TOSHIBA

TOSHIBA Original CMOS 16-Bit Microcontroller

TLCS-900/L1 Series

TMP91FW27UG
TMP91FW27FG

TOSHIBA CORPORATION

Semiconductor Company

Preface

Thank you very much for making use of Toshiba microcomputer LSI. Before using this LSI, refer to section “Points of Note and Restrictions”.

Especially, take care below cautions.

****CAUTION****

How to release the HALT mode

Usually, interrupts can release all halts stats. However, the interrupts = ($\overline{\text{NMI}}$, INT0, INTRTC), which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 5 clocks of f_{FPH}) with IDLE1 or STOP mode (IDLE2 is not applicable to this case). (In this case, an interupt request is kept on hold internally.)

If another interupt is generated after it has shifted to HALT mode completely, halt status can be released without difficultly. The priority of this interrupt is compare with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

CMOS 16-Bit Microcontrollers

TMP91FW27UG / TMP91FW27FG

1. Outline and Features

TMP91FW27 is a high-speed 16-bit microcontroller designed for the control of various mid-to large-scale equipment.

TMP91FW27UG and TMP91FW27FG come in a 64-pin flat package. Listed below are the features.

(1) High-speed 16-bit CPU (900/L1 CPU)
- Instruction mnemonics are upward-compatible with TLCS-90/900
- 16 Mbytes of linear address space
- General-purpose registers and register banks
- 16-bit multiplication and division instructions; bit transfer and arithmetic instructions
- Micro DMA: 4 channels (593 ns /2 bytes at 27 MHz)

(2) Minimum instruction execution time: 148 ns (at 27 MHz)

(3) Built-in RAM: 12 Kbytes
Built-in ROM: 128-Kbyte Flash memory
4-Kbyte mask ROM (used for booting)

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

(5) 8-bit timers: 6 channels

(6) 16-bit timers: 1 channel

(7) General-purpose serial interface: 2 channels
- UART/Synchronous mode: 2 channels
- IrDA Ver.1.0 (115.2 kbps) mode selectable: 1 channel

(8) Serial bus interface: 1 channel
- I^2C bus mode/clock synchronous mode selectable

(9) 10-bit AD converter (sample hold circuit is inside): 4 channels

(10) Watchdog timer

(11) Special timer for CLOCK

(12) Chip select/Wait controller: 4 blocks

(13) Interrupts: 34 interrupts
- 9 CPU interrupts: Software interrupt instruction and illegal instruction
- 21 internal interrupts: 7 priority levels are selectable
- 4 external interrupts: 7 priority levels are selectable (among 3 interrupts are selectable edge mode)

(14) Input/output ports: 53 pins

(15) Stand-by function

Three Halt modes: IDLE2 (programmable), IDLE1 and STOP

(16) Clock controller
- Clock gear function: Select a High-frequency clock fc to fc/16
- Special timer for CLOCK (fs = 32.768 kHz)

(17) Operating voltage
- Vcc = 2.7 V to 3.6 V (fc max = 27 MHz, flash memory read operation)
- Vcc = 2.2 V to 3.6 V (fc max = 16 MHz, flash memory read operation)
- Vcc = 2.7 V to 3.6 V (fc max = 27 MHz, flash memory erase/program operations)

(18)Package
- LQFP64-P-1010-0.50D (TMP91FW27UG)
- QFP64-P-1414-0.80A (TMP91FW27FG)

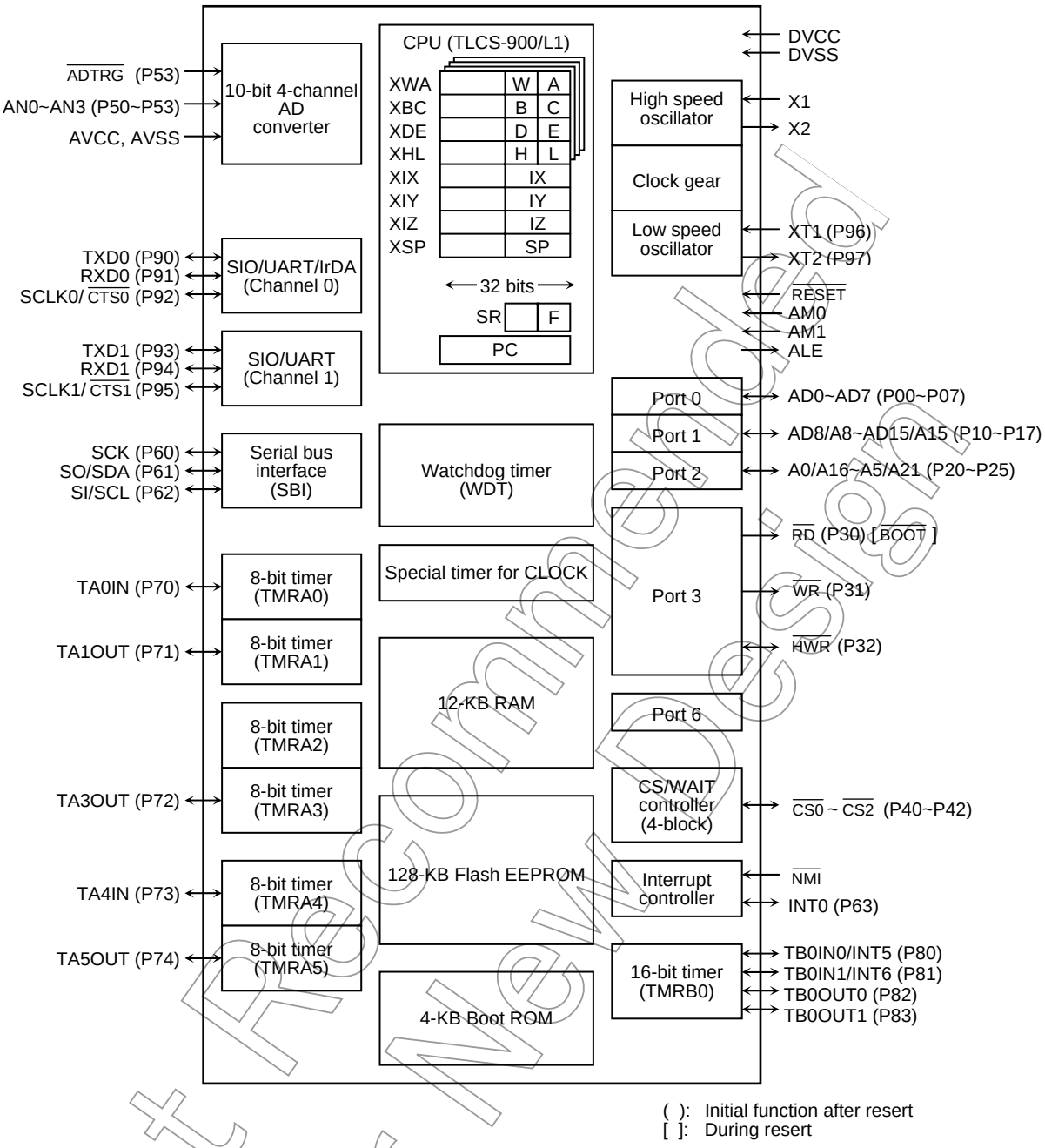


Figure 1.1 TMP91FW27 Block Diagram

2. Pin Assignment and Pin Functions

The assignment of input/output pins for the TMP91FW27, their names and functions are as follows:

2.1 Pin Assignment Diagram

Figure 2.1.1 shows the pin assignment of the TMP91FW27UG and TMP91FW27FG.

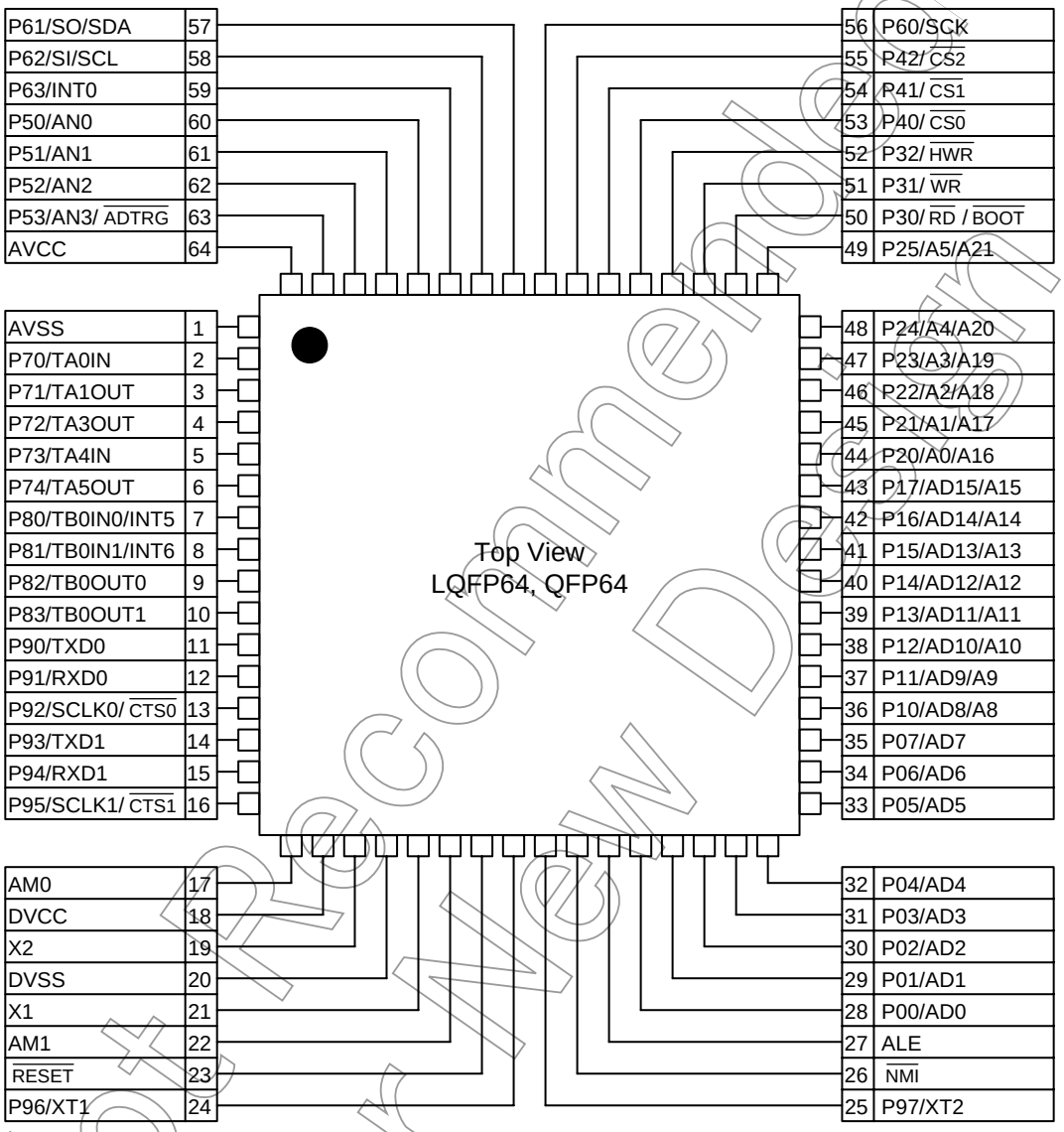


Figure 2.1.1 Pin Assignment Diagram (64-pin LQFP, QFP)

2.2 Pin Names and Functions

The names of the input/output pins and their functions are described below. Table 2.2.1 and Table 2.2.2 show Pin names and functions.

Table 2.2.1 Pin Names and Functions (1/2)

Pin Names	Number of Pins	I/O	Functions
P00 to P07	8	I/O	Port 0: I/O port that allows I/O to be selected at the bit level
AD0 to AD7		I/O	Address data (lower): 0 to 7 of address/data bus
P10 to P17	8	I/O	Port1: I/O port that allows I/O to be selected at the bit level
AD8 to AD15		I/O	Address data (upper): 8 to 15 of address/data bus
A8 to A15		Output	Address: 8 to 15 of address bus
P20 to P25	6	I/O	Port 2: I/O port that allows I/O to be selected at the bit level
A0 to A5		Output	Address: 0 to 5 of address bus
A16 to A21		Output	Address: 16 to 21 of address bus
P30	1	Output	Port 30: Output Port
$\overline{RD}$		Output	Read: Strobe signal for reading external memory When read internal area also, output $\overline{RD}$ by setting to P3<P30> = 0 and P3FC<P30F> = 1.
$\overline{BOOT}$		Input	This pin sets single boot mode (only during reset). For the details, please refer to section 3.2.3, "Operation modes".
P31	1	Output	Port 31: Output port
$\overline{WR}$		Output	Write: Strobe signal for writing data to pins AD0 to AD7
P32	1	I/O	Port 32: I/O port (with pull-up resistor)
$\overline{HWR}$		Output	High Write: Strobe signal for writing data to pins AD8 to AD15
P40	1	I/O	Port 40: I/O port (with pull-up resistor)
$\overline{CS0}$		Output	Chip select 0: Outputs "0" when address is within specified address area.
P41	1	I/O	Port41: I/O port (with pull-up resistor)
$\overline{CS1}$		Output	Chip select 1: Outputs "0" when address is within specified address area.
P42	1	I/O	Port 42: I/O port (with pull-up resistor)
$\overline{CS2}$		Output	Chip select 2: Outputs "0" when address is within specified address area.
P50 to P53	4	Input	Port 5: Input port
AN0 to AN3		Input	Analog input: Analog input pins of the AD converter
$\overline{ADTRG}$		Input	AD trigger: Pin used to request AD start (shared with P53).
P60	1	I/O	Port 60: I/O port
SCK		I/O	Serial bus interface clock I/O at SIO mode
P61	1	I/O	Port 61: I/O port
SO		Output	Serial bus interface send data at SIO mode
SDA		I/O	Serial bus interface send/receive data at I^2C mode Open-drain output mode by programmable
P62	1	I/O	Port 62: I/O port
SI		Input	Serial bus interface receive data at SIO mode
SCL		I/O	Serial bus interface clock I/O at I^2C mode Open-drain output mode by programmable
P63	1	I/O	Port 63: I/O port (Schmitt input)
INT0		Input	Interrupt request pin 0: Interrupt request pin with level/ rising/falling edge
P70	1	I/O	Port 70: I/O port
TA0IN		Input	8-bit timer 0 input: Input pin of 8-bit timer TMRA0
P71	1	I/O	Port 71: I/O port
TA1OUT		Output	8-bit timer 1 output: Output pin of 8-bit timer TMRA0 or TMRA1
P72	1	I/O	Port 72: I/O port
TA3OUT		Output	8-bit timer 3 output: Output pin of 8-bit timer TMRA2 or TMRA3

Table 2.2.2 Pin Names and Functions (2/2)

Pin Names	Number of Pins	I/O	Functions
P73 TA4IN	1	I/O Input	Port 73: I/O port 8-bit timer 4 Input: Input pin of 8-bit timer TMRA4
P74 TA5OUT	1	I/O Output	Port 74: I/O port 8-bit timer 5 output: Output pin of 8-bit timer TMRA4 or TMRA5
P80 TB0IN0 INT5	1	I/O Input Input	Port 80: I/O port 16-bit timer 0 Input 0: Input of count/capture trigger in 16-bit timer TMRB0 Interrupt request pin 5: Interrupt request pin with selectable rising/falling edge
P81 TB0IN1 INT6	1	I/O Input Input	Port 81: I/O port 16-bit timer 0 Input 1: Input of count/capture trigger in 16-bit timer TMRB0 Interrupt request pin 6: Interrupt request pin of rising edge
P82 TB0OUT0	1	I/O Output	Port 82: I/O port 16-bit timer 0 output 0: Outpit pin of 16-bit timer TMRB0
P83 TB0OUT1	1	I/O Output	Port 83: I/O port 16-bit timer 0 output 1: Output pin of 16-bit timer TMRB0
P90 TXD0	1	I/O Output	Port 90: I/O port Serial 0 send data: Open-drain output pin by programmable
P91 RXD0	1	I/O Input	Port 91: I/O port Serial 0 receive data
P92 SCLK0 $\overline{\text{CTS0}}$	1	I/O I/O Input	Port 92: I/O port Serial 0 clock I/O Serial 0 data send enable (Clear to send)
P93 TXD1	1	I/O Output	Port 93: I/O port Serial 1 send data: Open-drain output pin by programmable
P94 RXD1	1	I/O Input	Port 94: I/O port Serial 1 receive data
P95 SCLK1 $\overline{\text{CTS1}}$	1	I/O I/O Input	Port 95: I/O port Serial 1 clock I/O Serial 1 data send enable (Clear to send)
P96 XT1	1	I/O Input	Port 96: I/O port. Open-drain output pin. Low frequency oscillator connection pin
P97 XT2	1	I/O Output	Port 97: I/O port. Open-drain output pin. Low frequency oscillator connection pin
ALE	1	Output	Address latch enable (It can be set as prohibition of an output for noize reduction.)
$\overline{\text{NMI}}$	1	Input	Non-Maskable interrupt request pin: Interrupt request pin with programmable falling edge level or with both edge levels programmable (Schmitt input).
AM0 and AM1	2	Input	Operation mode: Fixed to AM1 = “1” and AM0 = “1”.
$\overline{\text{RESET}}$	1	Input	Reset: Initialize LSI. (Schmitt input, with pull-up resistor)
AVCC	1		Pin used to both power supply pin for AD converter and standard power supply for AD converter (H).
AVSS	1		Pin used to both GND pin for AD converter (0 V) and standard power supply pin for AD converter (L).
X1/X2	2	I/O	High frequency oscillator connection pin.
DVCC DVSS	1 1		Power supply pins (All DVCC pins should be connected with the power Supply pin). GND pins (All pins shuold be connected with GND(0V).)

3. Functional Description

This section shows the hardware configuration of the TMP91FW27 and explains how it operates.

This device is a version of the created by replacing the predecessor's internal mask ROM with a 128-Kbyte internal flash memory and expanding its internal RAM size to 12 Kbytes. The configuration and the functionality of this device are the same as those of the TMP91CP27. For the functions of this device that are not described here, refer to the TMP91CP27 data sheet.

3.1 Memory Map

Figure 3.1.1 shows a memory map of the TMP91FW27 in single-chip mode and its memory areas that can be accessed in each addressing mode of the CPU.

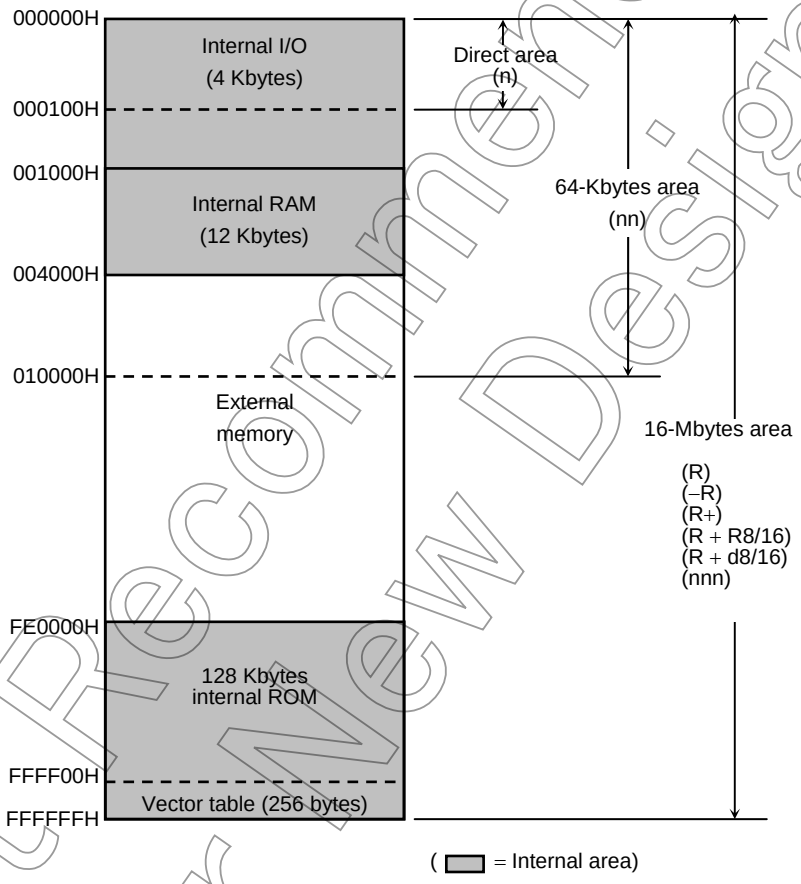


Figure 3.1.1 Memory Map (Single-Chip Mode)

3.2 Flash Memory

The TMP91FW27 incorporates flash memory that can be electrically erased and programmed using a single 3V power supply.

The flash memory is programmed and erased using JEDEC-standard commands. After a program or erase command is input, the corresponding operation is automatically performed internally. Erase operations can be performed by the entire chip (chip erase) or on a sector basis (sector erase).

The configuration and operations of the flash memory are described below.

3.2.1 Features

- Power supply voltage for program/erase operations
  Vcc = 2.7 V to 3.6 V (−10 °C to 40 °C)
- Configuration
  64 K × 16 bits (128 Kbytes)
- Functions
  Single-word programming
  Chip erase
  Sector erase
  Data polling/Toggle bit
- Sector size
  4 Kbytes × 32
- Mode control
  JEDEC-standard commands
- Programming method
  On-board programming
  Parallel programmer
- Security
  Write protection
  Read protection

3.2.2 Block Diagram

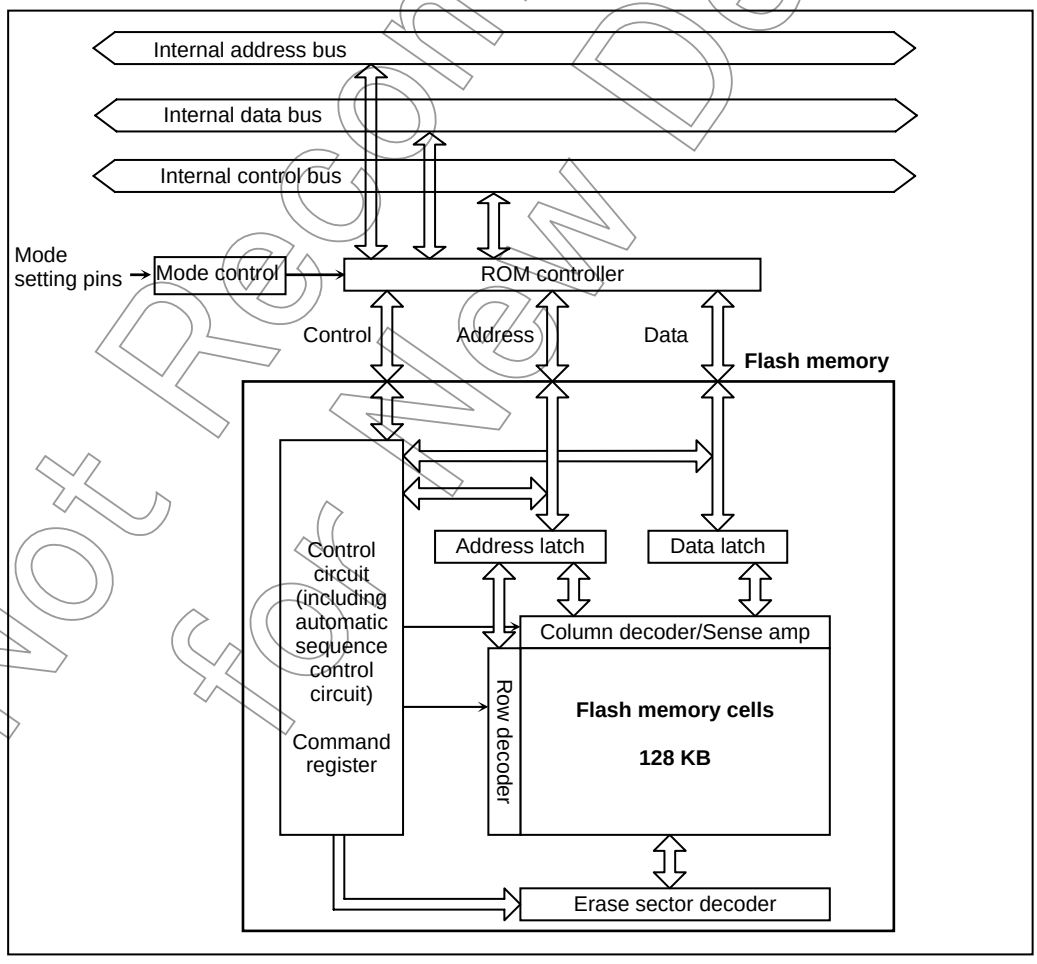


Figure 3.2.1 Block Diagram of Flash Memory Unit

3.2.3 Operation Modes

3.2.3.1 Overview

The following three types of operation modes are available to control program/erase operations on the flash memory.

Table 3.2.1 Description of Operation Modes

Operation Mode Name	Description
Single Chip mode	After reset release, the device starts up from the internal flash memory. Single Chip mode is further divided into two modes: "Normal mode" is a mode in which user application programs are executed, and "User Boot mode" is used to program the flash memory on-board. The means of switching between these two modes can be set by the user as desired. For example, it can be set so that Port 00 = "1" selects Normal mode and Port 00 = "0" selects User Boot mode. The user must include a routine to handle mode switching in a user application program.
Normal mode	In this mode, the device starts up from a user application program.
User Boot mode	In this mode, the flash memory can be programmed by a user-specified method.
Single Boot mode	After reset release, the device starts up from the internal boot ROM (mask ROM). The boot ROM includes an algorithm which allows a program for programming/erasing the flash memory on-board via a serial port to be transferred to the device's internal RAM. The transferred program is then executed in the internal RAM so that the flash memory can be programmed/erased by receiving data from an external host and issuing program/erase commands.
Programmer mode	This mode enables the internal flash memory to be programmed/erased using a general-purpose programmer. For programmers that can be used, please contact your local Toshiba sales representative.

Of the modes listed in Table 3.2.1, the internal flash memory can be programmed in User Boot mode, Single Boot mode and Programmer mode.

The mode in which the flash memory can be programmed/erased while mounted on the user board is defined as the on-board programming mode. Of the modes listed above, Single Boot mode and User Boot mode are classified as on-board programming modes. Single Boot mode supports Toshiba's proprietary programming/erase method using serial I/O. User Boot mode (within Single Chip mode) allows the flash memory to be programmed/erased by a user-specified method.

Programmer mode is provided with a read protect function which prohibits reading of ROM data. By enabling the read protect function upon completion of programming, the user can protect ROM data from being read by third parties.

The operation mode — Single Chip mode, Single Boot mode or Programmer mode — is determined during reset by externally setting the input levels on the AM0, AM1 and $\overline{\text{BOOT}}$ (P30) pins.

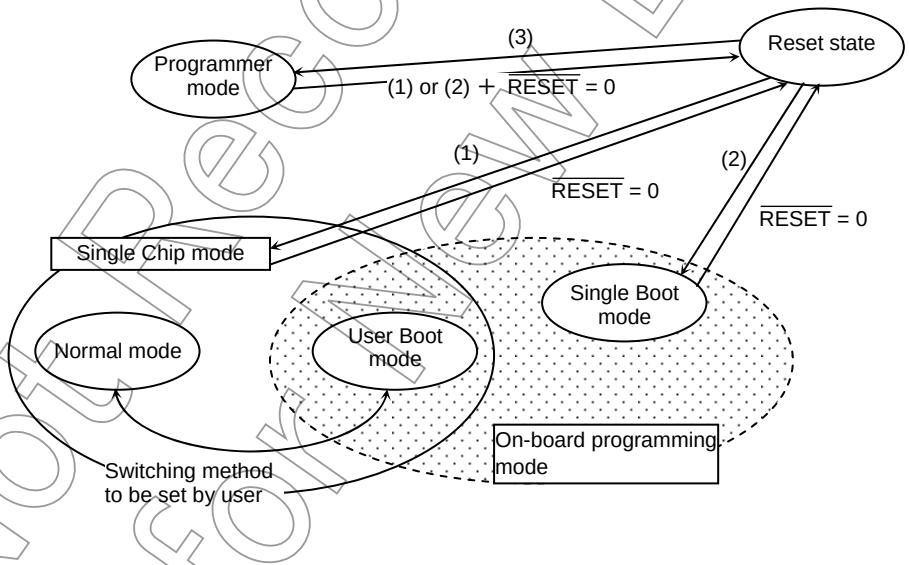
Except in Programmer mode which is entered with $\overline{\text{RESET}}$ held at "0", the CPU will start operating in the selected mode after the reset state is released. Once the operation mode has been set, make sure that the input levels on the mode setting pins are not changed during operation. Table 3.2.2 shows how to set each operation mode, and Figure 3.2.2 shows a mode transition diagram.

Table 3.2.2 Operation Mode Pin Settings

	Operation Mode	Input Pins			
		$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$ (P30)	AM1	AM0
(1)	Single Chip mode (Normal or User Boot mode)	↑	1	1	1
(2)	Single Boot mode	↑	0	1	1
(3)	Programmer mode	0	—	1	0

Although P30 is an output port, it becomes an input port with pull-up resistor only during a reset. After a reset, P30 operates as follows depending on the operation mode.

- Single chip mode: Output port (Without pull-up resistor)
- Single boot mode: Pull-up (Input gate is invalid, and output gate is in high impedance.)



Numbers in ( ) correspond to the operation mode pin settings shown in Table 3.2.2

Figure 3.2.2 Mode Transition Diagram

3.2.3.2 Reset Operation

To reset the device, hold the $\overline{\text{RESET}}$ input at "0" for at least 10 system clocks while the power supply voltage is within the rated operating voltage range and the internal high-frequency oscillator is oscillating stably.

3.2.3.3 Memory Map for Each Operation Mode

In this product, the memory map varies with operation mode. The memory map and sector address ranges for each operation mode are shown below.

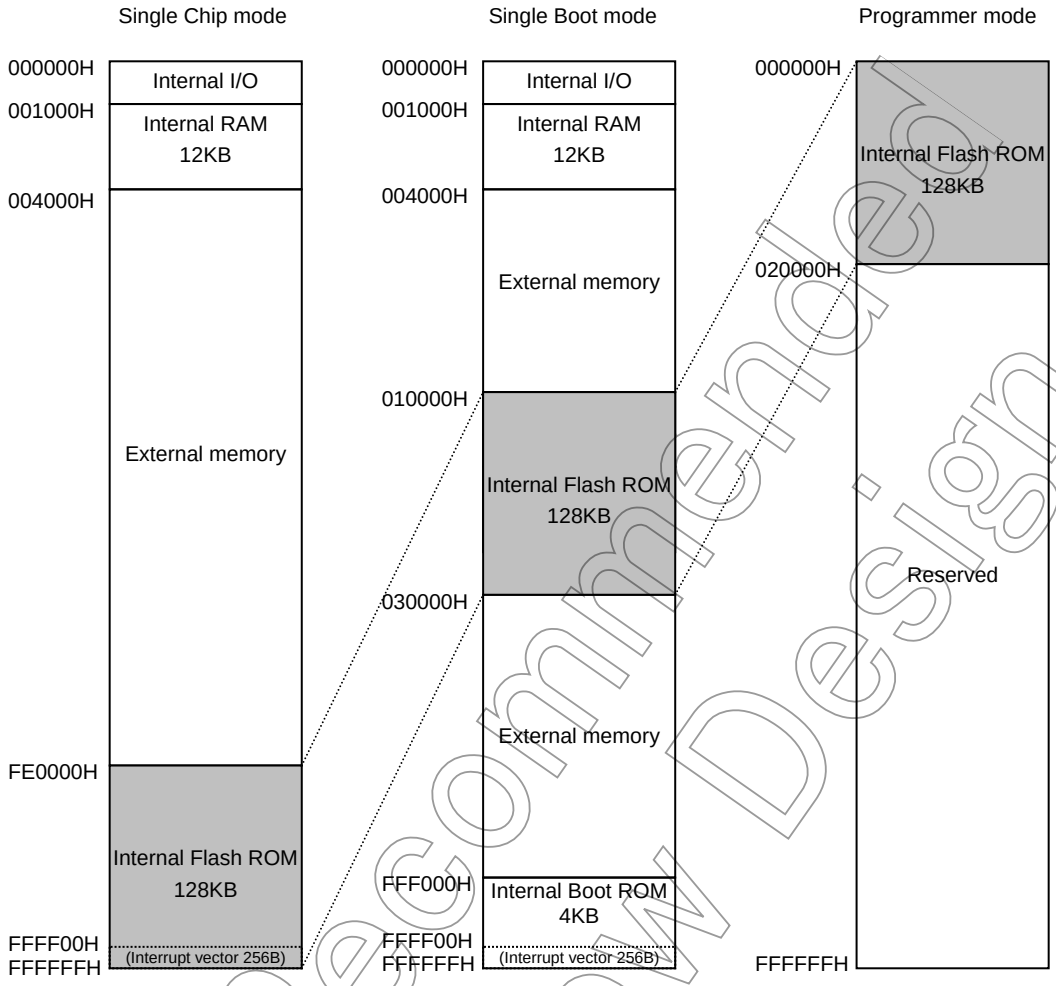


Figure 3.2.3 TMP91FW27 Memory Map for Each Operation Mode

Table 3.2.3 Sector Address Ranges for Each Operation Mode

	Single Chip Mode	Single Boot Mode
Sector-0	FE0000H to FE0FFFH	10000H to 10FFFH
Sector-1	FE1000H to FE1FFFH	11000H to 11FFFH
Sector-2	FE2000H to FE2FFFH	12000H to 12FFFH
Sector-3	FE3000H to FE3FFFH	13000H to 13FFFH
Sector-4	FE4000H to FE4FFFH	14000H to 14FFFH
Sector-5	FE5000H to FE5FFFH	15000H to 15FFFH
Sector-6	FE6000H to FE6FFFH	16000H to 16FFFH
Sector-7	FE7000H to FE7FFFH	17000H to 17FFFH
Sector-8	FE8000H to FE8FFFH	18000H to 18FFFH
Sector-9	FE9000H to FE9FFFH	19000H to 19FFFH
Sector-10	FEA000H to FEAFFFH	1A000H to 1AFFFH
Sector-11	FEB000H to FEBFFFH	1B000H to 1BFFFH
Sector-12	FEC000H to FECFFFH	1C000H to 1CFFFH
Sector-13	FED000H to FEDFFFH	1D000H to 1DFFFH
Sector-14	FEE000H to FEEFFFH	1E000H to 1EFFFH
Sector-15	FEF000H to FEFFFFH	1F000H to 1FFFFH
Sector-16	FF0000H to FF0FFFH	20000H to 20FFFH
Sector-17	FF1000H to FF1FFFH	21000H to 21FFFH
Sector-18	FF2000H to FF2FFFH	22000H to 22FFFH
Sector-19	FF3000H to FF3FFFH	23000H to 23FFFH
Sector-20	FF4000H to FF4FFFH	24000H to 24FFFH
Sector-21	FF5000H to FF5FFFH	25000H to 25FFFH
Sector-22	FF6000H to FF6FFFH	26000H to 26FFFH
Sector-23	FF7000H to FF7FFFH	27000H to 27FFFH
Sector-24	FF8000H to FF8FFFH	28000H to 28FFFH
Sector-25	FF9000H to FF9FFFH	29000H to 29FFFH
Sector-26	FFA000H to FFAFFFH	2A000H to 2AFFFH
Sector-27	FFB000H to FFBFFFH	2B000H to 2BFFFH
Sector-28	FFC000H to FFCFFFH	2C000H to 2CFFFH
Sector-29	FFD000H to FFDFFFH	2D000H to 2DFFFH
Sector-30	FFE000H to FFEFFFH	2E000H to 2EFFFH
Sector-31	FFF000H to FFFFFFH	2F000H to 2FFFFH

3.2.4 Single Boot Mode

In Single Boot mode, the internal boot ROM (mask ROM) is activated to transfer a program/erase routine (user-created boot program) from an external source into the internal RAM. This program/erase routine is then used to program/erase the flash memory. In this mode, the internal boot ROM is mapped into an area containing the interrupt vector table, in which the boot ROM program is executed. The flash memory is mapped into an address space different from the one into which the boot ROM is mapped (see Figure 3.2.3).

The device's SIO (SIO1) and the controller are connected to transfer the program/erase routine from the controller to the device's internal RAM. This program/erase routine is then executed to program/erase the flash memory.

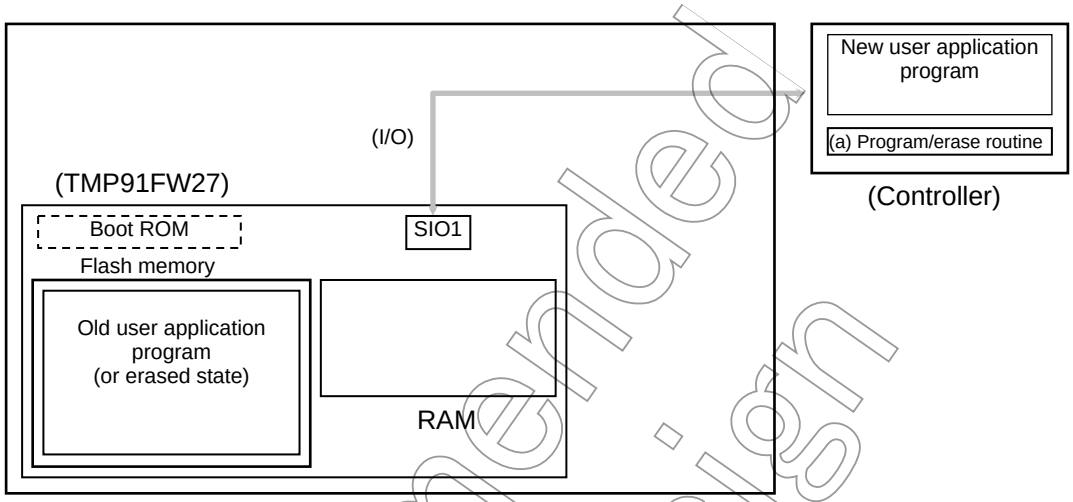
The program/erase routine is executed by sending commands and write data from the controller. The communications protocol between the device and the controller is described later in this manual. Before the program/erase routine can be transferred to the RAM, user password verification is performed to ensure the security of user ROM data. If the password is not verified correctly, the RAM transfer operation cannot be performed. In Single Boot mode, disable interrupts and use the interrupt request flags to check for an interrupt request.

Note: In Single Boot mode, the boot-ROM programs are executed in Normal mode. Do not change to another operation mode in the program/erase routine.

3.2.4.1 Using the program/erase algorithm in the internal boot ROM

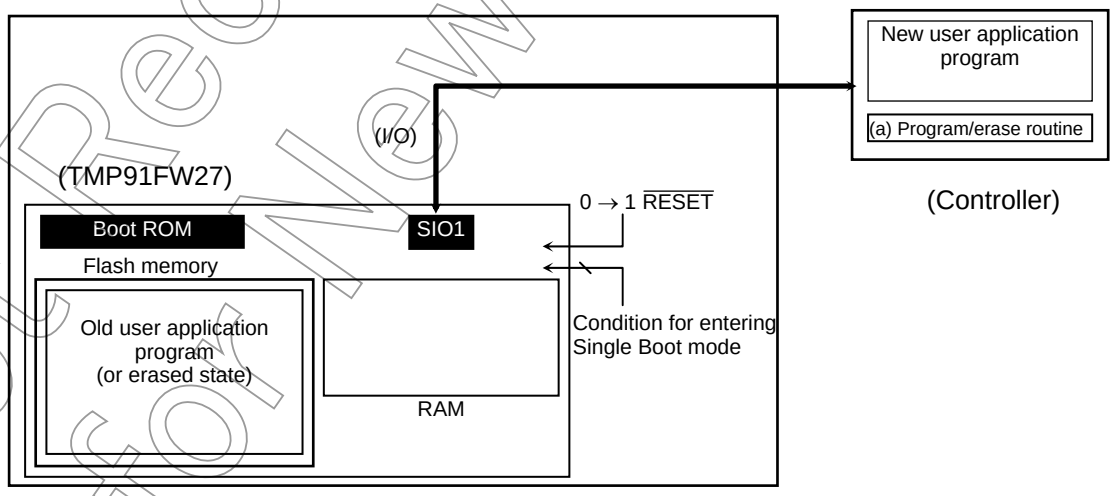
(Step-1) Environment setup

Since the program/erase routine and write data are transferred via SIO (SIO1), connect the device's SIO (SIO1) and the controller on the board. The user must prepare the program/erase routine (a) on the controller.



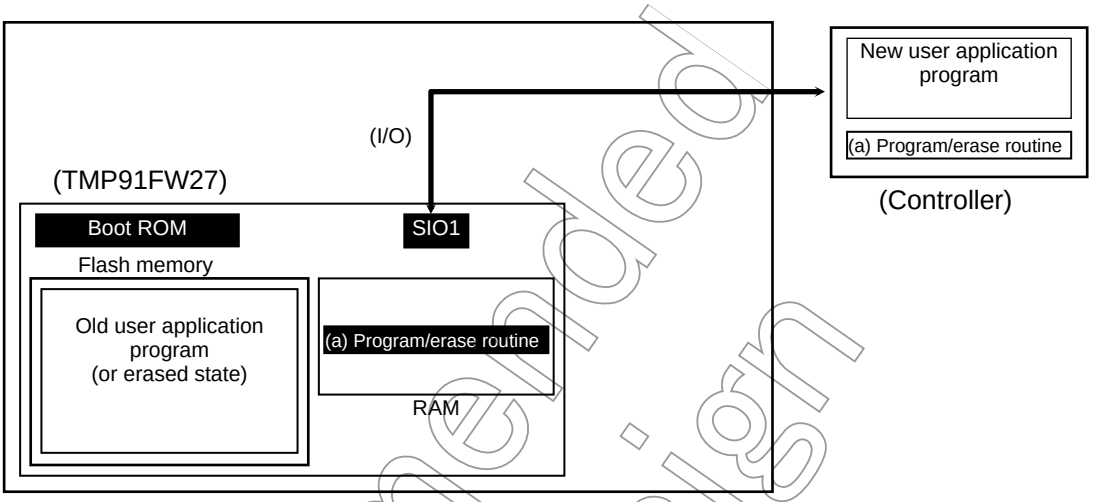
(Step-2) Starting up the internal boot ROM

Release the reset with the relevant input pins set for entering Single Boot mode. When the internal boot ROM starts up, the program/erase routine (a) is transferred from the controller to the internal RAM via SIO according to the communications procedure for Single Boot mode. Before this can be carried out, the password entered by the user is verified against the password written in the user application program. (If the flash memory has been erased, 12 bytes of "0xFF" are used as the password.)



(Step-3) Copying the program/erase routine to the RAM

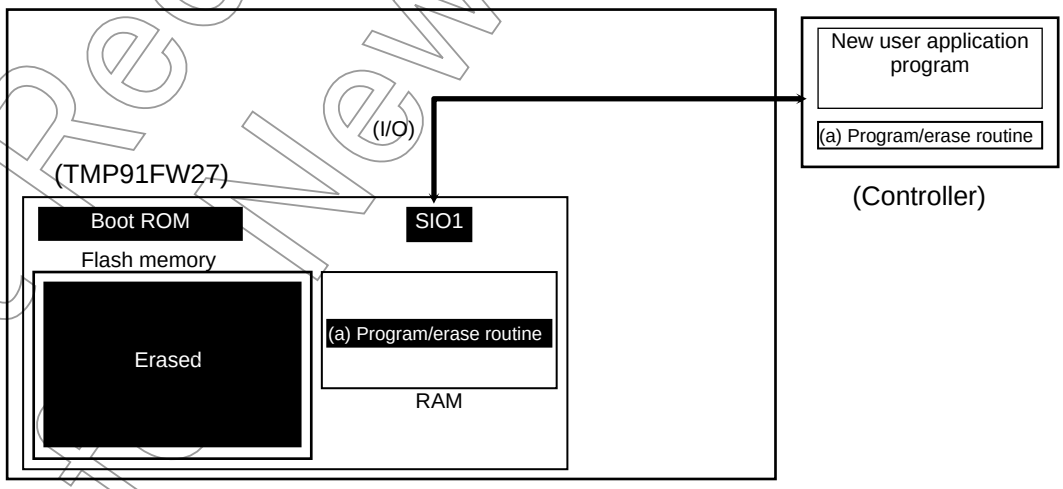
After password verification is completed, the boot ROM copies the program/erase routine (a) from the controller to the RAM using serial communications. The program/erase routine must be stored within the RAM address range of 001000H to 003DFFH.



(Step-4) Executing the program/erase routine in the RAM

Control jumps to the program/erase routine (a) in the RAM. If necessary, the old user application program is erased (sector erase or chip erase).

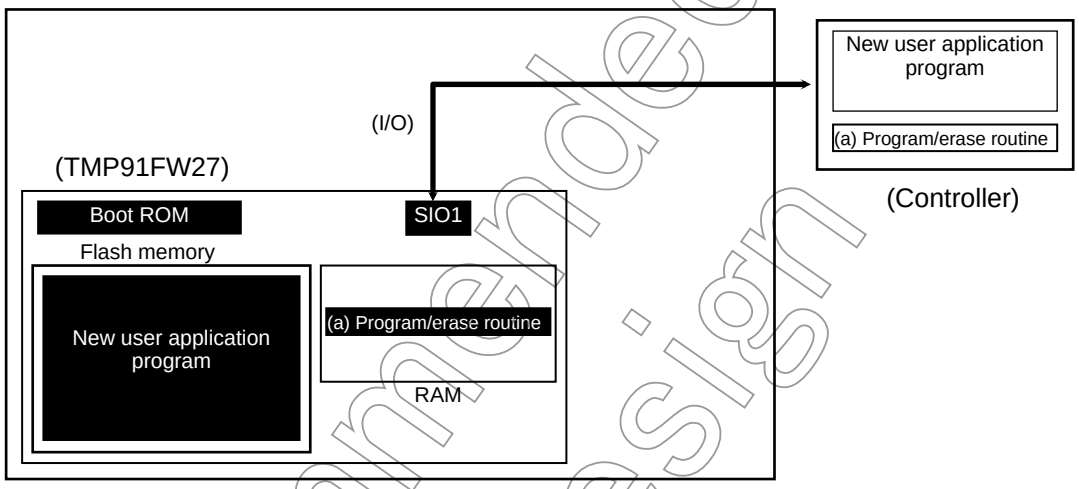
Note: The boot ROM is provided with an erase command, which enables the entire chip to be erased from the controller without using the program/erase routine. If it is necessary to erase data on a sector basis, incorporate the necessary code in the program/erase routine.



(Step-5) Copying the new user application program

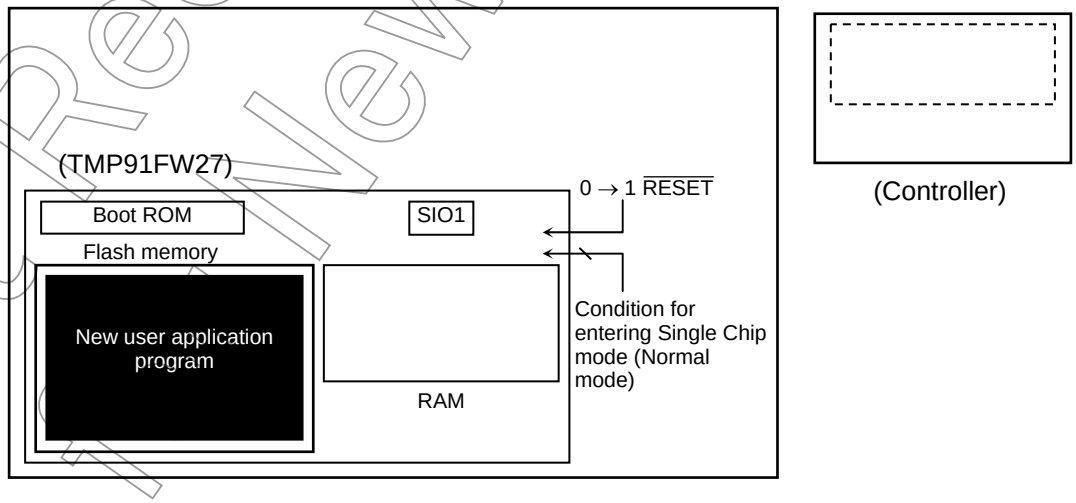
The program/erase routine (a) loads the new user application program from the controller into the erased area of the flash memory.

In the example below, the new user application program is transferred under the same communications conditions as those used for transferring the program/erase routine. However, after the program/erase routine has been transferred, this routine can be used to change the transfer settings (data bus and transfer source). Configure the board hardware and program/erase routine as desired.



(Step-6) Executing the new user application program

After the programming operation has been completed, turn off the power to the board and remove the cable connecting the device and the controller. Then, turn on the power again and start up the device in Single Chip mode to execute the new user application program.



3.2.4.2 Connection Examples for Single Boot Mode

In Single Boot mode the flash memory is programmed by serial transfer. Therefore, on-board programming is performed by connecting the device's SIO (SIO1) and the controller (programming tool) and sending commands from the controller to the device. Figure 3.2.4 shows an example of connection between the target board and a programming controller. Figure 3.2.5 shows an example of connection between the target board and an RS232C board.

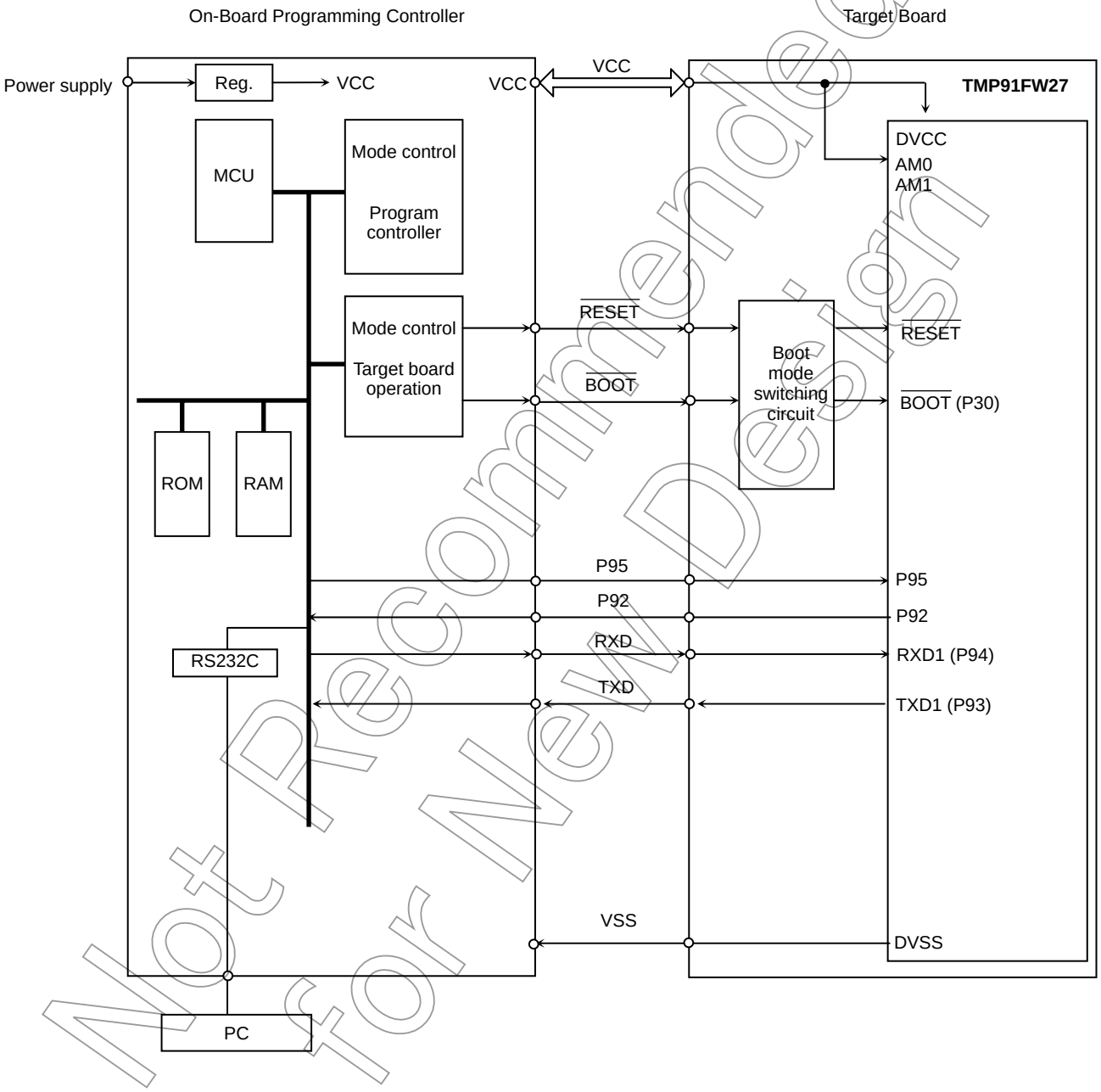


Figure 3.2.4 Example of Connection with an External Controller in Single Boot Mode

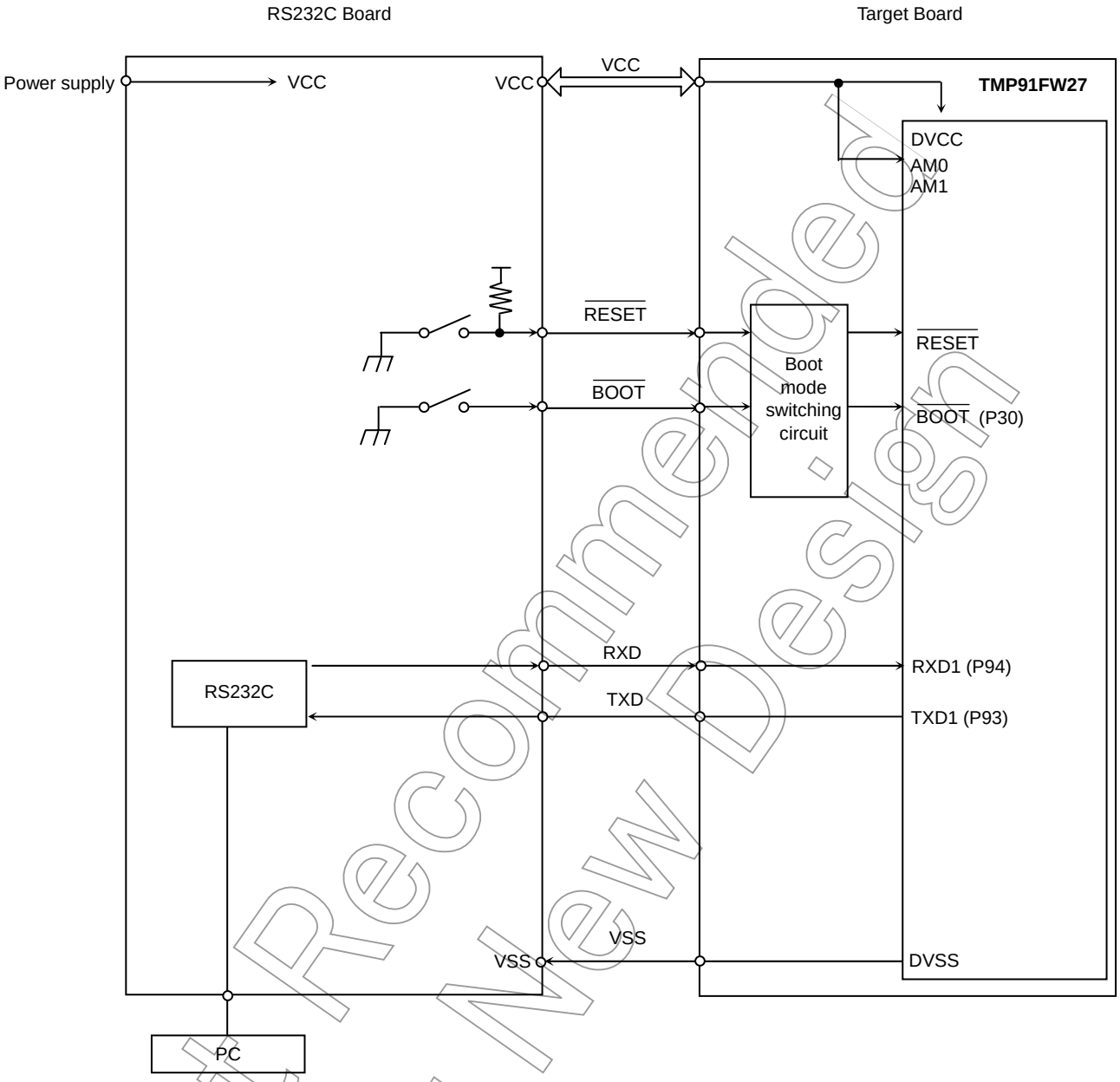


Figure 3.2.5 Example of Connection with an RS232C Board in Single Boot Mode

3.2.4.3 Mode Setting

To perform on-board programming, the device must be started up in Single Boot mode by setting the input pins as shown below.

- AM0,AM1 = 1
- $\overline{\text{BOOT}}$ = 0
- $\overline{\text{RESET}}$ = 0 → 1

Set the AM0, AM1, and $\overline{\text{BOOT}}$ pins as shown above with the $\overline{\text{RESET}}$ pin held at "0". Then, setting the $\overline{\text{RESET}}$ pin to "1" will start up the device in Single Boot mode.

3.2.4.4 Memory Maps

Figure 3.2.6 shows a comparison of the memory map for Normal mode (in Single Chip mode) and the memory map for Single Boot mode. In Single Boot mode, the flash memory is mapped to addresses 10000H to 2FFFFH (physical addresses) and the boot ROM (mask ROM) is mapped to addresses FFF000H to FFFFFFH.

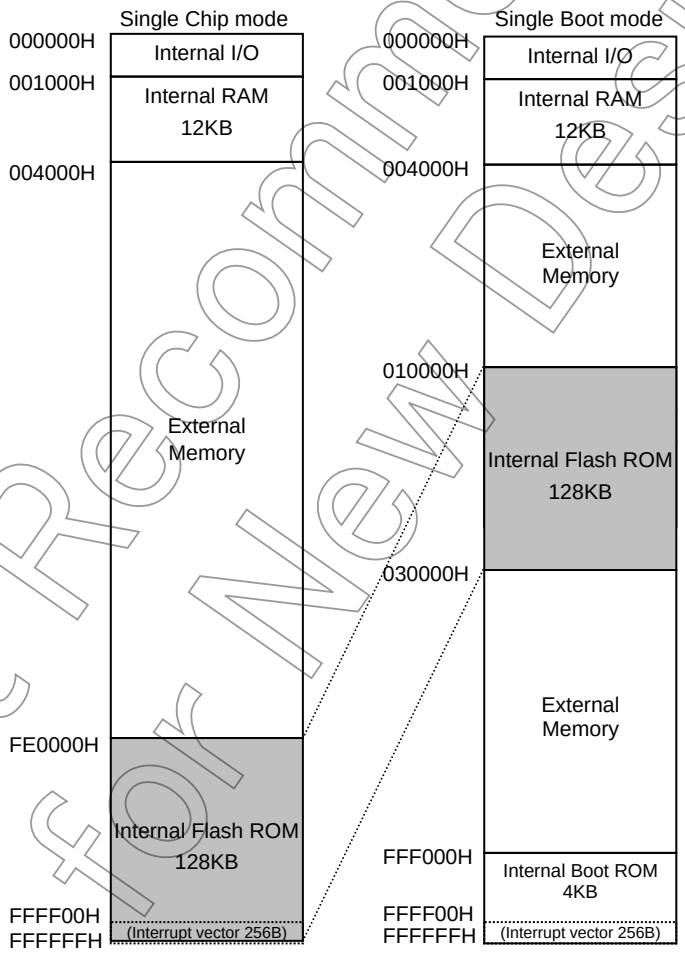


Figure 3.2.6 Comparison of Memory Maps

3.2.4.5 Interface Specifications

The SIO communications format in Single Boot mode is shown below. The device supports the UART (asynchronous communications) serial operation mode.

To perform on-board programming, the same communications format must also be set on the programming controller's side.

● UART (asynchronous) communications

- Communications channel : SIO channel 1 (For the pins to be used, see Table 3.2.4.)
- Serial transfer mode : UART (asynchronous communications) mode
- Data length : 8 bits
- Parity bit : None
- Stop bit : 1 bit
- Baud rate : See Table 3.2.5 and Table 3.2.6.

Table 3.2.4 Pin Connections

Pins		UART
Power supply pins	DVCC	○
	DVSS	○
Mode setting pins	AM1,AM0, $\overline{BOOT}$	○
Reset pin	$\overline{RESET}$	○
Communications pins	TXD1	○
	RXD1	○

Note: Unused pins are in the initial state after reset release.

Table 3.2.5 Baud Rate Table

SIO	Transfer Rate (bps)				
UART	115200	57600	38400	19200	9600

Table 3.2.6 Correspondence between Operating Frequency and Baud Rate in Single Boot Mode

Reference Baud Rate (bps)		9600		19200		38400		57600		115200	
Reference Frequency (MHz)	Supported Range (MHz)	Baud Rate (bps)	Error (%)	(bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)
8	7.83~8.14	9615	+0.16	—	—	—	—	—	—	—	—
10	9.64~10.02	9766	+1.73	19531	+1.73	39063	+1.73	—	—	—	—
11.0592	10.84~11.28	9600	0	19200	0	—	—	—	—	—	—
12.2880	12.05~12.53	9600	0	19200	0	38400	0	—	—	—	—
14.7456	14.46~15.04	9600	0	19200	0	38400	0	57600	0	115200	0
16	15.66~16.29	9615	+0.16	19231	+0.16	—	—	—	—	—	—
18.4320	18.07~18.80	9600	0	19200	0	—	—	57600	0	—	—
20	19.27~20.05	9766	+1.73	19531	+1.73	39063	+1.73	—	—	—	—
22.1184	21.68~22.56	9600	0	19200	0	38400	0	57600	0	—	—
24.5760	24.09~25.06	9600	0	19200	0	38400	0	—	—	—	—
25		9766	+1.73	19531	+1.73	39063	+1.73	—	—	—	—
25.8048	25.29~26.32	9600	0	—	—	—	—	57600	0	—	—
27	26.50~27.57	9588	−0.13	19176	−0.13	38352	−0.13	—	—	—	—

Reference frequency: The frequency of the high-speed oscillation circuit that can be used in Single Boot mode.
To program the flash memory using Single Boot mode, one of the reference frequencies must be selected as a high-speed clock.

Supported Range: The range of clock frequencies that are detected as each reference frequency. It may not be possible to perform Single Boot operations at clock frequencies outside of the supported range.

Note: To automatically detect the reference frequency (microcontroller clock frequency), the transfer baud rate error of the flash memory programming controller and the oscillation frequency error must be within ±2% in total.

3.2.4.6 Data Transfer Formats

Table 3.2.7 to Table 3.2.13 show the operation command data and the data transfer format for each operation mode.

Table 3.2.7 Operation Command Data

Operation Command Data	Operation Mode
10H	RAM Transfer
20H	Flash Memory SUM
30H	Product Information Read
40H	Flash Memory Chip Erase
60H	Flash Memory Protect Set

Table 3.2.8 Transfer Format of Single Boot Program [RAM Transfer]

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
Boot ROM	1st byte	Baud rate setting UART 86H	Desired baud rate (Note 1)	—
	2nd byte	—		ACK response to baud rate setting Normal (baud rate OK) ・UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (10H)		—
	4th byte	—		ACK response to operation command (Note 2) Normal 10H Error x1H Protection applied (Note 4) x6H Communications error x8H
	5th byte to 16th byte	Password data (12 bytes) (02FEF4H to 02FEFFH)		—
	17th byte	CHECKSUM value for 5th to 16th bytes		—
	18th byte	—		ACK response to CHECKSUM value (Note 2) Normal 10H Error 11H Communications error 18H
	19th byte	RAM storage start address 31 to 24 (Note 3)		—
	20th byte	RAM storage start address 23 to 16 (Note 3)		—
	21st byte	RAM storage start address 15 to 8 (Note 3)		—
	22nd byte	RAM storage start address 7 to 0 (Note 3)		—
	23rd byte	RAM storage byte count 15 to 8 (Note 3)		—
	24th byte	RAM storage byte count 7 to 0 (Note 3)		—
	25th byte	CHECKSUM value for 19th to 24th bytes (Note 3)		—
	26th byte	—		ACK response to CHECKSUM value (Note 2) Normal 10H Error 11H Communications error 18H
	27th byte to m'th byte	RAM storage data		—
	(m+1)th byte	CHECKSUM value for 27th to m'th bytes		—
	(m+2)th byte	—		ACK response to CHECKSUM value (Note 2) Normal 10H Error 11H Communications error 18H
RAM	(m+3)th byte	—		Jump to RAM storage start address

Note 1: For the desired baud rate setting, see Table 3.2.6.
Note 2: After sending an error response, the device waits for operation command data (3rd byte).
Note 3: The data to be transferred in the 19th to 25th bytes should be programmed within the RAM address range of 001000H to 003DFFH (11.5Kbytes).
Note 4: When read protection or write protection is applied, the device aborts the received operation command and waits for the next operation command data (3rd byte).

Table 3.2.9 Transfer Format of Single Boot Program [Flash Memory SUM]

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
Boot ROM	1st byte	Baud rate setting UART 86H	Desired baud rate (Note1)	—
	2nd byte	—		ACK response to baud rate setting Normal (baud rate OK) ・UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (20H)		—
	4th byte	—		ACK response to operation command (Note 2) Normal 20H Error x1H Communications error x8H
	5th byte	—		SUM (upper)
	6th byte	—		SUM (lower)
	7th byte	—		CHECKSUM value for 5th and 6th bytes
	8th byte	(Wait for the next operation command data)		—

Note 1: For the desired baud rate setting, see Table 3.2.6.
Note 2: After sending an error response, the device waits for operation command data (3rd byte).

Table 3.2.10 Transfer Format of Single Boot Program [Product Information Read] (1/2)

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
Boot ROM	1st byte	Baud rate setting UART 86H	Desired baud rate (Note 1)	—
	2nd byte	—		ACK response to baud rate setting Normal (baud rate OK) ・UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (30H)		—
	4th byte	—		ACK response to operation command (Note 2) Normal 30H Error x1H Communications error x8H
	5th byte	—		Flash memory data (address 02FEF0H)
	6th byte	—		Flash memory data (address 02FEF1H)
	7th byte	—		Flash memory data (address 02FEF2H)
	8th byte	—		Flash memory data (address 02FEF3H)
	9th byte to 20th byte	—		Part number (ASCII code, 12 bytes) 'TMP91FW27_ _ _ ' (from 9th byte)
	21st byte to 24th byte	—		Password comparison start address (4 bytes) F4H, FEH, 02H, 00H (from 21st byte)
	25th byte to 28th byte	—		RAM start address (4 bytes) 00H, 10H, 00H, 00H (from 25th byte)
	29th byte to 32nd byte	—		RAM (user area) end address (4 bytes) FFH, 3DH, 00H, 00H (from 29th byte)
	33rd byte to 36th byte	—		RAM end address (4 bytes) FFH, 3FH, 00H, 00H (from 33rd byte)
	37th byte to 40th byte	—		Dummy data (4 bytes) 00H,00H,00H,00H (from 37th byte)
	41st byte to 44th byte	—		Dummy data (4 bytes) 00H, 00H, 00H, 00H (from 41st byte)
	45th byte to 46th byte	—		FUSE information (2 bytes from 45th byte) Read protection/Write protection 1) Applied/Applied : 00H, 00H 2) Not applied/Applied : 01H, 00H 3) Applied/Not applied : 02H, 00H 4) Not applied/Not applied : 03H, 00H
	47th byte to 50th byte	—		Flash memory start address (4 bytes) 00H, 00H, 01H, 00H (from 47th byte)
	51st byte to 54th byte	—		Flash memory end address (4 bytes) FFH, FFH, 02H, 00H (from 51st byte)
	55th byte to 56th byte	—		Number of sectors in flash memory (2 bytes) 20H, 00H (from 55th byte)
	57th byte to 60th byte	—		Start address of flash memory sectors of the same size (4 bytes) 00H, 00H, 01H, 00H (from 57th byte)

Table 3.2.11 Transfer Format of Single Boot Program [Product Information Read] (2/2)

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
Boot ROM	61st byte to 64th byte	—		Size (in half words) of flash memory sectors of the same size (4 bytes) 00H, 08H, 00H, 00H (from 61st byte)
	65th byte	—		Number of flash memory sectors of the same size (1 byte) 20H
	66th byte	—		CHECKSUM value for 5th to 65th bytes
	67th byte	(Wait for the next operation command data)		—

Note 1: For the desired baud rate setting, see Table 3.2.6.
Note 2: After sending an error response, the device waits for operation command data (3rd byte).

Table 3.2.12 Transfer Format of Single Boot Program [Flash Memory Chip Erase]

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
Boot ROM	1st byte	Baud rate setting UART 86H	Desired baud rate (Note 1)	—
	2nd byte	—		ACK response to baud rate setting Normal (baud rate OK) ・UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (40H)		—
	4th byte	—		ACK response to operation command (Note2) Normal 40H Error x1H Communications error x8H
	5th byte	Erase Enable command data (54H)		—
	6th byte	—		ACK response to operation command (Note 2) Normal 54H Error x1H Communications error x8H
	7th byte	—		ACK response to Erase command Normal 4FH Error 4CH
	8th byte	—		ACK response Normal 5DH Error 60H
	9th byte	(Wait for the next operation command data)		—

Note 1: For the desired baud rate setting, see Table 3.2.6.
Note 2: After sending an error response, the device waits for operation command data (3rd byte).

Table 3.2.13 Transfer Format of Single Boot Program [Flash Memory Protect Set]

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
Boot ROM	1st byte	Baud rate setting UART 86H	Desired baud rate (Note 1)	—
	2nd byte	—		ACK response to baud rate setting Normal (baud rate OK) ・UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (60H)		—
	4th byte	—		ACK response to operation command (Note2) Normal 60H Error x1H Communications error x8H
	5th byte to 16th byte	Password data (12 bytes) (02FEF4H to 02FEFFH)		—
	17th byte	CHECKSUM value for 5th to 16th bytes		
	18th byte	—		ACK response to checksum value (Note 2) Normal 60H Error 61H Communications error 68H
	19th byte	—		ACK response to Protect Set command Normal 6FH Error 6CH
	20th byte	—		ACK response Normal 31H Error 34H
	21st byte	(Wait for the next operation command data)		—

Note 1: For the desired baud rate setting, see Table 3.2.6.
Note 2: After sending an error response, the device waits for operation command data (3rd byte).

3.2.4.7 Boot Program

When the device starts up in Single Boot mode, the boot program is activated.

The following explains the commands that are used in the boot program to communicate with the controller when the device starts up in Single Boot mode. Use this information for creating a controller for using Single Boot mode or for building a user boot environment.

1. RAM Transfer command

   In RAM transfer, data is transferred from the controller and stored in the device's internal RAM. When the transfer completes normally, the boot program will start running the transferred user program. Up to 11.5 Kbytes of data can be transferred as a user program. (This limit is implemented in the boot program to protect the stack pointer area.) The user program starts executing from the RAM storage start address.
   This RAM transfer function enables a user-created program/erase routine to be executed, allowing the user to implement their own on-board programming method. To perform on-board programming with a user program, the flash memory command sequences (see section 3.2.6) must be used. After the RAM Transfer command has been completed, the entire internal RAM area can be used.
   If read protection or write protection is applied on the device or a password error occurs, this command will not be executed.

2. Flash Memory SUM command

   This command calculates the SUM of 128 Kbytes of data in the flash memory and returns the result. There is no operation command available to the boot program for reading data from the entire area of the flash memory. Instead, this Flash Memory SUM command can be used. Reading the SUM value enables revision management of the application program.

3. Product Information Read command

   This command returns the information about the device including its part number and memory details stored in the flash memory at addresses 02FEF0H to 02FEF3H. This command can also be used for revision management of the application program.

4. Flash Memory Chip Erase command

   This command erases all the sectors in the flash memory. If read protection or write protection is applied on the device, all the sectors in the flash memory are erased and the read protection or write protection is cleared.
   Since this command is also used to restore the operation of the boot program when the password is forgotten, it does not include password verification.

5. Flash Memory Protect Set command

   This command sets both read protection and write protection on the device. However, if a password error occurs, this command will not be executed.
   When read protection is set, the flash memory cannot be read in Programmer mode. When write protection is set, the flash memory cannot be written in Programmer mode.

3.2.4.8 RAM Transfer Command (See Table 3.2.8)

1. From the controller to the device

   The data in the 1st byte is used to determine the baud rate. The 1st byte is transferred with receive operation disabled (SC1MOD0<RXE> = 0). (The baud rate is determined using an internal timer.)

   - To communicate in UART mode

     Send the value 86H from the controller to the target board using UART settings at the desired baud rate. If the serial operation mode is determined as UART, the device checks to see whether or not the desired baud rate can be set. If the device determines that the desired baud rate cannot be set, operation is terminated and no communications can be established.

2. From the device to the controller

   The data in the 2nd byte is the ACK response returned by the device for the serial operation mode setting data sent in the 1st byte. If the data in the 1st byte is found to signify UART and the desired baud rate can be set, the device returns 86H.

   - Baud rate determination

     The device determines whether or not the desired baud rate can be set. If it is found that the baud rate can be set, the boot program rewrites the BR1CR and BR1ADD values and returns 86H. If it is found that the desired baud rate cannot be set, operation is terminated and no data is returned. The controller sets a time-out time (5 seconds) after it has finished sending the 1st byte. If the controller does not receive the response (86H) normally within the time-out time, it should be considered that the device is unable to communicate. Receive operation is enabled (SC1MOD0<RXE> = 1) before 86H is written to the transmission buffer.

3. From the controller to the device

   The data in the 3rd byte is operation command data. In this case, the RAM Transfer command data (10H) is sent from the controller to the device.

4. From the device to the controller

   The data in the 4th byte is the ACK response to the operation command data in the 3rd byte. First, the device checks to see if the received data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined (They are the upper four bits of the immediately preceding operation command data).

   Next, if the data received in the 3rd byte corresponds to one of the operation commands given in Table 3.2.7, the device echoes back the received data (ACK response for normal reception). In the case of the RAM Transfer command, if read or write protection is not applied, 10H is echoed back and then execution branches to the RAM transfer processing routine. If protection is applied, the device returns the corresponding ACK response data (bit 2/1) x6H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

   After branching to the RAM transfer processing routine, the device checks the data in the password area. For details, see 3.2.4.15 “Password”.

If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

5. From the controller to the device

   The 5th to 16th bytes contain password data (12 bytes). The data in the 5th to 16th bytes is verified against the data at addresses 02FEF4H to 02FEFFH in the flash memory, respectively.

6. From the controller to the device

   The 17th byte contains CHECKSUM data. The CHECKSUM data sent by the controller is the two's complement of the lower 8-bit value obtained by summing the data in the 5th to 16th bytes by unsigned 8-bit addition (ignoring any overflow). For details on CHECKSUM, see 3.2.4.17 "How to Calculate CHECKSUM."

7. From the device to the controller

   The data in the 18th byte is the ACK response data to the 5th to 17th bytes (ACK response to the CHECKSUM value). The device first checks to see whether the data received in the 5th to 17th bytes contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) 18H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are the upper four bits of the immediately preceding operation command data, so the value of these bits is "1".
   Next, the device checks the CHECKSUM data in the 17th byte. This check is made to see if the lower 8-bit value obtained by summing the data in the 5th to 17th bytes by unsigned 8-bit addition (ignoring any overflow) is 00H. If the value is not 00H, the device returns the ACK response data for CHECKSUM error (bit 0) 11H and waits for the next operation command data (3rd byte).
   Finally, the device examines the result of password verification. If all the data in the 5th to 16th bytes is not verified correctly, the device returns the ACK response data for password error (bit 0) 11H and waits for the next operation command data (3rd byte).
   If no error is found in all the above checks, the device returns the ACK response data for normal reception 10 H.

8. From the controller to the device

   The data in the 19th to 22nd bytes indicates the RAM start address for storing block transfer data. The 19th byte corresponds to address bits 31 to 24, the 20th byte to address bits 23 to 16, the 21st byte to address bits 15 to 8, and the 22nd byte to address bits 7 to 0.

9. From the controller to the device

   The data in the 23rd and 24th bytes indicates the number of bytes to be transferred. The 23rd byte corresponds to bits 15 to 8 of the transfer byte count and the 24th byte corresponds to bits 7 to 0.

10. From the controller to the device

    The data in the 25th byte is CHECKSUM data. The CHECKSUM data sent by the controller is the two's complement of the lower 8-bit value obtained by summing the data in the 19th to 24th bytes by unsigned 8-bit addition (ignoring any overflow). For details on CHECKSUM, see 3.2.4.17 "How to Calculate CHECKSUM ."

    Note: The data in the 19th to 25th bytes should be placed within addresses 001000H to 003DFFH (11.5 Kbytes) in the internal RAM.

11. From the device to the controller

    The data in the 26th byte is the ACK response data to the data in the 19th to 25th bytes (ACK response to the CHECKSUM value).

    The device first checks to see whether the data received in the 19th to 25th bytes contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) 18H and waits for the next operation command (3rd byte). The upper four bits of the ACK response data are the upper four bits of the immediately preceding operation command data, so the value of these bits is "1".

    Next, the device checks the CHECKSUM data in the 25th byte. This check is made to see if the lower 8-bit value obtained by summing the data in the 19th to 25th bytes by unsigned 8-bit addition (ignoring any overflow) is 00H. If the value is not 00H, the device returns the ACK response data for CHECKSUM error (bit 0) 11H and waits for the next operation command data (3rd byte).

12. From the controller to the device

    The data in the 27th to m'th bytes is the data to be stored in the RAM. This data is written to the RAM starting at the address specified in the 19th to 22nd bytes. The number of bytes to be written is specified in the 23rd and 24th bytes.

13. From the controller to the device

    The data in the (m+1)th byte is CHECKSUM data. The CHECKSUM data sent by the controller is the two's complement of the lower 8-bit value obtained by summing the data in the 27th to m'th bytes by unsigned 8-bit addition (ignoring any overflow). For details on CHECKSUM, see 3.2.4.17 "How to Calculate CHECKSUM."

14. From the device to the controller

    The data in the (m + 2) th byte is the ACK response data to the 27th to (m+1)th bytes (ACK response to the CHECKSUM value).

    The device first checks to see whether the data in the 27th to (m+1)th byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) 18H and waits for the next operation command (3rd byte). The upper four bits of the ACK response are the upper four bits of the immediately preceding operation command data, so the value of these bits is "1".

    Next, the device checks the CHECKSUM data in the (m+1)th byte. This check is made to see if the lower 8-bit value obtained by summing the data in the 27th to (m+1)th bytes by unsigned 8-bit addition (ignoring any overflow) is 00H. If the value is not 00H, the device returns the ACK response data for CHECKSUM error (bit 0) 11H and waits for the next operation command data (3rd byte).

    If no error is found in all the above checks, the device returns the ACK response data for normal reception 10H.

15. From the device to the controller

    If the ACK response data in the (m + 2)th byte is 10H (normal reception), the boot program then jumps to the RAM start address specified in the 19th to 22nd bytes.

3.2.4.9 Flash Memory SUM command (See Table 3.2.9)

1. The data in the 1st and 2nd bytes is the same as in the case of the RAM Transfer command.

2. From the controller to the device

   The data in the 3rd byte is operation command data. The Flash Memory SUM command data (20H) is sent here.

3. From the device to the controller

   The data in the 4th byte is the ACK response data to the operation command data in the 3rd byte.
   The device first checks to see if the data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
   Then, if the data in the 3rd byte corresponds to one of the operation command values given in Table 3.2.7, the device echoes back the received data (ACK response for normal reception). In this case, 20H is echoed back and execution then branches to the flash memory SUM processing routine. If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

4. From the device to the controller

   The data in the 5th and 6th bytes is the upper and lower data of the SUM value, respectively. For details on SUM, see 3.2.4.16 “How to Calculate SUM .”

5. From the device to the controller

   The data in the 7th byte is CHECKSUM data. This is the two’s complement of the lower 8-bit value obtained by summing the data in the 5th and 6th bytes by unsigned 8-bit addition (ignoring any overflow).

6. From the controller to the device

   The data in the 8th byte is the next operation command data.

3.2.4.10 Product Information Read command (See Table 3.2.10 and Table 3.2.11)

1. The data in the 1st and 2nd bytes is the same as in the case of the RAM Transfer command.

2. From the controller to the device

   The data in the 3rd byte is operation command data. The Product Information Read command data (30H) is sent here.

3. From the device to the controller

   The data in the 4th byte is the ACK response data to the operation command data in the 3rd byte.
   The device first checks to see if the data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
   Then, if the data in the 3rd byte corresponds to one of the operation command values given in Table 3.2.7, the device echoes back the received data (ACK response for normal reception). In this case, 30H is returned and execution then branches to the product information read processing routine. If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

4. From the device to the controller

   The data in the 5th to 8th bytes is the data stored at addresses 02FEF0H to 02FEF3H in the flash memory. By writing the ID information of software at these addresses, the version of the software can be managed. (For example, 0002H can indicate that the software is now in version 2.)

5. From the device to the controller

   The data in the 9th to 20th bytes denotes the part number of the device. 'TMP91FW27_ _ _' is sent in ASCII code starting from the 9th byte.
   Note: An underscore ('_') indicates a space.

6. From the device to the controller

   The data in the 21st to 24th bytes is the password comparison start address. F4H, FEH, 02H and 00H are sent starting from the 21st byte.

7. From the device to the controller

   The data in the 25th to 28th bytes is the RAM start address. 00H, 10H, 00H and 00H are sent starting from the 25th byte.

8. From the device to the controller

   The data in the 29th to 32nd bytes is the RAM (user area) end address. FFH, 3DH, 00H and 00H are sent starting from the 29th byte.

9. From the device to the controller

   The data in the 33rd to 36th bytes is the RAM end address. FFH, 3FH, 00H and 00H are sent starting from the 33rd byte.

10. From the device to the controller

    The data in the 37th to 44th bytes is dummy data.

11. From the device to the controller

    The data in the 45th and 46th bytes contains the protection status and sector division information of the flash memory.
    - Bit 0 indicates the read protection status.
      - 0: Read protection is applied.
      - 1: Read protection is not applied.
    - Bit 1 indicates the write protection status.
      - 0: Write protection is applied.
      - 1: Write protection is not applied.
    - Bit 2 indicates whether or not the flash memory is divided into sectors.
      - 0: The flash memory is divided into sectors.
      - 1: The flash memory is not divided into sectors.
    - Bits 3 to 15 are sent as "0".

12. From the device to the controller

    The data in the 47th to 50th bytes is the flash memory start address. 00H, 00H, 01H and 00H are sent starting from the 47th byte.

13. From the device to the controller

    The data in the 51st to 54th bytes is the flash memory end address. FFH, FFH, 02H and 00H are sent starting from the 51st byte.

14. From the device to the controller

    The data in the 55th and 56th bytes indicates the number of sectors in the flash memory. 20H and 00H are sent starting from the 55th byte.

15. From the device to the controller

    The data in the 57th to 65th bytes contains sector information of the flash memory. Sector information is comprised of the start address (starting from the flash memory start address), sector size and number of consecutive sectors of the same size. Note that the sector size is represented in word units.
    The data in the 57th to 65th bytes indicates 4 Kbytes of sectors (sector 0 to sector 31).
    For the data to be transferred, see Table 3.2.10 and Table 3.2.11.

16. From the device to the controller

    The data in the 66th byte is CHECKSUM data. This is the two's complement of the lower 8-bit value obtained by summing the data in the 5th to 65th bytes by unsigned 8-bit addition (ignoring any overflow).

17. From the controller to the device

    The data in the 67th byte is the next operation command data.

3.2.4.11 Flash Memory Chip Erase Command (See Table 3.2.12)

1. The data in the 1st and 2nd bytes is the same as in the case of the RAM Transfer command.

2. From the controller to the device

   The data in the 3rd byte is operation command data. The Flash Memory Chip Erase command data (40H) is sent here.

3. From the device to the controller

   The data in the 4th byte is the ACK response data to the operation command data in the 3rd byte.
   The device first checks to see if the data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
   Then, if the data in the 3rd byte corresponds to one of the operation command values given in Table 3.2.7, the device echoes back the received data (ACK response for normal reception). In this case, 40H is echoed back. If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

4. From the controller to the device

   The data in the 5th byte is Erase Enable command data (54H).

5. From the device to the controller

   The data in the 6th byte is the ACK response data to the Erase Enable command data in the 5th byte.
   The device first checks to see if the data in the 5th byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined (They are the upper four bits of the immediately preceding operation command data.)
   Then, if the data in the 5th byte corresponds to the Erase Enable command data, the device echoes back the received data (ACK response for normal reception). In this case, 54H is echoed back and execution jumps to the flash memory chip erase processing routine. If the data in the 5th byte does not correspond to the Erase Enable command data, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

6. From the device to the controller

   The data in the 7th byte indicates whether or not the erase operation has completed successfully. If the erase operation has completed successfully, the device returns the end code (4FH). If an erase error has occurred, the device returns the error code (4CH).

7. From the device to the controller

   The data in the 8th byte is ACK response data. If the erase operation has completed successfully, the device returns the ACK response for erase completion (5DH). If an erase error has occurred, the device returns the ACK response for erase error (60H).

8. From the controller to the device

   The data in the 9th byte is the next operation command data.

3.2.4.12 Flash Memory Protect Set command (See Table 3.2.13)

1. The data in the 1st and 2nd bytes is the same as in the case of the RAM Transfer command.

2. From the controller to the device

   The data in the 3rd byte is operation command data. The Flash Memory Protect Set command data (60H) is sent here.

3. From the device to the controller

   The data in the 4th byte is the ACK response data to the operation command data in the 3rd byte.
   The device first checks to see if the data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data. The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
   Then, if the data in the 3rd byte corresponds to one of the operation command data values given in Table 3.2.7, the device echoes back the received data (ACK response for normal reception). In this case, 60H is echoed back and execution branches to the flash memory protect set processing routine.
   After branching to this routine, the data in the password area is checked. For details, see 3.2.4.15 "Password."
   If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

4. From the controller to the device

   The data in the 5th to 16th bytes is password data (12 bytes). The data in the 5th byte is verified against the data at address 02FEF4H in the flash memory and the data in the 6th byte against the data at address 02FEF5H. In this manner, the received data is verified consecutively against the data at the specified address in the flash memory. The data in the 16th byte is verified against the data at address 02FEFFH in the flash memory.

5. From the controller to the device

   The data in the 17th byte is CHECKSUM data. The CHECKSUM data sent by the controller is the two's complement of the lower 8-bit value obtained by summing the data in 5th to 16th bytes by unsigned 8-bit addition (ignoring any overflow). For details on CHECKSUM, see 3.2.4.17 "How to Calculate CHECKSUM."

6. From the device to the controller

   The data in the 18th byte is the ACK response data to the data in the 5th to 17th bytes (ACK response to the CHECKSUM value).
   The device first checks to see whether the data in the 5th to 17th bytes contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) 68H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are the upper four bits of the immediately preceding operation command data, so the value of these bits is "6".
   Then, the device checks the CHECKSUM data in the 17th byte. This check is made to see if the lower 8 bits of the value obtained by summing the data in the 5th to 17th bytes by unsigned 8-bit addition (ignoring any overflow) is 00H. If the value is not 00H, the device returns the ACK response data for CHECKSUM error (bit 0) 61H and waits for the next operation command data (3rd byte).
   Finally, the device examines the result of password verification. If all the data in the 5th to 16th bytes is not verified correctly, the device returns the ACK response data for password error (bit 0) 61H and waits for the next operation command data (3rd byte).
   If no error is found in the above checks, the device returns the ACK response data for normal reception 60H.

7. From the device to the controller

   The data in the 19th byte indicates whether or not the protect set operation has completed successfully. If the operation has completed successfully, the device returns the end code (6FH). If an error has occurred, the device returns the error code (6CH).

8. From the device to the controller

   The data in the 20th byte is ACK response data. If the protect set operation has completed successfully, the device returns the ACK response data for normal completion (31H). If an error has occurred, the device returns the ACK response data for error (34H).

9. From the device to the controller

   The data in the 21st byte is the next operation command data.

3.2.4.13 ACK Response Data

The boot program notifies the controller of its processing status by sending various response data. Table 3.2.14 to Table 3.2.19 show the ACK response data returned for each type of received data. The upper four bits of ACK response data are a direct reflection of the upper four bits of the immediately preceding operation command data. Bit 3 indicates a receive error and bit 0 indicates an operation command error, CHECKSUM error or password error.

Table 3.2.14 ACK Response Data to Serial Operation Mode Setting Data

Transfer Data	Meaning
86H	The device can communicate in UART mode. (Note)

Note: If the desired baud rate cannot be set, the device returns no data and terminates operation.

Table 3.2.15 ACK Response Data to Operation Command Data

Transfer Data	Meaning
x8H (Note)	A receive error occurred in the operation command data.
x6H (Note)	Terminated receive operation due to protection setting.
x1H (Note)	Undefined operation command data was received normally.
10H	Received the RAM Transfer command.
20H	Received the Flash Memory SUM command.
30H	Received the Product Information Read command.
40H	Received the Flash Memory Chip Erase command.
60H	Received the Flash Memory Protect Set command.

Note: The upper four bits are a direct reflection of the upper four bits of the immediately preceding operation command data.

Table 3.2.16 ACK Response data to CHECKSUM Data for RAM Transfer Command

Transfer Data	Meaning
18H	A receive error occurred.
11H	A CHECKSUM error or password error occurred.
10H	Received the correct CHECKSUM value.

Table 3.2.17 ACK Response Data to Flash Memory Chip Erase Operation

Transfer Data	Meaning
54H	Received the Erase Enable command.
4FH	Completed erase operation.
4CH	An erase error occurred.
5DH (Note)	Reconfirmation of erase operation
60H (Note)	Reconfirmation of erase error

Note: These codes are returned for reconfirmation of communications.

Table 3.2.18 ACK Response Data to CHECKSUM Data for Flash Memory Protect Set Command

Transfer Data	Meaning
68H	A receive error occurred.
61H	A CHECKSUM or password error occurred.
60H	Received the correct CHECKSUM value.

Table 3.2.19 ACK Response Data to Flash Memory Protect Set Operation

Transfer Data	Meaning
6FH	Completed the protect (read/write) set operation.
6CH	A protect (read/write) set error occurred.
31H (Note)	Reconfirmation of protect (read/write) set operation
34H (Note)	Reconfirmation of protect (read/write) set error

Note: These codes are returned for reconfirmation of communications.

3.2.4.14 Determining Serial Operation Mode

To communicate in UART mode, the controller should transmit the data value 86H as the first byte at the desired baud rate. Figure 3.2.7 shows the waveform of this operation.

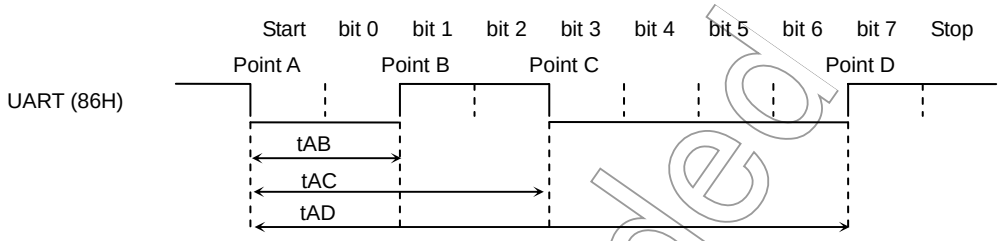


Figure 3.2.7 Data for Determining Serial Operation Mode

The boot program receives the first byte (86H) after reset release not as serial communications data. Instead, the boot program uses the first byte to determine the baud rate. The baud rate is determined by the output periods of tAB, tAC and tAD as shown in Figure 3.2.7 using the procedure shown in Figure 3.2.8.

The CPU monitors the level of the receive pin. Upon detecting a level change, the CPU captures the timer value to determine the baud rate.

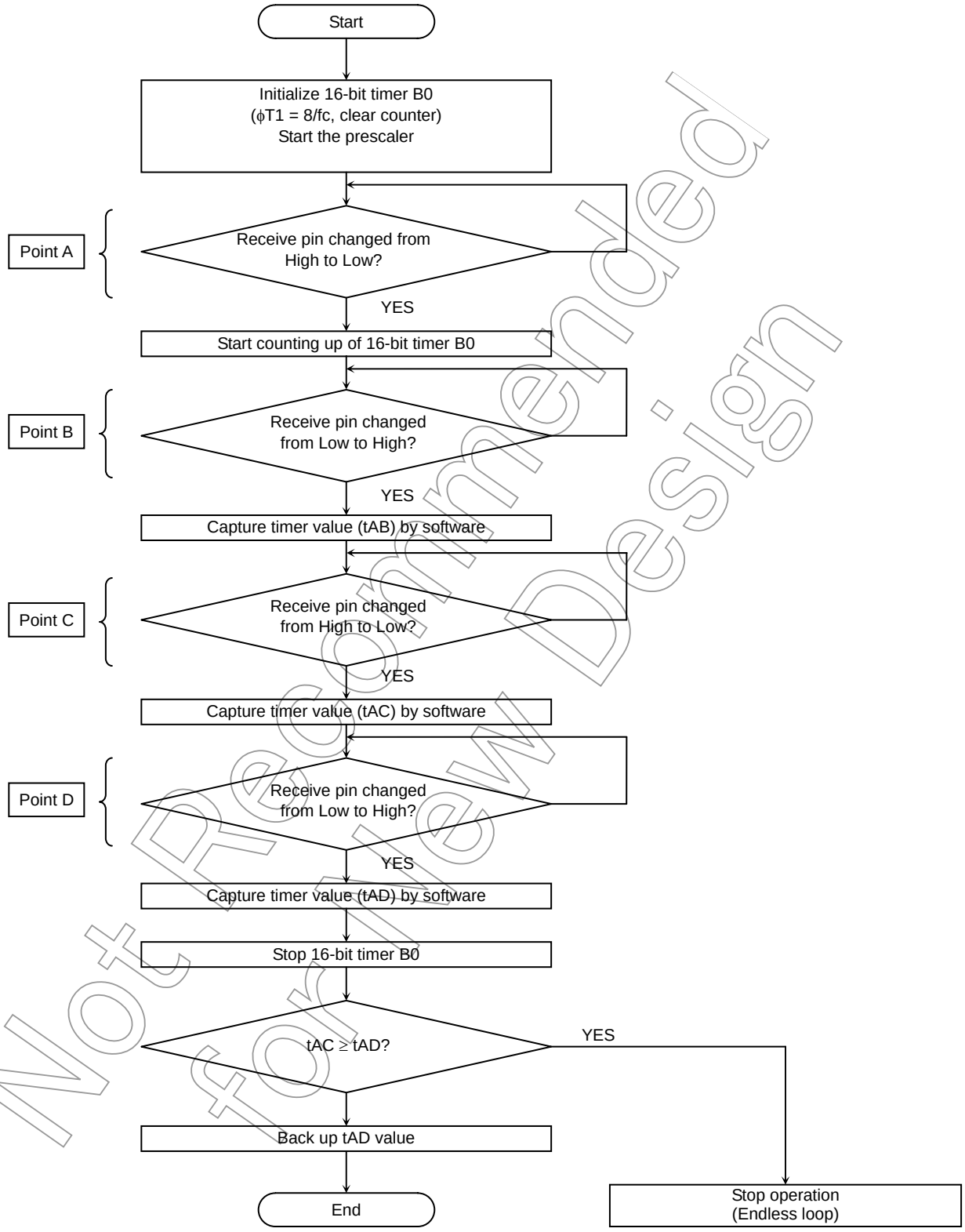


Figure 3.2.8 Flowchart for Serial Operation Mode Receive Operation

3.2.4.15 Password

When the RAM Transfer command (10H) or the Flash Memory Protect Set command (60H) is received as operation command data, password verification is performed. First, the device echoes back the operation command data (10H to 60H) and checks the data (12 bytes) in the password area (addresses 02FEF4H to 02FEFFH).

Then, the device verifies the password data received in the 5th to 16th bytes against the data in the password area as shown in Table 3.2.20.

Unless all the 12 bytes are verified correctly, a password error will occur.

A password error will also occur if all the 12 bytes of password data contain the same value. Only exception is when all the 12 bytes are "FFH" and verified correctly and the reset vector area (addresses 02FF00H to 02FF02H) is all "FFH". In this case, a blank device will be assumed and no password error will occur.

If a password error has occurred, the device returns the ACK response data for password error in the 18th byte.

Table 3.2.20 Password Verification Table

Receive data	Data to be verified against
5th byte	Data at address 02FEF4H
6th byte	Data at address 02FEF5H
7th byte	Data at address 02FEF6H
8th byte	Data at address 02FEF7H
9th byte	Data at address 02FEF8H
10th byte	Data at address 02FEF9H
11th byte	Data at address 02FEFAH
12th byte	Data at address 02FEFBH
13th byte	Data at address 02FEFCH
14th byte	Data at address 02FEFDH
15th byte	Data at address02FEFEH
16th byte	Data at address 02FEFFH

Example of data that cannot be specified as a password

For blank products (Note)

・The password of a blank product must be all "FFH" (FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH).

Note: A blank product is a product in which all the bytes in the password area (addresses 02FEF4H to 02FEFFH) and the reset vector area (addresses 02FF00H to 02FF02H) are "FFH".

For programmed products

・The same 12 consecutive bytes cannot be specified as a password.

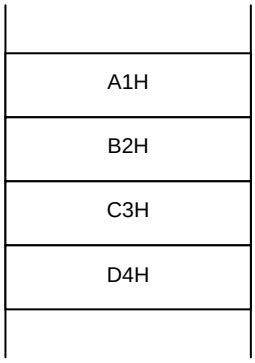
The table below shows password error examples.

Programmed product	1	2	3	4	5	6	7	8	9	10	11	12	Note
Error example 1	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	All "FF"
Error example 2	00H	00H	00H	00H	00H	00H	00H	00H	00H	00H	00H	00H	All "00"
Error example 3	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	All "5A"

3.2.4.16 How to Calculate SUM

SUM is calculated by summing the values of all data read from the flash memory by unsigned 8-bit addition and is returned as a word (16-bit) value. The resulting SUM value is sent to the controller in order of upper 8 bits and lower 8 bits. All the 128 Kbytes of data in the flash memory are included in the calculation of SUM. When the Flash Memory SUM command is executed, SUM is calculated in this way.

Example:



When SUM is calculated from the four data entries shown to the left, the result is as follows:

A1H + B2H + C3H + D4H = 02EAH
SUM upper 8 bits: 02H
SUM lower 8 bits: EAH

Thus, the SUM value is sent to the controller in order of 02H and EAH.

3.2.4.17 How to Calculate CHECKSUM

CHECKSUM is calculated by taking the two's complement of the lower 8-bit value obtained by summing the values of received data by unsigned 8-bit addition (ignoring any overflow). When the Flash Memory SUM command or the Product Information Read command is executed, CHECKSUM is calculated in this way. The controller should also use this CHECKSUM calculation method for sending CHECKSUM values.

Example: Calculating CHECKSUM for the Flash Memory SUM command

When the upper 8-bit data of SUM is E5H and the lower 8-bit data is F6H, CHECKSUM is calculated as shown below.

First, the upper 8 bits and lower 8 bits of the SUM value are added by unsigned operation.

E5H + F6H = 1DBH

Then, the two's complement of the lower 8 bits of this result is obtained as shown below. The resulting CHECKSUM value (25H) is sent to the controller.

0 – DBH = 25H

3.2.5 User Boot Mode (in Single Chip Mode)

User Boot mode, which is a sub mode of Single Chip mode, enables a user-created flash memory program/erase routine to be used. To do so, the operation mode of Single Chip mode must be changed from Normal mode for executing a user application program to User Boot mode for programming/erasing the flash memory.

For example, the reset processing routine of a user application program may include a routine for selecting Normal mode or User Boot mode upon entering Single Chip mode. Any mode-selecting condition may be set using the device's I/O to suit the user system.

To program/erase the flash memory in User Boot mode, a program/erase routine must be incorporated in the user application program in advance. Since the processor cannot read data from the internal flash memory while it is being programmed or erased, the program/erase routine must be executed from the outside of the flash memory. While the flash memory is being programmed/erased in User Boot mode, interrupts must be disabled.

The pages that follow explain the procedure for programming the flash memory using two example cases. In one case the program/erase routine is stored in the internal flash memory (1-A); in the other the program/erase routine is transferred from an external source (1-B).

3.2.5.1 (1-A) Program/Erase Procedure Example 1

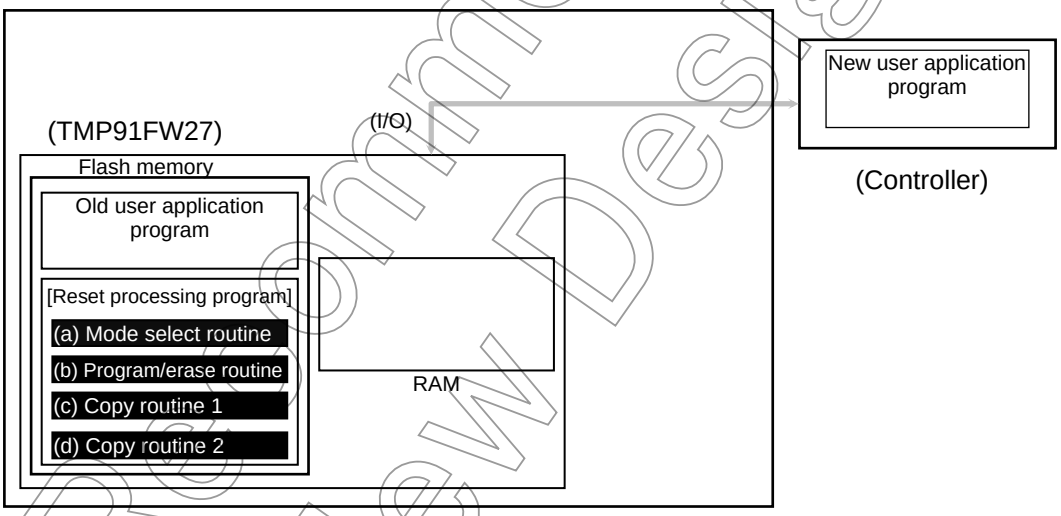
When the program/erase routine is stored in the internal flash memory

(Step-1) Environment setup

First, the condition (e.g. pin status) for entering User Boot mode must be set and the I/O bus for transferring data must be determined. Then, the device's peripheral circuitry must be designed and a corresponding program must be written. Before mounting the device on the board, it is necessary to write the following four routines into one of the sectors in the flash memory.

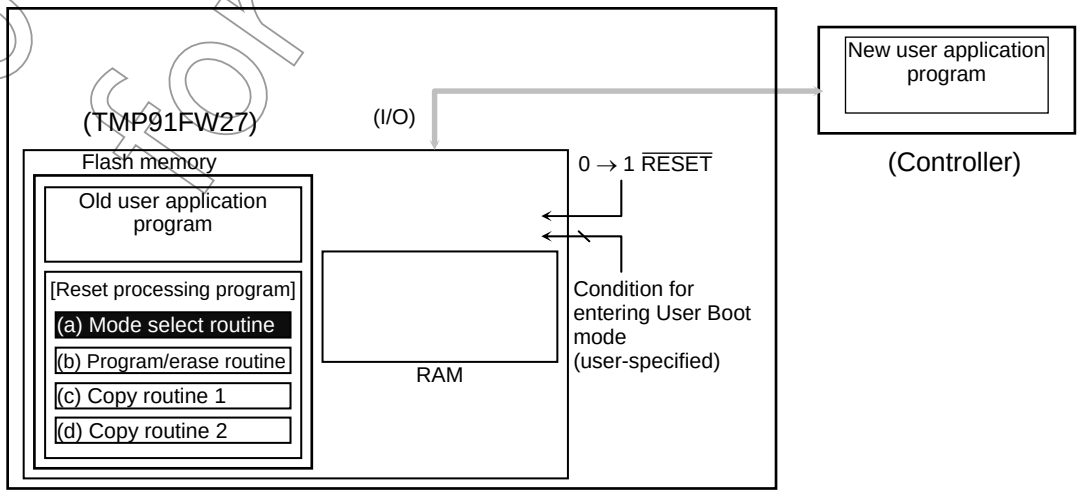
(a) Mode select routine : Selects Normal mode or User Boot mode.
(b) Program/erase routine : Loads program/erase data from an external source and programs/erases the flash memory.
(c) Copy routine 1 : Copies routines (a) to (d) into the internal RAM or external memory.
(d) Copy routine 2 : Copies routines (a) to (d) from the internal RAM or external memory into the flash memory.

Note: The above (d) is a routine for reconstructing the program/erase routine on the flash memory. If the entire flash memory is always programmed and the program/erase routine is included in the new user application program, this copy routine is not needed.



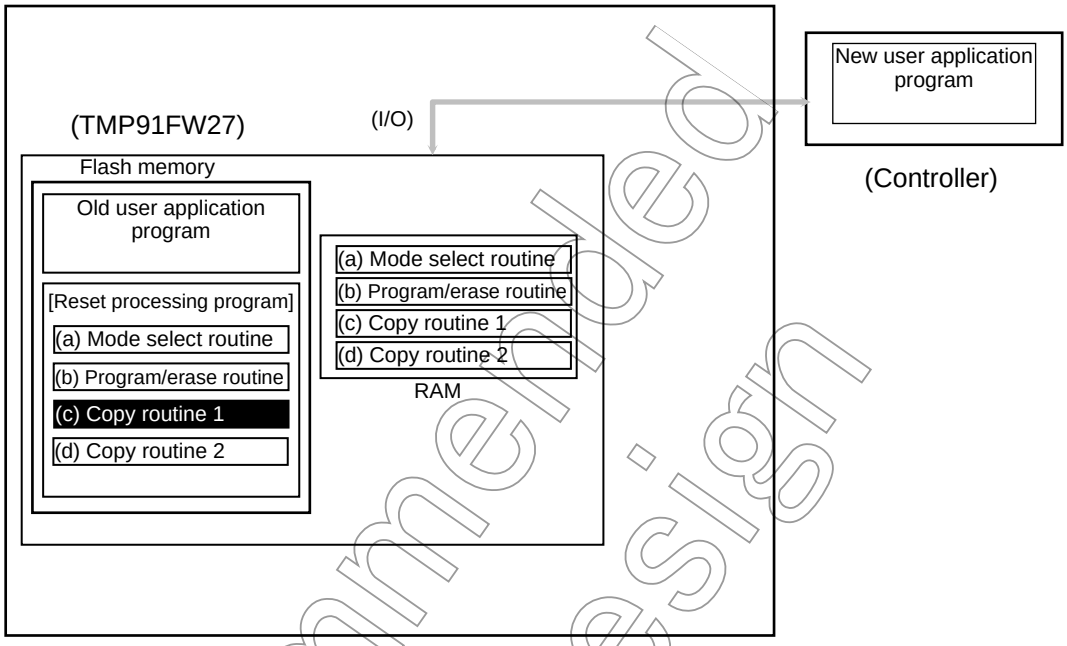
(Step-2) Entering User Boot mode (using the reset processing)

After reset release, the reset processing program determines whether or not the device should enter User Boot mode. If the condition for entering User Boot mode is true, User Boot mode is entered to program/erase the flash memory.



(Step-3) Copying the program/erase routine

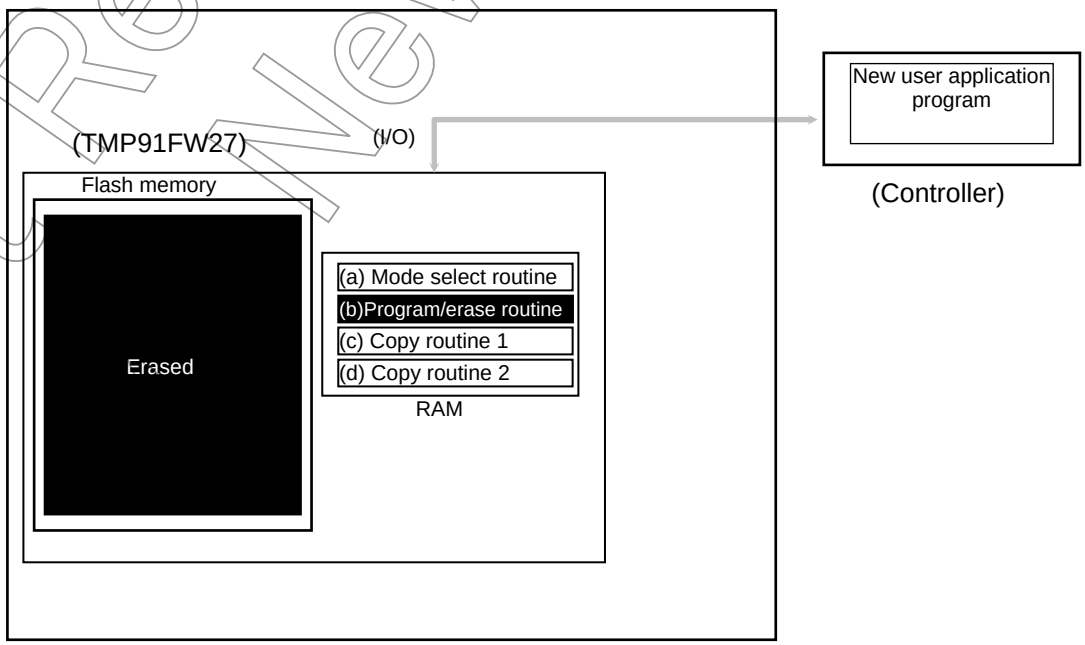
After the device has entered User Boot mode, the copy routine 1 (c) copies the routines (a) to (d) into the internal RAM or external memory (The routines are copied into the internal RAM here.)



(Step-4) Erasing the flash memory by the program/erase routine

Control jumps to the program/erase routine in the RAM and the old user program area is erased (sector erase or chip erase). (In this case, the flash memory erase command is issued from the RAM.)

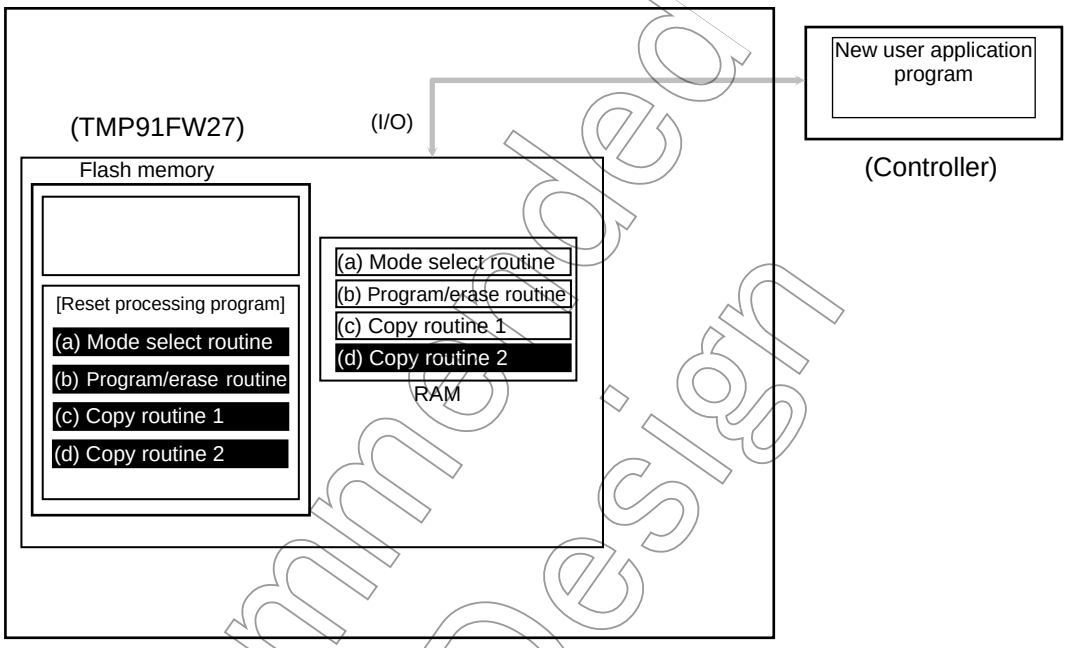
Note: If data is erased on a sector basis and the routines (a) to (d) are left in the flash memory, only the program/erase routine (b) need be copied into the RAM.



(Step-5) Restoring the user boot program in the flash memory

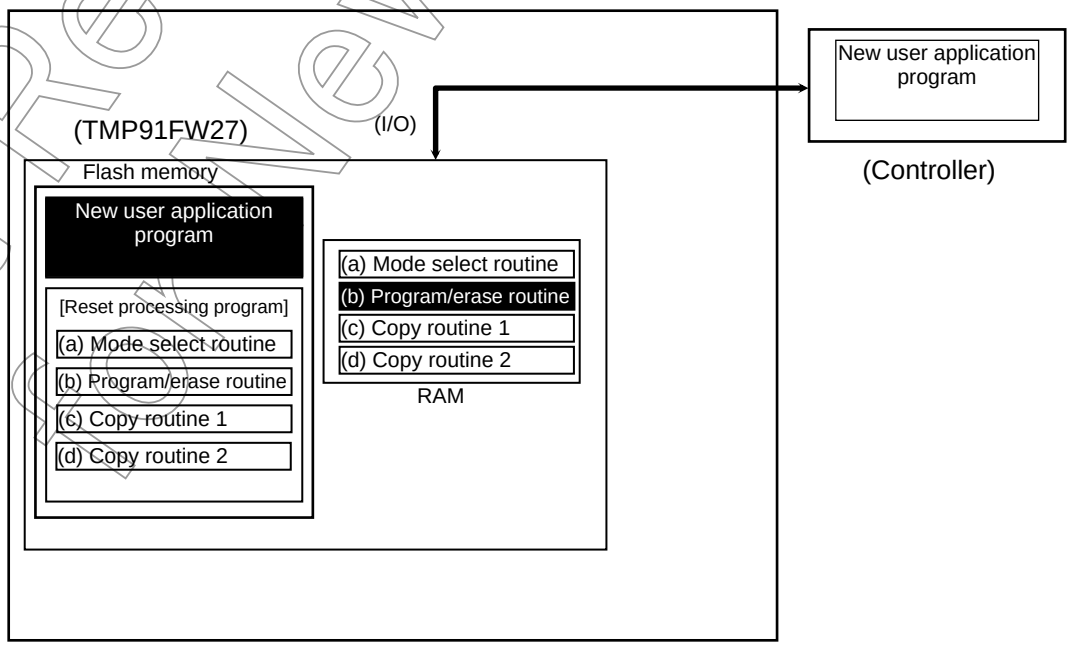
The copy routine 2 (d) in the RAM copies the routines (a) to (d) into the flash memory.

Note: If data is erased on a sector basis and the routines (a) to (d) are left in the flash memory, step 5 is not needed.



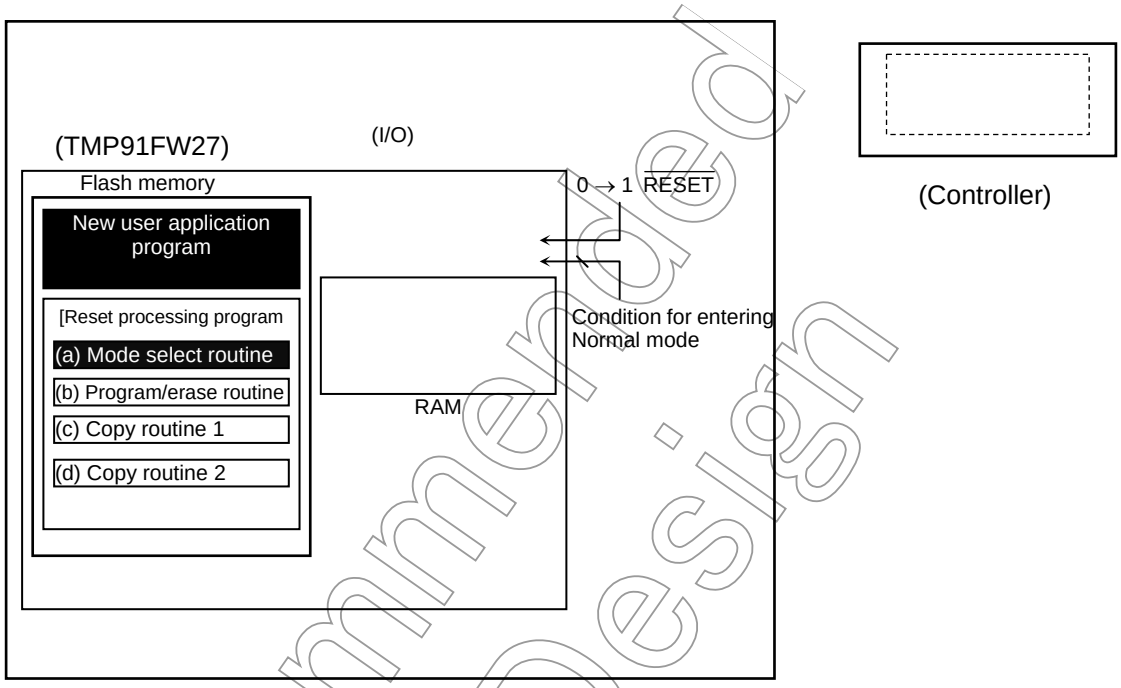
(Step-6) Writing the new user application program to the flash memory

The program/erase routine in the RAM is executed to load the new user application program from the controller into the erased area of the flash memory.



(Step-7) Executing the new user application program

The $\overline{\text{RESET}}$ input pin is driven Low ("0") to reset the device. The mode setting condition is set for Normal mode. After reset release, the device will start executing the new user application program.



3.2.5.2 (1-B) Program/Erase Procedure Example 2

In this example, only the boot program (minimum requirement) is stored in the flash memory and other necessary routines are supplied from the controller.

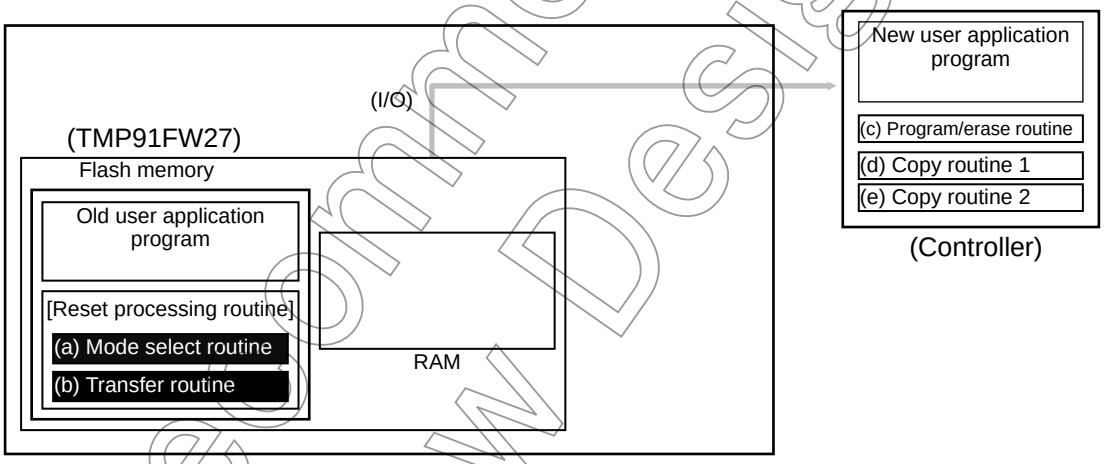
(Step-1) Environment setup

First, the condition (e.g. pin status) for entering User Boot mode must be set and the I/O bus for transferring data must be determined. Then, the device's peripheral circuitry must be designed and a corresponding program must be written. Before mounting the device on the board, it is necessary to write the following two routines into one on the sectors in the flash memory.

(a) Mode select routine : Selects Normal mode or User Boot mode.
(b) Transfer routine : Loads the program/erase routine from an external source.

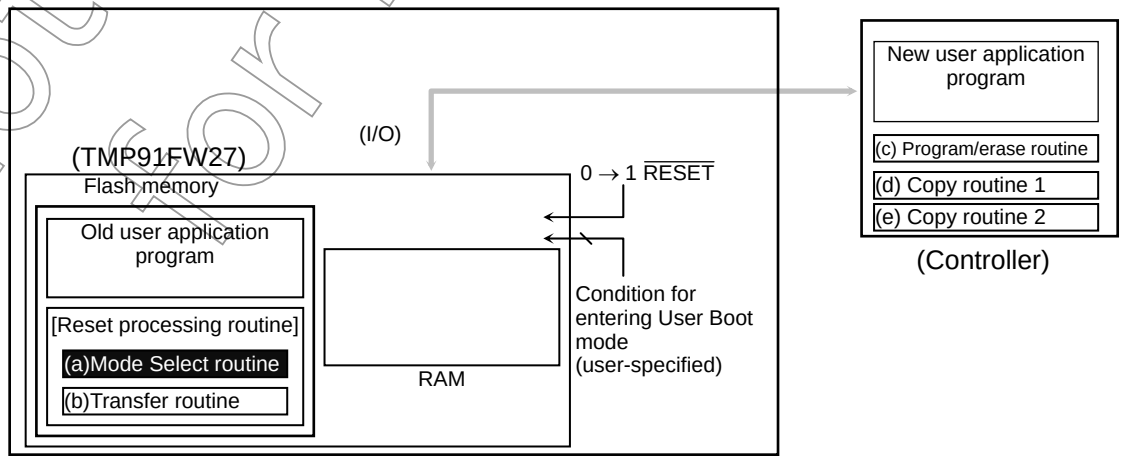
The following routines are prepared on the controller.

(c) Program/erase routine : Programs/erases the flash memory.
(d) Copy routine 1 : Copies routines (a) and (b) into the internal RAM or external memory.
(e) Copy routine 2 : Copies routines (a) and (b) from the internal RAM or external memory into the flash memory.



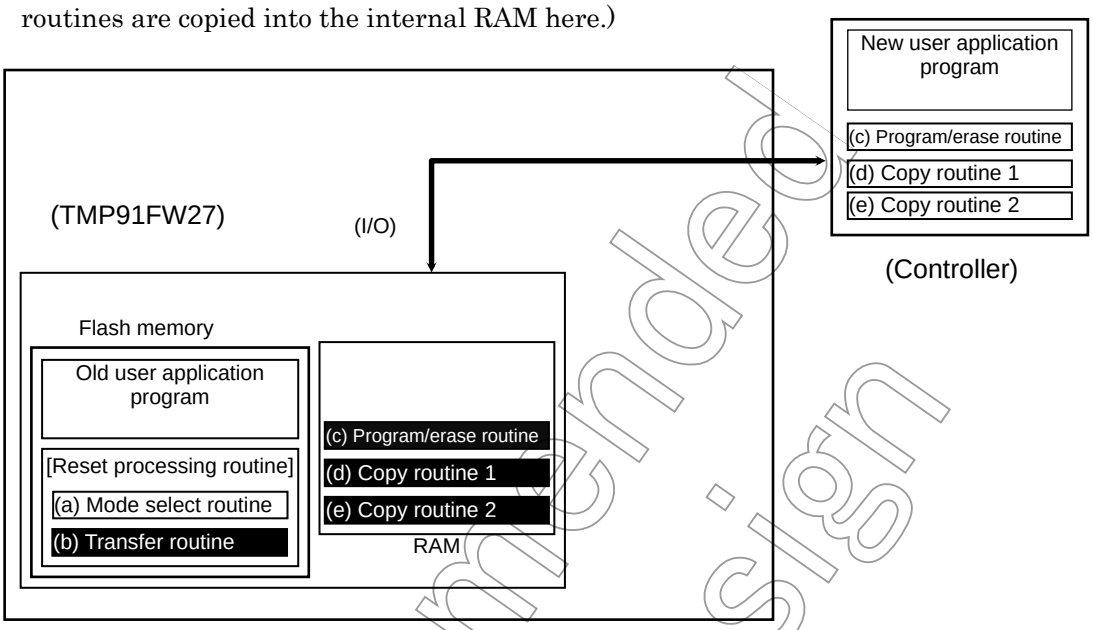
(Step-2) Entering User Boot mode (using the reset processing)

The following explanation assumes that these routines are incorporated in the reset processing program. After reset release, the reset processing program first determines whether or not the device should enter User Boot mode. If the condition for entering User Boot mode is true, User Boot mode is entered to program/erase the flash memory.



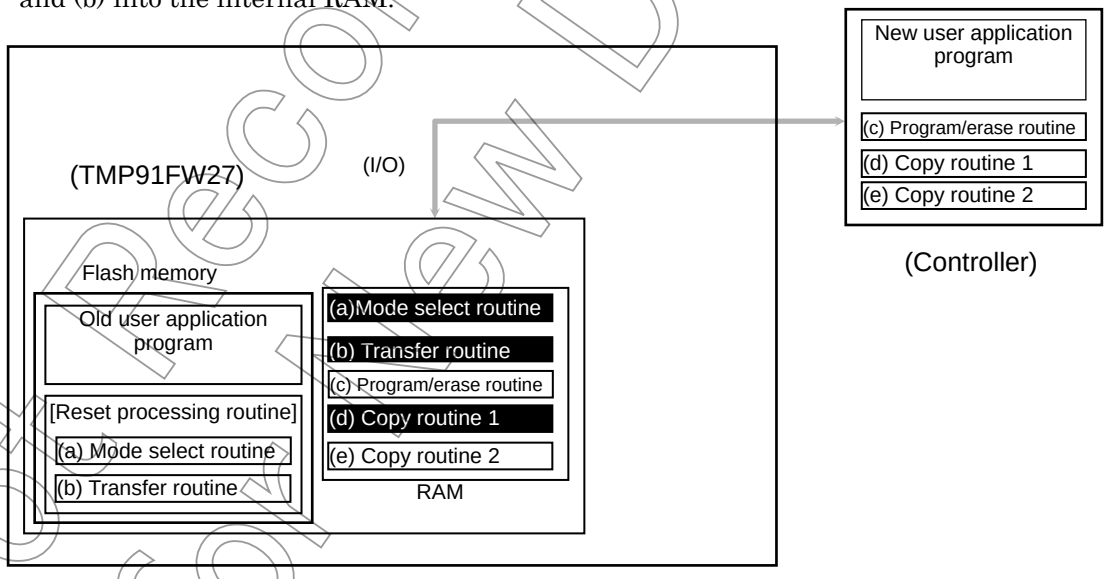
(Step-3) Copying the program/erase routine to the internal RAM

After the device has entered User Boot mode, the transfer routine (b) transfers the routines (c) to (e) from the controller to the internal RAM (or external memory). (The routines are copied into the internal RAM here.)



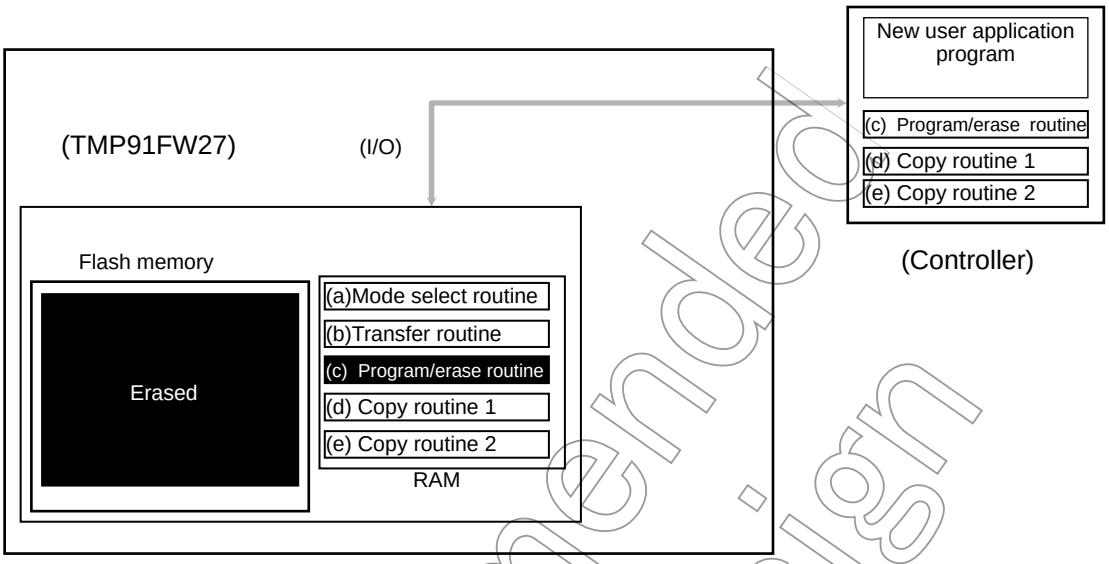
(Step-4) Executing the copy routine 1 in the internal RAM

Control jumps to the internal RAM and the copy routine 1 (d) copies the routines (a) and (b) into the internal RAM.



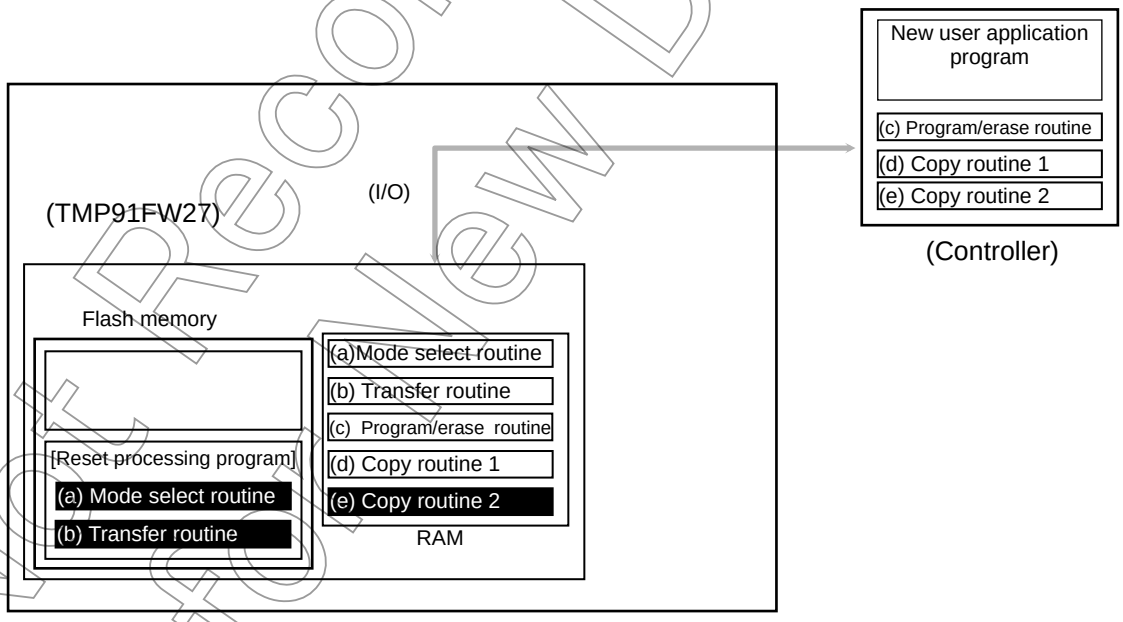
(Step-5) Erasing the flash memory by the program/erase routine

The program/erase routine (c) erases the old user program area.



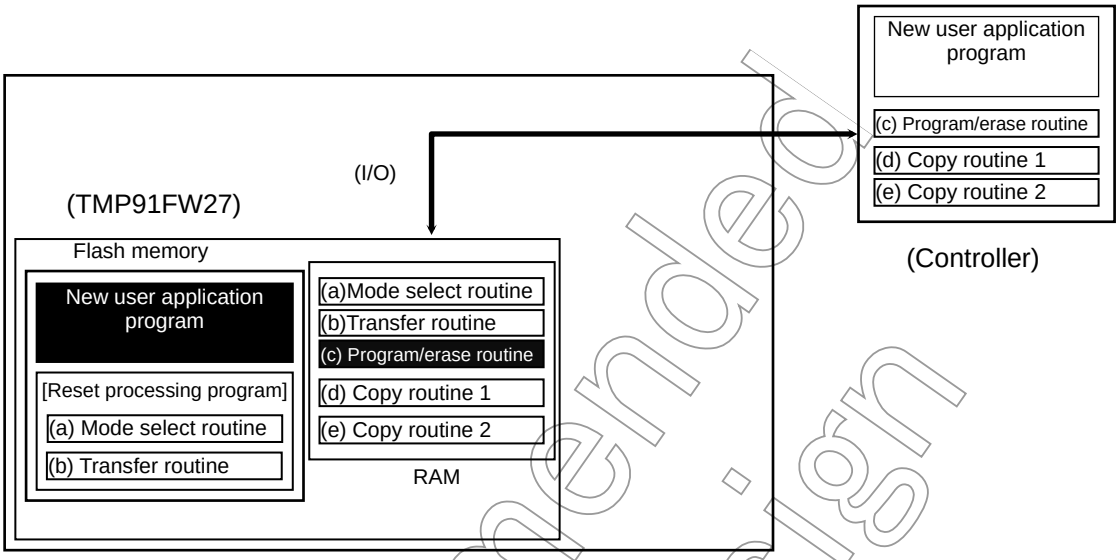
(Step-6) Restoring the user boot program in the flash memory

The copy routine (e) copies the routines (a) and (b) from the internal RAM into the flash memory.



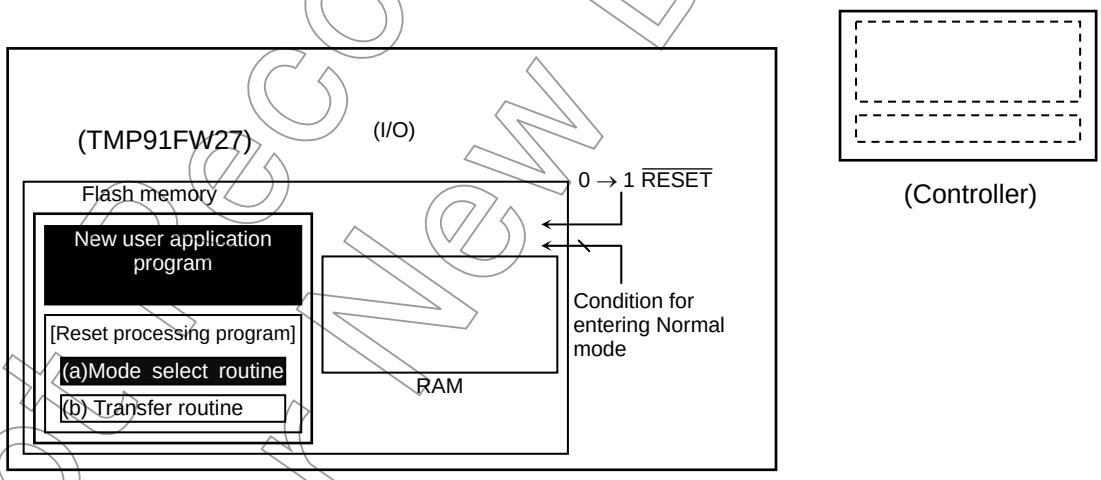
(Step-7) Writing the new user application program to the flash memory

The program/erase routine (c) in the RAM is executed to load the new user application program from the controller into the erased area of the flash memory.



(Step-8) Executing the new user application program

The $\overline{\text{RESET}}$ input pin is driven Low ("0") to reset the device. The mode setting condition is set for Normal mode. After reset release, the device will start executing the new user application program.



3.2.6 Flash Memory Command Sequences

The operation of the flash memory is comprised of six commands, as shown in Table 3.2.21. Addresses specified in each command sequence must be in an area where the flash memory is mapped. For details, see Table 3.2.3.

Table 3.2.21 Command Sequences

	Command Sequence	1st Bus Write Cycle		2nd Bus Write Cycle		3rd Bus Write Cycle		4th Bus Write Cycle		5th Bus Write Cycle		6th Bus Write Cycle	
		Addr.	Data	Addr.	Data	Addr.	Data	Addr.	Data	Addr.	Data	Addr.	Data
1	Single Word Program	AAAH	AAH	554H	55H	AAAH	A0H	PA (Note 1)	PD (Note 1)				
2	Sector Erase (4-KB Erase)	AAAH	AAH	554H	55H	AAAH	80H	AAAH	AAH	554H	55H	SA (Note 2)	30H
3	Chip Erase (All Erase)	AAAH	AAH	554H	55H	AAAH	80H	AAAH	AAH	554H	55H	AAAH	10H
4	Product ID Entry	AAAH	AAH	554H	55H	AAAH	90H						
5	Product ID Exit	xxH	F0H										
	Product ID Exit	AAAH	AAH	554H	55H	AAAH	F0H						
6	Read Protect Set	AAAH	AAH	554H	55H	AAAH	A5H	77EH	F0H (Note3)				
	Write Protect Set	AAAH	AAH	554H	55H	AAAH	A5H	77EH	0FH (Note3)				

Note 1: PA = Program Word address, PD = Program Word data
Set the address and data to be programmed. Even-numbered addresses should be specified here.

Note 2: SA = Sector Erase address, Each sector erase range is selected by address A23 to A12.

Note 3: When apply read protect and write protect, be sure to program the data of 00H.

Table 3.2.22 Hardware Sequence Flags

Status		D7	D6
During auto operation	Single Word Program	$\overline{D7}$	Toggle
	Sector Erase/Chip Erase	0	Toggle
	Read Protect Set/Write Protect Set	Cannot be used	Toggle

Note: D15 to D8 and D5 to D0 are “don’t care”.

3.2.6.1 Single Word Program

The Single Word Program command sequence programs the flash memory on a word basis. The address and data to be programmed are specified in the 4th bus write cycle. It takes a maximum of 60 μs to program a single word. Another command sequence cannot be executed until the write operation has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data is read consecutively. While a write operation is in progress, bit 6 of data is toggled each time it is read.

Note: To rewrite data to Flash memory addresses at which data (including FFFFH) is already written, make sure to erase the existing data by "sector erase" or "chip erase" before rewriting data.

3.2.6.2 Sector Erase (4-Kbyte Erase)

The Sector Erase command sequence erases 4 Kbytes of data in the flash memory at a time. The flash memory address range to be erased is specified in the 6th bus write cycle. For the address range of each sector, see Table 3.2.3. This command sequence cannot be used in Programmer mode.

It takes a maximum of 75 ms to erase 4 Kbytes. Another command sequence cannot be executed until the erase operation has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data is read consecutively. While a erase operation is in progress, bit 6 of data is toggled each time it is read.

3.2.6.3 Chip Erase (All Erase)

The Chip Erase command sequence erases the entire area of the flash memory.

It takes a maximum of 300 ms to erase the entire flash memory. Another command sequence cannot be executed until the erase operation has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data is read consecutively. While an erase operation is in progress, bit 6 of data is toggled each time it is read.

Erase operations clear data to FFH.

3.2.6.4 Product ID Entry

When the Product ID Entry command is executed, Product ID mode is entered. In this mode, the vendor ID, flash macro ID, flash size ID, and read/write protect status can be read from the flash memory. In Product ID mode, the data in the flash memory cannot be read.

3.2.6.5 Product ID Exit

This command sequence is used to exit Product ID mode.

3.2.6.6 Read Protect Set

The Read Protect Set command sequence applies read protection on the flash memory. When read protection is applied, the flash memory cannot be read in Programmer mode and the RAM Transfer command cannot be executed in Single Boot mode.

To cancel read protection, it is necessary to execute the Chip Erase command sequence. To check whether or not read protection is applied, read xxx77EH in Product ID mode. It takes a maximum of 60 μs to set read protection on the flash memory. Another command sequence cannot be executed until the read protection setting has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data can be read consecutively. While a read protect operation is in progress, bit 6 of data is toggled each time it is read.

3.2.6.7 Write Protect Set

The Write Protect Set command sequence applies write protection on the flash memory. When write protection is applied, the flash memory cannot be written to in Programmer mode and the RAM Transfer command cannot be executed in Single Boot mode.

To cancel write protection, it is necessary to execute the Chip Erase command sequence. To check whether or not write protection is applied, read xxx77EH in Product ID mode. It takes a maximum of 60 μs to set write protection. Another command sequence cannot be executed until the write protection setting has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data can be read consecutively. While a write protect operation is in progress, bit 6 of data is toggled each time it is read.

3.2.6.8 Hardware Sequence Flags

The following hardware sequence flags are available to check the auto operation execution status of the flash memory.

1) Data polling (D7)

When data is written to the flash memory, D7 outputs the complement of its programmed data until the write operation has completed. After the write operation has completed, D7 outputs the proper cell data. By reading D7, therefore, the operation status can be checked. While the Sector Erase or Chip Erase command sequence is being executed, D7 outputs "0". After the command sequence is completed, D7 outputs "1" (cell data). Then, the data written to all the bits can be read after waiting for 1 μs.

When read/write protection is applied, the data polling function cannot be used. Instead, use the toggle bit (D6) to check the operation status.

2) Toggle bit (D6)

When the Flash Memory Program, Sector Erase, Chip Erase, Write Protect Set, or Read Protect Set command sequence is executed, bit 6 (D6) of the data read by read operations outputs "0" and "1" alternately each time it is read until the processing of the executed command sequence has completed. The toggle bit (D6) thus provides a software means of checking whether or not the processing of each command sequence has completed. Normally, the same address in the flash memory is read repeatedly until the same data is read successively. The initial read of the toggle bit always returns "1".

Note: The flash memory incorporated in the TMP91FW27 does not have an exceed-time-limit bit (D5). It is therefore necessary to set the data polling time limit and toggle bit polling time limit so that polling can be stopped if the time limit is exceeded.

3.2.6.9 Data Read

Data is read from the flash memory in byte units or word units. It is not necessary to execute a command sequence to read data from the flash memory.

3.2.6.10 Programming the Flash Memory by the Internal CPU

The internal CPU programs the flash memory by using the command sequences and hardware sequence flags described above. However, since the flash memory cannot be read during auto operation mode, the program/erase routine must be executed outside of the flash memory.

The CPU can program the flash memory either by using Single Boot mode or by using a user-created protocol in Single Chip mode (User Boot).

1) Single Boot:

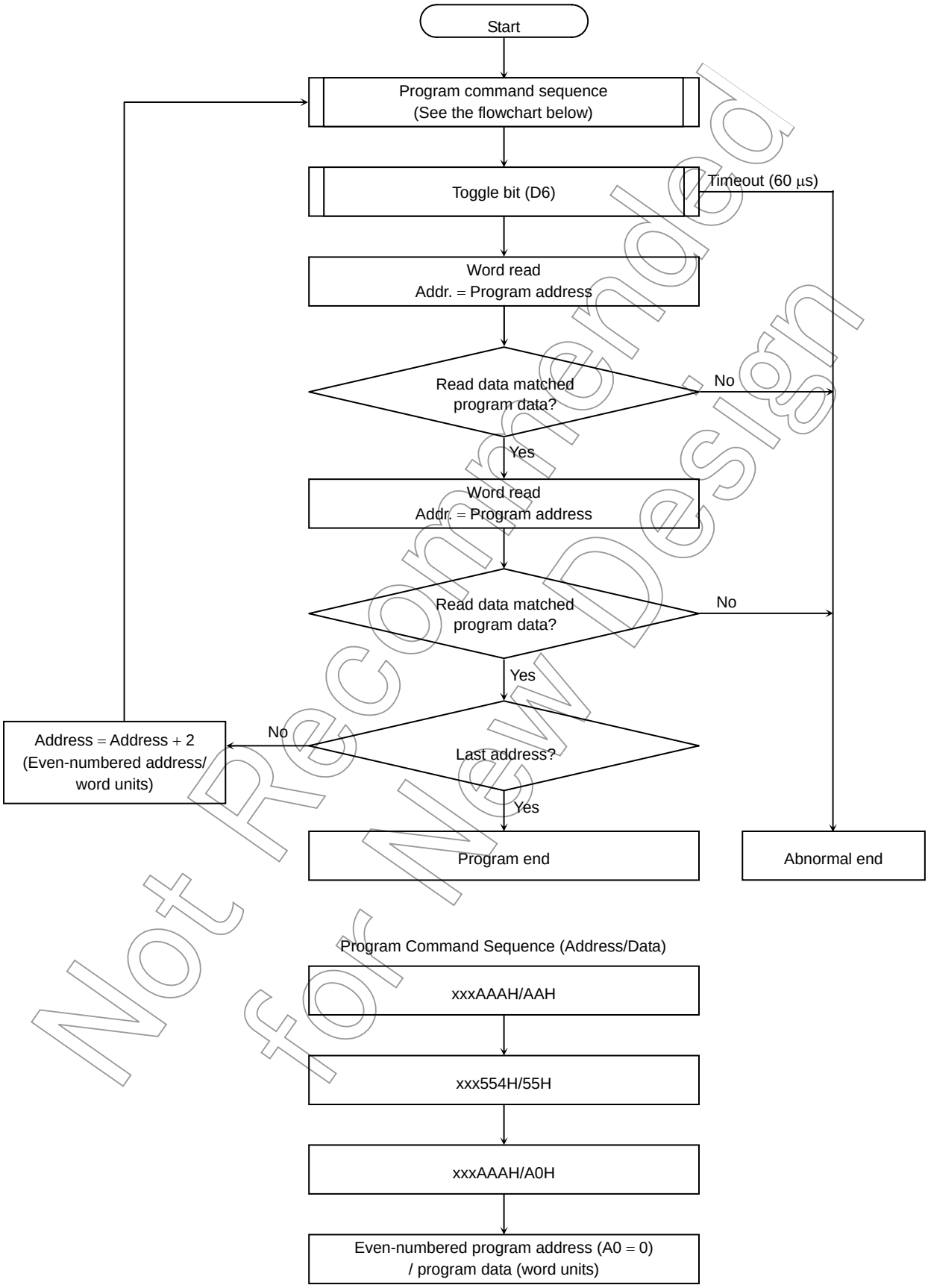
The microcontroller is started up in Single Boot mode to program the flash memory by the internal boot ROM program. In this mode, the internal boot ROM is mapped to an area including the interrupt vector table, in which the boot ROM program is executed. The flash memory is mapped to an address area different from the boot ROM area. The boot ROM program loads data into the flash memory by serial transfer. In Single Boot mode, interrupts must be disabled including non-maskable interrupts ($\overline{\text{NMI}}$, etc.).
For details, see 3.2.4"Single Boot Mode"

2) User Boot:

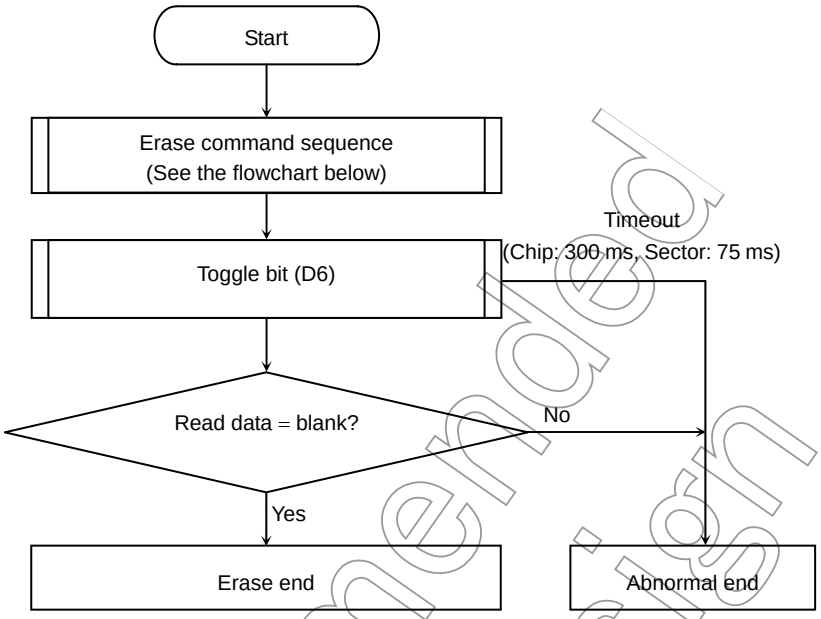
In this method, the flash memory is programmed by executing a user-created routine in Single Chip mode (normal operation mode). In this mode, the user-created program/erase routine must also be executed outside of the flash memory. It is also necessary to disable interrupts including non-maskable interrupts.
The user should prepare a flash memory program/erase routine (including routines for loading write data and writing the loaded data into the flash memory). In the main program, normal operation is switched to flash memory programming operation to execute the flash memory program/erase routine outside of the flash memory area. For example, the flash memory program/erase routine may be transferred from the flash memory to the internal RAM and executed there or it may be prepared and executed in external memory.
For details, see 3.2.5"Flash Memory".

Flowcharts: Flash memory access by the internal CPU

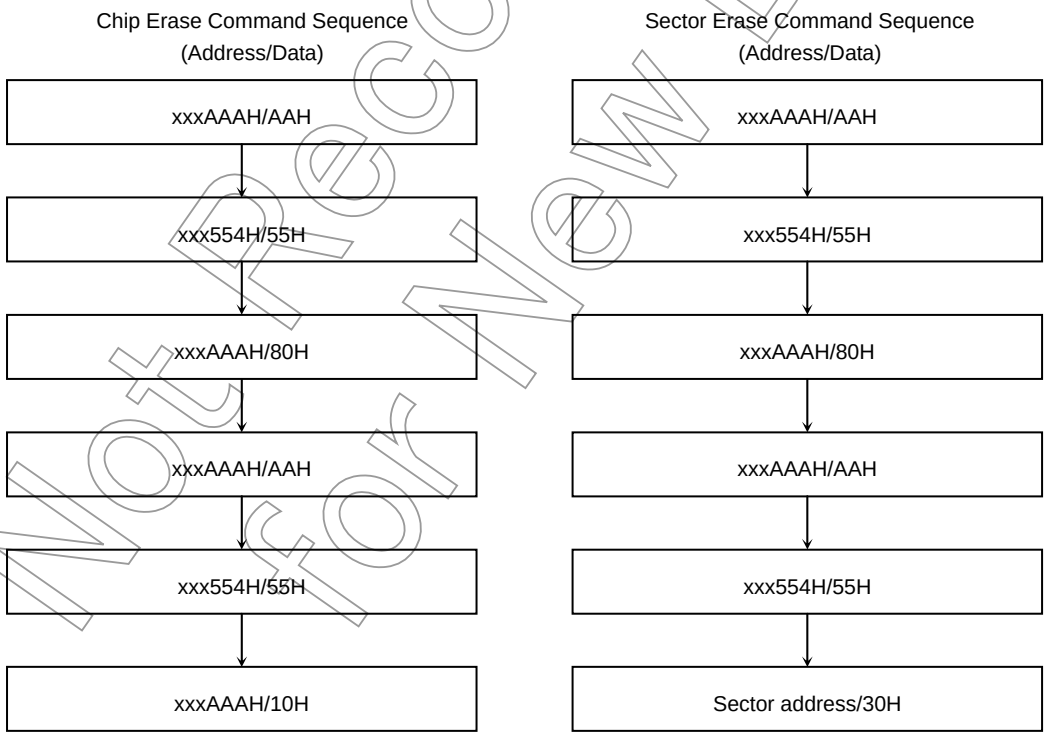
Single Word Program



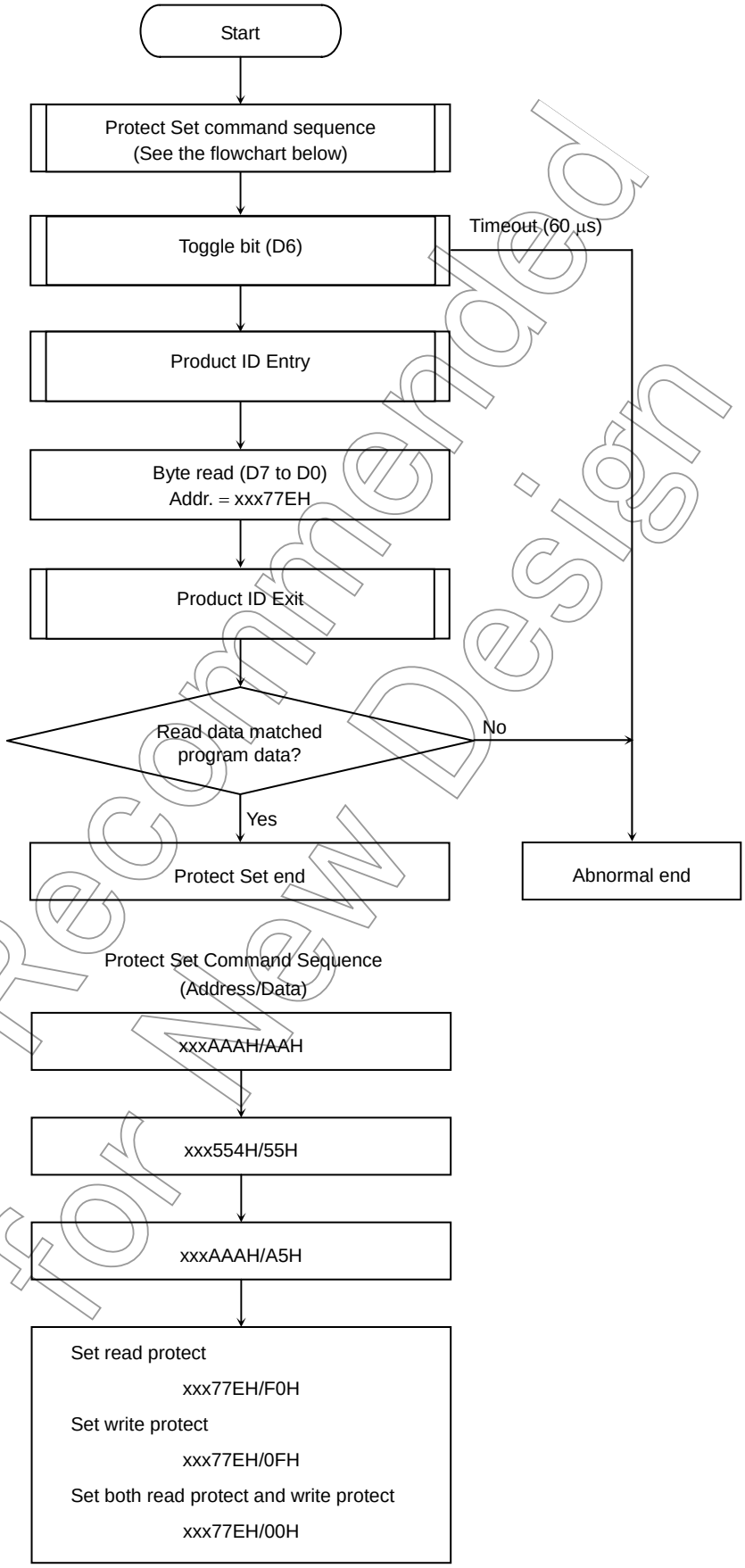
Chip Erase/Sector Erase



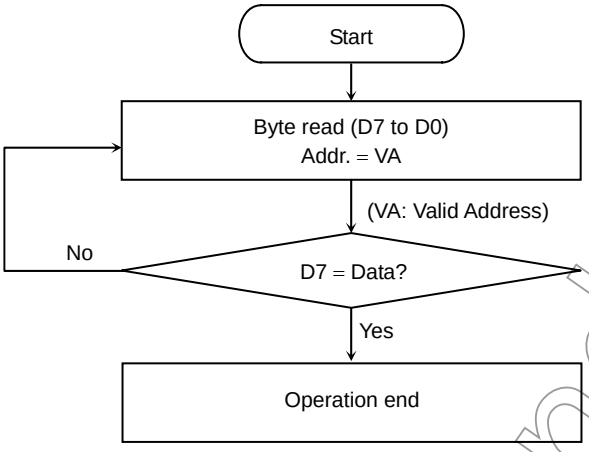
Note: In Chip Erase, whether or not the entire flash memory is blank is checked.
In Sector Erase, whether or not the selected sector is blank is checked.



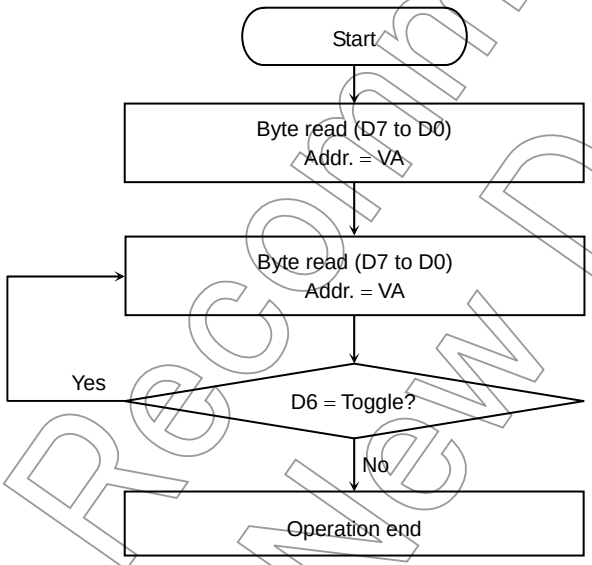
Read/Write Protect Set



Data Polling (D7)



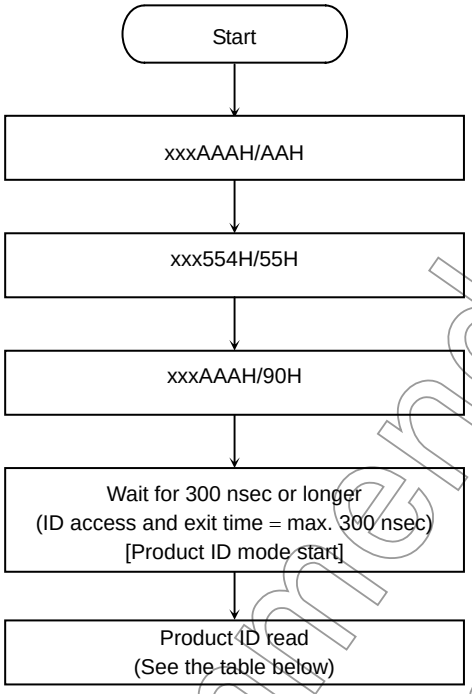
Toggle Bit (D6)



Note: Hardware sequence flags are read from the flash memory in byte units or word units.

VA: In Single Word Program, VA denotes the address to be programmed.
In Sector Erase, VA denotes any address in the selected sector.
In Chip Erase, VA denotes any address in the flash memory.
In Read Protect Set, VA denotes the protect set address (xx77EH).
In Write Protect Set, VA denotes the protect set address (xx77EH).

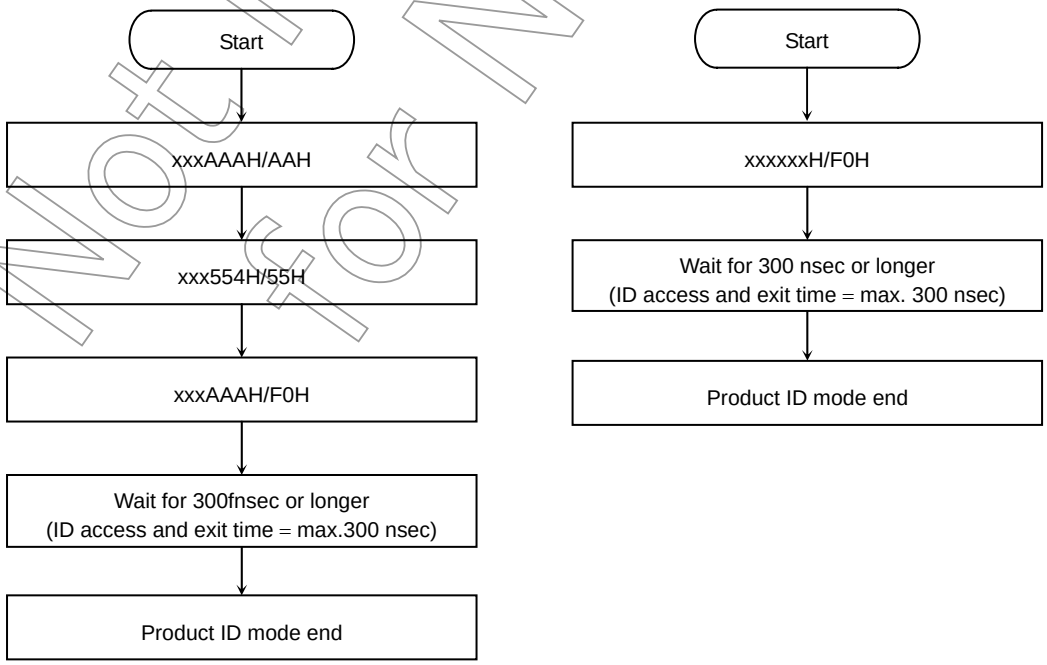
Product ID Entry



Read Values in Product ID Mode

	Address	Read Value
Vendor ID	xxxx00H	98H
Flash macro ID	xxxx02H	42H
Flash size ID	xxxx04H	1FH
Read/Write Protect status	xxx77EH	Data programmed when protection is set. When protection is not set, FFH.

Product ID Exit



(Example: Program to be loaded and executed in RAM)
Erase the flash memory (chip erase) and then write 0706H to address FE0000H.

```
;#### Flash memory chip erase processing ####
        ld      XIX, 0xFE0000               ; set start address
CHIPERASE:
        ld      (0xFE0AAA), 0xAA            ; 1st bus write cycle
        ld      (0xFE0554), 0x55            ; 2nd bus write cycle
        ld      (0xFE0AAA), 0x80            ; 3rd bus write cycle
        ld      (0xFE0AAA), 0xAA            ; 4th bus write cycle
        ld      (0xFE0554), 0x55            ; 5th bus write cycle
        ld      (0xFE0AAA), 0x10            ; 6th bus write cycle

        cal     TOGGLECHK                   ; check toggle bit

CHIPERASE_LOOP:
        ld      WA, (XIX+)                  ; read data from flash memory
        cp      WA, 0xFFFF                  ; blank data?
        j       ne, CHIPERASE_ERR           ; if not blank data, jump to error processing
        cp      XIX, 0xFFFFFF               ; end address (0xFFFFFF)?
        j       ULT, CHIPERASE_LOOP         ; check entire memory area and then end loop processing


;#### Flash memory program processing ####
        ld      XIX, 0xFE0000               ; set program address
        ld      WA, 0x0706                  ; set program data
PROGRAM:
        ld      (0xFE0AAA), 0xAA            ; 1st bus write cycle
        ld      (0xFE0554), 0x55            ; 2nd bus write cycle
        ld      (0xFE0AAA), 0xA0            ; 3rd bus write cycle
        ld      (XIX), WA                   ; 4th bus write cycle
        cal     TOGGLECHK                   ; check toggle bit

        ld      BC, (XIX)                   ; read data from flash memory
        cp      WA, BC.
        j       ne, PROGRAM_ERR             ; if programmed data cannot be read, error is determined
        ld      BC, (XIX)                   ; read data from flash memory
        cp      WA, BC
        j       ne, PROGRAM_ERR             ; if programmed data cannot be read, error is determined

PROGRAM_END:
        j       PROGRAM_END                 ; program operation end


;#### Toggle bit (D6) check processing ####
TOGGLECHK:
        ld      L, (XIX)
        and     L, 0y01000000               ; check toggle bit (D6)
        ld      H, L                        ; save first toggle bit data
TOGGLECHK1:
        ld      L, (XIX)
        and     L, 0y01000000               ; check toggle bit (D6)
        cp      L, H                        ; toggle bit = toggled?
        j       z, TOGGLECHK2               ; if not toggled, end processing
        ld      H, L                        ; save current toggle bit state
        j       TOGGLECHK1                  ; recheck toggle bit
TOGGLECHK2:
        ret


;####  Error processing ####
CHIPERASE_ERR:
        j       CHIPERASE_ERR               ; chip erase error

PROGRAM_ERR:
        j       PROGRAM_ERR                 ; program error
```

(Example: Program to be loaded and executed in RAM)
Erase data at addresses FF0000H to FF0FFFH (sector erase) and then write 0706H to address FF0000H.

```
;#### Flash memory sector erase processing ####
        ld      XIX, 0xFF0000               ; set start address
SECTORERASE:
        ld      (0xFE0AAA), 0xAA            ; 1st bus write cycle
        ld      (0xFE0554), 0x55            ; 2nd bus write cycle
        ld      (0xFE0AAA), 0x80            ; 3rd bus write cycle
        ld      (0xFE0AAA), 0xAA            ; 4th bus write cycle
        ld      (0xFE0554), 0x55            ; 5th bus write cycle
        ld      (XIX), 0x30                 ; 6th bus write cycle

        cal     TOGGLECHK                   ; check toggle bit

SECTORERASE_LOOP:
        ld      WA, (XIX+)                  ; read data from flash memory
        cp      WA, 0xFFFF                  ; blank data?
        j       ne, SECTORERASE_ERR         ; if not blank data, jump to error processing
        cp      XIX, 0xFF0FFF               ; end address (0xFF0FFF)?
        j       ULT, SECTORERASE_LOOP       ; check erased sector area and then end loop processing


;#### Flash memory program processing ####
        ld      XIX, 0xFF0000               ; set program address
        ld      WA, 0x0706                  ; set program data
PROGRAM:
        ld      (0xFE0AAA), 0xAA            ; 1st bus write cycle
        ld      (0xFE0554), 0x55            ; 2nd bus write cycle
        ld      (0xFE0AAA), 0xA0            ; 3rd bus write cycle
        ld      (XIX), WA                   ; 4th bus write cycle

        cal     TOGGLECHK                   ; check toggle bit

        ld      BC, (XIX)                   ; read data from flash memory
        cp      WA, BC
        j       ne, PROGRAM_ERR             ; if programmed data cannot be read, error is determined
        ld      BC, (XIX)                   ; read data from flash memory
        cp      WA, BC
        j       ne, PROGRAM_ERR             ; if programmed data cannot be read, error is determined

PROGRAM_END:
        j       PROGRAM_END                 ; program operation end


;#### Toggle bit (D6) check processing ####
TOGGLECHK:
        ld      L, (XIX)
        and     L, 0y01000000               ; check toggle bit (D6)
        ld      H, L                        ; save first toggle bit data
TOGGLECHK1:
        ld      L, (XIX)
        and     L, 0y01000000               ; check toggle bit (D6)
        cp      L, H                        ; toggle bit = toggled?
        j       z, TOGGLECHK2               ; If not toggled, end processing
        ld      H, L                        ; save current toggle bit state
        j       TOGGLECHK1                  ; Recheck toggle bit
TOGGLECHK2:
        ret


;#### Error processing ####
SECTORERASE_ERR:
        j       SECTORERASE_ERR             ;  sector erase error

PROGRAM_ERR:
        j       PROGRAM_ERR                 ; program error
```

(Example: Program to be loaded and executed in RAM)
Set read protection and write protection on the flash memory.

```
;#### Flash Memory Protect Set processing ####
        ld      XIX, 0xFE077E                   ; set protect address
PROTECT:
        ld      (0xFE0AAA), 0xAA                ; 1st bus write cycle
        ld      (0xFE0554), 0x55                ; 2nd bus write cycle
        ld      (0xFE0AAA), 0xA5                ; 3rd bus write cycle
        ld      (XIX), 0x00                     ; 4th bus write cycle

        cal     TOGGLECHK                       ; check toggle bit
        cal     PID_ENTRY                       ;
        ld      A, (XIX)                        ; read protected address
        cal     PID_EXIT                        ;
        cp      A, 0x00                         ;(0xFE077E)=0x00?
        j       ne, PROTECT_ERR                 ; protected?

PROTECT_END:
        j       PROTECT_END                     ; protect set operation completed

PROTECT_ERR:
        j       PROTECT_ERR                     ; protect set error


;#### Product ID Entry processing ####
PID_ENTRY:
        ld      (0xFE0AAA), 0xAA                ; 1st bus write cycle
        ld      (0xFE0554), 0x55                ; 2nd bus write cycle
        ld      (0xFE0AAA), 0x90                ; 3rd bus write cycle
        ; --- wait for 300 nsec or longer (execute NOP instruction [148nsec/@fFPH=27MHz] three times) ---
        nop
        nop
        nop                                     ; wait for 444 nsec
        ret


;#### Product ID Exit processing ####
PID_EXIT:
        ld      (0xFE0000), 0xF0                ; 1st bus write cycle
        ; --- wait for 300 nsec or longer (execute NOP instruction [148nsec/@fFPH=27MHz] three times) ---
        nop
        nop
        nop                                     ; wait for 444 nsec
        ret


;####  Toggle bit (D6) check processing ####
TOGGLECHK:
        ld      L, (XIX)
        and     L, 0y01000000                   ; check toggle bit (D6)
        ld      H, L                            ; save first toggle bit data
TOGGLECHK1:
        ld      L, (XIX)
        and     L, 0y01000000                   ; check toggle bit (D6)
        cp      L, H                            ; toggle bit = toggled?
        j       z, TOGGLECHK2                   ; if not toggled, end processing
        ld      H, L                            ; save current toggle bit state
        j       TOGGLECHK1                      ; recheck toggle bit
TOGGLECHK2:
        ret
```

(Example: Program to be loaded and executed in RAM)
Read data from address FE0000H.

```
;#### Flash memory read processing ####
READ:
        ld      WA, (0xFE0000)                  ; read data from flash memory
```

4. Electrical Characteristics

4.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Power supply voltage	Vcc	−0.5 to 4.0	V
Input voltage	VIN	−0.5 to Vcc + 0.5	
Output current (1 pin)	IOL	2	mA
Output current (1 pin)	IOH	−2	
Output current (Total)	ΣIOL	80	
Output current (Total)	ΣIOH	−80	
Power dissipation (Ta = 85°C)	PD	600	mW
Soldering temperature (10 s)	TSOLDER	260	°C
Storage temperature	TSTG	−65 to 150	
Operation temperature	TOPR	−40 to 85	
Number of Times Program Erase	N_{EW}	100	Cycle

Note: The absolute maximum ratings are rated values that 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 that include this device, ensure that no absolute maximum rating value will ever be exceeded.

Solderability of lead free products

Test parameter	Test condition	Note
Solderability	Use of Sn−37Pb solder Bath Solder bath temperature =230°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	Use of Sn−3.0Ag−0.5Cu solder bath Solder bath temperature = 245°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux (use of lead free)	

4.2 DC Characteristics (1/2)

Parameter		Symbol	Condition		Min	Typ.(Note)	Max	Unit
Power supply voltage (AVCC = DVCC, AVSS = DVSS = 0 V)		VCC	fc = 4 to 27 MHz	fs = 30 to 34 kHz	2.7		3.6	V
			fc = 2 to 16 MHz		2.2			
Power supply voltage (AVCC = DVCC, AVSS = DVSS = 0 V) for erase/program operations of flash memory		VCC	fc = 4 to 27 MHz, Ta = −10~40°C		2.7		3.6	V
Input low voltage	P00 to P17 (AD0 to AD15)	VIL	Vcc ≥ 2.7 V		−0.3		0.6	V
			Vcc < 2.7 V				0.2 Vcc	
	P20 to P97 (Except P63)	VIL1	Vcc ≥ 2.7 V				0.3 Vcc	
			Vcc < 2.7 V				0.2 Vcc	
	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$ P63 (INT0)	VIL2	Vcc ≥ 2.7 V				0.25 Vcc	
			Vcc < 2.7 V				0.15 Vcc	
	AM0 and AM1	VIL3	Vcc ≥ 2.7 V				0.3	
			Vcc < 2.7 V				0.3	
	X1	VIL4	Vcc ≥ 2.7 V				0.2 Vcc	
			Vcc < 2.7 V				0.1 Vcc	
Input high voltage	P00 to P17 (AD0 to AD15)	VIH	Vcc ≥ 2.7 V		2.0		Vcc + 0.3	V
			Vcc < 2.7 V		0.7 Vcc			
	P20 to P97 (Except P63)	VIH1	Vcc ≥ 2.7 V		0.7 Vcc			
			Vcc < 2.7 V		0.8 Vcc			
	$\overline{\text{RESET}}$, $\overline{\text{NMI}}$, P63 (INT0)	VIH2	Vcc ≥ 2.7 V		0.75 Vcc			
			Vcc < 2.7 V		0.85 Vcc			
	AM0 and AM1	VIH3	Vcc ≥ 2.7 V		Vcc − 0.3			
			Vcc < 2.7 V		Vcc − 0.3			
	X1	VIH4	Vcc ≥ 2.7 V		0.8 Vcc			
			Vcc < 2.7 V		0.9 Vcc			
Output low voltage		VOL	IOL = 1.6mA	Vcc ≥ 2.7 V			0.45	V
			IOL = 0.4mA	Vcc < 2.7 V			0.15 Vcc	
Output high voltage		VOH	IOH = −400 μA	Vcc ≥ 2.7 V	Vcc − 0.3			
			IOH = −200 μA	Vcc < 2.7 V	0.8 Vcc			

Note: Typical values are for when Ta = 25°C and V_{CC} = 3.0 V uncles otherwise noted.

DC Characteristics (2/2)

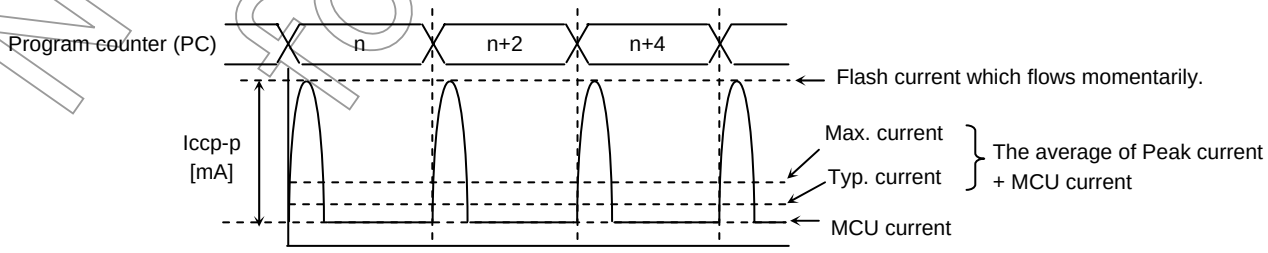
Parameter	Symbol	Condition	Min	Typ. (Note1)	Max	Unit
Input leakage current	ILI	$0.0 \leq V_{IN} \leq Vcc$		0.02	±5	μA
Output leakage current	ILO	$0.2 \leq V_{IN} \leq Vcc - 0.2$		0.05	±10	
Power down voltage (@STOP, RAM back up)	VSTOP	V IL2 = 0.2 Vcc, V IH2 = 0.8 Vcc	2.2		3.6	V
$\overline{RESET}$ pull-up resistor	RRST	Vcc = 2.7 V to 3.6 V	100		400	kΩ
		Vcc = 2.2 V	200		1000	
Pin capacitance	CIO	fc = 1 MHz			10	PF
Schmitt width $\overline{RESET}$, $\overline{NMI}$, INT0	VTH	Vcc ≥ 2.7 V	0.4	1.0		V
		Vcc < 2.7 V	0.3	0.8		
Programmable pull-up resistor	RKH	Vcc = 2.7 V to 3.6 V	100		400	kΩ
		Vcc = 2.2 V	200		1000	
NORMAL (Note 2)	Icc	Vcc = 2.7 V to 3.6 V fc = 27 MHz		13	20	mA
IDLE2				7	9	
IDLE1				3.4	4.5	
NORMAL (Note 2)		Vcc = 2.2 V (Typ. = 2.2V) fc = 16 MHz		5.5	8	mA
IDLE2				3.0	4.8	
IDLE1				1.5	2.9	
SLOW (Note 2)		Vcc = 2.2 V to 3.6 V fs = 32.768 kHz		20	55	μA
IDLE2				20	44	
IDLE1				10	40	
STOP		Vcc = 2.2 V to 3.6 V		1	25	μA
Peak current by intermitt operation	Iccp-p	Vcc = 2.2V to 3.6V		20		mA

Note 1: Typical values are for when Ta = 25°C and V_{CC} = 3.0 V unless otherwise noted.

Note 2: Icc measurement conditions (NORMAL, SLOW):
All functions are operational; output pins are open and input pins are fixed.

When the program is operating by the flash memory, or when data reed from the flash memory, the flash memory operate intermittently. Therefore, it outputs a peak current like a following diagram, momentarily. In this case, the power supply current; Icc (NORMAL/SLOW mode) is the sum of average value of a peak current and a MCU current value.

When designing the power supply, set to a circuit which a peak current can be supplyed. In SLOW mode, a defference of peak current and average current is large.



Flash memory intermittent operation

4.3 AC Characteristics

(1) Vcc = 2.7 V to 3.6 V

No.	Parameter	Symbol	Variable		f_{FPH} = 27 MHz		Unit
			Min	Max	Min	Max	
1	f_{FPH} period (= x)	t_{FPH}	37.0	31250	37.0		Ns
2	A0 to A15 valid → ALE falling	t_{AL}	0.5x – 6		12		ns
3	ALE falling → A0 to A15 hold	t_{LA}	0.5x – 16		2		ns
4	ALE high pulse width	t_{LL}	x – 20		17		ns
5	ALE falling → $\overline{RD}$ / $\overline{WR}$ falling	t_{LC}	0.5x – 14		4		ns
6	$\overline{RD}$ rising → ALE rising	t_{CLR}	0.5x – 10		8		ns
7	$\overline{WR}$ rising → ALE rising	t_{CLW}	x – 10		27		ns
8	A0 to A15 vlalid → $\overline{RD}$ / $\overline{WR}$ falling	t_{ACL}	x – 23		14		ns
9	A0 to A21 valid → $\overline{RD}$ / $\overline{WR}$ falling	t_{ACH}	1.5x – 26		29		ns
10	$\overline{RD}$ rising → A0 to A21 hold	t_{CAR}	0.5x – 13		5		ns
11	$\overline{WR}$ rising → A0 to A21 hold	t_{CAW}	x – 13		24		ns
12	A0 to A15 valid → D0 to D15 input	t_{ADL}		3.0x – 38		73	ns
13	A0 to A21 valid → D0 to D15 input	t_{ADH}		3.5x – 41		88	ns
14	$\overline{RD}$ falling → D0 to D15 input	t_{RD}		2.0x – 30		44	ns
15	$\overline{RD}$ low puse width	t_{RR}	2.0x – 15		59		ns
16	$\overline{RD}$ rising → D0 to D15 hold	t_{HR}	0		0		ns
17	$\overline{RD}$ rising → A0 to A15 output	t_{RAE}	x – 15		22		ns
18	$\overline{WR}$ low pulse width	t_{WW}	1.5x – 15		40		ns
19	D0 to D15 valid → $\overline{WR}$ rising	t_{DW}	1.5x – 35		20		ns
20	$\overline{WR}$ rising → D0 to D15 hold	t_{WD}	x – 25		12		ns
21	A0 to A21 valid → Port input	t_{APH}		3.5x – 89		40	ns
22	A0 to A21 valid → Port hold	t_{APH2}	3.5x		129		ns
23	A0 to A21 valid → Port valid	t_{AP}		3.5x + 80		209	ns

AC measurement conditions

- Output level: High 0.7 × Vcc/Low 0.3 × V_{CC}, C_L = 50 pF
- Input level: High 0.9 × Vcc/Low 0.1 × V_{CC}

Note: Symbol [x] in the above table means the period of clock f_{FPH}. It's half period the system clock f_{SYS} for CPU core.

The period of clock f_{FPH} depends on the clock gear setting or the selection of high/low oscillator frequency.

(2) Vcc = 2.2 V to 3.6 V

No.	Parameter	Symbol	Variable		f_{FPH} = 16 MHz		Unit
			Min	Max	Min	Max	
1	f_{FPH} period (= x)	t_{FPH}	62.5	31250	62.5		ns
2	A0 to A15 valid → ALE falling	t_{AL}	0.5x – 18		13		ns
3	ALE falling → A0 to A15 hold	t_{LA}	0.5x – 25		6		ns
4	ALE high pulse width	t_{LL}	x – 30		32		ns
5	ALE falling → $\overline{RD}$ / $\overline{WR}$ falling	t_{LC}	0.5x – 25		6		ns
6	$\overline{RD}$ rising → ALE rising	t_{CLR}	0.5x – 15		16		ns
7	$\overline{WR}$ rising → ALE rising	t_{CLW}	x – 15		47		ns
8	A0 to A15 vlalid → $\overline{RD}$ / $\overline{WR}$ falling	t_{ACL}	x – 30		32		ns
9	A0 to A21 valid → $\overline{RD}$ / $\overline{WR}$ falling	t_{ACH}	1.5x – 30		63		ns
10	$\overline{RD}$ rising → A0 to A21 hold	t_{CAR}	0.5x – 21		10		ns
11	$\overline{WR}$ rising → A0 to A21 hold	t_{CAW}	x – 25		37		ns
12	A0 to A15 valid → D0 to D15 input	t_{ADL}		3.0x – 50		137	ns
13	A0 to A21 valid → D0 to D15 input	t_{ADH}		3.5x – 56		162	ns
14	$\overline{RD}$ falling → D0 to D15 input	t_{RD}		2.0x – 50		75	ns
15	$\overline{RD}$ low puse width	t_{RR}	2.0x – 28		97		ns
16	$\overline{RD}$ rising → D0 to D15 hold	t_{HR}	0		0		ns
17	$\overline{RD}$ rising → A0 to A15 output	t_{RAE}	x – 25		37		ns
18	$\overline{WR}$ low pulse width	t_{WW}	1.5x – 29		64		ns
19	D0 to D15 valid → $\overline{WR}$ rising	t_{DW}	1.5x – 60		33		ns
20	$\overline{WR}$ rising → D0 to D15 hold	t_{WD}	x – 40		22		ns
21	A0 to A21 valid → Port input	t_{APH}		3.5x – 100		68	ns
22	A0 to A21 valid → Port hold	t_{APH2}	3.5x		218		ns
23	A0 to A21 valid → Port valid	t_{AP}		3.5x + 150		368	ns

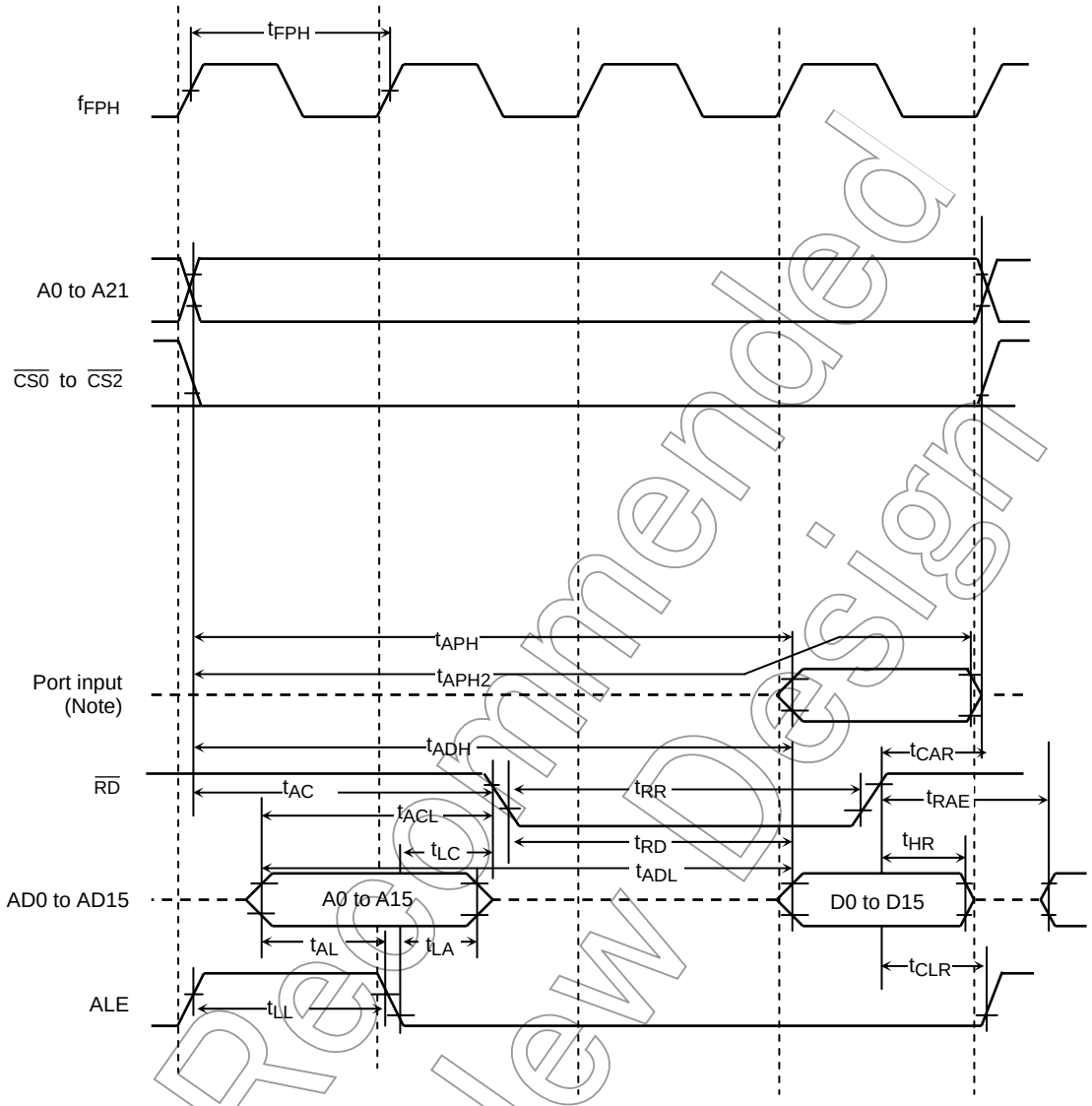
AC measurement conditions

- Output level: High 0.7 × Vcc/Low 0.3 × V_{CC}, C_L = 50 pF
- Input level: High 0.9 × Vcc/Low 0.1 × V_{CC}

Note: Symbol [x] in the above table means the period of clock f_{FPH}. It's half period the system clock f_{SYS} for CPU core.

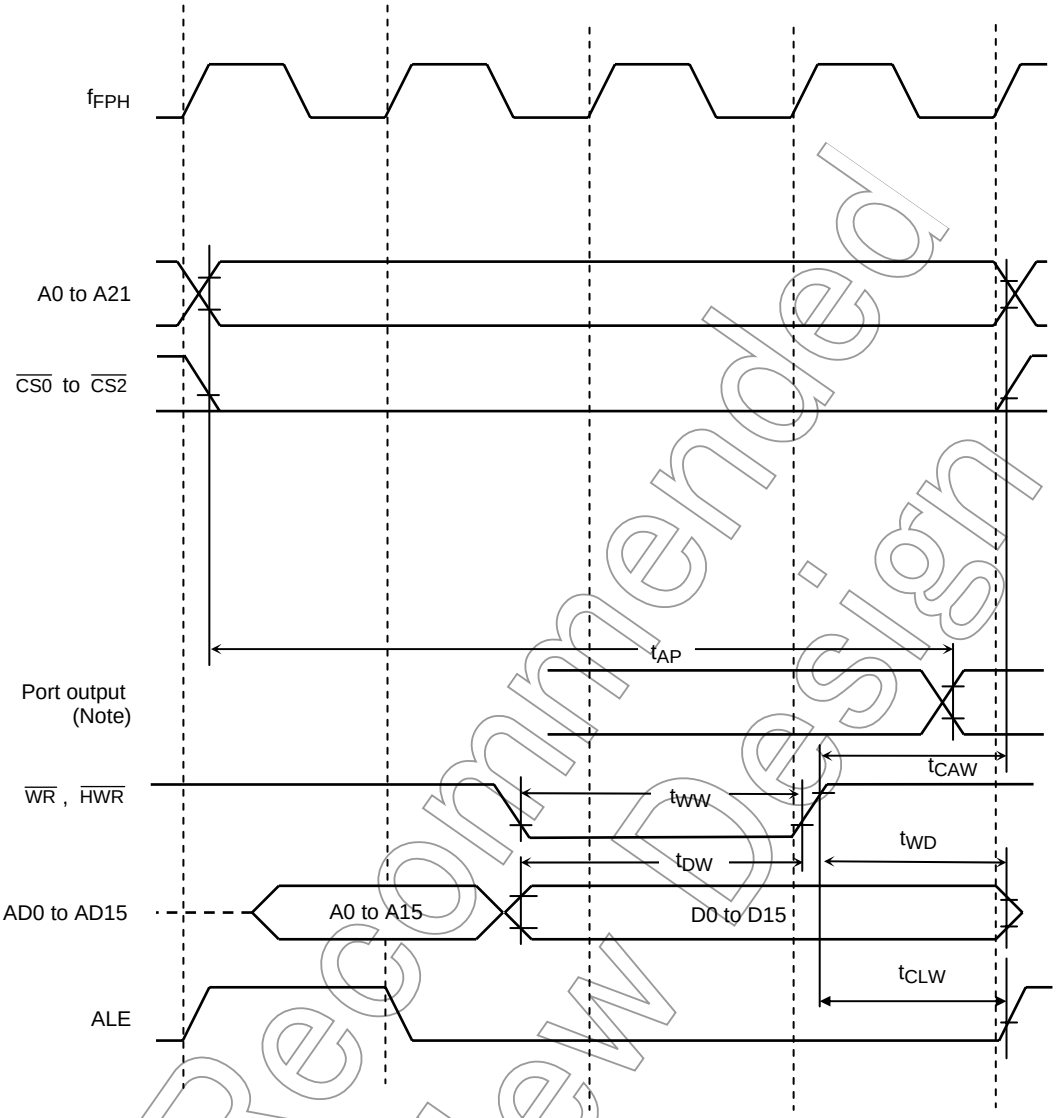
The period of clock f_{FPH} depends on the clock gear setting or the selection of high/low oscillator frequency.

(3) Read cycle



Note: Since the CPU accesses the internal area to read data from a port, the control signals of external pins such as $\overline{RD}$ and $\overline{CS}$ are not enabled. Therefore, the above waveform diagram should be regarded as depicting internal operation. Please also note that the timing and AC characteristics of port input/output shown above are typical representation. For details, contact your local Toshiba sales representative.

(4) Write cycle



Note: Since the CPU accesses the internal area to write data to a port, the control signals of external pins such as $\overline{WR}$ and $\overline{CS}$ are not enabled. Therefore, the above waveform diagram should be regarded as depicting internal operation. Please also note that the timing and AC characteristics of port input/output shown above are typical representation. For details, contact your local Toshiba sales representative.

4.4 AD Conversion Characteristics

AVCC = VCC, AVSS = VSS

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog input voltage	VAIN		AVSS		AVCC	V
Error (Not including quantization errors)	–	VCC = 2.2 V to 3.6 V		±1.0	±4.0	LSB

Note 1: 1 LSB = (AVCC − AVSS)/1024 [V]

Note 2: Minimum operation frequency:
The operaion of AD converter is guranteed only using fc (High frequency oscillator).
fs (Low frequency oscillator) is not guranteed. But When frequency of clock selected by clock gear is more than and eqaull 4 MHz in using fc, it is guranteed (fFPH ≥ 4 MHz).

Note 3: The value for Icc (Current of VCC pin) includes the current which flows through the AVCC pin.

4.5 Serial Channel Timing (I/O interface mode)

(1) SCLK input mode

Parameter	Symbol	Variable		10 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	16X		1.6		0.59		μs
Output data → SCLK rising/falling	t_{OSS}	$t_{SCY}/2 - 4X - 110$ (V_{CC} = 2.7 V to 3.6 V)		290		38		ns
		$t_{SCY}/2 - 4X - 180$ (V_{CC} = 2.2 V)		220		–		
SCLK rising/falling → Output data hold	t_{OHS}	$t_{SCY}/2 + 2X + 0$		1000		370		ns
SCLK rising /falling → Input data hold	t_{HSR}	3X + 10		310		121		ns
SCLK rising/falling → Valid data input	t_{SRD}		$t_{SCY} - 0$		1600		592	ns
Valid data input → SCLK rising/falling	t_{RDS}	0		0		0		ns

(2) SCLK ouptut mode

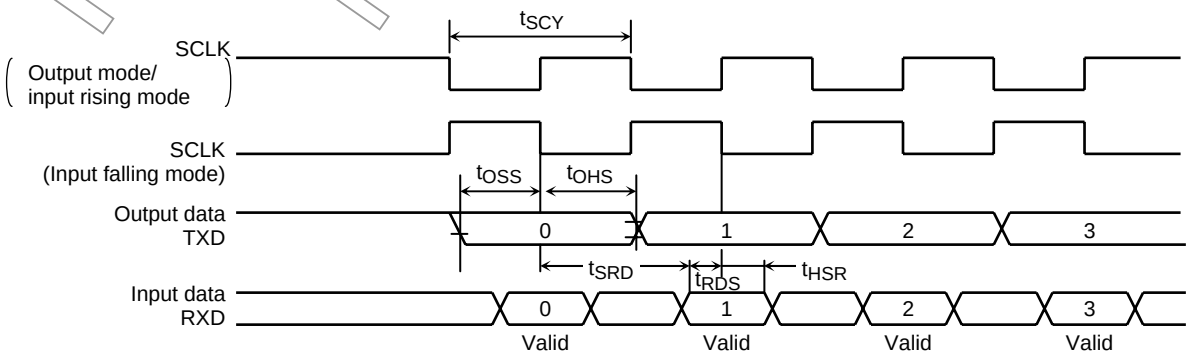
Parameter	Symbol	Variable		10 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	16X	8192X	1.6	819	0.59	303	μs
Output data → SCLK rising/falling	t_{OSS}	$t_{SCY}/2 - 40$		760		256		ns
SCLK rising/falling → Output data hold	t_{OHS}	$t_{SCY}/2 - 40$		760		256		ns
SCLK rising/falling → Input data hold	t_{HSR}	0		0		0		ns
SCLK rising/falling → Valid data input	t_{SRD}		$t_{SCY} - 1X - 180$		1320		375	ns
Valid data input → SCLK rising/falling	t_{RDS}	1X + 180		280		217		ns

Note 1: SCLK rising/falling: The rising edge is used in SCLK rising mode.
The falling edge is used in SCLK falling mode.

Note 2: 27 MHz and 10 MHz values are calculated from t_{SCY} = 16X case.

Note 3: Symbol [x] in the above table means the period of clock f_{FPH}. It's half period the system clock f_{SYS} for CPU core.

The period of clock f_{FPH} depends on the clock gear setting or the selection of high/low oscillator frequency.



4.6 Event Counter (TA0IN, TA4IN, TB0IN0 and TB0IN1)

Parameter	Symbol	Variable		10 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Clock period	t_{VCK}	8X + 100		900		396		ns
Clock low level pulse width	t_{VCKL}	4X + 40		440		188		ns
Clock high level pulse width	t_{VCKH}	4X + 40		440		188		ns

4.7 Interrupt and Capture

(1) $\overline{NMI}$ and INT0 Interrupts

Parameter	Symbol	Variable		10 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
$\overline{NMI}$ and INT0 low level pulse width	t_{INTAL}	4X + 40		440		188		ns
$\overline{NMI}$ and INT0 high level pulse width	t_{INTAH}	4X + 40		440		188		ns

(2) INT5 and INT6 interrupts, capture

INT5 and INT6 input pulse width depend on the system clock selection and clock selection for prescaler. Below table show pulse width of each operation clock.

System Clock Selection SYSCR1 <SYSCK>	Clock Selection for Prescaler SYSCR0 <PRCK1:0>	t_{INTBL} (INT5 and INT6 low level pulse width)		t_{INTBH} (INT5 and INT6 high level pulse width)		Unit
		Valiable	f_{FPH} = 27MHz	Valiable	f_{FPH} = 27MHz	
		Min	Min	Min	Min	
0 (fc)	00 (f_{FPH})	8X + 100	396	8X + 100	396	ns
	10 (fc/16)	128Xc + 0.1	4.8	128Xc + 0.1	4.8	μs
1 (fs)	00 (f_{FPH})	8X + 0.1	244.3	8X + 0.1	244.3	

Note 1: "Xc" shows period of clock fc in high frequency oscillator.

Note 2: Symbol [x] in the above table means the period of clock f_{FPH}. It's half period the system clock f_{SYS} for CPU core.

The period of clock f_{FPH} depends on the clock gear setting or the selection of high/low oscillator frequency.

4.8 Flash Characteristics

(1) Rewriting

Parameter	Condition	Min	Typ	Max	Unit
Guarantee on Flash-memory rewriting	Vcc = 2.7V to 3.6V, fc = 4 to 27 MHz Ta = −10 to 40 °C	–	–	100	Times

5. Port Section Equivalent Circuit Diagrams

- Reading the circuit diagrams

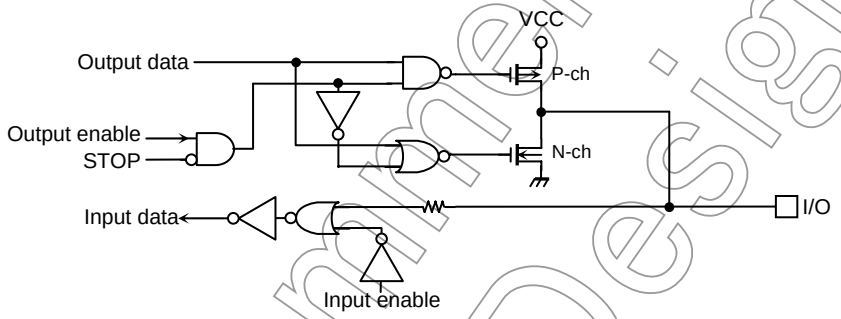
Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

The dedicated signal is described below.

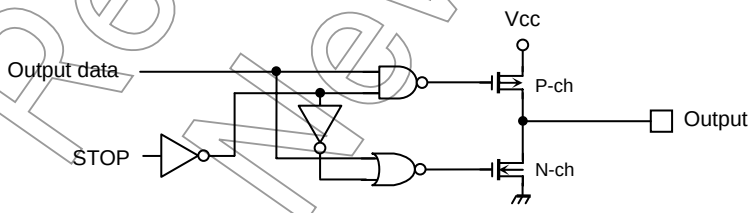
STOP : This signal becomes active 1 when the HALT mode setting register is set to the STOP mode (SYSCR2<HALTM1:0> = "01") and the CPU executes the HALT instruction. When the drive enable bit SYSCR2<DRVE> is set to "1", however STOP remains at "0".

- The input protection resistance ranges from several tens of ohms to several hundreds of ohms.

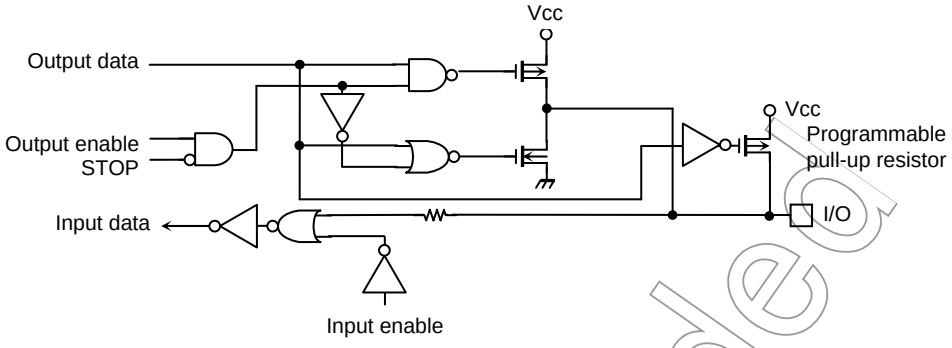
■ P0 (AD0~AD7), P1 (AD8~AD15, A8~A15), P2 (A16~A21, A0~A5), P60, P70~P74, P80~P83, P91~P92, P94~P95



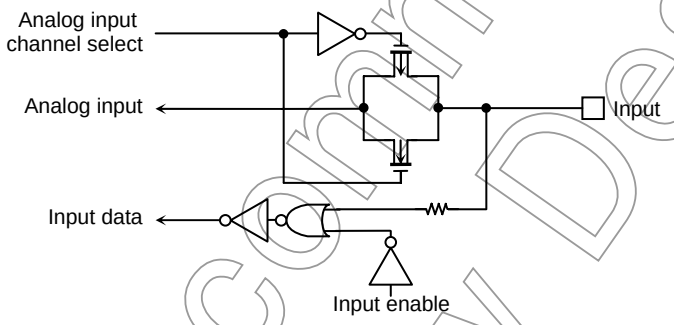
■ P30 ($\overline{RD}$), P31 ($\overline{WR}$)



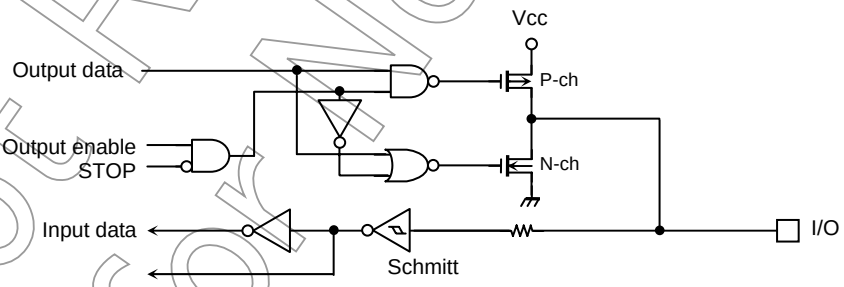
■ P32, P40~P42



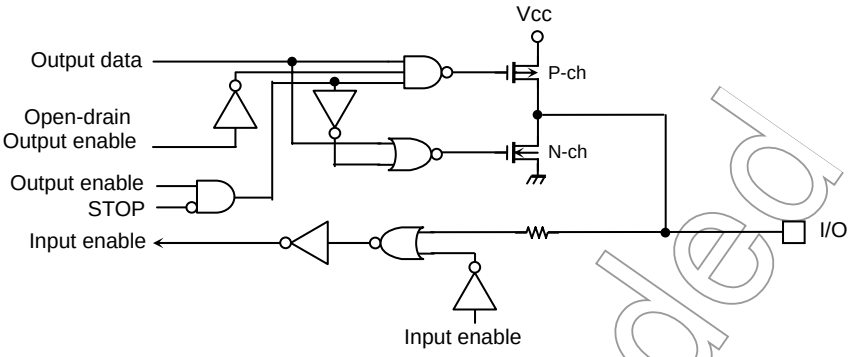
■ P5 (AN0~AN3)



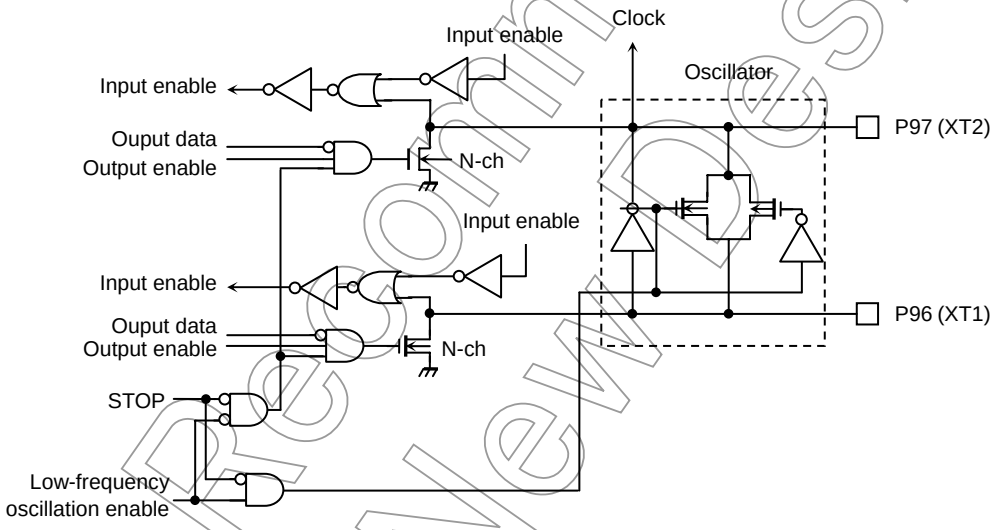
■ P63 (INT0)



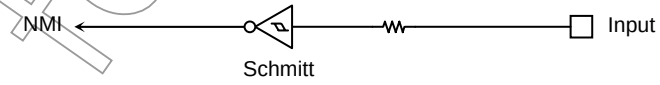
■ P61 (SO/SDA), P62 (SI/SCL), P90 (TXD0), P93 (TXD1)



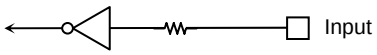
■ P96 (XT1), P97 (XT2)



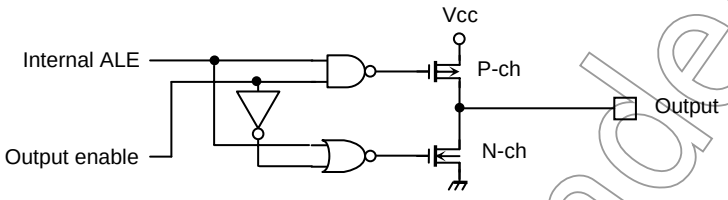
■ $\overline{\text{NMI}}$



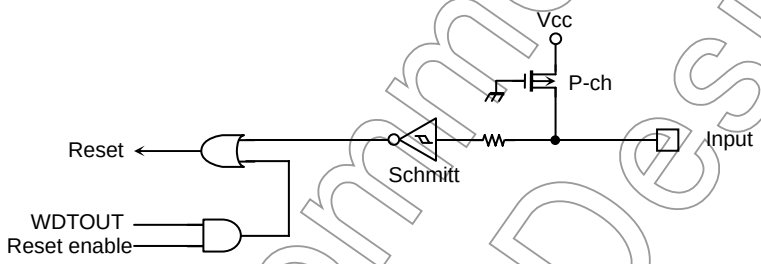
■ AM0~AM1



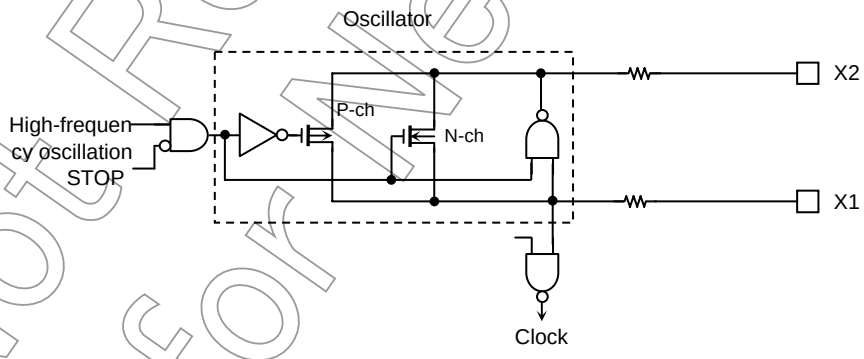
■ ALE



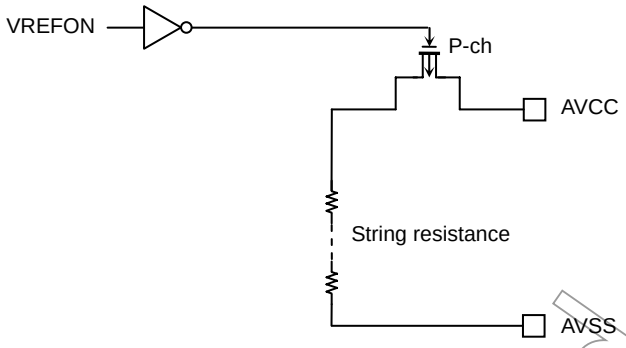
■ $\overline{\text{RESET}}$



■ X1, X2



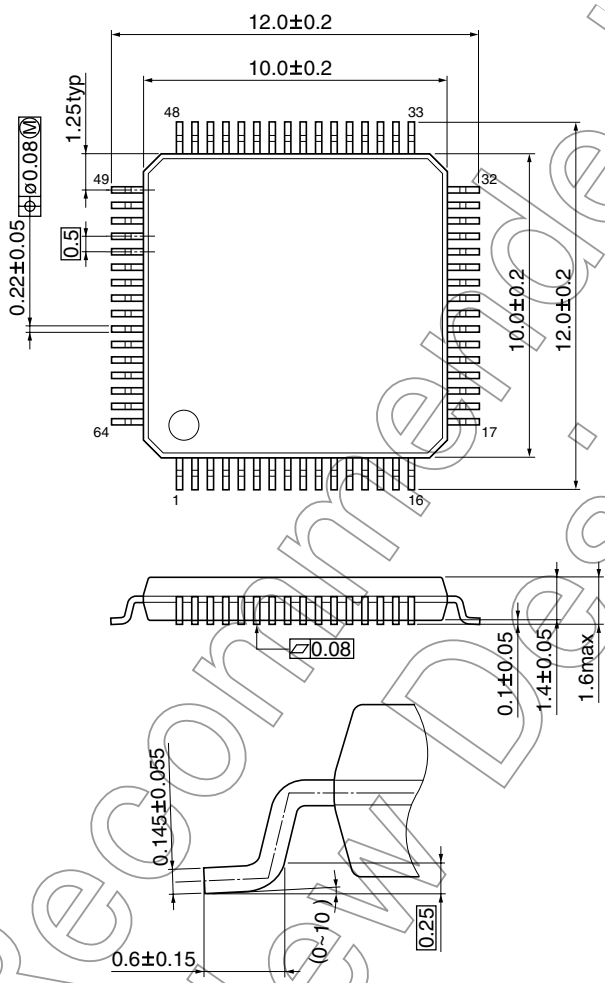
■ VREFH, VREFL



6. Package Dimensions

LQFP64-P-1010-0.50D

Unit: mm



QFP64-P-1414-0.80A

Unit: mm