TOSHIBA

TOSHIBA Original CMOS 8-Bit Microcontroller

TLCS-870/C Series

TMP86FM29UG

TMP86FM29FG

TOSHIBA CORPORATION

Semiconductor Company

The information contained herein is subject to change without notice. 021023 _ D

TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress.
It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property.
In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications.
Also, please keep in mind the precautions and conditions set forth in the "Handling Guide for Semiconductor Devices," or "TOSHIBA Semiconductor Reliability Handbook" etc. 021023_A

The Toshiba products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.).
These Toshiba products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury ("Unintended Usage"). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc. Unintended Usage of Toshiba products listed in this document shall be made at the customer's own risk. 021023_B

The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations. 060106_Q

The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TOSHIBA or others. 021023_C

The products described in this document may include products subject to the foreign exchange and foreign trade laws. 021023_F

For a discussion of how the reliability of microcontrollers can be predicted, please refer to Section 1.3 of the chapter entitled Quality and Reliability Assurance/Handling Precautions. 030619_S

Revision History

Date	Revision	
2004/3/1	1	First Release
2008/8/29	2	Contents Revised

Caution in Setting the UART Noise Rejection Time

When UART is used, settings of RXDNC are limited depending on the transfer clock specified by BRG. The combination "O" is available but please do not select the combination "–".

The transfer clock generated by timer/counter interrupt is calculated by the following equation :

Transfer clock [Hz] = Timer/counter source clock [Hz] ÷ TTREG set value

BRG setting	Transfer clock [Hz]	RXDNC setting			
		00 (No noise rejection)	01 (Reject pulses shorter than 31/fc[s] as noise)	10 (Reject pulses shorter than 63/fc[s] as noise)	11 (Reject pulses shorter than 127/fc[s] as noise)
000	fc/13	O	O	O	–
110 (When the transfer clock generated by timer/counter interrupt is the same as the right side column)	fc/8	O	–	–	–
	fc/16	O	O	–	–
	fc/32	O	O	O	–
The setting except the above		O	O	O	O

Comparison table of TMP86C829B/H29B/M29B/PM29A/PM29B/C929AXB and TMP86FM29 ☐ Difference

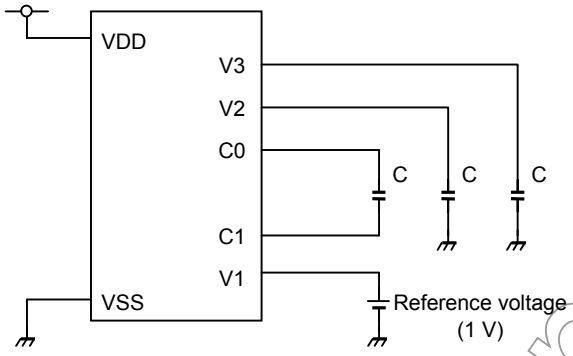
	TMP86C829B TMP86CH29B TMP86CM29B	TMP86PM29A TMP86PM29B	TMP86C929AXB (Emulation chip) (Note 3)	TMP86FM29F
ROM	8 K (Mask ROM) 16 K (Mask ROM) 32 K (Mask ROM)	32 K (OTP)	–	32 K (Flash)
RAM	512 1.5 K 1.5 K	1.5 K	–	2 K
I/O	42 pin		42 pin (MCU part)	42 pin
External Interrupt	5 pin			5 pin
AD Converter	10-bit AD converter × 8 ch			10-bit AD converter × 8 ch
Timer Counter	18-bit timer × 1 ch 8-bit timer × 4 ch			18-bit timer × 1 ch 8-bit timer × 4 ch
Serial Interface	8-bit UART / SIO × 1 ch			8-bit UART / SIO × 1 ch
LCD	32 seg × 4 com			32 seg × 4 com (Note 2)
Key-on Wakeup	4 ch			4 ch
Operating Voltage in MCU Mode	1.8 to 5.5 V at 4.2 MHz 2.7 to 5.5 V at 8 MHz 4.5 to 5.5 V at 16 MHz		1.8 to 5.25 V at 4.2 MHz 2.7 to 5.25 V at 8 MHz 4.5 to 5.25 V at 16 MHz	1.8 to 3.6 V at 4.2 MHz (External clock) 1.8 to 3.6 V at 8 MHz (Resonator) 2.7 to 3.6 V at 16 MHz
Operating Temperature in MCU Mode	−40 to 85°C		0 to 60°C	−40 to 85°C
Writing to Flash Memory	–			2.7 to 3.6V at 16 MHz 25°C ± 5°C
CPU Wait (Note 1)	N/A			Available

Note 1: The CPU wait is a CPU halt function for stabilizing of power supply of Flash memory. The CPU wait period is as follows. In the CPU wait period except RESET, CPU is halted but peripheral functions are not halted. Therefore, if the interrupt occurs during the CPU wait period, the interrupt latch is set. In this case, if the IMF has been set to "1", the interrupt service routine is executed after CPU wait period. For details refer to 2.14 "Flash Memory" in TMP86FM29 data sheet.

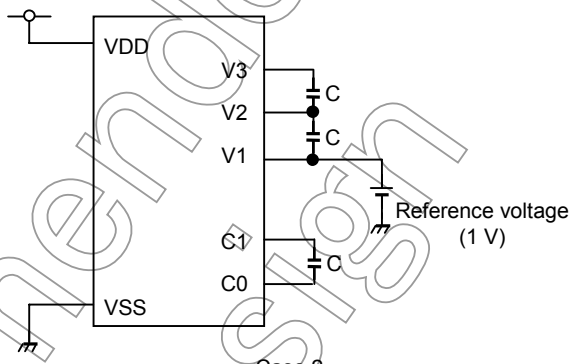
Thus, even if the same software is executed in 86FM29 and 86C829B/H29B/M29B/PM29A/PM29B /C929AXB, the operation process is not the same. Therefore, when the final operating confirmation on target application is executed for software development of Mask ROM Product (86C829B/H29B/M29B), not the Flash product (86FM29) but the OTP product (86PM29A/PM29B) should be used.

Condition	Wait Time	Halt/Operate	
		CPU	Peripherals
After reset release	2^{10}/fc[s]	Halt	Halt
Changing from STOP mode to NORMAL mode (at EEPCR<MNPWDW> = "1")	2^{10}/fc[s]	Halt	Operate
Changing from STOP mode to SLOW mode (at EEPCR<MNPWDW> = "1")	2^{3}/fs[s]	Halt	Operate
Changing from IDLE0/1/2 mode to NORMAL mode (at EEPCR<ATPWDW> = "0")	2^{10}/fc[s]	Halt	Operate
Changing from SLEEP0/1/2 mode to SLOW mode (at EEPCR<ATPWDW> = "0")	2^{3}/fs[s]	Halt	Operate

Note 2: The 86FM29 can not drive the 5V LCD panel because the electrical characteristics in 86FM29 is altered from 86C829B/H29B/M29B/PM29A/PM29B/C929AXB. The recommended operating condition of V3 pin in TMP86FM29 is 3.6V(max). For details, refer to “Electrical Characteristics”.
When the LCD booster circuit is used in 86FM29, connect the reference voltage and capacitor as shown in "case2". Though the method of "case1" has been recommended in 86C829B /H29B/M29B/PM29A/PM29B datasheets, the 86FM29 should not use method of "case1". Even if the method of "case1" is used in the 86C829B/H29B/M29B/PM29A/PM29B/C929AXB, the function and operation are not issue at all. However, if the "case2" is used, the booster ability becomes higher than "case1". Therefore, when the application board is designed newly in future, the method of "case2" is also recommended in 86C829B/H29B/M29B/PM29A/PM29B/C929AXB.



Case 1
(For 86C829B/H29B/M29B/PM29A/PM29B/C929AXB)



Case 2
(For 86FM29 and 86C829B/H29B/M29B/
PM29A/PM29B/C929AXB)

Note 3: Flash function, CPU wait period and serial PROM mode cannot be emulated in the 86C929AXB. If the software including the flash function is executed in 86C929AXB, the operation process differs from 86FM29.

CMOS 8-Bit Microcontroller

TMP86FM29UG/FG

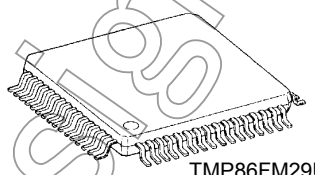
The TMP86FM29 is the high-speed, high-performance and low power consumption 8-bit microcomputer, including FLASH, RAM, LCD driver, multi-function timer/counter, serial interface (UART/SIO), a 10-bit AD converter and two clock generators on chip.

Product No.	FLASH	RAM	Package	Emulation Chip
TMP86FM29UG	32768 × 8 bits	1536 × 8 bits	P-LQFP64-1010-0.50E	TMP86C929AXB
TMP86FM29FG			P-QFP64-1414-0.80C	

Feautures

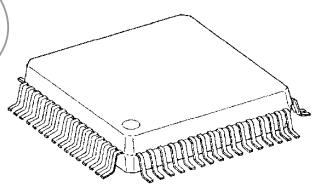
- 8-bit single chip microcomputer TLCS-870/C series
- Instruction execution time: 0.25 μs (at 16 MHz)
  122 μs (at 32.768 kHz)
- 132 types and 731 basic instructions
- 19 interrupt sources (External: 5, Internal: 14)
- Input/Output ports (39 pins)
  (Out of which 24 pins are also used as SEG pins)
- 18-bit timer counter: 1 ch
  - Timer, Event counter,
    Pulse width measurement,
    Frequencymeasurement modes
- 8-bit timer counter: 4 ch
  - Timer, Event counter,
    PWM output, Programmable divider output,
    PPG output modes
- Time Base Timer
- Divider output function

P-LQFP64-1010-0.50E



TMP86FM29UG

P-QFP64-1414-0.80C

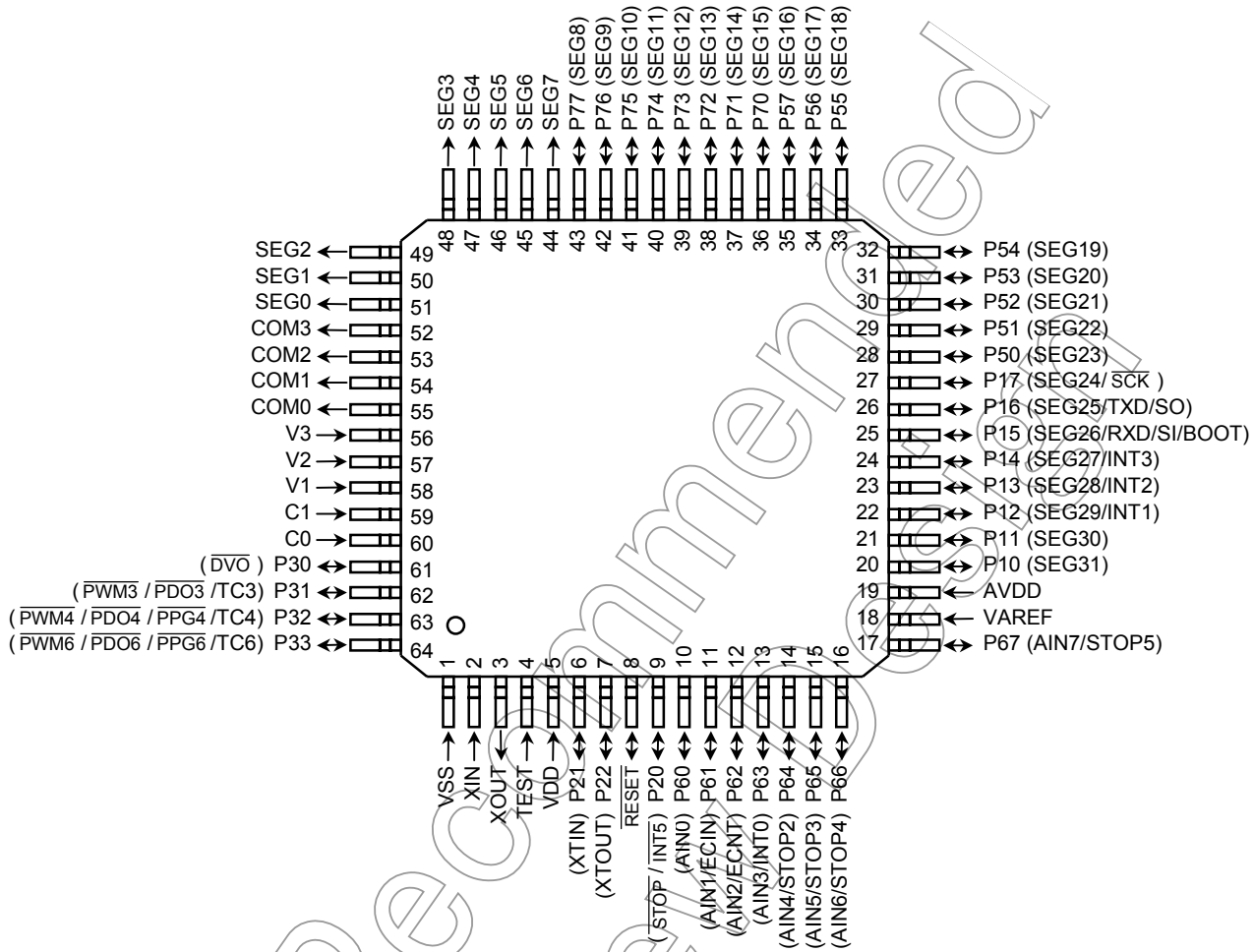


TMP86FM29FG

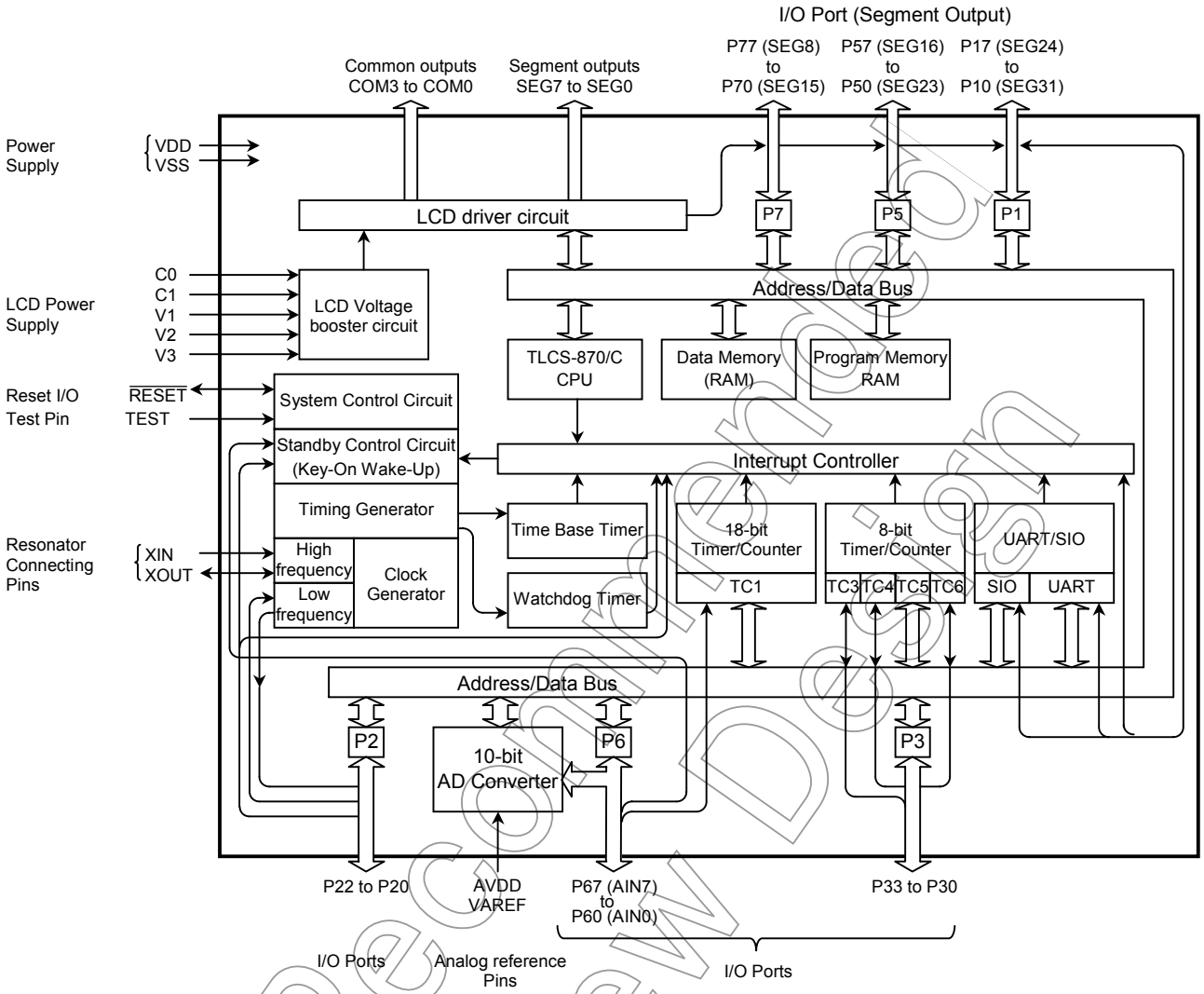
- Watchdog Timer
  - Interrupt source/reset output (programmable)
- Serial interface
  - 8-bit UART/SIO: 1ch
- 10-bit successive approximation type AD converter
  - Analog input: 8 ch
- Four Key-On Wake-Up pins
- LCD driver/controller
  - Built-in voltage booster for LCD driver
  - With displaymemory
  - LCD direct drive capability (max 32 seg × 4 com)
  - 1/4, 1/3, 1/2duties or static drive are programmably selectable
- Dual clock operation
  - Single/Dual-clock mode
- Nine power saving operating modes
  - STOP mode : Oscillation stops. Battery/Capacitor back-up.
    Port output hold/High-impedance.
  - SLOW 1, 2 mode : Low power consumption operation using low-frequency clock (32.768 kHz)
  - IDLE 0 mode : CPU stops, and peripherals operate using high-frequency clock of
    Time-Base-Timer. Release by falling edge of TBTCR <TBTCK> setting.
  - IDLE 1 mode : CPU stops, and peripherals operate using high-frequency clock.
    Release by interruputs.
  - IDLE 2 mode : CPU stops, and peripherals operate using high and low frequency clock.
    Release by interruputs.
  - SLEEP 0 mode : CPU stops, and peripherals operate using low-frequency clock of
    Time-Base-Timer. Release by falling edge of TBTCR <TBTCK> setting.
  - SLEEP 1 mode : CPU stops, and peripherals operate using low-frequency clock.
    Release by interrupts.
  - SLEEP 2 mode : CPU stops, and peripherals operate using high and low frequency clock.
    Release by interrupts.
- Wide operating voltage: 1.8 to 3.6 V at 8 MHz/32.768 kHz
  2.7 to 3.6 V at 16 MHz/32.768 kHz

Pin Assignments (Top View)

P-LQFP64-1010-0.50E
P-QFP64-1414-0.80C



Block Diagram



Pin Functions

Pin Name	Input/Output	Functions		
P17 (SEG24, $\overline{SCK}$)	I/O (I/O)	8-bit input/output port with latch. When used as input port, an external interrupt input, serial interface input/output and UART data input/output, the P1LCR must be set to "0" after setting output latch to "1". When used as a LCD segment output, the P1LCR must be set to "1".	Serial clock input/output	LCD segment outputs.
P16 (SEG25, TxD, SO)	I/O (Output)		UART data output Serial data output	
P15 (SEG26, RxD, SI BOOT)	I/O (I/O)		UART data input Serial data input Serial PROM mode control input	
P14 (SEG27, INT3)	I/O (I/O)		External interrupt 3 input	
P13 (SEG28, INT2)	I/O (I/O)		External interrupt 2 input	
P12 (SEG29, INT1)	I/O (I/O)		External interrupt 1 input	
P11 (SEG30)	I/O (Output)			
P10 (SEG31)	I/O (Output)			
P22 (XTOUT)	I/O (Output)	3-bit input/output port with latch. When used as an input port, the output latch must be set to "1".	Resonator connecting pins (32.768 kHz) For inputting external clock, XTIN is used and XOUT is opened.	
P21 (XTIN)	I/O (Input)			
P20 ($\overline{INT5}$, $\overline{STOP}$)	I/O (Input)		External interrupt input 5 or STOP mode release signal input	
P33 ($\overline{PWM6}$, $\overline{PDO6}$ $\overline{PPG6}$, TC6)	I/O (I/O)	4-bit programmable input/output port (Nch high current output). When used as a timer/counter output or divider output, the output latch must be set to "1". When used as an input port or timer/counter input, the P3OUTCR must be set to "0" after P3DR is set to "1".	Timer counter 6 input/output	
P32 ($\overline{PWM4}$, $\overline{PDO4}$ $\overline{PPG4}$, TC4)	I/O (I/O)		Timer counter 4 input/output	
P31 ($\overline{PWM3}$, $\overline{PDO3}$, TC3)	I/O (I/O)		Timer counter 3 input/output	
P30 ($\overline{DVO}$)	I/O (Output)		Divider output	
P57 (SEG16) to P50 (SEG23)	I/O (Output)	8-bit input/output port with latch. When used as a LCD segment output, the P5LCR must be set to "1".	LCD segment outputs	
P67 (AIN7, STOP5)	I/O (Input)	8-bit programmable input/output port (tri-state). Each bit of this port can be individually configured as an input or an output under software control. When used as an analog input, the P6CR must be set to "0" after setting output latch to "0". When used as an input port, a key on wake up input, an external interrupt input and timer/ counter input, the P6CR must be set to "0" after setting output latch to "1".	STOP 5 input	AD converter analog inputs
P66 (AIN6, STOP4)	I/O (Input)		STOP 4 input	
P65 (AIN5, STOP3)	I/O (Input)		STOP 3 input	
P64 (AIN4, STOP2)	I/O (Input)		STOP 2 input	
P63 (AIN3, $\overline{INT0}$)	I/O (Input)		External interrupt 0 input	
P62 (AIN2, ECNT)	I/O (Input)		Timer/Counter 1 input	
P61 (AIN1, ECIN)	I/O (Input)			
P60 (AIN0)	I/O (Input)			
P77 (SEG8) to P70 (SEG15)	I/O (Output)	8-bit input/output port with latch. When used as a LCD segment output, the P7LCR must be set to "1".	LCD segment outputs	
SEG7 to SEG0	Output	LCD segment outputs		
COM3 to COM0		LCD common outputs		
V3 to V1	LCD voltage booster pin	LCD voltage booster pin. Capacitors are required between C0 and C1 pin and V1/V2/V3 pin and GND.		
C1 to C0				
XIN, XOUT	Input Output	Resonator connecting pins for high-frequency clock. For inputting external clock, XIN is used and XOUT is opened.		
$\overline{RESET}$	I/O	Reset signal input or watchdog timer output/address-trap-reset output/system clock reset output		
TEST	Input	Test pin for out-going test, and the serial PROM mode control pin. Usually be fixed to low level. When the serial PROM mode starts, be fixed to "1".		
VDD, VSS	Power Supply	+5 V, 0 (GND)		
VAREF		Analog reference voltage inputs (High)		
AVDD		AD circuit power supply		

Operational Description

1. CPU Core Functions

The CPU core consists of a CPU, a system clock controller, and an interrupt controller.

This section provides a description of the CPU core, the program memory, the data memory, the external memory interface, and the reset circuit.

1.1 Memory Address Map

The TMP86FM29 memory consists of 5 blocks: FLASH memory, BOOT ROM, RAM, DBR (Data buffer register) and SFR (Special function register). They are all mapped in 64-Kbyte address space. Figure 1.1.1 shows the TMP86FM29 memory address map. The general-purpose registers are not assigned to the RAM address space.

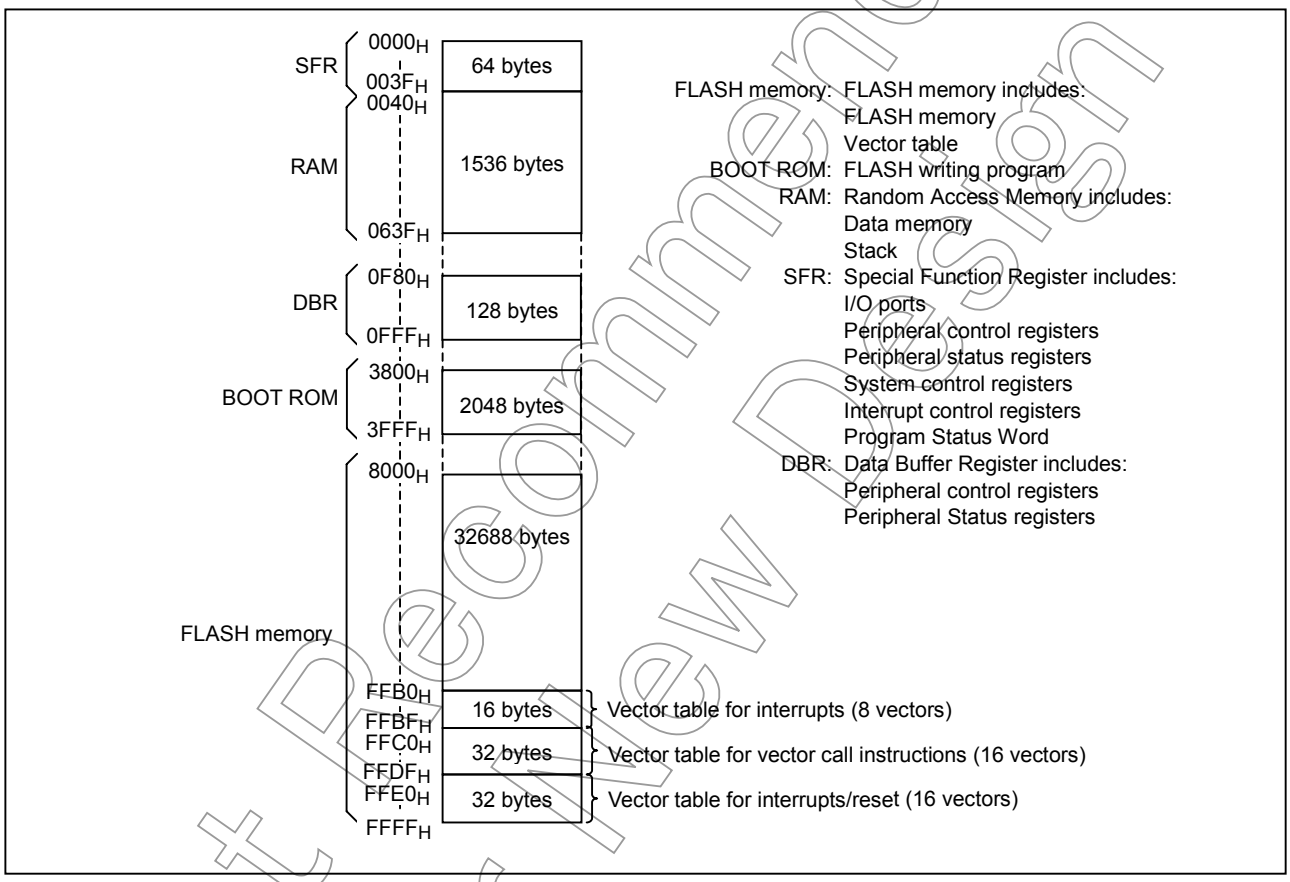


Figure 1.1.1 Memory Address Maps

1.2 Program Memory (FLASH)

The TMP86FM29 has a 32 K × 8 bits (Address 8000_H to $FFFF_H$) of Flash memory.

1.3 Data Memory (RAM)

The TMP86FM29 has 1536 bytes of internal RAM. The first 192 bytes (0040_H to $00FF_H$) of the internal RAM are located in the direct area; instructions with shorten operations are available against such an area.

The data memory contents become unstable when the power supply is turned on; therefore, the data memory should be initialized by an initialization routine.

```
Example:  Clears RAM to "00H".
          LD      HL, 0040H       ;   Start address setup
          LD      A, H            ;   Initial value (00H) setup
          LD      BC, 05FFH       ;
SRAMCLR:  LD      (HL), A
          INC     HL
          DEC     BC
          JRS     F, SRAMCLR
```

1.4 System Clock Controller

The system clock controller consists of a clock generator, a timing generator, and a standby controller.

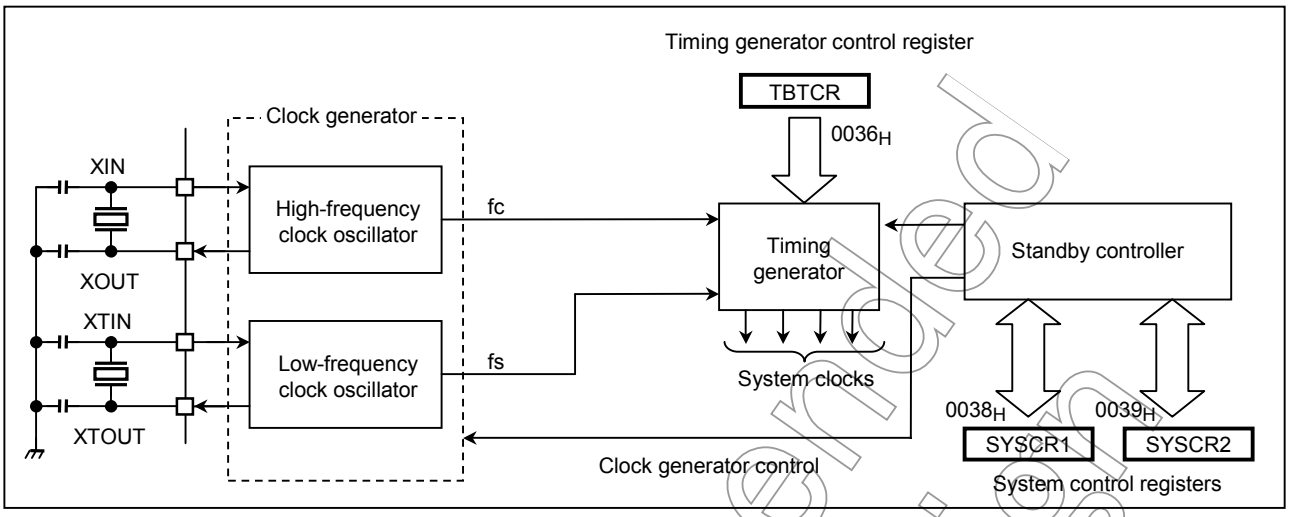


Figure 1.4.1 System Clock Control

1.4.1 Clock Generator

The clock generator generates the basic clock which provides the system clocks supplied to the CPU core and peripheral hardware. It contains two oscillation circuits: one for the high-frequency clock and one for the low-frequency clock. Power consumption can be reduced by switching of the standby controller to low-power operation based on the low-frequency clock.

The high-frequency (fc) and low-frequency (fs) clocks can easily be obtained by connecting a resonator between the XIN/XOUT and XTIN/XTOUT pins respectively. Clock input from an external oscillator is also possible. In this case, external clock is applied to XIN/XTIN pin with XOUT/XTOUT pin not connected.

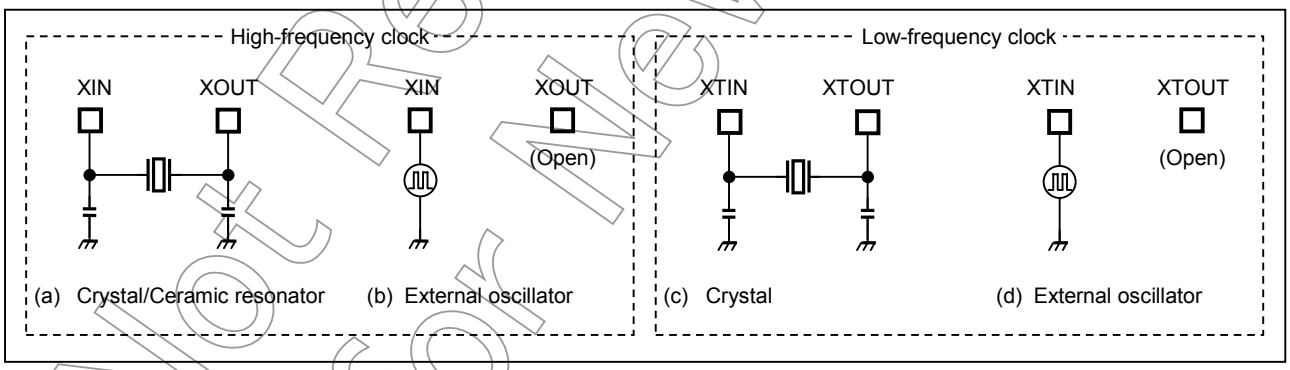


Figure 1.4.2 Examples of Resonator Connection

Note: The function to monitor the basic clock directly at external is not provided for hardware, however, with disabling all interrupts and watchdog timers, the oscillation frequency can be adjusted by monitoring the pulse which the fixed frequency is outputted to the port by the program.
The system to require the adjustment of the oscillation frequency should create the program for the adjustment in advance.

1.4.2 Timing Generator

The timing generator generates the various system clocks supplied to the CPU core and peripheral hardware from the basic clock (fc or fs). The timing generator provides the following functions.

a. Generation of main system clock
b. Generation of divider output ($\overline{DVO}$) pulses
c. Generation of source clocks for time base timer
d. Generation of source clocks for watchdog timer
e. Generation of internal source clocks for timer/counters and serial interface
f. Generation of warm-up clocks for releasing STOP mode

(1) Configuration of timing generator

The timing generator consists of a 2-stage prescaler, a 21-stage divider, a main system clock generator, and machine cycle counters.

An input clock to the 7th stage of the divider depends on the operating mode, TBTCR<DV7CK>, that is shown in Figure 1.4.4. As reset and STOP mode started/canceled, the prescaler and the divider are cleared to "0".

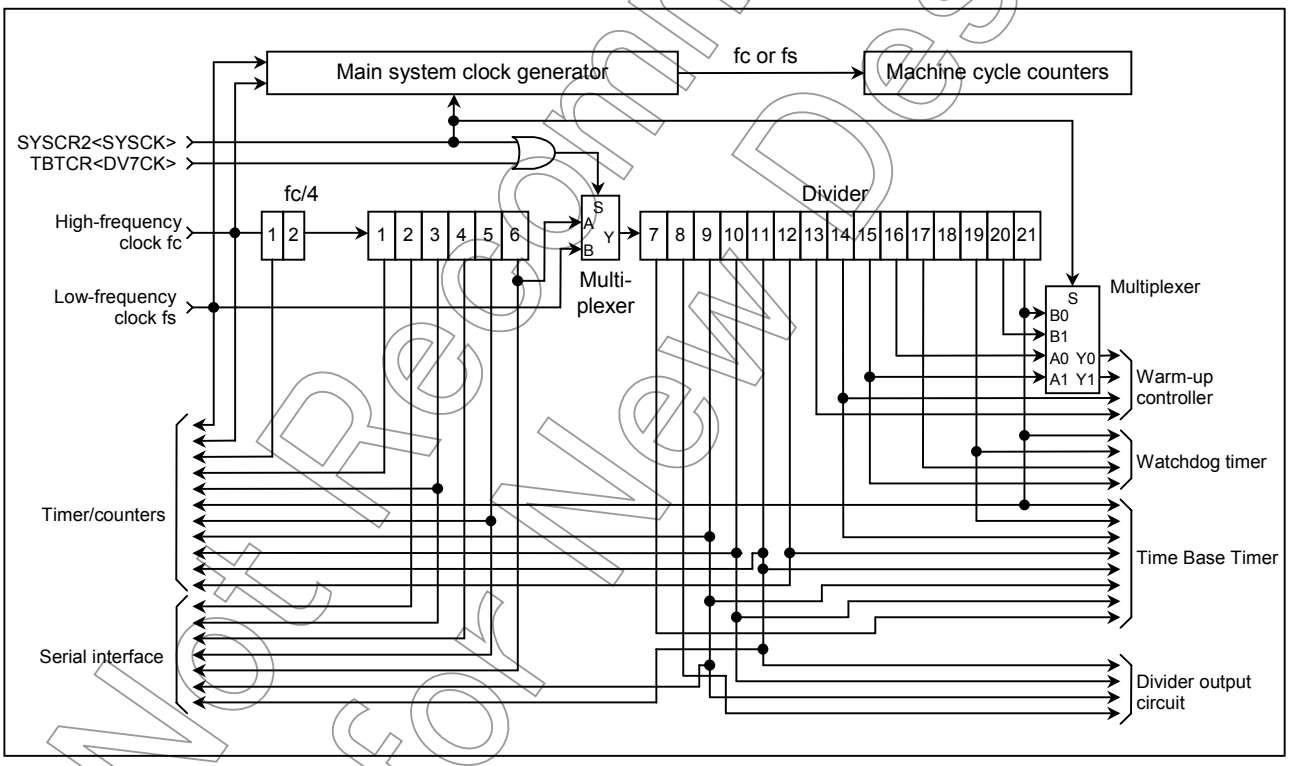


Figure 1.4.3 Configuration of Timing Generator

TBTCR (0036_H)

7	6	5	4	3	2	1	0
(DVOEN)	(DVOCK)		DV7CK	(TBTEN)	(TBTCK)		

(Initial value: 0000 0000)

DV7CK	Selection of input to the 7th stage of the divider	0: $fc/2^8$ [Hz] 1: fs	R/W

Note 1: In single clock mode, do not set DV7CK to "1".

Note 2: Do not set "1" on DV7CK while the low-frequency clock is not operated stably.

Note 3: fc: High-frequency clock [Hz], fc: Low-frequency clock [Hz], *: Don't care

Note 4: In SLOW1/2 and SLEEP1/2 modes, the DV7CK setting is ineffective, and fs is input to the 7th stage of the divider.

Note 5: When STOP mode is entered from NORMAL 1/2 mode, the DV7CK setting is ineffective during the warm-up period after release of STOP mode, and the 6th stage of the divider is input to the 7th stage during this period.

Figure 1.4.4 Timing Generator Control Register

(2) Machine cycle

Instruction execution and peripheral hardware operation are synchronized with the main system clock.

The minimum instruction execution unit is called an "machine cycle". There are a total of 10 different types of instructions for the TLCS-870/C series: ranging from 1-cycle instructions which require one machine cycle for execution to 10-cycle instructions which require 10 machine cycles for execution.

A machine cycle consists of 4 states (S0 to S3), and each state consists of one main system clock.

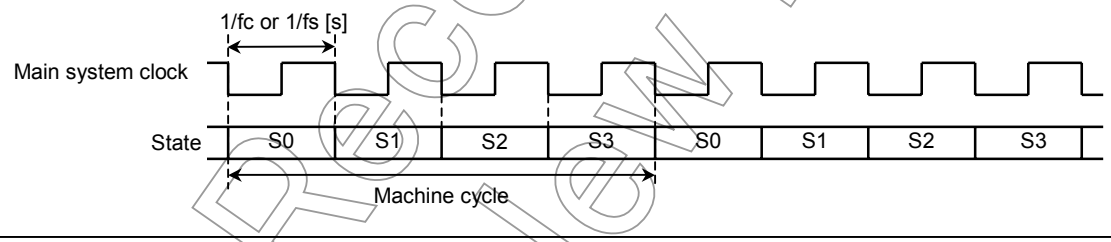


Figure 1.4.5 Machine Cycle

1.4.3 Operation Mode Control Circuit

The operation mode control circuit starts and stops the oscillation circuits for the high-frequency and low-frequency clocks, and switches the main system clock. There are two operating modes: single-clock and dual-clock. These modes are controlled by the system control registers (SYSCR1 and SYSCR2).

Figure 1.4.6 shows the operating mode transition diagram and Figure 1.4.7 shows the system control registers.

(1) Single-clock mode

Only the oscillation circuit for the high-frequency clock is used, and P21 (XTIN) and P22 (XTOUT) pins are used as input/output ports. The main-system clock is obtained from the high-frequency clock. In the single-clock mode, the machine cycle time is 4/fc [s].

a. NORMAL1 mode

In this mode, both the CPU core and on-chip peripherals operate using the high-frequency clock.

The TMP86FM29 is placed in this mode after reset.

b. IDLE1 mode

In this mode, the internal oscillation circuit remains active. The CPU and the watchdog timer are halted; however on-chip peripherals remain active (Operate using the high-frequency clock).

IDLE1 mode is started by SYSCR2<IDLE>, and IDLE1 mode is released to NORMAL1 mode by an interrupt request from the on-chip peripherals or external interrupt inputs. When the IMF (Interrupt master enable flag) is "1" (Interrupt enable), the execution will resume with the acceptance of the interrupt, and the operation will return to normal after the interrupt service is completed. When the IMF is "0" (Interrupt disable), the execution will resume with the instruction which follows the IDLE1 mode start instruction.

c. IDLE0 mode

In this mode, all the circuit, except oscillator and the time-base-timer, stops operation.

This mode is enabled by setting "1" on bit TGHALT on the system control register 2 (SYSCR2).

When IDLE0 mode starts, the CPU stops and the timing generator stops feeding the clock to the peripheral circuits other than TBT. Then, upon detecting the falling edge of the source clock selected with TBTCR<TBTCK>, the timing generator starts feeding the clock to all peripheral circuits.

When returned from IDLE0 mode, the CPU restarts operating, entering NORMAL1 mode back again. IDLE0 mode is entered and returned regardless of how TBTCR<TBTEN> is set. When IMF = "1", EF6 (TBT interrupt individual enable flag) = "1", and TBTCR<TBTEN> = "1", interrupt processing is performed. When IDLE0 mode is entered while TBTCR<TBTEN> = "1", the INTTBT interrupt latch is set after returning to NORMAL1 mode.

(2) Dual-clock mode

Both the high-frequency and low-frequency oscillation circuits are used in this mode. P21 (XTIN) and P22 (XTOUT) pins cannot be used as input/output ports. The main system clock is obtained from the high-frequency clock in NORMAL2 and IDLE2 modes, and is obtained from the low-frequency clock in SLOW and SLEEP modes. The machine cycle time is 4/fc [s] in the NORMAL2 and IDLE2 modes, and 4/fs [s] (122 μs at fs = 32.768 kHz) in the SLOW and SLEEP modes.

The TLCS-870/C is placed in the single-clock mode during reset. To use the dual-clock mode, the low-frequency oscillator should be turned on at the start of a program.

a. NORMAL2 mode

In this mode, the CPU core operates with the high-frequency clock. On-chip peripherals operate using the high-frequency clock and/or low-frequency clock.

b. SLOW2 mode

In this mode, the CPU core operates with the low-frequency clock, while both the high-frequency clock and the low-frequency clock are operated. On-chip peripherals are triggered by the low-frequency clock. As the SYSCK on SYSCR2 becomes "0", the hardware changes into NORMAL2 mode. As the XEN on SYSCR2 becomes "0", the hardware changes into SLOW1 mode. Do not clear XTEN to "0" during SLOW2 mode.

c. SLOW1 mode

This mode can be used to reduce power-consumption by turning off oscillation of the high-frequency clock. The CPU core and on-chip peripherals operate using the low-frequency clock.

Switching back and forth between SLOW1 and SLOW2 modes are performed by XEN bit on the system control register 2 (SYSCR2). In SLOW1 and SLEEP mode, the input clock to the 1st stage of the divider is stopped; output from the 1st to 6th stages is also stopped.

d. IDLE2 mode

In this mode, the internal oscillation circuit remain active. The CPU and the watchdog timer are halted; however, on-chip peripherals remain active (Operate using the high-frequency clock and/or the low-frequency clock). Starting and releasing of IDLE2 mode are the same as for IDLE1 mode, except that operation returns to NORMAL2 mode.

e. SLEEP1 mode

In this mode, the internal oscillation circuit of the low-frequency clock remains active. The CPU, the watchdog timer, and the internal oscillation circuit of the high-frequency clock are halted; however, on-chip peripherals remain active (Operate using the low-frequency clock). Starting and releasing of SLEEP mode are the same as for IDLE1 mode, except that operation returns to SLOW mode. In SLOW and SLEEP mode, the input clock to the 1st stage of the divider is stopped; output from the 1st to 6th stages is also stopped.

f. SLEEP2 mode

The SLEEP2 mode is the IDLE mode corresponding to the SLOW2 mode. The status under the SLEEP2 mode is same as that under the SLEEP1 mode, except for the oscillation circuit of the high-frequency clock.

g. SLEEP0 mode

In this mode, all the circuit, except oscillator and the time-base-timer, stops operation.

This mode is enabled by setting "1" on bit TGHALT on the system control register 2 (SYSCR2).

When SLEEP0 mode starts, the CPU stops and the timing generator stops feeding the clock to the peripheral circuits other than TBT. Then, upon detecting the falling edge of the source clock selected with TBTCR<TBTCK>, the timing generator starts feeding the clock to all peripheral circuits.

When returned from SLEEP0 mode, the CPU restarts operating, entering SLOW1 mode back again. SLEEP0 mode is entered and returned regardless of how TBTCR<TBTEN> is set. When IMF = "1", EF6 (TBT interrupt individual enable flag) = "1", and TBTCR<TBTEN> = "1", interrupt processing is performed. When SLEEP0 mode is entered while TBTCR<TBTEN> = "1", the INTTBT interrupt latch is set after returning to SLOW1 mode.

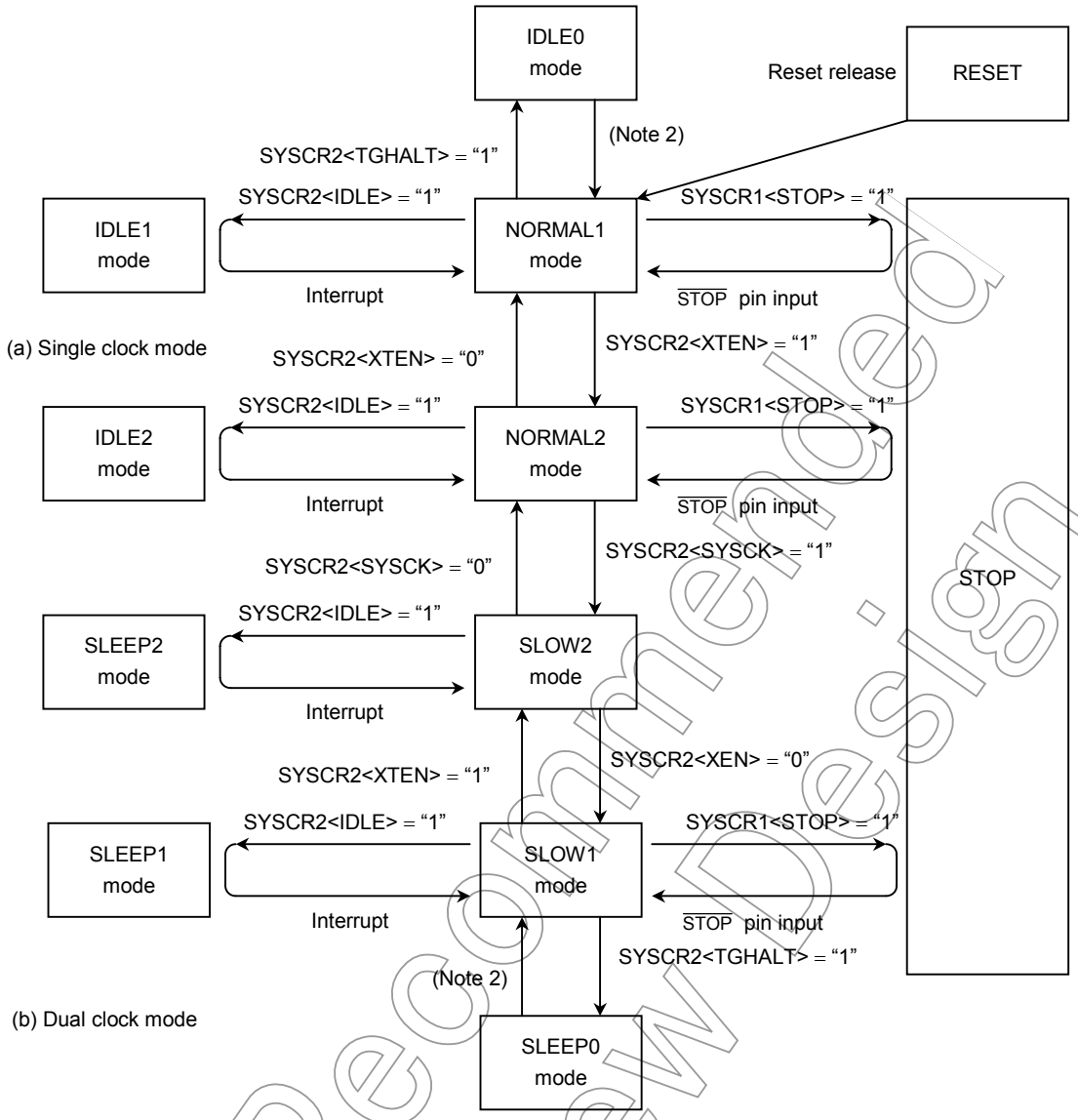
(3) STOP mode

In this mode, the internal oscillation circuit is turned off, causing all system operations to be halted. The internal status immediately prior to the halt is held with a lowest power consumption during STOP mode.

STOP mode is started by the system control register 1 (SYSCR1), and STOP mode is released by a inputting (either level-sensitive or edge-sensitive can be programmably selected) to the $\overline{\text{STOP}}$ pin or key on wake up pin input which is enabled by STOPCR. After the warm-up period is completed, the execution resumes with the instruction which follows the STOP mode start instruction.

Note 1: When the IDLE0/1/2 and SLEEP0/1/2 modes are started with the EEPCR<ATPWDW> = "0", the CPU wait period for stabilizing of the power supply of Flash control circuit is executed after being released from these mode.

Note 2: When the STOP mode is started with the EEPCR<MNPWDW> = "1", the CPU wait period for stablizing of the power supply of Flash control circuit is executed after in the STOP warm-up time.



Note 1: NORMAL1 and NORMAL2 modes are generically called NORMAL; SLOW1 and SLOW2 are called SLOW; IDLE0, IDLE1 and IDLE2 are called IDLE; SLEEP0, SLEEP1 and SLEEP2 are called SLEEP.

Note 2: The mode is released by falling edge of TBTCR<TBTCK> setting.

Operating Mode		Oscillator		CPU Core	TBT	Other Peripherals	Machine Cycle Time
		High Frequency	Low Frequency				
Single Clock	RESET	Oscillation	Stop	Reset	Reset	Reset	4/fc [s]
	NORMAL1			Operate	Operate	Operate	
	IDLE1			Halt			
	IDLE0					Halt	–
	STOP	Stop			Halt		
Dual Clock	NORMAL2	Oscillation	Oscillation	Operate with high frequency	Operate	Operate	4/fc [s]
	IDLE2			Halt			
	SLOW2			Operate with low frequency			4/fs [s]
	SLEEP2			Halt			
	SLOW1	Stop		Operate with low frequency			
	SLEEP1			Halt			
	SLEEP0					Halt	–
	STOP		Stop		Halt		

Figure 1.4.6 Operating Mode Transition Diagram

System Control Register 1

SYSCR1 (0038_H)

7	6	5	4	3	2	1	0
STOP	RELM	RETM	OUTEN	WUT			

(Initial value: 0000 00**)

STOP	STOP mode start	0: CPU core and peripherals remain active 1: CPU core and peripherals are halted (Start STOP mode)		R/W
RELM	Release method for STOP pin (P20)	0: Edge-sensitive release 1: Level-sensitive release		
RETM	Operating mode after STOP mode	0: Return to NORMAL1/2 mode 1: Return to SLOW1 mode		
OUTEN	Port output during STOP mode	0: High impedance 1: Output kept		
WUT	Warm-up time at releasing STOP mode (Note 8)		Return to NORMAL mode	Return to SLOW mode
		00	3×2^{16}/fc + (2^{19}/fc)	3×2^{13}/fs + (2^3/fs)
		01	2^{16}/fc + (2^{10}/fc)	2^{13}/fs + (2^3/fs)
		10	3×2^{14}/fc + (2^{10}/fc)	3×2^6/fs + (2^3/fs)
		11	2^{14}/fc + (2^{10}/fc)	2^6/fs + (2^3/fs)

Note 1: Always set RETM to “0” when transiting from NORMAL mode to STOP mode. Always set RETM to “1” when transiting from SLOW mode to STOP mode.

Note 2: When STOP mode is released with $\overline{\text{RESET}}$ pin input, a return is made to NORMAL1 regardless of the RETM contents.

Note 3: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don’t care

Note 4: Bits 1 and 0 in SYSCR1 are read as undefined data when a read instruction is executed.

Note 5: As the hardware becomes STOP mode under OUTEN = “0”, input value is fixed to “0”; therefore it may cause interrupt request on account of falling edge.

Note 6: When the key-on wake-up input (STOP2 to STOP5) is used, RELM should be set to “1”.

Note 7: Port P20 is used as $\overline{\text{STOP}}$ pin. Therefore, when stop mode is started, OUTEN does not affect to P20, and P20 becomes High-Z mode.

Note 8: When the STOP mode is started with the EEPCR<MNPWDW> = “1”, the CPU wait period for stabilizing of the power supply of Flash control circuit is executed after in the STOP warm-up time.
(The CPU wait period for FLASH is shown in parentheses)

System Control Register 2

SYSCR2 (0039_H)

7	6	5	4	3	2	1	0
XEN	XTEN	SYSCK	IDLE		TGHALT		

(Initial value: 1000 *0**)

XEN	High-frequency oscillator control	0: Turn off oscillation 1: Turn on oscillation	R/W
XTEN	Low-frequency oscillator control	0: Turn off oscillation 1: Turn on oscillation	
SYSCK	Main system clock select (write)/main system clock monitor (read)	0: High-frequency clock 1: Low-frequency clock	
IDLE	CPU and watchdog timer control (IDLE1/2, SLEEP1/2 mode)	0: CPU and watchdog timer remain active 1: CPU and watchdog timer are stopped (start IDLE1/2, SLEEP1/2 mode)	
TGHALT	TG control (IDLE0, SLEEP0 mode)	0: Feeding clock to all peripherals from TG 1: Stop feeding clock to peripherals except TBT from TG. (Start IDLE0, SLEEP0 mode)	

Note 1: A reset is applied ($\overline{\text{RESET}}$ pin output goes low) if both XEN and XTEN are cleared to “0”, XEN is cleared to “0” when SYSCK = “0”, or XTEN is cleared to “0” when SYSCK = “1”.

Note 2: *: Don’t care, TG: Timing generator

Note 3: Bits 3, 1and 0 in SYSCR2 are always read as undefined value.

Note 4: Do not set IDLE and TGHALT to “1” simultaneously.

Note 5: Because returning from IDLE0/SLEEP0 to NORMAL1/SLOW1 is executed by the asynchronous internal clock, the period of IDLE0/SLEEP0 mode might be shorter than the period setting by TBTCR<TBTCK>.

Note 6: When IDLE1/2 or SLEEP1/2 mode is released, IDLE is automatically cleared to “0”.

Note 7: When IDLE0 or SLEEP0 mode is released, TGHALT is automatically cleared to “0”.

Note 8: Before setting TGHALT to “1”, be sure to stop peripherals. If peripherals are not stopped, the interrupt latch of peripherals may be set after IDLE0 or SLEEP0 mode is released.

Figure 1.4.7 System Control Registers

1.4.4 Operating Mode Control

(1) STOP mode

STOP mode is controlled by the system control register 1, the $\overline{\text{STOP}}$ pin input and key-on wake-up input (STOP2 to STOP5) which is controlled by the STOP mode release control register (STOPCR).

The $\overline{\text{STOP}}$ pin is also used both as a port P20 and an $\overline{\text{INT5}}$ (External interrupt input 5) pin.

STOP mode is started by setting SYSCR1<STOP> to "1". During STOP mode, the following status is maintained.

a. Oscillations are turned off, and all internal operations are halted.
b. The data memory, registers, the program status word and port output latches are all held in the status in effect before STOP mode was entered.
c. The prescaler and the divider of the timing generator are cleared to "0".
d. The program counter holds the address 2 ahead of the instruction (e.g. [SET (SYSCR1).7]) which started STOP mode.

STOP mode includes a level-sensitive mode and an edge-sensitive mode, either of which can be selected with the SYSCR1<RELM>. Do not use any STOPx (x: 2 to 5) pin input for releasing STOP mode in edge-sensitive mode.

When the STOP mode is started with the EEPCR<MNPWDW> = "1", the CPU wait period for stabilizing of the power supply of Flash control circuit is executed after in the STOP warming-up time.

Note 1: The STOP mode can be released by either the STOP or key-on wake-up pin (STOP2 to STOP5). However, because the $\overline{\text{STOP}}$ pin is different from the key-on wake-up and can not inhibit the release input, the $\overline{\text{STOP}}$ pin must be used for releasing STOP mode.

Note 2: During stop period (from start of STOP mode to end of warm-up), due to changes in the external interrupt pin signal, interrupt latches may be set to "1" and interrupts may be accepted immediately after STOP mode is released. Before starting STOP mode, therefore, disable interrupts. Also, before enabling interrupts after STOP mode is released, clear unnecessary interrupt latches.

a. Level-sensitive release mode (RELM = "1")

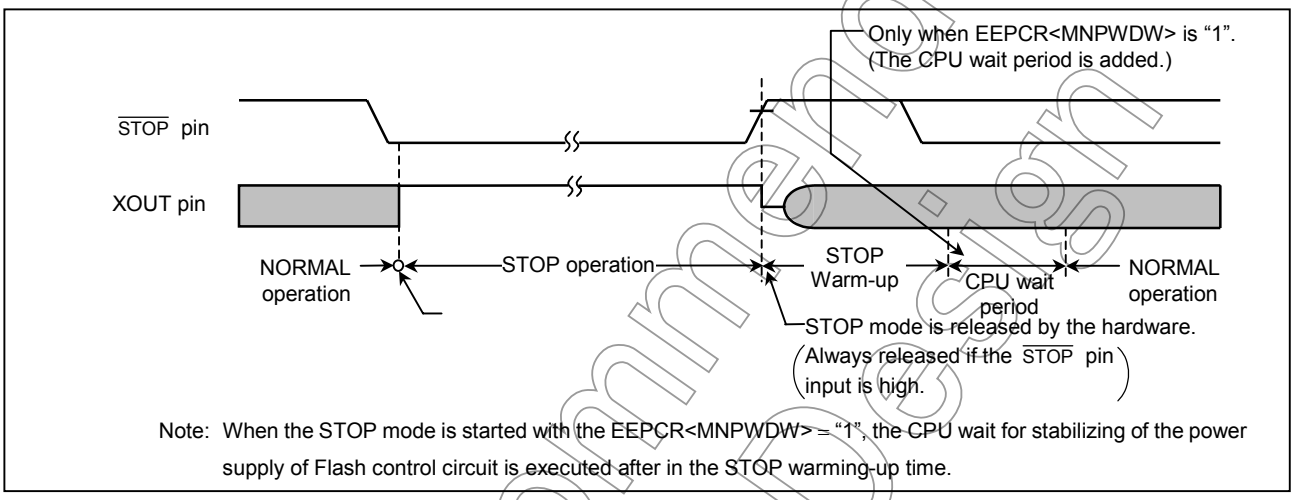
In this mode, STOP mode is released by setting the $\overline{\text{STOP}}$ pin high or setting the STOPx (x: 2 to 5) pin input which is enabled by STOPCR. This mode is used for capacitor back-up when the main power supply is cut off and long term battery back-up.

When the $\overline{\text{STOP}}$ pin input is high, executing an instruction which starts STOP mode will not place in STOP mode but instead will immediately start the release sequence (Warm-up). Thus, to start STOP mode in the level-sensitive release mode, it is necessary for the program to first confirm that the $\overline{\text{STOP}}$ pin input is low. The following two methods can be used for confirmation.

a. Testing a port P20.
b. Using an external interrupt input $\overline{\text{INT5}}$ ($\overline{\text{INT5}}$ is a falling edge-sensitive input).

```
Example 1:  Starting STOP mode from NORMAL mode by testing a port P20.
            LD      (SYSCR1), 01010000B     ;  Sets up the level-sensitive release mode
SSTOPH:     TEST    (P2PRD). 0              ;  Wait until the STOP pin input goes low
                                                level
            JRS     F, SSTOPH
            SET     (SYSCR1).7              ;  Starts STOP mode

Example 2:  Starting STOP mode from NORMAL mode with an INT5 interrupt.
PINT5:      TEST    (P2PRD). 0              ;  To reject noise, STOP mode does not
                                                start if port P20 is at high
            JRS     F, SINT5
            LD      (SYSCR1), 01010000B     ;  Sets up the level-sensitive release mode.
            SET     (SYSCR1). 7             ;  Starts STOP mode
SINT5:      RETI
```



Note: When the STOP mode is started with the EEPCR<MNPWDW> = "1", the CPU wait for stabilizing of the power supply of Flash control circuit is executed after in the STOP warming-up time.

Figure 1.4.8 Level-sensitive Release Mode

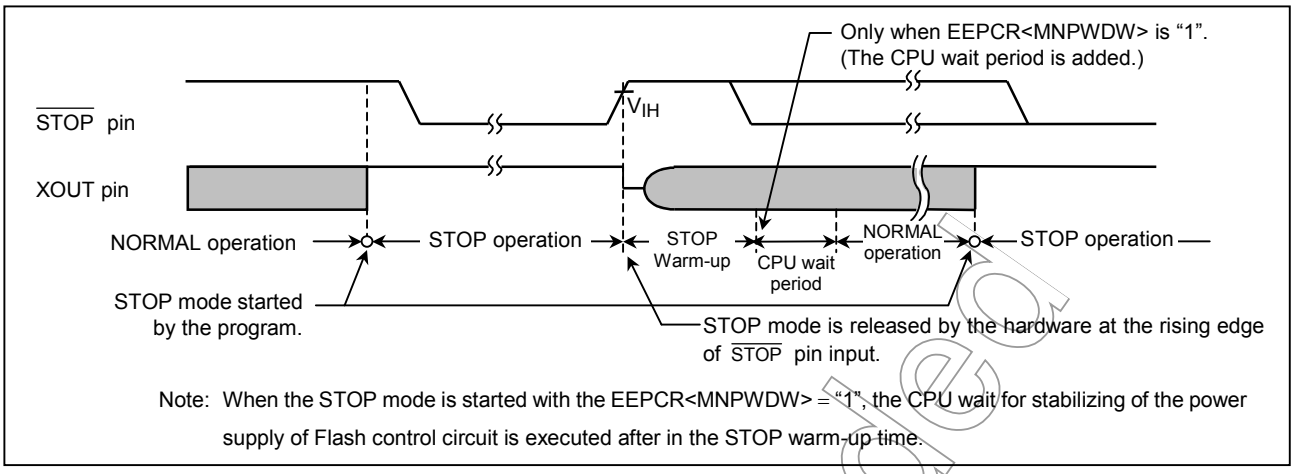
Note 1: Even if the $\overline{\text{STOP}}$ pin input is low after warming up start, the STOP mode is not restarted.

Note 2: In this case of changing to the level-sensitive mode from the edge-sensitive mode, the release mode is not switched until a rising edge of the $\overline{\text{STOP}}$ pin input is detected.

b. Edge-sensitive release mode (RELM = "0")

In this mode, STOP mode is released by a rising edge of the $\overline{\text{STOP}}$ pin input. This is used in applications where a relatively short program is executed repeatedly at periodic intervals. This periodic signal (For example, a clock from a low-power consumption oscillator) is input to the $\overline{\text{STOP}}$ pin. In the edge-sensitive release mode, STOP mode is started even when the $\overline{\text{STOP}}$ pin input is high level. Do not use any STOPx (x: 2 to 5) pin input for releasing STOP mode in edge-sensitive release mode.

```
Example:    Starting STOP mode from NORMAL mode
            LD      (SYSCR1), 10010000B     ;  Starts after specified to the edge-sensitive
                                                release mode
```



Note: When the STOP mode is started with the EEPCR<MNPWDW> = "1", the CPU wait for stabilizing of the power supply of Flash control circuit is executed after in the STOP warm-up time.

Figure 1.4.9 Edge-sensitive Release Mode

STOP mode is released by the following sequence.

a. In the dual-clock mode, when returning to NORMAL2, both the high-frequency and low-frequency clock oscillators are turned on; when returning to SLOW1 mode, only the low-frequency clock oscillator is turned on. In the single-clock mode, only the high-frequency clock oscillator is turned on.
b. A STOP warm-up period is inserted to allow oscillation time to stabilize. During STOP warm-up, all internal operations remain halted. Four different STOP warm-up times can be selected with the SYSCR1<WUT> in accordance with the resonator characteristics.
c. When the EEPCR<MNPWDW> is "1", the CPU wait period is inserted to stabilize the power supply of Flash control circuit. During CPU wait, though CPU operations remain halted, the peripheral function operation is resumed, and the counting of the timing generator is restarted. After the CPU wait is finished, normal operation resumes with the instruction following the STOP mode start instruction.
d. When the EEPCR<MNPWDW> is "0", normal operation resumes with the instruction following the STOP mode start instruction after the STOP Warm-up.

Note 1: When the STOP mode is released, the start is made after the prescaler and the divider of the timing generator are cleared to "0".

Note 2: STOP mode can also be released by inputting low level on the $\overline{\text{RESET}}$ pin, which immediately performs the normal reset operation.

Note 3: When STOP mode is released with a low hold voltage, the following cautions must be observed.
The power supply voltage must be at the operating voltage level before releasing STOP mode. The $\overline{\text{RESET}}$ pin input must also be "H" level, rising together with the power supply voltage. In this case, if an external time constant circuit has been connected, the $\overline{\text{RESET}}$ pin input voltage will increase at a slower pace than the power supply voltage. At this time, there is a danger that a reset may occur if input voltage level of the $\overline{\text{RESET}}$ pin drops below the non-inverting high-level input voltage (Hysteresis input).

Table 1.4.1 Warm-up Time Example (at fc = 16.0 MHz, fs = 32.768 kHz)

WUT	Warm-up Time [ms] (Note 2)	
	Return to NORMAL Mode	Return to SLOW Mode
00	12.288 + (0.064)	750 + (0.244)
01	4.096 + (0.064)	250 + (0.244)
10	3.072 + (0.064)	5.85 + (0.244)
11	1.024 + (0.064)	1.95 + (0.244)

Note 1: The warm-up time is obtained by dividing the basic clock by the divider: therefore, the warm-up time may include a certain amount of error if there is any fluctuation of the oscillation frequency when STOP mode is released. Thus, the warm-up time must be considered an approximate value.

Note 2: The CPU wait period for FLASH is shown in parentheses.

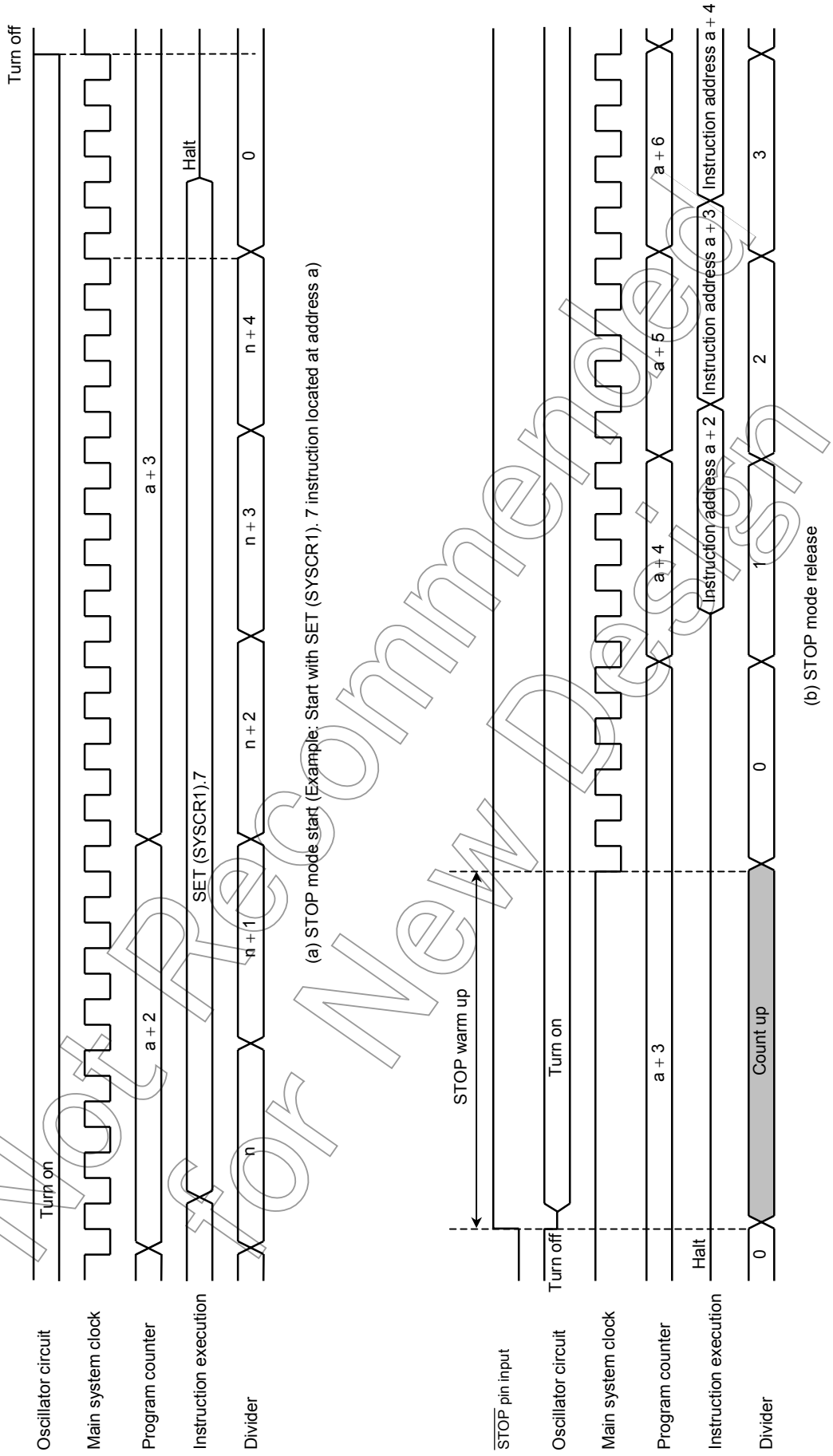


Figure 1.4.10 STOP Mode Start/Release (When EEPCR<MNPWDW> = “0”)

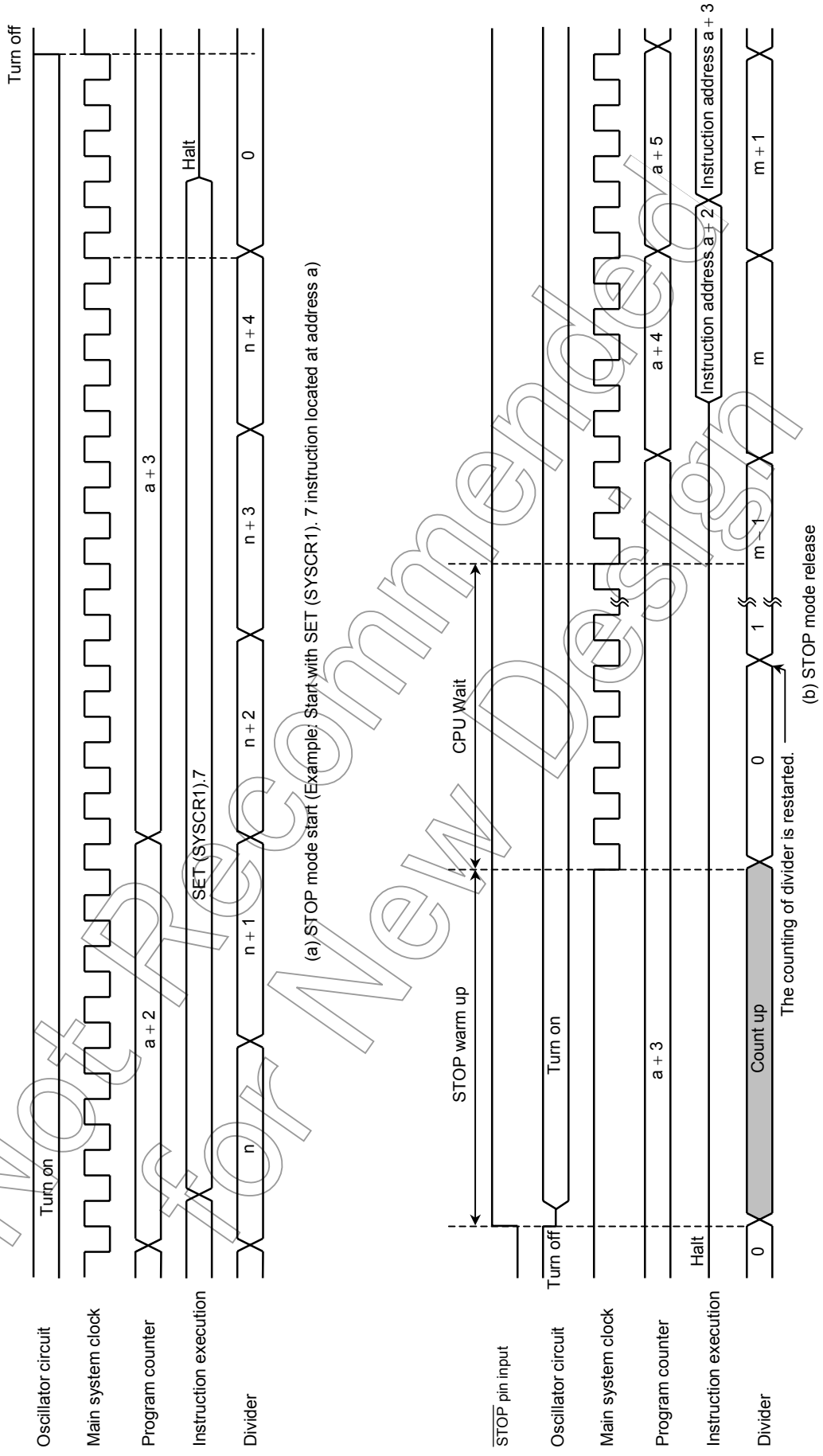


Figure 1.4.11 STOP Mode Start/Release (When EEPCR<MNPWDW> = “1”)

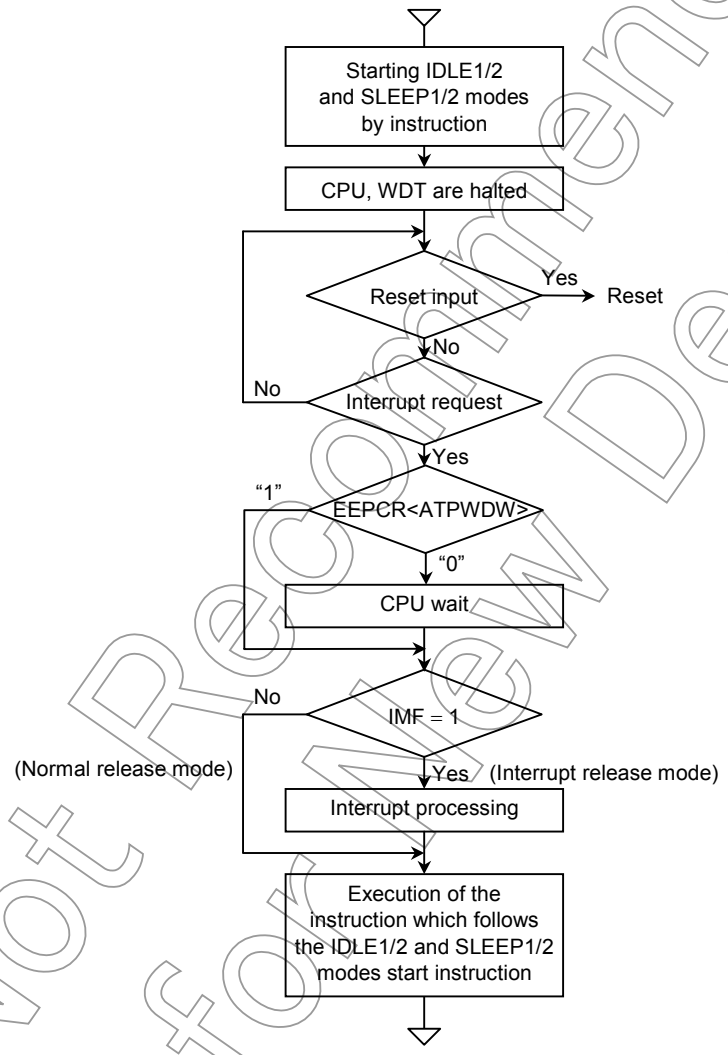
(2) IDLE1/2 mode, SLEEP1/2 mode

IDLE1/2 and SLEEP1/2 modes are controlled by the system control register 2 (SYSCR2) and maskable interrupts. The following status is maintained during these modes.

a. Operation of the CPU and watchdog timer (WDT) is halted. On-chip peripherals continue to operate.

b. The data memory, CPU registers, program status word and port output latches are all held in the status in effect before these modes were entered.

c. The program counter holds the address 2 ahead of the instruction which starts these modes.



Note 1: EEPCR<ATPWDW> is a bit1 in EEPCR, which is a control bit of the power supply circuit for flash.

Note 2: During CPU wait, though CPU operations remain halted, the peripheral function operation is resumed. Therefore in this time, though the interrupt latch might be set, interrupt operation is not executed until the CPU wait is finished.

Figure 1.4.12 IDLE1/2, SLEEP1/2 Modes

- Start the IDLE1/2 and SLEEP1/2 modes

  When IDLE1/2 and SLEEP1/2 modes start, set SYSCR2<IDLE> to "1".

- Release the IDLE1/2 and SLEEP1/2 modes

  IDLE1/2 and SLEEP1/2 modes include a normal release mode and an interrupt release mode. These modes are selected by interrupt master enable flag (IMF).

  After releasing IDLE1/2 and SLEEP1/2 modes, the SYSCR2<IDLE> is automatically cleared to "0" and the operation mode is returned to the mode preceding IDLE1/2 and SLEEP1/2 modes.

  When the IDLE1/2 and SLEEP1/2 modes are started with the EEPCR<ATPWDW> = "0", the CPU wait period for stabilizing of the power supply of Flash control circuit is added before the operation mode is returned to the preceding modes. The CPU wait time of IDLE1/2 is 2^{10}/fc [s] and that of SLEEP1/2 mode is 2^3/fs [s].

  IDLE1/2 and SLEEP1/2 modes can also be released by inputting low level on the $\overline{\text{RESET}}$ pin. After releasing reset, the operation mode is started from NORMAL1 mode.

  Note: During CPU wait, though CPU operations remain halted, but the peripheral function operation is resumed. Therefore in this time, though the interrupt latch might be set, interrupt operation is not executed until the CPU wait is finished.

(a) Normal release mode (IMF = "0")

IDLE1/2 and SLEEP1/2 modes are released by any interrupt source enabled by the individual interrupt enable flag (EF). After the interrupt is generated, the program operation is resumed from the instruction following the IDLE1/2 and SLEEP1/2 modes start instruction. Normally, the interrupt latches (IL) of the interrupt source used for releasing must be cleared to "0" by load instructions.

(b) Interrupt release mode (IMF = "1")

IDLE1/2 and SLEEP1/2 modes are released by any interrupt source enabled with the individual interrupt enable flag (EF). After the interrupt is processed, the program operation is resumed from the instruction following the instruction, which starts IDLE1/2 and SLEEP1/2 modes.

Note: When a watchdog timer interrupts is generated immediately before IDLE1/2 and SLEEP1/2 mode are started, the watchdog timer interrupt will be processed but IDLE1/2 and SLEEP1/2 mode will not be started.

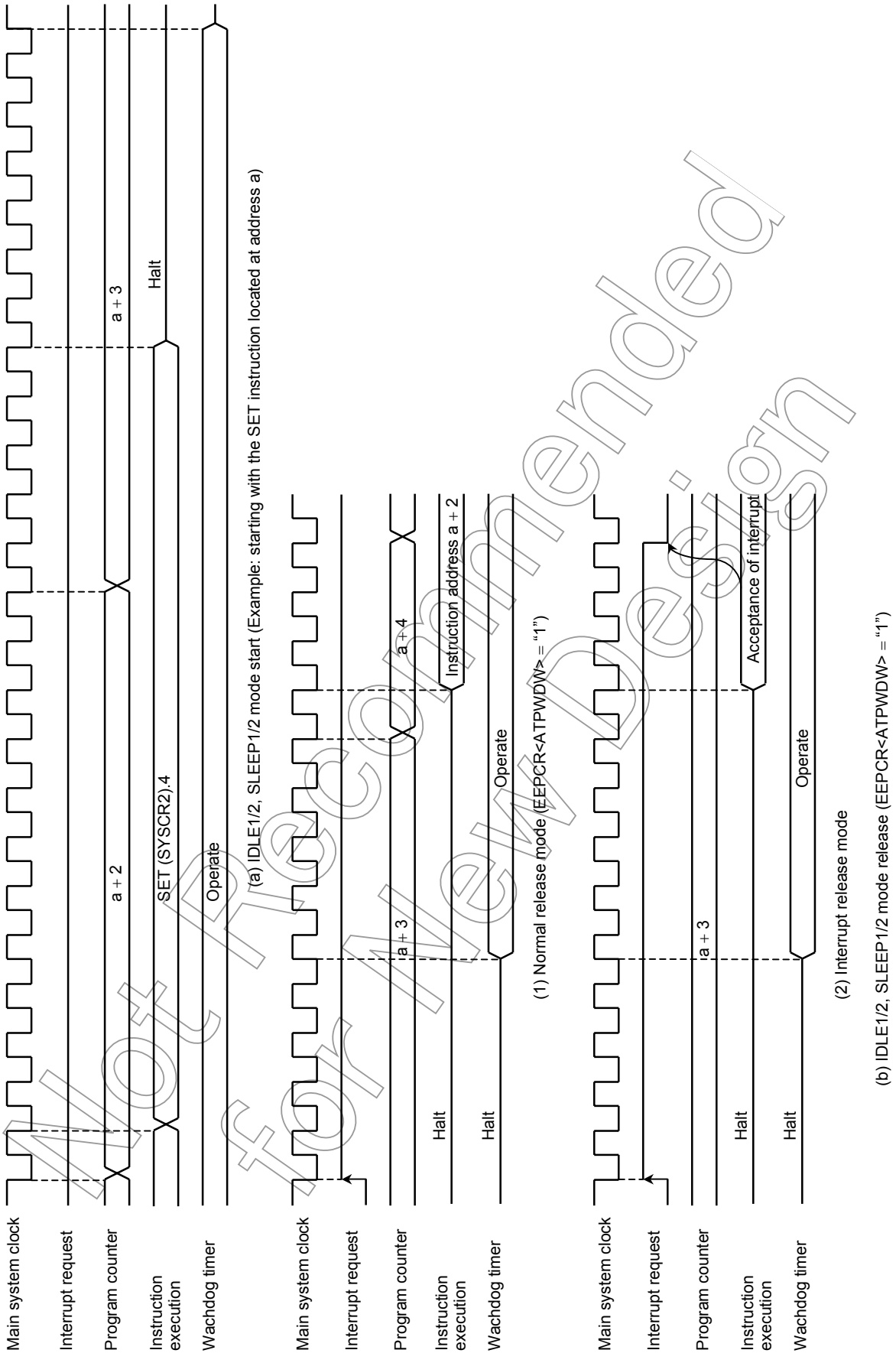


Figure 1.4.13 IDLE1/2, SLEEP1/2 Mode Start/Release

(3) IDLE0, SLEEP0 mode (IDLE0, SLEEP0)

IDLE0 and SLEEP0 modes are controlled by the system control register 2 (SYSCR2) and the time base timer control register (TBTCR). The following status is maintained during IDLE0 and SLEEP0 modes.

a. Timing generator stops feeding clock to peripherals except TBT.
b. The data memory, CPU registers, program status word and port output latches are all held in the status in effect before IDLE0 and SLEEP0 modes were entered.
c. The program counter holds the address 2 ahead of the instruction which starts IDLE0 and SLEEP0 modes.

Note: Before starting IDLE0 or SLEEP0 mode, be sure to stop (Disable) peripherals.

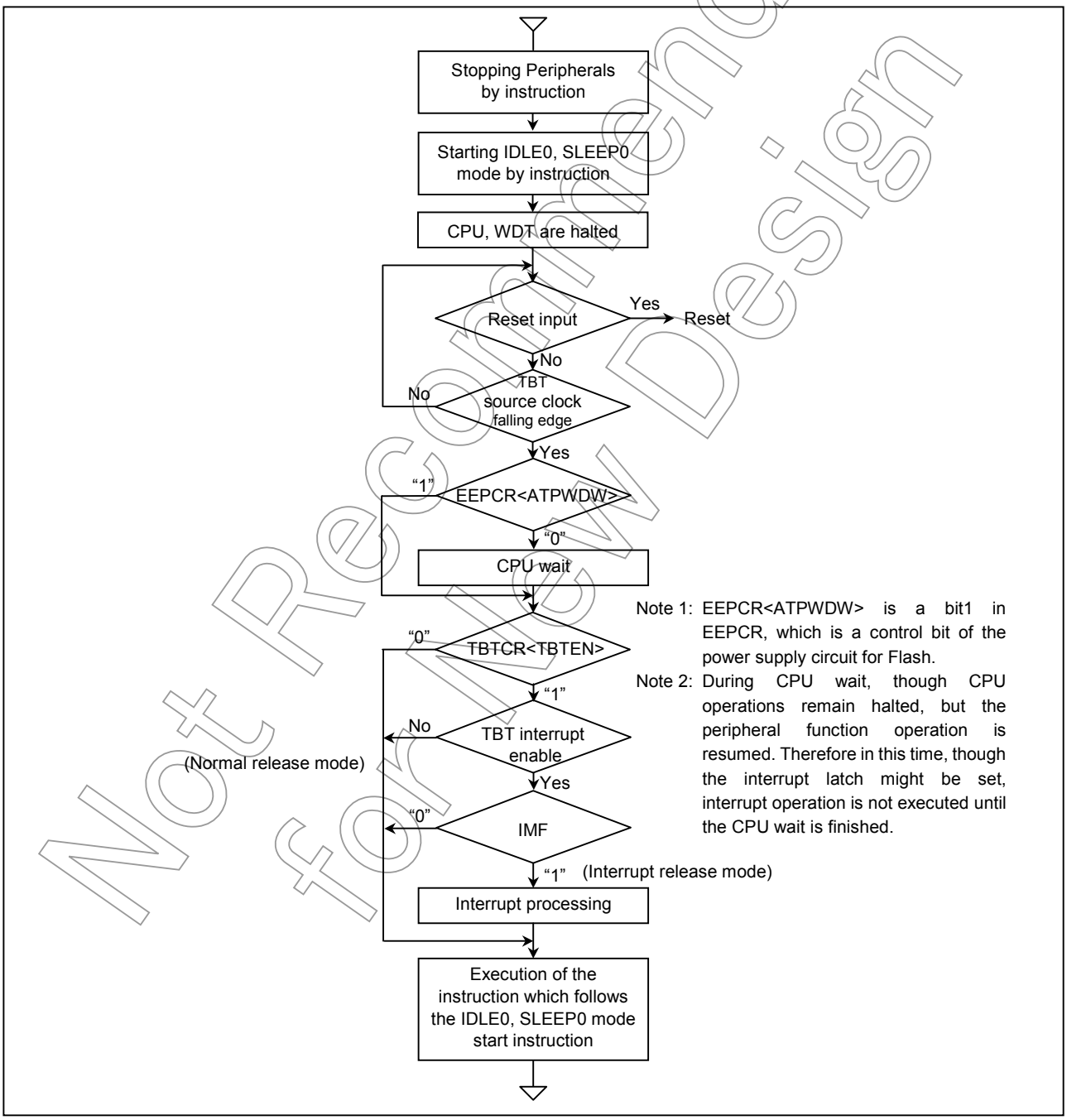


Figure 1.4.14 IDLE0, SLEEP0 Mode

- Start the IDLE0 and SLEEP0 modes

  Stop (Disable) peripherals such as a timer counter.

  When IDLE0 and SLEEP0 modes start, set SYSCR2<TGHALT> to "1".

- Release the IDLE0 and SLEEP modes

  IDLE0 and SLEEP0 modes include a normal release mode and an interrupt release mode.

  These modes are selected by interrupt master flag (IMF), individual interrupt enable-flag (EF6) for INTTBT and TBTCR<TBTEN>.

  After releasing IDLE0 and SLEEP0 modes, the SYSCR2<TGHALT> is automatically cleared to "0" and the operation mode is returned to the mode preceding IDLE0 and SLEEP0 modes. Before starting the IDLE0 or SLEEP0 mode, when the TBTCR<TBTEN> is set to "1", INTTBT interrupt latch is set to "1".

  When the IDLE0 and SLEEP0 modes are started with the EEPCR<ATPWDW> = "0", the CPU wait period for stabilizing of the power supply of Flash control circuit is added before the operation mode is returned to the preceding modes. The CPU wait time of IDLE0 is 2^{10}/fc [s] and that of SLEEP0 mode is 2^3/fs [s].

  IDLE0 and SLEEP0 modes can also be released by inputting low level on the $\overline{\text{RESET}}$ pin. After releasing reset, the operation mode is started from NORMAL1 mode.

  Note 1: IDLE0 and SLEEP0 modes start/release without reference to TBTCR<TBTEN> setting.

  Note 2: During CPU wait, though CPU operations remain halted, but the peripheral function operation is resumed. Therefore in this time, though the interrupt latch might be set, interrupt operation is not executed until the CPU wait is finished.

a. Normal release mode (IMF・EF6・TBTCR<TBTEN> = "0")

IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK>. After the falling edge is detected, the program operation is resumed from the instruction following the IDLE0 and SLEEP0 modes start instruction.

b. Interrupt release mode (IMF・EF6・TBTCR<TBTEN> = "1")

IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK> and INTTBT interrupt processing is started.

Note 1: Because returning from IDLE0, SLEEP0 to NORMAL1, SLOW1 is executed by the asynchronous internal clock, the period of IDLE0, SLEEP0 mode might be the shorter than the period setting by TBTCR<TBTCK>.

Note 2: When a watchdog timer interrupt is generated immediately before IDLE0/SLEEP0 mode is started, the watchdog timer interrupt will be processed but IDLE0/SLEEP0 mode will not be started.

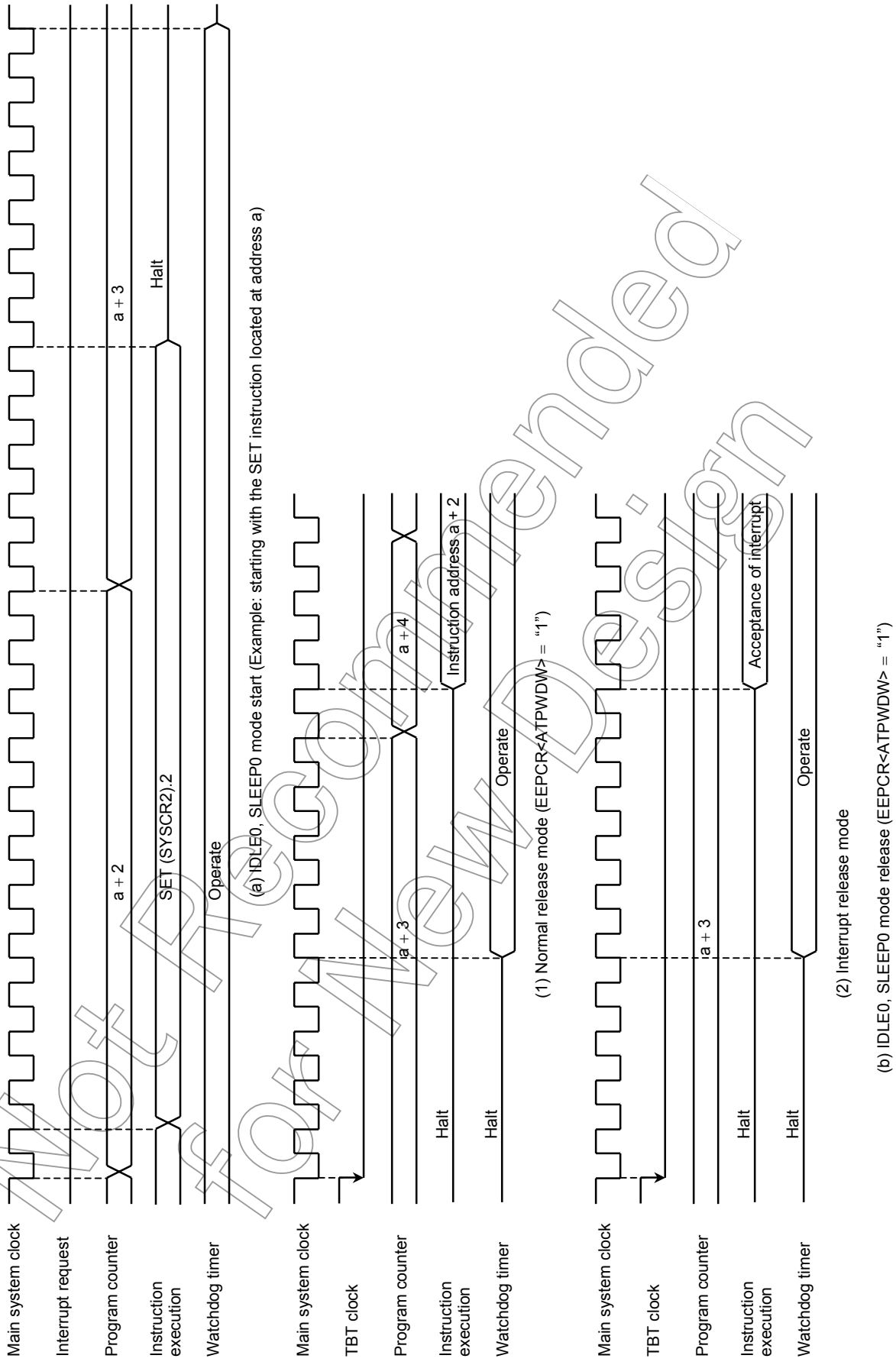


Figure 1.4.15 IDLE0, SLEEP0 Mode Start/Release

(4) SLOW mode

SLOW mode is controlled by the system control register 2 (SYSCR2).

The following is the methods to switch the mode with the warm-up counter (TC2).

a. Switching from NORMAL2 mode to SLOW1 mode

First, set SYSCR2<SYSCK> to switch the main system clock to the low-frequency clock for SLOW2 mode.

Next, clear SYSCR2<XEN> to turn off high-frequency oscillation.

Note: The high-frequency clock oscillation can be continued to return quickly to NORMAL2 mode. But starting STOP mode while SLOW mode, the high-frequency oscillation must be stopped.

When the low-frequency clock oscillation is unstable, wait until oscillation stabilizes before performing the above operations. The timer/counter 2 (TC2) can conveniently be used to confirm that low-frequency clock oscillation has stabilized.

Example 1: Switching from NORMAL2 mode to SLOW1 mode.

```
        SET     (SYSCR2). 5         ;  SYSCR2<SYSCK> ← 1
                                       (Switches the main system clock to the
                                       low-frequency clock for SLOW2)
        CLR     (SYSCR2). 7         ;  SYSCR2<XEN> ← 0
                                       (Turns off high-frequency oscillation)
```

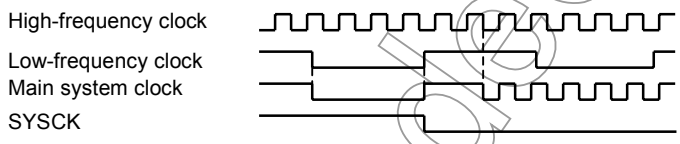
Example2: Switching to the SLOW1 mode after low-frequency clock has stabilized.

```
        SET     (SYSCR2). 6         ;  SYSCR2<XTEN> ← 1
        LD      (TC2CR), 14H        ;  Sets mode for TC2
        LDW     (TC2DRL), 8000H     ;  Sets warm-up time
                                       (Depend on oscillator accompanied)
        DI                          ;  IMF ← 0
        SET     (EIRE). 4           ;  Enables INTTC2
        EI                          ;  IMF ← 1
        SET     (TC2CR). 5          ;  Starts TC2
          :
PINTTC2: CLR    (TC2CR). 5          ;  Stops TC2
        SET     (SYSCR2). 5         ;  SYSCR2<SYSCK> ← 1
                                       (Switches the main system clock to the
                                       low-frequency clock)
        CLR     (SYSCR2). 7         ;  SYSCR2<XEN> ← 0
                                       (Turns off high-frequency oscillation)
        RETI
          :
VINTTC2: DW     PINTTC2             ;  INTTC2 vector table
```

b. Switching from SLOW1 mode to NORMAL2 mode

First, set SYSCR2<XEN> to turn on the high-frequency oscillation. When time for stabilization (Warm-up) has been taken by the timer/counter 2 (TC2), clear SYSCR2<SYSCK> to switch the main system clock to the high-frequency clock.

Note 1: After SYSCK is cleared to "0", executing the instructions is continued by the low-frequency clock for the period synchronized with low-frequency and high-frequency clocks.



Note 2: SLOW mode can also be released by inputting low level on the $\overline{\text{RESET}}$ pin, which immediately performs the reset operation. After reset, the TMP86FM29 is placed in NORMAL1 mode.

Example: Switching from the SLOW1 mode to the NORMAL2 mode
(fc = 16 MHz, warm-up time is = 4.0 ms).

```
        SET     (SYSCR2). 7         ;  SYSCR2<XEN> ← 1
                                       (Starts high-frequency oscillation)
        LD      (TC2CR), 10H        ;  Sets mode for TC2
                                       (Timer mode, fc for source)
        LD      (TC2DRH), 0F8H      ;  Sets warm-up time
                                       (Depend on oscillator accompanied)
        DI                          ;  IMF ← 0
        SET     (EIRE). 4           ;  Enables INTTC2
        EI                          ;  IMF ← 1
        SET     (TC2CR). 5          ;  Starts TC2
          :
PINTTC2: CLR    (TC2CR). 5          ;  Stops TC2
        CLR     (SYSCR2). 5         ;  SYSCR2<SYSCK> ← 0
                                       (Switches the main system clock to the
                                       high-frequency clock)
        RETI
          :
VINTTC2: DW     PINTTC2             ;  INTTC2 vector table
```

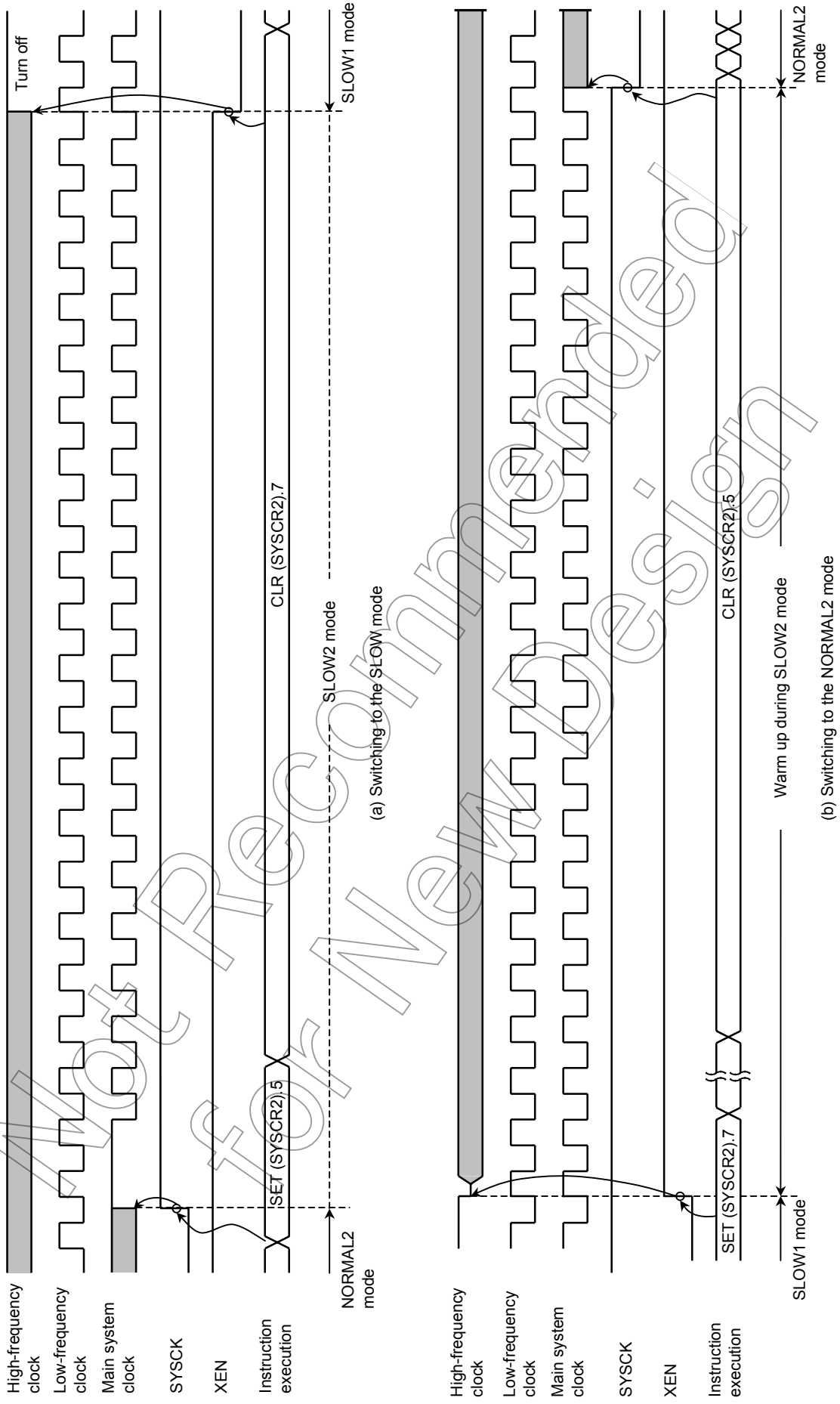


Figure 1.4.16 Switching between the NORMAL2 and SLOW Modes

1.5 Interrupt Control Circuit

The TMP86FM29 has a total (Reset is excluded) of 15 interrupt sources for 19 interrupt factors; 3 of the sources are multiplexed. Multiple interrupt with priorities is available. 4 of the internal factors are non-maskable interrupts, and the rest of them are maskable interrupts.

Interrupt sources are provided with interrupt latches (IL), which hold interrupt requests, and independent vectors. The interrupt latch is set to "1" by the generation of its interrupt request which requests the CPU to accept its interrupts. Interrupts are enabled or disabled by software using the interrupt master enable flag (IMF) and interrupt enable flag (EF). If more than one interrupts are generated simultaneously, interrupts are accepted in order which is dominated by hardware. However, there are no prioritized interrupt factors among non-maskable interrupts.

Table 1.5.1 Interrupt Sources

Interrupt Factors			Enable Condition	Interrupt Latch	Vector Address	Priority
Internal/External	(Reset)		Non-maskable	–	$FFFE_H$	High 1
Internal	INTSWI	(Software interrupt)	Non-maskable	–	$FFFC_H$	2
Internal	INTUNDEF	(Executed the Undefined Instruction interrupt)	Non-maskable	–	$FFFC_H$	2
Internal	INTATRAP	(Address Trap interrupt)	Non-maskable	IL_2	$FFFA_H$	2
Internal	INTWDT	(Watchdog Timer interrupt)	Non-maskable	IL_3	$FFF8_H$	2
External	$\overline{INT0}$	(External interrupt 0)	IMF· EF_4 = 1	IL_4	$FFF6_H$	5
External	INT1	(External interrupt 1)	IMF· EF_5 = 1	IL_5	$FFF4_H$	6
Internal	INTTBT	(Time Base Timer interrupt)	IMF· EF_6 = 1	IL_6	$FFF2_H$	7
External	INT2	(External interrupt2)	IMF· EF_7 = 1	IL_7	$FFF0_H$	8
Internal	INTTC1	(18-bit TC1 interrupt)	IMF· EF_8 = 1	IL_8	$FFEE_H$	9
Internal	INTRxD	(UART received interrupt)	IMF· EF_9 = 1	IL_9	$FFEC_H$	10
Internal	INTSIO	(SIO interrupt)				
Internal	INTTxD	(UART transmitted interrupt)	IMF· EF_{10} = 1	IL_{10}	$FFEA_H$	11
Internal	INTTC4	(TC4 interrupt)	IMF· EF_{11} = 1	IL_{11}	$FFE8_H$	12
Internal	INTTC6	(TC6 interrupt)	IMF· EF_{12} = 1	IL_{12}	$FFE6_H$	13
Internal	INTADC	(AD converter interrupt)	IMF· EF_{13} = 1	IL_{13}	$FFE4_H$	14
External	INT3	(External interrupt 3)	IMF· EF_{14} = 1	IL_{14}	$FFE2_H$	15
Internal	INTTC3	(TC3 interrupt)				
External	$\overline{INT5}$	(External interrupt 5)	IMF· EF_{15} = 1	IL_{15}	$FFE0_H$	Low 16
Internal	INTTC5	(TC5 interrupt)				

Note 1: The following interrupt factors share their interrupt source; the factor is selected on the register INTSEL.

1. INTRXD and INTSIO share the source whose priority is 10.
2. INT3 and INTTC3 share the source whose priority is 15.
3. $\overline{INT5}$ and INTTC5 share the source whose priority is 16.

Note 2: To use the address trap interrupt (INTATRAP), clear WDTCR1<ATOUT> to "0" (It is set for the "Reset request" after reset is released). For details, see 2.4.5 Address Trap.

Note 3: To use the watchdog timer interrupt (INTWDT), clear WDTCR1<WDTOUT> to "0" (It is set for the "Reset request" after reset is released). For details, see 2.4 Watchdog Timer.

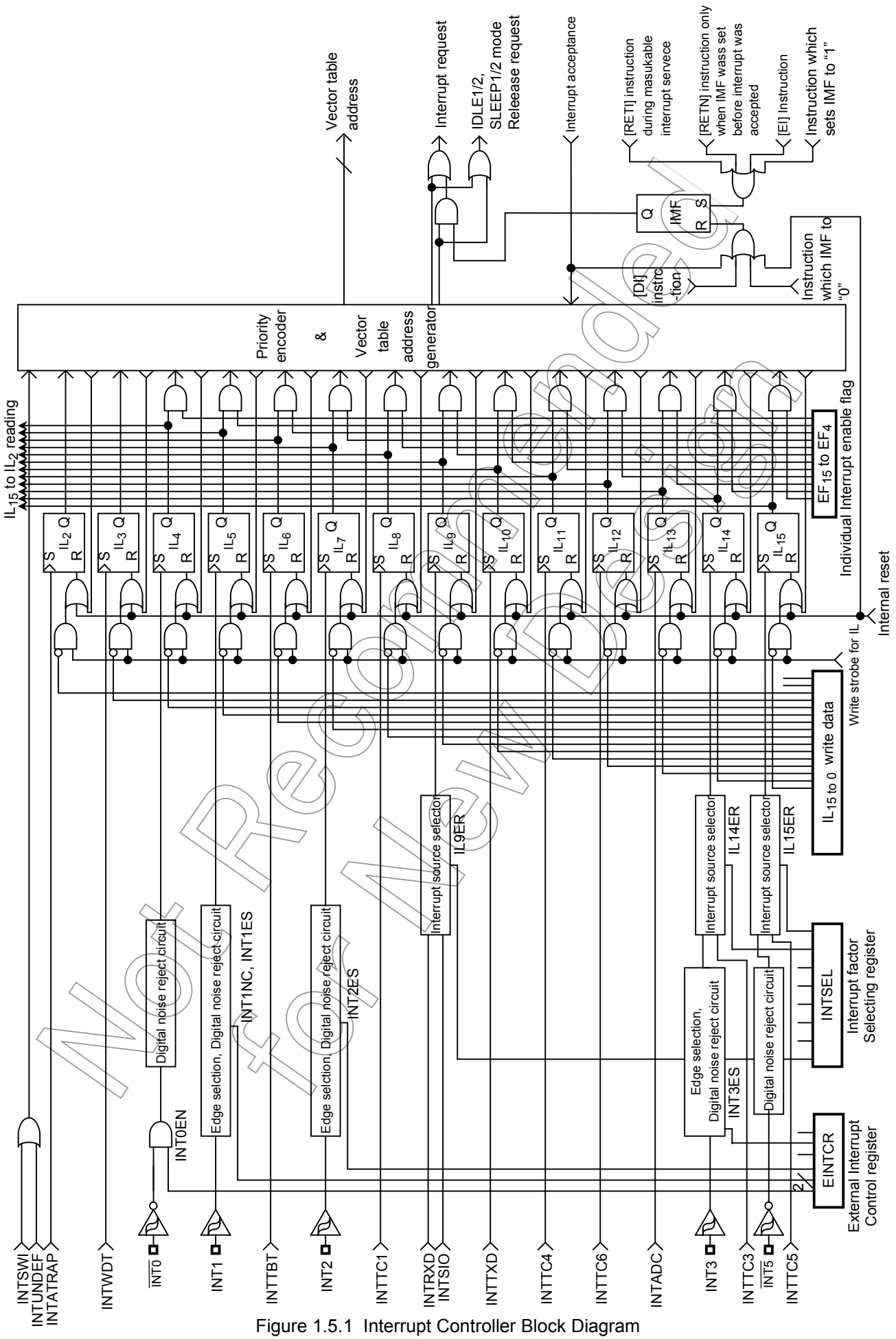


Figure 1.5.1 Interrupt Controller Block Diagram

(1) Interrupt latches (IL_{15} to IL_2)

An interrupt latch is provided for each interrupt source, except for a software interrupt. When interrupt request is generated, the latch is set to "1", and the CPU is requested to accept the interrupt if its interrupt is enabled. All interrupt latches are initialized to "0" during reset.

The interrupt latches are located on address $003C_H$ and $003D_H$ in SFR area. Except for IL_3 and IL_2, each latch can be cleared to "0" individually by instruction. (However, the read-modify-write instructions such as bit manipulation or operation instructions cannot be used. Interrupt request would be cleared inadequately if interrupt is requested while such instructions are executed.) Thus interrupt request can be canceled/initialized by software.

Interrupt latches are not set to "1" by an instruction. Since interrupt latches can be read, the status for interrupt requests can be monitored by software.

Note: When manipulating IL, clear IMF (to disable interrupts) beforehand.

```
Example 1: Clears interrupt latches
        DI                              ;   IMF ← 0
        LDW     (ILL), 1110100000111111B ;  IL12, IL10 to IL6 ← 0
        EI                              ;   IMF ← 1
Example 2: Reads interrupt latches
        LD      WA, (ILL)               ;   W ← ILH, A ← ILL
Example 3: Tests an interrupt latches
        TEST    (IL).7                  ;   IL7 = 1 then jump
        JR      F, SSET
```

(2) Interrupt enable register (EIR)

The interrupt enable register (EIR) enables and disables the acceptance of interrupts, except for the non-maskable interrupts (Software interrupt, undefined instruction interrupt, address trap interrupt and watchdog interrupt). Non-maskable interrupt is accepted regardless of the contents of the EIR.

The EIR consists of an interrupt master enable flag (IMF) and the individual interrupt enable flags (EF). These registers are located on address $003A_H$ and $003B_H$ in SFR area, and they can be read and written by an instructions (Including read-modify-write instructions such as bit manipulation or operation instructions).

a. Interrupt master enable flag (IMF)

The interrupt enable register (IMF) enables and disables the acceptance of the whole maskable-interrupt. While IMF = "0", all maskable interrupts are not accepted regardless of the status on each individual interrupt enable flag (EF). By setting IMF to "1", the interrupt becomes acceptable if the individuals are enabled. When an interrupt is accepted, IMF is cleared to "0" after the latest status on IMF is stacked. Thus the maskable interrupts which follow are disabled. By executing return interrupt instruction [RETI/RETN], the stacked data, which was the status before interrupt acceptance, is loaded on IMF again.

The IMF is located on bit0 in EIRL (Address: $003A_H$ in SFR), and can be read and written by an instruction. The IMF is normally set and cleared by [EI] and [DI] instruction respectively. During reset, the IMF is initialized to "0", and maskable interrupts are not accepted until it is set to "1".

b. Individual interrupt enable flags (EF_{15} to EF_4)

Each of these flags enables and disables the acceptance of its maskable interrupt. Setting the corresponding bit of an individual interrupt enable flag to "1" enables acceptance of its interrupt, and setting the bit to "0" disables acceptance. The individual interrupt enable flags (EF_{23} to EF_4) are located on EIRE, EIRL to EIRH (address: $002C_H$, $003A_H$ to $003B_H$ in SFR), and can be read and written by an instruction. During reset, all the individual interrupt enable flags (EF_{23} to EF_4) are initialized to "0" and all maskable interrupts are not accepted until they are set to "1".

Note: Before manipulating EF, be sure to clear IMF (Interrupt disabled). Then set IMF newly again after operating on the interrupt enables flag (EF). Normally, IMF is clear to "0" automatically on service routine. When IMF is set to "1" for using a multiple interrupt on service routine, be sure to process as is the case with EF.

Example 1: Enables interrupts individually and sets IMF

```
DI                          ;  IMF ← 0
LDW     (EIRL),             ;  EF14, EF13, EF11, EF7, EF5 ← "1"
  :     0110100010100000B      Note: IMF is not set.
  :
EI                          ;  IMF ← "1"
```

Example 2: C compiler description example

```
unsigned  int  _io (3AH)  EIRL;     ;  /* 3AH shows EIRL address */
_DI ( );
EIRL = 10100000B;
  :
  :
_EI ( );
```

Interrupt Latches

ILH, ILL ($003C_H$, $003D_H$)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IL_{15}	IL_{14}	IL_{13}	IL_{12}	IL_{11}	IL_{10}	IL_9	IL_8	IL_7	IL_6	IL_5	IL_4	IL_3	IL_2		
IL_H ($003D_H$)								IL_L ($003C_H$)							

(Initial value: 00000000 000000**)

		at RD	at WR	
IL_{23} to IL_2	Interrupt Latches	0: No interrupt request 1: Interrupt request	Clears the interrupt request (Note 1) (Interrupt Latch is not set.)	R/W

Note 1: IL_2 and IL_3 are prohibited from clearing.

Note 2: When manipulating IL, clear IMF (to disable interrupts) beforehand.

Note 3: Do not clear IL with read-modify-write instructions such as bit operations.

Interrupt Enable Registers

EIRH, EIRL ($003A_H$, $003B_H$)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EF_{15}	EF_{14}	EF_{13}	EF_{12}	EF_{11}	EF_{10}	EF_9	EF_8	EF_7	EF_6	EF_5	EF_4				IMF
EIR_H ($003B_H$)								EIR_L ($003A_H$)							

(Initial value: 00000000 00000***0)

EF_{23} to EF_4	Individual-interrupt enable flag (specified for each bit)	0: Disable the acceptance of each maskable interrupt. 1: Enable the acceptance of each maskable interrupt.	R/W
IMF	Interrupt master enable flag	0: Disable the acceptance of all maskable interrupts. 1: Enable the acceptance of all maskable interrupts.	

Note 1: *: Don't care

Note 2: When manipulating EF, clear IMF (to disable interrupts) beforehand.

Note 3: Do not set IMF to 1 simultaneously with EF15 to EF4.

Figure 1.5.2 Interrupt Latch (IL), Interrupt Enable Registers (EIR)

(3) Selecting interrupt factor (INTSEL)

Each interrupt factor, that shares its interrupt source with other factors, enables its interrupt latch (IL) only if it is selected on INTSEL. The interrupt controller does not hold the interrupt request, while the factor generates the interrupt request is not selected on INTSEL. Therefore, set INTSEL appropriately before interrupt factors arises.

Interrupt source selector

INTSEL ($003E_H$)

7	6	5	4	3	2	1	0
–	IL9ER	–	–	–	–	IL14ER	IL15ER

(Initial value: *0** **00)

IL9ER	Alternative of INTRXD or INTSIO	0: INTRXD 1: INTSIO	R/W
IL14ER	Alternative of INT3 or INTTC3	0: INT3 1: INTTC3	
IL15ER	Alternative of $\overline{INT5}$ or INTTC5	0: $\overline{INT5}$ 1: INTTC5	

Figure 1.5.3 Interrupt source selector

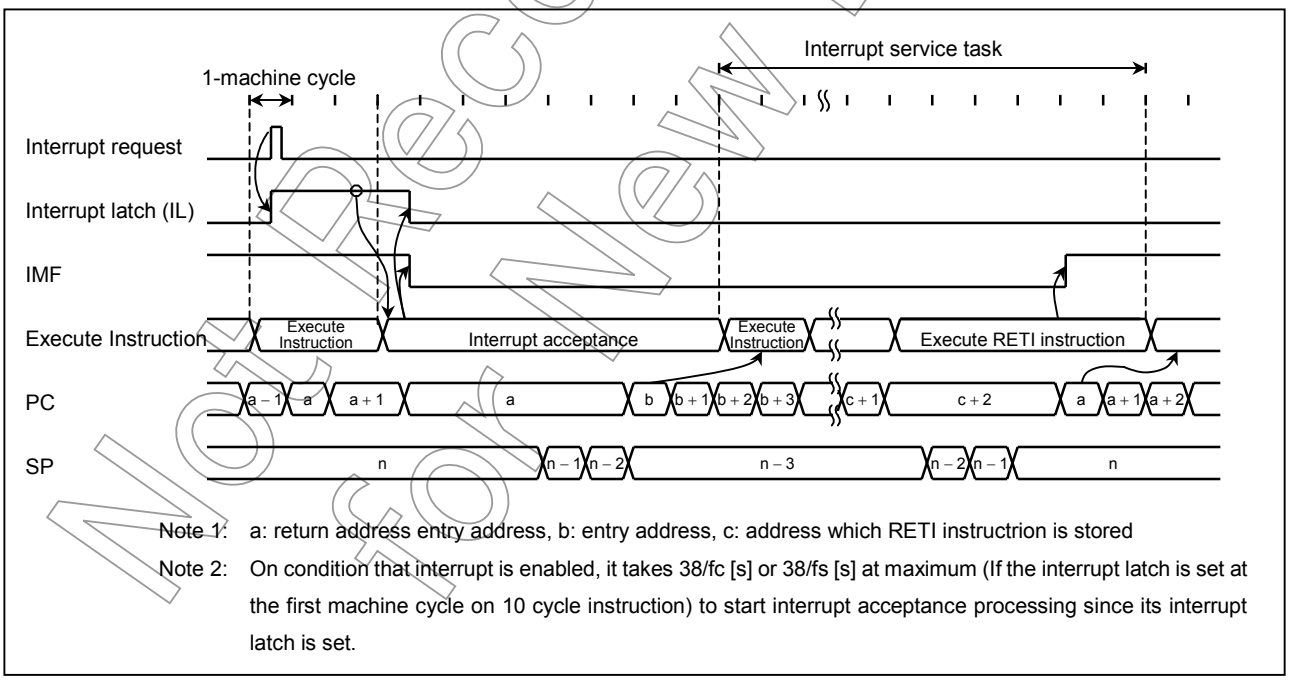
1.5.1 Interrupt Sequence

An interrupt request, which raised interrupt latch, is held, until interrupt is accepted or interrupt latch is cleared to "0" by resetting or an instruction. Interrupt acceptance sequence requires 8-machine cycles (4 μs at 8.0 MHz) after the completion of the current instruction. The interrupt service task terminates upon execution of an interrupt return instruction [RETI] (for maskable interrupts) or [RETN] (for non-maskable interrupts). Figure 1.5.4 shows the timing chart of interrupt acceptance processing.

(1) Interrupt acceptance processing is packaged as follows.

1. The interrupt master enable flag (IMF) is cleared to "0" in order to disable the acceptance of any following interrupt.
2. The interrupt latch (IL) for the interrupt source accepted is cleared to "0".
3. The contents of the program counter (PC) and the program status word, including the interrupt master enable flag (IMF), are saved (Pushed) on the stack in sequence of PSW + IMF, PCH, PCL. Meanwhile, the stack pointer (SP) is decremented by 3.
4. The entry address (Interrupt vector) of the corresponding interrupt service program, loaded on the vector table, is transferred to the program counter.
5. The instruction stored at the entry address of the interrupt service program is executed.

Note: When the contents of PSW are saved on the stack, the contents of IMF are also saved.

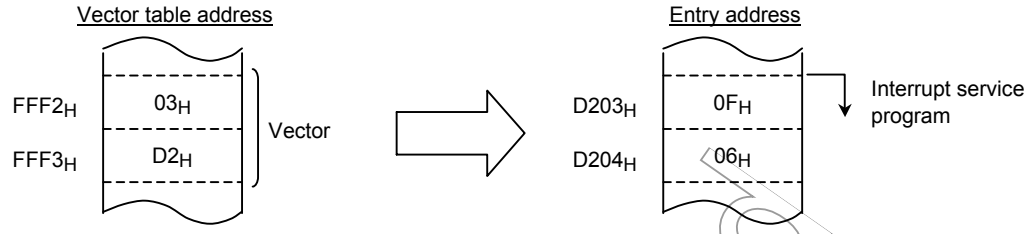


Note 1: a: return address entry address, b: entry address, c: address which RETI instructrion is stored

Note 2: On condition that interrupt is enabled, it takes 38/fc [s] or 38/fs [s] at maximum (If the interrupt latch is set at the first machine cycle on 10 cycle instruction) to start interrupt acceptance processing since its interrupt latch is set.

Figure 1.5.4 Timing Chart of Interrupt Acceptance/Return Interrupt Instruction

Example: Correspondence between vector table address for INTTBT and the entry address of the interrupt service program



A maskable interrupt is not accepted until the IMF is set to "1" even if the maskable interrupt higher than the level of current servicing interrupt is requested.

In order to utilize nested interrupt service, the IMF is set to "1" in the interrupt service program. In this case, acceptable interrupt sources are selectively enabled by the individual interrupt enable flags.

To avoid overloaded nesting, clear the individual interrupt enable flag whose interrupt is currently serviced, before setting IMF to "1". As for non-maskable interrupt, keep interrupt service shorter compared with length between interrupt requests; otherwise the status cannot be recovered as non-maskable interrupt would simply nested.

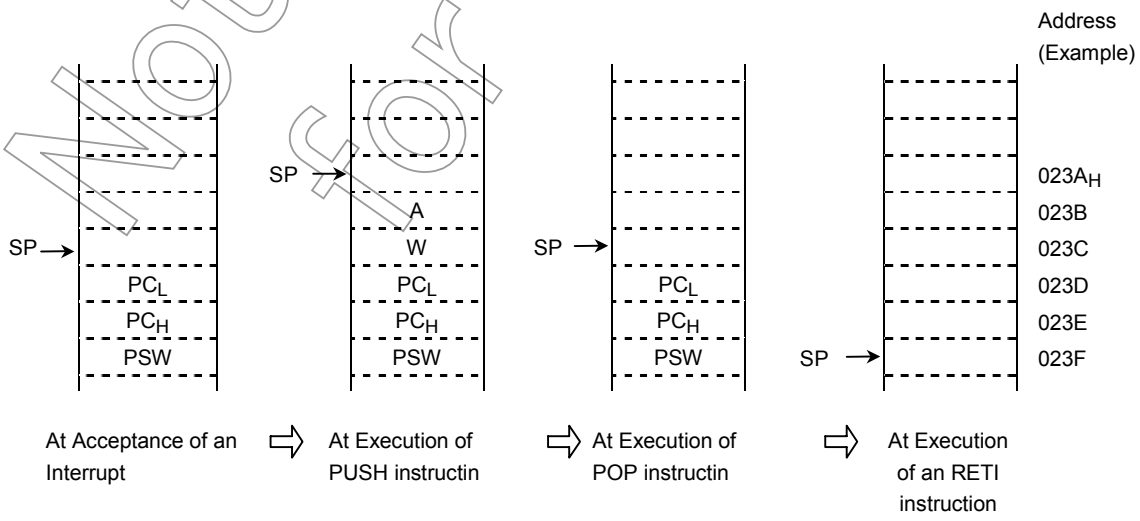
(2) Saving/restoring general-purpose registers

During interrupt acceptance processing, the program counter (PC) and the program status word (PSW, includes IMF) are automatically saved on the stack, but the accumulator and others are not. These registers are saved by software if necessary. When multiple interrupt services are nested, it is also necessary to avoid using the same data memory area for saving registers. The following methods are used to save/restore the general-purpose registers.

a. Using PUSH and POP instructions

To save only a specific register, PUSH and POP instructions are available.

```
Example:  Save/store register using PUSH and POP instructions
PINTxx:   PUSH    WA              ;  Save WA register
          (interrupt processing)
          POP     WA              ;  Restore WA register
                  RETI            ;  RETURN
```



b. Using data transfer instructions

To save only a specific register without nested interrupts, data transfer instructions are available.

```
Example:  Save/store register using data transfer instructions
PINTxx:   LD      (GSAVA), A        ;   Save A register
          (interrupt processing)
          LD      A, (GSAVA)        ;   Restore A register
          RETI                      ;   RETURN
```

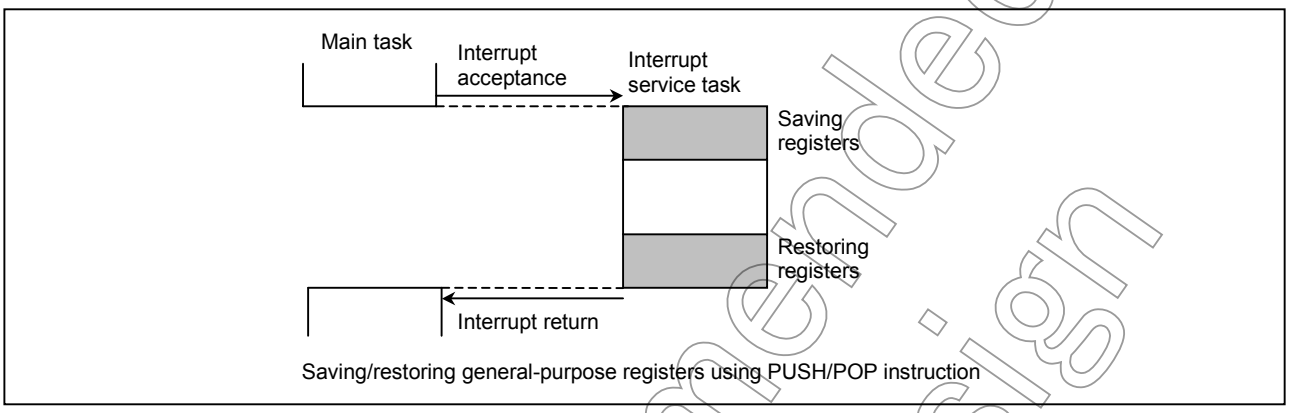


Figure 1.5.5 Saving/Restoring General-purpose Registers under Interrupt Processing

(3) Interrupt return

Interrupt return instructions [RETI]/[RETN] perform as follows.

[RETI]/[RETN] Interrupt Return
1. Program Counter (PC) and program status word (PSW, includes IMF) are restored from the stack.
2. Stack pointer (SP) is incremented by 3.

As for Address Trap interrupt (INTARTAP), it is required to alter stacked data for program counter (PC) to restarting address, during interrupt service program. Otherwise returning interrupt causes INTATRAP again. When interrupt acceptance processing has completed, stacked data for PC_L and PC_H are located on address (SP + 1) and (SP + 2) respectively.

Note: If [RETN] is executed with the above data unaltered, the program returns to the address trap area and INTATRAP occurs again.

Example 1: Returning from address trap interrupt (INTATRAP) service program

```
PINTxx:   POP    WA                    ;   Recover SP by 2
          LD     WA, Return Address    ;
          PUSH   WA                    ;   Alter stacked data
          (interrupt processing)
          RETN                         ;   RETURN
```

Example 2: Restarting without returning interrupt (In this case, PSW (includes IMF) before interrupt acceptance is discarded.)

```
PINTxx    INC    SP                    ;   Recover SP by 3
          INC    SP                    ;
          INC    SP                    ;
          (interrupt processing)
          LD     EIRL, data            ;   Set IMF to "1" or clear it to "0"
          JP     Restart Address       ;   Jump into restarting address
```

Note: It is recommended that stack pointer be return to rate before INTATRAP (increment 3 times), if return interrupt instruction [RETN] is not utilized during interrupt service program under INTATRAP (such as Example 2).

Interrupt requests are sampled during the final cycle of the instruction being executed. Thus, the next interrupt can be accepted immediately after the interrupt return instruction is executed.

Note: When the interrupt processing time is longer than the interrupt request generation time, the interrupt service task is performed but not the main task.

1.5.2 Software Interrupt (INTSW)

Executing the [SWI] instruction generates a software interrupt and immediately starts interrupt processing (INTSW is highest prioritized interrupt).

Use the [SWI] instruction only for detection of the address error or for debugging.

(1) Address error detection

FF_H is read if for some cause such as noise the CPU attempts to fetch an instruction from a non-existent memory address during single chip mode. Code FF_H is the SWI instruction, so a software interrupt is generated and an address error is detected. The address error detection range can be further expanded by writing FF_H to unused areas of the program memory. Address trap reset is generated in case that an instruction is fetched from RAM or SFR areas.

(2) Debugging

Debugging efficiency can be increased by placing the SWI instruction at the software break point setting address.

1.5.3 Undefined Instruction Interrupt (INTUNDEF)

Taking code which is not defined as authorized instruction for instruction causes INTUNDEF. INTUNDEF is generated when the CPU fetches such a code and tries to execute it. INTUNDEF is accepted even if non-maskable interrupt is in process. Contemporary process is broken and INTUNDEF interrupt process starts, soon after it is requested.

Note: The undefined instruction interrupt (INTUNDEF) forces CPU to jump into vector address, as software interrupt (SWI) does.

1.5.4 Address Trap Interrupt (INTATRAP)

Fetching instruction from unauthorized area for instructions (Address trapped area) causes reset-output or address trap interrupt (INTATRAP). INTATRAP is accepted even if non-maskable interrupt is in process. Contemporary process is broken and INTATRAP interrupt process starts, soon after it is requested.

Note: The operating mode under address trapped, whether to be reset-output or interrupt processing, is selected on watchdog timer control register (WDTCR).

1.5.5 External Interrupts

The TMP86FM29 has five external interrupt inputs. These inputs are equipped with digital noise reject circuits (Pulse inputs of less than a certain time are eliminated as noise).

Edge selection is also possible with INT1 to INT3. $\overline{INT0}$ /P63 pin can be configured as either an external interrupt input pin or an input/output port, and is configured as an input port during reset.

Edge selection, noise reject control and $\overline{INT0}$ /P63 pin function selection are performed by the external interrupt control register (EINTCR).

Table 1.5.2 External Interrupts

Source	Pin	Secondary Function Pin	Enable Conditions	Edge	Digital Noise Reject
INT0	$\overline{INT0}$	P63/AIN3	IMF = 1, EF_4 = 1, INT0EN = 1	Falling edge	Pulses of less than 2/fc [s] are eliminated as noise. Pulses of 7/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT1	INT1	P12/SEG29	IMF•EF_5 = 1	Falling edge or Rising edge	Pulses of less than 15/fc or 63/fc [s] are eliminated as noise. Pulses of 49/fc or 193/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT2	INT2	P13/SEG28	IMF•EF_7 = 1		Pulses of less than 7/fc [s] are eliminated as noise. Pulses of 25/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT3	INT3	P14/SEG27	IMF•EF_{14} = 1 IL14ER = 0		
INT5	$\overline{INT5}$	P20/ $\overline{STOP}$	IMF•EF_{15} = 1 IL15ER = 0	Falling edge	Pulses of less than 2/fc [s] are eliminated as noise. Pulses of 7/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.

Note 1: If a noiseless signal is input to the external interrupt pin in the NORMAL 1/2 or IDLE 1/2 mode, the maximum time from the edge of input signal until the IL is set is as follows:

(1) INT1 pin 55/fc [s] (INT1NC = 1), 199/fc [s] (INT1NC = 0)

(2) INT2, INT3 pin 31/fc [s]

Note 2: Even if the falling edge of $\overline{INT0}$ pin input is detected at INT0EN = 0, the interrupt latch IL_4 is not set.

Note 3: When data changed and did a change of I/O when used external interrupt ports as a normal ports, interrupt request signal occurs incorrectly. Handling of prohibition of interrupt enable register (EIR) is necessary.

Note 4: The maximum time from modifying INT1NC until a noise reject time is changed is 2^6/fc.

External interrupt control register

EINTCR (0037_H)

7	6	5	4	3	2	1	0
INT1NC	INT0EN			INT3ES	INT2ES	INT1ES	

(Initial value: 00** 000*)

INT1NC	Noise reject time select	0: Pulses of less than 63/fc [s] are eliminated as noise 1: Pulses of less than 15/fc [s] are eliminated as noise	R/W
INT0EN	P63/ $\overline{INT0}$ pin configuration	0: P63 input/output port 1: $\overline{INT0}$ pin (Port P63 should be set to an input mode)	
INT3ES INT2ES INT1ES	INT3 to INT1 edge select	0: Rising edge 1: Falling edge	

Note 1: fc: High-frequency clock [Hz], *: Don't care

Note 2: When the system clock frequency is switched between high and low or when the external interrupt control register (EINTCR) is overwritten, the noise canceller may not operate normally. It is recommended that external interrupts are disabled using the interrupt enable register (EIR).

Figure 1.5.6 External Interrupt Control Register

1.6 Reset Circuit

The TMP86FM29 has four types of reset generation procedures: an external reset input, an address trap reset output, a watchdog timer reset output and a system clock reset output. Table 1.6.1 shows on-chip hardware initialization by reset action.

Since the reset circuit has an 11-stage counter for generation of flash reset, which is the reset counter for stabilizing of the power supply for Flash, the reset period is 2^{10}/fc [s] (64 μs at 16.0 MHz).

Because the malfunction reset output circuit such as watchdog timer reset, address trap reset and system clock reset is not initialized when power is turned on, the $\overline{\text{RESET}}$ pin outputs "L" level for the maximum 24/fc [s] (1.5 μs at 16.0 MHz).

Therefore, the maximum reset period is 24/fc [s] + 2^{10}/fc [s] (65.5 μs at 16.0 MHz).

Table 1.6.1 shows on-chip hardware initialization by reset action.

Table 1.6.1 Initializing Internal Status by Reset Action

On-chip Hardware	Initial Value	On-chip Hardware	Initial Value
Program counter (PC)	($FFFE_H$)	Prescaler and Divider of timing generator	0
Stack pointer (SP)	Not initialized		
General-purpose registers (W, A, B, C, D, E, H, L, IX, IY)	Not initialized		
Jump status flag (JF)	Not initialized	Watchdog timer	Enable
Zero flag (ZF)	Not initialized	Output latches of I/O ports	Refer to I/O port circuitry
Carry flag (CF)	Not initialized		
Half carry flag (HF)	Not initialized		
Sign flag (SF)	Not initialized		
Overflow flag (VF)	Not initialized		
Interrupt master enable flag (IMF)	0		
Interrupt individual enable flags (EF)	0	Control registers	Refer to each of control register
Interrupt latches (IL)	0		
		RAM	Not initialized

1.6.1 External Reset Input

The $\overline{\text{RESET}}$ pin contains a Schmitt trigger (Hysteresis) with an internal pull-up resistor.

When the $\overline{\text{RESET}}$ pin is held at "L" level for at least 3 machine cycles (12/fc [s]) with the power supply voltage within the operating voltage range and oscillation stable, a reset is applied and the internal state is initialized.

When 2^{10}/fc (65.5 μs at 16 MHz) period passes after the $\overline{\text{RESET}}$ pin input goes high, the reset operation is released and the program execution starts at the vector address stored at addresses $FFFE_H$ to $FFFF_H$.

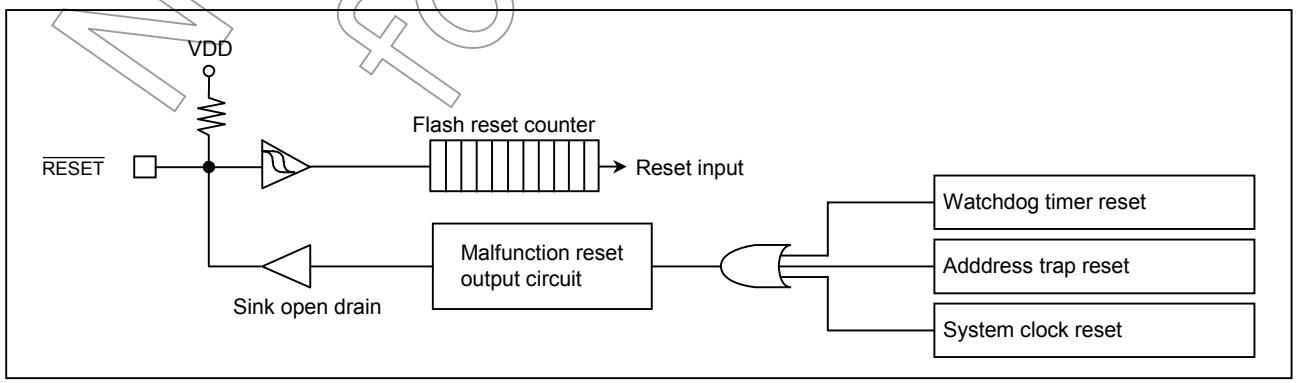
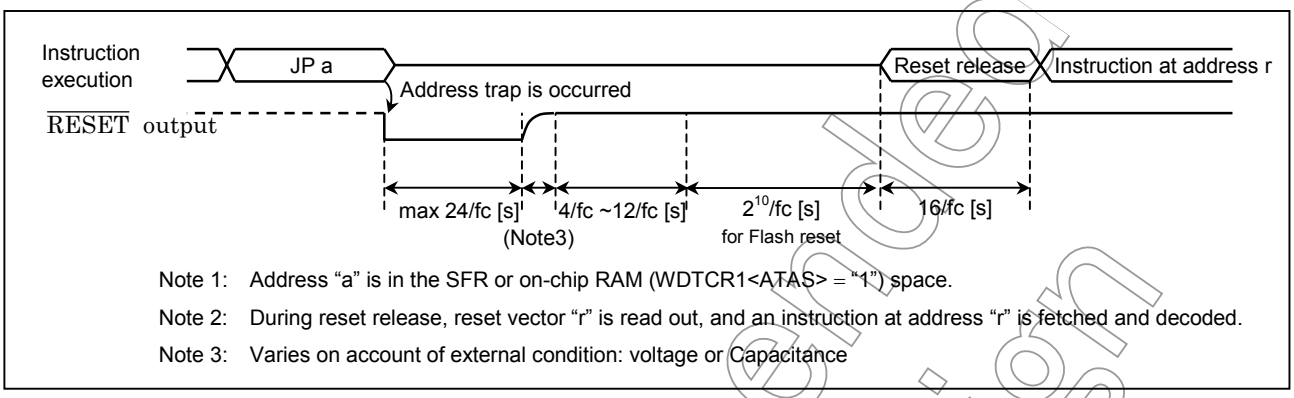


Figure 1.6.1 Reset Circuit

1.6.2 Address-Trap-Reset

If the CPU should start looping for some cause such as noise and an attempt be made to fetch an instruction from the on-chip RAM (when WDTCR1<ATAS> is set to "1") or the SFR area, address-trap-reset and the Flash reset will be generated. During address-trap-reset, the $\overline{\text{RESET}}$ pin outputs "L" level. The reset time is maximum 24/fc [s] + 2^{10}/fc [s] (65.5 μs at 16.0 MHz).



Note 1: Address "a" is in the SFR or on-chip RAM (WDTCR1<ATAS> = "1") space.

Note 2: During reset release, reset vector "r" is read out, and an instruction at address "r" is fetched and decoded.

Note 3: Varies on account of external condition: voltage or Capacitance

Figure 1.6.2 Address-Trap-Reset

Note: The operating mode under address trapped is alternative of reset or interrupt. Address trap or no address trap can be selected by WDTCR1<ATAS> for the internal RAM.

1.6.3 Watchdog Timer Reset

Refer to Section "2.4 Watchdog Timer".

1.6.4 System-Clock-Reset

If the condition as follows is detected, the system clock reset occurs automatically to prevent dead lock of the CPU. (The oscillation is continued without stopping.)

During system clock reset, the $\overline{\text{RESET}}$ pin outputs "L" level.

- In case of clearing SYSCR2<XEN> and SYSCR2<XTEN> simultaneously to "0".
- In case of clearing SYSCR2<XEN> to "0", when the SYSCR2<SYSCK> is "0".
- In case of clearing SYSCR2<XTEN> to "0", when the SYSCR2<SYSCK> is "1".

When the system clock reset is generated, the flash reset is also generated. Therefore, the maximum reset period is 24/fc [s] + 2^{10}/fc [s] (65.5 μs at 16.0 MHz).

2. On-Chip Peripherals Functions

2.1 Special Function Register (SFR)

The TMP86FM29 adopt the memory mapped I/O system, and all peripheral control and data transfers are performed through the special function register (SFR). The SFR is mapped on address 0000_H to $003F_H$, DBR is mapped on address $0F80_H$ to $0FFF_H$.

Figure 2.1.1 to Figure 2.1.2 indicate the special function register (SFR) and data buffer register (DBR) for TMP86FM29.

Address	Read	Write
0000_H	Reserved	
01	P1DR (P1 Port output latch)	
02	P2DR (P2 Port output latch)	
03	P3DR (P3 Port output latch)	
04	P3OUTCR (P3 Port output circuit control)	
05	P5DR (P5 Port output latch)	
06	P6DR (P6 Port output latch)	
07	P7DR (P7 Port output latch)	
08	P1PRD (P1 Terminal input)	–
09	P2PRD (P2 Terminal input)	–
0A	P3PRD (P3 Terminal input)	–
0B	P5PRD (P5 Terminal input)	–
0C	P6CR (P6 Port input/output control)	
0D	P7PRD (P7 Terminal input)	–
0E	ADCCR1 (AD control register 1)	
0F	ADCCR2 (AD control register 2)	
10	$TREG1A_L$	
11	$TREG1A_M$ (Timer register 1A)	
12	$TREG1A_H$	
13	TREG1B (Timer register 1B)	
14	TC1CR1 (Timer Counter 1 control 1)	
15	TC1CR2 (Timer Counter 1 control 2)	
16	TC1SR (TC1 Status)	–
17	Reserved	
18	TC3CR (Timer Counter 3 control)	
19	TC4CR (Timer Counter 4 control)	
1A	TC5CR (Timer Counter 5 control)	
1B	TC6CR (Timer Counter 6 control)	
1C	TTREG3 (Timer register 3)	
1D	TTREG4 (Timer register 4)	
1E	TTREG5 (Timer register 5)	
1F	TTREG6 (Timer register 6)	

Address	Read	Write
0020_H	ADCDR1 (AD converter register 1)	–
21	ADCDR2 (AD converter register 2)	–
22	Reserved	
23	Reserved	
24	Reserved	
25	UARTSR (UART Status register)	UARTCR1 (UART control register 1)
26	–	UARTCR2 (UART control register 2)
27	Reserved	
28	LCDCR (LCD Control register)	
29	P1LCR (P1 segment output control)	
2A	P5LCR (P5 segment output control)	
2B	P7LCR (P7 segment output control)	
2C	PWREG3 (Timer register 3)	
2D	PWREG4 (Timer register 4)	
2E	PWREG5 (Timer register 5)	
2F	PWREG6 (Timer register 6)	
30	Reserved	
31	Reserved	
32	Reserved	
33	Reserved	
34	–	WDTCR1 (watch dog timer control)
35	–	WDTCR2 (watch dog timer control)
36	TBTCR (TBT/TG/DVO control)	
37	EINTCR (External interrupt control)	
38	SYSCR1 (System control 1)	
39	SYSCR2 (System control 2)	
3A	EIR_L (Interrupt enable register)	
3B	EIR_H (Interrupt enable register)	
3C	IL_L (Interrupt latch)	
3D	IL_H (Interrupt latch)	
3E	INTSEL (Interrupt source selector)	
3F	PSW (Program Status word)	

Note 1: Do not access reserved areas by the program.

Note 2: –; Cannot be accessed.

Note 3: Write-only registers and interrupt latches cannot use the read-modify-write instructions (bit manipulation instructions such as SET, CLR, etc. and logical operation instructions such as AND, OR, etc.).

Figure 2.1.1 The special function register (SFR) for TMP86FM29 (1/2)

Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0F80H	SEG1				SEG0			
81	SEG3				SEG2			
82	SEG5				SEG4			
83	SEG7				SEG6			
84	SEG9				SEG8			
85	SEG11				SEG10			
86	SEG13				SEG12			
87	SEG15				SEG14			
88	SEG17				SEG16			
89	SEG19				SEG18			
8A	SEG21				SEG20			
8B	SEG23				SEG22			
8C	SEG25				SEG24			
8D	SEG27				SEG26			
8E	SEG29				SEG28			
8F	SEG31				SEG30			

Address	Read	Write
0F90H	SIOBR0 (SIO Buffer 0)	
91	SIOBR1 (SIO Buffer 1)	
92	SIOBR2 (SIO Buffer 2)	
93	SIOBR3 (SIO Buffer 3)	
94	SIOBR4 (SIO Buffer 4)	
95	SIOBR5 (SIO Buffer 5)	
96	SIOBR6 (SIO Buffer 6)	
97	SIOBR7 (SIO Buffer 7)	
98	–	SIOCR1 (SIO Control register 1)
99	SIOSR (SIO Status register)	SIOCR2 (SIO Control register 2)
9A		STOPCR (Key On Wake Up Control register)
9B	RDBUF (UART received data buffer)	TDBUF (UART transmited data buffer)
9C	Reserved	
·	Reserved	
·	Reserved	
·	Reserved	
E0	EEPCR (FLASH control)	
E1	EEPSR (FLASH status)	
·	Reserved	
·	Reserved	
·	Reserved	
FF	Reserved	

Note 1: Do not access reserved areas by the program.

Note 2: –; Cannot be accessed.

Note 3: Write-only registers and interrupt latches cannot use the read-modify-write instructions (bit manipulation instructions such as SET, CLR, etc. and logical operation instructions such as AND, OR, etc.).

Figure 2.1.2 The special function register (SFR) for TMP86FM29 (2/2)

2.2 I/O Ports

The TMP86FM29 have 6 parallel input/output ports (39 pins) as follows.

	Primary Function	Secondary Functions
Port P1	8-bit I/O port	External interrupt input, serial interface input/output, UART input/output, segment output and serial PROM mode control input.
Port P2	3-bit I/O port	Low-frequency resonator connections, external interrupt input, STOP mode release signal input.
Port P3	4-bit I/O port	Timer/counter input/output and divider output.
Port P5	8-bit I/O port	Segment output.
Port P6	8-bit I/O port	Analog input, external interrupt input, timer/counter input and STOP mode release signal input.
Port P7	8-bit I/O port	Segment output.

Each output port contains a latch, which holds the output data. All input ports do not have latches, so the external input data should be externally held until the input data is read from outside or reading should be performed several times before processing. Figure 2.2.1 shows input/output timing examples.

External data is read from an I/O port in the S1 state of the read cycle during execution of the read instruction. This timing cannot be recognized from outside, so that transient input such as chattering must be processed by the program.

Output data changes in the S2 state of the write cycle during execution of the instruction which writes to an I/O port.

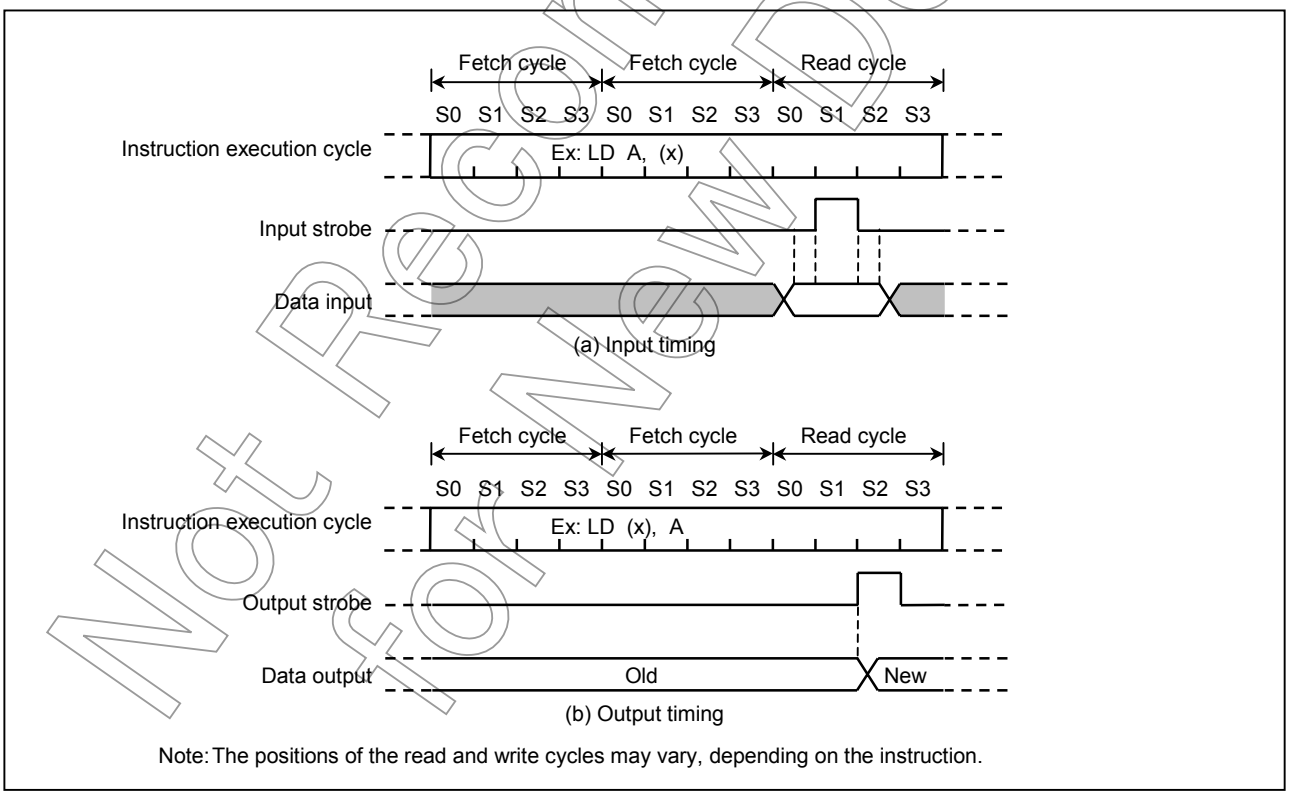


Figure 2.2.1 Input/Output timing (example)

2.2.1 Port P1 (P17 to P10)

Port P1 is an 8-bit input/output port which is also used as an external interrupt input, serial interface input/output, UART input/output, segment output of LCD and serial PROM mode control input. When used as a segment pins of LCD, the respective bit of P1LCR should be set to "1".

When used as an input port or a secondary function (except for segment) pins, the respective output latch (P1DR) should be set to "1" and its corresponding P1LCR bit should be set to "0". When used as an output port, the respective P1LCR bit should be set to "0". During reset, the output latch is initialized to "1".

For detail of the serial PROM mode, refer to "2.16 Serial PROM mode".

P1 port output latch (P1DR) and P1 port terminal input (P1PRD) are located on their respective address.

When read the output latch data, the P1DR should be read and when read the terminal input data, the P1PRD register should be read.

If the terminal input data which is configured as LCD segment output is read, unstable data is read.

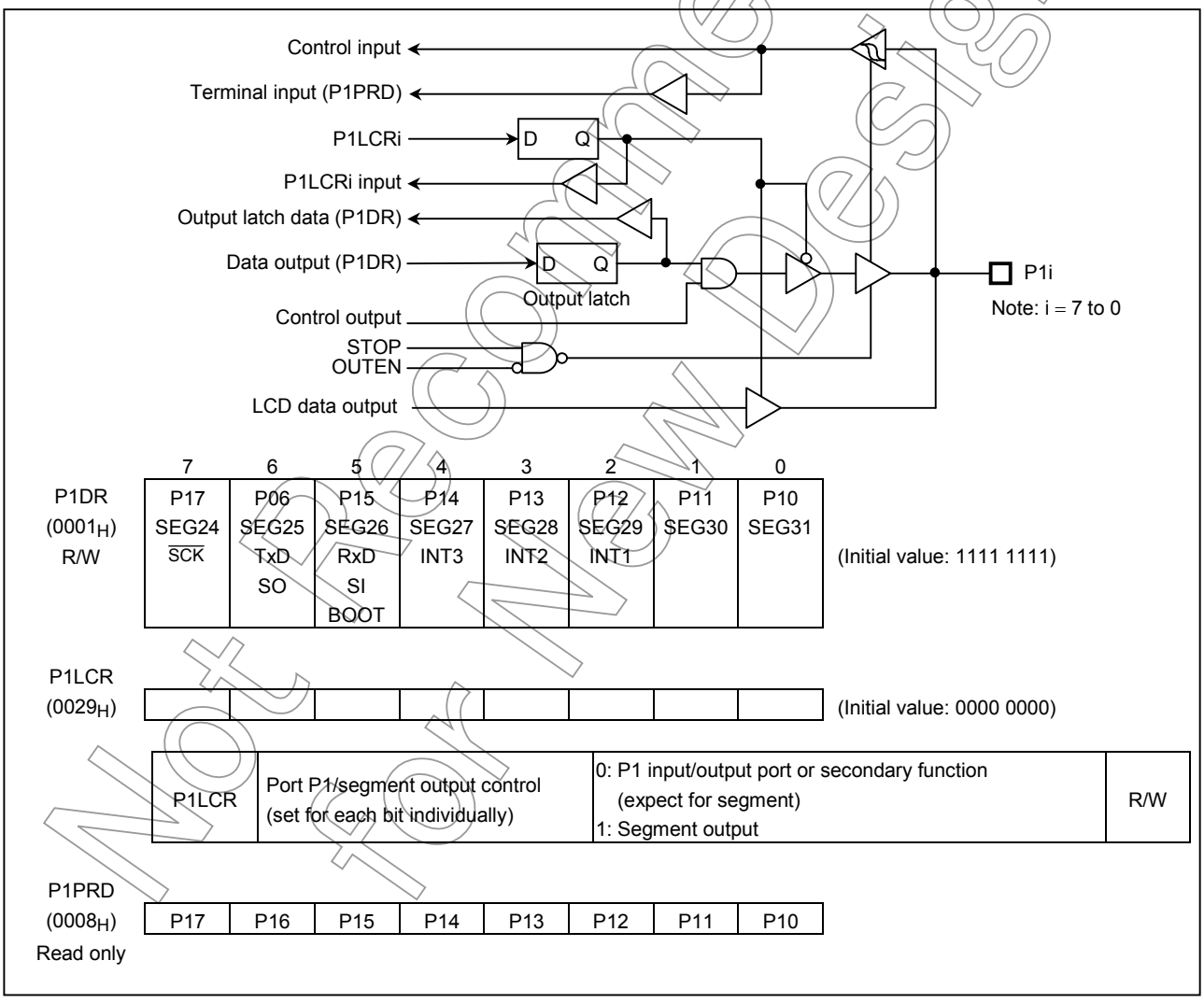


Figure 2.2.2 Port 1

2.2.2 Port P2 (P22 to P20)

Port P2 is a 3-bit input/output port.

It is also used as an external interrupt, a STOP mode release signal input, and low-frequency crystal oscillator connection pins. When used as an input port or a secondary function pins, respective output latch (P2DR) should be set to "1".

During reset, the P2DR is initialized to "1".

A low-frequency crystal oscillator (32.768 kHz) is connected to pins P21 (XTIN) and P22 (XTOUT) in the dual-clock mode. In the single-clock mode, pins P21 and P22 can be used as normal input/output ports.

It is recommended that pin P20 should be used as an external interrupt input, a STOP mode release signal input, or an input port. If it is used as an output port, the interrupt latch is set on the falling edge of the output pulse.

P2 port output latch (P2DR) and P2 port terminal input (P2PRD) are located on their respective address.

When read the output latch data, the P2DR should be read and when read the terminal input data, the P2PRD register should be read. If a read instruction is executed for port P2, read data of bits 7 to 3 are unstable.

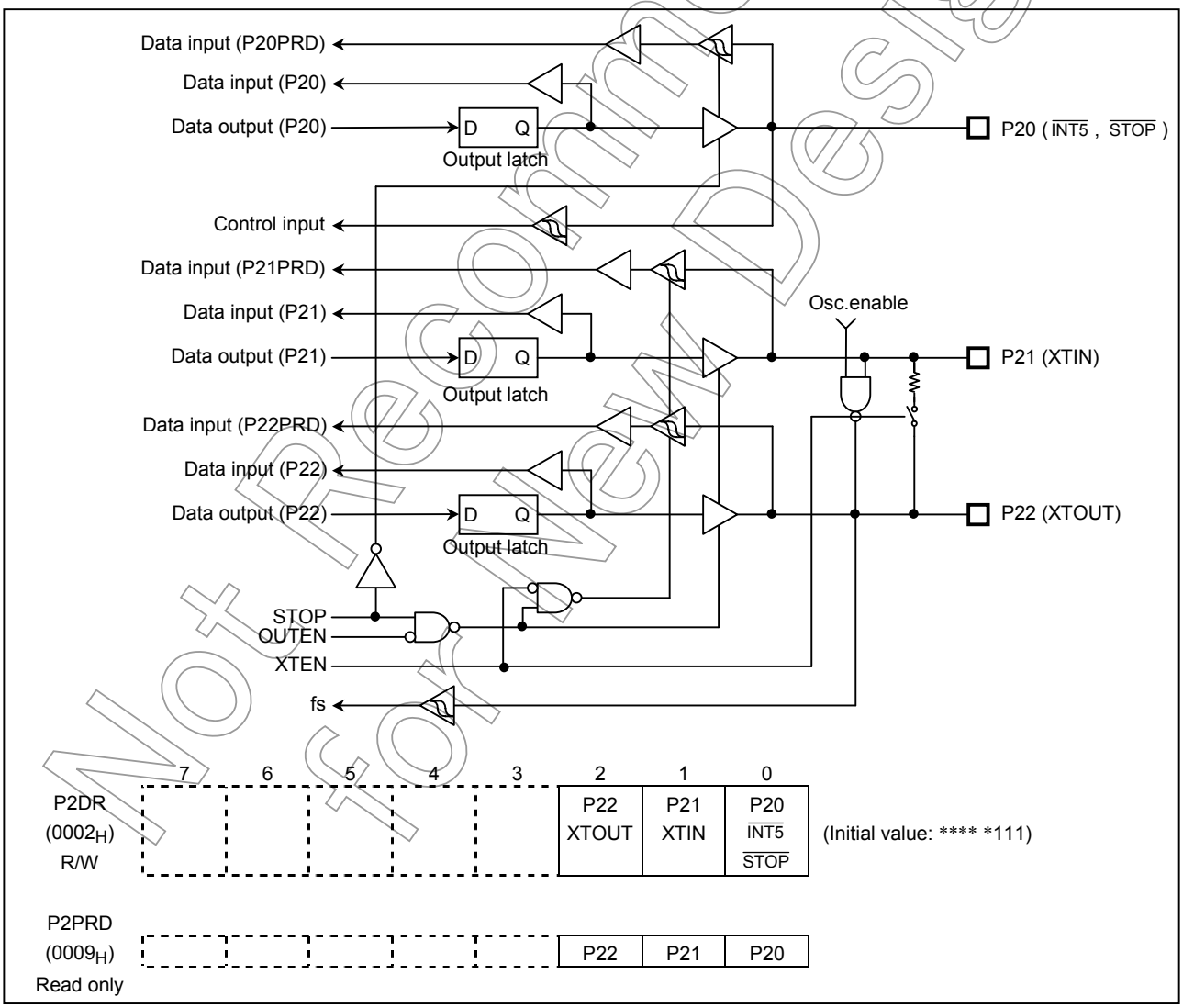


Figure 2.2.3 Port 2

Note: Port P20 is used as $\overline{\text{STOP}}$ pin. Therefore, when stop mode is started, OUTEN does not affect to P20, and P20 becomes High-Z mode.

2.2.3 Port P3 (P33 to P30)

Port P3 is a 4-bit input/output port.

It is also used as a timer/counter input/output, divider output.

When used as a timer/counter output or divider output, respective output latch (P3DR) should be set to "1".

It can be selected whether output circuit of P3 port is C-MOS output or a sink open drain individually, by setting P3OUTCR. When a corresponding bit of P3OUTCR is "0", the output circuit is selected to a sink open drain and when a corresponding bit of P3OUTCR is "1", the output circuit is selected to a C-MOS output. When used as an input port or timer/counter input, respective output control (P3OUTCR) should be set to "0" after P3DR is set to "1". During reset, the P3DR is initialized to "1", and the P3OUTCR is initialized to "0".

P3 port output latch (P3DR) and P3 port terminal input (P3PRD) are located on their respective address.

When read the output latch data, the P3DR should be read and when read the terminal input data, the P3PRD register should be read. If a read instruction is executed for port P3, read data of bits 7 to 4 are unstable.

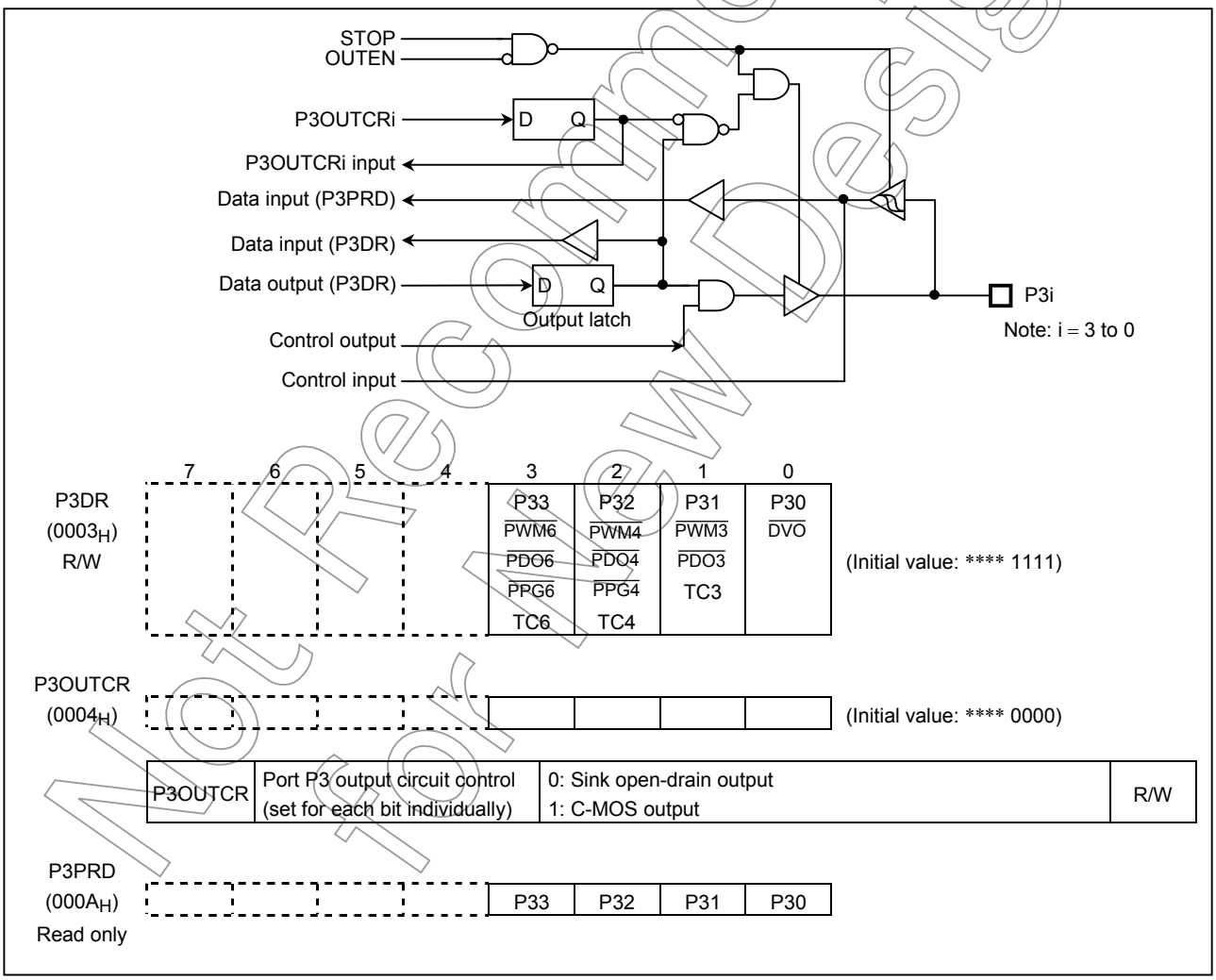


Figure 2.2.4 Port 3

2.2.4 Port P5 (P57 to P50)

Port P5 is an 8-bit input/output port which is also used as a segment pins of LCD.

When used as input port, the respective output latch (P5DR) should be set to "1".

During reset, the P5DR is initialized to "1".

When used as a segment pins of LCD, the respective bit of P5LCR should be set to "1". When used as an output port, the respective P5LCR bit should be set to "0".

P5 port output latch (P5DR) and P5 port terminal input (P5PRD) are located on their respective address.

When read the output latch data, the P5DR should be read and when read the terminal input data, the P5PRD register should be read. If the terminal input data which is configured as LCD segment output is read, unstable data is read.

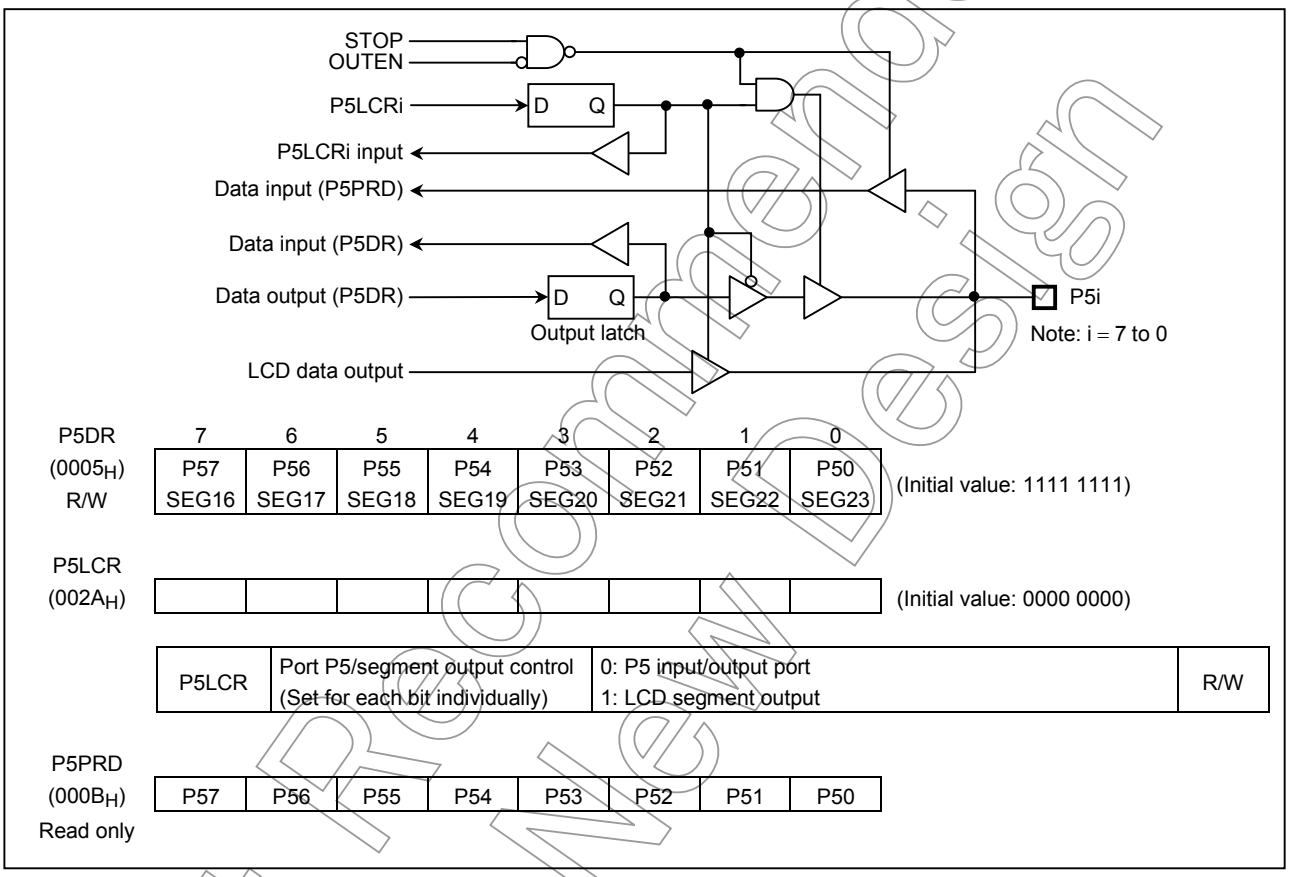


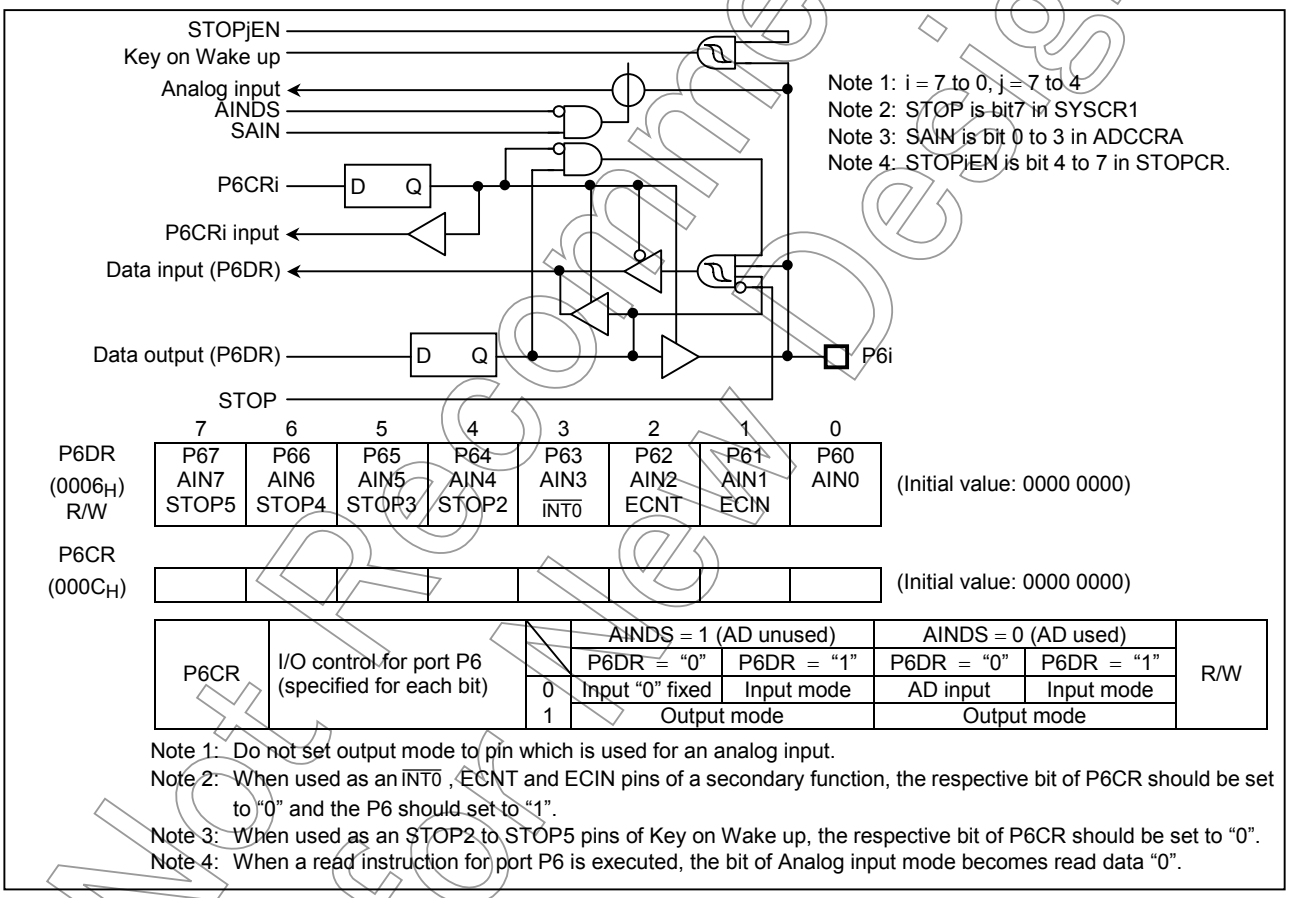
Figure 2.2.5 Port 5

2.2.5 Port P6 (P67 to P60)

Port P6 is an 8-bit input/output port which can be configured as an input or an output in one-bit unit. Port P6 is also used as an analog input, Key on Wake up input, timer/counter input and external interrupt input. Input/output mode is specified by the P6 control register (P6CR), the P6 output latch (P6DR), and AINDS (bit 4 in ADCCR1). During reset, P6CR and P6DR are initialized to "0" and AINDS is set to "1". At the same time, the input data of pins P67 to P60 are fixed to "0". To use port P6 as an input port, external interrupt input, timer/counter input or key on wake up input, set data of P6DR to "1" and P6CR to "0". To use it as an output port, set data of P6CR to "1". To use it as an analog input, set data of P6DR to "0" and P6CR to "0", and start the AD. It is the penetration electric current measures by the analog voltage.

Pins not used for analog input can be used as I/O ports. During AD conversion, output instructions should not be executed to keep a precision. In addition, a variable signal should not be input to a port adjacent to the analog input during AD conversion.

When the AD converter is in use (P6DR = 0), bits mentioned above are read as "0" by executing input instructions.



P6DR (0006_H) R/W

7	6	5	4	3	2	1	0
P67 AIN7 STOP5	P66 AIN6 STOP4	P65 AIN5 STOP3	P64 AIN4 STOP2	P63 AIN3 $\overline{INT0}$	P62 AIN2 ECNT	P61 AIN1 ECIN	P60 AIN0

(Initial value: 0000 0000)

P6CR ($000C_H$)

(Initial value: 0000 0000)

P6CR	I/O control for port P6 (specified for each bit)		AINDS = 1 (AD unused)		AINDS = 0 (AD used)		R/W
			P6DR = "0"	P6DR = "1"	P6DR = "0"	P6DR = "1"	
		0	Input "0" fixed	Input mode	AD input	Input mode	
		1	Output mode		Output mode		

Note 1: Do not set output mode to pin which is used for an analog input.
Note 2: When used as an $\overline{INT0}$, ECNT and ECIN pins of a secondary function, the respective bit of P6CR should be set to "0" and the P6 should set to "1".
Note 3: When used as an STOP2 to STOP5 pins of Key on Wake up, the respective bit of P6CR should be set to "0".
Note 4: When a read instruction for port P6 is executed, the bit of Analog input mode becomes read data "0".

Figure 2.2.6 Port 6 and P6CR

Note: Although P6DR is a read/writer register, because it is also used as an input mode control function, read-modify-write instructions such as bit manipulate instructions cannot be used.
Read-modify-write instruction writes the all data of 8-bit after data is read and modified. Because a bit setting Input mode read data of terminal, the output latch is changed by these instruction. So P6 port can not input data.

2.2.6 Port P7 (P77 to P70)

Port P7 is an 8-bit input/output port which is also used as a segment pins of LCD.

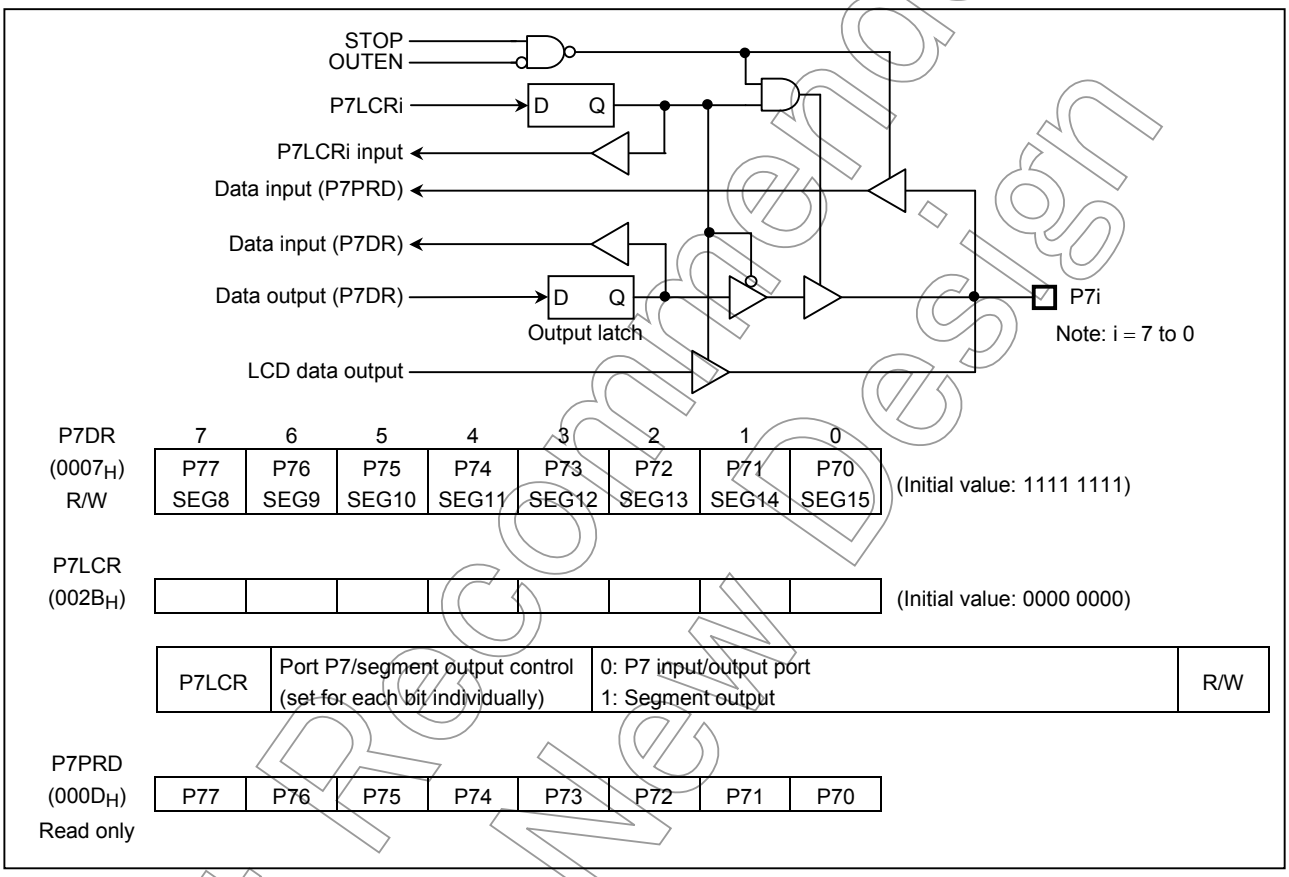
When used as input port, the respective output latch (P7DR) should be set to "1".

During reset, the P7DR is initialized to "1".

When used as a segment pins of LCD, the respective bit of P7LCR should be set to "1". When used as an output port, the respective P7LCR bit should be set to "0".

P7 port output latch (P7DR) and P7 port terminal input (P7PRD) are located on their respective address.

When read the output latch data, the P7DR should be read and when read the terminal input data, the P7PRD register should be read. If the terminal input data which is configured as LCD segment output is read, unstable data is read.



P7DR	7	6	5	4	3	2	1	0	
(0007_H)	P77	P76	P75	P74	P73	P72	P71	P70	(Initial value: 1111 1111)
R/W	SEG8	SEG9	SEG10	SEG11	SEG12	SEG13	SEG14	SEG15	

P7LCR									
($002B_H$)									(Initial value: 0000 0000)

P7LCR	Port P7/segment output control (set for each bit individually)	0: P7 input/output port 1: Segment output	R/W

P7PRD									
($000D_H$)	P77	P76	P75	P74	P73	P72	P71	P70	
Read only									

Figure 2.2.7 Port 7

2.3 Time Base Timer (TBT)

The time base timer generates time base for key scanning, dynamic displaying, etc. It also provides a time base timer interrupt (INTTBT).

An INTTBT is generated on the first falling edge of source clock (The divider output of the timing generator) after the time base timer has been enabled. The divider is not cleared by the program; therefore, only the first interrupt may be generated ahead of the set interrupt period (Figure 2.3.1 (b)).

The interrupt frequency (TBTCK) must be selected with the time base timer disabled (the interrupt frequency must not be changed with the disable from the enable state). Both frequency selection and enabling can be performed simultaneously.

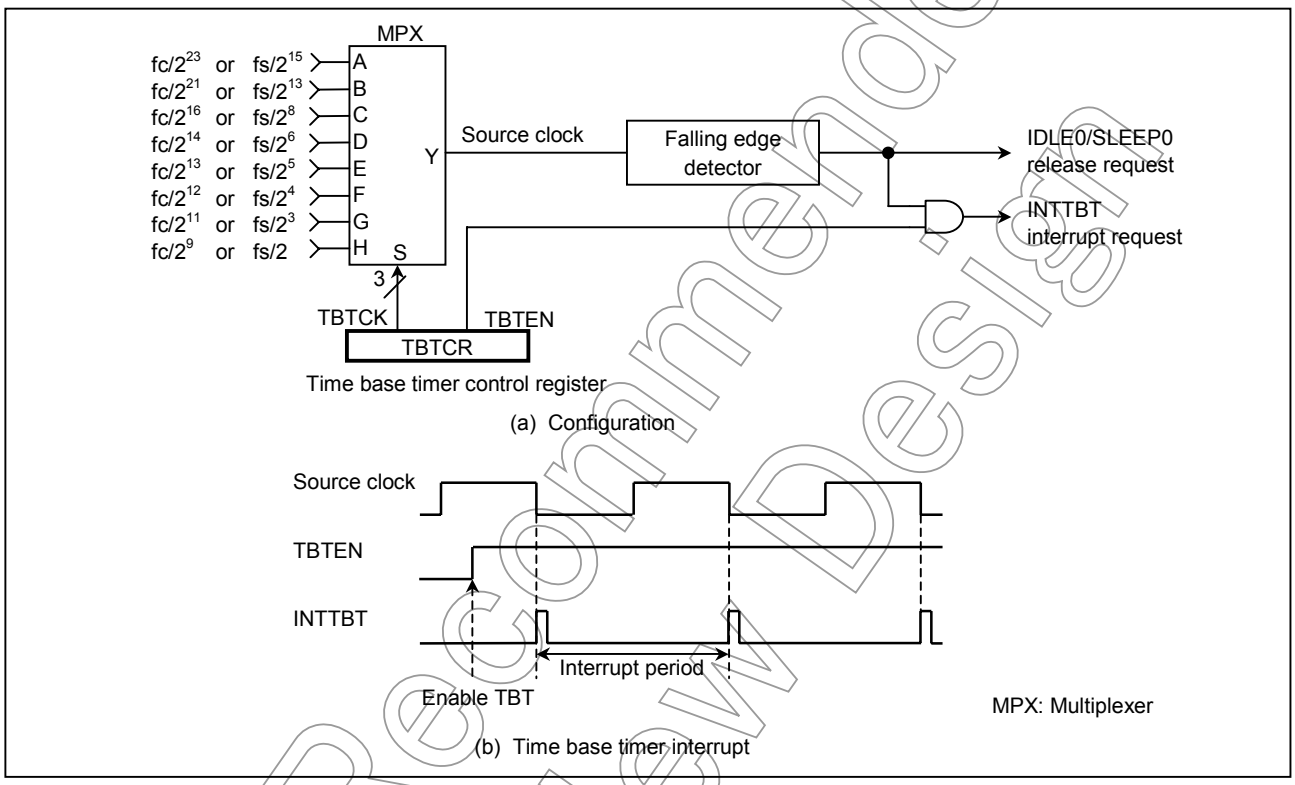


Figure 2.3.1 Time Base Timer

Example: Sets the time base timer frequency to fc/2^{16} [Hz] and enables an INTTBT interrupt.

```
LD      (TBTCR), 00000010B      ;   TBTCK ← 010
LD      (TBTCR), 00001010B      ;   TBTEN ← 1
DI                              ;   IMF ← 0
SET     (EIRL). 6
```

TBTCR (0036_H)

7	6	5	4	3	2	1	0
(DVOEN)	(DVOCK)		(DV7CK)	TBTEN	TBTCK		

(Initial value: 0000 0000)

TBTEN	Time base timer enable/disable	0: Disable 1: Enable			R/W
TBTCK	Time base timer interrupt frequency select [Hz]		NORMAL1/2, IDLE1/2 Mode	SLOW, SLEEP Mode	
			DV7CK = 0 / DV7CK = 1		
		000	$fc/2^{23}$ / $fs/2^{15}$	$fs/2^{15}$	
		001	$fc/2^{21}$ / $fs/2^{13}$	$fs/2^{13}$	
		010	$fc/2^{16}$ / $fs/2^{8}$	–	
		011	$fc/2^{14}$ / $fs/2^{6}$	–	
		100	$fc/2^{13}$ / $fs/2^{5}$	–	
		101	$fc/2^{12}$ / $fs/2^{4}$	–	
		110	$fc/2^{11}$ / $fs/2^{3}$	–	
		111	$fc/2^{9}$ / $fs/2$	–	

Note: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Figure 2.3.2 Time Base Timer Control Register

Table 2.3.1 Time Base Timer Interrupt Frequency (Example: fc = 16 MHz, fs = 32.768 kHz)

TBTCK	Time Base Timer Interrupt Frequency [Hz]		
	NORMAL1/2, IDLE1/2 Mode		SLOW, SLEEP Mode
	DV7CK = 0	DV7CK = 1	
000	1.91	1	1
001	7.63	4	4
010	244.14	128	–
011	976.56	512	–
100	1953.13	1024	–
101	3906.25	2048	–
110	7812.5	4096	–
111	31250	16384	–

2.4 Watchdog Timer (WDT)

The watchdog timer is a fail-safe system to rapidly detect the CPU malfunctions such as endless looping caused by noise or the like, or deadlock and resume the CPU to the normal state.

The watchdog timer signal for detecting malfunction can be selected either a “reset request” or a non-maskable “interrupt request”. However, selection is possible only once after reset. At first the “reset request” is selected.

When the watchdog timer is not being used for malfunction detection, it can be used as a timer to generate an interrupt at fixed intervals.

Note: Care must be given in system design so as to protect the watchdog timer from disturbing noise. Otherwise the Watchdog Timer may not fully exhibit its functionality.

2.4.1 Watchdog Timer Configuration

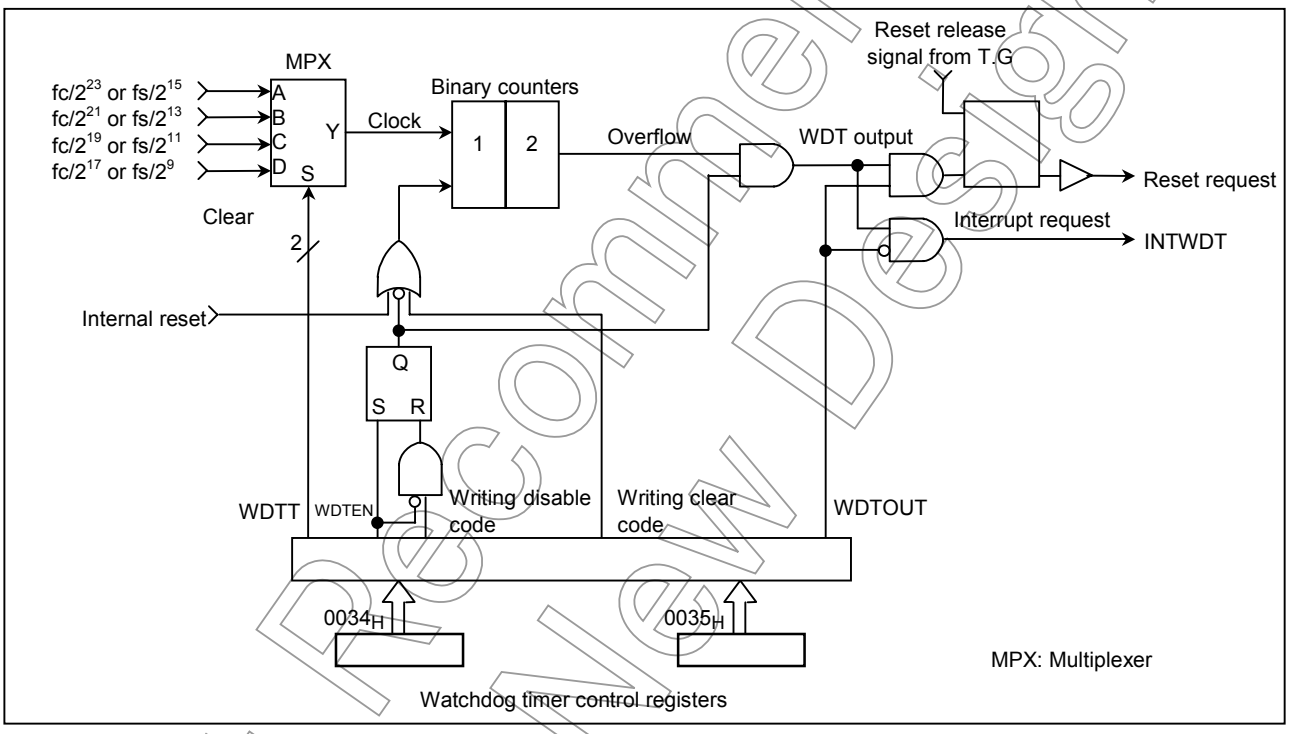


Figure 2.4.1 Watchdog Timer Configuration

2.4.2 Watchdog Timer Control

Figure 2.4.2 shows the watchdog timer control registers (WDTCR1, WDTCR2). The watchdog timer is automatically enabled after reset.

(1) Malfunction detection methods using the watchdog timer

The CPU malfunction is detected as follows.

1. Setting the detection time, selecting output, and clearing the binary counter.
2. Repeatedly clearing the binary counter within the setting detection time

If the CPU malfunctions such as endless looping or deadlock occur for any cause, the watchdog timer output will become active at the rising of an overflow from the binary counters unless the binary counters are cleared. At this time, when WDTCR1<WDTOUT> = "1", the $\overline{\text{RESET}}$ pin outpus "L" level and then the internal hardware is reseted. When WDTCR1<WDTOUT> = "0", a watchdog timer interrupt (INTWDT) is generated.

The watchdog timer temporarily stops counting in STOP mode including warm-up or IDLE mode, and automatically restarts (Continues counting) when the STOP/IDLE mode is released.

Note: The watchdog timer consists of an internal divider and a two-stage binary counter. When clear code $4E_H$ is written, only the binary counter is cleared, not the internal divider. Depending on the timing at which clear code $4E_H$ is written on the WDTCR2 register, the overflow time of the binary counter may be at minimum 3/4 of the time set in WDTCR1 <WDTT>. Thus, write the clear code using a shorter cycle than 3/4 of the time set in WDTCR1 <WDTT>.

Example: Sets the watchdog timer detection time to 2^{21}/fc [s] and resets the CPU malfunction.

```
SYSCR1      LD    (WDTCR2),  4EH          ;  Clears the binary counters
            LD    (WDTCR1),  00001101B    ;  WDTT ← 10, WDTOUT ← 1
Within 3/4 of WDT detection time
            LD    (WDTCR2), 4EH           ;  Clears the binary counters (Always clear
                                             immediately before and after changing
                                             WDTT)
Within 3/4 of WDT detection time
            LD    (WDTCR2), 4EH           ;  Clears the binary counters
            LD    (WDTCR2), 4EH           ;  Clears the binary counters
```

Watchdog Timer Register 1

WDTCR1 (0034H)

7	6	5	4	3	2	1	0
		(ATAS)	(ATOUT)	WDTEN	WDTT		WDTOUT

(Initial value: **11 1001)

WDTEN	Watchdog timer enable/disable	0: Disable (It is necessary to write the disable code to WDTCR2) 1: Enable				Write only
WDTT	Watchdog timer detection time [s]		NORMAL1/2 mode		SLOW mode	
			DV7CK = 0	DV7CK = 1		
		00	2^{25}/fc	2^{17}/fs	2^{17}/fs	
		01	2^{23}/fc	2^{15}/fs	2^{15}/fs	
		10	2^{21}/fc	2^{13}/fs	2^{13}/fs	
		11	2^{19}/fc	2^{11}/fs	2^{11}/fs	
WDTOUT	Watchdog timer output select	0: Interrupt request 1: Reset request				

Note 1: WDTOUT cannot be set to "1" by program after clearing WDTOUT to "0".

Note 2: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Note 3: WDTCR1 is a write-only register and must not be used with any of read-modify-write instructions.

Note 4: The watchdog timer must be disabled or the counter must be cleared immediately before entering to the STOP mode. When the counter is cleared, the counter must be cleared again immediately after releasing the STOP mode.

Note 5: To disable the watchdog timer, always write "$4E_H$" (Clear code) to WDTCR2 for clearing the binary counter before writing "0" to WDTEN, and then write "$B1_H$" (Disable code) to WDTCR2.
Also, immediately before these procedure, disable the interrupt mater flag (IMF) by DI instruction.

Watchdog Timer Register 2

WDTCR2 (0035H)

7	6	5	4	3	2	1	0

(Initial value: **** ****)

WDTCR2	Watchdog timer control code write register	$4E_H$: Watchdog timer binary counter clear (Clear code) $B1_H$: Watchdog timer disable (Disable code) $D2_H$: Enable assigning address trap area Others: Invalid	Write only

Note 1: The disable code is invalid unless written when WDTCR1<WDTEN> = 0.

Note 2: *: Don't care

Note 3: The binary counter of the watchdog timer must not be cleared by the interrupt task.

Note 4: Write clear code $4E_H$ within 3/4 of the time set in WDTCR1<WDTT>.

Figure 2.4.2 Watchdog Timer Control Registers

(2) Watchdog timer enable

The watchdog timer is enabled by setting WDTCR1<WDTEN> to "1".

WDTCR1<WDTEN> is initialized to "1" during reset, so the watchdog timer operates immediately after reset is released.

(3) Watchdog timer disable

To disable the watchdog time, write "$4E_H$" (Clear code) to WDTCR2 for clearing the binary counter before writing "0" to WDTCR1<WDTEN>, and then write "$B1_H$" (Disable code) to WDTCR2. The watchdog timer is not disabled if this procedure is reversed and the disable code is written to WDTCR2 before WDTCR1<WDTEN> is cleared to "0". Also, immediately before these procedure, disable the interrupt master flag (IMF) by DI instruction. During disabling the watchdog timer, the binary counters are cleared to "0".

```
Example: Disables watchdog timer
        DI                              ;   IMF ← 0
        LD      (WDTCR2), 4EH           ;   Clear the binary counter
        LDW     (WDTCR1), 0B101H        ;   WDTEN ← 0, WDTCR2 ← Disable code
```

Table 2.4.1 Watchdog Timer Detection Time (Example: fc = 16 MHz, fs = 32.768 kHz)

WDTT	Watchdog Timer Detection Time [s]		
	NORMAL1/2 Mode		SLOW Mode
	DV7CK = 0	DV7CK = 1	
00	2.097	4	4
01	524.288 m	1	1
10	131.072 m	250 m	250 m
11	32.768 m	62.5 m	62.5 m

2.4.3 Watchdog Timer Interrupt (INTWDT)

This is a non-maskable interrupt which can be accepted regardless of the contents of the EIR. If a watchdog timer interrupt or a software interrupt is already accepted, however, the new watchdog timer interrupt waits until the previous interrupt processing is completed (The end of the [RETN] instruction execution).

The stack pointer (SP) should be initialized before using the watchdog timer output as an interrupt source with WDTOUT.

```
Example: Watchdog timer interrupt setting up
        LD      SP, 023FH               ;   Sets the stack pointer
        LD      (WDTCR1), 00001000B     ;   WDTOUT ← 0
```

2.4.4 Watchdog Timer Reset

If the watchdog timer reset request occur, the $\overline{\text{RESET}}$ pin outputs "L" level and then the internal hardware is reseted. When the watchdog timer reset is generated, the flash reset is also generated. Therefore, the maximum reset period is 24/fc [s] + 2^{10}/fc [s] (65.5 μs at 16.0 MHz).

Note: The high-frequency clock oscillator also immediately turns on when a watchdog timer reset is generated in SLOW mode. In this case, the reset time may include a certain amount of error if there is any fluctuation of the oscillation frequency at starting the high-frequency clock oscillation. Therefore, the reset time must be considered an approximated value.

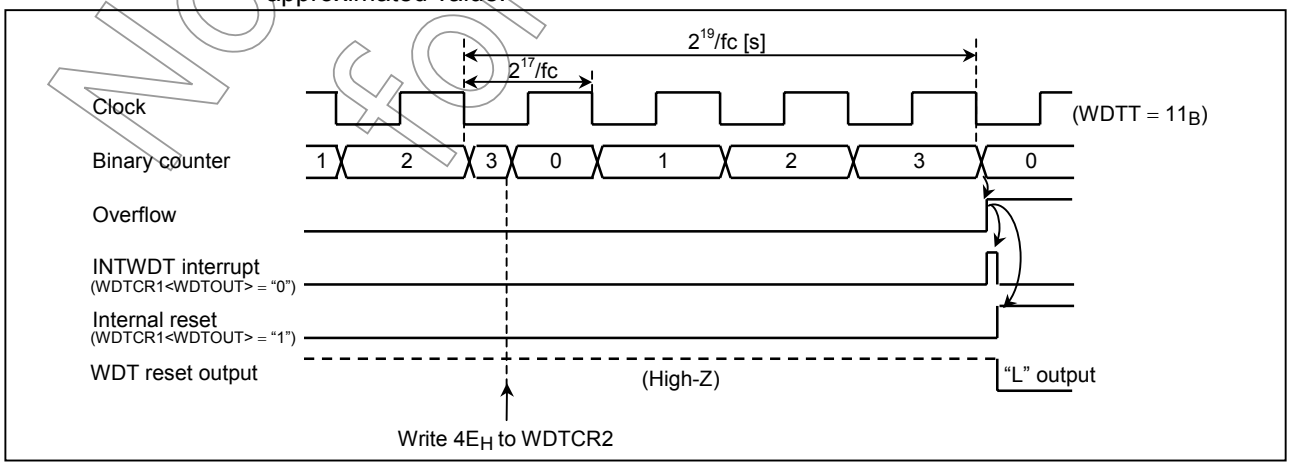


Figure 2.4.3 Watchdog Timer Interrupt/Reset

2.5 Address Trap

The watchdog timer control register 1, 2 shares its addresses with the control registers in case of address trap. These control registers for address trap are shown on Figure 2.5.1.

Watchdog Timer Control Register 1

WDTCR1 (0034_H)	7	6	5	4	3	2	1	0
	–	–	ATAS	ATOUT	(WDTEN)	(WDTT)		(WDTOUT)

(Initial value: **11 1001)

ATAS	Selection of address trap in internal RAM	0: No address trap 1: Address trap (After setting ATAS to "1", it is necessary to write the control code $D2_H$ to WDTCR2)	Write only
ATOUT	Selection of operation at address trap	0: Interrupt request 1: Reset request	

Watchdog Timer Control Register 2

WDTCR2 (0035_H)	7	6	5	4	3	2	1	0

(Initial value: **** ****)

WDTCR2	Watchdog timer control code and Address trapped area control code	$D2_H$: Address trapped area valid to set (ATRAP control code) $4E_H$: Watchdog timer binary counter clear (WDT clear code) $B1_H$: Watchdog timer disable (WDT disable code) Others: Invalid	Write only

Figure 2.5.1 Watchdog Timer Control Registers

(1) Selection of address trap in internal RAM (ATAS)

Using WDTCR1<ATAS>, address trap or no address trap can be selected for the internal RAM area. To execute an instruction in the internal RAM area, set "0" in WDTCR1<ATAS>. Setting in WDTCR1<ATAS> becomes valid after control code D2H is written in WDTCR2. Executing an instruction in the SFR/DBR area generates an address trap unconditionally regardless of the setting in WDTCR1<ATAS>.

(2) Selection of operation at address trap (ATOUT)

As the operation at address trap either interrupt generation or reset generation ($\overline{\text{RESET}}$ pin outputs "L" level) can be selected by WDTCR1<ATOUT>.

2.6 Divider Output (DVO)

Approximately 50% duty pulse can be output using the divider output circuit, which is useful for piezoelectric buzzer drive. Divider output is from pin P30 ($\overline{DVO}$). The P51 output latch should be set to "1".

Note: Selection of divider output frequency must be made while divider output is disabled. Also, in other words, when changing the state of the divider output frequency from enabled to disable, do not change the setting of the divider output frequency.

TBTCR (0036H)

7	6	5	4	3	2	1	0
DVOEN	DVOCK		(DV7CK)	(TBTEN)		(TBTCK)	

(Initial value: 0000 0000)

			NORMAL1/2 Mode		SLOW, SLEEP Mode	
			DV7CK = 0	DV7CK = 1		
DVOEN	Divider output enable/disable	0: Disable 1: Enable				R/W
DVOCK	Divider output ($\overline{DVO}$) frequency selection [Hz]	00	$fc/2^{13}$	$fs/2^5$	$fs/2^5$	
		01	$fc/2^{12}$	$fs/2^4$	$fs/2^4$	
		10	$fc/2^{11}$	$fs/2^3$	$fs/2^3$	
		11	$fc/2^{10}$	$fs/2^2$	$fs/2^2$	

Note: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Figure 2.6.1 Divider Output Control Register

Example: 1.95 kHz pulse output (at fc = 16.0 MHz)

```
SET     (P3DR).0              ;  P30 output latch ← "1"
LD      (TBTCR), 00000000B    ;  DVOCK ← "00"
LD      (TBTCR), 10000000B    ;  DVOEN ← "1"
```

Table 2.6.1 Divider Output Frequency (Example: at fc = 16.0 MHz, fs = 32.768 kHz)

DVOCK	Divider Output Frequency [Hz]		
	NORMAL1/2, IDLE1/2 Mode		SLOW, SLEEP Mode
	DV7CK = 0	DV7CK = 1	
00	1.953 k	1.024 k	1.024 k
01	3.906 k	2.048 k	2.048 k
10	7.813 k	4.096 k	4.096 k
11	15.625 k	8.192 k	8.192 k

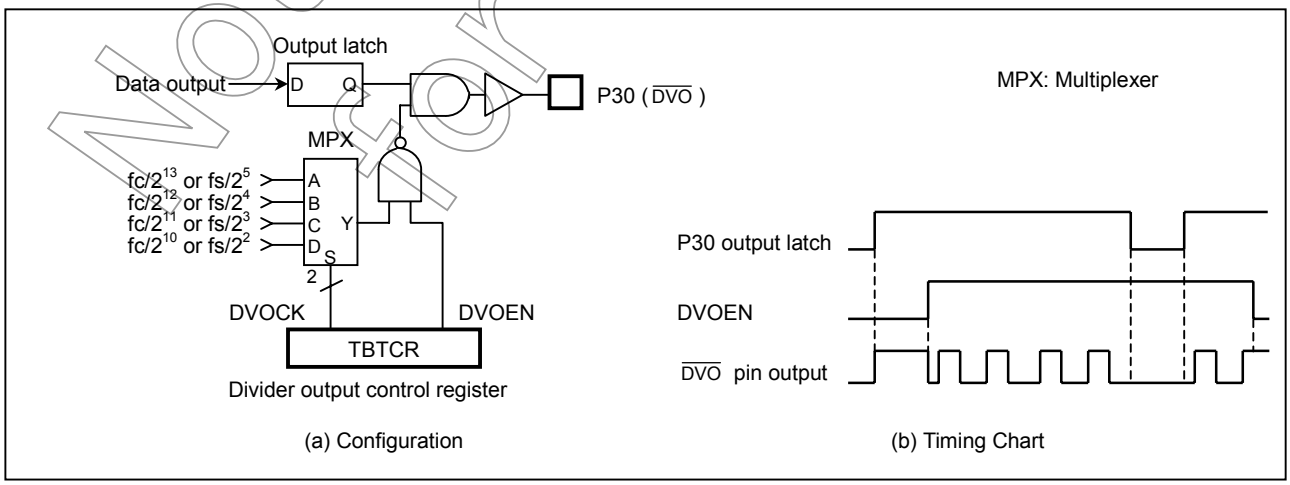


Figure 2.6.2 Divider Output

2.7 18-Bit Timer/Counter (TC1)

2.7.1 Configuration

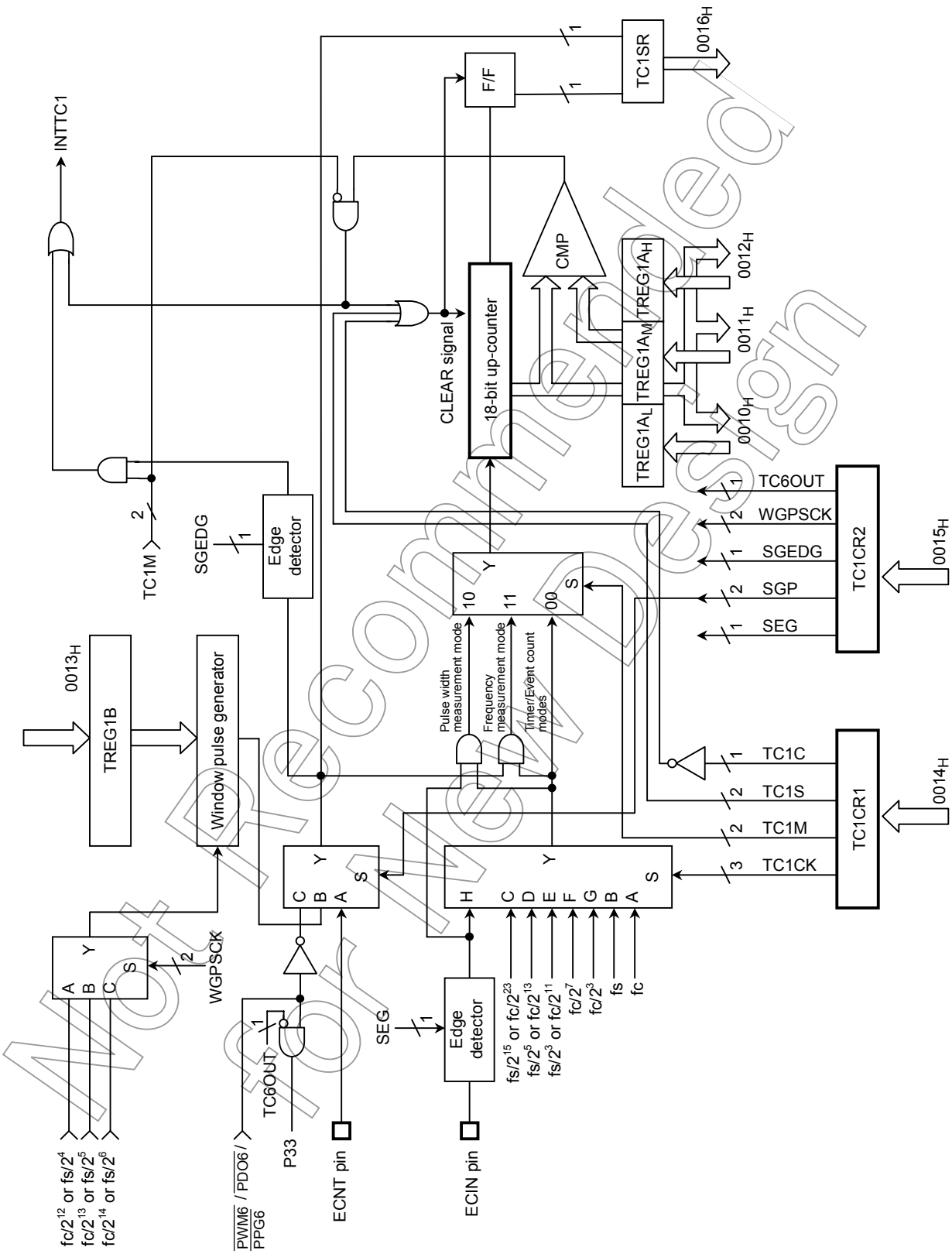


Figure 2.7.1 Timer/Counter 1

2.7.2 Control

The Timer/counter 1 is controlled by timer/counter 1 control registers (TC1CR1/TC1CR2), an 18-bit timer register (TREG1A), and an 8-bit internal window gate pulse setting register (TREG1B).

Timer register

17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$TREG1A_H$ (0012_H)		$TREG1A_M$ (0011_H)								$TREG1A_L$ (0010_H)							

TREG1A (0012_H, 0011_H, 0010_H) Read/Write (Initial value: 00 0000 0000 0000 0000)

Internal window gate pulse setting register

TREG1B (0013_H)

7	6	5	4	3	2	1	0
Ta				Tb			

(Initial value: 0000 0000)

Ta	Setting "H" level period of the window gate pulse	$(16 - Ta) \times 2^{13}$/fc or $(16 - Ta) \times 2^5$/fs [s]	R/W
Tb	Setting "L" level period of the window gate pulse	$(16 - Tb) \times 2^{13}$/fc or $(16 - Tb) \times 2^5$/fs [s]	

Note 1: WGPSCK = 01

Timer/counter 1 control register 1

TR1CR1 (0014_H)

7	6	5	4	3	2	1	0
TC1C	TC1S			TC1CK		TC1M	

(Initial value: 1000 1000)

TC1C		0: Counter/overflow flag clear request ("1" is automatically set after clearing.)	R/W
TC1S	TC1 start control	00: Stop and counter clear and overflow flag clear 10: Start *1: Reserved	
TC1CK	TC1 source clock select	000: Internal clock fc [Hz] 001: Internal clock fs [Hz] 010: Internal clock $fc/2^{23}$ or $fs/2^{15}$ [Hz] 011: Internal clock $fc/2^{13}$ or $fs/2^5$ [Hz] 100: Internal clock $fc/2^{11}$ or $fs/2^3$ [Hz] 101: Internal clock $fc/2^7$ [Hz] 110: Internal clock $fc/2^3$ [Hz] 111: External clock (ECIN pin input)	
TC1M	TC1 mode select	00: Timer/Event counter mode 01: Reserved 10: Pulse width measurement mode 11: Frequency measurement mode	

Note 1: fc; High-frequency clock [Hz] fs; Low-frequency clock [Hz] *; Don't care

Note 2: Writing to the low-byte of the timer register 1A ($TREG1A_L$, $TREG1A_M$), the compare function is inhibited until the high-byte ($TREG1A_H$) is written.

Note 3: Set the mode and source clock, and edge (selection) when the TC1 stops (TC1S = 00).

Note 4: "fc" can be selected as the source clock only in the timer mode during SLOW mode and in the pulse width measurement mode during NORMAL 1/2 or IDLE 1/2 mode.

Note 5: When a read instruction is executed to the timer register (TREG1A), the counter immediate value, not the register set value, is read out. Therefore it is impossible to read out the written value of TREG1A. To read the counter value, the read instruction should be executed when the counter stops to avoid reading unstable value.

Note 6: Set the timer register (TREG1A) to ≥ 1.

Note 7: When using the timer mode and pulse width measurement mode, set TC1CK (TC1 source clock select) to internal clock.

Note 8: When using the event counter mode, set TC1CK (TC1 source clock select) to external clock.

Note 9: Because the read value is different from the written value, do not use read-modify-write instructions to TREG1A.

Note 10: $fc/2^7$ and $fc/2^3$ can not be used as source clock in SLOW/SLEEP mode.

Note 11: The read data of bits 7 to 2 in TREG1AH are always "0". (Data "1" can not be written.)

Figure 2.7.2 Timer register/window gate pulse setting register/control register of the TC1

Timer/Counter 1 control register 2

TC1CR2	7	6	5	4	3	2	1	0	
(0015_H)	SEG	SGP		SGEDG	WGPSCK		TC6OUT	"0"	(Initial value: 0000 000*)

SEG	External input clock (ECIN) edge select	0: Counts at the falling edge 1: Counts at the falling/rising edges	R/W
SGP	Window gate pulse select	00: ECNT input 01: Internal window gate pulse (TREG1B) 10: $\overline{PWM6}$ / $\overline{PDO6}$ / $\overline{PPG6}$ (TC6) output 11: Reserved	
SGEDG	Window gate pulse interrupt edge select	0: Interrupts at the falling edge 1: Interrupts at the falling/rising edges	
WGPSCK	Window gate pulse source clock select	00: Select 2^{12}/fc or 2^4/fs 01: Select 2^{13}/fc or 2^5/fs 10: Select 2^{14}/fc or 2^6/fs 11: Reserved	
TC6OUT	TC6 output ($\overline{PWM6}$ / $\overline{PDO6}$ / $\overline{PPG6}$) external output select	0: Output to P33 1: No output to P33	

Note 1: fc; High-frequency clock [Hz] fs; Low-frequency clock [Hz] *; Don't care
Note 2: Set the mode, source clock, and edge (selection) when the TC1 stops (TC1S = 00).
Note 3: If there is no need to use $\overline{PWM6}$ / $\overline{PDO6}$ / $\overline{PPG6}$ as window gate pulse of TC1, always write "0" to TC6OUT.
Note 4: Make sure to write "0" to bit 0 in TC1CR2.

TC1 status register

TC1SR	7	6	5	4	3	2	1	0	
(0016_H)	HECF	HEOVF	"0"	"0"	"0"	"0"	"0"	"0"	(Initial value: 0000 0000)

HECF	Operating Status monitor	0: Stop (during Tb) or disable 1: Under counting (during Ta)	Read only
HEOVF	Counter overflow monitor	0: No overflow 1: Overflow status	

Figure 2.7.3 Control register of the TC1/status register

2.7.3 Function

TC1 has four operating modes. The timer mode of the TC1 is used at warm-up when switching form SLOW mode to NORMAL2 mode.

(1) Timer mode

In this mode, counting up is performed using the internal clock. The contents of TREGIA are compared with the contents of up-counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared. Counting up resumes after the counter is cleared.

Table 2.7.1 Source clock (internal clock) of Timer/Counter 1

Source Clock				Resolution		Maximum Time Setting	
NORMAL1/2, IDLE1/2 Mode		SLOW Mode	SLEEP Mode	fc = 16 MHz	fs = 32.768 kHz	fc = 16 MHz	fs = 32.768 kHz
DV7CK = 0	DV7CK = 0						
$fc/2^{23}$ [Hz]	$fs/2^{15}$ [Hz]	$fc/2^{15}$ [Hz]	$fc/2^{15}$ [Hz]	0.52 [s]	1 [s]	38.2 [h]	72.8 [h]
$fc/2^{13}$	$fs/2^{5}$	$fc/2^{5}$	$fc/2^{5}$	512 [μs]	0.98 [ms]	2.2 [min]	4.3 [min]
$fc/2^{11}$	$fs/2^{3}$	$fc/2^{3}$	$fc/2^{3}$	128 [μs]	244 [μs]	0.6 [min]	1.07 [min]
$fc/2^{7}$	$fc/2^{7}$	–	–	8 [μs]	–	2.1 [s]	–
$fc/2^{3}$	$fc/2^{3}$	–	–	0.5 [μs]	–	131.1 [ms]	–
fc	fc	fc (Note)	–	62.5 [ns]	–	16.4 [ms]	–
fs	fs	–	–	–	30.5 [μs]	–	8 [s]

Note: When fc is selected for the source clock in SLOW mode, the lower bits 11 of TREG1A is invalid, and a match of the upper bits 7 makes interrupts.

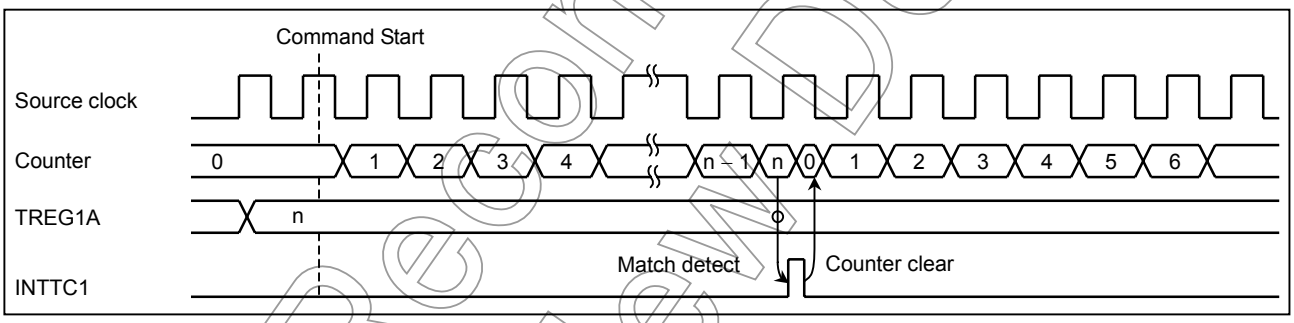


Figure 2.7.4 Timing chart for timer mode

(2) Event Counter mode

It is a mode to count up at the falling edge of the ECIN pin input. Both edges can not be used. The countents of TREG1A are compared with the contents of up-counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared. Counting up resums for ECIN pin input edge each after the counter is cleared. The maximum applied frequency is $fc/2^4$ [Hz] in NORMAL 1/2 or IDLE 1/2 mode and $fs/2^4$ [Hz] in SLOW or SLEEP mode. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width.

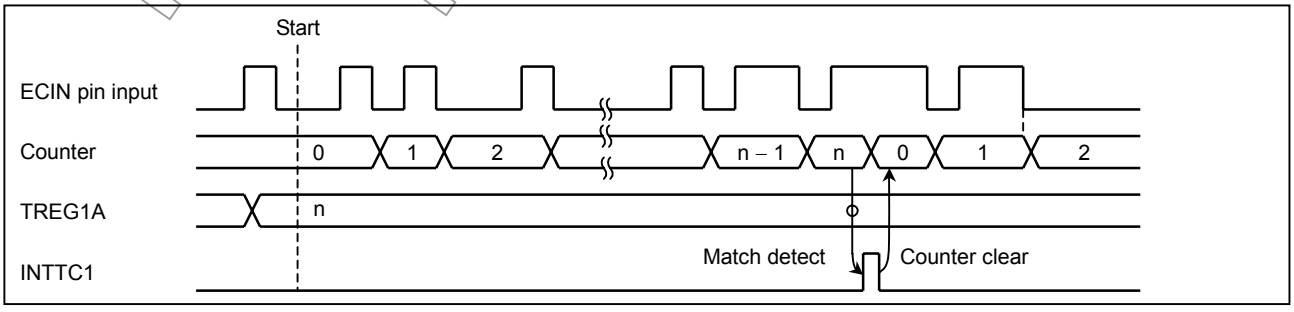


Figure 2.7.5 Pulse width measurement mode timing chart

(3) Pulse Width Measurement mode

In this mode, pulse widths are counted on the rising edge of logical AND-ed product between ECIN pin input (window pulse) and the internal clock. The internal clock is selected by TC1CK (bit 2, 3 and 4 in TC1CR1). An INTTC1 interrupt is generated at the falling edge of the window pulse or both rising and falling edges of the window pulse, that can be selected by SGEDG (bit 4 in TC1CR2). In the interrupt service program, read the contents of TREG1A while the count is stopped (ECIN pin is low), then clear the counter using TC1C (bit 7 in TC1CR1). When the counter is not cleared, counting up resumes by starting count-up. When TREG1A is counted up from $3FFFF_H$ to 00000_H, an overflow occurs. HEOVF (bit 6 in TC1SR) of the status register can monitor whether the overflows or not. HEOVF remains the old data until the counter is required to be cleared by TC1C.

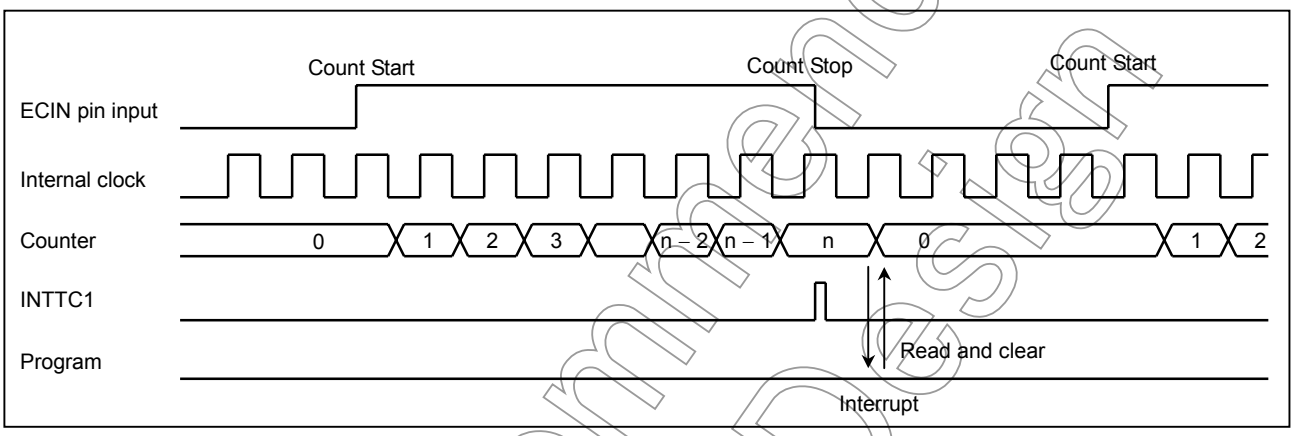


Figure 2.7.6 Pulse width measurement mode timing chart

Note 1: INTTC1 interrupt occurs when ECIN input is “1” and TC1S of TC1CR1 is written to “00”. According to the following step, when timer counter is stopped, INTTC1 interrupt latch should be cleared to “0”.

```
TC1STOP:
    ⋮          ⋮
    DI                          ;  Clear IMF
    CLR    (EIRH). EF8          ;  Clear EF8
    LD     (TC1CR1), 00011010B  ;  Stop timer couter 1
    LD     (ILH), 11111110B     ;  Clear IL8
    SET    (EIRH). EF8          ;  Set EF8
    EI                          ;  Set IMF
    ⋮          ⋮
```

Note 2: When SGEDG (window gate pulse interrupt edge select) is set to both edges and ECIN pin input is “1” in the pulse width measurement mode, an INTTC1 interrupt is generated by setting TC1S (TC1 start control) to “10” (start).

Note 3: In the pulse width measurement mode, HECF (operating status monitor) cannot used.

(4) Frequency Measurement mode

In this mode, the frequency of ECIN pin input pulse is measured. TC1CK is required to be set to the external clock. The edge of the input pulse is counted during "H" level of the window gate pulse selected by SGP (bit 5 and 6 in TC1CR2). Whether the input pulse is counted on the falling edge. An INTTC1 interrupt is generated on the falling edge or both the rising/falling edges of the window gate pulse, that can be selected by SGEDG (bit 4 in TC1CR2). To use ECNT terminal input as a window gate pulse, SGP (bit 5 and 6 in TC1CR2) should be set to "00". In the interrupt service program, read the contents of TREG1A while the count is stopped (window gate pulse is low), then clear the counter using TC1C. When the counter is not cleared, counting up resumes by stating count-up. The window pulse status can be monitored by HECF of the status register. HEOVF of the status register can monitor whether the binary counter overflows or not. In the overflow flag status, a new data is not input until the counter clear requests.

- Using TC6 output ($\overline{PWM6}$ / $\overline{PDO6}$ / $\overline{PPG6}$) for the window gate pulse, external output of $\overline{PWM6}$ / $\overline{PDO6}$ / $\overline{PPG6}$ to P33 can be controlled using TC6OUT (bit 1 in TC1CR2). Zero-clearing TC6OUT outputs $\overline{PWM6}$ / $\overline{PDO6}$ / $\overline{PPG6}$ to P33; setting 1 in TC6OUT does not output $\overline{PWM6}$ / $\overline{PDO6}$ / $\overline{PPG6}$ to P33. (TC6OUT is used to control output to P33 only. Thus, use the timer counter 6 control register to operate/stop $\overline{PWM6}$ / $\overline{PDO6}$ / $\overline{PPG6}$.)

- When the internal window gate pulse is selected, the window gate pulse is set as follows. The internal window gate pulse consists of "H" level period (Ta) that is counting time and "L" level period (Tb) that is counting stop time. Ta or Tb can be individually set by TREG1B. One cycle contains Ta + Tb.

  Note 1: Because the internal window gate pulse is generated in synchronization with the internal divider, it may be delayed for a maximum of one cycle of the source clock (WGPSCK) immediately after start of the timer.

  Note 2: Set the internal window gate pulse when the timer counter is not operating or during the Tb period. When Tb is overwritten during the Tb period, the update is valid from the next Tb period.

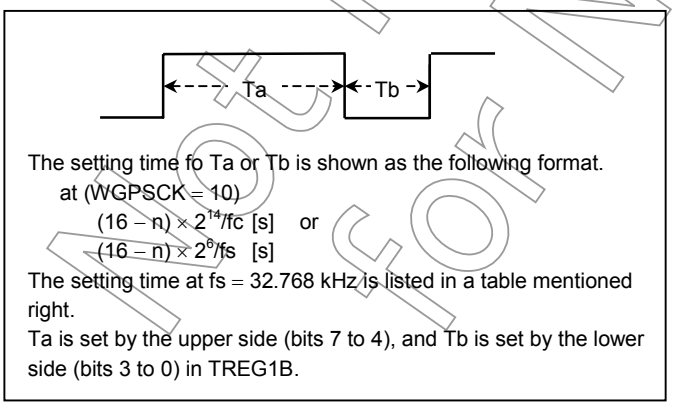


Figure 2.7.7 Window gate pulse format

Table 2.7.2 Setting Ta and Tb

(WGPSCK = 10, fs = 32.768 kHz)

Setting Value	Setting time	Setting Value	Setting time
0	31.25 ms	8	15.63 ms
1	29.30 ms	9	13.67 ms
2	27.34 ms	A	11.72 ms
3	25.39 ms	B	9.77 ms
4	23.44 ms	C	7.81 ms
5	21.48 ms	D	5.86 ms
6	19.53 ms	E	3.91 ms
7	17.58 ms	F	1.95 ms

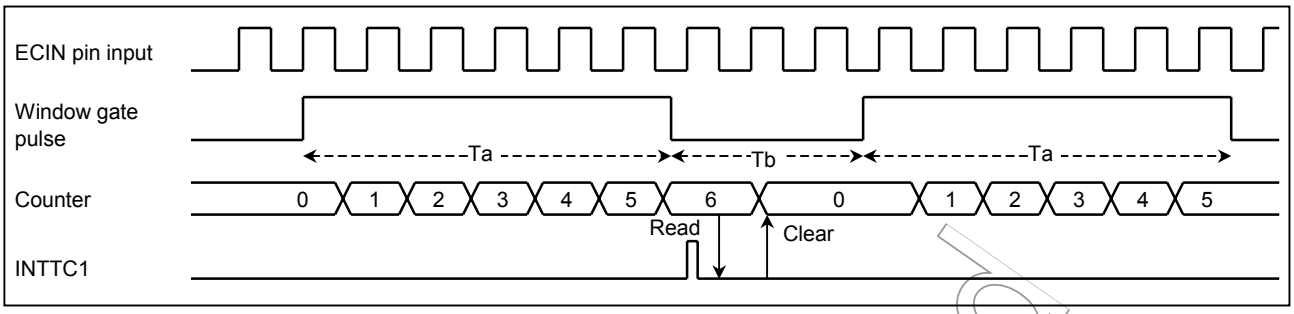


Figure 2.7.8 Timing chart for the frequency measurement mode
(ECIN falling edge count, window gate pulse falling interrupt)

2.8 8-Bit Timer/Counter (TC3, 4, 5, 6)

The TMP86FM29 has four channels of 8-bit timer/counter (TC3, 4, 5, 6). These timer/counter are used as timer, event counter, PWM, PPG and PDO. These are also available as a 16-bit timer/counter by cascade connection.

2.8.1 Configuration

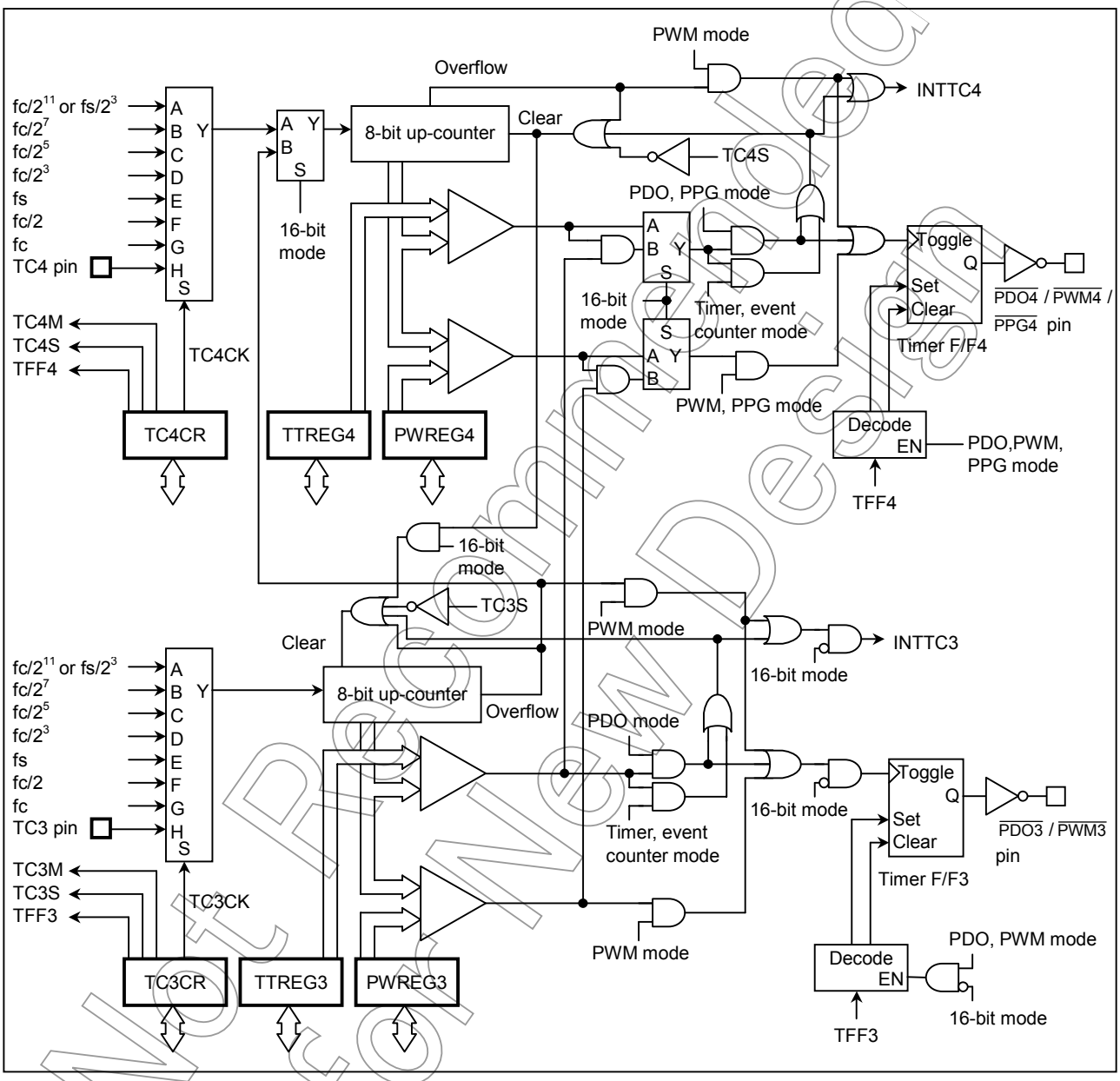


Figure 2.8.1 8-bit timer 3, 4

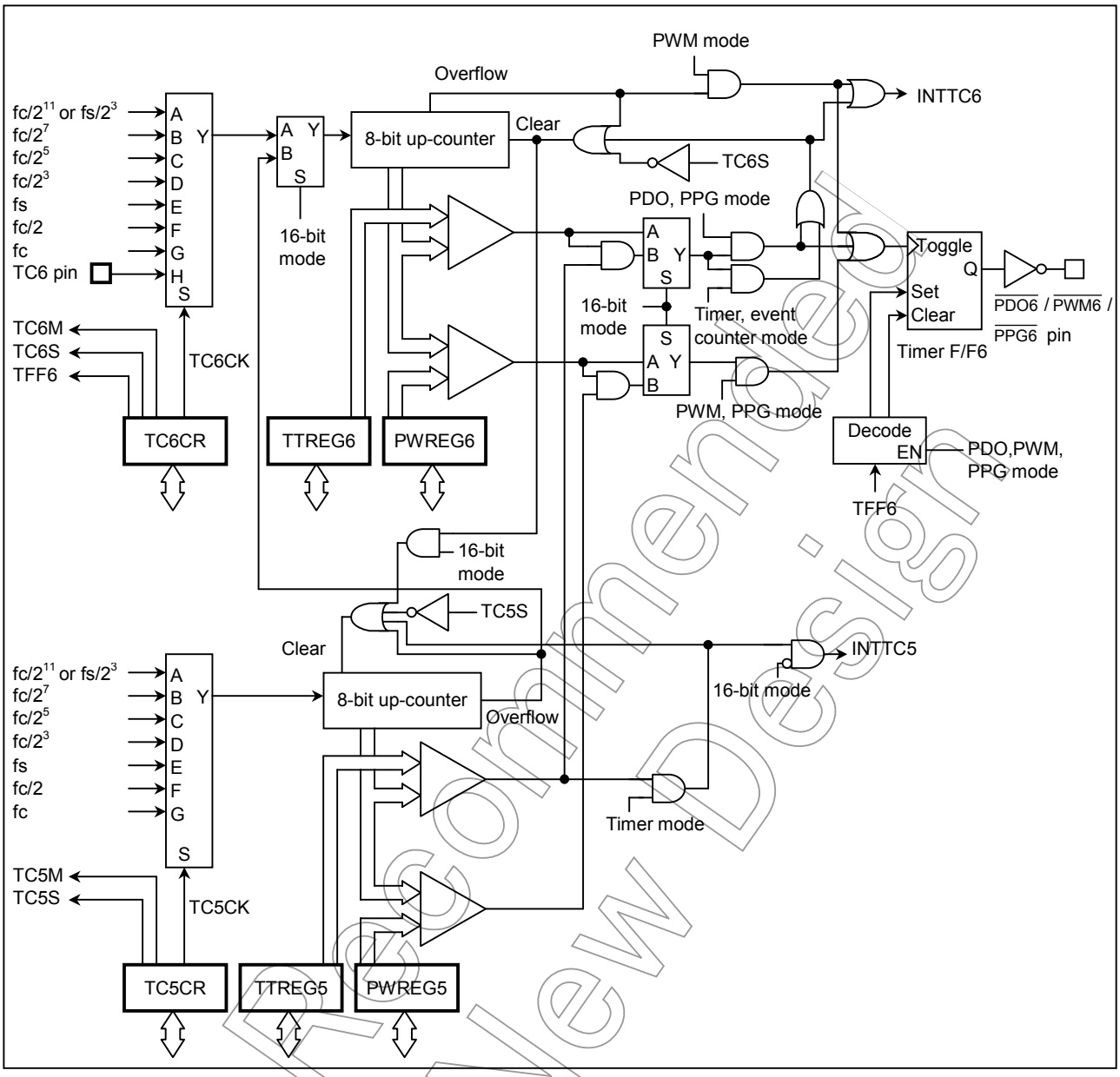


Figure 2.8.2 8-bit timer 5, 6

2.8.2 Control

The timer/counter 3 is controlled by a timer/counter 3 control register (TC3CR) and two 8-bit timer registers (TTREG3 and PWREG3).

Timer Register

TTREG3 ($001C_H$) R/W — 7 6 5 4 3 2 1 0 (Initial value: 1111 1111)

PWREG3 ($002C_H$) R/W — 7 6 5 4 3 2 1 0 (Initial value: 1111 1111)

Note 1: Do not change the timer register (TTREG3) while timer/counter is operating.

Note 2: Do not change the timer register (PWREG3) while timer/counter is operating, except 8-bit PWM and 16-bit PWM mode.

Timer/counter 3 control register

TC3CR (0018_H) — 7 TFF3 | 6 5 4 TC3CK | 3 TC3S | 2 1 0 TC3M (Initial value: 0000 0000)

TFF3	Timer F/F3 control	0: Clear 1: Set				R/W
TC3CK	TC3 source clock select [Hz]		NORMAL1/2, IDLE1/2 Mode		SLOW1/2 SLEEP1/2 Mode	
			DV7CK = 0	DV7CK = 1		
		000	$fc/2^{23}$	$fs/2^3$	$fs/2^3$	
		001	$fc/2^7$	$fc/2^7$	–	
		010	$fc/2^5$	$fc/2^5$	–	
		011	$fc/2^3$	$fc/2^3$	–	
		100	fs	fs	fs	
		101	fc/2	fc/2	–	
		110	fc	fc	fc (Note 8)	
		111	TC3 pin input			
TC3S	TC3 start control	0: Stop and counter clear 1: Command start				
TC3M	TC3 operating mode select	000: 8-bit timer/event counter mode 001: 8-bit programmable divider output (PDO) mode 010: 8-bit pulse width modulation (PWM) mode 011: 16-bit mode (Mode selection is controlled by TC4M) 1**: Reserved				

Note 1: fc; High-frequency clock [Hz] fs; Low-frequency clock [Hz]

Note 2: During TC3 operation, do not change TC3M, TC3CK and TFF3.

Note 3: When TC3 operation is stopped (TC3S = "1" → "0"), do not change TC3M, TC3CK and TFF3. But it is possible to change TC3M, TC3CK and TFF3 at the start timing (TC3S = "0" → "1").

Note 4: When used as 16-bit mode, the operating mode is selected by TC4CR<TC4M>, and TC3M should be set to "011".

Note 5: When used as 16-bit mode, only the source clock is selected by TC3CK, and start of operation and control of F/F are controlled by TC4CR<TC4S> and TC4CR<TFF4>.

Note 6: Selecting source clock depends on the operating mode, refer to Table 2.8.1 and Table 2.8.2 for details.

Note 7: Value of timer register depends on the operating mode, refer to Table 2.8.3 for details.

Note 8: When used as the SLOW and SLEEP modes, the "fc" of TC3 source clock can use only "fc warming-up counter" mode.

Figure 2.8.3 Timer 3 register and timer/counter 3 control register

The timer/counter 4 is controlled by a timer/counter 4 control register (TC4CR) and two 8-bit timer registers (TTREG4 and PWREG4).

Timer register

TTREG4 ($001D_H$) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

PWREG4 ($002D_H$) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

Note 1: Do not change the timer register (TTREG4) while timer/counter is operating.

Note 2: Do not change the timer register (PWREG4) while timer/counter is operating, except 8-bit PWM and 16-bit PWM mode.

Timer/Counter 4 control register

TC4CR (0019_H)

7	6	5	4	3	2	1	0
TFF4	TC4CK			TC4S	TC4M		

(Initial value: 0000 0000)

TFF4	Timer F/F4 control	0: Clear 1: Set			R/W
TC4CK	TC4 source clock select [Hz]		NORMAL1/2, IDLE1/2 Mode		SLOW1/2 SLEEP1/2 Mode
			DV7CK = 0	DV7CK = 1	
		000	$fc/2^{11}$	$fs/2^3$	$fs/2^3$
		001	$fc/2^7$	$fc/2^7$	–
		010	$fc/2^5$	$fc/2^5$	–
		011	$fc/2^3$	$fc/2^3$	–
		100	fs	fs	fs
		101	fc/2	fc/2	–
		110	fc	fc	–
		111	TC4 pin input		
TC4S	TC4 start control	0: Stop and counter clear 1: Command start			
TC4M	TC4 operating mode select	000: 8-bit timer/event counter mode 001: 8-bit programmable divider output (PDO) mode 010: 8-bit pulse width modulation (PWM) mode 011: Reserved 100: 16-bit timer/event counter mode 101: Warming-up counter mode 110: 16-bit programmable divider output (PDO) mode 111: 16-bit programmable pulse generate (PPG) output mode			

Note 1: fc; High-frequency clock [Hz] fs; Low-frequency clock [Hz]

Note 2: During TC4 operation, do not change TC4M, TC4CK and TFF4.

Note 3: When TC4 operation is stopped (TC4S = "1" → "0") , do not change TC4M, TC4CK and TFF4. But it is possible to change TC4M, TC4CK and TFF4 at the start timing (TC4S = "0" → "1").

Note 4: When TC4M is selected to "1**" (16-bit mode), the source clock is automatically selected to the over-flowing signal of TC3 counter.

Note 5: When used as 16-bit mode, the operating mode is selected by TC4M, and TC3CR<TC3M> should be set to "011".

Note 6: When used as 16-bit mode, only the source clock is selected by TC3CR<TC3CK>, and start of operation and control of F/F are controlled by TC4S and TFF4.

Note 7: Selecting source clock depends on the operating mode, refer to Table 2.8.1 and Table 2.8.2 for details.

Note 8: Value of timer register depends on the operating mode, refer to Table 2.8.3 for details.

Figure 2.8.4 Timer 4 register and timer/counter 4 control register

The timer/counter 5 is controlled by a timer/counter 5 control register (TC5CR) and two 8-bit timer registers (TTREG5 and PWREG5).

Timer register

TTREG5 ($001E_H$) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

PWREG5 ($002E_H$) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

Note 1: Do not change the timer register (TTREG5) while timer/counter is operating.
Note 2: Do not change the timer register (PWREG5) while timer/counter is operating, except 16-bit PWM mode.

Timer/Counter 5 control register

TC5CR ($001A_H$)

7	6	5	4	3	2	1	0
TFF4	TC5CK			TC5S	TC5M		

(Initial value: ∗000 0000)

			NORMAL1/2, IDLE1/2 Mode		SLOW1/2 SLEEP1/2 Mode
			DV7CK = 0	DV7CK = 1	
TC5CK	TC5 source clock select [Hz]	000	$fc/2^{11}$	$fs/2^3$	$fs/2^3$
		001	$fc/2^7$	$fc/2^7$	–
		010	$fc/2^5$	$fc/2^5$	–
		011	$fc/2^3$	$fc/2^3$	–
		100	fs	fs	fs
		101	fc/2	fc/2	–
		110	fc	fc	fc (Note 8)
		111	Reserved		
TC5S	TC5 start control	0: Stop and counter clear 1: Command start			
TC5M	TC5 operating mode select	000: 8-bit timer mode 001: Reserved 010: Reserved 011: 16-bit mode (Mode selection is controlled by TC6M) 1∗∗: Reserved			

Note 1: fc; High-frequency clock [Hz] fs; Low-frequency clock [Hz]
Note 2: During TC5 operation, do not change TC5M and TC5CK.
Note 3: When TC5 operation is stopped (TC5S = "1" → "0"), do not change TC5M and TC5CK. But it is possible to change TC5M and TC5CK at the start timing (TC5S = "0" → "1").
Note 4: When used as 16-bit mode, the operating mode is selected by TC6CR<TC6M>, and TC5M should be set to "011".
Note 5: When used as 16-bit mode, only the source clock is selected by TC5CK, and start of operation and control of F/F are controlled by TC6CR<TC6S> and TC6CR<TFF6>.
Note 6: Selecting source clock depends on the operating mode, refer to Table 2.8.1 and Table 2.8.2 for details.
Note 7: Value of timer register depends on the operating mode, refer to Table 2.8.3 for details.
Note 8: When used as the SLOW and SLEEP modes, the "fc" of TC5 source clock can use only "fc warming-up counter" mode.

Figure 2.8.5 Timer 5 register and timer/counter 5 control register

The timer/counter 6 is controlled by a timer/counter 6 control register (TC6CR) and two 8-bit timer registers (TTREG6 and PWREG6).

Timer register

TTREG6 ($001F_H$) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

PWREG4 ($002F_H$) R/W

7	6	5	4	3	2	1	0

(Initial value: 1111 1111)

Note 1: Do not change the timer register (TTREG6) while timer/counter is operating.

Note 2: Do not change the timer register (PWREG6) while timer/counter is operating, except 8-bit PWM and 16-bit PWM mode.

Timer/Counter 6 control register

TC6CR ($001B_H$)

7	6	5	4	3	2	1	0
TFF6	TC6CK			TC6S	TC6M		

(Initial value: 0000 0000)

TFF6	Timer F/F6 control	0: Clear 1: Set			R/W
TC6CK	TC6 source clock select [Hz]		NORMAL1/2, IDLE1/2 Mode		SLOW1/2 SLEEP1/2 Mode
			DV7CK = 0	DV7CK = 1	
		000	$fc/2^{11}$	$fs/2^3$	$fs/2^3$
		001	$fc/2^7$	$fc/2^7$	–
		010	$fc/2^5$	$fc/2^5$	–
		011	$fc/2^3$	$fc/2^3$	–
		100	fs	fs	fs
		101	fc/2	fc/2	–
		110	fc	fc	–
		111	TC6 pin input		
TC6S	TC6 start control	0: Stop and counter clear 1: Command start			
TC6M	TC6 operating mode select	000: 8-bit timer/event counter mode 001: 8-bit programmable divider output (PDO) mode 010: 8-bit pulse width modulation (PWM) mode 011: Reserved 100: 16-bit timer/event counter mode 101: Warming-up counter mode 110: 16-bit programmable divider output (PDO) mode 111: 16-bit programmable pulse generate (PPG) output mode			

Note 1: fc; High-frequency clock [Hz] fs; Low-frequency clock [Hz]

Note 2: During TC6 operation, do not change TC6M, TC6CK and TFF6.

Note 3: When TC6 operation is stopped (TC6S = "1" → "0") , do not change TC6M, TC6CK and TFF6. But it is possible to change TC6M, TC6CK and TFF6 at the start timing (TC6S = "0" → "1").

Note 4: When TC6M is selected to "1**" (16-bit mode), the source clock is automatically selected to the over-flowing signal of TC5 counter.

Note 5: When used as 16-bit mode, the operating mode is selected by TC6M, and TC5CR<TC5M> should be set to "011".

Note 6: When used as 16-bit mode, only the source clock is selected by TC5CR<TC5CK>, and start of operation and control of F/F are controlled by TC6S and TFF6.

Note 7: Selecting source clock depends on the operating mode, refer to Table 2.8.1 and Table 2.8.2 for details.

Note 8: Value of timer register depends on the operating mode, refer to Table 2.8.3 for details.

Note 9: If there is no need to use $\overline{PDO6}$ / $\overline{PWM6}$ / $\overline{PPG6}$ as window gate pulse of TC1, always write "0" to TC6OUT.

Figure 2.8.6 Timer 6 register and timer/counter 6 control register

Table 2.8.1 Operating mode and available source clock
(NORMAL1/2, IDLE1/2 mode)

Operating Mode	$fc/2^{11}$ or $fc/2^{3}$	$fc/2^{7}$	$fc/2^{5}$	$fc/2^{3}$	fs	fc/2	fc	TCi pin input
8-Bit Timer	○	○	○	○	–	–	–	–
8-Bit Event Counter	–	–	–	–	–	–	–	○
8-Bit PDO	○	○	○	○	–	–	–	–
8-Bit PWM	○	○	○	○	○	○	○	–
16-Bit Timer	○	○	○	○	–	–	–	–
16-Bit Event Counter	–	–	–	–	–	–	–	○
Warming-up Counter	–	–	–	–	○	–	–	–
16-Bit PWM	○	○	○	○	○	○	○	○
16-Bit PPG	○	○	○	○	–	–	–	○

Note 1: For 16-bit operation (16-bit Timer/Event Counter, Warming-up Counter, 16-bit PWM and 16-bit PPG), set its source clock on lower bits (TC3CK, TC5CK).

Note 2: i = 3, 4, 6 (8-bit mode)
i = 3 (16-bit mode)

Table 2.8.2 Operating mode and available source clock
(under SLOW1/2 mode, SLEEP1/2 mode)

Operating Mode	$fc/2^{11}$ or $fc/2^{3}$	$fc/2^{7}$	$fc/2^{5}$	$fc/2^{3}$	fs	fc/2	fc	TCi pin input
8-Bit Timer	○	–	–	–	–	–	–	–
8-Bit Event Counter	–	–	–	–	–	–	–	○
8-Bit PDO	○	–	–	–	–	–	–	–
8-Bit PWM	○	–	–	–	○	–	–	–
16-Bit Timer	○	–	–	–	–	–	–	–
16-Bit Event Counter	–	–	–	–	–	–	–	○
Warming-up Counter	–	–	–	–	–	–	○	–
16-Bit PWM	○	–	–	–	○	–	–	○
16-Bit PPG	○	–	–	–	–	–	–	○

Note 1: For 16-bit operation (16-bit Timer/Event Counter, Warming-up Counter, 16-bit PWM and 16-bit PPG), set its source clock on lower bits (TC3CK, TC5CK).

Note 2: i = 3, 4, 6 (8-bit mode)
i = 3 (16-bit mode)

Table 2.8.3 Restriction against the rate for comparing registers

Operating Mode	Authorized Rate for Register
8-Bit Timer/Event Counter	$1 \leqq (TTREGn) \leqq 255$
8-Bit PDO	$1 \leqq (TTREGn) \leqq 255$
8-Bit PWM	$2 \leqq (PWREGn) \leqq 254$
16-Bit Timer/Event Counter	$1 \leqq (TTREG4, 3) \leqq 65535$, $1 \leqq (TTREG6, 5) \leqq 65535$
fc Warming-up Counter	$256 \leqq (TTREG4, 3) \leqq 65535$, $256 \leqq (TTREG6, 5) \leqq 65535$
16-Bit PWM	$2 \leqq (PWREG4, 3) \leqq 65534$, $2 \leqq (PWREG6, 5) \leqq 65534$
16-Bit PPG	$1 \leqq (PWREG4, 3) < (TTREG4, 3) \leqq 65535$ and $(PWREG4, 3) + 1 < (TTREG4, 3)$ $1 \leqq (PWREG6, 5) < (TTREG6, 5) \leqq 65535$ and $(PWREG6, 5) + 1 < (TTREG6, 5)$

Note: n = 3 to 6

2.8.3 Function

Timer/Counter 3, 4, 5 and 6 have eight operating modes: 8-bit timer, 8-bit external trigger timer, 8-bit programmable divider output mode, 8-bit pulse width modulation output mode, 16-bit timer, 16-bit external trigger timer, 16-bit pulse width modulation output mode, 16-bit programmable pulse generator output mode.

16-bit timer mode can use Timer counter 3 and 4 (5, 6) by cascade connection.

(1) 8-Bit Timer Mode (Timer/Counter 3, 4, 5 and 6)

In this mode, counting up is performed using the internal clock. The contents of TTREGi are compared with the contents of up-counter. If a match is found, an INTTCi interrupt is generated, and the counter is cleared to "0". Counting up resumes after the counter is cleared.

Note 1: In the timer mode, always write TCjCR<TFFj> to "0". If TFFj is set to "1", unexpected pulse may be output from $\overline{\text{PDOj}}$ / $\overline{\text{PWMj}}$ / $\overline{\text{PPGj}}$ pin.

Note 2: In the timer mode, do not change the setting of timer registers (TTREGi) while timer/counter is operating. Since TTREGi is configured as one-stage register, a newly set value is immediately reflected on the timer register.

Note 3: j = 3, 4, 6 i = 3 to 6

Table 2.8.4 Timer/Counter 1 source clock (internal clock)

Source Clock			Resolution		Maximum Time Setting	
NORMAL1/2, IDLE1/2 Modes		SLOW1/2, SLEEP1/2 Modes	At fc = 16 MHz	At fs = 32.768 kHz	At fc = 16 MHz	At fs = 32.768 kHz
DV7CK = 0	DV7CK = 0					
$fc/2^{11}$ [Hz]	$fs/2^3$ [Hz]	$fs/2^3$ [Hz]	128 [μs]	244.14 [μs]	32.6 [ms]	62.3 [ms]
$fc/2^7$	$fc/2^7$	–	8 [μs]	–	2.0 [ms]	–
$fc/2^5$	$fc/2^5$	–	2 [μs]	–	510 [μs]	–
$fc/2^3$	$fc/2^3$	–	500 [ns]	–	127.5 [μs]	–

Example: Sets the timer mode with source clock $fc/2^7$ [Hz] and generates an interrupt 80 μs later (at fc = 16 MHz).

```
LDW  (TTREG4), 0AH          ; Sets the timer register (80 μs ÷ 2^7/fc = 0A_H)
DI
SET  (EIRH). EF11           ; Enables INTTC4 interrupt
EI
LD   (TC4CR), 00010000B     ; Sets the 8-bit timer mode and source clock (fc/2^7)
LD   (TC4CR), 00011000B     ; Starts TC4
```

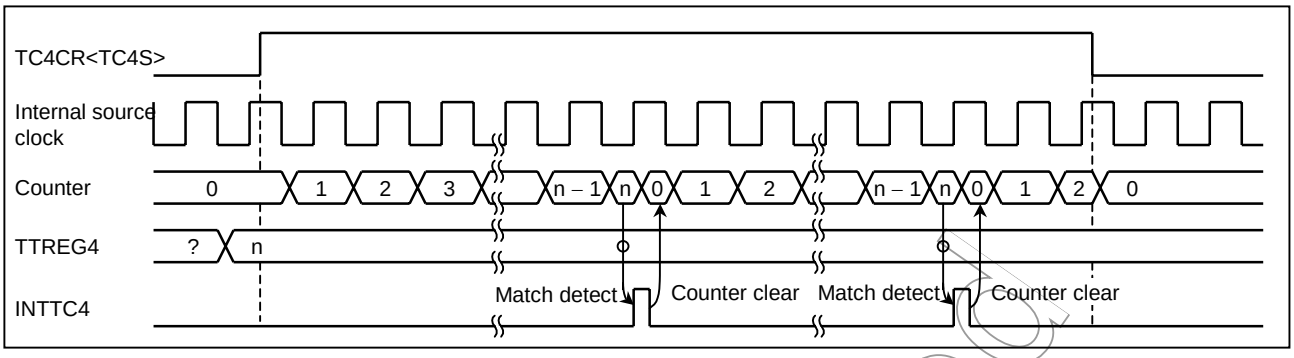


Figure 2.8.7 8-bit timer mode timing chart (in case of timer/counter 4)

(2) 8-Bit Event Counter Mode (Timer/Counter 3, 4 and 6)

In this mode, events are counted on the falling edge of TCj pin input. The contents of TTREGj are compared with the contents of up-counter. If a match is found, an INTTCj interrupt is generated, and the counter is cleared. The maximum applied frequency is fc/2^4 [Hz] in NORMAL1/2 or IDLE1/2 mode and fs/2^4 [Hz] in SLOW1/2 or SLEEP1/2 mode. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width.

Note 1: In the event counter mode, always write TCjCR<TFFj> to "0". If TFFj is set to "1", unexpected pulse may be output from $\overline{\text{PDOj}}$ / $\overline{\text{PWMj}}$ / $\overline{\text{PPGj}}$ pin.

Note 2: In the event counter mode, do not change the setting of timer registers (TTREGj) while timer/counter is operating. Since TTREGj is configured as one-stage register, a newly set value is immediately reflected on the timer register.

Note 3: j = 3, 4, 6

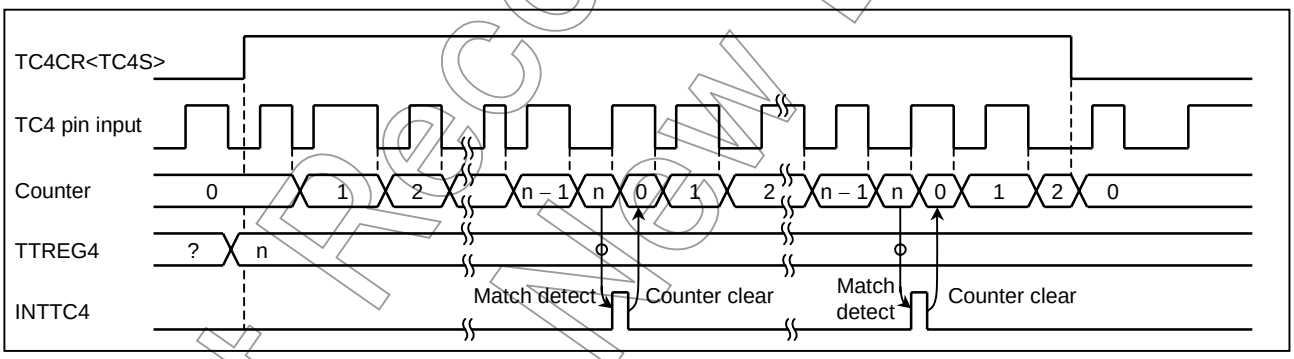


Figure 2.8.8 Event counter mode timing chart (in case of timer/counter 4)

(3) 8-Bit Programmable Divider Output (PDO) Mode (Timer/Counter 3, 4 and 6)

The internal clock is used for counting up. The contents of TTREGj are compared with the contents of the up-counter. Timer F/Fj output is toggled and the counter is cleared each time a match is found. Timer F/Fj output is inverted and output to the $\overline{PDOj}$ pin. When used as this mode, respective output latch should be set to "1". This mode can be used for 50% duty pulse output. Timer F/Fj can be initialized by program, and it is initialized to "0" during reset. An INTTCj interrupt is generated each time the PDOj output is toggled.

Example: Output a 1024 Hz pulse (at fc = 16.0 MHz = "0", in case of TC4)

```
LD    (TTREG4), 3DH        ; (1/1024 ÷ 2^7/fc) ÷ 2 = 3DH
LD    (TC4CR), 00010001B   ; Set the 8-bit PDO mode and source clock (fc/2^7)
LD    (TC4CR), 00011001B   ; Starts TC4
```

Note 1: In the programmable divider output(PDO) mode, do not change the setting of timer registers (TTREGj) while timer/counter is operating. Since TTREGj is configured as one-stage register, a newly set value is immediately reflected on the timer register.

Note 2: If PDO output is stopped during output operation, the output state is maintained at the state immediately before timer/counter is stopped. For changing the level of $\overline{PDOj}$ pin, modify TCjCR<TTFj> after timer/counter has been stopped. Do not execute halt of timer/counter and modification of TFFj simultaneously.

Example: Fixes $\overline{PDOj}$ output at high level after timer/counter is stopped

```
CLR   (TCjCR).3            ; Stops timer/counter.
CLR   (TCjCR).7            ; Sets PDOj output to high level output
```

Note 3: i = 3, 4, 6

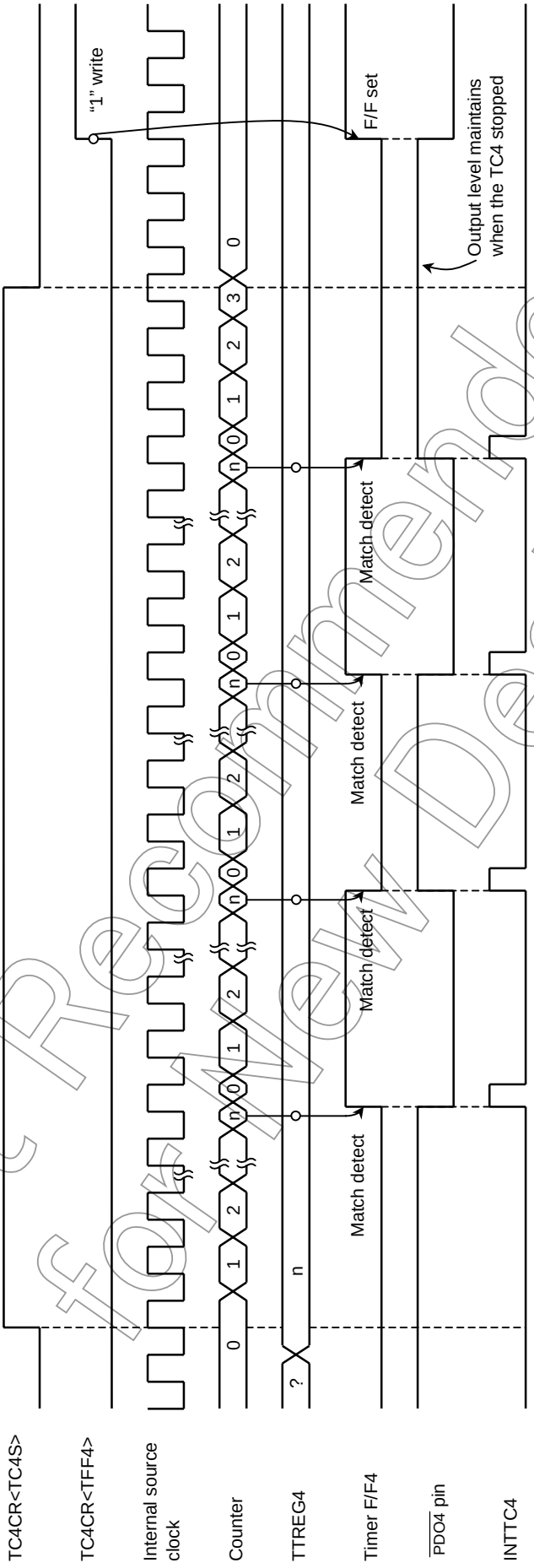


Figure 2.8.9 8-bit PDO mode timing chart (in case of timer/counter 4)

(4) 8-Bit Pulse Width Modulation (PWM) Output Mode (Timer/Counter 3, 4 and 6)

PWM output with a resolution of 8 bits is possible. The internal clock is used for counting up. The contents of PWREGi are compared with the contents of up-counter. If a match is found, the timer F/Fi output is toggled. The counter continues counting. And, when an overflow occurs, the timer F/Fi output is again toggled and the counter is cleared. Timer F/Fi output is inverted and output to the $\overline{\text{PWMi}}$ pin. An INTTCi interrupt is generated when an overflow occurs.

In PWM mode, because PWREGi becomes a 2-stage registers with shift register, it is possible to change the setting value of PWREGi while timer/counter is operating. Therefore, output can be altered continuously. The shift operation of PWREGi to shift register is executed at the INTTCi timing. While timer/counter is operating, the data by read instruction is not a setting value of PWREGi but a value of shift register. Thereofre, after writing to PWREGi, the reading data of PWREGi is previous value till INTTCi is generated.

While timer/counter stops, written value to PWREGi is shifted to shift register immediately.

Note 1: In PWM mode, write to the timer register PWREGi immediately after an INTTCi interrupt is generated (normally during the INTTCi interrupt service routine). If writing to PWREGi and INTTCi interrupt occur at the same time, the unstable value being written is shifted. This may cause pulses different from the set value to be output until the next INTTCi interrupt is generated.

Note 2: If PWM output is stopped during output operation, the output state is maintained at the state immediately before timer/counter is stopped. For changing the level of $\overline{\text{PWMi}}$, modify TCiCR<TTFi> after timer/counter has been stopped. Do not execute halt of timer/counter and modification of TFFi simultaneously.

Example: Fixes $\overline{\text{PWMi}}$ output at high level after timer/counter is stopped
CLR (TCiCR).3; Stops timer/counter.
CLR (TCiCR).7; Sets $\overline{\text{PWMi}}$ output to high level output

Note 3: Before starting STOP mode, disable PWM output. When the timer/counter is enabled and fc, fc/2 or fs is selected as the source clock, pulse is output from PWM pin during warming-up after releasing STOP mode.

Note 4: i = 3, 4, 6

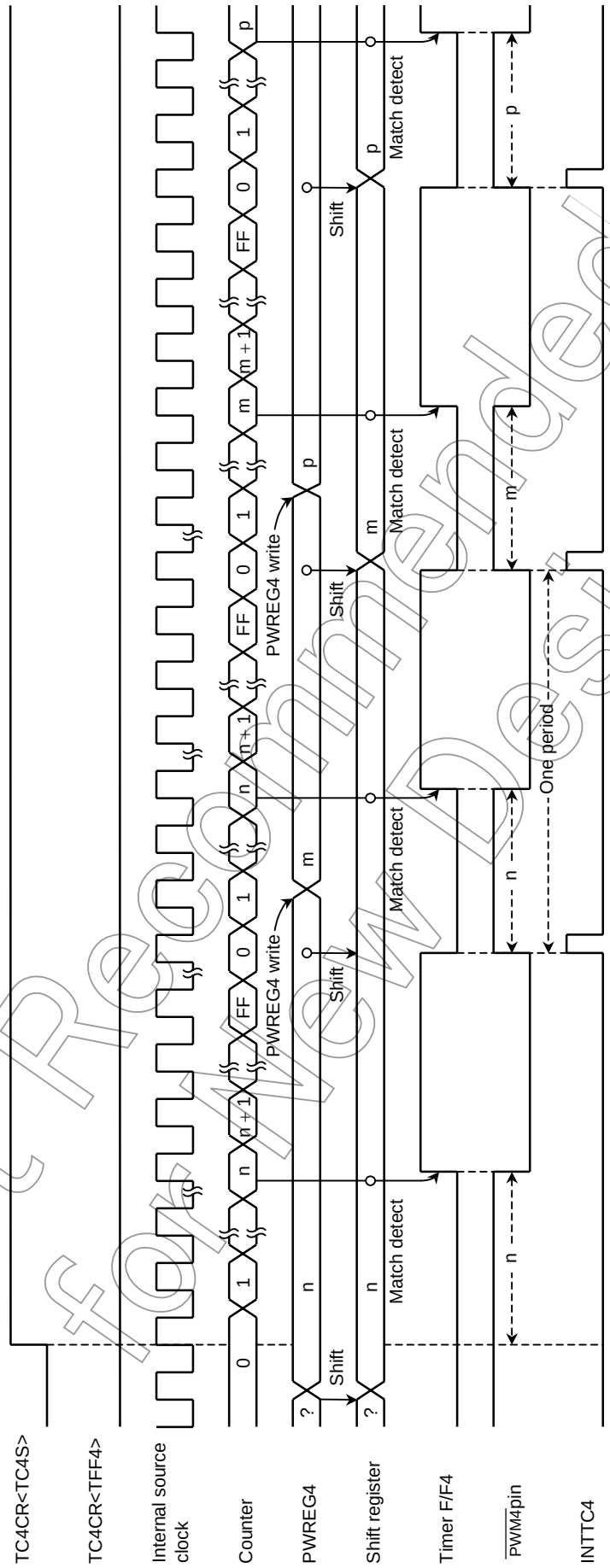


Figure 2.8.10 8-bit PWM mode timing chart (in case of timer/counter 4)

Table 2.8.5 PWM output mode

Source Clock			Resolution		Maximum Setting Time	
NORMAL1/2, IDLE1/2 Mode		SLOW1/2, SLEEP1/2 Mode	fc = 16 MHz	fs = 32.768 kHz	fc = 16 MHz	fs = 32.768 kHz
DV7CK = 0	DV7CK = 1					
$fc/2^{11}$ [Hz]	$fs/2^3$ [Hz]	$fs/2^3$ [Hz]	128 [μs]	244.14 [μs]	32.8 [ms]	62.5 [ms]
$fc/2^7$	$fc/2^7$	–	8 [μs]	–	2.05 [ms]	–
$fc/2^5$	$fc/2^5$	–	2 [μs]	–	512 [μs]	–
$fc/2^3$	$fc/2^3$	–	500 [ns]	–	128 [μs]	–
fs	fs	fs	30.5 [μs]	30.5 [μs]	7.81 [ms]	78.1 [ms]
fc/2	fc/2	–	125 [ns]	–	32 [μs]	–
fc	fc	–	62.5 [ns]	–	16 [μs]	–

(5) 16-Bit Timer Mode (Timer/counter 3 and 4, Timer/counter 5 and 6)

In this mode, counting up is performed using the internal clock.

Timer/counter 3 and 4 (5 and 6) are also available as a 16-bit timer mode by cascade connection.

a. 16-bit timer mode of Timer/counter 3 and 4

If a match is found, the INTTC4 interrupt is generated and the counter is cleared to "0". Counting up resumes after the counter is cleared. The timer register should write to the TTREG3 more first than TTREG4. The timer register must not write only either TTREG3 or TTREG4.

b. 16-bit timer mode of Timer/counter 5 and 6

If a match is found, the INTTC6 interrupt is generated and the counter is cleared to "0". Counting up resumes after the counter is cleared. The timer register should write to the TTREG5 more first than TTREG6. The timer register must not write only either TTREG5 or TTREG6.

Note 1: In the timer mode, always write TCjCR<TFFj> to "0".If TFFj is set to "1", unexpected pulse may be output from $\overline{PDOj}$ / $\overline{PWMj}$ / $\overline{PPGj}$ pin.

Note 2: In the timer mode, do not change the setting of timer registers (TTREGi) while timer/counter is operating. Since TTREGi is configured as one-stage register, a newly set value is immediately reflected on the timer register.

Note 3: j = 3, 4, 6 i = 3 to 6

Table 2.8.6 Source clock of 16-bit timer mode

Source Clock			Resolution		Maximum Setting Time	
NORMAL1/2, IDLE1/2 Mode		SLOW1/2, SLEEP1/2 Mode	fc = 16 MHz	fs = 32.768 kHz	fc = 16 MHz	fs = 32.768 kHz
DV7CK = 0	DV7CK = 1					
$fc/2^{11}$ [Hz]	$fs/2^3$ [Hz]	$fs/2^3$	128 [μs]	244.14 [μs]	8.39 [s]	16 [s]
$fc/2^7$	$fc/2^7$	–	8 [μs]	–	524.3 [ms]	–
$fc/2^5$	$fc/2^5$	–	2 [μs]	–	131.1 [μs]	–
$fc/2^3$	$fc/2^3$	–	500 [ns]	–	32.8 [μs]	–

Example: Set the 16-bit timer mode with source clock fc/2^7 [Hz] and generates an interrupt 300 [ms] later (at fc = 16 [MHz])

```
LDW  (TTREG3), 927CH    ; Sets the timer register
                          (300 ms ÷ 2^7/fc = 927C_H)
DI
SET  (EIRH). EF11       ; Enable INTTC4 interrupt
EI
LD   (TC3CR), 13H       ; Sets the 16-bit timer mode (lower) and
                          source clock
LD   (TC4CR), 04H       ; Sets the 16-bit timer mode (upper)
LD   (TC4CR), 0CH       ; Starts timer/counter
```

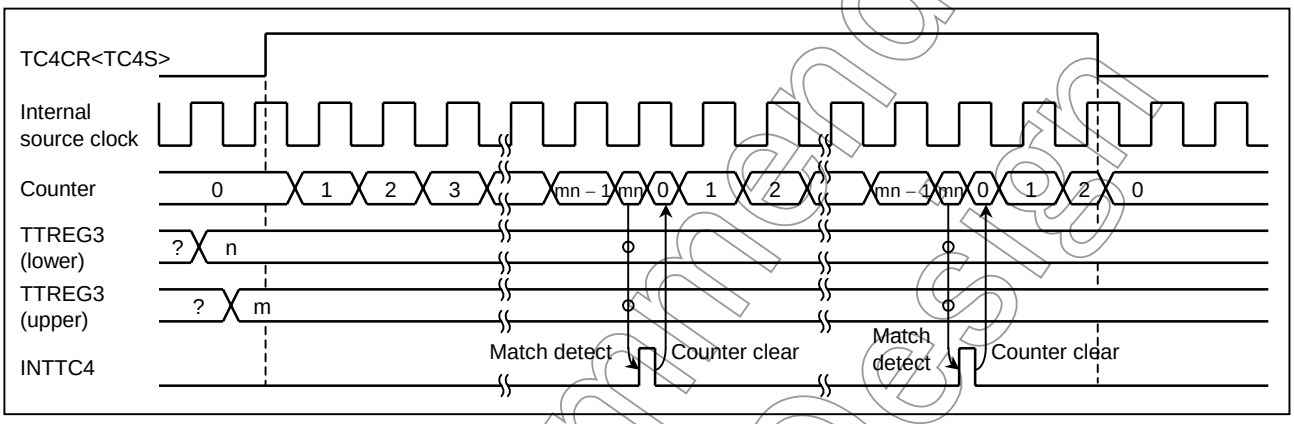


Figure 2.8.11 16-bit timer mode timing chart (in case of timer/counter 3 and 4)

(6) 16-Bit Event Counter Mode (Timer/counter 3 and 4)

In this mode, event are counted on the falling edge of the TC3 pin input. Timer/counter 5 and 6 can not use a 16-bit Event Counter Mode. Timer/counter 3 and 4 are also available as a 16-bit Event counter mode by cascade connection.

a. 16-bit event counter mode of Timer/counter 3 and 4

If a match is found, the INTTC4 interrupt is generated and the counter is cleared to "0". After the counter is cleared, counting up resumes every falling edge of TC3 input. The maximum applied frequency is fc/2^4 [Hz] in NORMAL1/2 or IDLE1/2 mode and fs/2^4 [Hz] in SLOW1/2 or SLEEP1/2 mode. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width. The timer register should write to the TTREG3 more first than TTREG4. The timer register must not write only either TTREG3 or TTREG4.

Note 1: In the event counter mode, always write TCjCR<TFFj> to "0". If TFFj is set to "1", unexpected pulse may be output from $\overline{\text{PDOj}}$ / $\overline{\text{PWMj}}$ / $\overline{\text{PPGj}}$ pin.

Note 2: In the event counter mode, do not change the setting of timer registers (TTREGj) while timer/counter is operating. Since TTREGj is configured as one-stage register, a newly set value is immediately reflected on the timer register.

Note 3: j = 3, 4

(7) 16-Bit Pulse Width Modulation (PWM) Output Mode
(Timer/counter 3 and 4, Timer/counter 5 and 6)

PWM output with a resolution of 16 bits is possible. Timer/counter 3 and 4 (5 and 6) are also available as a 16-bit PWM output mode by cascade connection.

a. 16-bit PWM output mode of Timer/counter 3 and 4

The contents of PWREG3/4 are compared with the contents of up-counter. If a match is found, the timer F/F4 output is toggled. The counter continues counting. And, when an overflow occurs, the timer F/F4 output is again toggled and the counter is cleared. Timer F/F4 output is inverted and output to the $\overline{\text{PWM4}}$ pin. An INTTC4 interrupt is generated when an overflow occurs. When used as $\overline{\text{PWM4}}$ pin, respective output latch should be set to "1". In PWM mode, because PWREG4/3 each becomes a 2-stage registers with shift register, it is possible to change the setting value of PWREG4/3 while timer/counter is operating. Therefore, output can be altered continuously. The shift operation of PWREG4/3 to shift register is executed at the INTTC4 timing. While timer/counter is operating, the data by read instruction is not a setting value of PWREG4/3 but a value of shift register. Therefore, after writing to PWREG4/3, the reading data of these registers is previous value till INTTC4 is generated.

While timer/counter stops, written value to PWREG4/3 is shifted to shift register immediately. When writing to PWREG4/3, always write to the lower side (PWREG3) and then the upper side (PWREG4) in that order. Writing to only lower side (PWREG3) or the upper side (PWREG4) has no effect.

b. 16-bit PWM output mode of Timer/counter 5 and 6

The contents of PWREG5/6 are compared with the contents of up-counter. If a match is found, the timer F/F6 output is toggled. The counter continues counting. And, when an overflow occurs, the timer F/F6 output is again toggled and the counter is cleared. Timer F/F6 output is inverted and output to the $\overline{\text{PWM6}}$ pin. An INTTC6 interrupt is generated when an overflow occurs. When used as $\overline{\text{PWM6}}$ pin, respective output latch should be set to "1". In PWM mode, because PWREG6/5 each becomes a 2-stage registers with shift register, it is possible to change the setting value of PWREG6/5 while timer/counter is operating. Therefore, output can be altered continuously. The shift operation of PWREG6/5 to shift register is executed at the INTTC6 timing. While timer/counter is operating, the data by read instruction is not a setting value of PWREG6/5 but a value of shift register. Therefore, after writing to PWREG6/5, the reading data of these registers is previous value till INTTC6 is generated.

While timer/counter stops, written value to PWREG6/5 is shifted to shift register immediately. When writing to PWREG6/5, always write to the lower side (PWREG5) and then the upper side (PWREG6) in that order. Writing to only lower side (PWREG5) or the upper side (PWREG6) has no effect.

Note 1: In PWM mode, write to the timer register PWREGm,n immediately after an INTTCm interrupt is generated (normally during the INTTCm interrupt service routine). If writing to PWREGm, n and INTTCm interrupt occur at the same time, the unstable value being written is shifted. This may cause pulses different from the set value to be output until the next INTTCm interrupt is generated.

Note 2: If PWM output is stopped during output operation, the output state is maintained at the state immediately before timer/counter is stopped. For changing the level of $\overline{PWMi}$, modify TCiCR<TTFi> after timer/counter has been stopped. Do not execute halt of timer/counter and modification of TFFi simultaneously.

Example: Fixes $\overline{PWMi}$ output at high level after timer/counter is stopped
CLR (TCiCR).3; Stops timer/counter
CLR (TCiCR).7; Sets $\overline{PWMi}$ output to high level output

Note 3: Before starting STOP mode, disable PWM output. When the timer/counter is enabled and fc, fc/2 or fs is selected as the source clock, pulse is output from PWM pin during warming-up after releasing STOP mode.

Note 4: m = 4 and n = 3, or m = 6 and n = 5. i = 4, 6.

Table 2.8.7 16-bit PWM output mode

Source Clock			Resolution		Maximum Setting Time	
NORMAL1/2, IDLE1/2 Mode		SLOW, SLEEP Mode	fc = 16 MHz	fs = 32.768 kHz	fc = 16 MHz	fs = 32.768 kHz
DV7CK = 0	DV7CK = 1					
$fc/2^{11}$	$fs/2^3$ [Hz]	$fs/2^3$	128 [μs]	244.14 [μs]	8.39 [s]	16 [s]
$fc/2^7$	$fc/2^7$	–	8 [μs]	–	524.3 [ms]	–
$fc/2^5$	$fc/2^5$	–	2 [μs]	–	131.1 [ms]	–
$fc/2^3$	$fc/2^3$	–	500 [ns]	–	32.8 [ms]	–
fs	fs	fs	30.5 [μs]	30.5 [μs]	2 [s]	2 [s]
fc/2	fc/2	–	125 [ns]	–	8.2 [ms]	–
fc	fc	–	62.5 [ns]	–	4.1 [ms]	–

Example: Extract the pulse, whose term and "high" width is 32.768 ms and 1 ms respectively, from P32 width 16-bit PWM mode
(at fc = 16 MHz = "0", DV7CK = 0)

```
LDW  (PWREG3), 07D0H   ; Sets pulse width
LD   (TC3CR), 33H      ; Sets the 16-bit PWM mode (lower) and
                         source clock (fc/2^3)
LD   (TC4CR), 056H     ; Sets the TFF4 to "0" and sets the 16-bit
                         PWM mode (upper)
LD   (TC4CR), 05EH     ; Starts timer/counter
```

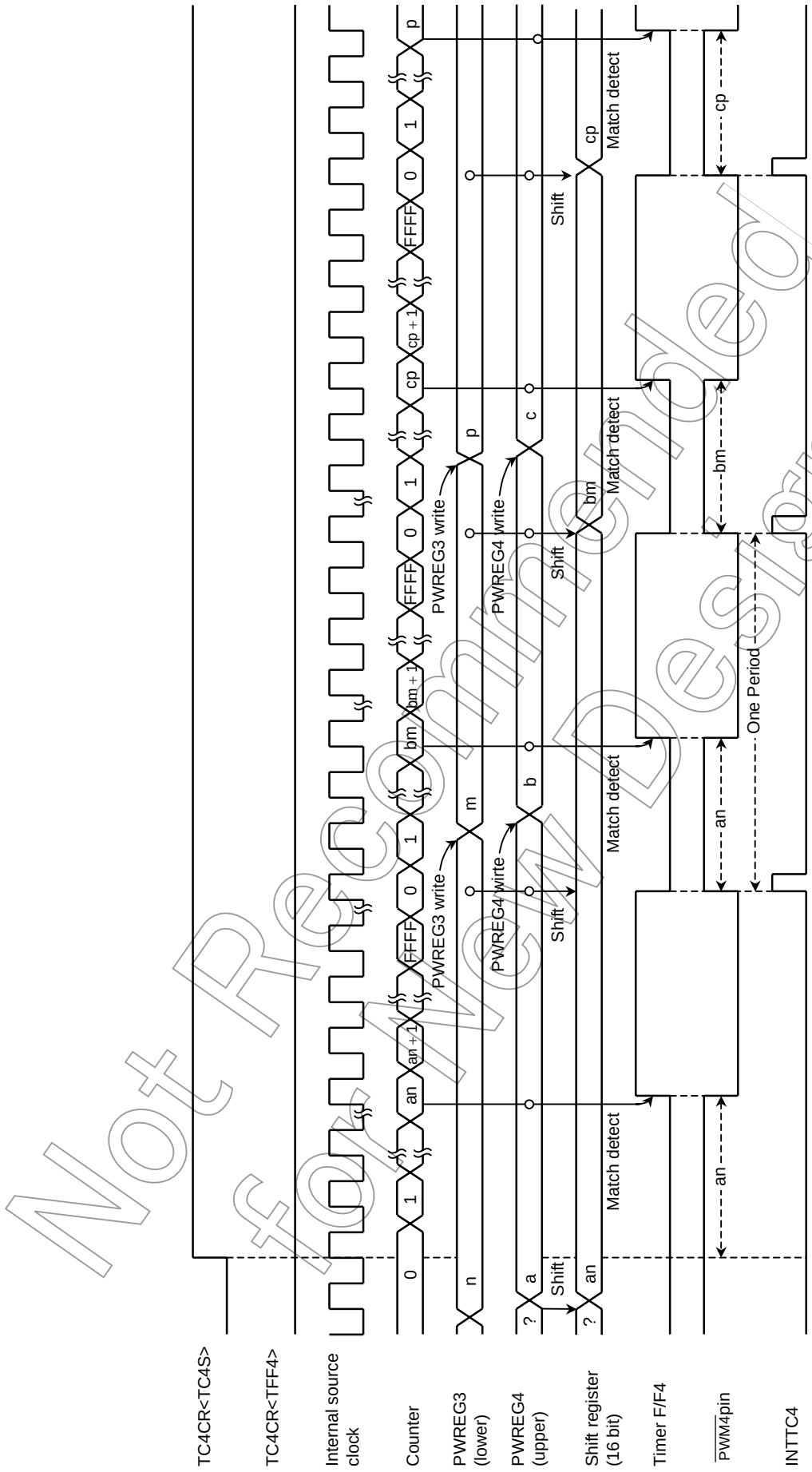


Figure 2.8.12 16-bit PWM mode timing chart (in case of timer/counter 3 and 4)

(8) 16-Bit Programmable Pulse Generate (PPG) output mode
(Timer/counter 3 and 4, Timer/counter 5 and 6)

PPG output with a resolution of 16 bits is possible. Timer/counter 3 and 4 (5 and 6) are also available as a 16-bit PPG output mode by cascade connection.

a. 16-bit PPG output mode of Timer/counter 3 and 4

First, the contents of PWREG3/4 are compared with the contents of up-counter. If a match is found, the timer F/F4 output is toggled. Next, timer F/F4 is again toggled and the counter is cleared by matching with TTREG3/4. The INTTC4 interrupt is generated at this time.

When used as $\overline{\text{PPG4}}$ pin, respective output latch should be set to "1". During reset, the F/F4 is initialized to "0".

The F/F4 output is configured by TC4CR<TFF4>. Therefore, the $\overline{\text{PPG4}}$ can output either output high or output low at first time. The timer register should write to the PWREG3/TTREG3 more first than PWREG4/TTREG4. The timer register must not write only either PWREG3/TTREG3 or PWREG4/TTREG4.

b. 16-bit PPG output mode of Timer/counter 5 and 6

First, the contents of PWREG5/6 are compared with the contents of up-counter. If a match is found, the timer F/F6 output is toggled. Next, timer F/F6 is again toggled and the counter is cleared by matching with TTREG5/6. The INTTC6 interrupt is generated at this time.

When used as $\overline{\text{PPG6}}$ pin, respective output latch should be set to "1". During reset, the F/F6 is initialized to "0".

The F/F6 output is configured by TC6CR<TFF6>. Therefore, the $\overline{\text{PPG6}}$ can output either output high or output low at first time. The timer register should write to the PWREG5/TTREG5 more first than PWREG6/TTREG6. The timer register must not write only either PWREG5/TTREG5 or PWREG6/TTREG6.

Example: Extract the pulse, whose term and "high" width is 16.385 ms and 1 ms respectively, from P32 with 16-bit PPG mode (at fc = 16.0 MHz, DV7CK = 0)

```
SET   (P3DR).2          ; Sets P32 output data latch to "1"
LDW   (PWREG3), 07D0H   ; Sets pulse width
LDW   (TTREG3), 8002H   ; Sets pulse term
LD    (TC3CR), 33H      ; Sets the 16-bit PPG mode (lower) and
                          source clock (fc/2^3)
LD    (TC4CR), 057H     ; Sets the TFF4 to "0" and sets the 16-bit
                          PPG mode(upper)
LD    (TC4CR), 05FH     ; Starts timer/counter
```

Note 1: In the programmable pulse generate (PPG) mode, do not change the setting of timer registers (PWREGi, TTREGi) while timer/counter is operating. Since PWREGi, TTREGi are configured as one-stage register, a newly set value is immediately reflected on the timer register.

Note 2: If PPG output is stopped during output operation, the output state is maintained at the state immediately before timer/counter is stopped. For changing the level of $\overline{\text{PPGj}}$, modify TCiCR<TTFi> after timer/counter has been stopped. Do not execute halt of timer/counter and modification of TFFj simultaneously.

Example: Fixes $\overline{\text{PPGj}}$ output at high level after timer/counter is stopped

```
CLR (TCjCR).3; Stops timer/counter
CLR (TCjCR).7; Sets PPGj output to high level output
```

Note 3: j = 4, 6 i = 3 to 6

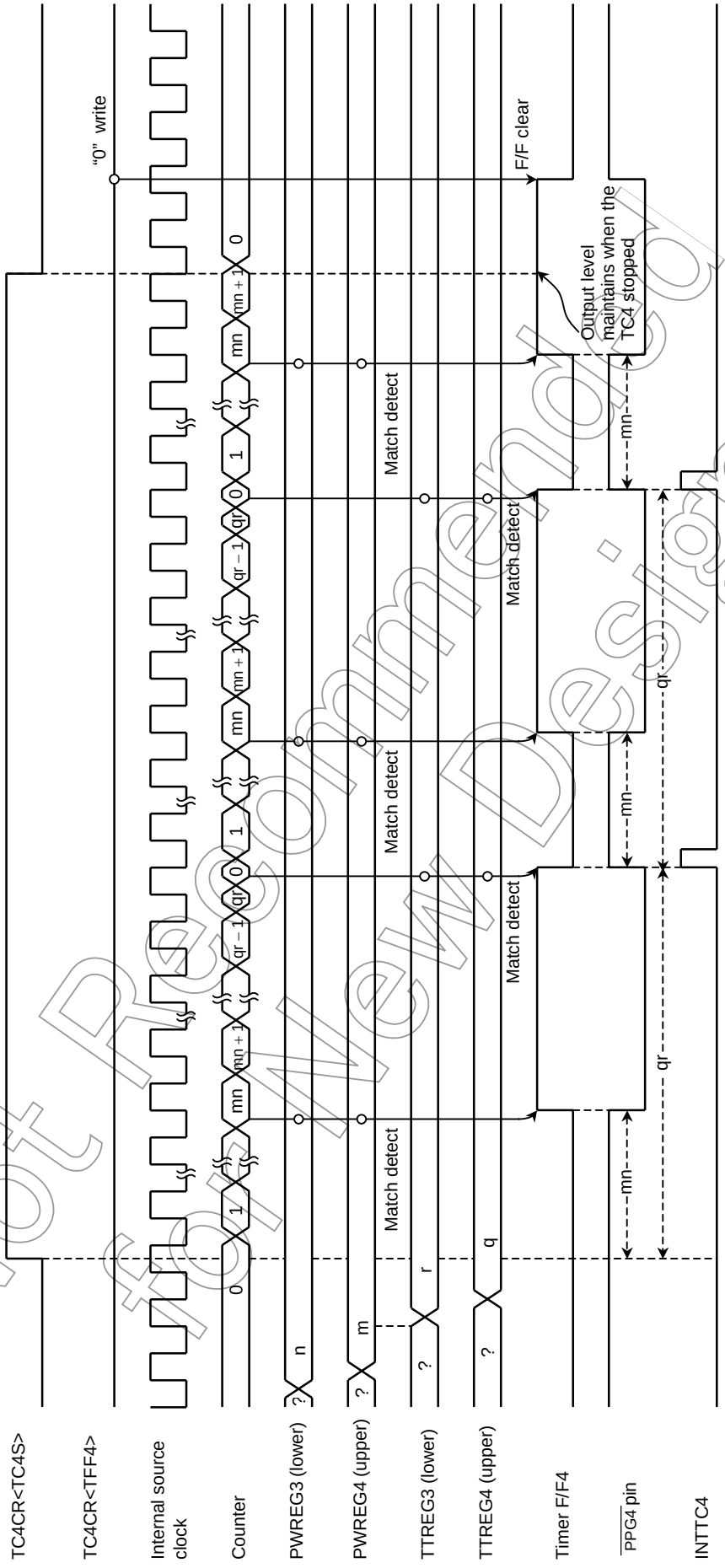


Figure 2.8.13 16-bit PPG mode timing chart (in case of timer/counter 3 and 4)

(9) Warming-up counter mode

In this mode, the warming-up period for switching the main system clock can be generated. Timer/counter 3 and 4 (5 and 6) are used as a 16-bit timer by cascade connection.

There are 2 modes in warming-up counter mode, one is a mode from NORMAL to SLOW and the other is a mode from SLOW to NORMAL.

Note 1: In the warming-up mode, always write TCiCR<TFFi> to "0". If TFFi is set to "1", unexpected pulse may be output from $\overline{\text{PDOi}}$ / $\overline{\text{PWMi}}$ / $\overline{\text{PPGi}}$ pin.

Note 2: In the warming-up mode, the lower 11 bits of TTREGm,n are ignored and an interrupt is generated by matching the upper 5 bits.

Note 3: i = 3, 4, 6 m = 4 and n = 3, or m = 6 and n = 5

a. Warming-up counter mode for low-frequency
(NORMAL1 → NORMAL2 → SLOW2 → SLOW1)

In this mode, it can obtain the warming-up period till the oscillation for low-frequency (fs) is stabilized.

Before timer/counter is started, turn on low-frequency oscillation by setting SYSCR2<XTEN> to "1".

After timer/counter is started by setting TCmCR<TCmS>, the contents of TTREGm, n are compared with the contents of up-counter. If a match is found, an INTTCm interrupt is generated, and the counter is cleared to "0".

In the interrupt service program, stop the timer/counter and change the system clock to low-frequency clock by setting SYSCR2<SYSCK> to "1".

After that, halt the high-frequency oscillation by clearing SYSCR2<XEN> to "0".

Table 2.8.8 Warming-up period for low-frequency oscillation (at fs = 32.768 kHz)

Min(at TTREGm, n = 0100H)	Max (at TTREGm, n = FF00H)
7.81 ms	1.99 s

Example: Switching to the SLOW1 mode after low-frequency clock has stabilized by using TC4, 3.

```
        SET    (SYSCR2).6        ; SYSCR2<XTEN> ← "1"
        LD     (TC3CR),43H       ; TFF3 = "0", fs for source clock, sets 16-bit
                                   mode
        LD     (TC4CR),05H       ; TFF4 = "0", sets warming-up counter
                                   mode
        LD     (TTREG3),8000H    ; sets warming-up time
                                   (depend on oscillator characteristics)
        DI                       ; IMF ← "0"
        SET    (EIRH).3          ; Enables INTTC4
        EI                       ; IMF ← "1"
        SET    (TC4CR).3         ; Starts TC4, 3
         :      :
PINTTC4: CLR   (TC4CR).3         ; Stops TC4, 3
        SET    (SYSCR2).5        ; SYSCR2<SYSCK> ← "1"
                                   (Switches the main system clock to the
                                   low-frequency clock)
        CLR    (SYSCR2).7        ; SYSCR2<XEN> ← "0"
                                   (Turns off low-frequency oscillation)
        RETI
         :      :
VINTTC4: DW    PINTTC4           ; INTTC4 vector table
```

b. Warming-up counter mode for high-frequency
(SLOW1 → SLOW2 → NORMAL2 → NORMAL1)

In this mode, it can obtain the warming-up period till the oscillation for high-frequency (fc) is stabilized.

Before timer/counter is started, turn on high-frequency oscillation by setting SYSCR2<XEN> to "1".

After timer/counter is started by setting TCmCR<TCmS>, the contents of TTREGm,n are compared with the contents of up-counter. If a match is found, an INTTCm interrupt is generated, and the counter is cleared to "0".

In the interrupt service program, stop the timer/counter and change the system clock to high-frequency clock by clearing SYSCR2<SYSCK> to "0".

After that, halt the low-frequency oscillation by clearing SYSCR2<XTEN> to "0".

Table 2.8.9 Warming-up period for high-frequency (at fc = 16 MHz)

Min (at TTREGm, n = 0100H)	Max (at TTREGm, n = FF00H)
16 μs	4.08 ms

Example: Switching to the NORMAL1 mode after high-frequency clock has stabilized by using TC4, 3.

```
        SET   (SYSCR2).7          ; SYSCR2<XEN> ← "1"
        LD    (TC3CR),63H         ; TFF3 = "0", fc for source clock, sets 16-bit
                                    mode
        LD    (TC4CR),05H         ; TFF4 = "0", sets warming-up counter
                                    mode
        LD    (TTREG3),0F800H     ; Sets warming-up time
                                    (depend on oscillator characteristics)
        DI                        ; IMF ← "0"
        SET   (EIRH).3            ; Enables INTTC4
        EI                        ; IMF ← "1"
        SET   (TC4CR).3           ; Starts TC4, 3
        :       :
PINTTC4: CLR  (TC4CR).3           ; Stops TC4, 3
        CLR   (SYSCR2).5          ; SYSCR2<SYSCK> ← "0"
                                    (Switches the main system clock to the
                                    high-frequency clock)
        CLR   (SYSCR2).6          ; SYSCR2<XTEN> ← "0"
                                    (Turns off high-frequency oscillation)
        RETI
        :       :
VINTTC4: DW   PINTTC4             ; INTTC4 vector table
```

2.9 UART (Asynchronous Serial Interface)

The TMP86FM29 has 1 channel of UART (asynchronous serial interface).

The UART is connected to external devices via RxD and TxD. RxD is also used as P15; TxD, as P16. To use P15 or P16 as the RxD or TxD pin, set P1 port output latches to "1".

2.9.1 Configuration

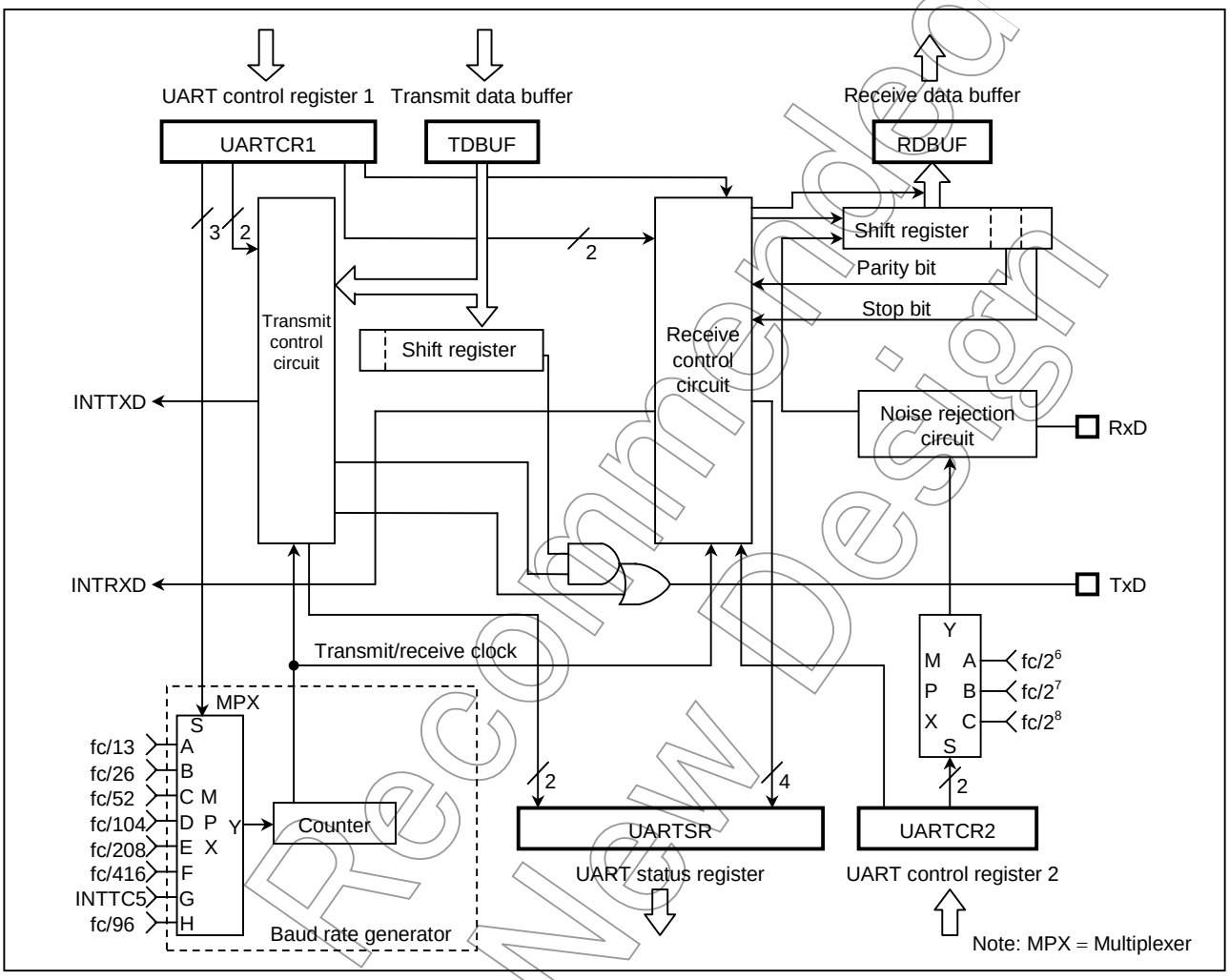


Figure 2.9.1 UART

2.9.2 Control

UART is controlled by the UART control registers (UARTCR1, UARTCR2). The operating status can be monitored using the UART status register (UARTSR).

UART control register

UARTCR1 (0025_H)

7	6	5	4	3	2	1	0
TXE	RXE	STBT	EVEN	PE	BRG		

(Initial value: 0000 0000)

BRG	Transmit clock select	000: fc/13 [Hz] 001: fc/26 010: fc/52 011: fc/104 100: fc/208 101: fc/416 110: TC5 (INTTC5) 111: fc/96	Write only
PE	Parity addition	0: No parity 1: Parity	
EVEN	Even-numbered parity	0: Odd-numbered parity 1: Even-numbered parity	
STBT	Transmit stop bit length	0: 1 bit 1: 2 bits	
RXE	Receive operation	0: Disable 1: Enable	
TXE	Transfer operation	0: Disable 1: Enable	

Note 1: When operations are disabled by setting TXE and RXE bit to "0", the setting becomes valid when data transmit or receive complete. When the transmit data is stored in the transmit data buffer, the data are not transmitted. Even if data transmit is enabled, until new data are written to the transmit data buffer, the current data are not transmitted.

Note 2: The transmit clock and the parity are common to transmit and receive.

Note 3: UARTCR1<RXE> and UARTCR1<TXE> should be set to "0" before UARTCR1<BRG> is changed

UARTCR2 (0026_H)

7	6	5	4	3	2	1	0
					RXDNC	STOPBR	

(Initial value: **** *000)

STOPBR	Receive stop bit length	0: 1 bit 1: 2 bits	Write only
RxDNC	Selection of RxD input noise rejection time	00: No noise rejection (hysteresis input) 01: Rejects pulses shorter than 31/fc [s] as noise 10: Rejects pulses shorter than 63/fc [s] as noise 11: Rejects pulses shorter than 127/fc [s] as noise	

Note: When UARTCR2<RXDNC> = "01", pulses longer than 96/fc [s] are always regarded as signals; when UARTCR2<RXDNC> = "10", longer than 192/fc [s]; and when UARTCR2<RXDNC> = "11", longer than 384/fc [s]

Figure 2.9.2 UART control register

UARTSR (0025_H)

7	6	5	4	3	2	1	0
PERR	FERR	OERR	RBFL	TEND	TBEP		

(Initial value: 0000 11**)

TBEP	Transmit data buffer empty flag	0: Transmit data buffer full 1: Transmit data buffer empty	Read only
TEND	Transmit end flag	0: Transmitting 1: Transmit end	
RBFL	Receive data buffer full flag	0: Receive data buffer empty 1: Receive data buffer full	
OERR	Overrun error flag	0: No overrun error 1: Overrun error	
FERR	Framing error flag	0: No framing error 1: Framing error	
PERR	Parity error flag	0: No parity error 1: Parity error	

Note: When an INTTxD is generated TBEP is set to "1" automatically.

UART receive data buffer

RDBUF ($0F9B_H$)

7	6	5	4	3	2	1	0

Read only
(Initial value: 0000 0000)

UART transmit data buffer

TDBUF ($0F9B_H$)

7	6	5	4	3	2	1	0

Write only
(Initial value: 0000 0000)

Figure 2.9.3 UART status register and data buffer registers

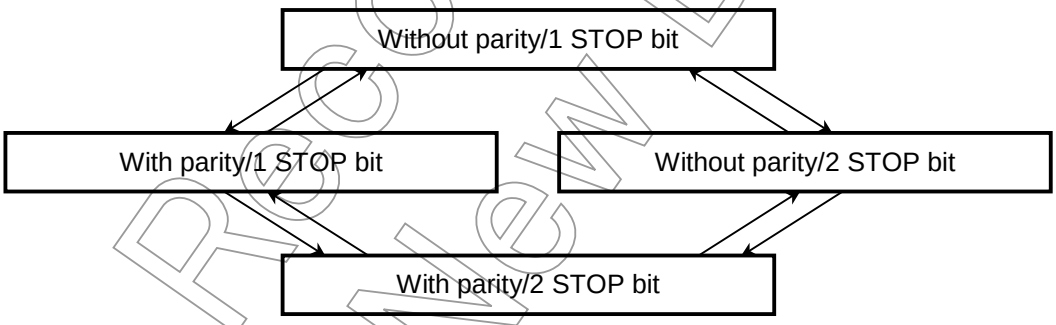
2.9.3 Transfer Data Format

In UART, a one-bit start bit (low level), stop bit (bit length selectable at high level, by UARTCR1<STBT>), and parity (select parity in UARTCR1<PE>; even-or odd-numbered parity by UARTCR1<EVEN>) are added to the transfer data. The transfer data formats are shown as follow.

Table 2.9.1 Transfer data format

PE	STBT	Frame length: 1	2	3	------	8	9	10	11	12
0	0	Start	Bit 0	Bit 1	------	Bit 6	Bit 7	Stop 1		
0	1	Start	Bit 0	Bit 1	------	Bit 6	Bit 7	Stop 1	Stop 2	
1	0	Start	Bit 0	Bit 1	------	Bit 6	Bit 7	Parity	Stop 1	
1	1	Start	Bit 0	Bit 1	------	Bit 6	Bit 7	Parity	Stop 1	Stop 2

Note: In order to switch the transmit data format, perform transmit operations in the following sequence except for the initial setting.



2.9.4 Transfer Rate

The baud rate of UART is set of UARTCR1<BRG>. The example of the baud rate shown as follows.

Table 2.9.2 Transfer rate

BRG	Source Clock		
	16 MHz	8 MHz	4 MHz
000	76800 [baud]	38400 [baud]	19200 [baud]
001	38400	19200	9600
010	19200	9600	4800
011	9600	4800	2400
100	4800	2400	1200
101	2400	1200	600

When TC5 is used as the UART transfer rate (when UARTCR1<BRG> = "110"), the transfer clock and transfer rate are determined as follows:

$$\text{Transfer clock} = \frac{\text{TC5 source clock}}{\text{TTREG5 set value}}$$

$$\text{Transfer rate} = \frac{\text{Transfer clock}}{16}$$

2.9.5 Data Sampling

The UART receiver keeps sampling input using the clock selected by UARTCR1<BRG> until a start bit is detected in RxD pin input. RT clock starts detecting "L" level of the RxD pin. Once a start bit is detected, the start bit, data bits, stop bit(s), and parity bit are sampled at three times of RT7, RT8, and RT9 during one receiver clock interval (RT clock). (RT0 is the position where the bit supposedly starts). Bit is determined according to majority rule (the data are the same twice or more out of three samplings).

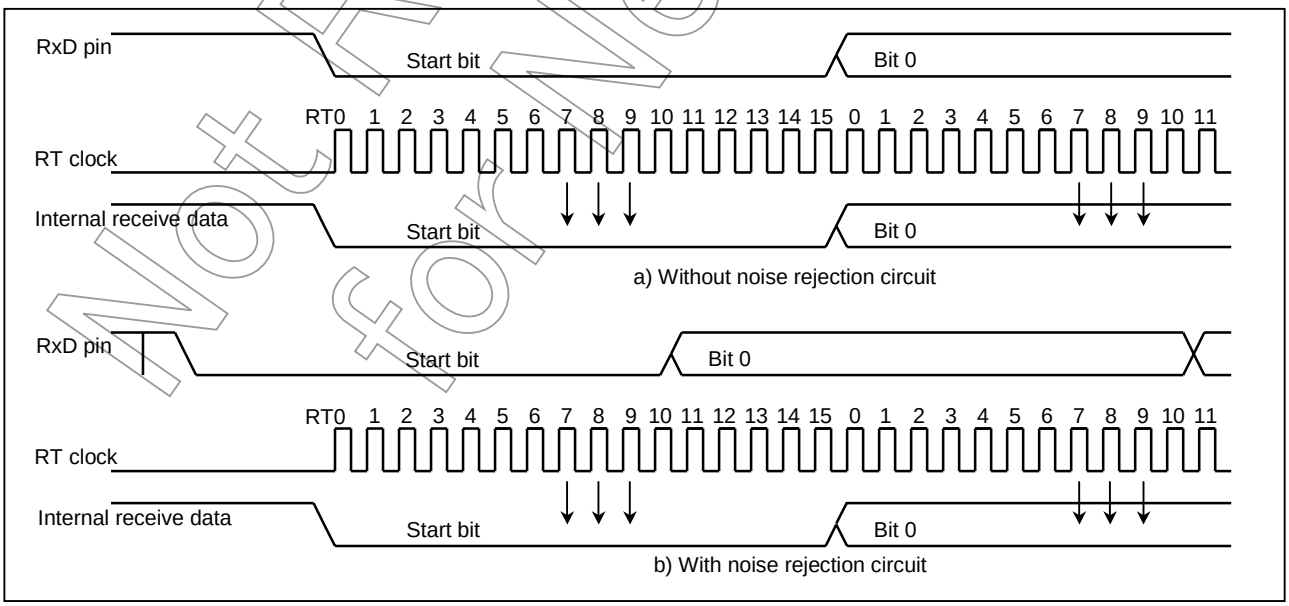


Figure 2.9.4 Data sampling

2.9.6 STOP Bit Length

Select a transmit stop bit length (1 or 2 bits) by UARTCR1<STBT>.

2.9.7 Parity

Set parity/no parity by UARTCR1<PE>; set parity type (odd-or even-numbered) by UARTCR1<EVEN>.

2.9.8 Transmit/Receive

(1) Data transmit

Set UARTCR1<TXE> to "1". Read UARTSR to check UARTSR<TBEP> = "1", then write data in TDBUF (transmit data buffer). Writing data in TDBUF zero-clears UARTSR<TBEP>, transfers the data to the transmit shift register and the data are sequentially output from the TxD pin. The data output include a one-bit start bit, stop bits whose number is specified in UARTCR1<STBT> and a parity bit if parity addition is specified. Select the data transfer baud rate using bits 0 to 2 in UARTCR1. When data transmit starts, transmit buffer empty flag UARTSR<TBEP> is set to "1" and an INTTXD interrupt is generated.

While UARTCR1<TXE> = "0" and from when "1" is written to UARTCR1<TXE> to when send data are written to TDBUF, the TXD pin is fixed at High level. When transmitting data, first read UARTSR, then write data in TDBUF. Otherwise, UARTSR<TBEP> is not zero-cleared and transmit does not start.

(2) Data receive

Set UARTCR1<RXE> to "1". When data are received via the RxD pin, the receive data are transferred to RDBUF (receive data buffer). At this time, the data transmitted include a start bit and stop bit(s) and a parity bit if parity addition is specified. When stop bit(s) are received, data only are extracted and transferred to RDBUF (receive data buffer). Then the receive buffer full flag UARTSR<RBFL> is set and an INTRXD interrupt is generated. Select the data transfer baud rate using bits 0 to 2 in UARTCR1.

If an overrun error (OERR) occurs when data are received, the data are not transferred to RDBUF (receive data buffer) but discarded; data in the RDBUF are not affected.

Note: When a receive operation is disabled by setting UARTCR1<RXE> bit to "0", the setting becomes valid when data receive is completed. However, if a framing error occurs in data receive, the receive-disabling setting may not become valid. if a framing error occurs, be sure to perform a re-receive operation.

2.9.9 Status Flag/Interrupt Signal

(1) Parity error

When parity determined using the receive data bits differs from the received parity bit, the parity error flag UARTSR<PERR> is set to "1". The UARTSR<PERR> is cleared to "0" when the RDBUF is read after reading the UARTSR.

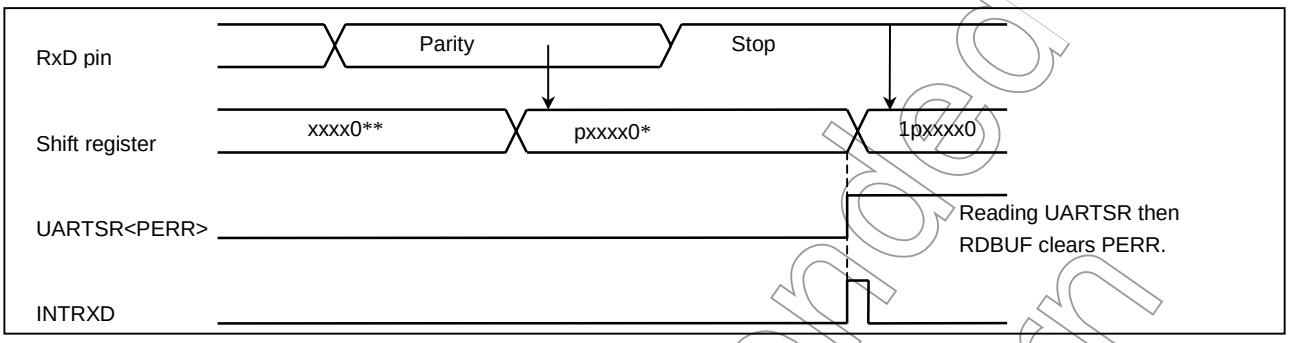


Figure 2.9.5 Generation of parity error

(2) Framing error

When "0" is sampled as the stop bit in the receive data, framing error flag UARTSR<FERR> is set to "1". The UARTSR<FERR> is cleared to "0" when the RDBUF is read after reading the UARTSR.

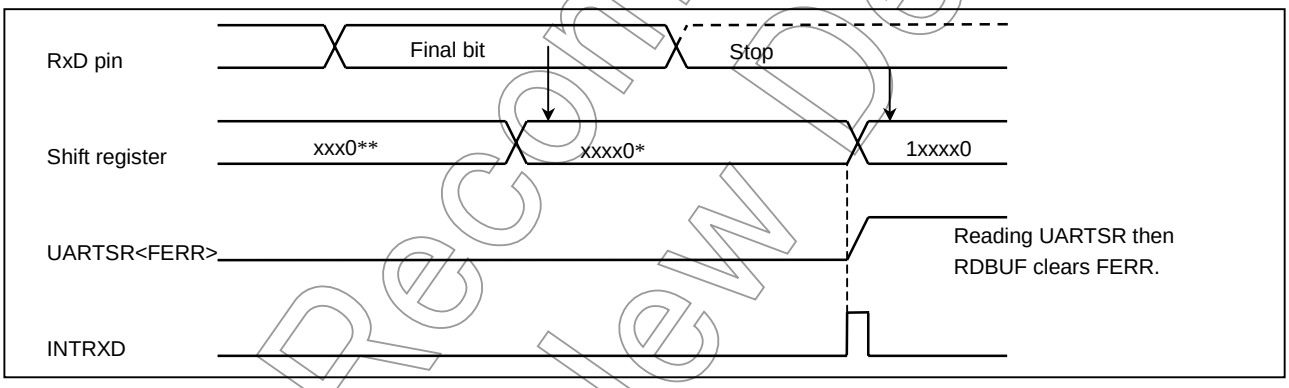


Figure 2.9.6 Generation of framing error

(3) Overrun error

When all bits in the next data are received while unread data are still in RDBUF, overrun error flag UARTSR<OERR> is set to “1”. In this case, the receive data is discarded; data in RDBUF are not affected. The UARTSR<OERR> is cleared to “0” when the RDBUF is read after reading the UARTSR.

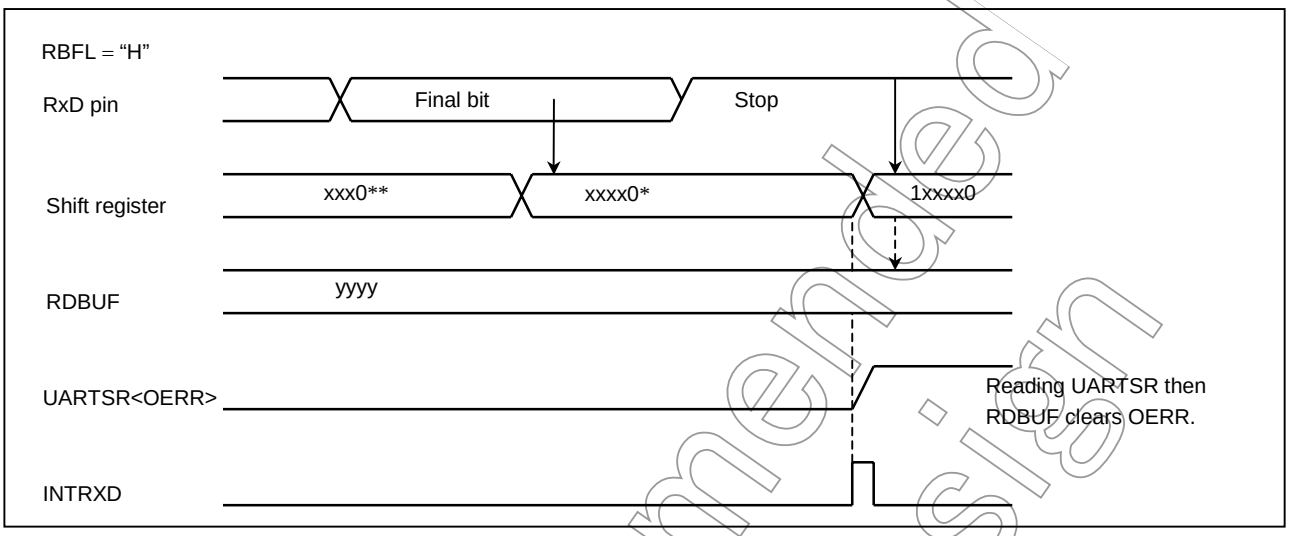


Figure 2.9.7 Generation of overrun error

(4) Receive data buffer full

Loading the received data in RDBUF sets receive data buffer full flag UARTSR<RBFL>. The UARTSR<RBFL> is cleared to “0” when the RDBUF is read after reading the UARTSR.

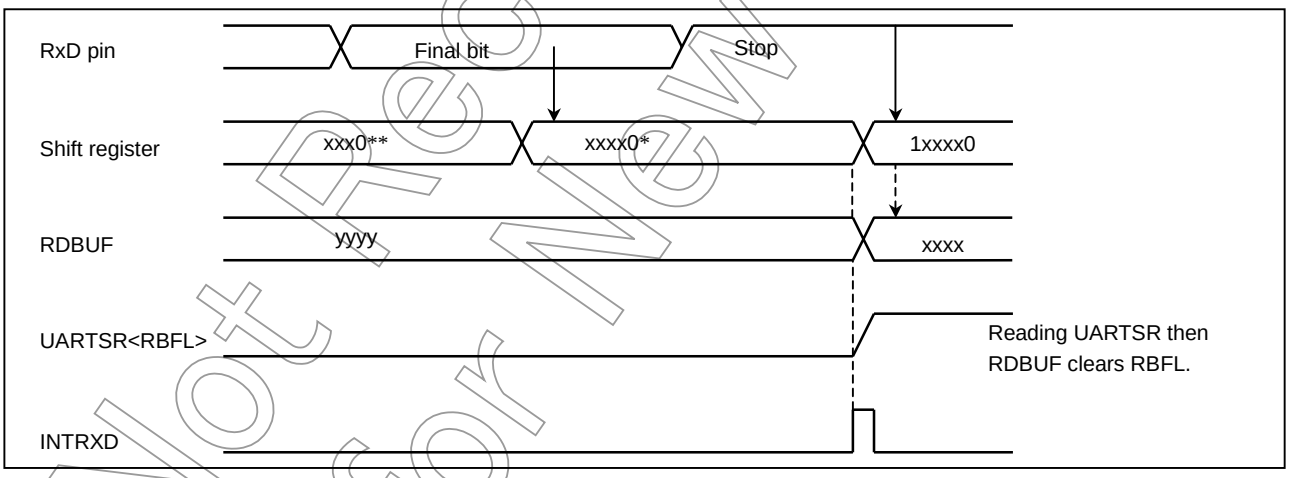


Figure 2.9.8 Generation of receive buffer full

(5) Transmit data buffer empty

When no data is in the transmit buffer TDBUF, UARTSR<TBEP> is set to “1”, that is, when data in TDBUF are transferred to the transmit shift register and data transmit starts, transmit data buffer empty flag UARTSR<TBEP> is set to “1”. The UARTSR<TBEP> is cleared to “0” when the TDBUF is written after reading the UARTSR.

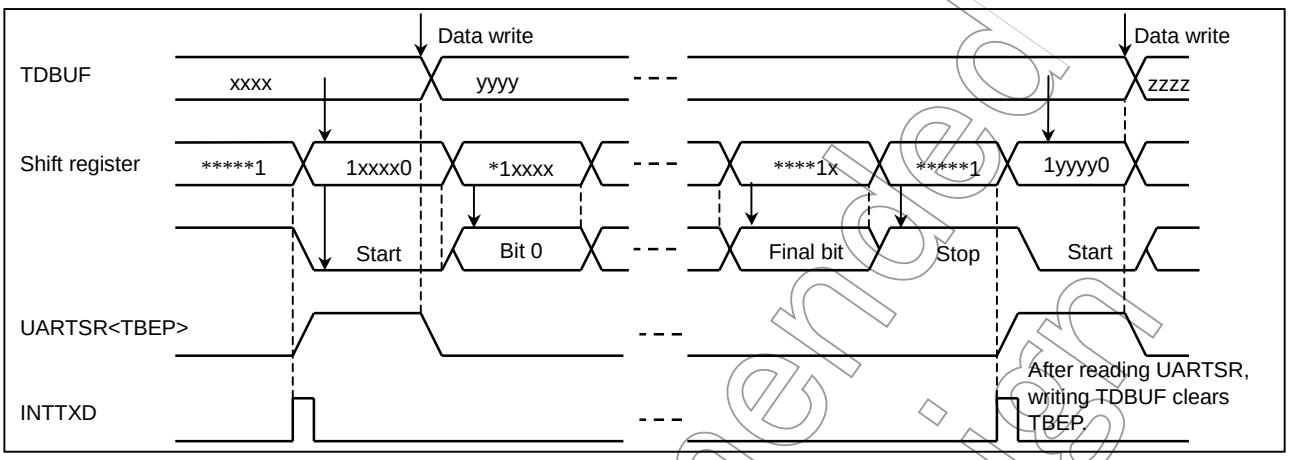


Figure 2.9.9 Generation of transmit buffer empty

(6) Transmit end flag

When data are transmitted and no data is in TDBUF (UARTSR<TBEP> = “1”), transmit end flag UARTSR<TEND> is set to “1”. The UARTSR<TEND> is cleared to “0” the data transmit is stated after writing the TDBUF.

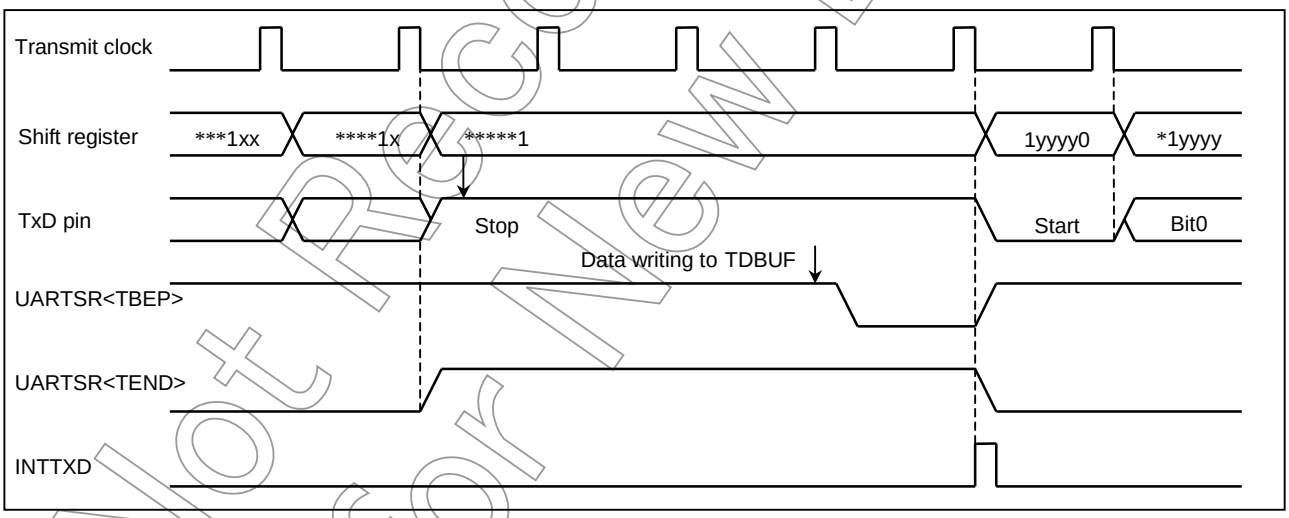


Figure 2.9.10 Generation of transmit buffer empty

2.10 Serial Interface (SIO)

The TMP86FM29 has one clocked-synchronous 8-bit serial interface. Serial interface has an 8-byte transmit and receive data buffer that can automatically and continuously transfer up to 64 bits of data.

The serial interface is connected to external devices via pins P16 (SO), P15 (SI), P17 ($\overline{SCK}$). The serial interface pins are also used as port P1. When these pins are used as serial interface pins, the correspondence output latch should be set to "1". In the transmit mode, pin P15 can be used as normal I/O port, and in the receive mode, the pin P16 can be used as normal I/O ports.

2.10.1 Configuration

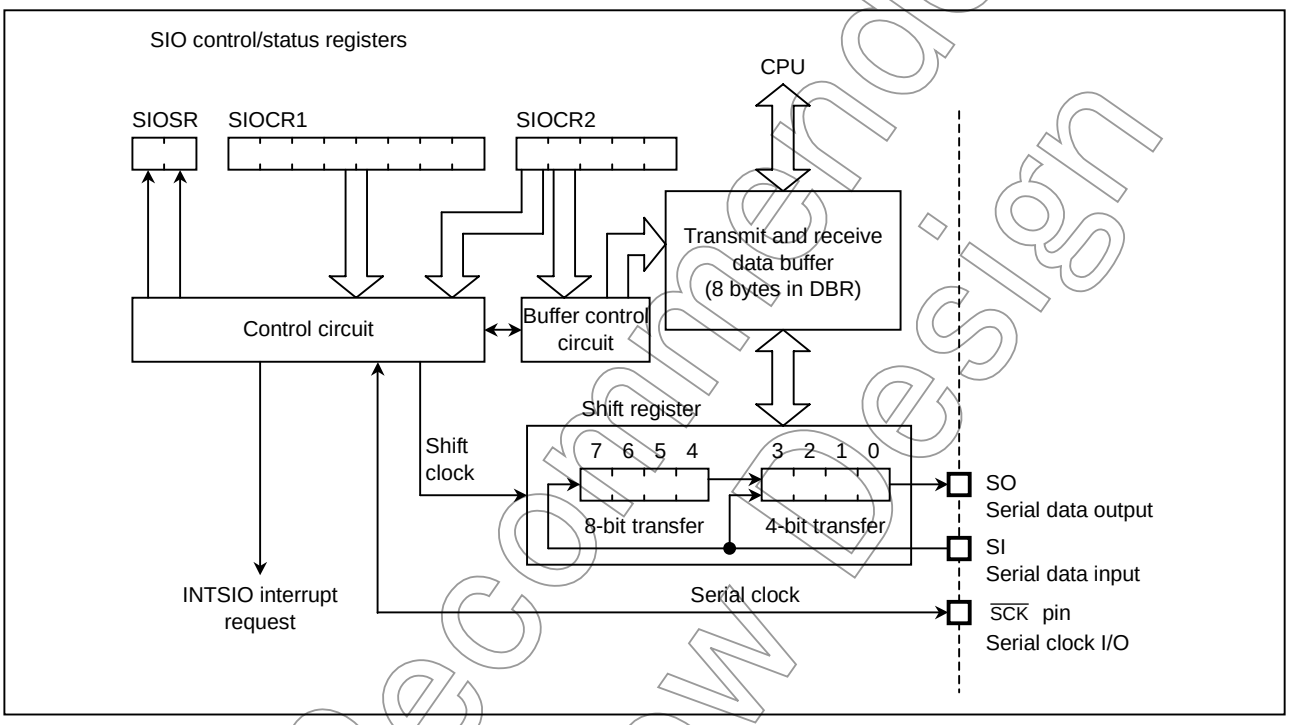


Figure 2.10.1 Serial interfaces

2.10.2 Control

The serial interface is controlled by SIO control registers (SIOCR1/SIOCR2). The serial interface status can be determined by reading SIO status register (SIOSR).

The transmit and receive data buffer is controlled by the SIOCR2<BUF>. The data buffer is assigned to address $0F90_H$ to $0F97_H$ for SIO in the DBR area, and can continuously transfer up to 8 words (bytes or nibbles) at one time. When the specified number of words has been transferred, a buffer empty (in the transmit mode) or a buffer full (in the receive mode or transmit/receive mode) interrupt (INTSIO) is generated.

When the internal clock is used as the serial clock in the 8-bit receive mode and the 8-bit transmit/receive mode, a fixed interval wait can be applied to the serial clock for each word transferred. Four different wait times can be selected with SIOCR2<WAIT>.

SIO Control Register 1

SIOCR1 (0F98H)

7	6	5	4	3	2	1	0
SIOS	SIOINH	SIOM			SCK		

(Initial value: 0000 0000)

SIOS	Transfer start/stop control	0: Transfer stop 1: Transfer start	Write only
SIOINH	Continue/Abort transfer	0: Continue transfer 1: Abort transfer (automatically cleared to “0” after abort)	
SIOM	Transfer mode select	000: 8-bit transmit mode 010: 4-bit transmit mode 100: 8-bit transmit/receive mode 101: 8-bit receive mode 110: 4-bit receive mode Except the above : Reserved	
SCK	Serial clock select	See table below	

	NORMAL1/2, IDLE1/2 Mode		SLOW1/2, SLEEP1/2 Mode
	DV7CK = 0	DV7CK = 1	
000	$fc/2^{13}$	$fs/2^{5}$	$fs/2^{5}$
001	$fc/2^{8}$	$fc/2^{8}$	–
010	$fc/2^{7}$	$fc/2^{7}$	–
011	$fc/2^{6}$	$fc/2^{6}$	–
100	$fc/2^{5}$	$fc/2^{5}$	–
101	$fc/2^{4}$	$fc/2^{4}$	–
110	Reserved		–
111	External clock (input from $\overline{SCK}$ pin)		

Note 1: fc; High-frequency clock [Hz] fs; Low-frequency clock [Hz]

Note 2: Set SIOS to “0” and SIOINH to “1” when setting the transfer mode or serial clock.

Note 3: SIOCR1 is write-only register, which cannot access any of in read-modify-write instruction such as bit operate, etc.

SIO Status Register

SIOSR (0F99H)

7	6	5	4	3	2	1	0
SIOF	SEF						

(Initial value: 00** ****)

SIOF	Serial transfer operating status monitor	0: Transfer terminated 1: Transfer in process	Read only
SEF	Shift operating status monitor	0: Shift operation terminated 1: Shift operation in process	

Note 1: The SIO1SR<SIOF> bit is cleared to “0” by clearing SIO1CR1<SIOS> to stop transferring or by setting SIO1CR1<SIOINH> to “1” to abort transfer.

Figure 2.10.2 SIO control register and status register (1/2)

SIO Control Register 2

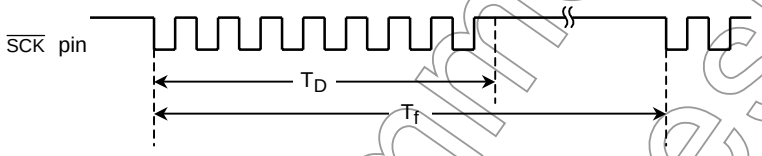
SIOCR2 (0F99$_H$)

7	6	5	4	3	2	1	0
			WAIT		BUF		

(Initial value: ***0 0000)

WAIT	Wait control	Always sets "00" except 8-bit transmit/receive mode. 00: $T_f = T_D$ (non-wait) 01: $T_f = 2T_D$ 10: $T_f = 4T_D$ (wait) 11: $T_f = 8T_D$	Write only
BUF	Number of transfer words	Buffer address used 000: 1 word transfer 0F90$_H$ 001: 2 words transfer 0F90 to 0F91$_H$ 010: 3 words transfer 0F90 to 0F92$_H$ 011: 4 words transfer 0F90 to 0F93$_H$ 100: 5 words transfer 0F90 to 0F94$_H$ 101: 6 words transfer 0F90 to 0F95$_H$ 110: 7 words transfer 0F90 to 0F96$_H$ 111: 8 words transfer 0F90 to 0F97$_H$	

Note :1 T_f; Frame time, T_D; Data transfer time



Note 2: The lower 4 bits of each buffer are used during 4-bit transfers. Zeros (0) are stored to the upper 4bits when receiving.

Note 3: Transmitting starts at the lowest address. Received data are also stored starting from the lowest address to the highest address. For example, in the case of SIO, the first buffer address transmitted is 0F90$_H$.

Note 4: The value to be loaded to BUF is held after transfer is completed.

Note 5: SIOCR2 must be set when the serial interface is stopped (SIOF = 0).

Note 6: SIOCR2 is write-only register, which cannot access any of in read-modify-write instruction such as bit operate, etc.

Note 7: *; Don't care

Figure 2.10.3 SIO control register and status register (2/2)

(1) Serial clock

a. Clock Source

SIOCR1<SCK> is able to select the following:

1. Internal clock

Any of four frequencies can be selected. The serial clock is output to the outside on the $\overline{SCK}$ pin. The $\overline{SCK}$ pin goes high when transfer starts.

When data writing (in the transmit mode) or reading (in the receive mode or the transmit/receive mode) cannot keep up with the serial clock rate, there is a wait function that automatically stops the serial clock and holds the next shift operation until the read/write processing is completed.

Table 2.10.1 Serial clock rate

	NORMAL1/2, IDLE1/2 Modes				SLOW, SLEEP Modes	
	DV7CK = 0		DV7CK = 1			
SCK	Clock	Baud Rate	Clock	Baud Rate	Clock	Baud Rate
000	$fc/2^{13}$	1.91 Kbps	$fs/2^5$	1024 bps	$fs/2^5$	1024 bps
001	$fc/2^8$	61.04 Kbps	$fc/2^8$	61.04 Kbps	–	–
010	$fc/2^7$	122.07 Kbps	$fc/2^7$	122.07 Kbps	–	–
011	$fc/2^6$	244.14 Kbps	$fc/2^6$	244.14 Kbps	–	–
100	$fc/2^5$	488.28 Kbps	$fc/2^5$	488.28 Kbps	–	–
101	$fc/2^4$	976.56 Kbps	$fc/2^4$	976.56 Kbps	–	–
110	–	–	–	–	–	–
111	External		External		External	

1 Kbit = 1024 bit

(fc = 16 MHz, fs = 32.768 kHz)

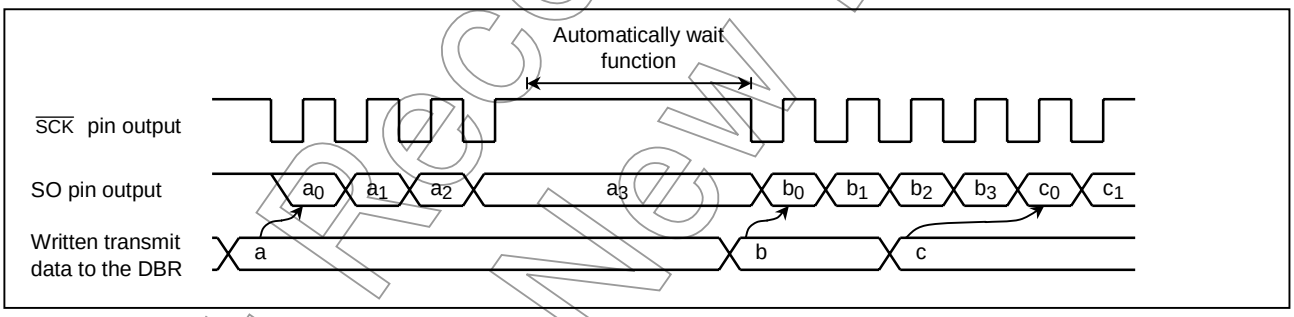
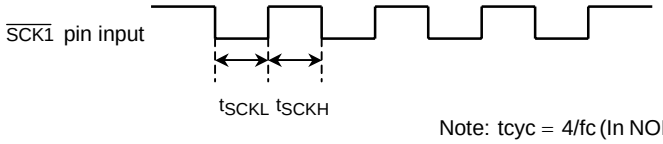


Figure 2.10.4 Wait function (example: 4-bit serial transfer)

2. External clock

An external clock connected to the $\overline{SCK}$ pin is used as the serial clock. In this case, the P17 ($\overline{SCK}$) output latch must be set to "1". To ensure shifting, both the high and low levels of the serial clock pulse width must be at least 4 machine cycles. This pulse is needed for the shift operation to execute certainly. Therefore, the maximum available transfer rate is 488.3 Kbits/s when the operation frequency is fc=16MHz.



t_{SCKL}, t_{SCKH} > 4 tcyc

Note: tcyc = 4/fc (In NORAML1/2, IDLE1/2 modes)
4/fs (In SLOW1/2, SLEEP1/2 modes)

b. Shift edge

The leading edge is used to transmit, and the trailing edge is used to receive.

1. Leading edge

Transmitted data are shifted on the leading edge of the serial clock (falling edge of the $\overline{SCK}$ pin input/output).

2. Trailing edge

Received data are shifted on the trailing edge of the serial clock (rising edge of the $\overline{SCK}$ pin input/output).

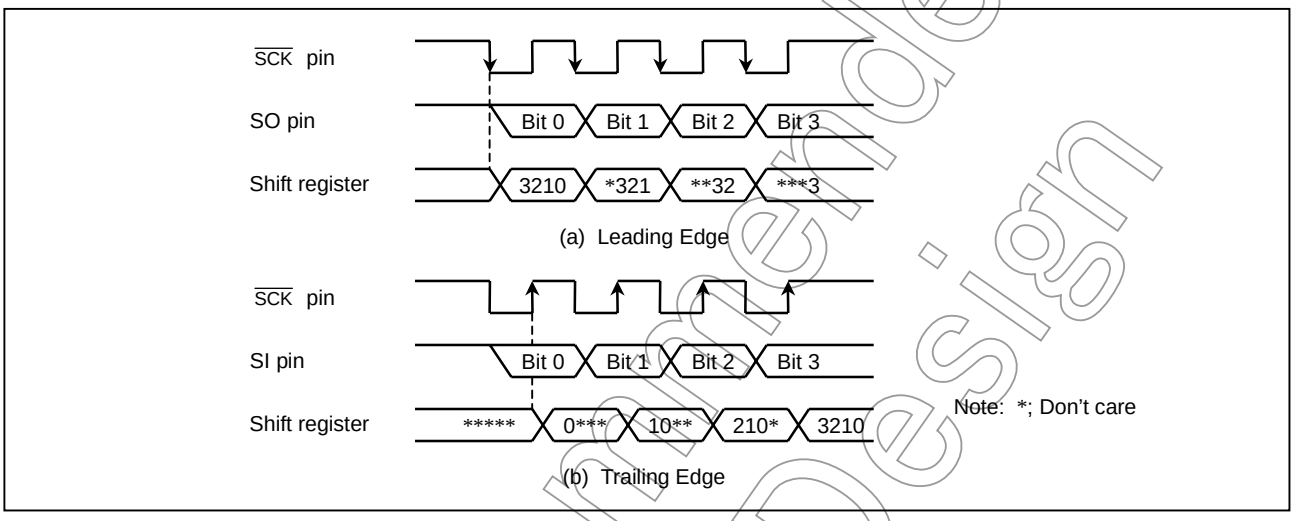


Figure 2.10.5 Shift edge

(2) Number of bits to transfer

Either 4-bit or 8-bit serial transfer can be selected. When 4-bit serial transfer is selected, only the lower 4 bits of the transmit/receive data buffer register are used. The upper 4 bits are cleared to "0" when receiving.

The data is transferred in sequence starting at the least significant bit (LSB).

(3) Number of words to transfer

Up to 8 words consisting of 4 bits of data (4-bit serial transfer) or 8 bits (8-bit serial transfer) of data can be transferred continuously. The number of words to be transferred is loaded to SIOCR2<BUF>.

An INTSIO interrupt is generated when the specified number of words has been transferred. If the number of words is to be changed during transfer, the serial interface must be stopped before making the change. The number of words can be changed during automatic-wait operation of an internal clock. In this case, the serial interface is not required to be stopped.

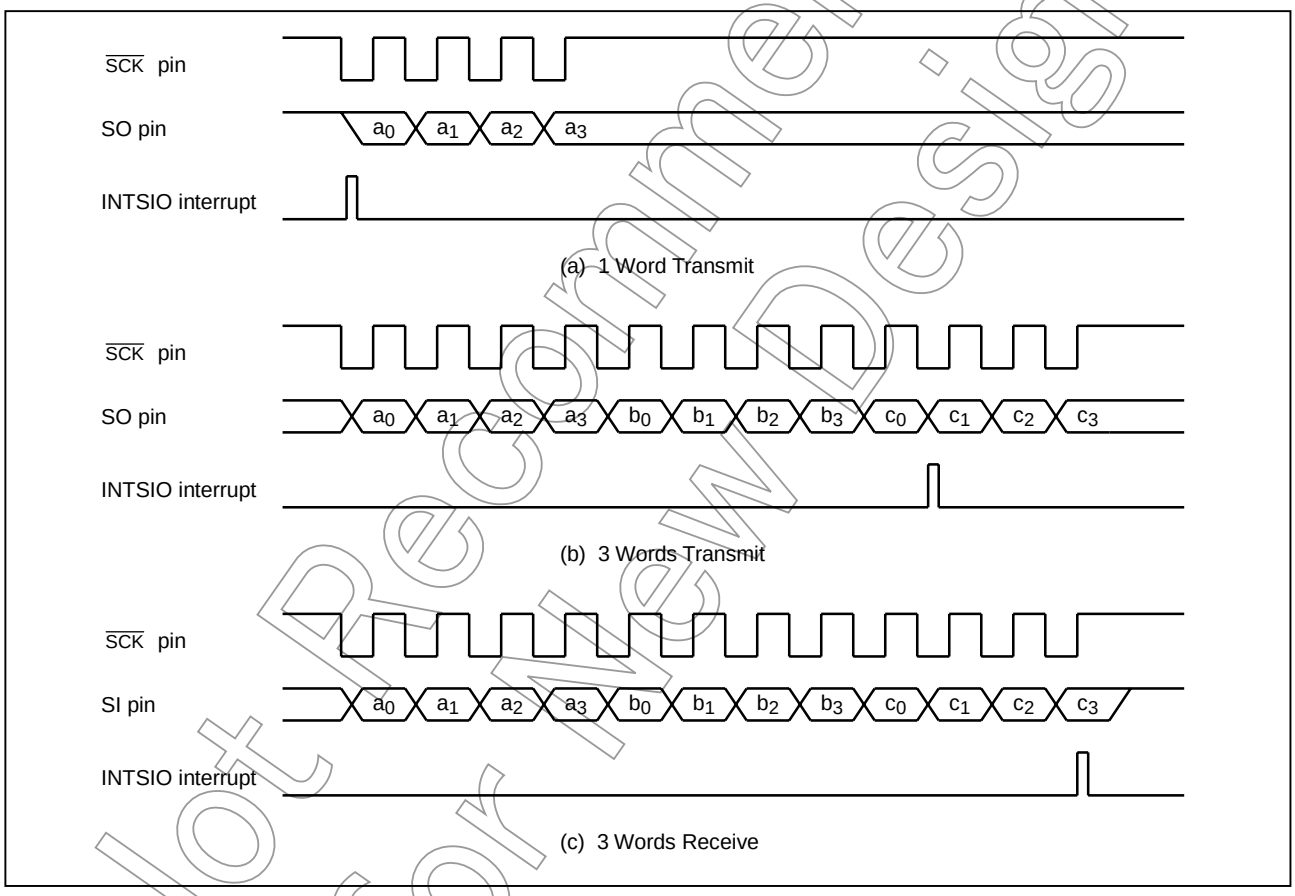


Figure 2.10.6 Number of bits to transfer (example: 4-bit serial transfer)

2.10.3 Transfer Mode

SIOCR1<SIOM> is used to select the transmit, receive, or transmit/receive mode.

(1) 4-bit and 8-bit Transmit Modes

In these modes, the SIOCR1 is set to the transmit mode and then the data to be transmitted first are written to the data buffer registers (DBR). After the data are written, the transmission is started by setting SIOCR1<SIOS> to "1". The data are then output sequentially to the SO pin in synchronous with the serial clock, starting with the least significant bit (LSB). As soon as the LSB has been output, the data are transferred from the data buffer register to the shift register. When the final data bit has been transferred and the data buffer register is empty, an INTSIO (buffer empty) interrupt is generated to request the next transmitted data.

When the internal clock is used, the serial clock will stop and an automatic-wait will be initiated if the next transmitted data are not loaded to the data buffer register by the time the number of data words specified with the SIOCR2<BUF> has been transmitted. Writing even one word of data cancels the automatic-wait; therefore, when transmitting two or more words, always write the next word before transmission of the previous word is completed.

Note: Automatic-waits are also canceled by writing to a DBR not being used as a transmit data buffer register; therefore, during SIO do not use such DBR for other applications. For example, when 3 words are transmitted, do not use the DBR of the remained 5 words.

When an external clock is used, the data must be written to the data buffer register before shifting next data. Thus, the transfer speed is determined by the maximum delay time from the generation of the interrupt request to writing of the data to the data buffer register by the interrupt service program.

The transmission is ended by clearing SIOCR1<SIOS> to "0" or setting SIOCR1<SIOINH> to "1" in buffer empty interrupt service program. That the transmission has ended can be determined from the status of SIOSR<SIOF> because SIOSR<SIOF> is cleared to "0" when a transfer is completed.

When SIOCR1<SIOINH> is set, the transmission is immediately ended and SIOSR<SIOF> is cleared to "0".

When an external clock is used, it is also necessary to clear SIOCR1<SIOS> to "0" before shifting the next data; otherwise, dummy data will be transmitted and the operation will end.

If it is necessary to change the number of words, SIOCR1<SIOS> should be cleared to "0", then SIOCR2<BUF> must be rewritten after confirming that SIOSR<SIOF> has been cleared to "0".

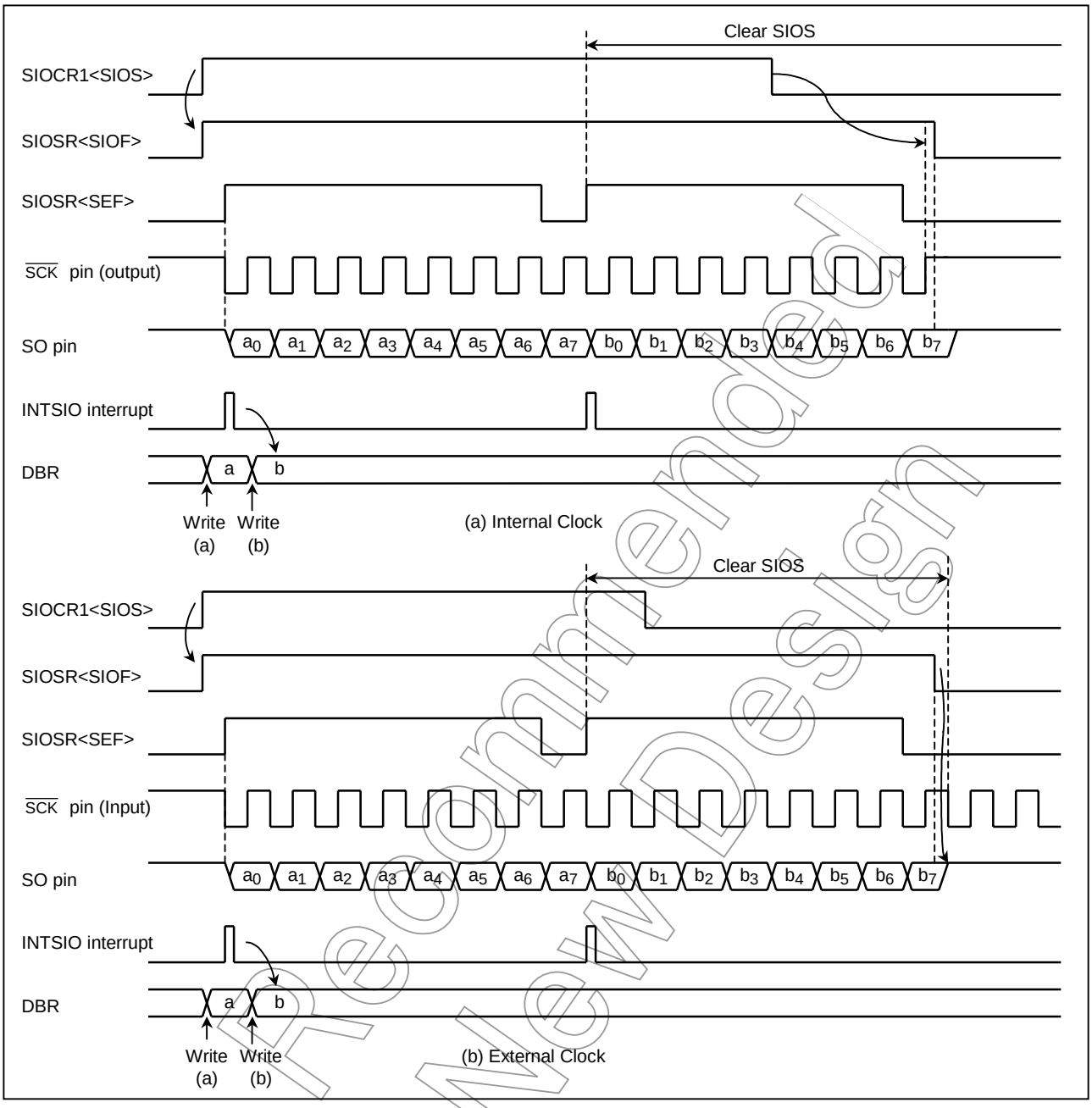


Figure 2.10.7 Transfer mode (example: 8-bit, 1 word transfer)

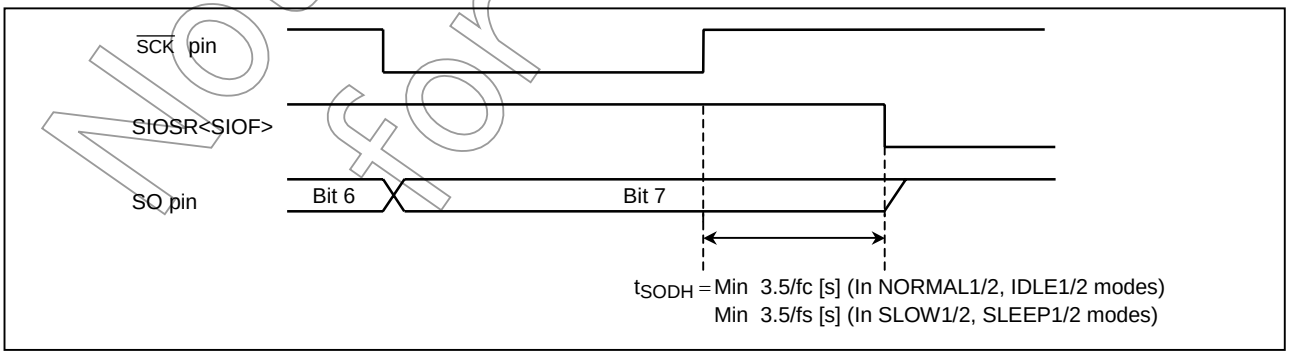


Figure 2.10.8 Transmitted data hold time at end of transmit

(2) 4-bit and 8-bit receive modes

After setting the control registers to the receive mode, set SIOCR1<SIOS> to "1" to enable receiving. The data are then transferred to the shift register via the SI pin in synchronous with the serial clock. When one word of data has been received, it is transferred from the shift register to the data buffer register (DBR). When the number of words specified with the SIOCR2<BUF> has been received, an INTSIO (buffer full) interrupt is generated to request that these data be read out. The data are then read from the data buffer registers by the interrupt service program.

When the internal clock is used, and the previous data are not read from the data buffer register before the next data are received, the serial clock will stop and an automatic-wait will be initiated until the data are read. A wait will not be initiated if even one data word has been read.

Note: Waits are also canceled by reading a DBR not being used as a received data buffer register is read; therefore, during SIO do not use such DBR for other applications.

When an external clock is used, the shift operation is synchronized with the external clock; therefore, the previous data are read before the next data are transferred to the data buffer register. If the previous data have not been read, the next data will not be transferred to the data buffer register and the receiving of any more data will be canceled. When an external clock is used, the maximum transfer speed is determined by the delay between the time when the interrupt request is generated and when the data received have been read.

The receiving is ended by clearing SIOCR1<SIOS> to "0" or setting SIOCR1<SIOINH> to "1" in buffer full interrupt service program. When SIOCR1<SIOS> is cleared, the current data are transferred to the buffer. After SIOCR1<SIOS >cleared, the receiving is ended at the time that the final bit of the data has been received. That the receiving has ended can be determined from the status of SIOSR<SIOF>. SIOSR<SIOF> is cleared to "0" when the receiving is ended. After confirmed the receiving termination, the final receiving data is read. When SIOCR1<SIOINH> is set, the receiving is immediately ended and SIOSR<SIOF> is cleared to "0". (The received data is ignored, and it is not required to be read out.)

If it is necessary to change the number of words in external clock operation, SIOCR1<SIOS> should be cleared to "0" then SIOCR2<BUF> must be rewritten after confirming that SIOSR<SIOF> has been cleared to "0".

If it is necessary to change the number of words in internal clock, during automatic-wait operation which occurs after completion of data receiving, SIOCR2<BUF> must be rewritten before the received data is read out.

Note: The buffer contents are lost when the transfer mode is switched. If it should become necessary to switch the transfer mode, end receiving by clearing SIOCR1<SIOS> to "0", read the last data and then switch the transfer mode.

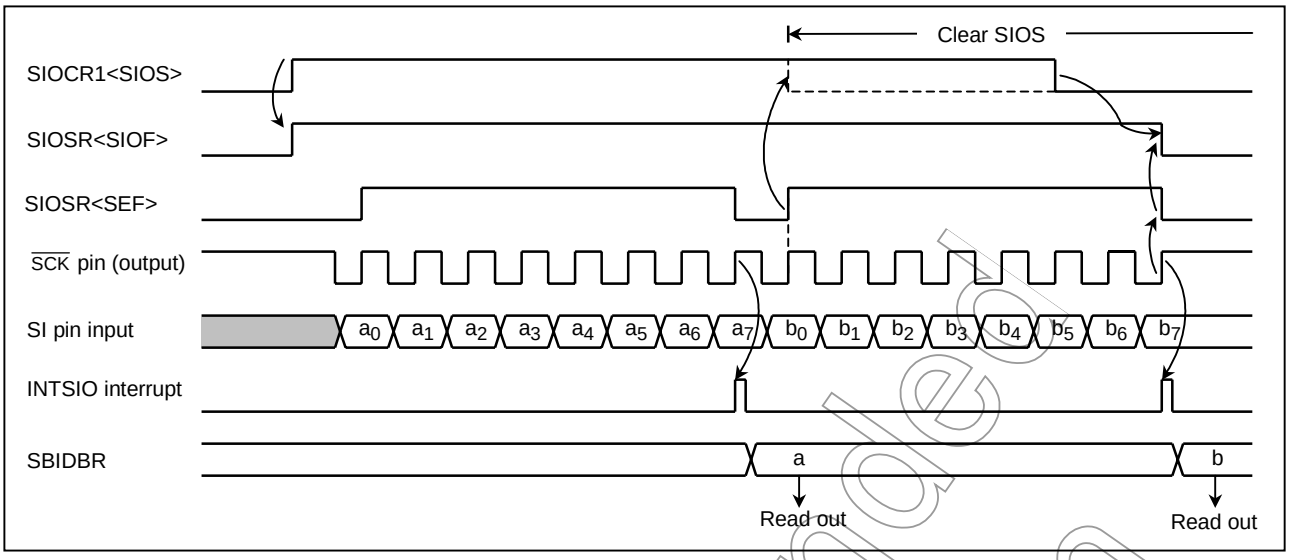


Figure 2.10.9 Receive mode (example: 8-bit, 1 word, internal clock)

(3) 8-bit transmit/receive mode

After setting the control registers to the 8-bit transmit/receive mode, write the data to be transmitted first to the data buffer registers (DBR). After that, enable transceiving by setting <SIOS> to "1". When transmitting, the data are output from the SO pin at leading edges of the serial clock. When receiving, the data are input to the SI pin at the trailing edges of the serial clock. 8-bit data are transferred from the shift register to the data buffer register. An INTSIO interrupt is generated when the number of data words specified with the <BUF> has been transferred. The interrupt service program reads the received data from the data buffer register and then writes the data to be transmitted. The data buffer register is used for both transmitting and receiving; therefore, always write the data to be transmitted after reading the received data.

When the internal clock is used, a wait is initiated until the received data are read and the next data are written. A wait will not be initiated if even one data word has been written.

Note: The wait is also canceld by writing to a DBR not being used as a transmit data buffer registers; therefore, during SIO do not use such DBR for other applications.

When an external clock is used, the shift operation is synchronized with the external clock; therefore, it is necessary to read the received data and write the data to be transmitted next before starting the next shift operation. When an external clock is used, the transfer speed is determined by the maximum delay between generation of an interrupt request and the received data are read and the data to be transmitted next are written.

The transmit/receive operation is ended by clearing SIOCR1<SIOS> to "0" or setting SIOSR<SIOINH> to "1" in interrupt service program.

When SIOCR1<SIOINH> is set, the transmit/receive operation is immediately ended and SIOSR<SIOF> is cleared to "0".

If it is necessary to change the number of words in external clock operation, SIOCR1<SIOS> should be cleared to "0", then SIOCR2<BUF> must be rewritten after confirming that SIOSR<SIOF> has been cleared to "0".

If it is necessary to change the number of words in internal clock, during automatic-wait operation which occurs after completion of transmit/receive operation, SIOCR2<BUF> must be rewritten before reading and writing of the receive/transmit data.

Note: The buffer contents are lost when the transfer mode is switched. If it should become necessary to switch the transfer mode, end receiving by clearing SIOCR1<SIOS> to "0", read the last data and then switch the transfer mode.

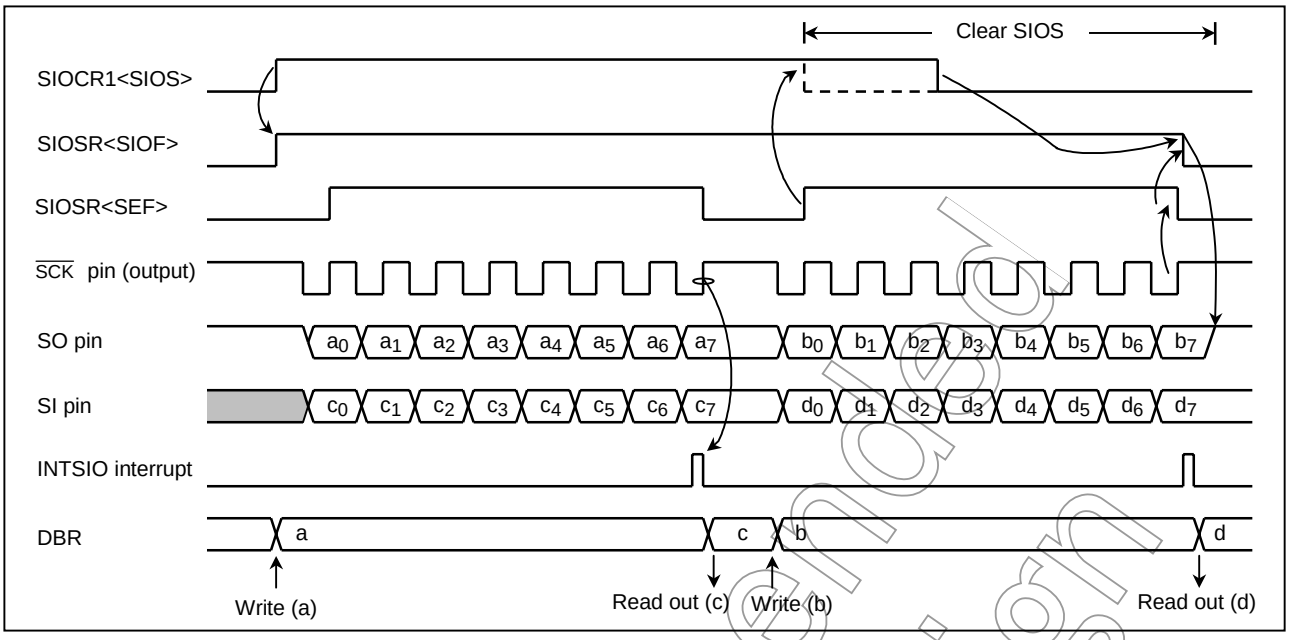


Figure 2.10.10 Transmit/Receive mode (example: 8-bit, 1word, internal clock)

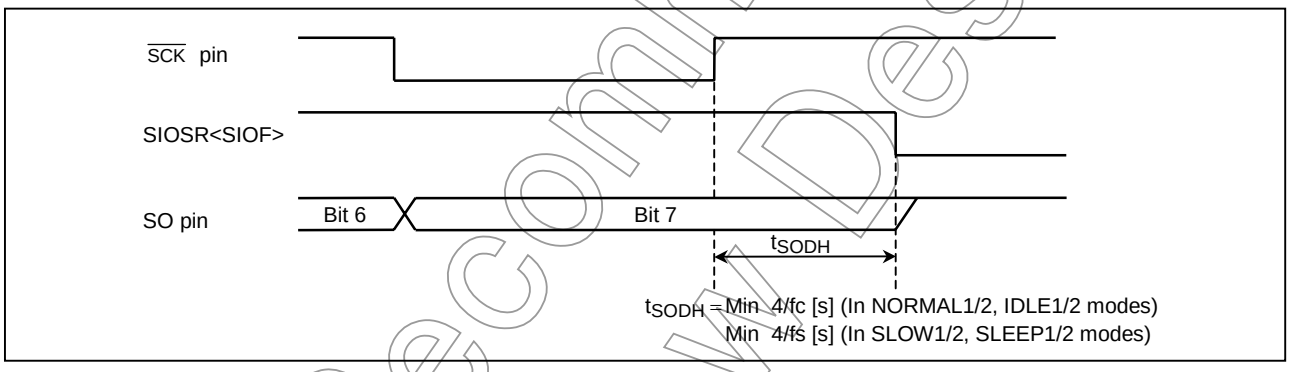


Figure 2.10.11 Transmitted data hold time at end of transmit/receive

2.11 10-bit AD Converter (ADC)

The TMP86FM29 have a 10-bit successive approximation type AD converter.

2.11.1 Configuration

The circuit configuration of the 10-bit AD converter is shown in Figure 2.11.1.

It consists of control registers ADCCR1 and ADCCR2, registers ADCDR1 and ADCDR2, a DA converter, a sample-and-hold circuit, a comparator, and a successive comparison circuit.

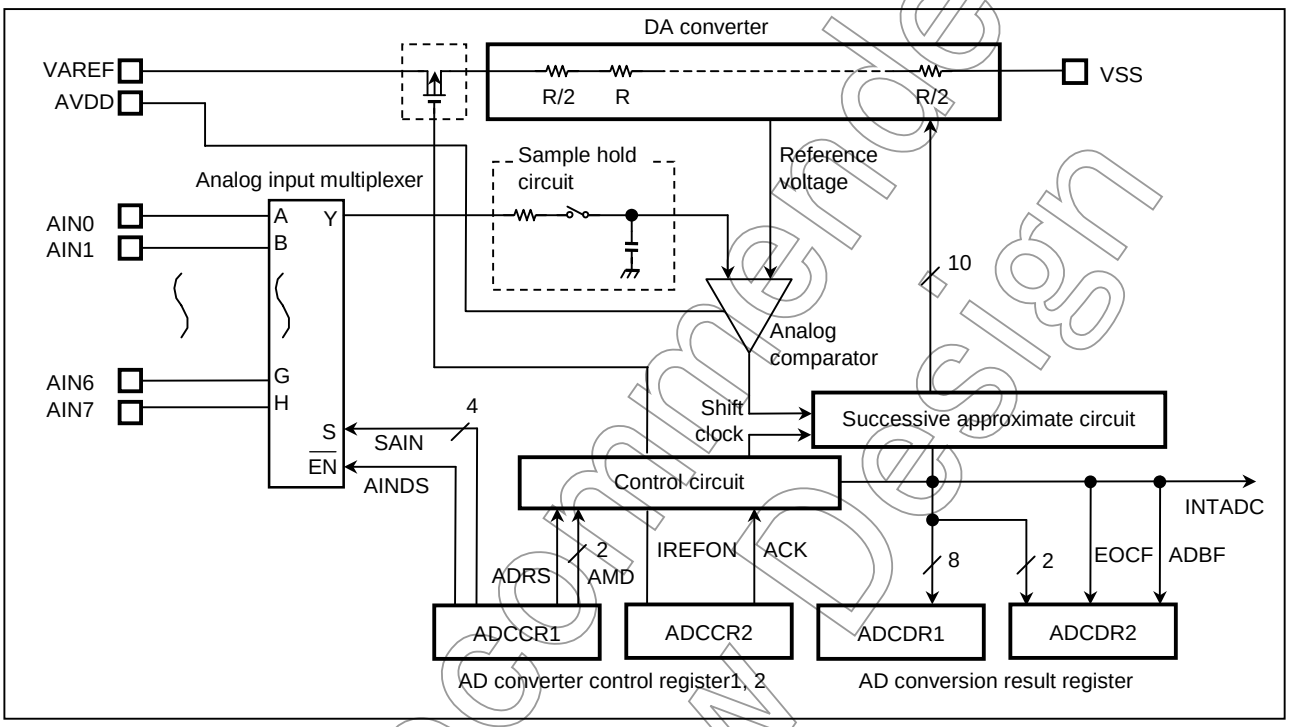


Figure 2.11.1 AD converter (ADC)

2.11.2 Register Configuration

The AD converter consists of the following four registers:

- AD converter control register 1 (ADCCR1)
- AD converter control register 2 (ADCCR2)
- AD converted value register 1/2 (ADCDR1/ADCDR2)

(1) AD converter control register 1 (ADCCR1)

This register selects the analog channels and operation mode (software start or repeat) in which to perform AD conversion and controls the AD converter as it starts operating.

(2) AD converter control register 2 (ADCCR2)

This register selects the AD conversion time and controls the connection of the DA converter (ladder resistor network).

(3) AD converted value register (ADCDR1)

This register is used to store the digital value (Bit9 to bit2) after being converted by the AD converter.

(4) AD converted value register (ADCDR2)

This register is used to store the digital value (Bit1 and bit0) after being converted by the AD converter, and then this register is also used to monitor the operating status of the AD converter.

The AD converter control register configurations are shown in Figure 2.11.2 and Figure 2.11.3.

AD Converter Control Register 1

ADCCR1	7	6	5	4	3	2	1	0	
($000E_H$)	ADRS	AMD		AINDS	SAIN				(Initial value: 0001 0000)

ADRS	AD conversion start	0: – 1: Start	R/W
AMD	AD Operating mode	00: AD operation disable 01: Software start mode 10: Reserved 11: Repeat mode	
AINDS	Analog input control	0: Analog input enable 1: Analog input disable	
SAIN	Analog input channel select	0000: Selects AIN0 0001: Selects AIN1 0010: Selects AIN2 0011: Selects AIN3 0100: Selects AIN4 0101: Selects AIN5 0110: Selects AIN6 0111: Selects AIN7 1***: Reserved	

Note 1: Select analog input when AD converter stops (ADCDR2<ADBF> = “0”).
Note 2: When the analog input is all use disabling, the AINDS should be set to “1”.
Note 3: During conversion, do not perform output instruction to maintain a precision for all of the pins. And port near to analog input, do not input intense signaling of change.
Note 4: The ADRS is automatically cleared to “0” after starting conversion.
Note 5: Do not set ADRS newly again during AD conversion. Before setting ADRS newly again, check ADCDR<EOCF> to see that the conversion is completed or wait until the interrupt signal (INTADC) is generated (e.g., interrupt handling routine).
Note 6: After STOP or SLOW mode are started, AD converter control register 1 (ADCCR1) is all initialized. Therefore, set the ADCCR1 newly again after exiting these modes.

AD Converter Control Register 2

ADCCR2	7	6	5	4	3	2	1	0	
($000F_H$)			IREFON	“1”		ACK		“0”	(Initial value: **00 0000)

IREFON	DA converter (ladder resistor) connection control	Inputting current to the ladder resistor 0: Connected only during AD conversion 1: Always connected							R/W
ACK	AD conversion time select	ACK	Conversion time	fc = 16 MHz	fc = 8 MHz	fc = 4 MHz	fc = 2 MHz		
		000	39/fc	–	–	–	19.5 μs		
		001	Reserved						
		010	78/fc	–	–	19.5 μs	39.0 μs		
		011	156/fc	–	19.5 μs	39.0 μs	156.0 μs		
		100	312/fc	19.5 μs	39.0 μs	156.0 μs	–		
		101	624/fc	39.0 μs	78.0 μs	–	–		
		110	1248/fc	78.0 μs	156.0 μs	–	–		
		111	Reserved						

Note 1: Settings for “–” in the above table are inhibited.
Note 2: Set conversion time by analog reference voltage (V_{AREF}) as follows.
V_{AREF} = 4.5 to 5.5 V (15.6 μ or more)
V_{AREF} = 2.7 to 5.5 V (31.2 μ or more)
V_{AREF} = 1.8 to 5.5 V (124.8 μ or more)
Note 3: Always set bit 0 in ADCCR2 to “0” and set bit 4 in ADCCR2 to “1”.
Note 4: When a read instruction for ADCCR2, bit 6 to 7 in ADCCR2 read in as undefined data.
Note 5: fc; High-frequency clock [Hz]
Note 6: After STOP or SLOW mode are started, AD converter control register 2 (ADCCR2) is all initialized. Therefore, set the ADCCR2 newly again after exiting these modes.

Figure 2.11.2 AD converter control register

AD Conversion Result Register

	7	6	5	4	3	2	1	0	
ADCDR1 (0020_H)	AD09	AD08	AD07	AD06	AD05	AD04	AD03	AD02	(Initial value: 0000 0000)
ADCDR2 (0021_H)	AD01	AD00	EOCF	ADBF					(Initial value: 0000 ****)

EOCF	AD conversion end flag	0: Before or during conversion 1: Conversion completed	Read only
ADBF	AD conversion busy flag	0: During stop of AD conversion 1: During AD conversion	

Note 1: The EOCF is cleared to "0" when reading the ADCDR1.
Therefore, the AD conversion result should be read to ADCDR2 more first than ADCDR1.
Note 2: ADBF is set to "1" when AD conversion starts and cleared to "0" when the AD conversion is finished. It also is cleared upon entering standby mode (STOP or SLOW).
Note 3: If a read instruction is executed for ADCDR2, read data of bit 3 to 0 are unstable.

Figure 2.11.3 AD converter result register

2.11.3 AD Converter Operation

(1) Set up the AD converter control register 1 (ADCCR1) as follows:

- Choose the channel to AD convert using AD input channel select (SAIN).
- Specify analog input enable for analog input control (AINDS).
- Specify AMD for the AD converter control operation mode (software or repeat mode).

(2) Set up the AD converter control register 2 (ADCCR2) as follows:

- Set the AD conversion time using AD conversion time (ACK). For details on how to set the conversion time, refer to Note 2 for AD converter control register 2.
- Choose IREFON for DA converter control.

(3) After setting up (1) and (2) above, set AD conversion start (ADRS) of AD converter control register 1 (ADCCR1) to "1".

(4) After an elapse of the specified AD conversion time, the AD converted value is stored in AD conversion result register 1 (ADCDR1), AD conversion result register (ADCDR2) and then the AD conversion end flag (EOCF) of AD conversion result register 2 (ADCDR2) is set to "1", upon which time AD conversion interrupt INTADC is generated.

(5) EOCF is cleared to "0" by a read of the conversion result. However, if reconverted before a register read, although EOCF is cleared the previous conversion result is retained until the next conversion is completed.

2.11.4 AD Converter Operation Modes

There are following two AD converter operation modes:

- Software start: AD conversion is performed once by setting AMD to “01B” and ADRS to “1”.
- Repeat mode: AD conversion is performed repeatedly by setting AMD to “11B” and ADRS to “1”.

(1) Software start mode

After setting ADCCR1<AMD> to “01” (Software start mode), set ADCCR1<ADRS> to “1”. AD conversion of the voltage at the analog input pin specified by ADCCR1<SAIN> is thereby started.

After completion of the AD conversion, the conversion result is stored in AD conversion result registers (ADCDR1, ADCDR2) and at the same time ADCDR2<EOCF> is set to “1”, the AD conversion finished interrupt (INTADC) is generated.

ADCCR1<ADRS> is automatically cleared to “0” after AD conversion has started. Do not set ADCCR1<ADRS> newly again (Restart) during AD conversion. Before setting ADCCR1<ADRS> newly again, check ADCDR2<EOCF> to see that the conversion is completed or wait until the interrupt signal (INTADC) is generated (e.g., interrupt handling routine).

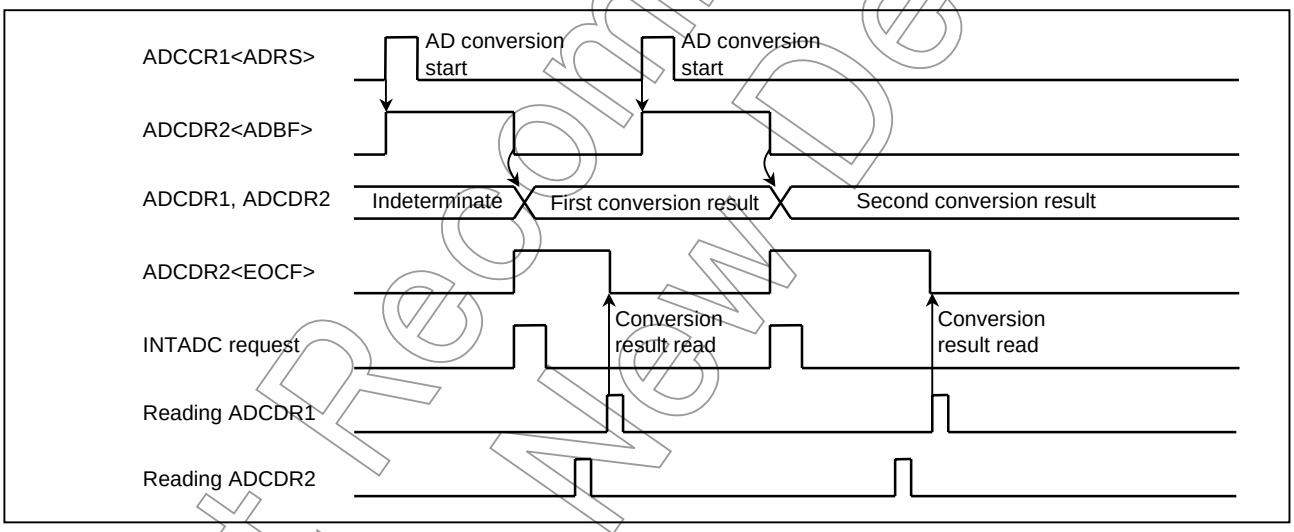


Figure 2.11.4 Operation in software start mode

Example: After selecting the conversion time of 19.5 μs at 16 MHz and the analog input channel AIN3 pin, perform AD conversion once. After checking EOCF, read the converted value, store the lower 2 bits in address 009EH and store the upper 8 bits in address $009F_H$ on RAM. The operation mode is software start mode.

```
; AIN SELECT
LD      (P6CR1), 00000000B      ;   P6CR1 bit 3 = 0
LD      (P6CR2), 00000000B      ;   P6CR2 bit 3 = 0
LD      (ADCCR1), 00100011B     ;   Select AIN3
LD      (ADCCR2), 11011000B     ;   Select conversion time (312/fc) and
                                    operation mode

; AD CONVERT START
SET     (ADCCR1). 7             ;   ADRS = 1
SLOOP:  TEST    (ADCDR2). 5     ;   EOCF = 1 ?
        JRS     T, SLOOP
; RESULT DATA READ
        LD      A, (ADCDR2)
        LD      (9EH), A
        LD      A, (ADCDR1)
        LD      (9FH), A
```

(2) Repeat mode

AD conversion of the voltage at the analog input pin specified by ADCCR1<SAIN> is performed repeatedly. In this mode, AD conversion is started by setting ADCCR1<ADRS> to "1" after setting ADCCR1<AMD> to "11".

After completion of the AD conversion, the conversion result is stored in AD conversion result registers (ADCDR1, ADCDR2) and at the same time ADCDR2<EOCF> is set to "1", the AD conversion finished interrupt (INTADC) is generated.

In repeat mode, each time one AD conversion is completed, the next AD conversion is started. To stop AD conversion, set ADCCR1<AMD> to "00B" (Disable mode). The AD convert operation is stopped immediately. The converted value at this time is not stored in the AD conversion result register.

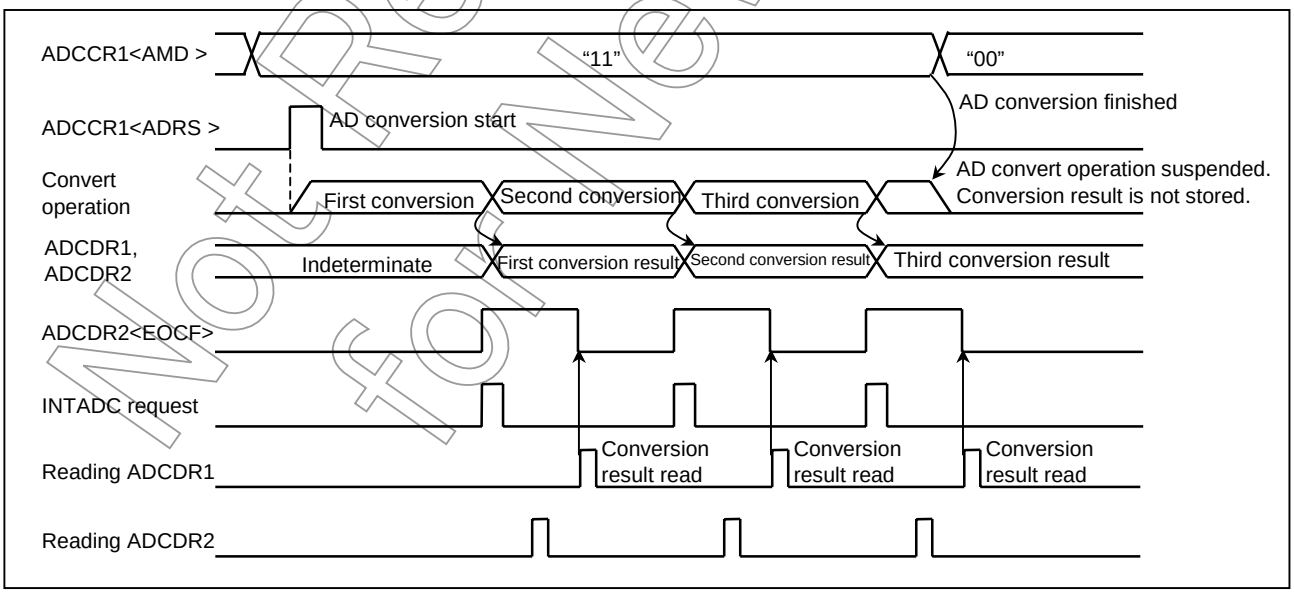


Figure 2.11.5 Operation in repeat mode

2.11.5 STOP and SLOW Modes during AD Conversion

When the STOP or SLOW mode is entered forcibly during AD conversion, the AD convert operation is suspended and the AD converter is initialized (ADCCR1 and ADCCR2 are initialized to initial value). Also, the conversion result is indeterminate. (Conversion results up to the previous operation are cleared, so be sure to read the conversion results before entering STOP or SLOW mode.) When released from STOP or SLOW mode, AD conversion is not automatically restarted. Therefore, when the AD converter is used again, it is necessary to restart AD conversion (Set ADCCR1<ADRS> to "1"). Note that since the analog reference voltage is automatically disconnected, there is no possibility of current flowing into the analog reference voltage.

2.11.6 Analog Input Voltage and AD Conversion Result

The analog input voltage is corresponded to the 10-bit digital value converted by the AD as shown in Figure 2.11.6.

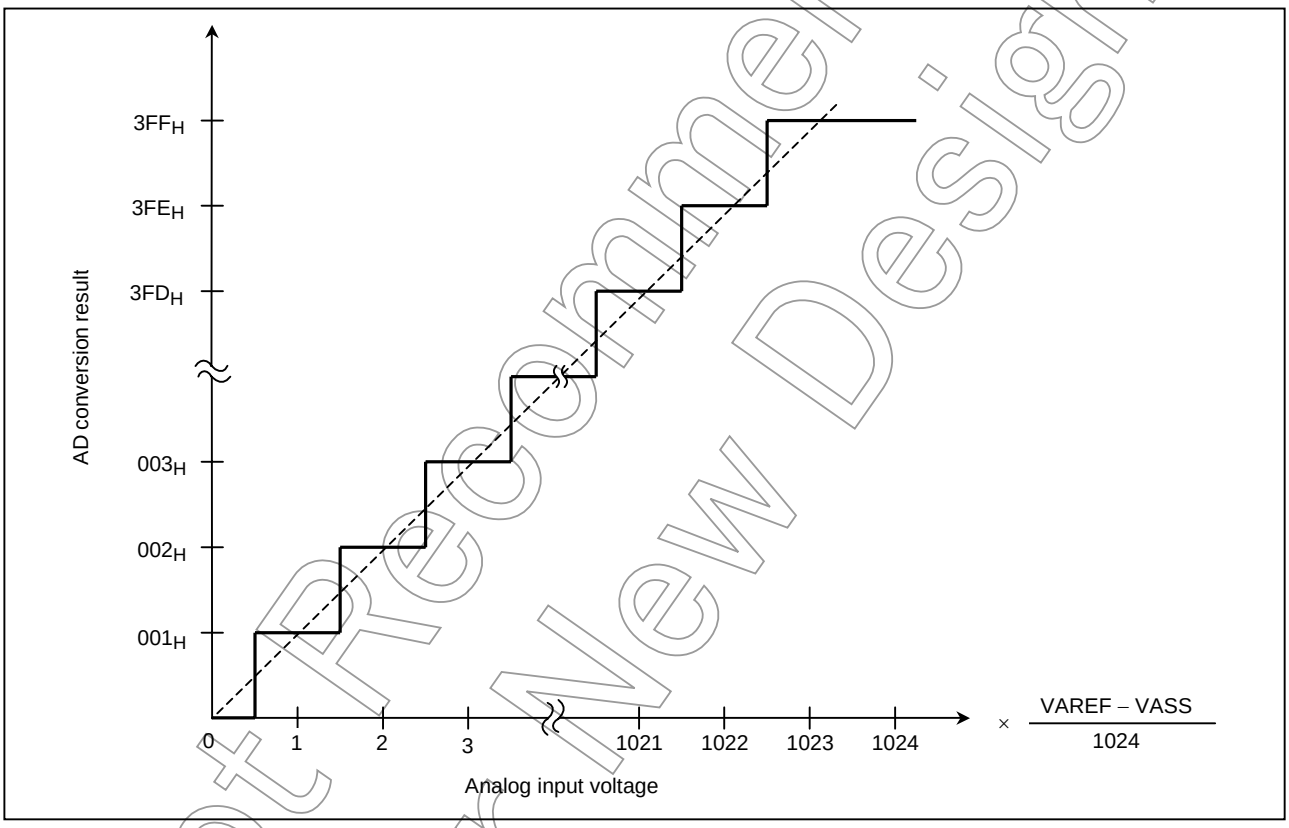


Figure 2.11.6 Analog input voltage and AD conversion result (typ.)

2.11.7 Precautions about AD Converter

(1) Analog input pin voltage range

Make sure the analog input pins (AIN0 to AIN7) are used at voltages within VSS below VAREF. If any voltage outside this range is applied to one of the analog input pins, the converted value on that pin becomes uncertain. The other analog input pins also are affected by that.

(2) Analog input shared pins

The analog input pins (AIN0 to AIN7) are shared with input/output ports. When using any of the analog inputs to execute AD conversion, do not execute input/output instructions for all other ports. This is necessary to prevent the accuracy of AD conversion from degrading. Not only these analog input shared pins, some other pins may also be affected by noise arising from input/output to and from adjacent pins.

(3) Noise countermeasure

The internal equivalent circuit of the analog input pins is shown in Figure 2.11.7. The higher the output impedance of the analog input source, more easily they are susceptible to noise. Therefore, make sure the output impedance of the signal source in your design is 5 kΩ or less. Toshiba also recommends attaching a capacitor external to the chip.

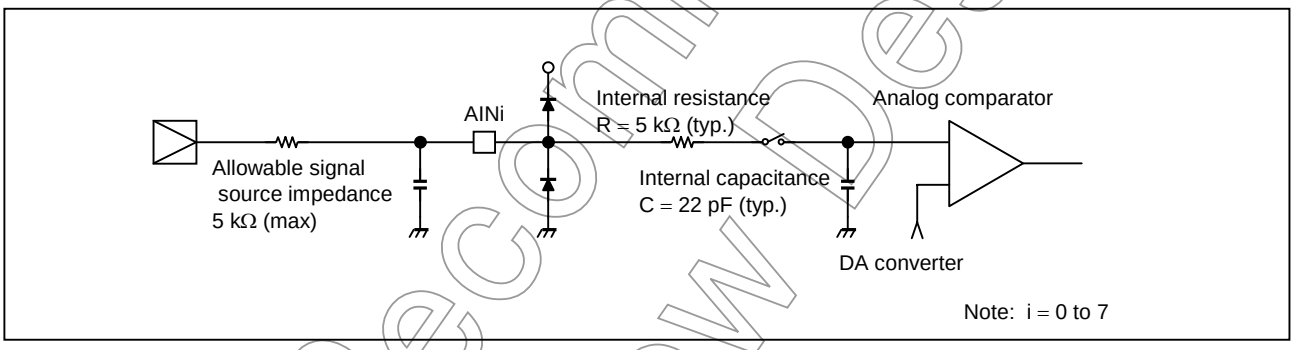


Figure 2.11.7 Analog input equivalent circuit and example of input pin processing

2.12 Key-On Wake-Up (KWU)

In the TMP86FM29, the STOP mode must be released by not only P20 ($\overline{INT5}$ / $\overline{STOP}$) pin but also P64 to P67 pins.

When the STOP mode is released by P64 to P67 pins, the P20 ($\overline{INT5}$ / $\overline{STOP}$) pin needs to be used.

2.12.1 Configuration

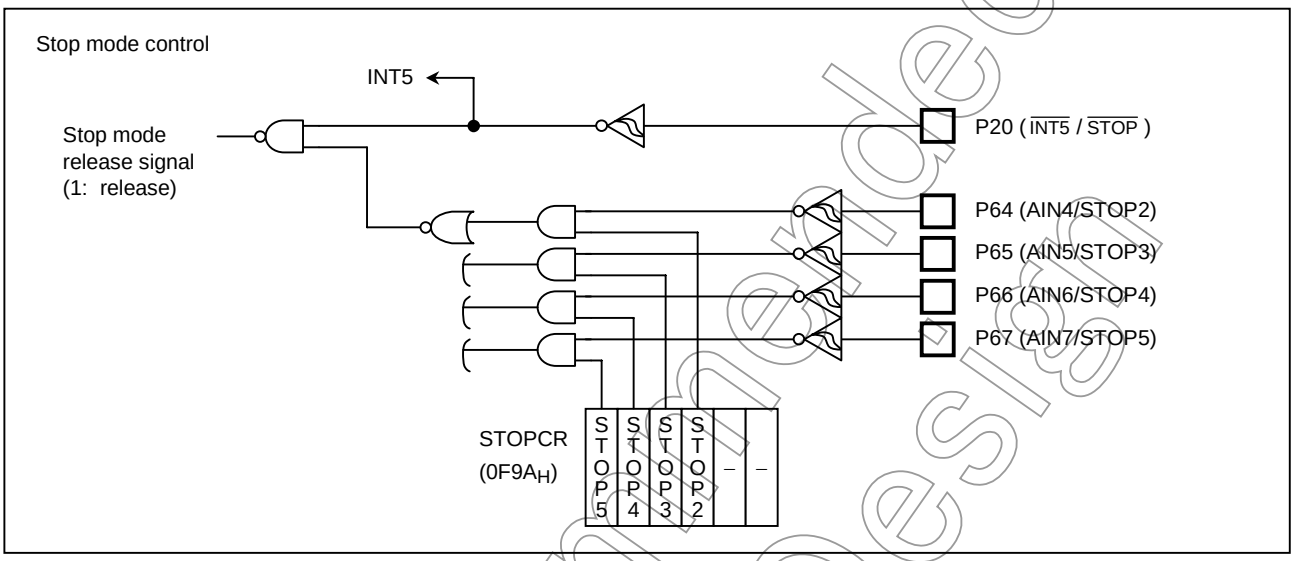


Figure 2.12.1 Key-on wake up circuit

Note: $\overline{STOP}$ pin doesn't have the control register such as STOPCR, so when STOP mode is relesed by STOPx (x; 2 to 5), $\overline{STOP}$ pin should be used as STOP function.

2.12.2 Control

P64 to P67 (STOP2 to STOP5) pin can controlled by Key-On Wake-Up control register (STOPCR). It can be configured as enable/disable in one-bit unit. When those pins are used by STOP mode release, those pins must be set input mode (P6CR, P6DR, ADCCR1).

STOP mode can be entered by setting up the System Control Register1 (SYSCR1), and can be exited by detecting the falling edge on STOP2 to 5 pins, which are enabled by STOPCR, for releasing STOP mode (Note 1). Also, because each level of the STOP2 to 5 can be confirmed by reading P6DR, check all STOP2 to 5 pins that is enabled by STOPCR before the STOP mode is started (Note 2).

Note 1: When the STOP mode release by edge mode (SYSCR1<RELM> = "0"), prohibit input from STOP2 to STOP5 or must be set "1" level into STOP2 to STOP5 pins.

Note 2: When the $\overline{\text{STOP}}$ pin input is high or STOP2 to STOP5 pin input which is enabled by STOPCR is low, executing an instruction which starts STOP mode will not place in STOP mode but instead will immediately start the release sequence (warm-up).

Table 2.12.1 Input Edge (Level) of Stop Mode Release

Terminal name	As both terminal	SYSCR1<RELM> = "1"	SYSCR1<RELM> = "0"
		Release edge (Level)	
STOP	P20/$\overline{\text{INT5}}$	"H" level (Note2)	Rising edge
STOP2	P64/AIN4	"L" level (Note 2)	Do not use key on wake up function (Note 1)
STOP3	P65/AIN5		
STOP4	P66/AIN6		
STOP5	P67/AIN7		

Key-On Wake-Up control register

STOPCR ($0F9A_H$)

7	6	5	4	3	2	1	0
STOP5	STOP4	STOP3	STOP2	–	–	–	–

(Initial value: 0000 ****)

STOP2	Stop mode released by P64 port	0: Disable 1: Enable	Write only
STOP3	Stop mode released by P65 port	0: Disable 1: Enable	
STOP4	Stop mode released by P66 port	0: Disable 1: Enable	
STOP5	Stop mode released by P67 port	0: Disable 1: Enable	

Figure 2.12.2 Key-on wake-up control register

2.13 LCD Driver

The TMP86FM29 has a driver and control circuit to directly drive the liquid crystal device (LCD). The pins to be connected to LCD are as follows:

a. Segment output port 8 pins (SEG7 to SEG0)
b. Segment output or P1, P5, P7 input/output port 24 pins (SEG31 to SEG8)
c. Common output port 4 pins (COM3 to COM0)

In addition, C0, C1, V1, V2, V3 pin are provided for the LCD driver's booster circuit.

The devices that can be directly driven is selectable from LCD of the following drive methods:

a. 1/4 Duty (1/3 Bias) LCD Max 128 Segments (8 segments × 16 digits)
b. 1/3 Duty (1/3 Bias) LCD Max 96 Segments (8 segments × 12 digits)
c. 1/2 Duty (1/2 Bias) LCD Max 64 Segments (8 segments × 8 digits)
d. Static LCD Max 32 Segments (8 segments × 4 digits)

2.13.1 Configuration

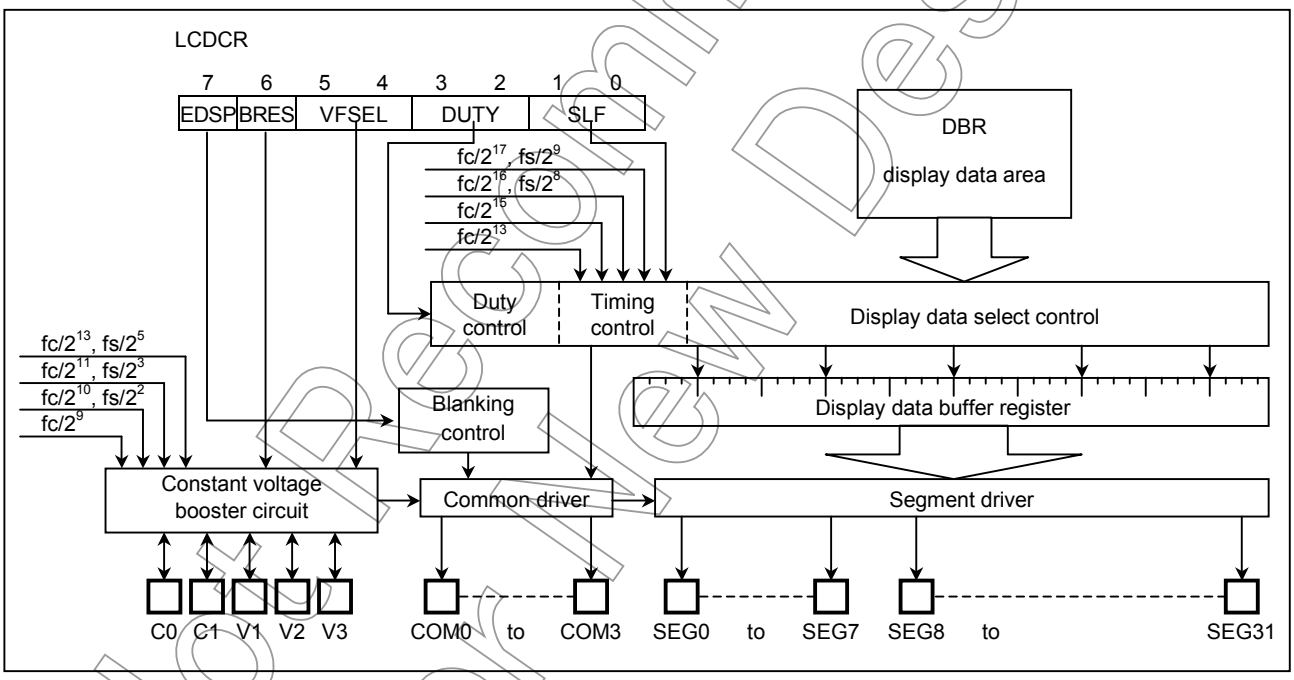


Figure 2.13.1 LDC driver

2.13.2 Control

The LCD driver is controlled using the LCD control register (LCDCR). The LCD driver's display is enabled using the EDSP.

LCDCR (0028_H)

7	6	5 4	3 2	1 0
EDSP	BRES	VFSEL	DUTY	SLF

(Initial value: 0000 0000)

			NORMAL1/2, IDLE1/2 modes		SLOW1/2	
			DV7CK=0	DV7CK=1	SLEEP1/2 modes	
SLF	Selection of LCD frame frequency [Hz]	00	$fc/2^{17}$	$fs/2^{9}$	$fs/2^{9}$	R/W
		01	$fc/2^{16}$	$fs/2^{8}$	$fs/2^{8}$	
		10	$fc/2^{15}$	$fc/2^{15}$	-	
		11	$fc/2^{13}$	$fc/2^{13}$	-	
DUTY	Selection of driving methods	00: 1/4 Duty (1/3 Bias) 01: 1/3 Duty (1/3 Bias) 10: 1/2 Duty (1/2 Bias) 11: Static				
			NORMAL1/2, IDLE1/2 modes		SLOW1/2	
			DV7CK=0	DV7CK=1	SLEEP1/2 modes	
VFSEL	Selection of boost frequency [Hz]	00	$fc/2^{13}$	$fs/2^{5}$	$fs/2^{5}$	
		01	$fc/2^{11}$	$fs/2^{3}$	$fs/2^{3}$	
		10	$fc/2^{10}$	$fs/2^{2}$	$fs/2^{2}$	
		11	$fc/2^{9}$	$fc/2^{9}$	-	
BRES	Booster circuit control	0: Disable (use divider resistance) 1: Enable				
EDSP	LCD Display Control	0: Blanking 1: Enables LCD display (Blanking is released)				

Note1: When <BRES>(Booster circuit control) is set to "0", $V_{DD} \geq V_3 \geq V_2 \geq V_1 \geq V_{SS}$ should be satisfied.
When <BRES> is set to "1", $5.5\ [V] \geq V_3 \geq V_{DD}$ should be satisfied.
If these conditions are not satisfied, it not only affects the quality of LCD display but also may damage the device due to over voltage of the port.

Note2: - ; Do not set the value.

Figure 2.13.2 LCD driver control register

(1) LCD driving methods

As for LCD driving method, 4 types can be selected by LCDCR<DUTY>. The driving method is initialized in the initial program according to the LCD used.

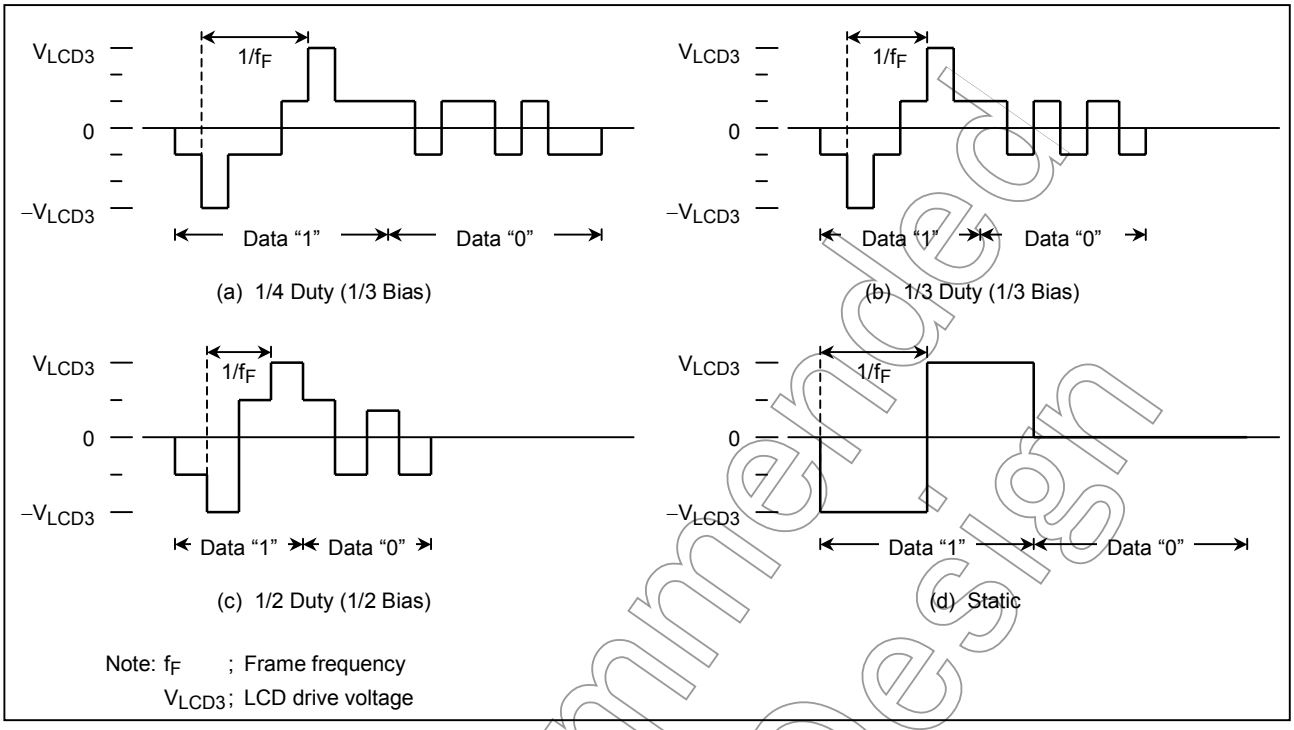


Figure 2.13.3 LCD drive waveform (COM-SEG pins)

(2) Frame frequency

Frame frequency (f_F) is set according to driving method and base frequency as shown in the following Table 2.13.1. The base frequency is selected by LCDCR<SLF> according to the frequency fc and fs of the basic clock to be used.

Table 2.13.1 Setting of LCD frame frequency

a. At the single clock mode. At the dual clock mode (DV7CK = 0).

SLF	Base frequency [Hz]	Frame frequency [Hz]			
		1/4 Duty	1/3 Duty	1/2 Duty	Static
00	$\frac{fc}{2^{17}}$	$\frac{fc}{2^{17}}$	$\frac{4}{3} \cdot \frac{fc}{2^{17}}$	$\frac{4}{2} \cdot \frac{fc}{2^{17}}$	$\frac{fc}{2^{17}}$
	(fc = 16 MHz)	122	163	244	122
	(fc = 8 MHz)	61	81	122	61
01	$\frac{fc}{2^{16}}$	$\frac{fc}{2^{16}}$	$\frac{4}{3} \cdot \frac{fc}{2^{16}}$	$\frac{4}{2} \cdot \frac{fc}{2^{16}}$	$\frac{fc}{2^{16}}$
	(fc = 8 MHz)	122	163	244	122
	(fc = 4 MHz)	61	81	122	61
10	$\frac{fc}{2^{15}}$	$\frac{fc}{2^{15}}$	$\frac{4}{3} \cdot \frac{fc}{2^{15}}$	$\frac{4}{2} \cdot \frac{fc}{2^{15}}$	$\frac{fc}{2^{15}}$
	(fc = 4 MHz)	122	163	244	122
	(fc = 2 MHz)	61	81	122	61
11	$\frac{fc}{2^{13}}$	$\frac{fc}{2^{13}}$	$\frac{4}{3} \cdot \frac{fc}{2^{13}}$	$\frac{4}{2} \cdot \frac{fc}{2^{13}}$	$\frac{fc}{2^{13}}$
	(fc = 1 MHz)	122	163	244	122

Note: fc; High-frequency clock [Hz]

b. At the dual clock mode (DV7CK = 1 or SYSCK = 1)

SLF	Base frequency [Hz]	Frame frequency [Hz]			
		1/4 Duty	1/3 Duty	1/2 Duty	Static
00	$\frac{fs}{2^{9}}$	$\frac{fs}{2^{9}}$	$\frac{4}{3} \cdot \frac{fs}{2^{9}}$	$\frac{4}{2} \cdot \frac{fs}{2^{9}}$	$\frac{fs}{2^{9}}$
	(fs = 32.768 kHz)	64	85	128	64
01	$\frac{fc}{2^{8}}$	$\frac{fs}{2^{8}}$	$\frac{4}{3} \cdot \frac{fs}{2^{8}}$	$\frac{4}{2} \cdot \frac{fs}{2^{8}}$	$\frac{fs}{2^{8}}$
	(fs = 32.768 kHz)	128	171	256	128

Note: fs; Low-frequency clock [Hz]

(3) Booster circuit for LCD driver

The LCD voltage booster pin can select the booster circuit or the divider resistance.

The booster circuit control is selected by LCDCR<BRES>.

The booster circuit boosts the output voltage for the segment/common signal by double (V2) and triple (V3) in relation to the on-chip output voltage (1 V typ.).

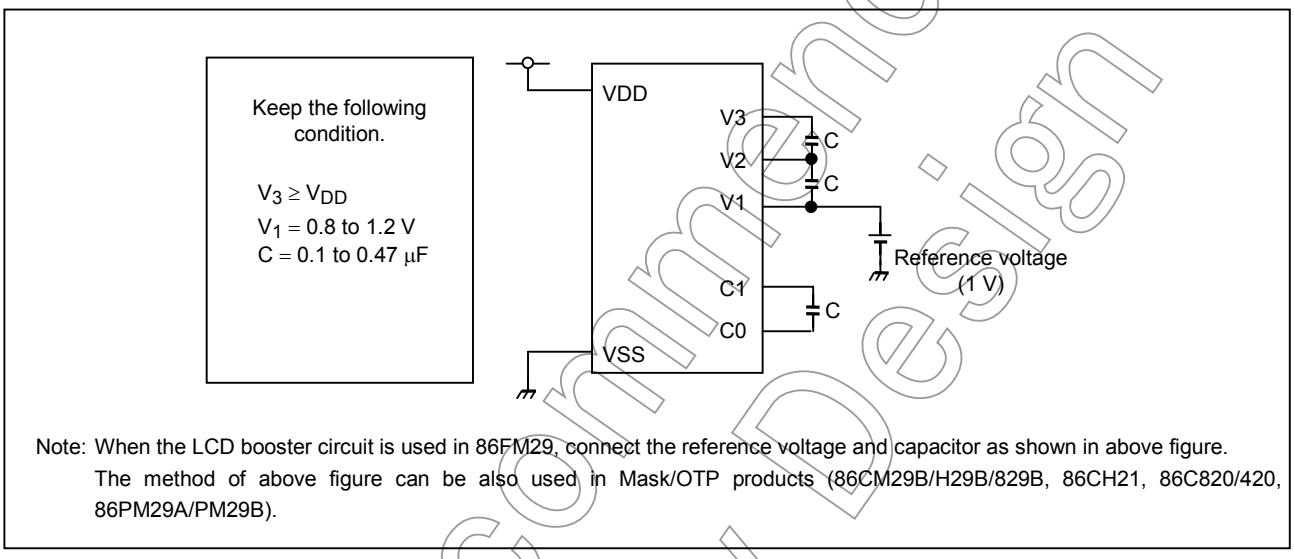
When used as the divider resistance, the V1, V2 and V3 pins are input by divider voltage of external supply.

When used as the booster circuit, the VLCD setting should be composed to 1/3 bias.

The selection of boost frequency is selected by LCDCR<VFSEL>.

Selecting the fast frequency using the SLFR in the command register (LCDCR) raises the drive capability of segment/common.

Table 2.13.2 shows the V3 pin current capacity and Boosting Frequency.



Note: When the LCD booster circuit is used in 86FM29, connect the reference voltage and capacitor as shown in above figure. The method of above figure can be also used in Mask/OTP products (86CM29B/H29B/829B, 86CH21, 86C820/420, 86PM29A/PM29B).

Figure 2.13.4 Example of a booster circuit (LCDCR<BRES> = "1")

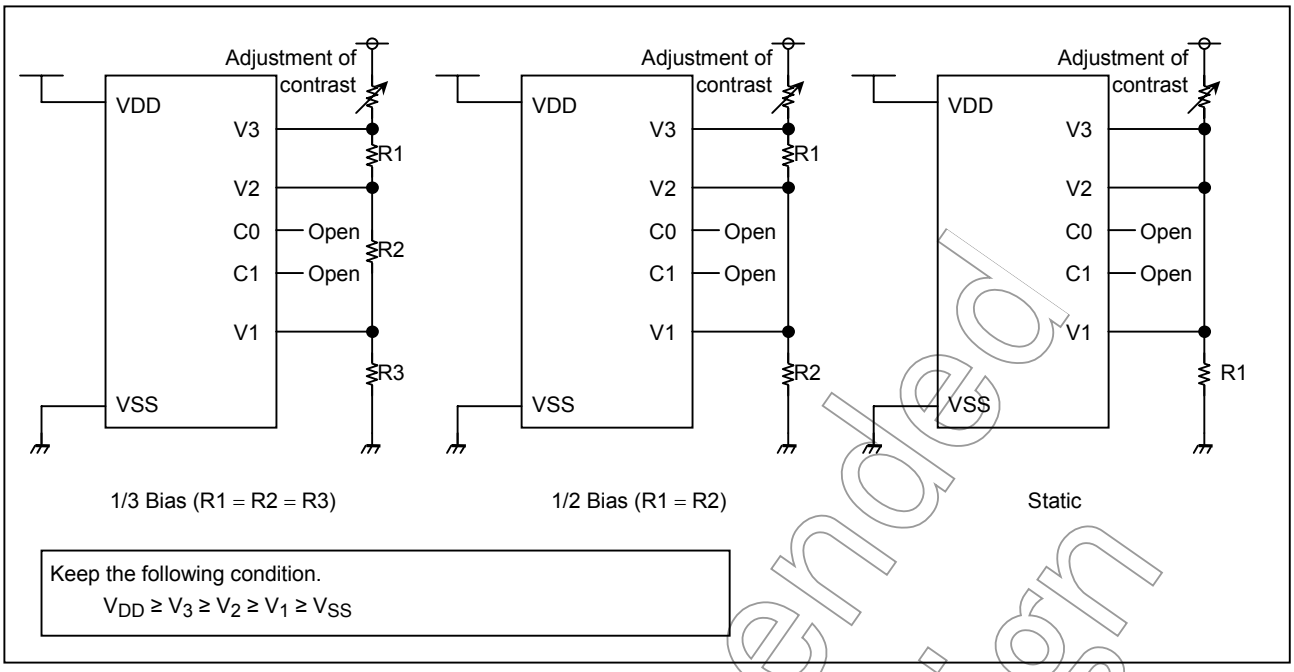


Figure 2.13.5 Example of divider resistance (LCDCR<BRES> = "0")

Table 2.13.2 V3 pin current capacity and boosting frequency (typ.)

VFSEL	Boosting frequency	fc = 16 MHz	fc = 8 MHz	fc = 4 MHz	fs = 32.768 kHz
00	$fc/2^{13}$ or $fs/2^{5}$	−37 mV / μA	−80 mV / μA	−138 mV / μA	−76 mV / μA
01	$fc/2^{11}$ or $fs/2^{3}$	−19 mV / μA	−24 mV / μA	−37 mV / μA	−23 mV / μA
10	$fc/2^{10}$ or $fs/2^{2}$	−17 mV / μA	−19 mV / μA	−24 mV / μA	−18 mV / μA
11	$fc/2^{9}$	−16 mV / μA	−17 mV / μA	−19 mV / μA	−

Note 1: The current capacity is the amount of voltage that falls per 1 μA.

Note 2: The boosting frequency should be selected depending on your LCD panel.

Note 3: For the reference pin V1, a current capacity ten times larger than the above is recommended to ensure stable operation.

2.13.3 LCD Display Operation

(1) Display data setting

Display data is stored to the display data area (assigned to address 0F80 to 0F8FH) in the DBR. The display data which are stored in the display data area is automatically read out and sent to the LCD driver by the hardware. The LCD driver generates the segment signal and common signal according to the display data and driving method. Therefore, display patterns can be changed by only over writing the contents of display data area by the program. Figure 2.13.6 shows the correspondence between the display data area and SEG/COM pins.

LCD light when display data is "1" and turn off when "0". According to the driving method of LCD, the number of pixels which can be driven becomes different, and the number of bits in the display data area which is used to store display data also becomes different.

Therefore, the bits which are not used to store display data as well as the data buffer which corresponds to the addresses not connected to LCD can be used to store general user process data (see Table 2.13.3).

Noto: The display data memory contents become unstable when the power supply is turned on; therefore, the display data memory should be initialized by an initiation routine.

Address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
$0F80_H$	SEG1				SEG0			
81	SEG3				SEG2			
82	SEG5				SEG4			
83	SEG7				SEG6			
84	SEG9				SEG8			
85	SEG11				SEG10			
86	SEG13				SEG12			
87	SEG15				SEG14			
88	SEG17				SEG16			
89	SEG19				SEG18			
8A	SEG21				SEG20			
8B	SEG23				SEG22			
8C	SEG25				SEG24			
8D	SEG27				SEG26			
8E	SEG29				SEG28			
8F	SEG31				SEG30			
	COM3	COM2	COM1	COM0	COM3	COM2	COM1	COM0

Figure 2.13.6 LCD display data area (DBR)

Table 2.13.3 Driving method and bit for display data

Driving methods	bit 7/3	bit 6/2	bit 5/1	bit 4/0
1/4 Duty	COM3	COM2	COM1	COM0
1/3 Duty	–	COM2	COM1	COM0
1/2 Duty	–	–	COM1	COM0
Static	–	–	–	COM0

Note: –; This bit is not used for display data

(2) Blanking

Blanking is enabled when EDSP is cleared to "0".

Blanking turns off LCD through outputting a GND level to SEG/COM pin.

When in STOP mode, EDSP is cleared to "0" and automatically blanked. To redisplay ICD after exiting STOP mode, it is necessary to set EDSP back to "1".

Note: During reset, the LCD segment outputs (SEG0 to SEG7) and LCD common outputs are fixed "0" level. But the multiplex terminal (P1, P5 and P7 ports) of input/output port and LCD segment output becomes high impedance. Therefore, when the reset input is long remarkably, ghost problem may appear in LCD display.

2.13.4 Control Method of LCD Driver

(1) Initial setting

Figure 2.13.7 shows the flowchart of initialization.

Example: To operate a 1/4 duty LCD of 32 segments × 4 commons at frame frequency fc/2^{16} [Hz] and boost frequency fc/2^{13} [Hz].

```
LD    (LCDCR),01000001B    ; Sets LCD driving method and frame
                             frequency.
                             Boost frequency
LD    (P5LCR),0FFH         ; Sets P5 port as segment output .
LD    (P1LCR),0FFH         ; Sets P1 port as segment output .
LD    (P7LCR),0FFH         ; Sets P7 port as segment output .
 ⋮      ⋮                  ; Sets the initial value of display data.
LD    (LCDCR),11000001B    ; Display enable
```

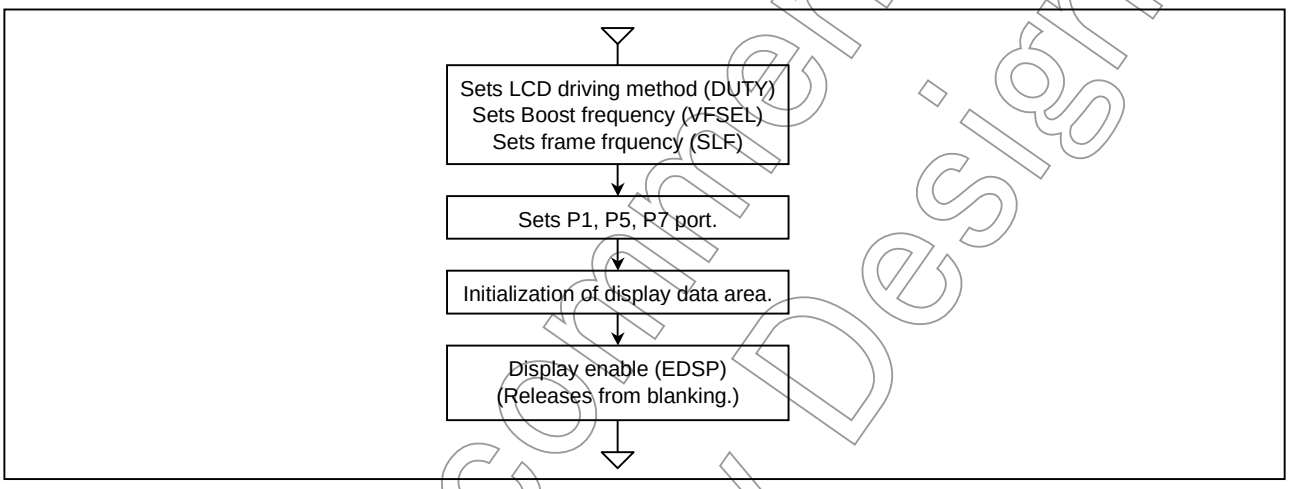


Figure 2.13.7 Initial setting of LCD driver

(2) Store of display data

Generally, display data are prepared as fixed data in program memory and stored in display data area by load command.

Example 1: To display using 1/4 duty LCD a numerical value which corresponds to the LCD data stored in data memory at address 80H (when pins COM and SEG are connected to LCD as in Figure 2.13.8), display data become as shown in Table 2.12.2.

```
        LD      A, (80H)
        ADD     A, TABLE–$–7
        LD      HL, 0F80H
        LD      W, (PC + A)
        LD      (HL), W
        RET
TABLE:  DB      11011111B, 00000110B,
                11100011B, 10100111B,
                00110110B, 10110101B,
                11110101B, 00010111B,
                11110111B, 10110111B
SNEXT:
```

Note: DB is a byte data difinition instruction.

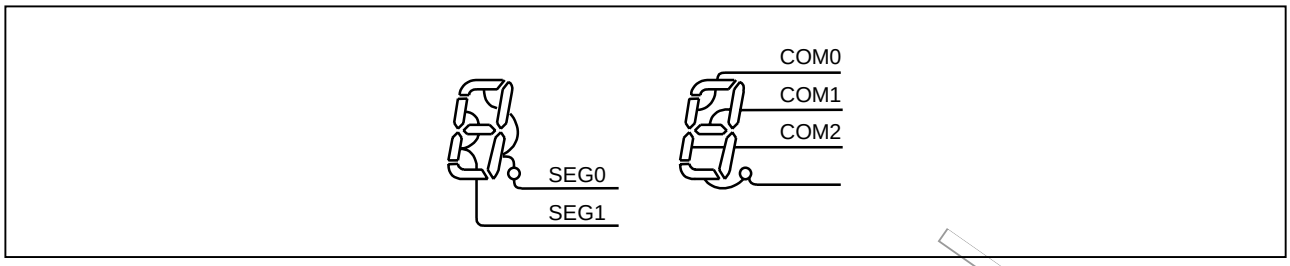


Figure 2.13.8 Example of COM, SEG pin connection (1/4 Duty)

Table 2.13.4 Example of Display Data (1/4 Duty)

No.	Display	Display Data	No.	Display	Display Data
0	0.	11011111	5	5	10110101
1	1	00000110	6	6	11110101
2	2	11100011	7	7	00000111
3	3	10100111	8	8	11110111
4	4	00110110	9	9	10110111

Example 2: Table 2.13.4 shows an example of display data which are displayed using 1/2 duty LCD in the same way as Table 2.13.5. The connection between pins COM and SEG are the same as shown in Figure 2.13.9.

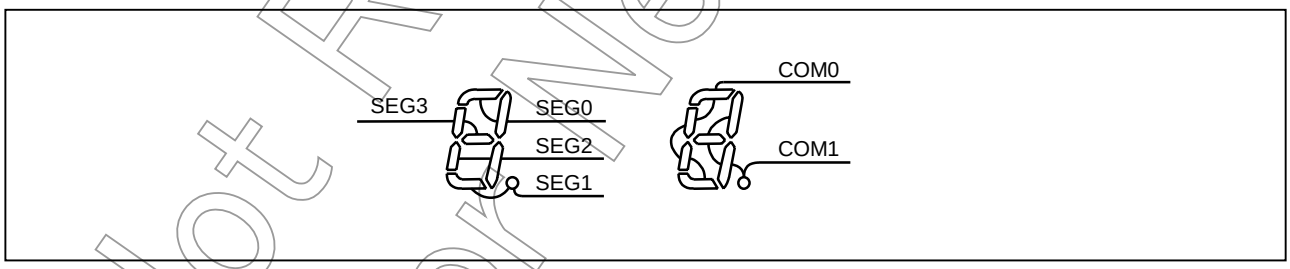


Figure 2.13.9 Example of COM, SEG pin connection

Table 2.13.5 Example of display data (1/2 Duty)

Number	Display data		Number	Display data	
	High order address	Low order address		High order address	Low order address
0	**01**11	**01**11	5	**11**10	**01**01
1	**00**10	**00**10	6	**11**11	**01**01
2	**10**01	**01**11	7	**01**10	**00**11
3	**10**10	**01**11	8	**11**11	**01**11
4	**11**10	**00**10	9	**11**10	**01**11

Note: *; Don't care

(3) Example of LCD drive output

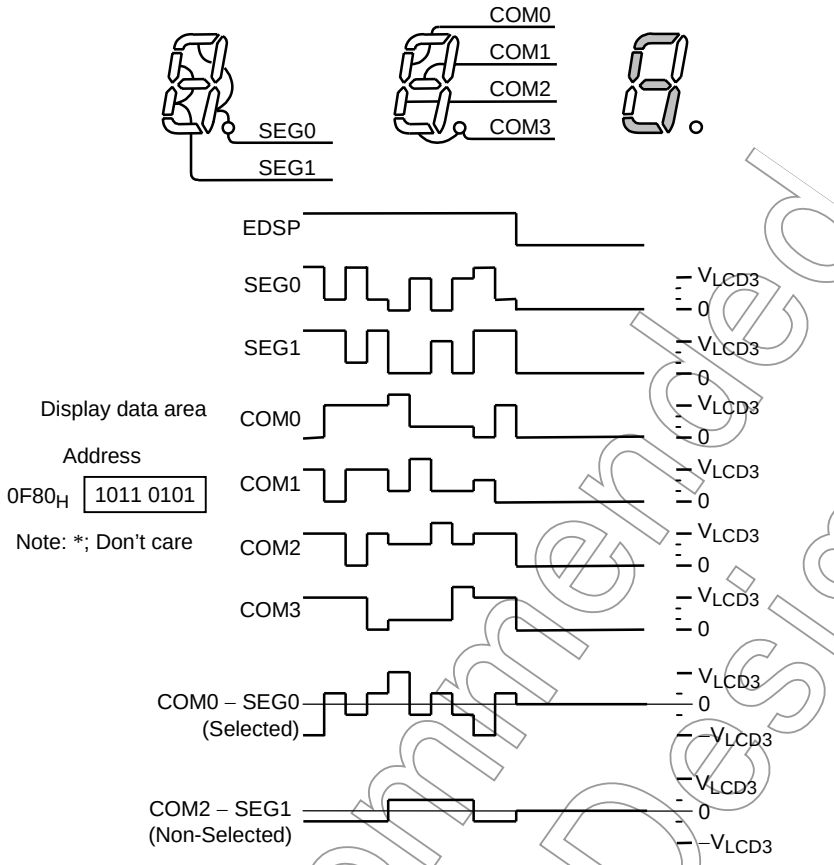


Figure 2.13.10 1/4 duty (1/3 bias) drive

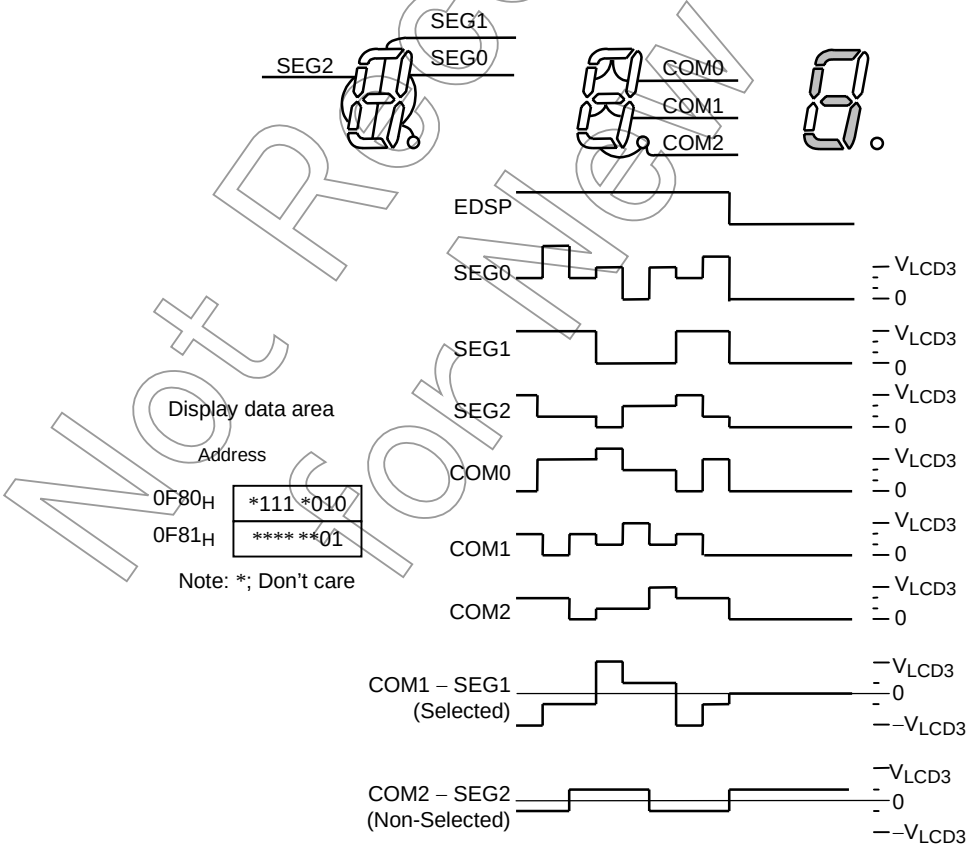


Figure 2.13.11 1/3 duty (1/3 bias) drive

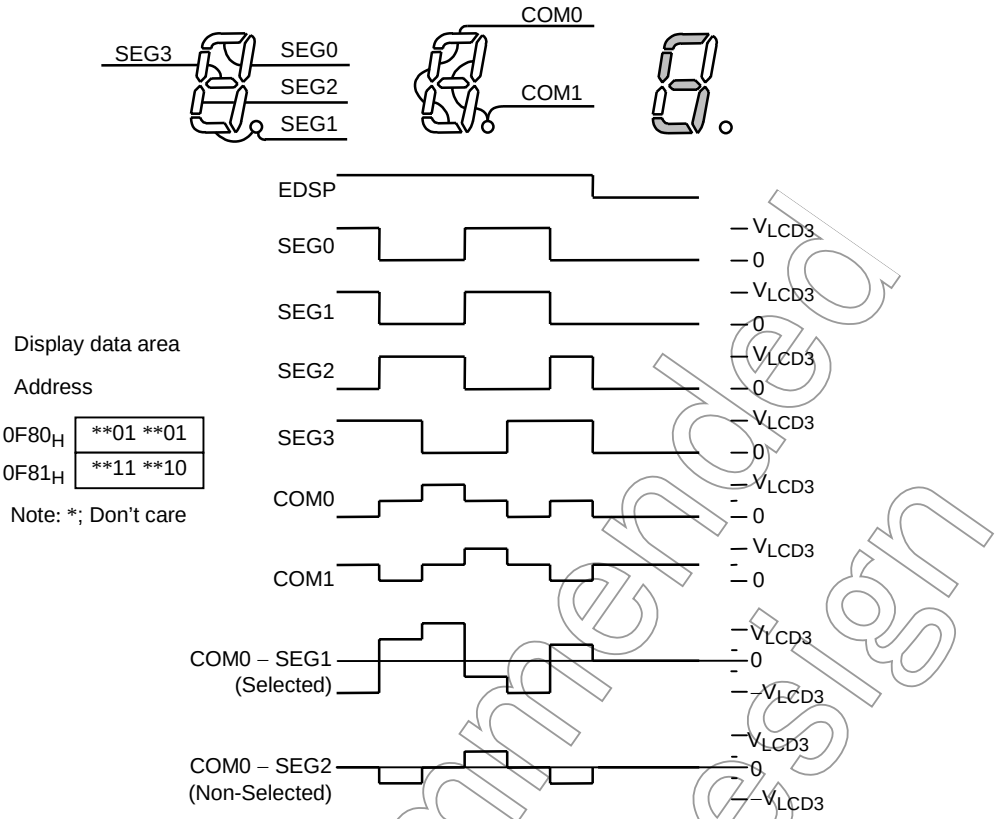


Figure 2.13.12 1/2 duty (1/3 bias) drive

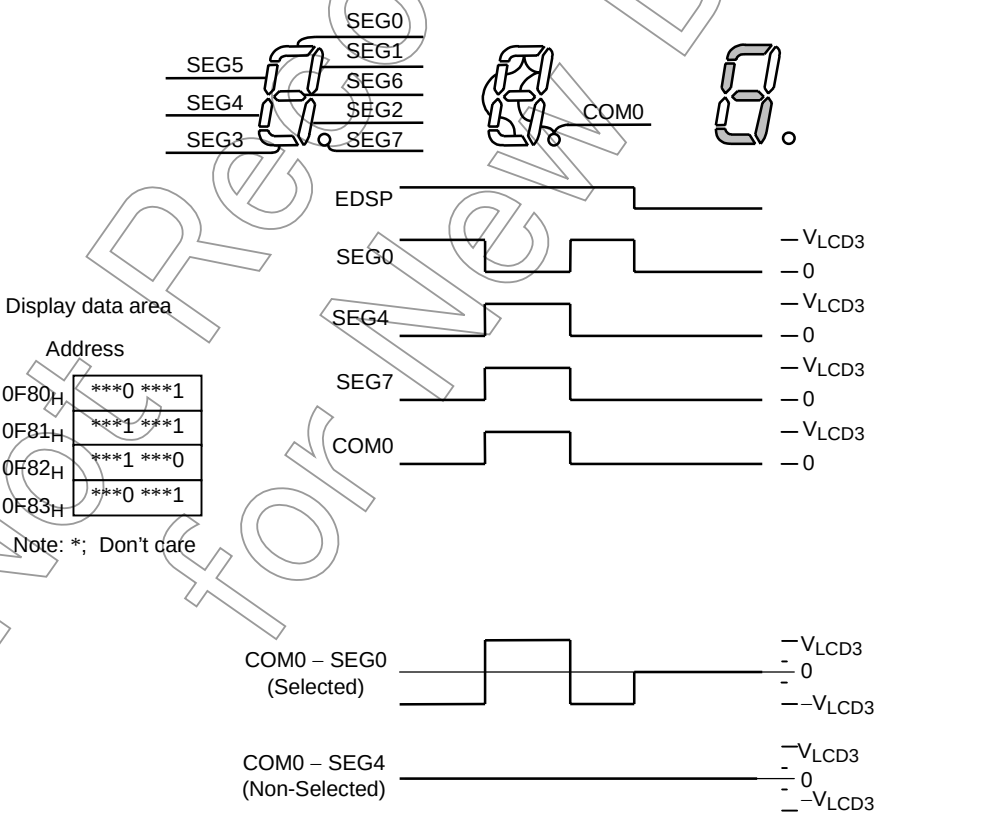


Figure 2.13.13 Static drive

2.14 FLASH Memory

2.14.1 Outline

The TMP86FM29 incorporates 32768 bytes of FLASH memory (Address 8000H to FFFFH). The writing to FLASH memory is controlled by FLASH control register (EEPCR) and FLASH status register (EEPSR).

To write data to the FLASH , execute the Serial PROM mode. For details about the Serial PROM mode, refer to “2.1 Serial PROM mode”.

The FLASH memory of the TMP86FM29 features:

- The FLASH memory is constructed of 512 pages FLASH memory and one page size is 64 bytes (512 pages × 64 bytes = 32768 bytes).
- The TMP86FM29 incorporates a 64-byte temporary data buffer. The data written to FLASH memory is temporarily stored in this data buffer. After 64 bytes data have been written to the temporary data buffer, the writing to FLASH memory automatically starts by page writing (The 64 bytes data are written to specified page of FLASH simultaneously). At the same time, page-by-page erasing occurs automatically. So, it is unnecessary to erase individual pages in advance.
- The FLASH control circuit incorporates an oscillator dedicated to the FLASH. So FLASH writing time is independent of the system clock frequency (fc). In addition, because an FLASH control circuit controls writing time for each FLASH memory cell, the writing time varies in each page (Typically 4 ms per page).
- Controlling the power for the FLASH control circuit (regulator and voltage step-up circuit) achieves low power consumption if the FLASH is not in use (Example: When the program is executed in RAM area).

2.14.2 Conditions for Accessing the FLASH Areas

The conditions for accessing the FLASH areas vary depending on each operation mode. The following tables shows FLASH are access conditions.

Table 2.14.1 FLASH Area Access Conditions

	Area	Operation Mode	
		MCU mode (Note 1)	Serial PROM mode (Note 2)
Flash Memory	8000H to FFFFH	Read/fetch only	Write/read/fetch supported

Note 1: “MCU mode” shows NORMAL1/2 and SLOW1/2 modes.

Note 2: “Serial PROM mode” shows the FLASH controlling mode. For details, refer to “2.1 Serial PROM mode”.

Note 3: “Fetch” means reading operation of FLASH data as an instruction by CPU.

2.14.3 Differences among Product Series

The specifications of the FLASH product (TMP86FM29) are different from those of the emulation chip (TMP86C929A) and the MASK/OTP products (TMP86C829B/CH29B/ CM29B/PM29A) as listed below. See 2.15.2 “Control ” for explanations about the control registers.

		FLASH Product (TMP86FM29)	Emulation Chip MASK/OTP Product
Rewriting the EEPCR register<EEPMD, EEPRS, MNPWDW>		It is possible to rewrite the EEPCR register only when the program execution area in use is RAM/BOOT-ROM.	The FLASH function is not executed because the emulation chip and the MASK/OTP products don't have EEPCR and EEPSR registers. Therefore, the software including the FLASH register can not be emulated by the emulation chip. And then if the software including the FLASH register is executed in the MASK or the emulation chip, the software process differs from the FLASH product.
FLASH write time (The emulation chip is written to emulation memory instead of FLASH)		Typically 4 ms (Independent of the system clock)	
Executing a read instruction/fetch to the 8000H to FFFFH area when EEPSR<BFBUSY> = “1”		If EEPSR<BFBUSY> = “1”, executing a read instruction/fetch to the FLASH area causes FFH to be read regardless of what the current ROM data is. Fetching FFH results in a software interrupt occurring.	
Executing a write instruction to the 8000H to FFFFH area when EEPCR<EEPMD> = “0011”, EEPSR<EWUPEN> = “1” and EEPSR<BFBUSY> = “0”.	MCU mode	The EEPSR<BFBUSY> stays at “0” (Write disabled).	
	Serial PROM mode	The EEPSR<BFBUSY> is set to “1” (write enabled).	
CPU wait for Flash (Wait period for stabilizing of the power supply of Flash control circuit)		The wait period is inserted in the releasing from Reset, STOP mode (EEPCR<MNPWDW>="1") and IDLE/SLEEP mode (EEPCR<ATPWDW>="0"). Even if the FLASH register is not used for software, the wait period is inserted in Reset process.	The wait period is not inserted. Even if the FLASH register is not used for software, the Reset and STOP process differs from the FLASH product.
BOOT-ROM		2 Kbytes are included in the 3800H to 3FFFH area.	The emulation chip and MASK/OTP products don't have the BOOT-ROM.
Operating voltage		VDD = 1.8 to 3.6 V	VDD = 1.8 to 5.5 V (MASK) VDD = 1.8 to 5.25 V(Emulation chip)

2.14.4 FLASH Memory Configuration

64 consecutive bytes in the FLASH area are treated as one group, which is defined as a page. The TMP86FM29 incorporates a one-page temporary data buffer. Writing data to FLASH is temporarily stored in this 64-byte data buffer. After 64 bytes data have been written to the temporary data buffer, these data are written to specified page of FLASH at a time. However, data can be read from any address byte by byte.

2.14.4.1 Page Configuration

The FLASH area has a page configuration of 64 bytes/page as shown below. The total number of bytes in it is 512 pages × 64 bytes (= 32768 bytes). The writeable area is 8000H to FFFFH in Serial PROM mode.

Note: The FLASH memory (8000H to FFFFH) can be written only in the Serial PROM mode. For details of the Serial PROM mode, refer to “2.1 Serial PROM mode”.

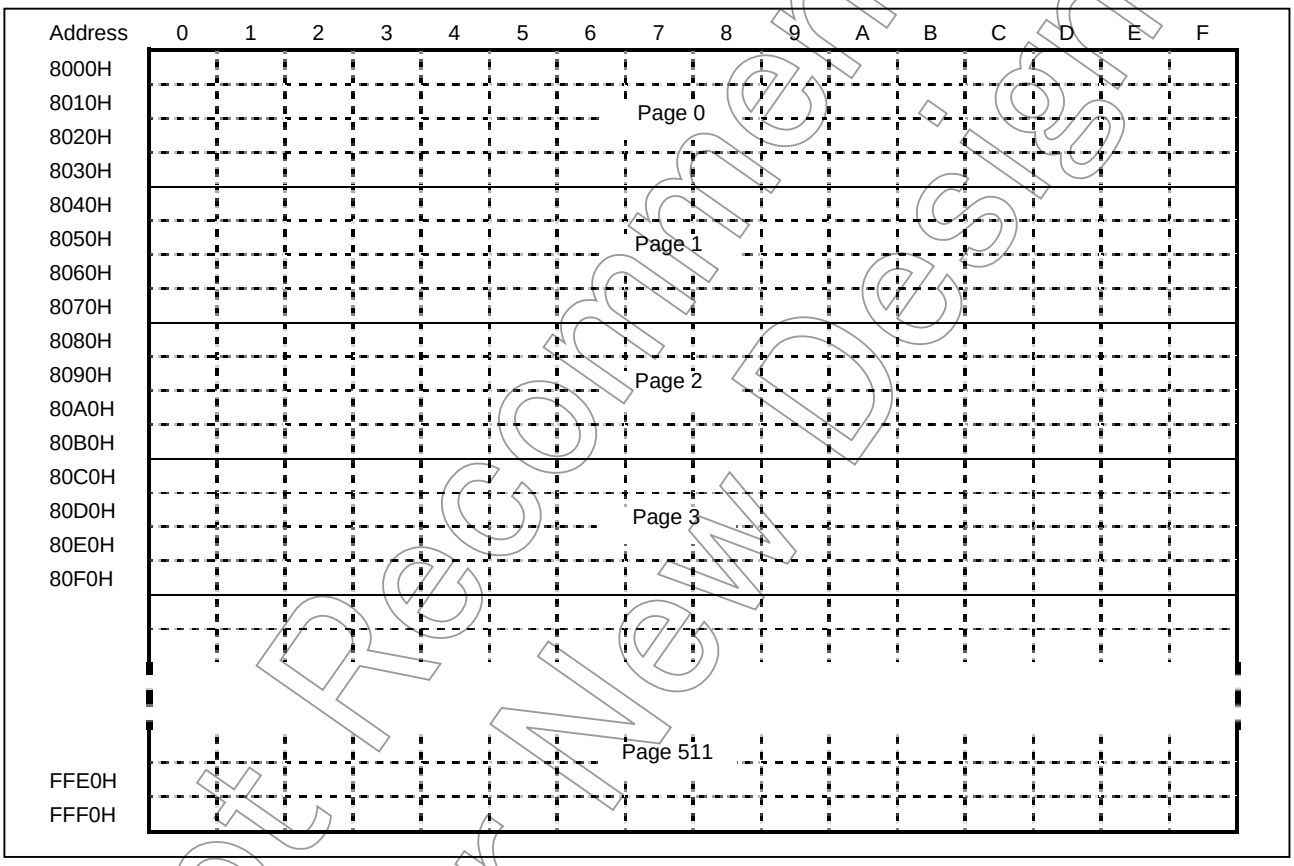


Figure 2.14.1 Page Configuration

2.15 FLASH Memory Control Circuit

2.15.1 Configuration

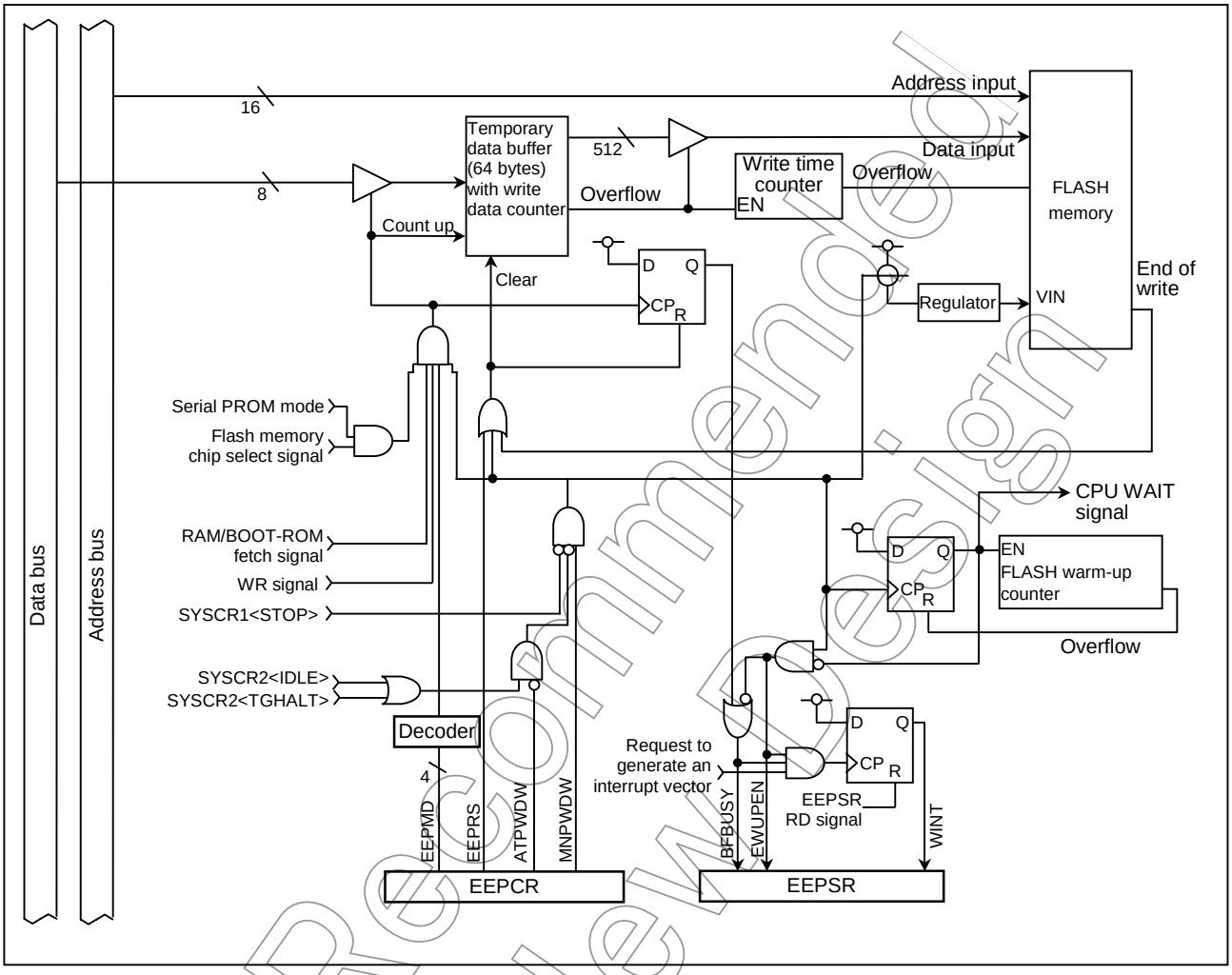


Figure 2.15.1 FLASH Memory Control

2.15.2 Control

The FLASH is controlled by FLASH control register (EEPCR) and FLASH status register (EEPSR).

FLASH Control Register

	7	6	5	4	3	2	1	0	
EEPCR (0FE0H)	EEPMD					EEPRS	ATPWDW	MNPWDW	(Initial value: 1100 *011)

			Program Execution Area	
			RAM/ BOOT	FLASH
EEPMD	FLASH write enable control (Write protect)	1100: FLASH write disable 0011: FLASH write enable Other values: Reserved	R/W	Read only
EEPRS	FLASH write forcible stop	0: – 1: FLASH writing is forced to stop. (The write data counter is initialized.) * After writing “1” to EEPRS, it is automatically cleared to “0”.		
ATPWDW	Automatic power control for the FLASH control circuit in the IDLE0/1/2, SLEEP0/1/2 modes. (This bit is available only when MNPWDW is set to “1”.)	0: Automatic power shut down is executed in IDLE0/1/2 and SLEEP0/1/2 modes. 1: Automatic power shut down is not executed in IDLE0/1/2 and SLEEP0/1/2 modes. (The power is always supplied in these modes.)		R/W
MNPWDW	Software-based power control for the FLASH control circuit	0: The power for the FLASH control circuit is turned off. 1: The power for the FLASH control circuit is turned on		Read only

Note 1: The EEPMD, EEPRS, and MNPWDW can be rewritten only when a program fetch is taking place in the RAM or BOOT-ROM area. If an attempt is made to rewrite the EEPCR register when a program is being executed in the FLASH area, the EEPMD, EEPRS, and MNPWDW keep holding the previous data; they are not rewritten.

Note 2: To write to the FLASH, set the EEPMD with “0011B” in advance when a program fetch is taking place in the RAM area.

Note 3: To forcibly stop writing of FLASH, set the EEPRS to “1” when a program fetch is taking place in the RAM area.

Note 4: The ATPWDW functions only if the MNPWDW is “1”. If the MNPWDW is “0”, the power for the FLASH control circuit is kept turned off regardless of the setting of the ATPWDW.

Note 5: When a STOP mode is executed, the power for the FLASH control circuit is turned off regardless of the setting of the ATPWDW. If the MNPWDW is “0”, entering/exiting the STOP mode allows the power for the FLASH control circuit to be kept turned off.

Note 6: Executing a read instruction to the EEPCR register results in bit3 being read as undefined. Bit2 is always read as “0”.

Note 7: The following attention is necessary when the MNPWDW is set or cleared.

When the MNPWDW is changed from “1” to “0”	Clear the interrupt master enable flag (IMF) to “0” in advance to disable an interrupt. After that, do not set IMF to “1” during EEPSR<EWUPEN> = “0”. If a watchdog timer is used as interrupt request, clear the binary counter for the watchdog timer just before MNPWDW is changed from “1” to “0”.
When the MNPWDW is changed from “0” to “1”	When write to or read from the Flash memory, make sure that the EEPSR<EWUPEN> is “1” by software. Once the MNPWDW is rewritten from “0” to “1” by software, keep performing software-based polling until the EEPSR<EWUPEN> becomes “1”.

Note 8: In MCU mode, the EEPMD and EEPRS should be always set to "1100B" and "0".

Figure 2.15.2 FLASH Control Register

FLASH status register

	7	6	5	4	3	2	1	0	
EEPSR (0FE1H)						WINT	EWUPEN	BFBUSY	(Initial value: **** *010)

WINT	Interrupt detection during a write to the FLASH		0: Not detected 1: Detected (Interrupt occurred) * WINT is automatically cleared to "0" when read instruction is executed to EEPSR.			Read only
EWUPEN	FLASH control circuit status monitor	Control circuit status	Operating (power on)		Halt (power off) or warm-up	
		FLASH status	Temporary data buffer empty	Writing	Disable	
			1	1	0	
BFBUSY	FLASH write busy flag		0	1	1	

Note 1: If a nonmaskable interrupt occurs during a write to the FLASH, the WINT is set to "1" and the writing is discontinued, and then warm-up (CPU wait) for the control circuit of Flash memory is executed. (The write data counter is initialized.) If WINT = "1" is detected in the nonmaskable interrupt service routine, a write is not completed successfully. So, it is necessary to try a write again. The content of the page to which a write is taking place may be changed to an unexpected value depending on the timing when the WINT becomes "1".

Note 2: Even if a nonmaskable interrupt occurs during an FLASH warm-up, the CPU stays at a halt until the warm-up is finished.

Note 3: The WINT is automatically cleared to "0" when a read instruction is executed to the EEPSR register.

Note 4: When MNPWDW is changed from "0" to "1", EWUPEN becomes "1" after taking 2^{10}/fc [s] (if SYSCK = "0") or 2^{3}/fs [s] (if SYSCK = "1"). Before accessing the FLASH, make sure that the EWUPEN is "1" in the RAM area.

Note 5: If the BFBUSY is "1", executing a read instruction or fetch to the FLASH area causes FFH to be read. Fetching FFH results in a software interrupt occurring.

Figure 2.15.3 FLASH Status Register

2.15.3 FLASH Write Enable Control (EEPCR<EEPMD>)

In the FLASH product, the control register can be used to disable a write to the FLASH (Write protect) in order to prevent a write to the FLASH from occurring by mistake because of a program error or microcontroller malfunction. To enable a write to the FLASH, set the EEPCR<EEPMD> with 0011B. To disable a write to the FLASH, set the EEPCR<EEPMD> with 1100B. A reset initializes the EEPCR<EEPMD> to 1100B to disable a write to the FLASH. Usually, set the EEPCR<EEPMD> with 1100B, except when it is necessary to write to the FLASH.

Note1: The FLASH memory (8000H to FFFFH) can be written only in the Serial PROM mode.

Note2: The EEPCR<EEPMD> can be rewritten only when a program is being executed in the RAM area. Executing a write instruction to the EEPCR<EEPMD> in the FLASH area does not change its setting.

Note3: This function can be used in Serial PROM mode. In MCU mode, the EEPCR<EEPMD> should be always set to “1100B”.

2.15.4 FLASH Write Forcible Stop (EEPCR<EEPRS>)

To forcibly stop a write to the FLASH, set the EEPCR<EEPRS> to "1". Setting the EEPCR<EEPRS> to "1" initializes the write data counter of data buffer and forcibly stops a write, and then a warm-up (CPU wait) for the control circuit of Flash memory is executed. After warm-up period, the EEPSR<BFBUSY> is cleared to "0". The warm-up period is 2^{10}/fc (SYSCK = "0") or 2^{3}/fs (SYSCK = "1"). After this, if writing to FLASH starts again, data is stored as the first byte of the temporary data buffer and sets the EEPSR<BFBUSY> to "1". Therefore, it is necessary to write 64 bytes data to the temporary data buffer.

After 1 to 63 bytes are saved to the temporary data buffer, if the EEPCR<EEPRS> is set to "1" the specified page of FLASH is not written. (It keeps previous data.)

Note 1: After 64 bytes are written to the temporary data buffer, the setting the EEPCR<EEPRS> to "1" may cause the writing the page of FLASH to an unexpected value.

Note 2: The EEPCR<EEPRS> can be rewritten only when a program is being executed in the RAM area. In the FLASH area, executing a write instruction to the EEPCR<EEPRS> does not affect its setting.

Note 3: During the warm-up period for Flash memory (CPU wait), the peripheral circuits continue operating, but the CPU stays at a halt until the warm-up is finished. Even if an interrupt latch is set to "1" by generating of interrupt request, an interrupt sequence doesn't start till the end of warm-up. If interrupts occur during a warm-up period with IMF = "1", the interrupt sequence which depends on interrupt priority will start after warm-up period.

Note 4: When the EEPCR<EEPRS> is set to "1" with EEPSR<BFBUSY> = "0", a warm-up is not executed.

Note 5: If executed a write or read instruction to the Flash area immediately after setting EEPCR<EEPRS>, insert one or more machine cycle instructions after setting EEPCR<EEPRS>.

Example: Reads the Flash memory data immediately after setting EEPCR<EEPRS> to "1"

```
LD      HL,8000H
LD      (EEPCR),3FH     ; Set EEPCR<EEPRS> to "1"
NOP                     ; NOP
                          (Do not execute write or read instruction immediately
                          after setting EEPCR<EEPRS>.)
LD      A,(HL)          ; Reads the data of address 8000H
                          (Write or read instruction to the Flash memory)
```

Note 6: This function can be used in Serial PROM mode. In MCU mode, the EEPCR<EEPRS> should be always set to "0".

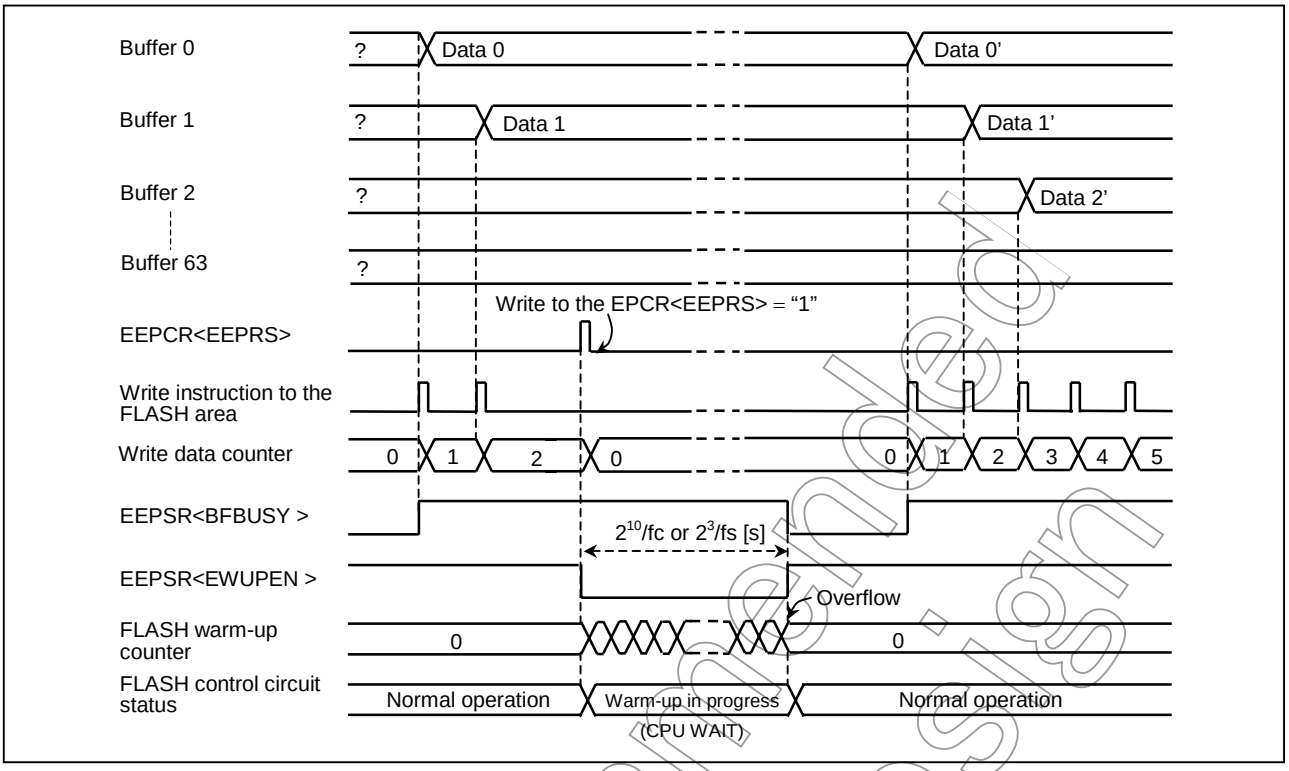


Figure 2.15.4 Write Data Counter Initialization and Write Forcible Stop

2.15.5 Power Control for the FLASH Control Circuit

For the FLASH product, it is possible to turn off the power for FLASH control circuit (such as a regulator) to suppress power consumption if the FLASH area is not accessed.

The EEPCR<MNPWDW> and EEPCR<ATPWDW> are used to control the power for the FLASH control circuit. If the power for the FLASH control circuit is turned off according to the setting of these registers, starting to use the circuits again needs to allow warm-up time for the power supply.

Table 2.15.1 Power Supply Warm-up Time (CPU wait) for the FLASH Control Circuit

NORMAL1/2 IDLE0/1/2 Mode	SLOW1/2 SLEEP0/1/2 Mode	STOP Mode (when EEPCR<MNPWDW> = "1")	
		To Return to a NORMAL Mode	To Return to a SLOW Mode
2^{10}/fc [s] (64 μs @16 MHz)	2^3/fs [s] (244 μs @32.768 kHz)	STOP warm-up time + 2^{10}/fc [s]	STOP warm-up time + 2^3/fs [s]

2.15.5.1 Software-based Power Control for the FLASH Control Circuit (EEPCR<MNPWDW>)

The EEPCR<MNPWDW> is a software-based power control bit for the FLASH control circuit. When a program is being executed in the RAM area, setting this bit enables software-based power control. Clearing the EEPCR<MNPWDW> to "0" immediately turns off the power for the FLASH control circuit. Once the EEPCR<MNPWDW> is switched from "0" to "1", before attempting a read or fetch from the FLASH area, it is necessary to insert a warm up period by software until the power supply is stabilized. In this case, because the CPU wait is not executed, any other instructions except accessing to Flash (write or read) are available. When MNPWDW is changed from "0" to "1", EWUPEN becomes "1" after taking 2^{10}/fc [s] (SYSCK = "0") or 2^3/fs [s] (SYSCK = "1"). Usually software-based polling should be performed until the EEPSR<EWUPEN> becomes "1". An example of setting is given below.

(1) Example of controlling the EEPCR<MNPWDW>

1. Transfer a program for controlling the EEPCR<MNPWDW> to the RAM area.
2. Release an address trap in the RAM area (Setup the WDTCR1 and WDTCR2 registers).
3. Jump to the control program transferred to the RAM area.
4. Clear the interrupt master enable flag (IMF ← "0").
5. Clear the binary counter if the watchdog timer is in use.
6. To turn off the power for the FLASH control circuit, clear the EEPCR<MNPWDW> to "0".
7. Perform CPU processing as required.
8. To access the FLASH area again, set the EEPCR<MNPWDW> to "1".
9. Keep program polling until the EEPSR<EWUPEN> becomes "1".
   (Upon completion of an FLASH warming-up, the EEPSR<EWUPEN> is set to "1". It takes 2^{10}/fc (SYSCK = "0") or 2^3/fs (SYSCK = "1") until EWUPEN becomes "1".)

This procedure enables the FLASH area to be accessed.

If the EEPCR<MNPWDW> is “1”, entering a STOP mode forcibly turns off the power for the FLASH control circuit. When the STOP mode is released, a STOP mode oscillation warm-up is carried out, and then the CPU wait period (Warm-up for stabilizing of FLASH power supply circuit) is automatically performed. If the EEPCR<MNPWDW> is “0”, entering/exiting the STOP mode keeps the power for the FLASH control circuit turned off.

Note 1: If the EEPSR<EWUPEN> is “0”, do not access (Fetch, read, or write) the FLASH area. Executing a read instruction or fetch to the FLASH area causes FFH to be read. Fetching FFH results in a software interrupt occurring.

Note 2: To clear the EEPCR<MNPWDW> to “0”, clear the interrupt master enable flag (IMF) to “0” in advance to disable an interrupt. After that, do not set IMF to “1” during EEPSR<EWUPEN> = “0”.

Note 3: If the EEPCR<MNPWDW> is “0”, generating a nonmaskable interrupt automatically rewrites the MNPWDW to “1” to warm up the FLASH control circuit (CPU wait). That time, the peripheral circuits continue operating, but the CPU stays at a halt until the warm-up is finished.

Note 4: The EEPCR<MNPWDW> can be rewritten only when a program is being executed in the RAM area. In the FLASH area, executing a write instruction to the EEPCR<MNPWDW> does not affect its setting.

Note 5: If a watchdog timer is used as interrupt request, clear the binary counter for the watchdog timer just before MNPWDW is changed from “1” to “0”.

Note 6: During the warm-up period with a software polling of EEPSR<EWUPEN>, if a nonmaskable interrupt occurs, the CPU stays at a halt until the warm-up is finished.

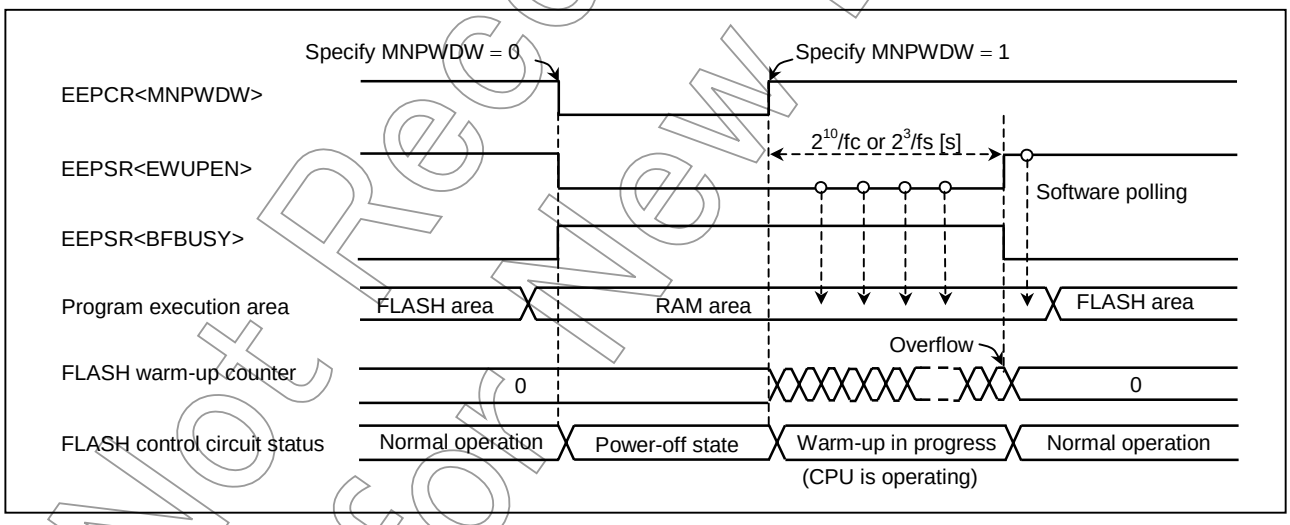


Figure 2.15.5 Software-based Power Control for the FLASH Control Circuit (EEPCR<MNPWDW>)

Example: Performing software-based power control for the FLASH control circuit

```
sRAMAREA:
            DI                          ;  Disable an interrupt (IMF ← "0")
            LD      (WDTCR2),4Eh        ;  Clear the binary counter if the watchdog
                                           timer is in use
            CLR     (EEPCR).0           ;  Clear the EEPCR<MNPWDW> to "0".
             |        |
            SET     (EEPCR).0           ;  Set the EEPCR<MNPWDW> to "1"
sLOOP1:     TEST    (EEPSR).1           ;  Monitor the EEPSR<EWUPEN> register.
            JRS     T,sLOOP1            ;  Jump to sLOOP1 if EEPSR<EWUPEN>
                                           = "0".
            JP      MAIN                ;  Jump to the FLASH area.
```

2.15.5.2 Automatic Power Control for the FLASH Control Circuit (EEPCR<ATPWDW>)

The EEPCR<ATPWDW> is an automatic power control bit for the FLASH control circuit. It is possible to suppress power consumption by automatically shutting down the power for the FLASH control circuit when an operation mode is changed to IDLE0/1/2 and SLEEP0/1/2 modes. This bit can be specified regardless of the area in which a program is being executed.

After the EEPCR<ATPWDW> is cleared to "0", entering an operation mode (IDLE0/1/2 or SLEEP0/1/2) where the CPU is at a halt automatically turns off the power for the FLASH control circuit. Once the operation mode is released, the warm-up time (CPU wait) is automatically counted to resume normal processing. The CPU wait period is either 2^{10}/fc (SYSCK = "0") or 2^3/fs (SYSCK = "1"). If the EEPCR<ATPWDW> is "1", releasing the operation mode does not cause the CPU wait.

If EEPCR<MNPWDW> = "1", executing a STOP mode forcibly turns off the power for the FLASH control circuit regardless of the setting of the EEPCR<ATPWDW>. When the STOP mode is released, a STOP mode oscillation warm-up is carried out, and then an FLASH control circuit warm-up (CPU wait) is automatically performed. If the EEPCR<MNPWDW> is "0", entering/exiting a STOP mode allows the power for the FLASH control circuit to be kept turned off.

Note 1: The EEPCR<ATPWDW> functions only if the EEPCR<MNPWDW> is "1". If the EEPCR<MNPWDW> is "0", the power for the FLASH control circuit is kept turned off when an operation mode is executed or released.

Note 2: During an FLASH warm-up (CPU wait), the peripheral circuits continue operating, but the CPU stays at a halt. Even if an interrupt latch is set under this condition, no interrupt process occurs until the CPU wait is completed. If the IMF is "1" when the interrupt latch is set, interrupt process takes place according to the interrupt priority after the CPU has started operating.

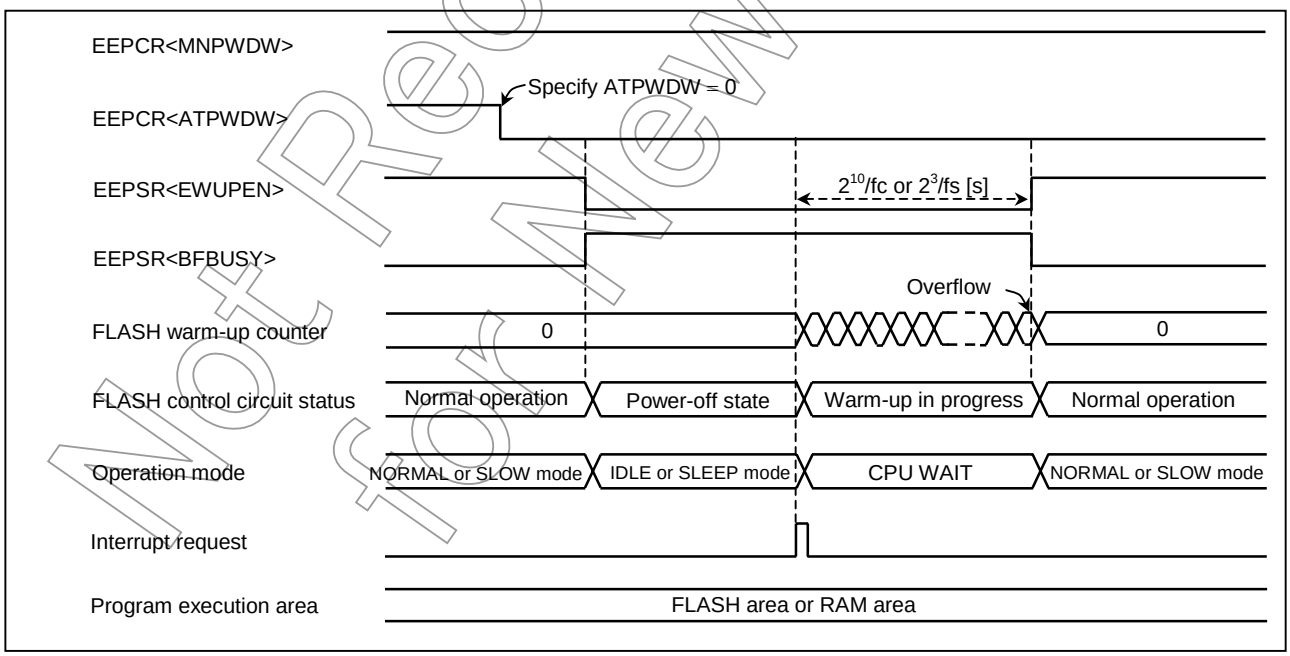


Figure 2.15.6 Automatic Power Control for the FLASH Control Circuit (EEPCR<ATPWDW>)

2.15.6 Accessing the FLASH Memory Area

During the writing to the FLASH memory area, neither a read nor fetch can be performed for the 8000H to FFFFH area. Therefore, to write the FLASH area, the program should be executed in the BOOTROM or RAM area. Basically, to write the FLASH area, the program can be executed in BOOTROM area by using the FLASH writing mode of the Serial PROM mode, but it can be also executed any user program in RAM area by using the RAM loader mode of the Serial PROM mode.

Explanation here is made of only the method of FLASH programming in RAM area. For detail about each operation mode of the Serial PROM mode, refer to “2.1 Serial PROM Mode”.

Although the writing to FLASH is executed on page-by-page, the reading from FLASH is executed on byte-by-byte.

If a nonmaskable interrupt occurs during a write to the FLASH (EEPSR<BFBUSY> = “1”), the WINT is set to “1” and the writing is discontinued, and then the warm-up (CPU wait) for control circuit of Flash memory is executed (The write data counter is also initialized). If WINT = “1” is detected in the nonmaskable interrupt service routine, a write is not completed successfully. So, it is necessary to try a write again. The warm-up period is 2^{10}/fc (SYSCK = “0”) or 2^{3}/fs (SYSCK = “1”). After 1 to 63 bytes are saved to the temporary data buffer, if an interrupt generates, the specified page of FLASH is not written. (It keeps previous data.)

Note 1: Writing to the FLASH area is enabled only in Serial PROM mode. For details of Serial PROM mode, refer to “2.1 Serial PROM mode”.

Note 2: After 64 bytes are written to the temporary data buffer, the generating of an interrupt may cause the writing the page of FLASH to an unexpected value.

Note 3: During the warm-up period for Flash memory (CPU wait), the peripheral circuits continue operating, but the CPU stays at a halt until the warm-up is finished. Even if an interrupt latch is set to “1” by generating of interrupt request, an interrupt sequence doesn’t start till the end of warm-up. If interrupts occur during a warm-up period with IMF = “1”, the interrupt sequence which depends on interrupt priority will start after warm-up period.

Note 4: When write the data to Flash memory from RAM area, disable all the nonmaskable interrupt by clearing interrupt master enable flag (IMF) to “0” beforehand.

2.15.6.1 FLASH Writing Program in the RAM Area

To develop the program in RAM, the write control program should be loaded from external device by using RAM loader mode in Serial PROM mode. Given below is an example of writing the control program in the RAM area.

(1) Example of writing program in the RAM area

1. Monitor the EEPSR<EWUPEN>. If it is “0”, set the EEPCR<MNPWDW> to “1”, and then start and keep polling until the EEPSR<EWUPEN> becomes “1”.
2. Clear the interrupt master enable flag (IMF ← “0”).
3. Set the EEPCR with “3BH” (to enable a write to the FLASH).
4. Execute a write instruction for 64 bytes to the FLASH area.
5. Start and keep polling by software until the EEPSR<BFBUSY> becomes “0”. (Upon completion of an erase and write to the FLASH cells, the EEPSR<BFBUSY> is set to “1”. For the FLASH product, the required write time is typically 4 ms.)
6. Set the EEPCR with “CBH” (to disable a write to the FLASH).

Note: See (2), “Method of specifying an address for a write to the FLASH,” for a description about the FLASH address to be specified at step 4 above.

(2) Method of specifying an address for a write to the FLASH

The FLASH page to be written is specified by the 10 high-order bits of the address of the first-byte data. The first-byte data is stored at the first address of the temporary data buffer. If the data to be written is, for example, 8040H, page 1 is selected, and the data is stored at the first address of the temporary data buffer. Even if the 6 low-order bits of the specified address is not 000000B, the first-byte data is always stored at the first address of the data buffer.

Any address can be specified as the second and subsequent address within FLASH area (8000H to FFFFH). The write data bytes are stored in the temporary data buffer in the sequence they are written, regardless of what address is specified. Usually, the address that is the same as the first-byte is specified for the second and subsequent address. A 16-bit transfer instruction (LDW) can also be used for writing to the temporary data buffer.

Example: Data bytes 00H to 3FH are written to page 1.
(Figure 2.15.7 shows the example of data buffer and pages.)

```
        DI                          ;  Disable an interrupt (IMF ← "0")
        LD      C,00H
        LD      HL,EEPCR            ;  Specify the EEPCR register address.
        LD      IX,8040H            ;  Specify a write address.
        LD      (HL),3BH            ;  Specify the EEPCR
sLOOP1:
        LD      (IX),C              ;  Store data to the temporary data buffer.
                                       (A write page is selected when the first
                                       byte is written.)
        INC     C                   ;  C = C + 1
        CMP     C,40H               ;  Jump to sLOOP1 if C is not 40H
        JR      NZ,sLOOP1
sLOOP2:
        TEST    (EEPSR).0
        JRS     F,sLOOP2            ;  Jump to sLOOP2 if EEPSR<BFBUSY> =
                                       "1".
        LD      (HL),0CBH           ;  Specify the EEPCR
```

Note: If the BFBUSY is "1", executing a read instruction or fetch to the FLASH area causes "FFH" to be read. Fetching "FFH" results in a software interrupt occurring.

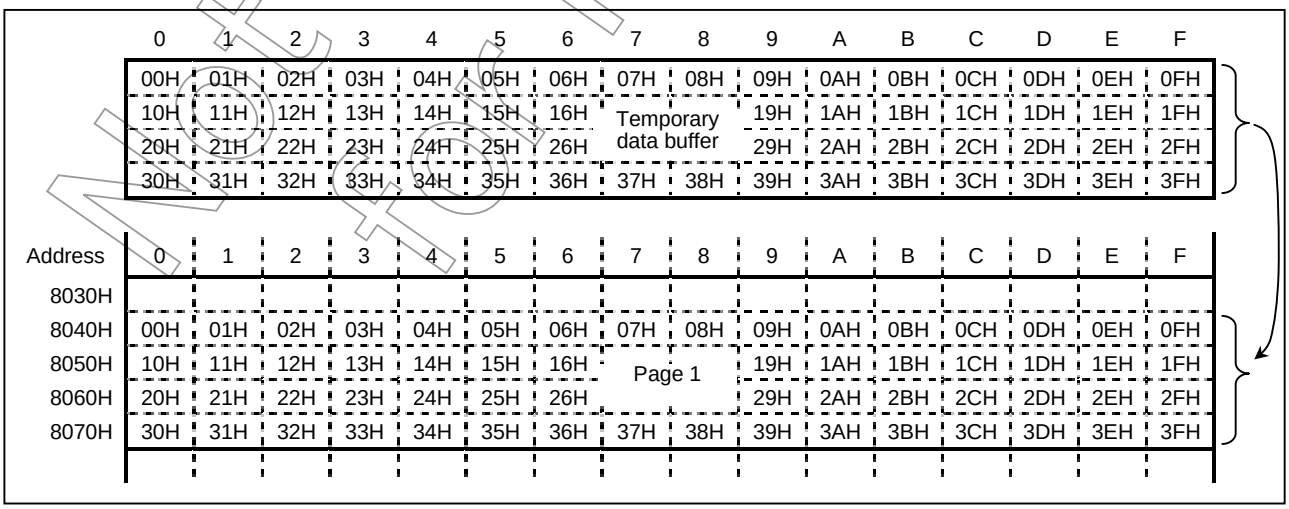


Figure 2.15.8 Data Buffer and Write Page (Example)

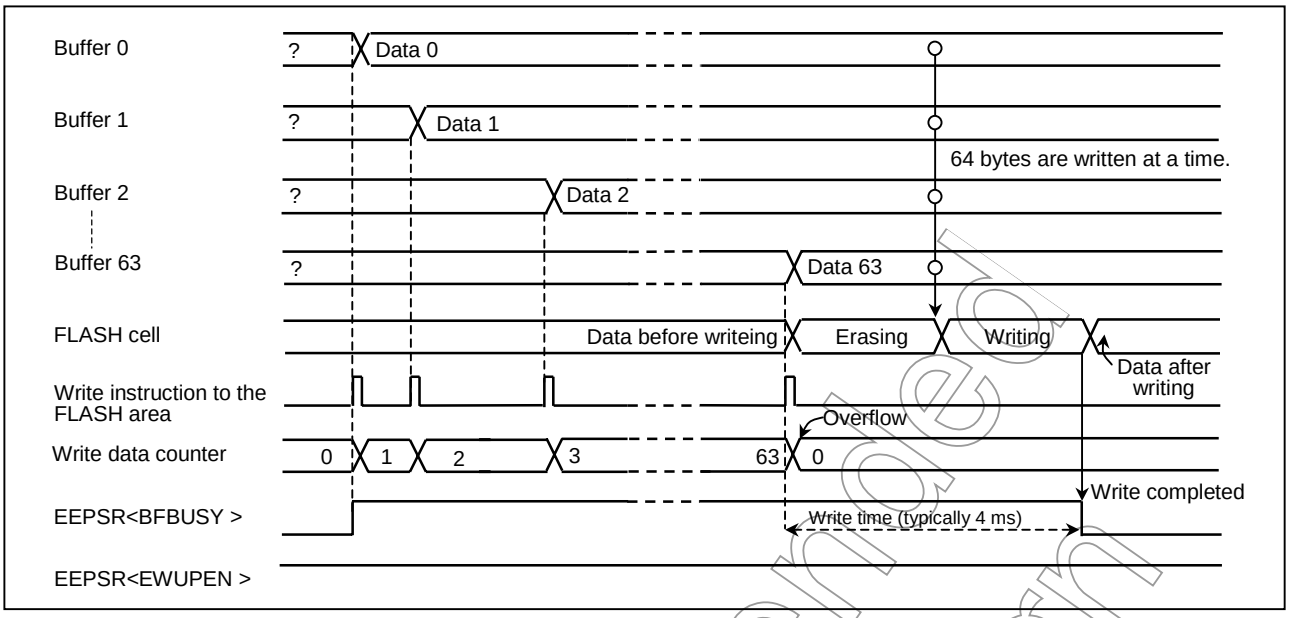


Figure 2.15.9 Write to the FLASH Memory Area

2.16 Serial PROM Mode

2.16.1 Outline

The TMP86FM29 has a 2 Kbytes BOOT-ROM for programming to FLASH memory. This BOOT-ROM is a mask ROM that contains a program to write the FLASH memory on-board. The BOOT-ROM is available in a serial PROM mode and it is controlled by P11 pin, BOOT(P15) pin,TEST pin and $\overline{\text{RESET}}$ pin, and is communicated via TXD (P16) and RXD (P15) pins. There are four operation modes in a serial PROM mode: FLASH memory writing mode, RAM loader mode, FLASH memory SUM output mode and Product discrimination code output mode. Operating area of serial PROM mode differs from that of MCU mode. The operating area of serial PROM mode shows in Table 2.16.1.

Table 2.16.1 Operating Area of Serial PROM Mode

Parameter	Min	Max	Unit
Operating voltage	2.7	3.6	V
High frequency (Note)	2	16	MHz
Temperature	25 ± 5		°C

Note: Even though included in above operating area, part of frequency can not be supported in serial PROM mode. For details, refer to Table 2.16.6.

2.16.2 Memory Mapping

The BOOT-ROM is mapped in address 3800_H to $3FFF_H$. The Figure 2.16.1 shows a memory mapping.

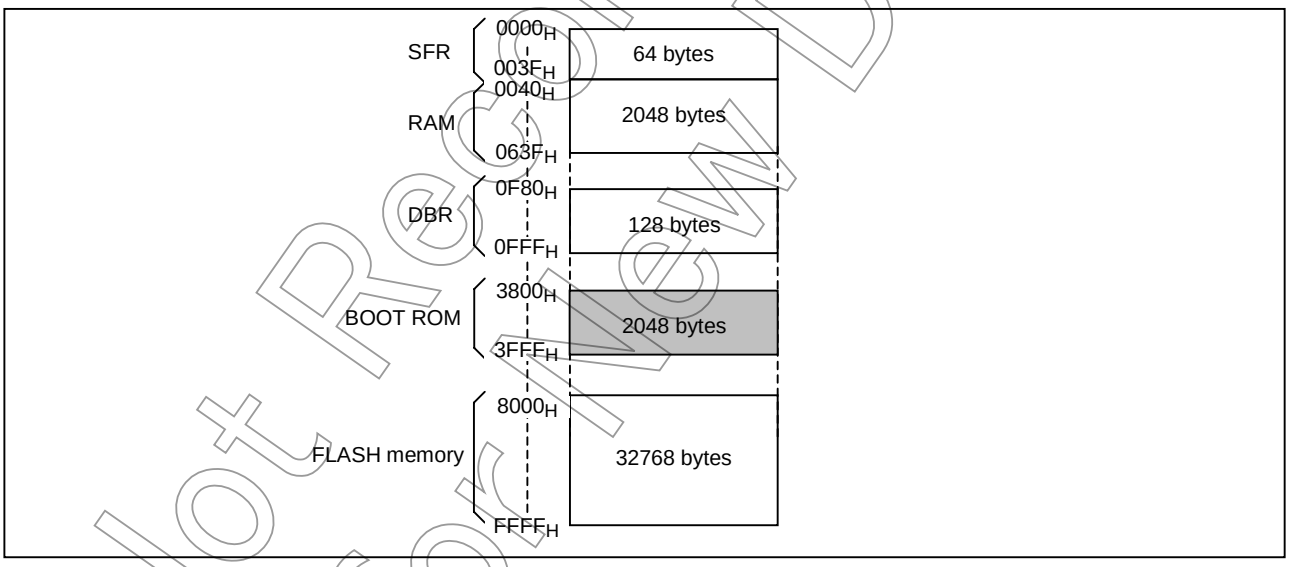


Figure 2.16.1 Memory Address Maps

2.16.3 Serial PROM Mode Setting

2.16.3.1 Serial PROM Mode Control Pins

To execute on-board programming, start the TMP86FM29 in serial PROM mode. Setting of a serial PROM mode is shown in Table 2.16.2.

Table 2.16.2 Serial PROM Mode Setting

Pin	Setting
BOOT/RXD pin (P15)	High
P11 pin	Low
$\overline{\text{RESET}}$, TESTpin	

2.16.3.2 Pin Function

In the serial PROM mode, TXD (P16) and RXD (P15) pins are used as a serial interface pin.

Table 2.16.3 Pin function in the Serial PROM Mode

Pin Name (Serial PROM Mode)	Input/ Output	Function		Pin Name (MCU Mode)
TXD	Output	Serial data output Connect the pull-up resistor.	(Note 1)	P16
RXD	Input	Serial data input		P15
$\overline{\text{RESET}}$	Input	Serial PROM mode control		$\overline{\text{RESET}}$
TEST	Input	Serial PROM mode control		TEST
P11	Input	Serial PROM mode control (Fix to "L" level)		P11
VDD, AVDD	Power Supply	2.7 V to 3.6 V		
VSS		0V		
VAREF		Open or equal with VDD		
P10, P12 to P14, P17 P20 to P22 P30 to P33 P50 to P57 P60 to P67 P70 to P77	I/O	Placed in High-Z state during serial PROM mode.		
SEG7 to SEG0 COM3 to COM0	Output	Placed to "L" level state during serial PROM mode.		
C0,C1,V3 to V1	LCD voltage booster pin	Connect the capacitor or open.		
XIN	Input	Resonator connecting pins for high-frequency clock. For inputting external clock, XIN is used and XOUT is opened.		(Note 2)
XOUT	Output			

Note 1: When the device is used as on-board writing and other parts are already mounted in place, be careful no to affect these communication control pins.

Note 2: Operating area of high frequency in serial PROM mode is from 2 MHz to 16 MHz.

To set a serial PROM mode, connect device pins as shown in Figure 2.16.2.

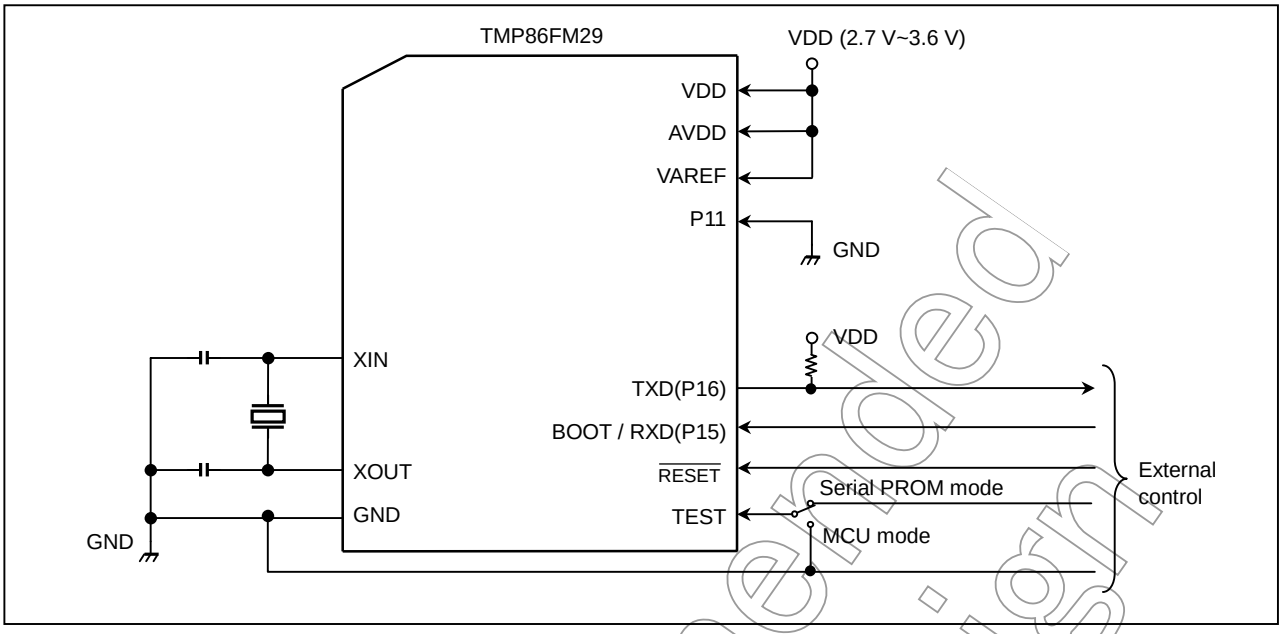


Figure 2.16.2 Serial PROM Mode Port Setting

2.16.3.3 Activating Serial PROM Mode

The following is a procedure of setting of serial PROM mode. Figure 2.16.3 shows a serial PROM mode timing.

(1) Turn on the power to the VDD pin.
(2) Set the P11 pin, TEST pin and $\overline{\text{RESET}}$ pin to low level.
(3) Set the BOOT/RXD pin (P15) to high level.
(4) Wait until the power supply and clock sufficiently stabilize.
(5) Set the TEST pin from low level to high level.
(6) Release the $\overline{\text{RESET}}$. (Set to high level)
(7) Input a matching data (5AH) to RXD pin after waiting for setup sequence.

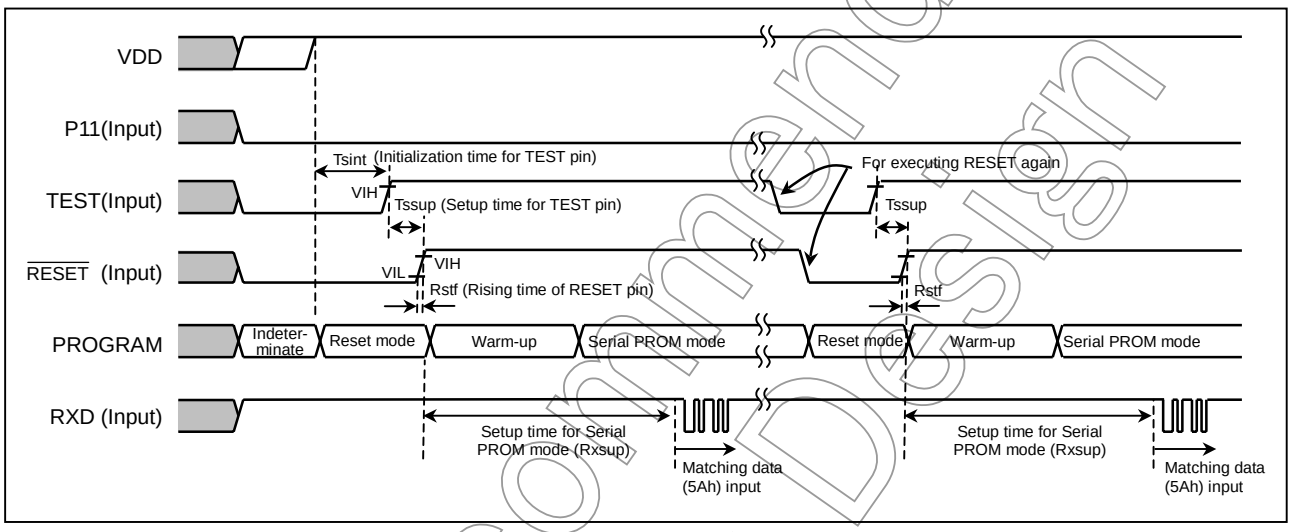


Figure 2.16.3 Serial PROM Mode Timing

Table 2.16.4 Serial PROM Mode Timing characteristics

Parameter		Symbol	The Number of Clock (fc)	Required Minimum Time	
				at fc = 2 MHz	at fc = 16 MHz
Setup time for TEST pin	Rstf > 512 / fc [s]	Tssup	-	1 ms	
	Rstf < 512 / fc [s]		-	0 *Note1	
Initialization time for TEST pin		Tsint	-	1ms	
Time from reset release until acceptance of start bit of RXD pin		RXsup	110000	55 ms	6.9 ms

Note 1: If Rstf is shorter than 512 / fc[s] due to using CMOS-type reset IC or Logic IC, the TEST pin can input the same pulse as the $\overline{\text{RESET}}$ pin input. (TEST pin can be directly connected to the $\overline{\text{RESET}}$ pin.) However, drive the pins carefully not to affect the pin's input level, as the TEST pin and the $\overline{\text{RESET}}$ pin have pull-down resistor and pull-up resistor built-in.

Note 2: fc; High-frequency clock

2.16.3.4 Examples of On-board writing

Figure 2.16.4 shows examples of On-board writing.

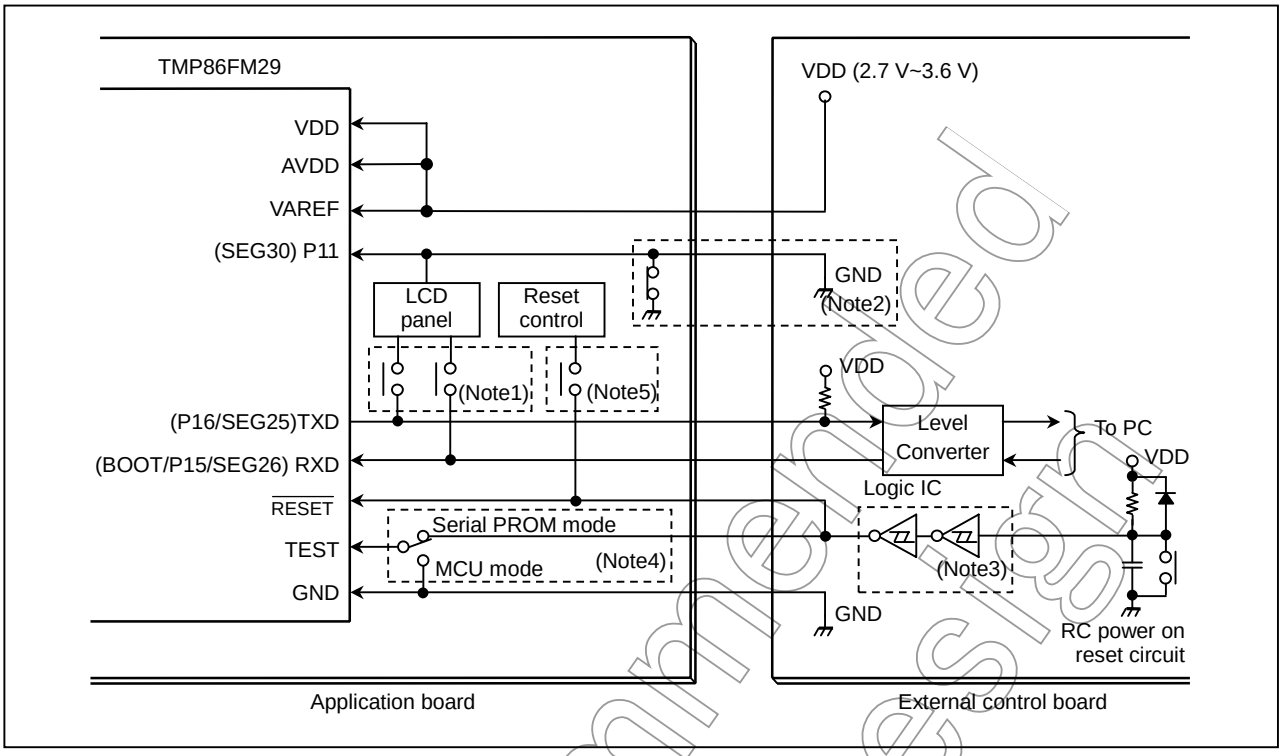


Figure 2.16.4 Examples of Onboard writing

Note 1: If capacity for LCD panel and other devices on the application board affect UART communication in Serial PROM mode, disconnect these pins by using a jumper or a switch.

Note 2: Set the P11 pin to GND. There are two ways. Set P11 pin to GND on the external board, or set it to GND by setting a jumper on the application board.

Note 3: If input signal has analog delay due to the use of such circuit as RC power on reset circuit, connect both TEST pin and $\overline{\text{RESET}}$ pin to logic ICs (Schmitt input IC such as TC74HC14). In this case, control the pin capacity to require the condition Rstf<512/fc[s].

Note4: In MCU mode, the TEST pin can be disconnected as it has a pull-down resistor built-in. However, we recommend connecting it to GND level to avoid noise influence.

Note5: If the RESET control circuit on the application board affects the Serial PROM mode to start, disconnect it by using a jumper, etc.

2.16.4 Interface Specifications for UART

The following shows the UART communication format used in serial PROM mode.

Before on-board programming can be executed, the communication format on the external controller side must also be set up in the same way as for this product.

Note that although the default baud rate is 9600 bps, it can be changed to other values as shown in Table 2.16.5. The Table 2.16.6 shows an operating frequency and baud rate in serial PROM mode. Except frequency which is not described in Table 2.16.6 can not use in serial PROM mode.

Baud rate (Default): 9600 bps
Data length: 8 bits
Parity addition: None
Stop bit length: 1 bit

Table 2.16.5 Baud Rate Modification Data

Baud rate modification data	04H	05H	06H	07H	0AH	18H	28H
Baud rate (bps)	76800	62500	57600	38400	31250	19200	9600

Table 2.16.6 Operating Frequency and Baud Rate in Serial PROM Mode

(Note 3)	Reference Baud Rate (bps)		76800		62500		57600		38400		31250		19200		9600	
	Baud Rate Modification Data		04H		05H		06H		07H		0AH		18H		28H	
	Ref. Frequency (MHz)	Area (MHz)	Baud rate (bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)
1	2	1.91~2.10	–	–	–	–	–	–	–	–	–	–	–	–	9615	+0.16
2	4	3.82~4.19	–	–	–	–	–	–	–	–	31250	0.00	19231	+0.16	9615	+0.16
	4.19	3.82~4.19	–	–	–	–	–	–	–	–	32734	+4.75	20144	+4.92	10072	+4.92
3	4.9152	4.70~5.16	–	–	–	–	–	–	38400	0.00	–	–	19200	0.00	9600	0.00
	5	4.70~5.16	–	–	–	–	–	–	39063	+1.73	–	–	19531	+1.73	9766	+1.73
4	6	5.87~6.45	–	–	–	–	–	–	-	–	–	–	–	–	9375	–2.34
	6.144	5.87~6.45	–	–	–	–	–	–	-	–	–	–	–	–	9600	0.00
5	7.3728	7.05~7.74	–	–		–	57600	0.00	-	–	–	–	19200	0.00	9600	0.00
6	8	7.64~8.39	–	–	62500	0.00	-	–	38462	+0.16	31250	0.00	19231	+0.16	9615	+0.16
7	9.8304	9.40~10.32	76800	0.00	–	–	-	–	38400	0.00	–	–	19200	0.00	9600	0.00
	10	9.40~10.32	78125	+1.73	–	–	-	–	39063	+1.73	–	–	19531	+1.73	9766	+1.73
8	12	11.75~12.90	–	–	–	–	57692	+0.16	-	–	31250	0.00	18750	–2.34	9375	–2.34
	12.288	11.75~12.90	–	–	–	–	59077	+2.56	-	–	32000	+2.40	19200	0.00	9600	0.00
	12.5	11.75~12.90	–	–	60096	–3.85	60096	+4.33	-	–	30048	–3.85	19531	+1.73	9766	+1.73
9	14.7456	14.10~15.48	–	–	–	–	57600	0.00	38400	0.00	–	–	19200	0.00	9600	0.00
10	16	15.27~16.77	76923	+0.16	62500	0.00	-	–	38462	+0.16	31250	0.00	19231	+0.16	9615	+0.16

Note 1: “Ref.Frequency” and “Area” show the high frequency area supported in serial PROM mode. Except the above frequency can not be supported in serial PROM mode even though the high frequency is included in area from 2 MHz to 16 MHz.

Note 2: The total error of frequency must be kept within +/–3% so that the auto-detection of frequency is executed correctly.

Note 3: An external controller should transmit a matching data repeatedly till the TMP86FM29 transmit an echo back data. Above number indicates a transmission number of times of matching data till transmission of echo back data.

2.16.5 Command

There are five commands in serial PROM mode. After reset release, the TMP86FM29 waits a matching data (5AH).

Table 2.16.7 Command in Serial PROM Mode

Command Data	Operation Mode	Remarks
5AH	Setup	Matching data. Always start with this command after reset release.
30H	FLASH memory writing	Writing to area from 8000H to FFFFH is enable.
60H	RAM loader	Writing to area from 0050H to 062FH is enable.
90H	FLASH memory SUM output	The checksum of entire FLASH area (from 8000H to FFFFH) is output in order of the upper byte and the lower byte.
C0H	Product discrimination code output	Product discrimination code, that is expressed by 13 bytes data, is output.

2.16.6 Operation Mode

There are four operating modes in serial PROM mode: FLASH memory writing mode, RAM loader mode, FLASH memory SUM output mode and Product discrimination code output mode. For details about these modes, refer to "(1) FLASH memory writing mode" through "(4) Product discrimination code output mode".

(1) FLASH memory writing mode

The data are written to the specified FLASH memory addresses. The controller should send the write data in the Intel Hex format (Binary). For details of writing data format, refer to "2.16.7 FLASH memory Writing Data Format".

If no errors are encountered till the end record, the SUM of 32 Kbytes of FLASH memory is calculated and the result is returned to the controller.

To execute the FLASH memory writing mode, the TMP86FM29 checks the passwords except a blank product. If the passwords did not match, the program is not executed.

(2) RAM loader mode

The RAM loader transfers the data into the internal RAM that has been sent from the controller in Intel Hex format. 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. After sending of SUM, the program jumps to the start address of RAM in which the first transferred data has been written. This RAM loader function provides the user's own way to control on-board programming.

To execute the RAM loader mode, the TMP86FM29 checks the passwords except a blank product. If the passwords did not match, the program is not executed.

(3) FLASH memory SUM output mode

The SUM of 32 Kbytes of FLASH memory is calculated and the result is returned to the controller.

The BOOT ROM does not support the reading function of the FLASH memory. Instead, it has this SUM command to use. By reading the SUM, it is possible to manage Revisions of application programs.

(4) Product discrimination code output mode

The product discrimination code is output as a 13-byte data, that includes the start address and the end address of ROM (In case of TMP86FM29, the start address is 8000H and the end address is FFFFH). Therefore, the controller can recognize the device information by using this function.

2.16.6.1 FLASH Writing Mode (Operation command: 30H)

Table 2.16.8 shows FLASH memory writing mode process.

Table 2.16.8 FLASH Writing Mode Process

	Number of Bytes Transferred	Transfer Data from External Controller to TMP86FM29	Baud Rate	Transfer Data from TMP86FM29 to External Controller
BOOT ROM	1st byte 2nd byte	Matching data (5Ah) –	9600 bps 9600 bps	– (Baud rate auto set) OK: Echo back data (5AH) Error: Nothing transmitted
	3rd byte 4th byte	Baud rate modification data (See Table 2.16.5) –	9600 bps 9600 bps	– OK: Echo back data Error: A1H × 3, A3H × 3 , 62H × 3 (Note 1)
	5th byte 6th byte	Operation command data (30H) –	Changed new baud rate Changed new baud rate	– OK: Echo back data (30H) Error: A1H × 3, A3H × 3 , 63H × 3 (Note 1)
	7th byte 8th byte	Address 15 to 08 in which to store Password count (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	9th byte 10th byte	Address 07 to 00 in which to store Password count (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	11th byte 12th byte	Address 15 to 08 in which to start Password comparison (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	13th byte 14th byte	Address 07 to 00 in which to start Password comparison (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	15th byte : m'th byte	Password string (Note 5) –	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	m'th + 1 byte : n'th – 2 byte	Intel Hex format (binary) (Note 2)	Changed new baud rate	–
	n'th – 1 byte	–	Changed new baud rate	OK: SUM (High) (Note 3) Error: Nothing transmitted
	n'th byte	–	Changed new baud rate	OK: SUM (Low) (Note 3) Error: Nothing transmitted
	n'th + 1 byte	(Wait for the next operation) (command data)	Changed new baud rate	–

Note 1: "xxH × 3" denotes that operation stops after sending 3 bytes of xxH. For details, refer to "2.16.8 Error Code".

Note 2: Refer to "2.16.10 Intel Hex Format (Binary)".

Note 3: Refer to "2.16.9 Checksum (SUM)".

Note 4: Refer to "2.16.11 Passwords".

Note 5: If all data of addresses from FFE0H to FFFFH are "00H" or "FFH", the passwords comparison is not executed because the device is considered as blank product. However, it is necessary to specify the password count storage addresses and the password comparison start address even though it is a blank product. If a password error occurs, the UART function of TMP86FM29 stops without returning error code to the controller. Therefore, when a password error occurs, the TMP86FM29 should be reset by $\overline{\text{RESET}}$ pin input.

Description of FLASH memory writing mode

1. The receive data in the 1st byte is the matching data. When the boot program starts in serial PROM mode, TMP86FM29 (Mentioned as “device” hereafter) waits for the matching data (5AH) to receive. Upon receiving the matching data, it automatically adjusts the UART's initial baud rate to 9,600bps.
2. When the device has received the matching data, the device transmits the data “5AH” as an echo back to the controller. If the device can not receive the matching data, the device does not transmit the echo back data and waits for the matching data again with changing baud rate. Therefore, the controller should send the matching data continuously until the device transmits the echo back data. An external controller should transmit a matching data repeatedly till the device transmit an echo back data. The transmission number of times of matching data varies by the frequency of device. For details, refer to Table 2.16.6.
3. The receive data in the 3rd byte is the baud rate modification data. The seven kinds of baud rate modification data shown in Table 2.16.5 are available. Even if baud rate changing is no need, be sure to send the initial baud rate data (28H: 9,600 bps).
4. When the 3rd byte data is one of the baud rate modification data corresponding to the device's operating frequency, the device sends the echo back data which is the same as received baud rate modification data. Then the baud rate is changed. If the 3rd byte data does not correspond to the baud rate modification data, the device stops UART function after sending 3 bytes of baud rate modification error code: (62H). The changing of baud rate is executed after transmitting the echo back data.
5. The receive data in the 5th byte is the command data (30H) to write the FLASH memory.
6. When the 5th byte is one of the operation command data shown in Table 2.16.7, the device sends the echo back data which is the same as received operation command data (in this case, 30H). If the 5th byte data does not correspond to the operation command data, the device stops UART function after sending 3 bytes of operation command error code: (63H).
7. The 7th byte is used as an upper bit (Bit15 to bit8) of the password count storage address. When the receiving is executed correctly (No error), the device does not send any data. If the receiving error or password error occur, the device does not send any data and stops UART function.
8. The 9th byte is used as a lower bit (Bit7 to bit0) of the password count storage address. When the receiving is executed correctly (No error), the device does not send any data. If the receiving error or password error occur, the device does not send any data and stops UART function.
9. The 11th byte is used as an upper bit (Bit15 to bit8) of the password comparison start address. When the receiving is executed correctly (No error), the device does not send any data. If the receiving error or password error occur, the device does not send any data and stops UART function.
10. The 13th byte is used as a lower bit (Bit7 to bit0) of the password comparison start address. When the receiving is executed correctly (No error), the device does not send any data. If the receiving error or password error occur, the device does not send any data and stops UART function.

11. The 15th through the m'th bytes are the password data. The number of passwords is the data (N) indicated by the password count storage address. The password data are compared for N entries beginning with the password comparison start address. The controller should send N bytes of password data to the device. If the passwords do not match, the device stops UART function without returning error code to the controller. If the data of addresses from FFE0H to FFFFH are all "FFH", the comparison of passwords is not executed because the device is considered as a blank product.
12. The receive data in the m'th + 1 through n'th – 2 byte are received as binary data in Intel Hex format. No received data are echoed back to the controller. The data which is not the start mark (3AH for ":") in Intel Hex format is ignored and does not send an error code to the controller until the device receives the start mark. After receiving the start mark, the device receives the data record, that consists of length of data, address, record type, writing data and checksum. After receiving the checksum of data record, the device waits the start mark data (3AH) again. The data of data record is temporarily stored to RAM and then, is written to specified FLASH memory by page (64 bytes) writing. For details of an organization of FLASH, refer to "2.16.7 Serial PROM Mode". Since after receiving an end record, the device starts to calculate the SUM, the controller should wait the SUM after sending the end record. If receive error or Intel Hex format error occurs, the device stops UART function without returning error code to the controller.
13. The n'th – 1 and the n'th bytes are the SUM value that is sent to the controller in order of the upper byte and the lower byte. For details on how to calculate the SUM, refer to "2.16.9 Checksum (SUM)". The SUM calculation is performed after detecting the end record, but the calculation is not executed when receive error or Intel Hex format error has occurred. The time required to calculate the SUM of the 32 Kbytes of FLASH memory area is approximately 100 ms at fc = 16 MHz. After the SUM calculation, the device sends the SUM data to the controller. After sending the end record, the controller can judge that the transmission has been terminated correctly by receiving the checksum.
14. After sending the SUM, the device waits for the next operation command data.

2.16.6.2 RAM Loader Mode (Operation Command: 60H)

Table 2.16.9 shows RAM loader mode process.

Table 2.16.9 RAM Loader Mode Process

	Number of Bytes Transferred	Transfer Data from External Controller to TMP86FM29	Baud Rate	Transfer Data from TMP86FM29 to External Controller
BOOT ROM	1st byte 2nd byte	Matching data (5AH) –	9600 bps 9600 bps	– (Baud rate auto set) OK: Echo back data (5AH) Error: Nothing transmitted
	3rd byte 4th byte	Baud rate modification data (See Table 2.16.5) –	9600 bps 9600 bps	– OK: Echo back data Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)
	5th byte 6th byte	Operation command data (60H) –	Changed new baud rate Changed new baud rate	– OK: Echo back data (60H) Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)
	7th byte 8th byte	Address 15 to 08 in which to store Password count (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	9th byte 10th byte	Address 07 to 00 in which to store Password count (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	11th byte 12th byte	Address 15 to 08 in which to start Password comparison (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	13th byte 14th byte	Address 07 to 00 in which to start Password comparison (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	15th byte : m'th byte	Password string (Note 5) –	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: Nothing transmitted
	m'th + 1 byte : n'th – 2 byte	Intel Hex format (Binary) (Note 2)	Changed new baud rate	–
	n'th – 1 byte	–	Changed new baud rate	OK: SUM (High) (Note 3) Error: Nothing transmitted
	n'th byte	–	Changed new baud rate	OK: SUM (Low) (Note 3) Error: Nothing transmitted
RAM	–	The program jumps to the start address of RAM in which the first transferred data has been written.		

Note 1: "xxH × 3" denotes that operation stops after sending 3 bytes of xxH. For details, refer to 2.16.8 "Error Code".

Note 2: Refer to 2.16.10 "Intel Hex Format (Binary)".

Note 3: Refer to 2.16.9 "Checksum (SUM)".

Note 4: Refer to 2.16.11 "Passwords".

Note 5: If all data of addresses from FFE0H to FFFFH are "00H" or "FFH", the passwords comparison is not executed because the device is considered as blank product. However, it is necessary to specify the password count storage addresses and the password comparison start address even though it is a blank product. If a password error occurs, the UART function of TMP86FM29 stops without returning error code to the controller. Therefore, when a password error

occurs, the TMP86FM29 should be reset by $\overline{\text{RESET}}$ pin input.

Note 6: Do not send only end record after transferring of password string. If the TMP86FM29 receives the end record only after reception of password string, it does not operate correctly.

Note 7: When the FLASH power supply is turned off in user's program by setting EEPCR<MNPWDW>, be sure to disable the watchdog timer (WDT) or to clear the binary counter of WDT immediately before.

Description of RAM loader mode

1. The process of the 1st byte through the 4th byte are the same as FLASH memory writing mode.
2. The receive data in the 5th byte is the RAM loader command data (60H) to write the user's program to RAM.
3. When the 5th byte is one of the operation command data shown in Table 2.16.7, the device sends the echo back data which is the same as received operation command data (in this case, 60H). If the 5th byte data does not correspond to the operation command data, the device stops UART function after sending 3 bytes of operation command error code: (63H).
4. The process of the 7th byte through the m'th byte are the same as FLASH memory writing mode.
5. The receive data in the m'th + 1 through n'th – 2byte are received as binary data in Intel Hex format. No received data are echoed back to the controller.
   The data which is not the start mark (3AH for ":") in Intel Hex format is ignored and does not send an error code to the controller until the device receives the start mark. After receiving the start mark, the device receives the data record, that consists of length of data, address, record type, writing data and checksum. After receiving the checksum of data record, the device waits the start mark data (3AH) again. The data of data record is written to specified RAM by the receiving data. Since after receiving an end record, the device starts to calculate the SUM, the controller should wait the SUM after sending the end record. If receive error or Intel Hex format error occurs, the UART function of TMP86FM29 stops without returning error code to the controller.
6. The n'th – 1 and the n'th bytes are the SUM value that is sent to the controller in order of the upper byte and the lower byte. For details on how to calculate the SUM, refer to 2.16.9 "Checksum (SUM)". The SUM calculation is performed after detecting the end record, but the calculation is not executed when receive error or Intel Hex format error has occurred.
   The SUM is calculated by the data written to RAM, but the length of data, address, record type and checksum in Intel Hex format are not included in SUM.
7. The boot program jumps to the first address that is received as data in Intel Hex format after sending the SUM to the controller.

2.16.6.3 FLASH Memory SUM Output Mode (Operation Command: 90H)

Table 2.16.10 shows FLASH memory SUM output mode process.

Table 2.16.10 FLASH Memory SUM Output Process

	Number of Bytes Transferred	Transfer Data from External Controller to TMP86FM29	Baud Rate	Transfer Data from TMP86FM29 to External Controller
BOOT ROM	1st byte 2nd byte	Matching data (5AH) –	9600 bps 9600 bps	– (Baud rate auto set) OK: Echo back data (5AH) Error: Nothing transmitted
	3rd byte 4th byte	Baud rate modification data (See Table 2.16.5) –	9600 bps 9600 bps	– OK: Echo back data Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)
	5th byte 6th byte	Operation command data (90H) –	Changed new baud rate Changed new baud rate	– OK: Echo back data (90H) Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)
	7th byte	–	Changed new baud rate	OK: SUM (High) (Note 2) Error: Nothing transmitted
	8th byte	–	Changed new baud rate	OK: SUM (Low) (Note 2) Error: Nothing transmitted
	9th byte	(Wait for the next operation) (Command data)	Changed new baud rate	–

Note 1: "xxH × 3" denotes that operation stops after sending 3 bytes of xxH. For details, refer to "2.16.8 Error Code".

Note 2: Refer to "2.16.9 Checksum (SUM)"

Description of FLASH memory SUM output mode

1. The process of the 1st byte through the 4th byte are the same as FLASH memory writing mode.
2. The receive data in the 5th byte is the FLASH memory SUM command data (90H) to calculate the entire FLASH memory.
3. When the 5th byte is one of the operation command data shown in Table 2.16.7, the device sends the echo back data which is the same as received operation command data (in this case, 90H). If the 5th byte data does not correspond to the operation command data, the device stops UART function after sending 3 bytes of operation command error code: (63H).
4. The 7th and the 8th bytes are the SUM value that is sent to the controller in order of the upper byte and the lower byte. For details on how to calculate the SUM, refer to "2.16.9 Checksum (SUM)".
5. After sending the SUM, the device waits for the next operation command data.

2.16.6.4 Product Discrimination Code Output Mode (Operation Command: C0H)

Table 2.16.11 shows product discrimination code output mode process.

Table 2.16.11 Product Discrimination Code Output Process

	Number of Bytes Transferred	Transfer Data from External Controller to TMP86FM29	Baud Rate	Transfer Data from TMP86FM29 to External Controller
BOOT ROM	1st byte 2nd byte	Matching data (5AH) –	9600 bps 9600 bps	– (Baud rate auto set) OK: Echo back data (5AH) Error: Nothing transmitted
	3rd byte 4th byte	Baud rate modification data (See Table 2.16.5) –	9600 bps 9600 bps	– OK: Echo back data Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)
	5th byte 6th byte	Operation command data (C0H) –	Changed new baud rate Changed new baud rate	– OK: Echo back data (C0H) Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)
	7th byte		Changed new baud rate	3AH Start mark
	8th byte		Changed new baud rate	0AH The number of transfer data (from 9th to 18th byte)
	9th byte		Changed new baud rate	02H Length of address (2 bytes)
	10th byte		Changed new baud rate	00H Reserved data
	11th byte		Changed new baud rate	00H Reserved data
	12th byte		Changed new baud rate	00H Reserved data
	13th byte		Changed new baud rate	00H Reserved data
	14th byte		Changed new baud rate	01H The number of ROM block (1 block)
	15th byte		Changed new baud rate	80H First address of ROM
	16th byte		Changed new baud rate	00H
	17th byte		Changed new baud rate	FFH End address of ROM
	18th byte		Changed new baud rate	FFH
	19th byte		Changed new baud rate	7FH Checksum of transferred data (from 9th to 18th byte)
	20th byte	(Wait for the next operation) (Command data)	Changed new baud rate	–

Note: "xxH × 3" denotes that operation stops after sending 3 bytes of xxH. For details, refer to "2.16.8 Error Code".

Description of product discrimination code output mode

1. The process of the 1st byte through the 4th byte are the same as FLASH memory writing mode.
2. The receive data in the 5th byte is the product discrimination code output command data (C0H).
3. When the 5th byte is one of the operation command data shown in Table 2.16.7, the device sends the echo back data which is the same as received operation command data (in this case, C0H). If the 5th byte data does not correspond to the operation command data, the device stops UART function after sending 3 bytes of operation command error code: (63H).
4. The 7th and the 19th bytes are the product discrimination code. For details, refer to 2.16.12 "Product Discrimination Code".
5. After sending the SUM, the device waits for the next operation command data.

2.16.7 FLASH memory Writing Data Format

FLASH area of TMP86FM29 consists of 512 pages and one page size is 64 bytes.

Writing to FLASH is executed by page writing. Therefore, it is necessary to send 64 bytes data (for one page) even though only a few bytes data are written. Figure 2.16.5 shows an organization of FLASH area. When the controller sends the writing data to the device, be sure to keep the format described below.

1. The address of data after receiving the FLASH writing command should be the first address of page. For example, in case of page 2, the first address should be 8080H.
2. If the last data's address of data record is not end address of page, the address of the next data record should be the address + 1. For example, if the last data's address is 802FH (Page 0), the address of the next data record should be 8030H (Page 0).

   Ex)

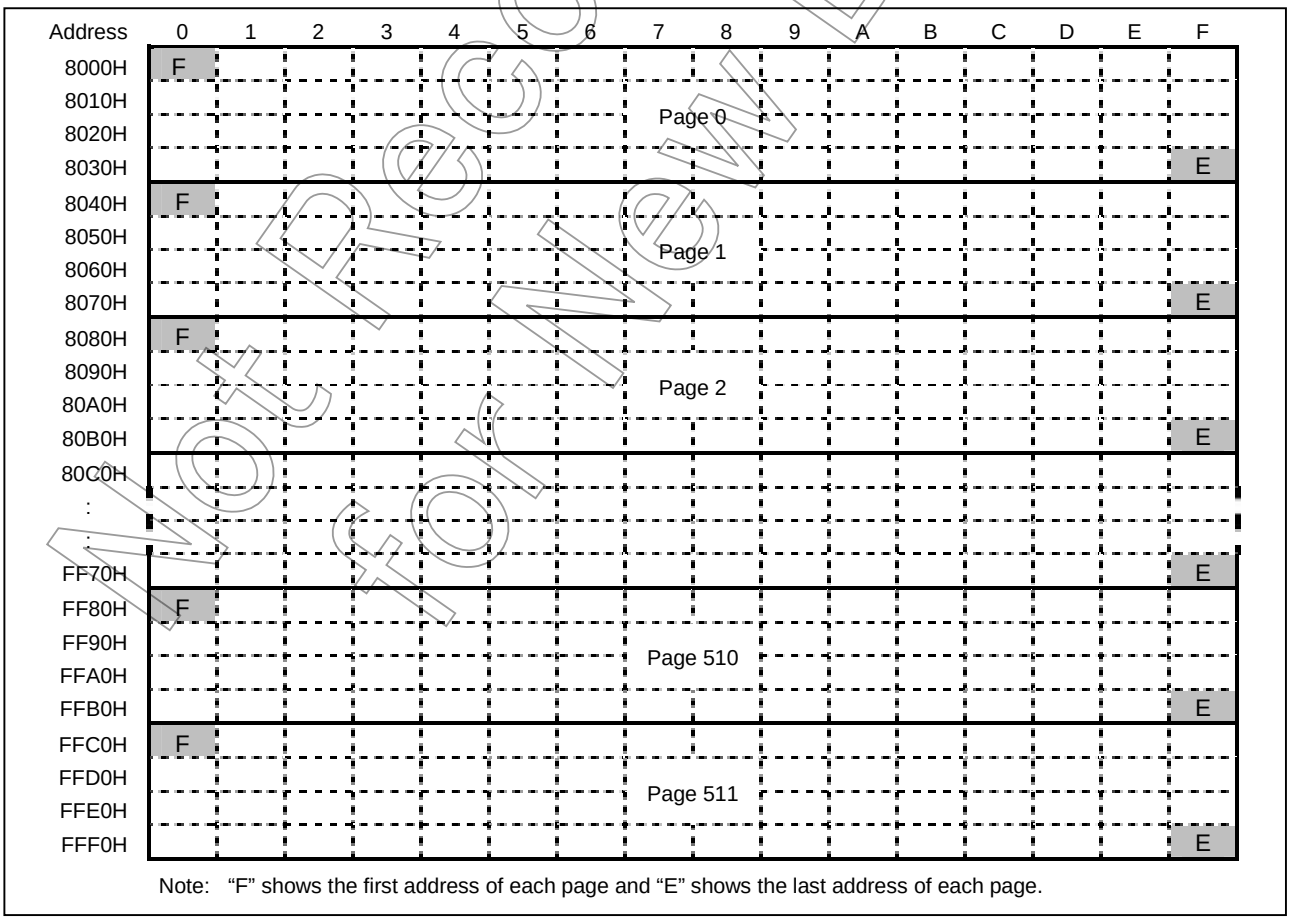
   ```
   :10802000202122232425262728292A2B2C2D2E2FD8  ' 8020H to 802FH data
   :10803000303132333435363738393A3B3C3D3E3FC8  ' 8030H to 803FH data
   ```

3. The last data's address of data record immediately before sending the end record should be the last address of page. For example, in case of page 1, the last data's address of data record should be 807FH.

   Ex)

   ```
   :10807000303132333435363738393A3B3C3D3E3F88  ' 8070H to 807FH data
   :00000001FF                                  ' End record
   ```

   Note: Do not write only the addresses from FFE0H to FFFFH when all data of FLASH memory are the same data. If these area are only written, the next operation can not be executed because of password error.



Note: "F" shows the first address of each page and "E" shows the last address of each page.

Figure 2.16.5 Organization of FLASH Area

2.16.8 Error Code

When the device detects an error, the error codes are sent to the controller.

Table 2.16.12 Error Code

Transmit Data	Meaning of Transmit Data
62H, 62H, 62H	Baud rate modification error occurred.
63H, 63H, 63H	Operating command error occurred.
A1H, A1H, A1H	Framing error in received data occurred.
A3H, A3H, A3H	Overrun error in received data occurred.

Note1: If password error occurs, the TMP86FM29 doesn't send error codes.

2.16.9 Checksum (SUM)

(1) Calculation method

SUM consists of byte + byte... + byte, the checksum of which is returned in word as the result.

Namely, data is read out in byte and checksum 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

The SUM returned when executing the FLASH memory write command, RAM loader command, or FLASH memory SUM command is calculated in the manner shown above.

(2) Calculation data

The data from which SUM is calculated are listed in Table 2.16.13 below.

Table 2.16.13 Checksum Calculation Data

Operating Mode	Calculation Data	Remarks
FLASH memory writing mode	Data in the entire area (32 Kbytes) of FLASH memory	Even when written to part of the FLASH area, data in the entire memory area (32 Kbytes) is calculated. The length of data, address, record type and checksum in Intel Hex format are not included in SUM.
FLASH memory SUM output mode		
RAM loader mode	Data written to RAM	The length of data, address, record type and checksum in Intel Hex format are not included in SUM.
Product Discrimination Code Output mode	Checksum of transferred data (from 9th to 18th byte)	For details, refer to 2.16.12 Product Discrimination Code.

2.16.10 Intel Hex Format (Binary)

1. After receiving the checksum of a record, the device waits for the start mark data (3AH for ":") of the next record. Therefore, the device ignores the data, which does not match the start mark data after receiving the checksum of a record.
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 bytes of data to be received (Upper and lower bytes of checksum). This is because after receiving the checksum of the end record, the boot program calculates the checksum and returns the calculated checksum in two bytes to the controller.
3. If a receive error or Intel Hex format error occurs, the UART function of TMP86FM29 stops without returning error code to the controller. In the following cases, an Intel Hex format error occurs:
   - When the record type is not 00H, 01H, or 02H
   - When a SUM error occurred
   - When the data length of an extended record (Type = 02H) is not 02H
   - When the address of an extended record (Type = 02H) is larger than 1000H and after that, receives the data record
   - When the data length of the end record (Type = 01H) is not 00H

2.16.11 Passwords

The eight or more bytes consecutive data in flash memory area can be used as password. In password check, TMP86FM29 compares these data with data which are transmitted from the external controller. The area in which passwords can be specified is located at addresses 8000H to FF9FH. The area from FFA0H to FFFFH can not be specified as passwords area. The device compares the stored passwords with the passwords, which are received from the controller. If all data of addresses from FFE0H to FFFFH are "00H" or "FFH", the passwords comparison is not executed because the device is considered as blank product. It is necessary to specify the password count storage addresses and the password comparison start address even though it is a blank product. Table 2.16.14 shows the password setting in the blank product and non blank product.

Table 2.16.14 Password Setting in the Blank Product and Non Blank Product

Password	Blank Product (Note 1)	Non Blank Product
PNSA (Password count storage addresses)	8000H ≤ PNSA ≤ FF9FH	8000H ≤ PNSA ≤ FF9FH
PCSA (Password comparison start address)	8000H ≤ PCSA ≤ FF9FH	8000H ≤ PCSA ≤ FFA0 − N
N (Password count)	*	8 ≤ N
Setting of password	No need	Need (Note 2)

Note 1: When all data of addresses from FFE0H to FFFFH area are “00H” or “FFH”, the device is judged as blank product.

Note 2: The same three or more bytes consecutive data can not be used as password. When the password includes the same consecutive data (three or more bytes), the password error occurs. If the password error occurred, the UART function of device stops without returning error code.

Note 3: *: Don’t care.

Note 4: When the password doesn't match the above condition, the password error occurs. If the password error occurred, the UART function of device stops without returning error code.

Note 5: In case of the blank product, the device receives Intel Hex Format immediately after receiving PCSA without receiving password strings. In this time, because the device ignores the data except the start mark data (3AH for “:”) as Intel Hex Format data, even if external controller transmitted dummy password strings, process operates correctly. However, if the dummy password strings contain data “3AH”, the device detects it as start mark data mistakenly, and device stops process without returning error code. Therefore, if these process becomes issue, the external controller should not transmit the dummy password strings.

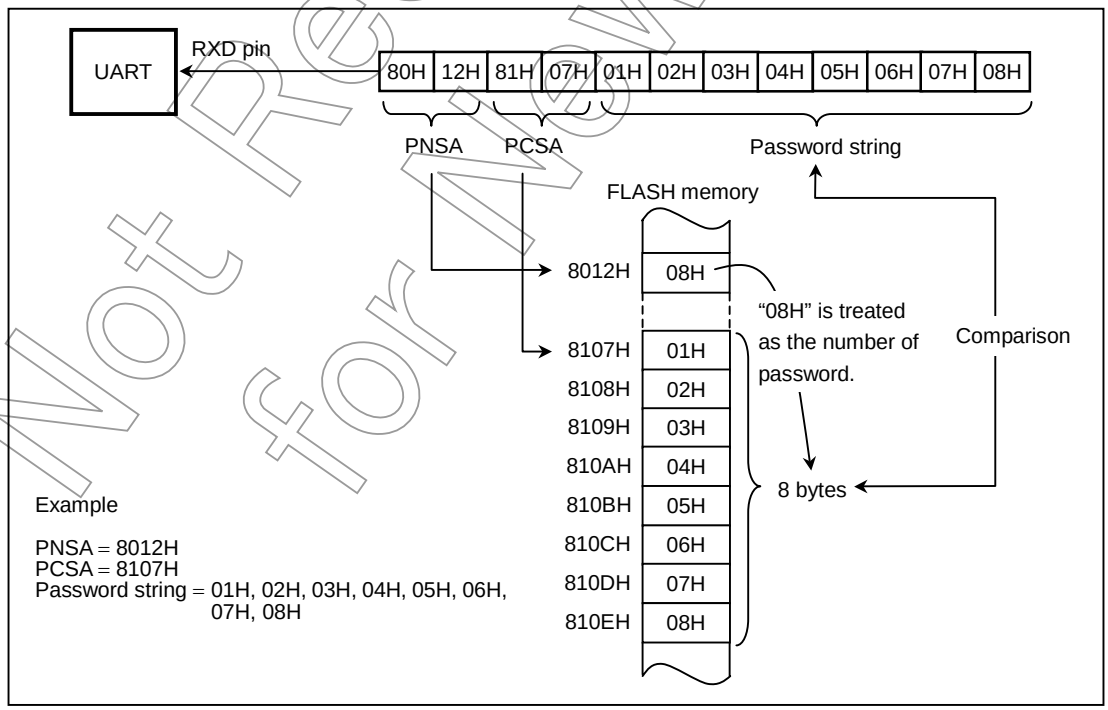


Figure 2.16.6 Example of password compare

2.16.11.1 Confirmation Method of the Blank Product and Non Blank Product

The external controller can confirm whether the device is the blank product or not, by transmission of data described below.

(1) Executes FLASH memory writing mode or RAM loader mode.
(2) Transmits the PNSA and PCSA.
(3) Transmits the end record.
(4) In case of the blank product, the device sends checksum of flash memory. In case of the non blank product, the device doesn't send checksum of flash memory but the UART function stops without sending any data.

The external controller can confirm the blank product and non blank product by receiving checksum.

Note: When the UART function stops in non blank product, the TMP86FM29 should be reset by pin reset input for restarting the Serial PROM Mode.

2.16.11.2 Password String

A string of passwords in the received data are compared with the data in the FLASH memory. In the following cases, a password error occurs:

- When the received data does not match the data in the FLASH memory

2.16.11.3 Handling of Password Error

If a password error occurs, the UART function of TMP86FM29 stops without returning error code to the controller. Therefore, when a password error occurs, the TMP86FM29 should be reset by $\overline{\text{RESET}}$ pin input.

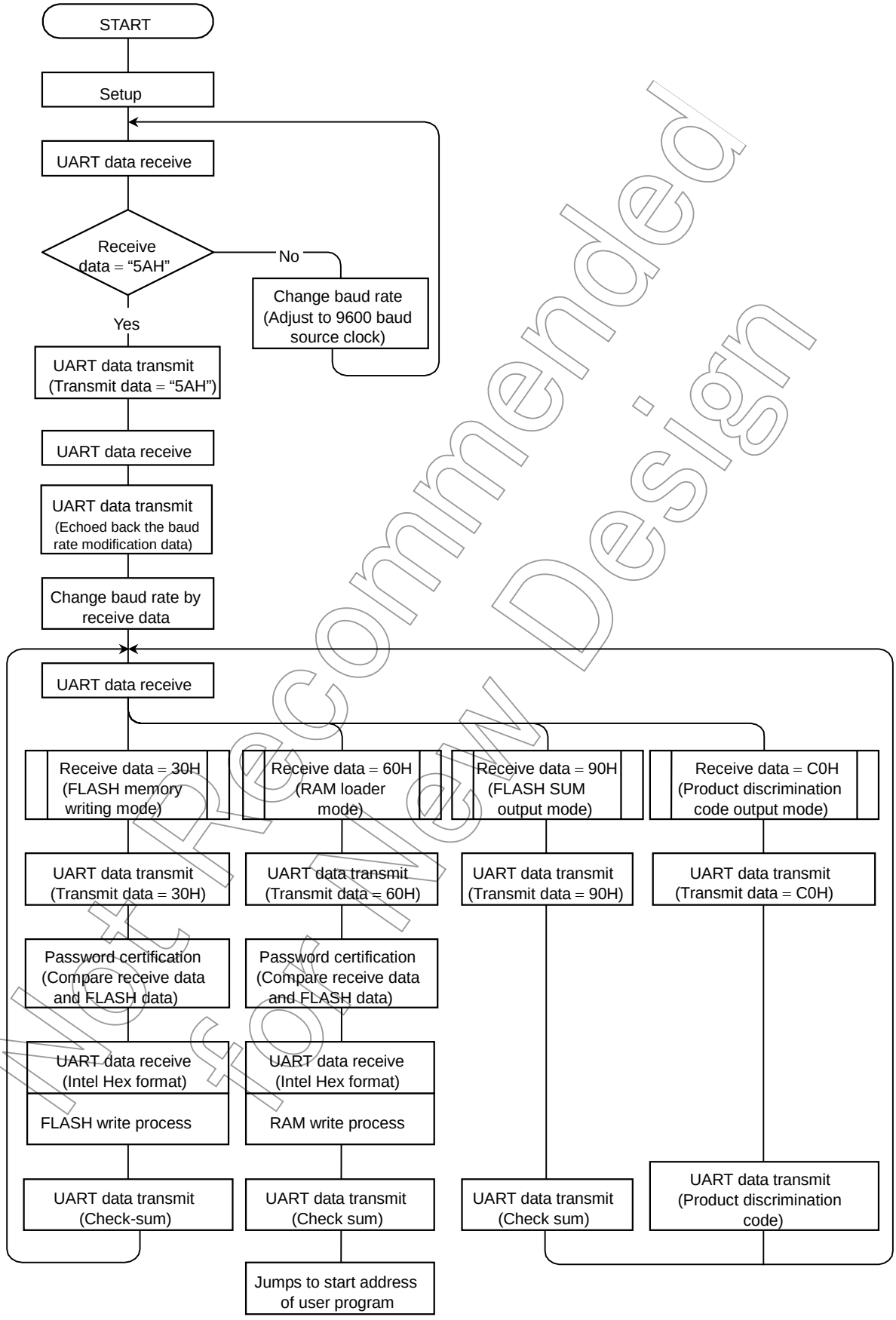
2.16.12 Product Discrimination Code

The product discrimination code is a 13-byte data, that includes the start address and the end address of ROM. Table 2.16.15 shows the product discrimination code format.

Table 2.16.15 Product Discrimination Code Format

Data	The Meaning of Data	In Case of TMP86FM29
1st	Start Mark (3AH)	3AH
2nd	The number of transfer data (from 3rd to 12th byte)	0AH
3rd	Length of address	02H
4th	Reserved data	00H
5th	Reserved data	00H
6th	Reserved data	00H
7th	Reserved data	00H
8th	The number of ROM block	01H
9th	The upper byte of the first address of ROM	80H
10th	The lower byte of the first address of ROM	00H
11th	The upper byte of the end address of ROM	FFH
12th	The lower byte of the end address of ROM	FFH
13th	Checksum of transferred data (from 3rd to 12th byte)	7FH

2.16.13 Flowchart



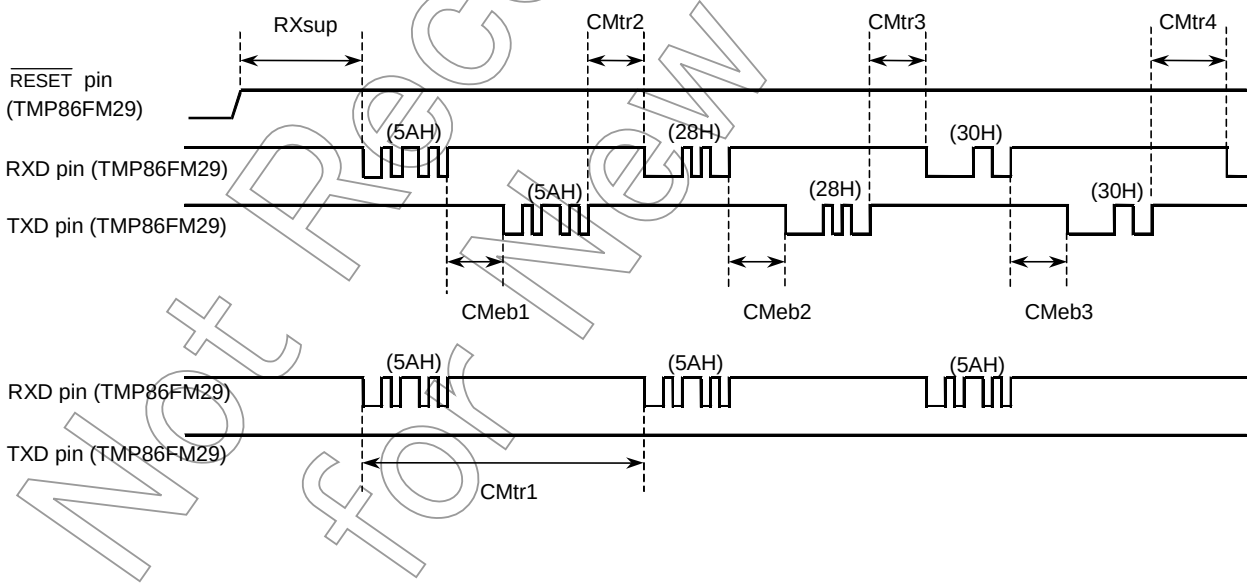
2.16.14 UART Timing

Table 2.16.16 UART Timing-1 (VDD = 2.7 V to 3.6 V, fc = 2 MHz to 16 MHz,Ta = 25°C)

Parameter	Symbol	The Number of Clock (fc)	Required Minimum Time	
			At fc = 2 MHz	At fc = 16 MHz
Time from the reception of a matching data until the output of an echo back	CMeb1	Approx. 600	300 μs	37.5 μs
Time from the reception of a baud rate modification data until the output of an echo back	CMeb2	Approx. 700	350 μs	43.7 μs
Time from the reception of an operation command until the output of an echo back	CMeb3	Approx. 600	300 μs	37.5 μs
Calculation time of checksum	CKsm	Approx. 1573000	786.5 ms	98.3 ms

Table 2.16.17 UART Timing-2 (VDD = 2.7 V to 3.6 V, fc = 2 MHz to 16 MHz,Ta = 25°C)

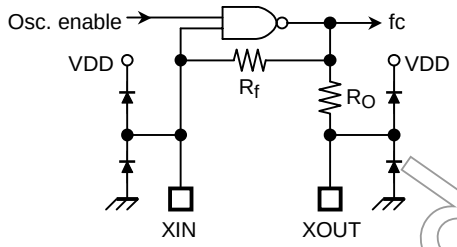
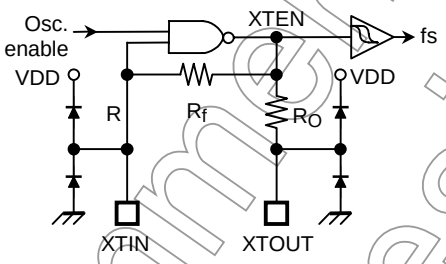
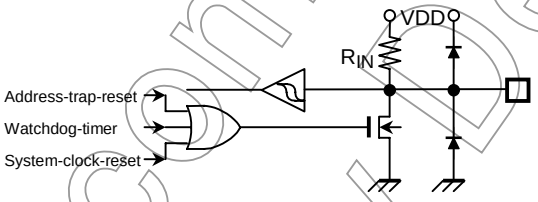
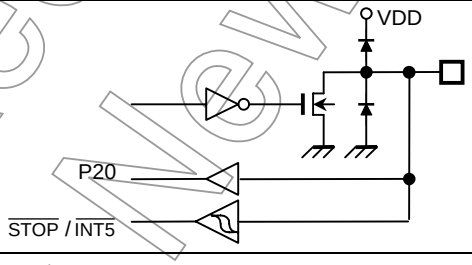
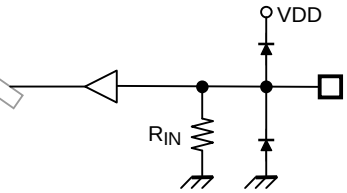
Parameter	Symbol	The Number of Clock (fc)	Required Minimum Time	
			At fc = 2 MHz	At fc = 16 MHz
Time from reset release until acceptance of start bit of RXD pin	RXsup	110000	55 ms	6.9 ms
Time between a matching data and the next matching data	CMtr1	28500	14.3 ms	1.8 ms
Time from the echo back of matching data until the acceptance of baud rate modification data	CMtr2	600	300 μs	37.5 μs
Time from the output of echo back of baud rate modification data until the acceptance of an operation command	CMtr3	750	375 μs	46.9 μs
Time from the output of echo back of operation command until the acceptance of Password count storage addresses	CMtr4	950	475 μs	59.4 μs



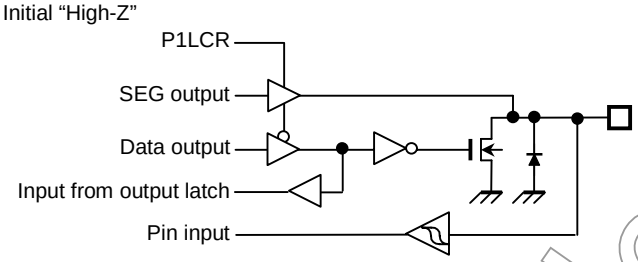
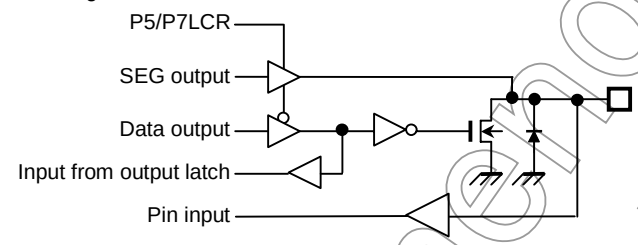
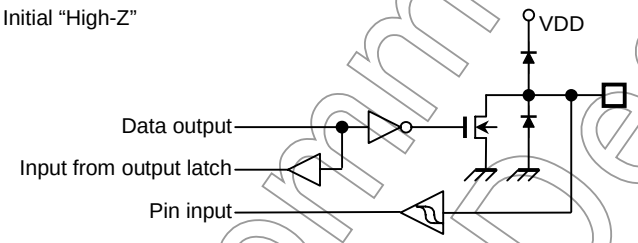
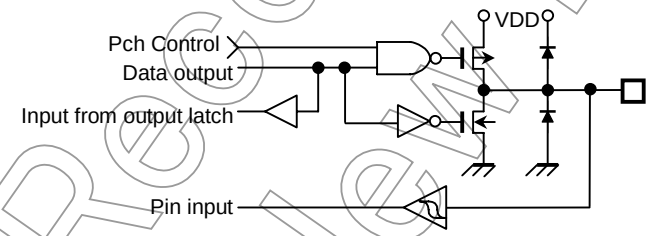
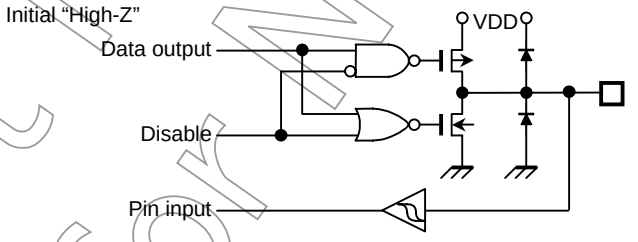
Input/Output Circuitry

(1) Control pins

The input/output circuitries of the TMP86FM29 control pins are shown below.

Control Pin	I/O	Input/Output Circuitry	Remarks
XIN XOUT	Input Output		Resonator connecting pins (high-frequency) R_f = 3 MΩ (typ.) R_O = 1 kΩ (typ.)
XTIN XTOUT	Input Output	NORMAL1 mode: Refer to port P2 NORMAL2 mode: 	Resonator connecting pins (Low-frequency) R_f = 20 MΩ (typ.) R_O = 220 kΩ (typ.)
$\overline{\text{RESET}}$	I/O		Sink open drain output Hysteresis input Pull-up resistor R_{IN} = 220 kΩ (typ.)
$\overline{\text{STOP}}$ / $\overline{\text{INT5}}$	Input		Hysteresis input
TEST	Input		Pull-down resistor R_{IN} = 70 kΩ (typ.)

(2) Input/Output Ports

Port	I/O	Input/Output Circuitry	Remarks
P1	I/O	Initial "High-Z" 	Sink open drain output Hysteresis input
P5 P7	I/O	Initial "High-Z" 	Sink open drain output
P2	I/O	Initial "High-Z" 	Sink open drain output Hysteresis input
P3	I/O	Initial "High-Z" 	Sink open drain or C-MOS output Hysteresis input High current output (Nch) (Programmable port option)
P6	I/O	Initial "High-Z" 	Tri-state I/O Hysteresis input

Note: Port P1, P5 and P7 are sink open drain output. But they are also used as a segment output of LCD. Therefore, absolute maximum ratings of port input voltage should be used in −0.3 to V_{DD} + 0.3 [V].

Electrical Characteristics

Absolute Maximum Ratings (V_{SS} = 0 V)

Parameter	Symbol	Pins	Rating	Unit
Supply voltage	V_{DD}		−0.3 to 4.0	V
	V_{LCD}	V3 pin	−0.3 to 4.0	
Input voltage	V_{IN}		−0.3 to V_{DD} + 0.3	
Output voltage	V_{OUT1}		−0.3 to V_{DD} + 0.3	
Output current (Per 1 pin)	I_{OUT1}	P3, P6 ports	−1.8	mA
	I_{OUT2}	P1, P2, P5, P6, P7 ports	3.2	
	I_{OUT3}	P3 ports	30	
Output current (Total)	ΣI_{OUT2}	P1, P2, P5, P6, P7 ports	60	
	ΣI_{OUT3}	P3 ports	80	
Power dissipation [Topr = 85°C]	PD		350	mW
Soldering temperature (Time)	Tsld		260 (10 s)	°C
Storage temperature	Tstg		−55 to 125	
Operating temperature	Topr		−40 to 85	

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.

Recommended Operating Condition-1 (MCU mode) (V_{SS} = 0 V, Topr = −40 to 85°C)

Parameter	Symbol	Pins	Condition		Min	Max	Unit
Supply voltage	V_{DD}		fc = 16 MHz	NORMAL1, 2 mode	2.7	3.6	V
				IDLE0, 1, 2 mode			
			fc = 8 MHz (In case of connecting the resonator)	NORMAL1, 2 mode	1.8		
				IDLE0, 1, 2 mode			
			fc = 4.2 MHz (In case of external clock input)	NORMAL1, 2 mode			
				IDLE0, 1, 2 mode			
			fs = 32.768 kHz	SLOW1, 2 mode			
				SLEEP0, 1, 2 mode	1.8		
				STOP mode			
Input high level	V_{IH1}	Except Hysteresis input	$V_{DD} \geq 2.7$ V		$V_{DD} \times 0.70$	V_{DD}	
	V_{IH2}	Hysteresis input			$V_{DD} \times 0.75$		
	V_{IH3}		$V_{DD} < 2.7$ V		$V_{DD} \times 0.90$		
Input low level	V_{IL1}	Except Hysteresis input	$V_{DD} \geq 2.7$ V		0	$V_{DD} \times 0.30$	
	V_{IL2}	Hysteresis input				$V_{DD} \times 0.25$	
	V_{IL3}		$V_{DD} < 2.7$ V			$V_{DD} \times 0.10$	
Clock frequency (In case of connecting the resonator)	fc	XIN, XOUT	V_{DD} = 1.8 to 3.6 V		1.0	8.0	MHz
			V_{DD} = 2.7 to 3.6 V			16.0	
	fs	XTIN, XTOUT	V_{DD} = 1.8 to 3.6 V		30.0	34.0	kHz
Clock frequency (In case of external clock input)	fc	XIN, XOUT	V_{DD} = 1.8 to 3.6 V		1.0	4.2	MHz
			V_{DD} = 2.7 to 3.6 V			16.0	
	fs	XTIN, XTOUT	V_{DD} = 1.8 to 3.6 V		30.0	34.0	kHz
LCD reference voltage	V1		Booster circuit is enable ($V3 \geq V_{DD}$)		0.8	1.2	V
Capacity for LCD booster circuit	C_{LCD}		LCD booster circuit is enable ($V3 \geq V_{DD}$)		0.1	0.47	μF

Note: The recommended operating conditions for a device are operating conditions under which it can be guaranteed that the device will operate as specified. If the device is used under operating conditions other than the recommended operating conditions (Supply voltage, operating temperature range, specified AC/DC values etc.), malfunction may occur. Thus, when designing products which include this device, ensure that the recommended operating conditions for the device are always adhered to.

Recommended Operating Condition-2 (Serial PROM mode) (V_{SS} = 0 V, Topr = 25°C ± 5°C)

Parameter	Symbol	Pins	Condition	Min	Max	Unit
Supply voltage	VDD		2 MHz ≤ fc ≤16 MHz	2.7	3.6	V
Clock frequency	fc	XIN, XOUT	VDD = 2.7 to 3.6 V	2.0	16.0	MHz

Note: The operating temperature area of serial PROM mode is 25°C ± 5°C and the operating area of high frequency of serial PROM mode is different from MCU mode.

DC Characteristics (V_{SS} = 0 V, Topr = –40 to 85°C)

Parameter	Symbol	Pins	Condition		Min	Typ.	Max	Unit
Hysteresis voltage	V_{HS}	Hysteresis input	V_{DD} = 3.3 V		–	0.4	–	V
Input current	I_{IN1}	TEST	V_{DD} = 3.6 V, V_{IN} = 0 V		–	–	–5	μA
	I_{IN2}	Sink open drain, Tri-state	V_{DD} = 3.6 V, V_{IN} = 3.6 V/0 V		–	–	±5	
	I_{IN3}	$\overline{RESET}$	V_{DD} = 3.6 V, V_{IN} = 3.6 V		–	–	+5	
Input resistance	R_{IN1}	TEST pull down	V_{DD} = 3.6 V, V_{IN} = 3.6 V		–	70	–	kΩ
	R_{IN2}	$\overline{RESET}$ pull up	V_{DD} = 3.6 V, V_{IN} = 0 V		100	220	450	
High frequency feedback resistor	R_{FB}	XOUT	V_{DD} = 3.6 V		–	3	–	MΩ
Low frequency feedback resistor	R_{FBT}	XTOUT	V_{DD} = 3.6 V		–	20	–	
Output leakage current	I_{LO}	Sink open drain, Tri-state	V_{DD} = 3.6 V V_{OUT} = 3.4V/0.2 V		–	–	±10	μA
Output high voltage	V_{OH}	CMOS, Tri-state	V_{DD} = 3.6 V, I_{OH} = –0.6 mA		3.2	–	–	V
Output low voltage	V_{OL}	Except XOUT, P3 port	V_{DD} = 3.6 V, I_{OL} = 0.9 mA		–	–	0.4	
Output low current	I_{OL}	P3 port	V_{DD} = 3.6 V, V_{OL} = 1.0 V		–	6	–	mA
LCD output voltage (LCD booster is enable)	$V_{2\text{-}3OUT}$	V2 pin	V3 ≥ V_{DD} Reference supply pin: V1 SEG/COM pin: No load		–	V1 x 2	–	V
		V3 pin			–	V1 x 3	–	
Supply current in NORMAL 1, 2 mode		Fetch area: Flash area	V_{DD} = 3.6 V V_{IN} = 3.4 V/0.2 V fc = 16 MHz fs = 32.768 kHz	MNP = "1"	–	5.3	7.3	mA
		Fetch area: RAM area		MNP = "0"	–	3.4	5.2	
Supply current in IDLE 0, 1, 2 mode				MNP•ATP = "1"	–	3.1	5.2	
				MNP•ATP = "0"	–	2.2	4.2	
Supply current in SLOW 1 mode		Fetch area: Flash area	V_{DD} = 3.0 V V_{IN} = 2.8 V/0.2 V fs = 32.768 kHz	MNP = "1"	–	850	1200	μA
		Fetch area: RAM area		MNP = "0"	–	7	19	
Supply current in SLEEP 1 mode				MNP•ATP = "1"	–	850	1200	
				MNP•ATP = "0"	–	5.5	17	
Supply current in SLEEP 0 mode				MNP•ATP = "1"	–	850	1200	
				MNP•ATP = "0"	–	4.5	15	
Supply current in STOP mode			V_{DD} = 3.6 V V_{IN} = 3.4 V/0.2 V		–	0.5	10	

Note 1: Typical values show those at Topr = 25°C.

Note 2: Input current (I_{IN1}, I_{IN2}): The current through pull-up or pull-down resistor is not included.

Note 3: I_{DD} does not include I_{REF} current.

Note 4: The supply currents of SLOW2 and SLEEP2 modes are equivalent to IDLE0, IDLE1, IDLE2.

Note 5: MNP (MNPWDW) shows bit0 in EEPCR register and ATP (ATPWDW) shows bit1 in EEPCR register.

Note 6: "Fetch" means reading operation of FLASH data as an instruction by CPU.

AD Conversion Characteristics (V_{SS} = 0.0 V, 2.7 V ≤ V_{DD} ≤ 3.6 V, Topr = –40 to 85°C)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage	V_{AREF}		A_{VDD} – 1.0	–	A_{VDD}	V
Power supply voltage of analog control circuit	A_{VDD}		V_{DD}			
Analog reference voltage range (Note 4)	ΔV_{AREF}		2.5	–	–	
Analog input voltage	V_{AIN}		V_{SS}	–	V_{AREF}	
Power supply current of analog reference voltage	I_{REF}	V_{DD} = A_{VDD} = V_{AREF} = 3.6 V V_{SS} = 0.0 V	–	0.35	0.61	mA
Non linearity error		V_{DD} = A_{VDD} = 2.7 V V_{SS} = 0.0 V V_{AREF} = 2.7 V	–	–	±2	LSB
Zero point error			–	–	±2	
Full scale error			–	–	±2	
Total error			–	–	±2	

(V_{SS} = 0.0 V, 2.0 V ≤ V_{DD} < 2.7 V, Topr = –40 to 85°C)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage	V_{AREF}		A_{VDD} – 0.6	–	A_{VDD}	V
Power supply voltage of analog control circuit	A_{VDD}		V_{DD}			
Analog reference voltage range (Note 4)	ΔV_{AREF}		2.0	–	–	
Analog input voltage	V_{AIN}		V_{SS}	–	V_{AREF}	
Power supply current of analog reference voltage	I_{REF}	V_{DD} = A_{VDD} = V_{AREF} = 2.0V V_{SS} = 0.0 V	–	0.20	0.34	mA
Non linearity error		V_{DD} = A_{VDD} = 2.0 V V_{SS} = 0.0 V V_{AREF} = 2.0 V	–	–	±4	LSB
Zero point error			–	–	±4	
Full scale error			–	–	±4	
Total error			–	–	±4	

(V_{SS} = 0.0 V, 1.8 V ≤ V_{DD} < 2.0 V, Topr = –10 to 85°C) (Note 5)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage	V_{AREF}		A_{VDD} – 0.1	–	A_{VDD}	V
Power supply voltage of analog control circuit	A_{VDD}		V_{DD}			
Analog reference voltage range (Note 4)	ΔV_{AREF}		1.8	–	–	
Analog input voltage	V_{AIN}		V_{SS}	–	V_{AREF}	
Power supply current of analog reference voltage	I_{REF}	V_{DD} = A_{VDD} = V_{AREF} = 1.8 V V_{SS} = 0.0 V	–	0.18	0.31	mA
Non linearity error		V_{DD} = A_{VDD} = 1.8 V V_{SS} = 0.0 V V_{AREF} = 1.8 V	–	–	±4	LSB
Zero point error			–	–	±4	
Full scale error			–	–	±4	
Total error			–	–	±4	

Note 1: The total error includes all errors except a quantization error, and is defined as a maximum deviation from the ideal conversion line.

Note 2: Conversion time is different in recommended value by power supply voltage.
About conversion time, please refer to “2.15.2 Register configration”.

Note 3: Please use input voltage to AIN input Pin in limit of VAREF – VSS.
When voltage of range outside is input, conversion value becomes unsettled and gives affect to other channel conversion value.

Note 4: Analog Reference Voltage Range: ΔVAREF = VAREF – VSS

Note 5: When AD is used with VDD < 2.0 V, the guaranteed temperature range varies with the operating voltage.

Note 6: When AD converter is not used, fix the AVDD pin and VAREFpin on the V_{DD} level.

AC Characteristics (V_{SS} = 0 V, V_{DD} = 2.7 to 3.6 V, Topr = –40 to 85°C)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Machine cycle time	tcy	NORMAL1, 2 mode	0.25	–	4	μs
		IDLE1, 2 mode				
		SLOW1, 2 mode	117.6	–	133.3	
		SLEEP1, 2 mode				
High Level clock pulse width	twcH	For external clock operation (XIN input), fc = 16 MHz	–	31.25	–	ns
Low level clock pulse width	twcL					
High level clock pulse width	twcH	For external clock operation (XTIN input), fs = 32.768 kHz	–	15.26	–	μs
Low level clock pulse width	twcL					

(V_{SS} = 0 V, V_{DD} = 1.8 to 3.6 V, Topr = –40 to 85°C)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Machine cycle time	tcy	NORMAL1, 2 mode	0.5	–	4	μs
		IDLE1, 2 mode				
		SLOW1, 2 mode	117.6	–	133.3	
		SLEEP1, 2 mode				
High level clock pulse width	twcH	For external clock operation (XIN input), fc = 4.2 MHz	–	119.04	–	ns
Low level clock pulse width	twcL					
High level clock pulse width	twcH	For external clock operation (XTIN input), fs = 32.768 kHz	–	15.26	–	μs
Low level clock pulse width	twcL					

Timer Counter 1 input (ECIN) Characteristics (V_{SS} = 0 V, Topr = -40 to 85°C)

Parameter	Symbol	Condition		Min	Typ.	Max	Unit
TC1 input (ECIN input)	t_{TC1}	Frequency measurement mode V_{DD} = 2.7 to 3.6 V	Single edge count	–	–	16	MHz
			Both edge count				
		Frequency measurement mode V_{DD} = 1.8 to 2.7 V	Single edge count	–	–	8	
			Both edge count				

Flash Characteristics (V_{SS} = 0 V)

Parameter	Condition	Min	Typ.	Max	Unit
Number of guaranteed writes (page writing) to Flash memory in serial PROM mode	V_{DD} = 2.7 to 3.6 V, 2 MHz ≤ fc ≤ 16 MHz (Topr = 25°C ± 5°C)	–	–	10^5	Times

Recommended Oscillating Conditions

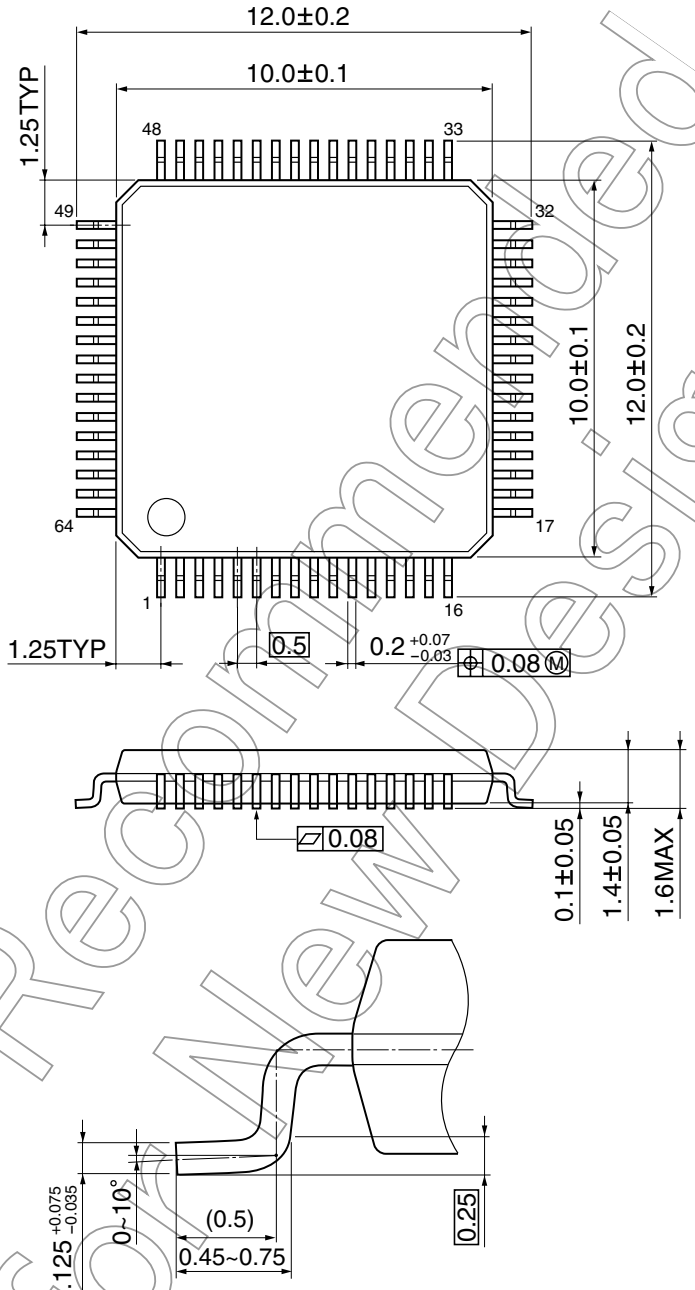
Note 1: An electrical shield by metal shield plate on the surface of IC package is recommended in order to protect the device from the high electric field stress applied from CRT (Cathodic Ray Tube) for continuous reliable operation.

Note 2: The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are subject to change. For up-to-date information, please refer to the following
http://www.murata.co.jp/search/index.html

Package Dimensions

P-LQFP64-1010-0.50E

Unit: mm



P-QFP64-1414-0.80C

Unit: mm