SN8P26L38

USER S MANUAL

Version 1.1

SN8P26L38

SONiX 8-Bit Micro-Controller

SONIX reserves the right to make change without further notice to any products herein to improve reliability, function or design. SONIX does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. SONIX products are not designed, intended, or authorized for us as components in systems intended, for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the SONIX product could create a situation where personal injury or death may occur. Should Buyer purchase or use SONIX products for any such unintended or unauthorized application. Buyer shall indemnify and hold SONIX and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use even if such claim alleges that SONIX was negligent regarding the design or manufacture of the part.

AMENDMENT HISTORY

Version	Date	Description
VER 0.1	Dec. 2007	First Issue.
VER 0.2	Jan. 2008	Modify system register table.
VER 0.3	Feb. 2008	Modify internal low RC fy0eR8c (o) Tj-0.316836.56 0.72 0.72 re f166.32 736.5o0.16224 T-T916

Table of Content

AMENDMENT HISTORY 2

1 PRODUCT OVERVIEW 8

1.1 FEATURES 8
1.2 SYSTEM BLOCK DIAGRAM 9
1.3 PIN ASSIGNMENT 10
1.4 PIN DESCRIPTIONS 11
1.5 PIN CIRCUIT DIAGRAMS 12

2 CENTRAL PROCESSOR UNIT (CPU) 14

2.1 PROGRAM MEMORY (ROM) 14
2.1.1 RESET VECTOR (0000H) *15*
2.1.2 INTERRUPT VECTOR (0008H) *16*
2.1.3 LOOK-UP TABLE DESCRIPTION *18*
2.1.4 JUMP TABLE DESCRIPTION *20*
2.1.5 CHECKSUM CALCULATION *22*
2.2 DATA MEMORY (RAM) 23
2.2.1 SYSTEM REGISTER *24*
2.2.1.1 SYSTEM REGISTER TABLE 24
2.2.1.2 SYSTEM REGISTER DESCRIPTION 24
2.2.1.3 BIT DEFINITION of SYSTEM REGISTER 25
2.2.2 ACCUMULATOR *27*
2.2.3 PROGRAM FLAG *28*
2.2.4 PROGRAM COUNTER *29*
2.2.5 H, L REGISTERS *32*
2.2.6 Y, Z REGISTERS *33*
2.2.7 R REGISTERS *34*
2.3 ADDRESSING MODE 35
2.3.1 IMMEDIATE ADDRESSING MODE *35*
2.3.2 DIRECTLY ADDRESSING MODE *35*
2.3.3 INDIRECTLY ADDRESSING MODE *35*
2.4 STACK OPERATION 36
2.4.1 OVERVIEW *36*
2.4.2 STACK REGISTERS *37*
2.4.3 STACK OPERATION EXAMPLE *38*
2.5 CODE OPTION TABLE 39
2.5.1 RESET_PIN CODE OPTION *39*
2.5.2 SECURITY CODE OPTION *39*

5.7.3 P1W WAKEUP CONTROL REGISTER *61*

6 INTERRUPT **62**

6.1 OVERVIEW 62
6.2 INTEN INTERRUPT ENABLE REGISTER 63
6.3 INTRQ INTERRUPT REQUEST REGISTER 63
6.4 GIE GLOBAL INTERRUPT OPERATION 64
6.5 PUSH, POP ROUTINE 65
6.6 EXTERNAL INTERRUPT OPERATION (INT0) 66
6.7 INT1 (P0.1) INTERRUPT OPERATION 68
6.8 T0 INTERRUPT OPERATION 69
6.9 T1 INTERRUPT OPERATION 71
6.10 TC1 INTERRUPT OPERATION 72
6.11 COMPARATOR INTERRUPT OPERATION (CMP0, CMP1) 73
6.12 SIO INTERRUPT OPERATION 74
6.13 UART INTERRUPT OPERATION 75
6.14 MULTI-INTERRUPT OPERATION 76

7 I/O PORT **77**

7.1 OVERVIEW 77
7.2 I/O PORT MODE REGISTER 78

8.4.4 TC1R AUTO-LOAD REGISTER *96*
8.4.5 TC1 CLOCK FREQUENCY OUTPUT (BUZZER) *97*
8.4.6 TC1 TIMER OPERATION SEQUENCE *98*
8.5 PWM1 MODE 99
8.5.1 OVERVIEW *99*
8.5.2 TC1IRQ AND PWM DUTY *100*
8.5.3 PWM PROGRAM EXAMPLE *101*
8.5.4 PWM1 DUTY CHANGING NOTICE *102*

9 ANALOG COMPARATOR 104

9.1 OVERVIEW 104
9.2 CMP0M REGISTER 106
9.3 CMP1M REGISTER 107
9.4 ANALOG COMPARATOR APPLICATION 108

10 IR OUTPUT 110

10.1 OVERVIEW 110
10.2 IR CONTROL REGISTER 111
10.2.1 IRM MODE REGISTER *111*
10.2.2 IRC COUNTING REGISTER *111*
10.2.3 IRR AUTO-LOAD REGISTER *112*
10.2.4 IRD IR DUTY CONTROL REGISTER *113*
10.2.5 IR OUTPUT OPERATION SEQUENCE *114*

11 SERIAL INPUT/OUTPUT TRANSCEIVER (SIO) 115

11.1 OVERVIEW 115
11.2 SIO OPERATION 115
11.3 SIOM MODE REGISTER 117
11.4 SIOB DATA BUFFER 118
11.5 SIOR REGISTER DESCRIPTION 119

12 UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART) 120

12.1 OVERVIEW 120
12.2 UART OPERATION 120
12.3 UART RECEIVER CONTROL REGISTER 123
12.4 UART TRANSMITTER CONTROL REGISTER 123
12.5 UART BAUD RATE CONTROL REGISTER 124
12.6 UART DATA BUFFER 125

13 INSTRUCTION TABLE 126

14 ELECTRICAL CHARACTERISTIC 127

1 PRODUCT OVERVIEW

1.1 FEATURES

- **Memory configuration**
  OTP ROM size: 8K * 16 bits.
  RAM size: 880 * 8 bits.

- **8 levels stack buffer**

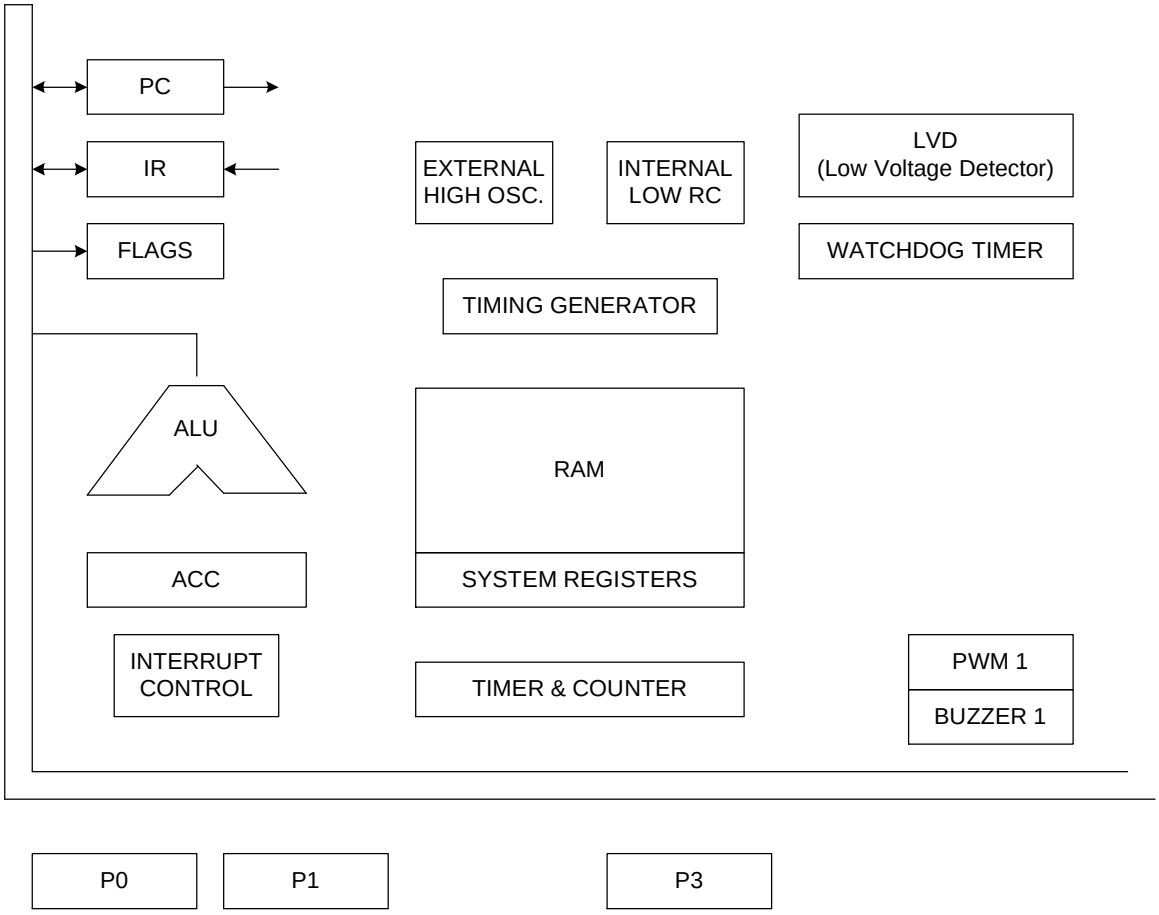
- **I/O pin configuration**
  Bi-directional: P0, P1, P2, P3, P4, P5
  Programmable open-drain: P1.0, P1.1, P5.0~P5.2, P3.2, P3.3.
  Wakeup:P0, P1 level change trigger.
  P1 wake-up function controlled by P1W.
  Pull-up resistors: P0, P1, P2, P3, P4, P5pl

- **Two 8-bit Timer/Counter**
  T0: Basic timer.
  TC1: Auto-reload timer/counter.

- **One RTC timer (T0).**
- **One channels PWM output.**
- **One channels buzzer output.**
- **One channel IR output (duty/cycle programmable PWM, TC0).**
- **On chip watchdog timer and clock source is internal low clock RC type (about 10KHz @3V).**

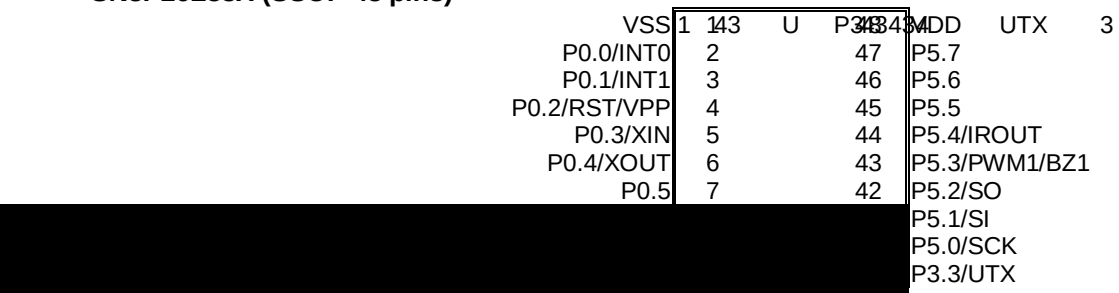
On-chip peripherals

1.2 SYSTEM BLOCK DIAGRAM



1.3 PIN ASSIGNMENT

SN8P26L38P (P-DIP 48 pins)
SN8P26L38X (SSOP 48 pins)



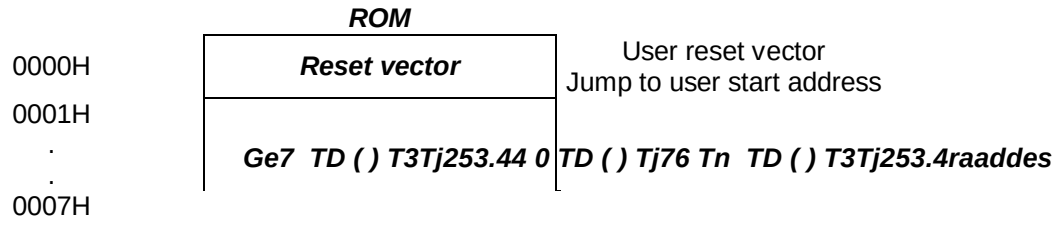
		PWM1: Programmable PWM output pin from TC1. P5.4: Bi-direction pin. Schmitt trigger structure as mimmat Pru.Min.pi-u Presitert

D 0 Tc (P)TO g BT133.48.24 1.478 re re f31.92 708(I/O .478 re85.2 f118.56 719.76 0.478 re2 re f162 719.744 4 0.778 re4e f161.28

2 CENTRAL PROCESSOR UNIT (CPU)

2.1 PROGRAM MEMORY (ROM)

F ***8K words ROM***



tQq.08448 Tc (vec) Ti0 Tc (t) Ti0.08112 Tc (o) Ti-0.08112 Tc (r) Ti/F5 10.08 Tf120.72 5.52 TD (U) Ti-0.24 Tc (s) Ti-0

2.1.1 RESET VECTOR (0000H)

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

- **Power On Reset (NT0=1, NPD=0).**
- **Watchdog Reset (NT0=0, NPD=0).**
- **External Reset (NT0=1, NPD=1).**

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. It is easy to know reset status from NT0, NPD flags of PFLAG register. The following example shows the way to define the reset vector in the program memory.

Ø 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
```

Ø Example: Defining Interrupt Vector. The interrupt service routine is following user program.

```
.CODE
                ORG     0               ; 0000H
                JMP     START           ; Jump to user program address.
                 "
                ORG     8               ; Interrupt vector.
                JMP     MY_IRQ          ; 0008H, Jump to interrupt service routine address.

                ORG     10H
START:                                  ; 0010H, The head of user program.
                 "                      ; User program.
                 "
                 "
                JMP     START           ; End of user program.
                 "
MY_IRQ:                                 ;The head of interrupt service routine.
                PUSH                    ; S
```

2.1.3 LOOK-UP TABLE DESCRIPTION

In the ROM's data lookup function, Y register is pointed to middle byte address (bit 8~bit 15) and Z register is pointed to low byte address (bit 0~bit 7) of ROM. After MOVC instruction executed, the low-byte data will be stored in ACC and high-byte data stored in R register.

Ø Example: To look up the ROM data located "TABLE1".

```
                B0MOV   Y, #TABLE1$M    ; To set lookup table1's middle address
                B0MOV   Z, #TABLE1$L    ; To set lookup table1's low address.
                MOVC                    ; To lookup data, R = 00H, ACC = 35H
                                        ; Increment the index address for next address.
                INCMS   Z               ; Z+1
                JMP     @F              ; Z is not overflow.
                INCMS   Y               ; Z overflow (FFH à 00), à Y=Y+1
                NOP                     ;
                                        ;
@@:             MOVC                    ; To lookup data, R = 51H, ACC = 05H.
                "                       ;
TABLE1:         DW      0035H           ; To define a word (16 bits) data.
                DW      5105H
                DW      2012H
                "
```

- ***Note: The Y register will not increase automatically when Z register crosses boundary from 0xFF to 0x00. Therefore, user must t***

Ø Example: Modify above example by "INC_YZ" macro.

```
                B0MOV     Y, #TABLE1$M    ; To set lookup table1's middle address
                B0MOV     Z, #TABLE1$L    ; To set lookup table1's low address.
                MOVC                      ; To lookup data, R = 00H, ACC = 35H

                INC_YZ                    ; Increment the index address for next address.
                                          ;
@@:             MOVC                      ; To lookup data, R = 51H, ACC = 05H.
                "                         ;
TABLE1:         DW        0035H           ; To define a word (16 bits) data.
                DW        5105H
                DW        2012H
                "
```

The other example of look-up table is to add Y or Z index register by accumulator. Please be careful if carry! happen.

Ø Example: Increase Y and Z register by B0ADD/ADD instruction.

```
                B0MOV     Y, #TABLE1$M    ; To set lookup table's middle address.
                B0MOV     Z, #TABLE1$L    ; To set lookup table's low address.

                B0MOV     A, BUF          ; Z = Z + BUF.
                B0ADD     Z, A

                B0BTS1    FC              ; Check the carry flag.
                JMP       GETDATA         ; FC = 0
                INCMS     Y               ; FC = 1. Y+1.
                NOP
GETDATA:                                  ;
                MOVC                      ; To lookup data. If BUF = 0, data is 0x0035
                                          ; If BUF = 1, data is 0x5105
                                          ; If BUF = 2, data is 0x2012
                "

TABLE1:         DW        0035H           ; To define a word (16 bits) data.
                DW        5105H
                DW        2012H
                "
```

Ø Example: "@JMP_A" application in SONIX macro file called "MACRO3.H".

```
B0MOV     A, BUF0      ;  BUF0! is from 0 to 4.
@JMP_A    5            ; The number of the jump table listing is five.
JMP       A0POINT      ; 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
JMP       A4POINT      ; ACC = 4, jump to A4POINT
```

If the jump table position is across a ROM boundary (0x00FF~0x0100), the @JMP_A! macro will adjust the jump table routine begin from next RAM boundary (0x0100).

Ø Example: "@JMP_A" operation.

; Before compiling program.

```
ROM address
          B0MOV     A, BUF0      ;  BUF0! is from 0 to 4.
          @JMP_A    5            ; The number of the jump table listing is five.
0X00FD    JMP       A0POINT      ; ACC = 0, jump to A0POINT
0X00FE    JMP       A1POINT      ; ACC = 1, jump to A1POINT
0X00FF    JMP       A2POINT      ; ACC = 2, jump to A2POINT
0X0100    JMP       A3POINT      ; ACC = 3, jump to A3POINT
0X0101    JMP       A4POINT      ; ACC = 4, jump to A4POINT
```

; After compiling program.

```
ROM address
          B0MOV     A, BUF0      ;  BUF0! is from 0 to 4.
          @JMP_A    5            ; The number of the jump table listing is five.
0X0100    JMP       A0POINT      ; ACC = 0, jump to A0POINT
0X0101    JMP       A1POINT      ; ACC = 1, jump to A1POINT
0X0102    JMP       A2POINT      ; ACC = 2, jump to A2POINT
0X0103    JMP       A3POINT      ; ACC = 3, jump to A3POINT
0X0104    JMP       A4POINT      ; ACC = 4, jump to A4POINT
```

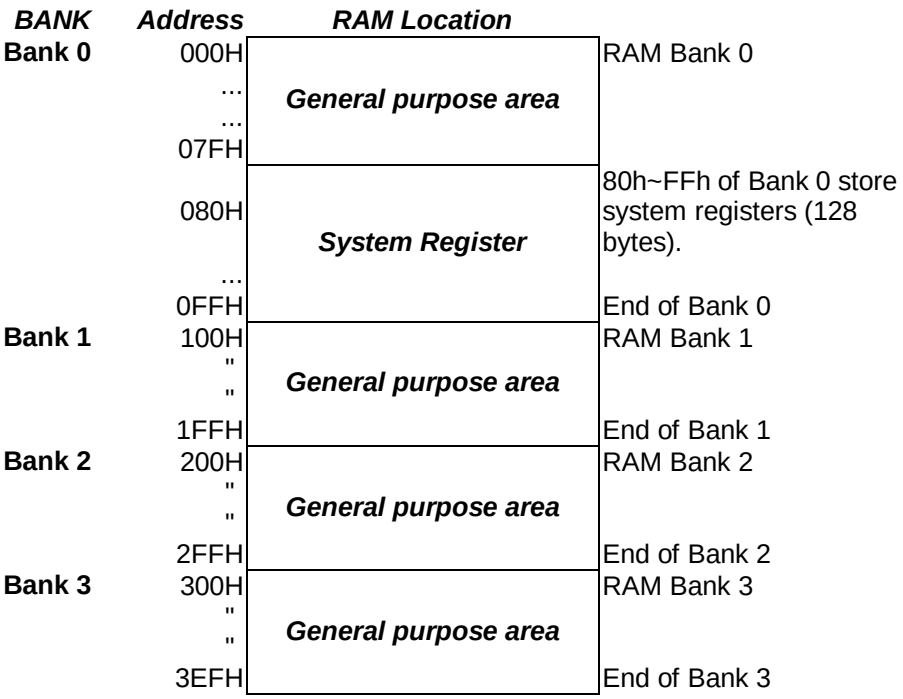
2.1.5 CHECKSUM CALCULATION

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

Ø Example: The demo program shows how to calculated Checksum from 00H to the e le hmw l"

2.2 DATA MEMORY (RAM)

F ***880 X 8-bit RAM***



The 880-byte general purpose RAM is separated into Bank 0~Bank 3. Accessing the two r

2.2.1.3BIT DEFINITION of SYSTEM REGISTER

Address	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	R/W	Remarks
080H	LBIT7	LBIT6	LBIT5	LBIT4	LBIT3	LBIT2	LBIT1	LBIT0	R/W	L
081H	HBIT7	HBIT6	HBIT5	HBIT4	HBIT3	HBIT2	HBIT1	HBIT0	R/W	

0F5H	1	1	1	[illegible]						
0F6H	S4PC7	S4PC6	S4PC5	S4PC4	S4PC3	S4PC2	S4PC1	S4PC0	R/W	STK4L
0F7H	1	1	1	S4PC12	S4PC11	S4PC10	S4PC9	S4PC8	R/W	STK4H
0F8H	S3PC7	S3PC6	S3PC5	S3PC4	S3PC3	S3PC2	S3PC1	S3PC0	R/W	STK3L
0F9H	1	1	1	S3PC12	S3PC11	S3PC10	S3PC9	S3PC8	R/W	STK3H
0FAH	S2PC7	S2PC6	S2PC5	S2PC4	S2PC3	S2PC2	S2PC1	S2PC0	R/W	STK2L
0FBH	1	1	1	S2PC12	S2PC11					

2.2.2 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 or DC) occurrence, then these flags will be set to PFLAG register.

ACC is not in data memory (RAM), so ACC can

2.2.3 PROGRAM FLAG

The PFLAG register contains the arithmetic status of ALU operation, system reset status and LVD detecting status. NT0, NPD bits indicate system reset status including power on reset, LVD reset, reset by external pin active and watchdog reset. C, DC, Z bits indicate the result status of ALU operation. LVD24, LVD28 bits indicate LVD detecting power voltage status. - ss W

086H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PFLAG	NT0	NPD	LVD28	LVD24	-	C	DC	Z
Read/Write	R/W	R/W	R	R	-	R/W	R/W	R/W
After reset	-	-	0	0	-	0	0	0

2.2.4 PROGRAM COUNTER

The program counter (PC) is a 13-bit binary counter separated into the high-byte 5 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 12Tc (t) Tj0 Tc (r)448 Tc (ed) Tj0.76 Tc ( ) T448 Tc (pe) Tj0 Tc

F 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 **"ADD M,A"**, **"ADC M,A"** and **"B0ADD 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.***

Ø Example: If PC = 0323H (PCH = 03H, PCL = 23H) ***Ø-0.08448 Tc (e) 0*** =03H,

2.2.5 H, L REGISTERS

The H and L registers are the 8-bit buffers. There are two major functions of these registers.

- can be used as general working registers
- can be used as RAM data pointers with @HL register

0 0 0 rg
0 0 612 792 re
W n
0 0 0 rg
0 0 0 RG
0.5 w
0 J
0 j
[] 0 d
1 i
0 g
0 G

2.2.6 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

et

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	-	-	-	-	-	-	-	-

gister and the

2.3 ADDRESSING MODE

2.3.1 IMMEDIATE ADDRESSING MODE

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

Ø Example: Move the immediate data 12H to ACC.

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

Ø Example: Move the immediate data 12H to R register.

```
B0MOV   R, #12H         ; To set an immediate data 12H into R register.
```

- ***Note: In immediate addressing mode application, the specific RAM must be 0x80~0x87 working register.***

2.3.2 DIRECTLY ADDRESSING MODE

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

Ø Example: Move 0x12 RAM location data into ACC.

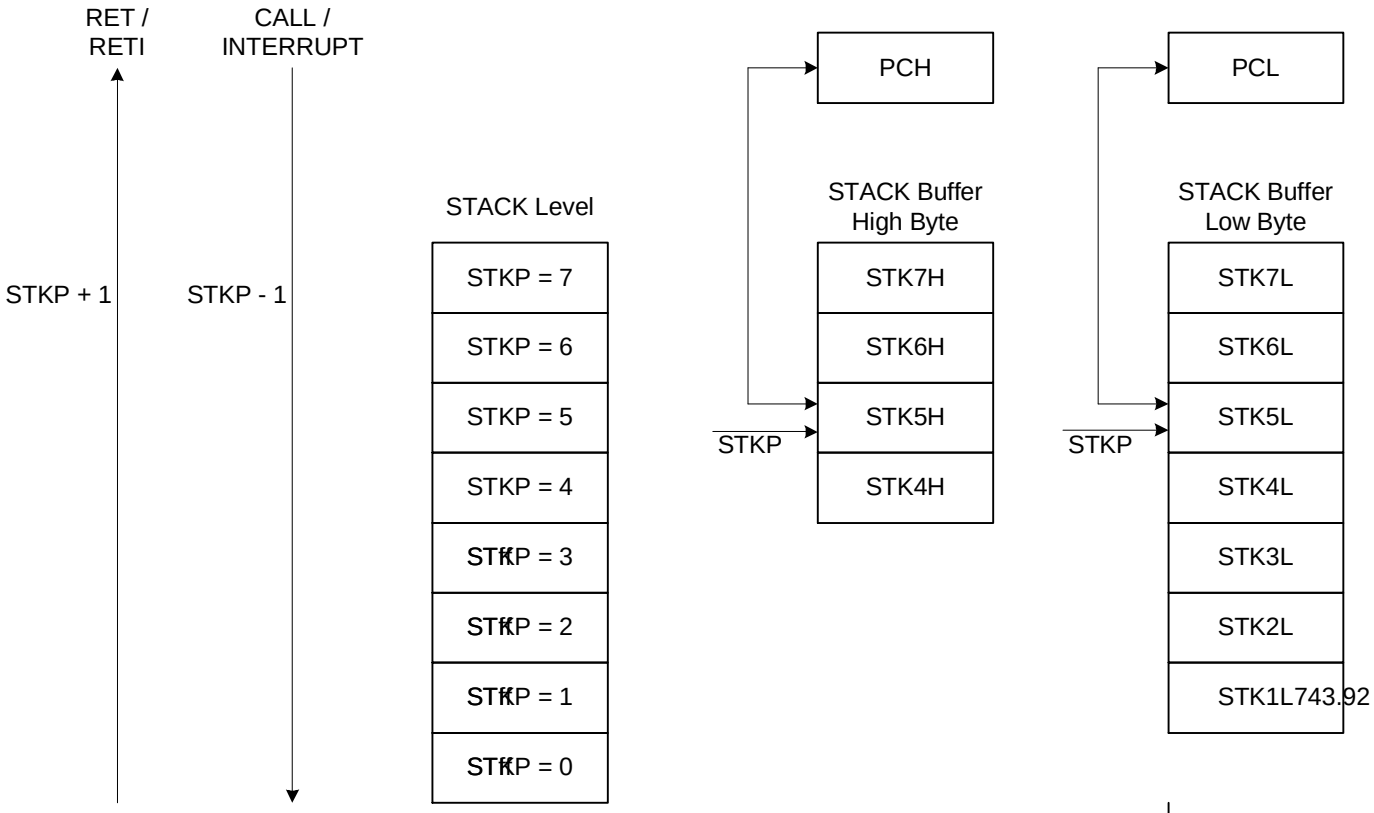
```
B0MOV   A, 12H          ; To get a content of RAM location 0x12 of ba
```

Example: Move 0

2.4 STACK OPERATION

2.4.1 OVERVIEW

The stack buffer has 8-level. These buffers are designed to push and pop up program counter's (PC) data when interrupt service routine and 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.



2.4.2 STACK REGISTERS

The stack pointer (STKP) is a 3-bit register to store the address used to access the stack buffer, 13-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 or during interrupt service routine. 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	GIE	-	-	-	-	STKPB2	STKPB1	STKPB0
Read/Write	R/W	-	-	-	-	R/W	R/W	R/W
After reset	0	-	-	-	-	1	1	1

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

Bit 7 **GIE:** Global interrupt control bit.
0 = Disable.
1 = Enable. Please refer e f

2.4.3

2.5 CODE OPTION TABLE

The code option is the system hardware configurations including oscillator type, watchdog timer operation, LVD option, reset pin option and OTP ROM security control. The code option items are as following table:

Code Option	Content	Function Description
Fcpu	Fhosc/1	Instruction cycle is oscillator clock.
	Fhosc/2	Instruction cycle is 2 oscillator clocks.
	Fhosc/4	Instruction cycle is 4 oscillator clocks.
	Fhosc/8	Instruction cycle is 8 oscillator clocks.
	IHRC_8M	High speed internal 8MHz RC. XIN/XOUT become to P0.3/P0.4 bi-direction I/O pins.
	IHRC_RTC	High speed internal 8MHz RC with 0.5sec RTC. XIN/XOUT become to P0.3/P0.4 bit-direction I/O pins.
	RC	Low cost RC for external high clock oscillator and XOUT becomes to P0.4 bit-direction I/O pin.
	32K X#tal	Low frequency, power saving crystal (e.g. 32.768KHz) for external high clock oscillator.
	8M X#tal	High speed crystal /resonator (e.g. 8MHz) for external high clock

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 007776 t0.084r

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.

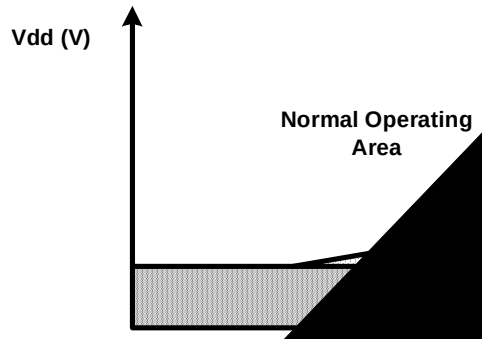
- **Power-up:** System detects the power voltage up and waits for power stable.
- **External reset (only external reset pin enable):** System checks external reset pin status. If external reset pin is not high level, the system keeps reset status and waits external reset pin released.
- **System initialization:** 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.

3.4 BROWN OUT RESET

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.

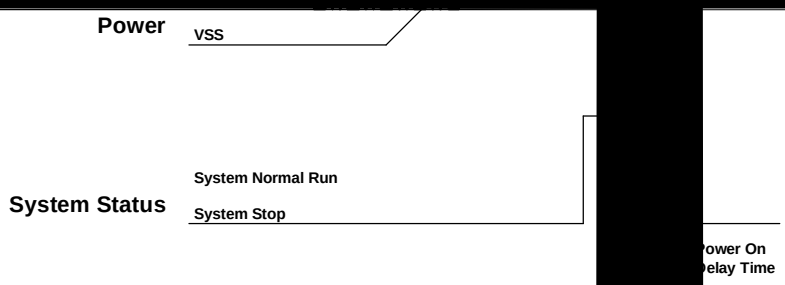
3.4.1 THE SYSTEM OPERATING VOLTAGE

To improve the brown out reset needs to know the system minimum o stem executing rate and power level. Different system executing rates hav ltage. The electrical characteristic section shows the system voltage to execu



Normally the system operation vo the reset voltage is decided by LVD detect level. Th ting rate upper even higher than system reset volta ge above the system reset voltage.

3.4.2 LOW V



086H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PFLAG	NT0	NPD	LVD28	LVD24	-	C	DC	Z
Read/Write	R/W	R/W	R	R	-	R/W	R/W	R/W
After reset	-	-						

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.

l LVD reset Tc (e) Tj 0 Tc (r) Tj - 0 . 0 8 7 1 6 . 56 0 TD (L) Tj 0 Tc (V

3.5 EXTERNAL RESET

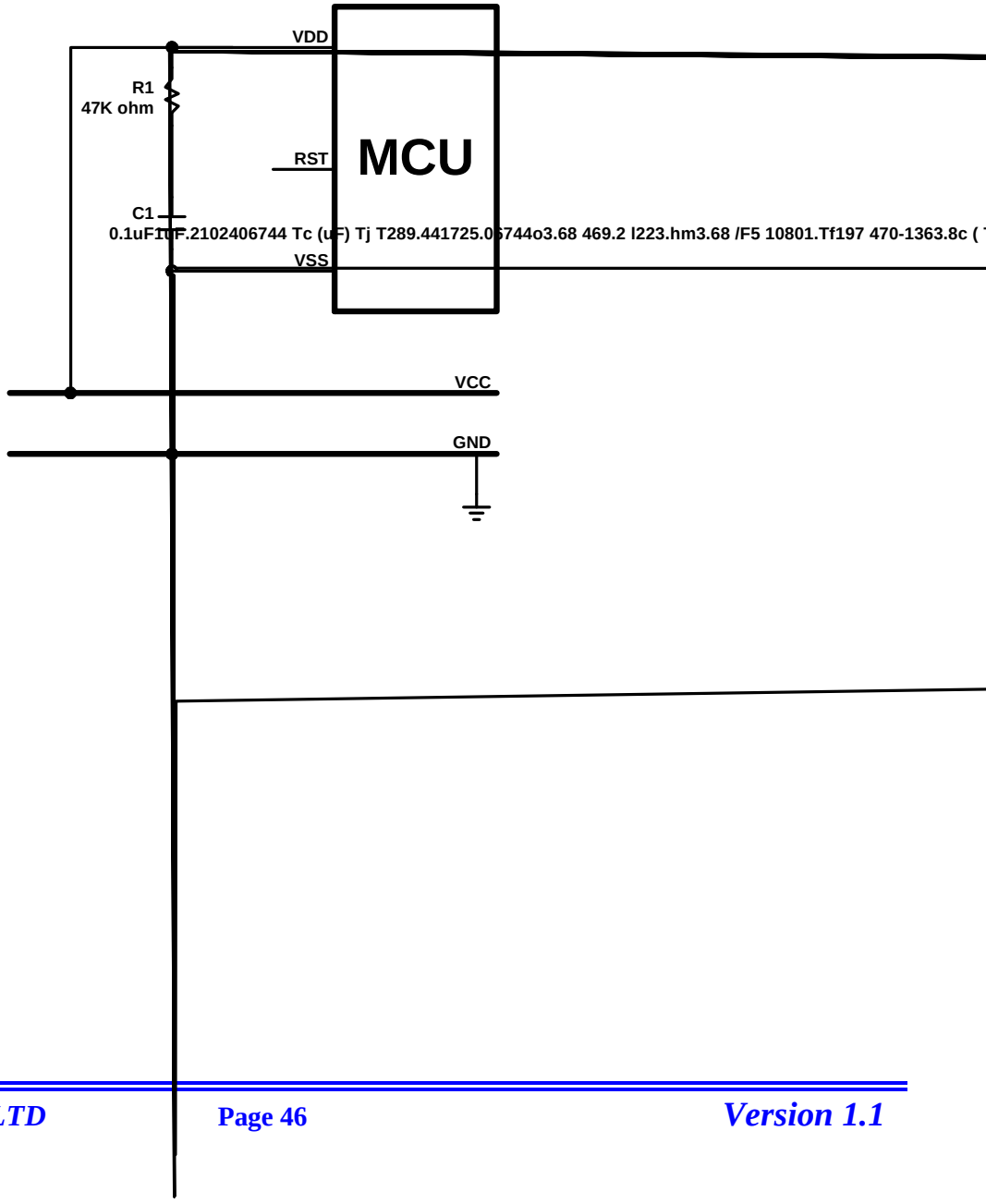
External reset function is controlled by Reset_Pin! code option. Set the code option as Reset! option to enable external reset function. External reset pin is Schmitt Trigger structure and low level active. The system is running when reset pin is high level voltage input. The reset pin receives the low 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.

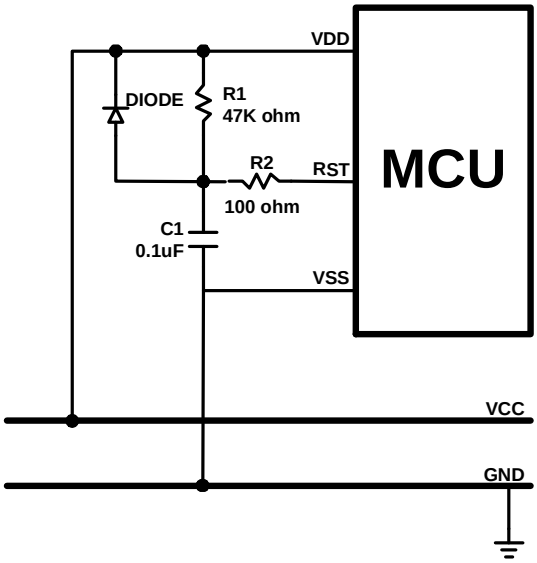
The external reset can reset the system during power on duration, and good external reset circuit can protect the system to avoid working at unusual power condition, e.g. brown out reset in AC power application "

3.6 EXTERNAL RESET CIRCUIT

3.6.1 Simply RC Reset Circuit

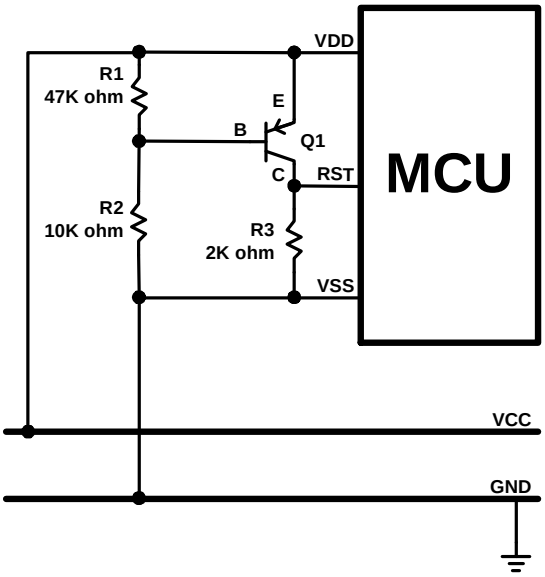


3.6.2 Diode & RC Reset Circuit



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

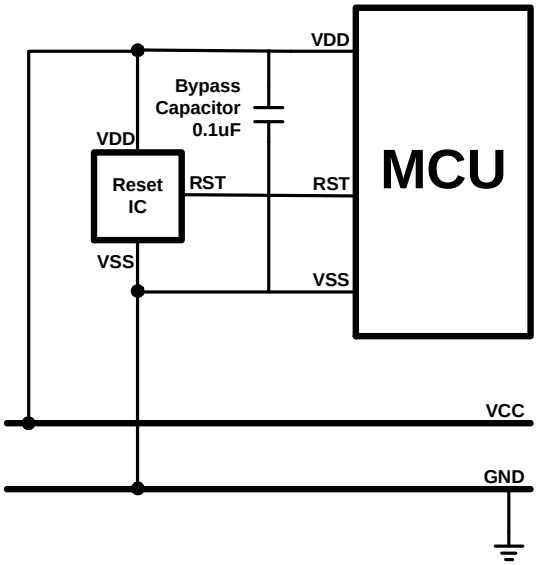
3.6.4 Voltage Bias Reset Circuit



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.

To decide the reset detect voltage by R1, R2 resistances. Select the right R1, R2 value to conform the applicatioTj-0.32448 Tc (c

3.6.5 External Reset IC



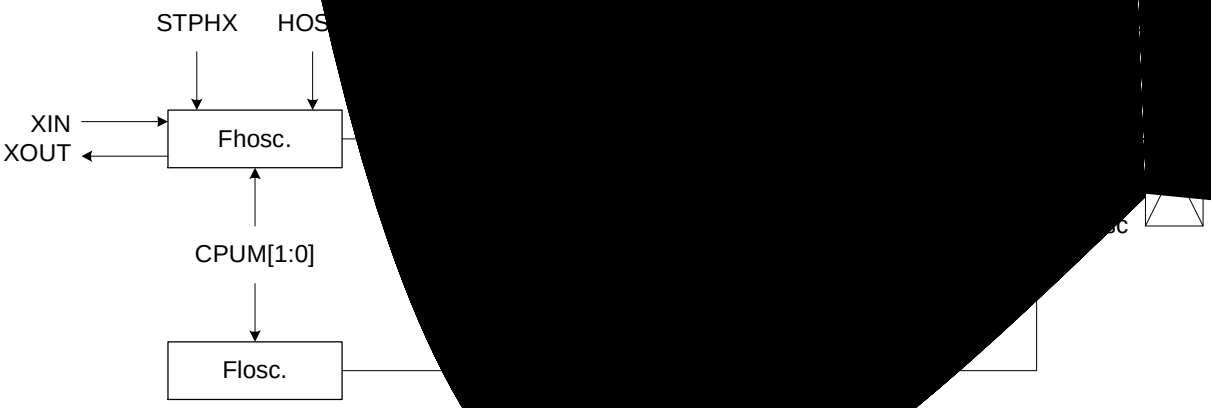
4 SYS

4.1 OVERVI

The micro-controller i
generated from the e
low-speed clock is ger
Both the high-speed c
divided by 4 to be the i

F Normal Mode (Hig
F Slow Mode (Low

4.2 CLOCK BL



4.4 OSCM REGISTER

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

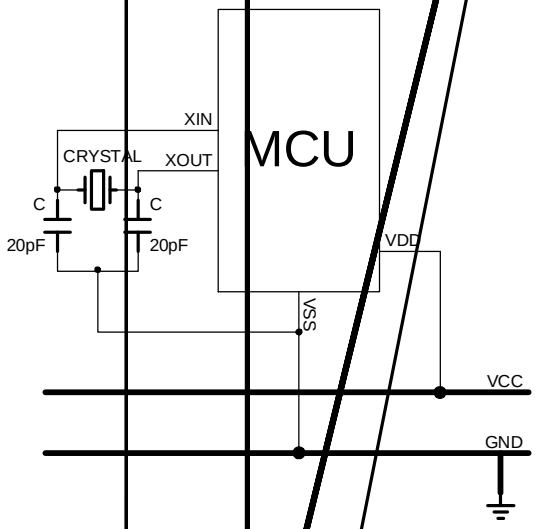
0CAH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
OSCM	0	0	0	CPUM1	CPUM0	CLKMD	STPHX	0
Read/Write	-	-	-	R/W	R/W	R/W	R/W	-
After reset	-	-	-	0	0	0	0	-

Bit 1 **STPH**

4.6 I0.108 Tc4 TD 0.072 Tc (4) 1

4.7.1 CRYSTAL/CERAMIC

Crystal/Ceramic devices are driven by XIN, XOUT pins. For high/normal/low frequency, the driving currents are different. High_Clk code option supports different frequencies. 8M option is for high speed (ex. 8MHz). 4M option is for normal speed (ex. 4MHz). 32K option is for low speed (ex. 32768Hz).



Note: Connect the R and C as near as possible to the VDD pin of micro-controller.

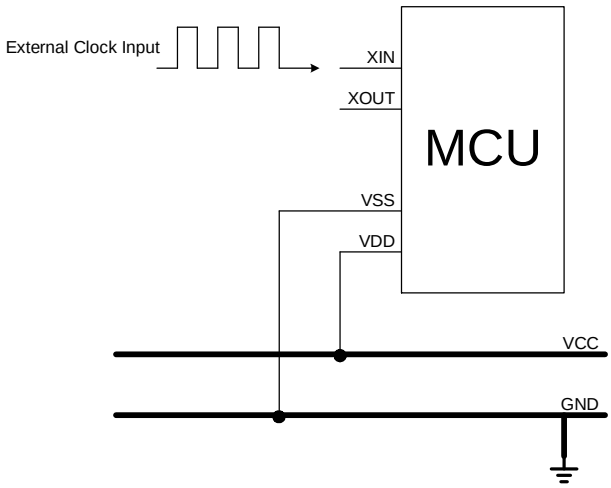
- ***Note: Connect the Crystal/Ceramic and C as near as possible to the XIN/XOUT/VSS pins of micro-controller.***

4.7.2 RC

Selecting RC oscillator is by RC option of High_Clk code option. RC type oscillator's frequency is up to 10MHz. Using R! value is to i

4.7.3 EXTERNAL CLOCK SIGNAL

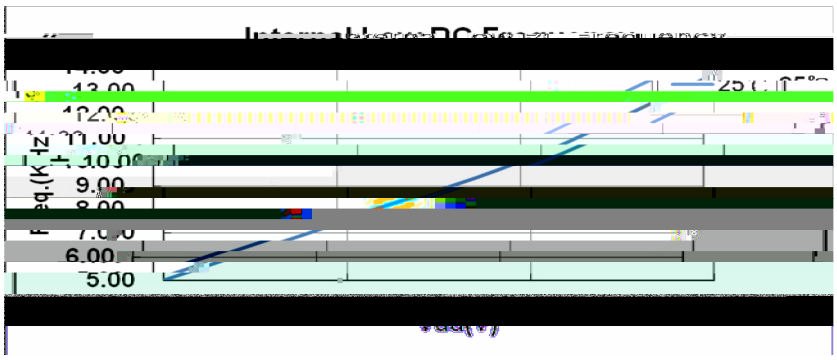
Selecting external clock signal input to be system clock is by RC option of High_Clk code option. The external clock signal is input from XIN pin. XOUT pin is general purpose I/O pin.



- ***Note: The GND of external oscillator circul***

4.8 SYSTEM LOW CLOCK

The system low clock source is the internal low-speed oscillator built in the micro-controller. The low-speed oscillator uses RC type oscillator circuit. The frequency is affected by the voltage and temperature of the system. In common condition, the frequency of the RC oscillator is about 10KHz at 3V. The relation between the RC frequency and voltage is as the following figure.



The internal low RC supports watchdog clock source and system slow mode controlled by CLKMD.

- **F** ***Flosc = Internal low RC oscillator (about 10KHz @3V).***
- **F** ***Slow mode Fcpu = Flosc / 4***

There are two conditions to stop internal low RC. One is power down mode, and the other is green mode of 455K mode and watchdog disable. If system is in 455K mode and watchdog disable, only 455K oscillator actives and system is under low power consumption.

Ø Example: Stop internal low-speed oscillator by power down mode.

```
B0BSET      FCPUM0          ; To stop external high-speed oscillator and internal low-speed
                            ; oscillator called power down mode (sleep mode).
```

- ***Note: The internal low-speed clock can t be turned off individually. It is controlled by CPUM0, CPUM1 bits of OSCM register.***

4.8.1 SYSTEM CLOCK MEASUREMENT

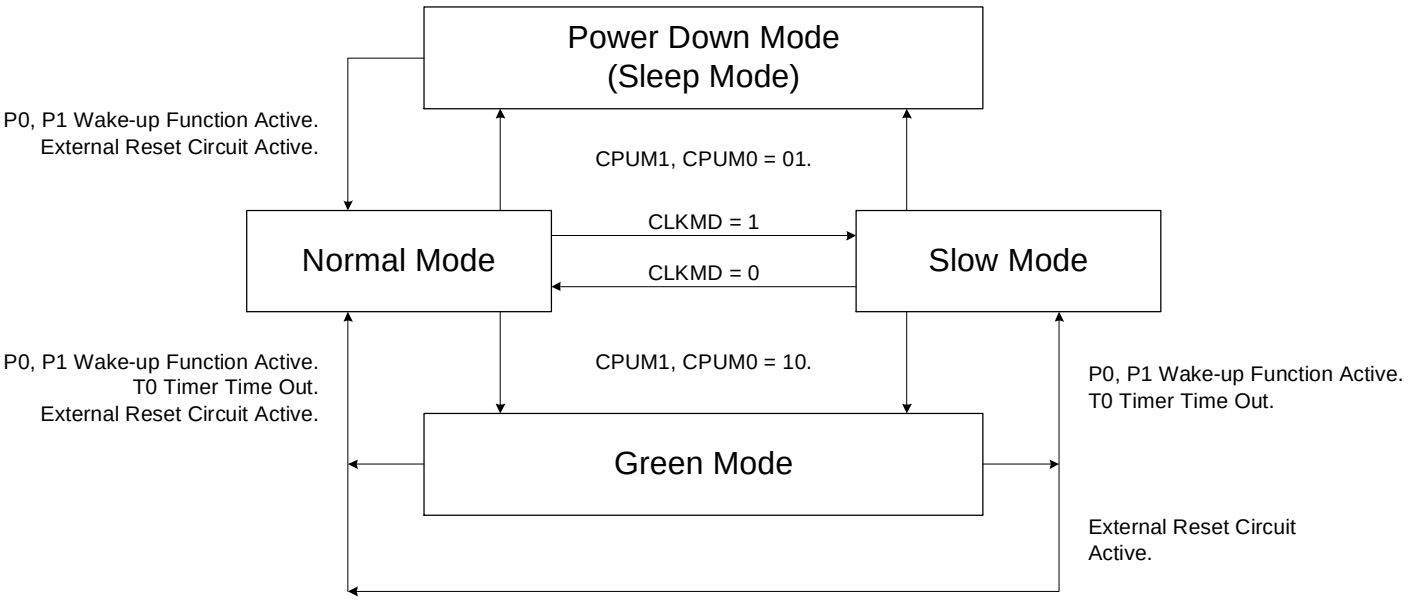
Under design period, the users can measure system clock speed by software instruction cycle (Fcpu). T46.08 re f31.92 34

5 SYSTEM OPERATION MODE

5.1 OVERVIEW

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

- High-speed mode
- Low-speed mode
- Power-down mode (Sleep mode)
- Green mode



System Mode Switching Diagram

Operating mode description

MODE	NORMAL	SLOW	GREEN	POWER DOWN (SLEEP)	REMARK
		HX	By STPHX	Stop	
		HX	By STPHX	Stop	
		g	Running	Stop	

5.5 GREEN MODE

The green mode is another system ideal status not like power down mode. In power down mode, all functions and hardware devices are disabled. But in green mode, the system clock source keeps running, so the power consumption of green mode is larger than power down mode. In green mode, the program isn't executed, but the timer with wake-up function actives as enabled, and the timer clock source is the non-stop system clock. The green mode has 2 wake-up sources. One is the P0, P1 level change trigger wake-up. The other one is internal timer with wake-up function occurring overflow. That's mean users can setup one fix period to timer, and the system is waked up until the time out. Inserting green mode is controlled by CPUM1 bit of OSCM register. When CPUM1=1, the system inserts into green mode. After system wake-up from green mode, the CPUM1 bit is disabled (zero status) automatically.

- The program stops executing, and full functions are disabled.
- Only the timer with wake-up function actives.
- The oscillator to be the system clock source keeps running, and the other oscillators operation is depend on system operation mode configuration.
- If inserting green mode from normal mode, the system insets to normal mode after wake-up.
- If inserting green mode from slow mode, the system insets to slow mode after wake-up.
- The green mode wake-up source are P0, P1 level change trigger and unique time overflow.

t

- ***Note: Sonix provides "GreenMode! macro to control green mode operation. It is necessary to use "@GreenMode! macro to control system inserting green mode.***
  The macro includes three instructions. Please take care the macro length as using BRANCH type instructions, e.g. bts0, bts1, b0bts0, b0bts1, ins, incms, decs, decms, cmprs, jmp, or the routine would be error.

5.6 OPERATING MODE CONTROL MACRO

Sonix provides operating mode control macros to switch system operating mode easily.

Ø Example: Switch normal/slow mode to green mode and enable T0 wake-up function.

```
; Set T0 timer wakeup function.
        B0BCLR      FT0IEN      ; To disable T0 interrupt service
        B0BCLR      FT0ENB      ; To disable T0 timer
        MOV         A,#20H      ;
        B0MOV       T0M,A       ; To set T0 clock = Fcpu / 64
        MOV         A,#74H
        B0MOV       T0C,A       ; To set T0C initial value = 74H (To set T0 interval = 10 ms)
        B0BCLR      FT0IEN      ; To disable T0 interrupt service
        B0BCLR      FT0IRQ      ; To clear T0 interrupt request
        B0BSET      FT0ENB      ; To enable T0 timer

; Go into green mode
        GreenMode               ; Declare  GreenMode! macro directly.
```

Ø Example: Switch normal/slow mode to

5.7 WAKEUP

5.7.1 OVERVIEW

Under power down mode (sleep mode) or green mode, program doesn#t execute. The wakeup trigger can wake the system up to normal mode or slow mode. The wakeup trigger sources are external trigger (P0, P1 level change) and internal trigger (T0 timer overflow).

- Power down mode is waked up to normal mode. The wakeup trigger is only external trigger (P0, P1 level change)
- Green mode is waked up to last mode (normal mode or slow mode). The wakeup triggers are external trigger (P0, P1 level change) and internal trigger (T0 timer overflow).

5.7.2 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 2048 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.

- ***Note: Wakeup from green mode is no wakeup time because the clock doesn t stop in green mode.***

The value of the wakeup time is as the following.

The Wakeup time = 1/Fosc * 2048 (sec) + high clock start-up time

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

Ø **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 * 2048 = 0.512 ms (Fosc = 4MHz)
The total wakeupThe wakeTha

5.7.3 P1W WAKEUP CONTROL REGISTER

Under power down mode (sleep mode) and green mode, the I/O ports with wakeup function are able to wake the system up to normal mode. The Port 0 and Port 1 have wakeup function. Port 0 wakeup function always enables, but the Port 1 is controlled by the P1W register.

0C0H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P1W	P17W	P16W	P15W	P14W	P13W	P12W	P11W	P10W
Read/Write	W	W	W	W	W	W	W	W
After reset	0	0	0	0	0	0	0	0

Bit[7:0] **P10W~P17W:** Port 1 wakeup function control bits.
0 = Disable P1n wakeup function.
1 = Enable P1n wakeup function.

6.2 INTEN INTERRUPT ENABLE REGISTER

INTEN is the interrupt request control register including three internal interrupts, two external interrupts enable control bits. One of the register to be set 1! is to enable the interrupt request function. Once of the interrupt occur, the stack is incremented and program jump to ORG 8 to execute interrupt service routines. The program exits the interrupt service routine when the returning interrupt service routine instruction (RETI) is executed.

i 0

C9H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0 0

6.3 INTRQ INTERRUPT REQUEST REGISTER

INTRQ is the interrupt request flag register. The register includes all interrupt request indication flags. Each one of the interrupt requests occurs, the bit of the INTRQ register would be set 1!. The INTRQ value needs to be clear by programming after detecting the flag. In the interrupt vector of program, users know the any interrupt requests occurring by the register and do the routine corresponding of the interrupt request.

0C8H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0 t

6.6 EXTERNAL INTERRUPT OPERATION (INT0)

Sonix provides 1 external interrupt sources in the micro-controller. INT0 is external interrupt trigger source and builds in edge trigger configuration function. When the external edge trigger occurs, the external interrupt request flag will be set to 1! no matter the external interrupt control bit enabled or disable. When external interrupt control bit is enabled and external interrupt edge trigger is occurring, the program counter will jump to the interrupt vector (ORG 8) and execute interrupt service routine.

The external interrupt builds in wake-up latch function. That means when the system is triggered wake-up from power down mode, the wake-up source is external interrupt source (P0.0), and the trigger edge direction matches interrupt edge configuration, the trigger edge will be latched, and the system executes interrupt service routine fist after wake-up.

0BFH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PEDGE	-	-	-	P00G1	P00G0	-	-	-
Read/Write	-	-	-	R/W	R/W	-	-	-
After reset	-	-	-	0	0	-	-	-

Bit[4:3] **P00G[1:0]:** INT0 edge trigger select bits.
00 = reserved,
01 = rising edge,
10 = falling edge,
11 = rising/falling bi-direction.

Ø Example: Setup INT0 interrupt request and bi-direction edge trigger.

```
MOV         A, #98H
B0MOV       PEDGE, A        ; Setet
                            bqutbt          PF Tj0 Tc (MP Tj-0.08448 Tc (0)) Tj0.07776 Tc ( I Tj0
  JMP Tj0223676 Tc ( )
```

6.7 INT1 (P0.1) INTERRUPT OPERATION

When the INT1 trigger occurs, the P01IRQ will be set to 1! no matter the P01IEN is enable or disable. If the P01IEN = 1 and the trigger event P01IRQ is also set to be 1!. As the result, the system will execute the interrupt vector (ORG 8). If the P01IEN = 0 and the trigger event P01IRQ is still set to be 1!. Moreover, the system won#t execute interrupt vector even when the P01IRQ is set to be 1!. Users need to be cautious with the operation under multi-interrupt situation.

- ***Note: The interrupt trigger direction of P0.1 is falling edge.***

Ø Example: INT1 interrupt request setup.

```
        B0BSET      FP01IEN     ; Enable INT1 interrupt service
        B0BCLR      FP01IRQ     ; Clear INT1 interrupt request flag
        B0BSET      FGIE        ; Enable GIE
```

Ø Example: INT1 interrupt service routine.

```
        ORG         8           ; Interrupt vector
        JMP         INT_SERVICE
INT_SERVICE:
        "                       ; Push routine to save ACC and PFLAG to buffers.
        B0BTS1      FP01IRQ     ; Check P01IRQ
        JMP         EXIT_INT    ; P01IRQ = 0, exit interrupt vector
        B0BCLR      FP01IRQ     ; Reset P01IRQ
        …                       ; INT1 interrupt service
```

6.8 T0 INTERRUPT OPERATION

When the T0C counter occurs overflow, the T0IRQ will be set to 1! however the T0IEN is enable or disable. If the T0IEN = 1, the trigger event will make the T0IRQ to be 1! and the system enter interrupt vector. If the T0IEN = 0, the trigger event will make the T0IRQ to be 1! but the system will not enter interrupt vector. Users need to care for the operation under multi-interrupt situation.

Ø Example: T0 interrupt request setup.

```
        B0BCLR      FT0IEN      ; Disable T0 interrupt service
        B0BCLR      FT0ENB      ; Disable T0 timer
        MOV         A, #20H     ;
        B0MOV       T0M, A      ; Set T0 clock = Fcpu / 64
        MOV         A, #74H     ; Set T0C initial value = 74H
        B0MOV       T0C, A      ; Set T0 interval = 10 ms

        B0BSET      FT0IEN      ; Enable T0 interrupt service
        B0BCLR      FT0IRQ      ; Clear T0 interrupt request flag
        B0BSET      FT0ENB      ; Enable T0 timer

        B0BSET      FGIE        ; Enable GIE
```

Ø Example: T0 interrupt service routine as no RTC function.

```
        ORG         8           ; Interrupt vector
        JMP         INT_SERVICE
INT_SERVICE:

        "                       ; Push routine to save ACC and PFLAG to buffers.

        B0BTS1      FT0IRQ      ; Check T0IRQ
        JMP         EXIT_INT    ; T0IRQ = 0, exit interrupt vector

        B0BCLR      FT0IRQ      ; Reset T0IRQ
        MOV         A, #74H
        B0MOV       T0C, A      ; Reset T0C.
        …                       ; T0 interrupt service routine
        "
EXIT_INT:
        "                       ; Pop routine to load ACC and PFLAG from buffers.


        RETI                    ; Exit interrupt vector
```

6.9 T1 INTERRUPT OPERATION

When the T1C (T1CH, T1CL) counter occurs overflow, the T1IRQ will be set to 1! however the T1IEN is enable or disable. If the T1IEN = 1, the trigger event will make the T1IRQ to be 1! and the system enter interrupt vector. If the T1IEN = 0, the trigger event will make the T1IRQ to be 1! but the system will not enter interrupt vector. Users need to care for the operation under multi-interrupt situation.

Ø Example: T1 interrupt request setu

6.10 TC1 INTERRUPT OPERATION

When the TC1C counter overflows, the TC1IRQ will be set to 1! no matter the TC1IEN is enable or disable. If the TC1IEN and the trigger event TC1IRQ is set to be 1!. As the result, the system will execute the interrupt vector. If the TC1IEN = 0, the trigger event TC1IRQ is still set to be 1!. Moreover, the system won#t execute interrupt vector even when the TC1IRQ is set to be 1!. Users need to be cautious with the operation under multi-interrupt situation.

Ø Example: TC1 interrupt request setup.

```
        B0BCLR      FTC1IEN         ; Disable TC1 interrupt service
        B0BCLR      FTC1ENB         ; Disable TC1 timer
        MOV         A, #20H         ;
        B0MOV       TC1M, A         ; Set TC1 clock = Fcpu / 64
        MOV         A, #74H         ; Set TC1C initial value = 74H
        B0MOV       TC1C, A         ; Set TC1 interval = 10 ms

        B0BSET      FTC1IEN         ; Enable TC1 interrupt service
        B0BCLR      FTC1IRQ         ; Clear TC1 interrupt request flag
        B0BSET      FTC1ENB         ; Enable TC1 timer

        B0BSET      FGIE            ; Enable GIE
```

Ø Example: TC1 interrupt service routine.

```
        ORG         8               ; Interrupt vector
        JMP         INT_SERVICE
INT_SERVICE:

        "                           ; Push routi
```

6.11 COMPARATOR INTERRUPT OPERATION (CMP0, CMP1)

Sonix provides 2 sets comparator with interrupt function in the micro-controller. The comparator interrupt trigger edge direction is the rising edge of comparator output . When the comparator output status transition occurs, the comparator interrupt request flag will be set to 1! no matter the comparator interrupt control bit status. The comparator interrupt flag doesn#t active only when comparator control bit is disabled. When comparator interrupt control bit is enabled and comparator interrupt edge trigger is occurring, the program counter will jump to the interrupt vector (ORG 8) and execute interrupt service routine.

09CH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMP0M	CM0EN	CM0IEN	CM0IRQ	CM0OEN	CM0REF	CM0OUT	CMS1	CMS0
Read/Write	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W
After Reset	0	0	0	0	0	0	0	0

Bit 6 **CM0I**

6.12 SIO INTERRUPT OPERATION OPE

6.13 UART INTERRUPT OPERATION

When the UART transmitter successfully, the RXIRQ/TXIRQ will be set to 1! no matter the RXIEN/TXIEN is enable or disable. If the RXIEN/TXIEN and the trigger event RXIRQ/TXIRQ is set to be 1!. As the result, the system will execute the interrupt vector. If the RXIEN/TXIEN = 0, the trigger event RXIRQ/TXIRQ is still set to be 1!. Moreover, the system won#t execute interrupt vector even when the RXIEN/TXIEN is set to be 1!. Users need to be cautious with the operation under multi-interrupt situation.

Ø Example: UART receive and transmit interrupt request setup.

```
B0BSET      FRXIEN      ; Enable UART receive interrupt service
B0BCLR      FRXIRQ      ; Clear UART receiv.08448 Tc (ei) Tj0.24 Tecee
```

6.14 MULTI-INTERRUPT OPERATION

Under certain condition, the software designer uses more than one interrupt requests. Processing multi-interrupt request requires setting the priority of the interrupt requests. The IRQ flags of interrupts are controlled by the interrupt event. Nevertheless, the IRQ flag 1! doesn#t mean the system will execute the interrupt vector. In addition, which means the IRQ flags can be set 1! by the events without enable the interrupt. Once the event occurs, the IRQ will be logic 1!. The IRQ and its trigger event relationship is as the below table.

Interrupt Name	*Trigger Event Description*
P00IRQ	P0.0 trigger controlled by PEDGE
P01IRQ	P0.1 falling edge trigger.
T0IRQ	T0C overflow
T1IRQ	T1C overflow
TC1IRQ	TC1C overflow
SIOIRQ	SIO transmitter successfully.
CM0IRQ	Comparator 0 output level transition.
CM1IRQ	Comparator 1 output level transition.
RXIRQ/TXIRQ	UART transmitter successfully.

For multi-interrupt conditions, two things need to be taking care of. One is to set the priority for 0.08448 Tc (he) Tj 0.0770 Tc (n
he83 0.08m. Cd (-0.0770 Tc (Pdt770 Tc (f) Tj 0.08448 Tc he01.04 1.44 ck Te (t)) Tj 0.077 (f) Tj -02 TD 0.08c (he)) f

7 I/O PORT

7.1 OVERVIEW

The micro-controller builds in 44 pin I/O. Most of the I/O pins are mixed with analog pins and special function pins. The I/O shared pin list is as following.

I/O Pin		Shared Pin		Shared Pin Control Condition
Name	Type	Name	Type	
P0.0	I/O	INT0	DC	P00IEN=1
P0.1	IO	INT1	DC	P01IEN=1

7.2 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

7.3 I/O PULL UP REGISTER

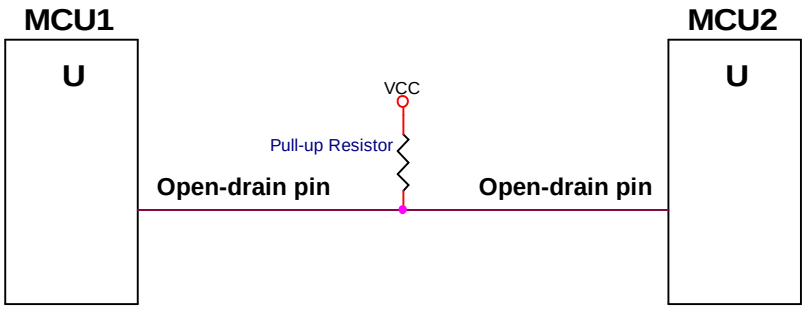
0E0H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P0UR	P07R	P06R	P05R	P04R	P03R	-	P01R	P00R
Read/Write	W	W	W	W	W	-	W	W
After reset	0	0	0	0	0	-	0	0

0E1H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P1UR	P17R	P16R	P15R	P14R	P13R	P12R	P11R	P10R
Read/Write	W	W	W	W	W	W	W	W
After reset	0	0	0	0	0	0	0	0

0E2H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P2UR	P27R	P26R	P25R	P24R	P23R	P22R	P21R	P20R

7.4 I/O OPEN-DRAIN REGISTER

P1.0/P1.1/P3.2/P3.3/P5.0/P5.1/P5.2 built in open-drain function. P1.0/P1.1/P3.2/P3.3/P5.0/P5.1/P5.2 must be set as output mode when enable open-drain function. Open-drain external circuit is as following.



The pull-up resistor is necessary. Open-drain output high is driven by pull-up resistor. Output low is sunken by MCU's pin.

0E9H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P1OC	P52OC	P51OC	P50OC	P33OC	P32OC	-	P11OC	P10OC
Read/Write	W	W	W	W	W	-	W	W
After reset	0	0	0	0	0	-	0	0 rrrrrrrrrr

Ø Example: Enable P1.0 to open-drain mode and output high.

```
B0BSET      P1.0        ; Set P1.0 buffer high.

B0BSET      P10M        ; Enable P1.0 output mode.
MOV         A, #01H     ; Enable P1.0 open-drain function.
B0MOV       P1OC, A
```

- ***Note: P1OC is write only register. Setting P10OC must be used "MOV! instructions.***

Ø Example: Disable P1.0 to open-drain mode and output low.

```
MOV         A, #0       ; Disable P1.0 open-drain function.
B0MOV       P1OC, A
```

- ***Note: After disable P1 open-drain function, P1 mode returns to last I/O mode.***

7.5 I/O PORT DATA REGISTER

8 TIMERS

8.1 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 (10KHz @3V).

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

VDD	*Internal Low RC Freq.*	*Watchdog Overflow Time*
3V	10KHz	819.2ms

The watchdog timer has three operating options controlled WatchDog! code option.

- **Disable:** Disable watchdog timer function.
- **Enable:** Enable watchdog timer function. Watchdog timer actives in normal mode and slow mode. In power down mode and green mode, the watchdog timer stops.
- **Always_On:** Enable watchdog timer function. The watchdog timer actives and not stop in power down mode and green mode.

In high noisy environment, the "Always_On" option of watchdog operations is the strongly recommendation to make the system reset under error situations and re-start again.

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

[redacted]				Bit 3	Bit 2	Bit 1	Bit 0
[redacted]							
[redacted]	W						
					R		

Ø 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.
- 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.

Ø 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.

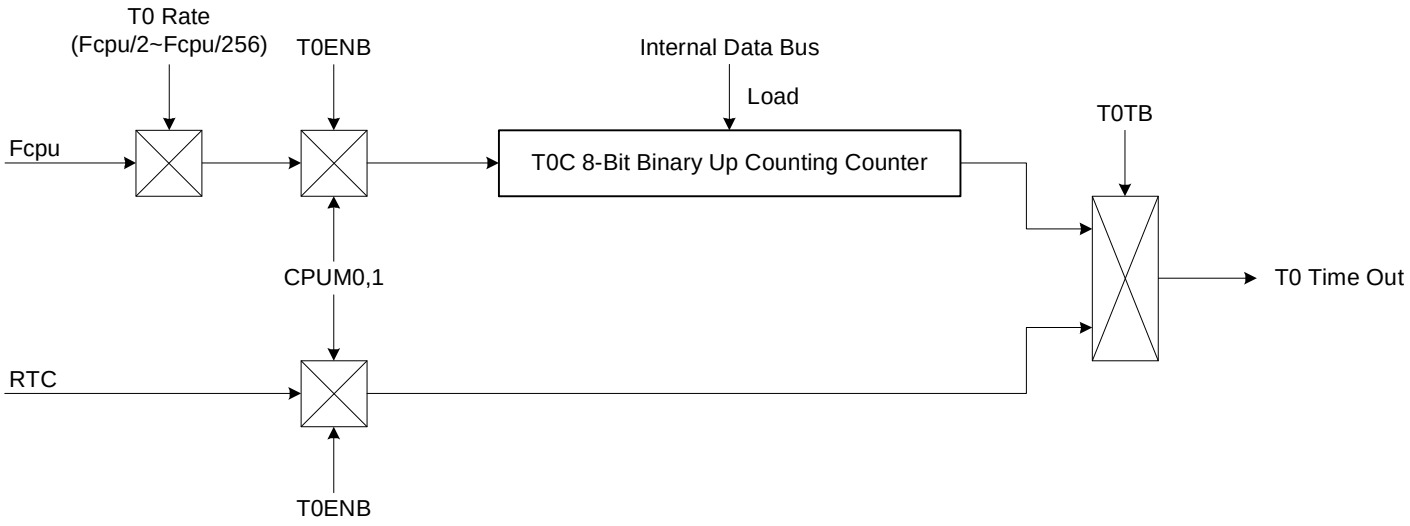
```
Main:
         "                          ; Check I/O.
         "                          ; Check RAM
Err:    JMP $                       ; I/O or RAM error. Program jump here and don't
                                    ; clear watchdog. Wait watchdog timer overflow to reset IC.

Correct:                            ; I/O and RAM are correct. Clear watchdog timer and
                                    ; execute program.
        MOV         A, #5AH         ; Clear the watchdog timer.
        B0MOV       WDTR, A
         "
        CALL        SUB1
        CALL        SUB2
         "
         "
         "
        JMP         MAIN
```

8.2 TIMER 0 (T0)

8.2.1 OVERVIEW

The T0 is an 8-bit binary up timer and event counter. If T0 timer occurs an overflow (from FFH to 00H), it will continue counting and issue a time-out signal to trigger T0 interrupt to request interrupt service.
The main purposes of the T0 timer is as following.

- **F** **8-bit programmable up counting timer:** Generates interrupts at specific time intervals based on the selected clock frequency.
- **F** **RTC timer:** Generates interrupts at real time intervals based on the selected clock source. **RTC function is only available in High_Clk code option = "IHRC_RTC".**
- **F** **Green mode wakeup function:** T0 can be green mode wake-up time as T0ENB = 1. System will be wake-up by T0 time out.



- ***Note:1. In RTC mode, clear T0IRQ must be after 1/2 RTC clock source (32768Hz), or the RTC interval time is error. The delay is about 16us and use T0 interrupt service routine executing time to be the***

8.2.2 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	-	-	-	T0TB
Read/Write	R/W	R/W	R/W	R/W	-	-	-	R/W
After reset	0	0	0	0	-	-	-	0

Bit 0 **T0TB:** RTC clock source control bit.
0 = Disable RTC (T0 clock source from Fcpu).
1 = Enable RTC.

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.

- ***Note: T0RATE is not available in RTC mode. The T0 interval time is fixed at 0.5 sec.***

8.2.4 T0 TIMER OPERATION SEQUENCE

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

Ø Stop T0 timer counting, disable T0 interrupt function and clear T0 interrupt request flag.

```
B0BCLR      FT0ENB          ; T0 timer.
B0BCLR      FT0IEN          ; T0 interrupt function is disabled.
B0BCLR      FT0IRQ          ; T0 interrupt request flag is cleared.
```

Ø Set T0 timer rate.

```
MOV         A, #0xxx0000b   ;The T0 rate control bits exist in bit4~bit6 of T0M. The
                            ; value is from x000xxxxb~x111xxxxb.
B0MOV       T0M,A           ; T0 timer is disabled.
```

Ø Set T0 clock source from Fcpu or RTC.

```
B0BCLR      FT0TB           ; Select T0 Fcpu clock source.
```

or

```
B0BSET      FT0TB           ; Select T0 RTC clock source.
```

Ø Set T0 interrupt interval time.

```
MOV         A,#7FH
B0MOV       T0C,A           ; Set T0C value.
```

Ø Set T0 timer function mode.

```
B0BSET      FT0IEN          ; Enable T0 interrupt function.
```

Ø Enable T0 timer.

```
B0BSET      FT0ENB          ; Enable T0 timer.
```

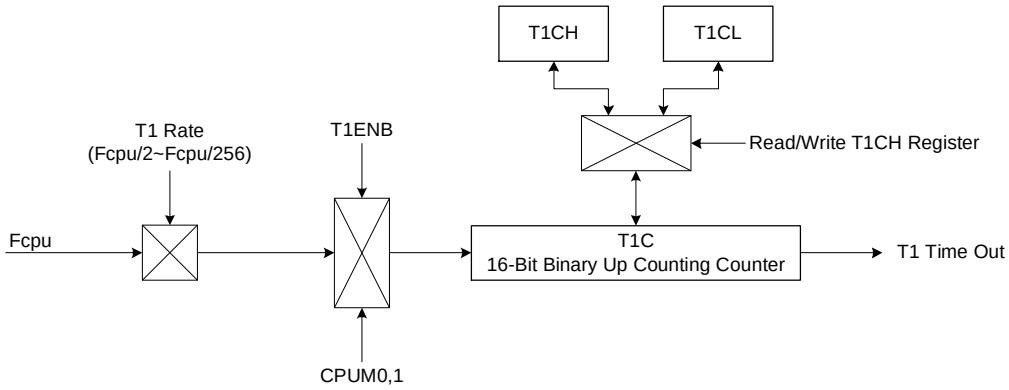
8.3 TIMER 1 (T1)

8.3.1 OVERVIEW

The T1 is an 16-bit binary up timer. If T1 timer occurs an overflow (from FFFFH to 0000H), it will continue counting and issue a time-out signal to trigger T1 interrupt to request interrupt service.

The main purposes of the T1 timer is as following.

F **16-bit programmable up counting timer:** Generates interrupts at specific time intervals based on the selected clock frequency.



8.3.2 T1M MODE REGISTER

0A0H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T1M	T1ENB	T1RATE2	T1RATE1	T1RATE0	-	-	-	-
Read/Write	R/W	R/W	R/W	R/W	-	-	-	-
After resetD -(-) TjET32.4 31060.24 310.8 0.48 0.4rR/W e					8			

8.3.3 T1CH, T1CL COUNTING REGISTER

T1C is an 16-bit counter register for T1 interval time control. T1CH is high byte of T1C. T1CL is low byte of T1C.

505.2ETQ q BT 4 5 9 4 2 9 8 T 3 2 2 T 0 2 2 1 f 3 2 . 4 8 6 4 . 4 8 f e 5 6 . 5 6 4 8 . 4 8 C n W 0 . 4 8 f 3 2 . 8 8 6 4

0A1H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T1CL	T1CL7	T1CL6	T1CL5	T1CL4	T1CL3	T1CL2	T1CL1	T1CL0
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

0A2H	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T1CH	T1CH7	T1CH6	T1CH5	T1CH4	T1CH3	T1CH2	T1CH1	T1CH0
Read/WrT								W

8.3.4 T1 TIMER OPERATION SEQUENCE

T1 timer operation sequence of setup T1 timer is as following.

F Stop T1 timer counting, disable T1 interrupt function and clear T1 interrupt request flag.

```
B0BCLR      FT1ENB          ; T1 timer.
B0BCLR      FT1IEN          ; T1 interrupt function is disabled.
B0BCLR      FT1IRQ          ; T1 interrupt request flag is cleared.
```

F Set T1 timer rate.

```
MOV         A, #0xxx0000b   ;The T1 rate control bits exist in bit4~bit6 of T1M. The
                            ; value is from x000xxxxb~x111xxxxb.
B0MOV       T1M,A           ; T1 timer is disabled.
```

F Set T1 interrupt interval time.

```
MOV         A,#7FH
B0MOV       T1CL,A          ; Set T1CL value.
MOV         A,#7FH
B0MOV       T1CH,A          ; Set T1CH value.
```

F Set T1 timer function mode.

```
B0BSET      FT1IEN          ; Enable T1 interrupt function.
```

F Enable T1 timer.

```
B0BSET      FT1ENB          ; Enable T1 timer.
```

8.4 TIMER/COUNTER 0 (TC1)

8.4.1 OVERVIEW

The TC1 is an 8-bit binary up counting timer with double buffers. TC1 has two clock sources including internal clock and external clock for counting a precision time. The internal clock source is from Fcpu. The external clock is INT1 from P0.1 pin (Falling edge trigger). Using TC1M register selects TC1C‡s clock source from internal or external. If TC1 timer occurs an overflow, it will continue counting and isslolotimesout sinal to triggeo TC1 internunt to r st interoum servicounal

8.4.2 TC1M MODE REGISTER

0DCH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TC1M	TC1ENB	TC1rate2	TC1rate1	TC1rate0	TC1CKS	ALOAD1	TC1OUT	PWM1OUT
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 0 **PWM1OUT:** PWM output control bit.
0 = Disable PWM output.
1 = Enable PWM output. PWM duty controlled by TC1OUT, ALOAD1 bits.

Bit 1 **TC1OUT:** TC1 time out toggle signal output control bit. **Only valid when PWM1OUT = 0.**
0 = Disable, P5.3 is I/O function.
1 = Enable, P5.3 is output TC1OUT signal.

Bit 2 **ALOAD1:** Auto-reload control bit. **Only valid when PWM1OUT = 0.**
0 = Disable TC1 auto-reload function.
1 = Enable TC1 auto-reload function.

Bit 3 **TC1CKS:** TC1 clock source select bit.
0 = Internal clock (Fcpu or Fosc).
1 = External clock from P0.1/INT1 pin.

Bit [6:4] **TC1RATE[2:0]:** TC1 internal clock select bits.
000 = fcpu/256.
001 = fcpu/128.
"
110 = fcpu/4.
111 = fcpu/2.

Bit 7 **TC1ENB:** TC1 counter control bit.
0 = Disable TC1 timer.
1 = Enable TC1 timer.

- ***Note: When TC1CKS=1, TC1 became an external event counter and TC1RATE is useless. No more P0.1 interrupt request will be raised. (P0.1IRQ will be always 0).***

esup.-0.16224 Tc ( ) Tj 35932448 6,1(p) Tj0 Tc (t) Tj0.07776 Tc ( ) Tj-0.08112 Tc (re) Tj-0.15888 Tc (q) Tj0.08112 T

est

8.4.3 TC1C COUNTING REGISTER

TC1C is an 8-bit counter register for TC1 interval time control.

xe X

0DDH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TC1C	TC1C7	TC1C6	TC1C5	TC1C4	TC1C3	TC1C2	TC1C1	TC1C0
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 TC1C initial value is as following.

TC1C initial value = N - (TC1 interrupt interval time * input clock)

N is TC1 overflow boundary number. TC1 timer overflow time has six types (TC1 timer, TC1 event counter, TC1 Fcpu clock source, TC1 Fosc clock source, PWM mode and no PWM mode). These parameters decide TC1 overflow time and valid value as follow table.

TC1CKS	PWM1	ALOAD1	TC1OUT	N	TC1C valid value	TC1C value binary type	Remark
	0	x	x	256	0x00~0xFF	00000000b~11111111b	Overflow per 256 count

8.4.4 TC1R AUTO-LOAD REGISTER

TC1 timer is with auto-load function controlled by ALOAD1 bit of TC1M. When TC1C overflow occurring, TC1R value will load to TC1C by system. It is easy to generate an accurate time, and users don#t reset TC1C during interrupt service routine.

TC1 is double buffer design. If new TC1R value is set by program, the new value is stored in 1st buffer. Until TC1 overflow occurs, the new value moves to real TC1R buffer. This way can avoid TC1 interval time error and glitch in PWM and Buzzer output.

- ***Note: Under PWM mode, auto-load is enabled automatically. The ALOAD1 bit is selecting overflow boundary.***

0DEH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TC1R	TC1R7	TC1R6	TC1R5	TC1R4	TC1R3	TC1R2	TC1R1	TC1R0
Read/Write	W	W	W	W	W	W	W	W
After reset	0	0	0	0	0	0	0	0

The equation of TC1R initial value is as following.

TC1R initial value = N - (TC1 interrupt interval time * input clock)

N is TC1 overflow boundary number. TC1 timer overflow time has six types (TC1 timer, TC1 n0.55776 T ( ) Tj0 Tc8 Tc (e) Tj0 Tc

8.4.5 TC1 CLOCK FREQUENCY OUTPUT (BUZZER)

Buzzer output (TC1OUT) is from TC1 timer/counter frequency output function. By setting the TC1 clock frequency, the clock signal is output to P5.3 and the P5.3 general purpose I/O function is auto-disable. The TC1OUT frequency is divided by 2 from TC1 interval time. TC1OUT frequency is 1/2 TC1 frequency. The TC1imTT

8.4.6 TC1 TIMER OPERATION SEQUENCE

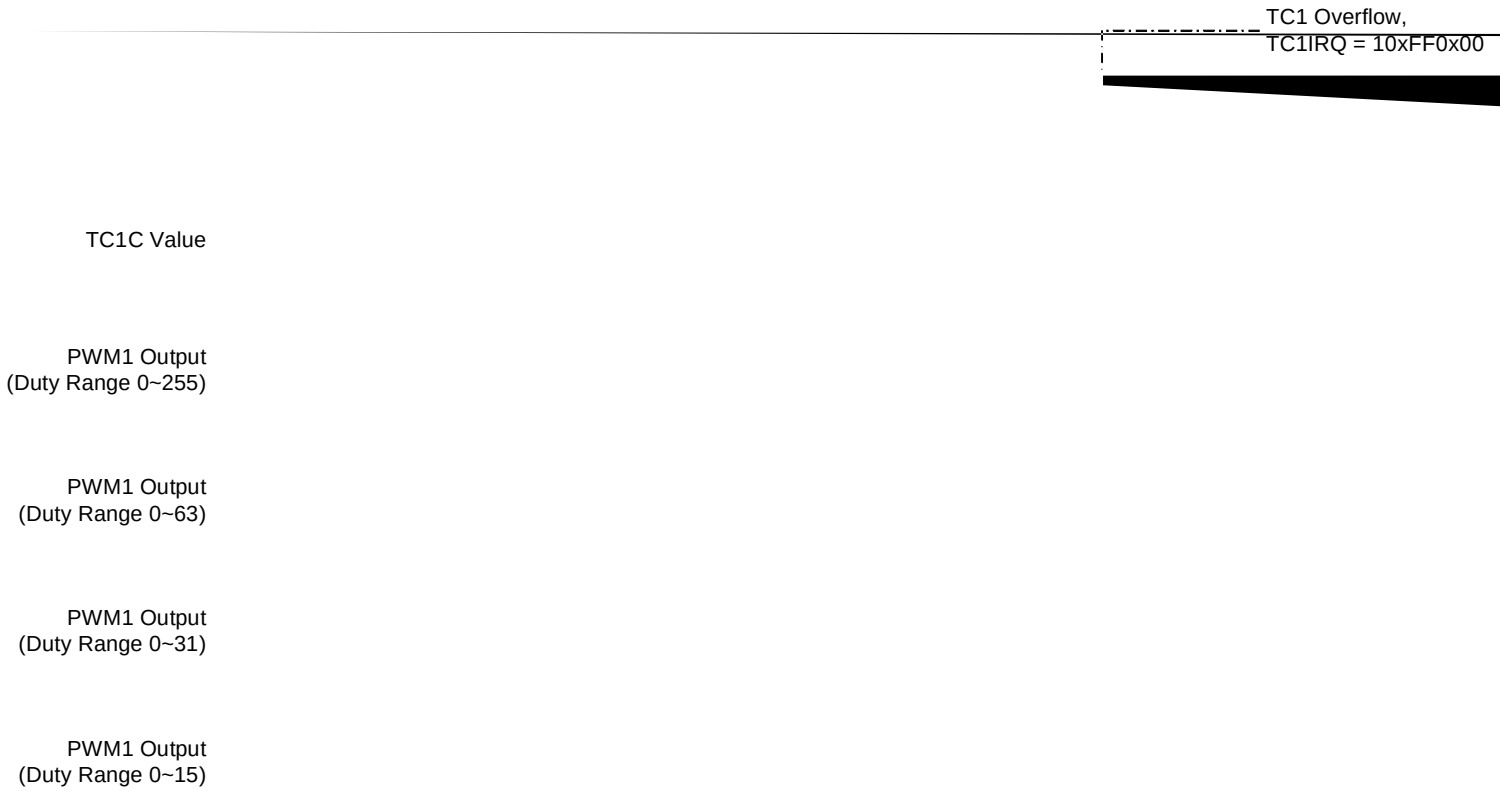
8.5 PWM1 MODE

8.5.1 OVERVIEW

PWM function is generated by TC1 timer counter and output the PWM signal to PWM1OUT pin (P5.3). The 8-bit counter counts modulus 256, 64, 32, 16 controlled by ALOAD1, TC1OUT bits. The value of the 8-bit counter (TC1C) is compared to the contents of the reference register (TC1R). When the reference register value (TC1R) is equal to the counter C1R he 88lel8g

8.5.2 TC1IRQ AND PWM DUTY

In PWM mode, the frequency of TC1IRQ is depended on PWM duty range. From following diagram, the TC1IRQ frequency is related with PWM duty.



8.5.3 PWM PROGRAM EXAMPLE

Ø Example: Setup PWM1 output from TC1 to PWM1OUT (P5.3). The external high-speed oscillator clock is 4MHz. Fcpu = Fosc/4. The duty of PWM is 30/256. The PWM frequency is about 1KHz. The PWM clock source is from external oscillator clock. TC1 rate is Fcpu/4. The TC1RATE2~TC1RATE1 = 110. TC1C = TC1R = 30.

```
MOV     A,#01100000B
B0MOV   TC1M,A          ; Set the TC1 rate to Fcpu/4

MOV     A,#30           ; Set the PWM duty to 30/256
B0MOV   TC1C,A
B0MOV   TC1R,A

B0BCLR  FTC1OUT         ; Set duty range as 0/256~255/256.
B0BCLR  FALOAD1
B0BSET  FPWM1OUT        ; Enable PWM1 output to P5.3 and disable P5.3 I/O function
B0BSET  FTC1ENB         ; Enable TC1 timer
```

- ***Note: The TC1R is write-only register. Don t process them using INCMS, DECMS instructions.***

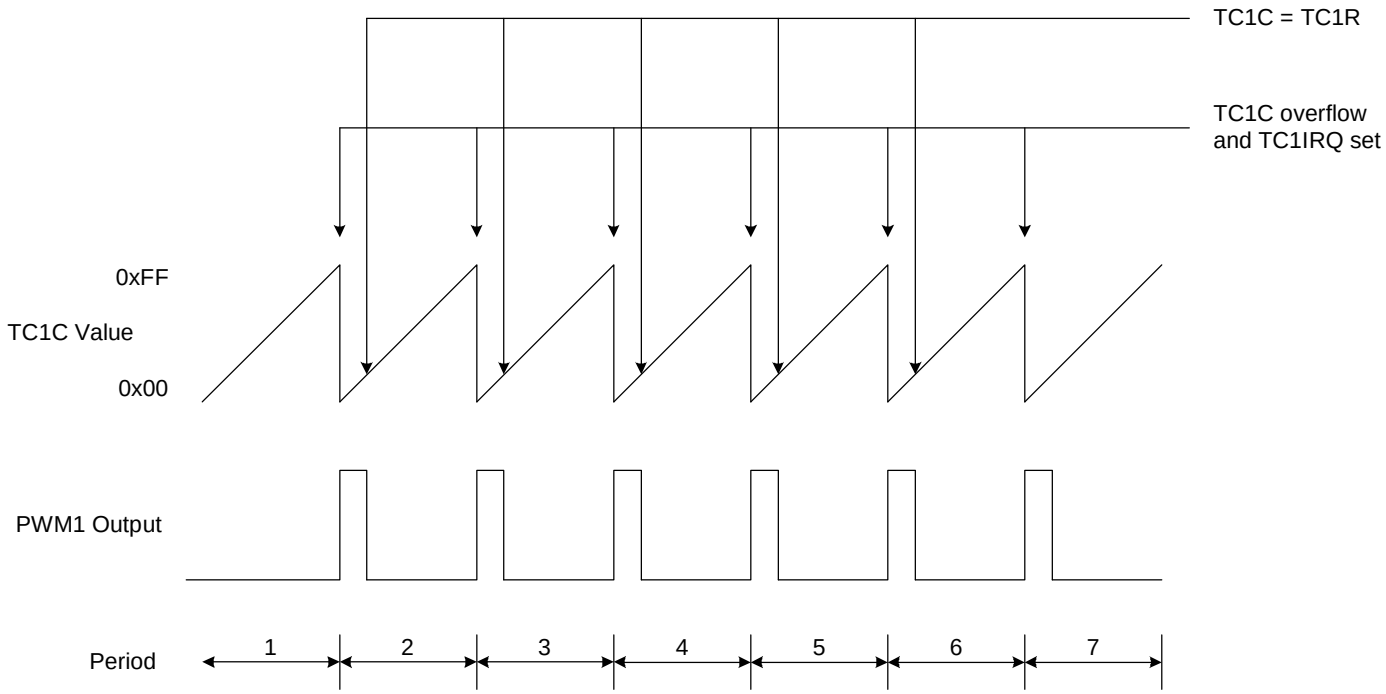
Ø Example: Modify TC1R registers' value.

```
MOV     A, #30H         ; Input a number using B0MOV instruction.
B0MOV   TC1R, A

INCMS   BUF0            ; Get the new TC1R value from the BUF0 buffer defined by
NOP                     ; programming.
B0MOV   A, BUF0
B0MOV   TC1R, A
```

N N

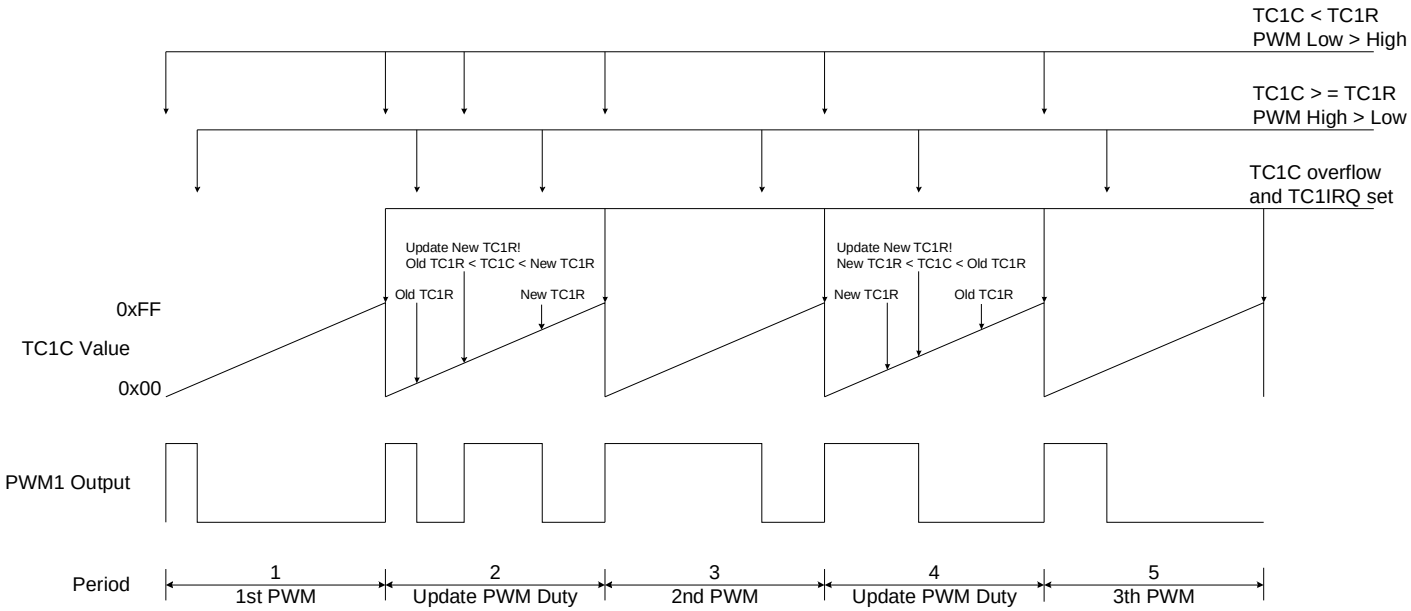
8.5.4 PWM1 DUTY CHANGING NOTICE

In PWM mode, the system will compare TC1C and TC1R all the time. When TC1C<TC1R, the PWM will output logic High!, when TC1C ! TC1R, the PWM will output logic Low!. If TC1C is changed in certain period, the PWM duty will change immediately. If TC1R is fixed all the time, the PWM waveform is also the same.



Above diagram is shown the waveform with fixed TC1R. In every TC1C

If TC1R is changing in the program processing, the PWM waveform will became as following diagram.

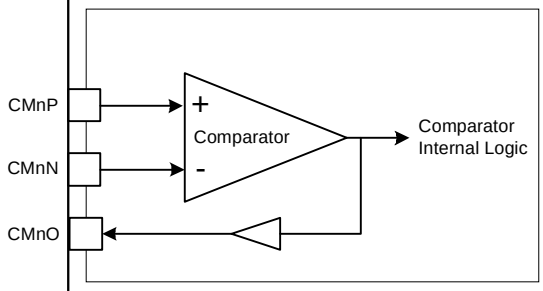


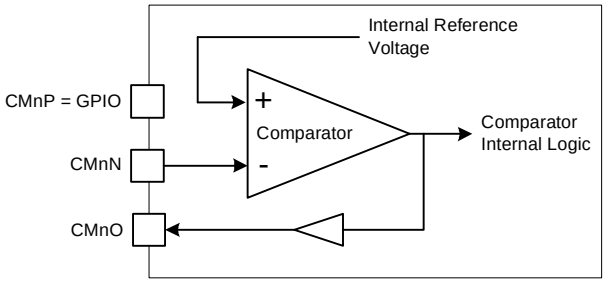
I

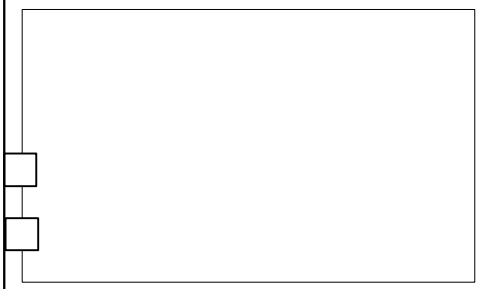
9 ANALOG COMPARATOR

9.1 OVERVIEW

The analog comparator function includes two channel analog comparators and internal reference voltage. The main purpose of the comparator is to compare DC power voltage for low power indicator. The analog comparator negative input pin, positive input pin and outpuin and out


CMnP
CMnN
CMnO
+
Comparator
-
Comparator
Internal Logic


Internal Reference
Voltage
CMnP = GPIO
CMnN
CMnO
+
Comparator
-
Comparator
Internal Logic



9.2 CMP0M REGISTER

e/tesetrse/

09CH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMP0M	CM0EN	CM0IEN	CM0IRQ	CM0OEN	CM0REF	CM0OUT	CMS1	CMS0
Read/Write	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W

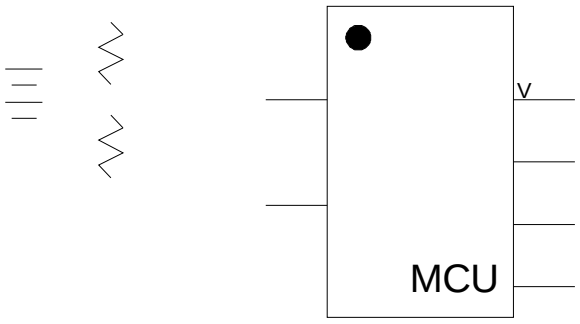
9.3 CMP1M REGISTER

09DH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMP1M	CM1EN	CM1IEN	CM1IRQ	CM1OEN	CM1REF	CM1OUT	-	-
Read/Write	R/W	R/W	R/W	R/W	R/W	R	-	-
After reset	0	0	0	0	0	0	-	-

Bit 7 **CM1EN:** Comparator 1 control bit.
0 = Disable. All comparator pins are GPIO.
1 =o e. ar1 Cam1EN: Cons7 ator s1 bit7

9.4 ANALOG COMPARATOR APPLICATION

This is a using the analog comparator to do two levels low battery detector. There are two low battery levels which are 2.2V and 2.0V. When the battery level is less than 2.2V, the system does low power process. When the battery level is less than 2.0V, the system does no power process. The battery detect level is 1/2 bias voltage of battery power source. The comparator positive voltage (reference voltage) is comparator internal reference voltage. The application circuit is as following.

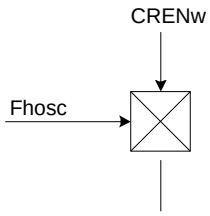


```
        JMP         LowBat      ; Is low battery status, go to low battery routine.
        JMP         NoBat       ; Is no battery status, go to no battery routine.

; Low battery process.
LowBat:
        "
        "
        JMP         Main        ; Return to main loop.

; No battery process.
NoBat:
        "
        "
        JMP         Main        ; Return to main loop.
```

10 IR OUTPUT

10.1 OVERVIEW

IR signal is generated by IR timer. The IR output pin is IROUT pin. When IREN bit of IRM is set as logic 1!, IROUT pin exchanges from GPIO to IR output mode. If CREN = 0 or system is in power down mode, IROUT pin is tied to low status. The IR timer is an 8-bit binary up counting timer for IR signal generator. The IR signal is duty/cycle changeable type controlled by IRR and IRD. IRR decides IR#s cycle and IRD decides IR#s duty. IR counter clock source is only from Fhosc (system high clock source), eg. IHRC_8M, 4MHz or 455KHz crystal. If Fhosc is 4MHz, the IR counter clock rate is 4MHz. IR timer only generate IR output and no interrupt function. When enable IR output function (CREN=1), IR output status is low level. IRC initial value is IRR and starts to count. When IRC=IRD, IR output status change to low level and finishes high duty operation. When IRC overflow occurs (IRC changes from 0xFF to 0x00), IR output low duty operation stops. System loads IRR into IRC automatically and next cycle starts.



8

10.2 IR CONTROL REGISTER

10.2.1 IRM MODE REGISTER

0DAH	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRM	-	-	-	-	-	-	IREN	CREN
Read/Write	-	-	-	-	-	-	R/W	R/W
After reset	-	-	-	-	-	-	0	0

Bit 1 **IREN:** IROUT pin control bit.
0 = Disable

10.2.3 IRR AUTO-LOAD REGISTER

IRR decides IR signal frequency. IR timer is with auto-load function. When IRC overflow occurs, IRR value will load to IRC. It is easy to generate an accurate time for IR signal cycle, and users don#t reset IRC during interrup41.03776 Tc Tc

10.2.4 IRD IR DUTY CONTROL REGISTER

The IR signal is duty changeable by IRD. IRD decides the IR output signal low pulse width length. When IRC=IRD, the IR signal changes fc (O) Tj-0Tc (s) Tj0.16224 T (fc (O19 Tj-0.0) Tj0.31776 Tc Tjo19 Tj-0.0)mj0.16224 Tc (i) Tj-0.0841

10.2.5 IR OUTPUT OPERATION SEQUENCE

F Set IRC and IRR for IR cycle.

```
MOV     A, #IRCYCVAL    ;IRC, IRR value for IR cycle.
MOV     IRC, A
MOV     IRR, A
```

F Set IRD for IR duty.

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

F Enable IR output.

```
BSET    FIREN           ; Set IROUT pin to IR carry output function.
BSET    FCREN           ; Set IR carry signal output.
```

11 SERIAL INPUT/OUTPUT TRANSCEIVER (R81 -t (X) Tj0.1.084 Tca (X) TE

The SIO supports 8-mode format controonone1(O) Tj1.l.07776 Tc (t) Tj0ed08448 Tc (8) Tj0 Tc (-) Tj0.24336 bTc (o) Tj-yc (c)

12.6 UART DATA BUFFER

URTXD1 initial value = 0000 0000 i

13 INSTRUCTION TABLE

Field	Mnemonic		Description	C	DC	Z	Cycle
	MOV	A,M	A ← M	-	-	√	1
		M,A	M ← A	-	-	-	1

14 ELECTRICAL CHARACTERISTIC

14.1 ABSOLUTE MAXIMUM RATING

Supply voltage (Vdd) - 0.3V ~ 6.0V
Input in voltage (Vin) Vss % 0.2V ~ Vdd + 0.2V
Operating ambient temperature (Topr)
SN8P26L38P, SN8P2L638X, SN8P26L38F 0°C ~ + 70°C
Storage ambient temperature (Tstor) %40°C ~ + 125°C

14.2 ELECTRICAL CHARACTERISTIC

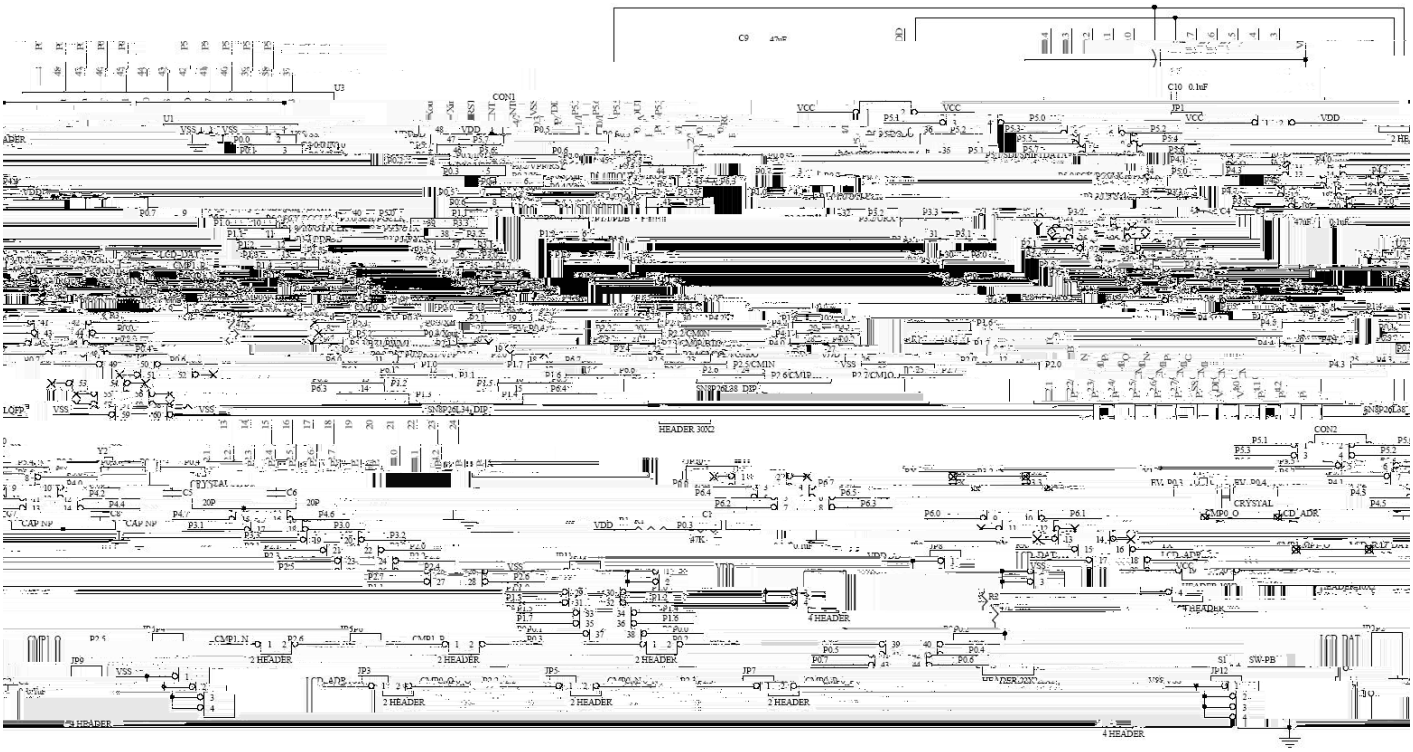
(All of voltages refer to Vss, Vdd = 3.0V, fosc = 4MHz, Fcpu=1MHZ, ambient temperature is 25°C unless otherwise note.)

PARAMETER	*SYM.*	*DESCRIPTION*	*MIN.*	*TYP.*	*MAX.*	*UNIT*
Operating voltage	Vdd	Normal mode, Vpp = Vdd, 25 "		3 TD (") Tj /F5 7.92 Tre f 0 TD M Tc 0.193		

15 DEVELOPMENT TOOL

SONIX provides ICE (in ci

- SN8P26L38 EV-kit Circuit



15.2 ICE and EV-KIT APPLICATION NOTIC

SN8P26L38 EV-kit includes comparator emulation module. There is a SN8P26L38P chip programmed emulating code to emulate comparator function. The SN8P26L38 comparator pins are shared with P2 GPIO pins. In ICE environment, the comparator pins isn't connected with GPIO pin.

- The Comparator emulation is from the SN8P26L34 EV-chip of SN8P26L38 EV-kit. For comparator emulation, input and output comparator signals from these pins.
- The P2 comparator shared pin GPIO emulation is from P2 pins of SN8P26L38 EV-kit.
- The "SN8P26L38" EV-kit po-ki10.07776 0.08448 Tc (26L38) Tj0.02 TD (l)2.0c (i10.07776 0.084 Tj-0.08448 Tc8448 Tc (po

i10.48 Tc (pa) Tj-0.242he

16 OTP PROGRAMMING P

16.1 The pin assignment of Easy Writer transi

Easy Writer JP1/JP2

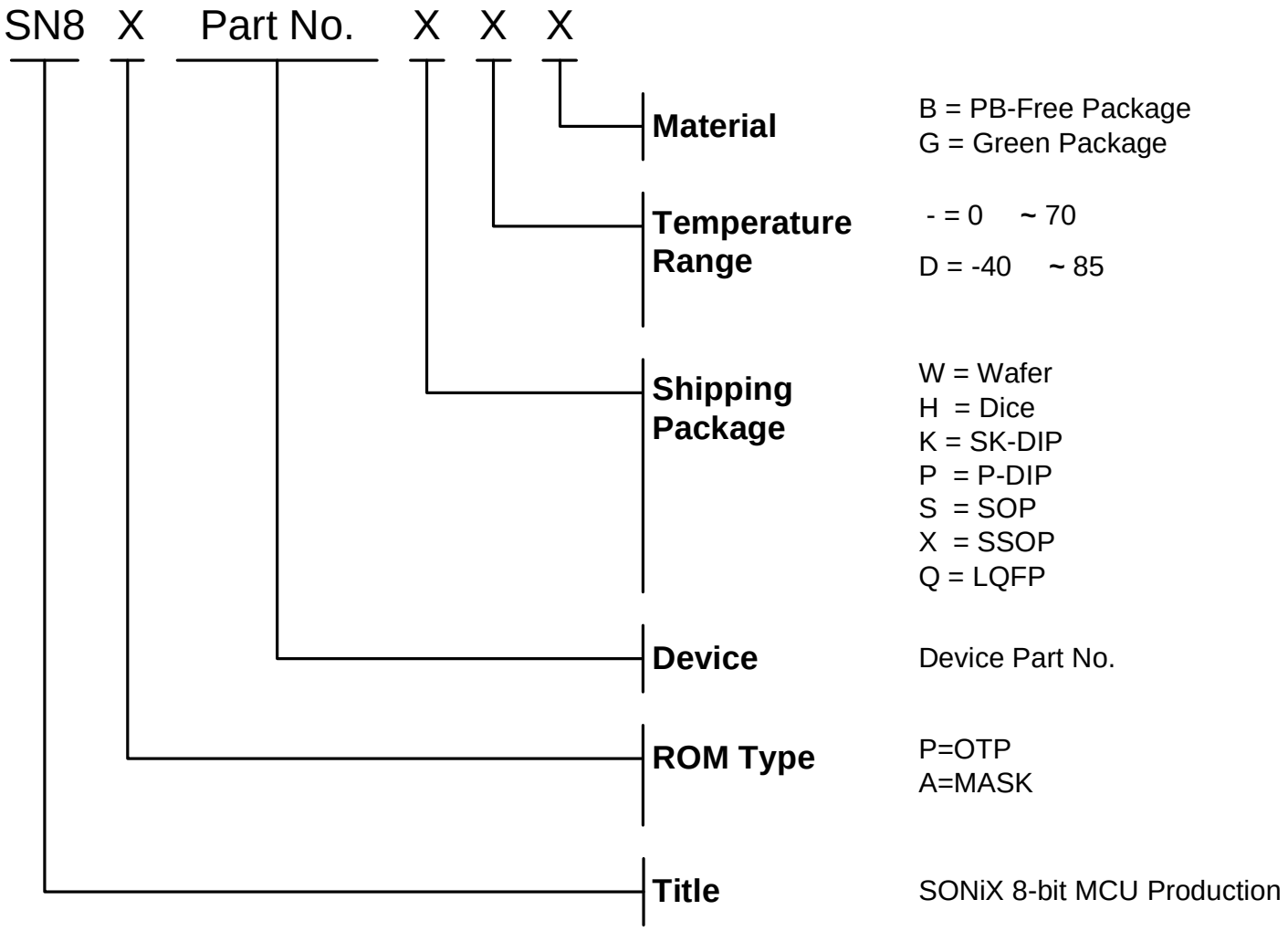
Easy Writer JP3 (

17 Marking Definition

17.1 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.

17.2 MARKING INDETIFICATION SYSTEM

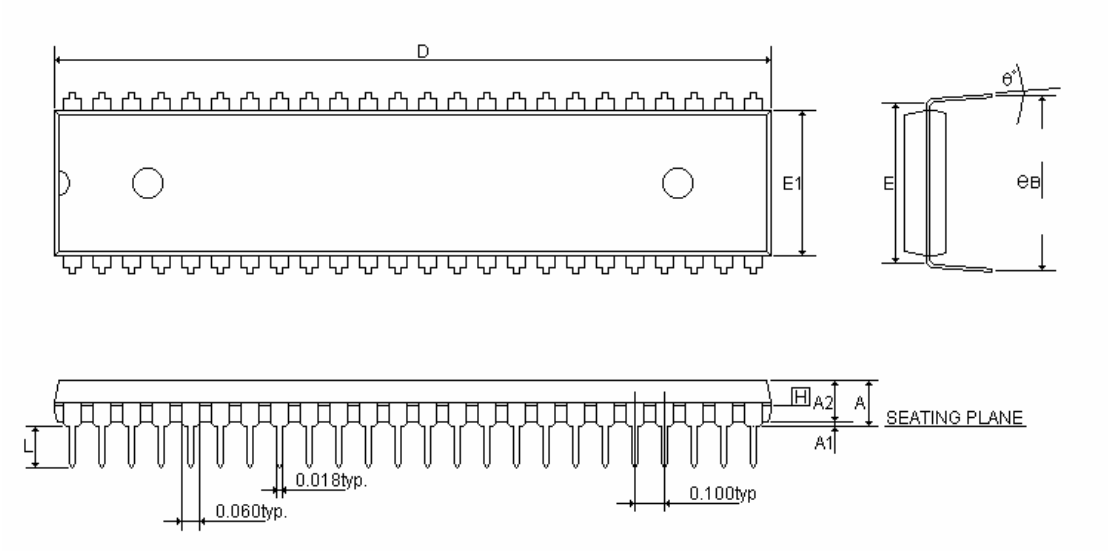


- ***Note: SN8P26L38 doesn t support -40℃~85℃ temperature range and MASK ROM type.***

17.3 MARKING EXAMPLE M33

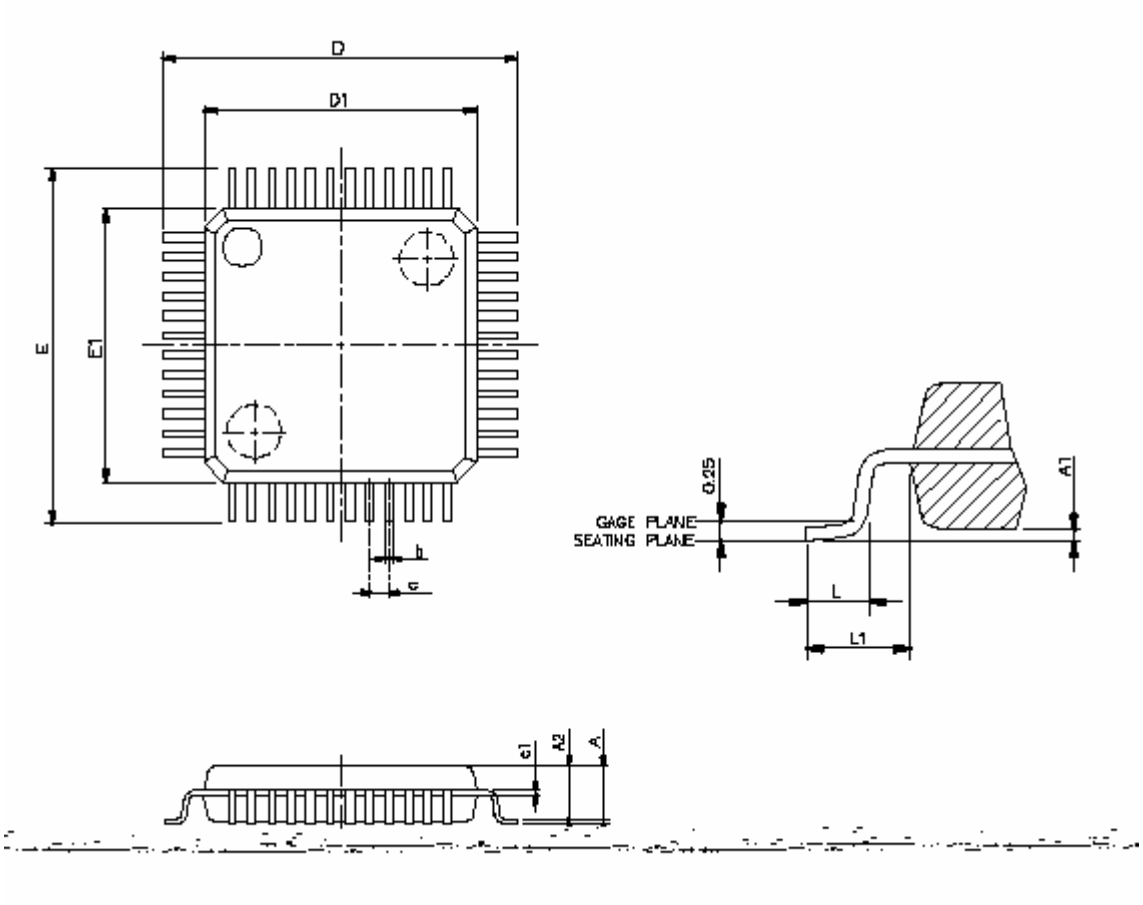
18 PACKAGE INFORMATION

18.1 P-DIP 48 PIN



SYMBOLS	N					
	MIN	NOR	MAX	MIN	NOR	MAX
	(inch)			(mm)		
A	-	-				

18.3 LQFP 48 PIN



SYMBOLS	MIN	NOR	MAX	MIN	NOR	MAX
	(inch)			(mm)		
A	-	-	0.063	-	-	1.6
A1	0.002	-	0.006	0.05	-	0.15 .

SONIX reserves the right to make change without further notice to any products herein to improve reliability, function or design. SONIX does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. SONIX products are not designed, intended, or authorized for us as components in systems intended, for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the SONIX product could create a situation where personal injury or death may occur. Should Buyer purchase or use SONIX products for any such unintended or unauthorized application. Buyer shall indemnify and hold SONIX and its officers , employees, subsidiaries, affiliates and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use even if such claim alleges that SONIX was negligent regarding the design or manufacture of the part.

Main Office:
Address: 10F-1, NO. 36, Taiyuan Stree., Chupei City, Hsinchu, Taiwan R.O.C.
Tel: 886-3-5600 888
Fax: 886-3-5600 889
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:
Unit 1519, Chevalier Commercial Center, No.8 Wang Hoi Road, Kowloon Bay, Kowloon. Hong Kong.
Tel: 852-2723-8086
Fax: 852-2723-9179
Technical Support by Email:
Sn8fae@sonix.com.tw