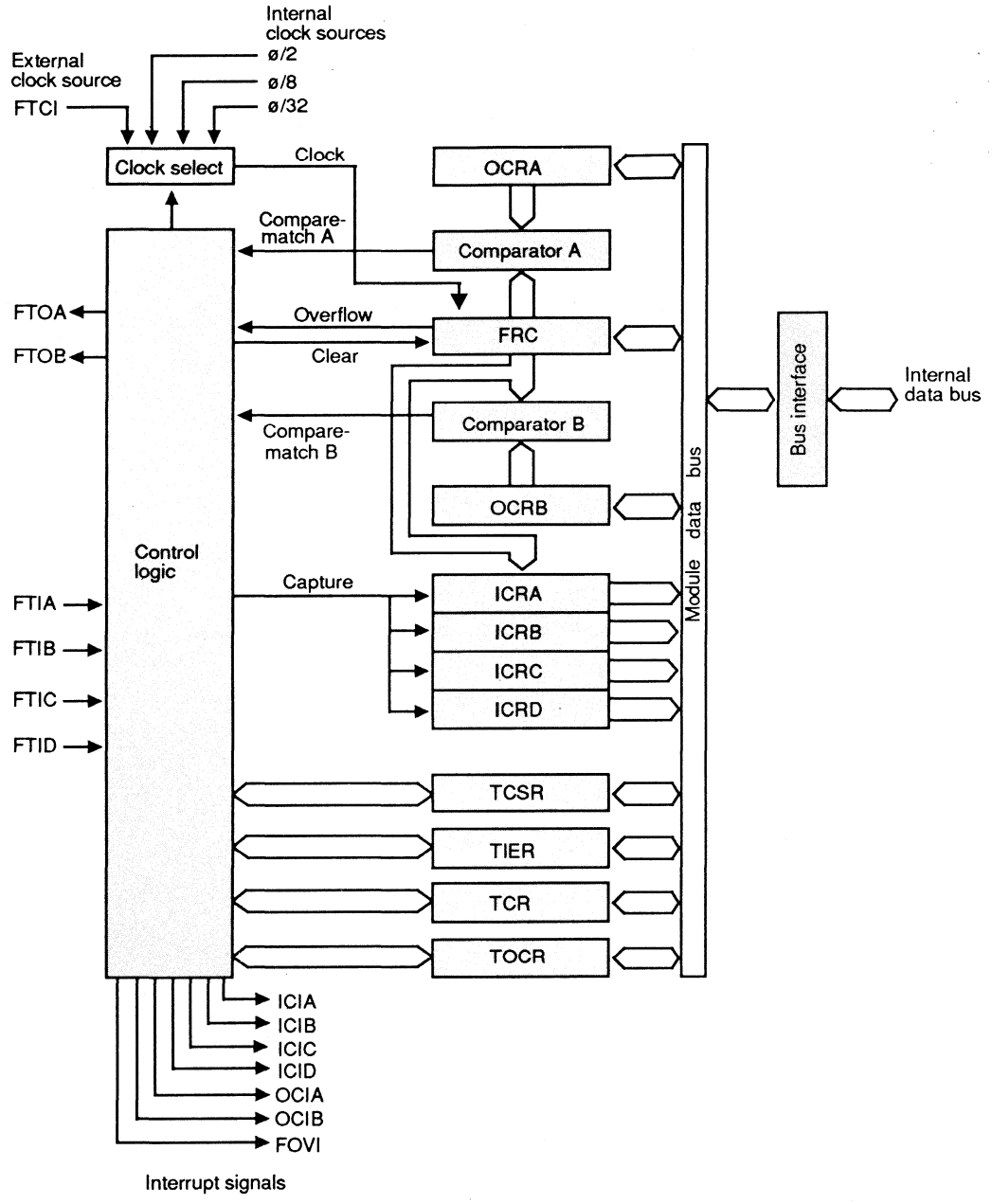
* In the buffer mode, the four 16-bit input capture registers are used in two pairs. In each pair, one of the registers is used as a buffer for the other.



BUFEA: Buffer Enable A
IEDGA: Input Edge Select A
IEDGC: Input Edge Select C
ICRC: Input Capture Register C
ICRA: Input Capture Register A
FRC: Free-Running Counter

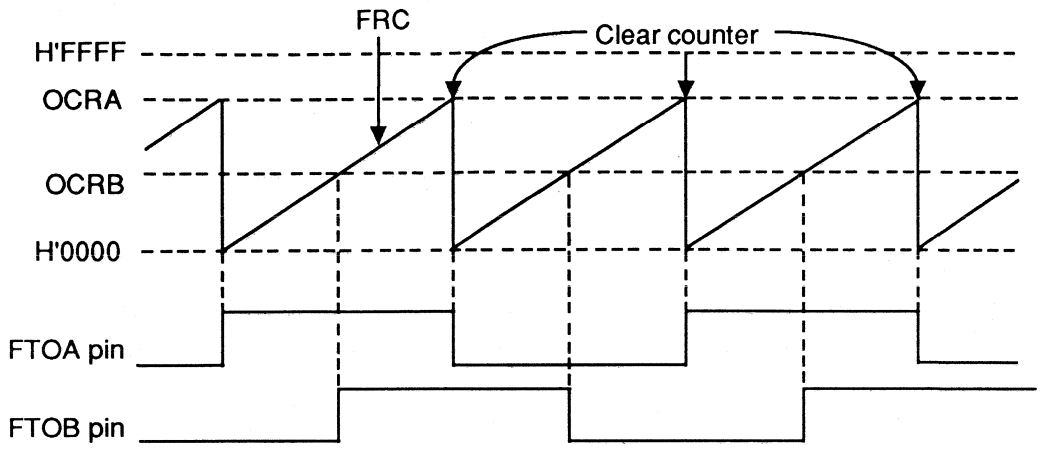
Buffer Operation

- **Block Diagram**



FRC:	Free-Running Counter (16 bits)
OCRA, B:	Output Compare Register A, B (16 bits)
ICRA, B, C, D:	Input Capture Register A, B, C, D (16 bits)
TCSR:	Timer Control/Status Register (8 bits)
TCR:	Timer Control Register (8 bits)
TIER:	Timer Interrupt Enable Register (8 bits)
TOCR:	Timer Output Compare Register (8 bits)

• **Example of 2-Phase Pulse Output:** An example of the output of two pulse signals with a 50% duty ratio and arbitrary phase difference is shown below.



Note: The counter clear A bit in the TCSR is set to "1." The output level is inverted by software at each compare-match event.

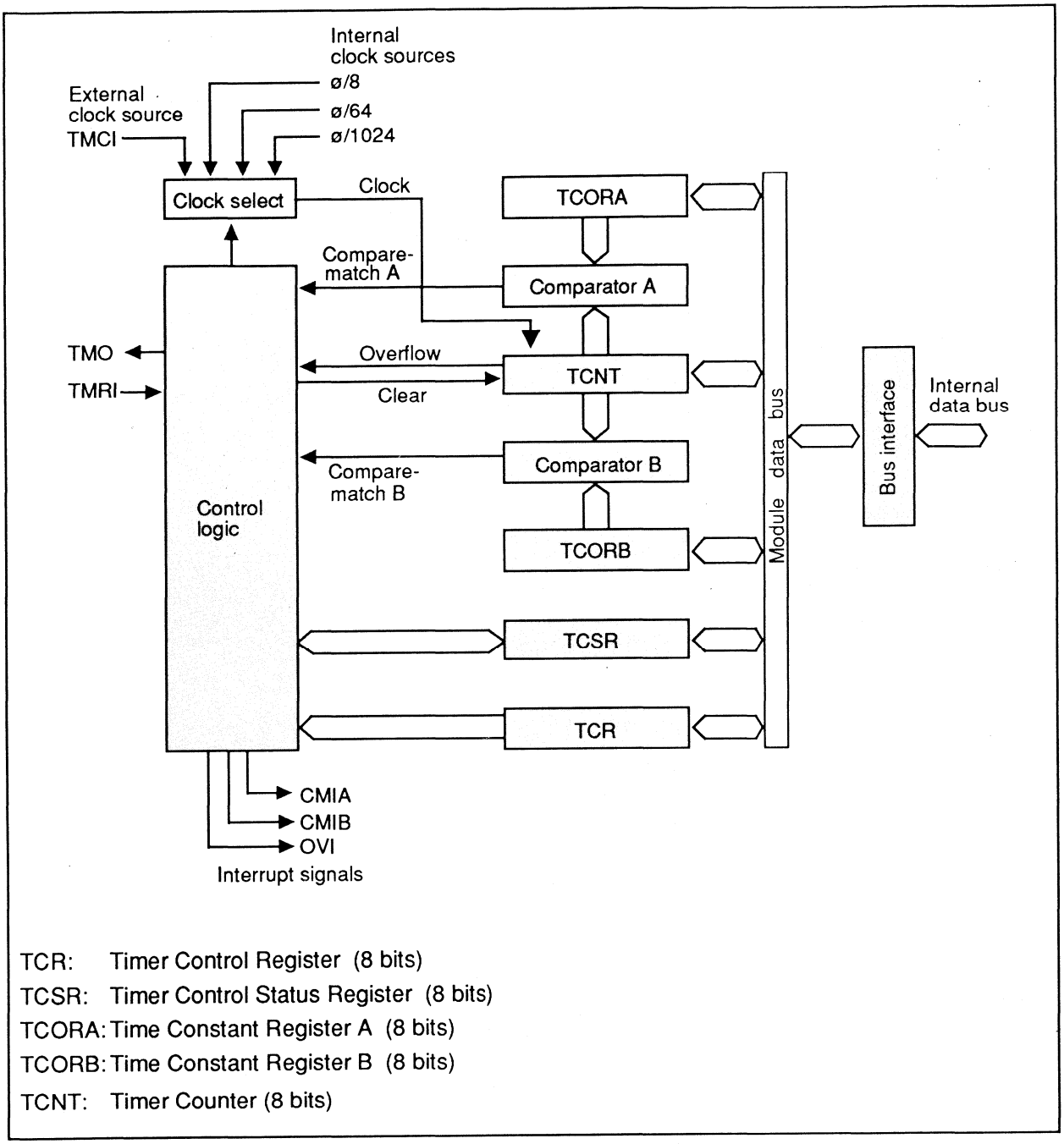
6.2 8-Bit Timer

The H8/330 chip includes a multifunction 8-bit timer module with two independent channels. Each uses an 8-bit counter as a time base. These 8-bit timers can be employed, for example, to generate arbitrary pulse outputs.

• **Features**

- Selection of four counter clock sources
  - — Three internal clock sources (ø/8, ø/64, ø/1024)
  - — External clock source
- Two independent comparators
  - — Two independent compare-match signals can be combined to control the timer output.
- The counter can be cleared by compare-match A or B, or by an external reset signal.
- Three independent interrupts
  - — Two compare-match interrupts
  - — One overflow interrupt

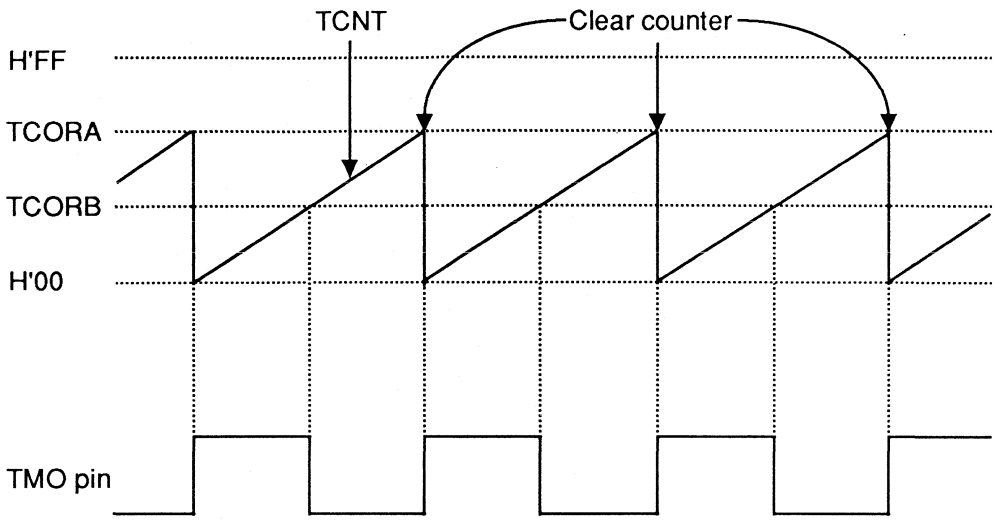
- **Block Diagram**



TCR: Timer Control Register (8 bits)
TCSR: Timer Control Status Register (8 bits)
TCORA: Time Constant Register A (8 bits)
TCORB: Time Constant Register B (8 bits)
TCNT: Timer Counter (8 bits)

Block Diagram of 8-Bit Timer

• **Example of Pulse Output:** An example of the output of a pulse signal with an arbitrary duty cycle is shown below.



Note: A bit in the TCR is set to have the counter cleared when it matches TCORA. TCORA determines the period and TCORB the duty cycle. The pulse signal is output without software intervention.

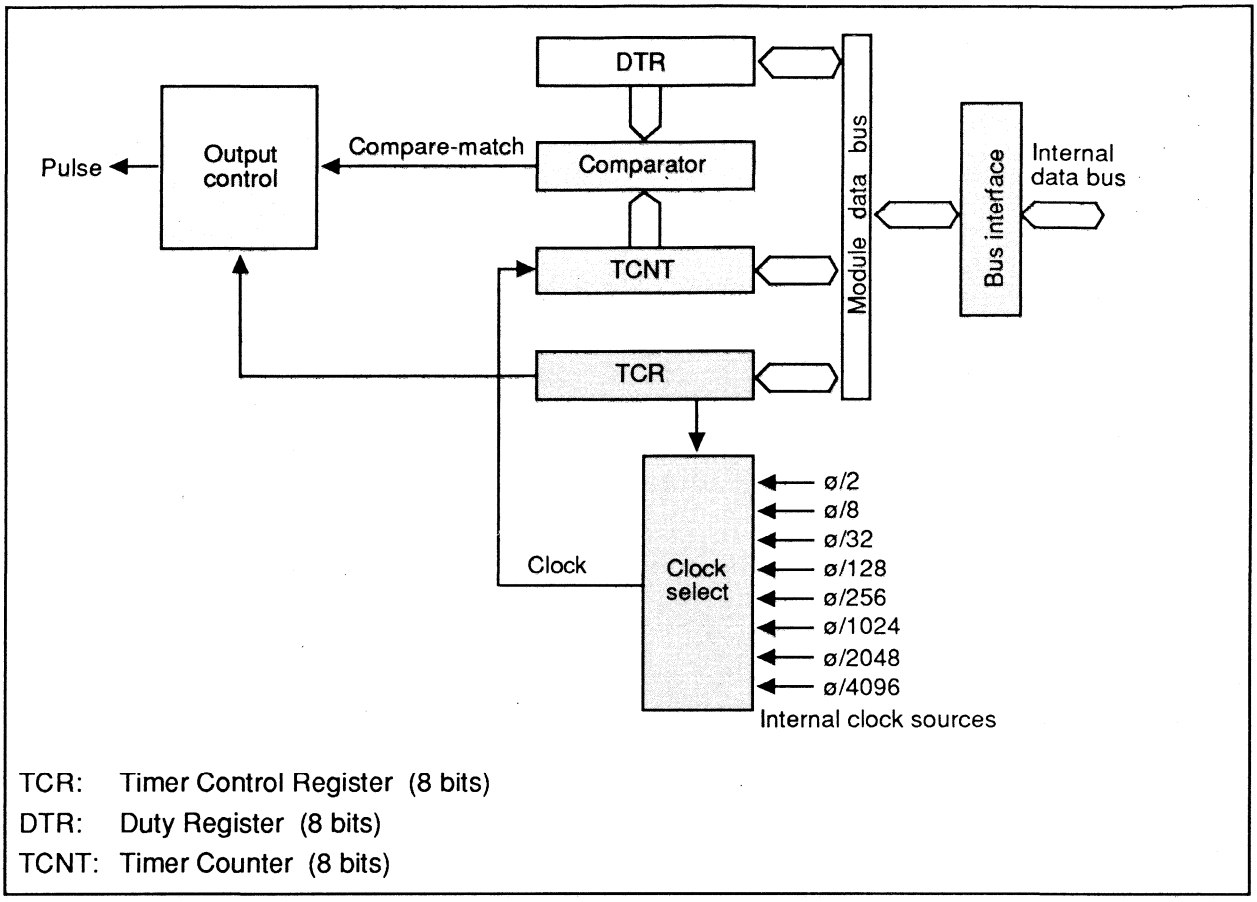
6.3 PWM Timer

The H8/330 has an on-chip PWM (Pulse Width Modulation) timer module with two independent channels. Each includes an 8-bit duty register (DTR) which can be set to give pulses with any duty ratio from 0 to 100%.

• **Features**

- Selection of eight frequencies
- Resolution: 1/250
- Selection of positive or negative output logic

• **Block Diagram**



Block Diagram of PWM Timer

• **PWM Period and Resolution:** The PWM period is specified by setting bits 2 to 0 (CKS2 to CKS0) of the timer control register (TCR). These bits select one of eight internal clock frequencies obtained by dividing the system clock frequency (ø).

			Description		
Bit 2	**Bit 1**	**Bit 0**			**PWM period**
CKS2	**CKS1**	**CKS0**	**Internal clock**	**Resolution**	**(Frequency)**
0	0	0	ø/2	200ns	50µs (20kHz)
0	0	1	ø/8	800ns	200µs (5kHz)
0	1	0	ø/32	3.2µs	800µs (1.25kHz)
0	1	1	ø/128	12.8µs	3.2ms (312.5Hz)
1	0	0	ø/256	25.6µs	6.4ms (156.3Hz)
1	0	1	ø/1024	102.4µs	25.6ms (39.1Hz)
1	1	0	ø/2048	204.8µs	51.2ms (19.5Hz)
1	1	1	ø/4096	409.6µs	102.4ms (9.8Hz)

Note: Values shown are for a 10MHz system clock (ø).

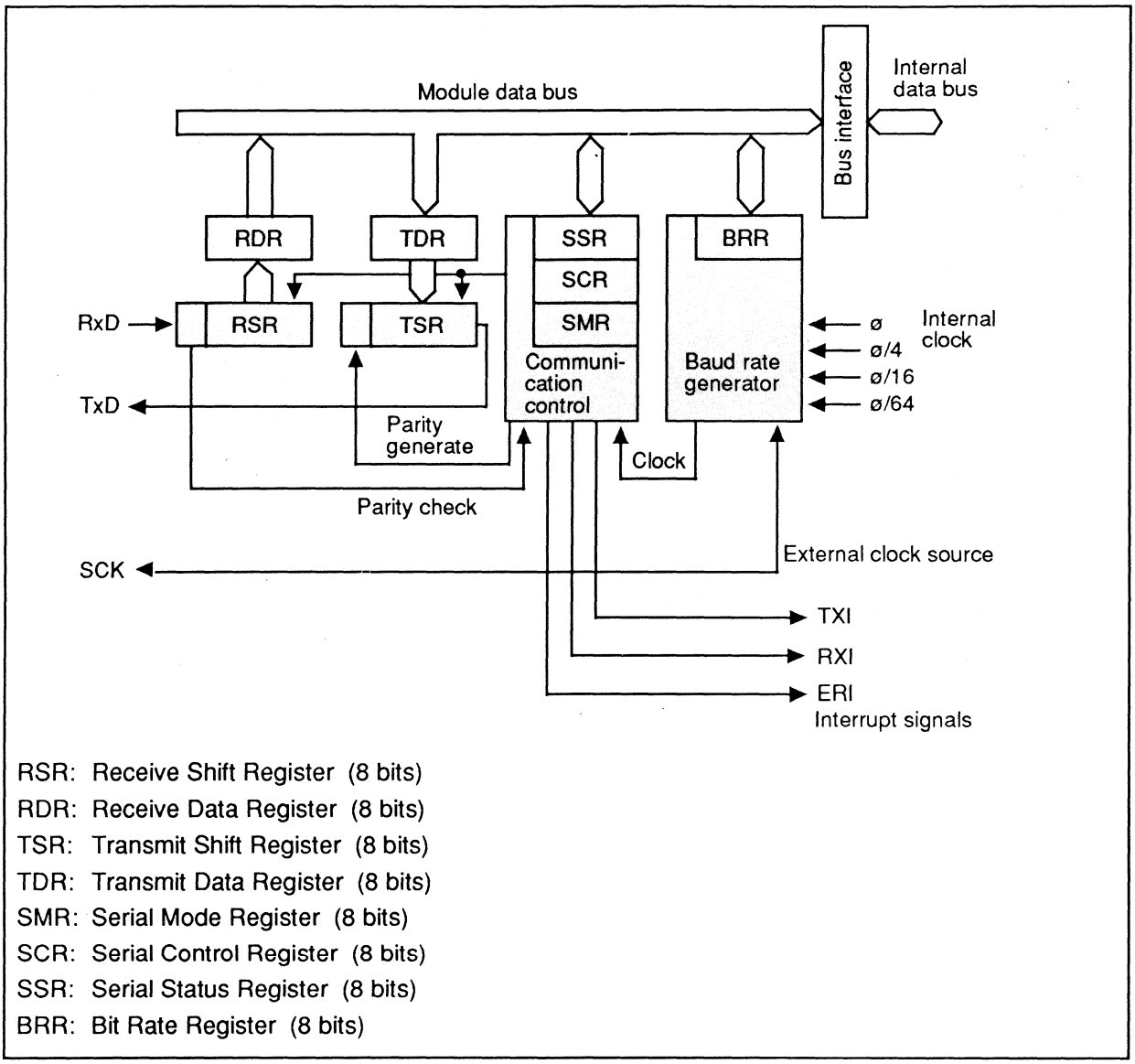
6.4 Serial Communication Interface

The H8/330 chip includes a single-channel serial communication interface (SCI). The SCI is an I/O module that communicates with an external device by asynchronous or clock-synchronized serial data transfer.

- **Features**

- Supports both synchronous and asynchronous communication.
  - — Separate input/output pins for each mode
- Supports full duplex communication.
- Can send and receive data continuously, using double-buffering data registers.
- Built-in baud rate generator can generate any clock rate.
- Selectable clock source: either the built-in baud rate generator or an external clock signal (ASCK or CSCK pin).
- Detects overrun errors, framing errors, and parity errors.
- Three independent interrupts: transmit-end, receive-end, and receive error.

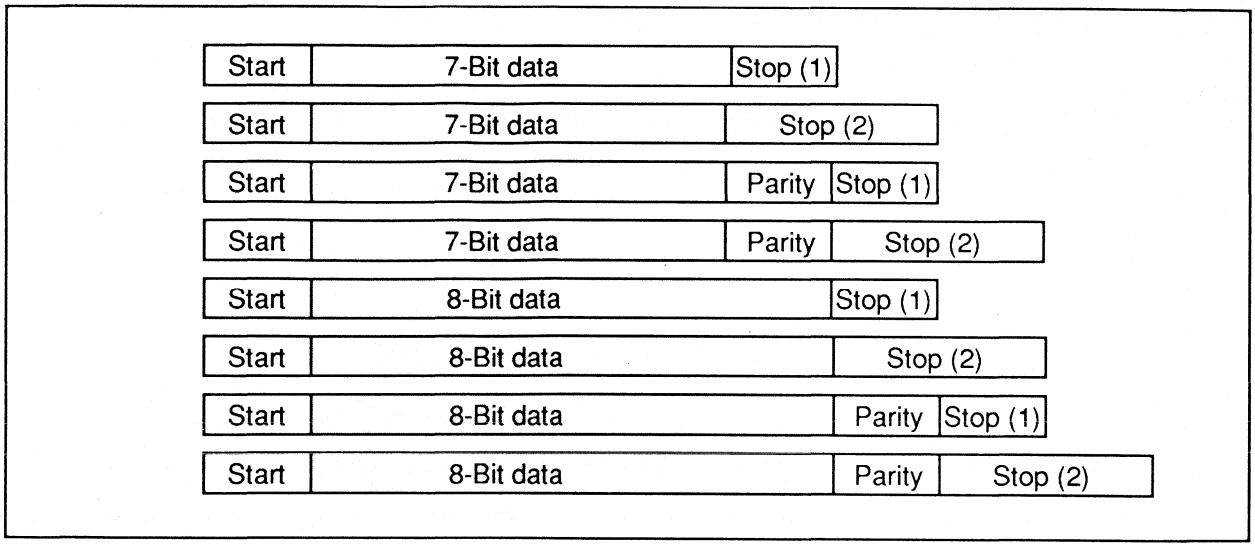
• **Block Diagram**



Block Diagram of Serial Communication Interface

• **Asynchronous Mode:** In the asynchronous communication mode individual characters are synchronized by start bits and stop bits.

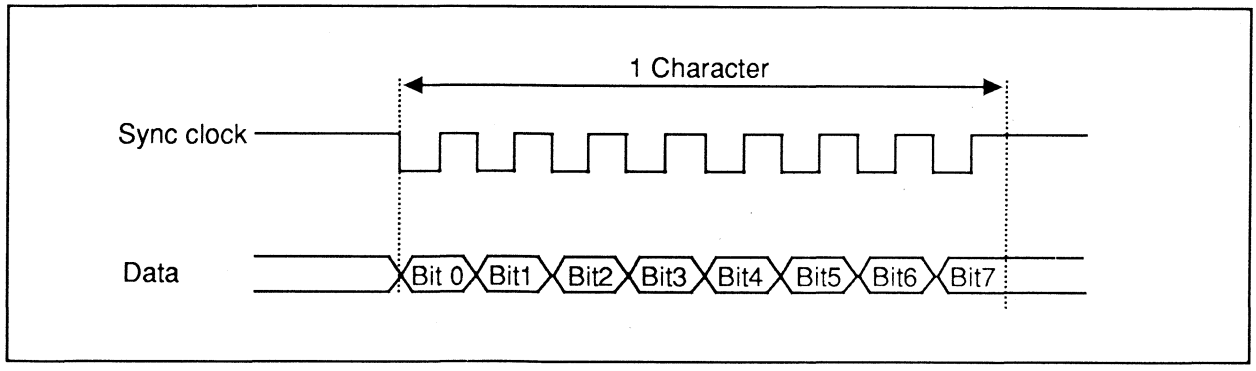
- Eight transfer formats
  - — Data length: 7 or 8 bits
  - — Stop bit length: 1 or 2 bits
  - — Parity bit: even, odd, or no parity
- Either the internal baud rate generator or an external clock input at the ASCK pin can be selected as the clock source.



Eight Data Formats

• **Synchronous Mode:** In the synchronous mode data transfer is synchronized with a clock pulse. This mode is suitable for continuous, high-speed serial communication.

- Data length: 8 bits/character
- Detects overrun errors
- The transmit/receive clock can be provided by the on-chip baud rate generator or by an external clock input at the CSCK pin.
- LSB-first: the least significant data bit is sent and received first.
- Can communicate with the HD64180, HD6301, and other chips supporting a synchronous communication mode.



Data Format in Synchronous Mode (Example)

• **BRR Settings for Typical Bit Rates (Asynchronous Mode)**

XTAL (MHz)	19.6608			20		
ø (MHz)	9.8304			10		
Bit rate (bits/s)	Baud rate generator input clock	BRR value	Error (%)	Baud rate generator input clock	BRR value	Error (%)
110	ø/16	174	–0.26	ø/64	43	+0.88
150	ø/16	127	0	ø/16	129	+0.16
300	ø/4	255	0	ø/16	64	+0.16
600	ø/4	127	0	ø/4	129	+0.16
1200	ø	255	0	ø/4	64	+0.16
2400	ø	127	0	ø	129	+0.16
4800	ø	63	0	ø	64	+0.16
9600	ø	31	0	ø	32	–1.36
19200	ø	15	0	ø	15	+1.73
38400	ø	7	0	ø	7	+1.73

Note: The error should preferably be 1% or less.

$$N = [OSC \div (64 \times 2^{2n} \times B)] \times 10^6 - 1$$

N: BRR value of baud rate generator; $0 \leq N \leq 255$

OSC: Frequency of crystal (MHz)

B: Bit rate (bits/s)

n: Baud rate generator input clock No.; n = 0, 1, 2, 3

CKS1	CKS0	n	Clock
0	0	0	ø
0	1	1	ø/4
1	0	2	ø/16
1	1	3	ø/64

- **BRR Settings for Typical Bit Rates (Synchronous Mode)**

XTAL (MHz)	10		16		20	
ø (MHz)	5		8		10	
Bit rate (bits/s)	Baud rate generator input clock	BRR value	Baud rate generator input clock	BRR value	Baud rate generator input clock	BRR value
250	—	—	ø/64	124	—	—
500	—	—	ø/16	249	—	—
1k	—	—	ø/16	124	—	—
2.5k	ø/4	124	ø/4	199	ø/4	249
5k	ø	249	ø/4	99	ø/4	124
10k	ø	124	ø	199	ø	249
25k	ø	49	ø	79	ø	99
50k	ø	24	ø	39	ø	49
100k	—	—	ø	19	ø	24
250k	ø	4	ø	7	ø	9
500k	—	—	ø	3	ø	4
1M	—	—	ø	1	—	—
2.5M					ø	0

Note: Blanks indicate unavailable settings. A dash (—) indicates that the setting is possible, but incurs error.

$$N = [OSC \div (8 \times 2^{2n} \times B)] \times 10^6 - 1$$

CKS1	CKS0	n	Clock
0	0	0	ø
0	1	1	ø/4
1	0	2	ø/16
1	1	3	ø/64

N: BRR value of baud rate generator; $0 \leq N \leq 255$

OSC: Frequency of crystal (MHz)

B: Bit rate (bits/s)

n: Baud rate generator input clock No.; n = 0, 1, 2, 3

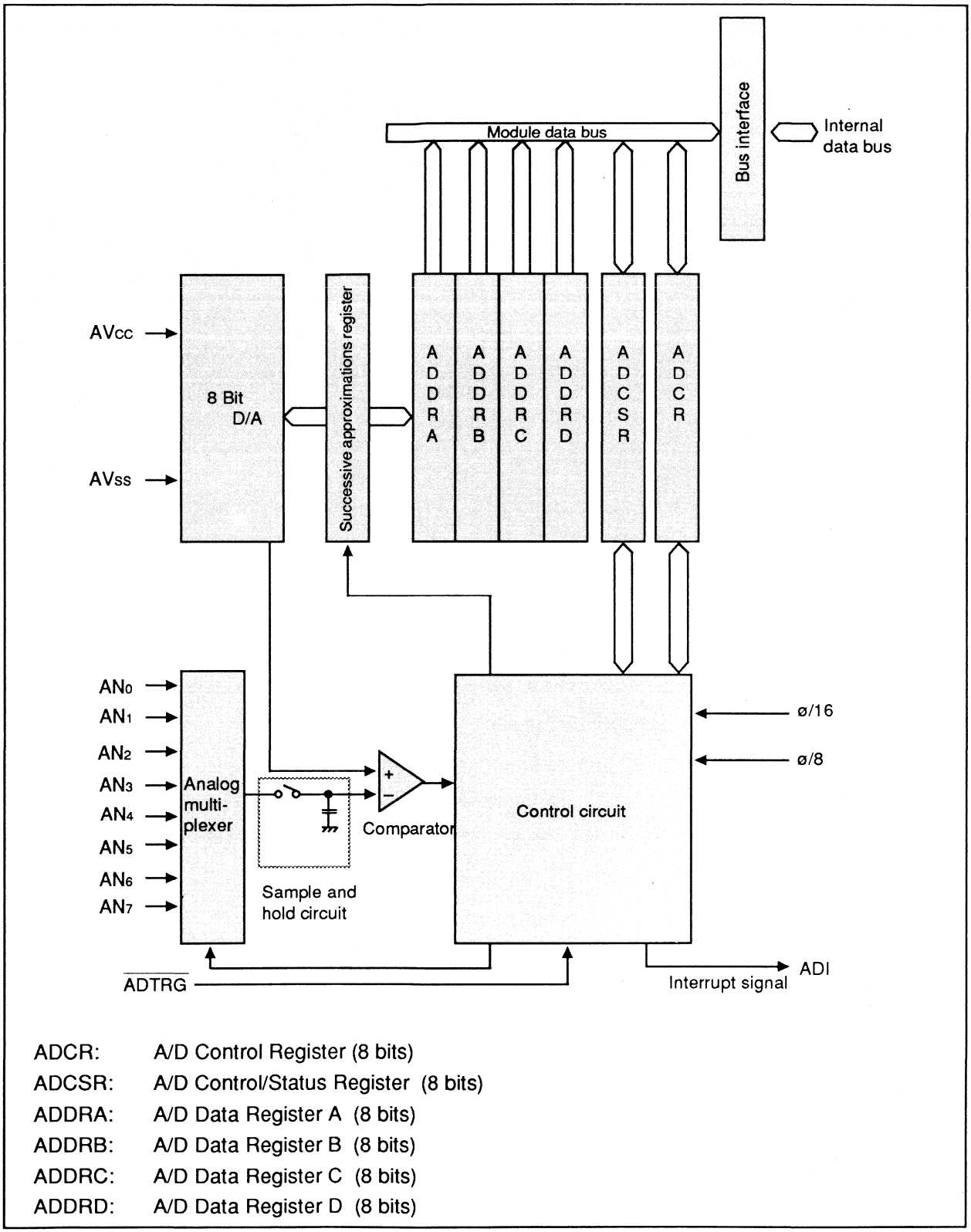
6.5 A/D Converter

One of the H8/330's on-chip supporting modules is an 8-bit analog-to-digital converter which can be programmed for input of up to eight analog signal channels.

- **Features**

- 8-Bit resolution
- Eight analog input pins
- Fast: minimum conversion time is 12.2μs per channel (at 10MHz)
- Selectable mode
  - — Single mode: A/D conversion of one channel
  - — Scan mode: Continuous conversion of 1 to 4 channels
- Four 8-bit data registers. A/D-converted results are transferred to data registers corresponding to each channel and stored.
- A/D conversion can be started by an external trigger signal
- Generates a CPU interrupt (ADI: A/D conversion end) at the completion of A/D conversion.

- **Block Diagram**



ADCR: A/D Control Register (8 bits)
ADCSR: A/D Control/Status Register (8 bits)
ADDRA: A/D Data Register A (8 bits)
ADDRB: A/D Data Register B (8 bits)
ADDRC: A/D Data Register C (8 bits)
ADDRD: A/D Data Register D (8 bits)

Block Diagram of A/D Converter

• **A/D Operation:** Analog-to-digital conversion is performed by successive approximations, with 8-bit resolution. The input channels are selected by the channel select bits (CH2 to CH0) of the ADCSR.

Bit 2	Bit 1	Bit 0	Channel(s) selected	
CH2	CH1	CH0	Single mode	Scan mode
0	0	0	AN0 (Initial value)	AN0
		1	AN1	AN1 to AN0
	1	0	AN2	AN2 to AN0
		1	AN3	AN3 to AN0
1	0	0	AN4	AN4
		1	AN5	AN5 to AN4
	1	0	AN6	AN6 to AN4
		1	AN7	AN7 to AN4

Single Mode: In this mode A/D conversion is performed on one channel under CPU control. Conversion starts when the CPU sets the ADST bit in the ADCSR. When the conversion is completed, the completion flag (ADF) is set. If the interrupt enable bit (ADIE) is also set, an A/D conversion end interrupt (ADI) is requested.

Scan Mode: This mode can be used to monitor analog inputs on one to four channels. When the CPU sets the ADST bit, A/D conversion starts on the first selected channel. As soon as conversion of the first channel is completed, conversion of the next channel begins. (If only one channel is selected, conversion of this channel begins again.) A/D conversion continues in this way until the ADST bit is cleared. The converted results for each channel are transferred to and stored in the data registers ADDRA to ADDRD.

The ADST bit can also be set by the A/D external trigger signal ($\overline{\text{ADTRG}}$), instead of by the CPU.

6.6 I/O Ports

The H8/330 has 58 input/output pins and 8 input-only pins.

The direction of the input/output pins is specified by setting or clearing a bit in a data direction register for each port.

• Port Functions

Port	Description	Pins	Expanded modes		Single-chip mode (mode 3)	
			Mode 1	Mode 2	Non-slave*	Slave*
Port 1	• 8-bit input-output port • Can drive LEDs	$P1_0$ – $P1_7$/ A_0 – A_7	Address bus (Low)	Input pins or address bus (Low) (When DDR = 1)	Input/output port	Input/output port
Port 2	• 8-bit input-output port • Can drive LEDs	$P2_0$ – $P2_7$/ A_8 – A_{15}	Address bus (High)	Input pins or address bus (High) (When DDR = 1)	Input/output port	Input/output port
Port 3	• 8-bit input-output port	$P3_0$ – $P3_7$/ DDB_0 – DDB_7/ D_0 – D_7	Data bus	Data bus	Input/output port	Dual-port RAM data bus
Port 4	• 8-bit input-output port	$P4_0$ – $P4_7$	Input/output port. Also provides input/output pins for 8-bit timers 0 and 1 ($TMCI_0$, TMO_0, $TMRI_0$, $TMCI_1$, TMO_1, $TMRI_1$) and output pins for PWM timers 0 and 1 (PW_0, PW_1)			
Port 5	• 3-bit input-output port	$P5_0$ – $P5_2$	Input/output port. Also provides asynchronous serial communication input/output pins (ATxD, ARxD, ASCK)			
Port 6	• 8-bit input-output port	$P6_0$ – $P6_7$	Input/output port. Also provides input/output pins for free-running timer (FTCI, FTOA, FTIA, FTIB, FTIC, FTID, FTOB) and interrupt input pins ($\overline{IRQ_6}$, $\overline{IRQ_7}$)			
Port 7	• 8-bit input port	$P7_0$ – $P7_7$	Input port. Also provides analog input pins for A/D converter			
Port 8	• 7-bit input-output port	$P8_0$ / RS_0 / E	E clock output pin (after reset) or input port (when DDR = 0)		Input/output port	Dual-port RAM register select signals (RS_0 to RS_3)
		$P8_1$ / RS_1 / $\overline{IOS}$	Input pin (after reset) or $\overline{IOS}$ output pin (when DDR = 1)			
		$P8_2$ / RS_2 $P8_3$ / RS_3	Input/output pins			
		$P8_4$ / CTxD / $\overline{IRQ_3}$ $P8_5$ / CRxD / $\overline{IRQ_4}$ $P8_6$ / CSCK / $\overline{IRQ_5}$	Input/output pins. Also provide synchronous serial communication input/output pins (CTxD, CRxD, CSCK) and external interrupt input pins ($\overline{IRQ_3}$ to $\overline{IRQ_5}$).			

Note: All ports except port 7 have programmable pull-up resistors that can be used in the input state.

* Non-slave: when the dual-port RAM is disabled (DPME = 0)
Slave: when the dual-port RAM is enabled (DPME = 1)

- **Port Functions (cont.)**

Port	Description	Pins	Expanded modes		Single-chip mode (mode 3)	
			Mode 1	Mode 2	Non-slave*	Slave*
Port 9	• 8-bit input-output port	P9₀ / $\overline{IRQ_2}$ / $\overline{ADTRG}$ P9₁ / $\overline{IRQ_1}$ P9₂ / $\overline{IRQ_0}$	Input/output pins, external interrupt pins ($\overline{IRQ_0}$ to $\overline{IRQ_2}$), and A/D conversion trigger input pin ($\overline{ADTRG}$).			
		P9₃ / $\overline{CS}$ / $\overline{RD}$	$\overline{RD}$ output		Input/output pins	Dual-port RAM $\overline{CS}$ input
		P9₄ / $\overline{OE}$ / $\overline{WR}$	$\overline{WR}$ output			Dual-port RAM $\overline{OE}$ input
		P9₅ / $\overline{RDY}$ / $\overline{AS}$	$\overline{AS}$ output			Dual-port RAM $\overline{RDY}$ output (NMOS open drain)
		P9₆ / ø	ø output		Input pin (after reset) or ø output (when DDR = 1)	
		P9₇ / $\overline{WE}$ / $\overline{WAIT}$	$\overline{WAIT}$ input		Input/output pin	Dual-port RAM $\overline{WE}$ input

Note: All ports except port 7 have programmable pull-up resistors that can be used in the input state.

* Non-slave: when the dual-port RAM is disabled (DPME = 0)
Slave: when the dual-port RAM is enabled (DPME = 1)

- **MOS Input Pull-Ups:** Ports 3 and 4 have MOS input pull-up transistors. These pull-ups are switched on and off by setting the bits in the data direction register (DDR) and data register (DR) as shown next.

DDR	DR	MOS Pull-Up
1	0	OFF
	1	
0	0	ON
	1	

- **Input/Output Characteristics**

— Preliminary —

Parameter		Symbol	Conditions	Min.	Max.	Unit
Output High voltage (all outputs)		V_{OH}	$I_{OH} = -200\mu A$	$V_{CC} - 0.5$	—	V
			$I_{OH} = -1mA$	3.5	—	V
Output Low voltage	All outputs	V_{OL}	$I_{OL} = 1.6mA$	—	0.4	V
	Port 1, 2		$I_{OL} = 10mA$	—	1.0	V
Input High voltage		V_{IH}		2.2	$V_{CC} + 0.3$	V
Input Low voltage		V_{IL}		-0.3	0.8	V
Schmitt trigger input voltages		V_{T^+}		2.0	3.5	V
		V_{T^-}		1.0	2.5	V
		$V_{T^+} - V_{T^-}$		0.4	—	V
Input pull-up current		$-I_p$	$V_{in} = 0V$	30	250	μA

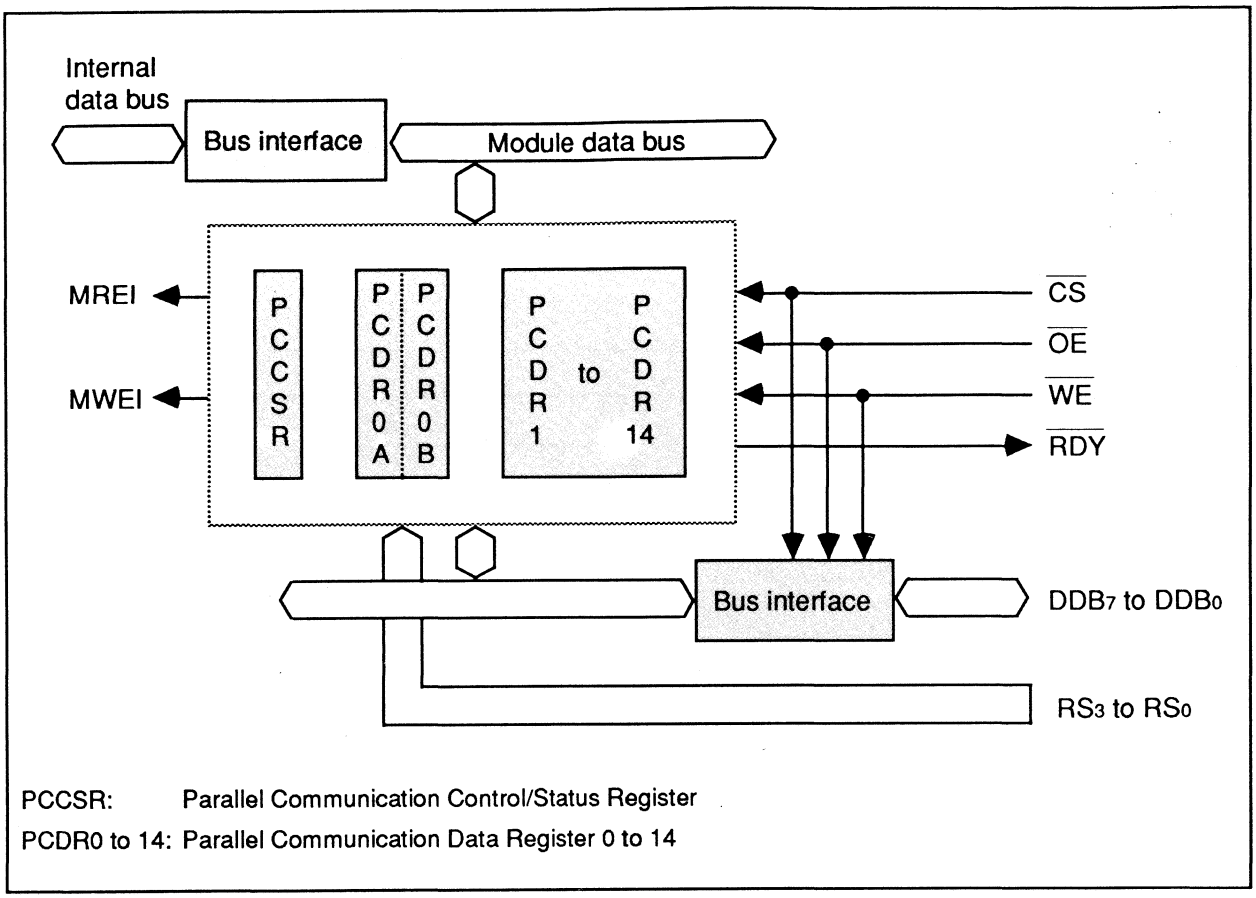
6.7 Dual-Port RAM

The H8/330 has an on-chip 15-byte dual-port RAM (DPRAM) that can be accessed by both the H8/330's CPU and an external CPU. The dual-port RAM can be used in the single-chip mode when the DPME bit is set to "1."

The dual-port RAM can be used for master-slave parallel communication in a master/slave microcomputer system in which the H8/330 is the slave.

- **Features**
  - — 15-Byte data register
  - — Standard memory interface simplifies connection to an external (master) CPU
  - — Simple master-slave protocol
  - — Can generate a master CPU interrupt request

- **Block Diagram**



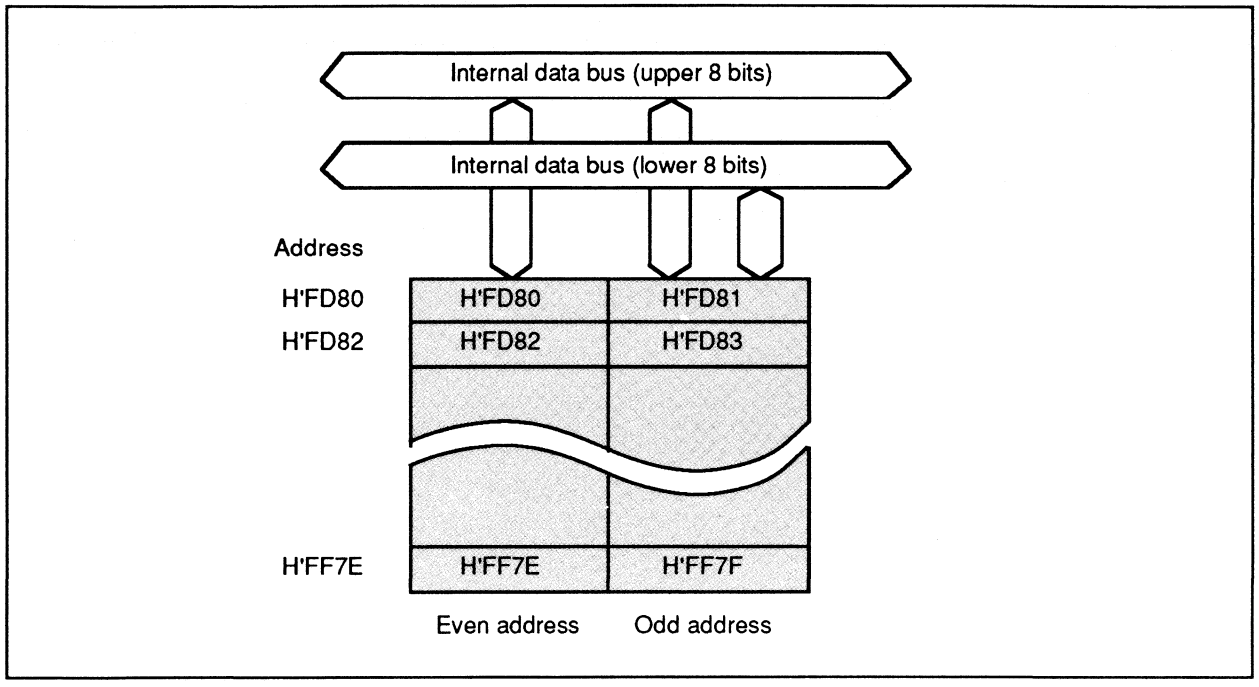
Block Diagram of Dual-Port RAM

6.8 RAM

The H8/330 has 512 bytes of high-speed static RAM on-chip. The RAM is linked to the CPU via a 16-bit data bus, so both byte and word data are accessed in two states. This allows particularly rapid transfer of word data.

The on-chip RAM can be enabled and disabled by the RAM enable (RAME) bit.

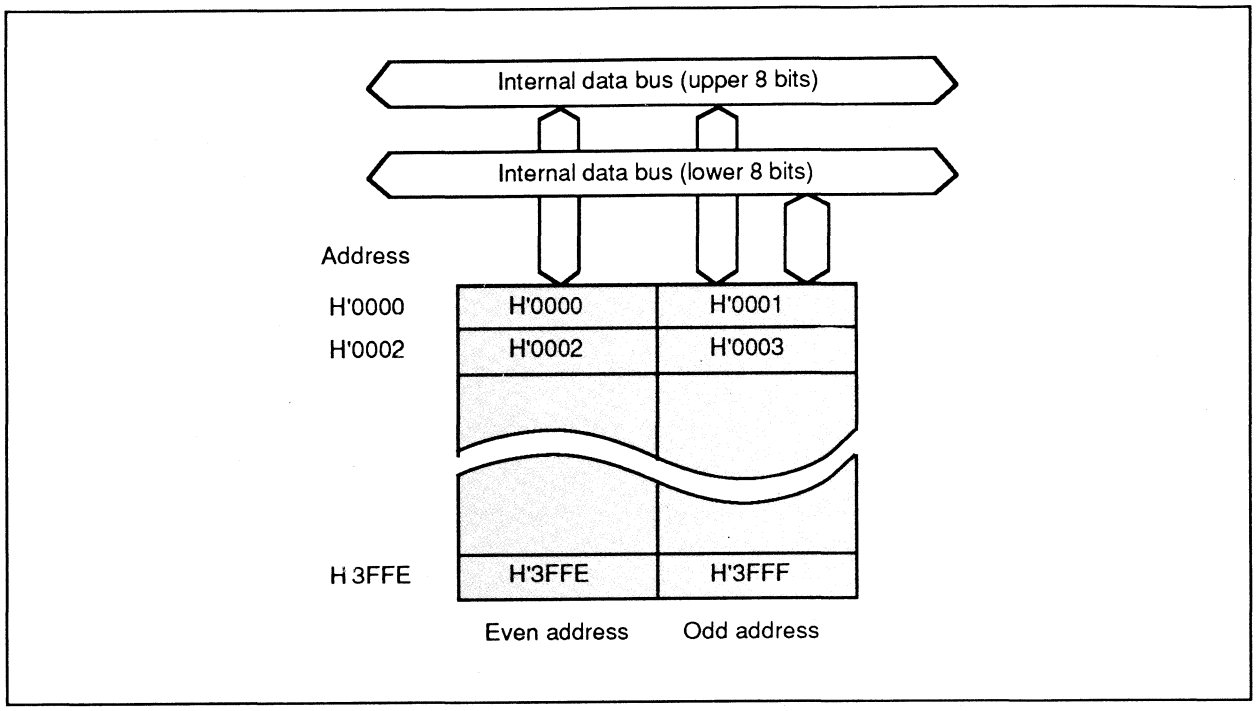
- **Block Diagram**



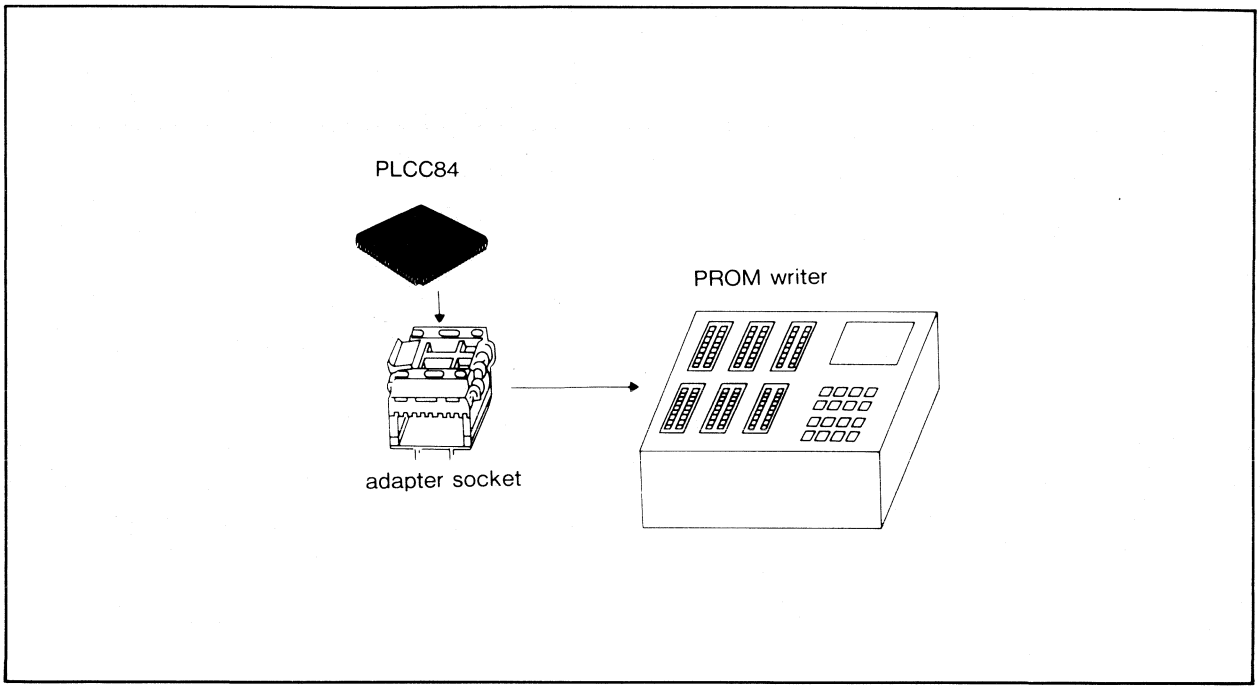
6.9 PROM (or Masked ROM)

The H8/330 has 16K bytes of on-chip PROM (or masked ROM). Like the on-chip RAM, the on-chip ROM is linked to the CPU via a 16-bit data bus and accessed in two states.

- **Block Diagram**



- **PROM Programming:** When set to the PROM mode, the H8/330 suspends its microcomputer functions and enables data to be written directly to the on-chip PROM. The on-chip PROM can be written and read according to the same specifications as the 27C256 EPROM (V_{PP} = 12.5V). If an adapter socket is used to convert from 80 or 84 to 28 pins, the PROM can be programmed easily using a commercially available PROM writer. The programmable addresses are H'0000 to H'3FFF.


PLCC84
PROM writer
adapter socket

Section 7. Power-Down Modes

In addition to the normal program execution state, the H8/330 has a power-down state in which the CPU halts to conserve power. There are three power-down modes: the sleep mode, the software standby mode, and the hardware standby mode. In the two standby modes the system clock also stops.

7.1 Sleep mode

Execution of the SLEEP instruction places the H8/330 in the sleep mode, in which the CPU halts but the on-chip supporting modules continue to operate.

Recovery from the sleep mode is possible by a reset or an interrupt. The CPU returns via the exception-handling state to the program execution state and starts servicing the interrupt, or executing the reset routine.

7.2 Software standby mode

When the software standby (SSBY) bit is set, execution of the SLEEP instruction places the H8/330 in the software standby mode. All chip functions halt, including the clock, but the contents of CPU registers and on-chip RAM are held.

Recovery from the software standby mode is by an external interrupt ($\overline{\text{NMI}}$ or $\overline{\text{IRQ}}_0$ to $\overline{\text{IRQ}}_2$). After the clock stabilizes, the CPU returns via the exception-handling state to the program execution state and starts servicing the interrupt.

By stopping the clock, the software standby mode reduces power consumption to an extremely low level.

7.3 Hardware standby mode

A Low input at the $\overline{\text{STBY}}$ pin places the H8/330 in the hardware standby mode. As in the software standby mode, all chip functions halt but the CPU register contents and on-chip RAM contents are held.

To recover from the hardware standby mode, first the $\overline{\text{RES}}$ pin should be made Low, then $\overline{\text{STBY}}$ should be made High, after which $\overline{\text{RES}}$ should be made High to restart from the reset state.

By stopping the clock, the hardware standby mode, like the software standby mode, reduces power consumption to an extremely low level.

Mode	Clock	CPU	Supporting functions	On-chip RAM, CPU registers	Recovery methods
Sleep	Runs	Halts	Run	Held	Interrupt—Interrupt is accepted and interrupt handling begins. $\overline{\text{RES}}$—Transition to reset state $\overline{\text{STBY}}$—Transition to the hardware standby mode
Software standby	Halts	Halts	Halt	Held	External interrupt— The clock starts, and after a clock stabilization time set in an on-chip timer, execution of the external interrupt handling routine starts automatically. $\overline{\text{RES}}$—Clock starts, followed by transition to reset state. $\overline{\text{STBY}}$—Transition to the hardware standby mode
Hardware standby	Halts	Halts	Halt	Held	Recovered by making the $\overline{\text{RES}}$ pin Low, then the $\overline{\text{STBY}}$ pin High; waiting for the clock to stabilize; then making $\overline{\text{RES}}$ High.

Section 8. Support Tools

The H8/330 programming environment includes both software and hardware tools for efficient program development.

- **Software**
  - — H8/300 C compiler
  - — H8/300 Assembler
  - — Linkage editor (including librarian and load module converter)
  - — H8/300 Simulator/debugger

- **Hardware**
  - — H8/330 ASE (Adaptive System Evaluator)

8.1 Software

- **C Compiler**
- Confirms to ANSI (American National Standards Institute) C-language specifications.
- Optimizing feature minimizes object program size.
- Can generate assembly-language output in addition to object code.

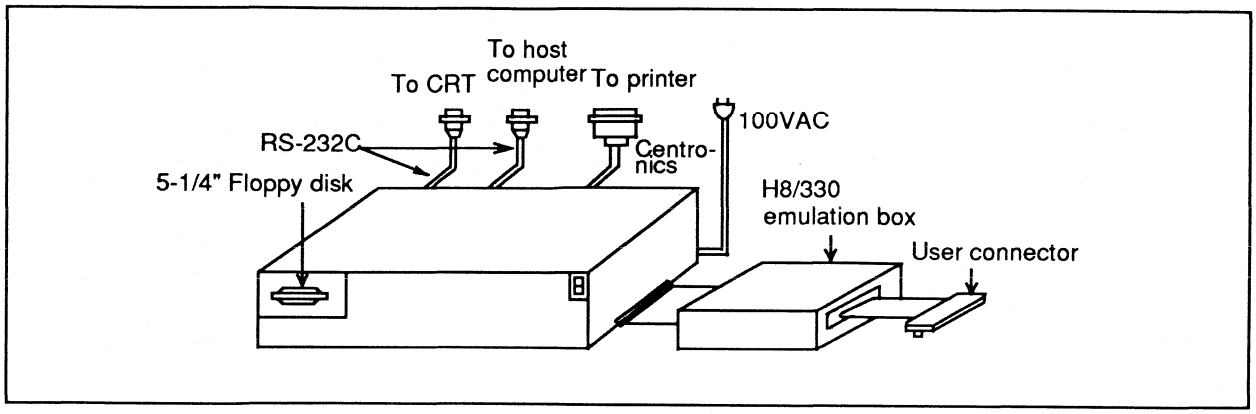
- **Assembler**
- Supports structured assembly.
- IEEE standard executable instructions and directives.
- Generates SYSROF object format.

- **Linkage Editor (Including Librarian and Load Module Converter)**
- The linkage editor links object programs output by the assembler or C compiler (in SYSROF format) to create relocatable load modules (in SYSROF format).
- The librarian stores user programs and libraries supplied by the compiler in files.
- The load module converter converts SYSROF-format load-module files output by the linkage editor to S-type files.

- **H8/300 Simulator/Debugger**
- Enables the user to debug programs on a host computer, without a target system.
- Can trace registers and provide breakpoint control.
- Permits simulation with relocatable object code.
- Supports multiuser debugging for team program development.

8.2 Hardware

- **Realtime emulator (ASE)**



H8/330 ASE

- **Hardware Breakpointing (2 Breakpoints)**
- Settable breakpoint conditions
  - — Address bus (range specification allowed)
  - — Data bus (range specification allowed)
  - — Program counter
  - — $\overline{RD}/\overline{WR}$ signal
  - — External interrupts ($\overline{NMI}$, $\overline{IRQ_0}$ to $\overline{IRQ_7}$)
  - — Condition-match count (Maximum count: 4095)
  - — Delay count (Maximum 4095 cycles)
- Sequential breakpointing
  - — Program execution halts when two breakpoints are passed in a specified order.

- **PC Breakpointing (255 Breakpoints)**
- Settable breakpoint conditions
  - — Program counter
  - — Condition-match count
- Sequential breakpointing
  - — Program execution halts when four breakpoints are passed in a specified order.

- **Triggering (One Trigger Point)**

Trace information can be viewed without stopping the CPU. The settable conditions and breakpoints are the same as for hardware .

• Realtime Trace

- Information traced
  - — Address bus
  - — Data bus
  - — External probe signals
- Trace information capture
  - — Free trace (all trace information generated during execution is captured)
  - — Range specification (specified trace capture conditions)
- Trace information search
  - — Information in the trace buffer can be searched according to specified conditions

• Coverage Function

- Indicates the percent of the address area covered in a program run.

• Performance Analysis

- Measures program execution efficiency.

• Parallel Mode

- Commands can be entered as the program executes, without halting execution.

• Symbolic Debugger

- Debugging can be carried out using source program labels or line numbers.

HITACHI EUROPEAN SALES LOCATIONS

Hitachi Europe GmbH
Electronic Components Division

Headquarter:
Hans-Pinsel-Str. 10A
D-8013 Haar bei München
Tel.: (0 89) 46 14-0; Telex: 5 22 593 hitc d
Telefax: (0 89) 46 31 51

Sales Offices:
North Germany / Benelux
Am Seestern 18; Postfach 11 05 36
D-4000 Düsseldorf 11
Tel.: (02 11) 52 83-0; Teletext: 2 114 096 HIEC D
Telefax: (02 11) 52 83-779

Central Germany
Fabrikstraße 17; D-7024 Filderstadt
Tel.: (07 11) 77 20 11
Teletex: 71 11 410 HITECS
Telefax: (07 11) 7 77 51 16

South Germany
Hans-Pinsel-Str. 10A
D-8013 Haar bei München
Tel.: (0 89) 46 14-0; Telex: 5 22 593 hitc d
Telefax: (0 89) 46 31 51

Italy
Via L. Rizzo 8; I-20151 Milano
Tel.: (02) 33 40 41 80; Telex: 32 33 77 hitec i
Telefax: (02) 33 40 41 52

Via Pescosolido 154; I-00158 Roma
Tel.: (06) 4 51 01 46, -1 47
Telefax: (06) 4 51 01 48

Spain
Sucursal en Espana
c/Buganvilla, 5; E-28036 Madrid
Tel.: 00 34/1/7 67 27 82, 7 67 27 92
Telefax: 00 34/1/3 83 85 11

France
Immeuble «Les Gémeaux»
2 Rue Antoine Etex; F-94020 Créteil
Tel.: (1) 43 39 45 00; Telex: 2 62 246
Telefax: (1) 43 39 84 93

Nissei Sangyo GmbH
Headquarters & Germany
Gustav-Heinemann-Ring 121; D-8000 München 83
Tel.: (0 89) 6 78 28-0; Telex: 5 29 963 nism d
Telefax: (0 89) 6 70 31 02 oder 6 70 43 52

Berlin
Bayreuther Str. 4; 1000 Berlin 30
Tel.: (0 30) 24 80 81; Telex: 18 36 70
Telefax: (0 30) 2 13 96 95

France
Nissei Sangyo S.A.R.L.
2, Rue Louis Armand; Tour Objectiv 5F
F-92700 Asnières
Tel.: (1) 47 93 67 10; Telefax: (1) 47 93 00 60

Spain
Nissei Sangyo France (S.A.R.L.)
Barcelona Office
Gran Via Carlos III, 98
Planta 1-4
Edificios Trade
E-09028 Barcelona
Tel.: (3) 330-97 53 / 98 05; Telefax (3) 339-09 95

Hitachi Europe GmbH
Electronic Components Division
(Continental Europe)
Hans-Pinsel-Straße 10A
D-8013 Haar/München, Germany
Telephone: +89 4 61 40;
Telex: 522 593 hitc d
Telefax: +89 46 31 51 / 46 30 68

Hitachi Europe Ltd.
Electronic Components Division
(Northern Europe)
Whitebrook Park, Lower Cookham Road,
Maidenhead, Berkshire, SL6 8YA,
United Kingdom
Telephone: +628 58 50 00;
Fax: +628 585 200