7641 Group

SINGLE-CHIP 8-BIT CMOS MICROCOMPUTER

DESCRIPTION

The 7641 group is the 8-bit microcomputer based on the 7600 series core (740 family core compatible) technology.

The 7641 group is designed for PC peripheral devices, including the USB, DMAC, Serial I/O, UART, Timer, Master CPU bus interface and so on.

FEATURES

<Microcomputer mode)

- Basic machine-language instructions 71
- Minimum instruction execution time ..... 83 ns
  (at 24 MHz oscillation frequency)
- Memory size
  - ROM ..... 32 Kbytes
  - RAM ..... 1 Kbytes
- Programmable input/output ports 66
- Software pull-up resistors ..... Built-in
- Interrupts ..... 24 sources, 24 vectors
  (external 5 including Key input, internal 18, software 1)
- USB function control unit
  - Transceiver ..... USB std. spec. ver.1.1
- Timers ..... 16-bit × 2 (Timers X, Y)
  8-bit × 3 (Timers 1, 2, 3)
- Serial I/O ..... 8-bit × 1
- UART ..... 8-bit × 2
- DMAC ..... 2 channels
- Master CPU bus interface ..... 2 bytes
- Special count source generator ..... 8-bit × 1
- Clock generating circuit ..... Built-in
  (connect to external ceramic resonator or quartz-crystal oscillator)
- Power source voltage
  - At 24 MHz oscillation frequency, ϕ = 12 MHz ..... 4.15 to 5.25 V
  - At 24 MHz oscillation frequency, ϕ = 6 MHz ..... 3.00 to 3.60 V
- Operating temperature range ..... –20 to 70°C
- Packages
  - FP ..... 80P6N-A (80-pin QFP)
  - HP ..... 80P6Q-A (80-pin LQFP)

<Flash memory mode>

- Power source voltage
  - At 24 MHz oscillation frequency, ϕ = 6 MHz ..... 4.15 to 5.25 V
  - At 24 MHz oscillation frequency, ϕ = 6 MHz ..... 3.00 to 3.60 V
- Program/Erase voltage ..... V_{PP} = 4.50 V to 5.25 V
  At 24 MHz oscillation frequency, ϕ = 6 MHz
- Memory size
  - Flash ROM ..... 32 Kbytes
  - RAM ..... 2.5 Kbytes
- Flash memory mode ..... 3 modes
  - Parallel I/O mode
  - Standard serial I/O mode
  - CPU rewrite mode
- Programming method ..... Programming in unit of byte
- Erasing method
  - Batch erasing
  - Block erasing
- Program/Erase control by software command
- Command number ..... 6 commands
- Number of times for programming/erasing 100
- ROM code protection
  - Available in parallel I/O mode and standard serial I/O mode

APPLICATION

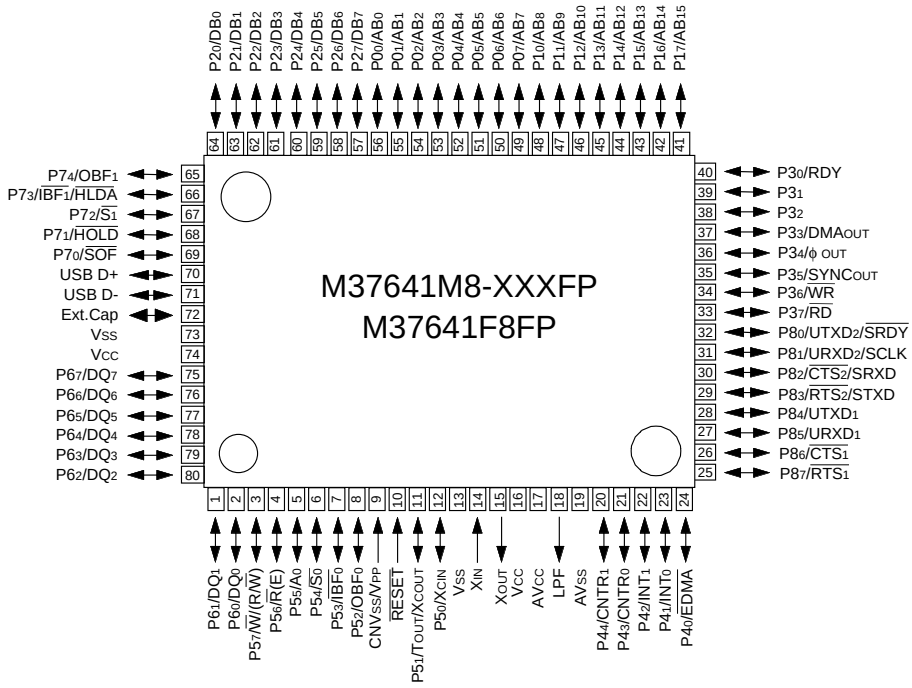
Audio, musical instrument, printer, scanner, modem, other PC peripheral devices

■Notes

1. The specifications of this product are subject to change because it is under development. Inquire the use of Mitsubishi Electric Corporation.

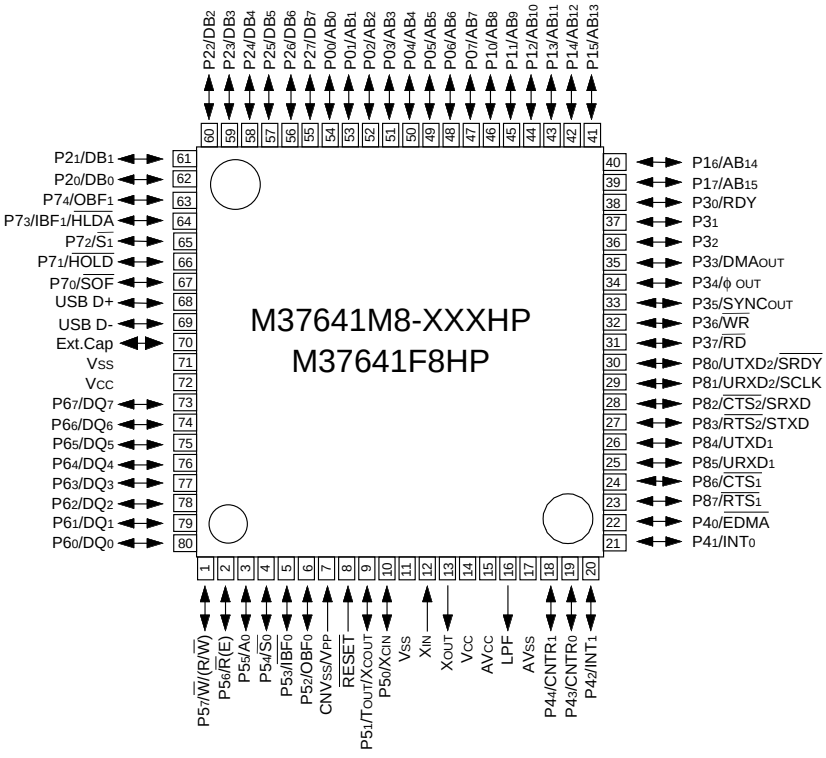
2. The flash memory version cannot be used for application embedded in the MCU card.

PIN CONFIGURATION (TOP VIEW)



Package type : 80P6N-A

Fig. 1 M37641M8-XXXFP, M37641F8FP pin configuration



Package type : 80P6Q-A

Fig. 2 M37641M8-XXXHP, M37641F8HP pin configuration

MITSUBISHI ELECTRIC

FUNCTIONAL BLOCK DIAGRAM (Package: 80P6N-A)

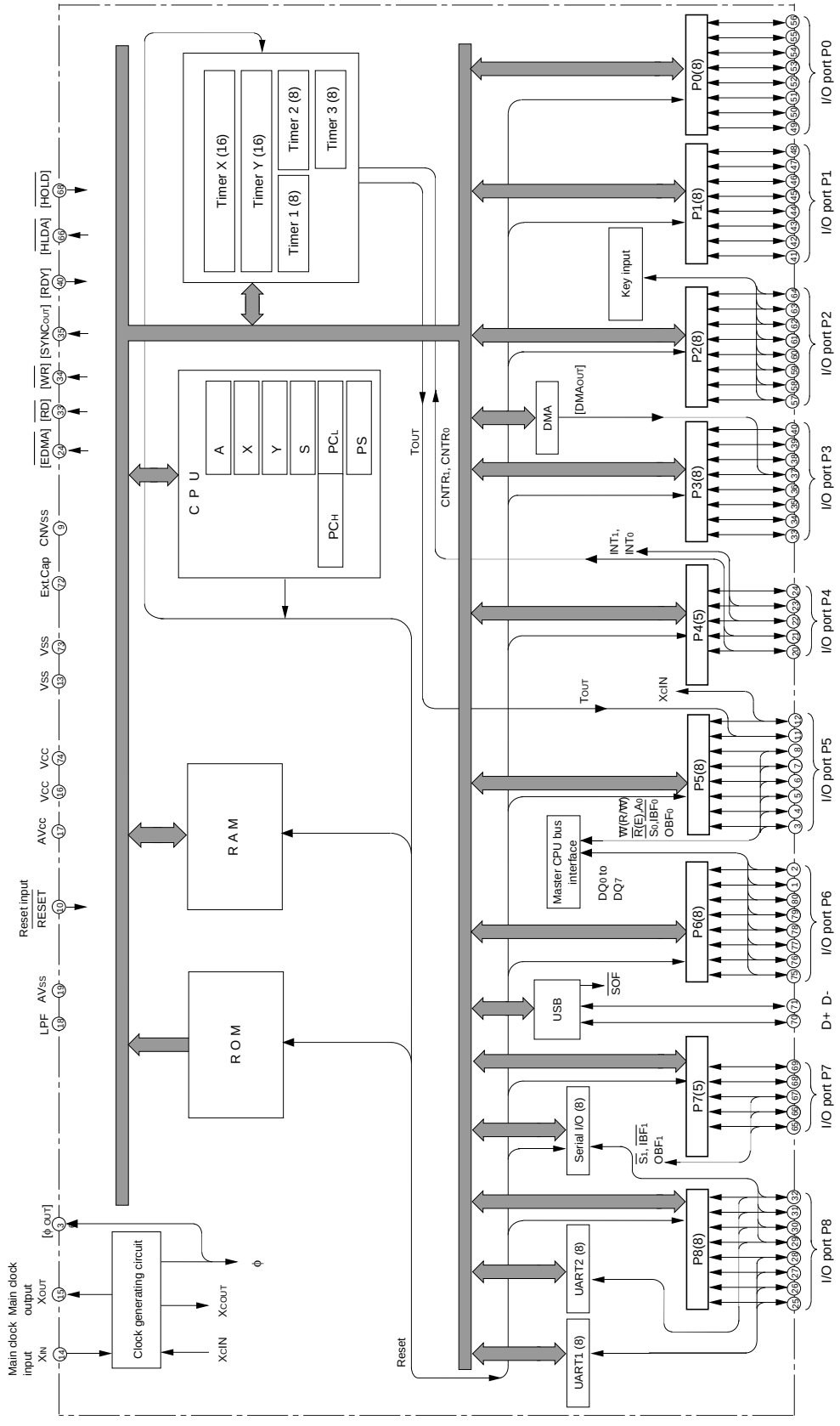


Fig. 3 Functional block diagram

PIN DESCRIPTION

Table 1 Pin description (1)

Pin	Name	Function	Function except a port function
VCC, VSS	Power source	• Apply 4.15 V – 5.25 V for 5 V version or 3.00 V – 3.60 V for 3 V version to the Vcc pin. Apply 0 V to the Vss pin.	
CNVss/VPP	CNVss	• This controls the MCU operating mode. Connect this pin to Vss. If connecting this pin to Vcc, the internal ROM is inhibited. In the flash memory version this pin functions as a VPP power supply input pin.	
AVss/AVcc	Analog power supply	• These pins are the power supply inputs for analog circuitry.	
$\overline{\text{RESET}}$	Reset input	• Reset input pin for active “L.”	
XIN	Clock input	• Connect a ceramic resonator or a quartz-crystal oscillator between the XIN and XOUT pins to set the oscillation frequency. • If an external clock is used, connect the clock source to the XIN pin and leave the XOUT pin open.	
XOUT	Clock output		
LPF	LPF	• Loop filter for the frequency synthesizer.	
Ext. Cap.	3.3 V line power supply	• Power supply input pin for 3.3 V USB line driver.	
USB D+	USB D+	• USB D+ voltage signal port. Connect a 27 to 33 Ω (recommended) resistor in series.	
USB D-	USB D-	• USB D- voltage signal port. Connect a 27 to 33 Ω (recommended) resistor in series.	
P00/AB0– P07/AB7	I/O port P0	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the address bus.	
P10/AB8– P17/AB15	I/O port P1	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the address bus.	
P20/DB0– P27/DB7	I/O port P2	• 8-bit I/O port. • CMOS compatible input level or VIHL input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the data bus.	• Key-on wake-up interrupt input pin
P30/RDY, P31, P32, P33/DMAOUT, P34/ϕ OUT, P35/SYNCOUT, P36/$\overline{\text{WR}}$, P37/$\overline{\text{RD}}$	I/O port P3 (See Remarks.)	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the control bus.	• External memory control pin
P40/$\overline{\text{EDMA}}$, P41/INT0, P42/INT1, P43/CNTR0, P44/CNTR1	I/O port P4	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When connecting an external memory, these function as the control bus.	• External memory control pin • External interrupt pin • Timer X, Timer Y pin
P50/XCIN, P51/TOUT/ XCOUT, P52/OBF0, P53/IBF0, P54/S0, P55/A0, P56/$\overline{\text{R}}$(E), P57/$\overline{\text{W}}$(R/$\overline{\text{W}}$)	I/O port P5	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When enabling the Master CPU bus interface function, CMOS or TTL input level can be selected as an input.	• Sub-clock generating input pin • Timers 1, 2 pulse output pins • Sub-clock generating output pin • Master CPU bus interface pin

Table 2 Pin description (2)

Pin	Name	Function	Function except a port function
P60/DQ0–P67/DQ7	I/O port P6	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output. • When enabling the bus interface function, CMOS or TTL input level can be selected as its input.	• Master CPU bus interface pin
P70/$\overline{SOF}$, P71/$\overline{HOLD}$, P72/$\overline{S1}$, P73/$\overline{IBF1}$/$\overline{HLDA}$, P74/OBF1	I/O port P7	• 5-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output.	• USB function pin • Master CPU bus interface pin
P80/UTXD2/$\overline{SRDY}$, P81/URXD2/SCLK, P82/$\overline{CTS2}$/SRXD, P83/$\overline{RTS2}$/STXD, P84/UTXD1, P85/URXD1, P86/$\overline{CTS1}$, P87/$\overline{RTS1}$	I/O port P8	• 8-bit I/O port. • CMOS compatible input level. • CMOS 3-state output structure. • I/O direction register allows each pin to be individually programmed as either input or output.	• Serial I/O pin • UART2 pin • UART1 pin

Remarks

•DMAOUT pin

If externally detecting the timing of DMA execution, use the signal from this pin. It is “H” level during DMA transferring. This signal is valid in the memory expansion and microprocessor modes.

•SYNCOUT pin

If externally detecting the timing of OP code fetch, use the signal from this pin. This signal is valid in the memory expansion and microprocessor modes.

PART NUMBERING

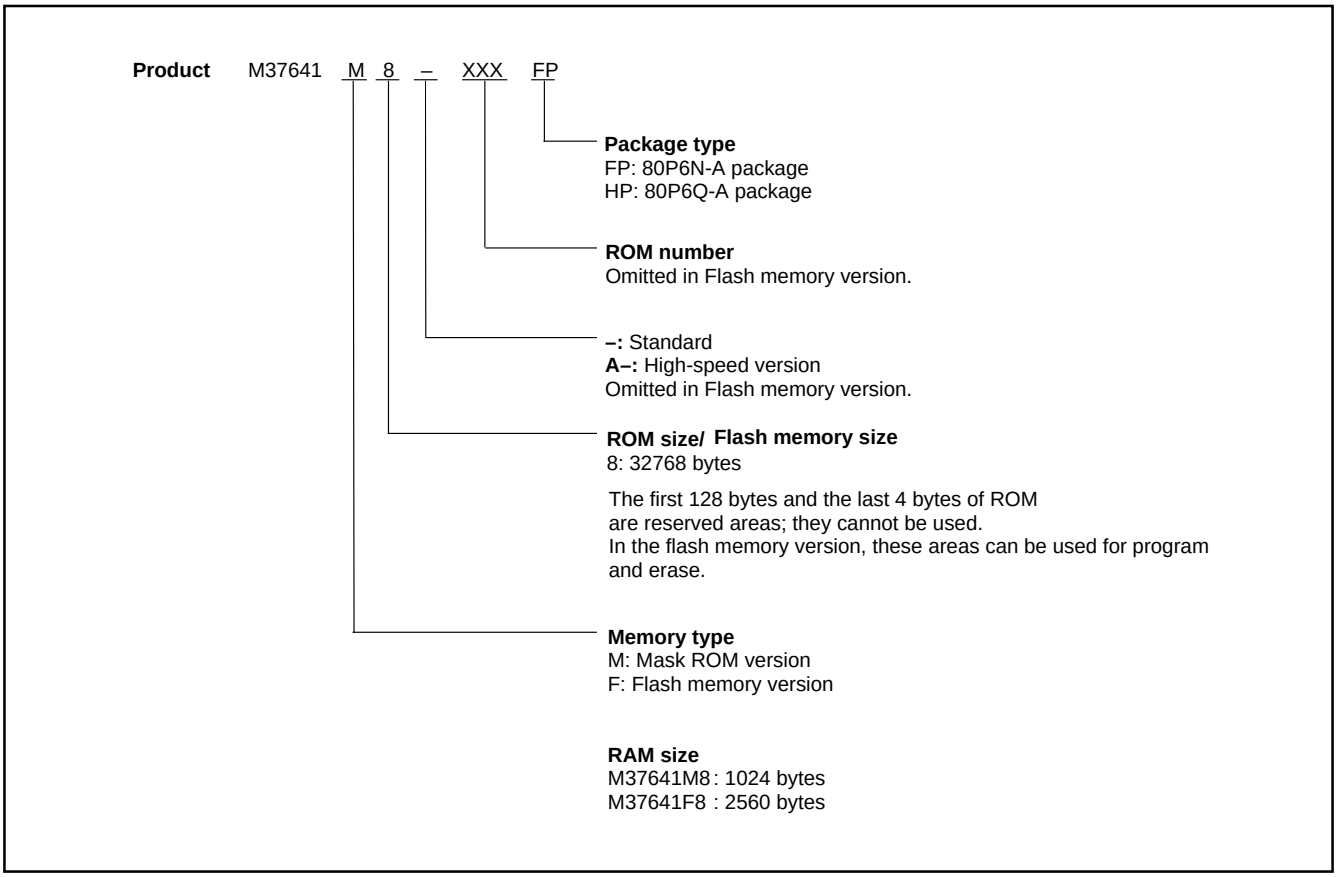


Fig. 4 Part numbering

GROUP EXPANSION

Mitsubishi plans to expand the 7641 group as follows.

Memory Type

Supports for mask ROM and flash memory versions.

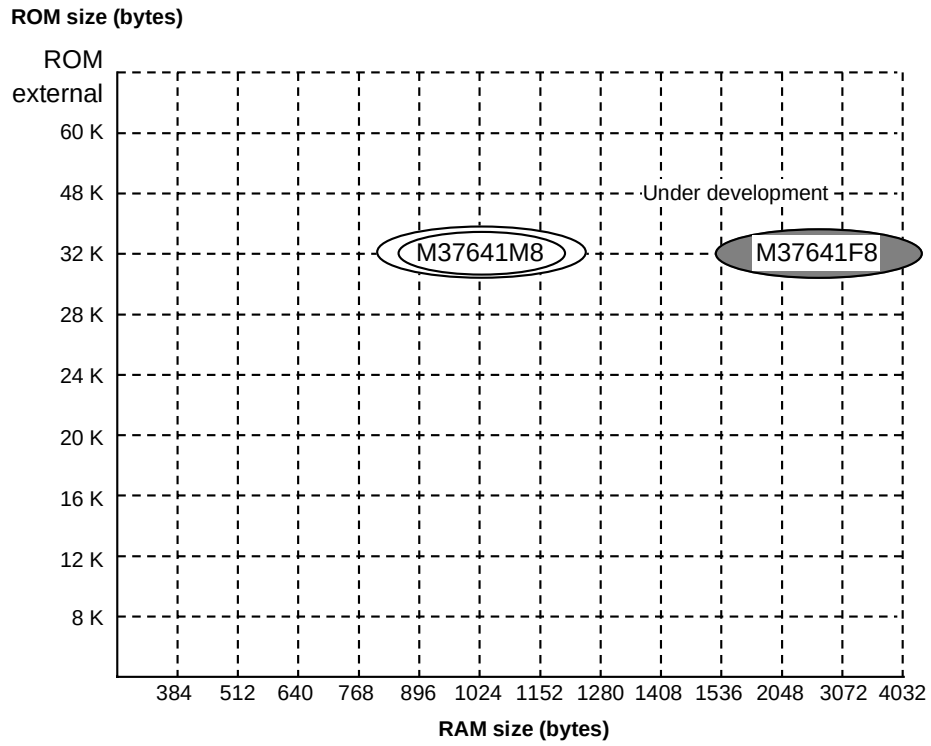
Memory Size

ROM size ..... 32 Kbytes
RAM size ..... 1024 to 2560 bytes

Packages

80P6N-A ..... 0.8 mm-pitch plastic molded QFP
80P6Q-A ..... 0.5 mm-pitch plastic molded LQFP

Memory Expansion Plan



Products under development or planning: the development schedule and specification may be revised without notice.

Fig. 5 Memory expansion plan

Currently planning products are listed below.

Table 3 Support products

As of Feb. 2001

Product name	ROM size (bytes) ROM size for User in ( )	RAM size (bytes)	Package	Remarks
M37641M8-XXXFP	32768 (32636)	1024	80P6N-A	Mask ROM version
M37641M8-XXXHP			80P6Q-A	
M37641F8FP	32768	2560	80P6N-A	Flash memory version
M37641F8HP			80P6Q-A	

FUNCTIONAL DESCRIPTION
CENTRAL PROCESSING UNIT (CPU)

The 7641 group uses the standard 7600 series instruction set. Refer to the 7600 Series Software Manual for details on the instruction set. The 7600 series has an upward compatible instruction set, of which instruction execution cycles are shortened, for 740 series.

[Accumulator (A)]

The accumulator is an 8-bit register. Data operations such as data transfer, etc., are executed mainly through the accumulator.

[Index Register X (X)]

The index register X is an 8-bit register. In the index addressing modes, the value of the OPERAND is added to the contents of register X and specifies the real address.

[Index Register Y (Y)]

The index register Y is an 8-bit register. In partial instruction, the value of the OPERAND is added to the contents of register Y and specifies the real address.

[Stack Pointer (S)]

The stack pointer is an 8-bit register used during subroutine calls and interrupts. This register indicates start address of stored area (stack) for storing registers during subroutine calls and interrupts.
The low-order 8 bits of the stack address are determined by the contents of the stack pointer. The high-order 8 bits of the stack address are determined by the stack page selection bit. If the stack page selection bit is “0” , the high-order 8 bits becomes “00_{16}”. If the stack page selection bit is “1”, the high-order 8 bits becomes “01_{16}”.
The operations of pushing register contents onto the stack and popping them from the stack are shown in Figure 7.
Store registers other than those described in Figure 7 with program when the user needs them during interrupts or subroutine calls.

[Program Counter (PC)]

The program counter is a 16-bit counter consisting of two 8-bit registers PC_H and PC_L. It is used to indicate the address of the next instruction to be executed.

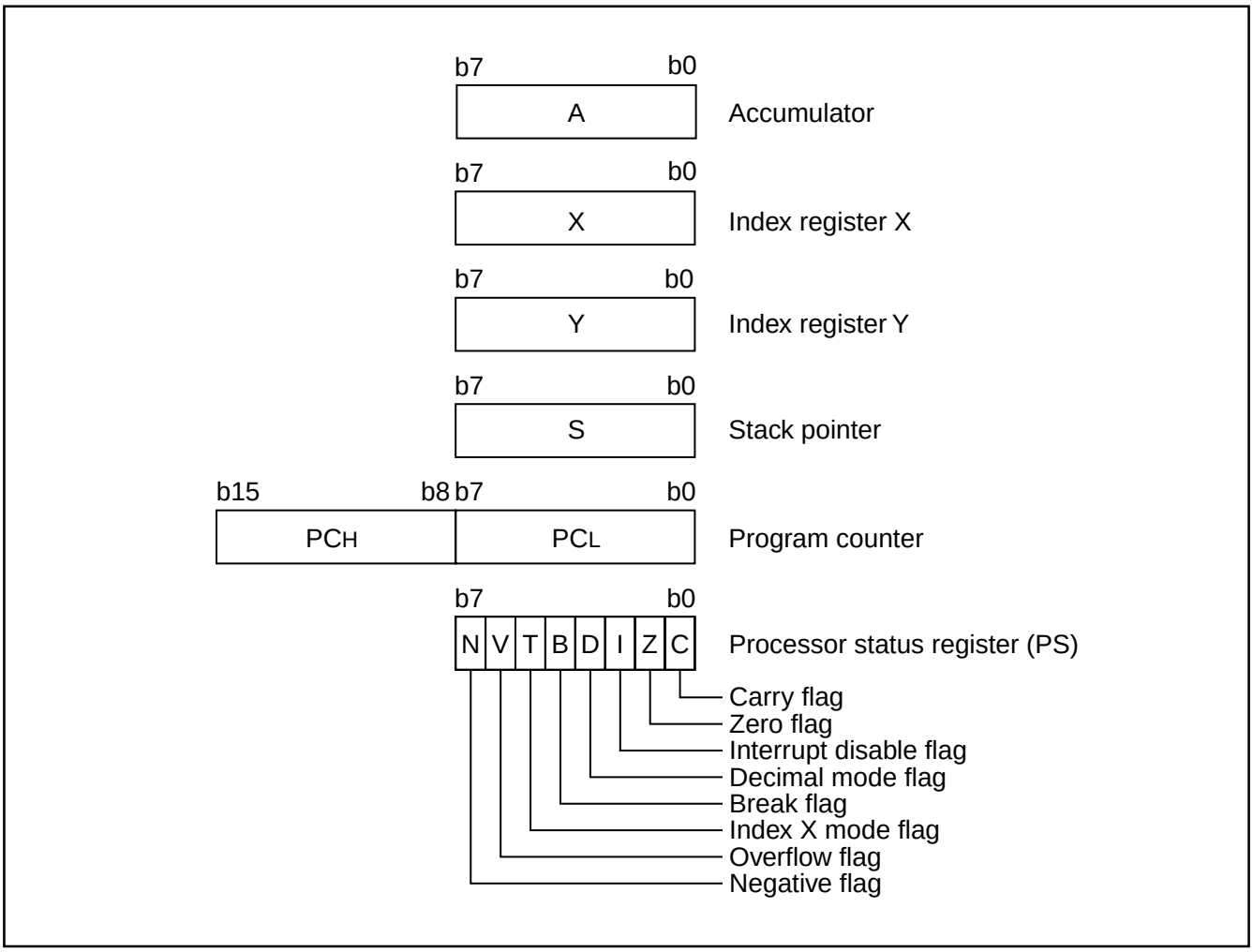


Fig. 6 7600 series CPU register structure

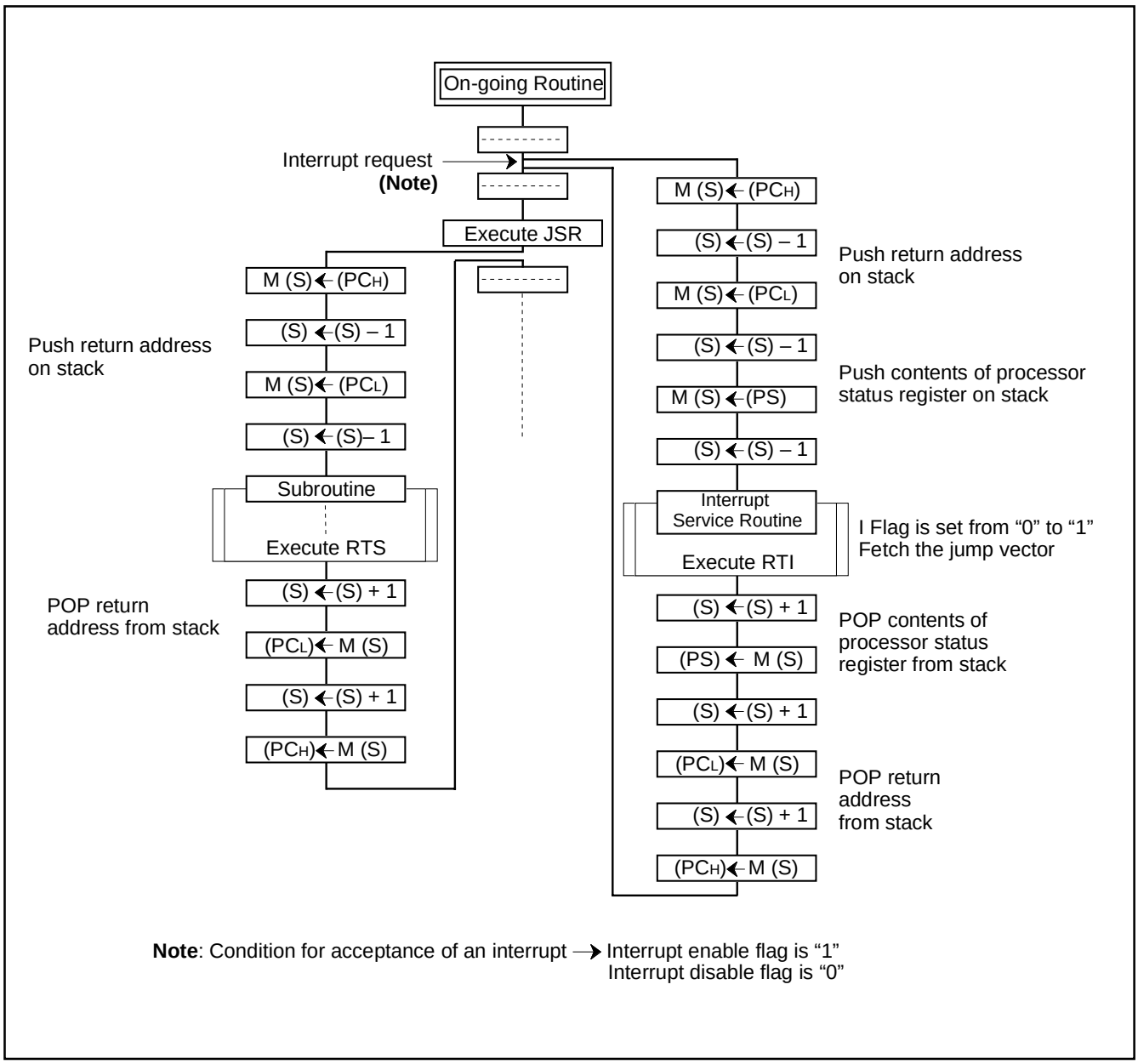


Fig. 7 Register push and pop at interrupt generation and subroutine call

Table 4 Push and pop instructions of accumulator or processor status register

	Push instruction to stack	Pop instruction from stack
Accumulator	PHA	PLA
Processor status register	PHP	PLP

[Processor status register (PS)]

The processor status register is an 8-bit register consisting of 5 flags which indicate the status of the processor after an arithmetic operation and 3 flags which decide MCU operation. Branch operations can be performed by testing the Carry (C) flag , Zero (Z) flag, Overflow (V) flag, or the Negative (N) flag. In decimal mode, the Z, V, N flags are not valid.

•Bit 0: Carry flag (C)

The C flag contains a carry or borrow generated by the arithmetic logic unit (ALU) immediately after an arithmetic operation. It can also be changed by a shift or rotate instruction.

•Bit 1: Zero flag (Z)

The Z flag is set if the result of an immediate arithmetic operation or a data transfer is “0”, and cleared if the result is anything other than “0”.

•Bit 2: Interrupt disable flag (I)

The I flag disables all interrupts except for the interrupt generated by the BRK instruction.

Interrupts are disabled when the I flag is “1”.

•Bit 3: Decimal mode flag (D)

The D flag determines whether additions and subtractions are executed in binary or decimal. Binary arithmetic is executed when this flag is “0”; decimal arithmetic is executed when it is “1”. Decimal correction is automatic in decimal mode. Only the ADC

•Bit 4: Break flag (B)

The B flag is used to indicate that the current interrupt was generated by the BRK instruction. The BRK flag in the processor status register is always “0”. When the BRK instruction is used to generate an interrupt, the processor status register is pushed onto the stack with the break flag set to “1”.

•Bit 5: Index X mode flag (T)

When the T flag is “0”, arithmetic operations are performed between accumulator and memory. When the T flag is “1”, direct arithmetic operations and direct data transfers are enabled between memory locations.

•Bit 6: Overflow flag (V)

The V flag is used during the addition or subtraction of one byte of signed data. It is set if the result exceeds +127 to -128. When the BIT instruction is executed, bit 6 of the memory location operated on by the BIT instruction is stored in the overflow flag.

•Bit 7: Negative flag (N)

The N flag is set if the result of an arithmetic operation or data transfer is negative. When the BIT instruction is executed, bit 7 of the memory location operated on by the BIT instruction is stored in the negative flag.

Table 5 Set and clear instructions of each bit of processor status register

	C flag	Z flag	I flag	D flag	B flag	T flag	V flag	N flag
Set instruction	SEC	–	SEI	SED	–	SET	–	–
Clear instruction	CLC	–	CLI	CLD	–	CLT	CLV	–

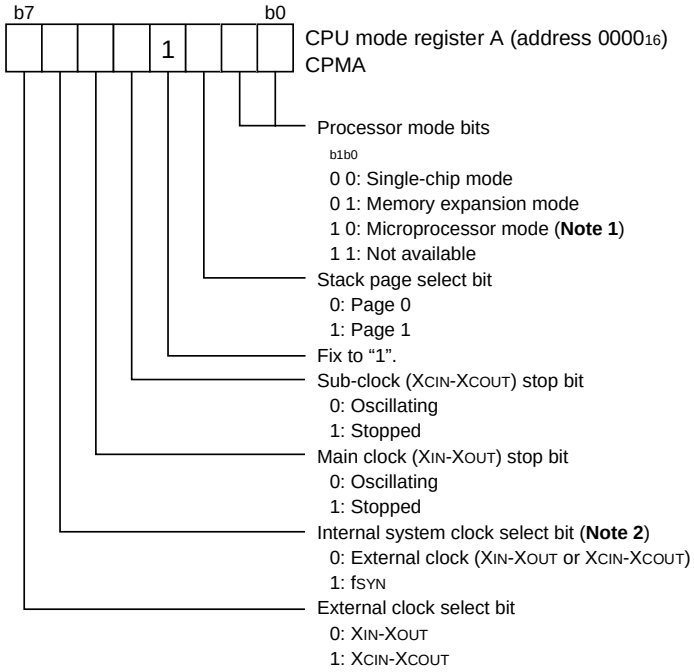
[CPU Mode Registers A, B (CPUMA, CPUMB)] 0000_{16}, 0001_{16}

The CPU mode register contains the stack page select bit and the CPU operating mode select bit and so on.

The CPU mode registers are allocated at address 0000_{16}, 0001_{16}.

■ Notes

Do not use the microprocessor mode in the flash memory version.



Notes 1: This is not available in the flash memory version.

2: When (CPMA 6, 7) = (0, 0), the internal system clock can be selected between f(XIN) or f(XIN)/2 by CCR7.

The internal clock ϕ is the internal system clock divided by 2.

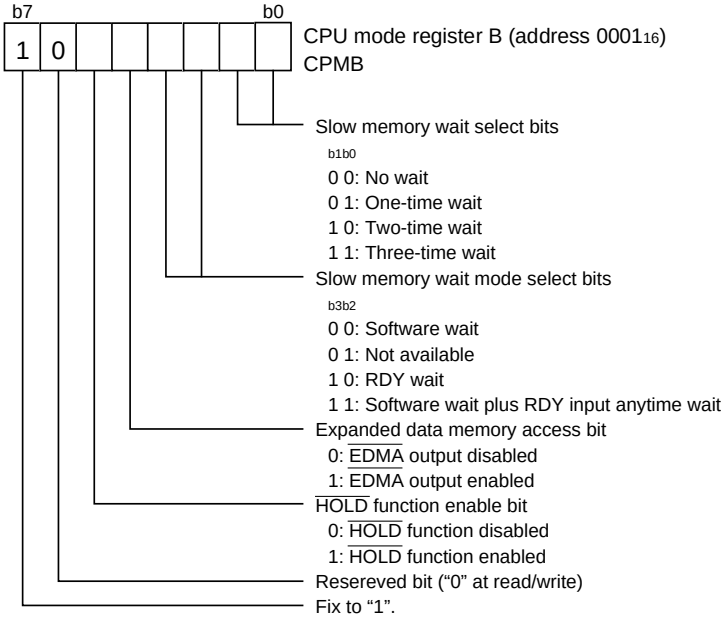


Fig. 8 Structure of CPU mode register

MEMORY

Special Function Register (SFR) Area

The Special Function Register area in the zero page contains control registers such as I/O ports and timers.

RAM

RAM is used for data storage and for stack area of subroutine calls and interrupts.

ROM

The first 128 bytes and the last 4 bytes of ROM are reserved for device testing and the rest is user area for storing programs. In the flash memory version, program and erase can be performed in the reserved area.

Interrupt Vector Area

The interrupt vector area contains reset and interrupt vectors.

Zero Page

Access to this area with only 2 bytes is possible in the zero page addressing mode.

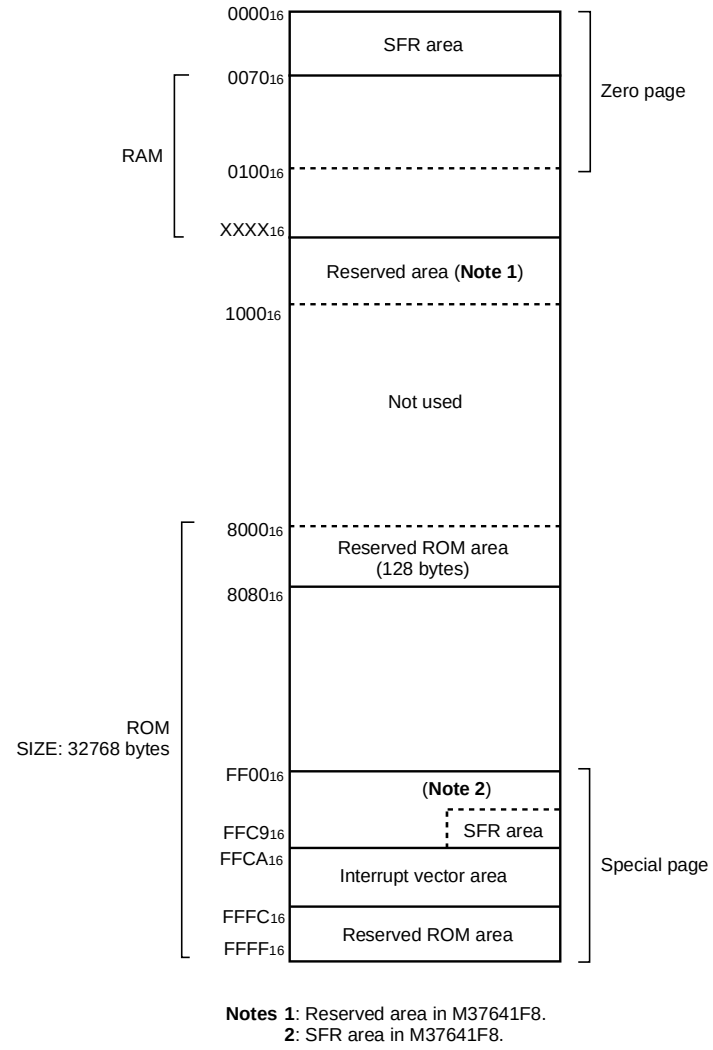
Special Page

Access to this area with only 2 bytes is possible in the special page addressing mode.

Refer to page 74 for the memory map of memory expansion and microprocessor modes.

RAM area

	RAM size (bytes)	Address $XXXX_{16}$
M37641M8	1024	$046F_{16}$
M37641F8	2560	$0A6F_{16}$



Notes 1: Reserved area in M37641F8.
2: SFR area in M37641F8.

Fig. 9 Memory map diagram

0000$_{16}$	CPU mode register A (CPUA)
0001$_{16}$	CPU mode register B (CPUB)
0002$_{16}$	Interrupt request register A (IREQA)
0003$_{16}$	Interrupt request register B (IREQB)
0004$_{16}$	Interrupt request register C (IREQC)
0005$_{16}$	Interrupt control register A (ICONA)
0006$_{16}$	Interrupt control register B (ICONB)
0007$_{16}$	Interrupt control register C (ICONC)
0008$_{16}$	Port P0 (P0)
0009$_{16}$	Port P0 direction register (P0D)
000A$_{16}$	Port P1 (P1)
000B$_{16}$	Port P1 direction register (P1D)
000C$_{16}$	Port P2 (P2)
000D$_{16}$	Port P2 direction register (P2D)
000E$_{16}$	Port P3 (P3)
000F$_{16}$	Port P3 direction register (P3D)
0010$_{16}$	Port control register (PTC)
0011$_{16}$	Interrupt polarity select register (IPOL)
0012$_{16}$	Port P2 pull-up control register (PUP2)
0013$_{16}$	USB control register (USBC)
0014$_{16}$	Port P6 (P6)
0015$_{16}$	Port P6 direction register (P6D)
0016$_{16}$	Port P5 (P5)
0017$_{16}$	Port P5 direction register (P5D)
0018$_{16}$	Port P4 (P4)
0019$_{16}$	Port P4 direction register (P4D)
001A$_{16}$	Port P7 (P7)
001B$_{16}$	Port P7 direction register (P7D)
001C$_{16}$	Port P8 (P8)
001D$_{16}$	Port P8 direction register (P8D)
001E$_{16}$	Resereved (**Note 1**)
001F$_{16}$	Clock control register (CCR)
0020$_{16}$	Timer XL (TXL)
0021$_{16}$	Timer XH (TXH)
0022$_{16}$	Timer YL (TYL)
0023$_{16}$	Timer YH (TYH)
0024$_{16}$	Timer 1 (T1)
0025$_{16}$	Timer 2 (T2)
0026$_{16}$	Timer 3 (T3)
0027$_{16}$	Timer X mode register (TXM)
0028$_{16}$	Timer Y mode register (TYM)
0029$_{16}$	Timer 123 mode register (T123M)
002A$_{16}$	Serial I/O shift register (SIOSHT)
002B$_{16}$	Serial I/O control register 1 (SIOCON1)
002C$_{16}$	Serial I/O control register 2 (SIOCON2)
002D$_{16}$	Special count source generator 1 (SCSG1)
002E$_{16}$	Special count source generator 2 (SCSG2)
002F$_{16}$	Special count source mode register (SCSGM)
0030$_{16}$	UART1 mode register (U1MOD)
0031$_{16}$	UART1 baud rate generator (U1BRG)
0032$_{16}$	UART1 status register (U1STS)
0033$_{16}$	UART1 control register (U1CON)
0034$_{16}$	UART1 transmit/receive buffer register 1 (U1TRB1)
0035$_{16}$	UART1 transmit/receive buffer register 2 (U1TRB2)
0036$_{16}$	UART1 RTS control register (U1RTSC)
0037$_{16}$	Resereved (**Note 1**)
0038$_{16}$	UART2 mode register (U2MOD)
0039$_{16}$	UART2 baud rate generator (U2BRG)
003A$_{16}$	UART2 status register (U2STS)
003B$_{16}$	UART2 control register (U2CON)
003C$_{16}$	UART2 transmit/receive buffer register 1 (U2TRB1)
003D$_{16}$	UART2 transmit/receive buffer register 2 (U2TRB2)
003E$_{16}$	UART2 RTS control register (U2RTSC)
003F$_{16}$	DMAC index and status register (DMAIS)
0040$_{16}$	DMAC channel x mode register 1 (DMAx1)
0041$_{16}$	DMAC channel x mode register 2 (DMAx2)
0042$_{16}$	DMAC channel x source register Low (DMAxSL)
0043$_{16}$	DMAC channel x source register High (DMAxSH)
0044$_{16}$	DMAC channel x destination register Low (DMAxDL)
0045$_{16}$	DMAC channel x destination register High (DMAxDH)
0046$_{16}$	DMAC channel x transfer count register Low (DMAxCL)
0047$_{16}$	DMAC channel x transfer count register High (DMAxCH)
0048$_{16}$	Data bus buffer register 0 (DBB0)
0049$_{16}$	Data bus buffer status register 0 (DBBS0)
004A$_{16}$	Data bus buffer control register 0 (DBBC0)
004B$_{16}$	Resereved (**Note 1**)
004C$_{16}$	Data bus buffer register 1 (DBB1)
004D$_{16}$	Data bus buffer status register 1 (DBBS1)
004E$_{16}$	Data bus buffer control register 1 (DBBC1)
004F$_{16}$	Resereved (**Note 1**)
0050$_{16}$	USB address register (USBA)
0051$_{16}$	USB power management register (USBPM)
0052$_{16}$	USB interrupt status register 1 (USBIS1)
0053$_{16}$	USB interrupt status register 2 (USBIS2)
0054$_{16}$	USB interrupt enable register 1 (USBIE1)
0055$_{16}$	USB interrupt enable register 2 (USBIE2)
0056$_{16}$	USB frame number register Low (USBSOFL)
0057$_{16}$	USB frame number register High (USBSOFH)
0058$_{16}$	USB endpoint index register (USBINDEX)
0059$_{16}$	USB endpoint x IN control register (IN_CSR)
005A$_{16}$	USB endpoint x OUT control register (OUT_CSR)
005B$_{16}$	USB endpoint x IN max. packet size register (IN_MAXP)
005C$_{16}$	USB endpoint x OUT max. packet size register (OUT_MAXP)
005D$_{16}$	USB endpoint x OUT write count register Low (WRT_CNTL)
005E$_{16}$	USB endpoint x OUT write count register High (WRT_CNTH)
005F$_{16}$	USB endpoint FIFO mode register (USBFIFOMR)
0060$_{16}$	USB endpoint 0 FIFO (USBFIFO0)
0061$_{16}$	USB endpoint 1 FIFO (USBFIFO1)
0062$_{16}$	USB endpoint 2 FIFO (USBFIFO2)
0063$_{16}$	USB endpoint 3 FIFO (USBFIFO3)
0064$_{16}$	USB endpoint 4 FIFO (USBFIFO4)
0065$_{16}$	Resereved (**Note 1**)
0066$_{16}$	Resereved (**Note 1**)
0067$_{16}$	Resereved (**Note 1**)
0068$_{16}$	Resereved (**Note 1**)
0069$_{16}$	Resereved (**Note 1**)
006A$_{16}$	Flash memory control register (FMCR) (**Note 2**)
006B$_{16}$	Resereved (**Note 1**)
006C$_{16}$	Frequency synthesizer control register (FSC)
006D$_{16}$	Frequency synthesizer multiply register 1 (FSM1)
006E$_{16}$	Frequency synthesizer multiply register 2 (FSM2)
006F$_{16}$	Frequency synthesizer divide register (FSD)

FFC9$_{16}$	ROM code protect control register (ROMCP) (**Note 3**)

Notes 1: Do not write any data to this addresses, because these areas are reserved.
2: This area is reserved in the mask ROM version.
3: This area is on the ROM in the mask ROM version.

Fig. 10 Memory map of special function register (SFR)

I/O PORTS

Direction Registers

The I/O ports P0–P8 have direction registers which determine the input/output direction of each individual pin. Each bit in a direction register corresponds to one pin, each pin can be set to be input port or output port.

When "0" is written to the bit corresponding to a pin, that pin becomes an input pin. When "1" is written to that bit, that pin becomes an output pin.

If data is read from a pin set to output, the value of the port output latch is read, not the value of the pin itself. Pins set to input are floating. If a pin set to input is written to, only the port output latch is written to and the pin remains floating.

Slew Rate Control

By setting bits 0 to 5 of the port control register (address 0010_{16}) to "1", slew rate control is enabled. VIHL or CMOS level can be used as a port P2 input level; CMOS or TTL level can be used as an input level of master CPU bus interface.

Pull-up Control

By setting the port P2 pull-up control register (address 0012_{16}), pull-up of each pin of port P2 can be controlled with a program.

However, the contents of port P2 pull-up control register do not affect ports programmed as the output ports but as the input ports.

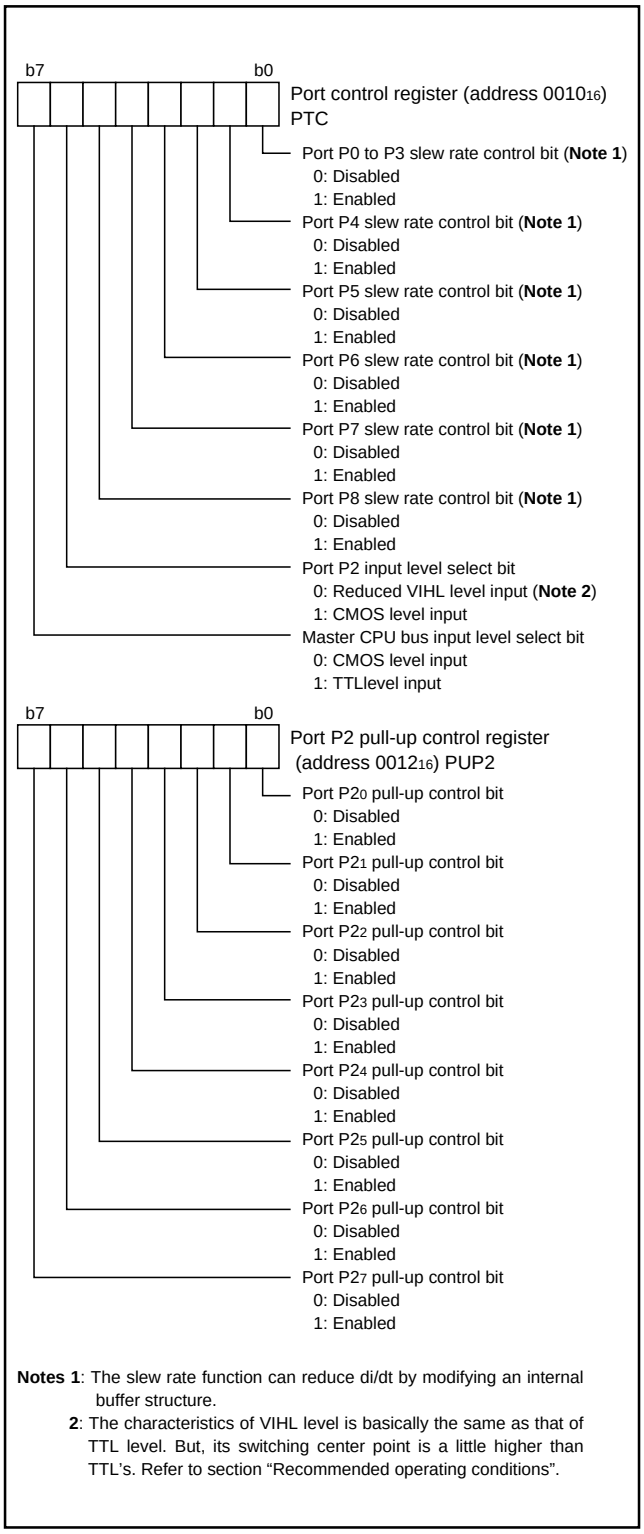


Fig. 11 Structure of port control and port P2 pull-up control registers

Table 6 List of I/O port function

Pin	Name	Input/Output	I/O format	Non-port function	Related SFRs	Ref. No.
P00/AB0–P07/AB7	Port P0	Input/Output, individual bits	CMOS input level CMOS 3-state output	Lower address output	CPU mode register A Port control register	(1)
P10/AB8–P17/AB15	Port P1			Higher address output		
P20/DB0–P27/DB7	Port P2		CMOS input level/VIHL input level CMOS 3-state output	Data bus I/O	CPU mode register A Port control register Port P2 pull-up control register	(2)
P30/RDY–P37/RD	Port P3		CMOS input level CMOS 3-state output	Control signal I/O	CPU mode register A CPU mode register B Port control register	(1) (3)
P40/EDMA,	Port P4					
P41/INT0, P42/INT1,				Control signal I/O External interrupt	CPU mode register A CPU mode register B	(4) (5)
P43/CNTR0, P44/CNTR1					Port control register Timer X mode register Timer Y mode register Interrupt polarity select register	
P50/XCIN, P51/TOUT/XCOUT	Port P5		CMOS input level CMOS 3-state output	Timer 1, Timer 2 output pin Sub-clock generating input pin	CPU mode register A Port control register Clock control register Timer 123 mode register	(6) (7)
P52/OBF0, P53/IBF0, P54/S0, P55/A0, P56/R(E), P57/W(R/W)			CMOS input level CMOS 3-state output CMOS input level/TTL input level in Master CPU bus inferface function	Master CPU bus interface I/O pin	Data bus buffer control register 0 Port control register	(8) (9) (10)
P60/DQ0–P67/DQ7	Port P6		CMOS input level/TTL input level CMOS 3-state output	Master CPU bus interface I/O pin	Data bus buffer control register 0 Port control register	(11)
P70/SOF,	Port P7		CMOS input level CMOS 3-state output	USB function output pin	USB control register Port control register	(12)
P71/HOLD, P72/S1, P73/IBF1/HLDA, P74/OBF1			CMOS input level CMOS 3-state output CMOS input level/TTL input level in Master CPU bus inferface function	Control signal I/O Master CPU bus interface I/O pin	Data bus buffer control register 1 Port control register CPU mode register B	(13) (14) (15) (16)
P80/UTXD2/SRDY, P81/URXD2/SCLK, P82/CTS2/SRXD, P83/RTS2/STXD, P84/UTXD1, P85/URXD1, P86/CTS1, P87/RTS1	Port P8		CMOS input level CMOS 3-state output	Serial I/O I/O pin UART2 I/O pin UART1 I/O pin	UART1, 2 control registers Serial I/O control register 1 Serial I/O control register 2 Port control register	(17) (18) (19) (20) (21) (22) (23) (24)

Notes 1: For details of the ports functions in modes other than single-chip mode, and how to use double-function ports as function I/O ports, refer to the applicable sections.

2: Make sure that the input level at each pin is either 0 V or VCC during execution of the STP instruction.
When an input level is at an intermediate potential, a rush current will flow from VCC to VSS through the input-stage gate.

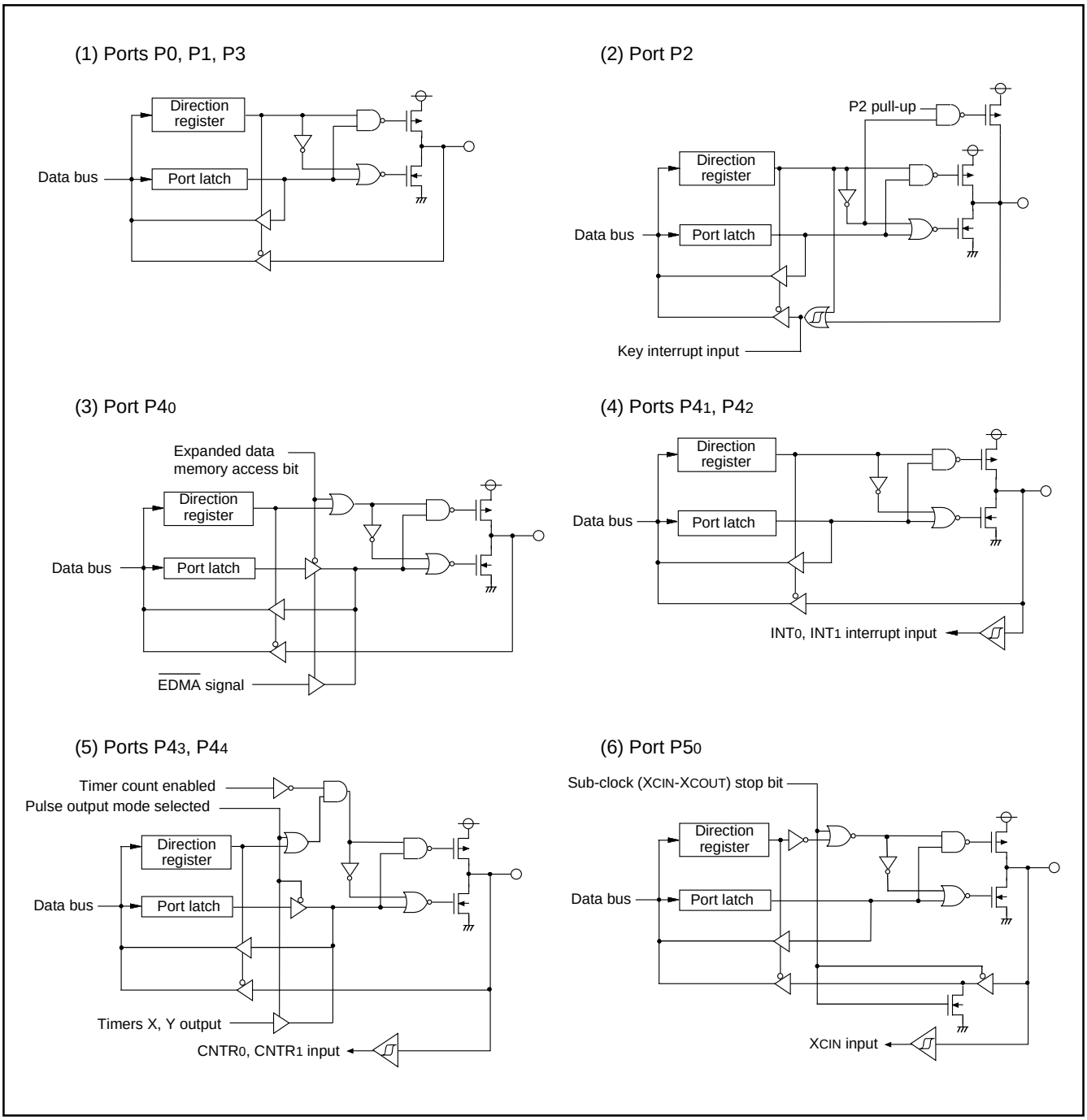
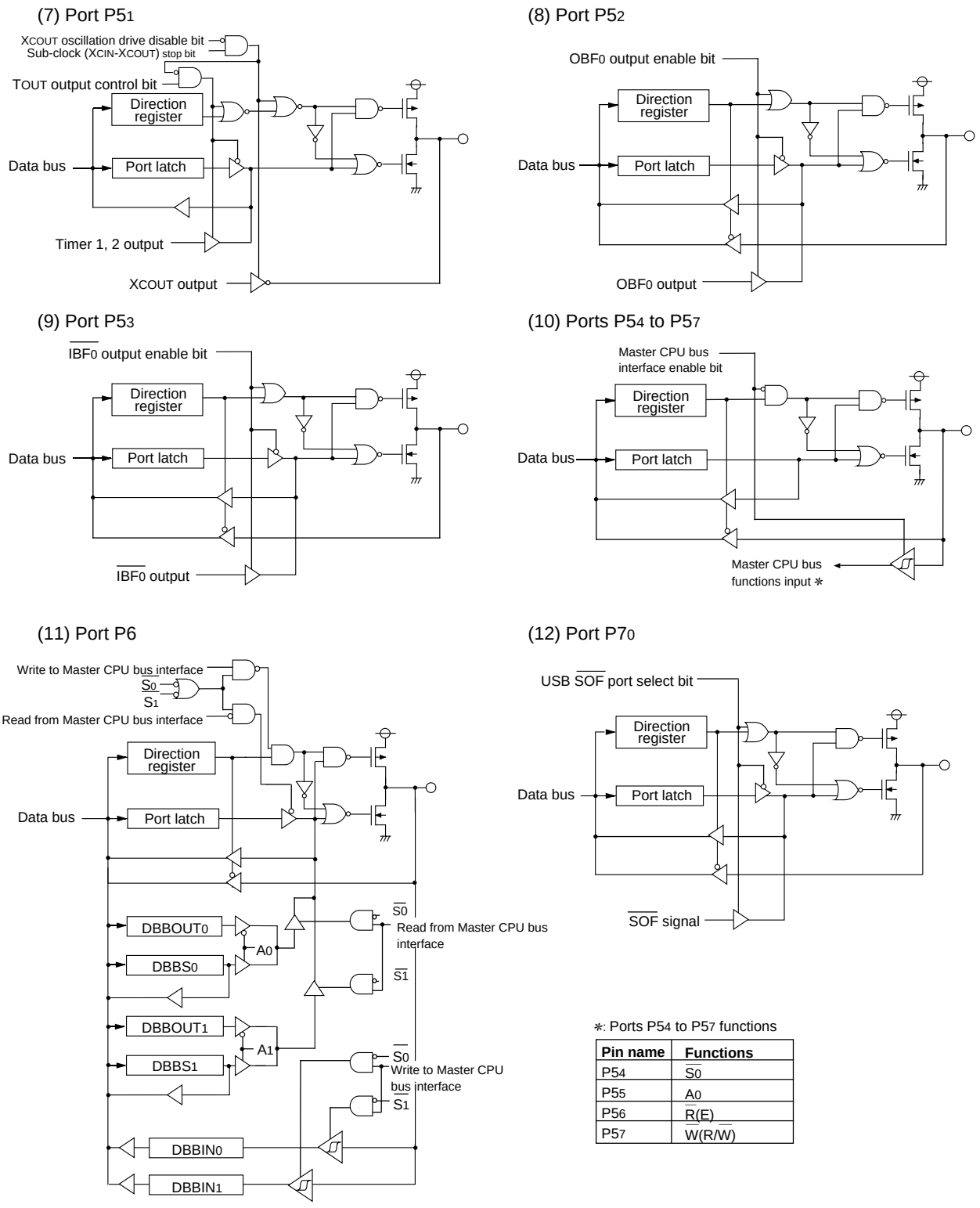


Fig. 12 Port block diagram (1)



*: Ports P54 to P57 functions

Pin name	Functions
P54	$\overline{S0}$
P55	A0
P56	$\overline{R(E)}$
P57	$\overline{W(R/W)}$

Fig. 13 Port block diagram (2)

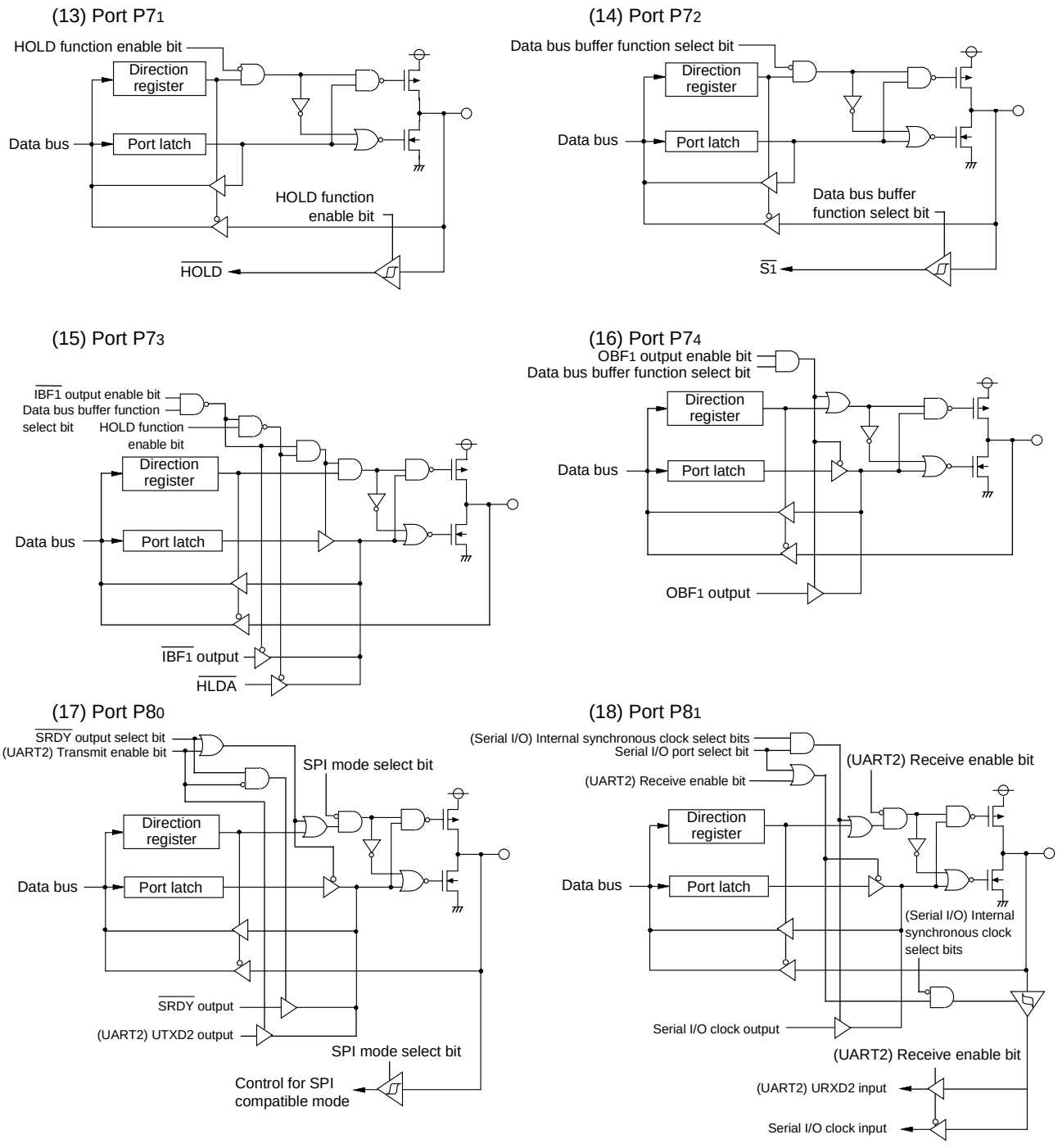


Fig. 14 Port block diagram (3)

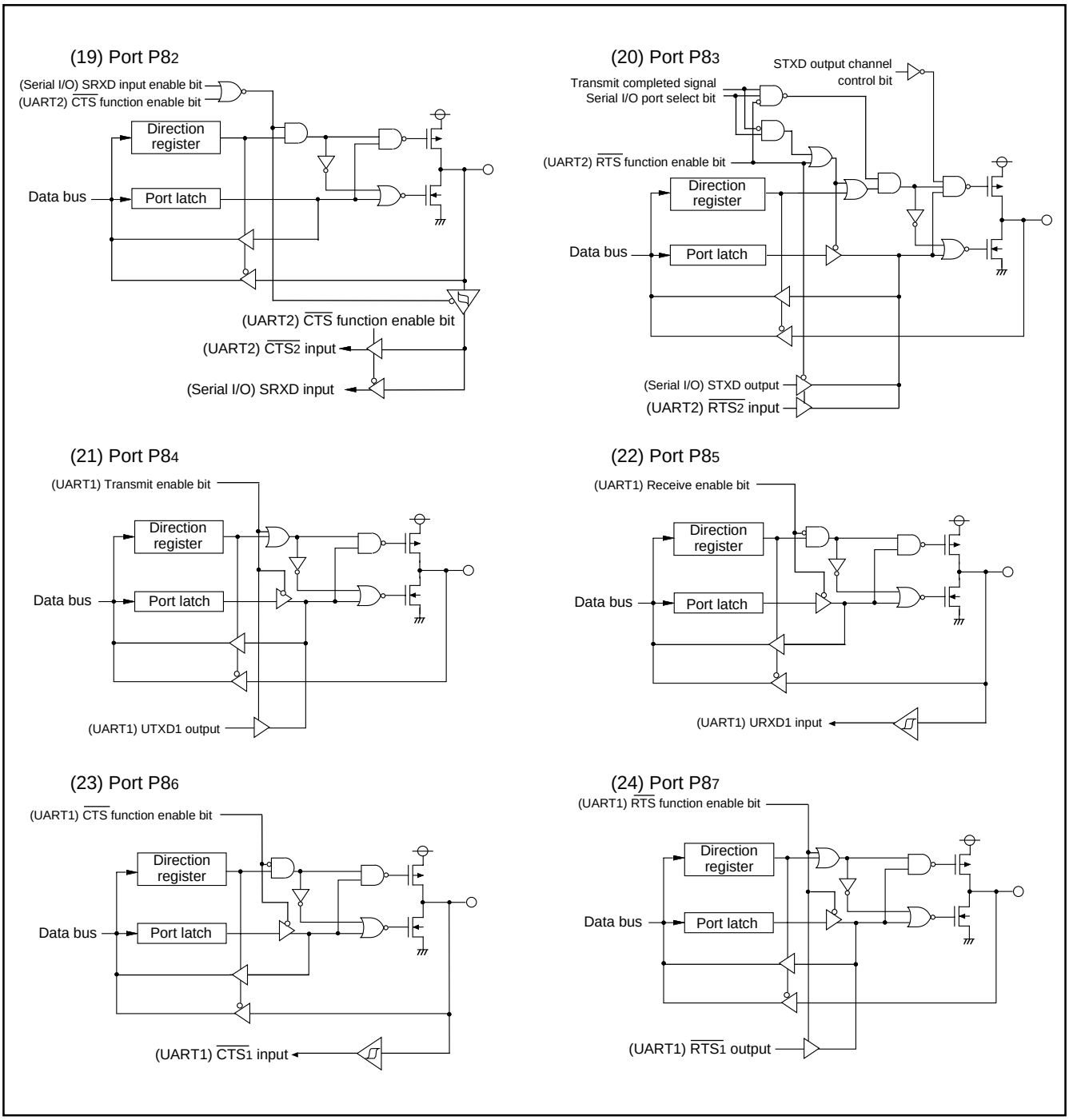


Fig. 15 Port block diagram (4)

INTERRUPTS

There are twenty-four interrupt sources: five externals, eighteen internals, and one software.

Interrupt Control

Each interrupt except the BRK instruction interrupt has both an Interrupt Request Bit and an Interrupt Enable Bit, and is controlled by the Interrupt Disable Flag (I). An interrupt occurs if the corresponding Interrupt Request and Enable Bits are “1” and the Interrupt Disable Flag is “0”.

Interrupt Enable Bits can be set or cleared by software. Interrupt Request Bits can be cleared by software, but cannot be set by software. Additionally, an active edge of INT_1 and INT_2 can be selected by using the interrupt edge select register (address 0011_{16}); an active edge of $CNTR_0$ can be done by using the timer X mode register (address 0027_{16}); an active edge of $CNTR_1$ can be done by using the timer Y mode register (address 0028_{16}).

The BRK instruction interrupt and reset cannot be disabled with any flag or bit. The I Flag disables all interrupts except the BRK instruction interrupt and reset. If several interrupts requests occur at the same time, the interrupt with the highest priority is accepted first.

Interrupt Operation

When an interrupt request occurs, the following operations are automatically performed:

1. The processing being executed is stopped.
2. The contents of the program counter and processor status register are automatically pushed onto the stack.
3. The Interrupt Disable Flag is set and the corresponding interrupt request bit is cleared.
4. The interrupt jump destination address is read from the vector table into the program counter.

■Notes

When setting the followings, the interrupt request bit may be set to “1”.

•When setting external interrupt active edge
Related register: Interrupt polarity select register (address 0011_{16})
Timer X mode register (address 0027_{16})
Timer Y mode register (address 0028_{16})

When not requiring for the interrupt occurrence synchronized with these setting, take the following sequence.

①Set the corresponding Interrupt Enable Bit to “0” (disabled).
②Set the Interrupt Edge Select Bit (Active Edge Switch Bit).
③Set the corresponding Interrupt Request Bit to “0” after 1 or more instructions have been executed.
④Set the corresponding Interrupt Enable Bit to “1” (enabled).

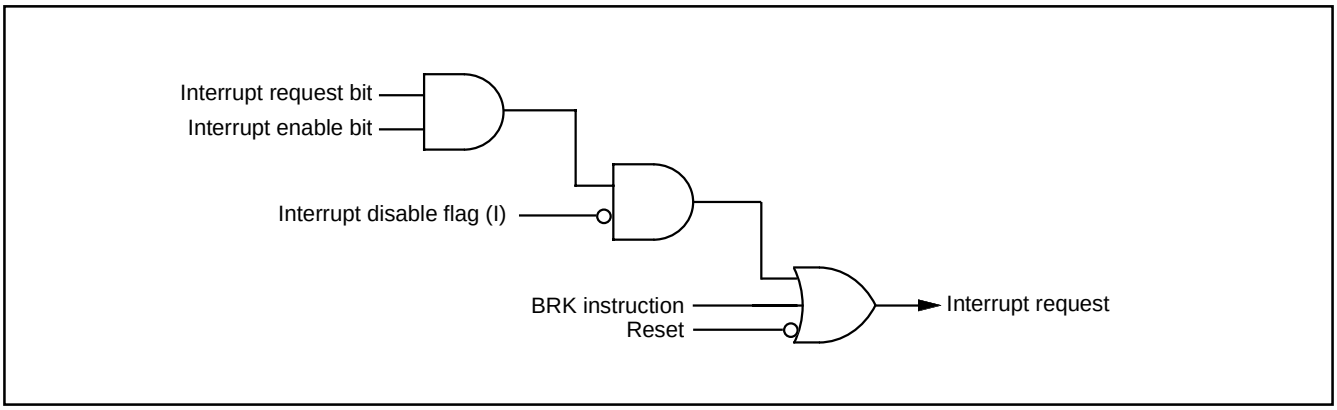


Fig. 16 Interrupt control

Table 7 Interrupt vector addresses and priority

Interrupt Source	Priority	Vector Addresses **(Note 1)**		Interrupt Request Generating Conditions	Remarks
		High	Low		
Reset **(Note 3)**	1	FFFB16	FFFA16	At reset	Non-maskable
USB function	2	FFF916	FFF816	(**Note 2**)	
USB SOF	3	FFF716	FFF616	At reception of SOF packet	
INT0	4	FFF516	FFF416	At detection of either rising or falling edge of INT0 intput	External interrupt (active edge selectable)
INT1	5	FFF316	FFF216	At detection of either rising or falling edge of INT1 input	External interrupt (active edge selectable)
DMAC0	6	FFF116	FFF016	At completion of DMAC0 transfer	
DMAC1	7	FFEF16	FFEE16	At completion of DMAC1 transfer	
UART1 receive buffer full	8	FFED16	FFEC16	At completion of UART1 reception	
UART1 transmit	9	FFEB16	FFEA16	At completion of UART1 transmission	
UART1 summing error	10	FFE916	FFE816	At detection of UART1 summing error	
UART2 receive buffer full	11	FFE716	FFE616	At completion of UART2 reception	
UART2 transmit	12	FFE516	FFE416	At completion of UART2 transmission	
UART2 summing error	13	FFE316	FFE216	At detection of UART2 summing error	
Timer X	14	FFE116	FFE016	At timer X underflow	
Timer Y	15	FFDF16	FFDE16	At timer Y underflow	
Timer 1	16	FFDD16	FFDC16	At timer 1 underflow	
Timer 2	17	FFDB16	FFDA16	At timer 2 underflow	
Timer 3	18	FFD916	FFD816	At timer 3 underflow	
CNTR0	19	FFD716	FFD616	At detection of either rising or falling edge of CNTR0 input	External interrupt (active edge selectable)
CNTR1	20	FFD516	FFD416	At detection of either rising or falling edge of CNTR1 input	External interrupt (active edge selectable)
Serial I/O	21	FFD316	FFD216	At completion of serial I/O transmission/reception	
Input buffer full	22	FFD116	FFD016	At writing to input data bus buffer	
Output buffer empty	23	FFCF16	FFCE16	At reading from output data bus buffer	
Key input (Key-on wake-up)	24	FFCD16	FFCC16	At falling of port P2 input logical level AND	External interrupt (falling valid)
BRK instruction	25	FFCB16	FFCA16	At BRK instruction execution	Non-maskable software interrupt

Notes 1: Vector addresses contain interrupt jump destination addresses.
2: USB function interrupt occurs owing to an interrupt request of the endpoint x (x = 0 to 4) IN, endpoint x OUT, overrun/underrun, USB reset or suspend/resume.
3: Reset functions in the same way as an interrupt with the highest priority.

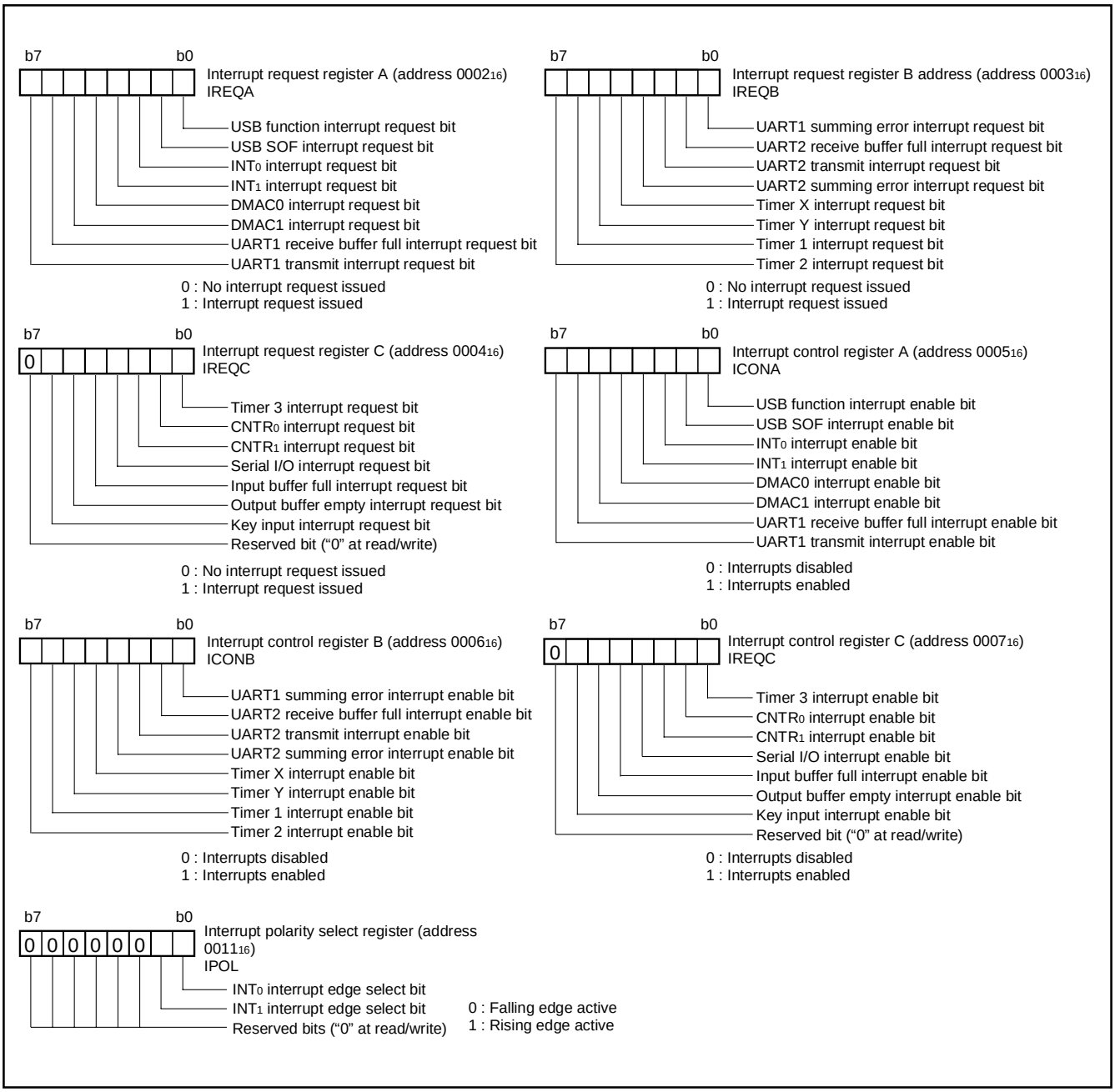


Fig. 17 Structure of interrupt-related registers

Key Input Interrupt (Key-on Wake-Up)

A key input interrupt request is generated by applying “L” level to any pin of port P2 that have been set to input mode. In other words, it is generated when AND of input level goes from “1” to “0”. An example of using a key input interrupt is shown in Figure 18, where an interrupt request is generated by pressing one of the keys consisted as an active-low key matrix which inputs to ports P2$_0$–P2$_4$.

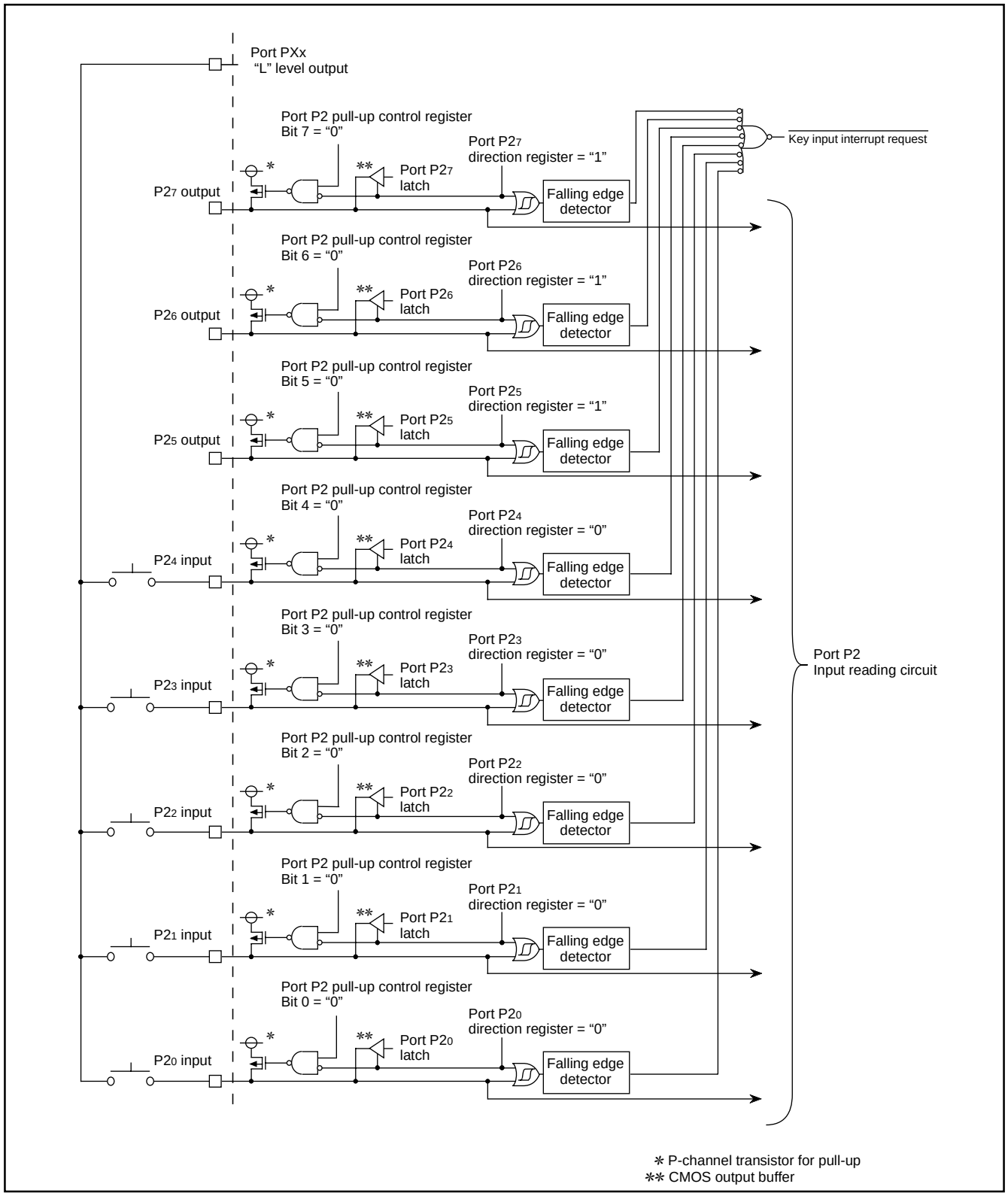


Fig. 18 Connection example when using key input interrupt and port P2 block diagram

TIMERS

The 7641 group has five timers: timer X, timer Y, timer 1, timer 2, and timer 3. Timer X and timer Y are 16-bit timers, and timer 1, timer 2, and timer 3 are 8-bit timers.

All timers are down count timers. When the timer reaches "00_{16}" or "0000_{16}", an underflow occurs at the next count pulse and the corresponding timer latch is reloaded into the timer and the count is continued. When a timer underflows, the interrupt request bit corresponding to that timer is set to "1".

Read and write operation on 16-bit timer must be performed for both high and low-order bytes. When reading a 16-bit timer, read the high-order byte first. When writing to a 16-bit timer, write the low-order byte first. The 16-bit timer cannot perform the correct operation when reading during the write operation, or when writing during the read operation.

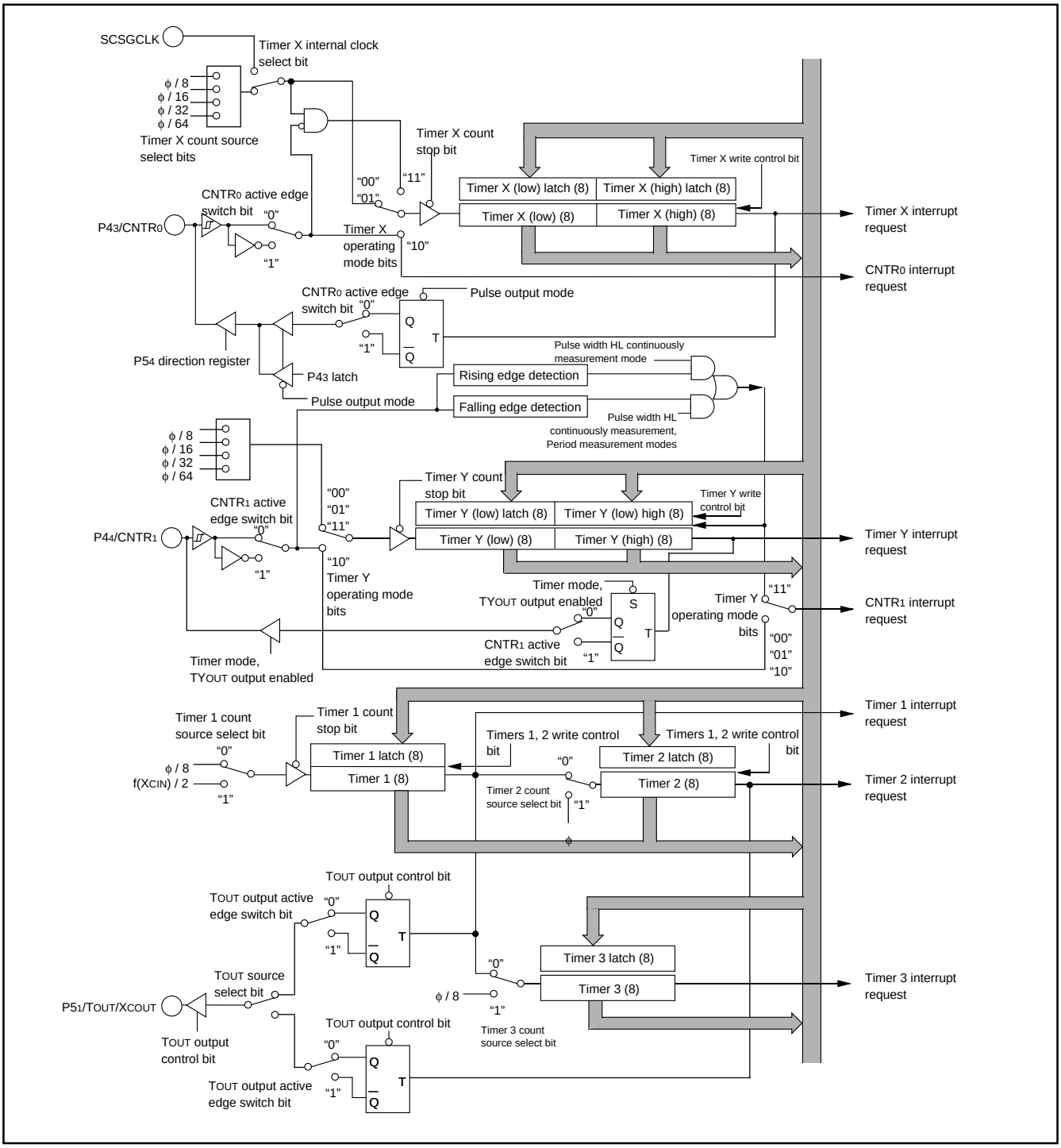


Fig. 19 Timer block diagram

Timer X

Timer X is a 16-bit timer that can be selected in one of four modes. The timer X's internal clock and count source can be selected and a write control is possible by using the timer X mode register.
In all modes the count operation can halt by setting the Timer X Count Stop Bit to "1". Additionally, each timer underflow sets the Interrupt Request Bit to "1".

(1) Timer Mode

The timer counts the SCSGCLK (Special Count Source Generator) or one of the internal clock ϕ divided by 8, 16, 32, 64.

(2) Pulse Output Mode

Each time the timer underflows, a signal output from the $CNTR_0$ pin is inverted. Except for this, the operation in pulse output mode is the same as in timer mode.
When the $CNTR_0$ Active Edge Switch Bit is "0", the $CNTR_0$ pin starts pulses output beginning at "H"; when this bit is "1", the $CNTR_0$ pin starts pulses output beginning at "L".
When using a timer in this mode, set the port $P4_3$ direction register to output mode.

(3) Event Counter Mode

The timer counts signals input through the $CNTR_0$ pin.
Except for this, the operation in event counter mode is the same as in timer mode.
When the $CNTR_0$ Active Edge Switch Bit is "0", the rising edge is counted; when this bit is "1", the falling edge is counted.
When using a timer in this mode, set the port $P4_3$ direction register to input mode.

(4) Pulse Width Measurement Mode

When the $CNTR_0$ Active Edge Switch Bit is "0", the timer counts while the input signal of $CNTR_0$ pin is at "H"; when it is "1", the timer counts while the input signal of $CNTR_0$ pin is at "L".
The timer counts the SCSGCLK or one of the internal clock ϕ divided by 8, 16, 32, 64 as its count source.
When using a timer in this mode, set the port $P4_3$ direction register to input mode.

■ Notes

● Timer X Write Control

If the Timer X Write Control Bit is "1", when the value is written in the address of timer X, the value is loaded only in the latch. The value in the latch is loaded in timer X after timer X underflows.
If the Timer X Write Control Bit is "0", when the value is written in the address of timer X, the value is loaded in the timer X and the latch at the same time.
When the value is to be written in latch only, unexpected value may be set in the high-order timer if the writing in high-order latch and the underflow of timer X are performed at the same timing.

● $CNTR_0$ Interrupt Active Edge Selection

The $CNTR_0$ interrupt active edge depends on the selection of $CNTR_0$ Active Edge Switch Bit.

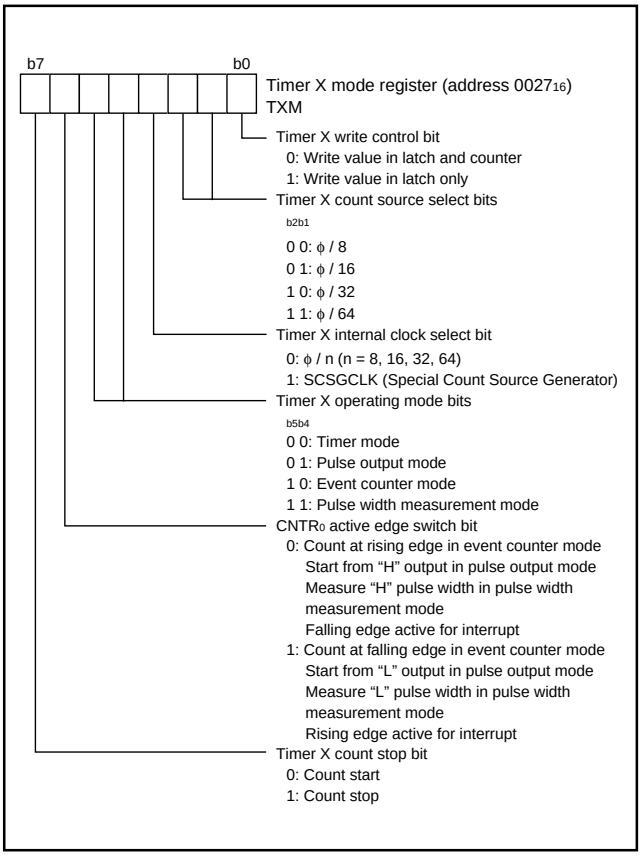


Fig. 20 Structure of timer X mode register

Timer Y

Timer Y is a 16-bit timer that can be selected in one of four modes.

(1) Timer Mode

The timer counts one of the internal clock ϕ divided by 8, 16, 32, 64.

● TYOUT Output Function

In the timer mode, a signal of which polarity is inverted each time the timer underflows is output from the CNTR1 pin. This is enabled by setting the Timer Y Output Control Bit to "1".

When the CNTR1 Active Edge Switch Bit is "0", the CNTR1 pin starts pulses output beginning at "H"; when this bit is "1", the CNTR1 pin starts pulses output beginning at "L".

When using a timer in this mode, set the port P44 direction register to output mode.

(2) Period Measurement Mode

CNTR1 interrupt request is generated at a rising/falling edge of CNTR1 pin input signal. Simultaneously, the value in timer Y latch is reloaded in timer Y and timer Y continues counting down. Except for the aforementioned operation, the operation in period measurement mode is the same as in timer mode. (The TYOUT output function is not usable.)

The timer value just before the reloading at rising/falling of CNTR1 pin input signal is retained until the timer Y is read once after the reload.

The rising/falling timing of CNTR1 pin input signal is found by CNTR1 interrupt.

When the CNTR1 Active Edge Switch Bit is "0", the falling edge is detected; when this bit is "1", the rising edge is detected.

When using a timer in this mode, set the port P44 direction register to input mode.

(3) Event Counter Mode

The timer counts signals input through the CNTR1 pin.

Except for this, the operation in event counter mode is the same as in timer mode. (The TYOUT output function is not usable.)

When the CNTR1 Active Edge Switch Bit is "0", the rising edge is counted; when this bit is "1", the falling edge is counted.

When using a timer in this mode, set the port P44 direction register to input mode.

(4) Pulse Width HL Continuously Measurement Mode

CNTR1 interrupt request is generated at both rising and falling edges of CNTR1 pin input signal. Except for this, the operation in pulse width HL continuously measurement mode is the same as in period measurement mode.

When using a timer in this mode, set the port P44 direction register to input mode.

■ Notes

● Timer Y Write Control

If the Timer Y Write Control Bit is "1", when the value is written in the address of timer Y, the value is loaded only in the latch. The value in the latch is loaded in timer Y after timer Y underflows.

If the Timer Y Write Control Bit is "0", when the value is written in the address of timer Y, the value is loaded in the timer Y and the latch at the same time.

When the value is to be written in latch only, unexpected value may be set in the high-order timer if the writing in high-order latch and the underflow of timer Y are performed at the same timing.

● CNTR1 Interrupt Active Edge Selection

The CNTR1 interrupt active edge depends on the selection of CNTR1 Active Edge Switch Bit.

However, in pulse width HL continuously measurement mode, CNTR1 interrupt request is generated at both rising and falling edges of CNTR1 pin input signal regardless of the setting of CNTR1 Active Edge Switch Bit.

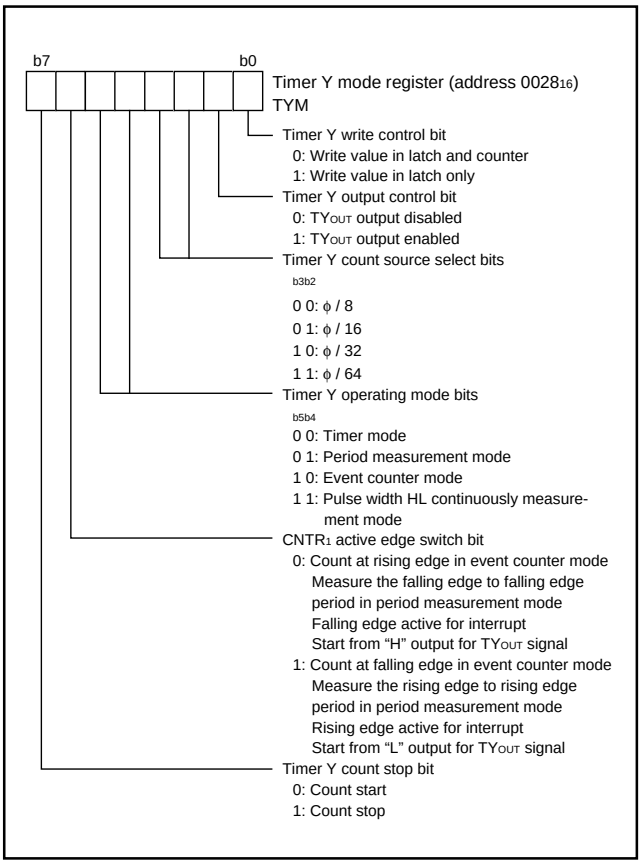


Fig. 21 Structure of timer Y mode register

Timer 1, Timer 2, Timer 3

Timer 1, timer 2, and timer 3 are 8-bit timers. The count source for each timer can be selected by timer 123 mode register.

● Timers 1, 2 Write Control

When the Timers 1, 2 Write Control Bit is "1" and the values are written in the address of timers 1 and 2, the values are loaded only in their latches. The values in the latches are loaded in timers 1 and 2 after timers 1 and 2 underflow.

When the Timers 1, 2 Write Control Bit is "0" and the values are written in the address of timers 1 and 2, the values are loaded in the timers 1 and 2 and their latches at the same time.

● Timers 1, 2 Output Control

A signal of which polarity is inverted each time the timer selected by the TOUT Factor Select Bit underflows is output from the TOUT pin. This is enabled by setting the TOUT Output Control Bit to "1".

When the TOUT Output Active Edge Switch Bit is "0", the TOUT pin starts pulses output beginning at "H"; when this bit is "1", the TOUT pin starts pulses output beginning at "L".

When using a timer in this mode, set the port P5$_1$ direction register to output mode.

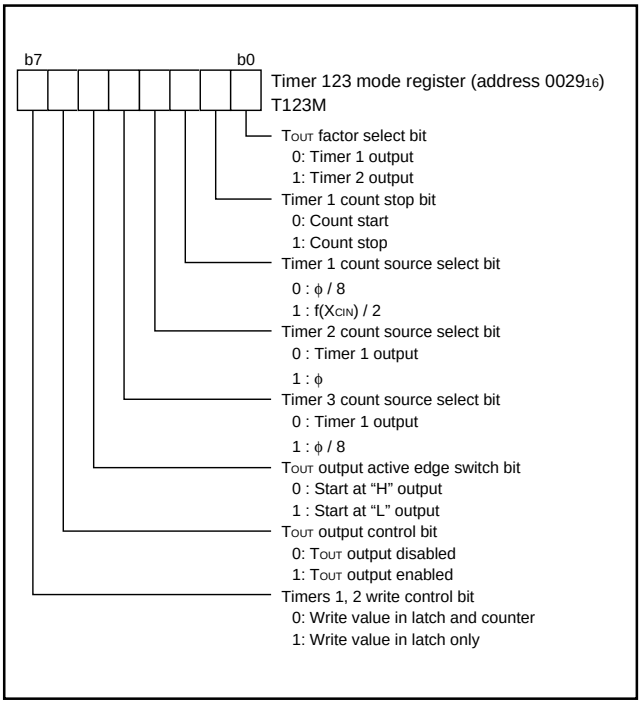


Fig. 22 Structure of timer 123 mode register

■ Notes

● Timer 1 to Timer 3

Switching of the count sources of timers 1 to 3 does not affect the values of reload latches. However, that may make count operation started. Therefore, write values again in the order of timers 1, 2 and then timer 3 after their count sources have been switched.

● Timers 1, 2 Write Control

When the value is to be written in latch only, unexpected value may be set in the timer if the writing in the latch and the timer underflow are performed at the same timing.

SERIAL I/O

The serial I/O can be used only for clock synchronous serial I/O. The transmitter and the receiver must use the same clock. If the internal clock is used, transfer is started by a write signal to the serial I/O shift register.

[Serial I/O Control Register 1 (SIOCON1)] 002B16
[Serial I/O Control Register 2 (SIOCON2)] 002C16
Each of the serial I/O control registers 1 and 2 contains eight bits which control various serial I/O functions.

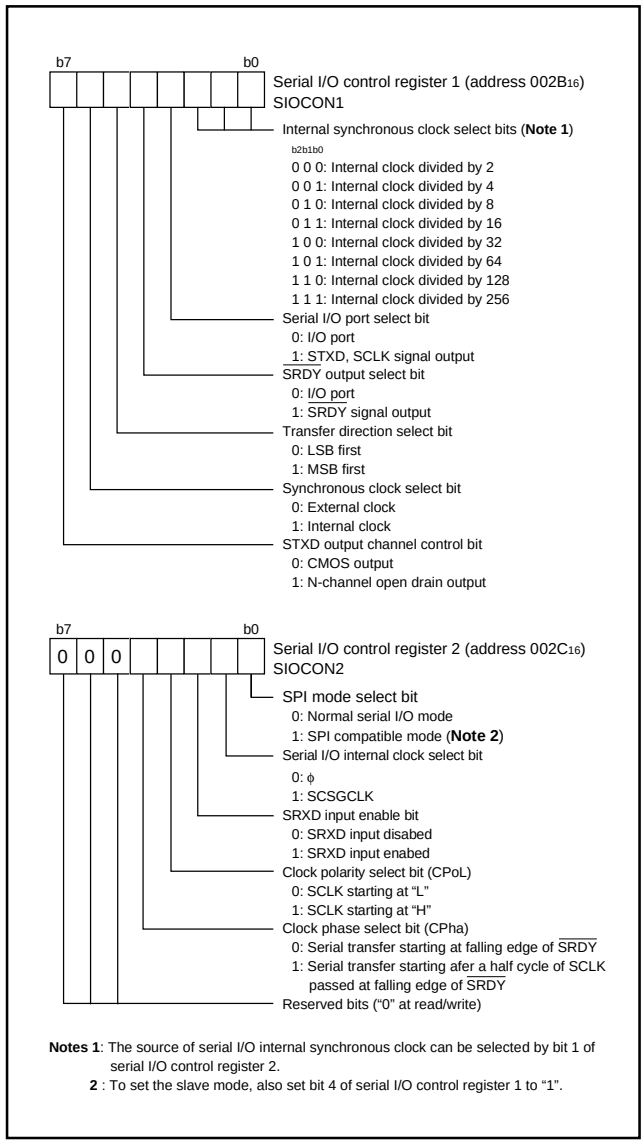


Fig. 23 Structure of serial I/O control registers 1, 2

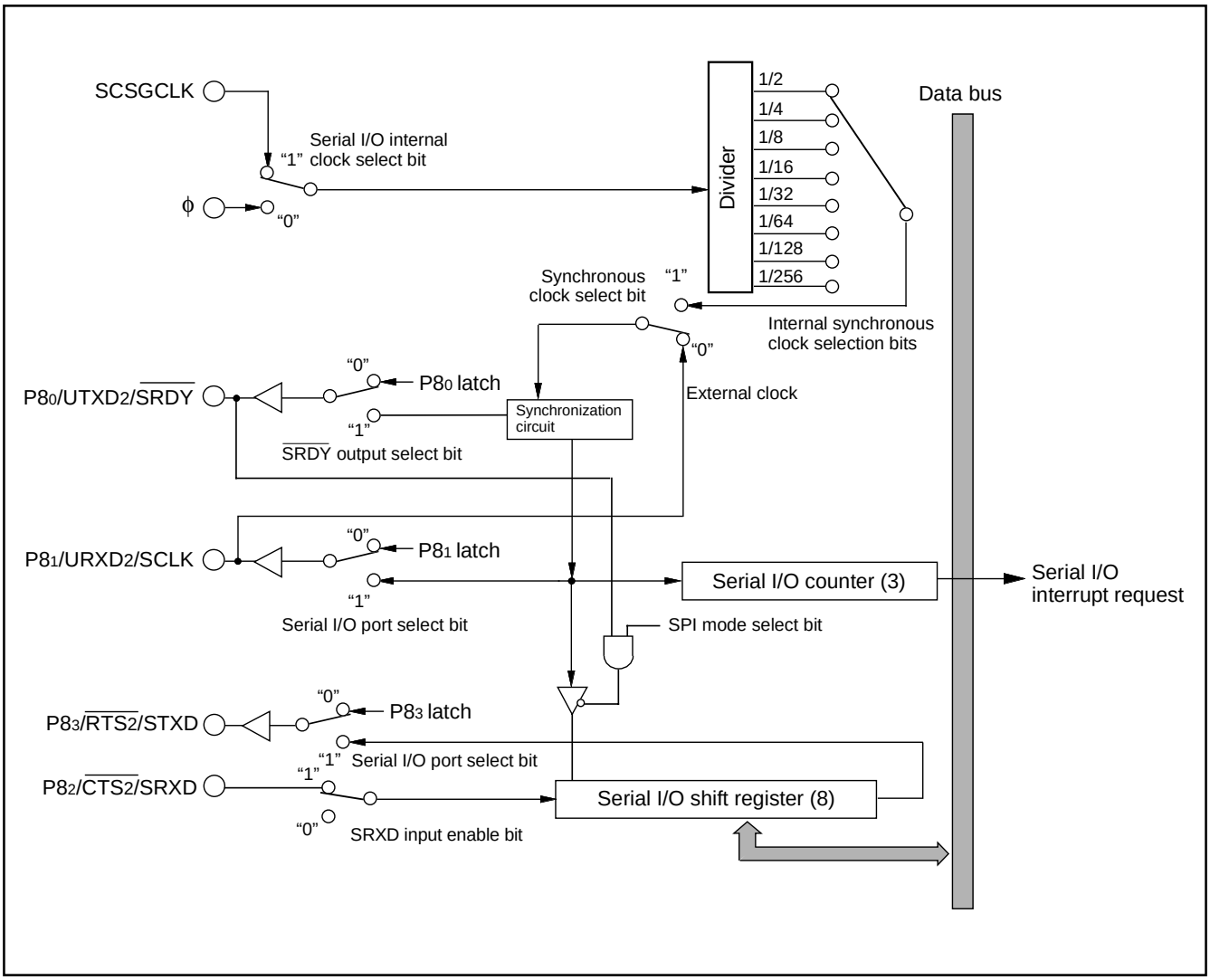


Fig. 24 Block diagram of serial I/O

Serial I/O Normal Operation

The serial I/O counter is set to “7” by writing operation to the serial I/O shift register (address 002A16). When the SRDY Output Select bit is “1”, the SRDY pin goes “L” after that writing. On the negative edge of the transfer clock the SRDY pin returns “H” and the data of the first bit is transmitted from the STXD pin. The remaining data are done from the STXD pin bit by bit on each falling edge of the transfer clock.

Additionally, the data is latched from the SRXD pin on each rising edge of the transfer clock and then the contents of the serial I/O shift register are shifted by one bit.

When the internal system clock is selected as the transfer clock, the followings occur at counting eight transfer clocks:

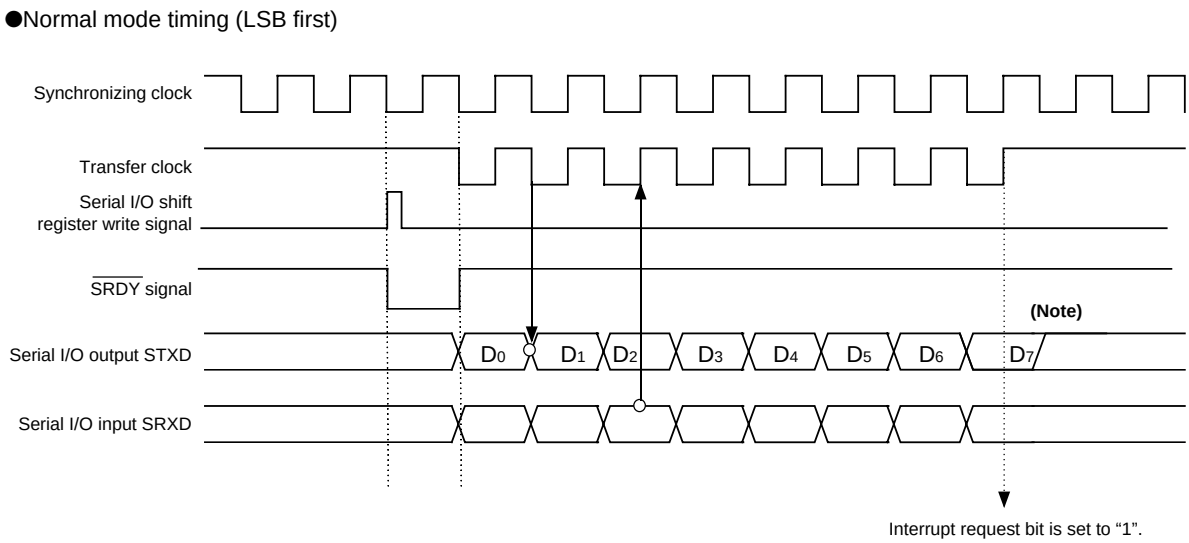
- The serial I/O counter reaches “0”
- The transfer clock halts at “H”
- The serial I/O interrupt request bit is set to “1”
- The STXD pin goes a high-impedance state after an 8-bit transfer is completed.

When the external clock is selected as the transfer clock, the followings occur at counting eight transfer clocks:

- The serial I/O counter reaches “0”
- The serial I/O interrupt request bit is set to “1”

In this case, the transfer clock needs to be controlled by the external source because the transfer clock does not halt. Additionally, the STXD pin does not go a high-impedance state after an 8-bit transfer is completed.

Figure 25 shows serial I/O timing.



Note: When the internal clock is selected as the transfer clock, the STXD pin goes to a high-impedance state after transfer completion.

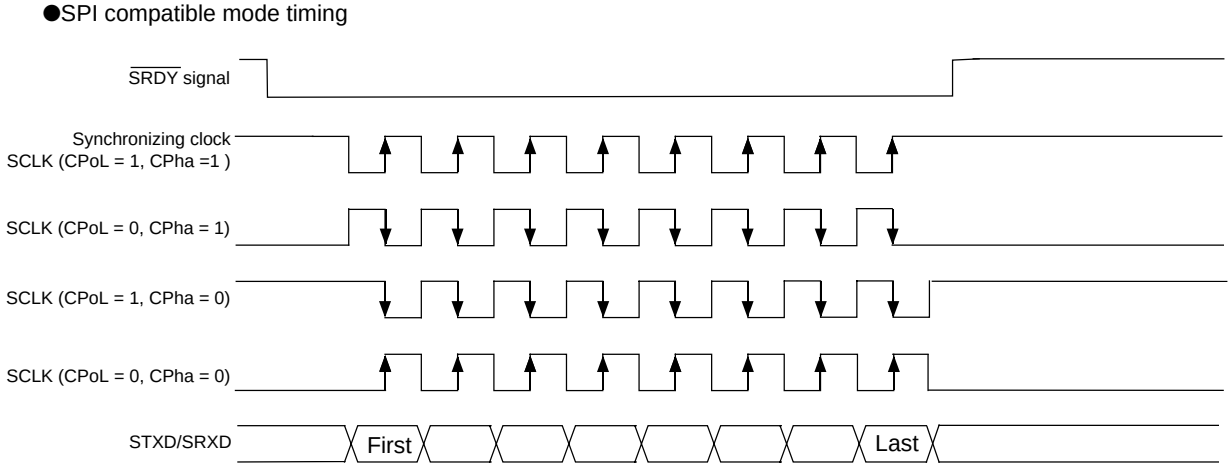


Fig. 25 Serial I/O timing

SPI Compatible Mode Operation

Setting the SPI Mode Select Bit (bit 0 of SIOCON2) puts the serial I/O in SPI compatible mode. The Synchronous Clock Select Bit (bit 6 of SIOCON1) determines whether the serial I/O is an SPI master or slave. When the external clock is selected (“0”), the serial I/O is in slave mode; When the internal clock is selected (“1”), the serial I/O is in master mode.

In SPI compatible mode the SRXD pin functions as a MISO (Master In/Slave Out) pin and the STXD pin functions as a MOSI (Master Out/Slave In) pin.

In slave mode the transmit data is output from the MISO pin and the receive data is input from the MISO pin. The $\overline{\text{SRD}}$ pin functions as the chip-select signal input pin from an external.

In master mode the transmit data is output from the MOSI pin and the receive data is input from the MISO pin. The $\overline{\text{SRD}}$ pin functions as the chip-select signal output pin to an external.

●Slave Mode Operation

In slave mode of SPI compatible mode 4 types of clock polarity and clock phase can be usable by bits 3 and 4 of serial I/O control register 2.

If the $\overline{\text{SRDY}}$ pin is held “H”, the shift clock is inhibited, the serial I/O counter is set to “7”. If the $\overline{\text{SRDY}}$ pin is held “L”, then the shift clock will start.

Make sure during transfer to maintain the $\overline{\text{SRDY}}$ input at “L” and not to write data to the serial I/O counter.

Figure 25 shows the serial I/O timing.

UART1, UART2

The UART consists of two channels: UART1 and UART2. Each has a dedicated timer provided to generate transfer clocks and operates independently. Both UART1 and UART2 have the same functions.

Twelve serial data transfer formats can be selected, and the transfer formats used by a transmitter and receiver must be identical.

The transmit and receive shift registers each have a buffer, but the two buffers have the same address in a memory. Since the shift register cannot be written to or read from directly, transmit data is written to the transmit buffer register, and receive data is read from the receive buffer register.

The transmit buffer register can also hold the next data to be transmitted, and the receive buffer register can hold a character while the next character is being received.

The transfer speed (baud rate) is expression as follows:

Transfer speed (baud rate) = fi / {(n + 1) × 16 }

n: The contents of UARTx (x = 1, 2) baud rate generator

fi: Using UART clock prescaling select bits, select any one of ϕ, ϕ/8, ϕ/32, ϕ/256, SCSGCLK, SCSGCLK/8, SCSGCLK/32 and SCSGCLK/256

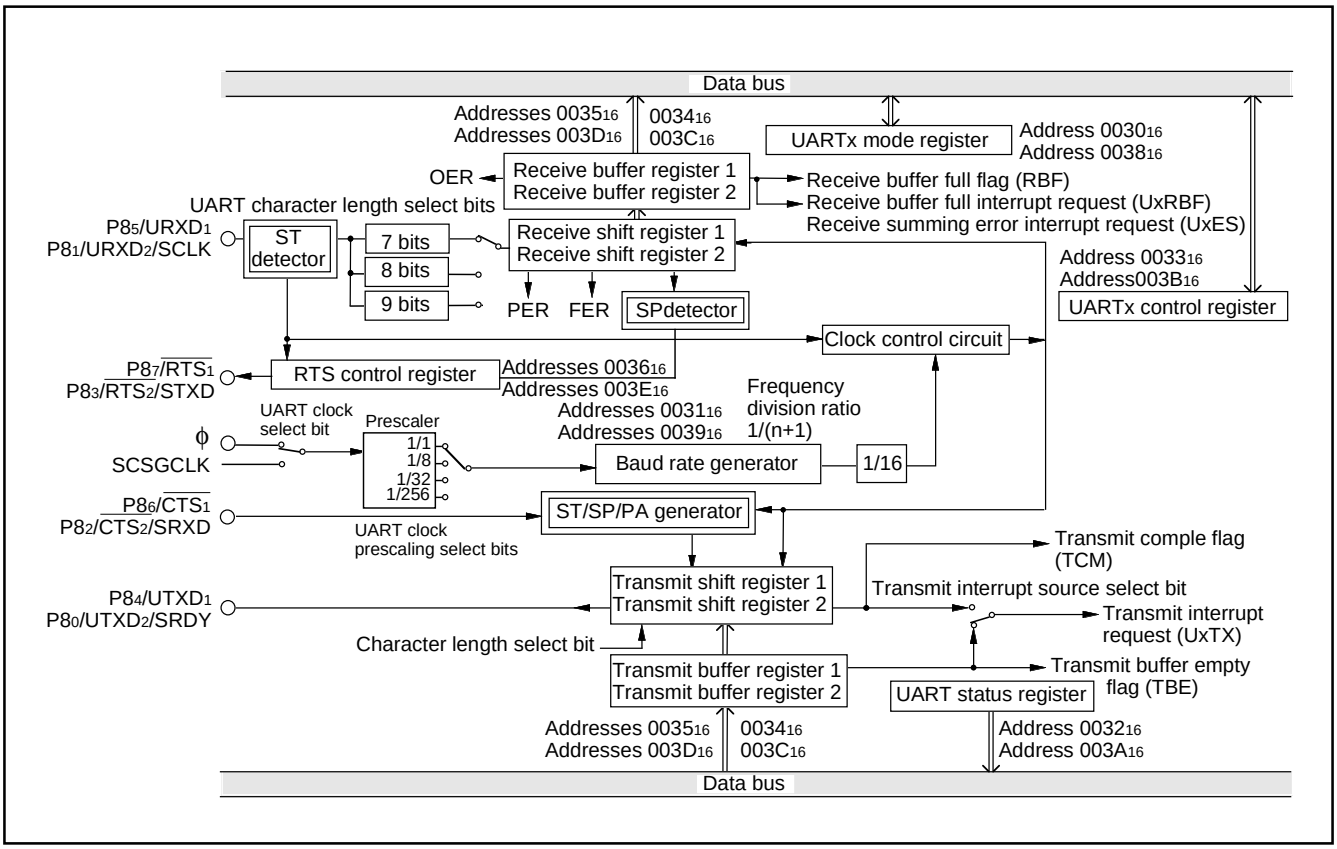


Fig. 26 UARTx (x = 1, 2) block diagram

UART Transmit Operation

Transmission starts when the Transmit Enable Bit is "1" and the Transmit Buffer Empty Flag is "0". Additionally, when $\overline{CTS}$ function enabled, the $\overline{CTS}$x pin must be "L" to be started. The data in which Start Bit and Stop Bit or Parity Bit are also added is transmitted from the low-order byte sequentially. When using 9-bit character length, set the data into the UARTx transmit buffer register 2 (high-order byte) first before the UARTx transmit buffer register 1 (low-order byte).

Once the transmission starts, the Transmit Enable Bit, the Transmit Buffer Empty Flag and the $\overline{CTS}$x pin state (when this is enabled) could not be checked until the transmission in progress has ended.

Transmission requires the following setup:

(1) Define a baud rate by setting a value n (n = 0 to 255) into UARTx baud rate generator (addresses 0031_{16}, 0039_{16}).
(2) Set the Transmit Initialization Bit (bit 2 of UxCON) to "1". This will set the UARTx status register to "03_{16}".
(3) Select the interrupt source with the Transmit Interrupt Source Select Bit (bit 4 of UxCON).
(4) Configure the data format and clock selection by setting the UARTx mode register.
(5) Set the $\overline{CTS}$ Function Enable Bit (bit 5 of UxCON) if $\overline{CTS}$ function will be used.
(6) Set the Transmit Enable Bit (bit 0 of UxCON) to "1".

If updating a value of UARTx baud rate generator while the data is being transmitted, be sure to disable the transmission before updating. If the former data remains in the UARTx transmit buffer registers 1 and 2 at retransmission, an undefined data might be output.

UART Receive Operation

Reception is enabled when the Receive Enable Bit is "1". Detection of the start bit makes transfer clocks generated and the data reception starts in the LSB first.

When using 9-bit character length, read the received data from the UARTx receive buffer register 2 (high-order byte) first before the UARTx receive buffer register 1 (low-order byte).

Reception requires the following setup:

(1) Define a baud rate by setting a value n (n = 0 to 255) into UARTx baud rate generator (addresses 0031_{16}, 0039_{16}).
(2) Set the Receive Initialization Bit (bit 3 of UxCON) to "1".
(3) Configure the data format and clock selection by setting the UARTx mode register.
(4) Set the $\overline{RTS}$ Function Enable Bit (bit 5 of UxCON) if $\overline{RTS}$ function will be used.
(5) Set the Receive Enable Bit (bit 1 of UxCON) to "1".

$\overline{CTS}$ (Clear-to-Send) Function

As a transmitter, the UART can be configured to recognize the Clear-to-Send ($\overline{CTS}$x) input as a handshaking signal. This is enabled by setting the $\overline{CTS}$ Function Enable Bit (bit 5 of UxCON) to "1". If $\overline{CTS}$ function is enabled, even when transmission is enabled and the UARTx transmit buffer register is filled with the data, the transmission never starts; but it will start when inputting "L" to the $\overline{CTS}$x pin.

Figures 27 and 28 show the UARTx transmit timings.

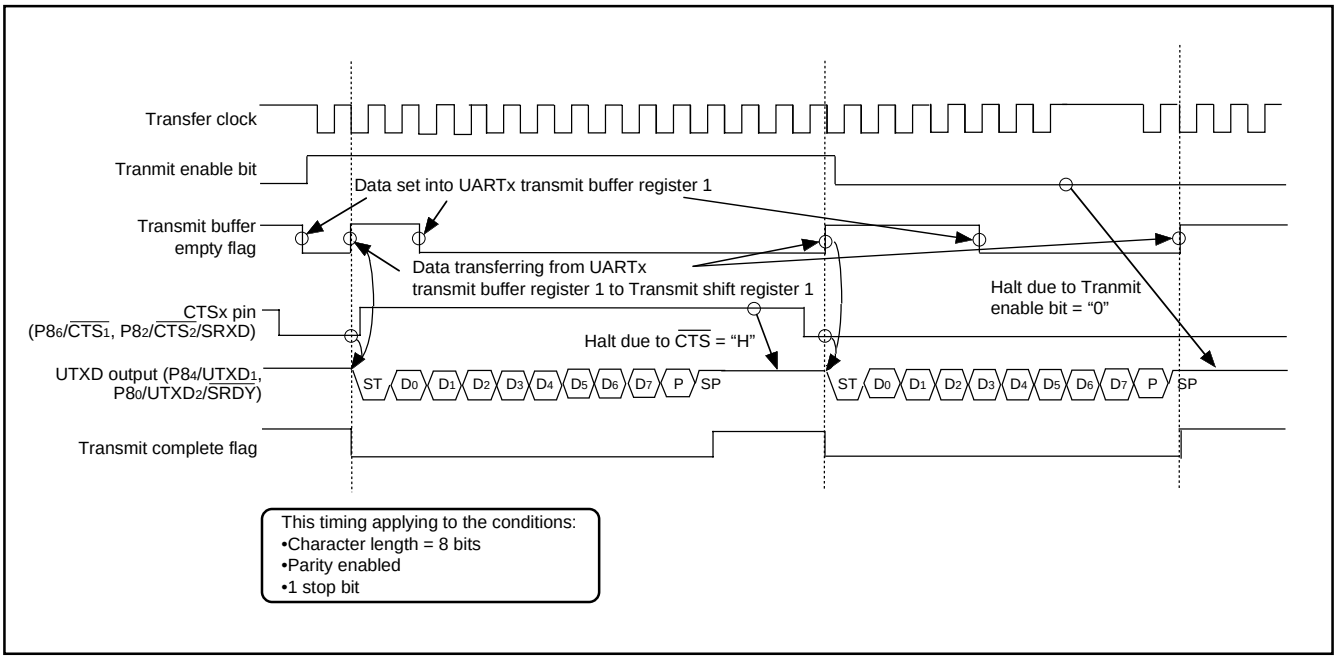


Fig. 27 UARTx transmit timing ($\overline{CTS}$ function enabled)

RTS (Request-to-Send) Function

As a receiver, the UART can be configured to generate the Request-to-Send (RTSx) handshaking signal. This is enabled by setting the RTS Function Enable Bit (bit 6 of UxCON) to "1".

When reception is enabled, that is the Receive Enable Bit is "1", the RTSx pin goes "L" to inform a transmitter that reception is possible. The RTSx pin goes "H" at reception starting and does "L" at receiving of the last bit.

The delay time from the reception of the last stop bit to the assertion of RTSx is selectable using the RTS Assertion Delay Count Select Bits.

When the Receive Enable Bit is set to "0" or the Receive initialization bit is set to "1", the RTSx pin goes "H". Even when the Receive Enable Bit is set to "1", the RTSx pin goes "H" if detecting an invalid start bit.

Figure 29 shows the UARTx receive timing.

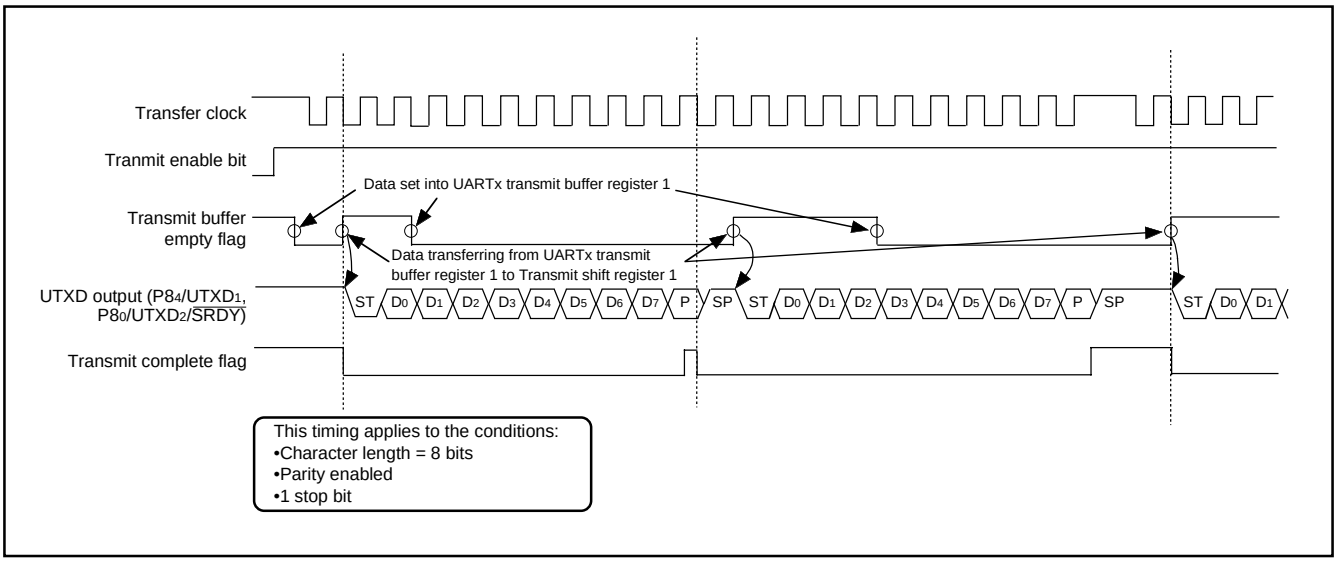


Fig. 28 UARTx transmit timing (CTS function disbled)

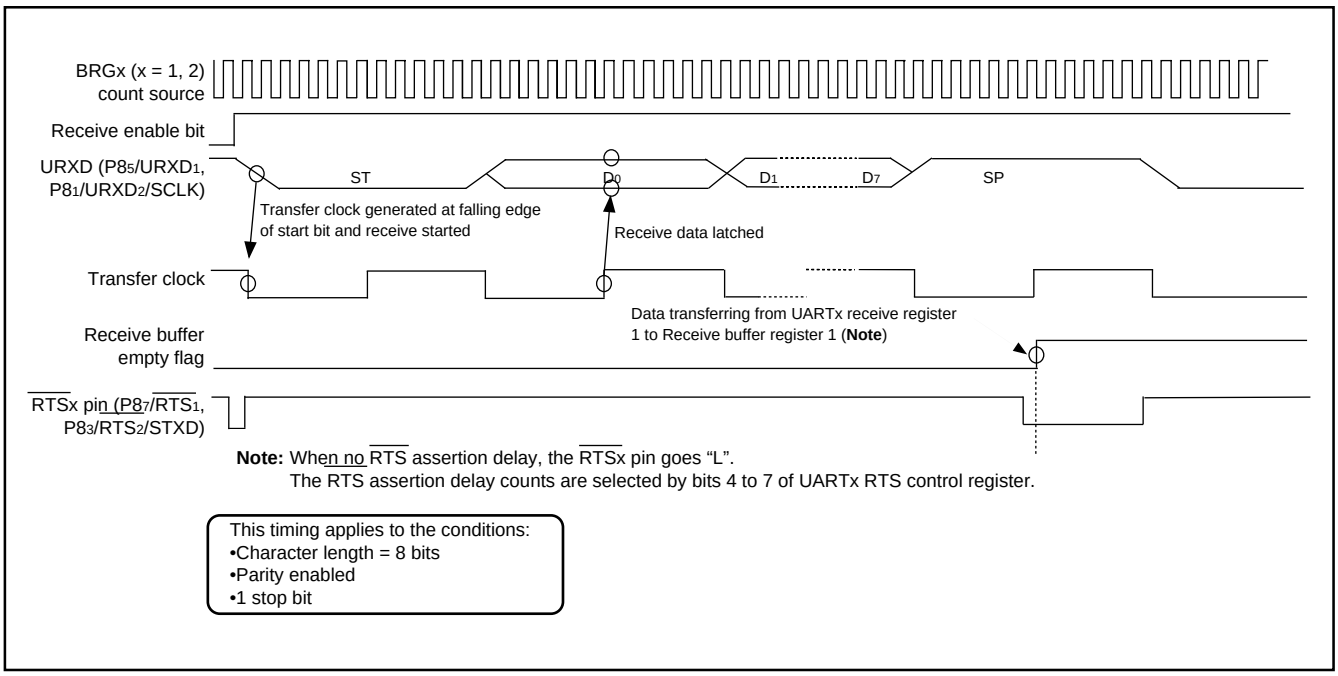


Fig. 29 UARTx transmit timing (RTS function enabled)

UART Address Mode

The UART address mode is intended for use to communicate between the specified MCUs in a multi-MCU environment. The UART address mode can be used in either an 8-bit or 9-bit character length. An address is identified by the MSB of the incoming data being “1”. The bit is “0” for non-address data.

When the MSB of the incoming data is “0” in the UART address mode, the Receive Buffer Full Flag is set to “1”, but the Receive Buffer Full Interrupt Request Bit is not set to “1”. When the MSB of the incoming data is “1”, normal receive operation is performed. In the UART address mode an overrun error is not detected for reception of the 2nd and onward bytes. An occurrence of framing error or parity error sets the Summing Error Interrupt Request Bit to “1” and the data is not received independent of its MSB contents.

Usage of UART address mode is explained as follows:

(1) Set the UART Address Mode Enable Bit to “1”.
(2) Sends the address data of a slave MCU first from a host MCU to all slave MCUs. The MSB of address data must be “1” and the remaining 7 bits specify the address.
(3) The all slave MCUs automatically check for the received data whether its stop bit is valid or not, and whether the parity error occurs or not (when the parity enabled). If these errors occur, the Framing Error Flag or Parity Error Flag and the Summing Error Flag are set to “1”. Then, the Summing Error Interrupt Request Bit is also set to “1”.
(4) When received data has no error, the all slave MCUs must judge whether the address of the received address data matches with their own addresses by a program. After the MSB being “1” is received, the UART Address Mode Enable Bit is automatically set to “0” (disabled).
(5) The UART Address Mode Enable Bit of the slave MCUs which have be judged that the address does not match with them must be set to “1” (enabled) again by a program to disable reception of the following data.
(6) Transmit the data of which MSB is “0” from the host MCU. The slave MCUs disabling the UART address mode receive the data, and their Receive Buffer Full Flags and the Receive Buffer Full Interrupt Request Bits are set to “1”. For the other slave MCUs enabling the UART address mode, their Receive Buffer Full Flag are set to “1”, but their Receive Buffer Full Interrupt Request Bits are not set to “1”.
(7) An overrun error cannot be detected after the first data has been received in UART Address Mode. Accordingly, even if the slave MCUs does not read the received data and the next data has been received, an overrun error does not occur.

Thus, a communication between a host MCU and the specified MCU can be realized.

[UARTx (x = 1, 2) Mode Register (UxMOD)] 0030_{16}, 0038_{16}

The UART x mode register consists of 8 bits which set a transfer data format and an used clock.

[UARTx (x = 1, 2) Baud Rate Generator (UxBRG)] 0031_{16}, 0039_{16}

The UARTx baud rate generator determines the baud rate for transfer.

The baud rate generator divides the frequency of the count source by 1/(n + 1), where n is the value written to the baud rate generator.

The reset cannot affect the contents of baud rate generator.

[UARTx (x = 1, 2) Status Register (UxSTS)] 0032_{16}, $003A_{16}$

The read-only UARTx status register consists of seven flags (bits 0 to 6) which indicate the UART operating status and various errors.

When the UART address mode is enabled , the setting and clearing conditions of each flag differ from the following explanations. These differences are explained in section “**UART Address Mode**”.

•Transmit complete flag (TCM)

In the case where no data is contained in the transmit buffer register, the Transmit Complete Flag (TCM) is set to “1” when the last bit in the transmit shift register is transmitted.

The TCM flag is also set to “1” at reset or initialization by setting the Transmit Initialization Bit (bit 2 of UxCON). It is set to “0” when transmission starts, and it is kept during the transmission.

•Transmit buffer empty flag (TBE)

The Transmit Buffer Empty Flag (TBE) is set to “1” when the contents of the transmit buffer register are loaded into the transmit shift register. The TBE flag is also set “1” at the hardware reset or initialization by setting the Transmit Initialization Bit. It is set to “0” when a write operation is performed to the low-order byte of the transmit buffer register.

•Receive buffer full flag (RBF)

The Receive Buffer Full Flag (RBF) is set to “1” when the last stop bit of the data is received. The RBF flag is set to “0” when the low-order byte of the receive buffer register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit.

●Receive Errors

If there is an error, it is detected at the same time that data is transferred from the receive shift register to the receive buffer register, and the Receive Buffer Full Flag is set to "1". The all error flags PER, FER, OER and SER are cleared to "0" when the UARTx status register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit.

The Summing Error Flag (SER) is set to "1" when any one of the PER, FER and OER is set to "1".

The Parity Error Flag (PER) is set to "1" when the sum total of 1s of received data and the parity does not correspond with the selection with the Parity Select Bit (PMD). It is enabled only if the Parity Enable Bit (bit 5 of UxMOD) is set to "1".

The Framing Error Flag (FER) is set to "1" when the number of stop bit of the received data does not correspond with the selection with the Stop Bit Length Select Bit (STB).

The Overrun Flag Flag (OER) is set to "1" if the previous data in the low-order byte of the receive buffer register 1 (addresses 0034_{16}, $003C_{16}$) is not read before the current receive operation is completed. It is also set "1" if any one of error flags is "1" for the previous data and the current receive operation is completed. Be sure to read UARTx status register to clear the error flags before the next reception has been completed.

[UARTx (x = 1, 2) Control Register (UxCON)] 0033_{16}, $003B_{16}$

The UARTx control register consists of eight control bits for the UARTx function. This register can enable the $\overline{CTS}$, $\overline{RTS}$ and UART address mode.

If the Transmit Enable Bit (TEN) is set to "0" (disabled) while a data is being transmitted, the transmitting operation will stop after the data has been transmitted. If the Receive Enable Bit (REN) is set to "0" (diabled) while a data is being received, the receiving operation will stop after the data has been received.

When setting the Transmit Initialization Bit (TIN) to "1", the TEN bit is set to "0" and the UARTx status register will be set to "03_{16}" after the data has been transmitted. To retransmit, set the TEN to "1" and set a data to the transmit buffer register again. The TIN bit will be cleared to "0" one cycle later after the TIN bit has been set to "1".

Setting the Receive Initialization Bit (RIN) to "1" sets all of the REN, RBF and the receive error flags (PER, FER, OER, SER) to "0". The RIN bit will be cleared to "0" one cycle later after the RIN bit has been set to "1".

When $\overline{CTS}$ or $\overline{RTS}$ function is disabled, pins $\overline{CTS_1}$ and $\overline{CTS_2}$ or $\overline{RTS_1}$ and $\overline{RTS_2}$ can be used as ordinary I/O ports, correspondingly.

[UARTx Transmit/Receive Buffer Registers 1, 2 (UxTRB1/ UxTRB2)] 0034_{16}, 0035_{16}, $003C_{16}$, $003D_{16}$

The transmit buffer register and the receive buffer register are located at the same address. The transmit buffer register is write-only and the receive buffer register is read-only. If a character bit length is 7 bits, the MSB of received data is invalid. If a character bit length is 7 or 8 bits, the received contents of UxTRB2 are also invalid. If a character bit length is 9 bits, the received high-order 7 bits of UxTRB2 are "0".

[UARTx (x = 1, 2) RTS Control Register (UxRTS)] 0036_{16}, $003E_{16}$

The delay time from the reception of the last stop bit to the assertion of $\overline{RTSx}$ is selectable using the $\overline{RTS}$ Assertion Delay Count Select Bits. If the stop bit is detected before $\overline{RTS}$ assertion delay time has expired, the $\overline{RTSx}$ pin is kept "H". The $\overline{RTS}$ assertion delay count starts after the last data reception is completed.

Setting the RIN bit to "1" resets the UxRTS. After setting the RIN bit to "1", set this UxRTS.

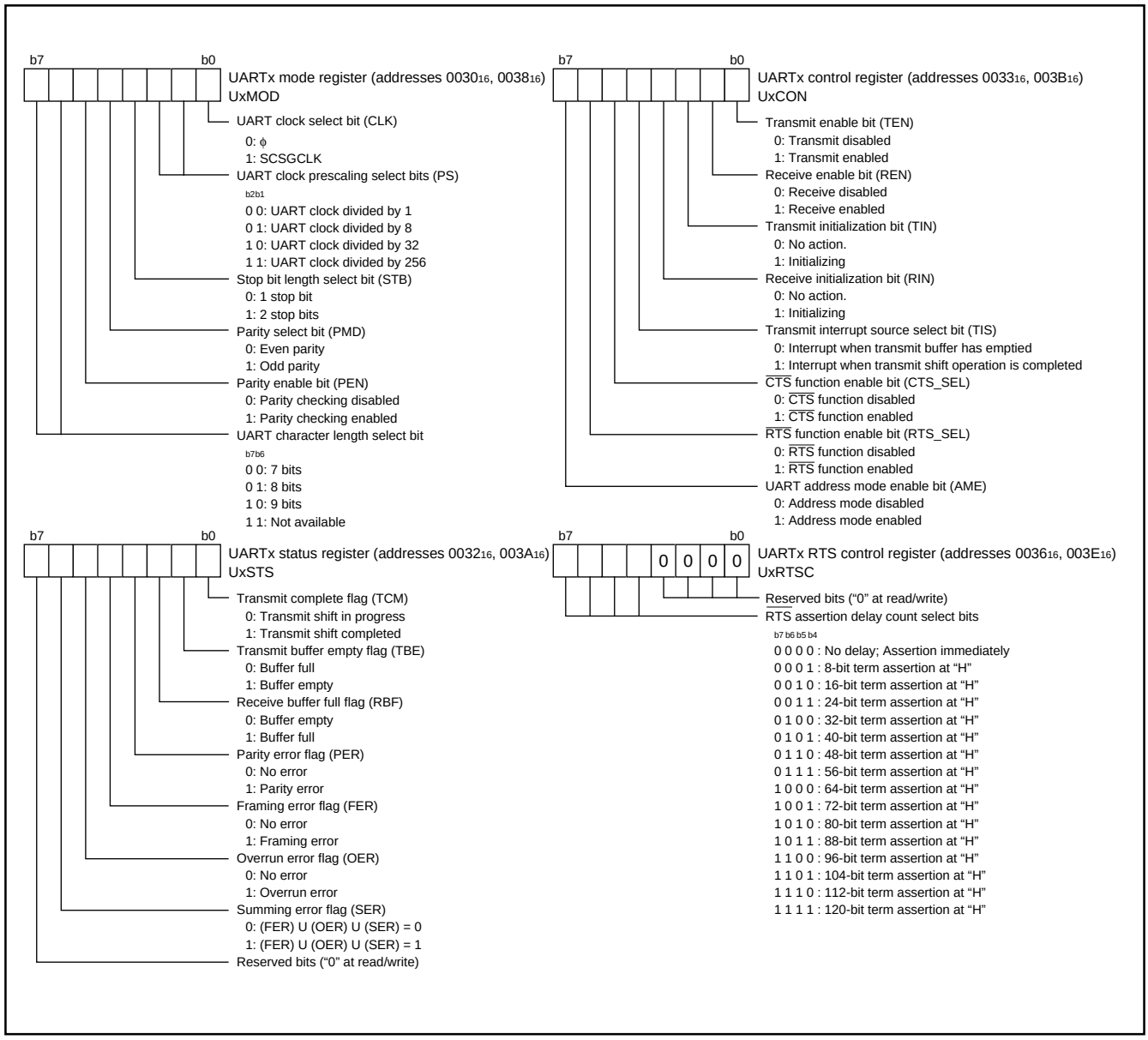


Fig. 30 Structure of UART related registers

DMAC

The 7641 group is equipped with 2 channels of DMAC (direct memory access controller) which enable high speed data transfer from a memory to a memory without use of the CPU.

The DMAC initiates the data transfer with an interrupt factor specified by the DMAC channel x (x = 0, 1) hardware transfer request source bit (DxCEN), or with a software trigger.

The DxTMS [DMA Channel x (x = 0, 1) Transfer Mode Selection Bit] selects one of two transfer modes; cycle steal mode or burst transfer mode. In the cycle steal mode, the DMAC transfers one byte of data for each request. In the burst transfer mode, the DMAC transfers the number of bytes data specified by the transfer count register for each request. The count register is a 16-bit counter; the maximum number of data is 65,536 bytes per one request.

Figure 31 shows the DMA control block diagram and Figure 32 shows the structure of DMAC related registers.

[DMAC Index and Status Register] DMAIS

The DMAC Index and Status Register consists of various control bits for the DMAC and its status flags.

The DMA Channel Index Bit (DCI) selects which channel (0 or 1) will be accessed, since the mode registers, source registers, destination registers and transfer count register of both DMAC channels share the same SFR addresses, respectively.

[DMAC Channel x (x = 0, 1) Mode Registers 1, 2] DMAxM1, DMAxM2

The 16 bits of DMAC Channel x Mode Registers 1 and 2 control each operation of DMAC channels 0 and 1.

When the DMAC Channel x (x = 0, 1) Write Bit (DxDWC) is “0”, data is simultaneously written into each latch and register of the Source Registers, Destination Register, and Transfer Count Registers. When this bit is “1”, data is written only into their latches.

When data is read from each register, it must be read from the higher bytes first, then the lower bytes. When writing data, write to the lower bytes first, then the higher bytes.

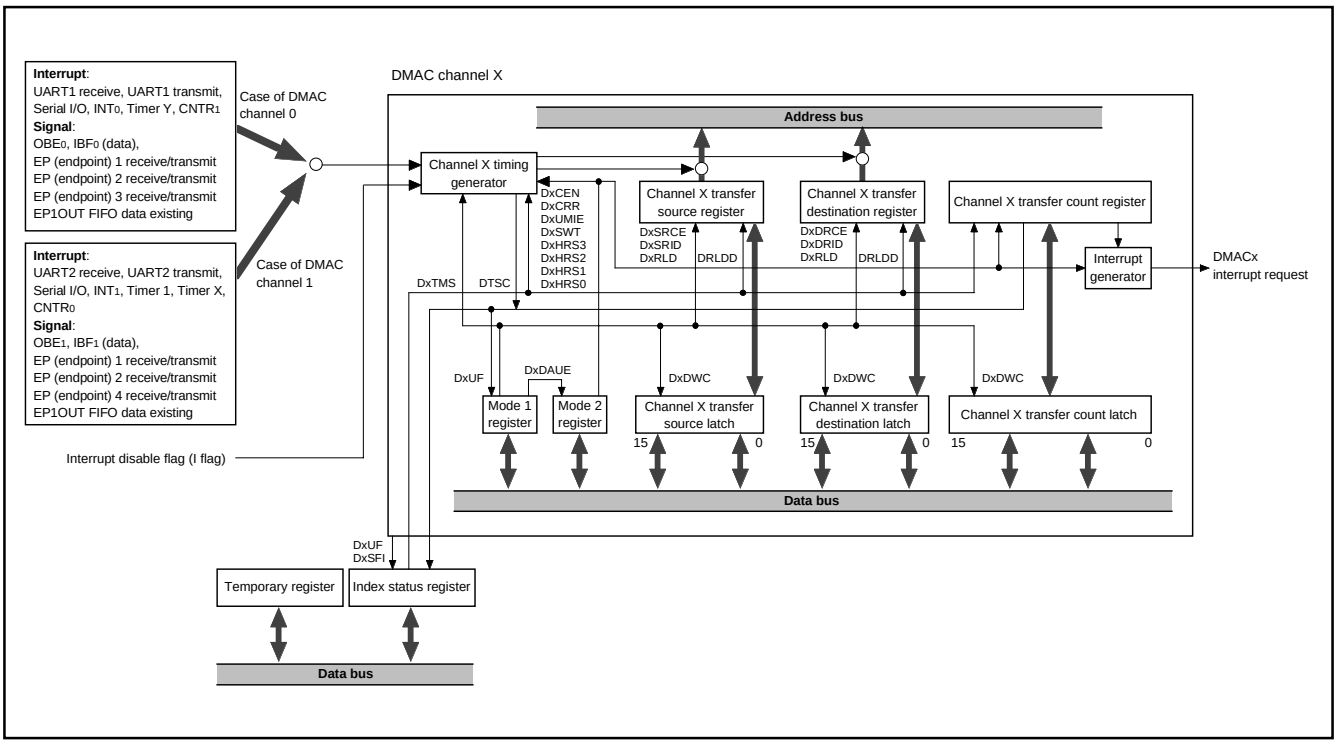
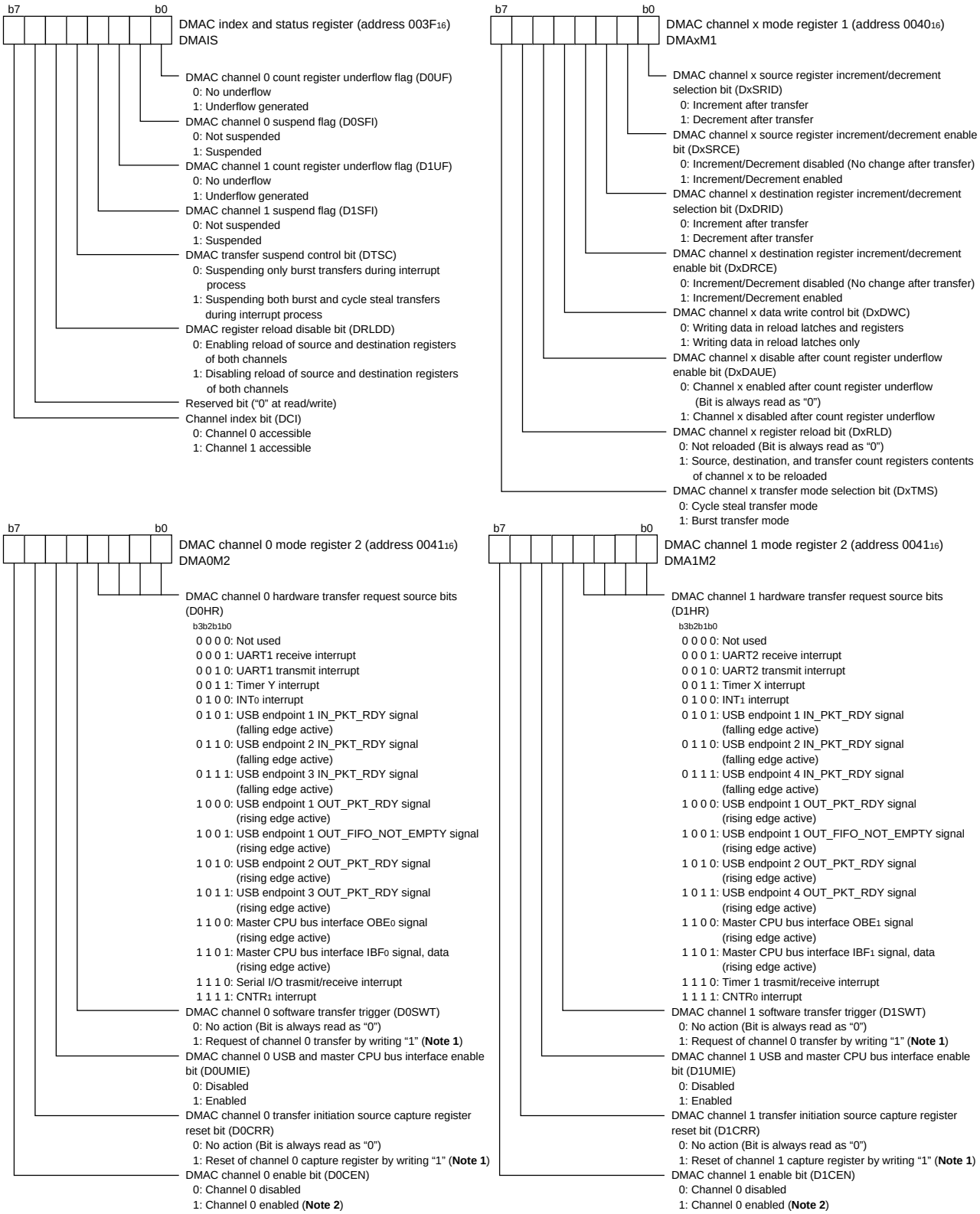


Fig. 31 DMACx (x = 0, 1) block diagram



Notes 1: This bit is automatically cleared to "0" after writing "1".
2: When setting this bit to "1", simultaneously set the DMAC channel x transfer initiation source capture register reset bit (bit 6 of DMAxM2) to "1".

Fig. 32 Structure of DMACx related register

(1) Cycle Steal Transfer Mode

When the DMAC Channel x (x = 0, 1) Transfer Mode Selection Bit (DxTMS) is set to “0”, the respective DMAC Channel x operates in the cycle steal transfer mode.

When a request of the specified transfer factor is generated, the selected channel transfers one byte of data from the address indicated by the Source Register into the address indicated by the Destination Register.

There are two kinds of DMA transfer triggers supported: hardware transfer factor and software trigger. Hardware transfer factors can be selected by the DMACx (x = 0, 1) Hardware Transfer Request Factor Bit (DxHR). To only use the Interrupt Request Bit, the interrupt can be disabled by setting its Interrupt Enable Bit of Interrupt Control Register to “0”.

The DMA transfer request as a software trigger can be generated by setting the DMA Channel x (x = 0, 1) Software Transfer Trigger Bit (DxSWT) to “1”.

The Source Registers and Transfer Destination Registers can be either decreased or increased by 1 after transfer completion by setting bits 0 to 3 in the DMAC Channel x (x = 0, 1) Mode Register. When the Transfer Count Register underflows, the Source Registers and Destination Registers are reloaded from their latches if the DMAC Register Reload Disable Bit (DRLDD) is “0”. The Transfer Count Register value is reloaded after an underflow regardless of DRLDD setting. At the same time, the DMAC Interrupt Request Bit and the DMA Channel x (x = 0, 1) Count Register Underflow Flag are set to “1”.

The DMAC Channel x Disable After Count Register Underflow Enable Bit (DxDAUE) is “1”, the DMAC Channel x Enable Bit (DxCEN) goes to “0” at an under flows of Transfer Count Register. By setting the DMAC Channel x (x = 0, 1) Register Reload Bit (DxRLD) to “1”, the Source Registers, Destination Registers, and Transfer Count Registers can be updated to the values in their respective latches.

When one signal among USB endpoint signals is selected as the hardware transfer request factor, and DMAC Channel x (x = 0, 1) USB and Master CPU Bus Interface Enable Bit (DxUMIE) is “1”; transfer between the USB FIFO and the master CPU bus interface input/output buffer can be performed effectively. This transfer function is only valid in the cycle steal mode. To validate this function, the DMAC Channel x (x = 0, 1) USB and the Master CPU Bus Interface Enable Bit (bit 5 of DxTR) must be set to “1”. The following shows an example of a transfer using this function.

Packet Transfer from USB FIFO to Master CPU Bus Interface Buffer

When the USB OUT_PKT_RDY is selected as the hardware transfer request factor; if the USB OUT_PKT_RDY is “1” and the master CPU bus interface output buffer is empty, the transfer request is generated and the transfer is initiated. The OUT_PKT_RDY retains “1” and a transfer request is generated each time the output buffer empties until all the data in the corresponding endpoint FIFO has been transferred.

The transfer ends when the last byte in the USB receive packet is transferred and the OUT_PKT_RDY flag goes to “0” (in the case of AUTO_CLR bit = “1”).

Byte Transfer from USB FIFO to Master CPU Bus Interface Buffer

When the USB Endpoint 1 OUT_FIFO_NOT_EMPTY is selected as a hardware transfer request factor, if there is data in the USB Endpoint 1 FIFO and the master CPU bus interface output buffer is empty; a transfer request is generated and the transfer is initiated. The transfer is performed by unit of one byte.

Transfer from Master CPU Bus Interface Buffer to USB FIFO

When the USB Endpoint X (X = 1 to 4) IN_PKT_RDY (IN_PKT_RDY = “0”) is selected as a hardware transfer request factor, if there is data in the master CPU bus interface output buffer and the data in the USB FIFO is within the specified packet size, a transfer request is generated.

The DMA transfer is terminated when a command (A0 = “1”) is input to the master CPU bus interface input buffer.

The timing chart for a cycle steal transfer caused by a hardware-related transfer request and a software trigger are shown in Figure 33 and 34, respectively.

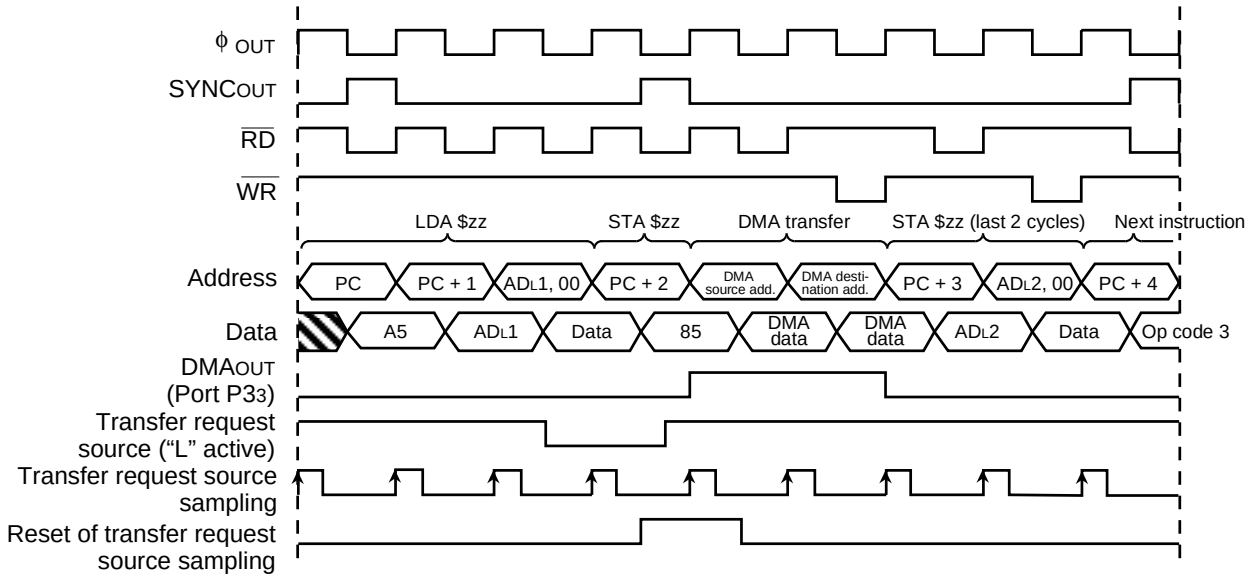


Fig. 33 Timing chart for cycle steal transfer caused by hardware-related transfer request

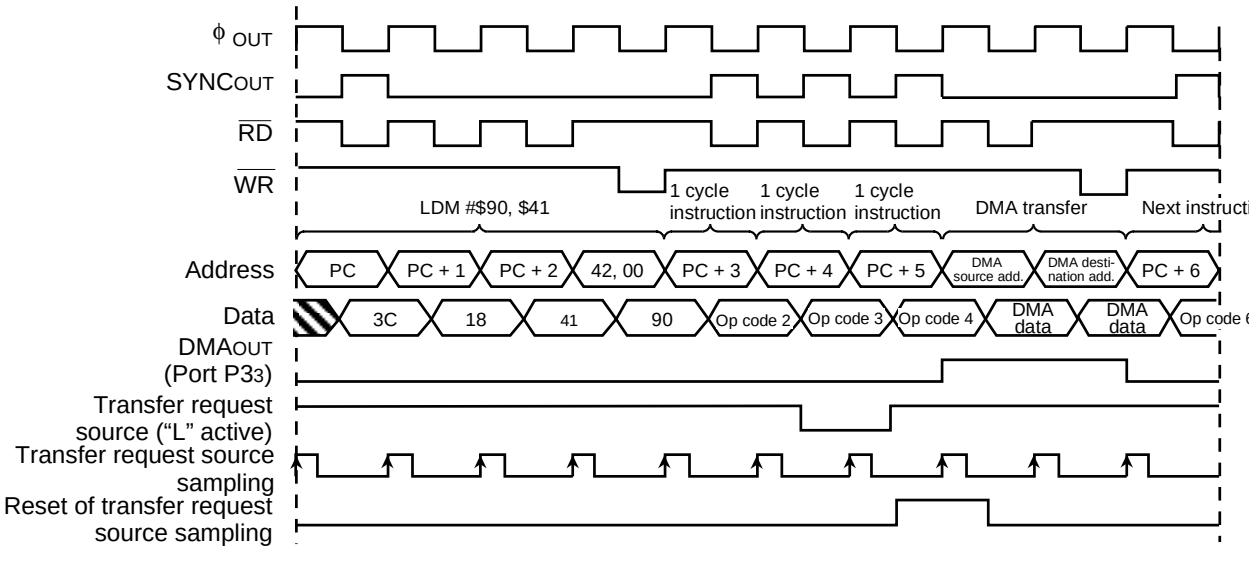


Fig. 34 Timing chart for cycle steal transfer caused by software trigger transfer request

(2) Burst Transfer Mode

When the DMAC Channel x Transfer Mode Selection Bit (DxTMS) is set to “1”, the respective DMAC channel operates in the burst transfer mode.

In the burst transfer mode, the DMAC continually transfers the number of bytes of data specified by the Transfer Count Register for one transfer request. Other than this, the burst transfer mode operation is the same as the cycle steal mode operation.

Priority

The DMAC places a higher priority on Channel-0 transfer requests than on Channel-1 transfer requests.

If a Channel-0 transfer request occurs during a Channel-1 burst transfer operation, the DMAC completes the next transfer source and destination read/write operation first, and then starts the Channel-0 transfer operation. As soon as the Channel-0 transfer is completed, the DMAC resumes the Channel-1 transfer operation.

When an interrupt request occurs during any DMA operation, the transfer operation is suspended and the interrupt process routine is initiated. During the interrupt operation, the DMAC automatically sets the corresponding DMAC Channel x (x = 0, 1) Flag to “1”. As soon as the CPU completes the interrupt operation, the DMAC clears the flag to “0” and resumes the original operation from the point where it was suspended.

The suspended transfer due to the interrupt can also be resumed during its interrupt process routine by writing “1” to the DMAC Channel x (x = 0,1) Enable Bit (DxCEN).

The timing charts for a burst transfer caused by a hardware-related transfer request are shown in Figure 35.

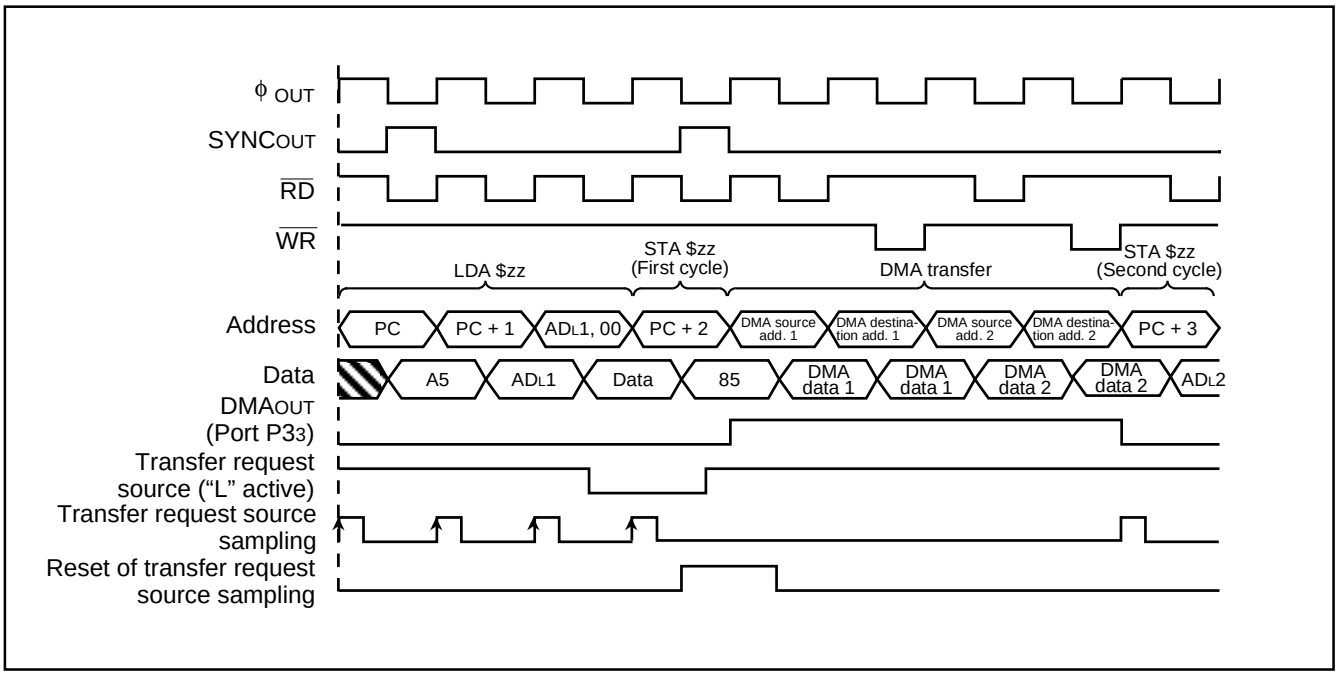


Fig. 35 Timing chart for burst transfer caused by hardware-related transfer request

USB FUNCTION

The 7641 Group MCU is equipped with a USB Function Control Unit (USB FCU). This USB FCU allows the MCU to communicate with a host PC using a minimum amount of the MCU power. This built-in USB FCU complies with USB Standard Specification Version 1.1 that supports four transfer types: Control Transfer, Isochronous Transfer, Interrupt Transfer, and Bulk Transfer. This built-in USB FCU performs the data transfer error detection and transfer retry operation by hardware. The default transfer mode of the USB FCU is bulk transfer mode at reset. The user must set the USB FCU for the required transfer mode by software.

The USB FCU has five endpoints (Endpoint 0 to Endpoint 4). The EPINDEX bit selects one of these five endpoints for the USB FCU to use. Each endpoint has IN (transmit) FIFO and OUT (receive) FIFO. To use the USB FCU, the USB enable bit (USBC7) must be set to “1”. There are two USB related interrupts supported for this MCU: USB Function Interrupt and USB SOF Interrupt.

Figure 36 shows the USB FCU (USB Function Control Unit) block diagram. The USB FCU consists of the SIE (Serial Interface Engine) performing the USB data transfer, GFI (Generic Function Interface) performing USB protocol handing, SIU (Serial Engine Interface Unit) performing a received address and endpoint decoding, MCI (Microcontroller Interface) handling the MCU interface or performing address decoding and synchronization of control signals, and the USB transceiver.

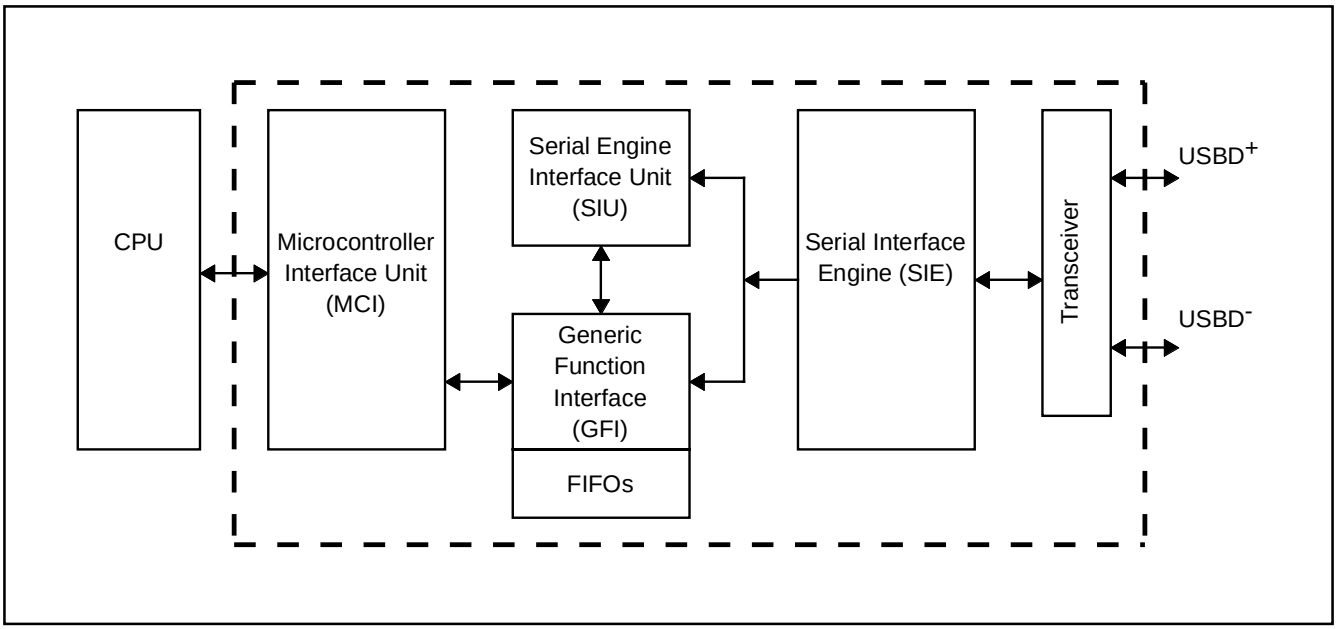


Fig. 36 USB FCU (USB Function Control Unit) block

USB Transmission

Endpoint 0 to Endpoint 4 have IN (transmit) FIFOs individually. Each endpoint's FIFO is configured in following way:

Endpoint 0: 16-byte
Endpoint 1: Mode 0: 512-byte
Mode 1: 1024-byte
Mode 2: 0-byte
Mode 3: 2048-byte
Mode 4: 768-byte
Mode 5: 880-byte
Endpoint 2: Mode 0: 32-byte
Mode 1: 128-byte
Endpoint 3: 16-byte
Endpoint 4: 16-byte

When Endpoint 1 or Endpoint 2 is used for data transmit, the IN FIFO size can be selected. Endpoint 1 and Endpoint 2 have programmable IN-FIFOs size; 6 modes for Endpoint 1, and 2 modes for Endpoint 2. Each mode can be selected by the USB endpoint FIFO mode selection register (address $005F_{16}$).

When writing data to the USB Endpoint-x FIFO (addresses 0060_{16} to 0064_{16}) in the SFR area, the internal write pointer for the IN FIFO is automatically increased by 1. When the AUTO_SET bit is "1" and if the stored data reaches to the max. packet value set in USB Endpoint x IN max. packet size register (address $005B_{16}$), the USB FCU sets the IN_PKT_RDY bit to "1". When the AUTO_SET bit is "0", the IN_PKT_RDY bit will not be automatically set to "1"; it must be set to "1" by software. (The AUTO_SET bit function is not applicable to Endpoint 0.)

The USB FCU transmits the data when it receives the next IN token. The IN_PKT_RDY bit automatically goes to "0" when the data transfer is complete.

●Isochronous transfer

Endpoints 1 to 4 can be used in isochronous transfer mode. When using isochronous transfer mode, the ISO/TOGGLE_INIT bit must be set to "1". When ISO_UPDATE = "1" and the corresponding endpoint's ISO/TOGGLE_INIT bit = "1", the USB FCU delays the rise of the IN_PKT_RDY bit until the next SOF signal transmission. In this way, the USB FCU can synchronize a transmit data to the SOF signal.

●Interrupt transfer mode

Endpoints 1 to 4 can be used in interrupt transfer mode. During a regular interrupt transfer, an interrupt transaction is similar to the bulk transfer. Therefore, there is no special setting required. When IN-endpoint is used for a rate feedback interrupt transfer, INTPT bit of the IN_CSR register must be set to "1". Once the INTPT bit is set, data toggling occurs every time a data packet is transmitted to the host, regardless of the handshake packet availability or type. The following steps show how to configure the IN-endpoint for the rate feedback interrupt transfer.

1. Set a value which is larger than 1/2 of the USB Endpoint-x FIFO size to the USB Endpoint x IN max. package size register.
2. Set INTPT bit to "1".
3. Flush the old data in the FIFO.
4. Load interrupt status information and set IN_PKT_RDY bit to "1".
5. Repeat steps 3 and 4.

In a real application, the function-side always has transfer data when the function sends an endpoint in a rate feedback interrupt. Accordingly, the USB FCU never returns a NAK against the host IN token for the rate feedback interrupt. The USB FCU always transmits data in the FIFO in response to an IN token, regardless of IN_PKT_RDY.

When MAXP size ≤ (a half of IN FIFO size), the IN FIFO can store two packets (called double buffer). At this time, the IN FIFO status can be checked by monitoring the IN_PKT_RDY bit and the TX_NOT_EPT flag. The TX_NOT_EPT flag is a read-only flag which shows the FIFO state. When IN_PKY_RDY = 0 and TX_NOT_EPT = 0, IN FIFO is empty. When IN_PKY_RDY = 0 and TX_NOT_EPT = 1, IN FIFO has one packet.

In double buffer mode, as long as the IN FIFO is not filled with double packets, IN_PKT_RDY will not be set to "1", even if it is set to "1" by software, but TX_NOT_EPT flag will be set to "1". In single buffer mode, if MAXP > (a half of IN FIFO), this condition never occurs.

When IN_PKT_RDY = "1" and TX_NOT_EPT = "1", IN FIFO holds two packets in double buffer mode and one packet in single packet mode. In single packet mode, when the IN_PKT_RDY bit is set to "1" by software, the TX_NOT_EPT flag is set to "1" as well. During double buffer mode, if you want to load two packets sequentially, you must set the IN_PKT_RDY bit to "1" each time a packet is loaded.

USB Reception

Endpoint 0 to Endpoint 4 have OUT (receive) FIFOs individually. Each endpoint's FIFO is configured in following way:

Endpoint 0: 16-byte
Endpoint 1: Mode 0: 800-byte
Mode 1: 1024-byte
Mode 2: 2048-byte
Mode 3: 0-byte
Mode 4: 1280-byte
Mode 5: 1168-byte
Endpoint 2: Mode 0: 32-byte
Mode 1: 128-byte
Endpoint 3: 16-byte
Endpoint 4: 16-byte

When Endpoint 1 or Endpoint 2 is used for data receive, the OUT FIFO size can be selected. Endpoint 1 and Endpoint 2 have programmable IN-FIFOs size; 6 modes for Endpoint 1, and 2 modes for Endpoint 2. Each mode can be selected by the USB endpoint FIFO mode selection register (address $005F_{16}$).
Data transmitted from the host-PC is stored in Endpoint x FIFO (0060_{16} to 0064_{16}). Every time the data is stored in the FIFO, the internal OUT FIFO write pointer is increased by 1. When one complete data packet is stored, the OUT_PKT_RDY flag is set to "1" and the number of received data packets is stored in USB Endpoint x OUT write count registers (Low and High). When the AUTO_CLR bit is "1" and the received data is read out from the OUT FIFO, the OUT_PKT_RDY flag is cleared to "0". When the AUTO_CLR bit is "1", the OUT_PKT_RDY flag will not be cleared automatically by the FIFO read; it must be cleared by software. (The AUTO-CLR bit function is not applicable in Endpoint 0.)

When MAXP size ≤ (a half of OUT FIFO size), the OUT_FIFO can receive 2 packets (double buffer). At this time, the OUT_ FIFO status can be checked by the OUT_PKT_RDY flag. When the FIFO holds two packets and one packet is read from the FIFO, the OUT_PKT_RDY flag is not cleared even if it is set to "0". (The flag returns from "0" to "1" in one ϕ cycle after the read-out). During double buffer mode, the USB Endpoint x OUT write count registers (Low and High) holds the number of previously received packets. This count register is updated after reading out one of packets in the OUT FIFO and clearing the OUT_PKT_RDY flag to "0".

When Endpoint 1 OUT is used and its MAXP size ≤ (one third of OUT FIFO size), this can be used as triple buffer. However, before the third EOF packet has been received, be sure to read the first packet data and clear the OUT_PKT_RDY flag.
If the third packet data has been received before the first packet data is read, an overrun error occurs and the third packet data is destroyed.

TOGGLE Initialization

In order to initialize the data toggle sequence bit of the endpoint, in other words, resetting the next data packet to DATA0; set the ISO/TOGGLE_INT bit to "1" and then clear back to "0".

USB Interrupts

The USB FCU has two interrupts, USB Function Interrupt and USB SOF (Start Of Frame) Interrupt.

●USB Function Interrupt (USBF-INT)

The USBF-INT is usable for the USB data flow control and power management. The USBF-INT request occurs at data transmit/receive completion, overrun/underrun, reset, or receiving suspend/resume signal. To enable this interrupt, the USB function interrupt enable bit in the interrupt control register A (address 0005_{16}) and the respective bit in the USB interrupt enable registers 1 and 2 (addresses 0005_{16} and 0005_{16}) must be set to "1". When setting bit 7 in USB interrupt enable register 2 to "1", the suspend interrupt and the resume interrupt are enabled.

Endpoint x (x = 0 to 4) IN interrupt request occurs when the USB Endpoint x IN interrupt status flag (INTST 0, 2, 4, 6, 8) of USB interrupt status registers 1 and 2 (addresses 0052_{16} and 0053_{16}) is "1". The USB Endpoint x IN interrupt status flag is set to "1" when the respective endpoint IN_PKT_RDY bit is "1".
Endpoint x (x = 0 to 4) OUT interrupt request occurs when the USB endpoint x OUT interrupt status flag (INTST3, 5, 7, 9) in USB interrupt status registers 1 and 2 is set to "1". The USB Endpoint x OUT interrupt status flag is set to "1" when the respective endpoint OUT_PKT_RDY flag is "1".

The overrun/underrun interrupt request occurs when the USB overrun/underrun interrupt status flag (INTST12) in USB interrupt status register 2 is set to "1". This flag is set to "1" when the FIFO data overruns or underruns in isochronous transfer mode.
The USB reset interrupt request occurs when the USB reset interrupt status flag (INTST13) in USB interrupt status register 2 is set to "1". This flag is set when the SE0 is detected on the D+/D- line for at least 2.5 μs. When this situation happens, all USB internal registers (addresses 0050_{16} to $005E_{16}$), except this flag, are initialized to the default state at reset. The USB reset interrupt is always enabled.

The suspend/resume interrupt request occurs when either the USB resume signal interrupt status flag (INTST14) or the USB suspend signal interrupt status flag (INTST15) in USB interrupt status register 2 is set to "1".
The bits in both interrupt status registers 1 and 2 can be cleared by writing "1" to each bit.

●USB SOF interrupt

The USB SOF interrupt is usable in isochronous transfers. This interrupt request occurs when an SOF packet is received. To enable a USB SOF interrupt, set the USB SOF interrupt enable bit of interrupt control register A to "1".

Suspend/Resume Functions

If no bus activity is detected on the D+/D- line for at least 3 ms, the USB suspend signal detect flag (SUSPEND) of the USB power control register (address 0051_{16}) and the USB suspend signal interrupt status flag of USB interrupt status register 2 are set to "1" and the suspend interrupt request occurs. The following procedure must be executed after pushing the internal registers (A, X, Y) to memories during the suspend interrupt process routine.

(1) Clear all bits of USB interrupt status register 1 (address 0052_{16}) and USB interrupt status register 2 (address 0053_{16}) to "0".
(2) Set the USB clock enable bit to "0". (After disabling the USB clock, do not write to any of the USB internal registers (addresses 0050_{16} to 0064_{16}), except for the USB control register (address 0013_{16}), clock control register (address $001F_{16}$), and frequency synthesizer control register (address $006C_{16}$).
(3) Set the frequency synthesizer enable bit to "0".
(4) Set the USB line driver current control bit to "1". (Always keep the USB line driver current control bit set to "0" during USB function operations. When operating at Vcc = 3.3 V, this bit does not need to be set.)
(5) Keep total drive current at 500 μA or less.
(6) Disable the timer 1 interrupt.
(7) Disable the timer 2 interrupt. (Disable all the other external interrupts.)
(8) Set the timer 1 interrupt request bit to "0".
(9) Set the timer 2 interrupt request bit to "0".
(10) Set the interrupt disable flag (I) to "0".
(11) Execute the STP instruction.

At this point, the MCU will be in stop mode (suspend mode). Before executing the STP instruction, make sure to set the USB function interrupt request bit (bit 0 at address 0002_{16}) to "0" and the USB function interrupt enable bit (bit 0 at address 0005_{16}) to "1".

The USB suspend detect signal flag goes to “0” when the USB resume signal detect flag (RESUME) is set to “1”. During suspend mode, if the clock operation is started up with a process (remote wake-up) other than the resume interrupt process (for example; reset or timer), make sure to clear the USB suspend detect signal flag to “0” when you set the USB remote wake-up bit to “1”. When the USB FCU is in suspend mode and detects a non-idle signal on the D+/D- line, the USB resume detect flag and the USB resume signal interrupt status flag both go to “1” and a resume interrupt request occurs. At this point, pull the internal registers (A, X, Y) in this interrupt process routine. Take the following procedure in the USB resume interrupt process.

(1) Set the USB line driver current control bit to “0”. (When operating at Vcc = 3.3 V, this bit does not need to be set.)
(2) Set the frequency synthesizer enable bit to “1” and set a 2 ms to 5 ms wait.
(3) Check the frequency synthesizer lock status bit. If “0”, it must be checked again after a 0.1 ms wait.
(4) Enable the USB clock.

Set the USB resume signal interrupt status flag to “0” after the wake-up sequence process. The USB resume detect flag goes to “0” at the same time. When the clock operation is started up with a remote wake-up, set the USB remote wake-up bit to “1” after the wake-up sequence process. (keep it set to “1” for a minimum of 10 ms and maximum of 15 ms). By doing this, the MCU will send a resume signal to the host CPU and let it know that the suspend state has been released.
After that, set the USB remote wake-up bit and the USB suspend detection flag to “0”, because the USB suspend detection flag is not automatically cleared to “0” with a remote wake-up.

[USB Control Register] USBC
When using the USB function, the USB enable bit must be set to “1”. The USB line driver supply bit must be set to “0” (DC-DC converter is disabled) when operating at Vcc = 3.3V. In this condition, the setting of the USB line driver current control bit has no effect on USB operations.
When the USB artificial SOF enable bit is set to “1”, the MCU judges that a SOF packet is received within 250 ns from a frame starting if an SOF packet is destroyed owing to some cause.

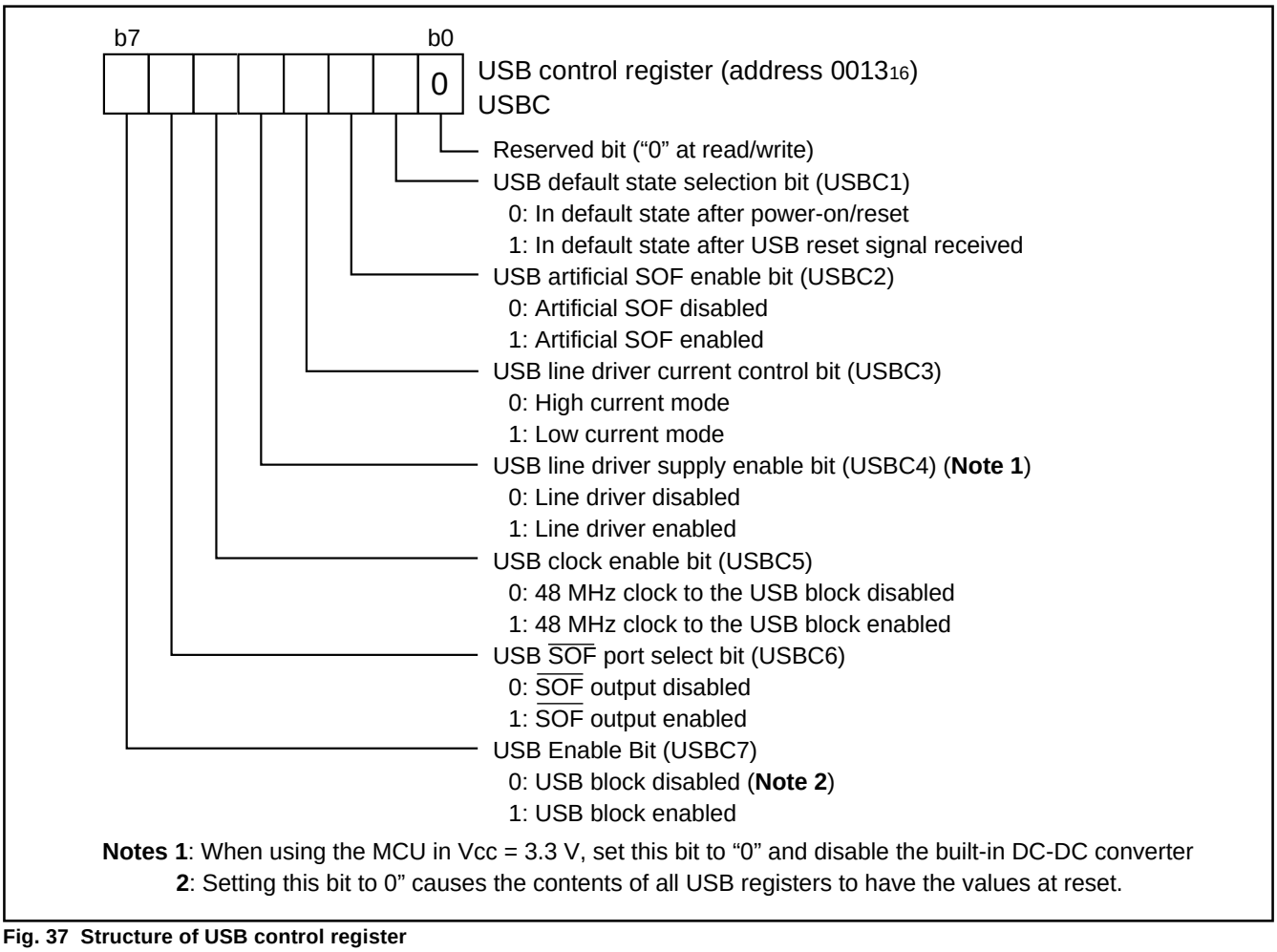


Fig. 37 Structure of USB control register

[USB Address Register] USBA

The USB address register maintains the USB function control unit address assigned by the host computer. When receiving the SET_ADDRESS, keep it in this register. The values of this register are “0” when the device is not yet configured. The values of this register are also set to “0” when the USB block is disabled (bit 7 of USB control register is set to “0”). In addition, no matter what value is written to this register, it will have no effect on the set value.

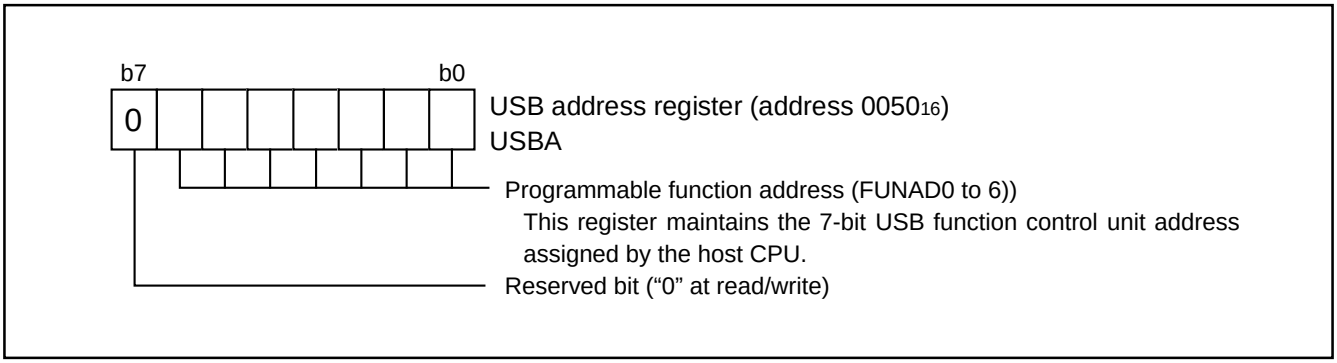


Fig. 38 Structure of USB address register

[USB Power Management Register] USBPM

The USB power management register is used for power management in the USB FCU. This register needs to be set only when using the remote wake-up to resume the MCU from suspend mode.

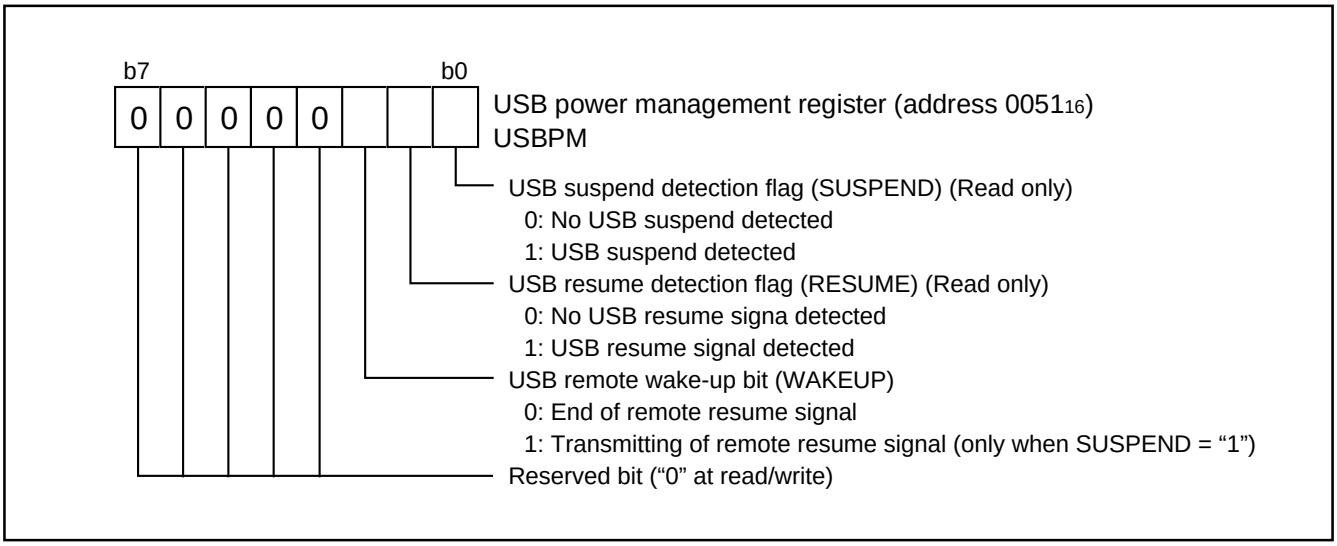


Fig. 39 Structure of USB power management register

[USB Interrupt Status Registers 1 and 2] USBIS1, USBIS2

The USB interrupt status registers are used to indicate the condition that caused a USB function interrupt to be generated. Each status flag and bit can be cleared to “0” by writing “1” to the corresponding bit. Make sure to write to/read from the USB interrupt status register 1 first and then USB interrupt status register 2.

When an IN token is received during an isochronous transfer, and the IN FIFO is empty, an underrun error occurs and INTST12 and IN_CSR2 are set to “1”. When an OUT token is received and the OUT FIFO is full, an overrun error occurs and INTST12 and OUT_CSR2 are set to “1”. Underruns and overruns are not detected by the CPU in bulk transfers and normal interrupt transfers, however in this case, the MCU will send a NAK signal to the host CPU.

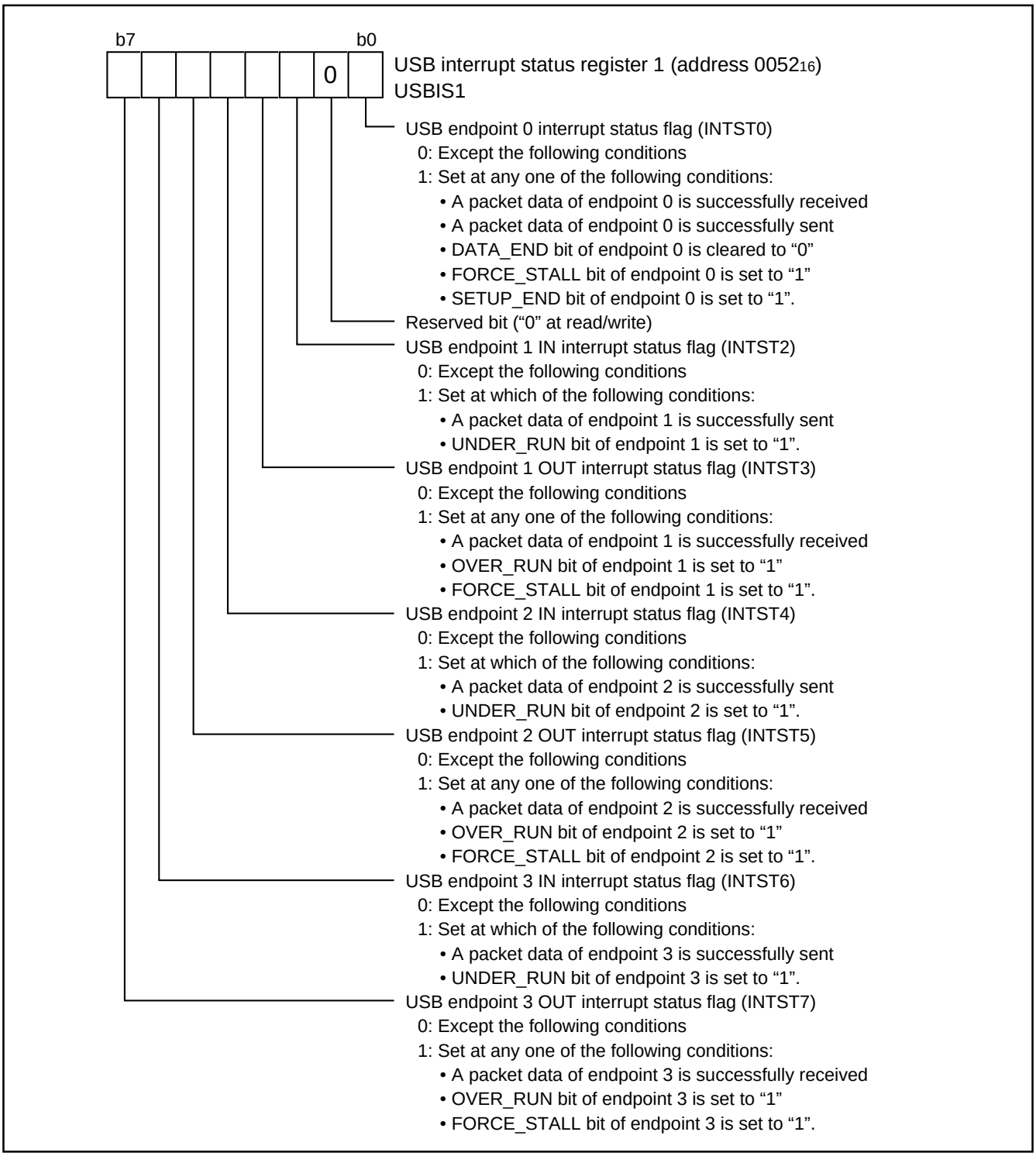


Fig. 40 Structure of USB interrupt status register 1

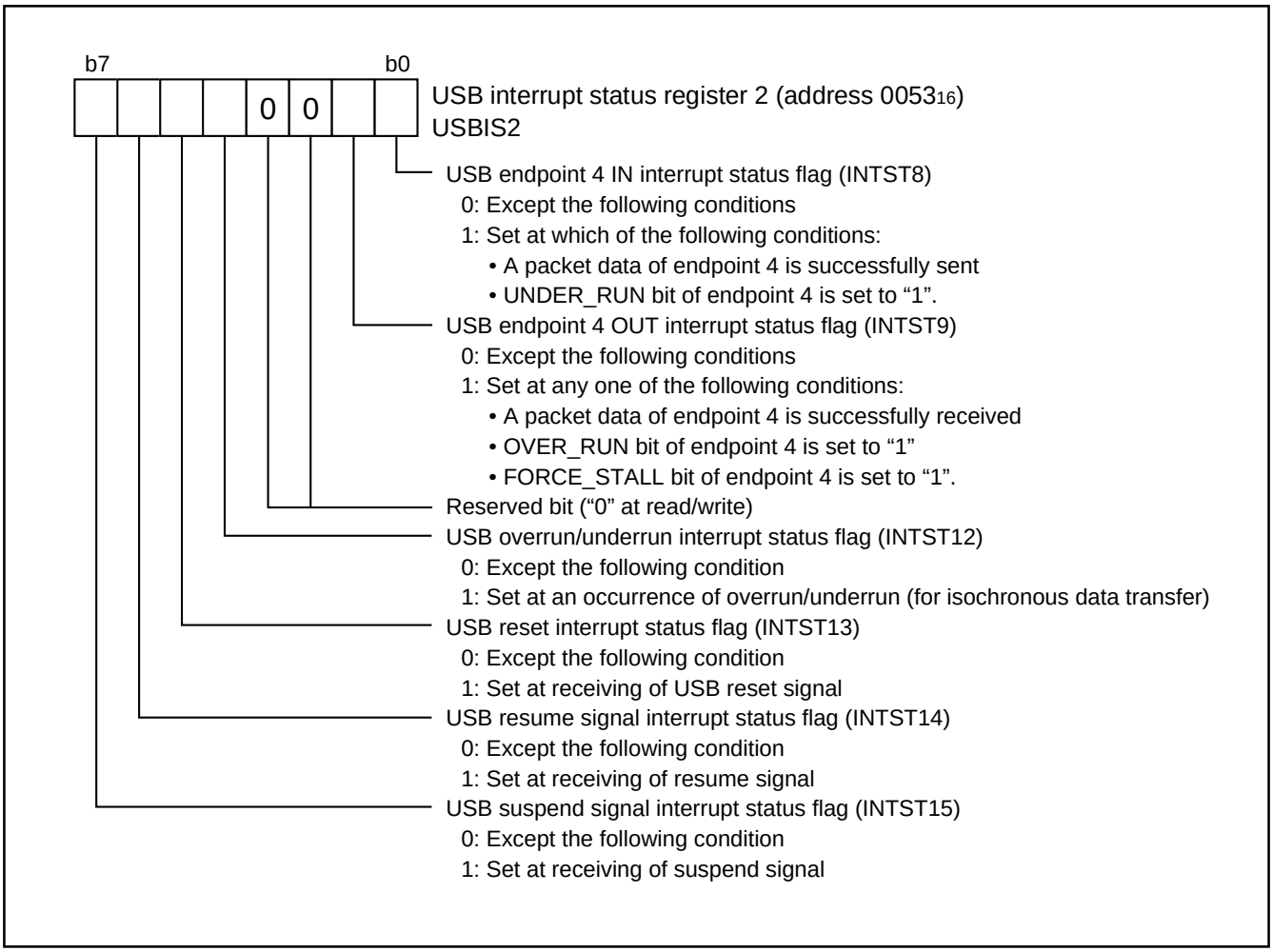


Fig. 41 Structure of USB interrupt status register 2

[USB Interrupt Enable Registers 1 and 2] USBIE1, USBIE2
The USB interrupt enable registers are used to enable the USB function interrupt. Upon reset, all USB interrupts except the USB suspend and USB resume interrupts are enabled.

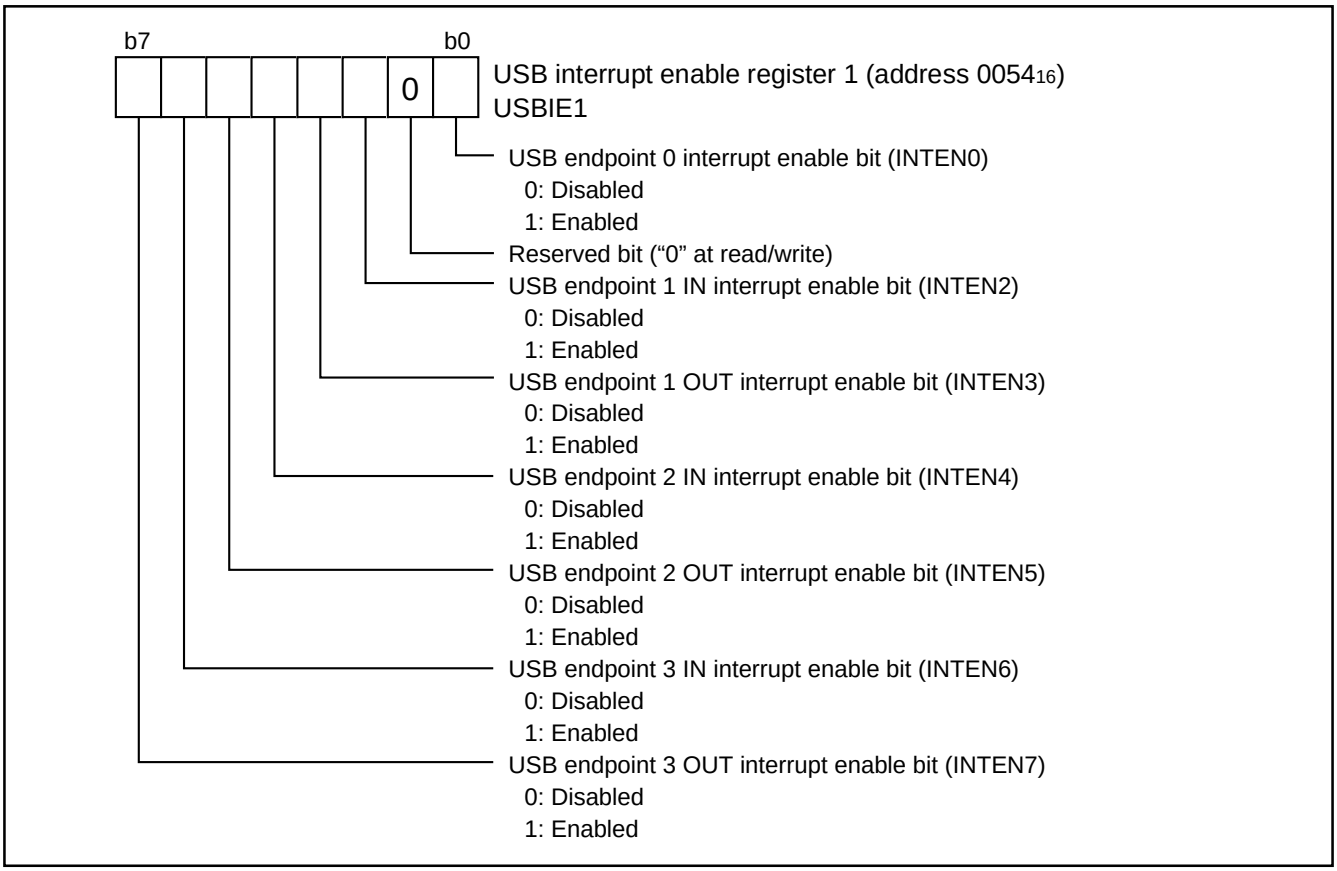


Fig. 42 Structure of USB interrupt enable register 1

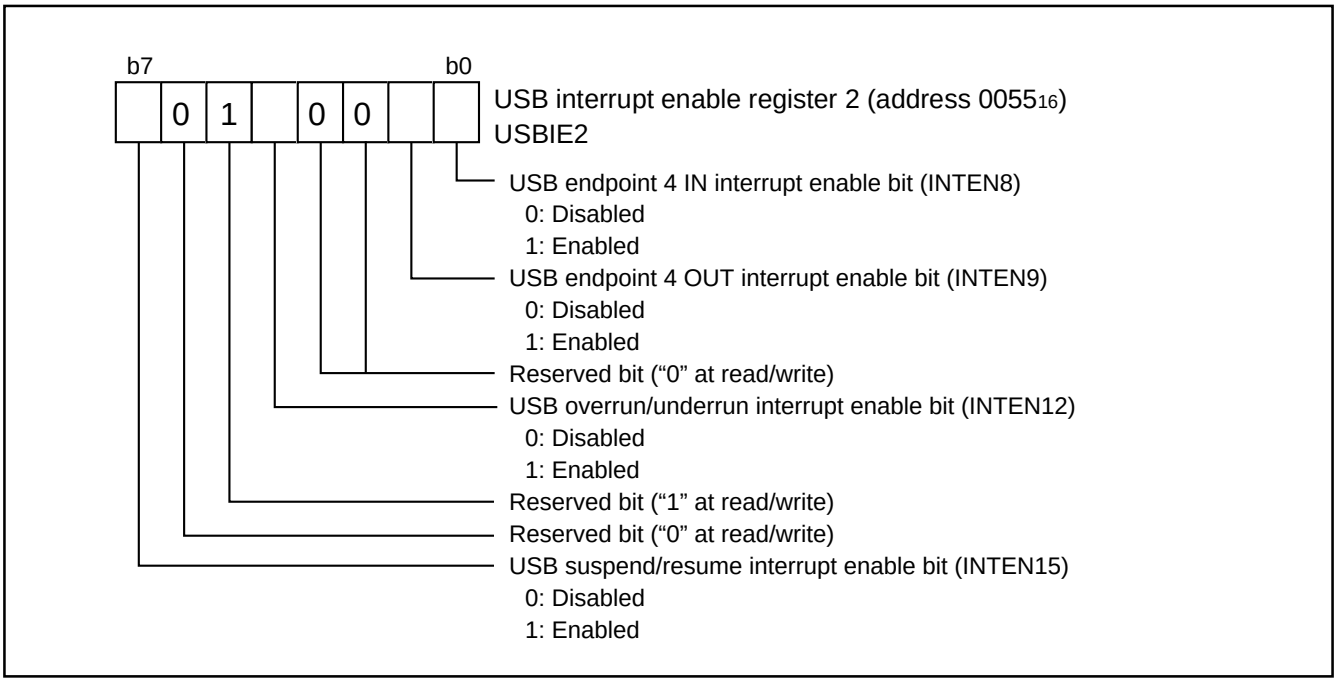


Fig. 43 Structure of USB interrupt enable register 2

[USB Frame Number Registers Low and High]
USBSOFL, USBFOFH

These 11-bit registers contain the frame number of the SOF token received from the host computer. These are read-only registers.

[USB Endpoint Index Register] USBINDEX

This register specifies the accessible endpoint. It serves as an index to endpoint-specific USB Endpoint x IN Control Register, USB Endpoint x OUT Control Register, USB Endpoint x IN Max. Packet Size Register, USB Endpoint x OUT Max. Packet Size Register, USB Endpoint x OUT Write Count Register, and USB FIFO Mode Selection Register (x = 0 to 4).

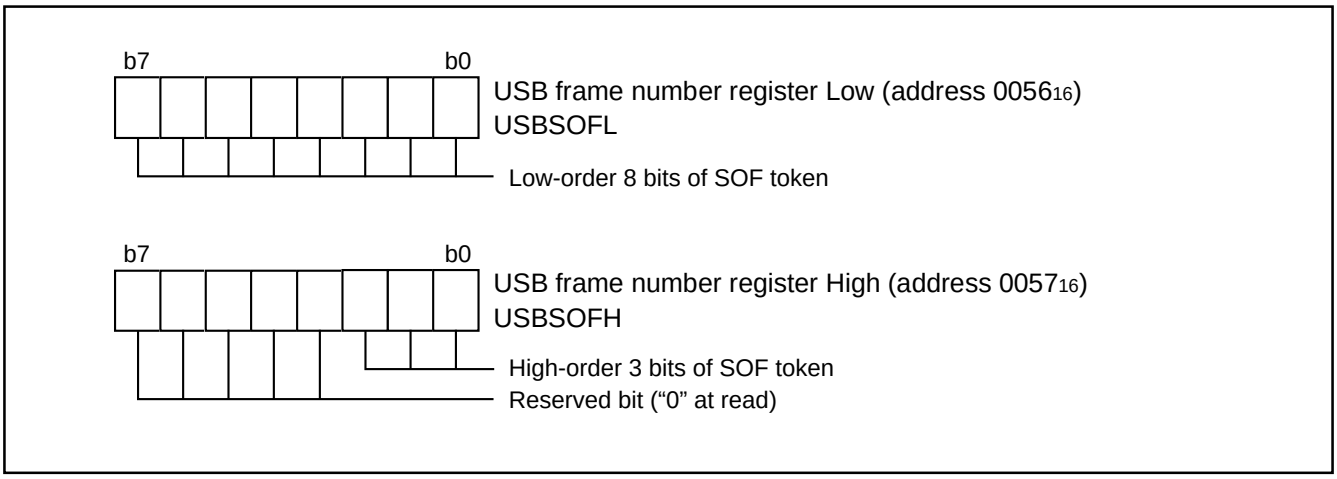


Fig. 44 Structure of USB frame number registers

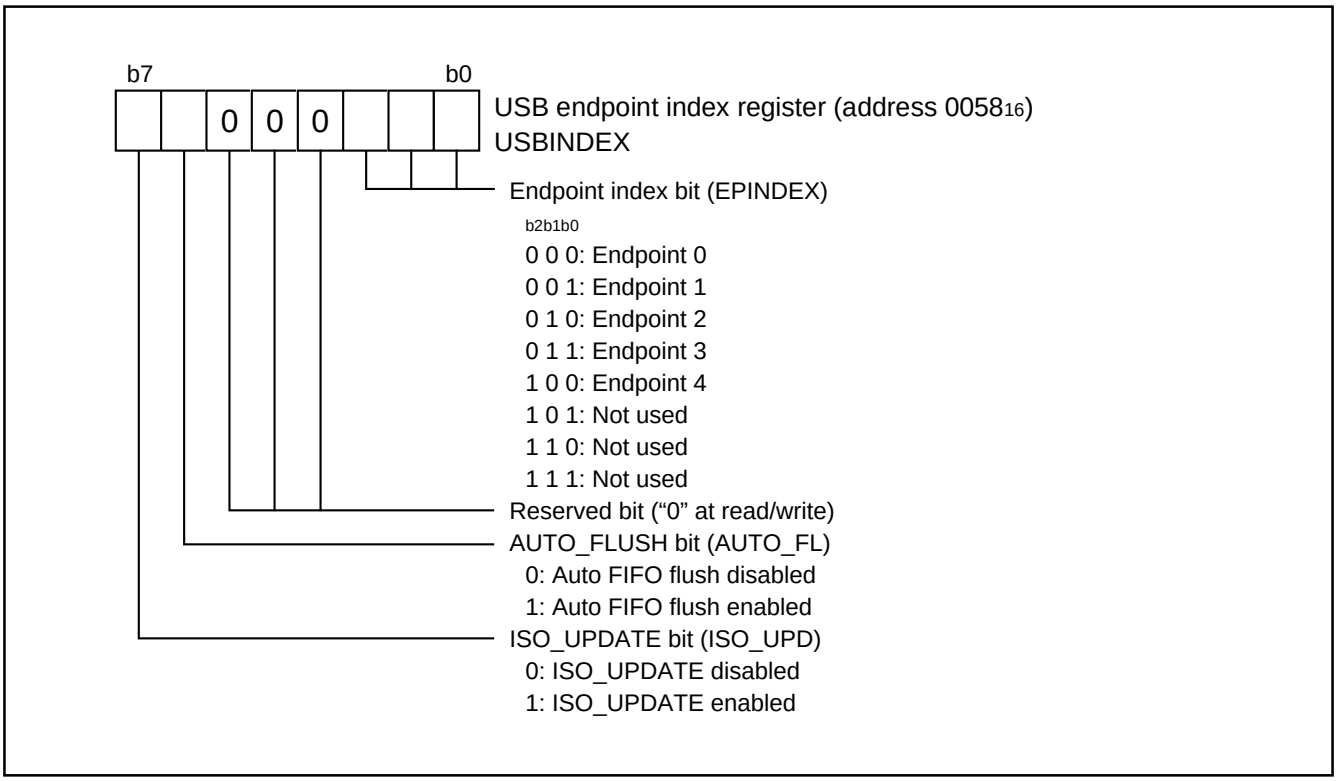


Fig. 45 Structure of USB frame number registers

[USB Endpoint 0 IN Control Register] IN_CSR

This register contains the control and status information of the endpoint 0. This USB FCU sets the OUT_PKT_RDY flag to "1" upon having received a data packet in the OUT FIFO. When reading its one data packet from the OUT FIFO, be sure to set this flag to "0".

After a SETUP token is received, the MCU is in the "decode wait state" until the OUT_PKT_RDY flag is cleared. If the OUT_PKT_RDY flag is not cleared (indicating that the host request has not been successfully decoded), the USB FCU keep returning a NAK to the host for all IN/OUT tokens.

Set the IN_PKT_RDY bit to "1" after the data packet has been written to the IN FIFO. If this bit is set to "1" even though nothing has been written to the IN FIFO, a "0" length data (NULL packet) is sent to the host. The SEND_STALL bit is for sending a STALL to the host if an unsupported request is received by the USB FCU. This bit must be set to "1" (which instructs the FCU USB to send a STALL signal) whenever an unsupported device request is received from the host. The setup process is as follows:

(1) Set SEND_STALL bit to "1"
(2) Set DATA_END bit to "1"
(3) Set OUT_PKT_RDY flag to "0".

Note that if "0" is written to the SEND_STALL bit before the CLEAR_FEATURE (endpoint STALL) request has been received, the next STALL will not be generated.

The DATA_END bit informs the USB FCU of the completion of the process indicated in the SETUP packet. Set this bit to "1" when the process requested in the SETUP packet is completed. (Control Read Transfer: set this bit after writing all of the requested data to the FIFO; Control Write Transfer: set this bit to "1" after reading all of the requested data from the FIFO.) When this bit is "1", the host request is ignored and a STALL is returned. After the status phase process is completed, the USB FCU automatically clears it to "0".

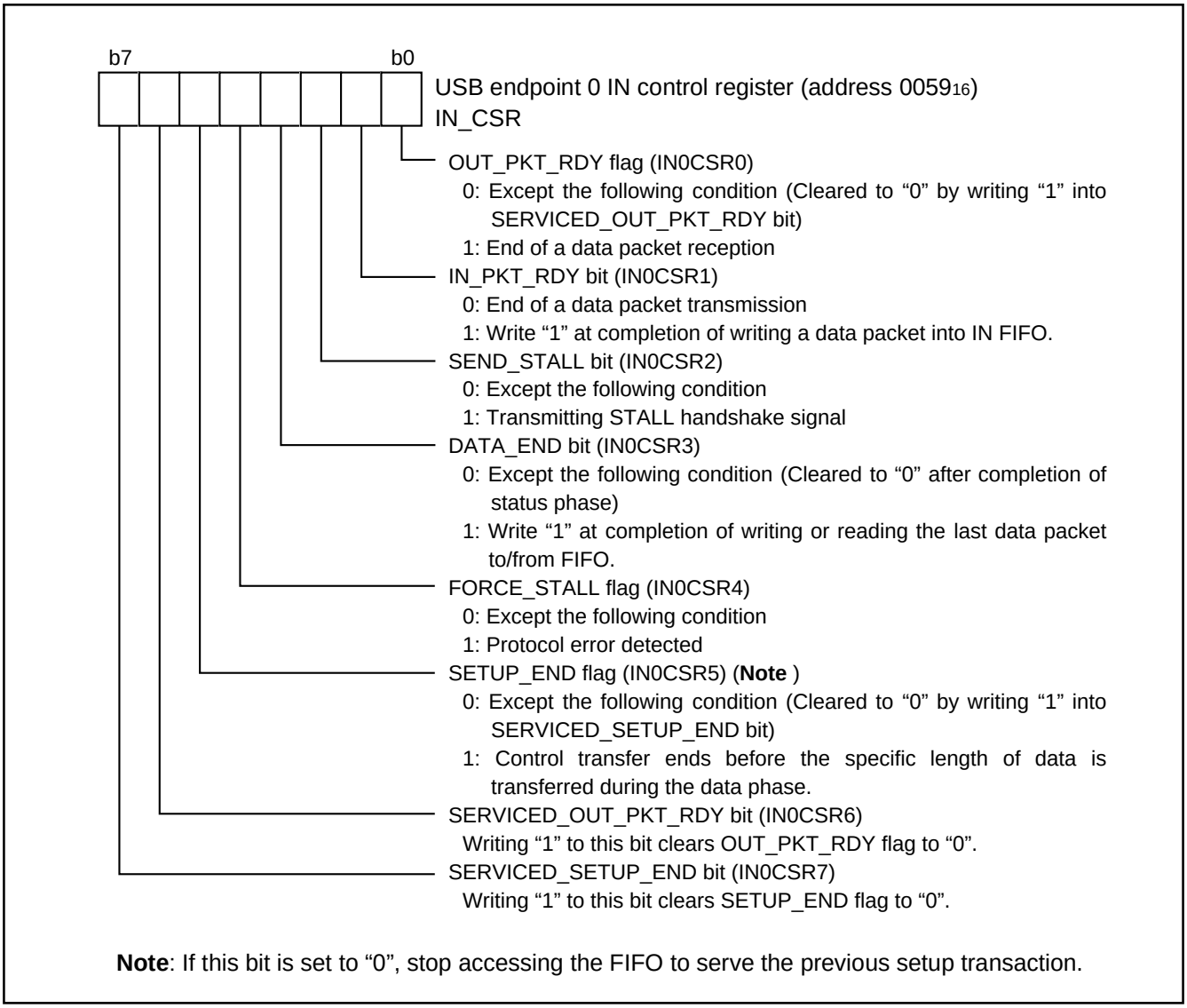
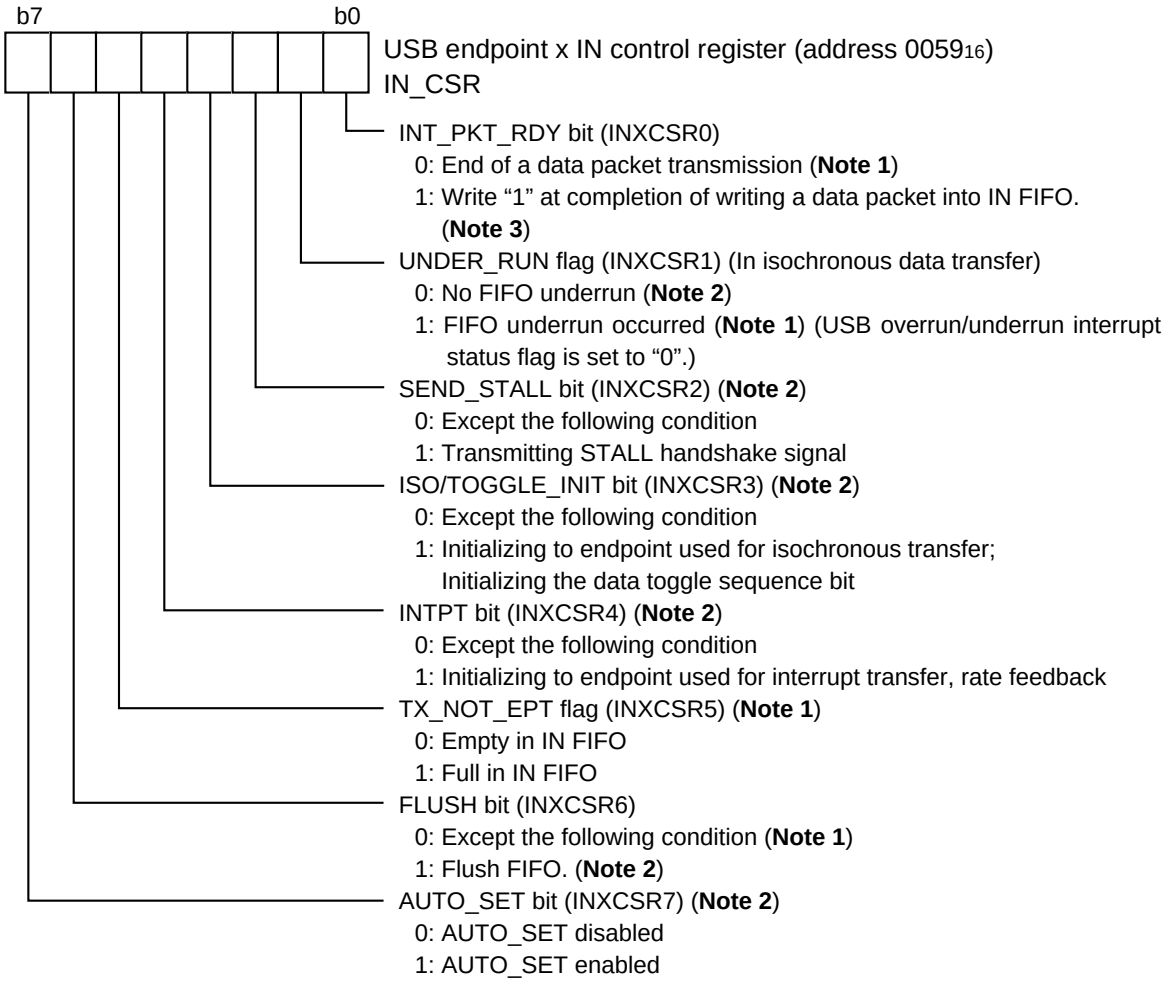


Fig. 46 Structure of USB endpoint 0 IN control register

[USB Endpoint x (x = 1 to 4) IN Control Register] IN_CSR

This register contains the control and status information of the respective IN Endpoints 1 to 4.

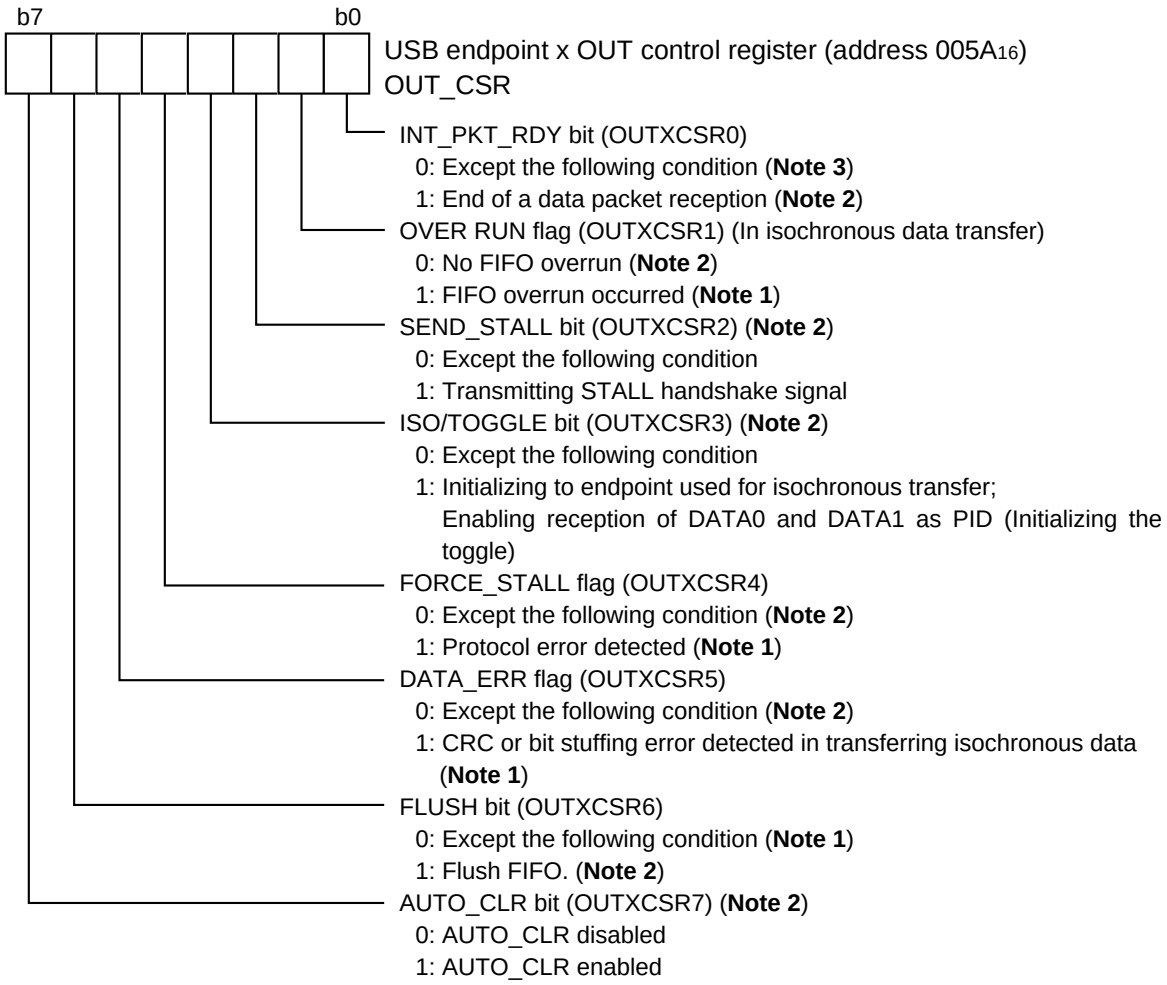
Set the IN_PKT_RDY bit to "1" after the data packet has been written to the IN FIFO. This bit is cleared to "0" when the data transfer is completed. In a bulk IN transfer, this bit is cleared when an ACK signal is received from the host. If an ACK signal is not received, this bit (and the TX_NOT_EMPTY bit) remains as "1". This same data packet is sent after the next IN token is received. The FLUSH bit is for flushing the data in the IN FIFO.



Notes 1: This bit is automatically set to "1" or cleared to "0".
2: The user must program to "1" or "0".
3: When AUTO_SET bit is "0", the user must set to "1". When AUTO_SET bit is "1", this bit is automatically set to "1".

Fig. 47 Structure of USB endpoint x (x = 1 to 4) IN control register

[USB Endpoint x (x = 1 to 4) OUT Control Register] OUT_CSR
This register contains the information and status of the respective OUT endpoints 1 to 4. In the endpoint 0, all bits are reserved and cannot be used (they will all be read out as “0”). The USB FCU sets the OUT_PKT_RDY flag to “1” after a data packet has been received into the OUT FIFO. After reading the data packet in the OUT FIFO, clear this flag to “0”. However, if there is still data in the OUT FIFO, the flag cannot be cleared even by writing “0” by software.



Notes 1: This bit is automatically set to “1” or cleared to “0”.
2: The user must program to “1” or “0”.
3: When AUTO_CLR bit is “0”, the user must clear to “0”. When AUTO_CLR bit is “1”, this bit is automatically cleared to “0”.

Fig. 48 Structure of USB endpoint x (x = 1 to 4) OUT control register

[USB Endpoint x (x = 0 to 4) IN Max. Packet Size Register] IN_MAXP

This register specifies the maximum packet size (MAXP) of an endpoint x IN packet. The value set for endpoint 1 is the number of transmitted bytes divided by 8, and the value set for endpoints 0, 2, 3, and 4 is the actual number of transmitted bytes. The CPU can change these values using the SET_DESCRIPTOR command.

The initial value for endpoints 0, 2, 3 and 4 is 8, and the initial value for endpoint 1 is 1.

[USB Endpoint x (x = 0 to 4) OUT Max. Packet Size Register] OUT_MAXP

This register specifies the maximum packet size (MAXP) of an Endpoint x OUT packet. The value set for endpoint 1 is the number of received bytes divided by 8, and the value set for endpoints 0, 2, 3, and 4 is the actual number of received bytes. The CPU can change these values using the SET_DESCRIPTOR command.

The initial value for endpoints 0, 2, 3, and 4 is 8, and the initial value for endpoint 1 is 1. When using the endpoint 0, both USB endpoint x IN max. packet size register (IN _MAXP) and USB endpoint x OUT max. packet size register (OUT_MAXP) are set to the same value. Changing one register's value effectively changes the value of the other register as well.

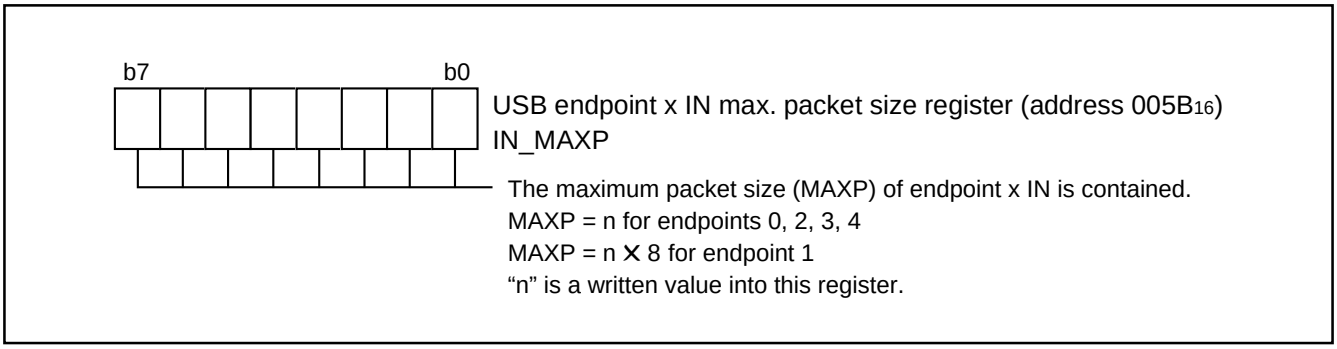


Fig. 49 Structure of USB endpoint x IN max. packet size register

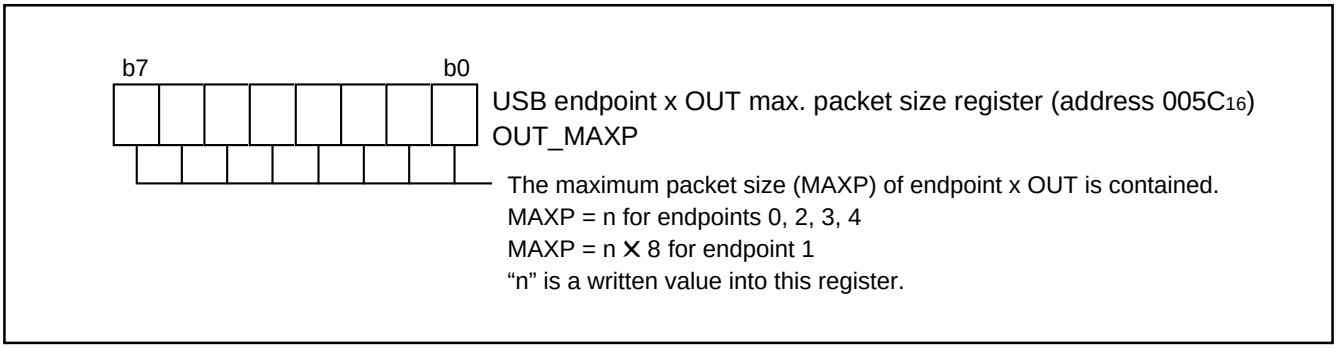


Fig. 50 Structure of USB endpoint x OUT max. packet size register

[USB endpoint x (x = 0 to 4) OUT Write Count Registers (Low and High)] WRT_CNTRL, WRT_CNTH

These registers contain the number of bytes in the endpoint x OUT FIFO. These are read-only registers. These two registers must be read after the USB FCU has received a packet of data from the host. When reading these registers, the lower byte must be read first, then the higher byte.

When the OUT FIF0 is in double buffer mode, the CPU first reads the received number of bytes of the former data packet. The next CPU read can obtain that of the new data packet.

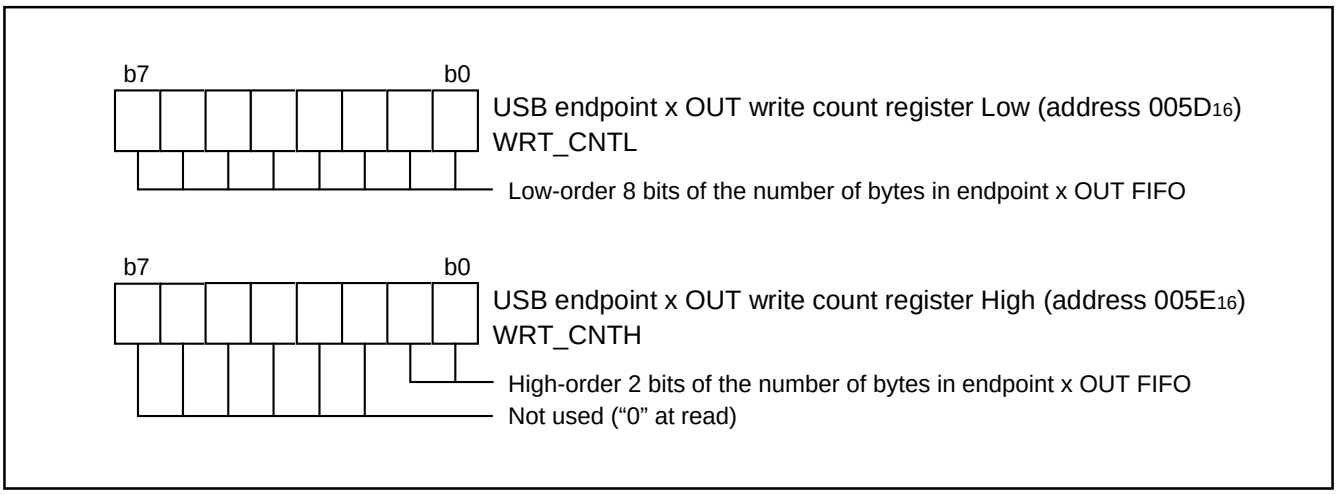


Fig. 51 Structure of USB endpoint x (x = 0 to 4) OUT write count registers

[USB Endpoint x (x = 0 to 4) FIFO Register] USBFIFOx

These registers are the USB IN (transmit) and OUT (receive) FIFO data registers. Write data to the corresponding register, and read data from the corresponding register.

When the maximum packet size is equal to or less than half the FIFO size, these registers function in double buffer mode and can hold two packets of data. When the IN_PKT_RDY bit is “0” and the TX_NOT_EMPTY bit is “1”, these bits indicate that one packet of data is stored in the IN FIFO. When the OUT FIFO is in double buffer mode, the OUT_PKT_RDY flag remains as “1” after the first packet of data is read out (it actually goes to “0” and returns to “1” after one ϕ cycle).

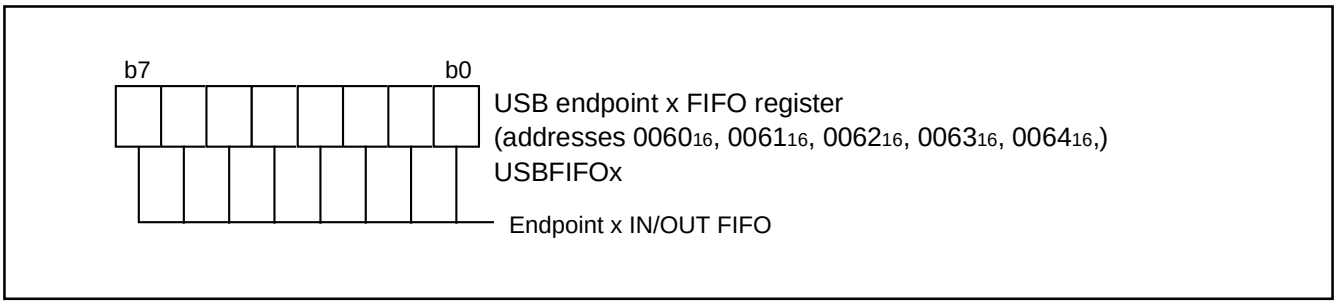
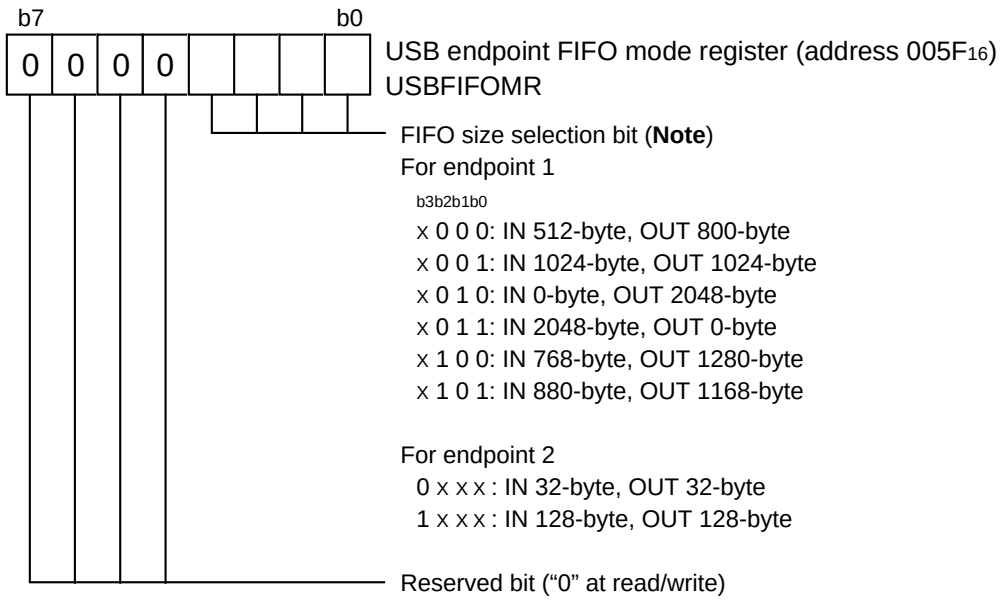


Fig. 52 Structure of USB endpoint x (x = 0 to 4) FIFO register

[USB Endpoint FIFO Mode Selection Register] USBFIFOMR

This register determines IN/OUT FIFO size mode for endpoint 1 or endpoint 2. This register is invalid when using endpoint 0, 3, or 4.



Note: The value set into “x” is invalid.

Fig. 53 Structure of USB endpoint FIFO mode selection register

MASTER CPU BUS INTERFACE

The 7641 group internally has a 2-byte bus interface which control signals from the host CPU side can operate (slave mode).
This bus interface allows the 7641 group to be directly connected with a R/W type of CPU bus or a $\overline{RD}$ and $\overline{WR}$ separated type of CPU bus. Figure 56 shows the block diagram of master CPU bus interface function.

The data bus buffer function I/O pins ($P5_2$ – $P5_7$, P6, $P7_2$–$P7_4$) also function as the normal I/O ports. When the Master CPU Bus Interface Enable bit of Data Bus Buffer Control Register (bit 6 of address $004A_{16}$) is "0", these pins become the normal I/O ports. When it is "1", these pins become the master CPU bus interface function pins.
Additionally, when using the master CPU bus interface function, set port P6 to input mode by setting "00_{16}" into its port direction register (address 0015_{16}).

The selection of either the single data bus buffer mode, which uses 1 byte: data bus buffer 0 only, or the double data bus buffer mode, which uses 2 bytes: data bus buffer 0 and data bus buffer 1, is performed by the Data Bus Buffer Function Select Bit of Data Bus Buffer Control Register 1 (bit 7 of address $004E_{16}$). Port $P7_2$ becomes $\overline{S_1}$ input pin in the double data bus buffer mode.
When data is written from the host CPU side, an input buffer full interrupt occurs. When data is read from the host CPU, an output buffer empty interrupt occurs. The 7641 group shares two input buffer full interrupt requests and two output buffer empty interrupt requests as shown in Figure 54, respectively.

The 7641 group can also operate the master CPU bus interface connecting with the Built-in DMAC. This could transfer a large amount of data fast.
An input signal level of data bus buffer function input pins can be selected between a CMOS level and a TTL level. Set it using the Master CPU Bus Input Level Select Bit of Port Control Register (address 0010_{16})
.

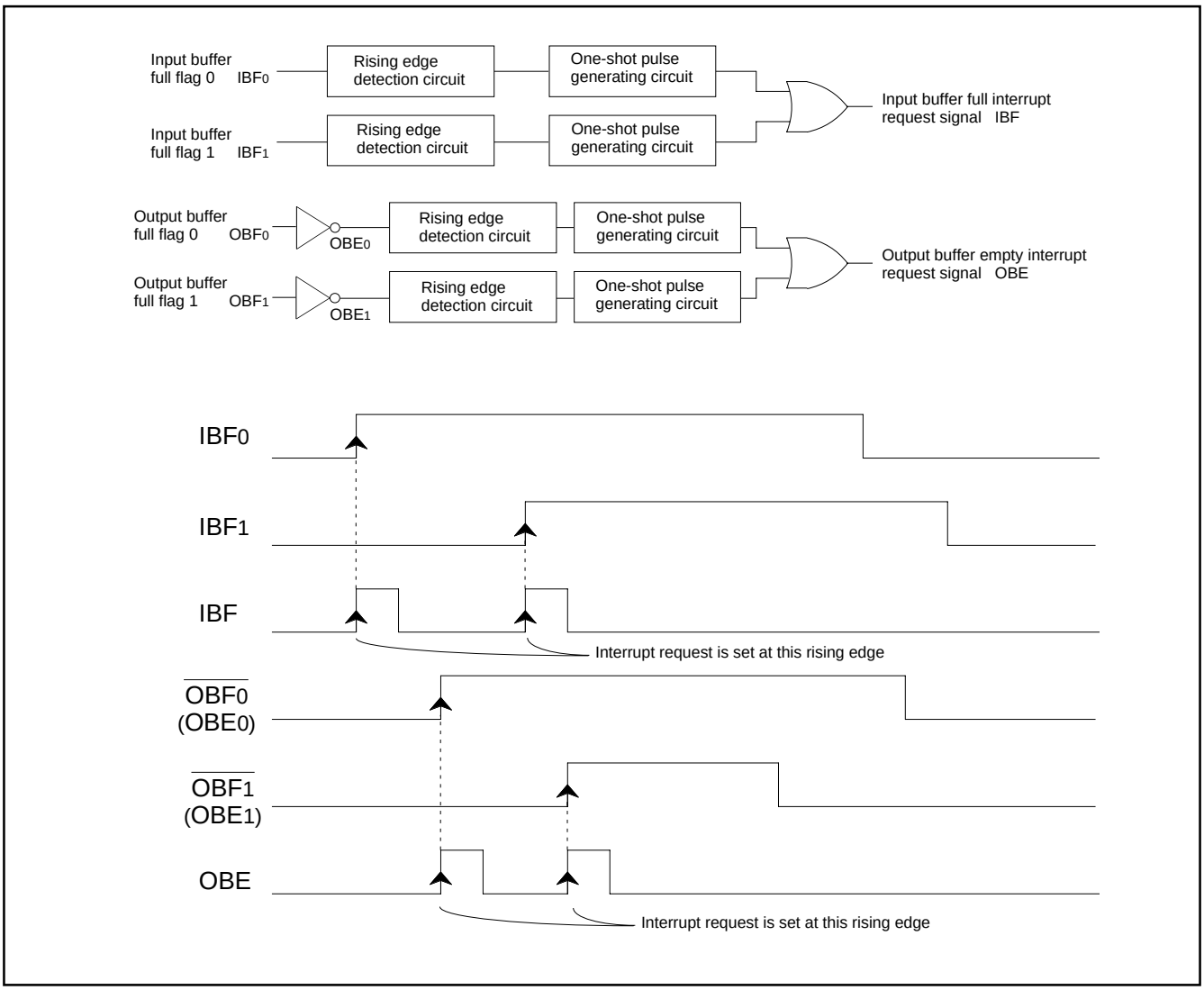


Fig. 54 Interrupt request circuit of data bus buffer

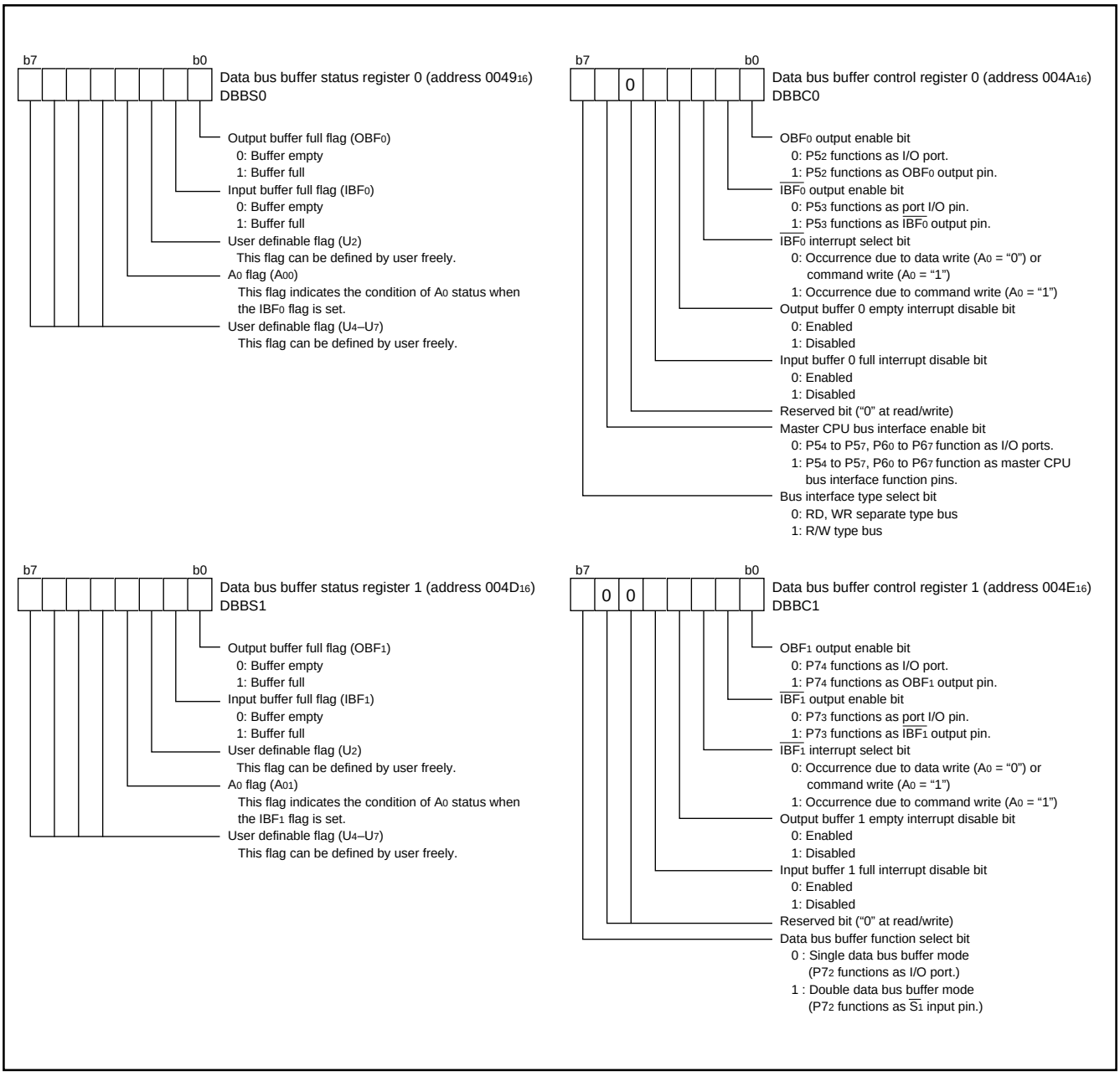


Fig. 55 Structure of master CPU bus interface related registers

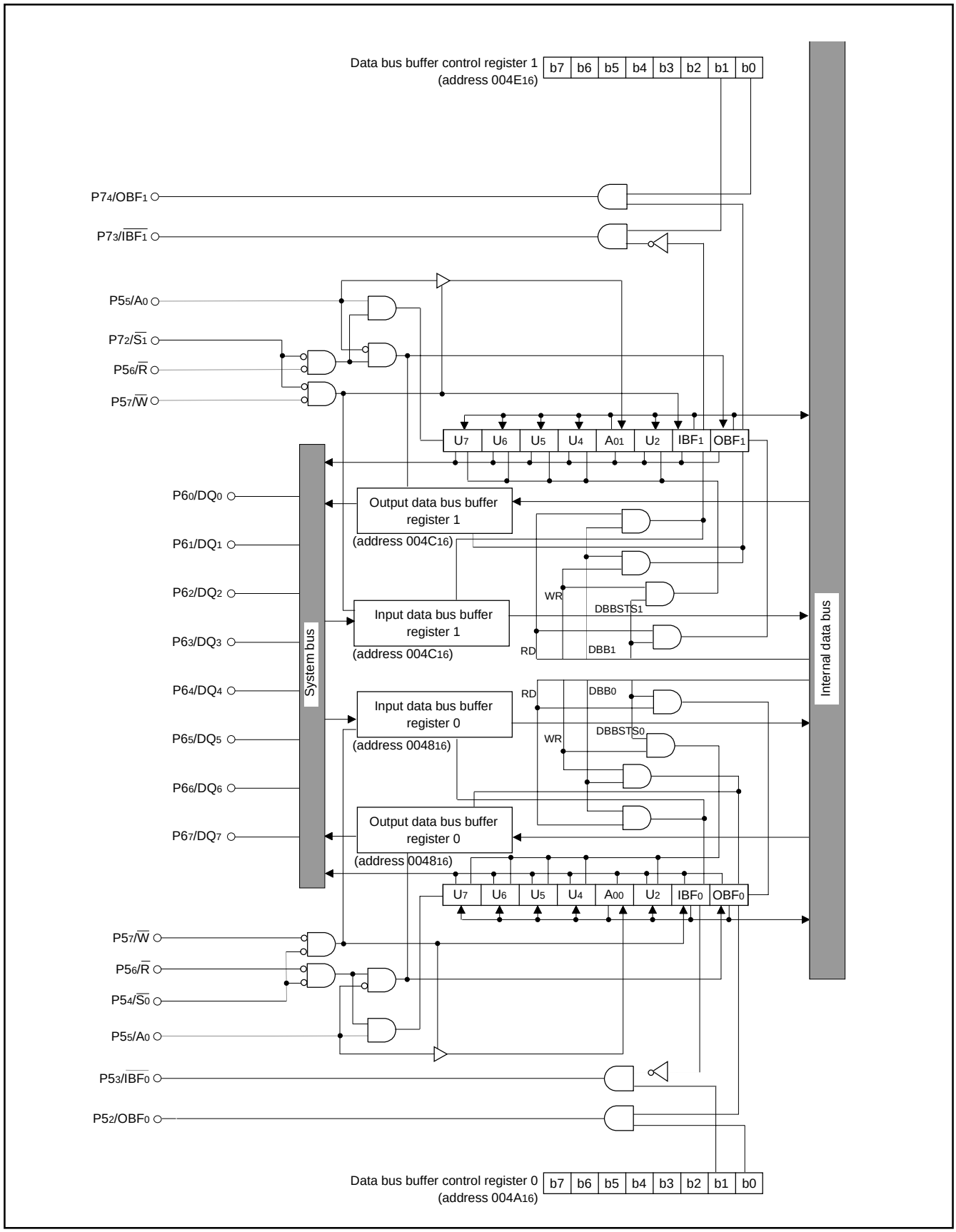


Fig. 56 Master CPU bus interface block diagram

[Data Bus Buffer Status Register 0, 1 (DBBS0, DBBS1)] 0049_{16}, $004D_{16}$

The data bus buffer status registers 0, 1 consist of eight bits each. Bits 0, 1, and 3 are read-only bits and indicate the status of the data bus buffer. Bits 2, 4, 5, 6, and 7 are user definable flags which can be programed, and can be read/written. The host CPU can only read this register when the A_0 pin is set to "H".

•Bit 0: Output buffer full flag OBF_0, OBF_1

When writing data to the output data bus buffer, this flag is set to "1". When reading the output data bus buffer from the host CPU, this flag is cleared to "0".

•Bit 1: Input buffer full flag IBF_0, IBF_1

When writing data from the host CPU to the input data bus buffer, this flag is set to "1". When reading the input data bus buffer from the slave CPU side, this flag is are cleared to "0".

•Bit 3: A0 flag A_{00}, A_{01}

When writing data from the host CPU to the input data bus buffer, the level of the A_0 pin is latched.

[Input Data Bus Buffer Registers 0, 1 ($DBBIN_0$, $DBBIN_1$)] 0048_{16}, $004C_{16}$

Data on the data bus is latched to $DBBIN_0$ or $DBBIN_1$ by writing request from the host CPU. Data of DBBINs can be read from the Data Bus Buffer Registers (address 0048_{16} or $004C_{16}$) on the SFR area.

[Output Data Bus Buffer Registers 0, 1 ($DBBOUT_0$, $DBBOUT_1$)] 0048_{16}, $004C_{16}$

When writing data to the Data Bus Buffer Registers (address 0048_{16} or $004C_{16}$) on the SFR area, data is set to $DBBOUT_0$ or $DBBOUT_1$. Data of DBBOUTs is output onto the data bus by performing the reading request from the host CPU when the A_0 pin is set to "L".

Table 8 Function description of control I/O pins of master CPU bus interface

Pin	Name	OBF_0 output enable bit	$\overline{IBF_0}$ output enable bit	OBF_1 output enable bit	$\overline{IBF_1}$ output enable bit	Input/ Output	Functions
$P5_2/OBF_0$	OBF_0	1	0	0	0	Output	Status output signal. OBF_0 signal is output.
$P5_3/\overline{IBF_0}$	$\overline{IBF_0}$	0	1	0	0	Output	Status output signal. $\overline{IBF_0}$ signal is output.
$P5_4/\overline{S_0}$	$\overline{S_0}$	—	—	—	—	Input	Chip select input. This is used for selecting the data bus buffer, which is selected at “L” level.
$P5_5/A_0$	A_0	—	—	—	—	Input	Address input. This is used for selecting DBBSTS and DBBOUT when the host CPU reads. This is used for distinguishing command from data when the host CPU writes.
$P5_6/\overline{R}$ (E)	$\overline{R}$ (E)	—	—	—	—	Input	This is a timing signal for reading data from the data bus buffer to the host CPU.
$P5_7/\overline{W}$ (R/$\overline{W}$)	$\overline{W}$ (R/$\overline{W}$)	—	—	—	—	Input	This is a timing signal for writing data to the data bus buffer by the host CPU.
$P7_2/\overline{S_1}$	$\overline{S_1}$	—	—	—	—	Input	Chip select input. This is used for selecting the data bus buffer, which is selected at “L” level.
$P7_3/\overline{IBF_1}/\overline{HLDA}$	$\overline{IBF_1}$	0	0	0	1	Output	Status output signal. $\overline{IBF_1}$ signal is output.
$P7_4/OBF_1$	OBF_1	0	0	1	0	Output	Status output signal. OBF_1 signal is output.

COUNT SOURCE GENERATOR

The 7641 Group has a built-in special count source generator, SCSG. This generator consists of two 8-bit timers: SCSG1 and SCSG2. The output of the special count source generator can be used as a clock source for the timer X, serial I/O and two UARTs.

SCSG Operation

Timers SCSG1 and SCSG2 are both down count timers. When the count reaches “0”, an underflow occurs at the next count source rising edge and the contents of the corresponding timer latch are loaded to the timer. The division ratio of each SCSG-x timer is given by 1 / (n+1), where “n” is the value set to the SCSG-x timer. The output of Timer SCSG1 is ANDed with the original clock (ϕ) to make a count source for Timer SCSG2.

The SCSG output is Clock SCSGCLK. The frequency is calculated as follows:

$$SCSGCLK = \phi \times \{n1 / (n1+1)\} \times \{1 / (n2+1)\}$$

n1: value set to SCSG1
n2: value set to SCSG2

If the SCSG1 Count Stop Bit (SCSGM1) is set to “1”, or Timer SCSG1 is set to “0”, the SCSG1 count stops. When this happens, the count source for Timer SCSG2 becomes ϕ.

Data Write Control

When the SCSG1 Data Write Control Bit or SCSG2 Data Write Control Bit is set to “0”, and data is written to the SCSG-x timer; the data is written to the corresponding latch and timer at the same time. When that bit is set to “1”, the data is only written to the latch.

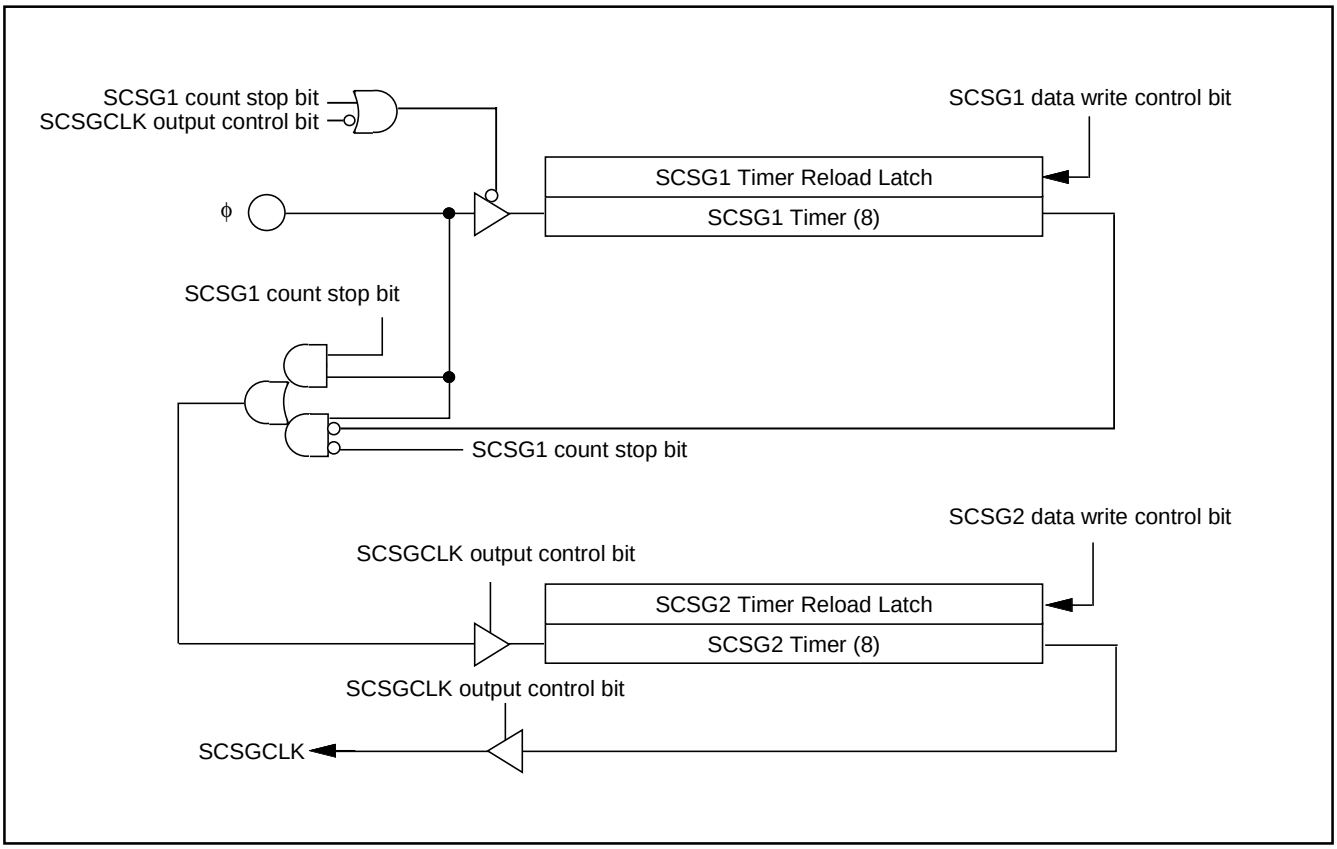


Fig. 57 Special count source generator block diagram

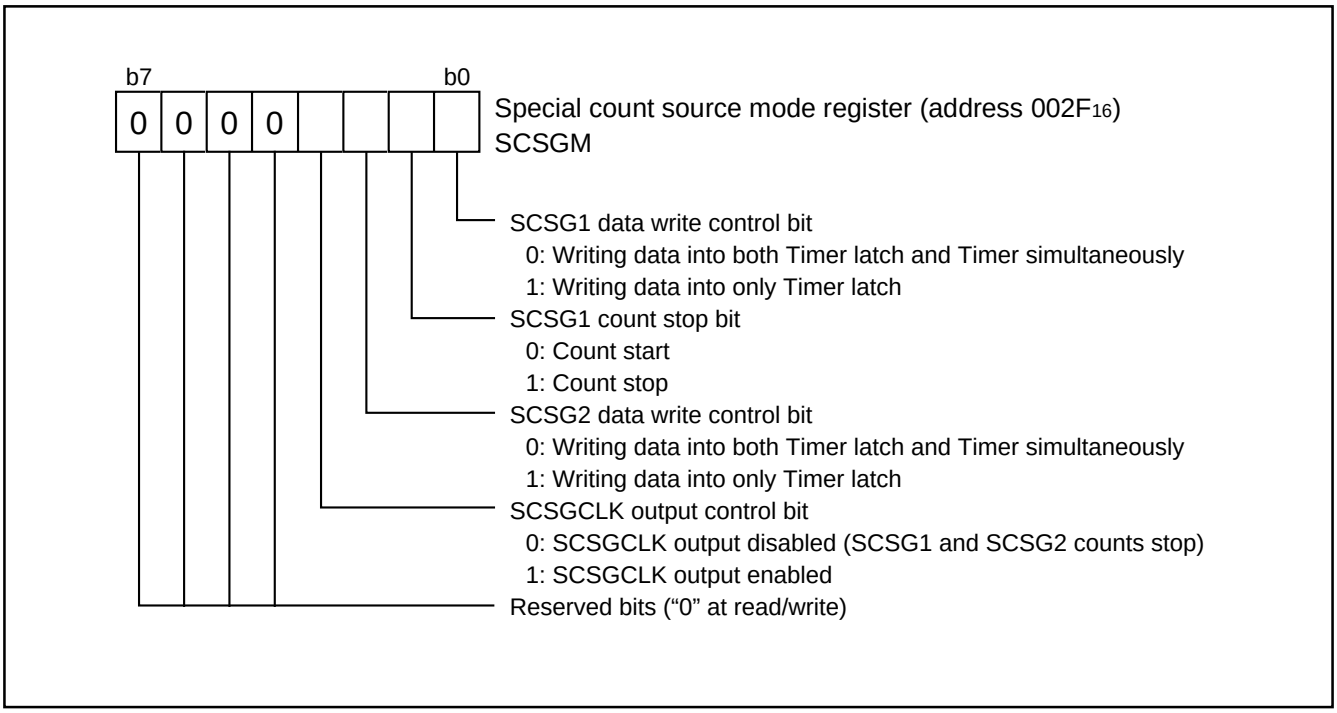


Fig. 58 Structure of special count source generator mode register

FREQUENCY SYNTHESIZER (PLL)

The frequency synthesizer generates the 48 MHz clock required by f_{USB} and f_{SYN}, which are multiples of the external input reference $f(X_{IN})$. Figure 59 shows the block diagram for the frequency synthesizer circuit.

The Frequency Synthesizer Input Bit selects either $f(X_{IN})$ or $f(X_{CIN})$ as an input clock f_{IN} for the frequency synthesizer.

The Frequency Synthesizer Multiply Register 2 (FSM2: address $006E_{16}$) divides f_{IN} to generate f_{PIN}, where

$f_{PIN} = f_{IN} / 2(n + 1)$, n: value set to FSM2.

When the value of Frequency Synthesizer Multiply Register 2 is set to 255, the division is not performed and f_{PIN} will equal f_{IN}.

f_{VCO} is generated according to the contents of Frequency Synthesizer Multiply Register 1 (FSM1: address $006D_{16}$), where

$f_{VCO} = f_{PIN} \times \{2(n + 1)\}$, n: value set to FSM1.

Set the value of FSM1 so that the value of f_{VCO} is 48 MHz.

f_{SYN} is generated according to the contents of the Frequency Synthesizer Divide Register (FSD: address $006F_{16}$), where

$f_{SYN} = f_{VCO} / 2(m + 1)$, m: value set to FSD.

When the value of the Frequency Synthesizer Divide Register is set to 255, the division is not performed and f_{SYN} becomes invalid.

[Frequency Synthesizer Control Register] FSC

Setting the Frequency Synthesizer Enable Bit (FSE) to "1" enables the frequency synthesizer. When the Frequency Synthesizer Lock Status Bit (LS) is "1" in the frequency synthesizer enabled, this indicates that f_{SYN} and f_{VCO} have correct frequencies.

■Notes

Make sure to connect a low-pulse filter to the LPF pin when using the frequency synthesizer. In addition, please refer to "Programming Notes: Frequency Synthesizer" when recovering from a Hardware Reset.

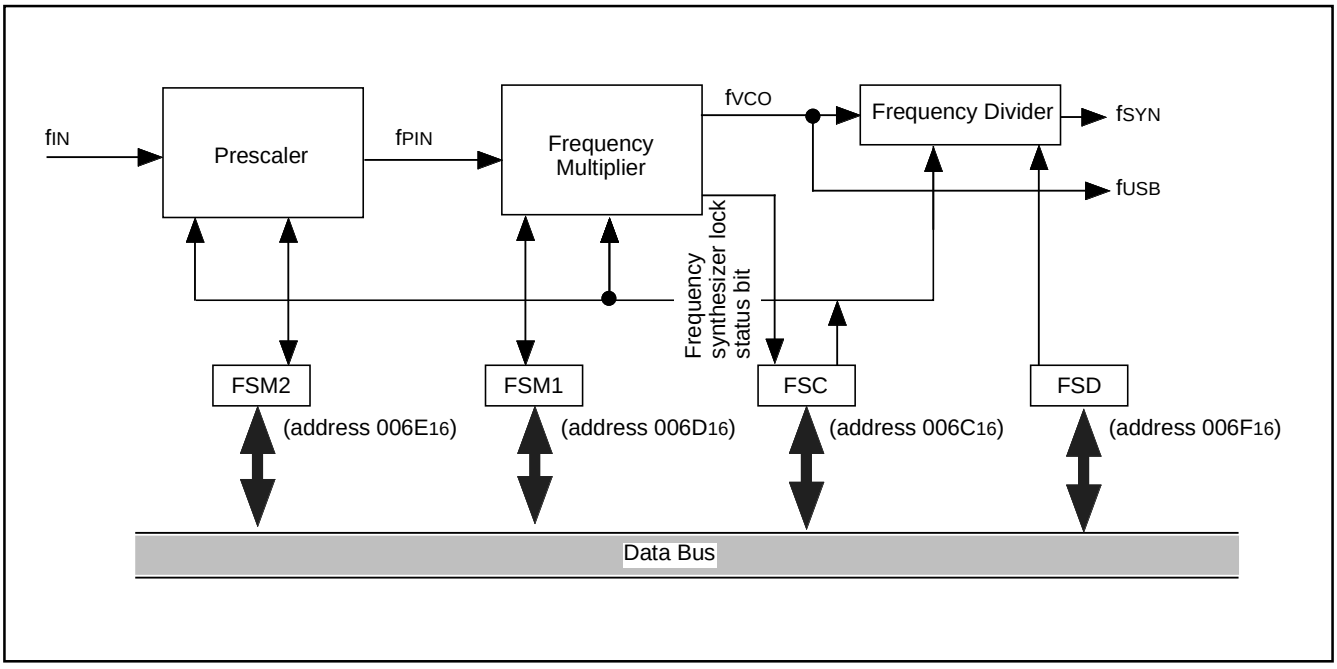
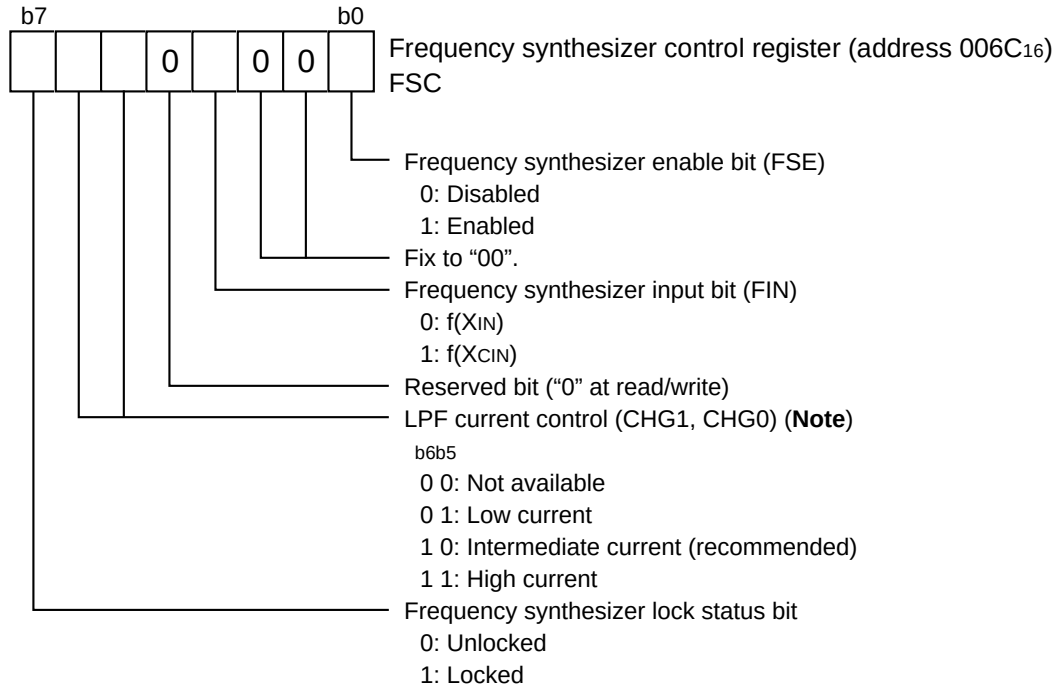


Fig. 59 Frequency synthesizer block diagram



Note: Bits 6 and 5 are set to (bit 6, bit 5) = (1, 1) at reset.
When using the frequency synthesizer, we recommend to set to (bit 6, bit 5) = (1, 0) after locking the frequency synthesizer.

Fig. 60 Structure of frequency synthesizer control register

RESET CIRCUIT

To reset the microcomputer, $\overline{\text{RESET}}$ pin should be held at an “L” level for 20 cycles or more of ϕ. Then the $\overline{\text{RESET}}$ pin is returned to an “H” level, and reset is released. They must be performed when the power source voltages are between 3.00 V and 3.60 V or 4.15 V and 5.25 V.

After the reset is completed, the program starts from the address contained in address FFFA_{16} (high-order byte) and address FFFB_{16} (low-order byte).

After oscillation has restarted, the timers 1 and 2 secures waiting time for the internal clock ϕ oscillation stabilized automatically by setting the timer 1 to “FF_{16}” and timer 2 to “01_{16}”. The internal clock ϕ retains “H” level until Timer 2’s underflow and it cannot be supplied until the underflow.

The pins state during reset are follows:

•When CNVss = “H”

Ports P0, P1, $P3_3$ to $P3_7$: Outputting

Pins other than above mentioned ports : Inputting

•When CNVss = “L”

All pins : Inputting.

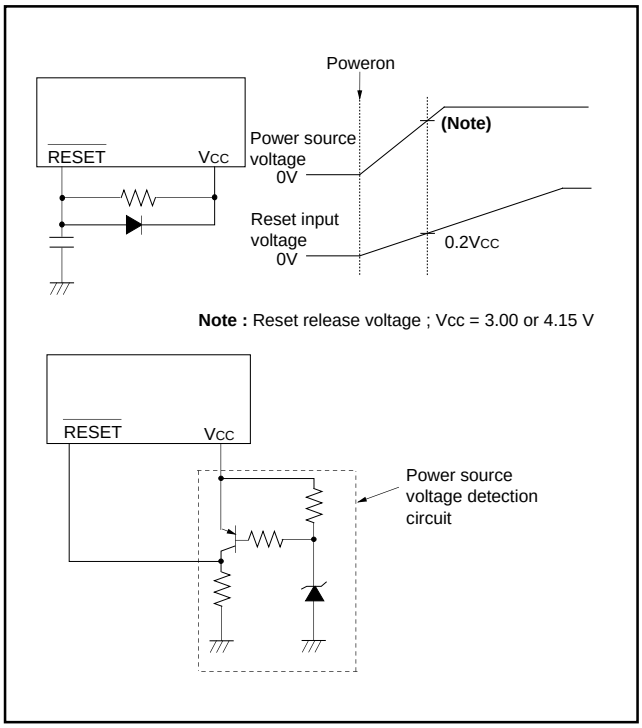


Fig. 61 Reset circuit example

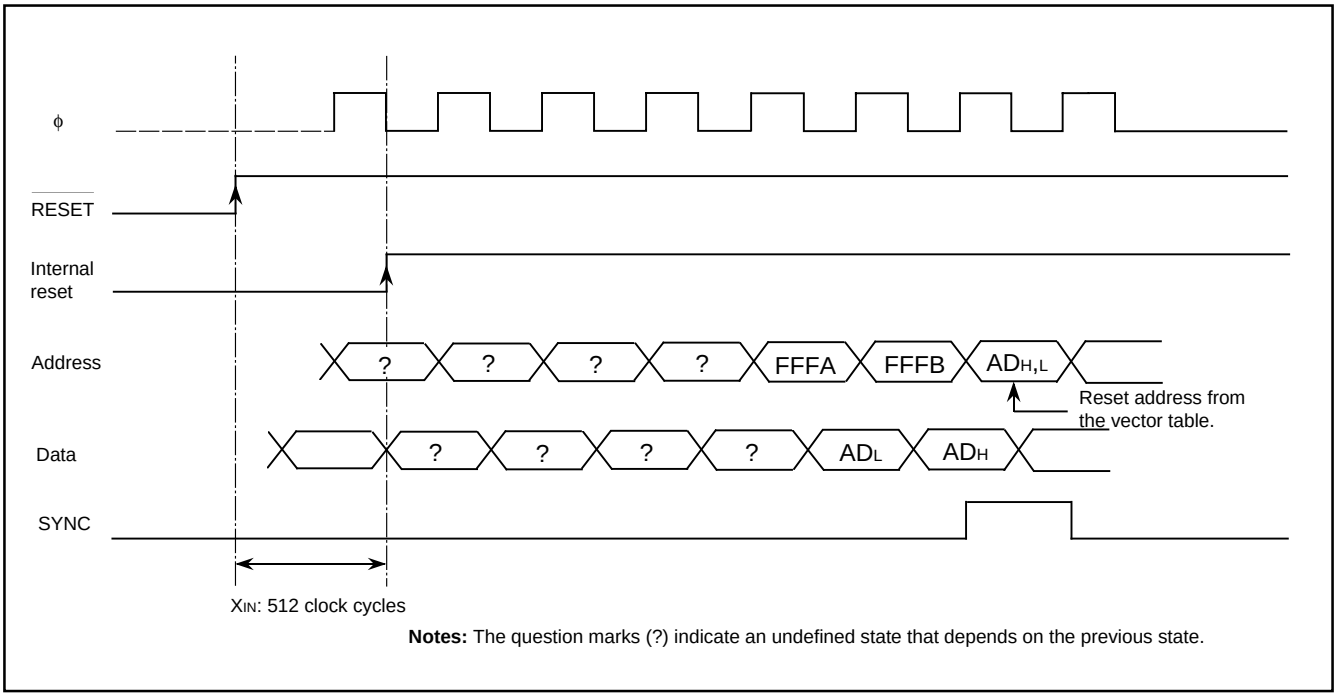


Fig. 62 Reset sequence

		Address	Register contents
(1)	CPU mode register A (CPUA)	0000_{16}	0 0 0 0 1 1 0 0
(2)	CPU mode register B (CPUB)	0001_{16}	1 0 0 0 0 0 1 1
(3)	Interrupt request register A (IREQA)	0002_{16}	00_{16}
(4)	Interrupt request register B (IREQB)	0003_{16}	00_{16}
(5)	Interrupt request register C (IREQC)	0004_{16}	00_{16}
(6)	Interrupt control register A (ICONA)	0005_{16}	00_{16}
(7)	Interrupt control register B (ICONB)	0006_{16}	00_{16}
(8)	Interrupt control register C (ICONC)	0007_{16}	00_{16}
(9)	Port P0 (P0)	0008_{16}	00_{16}
(10)	Port P0 direction register (P0D)	0009_{16}	00_{16}
(11)	Port P1 (P1)	$000A_{16}$	00_{16}
(12)	Port P1 direction register (P1D)	$000B_{16}$	00_{16}
(13)	Port P2 (P2)	$000C_{16}$	00_{16}
(14)	Port P2 direction register (P2D)	$000D_{16}$	00_{16}
(15)	Port P3 (P3)	$000E_{16}$	00_{16}
(16)	Port P3 direction register (P3D)	$000F_{16}$	00_{16}
(17)	Port control register (PTC)	0010_{16}	00_{16}
(18)	Interrupt polarity select register (IPOL)	0011_{16}	00_{16}
(19)	Port P2 pull-up control register (PUP2)	0012_{16}	00_{16}
(20)	USB control register (USBC)	0013_{16}	00_{16}
(21)	Port P6 (P6)	0014_{16}	00_{16}
(22)	Port P6 direction register (P6D)	0015_{16}	00_{16}
(23)	Port P5 (P5)	0016_{16}	00_{16}
(24)	Port P5 direction register (P5D)	0017_{16}	00_{16}
(25)	Port P4 (P4)	0018_{16}	00_{16}
(26)	Port P4 direction register (P4D)	0019_{16}	00_{16}
(27)	Port P7 (P7)	$001A_{16}$	00_{16}
(28)	Port P7 direction register (P7D)	$001B_{16}$	00_{16}
(29)	Port P8 (P8)	$001C_{16}$	00_{16}
(30)	Port P8 direction register (P8D)	$001D_{16}$	00_{16}
(31)	Clock control register (CCR)	$001F_{16}$	00_{16}
(32)	Timer XL (TXL)	0020_{16}	FF_{16}
(33)	Timer XH (TXH)	0021_{16}	FF_{16}
(34)	Timer YL (TYL)	0022_{16}	FF_{16}
(35)	Timer YH (TYH)	0023_{16}	FF_{16}
(36)	Timer 1 (T1)	0024_{16}	FF_{16}
(37)	Timer 2 (T2)	0025_{16}	0 0 0 0 0 0 0 1
(38)	Timer 3 (T3)	0026_{16}	FF_{16}
(39)	Timer X mode register (TXM)	0027_{16}	00_{16}
(40)	Timer Y mode register (TYM)	0028_{16}	00_{16}
(41)	Timer 123 mode register (T123M)	0029_{16}	00_{16}
(42)	Serial I/O control register 1 (SIOCON1)	$002B_{16}$	0 1 0 0 0 0 0 0
(43)	Serial I/O control register 2 (SIOCON2)	$002C_{16}$	0 0 0 1 1 0 0 0
(44)	Special count source generator 1 (SCSG1)	$002D_{16}$	FF_{16}
(45)	Special count source generator 2 (SCSG2)	$002E_{16}$	FF_{16}
(46)	Special count source mode register (SCSGM)	$002F_{16}$	00_{16}
(47)	UART1 mode register (U1MOD)	0030_{16}	00_{16}
(48)	UART1 status register (U1STS)	0032_{16}	0 0 0 0 0 0 1 1
(49)	UART1 control register (U1CON)	0033_{16}	00_{16}
(50)	UART1 RTS control register (U1RTSC)	0036_{16}	1 0 0 0 0 0 0 0
(51)	UART2 mode register (U2MOD)	0038_{16}	00_{16}
(52)	UART2 status register (U2STS)	$003A_{16}$	0 0 0 0 0 0 1 1
(53)	UART2 control register (U2CON)	$003B_{16}$	00_{16}
(54)	UART2 RTS control register (U2RTSC)	$003E_{16}$	1 0 0 0 0 0 0 0
(55)	DMAC index and status register (DMAIS)	$003F_{16}$	00_{16}
(56)	DMAC channel x mode register 1 (DMAx1)	0040_{16}	00_{16}
(57)	DMAC channel x mode register 2 (DMAx2)	0041_{16}	00_{16}
(58)	DMAC channel x source register Low (DMAxSL)	0042_{16}	00_{16}
(59)	DMAC channel x source register High (DMAxSH)	0043_{16}	00_{16}
(60)	DMAC channel x destination register Low (DMAxDL)	0044_{16}	00_{16}
(61)	DMAC channel x destination register High (DMAxDH)	0045_{16}	00_{16}
(62)	DMAC channel x transfer count register Low (DMAxCL)	0046_{16}	00_{16}
(63)	DMAC channel x transfer count register High (DMAxCH)	0047_{16}	00_{16}
(64)	Data bus buffer register 0 (DBB0)	0048_{16}	00_{16}
(65)	Data bus buffer status register 0 (DBBS0)	0049_{16}	00_{16}
(66)	Data bus buffer control register 0 (DBBC0)	$004A_{16}$	00_{16}
(67)	Data bus buffer register 1 (DBB1)	$004C_{16}$	00_{16}
(68)	Data bus buffer status register 1 (DBBS1)	$004D_{16}$	00_{16}
(69)	Data bus buffer control register 1 (DBBC1)	$004E_{16}$	00_{16}
(70)	USB address register (USBA)	0050_{16}	00_{16}
(71)	USB power management register (USBPM)	0051_{16}	00_{16}
(72)	USB interrupt status register 1 (USBIS1)	0052_{16}	00_{16}
(73)	USB interrupt status register 2 (USBIS2)	0053_{16}	00_{16}
(74)	USB interrupt enable register 1 (USBIE1)	0054_{16}	FF_{16}
(75)	USB interrupt enable register 2 (USBIE2)	0055_{16}	0 0 1 1 0 0 1 1
(76)	USB frame number register Low (USBSOFL)	0056_{16}	00_{16}
(77)	USB frame number register High (USBSOFH)	0057_{16}	00_{16}
(78)	USB endpoint index register (USBINDEX)	0058_{16}	00_{16}
(79)	USB endpoint x IN control register (IN_CSR)	0059_{16}	00_{16}
(80)	USB endpoint x OUT control register (OUT_CSR)	$005A_{16}$	00_{16}
(81)	USB endpoint x IN max. packet size register (IN_MAXP) **(Note 1)**	$005B_{16}$	0 0 0 0 1 0 0 0
(82)	USB endpoint x OUT max. packet size register (OUT_MAXP) **(Note 1)**	$005C_{16}$	0 0 0 0 1 0 0 0
(83)	USB endpoint x OUT write count register Low (WRT_CNTL)	$005D_{16}$	00_{16}
(84)	USB endpoint x OUT write count register High (WRT_CNTH)	$005E_{16}$	00_{16}
(85)	USB endpoint FIFO mode register (USBFIFOMR)	$005F_{16}$	00_{16}
(86)	Flash memory control register (FMCR) **(Note 3)**	$006A_{16}$	0 0 0 0 0 0 0 1
(87)	Frequency synthesizer control register (FSC)	$006C_{16}$	0 1 1 0 0 0 0 0
(88)	Frequency synthesizer multiply register 1 (FSM1)	$006D_{16}$	FF_{16}
(89)	Frequency synthesizer multiply register 2 (FSM2)	$006E_{16}$	FF_{16}
(90)	Frequency synthesizer divide register (FSM2)	$006F_{16}$	FF_{16}
(91)	ROM code protect control register (ROMCP) **(Note 3)**	FFC916	FF_{16}
(92)	Processor status register	(PS)	x x x x x 1 x x
(93)	Program counter	(PC_H)	$FFFB_{16}$ contents
		(PC_L)	$FFFA_{16}$ contents

X : Not fixed

Notes 1: When using the endpoint 1, this contents are "01_{16}".

2: Since the initial values for other than above mentioned registers and RAM contents are indefinite at reset, they must be set.

3: The flash memory control register and the ROM code protect control register exists in the flash memory version only.

Fig. 63 Internal status at reset

CLOCK GENERATING CIRCUIT

The 7641 group has two built-in oscillation circuits. An oscillation circuit can be formed by connecting a resonator between XIN and XOUT (XCIN and XCOUT). Use the circuit constants in accordance with the resonator manufacturer's recommended values. No external resistor is needed between XIN and XOUT since a feed-back resistor exists on-chip. However, an external feed-back resistor is needed between XCIN and XCOUT.

When using an external clock, input the clocks to the XIN or XCIN pin and leave the XOUT or XCOUT pin open.

Immediately after power on, only the XIN oscillation circuit starts oscillating, and XCIN and XCOUT pins function as I/O ports.

Frequency Control

The internal system clock can be selected among fSYN, f(XIN), f(XIN)/2, and f(XCIN). The internal clock ϕ is half the frequency of internal system clock.

(1) fSYN clock

This is made by the frequency synthesizer. f(XIN) or f(XCIN) can be selected as its input clock. See also section "FREQUENCY SYNTHESIZER".

(2) f(XIN) clock

The frequency of internal system clock is the frequency of XIN pin.

(3) f(XIN)/2 clock

The frequency of internal system clock is half the frequency of XIN pin.

(4) f(XCIN) clock

The frequency of internal system clock is the frequency of XCIN pin.

■Note

If you switch the oscillation between XIN - XOUT and XCIN - XCOUT, stabilize both XIN and XCIN oscillations. The sufficient time is required for the XCIN oscillation to stabilize, especially immediately after power on and at returning from the stop mode.

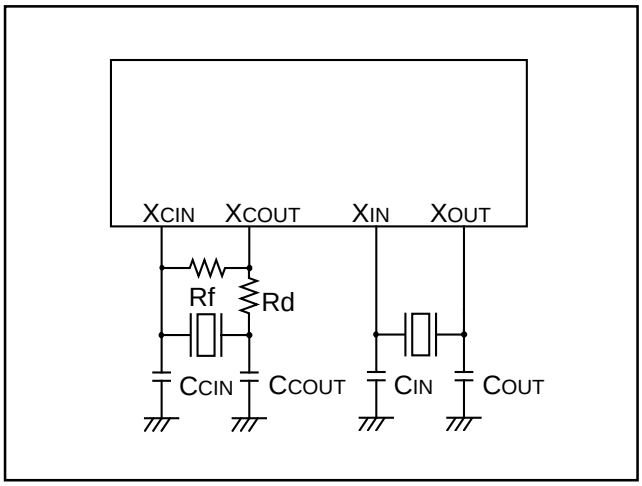


Fig. 64 Ceramic resonator or quartz-crystal oscillator external circuit

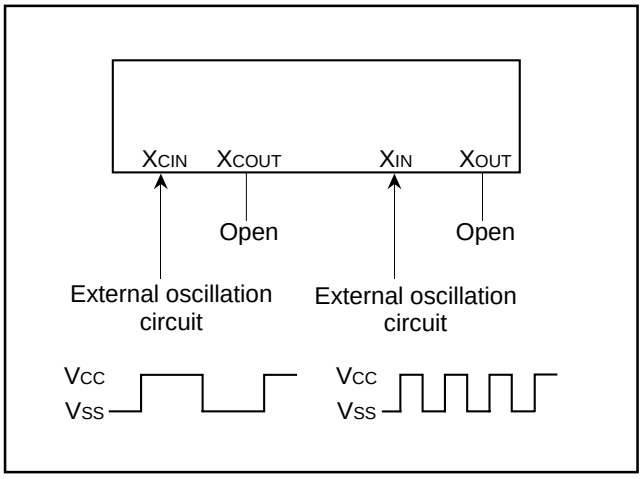


Fig. 65 External clock input circuit

MITSUBISHI ELECTRIC

(5) Low power dissipation mode

- The low power dissipation operation can be realized by stopping the main clock XIN when using f(XCIN) as the internal system clock. To stop the main clock, set the Main Clock (XIN-XOUT) Stop Bit of the CPU mode register A to "1".
- The low power dissipation operation can be realized by disabling the reversed amplifier when inputting external clocks to the XIN pin or XCIN pin. To disable the reversed amplifier, set the XCOUT Oscillation Drive Disable Bit (CCR5) or XOUT Oscillation Drive Disable Bit (CCR6) of the clock control register to "1".

Oscillation Control

(1) Stop mode

If the STP instruction is executed, the internal clock ϕ stops at "H" level, and XIN and XCIN oscillators stop. Then the timer 1 is set to "FF_{16}" and the internal clock ϕ divided by 8 is automatically selected as its count source. Additionally, the timer 2 is set to "01_{16}" and the timer 1's output is automatically selected as its count source.

Set the Timer 1 and Timer 2 Interrupt Enable Bits to disabled ("0") before executing the STP instruction. When using an external interrupt to release the stop mode, set the Interrupt Enable Bit to be used to enabled ("1") and the Interrupt Disable Flag (I) to "0".

Oscillator restarts at reset or when an external interrupt including USB resume interrupts is received, but the internal clock ϕ remains at "H" until the timer 2 underflows. The internal clock ϕ is supplied for the first time when the timer 2 underflows. Therefore make sure not to set the Timer 1 Interrupt Request Bit and Timer 2 Interrupt Request Bit to "1" before the STP instruction stops the oscillator.

(2) Wait mode

If the WIT instruction is executed, the internal clock ϕ stops at "H" level, but the oscillator does not stop. The internal clock ϕ restarts at reset or when an interrupt is received. Since the oscillator does not stop, normal operation can be started immediately after the internal clock ϕ is restarted.

Set the Interrupt Enable Bit to be used to release the wait mode to enabled ("1") and the Interrupt Disable Flag (I) to "0".

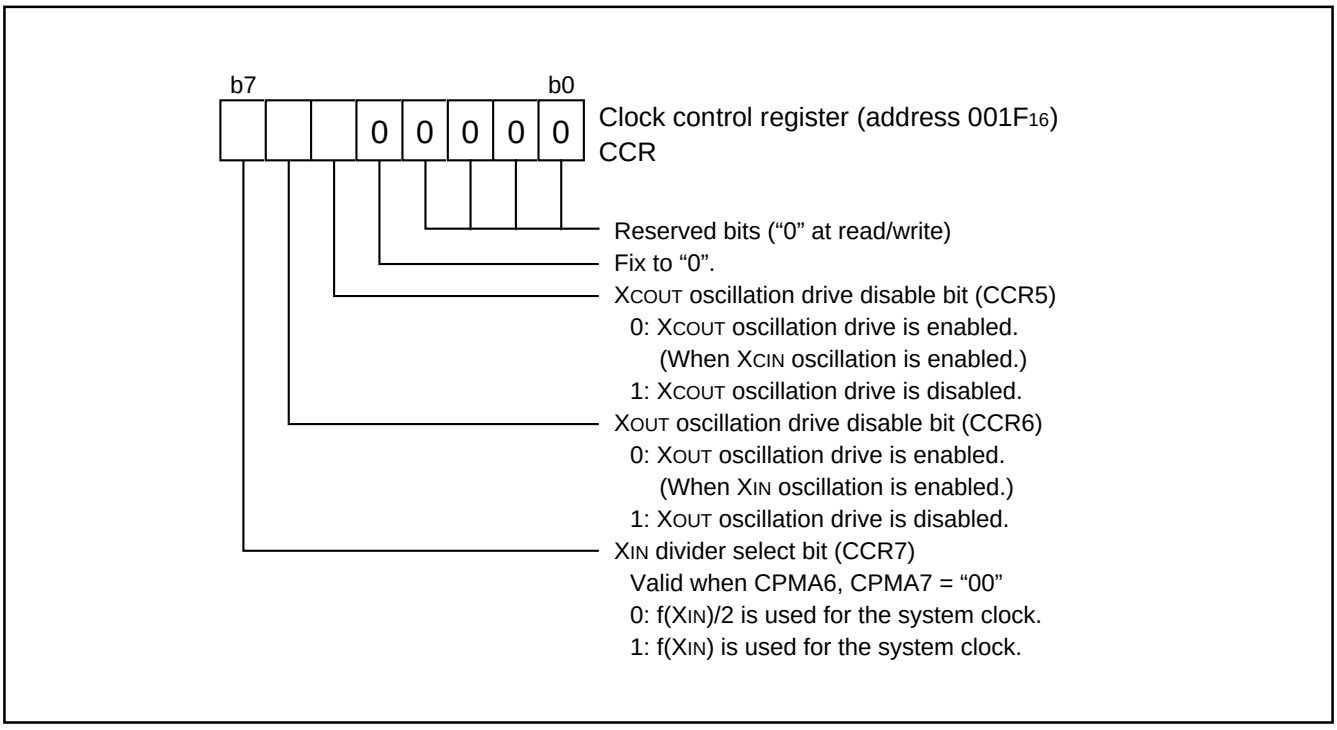


Fig. 66 Structure of clock control register

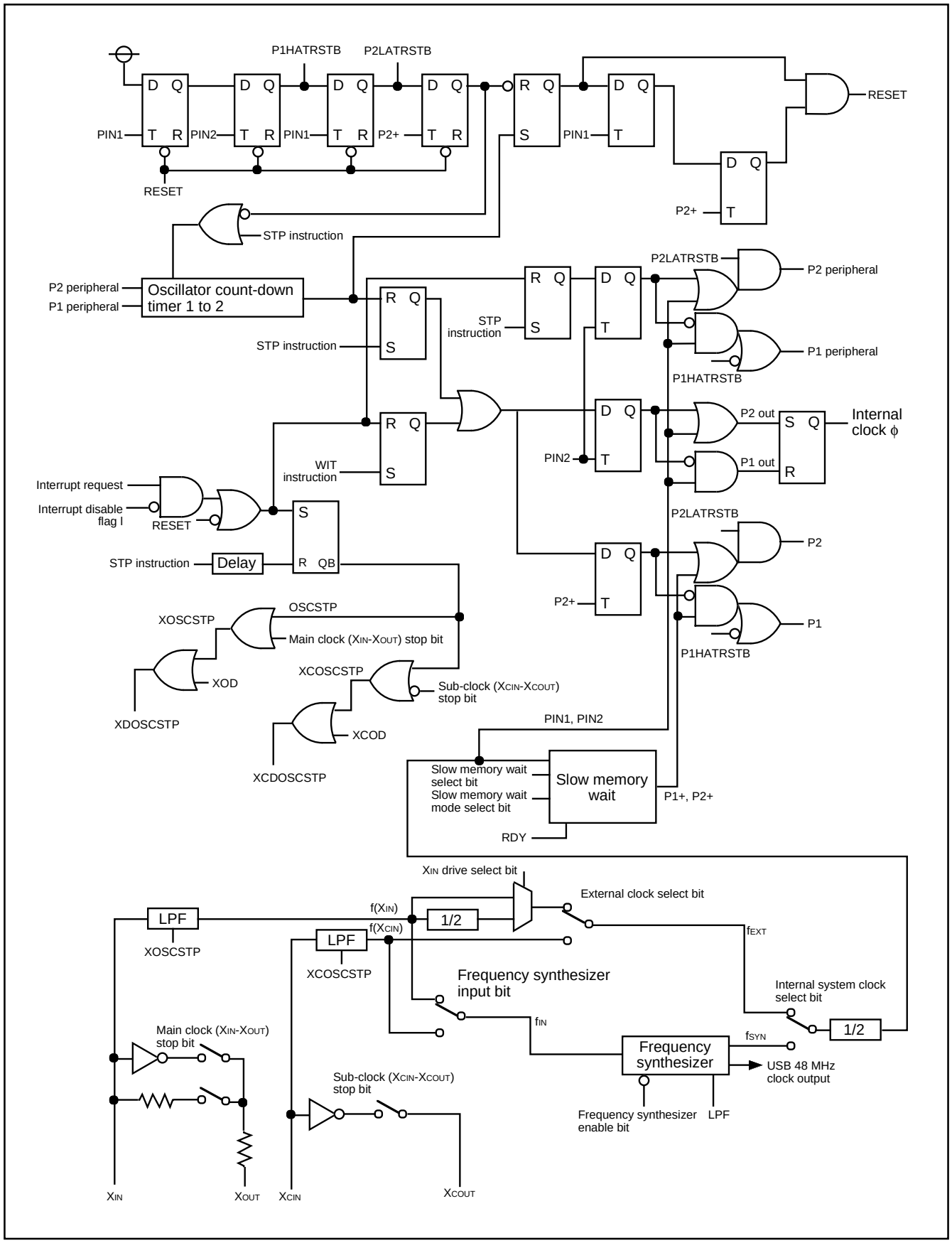
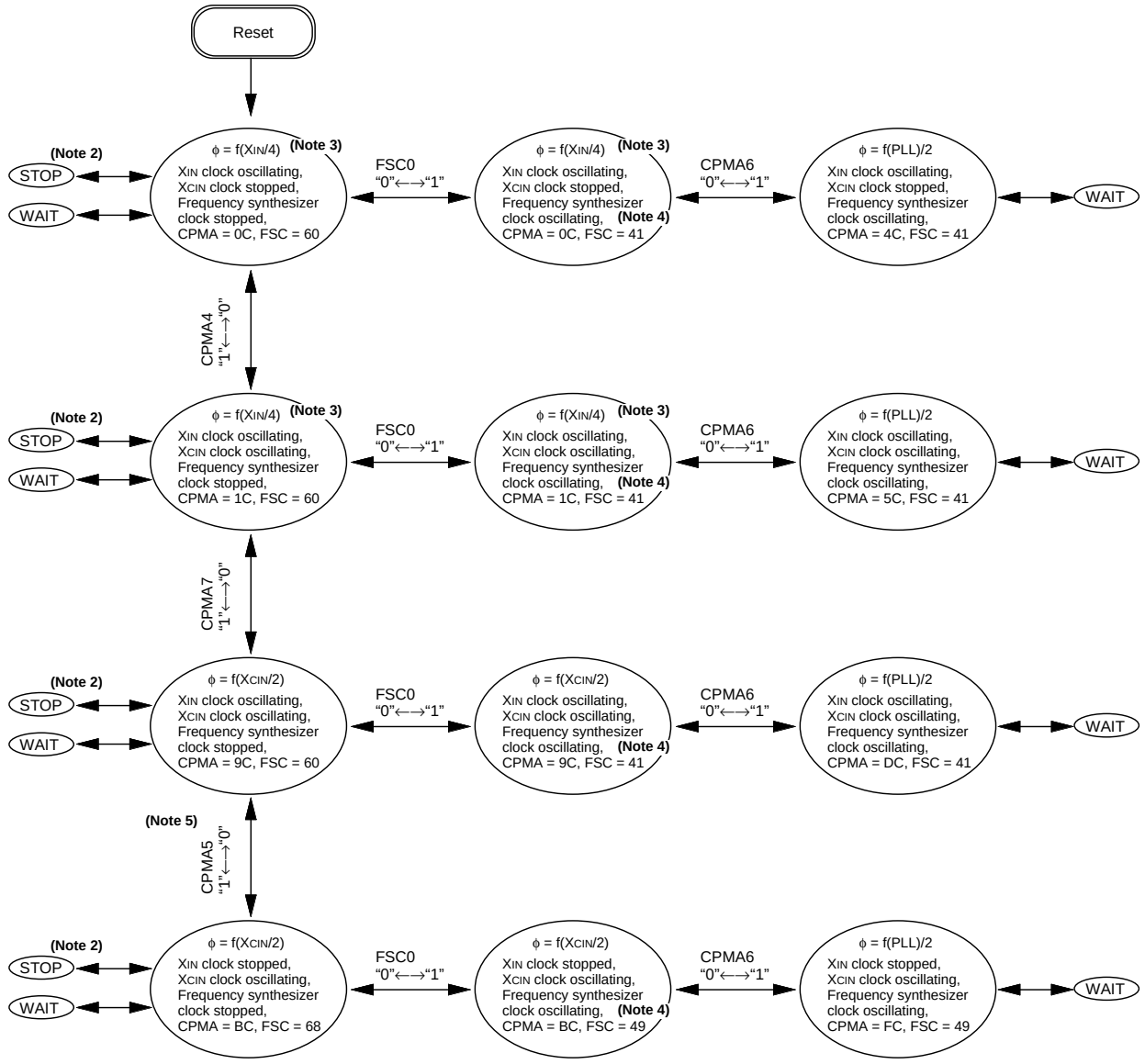


Fig. 67 Clock generating circuit block diagram



Notes 1 : Switch the mode by the allows shown between the mode blocks. (Do not switch between the modes directly without an allow.)
2 : In Stop mode, though the frequency synthesizer is not automatically disabled, the oscillator which sends clocks to the frequency synthesizer stops. Set the system clock and disable the frequency synthesizer before execution of the STP instruction.
3 : φ = f(XIN)/2 can be also used by setting the XIN divider select bit (CCR7) to "1". Then this diagram also applies to that case.
4 : The frequency synthesizer's input can be selected between XIN input and XCIN input regardless of the system clock. This diagram assumes the frequency synthesizer's input to be the system clock. Enable the oscillator to be used for the frequency synthesizer's input before enabling the frequency synthesizer.
5 : Select the XCIN input as the frequency synthesizer's input by setting the frequency synthesizer input bit (FSC3) to "1" before stopping XIN oscillation.

Remarks : This diagram assumes that:
•Stack page is page 1
•In single-chip mode
(Depending on the CPU mode register A)
•φ expresses the internal clock.

Fig. 68 State transitions of clock

PROCESSOR MODE

Single-chip mode, memory expansion mode, and microprocessor mode which is only in the mask ROM version can be selected by using the Processor Mode Bits of CPU mode register A (bits 0 and 1 of address 0000_{16}). In the memory expansion mode and microprocessor mode, a memory can be expanded externally via ports P0 to P3. In these modes, ports P0 to P3 lose their I/O port functions and become bus pins.

The port direction registers corresponding to those ports become external memory areas.

Table 9 Port functions in memory expansion mode and microprocessor mode

Port Name	Function
Port P0	Outputs low-order 8 bits of address.
Port P1	Outputs high-order 8 bits of address.
Port P2	Operates as I/O pins for data D_7 to D_0 (including instruction code).
Port P3	$P3_0$ is the RDY input pin. $P3_1$ and $P3_2$ function only as output pins $P3_3$ is the DMA_{OUT} output pin. $P3_4$ is the ϕ_{OUT} output pin. $P3_5$ is the $SYNC_{OUT}$ output pin. $P3_6$ is the $\overline{WR}$ output pin, and $P3_7$ is the $\overline{RD}$ output pin.
Port P4	$P4_0$ is the $\overline{EDMA}$ pin.

(1) Single-chip mode

Select this mode by resetting the MCU with CNV_{SS} connected to V_{SS}.

(2) Memory expansion mode

Select this mode by setting the Processor Mode Bits (b1, b0) to "01" in software with CNV_{SS} connected to V_{SS}. This mode enables external memory expansion while maintaining the validity of the internal ROM.

(3) Microprocessor mode

Select this mode by resetting the MCU with CNV_{SS} connected to V_{CC}, or by setting the Processor Mode Bits (b1, b0) to "10" in software with CNV_{SS} connected to V_{SS}. In the microprocessor mode, the internal ROM is no longer valid and an external memory must be used.

Do not set this mode in the flash memory version.

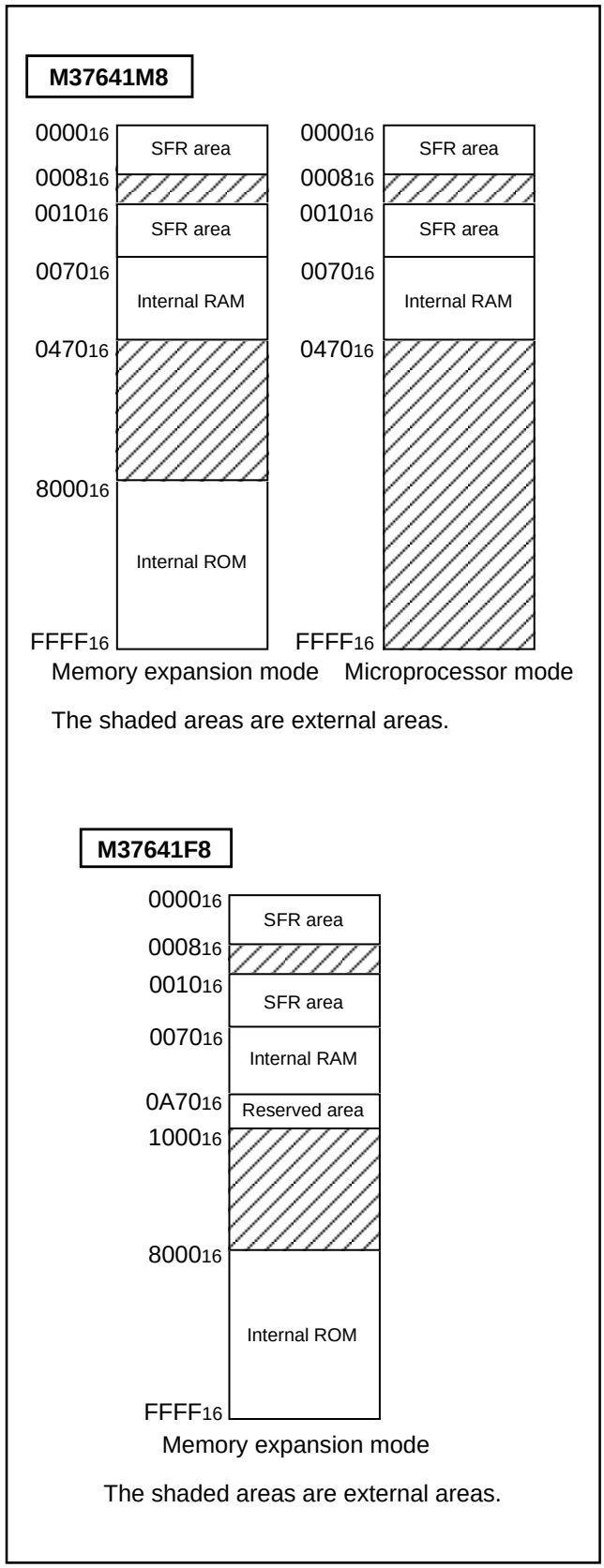
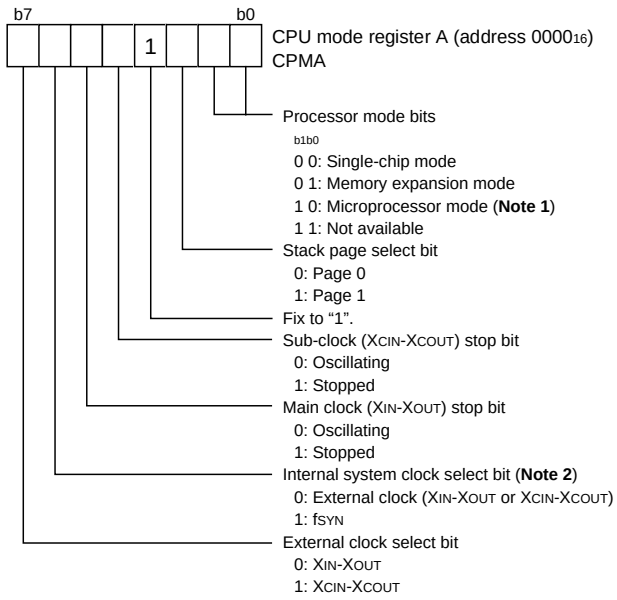


Fig. 69 Memory maps in processor modes other than single-chip mode



Notes 1: This is not available in the flash memory version.
2: When (CPMA 6, 7) = (0, 0), the internal system clock can be selected between f(XIN) or f(XIN)/2 by CCR7.
The internal clock ϕ is the internal system clock divided by 2.

Fig. 70 Structure of CPU mode register A

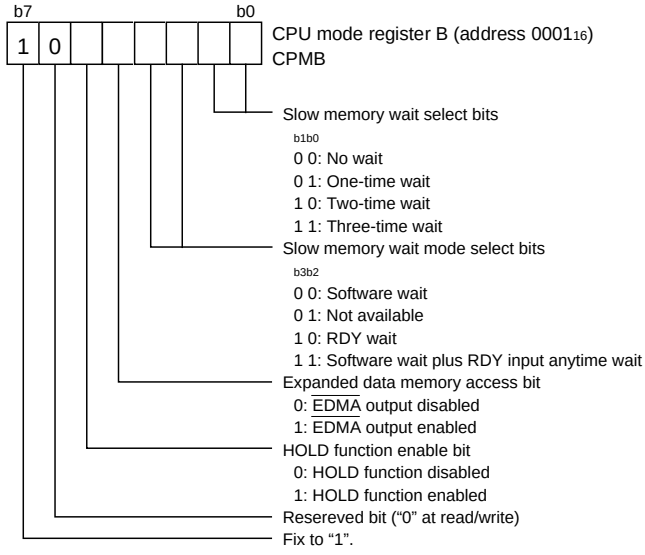


Fig. 71 Structure of CPU mode register B

Slow Memory Wait

The 7641 Group is equipped with the slow memory wait function (Software wait, RDY wait, and Extended RDY wait: software wait plus RDY input anytime wait) for easier interfacing with external devices that have long access times. The slow memory wait function can be enabled in the memory expansion mode and microprocessor mode. The appropriate wait mode is selected by setting bits 0 to 3 of CPU mode register B (address 0001_{16}). This function can extend the read cycle or write cycle only for access to an external memory. However, this wait function cannot be enabled for access to addresses 0008_{16} to $000F_{16}$.

(1) Software wait

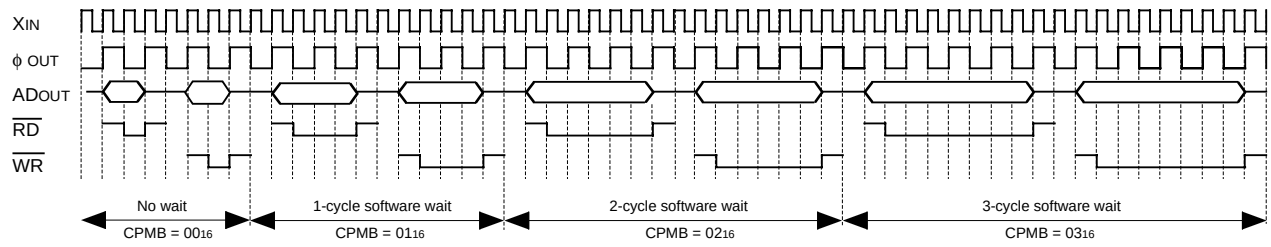
The software wait is selected by setting “00” to the Slow Memory Wait Mode Select Bits of CPU mode register B (address 0001_{16}). Read/write cycles (“L” width of $\overline{RD}$ pin/$\overline{WR}$ pin) can be extended by one to three ϕ cycles. The number of cycles to be extended can be selected with the Slow Memory Wait Select Bits. When the software wait function is selected, the RDY pin status becomes invalid.

(2) RDY wait

RDY Wait is selected by setting “10” to the Slow Memory Wait Mode Select Bits of CPU mode register B (address 0001_{16}). When a fixed time of “L” is input to the RDY pin at the beginning of a read/write cycle (before ϕ cycle falls), the MCU goes to the RDY state. The read/write cycle can then be extended by one to three ϕ cycles. The number of ϕ cycles to be added can be selected by the Slow Memory Wait Bits.

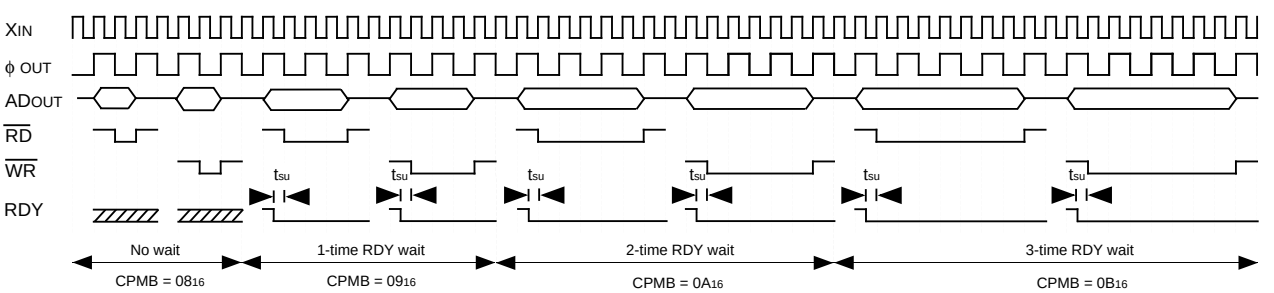
(3) Software wait + Extended RDY wait

Extended RDY Wait is selected by setting “11” to the Slow Memory Wait Mode Select Bits of CPU mode register B (address 0001_{16}). The read/write cycle can be extended when a fixed time of “L” is input to the RDY pin at the beginning of a read/write cycle (before ϕ cycle falls). The RDY pin state is checked continually at each fall of ϕ cycle until the RDY pin goes to “H”. When “H” is input to the RDY pin, the wait is released within 1, 2, or 3 ϕ cycles (as selected with the Slow Memory Wait Bits).



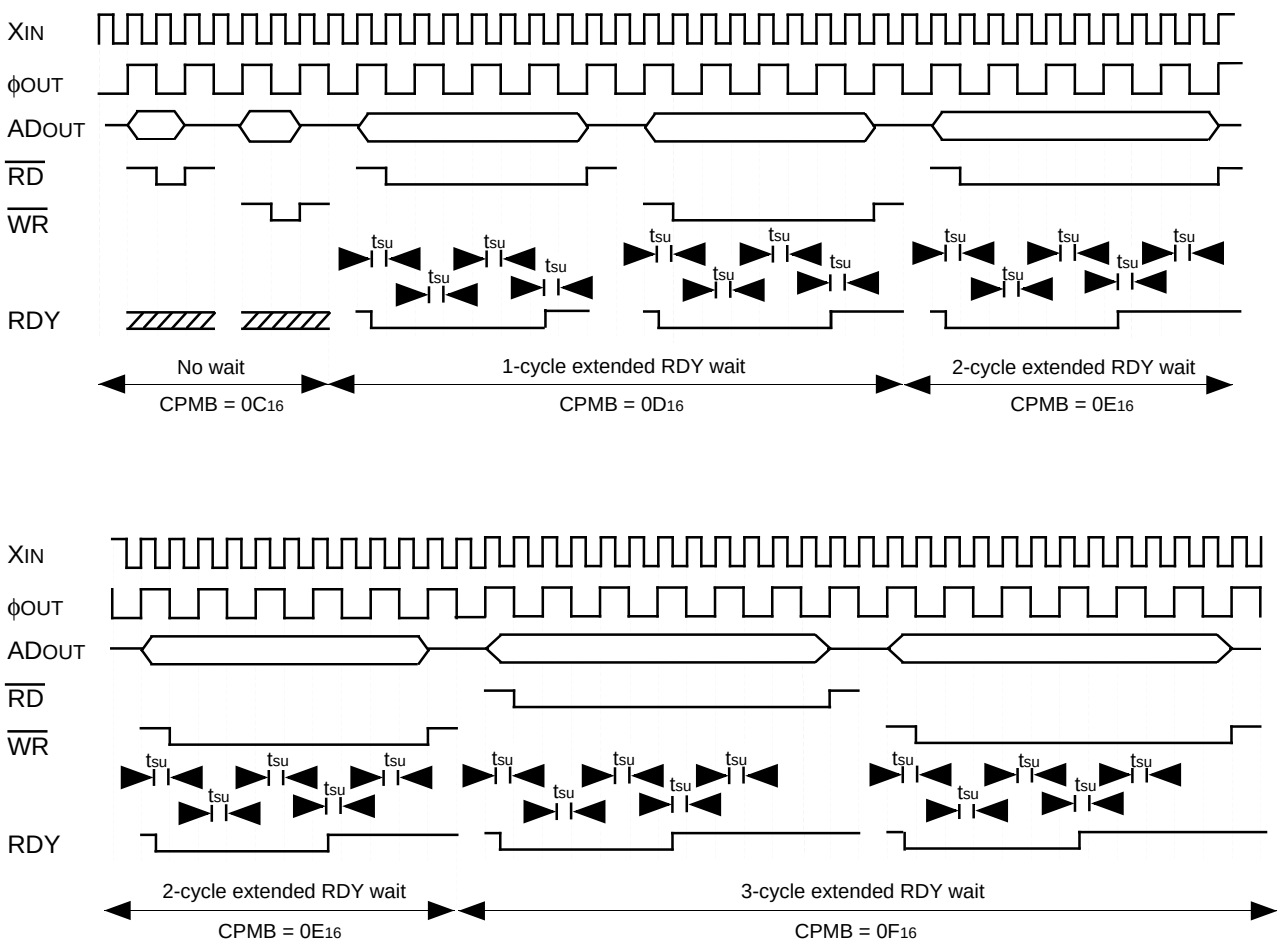
Note: This diagram assumes $\phi = X_{IN}/2$.

Fig. 72 Software wait timing diagram



Note: This diagram assumes $\phi = X_{IN}/2$.

Fig. 73 RDY wait timing diagram



Note: This diagram assumes $\phi = X_{IN}/2$.

Fig. 74 Extended RDY wait (software wait plus RDY input anytime wait) timing diagram

HOLD Function

The HOLD function is used for systems that consist of external circuits that access MCU buses without use of the CPU (Central Processing Unit). The HOLD function is used to generate the timing in which the MCU will relinquish the bus from the CPU to the external circuits. To use the HOLD function, set the HOLD function Enable Bit of CPU mode register B (address 0001_{16}) to "1". This function can be used with both the $\overline{\text{HOLD}}$ pin and the $\overline{\text{HLDA}}$ pin.

The $\overline{\text{HOLD}}$ signal is a signal from an external circuit requesting the MCU to relinquish use of the bus. When "L" level is input, the MCU goes to the HOLD state and remains so while the pin is at "L". The oscillator does not stop oscillating during the HOLD state, therefore allowing the internal peripheral functions to operate during this time.

When the MCU relinquishes use of the bus, "L" level is output from the $\overline{\text{HLDA}}$ pin. The MCU makes ports P0 and P1 (address buses) and port P2 (data bus) tri-state outputs and holds port $P3_7$ ($\overline{\text{RD}}$ pin) and port $P3_6$ ($\overline{\text{WR}}$ pin) "H" level. Port $P3_4$ (ϕ_{OUT} pin) continues to oscillate. This function is not valid when the MCU is using the $\overline{\text{IBF}}_1$ function with the $\overline{\text{HLDA}}$ pin.

Expanded Data Memory Access

In Expanded Data Memory Access Mode, the MCU can access a data area larger than 64 Kbytes with the LDA ($zz), Y (indirect Y) instruction and the STA ($zz), Y (indirect Y) instruction.

To use this mode, set the Expanded Data Memory Access Bit of CPU mode register B (address 0001_{16}) to "1". In this case, port $P4_0$ ($\overline{\text{EDMA}}$ pin) goes "L" level during the read/write cycle of the LDA or STA instruction.

The determination of which bank to access is done by using an I/O port to represent expanded addresses exceeding address bus AB_{15}. For example, when accessing 4 banks, use two I/O ports to represent address buses AB_{16} and AB_{17}.

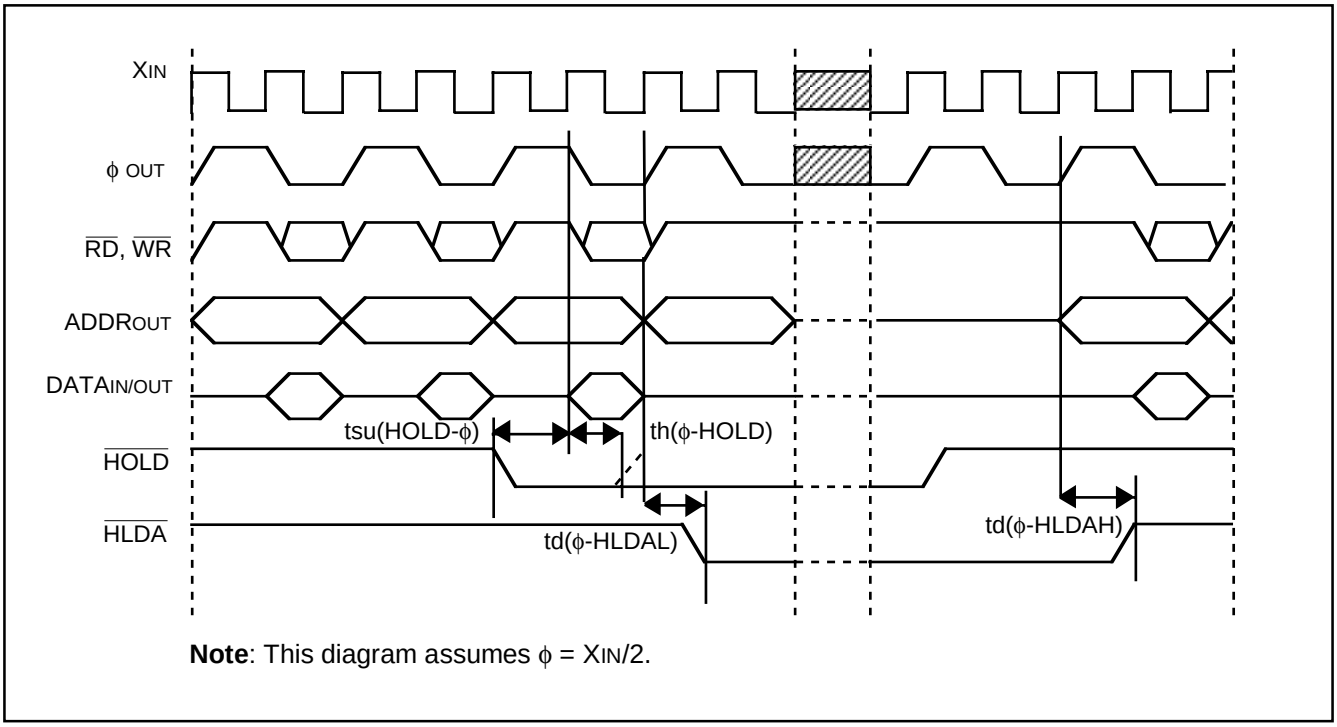


Fig. 75 Hold function timing diagram

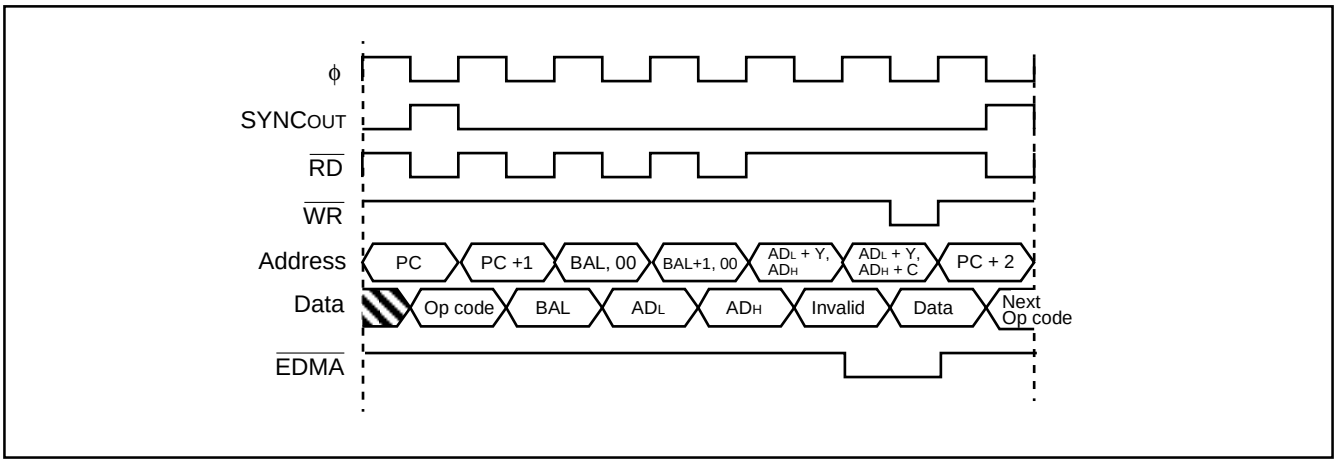


Fig. 76 STA ($ zz), Y instruction sequence when $\overline{\text{EDMA}}$ enabled

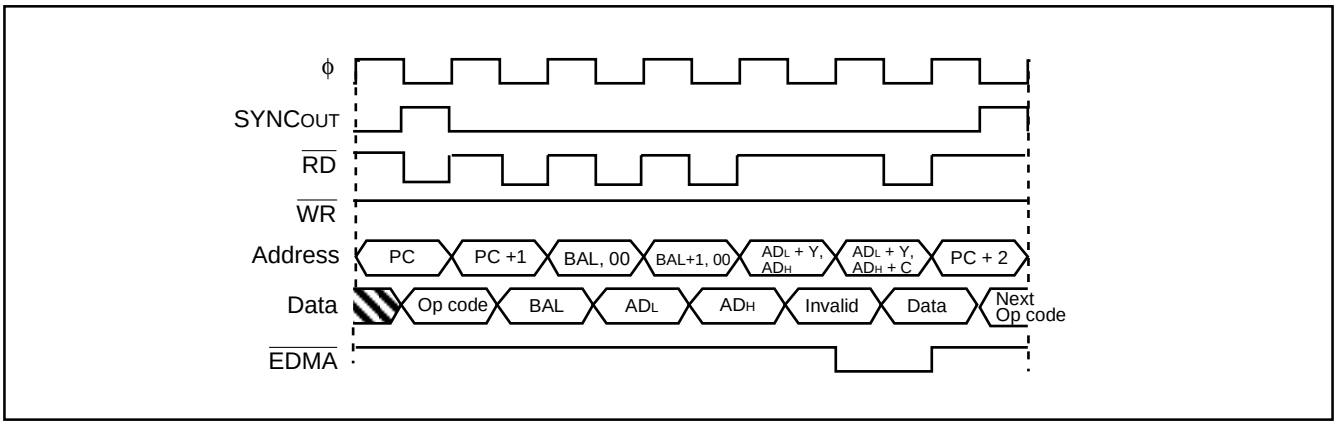


Fig. 77 LDA ($ zz), Y instruction sequence when $\overline{\text{EDMA}}$ enabled and T flag = "0"

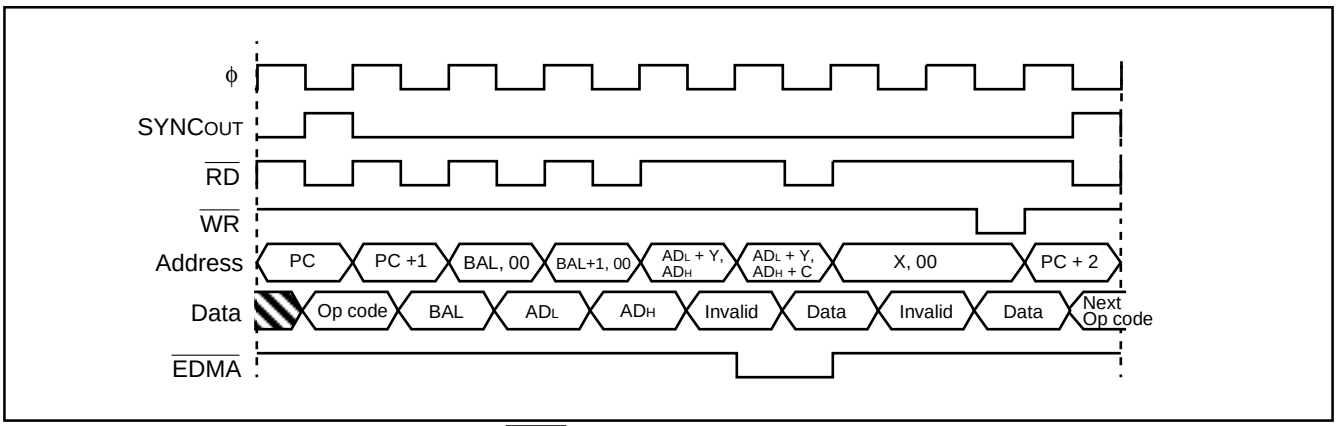


Fig. 78 LDA ($ zz), Y instruction sequence when $\overline{\text{EDMA}}$ enabled and T flag = "1"

ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings

Table 10 Absolute maximum ratings

Symbol	Parameter		Conditions	Ratings	Unit
V_{CC}	Power source voltage		All voltages are based on Vss. Output transistors are cut off.	–0.3 to 6.5	V
AVCC	Analog power source voltage			–0.3 to V_{CC}+0.3	V
V_I	Input voltage	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$		–0.3 to V_{CC}+0.3	V
V_I	Input voltage	$\overline{RESET}$, X_{IN}, X_{CIN}		–0.3 to V_{CC}+0.3	V
V_I	Input voltage	CNV_{SS} Mask ROM version		–0.3 to Vcc + 0.3	V
		CNV_{SS} Flash memory version		–0.3 to 6.5	V
V_I	Input voltage	USB D+, USB D–		–0.5 to 3.8	V
V_O	Output voltage	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$, X_{OUT}, X_{COUT}, LPF		–0.3 to V_{CC}+0.3	V
V_O	Output voltage	USB D+, USB D–		–0.5 to 3.8	V
P_d	Power dissipation (**Note**)		Ta = 25°C	750	mW
T_{opr}	Operating temperature			–20 to 70	°C
T_{stg}	Storage temperature			–40 to 125	°C

Note: The maximum power dissipation depends on the MCU's power dissipation and the specific heat consumption of the package.

Recommended Operating Conditions
In Vcc = 5 V

Table 11 Recommended operating conditions (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
V_{CC}	Power source voltage		4.15	5.0	5.25	V
AVcc	Analog reference voltage		4.15	5.0	V_{CC}	V
V_{SS}	Power source voltage			0		V
AVss	Analog reference voltage			0		V
V_{IH}	"H" input voltage	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	0.8V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage (Selecting VIHL level input)	$P2_0$–$P2_7$	0.5V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage (Selecting TTL level input for MBI input)	$P5_4$–$P5_7$, $P6_0$–$P6_7$, $P7_2$	2.0		V_{CC}	V
V_{IH}	"H" input voltage	$\overline{RESET}$, X_{IN}, X_{CIN}, CNVss	0.8V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage	USB D+, USB D–	2.0		3.8	V
V_{IL}	"L" input voltage	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	0		0.2V_{CC}	V
V_{IL}	"L" input voltage (Selecting VIHL level input)	$P2_0$–$P2_7$	0		0.16V_{CC}	V
V_{IL}	"L" input voltage (Selecting TTL level input for MBI input)	$P5_4$–$P5_7$, $P6_0$–$P6_7$, $P7_2$	0		0.8	V
V_{IL}	"L" input voltage	$\overline{RESET}$, X_{IN}, X_{CIN}, CNVss	0		0.2V_{CC}	V
V_{IL}	"L" input voltage	USB D+, USB D–			0.8	V
$\Sigma I_{OH(peak)}$	"H" total peak output current (**Note 1**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–80	mA
$\Sigma I_{OL(peak)}$	"L" total peak output current (**Note 1**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			80	mA
$\Sigma I_{OH(avg)}$	"H" total average output current (**Note 1**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–40	mA
$\Sigma I_{OL(avg)}$	"L" total average output current (**Note 1**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			40	mA
$I_{OH(peak)}$	"H" peak output current (**Note 2**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–10	mA
$I_{OL(peak)}$	"L" peak output current (**Note 2**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			10	mA
$I_{OH(avg)}$	"H" average output current (**Note 3**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–5.0	mA
$I_{OL(avg)}$	"L" average output current (**Note 3**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			5.0	mA
f($CNTR_0$)	Timer X input frequency (**Note 4**)				5.0	MHz
f($CNTR_1$)	Timer Y input frequency (**Note 4**)				5.0	MHz
f(X_{IN})	Main clock input frequency (**Notes 4, 5**)		1		24	MHz
f(X_{CIN})	Sub-clock input frequency (**Notes 4, 6**)			32.768	50/5.0	kHz/MHz

Notes 1: The total peak output current is the peak value of the peak currents flowing through all the applicable ports. The total average output current is the average value measured over 100 ms flowing through all the applicable ports.

2: The peak output current is the peak current flowing in each port.

3: The average output current is an average value measured over 100 ms.

4: The duty of oscillation frequency is 50 %.

5: Connect a ceramic resonator or a quartz-crystal oscillator between the XIN and XOUT pins. Its maximum oscillation frequency must be 24 MHz. However, make sure to set ϕ to 12 MHz or slower. More faster clocks are required as the f(XIN) when using the frequency synthesizer as possible.

6: Connect a ceramic resonator or a quartz-crystal oscillator between the XCIN and XCOUT pins. Its maximum oscillation frequency must be 50 kHz. Input an external clock having 5 MHz frequency (max.) from the XCIN pin.

Electrical Characteristics
In Vcc = 5 V

Table 12 Electrical characteristics (1) (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
V_{OH}	"H" output voltage $P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	I_{OH} = –10 mA	V_{CC}–2.0			V
V_{OH}	"H" output voltage USB D+, USB D-	USB+, and USB- pins pull-down via a resistor of 15 kΩ ± 5 % USB+ pin pull-up to Ext. Cap. pin via a resistor of 1.5 kΩ ± 5 %	2.8		3.6	V
V_{OL}	"L" output voltage $P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	I_{OL} = 10 mA			2.0	V
V_{OL}	"L" output voltage USB D+, USB D-	USB+, and USB- pins pull-down via a resistor of 15 kΩ ± 5 % USB+ pin pull-up to Ext. Cap. pin via a resistor of 1.5 kΩ ± 5 %			0.3	V
V_{T+}–V_{T-}	Hysteresis $CNTR_0$, $CNTR_1$, INT_0, INT_1, RDY, $\overline{HOLD}$, $P2_0$–$P2_7$			0.5		V
V_{T+}–V_{T-}	Hysteresis $\overline{URXD1}$, $\overline{URXD2}$ (SCLK), $\overline{CTS_2}$ (SRXD), $\overline{SRDY}$, $\overline{CTS_1}$			0.5		V
V_{T+}–V_{T-}	Hysteresis $\overline{RESET}$			0.5		V
I_{IH}	"H" input current $P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	V_I = V_{CC}			5.0	µA
I_{IH}	"H" input current $\overline{RESET}$, CNV_{SS}				5.0	µA
I_{IH}	"H" input current X_{IN}			9.0	20	µA
I_{IH}	"H" input current X_{CIN}				5.0	µA
I_{IL}	"L" input current $P0_0$–$P0_7$, $P1_0$–$P1_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	V_I = V_{SS}			–5.0	µA
I_{IL}	"L" input current $\overline{RESET}$				–5.0	µA
I_{IL}	"L" input current CNV_{SS}				–20	µA
I_{IL}	"L" input current X_{IN}			–9.0	–20	µA
I_{IL}	"L" input current X_{CIN}				–5.0	µA
I_{IL}	"L" input current $P2_0$–$P2_7$	V_I = V_{SS} Pull-ups "off"			–5.0	µA
		V_{CC} = 5.0 V, V_I = V_{SS} Pull-ups "on"	–30	–65	–140	µA
V_{RAM}		When clock is stopped	2.0		5.25	V

In Vcc = 5 V

Table 13 Electrical characteristics (2) (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
ICC	Power source current (Output transistor is isolated.)	Normal mode (**Note 1**) f(XIN) = 24 MHz, ϕ = 12 MHz USB operating Frequency synthesizer ON		40	90	mA
		Wait mode (**Note 2**) f(XIN) = 24 MHz, ϕ = 12 MHz USB enabled, USB clock stopped Frequency synthesizer ON		5.0	11	mA
		Wait mode (**Note 3**) f(XCIN) = 32 kHz, ϕ = 16 kHz USB disabled Frequency synthesizer OFF USB transceiver DC-DC converter OFF			10	μA
		Stop mode USB transceiver DC-DC converter ON Low current mode (USBC3 = “1”)		100	250	μA
		Stop mode USB transceiver DC-DC converter OFF Ta = 25 °C			1.0	μA
		Stop mode USB transceiver DC-DC converter OFF Ta = 70 °C			10	μA

<Test conditions>

Notes 1: Operating in single-chip mode
Clock input from XIN pin (XOUT oscillator stopped)
USB operating with USB transceiver DC-DC converter enabled
Operating functions: Frequency synthesizer, CPU, two UARTs, DMAC, Timers and Count source generator
Disabled functions: Master CPU bus interface and Serial I/O

2: Operating in single-chip mode with Wait mode
Clock input from XIN pin (XOUT oscillator stopped)
USB suspended due to USB clock stopped with USB transceiver DC-DC converter enabled
Operating functions: Frequency synthesizer, Timers and Count source generator
Disabled functions: CPU, two UARTs, DMAC, Master CPU bus interface and Serial I/O

3: Operating in single-chip mode with Wait mode
XIN - XOUT oscillator stopped
Clock input from XCIN pin (XCOUT oscillator stopped)
USB stopped, USB clock stopped and USB transceiver DC-DC converter disabled
Operating functions: Timers and Count source generator
Disabled functions: Frequency synthesizer, CPU, two UARTs, DMAC, Master CPU bus interface and Serial I/O

Timing Requirements
In Vcc = 5 V

Table 14 Timing requirements (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_{W(\overline{RESET})}$	Reset input “L” pulse width	2			µs
$t_{C(XIN)}$	Main clock input cycle time (**Note**)	41.66			ns
$t_{WH(XIN)}$	Main clock input “H” pulse width	0.4•tc(XIN)			ns
$t_{WL(XIN)}$	Main clock input “L” pulse width	0.4•tc(XIN)			ns
$t_{C(XCIN)}$	Sub-clock input cycle time	200			ns
$t_{WH(XCIN)}$	Sub-clock input “H” pulse width	0.4•tc(XCIN)			ns
$t_{WL(XCIN)}$	Sub-clock input “L” pulse width	0.4•tc(XCIN)			ns
$t_{C(INT)}$	INT_0, INT_1 input cycle time	200			ns
$t_{WH(INT)}$	INT_0, INT_1 input “H” pulse width	90			ns
$t_{WL(INT)}$	INT_0, INT_1 input “L” pulse width	90			ns
$t_{C(CNTRI)}$	$CNTR_0$, $CNTR_1$ input cycle time	200			ns
$t_{WH(CNTRI)}$	$CNTR_0$, $CNTR_1$ input “H” pulse width	80			ns
$t_{WL(CNTRI)}$	$CNTR_0$, $CNTR_1$ input “L” pulse width	80			ns
$t_{d(\phi\text{-TOUT})}$	Timer TOUT delay time			15	ns
$t_{d(\phi\text{-CNTR0})}$	Timer $CNTR_0$ delay time (Pulse output mode)			15	ns
$t_{C(CNTRE0)}$	Timer $CNTR_0$ input cycle time (Event counter mode)	200			ns
$t_{WH(CNTRE0)}$	Timer $CNTR_0$ input “H” pulse width (Event counter mode)	0.4•tc(CNTRE0)			ns
$t_{WL(CNTRE0)}$	Timer $CNTR_0$ input “L” pulse width (Event counter mode)	0.4•tc(CNTRE0)			ns
$t_{d(\phi\text{-CNTR1})}$	Timer $CNTR_1$ delay time (Pulse output mode)			15	ns
$t_{C(CNTRE1)}$	Timer $CNTR_1$ input cycle time (Event counter mode)	200			ns
$t_{WH(CNTRE1)}$	Timer $CNTR_1$ input “H” pulse width (Event counter mode)	0.4•tc(CNTRE1)			ns
$t_{WL(CNTRE1)}$	Timer $CNTR_1$ input “L” pulse width (Event counter mode)	0.4•tc(CNTRE1)			ns
$t_{C(SCLKE)}$	Serial I/O external clock input cycle time	400			ns
$t_{WH(SCLKE)}$	Serial I/O external clock input “H” pulse width	190			ns
$t_{WL(SCLKE)}$	Serial I/O external clock input “L” pulse width	180			ns
$t_{su(SRXD\text{-}SCLKE)}$	Serial I/O input setup time (external clock)	15			ns
$t_{h(SCLKE\text{-}SRXD)}$	Serial I/O input hold time (external clock)	10			ns
$t_{d(SCLKE\text{-}STXD)}$	Serial I/O output delay time (external clock)			25	ns
$t_{v(SCLKE\text{-}SRDY)}$	Serial I/O $\overline{SRDY}$ valid time (external clock)			26	ns
$t_{C(SCLKI)}$	Serial I/O internal clock output cycle time	166.66			ns
$t_{WH(SCLKI)}$	Serial I/O internal clock output “H” pulse width	0.5•tc(SCLKI) – 5			ns
$t_{WL(SCLKI)}$	Serial I/O internal clock output “L” pulse width	0.5•tc(SCLKI) – 5			ns
$t_{su(SRXD\text{-}SCLKI)}$	Serial I/O input setup time (internal clock)	20			ns
$t_{h(SCLKI\text{-}SRXD)}$	Serial I/O input hold time (internal clock)	5			ns
$t_{d(SCLKI\text{-}STXD)}$	Serial I/O output delay time (internal clock)			5	ns

Note: Make sure not to exceed 12 MHz of ϕ, in other words, $tc(\phi) \geq 83.33$ ns). For example, set bit 7 of the clock control register (CCR) to “0” in the case of $tc(XIN) < 41.66$ ns.

In Vcc = 5 V

Table 15 Master CPU bus interface (MBI; RD, WR separate type) (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{su}(S-R)	$\overline{S0}$, $\overline{S1}$ setup time for read	0			ns
t_{su}(S-W)	$\overline{S0}$, $\overline{S1}$ setup time for write	0			ns
t_h(R-S)	$\overline{S0}$, $\overline{S1}$ hold time for read	0			ns
t_h(W-S)	$\overline{S0}$, $\overline{S1}$ hold time for write	0			ns
t_{su}(A-R)	A0 setup time for read	10			ns
t_{su}(A-W)	A0 setup time for write	10			ns
t_h(R-A)	A0 hold time for read	0			ns
t_h(W-A)	A0 hold time for write	0			ns
t_w(R)	Read pulse width	50			ns
t_w(W)	Write pulse width	50			ns
t_{su}(D-W)	Data input setup time before write	25			ns
t_h(W-D)	Data input hold time after write	0			ns
t_a(R-D)	Data output enable time after read			40	ns
t_v(R-D)	Data output disable time after read	10			ns
t_v(R-OBF)	OBF output transmission time after read			40	ns
t_d(W-IBF)	$\overline{IBF}$ output transmission time after write			40	ns

In Vcc = 5 V

Table 16 Master CPU bus interface (MBI; R/$\overline{W}$ type) (Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{su}(S-E)	$\overline{S0}$, $\overline{S1}$ setup time	0			ns
t_h(E-S)	$\overline{S0}$, $\overline{S1}$ hold time	0			ns
t_{su}(A-E)	A0 setup time	10			ns
t_h(E-A)	A0 hold time	0			ns
t_{su}(RW-E)	R/$\overline{W}$ setup time	10			ns
t_h(E-RW)	R/$\overline{W}$ hold time	10			ns
t_w(E)	Enable pulse width	50			ns
t_w(E-E)	Enable pulse interval	50			ns
t_{su}(D-E)	Data input setup time before write	25			ns
t_h(E-D)	Data input hold time after write	0			ns
t_a(E-D)	Data output enable time after read			40	ns
t_v(E-D)	Data output disable time after read	10			ns
t_v(E-OBF)	OBF output transmission time after E inactive			40	ns
t_d(E-IBF)	$\overline{IBF}$ output transmission time after E inactive			40	ns

In Vcc = 5 V

Table 17 Timing requirements and switching characteristics in memory expansion and microprocessor modes

(Vcc = 4.15 to 5.25 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_C(\phi)$	ϕ clock cycle time	83.33			ns
$t_{WH}(\phi)$	ϕ clock "H" pulse width	0.5•tc(ϕ) – 5			ns
$t_{WL}(\phi)$	ϕ clock "L" pulse width	0.5•tc(ϕ) – 5			ns
$t_d(\phi$ -AH)	AB_{15}–AB_8 delay time			31	ns
$t_v(\phi$ -AH)	AB_{15}–AB_8 valid time	5			ns
$t_d(\phi$ -AL)	AB_7–AB_0 delay time			33	ns
$t_v(\phi$ -AL)	AB_7–AB_0 valid time	5			ns
$t_d(\phi$ -WR)	$\overline{WR}$ delay time			6	ns
$t_v(\phi$ -WR)	$\overline{WR}$ valid time	3			ns
$t_d(\phi$ -RD)	$\overline{RD}$ delay time			6	ns
$t_v(\phi$ -RD)	$\overline{RD}$ valid time	3			ns
$t_d(\phi$ -SYNC)	$SYNC_{OUT}$ delay time			6	ns
$t_v(\phi$ -SYNC)	$SYNC_{OUT}$ valid time	4			ns
$t_d(\phi$ -DMA)	DMA_{OUT} delay time			25	ns
$t_v(\phi$ -DMA)	DMA_{OUT} valid time	5			ns
t_{su}(RDY- ϕ)	RDY setup time	21			ns
$t_h(\phi$ -RDY)	RDY hold time	0			ns
t_{su}(HOLD- ϕ)	$\overline{HOLD}$ setup time	21			ns
$t_h(\phi$ -HOLD)	$\overline{HOLD}$ hold time	0			ns
$t_d(\phi$ -HLDAL)	$\overline{HOLD}$ "L" delay time			25	ns
$t_d(\phi$ -HLDAH)	$\overline{HOLD}$ "H" delay time			25	ns
t_{su}(DB- ϕ)	Data bus setup time	7			ns
$t_h(\phi$ -DB)	Data bus hold time	0			ns
$t_d(\phi$ -DB)	Data bus delay time			22	ns
$t_v(\phi$ -DB)	Data bus valid time (**Note 1**)	13			ns
$t_d(\phi$ -EDMA)	$\overline{EDMA}$ delay time			9	ns
$t_v(\phi$ -EDMA)	$\overline{EDMA}$ valid time	4			ns
t_{WL}(WR) (**Note 2**)	$\overline{WR}$ pulse width	0.5•tc(ϕ) – 5			ns
t_{WL}(RD) (**Note 2**)	$\overline{RD}$ pulse width	0.5•tc(ϕ) – 5			ns
t_d(AH-WR)	AB_{15}–AB_8 valid time before $\overline{WR}$	0.5•tc(ϕ) – 28			ns
t_d(AL-WR)	AB_7–AB_0 valid time before $\overline{WR}$	0.5•tc(ϕ) – 30			ns
t_v(WR-AH)	AB_{15}–AB_8 valid time after $\overline{WR}$	0			ns
t_v(WR-AL)	AB_7–AB_0 valid time after $\overline{WR}$	0			ns
t_d(AH-RD)	AB_{15}–AB_8 valid time before $\overline{RD}$	0.5•tc(ϕ) – 28			ns
t_d(AL-RD)	AB_7–AB_0 valid time before $\overline{RD}$	0.5•tc(ϕ) – 30			ns
t_v(RD-AH)	AB_{15}–AB_8 valid time after $\overline{RD}$	0			ns
t_v(RD-AL)	AB_7–AB_0 valid time after $\overline{RD}$	0			ns
t_{su}(RDY-WR)	RDY setup time before $\overline{WR}$	27			ns
t_h(WR-RDY)	RDY hold time after $\overline{WR}$	0			ns
t_{su}(RDY-RD)	RDY setup time before $\overline{RD}$	27			ns
t_h(RD-RDY)	RDY hold time after $\overline{RD}$	0			ns
t_{su}(DB-RD)	Data bus setup time before $\overline{RD}$	13			ns
t_h(RD-DB)	Data bus hold time after $\overline{RD}$	0			ns
t_d(WR-DB)	Data bus delay time before $\overline{WR}$			20	ns
t_v(WR-DB)	Data bus valid time after $\overline{WR}$ (**Note 1**)	10			ns
t_v(WR-EDMA)	$\overline{EDMA}$ delay time after $\overline{WR}$	2			ns
t_v(RD-EDMA)	$\overline{EDMA}$ valid time after $\overline{RD}$	2			ns
t_r(D+), t_r(D-)	USB output rise time, C_L = 50 pF	4		20	ns
t_f(D+), t_f(D-)	USB output fall time, C_L = 50 pF	4		20	ns

Notes 1: Test conditions: I_{OHL} = ± 5mA, C_L = 50 pF
2: t_{WL}(RD) = ((n + 0.5) • tc(PHI)) – 5 ns (n = wait number)
t_{WL}(WR) = ((n + 0.5) • tc(PHI)) – 5 ns (n = wait number)
For example, two software waits, PHI = 12 MHz operating
t_{WL}(RD) = 2.5 • tc(PHI) – 5 ns = 203.33 ns

Recommended Operating Conditions
In Vcc = 3 V

Table 18 Recommended operating conditions (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
V_{CC}	Power source voltage		3.0	3.3	3.6	V
AVcc	Analog reference voltage		3.0	3.3	V_{CC}	V
V_{SS}	Power source voltage			0		V
AVss	Analog reference voltage			0		V
V_{IH}	"H" input voltage	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	0.8V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage (Selecting VIHL level input)	$P2_0$–$P2_7$	0.5V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage	$\overline{RESET}$, X_{IN}, X_{CIN}, CNVss	0.8V_{CC}		V_{CC}	V
V_{IH}	"H" input voltage	USB D+, USB D–	2.0			V
V_{IL}	"L" input voltage	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	0		0.2V_{CC}	V
V_{IL}	"L" input voltage (Selecting VIHL level input)	$P2_0$–$P2_7$	0		0.16V_{CC}	V
V_{IL}	"L" input voltage	$\overline{RESET}$, X_{IN}, X_{CIN}, CNVss	0		0.2V_{CC}	V
V_{IL}	"L" input voltage	USB D+, USB D–			0.8	V
$\Sigma I_{OH(peak)}$	"H" total peak output current (**Note 1**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–80	mA
$\Sigma I_{OL(peak)}$	"L" total peak output current (**Note 1**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			80	mA
$\Sigma I_{OH(avg)}$	"H" total average output current (**Note 1**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–40	mA
$\Sigma I_{OL(avg)}$	"L" total average output current (**Note 1**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			40	mA
$I_{OH(peak)}$	"H" peak output current (**Note 2**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–10	mA
$I_{OL(peak)}$	"L" peak output current (**Note 2**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			10	mA
$I_{OH(avg)}$	"H" average output current (**Note 3**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			–5.0	mA
$I_{OL(avg)}$	"L" average output current (**Note 3**)	$P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$			5.0	mA
f($CNTR_0$)	Timer X input frequency (**Note 4**)				5.0	MHz
f($CNTR_1$)	Timer Y input frequency (**Note 4**)				5.0	MHz
f(X_{IN})	Main clock input frequency (**Notes 4, 5**)		1		24	MHz
f(X_{CIN})	Sub-clock input frequency (**Notes 4, 6**)			32.768	50/5.0	kHz/MHz

Notes 1: The total peak output current is the peak value of the peak currents flowing through all the applicable ports. The total average output current is the average value measured over 100 ms flowing through all the applicable ports.

2: The peak output current is the peak current flowing in each port.

3: The average output current is an average value measured over 100 ms.

4: The duty of oscillation frequency is 50 %.

5: Connect a ceramic resonator or a quartz-crystal oscillator between the XIN and XOUT pins. Its maximum oscillation frequency must be 24 MHz. However, make sure to set ϕ to 6 MHz or slower. More faster clocks are required as the f(XIN) when using the frequency synthesizer as possible.

6: Connect a ceramic resonator or a quartz-crystal oscillator between the XCIN and XCOUT pins. Its maximum oscillation frequency must be 50 kHz. Input an external clock having 5 MHz (max.) frequency from the XCIN pin.

Electrical Characteristics
In Vcc = 3 V

Table 19 Electrical characteristics (1) (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
V_{OH}	"H" output voltage $P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	I_{OH} = –1 mA	V_{CC}–1.0			V
V_{OH}	"H" output voltage USB D+, USB D-	USB+, and USB- pins pull-down via a resistor of 15 kΩ ± 5 % USB+ pin pull-up to Ext. Cap. pin via a resistor of 1.5 kΩ ± 5 %	2.8		3.6	V
V_{OL}	"L" output voltage $P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	I_{OL} = 1 mA			1.0	V
V_{OL}	"L" output voltage USB D+, USB D-	USB+, and USB- pins pull-down via a resistor of 15 kΩ ± 5 % USB+ pin pull-up to Ext. Cap. pin via a resistor of 1.5 kΩ ± 5 %	0		0.3	V
V_{T+}–V_{T-}	Hysteresis $CNTR_0$, $CNTR_1$, INT_0, INT_1, RDY, $\overline{HOLD}$, $P2_0$–$P2_7$			0.3		V
V_{T+}–V_{T-}	Hysteresis URXD1, URXD2 (SCLK), $\overline{CTS_2}$ (SRXD), $\overline{SRDY}$, $\overline{CTS_1}$			0.3		V
V_{T+}–V_{T-}	Hysteresis $\overline{RESET}$			0.3		V
I_{IH}	"H" input current $P0_0$–$P0_7$, $P1_0$–$P1_7$, $P2_0$–$P2_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	V_I = V_{CC}			5.0	µA
I_{IH}	"H" input current $\overline{RESET}$, CNV_{SS}				5.0	µA
I_{IH}	"H" input current X_{IN}			9.0	20	µA
I_{IH}	"H" input current X_{CIN}				5.0	µA
I_{IL}	"L" input current $P0_0$–$P0_7$, $P1_0$–$P1_7$, $P3_0$–$P3_7$, $P4_0$–$P4_4$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_4$, $P8_0$–$P8_7$	V_I = V_{SS}			–5.0	µA
I_{IL}	"L" input current $\overline{RESET}$				–5.0	µA
I_{IL}	"L" input current CNV_{SS}				–20	µA
I_{IL}	"L" input current X_{IN}			–9.0	–20	µA
I_{IL}	"L" input current X_{CIN}				–5.0	µA
I_{IL}	"L" input current $P2_0$–$P2_7$	V_I = V_{SS} Pull-ups "off"			–5.0	µA
		V_{CC} = 3.0 V, V_I = V_{SS} Pull-ups "on"	–10	–20	–50	µA
V_{RAM}		When clock is stopped	2.0			V

In Vcc = 3 V

Table 20 Electrical characteristics (2) (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
ICC	Power source current (Output transistor is isolated.)	Normal mode (**Note 1**) f(XIN) = 24 MHz, ϕ = 6 MHz USB operating Frequency synthesizer ON		25	45	mA
		Wait mode (**Note 2**) f(XIN) = 24 MHz, ϕ = 6 MHz USB enabled, USB clock stopped Frequency synthesizer ON		2.5	6	mA
		Wait mode (**Note 3**) f(XCIN) = 32 kHz, ϕ = 16 kHz USB disabled Frequency synthesizer OFF USB transceiver DC-DC converter OFF			6	µA
		Stop mode USB transceiver DC-DC converter ON Low current mode (USBC3 = “1”)		TBD	TBD	µA
		Stop mode USB transceiver DC-DC converter OFF Ta = 25 °C			1.0	µA
		Stop mode USB transceiver DC-DC converter OFF Ta = 70 °C			10	µA

<Test conditions>

Notes 1: Operating in single-chip mode
Clock input from XIN pin (XOUT oscillator stopped)
USB operating with USB transceiver DC-DC converter enabled
Operating functions: Frequency synthesizer, CPU, two UARTs, DMAC, Timers and Count source generator
Disabled functions: Master CPU bus interface and Serial I/O

2: Operating in single-chip mode with Wait mode
Clock input from XIN pin (XOUT oscillator stopped)
USB suspended due to USB clock stopped with USB transceiver DC-DC converter enabled
Operating functions: Frequency synthesizer, Timers and Count source generator
Disabled functions: CPU, two UARTs, DMAC, Master CPU bus interface and Serial I/O

3: Operating in single-chip mode with Wait mode
XIN - XOUT oscillator stopped
Clock input from XCIN pin (XCOUT oscillator stopped)
USB stopped, USB clock stopped and USB transceiver DC-DC converter disabled
Operating functions: Timers and Count source generator
Disabled functions: Frequency synthesizer, CPU, two UARTs, DMAC, Master CPU bus interface and Serial I/O

Timing Requirements
In Vcc = 3 V

Table 21 Timing requirements (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_{W(\overline{RESET})}$	Reset input “L” pulse width	2			µs
$t_{C(XIN)}$	Main clock input cycle time (**Note**)	41.66			ns
$t_{WH(XIN)}$	Main clock input “H” pulse width	0.4•tc(XIN)			ns
$t_{WL(XIN)}$	Main clock input “L” pulse width	0.4•tc(XIN)			ns
$t_{C(XCIN)}$	Sub-clock input cycle time	200			ns
$t_{WH(XCIN)}$	Sub-clock input “H” pulse width	0.4•tc(XCIN)			ns
$t_{WL(XCIN)}$	Sub-clock input “L” pulse width	0.4•tc(XCIN)			ns
$t_{C(INT)}$	INT_0, INT_1 input cycle time	250			ns
$t_{WH(INT)}$	INT_0, INT_1 input “H” pulse width	110			ns
$t_{WL(INT)}$	INT_0, INT_1 input “L” pulse width	110			ns
$t_{C(CNTRI)}$	$CNTR_0$, $CNTR_1$ input cycle time	250			ns
$t_{WH(CNTRI)}$	$CNTR_0$, $CNTR_1$ input “H” pulse width	110			ns
$t_{WL(CNTRI)}$	$CNTR_0$, $CNTR_1$ input “L” pulse width	110			ns
$t_{d(\phi\text{ -TOUT})}$	Timer TOUT delay time			17	ns
$t_{d(\phi\text{ -CNTR0})}$	Timer $CNTR_0$ delay time (Pulse output mode)			16	ns
$t_{C(CNTRE0)}$	Timer $CNTR_0$ input cycle time (Event counter mode)	250			ns
$t_{WH(CNTRE0)}$	Timer $CNTR_0$ input “H” pulse width (Event counter mode)	0.4•tc(CNTRE0)			ns
$t_{WL(CNTRE0)}$	Timer $CNTR_0$ input “L” pulse width (Event counter mode)	0.4•tc(CNTRE0)			ns
$t_{d(\phi\text{ -CNTR1})}$	Timer $CNTR_1$ delay time (Pulse output mode)			15	ns
$t_{C(CNTRE1)}$	Timer $CNTR_1$ input cycle time (Event counter mode)	250			ns
$t_{WH(CNTRE1)}$	Timer $CNTR_1$ input “H” pulse width (Event counter mode)	0.4•tc(CNTRE1)			ns
$t_{WL(CNTRE1)}$	Timer $CNTR_1$ input “L” pulse width (Event counter mode)	0.4•tc(CNTRE1)			ns
$t_{C(SCLKE)}$	Serial I/O external clock input cycle time	450			ns
$t_{WH(SCLKE)}$	Serial I/O external clock input “H” pulse width	220			ns
$t_{WL(SCLKE)}$	Serial I/O external clock input “L” pulse width	190			ns
$t_{su(SRXD-SCLKE)}$	Serial I/O input setup time (external clock)	20			ns
$t_{h(SCLKE-SRXD)}$	Serial I/O input hold time (external clock)	15			ns
$t_{d(SCLKE-STXD)}$	Serial I/O output delay time (external clock)			34	ns
$t_{v(SCLKE-SRDY)}$	Serial I/O $\overline{SRDY}$ valid time (external clock)			35	ns
$t_{C(SCLKI)}$	Serial I/O internal clock output cycle time	300			ns
$t_{WH(SCLKI)}$	Serial I/O internal clock output “H” pulse width	0.5•tc(SCLKI) – 5			ns
$t_{WL(SCLKI)}$	Serial I/O internal clock output “L” pulse width	0.5•tc(SCLKI) – 5			ns
$t_{su(SRXD-SCLKI)}$	Serial I/O input setup time (internal clock)	20			ns
$t_{h(SCLKI-SRXD)}$	Serial I/O input hold time (internal clock)	5			ns
$t_{d(SCLKI-STXD)}$	Serial I/O output delay time (internal clock)			5	ns

Note: Make sure not to exceed 6 MHz of ϕ, in other words, $tc(\phi) \geq 166.66$ ns).

In Vcc = 3 V

Table 22 Master CPU bus interface (MBI; RD, WR separate type) (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{su}(S-R)	$\overline{S0}$, $\overline{S1}$ setup time for read	0			ns
t_{su}(S-W)	$\overline{S0}$, $\overline{S1}$ setup time for write	0			ns
t_h(R-S)	$\overline{S0}$, $\overline{S1}$ hold time for read	0			ns
t_h(W-S)	$\overline{S0}$, $\overline{S1}$ hold time for write	0			ns
t_{su}(A-R)	A0 setup time for read	10			ns
t_{su}(A-W)	A0 setup time for write	10			ns
t_h(R-A)	A0 hold time for read	0			ns
t_h(W-A)	A0 hold time for write	0			ns
t_w(R)	Read pulse width	80			ns
t_w(W)	Write pulse width	80			ns
t_{su}(D-W)	Data input setup time before write	35			ns
t_h(W-D)	Data input hold time after write	0			ns
t_a(R-D)	Data output enable time after read			65	ns
t_v(R-D)	Data output disable time after read	10			ns
t_v(R-OBF)	OBF output transmission time after read			50	ns
t_d(W-IBF)	$\overline{IBF}$ output transmission time after write			50	ns

In Vcc = 3 V

Table 23 Master CPU bus interface (MBI; R/$\overline{W}$ type) (Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{su}(S-E)	$\overline{S0}$, $\overline{S1}$ setup time	0			ns
t_h(E-S)	$\overline{S0}$, $\overline{S1}$ hold time	0			ns
t_{su}(A-E)	A0 setup time	10			ns
t_h(E-A)	A0 hold time	0			ns
t_{su}(RW-E)	R/$\overline{W}$ setup time	10			ns
t_h(E-RW)	R/$\overline{W}$ hold time	10			ns
t_w(E)	Enable pulse width	80			ns
t_w(E-E)	Enable pulse interval	80			ns
t_{su}(D-E)	Data input setup time before write	35			ns
t_h(E-D)	Data input hold time after write	0			ns
t_a(E-D)	Data output enable time after read			65	ns
t_v(E-D)	Data output disable time after read	10			ns
t_v(E-OBF)	OBF output transmission time after E inactive			50	ns
t_d(E-IBF)	$\overline{IBF}$ output transmission time after E inactive			50	ns

In Vcc = 3 V

Table 24 Timing requirements and switching characteristics in memory expansion and microprocessor modes

(Vcc = 3.0 to 3.6 V, Vss = 0 V, Ta = –20 to 70°C, unless otherwise noted)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
$t_C(\phi)$	ϕ clock cycle time	166.66			µs
$t_{WH}(\phi)$	ϕ clock "H" pulse width	0.5•tc(ϕ) – 5			ns
$t_{WL}(\phi)$	ϕ clock "L" pulse width	0.5•tc(ϕ) – 5			ns
$t_d(\phi$ -AH)	AB_{15}–AB_8 delay time			45	ns
$t_v(\phi$ -AH)	AB_{15}–AB_8 valid time	7			ns
$t_d(\phi$ -AL)	AB_7–AB_0 delay time			47	ns
$t_v(\phi$ -AL)	AB_7–AB_0 valid time	7			ns
$t_d(\phi$ -WR)	$\overline{WR}$ delay time			8	ns
$t_v(\phi$ -WR)	$\overline{WR}$ valid time	4			ns
$t_d(\phi$ -RD)	$\overline{RD}$ delay time			8	ns
$t_v(\phi$ -RD)	$\overline{RD}$ valid time	3			ns
$t_d(\phi$ -SYNC)	$SYNC_{OUT}$ delay time			11	ns
$t_v(\phi$ -SYNC)	$SYNC_{OUT}$ valid time	4			ns
$t_d(\phi$ -DMA)	DMA_{OUT} delay time			26	ns
$t_v(\phi$ -DMA)	DMA_{OUT} valid time	9			ns
t_{su}(RDY- ϕ)	RDY setup time	35			ns
$t_h(\phi$ -RDY)	RDY hold time	0			ns
t_{su}(HOLD- ϕ)	$\overline{HOLD}$ setup time	21			ns
$t_h(\phi$ -HOLD)	$\overline{HOLD}$ hold time	0			ns
$t_d(\phi$ -HLDAL)	$\overline{HOLD}$ "L" delay time			30	ns
$t_d(\phi$ -HLDAH)	$\overline{HOLD}$ "H" delay time			30	ns
t_{su}(DB- ϕ)	Data bus setup time	9			ns
$t_h(\phi$ -DB)	Data bus hold time	0			ns
$t_d(\phi$ -DB)	Data bus delay time			30	ns
$t_v(\phi$ -DB)	Data bus valid time (**Note 1**)	15			ns
$t_d(\phi$ -EDMA)	$\overline{EDMA}$ delay time			12	ns
$t_v(\phi$ -EDMA)	$\overline{EDMA}$ valid time	8			ns
t_{WL}(WR) (**Note 2**)	$\overline{WR}$ pulse width	0.5•tc(ϕ) – 6			ns
t_{WL}(RD) (**Note 2**)	$\overline{RD}$ pulse width	0.5•tc(ϕ) – 6			ns
t_d(AH-WR)	AB_{15}–AB_8 valid time before $\overline{WR}$	0.5•tc(ϕ) – 33			ns
t_d(AL-WR)	AB_7–AB_0 valid time before $\overline{WR}$	0.5•tc(ϕ) – 35			ns
t_v(WR-AH)	AB_{15}–AB_8 valid time after $\overline{WR}$	0			ns
t_v(WR-AL)	AB_7–AB_0 valid time after $\overline{WR}$	0			ns
t_d(AH-RD)	AB_{15}–AB_8 valid time before $\overline{RD}$	0.5•tc(ϕ) – 33			ns
t_d(AL-RD)	AB_7–AB_0 valid time before $\overline{RD}$	0.5•tc(ϕ) – 35			ns
t_v(RD-AH)	AB_{15}–AB_8 valid time after $\overline{RD}$	0			ns
t_v(RD-AL)	AB_7–AB_0 valid time after $\overline{RD}$	0			ns
t_{su}(RDY-WR)	RDY setup time before $\overline{WR}$	45			ns
t_h(WR-RDY)	RDY hold time after $\overline{WR}$	0			ns
t_{su}(RDY-RD)	RDY setup time before $\overline{RD}$	45			ns
t_h(RD-RDY)	RDY hold time after $\overline{RD}$	0			ns
t_{su}(DB-RD)	Data bus setup time before $\overline{RD}$	18			ns
t_h(RD-DB)	Data bus hold time after $\overline{RD}$	0			ns
t_d(WR-DB)	Data bus delay time before $\overline{WR}$			28	ns
t_v(WR-DB)	Data bus valid time after $\overline{WR}$ (**Note 1**)	12			ns
t_v(WR-EDMA)	$\overline{EDMA}$ delay time after $\overline{WR}$	3			ns
t_v(RD-EDMA)	$\overline{EDMA}$ valid time after $\overline{RD}$	3			ns
t_r(D+), t_r(D-)	USB output rise time, C_L = 50 pF	4		20	ns
t_f(D+), t_f(D-)	USB output fall time, C_L = 50 pF	4		20	ns

Notes 1: Test conditions: I_{OHL} = ± 5mA, C_L = 50 pF
2: t_{WL}(RD) = ((n + 0.5) • tc(PHI)) – 5 ns (n = wait number)
t_{WL}(WR) = ((n + 0.5) • tc(PHI)) – 5 ns (n = wait number)
For example, two software waits, PHI = 12 MHz operating
t_{WL}(RD) = 2.5 • tc(PHI) – 5 ns = 203.33 ns

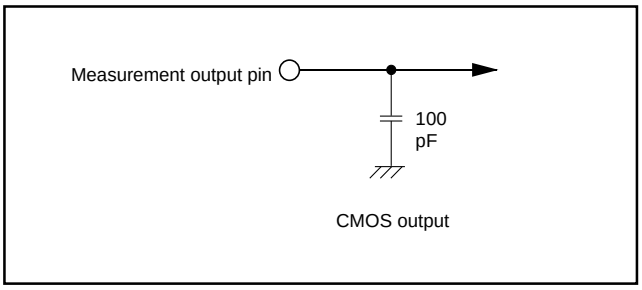
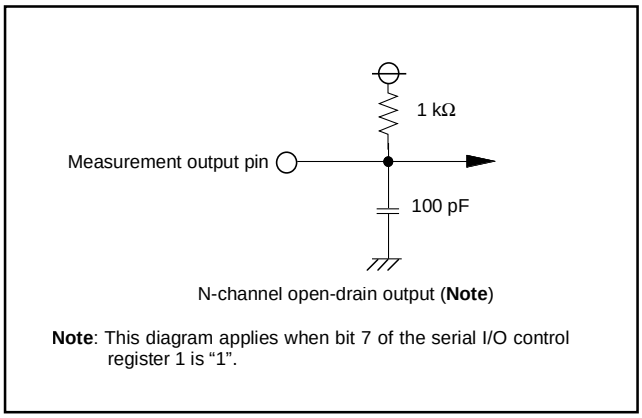


Fig. 79 Circuit for measuring output switching characteristics (1)



Note: This diagram applies when bit 7 of the serial I/O control register 1 is “1”.

Fig. 80 Circuit for measuring output switching characteristics (2)

● Timing diagram

[Interrupt]

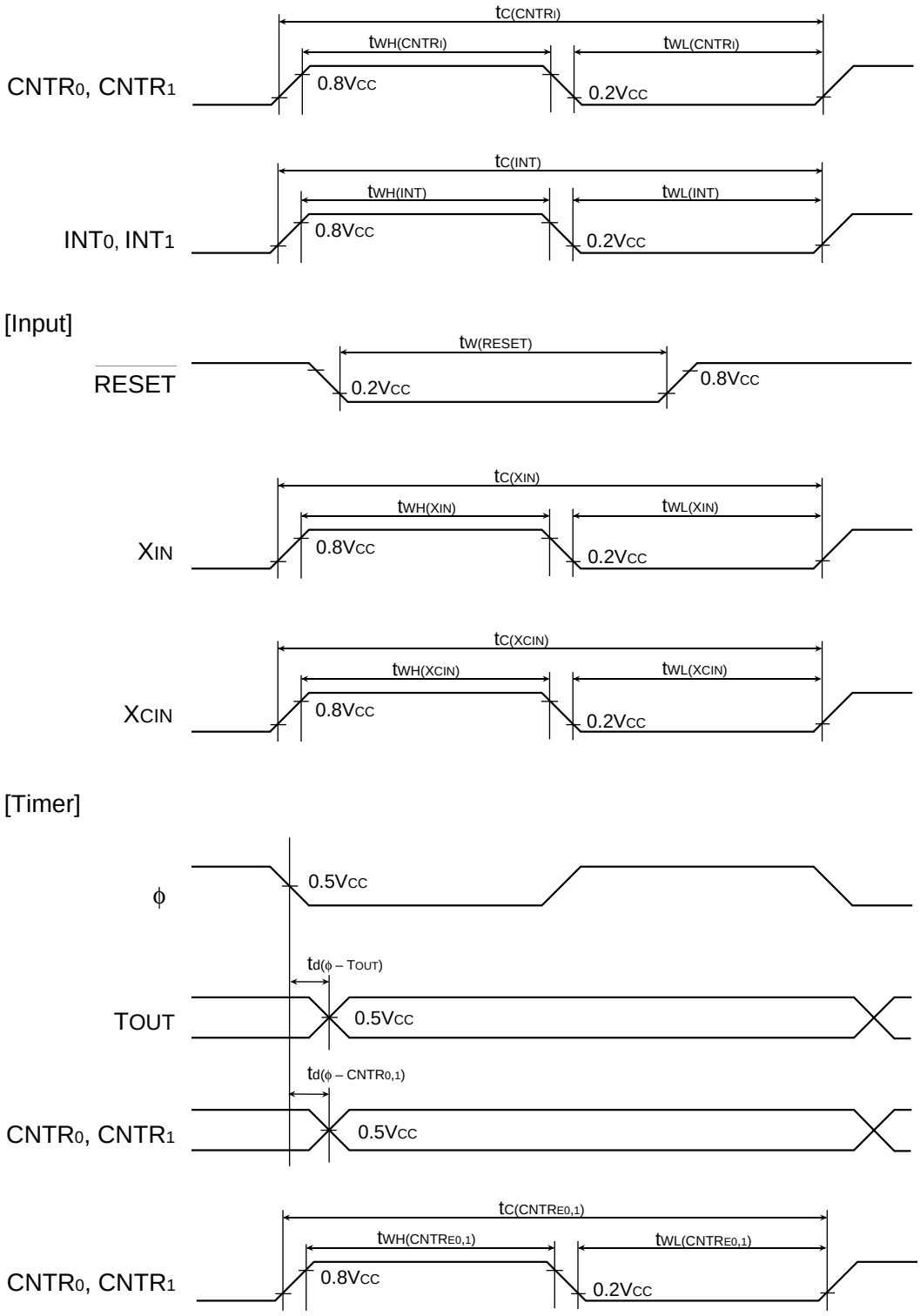


Fig. 81 Timing diagram (1)

● Timing diagram

[Serial I/O]

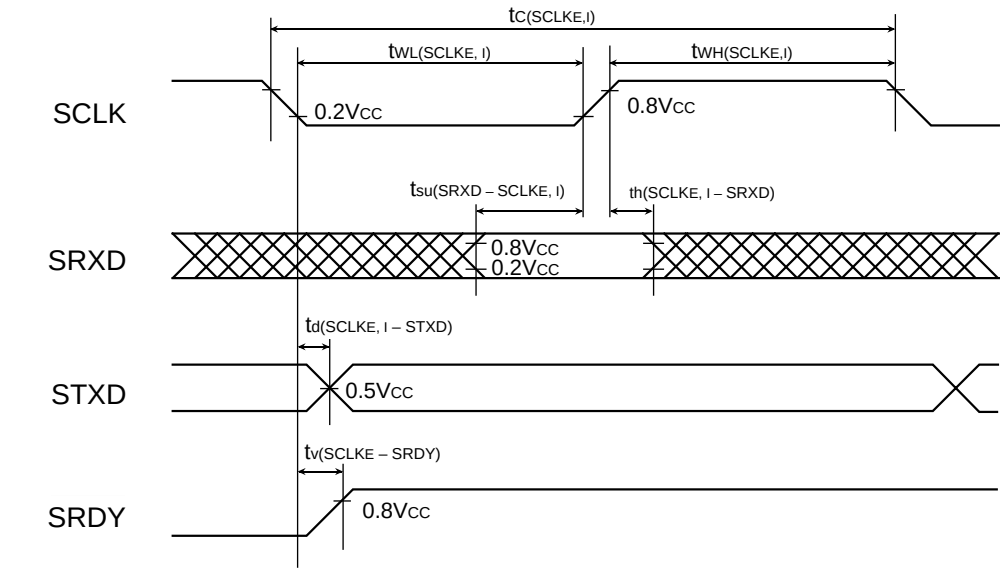
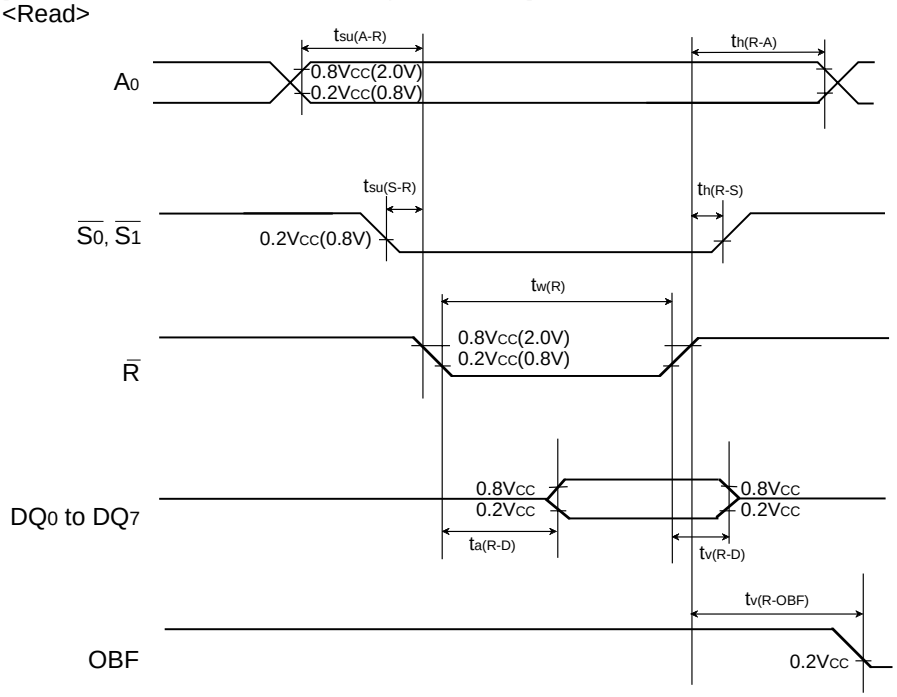


Fig. 82 Timing diagram (2)

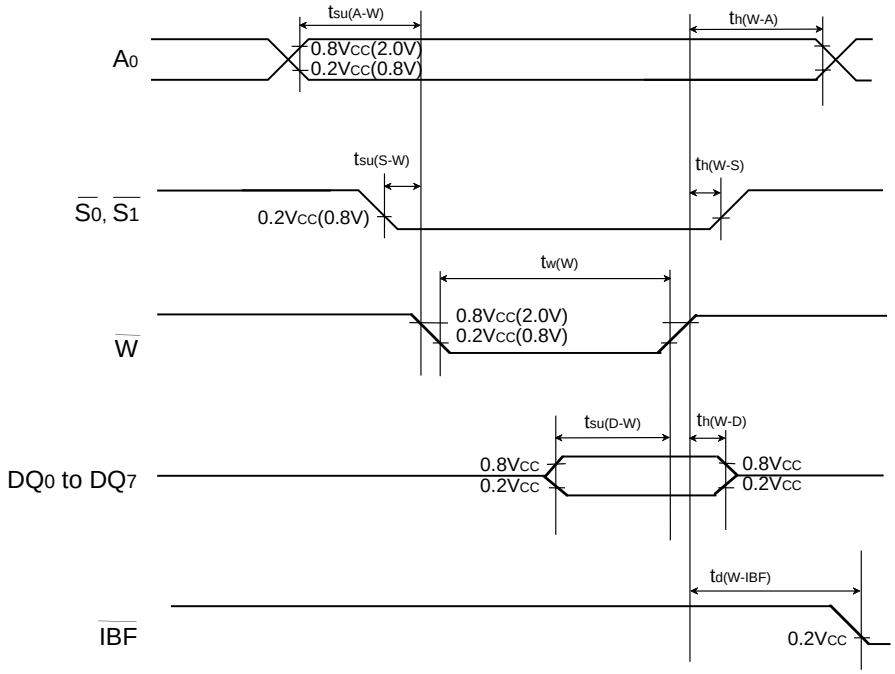
Fig. 83 Timing diagram (3)

● Timing diagram

[Master CPU bus interface: R/$\overline{W}$ separate mode]
<Read>



<Write>

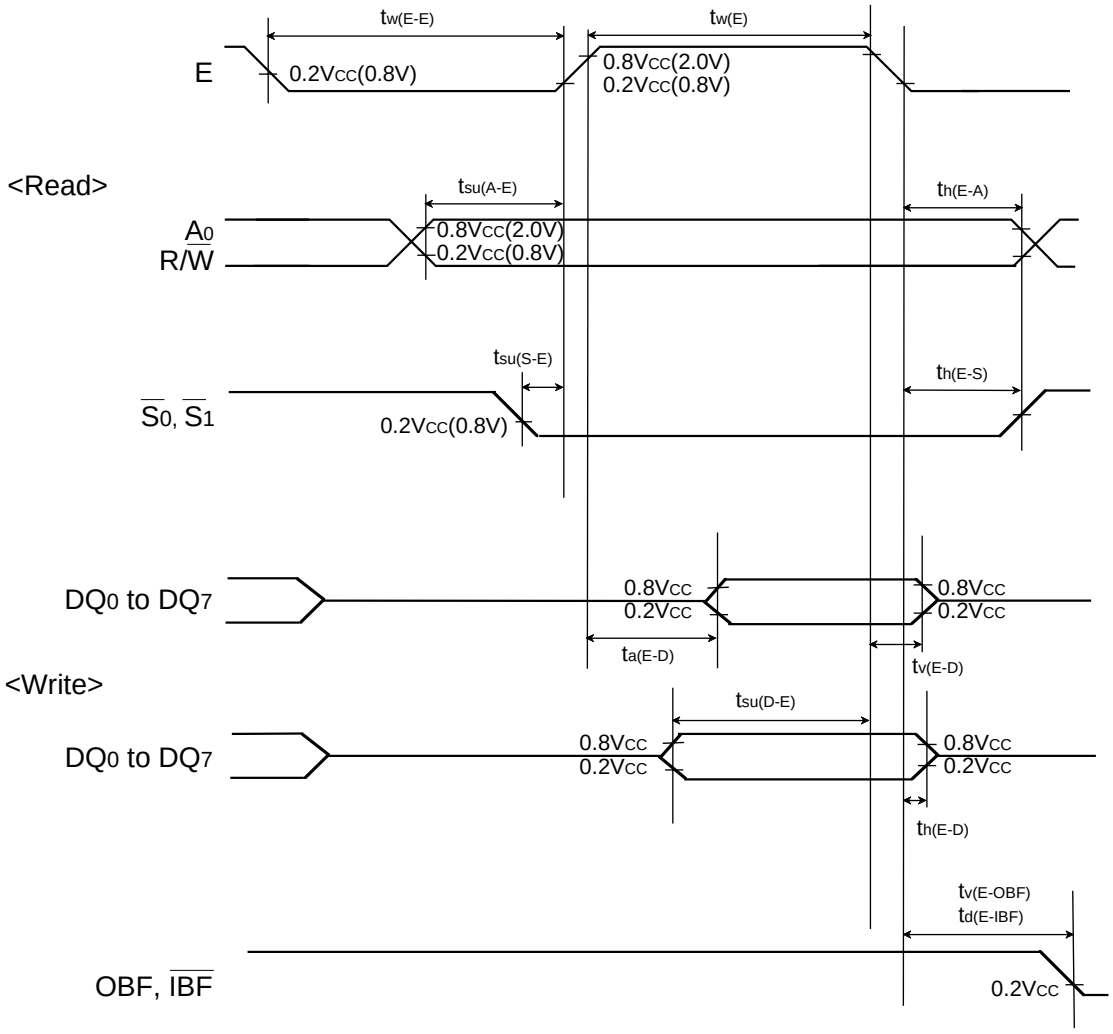


Note: This timing applies in the case of the master bus input level select bit (PTC7) = "1" (TTL level input)

Fig. 84 Timing diagram (4)

● Timing diagram

[Master CPU bus interface: R/$\overline{W}$ mode]



Note: This timing applies in the case of the master bus input level select bit (PTC7) = "1" (TTL level input)

Fig. 85 Timing diagram (5)

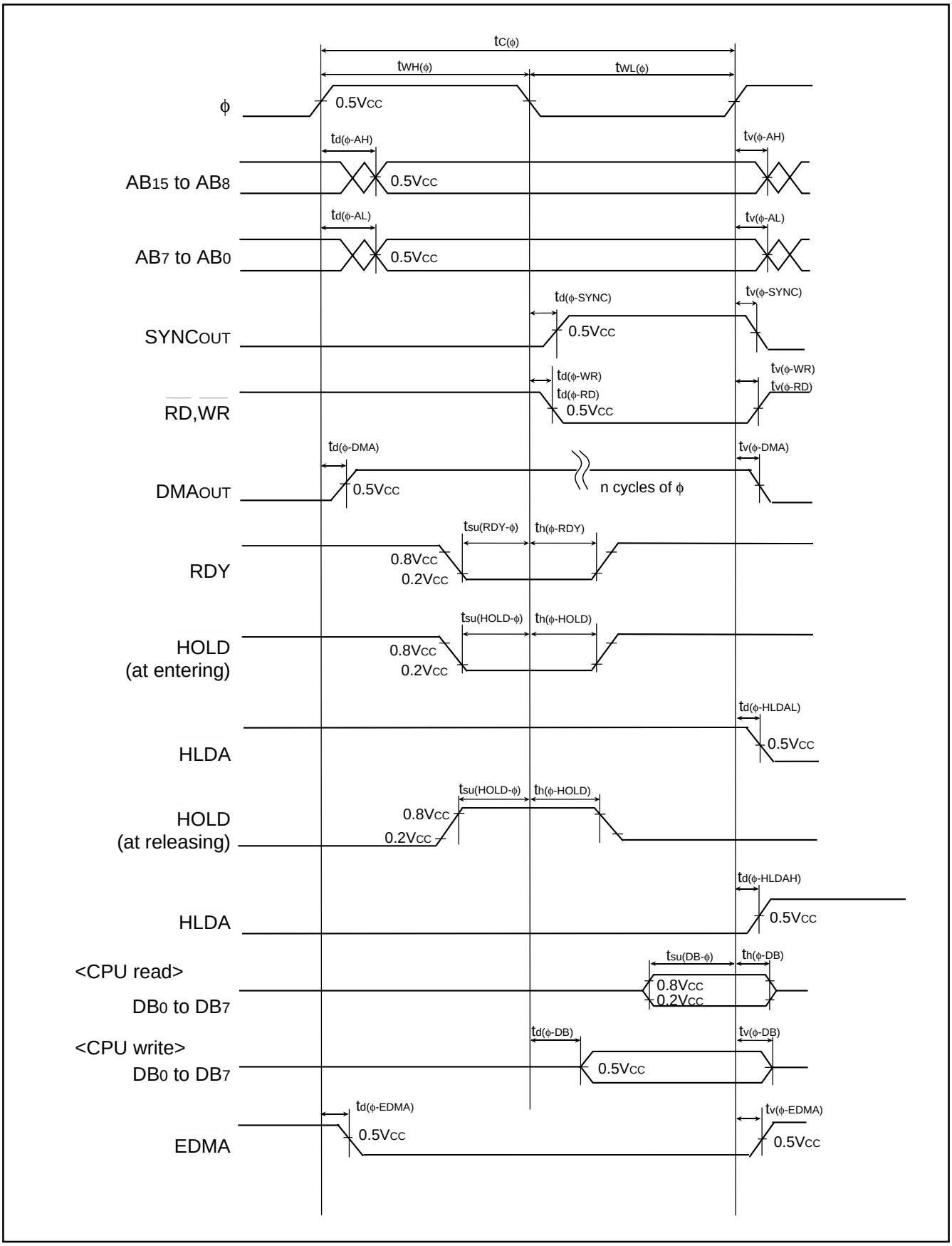


Fig. 86 Timing diagram (6); Memory expansion and microprocessor modes

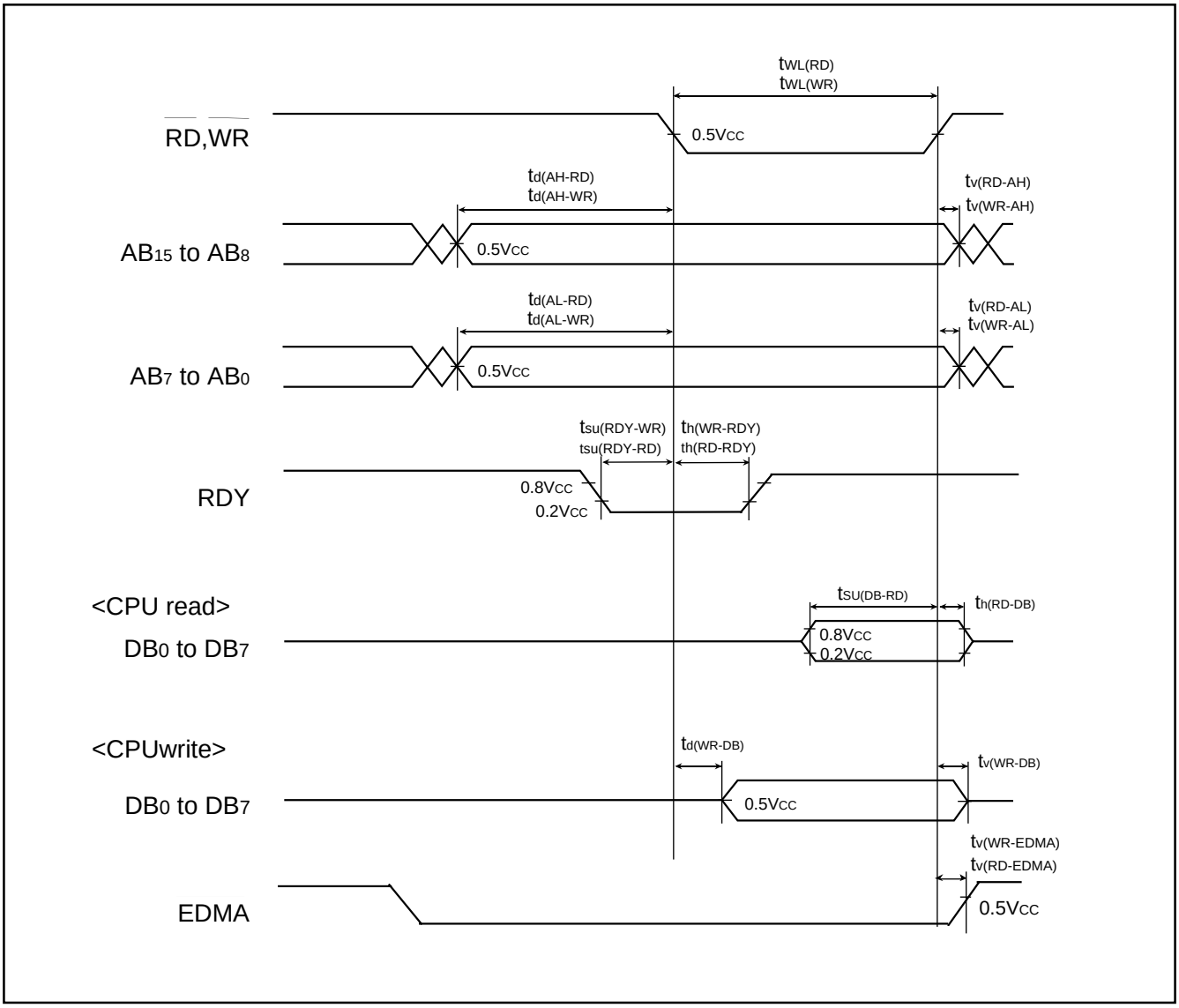


Fig. 87 Timing diagram (7); Memory expansion and microprocessor modes

FLASH MEMORY MODE

The M37641F8FP/HP (flash memory version) has an internal new DINOR (DIvided bit line NOR) flash memory that can be rewritten with a single power source when V_{CC} is 5 V, and 2 power sources when V_{PP} is 5 V and V_{CC} is 3.3 V in the CPU rewrite and standard serial I/O modes.

For this flash memory, three flash memory modes are available in which to read, program, and erase: the parallel I/O and standard serial I/O modes in which the flash memory can be manipulated using a programmer and the CPU rewrite mode in which the flash memory can be manipulated by the Central Processing Unit (CPU).

Summary

Table 25 lists the summary of the M37641F8 (flash memory version).

This flash memory version has some blocks on the flash memory as shown in Figure 88 and each block can be erased. The flash memory is divided into User ROM area and Boot ROM area.

In addition to the ordinary User ROM area to store the MCU operation control program, the flash memory has a Boot ROM area that is used to store a program to control rewriting in CPU rewrite and standard serial I/O modes. This Boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the factory. However, the user can write a rewrite control program in this area that suits the user's application system. This Boot ROM area can be rewritten in only parallel I/O mode.

Table 25 Summary of M37641F8 (flash memory version)

Item		Specifications
Power source voltage		Vcc = 3.00 – 3.60 V, 4.15 – 5.25 V (f(X_{IN}) = 24 MHz, ϕ = 6 MHz)
Program/Erase V_{PP} voltage		V_{PP} = 4.50 – 5.25 V (f(X_{IN}) = 24 MHz, ϕ = 6 MHz)
Flash memory mode		3 modes; Flash memory can be manipulated as follows: •Parallel I/O mode: Manipulated using an external programmer •Standard serial I/O mode: Manipulated using an external programmer •CPU rewrite mode: Manipulated by the Central Processing Unit (CPU).
Erase block division	User ROM area	See Figure 88.
	Boot ROM area	1 block (4 Kbytes) (**Note**)
Program method		Byte program
Erase method		Batch erasing/Block erasing
Program/Erase control method		Program/Erase control by software command
Number of commands		6 commands
Number of program/Erase times		100 times
ROM code protection		Available in parallel I/O mode and standard serial I/O mode

Note: The Boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the factory. This Boot ROM area can be rewritten in only parallel I/O mode.

(1) CPU Rewrite Mode

In CPU rewrite mode, the internal flash memory can be operated on (read, program, or erase) under control of the Central Processing Unit (CPU).

In CPU rewrite mode, only the User ROM area shown in Figure 88 can be rewritten; the Boot ROM area cannot be rewritten. Make sure the program and block erase commands are issued for only the User ROM area and each block area.

The control program for CPU rewrite mode can be stored in either User ROM or Boot ROM area. In the CPU rewrite mode, because the flash memory cannot be read from the CPU, the rewrite control program must be transferred to internal RAM area to be executed before it can be executed.

Microcomputer Mode and Boot Mode

The control program for CPU rewrite mode must be written into the User ROM or Boot ROM area in parallel I/O mode beforehand. (If the control program is written into the Boot ROM area, the standard serial I/O mode becomes unusable.)

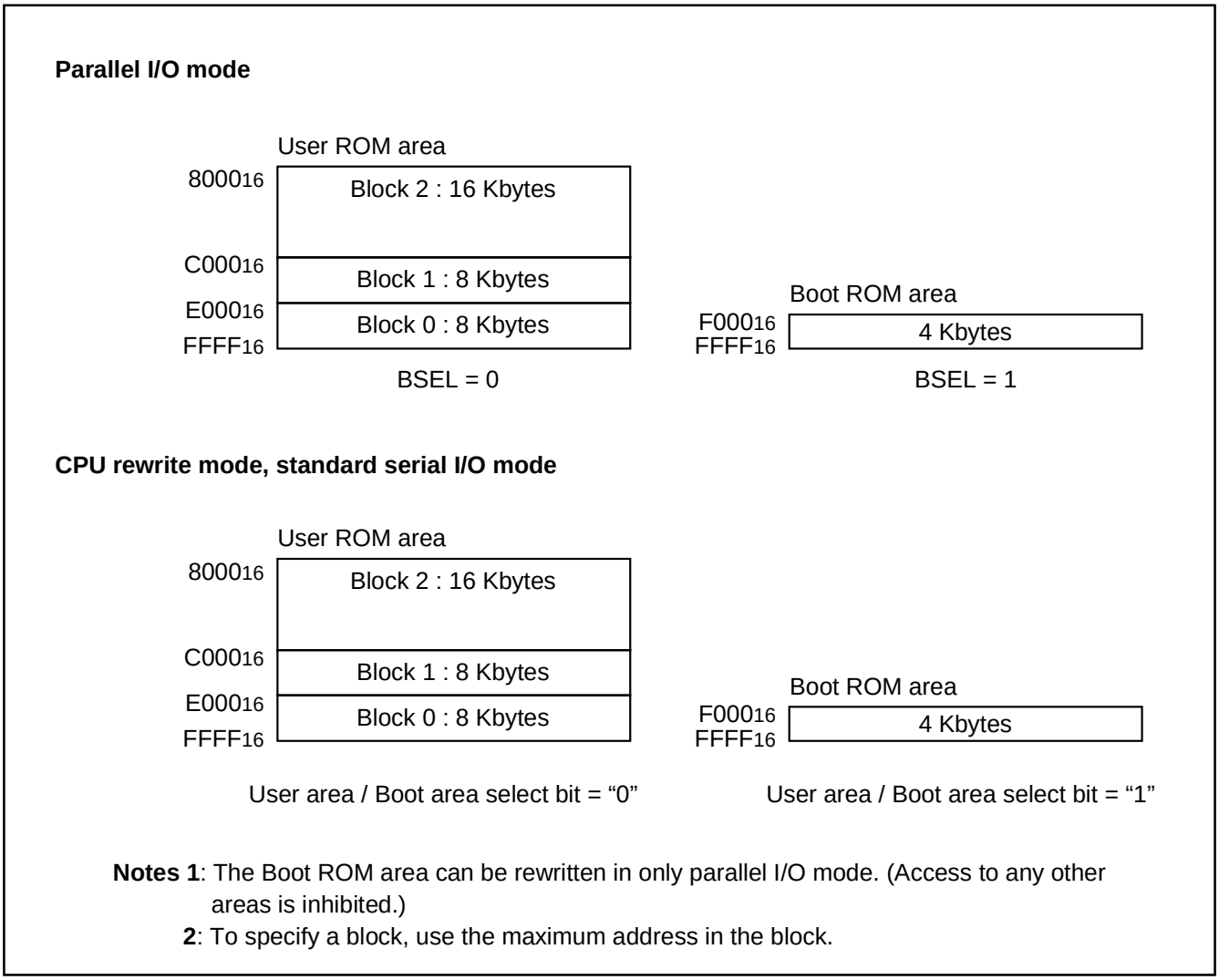
See Figure 88 for details about the Boot ROM area.

Normal microcomputer mode is entered when the microcomputer is reset with pulling CNVSS pin low. In this case, the CPU starts operating using the control program in the User ROM area.

When the microcomputer is reset by pulling the P81 pin high, the CNVSS pin high, the CPU starts operating using the control program in the Boot ROM area. This mode is called the “Boot” mode.

Block Address

Block addresses refer to the maximum address of each block. These addresses are used in the block erase command.



Notes 1: The Boot ROM area can be rewritten in only parallel I/O mode. (Access to any other areas is inhibited.)

2: To specify a block, use the maximum address in the block.

Fig. 88 Block diagram of built-in flash memory

Outline Performance (CPU Rewrite Mode)

CPU rewrite mode is usable in the single-chip, memory expansion or Boot mode. The only User ROM area can be rewritten in CPU rewrite mode.

In CPU rewrite mode, the CPU erases, programs and reads the internal flash memory as instructed by software commands. This rewrite control program must be transferred to a memory such as the internal RAM before it can be executed.

The MCU enters CPU rewrite mode by applying 4.50 V to 5.25 V to the CNVSS pin and setting "1" to the CPU Rewrite Mode Select Bit (bit 1 of address $006A_{16}$). Software commands are accepted once the mode is entered.

Use software commands to control program and erase operations. Whether a program or erase operation has terminated normally or in error can be verified by reading the status register.

Figure 89 shows the flash memory control register.

Bit 0 is the RY/$\overline{BY}$ status flag used exclusively to read the operating status of the flash memory. During programming and erase operations, it is "0" (busy). Otherwise, it is "1" (ready). This is equivalent to the RY/$\overline{BY}$ pin function in parallel I/O mode.

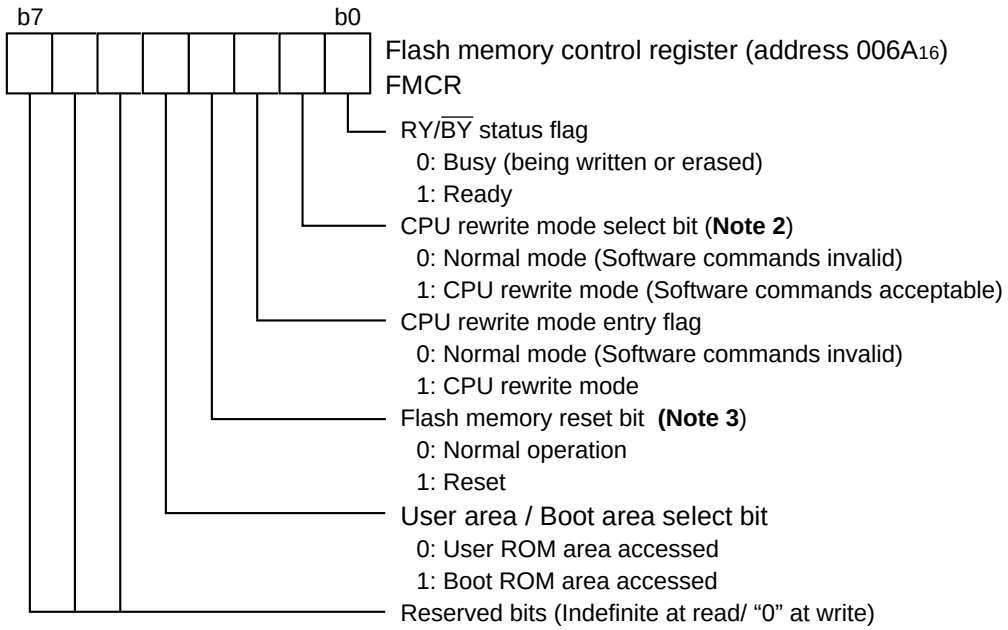
Bit 1 is the CPU Rewrite Mode Select Bit. When this bit is set to "1", the MCU enters CPU rewrite mode. Software commands are accepted once the mode is entered. In CPU rewrite mode, the CPU becomes unable to access the internal flash memory directly. Therefore, use the control program in a memory other than internal flash memory for write to bit 1. To set this bit to "1", it is necessary to write "0" and then write "1" in succession. The bit can be set to "0" by only writing "0".

Bit 2 is the CPU Rewrite Mode Entry Flag. This flag indicates "1" in CPU rewrite mode, so that reading this flag can check whether CPU rewrite mode has been entered or not.

Bit 3 is the flash memory reset bit used to reset the control circuit of internal flash memory. This bit is used when exiting CPU rewrite mode and when flash memory access has failed. When the CPU Rewrite Mode Select Bit is "1", setting "1" for this bit resets the control circuit. To set this bit to "1", it is necessary to write "0" and then write "1" in succession. To release the reset, it is necessary to set this bit to "0".

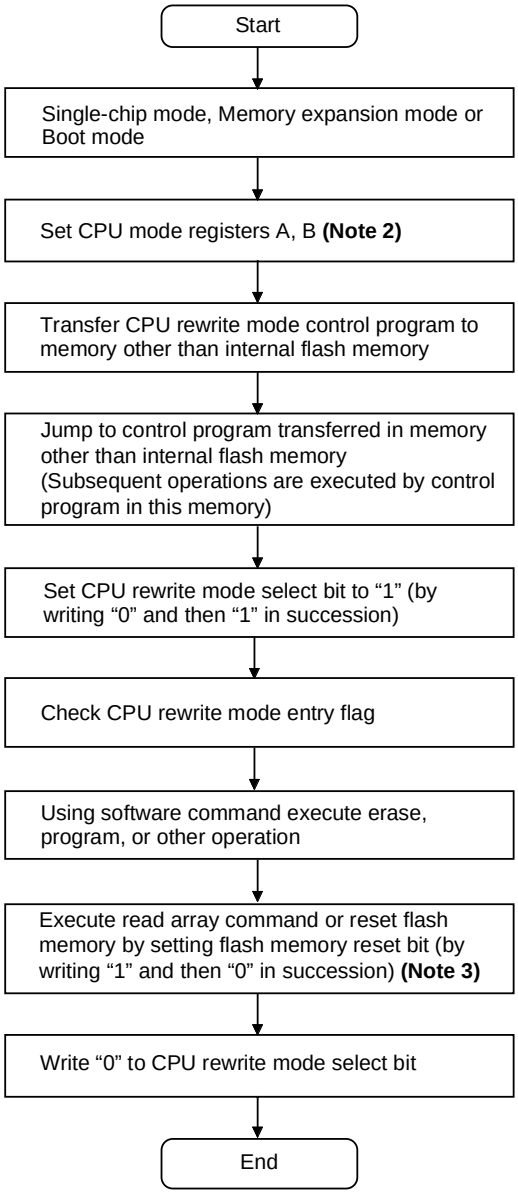
Bit 4 is the User Area/Boot Area Select Bit. When this bit is set to "1", Boot ROM area is accessed, and CPU rewrite mode in Boot ROM area is available. In Boot mode, this bit is set to "1" automatically. Reprogramming of this bit must be in a memory other than internal flash memory.

Figure 90 shows a flowchart for setting/releasing CPU rewrite mode.



Notes 1: The contents of flash memory control register are "XXX00001" just after reset release.
2: For this bit to be set to "1", the user needs to write "0" and then "1" to it in succession. If it is not this procedure, this bit will not be set to "1". Additionally, it is required to ensure that no interrupt will be generated during that interval.
Use the control program in the area except the built-in flash memory for write to this bit.
3: This bit is valid when the CPU rewrite mode select bit is "1". Set this bit 3 to "0" subsequently after setting bit 3 to "1".
4: Use the control program in the area except the built-in flash memory for write to this bit.

Fig. 89 Structure of flash memory control register



Notes 1: When starting the MCU in the single-chip mode or memory expansion mode, supply 4.5 V to 5.25 V to the CNVss pin until checking the CPU rewrite mode entry flag.
2: Set the main clock as follows depending on the X_{IN} divider select bit of clock control register (bit 7 of address $001F_{16}$):
When X_{IN} divider select bit = "0" ($\phi = f(X_{IN})/4$), the main clock is 24 MHz or less
When X_{IN} divider select bit = "1" ($\phi = f(X_{IN})/2$), the main clock is 12 MHz or less.
3: Before exiting the CPU rewrite mode after completing erase or program operation, always be sure to execute the read array command or reset the flash memory.

Fig. 90 CPU rewrite mode set/release flowchart

Notes on CPU Rewrite Mode

The below notes applies when rewriting the flash memory in CPU rewrite mode.

●Operation speed

During CPU rewrite mode, set the internal clock ϕ to 6 MHz or less using the X_{IN} Divider Select Bit (bit 7 of address $001F_{16}$).

●Instructions inhibited against use

The instructions which refer to the internal data of the flash memory cannot be used during CPU rewrite mode .

●Interrupts inhibited against use

The interrupts cannot be used during CPU rewrite mode because they refer to the internal data of the flash memory.

●Reset

Reset is always valid. If reset is performed on erasing, 5 ms is needed until the flash memory will return to its normal state. Accordingly, wait at least 5 ms before releasing reset.

Software Commands

Table 26 lists the software commands.
After setting the CPU Rewrite Mode Select Bit to "1", write a software command to specify an erase or program operation.
Each software command is explained below.

●Read Array Command (FF_{16})

The read array mode is entered by writing the command code "FF_{16}" in the first bus cycle. When an address to be read is input in one of the bus cycles that follow, the contents of the specified address are read out at the data bus (DB_0 to DB_7).
The read array mode is retained intact until another command is written.

●Read Status Register Command (70_{16})

When the command code "70_{16}" is written in the first bus cycle, the contents of the status register are read out at the data bus (DB_0 to DB_7) by a read in the second bus cycle.
The status register is explained in the next section.

●Clear Status Register Command (50_{16})

This command is used to clear the bits SR4 and SR5 of the status register after they have been set. These bits indicate that operation has ended in an error. To use this command, write the command code "50_{16}" in the first bus cycle.

●Program Command (40_{16})

Program operation starts when the command code "40_{16}" is written in the first bus cycle. Then, if the address and data to program are written in the 2nd bus cycle, program operation (data programming and verification) will start.
Whether the write operation is completed can be confirmed by reading the status register or the RY/$\overline{BY}$ Status Flag. When the program starts, the read status register mode is entered automatically and the contents of the status register is read at the data bus (DB_0 to DB_7). The status register bit 7 (SR7) is set to "0" at the same time the write operation starts and is returned to "1" upon completion of the write operation. In this case, the read status register mode remains active until the next command is written.
The RY/$\overline{BY}$ Status Flag is "0" during write operation and "1" when the write operation is completed as is the status register bit 7.
At program end, program results can be checked by reading the status register.

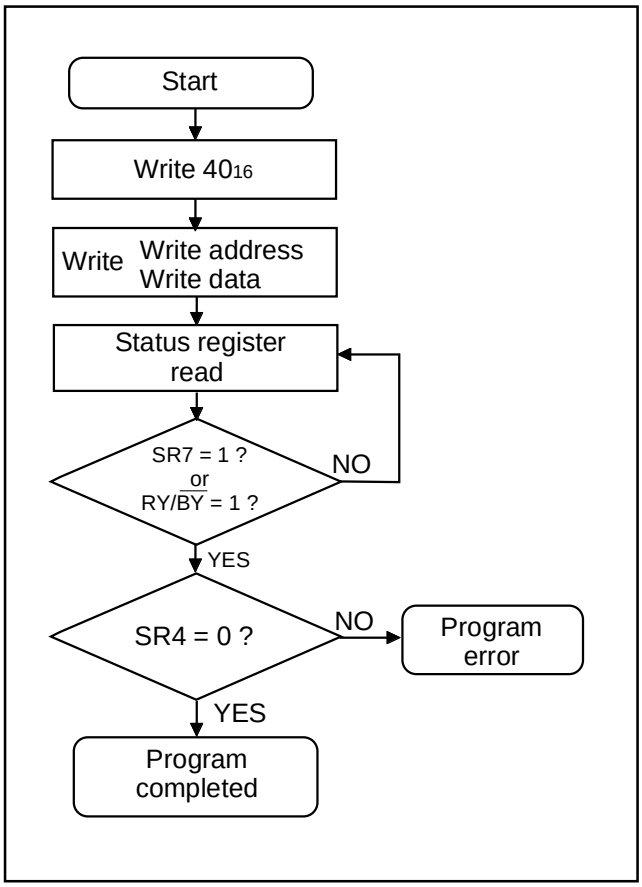


Fig. 91 Program flowchart

Table 26 List of software commands (CPU rewrite mode)

Command	Cycle number	First bus cycle			Second bus cycle		
		Mode	Address	Data (DB_0 to DB_7)	Mode	Address	Data (DB_0 to DB_7)
Read array	1	Write	X **(Note 4)**	FF_{16}			
Read status register	2	Write	X	70_{16}	Read	X	SRD **(Note 1)**
Clear status register	1	Write	X	50_{16}			
Program	2	Write	X	40_{16}	Write	WA **(Note 2)**	WD **(Note 2)**
Erase all blocks	2	Write	X	20_{16}	Write	X	20_{16}
Block erase	2	Write	X	20_{16}	Write	BA **(Note 3)**	$D0_{16}$

Notes 1: SRD = Status Register Data
2: WA = Write Address, WD = Write Data
3: BA = Block Address to be erased (Input the maximum address of each block.)
4: X denotes a given address in the User ROM area .

●Erase All Blocks Command (20_{16}/20_{16})

By writing the command code "20_{16}" in the first bus cycle and the confirmation command code "20_{16}" in the second bus cycle that follows, the operation of erase all blocks (erase and erase verify) starts.

Whether the erase all blocks command is terminated can be confirmed by reading the status register or the RY/$\overline{BY}$ Status Flag of flash memory control register. When the erase all blocks operation starts, the read status register mode is entered automatically and the contents of the status register can be read out at the data bus (DB_0 to DB_7). The status register bit 7 (SR7) is set to "0" at the same time the erase operation starts and is returned to "1" upon completion of the erase operation. In this case, the read status register mode remains active until another command is written.

The RY/$\overline{BY}$ Status Flag is "0" during erase operation and "1" when the erase operation is completed as is the status register bit 7.

After the erase all blocks end, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed.

●Block Erase Command (20_{16}/$D0_{16}$)

By writing the command code "20_{16}" in the first bus cycle and the confirmation command code "$D0_{16}$" and the blobk address in the second bus cycle that follows, the block erase (erase and erase verify) operation starts for the block address of the flash memory to be specified.

Whether the block erase operation is completed can be confirmed by reading the status register or the RY/$\overline{BY}$ Status Flag of flash memory control register. At the same time the block erase operation starts, the read status register mode is automatically entered, so that the contents of the status register can be read out. The status register bit 7 (SR7) is set to "0" at the same time the block erase operation starts and is returned to "1" upon completion of the block erase operation. In this case, the read status register mode remains active until the read array command (FF_{16}) is written.

The RY/$\overline{BY}$ Status Flag is "0" during block erase operation and "1" when the block erase operation is completed as is the status register bit 7.

After the block erase ends, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed.

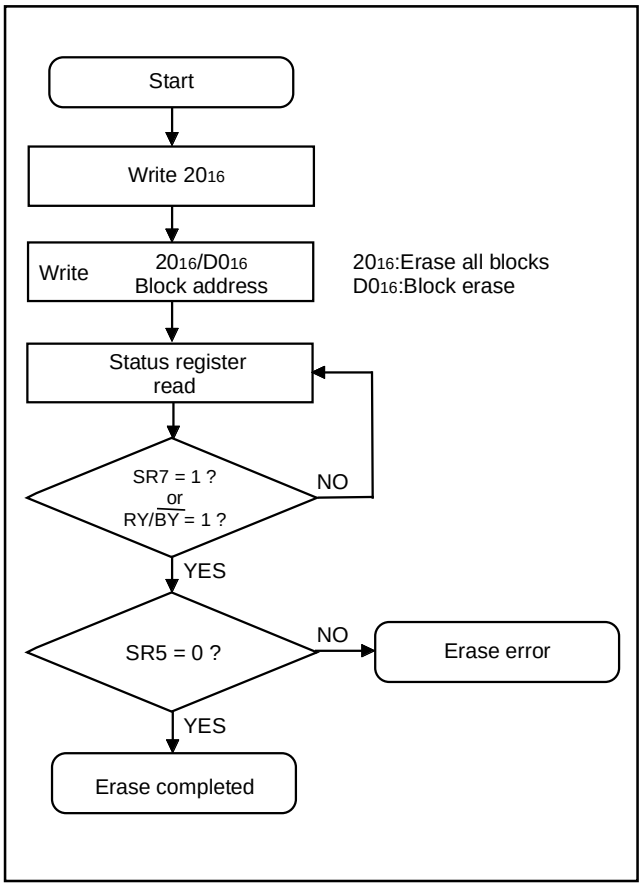


Fig. 92 Erase flowchart

Status Register

The status register shows the operating status of the flash memory and whether erase operations and programs ended successfully or in error. It can be read in the following ways:

(1) By reading an arbitrary address from the User ROM area after writing the read status register command (70_{16})
(2) By reading an arbitrary address from the User ROM area in the period from when the program starts or erase operation starts to when the read array command (FF_{16}) is input.

Also, the status register can be cleared by writing the clear status register command (50_{16}).
After reset, the status register is set to "80_{16}".
Table 27 shows the status register. Each bit in this register is explained below.

•Sequencer status (SR7)

The sequencer status indicates the operating status of the flash memory. This bit is set to "0" (busy) during write or erase operation and is set to "1" when these operations ends.
After power-on, the sequencer status is set to "1" (ready).

•Erase status (SR5)

The erase status indicates the operating status of erase operation. If an erase error occurs, it is set to "1". When the erase status is cleared, it is set to "0".

•Program status (SR4)

The program status indicates the operating status of write operation. When a write error occurs, it is set to "1".
The program status is set to "0" when it is cleared.

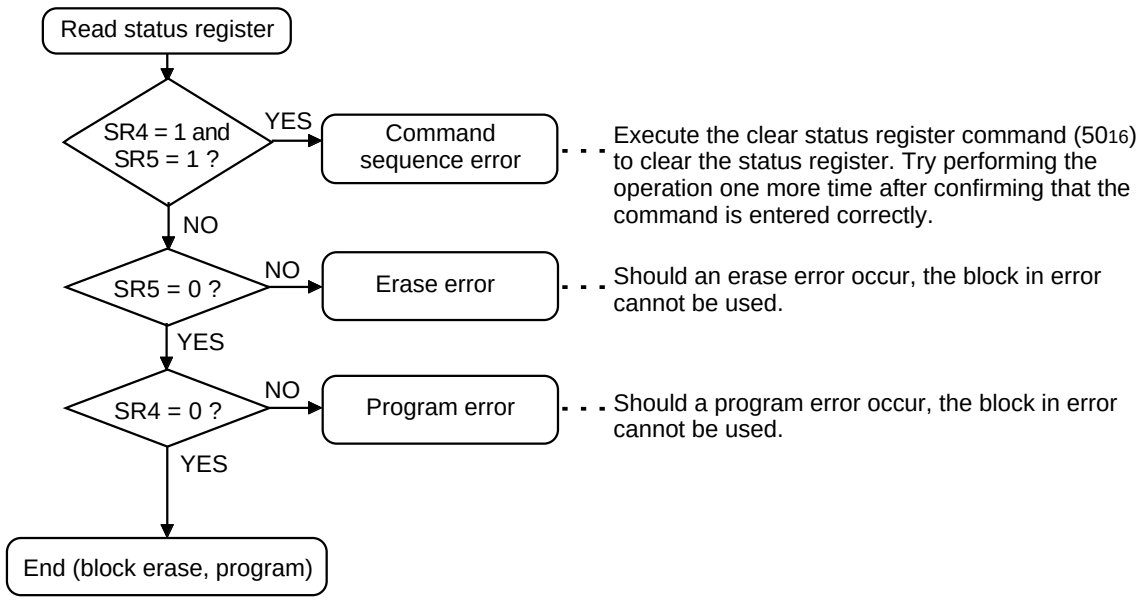
If "1" is written for any of the SR5 and SR4 bits, the program, erase all blocks, and block erase commands are not accepted. Before executing these commands, execute the clear status register command (50_{16}) and clear the status register.
Also, if any commands are not correct, both SR5 and SR4 are set to "1".

Table 27 Definition of each bit in status register

Each bit of SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Sequencer status	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase status	Terminated in error	Terminated normally
SR4 (bit4)	Program status	Terminated in error	Terminated normally
SR3 (bit3)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

Full Status Check

By performing full status check, it is possible to know the execution results of erase and program operations. Figure 93 shows a full status check flowchart and the action to be taken when each error occurs.



Note: When one of SR5 and SR4 is set to "1", none of the program, erase all blocks, and block erase commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

Fig. 93 Full status check flowchart and remedial procedure for errors

Functions To Inhibit Rewriting Flash Memory Version

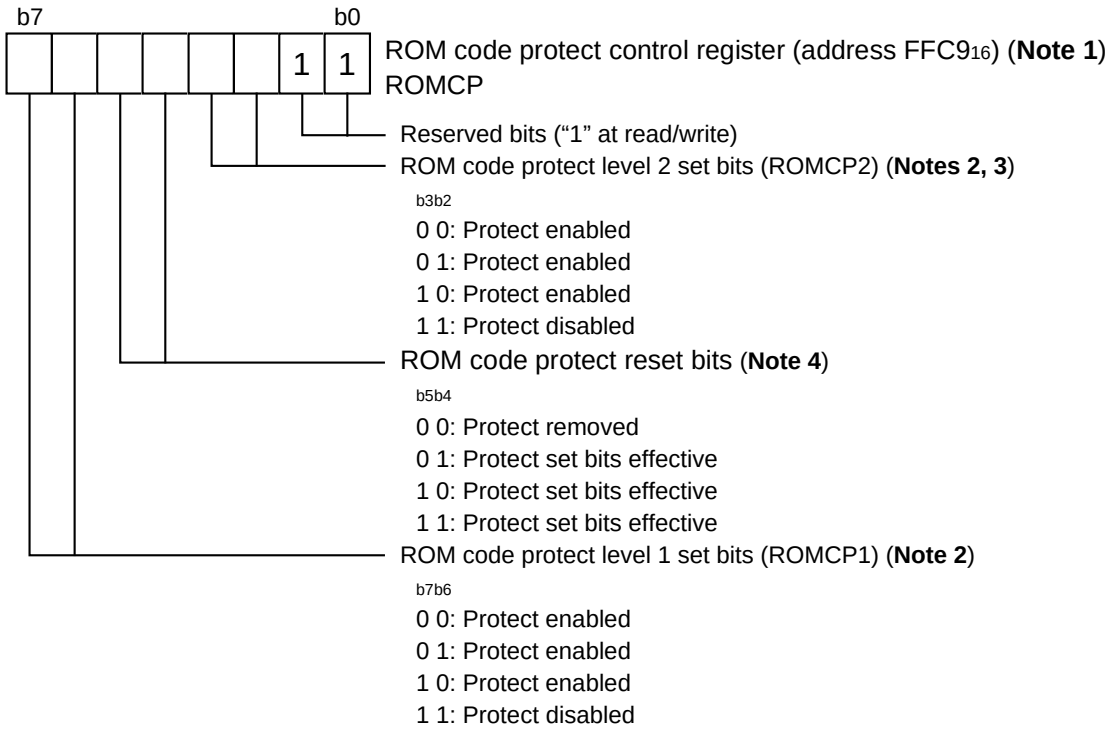
To prevent the contents of internal flash memory from being read out or rewritten easily, this MCU incorporates a ROM code protect function for use in parallel I/O mode and an ID code check function for use in standard serial I/O mode.

●ROM Code Protect Function

The ROM code protect function is the function to inhibit reading out or modifying the contents of internal flash memory by using the ROM code protect control register (address FFC9$_{16}$) in parallel I/O mode. Figure 94 shows the ROM code protect control register (address FFC9$_{16}$). (This address exists in the User ROM area.)

If one or both of the pair of ROM Code Protect Bits is set to “0”, the ROM code protect is turned on, so that the contents of internal flash memory are protected against readout and modification. The ROM code protect is implemented in two levels. If level 2 is selected, the flash memory is protected even against readout by a shipment inspection LSI tester, etc. When an attempt is made to select both level 1 and level 2, level 2 is selected by default.

If both of the two ROM Code Protect Reset Bits are set to “00”, the ROM code protect is turned off, so that the contents of internal flash memory can be read out or modified. Once the ROM code protect is turned on, the contents of the ROM Code Protect Reset Bits cannot be modified in parallel I/O mode. Use the serial I/O or CPU rewrite mode to rewrite the contents of the ROM Code Protect Reset Bits.



Notes 1: This area is on the ROM in the mask ROM version.
2: When ROM code protect is turned on, the internal flash memory is protected against readout or modification in parallel I/O mode.
3: When ROM code protect level 2 is turned on, ROM code readout by a shipment inspection LSI tester, etc. also is inhibited.
4: The ROM code protect reset bits can be used to turn off ROM code protect level 1 and ROM code protect level 2. However, since these bits cannot be modified in parallel I/O mode, they need to be rewritten in standard serial I/O mode or CPU rewrite mode.

Fig. 94 Structure of ROM code protect control register

ID Code Check Function

Use this function in standard serial I/O mode. When the contents of the flash memory are not blank, the ID code sent from the programmer is compared with the ID code written in the flash memory to see if they match. If the ID codes do not match, the commands sent from the programmer are not accepted. The ID code consists of 8-bit data, and its areas are FFC2$_{16}$ to FFC8$_{16}$. Write a program which has had the ID code preset at these addresses to the flash memory.

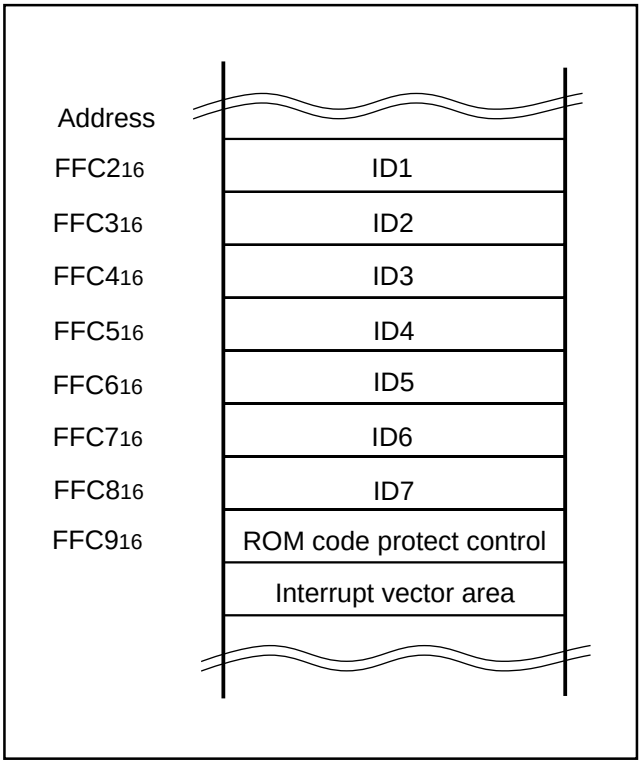


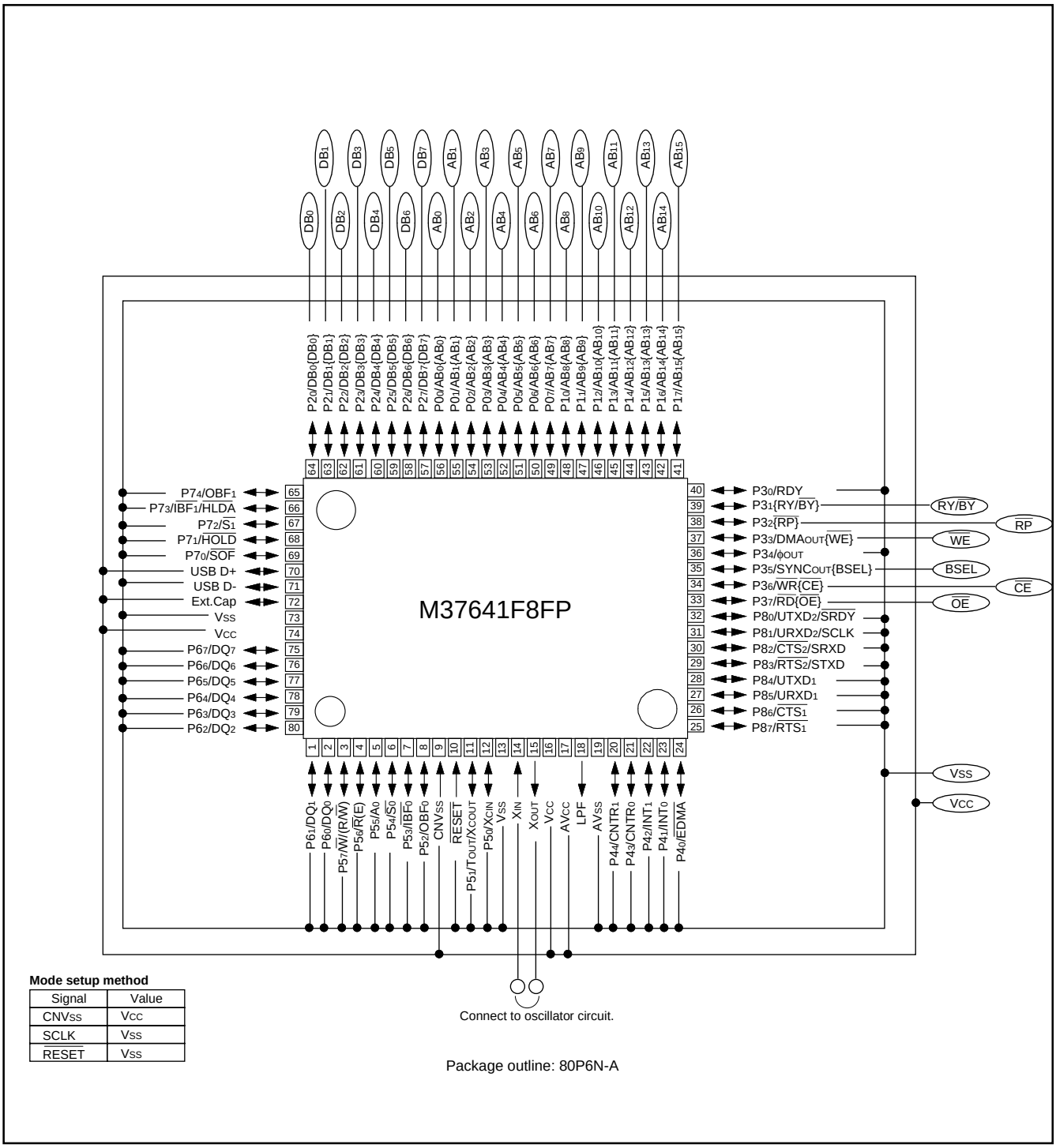
Fig. 95 ID code store addresses

(2) Parallel I/O Mode

The parallel I/O mode is entered by making connections shown in Figures 96 and 97 and then turning the Vcc power supply (5 V) on.

Table 28 Description of pin function (Flash Memory Parallel I/O Mode)

Pin name	Signal name	I/O	Function
V_{CC},V_{SS}	Power supply input		Apply 4.50 V – 5.25 V to the Vcc pin and 0 V to the Vss pin.
CNV_{SS}	CNV_{SS}		Connect this pin to V_{CC}.
$\overline{RESET}$	Reset input	I	Input “L” level.
X_{IN}	Clock input		Connect a ceramic or crystal resonator between the X_{IN} and X_{OUT} pins. When inputting an externally derived clock, input it from X_{IN} and leave X_{OUT} open.
X_{OUT}	Clock output		
AV_{CC}, AV_{SS}	Analog power supply input		Apply 4.50 V – 5.25 V to the AVcc pin and 0 V to the AVss pin.
LPF	LPF	O	Leave this pin open.
Ext.Cap	3.3 V line power supply input	I	Input “H” level.
USB D+	USB D+	I	Input “H” level.
USB D-	USB D-	I	Input “L” level.
$P0_0$ to $P0_7$	Address input	I	These are address AB_0–AB_7 input pins.
$P1_0$ to $P1_7$	Address input	I	These are address AB_8–AB_{15} input pins.
$P2_0$ to $P2_7$	Data I/O	I/O	These are data DB_0–DB_7 input/output pins.
$P3_0$	Input port $P3_0$	I	Input “L” level.
$P3_1$	RY/$\overline{BY}$ output	O	This is a RY/$\overline{BY}$ output pin.
$P3_2$	$\overline{RP}$ input	I	This is a $\overline{RP}$ input pin.
$P3_3$	$\overline{WE}$ input	I	This is a $\overline{WE}$ input pin.
$P3_4$	Input port $P3_4$	I	Input “L” level.
$P3_5$	BSEL input	I	This is a BSEL input pin.
$P3_6$	$\overline{CE}$ input	I	This is a $\overline{CE}$ input pin.
$P3_7$	$\overline{OE}$ input	I	This is a $\overline{OE}$ input pin.
$P4_0$ to $P4_4$	Input port P4	I	Input “L” level.
$P5_0$ to $P5_7$	Input port P5	I	Input “L” level.
$P6_0$ to $P6_7$	Input port P6	I	Input “L” level.
$P7_0$ to $P7_4$	Input port P7	I	Input “L” level.
$P8_0$ to $P8_7$	Input port P8	I	Input “L” level.

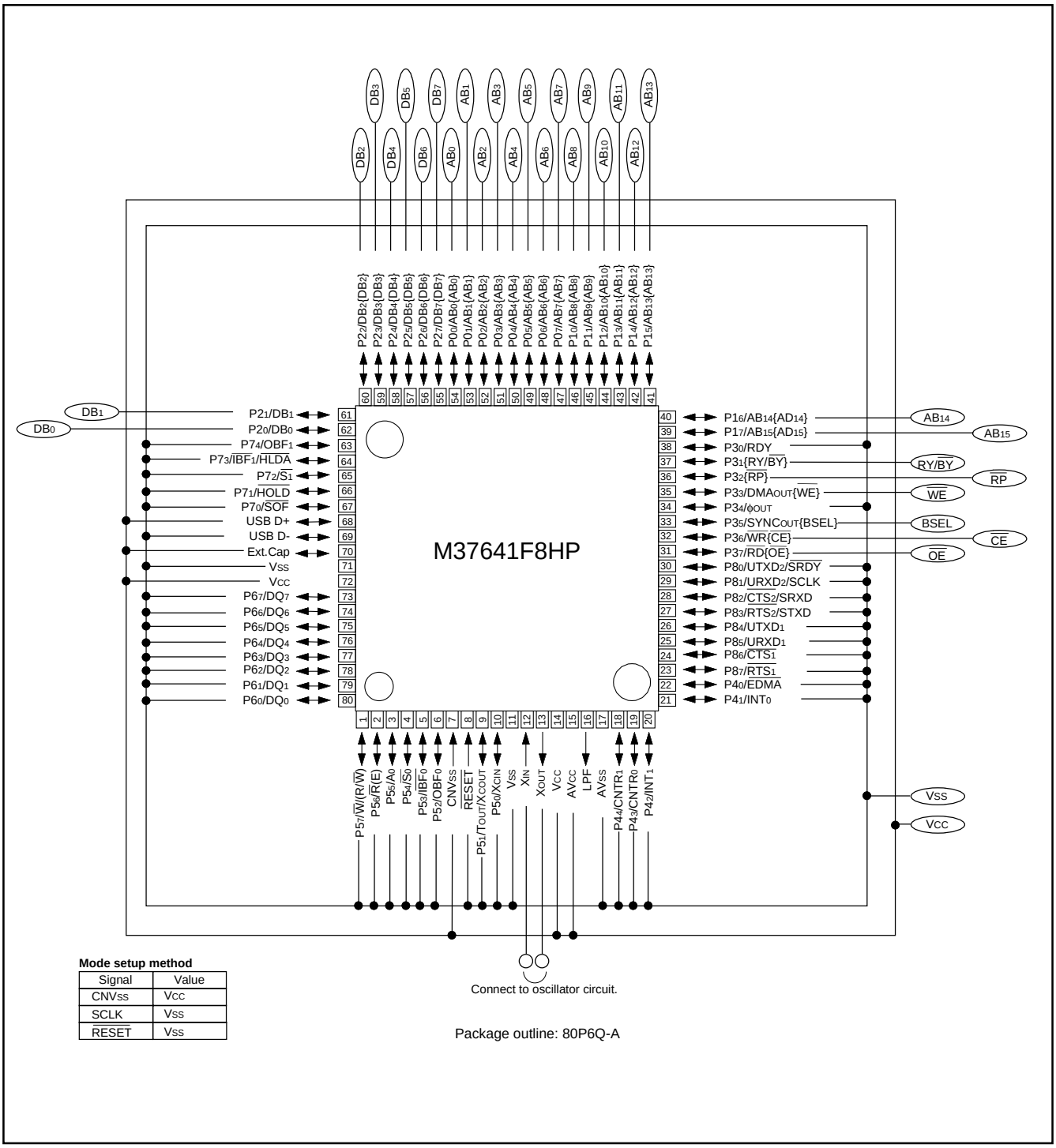


Mode setup method

Signal	Value
CNVSS	VCC
SCLK	VSS
RESET	VSS

Package outline: 80P6N-A

Fig. 96 Pin connection diagram in parallel I/O mode (1)



Mode setup method

Signal	Value
CNVSS	VCC
SCLK	VSS
RESET	VSS

Package outline: 80P6Q-A

Fig. 97 Pin connection diagram in parallel I/O mode (2)

Address

The User ROM has some blocks as shown in Figure 98. The block address is the maximum address value of each block.

User ROM and Boot ROM Areas

In parallel I/O mode, the User ROM and Boot ROM areas shown in Figure 98 can be rewritten. The BSEL pin is used to choose between these two areas. The User ROM area is selected by pulling the BSEL input low; the Boot ROM area is selected by driving the BSEL input high. Both areas of flash memory can be operated on in the same way.

Program and block erase operations can be performed only in the User ROM area.

The User ROM area is 32 Kbytes in size. In parallel I/O mode, it is located at addresses 8000_{16} through $FFFF_{16}$. The Boot ROM area is 4 Kbytes in size. In parallel I/O mode, it is located at addresses $F000_{16}$ through $FFFF_{16}$. Make sure program and block erase operations are always performed within this address range. (Access to any location outside this address range is prohibited.)

In the Boot ROM area, an erase block operation is applied to only one 4 Kbyte block. The boot ROM area has had a standard serial I/O mode control program stored in it when shipped from the Mitsubishi factory. Therefore, using the MCU in standard serial I/O mode, do not rewrite to the Boot ROM area.

Outline Performance (Parallel I/O Mode)

In parallel I/O mode, bus operation modes—Read, Output Disable, Standby, Write, and Deep Power Down— are selected by the state of the $\overline{CE}$, $\overline{OE}$, $\overline{WE}$, and $\overline{RP}$ input pins.

The contents of erase, program, and other operations are selected by writing a software command. The data, status register, etc. in a memory can only be read out by a read after software command input.

Program and erase operations are controlled using software commands.

The following explains about bus operation modes, software commands, and status register.

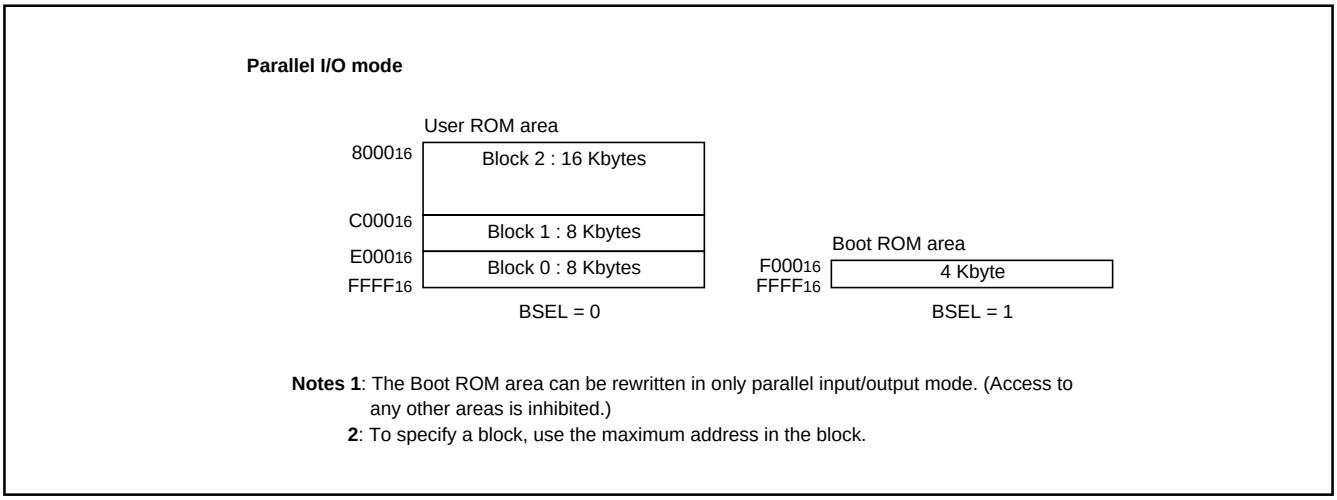


Fig. 98 Internal flash memory block diagram

Table 29 Relationship between control signals and bus operation modes

Mode \ Pin name		$\overline{CE}$	$\overline{OE}$	$\overline{WE}$	$\overline{RP}$	DB_0 to DB_7
Read	Array	V_{IL}	V_{IL}	V_{IH}	V_{IH}	Data output
	Status register	V_{IL}	V_{IL}	V_{IH}	V_{IH}	Status register data output
Output disabled		V_{IL}	V_{IH}	V_{IH}	V_{IH}	High impedance
Stand by		V_{IH}	X	X	V_{IH}	High impedance
Write	Program	V_{IL}	V_{IH}	V_{IL}	V_{IH}	Command/data input
	Erase all blocks	V_{IL}	V_{IH}	V_{IL}	V_{IH}	Command input
	Block erase	V_{IL}	V_{IH}	V_{IL}	V_{IH}	Command input
Deep power down		X	X	X	V_{IL}	High impedance

Note: X can be V_{IL} or V_{IH}.

Bus Operation Modes

●Read

The Read mode is entered by pulling the $\overline{OE}$ pin low when the $\overline{CE}$ pin is low and the $\overline{WE}$ and $\overline{RP}$ pins are high. There are two read modes: array and status register, which are selected by software command input. In read mode, the data corresponding to each software command entered is output from the data I/O pins DB_0–DB_7. The read array mode is automatically selected when the MCU is powered on or after releasing deep power down mode.

●Output Disable

The output disable mode is entered by pulling the $\overline{CE}$ pin low and the $\overline{WE}$, $\overline{OE}$, and $\overline{RP}$ pins high. Also, the data I/O pins are placed in the high-impedance state.

●Standby

The standby mode is entered by driving the $\overline{CE}$ pin high when the $\overline{RP}$ pin is high. Also, the data I/O pins are placed in the high-impedance state. However, if the $\overline{CE}$ pin is set high during erase or program operation, the internal control circuit does not halt immediately and normal power consumption is required until the operation under way is completed.

●Write

The write mode is entered by pulling the $\overline{WE}$ pin low when the $\overline{CE}$ pin is low and the $\overline{OE}$ and $\overline{RP}$ pins are high. In this mode, the MCU accepts the software commands or write data entered from the data I/O pins. A program, erase, or some other operation is initiated depending on the contents of the software command entered here. The input data such as address and software command is latched at the rising edge of $\overline{WE}$ or $\overline{CE}$ whichever occurs earlier.

●Deep Power Down

The deep power down mode is entered by pulling the $\overline{RP}$ pin low. Also, the data I/O pins are placed in the high-impedance state. When the MCU is freed from deep power down mode, the read array mode is selected and the contents of the status register is set to "80_{16}". If the $\overline{RP}$ pin is pulled low during erase or program operation, the operation under way is canceled and the data in the relevant block becomes invalid.

Software Commands

Table 30 lists the software commands. By inputting a software command from the data I/O pins (DB_0–DB_7) in write mode, specify the contents of the operation, such as erase or program operation, to be performed.

The following explains the contents of each software command.

●Read Array Command (FF_{16})

The read array mode is entered by writing the command code "FF_{16}" in the first bus cycle. When an address to be read is input in one of the bus cycles that follow, the contents of the specified address are output from the data I/O pins (DB_0–DB_7).

The read array mode is retained intact until another command is written.

The read array mode is also selected automatically when the MCU is powered on and after releasing deep power down mode.

●Read Status Register Command (70_{16})

When the command code "70_{16}" is written in the first bus cycle, the contents of the status register are output from the data I/O pins (DB_0–DB_7) by a read in the second bus cycle. Since the contents of the status register are updated at the falling edge of $\overline{OE}$ or $\overline{CE}$, the $\overline{OE}$ or $\overline{CE}$ signal must be input each time the status is read. The status register is explained in the next section.

●Clear Status Register Command (50_{16})

This command is used to clear the bits SR4, SR5 of the status register after they have been set. These bits indicate that operation has ended in an error. To use this command, write the command code "50_{16}" in the first bus cycle.

●Program Command (40_{16})

The program mode is entered when the command code "40_{16}" is written in the first bus cycle. Then, if the address and data to program are written in the second bus cycle, program operation (data programming and verification) will start.

Whether the write operation is completed can be confirmed by reading the status register or the RY/$\overline{BY}$ signal status. When the program starts, the read status register mode is entered automatically and the contents of the status register can be read out from the data I/O pins (DB_0 to D_{7B}). The status register bit 7 (SR7) is set to "0" at the same time the write operation starts and is returned to "1" upon completion of the write operation. In this case, the read status register mode remains active until the next command is written.2

The RY/$\overline{BY}$ pin is "L" during write operation and "H" when the write operation is completed as is the status register bit 7 (SR7).

At program end, program results can be checked by reading the status register.

Figure 99 shows the byte program flowchart.

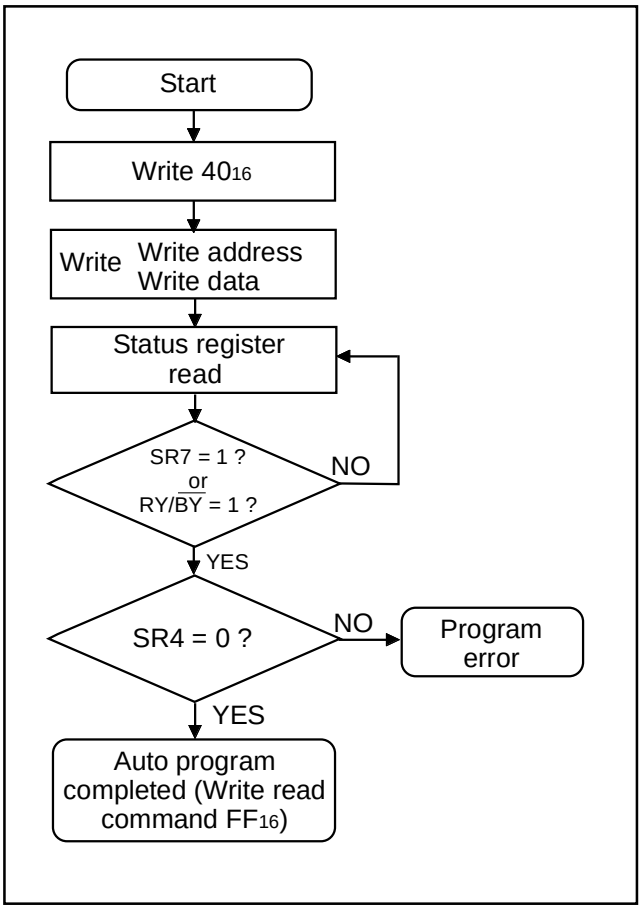


Fig. 99 Byte program flowchart

Table 30 Software command list (parallel I/O mode)

Command	Cycle number	First bus cycle			Second bus cycle		
		Mode	Address	Data (DB0 to DB7)	Mode	Address	Data (DB0 to DB7)
Read array	1	Write	X(Note 4)	FF16			
Read status register	2	Write	X	7016	Read	X	SRD (Note 1)
Clear status register	1	Write	X	5016			
Byte Program	2	Write	X	4016	Write	WA (Note 2)	WD (Note 2)
All block erase	2	Write	X	2016	Write	X	2016
Block erase	2	Write	X	2016	Write	BA (Note 3)	D016

Notes 1: SRD = Status Register Data
2: WA = Write Address, WD = Write Data
3: BA = Block Address to be erased (Input the maximum address of each block)
4: X denotes a given address in the User ROM area or Boot ROM area.

●Erase All Blocks Command (20_{16}/20_{16})

By writing the command code "20_{16}" in the first bus cycle and the confirmation command code "20_{16}" in the second bus cycle that follows, the erase all blocks (erase and erase verify) operation starts.

Whether the erase all blocks command is terminated can be confirmed by reading the status register or the RY/$\overline{BY}$ signal status .

When the erase all blocks operation starts, the read status register mode is entered automatically and the contents of the status register can be read out at the data I/O pins (DB_0 to DB_7). The status register bit 7 (SR7) is set to "0" at the same time the erase operation starts and is returned to "1" upon completion of the erase operation. In this case, the read status register mode remains active until another command is written.

The RY/$\overline{BY}$ pin is "L" during erase operation and "H" when the erase operation is completed as is the status register bit 7 (SR7).

At erase all blocks end, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed.

●Block Erase Command (20_{16}/$D0_{16}$)

By writing the command code "20_{16}" in the first bus cycle and the confirmation command code "$D0_{16}$" in the second bus cycle that follows to the block address of a flash memory block, the block erase (erase and erase verify) operation starts.

Whether the block erase operation is completed can be confirmed by reading the status register or the RY/$\overline{BY}$ pin state. When the block erase operation starts, the read status register mode is entered automatically and the contents of the status register can be read out at the data I/O pins (DB_0 to DB_7). The status register bit 7 (SR7) is set to "0" at the same time the block erase operation starts and is returned to "1" upon completion of the block erase operation. In this case, the read status register mode remains active until the read array command (FF_{16}) is written.

The RY/$\overline{BY}$ pin is "L" during block erase operation and "H" when the block erase operation is completed as is the status register bit 7 (SR7).

At block erase operation end, erase results can be checked by reading the status register. For details, refer to the section where the status register is detailed.

Figure 100 shows the erase flowchart.

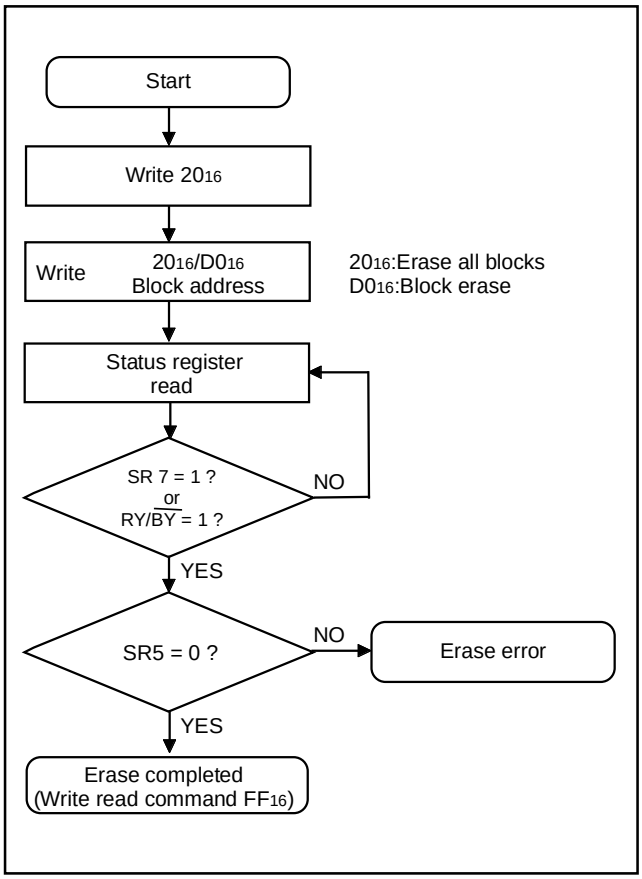


Fig. 100 Erase flowchart

Status Register

The status register indicates status such as whether an erase operation or a program ended successfully or in error. It can be read under the following conditions.

(1) In the read array mode when the read status register command (70_{16}) is written and the block address is subsequently read.
(2) When reading the User ROM area, in the period from when the program write or block erase starts to when the read array command (FF_{16}) is written.

The status register is cleared in the following situations.

(1) By writing the clear status register command (50_{16})
(2) In the deep power down mode
(3) In the power supply off state

Table 31 lists the definition of each status register bit. When power is turned on or returning from the deep power down mode, the status register becomes "80_{16}".

●Sequencer Status (SR7)

The sequencer status indicates the operating status of the flash memory. This bit is set to "0" (busy) during write or erase operation and is set to "1" (ready) when these operations end.
After power-on, the sequencer status is set to "1" (ready).

●Erase Status (SR5)

The erase status indicates the operating status of erase operation. If an erase error occurs, it is set to "1". When the erase status is cleared, it is set to "0".

●Program Status (SR4)

The program status indicates the operating status of write operation. When a write error occurs, it is set to "1".
The program status is set to "0" when it is cleared.

Table 31 Status register

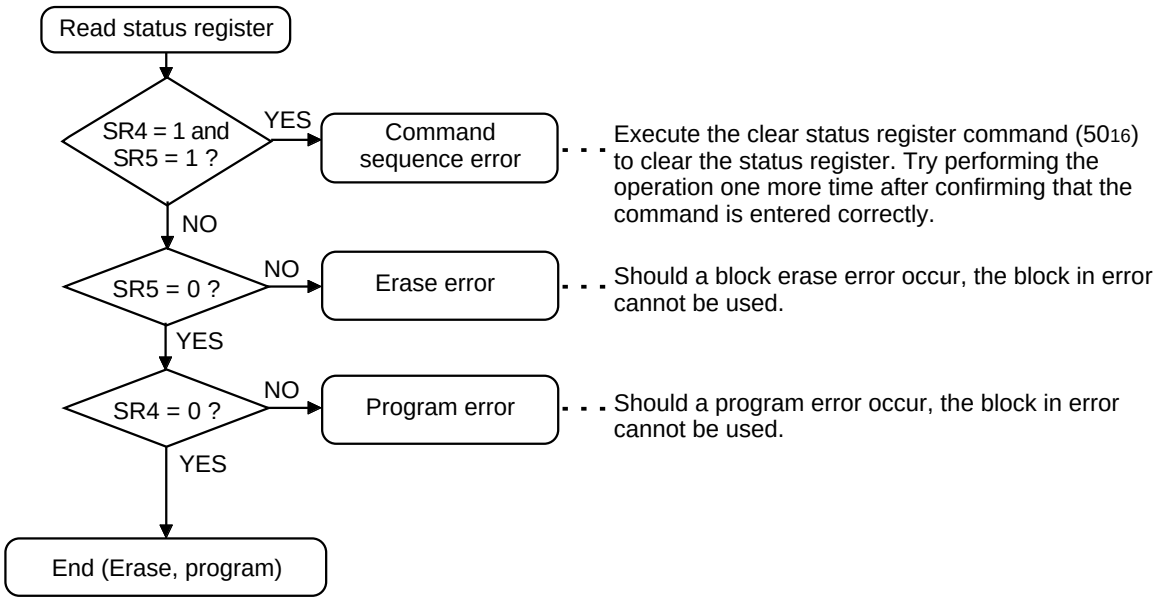
Each bit of SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (DB_7)	Sequencer status	Ready	Busy
SR6 (DB_6)	Reserved	-	-
SR5 (DB_5)	Erase status	Ended in error	Ended successfully
SR4 (DB_4)	Program status	Ended in error	Ended successfully
SR3 (DB_3)	Reserved	-	-
SR2 (DB_2)	Reserved	-	-
SR1 (DB_1)	Reserved	-	-
SR0 (DB_0)	Reserved	-	-

Full Status Check

By performing full status check, it is possible to know the execution results of erase and program operations. Figure 101 shows a full status check flowchart and the action to be taken when each error occurs.

Ready/Busy (RY/$\overline{BY}$) Pin

The RY/$\overline{BY}$ pin is an output pin (N-channel open drain output) which, such as the sequencer status (SR7), indicates the operating status of the flash memory. It is "L" level (busy) during program or erase operation and becomes the high impedance state (ready) when these operations end. The RY/$\overline{BY}$ pin requires an external pull-up operation.



Note: When one of SR5 and SR4 is set to "1", none of the program, erase all blocks, and block erase commands is accepted. Execute the clear status register command (5016) before executing these commands.

Fig. 101 Full status check flowchart and remedial procedure for errors

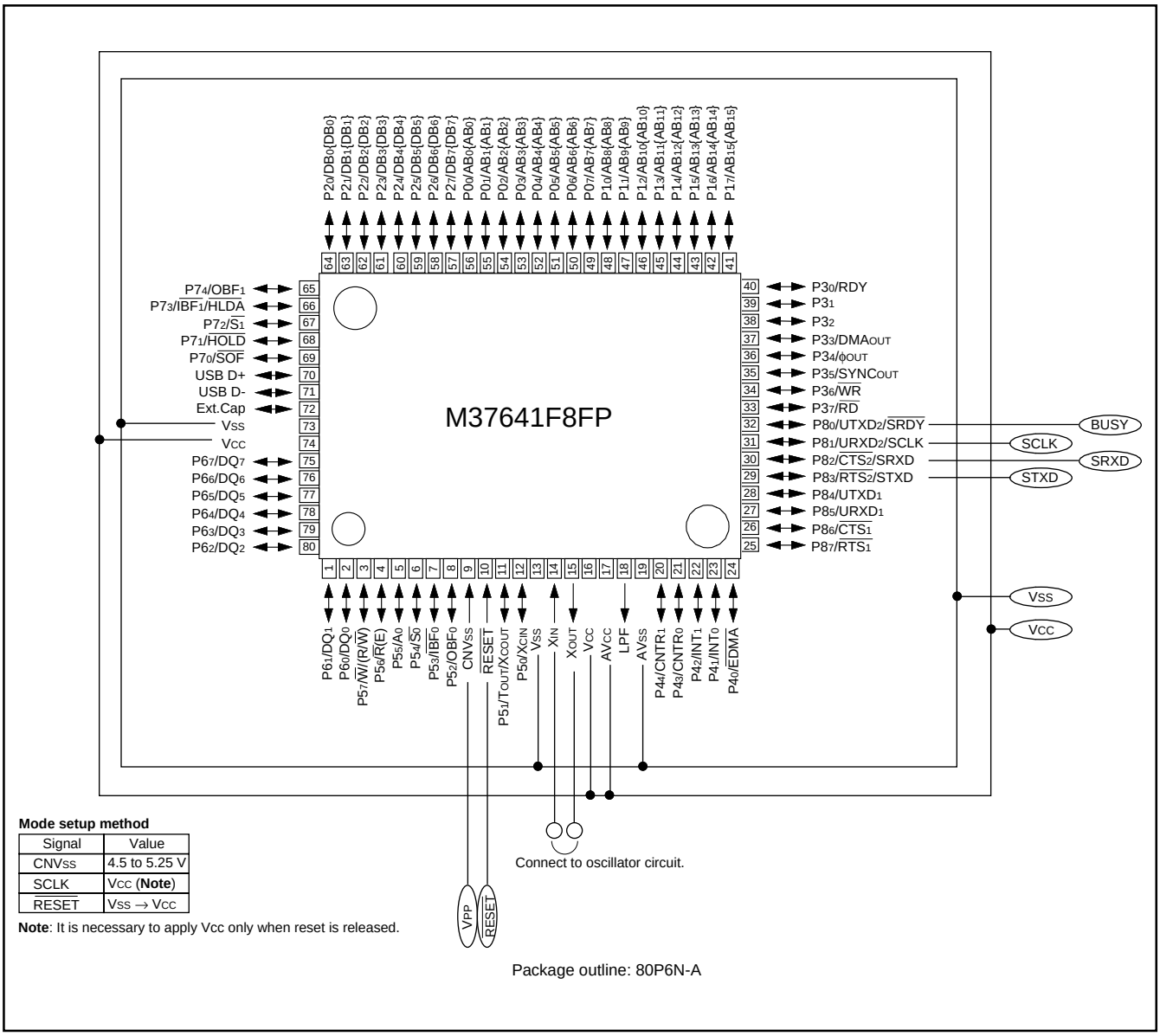
(3) Standard serial I/O Mode

The standard serial I/O mode inputs and outputs the software commands, addresses and data needed to operate (read, program, erase, etc.) the internal flash memory. This I/O is clock synchronized serial. This mode requires a purpose-specific peripheral unit.

Table 32 Description of pin function (Flash Memory Standard Serial I/O Mode)

Pin name	Signal name	I/O	Function
VCC,VSS	Power supply input		Apply 4.15 V – 5.25 V for 5 V version or 3.00 V – 3.60 V for 3 V version to the VCC pin. Apply 0 V to the Vss pin.
CNVSS	CNVSS	I	This controls the MCU operating mode. Connect this pin to VPP (= 4.50 V – 5.25 V
$\overline{\text{RESET}}$	Reset input	I	To reset, input “L” level for 20 cycles or longer clocks of ϕ.
XIN	Clock input		Connect a ceramic or crystal resonator between the XIN and XOUT pins. When inputting an externally derived clock, input it from XIN and leave XOUT open.
XOUT	Clock output		
AVCC, AVSS	Analog power supply input		Apply 4.15 V – 5.25 V for 5 V version or 3.00 V – 3.60 V for 3 V version to the AVCC pin. Apply 0 V to the AVss pin.
LPF	LPF	O	Loop filter for the frequency synthesizer.
Ext.Cap	3.3 V line power supply input	I	Power supply input pin for 3.3 V USB line driver.
USB D+	USB D+	I/O	USB D+ signal port.
USB D-	USB D-	I/O	USB D- signal port.
P00 to P07	I/O port P0	I/O	When these ports are not used, input “L” or “H” level, or leave them open in output mode.
P10 to P17	I/O port P1	I/O	
P20 to P27	I/O port P2	I/O	
P30 to P37	I/O port P3	I/O	
P40 to P44	I/O port P4	I/O	
P50 to P57	I/O port P5	I/O	
P60 to P67	I/O port P6	I/O	
P70 to P74	I/O port P7	I/O	
P80	BUSY output	O	This is a BUSY output pin.
P81	SCLK input	I	This is a serial clock input pin.
P82	SRXD input	I	This is a serial data input pin.
P83	STXDoutput	O	This is a serial data output pin.
P84 to P87	I/O port P8	I/O	When these ports are not used, input “L” or “H” level, or leave them open in output mode.

Note: The unused pins LPF, Ext.Cap, USB D+/USB D- require the same treatment as that of in the parallel I/O mode.



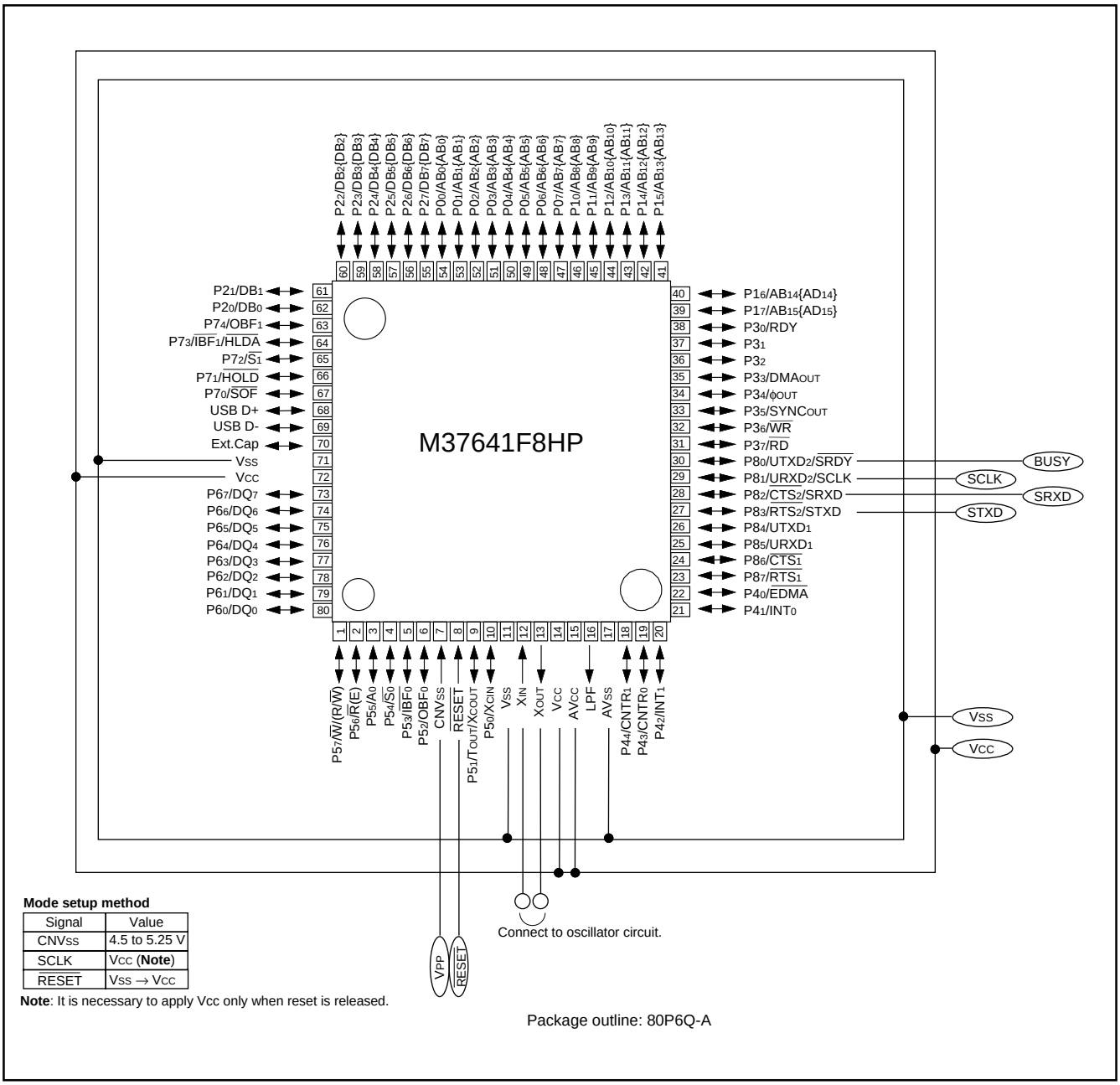
Mode setup method

Signal	Value
CNVSS	4.5 to 5.25 V
SCLK	VCC (**Note**)
RESET	VSS → VCC

Note: It is necessary to apply Vcc only when reset is released.

Package outline: 80P6N-A

Fig. 102 Pin connection diagram in standard serial I/O mode (1)



Mode setup method

Signal	Value
CNVSS	4.5 to 5.25 V
SCLK	VCC (**Note**)
$\overline{\text{RESET}}$	VSS → VCC

Note: It is necessary to apply Vcc only when reset is released.

Package outline: 80P6Q-A

Fig. 103 Pin connection diagram in standard serial I/O mode (2)

The standard serial I/O mode is different from the parallel I/O mode in that the CPU controls flash memory rewrite (uses the CPU rewrite mode), rewrite data input and so forth. The standard serial I/O mode is started by connecting "H" to the P81(SCLK) pin and "H" to the CNVSS pin (apply 4.5 V to 5.25 V to Vpp from an external source), and releasing the reset operation. (In the ordinary microcomputer mode, set CNVss pin to "L" level.)

This control program is written in the Boot ROM area when the product is shipped from Mitsubishi. Accordingly, make note of the fact that the standard serial I/O mode cannot be used if the Boot ROM area is rewritten in parallel I/O mode. Figures 102 and 103 show the pin connections for the standard serial I/O mode.

In standard serial I/O mode, serial data I/O uses the four serial I/O pins SCLK, SRXD, STXD and $\overline{\text{SRDY}}$ (BUSY). The SCLK pin is the transfer clock input pin through which an external transfer clock is input. The STXD pin is for CMOS output. The $\overline{\text{SRDY}}$ (BUSY) pin outputs "L" level when ready for reception and "H" level when reception starts.

Serial data I/O is transferred serially in 8-bit units.

In standard serial I/O mode, only the User ROM area shown in Figure 104 can be rewritten. The Boot ROM area cannot.

In standard serial I/O mode, a 7-byte ID code is used. When there is data in the flash memory, commands sent from the peripheral unit (programmer) are not accepted unless the ID code matches.

Outline Performance (Standard Serial I/O Mode)

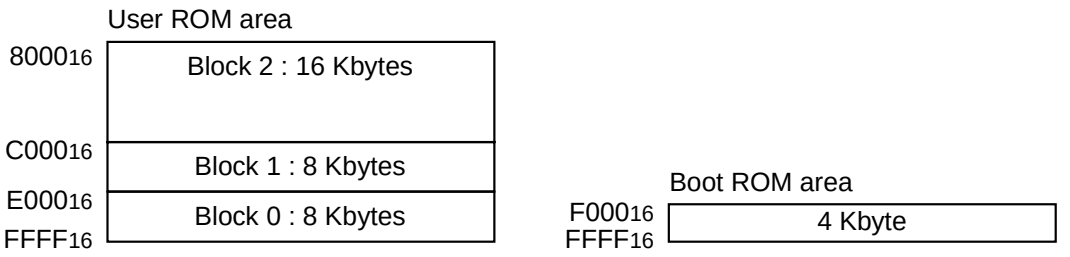
In standard serial I/O mode, software commands, addresses and data are input and output between the MCU and peripheral units (serial programer, etc.) using 4-wire clock-synchronized serial I/O. In reception, software commands, addresses and program data are synchronized with the rise of the transfer clock that is input to the SCLK pin, and are then input to the MCU via the SRXD pin. In transmission, the read data and status are synchronized with the fall of the transfer clock, and output from the STXD pin.

The STXD pin is for CMOS output. Transfer is in 8-bit units with LSB first.

When busy, such as during transmission, reception, erasing or program execution, the $\overline{\text{SRDY}}$ (BUSY) pin is "H" level. Accordingly, always start the next transfer after the $\overline{\text{SRDY}}$ (BUSY) pin is "L" level.

Also, data and status registers in a memory can be read after inputting software commands. Status, such as the operating state of the flash memory or whether a program or erase operation ended successfully or not, can be checked by reading the status register. Here following explains software commands, status registers, etc.

CPU rewrite mode, standard serial I/O mode



Notes 1: The Boot ROM area can be rewritten in only parallel I/O mode. (Access to any other areas is inhibited.)
2: To specify a block, use the maximum address in the block.

Fig. 104 Block digram of on-chip flash memory

Software Commands

Table 33 lists software commands. In standard serial I/O mode, erase, program and read are controlled by transferring software commands via the SRXD pin. Software commands are explained here below. Basically, the software commands of the standard serial I/O mode are as same as that of the parallel I/O mode, but 4 commands are added: ID check, download, version data output and Boot ROM area output functions.

Table 33 Software commands (Standard serial I/O mode)

	Control command	1st byte transfer	2nd byte	3rd byte	4th byte	5th byte	6th byte	.....	When ID is not verified
1	Page read	FF_{16}	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable
2	Page program	41_{16}	Address (middle)	Address (high)	Data input	Data input	Data input	Data input to 259th byte	Not acceptable
3	Block erase	20_{16}	Address (middle)	Address (high)	$D0_{16}$				Not acceptable
4	Erase all blocks	$A7_{16}$	$D0_{16}$						Not acceptable
5	Read status register	70_{16}	SRD output	SRD1 output					Acceptable
6	Clear status register	50_{16}							Not acceptable
7	ID check	$F5_{16}$	Address (low)	Address (middle)	Address (high)	ID size	ID1	To ID7	Acceptable
8	Download function	FA_{16}	Size (low)	Size (high)	Check-sum	Data input	To required number of times		Not acceptable
9	Version data output function	FB_{16}	Version data output	Version data output	Version data output	Version data output	Version data output	Version data output to 9th byte	Acceptable
10	Boot ROM area output function	FC_{16}	Address (middle)	Address (high)	Data output	Data output	Data output	Data output to 259th byte	Not acceptable

Notes1: Shading indicates transfer from the internal flash memory microcomputer to a programmer. All other data is transferred from a programmer to the internal flash memory microcomputer.
2: SRD refers to status register data. SRD1 refers to status register 1 data.
3: All commands can be accepted for the products of which boot ROM area is totally blank.
4: Address low is AB_0 to AB_7; Address middle is AB_8 to AB_{15}; Address high is AB_{16} to AB_{23}.

●Page Read Command

This command reads the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page read command as explained here following.

(1) Transfer the "FF_{16}" command code with the 1st byte.
(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.
(3) From the 4th byte onward, data (DB_0 to DB_7) for the page (256 bytes) specified with addresses A_8 to A_{23} will be output sequentially from the smallest address first synchronized with the fall of the clock.

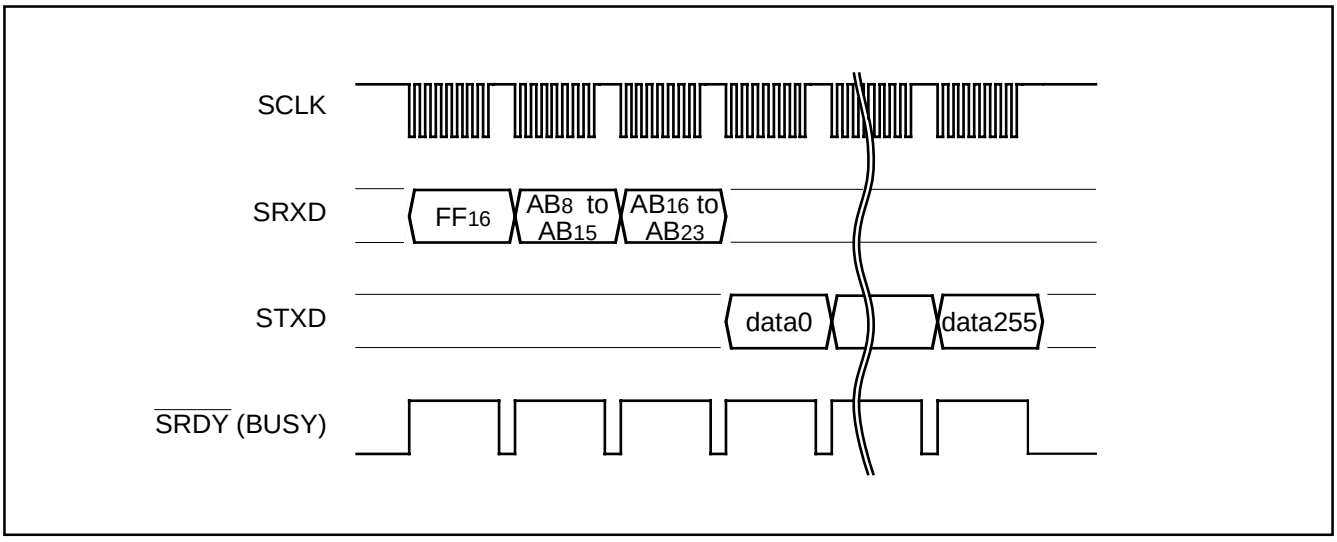


Fig. 105 Timing for page read

●Read Status Register Command

This command reads status information. When the "70_{16}" command code is transferred with the 1st byte, the contents of the status register (SRD) with the 2nd byte and the contents of status register 1 (SRD1) with the 3rd byte are read.

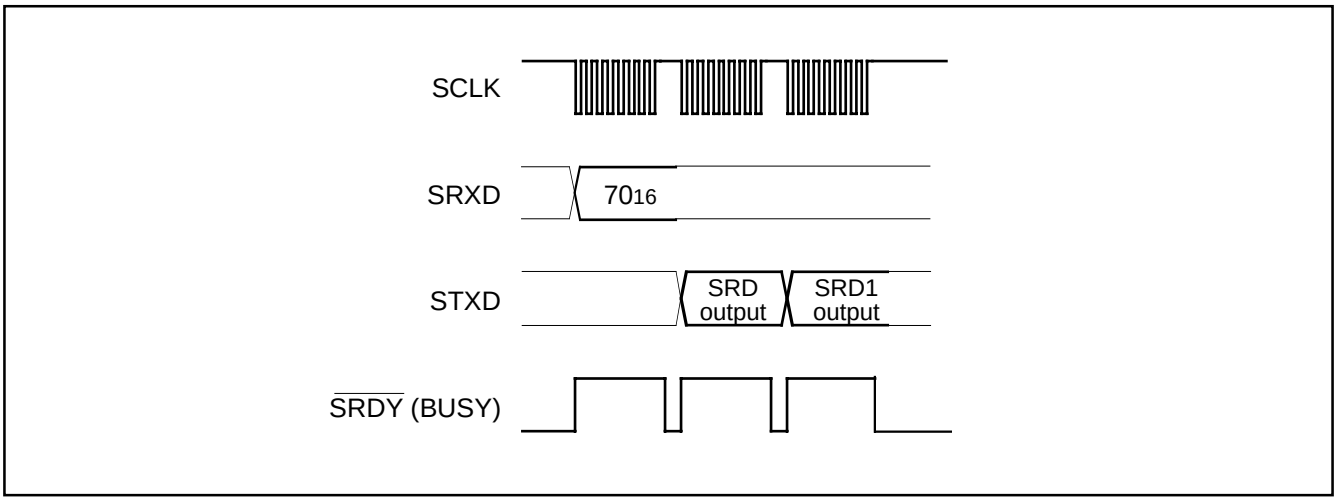


Fig. 106 Timing for reading status register

●Clear Status Register Command

This command clears the bits (SR3 to SR5) which are set when the status register operation ends in error. When the "50_{16}" command code is sent with the 1st byte, the aforementioned bits are cleared. When the clear status register operation ends, the $\overline{\text{SRDY}}$ (BUSY) signal changes from "H" to "L" level.

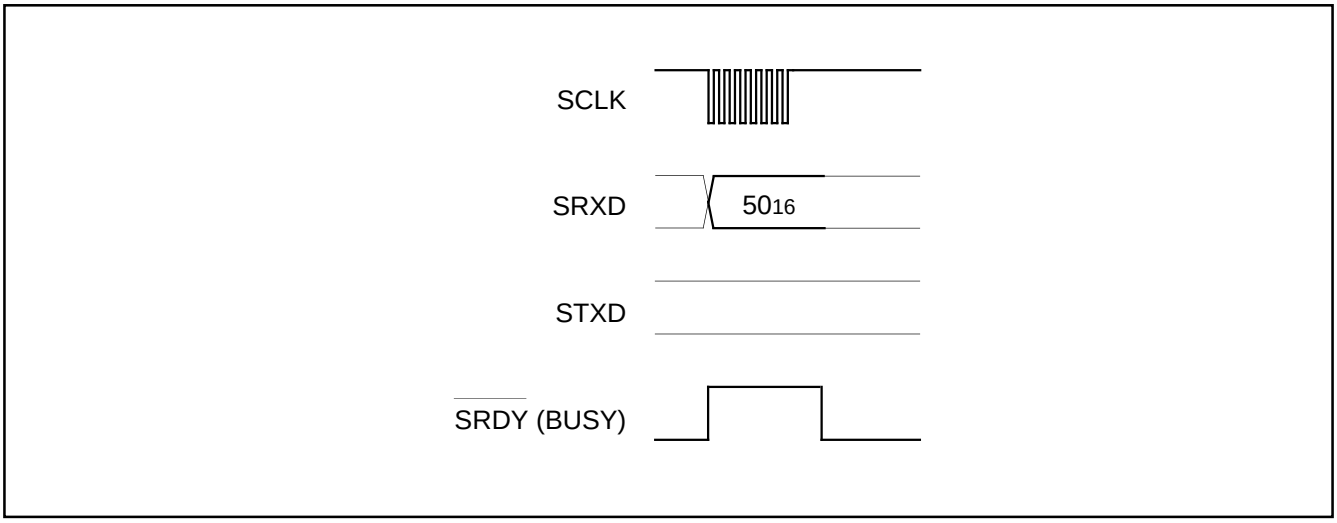


Fig. 107 Timing for clear status register

●Page Program Command

This command writes the specified page (256 bytes) in the flash memory sequentially one byte at a time. Execute the page program command as explained here following.

(1) Transfer the "41_{16}" command code with the 1st byte.

(2) Transfer addresses AB_8 to AB_{15} and AB_{16} to AB_{23} with the 2nd and 3rd bytes respectively.

(3) From the 4th byte onward, as write data (DB_0 to DB_7) for the page (256 bytes) specified with addresses A_8 to A_{23} is input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the $\overline{\text{SRDY}}$ (BUSY) signal changes from "H" to "L" level. The result of the page program can be known by reading the status register. For more information, see the section on the status register.

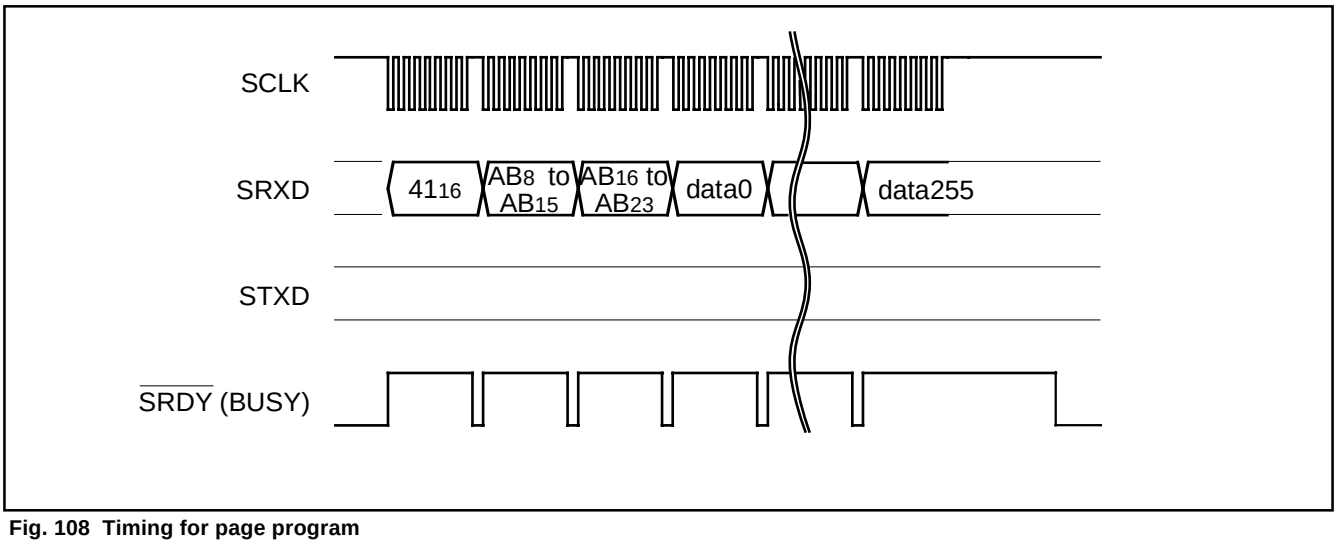


Fig. 108 Timing for page program

●Block Erase Command

This command erases the contents of the specified block. Execute the block erase command as explained here following.

(1) Transfer the "20_{16}" command code with the 1st byte.
(2) Transfer addresses AB_8 to AB_{15} and AB_{16} to AB_{23} with the 2nd and 3rd bytes respectively.
(3) Transfer the verify command code "$D0_{16}$" with the 4th byte. With the verify command code, the erase operation will start for the specified block in the flash memory. Set the addresses AB_8 to AB_{23} to the maximum address of the specified block.

When block erasing ends, the $\overline{SRDY}$ (BUSY) signal changes from "H" to "L" level. The result of the erase operation can be known by reading the status register.
For more information, see the section on the status register.

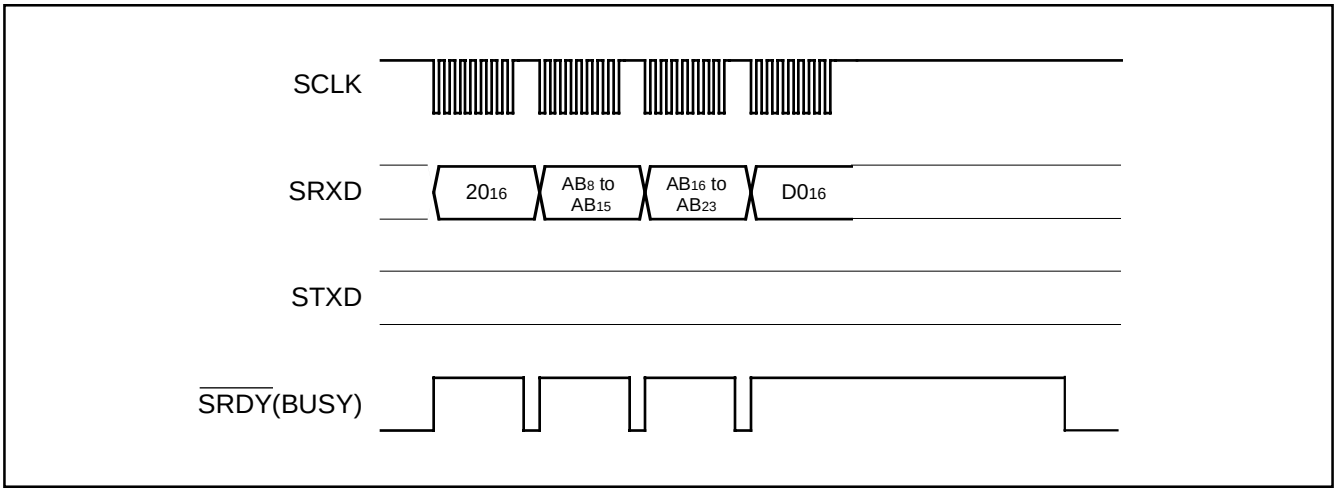


Fig. 109 Timing for block erasing

●Erase All Blocks Command

This command erases the contents of all blocks. Execute the erase all blocks command as explained here following.

(1) Transfer the "$A7_{16}$" command code with the 1st byte.
(2) Transfer the verify command code "$D0_{16}$" with the 2nd byte. With the verify command code, the erase operation will start and continue for all blocks in the flash memory.

When erase all blocks end, the $\overline{SRDY}$ (BUSY) signal changes from "H" to "L" level. The result of the erase operation can be known by reading the status register.

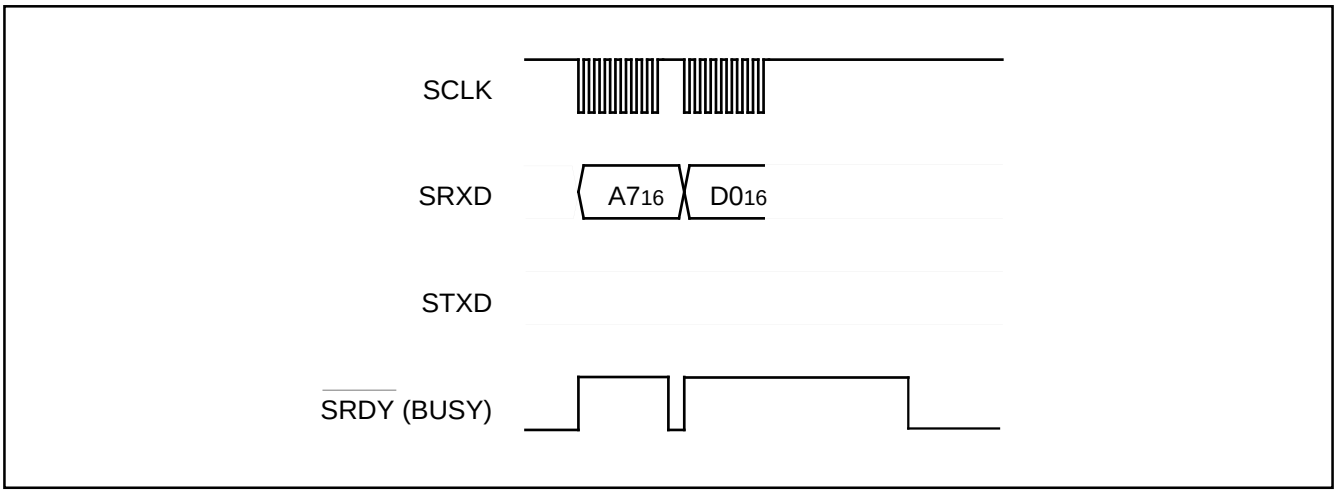


Fig. 110 Timing for erase all blocks

●Download Command

This command downloads a program to the RAM for execution. Execute the download command as explained here following.

(1) Transfer the "FA16" command code with the 1st byte.
(2) Transfer the program size with the 2nd and 3rd bytes.
(3) Transfer the check sum with the 4th byte. The check sum is added to all data sent with the 5th byte onward.
(4) The program to execute is sent with the 5th byte onward.

When all data has been transmitted, if the check sum matches, the downloaded program is executed. The size of the program will vary according to the internal RAM.

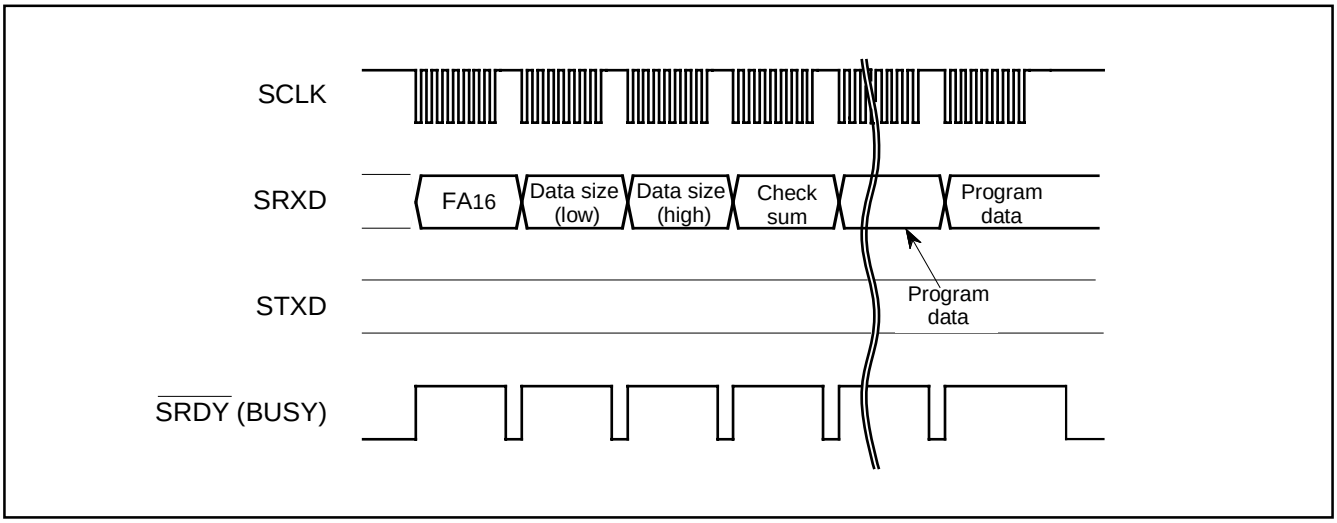


Fig. 111 Timing for download

●Version Information Output Command

This command outputs the version information of the control program stored in the Boot ROM area. Execute the version information output command as explained here following.

(1) Transfer the "FB_{16}" command code with the 1st byte.
(2) The version information will be output from the 2nd byte onward.

This data is composed of 8 ASCII code characters.

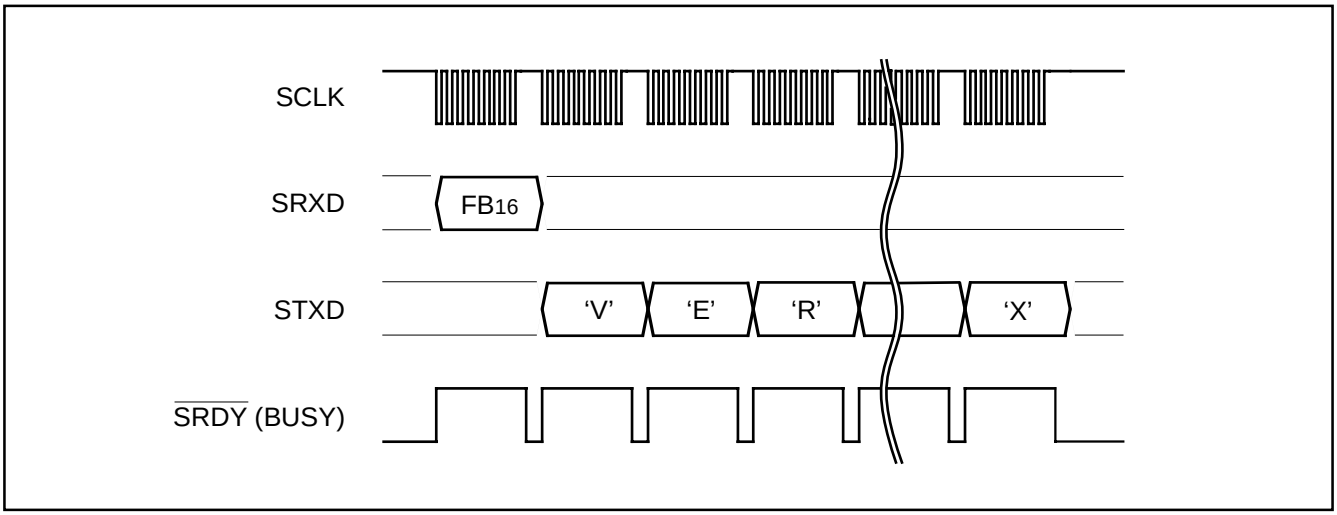


Fig. 112 Timing for version information output

●Boot ROM Area Output Command

This command reads the control program stored in the Boot ROM area in page (256 bytes) unit. Execute the Boot ROM area output command as explained here following.

(1) Transfer the "FC_{16}" command code with the 1st byte.
(2) Transfer addresses AB_8 to AB_{15} and AB_{16} to AB_{23} with the 2nd and 3rd bytes respectively.
(3) From the 4th byte onward, data (DB_0 to DB_7) for the page (256 bytes) specified with addresses AB_8 to AB_{23} will be output sequentially from the smallest address first synchronized with the fall of the clock.

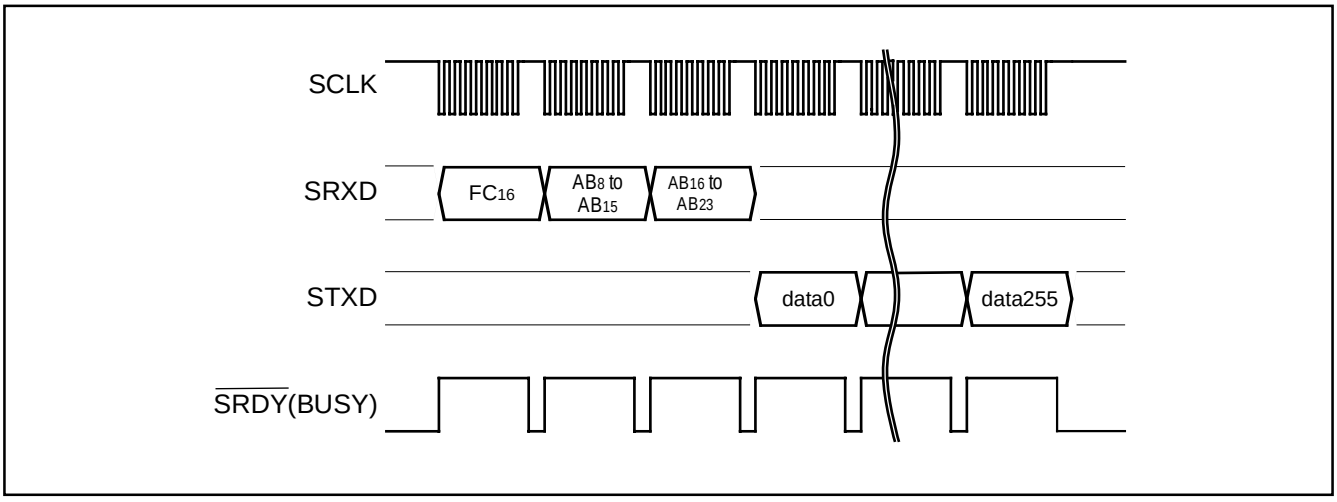


Fig. 113 Timing for Boot ROM area output

●ID Check

This command checks the ID code. Execute the boot ID check command as explained here following.

(1) Transfer the "F5$_{16}$" command code with the 1st byte.
(2) Transfer addresses AB$_0$ to AB$_7$, AB$_8$ to AB$_{15}$ and AB$_{16}$ to AB$_{23}$ ("00$_{16}$") of the 1st byte of the ID code with the 2nd and 3rd respectively.
(3) Transfer the number of data sets of the ID code with the 5th byte.
(4) Transfer the ID code with the 6th byte onward, starting with the 1st byte of the code.

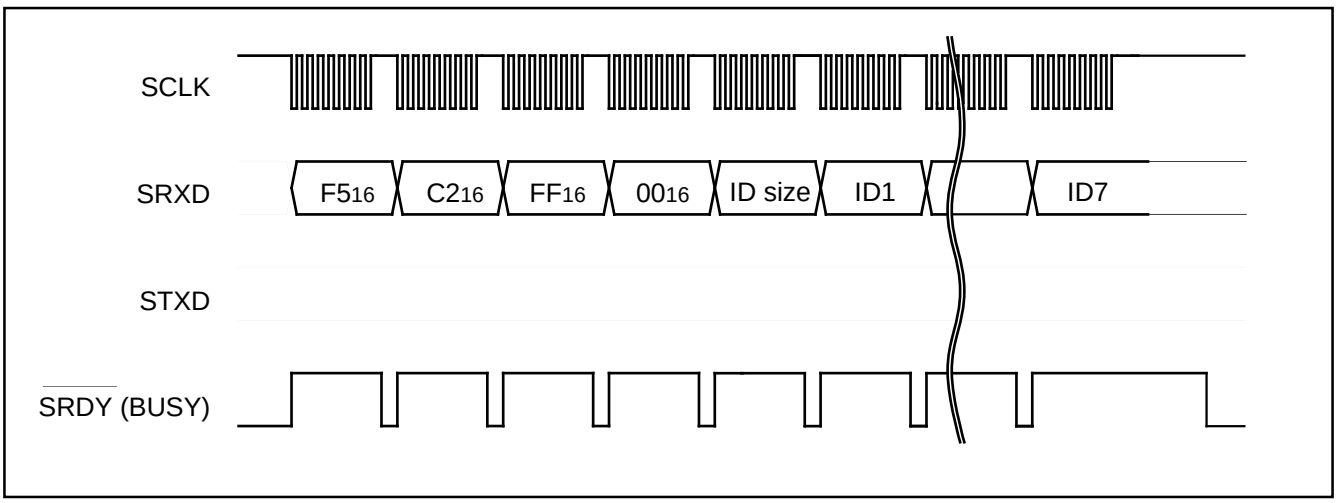


Fig. 114 Timing for ID check

●ID Code

When the flash memory is not blank, the ID code sent from the serial programmer and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the serial programmer is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses FFC2$_{16}$ to FFC9$_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.

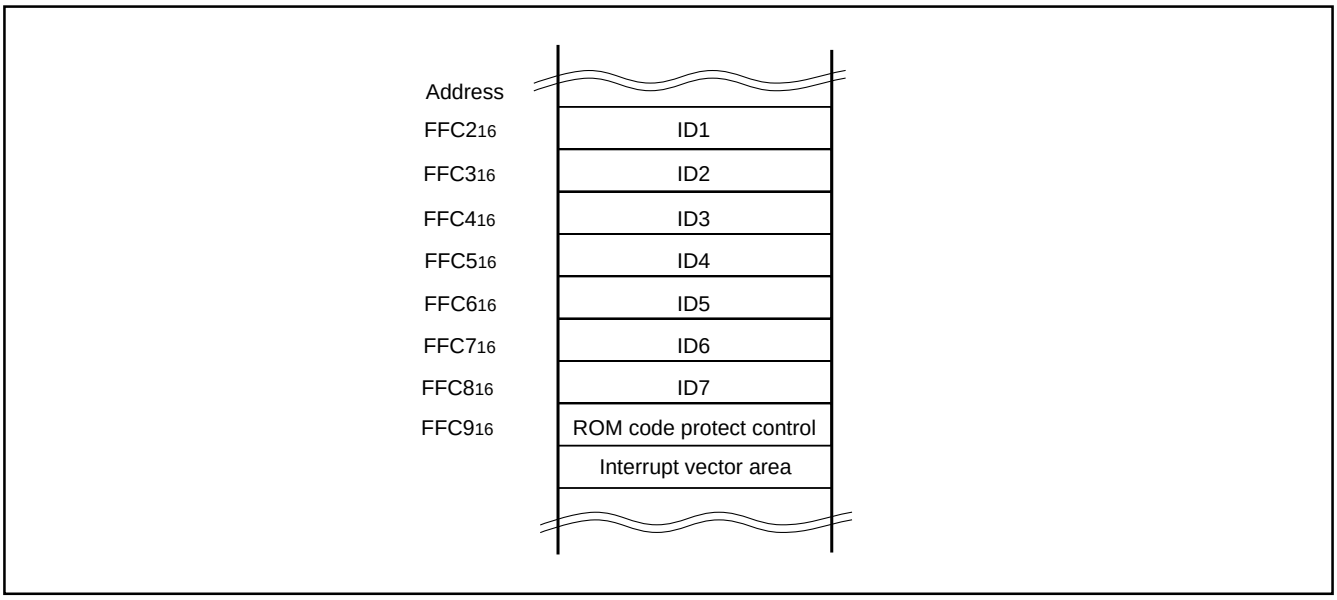


Fig. 115 ID code storage addresses

●Status Register (SRD)

The status register indicates operating status of the flash memory and status such as whether an erase operation or a program ended successfully or in error. It can be read by writing the read status register command (70_{16}). Also, the status register is cleared by writing the clear status register command (50_{16}).

Table 34 lists the definition of each status register bit. After releasing the reset, the status register becomes "80_{16}".

•Sequencer status (SR7)

The sequencer status indicates the operating status of the the flash memory.

After power-on and recover from deep power down mode, the sequencer status is set to "1" (ready).

This status bit is set to "0" (busy) during write or erase operation and is set to "1" upon completion of these operations.

•Erase status (SR5)

The erase status indicates the operating status of erase operation. If an erase error occurs, it is set to "1". When the erase status is cleared, it is set to "0".

•Program status (SR4)

The program status indicates the operating status of write operation. If a write error occurs, it is set to "1". When the program status is cleared, it is set to "0".

Table 34 Status register (SRD)

SRD0 bits	Status name	Definition	
		"1"	"0"
SR7 (bit7)	Sequencer status	Ready	Busy
SR6 (bit6)	Reserved	-	-
SR5 (bit5)	Erase status	Terminated in error	Terminated normally
SR4 (bit4)	Program status	Terminated in error	Terminated normally
SR3 (bit3)	Reserved	-	-
SR2 (bit2)	Reserved	-	-
SR1 (bit1)	Reserved	-	-
SR0 (bit0)	Reserved	-	-

●Status Register 1 (SRD1)

The status register 1 indicates the status of serial communications, results from ID checks and results from check sum comparisons. It can be read after the SRD by writing the read status register command (70_{16}). Also, status register 1 is cleared by writing the clear status register command (50_{16}).

Table 35 lists the definition of each status register 1 bit. This register becomes "00_{16}" when power is turned on and the flag status is maintained even after the reset.

•Boot update completed bit (SR15)

This flag indicates whether the control program was downloaded to the RAM or not, using the download function.

•Check sum consistency bit (SR12)

This flag indicates whether the check sum matches or not when a program, is downloaded for execution using the download function.

•ID check completed bits (SR11 and SR10)

These flags indicate the result of ID checks. Some commands cannot be accepted without an ID check.

•Data reception time out (SR9)

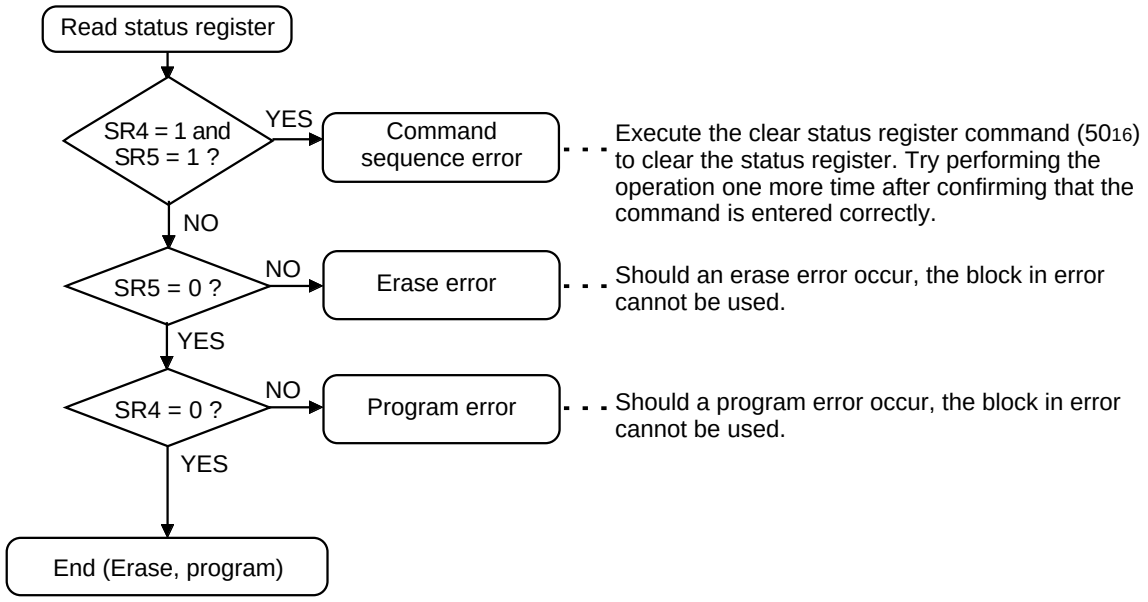
This flag indicates when a time out error is generated during data reception. If this flag is attached during data reception, the received data is discarded and the MCU returns to the command wait state.

Table 35 Status register 1 (SRD1)

SRD1 bits	Status name	Definition	
		"1"	"0"
SR15 (bit7)	Boot update completed bit	Update completed	Not Update
SR14 (bit6)	Reserved	-	-
SR13 (bit5)	Reserved	-	-
SR12 (bit4)	Checksum match bit	Match	Mismatch
SR11 (bit3) SR10 (bit2)	ID check completed bits	00 Not verified 01 Verification mismatch 10 Reserved 11 Verified	
SR9 (bit1)	Data reception time out	Time out	Normal operation
SR8 (bit0)	Reserved	-	-

Full Status Check

Results from executed erase and program operations can be known by running a full status check. Figure 116 shows a flowchart of the full status check and explains how to remedy errors which occur.

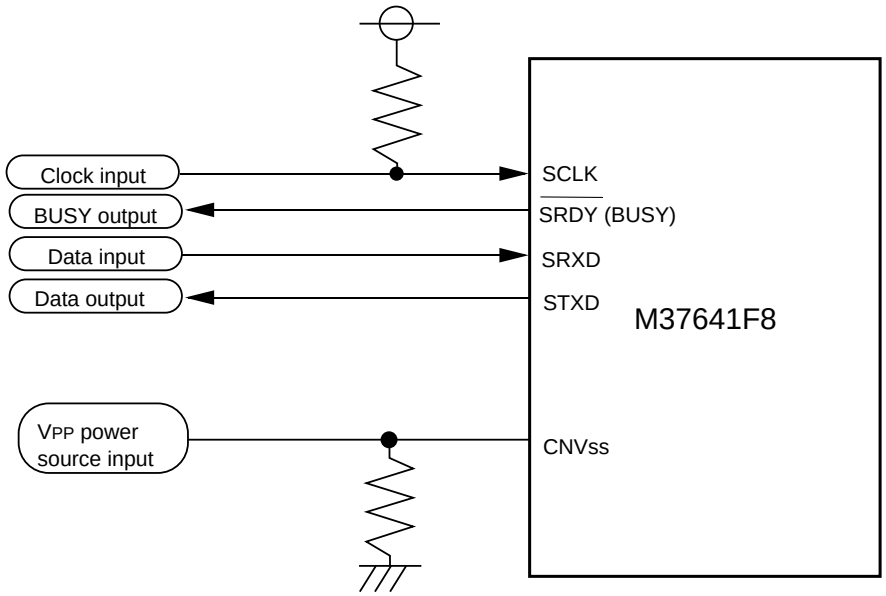


Note: When one of SR5 to SR4 is set to “1” , none of the program, erase all blocks, and block erase commands is accepted. Execute the clear status register command (50_{16}) before executing these commands.

Fig. 116 Full status check flowchart and remedial procedure for errors

Example Circuit Application for Standard Serial I/O Mode

Figure 117 shows a circuit application for the standard serial I/O mode. Control pins will vary according to a programmer, therefore see a programmer manual for more information.



Notes 1: Control pins and external circuitry will vary according to a programmer. For more information, see the programmer manual.

2: In this example, the Vpp power supply is supplied from an external source (programmer). To use the user's power source, connect to 4.5 V to 5.25 V.

Fig. 117 Example circuit application for standard serial I/O mode

Flash Memory Electrical Characteristics (Timing)

AC Electrical Characteristics (Timing)

(Ta = 20 to 70 °C, Vcc = 4.5 to 5.25 V, ϕ = 6 MHz unless otherwise noted)

Table 36 Read-only mode

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{RC}	Read cycle time	167			ns
$t_{a\ (AD)}$	Address access time			83.3	ns
$t_{a\ (CE)}$	$\overline{CE}$ access time			83.3	ns
$t_{a\ (OE)}$	$\overline{OE}$ access time			66.7	ns
t_{CLZ}	Output enable time (after $\overline{CE}$)	0			ns
$t_{DF(CE)}$	Output floating time (after $\overline{CE}$)			20.8	ns
t_{OLZ}	Output enable time (after $\overline{OE}$)	0			ns
$t_{DF(OE)}$	Output floating time (after $\overline{OE}$)			20.8	ns
t_{PHZ}	Output floating time (after $\overline{PR}$)			250	ns
t_{OH}	Output valid time (after $\overline{CE}$, $\overline{OE}$, address)	0			ns
t_{OEH}	Write recovery time (before read)	167			ns
t_{PS}	$\overline{RP}$ recovery time	10			µs

Note : Timing measurement condition is showed in Figure 118.

Table 37 Read / Write mode ($\overline{WE}$ control)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{WC}	Write cycle time	167			ns
t_{AS}	Address set up time	83.3			ns
t_{AH}	Address hold time	20.8			ns
t_{DS}	Data set up time	83.3			ns
t_{DH}	Data hold time	20.8			ns
t_{CS}	$\overline{CE}$ set up time	0			ns
t_{CH}	$\overline{CE}$ hold time	0			ns
t_{WP}	$\overline{WE}$ pulse width	83.3			ns
t_{WPH}	“H” write pulse width	41.7			ns
t_{GHWL}	Read recover time (before write)	167			ns
t_{DAP}	Program time		25	1000	µs
t_{DAE}	Erase all blocks time		1.5	20	s
t_{WHRL}	RY/$\overline{BY}$ delay time			167	ns
t_{PS}	$\overline{RP}$ recovery time	10			µs

Note : The read timing parameter in the command write operation mode is the same as that of the read-only mode.
Typical value is at Vcc = 5.0 V, Ta = 25 °C condition.

Table 38 Read / Write mode ($\overline{CE}$ control)

Symbol	Parameter	Limits			Unit
		Min.	Typ.	Max.	
t_{WC}	Write cycle time	167			ns
t_{AS}	Address set up time	83.3			ns
t_{AH}	Address hold time	20.8			ns
t_{DS}	Data set up time	83.3			ns
t_{DH}	Data hold time	20.8			ns
t_{CS}	$\overline{WE}$ set up time	0			ns
t_{CH}	$\overline{WE}$ hold time	0			ns
t_{WP}	$\overline{CE}$ pulse width	83.3			ns
t_{WPH}	"H" $\overline{CE}$ pulse width	41.7			ns
t_{DAP}	Program time		25	1000	µs
t_{DAE}	Erase all blocks time		1.5	20	s
t_{WHRL}	RY/$\overline{BY}$ delay time			167	ns
t_{PS}	$\overline{RP}$ recovery time	10			µs

Note : The read timing parameter in the command write operation mode is the same as that of the read-only mode.
Typical value is at V_{CC} = 5.0 V, Ta = 25 °C condition.

Table 39 Erase and program operation

Parameter	Min.	Typ.	Max.	Unit
Erase all blocks time		1.5		s
Block erase time		1.0		s
Program time (1byte)		25		µs

Table 40 V_{CC} power up / power down timing

Symbol	Parameter	Min.	Typ.	Max.	Unit
t_{VCS}	$\overline{RP}$ = V_{IH} set up time (after rised V_{CC} = V_{CC} min.)	10			µs

Note : Miserase or miswrite may happen, in case of noise pulse due to the power supply on or off is input to the control pins. Therefore disableing the write mode is need for preventing memory data from breaking at the power supply on or off. 10µs (min.) waiting time is need to initiate read or write operation after V_{CC} rises to V_{CC} min. at power supply on. The memory data is protected owing to keep the $\overline{RP}$ pin V_{IL} level at power supply off. The $\overline{RP}$ pin must be kept V_{IL} level for 10µs (min.) after V_{CC} rises to V_{CC} min. at the power supply on. The $\overline{RP}$ pin must be kept V_{IL} level until the V_{CC} falls to the GND level at power supply off. $\overline{RP}$ pin doesn't have latch mode, so that $\overline{RP}$ pin must be kept V_{IH} level during read, erase and program operation.

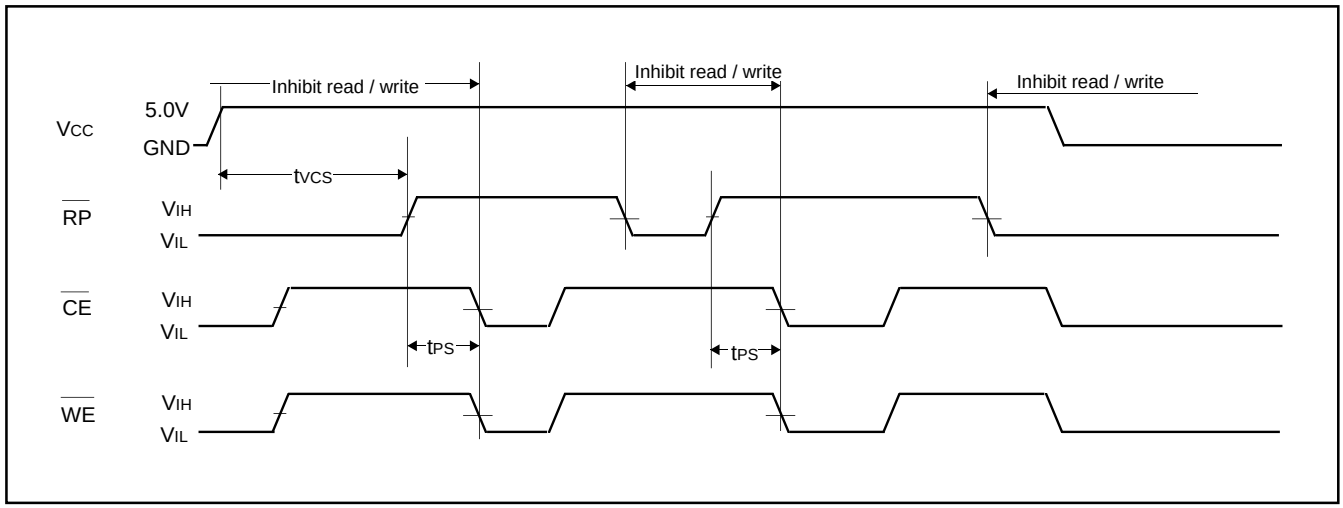


Fig. 118 Vcc power up / power down timing

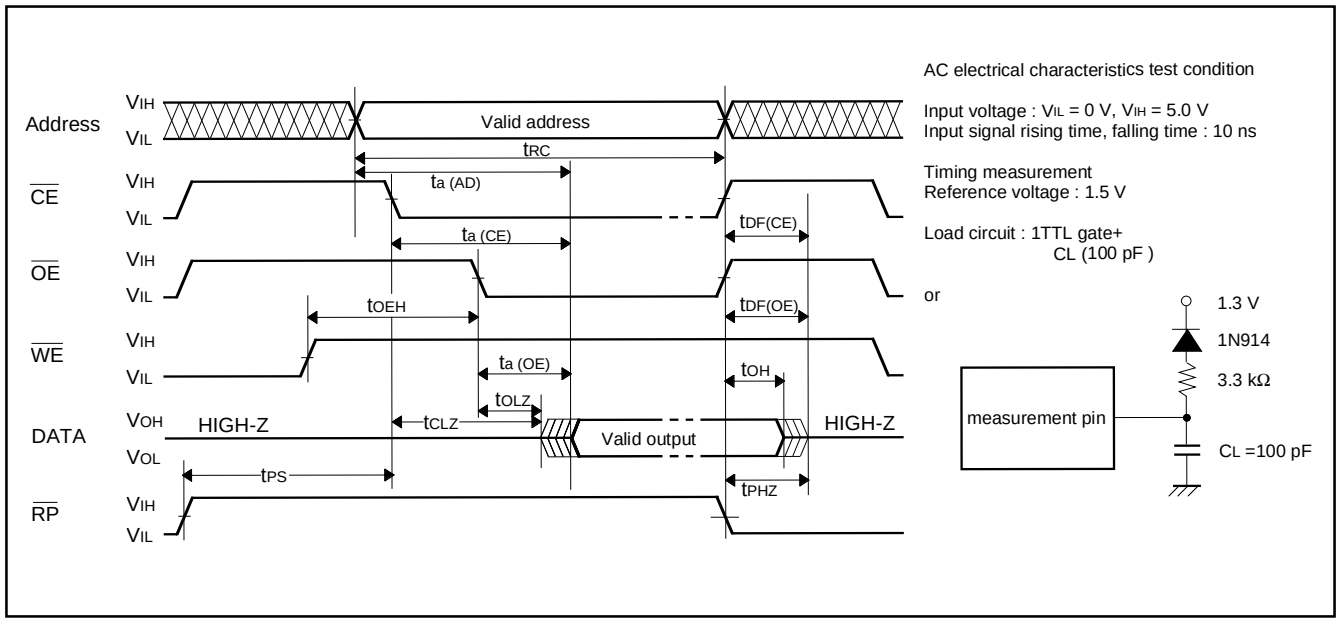


Fig. 119 AC wave for read operation and test condition

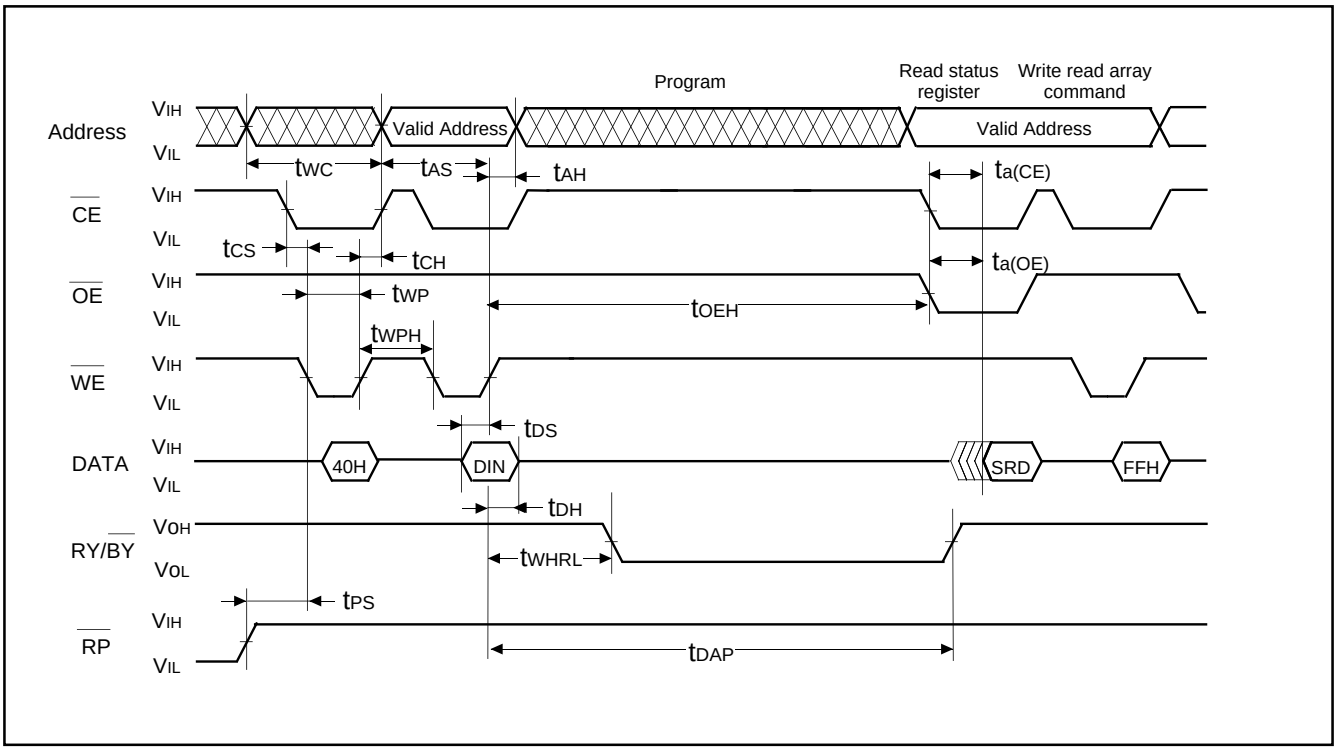


Fig. 120 AC wave for program operation ($\overline{WE}$ control)

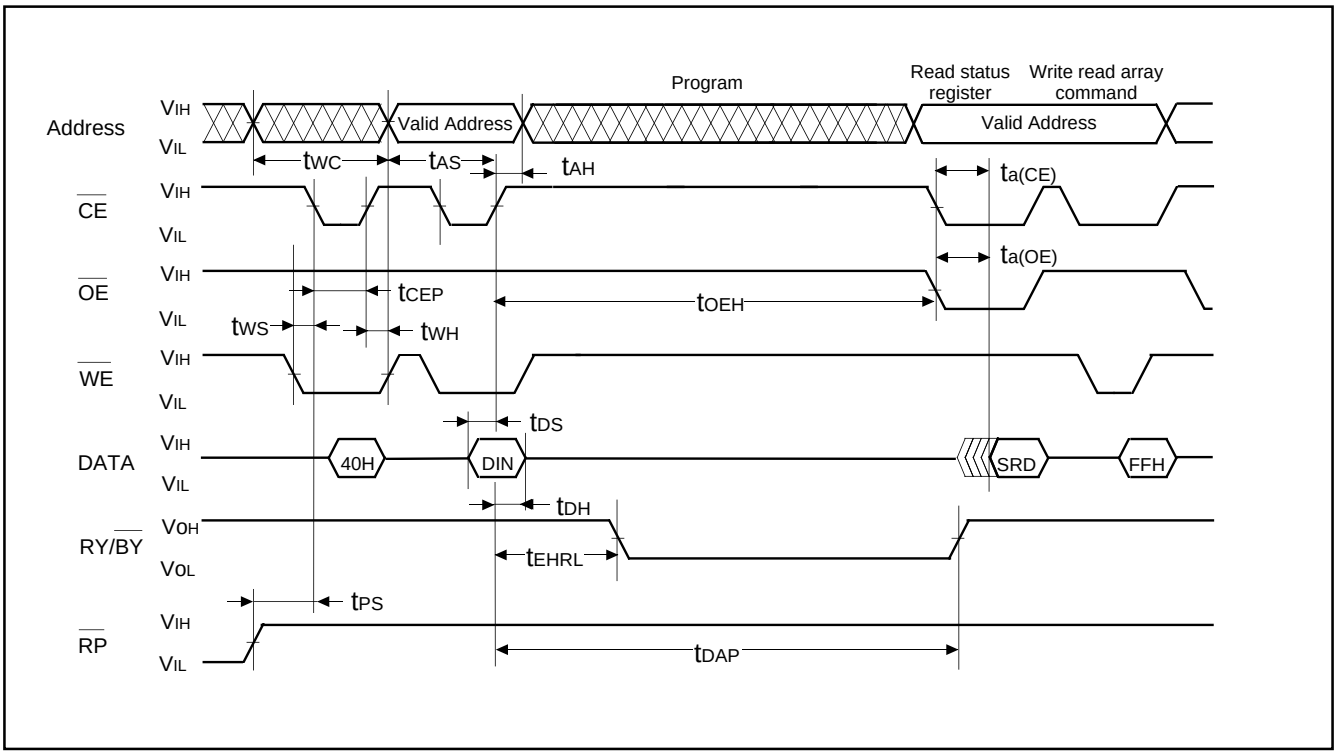


Fig.121 AC wave for program operation ($\overline{CE}$ control)

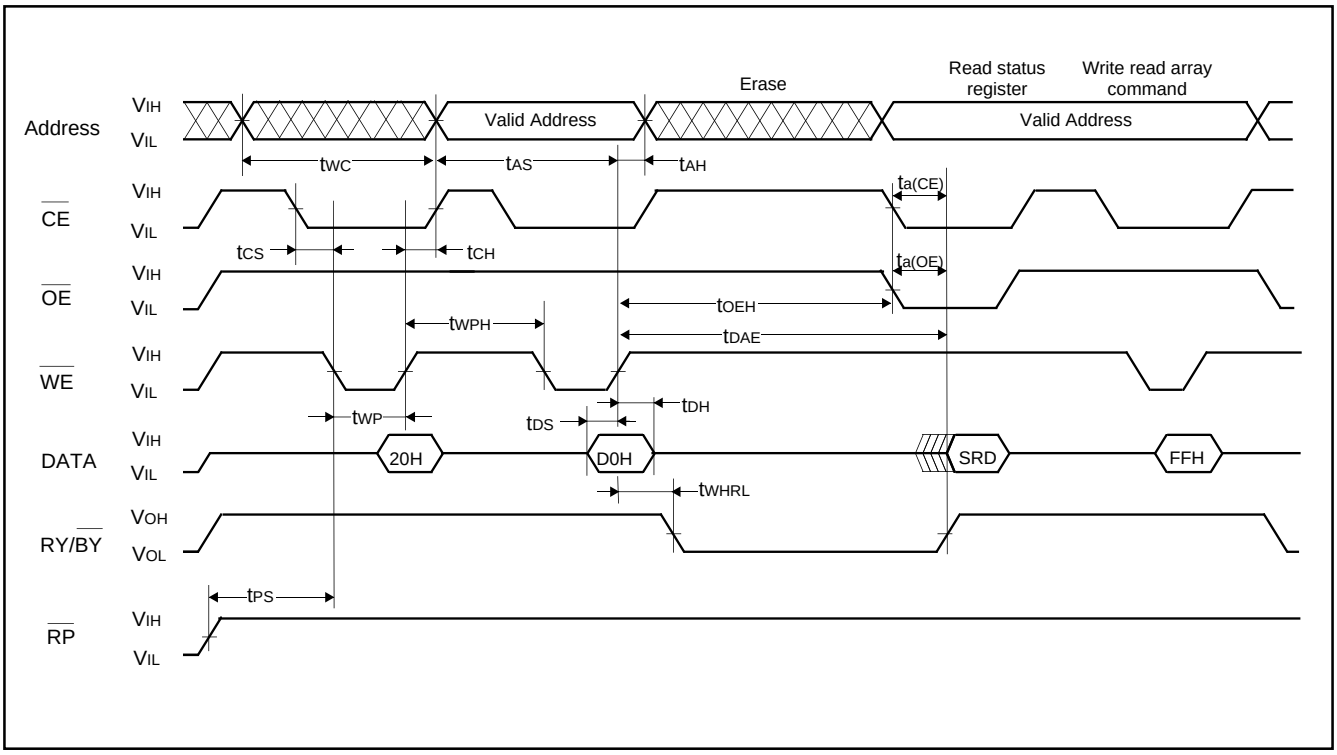


Fig. 122 AC wave for erase operation ($\overline{WE}$ control)

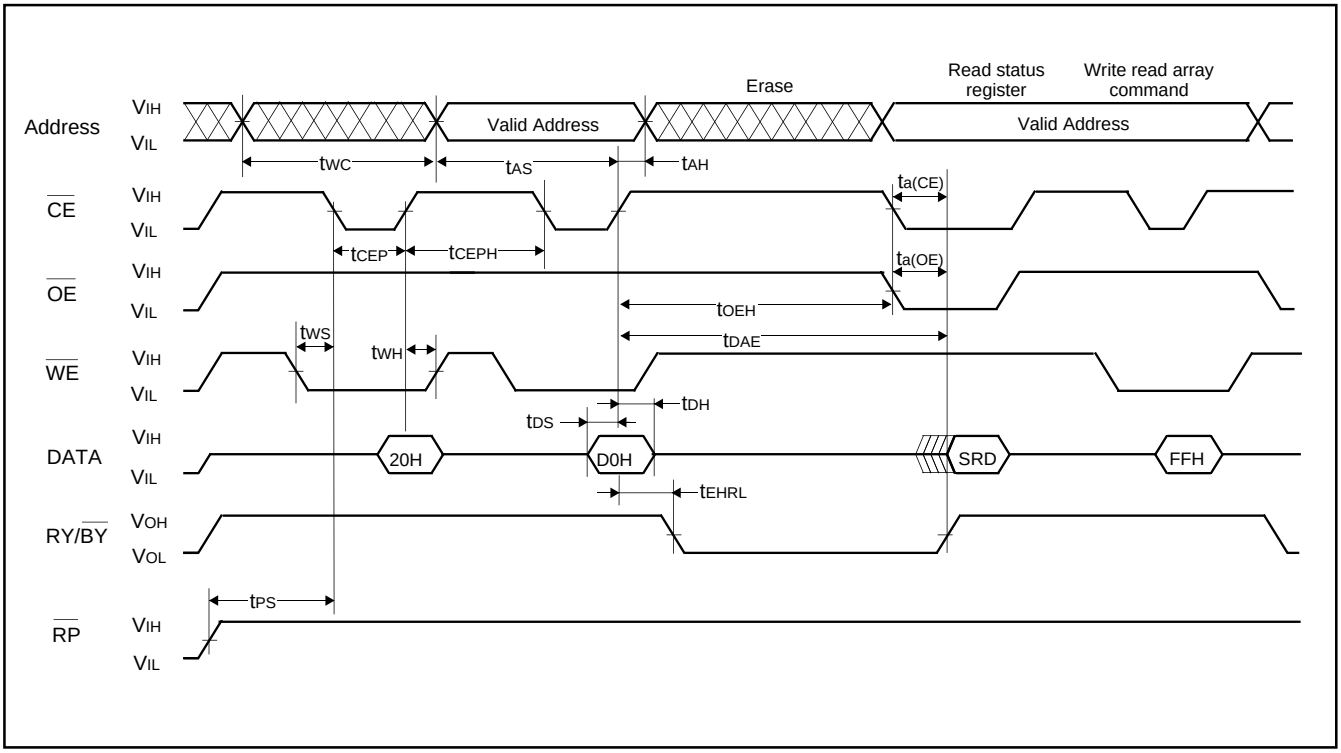


Fig. 123 AC wave for erase operation ($\overline{CE}$ control)

NOTES ON PROGRAMMING

Processor Status Register

•The contents of the processor status register (PS) after a reset are undefined, except for the interrupt disable flag (I) which is "1". After a reset, initialize flags which affect program execution. In particular, it is essential to initialize the index X mode (T) and the decimal mode (D) flags because of their effect on calculations.

•To reference the contents of the processor status register (PS), execute the **PHP** instruction once then read the contents of (S+1). If necessary, execute the **PLP** instruction to return the PS to its original status.
A **NOP** instruction must be executed after every **PLP** instruction.

•A **SEI** instruction must be executed before every **PLP** instruction.
A **NOP** instruction must be executed before every **CLI** instruction.

BRK Instruction

It can be detected that the **BRK** instruction interrupt event or the least priority interrupt event by referring the stored B flag state. Refer to the stored B flag state in the interrupt routine.

Decimal Calculations

When decimal mode is selected, the values of the V flags are invalid.
The carry flag (C) is set to "1" if a carry is generated as a result of the calculation, or is cleared to "0" if a borrow is generated. To determine whether a calculation has generated a carry, the C flag must be initialized to "0" before each calculation. To check for a borrow, the C flag must be initialized to "1" before each calculation.

Multiplication and Division Instructions

•The index X mode (T) and the decimal mode (D) flags do not affect the MUL and DIV instruction.

Instruction Execution Time

The instruction execution time is obtained by multiplying the frequency of the internal clock ϕ by the number of cycles needed to execute an instruction.
The number of cycles required to execute an instruction is shown in the list of machine instructions.

Timers

•If a value n (between 0 and 255) is written to a timer latch, the frequency division ratio is 1/(n+1).

•P5$_1$/XCOUT/TOUT pin cannot function as an I/O port when XCIN - XCOUT is oscillating. When XCIN - XCOUT oscillation is not used or XCOUT oscillation drive is disabled, this pin can function as the TOUT output pin of the timer 1 or 2.
When using the TOUT output function and f(XCIN) divided by 2 is used as the timer 1 count source (bit 2 of T123M = "1"), disable XCOUT oscillation drive (bit 5 of CCR = "1").

Ports

•When the data register (port latch) of an I/O port is modified with the bit managing instruction (**SEB**, **CLB** instructions) the value of the unspecified bit may be changed.

•In standby state (the stop mode by executing the **STP** instruction, and the wait mode by executing the **WIT** instruction) for low-power dissipation, do not make input levels of an I/O port "undefined", especially for I/O ports of the P-channel and the N-channel open-drain.
Pull-up (connect the port to Vcc) or pull-down (connect the port to Vss) these ports through a resistor.
When determining a resistance value, note the following points:
(1) External circuit
(2) Variation of output levels during the ordinary operation

When using built-in pull-up or pull-down resistor, note on varied current values.
(1) When setting as an input port : Fix its input level
(2) When setting as an output port : Prevent current from flowing out to external

Serial I/O

Do not write to the serial I/O shift register during a transfer when in SPI compatible mode.

UART

•The all error flags PER, FER, OER and SER are cleared to "0" when the UARTx status register is read, at the hardware reset or initialization by setting the Transmit Initialization Bit. These flags are also cleared to "0" by execution of bit test instructions such as **BBC** and **BCS**.

•The transmission interrupt request bit is set and the interrupt request is generated by setting the transmit enable bit to "1" even when selecting timing that either of the following flags is set to "1" as timing where the transmission interrupt is generated:
(1) Transmit buffer empty flag is set to "1"
(2) Transmit complete flag is set to "1".

Therefore, when the transmit interrupt is used, set the transmit interrupt enable bit to transmit enabled as the following sequence:
(1) Transmit enable bit is set to "1"
(2) Transmit interrupt request bit is set to "0"
(3) Transmit interrupt enable bit is set to "1".

•The receive buffer full interrupt request is not generated if receive errors are detected at receiving.

•If a character bit length is 7 bits, bit 7 of the UARTx transmit/receive buffer register 1 and bits 0 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are invalid at receiving.
If a character bit length is 8 bits, bits 0 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are invalid at receiving.
If a character bit length is 9 bits, bits 1 to 7 of the UARTx transmit/receive buffer register 2 are ignored at transmitting; they are "0" at receiving.

USB

•When the USB Reset Interrupt Status Flag is kept at "1", all other flags in the USB internal registers (addresses 0050_{16} to $005E_{16}$) will return to their reset status. However, the following registers are not affected by the USB reset: USB control register (address 0013_{16}), Frequency synthesizer control register (address $006C_{16}$), Clock control register (address $001F_{16}$), and USB endpoint-x FIFO register (addresses 0060_{16} to 0064_{16}).

•When not using the USB function, set the USB Line Driver Supply Enable Bit of the USB control register (address 0013_{16}) to "1" for power supply to the internal circuits (at Vcc = 5V).

•When using an isochronous transfer, set the FLUSH Bit (bit 6 of address 0059_{16} and bit 6 of address $005A_{16}$) as follows:
IN FIFO: use AUTO_FLUSH Bit (bit 6 of 0058_{16})
OUT FIFO: when OUT_PKT_RDY Bit is "1", set FLUSH Bit to "1"

•When the USB SOF Port Select Bit is "1", the reference pulse of 83.3 ns (ϕ = 12 MHz) is output from the P7$_0$/$\overline{\text{SOF}}$ pin and synchronized with the SOF packet.

•The IN_PKT_RDY Bit and OUT_PKT_RDY Flag can be set or cleared by software even when using the AUTO_SET or AUTO_CLR function.

•When writing to USB-related registers, set the USB Clock Enable Bit to "1", then perform the write after four ϕ cycle waits.

•When using the MCU at Vcc = 3.3V, set the USB Line Driver Supply Enable Bit to "0" (line driver disable). Note that setting the USB Line Driver Current Control Bit (USBC3) doesn't affect the USB operation.

•Read one packet data from the OUT FIFO before clearing the OUT_PKT_RDY Flag. If the OUT_PKT_RDY Flag is cleared while one packet data is being read, the internal read pointer cannot operate normally.

•Use the transfer instructions such as **LDA** and **STA** to set the registers: USB interrupt status registers 1, 2 (addresses 0052_{16}, 0053_{16}); USB endpoint 0 IN control register (address 0059_{16}); USB endpoint x IN control register (address 0059_{16}); USB endpoint x OUT control register (address $005A_{16}$). Do not use the read-modify-write instructions such as the **SEB** or the **CLB** instruction.
When writing to bits shown by Table 39 using the transfer instruction such as **LDA** or **STA**, a value which never affect its bit state is required. Take the following sequence to change these bits contents:
(1) Store the register contents onto a variable or a data register.
(2) Change the target bit on the variable or the data register. Simultaneously mask the bit so that its bit state cannot be changed. (See to Table 39.)
(3) Write the value from the variable or the data register to the register using the transfer instruction such as **LDA** or **STA.**

Table 39 Bits of which state might be changed owing to software write

Regster name	Bit name	Value not affecting state (**Note**)
USB endpoint 0 IN control register	IN_PKT_RDY (b1)	"0"
	DATA_END (b3)	"0"
	FORCE_STALL (b4)	"1"
USB endpoint x (x = 1 to 4) IN control register	IN_PKT_RDY (b0)	"0"
	UNDER_RUN (b1)	"1"
USB endpoint x (x = 1 to 4) OUT control register	OUT_PKT_RDY (b0)	"1"
	OVER_RUN (b1)	"1"
	FORCE_STALL (b4)	"1"
	DATA_ERR (b5)	"1"

Note: Writing this value will not change the bit state, because this value cannot be written to the bit by software.

Frequency Synthesizer

•The frequency synthesizer and DC-DC converter must be set up as follows when recovering from a Hardware Reset:

(1) Enable the frequency synthesizer after setting the frequency synthesizer related registers (addresses $006C_{16}$ to $006F_{16}$). Then wait for 2 ms.

(2) Check the Frequency Synthesizer Lock Status Bit. If "0", wait for 0.1 ms and then recheck.

(3) When using the USB built-in DC-DC converter, set the USB Line Driver Supply Enable Bit of the USB control register to "1". This setting must be done 2 ms or more after the setup described in step (1). The USB Line Driver Current Control Bit must be set to "0" at this time. (When Vcc = 3.3V, the setting explained in this step is not necessary.)

(4) After waiting for (C + 1) ms so that the external capacitance pin (Ext. Cap. pin) can reach approximately 3.3 V, set the USB Clock Enable Bit to "1". At this time, "C" equals the capacitance (μ F) of the capacitor connected to the Ext. Cap. pin. For example, if 2.2 μF and 0.1 μF capacitors are connected to the Ext. Cap. in parallel, the required wait will be (2.3 + 1) ms.

(5) After enabling the USB clock, wait for 4 or more ϕ cycles, and then set the USB Enable Bit to "1".

•Bits 6 and 5 of the frequency synthesizer control register (address $006C_{16}$) are initialized to "11" after reset release. Make sure to set bits 6 and 5 to "10" after the Frequency Synthesizer Lock Status Bit goes to "1".

•When using the frequency synthesized clock function, we recommend using the fastest frequency possible of f(X_{IN}) or f(X_{CIN}) as an input clock for the PLL. Owing to the PLL mechanism, the PLL controls the speed of multiplied clocks from the source clock. As a result, when the source clock input is lower, the generated clock becomes less stable. This is because more multipliers are needed and the speed control is very rough. Higher source clock input generates a stabler clock, as less multipliers are needed and the speed control is more accurate. However, if the input clock frequency is relatively high, the PLL clock generator can quickly lock-up the output clock to the source and make the output clock very stable.

•Set the value of frequency synthesizer multiply register 2 (FSM2) so that the fPIN is 1 MHZ or higher.

DMA

•In the memory expansion mode and microprocessor mode, the DMA_{OUT} pin outputs "H" during a DMA transfer.

•Do not access the DMAC-related registers by using a DMAC transfer. The destination address data and the source address data will collide in the DMAC internal bus.

•When using the USB FIFO as the DMA transfer source or DMA transfer destination, make sure that, if you use the AUTO_SET or AUTO_CLR function, short packet data does not get mixed in with the transfer data.

•When setting the DMAC channel x enable bit (bit 7 of address 0041_{16}) to "1", be sure simultaneously to set the DMAC channel x transfer initiation source capture register reset bit (bit 6 of address 0041_{16}) to "1". If this is not performed, an incorrect data will be transferred at the same time when the DMAC is enabled.

Memory Expansion Mode & Microprocessor Mode

•In both memory expansion mode and microprocessor mode, use the LDM instruction or STA instruction to write to port P3 (address $000E_{16}$). When using the Read-Modify-Write instruction (**SEB** instruction, **CLB** instruction) you will need to map a memory that the CPU can read from and write to.

•In the memory expansion mode, if the internal and external memory areas overlap, the internal memory becomes the valid memory for the overlapping area. When the CPU performs a read or a write operation on this overlapped area, the following things happen:

(1) Read
The CPU reads out the data in the internal memory instead of in the external memory. Note that, since the CPU will output a proper read signal, address signal, etc., the memory data at the respective address will appear on the external data bus.

(2) Write
The CPU writes data to both the internal and external memories.

•The wait function is serviceable at accessing an external memory.

Stop Mode

•When the STP instruction is executed, bit 7 of the clock control register (address $001F_{16}$) goes to "0". To return from stop mode, reset CCR7 to "1".

•When using f_{SYN} (set Internal System Clock Select Bit (CPMA6) to "1") as the internal system clock, switch CPMA6 to "0" before executing the **STP** instruction. Reset CPMA6 after the system returns from Stop Mode and the frequency synthesizer has stabilized.

CPMA6 does not need to be switched to "0" when using the **WIT** instruction.

•When the **STP** instruction is being executed, all bits except bit 4 of the timer 123 mode register (address 0029_{16}) are initialized to "0". It is not necessary to set T123M1 (Timer 1 Count Stop Bit) to "0" before executing the **STP** instruction. After returning from Stop Mode, reset the timer 1 (address 0024_{16}), timer 2 (address 0025_{16}), and the timer 123 mode register (address 0029_{16}).

USAGE NOTES

Oscillator Connection Notice

The built-in feedback register (400 Ω) is internally connected between pins XIN and XOUT.

Power Supply Pins Treatment Notice

Please connect 0.1 μF and 4.7 μF capacitors in parallel between pins Vcc and Vss, and pins AVss and AVcc.

These capacitors must be connected as close as possible between the DC supply and GND pins, and also the analog supply pin and corresponding GND pin.

Wiring patterns for these supply and GND pins must be wider than other signal patterns.

These filter capacitors should not be placed near the LPF pins as they will cause noise problems

Rest Pin Treatment Notice (Noise Elimination)

If the reset input signal rises very slowly, we recommend attaching a capacitor, such as a 1000 pF ceramic capacitor with excellent high frequency characteristics, between the RESET pin and the Vss pin.

Please note the following two issues for this capacitor connection.

(1) Capacitor wiring pattern must be as short as possible (within 20 mm).

(2) The user must perform an application level operation test.

LPF Pin Treatment Notice

All passive components must be located as close as possible to the LPF pin.

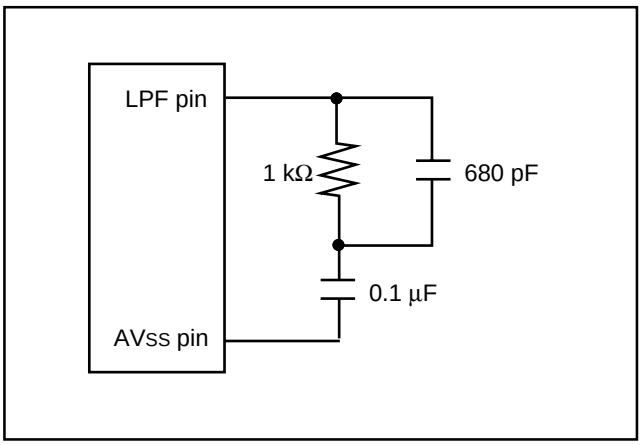


Fig. 124 Passive components near LPF pin

AVss and AVcc Pin Treatment Notice (Noise Elimination)

An insulation connector (Ferrite Beads) must be connected between AVss and Vss pins and between AVcc and Vcc pins.

USB Transceiver Treatment (Noise Elimination)

•The USB specification ver. 1.1 requires a driver -impedance 28 to 44 Ω. (Refer to Clause 7.1.1.1 Full-speed (12 Mb/s) Driver Characteristics in the USB specification.) In order to meet the USB specification impedance requirements, connect a resistor (27 Ω to 33 Ω recommended) in series between the USB D+ pin and the USB D- pin.

In addition, in order to reduce the ringing and control the falling/rising timing of USB D+/D- and a crossover point, connect a capacitor between the USB D+/D- pins and the Vss pin if necessary. The values and structure of those peripheral elements depend on the impedance characteristics and the layout of the printed circuit board. Accordingly, evaluate your system and observe waveforms before actual use and decide use of elements and the values of resistors and capacitors.

•Connect a capacitor between the Ext. Cap. pin and the Vss pin. The capacitor should have a 2.2 μF capacitor (Tantalum capacitor) and a 0.1 μF capacitor (ceramic capacitor) connected in parallel. Figure 125 for the proper positions of the peripheral components.

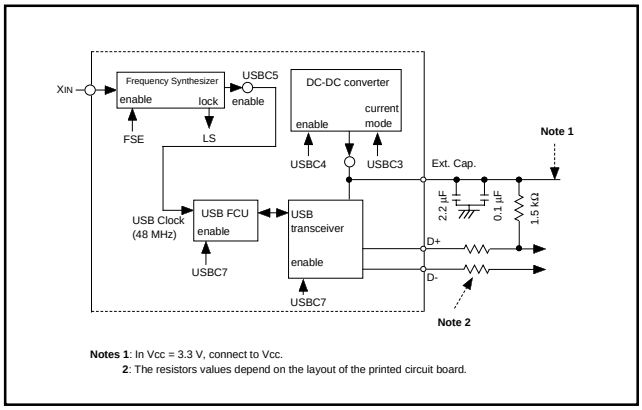


Fig.125 Peripheral circuit

•In Vcc = 3.3 V operation, connect the Ext. Cap. pin directly to the Vcc pin in order to supply power to the USB transceiver. In addition, you will need to disable the DC-DC converter in this operation (set bit 4 of the USB control register to “0”.) If you are using the bus powered supply in Vcc = 3.3 V operation, the DC-DC converter must be placed outside the MCU.

•Make sure the USB D+/D- lines do not cross any other wires. Keep a large GND area to protect the USB lines. Also, make sure you use a USB specification compliant connecter for the connection.

Clock Input/Output Pin Wiring (Noise Elimination)

(1) Make the wiring for the input/output pins as short as possible.
(2) Make the wiring across the grounding lead of the capacitor which is connected to an oscillator and the Vss pin of the MCU as short as possible (within 20 mm)
(3) Make sure to isolate the oscillation Vss pattern from other patterns for oscillation circuit-use only.

Oscillator Wiring (Noise Elimination)

(1) Keeping oscillator away from large current signal lines

Install a microcomputer (and especially an oscillator) as far as possible from signal lines, including USB signal lines, where a current larger than the tolerance of current value flows. When a large current flows through those signal lines, strong noise occurs because of mutual inductance.

(2) Installing oscillator away from signal lines where potential levels change frequently

Install an oscillator and a connecting pattern of an oscillator away from signal lines where potential levels change frequently. Also, do not cross such signal lines over the clock lines or the signal lines which are sensitive to noise.

Terminate Unused Pins

(1) Output ports : Open

(2) Input ports :

Connect each pin to Vcc or Vss through each resistor of 1 kΩ to 10 kΩ.

Ports that permit the selecting of a built-in pull-up or pull-down resistor can also use this resistor. As for pins whose potential affects to operation modes such as pins CNVss, INT or others, select the Vcc pin or the Vss pin according to their operation mode.

(3) I/O ports :

- Set the I/O ports for the input mode and connect them to Vcc or Vss through each resistor of 1 kΩ to 10 kΩ.
  Ports that permit the selecting of a built-in pull-up or pull-down resistor can also use this resistor. Set the I/O ports for the output mode and open them at “L” or “H”.

- When opening them in the output mode, the input mode of the initial status remains until the mode of the ports is switched over to the output mode by the program after reset. Thus, the potential at these pins is undefined and the power source current may increase in the input mode. With regard to an effects on the system, thoroughly perform system evaluation on the user side.

- Since the direction register setup may be changed because of a program runaway or noise, set direction registers by program periodically to increase the reliability of program.

- At the termination of unused pins, perform wiring at the shortest possible distance (20 mm or less) from microcomputer pins.

DATA REQUIRED FOR MASK ORDERS

The following are necessary when ordering a mask ROM production:

1. Mask ROM Order Confirmation Form
2. Mark Specification Form
3. Data to be written to ROM, in EPROM form (three identical copies) or one floppy disk.

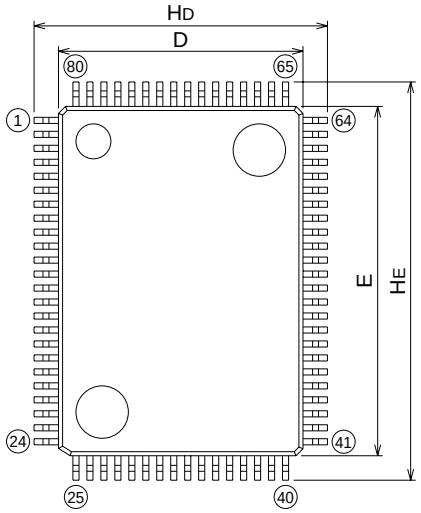
For the mask ROM confirmation and the mark specifications, refer to the “Mitsubishi MCU Technical Information” Homepage:
http://www.infomicom.maec.co.jp

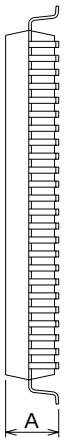
PACKAGE OUTLINE

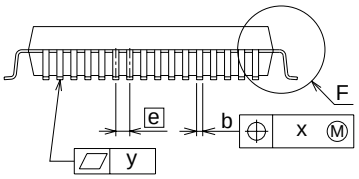
80P6N-A (MMP)

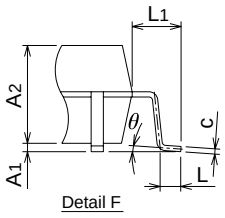
Plastic 80pin 14×20mm body QFP

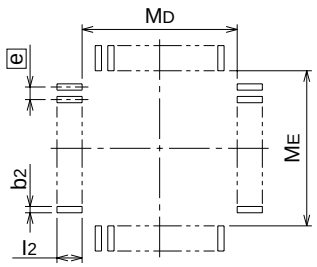
EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
QFP80-P-1420-0.80	–	1.58	Alloy 42











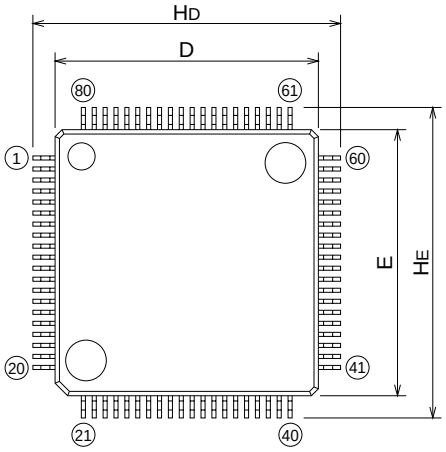
Recommended Mount Pad

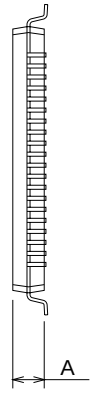
Symbol	Dimension in Millimeters		
	Min	Nom	Max
A	–	–	3.05
A_1	0	0.1	0.2
A_2	–	2.8	–
b	0.3	0.35	0.45
c	0.13	0.15	0.2
D	13.8	14.0	14.2
E	19.8	20.0	20.2
e	–	0.8	–
H_D	16.5	16.8	17.1
H_E	22.5	22.8	23.1
L	0.4	0.6	0.8
L_1	–	1.4	–
x	–	–	0.2
y	–	–	0.1
θ	0°	–	10°
b_2	–	0.5	–
l_2	1.3	–	–
M_D	–	14.6	–
M_E	–	20.6	–

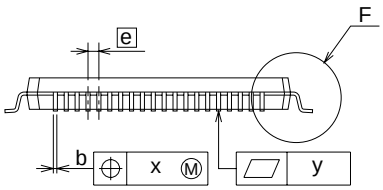
80P6Q-A (MMP)

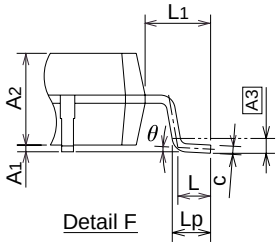
Plastic 80pin 12×12mm body LQFP

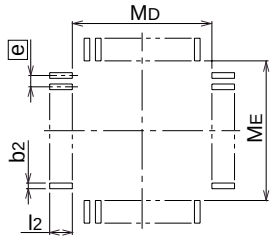
EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
LQFP80-P-1212-0.5	–	0.47	Cu Alloy











Recommended Mount Pad

Symbol	Dimension in Millimeters		
	Min	Nom	Max
A	–	–	1.7
A_1	0	0.1	0.2
A_2	–	1.4	–
b	0.13	0.18	0.28
c	0.105	0.125	0.175
D	11.9	12.0	12.1
E	11.9	12.0	12.1
e	–	0.5	–
H_D	13.8	14.0	14.2
H_E	13.8	14.0	14.2
L	0.3	0.5	0.7
L_1	–	1.0	–
Lp	0.45	0.6	0.75
A3	–	0.25	–
x	–	–	0.08
y	–	–	0.1
θ	0°	–	10°
b_2	–	0.225	–
I_2	0.9	–	–
M_D	–	12.4	–
M_E	–	12.4	–

MITSUBISHI ELECTRIC CORPORATION
HEAD OFFICE: 2-2-3, MARUNOUCHI, CHIYODA-KU, TOKYO 100-8310, JAPAN

MITSUBISHI ELECTRIC

REVISION HISTORY	7641 GROUP DATA SHEET

Rev.	Date	Description	
		Page	Summary
1.0	04/06/2001		First edition
2.0	05/17/2001	Page 1 Page 104 Page 105 Page 145 Page 146	Notes 2 are added. Fig.89 is revised: Explanation of bits 5 to 7 and Notes. Fig.90 is revised: Explanation of flow chart. “USB Transceiver Treatment” Line 9 is revised: “between the USB D+ pin and USB D- pin, or” is deleted. URL of Mitsubishi MCU Technical Information Homepage is revised: http://www.infomicom.maec.co.jp