Intermediate Electrical and Computer Engineering Design Experience

Watchdog Timer

EE3954

By:
Maarten Uijt de Haag, Tim Bambeck
Watchdog Timer

Mechanism to help your microcontroller reboot (restart) if your software hangs due to internal or external reasons.

Suppose, PIC#1 transmits a code to the device and waits for the reception of a character, but now the transmitted character got lost and PIC#1 will be waiting there forever.

Solution: Time Out
Watchdog Timer

In the previous example you would get stuck in this loop:

```
MAIN: ... 
      ... 
CHECK: btfss PIR1, RCIF ; check if UART rcvd byte
            goto CHECK ; if no, go check again
            movf RCREG,W ; if yes, put byte into W
            return ; return to MAIN
```

When the Watchdog timer is enabled you must perform a particular instruction ‘clrwdt’ in your program at a regular basis; if this instruction is not executed for a particular period of time (the time out period) the microcontroller will reset!
Watchdog Timer
Example

Update to the previous code including the watchdog timer:

MAIN:
...
...
CHECK: btfs PIR1, RCIF ; check if UART rcvd byte
goto CHECK ; if no, go check again
movf RCREG,W ; if yes, put byte into W
clrwdt ; reset watchdog timer
return ; return to MAIN

If the program gets stuck in the loop because the device is not transmitting information back, the “clrwdt” instruction is not executed, the program will time out, and the microcontroller will be reset.
Remember from Reset Notes ?/!

**Reset**
To place the device in a known state

- **Power-On Reset (POR)**
- **Brown-Out Reset (BOR)**
- **Parity Error Reset (PER)**
- **Watchdog Timer (WDT)**

**MCLR**
- during Normal Operation
- during Sleep
REGISTER 12-1: CONFIGURATION WORD (ADDRESS 2007h)

| CP1 | CP0 | DEBUG | — | WRT | CPD | LVP | BODEN | CP1 | CP0 | PWRT | WDTE | FOSC1 | FOSC0 |

bit13

in Flash EPROM

Microcontroller device used in the EE385A laboratory

pp122 in the PIC16F87X Datasheet (DS)
Recover from a RESET due to the Watchdog Timer

Register 3-2:  STATUS Register

<table>
<thead>
<tr>
<th>bit 7</th>
<th>bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRP</td>
<td>RP1</td>
</tr>
<tr>
<td>RP0</td>
<td>TO</td>
</tr>
<tr>
<td></td>
<td>PD</td>
</tr>
<tr>
<td></td>
<td>Z</td>
</tr>
<tr>
<td></td>
<td>DC</td>
</tr>
<tr>
<td></td>
<td>C</td>
</tr>
</tbody>
</table>

- **TO**: Time-out bit
  - 1 = After power-up, CLRWDT instruction, or SLEEP instruction
  - 0 = A WDT time-out occurred

- **PD**: Power-down bit
  - 1 = After power-up or by the CLRWDT instruction
  - 0 = By execution of the SLEEP instruction

Table 3-2:  Status Bits and Their Significance

<table>
<thead>
<tr>
<th>POR</th>
<th>BOR(^{(1)})</th>
<th>TO</th>
<th>PD</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>x</td>
<td>1</td>
<td>1</td>
<td>Power-on Reset</td>
</tr>
<tr>
<td>0</td>
<td>x</td>
<td>0</td>
<td>x</td>
<td>Illegal, TO is set on POR</td>
</tr>
<tr>
<td>0</td>
<td>x</td>
<td>x</td>
<td>0</td>
<td>Illegal, PD is set on POR</td>
</tr>
<tr>
<td>1(^{(2)})</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>Brown-out Reset</td>
</tr>
<tr>
<td>1(^{(2)})</td>
<td>1(^{(2)})</td>
<td>0</td>
<td>1</td>
<td>WDT Reset</td>
</tr>
<tr>
<td>1(^{(2)})</td>
<td>1(^{(2)})</td>
<td>0</td>
<td>0</td>
<td>WDT Wake-up</td>
</tr>
<tr>
<td>1(^{(2)})</td>
<td>1(^{(2)})</td>
<td>u</td>
<td>u</td>
<td>MCLR reset during normal operation</td>
</tr>
<tr>
<td>1(^{(2)})</td>
<td>1(^{(2)})</td>
<td>1</td>
<td>0</td>
<td>MCLR reset during SLEEP</td>
</tr>
</tbody>
</table>
Watchdog Timer

Setup and Time-out Period

The Watchdog timer uses a separate free-running RC oscillator.

The Watchdog timer also uses part of TIMERO to time out.

Nominal time-out period is 18ms.
Watchdog Timer
Setup and Time-out Period

Figure 26-1: Watchdog Timer Block Diagram

If the pre-scaler is assigned to TIMER 0 it cannot be used for the Watchdog timer.
Compare to Timer0

Figure 11-1: Timer0 Block Diagram

If the pre-scaler is assigned to TIMER 0 it can not be used for the Watchdog timer
Register 26-1: OPTION_REG Register

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2:0</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBPU (1): Weak Pull-up Enable bit</td>
<td>INTEDG: Interrupt Edge Select bit</td>
<td>T0CS: TMR0 Clock Source Select bit</td>
<td>T0SE: TMR0 Source Edge Select bit</td>
<td>PSA: Prescaler Assignment bit</td>
<td>PS2:PS0: TMR0 Prescaler/WDT Postscaler Rate Select bits</td>
</tr>
</tbody>
</table>

- **RBPU**: 1 = Weak pull-ups are disabled, 0 = Weak pull-ups are enabled by individual port latch values.
- **INTEDG**: 1 = Interrupt on rising edge of INT pin, 0 = Interrupt on falling edge of INT pin.
- **T0CS**: 1 = Transition on T0CKI pin, 0 = Internal instruction cycle clock (CLKOUT).
- **T0SE**: 1 = Increment on high-to-low transition on T0CKI pin, 0 = Increment on low-to-high transition on T0CKI pin.
- **PSA**: 1 = Prescaler is assigned to the WDT, 0 = Prescaler is assigned to the Timer0 module.
- **PS2:PS0**: TMR0 Prescaler/WDT Postscaler Rate Select bits

<table>
<thead>
<tr>
<th>Bit Value</th>
<th>TMR0 Rate</th>
<th>WDT Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>000</td>
<td>1:2</td>
<td>1:1</td>
</tr>
<tr>
<td>001</td>
<td>1:4</td>
<td>1:2</td>
</tr>
<tr>
<td>010</td>
<td>1:8</td>
<td>1:4</td>
</tr>
<tr>
<td>011</td>
<td>1:16</td>
<td>1:8</td>
</tr>
<tr>
<td>100</td>
<td>1:32</td>
<td>1:16</td>
</tr>
<tr>
<td>101</td>
<td>1:64</td>
<td>1:32</td>
</tr>
<tr>
<td>110</td>
<td>1:128</td>
<td>1:64</td>
</tr>
<tr>
<td>111</td>
<td>1:256</td>
<td>1:128</td>
</tr>
</tbody>
</table>

Time-out period is $128 \times 18\text{ms} = 2304\text{ms} = 2.3\text{sec.}$
What happens when WatchDog Timer - times out

There are two Scenarios:

1. WatchDog Timer - times out and causes a “RESET”
   - Next instruction executed is at 0x0000

2. WatchDog Timer - times out while PIC is in SLEEP:
   - PIC continues executing the instruction that followed the SLEEP Instruction.
Sleep

Instruction to put the device in its lowest power-consumption mode

I/O Ports maintain the status they had!

For lowest current consumption in this mode, all I/O pins should be either at VDD, or VSS, with no external circuitry drawing current from the I/O pin and the modules that are specified to have a delta sleep current should be disabled. I/O pins that are hi-impedance inputs should be pulled high or low externally to avoid switching currents caused by floating inputs. The T0CKI input should also be at VDD or VSS for lowest current consumption. The contribution from on-chip pull-ups on PORTB should be considered.
Sleep

Instruction to put the device in its lowest power-consumption mode

• Wake-up from SLEEP using:
  - Any device Reset,
  - Watchdog timer **Wake-up** (if enabled),
  - Any peripheral device that can set its interrupt flag while in sleep mode:
    • External INT pin, change on port pin, ADC, Timer 1, SSP, Capture,
Sleep (Resets WDT)

Wake-up Procedure from Watchdog Timer

Resume program by executing instruction that follows SLEEP instruction.
Sleep

*Wake-up Procedure from an Interrupt*

Wake-up regardless of GIE bit !!

If GIE = 0

Resume program by executing the instruction that follows SLEEP instruction.

If GIE = 1

Execute the instruction after the SLEEP instruction

Then:

Branch to program memory address 0x004
www.piclist.com