Data Book

16bit Micro controller
TLCS-900/L1 series

TMP91C829F

REV1.2 September 7, 2001

Table of Contents

TLCS-900/L1 Devices

TMP91C829F

REV	PAGE	Modification Item
1.2	169	Serial Channel Timing modified “14X”→”16X”

CMOS 16-Bit Microcontrollers

TMP91C829F

1. OUTLINE AND FEATURES

TMP91C829 is a high-speed 16-bit microcontroller designed for the control of various mid- to large-scale equipment.With 2 Kbytes of boot ROM included , it allows your programs to be erased and rewritten on board.

TMP91C829 comes in a 100-pin flat package.

Listed below are the features.

(1) High-speed 16-bit CPU (900/L1 CPU)
- Instruction mnemonics are upward-compatible with TLCS-90/900
- 16 Mbytes of linear address space
- General-purpose registers and register banks
- 16-bit multiplication and division instructions; bit transfer and arithmetic instructions
- Micro DMA: Four-channels (444 ns/2 bytes at 36 MHz)

(2) Minimum instruction execution time: 111 ns (at 36 MHz)

(3) Built-in RAM: 8 Kbytes
Built-in ROM: None
Built-in Boot ROM: 2 Kbytes

(4) External memory expansion
- Expandable up to 16 Mbytes (shared program/data area)
- Can simultaneously support 8-/16-bit width external data bus
  ⋯ Dynamic data bus sizing

(5) 8-bit timers: 6 channels

(6) 16-bit timer/event counter: 1 channel

(7) Serial bus interface: 2 channel

(8) 10-bit AD converter: 8 channels

(9) Watchdog timer

(10) Chip Select/Wait controller: 4 blocks

(11) Interrupts: 33 interrupts

- 9 CPU interrupts: Software interrupt instruction and illegal instruction
- 17 internal interrupts: 7 priority levels are selectable.
- 7 external interrupts: 7 priority levels are selectable.
  (Level mode, rising edge mode and falling edge mode are selectable)

(12) Input/output ports: 54 pins

(13) Standby function
Three Halt modes: Idle2 (programmable), Idle1, Stop

(14) Operating voltage

- VCC (5V) = 4.75 V to 5.25 V (fc max = 36 MHz)
- VCC (3V) = 3.0 V to 3.6 V (fc max = 36 MHz)

(15) Package

- 100-pin QFP: P-LQFP100-1414-0.50B/D

Power on and power off of the supply

Power on and power off of the supply require the simultaneous execution of the 5 V power suply and 3.3 V power supply. When power on and power off of the supply is performed on eigher of them, overlap current may run into the internal logic. Leaving overlap current running results in increase of power dissipation and short LSI life.

Please avoid leaving either of power supplies on.

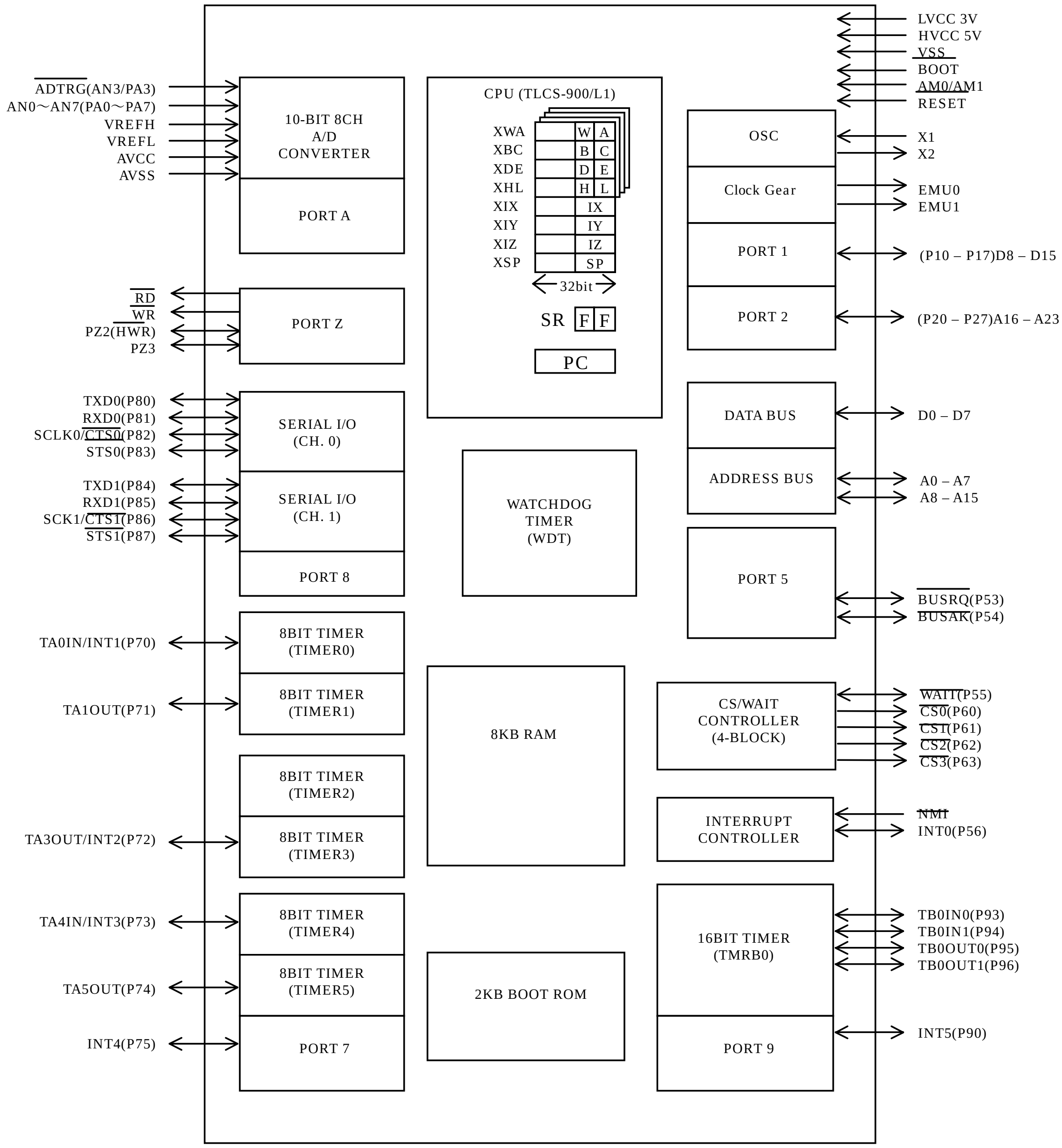


Figure 1 TMP91C829 Block Diagram

2. PIN ASSIGNMENT AND PIN FUNCTIONS

The assignment of input/output pins for the TMP91C829F, their names and functions are as follows:

2.1 Pin Assignment Diagram

Figure 2.1 shows the pin assignment of the TMP91C829F.

PIN ASSIGNMENT DIAGRAM

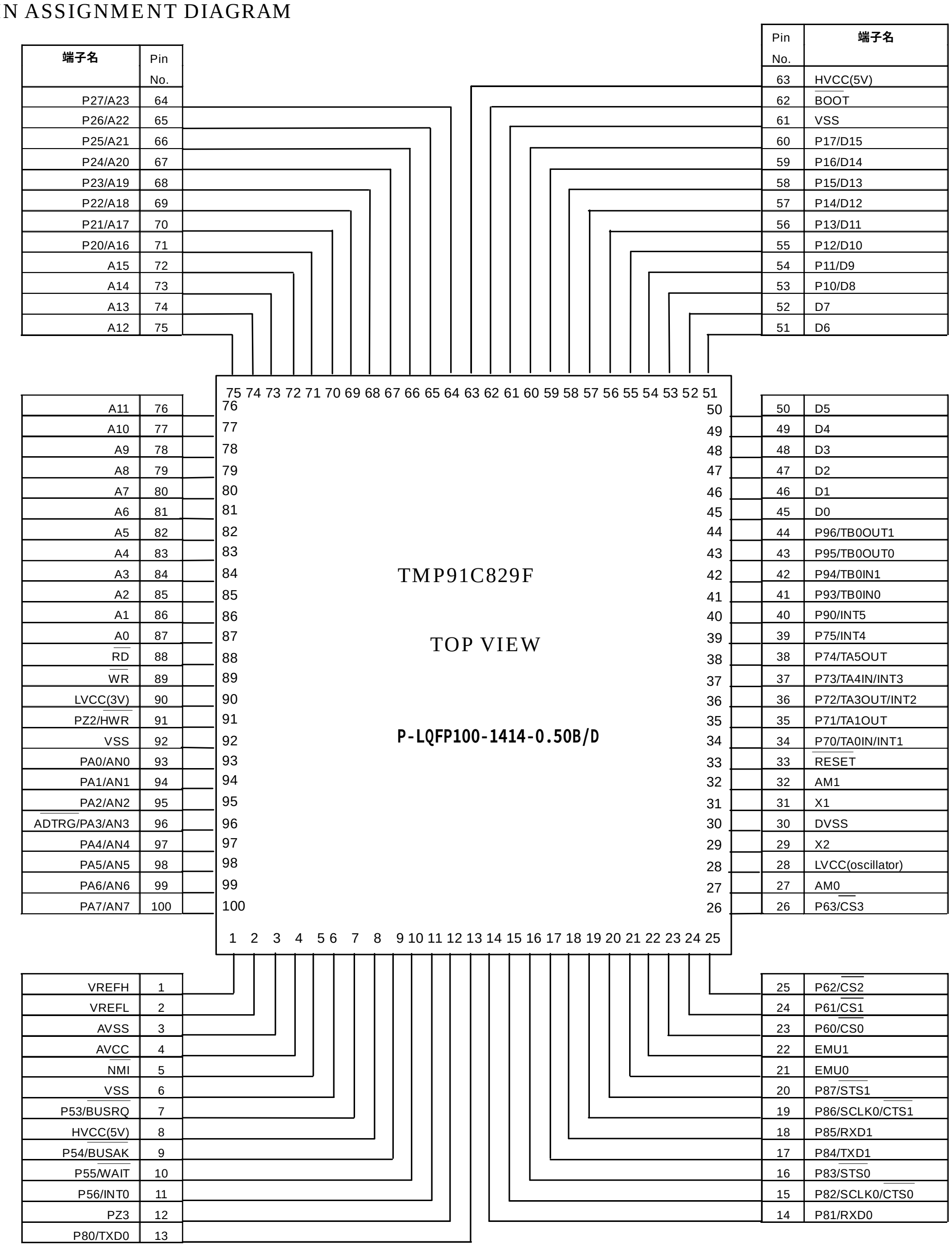


Figure 2.1 Pin assignment diagram (100-pin LQFP)

2.2 Pin Names and Functions

The names of the input/output pins and their functions are described below.

Table 2.2 Pin names and functions.

Pin Name	Number of Pins	I/O	Functions
D0 to D7	8	I/O	Data (lower): bits 0 to 7 of data bus
P10 to P17 D8 to D15	8	I/O I/O	Port 1: I/O port that allows I/O to be selected at the bit level (When used to the external 8bit bus) Data (upper): bits 8 to15 of data bus
P20 to P27 A16 to A23	8	Output Output	Port 2: Output port Address: bits 16 to 23 of address bus
A8 to A15	8	Output	Address: bits 8 to 15 of address bus
A0 to A7	8	Output	Address: bits 0 to 7 of address bus
$\overline{\text{RD}}$	1	Output	Read: strobe signal for reading external memory
$\overline{\text{WR}}$	1	Output	Write: strobe signal for writing data to pins D0 to D7
P53 $\overline{\text{BUSRQ}}$	1	I/O Input	Port 53: I/O port (with pull-up resistor) Bus Request: signal used to request Bus Release (high-impedance)
P54 $\overline{\text{BUSAK}}$	1	I/O Output	Port 54: I/O port (with pull-up resistor) Bus Acknowledge: signal used to acknowledge Bus Release (high-impedance)
P55 $\overline{\text{WAIT}}$	1	I/O Input	Port 55: I/O port (with pull-up resistor) Wait: pin used to request CPU bus wait.
P56 INT0	1	I/O Input	Port 56: I/O port (with pull-up resistor) Interrupt request pin0: Interrupt request pin with programmable level / rising edge/ falling edge
P60 $\overline{\text{CS0}}$	1	Output Output	Port 60:Output port Chip select 0: Outputs “0” when address is within specified address area.
P61 $\overline{\text{CS1}}$	1	Output Output	Port 61:Output port Chip Select 1: outputs “0” when address is within specified address area
P62 $\overline{\text{CS2}}$	1	Output Output	Port 62: Output port Chip Select 2: outputs “0” when address is within specified address area
P63 $\overline{\text{CS3}}$	1	Output Output	Port 63:Output port Chip Select 3: outputs “0” when address is within specified address area
P70 TA0IN INT1	1	I/O Input Input	Port 70: I/O port Timer A0 Input Interrupt request pin2: Interrupt request pin with programmable level / rising edge / falling edge
P71 TA1OUT	1	I/O Output	Port 71: I/O port TimerA0 or Timer A1 Output
P72 TA3OUT INT2	1	I/O Output Input	Port 72: I/O port Timer A2 or Timer A3 Output: Interrupt request pin2: Interrupt request pin with programmable level / rising edge /falling edge

Pin Name	Number of Pins	I/O	Functions
P73 TA4IN INT3	1	I/O Input Input	Port 73: I/O port Timer A4 Input Interrupt request pin3: Interrupt request pin with programmable level / rising edge/ falling edge.
P74 TA5OUT	1	I/O Output	Port 74: I/O port Timer A4 or Timer A5 output
P75 INT4	1	I/O Input	Port 75: I/O port Interrupt request pin4 : Interrupt request pin with programmable
P80 TXD0	1	I/O Output	Port 80: I/O port (with pull-up resistor) Serial Send Data 0:Programmable open drain outpin output pin
P81 RXD0	1	I/O Input	Port 81: I/O port (with pull-up resistor) Serial Receive Data 0
P82 SCLK0 $\overline{\text{CTS0}}$	1	I/O Input I/O	Port 82: I/O port: (With pull-up resistor) Serial Clock I/O 0 Serial Data Send Enable 0 (Clear to Send)
P83 $\overline{\text{STS0}}$	1	I/O	Port 83: I/O port (With pull-up resistor)
P84 TXD1	1	I/O Output	Port 84: I/O port (With pull-up resistor) Serial Send Data 0:Programmable open drain outpin output pin
P85 RXD1	1	I/O Input	Port 85: I/O port (with pull-up resistor) Serial Receive Data 1
P86 SCLK1 $\overline{\text{CTS1}}$	1	I/O Input I/O	Port 86: I/O port: (With pull-up resistor) Serial Clock I/O 1 Serial Data Send Enable 1 (Clear to Send)
P87 $\overline{\text{STS1}}$	1	I/O	Port 87: I/O port (With pull-up resistor)
P90 INT5	1	I/O Input	Port 90: I/O port Interrupt Request Pin 5: interrupt request pin with programmable level/rising edge/ falling edge
P93 TB0IN0	1	I/O Input	Port 93: I/O port Timer B0 Input 0
P94 TB0IN1	1	I/O Input	Port 94: I/O port Timer B0 Input 1
P95 TB0OUT0	1	I/O Output	Port 95: I/O port Timer B0 Output 0
P96 TB0OUT1	1	I/O Output	Port 96: I/O port Timer B0 Output 1
PA0 to PA7 AN0 to AN7 $\overline{\text{ADTRG}}$	8	Input Input Input	Port A0 to A7: Pin used to input port Analog input 0 to 7: Pins used to input to A/D converter A/D trigger: signal used to request A/D start (PA3)
PZ2 $\overline{\text{HWR}}$	1	I/O Output	Port Z2: I/O port (with pull-up resistor) High Write: strobe signal for writing data to pins D8 to D15
PZ3	1	I/O	Port Z3: I/O port (with pull-up resistor)

Pin Name	Number of Pins	I/O	Functions
$\overline{BOOT}$	1	Input	This pin sets boot mode (with pull-up resistor)
$\overline{NMI}$	1	Input	Non-Maskable Interrupt Request Pin: interrupt request pin with programmable falling edge level or with both edge levels programmable
AM0 to 1	2	Input	Address mode : External data bus with select pin When external 16-bit bus is fixed or external 8/16 bit buses are mixed, AM1="0" , AM0= "1" When external 8-bit bus is fixed, AM1="0" , AM0="0"
$\overline{RESET}$	1	Input	Reset: initializes TMP91C219F. (With pull-up resistor)
VREFH	1	Input	Pin for reference voltage input to AD converter (H)
VREFL	1	Input	Pin for reference voltage input to AD converter (L)
AVCC	1	I/O	Power supply pin for A/D converter
AVSS	1		GND supply pin for A/D converter
X1/X2	2		Oscillator connection pins
HVCC	2		Power supply pins(5V)
LVCC	2		Power supply pins(3V)
DVSS	3		GND pins (0 V)
EMU0	1	Output	Open pin
EMU1	1	output	Open pin

Note: An external DMA controller cannot access the device's built-in memory or built-in I/O devices using the $\overline{BUSRQ}$ and $\overline{BUSAK}$ signal.

Note: All pins which have a built-in pull-up resistor (other than the $\overline{RESET}$ pin and the $\overline{BOOT}$ pin) can be dicsonnected from the resistor in software.

3. Operation

This section describes the basic components, functions and operation of the TMP91C829. Notes and restrictions which apply to the various items described here are outlined in Section 7. Precautions and Restrictions at the end of this databook.

3.1 CPU

The TMP91C829 incorporates a high-performance 16-bit CPU (the 900/L1 CPU). For a description of this CPU's operation, please refer to the section of this databook which describes the TLCS-900/L1 CPU.

The following sub-sections describe functions peculiar to the CPU used in the TMP91C829; these functions are not covered in the section devoted to the TLCS-900/L1 CPU.

3.1.1 Reset

When resetting the TMP91C829 microcontroller, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the RESET input Low for at least 10 system clocks (ten states: 8.89 μs at 36 MHz). And clock gear is initialized to 1/16 mode after reset is released, so clock mode start at 1/16 of maximum speed mode.

When the Reset has been accepted, the CPU performs the following:

- Sets the Program Counter (PC) as follows in accordance with the Reset Vector stored at address FFFF00H to FFFF02H:

  PC<0 to 7> ← data in location FFFF00H
  PC<8 to 15> ← data in location FFFF01H
  PC<16 to 23> ← data in location FFFF02H

- Sets the Stack Pointer (XSP) to 100H.
- Sets bits <IFF0 to IFF2> of the Status Register (SR) to 111 (thereby setting the Interrupt Level Mask Register to level 7).
- Sets the <MAX> bit of the Status Register to 1 (MAX Mode).
  (Note: As this product does not support MIN Mode, do not write a 0 to the <MAX> bit.)
- Clears bits <RFP0 to RFP2> of the Status Register to 000 (thereby selecting Register Bank 0).

When the Reset is cleared, the CPU starts executing instructions according to the Program Counter settings. CPU internal registers not mentioned above do not change when the Reset is cleared.

When the Reset is accepted, the CPU sets internal I/O, ports and other pins as follows.

- Initializes the internal I/O registers.
- Sets the port pins, including the pins that also act as internal I/O, to General-Purpose Input or Output Port Mode.

Note: The CPU internal register (except to PC, SR, XSP) and internal RAM data do not change by resetting.

Figure 3.1.1 shows the timing of a Reset for the TMP91C829.

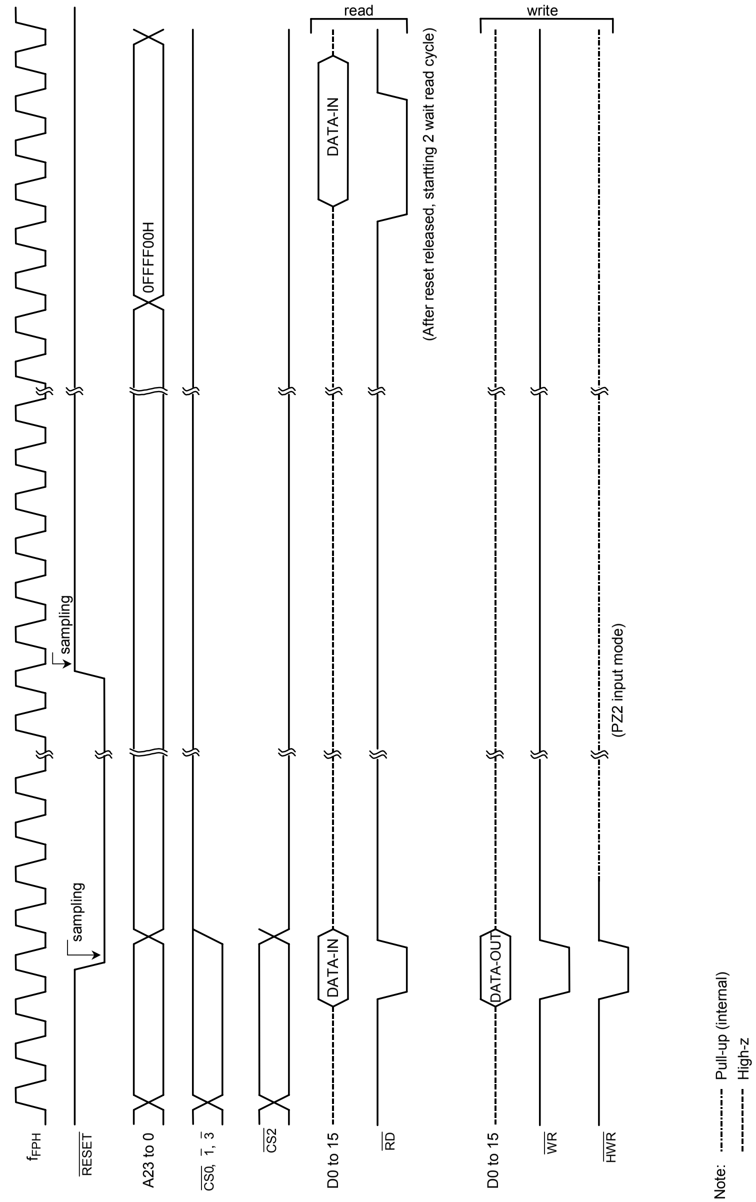


Figure 3.1.1 TMP91C829 Reset Timing Example

3.2 Outline of Operation Modes

There are multi-chip and multi-boot modes. Which mode is selected depends on the device's pin state after a reset.

- Multi-chip mode: The device nomally operations in this mode. After a reset, the device starts executing the external memory program.
- Multi-boot mode: This mode is used to rewrite the external flash memory by serial transfer (UART) or ATAPI transfer.
  After a reset, internal boot program starts up, executing a on-board rewrite program.

Table 3.2.1 Operation Mode Setup Table

Operation Mode	Mode Setup Input Pin	
	$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$
Multi-chip Mode		H
Multi-boot Mode		L

3.3 Memory Map

Figure 3.3.1 is a memory map of the TMP91C829F.

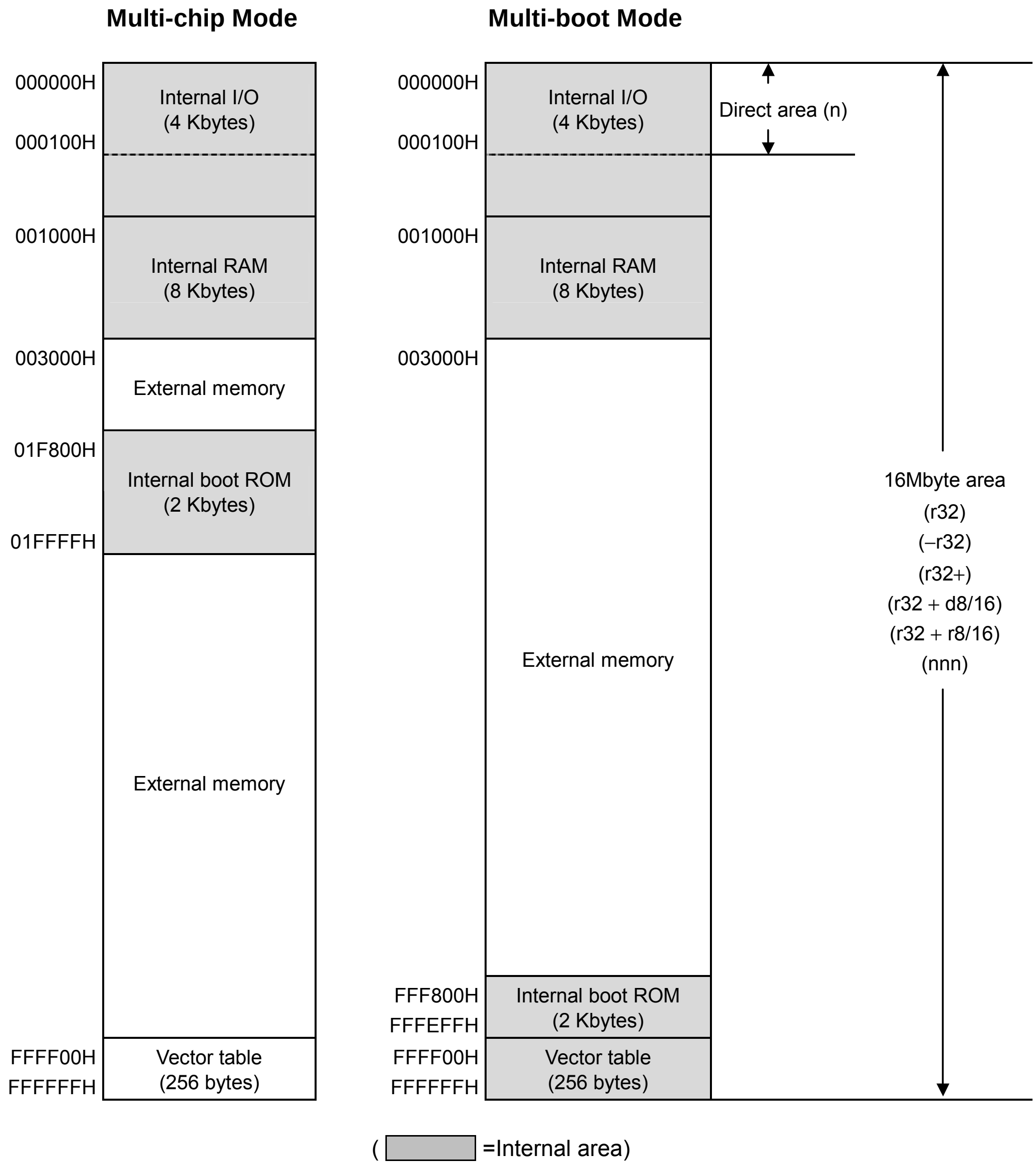


Figure 3.3.1 TMP91C829 Memory Map

3.4 Triple Clock Function and Standby Function

The TMP91C829 contains (1) a clock gearing system, (2) a standby controller and (3) a noise-reducing circuit. It is used for low-power, low-noise systems.

The clock operating mode is as follows: (a) Single Clock Mode (X1, X2 pins only).

Figure 3.4.1 shows a transition figure.

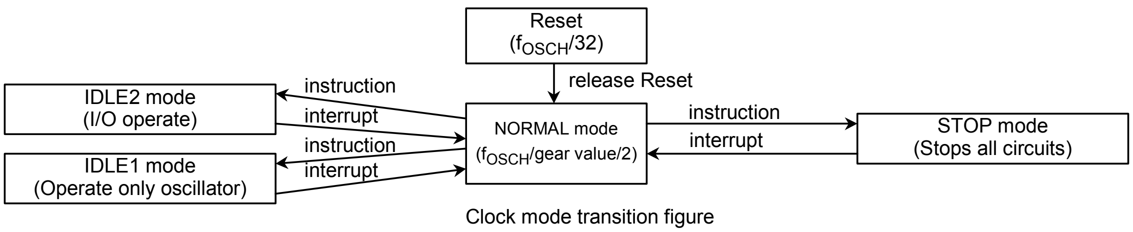


Figure 3.4.1 System clock block diagram

The clock frequency input from the X1 and X2 pins is called fc . In case of TMP91C829, fc = f_{FPH}. The system clock f_{SYS} is defined as the divided clock of f_{FPH}, and one cycle of f_{SYS} is regrred to as one state.

3.4.1 Block diagram of system clock

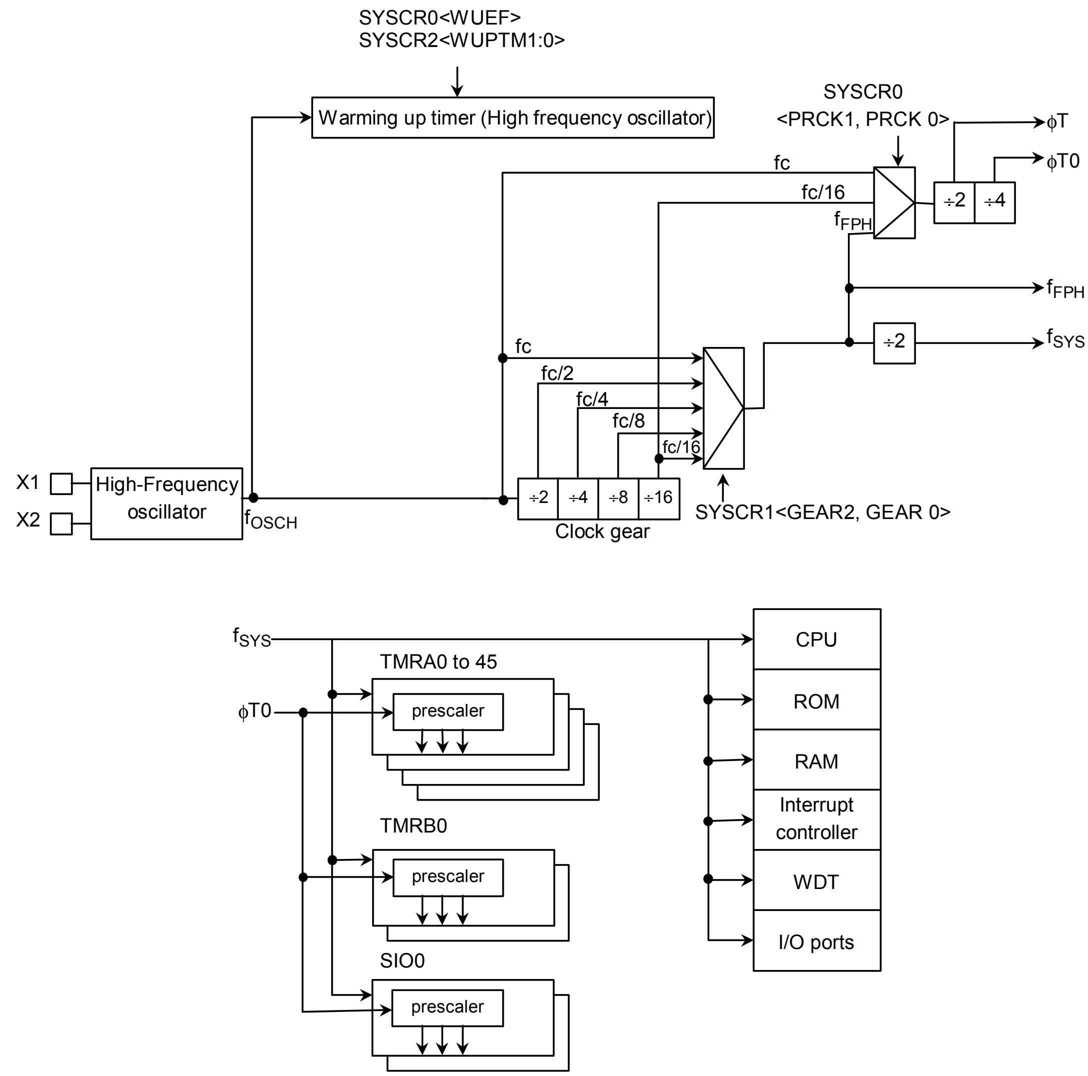


Figure 3.4.2 Block Diagram of System clock

3.4.2 SFR

		7	6	5	4	3	2	1	0
SYSCR0 (00E0H)	Bit symbol	—	—	—	—	—	WUEF	PRCK1	PRCK0
	Read/Write	R/W							
	After reset	1	0	1	0	0	0	0	0
	Function	Always Write 1	Always Write 0	Always Write 1	Always Write 0	Always Write 0	Warm-up Timer Write 0: Don't care Write 1: start timer Read 0: end warm-up Read 1: do not end warm-up	Select prescaler clock 00: f_{FPH} 01: reserved 10: fc/16 11: reserved	

		7	6	5	4	3	2	1	0
SYSCR1 (00E1H)	Bit symbol					—	GEAR2	GEAR1	GEAR0
	Read/Write					R/W			
	After reset					0	0	0	0
	Function					Always Write 0	Select gear value of high frequency (fc) 000: fc 001: fc/2 010: fc/4 011: fc/8 100: fc/16 101: (reserved) 110: (reserved) 111: (reserved)		

		7	6	5	4	3	2	1	0
SYSCR2 (00E2H)	Bit symbol		—	WUPTM1	WUPTM0	HALTM1	HALTM0		DRVE
	Read/Write		R/W	R/W	R/W	R/W	R/W		R/W
	After reset		0	1	0	1	1		0
	Function		Always Write 0	Warm-Up Timer 00: reserved 01: 2^8/inputted frequency 10: 2^{14} 11: 2^{16}		HALT mode 00: reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			1: Drive the pin during STOP/ IDLE1 mode

Figure 3.4.3 SFR for system clock

		7	6	5	4	3	2	1	0
EMCCR0 (00E3H)	Bit symbol	PROTECT	—	—	—	—	EXTIN	—	—
	Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	1	0	0	0	1	1
	Function	Protect flag 0: OFF 1: ON	Always Write 0	Always Write 1	Always Write 0	Always Write 0	1: External clock	Always Write 1	Always Write 1
EMCCR1 (00E4H)	Bit symbol	Writing 1FH turns protections off. Writing any value other than 1FH turns protection on.							
	Read/Write								
	After reset								
	Function								

Figure 3.4.4 SFR for noise-reducing

3.4.3 System clock controller

The system clock controller generates the system clock signal (f_{SYS}) for the CPU core and internal I/O. It contains a clock gear circuit for high-frequency (fc) operation. The register SYSCR1<GEAR0 to GEAR2> sets the high-frequency clock gear to either 1, 2, 4, 8 or 16 (fc, fc/2, fc/4, fc/8 or fc/16). These functions can reduce the power consumption of the equipment in which the device is installed.

The initialization<GEAR0 to GEAR2> = 100 will cause the system clock (f_{SYS}) to be set to fc/32 (fc/16 $\times$ 1/2) after a Reset.

For example, f_{SYS} is set to 1.125 MHz when the 36 MHz oscillator is connected to the X1 and X2 pins.

(1) Clock gear controller

The f_{FPH} is set according to the contents of the Clock Gear Select Register SYSCR1<GEAR0 to GEAR2> to either fc, fc/2, fc/4, fc/8 or fc/16. Using the clock gear to select a lower value of f_{FPH} reduces power consumption.

Example: Changing to a high-frequency gear

```
SYSCR1    EQU    00E1H

          LD     (SYSCR1), XXXX0000B    ;    Changes fSYS to fc/2.
```

X: Don't care

(Changing to high-frequency clock gear)

To change the clock gear, write the appropriate value to the SYSCR1<GEAR0 to GEAR2> register. The value of f_{FPH} will not change until a period of time equal to the warm-up time has elapsed from the point at which the register is written to.

There is a possibility that the instruction immediately following the instruction which changes the clock gear will be executed before the new clock setting comes into effect. To ensure that this does not happen, insert a dummy instruction (to execute a Write cycle) as follows:

```
Example:
SYSCR1    EQU    00E1H
          LD     (SYSCR1), XXXX0001B    ;    Changes fSYS to fc/4.
          LD     (DUMMY), 00H           ;    Dummy instruction
          Instruction to be executed after clock gear has changed
```

(2) Internal clock pin output function

The P84/SCOUT pin outputs an internal clock: f_{FPH} .

The following combination of settings – Port 8 Control Register P8CR<P84C> = 1 and P8FC<P84F> = 1 – specifies that a clock signal will be output on the SCOUT pin.

Table 3.4.1 shows the pin state of the P84/SCOUT pin when it is selected for clock output in the different operation modes.

Table 3.4.1 SCOUT pin states in different operation modes

NORMAL, SLOW	HALT Mode		
	IDLE2	IDLE1	STOP
Outputs f_{FPH} clock.		Fixed to 0 or 1	

3.4.4 Prescaler clock controller

For the internal I/O (TMRA01 to TMRA45, TMRB0 and SIO0) there is a prescaler which can divide the clock.

The ϕT clock input to the prescaler is either the clock f_{FPH} divided by 2 or the clock fc/16 divided by 2. The setting of the SYSCR0 <PRCK0 to PRCK1> register determines which clock signal is input.

The ϕT0 clock input to the prescaler is either the clock f_{FPH} divided by 4 or the clock fc/16 divided by 4. The setting of the SYSCR0 <PRCK0 to PRCK1> register determines which clock signal is input.

3.4.5 Noise reduction circuits

Noise reduction circuits are built in, allowing implementation of the following features.

(1) Single drive for high-frequency oscillator

(2) Protection of register contents

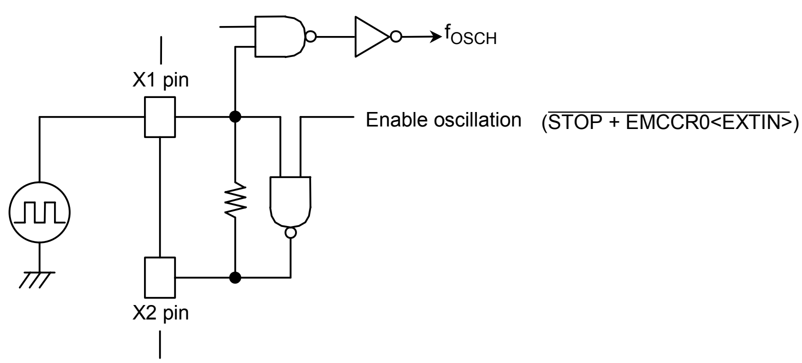
The above functions are performed by making the appropriate settings in the EMCCR0 and EMCCR1 registers.

(1) Single drive for high-frequency oscillator

(Purpose)

Not need twin-drive and protect mistake-operation by inputted noise to X2 pin when the external-oscillator is used.

(Block diagram)



(Setting method)

When a 1 is written to the EMCCR0<EXTIN>, the oscillator is disabled and is operated as a buffer. The X2 pin always outputs a 1.

<EXTIN> is initialized to 0 by a Reset.

(2) Protection of register contents

(Purpose)

An item for mistake-operation by inputted noise.

To execute the program certainty which is occurred mistake-operation, the protect-register can be disabled write-operation for the specific SFR.

Write-disabled SFRs

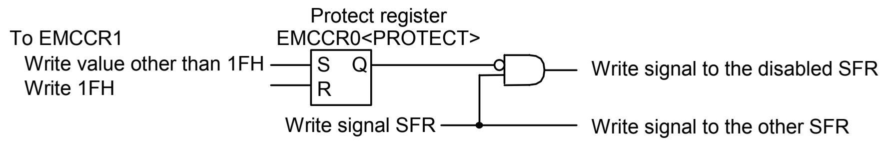
1. CS/WAIT controller

   B0CS, B1CS, B2CS, B3CS, BEXCS,
   MSAR0, MSAR1, MSAR2, MSAR3,
   MAMR0, MAMR1, MAMR2, MAMR3

2. Clock gear (only EMCCR1 can be written to.)

   SYSCR0, SYSCR1, SYSCR2, EMCCR0

(Block diagram)



(Setting method)

Writing any value other than 1FH to the EMCCR1 register turns on protection, thereby preventing the CPU from writing to the specific SFR.

Writing 1FH to EMCCR1 turns off protection.

The protection status is set in EMCCR0<PROTECT>.

Resetting initializes the protection status to OFF.

3.4.6 Standby controller

(1) HALT Modes

When the HALT instruction is executed, the operating mode switches to IDLE2, IDLE1 or STOP Mode, depending on the contents of the SYSCR2<HALTM1,HALTM0> register.

The subsequent actions performed in each mode are as follows:

① IDLE2: The CPU only is halted.
In IDLE2 Mode internal I/O operations can be performed by setting the following registers.
Table 3.4.2 shows the registers of setting operation during IDLE2 Mode.

Table 3.4.2 The registers of setting operation during IDLE2 Mode

Internal I/O	SFR
TMRA01	TA01RUN<I2TA01>
TMRA23	TA23RUN<I2TA23>
TMRA45	TA45RUN<I2TA45>
TMRB0	TB0RUN<I2TB0>
SIO0	SC0MOD1<I2S0>
AD converter	ADMOD1<I2AD>
WDT	WDMOD<I2WDT>

② IDLE1: Only the oscillator to operate.

③ STOP: All internal circuits stop operating.

The operation of each of the different HALT Modes is described in Table 3.4.3.

Table 3.4.3 I/O operation during HALT Modes

HALT Mode		IDLE2	IDLE1	STOP
SYSCR2 <HALTM1:0>		11	10	01
Block	CPU	Stop		
	I/O ports	Maintain same state as when HALT instruction was executed.		See Table 3.4.6
	TMRA, TMRB	Can be selected	Stopped	
	SIO			
	AD converter			
	WDT			
	Interrupt controller	Operational		

(2) How to clear a HALT mode

The Halt state can be cleared by a Reset or by an interrupt request. The combination of the value in <IFF0 to IFF2> of the Interrupt Mask Register and the current HALT mode determine in which ways the HALT mode may be cleared. The details associated with each type of Halt state clearance are shown in Table 3.4.4.

- Clearance by interrupt request

Whether or not the HALT mode is cleared and subsequent operation depends on the status of the generated interrupt. If the interrupt request level set before execution of the HALT instruction is greater than or equal to the value in the Interrupt Mask Register, the following sequence takes place: the HALT mode is cleared, the interrupt is then processed, and the CPU then resumes execution starting from the instruction following the HALT instruction. If the interrupt request level set before execution of the HALT instruction is less than the value in the Interrupt Mask Register, the HALT mode is not cleared. (If a non-maskable interrupt is generated, the Halt mode is cleared and the interrupt processed, regardless of the value in the Interrupt Mask Register.)

However, for INT0 to INT4 only, even if the interrupt request level set before execution of the HALT instruction is less than the value in the Interrupt Mask Register, the HALT mode is cleared. In this case, the interrupt is not processed and the CPU resumes execution starting from the instruction following the HALT instruction. The interrupt request flag remains set to 1.

- Clearance by Reset

Any Halt state can be cleared by a Reset.

When STOP Mode is cleared by a RESET signal, sufficient time (at least 3 ms) must be allowed after the Reset for the operation of the oscillator to stabilize.

When a HALT mode is cleared by resetting, the contents of the internal RAM remain the same as they were before execution of the HALT instruction. However, all other settings are re-initialized. (Clearance by an interrupt affects neither the RAM contents nor any other settings – the state which existed before the HALT instruction was executed is retained.)

Table 3.4.4 Source of Halt state clearance and Halt clearance operation

Status of Received Interrupt			Interrupt Enabled (interrupt level) ≥ (interrupt mask)			Interrupt Disabled (interrupt level) < (interrupt mask)		
HALT mode			IDLE2	IDLE1	STOP	IDLE2	IDLE1	STOP
Source of Halt state clearance	Interrupt	NMI	♦	♦	♦ *1	—	—	—
		INTWDT	♦	×	×	—	—	—
		INT0 to 4	♦	♦	♦ *1	○	○	○ *1
		INT5	♦	×	×	×	×	×
		INTTA0 to 5	♦	×	×	×	×	×
		INTTB-00, 01, OF0	♦	×	×	×	×	×
		INTRX0, TX0	♦	×	×	×	×	×
		INTRX1, TX1	♦	×	×	×	×	×
		INTAD	♦	×	×	×	×	×
	RESET		♦	♦	♦	♦	♦	♦

♦: After clearing the HALT mode, CPU starts interrupt processing. (RESET initializes the microcont.)

○: After clearing the HALT mode, CPU resumes executing starting from instruction following the HALT instruction.

×: Cannot be used to clear the HALT mode.

—: The priority level (interrupt request level) of non-maskable interrupts is fixed to 7, the highest priority level. There is not this combination type.

*1: The HALT mode is cleared when the warm-up time has elapsed.

Note: When the HALT mode is cleared by INT0 to 4 interrupt of the level mode in the interrupt enabled status, hold level H until starting interrupt processing. If level L is set before holding level L, interrupt processing is correctly started.

(Example - clearing IDLE1 Mode)

An INT0 interrupt clears the Halt state when the device is in IDLE1 Mode.

```
Address
8203H    LD     (IIMC0), 00H        ; Selects INT0 interrupt rising edge.
8206H    LD     (INTE0AD), 06H      ; Sets INT0 interrupt level to 6.
8209H    EI     5                   ; Sets interrupt level to 5 for CPU.
820BH    LD     (SYSCR2), 28H       ; Sets HALT mode to IDLE1 Mode.
820EH    HALT                       ; Halts CPU.

   INT0                                   INT0 interrupt routine

                                          RETI
820FH    LD     XX, XX
```

(3) Operation

① IDLE2 Mode

In IDLE2 Mode only specific internal I/O operations, as designated by the IDLE2 Setting Register, can take place. Instruction execution by the CPU stops.

Figure 3.4.5 illustrates an example of the timing for clearance of the IDLE2 Mode Halt state by an interrupt.

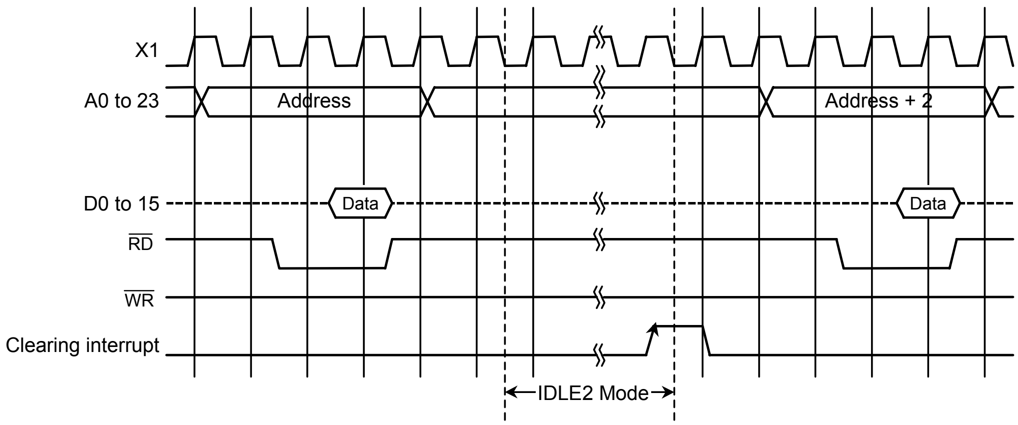


Figure 3.4.5 Timing chart for IDLE2 Mode Halt state cleared by interrupt

② IDLE1 Mode

In IDLE1 Mode, only the internal oscillator and the RTC continue to operate. The system clock in the MCU stops.

In the Halt state, the interrupt request is sampled asynchronously with the system clock; however, clearance of the Halt state (i.e. restart of operation) is synchronous with it.

Figure 3.4.6 illustrates the timing for clearance of the IDLE1 Mode Halt state by an interrupt.

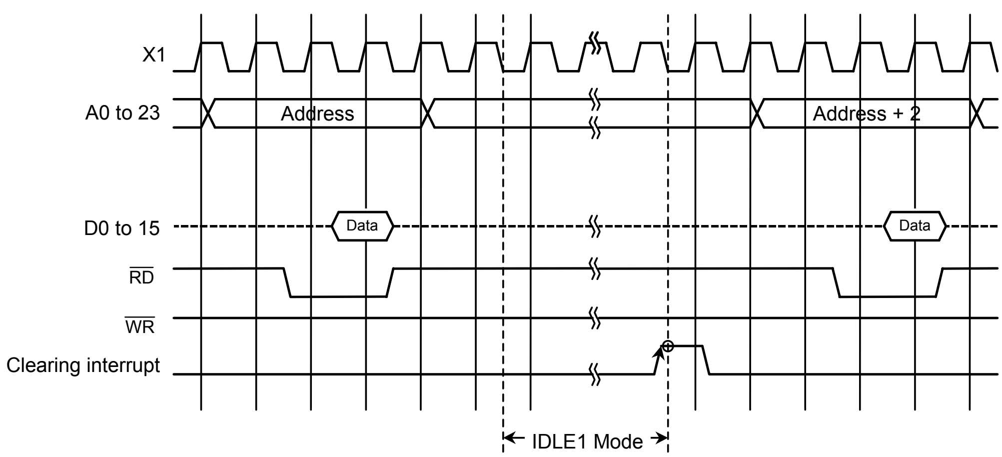


Figure 3.4.6 Timing chart for IDLE1 Mode Halt state cleared by interrupt

③ STOP Mode

When STOP Mode is selected, all internal circuits stop, including the internal oscillator Pin status in STOP Mode depends on the settings in the SYSCR2<DRVE> register. Table 3.4.6 summarizes the state of these pins in STOP Mode.

After STOP Mode has been cleared system clock output starts when the warm-up time has elapsed, in order to allow oscillation to stabilize. See the sample warm-up times in Table 3.4.5.

Figure 3.4.7 illustrates the timing for clearance of the STOP Mode Halt state by an interrupt.

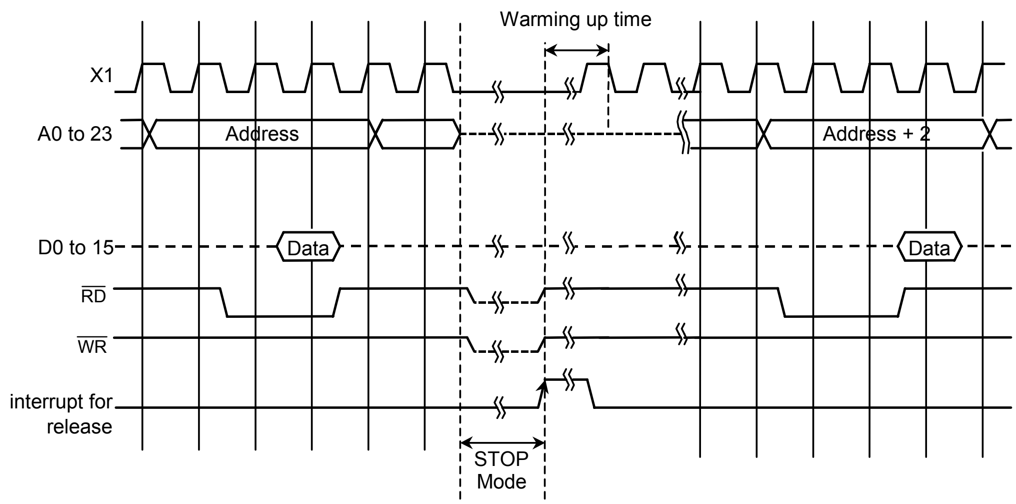


Figure 3.4.7 Timing chart for STOP Mode Halt state cleared by interrupt

Table 3.4.5 Sample warm-up times after clearance of STOP Mode

@f_{OSCH} = 36 MHz

SYSCR2<WUPTM1,WUPTM0>		
01 (2^8)	10 (2^{14})	11 (2^{16})
7.1 μs	0.455 ms	1.820 ms

Table 3.4.6 Pin states in STOP Mode

Pin Names	I/O	<DRVE> = 0	<DRVE> = 1
D0 to 7	Input/ Output Mode	—	—
P10 to 17(D8 to 15)	Input Mode Output Mode Input/output Mode	— — —	— Output —
P20 to 27(A16 to 23), A0 to 15	Input/output Mode Output		Input/Output Output
$\overline{RD}$, $\overline{WR}$	Output pin	—	Output
PZ2, PZ3	Input Mode Output Mode	PU* PU*	Input Output
P53 to P56	Input Mode Output Mode	PU* PU*	Input Output
P60 to P63	Output Mode	—	Output
P70 to P75	Input Mode Output Mode	— —	Input Output
P80 to P87	Input Mode Output Mode	PU* PU*	Input Output
P90,P93 to 97	Input Mode Output Mode	— —	Input Output
PA0 to PA7	Input Mode	—	—
$\overline{NMI}$	Input pin	Input	Input
$\overline{RESET}$	Input	Input	Input
AM0, AM1	Input	Input	Input
X1	Input	—	—
X2	Output	H Level Output	H Level Output

—: Input pin invalid (Input Mode); output pin High-Impedance (Output Mode).

Input: Input gate in operation. Input voltage should be fixed to L or H so that input pin stays constant.

Output: Output state

PU*: Programmable pull-up pin. Input Gate Disabled state. No through-current even if the pin is set to High-Impedance.

3.5 Interrupts

Interrupts are controlled by the CPU Interrupt Mask Register SR<IFF2:0> and by the built-in interrupt controller.

The TMP91C829 has a total of 33 interrupts divided into the following five types:

- Interrupts generated by CPU: 9 sources
  (Software interrupts,Illegal Instruction interrupt)
- Interrupts on external pins ($\overline{\text{NMI}}$ and INT0 to INT5): 7 sources
- Internal I/O interrupts: 19 sources

A (fixed) individual interrupt vector number is assigned to each interrupt.

One of seven (variable) priority level can be assigned to each maskable interrupt.

The priority level of non-maskable interrupts are fixed at 7 as the highest level.

When an interrupt is generated, the interrupt controller sends the piority of that interrupt to the CPU.If multiple interrupts are generated simultaneously, the interrupt controller sends the interrupt with the highest priority to the CPU.(The highest priority is level 7 using for non-maskable interrupts.)

The CPU compares the priority level of the interrupt with the value of the CPU interrupt mask register <IFF[2:0]>. If the priority level of the interrupt is higher than the value of the interrupt mask register, the CPU accepts the interrupt.

The interrupt mask register <IFF[2:0]> value can be updated using the value of the EI instruction (EI num sets <IFF[2:0]> data to num).

For example, specifying “EI 3” enables the maskable interrupts which priority level set in the interrupt controller is 3 or higher, and also non-maskable interrupts.

Operationally, the DI instruction (<IFF[2:0]> = 7) is identical to the EI 7 instruction. DI instruction is used to disable maskable interrupts because of the priority level of maskable interrupts is 0 to 6. The EI instruction is vaild immediately after execution.

In addition to the above general-purpose interrupt processing mode, TLCS-900/L1 has a micro DMA interrupt processing mode as well. The CPU can transfer the data (1/2/4 bytes) automatically in micro DMA mode, therefore this mode is used for speed-up interrupt processing, such as transferring data to the internal or external peripheral I/O. Moreover,TMP91C829 has software start function for micro DMA processing request by the software not by the hardware interrupt.

Figure 3.5.1 shows the overall interrupt processing flow.

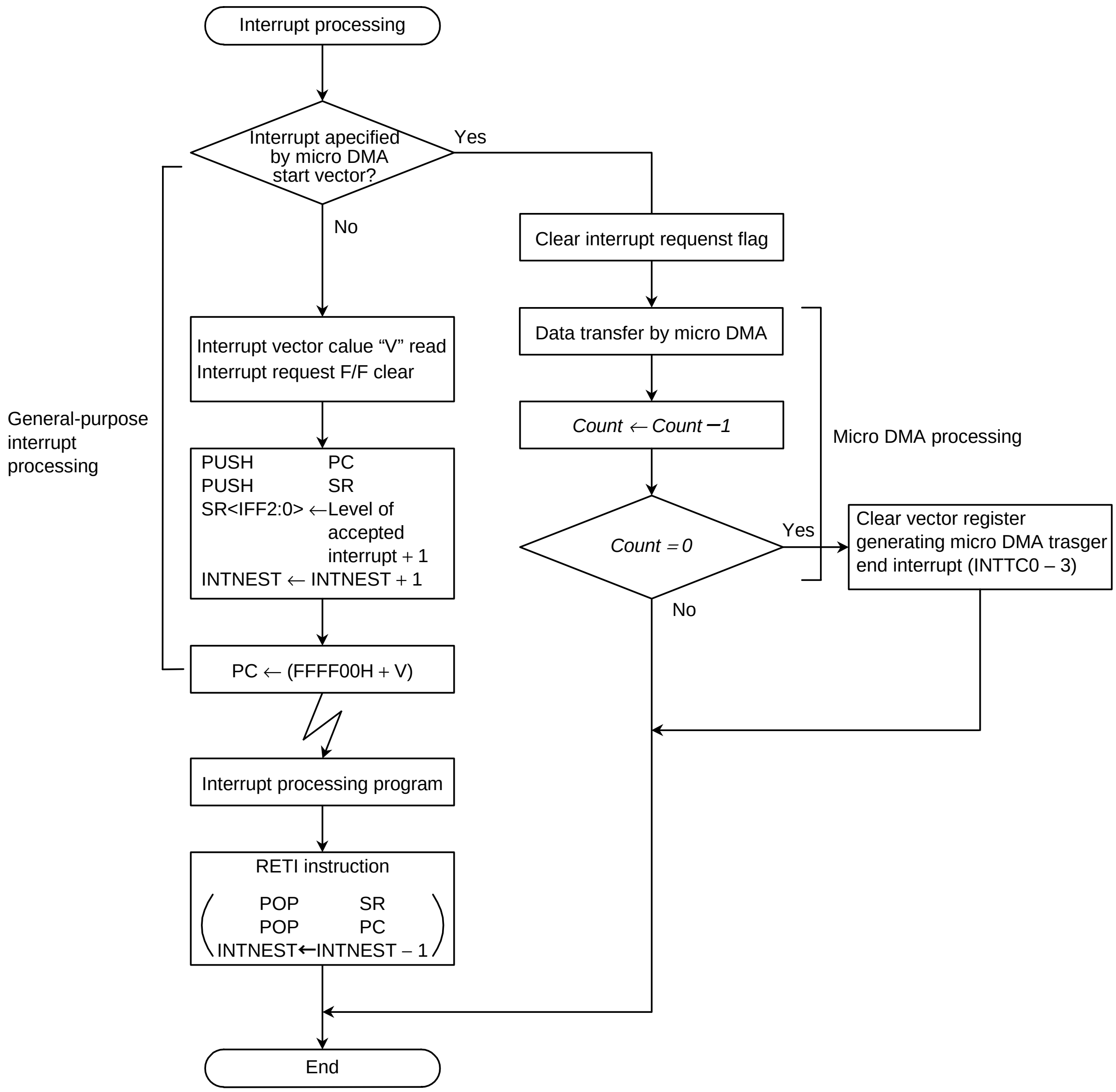


Figure 3.5.1 Interrupt and micro DMA processing sequence

3.5.1 General-purpose interrupt processing

When the CPU accepts an interrupt, it usually performs the following sequence of operations. That is also the same as TLCS-900/L and TLCS-900/H.

(1) The CPU reads the interrupt vector from the interrupt controller.
If the same level interrupts occur simultaneously, the interrupt controller generates an interrupt vector in accordance with the default priority and clears the interrupt request.
(The default priority is already fixed for each interrupt: the smaller vector value has the higher priority level.)

(2) The CPU pushes the value of Program Counter(PC) and Status Register(SR) onto the stack area (indicated by XSP).

(3) The CPU sets the value which is the priority level of the accepted interrupt plus 1(+1) to the Interrupt Mask Register <IFF[2:0]>. However, if the priority level of the accepted interrupt is 7, the register's value is set to 7.

(4) The CPU increases the interrupt nesting counter INTNEST by 1(+1).

(5) The CPU jumps to the address indicated by the data at address "FFFF00H + interrupt vector" and starts the interrupt processing routine.

The above processing time is 18-states(1.0 usec. at 36MHz) as the best case(16bits data-bus width and 0-wait).

When the CPU compled the interrupt processing, use the RETI instruction to return to the main routine. RETI restores the contents of Program Counter(PC) and Status Register(SR) from the stack and decreases the Interrupt Nesting counter INTNEST by 1(-1).

Non-maskable interrupts cannot be disabled by a user program. Maskable interrupts, however, can be enabled or disabled by a user program. A program can set the priority level for each interrupt source. (A priority level setting of 0 or 7 will disable an interrupt request.)

If an interrupt request which has a priority level equal to or greater than the value of the CPU Interrupt Mask Register <IFF[2:0]> comes out, the CPU accepts its interrupt. Then, the CPU Interrupt Mask Register <IFF[2:0]> is set to the value of the priority level for the accepted interrupt plus 1(+1).

Therefore, if an interrupt is generated with a higher level than the current interrupt during its processing, the CPU accepts the later interrupt and goes to the nesting status of interrupt processing.

Moreover, if the CPU receives another interrupt request while performing the said (1) to (5) processing steps of the current interrupt, the latest interrupt request is sampled immediately after execution of the first instruction of the current interrupt processing routine. Specifying DI as the start instruction disables maskable interrupt nesting.

A Reset initializes the Interrupt Mask Register <IFF[2:0]> to 111, disabling all maskable interrupts.

Table 3.5.1 shows the TMP91C829 interrupt vectors and micro DMA start vectors. The address FFFF00H to FFFFFFH (256 bytes) is assigned for the interrupt vector area.

Table 3.5.1 TMP91C829F interrupt vectors and micro DMA start vectors

Default Priority	Type	Interrupt Source or Source of Micro DMA Request	Vector Value	Vector Reference Address	Micro DMA Start Vector
1	Non-maskable	Reset or [SWI0] instruction	0000H	FFFF00H	—
2		[SWI1] instruction	0004H	FFFF04H	—
3		Illegal instruction or [SWI2] instruction	0008H	FFFF08H	—
4		[SWI3] instruction	000CH	FFFF0CH	—
5		[SWI4] instruction	0010H	FFFF10H	—
6		[SWI5] instruction	0014H	FFFF14H	—
7		[SWI6] instruction	0018H	FFFF18H	—
8		[SWI7] instruction	001CH	FFFF1CH	—
9		$\overline{\text{NMI}}$: NMI pin input	0020H	FFFF20H	—
10		INTWD: Watchdog Timer	0024H	FFFF24H	—
—	Maskable	Micro DMA	—	—	—
11		INT0: INT0 pin input	0028H	FFFF28H	0AH
12		INT1: INT1 pin input	002CH	FFFF2CH	0BH
13		INT2: INT2 pin input	0030H	FFFF30H	0CH
14		INT3: INT3 pin input	0034H	FFFF34H	0DH
15		INT4: INT4 pin input	0038H	FFFF38H	0EH
16		INT5: INT5 pin input	003CH	FFFF3CH	0FH
17		(reserved)	0040H	FFFF40H	10H
18		(reserved)	0044H	FFFF44H	11H
19		(reserved)	0048H	FFFF48F	12H
20		INTTA0: 8-bit timer 0	004CH	FFFF4CH	13H
21		INTTA1: 8-bit timer 1	0050H	FFFF50H	14H
22		INTTA2: 8-bit timer 2	0054H	FFFF54H	15H
23		INTTA3: 8-bit timer 3	0058H	FFFF58H	16H
24		INTTA4: 8-bit timer 4	005CH	FFFF5CH	17H
25		INTTA5: 8-bit timer 5	0060H	FFFF60H	18H
26		(reserved)	0064H	FFFF64H	19H
27		(reserved)	0068H	FFFF68H	1AH
28		INTTB00: 16-bit timer 0 (TB0RG0)	006CH	FFFF6CH	1BH
29		INTTB01: 16-bit timer 0 (TB0RG1)	0070H	FFFF70H	1CH
30		(reserved)	0074H	FFFF74H	1DH
31		(reserved)	0078H	FFFF78H	1EH
32		INTTBOF0: 16-bit timer 0 (overflow)	007CH	FFFF7CH	1FH
33		(reserved)	0080H	FFFF80H	20H
34		INTRX0: Serial receive (Channel 0)	0084H	FFFF84H	21H
35		INTTX0: Serial transmission (Channel 0)	0088H	FFFF88H	22H
36		INTRX1: Serial receive (Channel 1)	008CH	FFFF8CH	23H
37		INTTX1: Serial transmission (Channel 1)	0090H	FFFF09H	24H
38		(reserved)	0094H	FFFF94H	25H
39		(reserved)	0098H	FFFF98H	26H
40		INTAD: AD conversion end	009CH	FFFF9CH	27H
41		INTTC0: Micro DMA end (Channel 0)	00A0H	FFFFA0H	28H
42		INTTC1: Micro DMA end (Channel 1)	00A4H	FFFFA4H	29H
43		INTTC2: Micro DMA end (Channel 2)	00A8H	FFFFA8H	2AH
44		INTTC3: Micro DMA end (Channel 3)	00ACH	FFFFACH	2BH
— to —		(reserved)	00B0H to 00FCH	FFFFB0H to FFFFFCH	— to —

3.5.2 Micro DMA processing

In addition to general-purpose interrupt processing, the TMP91C829 supprots a micro DMA function. Interrupt requests set by micro DMA perform micro DMA processing at the highest priority level (level 6) among maskable interrupts, regardless of the priority level of the particular interrupt source. Micro. The micro DMA has 4 channels and is possible continuous transmission by specifing the say later burst mode.

Because the micro DMA function has been implemented with the cooperative operation of CPU, when CPU goes to a stand-by mode by HALT instruction, the requirement of micro DMA will be ignored (pending).

(1) Micro DMA operation

When an interrupt request specified by the micro DMA start vector register is generated, the micro DMA triggers a micro DMA request to the CPU at interrupt priority level 6 and starts processing the request in spite of any interrupt source's level. The micro DMA is ignored on <IFF[2:0]>=??

The 4 micro DMA channels allow micro DMA processing to be set for up to 4 types of interrupts at any one time. When micro DMA is accepted, the interrupt request flip-flop assigned to that channel is cleared.

The data are automatically transferred once(1/2/4 bytes) from the transfer source address to the transfer destination address set in the control register, and the transfer counter is decreased by 1(-1).

If the decreased result is 0, the micro DMA transfer end interrupt (INTTC0 to INTTC3) passes from the CPU to the interrupt controller. In addition, the micro DMA start vector register DMAnV is cleared to 0, the next micro DMA is disabled and micro DMA processing completes. If the decreased result is other than "0", the micro DMA processing completes if it isn't specified the say later burst mode. In this case, the micro DMA transfer end interrupt (INTTC0 to INTTC3) aren't generated.

If an interrupt request is triggered for the interrupt source in use during the interval between the clearing of the micro DMA start vector and the next setting, general-purpose interrupt processing executes at the interrupt level set. Therefore, if only using the interrupt for starting the micro DMA (not using the interrupts as a general-purpose interrupt: level 1 to 6), first set the interrupts level to 0 (interrupt requests disabled).

If using micro DMA and general-purpose interrupts together, first set the level of the interrupt used to start micro DMA processing lower than all the other interrupt levels. In this case, the cause of general interrupt is limited to the edge interrupt.

The priority of the micro DMA transfer end interrupt (INTTC0 to INTTC3) is defined by the interrupt level and the default priority as the same as the other maskable interrupt.

If a micro DMA request is set for more than one channel at the same time, the priority is not based on the interrupt priority level but on the channel number. The smaller channel number has the higher priority (Channel 0 (high) > channel 3 (low)).

While the register for setting the transfer source/transfer destination addresses is a 32-bit control register, this register can only effectively output 24-bit addresses. Accordingly, micro DMA can access 16 Mbytes (the upper eight bits of the 32 bits are not valid).

Three micro DMA transfer modes are supported: 1-byte transfer, 2-byte (one-word) transfer, and 4-byte transfer. After a transfer in any mode, the transfer source / destination addresses are increased, decreased, or remain unchanged.

This simplifies the transfer of data from I/O to memory, from memory to I/O, and from I/O to I/O. For details of the transfer modes, see (4) “Transfer Mode Register”. As the transfer counter is a 16-bit counter, micro DMA processing can be set for up to 65536 times per interrupt source.(The micro DMA processing count is maximized when the transfer counter initial value is set to 0000H.)

Micro DMA processing can be started by the 23 interrupts shown in the micro DMA start vectors of Figure 3.5.1 and by the micro DMA soft start, making a total of 24 interrupts.

Figure 3.5.2 shows the word transfer micro DMA cycle in transfer destination address INC mode (except for Counter mode, the same as for other modes).

(The conditions for this cycle are based on an external 16-bit bus, 0 waits, trandfer source/transfer destination addresses both even-numberd values).

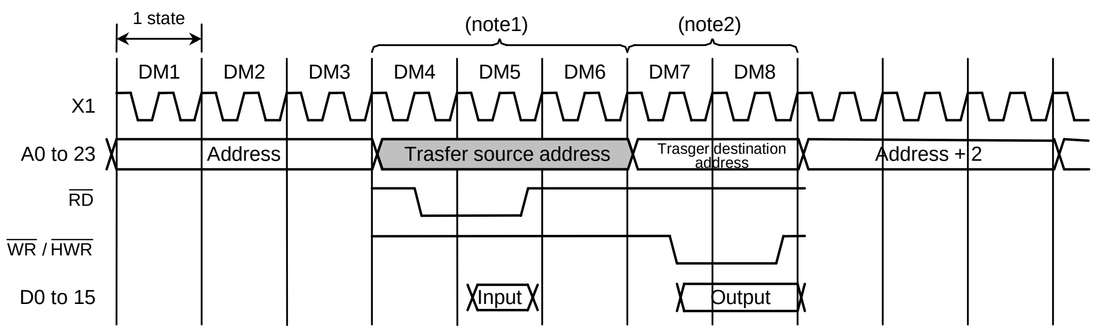


Figure 3.5.2 Timing for micro DMA cycle

States 1 to 3: Instruction fetch cycle (gets next address code).

If 3 bytes and more instruction codes are inserted in the instruction queue buffer, this cycle becomes a dummy cycle.

States 4 to 5: Micro DMA read cycle

State 6: Dummy cycle (the address bus remains unchanged from state 5)

States 7 to 8: Micro DMA write cycle

Note1: If the source address area is an 8-bit bus, it is increased by two states.
If the source address area is a 16-bit bus and the address starts from an odd number, it is increased by two states.

Note2: If the destination address area is an 8-bit bus, it is increased by two states.
If the destination address area is a 16-bit bus and the address starts from an odd number, it is increased by two states.

(2) Soft start function

In addition to starting the micro DMA function by interrupts, TMP91C815 includes a micro DMA software start function that starts micro DMA on the generation of the write cycle to the DMAR register.

Writing 1 to each bit of DMAR register causes micro DMA once. At the end of transfer, the corresponding bit of the DMAR register is automatically cleared to 0.

Only one-channel can be set once for micro DMA. (Do not write 1 to plural bits.)

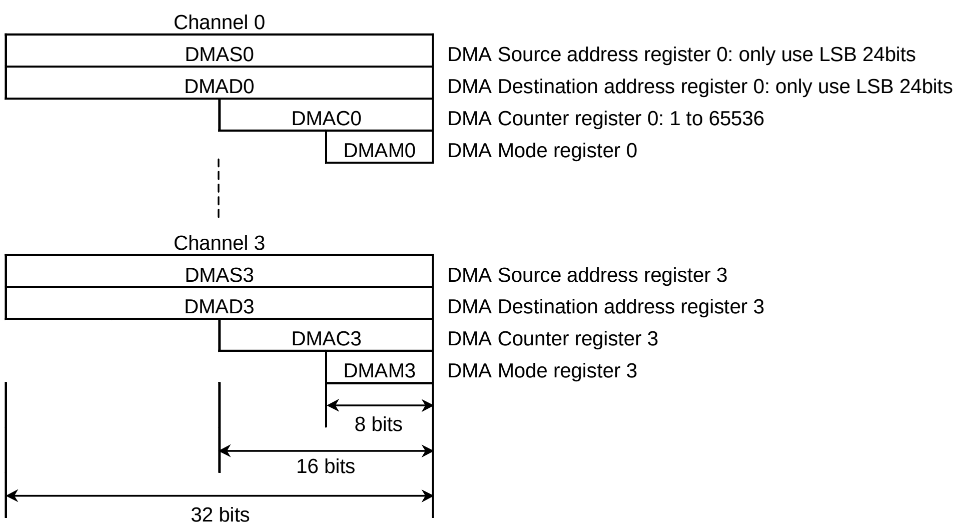
When writing again 1 to the DMAR register, check whether the bit is 0 before writing 1.

When a burst is specified by DMAB register, data is continuously transferred until the value in the micro DMA transfer counter is 0 after start up of the micro DMA.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA Request Register	89h (no RMW)					DMA Request			
							DMAR3	DMAR2	DMAR1	DMAR0
							R/W			
							0	0	0	0

(3) Transfer control registers

The transfer source address and the transfer destination address are set in the following registers. Data setting for these registers is done by an “LDC cr,r” instruction.



(4) Detailed description of the Transfer Mode Register

	← 8 bits →			
DMAM0 to DMAM3	0	0	0	Mode

Note: When setting a value in this register, write 0 to the upper 3 bits.

			Number of Transfer Bytes	Mode Description	Number of Execution States	Minimum Execution Time @ fc = 36 MHz
000 (fixed)	000	00	Byte transfer	Transfer Destination Address INC Mode ....I/O to memory (DMADn+) ← (DMASn) DMACn ← DMACn – 1 If DMACn = 0, then INTTCn is generated.	8 states	444 ns
		01	Word transfer		12 states	667 ns
		10	4-byte transfer			
	001	00	Byte transfer	Transfer Destination Address DEC Mode ....I/O to memory (DMADn–) ← (DMASn) DMACn ← DMACn – 1 If DMACn = 0, then INTTCn is generated.	8 states	444 ns
		01	Word transfer		12 states	667 ns
		10	4-byte transfer			
	010	00	Byte transfer	Transfer Source Address INC Mode ....Memory to I/O (DMADn) ← (DMASn+) DMACn ← DMACn – 1 If DMACn = 0, then INTTCn is generated.	8 states	444ns
		01	Word transfer		12 states	667 ns
		10	4-byte transfer			
	011	00	Byte transfer	Transfer Source Address DEC Mode ....Memory to I/O (DMADn) ← (DMASn–) DMACn ← DMACn – 1 If DMACn = 0, then INTTCn is generated.	8 states	444ns
		01	Word transfer		12 states	667 ns
		10	4-byte transfer			
	100	00	Byte transfer	Fixed Address Mode ....I/O to I/O (DMADn) ← (DMASn–) DMACn ← DMACn – 1 If DMACn = 0, then INTTCn is generated.	8 states	444 ns
		01	Word transfer		12 states	667 ns
		10	4-byte transfer			
	101	00	Counter Mode ....For counting number of times interrupt is generated DMASn ← DMASn + 1 DMACn ← DMACn – 1 If DMACn = 0, then INTTCn is generated.		5 states	278 ns

Note1: "n" is the corresponding micro DMA channels 0 to 3

DMADn +/DMASn+ : Post-increment (increment register value after transfer)

DMADn –/DMASn– : Post-decrement (decrement register value after transfer)

The I/Os in the table mean fixed address and the memory means increment(INC) or decrement(DEC) addresses.

Note2: Execution time is under the condition of:

16bit bus width(both translation and destination address area) / 0 wait /

fc = 36MHz / selected high frequency mode (fc × 1)

Note3: Do not use an undefined code for the transfer mode register except for the defined codes listed in the above table.

3.5.3 Interrupt controller operation

The block diagram in Figure 3.5.3 shows the interrupt circuits. The left-hand side of the diagram shows the interrupt controller circuit. The right-hand side shows the CPU interrupt request signal circuit and the halt release circuit.

For each of the 24 interrupt channels there is an interrupt request flag (consisting of a flip-flop), an interrupt priority setting register and a micro DMA start vector register. The interrupt request flag latches interrupt requests from the peripherals. The flag is cleared to zero in the following cases:

- when reset occurs
- when the CPU reads the channel vector after accepted its interrupt
- when executing an instruction that clears the interrupt (write DMA start vector to INTCLR register)
- when the CPU receives a micro DMA request (when micro DMA is set)
- when the micro DMA burst transfer is terminated

An interrupt priority can be set independently for each interrupt source by writing the priority to the interrupt priority setting register (e.g. INTE0AD or INTE12). 6 interrupt priorities levels (1 to 6) are provided. Setting an interrupt source's priority level to 0 (or 7) disables interrupt requests from that source. The priority of non-maskable interrupts (NMI pin interrupts and Watch dog Timer interrupts) is fixed at 7. If interrupt request with the same level are generated at the same time, the default priority (the interrupt with the lowest priority or, in other words, the interrupt with the lowest vector value) is used to determine which interrupt request is accepted first.

The 3rd and 7th bits of the interrupt priority setting register indicate the state of the interrupt request flag and thus whether an interrupt request for a given channel has occurred.

The interrupt controller sends the interrupt request with the highest priority among the simulateous interrupts and its vector address to the CPU. The CPU compares the priority value <IFF[2:0]> in the Status Register by the interrupt request signal with the priority value set;if the latter is higher, the interrupt is accepted. Then the CPU sets a value higher than the priority value by 1(+1) in the CPU SR <IFF[2:0]>. Interrupt request where the priority value equals or is higher than the set value are accepted simultaneously during the previous interrupt routine.

When interrupt processing is completed (after execution of the RETI instruction), the CPU restores the priority value saved in the stack before the interrupt was generated to the CPU SR<IFF[2:0]>.

The interrupt controller also has registers (4 channels) used to store the micro DMA start vector. Writing the start vector of the interrupt source for the micro DMA processing (see Table 3.5.1), enables the corresponding interrupt to be processed by micro DMA processing. The values must be set in the micro DMA parameter register (e.g. DMAS and DMAD) prior to the micro DMA processing.

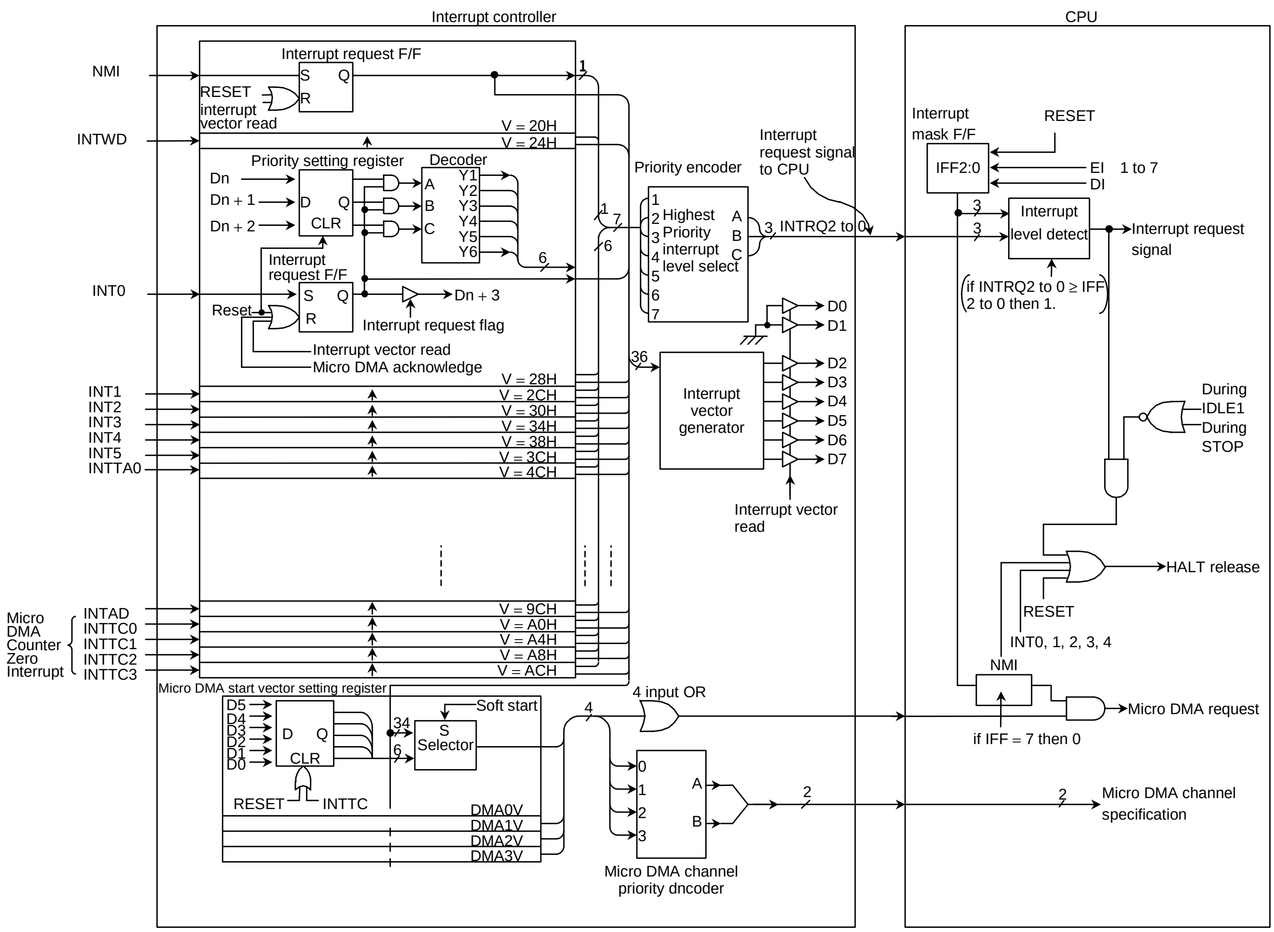


Figure 3.5.3 Block Diagram of Interrupt Controller

(1) Interrupt priority setting registers

Name	Symbol	Address	7	6	5	4	3	2	1	0	
INTE0 & INTAD Enable	INTE0AD	90h	INTAD				INT0				← Interrupt source
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0	← Bit symbol
			R	R/W			R	R/W			← Read/Write
			0	0	0	0	0	0	0	0	← After Reset
INT1 & INT2 Enable	INTE12	91h	INT2				INT1				
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	
INT3 & INT4 Enable	INTE34	92h	INT4				INT3				
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	
INT5 Enable	INTE5	93h					INT5				
							I5C	I5M2	I5M1	I5M0	
							R	R/W			
							0	0	0	0	
INTTA0 & INTTA1 Enable	INTETA01	95h	INTTA1 (TMRA1)				INTTA0 (TMRA0)				
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	
INTTA2 & UNTA3 Enable	INTETA23	96h	INTTA3 (TMRA3)				INTTA2 (TMRA2)				
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	
INTTA4 & INTTA5 Enable	INTETA45	97h	INTTA5 (TMRA5)				INTTA4 (TMRA4)				
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	

Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Name	Symbol	Address	7	6	5	4	3	2	1	0	
Interrupt Enable TMRB0	INTETB0	99H	INTTB01 (TMRB0)				INTTB00 (TMRB0)				Interrupt source
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0	Bit symbol
			R	R/W			R	R/W			Read/Write
			0	0	0	0	0	0	0	0	After Reset
Interrupt Enable TMRB0V (over flow)	INTETB0V	9BH	(Reserved)				INTTBOF0 (over flow)				
							ITF0C	ITF0M2	ITF0M1	ITF0M0	
							R	R/W			
							0	0	0	0	
Interrupt Enable Serial 0	INTES0	9CH	INTTX0				INTRX0				
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	
Interrupt Enable Serial 1	INTES1	9DH	INTTX1				INTRX1				
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	
INTTC0 & INTTC1 Enable	INTETC01	A0H	INTTC1				INTTC0				
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	
INTTC2 & INTTC3 Enable	INTETC23	A1H	INTTC3				INTTC2				
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0	
			R	R/W			R	R/W			
			0	0	0	0	0	0	0	0	

Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

(2) External interrupt control

Name	Symbol	Address	7	6	5	4	3	2	1	0
Interrupt Input Mode control 0	IIMC0	8CH (no RMW)	—	I2EDGE	I2LE	I1DGE	I1LE	I0EDGE	I0LE	NMIREE
			W							
			0	0	0	0	0	0	0	0
			Write 0	INT2EDGE 0: Rising 1: Falling	INT2EDGE 0: Edge 1: Level	INT1EDGE 0: Rising 1: Falling	INT1EDGE 0: Edge 1: Level	INT0EDGE 0: Rising 1: Falling	INT0 0: Edge 1: Level	1: Operates even on rising + falling edge of $\overline{NMI}$

INT2 level Enable

0	Edge detect INT
1	Level INT

INT1 level Enable

0	Edge detect INT
1	Level INT

INT0 level Enable

0	Edge detect INT
1	Level INT

NMI rising edge Enable

0	INT request generation at falling edge
1	INT request generation at rising/falling edge

Name	Symbol	Address	7	6	5	4	3	2	1	0
Interrupt Input Mode control1	IIMC1	8DH (no RMW)		I5EDGE	I5LE	I4EDGE	I4LE	I3EDGE	I3LE	
			W							
				0	0	0	0	0	0	
				INT5EDGE 0: Rising 1: Falling	INT5 0: Edge 1: Level	INT4EDGE 0: Rising 1: Falling	INT4 0: Edge 1: Level	INT3EDGE 0: Rising 1: Falling	INT3 0: Edge 1: Level	

INT5 level Enable

0	Edge detect INT
1	Level INT

INT4 level Enable

0	Edge detect INT
1	Level INT

INT3 level Enable

0	Edge detect INT
1	Level INT

When switching IIMC0 and 1 registers, first every FC registers in port which built-in INT function set to 0.

Setting functions on External Interrupt pins

Interrupt pin	Mode	Setting method
NMI	Falling edge	<NMIREE>=0
	Both falling and Rising edges	<NMIREE>=1
INT0	Rising edge	<I0LE>=0,<I0EDGE>=0
	Falling edge	<I0LE>=0,<I0EDGE>=1
	High level	<I0LE>=1,<I0EDGE>=0
	Low level	<I0LE>=1,<I0EDGE>=1
INT1	Rising edge	<I1LE>=0,<I1EDGE>=0
	Falling edge	<I1LE>=0,<I1EDGE>=1
	High level	<I1LE>=1,<I1EDGE>=0
	Low level	<I1LE>=1,<I1EDGE>=1
INT2	Rising edge	<I2LE>=0,<I2EDGE>=0
	Falling edge	<I2LE>=0,<I2EDGE>=1
	High level	<I2LE>=1,<I2EDGE>=0
	Low level	<I2LE>=1,<I2EDGE>=1
INT3	Rising edge	<I3LE>=0,<I3EDGE>=0
	Falling edge	<I3LE>=0,<I3EDGE>=1
	High level	<I3LE>=1,<I3EDGE>=0
	Low level	<I3LE>=1,<I3EDGE>=1
INT4	Rising edge	<I4LE>=0,<I4EDGE>=0
	Falling edge	<I4LE>=0,<I4EDGE>=1
	High level	<I4LE>=1,<I4EDGE>=0
	Low level	<I4LE>=1,<I4EDGE>=1
INT5	Rising edge	<I5LE>=0,<I5EDGE>=0
	Falling edge	<I5LE>=0,<I5EDGE>=1
	High level	<I5LE>=1,<I5EDGE>=0
	Low level	<I5LE>=1,<I5EDGE>=1

(3) Interrupt request flag clear register

The interrupt request flag is cleared by writing the appropriate micro DMA start vector, as given in Table 3.5.1, to the register INTCLR.

For example, to clear the interrupt flag INT0, perform the following register operation after execution of the DI instruction.

INTCLR ← 0AH Clears interrupt request flag INT0.

Name	Symbol	Address	7	6	5	4	3	2	1	0
Interrupt Clear Control	INTCLR	88H (no RMW)			CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			W							
			0	0	0	0	0	0	0	0
			Interrupt Vector							

(4) Micro DMA start vector registers

These registers assign micro DMA processing to an sets which source corresponds to DMA. The interrupt source whose micro DMA start vector value matches the vector set in one of these registers is designated as the micro DMA start source.

When the micro DMA transfer counter value reaches zero, the micro DMA transfer end interrupt corresponding to the channel is sent to the interrupt controller, the micro DMA start vector register is cleared, and the micro DMA start source for the channel is cleared. Therefore, in order for micro DMA processing to continue, the micro DMA start vector register must be set again during processing of the micro DMA transfer end interrupt.

If the same vector is set in the micro DMA start vector registers of more than one channel, the lowest numbered channel takes priority.

Accordingly, if the same vector is set in the micro DMA start vector registers for two different channels, the interrupt generated on the lower-numbered channel is executed until micro DMA transfer is complete. If the micro DMA start vector for this channel has not been set in the channel's micro DMA start vector register again, micro DMA transfer for the higher-numbered channel will be commenced. (This process is known as micro DMA chaining.)

Name	Symbol	Address	7	6	5	4	3	2	1	0
DMA0 Start Vector	DMA0V	80H (no RMW)			DMA0 Start Vector					
					DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
DMA1 Start Vector	DMA1V	81H (no RMW)			DMA1 Start Vector					
					DMA1V5	DMA1V4	DMA1V3	DMA0V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
DMA2 Start Vector	DMA2V	82H (no RMW)			DMA2 Start Vector					
					DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
DMA3 Start Vector	DMA3V	83H (no RMW)			DMA3 Start Vector					
					DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0

(5) Specification of a micro DMA burst

Specifying the micro DMA burst function causes micro DMA transfer, once started, to continue until the value in the Transfer Counter Register reaches zero. Setting any of the bits in the register DMAB which correspond to a micro DMA channel (as shown below) to 1 specifies that any micro DMA transfer on that channel will be a burst transfer.

Name	Symbol	Address	7	6	5	4	3	2	1	0
DMA Software Request Register	DMAR	89H (no RMW)					DMAR3	DMAR2	DMAR1	DMAR0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: DMA Software request			
DMA Burst Register	DMAB	8AH (no RMW)					DMAB3	DMAB2	DMAB1	DMAB0
							R/W			
							0	0	0	0

(6) Notes

The instruction execution unit and the bus interface unit in this CPU operate independently. Therefore if, immediately before an interrupt is generated, the CPU fetches an instruction which clears the corresponding interrupt request flag (Note), the CPU may execute this instruction in between accepting the interrupt and reading the interrupt vector. In this case, the CPU will read the default vector 0008H and jump to interrupt vector address FFFF08H.

To avoid this, an instruction which clears an interrupt request flag should always be preceded by a DI instruction.

Thus, before a POP SR instruction is executed, changing the value of the Interrupt Mask Register <IFF2 to IFF0>, a DI instruction should be used to disable interrupts. In addition, please note that the following two circuits are exceptional and demand special attention.

INT0 to 5 Level Mode	In Level Mode INT0 is not an edge-triggered interrupt. Hence, in Level Mode the interrupt request flip-flop for INT0 does not function. The peripheral interrupt request passes through the S input of the flip-flop and becomes the Q output. If the interrupt input mode is changed from Edge Mode to Level Mode, the interrupt request flag is cleared automatically.

	(For example: in case of INT0)
	If the CPU enters the interrupt response sequence as a result of INT0 going from 0 to 1, INT0 must then be held at 1 until the interrupt response sequence has been completed. If INT0 is set to Level Mode so as to release a HALT state, INT0 must be held at 1 from the time INT0 changes from 0 to 1 until the HALT state is released. (Hence, it is necessary to ensure that input noise is not interpreted as a 0, causing INT0 to revert to 0 before the HALT state has been released.)
	When the mode changes from Level Mode to Edge Mode, interrupt request flags which were set in Level Mode will not be cleared. Interrupt request flags must be cleared using the following sequence.
	DI
	LD (IIMC0), 00H; Switches interrupt input mode from Level Mode to Edge Mode.
	LD (INTCLR), 0AH; Clears interrupt request flag.
	EI

INTRX	The interrupt request flip-flop can only be cleared by a Reset or by reading the Serial Channel Receive Buffer. It cannot be cleared by an instruction.

Note: The following instructions or pin input state changes are equivalent to instructions which clear the interrupt request flag.

INT0 to 5: Instructions which switch to Level Mode after an interrupt request has been generated in Edge Mode.
The pin input changes from High to Low after an interrupt request has been generated in Level Mode. (H $\rightarrow$ L)

INTRX: Instructions which read the Receive Buffer

3.6 Port Functions

The TMP91C829 features 53 bit settings which relate to the various I/O ports.

As well as general-purpose I/O port functionality, the port pins also have I/O functions which relate to the built-in CPU and internal I/Os. Table 3.6.1 lists the functions of each port pin. Table 3.6.2 lists I/O registers and their specifications.

Table 3.6.1 Port functions (R: ↑ = with programmable pull-up resistor)

Port Name	Pin Name	Number of Pins	Direction	R	Direction Setting Unit	Pin Name for Internal Function
Port 1	P10 to P17	8	I/O	-	Bit	D8 to D15
Port 2	P20 to P27	8	Output	-	Bit	A16 to A23
Port 5	P53	1	I/O	↑	Bit	$\overline{\text{BUSRQ}}$
	P54	1	I/O	↑	Bit	$\overline{\text{BUSAK}}$
	P55	1	I/O	↑	Bit	$\overline{\text{WAIT}}$
	P56	1	I/O	↑	Bit	INT0
Port 6	P60	1	Output	-	Bit	$\overline{\text{CS0}}$
	P61	1	Output	-	Bit	$\overline{\text{CS1}}$
	P62	1	Output	-	Bit	$\overline{\text{CS2}}$
	P63	1	Output	-	Bit	$\overline{\text{CS3}}$
Port 7	P70	1	I/O	-	Bit	TA0IN /INT1
	P71	1	I/O	-	Bit	TA1OUT
	P72	1	I/O	-	Bit	TA3OUT/INT2
	P73	1	I/O	-	Bit	TA4IN/INT3
	P74	1	I/O	-	Bit	TA5OUT
	P75	1	I/O	-	Bit	INT4
Port 8	P80	1	I/O	↑	Bit	TXD0
	P81	1	I/O	↑	Bit	RXD0
	P82	1	I/O	↑	Bit	SCLK0/ $\overline{\text{CTS0}}$
	P83	1	I/O	↑	Bit	$\overline{\text{STS0}}$
	P84	1	I/O	↑	Bit	TXD1
	P85	1	I/O	↑	Bit	RXD1
	P86	1	I/O	↑	Bit	SCLK1/$\overline{\text{CTS1}}$
	P87	1	I/O	↑	Bit	$\overline{\text{STS1}}$
Port 9	P90	1	I/O	-	Bit	INT5
	P93	1	I/O	-	Bit	TB0IN0
	P94	1	I/O	-	Bit	TB0IN1
	P95	1	I/O	-	Bit	TB0OUT0
	P96	1	I/O	-	Bit	TB0OUT1
Port A	PA3	1	Input	-	(Fixed)	$\overline{\text{ADTRG}}$
	PA0 to 7	7	Input	-	(Fixed)	AN0 to AN7
Port Z	PZ2	1	I/O	↑	Bit	$\overline{\text{HWR}}$
	PZ3	1	I/O	↑	Bit	

Table 3.6.2 (a) I/O Registers and Their Specifications X: Don't care

Port	Name	Specification	I/O Registers		
			Pn	PnCR	PnFC
Port 1	P10 to P17	Input port	×	0	0
		Output port	×	1	0
		D8 to D15 bus	×	1	1
Port 2	P20 to P27	Output port	×	1	0
		A16 to A23 output	×	1	1
Port Z	PZ2	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
		$\overline{\text{HWR}}$ output	×	1	1
	PZ3	Input port (without PU)	0	0	None
		Input port (with PU)	1	0	
		Output port	×	1	
Port 5	P53	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
		$\overline{\text{BUSRQ}}$ Input (without PU)	0	0	1
		$\overline{\text{BUSRQ}}$ Input (with PU)	1	0	1
	P54	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
		BUSAK output	×	1	1
	P55	Input port / WAIT input (without PU)	0	0	None
		Input port / WAIT input (with PU)	1	0	
		Output port	×	1	
	P56	Input port / INT0 input (without PU)	0	0	1
		Input port / INT0 input (with PU)	1	0	1
		Output port	×	1	0
Port 6	P60 to P63	Output port	×	None	0
	P60	$\overline{\text{CS0}}$ output	×		1
	P61	$\overline{\text{CS1}}$ output	×		1
	P62	$\overline{\text{CS2}}$ output	×		1
	P63	$\overline{\text{CS3}}$ output	×		1
Port 7	P70 to P75	Input port	×	0	0
		Output port	×	1	0
	P70	TA0IN input	×	0	None
		INT1 input	×	0	1
	P71	TA1OUT output	×	1	1
	P72	TA3OUT output	×	1	1
		INT2 input	×	0	1
	P73	TA4IN input	×	0	None
		INT3 input	×	0	1
	P74	TA5OUT output	×	1	1
	P75	INT4 input	×	0	1

Table 3.6.2 (b) I/O Registers and Their Specifications X: Don't care

Port	Name	Specification	I/O Registers		
			Pn	PnCR	PnFC
Port 8	P80	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
		TXD0 output	×	1	1
	P81	Input port /RXD0 input (without PU)	0	0	None
		Input port /RXD0 input (with PU)	1	0	
		Output port	×	1	
	P82	Input port /SCLK0/CTS0 input (without PU)	0	0	0
		Input port /SCLK0/CTS0 input (with PU)	1	0	0
		Output port	×	1	0
		SCLK0 output	×	1	1
	P83	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
		$\overline{\text{STS0}}$ output	×	1	1
	P84	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
		TXD1 output	×	1	1
	P85	Input port /RXD1 input (without PU)	0	0	None
		Input port /RXD1 input (with PU)	1	0	
		Output port	×	1	
	P86	Input port /SCLK1/CTS1 input (without PU)	0	0	0
		Input port /SCLK1/CTS1 input (with PU)	1	0	0
		Output port	×	1	0
		SCLK1 output	×	1	1
	P87	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
		$\overline{\text{STS1}}$ output	×	1	1
Port 9	P90	Input port	×	0	0
		Output port	×	1	0
		INT5 input	×	0	1
	P93 to P96	Input port	×	0	None
		Output port	×	1	
	P93	TB0IN0 input	×	0	
	P94	TB0IN1 input	×	0	
	P95	TB0OUT0 output	×	1	1
	P96	TB0OUT1 output	×	1	1
Port A	PA3	Input port	×	None	
		$\overline{\text{ADTRG}}$ input	×		
	PA0 to PA7	Input port	×		
		AN0 to AN7	×		

Note 1: When PA1 to PA4 are used as AD converter input channels, a 3-bit field in the AD Mode Control Register ADMOD1<ADCH2 to ADCH0> is used to select the channel.

Note 2: When PA0 is used as the $\overline{\text{ADTRG}}$ input, ADMOD1<ADTRGE> is used to enable external trigger input.

After a Reset the port pins listed below function as general-purpose I/O port pins.

A Reset sets I/O pins which can be programmed for either input or output to be input port pins.

Setting the port pins for internal function use must be done in software.

Note about bus release and programmable pull-up I/O port pins

When the bus is released (i.e. when $\overline{\text{BUSAK}} = 0$), the output buffers for D0 to D15, A0 to A23, and the control signals ($\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{HWR}}$ and $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$) are off and are set to High-Impedance.

However, the output of built-in programmable pull-up resistors are kept before the bus is released. These programmable pull-up resistors can be selected ON/OFF by programmable when they are used as the input ports.

When they are used as output ports, they cannot be turned ON/OFF in software.
Table 3.6.3 shows the pin states after the bus has been released.

Table 3.6.3 Pin states (after bus release)

Pin Names	Pin State (after bus release)	
	Used as port	Used for function
P10 to P17 (D8 to D15)	Unchanged (i.e. not set to High-Impedance (Hi-Z))	High-Impedance (Hi-Z)
P20 to P27 (A16 to 23)	Unchanged (i.e. not set to High-Impedance (Hi-Z))	First all bits are set High, then they are set to High-Impedance (Hi-Z).
$\overline{\text{RD}}$ $\overline{\text{WR}}$	↑	↑
PZ2 ($\overline{\text{HWR}}$)	↑	The output buffer is set to OFF. The programmable pull-up resistor is set to ON irrespective of the output latch.
P60 ($\overline{\text{CS0}}$) P61 ($\overline{\text{CS1}}$) P62 ($\overline{\text{CS2}}$) P63 ($\overline{\text{CS3}}$)	↑	↑

Figure 3.6.1 shows an example external interface circuit when the bus release function is used.

When the bus is released, neither the internal memory nor the internal I/O can be accessed. However, the internal I/O continues to operate. As a result, the watchdog timer also continues to run. Therefore, the bus release time must be taken into account and care must be taken when setting the detection time for the WDT.

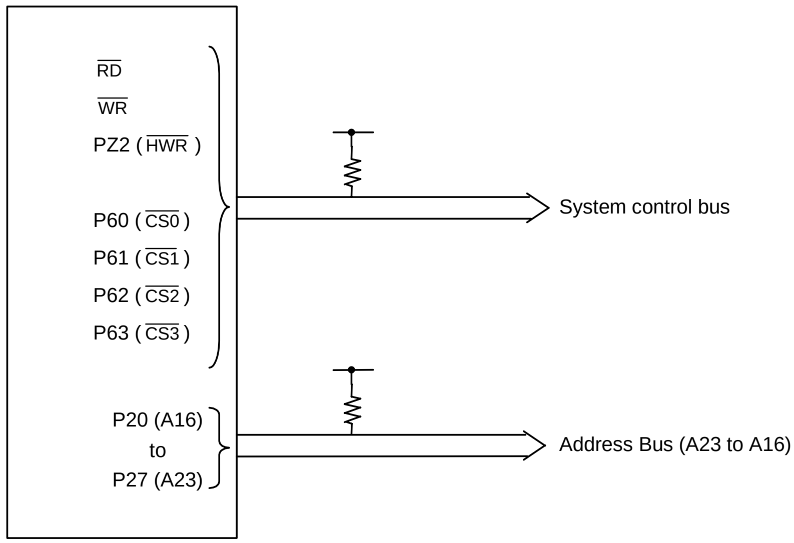


Figure 3.6.1 Interface circuit example (using bus release function)

The above circuit is necessary to set the signal level when the bus is released.

A reset sets ($\overline{RD}$) and ($\overline{WR}$), P60 ($\overline{CS0}$), P61 ($\overline{CS1}$), P62 ($\overline{CS2}$), P63 ($\overline{CS3}$) to output, and PZ2 ($\overline{HWR}$) and P54 ($\overline{BUSAK}$) to input with pull-up resistor.

3.6.1 Port 1 (P10 to P17)

Port 1 is an 8-bit general-purpose I/O port. Each bit can be set individually for input or output using the control register P1CR. Resetting , the control register P1CR to 0 and sets Port 1 to input mode.

In addition to functioning as a general-purpose I/O port, Port 1 can also function as an address data bus (D8 to 15).

In case of AM1 = 0 , and AM = 1 (outside 16-bit data bus), port 1 always functions as the data bus (D8 to D15) irrespective of the setting in P1CR control register.

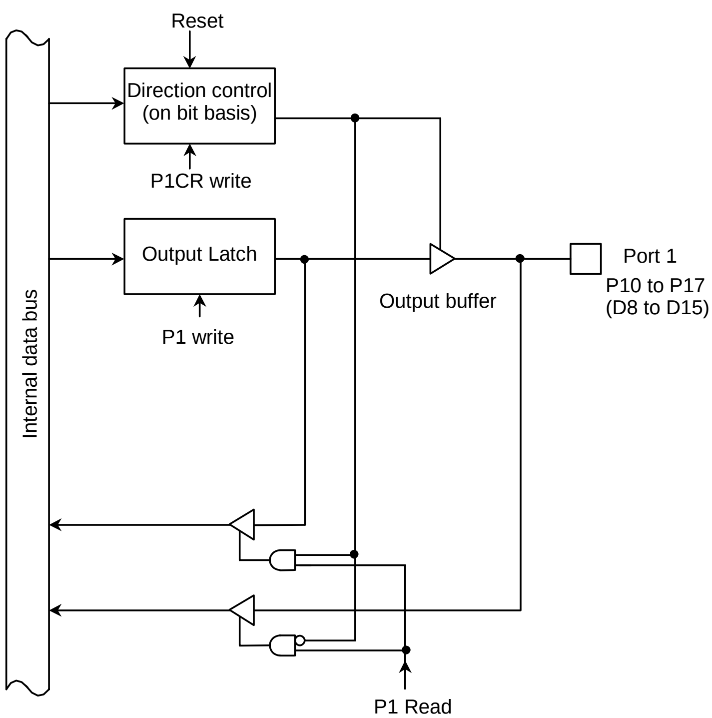


Figure 3.6.2 Port 1

Port 1 Register

P1 (0001H)

	7	6	5	4	3	2	1	0
Bit symbol	P17	P16	P15	P14	P13	P12	P11	P10
Read/Write	R/W							
After Reset	Input mode (Output latch register is cleared to 0.)							

Port 1 Control Register

P1CR (0004H)

	7	6	5	4	3	2	1	0
Bit symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0
Function	0: IN 1: OUT							

Note: Read-modify-write is prohibited for P1CR.

Port 1 I/O setting

0	Input
1	Output

Figure 3.6.3 Register for Port 1

3.6.2 Port 2 (P20 to P27)

Port 2 is an 8-bit output port. In addition to functioning as a output port, Port 2 can also function as an address bus (A16 to A23).

Each bit can be set individually for address bus using the function register P2FC. Resetting sets all bits of the function register P2FC to 1 and sets Port 2 to address bus.

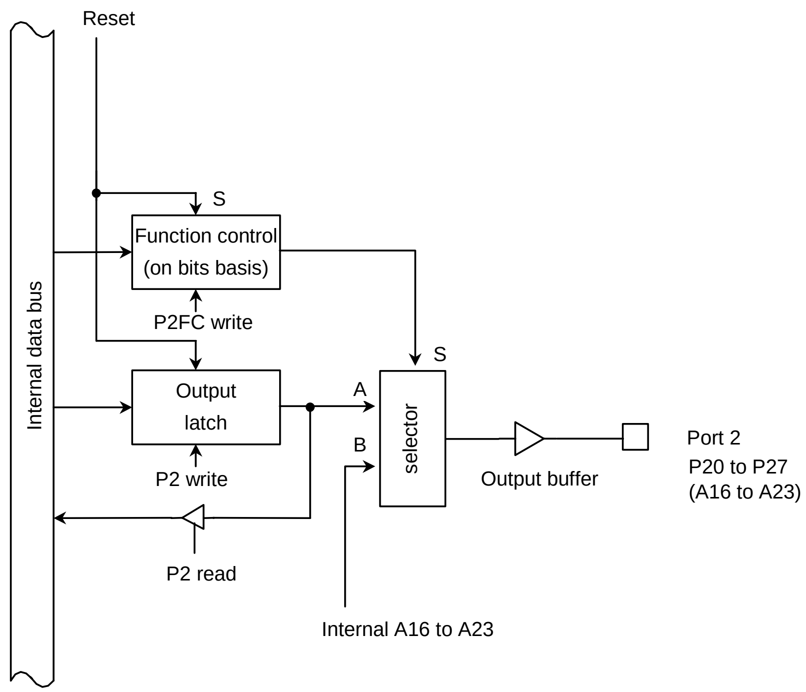


Figure 3.6.4 Port 2

Port 2 Register

		7	6	5	4	3	2	1	0
P2 (0006H)	Bit symbol	P27	P26	P25	P24	P23	P22	P21	P20
	Read/Write	R/W							
	After Reset	Output latch register is set to 1							

Port 2 Function Register

		7	6	5	4	3	2	1	0
P2FC (0009H)	Bit symbol	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
	Read/Write	W							
	After Reset	1	1	1	1	1	1	1	1
	Function	0: Port 1: Address bus (A23 to A16)							

Note: Read-modify-write is prohibited for P2FC.

Figure 3.6.5 Register for Port 2

3.6.3 Port 5 (P53 to P56)

Port 5 is an 4-bit general-purpose I/O port. I/O is set using control register P5CR and P5FC. Resetting resets all bits of the output latch P5 to 1, the control register P5CR and the function register P5FC to 0 and sets P52 to P56 to input mode with pull-up register.

In addition to functioning as a general-purpose I/O port, Port 5 also functions as I/O for the CPU's control / status signal.

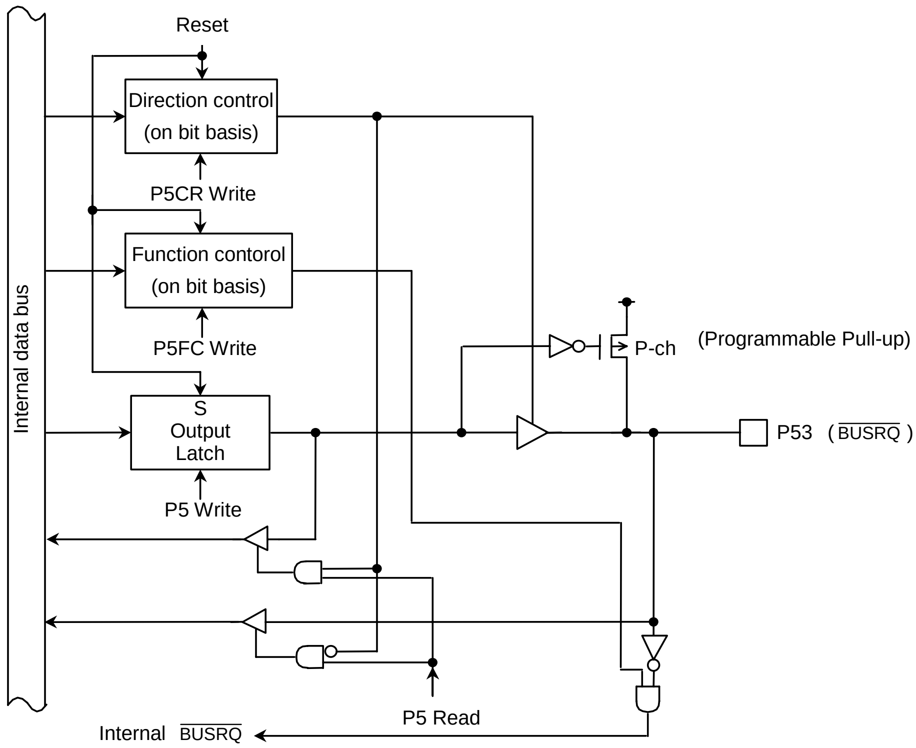


Figure 3.6.6 Port 53

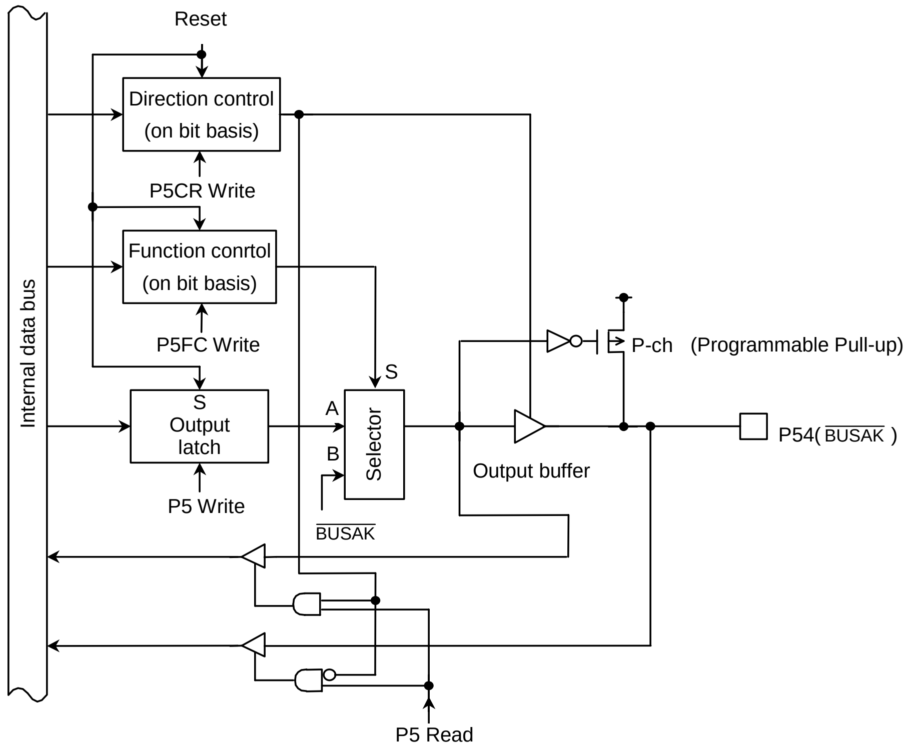


Figure 3.6.7 Port 54

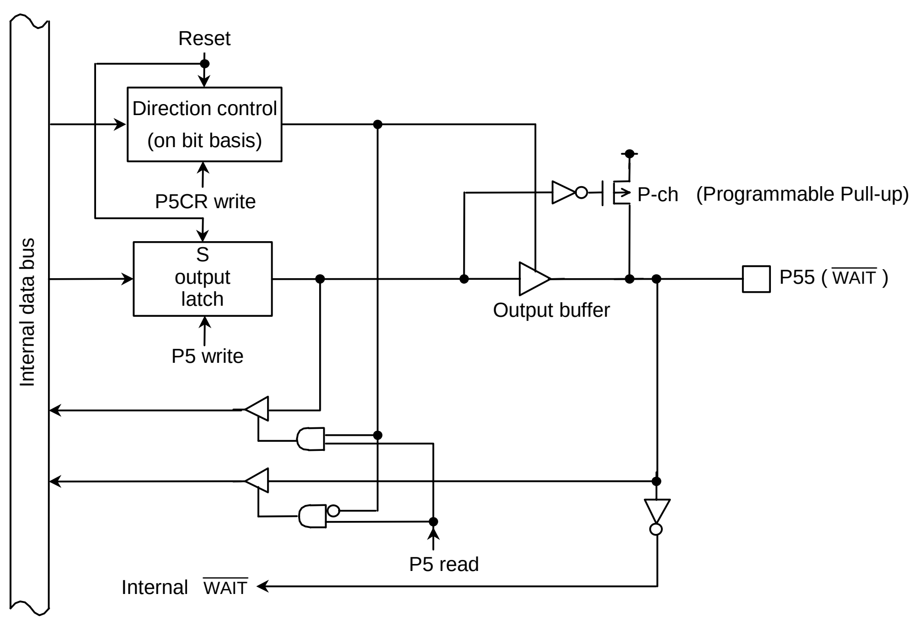


Figure 3.6.8 Port 55

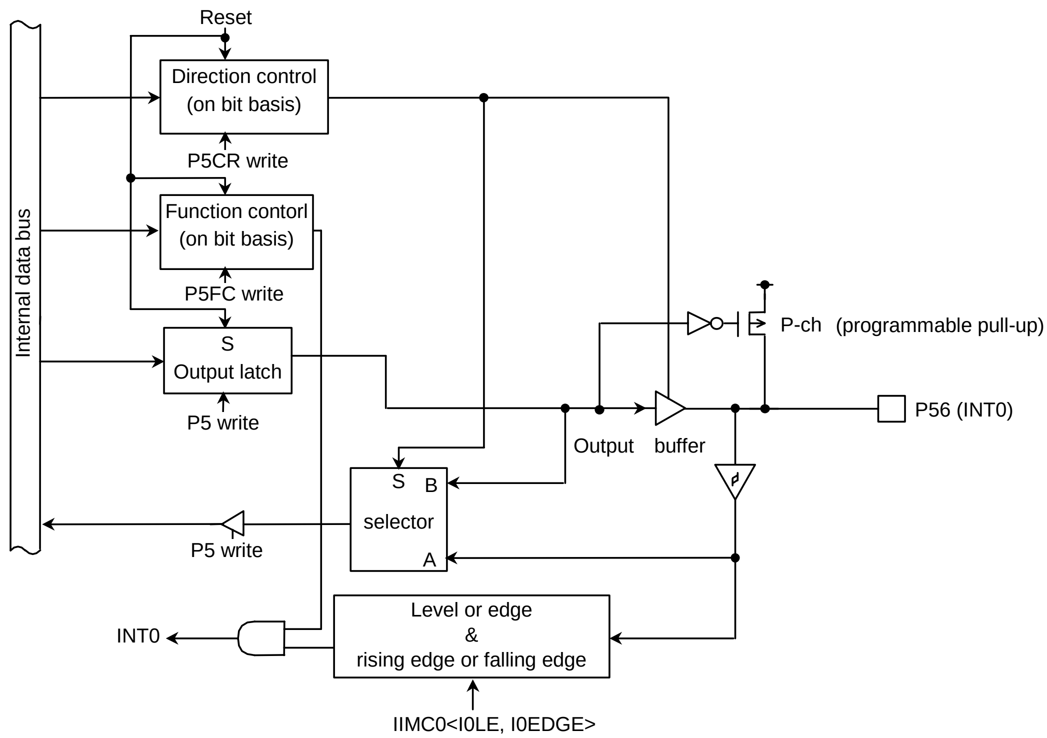


Figure 3.6.9 Port 56

Port 5 Register

		7	6	5	4	3	2	1	0
P5 (000DH)	Bit symbol		P56	P55	P54	P53			
	Read/Write		R/W						
	After Reset		Input mode (With Pull-up)						
			1	1	1	1			

Port 5 Control Register

		7	6	5	4	3	2	1	0
P5CR (0010H)	Bit symbol		P56C	P55C	P54C	P53C			
	Read/Write		W						
	After Reset		0	0	0	0			
			0: IN 1: OUT						

I/O setting

0	Input
1	Output

Port 5 Function Register

		7	6	5	4	3	2	1	0
P5FC (0011H)	Bit symbol		P56F		P54F	P53F			
	Read/Write				W				
	After Reset		0		0	0			
	Function	Always Write 0	0: PORT 1: INT0 Input		0: PORT 1: $\overline{\text{BUSAK}}$	0: PORT 1: $\overline{\text{BUSRQ}}$			

Note1: Read-modify-write is prohibited for register P5CR, P5FC.

Note2: When port5 is used in the input mode, P5 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode. Setting the built-in pull-up resistor may be depended on the States of the input pin.

Note3: When P55 pin is used as a $\overline{\text{WAIT}}$ pin ,set P5CR<P55C> to 0 and Chip Select/WAIT control register <BnW2:0> to 010.

Figure 3.6.10 Register for Port 5

3.6.4 Port 6 (P60 to P63)

Port 6 is a 4-bit output port. When reset, the P62 latch is cleared to 0 while the P60-P63 output latches are set to 1.

In addition to functioning as an output port, this port can output standard chip select signals ($\overline{CS0}$ to $\overline{CS3}$). These settings are made by using the P6FC register. When reset, the P6FC register has all of its bits cleared to 0, so that the port is set for output mode.

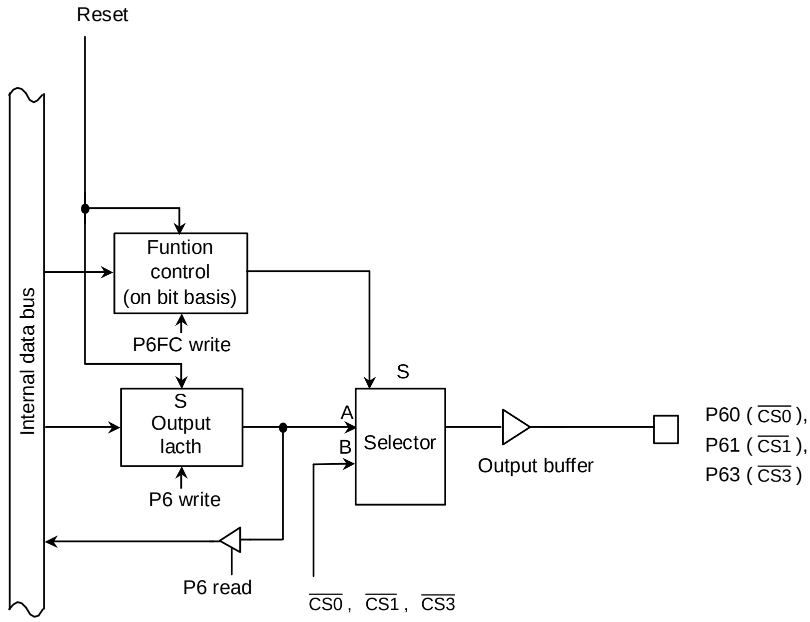


Figure 3.6.11 Port 60, 61, 63

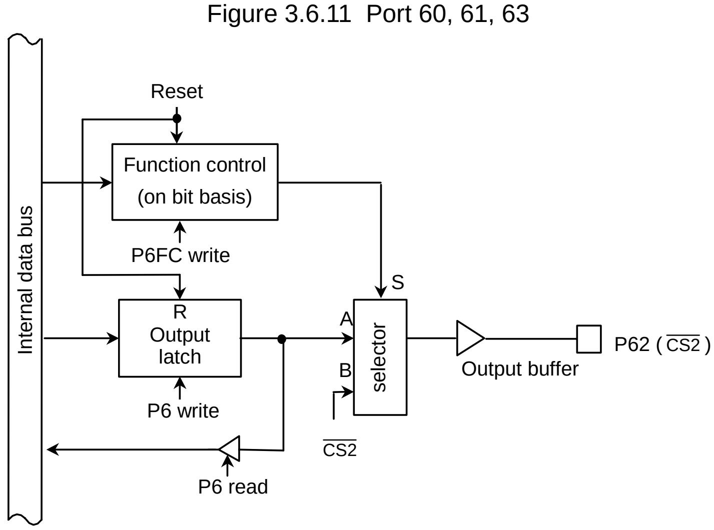


Figure 3.6.12 Port 62

Port 6 Register

		7	6	5	4	3	2	1	0
P6 (0012H)	Bit symbol					P63	P62	P61	P60
	Read/Write					R/W			
	After Reset					1	0	1	1

Port 6 Function Register

		7	6	5	4	3	2	1	0
P6FC (0015H)	Bit symbol					P63F	P62F	P61F	P60F
	Read/Write					W			
	After Reset					0	0	0	0
	Function					0: PORT1 1: $\overline{CS}$			

Note: Read-Modify-Write is prohibited for the registers P6FC.

0	Port (P60)
1	$\overline{CS0}$

0	Port (P61)
1	$\overline{CS1}$

0	Port (P62)
1	$\overline{CS2}$

0	Port (P63)
1	$\overline{CS3}$

Figure 3.6.13 Register for Port 6

3.6.5 Port 7 (P70 to P75)

Port 7 is a 6-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets Port 7 to be an input port. In addition to functioning as a general-purpose I/O port, the individual port pins can also have the following functions: port pins 70 and 73 can function as the inputs TA0IN and TA4IN to the 8-bit timer, and port pins 71, 72 and 74 can function as the 8-bit timer outputs TA1OUT, TA3OUT and TA5OUT. For each of the output pins, timer output can be enabled by writing a 1 to the corresponding bit in the Port 7 Function Register (P7FC).

Resetting resets all bits of the registers P7CR and P7FC to 0, and sets all bits to be input port pins.

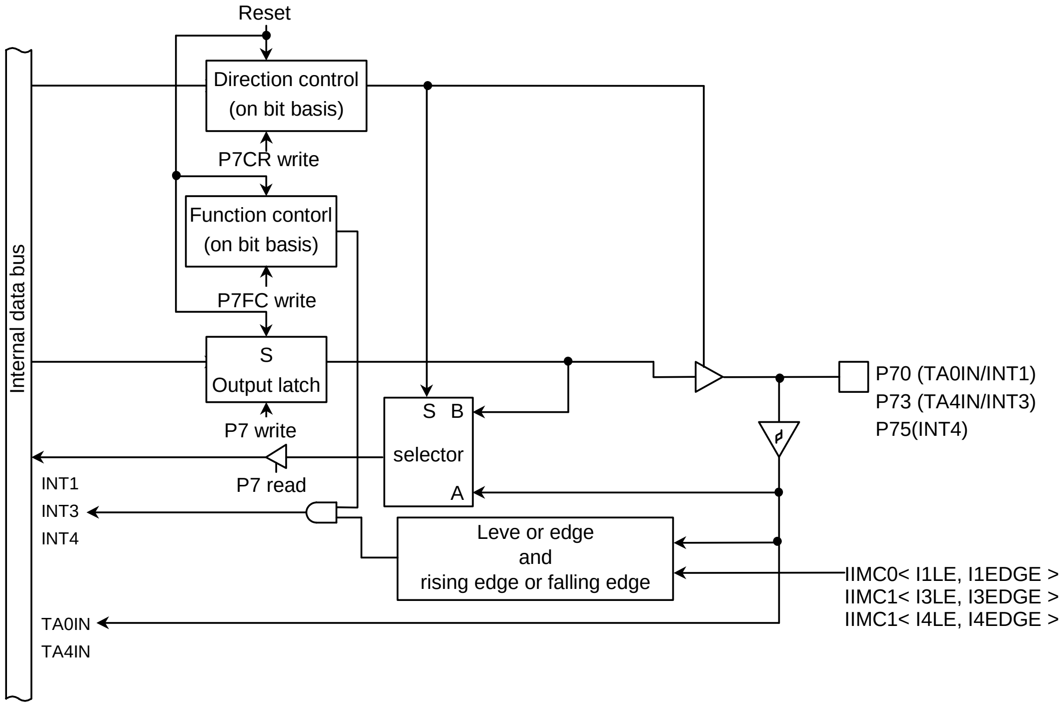


Figure 3.6.14 Port 70,73,75

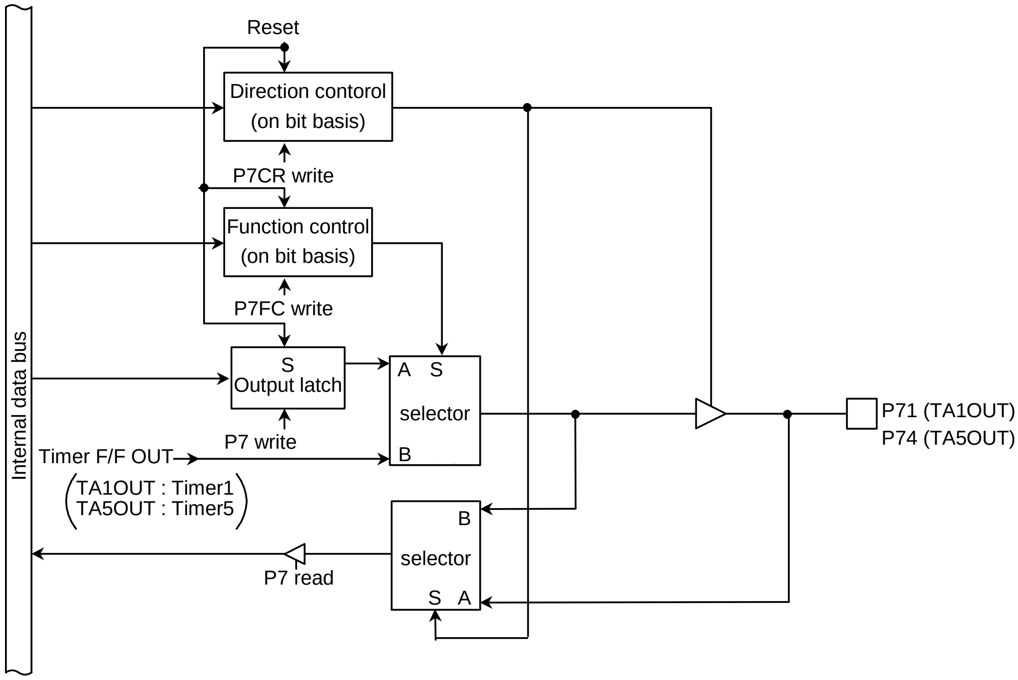


Figure 3.6.15 Port 71, 74

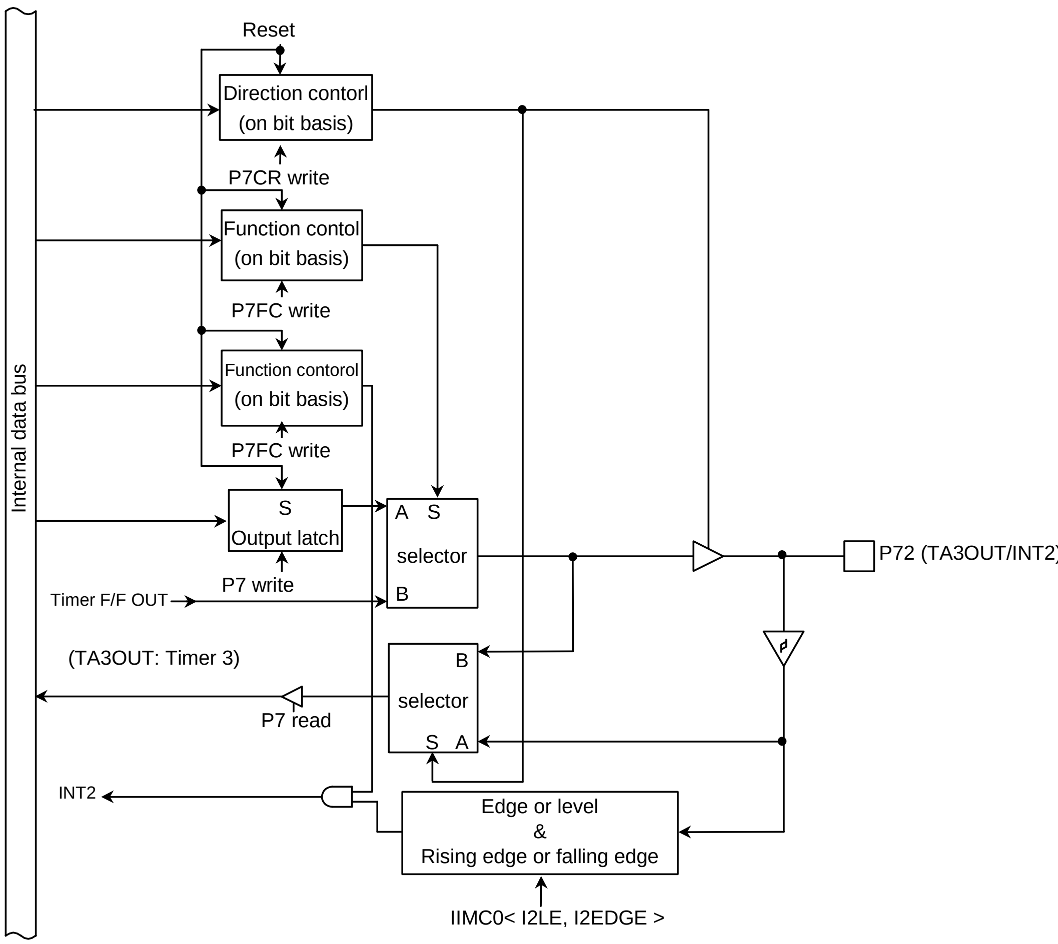


Figure 3.6.16 Port 72

Port 7 Register

		7	6	5	4	3	2	1	0
P7 (0013H)	Bit symbol			P75	P74	P73	P72	P71	P70
	Read/Write			R/W					
	After Reset			Input mode					
				1	1	1	1	1	1

Port 7 Control Register

		7	6	5	4	3	2	1	0
P7CR (0016H)	Bit symbol			P75C	P74C	P73C	P72C	P71C	P70C
	Read/Write			W					
	After Reset			0	0	0	0	0	0
				0: IN 1: OUT					

Port 7 I/O setting

0	Input
1	Output

Port 6 Function Register

		7	6	5	4	3	2	1	0
P7FC (0017H)	Bit symbol		P72F2	P75F	P74F	P73F	P72F1	P71F	P70F
	Read/Write		W	W		W	W		W
	After Reset		0	0	0	0	0	0	0
	Function		0: PORT 1: INT2 input	0: PORT 1: INT4 input	0: PORT 1: TA5OUT	0: PORT 1: INT3 input	0: PORT 1: TA3OUT	0: PORT 1: TA1OUT	0: PORT 1: INT1 input

Note: Read-Modify-Write is prohibited for the registers P7CR and P7FC.

Setting P71 as timer output1

P7FC<P71F>	1
P7CR<P71C>	1

Setting P72 as timer output3

P7FC<P72F1>	1
P7CR<P72C>	1

Setting P74 as timer output5

P7FC<P74F>	1
P7CR<P74C>	1

Figure 3.6.17 Port 7 registers

3.6.6 Port 8 (P80 to P87)

- Port pins 80 to 87

Port pins 80 to 87 constitute a 8-bit general-purpose I/O port. Each bit can be set individually for input or output. Resetting sets P80 to P87 to be an input port. It also sets all bits of the output latch register to 1.

In addition to functioning as general-purpose I/O port pins, P80 to P87 can also function as the I/O for serial channels 0.These function can be enabled for I/O by writing a 1 to the corresponding bit of the Port 8 Function Register (P8FC).

Resetting resets all bits of the registers P8CR and P8FC to 0 and sets all bits to be input port pins.(with pull-up resistors).

(1) Port pin 80 (TXD0), 84 (TXD1)

As well as functioning as I/O port pins, port pin 80, 84 can also function as serial channel TXD output pins.

These port pins feature a programmable open-drain function.

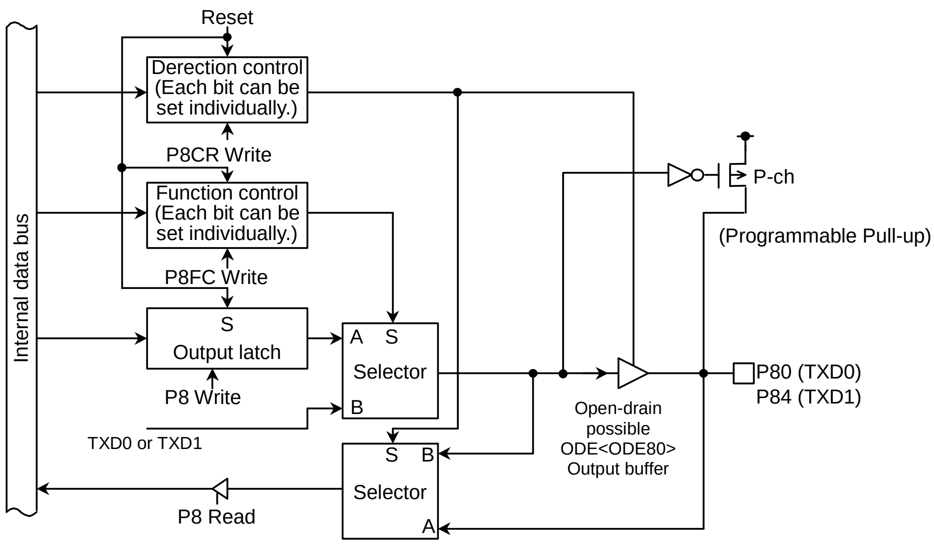


Figure 3.6.18 Port pins 80, 84

(2) Port pin 81 (RXD0), 85 (RXD1)

Port pin 81, 85 are I/O port pins and can also be used as RXD input pin for the serial channels.

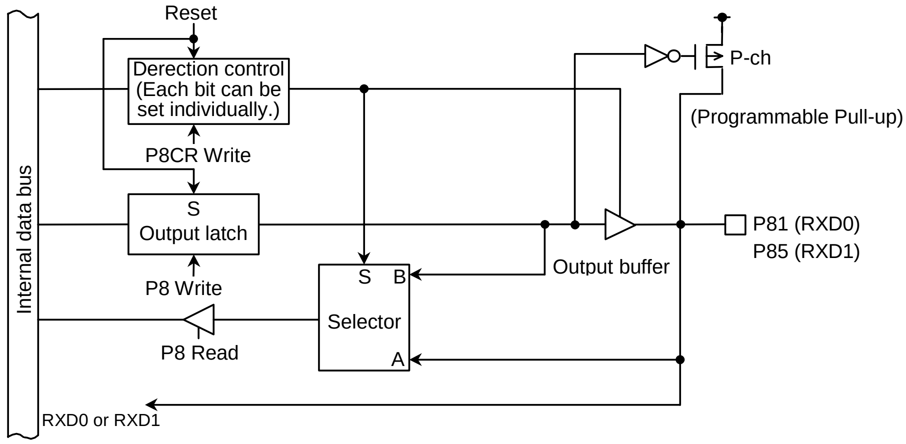


Figure 3.6.19 Port pins 81, 85

(3) Port pins 82 ($\overline{\text{CTS0}}$ /SCLK0), 86 ($\overline{\text{CTS1}}$/SCLK1)

Port pins 82, 86 are I/O port pins and can also be used as the $\overline{\text{CTS}}$ input pins or SCLK I/O pins for the serial channels.

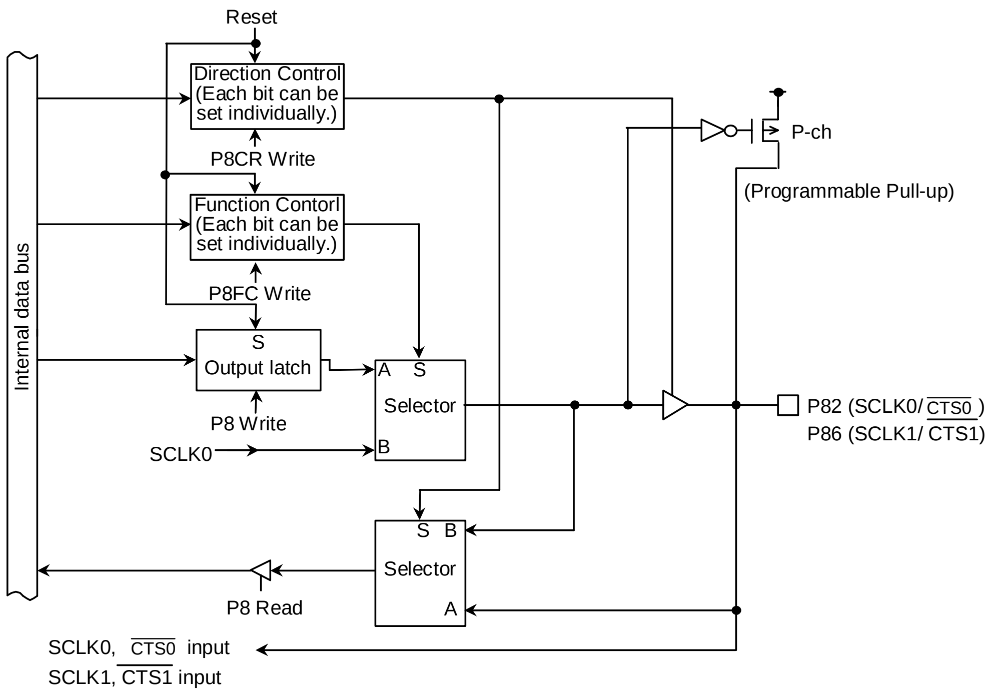


Figure 3.6.20 Ports 82, 86

(4) Port pin 83 (/STS0), 87 (/STS1)

Port pin 83, 87 are I/O port pins and can also be used as $\overline{\text{STS}}$ output pin for the received data request signal.

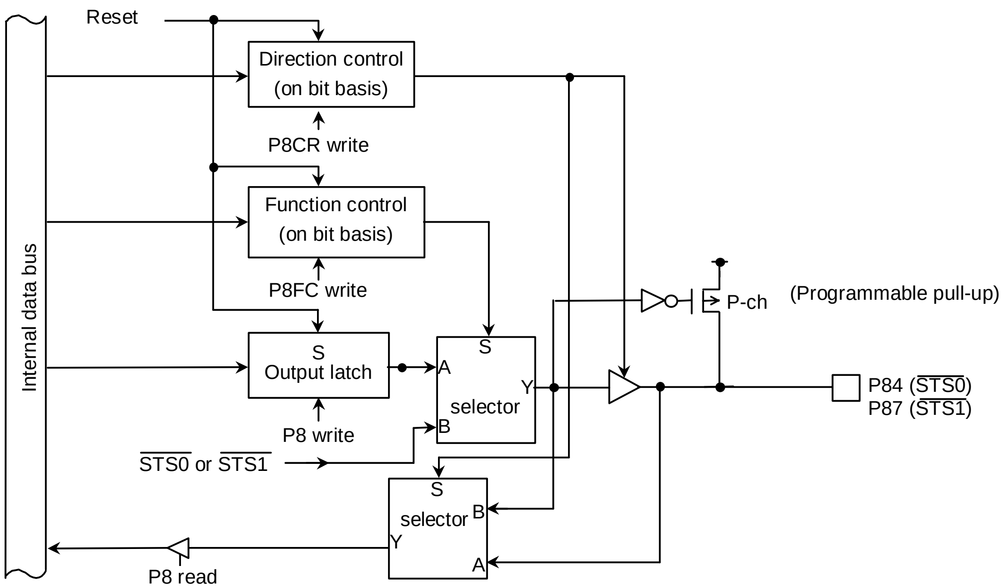


Figure 3.6.21 Port pin 84, 87

Port 8 Register

		7	6	5	4	3	2	1	0
P8 (0018H)	Bit symbol	P87	P86	P85	P84	P83	P82	P81	P80
	Read/Write	R/W							
	After Reset	Input Mode (with pull-up resistor)							
		1	1	1	1	1	1	1	1

Port 8 Control Register

		7	6	5	4	3	2	1	0
P8CR (001AH)	Bit symbol	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
	Read/Write	W							
	After Reset	0	0	0	0	0	0	0	0
		0: IN 1: OUT							

Port 8 I/O setting

0	Input
1	Output

Port 8 Function Register

		7	6	5	4	3	2	1	0
P8FC (001BH)	Bit symbol	P87F	P86F		P84F	P83F	P82F		P80F
	Read/Write	W	W		W	W	W		W
	After Reset	0	0		0	0	0		0
	Function	0: PORT 1: STS1 output	0: PORT 1: SCLK1 output		0: PORT 1: TXD1 output	0: PORT 1: STS0 output	0: PORT 1: SCLK0 output		0: PORT 1: TXD0 input

To set P80, 84 for TXD0, TXD1 output

P8FC<P80F><P84F>	1
P8CR<P80C><P84C>	1

To set P82, P86 for SCLK0, SCLK1 output

P8FC<P82F><P86F>	1
P8CR<P82C><P86C>	1

To set P83, P87 for STS0, STS1 output

P8FC<P83F><P87F>	1
P8CR<P83C><P87C>	1

Note 1: Read-Modify-Write is prohibited for the registers P8CR and P8FC.

Note 2: Writing 1 to bit 0 of the ODE register sets the TXD0 pin to be open-drain.

No register is provided for switching between the I/O port and RXD input functions of the P81/RXD0 pin. Hence, when Port 8 is used as an input port, the serial data input signals received on that pin are also input to the SIO.

Figure 3.6.22 Port 8 register

3.6.7 Port 9 (P90, P93 to P96)

Port 9 is an 8-bit general-purpose I/O port. Each bit can be set individually for input or output, Resetting sets port9 to be an input port,It also sets all bits in the output latch register P9 to 1.In addtion to functioning as a general-purpose I/O port, the various pins of Port 9 can also function as the clock input for the 16-bit timer flipflop putput,on as input INT5.These functions cn be enabled by writing a 1 to the corresponding bits in the Port 9 function registers(P9FC).

(1) P90

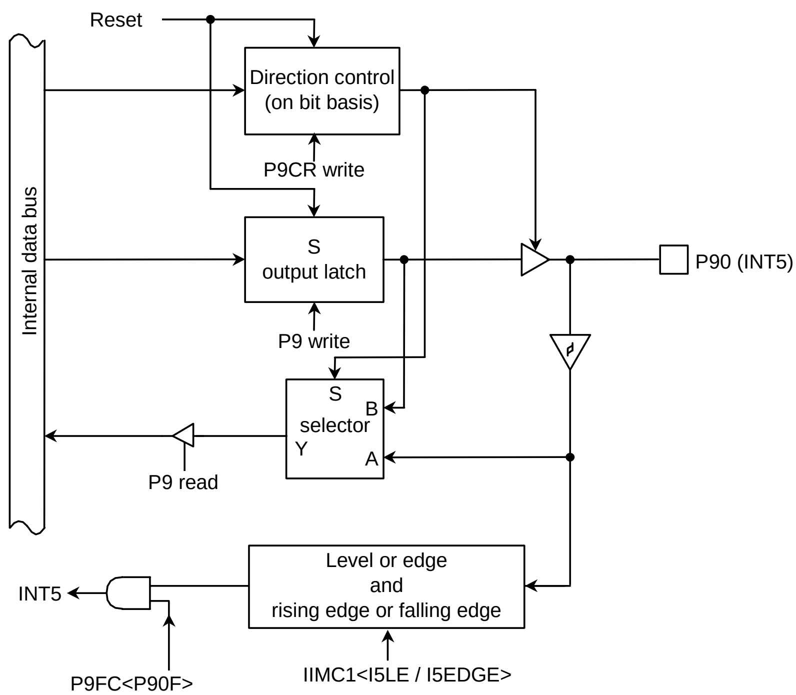


Figure 3.6.23 Port 90

(2) P93 to P96

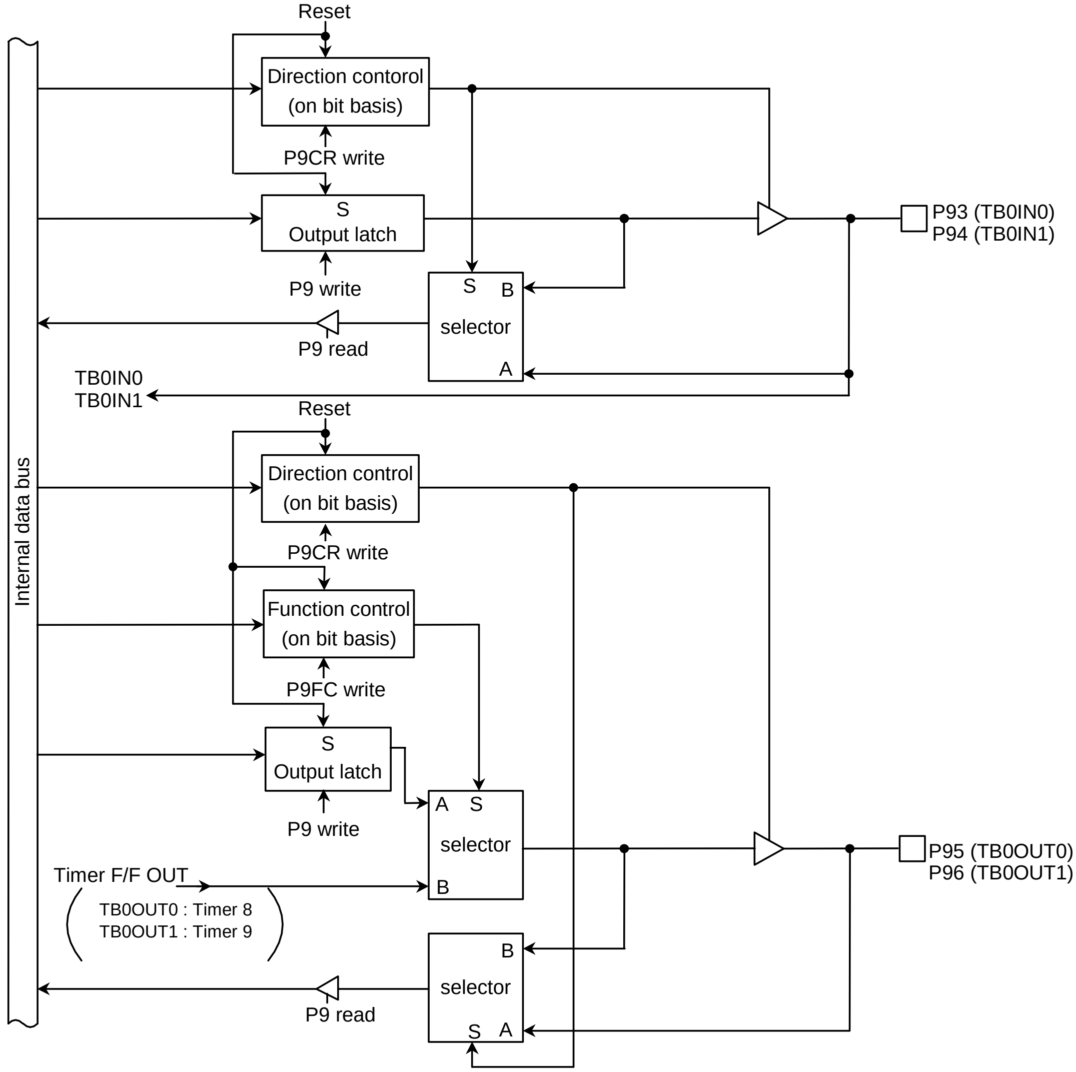


Figure 3.6.24 Port pins P93 to P96

Port 9 Register

P9 (0019H)

	7	6	5	4	3	2	1	0
Bit symbol		P96	P95	P94	P93			P90
Read/Write	R/W							
After Reset	Input Mode							
		1	1	1	1			1

Port 9 Control Register

P9CR (001CH)

	7	6	5	4	3	2	1	0
Bit symbol		P96C	P95C	P94C	P93C			P90C
Read/Write	W							
After Reset		0	0	0	0			0
	0: IN 1: OUT							

Port 9 I/O setting

0	Input
1	Output

Port 9 Function Register

P9FC (001DH)

	7	6	5	4	3	2	1	0
Bit symbol		P96F	P95F					P90F
Read/Write		W	W					W
After Reset		0	0					0
Function		0: PORT 1: TB0OUT1	0: PORT 1: TB0OUT0					0: PORT 1: INT5 input

To set P95 for Timer 8 output

1	P9FC<P95F>
1	P9CR<P95C>

To set P96 for Timer 9 output

1	P9FC<P96F>
1	P9CR<P96C>

Note: Read-Modify-Write is prohibited for the registers P9CR and P9FC.

Figure 3.6.25 Port 9 registers

3.6.8 Port A (PA0 to PA7)

Port A is an 8-bit input port and can also be used as the analog input pins for the internal AD converter.

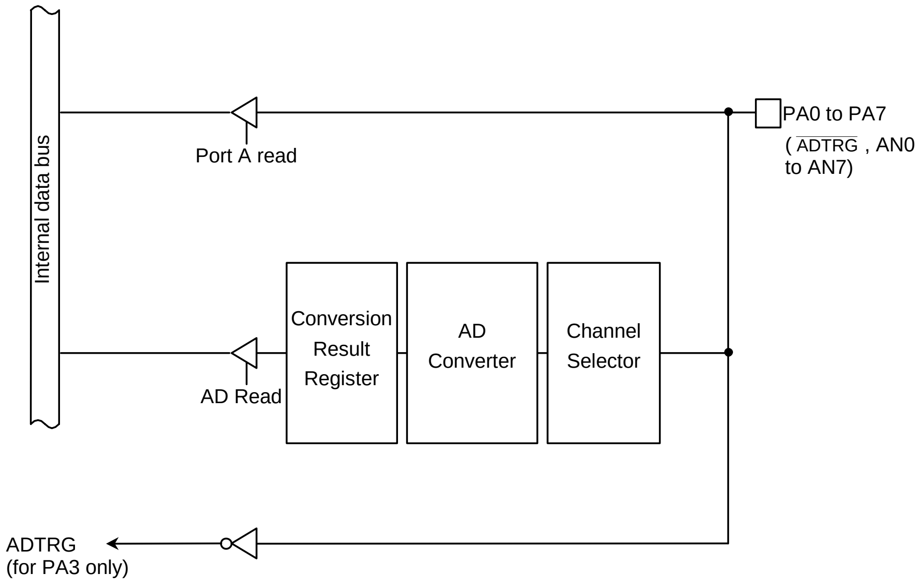


Figure 3.6.26 Port A

PortA Register

		7	6	5	4	3	2	1	0
PA (0019H)	Bit symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
	Read/Write	R							
	After Reset	Input Mode							

Note: The input channel selection of AD Converter and the permission of ADTRG input are set by AD Converter mode register ADMOD1.

Figure 3.6.27 Port A Register

3.6.9 Port Z (PZ2, PZ3)

Port Z is a 4-bit general-purpose I/O port. I/O is set using control register PZCR and PZFC. Resetting resets all bits of the output latch PZ to 1, the control register PZCR and the function register PZFC to 0 and sets PZ2 and PZ3 to input mode with pull-up register.

In addition to functioning as a general-purpose I/O port. Port Z also functions as I/O for the CPU's control /status signal.

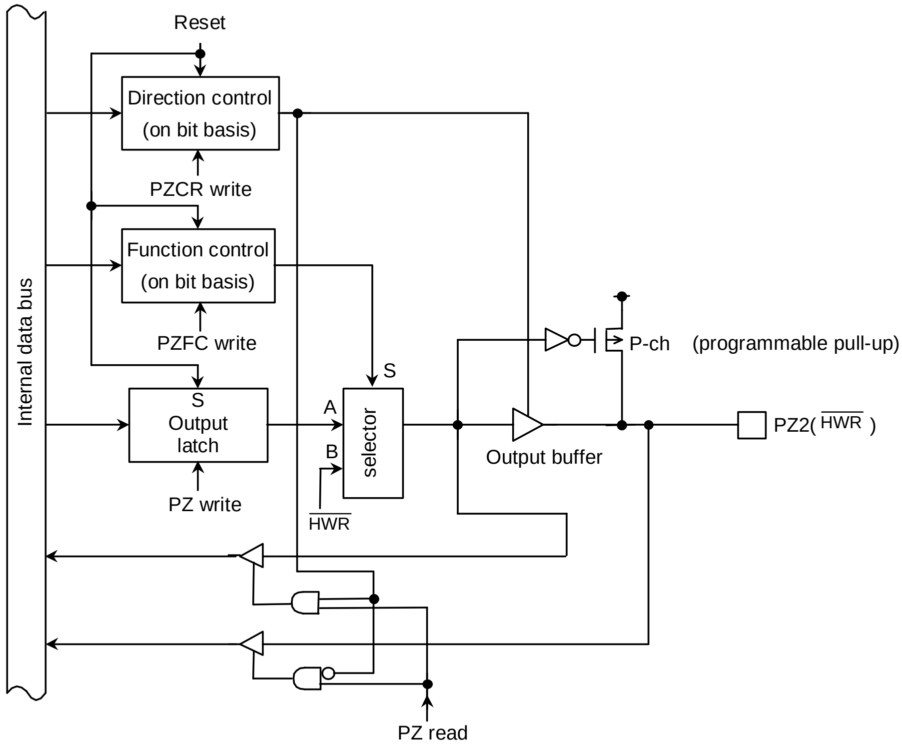


Figure 3.6.28 Port Z2

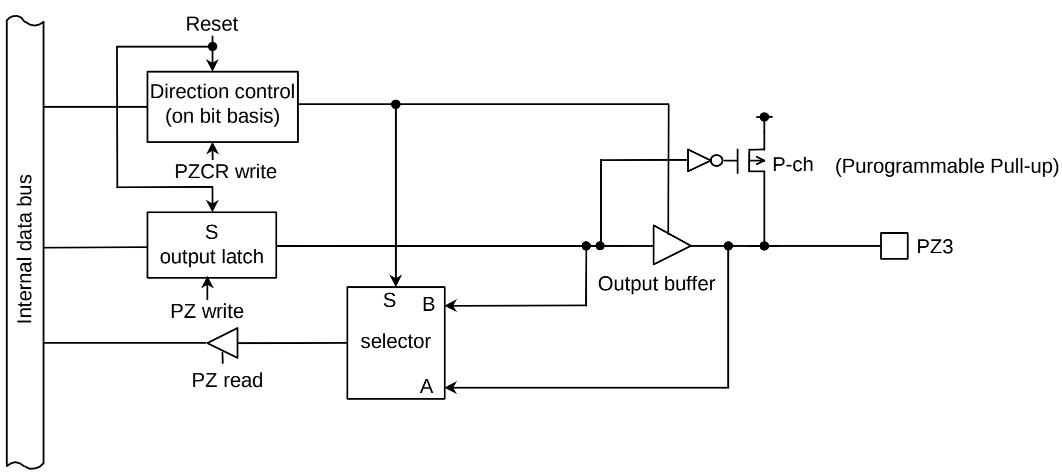


Figure 3.6.29 Port Z3

Port Z Register

<table>
<tr><td rowspan="5">PZ
(007DH)</td><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Bit symbol</td><td></td><td></td><td></td><td></td><td>PZ3</td><td>PZ2</td><td></td><td></td></tr>
<tr><td>Read/Write</td><td colspan="4">W</td><td colspan="2">R/W</td><td></td><td></td></tr>
<tr><td rowspan="2">After Reset</td><td colspan="4" rowspan="2"></td><td colspan="2">Input mode</td><td rowspan="2"></td><td rowspan="2"></td></tr>
<tr><td>1</td><td>1</td></tr>
</table>

Port Z control register

<table>
<tr><td rowspan="5">PZCR
(007EH)</td><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Bit symbol</td><td></td><td></td><td></td><td></td><td>PZ3</td><td>PZ2</td><td></td><td></td></tr>
<tr><td>Read/Write</td><td colspan="4"></td><td colspan="2">W</td><td></td><td></td></tr>
<tr><td rowspan="2">After Reset</td><td></td><td></td><td></td><td></td><td>0</td><td>0</td><td></td><td></td></tr>
<tr><td colspan="4"></td><td colspan="2">0: IN 1: OUT</td><td></td><td></td></tr>
</table>

Setting port Z as I/O

0	Input
1	Output

Port Z control register

<table>
<tr><td rowspan="5">PZFC
(007FH)</td><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td>Bit symbol</td><td></td><td></td><td></td><td></td><td></td><td>PZ2F</td><td></td><td></td></tr>
<tr><td>Read/Write</td><td colspan="4"></td><td></td><td>W</td><td></td><td></td></tr>
<tr><td>After Reset</td><td></td><td></td><td></td><td></td><td></td><td>0</td><td></td><td></td></tr>
<tr><td>Function</td><td colspan="4"></td><td></td><td>0: PORT
1: HWR</td><td></td><td></td></tr>
</table>

Figure 3.6.30 Port Z registers

3.7 Chip Select/Wait Controller

On the TMP91C829, four user-specifiable address areas (CS0 to CS3) can be set. The data bus width and the number of waits can be set independently for each address area (CS0 to CS3 plus any other).

The pins $\overline{CS0}$ to $\overline{CS3}$ (which can also function as port pins P60 to P63) are the respective output pins for the areas CS0 to CS3. When the CPU specifies an address in one of these areas, the corresponding $\overline{CS0}$ to $\overline{CS3}$ pin outputs the Chip Select signal for the specified address area (in ROM or SRAM). However, in order for the Chip Select signal to be output, the Port 6 Function Register P6FC must be set. External connection of ROM and SRAM is supported.

The areas CS0 to CS3 are defined by the values in the Memory Start Address Registers MSAR0 to MSAR3 and the Memory Address Mask Registers MAMR0 to MAMR3.

The Chip Select/Wait Control Registers B0CS to B3CS and BEXCS should be used to specify the Master Enable/Disable status the data bus width and the number of waits for each address area.

The input pin which controls these states is the Bus Wait Request pin ($\overline{WAIT}$).

3.7.1 Specifying an Address Area

The address areas CS0 to CS3 are specified using the Memory Start Address Registers (MSAR0 to MSAR3) and the Memory Address Mask Registers (MAMR0 to MAMR3).

During each bus cycle, a compare operation is performed to determine whether or not the address specified on the bus corresponds to a location in one of the areas CS0 to CS3. If the result of the comparison is a match, it indicates that the corresponding CS area is to be accessed. If so, the corresponding $\overline{CS0}$ to $\overline{CS3}$ pin outputs the Chip Select signal and the bus cycle proceeds according to the settings in the corresponding B0CS to B3CS chip select/wait control register. (See 3.7.2, Chip Select/Wait Control Registers.)

(1) Memory Start Address Registers

Figure 3.7.1 shows the Memory Start Address Registers. The Memory Start Address Registers MSAR0 to MSAR3 determine the start addresses for the memory areas CS0 to CS3 respectively. The eight most significant bits (A23 to A16) of the start address should be set in <S23 to S16>. The 16 least significant bits of the start address (A15 to A0) are fixed to 0. Thus the start address can only be set to lie on a 64-Kbyte boundary, starting from 000000H. Figure 3.7.2 shows the relationship between the value set in the start address register and the start address.

Memory Start Address Registers (for areas CS0 to CS3)

MSAR0 (00C8H) / MSAR1 (00CAH)
MSAR2 (00CCH)/ MSAR3 (00CEH)

	7	6	5	4	3	2	1	0
Bit symbol	S23	S22	S21	S20	S19	S18	S17	S16
Read/Write	R/W							
After Reset	1	1	1	1	1	1	1	1
Function	Determines A23 to A16 of start address.							

→ Sets start addresses for areas CO0 to CS3.

Figure 3.7.1 Memory Start Address Register

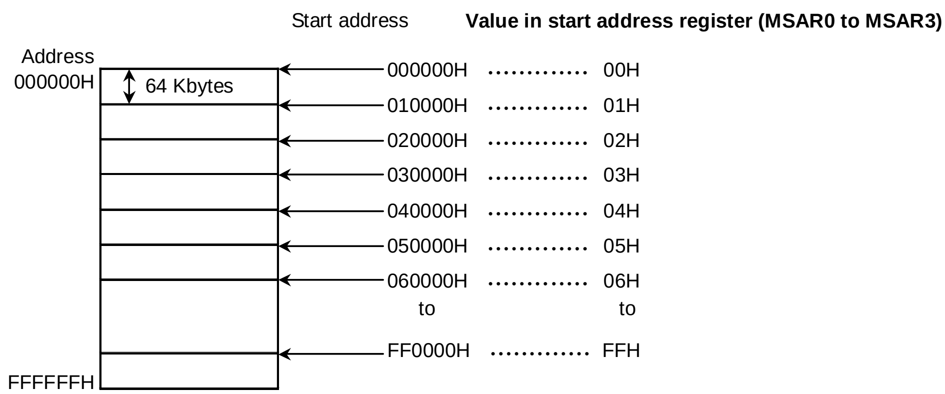


Figure 3.7.2 Relationship Between Start Address and start Address Register Value

(2) Memory Address Mask Registers

Figure 3.7.3 shows the Memory Address Mask Registers. The size of each of the areas CS0 to CS3 can be set by specifying a mask in the corresponding memory address mask register (MAMR0 to MAMR3). Each bit in a memory address mask register (MAMR0 to MAMR3) which is set to 1 masks the corresponding bit of the start address which has been set in the corresponding memory start address register (MSAR0 to MSAR3). The compare operation used to determine whether or not a bus address is in one of the areas CS0 to CS3 only compares address bits for which a 0 has been set in the corresponding bit position in the corresponding memory address mask register.

Also, the address bits which each memory address mask register can mask vary from register to register; hence, the possible size settings for the areas CS0 to CS3 differ accordingly.

Memory address mask register (for CS0 area)

		7	6	5	4	3	2	1	0
MAMR0 (00C9H)	Bit symbol	V20	V19	V18	V17	V16	V15	V14 to 9	V8
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
	Function	Sets size of CS0 area 0: used for address compare							

Range of possible settings for CS0 area size: 256 bytes to 2 Mbytes.

Memory address mask register (CS1)

		7	6	5	4	3	2	1	0
MAMR1 (00CBH)	Bit symbol	V21	V20	V19	V18	V17	V16	V15 to 9	V8
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
	Function	Sets size of CS0 area 0: used for address compare							

Range of possible settings for CS1 area size: 256 bytes to 4M bytes.

Memory address mask register (CS2, CS3)

		7	6	5	4	3	2	1	0
MAMR2 (00CDH) / MAMR3 (00CFH)	Bit symbol	V22	V21	V20	V19	V18	V17	V16	V15
	Read/Write	R/W							
	After Reset	1	1	1	1	1	1	1	1
	Function	Sets size of CS2 or CS3 area 0: used for address compare							

Range of possible settings for CS2 and CS3 area sizes: 32 Kbytes to 8 Mbytes.

Figure 3.7.3 Memory Address mask Registers

(3) Setting Memory Start Addresses and Address Areas

Figure 3.7.4 shows an exa to ple in which CS0 is specified to be a 64-Kbyte address area starting at 010000H.

First, MSAR0<S23 to S16>, the eight most significant bits of the start address register and which correspond to the memory start address, are set to 01H. Next, based on the desired CS0 area size, the difference between the start address and the end address (01FFFFH) is calculated. Bits 20 to 8 of this result constitute the mask value for the desired CS0 area size. Setting this value in MAMR0<V20 to V8> (bits 20 to 8 of the memory address mask register) sets the desired area size for CS0. In this example 07H is set in MAMR0, specifying an area size of 64 Kbytes.

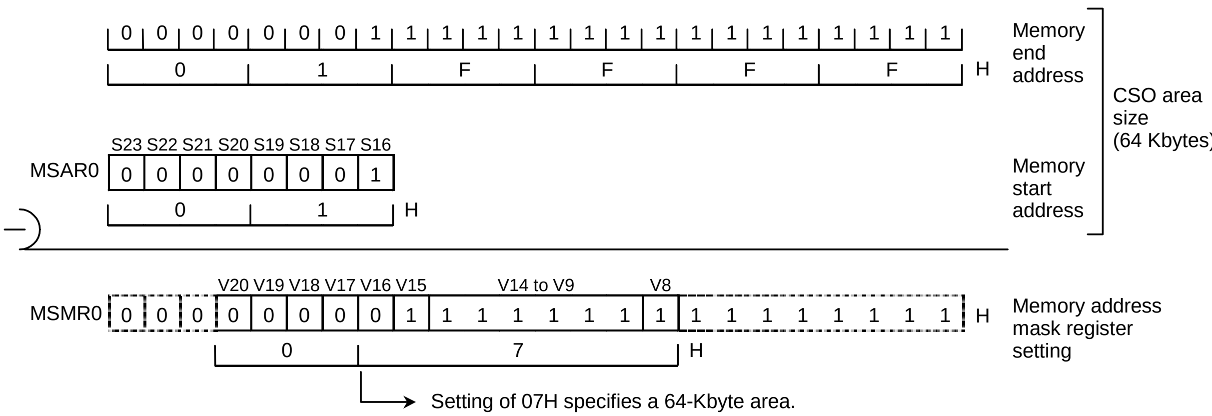


Figure 3.7.4 Example showing how to set the CS0 area

A Reset sets MSAR0 to MSAR3 and MAMR0 to MAMR3 to FFH. In addition, B0CS<B0E>, B1CS<B1E> and B3CS<B3E> are reset to 0, disabling the CS0, CS1 and CS3 areas. However, since a Reset resets B2CS<B2M> to 0 and sets B2CS<B2E> to 1, CS2 is enabled with the address range 001800H to 001F7FFH, 020000H to FFFFFFH. When addresses outside the areas specified as CS0 to CS3 are accessed, the bus width and number of waits specified in BEXCS are used. (See 3.6.2, Chip Select/Wait Control Registers.)

(4) Address Area Size Specification

Table 3.7.1 shows the valid area sizes for each CS area and indicates which method can be used to make the size setting. A Δ indicates that it is not possible to set the area size in question using the memory start address register and memory address mask register. If an area size for a CS area marked Δ in the table is to be set, the start address must either be set to 000000H or to a value that is greater than 000000H by an integer multiple of the desired area size.

If the CS2 area is set to 16 Mbytes or if two or more areas overlap, the lowest-numbered CS area has highest priority (e.g. CS0 has a higher priority than any other area).

Example: To set the area size for CS0 to 128 Kbytes:

① Valid start addresses

000000H
128K bytes
020000H
128K bytes
040000H
128K bytes
060000H

Any of these addresses may be set as the start address.

② Invalid start addresses

000000H
64K bytes
010000H
128K bytes
030000H
128K bytes
050000H

This is not an integer multiple of the desired area size setting. Hence, none of these addresses can be set as the start address.

Table 3.7.1 Valid area sizes for each CS area

CS area \ Size (bytes)	256	512	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M
CS0	○	○	○	○	Δ	Δ	Δ	Δ	Δ		
CS1	○	○		○	Δ	Δ	Δ	Δ	Δ	Δ	
CS2			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ
CS3			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ

3.7.2 Chip Select/Wait Control Registers

Figure 3.7.5 lists the Chip Select/Wait Control Registers.

The Master Enable/Disable, Chip Select output waveform, data bus width and number of wait states for each address area (CS0 to CS3 plus any other) are set in the respective Chip Select/Wait Control Registers, B0CS to B3CS or BEXCS.

Chip Select/Wait Control Register

		7	6	5	4	3	2	1	0
B0CS (00C0H)	Bit symbol	B0E		B0OM1	B0OM0	B0BUS	B0W2	B0W1	B0W0
	Read/Write	W		W					
Read-Modify-Write instructions are prohibited.	After Reset	0		0	0	0	0	0	0
	Function	0: Disable 1: Enable		Chip Select output waveform selection 00: For ROM/SRAM 01: 10: Don't care 11:		Data bus width 0: 16 bits 1: 8 bits	Number of Waits 000: 2 waits 001: 1 wait 010: 1 wait + N 011: 0 waits	1xx: Reserved	
B1CS (00C1H)	Bit symbol	B1E		B1OM1	B1OM0	B1BUS	B1W2	B1W1	B1W0
	Read/Write	W		W					
Read-Modify-Write instructions are prohibited.	After Reset	0		0	0	0	0	0	0
	Function	0: Disable 1: Enable		Chip Select output waveform selection 00: For ROM/SRAM 01: 10: Don't care 11:		Data bus width 0: 16 bits 1: 8 bits	Number of Waits 000: 2 waits 001: 1 wait 010: 1 wait + N 011: 0 waits	1xx: Reserved	
B2CS (00C2H)	Bit symbol	B2E	B2M	B2OM1	B2OM0	B2BUS	B2W2	B2W1	B2W0
	Read/Write	W							
Read-Modify-Write instructions are prohibited.	After Reset	1	0	0	0	0	0	0	0
	Functions	0: Disable 1: Enable	CS2 area selection 0: 16-Mbyte area 1: CS area	Chip Select output waveform selection 00: For ROM/SRAM 01: 10: Don't care 11:		Data bus width 0: 16 bits 1: 8 bits	Number of waits 000: 2 waits 001: 1 wait 010: 1 wait + N 011: 0 waits	1xx: Reserved	
B3CS (00C3H)	Bit symbol	B3E		B3OM1	B3OM0	B3BUS	B3W2	B3W1	B3W0
	Read/Write	W		W					
Read-Modify-Write instructions are prohibited.	After Reset	0		0	0	0	0	0	0
	Functions	0: Disable 1: Enable		Chip Select output waveform selection 00: For ROM/SRAM 01: 10: Don't care 11:		Data bus width 0: 16 bits 1: 8 bits	Number of waits 000: 2 waits 001: 1 wait 010: 1 wait + N 011: 0 waits	1xx: Reserved	
BEXCS (00C7H)	Bit symbol					BEXBUS	BEXW2	BEXW1	BEXW0
	Read/Write						W		
Read-Modify-Write instructions are prohibited.	After Reset					0	0	0	0
	Functions					Data bus width 0: 16 bits 1: 8 bits	Number of Waits 000: 2 waits 001: 1 wait 010: 1 wait + N 011: 0 waits	1xx: Reserved	

Master enable bit

0	CS area disable
1	CS area enable

CS2 area selection

0	16-Mbyte area
1	Specified address area

Chip select output waveform selection

00	For ROM/SRAM
01	
10	Don't care
11	

Number of address area waits (See 3.7.2, (3) Wait Control.)

Data bus width selection

0	16-bit data bus
1	8-bit data bus

Figure 3.7.5 Chip Select/Wait Control Registers

(1) Master Enable bits

Bit 7 (<B0E>, <B1E>, <B2E> or <B3E>) of a chip select/wait control register is the master bit which is used to enable or disable settings for the corresponding address area. Writing 1 to this bit enables the settings. A Reset disables <B0E>, <B1E> and <B3E> (i.e sets them to 0) and enables <B2E> (i.e. sets it to 1). Hence after a Reset only the CS2 area is enabled.

(2) Data bus width selection

Bit 3 (<B0BUS>, <B1BUS>, <B2BUS>, <B3BUS> or <BEXBUS>) of a chip select/wait control register specifies the width of the data bus. This bit should be set to 0 when memory is to be accessed using a 16-bit data bus, and to 1 when an 8-bit data bus is to be used.

This process of changing the data bus width according to the address being accessed is known as dynamic bus sizing. For details of this bus operation see Figure 3.7.2.

Table 3.7.2 Dynamic bus sizing

Operand Data Bus Width	Operand Start Address	Memory Data Bus Width	CPU Address	CPU Data	
				D15 to D8	D7 to D0
8 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
		16 bits	2n + 0	xxxxx	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
		16 bits	2n + 1	b7 to b0	xxxxx
16 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
			2n + 1	xxxxx	b15 to b8
		16 bits	2n + 0	b15 to b8	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
			2n + 2	xxxxx	b15 to b8
		16 bits	2n + 1	b7 to b0	xxxxx
			2n + 2	xxxxx	b15 to b8
32 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
			2n + 1	xxxxx	b15 to b8
			2n + 2	xxxxx	b23 to b16
			2n + 3	xxxxx	b31 to b24
		16 bits	2n + 0	b15 to b8	b7 to b0
			2n + 2	b31 to b24	b23 to b16
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
			2n + 2	xxxxx	b15 to b8
			2n + 3	xxxxx	b23 to b16
			2n + 4	xxxxx	b31 to b24
		16 bits	2n + 1	b7 to b0	xxxxx
			2n + 2	b23 to b16	b15 to b8
			2n + 4	xxxxx	b31 to b24

Input data in bit positions marked xxxxx is ignored during a read. During a write, the bus lines corresponding to these bit positions go High-Impedance and the Write Strobe signal for the bus remains Inactive.

(3) Wait control

Bits 0 to 2 (<B0W0 to B0W2>, <B1W0 to B1W2>, <B2W0 to B2W2>, <B3W0 to B3W2> or <BEXW0 to BEXW2>) of a chip select/wait control register specify the number of waits that are to be inserted when the corresponding memory area is accessed.

The following types of wait operation can be specified using these bits. Bit settings other than those listed in the table should not be made.

Table 3.7.3 Wait operation settings

<BxW2 to BxW0>	No. of Waits	Wait Operation
000	2WAIT	Inserts a wait of two states, irrespective of the $\overline{\text{WAIT}}$ pin state.
001	1WAIT	Inserts a wait of one state, irrespective of the $\overline{\text{WAIT}}$ pin state.
010	1WAIT + N	Inserts one wait state, then continuously samples the state of the $\overline{\text{WAIT}}$ pin. While the $\overline{\text{WAIT}}$ pin remains Low, the wait continues; the bus cycle is prolonged until the pin goes High.
011	0WAIT	Ends the bus cycle without a wait, regardless of the $\overline{\text{WAIT}}$ pin state.
1xx	Reserved	Do not set.

A Reset sets these bits to 000 (2 waits).

(4) Bus width and wait control for an area other than CS0 to CS3

The chip select/wait control register BEXCS controls the bus width and number of waits when memory locations which are not in one of the four user-specified address areas (CS0 to CS3) are accessed. The BEXCS register settings are always enabled for areas other than CS0 to CS3.

(5) Selecting 16-Mbyte area/specified address area

Setting B2CS<B2M> (bit 6 of the chip select/wait control register for CS2) to 0 designates the 16-Mbyte area 001800H to 001F7FFH, 020000H to FFFFFFH as the CS2 area. Setting B2CS<B2M> to 1 designates the address area specified by the start address register MSAR2 and the address mask register MAMR2 as CS2 (i.e. if B2CS<B2M> = 1, CS2 is specified in the same manner as CS0, CS1 and CS3 are).

A Reset clears this bit to 0, specifying CS2 as a 16-Mbyte address area.

(6) Procedure for setting chip select/wait control

When using the chip select/wait control function, set the registers in the following order:

① Set the Memory Start Address Registers MSAR0 to MSAR3.

Set the start addresses for CS0 to CS3.

② Set the Memory Address Mask Registers MAMR0 to MAMR3.

Set the sizes of CS0 to CS3.

③ Set the chip select/wait control registers B0CS to B3CS.

Set the Chip Select output waveform, data bus width, number of waits and Master Enable/Disable status for $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$.

The CS0 to CS3 pins can also function as pins P60 to P63. To output a Chip Select signal using one of these pins, set the corresponding bit in the Port 6 Function Register P6FC to 1.

If a CS0 to CS3 address is specified which is actually an internal I/O, RAM or ROM area address, the CPU accesses the internal address area and no Chip Select signal is output on any of the $\overline{\text{CS0}}$ to $\overline{\text{CS3}}$ pins.

Setting example:

In this example CS0 is set to be the 64-Kbyte area 010000H to 01FFFFH. The bus width is set to 16 bits and the number of waits is set to 0.

MSAR0 = 01H .... Start address: 010000H

MAMR0 = 07H .... Address area: 64 Kbytes

B0CS = 83H............... ROM/SRAM, 16-bit data bus, zero waits, CS0 area settings enabled

3.7.3 Connecting external memory

Figure 3.7.6 shows an example of how to connect external memory to the TMP91C829.

In this example the ROM is connected using a 16-bit bus. The RAM and I/O are connected using an 8-bit bus.

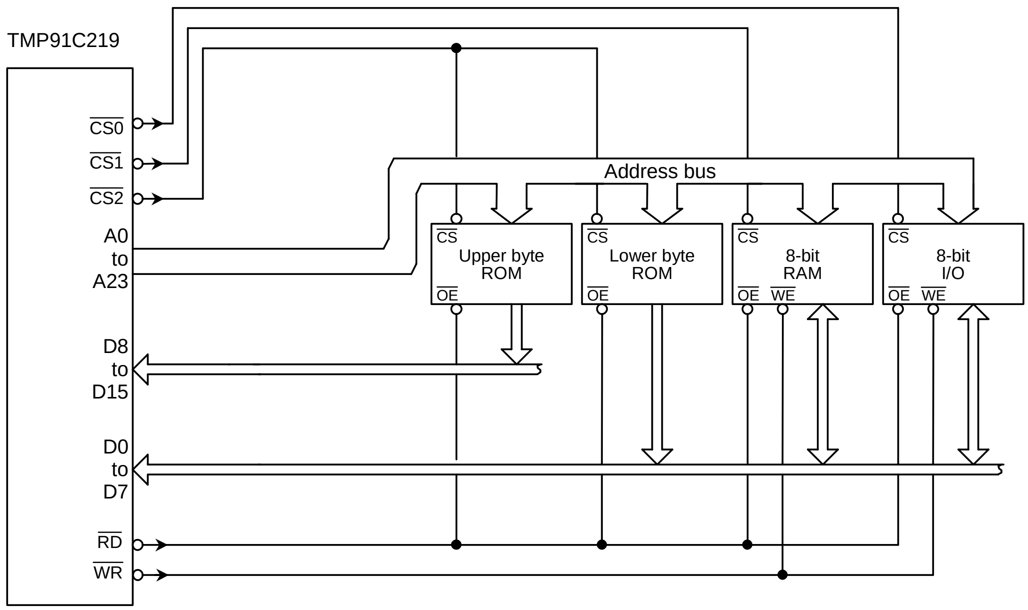


Figure 3.7.6 Example of external memory connection
(ROM uses 16-bit bus; RAM and I/O use 8-bit bus.)

A Reset clears all bits of the Port 4 Control Register P6CR and the Port 6 Function Register P6FC to 0 and disables output of the CS signal. To output the CS signal, the appropriate bit must be set to 1.

3.8 8-bit Timers (TMRA)

The TMP91C829 features six built-in 8-bit timers.

These timers are paired into three modules: TMRA01, TMRA23 and TMRA45. Each module consists of two channels and can operate in any of the following four operating modes.

- 8-Bit Interval Timer Mode
- 16-Bit Interval Timer Mode
- 8-Bit Programmable Square Wave Pulse Generation Output Mode (PPG – variable duty cycle with variable period)
- 8-Bit Pulse Width Modulation Output Mode (PWM – variable duty cycle with constant period)

Figure 3.8.1 to 3.8.3 show block diagrams for TMRA01, TMRA23 and TMRA45.

Each channel consists of an 8-bit up-counter, an 8-bit comparator and an 8-bit timer register. In addition, a timer flip-flop and a prescaler are provided for each pair of channels.

The operation mode and timer flip-flops are controlled by five control SFRs (special-function registers).

Each of the four modules (TMRA01, TMRA23 and TMRA45) can be operated independently. All modules operate in the same manner; hence only the operation of TMRA01 is explained here.

Table 3.8.1 Registers and pins for each module

	Module	TMRA01	TMRA23	TMRA45
External pin	Input pin for external clock	TA0IN (shared with P70)	No	TA4IN (shared with P73)
	Output pin for timer flip-flop	TA1OUT (shared with P71)	TA3OUT (shared with P72)	TA5OUT (shared with P74)
SFR (address)	Timer run register	TA01RUN (0100H)	TA23RUN (0108H)	TA45RUN (0110H)
	Timer register	TA0REG (0102H) TA1REG (0103H)	TA2REG (010AH) TA3REG (010BH)	TA4REG (0112H) TA5REG (0113H)
	Timer mode register	TA01MOD (0104H)	TA23MOD (010CH)	TA45MOD (0114H)
	Timer flip-flop control register	TA1FFCR (0105H)	TA3FFCR (010DH)	TA5FFCR (0115H)

3.8.1 Block diagrams

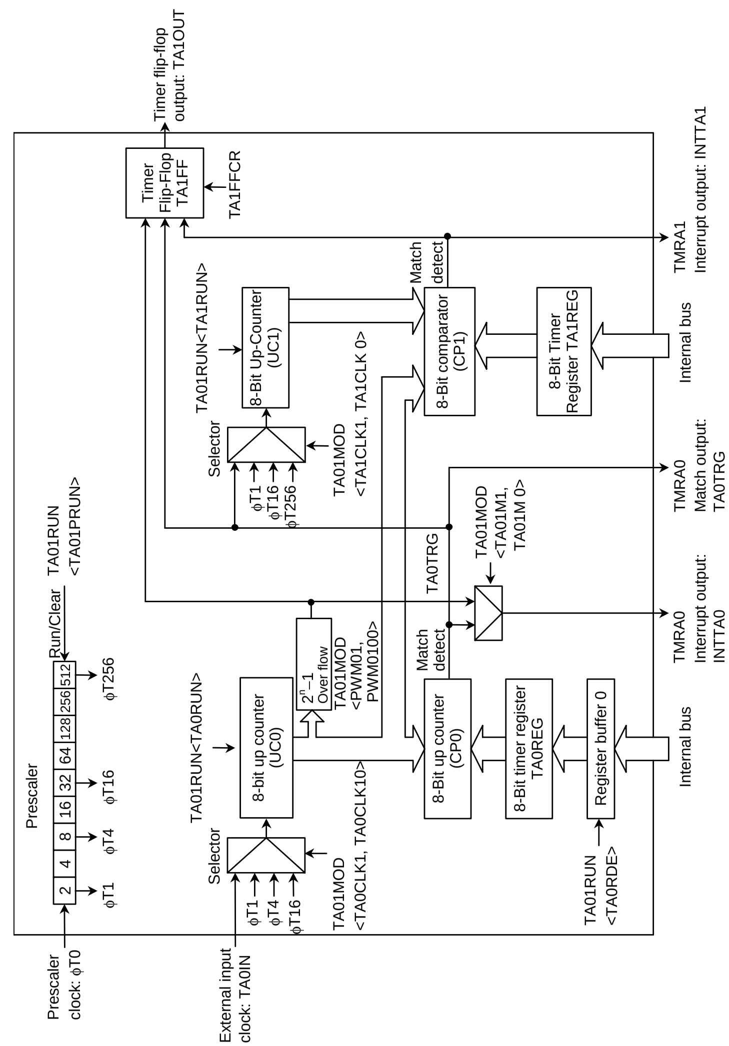


Figure 3.8.1 TMRA01 block diagram

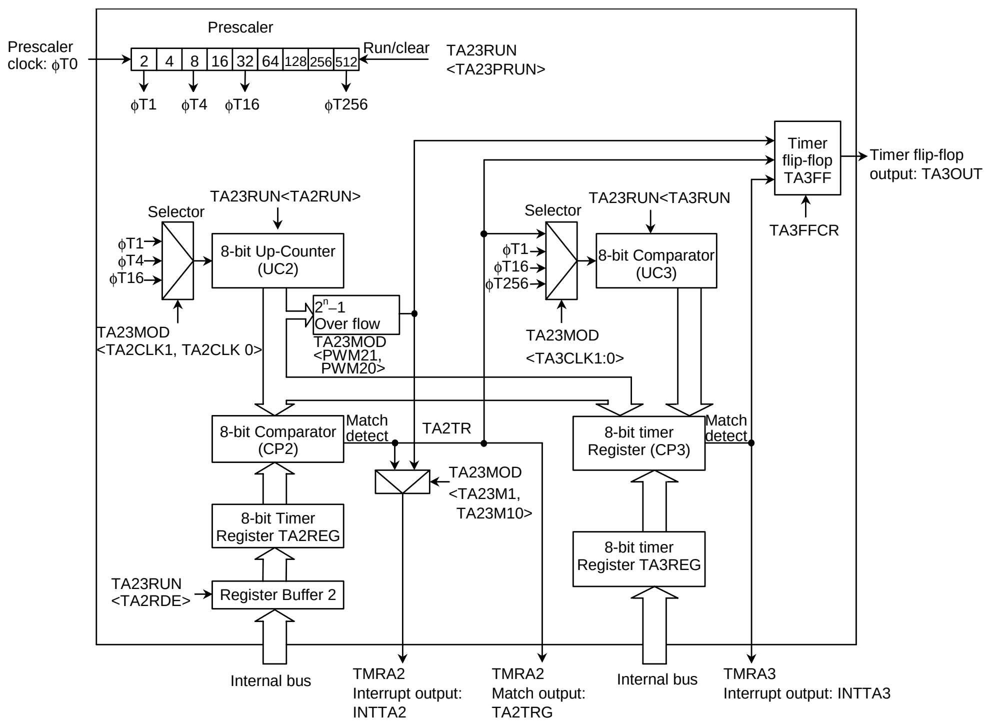


Figure 3.8.2 TMRA23 block diagram

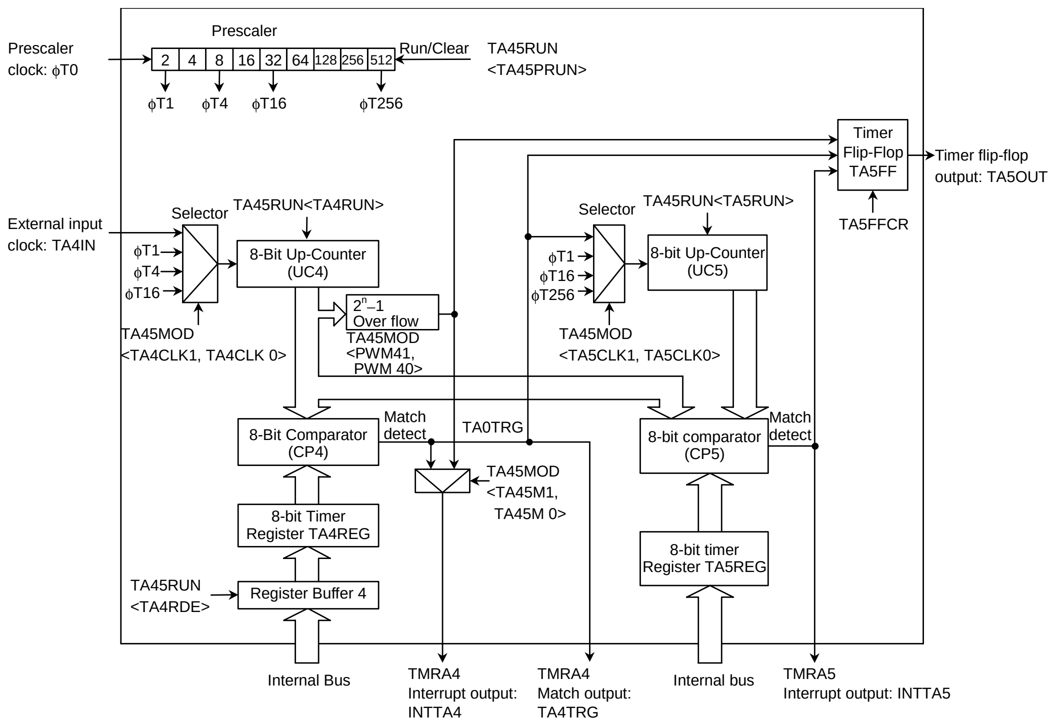


Figure 3.8.3 TMRA45 block diagram

3.8.2 Operation of each circuit

(1) Prescalers

A 9-bit prescaler generates the input clock to TMRA01.

The clock φT0 is divided by 4 and input to this prescaler. φT0 can be either f_{FPH} or fc/16 and is selected using the Prescaler Clock Selection Register SYSCR0<PRCK1,PRCK0>.

The prescaler's operation can be controlled using TA01RUN<TA0PRUN> in the timer control register. Setting <TA0PRUN> to 1 starts the count; setting <TA0PRUN> to 0 clears the prescaler to zero and stops operation. Table 3.8.2 shows the various prescaler output clock resolutions.

Table 3.8.2 Prescaler output clock resolution

@fc = 36 MHz

Prescaler Clock Selection <PRCK1,PRCK0>	Gear Value <GEAR2 to GEAR0>	Prescaler Output Clock Resolution			
		φT1	φT4	φT16	φT256
(f_{FPH})	000 (fc)	fc/2^3 (0.22 μs)	fc/2^5 (0.9 μs)	fc/2^7 (3.6 μs)	fc/2^{11} (57 μs)
	001 (fc/2)	fc/2^4 (0.4 μs)	fc/2^6 (1.8 μs)	fc/2^8 (7.1 μs)	fc/2^{12} (114 μs)
	010 (fc/4)	fc/2^5 (0.9 μs)	fc/2^7 (3.6 μs)	fc/2^9 (14 μs)	fc/2^{13} (228 μs)
	011 (fc/8)	fc/2^6 (1.8 μs)	fc/2^8 (7.1 μs)	fc/2^{10} (28 μs)	fc/2^{14} (455 μs)
	100 (fc/16)	fc/2^7 (3.6 μs)	fc/2^9 (14 μs)	fc/2^{11} (57 μs)	fc/2^{15} (910 μs)
10 (fc/16 clock)	XXX	fc/2^7 (3.6 μs)	fc/2^9 (14 μs)	fc/2^{11} (57 μs)	fc/2^{15} (910 μs)

xxx: Don't care

(2) Up-counters (UC0 and UC1)

These are 8-bit binary counters which count up the input clock pulses for the clock specified by TA01MOD.

The input clock for UC0 is selectable and can be either the external clock input via the TA0IN pin or one of the three internal clocks φT1, φT4 or φT16. The clock setting is specified by the value set in TA01MOD<TA01CLK1,TA01CLK0>.

The input clock for UC1 depends on the operation mode. In 16-Bit Timer Mode, the overflow output from UC0 is used as the input clock. In any mode other than 16-Bit Timer Mode, the input clock is selectable and can either be one of the internal clocks φT1, φT16 or φT256, or the comparator output (the match detection signal) from TMRA0.

For each interval timer the timer operation control register bits TA01RUN<TA0RUN> and TA01RUN<TA1RUN> can be used to stop and clear the up-counters and to control their count. A Reset clears both up-counters, stopping the timers.

(3) Timer registers (TA0REG and TA1REG)

These are 8-bit registers which can be used to set a time interval. When the value set in the timer register TA0REG or TA1REG matches the value in the corresponding up-counter, the Comparator Match Detect signal goes Active. If the value set in the timer register is 00H, the signal goes Active when the up-counter overflows.

The TA0REG are double buffer structure, each of which makes a pair with register buffer.

The setting of the bit TA01RUN<TA0RDE> determines whether TA0REG's double buffer structure is enabled or disabled. It is disabled if <TA0RDE> = 0 and enabled if <TA0RDE> = 1.

When the double buffer is enabled, data is transferred from the register buffer to the timer register when a $2^n - 1$overflow occurs in PWM Mode, or at the start of the PPG cycle in PPG Mode. Hence the double buffer cannot be used in Timer Mode.

A Reset initializes <TA0RDE> to 0, disabling the double buffer. To use the double buffer, write data to the timer register, set <TA0RDE> to 1, and write the following data to the register buffer. Figure 3.8.4 shows the configuration of TA0REG.

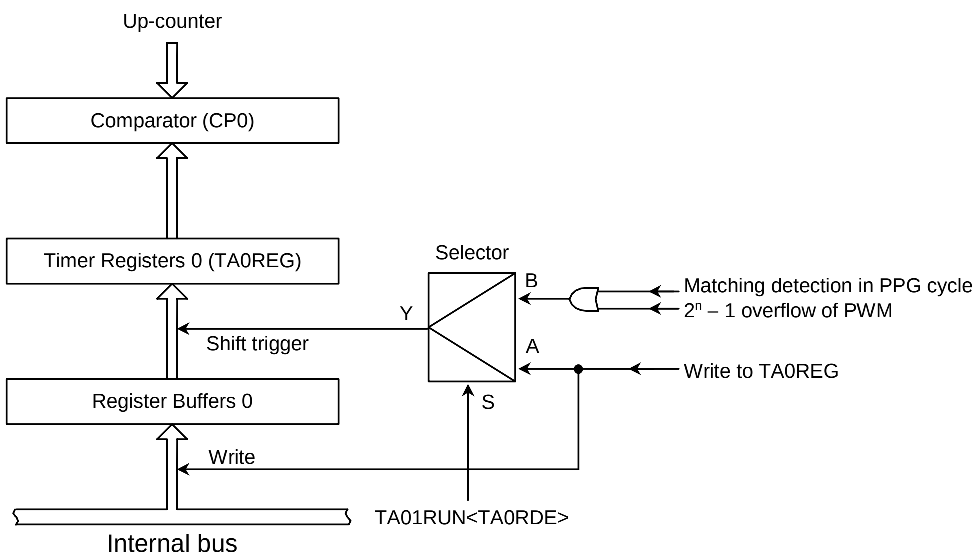


Figure 3.8.4 Configuration of TA0REG

Note: The same memory address is allocated to the timer register and the register buffer. When <TA0RDE> = 0, the same value is written to the register buffer and the timer register; when <TA0RDE> = 1, only the register buffer is written to.

The address of each timer register is as follows.

TA0REG: 000102H	TA1REG: 000103H
TA2REG: 00010AH	TA3REG: 00010BH
TA4REG: 000112H	TA5REG: 000113H

All these registers are write-only and cannot be read.

(4) Comparator (CP0)

The comparator compares the value in an up-counter with the value set in a timer register. If they match, the up-counter is cleared to zero and an interrupt signal (INTTA0 or INTTA1) is generated. If timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

(5) Timer flip-flop (TA1FF)

The timer flip-flop (TA1FF) is a flip-flop inverted by the match detect signal (8-bit comparator output) of each interval timer.

Whether inversion is enabled or disabled is determined by the setting of the bit TA1FFCR<TAFF1IE> in the Timer Flip-Flop Control Register.

A Reset clears the value of TA1FF to 0. Writing 01 or 10 to TA1FFCR<TAFF1C1, TAFF1C0> sets TA1FF to 0 or 1. Writing 00 to these bits inverts the value of TA1FF (this is known as software inversion).

The TA1FF signal is output via the TA1OUT pin (which can also be used as P71). When this pin is used as the timer output, the timer flip-flop should be set beforehand using the Port 7 Function Register P7FC.

3.8.3 SFRs

TMRA01 Run Register

TA01RUN (0100H)

	7	6	5	4	3	2	1	0
Bit symbol	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
Read/Write	R/W				R/W			
After Reset	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		

TA0REG double buffer control

0	Disable
1	Enable

Timer Run/Stop control

0	Stop & Clear
1	Run (count up)

I2TA01: Operation in IDLE2 Mode
TA01PRUN: Run prescaler
TA1RUN: Run Timer 1
TA0RUN: Run Timer 0

Note: The values of bits 4 to 6 of TA01RUN are undefined when read.

TMRA23 Run Register

TA23RUN (0108H)

	7	6	5	4	3	2	1	0
Bit symbol	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
Read/Write	R/W				R/W			
After Reset	0				0	0	0	0
Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		

TA0REG double buffer control

0	Disable
1	Enable

Timer Run/Stop control

0	Stop & Clear
1	Run (count up)

I2TA23: Operation in IDLE2 Mode
TA23PRUN: Run prescaler
TA3RUN: Run Timer 3
TA2RUN: Run Timer 2

Note: The values of bits 4 to 6 of TA23RUN are undefined when read.

Figure 3.8.5 Register for TMRA

TMRA45 Run Register

TA45RUN (0110H)		7	6	5	4	3	2	1	0
	Bit symbol	TA4RDE				I2TA45	TA45PRUN	TA5RUN	TA4RUN
	Read/Write	R/W				R/W			
	After Reset	0				0	0	0	0
	Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		

TA0REG double buffer control

0	Disable
1	Enable

Timer Run/Stop control

0	Stop & Clear
1	Run (count up)

I2TA45: Operation during IDLE2 Mode
TA45PRUN: Run for prescaler
TA5RUN: Run Timer 5
TA4RUN: Run Timer 4

Note: The values of bits 4 to 6 of TA45RUN are undefined when read.

Figure 3.8.6 TMRA registers

TMRA01 Mode Register

TA01MOD (0104H)

	7	6	5	4	3	2	1	0
Bit symbol	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-Bit Timer Mode 01: 16-Bit Timer Mode 10: 8-Bit PPG Mode 11: 8-Bit PWM Mode		PWM cycle 00: reserved 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Source clock for TMRA1 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA0 00: TA0IN pin 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA0 source clock selection

00	TA0IN (external input)
01	ϕT1 (prescaler)
10	ϕT4 (prescaler)
11	ϕT16 (prescaler)

TMRA1 source clock selection

	TA01MOD <TA01M1 to TA01M0> $\neq$ 01	TA01MOD <TA01M1 to TA01M 0> = 01
00	Comparator output from TMRA0	Overflow output from TMRA0 (16-Bit Timer Mode)
01	ϕT1	
10	ϕT16	
11	ϕT256	

PWM cycle selection

00	reserved
01	$(2^6 - 1) \times$ clock source
10	$(2^7 - 1) \times$ clock source
11	$(2^8 - 1) \times$ clock source

TMRA01 operation mode selection

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA0), 8-bit timer (TMRA1)

Figure 3.8.7 TMRA registers

TMRA23 Mode Register

TA23MOD (010CH)		7	6	5	4	3	2	1	0
	Bit symbol	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
	Read/Write	R/W							
	After Reset	0	0	0	0	0	0	0	0
	Function	Operation mode 00: 8-Bit Timer Mode 01: 16-Bit Timer Mode 10: 8-Bit PPG Mode 11: 8-Bit PWM Mode		PWM cycle 00: reserved 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		TMRA3 clock for TMRA3 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA2 clock for TMRA2 00: reserved 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA2 source clock selection

00	Do not set
01	ϕT1 (prescaler)
10	ϕT4 (prescaler)
11	ϕT16 (prescaler)

TMRA3 source clock selection

	TA23MOD <TA23M1 to TA23M0> ≠ 01	TA23MOD <TA23M1 to TA23M 0> = 01
00	Comparator output from TMRA2	Overflow output from TMRA2 (16-Bit Timer Mode)
01	ϕT1	
10	ϕT16	
11	ϕT256	

PWM cycle selection

00	reserved
01	$(2^6 - 1) \times$ clock source
10	$(2^7 - 1) \times$ clock source
11	$(2^8 - 1) \times$ clock source

TMRA23 operation mode selection

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA0), 8-bit timer (TMRA3)

Figure 3.8.8 TMRA registers

TMRA45 Mode Register

TA45MOD (0114H)		7	6	5	4	3	2	1	0
	Bit symbol	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
	Read/Write	R/W							
	After Reset	0	0	0	0	0	0	0	0
	Function	Operation mode 00: 8-Bit Timer Mode 01: 16-Bit Timer Mode 10: 8-Bit PPG Mode 11: 8-Bit PWM Mode		PWM cycle 00: reserved 01: $2^6 – 1$ 10: $2^7 – 1$ 11: $2^8 – 1$		Source clock for TMRA5 00: TA4TRG 01: φT1 10: φT16 11: φT256		Source clock for TMRA4 00: TA4IN pin 01: φT1 10: φT4 11: φT16	

Source clock for TMRA4

00	TA4IN (external input)
01	φT1 (prescaler)
10	φT4 (prescaler)
11	φT16 (prescaler)

Soruce clock for TMRA5

	TA45MOD <TA45M1 to TA45M0> ≠ 01	TA45MOD <TA45M1 to TA45M 0> = 01
00	Comparator output from TMRA4	Overflow output from TMRA4 (16-Bit Timer Mode)
01	φT1	
10	φT16	
11	φT256	

PWM cycle

00	reserved
01	$(2^6 – 1) \times$ clock source
10	$(2^7 – 1) \times$ clock source
11	$(2^8 – 1) \times$ clock source

Operation mode for TMRA45

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA4), 8-bit timer (TMRA5)

Figure 3.8.9 Register for TMRA

TMRA1 Flip-Flop Control Register

TA1FFCR (0105H)

Read-Modify-Write instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol					TAFF1C1	TAFF1C0	TAFF1IE	TAFF1IS
Read/Write					W		R/W	
After Reset					1	1	0	0
Function					00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don't care		TA1FF Control for inversion 0: Disable 1: Enable	TA1FF Inversion select 0: TMRA0 1: TMRA1

Inverse signal for Timer Flip-Flop 1 (TA1FF)
(Don't care except in 8-Bit Timer Mode)

0	Inversion by TMRA0
1	Inversion by TMRA1

Inversion of TA1FF

0	Disabled
1	Enabled

Control of TA1FF

00	Inverts the value of TA1FF
01	Sets TA1FF to 1
10	Clears TA1FF to 0
11	Don't care

Note: The values of bits 4 to 6 of TA1FFCR are undefined when read.

Figure 3.8.10 TMRA registers

TMRA3 Flip-Flop Control Register

TA3FFCR (010DH)

Read-Modify-Write instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol					TAFF3C1	TAFF3C0	TAFF3IE	TAFF3IS
Read/Write					W		R/W	
After Reset					1	1	0	0
Function					00: Invert TA3FF 01: Set TA3FF 10: Clear TA3FF 11: Don't care		TA3FF Control for inversion 0: Disable 1: Enable	TA3FF Inversion select 0: TMRA2 1: TMRA3

Inverse signal for Timer Flip-Flop 3 (TA3FF)
(Don't care except in 8-Bit Timer Mode)

0	Inversion by TMRA2
1	Inversion by TMRA3

Inversion of TA3FF

0	Disabled
1	Enabled

Control of TA3FF

00	Inverts the value of TA3FF
01	Sets TA3FF to 1
10	Clears TA3FF to 0
11	Don't care

Note: The values of bits 4 to 6 of TA3FFCR are undefined when read.

Figure 3.8.11 TMRA register

TMRA5 Flip-Flop Control Register

TA5FFCR (0115H)

Read-Modify-Write instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol					TAFF5C1	TAFF5C0	TAFF5IE	TAFF5IS
Read/Write					W		R/W	
After Reset					1	1	0	0
Function					00: Invert TA5FF 01: Set TA5FF 10: Clear TA5FF 11: Don't care		TA5FF Control for inversion 0: Disable 1: Enable	TA5FF Inversion select 0: TMRA4 1: TMRA5

Inverse signal for Timer Flip-Flop 5 (TA5FF)
(Don't care except in 8-Bit Timer Mode)

0	Inversion by TMRA4
1	Inversion by TMRA5

Inversion of TA5FF

0	Disabled
1	Enabled

Control of TA5FF

00	Inverts the value of TA5FF
01	Sets TA5FF to 1
10	Clears TA5FF to 0
11	Don't care

Note: The values of bits 4 to 6 of TA5FFCR are undefined when read.

Figure 3.8.12 Register for TMRA

3.8.4 Operation in each mode

(1) 8-Bit Timer Mode

Both TMRA0 and TMRA1 can be used independently as 8-bit interval timers.

① Generating interrupts at a fixed interval (using TMRA1)

To generate interrupts at constant intervals using TMRA1 (INTTA1), first stop TMRA1 then set the operation mode, input clock and a cycle to TA01MOD and TA1REG register, respectively. Then, enable the interrupt INTTA1 and start TMRA1 counting.

Example: To generate an INTTA1 interrupt every 8.8 μseconds at fc = 36 MHz, set each register as follows:

* Clock state

System clock: High frequency (fc)

Prescaler clock: f_{FPH}

		MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	←	–	–	X	X	–	–	0	–	Stop TMRA1 and clear it to 0.
TA01MOD	←	0	0	X	X	1	0	X	X	Select 8-Bit Timer Mode and select φT1 (0.22 μs at fc = 36 MHz) as the input clock.
TA1REG	←	0	0	1	0	1	0	0	0	Set TA1REG to 8.8 μs ÷ φT1 = 40 = 28H
INTETA01	←	X	1	0	1	–	–	–	–	Enable INTTA1 and set it to Level 5.
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

Note: X = Don't care; "–" = No change

Select the input clock using Table 3.8.4

Note: The input clocks for TMRA0 and TMRA1 differ as follows:
TMRA0: Uses TA0IN input and can be selected from φT1, φT4 or φT16
TMRA1: Match output of TMRA0 and can be selected from φT1, φT16, φT256

② Generating a 50% duty ratio square wave pulse

The state of the timer flip-flop (TA1FF) is inverted at constant intervals and its status output via the timer output pin (TA1OUT).

Example: To output a 1.32 μs square wave pulse from the TA1OUT pin at fc = 36 MHz, use the following procedure to make the appropriate register settings. This example uses TMRA1; however, either TMRA0 or TMRA1 may be used.

* Clock state
- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: f_{FPH}

		7	6	5	4	3	2	1	0	
TA01RUN	←	–	X	X	X	–	–	0	–	Stop TMRA1 and clear it to 0.
TA01MOD	←	0	0	X	X	0	1	–	–	Select 8-Bit Timer Mode and select ϕT1 (0.22 μs at fc = 36 MHz) as the input clock.
TA1REG	←	0	0	0	0	0	0	1	1	Set the timer register to 1.32 μs ÷ ϕT1 ÷ 2 = 3
TA1FFCR	←	X	X	X	X	1	0	1	1	Clear TA1FF to 0 and set it to invert on the match detect signal from TMRA1.
P7CR	←	X	X	–	–	–	–	1	–	Set P71 to function as the TA1OUT pin.
P7FC	←	X	X	–	–	X	–	1	X	
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

Note: X = Don't care; "–" = No change

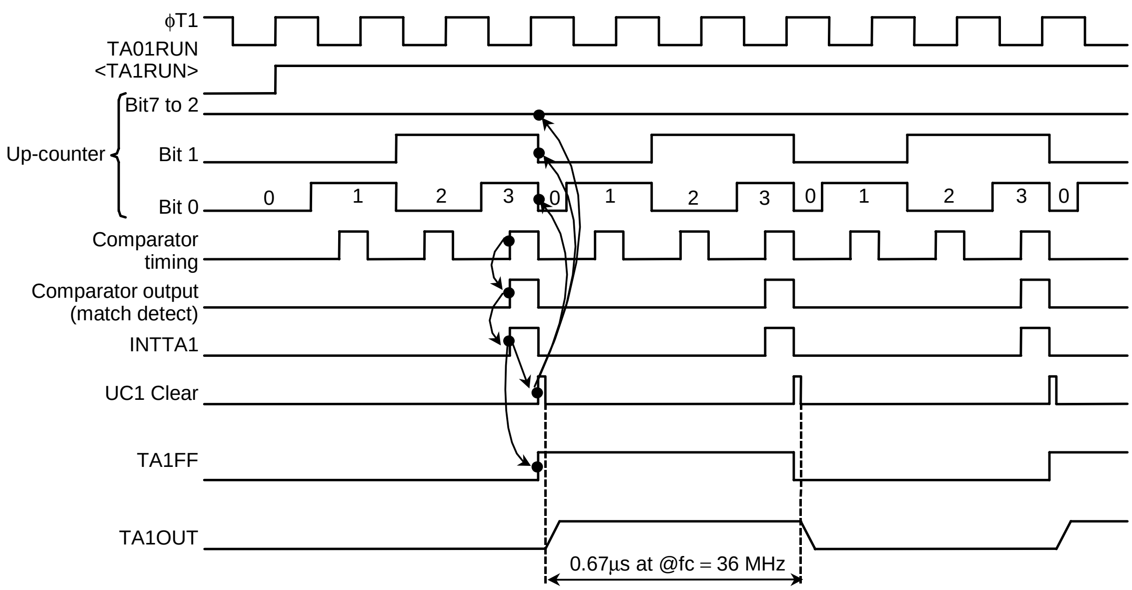


Figure 3.8.13 Square wave output timing chart (50% Duty)

③ Making TMRA1 count up on the match signal from the TMRA0 comparator

Select 8-Bit Timer Mode and set the comparator output from TMRA0 to be the input clock to TMRA1.

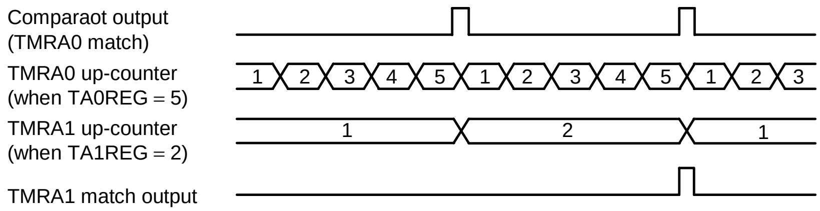


Figure 3.8.14 TMRA1 count up on signal from TMRA0

(2) 16-Bit Timer Mode

A 16-bit interval timer is configured by pairing the two 8-bit timers TMRA0 and TMRA1.

To make a 16-bit interval timer in which TMRA0 and TMRA1 are cascaded together, set TA01MOD <TA01M1,TA01M0> to 01.

In 16-Bit Timer Mode, the overflow output from TMRA0 is used as the input clock for TMRA1, regardless of the value set in TA01MOD<TA01CLK1,TA01CLK0>. Table 3.8.4 shows the relationship between the timer (interrupt) cycle and the input clock selection.

Setting example: To generate an INTTA1 interrupt every 0.225 seconds at fc = 36 MHz, set the timer registers TA0REG and TA1REG as follows:

∗ Clock state

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

If ϕT16 (3.6 μs at 36 MHz) is used as the input clock for counting, set the following value in the registers: 0.225 s ÷ 3.6 μs = 62500 = F424H; i.e. set TA1REG to F4H and TA0REG to 24H.

The comparator match signal is output from TMRA0 each time the up-counter UC0 matches TA0REG, where the up-counter UC0 is not be cleared.

In the case of the TMRA1 comparator, the match detect signal is output on each comparator pulse on which the values in the up-counter UC1 and TA1REG match. When the match detect signal is output simultaneously from both the comparators TMRA0 and TMRA1, the up-counters UC0 and UC1 are cleared to 0 and the interrupt INTTA1 is generated. Also, if inversion is enabled, the value of the timer flip-flop TA1FF is inverted.

Example: When TA1REG = 04H and TA0REG = 80H

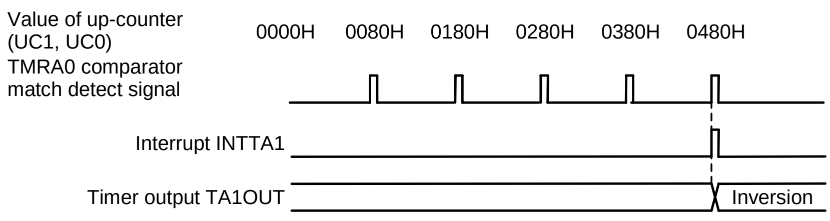


Figure 3.8.15 Timer output by 16-Bit Timer Mode

(3) 8-Bit PPG (Programmable Pulse Generation) Output Mode

Square wave pulses can be generated at any frequency and duty ratio by TMRA0. The output pulses may be active-Low or active-High. In this mode TMRA1 cannot be used.

TMRA0 outputs pulses on the TA1OUT pin (which can also be used as P71).

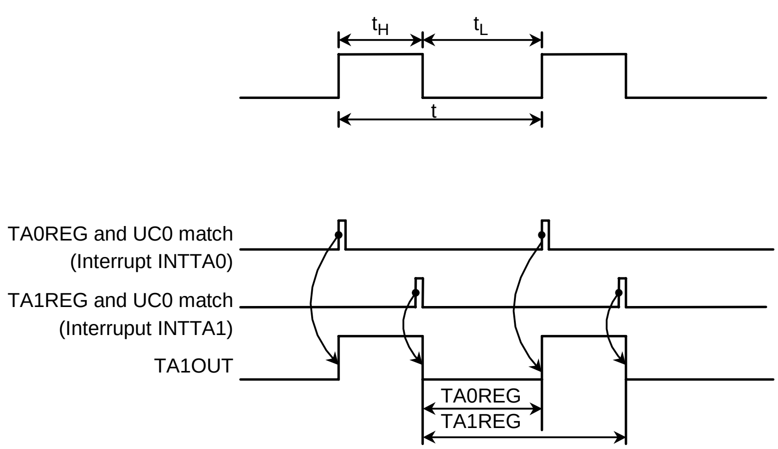


Figure 3.8.16 8 bit PPG output waveforms

In this mode a programmable square wave is generated by inverting the timer output each time the 8-bit up-counter (UC0) matches the value in one of the timer registers TA0REG or TA1REG.

The value set in TA0REG must be smaller than the value set in TA1REG.

Although the up-counter for TMRA1 (UC1) is not used in this mode, TA01RUN<TA1RUN> should be set to 1 so that UC1 is set for counting.

Figure 3.8.17 shows a block diagram representing this mode.

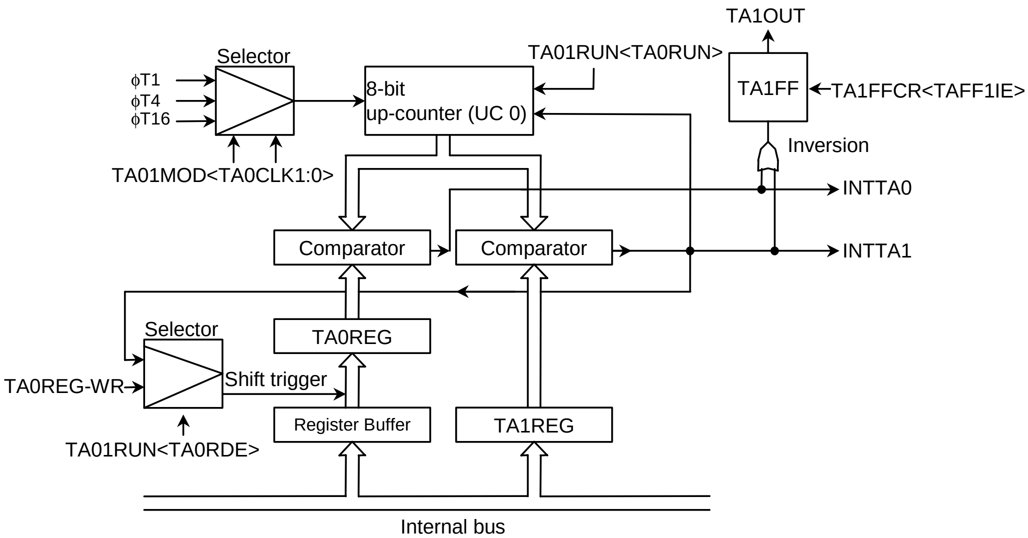


Figure 3.8.17 Block diagram of 8-Bit PPG Output Mode

If the TA0REG double buffer is enabled in this mode, the value of the register buffer will be shifted into TA0REG each time TA1REG matches UC0.

Use of the double buffer facilitates the handling of low-duty waves (when duty is varied).

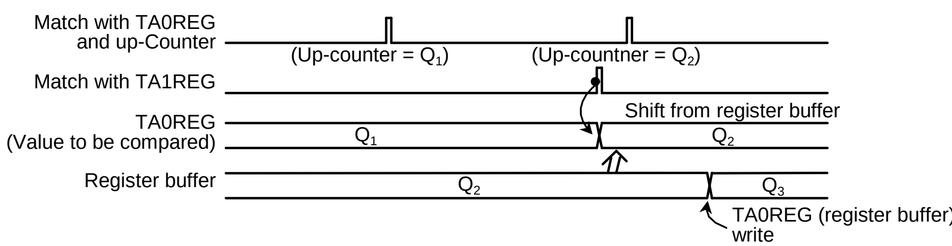
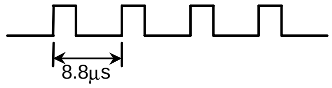


Figure 3.8.18 Operation of register buffer

Example: To generate 1/4-duty 113.636kHz pulses (at fc = 36 MHz):



∗ Clock state
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

Calculate the value which should be set in the timer register.
To obtain a frequency of 113.636 kHz, the pulse cycle t should be:

$t = 1/113.636\ \text{kHz} = 8.8\ \mu s$

$\phi T1 = 0.22\ \mu s$ (at 36 MHz);

$8.8\ \mu s \div 0.22\ \mu s = 40$

Therefore set TA1REG to 40 (28H)

The duty is to be set to 1/4: $t \times 1/4 = 8.8\ \mu s \times 1/4 = 2.2\ \mu s$

$2.2\ \mu s \div 0.22\ \mu s = 10$

Therefore, set TA0REG = 10 = 0AH.

		7	6	5	4	3	2	1	0	
TA01RUN	←	0	X	X	X	–	0	0	0	Stop TMRA0 and TMRA01 and clear it to “0”.
TA01MOD	←	1	0	X	X	X	X	0	1	Set the 8-bit PPG mode, and select ϕT1 as input clock.
TA0REG	←	0	0	0	0	1	0	1	0	Write 0AH
TA1REG	←	0	0	1	0	1	0	0	0	Write 28H
TA1FFCR	←	X	X	X	X	0	1	1	X	Set TA1FF, enabling both inversion and the double buffer. 10 generates a negative logic pulse.
P7CR	←	X	X	–	–	–	–	1	–	Set P71 as the TA1OUT pin.
P7FC	←	X	X	–	–	X	–	1	X	
TA01RUN	←	1	X	X	X	–	1	1	1	Start TMRA0 and TMRA01 counting.

Note: X = Don’t care; “–” = No change

(4) 8-Bit PWM Output Mode

This mode is only valid for TMRA0. In this mode, a PWM pulse with the maximum resolution of 8 bits can be output.

When TMRA0 is used the PWM pulse is output on the TA1OUT pin (which is also used as P71). TMRA1 can also be used as an 8-bit timer.

The timer output is inverted when the up-counter (UC0) matches the value set in the timer register TA0REG or when $2^n - 1$ counter overflow occurs (n = 6, 7 or 8 as specified by TA01MOD<PWM01 to PWM00>). The up-counter UC0 is cleared when $2^n - 1$ counter overflow occurs.

The following conditions must be satisfied before this PWM mode can be used.

Value set in TA0REG < value set for $2^n - 1$ counter overflow
Value set in TA0REG ≠ 0

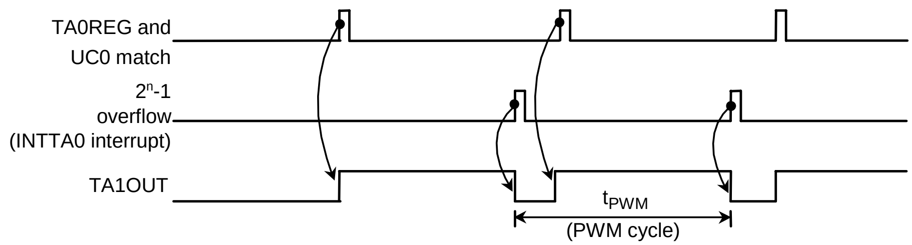


Figure 3.8.19 8-bit PWM waveforms

Figure 3.8.20 shows a block diagram representing this mode.

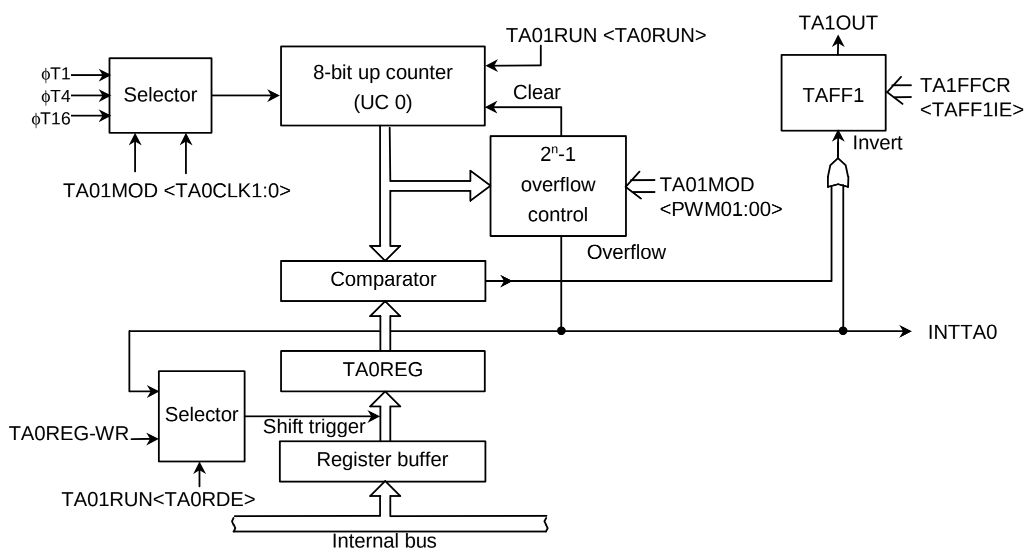


Figure 3.8.20 Block diagram of 8-Bit PWM Mode

In this mode the value of the register buffer will be shifted into TA0REG if $2^n - 1$ overflow is detected when the TA0REG double buffer is enabled.

Use of the double buffer facilitates the handling of low duty ratio waves.

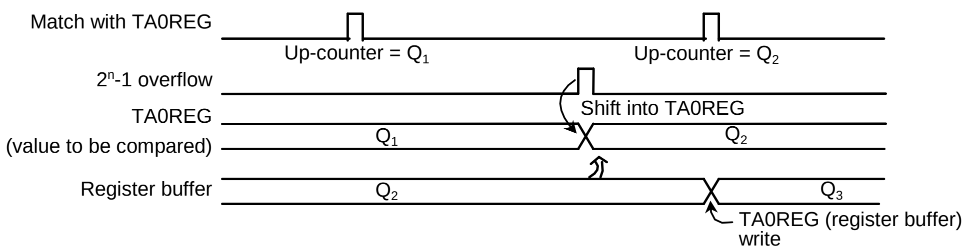
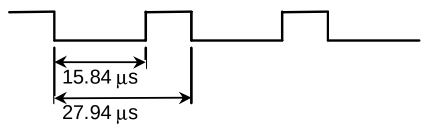


Figure 3.8.21 Register buffer operation

Example: To output the following PWM waves on the TA1OUT pin at fc = 36 MHz:



* Clock state
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

To achieve a 27.94 μs PWM cycle by setting φT1 to 0.22 μs (at fc = 36 MHz):

27.94 μs ÷ 0.22 μs = 127

$2^n - 1 = 127$

Therefore n should be set to 7.

Since the low-level period is 15.84 μs when φT1 = 0.22 μs,

set the following value for TA0REG:

15.84 μs ÷ 0.22 μs = 72 = 48H

		MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	←	–	X	X	X	–	–	–	0	Stop TMRA0 and clear it to 0.
TA01MOD	←	1	1	1	0	–	–	0	1	Select 8-Bit PWM Mode (cycle: $2^7 - 1$) and select φT1 as the input clock.
TA0REG	←	0	1	0	0	1	0	0	0	Write 48H.
TA1FFCR	←	X	X	X	X	1	0	1	X	Clear TA1FF to 0, enable the inversion and double buffer.
P7CR	←	X	X	–	–	–	–	1	–	Set P71 and the TA1OUT pin.
P7FC	←	X	X	–	–	X	–	1	X	
TA01RUN	←	1	X	X	X	–	1	1	1	Start TMRA0 counting.

Note: X = Don’t care; “–” = No change

Table 3.8.3 PWM cycle

@fc = 36 MHz

Select Prescaler Clock <PRCK1 to PRCK0>	Gear Value <GEAR2 to GEAR0>	PWM cycle								
		$2^6 – 1$			$2^7 – 1$			$2^8 – 1$		
		φT1	φT4	φT16	φT1	φT4	φT16	φT1	φT4	φT16
00 (f_{FPH})	000 (fc)	12.6 μs	56.7 μs	66.6 μs	25.4 μs	114 μs	457 μs	51 μs	230 μs	918 μs
	001 (fc/2)	25.2 μs	113 μs	447 μs	50.8 μs	229 μs	901 μs	102 μs	459 μs	1811 μs
	10 (fc/4)	56.7 μs	227 μs	895 μs	114 μs	457 μs	1803 μs	230 μs	918 μs	3621 μs
	011 (fc/8)	113 μs	447 μs	1789 μs	229 μs	902 μs	3607 μs	459 μs	1811 μs	7242 μs
	00 (fc/16)	227 μs	895 μs	3585 μs	457 μs	1803 μs	7226 μs	918 μs	3621 μs	14510 μs
10 (fc/16 clcok)	XXX	227 μs	895 μs	3585 μs	457 μs	1803 μs	7226 ms	918 μs	3621 μs	14510 μs

XXX: Don’t care

(5) Settings for each mode

Table 3.8.4 shows the SFR settings for each mode.

Table 3.8.4 Timer mode setting registers

Register name	TA01MOD				TA1FFCR
<Bit Symbol>	<TA01M1:TA01M 0>	<PWM01:00>	<TA1CLK1:0>	<TA0CLK1:0>	TAFF1IS
Function	Timer mode	PWM cycle	Upper timer input clock	Lower timer input clock	Timer F/F invert signal select
8-bit timer × 2 channels	00	—	Lower timer match φT1, φT16, φT256 (00, 01, 10, 11)	External clock φT1, φT4, φT16 (00, 01, 10, 11)	0: Lower timer output 1: Upper timer output
16-bit timer mode	01	—	—	External clock φT1, φT4, φT16 (00, 01, 10, 11)	—
8-bit PPG × 1 channel	10	—	—	External clock φT1, φT4, φT16 (00, 01, 10, 11)	—
8-bit PWM × 1 channel	11	$2^6 – 1$, $2^7 – 1$, $2^8 – 1$ (01, 10, 11)	—	External clock φT1, φT4, φT16 (00, 01, 10, 11)	—
8-bit timer × 1 channel	11	—	φT1, φT16 , φT256 (01, 10, 11)	—	Output disabled

Note: “–” = Don’t care

3.9 16-Bit Timer/Event Counters (TMRB)

The TMP91C829 incorporates multifunctional 16-bit timer/event counter (TMRB0) which has the following operation modes:

- 16-Bit Interval Timer Mode
- 16-Bit Event Counter Mode
- 16-Bit Programmable Pulse Generation (PPG) Mode

The timer/event counter channel consists of a 16-bit up-counter, two 16-bit timer registers (one of them with a double-buffer structure), two 16-bit capture registers, two comparators, a capture input controller, a timer flip-flop and a control circuit.

The timer/event counter is controlled by an 11-byte control SFR.

This chapter consists of the following items:

Table 3.9.1 Differences between TMRB0

Spec \ Channel		TMRB0
External Pins	External clock / Capture trigger input pins	TB0IN0 (also used as P93) TB0IN1 (also used as P94)
	Timer flip-flop output pins	TB0OUT0 (also used as P95) TB0OUT1 (also used as P96)
SFR (address)	Timer Run Register	TB0RUN (0180H)
	Timer Mode Register	TB0MOD (0182H)
	Timer Flip-Flop Control Register	TB0FFCR (0183H)
	Timer Register	TB0RG0L (0188H) TB0RG0H (0189H) TB0RG1L (018AH) TB0RG1H (018BH)
	Capture Register	TB0CP0L (018CH) TB0CP0H (018DH) TB0CP1L (018EH) TB0CP1H (018FH)

3.9.1 Block diagrams

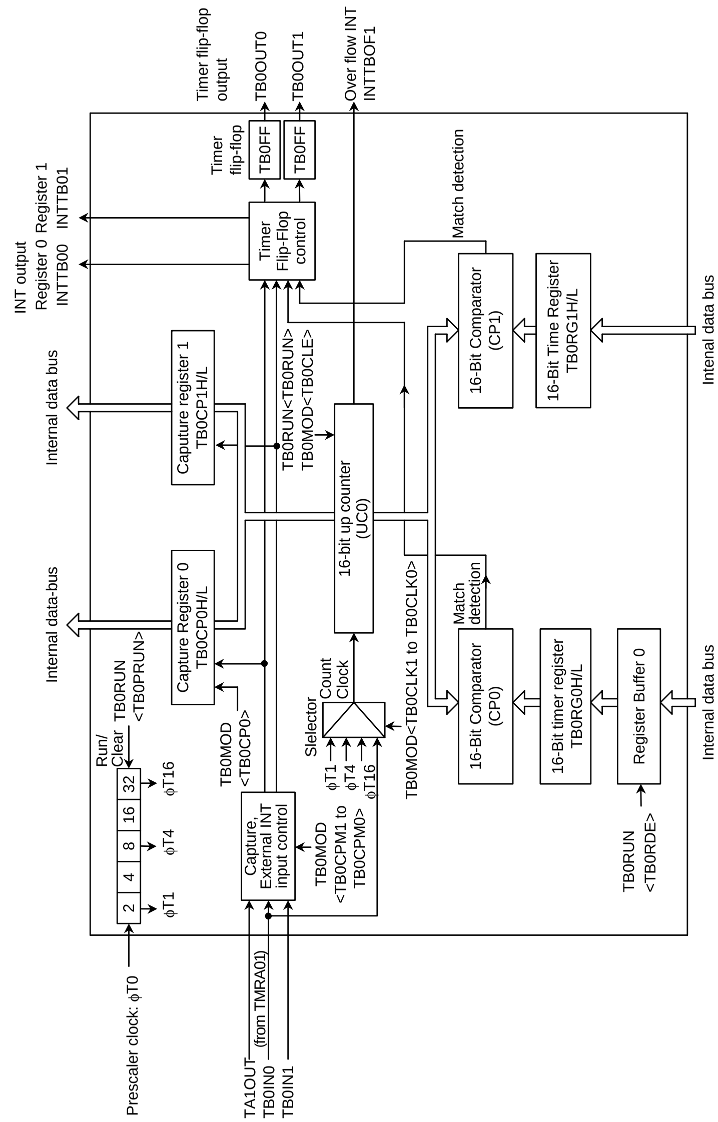


Figure 3.9.1 Block Diagram of TMRB0

3.9.2 Operation of each block

(1) Prescaler

The 5-bit prescaler generates the source clock for TMRB0. The prescaler clock (φT0) is divided clock (divided by 4) from selected clock by the register SYSCR0<PRCK1 to PRCK0> of clock-gear.

This prescaler can be started or stopped using TB0RUN<TB0RUN>. Counting starts when <TB0RUN> is set to 1; the prescaler is cleared to zero and stops operation when <TB0RUN> is set to 0.

Table 3.9.2 Prescaler clock resolution

@fc = 36 MHz

Prescaler Clock Selection <PRCK1 to PRCK0>	Clock Gear Value <GEAR2 to GEAR0>	Prescaler Clock Resolution		
		φT1	φT4	φT16
00 (f_{FPH})	000 (fc)	fc/2^3 (0.2 μs)	fc/2^5 (0.9 μs)	fc/2^7 (3.6 μs)
	001 (fc$_{/2}$)	fc/2^4 (0.4 μs)	fc/2^6 (1.8 μs)	fc/2^8 (7.1 μs)
	010 (fc$_{/4}$)	fc/2^5 (0.9 μs)	fc/2^7 (3.6 μs)	fc/2^9 (14 μs)
	011 (fc$_{/8}$)	fc/2^6 (1.8 μs)	fc/2^8 (7.1 μs)	fc/2^{10} (28 μs)
	100 (fc$_{/16}$)	fc/2^7 (3.6 μs)	fc/2^9 (14 μs)	fc/2^{11} (57 μs)
10 (fc/16 clock)	XXX	fc/2^7 (3.6 μs)	fc/2^9 (14 μs)	fc/2^{11} (57 μs)

xxx: Don't care

(2) Up-counter (UC0)

UC0 is a 16-bit binary counter which counts up pulses input from the clock specified by TB0MOD<TB0CLK1,TB0CLK0>.

Any one of the prescaler internal clocks φT1, φTB0 and φT16 or an external clock input via the TB0IN0 pin can be selected as the input clock. Counting or stopping & clearing of the counter is controlled by TB0RUN<TB0RUN>.

When clearing is enabled, the up-counter UC0 will be cleared to zero each time its value matches the value in the timer register TB0RG1H/L. Clearing can be enabled or disabled using TB0MOD<TB0CLE>.

If clearing is disabled, the counter operates as a free-running counter.

A Timer Overflow interrupt (INTTBOF0) is generated when UC0 overflow occurs.

(3) Timer registers (TB0RG0H/L and TB0RG1H/L)

These two 16-bit registers are used to set the interval time. When the value in the up-counter UC0 matches the value set in this timer register, the Comparator Match Detect signal will go Active.

Setting data for timer register is executed using 2 byte data transfer instruction or using 1 byte date transfer instruction twice for lower 8 bits and upper 8 bits in order. The TB0RG0 timer register has a double-buffer structure, which is paired with register buffer. The value set in TB0RUN<TB0RDE> determines whether the double-buffer structure is enabled or disabled: it is disabled when <TB0RDE> = 0, and enabled when <TB0RDE> = 1.

When the double buffer is enabled, data is transferred from the register buffer to the timer register when the values in the up-counter (UC0) and the timer register TB0RG1 match.

After a Reset, TB0RG0 and TB0RG1 are undefined. If the 16-bit timer is to be used after a Reset, data should be written to it beforehand.

On a Reset TB0RUN<TB0RDE> is initialized to 0, disabling the double buffer. To use the double buffer, write data to the timer register, set <TB0RDE> to 1, then write data to the register buffer as shown below.

TB0RG0 and the register buffer both have the same memory addresses (000188H and 000189H) allocated to them. If <TB0RDE> = 0, the value is written to both the timer register and the register buffer. If <TB0RDE> = 1, the value is written to the register buffer only.

The addresses of the Timer Registers are as follows:

TMRB0

TB0RG0		TB0RG1	
Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
000189H	000188H	00018BH	00018AH

The Timer Registers are write-only registers and thus cannot be read.

(4) Capture Registers (TB0CP0H/L and TB0CP1H/L)

These 16-bit registers are used to latch the values in the up-counter UC0.

Data in the Capture Registers should be read using a 2-byte data load instruction or two 1-byte data load instructions. The least significant byte is read first, followed by the most significant byte.

The addresses of the Capture Registers are as follows:

TMRB0

TB0CP0		TB0CP1	
MS 8 bits	LS 8 bits	MS 8 bits	LS 8 bits
00018DH	00018CH	00018FH	00018EH

The Capture Registers are read-only registers and thus cannot be written to.

(5) Capture input control

This circuit controls the timing to latch the value of up-counter UC0 into TB0CP0, TB0CP1. The latch timing for the capture register is determined by TB0MOD<TB0CPM1, TB0CPM0>.

In addition, the value in the up-counter can be loaded into a capture register by software. Whenever 0 is written to TB0MOD<TB0CP0>, the current value in the up-counter is loaded into capture register TB0CP0. It is necessary to keep the prescaler in Run Mode (i.e. TB0RUN<TB0PRUN> must be held at a value of 1).

(6) Comparators (CP0 and CP1)

CP0 and CP1 are 16-bit comparators which compare the value in the up-counter UC0 with the value set in TB0RG0 or TB0RG1 respectively, in order to detect a match. If a match is detected, the comparator generates an interrupt (INTTB00 or INTTB01 respectively).

(7) Timer flip-flops (TB0FF0 and TB0FF1)

These flip-flops are inverted by the match detect signals from the comparators and the latch signals to the Capture Registers. Inversion can be enabled and disabled for each element using TB0FFCR<TB0C1T1, TB0C0T1, TB0E1T1, TB0E0T1>. After a Reset the value of TB0FF0 is undefined. If 00 is written to TB0FFCR<TB0FF0C1, TB0FF0C0> or <TB0FF1C1, TB0FF1C0>, TB0FF0 will be inverted. If 01 is written to the capture registers, the value of TB0FF0 will be set to 1.If 10 is written to the capture registers, the value of TB0FF0 will be set to 0. The values of TB0FF0 and TB0FF1 can be output via the Timer Output pins TB0OUT0 (which is shared with P95) and TB0OUT1 (which is shared with P96). Timer output should be specified using the Port 9 Function Register.

3.9.3 SFR

TMRB0 Run Register

TB0RUN (0180H)

	7	6	5	4	3	2	1	0
Bit symbol	TB0RDE	—			I2TB0	TB0PRUN		TB0RUN
Read/Write	R/W	R/W			R/W	R/W		R/W
After Reset	0	0			0	0		0
Function	Double Buffer 0: Disable 1: Enable	Write 0			IDLE2 0: Stop 1: Operate	Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		

Count operation

0	Stop and Clear
1	Count

I2TB0: Operation during IDLE2-mode
TB0PRUN: Operation of prescaler
TB0RUN: Operation of TMRB0

Note: The 1, 4 and 5 of TB0RUN are read as undefined value.

Figure 3.9.2 The Registers for TMRB

TMRB0 Run Register

TB0MOD (0182H)		7	6	5	4	3	2	1	0
	Bit symbol	TB0CT1	TB0ET1	TB0CP01	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
	Read/Write	R/W		W*	R/W				
	After Reset	0	0	1	0	0	0	0	0
	Function	TB0FF1 inversion 0: Disable trigger 1: Enable trigger Invert when the UC value is captured to TB0CP1.	Invert when the UC value matches the value in TB0RG1.	Execute software capture 0: Execute 1: Don't care	Capture timing 00: Disable 01: TB0IN0 ↑ TB0IN1 ↑ 10: TB0IN0 ↑ TB0IN0 ↓ 11: TA0TRG ↑ TA0TRG ↓		Control up-counter 0: Disable clearing 1: Enable clearing	TMRB0 source clock 00: TB0IN0 pin 01: φT1 10: φT4 11: φT16	

TMRB0 source clock

00	TB0IN0 pin
01	φT1
10	φT4
11	φT16

Up-counter clear control

0	Disable
1	TB0RG1 clearing on match with TB0RG1.

Capture

	Capture control
00	Disable
01	CAP0 at TB0IN0 rising CAP1 at TB0IN1 rising
10	CAP0 at TB0IN0 rising CAP1 at TB0IN0 rising
11	CAP0 at TA0TRG rising CAP1 at TB0TRG falling

Software capture

0	The value in the up-counter is captured to TB0CP0.
1	Don't care

Figure 3.9.3 The registers for TMRB

TMRB0 Flip-Flop Control Register

TB0FFCR (0183H)		7	6	5	4	3	2	1	0
	Bit symbol	TB0FF1C1	TB0FF1C0	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
	Read/Write	W*		R/W				W*	
	After Reset	0	0	1	0	0	0	0	0
	Function	Control TB0FF1 00: Invert 01: Set 10: Clear 11: Don't care * Always read as 11		TB0FF0 inversion trigger 0: Disable trigger Enable trigger Invert when the UC value is loaded in to TB0CP1.	Invert when the UC value is loaded in to TB0CP0.	Invert when the UC value matches the value in TB0RG1.	Invert when the UC value matches the value in TB0RG0.	Control TB0FF0 00: Invert 01: Set 10: Clear 11: Don't care * Always read as 11	

TB0FF0 control

00	Invert
01	Set to 11
10	Clear to 0
11	Don't care

Inverted when the UC value is loaded in to TB0CP1.

0	Disable trigger
1	Enable trigger

Inverted when the UC value is loaded in to TB0CP0.

0	Disable trigger
1	Enable trigger

Inverted when the UC value matches the valued in TB0RG1.

0	Disable trigger
1	Enable trigger

Inverted when the UC value matches the valued in TB0RG0.

0	Disable trigger
1	Enable trigger

Figure 3.9.4 The Registers for TMRB

3.9.4 Operation in each mode

(1) 16-Bit Interval Timer Mode

Generating interrupts at fixed intervals

In this example, the interrupt INTTB01 is set to be generated at fixed intervals. The interval time is set in the timer register TB0RG1.

```
              7  6  5  4  3  2  1  0
TB0RUN    ←   0  0  X  X  –  0  X  0      Stop TMRB0.
INTETB01  ←   X  1  0  0  X  0  0  0      Enable INTTB01 and set Interrupt Level 4. Disable
                                          INTTB00.
TB0FFCR   ←   1  1  0  0  0  0  1  1      Disable the trigger.
TB0MOD    ←   0  0  1  0  0  1  *  *      Select internal clock for input and
                    (** = 01, 10, 11)     disable the capture function.
TB0RG1    ←   *  *  *  *  *  *  *  *      Set the interval time (16 bits).
              *  *  *  *  *  *  *  *
TB0RUN    ←   0  0  X  X  –  1  X  1      Start TMRB0.
```

Note: X = Don't care; "–" = No change

(2) 16-Bit Event Counter Mode

As described above, in 16-Bit Timer Mode, if the external clock (TB0IN0 pin input) is selected as the input clock, the timer can be used as an event counter. To read the value of the counter, first perform software capture once, then read the captured value.

```
              7  6  5  4  3  2  1  0
TB0RUN    ←   0  0  X  X  –  0  X  0      Stop TMRB0.
P8CR          -  -  -  -  0  -  -  -      Set P93 input mode
INTETB01  ←   X  1  0  0  X  0  0  0      Enable INTTB01 and set Interrupt Level 4. Disable
                                          INTTB00.
TB0FFCR   ←   1  1  0  0  0  0  1  1      Disable the trigger.
TB0MOD    ←   0  0  1  0  0  1  0  0      Select TB0IN0 as the input clock.
TB0RG1    ←   *  *  *  *  *  *  *  *      Set the number of counts (16 bits).
TB0RUN    ←   0  0  X  X  –  1  X  1      Start TMRB0.
```

Note: X = Don't care; "–" = No change

When the timer is used as an event counter, set the prescaler in Run Mode (i.e. with TB0RUN<TB0PRUN> = 1).

(3) 16-Bit Programmable Pulse Generation (PPG) Output Mode

Square wave pulses can be generated at any frequency and duty ratio. The output pulse may be either Low-active or High-active.

The PPG mode is obtained by inversion of the timer flip-flop TB0FF0 that is to be enabled by the match of the up-counter UC0 with timer register TB0RG0 or TB0RG1 and to be output to TB0OUT0. In this mode the following conditions must be satisfied.

(Value set in TB0RG0) < (Value set in TB0RG1)

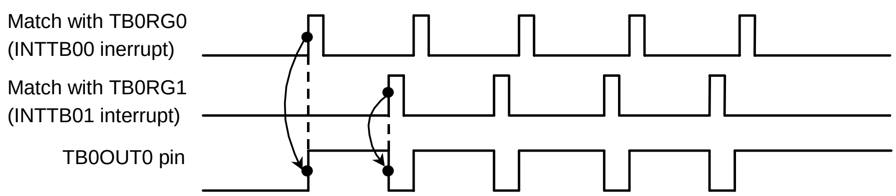


Figure 3.9.5 Programmable Pulse Generation (PPG) Output Waveforms

When the TB0RG0 double buffer is enabled in this mode, the value of Register Buffer 0 will be shifted into TB0RG0 at match with TB0RG1. This feature facilitates the handling of low-duty waves.

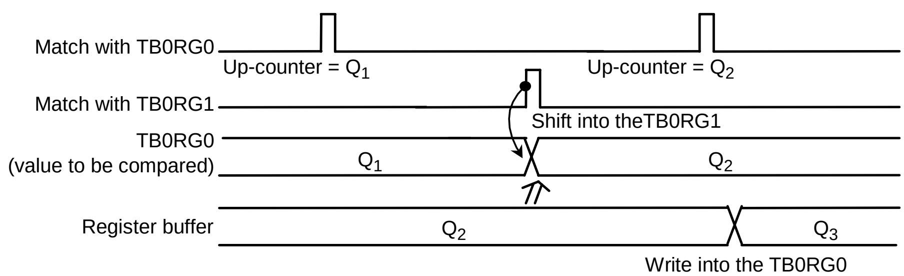


Figure 3.9.6 Operation of Register Buffer

The following block diagram illustrates this mode.

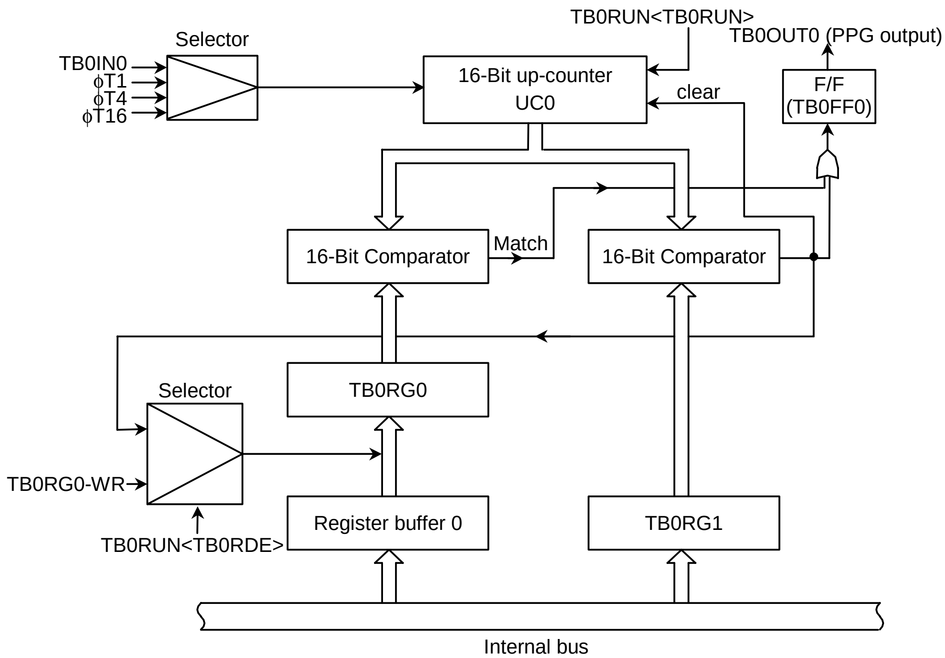


Figure 3.9.7 Block Diagram of 16-BIT Mode

The following example shows how to set 16-Bit PPG Output Mode:

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Disable the TB0RG0 double buffer and stop TMRB0.
TB0RG0	←	*	*	*	*	*	*	*	*	Set the duty ratio (16 bits).
TB0RG1	←	*	*	*	*	*	*	*	*	Set the frequency (16 bits).
TB0RUN	←	1	0	X	X	–	0	X	0	Enable the TB0RG0 double buffer. (The duty and frequency are changed on an INTTB01 interrupt.)
TB0FFCR	←	X	X	0	0	1	1	1	0	Set the mode to invert TB0FF0 at the match with TB0RG0/TB0RG1. Set TB0FF0 to 0.
TB0MOD	←	0	0	1	0	0	1	*	*	Select the internal clock as the input clock and disable
				(** = 01, 10, 11)						the capture function.
P9CR	←	–	–	1	–	–	–	–	–	Set P95 to function as TB0OUT0.
P9FC	←	X	–	1	X	X	X	X	-	
TB0RUN	←	1	0	X	X	–	1	X	1	Start TMRB0.

Note: X = Don't care; "–" = No change

3.10 Serial Channel

TMP91C829 includes one serial I/O channel. Either UART Mode (asynchronous transmission) or I/O Interface Mode (synchronous transmission) can be selected.

- I/O Interface Mode —— Mode 0: For transmitting and receiving I/O data using the synchronizing signal SCLK for extending I/O.

- UART Mode
  - Mode 1: 7-bit data
  - Mode 2: 8-bit data
  - Mode 3: 9-bit data

In Mode 1 and Mode 2 a parity bit can be added. Mode 3 has a wake-up function for making the master controller start slave controllers via a serial link (a multi-controller system).

Figure 3.10.4 and 3 are block diagrams.

Table 3.10.1 Channels 0 and 1

	Channel 0	Channel 1
Pin Name	TXD0 (P80) RXD0 (P81) $\overline{\text{CTS0}}$ /SCLK0 (P82) /STS0 (P83)	TXD1 (P84) RXD1 (P85) $\overline{\text{CTS0}}$ /SCLK1 (P86) /STS1 (P87)

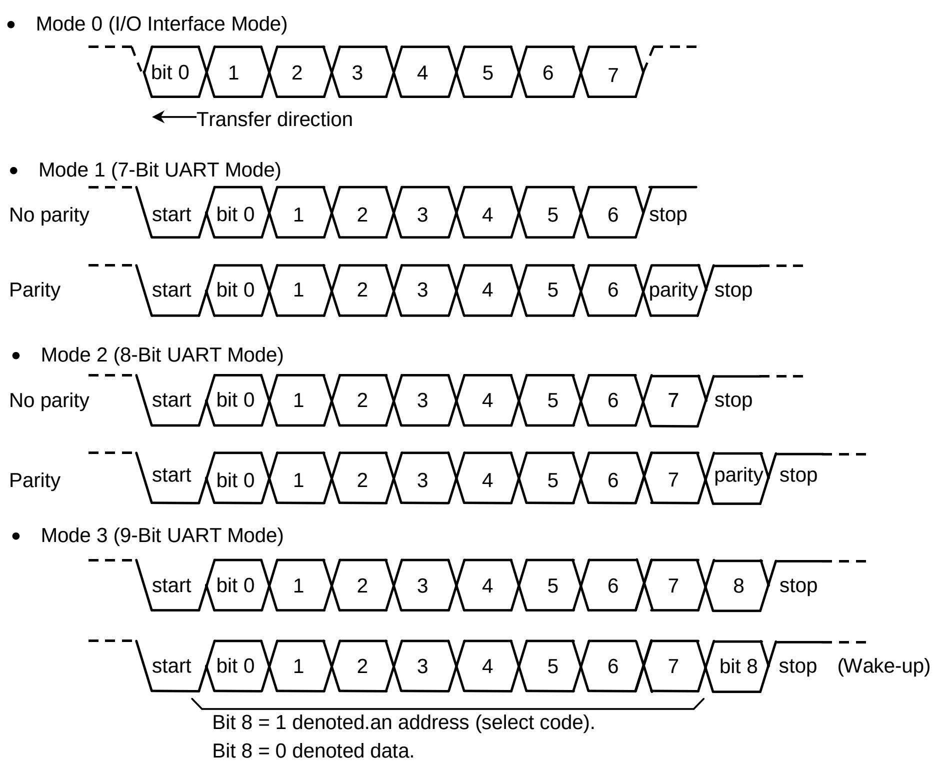


Figure 3.10.1 Data formats

3.10.1 Block diagrams

Figure 3.10.2 is a block diagram representing Serial Channel 0.

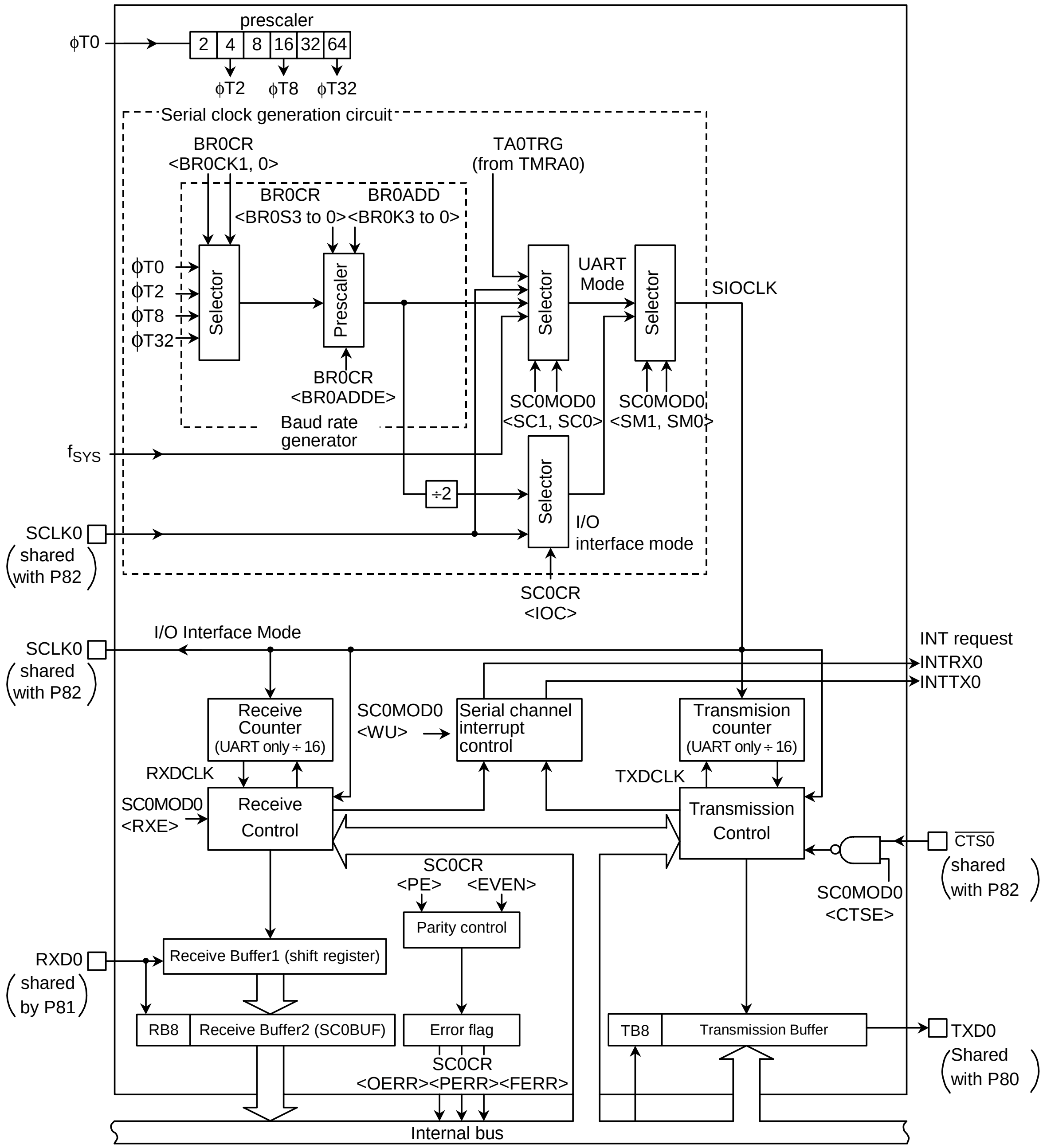


Figure 3.10.2 Block diagram of the Serial Channel 0

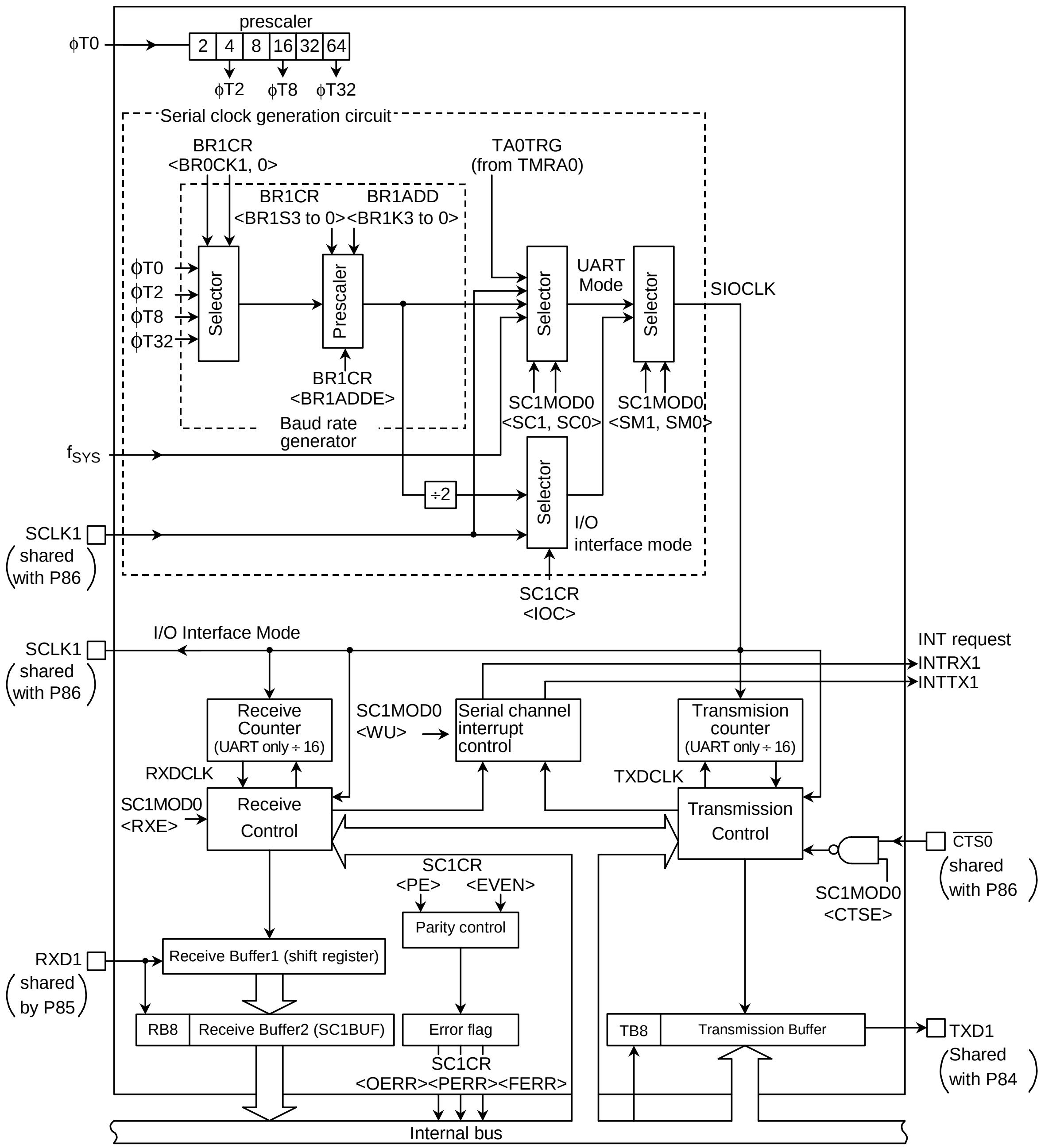


Figure 3.10.3 Block diagram of the Serial Channel 1

3.10.2 Operation of each circuit

(1) Prescaler, Prescaler clock select

There is a 6-bit prescaler for waking serial clock. The clock selected using SYSCR<PRCK1:PRCK0> is divided by 4 and input to the prescaler as ϕT0. The prescaler can be run by selecting the baud rate generator as the waking serial clock.

Table 3.10.2 shows prescaler clock resolution into the baud rate generator.

Table 3.10.2 Prescaler Clock Resolution to Baud Rate Generator

Select Prescaler Clock <PRCK1 to PRCK0>	Gear Value <GEAR2 to GEAR0>	Prescaler Output Clock Resolution			
		ϕT0	ϕT2	ϕT8	ϕT32
00 (f_{FPH})	000 (fc)	$fc/2^2$	$fc/2^4$	$fc/2^6$	$fc/2^8$
	001 (fc/2)	$fc/2^3$	$fc/2^5$	$fc/2^7$	$fc/2^9$
	010 (fc/4)	$fc/2^4$	$fc/2^6$	$fc/2^8$	$fc/2^{10}$
	011 (fc/8)	$fc/2^5$	$fc/2^7$	$fc/2^9$	$fc/2^{11}$
	100 (fc/16)	$fc/2^6$	$fc/2^8$	$fc/2^{10}$	$fc/2^{12}$
10 (fc/16 clock)	XXX	—	$fc/2^8$	$fc/2^{10}$	$fc/2^{12}$

Note: X = Don't care; "–" = Cannot be used

The Baud Rate Generator selects between 4 clock inputs : ϕT0, ϕT2, ϕT8, and ϕT32 among the prescaler outputs.

(2) Baud rate generator

The baud rate generator is a circuit which generates transmission and receiving clocks which determine the transfer rate of the serial channels.

The input clock to the baud rate generator, φT0, φT2, φT8 or φT32, is generated by the 6-bit prescaler which is shared by the timers. One of these input clocks is selected using the BR0CR<BR0CK1 to BR0CK0> field in the Baud Rate Generator Control Register.

The baud rate generator includes a frequency divider, which divides the frequency by 1 or $N + \frac{(16 - K)}{16}$ to 16 values, determining the transfer rate.

The transfer rate is determined by the settings of BR0CR<BR0ADDE, BR0S3 to BR0S0> and BR0ADD<BR0K3 to BR0K0>.

- In UART Mode

(1) When BR0CR<BR0ADDE> = 0

The settings BR0ADD<BR0K3 to BR0K0> are ignored. The baud rate generator divides the selected prescaler clock by N, which is set in BR0CK<BR0S3 to BR0S0>. (N = 1, 2, 3 ⋯ 16)

(2) When BR0CR<BR0ADDE> = 1

The N + (16 – K) / 16 division function is enabled. The baud rate generator divides the selected prescaler clock by N + (16 – K) / 16 using the value of N set in BR0CR<BR0S3 to BR0S0> (N = 2, 3 ⋯ 15) and the value of K set in BR0ADD<BR0K3 to BR0K0> (K = 1, 2, 3 ⋯ 15)

Note: If N = 1 or N = 16, the N + (16 – K) / 16 division function is disabled. Set BR0CR<BR0ADDE> to 0.

- In I/O Interface Mode

The N + (16 – K) / 16 division function is not available in I/O Interface Mode. Set BR0CR<BR0ADDE> to 0 before dividing by N.

The method for calculating the transfer rate when the baud rate generator is used is explained below.

- In UART Mode

$$\text{Baud Rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

- In I/O Interface Mode

$$\text{Baud Rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 2$$

- Integer divider (N divider)

For example, when the source clock frequency (fc) = 12.288 MHz, the input clock frequency = φT2 (fc/16), the frequency divider N (BR0CR<BR0S3 to BR0S0>) = 5, and BR0CR<BR0ADDE> = 0, the baud rate in UART Mode is as follows:

∗ Clock state
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: System clock

$$\text{Baud Rate} = \frac{\text{fc}/16}{5} \div 16$$
$$= 12.288 \times 10^6 \div 16 \div 5 \div 16 = 9600 \text{ (bps)}$$

Note: The N + (16 – K) / 16 division function is disabled and setting BR0ADD<BR0K3 to BR0K0> is invalid.

- N + (16-K)/16 divider (UART Mode only)

Accordingly, when the source clock frequency (fc) = 4.8 MHz, the input clock frequency = ϕT0 , the frequency divider N (BR0CR<BR0S3 to BR0S0>) = 7, K (BR0ADD<BR0K3 to BR0K0>) = 3, and BR0CR <BR0ADDE> = 1, the baud rate in UART Mode is as follows:

$\ast$ Clock state
- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: System clock

$$\text{Baud Rate} = \frac{fc/4}{7 + (16 - 3)/16} \div 16$$

$$= 4.8 \times 10^6 \div 4 \div (7 + 13/16) \div 16 = 9600 \text{ (bps)}$$

Table 3.10.3 and 3.10.4 show examples of UART Mode transfer rates.

Additionally, the external clock input is available in the serial clock. (Serial Channels 0 and 1). The method for calculating the baud rate is explained below:

- In UART Mode

  Baud rate = external clock input frequency ÷ 16

  It is necessary to satisfy (external clock input cycle) ≥ fc / 4

- In I/O Interface Mode

  Baud rate = external clock input frequency

  It is necessary to satisfy (external clock input cycle) ≥ 16 / fc

Table 3.10.3 Transfer rate selection (when baud rate generator Is used and BR0CR <BR0ADDE> = 0)

Unit (kbps)

fc [MHz]	Input Clock / Frequency Divider	φT0	φT2	φT8	φT32
9.830400	2	76.800	19.200	4.800	1.200
	4	38.400	9.600	2.400	0.600
	8	19.200	4.800	1.200	0.300
	0	9.600	2.400	0.600	0.150
12.288000	5	38.400	9.600	2.400	0.600
	A	19.200	4.800	1.200	0.300
14.745600	2	115.200			
	3	76.800	19.200	4.800	1.200
	6	38.400	9.600	2.400	0.600
	C	19.200	4.800	1.200	0.300

Note 1: Transfer rates in I/O Interface Mode are eight times faster than the values given above.

Note 2: The values in this table are calculated for when fc is selected as the system clock, the clock gear is set for fc and the system clock is the prescaler clock input.

Table 3.10.4 Selection of Transfer Rate (When TMRA0 with input Clock φT1 is used)

Unit (kbps)

TA0REG0 \ fc	12.288 MHz	12 MHz	9.8304 MHz	8 MHz	6.144 MHz
1H	96		76.8	62.5	48
2H	48		38.4	31.25	24
3H	32	31.25			16
4H	24		19.2		12
5H	19.2				9.6
8H	12		9.6		6
AH	9.6				4.8
10H	6		4.8		3
14H	4.8				2.4

Method for calculating the transfer rate (when TMRA0 is used):

$$\text{Transfer rate} = \frac{\text{Clock frequency determined by SYSCR0<PRCK1, PRCK0>}}{\text{TA0REG} \times \underline{8} \times 16}$$

(when TMRA0 (input clock φT1) is used)

Note 1: The TMRA0 match detect signal cannot be used as the transfer clock in I/O Interface Mode.

Note 2: The values in this table are calculated for when fc is selected as the system clock, the clock gear is set for fc and the system clock is the prescaler clock input.

(3) Serial clock generation circuit

This circuit generates the basic clock for transmitting and receiving data.

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = 0, the basic clock is generated by dividing the output of the baud rate generator by 2, as described previously.

In SCLK Input Mode with the setting SC0CR<IOC> = 1, the rising edge or falling edge will be detected according to the setting of the SC0CR<SCLKS> register to generate the basic clock.

- In UART Mode

The SC0MOD0 <SC1 to SC0> setting determines whether the baud rate generator clock, the internal system clock fSYS, the match detect signal from timer TMRA0 or the external clock (SCLK0) is used to generate the basic clock SIOCLK.

(4) Receiving counter

The receiving counter is a 4-bit binary counter used in UART Mode which counts up the pulses of the SIOCLK clock. It takes 16 SIOCLK pulses to receive 1 bit of data; each data bit is sampled three times – on the 7th, 8th and 9th clock cycles.

The value of the data bit is determined from these three samples using the majority rule.

For example, if the data bit is sampled respectively as 1, 0 and 1 on 7th, 8th and 9th clock cycles, the received data bit is taken to be 1. A data bit sampled as 0, 0 and 1 is taken to be 0.

(5) Receiving control

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = 0, the RXD0 signal is sampled on the rising edge of the shift clock which is output on the SCLK0 pin.

In SCLK Input Mode with the setting SC0CR<IOC> = 1, the RXD0 signal is sampled on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting.

- In UART Mode

The receiving control block has a circuit which detects a start bit using the majority rule. Received bits are sampled three times; when two or more out of three samples are 0, the bit is recognized as the start bit and the receiving operation commences.

The values of the data bits that are received are also determined using the majority rule.

(6) The Receiving Buffers

To prevent Overrun errors, the Receiving Buffers are arranged in a double-buffer structure.

Received data is stored one bit at a time in Receiving Buffer 1 (which is a shift register). When 7 or 8 bits of data have been stored in Receiving Buffer 1, the stored data is transferred to Receiving Buffer 2 (SC0BUF); this causes an INTRX0 interrupt to be generated. The CPU only reads Receiving Buffer 2 (SC0BUF). Even before the CPU has finished reading the contents of Receiving Buffer 2 (SC0BUF), more data can be received and stored in Receiving Buffer 1. However, if Receiving Buffer 2 (SC0BUF) has not been read completely before all the bits of the next data item are received by Receiving Buffer 1, an Overrun error occurs. If an Overrun error occurs, the contents of Receiving Buffer 1 will be lost, although the contents of Receiving Buffer 2 and SC0CR<RB8> will be preserved.

SC0CR<RB8> is used to store either the parity bit – added in 8-Bit UART Mode – or the most significant bit (MSB) – in 9-Bit UART Mode.

In 9-Bit UART Mode the wake-up function for the slave controller is enabled by setting SC0MOD0<WU> to 1; in this mode INTRX0 interrupts occur only when the value of SC0CR<RB8> is 1.

(7) Transmission counter

The transmission counter is a 4-bit binary counter which is used in UART Mode and which, like the receiving counter, counts the SIOCLK clock pulses; a TXDCLK pulse is generated every 16 SIOCLK clock pulses.

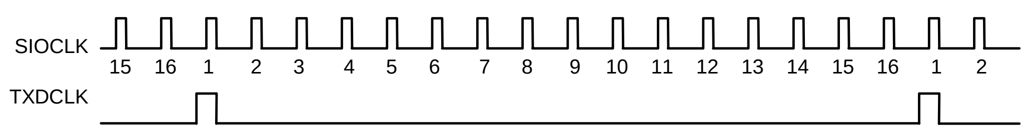


Figure 3.10.4 Generation of the transmission clock

(8) Transmission controller

- In I/O Interface Mode

In SCLK Output Mode with the setting SC0CR<IOC> = 0, the data in the Transmission Buffer is output one bit at a time to the TXD0 pin on the rising edge of the shift clock which is output on the SCLK0 pin.

In SCLK Input Mode with the setting SC0CR<IOC> = 1, the data in the Transmission Buffer is output one bit at a time on the TXD0 pin on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting.

- In UART Mode

When transmission data sent from the CPU is written to the Transmission Buffer, transmission starts on the rising edge of the next TXDCLK, generating a transmission shift clock TXDSFT.

Handshake function

Serial Channels 0 and 1 each have a $\overline{\mathrm{CTS0}}$ pin. Use of this pin allows data can be sent in units of one frame; thus, Overrun errors can be avoided. The handshake functions is enabled or disabled by the SC0MOD <CTSE> setting.

When the $\overline{\mathrm{CTS0}}$ pin foes High on completion of the current data send, data transmission is halted until the $\overline{\mathrm{CTS0}}$ pin foes Low again. However, the INTTX0 Interrupt is generated, it requests the next data send to the CPU. The next data is written in the Transmission Buffer and data sending is halted.

Although there is no $\overline{\mathrm{RTS}}$ pin, a handshake function can easily be configured by assigning any port to perform the $\overline{\mathrm{RTS}}$ function. The RTS should be output High to request send data halt after data receive is completed by software in the RXD interrupt routine.

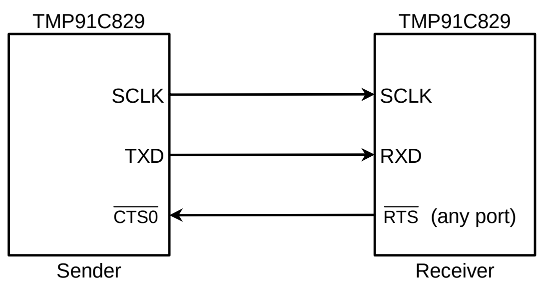
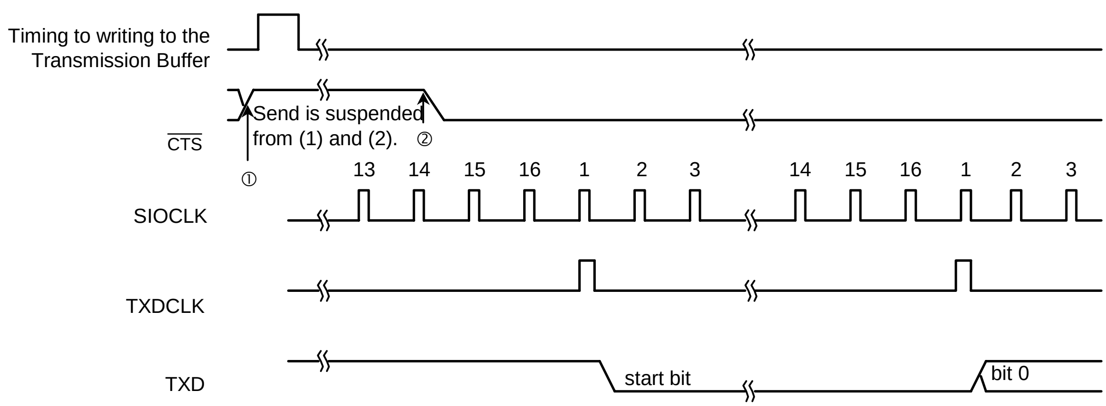


Figure 3.10.5 Handshake function



Note 1: If the $\overline{\mathrm{CTS}}$ signal goes High during transmission, no more data will be sent after completion of the current transmission.

Note 2: Transmission starts on the first falling edge of the TXDCLK clock after the $\overline{\mathrm{CTS}}$ signal has fallen.

Figure 3.10.6 $\overline{\mathrm{CTS}}$ (Clear to send) Timing

(9) Transmission Buffer

The Transmission Buffer (SC0BUF) shifts out and sends the transmission data written from the CPU, in order one bit at a time starting with the least significant bit (LSB) and finishing with the most significant bit (MSB). When all the bits have been shifted out, the empty Transmission Buffer generates an INTTX0 interrupt.

(10) Parity control circuit

When SC0CR<PE> in the Serial Channel Control Register is set to 1, it is possible to transmit and receive data with parity. However, parity can be added only in 7-Bit UART Mode or 8-Bit UART Mode. The SC0CR<EVEN> field in the Serial Channel Control Register allows either even or odd parity to be selected.

In the case of transmission, parity is automatically generated when data is written to the Transmission Buffer SC0BUF. The data is transmitted after the parity bit has been stored in SC0BUF<TB7> in 7-Bit UART Mode or in SC0MOD0<TB8> in 8-Bit UART Mode. SC0CR<PE> and SC0CR<EVEN> must be set before the transmission data is written to the Transmission Buffer.

In the case of receiving, data is shifted into Receiving Buffer 1, and the parity is added after the data has been transferred to Receiving Buffer 2 (SC0BUF), and then compared with SC0BUF<RB7> in 7-Bit UART Mode or with SC0CR<RB8> in 8-Bit UART Mode. If they are not equal, a Parity error is generated and the SC0CR<PERR> flag is set.

(11) Error flags

Three error flags are provided to increase the reliability of data reception.

1. Overrun error <OERR>

   If all the bits of the next data item have been received in Receiving Buffer 1 while valid data still remains stored in Receiving Buffer 2 (SC0BUF), an Overrun error is generated.

2. Parity error <PERR>

   The parity generated for the data shifted into Receiving Buffer 2 (SC0BUF) is compared with the parity bit received via the RXD pin. If they are not equal, a Parity error is generated.

3. Framing error <FERR>

   The stop bit for the received data is sampled three times around the center. If the majority of the samples are 0, a Framing error is generated.

(12) Timing generation

① In UART Mode

Receiving

Mode	9-Bit (Note)	8-Bit + Parity (Note)	8-Bit, 7-Bit + Parity, 7-Bit
Interrupt timing	Center of last bit (bit 8)	Center of last bit (parity bit)	Center of stop bit
Framing error timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity error timing	-	Center of last bit (parity bit)	Center of stop bit
Overrun error timing	Center of last bit (bit 8)	Center of last bit (parity bit)	Center of stop bit

Note: In 9-Bit Mode and 8-Bit + Parity Mode, interrupts coincide with the ninth bit pulse. Thus, when servicing the interrupt, it is necessary to allow a 1-bit period to elapse (so that the stop bit can be transferred) in order to allow proper framing error checking.

Transmitting

Mode	9-Bit	8-Bit + Parity	8-Bit, 7-Bit + Parity, 7-Bit
Interrupt timing	Just before stop bit is transmitted	Just before last data bit is transmitted	Just before last data bit is transmitted

② I/O interface

Transmission Interrupt timing	SCLK Output Mode	Immediately after rise of last SCLK signal. (See Figure 3.10.19)
	SCLK Input Mode	Immediately after rise of last SCLK signal Rising Mode, or immediately after fall in Falling Mode. (See Figure 3.10.20)
Receiving Interrupt timing	SCLK Output Mode	Timing used to transfer received to data Receive Buffer 2 (SC0BUF) (i.e. immediately after last SCLK). (See Figure 3.10.21)
	SCLK Input Mode	Timing used to transfer received data to Receive Buffer 2 (SC0BUF) (i.e. immediately after last SCLK). (See Figure 3.10.22)

3.10.3 SFR

SC0MOD0
(0202H)

	7	6	5	4	3	2	1	0
Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function	Transfer data bit 8	Hand shake 0: CTS disable 1: CTS enable	Receive function 0: Receive disable 1: Receive enable	Wake up function 0: disable 1: enable	Serial Transmission Mode 00: I/O interface Mode 01: 7-bit UART Mode 10: 8-bit UART Mode 11: 9-bit UART Mode		Serial transmission clock (UART) 00: TMRA0 trigger 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clcok (SCLK0 input)	

Serial transmission clock source (UART)

00	Timer TMRA0 match detect signal
01	Baud rate generator
10	Internal clock f_{SYS}
11	External clock (SCLK0 input)

Note: The clock selection for the I/O interface mode is controlled by the serial bontrol register (SC0CR).

Serial Transmission Mode

00	I/O Interface Mode	
01	UART	7-bit mode
10		8-bit mode
11		9-bit mode

Wake-up function

	9-Bit UART	Other Modes
0	Interrupt generated when data is received	Don't care
1	Interrupt generated only when RB8 = 1	

Receiving Function

0	Receive disabled
1	Receive enabled

Handshake function ($\overline{\text{CTS}}$ pin) Enable

0	Disabled (always transferable)
1	Enabled

Transmission data bit 8

Figure 3.10.7 Serial Mode Control Register (channel 0, SC0MOD0)

SC1MOD0
(0209H)

	7	6	5	4	3	2	1	0
Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function	Transfer data bit 8	Hand shake 0: CTS disable 1: CTS enable	Receive function 0: Receive disable 1: Receive enable	Wake up function 0: disable 1: enable	Serial Transmission Mode 00: I/O interface Mode 01: 7-bit UART Mode 10: 8-bit UART Mode 11: 9-bit UART Mode		Serial transmission clock (UART) 00: TMRA0 trigger 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clcok (SCLK0 input)	

Serial transmission clock source (UART)

00	Timer TMRA0 match detect signal
01	Baud rate generator
10	Internal clock f_{SYS}
11	External clock (SCLK0 input)

Note: The clock selection for the I/O interface mode is controlled by the serial bontrol register (SC0CR).

Serial Transmission Mode

00	I/O Interface Mode	
01	UART	7-bit mode
10		8-bit mode
11		9-bit mode

Wake-up function

	9-Bit UART	Other Modes
0	Interrupt generated when data is received	Don't care
1	Interrupt generated only when RB8 = 1	

Receiving Function

0	Receive disabled
1	Receive enabled

Handshake function ($\overline{CTS}$ pin) Enable

0	Disabled (always transferable)
1	Enabled

Transmission data bit 8

Figure 3.10.8 Serial Mode Control Register (channel 1, SC1MOD0)

SC0CR
(0201H)

	7	6	5	4	3	2	1	0
Bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (cleared to 0 when read)			R/W	
After Reset	0	0	0	0	0	0	0	0
Function	Received data bit 8	Parity 0: odd 1: even	Parity addition 0: disable 1: enable	1: error Overrun	Parity	Framing	0: SCLK0 1: SCLK0	0: baud rate generator 1: SCLK0 pin input

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Edge selection for SCLK pin

0	Transmits and receivers data on rising edge of SCLK0.
1	Transmits and receivers data on falling edge SCLK0.

Framing Error flag
Parity Error flag
Overrun Error flag
} cleared to 0 when read

Parity addition enable

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data 8

Note: As all error flags are cleared after reading do not test only a single bit with a bit-testing instruction.

Figure 3.10.9 Serial Control Register (channel 0, SC0CR)

SC1CR (0208H)

	7	6	5	4	3	2	1	0
Bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (cleared to 0 when read)			R/W	
After Reset	0	0	0	0	0	0	0	0
Function	Received data bit 8	Parity 0: odd 1: even	Parity addition 0: disable 1: enable	1: error Overrun	Parity	Framing	0: SCLK0 1: SCLK0	0: baud rate generator 1: SCLK0 pin input

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Edge selection for SCLK pin

0	Transmits and receivers data on rising edge of SCLK0.
1	Transmits and receivers data on falling edge SCLK0.

Framing Error flag
Parity Error flag
Overrun Error flag
} cleared to 0 when read

Parity addition enable

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data 8

Note: As all error flags are cleared after reading do not test only a single bit with a bit-testing instruction.

Figure 3.10.10 Serial Control Register (channel 1, SC1CR)

BROCR
(0203H)

	7	6	5	4	3	2	1	0
Bit symbol	—	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function	Received data Bit 8	+(16–K)/16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Setting of the Divided frequency			

+(16 – K)/16 division enable

0	Disable
1	Enable

Setting the input clock of baud rate generator

00	Internal clock ϕT0
01	Internal clock ϕT2
10	Internal clock ϕT8
11	Internal clock ϕT32

BR0ADD
(0204H)

	7	6	5	4	3	2	1	0
Bit symbol					BR0K3	BR0K2	BR0K1	BR0K0
Read/Write					R/W			
After Reset					0	0	0	0
Function					Sets frequency divisor K (divided by N = (16 – K)/16)			

Sets baud rate generator frequency divisor

DR0ADD <BR0K3:0> \ BR0CR <BR0S3:0>	BR0CR<BR0ADDE> = 1: 0000 (N = 16) or 0001 (N = 1)	BR0CR<BR0ADDE> = 1: 0000 (N = 2) or 1111 (N = 15)	BR0CR<BR0ADDE> = 0: 0001 (N 1) (UART only) to 1111 (N = 15) 0000 (N = 16)
0000	Disable	Disable	Divided by N
0001 (K = 1) to 1111 (K = 15)	Disable	Divided by $N + \frac{(16-K)}{16}$	

Note 1: The baud rate generator can be set 1 when UART mode and disable + (16 – K)/16 division function. Don't use in I/O interface mode.

Note 2: Set BR0CR <BR0ADDE> to 1 after setting K (K = 1 to 15) to BR0ADD<BR0K3 to 0> when + (16 – K)/16 division function is used.

Note 3: + (16 – K)/16 division function is possible to use in only UART mode.
Set BR0CR <BR0ADDE> to 0 and disable + (16 – K)/16 division function in I/O interface mode.

Figure 3.10.11 Baud rate generator control (channel 0、BR0CR, BR0ADD)

BR1CR (020AH)

	7	6	5	4	3	2	1	0
Bit symbol	—	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function	Received data Bit 8	+(16–K)/16 division 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Setting of the Divided frequency			

+(16 – K)/16 division enable

0	Disable
1	Enable

Setting the input clock of baud rate generator

00	Internal clock φT0
01	Internal clock φT2
10	Internal clock φT8
11	Internal clock φT32

BR1ADD (020BH)

	7	6	5	4	3	2	1	0
Bit symbol					BR1K3	BR1K2	BR1K1	BR1K0
Read/Write					R/W			
After Reset					0	0	0	0
Function					Sets frequency divisor K (divided by N = (16 – K)/16)			

Sets baud rate generator frequency divisor

DR1ADD <BR1K3:0> \ BR1CR <BR1S3:0>	BR0CR<BR1ADDE> = 1: 0000 (N = 16) or 0001 (N = 1)	BR0CR<BR1ADDE> = 1: 0000 (N = 2) or 1111 (N = 15)	BR1CR<BR1ADDE> = 0: 0001 (N 1) (UART only) to 1111 (N = 15) 0000 (N = 16)
0000	Disable	Disable	Divided by N
0001 (K = 1) to 1111 (K = 15)	Disable	Divided by $N + \frac{(16 - K)}{16}$	Divided by N

Note 1: The baud rate generator can be set 1 when UART mode and disable + (16 – K)/16 division function. Don't use in I/O interface mode.

Note 2: Set BR1CR <BR1ADDE> to 1 after setting K (K = 1 to 15) to BR1ADD<BR1K3 to 0> when + (16 – K)/16 division function is used.

Note 3: + (16 – K)/16 division function is possible to use in only UART mode.
Set BR1CR <BR1ADDE> to 0 and disable + (16 – K)/16 division function in I/O interface mode.

Figure 3.10.12 Baud rate generator control (channel 1、BR1CR, BR1ADD)

SC0BUF
(0200H)

	7	6	5	4	3	2	1	0	
	TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)
	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Reveiving)

Note: Prohibit Read modify write for SC0BUF.

Figure 3.10.13 Serial Transmission/Receiving Buffer Registers (channel 0, SC0BUF)

SC0MOD1
(0205H)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0						STSEN0
Read/Write	R/W	R/W						W
After Reset	0	0						1
Function	IDLE2 0: Stop 1: Run	duplex 0: half 1: full						STS0 0:Enable 1:Disable

Figure 3.10.14 Serial Mode Control Register 1 (channel 0, SC0MOD1)

SC1BUF
(0207H)

	7	6	5	4	3	2	1	0	
	TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)
	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Reveiving)

Note: Prohibit Read modify write for SC1BUF.

Figure 3.10.15 Serial Transmission/Receiving Buffer Registers (channel 1, SC1BUF)

SC1MOD1
(020CH)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0						STSEN0
Read/Write	R/W	R/W						W
After Reset	0	0						1
Function	IDLE2 0: Stop 1: Run	duplex 0: half 1: full						STS1 0:Enable 1:Disable

Figure 3.10.16 Serial Mode Control Register 1 (channel 1, SC1MOD1)

3.10.4 Operation in each mode

(1) Mode 0 (I/O Interface Mode)

This mode allows an increase in the number of I/O pins available for transmitting data to or receiving data from an external shift register.

This mode includes the SCLK output mode to output synchronous clock SCLK and SCLK input external synchronous clock SCLK.

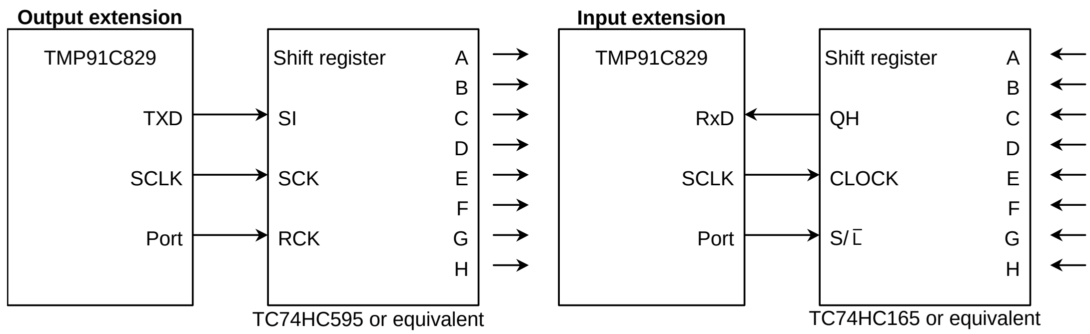


Figure 3.10.17 SCLK Output Mode connection example

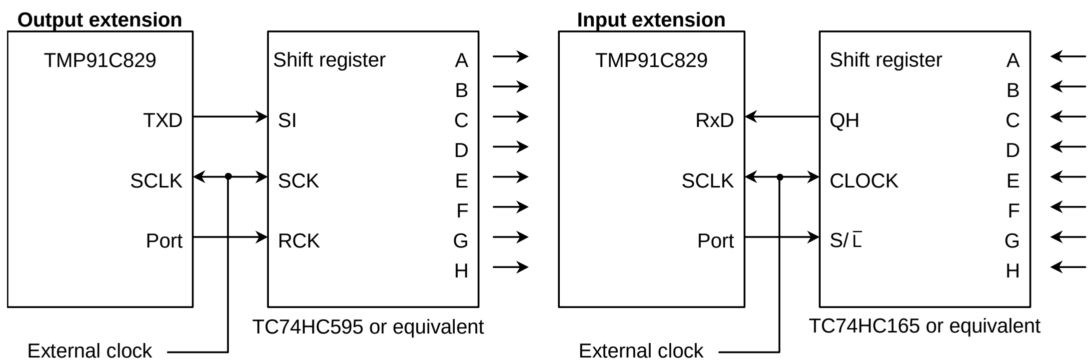


Figure 3.10.18 Example of SCLK Input Mode Connection

① Transmission

In SCLK Output Mode 8-bit data and a synchronous clock are output on the TXD0 and SCLK0 pins respectively each time the CPU writes the data to the Transmission Buffer.

When all the data has been output, INTES0 <ITX0C> is set to 1, causing an INTTX0 interrupt to be generated.

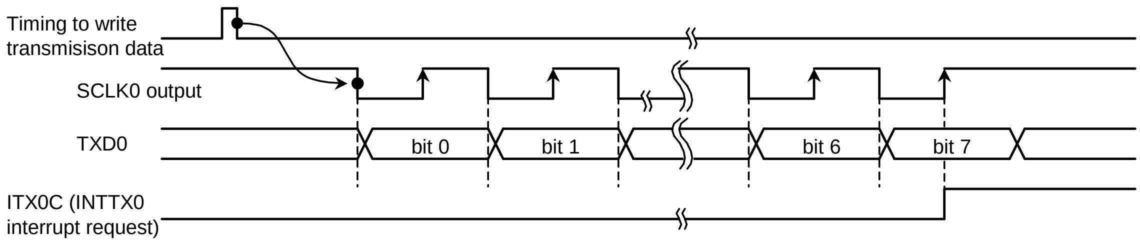


Figure 3.10.19 Transmitting Operation in I/O Interface Mode (SCLK0 Output Mode) (Channel 0)

In SCLK Input Mode, 8-bit data is output on the TXD0 pin when the SCLK0 input becomes Active after the data has been written to the Transmission Buffer by the CPU.

When all the data has been output, INTES0 <ITX0C> is set to 1, causing an INTTX0 interrupt to be generated.

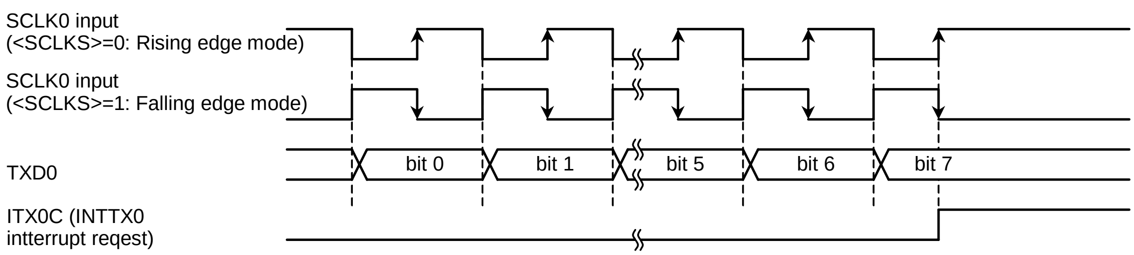


Figure 3.10.20 Transmitting Operation in I/O Interface Mode (SCLK0 Input Mode) (channel 0)

② Receiving

In SCLK Output Mode the synchronous clock is output on the SCLK0 pin and the data is shifted to Receiving Buffer 1. This is initiated when the Receive Interrupt flag INTES0<IRX0C> is cleared as the received data is read. When 8-bit data is received, the data is transferred to Receiving Buffer 2 (SC0BUF) following the timing shown below and INTES0<IRX0C> is set to 1 again, causing an INTRX0 interrupt to be generated.

Setting SC0MOD0<RXE>to 1 initiates SCLK0 output.

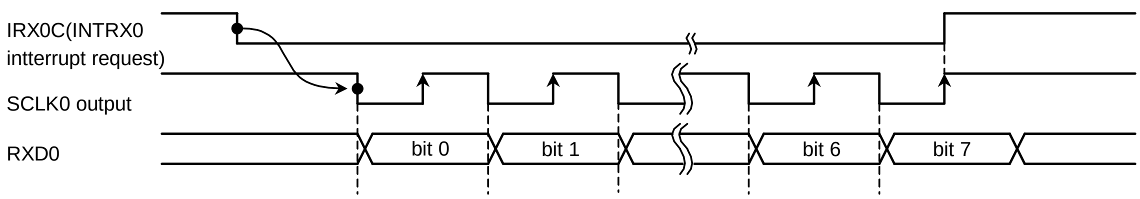


Figure 3.10.21 Receiving operation in I/O Interface Mode (SCLK0 Output Mode) (Channel 0)

In SCLK Input Mode the data is shifted to Receiving Buffer 1 when the SCLK input goes Active. The SCLK input goes Active when the Receive Interrupt flag INTES0 <IRX0C> is cleared as the received data is read. When 8-bit data is received, the data is shifted to Receiving Buffer 2 (SC0BUF) following the timing shown below and INTES0 <IRX0C> is set to 1 again, causing an INTRX0 interrupt to be generated.

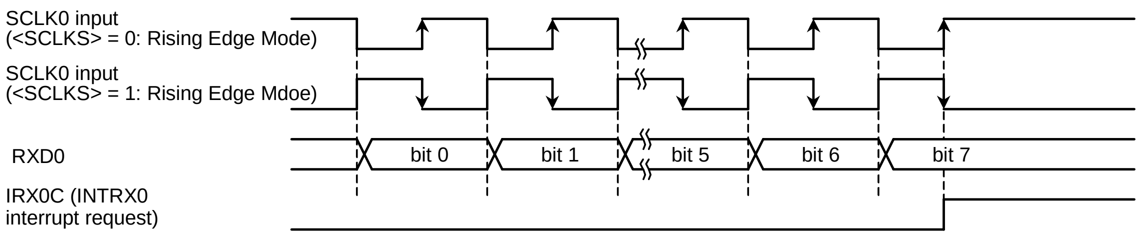


Figure 3.10.22 Receiving Operation in I/O interface Mode (SCLK0 Input Mode) (Channel 0)

Note: The system must be put in the Receive Enable state (SCMOD0<RXE> = 1) before data can be received.

③ Transmission and Receiving (Full Duplex Mode)

When Full Duplex Mode is used, set the Receive Interrupt Level to 0 and set enable the level of transmit interrupt. Ensure that the program which transmits the interrupt reads the receiving buffer before setting the next transmit data.

The following is an example of this:

Example: Channel 0, SCLK output
Baud rate = 9600 bps
fc = 14.7456 MHz
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

Main routine

	7	6	5	4	3	2	1	0	
									Set the INTTX0 level to 1.
INTES0	0	0	0	1	0	0	0	0	Set the INTRX0 level to 0.
P8CR	–	–	–	–	–	1	0	1	Set P80, P81 and P82 to function as the TXD0, RXD0 and SCLK0 pins respectively.
P8FC	–	–	–	–	–	1	–	1	
SC0MOD0	0	0	0	0	0	0	0	0	Select I/O Interface Mode.
SC0MOD1	1	1	0	0	0	0	0	0	Select Full Duplex Mode.
SC0CR	0	0	0	0	0	0	0	0	Sclk_out, transmit on negative edge, receive on positive edge
BR0CR	0	0	1	1	0	0	1	1	Baud rate = 9600 bps
SC0MOD0	0	0	1	0	0	0	0	0	Enable receiving
SC0BUF	*	*	*	*	*	*	*	*	Set the transmit data and start.

INTTX0 interrupt routine

Acc SC0BUF									Read the receiving buffer.
SC0BUF	–	–	X	X	–	1	X	X	Set the next transmit data.

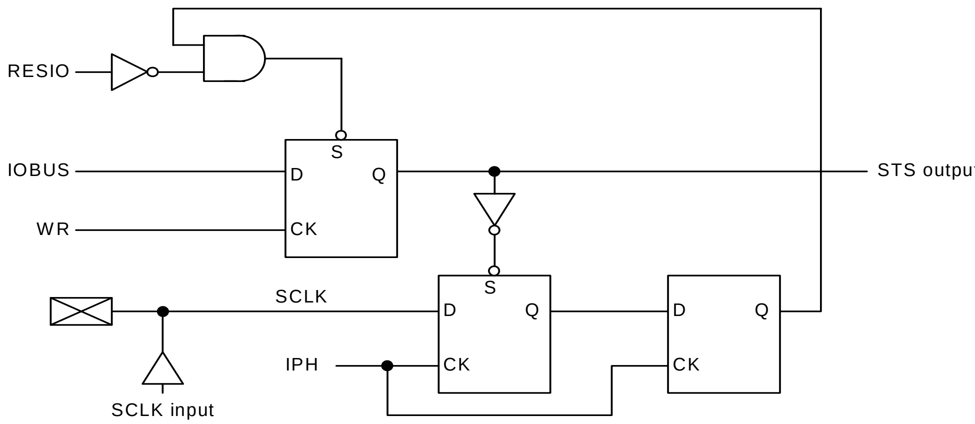
Note: X = Don't care; "–" = No change

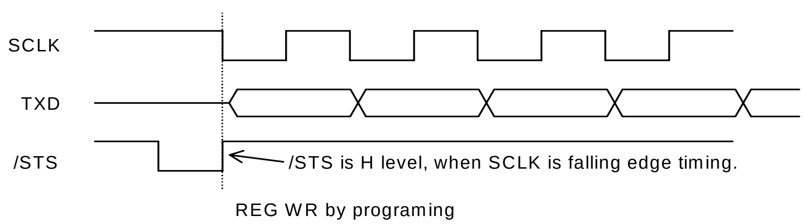
This UPU have STS0, STS1 pins that request the next data send to the CPU. P8CR sets to output mode, P8FC sets STS using mode, and bit 0 of SC0MOD1 (SC1MOD1) register sets H revel. And then STS is enable to start to transfer the data.

When SCLK signal is exactly falling edge, STS is disable.

And when it is ended to transfer 8-bits data, you set the STS function is Enable and you set to request to the another CPU the next data.

In SCLK output mode you can not use this STS function.



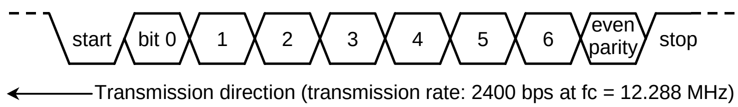


(2) Mode 1 (7-bit UART Mode)

7-Bit UART Mode is selected by setting the Serial Channel Mode Register SC0MOD0<SM1,SM0> field to 01.

In this mode a parity bit can be added. Use of a parity bit is enabled or disabled by the setting of the Serial Channel Control Register SC0CR<PE> bit; whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to 1 (enabled).

Setting example: When transmitting data of the following format, the control registers should be set as described below. This explanation applies to Channel 0.



* Clock state
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: System clock

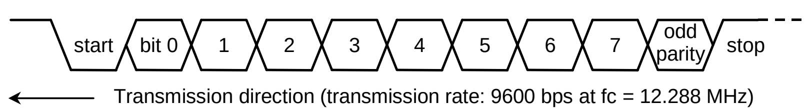
		7	6	5	4	3	2	1	0	
P8CR	←	–	–	–	–	–	–	–	1	Set P80 to function as the TXD0 pin.
P8FC	←	–	–	–	–	–	–	–	1	
SC0MOD	←	X	0	–	X	0	1	0	1	Select 7-Bit UART Mode.
SC0CR	←	X	1	1	X	X	X	0	0	Add even parity.
BR0CR	←	0	0	1	0	0	1	0	1	Set the transfer rate to 2400 bps.
INTES0	←	1	1	0	0	–	–	–	–	Enable the INTTX0 interrupt and set it to Interrupt Level 4.
SC0BUF	←	*	*	*	*	*	*	*	*	Set data for transmission.

Note: X = Don’t care; “–” = No change

(3) Mode 2 (8-Bit UART Mode)

8-Bit UART Mode is selected by setting SC0MOD0<SM1,SM0> to 10. In this mode a parity bit can be added (use of a parity bit is enabled or disabled by the setting of SC0CR<PE>); whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to 1 (enabled).

Setting example: When receiving data of the following format, the control registers should be set as described below.



* Clock state
System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: System clock

Main settings

```
          7 6 5 4 3 2 1 0
P8CR   ←  – – – – – – 0 –     Set P80 to function as the TXD0 pin.
SC0MOD ←  – 0 1 X 1 0 0 1     Enable receiving in 8-Bit UART Mode.
SC0CR  ←  X 0 1 X X X 0 0     Add even parity.
BR0CR  ←  0 0 0 1 0 1 0 1     Set the transfer rate to 9600 bps.
INTES0 ←  – – – – 1 1 0 0     Enable the INTTX0 interrupt and set it to Interrupt Level 4.
```

Interrupt processing

```
Acc     ← SC0CR AND 00011100  } Check for errors.
if Acc  ≠ 0 then ERROR        }
Acc     ← SC0BUF                Read the received data.
```

Note: X = Don't care; "–" = No change

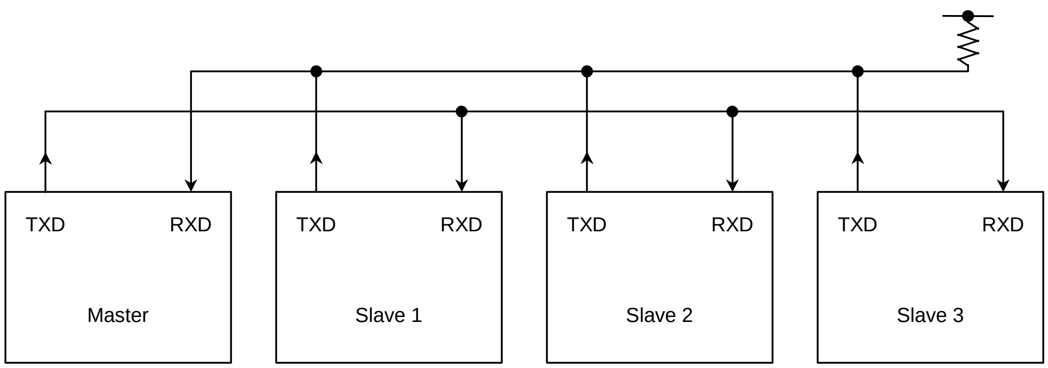
(4) Mode 3 (9-Bit UART Mode)

9-Bit UART Mode is selected by setting SC0MOD0<SM1,SM0> to 11. In this mode parity bit cannot be added.

In the case of transmission the MSB (9th bit) is written to SC0MOD0<TB8>. In the case of receiving it is stored in SC0CR<RB8>. When the buffer is written and read, the MSB is read or written first, before the rest of the SC0BUF data.

Wake-up function

In 9-Bit UART Mode, the wake-up function for slave controllers is enabled by setting SC0MOD0<WU> to 1. The interrupt INTRX0 can only be generated when<RB8> = 1.



Note: The TXD pin of each slave controller must be in Open-Drain Output Mode.

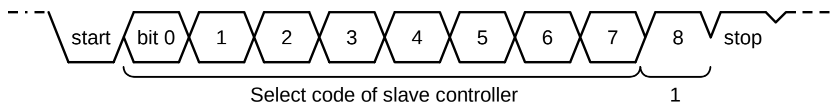
Figure 3.10.23 Serial Link using Wake-up function

Protocol

① Select 9-Bit UART Mode on the master and slave controllers.

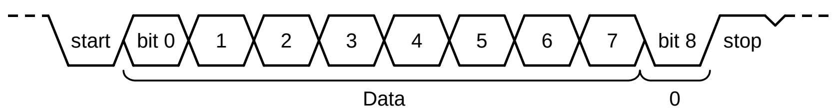
② Set the SC0MOD0<WU> bit on each slave controller to 1 to enable data receiving.

③ The master controller transmits data one frame at a time. Each frame includes an 8-bit select code which identifies a slave controller. The MSB (bit 8) of the data (<TB8>) is set to 1.



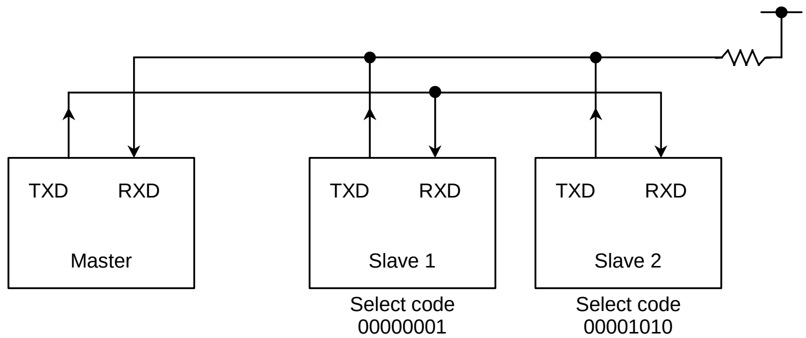
④ Each slave controller receives the above frame. Each controller checks the above select code against its own select code. The controller whose code matches clears its WU bit to 0.

⑤ The master controller transmits data to the specified slave controller (the controller whose SC0MOD<WU> bit has been cleared to 0). The MSB (bit 8) of the data (<TB8>) is cleared to 0.



⑥ The other slave controllers (whose <WU> bits remain at 1) ignore the received data because their MSBs (bit 8 or <RB8>) are set to 0, disabling INTRX0 interrupts.
The slave controller whose WU bit = 0 can also transmit to the master controller. In this way it can signal the master controller that the data transmission from the master controller has been completed.

Setting example: To link two slave controllers serially with the master controller using the internal clock f_{SYS} as the transfer clock.



Since Serial Channels 0 and 1 operate in exactly the same way, Channel 0 only is used for the purposes of this explanation.

- Setting the master controller

Main

P8CR	← - - - - - - 0 1	Set P80 and P81 to function as the TXD0 and RXD0 pins respectively.
P8FC	← - - - - - - X 1	
INTES0	← 1 1 0 0 1 1 0 1	Enable the INTTX0 interrupt and set it to Interrupt Level 4. Enable the INTRX0 interrupt and set it to Interrupt Level 5.
SC0MOD0	← 1 0 1 0 1 1 1 0	Set f_{SYS} as the transmission clock for 9-Bit UART Mode.
SC0BUF	← 0 0 0 0 0 0 0 1	Set the select code for slave controller 1.

INTTX0 interrupt

SC0MOD0	← 0 - - - - - - -	Set TB8 to 0.
SC0BUF	← * * * * * * * *	Set data for transmission.

- Setting the slave controller

Main

P8CR	← - - - - - - 0 1	Select P81 and P80 to function as the RXD0 and TXD0 pins respectively (open-drain output).
P8FC	← - - - - - - X 1	
ODE	← X X X X X X – 1	
INTES0	← 1 1 0 1 1 1 1 0	Enable INTRX0 and INTTX0.
SC0MOD0	← 0 0 1 1 1 1 1 0	Set <WU> to 1 in 9-Bit UART Transmission Mode using f_{SYS} as the transfer clock.

INTRX0 interrupt

Acc ← SC0BUF
if Acc = select code
then SC0MOD0 ← - - - 0 - - - - Clear <WU> to 0.

3.11 Analog/Digital Converter

The TMP91C829 incorporates a 10-bit successive approximation-type analog/digital converter (AD converter) with 8-channel analog input.

Figure 3.11.1 is a block diagram of the AD converter. The 8-channel analog input pins (AN0 to AN7) are shared with the input-only port Port A and can thus be used as an input port.

Note: When IDLE2, IDLE1 or STOP Mode is selected, so as to reduce the power, with some timings the system may enter a standby mode even though the internal comparator is still enabled. Therefore be sure to check that AD converter operations are halted before a HALT instruction is executed.

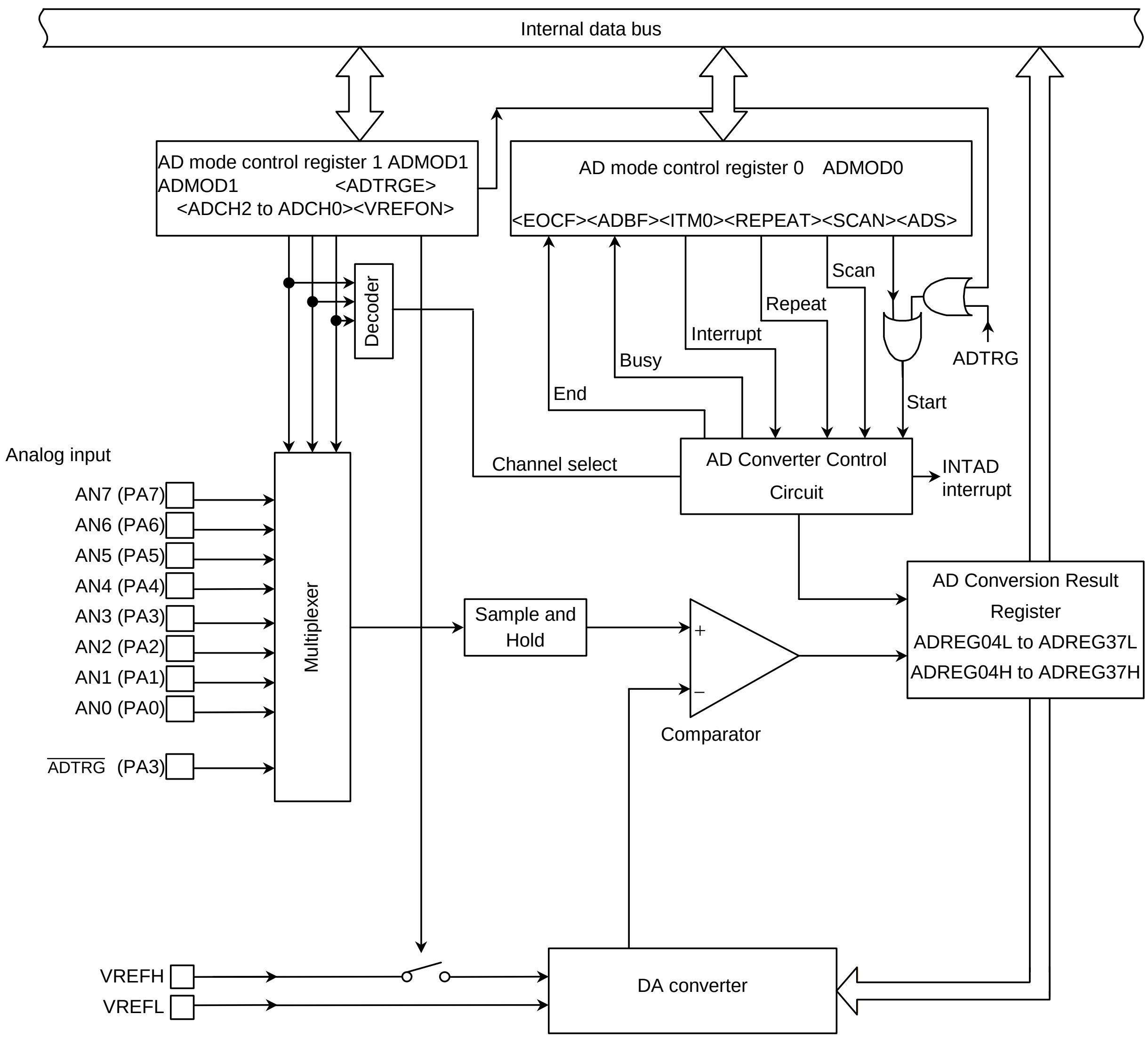


Figure 3.11.1 Block diagram of AD converter

3.11.1 Analog/Digital converter registers

The AD converter is controlled by the two AD Mode Control Registers: ADMOD0 and ADMOD1. The eight AD Conversion Data Upper and Lower Registers (ADREG04H/L, ADREG15H/L, ADREG26H/L and ADREG37H/L) store the results of AD conversion.

Figure 3.11.2 shows the registers related to the AD converter.

AD Mode Control Register 0

ADMOD0 (02B0H)

	7	6	5	4	3	2	1	0
Bit symbol	EOCF	ADBF	—	—	ITM0	REPEAT	SCAN	ADS
Read/Write	R		R/W					
After Reset	0	0	0	0	0	0	0	0
Function	AD Conversion End flag 0: conversion in Progress 1: conversion complete	AD Conversion Busy flag 0: conversion stopped 1: conversion in progress	Note: Always fixed to 0	Note: Always fixed to 0	Interrupt specification in conversion channel fixed repeat mode 0: every conversion 1: every fourth conversion	repeat mode specification Single Conversion 1: Repeat Conversion Mode	Scan mode specification 0: Conversion Channel Fixed Mode 1: Conversion Channel Scan Mode	AD conversion start 0: Don't care 1: start conversion Always 0 when read

AD conversion start

0	Don't care
1	Start AD conversion

Note: Always read as 0.

AD scan mode setting

0	AD Conversion Channel Fixed Mode
1	AD Conversion Channel Scan Mode

AD repeat mode setting

0	AD Single Conversion Mode
1	AD Repeat Conversion Mode

Specify AD conversion interrupt for Channel Fixed Repeat Conversion Mode

	Channel Fixed Repeat Conversion Mode <SCAN> = 0, <REPEAT> = 1
0	Generates interrupt every conversion.
1	Generates interrupt every fourth conversion.

AD Conversion Busy flag

0	AD conversion stopped
1	AD conversion in progress

AD Conversion End flag

0	Before or during AD conversion
1	AD conversion complete

Figure 3.11.2 AD Converter Related Register

AD Mode Control Register 1

		7	6	5	4	3	2	1	0
ADMOD1 (02B1H)	Bit symbol	VREFON	I2AD			ADTRGE	ADCH2	ADCH1	ADCH0
	Read/Write	R/W	R/W			R/W			
	After Reset	0	0			0	0	0	0
	Function	VREF application control 0: OFF 1: ON	IDLE2 0: Stop 1: Operate			AD external trigger start control 0: disable 1: enable	Analog input channel selection		

Analog input channel selection

<SCAN> / <ADCH2, ADCH1, ADCH0>	0 (channel fixed)	1 (channel scanned)
000	AN0	AN0
001	AN1	AN0 → AN1
010	AN2	AN0 → AN1 → AN2
011	AN3	AN0 → AN1 → AN2 → AN3
100	AN4	AN4
101	AN5	AN4 → AN5
110	AN6	AN4 → AN5 → AN6
111	AN7	AN4 → AN5 → AN6 → AN7

AD conversion start control by external trigger ($\overline{\text{ADTRG}}$ input)

0	Disabled
1	Enabled

IDLE2 control

0	Stopped
1	In operation

Control of application of reference voltage to AD converter

0	OFF
1	ON

Before starting conversion (before writing 1 to ADMOD0 <ADS>), set the <VREFON> bit to 1.

Figure 3.11.3 AD Converter Related Register

AD Conversion Data Low Register 0/4

ADREG04L (02A0H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR01	ADR00						ADR0RF
Read/Write	R							R
After Reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD Conversion Data Storage flag 1: Conversion result stored

AD Conversion Data Upper Register 0/4

ADREG04H (02A1H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
Read/Write	R							
After Reset	Undefined							
Function	Stores upper eight bits AD conversion result.							

AD Conversion Data Lower Register 1/5

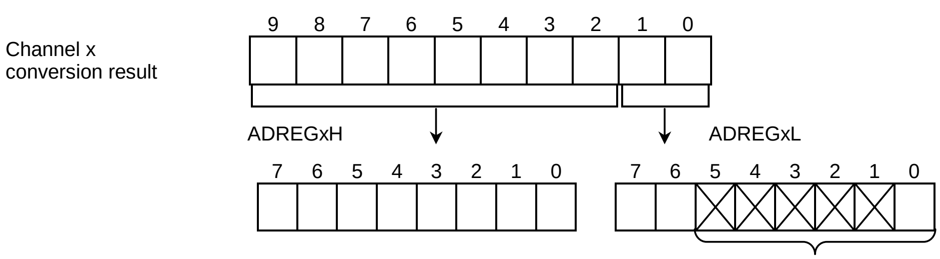
ADREG15L (02A2H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR11	ADR10						ADR1RF
Read/Write	R							R
After Reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD Conversion Result flag 1: Conversion result stored

AD Conversion Data Upper Register 1/5

ADREG15H (02A3H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
Read/Write	R							
After Reset	Undefined							
Function	Stores upper eight bits AD conversion result.							



- Bits 5-1 are always read as 1.
- Bit 0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.11.4 AD Converter Related Registers

AD Conversion Result Lower Register 2/6

		7	6	5	4	3	2	1	0
ADREG26L (02A4H)	Bit symbol	ADR21	ADR20						ADR2RF
	Read/Write	R							R
	After Reset	Undefined							0
	Function	Stores lower 2 bits of AD conversion result.							AD conversion data storage flag 1: Conversion result stored

AD Conversion Data Upper Register 2/6

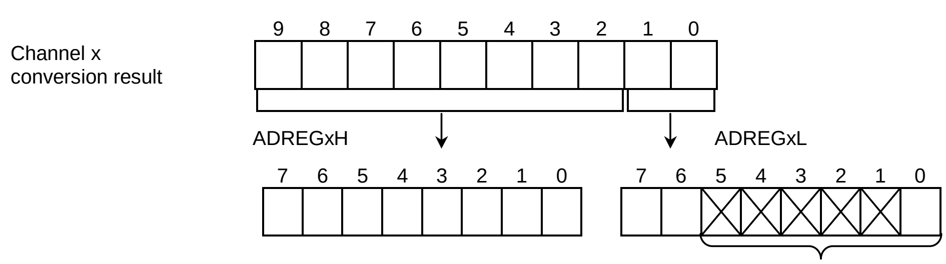
		7	6	5	4	3	2	1	0
ADREG26H (02A5H)	Bit symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper eight bits of AD conversion result.							

AD Conversion Data Lower Register 3/7

		7	6	5	4	3	2	1	0
ADREG37H (02A6H)	Bit symbol	ADR31	ADR30						ADR3RF
	Read/Write	R							R
	After Reset	Undefined							0
	Function	Stores lower 2 bits of AD conversion result							AD Date Storage 1: Conversion result stored

AD Conversion Result Upper Register 3/7

		7	6	5	4	3	2	1	0
ADREG37H (02A7H)	Bit symbol	ADR39	ADR48	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper eight bits of AD conversion result.							



- Bits 5-1 are always read as 1.
- Bit 0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.11.5 AD Converter Related Registers

3.11.2 Description of operation

(1) Analog reference voltage

A high-level analog reference voltage is applied to the VREFH pin; a low-level analog reference voltage is applied to the VREFL pin. To perform AD conversion, the reference voltage, the difference between VREFH and VREFL, is divided by 1024 using string resistance. The result of the division is then compared with the analog input voltage.

To turn off the switch between VREFH and VREFL, write a 0 to ADMOD1<VREFON> in AD Mode Control Register 1. To start AD conversion in the OFF state, first write a 1 to ADMOD1<VREFON>, wait 3 μs until the internal reference voltage stabilizes (this is not related to fc), then set ADMOD0<ADS> to 1.

(2) Analog input channel selection

The analog input channel selection varies depends on the operation mode of the AD converter.

- In Analog Input Channel Fixed Mode (ADMOD0<SCAN> = 0)
  Setting ADMOD1<ADCH2 to ADCH0> selects one of the input pins AN0 to AN7 as the input channel.
- In Analog Input Channel Scan Mode (ADMOD0<SCAN> = 1)
  Setting ADMOD1<ADCH2 to ADCH0> selects one of the four scan modes.

Table 3.11.1 illustrates analog input channel selection in each operation mode.

On a Reset, ADMOD0<SCAN> is set to 0 and ADMOD1<ADCH2 to ADCH0> is initialized to 000. Thus pin AN0 is selected as the fixed input channel. Pins not used as analog input channels can be used as standard input port pins.

Table 3.11.1 Analog input channel selection

<ADCH2 to 0>	Channel fixed <SCAN> = 0	Channel scan <SCAN> = 1
000	AN0	AN0
001	AN1	AN0 → AN1
010	AN2	AN0 → AN1 → AN2
011	AN3	AN0 → AN1 → AN2 → AN3
100	AN4	AN4
101	AN5	AN4 → AN5
110	AN6	AN4 → AN5 → AN6
111	AN7	AN4 → AN5 → AN6 → AN7

(3) Starting AD Conversion

To start AD conversion, write a 1 to ADMOD0<ADS> in AD Mode Control Register 0 or ADMOD1<ADTRGE> in AD Mode Control Register 1, pull the $\overline{\text{ADTRG}}$ pin input from High to Low. When AD conversion starts, the AD Conversion Busy flag ADMOD0<ADBF> will be set to 1, indicating that AD conversion is in progress.

Writing a 1 to ADMOD0<ADS> during AD conversion restarts conversion. At that time, to determine whether the AD conversion results have been preserved, check the value of the conversion data storage flag ADREGxxL<ADRxRF>.

During AD conversion, a falling edge input on the $\overline{\text{ADTRG}}$ pin will be ignored.

(4) AD conversion modes and the AD Conversion End interrupt

The four AD conversion modes are:

- Channel Fixed Single Conversion Mode
- Channel Scan Single Conversion Mode
- Chanel Fixed Repeat Conversion Mode
- Channel Scan Repeat Conversion Mode

The ADMOD0<REPET> and ADMOD0<SCAN> settings in AD Mode Control Register 0 determine the AD mode setting.

Completion of AD coversion triggers an INTAD AD Conversion End interrupt request. Also, ADMOD0<EOCF> will be set to 1 to indicate that AD conversion has been completed.

① Channel Fixed Single Conversion Mode

Setting ADMOD0<REPET> and ADMOD0<SCAN> to 00 selects Conversion Channel Fixed Single Conversion Mode.

In this mode data on one specified channel is converted once only. When the conversion has been completed, the ADMOD0<EOCF> flag is set to 1, ADMOD0<ADBF> is cleared to 0, and an INTAD interrupt request is generated.

② Channel Scan Single Conversion Mode

Setting ADMOD0<REPET> and ADMOD0<SCAN> to 01 selects Conversion Channel Scan Single Conversion Mode.

In this mode data on the specified scan channels is converted once only. When scan conversion has been completed, ADMOD0<EOCF> is set to 1, ADMOD0<ADBF> is cleared to 0, and an INTAD interrupt request is generated.

③ Channel Fixed Repeat Conversion Mode

Setting ADMOD0<REPET> and ADMOD0<SCAN> to 10 selects Conversion Channel Fixed Repeat Conversion Mode.

In this mode data on one specified channel is converted repeatedly. When conversion has been completed, ADMOD0<EOCF> is set to 1 and ADMOD0<ADBF> is not cleared to 0 but held at 1. INTAD interrupt request generation timing is determined by the setting of ADMOD0<ITM0>.

Setting <ITM0> to 0 generates an interrupt request every time an AD conversion is completed.

Setting <ITM0> to 1 generates an interrupt request on completion of every fourth conversion.

④ Channel Scan Repeat Conversion Mode

Setting ADMOD0<REPET> and ADMOD0<SCAN> to 11 selects Conversion Channel Scan Repeat Conversion Mode.

In this mode data on the specified scan channels is converted repeatedly. When each scan conversion has been completed, ADMOD0<EOCF> is set to 1 and an INTAD interrupt request is generated. ADMOD0<ADBF> is not cleared to 0 but held at 1.

To stop conversion in a repeat conversion mode (i.e. in cases ③ and ④), write a 0 to ADMOD0<REPET>. After the current conversion has been completed, the repeat conversion mode terminates and ADMOD0<ADBF> is cleared to 0.

Switching to a halt state (IDLE2 Mode with ADMOD1<I2AD> cleared to 0, IDLE1 Mode or STOP Mode) immediately stops operation of the AD converter even when AD conversion is still in progress. In repeat conversion modes (i.e. in cases ③ and ④), when the halt is released, conversion restarts from the beginning. In single conversion modes (i.e. in cases ① and ②), conversion does not restart when the halt is released (the converter remains stopped).

Table 3.11.2 shows the relationship between the AD conversion modes and interrupt requests.

Table 3.11.2 Relationship Between AD Conversion Modes and Interrupt Requests

Mode	Interrupt Request Generation	ADMOD0		
		<ITM0>	<REPEAT>	<SCAN>
Channel Fixed Single Conversion Mode	After completion of conversion	X	0	0
Channel Scan Single Conversion Mode	After completion of scan conversion	X	0	1
Channel Fixed Repeat Conversion Mode	Every conversion	0	1	0
	Every forth conversion	1		
Channel Scan Repeat Conversion Mode	After completion of every scan conversion	X	1	1

X: Don't care

(5) AD conversion time

202 states (11.22 μs @ f_{FPH} = 36 MHz) are required for the AD conversion of one channel.

(6) Storing and reading the results of AD conversion

The AD Conversion Data Upper and Lower Registers (ADREG04H/L to ADREG37H/L) store the results of AD conversion. (ADREG04H/L to ADRG37H/L are read-only registers.)

In Channel Fixed Repeat Conversion Mode, the conversion results are stored successively in registers ADREG04H/L to ADRG37H/L. In other modes the AN0 and AN4, AN1 and AN5, AN2 and AN6, AN3 and AN7 conversion results are stored in ADREG04H/L, ADREG15H/L, ADREG26H/L and ADREG37H/L respectively.

Table 3.11.3 shows the correspondence between the analog input channels and the registers which are used to hold the results of AD conversion.

Table 3.11.3 Correspondence Between Analog Input Channels and AD Conversion Result Registers

Analog input channel (Port A)	AD Conversion Result Register	
	Conversion modes other than at right	Channel fixed repeat conversion mode (every 4 th conversion)
AN0	ADREG04H/L	ADREG04H/L
AN4		
AN1	ADREG15H/L	ADREG15H/L
AN5		
AN2	ADREG26H/L	ADREG26H/L
AN6		
AN3	ADREG37H/L	ADREG37H/L
AN7		

<ADRxRF>, bit 0 of the AD conversion data lower register, is used as the AD conversion data storage flag. The storage flag indicates whether the AD conversion result register has been read or not. When a conversion result is stored in the AD conversion result register, the flag is set to 1. When either of the AD conversion result registers (ADREGxH or ADREGxL) is read, the flag is cleared to 0.

Reading the AD conversion result also clears the AD Conversion End flag ADMOD0<EOCF> to 0.

Setting example:

① Convert the analog input voltage on the AN3 pin and write the result, to memory address 0800H using the AD interrupt (INTAD) processing routine.

Main routine:

```
              7 6 5 4 3 2 1 0
 INTE0AD   ←  X 1 0 0 - - - -     Enable INTAD and set it to Interrupt Level 4.
 ADMOD1    ←  1 1 X X 0 0 1 1     Set pin AN3 to be the analog input channel.
 ADMOD0    ←  X X 0 0 0 0 0 1     Start conversion in Channel Fixed Single Conversion Mode.
```

Interrupt routine processing example:

```
 WA        ← ADREG37              Read value of ADREG37L and ADREG37H into 16-bit
                                  general-purpose register WA.
 WA        > > 6                  Shift contents read into WA six times to right and zero-fill upper
                                  bits.
 (0800H)   ← WA                   Write contents of WA to memory address 0800H.
```

② This example repeatedly converts the analog input voltages on the three pins AN0, AN1 and AN2, using Channel Scan Repeat Conversion Mode.

```
 INTE0AD   ←  X 0 0 0 - - - -     Disable INTAD.
 ADMOD1    ←  1 1 X X 0 0 1 0     Set pins AN0 to AN2 to be the analog input channels.
 ADMOD0    ←  X X 0 0 0 1 1 1     Start conversion in Channel Scan Repeat Conversion Mode.
```

Note: X = Don’t care; “–” = No change

3.12 Watchdog timer (runaway detection timer)

The TMP91C829 features a watchdog timer for detecting runaway.

The watchdog timer (WDT) is used to return the CPU to Normal state when it detects that the CPU has started to malfunction (runaway) due to causes such as noise. When the watchdog timer detects a malfunction, it generates a non-maskable interrupt INTWD to notify the CPU of the malfunction.

Connecting the watchdog timer output to the Reset pin internally forces a reset.

3.12.1 Configuration

Figure 3.12.1 is a block diagram of he watchdog timer (WDT).

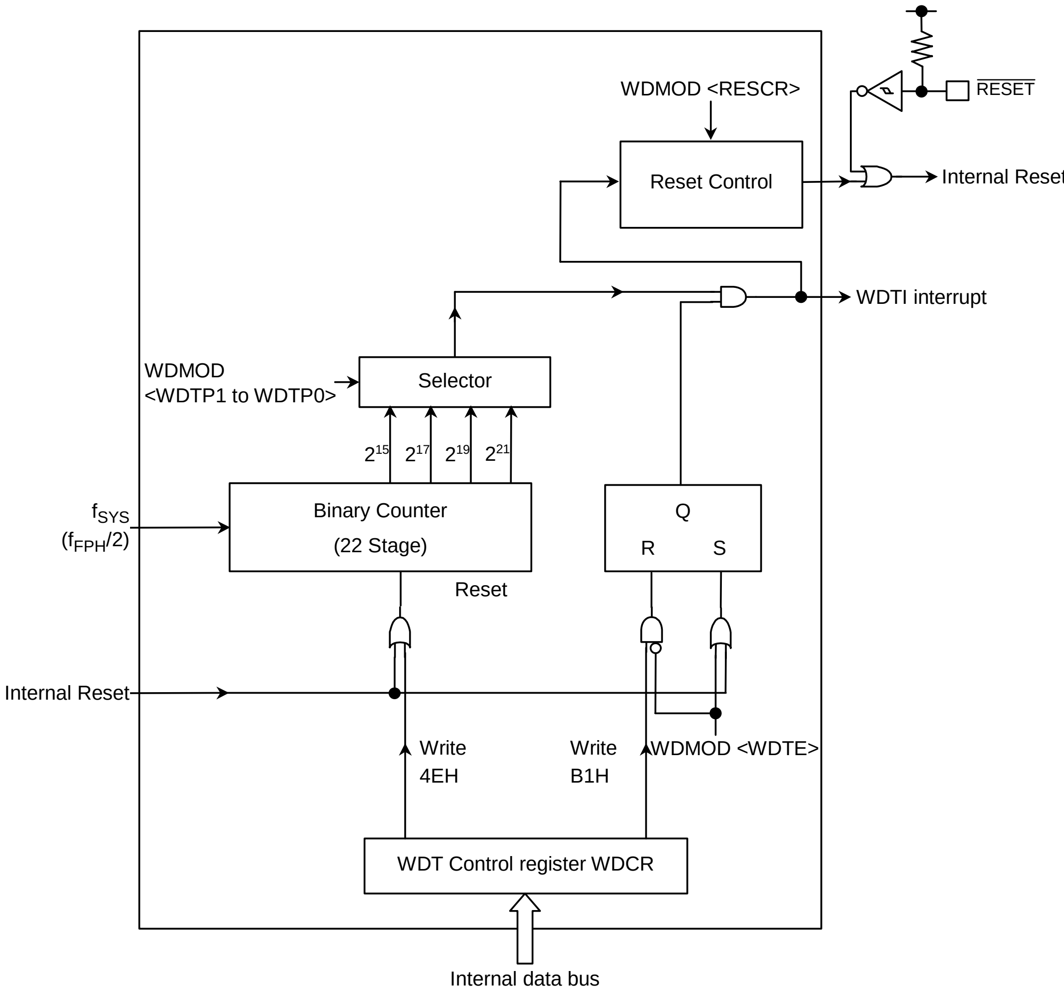


Figure 3.12.1 Block diagram of watchdog timer

Note:	The watchdog timer cannot operate by disturbance noise in some case. Take care when design the device.

The watchdog timer consists of a 22-stage binary counter which uses the system clock (f_{SYS}) as the input clock. The binary counter can output $f_{SYS}/2^{15}$, $f_{SYS}/2^{17}$, $f_{SYS}/829$ and $f_{SYS}/2^{21}$. Selecting one of the outputs using WDMOD<WDTP1,WDTP0> generates a Watchdog interrupt and outputs watchdog timer out when an overflow occurs.

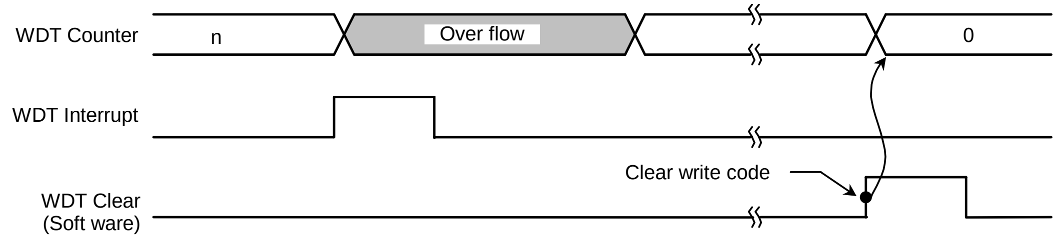


Figure 3.12.2 Normal Mode

The runaway detection result can also be connected to the Reset pin internally.

In this case, the reset time will be between 22 and 29 states as shown in Figure 3.12.3.

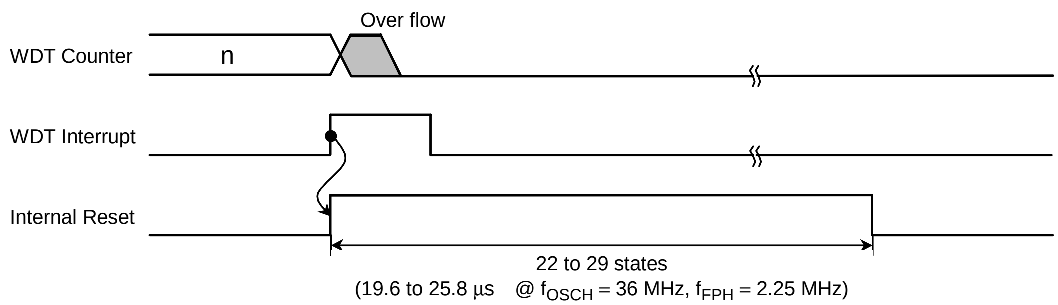


Figure 3.12.3 Reset Mode

3.12.2 Control registers

The watchdog timer WDT is controlled by two control registers WDMOD and WDCR.

(1) Watchdog Timer Mode Register (WDMOD)

① Setting the detection time for the watchdog timer in <WDTP>

This 2-bit register is used for setting the watchdog timer interrupt time used when detecting runaway. On a Reset this register is initialized to WDMOD<WDTP1,WDTP0> = 00.

The detection times for WDT are shown in Figure 3.12.4.

② Watchdog Timer Enable/Disable Control Register <WDTE>

On a Reset WDMOD<WDTE> is initialized to 1, enabling the watchdog timer. To disable the watchdog timer, it is necessary to set this bit to 0 and to write the disable code (B1H) to the Watchdog Timer Control Register WDCR. This makes it difficult for the watchdog timer to be disabled by runaway.

However, it is possible to return the watchdog timer from the disabled state to the enabled state merely by setting <WDTE> to 1.

③ Watchdog timer out reset connection <RESCR>

This register is used to connect the output of the watchdog timer with the RESET terminal internally. Since WDMOD<RESCR>is initialized to 0 on a Reset, a Reset by the watchdog timer will not be performed.

(2) Watchdog Timer Control Register (WDCR)

This register is used to disable and clear the binary counter for the watchdog timer.

- Disable control

The watchdog timer can be disabled by clearing WDMOD<WDTE> to 0 and then writing the disable code (B1H) to the WDCR register.

```
WDMOD   ← 0 - - - - - - -     Clear WDMOD<WDTE>to 0.
WDCR    ← 1 0 1 1 0 0 0 1     Write the disable code (B1H).
```

- Enable control

Set WDMOD<WDTE>to 1.

- Watchdog timer clear control

To clear the binary counter and cause counting to resume, write the clear code (4EH) to the WDCR register.

```
WDCR    ← 0 1 0 0 1 1 1 0     Write the clear code (4EH).
```

WDMOD (0300H)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0			I2WDT	RESCR	—
Read/Write	R/W	R/W				R/W		R/W
After Reset	1	0	0			0	0	0
Function	WDT control 1: enable	Select detecting time 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $829/f_{SYS}$ 11: $2^{21}/f_{SYS}$				IDLE2 0: Stop 1: Operate	1: Internally connects WDL out to the reset pin	Always write 0

Watchdog timer out control

0	—
1	Connects WDT out to a reset

IDLE2 Control

0	Stop
1	Operation f

Watchdog timer detection time

fc = 36 MHz

SYSCR1 Gear Value <GEAR2 to GEAR0>	Watchdog Timer Detection Time WDMOD<WDTP1 to WDTP0>			
	00	01	10	11
000 (fc)	1.82 ms	7.28 ms	29.1 ms	116.5 ms
001 (fc/2)	3.64 ms	14.56 ms	58.2 ms	233.0 ms
010 (fc/4)	7.28 ms	29.12 ms	116.4 ms	466.0 ms
011 (fc/8)	14.56 ms	58.24 ms	232.8 ms	932.0 ms
100 (fc/16)	29.12 ms	116.48 ms	465.6 ms	1864.0 ms

Watchdog timer Enable/Disable control

0	Disabled
1	Enabled

Figure 3.12.4 Watchdog Timer Mode Register

WDCR (0301H)

	7	6	5	4	3	2	1	0
Bit symbol	—							
Read/Write	W							
After reset	—							
Function	B1H: WDT disable code 4EH: WDT clear code							

Disable/Clear WDT

B1H	Disable code
4EH	Clear code
Others	Don't care

Figure 3.12.5 Watchdog Timer Control Register

3.12.3 Operation

The watchdog timer generates an INTWD interrupt when the detection time set in the WDMOD<WDTP1,WDTP0> has elapsed. The watchdog timer must be zero-cleared in software before an INTWD interrupt will be generated. If the CPU malfunctions (i.e. if runaway occurs) due to causes such as noise, but does not execute the instruction used to clear the binary counter, the binary counter will overflow and an INTWD interrupt will be generated. The CPU will detect malfunction (runaway) due to the INTWD interrupt and in this case it is possible to return to the CPU to normal operation by means of an anti-mulfunction program. By connecting the Watchdog Timer Out pin to a peripheral device's reset input, the occurrence of a CPU malfunction can also be relayed to other devices.

The watch dog timer works immediately after reset.

The watchdog timer does not operate in IDLE1 or STOP Mode,
as the binary counter continues counting during bus release (When $\overline{\text{BUSAK}}$ goes Low).

When the device is in IDLE2 Mode, the operation of WDT depends on the WDMOD<I2WDT> setting. Ensure that WDMOD<I2WDT> is set before the device enters IDLE2 Mode.

Example: ① Clear the binary counter.

```
WDCR   ← 0 1 0 0 1 1 1 0     Write the clear code (4EH).
```

② Set the watchdog timer detection time to 2^{17} / f_{SYS}.

```
WDMOD  ← 1 0 1 - - - - -
```

③ Disable the watchdog timer.

```
WDMOD  ← 0 - - - - - X X     Clear WDTE to 0.
WDCR   ← 1 0 1 1 0 0 0 1     Write the disable code (B1H).
```

3.13 Multi-Vector Control

3.13.1 Multi-Vector Controller

(1) Outline

By rewriting the value of multi-vector control resister (MVEC 0 and 1), a vector table is arbitrarily movable.

(2) Control resister

The amount of 228 bytes become an interruption vector area from the value set as vector control resister (MVEC 0 and 1).

Vector control resister composition

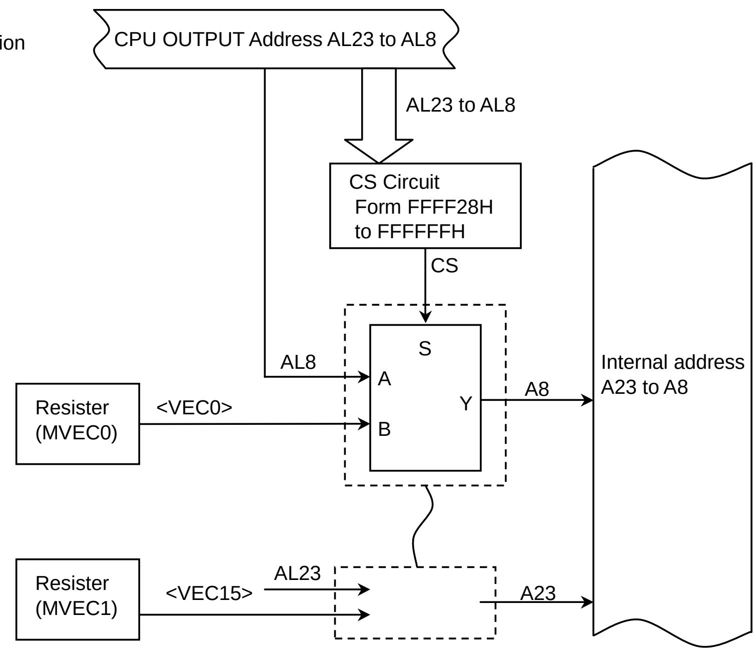
MVEC0 (00AEH)

	7	6	5	4	3	2	1	0
Bit symbol	VEC7	VEC6	VEC5	VEC4	VEC3	VEC2	VEC1	VEC0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	1	1	1	1	1	1	1	1
Function	Vector Address A15 to A8							

MVEC1 (00AFH)

	7	6	5	4	3	2	1	0
Bit symbol	VEC15	VEC14	VEC13	VEC12	VEC11	VEC10	VEC9	VEC8
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	1	1	1	1	1	1	1	1
Function	Vector Address A23 to A16							

Circuit composition



Note: Write MVEC1,0 after making an interruption prohibition state.

3.13.2 Multi-Boot Mode

(1) Outline

The TMP91C829 has multi-boot mode available as an on-board programming operation mode. When in multi-boot mode, the boot ROM is mapped into memory space. This boot ROM is a mask ROM that contains a program to rewrite the flash memory on-board.

Rewriting is accomplished by connecting the TMP91C829's SIO and the programming tool (controller) and then sending commands from the controller to the target board.

The boot program included in the boot ROM only has the function of a loader for transferring program data from an external source into the device's internal RAM.

Rewriting can be performed by UART. From 1000H to 105FH in device's internal RAM is work area of boot program. Don't transfer program data in this work area.

Figure 3.12.1 shows an example of how to connect the programming controller and the target board. (When ROM has 16-bit data bus.)

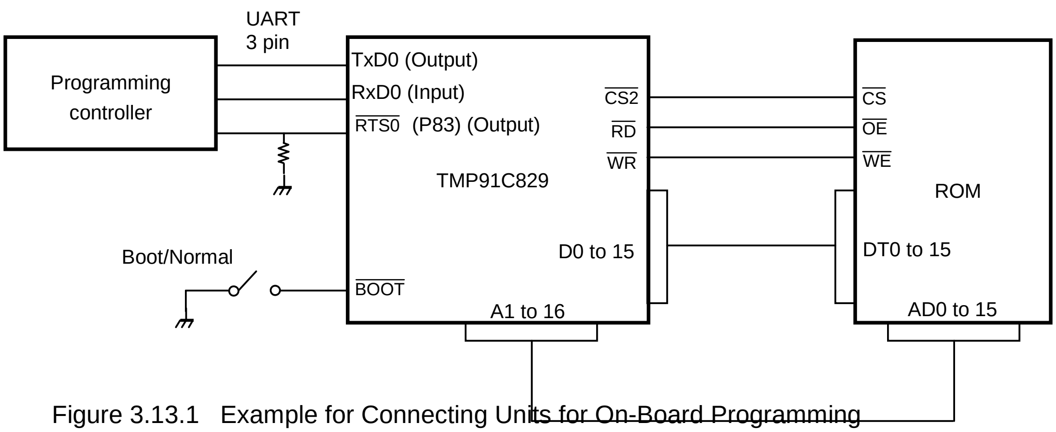
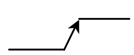


Figure 3.13.1 Example for Connecting Units for On-Board Programming

(2) Mode setting

To execute on-board programming, start the TMP91C829 in multi-boot mode. Settings necessary to start up in multi-boot mode are shown below.

$\overline{\text{BOOT}}$ = L

$\overline{\text{RESET}}$ = 

After setting the $\overline{\text{BOOT}}$ pin each to the above conditions and a $\overline{\text{RESET}}$, the TMP91C829 start up in multi-boot mode.

(3) Memory Map

Figure 3.12.2 shows memory maps for multi-chip and multi-boot modes. When start up in multi-boot mode, internal boot ROM is mapped in FFF800H address, the boot program starts up.

When start up in multi-chip mode, internal boot ROM is mapped in 1F800H address, it can be made to operate arbitrarily by the user. Program starting address is 1F800H.

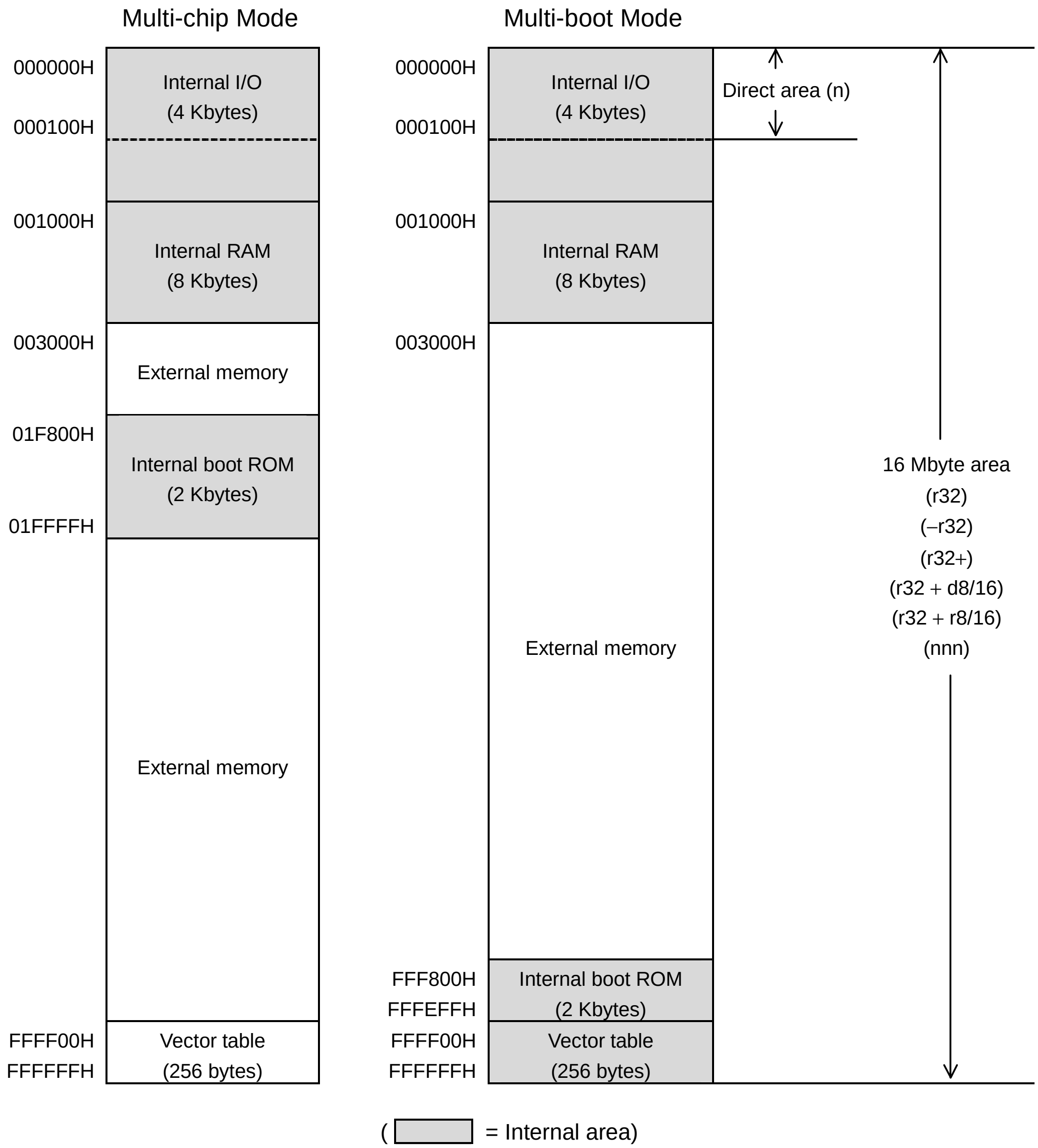


Figure 3.13.2 TMP91C829 Memory Map

(4) SIO interface specifications

The following shows the SIO communication format in multi-boot mode.

Before on-board programming can be executed, the communication format on the programming controller side must also be set up in the same way as for the TMP91C829.

Note that although the default baud rate is 9600 bps, it can be changed to other values as shown in Table 3.13.3.

Serial transfer mode	:	UART(asynchronous communication)mode, full-duplex communication
Data length	:	8-bits
Parity bit	:	None
STOP bit	:	1-bit
Handshake	:	Micro-controller (P83) → Programming controller
Baud rate(default)	:	9600 bps

(5) SIO data transfer format

Table 3.13.1 through 3.13.6 show supported frequencies, data transfer format, baud rate modification commands, operation commands, version management information, and frequency measurement result with data store location, respectively.

Also refer to the description of boot program operation in the latter pages of this manual as you read these tables.

Table 3.13.1 Supported Frequencies

16.000 MHz	20.000 MHz	22.579 MHz	25.000 MHz	32.000 MHz	33.868 MHz	36.000 MHz

Table 3.13.2 Transfer Format

	Number of Bytes Transferred	Transfer Data from Controller to TMP91C829	Baud Rate	Transfer Data from TMP91C829 to Controller
BOOT ROM	1st byte 2nd byte	Matching data (5AH) —	9600 bps 9600 bps	— (Frequency measurement and baud rate auto set) OK: Echoback data (5AH) NG: Nothing transmitted
	3rd byte : 6th byte	—	9600 bps	Version management information (See Table 3.13.5)
	7th byte	—	9600 bps	Frequency information (See Table 3.13.6)
	8th byte 9th byte	Baud rate modification command (See Table 3.13.3) —	9600 bps 9600 bps	— OK: Echoback data NG: Error code X 3
	10th byte : n'th -4 byte	User program Extended Intel Hex format(binary)	Changed new baud rate	NG: Operation stop by checksum error
	n'th -3 byte	—	Changed new baud rate	OK:SUM(High) (See (6) (iii) Notes on SUM)
	n'th -2 byte	—	Changed new baud rate	OK:SUM(Low)
	n'th -1 byte n'th byte	User program start command (C0H) (See Table 3.13.4) —	Changed new baud rate Changed new baud rate	— OK: Echoback data (C0H) NG: Error code X 3
RAM	—	JUMP to user program start address		

Error code X 3 means sending an error code three times. Example, when error code is 62H, TMP91C829 sends 62H three times. About error code, see (6)(ii) Error Code.

Table 3.13.3 Baud Rate Modification Command

Baud Rate (bps)	9600	19200	38400	57600	115200
Modification command	28H	18H	07H	06H	03H

Table 3.13.4 Operation Command

Operation command	Operation
C0H	Start user program

Table 3.13.5 Version Management Information

Version Information	ASCII code
FRM1	46H, 52H, 4DH, 31H

Table 3.13.6 Frequency Measurement Result Data

Frequency of Resonator (MHz)	16.000	20.000	22.579	25.000	32.000	33.868	36.000
1000H (RAM store address)	00H	01H	02H	03H	04H	05H	06H

(6) Description of SIO boot program operation

When you start the TMP91C829 in multi-boot mode, the boot program starts up. The boot program provides the RAM loader function described below.

RAM loader

The RAM loader transfers the data sent from the controller in Extended Intel Hex format into the internal RAM. When the transfer has terminated normally, the RAM loader calculates the SUM and sends the result to the controller before it starts executing the user program. The execution start address is the first address received. This RAM loader function provides the user’s own way to control on-board programming.

To execute on-board programming in the user program, you need to use the flash memory command sequence to be connected. (Must be matched to the flash memory addresses in multi-boot mode).

(i) Operational procedure of RAM loader

1. Connect the serial cable. Make sure to perform connection before resetting the microcontroller.
2. Set the $\overline{\text{BOOT}}$ pin to “Boot” and reset the micro-controller.
3. The receive data in the 1st byte is the matching data. When the boot program starts in multi-boot mode, it goes to a state in which it waits for the matching data to receive. Upon receiving the matching data, it automatically adjusts the serial channels’ initial baud rate to 9600 bps. The matching data is 5AH.
4. The 2nd byte is used to echo back 5AH to the controller upon completion of the automatic baud rate setting in the first byte. If the device fails in automatic baud rate setting, it goes to an idle state.
5. The 3rd byte through 6th byte are used to send the version management information of the boot program in ASCII code. The controller should check that the correct version of the boot program is used.

6. The 7th byte is used to send information of the measured frequency.
   The controller should check that the frequency of the resonator is measured correctly.
7. The receive data in the 8th byte is the baud rate modification data. The five kinds of baud rate modification data shown in Table 3.13.3 are available. Even when you do not change the baud rate, be sure to send the initial baud rate data (28H;9600 bps). Baud rate modification becomes effective after the echoback transmission is completed.
8. The 9th byte is used to echo back the received data to the controller when the data received in the 8th byte is one of the baud rate modification data corresponding to the device's operating frequency. Then the baud rate is changed. If the received baud rate data does not correspond to the device's operating frequency, the device goes to an idle state after sending 3 bytes of baud rate modification error code (62H).
9. The receive data in the 10th byte through n'th – 4 byte is received as binary data in Extended Intel Hex format. No received data is echoed back to the controller.
   The RAM loader processing routine ignores the received data until it receives the start mark (3AH for ":") in Extended Intel Hex format. Nor does it send error code to the controller. After receiving the start mark, the routine receives a range of data from the data length to checksum and writes the received data to the specified RAM addresses successively.
   After receiving one record of data from start mark to checksum, the routine goes to a start mark waiting state again.
   If a receive error or checksum error of Extended Hex format occurs, the device goes to an idle state without returning error code to the controller.
   Because the RAM loader processing routine executes a SUM calculation routine upon detecting the end record, the controller should be placed in a SUM waiting state after sending the end record to the device.
10. The n'th – 3 byte and the n'th – 2 byte are the SUM value that is sent to the controller in order of upper byte and lower byte. For details on how to calculate the SUM, refer to "Notes on SUM" in the latter page of this manual. The SUM calculation is performed only when no write error, receive error, or Extended Intel Hex format error has been encountered after detecting the end record. Soon after calculation of SUM, the device sends the SUM data to the controller. The controller should determine whether writing to the RAM has terminated normally depending on whether the SUM value is received after sending the end record to the device.
11. After sending the SUM, the device goes to a state waiting for the user program start code. If the SUM value is correct, the controller should send the user program start command to the n'th – 1 byte. The user program start command is C0H.
12. The n'th byte is used to echo back the user program start code to the controller. After sending the echoback to the controller, the stack pointer is set to 105FH and the boot program jumps to the first address that is received as data in Extended Intel Hex format.
13. If the user program start code is wrong or a receive error occurs, the device goes to an idle state after returning three bytes of error code to the controller.

(ii) Error Code

The boot program sends the processing status to the controller using various code. The error code is listed in the table below.

Table 3.13.7 Error Code

Error Code	Meaning of Error Code
62H	Baud rate modification error occurred.
64H	Operation command error occurred.
A1H	Framing error in received data occurred.
A3H	Overrun error in received data occurred.

*1: When a receive error occurs when receiving the user program, the device does not send the error code to the controller.

*2: After sending the error code, the device goes to an idle state.

(iii) Notes on SUM

1. Calculation method

SUM consists of byte+byte.....+byte , the sum of which is returned in word as the result. Namely, data is read out in byte and sum of which is calculated, with the result returned in word.

Example:

A1H
B2H
C3H
D4H

If the data to be calculated consists of the four bytes shown to the left, SUM of the data is:

A1H+B2H+C3H+D4H = 02EAH

SUM(HIGH) = 02H

SUM(LOW) = EAH

2. Calculation data

The data from which SUM is calculated is the RAM data from the first address received to the last address received.

The received RAM write data is not the only data to be calculated for SUM. Even when the received addresses are noncontiguous and there are some unwritten areas, data in the entire memory area is calculated. The user program should not contain unwritten gaps.

(iv) Notes on Extended Intel Hex Format (binary)

1. After receiving the checksum of a record, the device waits for the start mark (3AH for ":") of the next record. Therefore, the device ignores all data received between records during that time unless the data is 3AH.
2. Make sure that once the controller program has finished sending the checksum of the end record, it does not send anything and waits for two byes of data to be received (upper and lower bytes of SUM). This is because after receiving the checksum of the end record, the boot program calculates the SUM and returns the calculated SUM in two bytes to the controller.
3. It becomes the cause of incorrct operation to write to areas out of device's internal RAM. Therefore, when an extended record is transmitted, be sure to set a paragraph address to 0000H.
4. Always make sure the first record type is an extended record. Because the initial value of the address pointer is 00H.

5. Transmit a user program not by the ASCII code but by binary. However, start mark “:” is 3AH (ASCII code).

Example: Transmit data in the case of writing in 16 bytes data from address 1060H

```
Data Record    3A 10 1060 00 0607F100030000F201030000B1F16010 77
               |  |   |   ↓ Data                              Checksum
               |  |   ↓ Record Type
               |  ↓ Address
               ↓ Number of Data
               “:” (Start Mark)

End Record     3A 00 0000 01 FF
               |  |   |   ↓ Checksum
               |  |   ↓ Record Type
               |  ↓ Address
               ↓ Number of Data
               “:” (Start Mark)
```

(v) Error When Receiving User Program

If the following errors occur in Extended Intel Hex format when receiving the user program, the device goes to an idle state.

- When the record type is not 00H, 01H, 02H
- When a checksum error occurs

(vi) Error between Frequency Measurement and Baud Rate

The boot program measures the resonator frequency when receiving matching data. If an error is under 3%, the boot program decides on that frequency. Since there is an overlap between the margin of 3% for 32.000 MHz and 33.868 MHz, the boundary is set at the intermediate value between the two. The baud rate is set based on the measured frequency. Each baud rate includes a set error shown in Table 3.13.8. For example, in the case of 20.000 MHz and 9600 bps, the baud rate is actually set at 9615.38 bps with an error of 0.2%. To establish communication, the sum of the baud rate set error shown in Table 3.13.8 and the frequency error need to be under 3%.

Table 3.13.8 Set Error of Each Baud Rate (%)

	9600 bps	19200 bps	38400 bps	57600 bps	115200 bps
16.000 MHz	0.2	0.2	0.2	–0.6	–0.8
20.000 MHz	0.2	0.2	0.2	–0.2	0.9
22.579 MHz	0	0.7	0	0	0
25.000 MHz	–0.2	0.5	–0.1	0.5	0.5
32.000 MHz	0.1	0.2	0.2	0	0.6
33.868 MHz	0.2	0.2	0.2	0	0.7
36.000 MHz	0.2	0.2	–0.7	0.2	0.2

(7) Ports setup of the boot program

Only ports shown in Table 3.13.9 are set up in the boot program. At the time of boot program use, be careful of the influence on a user system. Do not use $\overline{\text{CS0}}$ space and P60 in the system which uses the boot program.

Other ports are not setting up, and are the reset state or the state of boot program starting.

Table 3.13.9 Ports setting list

Ports	Function	Input/Output	High/Low	Notes
P60	$\overline{\text{CS0}}$	Output	—	$\overline{\text{CS0}}$ space is 20000H to 201FFH
P61	Port	Output	—	
P62	Port	Output	High	
P63	Port	Output	—	
P80	Port	Input	High	Not open drain port. This port becomes TxD0 after matching data reception.
P81	RxD0	Input	High	
P82	Port	Input	—	
P83	Port	Input	Low	This port is set as the output and becomes $\overline{\text{RTS0}}$ after matching data reception.
P84	Port	Input	—	
P85	Port	Input	—	
P86	Port	Input	—	
P87	Port	Input	—	

—: Un-setting up

(8) Setting Method of Microcontroller Peripherals

Although P83 has the $\overline{\text{RTS0}}$ function, it is initially in a high impedance state and not set as $\overline{\text{RTS0}}$. To establish serial communication, attach a pull-down resister to P83.

4. Electrical Characteristics (tentative)

4.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Power Supply Voltage (5 V)	HVcc	–0.5 to 5.75	V
Power Supply Voltage (3 V)	LVcc	–0.5 to 4.0	V
Input Voltage	VIN	–0.5 to Vcc + 0.5	V
Output Current (per pin)	IOL	2	mA
Output Current (per pin)	IOH	–2	mA
Output Current (total)	ΣIOL	80	mA
Output Current (total)	ΣIOH	–80	mA
Power Dissipation (Ta = 85°C)	PD	600	mW
Soldering Temperature (10 s)	TSOLDER	260	°C
Storage Temperature	TSTG	–65 to 150	°C
Operating Temperature	TOPR	–20 to 70	°C

Note: The absolute maximum ratings are rated values which must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products which include this device, ensure that no absolute maximum rating value will ever be exceeded.

4.2 DC Characteristics (1/2)

<table>
<tr><th colspan="2">Parameter</th><th>Symbol</th><th>Condition</th><th>Min</th><th>Typ. (Note)</th><th>Max</th><th>Unit</th></tr>
<tr><td colspan="2">Power Supply Voltage (5V)
(AVcc = HVcc)
(AVss = DVss = 0 V)</td><td>HVCC</td><td>fc = 10 to 36 MHz</td><td>4.75</td><td></td><td>5.25</td><td>V</td></tr>
<tr><td colspan="2">Power Supply Voltage (3V)</td><td>LVCC</td><td>fc = 10 to 36 MHz</td><td>3.0</td><td></td><td>3.6</td><td></td></tr>
<tr><td rowspan="5">Input Low Voltage</td><td>D0 to D7, P10 to P17
(D9 to D15)</td><td>HV_{IL}</td><td></td><td rowspan="5">–0.3</td><td></td><td>0.8</td><td rowspan="10">V</td></tr>
<tr><td>The other Ports</td><td>V_{IL1}</td><td></td><td></td><td>0.3 HVcc</td></tr>
<tr><td>$\overline{RESET}$, $\overline{NMI}$
P56 (INT0), P70 (INT1)
P72 (INT2), P73 (INT3)
P75 (INT4), P90 (INT5)</td><td>V_{IL2}</td><td></td><td></td><td>0.25 HVcc</td></tr>
<tr><td>AM0, 1</td><td>V_{IL3}</td><td></td><td></td><td>0.3</td></tr>
<tr><td>X1</td><td>V_{IL4}</td><td></td><td></td><td>0.2 LVcc</td></tr>
<tr><td rowspan="5">Input Low Voltage</td><td>D0 to D7, P10 to P17 (D9 to D15)</td><td>V_{IH}</td><td></td><td>2.2</td><td></td><td rowspan="4">HVcc + 0.3</td></tr>
<tr><td>The other Ports</td><td>V_{IH1}</td><td></td><td>0.7 HVcc</td><td></td></tr>
<tr><td>$\overline{RESET}$, $\overline{NMI}$
P56 (INT0), P70 (INT1)
P72 (INT2), P73 (INT3)
P75 (INT4), P90 (INT5)</td><td>V_{IH2}</td><td></td><td>0.75 HVcc</td><td></td></tr>
<tr><td>AM0, 1</td><td>V_{IH3}</td><td></td><td>HVcc – 0.3</td><td></td></tr>
<tr><td>X1</td><td>V_{IH4}</td><td></td><td>0.8 LVcc</td><td></td><td>LVcc + 0.3</td></tr>
<tr><td colspan="2">Output Low Voltage</td><td>V_{OL}</td><td>IOL = 1.6 mA</td><td></td><td></td><td>0.45</td><td rowspan="2">V</td></tr>
<tr><td colspan="2">Output High Voltage</td><td>V_{OH}</td><td>IOH = – 400 μA</td><td>4.2</td><td></td><td></td></tr>
</table>

Note: Typical values are for when Ta = 25 °C and HVcc = 5.0 V LVcc = 3.3 V uncles otherwise noted.

DC Characteristics (2/2)

Parameter	Symbol	Min	Typ. (Note)	Max	Condition	Unit
Input Leakage Current	ILI		0.02	±5	$0.0 \le V_{IN} \le$ HVcc	μA
Output Leakage Current	ILO		0.05	±10	$0.2 \le V_{IN} \le$ HVcc – 0.2	μA
Power Down Voltage (@STOP, RAM back-up)	VSTOP	2.0		3.6	V_{IL2} = 0.2 HVcc, V_{IH2} = 0.8 HVcc	V
$\overline{\text{RESET}}$ Pull-up Resistor	RRST	40		200	HVcc = 5 V ± 5%	kΩ
Pin Capacitance	CIO			10	Fc = 1 MHz	pF
Schmitt Width $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, INT0	VTH	0.4	1.0			V
Programmable Pull-up Resistor	RKH	40		200	HVcc = 5 V ± 5%	kΩ
NORMAL (Note 2)	Icc			40	HVcc = 5 V ± 5% LVcc = 3.0 to 3.6V fc = 36 MHz	mA
IDLE2				20		
IDLE1				14		
STOP				100	HVcc = 5 V ± 5% LVcc = 3.0 to 3.6V Ta ≤ 70°C	μA

Note 1: Typical values are for when Ta = 25 °C and HVcc = 5.0 V LVcc = 3.3 V unless otherwise noted.

Note 2: Icc measurement conditions (NORMAL):
All functions are operational; output pins are open and input pins are fixed.

4.3 AC Characteristics

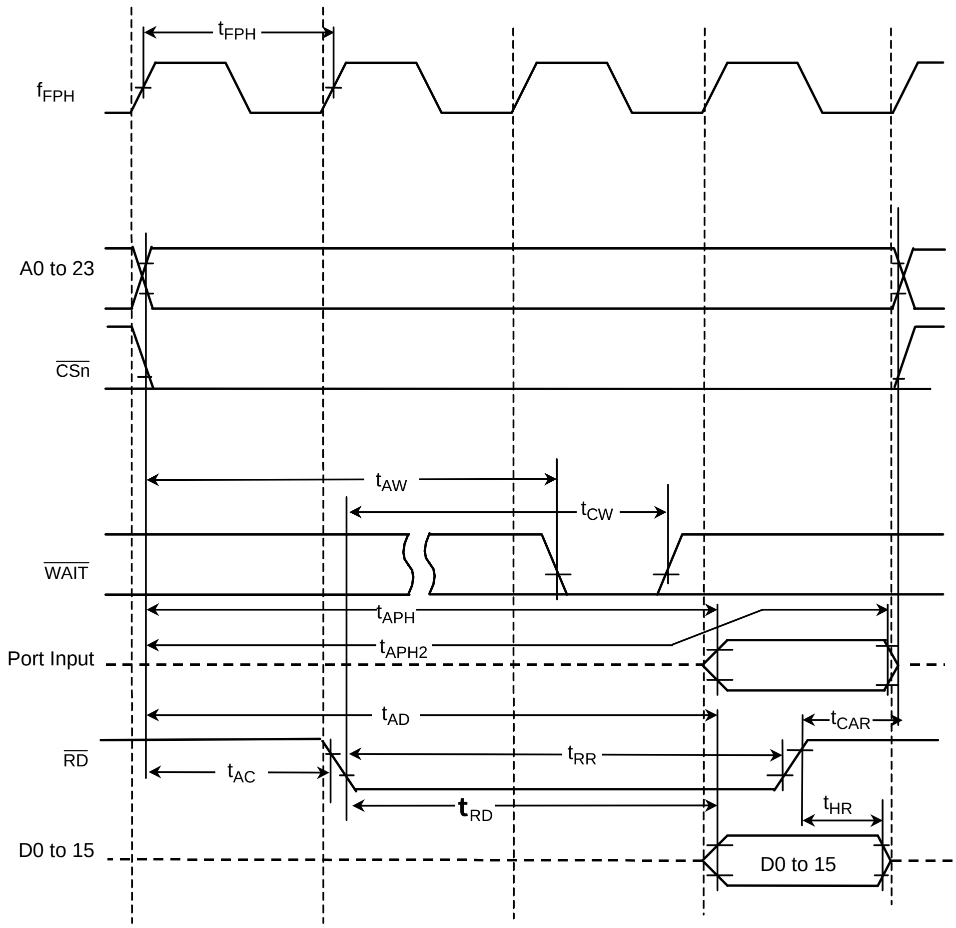
(1) HVcc = 5.0 V ± 5%, LVcc = 3.0 to 3.6 V

No.	Parameter	Symbol	Variable		f_{FPH} = 36 MHz		Unit
			Min	Max	Min	Max	
1	f_{FPH} Period (= x)	t_{FPH}	27.6	100	27.6		ns
2	A0 to 23 Vaild → $\overline{RD}$ / $\overline{WR}$ Fall	t_{AC}	x – 26		1.6		ns
3	$\overline{RD}$ Rise → A0 to A23 Hold	t_{CAR}	0.5x –13.8		0.0		ns
4	$\overline{WR}$ Rise → A0 to A23 Hold	t_{CAW}	x – 13		14.6		ns
5	A0 to A23 Valid → D0 to D15 Input	t_{AD}		3.5x – 40		56.6	ns
6	$\overline{RD}$ Fall → D0 to D15 Input	t_{RD}		2.5x – 34		35.0	ns
7	$\overline{RD}$ Low Width	t_{RR}	2.5x – 25		44.0		ns
8	$\overline{RD}$ Rise → D0 to A15 Hold	t_{HR}	0		0		ns
9	$\overline{WR}$ Low Width	t_{WW}	2.0x – 25		30.2		ns
10	D0 to D15 Valid → $\overline{WR}$ Rise	t_{DW}	1.5x – 35		6.4		ns
11	$\overline{WR}$ Rise → D0 to D15 Hold (1WAIT+n)	t_{WD}	x – 25		2.6		ns
12	A0 to A23 Valid → $\overline{WAIT}$ Input (1WAIT+n)	t_{AW}		3.5x – 60		36.6	ns
13	$\overline{RD}$ / $\overline{WR}$ Fall → $\overline{WAIT}$ Hold	t_{CW}	2.5x + 0		69.0		ns
14	A0 to A23 Valid → PORT Input	t_{APH}		3.5x – 76		20.0	ns
15	A0 to A23 Valid → PORT Hold	t_{APH2}	3.5x		96.6		ns
16	A0 to A23 Valid → PORT Valid	t_{APO}		3.5x + 60		156.6	ns

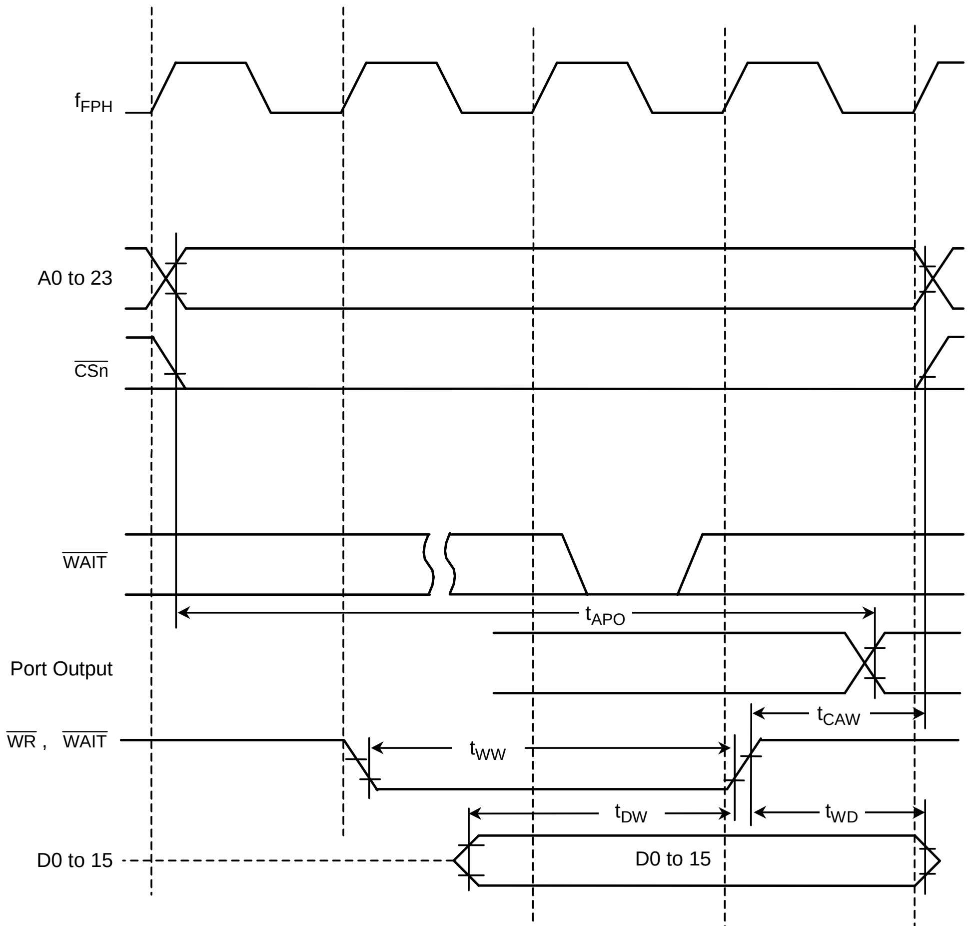
AC Measuring Conditions
- Output Level : High = 2.2 V, Low = 0.8 Vcc, CL = 50 pF
- Input Level : High = 2.4 V, Low = 0.45 V (D0 to D15)
  : High 0.8 Vcc / Low 0.2 Vcc (except D0 to D15)

Note: Symbol "x" in the above table means the period of clock "f_{FPH}", it's half period of the system clock "f_{SYS}" for CPU core. The period of f_{FPH} depends on the clock gear setting.

(2) Read Cycle


tFPH
fFPH
A0 to 23
CSn
tAW
tCW
WAIT
tAPH
tAPH2
Port Input
tAD
tCAR
RD
tRR
tAC
tRD
tHR
D0 to 15
D0 to 15

(3) Write Cycle



4.4 AD Conversion Characteristics

AVcc = HVcc, AVss = Vss

parameter	Symbol	Min	Typ.	Max	Unit
Analog Reference Voltage (+)	VREFH	HV_{CC} – 0.2 V	HV_{CC}	HV_{CC}	V
Analog Reference Voltage (–)	VREFL	DV_{SS}	DV_{SS}	DVss + 0.2 V	V
Analog Input Voltage Range	VAIN	V_{REFL}		V_{REFH}	V
Analog Current for Analog Reference Voltage <VREFON> = 1	IREF (VREFL = 0V)		0.85	1.20	mA
<VREFON> = 0			0.02	5.0	μA
Error (not including quantizing errors)	—		± 1.0	± 4.0	LSB

Note 1: 1 LSB = (VREFH – VREFL)/1024 [V]

Note 2: The value for Icc includes the current which flows through the AVcc pin.

4.5 Serial Channel Timing (I/O Internal Mode)

Note: Symbol “x” in the above table means the period of clock “f_{FPH}”, it’s half period of the system clock “f_{SYS}” for CPU core. The period of f_{FPH} depends on the clock gear setting .

(1) SCLK Input Mode

Parameter	Symbol	Variable		36 MHz (Note)		Unit
		Min	Max	Min	Max	
SCLK Period	t_{SCY}	16X		0.44		μs
Output Data → SCLK Rising/Falling Edge*	t_{OSS}	t_{SCY}/2-4X-85		25		ns
SCLK Rising/Falling Edge* → Output Data Hold	t_{OHS}	t_{SCY}/2 + 2X + 0		276		ns
SCLK Rising/Falling Edge* → Input Data Hold	t_{HSR}	3X + 10		92		ns
SCLK Rising/Falling Edge* → Valid Data Input	t_{SRD}		t_{SCY} – 0		440	ns
Valid Data Input → SCLK Rising/Falling Edge*	t_{RDS}	0		0		ns

*) SCLK Rinsing/Falling Edge: The rising edge is used in SCLK Rising Mode.
The falling edge is used in SCLK Falling Mode.

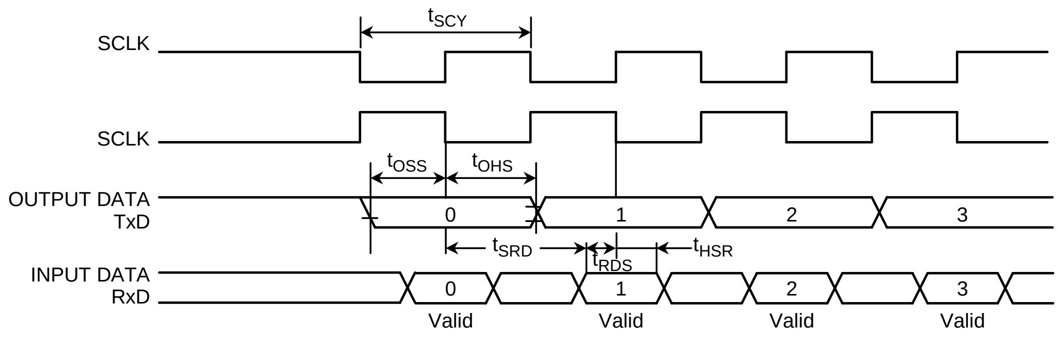
Note: at t_{SCY} = 16X

(2) SCLK Output Mode

Parameter	Symbol	Variable		36 MHz (Note)		Unit
		Min	Max	Min	Max	
SCLK Period (programable)	t_{SCY}	16X	8192X	0.44		μs
Output Data →SCLK Rising/Falling Edge*	t_{OSS}	t_{SCY}/2 – 40		180		ns
SCLK Rising/Falling Edge* → Output Data Hold	t_{OHS}	t_{SCY}/2 – 40		180		ns
SCLK Rising/Falling Edge* → Input Data Hold	t_{HSR}	0		0		ns
SCLK Rising/Falling Edge* → Valid Data Input	t_{SRD}		t_{SCY}/2 – 1X – 90		324	ns
Valid Data Input → SCLK Rising/Falling Edge*	t_{RDS}	1X + 90		117		ns

*) SCLK Rinsing/Falling Edge: The rising edge is used in SCLK Rising Mode.
The falling edge is used in SCLK Falling Mode.

Note: at t_{SCY} = 16X



4.6 Event Counter (TA0IN, TA4IN, TB0IN0, TB0IN1, TB1IN0, TB1IN1)

Parameter	Symbol	Variable		36 MHz		Unit
		Min	Max	Min	Max	
Clock Perild	t_{VCK}	8X + 100		320		ns
Clock Low Level Width	t_{VCKL}	4X + 40		150		ns
Clock High Level Width	t_{VCKH}	4X + 40		150		ns

Note: Symbol "x" in the above table means the period of clock "f_{FPH}", it's half period of the system clock "f_{SYS}" for CPU core. The period of f_{FPH} depends on the clock gear setting .

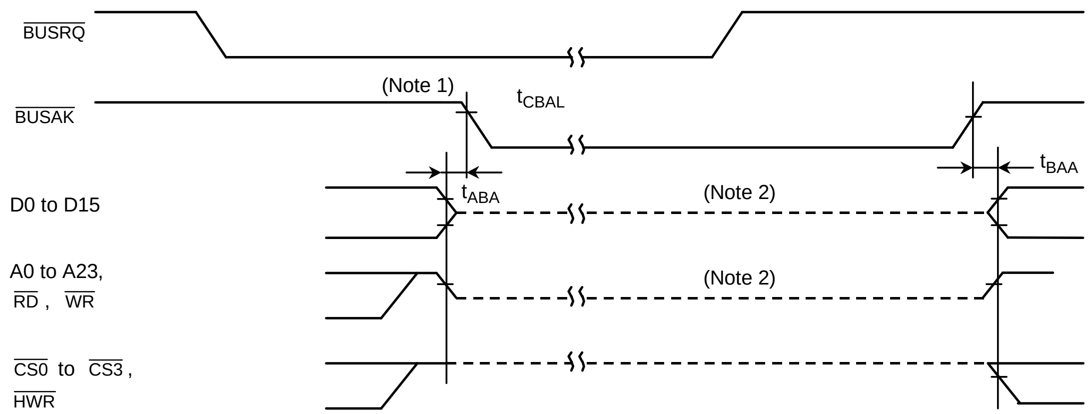
4.7 Interrupts

Note: Symbol "x" in the above table means the period of clock "f_{FPH}", it's half period of the system clock "f_{SYS}" for CPU core. The period of f_{FPH} depends on the clock gear setting .

(1) $\overline{\text{NMI}}$, INT0 to INT5 Interrupts

Parameter	Symbol	Variable		36 MHz		Unit
		Min	Max	Min	Max	
$\overline{\text{NMI}}$, INT0 to INT5 Low level width	t_{INTAL}	4X + 40		150		ns
$\overline{\text{NMI}}$, INT0 to INT5 High level width	t_{INTAH}	4X + 40		150		ns

4.8 Bus Request/Bus Acknowledge



Parameter	Symbol	Variable		f_{FPH} = 36 MHz		Unit
		Min	Max	Min	Max	
Output Buffer to $\overline{BUSAK}$ Low	t_{ABA}	0	80	0	80	ns
$\overline{BUSAK}$ High to output Buffer On	t_{BAA}	0	80	0	80	ns

Note 1: Even if the $\overline{BUSRQ}$ Signal foes Low, the bus will not be released while the $\overline{WAIT}$ signal is Low. The bus will only be released when $\overline{BUSRQ}$ goes Low while $\overline{WAIT}$ is High.

Note 2: This line shows only that the output buffer is in the Off state.
It does not indicate that the signal level is fixed.
Just after the bus is released, the signal level set before the bus was released is maintained dynamically by the external capacitance. Therefore, to fix the signal level using an external resister during bus release, careful design is necessary, since fixing of the level is delayed.
The internal programmable pull-up/pull-down resistor is switched between the Active and Non-Active states by the internal signal.

5. Table of SFRs

(SFR; special function register)

The SFRs include the I/O ports and peripheral control registers allocated to the 4-Kbyte address space from 000000H to 000FFFH.

(1) I/O Port

(2) I/O Port Control

(3) Interrupt Control

(4) Chip Select / Wait Control

(5) Clock Gear

(6) 8-bit Timer

(7) 16-bit Timer

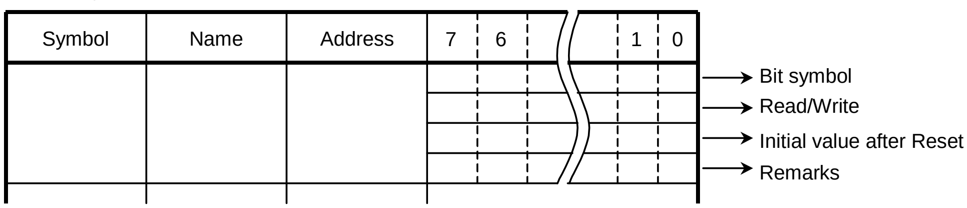
(8) UART/Serial Channel

(9) AD Converter

(10) Watchdog Timer

(11) Multi Vector Controllor

Table layout



Note: "Prohibit RMW" in the a table means that you cannot use RMW instructions on these register.

Example: When setting bit0 only of the registerP0CR, the instruction "SET 0, (0002G)" cannot be used. The LD (transfer) instruction must be used to write all eight bits.

Read/Write

R/W; Both read and write are possible.

R; Only read is possible.

W; Only write is possible.

W*; Both read and write are possible (when this bit is read as1)

Prohibit RMW; Read-Modify-Write instructions are prohibited. (The EX, ADD, ADC, BUS, SBC, INC, DEC, AND, OR, XOR, STCF, RES, SET, CHG, TEST, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD and RRD instruction are read-modify-write instructions.)

Prohibit RMW*; Read-modify-write is prohibited when controlling the pull-up resistor.

Table 5.1 Address map SFRs

[1]PORT

Address	Name
0000H	
1H	P1
2H	
3H	
4H	P1CR
5H	
6H	P2
7H	
8H	
9H	P2FC
AH	
BH	
CH	
DH	P5
EH	
FH	

Address	Name
0010H	P5CR
1H	P5FC
2H	P6
3H	P7
4H	P6CR
5H	P6FC
6H	P7CR
7H	P7FC
8H	P8
9H	P9
AH	P8CR
BH	P8FC
CH	P9CR
DH	P9FC
EH	PA
FH	

Address	Name
0020H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	ODE

Address	Name
0070H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	PZ
EH	PZCR
FH	PZFC

[2] INTC

Address	Name
0080H	DMA0V
1H	DMA1V
2H	DMA2V
3H	DMA3V
4H	
5H	
6H	
7H	
8H	INTCLR
9H	DMAR
AH	DMAB
BH	
CH	IIMC0
DH	IIMC1
EH	
FH	

Address	Name
0090H	INTE0AD
1H	INTE12
2H	INTE34
3H	INTE5
4H	
5H	INTETA01
6H	INTETA23
7H	INTETA45
8H	
9H	INTETB01
AH	
BH	INTETB0V
CH	INTES0
DH	INTES1
EH	
FH	

Address	Name
00A0H	INTETC01
1H	INTETC23
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	MVEC0
FH	MVEC1

Note: Do not access to the unnamed addresses, i.e. addresses to which no register has been allocated.

[3] CS/WAIT

Address	Name
00C0H	B0CS
1H	B1CS
2H	B2CS
3H	B3CS
4H	
5H	
6H	
7H	BEXCS
8H	MSAR0
9H	MAMR0
AH	MSAR1
BH	MAMR1
CH	MSAR2
DH	MAMR2
EH	MSAR3
FH	MAMR3

[4] CGEAR, DFM

Address	Name
00E0H	SYSCR0
1H	SYSCR1
2H	SYSCR2
3H	EMCCR0
4H	EMCCR1
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[5] TMRA

Address	Name
0100H	TA01RUN
1H	
2H	TA0REG
3H	TA1REG
4H	TA01MOD
5H	TA1FFCR
6H	
7H	
8H	TA23RUN
9H	
AH	TA2REG
BH	TA3REG
CH	TA23MOD
DH	TA3FFCR
EH	
FH	

Address	Name
0110H	TA45RUN
1H	
2H	TA4REG
3H	TA5REG
4H	TA45MOD
5H	TA5FFCR
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses, i.e. addresses to which no register has been allocated.

[6] TMRB

Address	Name
0180H	TB0RUN
1H	
2H	TB0MOD
3H	TB0FFCR
4H	
5H	
6H	
7H	
8H	TB0RG0L
9H	TB0RG0H
AH	TB0RG1L
BH	TB0RG1H
CH	TB0CP0L
DH	TB0CP0H
EH	TB0CP1L
FH	TB0CP1H

[7] UART/SIO

Address	Name
0200H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	
7H	SC1BUF
8H	SC1CR
9H	SC1MOD0
AH	BR1CR
BH	BR1ADD
CH	SC1MOD1
DH	
EH	
FH	

[8] 10-bit ADC

Address	Name
02A0H	ADREG04L
1H	ADREG04H
2H	ADREG15L
3H	ADREG15H
4H	ADREG26L
5H	ADREG26H
6H	ADREG37L
7H	ADREG37H
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
02B0H	ADMOD0
1H	ADMOD1
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses i.e. addresses to which no register has been allocated.

[9] WDT

Address	Name
0300H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses, i.e. addresses to which no register has been allocated.

(1) I/O port

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1	PORT1	01H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			0	0	0	0	0	0	0	0
			Input mode							
P2	PORT2	06H	P27	P26	P25	P24	P23	P22	P21	P20
			R/W							
			1	1	1	1	1	1	1	1
			Input mode							
P5	PORT5	0DH		P56	P55	P54	P53			
			R/W							
			1							
			Input mode (With Pull-up resistor)							
P6	PORT6	12H					P63	P62	P61	P60
							R/W			
							1	0	1	1
P7	PORT7	13H			P75	P74	P73	P72	P71	P70
					R/W					
					1	1	1	1	1	1
					Input mode					
P8	PORT8	18H	P87	P86	P85	P84	P83	P82	P81	P80
			R/W							
			1	1	1	1	1	1	1	1
			Input mode							
P9	PORT9	19H		P96	P95	P94	P93			P90
			R							
				1	1	1	1			1
			Input mode (With Pull-up resistor)							
PA	PORTA	1EH	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
			R							
			Input mode							
PZ	PORTZ	7DH					PZ3	PZ2		
							R/W			
							1	1		
							Input mode			

(2) I/O port control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P1CR	PORT1 Control	04H (Prohibit RMW)	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT							
P2FC	PORT2 Function	09H (Prohibit RMW)	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
			W							
			1	1	1	1	1	1	1	1
			0: Port, 1: Address bus (A23-A16)							
P5CR	PORT5 Control	10H (Prohibit RMW)		P56C	P55C	P54C	P53C			
			W							
				0	0	0	0			
			0: IN 1: OUT							
P5FC	PORT5 Function	11H (Prohibit RMW)		P56F		P54F	P53F			
			W							
				0		0	0			
				0: PORT 1: INT0		0: PORT 1: $\overline{BUSAK}$	0: PORT 1: $\overline{BUSRQ}$			
P6FC	PORT6 Function	15H (Prohibit RMW)					P63F	P62F	P61F	P60F
							W			
							0	0	0	0
							0: PORT 1: $\overline{CS3}$	0: PORT 1: $\overline{CS2}$	0: PORT 1: $\overline{CS1}$	0: PORT 1: $\overline{CS0}$
P7CR	PORT7 Control	16H (Prohibit RMW)			P75C	P74C	P73C	P72C	P71C	P70C
					W					
					0	0	0	0	0	0
					0 : IN 1 : OUT					
P7FC	PORT7 Function	17H (Prohibit RMW)		P72F2	P75F	P74F	P73F	P72F1	P71F	P70F
				W	W	W	W	W	W	W
				0	0	0	0	0	0	0
				0: PORT 1: INT2	0: PORT 1: INT4	0: PORT 1: TA5OUT	0: PORT 1: INT3	0: PORT 1: TA3OUT	0: PORT 1: TA1OUT	0: PORT 1: INT1
P8CR	PORT8 Control	1AH (Prohibit RMW)	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT							
P8FC	PORT8 Function	1BH (Prohibit RMW)	P87F	P86F		P84F	P83F	P82F		P80F
			W	W		W	W	W		W
			0	0		0	0	0		0
			0: PORT 1: STS1	0: PORT 1: SCLK1		0: PORT 1: TXD1	0: PORT 1: STS0	0: PORT 1: SCLK0		0: PORT 1: TXD0

I/O Port control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P9CR	PORT9 Control	1CH (Prohibit RMW)		P96C	P95C	P94C	P93C			P90C
			W							
				0	0	0	0			0
			0: IN 1: OUT							
P9FC	PORT9 Function	1DH (Prohibit RMW)		P96F	P95F					P90F
				W	W					W
				0	0					0
				0: PORT 1: TB0OUT1	0: PORT 1: TB0OUT0					0: PORT 1: TNT5
PZCR	PORT5 Control	7EH (Prohibit RMW)					PZ3C	PZ2C		
							W			
							0	0		
							0: IN 1: OUT			
PZFC	PORT5 Function	7FH (Prohibit RMW)						PZ2F		
								W		
								0		
								0: PORT 1: $\overline{\text{HWR}}$		
ODE	Sirial Open Drain	2FH (Prohibit RMW)				ODE81				ODE80
						W				W
										0
						1: P81ODE				1: P80ODE

(3) Interrupt control (1/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0AD	Interrupt Enable 0 & AD	90H	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTAD	Interrpt request level			1: INT0	Interrpt request level		
INTE12	Interrupt Enable 2/1	91H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT2	Interrupt request level			1: INT1	Interrpt request level		
INTE34	Interrupt Enable 4/3	92H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT4	Interrupt request level			1: INT3	Interrpt request level		
INTE5	Interrupt Enable 5	93H					INT5			
							I5C	I5M2	I5M1	I5M0
							R	R/W		
							0	0	0	0
							1: INT5	Interrpt request level		
INTETA01	Interrupt Enable Timer A 1/0	95H	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA1	Interrpt request level			1: INTTA0	Interrpt request level		
INTETA23	Interrupt Enable Timer A 3/2	96H	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA3	Interrpt request level			1: INTTA2	Interrpt request level		
INTETA45	Interrupt Enable Timer A 3/2	97H	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA5	Interrpt request level			1: INTTA4	Interrpt request level		
INTETB0	Interrupt Enable Timer B0	99H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTB01	Interrpt request level			1: INTTB00	Interrpt request level		
INTETB0V	Interrupt Enable Timer B0 (over flow)	9BH					INTTBOF0 (TMRB0 over flow)			
							ITF0C	ITF0M2	ITF0M1	ITF0M0
							R	R/W		
							0	0	0	0
							1: INTTBOF0	Interrpt request level		

Interrupt control (2/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTES0	Interrupt Enable Serial 0	9CH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTX0	Interrpt request level			1: INTRX0	Interrpt request level		
INTETC-01	Interrupt Enable TC0/1	A0H	INTTC1				INTTC0			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC-23	Interrupt Enable TC2/3	A1H	INTTC3				ITC2M0			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

Interrupt control (3/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA 0 Request Vector	80H			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
					DMA0 start vector					
DMA1V	DMA 1 Request Vector	81H			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
					DMA1 start vector					
DMA2V	DMA 2 Request Vector	82H			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
					DMA2 start vector					
DMA3V	DMA 3 Request Vector	83H			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0
					DMA3 start vector					
INTCLR	Interrupt Clear Control	88H (Prohibit RMW)			CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
					W					
					—	—	—	—	—	—
					Clear interrupt request DMAflag by writing to DMA start vector					
DMAR	DMA Software Request Register	89H					DMAR3	DMAR2	DMAR1	DMAR0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: DMA request in software			
DMAB	DMA Burst Request Register	8AH					DMAB3	DMAB2	DMAB1	DMAB0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1 : DMA request on burst mode			
IIMC0	Interrupt Input Mode Control 0	8CH (Prohibit RMW)		I2EDGE	I2LE	I1EDGE	I1LE	I0EDGE	I0LE	NMIREE
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			Always Write "0"	INT2 edge 0: Rising 1: Falling	INT2 0: edge 1: level	INT1 edge 0: Rising 1: Falling	INT1 0: edge 1: level	INT0 edge 0: Rising 1: Falling	INT0 0: edge 1: level	1: $\overline{\text{NMI}}$ operation even on NMI rising Edge
IIMC1	Interrupt Input Mode Control 1	8DH (Prohibit RMW)		I5EDGE	I5LE	I4EDGE	I4LE	I3EDGE	I3LE	
			W	W	W	W	W	W	W	
			0	0	0	0	0	0	0	
			Always Write "0"	INT5 edge 0: Rising 1: Falling	INT5 0: edge 1: level	INT4 edge 0: Rising 1: Falling	INT4 0: edge 1: level	INT3 edge 0: Rising 1: Falling	INT3 0: edge 1: level	

(4) Chip select / Wait control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B0CS	Block 0 CS/WAIT control Register	C0H (Prohibit RMW)	B0E		B00M1	B00M0	B0BUS	B0W2	B0W1	B0W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: DIS 1: EN		00: ROM/SRAM 01: 10: Reserved 11:		Data bus Width 0: 16 bit 1: 8 bit	000: 2WAIT 001: 1WAIT 010: 1 + NWAIT 011: 0WAIT	1xx: Reserved	
B1CS	Block 1 CS/WAIT control Register	C1H (prohibit RMW)	B1E		B10M1	B10M0	B1BUS	B1W2	B1W1	B1W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: DIS 1: EN		00: ROM/SRAM 01: 10: Reserved 11:		Data bus Width 0: 16 bit 1: 8 bit	000: 2WAIT 001: 1WAIT 010: 1 + NWAIT 011: 0WAIT	1xx: Reserved	
B2CS	Block 2 CS/WAIT control Register	C2H (prohibit RMW)	B2E	B2M	B20M1	B20M0	B2BUS	B2W2	B2W1	B2W0
			W	W	W	W	W	W	W	W
			1	0	0	0	0	0	0	0
			0: DIS 1: EN	0: 16 M space 1: eria setting	00: ROM/SRAM 01: 10: Reserved 11:		Data bus Width 0: 16 bit 1: 8 bit	000: 2WAIT 001: 1WAIT 010: 1 + NWAIT 011: 0WAIT	1xx: Reserved	
B3CS	Block 3 CS/WAIT control Register	C3H (Prohibit RMW)	B3E		B30M1	B30M0	B3BUS	B3W2	B3W1	B3W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: DIS 1: EN		00: ROM/SRAM 01: 10: Reserved 11:		Data bus Width 0: 16 bit 1: 8 bit	000: 2WAIT 001: 1WAIT 010: 1 + NWAIT 011: 0WAIT	1xx: Reserved	
BEXCS	External CS/WAIT control Register	C7H (Prohibit RMW)					BEXBUS	BEXW2	BEXW1	BEXW0
							W	W	W	W
							0	0	0	0
							Data bus Width 0: 16 bit 1: 8 bit	000: 2WAIT 001: 1WAIT 010: 1 + NWAIT 011: 0WAIT	1xx: Reserved	
MSAR0	Memory Start Address Reg0	C8H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16							
MAMR0	Memory Address Mask Reg0	C9H	V20	V19	V18	V17	V16	V15	V14~9	V8
			R/W							
			1	1	1	1	1	1	1	1
			CS0 Area size 0: enable to address comparision							
MSAR1	Memory Start Address Reg1	CAH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Stat address A23 to A16							
MAMR1	Memory Address Mask Reg1	CBH	V21	V20	V19	V18	V17	V16	V15~9	V8
			R/W							
			1	1	1	1	1	1	1	
			CS1area size 0: enable to address comparsion							

Chip select /Wait control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MSAR2	Memory Start Address Reg2	CCH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16							
MAMR2	Memory Address Mask Reg2	CDH	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			CS2area size 0:enable address comparsion							
MSAR3	Memory Start Address Reg3	CEH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16							
MAMR3	Memory Address Mask Reg3	CFH	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			CS3 area size 0: enable to address comparsion							

(5) Clock Gear

Symbol	Name	Address	7	6	5	4	3	2	1	0
SYSCR0	System Clock Control Register 0	E0H	—	—	—	—	—	WUEF	PRCK1	PRCK0
			R/W							
			1	0	1	0	0	0	0	0
			Always Write 1	Always Write 0	Always Write 1	Always Write 0	Always Write 0	Warm-up timer 0 write: Don't care 1 write: start timer 0 read: end warm-up 1 read: not end warm-up	Prscaler clock seleciton 00: f_{FPH} 01: reserved 10: fc/16 11: reserved	
SYSCR1	System Clock Control Register 1	E1H					—	GEAR2	GEAR1	GEAR0
							R/W			
							0	1	0	0
							Always Write 0	High-frequency gear value selection (fc) 000: fc 001: fc/2 010: fc/4 011: fc/8 100: fc/16 101: (reserved) 110: (reserved) 111: (reserved)		
SYSCR2	System Clock Control Register 2	E2H		—	WUPTM1	WUPTM0	HALTM1	HALTM0		DRVE
				R/W	R/W	R/W	R/W	R/W		R/W
				0	1	0	1	1		0
				Always Write 0	Warming-up time 00: reserved 01: 2^8/inputt frequency 10: 2^{14} 11: 2^{16}		00: reserved 01: STOP Mode 10: IDLE1 Mode 11: IDLE2 Mode			1: Drive the pin in STOP/ IDLE Mode
EMCCR0	EMC Control Register 0	E3H	PROTECT	—	—	—	—	EXTIN	—	—
			R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			0	0	1	0	0	0	1	1
			Protection flag 0: OFF 1: ON	Always write 0	Always write 1	Always write 0	Always write 0.	01: fc is external clock.	Always write 1	Always write 1
EMCCR1	EMC Control Register 1	E4H	Protection is turned OFF by writing 1FH. Protection is turned ON by writing any value other than 1FH.							

Note: EMCCR1

If protection is on, write operations to the following SFRs are not possible.

1. CS/WAIT control
   B0CS, B1CS, B2CS, B3CS, BEXCS,
   MSAR0, MSAR1, MSAR2, MSAR3,
   MAMR0, MAMR1, MAMR2, and MAMR3
2. Clock Gear (only EMCCR1 can be written to)
   SYSCR0, SYSCR1, SYSCR2 and EMCCR0

(6) 8-Bit Timer (1/2)

(6–1) TMRA01

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA01–RUN	Timer RUN	100H	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
			R/W				R/W	R/W	R/W	R/W
			0				0	0	0	0
			Double Buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	8-Bit Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		
TA0REG	8-Bit Timer Register 0	102H (Prohibit RMW)	–							
			W							
			Undefined							
TA1REG	8-Bit Timer Register 1	103H (Prohibit RMW)	–							
			W							
			Undefined							
TA01-MOD	8-Bit Timer Source CLK & MODE	104H	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00: 8-Bit Timer 01: 16-Bit Timer 10: 8-Bit PPG 11: 8-Bit PWM		00: Reserved 01: $2^6 – 1$ PWM cycle 10: $2^7 – 1$ 11: $2^8 – 1$		00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TA0IN pin 01: ϕT1 10: ϕT4 11: ϕT16	
TA1FFCR	8-Bit Timer Flip-Flop Control	105H					TAFF1C1	TAFF1C0	TAFF1IE	TAFF1IS
							W*		R/W	
							1	1	0	0
							00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don’t care		1: TA1FF Invert Enable	0: TMRA0 1: TMRA1 inversion

(6–2) TMRA23

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA23-RUN	Timer RUN	108H	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
			R/W				R/W	R/W	R/W	R/W
			0				0	0	0	0
			Double Buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	8-Bit Timer Run/Stop control 0: Stop & Clear 1: Run (count up)		
TA2REG	8-Bit Timer Register 0	10AH (Prohibit RMW)	–							
			W							
			Undefined							
TA3REG	8-Bit Timer Register 1	10BH (Prohibit RMW)	–							
			W							
			Undefined							
TA23-MOD	8-Bit Timer Source CLK & MODE	10CH	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00: 8-Bit Timer 01: 16-Bit Timer 10: 8-Bit PPG 11: 8-Bit PWM		00: Reserved 01: $2^6 –1$ PWM cycle 10: $2^7 – 1$ 11: $2^8 – 1$		00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: Reserved 01: ϕT1 10: ϕT4 11: ϕT16	
TA3FFCR	8-Bit Timer Flip-Flop Control	10DH					TAFF3C1	TAFF3C0	TAFF3IE	TAFF3IS
							W*		R/W	
							1	1	0	0
							00: Invert TA3FF 01: Set TA3FF 10: Clear TA1FF 11: Don’t care		1: TA3FF Invert Enable	0: TMRA2 1: TMRA3 inversion

8-bit Timer (2/2)

(6-3) TMRA45

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA45- RUN	Timer RUN	110H	TA4RDE				I2TA45	TA45PRUN	TA5RUN	TA4RUN
			R/W				R/W	R/W	R/W	R/W
			0				0	0	0	0
			Double Buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	8 bit Timer Run/Stop Control 0: Stop & Clear 1: Run (Count up)		
TA4REG	8-Bit Timer Register 0	112H (Prohibit RMW)	—							
			W							
			Undefined							
TA5REG	8-Bit Timer Register 1	113H (Prohibit RMW)	—							
			W							
			Undefined							
TA45- MOD	8-Bit Timer Source CLK & MODE	114H	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00: 8-Bit Timer 01: 16-Bit Timer 10: 8-Bit PPG 11: 8-Bit PWM		00: Reserved 01: $2^6 - 1$ PWM cycle 10: $2^7 - 1$ 11: $2^8 - 1$		00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TA4IN pin 01: ϕT1 10: ϕT4 11: ϕT16	
TA5FFCR	8-Bit Timer Flip-Flop Control	115H					TAFF5C1	TAFF5C0	TAFF5IE	TAFF5IS
							W*		R/W	
							1	1	0	0
							00: Invert TA5FF 01: SET TA5FF 10: Clear TA5FF 11: Don't care		1: TA5FF Invert Enable	0: Timer4 1: Timer5 inversion

(7) 16-Bit Timer (1/2)

(7-1) TMRB0

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB0RUN	Timer Control	180H	TB0RDE	—			I2TB0	TB0PRUN		TB0RUN
			R/W	R/W			R/W	R/W		R/W
			0	0			0	0		0
			Double Buffer 0: Disable 1: Enable	Always write 0.			IDLE2 0: Stop 1: Operate	16 Bit Timer Run/Stop control 0: Stop&Clear 1: Run (count up)		
TB0-MOD	16-Bit Timer Source CLK & MODE	182H	TB0CT1	TB0ET1	TB0CPOI	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			TB0FF1 INV TRG 0: TRG Disable 1: TRG Enable		0: Soft capture 1: Don’t care	Capture Timing (TB0IN0, TB0IN1) 00: disable 01: ↑, ↑ 10: ↑, ↓ 11: ↑, ↓ (TA1OUT)		1: UC0 Clear Enable	Source Clock 00: TB0IN0 pin 01: ϕT1 10: ϕT4 11: ϕT16	
TB0FFCR	16-Bit Timer Flip-Flop Control	183H	TB0FF1C1	TB0FF1C0	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	0	0
			00: Invert TB0FF1 01: Set 10: Clear 11: Don’t care		TB0FF0 Invert Trigger 0: trigger Disable 1: trigger Enable				00: Invert TB0FF0 01: Set 10: Clear 11: Don’t care	
TB0RG0L	16-Bit Timer Register 0L	188H (Prohibit RMW)	—							
			W							
			Undefined							
TB0RG0H	16-Bit Timer Register 0H	189H (Prohibit RMW)	—							
			W							
			Undefined							
TB0RG1L	16-Bit Timer Register 1L	18AH (Prohibit RMW)	—							
			W							
			Undefined							
TB0RG1H	16-Bit Timer Register 1H	18BH (Prohibit RMW)	—							
			W							
			Undefined							
TB0CP0L	Capture Register 0L	18CH	—							
			R							
			Undefined							
TB0CP0H	Capture Register 0H	18DH	—							
			R							
			Undefined							
TB0CP1L	Capture Register 1L	18EH	—							
			R							
			Undefined							
TB0CP1H	Capture Register 1H	18FH	—							
			R							
			Undefined							

(8) UART/Serial Channel

(8-1) UART/SIO Channel 0

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial Channel 0 Buffer	200H	RB7/TB7	RB6/TB6	RB5/TB5	RB4/TB4	RB3/TB3	RB2/TB2	RB1/TB1	RB0/TB0
			R (receiving)/W (transmission)							
			Undefined							
SC0CR	Serial Channel 0 Control	201H	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (cleared to 0 by reading)			R/W	
			0	0	0	0	0	0	0	0
			Receiving data bit 8	Parity 0: Odd 1: Even	1: Parity Enable	1: Error			0:SCLK0↑ 1:SCLK0↓	1: Input SCLK0 pin
						Over run	Parity	Framing		
SC0-MOD0	Serial Channel 0 Mode0	202H	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transmission data bit 8	1: CTS Enable	1: Receive Enable	1: Wake-up Enable	00: I/O Interface 01: UART 7-Bit 10: UART 8-Bit 11: UART 9-Bit		00: TA0TRG 01: baud rate generator 10: internal clock f_{SYS} 11: external clock SCLK0	
BR0CR	Baud Rate Control	203H	—	BR0ADD	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
			R/W							
			0	0	0		0	0	0	0
			Always write 0.	1: (16-K) /16 divided Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Set the frequency divisor N. 0 to F			
BR0-ADD	Serial Channel 0 K setting Reg	204H					BR0K3	BR0K2	BR0K1	BR0K0
							R/W			
							0	0	0	0
							Baud Rate0 K 1 to F			
SC0-MOD1	Serial Channel 0 Mode1	205H	I2S0	FDPX0						STSEN0
			R/W	R/W						W
			0	0						1
			IDLE2 0: Stop 1: Operate	I/O interface 1: Full Duplex 0: Half Duplex						STS0 1: Output 0: Stop

(8-2) UART/SIO Channel 1

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC1BUF	Serial Channel 1 Buffer	208H	RB7/TB7	RB6/TB6	RB5/TB5	RB4/TB4	RB3/TB3	RB2/TB2	RB1/TB1	RB0/TB0
			R (receiving)/W (transmission)							
			Undefined							
SC1CR	Serial Channel 1 Control	209H	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (cleared to 0 by reading)			R/W	
			0	0	0	0	0	0	0	0
			Receiving data bit 8	Parity 0: Odd 1: Even	1: Parity Enable	1: Error Over run	Parity	Framing	0:SCLK0↑ 1:SCLK0↓	1: Input SCLK0 pin
SC1-MOD0	Serial Channel 1 Mode0	20AH	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transmission data bit 8	1: CTS Enable	1: Receive Enable	1: Wake-up Enable	00: I/O Interface 01: UART 7-Bit 10: UART 8-Bit 11: UART 9-Bit		00: TA0TRG 01: baud rate generator 10: internal clock f_{SYS} 11: external clock SCLK0	
BR1CR	Baud Rate Control	20BH	—	BR1ADD	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
			R/W							
			0	0	0		0	0	0	0
			Always write 0.	1: (16-K) /16 divided Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Set the frequency divisor N. 0 to F			
BR1-ADD	Serial Channel 1 K setting Reg	20CH					BR1K3	BR1K2	BR1K1	BR1K0
							R/W			
							0	0	0	0
							Baud Rate0 K 1 to F			
SC1-MOD1	Serial Channel 1 Mode1	20DH	I2S1	FDPX1						STSEN1
			R/W	R/W						W
			0	0						1
			IDLE2 0: Stop 1: Operate	I/O interface 1: Full Duplex 0: Half Duplex						STS1 1: Output 0: Stop

(9) AD Converter

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD 0	AD MODE Reg0	2B0H	EOCF	ADBF	—	ITM1	ITM0	REPEAT	SCAN	ADS
			R		R/W	R/W	R/W	R/W	R/W	R/W
			0	0	0	0	0	0	0	0
			1: End	1: busy	Always write 0	Interrupt in Repeat Mode		1: Repeat	1: Scan	1: Start
ADMOD 1	AD MODE Reg1	2B1H	VREFON	I2AD			ADTRGE	ADCH2	ADCH1	ADCH0
			R/W	R/W			R/W	R/W		
			0	0			0	0	0	0
			1: VREF On	IDLE2 0: Abort 1: Operate			1: Enable for external start	Input channel 000: AN0 AN0 001: AN1 AN0 →AN1 010: AN2 AN0 → AN1 → AN2 011: AN3 AN0 → AN1 → AN2 → AN3 100: AN4 AN4 101: AN5 AN4 →AN5 110: AN6 AN4 → AN5 → AN6 111: AN7 AN4 → AN5 → AN6 → AN7		
ADMOD 2	AD MODE Reg2	2B2H	ADM27	ADM26	ADM25	ADM24	ADM23	ADM22	ADM21	ADM20
			R/W							
			0	0	0	1	0	0	0	1
			Please Write “1E”							
ADMOD 3	AD MODE Reg3	2B3H	ADM37	ADM36	ADM35	ADM34	ADM33	ADM32	ADM31	ADM30
			R/W							
			1	1	0	0	1	1	1	1
			Please Write “CF”							
AD REG04L	AD Result Reg 0/4 low	2A0H	ADR01	ADR00						ADR0RF
			R							R
			Undefined							0
AD REG04H	AD Result Reg 0/4 high	2A1H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			Undefined							
AD REG15L	AD Result Reg 1/5 low	2A2H	ADR11	ADR10						ADR1RF
			R							R
			Undefined							0
AD REG15H	AD Result Reg 1/5 high	2A3H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			Undefined							
AD REG26L	AD Result Reg 2/6 low	2A4H	ADR21	ADR20						ADR2RF
			R							R
			Undefined							0
AD REG26H	AD Result Reg 2/6 high	2A5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			Undefined							
AD REG37L	AD Result Reg 3/7 low	2A6H	ADR31	ADR30	—	—	—	—	—	ADR3RF
			R							R
			Undefined							0
AD REG37H	AD Result Reg 3/7 high	2A7H	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
			R							
			Undefined							

(10) Watchdog Timer

Symbol	Name	Address	7	6	5	4	3	2	1	0
WDMOD	WDT MODE Reg	300H	WDTE	WDTP1	WDTP0			I2WDT	RESCR	—
			R/W	R/W	R/W			R/W	R/W	R/W
			1	0	0			0	0	0
			1: WDT Enable	00: $2^{15}/f_{sys}$ 01: $2^{17}/f_{sys}$ 10: $829/f_{sys}$ 11: $2^{21}/f_{sys}$				IDLE2 0: Abort 1: Operate	1: RESET connect internally WDT out to Reset pin	Always write 0.
WDCR	WD Control	301H	—							
			W							
			—							
			B1H: WDT Disable 4EH: WDT Clear							

(11) Multi Vector Controllor

Symbol	Name	Address	7	6	5	4	3	2	1	0
MVEC0	MULI VECTA Control	00AEH	VEC7	VEC6	VEC5	VEC4	VEC3	VEC2	VEC1	VEC0
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			1	1	1	1	1	1	1	1
			Vector Address A15 to A8							

Symbol	Name	Address	7	6	5	4	3	2	1	0
MVEC1	MULI VECTA Control	00AFH	VEC15	VEC14	VEC13	VEC12	VEC11	VEC10	VEC9	VEC8
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			1	1	1	1	1	1	1	1
			Vector Address A23 to A16							

Notes
Write MVEC1,0 after making an interruption prohibition state.

6. Port Section Equivalent Circuit Diagrams

- Reading the Circuit Diagrams

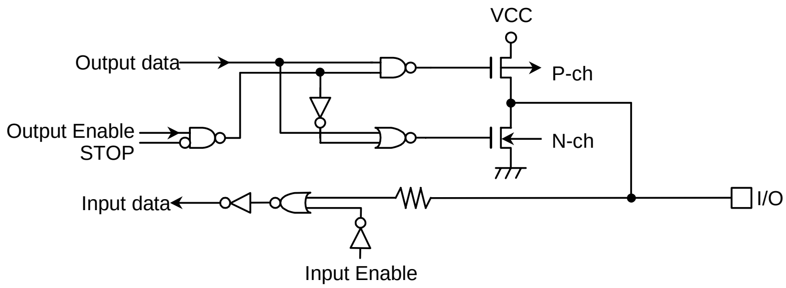
  The gate symbols used are essentially the same as those used for the standard CMOS logic IC [74HCXX] Series.

  The dedicated signal is described below.

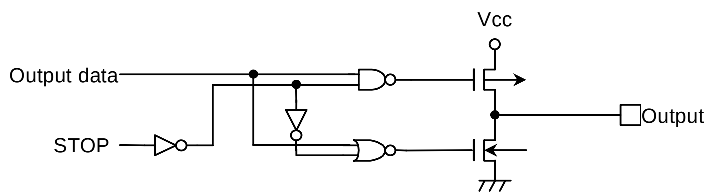
  STOP: This signal becomes Active (1) when the Halt Mode setting Register is set to STOP Mode (i.e. when SYSCR2 <HALTM1, 0> = 0, 1) and the CPU executes the HALT instruction. When the Drive Enable bit SYSCR2 <DRVE> is set to 1, however, STOP will remains at 0.

- The input protection resistances ranges from several tens of ohms to several hundreds of ohms.

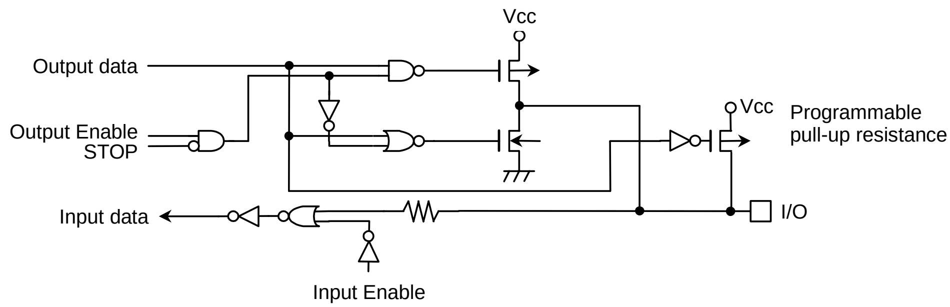
■ D0 to D7, P10 to P17, P20 to P27, A0 to A15, P71, P74, P90, P93 to P96



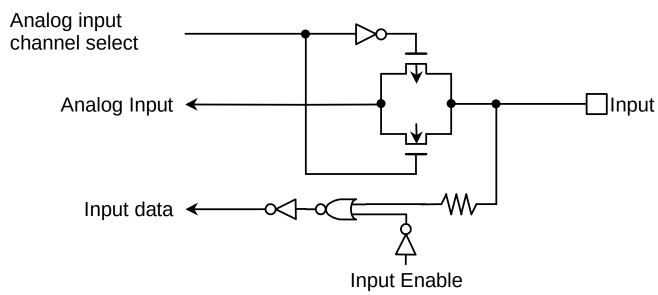
■ $\overline{RD}$, $\overline{WR}$, P60 to P63



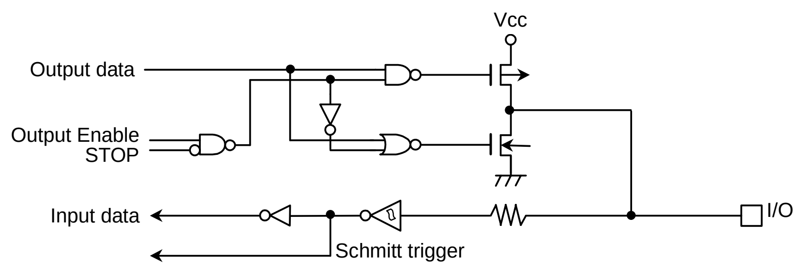
■ P53 to P55, P80 to P87, PZ2, PZ3



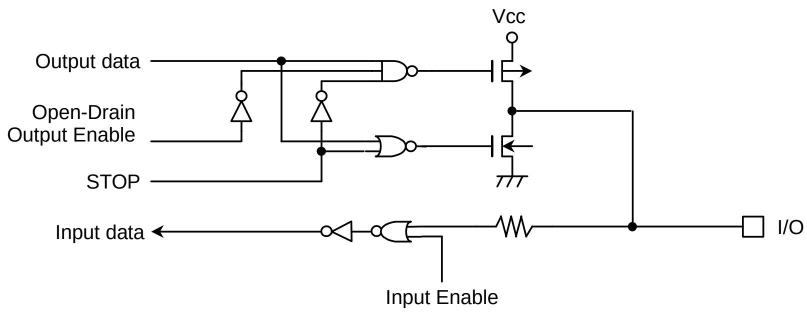
■ PA (AN0 to AN7)



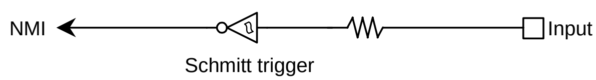
■ P56 (INT0), P70(INT1), P72(INT2),P73(INT3),P75(INT4),P90(INT5)



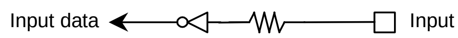
■ P80 (TXD0)



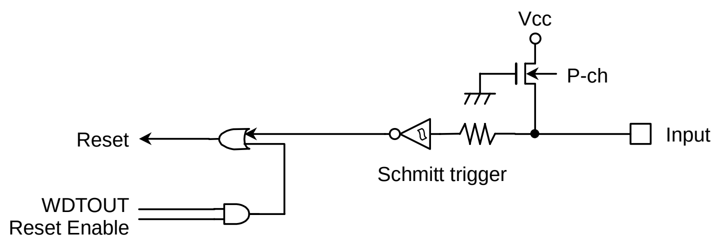
■ $\overline{\text{NMI}}$



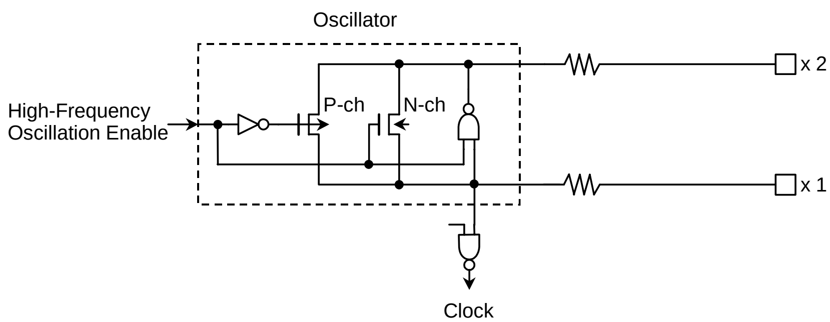
■ AM0 to AM1



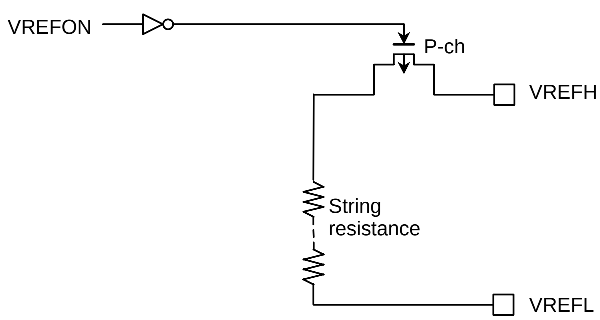
■ $\overline{\text{RESET}}$



■ X1 and X2



■ VREFH and VREFL



7. Points to Note and Restrictions

(1) Notation

a) The notation for built-in / I/O registers is as follows register symbol <bit symbol>
e.g.) TA01RUN <TA0RUN> denotes bit TA0RUN of register TA01RUN.

b) Read-modify-write instructions

An instruction in which the CPU reads data from memory and writes the data to the same memory location in one instruction.

Example 1) SET 3, (TA01RUN) ··· Set bit 3 of TA01RUN.

Example 2) INC 1, (100H) ··· Increment the data at 100H.

- Examples of read-modify-write instructions on the TLCS-900

Exchange instruction

EX (mem), R

Arithmetic operations

ADD (mem), R/# ADC (mem), R/#
SUB (mem), R/# SBC (mem), R/#
INC #3, (mem) DEC #3, (mem)

Logic operations

AND (mem), R/# OR (mem), R/#
XOR (mem), R/#

Bit manipulation operations

STCF #3/A, (mem) RES #3, (mem)
SET #3, (mem) CHG #3, (mem)
TSET #3, (mem)

Rotate and shift operations

RLC (mem) RRC (mem)
RL (mem) RR (mem)
SLA (mem) SRA (mem)
SLL (mem) SRL (mem)
RLD (mem) RRD (mem)

c) fc, f_{FPH}, f_{SYS} and one state

The clock frequency input on ins X1 and 2 is called f_{OSCH}. The clock selected by DFMCR0 <ACT1~ACT0> is called fc.

The clock selected by SYSCR1 <SYSCK> is called f_{FPH}. The clock frequency give by f_{FPH} divided by 2 is called f_{SYS}.

One cycle of f_{SYS} is referred to as one state.

(2) Points to note

a) AM0 and AM1 pins

Fix these pins to V_{CC} unless changing voltage.

b) EMU0and EMU1

Open pins.

c) Reserved address areas

The TMP91C829 noes not have any reserved areas.

d) Halt mode (IDLE1)

When IDLE1 Mode is used (in which oscillator operation only occurs), set RTCCR <RTCRUN> to 0 stop the timer for the real-time clock before the HALT instructions is executed.

e) Warm-up counter

The warm-up counter operates when STOP Mode is released, even if the system is using an external oscillator. As a result a time equivalent to the warm-up time elapses between input of the release request and output of the system clock.

f) Programmable pull-up resistance

The programmable pull-up resistor can be turned ON/OFF by a program when the ports are set for use as input ports. When the ports are set for use as output ports, they cannot be turned ON/OFF by a program.

The data registers (e.g. P3) are used to turn the pull-up/-down resistors ON/OFF. Consequently read-Modify-write instructions are prohibited.

g) Bus releasing function

Please refer to the Note about bus release in Section 3.5, Functions of Ports. The pin state is written when the bus is released.

h) Watchdog timer

The watchdog timer starts operation immediately after a Reset is released. When the watchdog timer is not to be used, disable it.

i) WatchDog timer

When the bus is released, neither internal memory nor internal I/O can be accessed. However, the internal I/O continues to operate. Hence the watchdog timer continues to run. Therefore be careful about the bus releasing time and set the detection timer of watchdog timer.

j) AD converter

The string resistor between the VREFH and VREFL pins can be cut by a program so as to reduce power consumption. When STOP Mode is used, disable the resistor using the program before the HALT instruction is executed.

k) CPU (micro DMA)

Only the LDC cr, r and LDC r, cr instructions can be used to access the control registers in the CPU (e.g. the Transfer Source Address Register (DMASn)).

l) Undefined SFR

The value of an undefined bit in an SFR is undefined when read.

m) POP SR instruction

Please execute the POP SR instruction during DI condition.