H8S/2633 Series

Overview

HITACHI

ADE-802-224
Rev. 1.0
2/01/99
Hitachi, Ltd.

Hitachi semiconductor

Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.
2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.
3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.
4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.
5. This product is not designed to be radiation resistant.
6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.
7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

Preface

Hitachi's H8S family of single-chip microcomputers comprises a number of new series offering the high performance and low power consumption of the existing H8 Series, which is widely used for machine control, etc., together with significantly greater ease of use.

The H8S/2000 Series features CPU object-level compatibility with the H8/300H Series, H8/300 Series, and H8/300L Series within the H8 Series.

Series	Features
H8S/2000	Upward-compatible with the H8/300H Series and H8/300 Series; twice the performance at the same frequency
H8/300H	16-Mbyte linear address space; upward-compatible with the H8/300 Series; concise instruction set; powerful word-size and longword-size arithmetic instructions
H8/300	64-kbyte address space; general register system; concise instruction set; powerful bit manipulation instructions
H8/300L	Same CPU as the H8/300 Series; consumer application oriented supporting modules; low voltage, low power consumption

This manual gives an overview of the H8S/2633 Series of products for single-chip applications within the H8S Series.

Intended Readership: This Overview is intended for readers who have a basic understanding of microcomputers, and are looking for information on the features and functions of the H8S/2633 Series. Readers undertaking system design using these products, or requiring more detailed information on their use, should refer to the relevant Hardware Manuals and the H8S/2600 and H8S/2000 Series Programming Manual.

Related Documents

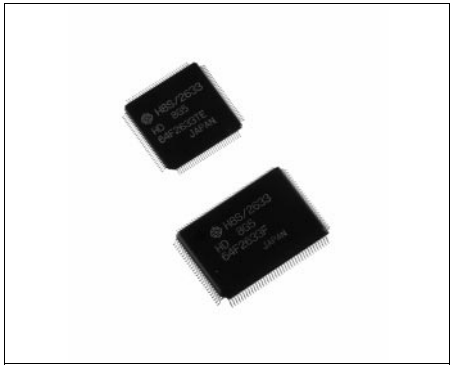
Contents	Title	Document No.
H8S/2633 Series hardware	H8S/2633 Series Hardware Manual	ADE-602-165
H8S/2000 Series execution instructions	H8S/2600 Series and H8S/2000 Series Programming Manual	ADE-602-083A

The product specifications in this Overview are subject to change without notice. The relevant Hardware Manual must be used when undertaking product design.

Contents

Section 1 Overview of H8S/2633 Series

1.1 Features of H8S/2633 Series



H8S/2633 Series microcomputers are designed for faster instruction execution, using a realtime control oriented CPU with an internal 32-bit architecture, and can run programs based on the C high-level language efficiently. As well as large-capacity ROM and RAM, these microcomputers include comprehensive on-chip supporting modules needed for control systems, simplifying the implementation of sophisticated, high-performance systems.

High-performance H8S/2600 CPU

- General-register architecture
  - — Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)
- High-speed operation suitable for realtime control
  - — 25 MHz maximum operating frequency (V_{CC} = 3.3 V ±0.3 V, PV_{CC} = 5 V ±10%)
  - — High-speed arithmetic operations
    - 8/16/32-bit register-register add/subtract: 40 ns
    - 16 × 16-bit register-register multiply: 160 ns
    - 16 × 16 + 42-bit multiply-and-accumulate: 160 ns
    - 32 ÷ 16-bit register-register divide: 800 ns
- Instruction set suitable for high-speed operation
  - — Sixty-nine types of basic instructions
  - — 8/16/32-bit transfer instructions
  - — Unsigned/signed multiply and divide instructions
  - — Multiply-and-accumulate instruction
  - — Powerful bit manipulation instructions
- CPU operating mode
  - — Advanced mode: Maximum 16-Mbyte address space

256-kbyte flash memory or 256-/192-/128-kbyte mask ROM on-chip*

Note: * In planning stage

On-chip high-speed static RAM

- 16/12/8 kbytes

Bus controller on-chip

- Address space divided into 8 areas, with bus specifications settable independently for each area
- Chip select output possible for areas 0 to 7
- Selection of 8-bit or 16-bit access space for each area
- 2-state or 3-state access space can be set for each area
- Number of program wait states can be set for each area
- Burst ROM directly connectable
- Up to 8 Mbytes of DRAM can be directly connected (or interval timer can be used)
- External bus release function

DMA controller (DMAC)

- Selection of short address mode or full address mode
- Four channels in short address mode, two channels in full address mode
- Transfer possible in repeat mode, block transfer mode, etc.
- Single address mode transfer possible
- Can be activated by internal interrupt

Data transfer controller (DTC)

- Activated by internal interrupt or software
- Multiple transfers or multiple types of transfer possible for one activation source
- Transfer possible in repeat mode, block transfer mode, etc.
- Request can be sent to CPU for interrupt that activated DTC

16-bit timer-pulse unit (TPU)

- Six-channel 16-bit timer on-chip
- Pulse I/O processing capability for up to 16 pins
- Automatic 2-phase encoder count capability

Programmable pulse generator (PPG)

- Maximum 8-bit pulse output possible with TPU as time base
- Output trigger selectable in 4-bit groups
- Non-overlap interval can be set
- Direct output or inverse output setting possible

Four 8-bit timer (TMR) channels

Features of each channel:

- 8-bit up-counter (external event count capability)
- Two time constant registers
- 2-channel cascaded connection possible

Two watchdog timer (WDT) channels

- Watchdog timer or interval timer function selectable
- Subclock operation possible (channel 1 only)

14-bit PWM timer (PWM)

- Maximum of 4 outputs
- Resolution: 1/16384
- Maximum carrier frequency: 390.6 kHz (operating at 25 MHz)

Five serial communication interface (SCI) channels (SCI0 to SCI4)

- Asynchronous mode or synchronous mode selectable
- Multiprocessor communication function
- Smart card interface function

IrDA-equipped SCI1 channel (SCI0)

- Supports IrDA standard version 1.0
- TxD and RxD encoding/decoding in IrDA format
- Synchronization mode or clock synchronization mode selectable
- Multiprocessor communications function
- Smart card interface function

A/D converter

- Resolution: 10 bits
- Input: 16 channels
- High-speed conversion: 10.72 μs minimum conversion time (25 MHz operation)
- Single or scan mode selectable
- Sample-and-hold function
- A/D conversion can be activated by external trigger or timer trigger

D/A converter

- Resolution: 8 bits
- Output: 4 channels

Two PC break controller channels

- Supports debugging functions by means of PC break interrupts
- Two break channels

I/O ports

- 73 I/O pins
- 16 input-only pins

Interrupt controller

- Nine external interrupt pins (NMI, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ7}}$)
- 72 internal interrupt sources (including options)
- Eight priority levels settable

Power-down state

- Medium-speed mode
- Sleep mode
- Subactive mode
- Subsleep mode
- Watch mode
- Module stop mode
- Software standby mode
- Hardware standby mode

Four MCU operating modes

Mode	CPU Operating Mode	Description	On-Chip ROM	External Data Bus	
				Initial Value	Maximum Value
4	Advanced	On-chip ROM disabled expansion mode	Disabled	16 bits	16 bits
5		On-chip ROM disabled expansion mode	Disabled	8 bits	16 bits
6		On-chip ROM enabled expansion mode	Enabled	8 bits	16 bits
7		Single-chip mode	Enabled	—	—

On-chip clock pulse generator (1:1 oscillation)

- Built-in duty correction circuit

Packages

- 120-pin plastic TQFP (TFP-120)
- 128-pin plastic QFP (FP-128)

Product lineup (preliminary)

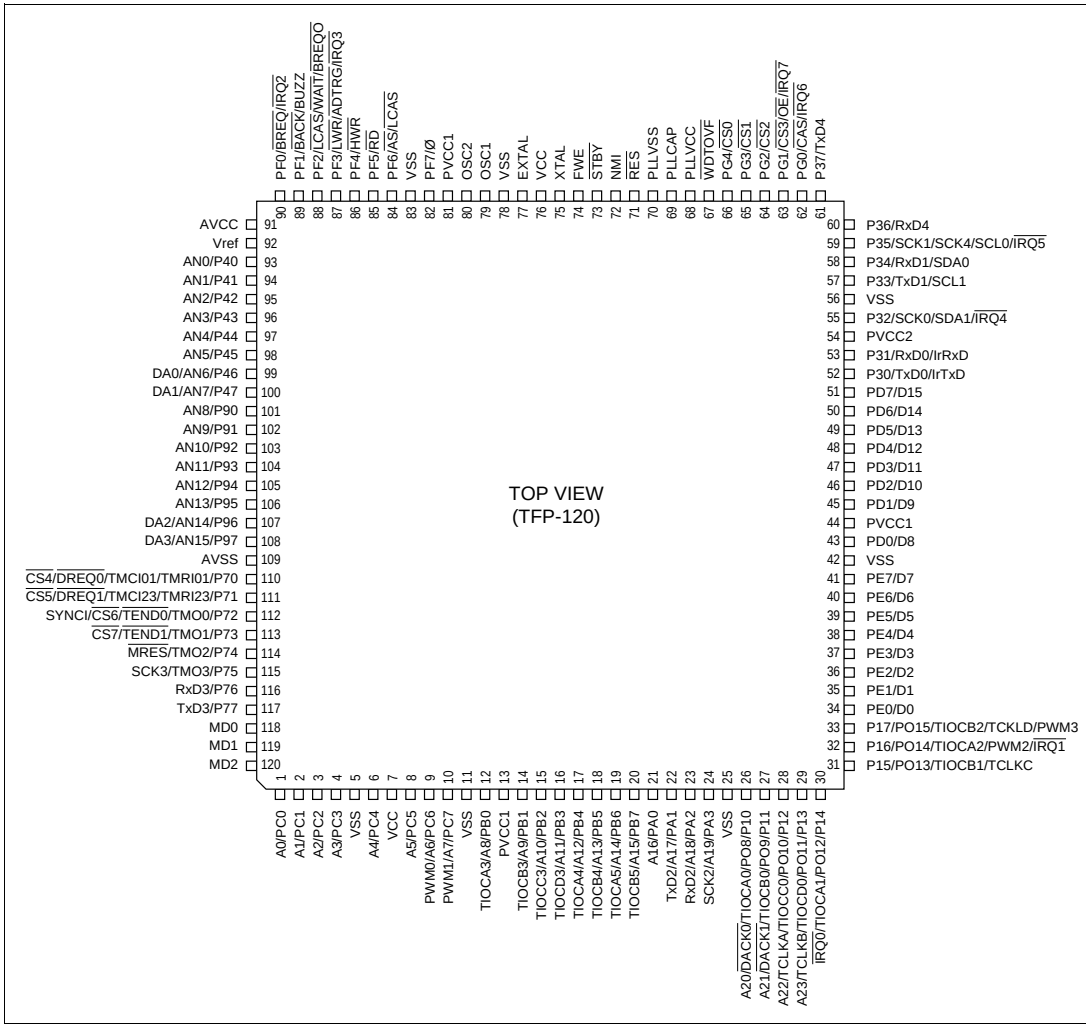
Product Name			
Mask ROM Version	F-ZTAT™ Version	ROM/RAM (Bytes)	Packages
HD6432633	HD64F2633	256 k/16 k*	TFP-120, FP-128
HD6432632	—	192 k/12 k*	
HD6432631	—	128 k/8 k*	

Note: F-ZTAT™ is a trademark of Hitachi Ltd.
* In planning stage

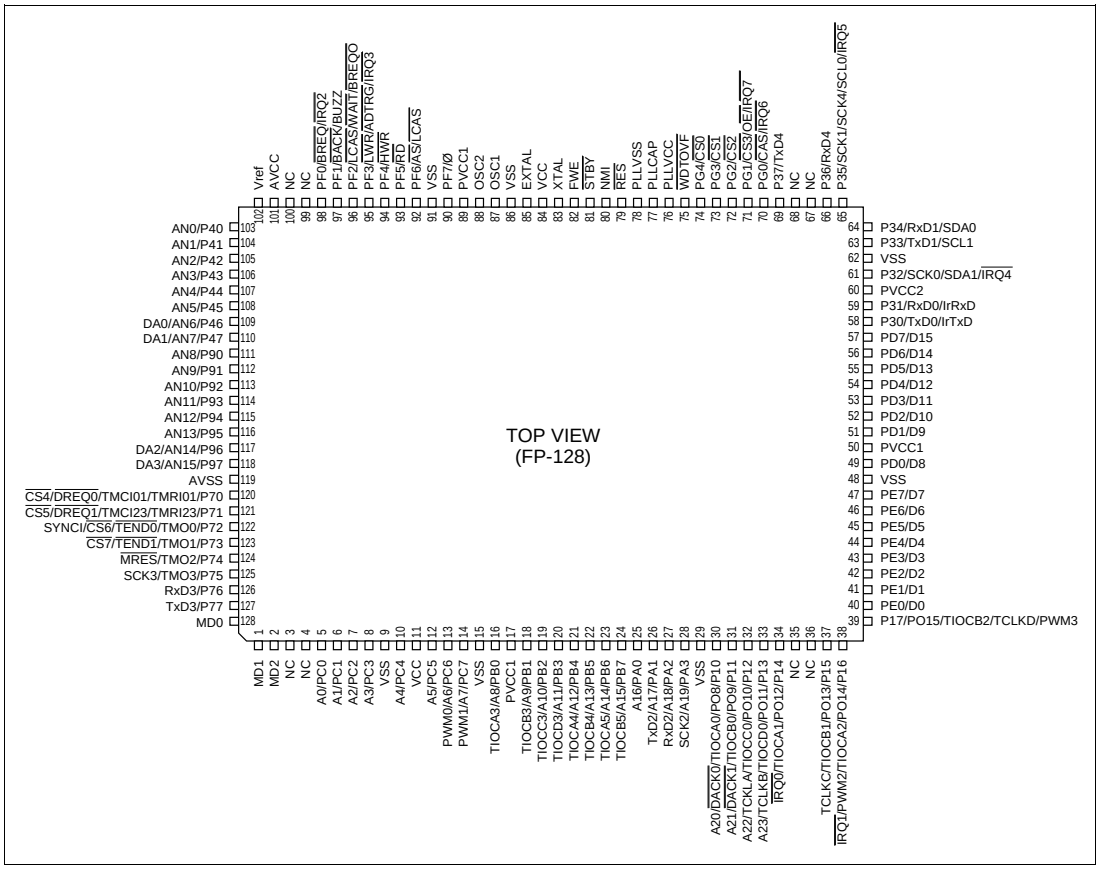
1.2 Pin Arrangement and Functions

H8S/2633 Series Pin Arrangement

- 120-pin plastic TQFP (TFP-120)



- 128-pin plastic QFP (FP-128)



HITACHI

Pin Functions

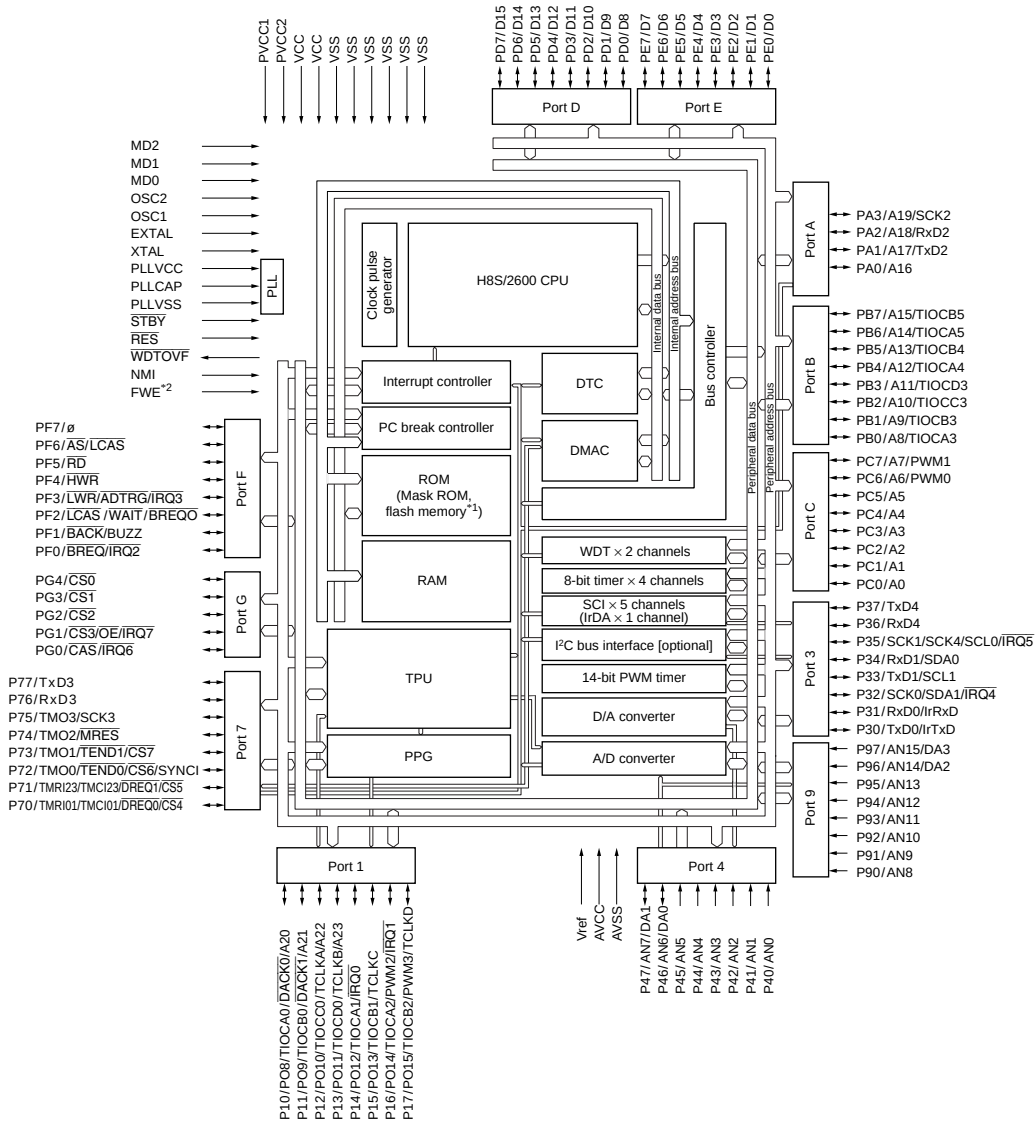
Type	Symbol	I/O	Name and Function
Power	VCC	Input	Power supply
	PVCC1	Input	Power supply pin for ports 1, 7, A to G and $\overline{\text{WDTOVF}}$ pin.
	PVCC2	Input	Power supply pin for port 3
	VSS	Input	Ground Connected to the power supply (0 V).
Clock	PLLVCC	Input	PLL power supply
	PLLVCC	Input	PLL ground
	PLLCAP	Input	PLL capacitance
	XTAL	Input	Connects to a crystal oscillator.
	EXTAL	Input	Connects to a crystal oscillator, or external clock input.
	OSC1	Input	Connects to a 32.768 kHz crystal oscillator
	OSC2	Input	Connects to a 32.768 kHz crystal oscillator
	ø	Output	System clock
Operating mode control	MD2 to MD0	Input	Set mode pins
System control	$\overline{\text{RES}}$	Input	Reset input
	$\overline{\text{MRES}}$	Input	Manual reset input
	$\overline{\text{STBY}}$	Input	Standby
	$\overline{\text{BREQ}}$	Input	Bus request
	$\overline{\text{BREQO}}$	Output	Bus request output
	$\overline{\text{BACK}}$	Output	Bus request acknowledge
	FWE	Input	Flash write enable
Interrupt signal	NMI	Input	Nonmaskable interrupt
	$\overline{\text{IRQ7}}$ to $\overline{\text{IRQ0}}$	Input	Interrupt request 7 to 0
Address bus	A23 to A0	Output	Address bus
Data bus	D15 to D0	I/O	Data bus

Type	Symbol	I/O	Name and Function
Bus control	$\overline{CS7}$ to $\overline{CS0}$	Output	Chip select
	$\overline{AS}$	Output	Address strobe
	$\overline{RD}$	Output	Read
	$\overline{HWR}$	Output	High write
	$\overline{LWR}$	Output	Low write
	$\overline{CAS}$	Output	Upper column address strobe/column address strobe
	$\overline{LCAS}$	Output	Lower column address strobe
	$\overline{WAIT}$	Input	Wait
DMA controller (DMAC)	$\overline{DREQ1}$, $\overline{DREQ0}$	Input	DMA request 1 and 0
	$\overline{TEND1}$, $\overline{TEND0}$	Output	DMA transfer end 1 and 0
	$\overline{DACK1}$, $\overline{DACK0}$	Output	DMA transfer acknowledge 1 and 0
16-bit timer-pulse unit (TPU)	TCLKA to TCLKD	Input	Clock input A to D
	TIOCA0, TIOCB0, TIOCC0, TIOCD0	I/O	Input capture/output compare match A0 to D0
	TIOCA1, TIOCB1	I/O	Input capture/output compare match A1 and B1
	TIOCA2, TIOCB2	I/O	Input capture/output compare match A2 and B2
	TIOCA3, TIOCB3, TIOCC3, TIOCD3	I/O	Input capture/output compare match A3 to D3
	TIOCA4, TIOCB4	I/O	Input capture/output compare match A4 and B4
	TIOCA5, TIOCB5	I/O	Input capture/output compare match A5 and B5
Programmable pulse generator (PPG)	PO15 to PO8	Output	Pulse output 15 to 8
8-bit timer	TMO0 to TMO3	Output	Compare match output
	TMCI01, TMCI23	Input	Counter external clock input
	TMRI01, TMRI23	Input	Counter external reset input
14-bit PWM timer (PWMX)	PWM0 to PWM3	Output	PWMX timer output
Watchdog timer (WDT)	$\overline{WDTOVF}$	Output	Watchdog timer overflow
	BUZZ	Output	BUZZ output

Type	Symbol	I/O	Name and Function
Serial communication interface (SCI)/ Smart Card interface	TxD4, TxD3, TxD2, TxD1, TxD0	Output	Transmit data (channel 4, 3, 2, 1, 0)
	RxD4, RxD3, RxD2, RxD1, RxD0	Input	Receive data (channel 4, 3, 2, 1, 0)
	SCK4, SCK3, SCK2, SCK1, SCK0	I/O	Serial clock (channel 4, 3, 2, 1, 0)
IrDA-equipped SCI (SCI0)	IrTxD	Output	IrDA transmission data
	IrRxD	Input	IrDA receive data
I^2C bus interface (IIC) (optional)	SCL0, SCL1	I/O	I^2C clock input/output (channel 0, 1)
	SDA0, SDA1	I/O	I^2C data input/output (channel 0, 1)
A/D converter (ADC)	AN15 to AN0	Input	Analog input
	$\overline{\text{ADTRG}}$	Input	A/D conversion external trigger input
D/A converter	DA3 to DA0	Output	Analog output
A/D converter and D/A converter	AVCC	Input	Analog power supply
	AVSS	Input	Analog ground connected to the system power supply (0 V)
	Vref	Input	Analog reference voltage
I/O ports	P17 to P10	I/O	Port 1
	P37 to P30	I/O	Port 3
	P47 to P40	Input	Port 4
	P77 to P70	I/O	Port 7
	P97 to P90	Input	Port 9
	PA3 to PA0	I/O	Port A
	PB7 to PB0	I/O	Port B
	PC7 to PC0	I/O	Port C
	PD7 to PD0	I/O	Port D
	PE7 to PE0	I/O	Port E
	PF7 to PF0	I/O	Port F
	PG4 to PG0	I/O	Port G

1.3 Block Diagram

Internal Block Diagram of H8S/2633 Series



Notes: 1. Flash memory applies to the H8S/2633 only.
2. The FWE pin is used only in the flash memory version.

Section 2 CPU

2.1 Features

The H8S/2600 CPU is a high-speed central processing unit with an internal 32-bit architecture that is upward-compatible with the H8/300 and H8/300H CPUs. The H8S/2600 CPU has sixteen 16-bit general registers, can address a 16-Mbyte (architecturally 4-Gbyte) linear address space, and is ideal for realtime control.

Features

- Upward-compatible with H8/300 and H8/300H CPUs
  - Can execute H8/300 and H8/300H object programs
- General-register architecture
  - Sixteen 16-bit general registers (also usable as sixteen 8-bit registers or eight 32-bit registers)
- Sixty-nine basic instructions
  - 8/16/32-bit arithmetic and logic instructions
  - Multiply and divide instructions
  - Powerful bit-manipulation instructions
  - Multiply-and-accumulate instruction
- Eight addressing modes
  - Register direct [Rn]
  - Register indirect [@ERn]
  - Register indirect with displacement [@(d:16,ERn) or @(d:32,ERn)]
  - Register indirect with post-increment or pre-decrement [@ERn+ or @–ERn]
  - Absolute address [@aa:8, @aa:16, @aa:24, or @aa:32]
  - Immediate [#xx:8, #xx:16, or #xx:32]
  - Program-counter relative [@(d:8,PC) or @(d:16,PC)]
  - Memory indirect [@@aa:8]
- 16-Mbyte address space
  - Program: 16 Mbytes
  - Data: 16 Mbytes (4 Gbytes architecturally)
- High-speed operation
  - All frequently used instructions execute in one to two states
  - Maximum operating frequency: 25 MHz
  - 8/16/32-bit register-register add/subtract: 40 ns
  - 8 × 8-bit register-register multiply: 120 ns

— 16 ÷ 8-bit register-register divide: 480 ns
— 16 × 16-bit register-register multiply: 160 ns
— 32 ÷ 16-bit register-register divide: 800 ns

- CPU operating mode
  — Advanced mode
- Power-down state
  — Transition to power-down state by SLEEP instruction
  — CPU operating clock can be selected

Differences between H8S/2600 CPU and H8S/2000 CPU: The differences between the H8S/2600 CPU and the H8S/2000 CPU are as shown below.

- Register configuration
  The MAC register is supported only by the H8S/2600 CPU.
- Basic instructions
  The four instructions MAC, CLRMAC, LDMAC, and STMAC are supported only by the H8S/2600 CPU.
- Number of execution states
  The number of execution states of the MULXU and MULXS instructions.

		Execution States	
Instruction	**Mnemonic**	**H8S/2600**	**H8S/2000**
MULXU	MULXU.B Rs, Rd	3	12
	MULXU.W Rs, ERd	4	20
MULXS	MULXS.B Rs, Rd	4	13
	MULXS.W Rs, ERd	5	21

In addition, there are differences in address space, CCR and EXR register functions, power-down modes, etc., depending on the model. Refer to the hardware manual for the relevant product for details.

Differences from H8/300 CPU: In comparison to the H8/300 CPU, the H8S/2600 CPU has the following enhancements.

- More general registers and control registers
  — Eight 16-bit expanded registers, and one 8-bit and two 32-bit control registers, have been added.
- Address space
  — The 16-Mbyte address space can be used effectively

- Enhanced addressing
  - The addressing modes have been enhanced to make effective use of the 16-Mbyte address space.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - Signed multiply and divide instructions have been added.
  - Two-bit shift instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions execute twice as fast.

Differences from H8/300H CPU: In comparison to the H8/300H CPU, the H8S/2600 CPU has the following enhancements.

- Additional control register
  - One 8-bit and two 32-bit control registers have been added.
- Enhanced instructions
  - Addressing modes of bit-manipulation instructions have been enhanced.
  - A multiply-and-accumulate instruction has been added.
  - Two-bit shift instructions have been added.
  - Instructions for saving and restoring multiple registers have been added.
  - A test and set instruction has been added.
- Higher speed
  - Basic instructions execute twice as fast.

2.2 Register Configuration

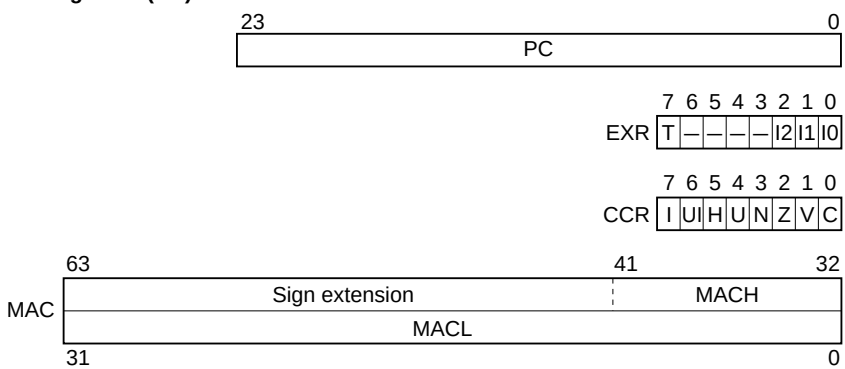
The H8S/2600 CPU has general registers and control registers. The eight 32-bit general registers all have identical functions and can be used as either address registers or data registers. The control registers are the 24-bit program counter (PC), 8-bit extended register (EXR), 8-bit condition code register (CCR), and 64-bit multiply-and-accumulate register (MAC).

CPU Internal Registers

General Registers (Rn) and Extended Registers (En)

	15 07	07 0	
ER0	E0	R0H	R0L
ER1	E1	R1H	R1L
ER2	E2	R2H	R2L
ER3	E3	R3H	R3L
ER4	E4	R4H	R4L
ER5	E5	R5H	R5L
ER6	E6	R6H	R6L
ER7 (SP)	E7	R7H	R7L

Control Registers (CR)



Legend:

SP:	Stack pointer	H:	Half-carry flag
PC:	Program counter	U:	User bit
EXR:	Extend register	N:	Negative flag
T:	Trace bit	Z:	Zero flag
I2 to I0:	Interrupt mask bits	V:	Overflow flag
CCR:	Condition-code register	C:	Carry flag
I:	Interrupt mask bit	MAC:	Multiply-accumulate register
UI:	User bit or interrupt mask bit*		

Note: * In these series, this bit cannot be used as an interrupt mask.

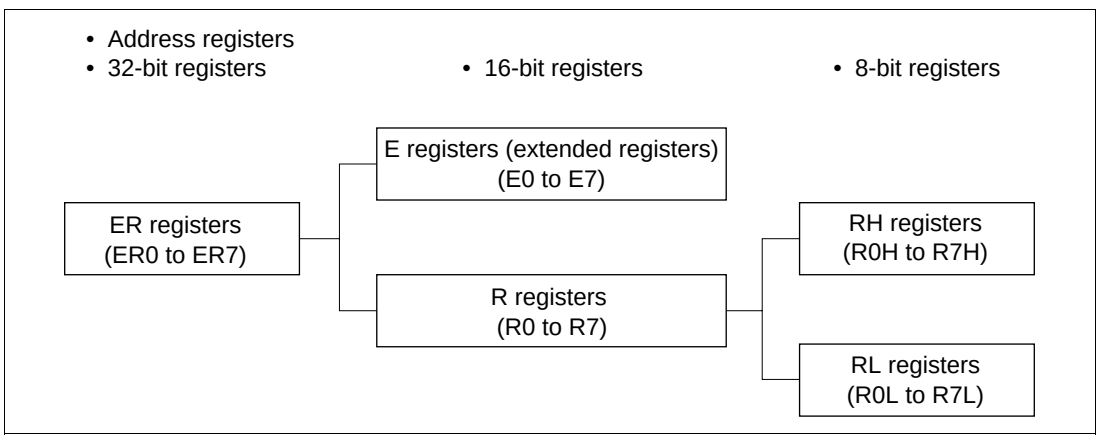
General Registers: The CPU has eight 32-bit general registers. These general registers are all functionally alike and can be used as both address registers and data registers. When a general register is used as a data register, it can be accessed as a 32-bit, 16-bit, or 8-bit register. When the general registers are used as 32-bit registers or address registers, they are designated by the letters ER (ER0 to ER7).

The ER registers divide into 16-bit general registers designated by the letters E (E0 to E7) and R (R0 to R7). These registers are functionally equivalent, providing a maximum sixteen 16-bit registers. The E registers (E0 to E7) are also referred to as extended registers.

The R registers divide into 8-bit general registers designated by the letters RH (R0H to R7H) and RL (R0L to R7L). These registers are functionally equivalent, providing a maximum sixteen 8-bit registers.

The figure below illustrates the usage of the general registers. The usage of each register can be selected independently.

Usage of General Registers



Control Registers: The control registers are the 24-bit program counter (PC), 8-bit extend register (EXR), 8-bit condition-code register (CCR), and 64-bit multiply-accumulate register (MAC).

- **Program Counter (PC)**

  This 24-bit counter indicates the address of the next instruction the CPU will execute. The length of all CPU instructions is 2 bytes (one word), so the least significant PC bit is ignored. (When an instruction code is read, the least significant bit is regarded as 0.)

- **Extend Register (EXR)**

  This 8-bit register comprises a trace bit (T) and interrupt mask bits (I2 to I0).

  — Bit 7—Trace Bit (T)

  Specifies whether or not trace mode is set. When this bit is cleared to 0, instructions are executed sequentially. When set to 1, trace exception handling is started each time an instruction is executed.

  — Bits 6 to 3—Reserved

  — Bits 2 to 0—Interrupt Mask Bits (I2 to I0)

  These bits specify the interrupt request mask level (0 to 7). See section 2.9, Interrupts, for details.

  EXR can be manipulated by the LDC, STC, ANDC, ORC, and XORC instructions. Except in the case of STC, interrupts (including NMI) are not accepted for 3 states after the instruction is executed.

- **Condition-Code Register (CCR)**

  This 8-bit register contains internal CPU status information, including an interrupt mask bit (I) and half-carry (H), negative (N), zero (Z), overflow (V), and carry (C) flags.

  — Bit 7—Interrupt Mask Bit (I)

  Masks interrupts other than NMI when set to 1. (NMI is accepted regardless of the I bit setting.) The I bit is set to 1 when exception handling execution is started. For details, refer to section 2.9, Interrupts.

  — Bit 6—User Bit or Interrupt Mask Bit (UI)

  Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions. In this series, this bit cannot be used as an interrupt mask.

  — Bit 5—Half-Carry Flag (H)

  When the ADD.B, ADDX.B, SUB.B, SUBX.B, CMP.B, or NEG.B instruction is executed, this flag is set to 1 if there is a carry or borrow at bit 3, and cleared to 0 otherwise. When the ADD.W, SUB.W, CMP.W, or NEG.W instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 11, and cleared to 0 otherwise. When the ADD.L, SUB.L, CMP.L, or NEG.L instruction is executed, the H flag is set to 1 if there is a carry or borrow at bit 27, and cleared to 0 otherwise.

  — Bit 4—User Bit (U)

  Can be written and read by software using the LDC, STC, ANDC, ORC, and XORC instructions.

  — Bit 3—Negative Flag (N)

  Stores the value of the most significant bit (sign bit) of data.

  — Bit 2—Zero Flag (Z)

  Set to 1 to indicate zero data, and cleared to 0 to indicate non-zero data.

  — Bit 1—Overflow Flag (V)

  Set to 1 when an arithmetic overflow occurs, and cleared to 0 at other times.

— Bit 0—Carry Flag (C)

Set to 1 when a carry occurs, and cleared to 0 otherwise. Used by:

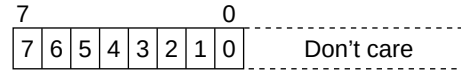
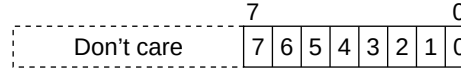
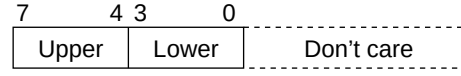
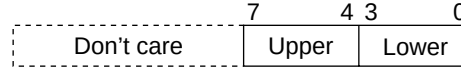
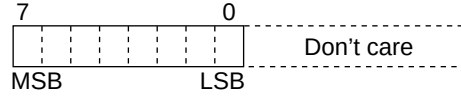
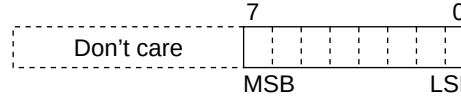
- Add instructions, to indicate a carry
- Subtract instructions, to indicate a borrow
- Shift and rotate instructions, to store the carry

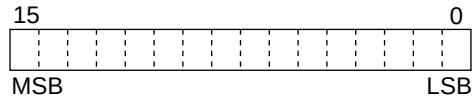
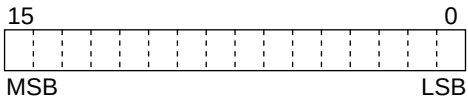
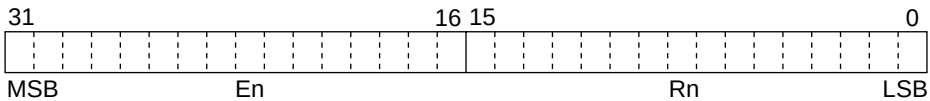
The carry flag is also used as a bit accumulator by bit-manipulation instructions.

2.3 Data Formats

The CPU can process 1-bit, 4-bit (BCD), 8-bit (byte), 16-bit (word), and 32-bit (longword) data. Bit-manipulation instructions operate on 1-bit data by accessing bit n (n = 0, 1, 2, …, 7) of byte operand data. The DAA and DAS decimal-adjust instructions treat byte data as two digits of 4-bit BCD data.

General Register Data Formats

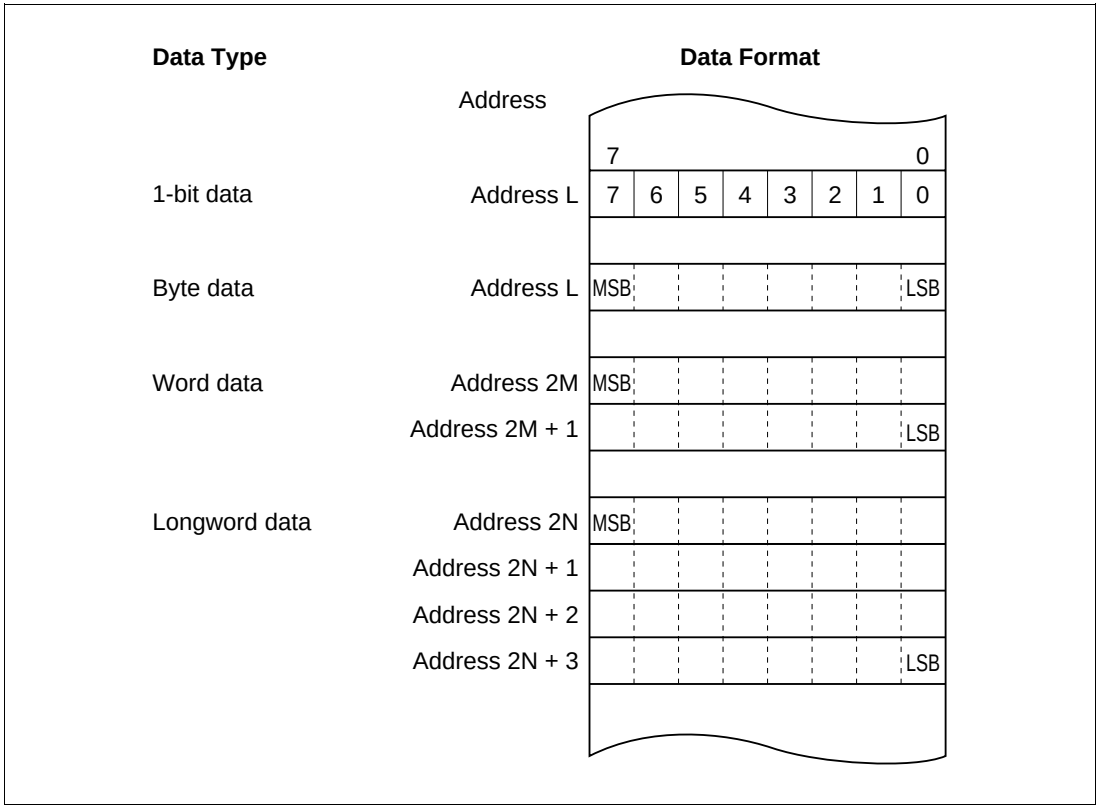
Data Type	Register Number	Data Format
1-bit data	RnH	
1-bit data	RnL	
4-bit BCD data	RnH	
4-bit BCD data	RnL	
Byte data	RnH	
Byte data	RnL	

Data Type	Register Number	Data Format
Word data	Rn	
Word data	En	
Longword data	ERn	

Legend:

ERn: General register ER
En: General register E
Rn: General register R
RnH: General register RH
RnL: General register RL
MSB: Most significant bit
LSB: Least significant bit

Memory Data Formats



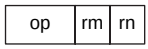
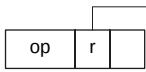
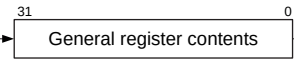
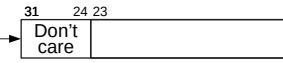
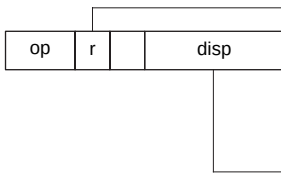
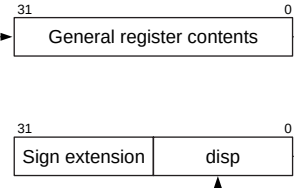
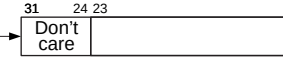
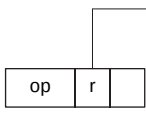
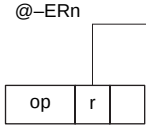
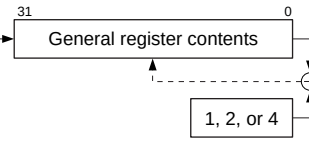
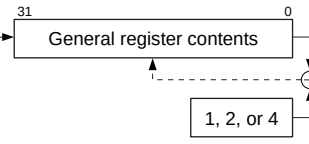
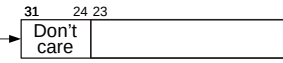
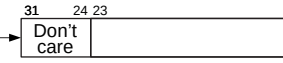
2.4 Addressing Modes

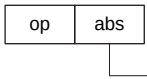
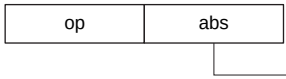
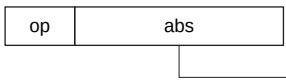
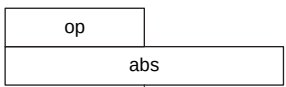
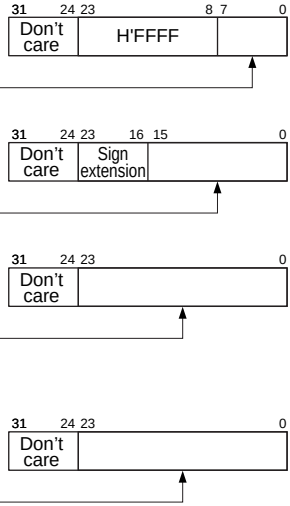
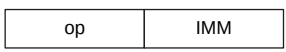
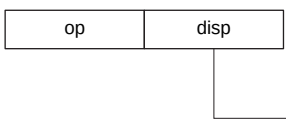
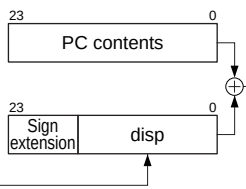
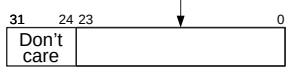
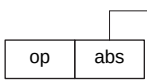
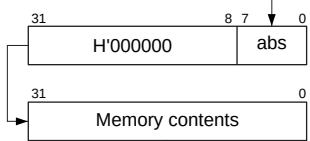
The H8S/2600 CPU supports eight addressing modes.

Addressing Modes

No.	Addressing Mode	Symbol
1	Register direct	Rn
2	Register indirect	@ERn
3	Register indirect with displacement	@(d:16,ERn)/@(d:32,ERn)
4	Register indirect with post-increment Register indirect with pre-decrement	@ERn+ @–ERn
5	Absolute address	@aa:8/@aa:16/@aa:24/@aa:32
6	Immediate	#xx:8/#xx:16/#xx:32
7	Program-counter relative	@(d:8,PC)/@(d:16,PC)
8	Memory indirect	@@aa:8

Effective Address Calculation

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
1	Register direct (Rn) 		Operand is general register contents.
2	Register indirect (@ERn) 		
3	Register indirect with displacement @(d:16, ERn) or @(d:32, ERn) 		
4	Register indirect with post-increment or pre-decrement • Register indirect with post-increment @ERn+  • Register indirect with pre-decrement @–ERn 	  Operand Size / Value Added or Subtracted Byte: 1 Word: 2 Longword: 4	 

No.	Addressing Mode and Instruction Format	Effective Address Calculation	Effective Address (EA)
5	Absolute address @aa:8  @aa:16  @aa:24  @aa:32  		
6	Immediate #xx:8/#xx:16/#xx:32  		Operand is immediate data.
7	Program-counter relative @(d:8, PC)/@(d:16, PC)  	 	
8	Memory indirect @@aa:8 • Advanced mode  	 	

2.5 Instruction Set

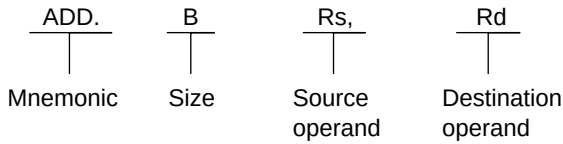
The H8S/2600 CPU has 69 types of instructions.

Features

- Upward-compatible at object level with H8/300H and H8/300 CPUs
- General register architecture
- 8/16/32-bit transfer instructions and arithmetic and logic instructions
  - — Byte (B), word (W), and longword (L) formats for transfer instructions and basic arithmetic and logic instructions
- Unsigned and signed multiply and divide instructions
- Powerful bit-manipulation instructions
- Instructions for saving and restoring multiple registers

Assembler Format

The ADD instruction format is shown below as an example.



- Data transfer Instructions

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Advanced
MOV	MOV.B #xx:8,Rd	B	2									#xx:8→Rd8	—	—	↕	↕	0	—	1
	MOV.B Rs,Rd	B		2								Rs8→Rd8	—	—	↕	↕	0	—	1
	MOV.B @ERs,Rd	B			2							@ERs→Rd8	—	—	↕	↕	0	—	2
	MOV.B @(d:16,ERs),Rd	B				4						@(d:16,ERs)→Rd8	—	—	↕	↕	0	—	3
	MOV.B @(d:32,ERs),Rd	B				8						@(d:32,ERs)→Rd8	—	—	↕	↕	0	—	5
	MOV.B @ERs+,Rd	B					2					@ERs→Rd8,ERs32+1→ERs32	—	—	↕	↕	0	—	3
	MOV.B @aa:8,Rd	B						2				@aa:8→Rd8	—	—	↕	↕	0	—	2
	MOV.B @aa:16,Rd	B						4				@aa:16→Rd8	—	—	↕	↕	0	—	3
	MOV.B @aa:32,Rd	B						6				@aa:32→Rd8	—	—	↕	↕	0	—	4
	MOV.B Rs,@ERd	B			2							Rs8→@ERd	—	—	↕	↕	0	—	2
	MOV.B Rs,@(d:16,ERd)	B				4						Rs8→@(d:16,ERd)	—	—	↕	↕	0	—	3
	MOV.B Rs,@(d:32,ERd)	B				8						Rs8→@(d:32,ERd)	—	—	↕	↕	0	—	5
	MOV.B Rs,@-ERd	B					2					ERd32-1→ERd32,Rs8→@ERd	—	—	↕	↕	0	—	3
	MOV.B Rs,@aa:8	B						2				Rs8→@aa:8	—	—	↕	↕	0	—	2
	MOV.B Rs,@aa:16	B						4				Rs8→@aa:16	—	—	↕	↕	0	—	3
	MOV.B Rs,@aa:32	B						6				Rs8→@aa:32	—	—	↕	↕	0	—	4
	MOV.W #xx:16,Rd	W	4									#xx:16→Rd16	—	—	↕	↕	0	—	2
	MOV.W Rs,Rd	W		2								Rs16→Rd16	—	—	↕	↕	0	—	1
	MOV.W @ERs,Rd	W			2							@ERs→Rd16	—	—	↕	↕	0	—	2
	MOV.W @(d:16,ERs),Rd	W				4						@(d:16,ERs)→Rd16	—	—	↕	↕	0	—	3
	MOV.W @(d:32,ERs),Rd	W				8						@(d:32,ERs)→Rd16	—	—	↕	↕	0	—	5
	MOV.W @ERs+,Rd	W					2					@ERs→Rd16,ERs32+2→ERs32	—	—	↕	↕	0	—	3
	MOV.W @aa:16,Rd	W						4				@aa:16→Rd16	—	—	↕	↕	0	—	3
	MOV.W @aa:32,Rd	W						6				@aa:32→Rd16	—	—	↕	↕	0	—	4
	MOV.W Rs,@ERd	W			2							Rs16→@ERd	—	—	↕	↕	0	—	2
	MOV.W Rs,@(d:16,ERd)	W				4						Rs16→@(d:16,ERd)	—	—	↕	↕	0	—	3
	MOV.W Rs,@(d:32,ERd)	W				8						Rs16→@(d:32,ERd)	—	—	↕	↕	0	—	5
	MOV.W Rs,@-ERd	W					2					ERd32-2→ERd32,Rs16→@ERd	—	—	↕	↕	0	—	3
	MOV.W Rs,@aa:16	W						4				Rs16→@aa:16	—	—	↕	↕	0	—	3
	MOV.W Rs,@aa:32	W						6				Rs16→@aa:32	—	—	↕	↕	0	—	4
	MOV.L #xx:32,ERd	L	6									#xx:32→ERd32	—	—	↕	↕	0	—	3
	MOV.L ERs,ERd	L		2								ERs32→ERd32	—	—	↕	↕	0	—	1
	MOV.L @ERs,ERd	L			4							@ERs→ERd32	—	—	↕	↕	0	—	4
	MOV.L @(d:16,ERs),ERd	L				6						@(d:16,ERs)→ERd32	—	—	↕	↕	0	—	5
	MOV.L @(d:32,ERs),ERd	L				10						@(d:32,ERs)→ERd32	—	—	↕	↕	0	—	7
	MOV.L @ERs+,ERd	L					4					@ERs→ERd32,ERs32+4→ERs32	—	—	↕	↕	0	—	5
	MOV.L @aa:16,ERd	L						6				@aa:16→ERd32	—	—	↕	↕	0	—	5
	MOV.L @aa:32,ERd	L						8				@aa:32→ERd32	—	—	↕	↕	0	—	6
	MOV.L ERs,@ERd	L			4							ERs32→@ERd	—	—	↕	↕	0	—	4
	MOV.L ERs,@(d:16,ERd)	L				6						ERs32→@(d:16,ERd)	—	—	↕	↕	0	—	5
	MOV.L ERs,@(d:32,ERd)	L				10						ERs32→@(d:32,ERd)	—	—	↕	↕	0	—	7
	MOV.L ERs,@-ERd	L					4					ERd32-4→ERd32,ERs32→@ERd	—	—	↕	↕	0	—	5
	MOV.L ERs,@aa:16	L						6				ERs32→@aa:16	—	—	↕	↕	0	—	5
	MOV.L ERs,@aa:32	L						8				ERs32→@aa:32	—	—	↕	↕	0	—	6
POP	POP.W Rn	W									2	@SP→Rn16,SP+2→SP	—	—	↕	↕	0	—	3
	POP.L ERn	L									4	@SP→ERn32,SP+4→SP	—	—	↕	↕	0	—	5
PUSH	PUSH.W Rn	W									2	SP-2→SP,Rn16→@SP	—	—	↕	↕	0	—	3
	PUSH.L ERn	L									4	SP-4→SP,ERn32→@SP	—	—	↕	↕	0	—	5
LDM	LDM @SP+,(ERm-ERn)	L									4	(@SP→ERn32,SP+4→SP) Repeated for each register restored	—	—	—	—	—	—	7/9/11 [1]
STM	STM (ERm-ERn),@-SP	L									4	(SP-4→SP,ERn32→@SP) Repeated for each register saved	—	—	—	—	—	—	7/9/11 [1]
MOVFPE	MOVFPE @aa:16,Rd	Cannot be used with the H8S/2600 CPU.																	[2]
MOVTPE	MOVTPE Rs,@aa:16																		[2]

- Arithmetic instructions

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Advanced
ADD	ADD.B #xx:8,Rd	B	2									Rd8+#xx:8→Rd8	—	↕	↕	↕	↕	↕	1
	ADD.B Rs,Rd	B		2								Rd8+Rs8→Rd8	—	↕	↕	↕	↕	↕	1
	ADD.W #xx:16,Rd	W	4									Rd16+#xx:16→Rd16	—	[3]	↕	↕	↕	↕	2
	ADD.W Rs,Rd	W		2								Rd16+Rs16→Rd16	—	[3]	↕	↕	↕	↕	1
	ADD.L #xx:32,ERd	L	6									ERd32+#xx:32→ERd32	—	[4]	↕	↕	↕	↕	3
	ADD.L ERs,ERd	L		2								ERd32+ERs32→ERd32	—	[4]	↕	↕	↕	↕	1
ADDX	ADDX #xx:8,Rd	B	2									Rd8+#xx:8+C→Rd8	—	↕	↕	[5]	↕	↕	1
	ADDX Rs,Rd	B		2								Rd8+Rs8+C→Rd8	—	↕	↕	[5]	↕	↕	1
ADDS	ADDS #1,ERd	L		2								ERd32+1→ERd32	—	—	—	—	—	—	1
	ADDS #2,ERd	L		2								ERd32+2→ERd32	—	—	—	—	—	—	1
	ADDS #4,ERd	L		2								ERd32+4→ERd32	—	—	—	—	—	—	1
INC	INC.B Rd	B		2								Rd8+1→Rd8	—	—	↕	↕	↕	—	1
	INC.W #1,Rd	W		2								Rd16+1→Rd16	—	—	↕	↕	↕	—	1
	INC.W #2,Rd	W		2								Rd16+2→Rd16	—	—	↕	↕	↕	—	1
	INC.L #1,ERd	L		2								ERd32+1→ERd32	—	—	↕	↕	↕	—	1
	INC.L #2,ERd	L		2								ERd32+2→ERd32	—	—	↕	↕	↕	—	1
DAA	DAA Rd	B		2								Rd8 decimal adjust → Rd8	—	*	↕	↕	*	↕	1
SUB	SUB.B Rs,Rd	B		2								Rd8-Rs8→Rd8	—	↕	↕	↕	↕	↕	1
	SUB.W #xx:16,Rd	W	4									Rd16-#xx:16→Rd16	—	[3]	↕	↕	↕	↕	2
	SUB.W Rs,Rd	W		2								Rd16-Rs16→Rd16	—	[3]	↕	↕	↕	↕	1
	SUB.L #xx:32,ERd	L	6									ERd32-#xx:32→ERd32	—	[4]	↕	↕	↕	↕	3
	SUB.L ERs,ERd	L		2								ERd32-ERs32→ERd32	—	[4]	↕	↕	↕	↕	1
SUBX	SUBX #xx:8,Rd	B	2									Rd8-#xx:8-C→Rd8	—	↕	↕	[5]	↕	↕	1
	SUBX Rs,Rd	B		2								Rd8-Rs8-C→Rd8	—	↕	↕	[5]	↕	↕	1
SUBS	SUBS #1,ERd	L		2								ERd32-1→ERd32	—	—	—	—	—	—	1
	SUBS #2,ERd	L		2								ERd32-2→ERd32	—	—	—	—	—	—	1
	SUBS #4,ERd	L		2								ERd32-4→ERd32	—	—	—	—	—	—	1
DEC	DEC.B Rd	B		2								Rd8-1→Rd8	—	—	↕	↕	↕	—	1
	DEC.W #1,Rd	W		2								Rd16-1→Rd16	—	—	↕	↕	↕	—	1
	DEC.W #2,Rd	W		2								Rd16-2→Rd16	—	—	↕	↕	↕	—	1
	DEC.L #1,ERd	L		2								ERd32-1→ERd32	—	—	↕	↕	↕	—	1
	DEC.L #2,ERd	L		2								ERd32-2→ERd32	—	—	↕	↕	↕	—	1
DAS	DAS Rd	B		2								Rd8 decimal adjust → Rd8	—	*	↕	↕	*	—	1
MULXU	MULXU.B Rs,Rd	B		2								Rd8×Rs8→Rd16 (unsigned multiplication)	—	—	—	—	—	—	12
	MULXU.W Rs,ERd	W		2								Rd16×Rs16→ERd32 (unsigned multiplication)	—	—	—	—	—	—	20
MULXS	MULXS.B Rs,Rd	B		4								Rd8×Rs8→Rd16 (signed multiplication)	—	—	↕	↕	—	—	13
	MULXS.W Rs,ERd	W		4								Rd16×Rs16→ERd32 (signed multiplication)	—	—	↕	↕	—	—	21
DIVXU	DIVXU.B Rs,Rd	B		2								Rd16÷Rs8→Rd16 (RdH: remainder, RdL: quotient) (unsigned division)	—	—	[6]	[7]	—	—	12
	DIVXU.W Rs,ERd	W		2								ERd32÷Rs16→ERd32 (Ed: remainder, Rd: quotient) (unsigned division)	—	—	[6]	[7]	—	—	20
DIVXS	DIVXS.B Rs,Rd	B		4								Rd16÷Rs8→Rd16 (RdH: remainder, RdL: quotient) (signed division)	—	—	[8]	[7]	—	—	13
	DIVXS.W Rs,ERd	W		4								ERd32÷Rs16→ERd32 (Ed: remainder, Rd: quotient) (signed division)	—	—	[8]	[7]	—	—	21
CMP	CMP.B #xx:8,Rd	B	2									Rd8-#xx:8	—	↕	↕	↕	↕	↕	1
	CMP.B Rs,Rd	B		2								Rd8-Rs8	—	↕	↕	↕	↕	↕	1
	CMP.W #xx:16,Rd	W	4									Rd16-#xx:16	—	[3]	↕	↕	↕	↕	2
	CMP.W Rs,Rd	W		2								Rd16-Rs16	—	[3]	↕	↕	↕	↕	1
	CMP.L #xx:32,ERd	L	6									ERd32-#xx:32	—	[4]	↕	↕	↕	↕	3
	CMP.L ERs,ERd	L		2								ERd32-ERs32	—	[4]	↕	↕	↕	↕	1
NEG	NEG.B Rd	B		2								0-Rd8→Rd8	—	↕	↕	↕	↕	↕	1
	NEG.W Rd	W		2								0-Rd16→Rd16	—	↕	↕	↕	↕	↕	1
	NEG.L ERd	L		2								0-ERd32→ERd32	—	↕	↕	↕	↕	↕	1
EXTU	EXTU.W Rd	W		2								0 → (<bits 15 to 8> of Rd16)	—	—	0	↕	0	—	1
	EXTU.L ERd	L		2								0 → (<bits 31 to 16> of ERd32)	—	—	0	↕	0	—	1

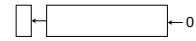
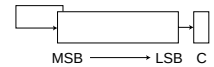
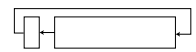
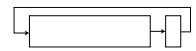
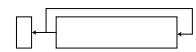
- Arithmetic instructions (cont)

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Advanced
EXTS	EXTS.W Rd	W		2								(<bit 7> of Rd16) → (<bits 15 to 8> of Rd16)	—	—	↕	↕	0	—	1
	EXTS.L ERd	L		2								(<bit 15> of ERd32) → (<bits 31 to 16> of ERd32)	—	—	↕	↕	0	—	1
TAS	TAS @ERd	B			4							@ERd-0 → CRR set, (1) → (<bit 7> of @ERd)	—	—	↕	↕	0	—	4
MAC	MAC @ERn+,@ERm+	—					4					@ERn×@ERm+MAC→MAC (signed multiplication) @ERn+2→ERn, ERm+2→ERm	—	—	— [11]	— [11]	— [11]	—	4
CLRMAC	CLRMAC	—									2	0→MACH, MACL	—	—	—	—	—	—	2 [12]
LDMAC	LDMAC ERs,MACH	L		2								ERs→MACH	—	—	—	—	—	—	2 [12]
	LDMAC ERs,MACL	L		2								ERs→MACL	—	—	—	—	—	—	2 [12]
STMAC	STMAC MACH,ERd	L		2								MACH→ERd	—	—	↕	↕	↕	—	1 [12]
	STMAC MACL,ERd	L		2								MACL→ERd	—	—	↕	↕	↕	—	1 [12]

- Logical instructions

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Advanced
AND	AND.B #xx:8,Rd	B	2									Rd8∧#xx:8→Rd8	—	—	↕	↕	0	—	1
	AND.B Rs,Rd	B		2								Rd8∧Rs8→Rd8	—	—	↕	↕	0	—	1
	AND.W #xx:16,Rd	W	4									Rd16∧#xx:16→Rd16	—	—	↕	↕	0	—	2
	AND.W Rs,Rd	W		2								Rd16∧Rs16→Rd16	—	—	↕	↕	0	—	1
	AND.L #xx:32,ERd	L	6									ERd32∧#xx:32→ERd32	—	—	↕	↕	0	—	3
	AND.L ERs,ERd	L		4								ERd32∧ERs32→ERd32	—	—	↕	↕	0	—	2
OR	OR.B #xx:8,Rd	B	2									Rd8∨#xx:8→Rd8	—	—	↕	↕	0	—	1
	OR.B Rs,Rd	B		2								Rd8∨Rs8→Rd8	—	—	↕	↕	0	—	1
	OR.W #xx:16,Rd	W	4									Rd16∨#xx:16→Rd16	—	—	↕	↕	0	—	2
	OR.W Rs,Rd	W		2								Rd16∨Rs16→Rd16	—	—	↕	↕	0	—	1
	OR.L #xx:32,ERd	L	6									ERd32∨#xx:32→ERd32	—	—	↕	↕	0	—	3
	OR.L ERs,ERd	L		4								ERd32∨ERs32→ERd32	—	—	↕	↕	0	—	2
XOR	XOR.B #xx:8,Rd	B	2									Rd8⊕#xx:8→Rd8	—	—	↕	↕	0	—	1
	XOR.B Rs,Rd	B		2								Rd8⊕Rs8→Rd8	—	—	↕	↕	0	—	1
	XOR.W #xx:16,Rd	W	4									Rd16⊕#xx:16→Rd16	—	—	↕	↕	0	—	2
	XOR.W Rs,Rd	W		2								Rd16⊕Rs16→Rd16	—	—	↕	↕	0	—	1
	XOR.L #xx:32,ERd	L	6									ERd32⊕#xx:32→ERd32	—	—	↕	↕	0	—	3
	XOR.L ERs,ERd	L		4								ERd32⊕ERs32→ERd32	—	—	↕	↕	0	—	2
NOT	NOT.B Rd	B		2								¬Rd8→Rd8	—	—	↕	↕	0	—	1
	NOT.W Rd	W		2								¬Rd16→Rd16	—	—	↕	↕	0	—	1
	NOT.L ERd	L		2								¬ERd32→ERd32	—	—	↕	↕	0	—	1

- Shift instructions

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Advanced
SHAL	SHAL.B Rd	B		2									—	—	↕	↕	↕	↕	1
	SHAL.B #2,Rd	B		2									—	—	↕	↕	↕	↕	1
	SHAL.W Rd	W		2									—	—	↕	↕	↕	↕	1
	SHAL.W #2,Rd	W		2									—	—	↕	↕	↕	↕	1
	SHAL.L ERd	L		2									—	—	↕	↕	↕	↕	1
	SHAL.L #2,ERd	L		2									—	—	↕	↕	↕	↕	1
SHAR	SHAR.B Rd	B		2									—	—	↕	↕	0	↕	1
	SHAR.B #2,Rd	B		2									—	—	↕	↕	0	↕	1
	SHAR.W Rd	W		2									—	—	↕	↕	0	↕	1
	SHAR.W #2,Rd	W		2									—	—	↕	↕	0	↕	1
	SHAR.L ERd	L		2									—	—	↕	↕	0	↕	1
	SHAR.L #2,ERd	L		2									—	—	↕	↕	0	↕	1
SHLL	SHLL.B Rd	B		2									—	—	↕	↕	0	↕	1
	SHLL.B #2,Rd	B		2									—	—	↕	↕	0	↕	1
	SHLL.W Rd	W		2									—	—	↕	↕	0	↕	1
	SHLL.W #2,Rd	W		2									—	—	↕	↕	0	↕	1
	SHLL.L ERd	L		2									—	—	↕	↕	0	↕	1
	SHLL.L #2,ERd	L		2									—	—	↕	↕	0	↕	1
SHLR	SHLR.B Rd	B		2									—	—	0	↕	0	↕	1
	SHLR.B #2,Rd	B		2									—	—	0	↕	0	↕	1
	SHLR.W Rd	W		2									—	—	0	↕	0	↕	1
	SHLR.W #2,Rd	W		2									—	—	0	↕	0	↕	1
	SHLR.L ERd	L		2									—	—	0	↕	0	↕	1
	SHLR.L #2,ERd	L		2									—	—	0	↕	0	↕	1
ROTXL	ROTXL.B Rd	B		2									—	—	↕	↕	0	↕	1
	ROTXL.B #2,Rd	B		2									—	—	↕	↕	0	↕	1
	ROTXL.W Rd	W		2									—	—	↕	↕	0	↕	1
	ROTXL.W #2,Rd	W		2									—	—	↕	↕	0	↕	1
	ROTXL.L ERd	L		2									—	—	↕	↕	0	↕	1
	ROTXL.L #2,ERd	L		2									—	—	↕	↕	0	↕	1
ROTXR	ROTXR.B Rd	B		2									—	—	↕	↕	0	↕	1
	ROTXR.B #2,Rd	B		2									—	—	↕	↕	0	↕	1
	ROTXR.W Rd	W		2									—	—	↕	↕	0	↕	1
	ROTXR.W #2,Rd	W		2									—	—	↕	↕	0	↕	1
	ROTXR.L ERd	L		2									—	—	↕	↕	0	↕	1
	ROTXR.L #2,ERd	L		2									—	—	↕	↕	0	↕	1
ROTL	ROTL.B Rd	B		2									—	—	↕	↕	0	↕	1
	ROTL.B #2,Rd	B		2									—	—	↕	↕	0	↕	1
	ROTL.W Rd	W		2									—	—	↕	↕	0	↕	1
	ROTL.W #2,Rd	W		2									—	—	↕	↕	0	↕	1
	ROTL.L ERd	L		2									—	—	↕	↕	0	↕	1
	ROTL.L #2,ERd	L		2									—	—	↕	↕	0	↕	1
ROTR	ROTR.B Rd	B		2									—	—	↕	↕	0	↕	1
	ROTR.B #2,Rd	B		2									—	—	↕	↕	0	↕	1
	ROTR.W Rd	W		2									—	—	↕	↕	0	↕	1
	ROTR.W #2,Rd	W		2									—	—	↕	↕	0	↕	1
	ROTR.L ERd	L		2									—	—	↕	↕	0	↕	1
	ROTR.L #2,ERd	L		2									—	—	↕	↕	0	↕	1

- Bit manipulation instructions

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Advanced
BSET	BSET #xx:3,Rd	B		2								(#xx:3 of Rd8)←1	—	—	—	—	—		1
	BSET #xx:3,@ERd	B			4							(#xx:3 of @ERd)←1	—	—	—	—	—		4
	BSET #xx:3,@aa:8	B						4				(#xx:3 of @aa:8)←1	—	—	—	—	—		4
	BSET #xx:3,@aa:16	B						6				(#xx:3 of @aa:16)←1	—	—	—	—	—		5
	BSET #xx:3,@aa:32	B						8				(#xx:3 of @aa:32)←1	—	—	—	—	—		6
	BSET Rn,Rd	B		2								(Rn8 of Rd8)←1	—	—	—	—	—		1
	BSET Rn,@ERd	B			4							(Rn8 of @ERd)←1	—	—	—	—	—		4
	BSET Rn,@aa:8	B						4				(Rn8 of @aa:8)←1	—	—	—	—	—		4
	BSET Rn,@aa:16	B						6				(Rn8 of @aa:16)←1	—	—	—	—	—		5
	BSET Rn,@aa:32	B						8				(Rn8 of @aa:32)←1	—	—	—	—	—		6
BCLR	BCLR #xx:3,Rd	B		2								(#xx:3 of Rd8)←0	—	—	—	—	—		1
	BCLR #xx:3,@ERd	B			4							(#xx:3 of @ERd)←0	—	—	—	—	—		4
	BCLR #xx:3,@aa:8	B						4				(#xx:3 of @aa:8)←0	—	—	—	—	—		4
	BCLR #xx:3,@aa:16	B						6				(#xx:3 of @aa:16)←0	—	—	—	—	—		5
	BCLR #xx:3,@aa:32	B						8				(#xx:3 of @aa:32)←0	—	—	—	—	—		6
	BCLR Rn,Rd	B		2								(Rn8 of Rd8)←0	—	—	—	—	—		1
	BCLR Rn,@ERd	B			4							(Rn8 of @ERd)←0	—	—	—	—	—		4
	BCLR Rn,@aa:8	B						4				(Rn8 of @aa:8)←0	—	—	—	—	—		4
	BCLR Rn,@aa:16	B						6				(Rn8 of @aa:16)←0	—	—	—	—	—		5
	BCLR Rn,@aa:32	B						8				(Rn8 of @aa:32)←0	—	—	—	—	—		6
BNOT	BNOT #xx:3,Rd	B		2								(#xx:3 of Rd8)← [¬(#xx:3 of Rd8)]	—	—	—	—	—		1
	BNOT #xx:3,@ERd	B			4							(#xx:3 of @ERd)← [¬(#xx:3 of @ERd)]	—	—	—	—	—		4
	BNOT #xx:3,@aa:8	B						4				(#xx:3 of @aa:8)← [¬(#xx:3 of @aa:8)]	—	—	—	—	—		4
	BNOT #xx:3,@aa:16	B						6				(#xx:3 of @aa:16)←[¬(#xx:3 of @aa:16)]	—	—	—	—	—		5
	BNOT #xx:3,@aa:32	B						8				(#xx:3 of @aa:32)← [¬(#xx:3 of @aa:32)]	—	—	—	—	—		6
	BNOT Rn,Rd	B		2								(Rn8 of Rd8)← [¬(Rn8 of Rd8)]	—	—	—	—	—		1
	BNOT Rn,@ERd	B			4							(Rn8 of @ERd)← [¬(Rn8 of @ERd)]	—	—	—	—	—		4
	BNOT Rn,@aa:8	B						4				(Rn8 of @aa:8)← [¬(Rn8 of @aa:8)]	—	—	—	—	—		4
	BNOT Rn,@aa:16	B						6				(Rn8 of @aa:16)← [¬(Rn8 of @aa:16)]	—	—	—	—	—		5
	BNOT Rn,@aa:32	B						8				(Rn8 of @aa:32)← [¬(Rn8 of @aa:32)]	—	—	—	—	—		6
BTST	BTST #xx:3,Rd	B		2								¬(#xx:3 of Rd8)→Z	—	—	—	↕	—		1
	BTST #xx:3,@ERd	B			4							¬(#xx:3 of @ERd)→Z	—	—	—	↕	—		3
	BTST #xx:3,@aa:8	B						4				¬(#xx:3 of @aa:8)→Z	—	—	—	↕	—		3
	BTST #xx:3,@aa:16	B						6				¬(#xx:3 of @aa:16)→Z	—	—	—	↕	—		4
	BTST #xx:3,@aa:32	B						8				¬(#xx:3 of @aa:32)→Z	—	—	—	↕	—		5
	BTST Rn,Rd	B		2								¬(Rn8 of Rd8)→Z	—	—	—	↕	—		1
	BTST Rn,@ERd	B			4							¬(Rn8 of @ERd)→Z	—	—	—	↕	—		3
	BTST Rn,@aa:8	B						4				¬(Rn8 of @aa:8)→Z	—	—	—	↕	—		3
	BTST Rn,@aa:16	B						6				¬(Rn8 of @aa:16)→Z	—	—	—	↕	—		4
	BTST Rn,@aa:32	B						8				¬(Rn8 of @aa:32)→Z	—	—	—	↕	—		5
BLD	BLD #xx:3,Rd	B		2								(#xx:3 of Rd8)→C	—	—	—	—	—	↕	1
	BLD #xx:3,@ERd	B			4							(#xx:3 of @ERd)→C	—	—	—	—	—	↕	3
	BLD #xx:3,@aa:8	B						4				(#xx:3 of @aa:8)→C	—	—	—	—	—	↕	3
	BLD #xx:3,@aa:16	B						6				(#xx:3 of @aa:16)→C	—	—	—	—	—	↕	4
	BLD #xx:3,@aa:32	B						8				(#xx:3 of @aa:32)→C	—	—	—	—	—	↕	5
BILD	BILD #xx:3,Rd	B		2								¬(#xx:3 of Rd8)→C	—	—	—	—	—	↕	1
	BILD #xx:3,@ERd	B			4							¬(#xx:3 of @ERd)→C	—	—	—	—	—	↕	3
	BILD #xx:3,@aa:8	B						4				¬(#xx:3 of @aa:8)→C	—	—	—	—	—	↕	3
	BILD #xx:3,@aa:16	B						6				¬(#xx:3 of @aa:16)→C	—	—	—	—	—	↕	4
	BILD #xx:3,@aa:32	B						8				¬(#xx:3 of @aa:32)→C	—	—	—	—	—	↕	5
BST	BST #xx:3,Rd	B		2								C→(#xx:3 of Rd8)	—	—	—	—	—		1
	BST #xx:3,@ERd	B			4							C→(#xx:3 of @ERd)	—	—	—	—	—		4
	BST #xx:3,@aa:8	B						4				C→(#xx:3 of @aa:8)	—	—	—	—	—		4
	BST #xx:3,@aa:16	B						6				C→(#xx:3 of @aa:16)	—	—	—	—	—		5
	BST #xx:3,@aa:32	B						8				C→(#xx:3 of @aa:32)	—	—	—	—	—		6

- Bit manipulation instructions (cont)

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Advanced
BIST	BIST #xx:3,Rd	B		2								¬C→(#xx:3 of Rd8)	—	—	—	—	—	—	1
	BIST #xx:3,@ERd	B			4							¬C→(#xx:3 of @ERd)	—	—	—	—	—	—	4
	BIST #xx:3,@aa:8	B						4				¬C→(#xx:3 of @aa:8)	—	—	—	—	—	—	4
	BIST #xx:3,@aa:16	B						6				¬C→(#xx:3 of @aa:16)	—	—	—	—	—	—	5
	BIST #xx:3,@aa:32	B						8				¬C→(#xx:3 of @aa:32)	—	—	—	—	—	—	6
BAND	BAND #xx:3,Rd	B		2								C∧(#xx:3 of Rd8)→C	—	—	—	—	—	↕	1
	BAND #xx:3,@ERd	B			4							C∧(#xx:3 of @ERd)→C	—	—	—	—	—	↕	3
	BAND #xx:3,@aa:8	B						4				C∧(#xx:3 of @aa:8)→C	—	—	—	—	—	↕	3
	BAND #xx:3,@aa:16	B						6				C∧(#xx:3 of @aa:16)→C	—	—	—	—	—	↕	4
	BAND #xx:3,@aa:32	B						8				C∧(#xx:3 of @aa:32)→C	—	—	—	—	—	↕	5
BIAND	BIAND #xx:3,Rd	B		2								C∧[¬(#xx:3 of Rd8)]→C	—	—	—	—	—	↕	1
	BIAND #xx:3,@ERd	B			4							C∧[¬(#xx:3 of @ERd)]→C	—	—	—	—	—	↕	3
	BIAND #xx:3,@aa:8	B						4				C∧[¬(#xx:3 of @aa:8)]→C	—	—	—	—	—	↕	3
	BIAND #xx:3,@aa:16	B						6				C∧[¬(#xx:3 of @aa:16)]→C	—	—	—	—	—	↕	4
	BIAND #xx:3,@aa:32	B						8				C∧[¬(#xx:3 of @aa:32)]→C	—	—	—	—	—	↕	5
BOR	BOR #xx:3,Rd	B		2								C∨(#xx:3 of Rd8)→C	—	—	—	—	—	↕	1
	BOR #xx:3,@ERd	B			4							C∨(#xx:3 of @ERd)→C	—	—	—	—	—	↕	3
	BOR #xx:3,@aa:8	B						4				C∨(#xx:3 of @aa:8)→C	—	—	—	—	—	↕	3
	BOR #xx:3,@aa:16	B						6				C∨(#xx:3 of @aa:16)→C	—	—	—	—	—	↕	4
	BOR #xx:3,@aa:32	B						8				C∨(#xx:3 of @aa:32)→C	—	—	—	—	—	↕	5
BIOR	BIOR #xx:3,Rd	B		2								C∨[¬(#xx:3 of Rd8)]→C	—	—	—	—	—	↕	1
	BIOR #xx:3,@ERd	B			4							C∨[¬(#xx:3 of @ERd)]→C	—	—	—	—	—	↕	3
	BIOR #xx:3,@aa:8	B						4				C∨[¬(#xx:3 of @aa:8)]→C	—	—	—	—	—	↕	3
	BIOR #xx:3,@aa:16	B						6				C∨[¬(#xx:3 of @aa:16)]→C	—	—	—	—	—	↕	4
	BIOR #xx:3,@aa:32	B						8				C∨[¬(#xx:3 of @aa:32)]→C	—	—	—	—	—	↕	5
BXOR	BXOR #xx:3,Rd	B		2								C⊕(#xx:3 of Rd8)→C	—	—	—	—	—	↕	1
	BXOR #xx:3,@ERd	B			4							C⊕(#xx:3 of @ERd)→C	—	—	—	—	—	↕	3
	BXOR #xx:3,@aa:8	B						4				C⊕(#xx:3 of @aa:8)→C	—	—	—	—	—	↕	3
	BXOR #xx:3,@aa:16	B						6				C⊕(#xx:3 of @aa:16)→C	—	—	—	—	—	↕	4
	BXOR #xx:3,@aa:32	B						8				C⊕(#xx:3 of @aa:32)→C	—	—	—	—	—	↕	5
BIXOR	BIXOR #xx:3,Rd	B		2								C⊕[¬(#xx:3 of Rd8)]→C	—	—	—	—	—	↕	1
	BIXOR #xx:3,@ERd	B			4							C⊕[¬(#xx:3 of @ERd)]→C	—	—	—	—	—	↕	3
	BIXOR #xx:3,@aa:8	B						4				C⊕[¬(#xx:3 of @aa:8)]→C	—	—	—	—	—	↕	3
	BIXOR #xx:3,@aa:16	B						6				C⊕[¬(#xx:3 of @aa:16)]→C	—	—	—	—	—	↕	4
	BIXOR #xx:3,@aa:32	B						8				C⊕[¬(#xx:3 of @aa:32)]→C	—	—	—	—	—	↕	5

- Branch instructions

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation		Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		Branch Condition	I	H	N	Z	V	C	Advanced
Bcc	BRA d:8(BT d:8)	—							2			if condition is true then	Always	—	—	—	—	—	—	2
	BRA d:16(BT d:16)	—							4			PC←PC+d		—	—	—	—	—	—	3
	BRN d:8(BF d:8)	—							2			else next;	Never	—	—	—	—	—	—	2
	BRN d:16(BF d:16)	—							4					—	—	—	—	—	—	3
	BHI d:8	—							2				C∨Z=0	—	—	—	—	—	—	2
	BHI d:16	—							4					—	—	—	—	—	—	3
	BLS d:8	—							2				C∨Z=1	—	—	—	—	—	—	2
	BLS d:16	—							4					—	—	—	—	—	—	3
	BCC d:8(BHS d:8)	—							2				C=0	—	—	—	—	—	—	2
	BCC d:16(BHS d:16)	—							4					—	—	—	—	—	—	3
	BCS d:8(BLO d:8)	—							2				C=1	—	—	—	—	—	—	2
	BCS d:16(BLO d:16)	—							4					—	—	—	—	—	—	3
	BNE d:8	—							2				Z=0	—	—	—	—	—	—	2
	BNE d:16	—							4					—	—	—	—	—	—	3
	BEQ d:8	—							2				Z=1	—	—	—	—	—	—	2
	BEQ d:16	—							4					—	—	—	—	—	—	3
	BVC d:8	—							2				V=0	—	—	—	—	—	—	2
	BVC d:16	—							4					—	—	—	—	—	—	3
	BVS d:8	—							2				V=1	—	—	—	—	—	—	2
	BVS d:16	—							4					—	—	—	—	—	—	3
	BPL d:8	—							2				N=0	—	—	—	—	—	—	2
	BPL d:16	—							4					—	—	—	—	—	—	3
	BMI d:8	—							2				N=1	—	—	—	—	—	—	2
	BMI d:16	—							4					—	—	—	—	—	—	3
	BGE d:8	—							2				N⊕V=0	—	—	—	—	—	—	2
	BGE d:16	—							4					—	—	—	—	—	—	3
	BLT d:8	—							2				N⊕V=1	—	—	—	—	—	—	2
	BLT d:16	—							4					—	—	—	—	—	—	3
	BGT d:8	—							2				Z∨(N⊕V)=0	—	—	—	—	—	—	2
	BGT d:16	—							4					—	—	—	—	—	—	3
	BLE d:8	—							2				Z∨(N⊕V)=1	—	—	—	—	—	—	2
	BLE d:16	—							4					—	—	—	—	—	—	3
JMP	JMP @ERn	—			2							PC←ERn		—	—	—	—	—	—	2
	JMP @aa:24	—						4				PC←aa:24		—	—	—	—	—	—	3
	JMP @@aa:8	—								2		PC←@aa:8		—	—	—	—	—	—	5
BSR	BSR d:8	—							2			PC→@-SP,PC←PC+d:8		—	—	—	—	—	—	4
	BSR d:16	—							4			PC→@-SP,PC←PC+d:16		—	—	—	—	—	—	5
JSR	JSR @ERn	—			2							PC→@-SP,PC←ERn		—	—	—	—	—	—	4
	JSR @aa:24	—						4				PC→@-SP,PC←aa:24		—	—	—	—	—	—	5
	JSR @@aa:8	—								2		PC→@-SP,PC←@aa:8		—	—	—	—	—	—	6
RTS	RTS	—									2	PC←@SP+		—	—	—	—	—	—	5

- System control instructions

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Advanced
TRAPA	TRAPA #xx:2	—										PC→@-SP,CCR→@-SP, EXR→@-SP,<vector>→PC	1	—	—	—	—	—	8 [9]
RTE	RTE	—										EXR←@SP+,CCR←@SP+, PC←@SP+	↕	↕	↕	↕	↕	↕	5 [9]
SLEEP	SLEEP	—										Transition to the power-down state	—	—	—	—	—	—	2
LDC	LDC #xx:8,CCR	B	2									#xx:8→CCR	↕	↕	↕	↕	↕	↕	1
	LDC #xx:8,EXR	B	4									#xx:8→EXR	—	—	—	—	—	—	2
	LDC Rs,CCR	B		2								Rs8→CCR	↕	↕	↕	↕	↕	↕	1
	LDC Rs,EXR	B		2								Rs8→EXR	—	—	—	—	—	—	1
	LDC @ERs,CCR	W			4							@ERs→CCR	↕	↕	↕	↕	↕	↕	3
	LDC @ERs,EXR	W			4							@ERs→EXR	—	—	—	—	—	—	3
	LDC @(d:16,ERs),CCR	W				6						@(d:16,ERs)→CCR	↕	↕	↕	↕	↕	↕	4
	LDC @(d:16,ERs),EXR	W				6						@(d:16,ERs)→EXR	—	—	—	—	—	—	4
	LDC @(d:32,ERs),CCR	W				10						@(d:32,ERs)→CCR	↕	↕	↕	↕	↕	↕	6
	LDC @(d:32,ERs),EXR	W				10						@(d:32,ERs)→EXR	—	—	—	—	—	—	6
	LDC @ERs+,CCR	W					4					@ERs→CCR,ERs32+2→ERs32	↕	↕	↕	↕	↕	↕	4
	LDC @ERs+,EXR	W					4					@ERs→EXR,ERs32+2→ERs32	—	—	—	—	—	—	4
	LDC @aa:16,CCR	W						6				@aa:16→CCR	↕	↕	↕	↕	↕	↕	4
	LDC @aa:16,EXR	W						6				@aa:16→EXR	—	—	—	—	—	—	4
	LDC @aa:32,CCR	W						8				@aa:32→CCR	↕	↕	↕	↕	↕	↕	5
	LDC @aa:32,EXR	W						8				@aa:32→EXR	—	—	—	—	—	—	5
STC	STC CCR,Rd	B		2								CCR→Rd8	—	—	—	—	—	—	1
	STC EXR,Rd	B		2								EXR→Rd8	—	—	—	—	—	—	1
	STC CCR,@ERd	W			4							CCR→@ERd	—	—	—	—	—	—	3
	STC EXR,@ERd	W			4							EXR→@ERd	—	—	—	—	—	—	3
	STC CCR,@(d:16,ERd)	W				6						CCR→@(d:16,ERd)	—	—	—	—	—	—	4
	STC EXR,@(d:16,ERd)	W				6						EXR→@(d:16,ERd)	—	—	—	—	—	—	4
	STC CCR,@(d:32,ERd)	W				10						CCR→@(d:32,ERd)	—	—	—	—	—	—	6
	STC EXR,@(d:32,ERd)	W				10						EXR→@(d:32,ERd)	—	—	—	—	—	—	6
	STC CCR,@-ERd	W					4					ERd32-2→ERd32,CCR→@ERd	—	—	—	—	—	—	4
	STC EXR,@-ERd	W					4					ERd32-2→ERd32,EXR→@ERd	—	—	—	—	—	—	4
	STC CCR,@aa:16	W						6				CCR→@aa:16	—	—	—	—	—	—	4
	STC EXR,@aa:16	W						6				EXR→@aa:16	—	—	—	—	—	—	4
	STC CCR,@aa:32	W						8				CCR→@aa:32	—	—	—	—	—	—	5
	STC EXR,@aa:32	W						8				EXR→@aa:32	—	—	—	—	—	—	5
ANDC	ANDC #xx:8,CCR	B	2									CCR∧#xx:8→CCR	↕	↕	↕	↕	↕	↕	1
	ANDC #xx:8,EXR	B	4									EXR∧#xx:8→EXR	—	—	—	—	—	—	2
ORC	ORC #xx:8,CCR	B	2									CCR∨#xx:8→CCR	↕	↕	↕	↕	↕	↕	1
	ORC #xx:8,EXR	B	4									EXR∨#xx:8→EXR	—	—	—	—	—	—	2
XORC	XORC #xx:8,CCR	B	2									CCR⊕#xx:8→CCR	↕	↕	↕	↕	↕	↕	1
	XORC #xx:8,EXR	B	4									EXR⊕#xx:8→EXR	—	—	—	—	—	—	2
NOP	NOP	—									2	PC←PC+2	—	—	—	—	—	—	1

- Block transfer instructions

Mnemonic		Operand Size	Addressing Mode/Instruction Length (Bytes)									Operation	Condition Code						Number of States*1	
			#xx	Rn	@ERn	@(d,ERn)	@-ERn/@ERn+	@aa	@(d,PC)	@@aa	|		I	H	N	Z	V	C	Normal	Advanced
EEPMOV	EEPMOV.B	—									4	if R4L≠0 Repeat @ER5→@ER6 R5+1→R5 R6+1→R6 R4L-1→R4L Until R4L=0 else next;	—	—	—	—	—	—	4+2n*2	
	EEPMOV.W	—									4	if R4≠0 Repeat @ER5→@ER6 R5+1→R5 R6+1→R6 R4-1→R4 Until R4=0 else next;	—	—	—	—	—	—	4+2n*2	

Notes: *1: The number of states is the number of states required for execution when the instruction code and operands are located in on-chip memory.
*2: n is the initial value of R4L or R4.
[1] 7 states when the number of restored/saved registers is 2, 9 states when 3, and 11 states when 4.
[2] Cannot be used with the H8S/2633 Series.
[3] Set to 1 when there is a carry from or borrow to bit 11; otherwise cleared to 0.
[4] Set to 1 when there is a carry from or borrow to bit 27; otherwise cleared to 0.
[5] If the result is zero, the previous value of the flag is retained; otherwise the flag is cleared to 0.
[6] Set to 1 if the divisor is negative: otherwise cleared to 0.
[7] Set to 1 if the divisor is zero; otherwise cleared to 0.
[8] Set to 1 if the quotient is negative; otherwise cleared to 0.
[9] When EXR is valid, the number of states is increased by 1.

Number of States Required for Execution

The number of states shown in the instruction set table is the number of states required for execution when the operation code and operand data are located in a one-cycle area on which word access is possible, such as on-chip memory. When the operation code or operand data is accessed from an on-chip supporting module or an external address, the number of states increases as shown in the table below.

- Number of States Required for Execution (Cycle)

	Access Conditions						
				External Device			
		On-Chip Supporting Module		8-Bit Bus		16-Bit Bus	
Execution State (Cycle)	**On-Chip Memory**	**8-Bit Bus**	**16-Bit Bus**	**2-State Access**	**3-State Access**	**2-State Access**	**3-State Access**
Instruction fetch	1	4	2	4	6 + 2m	2	3 + m
Branch instruction read							
Stack operation							
Byte data access		2		2	3 + m		
Word data access		4		4	6 + 2m		
Internal operation	1	1	1	1	1	1	1

Legend
m: Number of wait states inserted in external device access

Condition Code Notation

Symbol	Meaning
↕	Changes according to operation result.
*	Indeterminate (value not guaranteed).
0	Always cleared to 0.
1	Always set to 1.
—	Not affected by operation result.

Rd	General register (destination)*
Rs	General register (source)*
Rn	General register*
ERn	General register (32-bit register)
MAC	Multiply-accumulate register (32-bit register)
(EAd)	Destination operand
(EAs)	Source operand
EXR	Extend register
CCR	Condition-code register
N	N (negative) flag in CCR
Z	Z (zero) flag in CCR
V	V (overflow) flag in CCR
C	C (carry) flag in CCR
PC	Program counter
SP	Stack pointer
#IMM	Immediate data
disp	Displacement
+	Addition
–	Subtraction
×	Multiplication
÷	Division
∧	Logical AND
∨	Logical OR
⊕	Logical exclusive OR
→	Transfer from left-hand operand to right-hand operand, or transition from left-hand state to right-hand state
¬	NOT (logical complement)
( ) < >	Operand contents
:8/:16/:24/:32	8-, 16-, 24-, or 32-bit length

Note: * General registers include 8-bit registers (R0H to R7H, R0L to R7L), 16-bit registers (R0 to R7, E0 to E7), and 32-bit registers (ER0 to ER7).

2.6 Basic Bus Timing

The CPU operates on the basis of the system clock (ø). One ø clock cycle is called a state. The bus cycle consists of one, two, or three states. Different access methods are used for on-chip memory, on-chip supporting modules, and the external address space.

System Clock

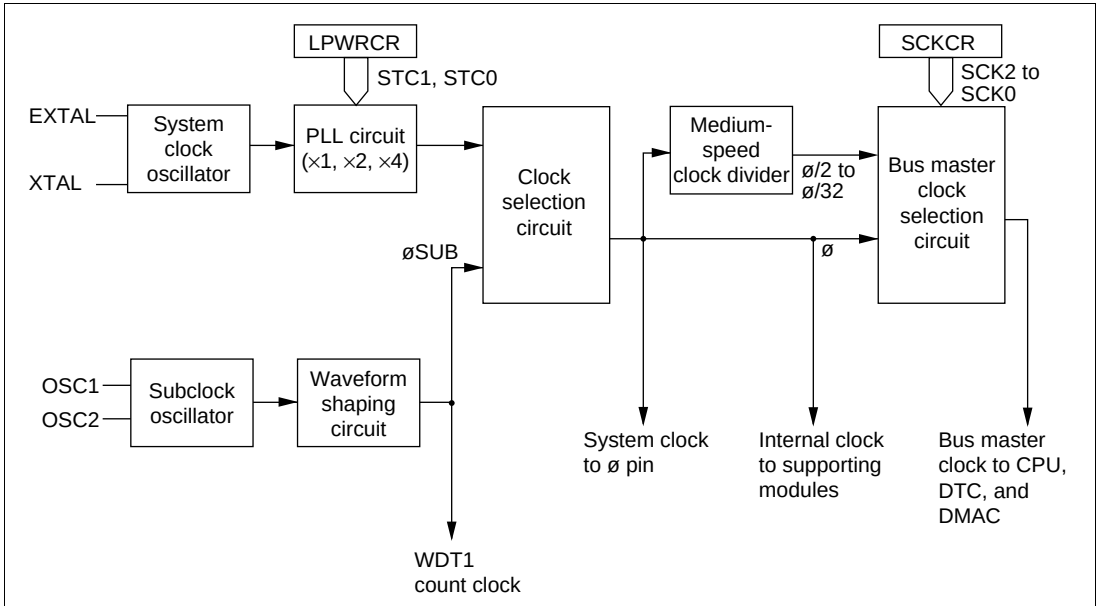
The system clock (ø) is generated by the oscillator, the PLL circuit, and the subclock oscillator.

The following two methods can be used to supply a clock to the oscillator. In either case, the input clock should not exceed 20 MHz.

1. Connecting a crystal resonator to the EXTAL and XTAL pins
2. Inputting an external clock to the EXTAL pin

The frequency of the clock from the oscillator can be multiplied by a factor of 1, 2, or 4 by means of the PLL circuit. Ensure that, after multiplication, the clock frequency does not exceed the maximum operating frequency of the chip.

To supply a clock to the subclock oscillator, connect a 32.768 kHz crystal resonator to the OSC1 and OSC2 pins.



Legend:
LPWRCR: Low-power control register
SCKCR: System clock control register

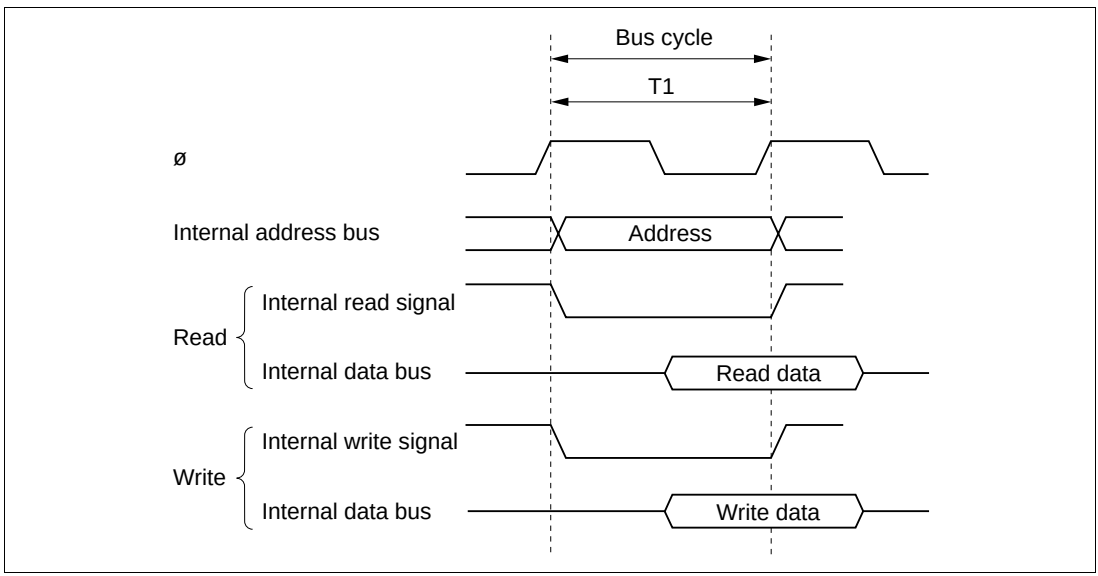
CPU Read/Write Cycles

The CPU operates on the basis of the system clock (ø). One ø clock cycle is called a state, and a bus cycle consists of one, two, or three states. Different access methods are used for on-chip memory, on-chip supporting modules, and external address space. Access to the external address space can be controlled by the bus controller.

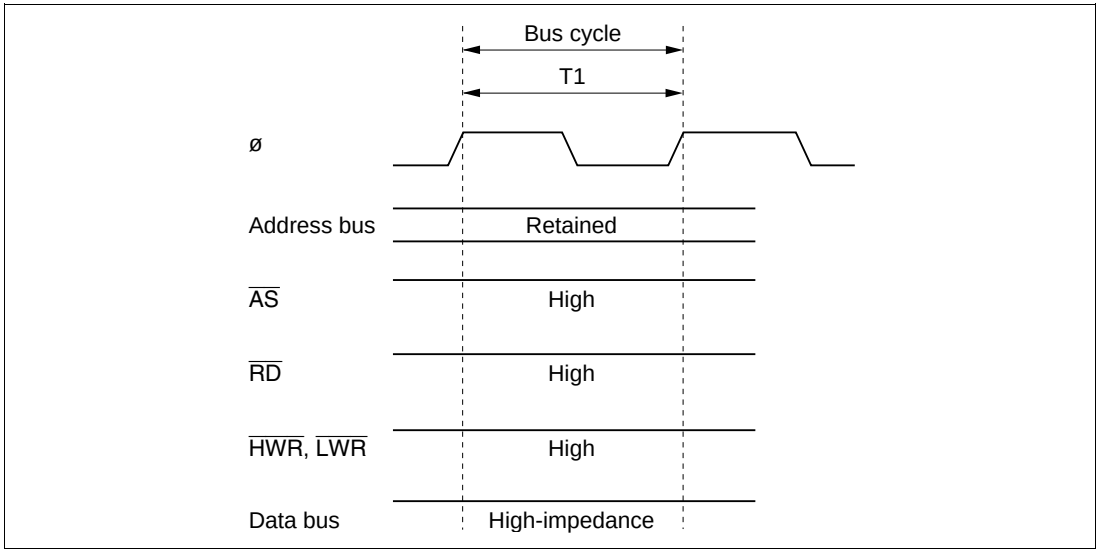
On-Chip Memory

On-chip memory is accessed in one state. The data bus is 16 bits wide, permitting both byte and word access.

- On-Chip Memory Access Cycle (One-State Access)



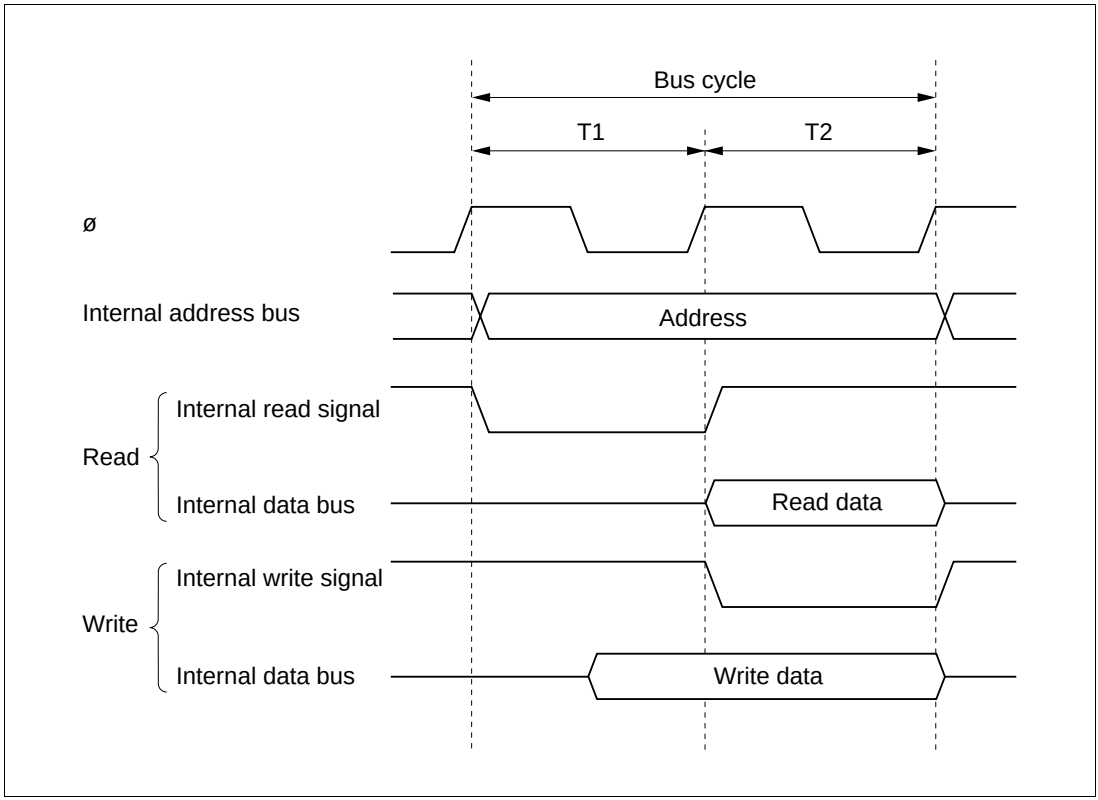
- Pin States during On-Chip Memory Access



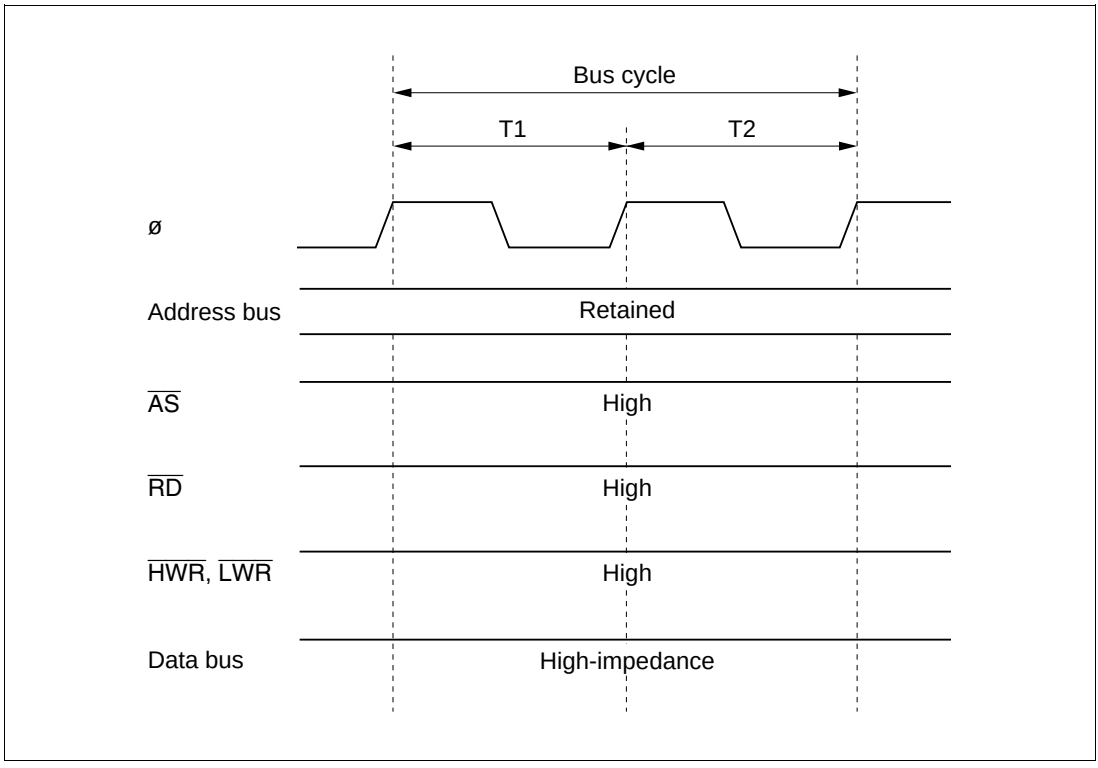
On-Chip Supporting Modules

The on-chip supporting modules are accessed in two states. The data bus is either 8 bits or 16 bits wide, depending on the internal I/O register.

- On-Chip Supporting Module Access Cycle (Two-State Access)



- Pin States during On-Chip Supporting Module Access



External Address Space

The external address space is accessed with an 8-bit or 16-bit data bus width in a two-state or three-state bus cycle. In three-state access, wait states can be inserted. For further details, refer to section 3.2, Bus Controller (BSC).

2.7 Processing States

The H8S/2600 CPU has five processing states: the reset state, program execution state, exception-handling state, bus-released state, and power-down state.

Reset State: State in which the CPU and all on-chip supporting modules are initialized and halted.

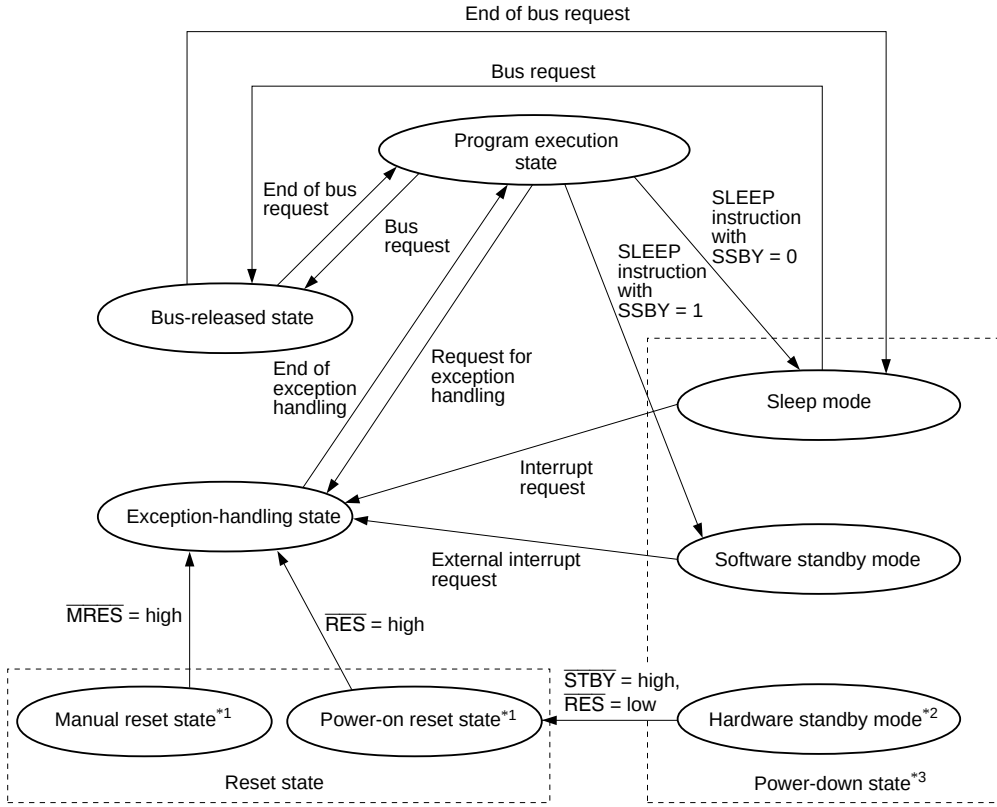
Program Execution State: State in which the CPU executes the program sequentially.

Exception-Handling State: Transient state in which the normal processing flow is altered and exception handling executed as the result of a reset, interrupt, or trap instruction exception handling source.

Bus-Released State: State in which the external bus is released in response to a bus request signal from a bus master other than the CPU.

Power-Down State: State in which CPU operation is stopped, and power consumption is kept low (sleep mode, software standby mode, hardware standby mode, subsleep mode, watch mode). The power-down state also includes medium-speed mode, module stop mode, and subactive mode.

State Transition Diagram



Notes: 1. From any state except hardware standby mode, a transition to the power-on reset state occurs whenever $\overline{RES}$ goes low. From any state except hardware standby mode or the power-on reset state, a transition to the manual reset state occurs whenever $\overline{MRES}$ goes low. A transition can also be made to the reset state when the watchdog timer overflows.

2. From any state, a transition to hardware standby mode occurs whenever $\overline{STBY}$ goes low.
3. A watch mode, subactive mode, and subsleep mode are also provided. For details, see section 4, Power-Down Modes.

2.8 Exception Handling

H8S/2600 CPU exception handling is activated by a reset, a trap instruction, or an interrupt. A priority system is provided for exception handling, and simultaneously generated exceptions are handled in order of priority.

Exception Types and Priority

Priority	Exception Type	Start of Exception Handling
High	Reset	Starts immediately after a low-to-high transition at the $\overline{\text{RES}}$ pin, or when the watchdog timer overflows
	Trace	Starts when execution of the current instruction or exception handling ends, if the trace (T) bit is set to 1
	Direct transition	Starts when a direct transition occurs as the result of SLEEP instruction execution
	Interrupt	Starts when execution of the current instruction or exception handling ends, if an interrupt request has been issued
Low	Trap instruction (TRAPA)	Started by execution of a trap instruction (TRAPA)

Exception Handling Operation

Exception handling is initiated by various exception handling sources. Trap instruction exception handling is always accepted in the program execution state. Trap instructions and interrupts are handled as follows:

1. The program counter (PC), condition code register (CCR), and extend register (EXR) are pushed onto the stack.
2. The interrupt mask bits are updated. The T bit is cleared to 0.
3. A vector address corresponding to the activation source is generated, and program execution starts from the address indicated in that vector address.

For a reset exception, steps 2 and 3 above are carried out.

Exception Vector Table

Exception Source		Vector Number	Vector Address*[1]
Power-on reset		0	H'0000 to H'0003
Manual reset		1	H'0004 to H'0007
Reserved for system use		2	H'0008 to H'000B
		3	H'000C to H'000F
		4	H'0010 to H'0013
Trace		5	H'0014 to H'0017
Direct transition		6	H'0018 to H'001B
External interrupt	NMI	7	H'001C to H'001F
Trap instruction (4 sources)		8	H'0020 to H'0023
		9	H'0024 to H'0027
		10	H'0028 to H'002B
		11	H'002C to H'002F
Reserved for system use		12	H'0030 to H'0033
		13	H'0034 to H'0037
		14	H'0038 to H'003B
		15	H'003C to H'003F
External interrupt	IRQ0	16	H'0040 to H'0043
	IRQ1	17	H'0044 to H'0047
	IRQ2	18	H'0048 to H'004B
	IRQ3	19	H'004C to H'004F
	IRQ4	20	H'0050 to H'0053
	IRQ5	21	H'0054 to H'0057
	IRQ6	22	H'0058 to H'005B
	IRQ7	23	H'005C to H'005F
Internal interrupt*[2]		24 to 127	H'0060 to H'0063 to H'01FC to H'01FF

Notes: 1. Lower 16 bits of the address.

2. For details of internal interrupt vectors, see section 2.9, Interrupts.

2.9 Interrupts

Interrupts are controlled by the interrupt controller. There are a total of 81 interrupt sources, comprising nine external interrupts from the external pins (NMI, $\overline{\text{IRQ0}}$ to $\overline{\text{IRQ7}}$), and 72 internal interrupts from on-chip supporting modules (including options). A separate vector number is assigned to each interrupt.

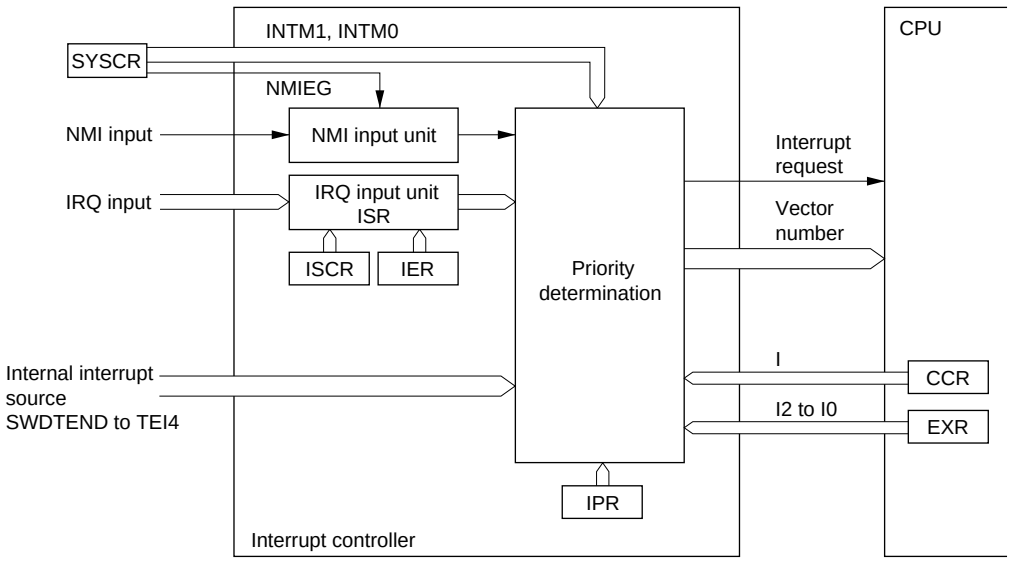
Interrupt Control

Either of two interrupt control modes can be set by means of the INTM1 and INTM0 bits in the system control register (SYSCR).

The interrupt controller controls interrupts on the basis of the control mode set by the INTM1 and INTM0 bits, the interrupt priorities set by interrupt priority register (IPR), and the masking conditions set by the I bit in CCR and bits I2 to I0 in EXR.

NMI is the highest-priority interrupt, and is always accepted.

Block Diagram of Interrupt Controller

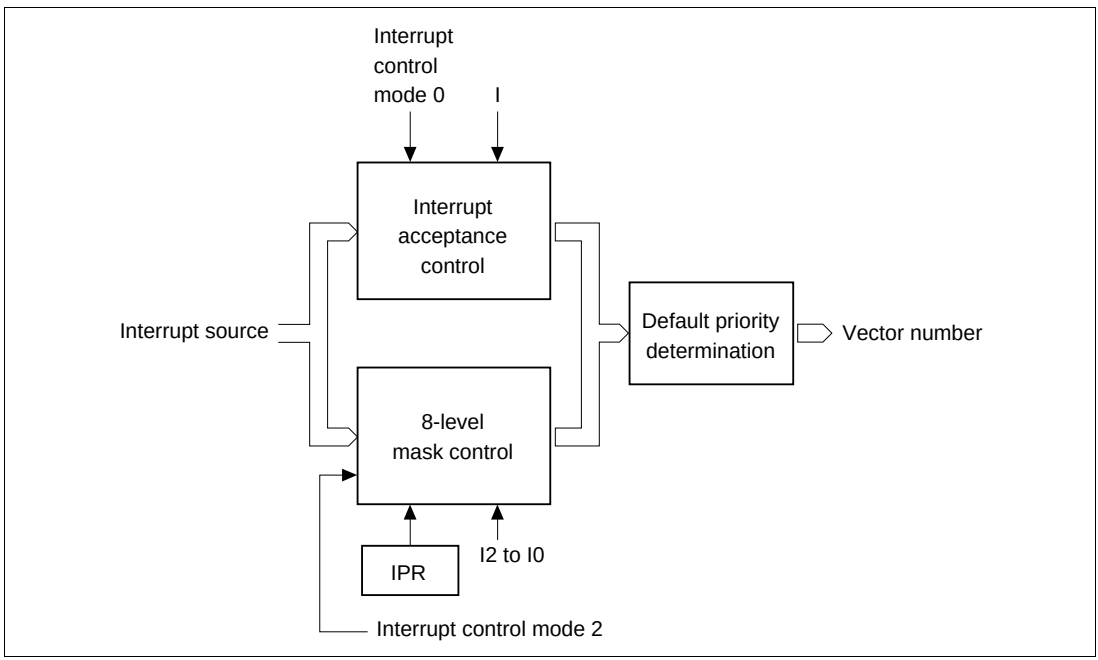


Legend:
ISCR: IRQ sense control register
IER: IRQ enable register
ISR: IRQ status register
IPR: Interrupt priority register
SYSCR: System control register

Interrupt Control Modes

Interrupt Control Mode	SYSCR INTM1	SYSCR INTM0	Priority Setting Registers	Interrupt Mask Bits	Description
0	0	0	—	I	Interrupt mask control is performed by the I bit.
—		1	—	—	Setting prohibited
2	1	0	IPR	I2 to I0	8-level interrupt mask control is performed by bits I2 to I0. 8 priority levels can be set with IPR.
—		1	—	—	Setting prohibited

- Block Diagram of Interrupt Control Operation



Interrupt Control Mode 0: Enabling and disabling of IRQ interrupts and on-chip supporting module interrupts can be set by means of the I bit in CCR. Interrupts are enabled when the I bit is cleared to 0, and disabled when set to 1.

Interrupt Control Mode 2: Eight-level masking can be implemented for IRQ interrupts and on-chip supporting module interrupts by comparing the interrupt mask level bits (I2 to I0) in EXR and the IPR priority level.

Interrupt Sources, Vector Addresses, and Interrupt Priorities

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address*	IPR	Priority
NMI	External pin	7	H'001C		High
IRQ0		16	H'0040	IPRA6 to 4	
IRQ1		17	H'0044	IPRA2 to 0	
IRQ2 IRQ3		18 19	H'0048 H'004C	IPRB6 to 4	
IRQ4 IRQ5		20 21	H'0050 H'0054	IPRB2 to 0	
IRQ6 IRQ7		22 23	H'0058 H'005C	IPRC6 to 4	
SWDTEND (software activated data transfer end)	DTC	24	H'0060	IPRC2 to 0	
WOVI0 (interval timer 0)	Watchdog timer 0	25	H'0064	IPRD6 to 4	
CMI	Refresh controller	26	H'0068	IPRD2 to 0	
PC break	PC break	27	H'006C	IPRE6 to 4	
ADI (A/D conversion end)	A/D	28	H'0070	IPRE2 to 0	
WOVI1 (interval timer 1)	Watchdog timer 1	29	H'0074		
TGI0A (TGR0A input capture/ compare match)	TPU channel 0	32	H'0080	IPRF6 to 4	
TGI0B (TGR0B input capture/ compare match)		33	H'0084		
TGI0C (TGR0C input capture/ compare match)		34	H'0088		
TGI0D (TGR0D input capture/ compare match)		35	H'008C		
TCI0V (overflow 0)		36	H'0090		
TGI1A (TGR1A input capture/ compare match)	TPU channel 1	40	H'00A0	IPRF2 to 0	
TGI1B (TGR1B input capture/ compare match)		41	H'00A4		
TCI1V (overflow 1)		42	H'00A8		
TCI1U (underflow 1)		43	H'00AC		Low

Note: * Lower 16 bits of the start address.

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address*	IPR	Priority
TGI2A (TGR2A input capture/ compare match)	TPU channel 2	44	H'00B0	IPRG6 to 4	High
TGI2B (TGR2B input capture/ compare match)		45	H'00B4		
TCI2V (overflow 2)		46	H'00B8		
TCI2U (underflow 2)		47	H'00BC		
TGI3A (TGR3A input capture/ compare match)	TPU channel 3	48	H'00C0	IPRG2 to 0	
TGI3B (TGR3B input capture/ compare match)		49	H'00C4		
TGI3C (TGR3C input capture/ compare match)		50	H'00C8		
TGI3D (TGR3D input capture/ compare match)		51	H'00CC		
TCI3V (overflow 3)		52	H'00D0		
TGI4A (TGR4A input capture/ compare match)	TPU channel 4	56	H'00E0	IPRH6 to 4	
TGI4B (TGR4B input capture/ compare match)		57	H'00E4		
TCI4V (overflow 4)		58	H'00E8		
TCI4U (underflow 4)		59	H'00EC		
TGI5A (TGR5A input capture/ compare match)	TPU channel 5	60	H'00F0	IPRH2 to 0	
TGI5B (TGR5B input capture/ compare match)		61	H'00F4		
TCI5V (overflow 5)		62	H'00F8		
TCI5U (underflow 5)		63	H'00FC		
CMIA0 (compare match A)	8-bit timer channel 0A	64	H'0100	IPRI6 to 4	
CMIB0 (compare match B)		65	H'0104		
OVI0 (overflow)		66	H'0108		
CMIA1 (compare match A)	8-bit timer channel 1A	68	H'0110	IPRI2 to 0	
CMIB1 (compare match B)		69	H'0114		
OVI1 (overflow)		70	H'0118		Low

Note: * Lower 16 bits of the start address.

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address*	IPR	Priority
DEND0A (channel 0/channel 0A transfer end)	DMAC	72	H'0120	IPRJ6 to 4	High
DEND0B (channel 0B transfer end)		73	H'0124		
DEND1A (channel 1/channel 1A transfer end)		74	H'0128		
DEND1B (channel 1B transfer end)		75	H'012C		
ERI0 (receive error 0)	SCI channel 0	80	H'0140	IPRJ2 to 0	
RXI0 (reception completed 0)		81	H'0144		
TXI0 (transmit data empty 0)		82	H'0148		
TEI0 (transmission end 0)		83	H'014C		
ERI1 (receive error 1)	SCI channel 1	84	H'0150	IPRK6 to 4	
RXI1 (reception completed 1)		85	H'0154		
TXI1 (transmit data empty 1)		86	H'0158		
TEI1 (transmission end 1)		87	H'015C		
ERI2 (receive error 2)	SCI channel 2	88	H'0160	IPRK2 to 0	
RXI2 (reception completed 2)		89	H'0164		
TXI2 (transmit data empty 2)		90	H'0168		
TEI2 (transmission end 2)		91	H'016C		
CMIA2 (compare match A)	8-bit timer channel 2A	92	H'0170	IPRL6 to 0	
CMIB2 (compare match B)		93	H'0174		
OVI2 (overflow)		94	H'0178		
CMIA3 (compare match A)	8-bit timer channel 3A	96	H'0180		
CMIB3 (compare match B)		97	H'0184		
OVI3 (overflow)		98	H'0188		
IICI0 (1-byte transmission/ reception completed)	IIC channel 0 [option]	100	H'0190	IPRL2 to 0	
DDCSWI (format switch)		101	H'0194		
IICI1 (1-byte transmission/ reception completed)	IIC channel 1 [option]	102	H'0198		
Reserved		103	H'019C		Low

Note: * Lower 16 bits of the start address.

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address*	IPR	Priority
ERI3 (receive error 3)	SCI channel 3	120	H'01E0	IPRO6 to 0	High
RXI3 (reception completed 3)		121	H'01E4		
TXI3 (transmit data empty 3)		122	H'01E8		
TEI3 (transmission end 3)		123	H'01EC		
ERI4 (receive error 4)	SCI channel 4	124	H'01F0	IPRO2 to 0	
RXI4 (reception completed 4)		125	H'01F4		
TXI4 (transmit data empty 4)		126	H'01F8		
TEI4 (transmission end 4)		127	H'01FC		Low

Note: * Lower 16 bits of the start address.

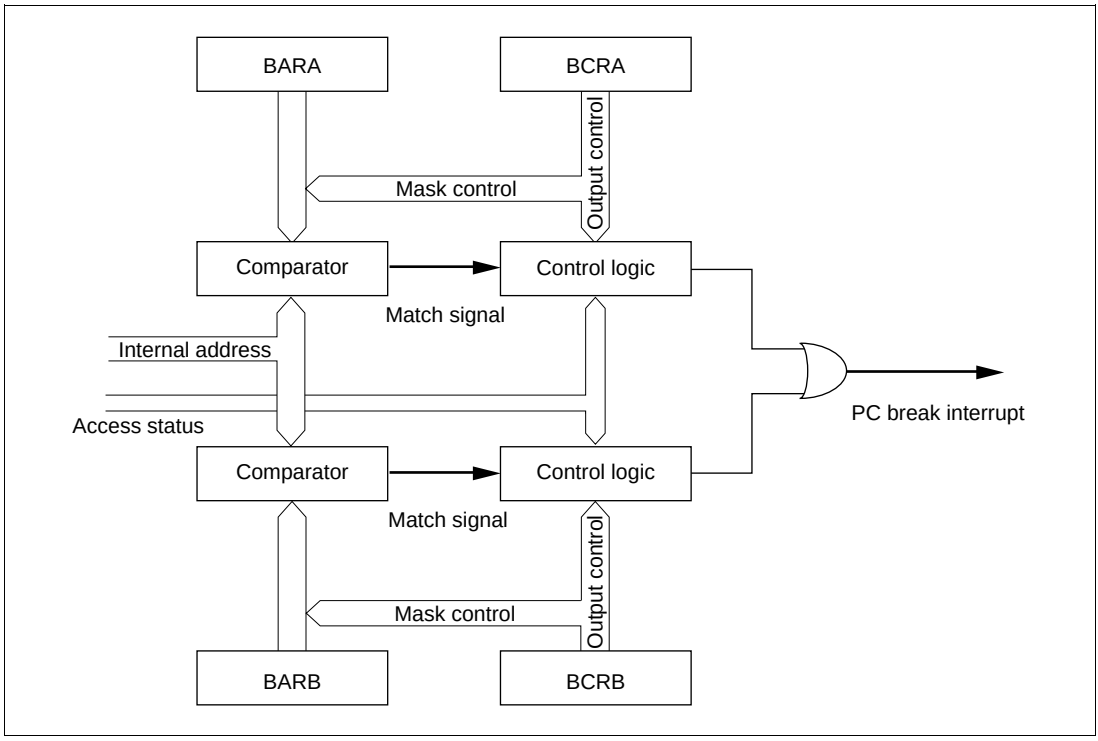
2.10 PC Break Controller (PBC)

This series has a two-channel on-chip PC break controller (PBC) providing functions that simplify program debugging. Using these functions, it is easy to create a sophisticated self-monitoring debugger, enabling programs to be debugged with the chip alone, without using a large-scale in-circuit emulator.

Features

- Two break channels (A and B)
- The following can be set as break compare conditions:
  - 24 address bits
    Bit masking possible
  - Bus cycle
    Instruction fetch
    Data access: data read, data write, data read/write
  - Bus master
    Either CPU or CPU/DTC can be selected
- The timing of PC break exception handling after the occurrence of a break condition is as follows:
  - Immediately before execution of the instruction fetched at the set address (instruction fetch)
  - Immediately after execution of the instruction that accesses data at the set address (data access)
- Module stop mode can be set
  - As the initial setting, PBC operation is halted. Register access is enabled by exiting module stop mode.

Block Diagram of PBC



2.11 Operating Modes

In this series, there are four operating modes. The operating mode is determined by the settings of the mode pins (MD2 to MD0).

Mode 4: The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is disabled. Pins P13 to P10 and ports A, B and C function as an address bus, ports D and E function as a data bus, and part of port F carries bus control signals.

The initial bus mode after a reset is 16 bits, with 16-bit access to all areas. However, if 8-bit access is designated by the bus controller for all areas, the bus mode switches to 8 bits.

Mode 5: The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is disabled. Pins P13 to P10 and ports A, B and C function as an address bus, port D functions as a data bus, and part of port F carries bus control signals.

The initial bus mode after a reset is 8 bits, with 8-bit access to all areas. However, if any area is designated for 16-bit access by the bus controller, the bus mode switches to 16 bits and port E becomes a data bus.

Mode 6: The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled. Pins P13 to P10 and ports A, B and C function as input ports immediately after a reset. These pins can be set to output addresses by setting the corresponding bits to 1 in the pin function control register (PFCR) in the case of pins P13 to P10 and ports A and B, or in the data direction register (DDR) for port C. Port D functions as a data bus, and part of port F carries bus control signals.

The initial bus mode after a reset is 8 bits, with 8-bit access to all areas. However, if any area is designated for 16-bit access by the bus controller, the bus mode switches to 16 bits and port E becomes a data bus.

Mode 7: The CPU can access a 16-Mbyte address space in advanced mode. The on-chip ROM is enabled, but external addresses cannot be accessed.

All I/O ports are available for use as input-output ports.

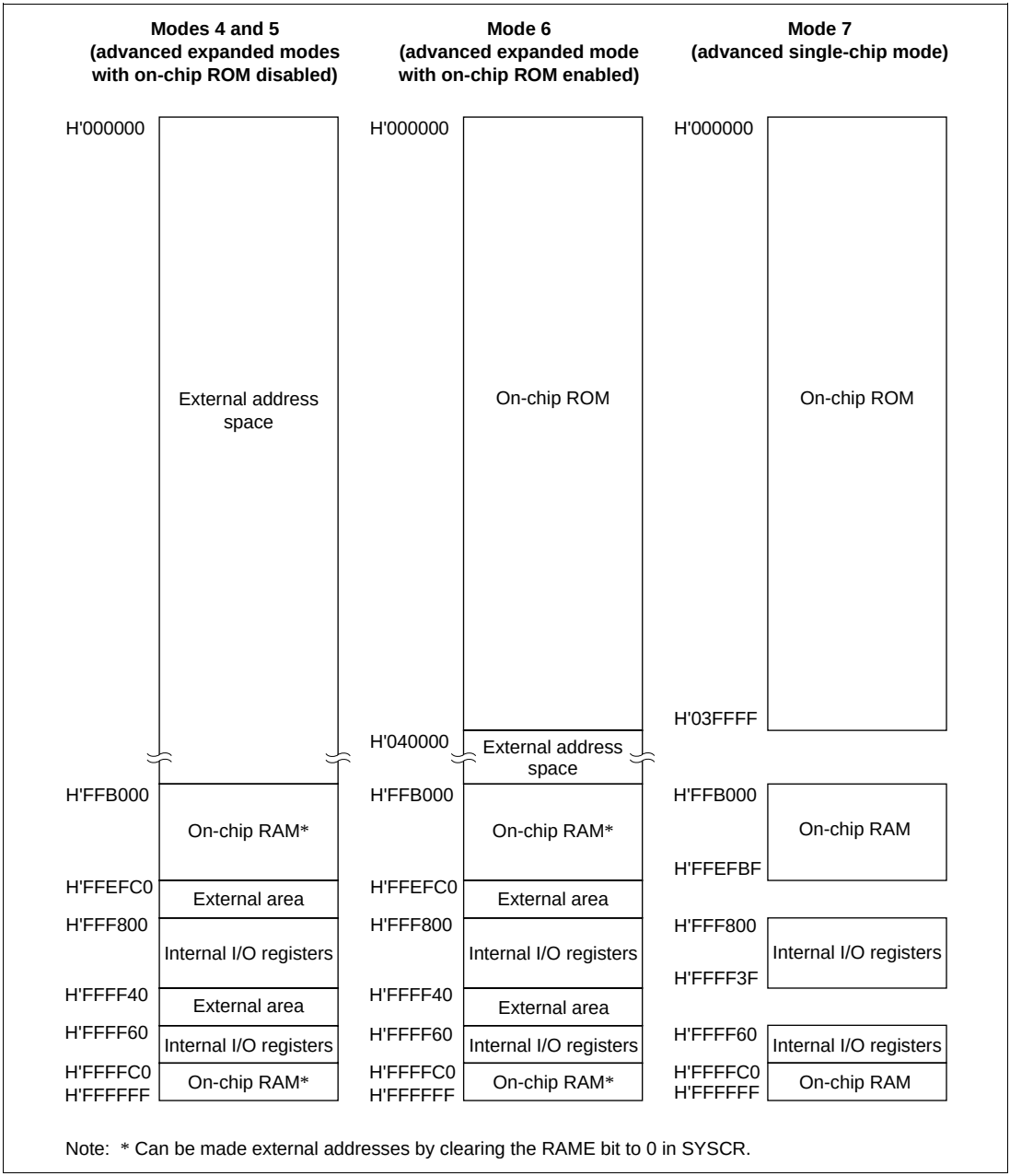
MCU Operating Modes

MCU Operating Mode	MD2	MD1	MD0	CPU Operating Mode	Description	On-Chip ROM	External Data Bus	
							Initial Width	Max. Width
4	1	0	0	Advanced	On-chip ROM disabled, expanded mode	Disabled	16 bits	16 bits
5			1				8 bits	16 bits
6		1	0		On-chip ROM enabled, expanded mode	Enabled	8 bits	16 bits
7			1		Single-chip mode		—	—

2.12 Address Maps

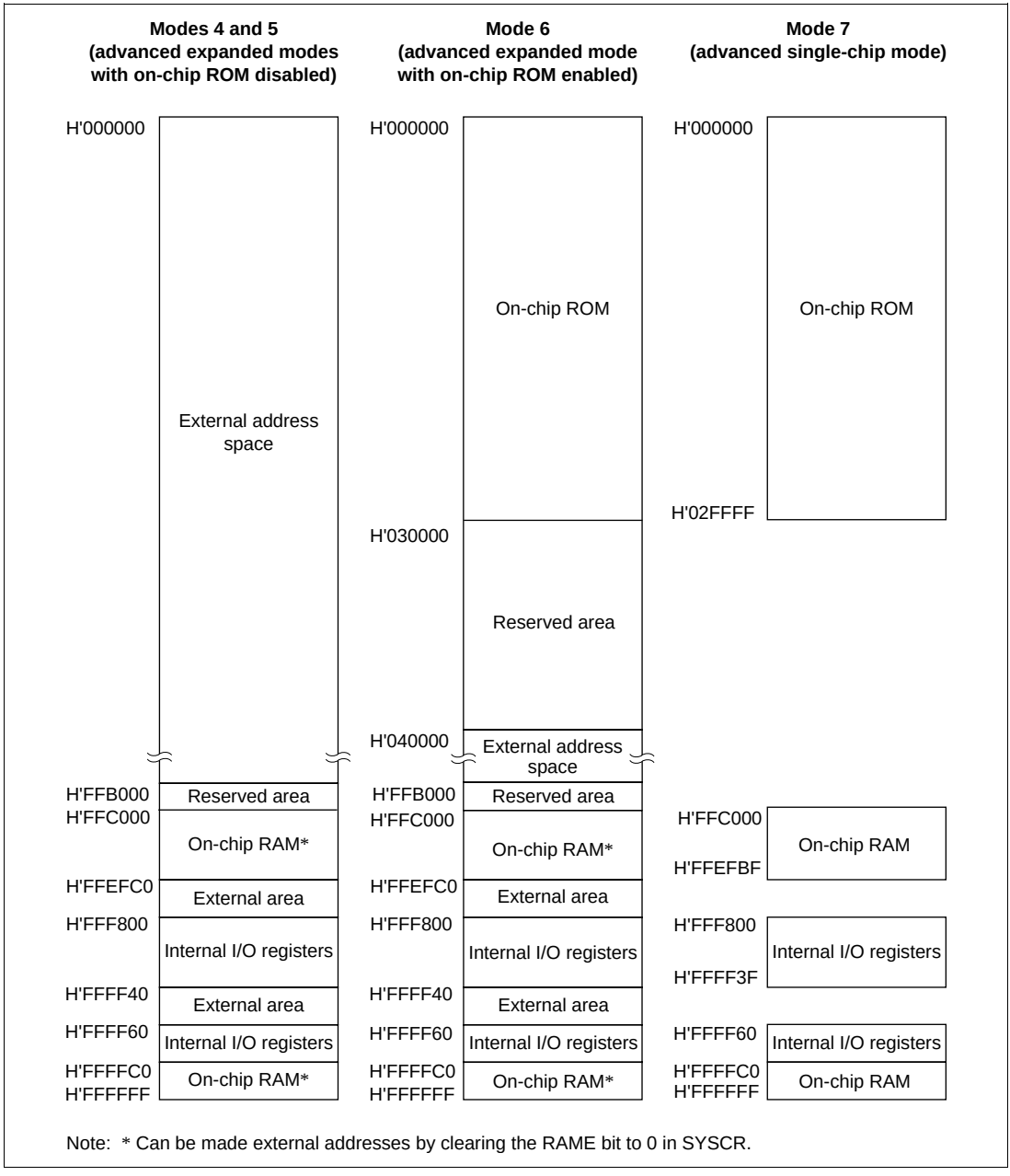
The address space is 16 Mbytes in modes 4 to 7 (advanced modes).

H8S/2633 Address Maps in Each Operating Mode

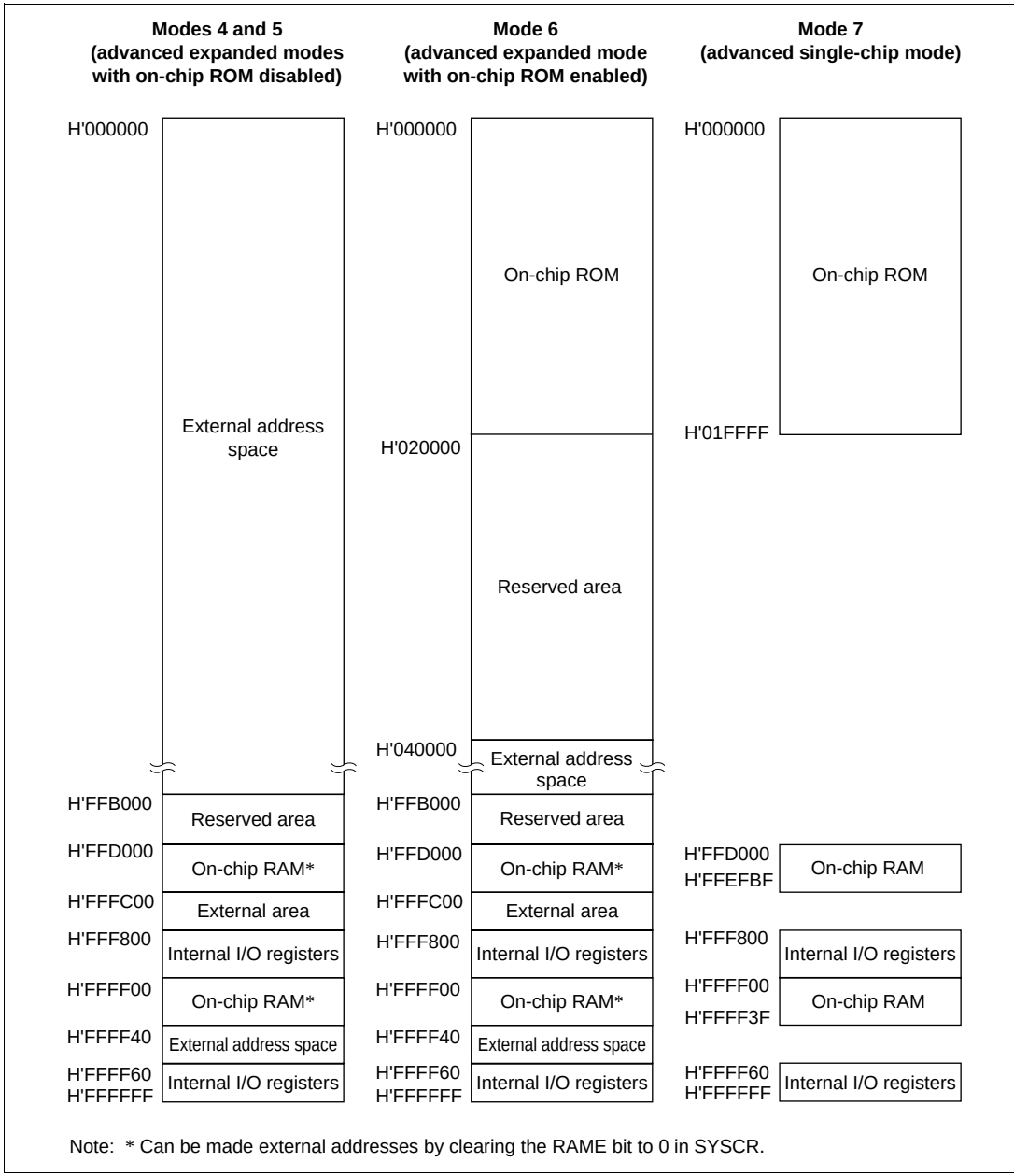


Note: * Can be made external addresses by clearing the RAME bit to 0 in SYSCR.

H8S/2632 Address Maps in Each Operating Mode



Note: * Can be made external addresses by clearing the RAME bit to 0 in SYSCR.



Note: * Can be made external addresses by clearing the RAME bit to 0 in SYSCR.

Section 3 Supporting Modules

3.1 Bus Controller (BSC)

The H8S/2633 Series has a built-in bus controller (BSC) that manages the external address space divided into eight areas. The bus specifications, such as bus width and number of access states, can be set independently for each area, enabling multiple memories to be connected easily.

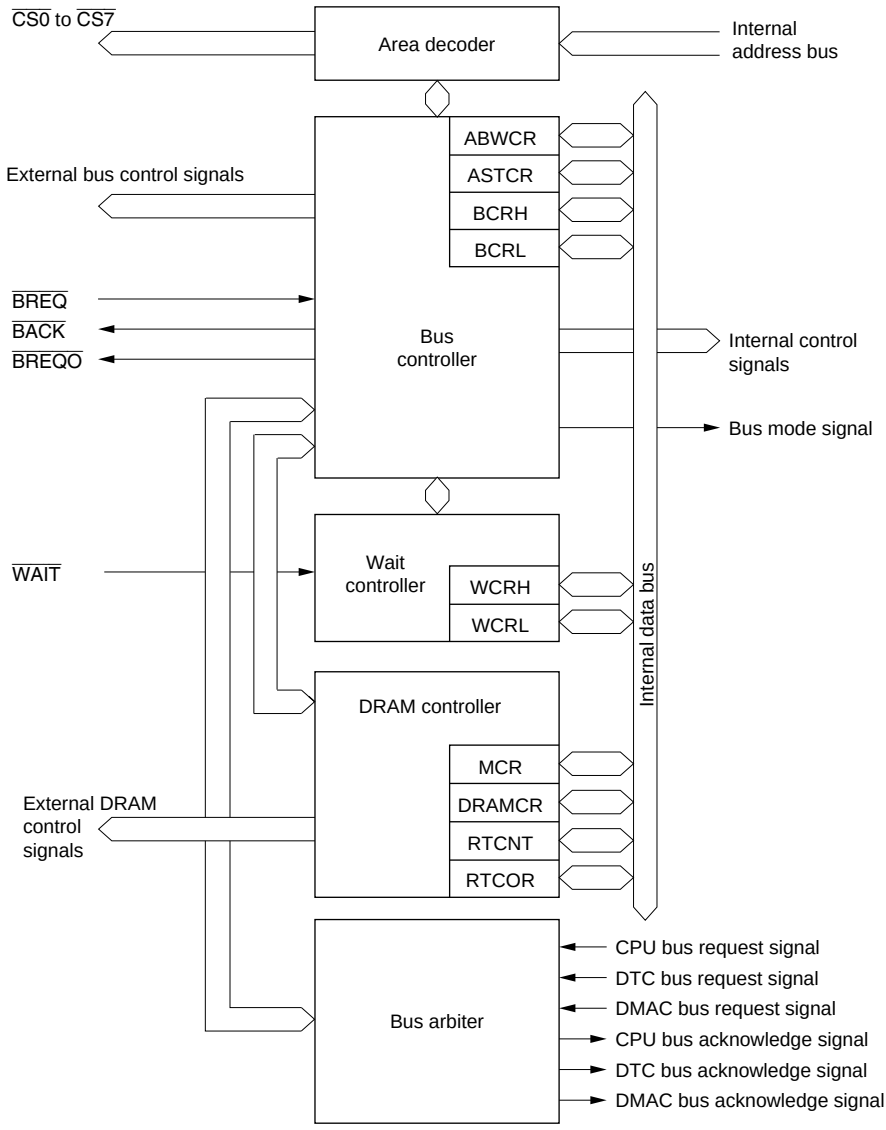
The bus controller also has a bus arbitration function, and controls the operation of the internal bus masters: the CPU, DMA controller (DMAC), and data transfer controller (DTC).

Features

- Manages external address space in area units
  - — Manages the external space as 8 areas of 2-Mbytes
  - — Bus specifications can be set independently for each area
  - — DRAM/burst ROM interfaces can be set
- Basic bus interface
  - — Chip select ($\overline{CS0}$ to $\overline{CS7}$) can be output for areas 0 to 7
  - — 8-bit access or 16-bit access can be selected for each area
  - — 2-state access or 3-state access can be selected for each area
  - — Program wait states can be inserted for each area
- DRAM interface
  - — DRAM interface can be set for areas 2 to 5 (in advanced mode)
  - — Row address/column address multiplexed output (8/9/10 bits)
  - — 2-CAS method
  - — Burst operation (fast page mode)
  - — T_P cycle insertion to secure RAS precharging time
  - — Choice of CAS-before-RAS refreshing or self-refreshing
- Burst ROM interface
  - — Burst ROM interface can be set for area 0
  - — Choice of 1- or 2-state burst access
- Idle cycle insertion
  - — An idle cycle can be inserted in case of an external read cycle between different areas
  - — An idle cycle can be inserted in case of an external write cycle immediately after an external read cycle
- Write buffer functions
  - — External write cycle and internal access can be executed in parallel
  - — DMAC single-address mode and internal access can be executed in parallel

- Bus arbitration function
  - — Includes a bus arbiter that arbitrates bus mastership among the CPU, DMAC, and DTC
- Other features
  - — Refresh counter (refresh timer) can be used as an interval timer
  - — External bus release function

Block Diagram of Bus Controller



Legend:

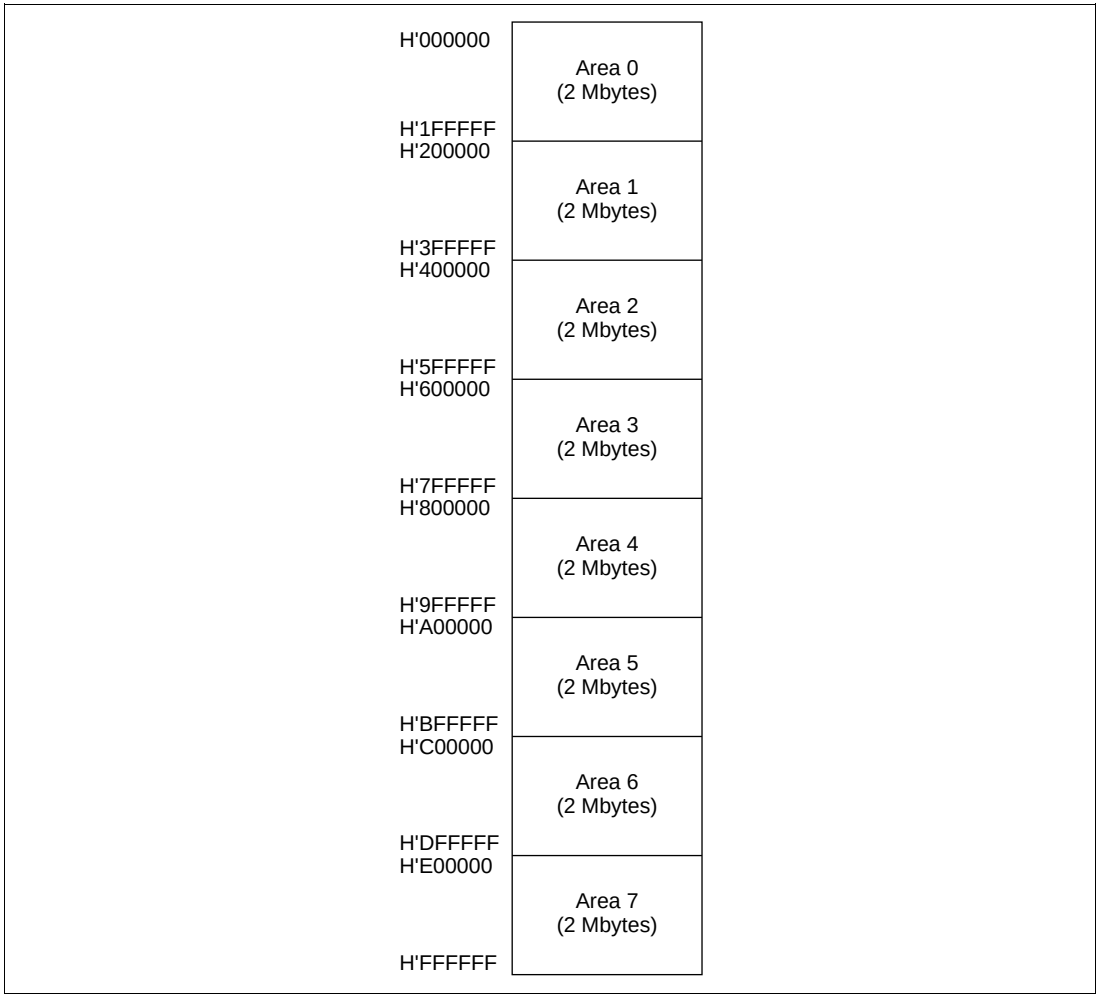
ABWCR:	Bus width control register	WCRL:	Wait control register L
ASTCR:	Access state control register	MCR:	Memory control register
BCRH:	Bus control register H	DRAMCR:	DRAM control register
BCRL:	Bus control register L	RTCNT:	Refresh timer counter
WCRH:	Wait control register H	RTCOR:	Refresh time constant register

3.1.1 Area Partitioning

The bus controller partitions the 16-Mbyte address space into eight areas, 0 to 7, in 2-Mbyte units, and performs bus control for external space in area units.

Area partitioning is only effective in expanded mode, and has no significance in single-chip mode.

Overview of Area Partitioning



Bus Specifications

The external address space bus specifications consist of three elements: bus width, number of access states, and number of program wait states. The bus width and number of access states for on-chip memory and internal I/O registers are fixed , and are not affected by the bus controller.

Bus specifications can be set as shown below by means of the bus controller control registers.

- Bus Specifications for Each Area (Basic Bus Interface)

ABWCR	ASTCR	WCRH, WCRL		Bus Specifications (Basic Bus Interface)		
ABWn	ASTn	Wn1	Wn0	Bus Width	Access States	Program Wait States
0	0	—	—	16	2	0
	1	0	0		3	0
			1			1
		1	0			2
			1			3
1	0	—	—	8	2	0
	1	0	0		3	0
			1			1
		1	0			2
			1			3

Memory Interfaces

This series' memory interfaces comprise **(1) a basic bus interface** that allows direct connection of ROM, SRAM, and so on; **(2) a DRAM interface** that allows direct connection of DRAM; and **(3) a burst ROM interface** that allows direct connection of burst ROM. The interface can be designated independently for each area.

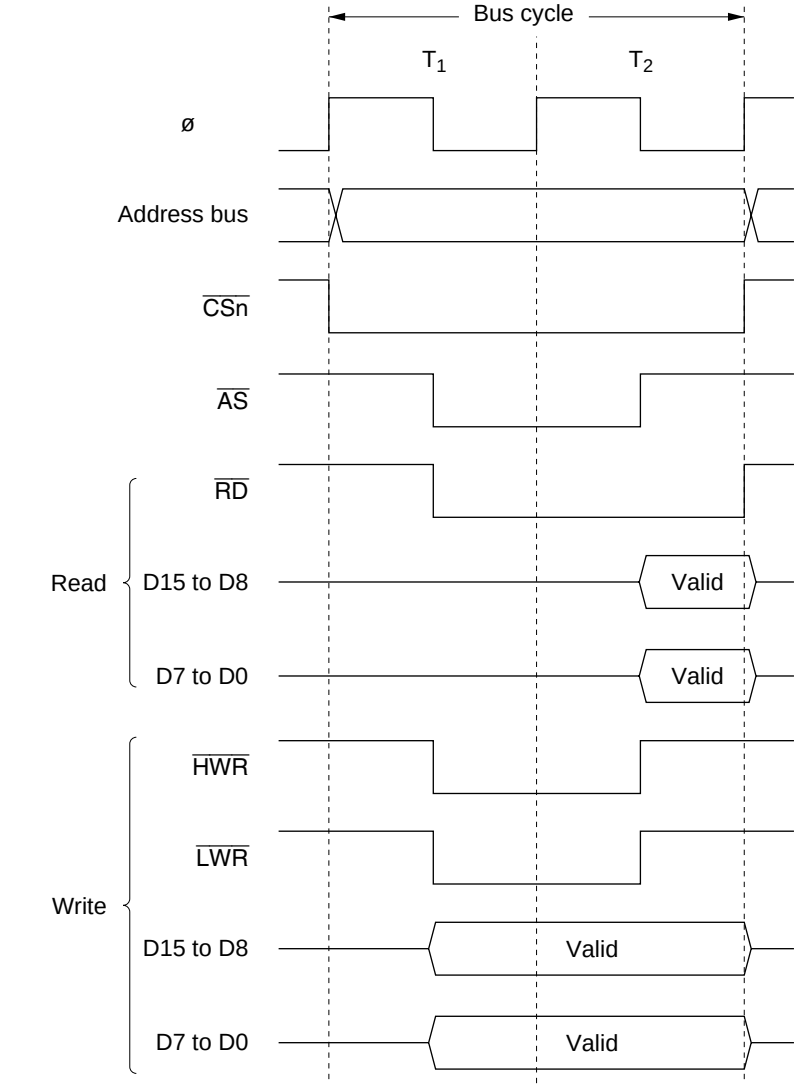
3.1.2 Basic Bus Interface

This interface can be designated for areas 0 to 7. When external space is accessed, the chip select signal ($\overline{CS0}$ to $\overline{CS7}$) for the relevant area (0 to 7) can be output.

In 3-state access space, 0 to 3 program wait states or a pin wait by means of the $\overline{WAIT}$ pin can be inserted.

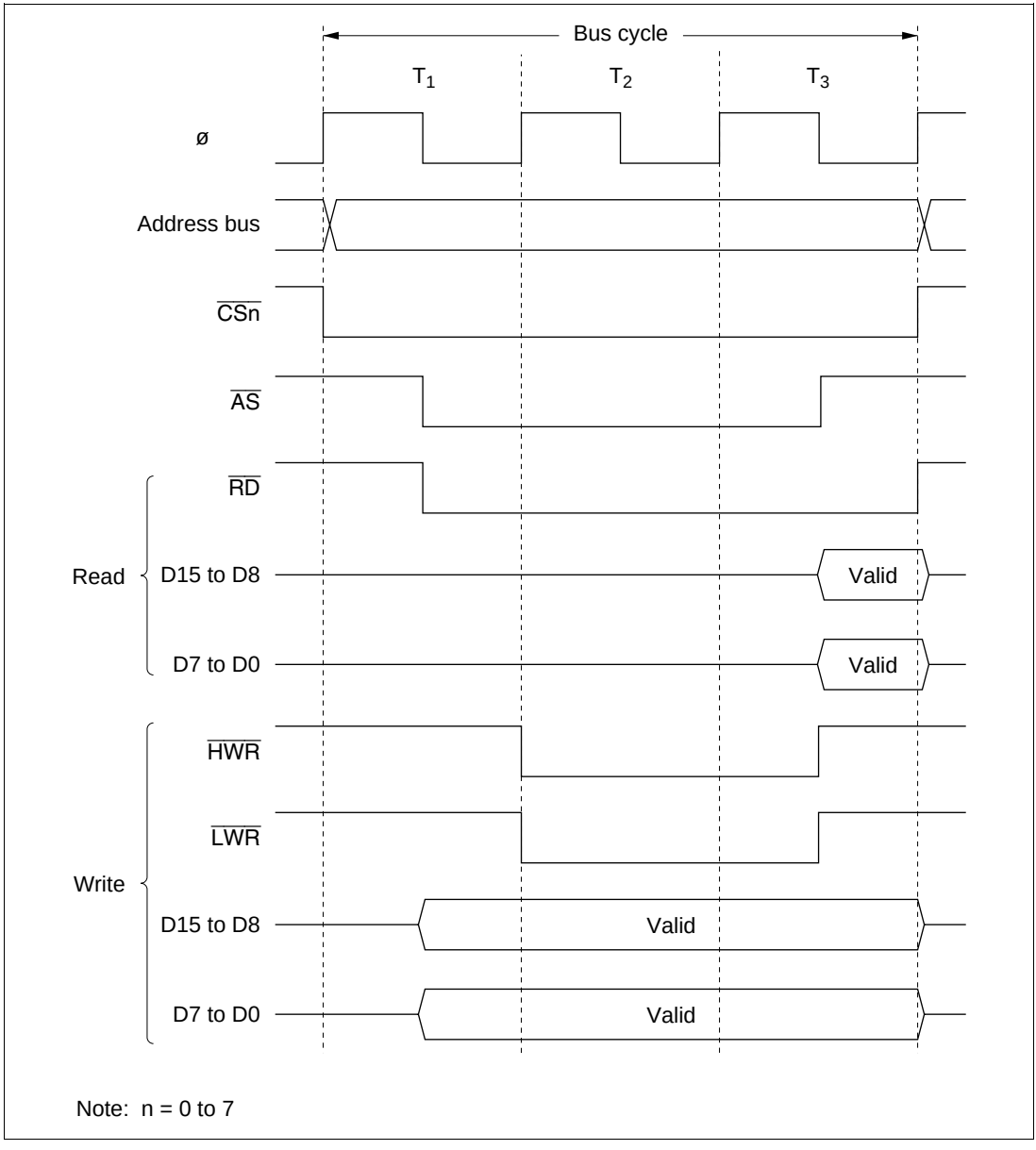
After a reset, all areas are designated as basic bus interface, 3-state access space (the bus width is determined by the MCU operating mode).

Basic Bus Timing (Word Access to 16-Bit 2-State Access Space)



Note: n = 0 to 7

Basic Bus Timing (Word Access to 16-Bit 3-State Access Space)



Note: n = 0 to 7

3.1.3 DRAM Interface

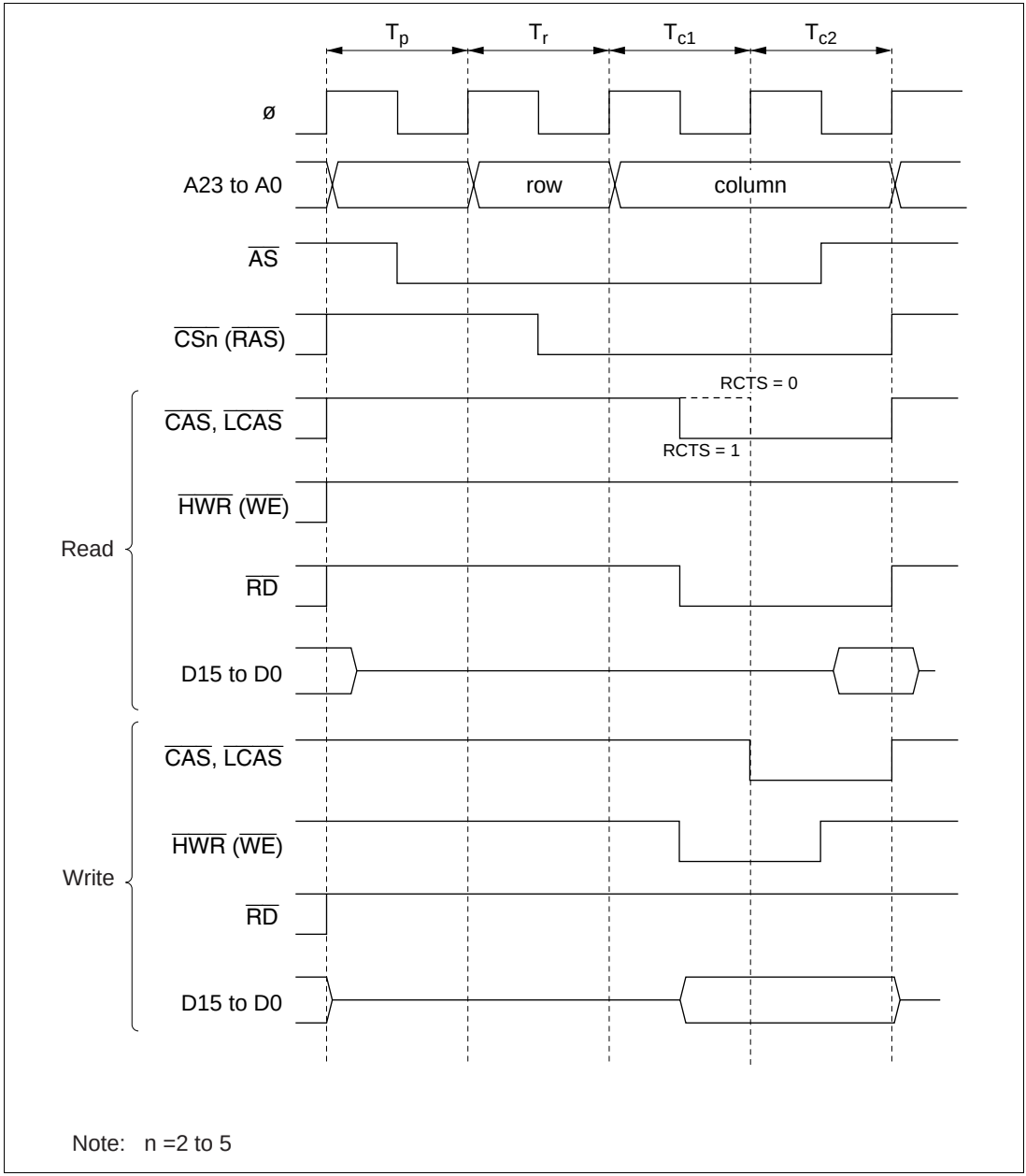
External space areas 2 to 5 can be designated as DRAM space, and DRAM interfacing performed. With the DRAM interface, DRAM can be directly connected to the H8S/2633 Series. Selectable DRAM space settings are: one area (area 2); two areas (areas 2 and 3); and four areas (areas 2 to 5). In an area designated as DRAM space, the $\overline{CS}$ pin functions as the $\overline{RAS}$ pin.

Features

- 2/4/8-Mbyte DRAM space can be set
- Address multiplexing
  - — Row address and column address are multiplexed.
  - — Selection of 8, 9, or 10 bits as the row address shift size
- Basic timing
  - — 4-state basic timing
  - — Wait state insertion possible
- DRAM interface
  - — 2-CAS system for control signals required for DRAM byte access
- Burst operation
  - — Fast page mode
- Refresh control
  - — Selection of CAS-before-RAS refreshing or self-refreshing
  - — Can be used as interval timer

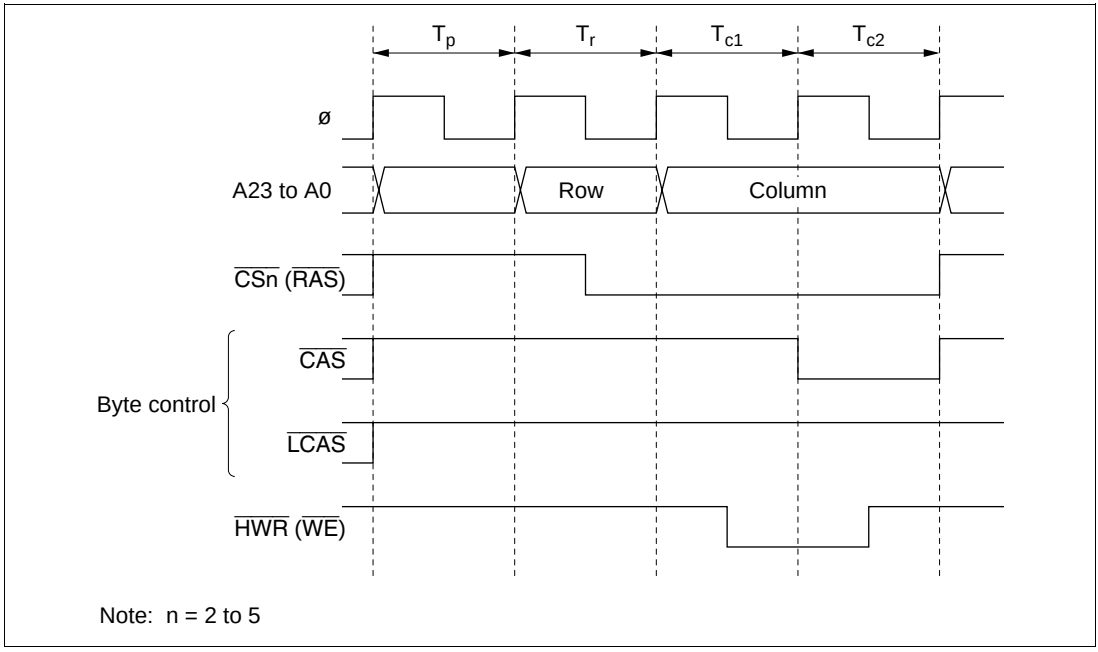
DRAM Basic Timing

- Basic Access Timing (2-CAS System)



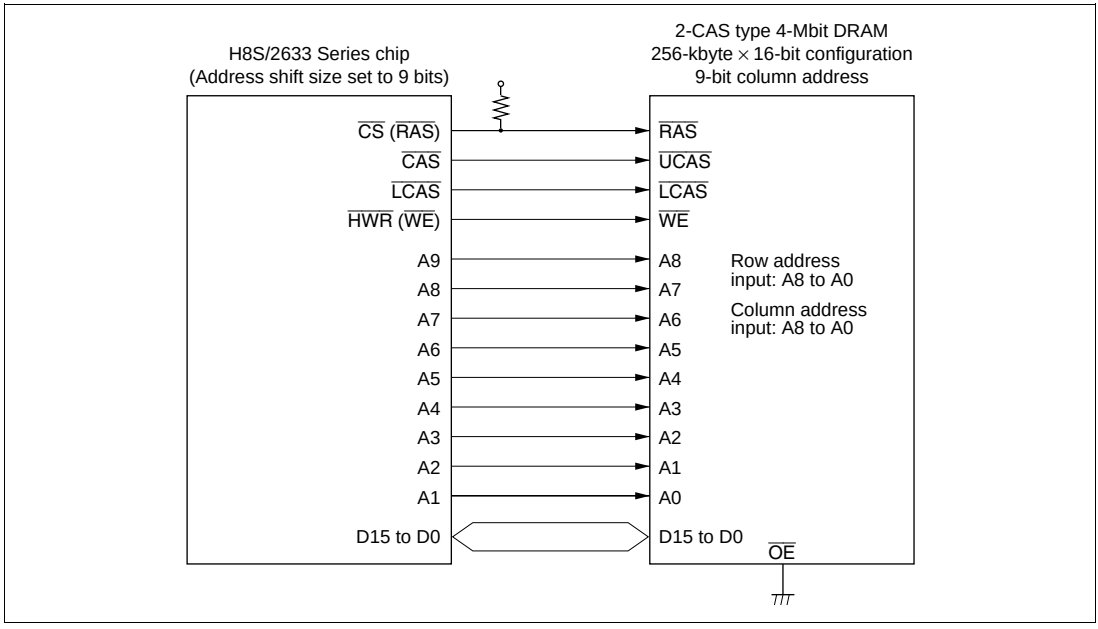
Note: n =2 to 5

- Byte Access Control Timing (Upper Byte Write Access)



Note: n = 2 to 5

- Example of 2-CAS Type (LCASS = 0) Connection

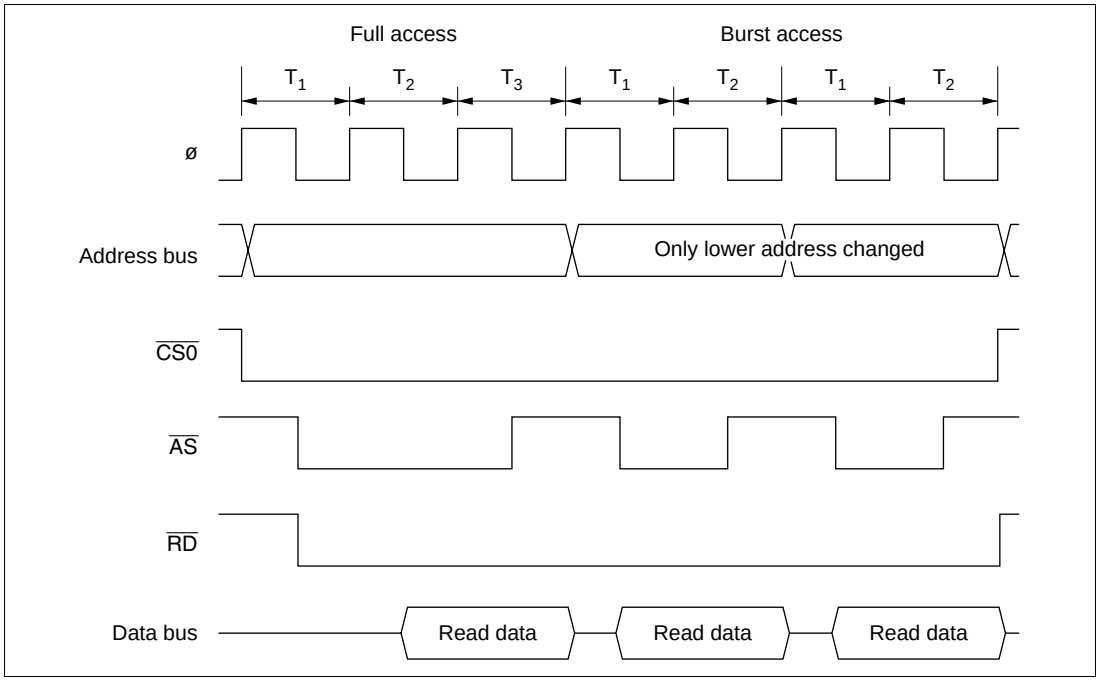


3.1.4 Burst ROM Interface

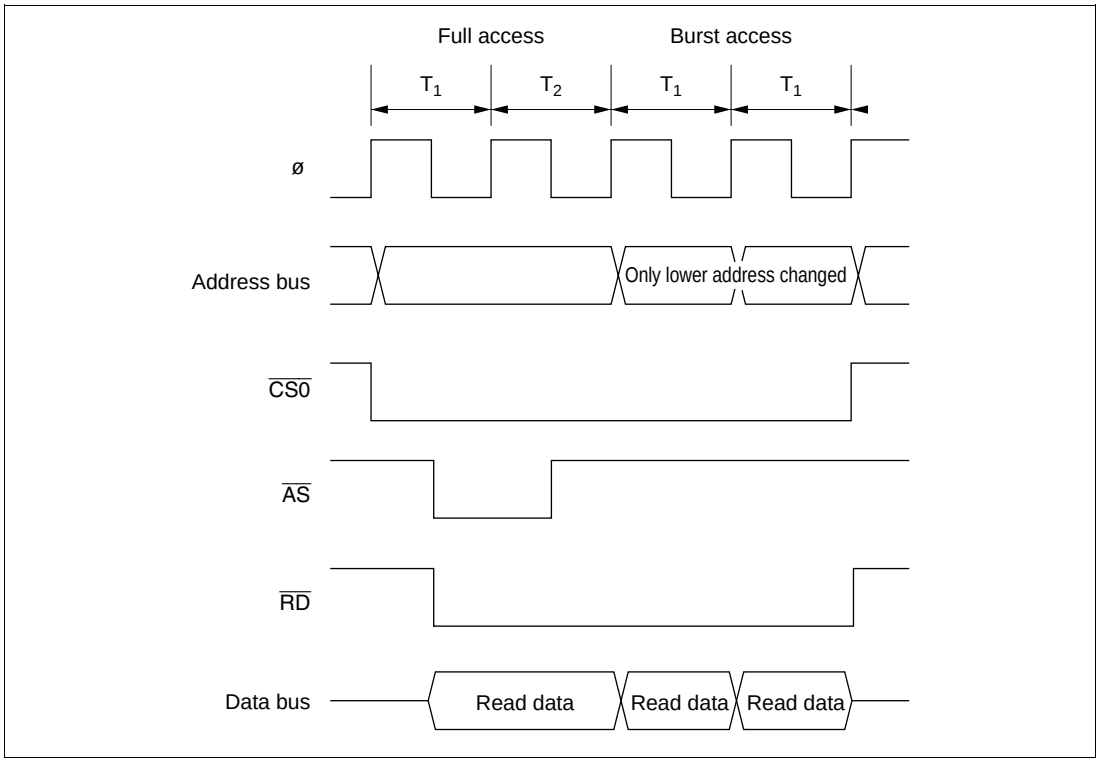
External space area 0 can be designated as burst ROM space, and burst ROM space interfacing can be performed. The burst ROM space interface enables 16-bit configuration ROM with burst access capability to be accessed at high speed.

Consecutive burst accesses of a maximum 4 words or 8 words can be performed for CPU instruction fetches only. One or two states can be selected for burst access.

- Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 1)



- Example of Burst ROM Access Timing (When AST0 = BRSTS1 = 0)



3.2 DMA Controller (DMAC)

The DMA controller (DMAC) can carry out data transfer on up to 4 channels (channels 0A, 0B, 1A, and 1B). Short address transfer can be performed on each channel independently, and full address transfer is possible by using pairs of channels.

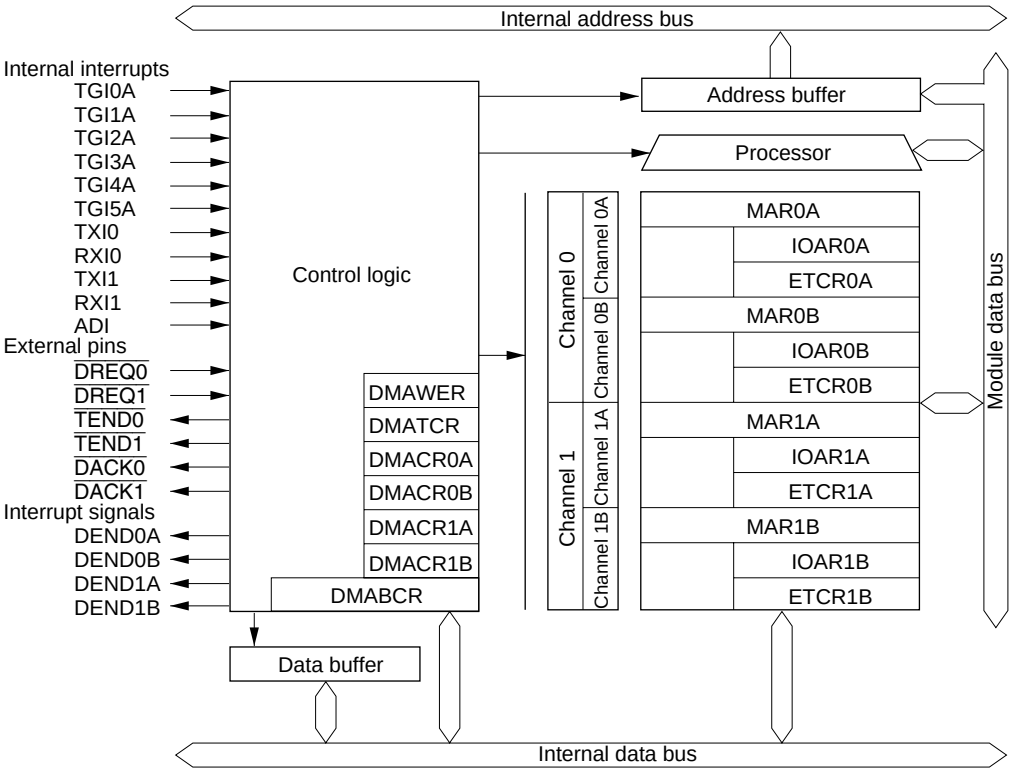
Features

- Choice of short address mode or full address mode

  Short address mode
  - — Maximum of 4 channels can be used
  - — Choice of dual address mode or single address mode
  - — In dual address mode, one of the two addresses, transfer source and transfer destination, is specified as 24 bits and the other as16 bits
  - — In single address mode, transfer source or transfer destination address only is specified as 24 bits
  - — In single address mode, transfer can be performed in one bus cycle
  - — Choice of sequential mode, idle mode, or repeat mode for dual address mode and single address mode

  Full address mode
  - — Maximum of 2 channels can be used
  - — Transfer source and transfer destination address specified as 24 bits
  - — Choice of normal mode or block transfer mode
- 16-Mbyte address space can be specified directly
- Byte or word can be set as the transfer unit
- Activation sources: internal interrupt, external request, auto-request (depending on transfer mode)
  - — Six 16-bit timer-pulse unit (TPU) compare match/input capture interrupts
  - — Serial communication interface (SCI0, SCI1) transmission complete interrupt, reception complete interrupt
  - — A/D converter conversion end interrupt
  - — External request
  - — Auto-request
- Module stop mode can be set
  - — The initial setting enables DMAC registers to be accessed. DMAC operation is halted by setting module stop mode

Block Diagram



Legend:

DMAWER:	DMA write enable register
DMATCR:	DMA terminal control register
DMABCR:	DMA band control register (for all channels)
DMACR:	DMA control register
MAR:	Memory address register
IOAR:	I/O address register
ETCR:	Executive transfer count register

Transfer Modes

The DMAC has the transfer modes shown in the table below. In short address mode, up to four-channel transfer is possible, with channels A and B operating independently. In full address mode, up to two-channel transfer is possible, with channels A and B combined.

- Overview of Transfer Modes

		Address Register Bit Length	
Transfer Mode	**Transfer Source**	**Source**	**Destination**
Dual address mode • Sequential mode — 1-byte or 1-word transfer executed for one transfer request — Memory address incremented/ decremented by 1 or 2 — 1 to 65536 transfers • Idle mode — 1-byte or 1-word transfer executed for one transfer request — Memory address fixed — 1 to 65536 transfers • Repeat mode — 1-byte or 1-word transfer executed for one transfer request — Memory address incremented/ decremented by 1 or 2 — After specified number of transfers (1 to 256), initial state is restored and operation continues	• TPU channel 0 to 5 compare match/input capture A interrupt • SCI transmission complete interrupt • SCI reception complete interrupt • A/D converter conversion end interrupt • External request	24/16	16/24
Single address mode • 1-byte or 1-word transfer executed for one transfer request • Transfer in 1 bus cycle using $\overline{\text{DACK}}$ pin in place of address specifying I/O • Specifiable for sequential, idle, and repeat modes	• External request	24/$\overline{\text{DACK}}$	$\overline{\text{DACK}}$/24

Transfer Mode	Transfer Source	Address Register Bit Length	
		Source	Destination
• Normal mode Auto-request — Transfer request retained internally — Transfers continue for the specified number of times (1 to 65536) — Choice of burst or cycle steal transfer	• Auto-request	24	24
External request — 1-byte or 1-word transfer executed for one transfer request — 1 to 65536 transfers	• External request		
• Block transfer mode — Specified block size transfer executed for one transfer request — 1 to 65536 transfers — Either source or destination specifiable as block area — Block size: 1 to 256 bytes or words	• TPU channel 0 to 5 compare match/input capture A interrupt • SCI transmission complete interrupt • SCI reception complete interrupt • External request • A/D converter conversion end interrupt	24	24

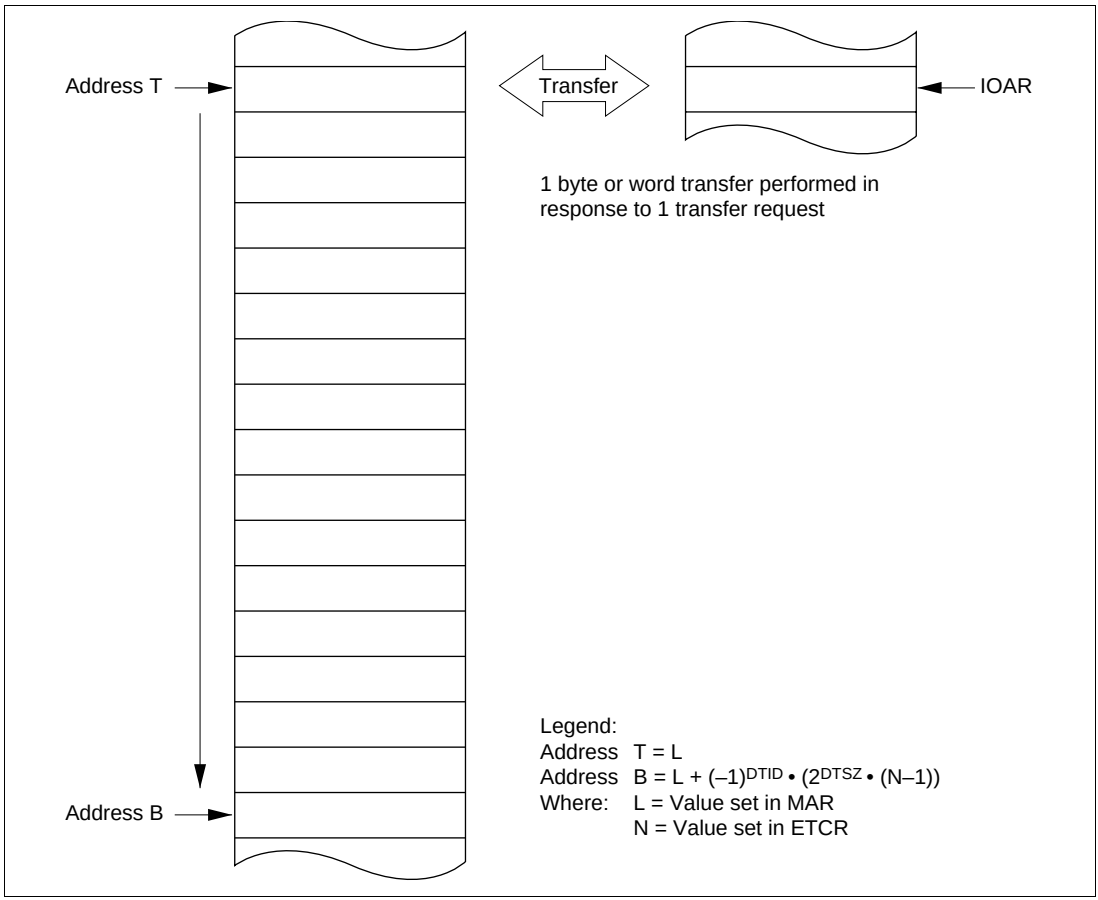
Short Address Mode

There are two kinds of short address mode—dual address mode and single address mode. For each of these modes, there is a choice of the following: (1) sequential mode, (2) idle mode, (3) repeat mode, (4) single address mode.

In short address mode, data transfer can be performed on a maximum of four channels.

(1) Sequential mode

In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. An interrupt request can be sent to the CPU or DTC when the specified number of transfers have been completed. One address is specified as 24 bits, and the other as 16 bits. The transfer direction is programmable.



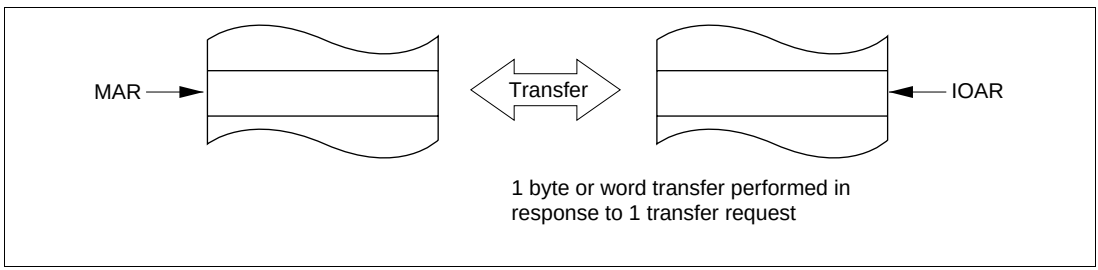
Operation in Sequential Mode

(2) Idle mode

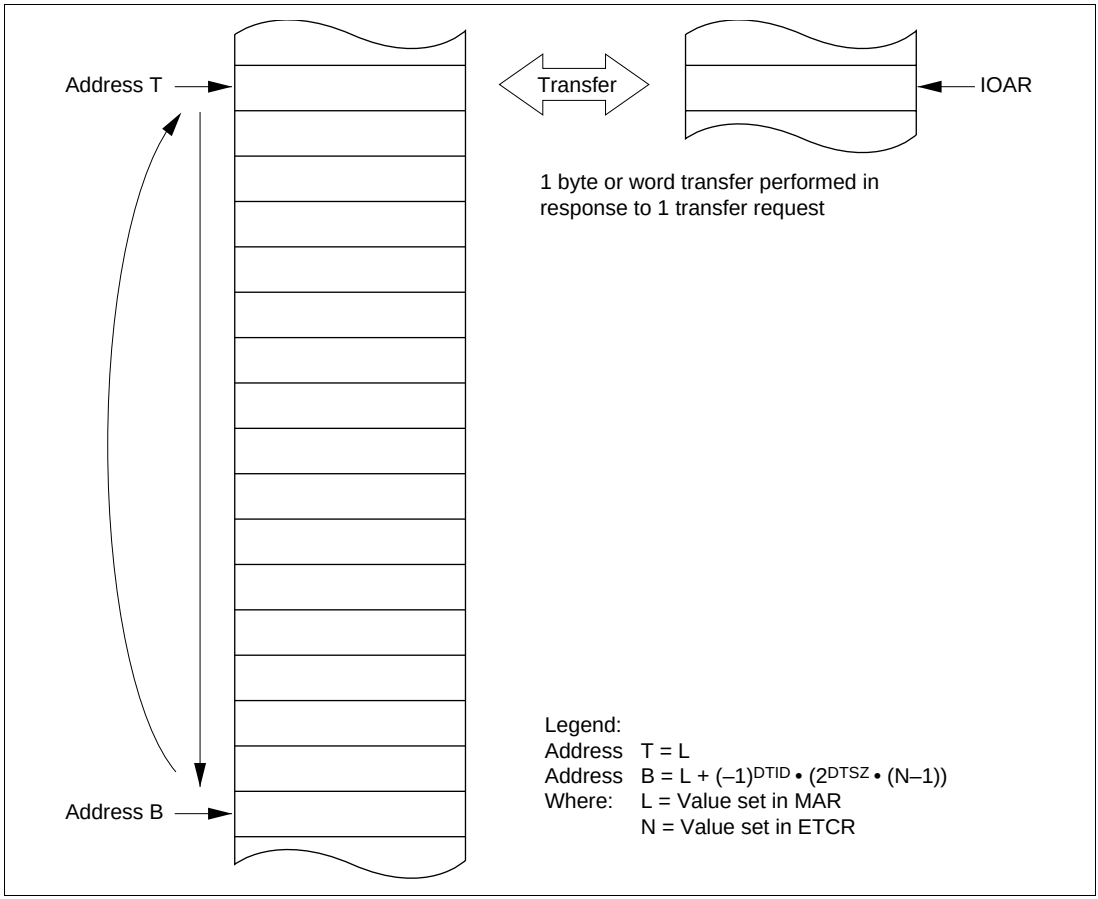
In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. An interrupt request can be sent to the CPU or DTC when the specified number of transfers have been completed. One address is specified as 24 bits, and the other as 16 bits. The transfer source address and transfer destination address are fixed. The transfer direction is programmable.

(3) Repeat mode

In response to a single transfer request, the specified number of transfers are carried out, one byte or one word at a time. When the specified number of transfers have been completed, the addresses and transfer counter are restored to their original settings, and operation is continued. No interrupt request is sent to the CPU or DTC. One address is specified as 24 bits, and the other as 16 bits. The transfer direction is programmable.



Operation in Idle Mode

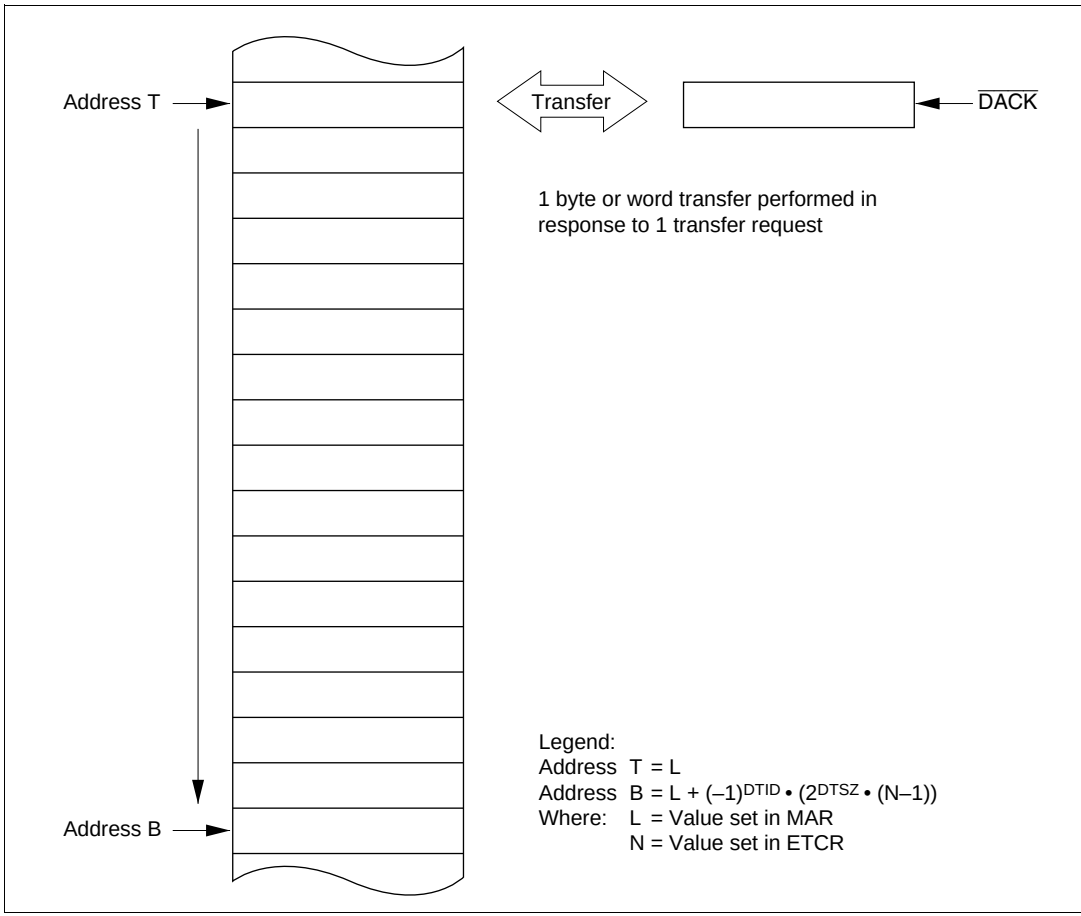


Operation in Repeat Mode

(4) Single address mode

In response to a single transfer request, the specified number of transfers are carried out between external memory and an external device, one byte or one word at a time. Unlike dual address mode, source and destination accesses are performed in parallel. Therefore, either the source or the destination is an external device which can be accessed with a strobe alone, using the $\overline{\text{DACK}}$ pin. One address is specified as 24 bits, and for the other, the pin is set automatically. The transfer direction is programmable.

Sequential, idle, and repeat modes can also be specified for single address mode. Single address mode can only be used on channel B.



Operation in Single Address Mode (When Sequential Mode is Specified)

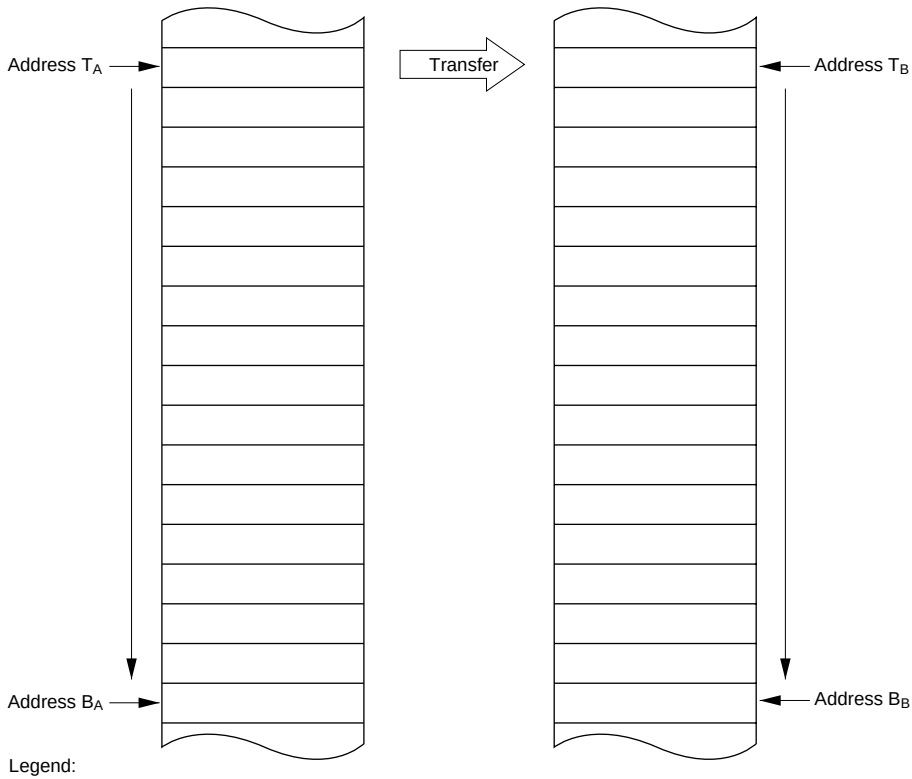
Full Address Mode

Full address mode includes (5) normal mode and (6) block transfer mode.

In full address mode, data transfer can be performed on a maximum of two channels, with channels A and B combined.

(5) Normal Mode

One byte or word is transferred per transfer request, and a designated number of these transfers are executed. A CPU or DTC interrupt can be requested on completion of the designated number of transfers. Both addresses are 24-bit addresses. There are two transfer requests (activation sources)—an external request and an auto-request.

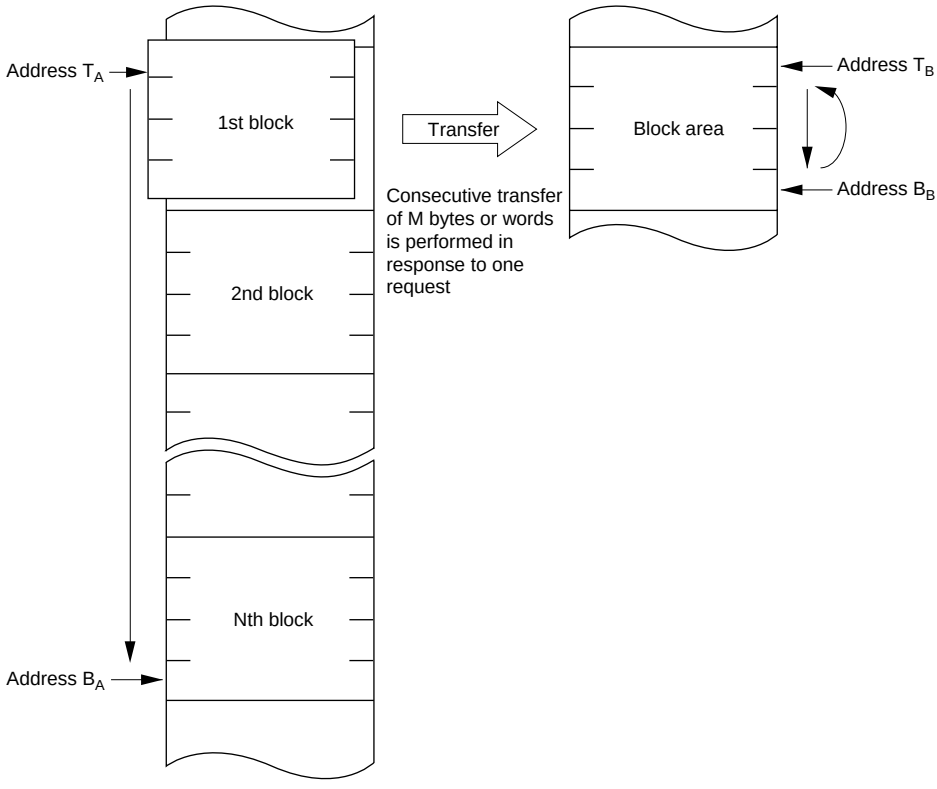


Legend:
Address $T_A = L_A$
Address $T_B = L_B$
Address $B_A = L_A + SAIDE \bullet (-1)^{SAID} \bullet (2^{DTSZ} \bullet (N-1))$
Address $B_B = L_B + DAIDE \bullet (-1)^{DAID} \bullet (2^{DTSZ} \bullet (N-1))$
Where: L_A = Value set in MARA
L_B = Value set in MARB
N = Value set in ETCRA

Operation in Normal Mode

(6) Block Transfer Mode

One block of the specified size is transfer per request, and a designated number of block transfers are executed. At the end of each block transfer, one address is restored to its initial value. When the designated number of blocks have been transferred, a CPU or DTC interrupt can be requested. Both addresses are 24-bit addresses.



Legend:
Address T_A = L_A
Address T_B = L_B
Address B_A = L_A + SAIDE • $(-1)^{SAID}$ • $(2^{DTSZ}$ • (M•N–1))
Address B_B = L_B + DAIDE • $(-1)^{DAID}$ • $(2^{DTSZ}$ • (N–1))
Where: L_A = Value set in MARA
L_B = Value set in MARB
N = Value set in ETCRB
M = Value set in ETCRAH and ETCRAL

Operation in Block Transfer Mode (When BLKDIR = 0: MARB is Block Area)

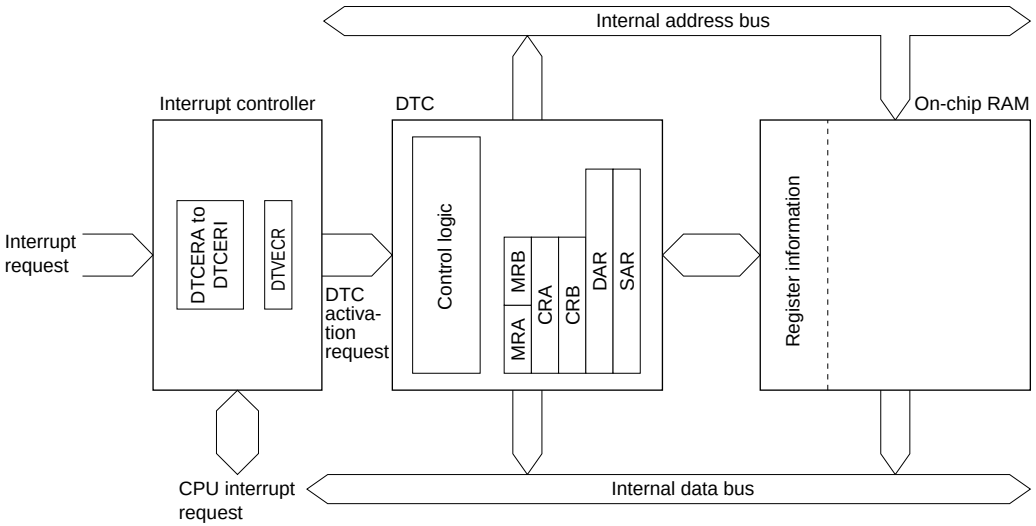
3.3 Data Transfer Controller (DTC)

The data transfer controller (DTC) is activated by an interrupt or software, and can transfer data without imposing any load on the CPU.

Features

- Transfer possible over any number of channels
  - — Transfer information is stored in memory
  - — One activation source can trigger a number of data transfers (chain transfer)
- Variety of transfer modes
  - — Normal, repeat, and block transfer modes available
  - — Incrementing, decrementing, or fixing of source and destination addresses can be selected
- Direct specification of 16-Mbyte address space possible
  - — 24-bit specification of transfer source and destination addresses
- Transfer can be set in byte or word units
- A CPU interrupt can be requested for the interrupt that activated the DTC
  - — An interrupt request can be issued to the CPU after one data transfer ends
  - — An interrupt request can be issued to the CPU after all specified data transfers have ended
- Can be activated by software
- Module stop mode can be set
  - — The initial setting enables DTC registers to be accessed. DTC operation is halted by setting module stop mode.

DTC Block Diagram



Legend:

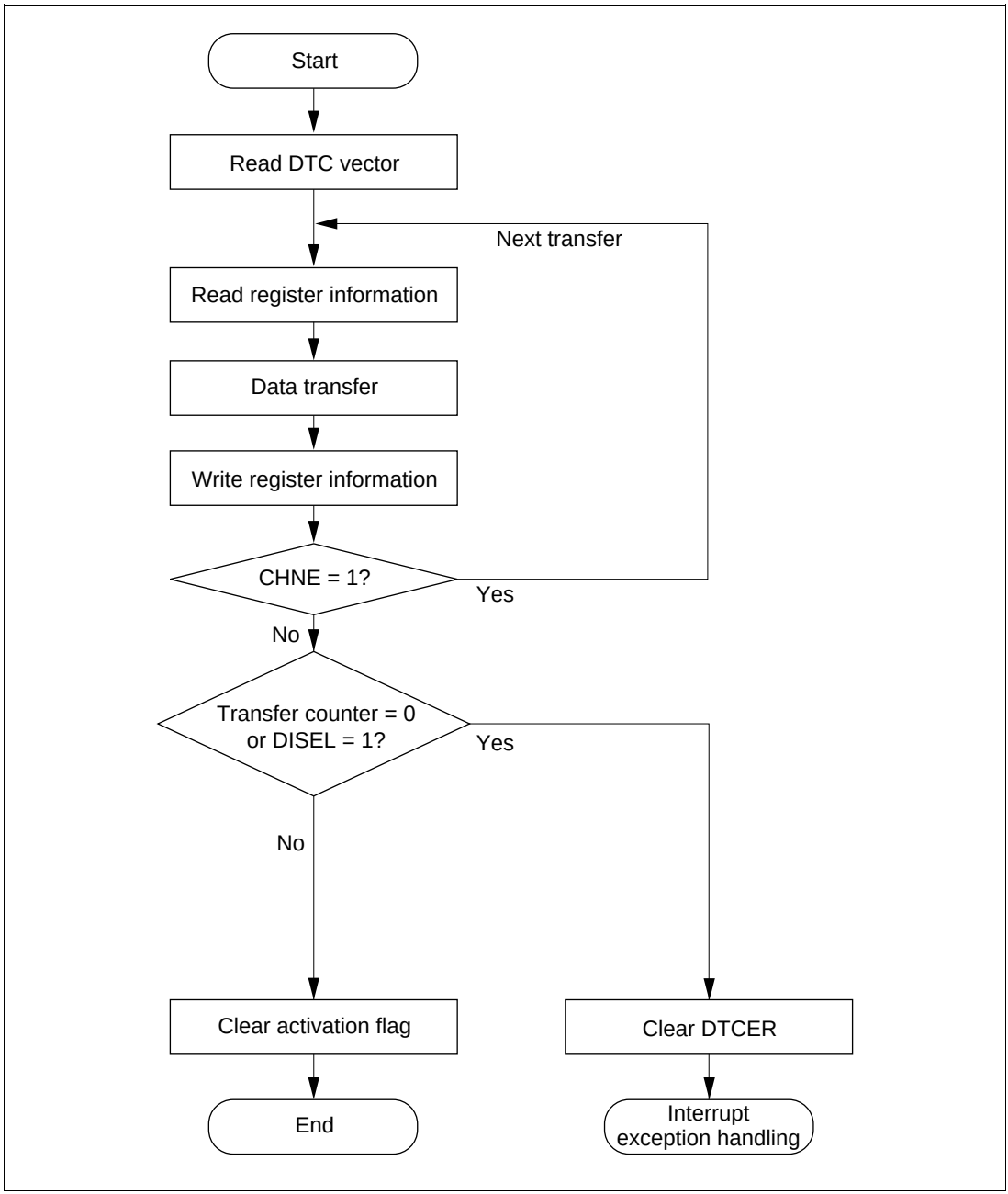
MRA, MRB:	DTC mode registers A and B
CRA, CRB:	DTC transfer count registers A and B
SAR:	DTC source address register
DAR:	DTC destination address register
DTCERA to DTCERI:	DTC enable registers A to I
DTVECR:	DTC vector register

Data Transfer Operation

When a DTC activation source occurs, the DTC reads register information previously stored in memory, and transfers data on the basis of that register information. After the data transfer, it writes updated register information back to memory.

Pre-storage of register information in memory makes it possible to transfer data over any required number of channels. The DTC can also execute a number of transfers with a single activation source (chain transfer).

- Flowchart of DTC Operation



DTC Activation Sources

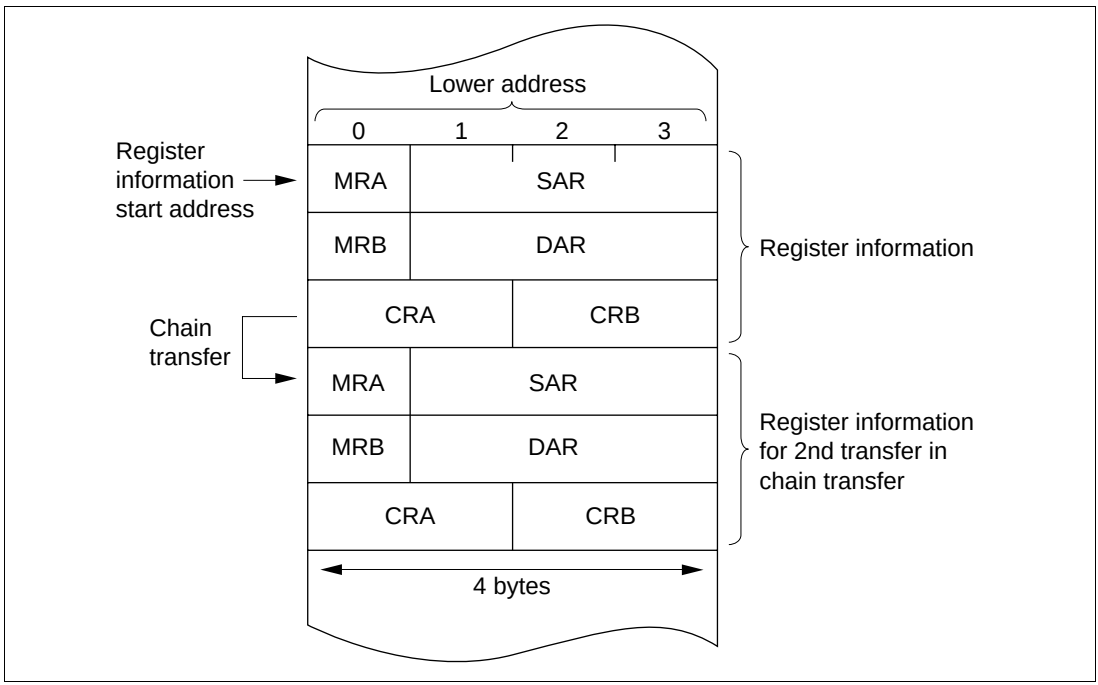
The DTC is activated by an interrupt source or by a vector number write to the DTC vector register (DTVECR) by software. An interrupt request can be designated as a CPU interrupt source or a DTC activation source.

When an interrupt has been designated a DTC activation source, existing CPU mask level and interrupt controller priorities have no effect. If there is more than one activation source at the same time, the DTC operates in accordance with the default priorities.

- Interrupt Sources and DTC Vector Address

  The DTC vector address indicates the start address of the register information in memory. The MRA, SAR, MRB, DAR, CRA, and CRB registers are located in that order from the start address of the register information. Locate the register information in the on-chip RAM (addresses H'FFEBC0 to H'FFEFBF).

- Location of DTC Register Information in Address Space



- Interrupt Sources, DTC Vector Addresses, and Corresponding DTCEs

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address	DTCE*	Priority
Write to DTVECR	Software	DTVECR	H'0400+ (DTVECR [6:0]<<1)	—	High
IRQ0	External pin	16	H'0420	DTCEA7	
IRQ1		17	H'0422	DTCEA6	
IRQ2		18	H'0424	DTCEA5	
IRQ3		19	H'0426	DTCEA4	
IRQ4		20	H'0428	DTCEA3	
IRQ5		21	H'042A	DTCEA2	
IRQ6		22	H'042C	DTCEA1	
IRQ7		23	H'042E	DTCEA0	
ADI (A/D conversion end)	A/D	28	H'0438	DTCEB6	
TGI0A (GR0A compare match/ input capture)	TPU channel 0	32	H'0440	DTCEB5	
TGI0B (GR0B compare match/ input capture)		33	H'0442	DTCEB4	
TGI0C (GR0C compare match/ input capture)		34	H'0444	DTCEB3	
TGI0D (GR0D compare match/ input capture)		35	H'0446	DTCEB2	
TGI1A (GR1A compare match/ input capture)	TPU channel 1	40	H'0450	DTCEB1	
TGI1B (GR1B compare match/ input capture)		41	H'0452	DTCEB0	
TGI2A (GR2A compare match/ input capture)	TPU channel 2	44	H'0458	DTCEC7	
TGI2B (GR2B compare match/ input capture)		45	H'045A	DTCEC6	Low

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address	DTCE*	Priority
TGI3A (GR3A compare match/ input capture)	TPU channel 3	48	H'0460	DTCEC5	High
TGI3B (GR3B compare match/ input capture)		49	H'0462	DTCEC4	
TGI3C (GR3C compare match/ input capture)		50	H'0464	DTCEC3	
TGI3D (GR3D compare match/ input capture)		51	H'0466	DTCEC2	
TGI4A (GR4A compare match/ input capture)	TPU channel 4	56	H'0470	DTCEC1	
TGI4B (GR4B compare match/ input capture)		57	H'0472	DTCEC0	
TGI5A (GR5A compare match/ input capture)	TPU channel 5	60	H'0478	DTCED5	
TGI5B (GR5B compare match/ input capture)		61	H'047A	DTCED4	
CMIA0A	8-bit timer channel 0A	64	H'0480	DTCED3	
CMIB0A		65	H'0482	DTCED2	
CMIA1A	8-bit timer channel 1A	68	H'0488	DTCED1	
CMIB1A		69	H'048A	DTCED0	
DEND0A (channel 0/channel 0A transfer end)	DMAC	72	H'0490	DTCEE7	
DEND0B (channel 0B transfer end)		73	H'0492	DTCEE6	
DEND1A (channel 1/channel 1A transfer end)		74	H'0494	DTCEE5	
DEND1B (channel 1B transfer end)		75	H'0496	DTCEE4	
RXI0 (reception complete 0)	SCI channel 0	81	H'04A2	DTCEE3	
TXI0 (transmit data empty 0)		82	H'04A4	DTCEE2	
RXI1 (reception complete 1)	SCI channel 1	85	H'04AA	DTCEE1	
TXI1 (transmit data empty 1)		86	H'04AC	DTCEE0	
RXI2 (reception complete 2)	SCI channel 2	89	H'04B2	DTCEF7	
TXI2 (transmit data empty 2)		90	H'04B4	DTCEF6	Low

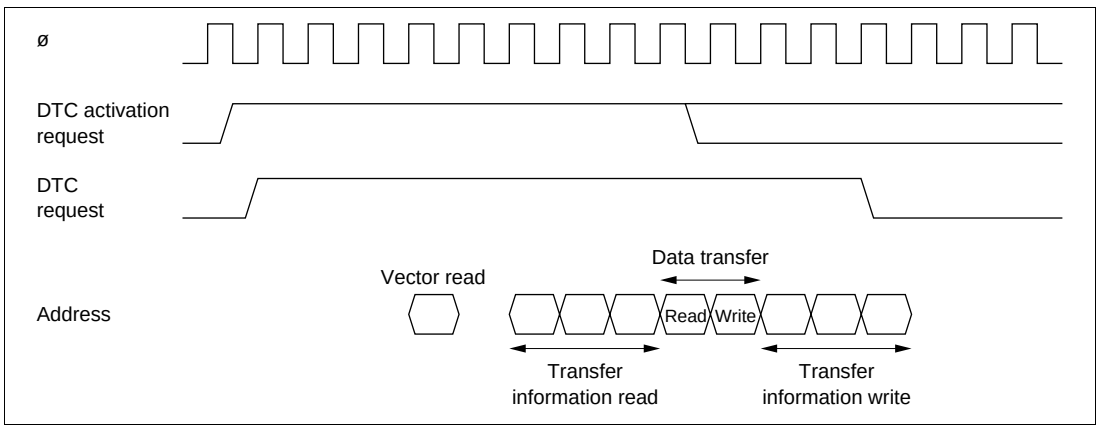
Note: * DTCE bits with no corresponding interrupt are reserved, and should be written with 0.

Interrupt Source	Origin of Interrupt Source	Vector Number	Vector Address	DTCE*	Priority
CMIA2A	8-bit timer channel 2A	92	H'04B8	DTCEF5	High
CMIB2A		93	H'04BA	DTCEF4	
CMIA3A	8-bit timer channel 3A	96	H'04C0	DTCEF3	
CMIB3A		97	H'04C2	DTCEF2	
IICI0 (1-byte transmission/ reception completed	IIC channel 0 [option]	100	H'04C8	DTCEF1	
IICI1 (1-byte transmission/ reception completed	IIC channel 1 [option]	102	H'04CC	DTCEF0	
RXI3 (recepiton complete 3)	SCI channel 3	121	H'04F2	DTCEI7	
TXI3 (transmit data empty 3)		122	H'04F4	DTCEI6	
RXI4 (recepiton complete 4)	SCI channel 4	125	H'04FA	DTCEI5	
TXI4 (transmit data empty 4)		126	H'04FC	DTCEI4	Low

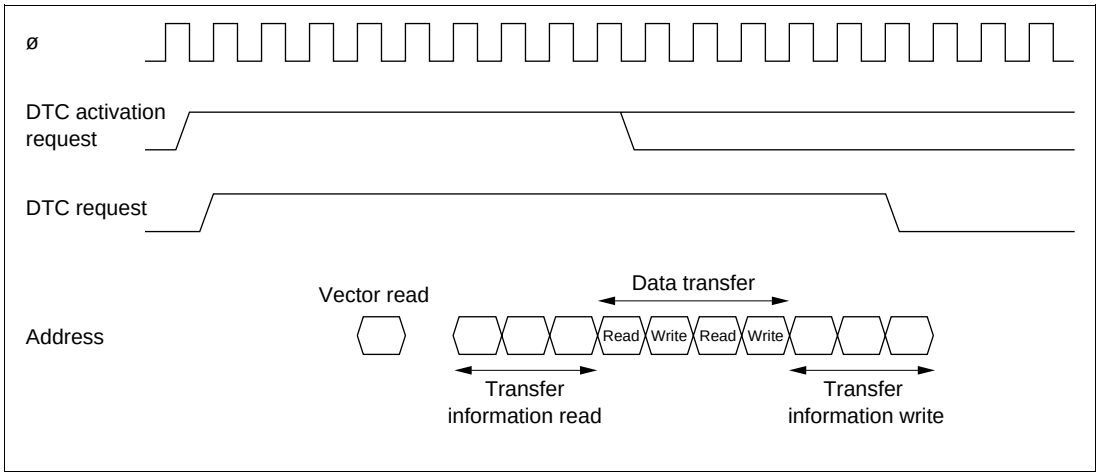
Note: * DTCE bits with no corresponding interrupt are reserved, and should be written with 0.

DTC Operation Timing

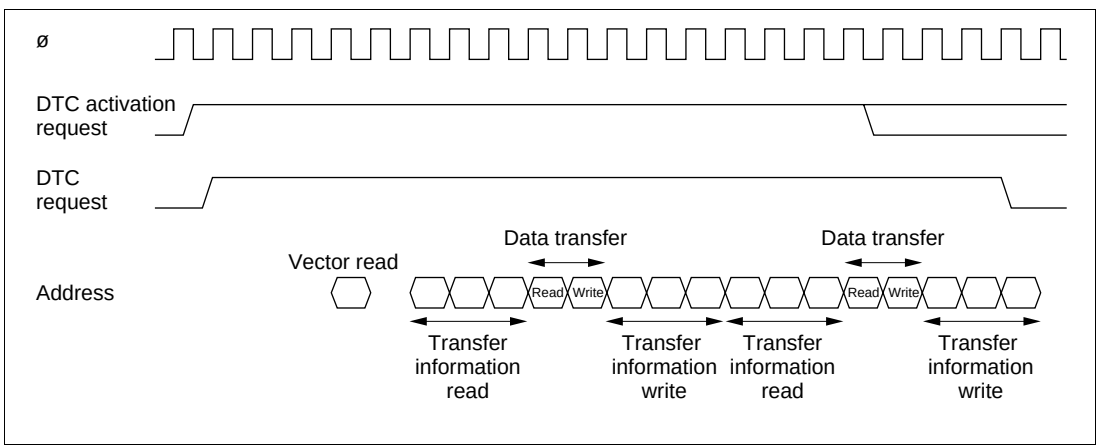
- Example in Normal Mode or Repeat Mode



- Example of Block Transfer Mode (Block Size = 2)



- Example of Chain Transfer



Number of DTC Execution States

Mode	Vector Read I	Register Information Read/Write J	Data Read K	Data Write L	Internal Operations M
Normal	1	6	1	1	3
Repeat	1	6	1	1	3
Block transfer	1	6	N	N	3

N: Block size (initial setting of CRA)

- Number of States Required in Each Execution State

Access To:			On-Chip RAM	On-Chip ROM	Internal I/O Registers		External Devices			
Bus width			32	16	8	16	8	8	16	16
Access states			1	1	2	2	2	3	2	3
Execution state	Vector read	S_I	—	1	—	—	4	6+2m	2	3+m
	Register information read/write	S_J	1	—	—	—	—	—	—	—
	Byte data read	S_K	1	1	2	2	2	3+m	2	3+m
	Word data read	S_K	1	1	4	2	4	6+2m	2	3+m
	Byte data write	S_L	1	1	2	2	2	3+m	2	3+m
	Word data write	S_L	1	1	4	2	4	6+2m	2	3+m
	Internal operation	S_M	1	1	1	1	1	1	1	1

Note: The number of execution states is calculated from the formula below.

Number of execution states = $I \cdot S_I + \Sigma (J \cdot S_J + K \cdot S_K + L \cdot S_L) + M \cdot S_M$

Σ indicates the sum of all transfers activated by one activation event (the number in which the CHNE bit is set to 1, plus 1).

Transfer Modes

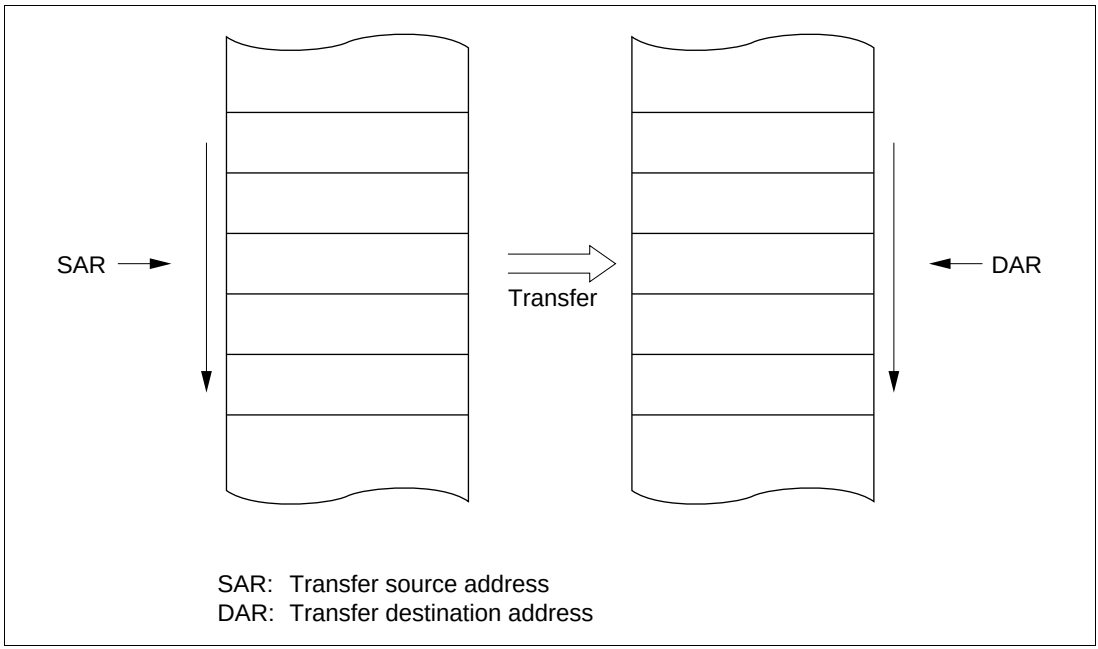
There are three DTC transfer modes—normal mode, repeat mode, and block transfer mode.

In the DTC, the 24-bit source address register (SAR) designates the transfer source address and the 24-bit destination address register (DAR) designates the transfer destination address. After each transfer, SAR and DAR are independently incremented, decremented, or left unchanged.

- Overview of DTC Functions

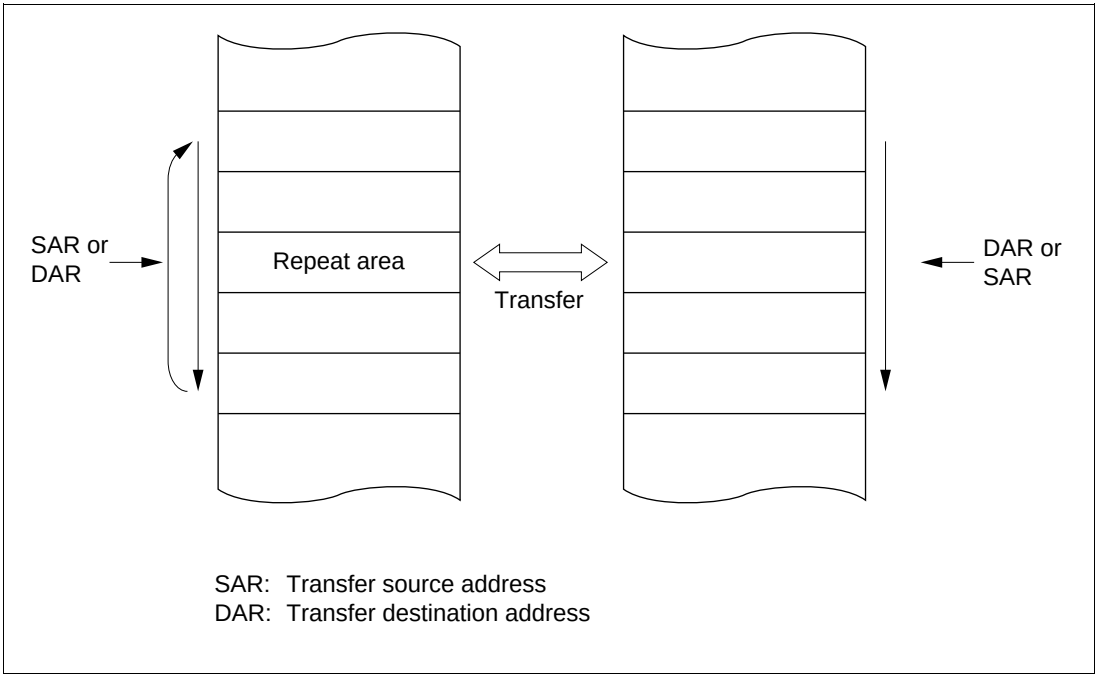
		Address Registers	
Transfer Mode	**Activation Source**	**Transfer Source**	**Transfer Destination**
• Normal mode — 1-byte or 1-word transfer executed for one transfer request — Memory addresses are incremented or decremented by 1 or 2 — Up to 65,536 transfers possible • Repeat mode — 1-byte or 1-word transfer executed for one transfer request — Memory addresses are incremented or decremented by 1 or 2 — After the specified number of transfers (1 to 256), the initial state is restored and operation continues • Block transfer mode — One transfer request transfers a block of the specified size — Block size is from 1 to 256 bytes or words — Up to 65,536 transfers possible — A block area can be designated at either the source or destination	• IRQ • TPU TGI • 8-bit timer CMI • SCI TXI or RXI • A/D converter ADI • DMAC DEND • Software	24 bits	24 bits

Operation in Normal Mode: In normal mode, one byte or one word of data is transferred in one operation. From 1 to 65,536 transfers can be specified. When the specified number of transfers have ended, a CPU interrupt can be requested.



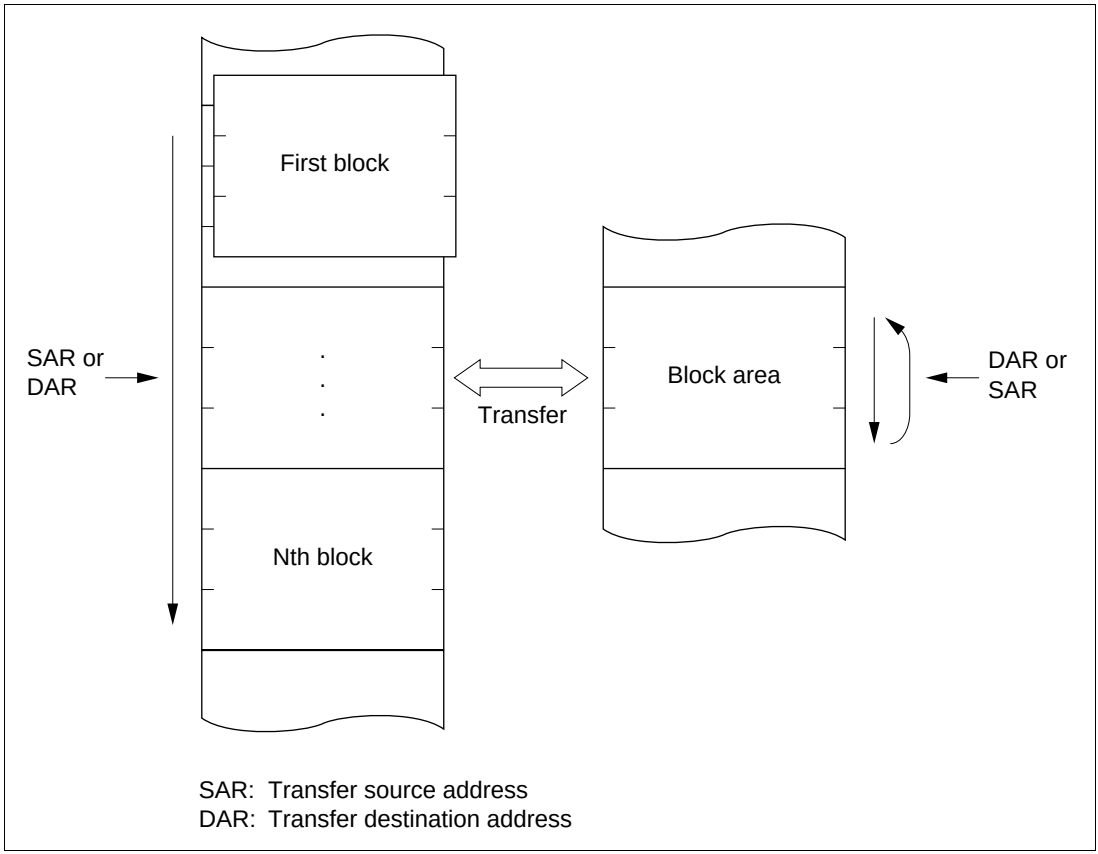
SAR: Transfer source address
DAR: Transfer destination address

Operation in Repeat Mode: In repeat mode, one byte or one word of data is transferred in one operation. From 1 to 256 transfers can be specified. When the specified number of transfers have ended, the initial state specified in the transfer counter and repeat area is restored and transfer is repeated.



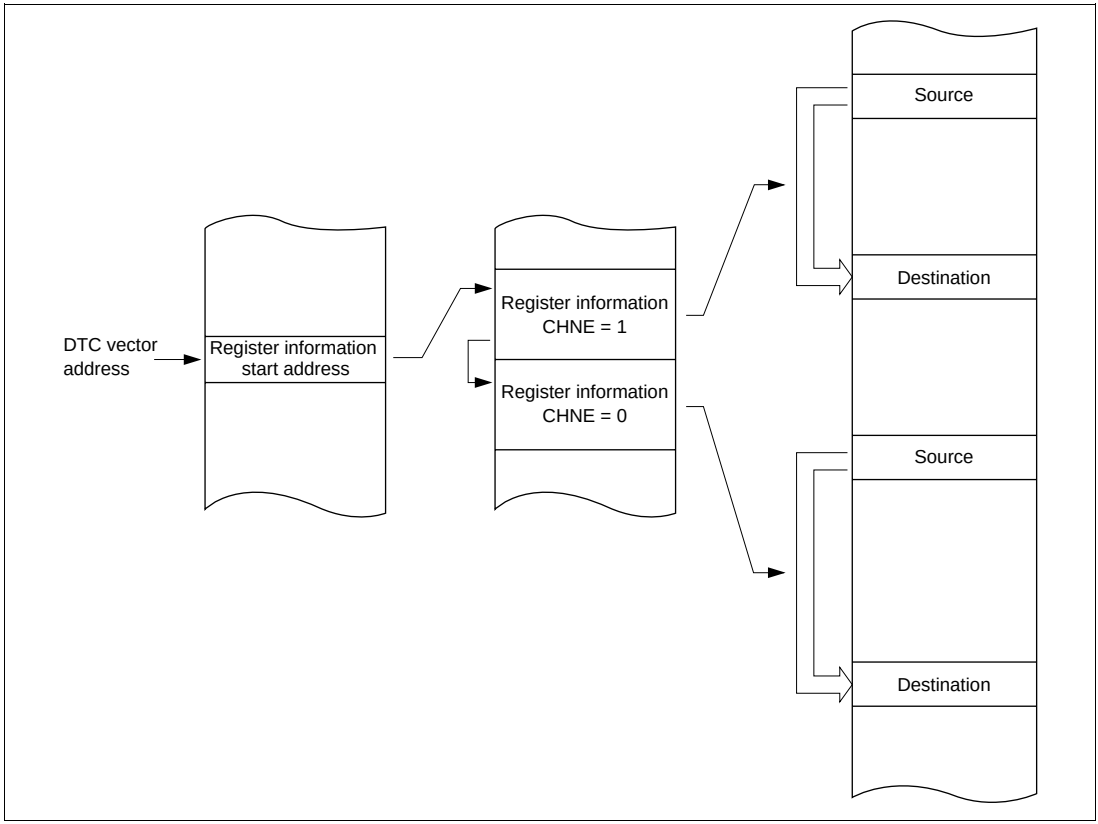
Operation in Block Transfer Mode: In block transfer mode, one operation transfers one block of data. Either the transfer source or the transfer destination is specified as a block area. The block size is 1 to 256. When the transfer of one block ends, the initial settings of the block size counter and the address register specified in the block area are restored. The other address register is continually incremented, decremented, or left unchanged.

From 1 to 65,536 transfers can be specified. When the specified number of transfers have ended, a CPU interrupt can be requested.



Chain Transfer

Setting the CHNE bit to 1 in DTC mode register B enables a number of data transfers to be performed consecutively in response to a single transfer request.



3.4 I/O Ports

This series has ten input/output ports (ports 1, 3, 7, A, B, C, D, E, F, and G), and two input ports (ports 4 and 9).

Each port includes a data direction register (DDR) that controls input/output, a data register (DR) that stores output data, and a port register (PORT) used to read the pin states.

Ports A to E incorporate a MOS input pull-up, and in addition to DDR and DR, have a MOS input pull-up control register (PCR) that turns the MOS input pull-up on or off.

H8S/2633 Series Port Functions in Each Operating Mode

Port	Description	Pins	Mode 4	Mode 5	Mode 6	Mode 7
Port 1	• 8-bit I/O port • Schmitt-triggered input (P16, P14)	P17/PO15/TIOCB2/PWM3/TCLKD P16/PO14/TIOCA2/PWM2/$\overline{\text{IRQ1}}$ P15/PO13/TIOCB1/TCLKC P14/PO12/TIOCA1/$\overline{\text{IPQ0}}$ P13/PO11/TIOCD0/TCLKB/A23 P12/PO10/TIOCC0/TCLKA/A22 P11/PO9/TIOCB0/$\overline{\text{DACK1}}$/A21 P10/PO8/TIOCA0/$\overline{\text{DACK0}}$/A20	8-bit I/O port also functioning as DMA controller output pins ($\overline{\text{DACK0}}$, $\overline{\text{DACK1}}$), TPU I/O pins (TCLKA, TCLKB, TCLKC, TCLKD, TIOCA0, TIOCB0, TIOCC0, TIOCD0, TIOCA1, TIOCB1, TIOCA2, TIOCB2), PPG output pins (PO15 to PO8), interrupt input pins ($\overline{\text{IRQ0}}$, $\overline{\text{IRQ1}}$), 14-bit PWM output pins (PWM2, PWM3), and address outputs (A20 to A23)			8-bit I/O port also functioning as DMA controller output pins ($\overline{\text{DACK0}}$, $\overline{\text{DACK1}}$), TPU I/O pins (TCLKA, TCLKB, TCLKC, TCLKD, TIOCA0, TIOCB0, TIOCC0, TIOCD0, TIOCA1, TIOCB1, TIOCA2, TIOCB2), PPG output pins (PO15 to PO8), interrupt input pins ($\overline{\text{IRQ0}}$, $\overline{\text{IRQ1}}$), and 14-bit PWM output pins (PWM2, PWM3)
Port 3	• 8-bit I/O port • Open-drain output capability • Schmitt-triggered input (P35, P32)	P37/TxD4 P36/RxD4 P35/SCK1/SCK4/SCL0/$\overline{\text{IRQ5}}$ P34/RxD1/SDA0 P33/TxD1/SCL1 P32/SCK0/SDA1/$\overline{\text{IRQ4}}$ P31/RxD0/IrRxD P30/TxD0/IrTxD	8-bit I/O port also functioning as SCI (channel 0, 1, and 4) I/O pins (TxD0, RxD0, SCK0, IrTxD, IrRxD, TxD1, RxD1, SCK1, TxD4, RxD4, SCK4), interrupt input pins ($\overline{\text{IRQ4}}$, $\overline{\text{IRQ5}}$), and IIC (channel 0 and 1) I/O pins (SCL0, SDA0, SCL1, SDA1)			

Port	Description	Pins	Mode 4	Mode 5	Mode 6	Mode 7
Port 4	• 8-bit input port	P47/AN7/DA1 P46/AN6/DA0 P45/AN5 P44/AN4 P43/AN3 P42/AN2 P41/AN1 P40/AN0	8-bit input port also functioning as A/D converter analog inputs (AN7 to AN0) and D/A converter analog outputs (DA1, DA0)			
Port 7	• 8-bit I/O port	P77/TxD3 P76/RxD3 P75/TMO3/SCK3 P74/TMO2/$\overline{\text{MRES}}$ P73/$\overline{\text{TEND1}}$/TMO1/$\overline{\text{CS7}}$ P72/$\overline{\text{TEND0}}$/TMO0/$\overline{\text{CS6}}$/SYNCI P71/TMRI23/TMCI23/$\overline{\text{CS5}}$/$\overline{\text{DREQ1}}$ P70/TMRI01/TMCI01/$\overline{\text{CS4}}$/$\overline{\text{DREQ0}}$	8-bit I/O port also functioning as 8-bit timer I/O pins (TMRI01, TMCI01, TMRI23, TMCI23, TMO0, TMO1, TMO2, TMO3), DMAC I/O pins ($\overline{\text{DREQ0}}$, $\overline{\text{TEND0}}$, $\overline{\text{DREQ1}}$, $\overline{\text{TEND1}}$), bus control output pins ($\overline{\text{CS4}}$ to $\overline{\text{CS7}}$), the IIC input pin (SYNCI), SCI I/O pins (SCK3, RxD3, TxD3), and the manual reset input pin ($\overline{\text{MRES}}$)			8-bit I/O port also functioning as 8-bit timer I/O pins (TMRI01, TMCI01, TMRI23, TMCI23, TMO0, TMO1, TMO2, TMO3), DMAC I/O pins ($\overline{\text{DREQ0}}$, $\overline{\text{TEND0}}$, $\overline{\text{DREQ1}}$, $\overline{\text{TEND1}}$), the IIC input pin (SYNCI), SCI I/O pins (SCK3, RxD3, TxD3), and the manual reset input pin ($\overline{\text{MRES}}$)
Port 9	• 8-bit input port	P97/AN15/DA3 P96/AN14/DA2 P95/AN13 P94/AN12 P93/AN11 P92/AN10 P91/AN9 P90/AN8	8-bit input port also functioning as A/D converter analog inputs (AN15 to AN8) and D/A converter analog outputs (DA3, DA2)			
Port A	• 4-bit I/O port • Built-in MOS input pull-up • Open-drain output capability	PA3/A19/SCK2 PA2/A18/RxD2 PA1/A17/TxD2 PA0/A16	4-bit I/O port also functioning as SCI (channel 2) I/O pins (TxD2, RxD2, SCK2) and address outputs (A19 to A16)			4-bit I/O port also functioning as SCI (channel 2) I/O pins (TxD2, RxD2, SCK2)

Port	Description	Pins	Mode 4	Mode 5	Mode 6	Mode 7
Port B	• 8-bit I/O port • Built-in MOS input pull-up • Open-drain output capability	PB7/A15/TIOCB5 PB6/A14/TIOCA5 PB5/A13/TIOCB4 PB4/A12/TIOCA4 PB3/A11/TIOCD3 PB2/A10/TIOCC3 PB1/A9/TIOCB3 PB0/A8/TIOCA3	8-bit I/O port also functioning as TPU I/O pins (TIOCB5, TIOCA5, TIOCB4, TIOCA4, TIOCD3, TIOCC3, TIOCB3, TIIOCA3) and address outputs (A15 to A8)			8-bit I/O port also functioning as TPU I/O pins (TIOCB5, TIOCA5, TIOCB4, TIOCA4, TIOCD3, TIOCC3, TIOCB3, TIIOCA3)
Port C	• 8-bit I/O port • Built-in MOS input pull-up • Open-drain output capability	PC7/A7/PWM1 PC6/A6/PWM0 PC5/A5 to PC0/A0	8-bit I/O port also functioning as 14-bit PWM (channel 1 and 0) output pins (PWM1, PWM0) and address outputs (A7 to A0)			8-bit I/O port also functioning as 14-bit PWM (channel 1 and 0) output pins (PWM1, PWM0)
Port D	• 8-bit I/O port • Built-in MOS input pull-up	PD7/D15 to PD0/D8	Data bus input/output			I/O port
Port E	• 8-bit I/O port • Built-in MOS input pull-up	PE7/D7 to PE0/D0	In 8-bit-bus mode: I/O port In 16-bit-bus mode: data bus input/output			I/O port
Port F	• 8-bit I/O port	PF7/ø	When DDR = 0: input port When DDR = 1 (after reset): ø output			When DDR = 0 (after reset): input port When DDR = 1: ø output
		PF6/$\overline{AS}$/$\overline{LCAS}$ PF5/$\overline{RD}$ PF4/$\overline{HWR}$ PF3/$\overline{LWR}$/$\overline{ADTRG}$/$\overline{IRQ3}$	$\overline{RD}$, $\overline{HWR}$, $\overline{LWR}$ outputs $\overline{ADTRG}$, $\overline{IRQ3}$ input When LCASS = 0: $\overline{AS}$ output When RMTS2 to RMTS0 = B'001 to B'011, CW2 = 0, and LCASS = 1: $\overline{LCAS}$ output			I/O port $\overline{ADTRG}$, $\overline{IRQ3}$ input
		PF2/$\overline{LCAS}$/$\overline{WAIT}$/$\overline{BREQO}$	When WAITE = 0 and BREQOE = 0 (after reset): I/O port When WAITE = 1 and BREQOE = 0: $\overline{WAIT}$ input When WAITE = 0 and BREQOE = 1: $\overline{BREQO}$ input When RMTS2 to RMTS0 = B'001 to B'011, CW2 = 0, and LCASS = 0: $\overline{LCAS}$ output			I/O port
		PF1/$\overline{BACK}$/BUZZ PF0/$\overline{BREQ}$/$\overline{IRQ2}$	When BRLE = 0 (after reset): I/O port When BRLE = 1: $\overline{BREQ}$ input, $\overline{BACK}$ output BUZZ output, $\overline{IRQ2}$ output			BUZZ output $\overline{IRQ2}$ output I/O port

Port	Description	Pins	Mode 4	Mode 5	Mode 6	Mode 7
Port G	• 5-bit I/O port	PG4/$\overline{CS0}$	When DDR = 0*1: input port When DDR = 1*2: $\overline{CS0}$ output			I/O port
		PG3/$\overline{CS1}$ PG2/$\overline{CS2}$ PG1/$\overline{CS3}$/$\overline{OE}$/$\overline{IRQ7}$	When DDR = 0 (after reset): input port When DDR = 1: $\overline{CS1}$, $\overline{CS2}$, $\overline{CS3}$ outputs $\overline{OE}$ output, $\overline{IRQ7}$ input			I/O port, $\overline{IRQ7}$ input
		PG0/$\overline{CAS}$/$\overline{IRQ6}$	DRAM space set: $\overline{CAS}$ output Otherwise (after reset): I/O port $\overline{IRQ6}$ input			I/O port, $\overline{IRQ6}$ input

Notes: 1. After a reset in mode 6
2. After a reset in mode 4 or 5

3.5 16-Bit Timer Pulse Unit (TPU)

The 16-bit timer pulse unit (TPU) comprises six 16-bit timer channels.

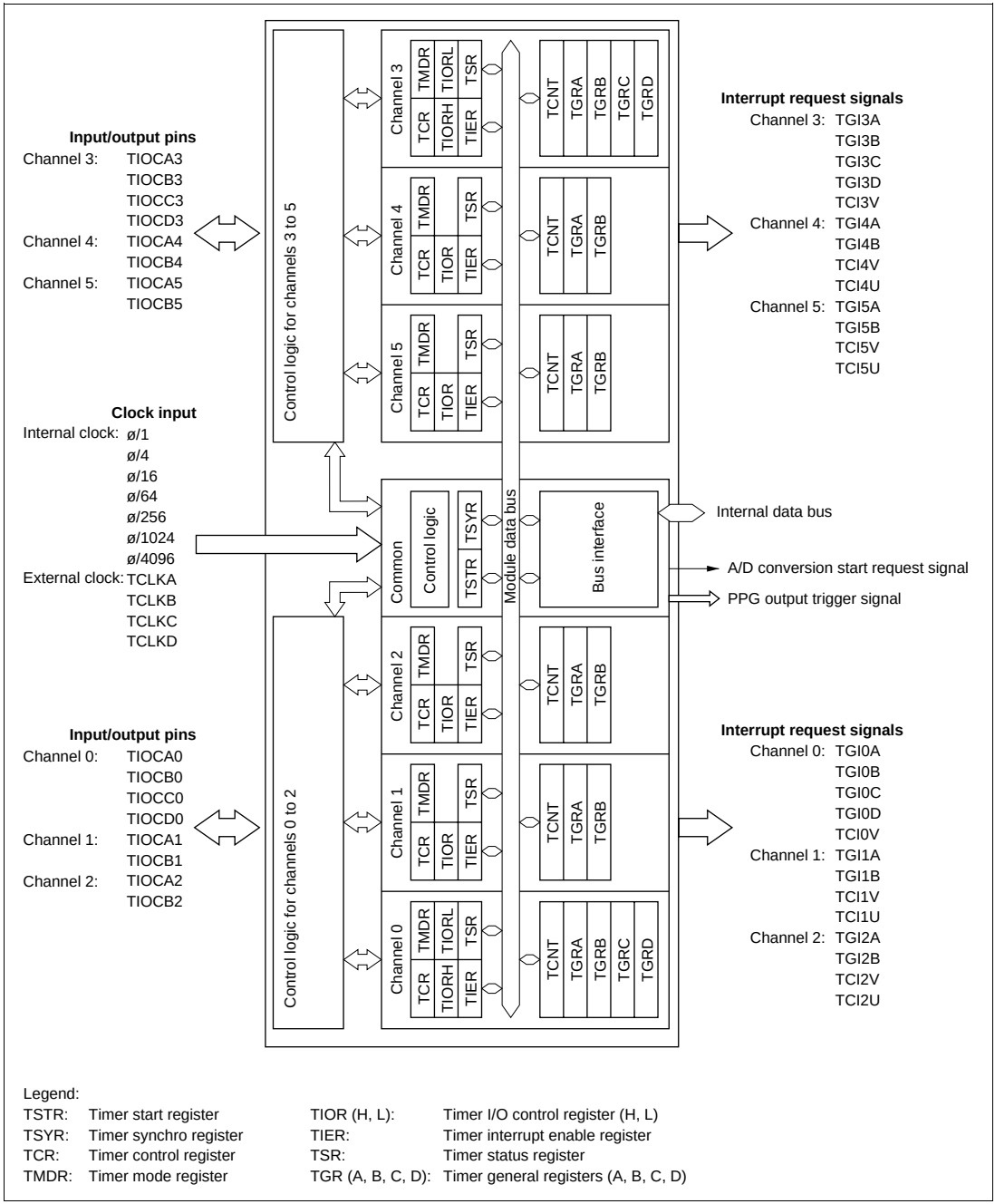
The TPU can perform PWM output, pulse width measurement, and two-phase encoder processing, and can activate the data transfer controller (DTC) and DMA controller (DMAC). It can also generate a programmable pulse generator (PPG) output trigger and A/D converter start trigger.

Features

- Maximum 16-pulse input/output
  - — A total of 16 timer general registers (TGRs) are provided (four each for channels 0 and 3, and two each for channels 1, 2, 4, and 5), each of which can be set independently as an output compare/input capture register
  - — For channels ø and 3, TGRC and TGRD can be used as buffer registers
- Selection of 8 counter input clocks for each channel
  - — Internal clocks: ø, ø/4, ø/16, ø/64, ø/256, ø/1024, ø/4096
  - — External clocks: TCLKA, TCLKB, TCLKC, TCLKD
- The following operations can be set for each channel:
  - — Waveform output at compare match: Selection of 0, 1, or toggle output
  - — Input capture function: Selection of rising edge, falling edge, or both edge detection
  - — Counter clear operation: Counter clearing possible by compare match or input capture
  - — Synchronous operation:
    - Multiple timer counters (TCNT) can be written to simultaneously
    - Simultaneous clearing by compare match and input capture possible
    - Register simultaneous input/output possible by counter synchronous operation
  - — PWM mode:
    - Any PWM output duty can be set
    - Maximum of 15-phase PWM output possible by combination with synchronous operation
- Buffer operation settable for channels 0 and 3
  - — Input capture register double-buffering possible
  - — Automatic rewriting of output compare register possible
- Phase counting mode settable independently for each of channels 1, 2, 4, and 5
  - — Two-phase encoder pulse up/down-count possible
- Cascaded connection operation
  - — Channel 2 (channel 5) input clock operates as 32-bit counter by setting channel 1 (channel 4) overflow/underflow
- Fast access via internal 16-bit bus
  - — Fast access is possible via a 16-bit bus interface

- 26 interrupt sources
  - For channels 0 and 3, four compare match/input capture dual-function interrupts and one overflow interrupt can be requested independently
  - For channels 1, 2, 4, and 5, two compare match/input capture dual-function interrupts, one overflow interrupt, and one underflow interrupt can be requested independently
- Automatic transfer of register data
  - Block transfer, 1-word data transfer, and 1-byte data transfer possible by data transfer controller (DTC) or DMA controller (DMAC) activation
- Programmable pulse generator (PPG) output trigger can be generated
  - Channel 0 to 3 compare match/input capture signals can be used as PPG output trigger
- A/D converter conversion start trigger can be generated
  - Channel 0 to 5 compare match A/input capture A signals can be used as A/D converter conversion start trigger
- Module stop mode can be set
  - As the initial setting, TPU operation is halted. Register access is enabled by exiting module stop mode.

Block Diagram of TPU



Legend:

TSTR:	Timer start register	TIOR (H, L):	Timer I/O control register (H, L)
TSYR:	Timer synchro register	TIER:	Timer interrupt enable register
TCR:	Timer control register	TSR:	Timer status register
TMDR:	Timer mode register	TGR (A, B, C, D):	Timer general registers (A, B, C, D)

- List of TPU Interrupts

Channel	Interrupt Source	Description	DMAC Activation	DTC Activation	Priority
0	TGI0A	TGR0A input capture/compare match	Possible	Possible	High
	TGI0B	TGR0B input capture/compare match	Not possible	Possible	
	TGI0C	TGR0C input capture/compare match	Not possible	Possible	
	TGI0D	TGR0D input capture/compare match	Not possible	Possible	
	TCI0V	TCNT0 overflow	Not possible	Not possible	
1	TGI1A	TGR1A input capture/compare match	Possible	Possible	
	TGI1B	TGR1B input capture/compare match	Not possible	Possible	
	TCI1V	TCNT1 overflow	Not possible	Not possible	
	TCI1U	TCNT1 underflow	Not possible	Not possible	
2	TGI2A	TGR2A input capture/compare match	Possible	Possible	
	TGI2B	TGR2B input capture/compare match	Not possible	Possible	
	TCI2V	TCNT2 overflow	Not possible	Not possible	
	TCI2U	TCNT2 underflow	Not possible	Not possible	
3	TGI3A	TGR3A input capture/compare match	Possible	Possible	
	TGI3B	TGR3B input capture/compare match	Not possible	Possible	
	TGI3C	TGR3C input capture/compare match	Not possible	Possible	
	TGI3D	TGR3D input capture/compare match	Not possible	Possible	
	TCI3V	TCNT3 overflow	Not possible	Not possible	
4	TGI4A	TGR4A input capture/compare match	Possible	Possible	
	TGI4B	TGR4B input capture/compare match	Not possible	Possible	
	TCI4V	TCNT4 overflow	Not possible	Not possible	
	TCI4U	TCNT4 underflow	Not possible	Not possible	
5	TGI5A	TGR5A input capture/compare match	Possible	Possible	
	TGI5B	TGR5B input capture/compare match	Not possible	Possible	
	TCI5V	TCNT5 overflow	Not possible	Not possible	
	TCI5U	TCNT5 underflow	Not possible	Not possible	Low

Note: This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

Operation

Each channel has a TCNT and TGR register. TCNT performs up-counting, and is also capable of free-running operation, cyclic counting, and external event counting. Each TGR can be used as an input capture register or output compare register.

- Buffer Operation
  - — When TGR is an output compare register

    When a compare match occurs, the value in the buffer register for the relevant channel is transferred to TGR.
  - — When TGR is an input capture register

    When input capture occurs, the value in TCNT is transferred to TGR and the value previously held in TGR is transferred to the buffer register.

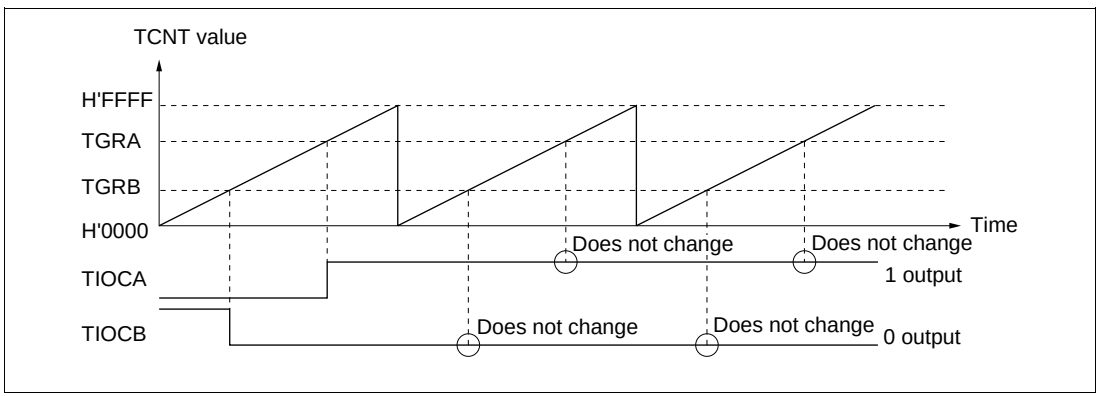
Waveform Output by Compare-Match

0, 1, or toggle output can be selected.

- Example of 0 Output/1 Output Operation

  In this example, TCNT has been designated as a free-running counter, and settings have been made so that 0 is output by compare match A, and 1 is output by compare match B.

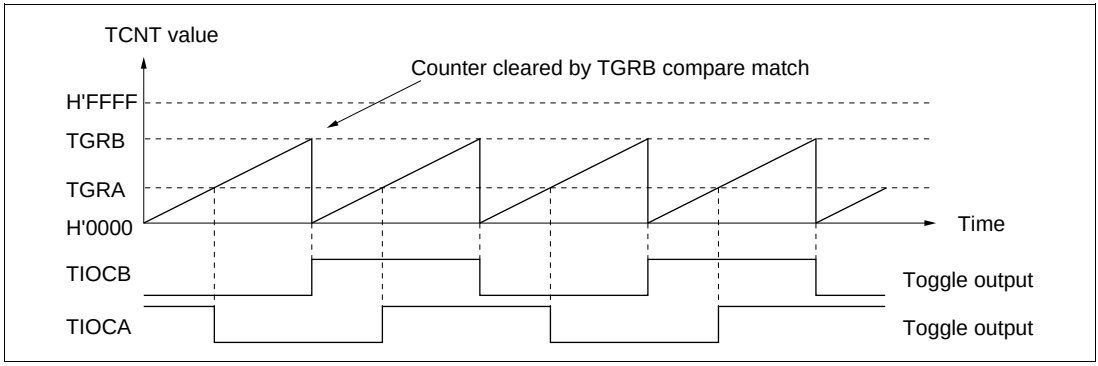
Example of 0 Output/1 Output Operation



- Example of Toggle Output

  In this example, settings have been made so that TCNT counter clearing is performed by compare match B, and output is toggled by both by compare match A and compare match B.

Example of Toggle Output Operation



PWM Modes

In PWM mode, PWM waveforms are output from the output pins. There are two PWM modes—PWM mode 1 with a maximum of 8-phase pulse output, and PWM mode 2 with a maximum of 15-phase pulse output.

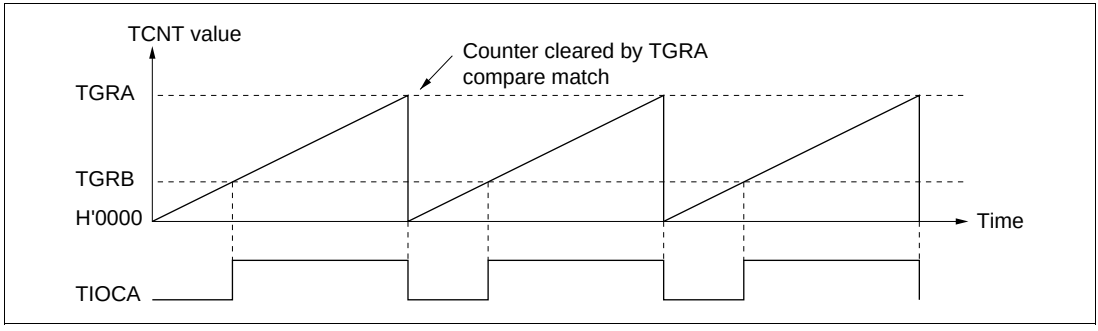
PWM Mode 1: PWM output is generated by pairing TGRA with TGRB and TGRC with TGRD.

In PWM mode 1, a maximum 8-phase PWM output is possible.

- Example of operation in PWM mode 1

  In this example, TGRA compare match is set as the TCNT clearing source, 0 is set for the TGRA initial output value and output value, and 1 is set as the TGRB output value. In this case, the value set in TGRA is the cycle, and the value set in TGRB is the duty.

Operation in PWM Mode 1

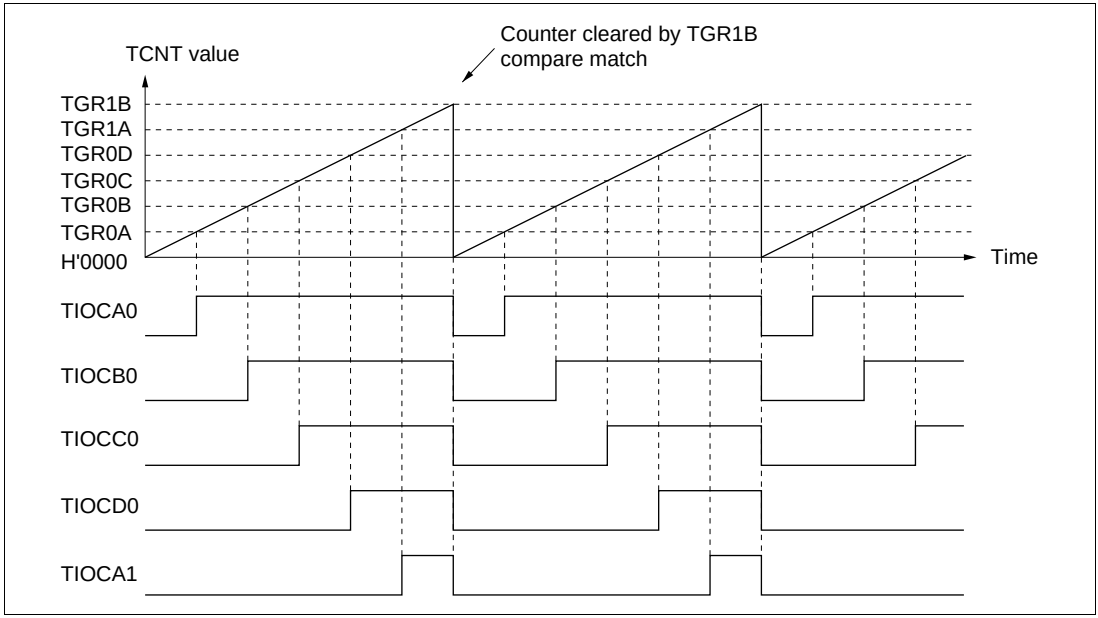


PWM Mode 2: PWM output is generated using one TGR register as the cycle register and the others as duty registers. In PWM mode 2, a maximum 15-phase PWM output is possible by combined use with synchronous operation.

- Example of operation in PWM mode 2

  In this example, synchronous operation is designated for channels 0 and 1, TGR1B compare match is set as the TCNT clearing source, and 0 is set for the initial output value and 1 for the output value of the other TGR registers, to output a 5-phase PWM waveform. In this case, the value set in TGR1B is the cycle, and the value set in the other TGR registers is the duty.

Operation in PWM Mode 2



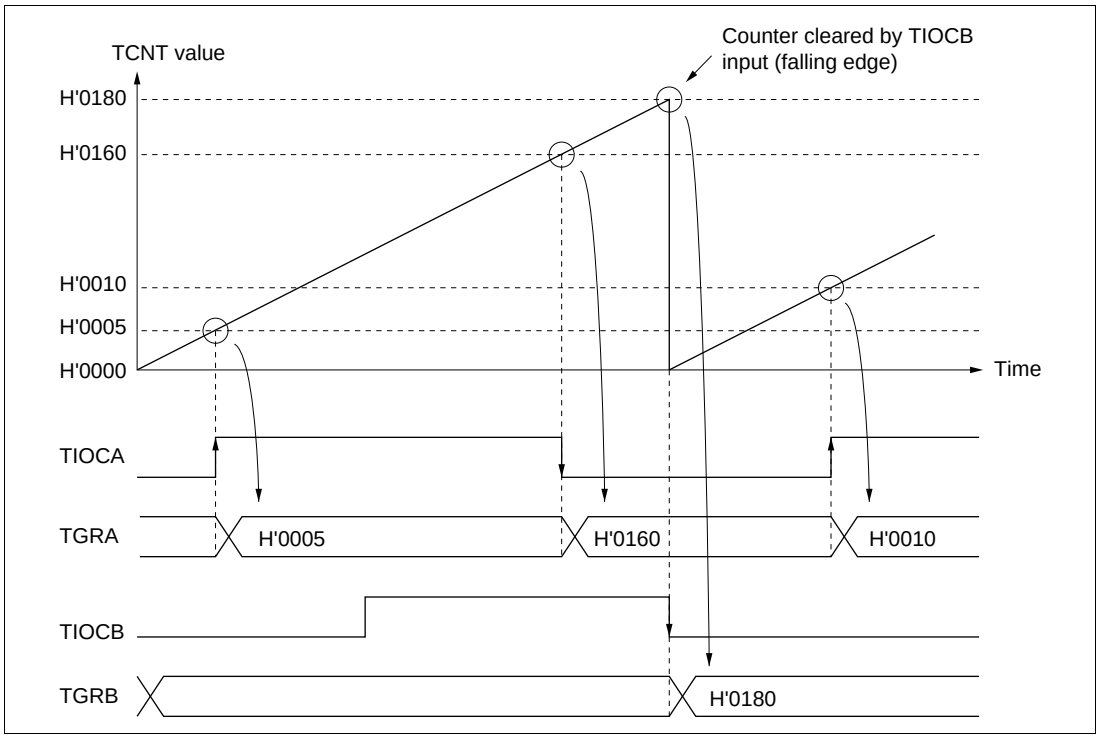
Input Capture Operation

The TCNT value can be transferred to TGR on detection of the TIOC pin input edge.

Rising edge, falling edge, or both edges can be selected as the input edge.

- Example of input capture operation

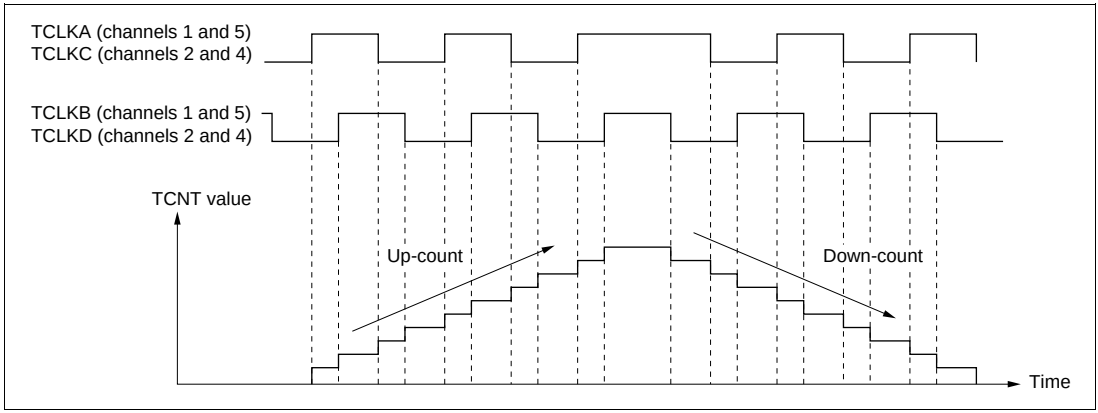
  In this example, both rising and falling edges have been selected as the TIOCA pin input edge, falling edge has been selected as the TIOCB pin input edge, and counter clearing by TGRB input capture has been designated for TCNT.



Phase Counting Mode

In phase counting mode, the phase difference between two external clock inputs is detected and TCNT operates as an up/down-counter. There are four modes (phase counting modes 1 to 4) with different setting conditions. These modes can be set for channels 1, 2, 4, and 5.

- Example of Operation in Phase Counting Mode 1



- Up/Down-Count Conditions in Phase Counting Mode

TCLKA (Channels 1 and 5) TCLKC (Channels 2 and 4)	TCLKB (Channels 1 and 5) TCLKD (Channels 2 and 4)	Phase Counting Mode 1	2	3	4
High level	↑	Up-count	—	—	Up-count
Low level	↓				
↑	Low level				—
↓	High level		Up-count	Up-count	
High level	↓	Down-count	—	Down-count	Down-count
Low level	↑			—	
↑	High level				—
↓	Low level		Down-count		

Legend
↑ : Rising edge
↓ : Falling edge
— : Don't care

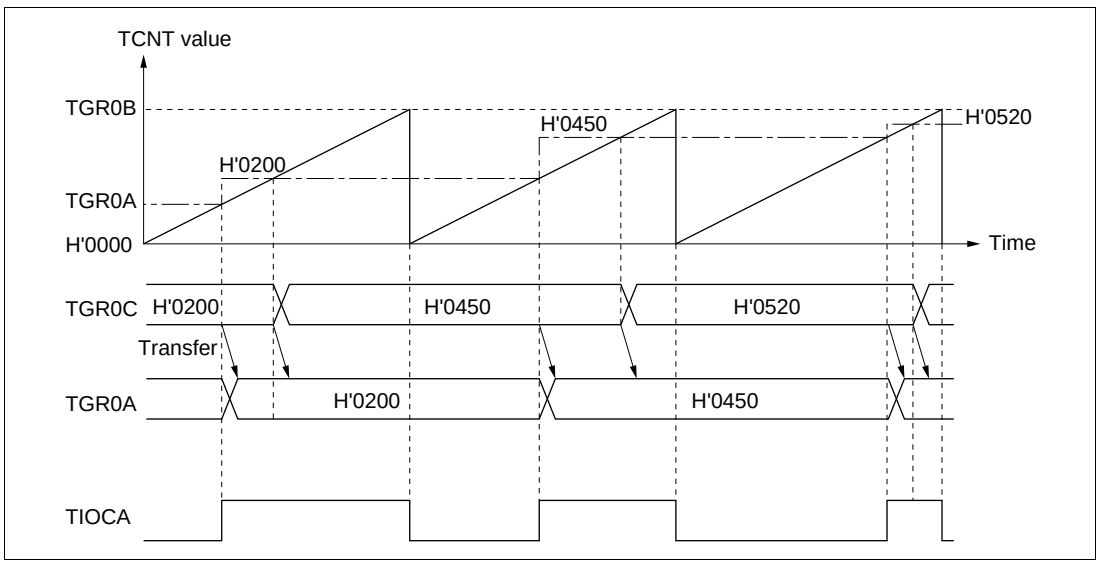
Buffer Operation

Buffer operation, provided for channels 0 and 3, enables TGRC and TGRD to be used as buffer registers.

- Example of buffer operation (1) (When TGR is an output compare register)

  In this example, PWM mode 1 has been designated for channel 0, and buffer operation has been designated for TGRA and TGRC. The settings used are TCNT clearing by a compare match B, 1 output at compare match A, and 0 output at compare match B. When a compare match A occurs, the output is changed and the value in buffer register TGRC is simultaneously transferred to timer general register TGRA.

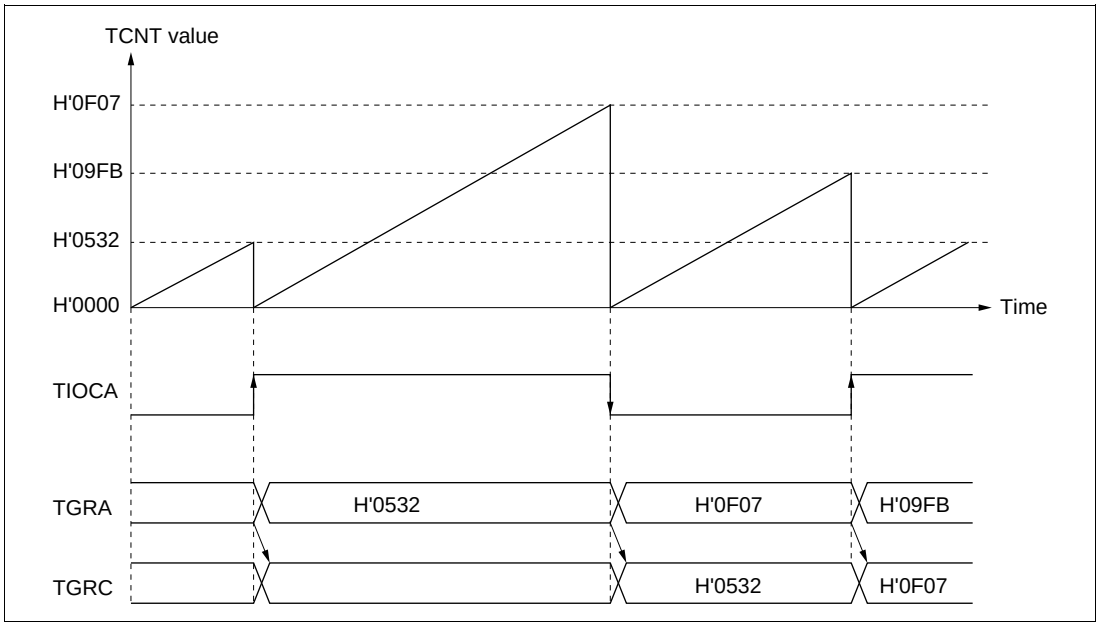
Example of Buffer Operation (1) (When TGR is an Output Compare Register)



- Example of buffer operation (2) (When TGR is an input capture register)

  In this example, TGRA has been designated as an input capture register, and buffer operation has been designated for TGRA and TGRC. Counter clearing by TGRA input capture has been set for TCNT, and detection of both rising and falling edges has been selected for the TIOCA pin. When the TCNT value is stored in TGRA upon occurrence of input capture A, the value previously stored in TGRA is simultaneously transferred to TGRC.

Example of Buffer Operation (2) (When TGR is an Input Capture Register)



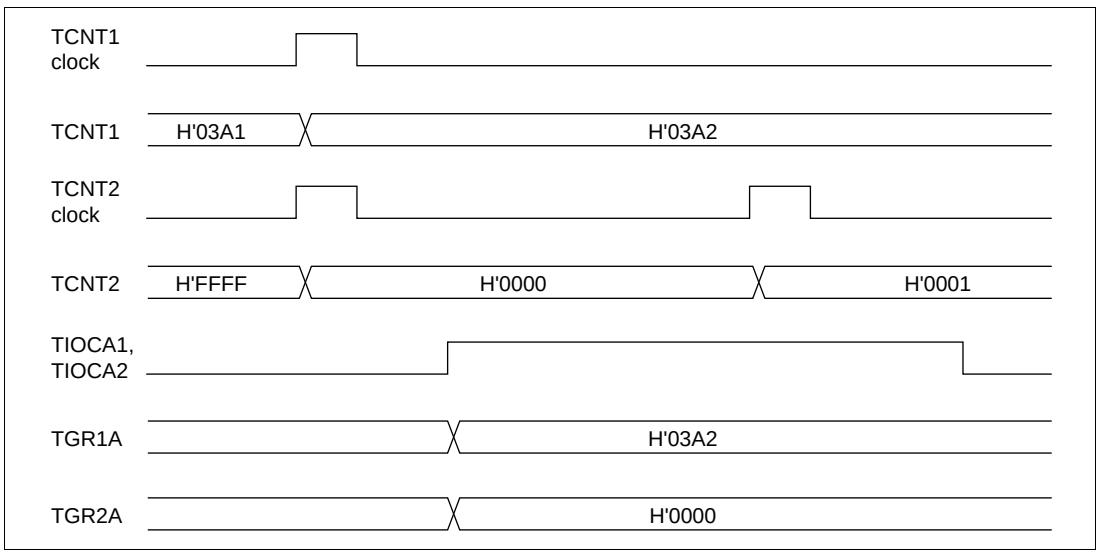
Cascading

In cascaded operation, two 16-bit counters for different channels are used together as a 32-bit counter. Channels 1 and 2, and channels 4 and 5, can be cascaded.

- Example of cascaded operation

  In this example, counting upon TCNT2 overflow/underflow has been set for TCNT1, TGR1A and TGR2A have been designated as input capture registers, and TIOC pin rising edge detection has been selected. When a rising edge is input to the TIOCA1 and TIOCA2 pins simultaneously, the upper 16 bits of the 32-bit data are transferred to TGR1A, and the lower 16 bits to TGR2A.

Example of Cascaded Operation (32-bit Input Capture Operation)



Synchronous Operation

When synchronous operation is designated for a channel, TCNT for that channel performs synchronous presetting and clearing. That is, when TCNT for a channel designated for synchronous operation is rewritten, the TCNT counters for the other channels are also rewritten at the same time. When any clearing condition occurs, the TCNT counters for the other channels are also cleared simultaneously.

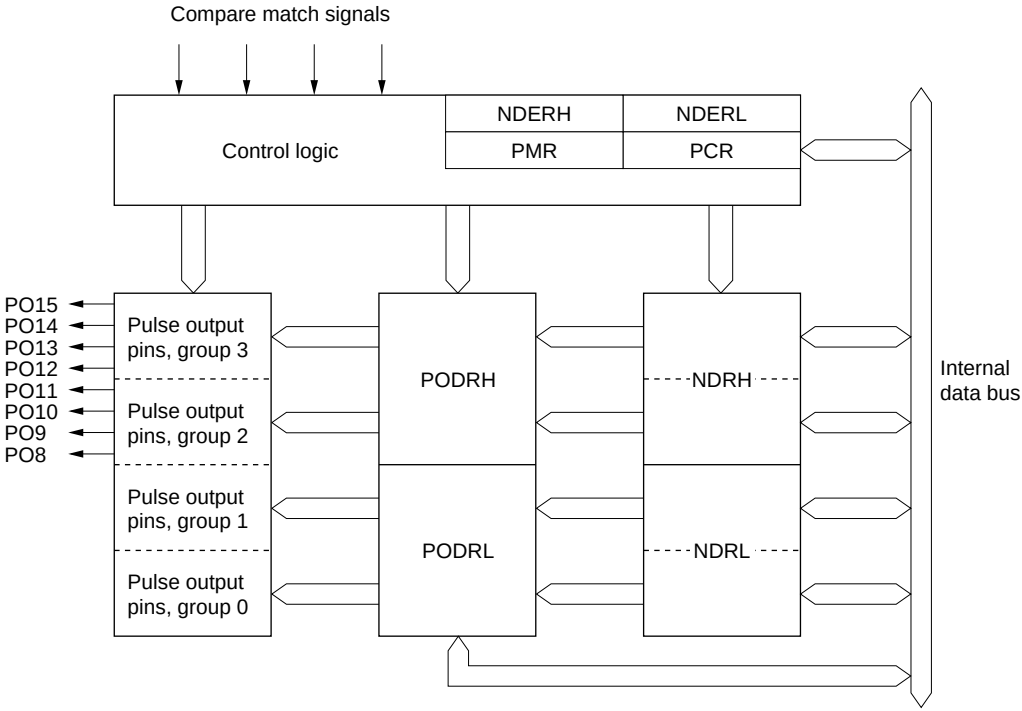
3.6 Programmable Pulse Generator (PPG)

The programmable pulse generator (PPG) can handle up to 8 outputs simultaneously, using a signal from the 16-bit timer-pulse unit (TPU) as its input.

Features

- 8-bit output data
  - — Maximum 8-bit data can be output, and pulse output can be enabled on a bit-by-bit basis
- Two output groups
  - — Output trigger signals can be selected in 4-bit groups to provide a maximum of two 4-bit outputs
- Selectable output trigger signals
  - — Output trigger signals can be selected for each group from the compare match signals of four TPU channels
- Non-overlap operation
  - — A non-overlap interval can be set between pulse outputs
- Can operate together with the data transfer controller (DTC) and DMA controller (DMAC)
  - — The compare match signals selected as output trigger signals can activate the DTC or DMAC for sequential output of data without CPU intervention
- Inverted output can be specified
  - — Inverted data can be output for each group
- Module stop mode can be set
  - — As the initial setting, PPG operation is halted. Register access is enabled by exiting module stop mode

Block Diagram of PPG



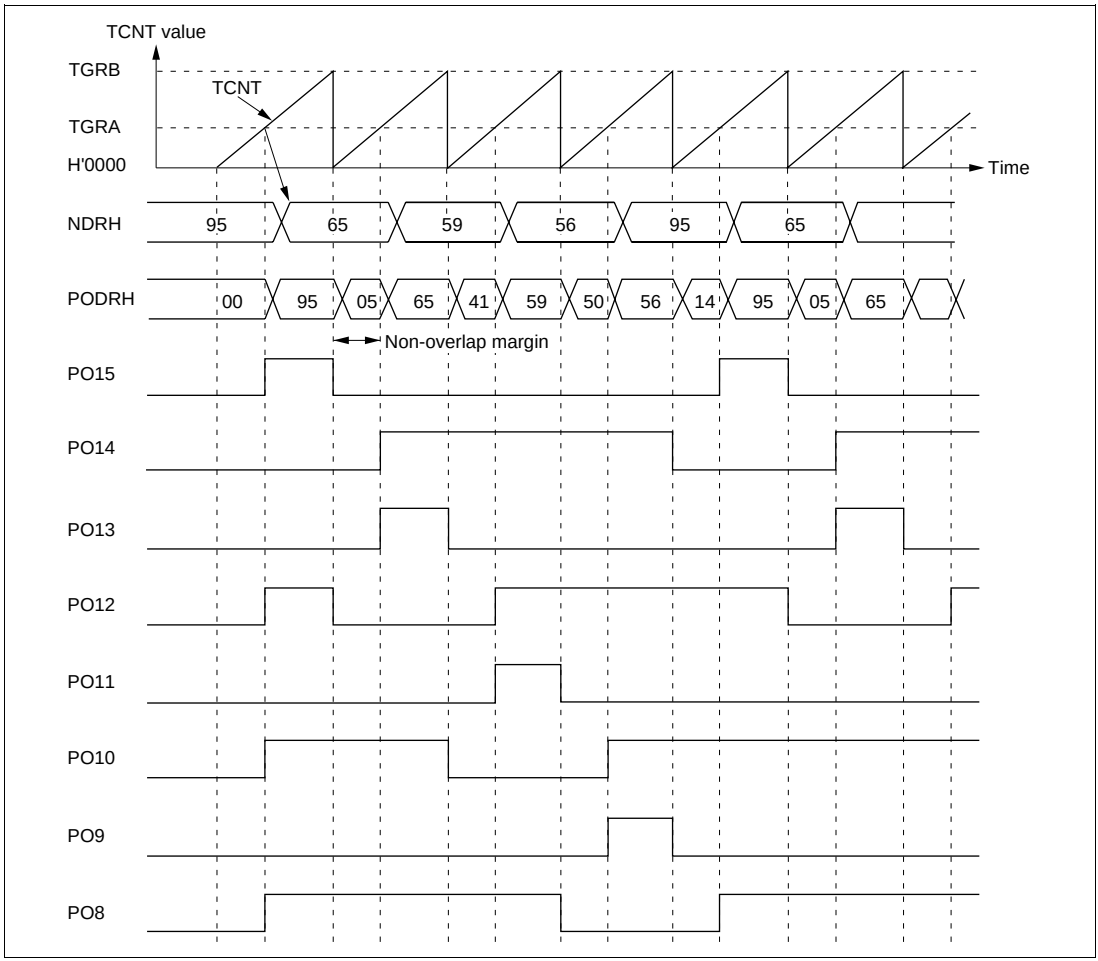
Legend:
PMR: PPG output mode register
PCR: PPG output control register
NDERH: Next data enable register H
NDERL: Next data enable register L
NDRH: Next data register H
NDRL: Next data register L
PODRH: Output data register H
PODRL: Output data register L

Example of Four-Phase Complementary Non-Overlapping Output

In this example, pulse output is used for four-phase complementary non-overlapping pulse output.

When a TGRB compare match occurs, outputs change from 1 to 0. When a TGRA compare match occurs, outputs change from 0 to 1. Set the non-overlap margin in the TPU TGRA for which the output trigger is selected, and set the cycle in TGRB.

If the DTC or DMAC is set for activation by a TGIA interrupt, pulse output can be performed without imposing a load on the CPU.



Non-Overlapping Pulse Output Example
(Four-Phase Complementary Non-Overlapping Output)

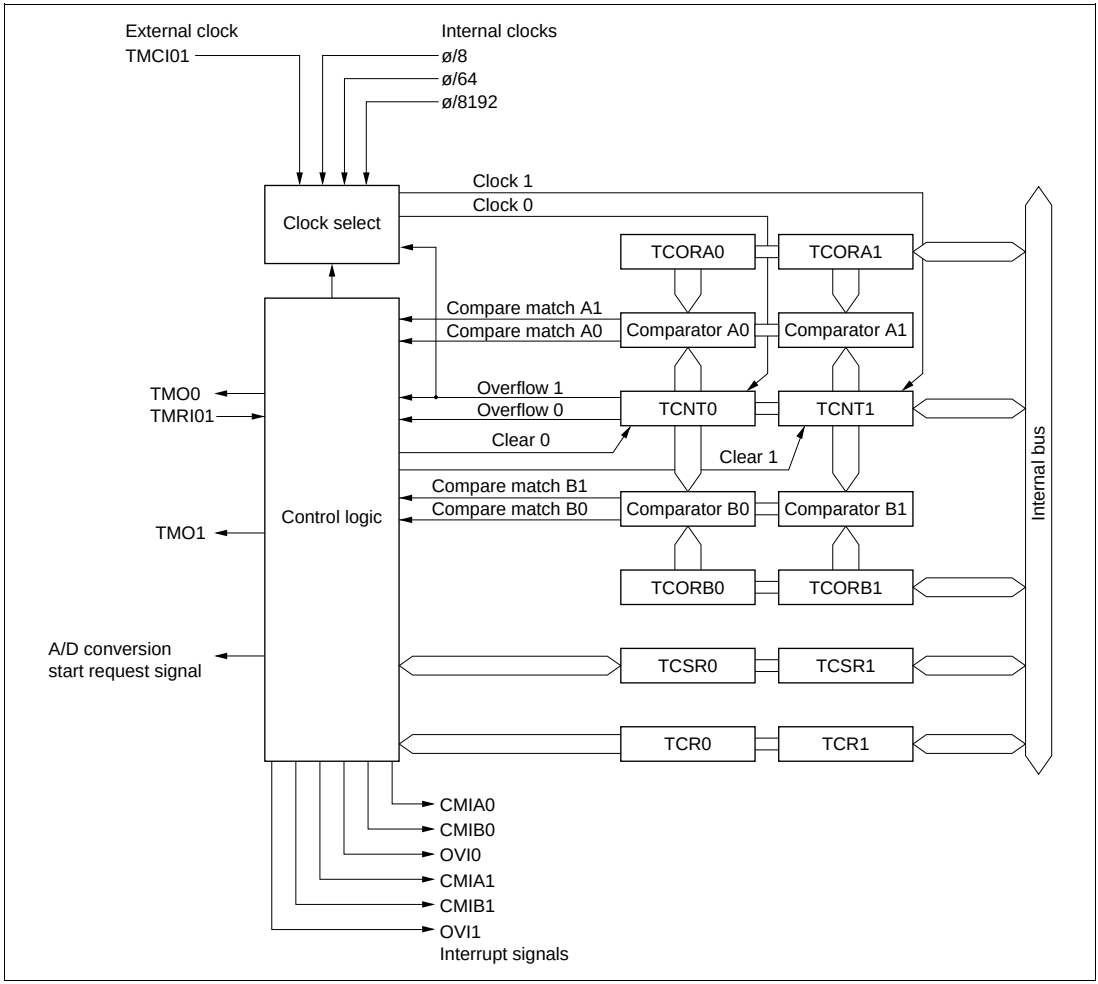
3.7 8-Bit Timer (TMR)

This series includes an 8-bit timer with four channels (TMR0 to TMR3) based on an 8-bit counter. The 8-bit timer can be used for a variety of applications as a multifunctional timer, including pulse output with an arbitrary duty cycle.

Features

- Selection of four input clock sources
  - — The clock source can be selected from three internal clocks (ø/8, ø/64, or ø/8192) or an external clock (external event counting is possible).
- Counter clearing specification
  - — The counters can be cleared on compare match A or B, or by an external reset signal.
- Timer output controlled by combination of two compare match signals
  - — The combination of two independent compare match signals provides for various applications, including pulse output or PWM output with any duty cycle.
- Provision for cascading of two channels
  - — Operation as a 16-bit timer is possible, using channel 0 for the upper 8 bits and channel 1 for the lower 8 bits (16-bit count mode).
  - — Channel 1 can be used to count channel 0 compare matches (compare match count mode).
- Three interrupt sources for each channel
  - — There are two compare match sources and one overflow source, capable of independent requests.
- A/D converter conversion start trigger can be generated
  - — Channel 0 compare match A signal can be used as an A/D converter conversion start trigger.
- Module stop mode can be set
  - — As the initial setting, 8-bit timer operation is halted. Register access is enabled by exiting module stop mode.

Block Diagram



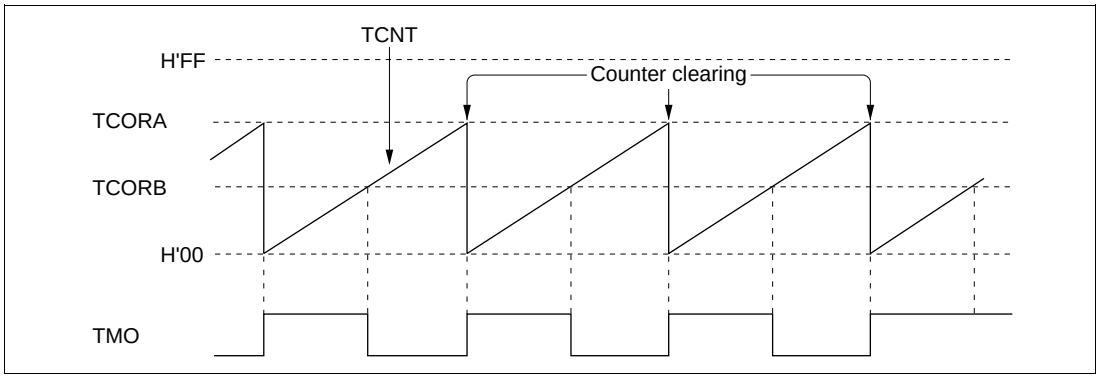
Interrupt Source and Data Transfer Controller (DTC) Activation

Channel	Interrupt Source	Description	DTC Activation	Priority*
0	CMIA0	Interrupt by CMFA	Possible	High
	CMIB0	Interrupt by CMFB	Possible	
	OVI0	Interrupt by OVF	Not possible	
1	CMIA1	Interrupt by CMFA	Possible	
	CMIB1	Interrupt by CMFB	Possible	
	OVI1	Interrupt by OVF	Not possible	
2	CMIA2	Interrupt by CMFA	Possible	
	CMIB2	Interrupt by CMFB	Possible	
	OVI2	Interrupt by OVF	Not possible	
3	CMIA3	Interrupt by CMFA	Possible	
	CMIB3	Interrupt by CMFB	Possible	
	OVI3	Interrupt by OVF	Not possible	Low

Note: * This table shows the initial state immediately after a reset. The relative channel priorities can be changed by the interrupt controller.

Example of Pulse Output

TCR is used to set counter clearing by a TCORA compare match. The cycle is set in TCORA, and the duty in TCORB. The pulses shown below can be output continuously without software intervention.



Example of Pulse Output

3.8 14-Bit PWM Timer (PWM)

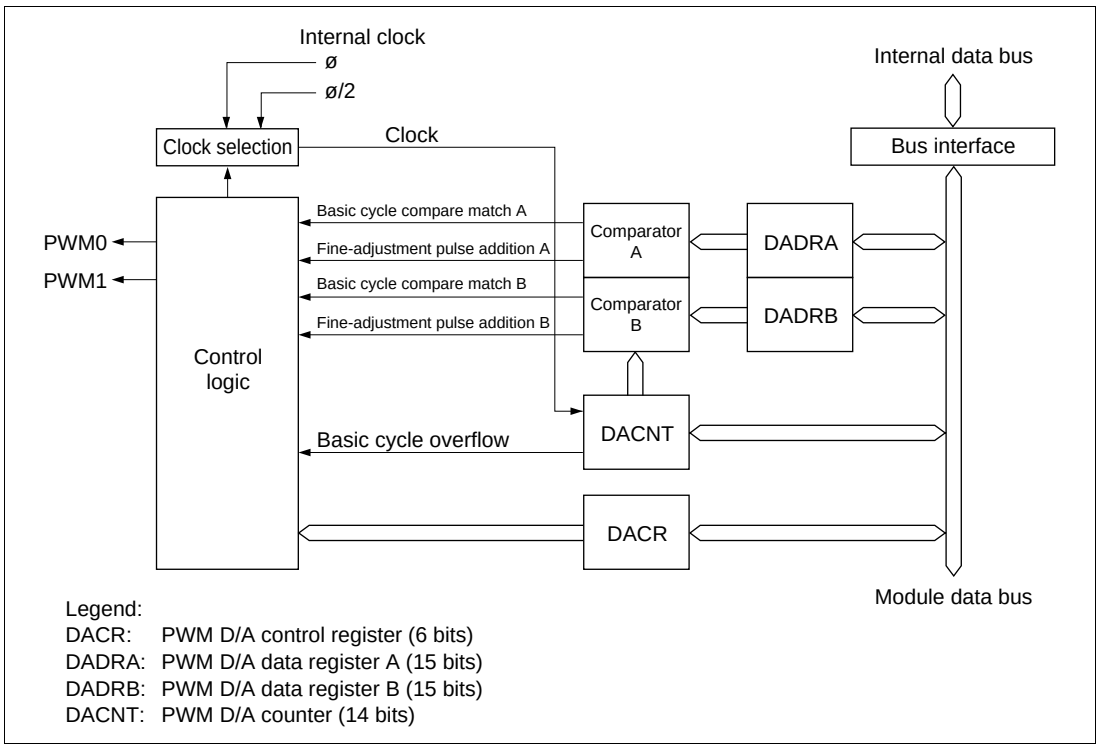
This series has a four-channel on-chip PWM (pulse width modulator). Connection of an external low-pass filter enables this module to be used as a 14-bit D/A converter.

Two PWM channels share the same counter (DACNT) and control register (DACR).

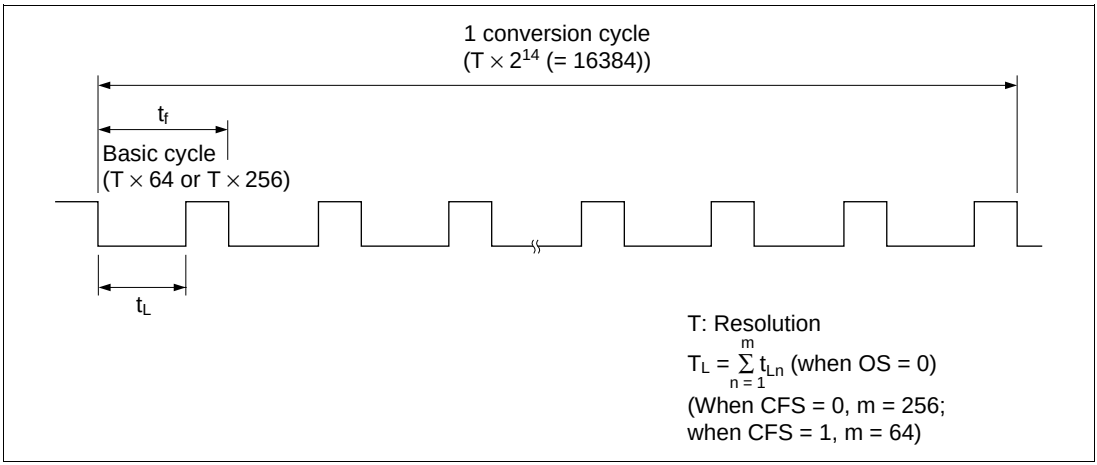
Features

- Low-ripple pulse division method
- Selection of two resolutions and two basic cycles
  - The system clock cycle or twice the system clock cycle can be selected for the resolution
  - T × 64 or T × 256 (where T = resolution) can be selected as the basic cycle
- Selection of four operating speeds
  - Any of four operating clocks can be selected (two basic cycles x two resolutions)

Block Diagram of PWM (D/A) (For Two Channels)



PWM (D/A) Operation



3.9 Watchdog Timer (WDT)

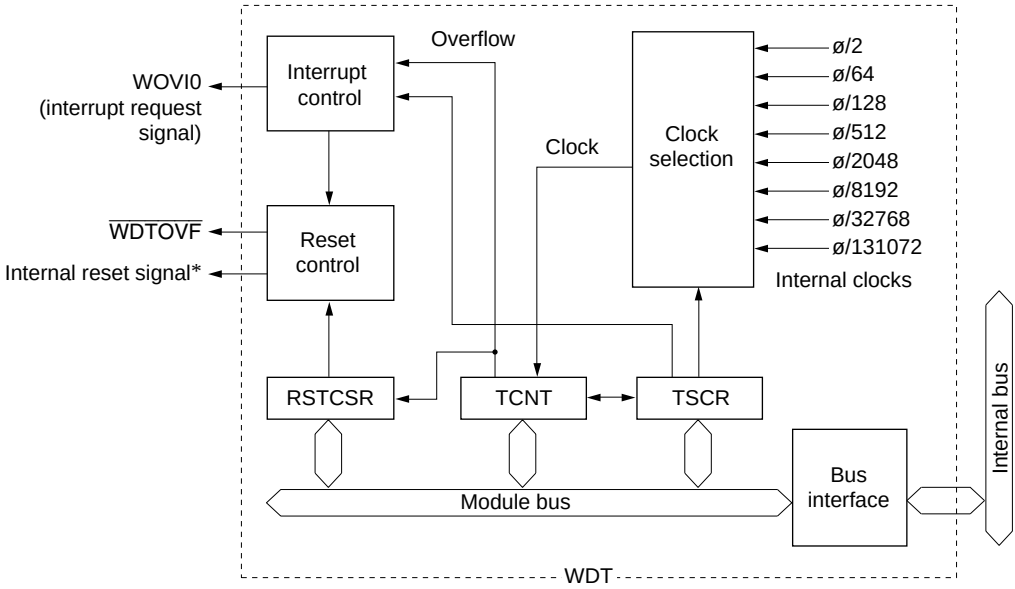
This series has a two-channel on-chip watchdog timer (WDT) for monitoring system operation. When this watchdog function is not needed, the WDT can be used as an interval timer.

When the subclock is selected as the input clock, the WDT can be used as a realtime clock timer.

Features

- Switchable between watchdog timer mode and interval timer mode
- $\overline{\text{WDTOVF}}$ output when in watchdog timer mode
  - — If the counter overflows, the WDT outputs the $\overline{\text{WDTOVF}}$ signal externally. It is possible to select whether the chip is internally reset or an NMI interrupt is generated at the same time. A power-on reset or manual reset can be selected as this internal reset.
- Interrupt generation when in interval timer mode
  - — If the counter overflows, the WDT generates an interval timer interrupt.
- Choice of eight counter input clocks for WDT0 and 16 for WDT1
  - — The maximum WDT interval is the subclock cycle $\times$ 131072 $\times$ 256.
  - — The subclock can be selected for the WDT1 input counter. When the subclock is selected, the maximum interval is the subclock cycle $\times$ 256 $\times$ 256.
- Selected clock can be output from the BUZZ output pin (WDT1)

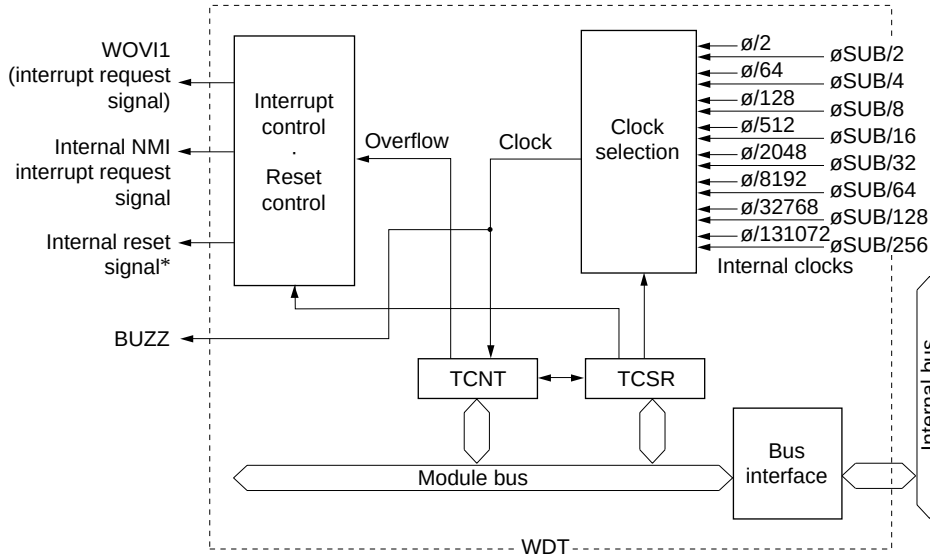
Block Diagram of WDT0



Legend:
TCSR: Timer control/status register
TCNT: Timer counter
RSTCSR: Reset control/status register

Note: * An internal reset signal can be generated by means of a register setting. Either a power-on or manual reset can be selected.

Block Diagram of WDT1



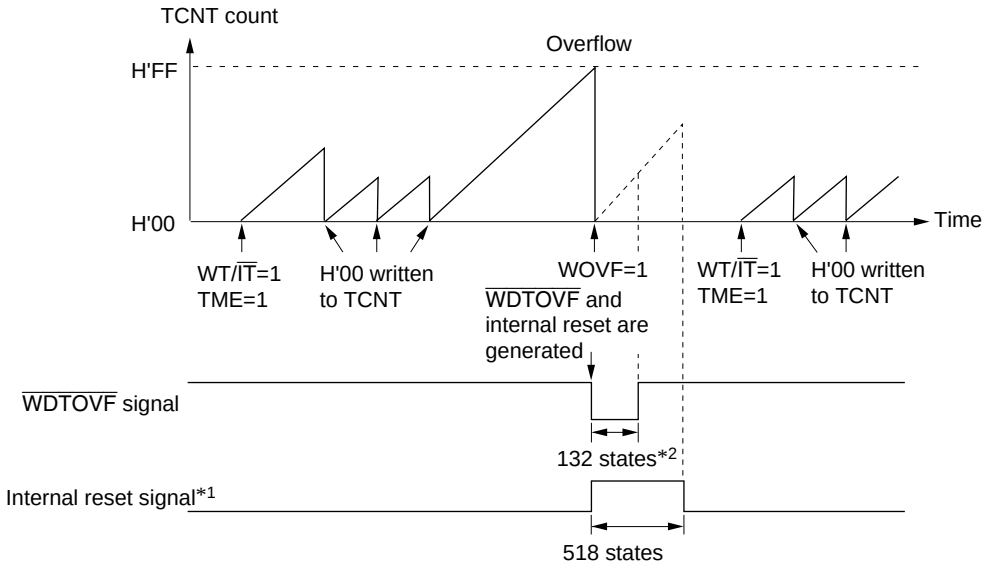
Legend:
TCSR: Timer control/status register
TCNT: Timer counter

Note: * An internal reset signal can be generated by means of a register setting. The reset is a power-on reset.

Watchdog Timer Operation

The examples below show this timer used as a watchdog timer. The timer counter (TCNT) starts counting up using the specified clock.

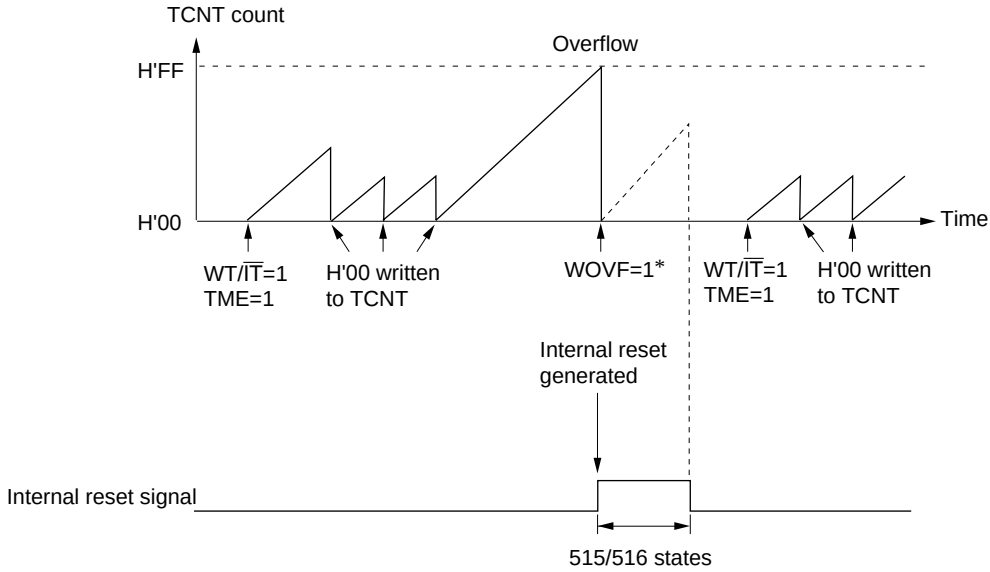
- WDT0 Watchdog Timer Operation



Legend:
WT/$\overline{IT}$: Timer mode select bit
TME: Timer enable bit

Notes: 1. The internal reset signal is generated only if the RSTE bit is set to 1.
2. 130 states when the RSTE bit is cleared to 0.

- WDT1 Watchdog Timer Operation

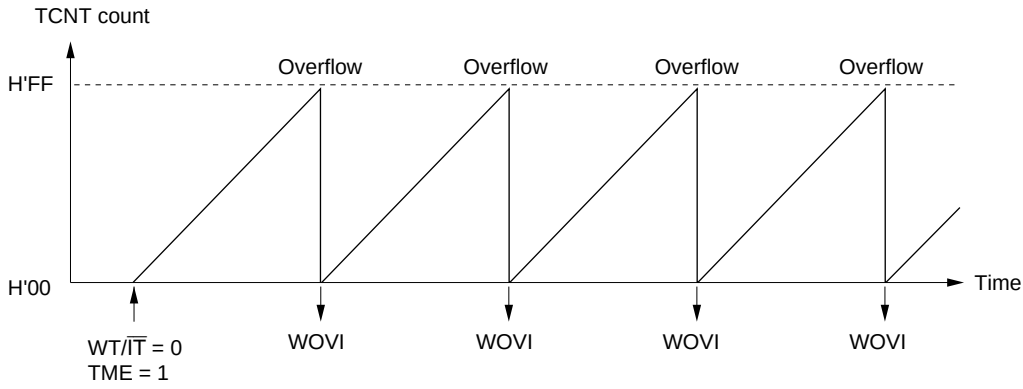


Legend:
WT/$\overline{IT}$: Timer mode select bit
TME: Timer enable bit

Note: * After WOVF is set to 1, it is cleared to 0 by an internal reset.

Interval Timer Operation

An example of the use of the WDT as an interval timer is shown here. The timer counter (TCNT) starts counting up on the specified clock, and an interval timer interrupt (WOVI) occurs each time TCNT overflows. This function can be used to generate interrupt requests at regular intervals.



Legend:
WOVI: Interval timer interrupt request generation

3.10 Serial Communication Interface (SCI, IrDA)

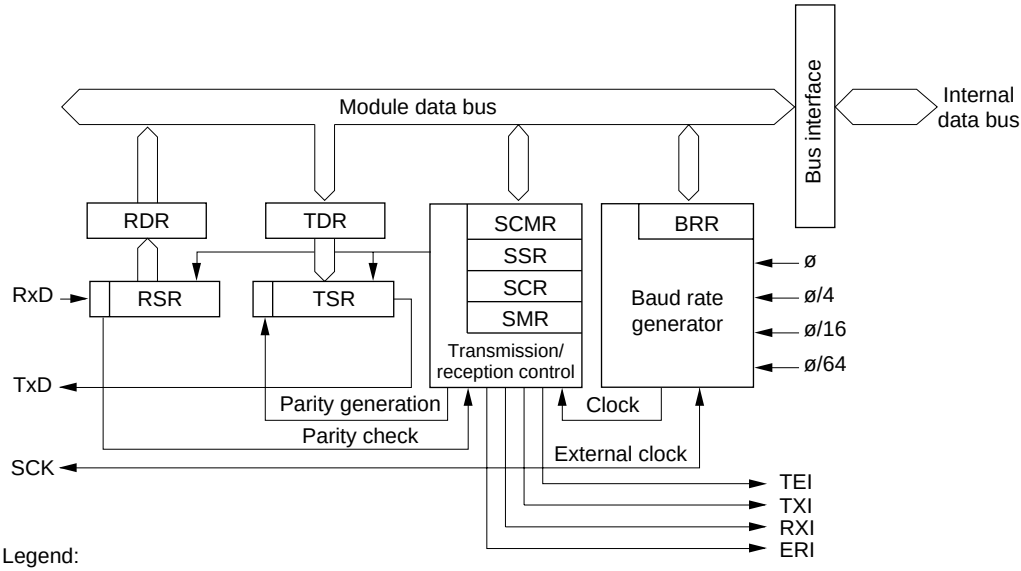
This series has a serial communication interface (SCI) with five independent channels. All five channels have the same functions, and can handle both asynchronous and synchronous serial communication. A function is also provided for serial communication between processors (multiprocessor communication function).

One of the five SCI channels is capable of sending and receiving IrDA communications waveforms (based on IrDA Version 1.0).

Features

- Choice of asynchronous or synchronous serial communication mode
- Full-duplex communication capability
- Data register double-buffering enables continuous transmission/reception
- Internal dedicated baud rate generator allows any bit rate to be selected
- Selection of internal baud rate generator or external clock input (SCK pin) as serial clock source
- Detection of three receive errors
  - — Overrun errors, framing errors, and parity errors can be detected
- Break detection
- Four interrupt sources
  - — Four interrupt sources—transmit data empty, transmission end, receive data full, and receive error—that can issue requests independently
  - — The transmit data empty interrupt and receive data full interrupt can activate the DMA controller (DMAC) or data transfer controller (DTC) to execute data transfer
- Built-in multiprocessor communication function
- Selection of LSB-first or MSB-first transfer
  - — This choice can be made regardless of the communication mode (with the exception of 7-bit data transfer in asynchronous mode)
- Module stop mode can be set
  - — As the initial setting, SCI operation is halted. Register access is enabled by exiting module stop mode.

Block Diagram (One Channel)



Legend:

SCMR:	Smart card mode register
RSR:	Receive shift register
RDR:	Receive data register
TSR:	Transmit shift register
TDR:	Transmit data register
SMR:	Serial mode register
SCR:	Serial control register
SSR:	Serial status register
BRR:	Bit rate register

SCI Interrupt Sources

Channel	Interrupt Source	Description	DTC Activation	DMAC Activation	Priority*
0	ERI	Interrupt due to receive error (ORER, FER, or PER)	Not possible	Not possible	High
	RXI	Interrupt due to receive data full (RDRF)	Possible	Possible	
	TXI	Interrupt due to transmit data empty (TDRE)	Possible	Possible	
	TEI	Interrupt due to transmission end (TEND)	Not possible	Not possible	
1	ERI	Interrupt due to receive error (ORER, FER, or PER)	Not possible	Not possible	
	RXI	Interrupt due to receive data full (RDRF)	Possible	Possible	
	TXI	Interrupt due to transmit data empty (TDRE)	Possible	Possible	
	TEI	Interrupt due to transmission end (TEND)	Not possible	Not possible	
2	ERI	Interrupt due to receive error (ORER, FER, or PER)	Not possible	Not possible	
	RXI	Interrupt due to receive data full (RDRF)	Possible	Not possible	
	TXI	Interrupt due to transmit data empty (TDRE)	Possible	Not possible	
	TEI	Interrupt due to transmission end (TEND)	Not possible	Not possible	
3	ERI	Interrupt due to receive error (ORER, FER, or PER)	Not possible	Not possible	
	RXI	Interrupt due to receive data full (RDRF)	Possible	Not possible	
	TXI	Interrupt due to transmit data empty (TDRE)	Possible	Not possible	
	TEI	Interrupt due to transmission end (TEND)	Not possible	Not possible	Low

Channel	Interrupt Source	Description	DTC Activation	DMAC Activation	Priority*
4	ERI	Interrupt due to receive error (ORER, FER, or PER)	Not possible	Not possible	High
	RXI	Interrupt due to receive data full (RDRF)	Possible	Not possible	
	TXI	Interrupt due to transmit data empty (TDRE)	Possible	Not possible	
	TEI	Interrupt due to transmission end (TEND)	Not possible	Not possible	Low

Note: * This table shows the initial state immediately after a reset. Relative priorities among channels can be changed by means of the interrupt controller.

SCI Asynchronous Communication

Asynchronous mode is a serial communication mode in which synchronization is achieved on a character by character basis, using a start bit and one or two stop bits.

- Twelve serial data transfer formats
  - — Data length: 7 or 8 bits
  - — Stop bit length: 1 or 2 bits
  - — Parity: Even/odd/none
  - — Multiprocessor bit: 1 or 0
- Selection of internal baud rate generator or external clock from SCK pin as clock source
- Transmit/receive clock can be output from SCK pin
- Break detection
  - — A break can be detected by reading the RxD pin level directly in case of a framing error
- Multiprocessor communication capability

Serial Transfer Formats and Frame Lengths in Asynchronous Mode

SMR Settings				Serial Transmit/Receive Format and Frame Length											
CHR	PE	MP	STOP	1	2	3	4	5	6	7	8	9	10	11	12
0	0	0	0	S	8-bit data								STOP		
0	0	0	1	S	8-bit data								STOP	STOP	
0	1	0	0	S	8-bit data								P	STOP	
0	1	0	1	S	8-bit data								P	STOP	STOP
1	0	0	0	S	7-bit data								STOP		
1	0	0	1	S	7-bit data								STOP	STOP	
1	1	0	0	S	7-bit data								P	STOP	
1	1	0	1	S	7-bit data								P	STOP	STOP
0	—	1	0	S	8-bit data								MPB	STOP	
0	—	1	1	S	8-bit data								MPB	STOP	STOP
1	—	1	0	S	7-bit data								MPB	STOP	
1	—	1	1	S	7-bit data								MPB	STOP	STOP

Legend:

S: Start bit

STOP: Stop bit

P: Parity bit

MPB: Multiprocessor bit

Multiprocessor Communication Function

A multiprocessor format, in which a multiprocessor bit is added to the transfer data, can be used for serial communication, enabling data transfer to be performed among a number of processors.

The transmitting station first sends the ID of the receiving station with which it wants to perform serial communication as data with a “1” MPB (multiprocessor bit) added. It then sends transmit data as data with a “0” MPB added.

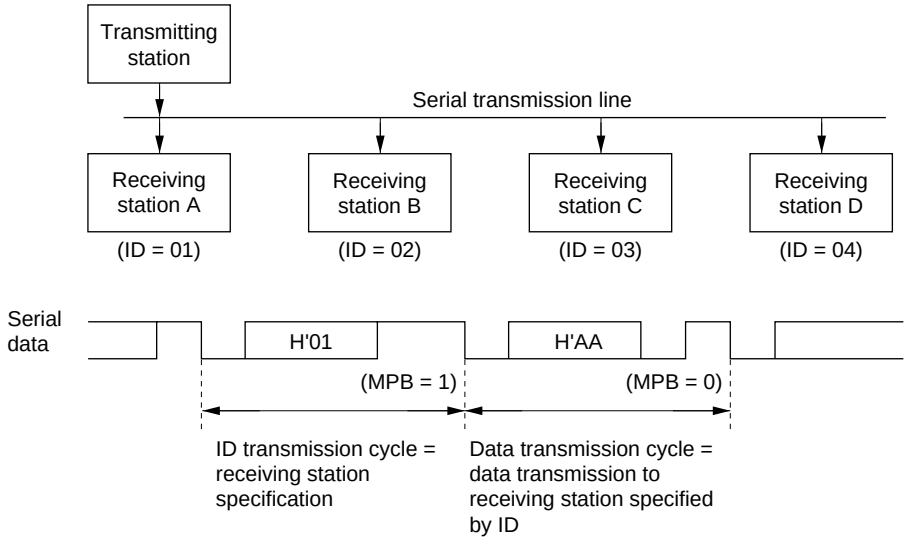
Receiving stations skip data until data with a “1” MPB is received. Each receiving station then compares that data with its own ID. The station whose ID matches then continues with reception, and accepts data. Stations whose ID does not match continue to skip the data until data with a “1” MPB is sent again.

SCI Synchronous Communication

In synchronous mode, data is transmitted or received in synchronization with clock pulses, making it suitable for continuous, high-speed serial communication.

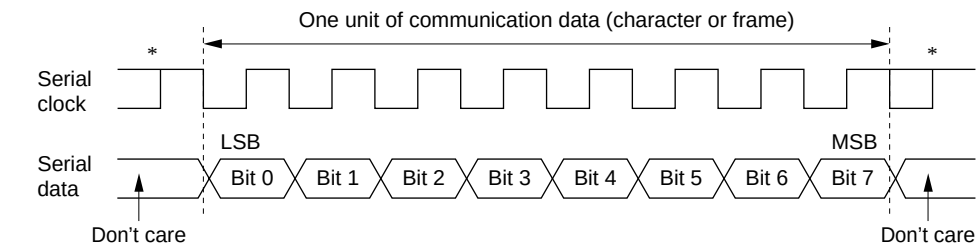
- Data length: 8 bits per character
- Overrun error detection
- Selection of internal baud rate generator or external clock from SCK pin as transmit/receive clock source
- Selection of LSB-first or MSB-first transfer
- Communication is possible with chips provided with a synchronous mode, such as the H8 Series, HD64180, and HD6301

Example of Inter-Processor Communication Using Multiprocessor Format (Transmission of Data H'AA to Receiving Station A)



Legend:
MPB: Multiprocessor bit

Data Format in Synchronous Communication



Note: * High except in continuous transfer

BRR Settings for Various Bit Rates (Synchronous Mode)

Bit Rate (bits/s)	Operating Frequency ø (MHz)													
	ø = 2		ø = 4		ø = 8		ø = 10		ø = 16		ø = 20		ø = 25	
	n	N	n	N	n	N	n	N	n	N	n	N	n	N
110	3	70	—	—										
250	2	124	2	249	3	124	—	—	3	249				
500	1	249	2	124	2	249	—	—	3	124	—	—		
1 k	1	124	1	249	2	124	—	—	2	249	—	—	3	97
2.5 k	0	199	1	99	1	199	1	249	2	99	2	124	2	155
5 k	0	99	0	199	1	99	1	124	1	199	1	249	2	77
10 k	0	49	0	99	0	199	0	249	1	99	1	124	1	155
25 k	0	19	0	39	0	79	0	99	0	159	0	199	0	249
50 k	0	9	0	19	0	39	0	49	0	79	0	99	0	124
100 k	0	4	0	9	0	19	0	24	0	39	0	49	0	62
250 k	0	1	0	3	0	7	0	9	0	15	0	19	0	24
500 k	0	0*	0	1	0	3	0	4	0	7	0	9	—	—
1 M			0	0*	0	1			0	3	0	4	—	—
2.5 M							0	0*			0	1	—	—
5 M											0	0*	—	—

Note: As far as possible, the setting should be made so that the error is no more than 1%.

Legend:

Blank: Cannot be set.

—: Can be set, but there will be a degree of error.

*: Continuous transfer is not possible.

The BRR setting is found from the following formulas.

Asynchronous mode:

$$N = \frac{\phi}{64 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Synchronous mode:

$$N = \frac{\phi}{8 \times 2^{2n-1} \times B} \times 10^6 - 1$$

Where B: Bit rate (bits/s)
N: BRR setting for baud rate generator ($0 \leq N \leq 255$)
ø: Operating frequency (MHz)
n: Baud rate generator input clock (n = 0 to 3)
(See the table below for the relation between n and the clock.)

n	Clock
0	ø
1	ø/4
2	ø/16
3	ø/64

IrDA Operation

When the IrDA function is enabled by means of the IrE bit in the IrDA control register (IrCR), the SCI channel 0 TxD0 and RxD0 signals are subjected to waveform encoding/decoding conforming to IrDA standard version 1.0 (IrTxD and IrRxD pins). This makes it possible to carry out infrared transmission and reception conforming to the IrDA standard version 1.0 system by connecting an infrared transmission/reception transceiver/receiver.

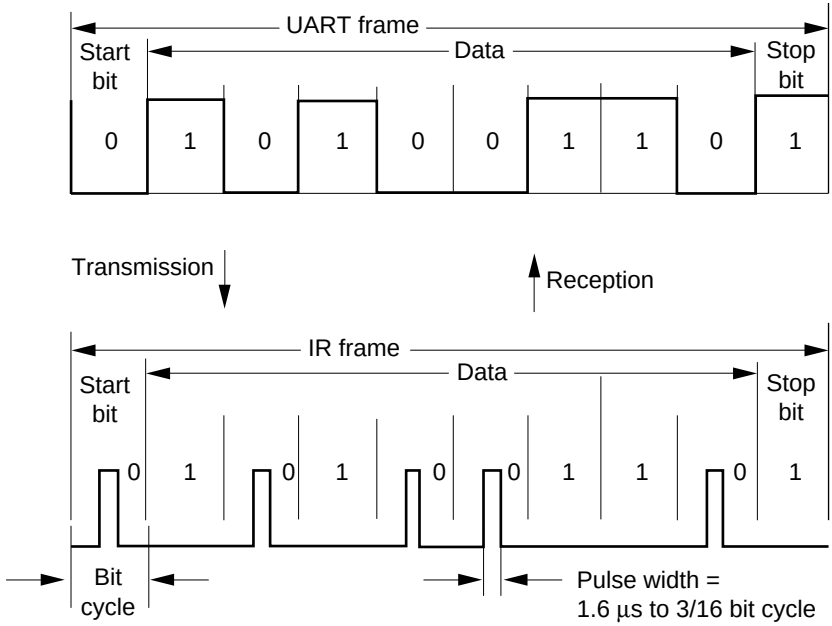
- Transmission

  In transmission, the output signal (UART frame) from the SCI is converted to an IR frame by means of the IrDA interface.

- Reception

  In reception, IR frame data is converted to a UART frame by means of the IrDA interface, and input to the SCI.

- IrDA transmit/receive operations



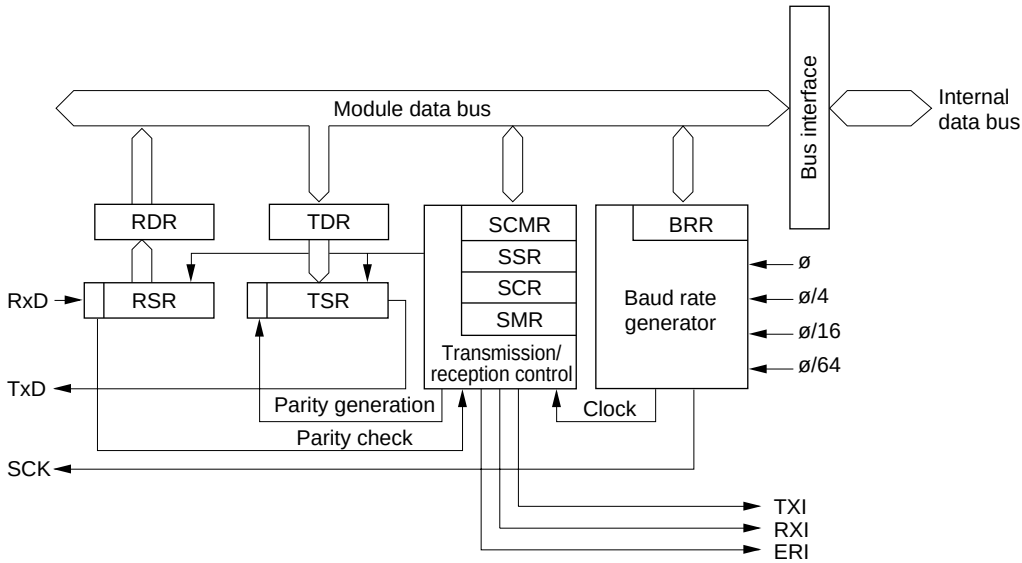
3.11 Smart Card Interface

The SCI supports a smart card interface as an IC card interface serial communication function conforming to ISO/IEC7816-3 (Identification Card).

Features

- Asynchronous mode
  - — Data length: 8 bits
  - — Parity bit generation and checking
  - — Transmission of error signal (parity error) in receive mode
  - — Error signal detection and automatic data retransmission in transmit mode
  - — Direct convention and inverse convention both supported
- Internal baud rate generator allows any bit rate to be selected
- Three interrupt sources
  - — Three interrupt sources—transmit data empty, receive data full, and transmit/receive error—that can issue requests independently
  - — The transmit data empty interrupt and receive data full interrupt can activate the DMA controller (DMAC) or data transfer controller (DTC) to execute data transfer

Block Diagram



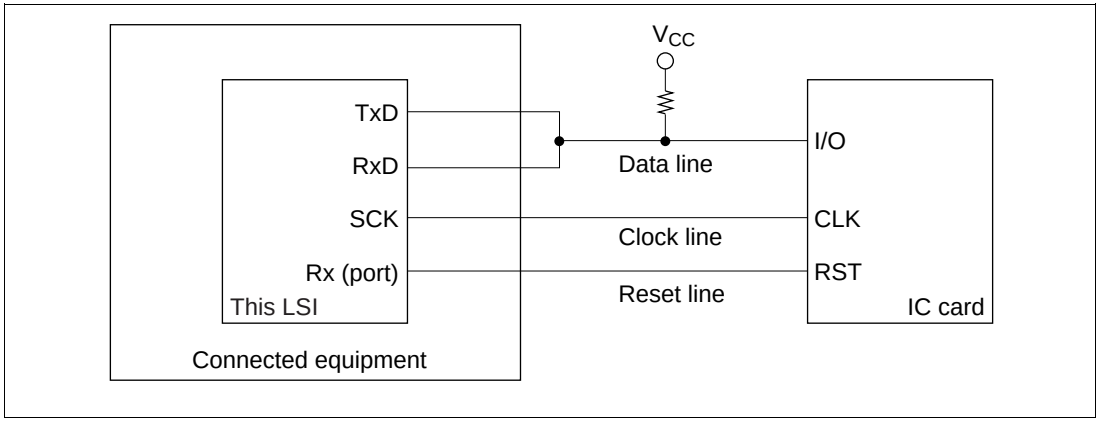
Legend:

SCMR:	Smart card mode register	SMR:	Serial mode register
RSR:	Receive shift register	SCR:	Serial control register
RDR:	Receive data register	SSR:	Serial status register
TSR:	Transmit shift register	BRR:	Bit rate register
TDR:	Transmit data register		

Operation

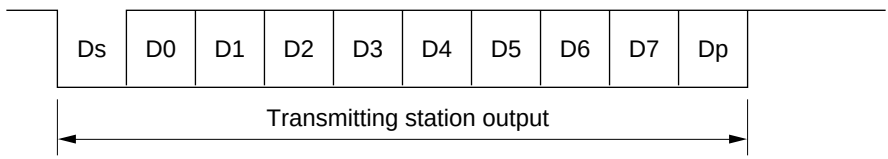
- Overview
  - — Only asynchronous communication is supported, with one frame consisting of 8-bit data plus a parity bit.
  - — In transmission, a guard time of at least 2 etu (1 etu in block transfer mode) (Elementary Time Unit: the time for transfer of one bit) is left between the end of the parity bit and the start of the next frame.
  - — If a parity error is detected during reception, a low error signal level is output for a 1 etu period 10.5 etu after the start bit (except in block transfer mode).
  - — If the error signal is sampled during transmission, the same data is transmitted automatically after the elapse of 2 etu or longer (except in block transfer mode).

Schematic Diagram of Smart Card Interface Pin Connections

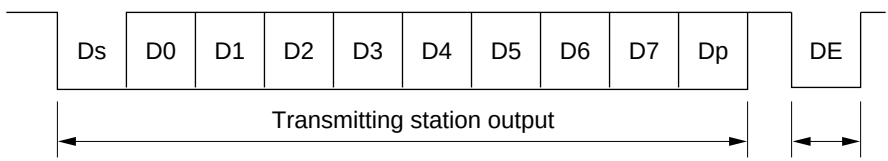


Smart Card Interface Data Format

When there is no parity error



When a parity error occurs



Legend:
Ds: Start bit
D0 to D7: Data bits
Dp: Parity bit
DE: Error signal

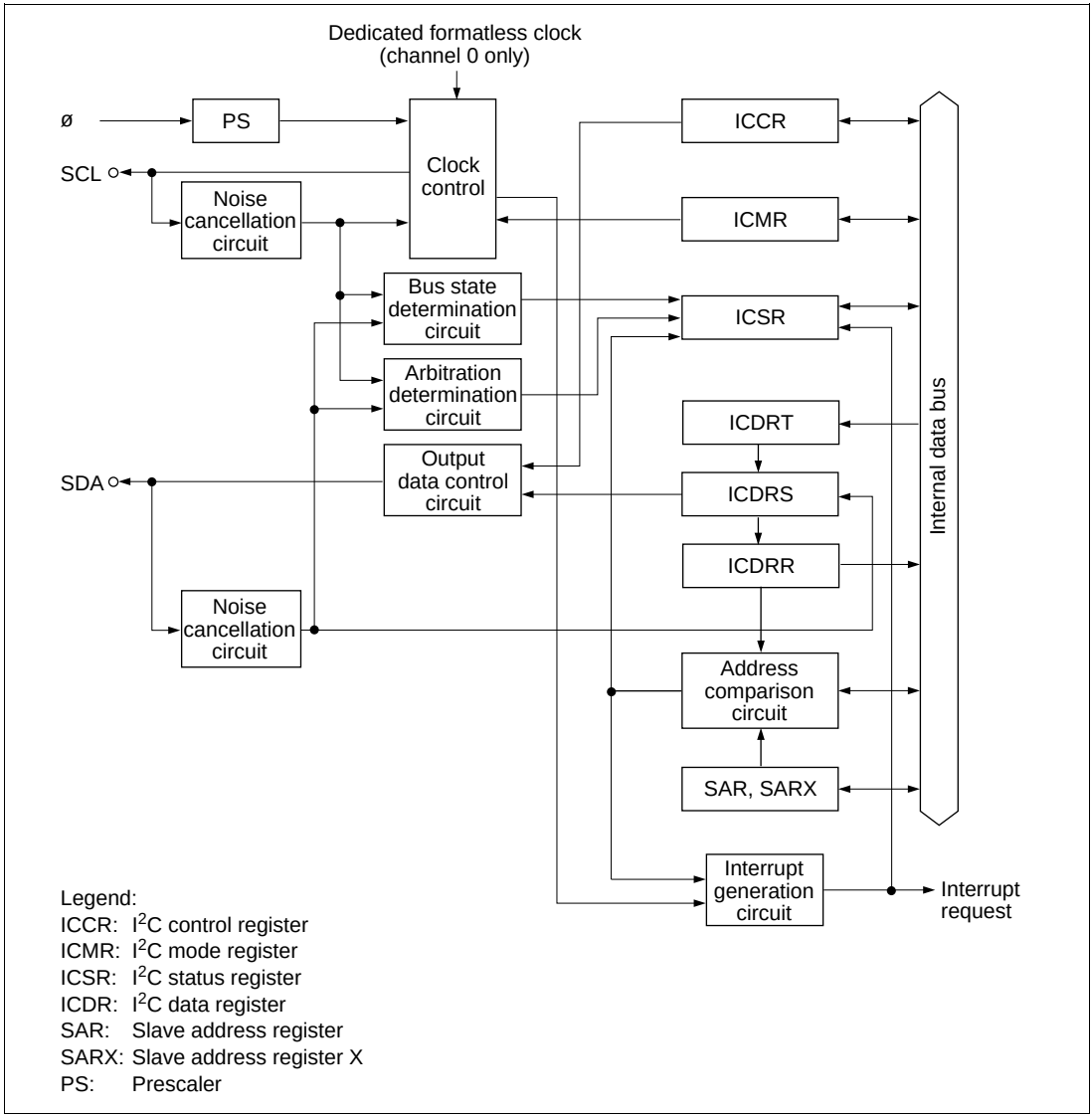
3.12 I²C Bus Interface [Option]

This series can be equipped with an optional two-channel serial interface conforming to the I²C bus (Inter IC Bus) interface proposed by Philips. Data transfer using the I²C bus interface requires only one data line (SDA) and one clock line (SCL) for each channel, providing an economical solution in terms of connectors and printed circuit board area.

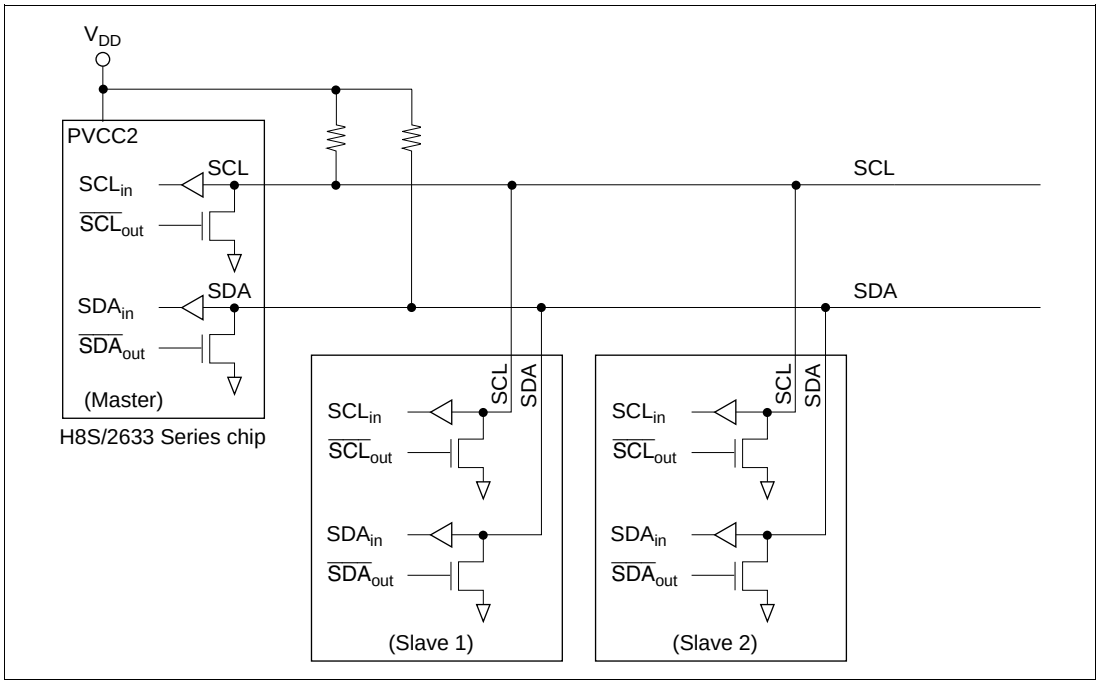
Features

- Selection of addressing format or non-addressing format
  - — I²C bus format: addressing format with acknowledge bit, master and slave operation
  - — Serial format: non-addressing format without acknowledge bit, master operation only
- I²C bus format conforms to Philips I²C bus interface
- Two ways of setting slave address (I²C bus format)
- Start and stop conditions generated automatically in master mode (I²C bus format)
- Selection of acknowledge output level when receiving (I²C bus format)
- Automatic loading of acknowledge bit when transmitting (I²C bus format)
- Wait function in master mode (I²C bus format)
  - — A wait can be inserted by driving the SCL pin low after transfer of data, excluding the acknowledge bit. The wait state is cleared by clearing the interrupt flag.
- Wait function in slave mode (I²C bus format)
  - — A wait request can be generated by driving the SCL pin low after transfer of data transfer, excluding the acknowledge bit. The wait request is cleared when the next transfer becomes possible.
- Three interrupt sources
  - — Data transfer end (including transmit mode transition with I²C bus format and address reception after loss of master arbitration)
  - — Address match: when any slave address matches or the general call address is received in slave receive mode with the I²C bus format
  - — Stop condition detection
- Selection of 16 internal clocks (in master mode)
- Direct bus drive (with SCL and SDA pins)
  - — Two pins—P35/SCL0 and P34/SDA0—(normally NMOS push-pull outputs) function as NMOS open-drain outputs when the bus drive function is selected.
  - — Two pins—P33/SCL1 and P32/SDA1—(normally CMOS pins) function as NMOS-only outputs when the bus drive function is selected.
- Automatic switching from formatless mode to I²C bus format (channel 0 only)
  - — Formatless operation (no start/stop conditions, non-addressing mode) in slave mode
  - — Operation using a common data pin (SDA) and independent clock pins (VSYNCI, SCL)
  - — Automatic switching from formatless mode to I²C bus format on the fall of the SCL pin

Block Diagram



Example of I²C Bus Interface Connections (when H8S/2633 Series Chip is Master)



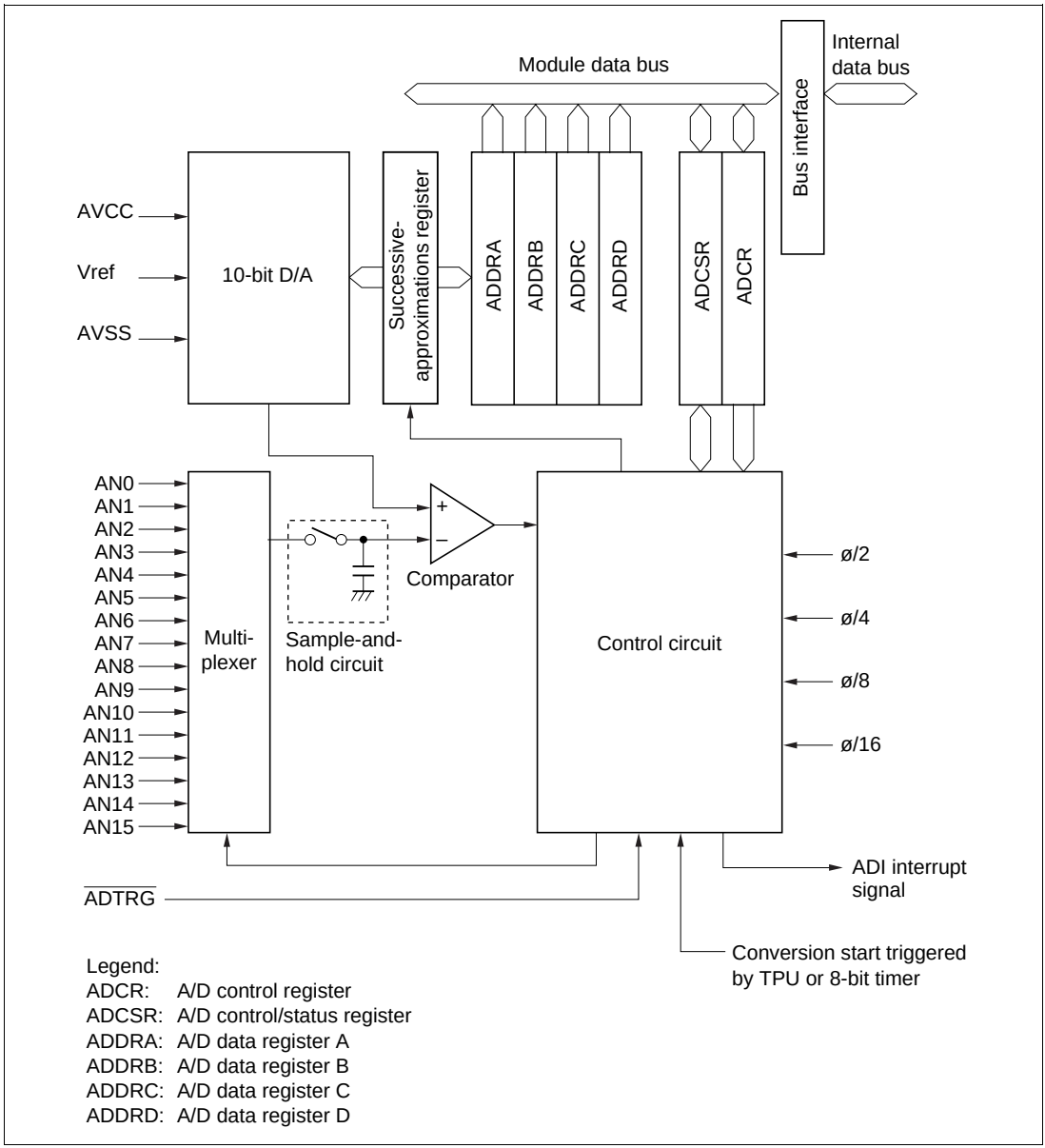
3.13 A/D Converter

This series incorporates an on-chip A/D converter with 10-bit precision. Analog signals can be input on up to 16 channels.

Features

- 10-bit resolution
- 16 input channels
- Settable analog conversion voltage range
  - — Conversion of analog input from 0 V to $V_{ref,}$ with the reference voltage pin (Vref) as the analog reference voltage
- High-speed conversion
  - — Minimum conversion time: 13.3 µs per channel (at 20 MHz operation)
- Selection of single mode or scan mode
  - — Single mode: A/D conversion on one channel
  - — Scan mode: continuous A/D conversion on one to four channels
- Three kinds of conversion start
  - — Selection of software or timer conversion start trigger (TPU or 8-bit timer), or $\overline{\text{ADTRG}}$ pin
- Four data registers
  - — Conversion results held in a 16-bit data register for each channel
- Sample-and-hold function
- A/D conversion end interrupt generation
  - — A/D conversion end interrupt (ADI) request can be generated at the end of A/D conversion
- Module stop mode can be set
  - — As the initial setting, A/D converter operation is halted. Register access is enabled by exiting module stop mode.

Block Diagram



Input Channel Setting

16-channel analog input is performed by means of the scan mode bit (SCAN) and channel select bits (CH3 to CH0) in ADCSR.

				Description	
CH3	**CH2**	**CH1**	**CH0**	**Single Mode (SCAN = 0)**	**Scan Mode (SCAN = 1)**
0	0	0	0	AN0 (initial value)	AN0
			1	AN1	AN0, AN1
		1	0	AN2	AN0 to AN2
			1	AN3	AN0 to AN3
	1	0	0	AN4	AN4
			1	AN5	AN4, AN5
		1	0	AN6	AN4 to AN6
			1	AN7	AN4 to AN7
1	0	0	0	AN8	AN8
			1	AN9	AN8, AN9
		1	0	AN10	AN8 to AN10
			1	AN11	AN8 to AN11
	1	0	0	AN12	AN12
			1	AN13	AN12, AN13
		1	0	AN14	AN12 to AN14
			1	AN15	AN12 to AN15

Operation

The successive approximation method is used for A/D conversion, with a 10-bit resolution. There are two operating modes—single or scan.

- Single Mode

  Single mode is selected when A/D conversion is to be performed on a single channel only.

  A/D conversion is started when the ADST bit is set to 1, according to the specified A/D conversion start condition.

  On completion of conversion, the ADF flag is set to 1. If the ADIE bit is set to 1 at this time, an ADI interrupt request is generated.

- Scan Mode

  Scan mode is selected when A/D conversion is to be performed repeatedly on a number of channels.

  Once the ADST bit is set to 1 according to the specified A/D conversion start condition, A/D conversion is performed repeatedly on the selected channels until the ADST bit is cleared to 0 by software.

  An ADI interrupt request can be generated on completion of the first conversion operation for all the selected channels.

3.14 D/A Converter

This series has two on-chip D/A converters with 8-bit-precision. Analog signals can be output on up to four channels.

Features

- Eight-bit resolution
- Two output channels × 2
- Maximum conversion time of 10 µs (with 20 pF load capacitance)
- Output voltage of 0 V to V_{ref}
- D/A output hold function in software standby mode
- Module stop mode can be set
  - As the initial setting, D/A converter operation is halted. Register access is enabled by exiting module stop mode.

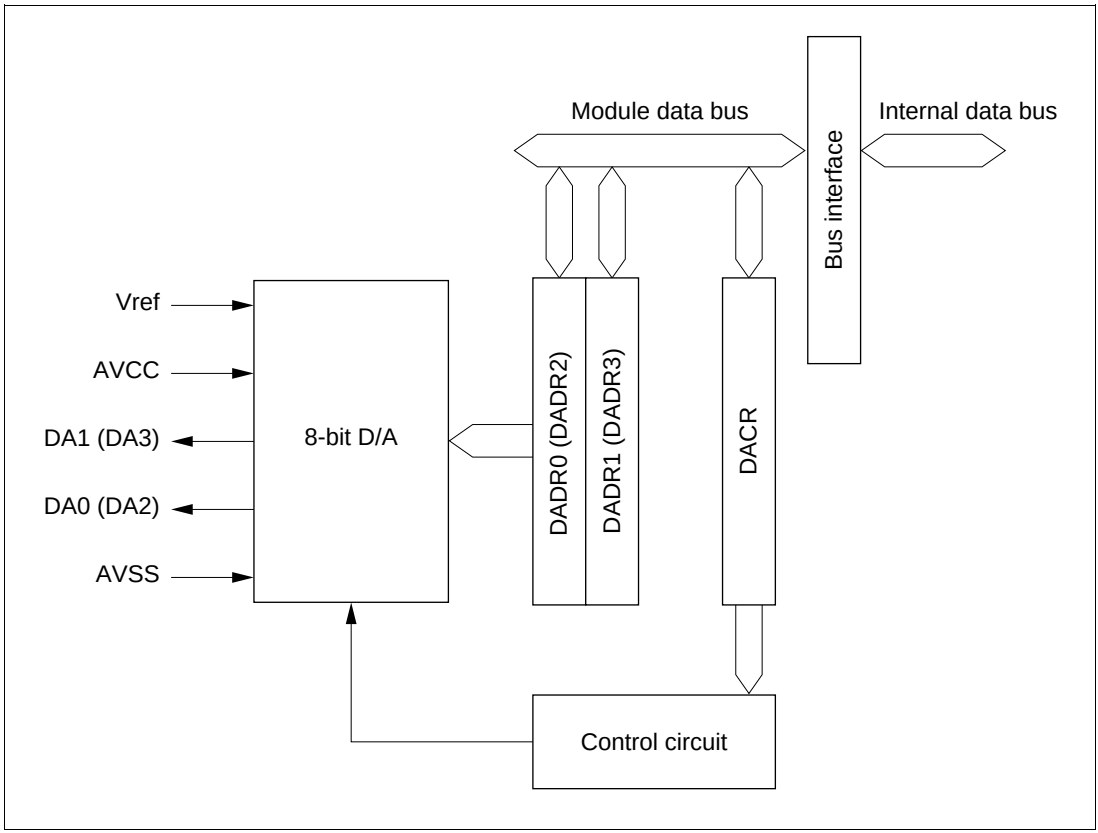
Operation

D/A converter operation is enabled by setting the D/A output enable bit to 1. While this bit is set to 1, DADR contents are constantly converted and output to the corresponding pin.

The output value is:

$$\frac{\text{DADR contents}}{256} \times V_{ref}$$

Block Diagram



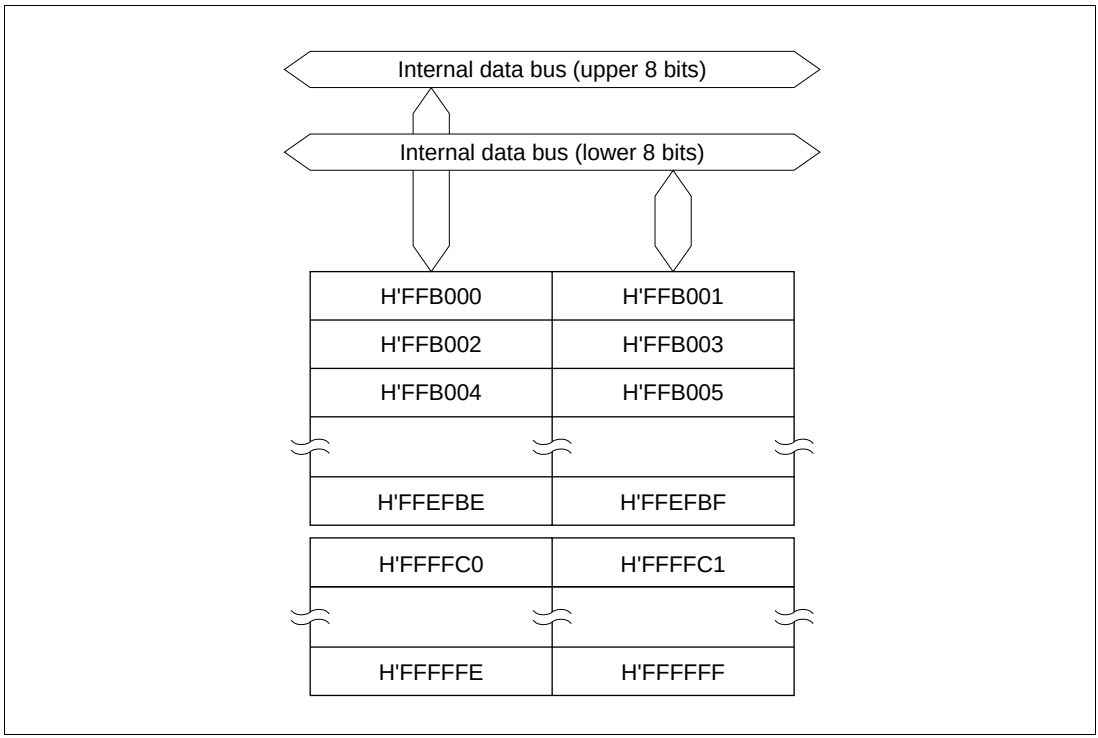
3.15 RAM

The H8S/2633 has 16 kbytes of on-chip high-speed static RAM, the H8S/2632 has 12 kbytes, and the H8S/2631 has 8 kbytes.

The on-chip RAM is connected to the bus master by a 16-bit data bus, enabling both byte data and word data to be accessed in one state. This makes it possible to perform fast word data transfer.

The on-chip RAM can be enabled or disabled by means of the RAM enable bit (RAME) in the system control register (SYSCR).

Block Diagram of RAM (H8S/2633)



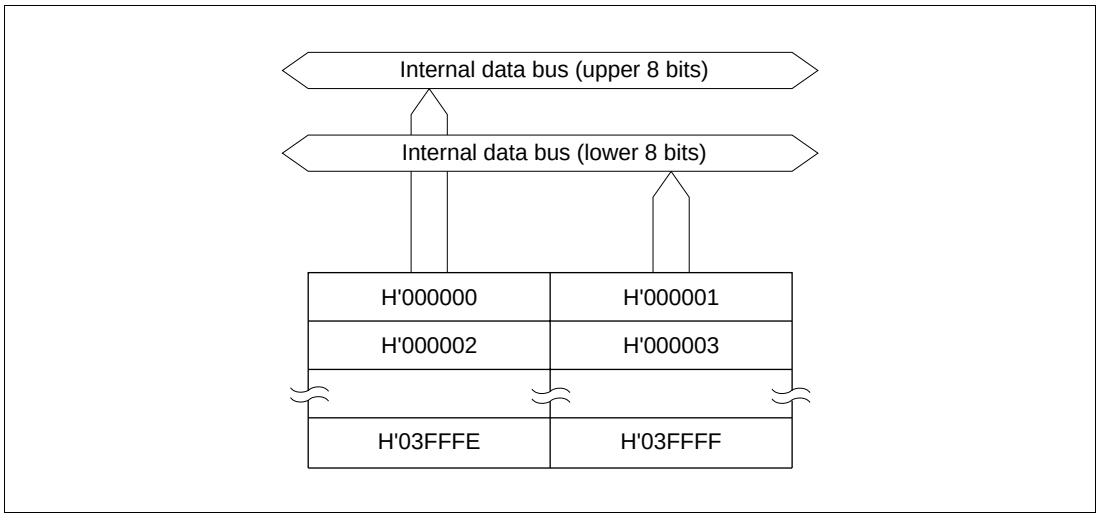
3.16 ROM (Preliminary)

The H8S/2633 has 256 kbytes of on-chip flash memory or mask ROM. The H8S/2632 has 192 kbytes of on-chip mask ROM, and the H8S/2631 has 128 kbytes.

The ROM is connected to the bus master by a 16-bit data bus, enabling both byte data and word data to be accessed in one state. This allows rapid instruction fetches and improves processing speed.

In addition to erasing and programming with a dedicated PROM programmer, the flash memory version can also be erased and programmed on-board.

ROM Block Diagram (H8S/2633)

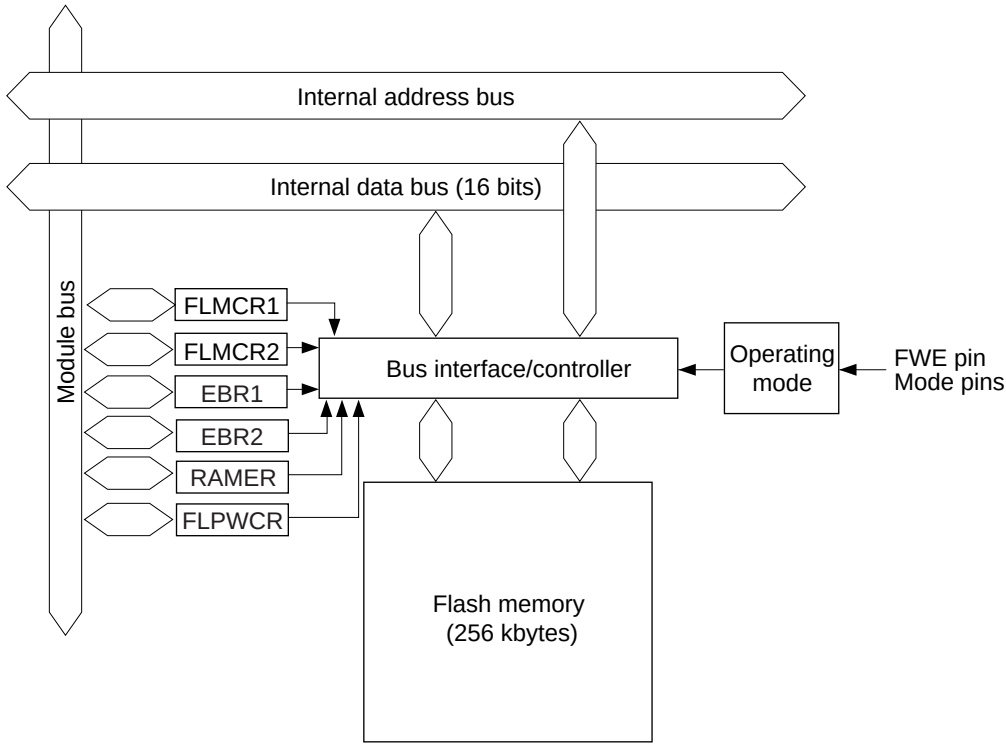


On-chip ROM is enabled or disabled by means of mode pins MD1 and MD0.

Features of Flash Memory

- Four operating modes
  - — Program mode
  - — Erase mode
  - — Program-verify mode
  - — Erase-verify mode
- Programming/erase methods
  - — 128 bytes programmed simultaneously
  - — 4-kbyte, 32-kbyte, or 64-kbyte units can be set for block erasing
- Programming/erase times
  - — Programming time: T.B.D. ms (typ.) per 128-byte programming operation
    T.B.D. µs (typ.) per byte
  - — Erase time: T.B.D. ms (typ.) per block
- Reprogramming up to 100 times
- On-board programming modes
  - — Boot mode
  - — User program mode
- Automatic bit rate adjustment
  - — With data transfer in boot mode, the bit rate of the chip can be automatically adjusted to match the transfer bit rate of the host
- Flash memory emulation in RAM
  - — Flash memory programming can be emulated in real time by overlapping a part of RAM onto flash memory.
- Protect modes
  - — There are three protect modes, hardware, software, and error protect, which allow protected status to be designated for flash memory program/erase/verify operations
- Writer mode
  - — On-board programming mode and writer mode, using a PROM programmer, are available as flash memory program/erase modes

Block Diagram of Flash Memory (H8S/2633)

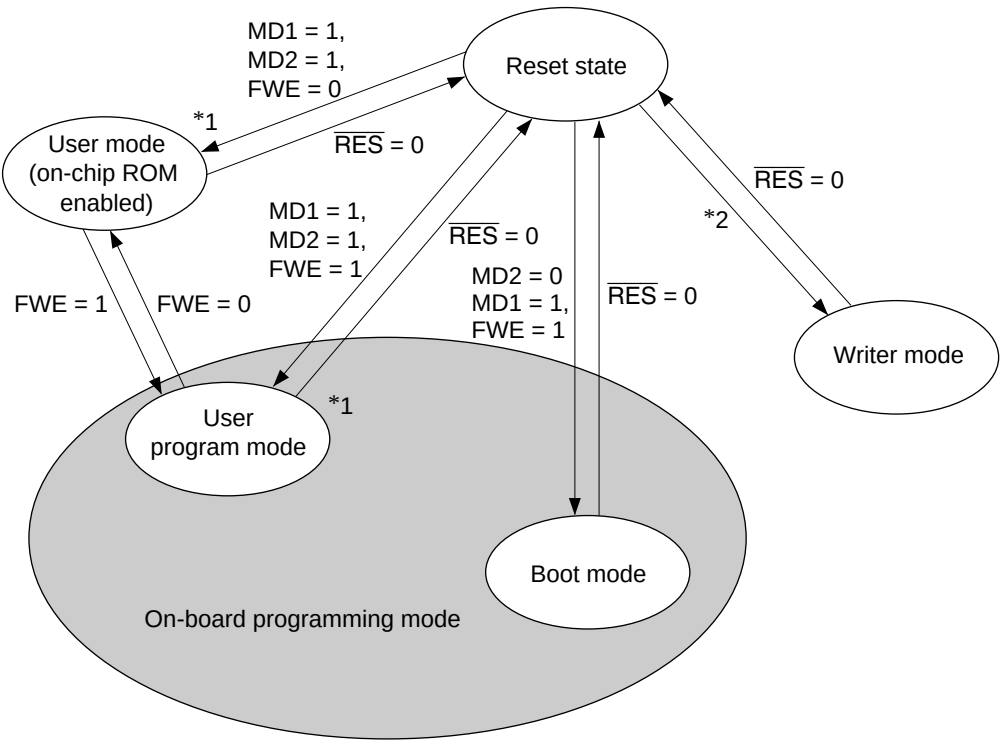


Legend:
FLMCR1: Flash memory control register 1
FLMCR2: Flash memory control register 2
EBR1: Erase block register 1
EBR2: Erase block register 2
RAMER: RAM emulation register
FLPWCR: Flash memory power control register

Mode Transitions

When the mode pins and FWE pins are set in the reset state and a reset-start is executed, the MCU enters one of the operating modes as shown in the figure below. In user mode, flash memory can be read but not programmed or erased.

Flash memory can be programmed and erased in boot mode, user program mode, and writer mode.



Notes: Only make a transition between user mode and user program mode when the CPU is not accessing the flash memory.

1. RAM emulation possible
2. MD0 = 0, MD1 = 0, MD2 = 0, P14 = 0, P16 = 0, PF0 = 1

Section 4 Power-Down Modes

In addition to the normal program execution state, this series has power-down modes in which operation of the CPU and oscillator is halted and power consumption is reduced. Low-power operation can be achieved by individually controlling the CPU, on-chip supporting modules, and so on.

This series' operating modes are as follows:

1. High-speed mode
2. Medium-speed mode
3. Subactive mode
4. Sleep mode
5. Subsleep mode
6. Watch mode
7. Module stop mode
8. Software standby mode
9. Hardware standby mode

Of these, 2 to 9 are power-down modes. Sleep mode and subsleep mode are CPU modes, medium-speed mode is a CPU and bus master mode, subactive mode is a CPU, bus master, and on-chip supporting module mode, and module stop mode is an on-chip supporting module mode (including bus masters other than the CPU). A combination of certain of these modes can be set.

Medium-Speed Mode: When bits SCK2, SCK1, and SCK0 in the standby control register (SBYCR) are set to 1 in high-speed mode, medium-speed mode is entered as soon as the current bus cycle ends. In medium-speed mode, the bus masters—the CPU, DMAC, and DTC—operate on the operating clock (ø/2, ø/4, ø/8, ø/16, or ø/32) specified by bits SCK2 to SCK0. However, on-chip supporting functions other than the bus masters operate on the high-speed clock (ø).

Subactive Mode: If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, the DTON bit and LSON bit in the low-power control register (LPWRCR) are both set to 1, and the PSS bit in the timer control/status register (TCSR (WDT1)) is set to 1, the CPU enters subactive mode. A transition to subactive mode is also caused by an interrupt generated in watch mode when the LSON bit in LPWRCR is set to 1, and by an interrupt generated in subsleep mode.

In subactive mode, the CPU performs low-speed sequential program execution using the subclock, and supporting functions other than TMR0 to TMR3, WDT0, and WDT1 are halted.

Sleep Mode: If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, the CPU enters sleep mode. In sleep mode, CPU operation stops but the contents of the CPU's internal registers are retained. Other supporting functions do not stop.

Subsleep Mode: If a SLEEP instruction is executed when the SSBY bit in SBYCR is cleared to 0, the LSON bit in LPWRCR is set to 1, and the PSS bit in TCSR (WDT1) is set to 1, the CPU enters subsleep mode.

In subsleep mode, CPU operation stops, together with the operation of supporting functions other than TMR0 to TMR3, WDT0, and WDT1. As long as the specified voltage is supplied, the contents of CPU registers, some on-chip supporting module registers, and on-chip RAM are retained, and I/O ports retain their states prior to the transition.

Watch Mode: If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, the DTON bit in LPWRCR is cleared to 0, and the PSS bit in TCSR (WDT1) is set to 1, the CPU enters watch mode.

In watch mode, CPU operation stops, together with the operation of supporting functions other than WDT1. As long as the specified voltage is supplied, the contents of CPU registers, some on-chip supporting module registers, and on-chip RAM are retained, and I/O ports retain their states prior to the transition.

Module Stop Mode: Module stop mode can be used to start and stop individual on-chip supporting modules.

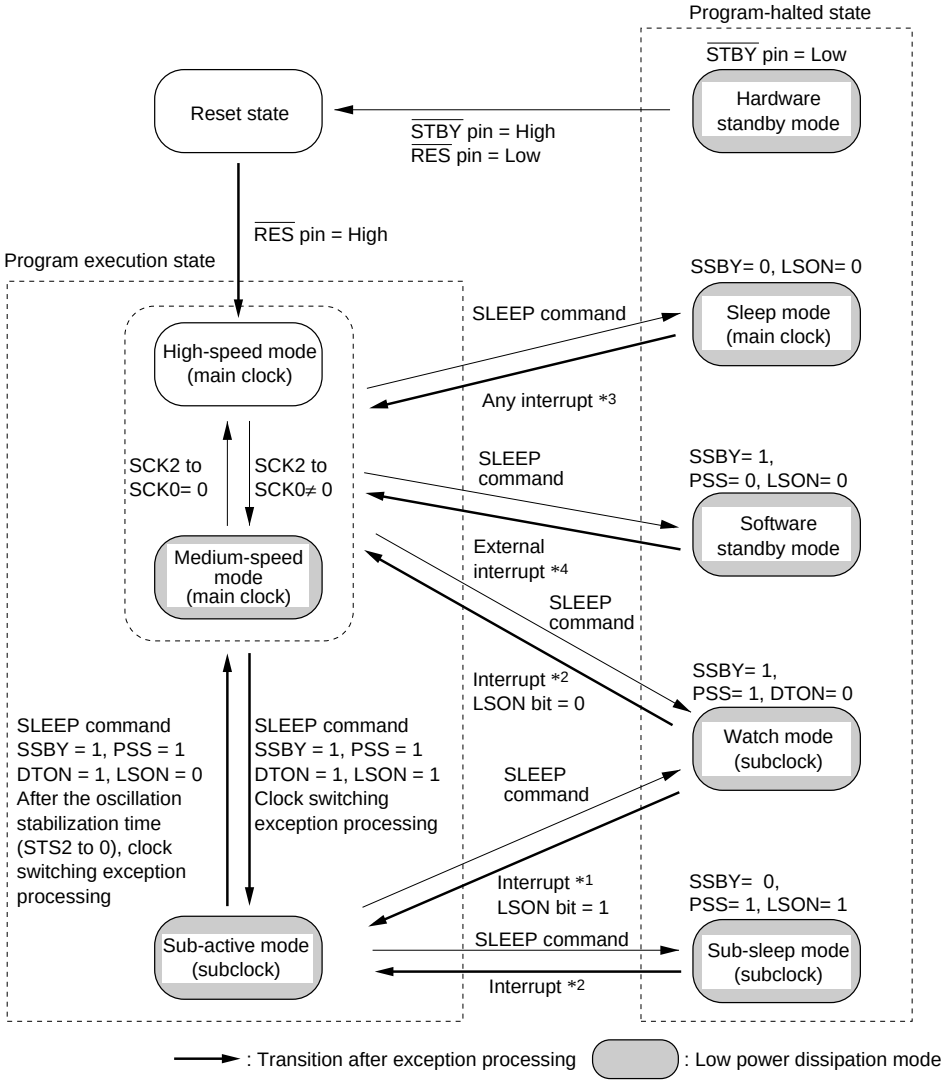
When the MSTP bit corresponding to a particular supporting function in the module stop control register (MSTPCR) is set to 1, operation of the specified module stops at the end of the bus cycle and a transition is made to module stop mode. The CPU continues operating independently.

Software Standby Mode: If a SLEEP instruction is executed when the SSBY bit in SBYCR is set to 1, the LSON bit in LPWRCR is cleared to 0, and the PSS bit in TCSR (WDT1) is cleared to 0, software standby mode is entered. In this mode, the CPU, on-chip supporting functions, and oscillator all stop. However, the contents of the CPU's internal registers, RAM data, and the states of on-chip supporting functions other than the SCI, 14-bit PWM, and A/D, and the states of I/O ports, are retained.

Hardware Standby Mode: When the $\overline{\text{STBY}}$ pin is driven low, a transition is made to hardware standby mode from any state.

In hardware standby mode, all functions enter the reset state and stop operation, resulting in extremely low power consumption. As long as the prescribed voltage is supplied, on-chip RAM data is retained. I/O ports are set to the high-impedance state.

Mode Transition Diagram



Notes: 1. NMI, IRQ0 to IRQ7, and WDT1 interrupts
2. NMI, IRQ0 to IRQ7, IWDT0 interrupts, WDT1 interrupt, and TMR0 to TMR3 interrupts
3. All interrupts
4. NMI and IRQ0 to IRQ7

- When a transition is made between modes by means of an interrupt, the transition cannot be made on interrupt source generation alone. Ensure that interrupt handling is performed after accepting the interrupt request.
- From any state except hardware standby mode, a transition to the reset state occurs when $\overline{\text{RES}}$ is driven Low.
- From any state, a transition to hardware standby mode occurs when $\overline{\text{STBY}}$ is driven low.
- Always select high-speed mode before making a transition to watch mode or sub-active mode.

Chip Internal States in Each Mode

Function		High-Speed	Medium-Speed	Sleep	Module Stop	Watch	Sub-active	Sub-sleep	Software Standby	Hardware Standby
System clock oscillator		Function-ing	Function-ing	Function-ing	Function-ing	Halted	Halted	Halted	Halted	Halted
Subclock oscillator		Function-ing	Function-ing	Function-ing	Function-ing	Function-ing	Function-ing	Function-ing	Function-ing	Halted
CPU	Instruction registers	Function-ing	Medium-speed operation	Halted (retained)	High/ medium-speed operation	Halted (retained)	Subclock operation	Halted (retained)	Halted (retained)	Halted (unde-fined)
External interrupts	NMI	Function-ing	Function-ing	Function-ing	Function-ing	Function-ing	Function-ing	Function-ing	Function-ing	Halted
	IRQ0 to IRQ7									
Supporting functions	WDT1	Function-ing	Function-ing	Function-ing	Halted (retained)	Subclock operation	Subclock operation	Subclock operation	Halted (retained)	Halted (reset)
	WDT0	Function-ing	Function-ing	Function-ing	Halted (retained)	Halted (retained)	Subclock operation	Subclock operation	Halted (retained)	Halted (reset)
	TMR									
	DMAC	Function-ing	Medium-speed operation	Function-ing	Halted (retained)	Halted (retained)	Halted (retained)	Halted (retained)	Halted (retained)	Halted (reset)
	DTC									
	TPU	Function-ing	Function-ing	Function-ing	Halted (retained)	Halted (retained)	Halted (retained)	Halted (retained)	Halted (retained)	Halted (reset)
	IIC0									
	IIC1									
	PCB									
	PPG									
	D/A0, D/A1									
	SCI0	Function-ing	Function-ing	Function-ing	Halted (reset)	Halted (reset)	Halted (reset)	Halted (reset)	Halted (reset)	Halted (reset)
	SCI1									
	SCI2									
	SCI3									
	SCI4									
	PWM0, PWM1									
	A/D									
	RAM	Function-ing	Function-ing	Function-ing (DTC)	Function-ing	Retained	Function-ing	Retained	Retained	Retained
	I/O	Function-ing	Function-ing	Function-ing	Function-ing	Retained	Function-ing	Retained	Retained	High impedance

Notes: “Halted (retained)” means that internal register values are retained. The internal state is “operation suspended.”
“Halted (reset)” means that internal register values and internal states are initialized.
In module stop mode, only modules for which a stop setting has been made are halted (reset or retained).

Power-Down Mode Transition Conditions

State before Transition	Control Bit States when Transition Is Made				State after Transition by SLEEP Instruction	State after Recovery by Interrupt
	SSBY	PSS	LSON	DTON		
High-speed/ medium-speed	0	*	0	*	Sleep	High-speed/medium-speed
	0	*	1	*	—	—
	1	0	0	*	Software standby	High-speed/medium-speed
	1	0	1	*	—	—
	1	1	0	0	Watch	High-speed
	1	1	1	0	Watch	Subactive
	1	1	0	1	—	—
	1	1	1	1	Subactive	—
Subactive	0	0	*	*	—	—
	0	1	0	*	—	—
	0	1	1	*	Subsleep	Subactive
	1	0	*	*	—	—
	1	1	0	0	Watch	High-speed
	1	1	1	0	Watch	Subactive
	1	1	0	1	High-speed	—
	1	1	1	1	—	—

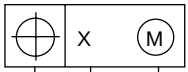
*: Don't care

—: Do not set.

Packages

Package Dimension Diagrams (Unit: mm)

Indication according to geometrical tolerances

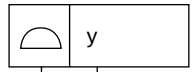


Tolerance method based on maximum solid state

Tolerance value

Kind of geometrical tolerance (in this case, positional tolerance)

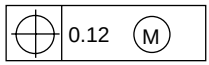
Pin precision y indication



Allowable value

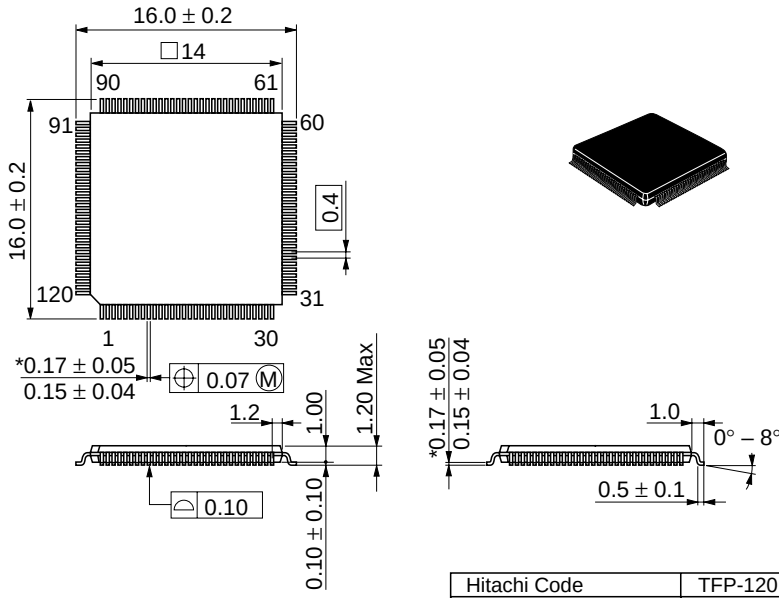
Pin precision

Example:



This indicates that the allowable pin displacement from the true central position is 0.12 mm when pin width b is the maximum dimension. If b is smaller than the maximum dimension, the tolerance can be extended accordingly.

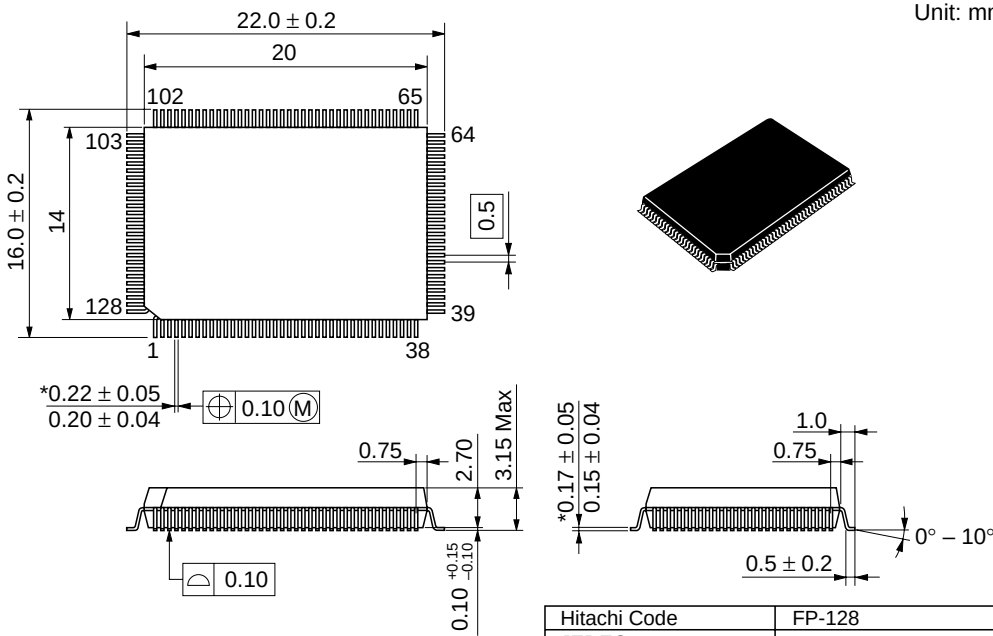
Unit: mm



Hitachi Code	TFP-120
JEDEC	—
EIAJ	Conforms
Weight (reference value)	0.5 g

*Dimension including the plating thickness
Base material dimension

Unit: mm



Hitachi Code	FP-128
JEDEC	—
EIAJ	—
Weight (reference value)	1.7 g

*Dimension including the plating thickness
Base material dimension

H8S/2633 Series Overview

Publication Date: 1st Edition, February 1999
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits Group
Hitachi, Ltd.
Edited by: Technical Documentation Group
UL Media Co., Ltd.