# SN8PC13

## USER'S MANUAL

Version 0.5

# SONiX 8-Bit Micro-Controller

## AMENDMENT HISTORY

| Version | Date | Description |
|---|---|---|
| VER 0.1 | Sep. 2005 | 1. First issue |
| | Nov.2005 | 1. ADD Brown-Out reset circuit. |
| VER 0.2 | Dec 2005 | 1. Modify Topr value.<br>2. Modify SN8PC13_V02 Brown-Out Reset description<br>3. Remove power consumption(Pc)<br>4. Modify M2IDE 1.07.<br>5. Modify ELECTRICAL CHARACTERISTIC. |
| VER 0.3 | Jan 2006 | 1. Remove P5.4.<br>2. Modify IR output description. IROUT pin is low status when IROUT=0 or the system is in power down. |
| VER 0.4 | May 2006 | 1. Modify working voltage range to LVD~5.5V. |
| VER 0.5 | Nov. 2006 | 1. Add Marking Definition. |

# Table of Content

# 1 PRODUCT OVERVIEW

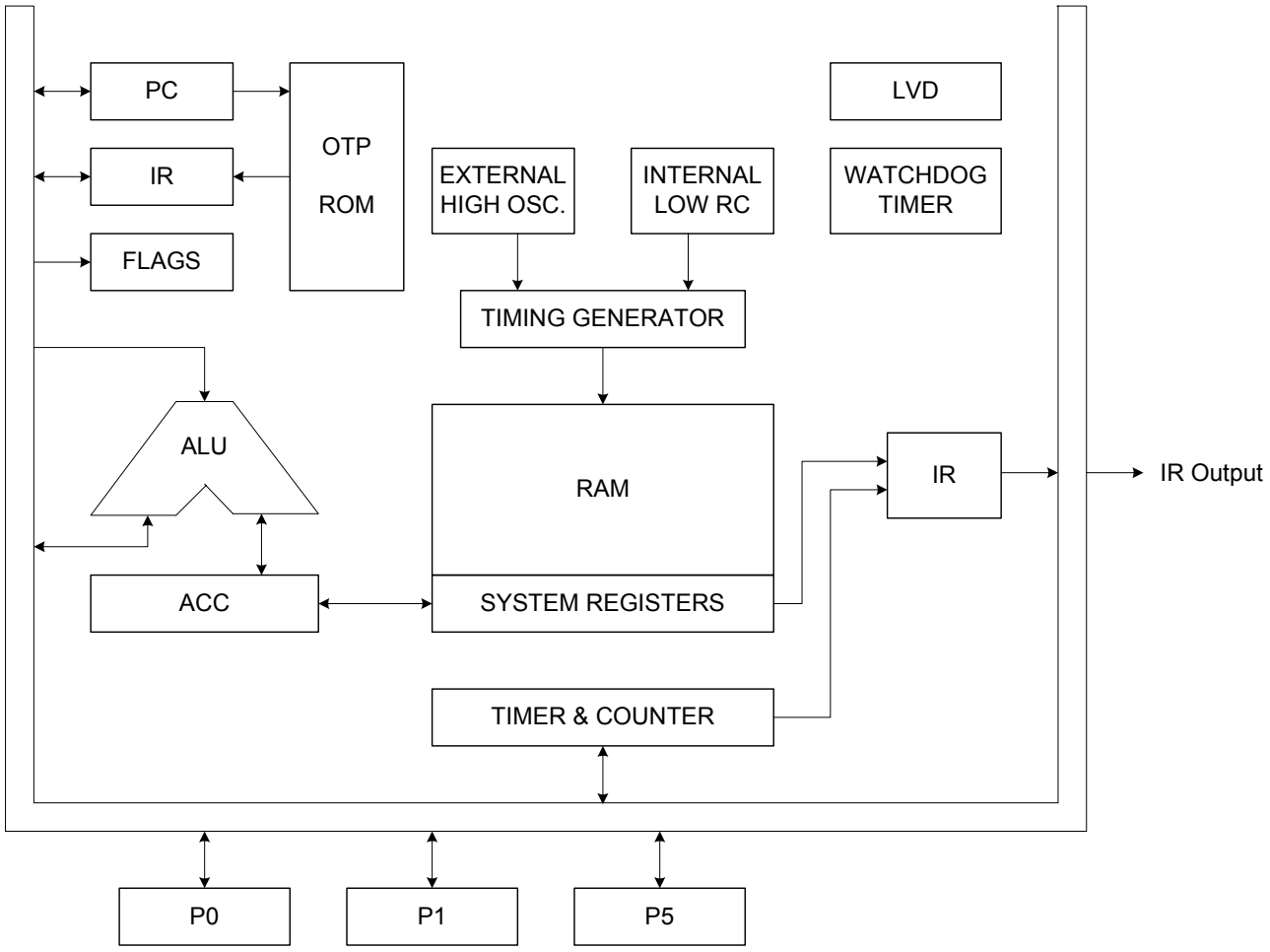
## 1.1 FEATURES

- **Memory configuration**
  OTP ROM size: 2K * 16 bits.
  RAM size: 48 * 8 bits.
- Four levels stack buffer
- **I/O pin configuration**
  Bi-directional: P1, P0.6, P0.7
  Input only: P0.0~P0.4
  IR output: IROUT
  Wakeup: P0, P1 level change trigger.
  Pull-up resisters: P0, P1
- **Powerful instructions**
  **Four clocks per instruction cycle (4T)**
  Most of instructions are one cycle only.
  All ROM area JMP instruction.
  All ROM area CALL address instruction.
  All ROM area lookup table function (MOVC)
- **Two 8-bit Timer/Counter**
  T0: Basic timer
  TC0: For IR output
- **One channel IR output.**
- **On chip watchdog timer and clock source is internal low clock RC type (16KHz @3V, 32KHz @5V).**
- **Single system clocks**
  External high clock: 4 MHz crystal/resonator
- **Operating modes**
  Normal mode: Both high and low clock active
  Sleep mode: Both high and low clock stop
- **Package (Chip form support)**
  PDIP 20 pins
  SOP 20 pins
  SSOP 20 pins

☞ **Features Selection Table**

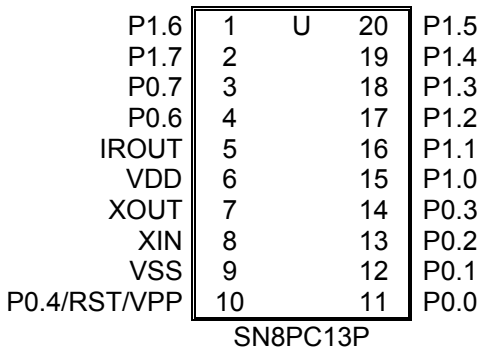
| CHIP | ROM | RAM | Stack | T0 | Osc. | I/O | IR Output | Wakeup Pin No. | Package |
|---|---|---|---|---|---|---|---|---|---|
| SN8PC01 | 0.5K*16 | 32 | 4 | - | 455KHz | 16 | Fix 38KHz | 9 | DIP20/SOP20 |
| SN8PC13 | 2K*16 | 48 | 4 | V | 4MHz | 16 | Duty, cycle programmable | 15 | DIP20/SOP20/SSOP20 |

## 1.2 SYSTEM BLOCK DIAGRAM




## 1.3 PIN ASSIGNMENT

**SN8PC13P (P-DIP 20 pins)**
**SN8PC13S (SOP 20 pins)**
**SN8PC13X (SSOP 20 pins)**



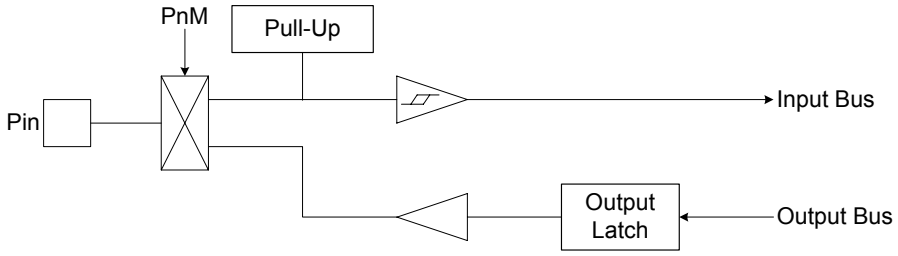

SN8PC13P
SN8PC13S
SN8PC13X

## 1.4 PIN DESCRIPTIONS

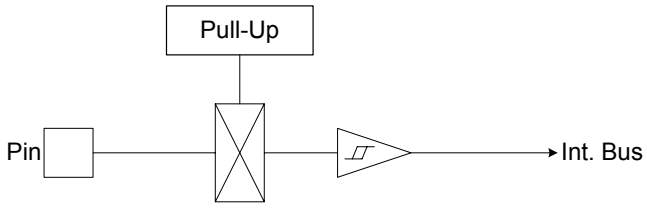
| PIN NAME | TYPE | DESCRIPTION |
|---|---|---|
| VDD, VSS | P | Power supply input pins for digital circuit. |
| XIN | I | XIN: Oscillator input pin. |
| XOUT | O | XOUT: Oscillator output pin. |
| P1.0 | I/O | P1.0: Port 1.0 bi-direction pin.<br>Schmitt trigger structure.<br>Built-in pull-up resistors as input mode.<br>Build-in wake-up function by edge trigger. |
| P1.1 | I/O | P1.1: Port 1.1 bi-direction pin.<br>Schmitt trigger structure.<br>Built-in pull-up resistors as input mode.<br>Build-in wake-up function by edge trigger. |
| P1 [7:2] | I/O | P1: Port 1 bi-direction pin.<br>Schmitt trigger structure.<br>Built-in pull-up resistors as input mode.<br>Build-in wake-up function by edge trigger. |
| P0 [3:0] | I | P0.0~P0.3: Port 0 input-only pin.<br>Schmitt trigger structure.<br>Built-in pull-up resistors.<br>Build-in wake-up function by edge trigger. |
| P0.4/RST/VPP | I, P | RST is system external reset input pin under Ext_RST mode.<br>Schmitt trigger structure, active “low”, normal stay to “high”.<br>P0.4 is input only pin without pull-up resistor under P0.4 mode.<br>Add the 100 ohm external resistor on P1.4,when it is set to be input pin.<br>Build-in wake-up function by edge trigger.<br>OTP 12.3V power input pin in programming mode. |
| P0 [7:6] | I/O | P0[7:6]: Port 0 bi-direction pin.<br>Schmitt trigger structure.<br>Built-in pull-up resistors as input mode.<br>Build-in wake-up function by edge trigger. |
| IROUT | O | IROUT: IR output pin. |

# 1.5 PIN CIRCUIT DIAGRAMS

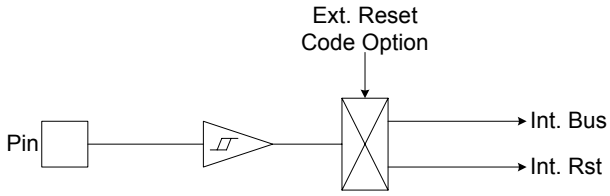
***Port 1, P0.6, P0.7 structure:***



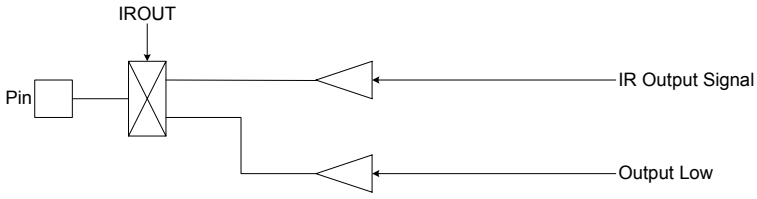

***Port 0.0~0.3 structure:***




***Port 0.4 structure:***




***IROUT structure:***

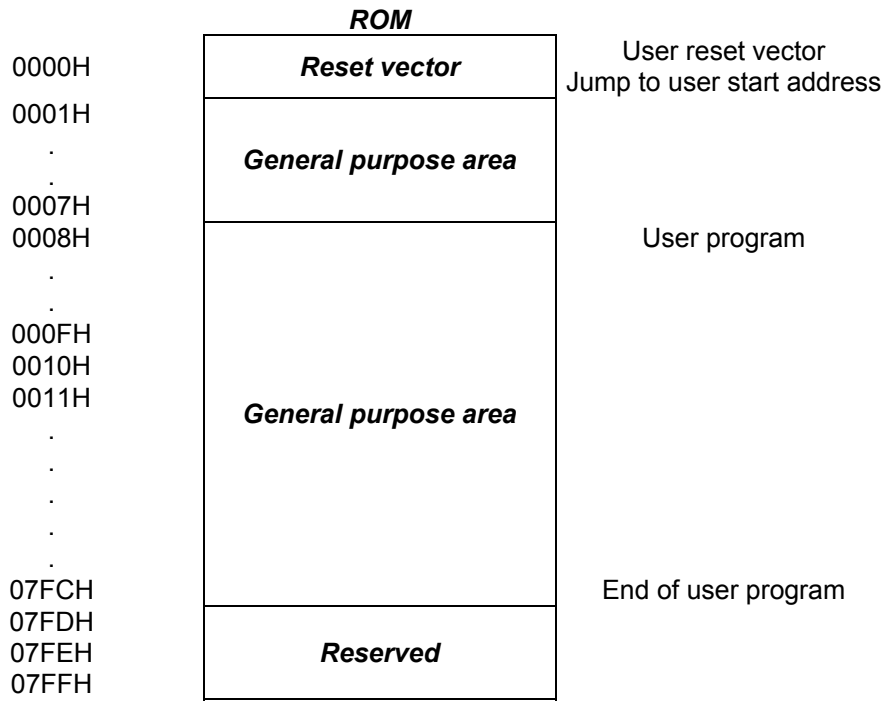


# 2 CENTRAL PROCESSOR UNIT (CPU)

## 2.1 MEMORY MAP

### 2.1.1 PROGRAM MEMORY (ROM)

☞ ***2K words ROM***



### 2.1.1.1 RESET VECTOR (0000H)

A one-word vector address area is used to execute system reset.

- ☞ **Power On Reset**
- ☞ **Watchdog Rese**
- ☞ **External Reset**

After power on reset, external reset or watchdog timer overflow reset, then the chip will restart the program from address 0000h and all system registers will be set as default values. The following example shows the way to define the reset vector in the program memory.

➢ **Example: Defining Reset Vector**

```
                ORG     0           ; 0000H
                JMP     START       ; Jump to user program address.
                …

                ORG     10H
START:                              ; 0010H, The head of user program.
                …                   ; User program
                …

                ENDP                ; End of program
```

### 2.1.1.2 CHECKSUM CALCULATION

The last ROM address are reserved area. User should avoid these addresses (last address) when calculate the Checksum value.

➢ **Example: The demo program shows how to calculated Checksum from 00H to the end of user's code.**

```
                MOV     A,#END_USER_CODE$L
                MOV     END_ADDR1, A            ; Save low end address to end_addr1
                MOV     A,#END_USER_CODE$M
                MOV     END_ADDR2, A            ; Save middle end address to end_addr2
                MOV     A, #0                   ; Set Y to 00H
                MOV     Y, A
                MOV     Z, A                    ; Set Z to 00H
@@:
                MOVC
                BCLR    FC                      ; Clear C flag
                ADC     DATA1, A                ; Add A to Data1
                MOV     A, R
                ADC     DATA2, A                ; Add R to Data2
                JMP     END_CHECK               ; Check if the YZ address =   the end of code
AAA:
                INCMS   Z                       ; Z=Z+1
                JMP     @B                      ; If Z != 00H calculate to next address
                JMP     Y_ADD_1                 ; If Z = 00H increase Y
END_CHECK:
                MOV     A, END_ADDR1
                CMPRS   A, Z                    ; Check if Z = low end address
                JMP     AAA                     ; If Not jump to checksum calculate
                MOV     A, END_ADDR2
                CMPRS   A, Y                    ; If Yes, check if Y = middle end address
                JMP     AAA                     ; If Not jump to checksum calculate
                JMP     CHECKSUM_END            ; If Yes checksum calculated is done.

Y_ADD_1:
                INCMS   Y                       ; Increase Y
                NOP
                JMP     @B                      ; Jump to checksum calculate
CHECKSUM_END:
                …
                …
END_USER_CODE:                                  ; Label of program end
```

## 2.1.2 CODE OPTION TABLE

| Code Option | Content | Function Description |
|---|---|---|
| Watch_Dog | Always_On | Watchdog timer is always on enable even in power down and green mode. |
| | Enable | Enable watchdog timer. Watchdog timer stops in power down mode and green mode. |
| | Disable | Disable Watchdog function. |
| Reset_Pin | Reset | Enable External reset pin. |
| | P04 | Enable P0.4 input only without pull-up resister. |
| Security | Enable | Enable ROM code Security function. |
| | Disable | Disable ROM code Security function. |

## 2.1.3 DATA MEMORY (RAM)

☞ ***48 X 8-bit RAM***

| | Address | RAM location | |
|---|---|---|---|
| **BANK 0** | 000h<br>“<br>“<br>“<br>“<br>“<br>02Fh | ***General purpose area*** | |
| | 080h<br>“<br>“<br>“<br>“<br>“ | ***System register*** | 80h~FFh of Bank 0 store system registers (128 bytes). |
| | 0FFh | ***End of bank 0 area*** | |

# 2.1.4 SYSTEM REGISTER

## 2.1.4.1 SYSTEM REGISTER TABLE

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8 | - | - | R | Z | Y | - | PFLAG | - | - | - | - | - | - | - | - | - |
| 9 | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| A | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - | - |
| B | - | - | - | - | - | - | - | - | P0M | - | - | - | - | - | - | - |
| C | - | P1M | - | - | - | - | - | - | INTRQ | - | OSCM | - | WDTR | TC0R | PCL | PCH |
| D | P0 | P1 | - | - | - | - | - | - | T0M | T0C | TC0M | TC0C | - | - | - | STKP |
| E | - | - | - | - | - | - | - | @YZ | TC0D | - | - | - | - | - | - | - |
| F | - | - | - | - | - | - | - | - | STK3L | STK3H | STK2L | STK2H | STK1L | STK1H | STK0L | STK0H |

## 2.1.4.2 SYSTEM REGISTER DESCRIPTION

PFLAG = ROM page and special flag register.
@YZ = RAM YZ indirect addressing index pointer.
PnM = Port n input/output mode register.
INTRQ = T0 overflow flag register.
OSCM = Oscillator mode register.
T0M = T0 mode register.
TC0M = TC0 mode register.
TC0R = TC0 auto-reload data buffer.
TC0D = IR output duty register.
STKP = Stack pointer buffer.

R = Working register and ROM look-up data buffer.
Y, Z = Working, @YZ and ROM addressing register.
Pn = Port n data buffer.
PnUR = Port n pull-up resister control register.
PCH, PCL = Program counter.
T0C = TC0 counting register.
TC0C = TC0 counting register.
WDTR = Watchdog timer clear register.
STK0~STK3 = Stack 0 ~ stack 3 buffer.

### 2.1.4.3 BIT DEFINITION of SYSTEM REGISTER

| Address | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 | R/W | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|
| 082H | RBIT7 | RBIT6 | RBIT5 | RBIT4 | RBIT3 | RBIT2 | RBIT1 | RBIT0 | R/W | R |
| 083H | ZBIT7 | ZBIT6 | ZBIT5 | ZBIT4 | ZBIT3 | ZBIT2 | ZBIT1 | ZBIT0 | R/W | Z |
| 084H | YBIT7 | YBIT6 | YBIT5 | YBIT4 | YBIT3 | YBIT2 | YBIT1 | YBIT0 | R/W | Y |
| 086H | - | - | - | - | - | C | | Z | R/W | PFLAG |
| 0B8H | P07M | P06M | - | - | - | - | - | - | R/W | P0M |
| 0C1H | P17M | P16M | P15M | P14M | P13M | P12M | P11M | P10M | R/W | P1M I/O direction |
| 0C8H | - | - | - | T0IRQ | - | - | - | - | R/W | INTRQ |
| 0CAH | 0 | 0 | 0 | CPUM1 | CPUM0 | 0 | 0 | 0 | R/W | OSCM |
| 0CCH | WDTR7 | WDTR6 | WDTR5 | WDTR4 | WDTR3 | WDTR2 | WDTR1 | WDTR0 | W | WDTR |
| 0CDH | TC0R7 | TC0R6 | TC0R5 | TC0R4 | TC0R3 | TC0R2 | TC0R1 | TC0R0 | W | TC0R |
| 0CEH | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 | R/W | PCL |
| 0CFH | - | - | - | - | - | PC10 | PC9 | PC8 | R/W | PCH |
| 0D0H | P07 | P06 | - | P04 | P03 | P02 | P01 | P00 | R/W | P0 data buffer |
| 0D1H | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 | R/W | P1 data buffer |
| 0D8H | T0ENB | T0rate2 | T0rate1 | T0rate0 | - | - | - | - | R/W | T0M |
| 0D9H | T0C7 | T0C6 | T0C5 | T0C4 | T0C3 | T0C2 | T0C1 | T0C0 | R/W | T0C |
| 0DAH | - | - | - | - | - | - | - | IR0OUT | R/W | TC0M |
| 0DBH | TC0C7 | TC0C6 | TC0C5 | TC0C4 | TC0C3 | TC0C2 | TC0C1 | TC0C0 | R/W | TC0C |
| 0DFH | - | - | - | - | - | - | STKPB1 | STKPB0 | R/W | STKP stack pointer |
| 0E7H | @YZ7 | @YZ6 | @YZ5 | @YZ4 | @YZ3 | @YZ2 | @YZ1 | @YZ0 | R/W | @YZ index pointer |
| 0E8H | TC0D7 | TC0D6 | TC0D5 | TC0D4 | TC0D3 | TC0D2 | TC0D1 | TC0D0 | W | TC0D |
| 0F8H | S3PC7 | S3PC6 | S3PC5 | S3PC4 | S3PC3 | S3PC2 | S3PC1 | S3PC0 | R/W | STK3L |
| 0F9H | - | - | - | - | - | S3PC10 | S3PC9 | S3PC8 | R/W | STK3H |
| 0FAH | S2PC7 | S2PC6 | S2PC5 | S2PC4 | S2PC3 | S2PC2 | S2PC1 | S2PC0 | R/W | STK2L |
| 0FBH | - | - | - | - | - | S2PC10 | S2PC9 | S2PC8 | R/W | STK2H |
| 0FCH | S1PC7 | S1PC6 | S1PC5 | S1PC4 | S1PC3 | S1PC2 | S1PC1 | S1PC0 | R/W | STK1L |
| 0FDH | - | - | - | - | - | S1PC10 | S1PC9 | S1PC8 | R/W | STK1H |
| 0FEH | S0PC7 | S0PC6 | S0PC5 | S0PC4 | S0PC3 | S0PC2 | S0PC1 | S0PC0 | R/W | STK0L |
| 0FFH | - | - | - | - | - | S0PC10 | S0PC9 | S0PC8 | R/W | STK0H |

✷ ***Note:***

1. ***To avoid system error, please be sure to put all the "0" and "1" as it indicates in the above table.***
2. ***All of register names had been declared in SN8ASM assembler.***
3. ***One-bit name had been declared in SN8ASM assembler with "F" prefix code.***
4. ***"bset", "bclr" instructions are only available to the "R/W" registers.***
5. ***For detail description, please refer to the "System Register Quick Reference Table".***

### 2.1.4.4 ACCUMULATOR

The ACC is an 8-bit data register responsible for transferring or manipulating data between ALU and data memory. If the result of operating is zero (Z) or there is carry (C) occurrence, then these flags will be set to PFLAG register.

- **Example: Read and write ACC value.**

```
; Read ACC data and store in BUF data memory
        MOV     BUF, A
; Write a immediate data into ACC
        MOV     A, #0FH
; Write ACC data from BUF data memory
        MOV     A, BUF
```

### 2.1.4.5 PROGRAM FLAG

The PFLAG register contains the arithmetic status of ALU operation. C, Z bits indicate the result status of ALU operation.

| 086H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **PFLAG** | - | - | - | - | - | C | - | Z |
| Read/Write | - | - | - | - | - | R/W | - | R/W |
| After reset | - | - | - | - | - | 0 | - | 0 |

Bit 2 **C:** Carry flag
1 = Addition with carry, subtraction without borrowing, rotation with shifting out logic “1”, comparison result ≥ 0.
0 = Addition without carry, subtraction with borrowing signal, rotation with shifting out logic “0”, comparison result < 0.

Bit 0 **Z:** Zero flag
1 = The result of an arithmetic/logic/branch operation is zero.
0 = The result of an arithmetic/logic/branch operation is not zero.

### 2.1.4.6 PROGRAM COUNTER

The program counter (PC) is a 11-bit binary counter separated into the high-byte 3 and the low-byte 8 bits. This counter is responsible for pointing a location in order to fetch an instruction for kernel circuit. Normally, the program counter is automatically incremented with each instruction during program execution.

Besides, it can be replaced with specific address by executing CALL or JMP instruction. When JMP or CALL instruction is executed, the destination address will be inserted to bit 0 ~ bit 10.

| | Bit 15 | Bit 14 | Bit 13 | Bit 12 | Bit 11 | Bit 10 | Bit 9 | Bit 8 | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **PC** | - | - | - | - | - | PC10 | PC9 | PC8 | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
| After reset | - | - | - | - | - | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | PCH | | | | | | | | PCL | | | | | | | |

☞ **ONE ADDRESS SKIPPING**

There are nine instructions (CMPRS, IINCMS, DECMS, BTS0, BTS1) with one address skipping function. If the result of these instructions is true, the PC will add 2 steps to skip next instruction.

***If the condition of bit test instruction is true, the PC will add 2 steps to skip next instruction.***

```
            BTS1    FC          ; To skip, if Carry_flag = 1
            JMP     C0STEP      ; Else jump to C0STEP.
            …
            …
C0STEP:     NOP

            MOV     A, BUF0     ; Move BUF0 value to ACC.
            BTS0    FZ          ; To skip, if Zero flag = 0.
            JMP     C1STEP      ; Else jump to C1STEP.
            …
            …
C1STEP:     NOP
```

***If the ACC is equal to the immediate data or memory, the PC will add 2 steps to skip next instruction.***

```
            CMPRS   A, #12H     ; To skip, if ACC = 12H.
            JMP     C0STEP      ; Else jump to C0STEP.
            …
            …
C0STEP:     NOP
```

***If the destination increased by 1, which results overflow of 0xFF to 0x00, the PC will add 2 steps to skip next instruction.***

**INCMS instruction:**

```
            INCMS     BUF0
            JMP       C0STEP      ; Jump to C0STEP if BUF0 is not zero.
            …
            …
C0STEP:     NOP
```

***If the destination decreased by 1, which results underflow of 0x00 to 0xFF, the PC will add 2 steps to skip next instruction.***

**DECMS instruction:**

```
            DECMS     BUF0
            JMP       C0STEP      ; Jump to C0STEP if BUF0 is not zero.
            …
            …
C0STEP:     NOP
```

☞ **MULTI-ADDRESS JUMPING**

Users can jump around the multi-address by either JMP instruction or ADD M, A instruction (M = PCL) to activate multi-address jumping function. Program Counter supports **"ADC M,A"** instructions for carry to PCH when PCL overflow automatically. For jump table or others applications, users can calculate PC value by the three instructions and don't care PCL overflow problem.

✸ ***Note: PCH only support PC up counting result and doesn't support PC down counting. When PCL is carry after PCL+ACC, PCH adds one automatically. If PCL borrow after PCL–ACC, PCH keeps value and not change.***

➢ **Example: If PC = 0323H (PCH = 03H, PCL = 23H)**

```
; PC = 0323H
            MOV       A, #28H
            MOV       PCL, A      ; Jump to address 0328H
            …

; PC = 0328H
            MOV       A, #00H
            MOV       PCL, A      ; Jump to address 0300H
            …
```

➢ **Example: If PC = 0323H (PCH = 03H, PCL = 23H)**

```
; PC = 0323H
            ADC       PCL, A      ; PCL = PCL + ACC, the PCH cannot be changed.
            JMP       A0POINT     ; If ACC = 0, jump to A0POINT
            JMP       A1POINT     ; ACC = 1, jump to A1POINT
            JMP       A2POINT     ; ACC = 2, jump to A2POINT
            JMP       A3POINT     ; ACC = 3, jump to A3POINT
            …
            …
```

### 2.1.4.7 Y, Z REGISTERS

The Y and Z registers are the 8-bit buffers. There are three major functions of these registers.

- can be used as general working registers
- can be used as RAM data pointers with @YZ register
- can be used as ROM data pointer with the MOVC instruction for look-up table

| 084H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Y** | YBIT7 | YBIT6 | YBIT5 | YBIT4 | YBIT3 | YBIT2 | YBIT1 | YBIT0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| After reset | - | - | - | - | - | - | - | - |

| 083H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **Z** | ZBIT7 | ZBIT6 | ZBIT5 | ZBIT4 | ZBIT3 | ZBIT2 | ZBIT1 | ZBIT0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| After reset | - | - | - | - | - | - | - | - |

➢ **Example: Uses Y, Z register as the data pointer to access data in the RAM address 025H of bank0.**

```
MOV     A, #00H         ; To set RAM bank 0 for Y register
MOV     Y, A
MOV     A, #25H         ; To set location 25H for Z register
MOV     Z, A
MOV     A, @YZ          ; To read a data into ACC
```

➢ **Example: Uses the Y, Z register as data pointer to clear the RAM data.**

```
        MOV     A, #0           ; Y = 0, bank 0
        MOV     Y, A
        MOV     A, #07FH        ; Z = 7FH, the last address of the data memory area
        MOV     Z, A

CLR_YZ_BUF:
        MOV     A, #0           ; Clear @YZ to be zero
        MOV     @YZ, A

        DECMS   Z               ; Z – 1, if Z= 0, finish the routine
        JMP     CLR_YZ_BUF      ; Not zero

        MOV     A, #0           ; Clear @YZ to be zero
        MOV     @YZ, A
END_CLR:                        ; End of clear general purpose data memory area of bank 0
        …
```

### 2.1.4.8 R REGISTERS

R register is an 8-bit buffer. There are two major functions of the register.

- Can be used as working register
- For store high-byte data of look-up table
  (MOVC instruction executed, the high-byte data of specified ROM address will be stored in R register and the low-byte data will be stored in ACC).

| 082H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **R** | RBIT7 | RBIT6 | RBIT5 | RBIT4 | RBIT3 | RBIT2 | RBIT1 | RBIT0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| After reset | - | - | - | - | - | - | - | - |

✷ ***Note: Please refer to the "LOOK-UP TABLE DESCRIPTION" about R register look-up table application.***

# 2.2 ADDRESSING MODE

## 2.2.1 IMMEDIATE ADDRESSING MODE

The immediate addressing mode uses an immediate data to set up the location in ACC or specific RAM.

- **Example: Move the immediate data 12H to ACC.**

```
MOV     A, #12H         ; To set an immediate data 12H into ACC.
```

## 2.2.2 DIRECTLY ADDRESSING MODE

The directly addressing mode moves the content of RAM location in or out of ACC.

- **Example: Move 0x12 RAM location data into ACC.**

```
MOV     A, 12H          ; To get a content of RAM location 0x12 of bank 0 and save in
                        ACC.
```

- **Example: Move ACC data into 0x12 RAM location.**

```
MOV     12H, A          ; To get a content of ACC and save in RAM location 12H of
                        bank 0.
```

## 2.2.3 INDIRECTLY ADDRESSING MODE

The indirectly addressing mode is to access the memory by the data pointer registers (Y/Z).

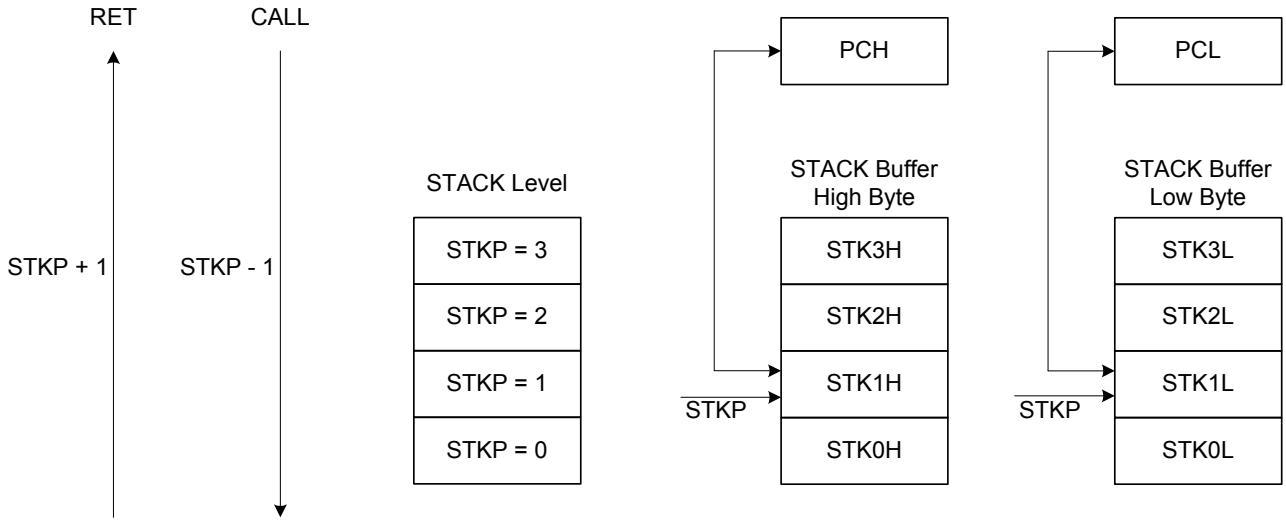
- **Example: Indirectly addressing mode with @YZ register.**

```
MOV     A, #0           ; To clear Y register to access RAM bank 0.
MOV     Y, A
MOV     A, #12H         ; To set an immediate data 12H into Z register.
MOV     Z, A
MOV     A, @YZ          ; Use data pointer @YZ reads a data from RAM location
                        ; 012H into ACC.
```

# 2.3 STACK OPERATION

## 2.3.1 OVERVIEW

The stack buffer has 4-level. These buffers are designed to push and pop up program counter's (PC) data when "CALL" instruction are executed. The STKP register is a pointer designed to point active level in order to push or pop up data from stack buffer. The STKnH and STKnL are the stack buffers to store program counter (PC) data.



## 2.3.2 STACK REGISTERS

The stack pointer (STKP) is a 2-bit register to store the address used to access the stack buffer, 11-bit data memory (STKnH and STKnL) set aside for temporary storage of stack addresses.

The two stack operations are writing to the top of the stack (push) and reading from the top of stack (pop). Push operation decrements the STKP and the pop operation increments each time. That makes the STKP always point to the top address of stack buffer and write the last program counter value (PC) into the stack buffer.

The program counter (PC) value is stored in the stack buffer before a CALL instruction executed. Stack operation is a LIFO type (Last in and first out). The stack pointer (STKP) and stack buffer (STKnH and STKnL) are located in the system register area bank 0.

| 0DFH | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **STKP** | - | - | - | - | - | - | STKPB1 | STKPB0 |
| Read/Write | - | - | - | - | - | - | R/W | R/W |
| After reset | - | - | - | - | - | - | 1 | 1 |

Bit[2:0] **STKPBn:** Stack pointer (n = 0 ~ 1)

- **Example: Stack pointer (STKP) reset, we strongly recommended to clear the stack pointers in the beginning of the program.**

```
MOV     A, #00000011B
MOV     STKP, A
```

| 0F0H~0F8H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **STKnH** | - | - | - | - | - | SnPC10 | SnPC9 | SnPC8 |
| Read/Write | - | - | - | - | - | R/W | R/W | R/W |
| After reset | - | - | - | - | - | 0 | 0 | 0 |

| 0F0H~0F8H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **STKnL** | SnPC7 | SnPC6 | SnPC5 | SnPC4 | SnPC3 | SnPC2 | SnPC1 | SnPC0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

***STKn = STKnH , STKnL (n = 3 ~ 0)***

## 2.3.3 STACK OPERATION EXAMPLE

The two kinds of Stack-Save operations refer to the stack pointer (STKP) and write the content of program counter (PC) to the stack buffer are CALL instruction. Under each condition, the STKP decreases and points to the next available stack location. The stack buffer stores the program counter about the op-code address. The Stack-Save operation is as the following table.

| *Stack Level* | *STKP Register* | | *Stack Buffer* | | *Description* |
|---|---|---|---|---|---|
| | **STKPB1** | **STKPB0** | **High Byte** | **Low Byte** | |
| **0** | 1 | 1 | Free | Free | - |
| **1** | 1 | 0 | STK0H | STK0L | - |
| **2** | 0 | 1 | STK1H | STK1L | - |
| **3** | 0 | 0 | STK2H | STK2L | - |
| **4** | 1 | 1 | STK3H | STK3L | - |
| **> 4** | 1 | 0 | - | - | **Stack Over, error** |

There are Stack-Restore operations correspond to each push operation to restore the program counter (PC). The RET instruction is for CALL instruction. When a pop operation occurs, the STKP is incremented and points to the next free stack location. The stack buffer restores the last program counter (PC) to the program counter registers. The Stack-Restore operation is as the following table.

| *Stack Level* | *STKP Register* | | *Stack Buffer* | | *Description* |
|---|---|---|---|---|---|
| | **STKPB1** | **STKPB0** | **High Byte** | **Low Byte** | |
| **4** | 1 | 1 | STK3H | STK3L | - |
| **3** | 0 | 0 | STK2H | STK2L | - |
| **2** | 0 | 1 | STK1H | STK1L | - |
| **1** | 1 | 0 | STK0H | STK0L | - |
| **0** | 1 | 1 | Free | Free | - |

# 3 RESET

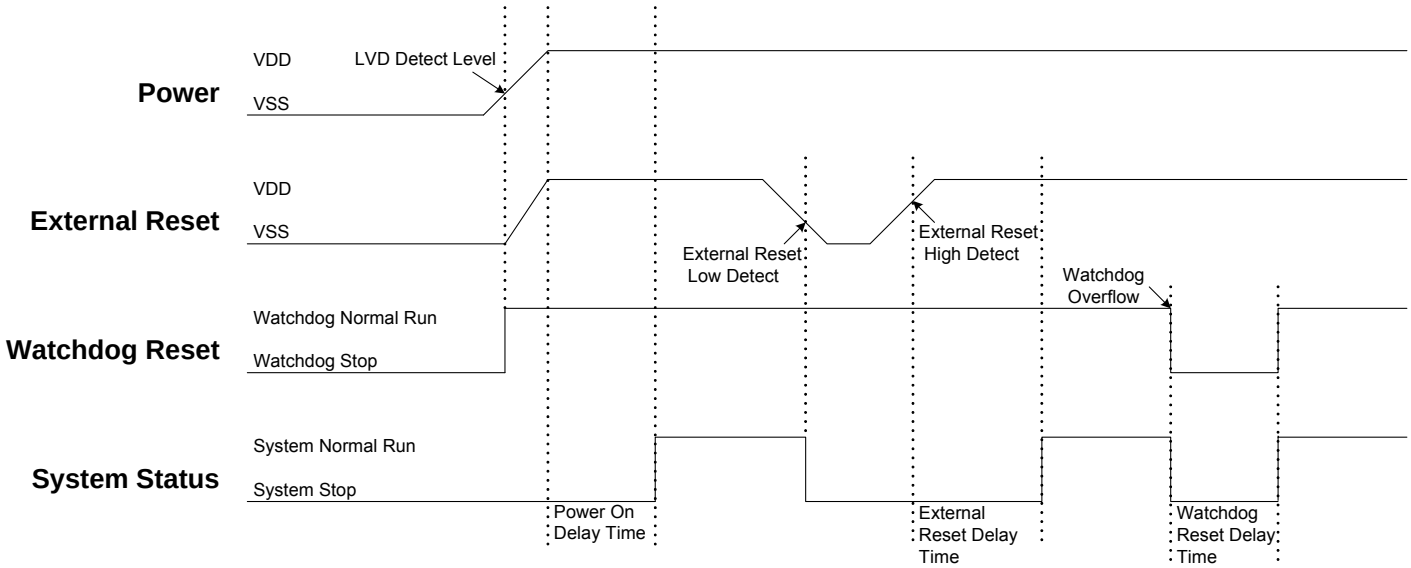
## 3.1 OVERVIEW

The system would be reset in three conditions as following.

- Power on reset
- Watchdog reset
- Brown out reset
- External reset (only supports external reset pin enable situation)

When any reset condition occurs, all system registers keep initial status, program stops and program counter is cleared. After reset status released, the system boots up and program starts to execute from ORG 0.

Finishing any reset sequence needs some time. The system provides complete procedures to make the power on reset successful. For different oscillator types, the reset time is different. That causes the VDD rise rate and start-up time of different oscillator is not fixed. RC type oscillator's start-up time is very short, but the crystal type is longer. Under client terminal application, users have to take care the power on reset time for the master terminal requirement. The reset timing diagram is as following.



## 3.2 POWER ON RESET

The power on reset depend no LVD operation for most power-up situations. The power supplying to system is a rising curve and needs some time to achieve the normal voltage. Power on reset sequence is as following.

- **Power-up:** System detects the power voltage up and waits for power stable.
- **External reset (only external reset pin enable):** System checks external reset pin status. If external reset pin is not high level, the system keeps reset status and waits external reset pin released.
- **System initialization:** All system registers is set as initial conditions and system is ready.
- **Oscillator warm up:** Oscillator operation is successfully and supply to system clock.
- **Program executing:** Power on sequence is finished and program executes from ORG 0.

## 3.3 WATCHDOG RESET

Watchdog reset is a system protection. In normal condition, system works well and clears watchdog timer by program. Under error condition, system is in unknown situation and watchdog can't be clear by program before watchdog timer overflow. Watchdog timer overflow occurs and the system is reset. After watchdog reset, the system restarts and returns normal mode. Watchdog reset sequence is as following.

- **Watchdog timer status:** System checks watchdog timer overflow status. If watchdog timer overflow occurs, the system is reset.
- **System initialization:** All system registers is set as initial conditions and system is ready.
- **Oscillator warm up:** Oscillator operation is successfully and supply to system clock.
- **Program executing:** Power on sequence is finished and program executes from ORG 0.

Watchdog timer application note is as following.

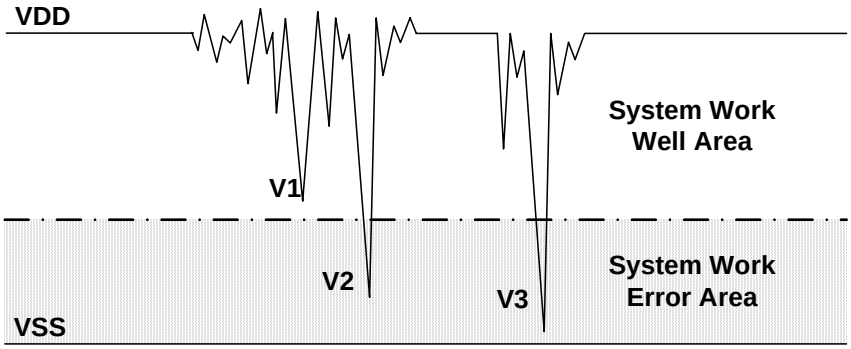
- Before clearing watchdog timer, check I/O status and check RAM contents can improve system error.
- Don't clear watchdog timer in interrupt vector and interrupt service routine. That can improve main routine fail.
- Clearing watchdog timer program is only at one part of the program. This way is the best structure to enhance the watchdog timer function.

✷ ***Note: Please refer to the "WATCHDOG TIMER" about watchdog timer detail information.***

# 3.4 BROWN OUT RESET

## 3.4.1 BROWN OUT DESCRIPTION

The brown out reset is a power dropping condition. The power drops from normal voltage to low voltage by external factors (e.g. EFT interference or external loading changed). The brown out reset would make the system not work well or executing program error.




**Brown Out Reset Diagram**

The power dropping might through the voltage range that's the system dead-band. The dead-band means the power range can't offer the system minimum operation power requirement. The above diagram is a typical brown out reset diagram. There is a serious noise under the VDD, and VDD voltage drops very deep. There is a dotted line to separate the system working area. The above area is the system work well area. The below area is the system work error area called dead-band. V1 doesn't touch the below area and not effect the system operation. But the V2 and V3 is under the below area and may induce the system error occurrence. Let system under dead-band includes some conditions.

**DC application:**

The power source of DC application is usually using battery. When low battery condition and MCU drive any loading, the power drops and keeps in dead-band. Under the situation, the power won't drop deeper and not touch the system reset voltage. That makes the system under dead-band.

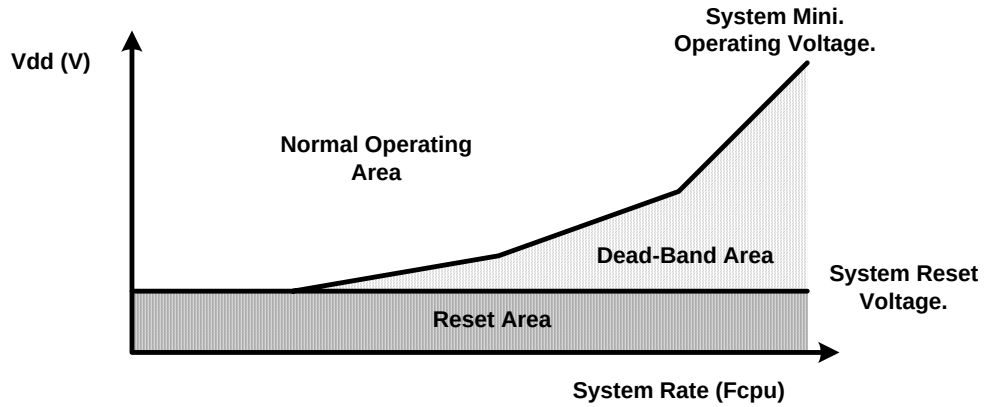
**AC application:**

In AC power application, the DC power is regulated from AC power source. This kind of power usually couples with AC noise that makes the DC power dirty. Or the external loading is very heavy, e.g. driving motor. The loading operating induces noise and overlaps with the DC power. VDD drops by the noise, and the system works under unstable power situation.

The power on duration and power down duration are longer in AC application. The system power on sequence protects the power on successful, but the power down situation is like DC low battery condition. When turn off the AC power, the VDD drops slowly and through the dead-band for a while.

## 3.4.2 THE SYSTEM OPERATING VOLTAGE DECSRIPTION

To improve the brown out reset needs to know the system minimum operating voltage which is depend on the system executing rate and power level. Different system executing rates have different system minimum operating voltage. The electrical characteristic section shows the system voltage to executing rate relationship.




Normally the system operation voltage area is higher than the system reset voltage to VDD, and the reset voltage is decided by LVD detect level. The system minimum operating voltage rises when the system executing rate upper even higher than system reset voltage. The dead-band definition is the system minimum operating voltage above the system reset voltage.

## 3.4.3 BROWN OUT RESET IMPROVEMENT

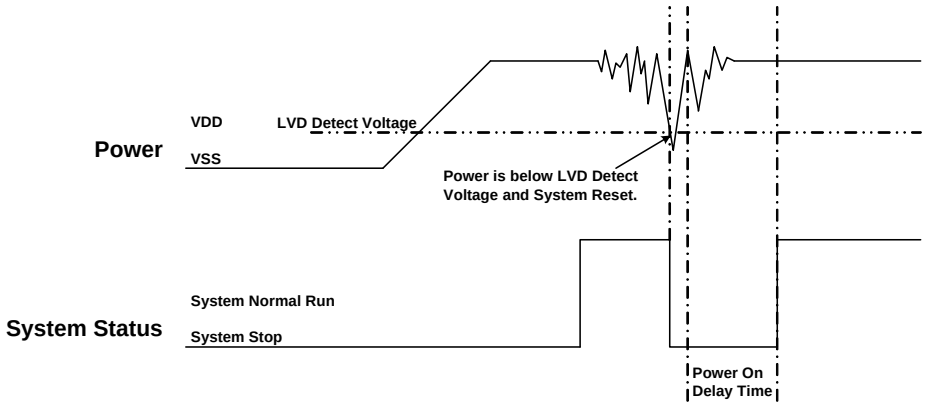
**How to improve the brown reset condition?** There are some methods to improve brown out reset as following.

- **LVD reset**
- **Watchdog reset**
- **Reduce the system executing rate**
- **External reset circuit. (Zener diode reset circuit, Voltage bias reset circuit, External reset IC)**

✷ ***Note:***

1. ***The " Zener diode reset circuit", "Voltage bias reset circuit" and "External reset IC" can completely improve the brown out reset, DC low battery and AC slow power down conditions.***
2. ***For AC power application and enhance EFT performance, the system clock is 4MHz/4 (1 mips) and use external reset (" Zener diode reset circuit", "Voltage bias reset circuit", "External reset IC"). The structure can improve noise effective and get good EFT characteristic.***

**LVD reset:**




The LVD (low voltage detector) is built-in Sonix 8-bit MCU to be brown out reset protection. When the VDD drops and is below LVD detect voltage, the LVD would be triggered, and the system is reset. The LVD detect level is different by each MCU. The LVD voltage level is a point of voltage and not easy to cover all dead-band range. Using LVD to improve brown out reset is depend on application requirement and environment. If the power variation is very deep, violent and trigger the LVD, the LVD can be the protection. If the power variation can touch the LVD detect level and make system work error, the LVD can't be the protection and need to other reset methods. More detail LVD information is in the electrical characteristic section.

**Watchdog reset:**
The watchdog timer is a protection to make sure the system executes well. Normally the watchdog timer would be clear at one point of program. Don't clear the watchdog timer in several addresses. The system executes normally and the watchdog won't reset system. When the system is under dead-band and the execution error, the watchdog timer can't be clear by program. The watchdog is continuously counting until overflow occurrence. The overflow signal of watchdog timer triggers the system to reset, and the system return to normal mode after reset sequence. This method also can improve brown out reset condition and make sure the system to return normal mode.
If the system reset by watchdog and the power is still in dead-band, the system reset sequence won't be successful and the system stays in reset status until the power return to normal range.

**Reduce the system executing rate:**
If the system rate is fast and the dead-band exists, to reduce the system executing rate can improve the dead-band. The lower system rate is with lower minimum operating voltage. Select the power voltage that's no dead-band issue and find out the mapping system rate. Adjust the system rate to the value and the system exits the dead-band issue. This way needs to modify whole program timing to fit the application requirement.

**External reset circuit:**
The external reset methods also can improve brown out reset and is the complete solution. There are three external reset circuits to improve brown out reset including "Zener diode reset circuit", "Voltage bias reset circuit" and "External reset IC". These three reset structures use external reset signal and control to make sure the MCU be reset under power dropping and under dead-band. The external reset information is described in the next section.

# 3.5 EXTERNAL RESET

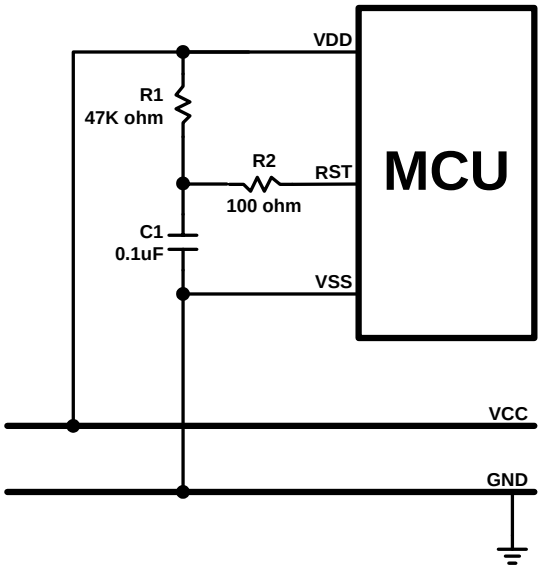
External reset function is controlled by “Reset_Pin” code option. Set the code option as “Reset” option to enable external reset function. External reset pin is Schmitt Trigger structure and low level active. The system is running when reset pin is high level voltage input. The reset pin receives the low voltage and the system is reset. The external reset operation actives in power on and normal running mode. During system power-up, the external reset pin must be high level input, or the system keeps in reset status. External reset sequence is as following.

- **External reset (only external reset pin enable):** System checks external reset pin status. If external reset pin is not high level, the system keeps reset status and waits external reset pin released.
- **System initialization:** All system registers is set as initial conditions and system is ready.
- **Oscillator warm up:** Oscillator operation is successfully and supply to system clock.
- **Program executing:** Power on sequence is finished and program executes from ORG 0.

The external reset can reset the system during power on duration, and good external reset circuit can protect the system to avoid working at unusual power condition, e.g. brown out reset in AC power application…

# 3.6 EXTERNAL RESET CIRCUIT

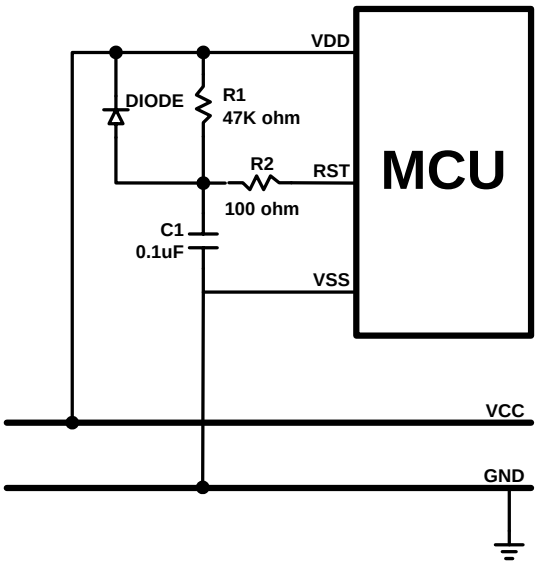
## 3.6.1 Simply RC Reset Circuit




This is the basic reset circuit, and only includes R1 and C1. The RC circuit operation makes a slow rising signal into reset pin as power up. The reset signal is slower than VDD power up timing, and system occurs a power on signal from the timing difference.

✸ ***Note: The reset circuit is no any protection against unusual power or brown out reset.***

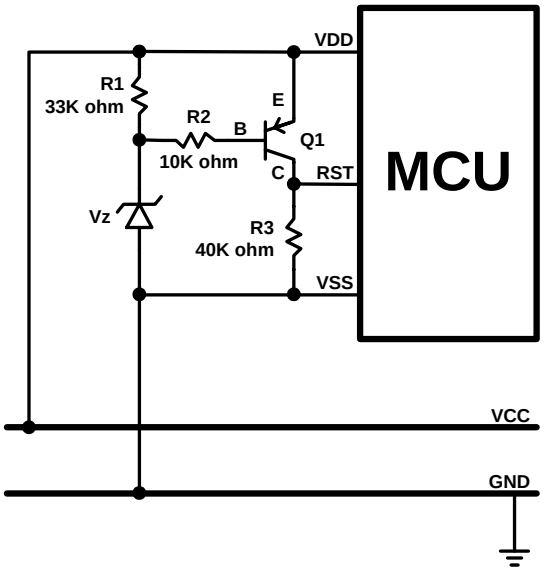
## 3.6.2 Diode & RC Reset Circuit




This is the better reset circuit. The R1 and C1 circuit operation is like the simply reset circuit to make a power on signal. The reset circuit has a simply protection against unusual power. The diode offers a power positive path to conduct higher power to VDD. It is can make reset pin voltage level to synchronize with VDD voltage. The structure can improve slight brown out reset condition.

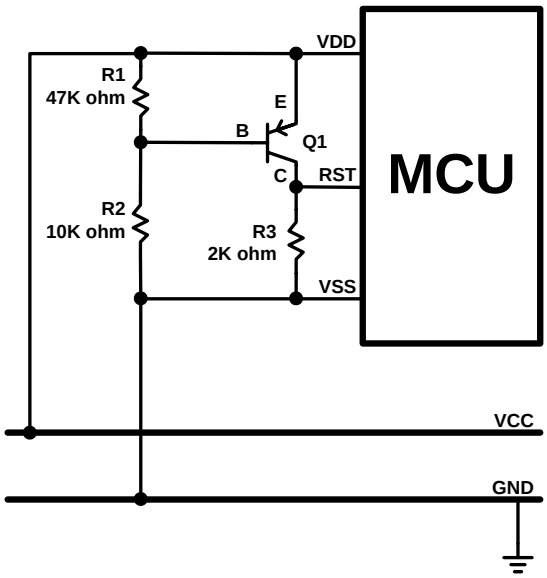
✸ ***Note: The R2 100 ohm resistor of "Simply reset circuit" and "Diode & RC reset circuit" is necessary to limit any current flowing into reset pin from external capacitor C in the event of reset pin breakdown due to Electrostatic Discharge (ESD) or Electrical Over-stress (EOS).***

## 3.6.3 Zener Diode Reset Circuit




The zener diode reset circuit is a simple low voltage detector and can **improve brown out reset condition completely**. Use zener voltage to be the active level. When VDD voltage level is above "Vz + 0.7V", the C terminal of the PNP transistor outputs high voltage and MCU operates normally. When VDD is below "Vz + 0.7V", the C terminal of the PNP transistor outputs low voltage and MCU is in reset mode. Decide the reset detect voltage by zener specification. Select the right zener voltage to conform the application.

## 3.6.4 Voltage Bias Reset Circuit

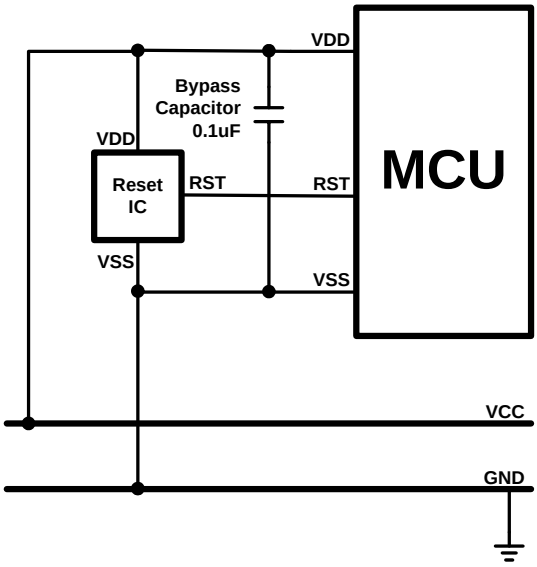



The voltage bias reset circuit is a low cost voltage detector and can **improve brown out reset condition completely**. The operating voltage is not accurate as zener diode reset circuit. Use R1, R2 bias voltage to be the active level. When VDD voltage level is above or equal to "0.7V x (R1 + R2) / R1", the C terminal of the PNP transistor outputs high voltage and MCU operates normally. When VDD is below "0.7V x (R1 + R2) / R1", the C terminal of the PNP transistor outputs low voltage and MCU is in reset mode.

Decide the reset detect voltage by R1, R2 resistances. Select the right R1, R2 value to conform the application. In the circuit diagram condition, the MCU's reset pin level varies with VDD voltage variation, and the differential voltage is 0.7V. If the VDD drops and the voltage lower than reset pin detect level, the system would be reset. If want to make the reset active earlier, set the R2 > R1 and the cap between VDD and C terminal voltage is larger than 0.7V. The external reset circuit is with a stable current through R1 and R2. For power consumption issue application, e.g. DC power system, the current must be considered to whole system power consumption.

✸ ***Note: Under unstable power condition as brown out reset, "Zener diode rest circuit" and "Voltage bias reset circuit" can protects system no any error occurrence as power dropping. When power drops below the reset detect voltage, the system reset would be triggered, and then system executes reset sequence. That makes sure the system work well under unstable power situation.***

## 3.6.5 External Reset IC




The external reset circuit also use external reset IC to enhance MCU reset performance. This is a high cost and good effect solution. By different application and system requirement to select suitable reset IC. The reset circuit can improve all power variation.

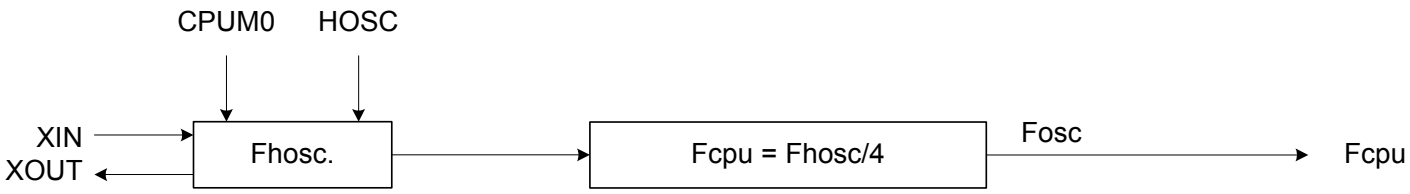
# 4 SYSTEM CLOCK

## 4.1 OVERVIEW

The micro-controller is a single clock system. The high-speed clock is generated from the external oscillator circuit. The high-speed clock can be system clock (Fosc). The system clock is divided by 4 to be the instruction cycle (Fcpu).

☞ **Normal Mode (High Clock): Fcpu = Fhosc/4.**

## 4.2 CLOCK BLOCK DIAGRAM




- HOSC: High_Clk code option.
- Fhosc: External high-speed clock.
- Fosc: System clock source.
- Fcpu: Instruction cycle.

## 4.3 OSCM REGISTER

The OSCM register is an oscillator control register. It controls oscillator status, system mode.

| 0CAH | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **OSCM** | 0 | 0 | 0 | 0 | CPUM0 | 0 | 0 | 0 |
| Read/Write | - | - | - | - | R/W | - | - | - |
| After reset | - | - | - | - | 0 | - | - | - |

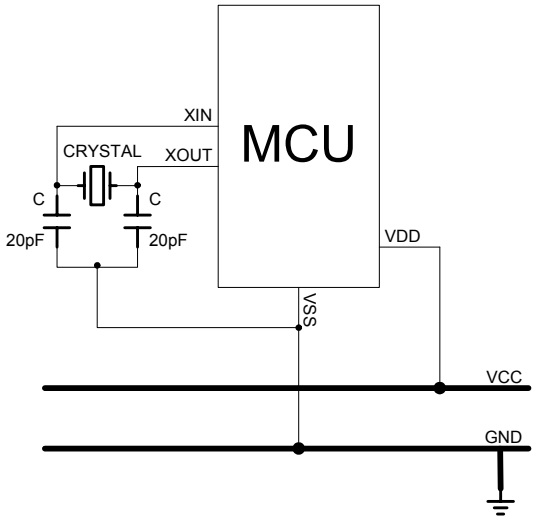
Bit 3 **CPUM0:** Power down mode (sleep mode) control bit.
0 = normal.
1 = power down mode (sleep mode).

➢ **Example: When entering the power down mode (sleep mode), high-speed oscillator will be stopped.**

```
BSET    FCPUM0      ; To stop external high-speed oscillator called power down
                    ; mode (sleep mode).
```

## 4.4 SYSTEM HIGH CLOCK

Crystal/Ceramic devices are driven by XIN, XOUT pins. Only support 4MHz oscillator frequency.



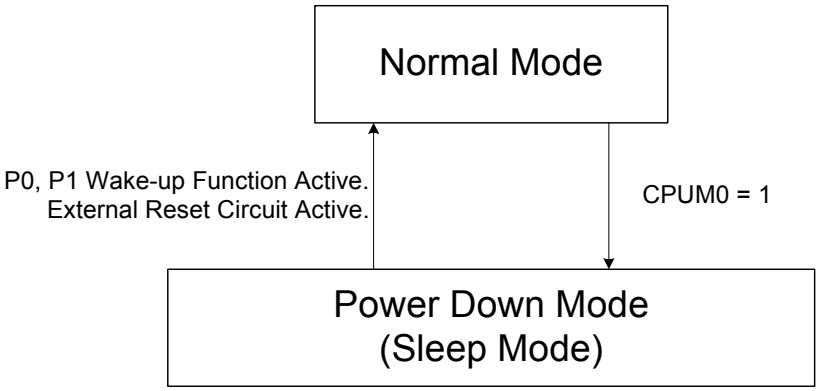

✸ ***Note: Connect the Crystal/Ceramic and C as near as possible to the XIN/XOUT/VSS pins of micro-controller.***

# 5 SYSTEM OPERATION MODE

## 5.1 OVERVIEW

The chip is featured with low power consumption by switching around two different modes as following.

- Normal mode (High-speed mode)
- Power-down mode (Sleep mode)




**System Mode Switching Diagram**

***Operating mode description***

| MODE | NORMAL | POWER DOWN (SLEEP) | REMARK |
|---|---|---|---|
| EHOSC | Running | Stop | |
| CPU instruction | Executing | Stop | |
| T0 timer | *Active | Inactive | * Active if T0ENB=1 |
| Watchdog timer | By Watch_Dog Code option | By Watch_Dog Code option | Refer to code option description |
| Wakeup source | - | P0, P1, Reset | |

**EHOSC**: External high clock

## 5.2 SYSTEM MODE SWITCHING

➢ **Example: Switch normal/slow mode to power down (sleep) mode.**

```
BSET        FCPUM0          ; Set CPUM0 = 1.
```

✷ ***Note: During the sleep, only the wakeup pin and reset can wakeup the system back to the normal mode.***

# 5.3 WAKEUP

## 5.3.1 OVERVIEW

Under power down mode (sleep mode), program doesn't execute. The wakeup trigger can wake the system up to normal mode. The wakeup trigger sources are external trigger (P0, P1 level change).

- Power down mode is waked up to normal mode. The wakeup trigger is only external trigger (P0, P1 level change)

## 5.3.2 WAKEUP TIME

When the system is in power down mode (sleep mode), the high clock oscillator stops. When waked up from power down mode, MCU waits for 4096 external high-speed oscillator clocks as the wakeup time to stable the oscillator circuit. After the wakeup time, the system goes into the normal mode.

The value of the wakeup time is as the following.

***The Wakeup time = 1/Fosc * 4096 (sec) + high clock start-up time***

✸ ***Note: The high clock start-up time is depended on the VDD and oscillator type of high clock.***

➢ **Example: In power down mode (sleep mode), the system is waked up. After the wakeup time, the system goes into normal mode. The wakeup time is as the following.**

***The wakeup time = 1/Fosc * 4096 = 1.024 ms (Fosc = 4MHz)***

***The total wakeup time = 1.024ms + oscillator start-up time***

# 6 I/O PORT

There are 16 I/O pins and three ports in the MCU. All pins include pull-up registers and with power down mode wake-up function in input mode. The pull-up resistor and wake-up function are fixed.

- P0.0~P0.3 are input only pin.
- P0.4 is input only pin and shared with external reset pin controlled by code option.
- P0.6, P0.7 are bit-direction I/O controlled by P0M register.
- P1 is bit-direction I/O port controlled by P0M register.
- IROUT is IR signal output pin as IROUT=1. When IR output disable and the system is in power down mode, IROUT outputs low status. (More detail information in "IR OUTPUT" section.)

## 6.1 I/O PORT MODE

The port direction is programmed by PnM register. All I/O ports can select input or output direction.

| 0B8H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **P0M** | P07M | P06M | - | - | - | - | - | - |
| Read/Write | R/W | R/W | - | - | - | - | - | - |
| After reset | 0 | 0 | - | - | - | - | - | - |

| 0C1H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **P1M** | P17M | P16M | P15M | P14M | P13M | P12M | P12M | P10M |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Bit[7:0] **PnM[7:0]:** Pn mode control bits. (n = 0~1).
0 = Pn is input mode.
1 = Pn is output mode.

✸ ***Note: Users can program them by bit control instructions (BSET, BCLR).***

➢ **Example: I/O mode selecting**

```
MOV     A, #0H          ; Set all ports to be input mode.
MOV     P0M, A
MOV     P1M, A

MOV     A, #0FFH        ; Set all ports to be output mode.
MOV     P0M, A
MOV     P1M, A

BCLR    P1M.2           ; Set P1.2 to be input mode.

BSET    P1M.2           ; Set P1.2 to be output mode.
```

## 6.2 I/O PORT DATA REGISTER

| 0D0H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **P0** | P07 | P06 | - | P04 | P03 | P02 | P01 | P00 |
| Read/Write | R/W | R/W | - | R | R | R | R | R |
| After reset | 0 | 0 | - | 0 | 0 | 0 | 0 | 0 |

| 0D1H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **P1** | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
| Read/Write | R | R/W | R | R/W | R/W | R/W | R/W | R/W |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

✸ ***Note: The P04 keeps "1" when external reset enable by code option.***
✸ ***Note: When set P1.4 to input mode, please add the series external 100 ohm on it.***

➢ **Example: Read data from input port.**

```
MOV     A, P0       ; Read data from Port 0
MOV     A, P1       ; Read data from Port 1
```

➢ **Example: Write data to output port.**

```
MOV     A, #0FFH    ; Write data FFH to all Port.
MOV     P0, A
MOV     P1, A
```

➢ **Example: Write one bit data to output port.**

```
BSET    P1.3        ; Set P1.3 and P5.5 to be "1".

BCLR    P1.3        ; Set P1.3 and P5.5 to be "0".
```

# 7 TIMERS

## 7.1 WATCHDOG TIMER

The watchdog timer (WDT) is a binary up counter designed for monitoring program execution. If the program goes into the unknown status by noise interference, WDT overflow signal raises and resets MCU. Watchdog clock controlled by code option and the clock source is internal low-speed oscillator (16KHz @3V, 32KHz @5V).

***Watchdog overflow time = 8192 / Internal Low-Speed oscillator (sec).***

| *VDD* | *Internal Low RC Freq.* | *Watchdog Overflow Time* |
|---|---|---|
| 3V | 16KHz | 512ms |
| 5V | 32KHz | 256ms |

✷ ***Note: If watchdog is "Always_On" mode, it keeps running event under power down mode or green mode.***

Watchdog clear is controlled by WDTR register. Moving **0x5A** data into WDTR is to reset watchdog timer.

| 0CCH | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **WDTR** | WDTR7 | WDTR6 | WDTR5 | WDTR4 | WDTR3 | WDTR2 | WDTR1 | WDTR0 |
| Read/Write | W | W | W | W | W | W | W | W |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

➢ **Example: An operation of watchdog timer is as following. To clear the watchdog timer counter in the top of the main routine of the program.**

```
Main:
        MOV     A, #5AH         ; Clear the watchdog timer.
        MOV     WDTR, A
        …
        …
        CALL    SUB1
        CALL    SUB2
        …
        …
        JMP     MAIN
```

➢ **Example: Clear watchdog timer by @RST_WDT macro.**

```
Main:
        @RST_WDT                ; Clear the watchdog timer.
        …
        …
        CALL    SUB1
        CALL    SUB2
        …
        …
        JMP     MAIN
```

Watchdog timer application note is as following.

- Before clearing watchdog timer, check I/O status and check RAM contents can improve system error.
- Clearing watchdog timer program is only at one part of the program. This way is the best structure to enhance the watchdog timer function.

➢ **Example: An operation of watchdog timer is as following. To clear the watchdog timer counter in the top of the main routine of the program.**

```
Main:
                …                               ; Check I/O.
                …                               ; Check RAM
Err:            JMP $                           ; I/O or RAM error. Program jump here and don't
                                                ; clear watchdog. Wait watchdog timer overflow to reset IC.

Correct:                                        ; I/O and RAM are correct. Clear watchdog timer and
                                                ; execute program.
                @RST_WDT                        ; Only one clearing watchdog timer of whole program.
                …
                CALL            SUB1
                CALL            SUB2
                …
                …
                …
                JMP             MAIN
```

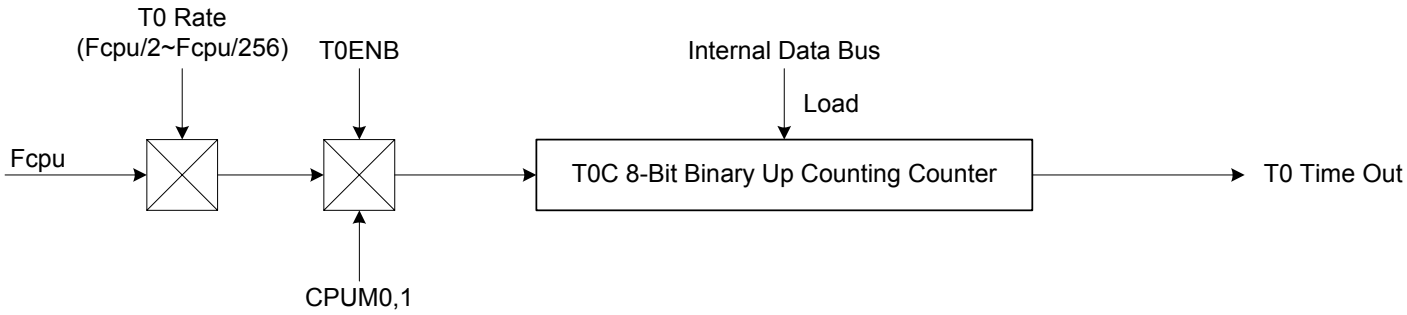
# 7.2 TIMER 0 (T0)

## 7.2.1 OVERVIEW

The T0 is an 8-bit binary up timer without interrupt function. If T0 timer occurs an overflow (from FFH to 00H), it will continue counting and issue a time-out signal to trigger T0 interrupt request flag and set T0IRQ (T0IRQ=1). Tracking T0IRQ status is to get right timer period by polling program.

The main purposes of the T0 timer is as following.

☞ **8-bit programmable up counting timer:** Generates timer overflow request at specific time intervals based on the selected clock frequency.




## 7.2.2 T0M MODE REGISTER

| 0D8H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **T0M** | T0ENB | T0rate2 | T0rate1 | T0rate0 | - | - | - | - |
| Read/Write | R/W | R/W | R/W | R/W | - | - | - | - |
| After reset | 0 | 0 | 0 | 0 | - | - | - | - |

Bit [6:4] **T0RATE[2:0]:** T0 internal clock select bits.
000 = fcpu/256.
001 = fcpu/128.
…
110 = fcpu/4.
111 = fcpu/2.

Bit 7 **T0ENB:** T0 counter control bit.
0 = Disable T0 timer.
1 = Enable T0 timer.

## 7.2.3 T0C COUNTING REGISTER

T0C is an 8-bit counter register for T0 interval time control.

| 0D9H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **T0C** | T0C7 | T0C6 | T0C5 | T0C4 | T0C3 | T0C2 | T0C1 | T0C0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The equation of T0C initial value is as following.

***T0C initial value = 256 - (T0 interrupt interval time * input clock)***

- **Example: To set 10ms interval time for T0 interrupt. High clock is external 4MHz. Fcpu=Fosc/4. Select T0RATE=010 (Fcpu/64).**

*T0C initial value = 256 - (T0 interrupt interval time * input clock)*
*= 256 - (10ms * 4MHz / 4 / 64)*
*= 256 - ($10^{-2}$ * 4 * $10^{6}$ / 4 / 64)*
*= 100*
*= 64H*

***The basic timer table interval time of T0.***

| T0RATE | T0CLOCK | High speed mode (Fcpu = 4MHz / 4) | | Low speed mode (Fcpu = 32768Hz / 4) | |
|---|---|---|---|---|---|
| | | Max overflow interval | One step = max/256 | Max overflow interval | One step = max/256 |
| 000 | Fcpu/256 | 65.536 ms | 256 us | 8000 ms | 31250 us |
| 001 | Fcpu/128 | 32.768 ms | 128 us | 4000 ms | 15625 us |
| 010 | Fcpu/64 | 16.384 ms | 64 us | 2000 ms | 7812.5 us |
| 011 | Fcpu/32 | 8.192 ms | 32 us | 1000 ms | 3906.25 us |
| 100 | Fcpu/16 | 4.096 ms | 16 us | 500 ms | 1953.125 us |
| 101 | Fcpu/8 | 2.048 ms | 8 us | 250 ms | 976.563 us |
| 110 | Fcpu/4 | 1.024 ms | 4 us | 125 ms | 488.281 us |
| 111 | Fcpu/2 | 0.512 ms | 2 us | 62.5 ms | 244.141 us |

## 7.2.4 T0 TIMER OPERATION SEQUENCE

T0 timer operation sequence of setup T0 timer is as following.

☞ **Stop T0 timer counting, clear T0 interrupt request flag.**

```
BCLR    FT0ENB          ; T0 timer.
BCLR    FT0IRQ          ; T0 time out request flag is cleared.
```

☞ **Set T0 timer rate.**

```
MOV     A, #0xxx0000b   ;The T0 rate control bits exist in bit4~bit6 of T0M. The
                        ; value is from x000xxxxb~x111xxxxb.
MOV     T0M,A           ; T0 timer is disabled.
```

☞ **Set T0 interrupt interval time.**

```
MOV     A,#7FH
MOV     T0C,A           ; Set T0C value.
```

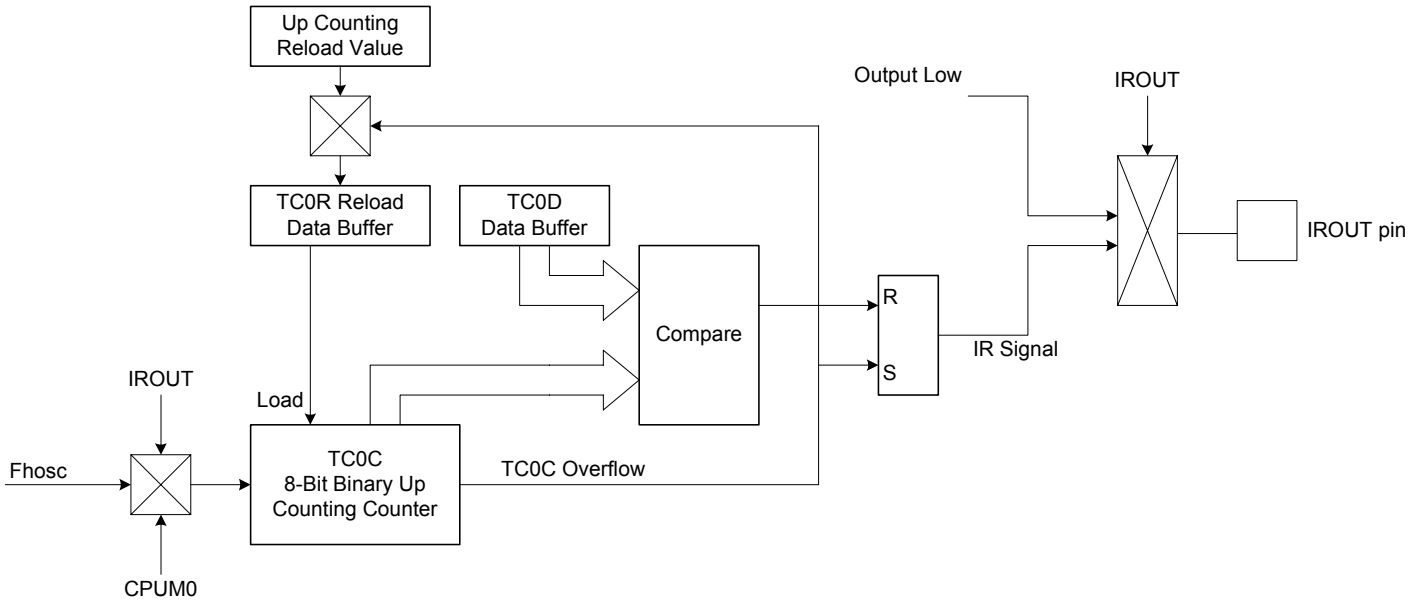
☞ **Enable T0 timer.**

```
BSET    FT0ENB          ; Enable T0 timer.
```

# 8 IR OUTPUT

## 8.1 OVERVIEW

IR signal is generated by TC0 timer. The IR output pin is IROUT pin. When IROUT bit of TC0M is logic “1”, IROUT pin outputs IR signal. If IROUT = 0 or system is in power down mode, IRPUT pin is tied to low status. The TC0 is an 8-bit binary up counting timer for IR signal generator. The IR signal is duty/cycle changeable type controlled by TC0R and TC0D. TC0R decides IR cycle and TC0D decides IR duty. TC0 clock source is only from Fhosc (external high clock source), eg. 4MHz crystal. If external oscillator is 4MHz, the TC0 clock rate is 4MHz. TC0 only generate IR output and no interrupt function. When enable IR output function (IROUT=1), IR output status is high level. TC0C initial value is TC0R and starts to count. When TC0C=TC0D, IR output status change to low level and finishes high duty operation. When TC0C overflow occurs (TC0C changes from 0xFF to 0x00), IR output low duty operation stops. System loads TC0R into TC0C automatically and next cycle starts.



# 8.2 IR CONTROL REGISTER

## 8.2.1 TC0M MODE REGISTER

| 0DAH | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **TC0M** | - | - | - | - | - | - | - | IROUT |
| Read/Write | - | - | - | - | - | - | - | R/W |
| After reset | - | - | - | - | - | - | - | 0 |

Bit 0 **IROUT:** IR output control bit.
0 = Disable IR output. IROUT pin is low status..
1 = Enable IR output. IR signal output to IROUT pin.

✷ ***Note: When IROUT=0 or the system is in power down mode, IROUT pin keeps low status.***

## 8.2.2 TC0C COUNTING REGISTER

TC0C is an 8-bit counter register for TC0 interval time control.

| 0DBH | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **TC0C** | TC0C7 | TC0C6 | TC0C5 | TC0C4 | TC0C3 | TC0C2 | TC0C1 | TC0C0 |
| Read/Write | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

✷ ***Note: Set TC0C=TC0R before IR output enable to make sure the first cycle correct.***

## 8.2.3 TC0R AUTO-LOAD REGISTER

**TC0R decides IR signal frequency.** TC0 timer is with auto-load function. When TC0C overflow occurs, TC0R value will load to TC0C. It is easy to generate an accurate time for IR signal cycle, and users don't reset TC0C during interrupt service routine.

TC0 is double buffer design. If new TC0R value is set by program, the new value is stored in 1st buffer. Until TC0 overflow occurs, the new value moves to real TC0R buffer.

| 0CDH | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **TC0R** | TC0R7 | TC0R6 | TC0R5 | TC0R4 | TC0R3 | TC0R2 | TC0R1 | TC0R0 |
| Read/Write | W | W | W | W | W | W | W | W |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The equation of TC0Rinitial value is as following.

***TC0R initial value = 256 - (TC0 interrupt interval time * input clock)***

✷ ***Note: The input clock is 4MHz of external 4MHz oscillator.***

➢ **Example: Set IR cycle frequency is 38KHz. Input clock is 4MHz.**

*TC0R initial value = 256 - (TC0 interrupt interval time * input clock)*

*TC0 interval time = 1/38KHz = 26.3us*
*Input clock = external oscillator 4MHz.*

*TC0R = 256 - (26.3us * 4MHz)*
*= 150.8*
***≈ 151***
***= 97h***

## 8.2.4 TC0D IR DUTY CONTROL REGISTER

The IR signal is duty changeable by TC0D. TC0D decides the IR output signal high pulse width length. When TC0C=TC0D, the IR signal changes from high pulse to low pulse. The low pulse stops when TC0C overflow. The high pulse width is TC0D-TC0R, and the low pulse width is 256-TC0D. It is easy to modulate IR duty/cycle by TC0R and TC0D registers.

| 0E8H | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 |
|---|---|---|---|---|---|---|---|---|
| **TC0D** | TC0D7 | TC0D6 | TC0D5 | TC0D4 | TC0D3 | TC0D2 | TC0D1 | TC0D0 |
| Read/Write | W | W | W | W | W | W | W | W |
| After reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

The equation of TC0Dinitial value is as following.

***TC0D initial value = TC0R + (256-TC0R) / (1/IR duty)***

➢ **Example: Set TC0D for 38KHz IR and duty is 1/3. Input clock is 4MHz.**

*TC0D initial value = TC0R + (256-TC0R) / (1/IR duty)*

*TC0R of 38KHz = 151*

*TC0D = 151 + (256-151)/(1/ (1/3))*
*= 186*
***= BAh***

***Common IR signal table. System clock is 4MHz.***

| IR Freq. (KHz) | TC0C TC0D | | TC0D | | | | | | Freq. Error Rate |
|---|---|---|---|---|---|---|---|---|---|
| | | | 1/2 duty | | 1/3 duty | | 1/4duty | | |
| | DEC | HEX | DEC | HEX | DEC | HEX | DEC | HEX | |
| 32 | 131 | 83 | 193.50 | C1 | 172.67 | AC | 162.25 | A2 | 0.00% |
| 36 | 145 | 91 | 200.50 | C8 | 182.00 | B6 | 172.75 | AC | 0.10% |
| 38 | 151 | 97 | 203.50 | CB | 186.00 | BA | 177.25 | B1 | 0.25% |
| 39.2 | 154 | 9A | 205.00 | CD | 188.00 | BC | 179.50 | B3 | 0.04% |
| 40 | 156 | 9C | 206.00 | CE | 189.33 | BD | 181.00 | B5 | 0.00% |
| 56 | 185 | B9 | 220.50 | DC | 208.67 | D0 | 202.75 | CA | 0.60% |

## 8.2.5 IR OUTPUT OPERATION SEQUENCE

☞ **Set TC0C and TC0R for IR cycle.**

```
MOV     A, #IRCYCVAL    ;TC0C, TC0R value for IR cycle.
MOV     TC0C, A
MOV     TC0R, A
```

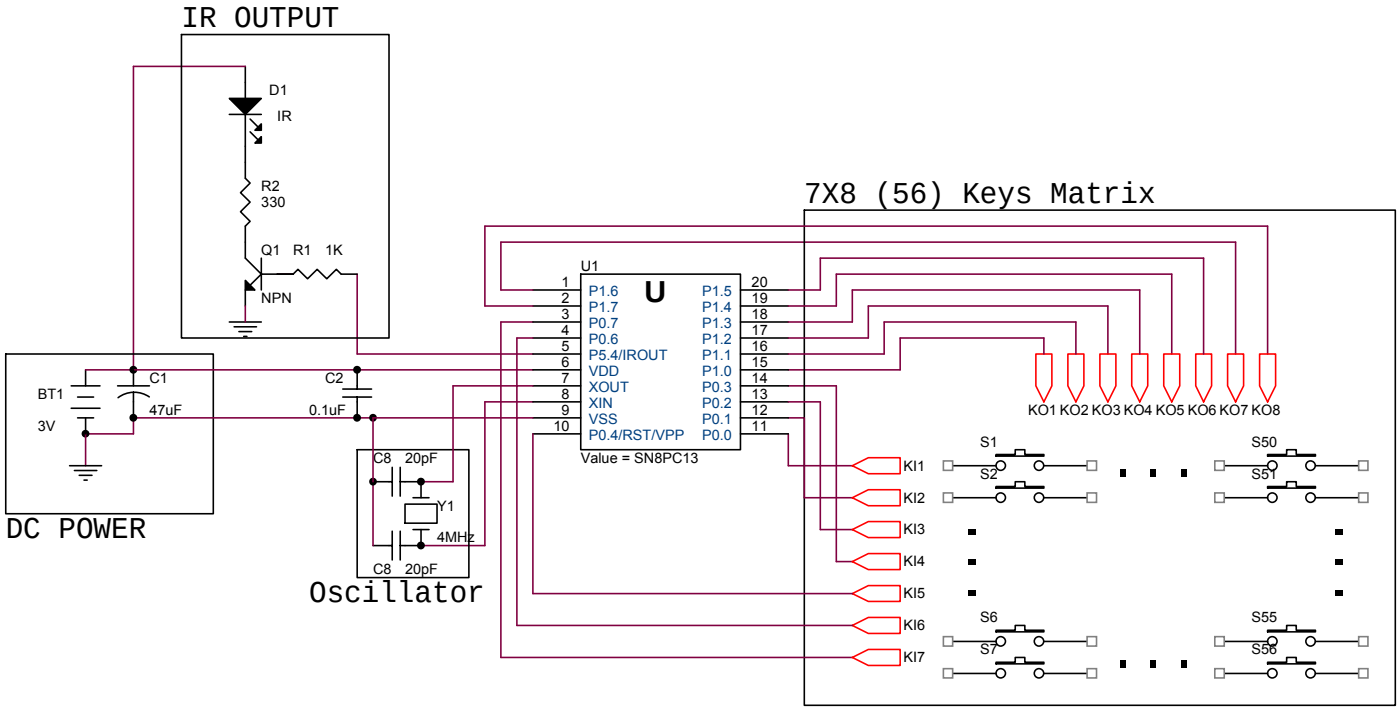
☞ **Set TC0D for IR duty.**

```
MOV     A, #IRDUTYVAL   ;TC0D value for IR duty.
MOV     TC0D, A
```

☞ **Enable IR output.**

```
BSET    FIROUT          ; Enable IR output from IROUT pin.
```

# 8.3 IR REMOTE CONTROLLER APPLICATION CIRCUIT




The application circuit is a typical IR remote controller circuit.

- Reset pin set as P0.4 input only pin for key matrix input pin. If the project needs external reset pin, the maximum key number is only 49 keys.
- The keys are low active type and pulled by P0 internal 100k ohm pull-up resistors.
- The system clock source is 4MHz crystal or ceramic.
- IR transmitter is driven by a transistor and there must be connected a resistor between IROUT pin and transistor's B terminal. The resistor value must be greater than 330 ohm better, or the current through IROUT pin is over standard 15mA.
- The IR power is supplied from battery power terminal and after C1 47uF capacitor to make sure power stable while IR transmitting.
- The IR circuit is drive type. The IROUT pin initial value is low status. When stop IR transmitting or the system is in power down mode, the IROUT pin keeps low status and won't be turn on to consume power. **The IR only supports drive type.**
- The C2 is MUC's power bypass capacitor. The value is 0.1uF and it must be beside IC as closer as possible.

# 9 INSTRUCTION TABLE

| Field | Mnemonic | | Description | C | Z | Cycle |
|---|---|---|---|---|---|---|
| MOVE | MOV | A,M | A ← M | - | √ | 1 |
| | MOV | M,A | M ← A | - | - | 1 |
| | MOV | A,I | A ← I | - | - | 1 |
| | MOVC | | R, A ← ROM [Y,Z] | - | - | 2 |
| ARITHMETIC | ADC | A,M | A ← A + M + C, if occur carry, then C=1, else C=0 | √ | √ | 1 |
| | SBC | A,M | A ← A - M - /C, if occur borrow, then C=0, else C=1 | √ | √ | 1 |
| LOGIC | AND | A,M | A ← A and M | - | √ | 1 |
| | OR | A,M | A ← A or M | - | √ | 1 |
| | XOR | A,M | A ← A xor M | - | √ | 1 |
| | RRC | M | A ← RRC M | √ | - | 1 |
| | BCLR | M.b | M.b ← 0 | - | - | 1 |
| | BSET | M.b | M.b ← 1 | - | - | 1 |
| BRANCH | CMPRS | A,I | ZF,C ← A - I, If A = I, then skip next instruction | √ | √ | 1+S |
| | INCMS | M | M ← M + 1, If M = 0, then skip next instruction | - | - | 1+S |
| | DECMS | M | M ← M - 1, If M = 0, then skip next instruction | - | - | 1+S |
| | BTS0 | M.b | If M.b = 0, then skip next instruction | - | - | 1+S |
| | BTS1 | M.b | If M.b = 1, then skip next instruction | - | - | 1+S |
| | JMP | d | PC15/14 ← RomPages1/0, PC13~PC0 ← d | - | - | 2 |
| | CALL | d | Stack ← PC15~PC0, PC15/14 ← RomPages1/0, PC13~PC0 ← d | - | - | 2 |
| MIS | RET | | PC ← Stack | - | - | 2 |
| | NOP | | No operation | - | - | 1 |

***Note: 1. "M" is system register or RAM.***
***2. If branch condition is true then "S = 1", otherwise "S = 0".***

# 10 ELECTRICAL CHARACTERISTIC

## 10.1 ABSOLUTE MAXIMUM RATING

Supply voltage (Vdd)..... - 0.3V ~ 6.0V
Input in voltage (Vin)..... Vss – 0.2V ~ Vdd + 0.2V
Operating ambient temperature (Topr), SN8PC13P, SN8PC13S, SN8PC13X ..... 0°C ~ + 70°C
Storage ambient temperature (Tstor) ..... –40°C ~ + 125°C

## 10.2 ELECTRICAL CHARACTERISTIC

*(All of voltages refer to Vss, Vdd = 5.0V, fosc = 4MHz,Fcpu=1MHZ, ambient temperature is 25℃ unless otherwise note.)*

| PARAMETER | SYM. | DESCRIPTION | | MIN. | TYP. | MAX. | UNIT |
|---|---|---|---|---|---|---|---|
| Operating voltage | Vdd | Normal mode, Vpp = Vdd. | | LVD | - | 5.5 | V |
| RAM Data Retention voltage | Vdr | | | - | 1.5* | - | V |
| Vdd rise rate | Vpor | Vdd rise rate to ensure internal power-on reset | | 0.05 | - | - | V/ms |
| Input Low Voltage | ViL1 | All input ports | | Vss | - | 0.3Vdd | V |
| | ViL2 | Reset pin | | Vss | - | 0.2Vdd | V |
| Input High Voltage | ViH1 | All input ports | | 0.7Vdd | - | Vdd | V |
| | ViH2 | Reset pin | | 0.9Vdd | - | Vdd | V |
| Reset pin leakage current | Ilekg | Vin = Vdd | | - | - | 2 | uA |
| I/O port pull-up resistor | Rup | Vin = Vss , Vdd = 3V | | 100 | 200 | 300 | KΩ |
| | | Vin = Vss , Vdd = 5V | | 50 | 100 | 150 | |
| I/O port input leakage current | Ilekg | Pull-up resistor disable, Vin = Vdd | | - | - | 2 | uA |
| I/O output source current | IoH | Vop = Vdd – 0.5V | | - | 12* | - | mA |
| sink current | IoL | Vop = Vss + 0.5V | | - | 15* | - | |
| Supply Current | Idd1 | normal Mode (No loading) | Vdd= 5V, Fcpu=4Mhz/4 | - | 2.5 | 5 | mA |
| | | | Vdd= 3V, Fcpu=4Mhz/4 | - | 1 | 2 | mA |
| | Idd2 | Sleep Mode | Vdd= 5V | - | 0.8 | 1.6 | uA |
| | | | Vdd= 3V | - | 0.7 | 1.4 | uA |
| LVD Voltage | Vdet0 | Low voltage reset level. 0ºC~70ºC. | | 1.6 | 2.0 | 2.1 | V |

***These parameters are for design reference, not tested.**

# 11 DEVELOPMENT TOOL

SN8PC13 development tools are as following.

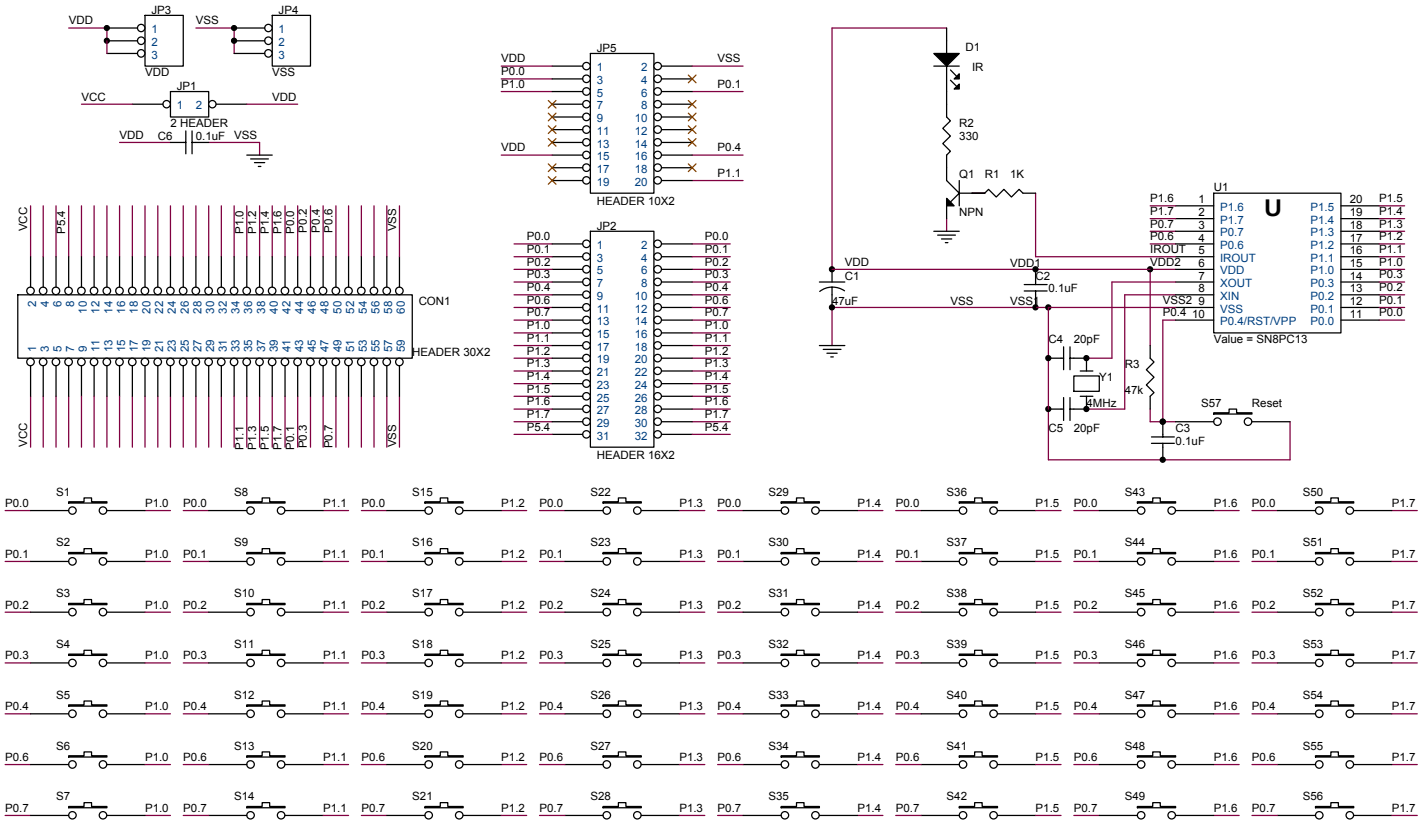
- **ICE version: SN8ICE2K.**
- **IDE version: M2IDE_V107 later.**
- **Writer: EZ-writer, MP-writer. Use "SN8PC13 MPxxxx" transition board.**
- **SN8PC13 Ev-kit.**

SN8PC13 IDE files haven't included into M2IDE. Please set these files into IDE paths as following.

- Move "**SN8PC13.bit**" into "**C:\Sonix\M2IDE_Vxxx\Bit_File**".
- Move "**SN8PC13.bol**" into "**C:\Sonix\M2IDE_Vxxx\Symbol**".
- Move "**SN8PC13.inc**" into "**C:\Sonix\M2IDE_Vxxx\use_inc2**".
- Move "**SN8PC13.prg**" into "**C:\Sonix\M2IDE_Vxxx\use_prg**".

Include these file into IDE path, the M2IDE can simulate and program SN8PC13.

For IR remote controller development, Sonix provides a "SN8PC13 EV-kit" and supports M2IDE_V107 and later IDE. The Ev-kit includes 56keys and IR output circuit. The EV-kit circuit is as following.



# 12 OTP PROGRAMMING PIN

## 12.1 The pin assignment of Easy Writer transition board socket:

**Easy Writer JP1/JP2**

| | | | |
|---|---|---|---|
| VSS | 2 | 1 | VDD |
| CE | 4 | 3 | CLK/PGCLK |
| OE/ShiftDat | 6 | 5 | PGM/OTPCLK |
| D0 | 8 | 7 | D1 |
| D2 | 10 | 9 | D3 |
| D4 | 12 | 11 | D5 |
| D6 | 14 | 13 | D7 |
| VPP | 16 | 15 | VDD |
| RST | 18 | 17 | HLS |
| ALSB/PDB | 20 | 19 | - |

**JP1 for MP transition board**

**Easy Writer JP3 (Mapping to 48-pin text tool)**

| | | | |
|---|---|---|---|
| DIP1 | 1 | 48 | DIP48 |
| DIP2 | 2 | 47 | DIP47 |
| DIP3 | 3 | 46 | DIP46 |
| DIP4 | 4 | 45 | DIP45 |
| DIP5 | 5 | 44 | DIP44 |
| DIP6 | 6 | 43 | DIP43 |
| DIP7 | 7 | 42 | DIP42 |
| DIP8 | 8 | 41 | DIP41 |
| DIP9 | 9 | 40 | DIP40 |
| DIP10 | 10 | 39 | DIP39 |
| DIP11 | 11 | 38 | DIP38 |
| DIP12 | 12 | 37 | DIP38 |
| DIP13 | 13 | 36 | DIP36 |
| DIP14 | 14 | 35 | DIP35 |
| DIP15 | 15 | 34 | DIP34 |
| DIP16 | 16 | 33 | DIP33 |
| DIP17 | 17 | 32 | DIP32 |
| DIP18 | 18 | 31 | DIP31 |
| DIP19 | 19 | 30 | DIP30 |
| DIP20 | 20 | 29 | DIP29 |
| DIP21 | 21 | 28 | DIP28 |
| DIP22 | 22 | 27 | DIP27 |
| DIP23 | 23 | 26 | DIP26 |
| DIP24 | 24 | 25 | DIP25 |

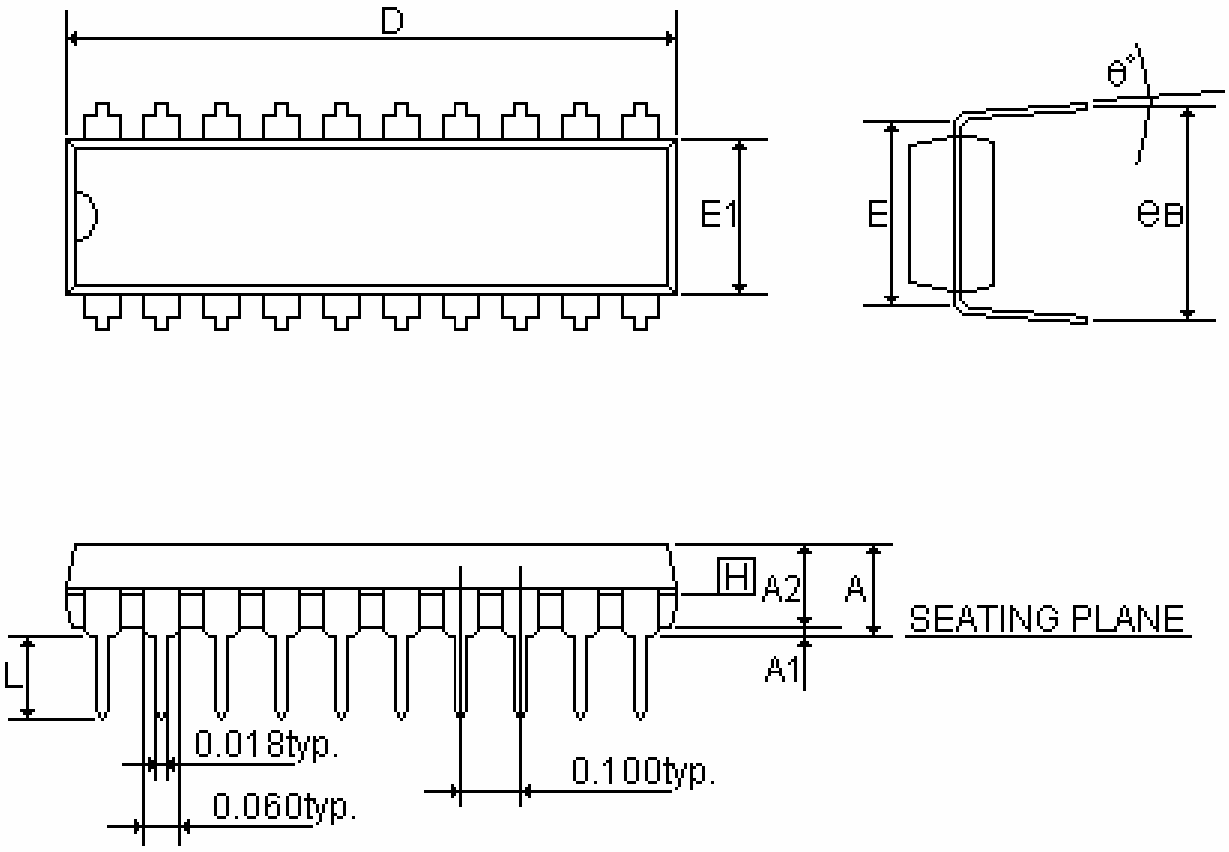
**JP3 for MP transition board**

## 12.2 Programming Pin Mapping:

| Programming Information of SN8PC13 | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Chip Name | | SN8PC13P/S/X | | | | | | | |
| EZ Writer / MP Writer Connector | | OTP IC / JP3 Pin Assigment | | | | | | | |
| Number | Name | Number | Pin | | | | | | |
| 1 | VDD | 6 | VDD | | | | | | |
| 2 | GND | 9 | VSS | | | | | | |
| 3 | CLK | 11 | P0.0 | | | | | | |
| 4 | CE | - | - | | | | | | |
| 5 | PGM | 15 | P1.0 | | | | | | |
| 6 | OE | 12 | P0.1 | | | | | | |
| 7 | D1 | - | - | | | | | | |
| 8 | D0 | - | - | | | | | | |
| 9 | D3 | - | - | | | | | | |
| 10 | D2 | - | - | | | | | | |
| 11 | D5 | - | - | | | | | | |
| 12 | D4 | - | - | | | | | | |
| 13 | D7 | - | - | | | | | | |
| 14 | D6 | - | - | | | | | | |
| 15 | VDD | - | - | | | | | | |
| 16 | VPP | 10 | RST | | | | | | |
| 17 | HLS | - | - | | | | | | |
| 18 | RST | - | - | | | | | | |
| 19 | - | - | - | | | | | | |
| 20 | ALSB/PDB | 16 | P1.1 | | | | | | |

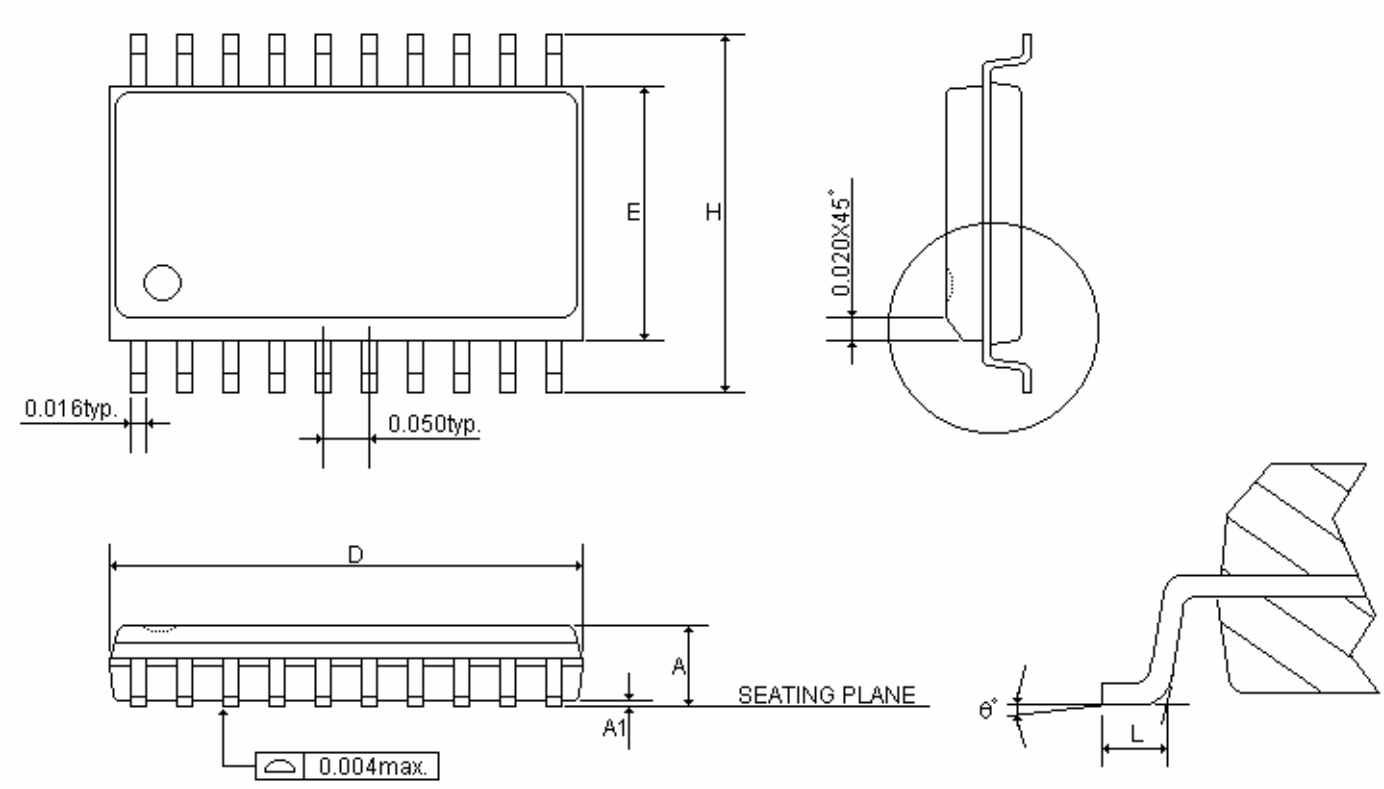
# 13 PACKAGE INFORMATION

## 13.1 P-DIP 20 PIN



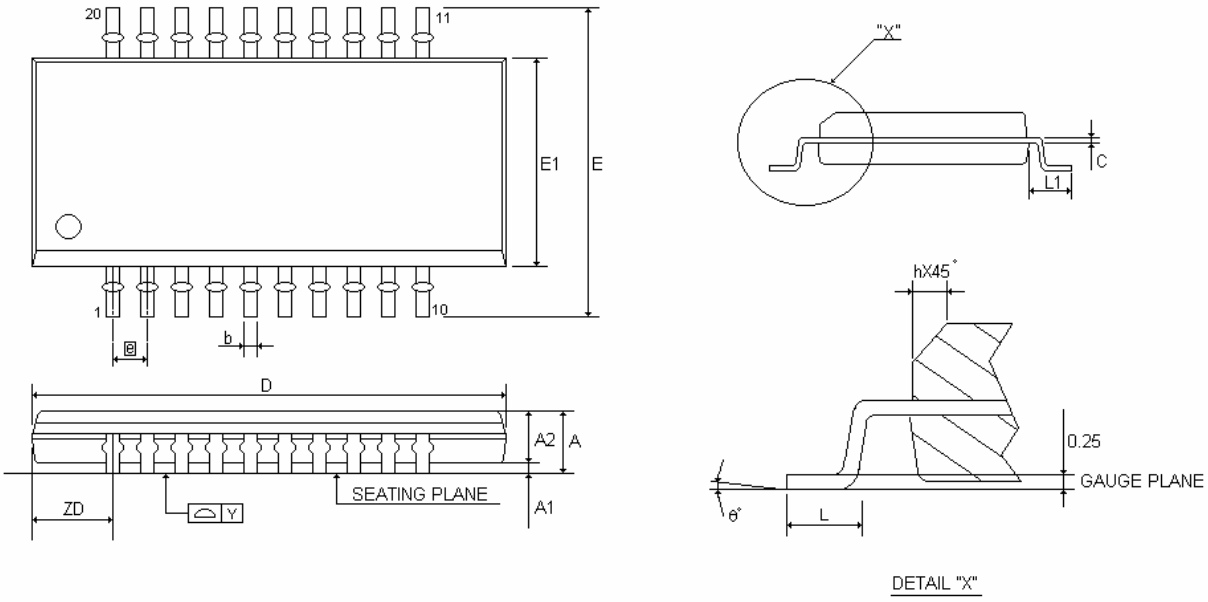

| SYMBOLS | MIN | NOR | MAX | MIN | NOR | MAX |
|---|---|---|---|---|---|---|
| | (inch) | | | (mm) | | |
| A | - | - | 0.210 | - | - | 5.334 |
| A1 | 0.015 | - | - | 0.381 | - | - |
| A2 | 0.125 | 0.130 | 0.135 | 3.175 | 3.302 | 3.429 |
| D | 0.980 | 1.030 | 1.060 | 24.892 | 26.162 | 26.924 |
| E | | 0.300 | | | 7.620 | |
| E1 | 0.245 | 0.250 | 0.255 | 6.223 | 6.350 | 6.477 |
| L | 0.115 | 0.130 | 0.150 | 2.921 | 3.302 | 3.810 |
| eB | 0.335 | 0.355 | 0.375 | 8.509 | 9.017 | 9.525 |
| θ° | 0° | 7° | 15° | 0° | 7° | 15° |

## 13.2 SOP 20 PIN




| SYMBOLS | MIN | NOR | MAX | MIN | NOR | MAX |
|---|---|---|---|---|---|---|
| | (inch) | | | (mm) | | |
| A | 0.093 | 0.099 | 0.104 | 2.362 | 2.502 | 2.642 |
| A1 | 0.004 | 0.008 | 0.012 | 0.102 | 0.203 | 0.305 |
| D | 0.496 | 0.502 | 0.508 | 12.598 | 12.751 | 12.903 |
| E | 0.291 | 0.295 | 0.299 | 7.391 | 7.493 | 7.595 |
| H | 0.394 | 0.407 | 0.419 | 10.008 | 10.325 | 10.643 |
| L | 0.016 | 0.033 | 0.050 | 0.406 | 0.838 | 1.270 |
| θ° | 0° | 4° | 8° | 0° | 4° | 8° |

## 13.3 SSOP 20 PIN



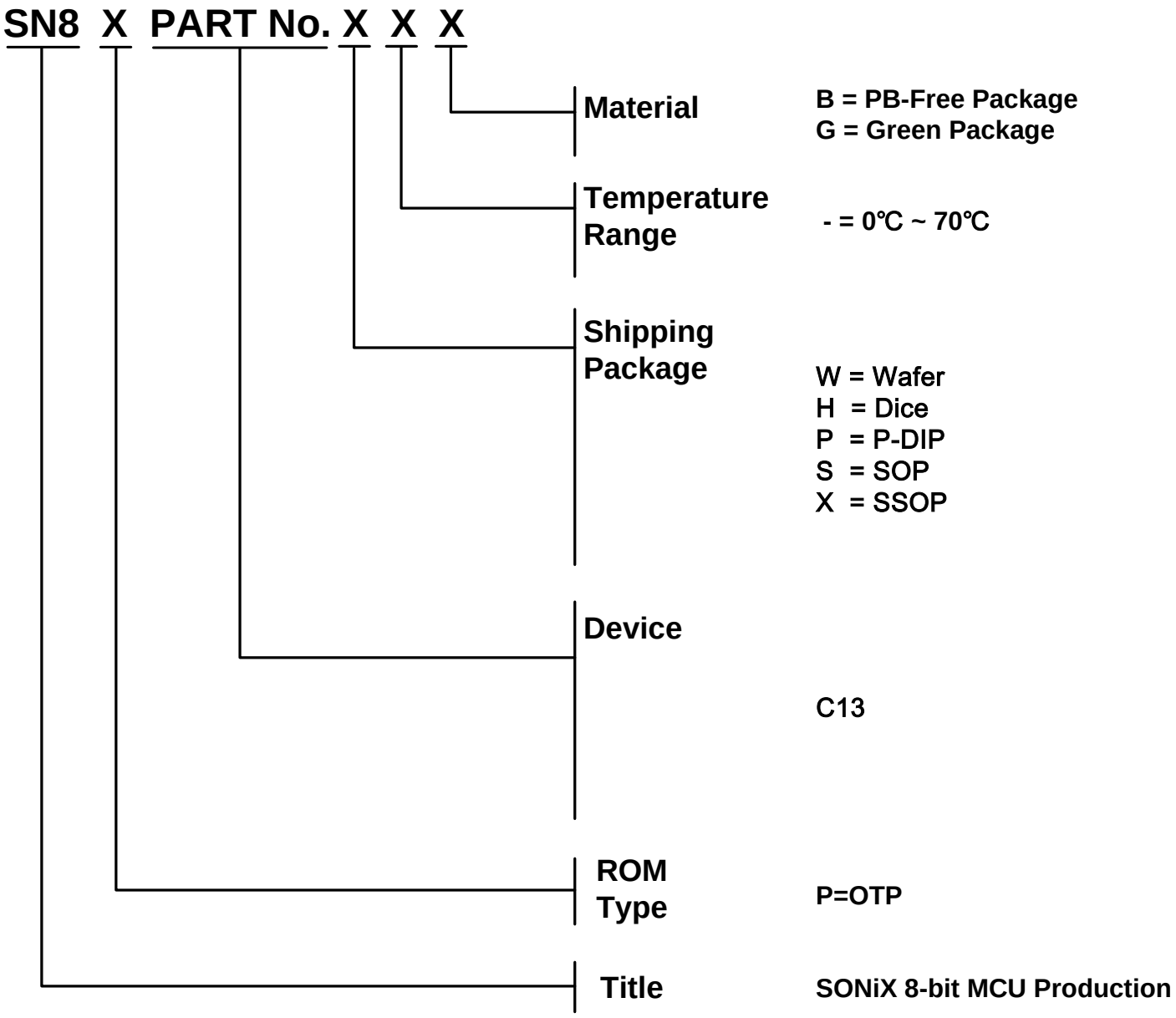

| SYMBOLS | MIN | NOR | MAX | MIN | NOR | MAX |
|---|---|---|---|---|---|---|
| | (inch) | | | (mm) | | |
| A | 0.053 | 0.063 | 0.069 | 1.350 | 1.600 | 1.750 |
| A1 | 0.004 | 0.006 | 0.010 | 0.100 | 0.150 | 0.250 |
| A2 | - | - | 0.059 | - | - | 1.500 |
| b | 0.008 | 0.010 | 0.012 | 0.200 | 0.254 | 0.300 |
| c | 0.007 | 0.008 | 0.010 | 0.180 | 0.203 | 0.250 |
| D | 0.337 | 0.341 | 0.344 | 8.560 | 8.660 | 8.740 |
| E | 0.228 | 0.236 | 0.244 | 5.800 | 6.000 | 6.200 |
| E1 | 0.150 | 0.154 | 0.157 | 3.800 | 3.900 | 4.000 |
| [e] | | 0.025 | | | 0.635 | |
| h | 0.010 | 0.017 | 0.020 | 0.250 | 0.420 | 0.500 |
| L | 0.016 | 0.025 | 0.050 | 0.400 | 0.635 | 1.270 |
| L1 | 0.039 | 0.041 | 0.043 | 1.000 | 1.050 | 1.100 |
| ZD | | 0.059 | | | 1.500 | |
| Y | - | - | 0.004 | - | - | 0.100 |
| θ° | 0° | - | 8° | 0° | - | 8° |

# 14 Marking Definition

## 14.1 INtroduction

There are many different types in Sonix 8-bit MCU production line. This note listed the production definition of all 8-bit MCU for order or obtain information. This definition is only for Blank OTP MCU.

## 14.2 Marking indetification system

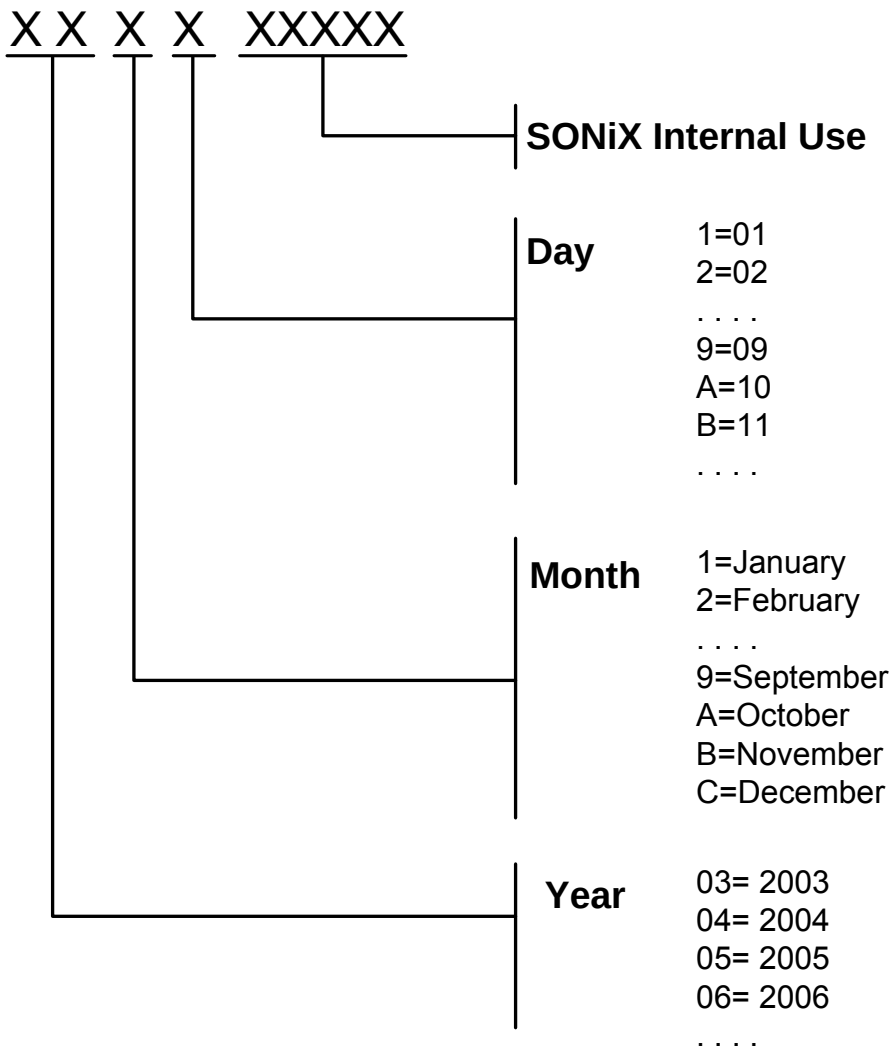


## 14.3 Marking Example

| Name | ROM Type | Device | Package | Temperature | Material |
|---|---|---|---|---|---|
| **SN8PC13PG** | OTP | C13 | P-DIP | 0℃~70℃ | Green Package |
| **SN8PC13SB** | OTP | C13 | SOP | 0℃~70℃ | PB-Free Package |

## 14.4 Datecode system

There are total 8~9 letters of SONiX datecode system. The final four or five char. are for Sonix inside use only, and the first 4 indicate the Package date including Year/Month/Date. The detail information is following:



**Main Office:**
Address: 9F, NO. 8, Hsien Cheng 5th St, Chupei City, Hsinchu, Taiwan R.O.C.
Tel: 886-3-551 0520
Fax: 886-3-551 0523

**Taipei Office:**
Address: 15F-2, NO. 171, Song Ted Road, Taipei, Taiwan R.O.C.
Tel: 886-2-2759 1980
Fax: 886-2-2759 8180

**Hong Kong Office:**
Address: Flat 3 9/F Energy Plaza 92 Granville Road, Tsimshatsui East Kowloon.
Tel: 852-2723 8086
Fax: 852-2723 9179

**Technical Support by Email:**
Sn8fae@sonix.com.tw