**Preliminary**

# TTR011

## § Features

- ◆ Tontek RISC 4-bit CPU core with 2-level stacks
- ◆ All instructions are one word length
- ◆ Most of instructions need one machine cycle (2 CPU clocks) except read table instruction RTB
- ◆ Advanced CMOS process
- ◆ 1Kx16 OTP ROM
- ◆ 56x4 SRAM
- ◆ Operating voltage: 2.0V~5.50V
- ◆ Two high-speed clock OSCH selectable: on-chip 910kHz oscillator, external RC oscillator
- ◆ One low-speed clock OSCL: on-chip 16kHz oscillator
- ◆ Power-on default is low speed mode
- ◆ Four operation modes
  - STOP mode: CPU, OSCH and OSCL stop
  - GREEN mode: CPU and OSCH stop; OSCL active for periodical ADC measure and wake-up
  - Low power mode: OSCH stop, OSCL active for CPU and peripheral circuit
  - Normal mode: Both OSCH and OSCL active, CPU clock from OSCH
- ◆ 11 I/O pins + 1 input
  - PA, PB and PC can be configured as wake-up input pins
  - PB and PC can be programmed as input pins with pull-high resistors
  - Four input pins can be defined as external interrupt input pins
  - Shared with 5 ADC input channels and 1 buzzer output
- ◆ 5-channel 4-bit ADC
- ◆ One time base timer
  - In GREEN mode, its clock is from the overflow of timer/counter B and its overflow wake up CPU from GREEN mode.
  - In other mode, its clock is from OSCL
- ◆ Two 8-bit timer/counter
  - One 8-bit timer/counter B with two sets of auto-preload data to generate periodic waveform for buzzer function.
  - In GREEN mode, the overflow of timer B will trigger ADC check procedure and increment of time base timer 1.
  - One 8-bit timer/counter C with auto reload function

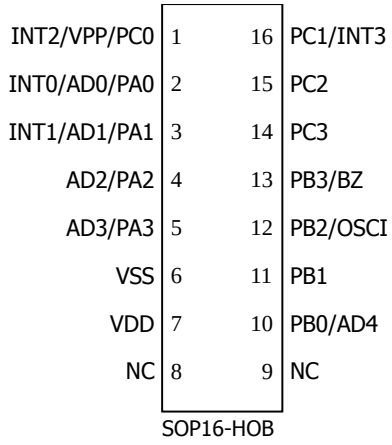
**Preliminary**

- ◆ Eight interrupts:
  - Internal interrupts: ADC, timer/counter B, timer/counter C, time base timer
  - External interrupts: INT0, INT1, INT2, INT3
- ◆ Two on-chip system protections:
  - Watch dog timer(WDT) circuit
  - 4-level low voltage detector: 2.0V, 2.2V, 3.3V, 4.5V
- ◆ MCU executes next address instruction or interrupt servicing after wake up
- ◆ Package: SOP 8/16 pins

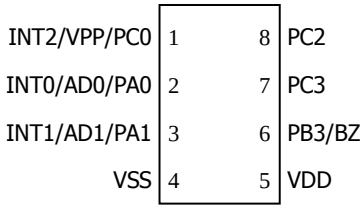
# § Applications

TTR011 can be used in alarm systems in supermarkets, discount stores, etc.

# § Pin configuration




SOP16-HOB




SOP8-FO8

# § Ordering Form:

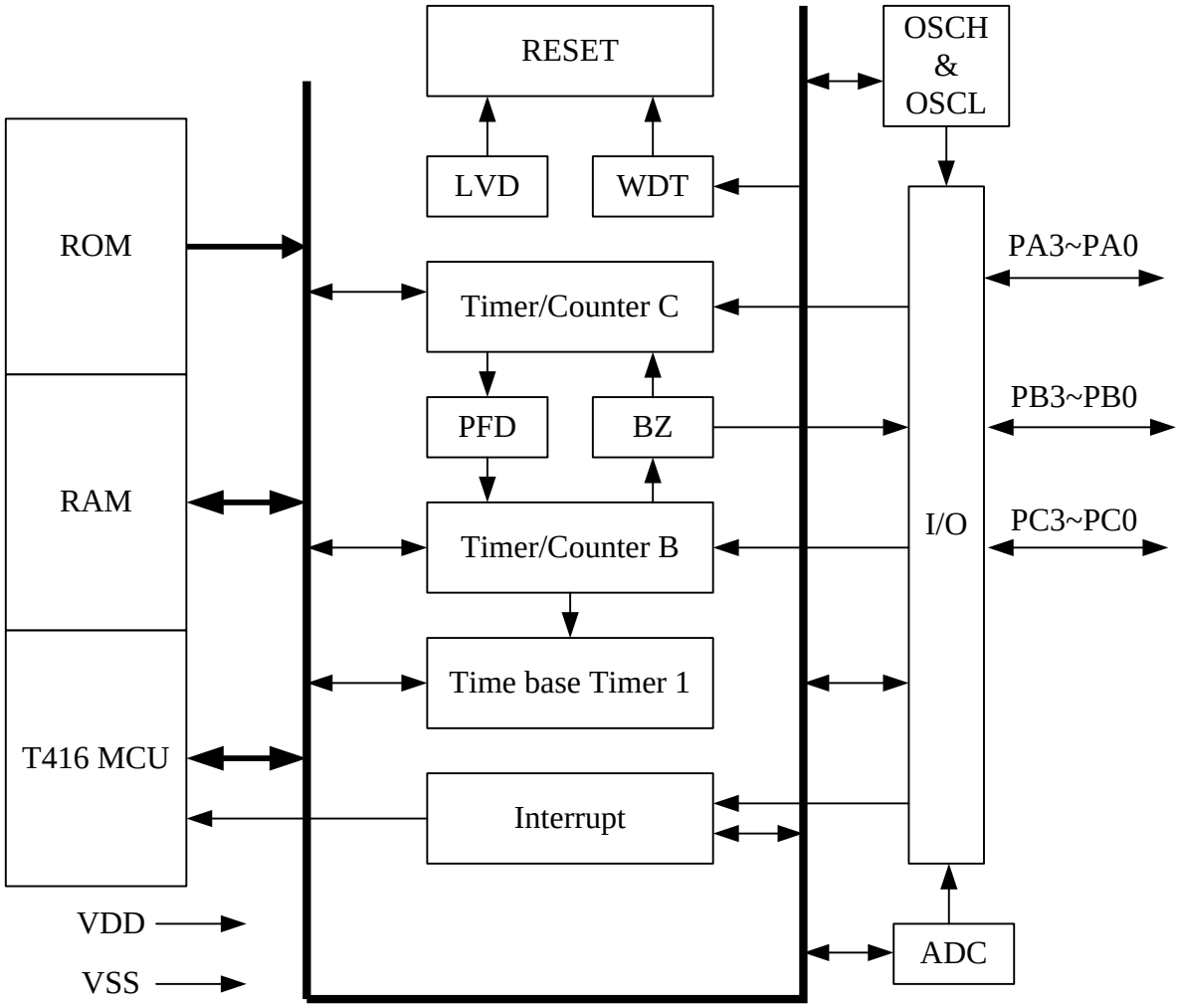
a. Package form : TTR011-zzz

b. Chip form :

c. Wafer base :

# § Block Diagram

**Figure 1: Block diagram**



# § Pin Descriptions

| Pin Name | Function | Type | Pin Description |
|---|---|---|---|
| **PA0**/INT0/AD0 | PA0 | I/O | Input or output port determined by software |
| | INT0 | I | External interrupt input as input mode determined |
| | AD0 | I | ADC input channel configured by software |
| **PA1**/INT1/AD1 | PA1 | I/O | Input or output port determined by software |
| | INT1 | I | External interrupt input as input mode determined |
| | AD1 | I | ADC input channel configured by software |
| **PA2**/AD2 | PA2 | I/O | Input or output port determined by software |
| | AD2 | I | ADC input channel configured by software |
| **PA3**/AD3 | PA3 | I/O | Input or output port determined by software |
| | AD3 | I | ADC input channel configured by software |
| **PB0**/AD4 | PB0 | I/O | Input with or without pull-high resistor or output port determined by software |
| | AD4 | I | ADC input channel configured by software |
| **PB1** | PB1 | I/O | Input with or without pull-high resistor or output port determined by software |
| **PB2**/OSCI | PB2 | I/O | Input with or without pull-high resistor or output port determined by software |
| | OSCI | I | external oscillator PAD Connect resister to VSS |
| **PB3**/BZ | PB3 | I/O | Input with or without pull-high resistor or output port determined by software |
| | BZ | O | Buzzer output |
| **PC0**/INT2/VPP | PC0 | I | Input port with or without pull-high resistor determined by software<br>One of the TCPC clock sources as input mode determined |
| | INT2 | I | External interrupt input as input mode determined |
| | VPP | P | OTP ROM Programming pin: 12.5V |
| **PC1**/INT3 | PC1 | I/O | Input with or without pull-high resistor or output port determined by software<br>One of the TCPC clock sources as input mode determined |
| | INT3 | I | External interrupt input as input mode determined |
| **PC2** | PC2 | I/O | Input with or without pull-high resistor or output port determined by software<br>One of the TCPB clock sources as input mode determined |
| **PC3** | PC3 | I/O | Input with or without pull-high resistor or output port determined by software |
| **VDD** | VDD | Power | Positive power supply |
| **VSS** | VSS | Power | Negative power supply, ground |

# § Absolute Maximum Ratings

| ITEM | SYMBOL | RATING |
|---|---|---|
| Operating Temperature | Top | -20℃ ~ +70℃ |
| Storage Temperature | Tst | -50℃ ~ +125℃ |
| Supply Voltage | VDD | VSS-0.3V ~ VSS+6.0V |
| OTP Supply Voltage | VPP | VSS-0.3V ~ VSS+12.5V |
| Input Voltage | Vin | VSS-0.3V ~ VDD+0.3V |
| ESD(Human Body Mode) | ESD | >5kV |
| Note: VSS symbolizes for system ground | | |

# § DC Characteristics

| Parameter | Symbol | Test Condition | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Operating Voltage | VDD | OSCH generated by external RC oscillator | 2.0 | | 5.5 | V |
| | | OSCH generated by on-chip 910kHZ oscillator | 2.0 | | 5.5 | V |
| Operating Current | $I_{nd1}$ | Normal mode, no load, VDD=3.0V, OSCH=3.64MHz | | 1.0 | 2.0 | mA |
| | $I_{nd3}$ | Normal mode, no load, VDD=3.0V, OSCH=910kHz | | 0.4 | 0.7 | |
| GREEN mode Current | Istb1 | ADC OFF, OSCH stop, OSCL active, VDD=3.0V, no load | | 0.8 | 1.2 | uA |
| | Istb2 | ADC measurement cycle time=32ms, OSCH stop, OSCL active, VDD=3.0V, no load | | 1.1 | 1.5 | uA |
| Input low voltage | $V_{IL}$ | Input Low Voltage | 0 | | 0.2 | VDD |
| Input high voltage | $V_{IH}$ | Input High Voltage | 0.8 | | 1.0 | VDD |
| Sink Current of output | $I_{OL}$ | VDD=3.0V, $V_{OL}$=0.6V | 2 | 4 | | mA |
| Source Current of output | $I_{OH}$ | VDD=3.0V, $V_{OH}$=2.4V | | -4 | -2 | mA |
| Pull-high Resistor of PB and PC | $R_{PH}$ | VDD=3.0V | 50 | 100 | 150 | KΩ |
| (ambient temperature is 25℃) | | | | | | |

# § AC Characteristics

| Parameter | Test Condition | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| High-speed clock OSCH Frequency | External RC oscillator | 400K | | 4M | Hz |
| | On-chip 910kHz oscillator | 882k | 910K | 937k | |
| System stable time after power up | Stable time=(OSCL startup time)+(1/OSCL)X256 | | 17 | | ms |
| Wake up time | Wake up time to low power mode =(OSCL startup time)+(1/OSCL)X4 | | 1 | | ms |
| | Wake up time to normal mode =(OSCH startup time)+(1/OSCH)X4 | | 1 | | ms |
| (VDD=3V, ambient temperature is 25℃) | | | | | |

# § Memory Map

| ROM ADDRESS | RAM ADDRESS | Function |
|---|---|---|
| $000_H$ | | Reset vector<br>Jump to start address |
| $001_H$ | | Interrupt vector |
| $002_H$~$3FF_H$ | | Program ROM |
| | $000_H$~$007_H$ | File registers |
| | $008_H$~$01F_H$ | Peripheral registers |
| | $020_H$~$057_H$ | Working RAM [56x4] |
| | $058_H$~$068_H$ | Peripheral registers |

# § File registers and Peripheral registers:

This table gives a list of the file registers and peripheral registers. "**Default"** indicates each register status upon power on reset. **"R/W"** indicates the register can be read and written. **"R/W0"** indicates the register can be read and only written to "0". **"R"** is read-only. **"W"** is write-only.

| Address | Resister | Bit3 | Bit2 | Bit1 | Bit0 | R/W | Default | Description |
|---|---|---|---|---|---|---|---|---|
| $000_H$ | (DP1) | - | - | - | - | R/W | uuuu | Indirect addressing register |
| $001_H$ | ACC | ACC3 | ACC2 | ACC1 | ACC0 | R/W | uuuu | Accumulator & Read Table 1st data |
| $002_H$ | TB1 | D7 | D6 | D5 | D4 | R/W | uuuu | Read Table 2nd data |
| $003_H$ | TB2 | D11 | D10 | D9 | D8 | R/W | uuuu | Read Table 3rd data |
| $004_H$ | TB3 | D15 | D14 | D13 | D12 | R/W | uuuu | Read Table 4th data |
| $005_H$ | DPL | A3 | A2 | A1 | A0 | R/W | 0000 | Data Pointer low nibble |
| $006_H$ | DPM | A7 | A6 | A5 | A4 | R/W | 0000 | Data Pointer middle nibble |
| $007_H$ | DPH | A11 | A10 | A9 | A8 | R/W | 0000 | Data Pointer high nibble |
| $008_H$ | PS | LVR_EN | H/L | GREEN | STOP | R/W | 0000 | Power saving control register |
| $009_H$ | INTC1 | ADC_IE | TCPC_IE | TCPB_IE | TB1_IE | R/W | 0000 | Internal interrupt enable register |
| $00A_H$ | INTF1 | ADC_IF | TCPC_IF | TCPB_IF | TB1_IF | R/W0 | 0000 | Internal interrupt request flag register |
| $00B_H$ | CTLA | LVD_DET (R) | LVD_EN (R/W) | TCPCD_LD (R/W) | TCP_RD (R/W) | | 0000 | LVD and timer/counter Control register |
| $00C_H$ | TBC/BZC | BZ_EN | - | TB1 | TB0 | R/W | 0-11 | Time Base Timer/buzzer control register |
| $00D_H$ | PAC | PAC3 | PAC2 | PAC1 | PAC0 | R/W | 1111 | Port A control register |
| $00E_H$ | - | - | - | - | - | - | - | Reserved |
| $00F_H$ | PA | PA3 | PA2 | PA1 | PA0 | R/W | uuuu | Port A data register |
| $010_H$ | PB | PB3 | PB2 | PB1 | PB0 | R/W | uuuu | Port B data register |
| $011_H$ | PC | PC3 (R/W) | PC2 (R/W) | PC1 (R/W) | PC0 (R) | | uuuu | Port C data register |
| $012_H$ | WK_FG | TB_WK | PC_WK | PB_WK | AD/PA_WK | R/W0 | 0000 | Wake up flag register |
| $013_H$ | TCPBC | TCPB_LD | TCPB_CK1 | TCPB_CK0 | TCPB_EN | R/W | 0000 | Timer/counter B(TCPB) control register |
| $014_H$ | TCPBDLL | TCPBDL3 | TCPBDL2 | TCPBDL1 | TCPBDL0 | R/W | 0000 | Low nibble register for low data TCPBDL |
| $015_H$ | TCPBDLH | TCPBDL7 | TCPBDL6 | TCPBDL5 | TCPBDL4 | R/W | 0000 | High nibble register for low data TCPBDL |
| $016_H$ | TCPBDHL | TCPBDH3 | TCPBDH2 | TCPBDH1 | TCPBDH0 | R/W | 0000 | Low nibble register for high data TCPBDH |
| $017_H$ | TCPBDHH | TCPBDH7 | TCPBDH6 | TCPBDH5 | TCPBDH4 | R/W | 0000 | High nibble register for high data TCPBDH |
| $018_H$ | INTC2 | INT3_IE | INT2_IE | INT1_IE | INT0_IE | R/W | 0000 | External interrupt enable register |
| $019_H$ | INTF2 | INT3_IF | INT2_IF | INT1_IF | INT0_IF | R/W0 | 0000 | External interrupt request flag register |
| $01A_H$ | ADC_CTL1 | ADC_EN | AD_CH2 | AD_CH1 | AD_CH0 | R/W | 0000 | ADC control register 1 |
| $01B_H$ | TCPCC | TCPC_CK2 | TCPC_CK1 | TCPC_CK0 | TCPC_EN | R/W | 0000 | Timer/counter C(TCPC) control register |
| $01C_H$ | TCPCDL | TCPCD3 | TCPCD2 | TCPCD1 | TCPCD0 | R/W | 0000 | Low nibble register for data TCPCD |

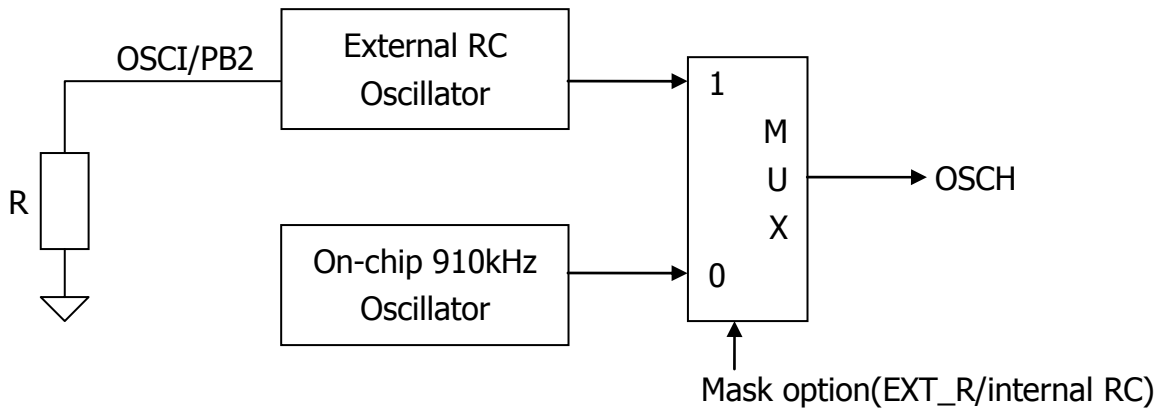
| Address | Resister | Bit3 | Bit2 | Bit1 | Bit0 | R/W | Default | Description |
|---|---|---|---|---|---|---|---|---|
| 01D H | TCPCDH | TCPCD7 | TCPCD6 | TCPCD5 | TCPCD4 | R/W | 0000 | High nibble register for data TCPCD |
| 01E H | PBC | PBC3 | PBC2 | PBC1 | PBC0 | R/W | 1111 | Port B control register |
| 01F H | PCC | PCC3 (R/W) | PCC2 (R/W) | PCC1 (R/W) | PCC0 (R) | | 1111 | Port C control register |
| 058 H | AD0_H | AD0_H3 | AD0_H2 | AD0_H1 | AD0_H0 | R/W | uuuu | High limit voltage for ADC0 wake up function |
| 059 H | AD0_L | AD0_L3 | AD0_L2 | AD0_L1 | AD0_L0 | R/W | uuuu | Low limit voltage for ADC0 wake up function |
| 05A H | AD1_H | AD1_H3 | AD1_H2 | AD1_H1 | AD1_H0 | R/W | uuuu | High limit voltage for ADC1 wake up function |
| 05B H | AD1_L | AD1_L3 | AD1_L2 | AD1_L1 | AD1_L0 | R/W | uuuu | Low limit voltage for ADC1 wake up function |
| 05C H | AD2_H | AD2_H3 | AD2_H2 | AD2_H1 | AD2_H0 | R/W | uuuu | High limit voltage for ADC2 wake up function |
| 05D H | AD2_L | AD2_L3 | AD2_L2 | AD2_L1 | AD2_L0 | R/W | uuuu | Low limit voltage for ADC2 wake up function |
| 05E H | AD3_H | AD3_H3 | AD3_H2 | AD3_H1 | AD3_H0 | R/W | uuuu | High limit voltage for ADC3 wake up function |
| 05F H | AD3_L | AD3_L3 | AD3_L2 | AD3_L1 | AD3_L0 | R/W | uuuu | Low limit voltage for ADC3 wake up function |
| 060 H | ADC_D | ADC_D3 | ADC_D2 | ADC_D1 | ADC_D0 | R | uuuu | ADC data register |
| 061 H | ADC_PA | AD3/PA3 | AD2/PA2 | AD1/PA1 | AD0/PA0 | R/W | 0000 | ADC input channel and PA select register |
| 062 H | AD_WKEN | AD3WKEN | AD2WKEN | AD1WKEN | AD0WKEN | R/W | 0000 | ADC wake up function control register |
| 063 H | ADC_CTL2 | ADCK1 | ADCK0 | - | AD4/PB0 | R/W | 00-0 | ADC control register 2 |
| 064 H | PB_PH | PB3_PH | PB2_PH | PB1_PH | PB0_PH | R/W | 0000 | Port B pull-high register |
| 065 H | PC_PH | PC3_PH | PC2_PH | PC1_PH | PC0_PH | R/W | 0000 | Port C pull-high register |
| 066 H | PT_WKEN | - | PC_WKEN | PB_WKEN | PA_WKEN | R/W | -000 | Port wake up function control register |
| 067 H | Mod_RC16KL | M16K3 | M16K2 | M16K1 | M16K0 | R/W | 0000 | Low nibble of RC16hz modify register |
| 068 H | Mod_RC16KH | M16K7 | M16K6 | M16K5 | M16K4 | R/W | 1000 | High nibble of RC16hz modify register |

# § System Clock and Operation Modes

There are two clocks, OSCH and OSCL, to be supplied to the CPU and peripheral hardware. OSCL is the low-speed clock. It is generated from on-chip 16kHz oscillator.

OSCH is the high-speed clock. It can be generated from on-chip 910kHz oscillator or external RC type oscillator that is defined by mask option. If external RC type oscillator is selected, there should be a resistor connected between OSCI pin and VSS.

**Figure 2: OSCH circuit**



✧ PS : Power saving control register [R/W] : $008_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | LVR_EN | H/L | GREEN | STOP |
| Read/write | R/W | R/W | R/W | R/W |

STOP : STOP mode enable (0: inactive; 1: active)

GREEN : GREEN mode enable (0: inactive; 1: active)

H/L : CPU clock select

=1 : select high speed clock (OSCH) , CPU enters normal mode

=0 : select low speed clock (OSCL) , CPU enters low power mode

LVR_EN : (ROM-ready type) LVR enable

=0 : disable LVR

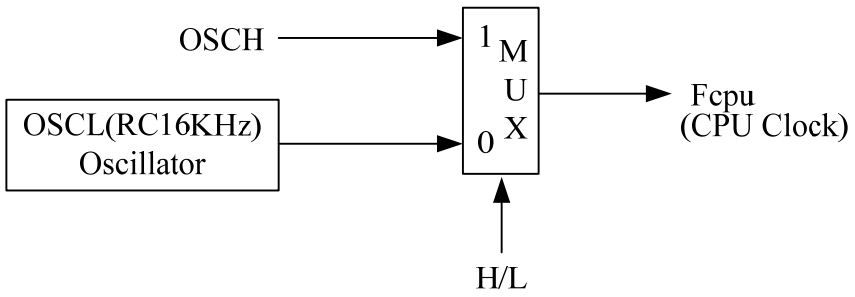
=1 : enable LVR

## ◆ CPU clock (Normal mode & low power mode)

The CPU clock comes from OSCH or OSCL clocks which are controlled by H/L bit in PS register. When the MCU sets H/L bit = "1", The CPU enters normal mode and operates in the high speed clock (OSCH). In this mode, the OSCL is also active.

When the MCU sets H/L bit = "0", The CPU enters low power mode and operates in the low speed clock (OSCL). In this mode, the OSCH will be turn off. The high speed oscillators are shut down to save the power consumption. The others circuits of the MCU which use the OSCH clock are also idle to reduce the power consumption. The CPU also enters low power mode after power-up and reset.

**Figure 3: CPU clock circuit**



## ◆ Power saving modes (STOP mode & GREEN mode )

The MCU enters STOP or GREEN mode by writing STOP bit or GREEN bit in PS register. During these modes, MCU holds some of the internal circuits to save the power consumption.

When the MCU sets STOP bit = "1", the MCU enter the STOP mode. Both of the clocks, OSCH and OSCL, are turn off. The MCU can be released from STOP mode by reset, interrupt, or wake-up event. When the system is waked up, it need a warm-up time for the stability of system clock running.

When the MCU sets GREEN bit = "1", the MCU enter the GREEN mode. In GREEN mode, the OSCH clock will be stopped, and RC16Khz oscillator (OSCL) is still on. The system can wake up from auto time-out (TB1) and ADC change (in ADC check procedure) wake-up for the alarm system application.

✧ Power saving modes condition & Release

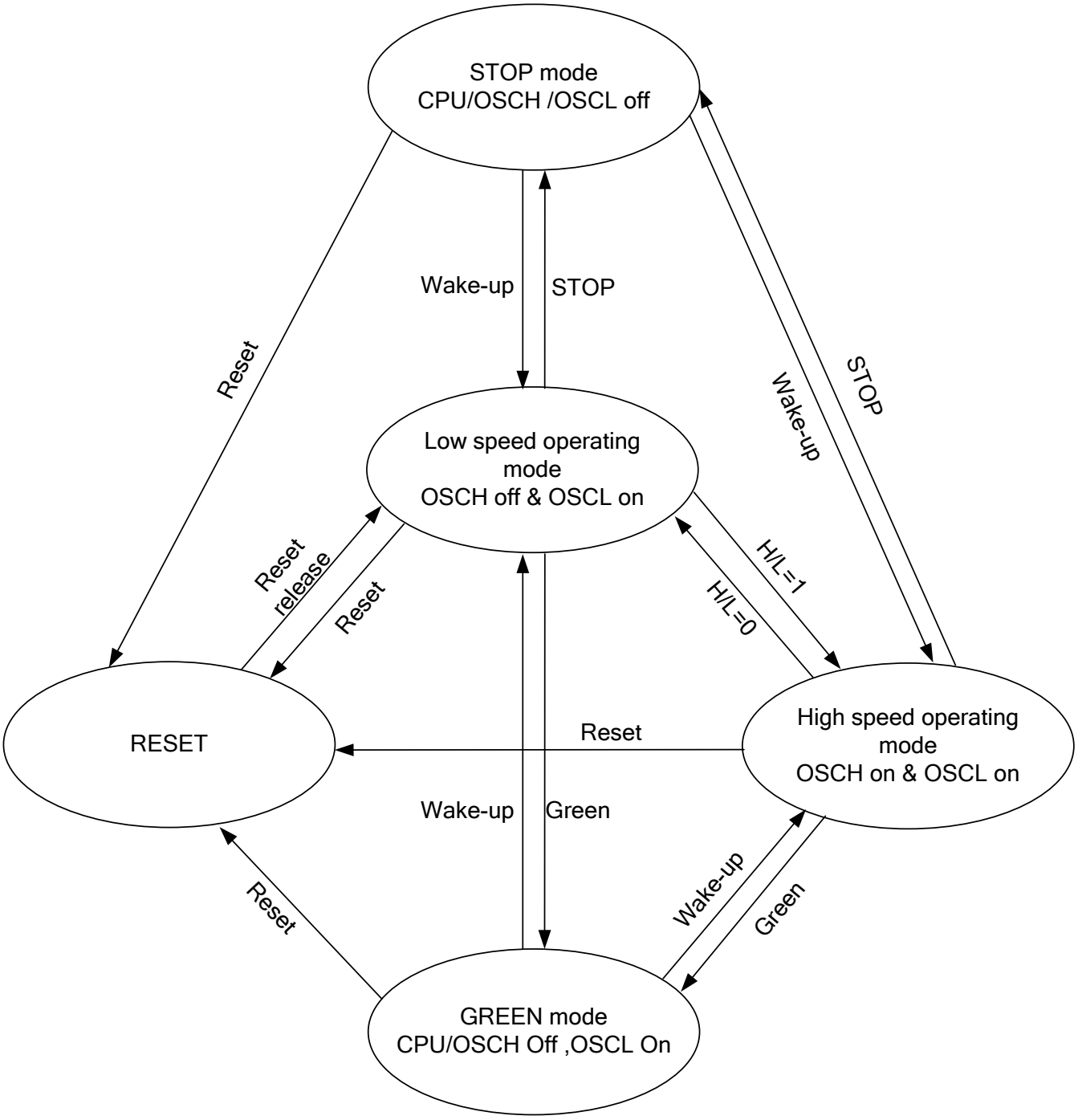
| Modes | STOP mode | GREEN mode |
|---|---|---|
| **High speed oscillator** | **Stopped** | **Stopped** |
| **Low speed oscillator** | **Stopped** | **Keep Operating** |
| **CPU clock** | **Stopped** | **Stopped** |
| **CPU internal status** | Stop & Retain the status | |
| **Memory, Flag, Register, I/O** | Retain the status | |
| **Program counter** | Hold the next executed address | |
| **Peripherals: Time base, Timers, Interrupts** | Stopped & Retain | Keep Operating |
| **Watch Dog Timer** | Disabled & cleared | |
| **Release Condition** | Reset ,<br>external INT,<br>Input change wake-up | Reset ,<br>external INT,<br>internal INT,<br>Input change wake-up,<br>TB1 time-out wake-up,<br>ADC change wake-up |

## ◆ MCU System Operation Modes

The MCU has the 4 operation modes, including normal mode, low speed mode , STOP mode , and GREEN mode . After power on reset, the MCU will go into low speed mode

automatically and the CPU operate in the low speed clock (OSCL). After wake up from STOP/GREEN mode, the MCU will resume the last operation mode.

**Figure 4: state diagram of system operation modes**



## ◆ OSCILLATOR STABLE TIME (OST)

The system oscillator generates the system control timing for CPU core or peripheral devices with fixed control phase, so the waveform of oscillator becomes sensitive to many kinds of noise. The abnormal duty is especially fatal for CPU. Any switching of clock source needs oscillation stable time (OST) to make sure the oscillation is stable and synchronized with CPU timing phase.

The OST is 256's RC16K clocks after power up. When waking up to normal mode, The OST is 4's OSCH clocks. When waking up to low power mode, the OST is 4's RC16K (OSCL) clock.

## ◆ Adjustment of OSCL (RC16KHz)

The OSCL is RC type oscillator. The resistance and capacitance of the built-in resistor and capacitor will vary piece by piece at the wafers. Then the frequency of OSCL will vary as the resistance and capacitance vary. When the chip is used in the hot or cold environment, the frequency of OSCL will change as the temperature of environment. It will also change as the power voltage of the chip is change. There is above 100% variation of the OSCL frequency in the worst case.

This chip provides a way to correct the OSCL frequency. If on-chip 910KHz is used as OSCH, it provides a 3% accurate oscillator to MCU. The user software can compare OSCL with OSCH by the timer/counter B (TCPB) and C (TCPC). If the OSCL is slower than 16KHz, decreasing Mod_RC16KH or Mod_RC16KL will speed up 16KHz. If the OSCL is faster than 16KHz, increasing Mod_RC16KH or Mod_RC16KL speed up 16KHz. Every 1 increment will slow down about 1.5% of frequency at RC16KHz. Here is a simple method to adjust the OSCL frequency:

Step 1: set TCPC condition – TCPCD = $FF_H$

Step 2: set TCPB conditions – TCPBDH = $FF_H$, TCPBDL = $FF_H$

Step 3: enable TCPC at conditions – Input clock = RC16KHz, auto-reload

Step 4: enable TCPB at conditions – Input clock = Fcpu, auto-reload

Step 5: count 6 overflows from TCPB, and record how many of TCPCOV occur

Step 6: Compute the total count of RC16KHz – Read TCPC counter value and do 1's complement, and add 255 for every TCPCOV

Step 7: disable TCPB and TCPC

Step 8: if the total count < 54 , subtract 1 to M16K7~0 and go to Step 3

Step 9: if the total count > 55 , add 1 from M16K7~0 and go to Step 3

Step 10: wait about 10 min. then go to Step 1 to do adjustment again

✧ Mod_RC16KL : Low nibble of RC16KHz modify register [R/W] : $067_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | M16K3 | M16K2 | M16K1 | M16K0 |
| Read/write | R/W | R/W | R/W | R/W |

M16K3~0 : low nibble to modify 16KHz frequency

✧ Mod_RC16KH : High nibble of RC16KHz modify register [R/W] : $068_H$ , default value [1000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | M16K7 | M16K6 | M16K5 | M16K4 |
| Read/write | R/W | R/W | R/W | R/W |

M16K7~4 : high nibble to modify 16KHz frequency

# § Interrupts

The CPU provides only 1 interrupt vector ($001H) for all interrupts. All of them have the same priority in the hardware. The first coming interrupt will be served firstly. When above two interrupts are coming, the software can decide the priority to serve them.

Interrupt sources include three timer/counter overflow interrupts (TB1 INT, TCPB INT, and TCPC INT), ADC conversion complete interrupt, and four external interrupts ( INT0 INT from AD0/PA0/INT0 pin , INT1 INT from AD1/PA1/INT1 pin , INT2 INT from PC0/VPP/INT2 pin, and INT3 INT from PC1/INT3 pin) .

The interrupt enable registers (INTCx) contain the interrupt enable control bits to enable and disable corresponding interrupt request and the corresponding interrupt request flags in the INTFx registers. Before finishing the INT service routine, another INT request will keep waiting until program return from interrupt routine.

✧ INTC1 : Internal interrupt enable register [R/W] : $009_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | ADC _IE | TCPC_IE | TCPB_IE | TB1_IE |
| Read/Write | R/W | R/W | R/W | R/W |

TB1_IE: Enable interrupt of time base timer 1. (0: disable; 1: enable)

TCPB_IE: Enable interrupt of timer/counter B. (0: disable; 1: enable)

TCPC_IE: Enable interrupt of timer/counter C. (0: disable; 1: enable)

ADC_IE : Enable interrupt of the ADC conversion. (0: disable; 1: enable)

✧ INTC2 : External interrupt enable register [R/W] : $018_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | INT3_IE | INT2_IE | INT1_IE | INT0_IE |
| Read/Write | R/W | R/W | R/W | R/W |

INT0_IE: Enable the external AD0/PA0/INT0 interrupt (0: disable; 1: enable)

INT1_IE: Enable the external AD1/PA1/INT1 interrupt (0: disable; 1: enable)

INT2_IE: Enable the external PC0/VPP/INT2 interrupt (0: disable; 1: enable)

INT3_IE: Enable the external PC1/INT3 interrupt (0: disable; 1: enable)

✧ INTF1 : Internal interrupt request flag register [R/W0] : $00A_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | ADC_IF | TCPC_IF | TCPB_IF | TB1_IF |

| Read/Write | R/W0 | R/W0 | R/W0 | R/W0 |
|---|---|---|---|---|

TB1_IF: TIME base Timer1 interrupt request flag for overflow. (0: inactive; 1: active)
TCPB_IF: Timer/counter B interrupt request flag for overflow. (0: inactive; 1: active)
TCPC_IF: Timer/counter C interrupt request flag for overflow. (0: inactive; 1: active)
ADC_IF: ADC interrupt request flag for conversion complete. (0: inactive; 1: active)

✧ INTF2 : External interrupt request flag register [R/W0] : $019_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | INT3_IF | INT2_IF | INT1_IF | INT0_IF |
| Read/Write | R/W0 | R/W0 | R/W0 | R/W0 |

INT0_IF: INT0 interrupt request flag for change. (0: inactive; 1: active)
INT1_IF: INT1 interrupt request flag for change. (0: inactive; 1: active)
INT2_IF: INT2 interrupt request flag for change. (0: inactive; 1: active)
INT3_IF: INT3 interrupt request flag for change. (0: inactive; 1: active)

If the interrupt request needs service, the user software may set the corresponding interrupt enable bit. The internal timer/counter interrupt flag (TCPx_IF) is set to 1, resulting from the timer/counter overflow.

When the corresponding interrupts are enabled and their flag bits are set to 1, the CPU will active the interrupt service routine. Then CPU reads the service flag bits and check the request priority then proceeds with the relative interrupt service. After CPU writes the corresponding bits to 0 in the INTF register, the INT flag bit will be cleared to 0 (using STX #n, $m instruction). The CPU can only write "0" to clear the flag bit. Writing "1" to flag bit will do nothing to it.

# § Watch Dog Timer (WDT)

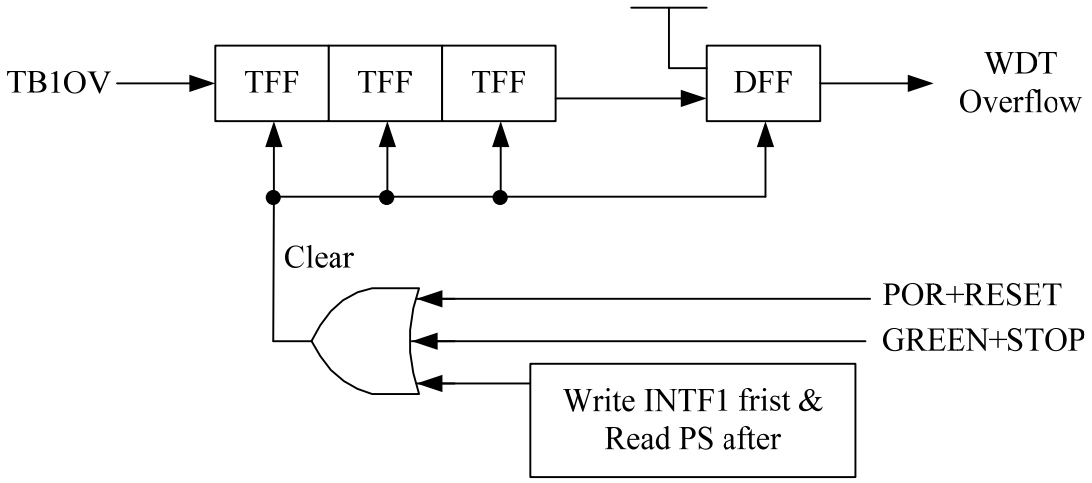
The clock of watch dog timer comes from time base overflow (TB1OV). The time-up signal of WDT can prevent a software malfunction or abnormal sequence from jumping to an unknown memory location, before the system is going to fatal failure. Normally, when the time-up signal of WDT is active, it will reset the chip. At the same time, program and hardware can be initialized to resume system back to normal operation. The chip needs 2 steps to clear WDT. The programmer writes INTF1 with $F data first to enable the WDT clear, and then reads the power saving (PS) control register after. After finishing the two write/read steps in order, the watch dog timer will be cleared. User should well arrange the two command steps for avoiding the dead lock loop. *User should keep in minds that always reset WDT at main program and*

*never clear the WDT in the interrupt routine.*

**The max period of WDT = (TB1OV cycle time) * 8**

In GREEN mode, The time base timer 1 is combined with timer/counter TCPB to generate a long period timer for auto wake-up function. The WDT is cleared in this mode. The TB1OV is also ignored by the WDT. In STOP mode, all the oscillators are turned off. The WDT is also cleared.

**Figure 5: WDT circuit**



# § RESET

The chip has three kinds of reset sources: POR (power on reset), Watch dog timer reset, LVR (low voltage reset). The reset feature can be divided into 2 kinds of groups. One is system reset and the other is CPU reset. The system reset will initialize the CPU and peripheral device with default state. The CPU reset only initializes the CPU state and keeps the peripheral state no change.

## ◆ System reset

**.POR (power on reset)**

The chip provides automatic reset function when the power is turned on. The reset will be active by POR when the VDD is below 1.4V and its rising slope (from 0.1VDD up to o.9VDD) is less than 10ms.

## ◆ CPU reset

**.Watch Dog Timer Reset**

The reset will happen automatically when the watchdog timer runs overflow. If the watchdog timer is cleared regularly by the user software before overflow, no watchdog reset will occur. While the CPU is forced into abnormal state, the regular software procedure is disrupted and the watchdog timer may overflow by no clear coming. The overflow of WDT will generate reset to return CPU to the reset procedure. Thus the CPU goes back to normal operation. The H/L and GREEN and STOP bits will also be cleared by WDT.

**. LVR (low voltage reset)**

The LVR is a low system voltage detector. As the operating voltage falls below the detected window, then the LVR will make the CPU start the reset procedure. The LVR can be disabled by setting the LVREN bit of PS register to "0". When the low system voltage occurs in this condition, The CPU will cease to do any instruction. But the system states have opportunity to be corrupted by this low system voltage. The following table shows the relationship between LVR and detected voltage. The H/L and GREEN and STOP bits will also be cleared by LVR.

| LVR | System RESET | Detected Vop |
|---|---|---|
| 0→1 | active | 1.2V~1.8V |

| | | |
|---|---|---|
| 1→0 | inactive | >1.8V |

For OTP MCU, hardware needs to download the mask options into registers. It happens at power on reset, watch dog overflow reset and LVR reset.

# § System Register Description

$000(DP1 ) : Index == 虛擬指標暫存器

這不是一個實際的暫存器,只用來當做間接定址的指標,當對此位址讀寫或運算或讀取ROM TABLE 時,CPU 實際是以DPH,DPM,DPL 所組成的 12 位元為地址線,讀取ROM 的資料或是對RAM 做運算或讀寫資料.

$001 : Acc == 累積器

除了當成一般暫存器使用外;當需要將暫存器或RAM 的資料轉換到其他RAM 或暫存器時累積器是一個重要的中繼站,先使用LDA 指令讀取暫存器或RAM 的資料暫存在累積器,再用STX 指令存到想要存放的暫存器或RAM 中,來達成暫存器或RAM 的資料轉換;此外在做邏輯或算數運算時,若是選擇結果放到Acc,則運算完的結果將會存到累積器中;在使用直接指令模式運算,累積器是二個運算來源之一;若是使用RTB 指令讀取ROM TABLE 時,讀取到的資料是 16 位元,其中最低的 4 位元(D3~D0)也會暫存在累積器中

$002 : TB1 == 讀表暫存器 1

此暫存器提供我們在使用RTB 指令讀取ROM TABLE 時,暫時存放讀取 16 位元資料中的次低 4 位元(D7~D4)用;在不使用讀表功能時,此暫存器可當作一般暫存器使用.

$003 : TB2 == 讀表暫存器 2

與TB1 同樣功能,此暫存器提供我們在使用RTB 指令讀取ROM TABLE 時,暫時存放讀取 16 位元資料中的次高 4 位元(D11~D8)用;在不使用讀表功能時,此暫存器可當作一般暫存器使用.

$004 : TB3 == 讀表暫存器 3

同樣的此暫存器提供我們在使用RTB 指令讀取ROM TABLE 時,暫時存放讀取 16 位元資料中的最高 4 位元(D15~D12)用;在不使用讀表功能時,此暫存器可當作一般暫存器使用.

$005：DPL == 低位指標暫存器,

使用間接方式對RAM 位址讀寫或運算時,此暫存器是指標(DP)的 10 條地址線中最低的 4 位元(A3~A0);同時亦是使用RTB 指令讀取ROM TABLE 資料的 12 位元地址線中最低的 4 位元(A3~A0),所以運用在上述二種方式使用之前必須先設定此暫存器.若是與上述的使用不衝突時可當作一般暫存器使用.

$006：DPM == 中位指標暫存器,

與DPL 功能相同,在使用間接方式對RAM 位址讀寫或運算時,此暫存器是指標(DP)的 10 條地址線之中間的 4 位元(A7~A4);同時亦是使用RTB 指令讀取ROM TABLE 資料的 12 位元地址線之中間的 4 位元(A7~A4),所以運用在上述二種方式使用之前必須先設定此暫存器.若是與上述的使用不衝突時可當作一般暫存器使用.

$007：DPH == 高位指標暫存器,

與上述二暫存器功能相同,在使用間接方式對RAM 位址讀寫或運算時,此暫存器是指標(DP)的 10 條地址線中最高的 2 位元(A9~A8),A10 及A11 會被忽略掉一般設定為"0";同時亦是使用RTB 指令讀取ROM TABLE 資料的 12 位元地址線中最高的 4 位元(A11~A8),所以運用在上述二種方式使用之前必須先設定此暫存器.若是與上述的使用不衝突時可當作一般暫存器使用.

# § Wake up

After the CPU wakes up from STOP or GREEN mode, it can know which source makes wake-up occur from the wake-up flag. Then the user software may choose the proper routine for the dedicated wake-up source.

There are 3 sources which make the CPU wake-up from STOP mode. They include : power on reset, external INT, and input change. Except power on reset which initializes the whole system, the others make the CPU execute the instruction after the last one which is executed before STOP mode occurs.

There are 6 sources which make the CPU wake-up from GREEN mode. They are : power on reset, external INT, internal INT, input change, auto time-out (TB1), and ADC change (in ADC check procedure). Except power on reset which initializes the whole system, the others make the CPU execute the instruction after the last one which is executed before GREEN mode occurs.

Note: The instruction that follow the STOP or GREEN mode (set by PS register) must be NOP, and two NOP instructions must be followed, it make sure to wake up success.

## ◆ Port wake up enable

The Port Wake-up enable register can enable or disable the wake-up from the input change of the ports. There are 3 port wake-up enable bits for 3 ports each. All pins in the same port are controlled by the same wake up enable bit.

✧ PT_WKEN : Port wake up enable register [R/W] : $066_H$, default value [-000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | -- | PC_WKEN | PB_WKEN | PA_WKEN |
| Read/Write | R0 | R/W | R/W | R/W |

PA_WKEN : control PORT A (pa0~3) input change wake up when those are input pins.
=0 : disable wake-up
=1 : enable wake-up

PB_WKEN : control PORT B (pb0~3) input change wake up when those are input pins.
=0 : disable wake-up
=1 : enable wake-up

PC_WKEN : control PORT C (pc0~3) input change wake up when those are input pins.
=0 : disable wake-up
=1 : enable wake-up

## ◆ Wake up flag

The Wake-up flag register records which wake-up sources take place. When waked up, the user software should read this register to choose the corresponding service routine for each flag bit. The corresponding wake-up flag bit must be cleared before entering STOP or GREEN mode. After CPU writes the corresponding bit to "0" in the register, the wake-up flag bit will be cleared to 0 (using STX #n, $m instruction). The user software can only write "0" to clear the wake-up flag bit. Writing "1" to the flag bit will do nothing to it.

✧ WK_FG : wake-up flag register [R/W0] : $012_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TB_WK | PC_WK | PB_WK | AD/PA_WK |
| Read/Write | R/W0 | R/W0 | R/W0 | R/W0 |

AD/PA_WK: The wake-up flag for ADC max/min range check fail in GREEN mode, and input change of any input pins of PA (Pa0~3) (0: inactive; 1: active)

PB_WK: The wake-up flag for input change of any input pins of PB (Pb0~3) (0: inactive; 1: active)

PC_WK: The wake-up flag for input change of any input pins of PC (Pc0~3) (0: inactive; 1: active)

TB_WK: The wake-up flag for overflow of time base timer 1 in GREEN mode (0: inactive; 1: active)

# § Peripheral function description:

## ◆ Control register

The control register includes timer/counter control bits and low voltage detect/control bits. The user software can turn on LVD and find out whether the power is below the low voltage or not in this register. There are four voltages selected in the mask option register. Because the LVD consume some operation current, so it will be automatically turned off when entering into STOP or GREEN mode. It takes 500us to be stable after enabling the LVD_EN bit. Do not read the LVD_DET bit before the stable of the LVD circuit.

✧ CTLA : LVD and time/counter control register [R/W] : $00B_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | LVD_DET | LVD_EN | TCPC_LD | TCP_RD |
| Read/Write | R | R/W | R/W | R/W |

TCP_RD : control reading from counter data register or pre-load data register of the timer/counter B & C.
=0 : read from counter data register
=1 : read from pre-load data register

TCPC_LD : enable auto-reload function of timer/counter C
=0 : disable auto-reload function
=1 : enable auto-reload function

LVD_EN : enable low voltage detect function
=0 : disable low voltage detect function
=1 : enable low voltage detect function

LVD_DET : show whether the power below low voltage or not
=0 : the power not in low voltage state
=1 : detect a low voltage happen

## ◆ Time Base Timer 1 and Buzzer control

The register includes the four selections of the overflow time in time base timer 1, and buzzer output enable.

The clock of time base timer 1 is from OSCL (16KHz) normally. But in GREEN mode, it gets the clock from the overflow of time/counter B, and its overflow wakes up CPU. But the

overflow will not generate a interrupt request, even when the TB1_IE bit is set in the INTC1 register. Also the interrupt flag bit, ADC_IF, is leave no change in GREEN mode. The combination of timer/counter B and time base timer1 let the MCU stay a long period time in the GREEN mode.

✧ TBC/BZC : Time base timer/buzzer control register [R/W] : $00C_H$ , default value [0-11]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | BZ_EN | -- | TB1 | TB0 |
| Read/Write | RW | R0 | R/W | R/W |

TB1, TB0：select overflow count of time base timer 1 in GEEN mode, or overflow time of time base timer 1 as the following table:

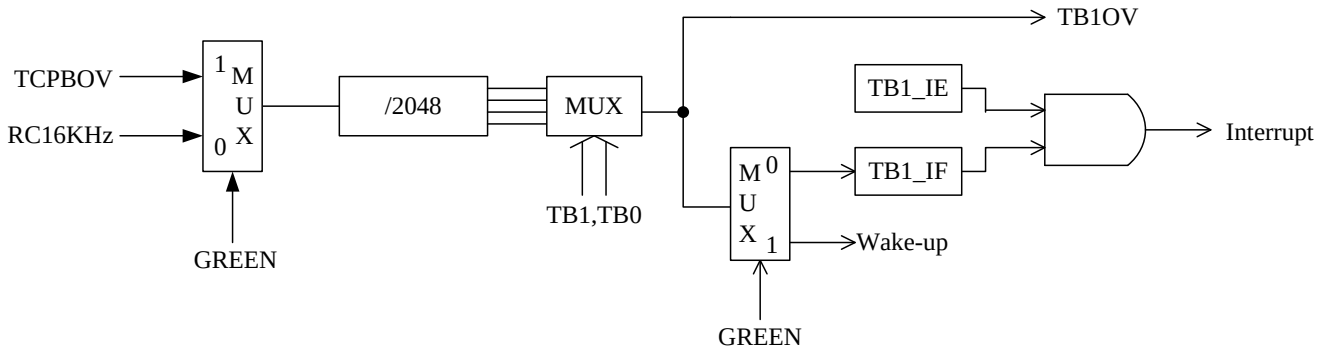
| TB1 | TB0 | GREEN mode | Other modes | |
|---|---|---|---|---|
| | | Overflow count | Overflow time | Overflow frequency |
| 0 | 0 | 256 | 15.625 ms | 16kHz / 256 = 64 Hz |
| 0 | 1 | 512 | 31.25 ms | 16kHz / 512 = 32 Hz |
| 1 | 0 | 1024 | 62.5 ms | 16kHz / 1024 = 16 Hz |
| 1 | 1 | 2048 | 125 ms | 16kHz / 2048 = 8 Hz |

BZ_EN : enable the output pin of buzzer

=0 : PB3/BZ defined as PB3

=1 : PB3/BZ defined as BZ output pin

**Figure 6: Time base timer 1 circuit**




## ◆ 8 bits Timer/counter B for TCPB

The 8-bit timer/counter B with two pre-load data buffers provides four clock sources to be selected. The clock sources includes: Fcpu (CPU clock), PC2, OSCL (16Khz), and PFD clock from TCPC. When the TCPB_EN bit is "0", the pre-load data buffers, TCPBDHH and TCPBDHL, will keep being loaded into counter. Any value written to TCPBDHH and TCPBDHL is also

loaded to counter. After writing the TCPB_EN to "1", it depends on auto-load function to control how to reload to counter.

The overflow signal of TCPB is TCPBOV. The rising edge of TCPBOV will set the relative INT flag. If disabling auto-reload function by writing TCPB_LD to "0", the TCPBOV will be active after TCPB counts up from the 1's complement of TCPBDH to "$FF_H$". If enabling auto-reload function, the TCPBOV will be active until TCPB counts up both the values reloaded in order from the TCPBDH and TCPBDL.

In the condition that TCPBDH is written "$00_H$", Enabling TCPB will make a immediate counter overflow. It will trigger reload TCPBHL in the auto-reload mode. Or it will activate TCPBOV and count across to "$00_H$" when the auto-reload function is disabled.

After the BZ_EN bit is set to 1 in the TBC/BZC register, TCPB provides buzzer function by directing the TCPBOV signal waveform to the PB3 pin. No matter what the TCPB_LD bit is, the auto-reload function will be active.

When setting the TCP_RD bit to 0 in the CTLA register, Reading TCPBDH or TCPBDL get the same value from the counter of TCPB. Because the counter is loaded the 1's complement of TCPBDH or TCPBDL and counts up, the value read from the counter needs to be translated. If writing TCP_RD bit to 1, Reading TCPBDH (or TCPBDL) get the value from the pre-load data buffer of TCPBDH (or TCPBDL).

✧ TCPBC: Timer/counter B control register [R/W] : $013_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TCPB_LD | TCPB_CK1 | TCPB_CK0 | TCPB_EN |
| Read/Write | R/W | R/W | R/W | R/W |

TCPB_EN : control enable timer/counter B counting

=0 : disable TCPB

=1 : enable TCPB

TCPB_CK1, TCPB_CK0 : select clock sources for TCPB except GREEN mode as the following table:

| TCPB_CK1 | TCPB_CK0 | clock source |
|---|---|---|
| 0 | 0 | Fcpu (CPU clock) |
| 0 | 1 | PC2 |
| 1 | 0 | RC16KHz (OSCL) |
| 1 | 1 | PFD (from TCPC) |

TCPB_LD : control auto-reload function of timer/counter B

=0 : disable auto-reload function

=1 : enable auto-reload function

✧ TCPBDLL: Low nibble register for low data TCPBDL [R/W] : $014_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TCPBDL3 | TCPBDL2 | TCPBDL1 | TCPBDL0 |
| Read/Write | R/W | R/W | R/W | R/W |

TCPBDL3~TCPBDL0: low nibble for low data buffer TCPBDL.

✧ TCPBDLH: High nibble register for low data TCPBDL [R/W] : $015_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TCPBDL7 | TCPBDL6 | TCPBDL5 | TCPBDL4 |
| Read/Write | R/W | R/W | R/W | R/W |

TCPBDL7~TCPBDL4: high nibble for low data buffer TCPBDL.

✧ TCPBDHL: low nibble register for high data TCPBDH [R/W] : $016_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TCPBDH3 | TCPBDH2 | TCPBDH1 | TCPBDH0 |
| Read/Write | R/W | R/W | R/W | W |

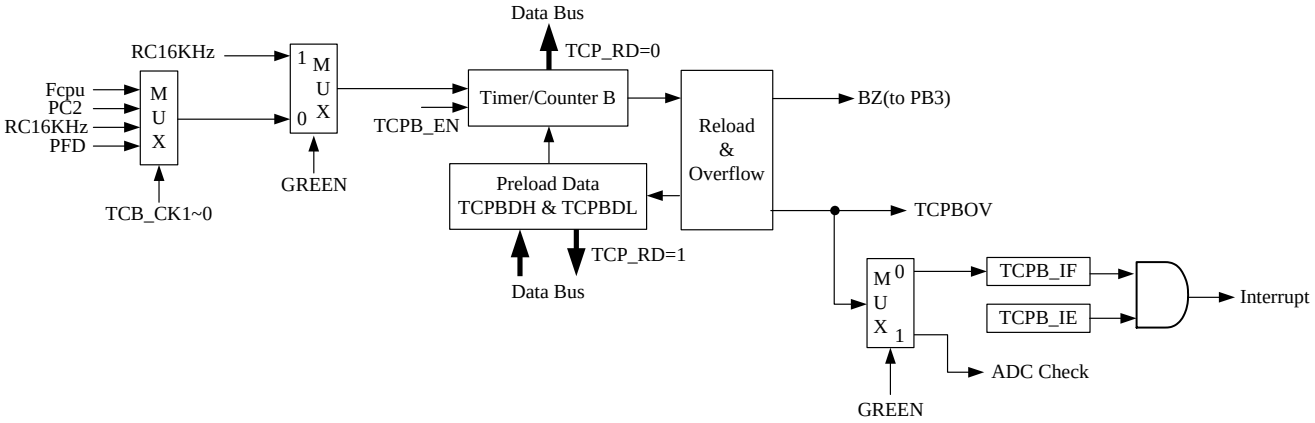
TCPBDH3~TCPBDH0: low nibble for high data buffer TCPBDH.

✧ TCPBDHH: high nibble register for high data TCPBDH [R/W] : $017_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TCPBDH7 | TCPBDH6 | TCPBDH5 | TCPBDH4 |
| Read/Write | R/W | R/W | R/W | R/W |

TCPBDH7~TCPBDH4: high nibble for high data buffer TCPBDH.

**Figure 7: Timer/counter B (TCPB) circuit**




## ➢ Timer

When TCPB works as a timer in auto-reload mode, the user software sets the two pre-load data buffers, TCPBDH (output high level) and TCPBDL (output low level), for periodic timing generation. After initialization, the user software starts TCPB to count by setting TCPB_EN to "1", the TCPB cycle period is:

**Tc = (selected clock cycle) * (TCPBDH+TCPBDL)**

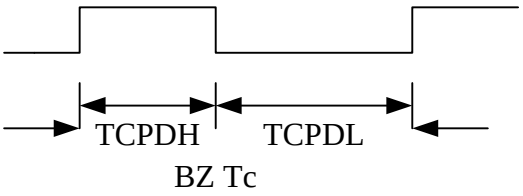
When writing data to the TCPBDH, the data will also be loaded to counter if TCPB_EN is "0". The 1's complement value of TCPBDH will be pre-loaded into counter TCPB as initial value. Writing TCPB_EN to "1" starts the count-up function. The timer reloads TCPBDL to counter as TCPB up-counts and reaches the value of "$FF_H$" or 255. After TCPBDL reloaded to counter, the next reload data is TCPBDH. The TCPBDH and TCPBDL will be re-loaded to counter alternately. When the TCPBDH is reloaded, the overflow TCPBOV and the interrupt request flag TCPB_IF will be active. If enabling the corresponding interrupt enable bit, the INT circuit will activate MCU interrupt service routine.

## ➢ BZ

The buzzer feature is active in TCPB, as the BZ_EN bit is set to 1 in the TBC/BZC register. The buzzer output the waveform to the PB3 pin. The auto-reload function will be turned on automatically no matter what the TCPB_LD bit is. Usually the user software will disable the interrupt feature for tone or melody generation. The BZ frequency is:

**BZ frequency = (selected clock frequency) / (TCPDH+TCPDL)**

**Figure 8: Buzzer period**




## ➢ TCPB and TB1 in GREEN mode

The combination of TCPB and TB1 provides many features in GREEN mode. In this mode, the input clock of TCPB is fixed to OSCL (16KHz), and runs in auto-reload mode. After automatically counting up and reloading TCPBDH/TCPBDL in order, the overflow TCPBOV is active to generate one ADC check procedure which measure ADC at one selected channel and check the conversion result with the High/Low limit by hardware circuit. The four channels AD0~3 will be selected in order at each overflow. Before the ADC check procedure for some channel going, it will confirm The ADxWKEN bit of the channel is set to "1" in the AD_WKEN register. If the user software doesn't want to do this check procedure for some channel, it can set the corresponding ADxWKEN bit to "0", and leave nothing to do in this overflow. The period time for this ADC check procedure is:

**Tp_adcc = (period of 16KHz) * (TCPBDH+TCPBDL)**

No matter what the TCPB_IE bit is, the interrupt from the overflow TCPBOV will be disabled in GREEN mode, and its interrupt flag is also left alone. No matter what the ADC_IE bit is, the interrupt after ADC conversion complete will be disabled in the ADC check procedure, and its interrupt flag is also left no change. If the ADC conversion result is above the high limit ADx_H or below the low limit ADx_L in the ADC check procedure, the wake-up will be invoked. The user software can read the ADC wakeup bit of the WK_FG register to make sure.

The input clock of TB1 is fixed to TCPBOV in GREEN mode. The overflow TB1OV will set the Auto Time Out bit of the WK_FG register and invoke the CPU to wake up. No matter what the TB1_IE bit is, the interrupt from the overflow TB1OV will be disabled in GREEN mode, and its interrupt flag is also left no change. The period time for TB1OV is:

**Tp_tb1 = (Tp_adc) * (selected overflow count)**

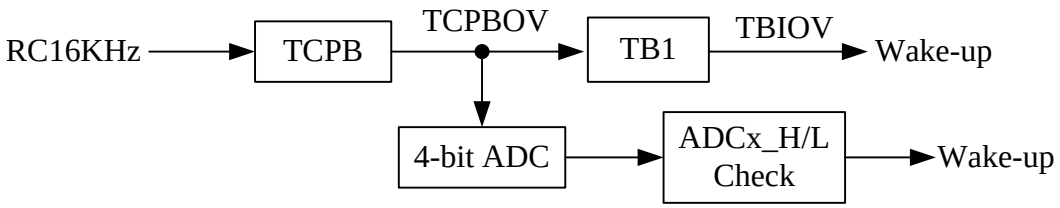
The user software can choose the period time to do the ADC check procedure. If its power consumption is the most concern, it can select the maximum period time of TCPBOV:

**Max. Tp_adcc = (62.5 us) * (256+256) ~= 32 ms**

If the longest GREEM mode is also wanted, the user software can choose the longest overflow time of TB1. The maximum period time is:

**Max. Tp_tb1 = (31.875 ms) * (2048) ~= 65.28 s**

**Figure 9: Combination circuits in GREEN mode**




## ◆ 8 bits Timer/counter C for TCPC

The 8-bits timer/counter C with pre-load data buffer provides five clock sources. The clock sources includes: OSCL (16Khz), PC0, PC1, Fcpu (CPU clock), and BZ clock from TCPB. The TCPCOV is the overflow signal of TCPC, and its rising edge will set the relative INT flag.

When the TCPC_EN bit in the TCPCC register is "0", the pre-load data buffers, TCPCDH and TCPCDL, will keep being loaded into counter. Any value written to TCPCDH and TCPCDL is also loaded to counter. After setting the TCPC_EN bit to "1", the counter starts to increase according to the clock source. The counter will be reloaded from TCPCDH and TCPCDL until overflow occurs, if enabling the auto-load function by writing the TCPC_LD bit to "1" in the CTLA register.

The overflow signal of TCPC is TCPCOV. The rising edge of TCPCOV will set the relative INT flag. The TCPCOV will be active after TCPC counts up from the 1's complement of TCPCD to "$FF_H$". In the condition that TCPCD is written "$00_H$", Enabling TCPC will make a immediate counter overflow. It will activate TCPCOV, and count across to "$00_H$".

When setting the TCP_RD bit to "0" in the CTLA register, Reading TCPCDL and TCPCDH will get the values from the counter of TCPC. Because the counter is loaded the 1's complement of TCPCD and counts up, the value read from the counter needs to be translated. If setting TCP_RD bit to 1, Reading TCPCDL and TCPCDH get the values from the pre-load data buffer of TCPC.

✧ TCPCC: Timer/counter C control register [R/W] : $01B_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TCPC_CK2 | TCPC_CK1 | TCPC_CK0 | TCPC_EN |

| Read/Write | R/W | R/W | R/W | R/W |
|---|---|---|---|---|

TCPC_EN : control enable timer/counter C counting

=0 : disable TCPC

=1 : enable TCPC

TCPC_CK2, TCPC_CK1, & TCPC_CK0 : select clock sources for TCPC as the following table:

| TCPC_CK2 | TCPC_CK1 | TCPC_CK0 | clock source |
|---|---|---|---|
| 0 | 0 | 0 | RC16KHz (OSCL) |
| 0 | 0 | 1 | PC0 |
| 0 | 1 | 0 | PC1 |
| 0 | 1 | 1 | Fcpu (CPU clock) |
| 1 | x | x | BZ (from TCPB) |

✧ TCPCDL: Low nibble register for data TCPCD [R/W] : $01C_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TCPCD3 | TCPCD2 | TCPCD1 | TCPCD0 |
| Read/Write | R/W | R/W | R/W | R/W |

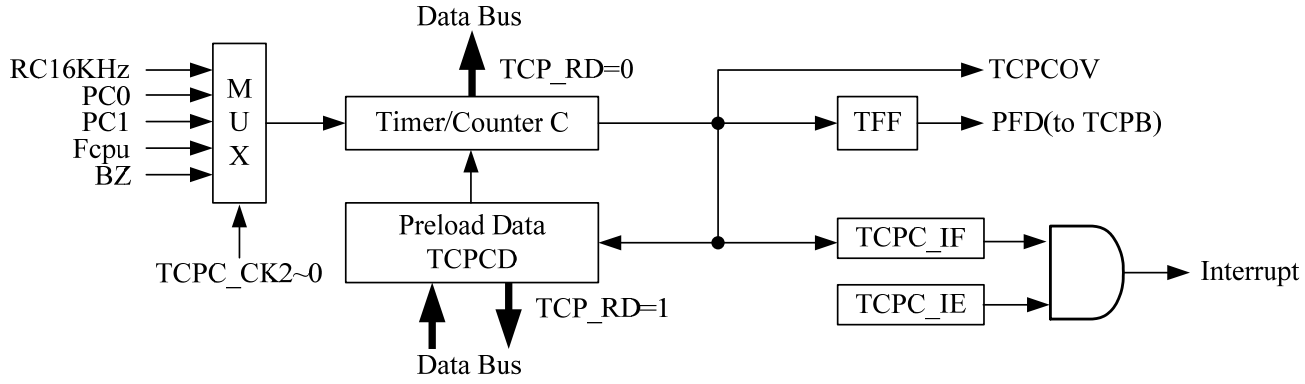
TCPCD3~TCPCD0: low nibble for data buffer TCPCD.

✧ TCPCDH: High nibble register for data TCPCD [R/W] : $01D_H$ , default value [0000]

| Register | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|
| Bit Name | TCPCD7 | TCPCD6 | TCPCD5 | TCPCD4 |
| Read/Write | R/W | R/W | R/W | R/W |

TCPD7~TCPD4: high nibble for data buffer TCPCD.

**Figure 10: Timer/counter C (TCPC) circuit**



## ➢ Timer

When TCPC works as a timer in auto-reload mode, the user software sets the pre-load data buffer TCPCD for periodic timing generation. After initialization, the user software starts TCPC to count by setting TCPC_EN to "1", the TCPC cycle period is:

**Tc = (selected clock cycle) * (TCPCD)**

When writing data to the TCPCD, the data will also be loaded to counter if TCPC_EN is "0". The 1's complement value of TCPCD will be pre-loaded into counter TCPC as initial value. Writing TCPC_EN to "1" starts the count-up function. If in auto-reload mode, the timer reloads TCPCD to counter as TCPC up-counts and reaches the value of "$FF_H$" or 255. At the same time, the overflow TCPCOV and the interrupt request flag TCPC_IF will be active. If enabling the corresponding interrupt enable bit, the INT circuit will activate MCU interrupt service routine.

## ➢ PFD

The PFD signal comes from which divides the overflow TCPCOV by two. The TCPB can select PFD as the clock source. This provides a longer period of timer by combining TCPB and TCPC.

## ◆ PORT A , B , C

There are three ports can be used. All the pins of the ports are I/O pins. The pins of port A also are combined with the analog input pins of ADC. Writing the I/O definition registers (PAC, PBC, and PCC) can change them to be input or output pins.

In output mode, writing the data register (PA, PB, and PC) sets them to what values they output, and reading the data registers get what values are stored from register. In input mode, reading the data registers get what values are inputted from pins.

Writing the pull-high registers (PB_PH and PC_PH) turn-on or turn-off the pull-high of the input pins. The following table collects all the registers about these ports.

| Address | Register | Bit3 | Bit2 | Bit1 | Bit0 | Default |
|---|---|---|---|---|---|---|
| $00F_H$ | PA | PA3 | PA2 | PA1 | PA0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $010_H$ | PB | PB3 | PB2 | PB1 | PB0 | uuuu |
| | | R/W | R/W | R/W | R/W | |

| | | | | | | |
|---|---|---|---|---|---|---|
| $011_H$ | PC | PC3 | PC2 | PC1 | PC0 | uuuu |
| | | R/W | R/W | R/W | R | |
| $00D_H$ | PAC | PAC3 | PAC2 | PAC1 | PAC0 | 1111 |
| | | R/W | R/W | R/W | R/W | |
| $01E_H$ | PBC | PBC3 | PBC2 | PBC1 | PBC0 | 1111 |
| | | R/W | R/W | R/W | R/W | |
| $01F_H$ | PCC | PCC3 | PCC2 | PCC1 | PCC0 | 1111 |
| | | R/W | R/W | R/W | R | |
| $064_H$ | PB_PH | PB3_PH | PB2_PH | PB1_PH | PB0_PH | 0000 |
| | | R/W | R/W | R/W | R/W | |
| $065_H$ | PC_PH | PC3_PH | PC2_PH | PC1_PH | PC0_PH | 0000 |
| | | R/W | R/W | R/W | R/W | |

- ✧ PA [$00F_H$] : Port A DATA register
- ✧ PAC[$00D_H$] : Port A I/O definition register
  - =1 : setting it to be input pin.
  - =0 : setting it to be output pin
- ✧ PB [$010_{H]}$ : Port B DATA register
- ✧ PBC [$01E_H$] : Port B I/O definition register
  - =1 : setting it to be input pin
  - =0 : setting it to be output pin
- ✧ PB_PH [$064_H$] : Port B Pull_high control
  - =1 : enable pull_high resister when it is input pin
  - =0 : disable pull_high resister
- ✧ PC [$011_H$] : Port C DATA register
- ✧ PCC [$01F_H$] : Port C I/O definition register
  - =1 : setting to be input pin.
  - =0 : setting to be output pin
- ✧ PC_PH [$065_H$] : Port C Pull_high enable
  - =1 : enable pull_high resister when it is input pin
  - =0 : disable pull_high resister

## ◆ 4-bit ADC

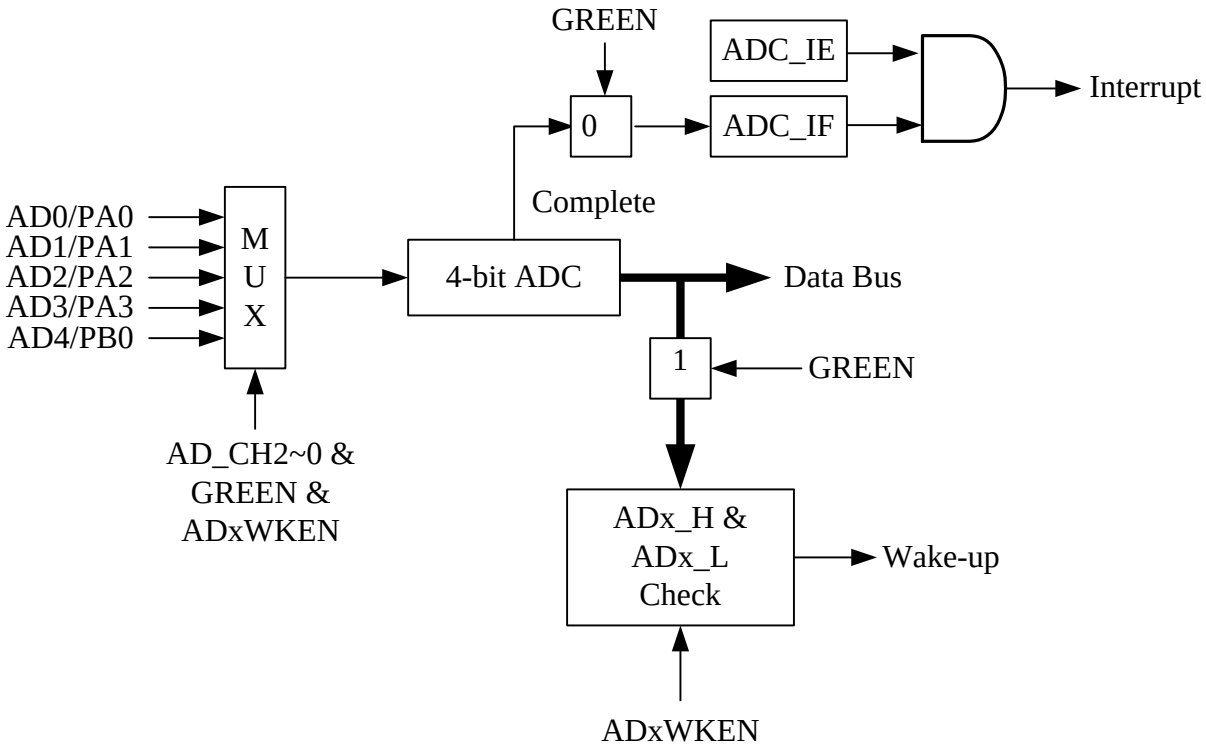
The following table collects all the registers about 4-bit ADC.

| Address | Register | Bit3 | Bit2 | Bit1 | Bit0 | Default |
|---|---|---|---|---|---|---|
| $01A_H$ | ADC_CTL1 | ADEN | AD_CH2 | AD_CH1 | AD_CH0 | 0000 |
| | | R/W | R/W | R/W | R/W | |
| $058_H$ | AD0_H | AD0_TOP3 | AD0_TOP2 | AD0_TOP1 | AD0_TOP0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $059_H$ | AD0_L | AD0_BOT3 | AD0_BOT2 | AD0_BOT1 | AD0_BOT0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $05A_H$ | AD1_H | AD1_TOP3 | AD1_TOP2 | AD1_TOP1 | AD1_TOP0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $05B_H$ | AD1_L | AD1_BOT3 | AD1_BOT2 | AD1_BOT1 | AD1_BOT0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $05C_H$ | AD2_H | AD2_TOP3 | AD2_TOP2 | AD2_TOP1 | AD2_TOP0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $05D_H$ | AD2_L | AD2_BOT3 | AD2_BOT2 | AD2_BOT1 | AD2_BOT0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $05E_H$ | AD3_H | AD3_TOP3 | AD3_TOP2 | AD3_TOP1 | AD3_TOP0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $05F_H$ | AD3_L | AD3_BOT3 | AD3_BOT2 | AD3_BOT1 | AD3_BOT0 | uuuu |
| | | R/W | R/W | R/W | R/W | |
| $060_H$ | ADC_D | AD_D3 | AD_D2 | AD_D1 | AD_0 | uuuu |
| | | R | R | R | R | |
| $061_H$ | ADC_PA | AD3/PA3 | AD2/PA2 | AD1/PA1 | AD0/PA0 | 0000 |
| | | R/W | R/W | R/W | R/W | |
| $062_H$ | AD_WKEN | AD3WKEN | AD2WKEN | AD1WKEN | AD0WKEN | 0000 |
| | | R/W | R/W | R/W | R/W | |
| $063_H$ | ADC_CTL2 | ADCK1 | ADCK0 | -- | AD4/PB0 | 00-0 |
| | | R/W | R/W | R0 | R/W | |
| $009_H$ | INTC1 | ADC _IE | TCPC_IE | TCPB_IE | TCPA_IE | 0000 |
| | | R/W | R/W | R/W | R/W | |
| $00A_H$ | INTF1 | ADC_IF | TCPC_IF | TCPB_IF | TCPA_IF | 0000 |
| | | R/W0 | R/W0 | R/W0 | R/W0 | |

| Bit Name | FUNCTION |
|---|---|
| ADC_IE | =0：Disable interrupt of the A/D conversion<br>=1：Enable interrupt of the A/D conversion |
| ADC_IF | =0：No complete of A/D conversion<br>=1：A/D conversion complete flag.<br>After starting the A/D conversion, the user software can read this flag to see whether the conversion completes or not. This flag can only be cleared to "0" by writing "0". Writing "1" to flag bit will do nothing to it. |
| ADEN | =0：Disable A/D<br>=1：Enable A/D. |
| AD_CH2,<br>AD_CH1,<br>AD_CH0 | Select A/D input channel as:<br>000 : select AD0<br>001 : select AD1<br>010 : select AD2<br>011 : select AD3<br>100 : select AD4 |
| ADCK1,<br>ADCK0 | Select ADC clock as:<br>00：select OSCH clock,<br>01：select OSCH/4 clock,<br>10：select OSCH/16 clock,<br>11：select OSCL clock. |
| AD0/PA0 | =0 : pad option is I/O PORT PA0<br>=1 : pad option is AD0 channel |
| AD1/PA1 | =0 : pad option is I/O PORT PA1<br>=1 : pad option is AD1 channel |
| AD2/PA2 | =0 : pad option is I/O PORT PA2<br>=1 : pad option is AD2 channel |
| AD3/PA3 | =0 : pad option is I/O PORT PA3<br>=1 : pad option is AD3 channel |
| AD4/PB0 | =0 : pad option is I/O PORT PB0<br>=1 : pad option is AD4 channel |
| AD_DATA3~0 | "W" to trigger conversion start.<br>"R" to read the result values of 4-bit A/D conversion.<br>After enabling A/D, writing this register will start conversion. The write data will not affect these data bits. The user software reads the ADC_IF bit to monitor the complete of A/D conversion. |
| AD0_TOP3~0 | AD0 high limit to be checked in GREEN mode.<br>It wakes up MCU when above the limit. Their initial values are un-defined, |

| Bit Name | FUNCTION |
|---|---|
| | so writing AD0_H before entering GREEN mode. |
| AD0_BOT3~0 | AD0 low limit to be checked in GREEN mode.<br>It wakes up MCU when below the limit. Their initial values are un-defined, so writing AD0_L before entering GREEN mode. |
| AD1_TOP3~0 | AD1 high limit to be checked in GREEN mode.<br>It wakes up MCU when above the limit. Their initial values are un-defined, so writing AD1_H before entering GREEN mode. |
| AD1_BOT3~0 | AD1 low limit to be checked in GREEN mode.<br>It wakes up MCU when below the limit. Their initial values are un-defined, so writing AD1_L before entering GREEN mode. |
| AD2_TOP3~0 | AD2 high limit to be checked in GREEN mode.<br>It wakes up MCU when above the limit. Their initial values are un-defined, so writing AD2_H before entering GREEN mode. |
| AD2_BOT3~0 | AD2 low limit to be checked in GREEN mode.<br>It wakes up MCU when below the limit. Their initial values are un-defined, so writing AD2_L before entering GREEN mode. |
| AD3_TOP3~0 | AD3 high limit to be checked in GREEN mode.<br>It wakes up MCU when above the limit. Their initial values are un-defined, so writing AD3_H before entering GREEN mode. |
| AD3_BOT3~0 | AD3 low limit to be checked in GREEN mode.<br>It wakes up MCU when below the limit. Their initial values are un-defined, so writing AD3_L before entering GREEN mode. |
| AD0WKEN | =0 : disable AD0 check and wake-up in GREEN mode<br>=1 : enable AD0 check and wake-up in GREEN mode.<br>In GREEN mode, AD0 will be checked one time in one of the continuous 4 overflows of TCPB. If this enable bit is "1", the A/D will converse the input signal of AD0 channel first. Then the conversion value will be compared with AD0_H and AD0_L register. If the value is above AD0_TOP3~0 or below AD0_BOT3~0, the wake-up takes place. It will do nothing at the coming overflow when the AD0WKEN bit is "0". |
| AD1WKEN | =0 : disable AD1 check and wake-up in GREEN mode<br>=1 : enable AD1 check and wake-up in GREEN mode.<br>It does the similar operation as that at AD0WKEN. |
| AD2WKEN | =0 : disable AD2 check and wake-up in GREEN mode<br>=1 : enable AD2 check and wake-up in GREEN mode.<br>It does the similar operation as that at AD0WKEN. |
| AD3WKEN | =0 : disable AD3 check and wake-up in GREEN mode<br>=1 : enable AD3 check and wake-up in GREEN mode.<br>It does the similar operation as that at AD0WKEN. |

**Figure 11: 4-bit ADC and H/L limit check circuit**




## ➢ A/D conversion

The user software is used to set the I/O definition after power-on. In the power-on reset routine, it can set the ADC_PA register for AD3~0 channels, and the ADC_CTL2 for AD4 channel. If the user software wants to do A/D conversion for some channel, it need to enable ADC by writing "1" to the ADEN bit and the channel number to the AD_CH2~0 bits in the ADC_CTL1 register firstly. Secondly, it chooses the proper clock for A/D conversion at the ADCK1~0 bits of ADC_CTL2 register. If the ADCK1~0 bit is "11", the OSCL clock is used for conversion. The conversion time is:

**$T_{OSCL_CONV}$ = (period of RC16KHz) * 6**

If the ADCK1~0 bit is other, the divided clock of OSCH is used for conversion. The conversion time is:

**$T_{OSCH_CONV}$ = (period of selected ADC clock) * 32**

Next, it clear the ADC_IF bit in the INTF1 register. Now it writes any value to the ADC_D register to start the conversion. It will monitor the conversion from the state of the ADC_IF bit.

It means the conversion complete when the ADC_IF bit changes to "1". The result of the A/D conversion will be found in the ADC_D register. The following is a simple ADC flow:

Step 1: Select A/D input pins –
Set ADC_PA register and AD4/PB0 bit in ADC_CTL2 register

Step 2: Select A/D channel and enable –
Set ADEN = 1, AD_CH2~0 = wanted channel in ADC_CTL1 register

Step 3: Select A/D input clock –
Set ADCK1~0 in ADC_CTL2 register

Step 4: Clear ADC interrupt request flag –
Set ADC_IF = 0 in INTF1 register

Step 5: Start conversion –
Write $F_H$ to ADC_D register

Step 6: Wait conversion complete –
Check if ADC_IF = 1 in INTF1 register

Step 7: Read the result value of conversion –
Read ADC_D register

Step 8: Disable A/D –
Set ADEN = 0 in ADC_CTL1 register

Step 9: Wait the next conversion, then go to Step 2

If interrupt service is considered, it can set the ADC_IE bit in the INTC1 register before conversion. Then the conversion complete will set the ADC_IF bit automatically. The user software can check this bit in the interrupt service routine. The following is a simple ADC flow with interrupt:

Step 1: Select A/D input pins –
Set ADC_PA register and AD4/PB0 bit in ADC_CTL2 register

Step 2: Select A/D channel and enable –
Set ADEN = 1, AD_CH2~0 = wanted channel in ADC_CTL1 register

Step 3: Select A/D input clock –
Set ADCK1~0 in ADC_CTL2 register

Step 4: Clear ADC interrupt request flag –
Set ADC_IF = 0 in INTF1 register

Step 5: Enable ADC interrupt –
Set ADC_IE = 1 in INTC1 register

Step 6: Start conversion –

Write $F_H$ to ADC_D register

Step 7: Wait interrupt from conversion complete –

Step 8: Check ADC interrupt request flag in interrupt routine –
Read the ADC_IF bit in INTF1 register

Step 9: Check ADC_IF = 0? in interrupt routine
Yes, go to Step 10
No, does the proper routine then go to Step 7

Step 10: Read the result value of conversion in interrupt routine –
Read ADC_D register

Step 11: Disable A/D and return from interrupt routine –
Set ADEN = 0 in ADC_CTL1 register

Step 12: Wait the next conversion, then go to Step 2

Sampling channel voltage takes place at the first 2 or 16 clocks of the A/D conversion in according to the selected ADC clock. It may take more time to sample enough voltage for the critical application. The user can presume there are a 10pF capacitor and a 10Kohm resistor in internal of the A/D input channel. The settling time of sampling can be computed by adding the external loading to them.

## ➢ ADC check procedure in GREEN mode

In GREEN mode, the MCU will generate a ADC check procedure when the timer/counter B overflows. The CPU will not be waked up by this overflow. The ADC check procedure will check only one channel at each overflow. The next overflow will select the next channel to be checked. The 4 channels (AD0~3) will be checked in order. If the ADxWKEN bit is "0", it will ignore the check procedure in this overflow. The ADEN bit has no effect on ADC check procedure.

After A/D converse the input signal of the selected channel, the conversion value will be compared with ADx_H and ADx_L register. If the value is above ADx_TOP3~0 or below ADx_BOT3~0, the wake-up takes place. It will do nothing at this overflow when the ADxWKEN bit is "0". Only the OSCL clock is active in GREEN mode. It is used all the time in the ADC check procedure in spite of what the ADCK1~0 are. The conversion time in GREEN mode is:

**$T_{GREEN_CONV}$ = (period of RC16KHz) * 6**

No matter what the ADC_IE bit is, the A/D interrupt will be ignored in GREEN mode. The interrupt request flag ADC_IF is also ignored in GREEN mode. Sampling channel voltage takes

place at the first 2 clocks of the A/D conversion. The following is a simple ADC check flow for GREEN mode:

Step 1: Select A/D input pins –
Set ADC_PA register

Step 2: Enable wanted A/D wake-up channel –
Set AD_WKEN register

Step 3: Set the period time in TCPB for ADC check –
Set TCPBDH and TCPBDL register

Step 4: Set the overflow count (period time) in TB1 for CPU wake-up –
Set the TB1~0 bits in TBC/BZC register

Step 5: Select High/Low limit for the wanted channel –
Set ADx_H and ADx_L register

Step 6: Enter GREEN mode –
Set GREEN = 1 in PS register

Step 7: Wait ADC check from TCPB overflow by hardware –
if wake-up from TB1 overflow, go to Step 15 –

Step 8: AD0 conversion and compare AD0_H and AD0_L by hardware –
AD0WKEN = 0? Yes, go to Step 9; No, do conversion and compare
if ADC_D > AD0_H or ADC_D < AD0_L, go to Step 15

Step 9: Wait ADC check from TCPB overflow by hardware –
if wake-up from TB1 overflow, go to Step 15 –

Step 10: AD1 conversion and compare AD1_H and AD1_L by hardware –
AD1WKEN = 0? Yes, go to Step 11; No, do conversion and compare
if ADC_D > AD1_H or ADC_D < AD1_L, go to Step 15

Step 11: Wait ADC check from TCPB overflow by hardware –
if wake-up from TB1 overflow, go to Step 15 –

Step 12: AD2 conversion and compare AD2_H and AD2_L by hardware –
AD2WKEN = 0? Yes, go to Step 13; No, do conversion and compare
if ADC_D > AD2_H or ADC_D < AD2_L, go to Step 15

Step 13: Wait ADC check from TCPB overflow by hardware –
if wake-up from TB1 overflow, go to Step 15 –

Step 14: AD3 conversion and compare AD3_H and AD3_L by hardware –
AD3WKEN = 0? Yes, go to Step 7; No, do conversion and compare
if ADC_D > AD3_H or ADC_D < AD3_L, go to Step 15
go to Step 7

Step 15: Wake-up and check wake-up flag –

Read the AD/PA_WK and TB_WK bits in WK_FG register

Step 16: Check AD/PA_WK = 1? Yes, does the proper routine for ADC wake-up

Step 17: Check TB_WK = 1? Yes, does the proper routine for CPU wake-up

Step 18: Wait enter GREEN mode, then go to Step 2

# § Mask Option Table:

✧ MOPTION 1: Mask option register , default value [00-0]

| Register | **Bit3** | **Bit2** | **Bit1** | **Bit0** |
|---|---|---|---|---|
| Bit Name | LVD_opt1 | LVD_opt0 | -- | EXT_R/ internal RC |

✧ MOPTION 2: Mask option register , default value [0000]

| Register | **Bit7** | **Bit6** | **Bit5** | **Bit4** |
|---|---|---|---|---|
| Bit Name | INT3_edge | INT2_edge | INT1_edge | INT0_edge |

Bit0(EXT_R/internal RC) :

=0 : internal 910Khz RC oscillator

=1 : external R oscillator.

Bit3:bit2 (LVD_opt1:LVD_opt0) : define LVD voltage

0 0 = 2.0V (+/- 8%)

0 1 = 2.2V (+/- 8%)

1 0 = 3.3V (+/- 8%)

1 1 = 4.5V (+/- 8%)

Bit4 (INT0_edge) :

=0 : PA0/INT0 falling edge trigger

=1 PA0/INT0 rising edge trigger

Bit5 (INT1_edge) :

=0 : PA1/INT1 falling edge trigger

=1 : PA1/INT1 rising edge trigger

Bit6 (INT2_edge) :

=0 : PC0/INT2 falling edge trigger

=1 : PC0/INT2 rising edge trigger

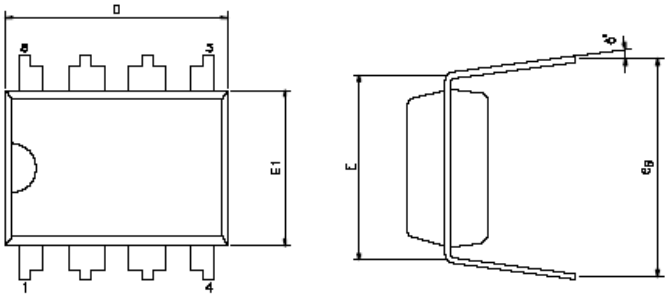
Bit7 (INT3_edge) :

=0 : PC1/INT3 falling edge trigger

=1 : PC1/INT3 rising edge trigger

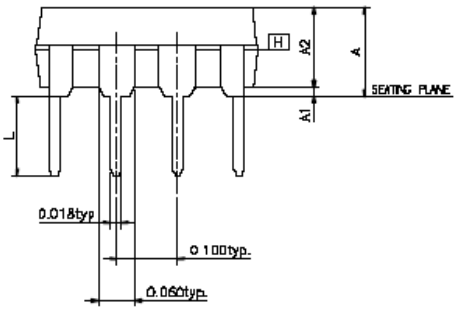
# § Package Information

DIP 8 pins




| SYMBOLS | MIN. | NOR. | MAX. |
|---|---|---|---|
| A | – | – | 0.210 |
| A1 | 0.015 | – | – |
| A2 | 0.125 | 0.130 | 0.135 |
| D | 0.355 | 0.365 | 0.400 |
| E | | 0.300 BSC. | |
| E1 | 0.245 | 0.250 | 0.255 |
| L | 0.115 | 0.130 | 0.150 |
| $e_B$ | 0.335 | 0.355 | 0.375 |
| θ° | 0 | 7 | 15 |

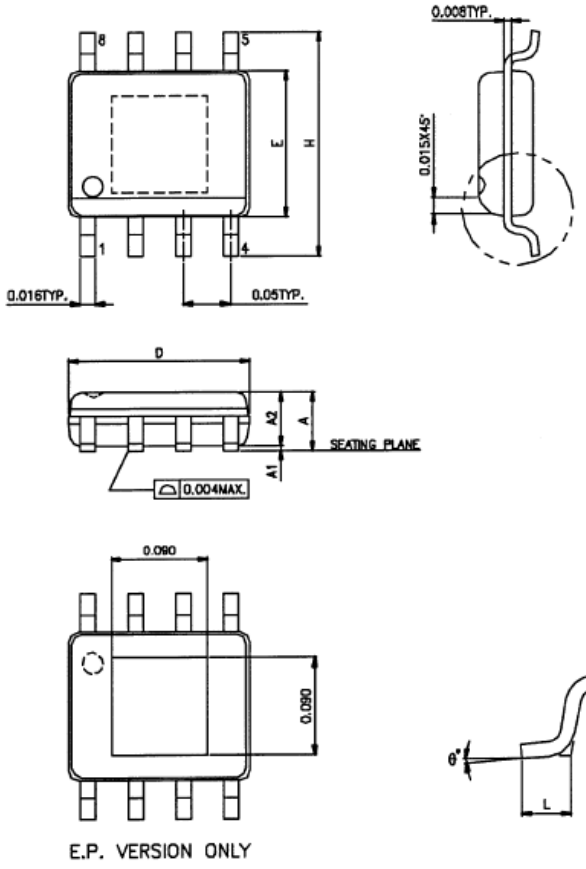
UNIT : INCH




NOTES:

1. JEDEC OUTLINE : MS-001 BA
2. "D","E1" DIMENSIONS DO NOT INCLUDE MOLD FLASH OR PROTRUSIONS. MOLD FLASH OR PROTRUSIONS SHALL NOT EXCEED .010 INCH.
3. eB IS MEASURED AT THE LEAD TIPS WITH THE LEADS UNCONSTRAINED.
4. POINTED OR ROUNDED LEAD TIPS ARE PREFERRED TO EASE INSERTION.
5. DISTANCE BETWEEN LEADS INCLUDING DAM BAR PROTRUSIONS TO BE .005 INCH MININUM.
6. DATUM PLANE H COINCIDENT WITH THE BOTTOM OF LEAD, WHERE LEAD EXITS BODY.

SOP 8 pins

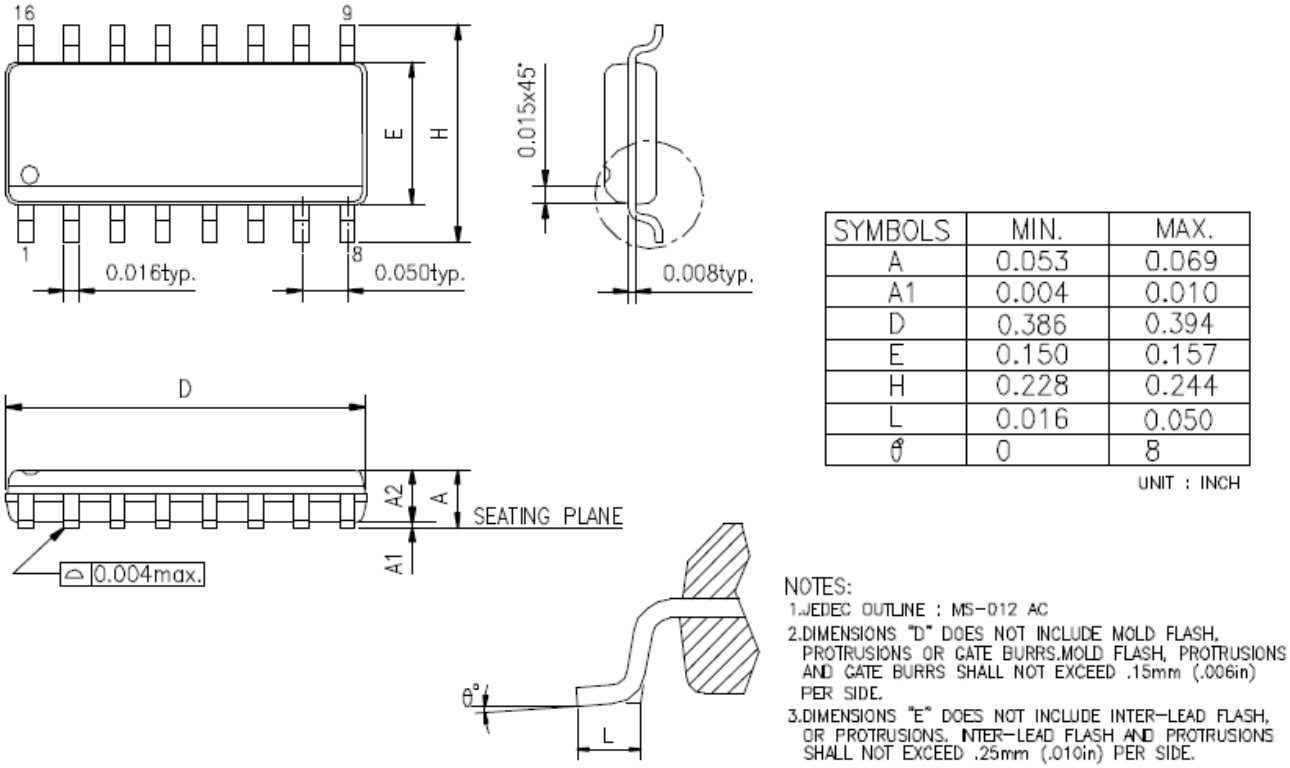



| SYMBOLS | MIN. | MAX. |
|---|---|---|
| A | 0.053 | 0.069 |
| A1 | 0.004 | 0.010 |
| A2 | – | 0.059 |
| D | 0.189 | 0.196 |
| E | 0.150 | 0.157 |
| H | 0.228 | 0.244 |
| L | 0.016 | 0.050 |
| θ° | 0 | 8 |

UNIT : INCH

NOTES:
1. JEDEC OUTLINE : MS-012 AA / E.P. VERSION : N/A
2. DIMENSIONS "D" DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS.MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED .15mm (.006in) PER SIDE.
3. DIMENSIONS "E" DOES NOT INCLUDE INTER-LEAD FLASH, OR PROTRUSIONS. INTER-LEAD FLASH AND PROTRUSIONS SHALL NOT EXCEED .25mm (.010in) PER SIDE.

SOP 16 pins




| SYMBOLS | MIN. | MAX. |
|---|---|---|
| A | 0.053 | 0.069 |
| A1 | 0.004 | 0.010 |
| D | 0.386 | 0.394 |
| E | 0.150 | 0.157 |
| H | 0.228 | 0.244 |
| L | 0.016 | 0.050 |
| θ | 0 | 8 |

UNIT : INCH

NOTES:
1.JEDEC OUTLINE : MS-012 AC
2.DIMENSIONS "D" DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS.MOLD FLASH, PROTRUSIONS AND GATE BURRS SHALL NOT EXCEED .15mm (.006in) PER SIDE.
3.DIMENSIONS "E" DOES NOT INCLUDE INTER-LEAD FLASH, OR PROTRUSIONS. INTER-LEAD FLASH AND PROTRUSIONS SHALL NOT EXCEED .25mm (.010in) PER SIDE.

# § Modified Record:

1. V 0.46 →V0.47, Add top mark data. 2011/5/24
2. V 0.47 →V0.48, 2011/6/7
   Page 25 : BZ frequency = (selected clock frequency) / (TCPDH+TCPDL)/2 →
   BZ frequency = (selected clock frequency) / (TCPDH+TCPDL)
   Page 38 : add LVD tolerance (+/- 8%)
3. Add two NOPs instructions must follow STOP and GREEN mode. 2013/07/01
4. Modify Mod_RC16K register increment will slow down OSCL. 2013/07/10
5. Power-on default is low speed mode. 2013/07/10
6. Operation Voltage 2.0V ~5.0V. 2013/07/10
7. Two kinds of package SOP8-FO8/SOP16-HOB. 2013/07/10
8. Operation Voltage 2.0V ~5.5V. 2015/09/04