
8-BIT SINGLE CHIP MICROCONTROLLERS

GMS800 Series Application Note

FAQ

MCU DESIGN GUIDE (Ver. 1.03F)

Mar., 2000



Semiconductor Group of Hyundai Electronics Industrial Co., Ltd.

STOP MODE

Q1 Current consumption is over the Spec. in STOP mode.

A) May set the voltage to all of the pins as well as to not-connected ones to 0V or 5V. Analog input pin and general I/O should be set to 0V or 5V, too. Note that before STOP mode entering, AV_{DD} pin should be asserted 0V or 5V.

Related MCU: All MCUs

Q2 Slight current flows in Open drain port.

A) In GMS81C5032, even though port selection is open-drain and High is output, because input buffer is in floating state, leakage current flows slightly. At this moment if Low is output, the leakage current flow does not happen.

Related MCU: GMS81C5032

Q3 The current of about 70uA flows by R56 and R57 pins in GMS81504.

In case of GMS81504(T) SDIP package, R56(#15), R57(#16) pins have pull-up resistors internally. These pins don't activate the resistors in output mode, but in input mode, they operate with pull-up resistors. To reduce current consumption in input mode, connect R56 and R57 to 5V.

Related MCU: GMS81504T

Q4 In spite of STOP instruction, MCU does not enter into STOP mode.

A) Before completing STOP instruction, clear all interrupt request flags. If interrupt request flag is set before STOP instructions, the MCU runs as if it doesn't perform STOP instruction, even though it already completed them.

Write lines to clear all interrupt request flags (IRQH, IRQL) before STOP instruction as shown in the below example.

After the STOP instruction, you should write two lines of NOP.

Example:

```
:
:
LDM    IRQH, #0
LDM    IRQL, #0
STOP
NOP
NOP
:
:
```

Related MCU: All MCUs

OTP

Q5 In OTP writing, it requires more time comparing to other companies' products.

A) The OTP MCU of GMS800 series is initially designed using Intel's Intelligent mode which uses writing pulse time as 1 ms. Therefore it requires rather more time than Quick pulse mode which uses 0.1 ms.

To satisfy customers' complaints about this, in case of ordering OTPs in large quantities, LGS is ready to supply already programmed OTP from operation.

For more information and an order, please contact to LGS sales office.

GMS87Cxxxx which is recently developed are improved on these problems by using writing pulse of 100 μ s.

Related MCU: GMS81504T, GMS81608T, GMS81516AT, GMS84512T, GMS84524T

Q6 In OTP writing, it may not programmed in standard EPROM 27C256 programming mode.

A) GMS800 series MCU can be programmed by standard EPROM Programmer(Writer). But GMS800 series OTP may not programmed because it is designed with "Edge Detect", different to standard EPROM 27C256 mode whose \overline{OE} pins are "Level Detect". If OTP write doesn't work in standard 27C256 mode, we recommend you to use equipment registered to be compatible with relevant MCU instead of standard EPROM 27C256.

GMS815xxBT which will replace GMS815xxAT will improve in these problems. And LGS will provide private single and gang programmer for GMS87C1202,1408, and etc.

Recommended Equipment: ALL-11, GANG-08

Related MCU: GMS81504T, GMS81608T, GMS81516AT, GMS84512T, GMS84524T

Q7 R60 pin is set as input pin, but it doesn't be read as either Low or High.

A) One pin of R60~R67 should be set as ADC input. In this case, pin which is selected as ADC input doesn't be input digitally. Set the R60 pin as digital input by changing ADCM register value to other value. At MCU's initial power-on, R60 pin is set as ADC input.

But GMS81504T has no R60~R63 pins. In case of using R64 pin as ADC input, digital input doesn't work.

Related MCU: GMS81504T, GMS81608T, GMS81516AT

Q8 Buzzer works well in Emulator, but after OTP MCU setting it doesn't.

A) Set R55 pin to output mode. Few differences between emulator and OTP MCU cause this trouble, therefore regardless of OTP or MASK MCU, set pins to output mode by PMR register.

Related MCU: GMS81504, GMS81504T

Q9 It works well in Emulator, but after OTP device setting, it doesn't work at all.

A) First, codes STACK and User memory area so that they do not conflict but share each other.

For example, addresses from 0_H to 5F_H should be coded as User memory area and from 60_H to 7F_H as STACK area. But if data area invades STACK area over 5F_H or STACK area invades data memory area over 60_H, this program doesn't run.

Even though these two parts are programmed to share each other, the Emulator with GMS815/6xx EVA works well because it uses page 1 as STACK area, but when it runs as OTP device, it uses page 0 that makes conflictions of data between STACK area and User memory area. For this reason, in case of GMS81504T, much consideration should be given.

Secondly, STACK pointer must be initialized. In condition that STACK pointer is not assigned, program may not run.

In GMS81504, we recommend you to set STACK pointer to 7F_H. Without initializing STACK pointer, program works well in the Emulator, but in OTP device and Masked device it doesn't. For more detailed information, please refer to "Q25".

GMS81508A/GMS81516A are recommended to set STACK pointer to FE_H.

For other MCU cases, see the below table.

Device Name	STACK pointer setting
GMS81608	3F _H
GMS87C1202	7F _H
GMS87C1408	BF _H

Thirdly, check the states of reset circuit, oscillation circuit, and test pin.

Is Reset wave form applied properly?

Does oscillator work well ?

Is TEST pin connected to 5V ? GMS81504, 81604, 81608 have TEST pins.

In Emulator operation, the TEST pin states of reset circuit and oscillator circuit in target system don't affect the operation at all. But in OTP and MASK MCU operation, these circuits are needed.

The GMS87C1xxx emulator is connected to the reset circuit of target system.

Related MCU: All MCUs

FUNCTION

Q10 TM0, TDR0, IE are set related to "Timer0", but interrupt doesn't occur correctly.

A) Check the registers and flags below.

TM0 Register: Check if TIMER clock source is set to internal clock in T0SL[1:0]
 Check CAP0 bit is cleared to "0". In capture mode, interrupt doesn't occur.
 Check T0ST bit is set to "1".
 Check T0CN is set to "1". If it is "0", clock isn't supplied.

TDR0 : Write appropriate value from 01H to 0FFH.

IENH,IENL : Check if the bit to enable TIMER0 is set to "1".

The registers below which are related to timer interrupt should be written in appropriate values. The following are the examples. Set the value properly in accordance with user's system.

Example:

```

:
:
LDM    TM0 ,#7FH
LDM    TDR0 ,#250
LDM    IENH ,#0FFH
EI
:
:

```

Related MCU: All MCUs

Q11 Timer/Event counter is used as CAPTURE mode. There is a signal on interrupt pin, but the capture data has a strange value.

A) Check if external interrupt pin and TIMER match correctly. If INT0 pin is used, TIMER 0 will be captured, if INT2 pin is used, TIMER2 will be captured. When using INT0, if you read the TIMER2 capture value, that value is unrelated to INT0.

In capture mode, timer interrupt doesn't occur. (external interrupts still occur normally.)

Related MCU: All MCUs

Q12 Pin is set to output mode, but output data value doesn't appear on the pin.

A) Check the PMR register. Each pin of MCU shares and uses overlapping not only general I/O functions but also buzzer activation, external interrupt input, timer division clock output. The selection of these alternated functions is determined by Port Mode Register Setting corresponding to each port. For example, even though the pin is set to output mode, if the external interrupt function is selected by Port Mode Register, automatically, it operates as an input.

Related MCU: All MCUs

Q13 In case of using thermistor to ADC pin, an open or short is detected unstably.

A) In short or open case, ADC input voltage is 0V or 5V, and ADC value is considered 00_H or FF_H respectively. But this value is an ideal theoretical value. Actually they are read slightly differently as 00_H~03_H/FD_H~FF_H because of ADC errors and power ruffle. Therefore it is better to judge short or open with some margins. Instead of ADC value 00_H, set some margins comparing it with 05_H. The value F0_H is reasonable with same margin for FF_H.

Related MCU: GMS8xC1102, GMS8xC1202, GMS8xC1404, GMS8xC1408, GMS81504(T), GMS81604, GMS81608(T), GMS81508A, GMS81516A(T)

Q14 The result of ADC isn't correct. It seems that 8-bit resolution error occurs.

A) The error of ADC result can be out of the guaranteed range according to the power condition of AV_{DD} pin and ADC input pin in ADC. Check the ruffle content of power. In ADC use, voltage regulation and power ruffle filter circuit should be concerned. It can be improved by adding the bypass capacitor between pin and circuit ground. It is recommended to give ADC pin about 0.01uF~0.1uF. Also it is recommended to use the average value by reading values several times.

Related MCU: GMS8xC1102, GMS8xC1202, GMS8xC1404, GMS8xC1408, GMS81504(T), GMS81604, GMS81608(T), GMS81508A, GMS81516A(T)

Q15 It runs only for about 2~3 seconds after power-on and repeats resets.

A) This is software error related to Watchdog timer.

First, in using Watchdog timer, it should be programmed that Watchdog Timer Counter Register is cleared every time by the shorter cycle than its cycle.

Secondly, byte manipulation instruction should be used in writing certain value to CKCTLR (Clock Control Register). If bit manipulation instruction is used, other 7 bits except for the written one could be changed with undesirable values.

Registers of MCU has two kinds, one is Write-only registers, another is Read/Write registers. Write-only registers are controlled by only byte manipulation instruction, Read/Write registers are controlled by both byte and bit manipulation instruction.

Example:

Register name	Usage Example
CKCTLR	LDM CKCTLR, #33H
TM0	LDM TM0, #23 SET1 TM0.3

Related MCU: All MCUs

Q16 At PWM output start command, one first pulse, i.e. one initial output pulse outputs

abnormally.

A) After stopping PWM timer clock, set cycle and duty register value. If user writes register values while timer is in operation, these register could be set with certain values.

Related MCU: GMS8xC1102, GMS8xC1202, GMS8xC1404, GMS8xC1408, GMS81508A, GMS81516A(T)

MISCELLANEOUS

Q17 What is Impulse Noise Immunity (EMS) of power in electric products?

A) Generally, electric products and power generate lots of noises by rapid control of large amount of current due to motor or arc electric discharge. MCU should be properly designed to prevent malfunction by the noise. Of course, removing noises should be done by various filters and MCU itself should have basically more noise tolerance to give more help to system manufacturers. First generation MCUs of GMS800 series have been pointed out that they should be improved.

Currently produced and developed MCUs are designed to have more EMS tolerance by considering power circuit design from chip layout stage, we put all our experience and research that have made in order to reduce customers' worries about impulse power noise in using our MCU.

As a result, actually the LG Electronics company's products which highly requires electrical noise tolerance such as laundry machine, air conditioner, induction rice cooker, ultraviolet sterilizing dish drier are being produced without any defects.

The production of GMS81516 which has somewhat bad impulse noise immunity is terminated and currently GMS81516A which has advanced noise immunity is supplied. GMS81516B which will be available at the end of 1999 will have much improved noise immunity.

Related MCU: All MCUs

Q18 In the development stage, all test items are passed with OTP, but with Masked MCUs they aren't.

A) The OTP and Masked MCU of LGS are almost equal in their characteristics because they are processed with the same design rule. Till now in production, OTP and MASK in LGS MCU show the good results similarly. Sometimes, in other companys' cases, development tests with OTP are all passed but with MASK MCU aren't. We know that it must be really embarrassing from developer's standpoint.

Related MCU: All MCUs

Q19 Oscillation is unstable. The design idea to run it more stably is needed.

A) Oscillation circuit uses general crystal oscillator or ceramic resonator. In PCB Artwork, layout of two lines should be symmetrical, and the shorter distance, the better. It is better to wrap the surroundings with ground pattern. The PCB should be designed that there is no other signal line which intercourses this line layout.

Related MCU: All MCUs

Q20 How to drive the pin connected to LED and 200 Ω resistor via GND? High level output voltage is only 3V.

A) The design should be changed for pin to run active low. GMS800 series MCU has strong sink capability rather than current source capability. If you turn it on with low level output, then even with 15mA current, pin voltage doesn't go up more than 1.0 V.

Related MCU: All MCUs

Q21 Are there any frequency data according to RC value in RC Oscillation of GMS87C1202?

A) The below RC Oscillator data are real measured data. To get Oscillator frequency, refer to the below data with the control of R and C values. The result of MASK MCU will be informed later in the revised and enlarged data book.

X_{OUT} is the value of X_{IN} divided into 4, therefore F_{OSC} is 4 by X_{OUT} value.

* RC Oscillation value

- at $V_{CC} = 5V$, Cap. = 24pF

Resistor	Xout	System Frequency($X_{out} \times 4$)
4.7K	547kHz	(547*4) kHz
6.2K	805kHz	(805*4) kHz
10K	795kHz	(795*4) kHz
20K	500kHz	(500*4) kHz
30K	360kHz	(360*4) kHz
47K	238kHz	(238*4) kHz
82K	145kHz	(145*4) kHz

- at $V_{CC} = 5V$, Cap. = 30pF

Resistor	Xout	System Frequency($X_{out} \times 4$)
4.7K	497kHz	(497*4) kHz
6.2K	731kHz	(731*4) kHz
10K	700kHz	(700*4) kHz
20K	450kHz	(450*4) kHz
30K	326kHz	(326*4) kHz
47K	215kHz	(215*4) kHz
82K	132kHz	(132*4) kHz

- at $V_{CC} = 3V$, Cap. = 7pF

Resistor	Xout	System Frequency($X_{out} \times 4$)
8.2K	376kHz	(376*4) kHz
10K	569kHz	(569*4) kHz
15K	595kHz	(595*4) kHz
20K	536kHz	(536*4) kHz
30K	420kHz	(420*4) kHz
47K	292kHz	(292*4) kHz
82K	185kHz	(185*4) kHz

- at $V_{CC} = 3V$, Cap. = 15pF

Resistor	Xout	System Frequency($X_{out} \times 4$)
8.2K	200kHz	(200*4) kHz
10K	430kHz	(430*4) kHz
15K	456kHz	(456*4) kHz
20K	413kHz	(413*4) kHz

30K	325kHz	(325*4) kHz
47K	226kHz	(226*4) kHz
82K	143kHz	(143*4) kHz

Related MCU: GMS87C1102, GMS87C1202

Q22 How to design the Reset circuit with RC?

A) For secure Reset, we recommend to use Reset IC.

When designing Reset circuit with RC, users should firsthand control R and C values within the recommended values below. However, Reset may not work in instant-power-on/off-very-fast-repetition test.

(Generally 10 kΩ, 10 uF will be enough.)

- Use the Reset IC(KIA7042)
- RC Reset circuit ---> R=1 kΩ~10 kΩ, C=4.7 uF~10 uF

Q23 Glitch pulses appear on LCD Segment pin.

A) The glitch pulses occur when software writes data repeatedly by a cycle shorter than 10 ms at LCD display memory (RAM address: 0C00_H~0C4F_H). It will be removed by refreshing a period at a cycle larger than 10ms or instead by writing data. The reason is that LCD display memory is designed to read/write at the same time.

Related MCU: GMS81C3004

Q24 Brightness of each segment in LCD display panel is not equal.

A) In the KeyScan function, on account of Keyscan strobe signal to SEG. pin, display panel is affected. In this case, using external bias resistance to V_{CL} pin will reduce the influence. The resistance value of under 15 kΩ is recommended according to the test results. But considering current consumption, the most perfect way is to disable all keyscan functions and then make a software keyscan routine by using general I/O.

Related MCU: GMS81C3004

Q25 Designing the GMS81504 application system requires some cautions on STACK usage.

A) If data is lost according to the collision of data memory and STACK memory area, then system doesn't work properly. Therefore users should be careful about this.

1. STACK pointer should be initialized. Otherwise, OTP and MASK MCU except for Emulator can have abnormal operations at every power on.

It's because Emulator doesn't make any collision since stack and user data areas use different pages, which is different from Masked or OTP MCU.

2. Sometimes subroutine call is used for clearing all RAM data, then first STACK pointer should be initialized before the execution of subroutine clearing STACK area.

The below is wrong example usage.

Example: Wrong example

```

                ORG          0F000H          ;Program Start Address
RESET:         DI           ;Disable All Interrupts
                ;
                LDX          #7FH           ;Stack Pointer Initialize
                TXSP        ;SP. <- #7FH, Set upper 8 Bits to "0"
                CALL        RAM_CLR
                :
                :

RAM_CLR:      LDX          #0
CLR_Lp:      LDA          #0              ;RAM Clear(!0000H->!007FH)
                STA          {X}+        ;M(X) <- A, then X <- X+1
                CMPX       #080H        ;X = #080H ?
                BNE        CLR_Lp
                RET

```

The above program malfunctions because the address to return is cleared by RAM_CLR routine operation.

The problem is that even in this case the emulator works well. The reason is the Emulator uses 100H~1FEH as STACK area which accordingly prevents the collision. At this point, the Emulator and OTP device have a difference.

There will be no problem if you set the program as follows.

Example: Correct example

```

                ORG          0F000H          ;Program Start Address
RESET:         DI           ;Disable All Interrupts
                LDX          #0
RAM_CLR:      LDA          #0              ;RAM Clear(!0000H->!007FH)
                STA          {X}+        ;M(X) <- A, then X <- X+1
                CMPX       #080H        ;X = #080H ?
                BNE        RAM_CLR
                ;
                LDX          #7FH        ;Set Stack Pointer
                TXSP        ;SP. <- #7FH, Fixed upper 8 Bits to "0"
                :
                :

```

Related MCU: GMS81504, GMS81504T

Q26 For the GMS81516A, it is recommended to set initial value of STACK pointer to FE_H address, not FF_H.

A) The above MCUs have reserved 1FF_H address memory. So, set the initial value as FE_H.

Example:

```

                ORG          0C000H          ;Program Start Address
RESET:         DI           ;Disable All Interrupts

```

```

RAM_CLR:      LDX      #0
              LDA      #0                ;RAM Clear(!0000H->!00BFH)
              STA      {X}+             ;M(X) <- A, then X <- X+1
              CMPX    #0C0H            ;X = #0C0H ?
              BNE     RAM_CLR
              ;
              LDX     #0FEH            ;Stack Pointer Initial
              TXSP    ;SP. <- #0FEH, Set upper 8 Bits to "01H"
              :
              :

```

Related MCU: GMS81508A, GMS81516A, GMS81516AT

Q27 What's the purpose of GMS81516A MP(#2) pin?

GMS81516 is designed to run as Microprocessor mode in the first stage of design. But currently produced GMS81516A does not support this function and only pin remains called \overline{MP} , used to connect to V_{DD} .

Q28 How to store 16-bit data to RAM by using "LDM" instruction in coding?

A) Input them using "<", ">" mark.

Example 1:

```

LABEL  DS    2
        LDM  LABEL, #<560      ; Lower byte 30H of 560(0230H) is stored in LABEL.
        LDM  LABEL+1, #>560   ; Upper byte 02H is stored in LABEL+1.

```

Example 2:

```

TEMPA  EQU  <560              ; Assign lower byte 30H of 560(0230H) at TEMPB
        LDM  TEMPB, #TEMPA

```

Q29 How to assemble and link modulated sources which are made of more than 2 sources in GMS800 Series?

A) Assembler and Linker are used under DOS environment, so only conventional memory is used. Therefore program source file becomes large, then Assemble and Link can not be completed as one file. The following is how to code by dividing source files into several modules.

(For more details, please refer to MDS manual.)

1. Assigning variables

```

Public symbol, label, ...      ;Assign Symbol, Label, Constant, and Bit
                               which affect external files.

Extrn Constant symbol1, symbol2, ... ;Assign constant defined as EQU
                                   (symbol1 equ 0C0h) at external file if it is used in
                                   current file. (without distriiction of capital
                                   and small letter)

```

```

Extrn Label label1,label2,...      ;Assign LABEL(label1:) of external file is used
                                   in current file. (without distiction of capital
                                   and small letter)

Extrn Bit symbol3,symbol4,...      ;Assign defined Bit as EQU(symbol4 EQU 0, symbol5)
                                   at external file if it is used it current file.
                                   (without distiction of capital and small letter)

Extrn Ram symbol5, symbol6,...      ; Assign variable defined as DS(symbol5 DS 1)
                                   at external file if it is used it current file.
                                   (without distiction of capital and small letter)

Extrn Ram1 symbol7, symbol8,...     ; Assign defined variable as DS1(symbol7 Ds1 1)
                                   at external file if it is used it current file.
                                   (without distiction of capital and small letter)

```

2. How to Assemble and Link.

Lets make an example with FILE1.ASM and FILE2.ASM.

- Assemble files separately. *.OBJ files are created respectively.

```

C:\XASM8 FILE1.ASM [ENTER]
C:\XASM8 FILE2.ASM [ENTER]

```

- Link these 2 files as one unit.

```

C:\XLINK8 FILE1.OBJ FILE2.OBJ /CPU=815xx [ENTER]

```

Q30 Be careful in controlling PMR during external interrupt routine.

A) During the interrupt routine process after receiving of an external interrupt, Interrupt pin may be changed to general pin by controlling PMR in order to read and check the port state in this routine.

In this case, an interrupt request flag occurs again, therefore this falg should be cleared by software. In Emulator, this kind of case doesn't occur regardless of clearing the interrupt request flag.

Example 1 : Not using EI in the interrupt routine.

```

INT0:      :
           LDM      PMR4,#0          ;INT0 -> R40
           CLR1     INT0R            ;Clear interrupt request Flag
           TST1     R4DD.0          ;Test R40 port
           :
           :
           LDM      PMR4,#1          ;Return to INT0
           CLR1     INT0R            ;Clear interrupt request Flag
           RETI

```

Example 2. : Using EI in the interrupt routine.

```

INT0:      :
           LDM      PMR4,#0          ;To read, change INT0 to R40
           CLR1     INT0R            ;Clear interrupt request Flag
           EI

```

```

TST1      R4DD.0      ;Test R40 port
:
:
DI
LDM       PMR4,#1     ;Return to INT0
CLR1      INT0R       ;Clear interrupt request Flag
RETI

```

Related MCU: GMS81508A, GMS81516A, GMS81516AT

Q31 Be careful when the operation is something wrong in using BMI or BPL instruction after CMP instruction.

A) The BCC or BCS is recommended to use after the CMP instruction. It is proper to use BMI or BPL instruction in order to jump when the count changes from FF_H to 00_H or from 00_H to FF_H in increment or decrement routine. AND the BEQ instruction is used when the count changes from 1 to 0 in decrement routine.

After the CMP instruction, the MSB of accumulator is used for negative flag.

Example 1: Good usage, after subtraction the carry flag is set(C=1).

```

:
LDA       #0A0H      ;M(VALUE) --> Acc.
CMP       #2H        ;Clear interrupt request Flag
BCC       MINUS
:
:
:PLUS
MINUS:    :
:MINUS

```

Example 2: Wrong usage, the result of calculation is 9EH, the negative flag is set(N=1).

```

:
LDA       #0A0H      ;M(VALUE) -> Acc.
CMP       #2H
BMI       MINUS
:
:
:PLUS
MINUS:    :
:MINUS

```

Related MCU: All MCU

Application Design of Reset Circuit

When the power is supplied firstly, the stable and accurate reset circuit is need to be organized to operate the MCU. Refer to below contents when designing MCU circuit.

1. Auto Reset Circuit

The below reset circuit explains power-on reset and auto reset operation at $V_{CC} < V_{TH}$.

(V_{TH} : Threshold level)

1.1 Standard Circuit

The figure 1.1 is the standard of auto reset circuit.

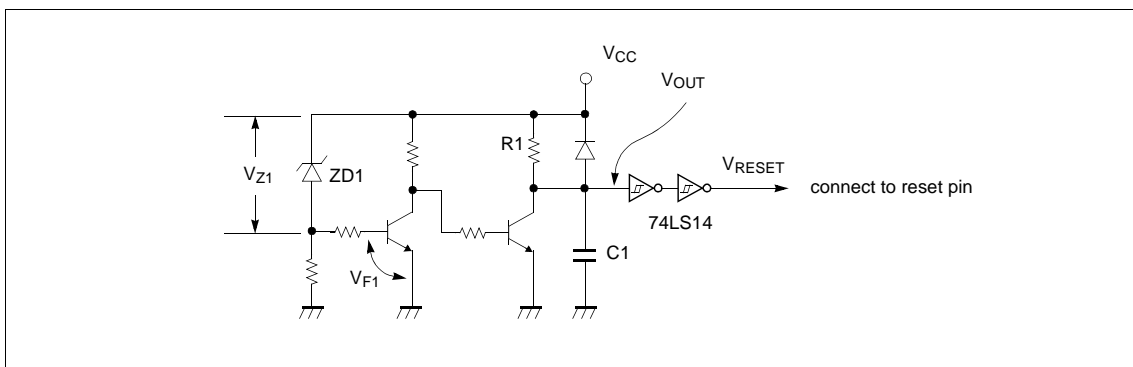


Figure 1-1 Standard Circuit of Auto Reset

1.2 Explain of Operation

Power-on reset

In Figure 1-2, the V_{OUT} is the same as the voltage charged in $C1$ through $R1$. The rising rime of V_{OUT} is decided by charging time($C1 \times R1$) and power-on reset is operated.

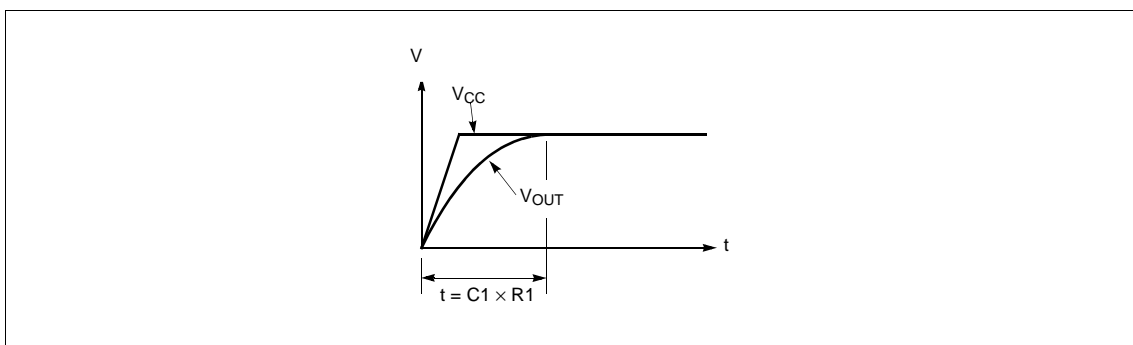


Figure 1-2 Pulse relation of V_{CC} and V_{OUT}

Auto Reset

In Figure 1-1, the auto reset circuit operation is explained like below.

The voltage V_{Z1} supplied to ZD_1 and the turn-on voltage V_{F1} of TR_1 are kept uniform regardless of V_{CC} change. And the V_{OUT} is decided out of following several fomulas.

$$V_{OUT} = V_{CC} \text{ if } V_{CC} > V_{Z1} + V_{F1}$$

$$V_{OUT} = \text{GND} \text{ if } V_{CC} < V_{Z1} + V_{F1}$$

Output pulse of V_{OUT}

If the V_{CC} descends under V_{TH} , V_{OUT} is low and if the V_{CC} ascends over the V_{TH} , V_{OUT} is high.

For this reason, the stable reset operation can be expected. Refer to Figure 1-3.

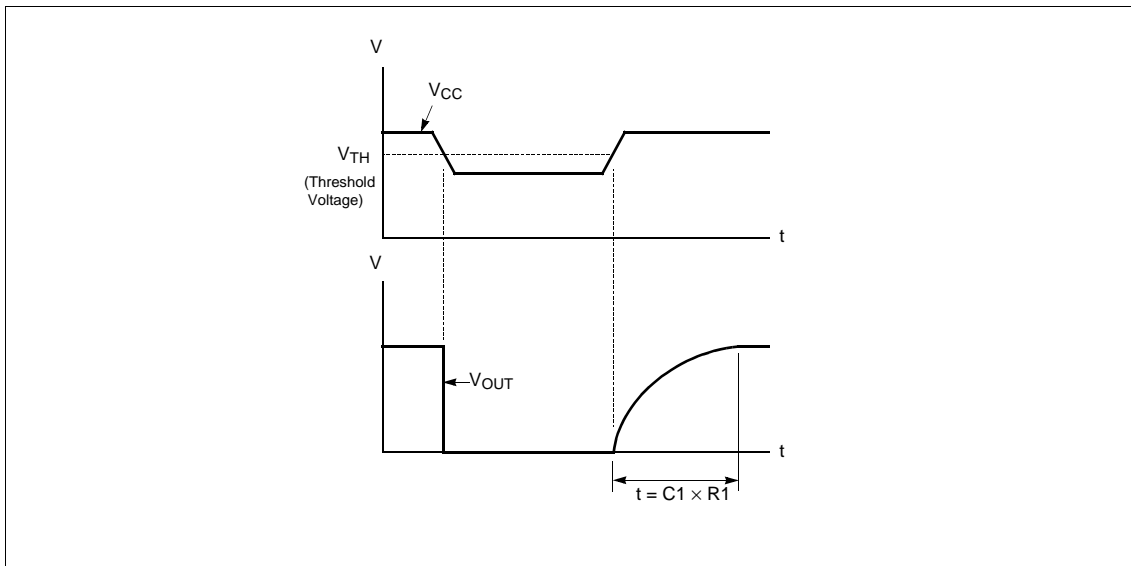


Figure 1-3 V_{OUT} in Auto Reset Operation

2. Manual Reset Circuit

When the MCU is reset by the external reset switch, the switch chattering should not be occurred.

The Figure 2-1 is chattering protection circuit.

2.1 Standard Circuit

The Figure 2-1 is standard circuit of manual reset.

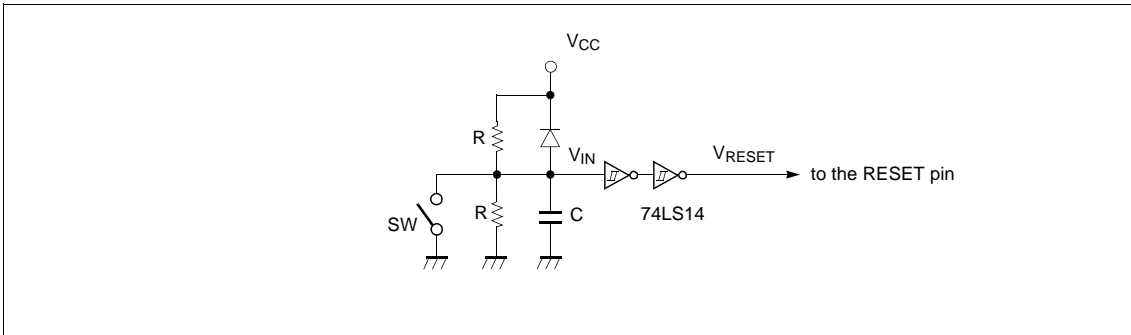


Figure 2-1 Standard Circuit of Manual Reset

Operation of Power-on reset

During power-up, the rising time of V_{IN} charged on condenser C is decided by charging time($C \times R$) like as Figure 2-2. And the MCU is reset by V_{IN} .

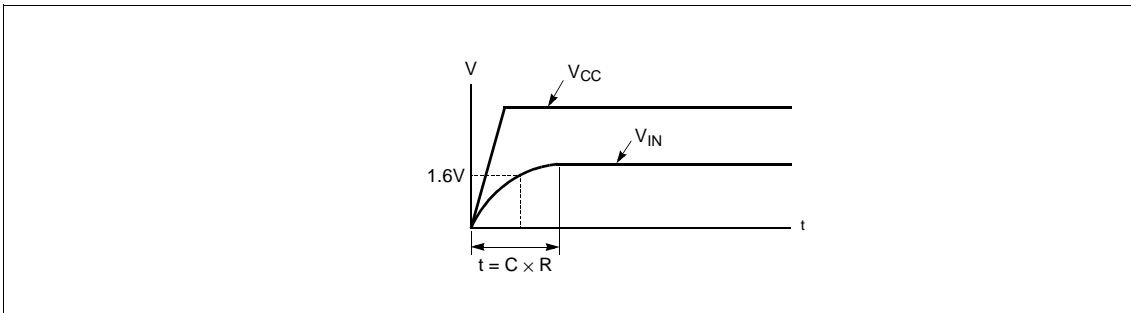


Figure 2-2 V_{IN} in Power-on Reset Circuit

Operation of Manual Reset

If the switch is on, the V_{RESET} is the same as GND level and the MCU is reset. If the switch is off, the condenser starts charging, and the rising time of charged voltage V_{IN} is decided by charging time($C \times R$). The condenser and Schmitt trigger IC remove the chattering. Refer to Figure 2-3.

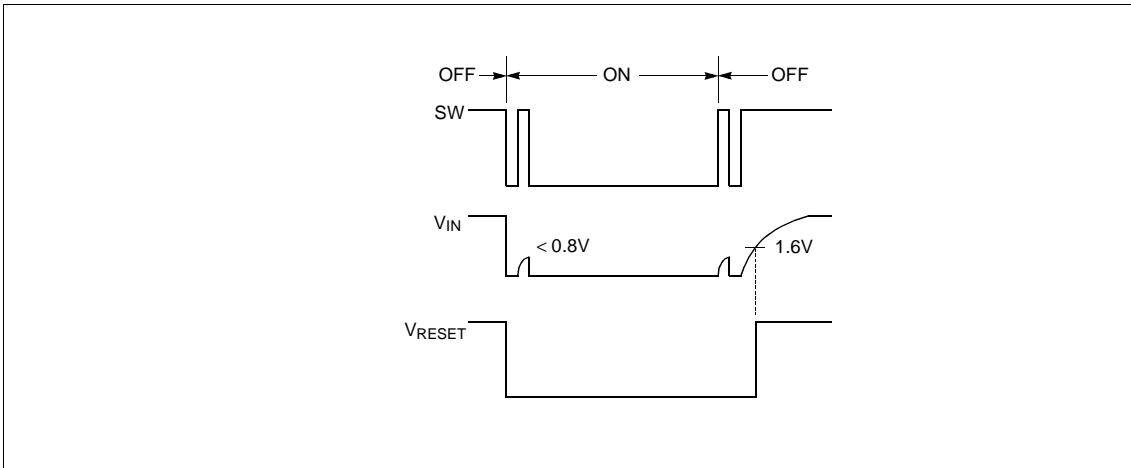
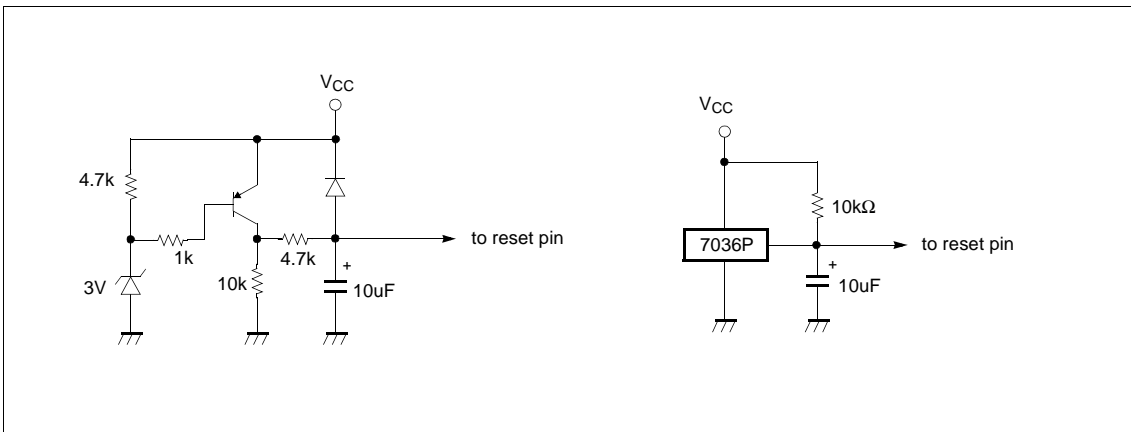


Figure 2-3 Timing Chart of Manual Reset

When the switch changes from Off to On, $V_{RESET} = GND$ at $V_{IN} < 0.8V$,
 from On to Off, $V_{RESET} = V_{CC}$ at $V_{IN} > 1.6V$.

3. Example of Practical Application

The left figure represents the reset circuit composed of zener diode and Tr. Cap., the right figure is the reset circuit using reset IC.



CONTACT PERSONS

The persons in charge of each device of GMS800 series MCU are listed below. If you have any question, please contact us by E-mail or phone.

FACSIMILE: 82-431-270-4075

GMS815xx General Purpose MCU	Hwang Sung Jae	82-431-270-4071	hsjd@hmec.co.kr
GMS87C1xxx Small MCU	Lee Jong Hwan	82-431-270-4072	jjong@hmec.co.kr
GMS90 MCS-51 compatible MCU	Yoon Kyung Sang	82-431-270-4082	ksyun@hmec.co.kr
GMS81C3xxx LCD MCU	Jung Dong Su	82-431-270-4081	jdsoo@hmec.co.kr
GMS84xxx TV MCU	Lim Byung Jin	82-546-470-2249	bjinlim@hmec.co.kr
GMS810xx Remocon MCU	Kang Myung Jin	82-431-270-4078	kanghan@hmec.co.kr
MDS (Emulator)	Ha Seung Duk	82-431-270-4073	haseungduk@hmec.co.kr
MCU Sales	Park Dong Hee	82-2-3459-3665	david@hmec.co.kr

HYUNDAI MicroElectronics Co., Ltd
MCU APPLICATION DESIGN TEAM. MAR., 2000

INDEX TABLE

FAQ on Functions

FUNCTION ITEMS	QUESTION ITEMS
General	Q1 Q4 Q9 Q12 Q17 Q18 Q19 Q20 Q22
ADC	Q13 Q14
Timer	Q10 Q11 Q15
Assembler	Q28 Q29

FAQ on Device

DEVICE NAME	QUESTION ITEMS
GMS81C1102 GMS87C1102	Q16 Q16 Q21
GMS81C1202 GMS87C1202	Q16 Q16 Q21
GMS81C1404 GMS87C1404	Q16 Q16
GMS81C1404 GMS87C1404	Q16 Q16
GMS81C3004	Q23 Q24
GMS81C5032 GMS87C5032	Q2 Q2
GMS81504 GMS81504T	Q3 Q7 Q8 Q25 Q3 Q5 Q6 Q7 Q8 Q25
GMS81508A GMS81516A GMS81516AT	Q7 Q26 Q27 Q30 Q7 Q26 Q27 Q30 Q5 Q6 Q7 Q26 Q27 Q30
GMS81524B GMS81524BT	Q5 Q6
GMS81604 GMS81608 GMS81608T	Q7 Q7 Q5 Q6 Q7
GMS84512A GMS84512AT	Q5 Q6
GMS84524A GMS84524AT	Q5 Q6