Renesas Electronics America Inc - M30620FCAFP#U5 Datasheet

Welcome to E-XFL.COM

What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

Applications of "Embedded - Microcontrollers"

Details	
Product Status	Active
Core Processor	M16C/60
Core Size	16-Bit
Speed	16MHz
Connectivity	SIO, UART/USART
Peripherals	DMA, PWM, WDT
Number of I/O	85
Program Memory Size	128KB (128K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	10K x 8
Voltage - Supply (Vcc/Vdd)	4.2V ~ 5.5V
Data Converters	A/D 10x10b; D/A 2x8b
Oscillator Type	Internal
Operating Temperature	-20°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	100-BQFP
Supplier Device Package	100-QFP (14x20)
Purchase URL	https://www.e-xfl.com/product-detail/renesas-electronics-america/m30620fcafp-u5

Mitsubishi plans to release the following products in the M16C/62A (80-pin version) group:

(1) Support for mask ROM version and flash memory version

(2) ROM capacity

(3) Package

80P6S-A : Plastic molded QFP (mask ROM and flash memory versions)

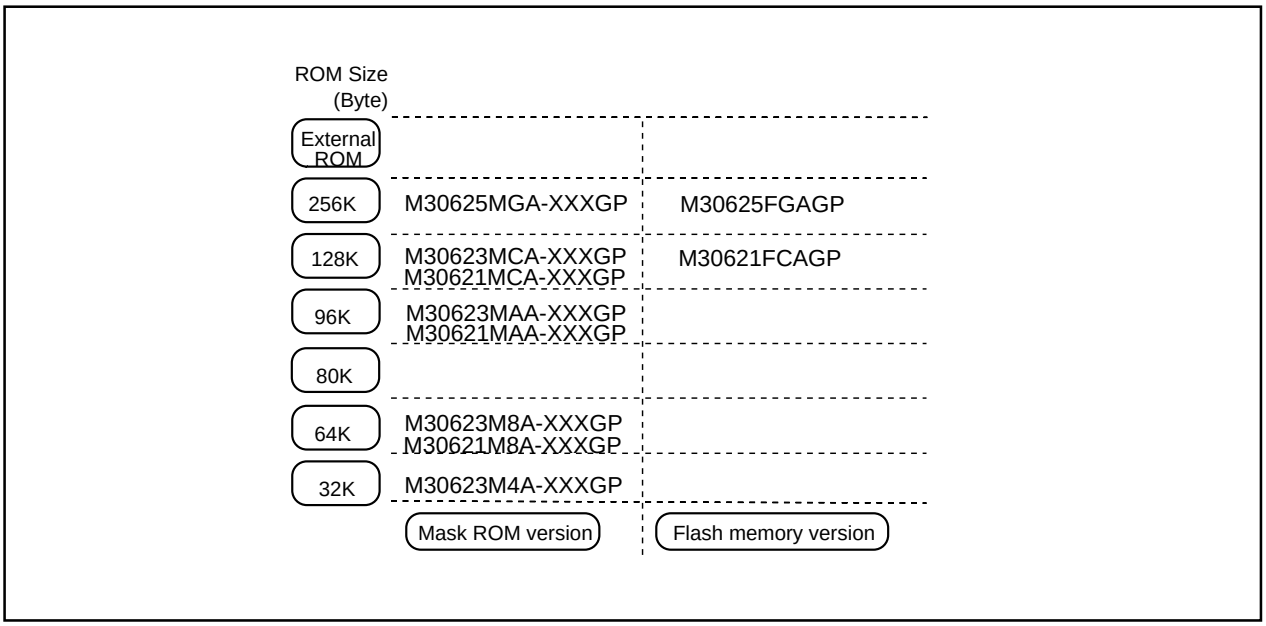


Figure 1.1.3. ROM expansion

The M16C/62A (80-pin version) group products currently supported are listed in Table 1.1.2.

Table 1.1.2. M16C/62A (80-pin version) group

As of November 2001

Type No	ROM capacity	RAM capacity	Package type	Remarks
M30623M4A-XXXGP	32 Kbytes	3 Kbytes	80P6S-A	mask ROM version
M30623M8A-XXXGP	64 Kbytes	4 Kbytes	80P6S-A	
M30623MAA-XXXGP	96 Kbytes	5 Kbytes	80P6S-A	
M30623MCA-XXXGP	128 Kbytes	5 Kbytes	80P6S-A	
M30621M8A-XXXGP	64 Kbytes	10 Kbytes	80P6S-A	
M30621MAA-XXXGP	96 Kbytes	10 Kbytes	80P6S-A	
M30621MCA-XXXGP	128 Kbytes	10 Kbytes	80P6S-A	
M30625MGA-XXXGP	256 Kbytes	20 Kbytes	80P6S-A	
M30621FCAGP	128 Kbytes	10 Kbytes	80P6S-A	Flash memory 5V version
M30625FGAGP	256 Kbytes	20 Kbytes	80P6S-A	

Software wait

A software wait can be inserted by setting the wait bit (bit 7) of the processor mode register 1 (address 0005_{16}) (Note).

A software wait is inserted in the internal ROM/RAM area by setting the wait bit of the processor mode register 1. When set to “0”, each bus cycle is executed in one BCLK cycle. When set to “1”, each bus cycle is executed in two BCLK cycles. After the microcomputer has been reset, this bit defaults to “0”. Set this bit after referring to the recommended operating conditions (main clock input oscillation frequency) of the electric characteristics.

The SFR area is always accessed in two BCLK cycles regardless of the setting of this control bit.

Table 1.8.1 shows the software wait and bus cycles. Figure 1.8.3 shows example bus timing when using software waits.

Note: Before attempting to change the contents of the processor mode register 1, set bit 1 of the protect register (address $000A_{16}$) to “1”.

Table 1.8.1. Software waits and bus cycles

Area	Wait bit	Bus cycle
SFR	Invalid	2 BCLK cycles
Internal ROM/RAM	0	1 BCLK cycle
	1	2 BCLK cycles

Clock Output

In single-chip mode, the clock output function select bits (bits 0 and 1 at address 0006_{16}) enable f_8, f_{32}, or fc to be output from the $P5_7$/CLK_{OUT} pin. When the WAIT peripheral function clock stop bit (bit 2 at address 0006_{16}) is set to “1”, the output of f_8 and f_{32} stops when a WAIT instruction is executed.

Stop Mode

Writing “1” to the all-clock stop control bit (bit 0 at address 0007_{16}) stops all oscillation and the microcomputer enters stop mode. In stop mode, the content of the internal RAM is retained provided that V_{CC} remains above 2V.

Because the oscillation , BCLK, f_1 to f_{32}, f_{1SIO2} to f_{32SIO2}, f_C, f_{C32}, and f_{AD} stops in stop mode, peripheral functions such as the A-D converter and watchdog timer do not function. However, timer A and timer B operate provided that the event counter mode is set to an external pulse, and UARTi(i = 0 to 2), SI/O3,4 functions provided an external clock is selected. Table 1.9.2 shows the status of the ports in stop mode.

Stop mode is cancelled by a hardware reset or an interrupt. If an interrupt is to be used to cancel stop mode, that interrupt must first have been enabled, and the priority level of the interrupt which is not used to cancel must have been changed to 0. If returning by an interrupt, that interrupt routine is executed. If only a hardware reset or an $\overline{NMI}$ interrupt is used to cancel stop mode, change the priority level of all interrupt to 0, then shift to stop mode.

When shifting from high-speed/medium-speed mode to stop mode and at a reset, the main clock division select bit 0 (bit 6 at address 0006_{16}) is set to “1”. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.

Table 1.9.2. Port status during stop mode

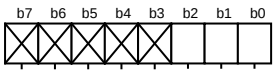
Pin		Single-chip mode
Port		Retains status before stop mode
CLK_{OUT}	When fc selected	“H”
	When f_8, f_{32} selected	Retains status before stop mode

Protection

The protection function is provided so that the values in important registers cannot be changed in the event that the program runs out of control. Figure 1.9.6 shows the protect register. The values in the processor mode register 0 (address 0004_{16}), processor mode register 1 (address 0005_{16}), system clock control register 0 (address 0006_{16}), system clock control register 1 (address 0007_{16}), port P9 direction register (address $03F3_{16}$), SI/O3 control register (address 0362_{16}), and SI/O4 control register (address 0366_{16}) can only be changed when the respective bit in the protect register is set to “1”. Therefore, important outputs can be allocated to port P9.

If, after “1” (write-enabled) has been written to the port P9 direction register and SI/Oi control register (i=3,4) write-enable bit (bit 2 at address $000A_{16}$), a value is written to any address, the bit automatically reverts to “0” (write-inhibited). However, the system clock control registers 0 and 1 write-enable bit (bit 0 at $000A_{16}$) and processor mode register 0 and 1 write-enable bit (bit 1 at $000A_{16}$) do not automatically return to “0” after a value has been written to an address. The program must therefore be written to return these bits to “0”.

Protect register



Symbol: PRCR
Address: $000A_{16}$
When reset: $XXXXX000_2$

Bit symbol	Bit name	Function	R	W
PRC0	Enables writing to system clock control registers 0 and 1 (addresses 0006_{16} and 0007_{16})	0 : Write-inhibited 1 : Write-enabled	○	○
PRC1	Enables writing to processor mode registers 0 and 1 (addresses 0004_{16} and 0005_{16})	0 : Write-inhibited 1 : Write-enabled	○	○
PRC2	Enables writing to port P9 direction register (address $03F3_{16}$) and SI/Oi control registers (i=3,4) (addresses 0362_{16} and 0366_{16}) (Note)	0 : Write-inhibited 1 : Write-enabled	○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate.			—	—

Note: Writing a value to an address after “1” is written to this bit returns the bit to “0” . Other bits do not automatically return to “0” and they must therefore be reset by the program.

Figure 1.9.6. Protect register

Rewrite the interrupt control register

To rewrite the interrupt control register, do so at a point that does not generate the interrupt request for that register. If there is possibility of the interrupt request occur, rewrite the interrupt control register after the interrupt is disabled. The program examples are described as follow:

Example 1:

```
INT_SWITCH1:
    FCLR    I             ; Disable interrupts.
    AND.B   #00h, 0055h   ; Clear TA0IC int. priority level and int. request bit.
    NOP                   ; Four NOP instructions are required when using HOLD function.
    NOP
    FSET    I             ; Enable interrupts.
```

Example 2:

```
INT_SWITCH2:
    FCLR    I             ; Disable interrupts.
    AND.B   #00h, 0055h   ; Clear TA0IC int. priority level and int. request bit.
    MOV.W   MEM, R0       ; Dummy read.
    FSET    I             ; Enable interrupts.
```

Example 3:

```
INT_SWITCH3:
    PUSHC   FLG           ; Push Flag register onto stack
    FCLR    I             ; Disable interrupts.
    AND.B   #00h, 0055h   ; Clear TA0IC int. priority level and int. request bit.
    POPC    FLG           ; Enable interrupts.
```

The reason why two NOP instructions (four when using the HOLD function) or dummy read are inserted before FSET I in Examples 1 and 2 is to prevent the interrupt enable flag I from being set before the interrupt control register is rewritten due to effects of the instruction queue.

When a instruction to rewrite the interrupt control register is executed but the interrupt is disabled, the interrupt request bit is not set sometimes even if the interrupt request for that register has been generated. This will depend on the instruction. If this creates problems, use the below instructions to change the register.

Instructions : AND, OR, BCLR, BSET

Precautions for Interrupts

(1) Reading address 00000$_{16}$

- When maskable interrupt is occurred, CPU reads the interrupt information (the interrupt number and interrupt request level) in the interrupt sequence.
  The interrupt request bit of the certain interrupt written in address 00000$_{16}$ will then be set to “0”.
  Even if the address 00000$_{16}$ is read out by software, “0” is set to the enabled highest priority interrupt source request bit. Therefore interrupt can be canceled and unexpected interrupt can occur.
  Do not read address 00000$_{16}$ by software.

(2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to 0000$_{16}$. Accepting an interrupt before setting a value in the stack pointer may become a factor of runaway. Be sure to set a value in the stack pointer before accepting an interrupt. When using the $\overline{\text{NMI}}$ interrupt, initialize the stack point at the beginning of a program. Concerning the first instruction immediately after reset, generating any interrupts including the $\overline{\text{NMI}}$ interrupt is prohibited.

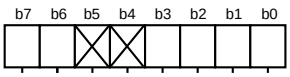
(3) The $\overline{\text{NMI}}$ interrupt

- The $\overline{\text{NMI}}$ interrupt can not be disabled. Be sure to connect $\overline{\text{NMI}}$ pin to Vcc via a pull-up resistor if unused. Be sure to work on it.
- The $\overline{\text{NMI}}$ pin also serves as P8$_5$, which is exclusively input. Reading the contents of the P8 register allows reading the pin value. Use the reading of this pin only for establishing the pin level at the time when the $\overline{\text{NMI}}$ interrupt is input.
- Do not reset the CPU with the input to the $\overline{\text{NMI}}$ pin being in the “L” state.
- Do not attempt to go into stop mode with the input to the $\overline{\text{NMI}}$ pin being in the “L” state. With the input to the $\overline{\text{NMI}}$ being in the “L” state, the CM10 is fixed to “0”, so attempting to go into stop mode is turned down.
- Do not attempt to go into wait mode with the input to the $\overline{\text{NMI}}$ pin being in the “L” state. With the input to the $\overline{\text{NMI}}$ pin being in the “L” state, the CPU stops but the oscillation does not stop, so no power is saved. In this instance, the CPU is returned to the normal state by a later interrupt.
- Signals input to the $\overline{\text{NMI}}$ pin require an “L” level of 1 clock or more, from the operation clock of the CPU.

(4) External interrupt

- Either an “L” level or an “H” level of at least 250 ns width is necessary for the signal input to pins $\overline{\text{INT0}}$ to $\overline{\text{INT2}}$ regardless of the CPU operation clock.
- When the polarity of the $\overline{\text{INT0}}$ to $\overline{\text{INT2}}$ pins is changed, the interrupt request bit is sometimes set to “1”. After changing the polarity, set the interrupt request bit to “0”. Figure 1.10.13 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

DMA0 request cause select register



Symbol	Address	When reset
DM0SL	$03B8_{16}$	00_{16}

Bit symbol	Bit name	Function	R	W
DSEL0	DMA request cause select bit	b3 b2 b1 b0 0 0 0 0 : Falling edge of INT0 pin 0 0 0 1 : Software trigger 0 0 1 0 : Timer A0 0 0 1 1 : Timer A1 0 1 0 0 : Timer A2 0 1 0 1 : Timer A3 0 1 1 0 : Timer A4 (DMS=0) /two edges of INT0 pin (DMS=1) 0 1 1 1 : Timer B0 (DMS=0) Timer B3 (DMS=1) 1 0 0 0 : Timer B1 (DMS=0) Timer B4 (DMS=1) 1 0 0 1 : Timer B2 (DMS=0) Timer B5 (DMS=1) 1 0 1 0 : UART0 transmit 1 0 1 1 : UART0 receive 1 1 0 0 : UART2 transmit 1 1 0 1 : UART2 receive 1 1 1 0 : A-D conversion 1 1 1 1 : UART1 transmit	○	○
DSEL1			○	○
DSEL2			○	○
DSEL3			○	○
Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”.			–	–
DMS	DMA request cause expansion select bit	0 : Normal 1 : Expanded cause	○	○
DSR	Software DMA request bit	If software trigger is selected, a DMA request is generated by setting this bit to “1” (When read, the value of this bit is always “0”)	○	○

Figure 1.12.2. DMAC register (1)

(2) DMAC transfer cycles

Any combination of even or odd transfer read and write addresses is possible. Table 1.12.2 shows the number of DMAC transfer cycles.

The number of DMAC transfer cycles can be calculated as follows:

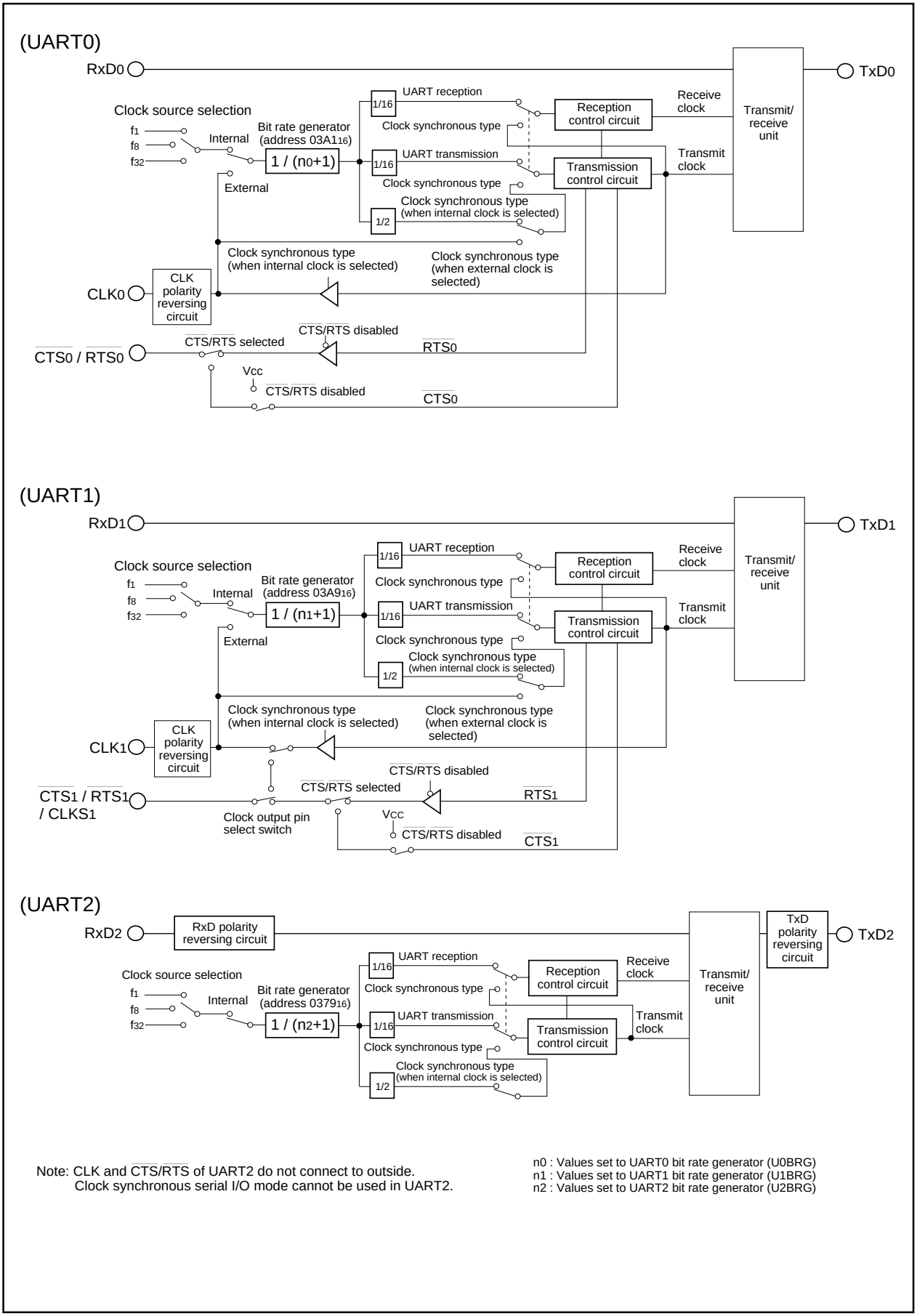
No. of transfer cycles per transfer unit = No. of read cycles x j + No. of write cycles x k

Table 1.12.2. No. of DMAC transfer cycles

Transfer unit	Bus width	Access address	Single-chip mode	
			No. of read cycles	No. of write cycles
8-bit transfers (DMBIT= “1”)	16-bit (BYTE= “L”)	Even	1	1
		Odd	1	1
16-bit transfers (DMBIT= “0”)	16-bit (BYTE = “L”)	Even	1	1
		Odd	2	2

Coefficient j, k

Internal memory		
Internal ROM/RAM No wait	Internal ROM/RAM With wait	SFR area
1	2	2



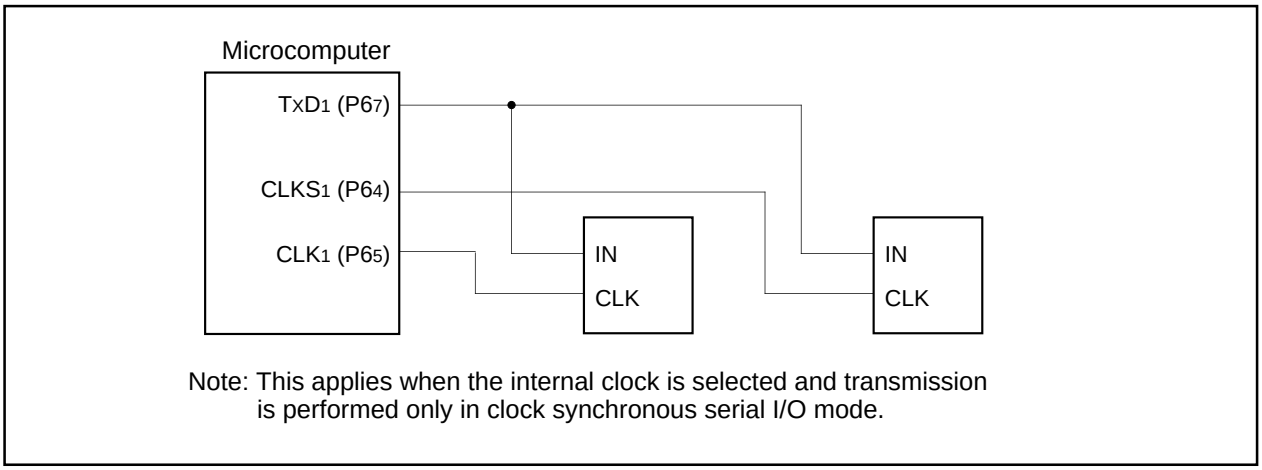
Note: CLK and CTS/RTS of UART2 do not connect to outside.
Clock synchronous serial I/O mode cannot be used in UART2.

n0 : Values set to UART0 bit rate generator (U0BRG)
n1 : Values set to UART1 bit rate generator (U1BRG)
n2 : Values set to UART2 bit rate generator (U2BRG)

Figure 1.14.1. Block diagram of UARTi (i = 0 to 2)

(c) Transfer clock output from multiple pins function (UART1)

This function allows the setting two transfer clock output pins and choosing one of the two to output a clock by using the CLK and CLKS select bit (bits 4 and 5 at address 03B0$_{16}$). (See Figure 1.14.3.) The multiple pins function is valid only when the internal clock is selected for UART1. Note that when this function is selected, UART1 $\overline{CTS}$/$\overline{RTS}$ function cannot be used.



Note: This applies when the internal clock is selected and transmission is performed only in clock synchronous serial I/O mode.

Figure 1.14.14. The transfer clock output from the multiple pins function usage

(d) Continuous receive mode

If the continuous receive mode enable bit (bits 2 and 3 at address 03B0$_{16}$, bit 5 at address 037D$_{16}$) is set to “1”, the unit is placed in continuous receive mode. In this mode, when the receive buffer register is read out, the unit simultaneously goes to a receive enable state without having to set dummy data to the transmit buffer register back again.

(4) Repeat sweep mode 0

In repeat sweep mode 0, the pins selected using the A-D sweep pin select bit are used for repeat sweep A-D conversion. Table 1.15.5 shows the specifications of repeat sweep mode 0. Figure 1.15.7 shows the A-D control register in repeat sweep mode 0.

Table 1.15.5. Repeat sweep mode 0 specifications

Item	Specification
Function	The pins selected by the A-D sweep pin select bit are used for repeat A-D conversion
Start condition	Writing “1” to A-D conversion start flag
Stop condition	Writing “0” to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	AN_0 and AN_1 (2 pins), AN_0 to AN_3 (4 pins), AN_0 to AN_5 (6 pins), or AN_0 to AN_7 (8 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin (at any time)

A-D control register 0 (Note)

b7	b6	b5	b4	b3	b2	b1	b0
			1	1			

Symbol: ADCON0 — Address: $03D6_{16}$ — When reset: $00000XXX_2$

Bit symbol	Bit name	Function	R	W
CH0	Analog input pin select bit	Invalid in repeat sweep mode 0	O	O
CH1			O	O
CH2			O	O
MD0	A-D operation mode select bit 0	b4 b3 1 1 : Repeat sweep mode 0	O	O
MD1			O	O
TRG	Trigger select bit	0 : Software trigger 1 : AD_{TRG} trigger	O	O
ADST	A-D conversion start flag	0 : A-D conversion disabled 1 : A-D conversion started	O	O
CKS0	Frequency select bit 0	0 : $f_{AD}/4$ is selected 1 : $f_{AD}/2$ is selected	O	O

Note: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.

A-D control register 1 (Note 1)

b7	b6	b5	b4	b3	b2	b1	b0
		1			0		

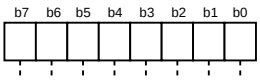
Symbol: ADCON1 — Address: $03D7_{16}$ — When reset: 00_{16}

Bit symbol	Bit name	Function	R	W
SCAN0	A-D sweep pin select bit	When single sweep and repeat sweep mode 0 are selected b1 b0 0 0 : AN_0, AN_1 (2 pins) 0 1 : AN_0 to AN_3 (4 pins) 1 0 : AN_0 to AN_5 (6 pins) 1 1 : AN_0 to AN_7 (8 pins)	O	O
SCAN1			O	O
MD2	A-D operation mode select bit 1	Set to “0” when this mode is selected	O	O
BITS	8/10-bit mode select bit	0 : 8-bit mode 1 : 10-bit mode	O	O
CKS1	Frequency select bit 1	0 : $f_{AD}/2$ or $f_{AD}/4$ is selected 1 : f_{AD} is selected	O	O
VCUT	Vref connect bit	1 : Vref connected	O	O
OPA0	External op-amp connection mode bit (Note 2)	b7 b6 0 0 : ANEX0 and ANEX1 are not used 0 1 : ANEX0 input is A-D converted 1 0 : ANEX1 input is A-D converted 1 1 : External op-amp connection mode	O	O
OPA1			O	O

Note 1: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: Neither “01” nor “10” can be selected with the external op-amp connection mode bit.

Figure 1.15.7. A-D conversion register in repeat sweep mode 0

Port Pi direction register (Note 1, 2)

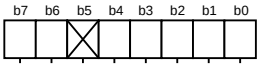


Symbol	Address	When reset
PDi (i = 0 to 10, except 8)	$03E2_{16}$, $03E3_{16}$, $03E6_{16}$, $03E7_{16}$, $03EA_{16}$ $03EB_{16}$, $03EE_{16}$, $03EF_{16}$, $03F3_{16}$, $03F6_{16}$	00_{16} 00_{16}

Bit symbol	Bit name	Function	R	W
PDi_0	Port Pi_0 direction register	0 : Input mode (Functions as an input port) 1 : Output mode (Functions as an output port) (i = 0 to 10 except 8)	○	○
PDi_1	Port Pi_1 direction register		○	○
PDi_2	Port Pi_2 direction register		○	○
PDi_3	Port Pi_3 direction register		○	○
PDi_4	Port Pi_4 direction register		○	○
PDi_5	Port Pi_5 direction register		○	○
PDi_6	Port Pi_6 direction register		○	○
PDi_7	Port Pi_7 direction register		○	○

Note 1: Set bit 2 of protect register (address $000A_{16}$) to “1” before rewriting to the port P9 direction register.
Note 2: P_1, $P4_4$ to $P4_7$, $P7_2$ to $P7_5$, $P9_1$ are not connected to the outside, but are present within the microcomputer, so set the direction registers to output so that these pin are reserved for future use.

Port P8 direction register



Symbol	Address	When reset
PD8	$03F2_{16}$	$00X00000_2$

Bit symbol	Bit name	Function	R	W
PD8_0	Port $P8_0$ direction register	0 : Input mode (Functions as an input port) 1 : Output mode (Functions as an output port)	○	○
PD8_1	Port $P8_1$ direction register		○	○
PD8_2	Port $P8_2$ direction register		○	○
PD8_3	Port $P8_3$ direction register		○	○
PD8_4	Port $P8_4$ direction register		○	○
	Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate.		—	—
PD8_6	Port $P8_6$ direction register	0 : Input mode (Functions as an input port) 1 : Output mode (Functions as an output port)	○	○
PD8_7	Port $P8_7$ direction register		○	○

Figure 1.18.6. Direction register

VCC = 5V

Timing requirements (referenced to VCC = 5V, VSS = 0V at Topr = – 20°C to 85°C / – 40°C to 85°C (*) unless otherwise specified)

* : Specify a product of -40°C to 85°C to use it.

Table 1.20.7. External clock input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t_c	External clock input cycle time	62.5		ns
$t_{w(H)}$	External clock input HIGH pulse width	25		ns
$t_{w(L)}$	External clock input LOW pulse width	25		ns
t_r	External clock rise time		15	ns
t_f	External clock fall time		15	ns

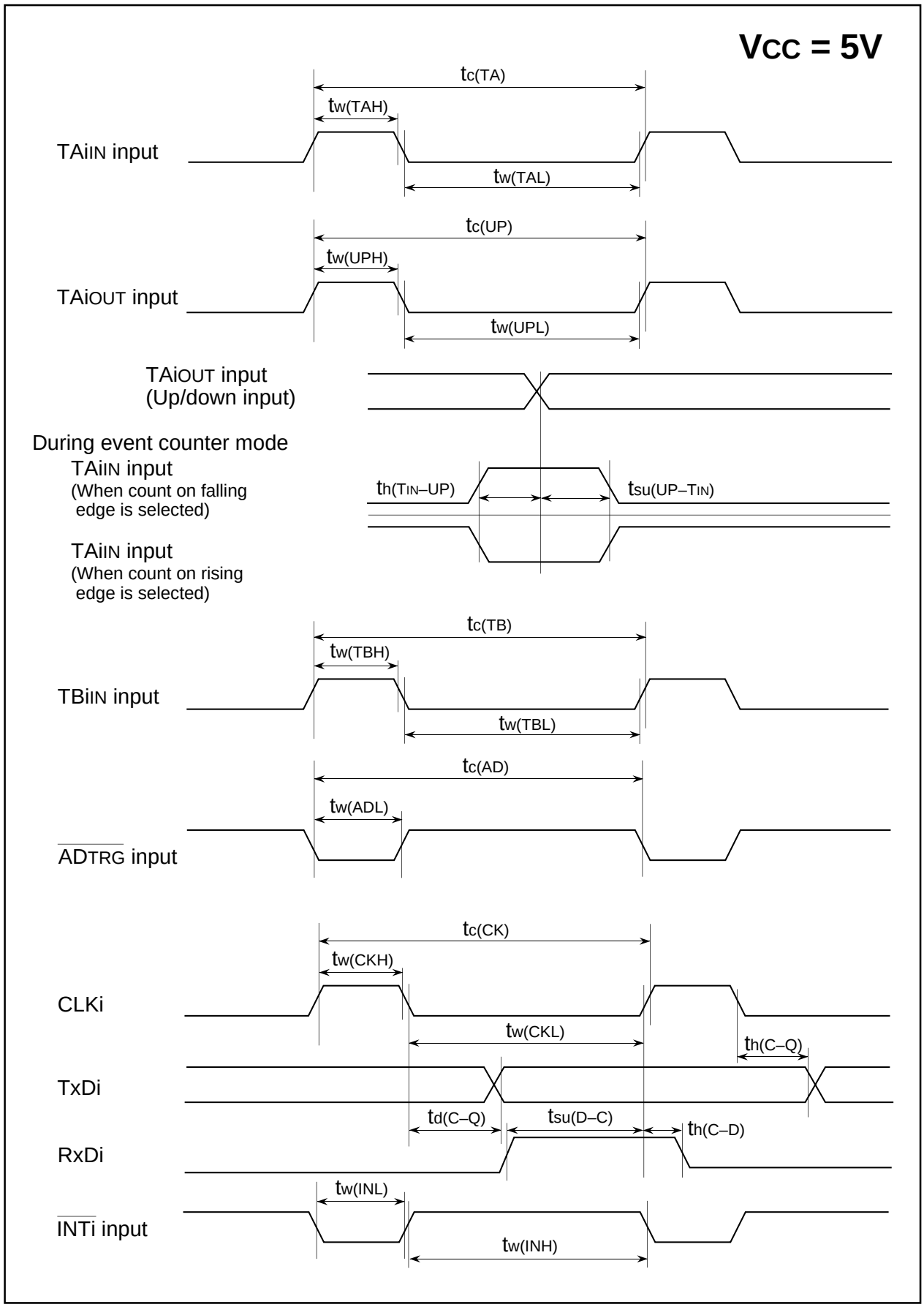


Figure 1.20.1. Vcc=5V timing diagram

GZZ-SH13-58B<98A0>

Mask ROM number	

MITSUBISHI ELECTRIC-CHIP 16-BIT MICROCOMPUTER M30623M8A-XXXGP MASK ROM CONFIRMATION FORM

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ✵ .

✵	Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
		Date issued	Date :			

✵1. Check sheet

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in.
Prepare 3.5 inches 2HD (IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. : ☐ M30623M8A-XXXGP

File code : ☐☐☐☐☐☐☐☐ (hex)

Mask file name : ☐☐☐☐☐☐☐☐ .MSK (alpha-numeric 8-digit)

✵2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.

For the M30623M8A-XXXGP, submit the 80P6S mark specification sheet.

✵3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

(1) Which kind of XIN-XOUT oscillation circuit is used?

☐ Ceramic resonator ☐ Quartz-crystal oscillator

☐ External clock input ☐ Other ( )

What frequency do not use?

f(XIN) = ☐ MHz

Page Program Command (41$_{16}$)

Page program allows for high-speed programming in units of 256 bytes. Page program operation starts when the command code "41$_{16}$" is written in the first bus cycle. In the second bus cycle through the 129th bus cycle, the write data is sequentially written 16 bits at a time. At this time, the addresses A$_0$-A$_7$ need to be incremented by 2 from "00$_{16}$" to "FE$_{16}$." When the system finishes loading the data, it starts an auto write operation (data program and verify operation).

Whether the auto write operation is completed can be confirmed by reading the status register or the flash memory control register 0. At the same time the auto write operation starts, the read status register mode is automatically entered, so the content of the status register can be read out. The status register bit 7 (SR7) is set to 0 at the same time the auto write operation starts and is returned to 1 upon completion of the auto write operation. In this case, the read status register mode remains active until the Read Array command (FF$_{16}$) or Read Lock Bit Status command (71$_{16}$) is written or the flash memory is reset using its reset bit.

The RY/$\overline{BY}$ status flag of the flash memory control register 0 is 0 during auto write operation and 1 when the auto write operation is completed as is the status register bit 7.

After the auto write operation is completed, the status register can be read out to know the result of the auto write operation. For details, refer to the section where the status register is detailed.

Figure 1.22.4 shows an example of a page program flowchart.

Each block of the flash memory can be write protected by using a lock bit. For details, refer to the section where the data protect function is detailed.

Additional writes to the already programmed pages are prohibited.

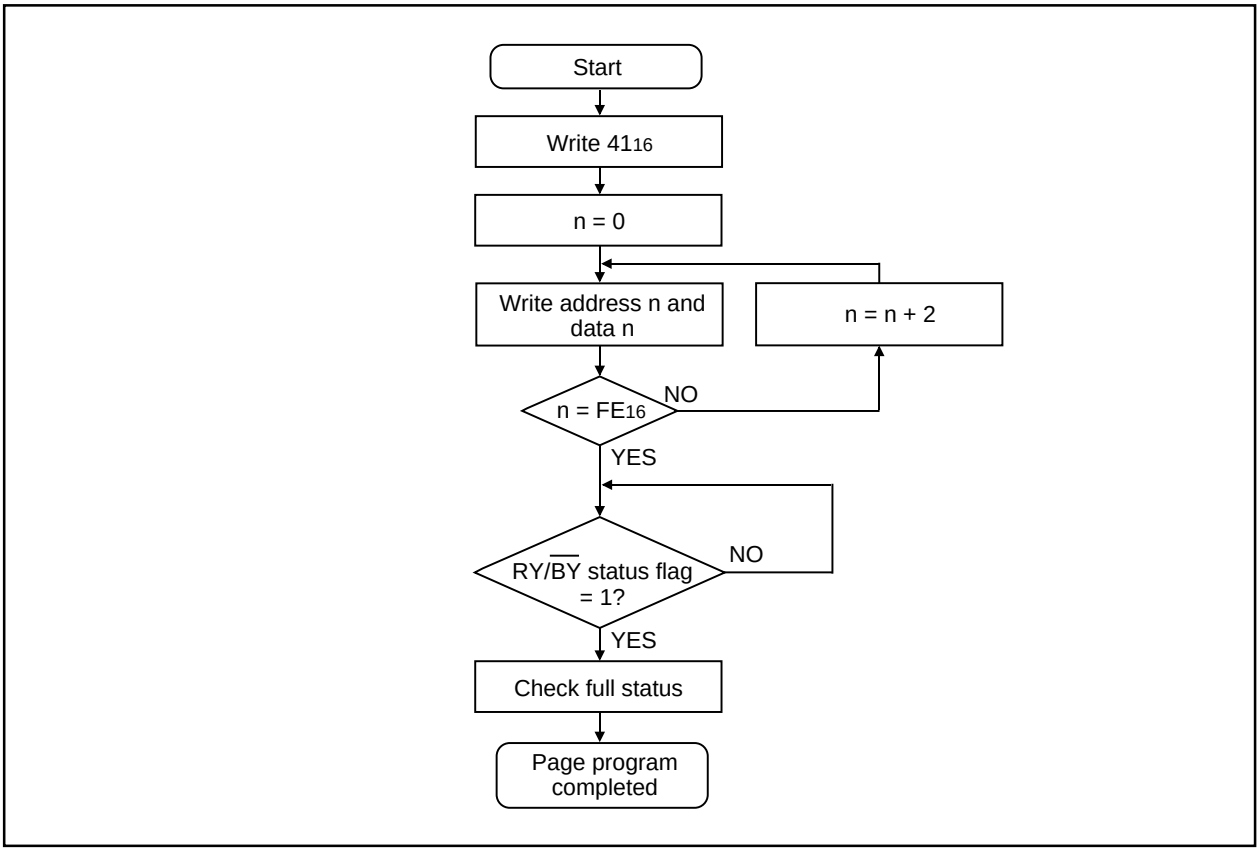


Figure 1.22.4. Page program flowchart

ID Code Check Function

Use this function in standard serial I/O mode. When the contents of the flash memory are not blank, the ID code sent from the peripheral unit is compared with the ID code written in the flash memory to see if they match. If the ID codes do not match, the commands sent from the peripheral unit are not accepted. The ID code consists of 8-bit data, the areas of which, beginning with the first byte, are $0FFFDF_{16}$, $0FFFE3_{16}$, $0FFFEB_{16}$, $0FFFEF_{16}$, $0FFFF3_{16}$, $0FFFF7_{16}$, and $0FFFFB_{16}$. Write a program which has had the ID code preset at these addresses to the flash memory.

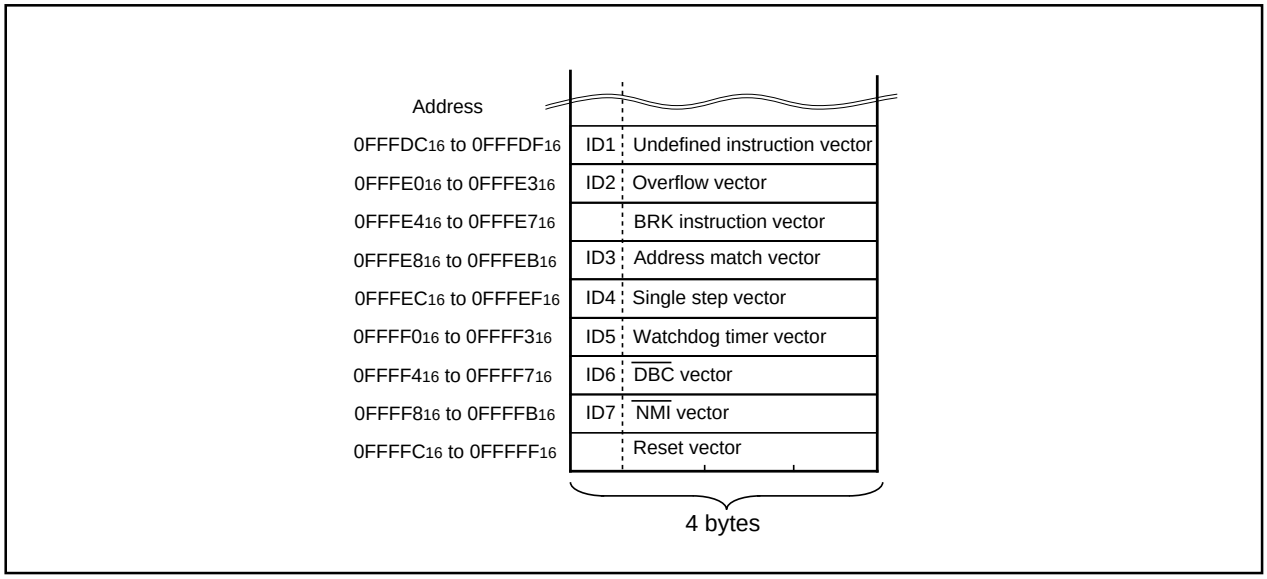


Figure 1.23.2. ID code store addresses

Clear Status Register Command

This command clears the bits (SR3–SR5) which are set when the status register operation ends in error. When the "50_{16}" command code is sent with the 1st byte, the aforementioned bits are cleared. When the clear status register operation ends, the RTS_1 (BUSY) signal changes from the "H" to the "L" level.

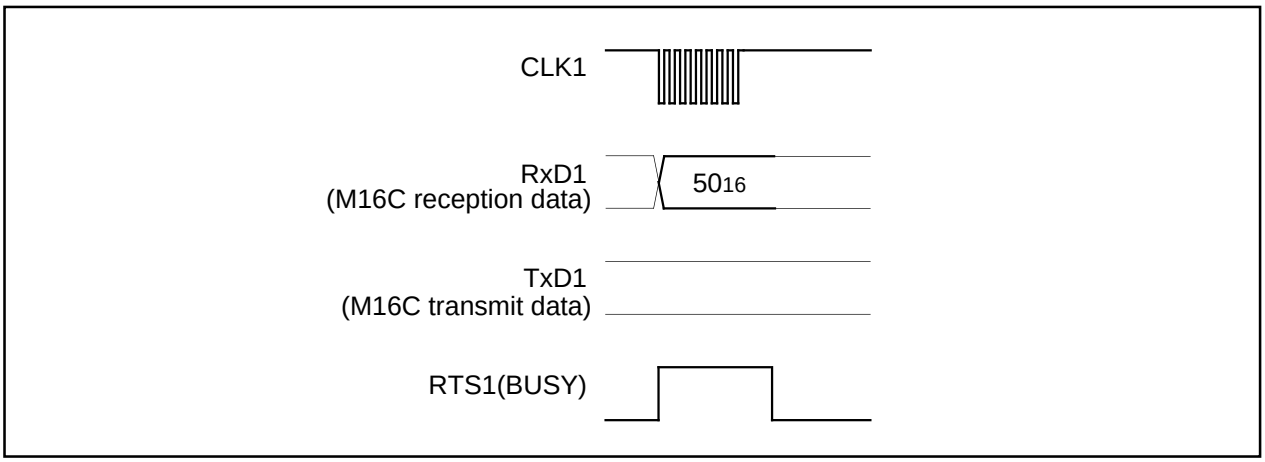


Figure 1.25.4. Timing for clearing the status register

Page Program Command

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

(1) Transfer the "41_{16}" command code with the 1st byte.
(2) Transfer addresses A_8 to A_{15} and A_{16} to A_{23} with the 2nd and 3rd bytes respectively.
(3) From the 4th byte onward, as write data (D_0–D_7) for the page (256 bytes) specified with addresses A_8 to A_{23} is input sequentially from the smallest address first, that page is automatically written.

When reception setup for the next 256 bytes ends, the RTS_1 (BUSY) signal changes from the "H" to the "L" level. The result of the page program can be known by reading the status register. For more information, see the section on the status register.

Each block can be write-protected with the lock bit. For more information, see the section on the data protection function. Additional writing is not allowed with already programmed pages.

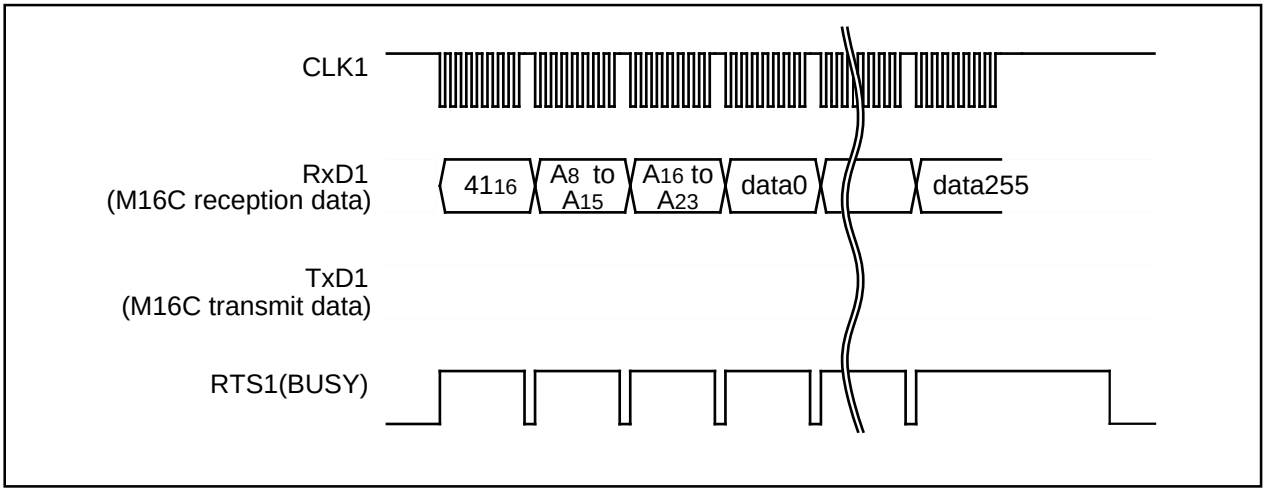
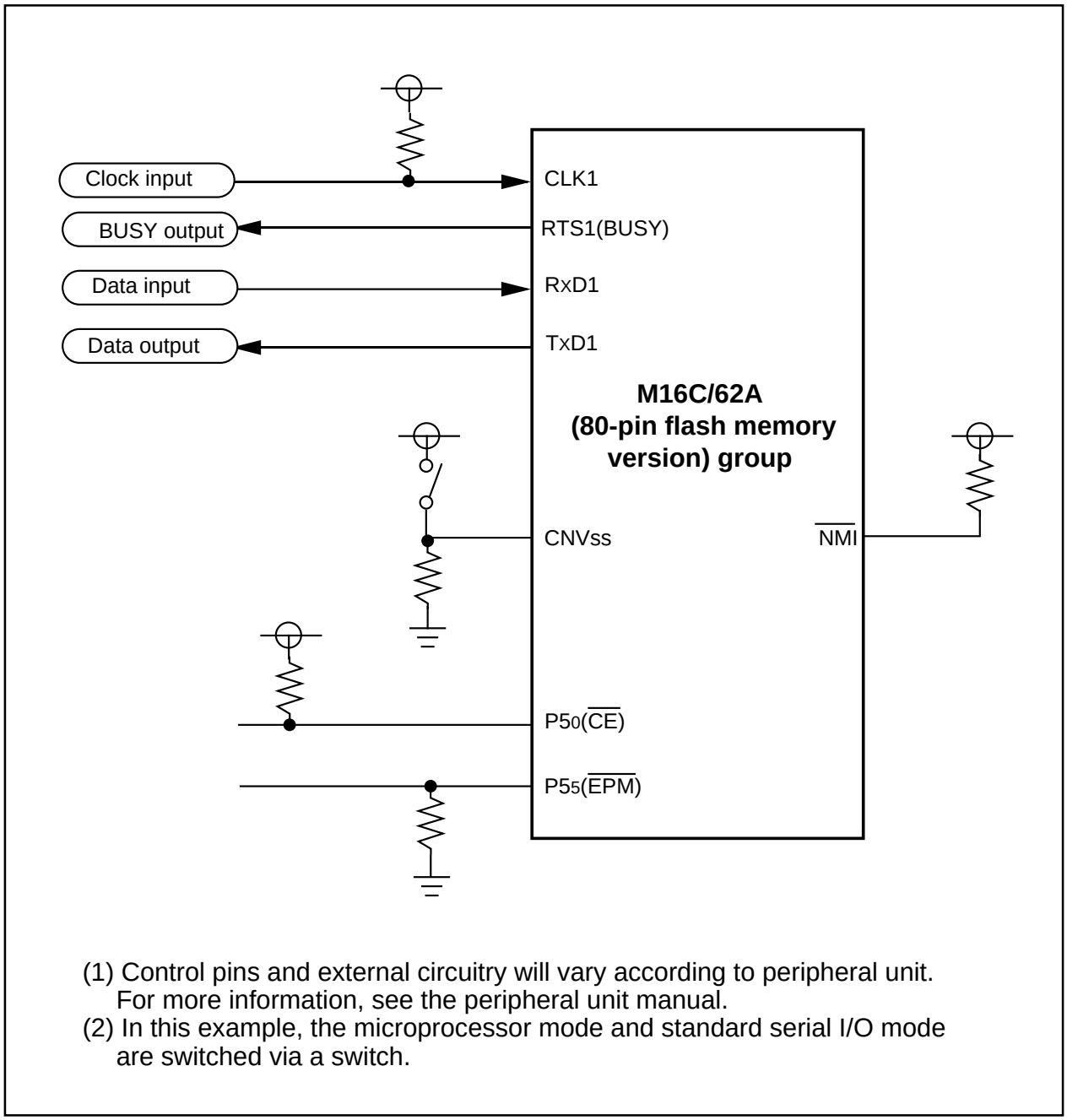


Figure 1.25.5. Timing for the page program

Example Circuit Application for The Standard Serial I/O Mode 1

The below figure shows a circuit application for the standard serial I/O mode 1. Control pins will vary according to programmer, therefore see the peripheral unit manual for more information.



(1) Control pins and external circuitry will vary according to peripheral unit. For more information, see the peripheral unit manual.
(2) In this example, the microprocessor mode and standard serial I/O mode are switched via a switch.

Figure 1.25.20. Example circuit application for the standard serial I/O mode 1

ID Check

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

(1) Transfer the "F5$_{16}$" command code with the 1st byte.

(2) Transfer addresses A$_0$ to A$_7$, A$_8$ to A$_{15}$ and A$_{16}$ to A$_{23}$ of the 1st byte of the ID code with the 2nd, 3rd and 4th bytes respectively.

(3) Transfer the number of data sets of the ID code with the 5th byte.

(4) The ID code is sent with the 6th byte onward, starting with the 1st byte of the code.

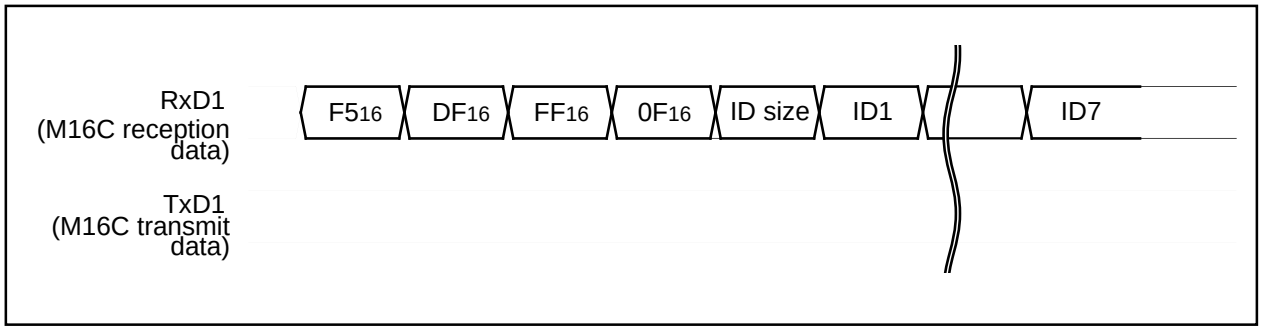


Figure 1.25.35. Timing for the ID check

ID Code

When the flash memory is not blank, the ID code sent from the peripheral units and the ID code written in the flash memory are compared to see if they match. If the codes do not match, the command sent from the peripheral units is not accepted. An ID code contains 8 bits of data. Area is, from the 1st byte, addresses 0FFFDF$_{16}$, 0FFFE3$_{16}$, 0FFFEB$_{16}$, 0FFFEF$_{16}$, 0FFFF3$_{16}$, 0FFFF7$_{16}$ and 0FFFFB$_{16}$. Write a program into the flash memory, which already has the ID code set for these addresses.

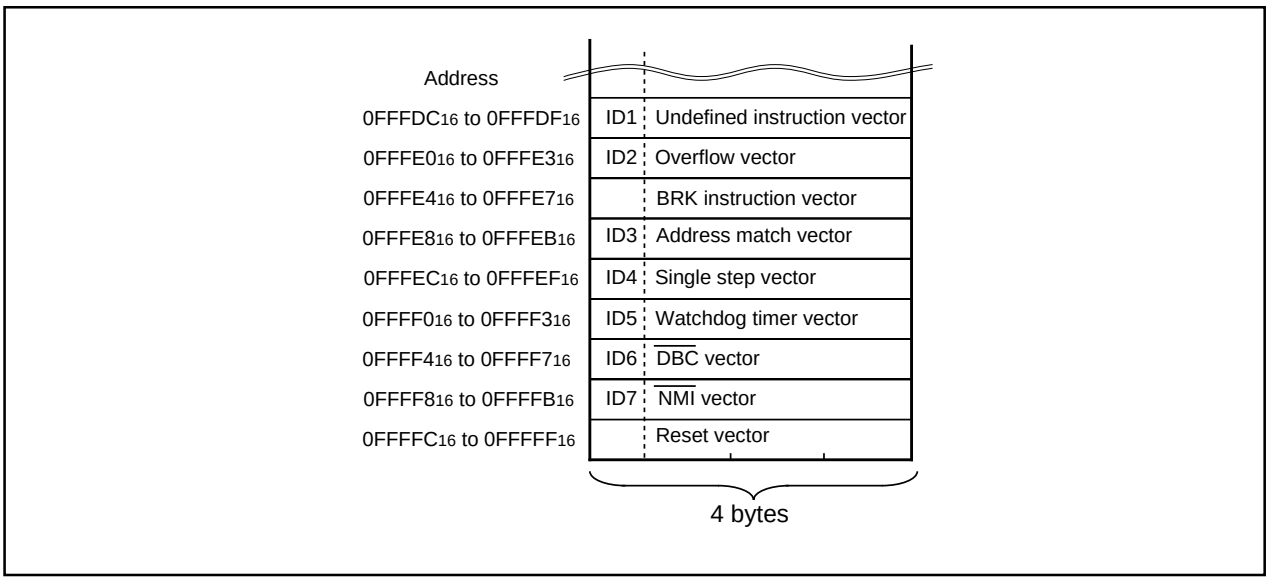


Figure 1.25.36. ID code storage addresses