NetMedia

Basic Express BX-24 Application Note

Counting Pulses With Hardware Interrupts

Introduction

This application note illustrates how to use BasicX hardware interrupts to count pulses. The WaitForInterrupt system call is used to get access to the INT1 pin on the BX-24 processor. As soon as a task calls WaitForInterrupt, the task is blocked until the interrupt occurs, which means there is typically very little CPU overhead being used, as compared to alternative approaches such as polling loops.

System call WaitForInterrupt

WaitForInterrupt allows BasicX programs to respond quickly to hardware interrupts on an I/O pin. In the following example program, task PulseCountTask runs in a continuous loop calling WaitForInterrupt. The task blocks at the call until a rising edge appears on the INT1 pin, which is pin 11 on a BX-24.

Whenever the interrupt occurs, counter PulseCount is incremented, and the BX-24's red LED is pulsed in order to provide visual feedback.

Another task in the main program converts the pulse count to a string and transmits it through the Com1 serial port. The pulse count is transmitted once per second. Example code (a similar program is included in file CountingPulses.bas, which accompanies this document):

```
Private Const StackSize As Integer = 25
Private PulseCountStack(1 To StackSize) As Byte
Private PulseCount As Integer
'-------------------------------------------------------
Public Sub Main()

    Call InitializeBX24

    PulseCount = 0

    CallTask "PulseCountTask", PulseCountStack

    Do
        Debug.Print CStr(PulseCount)
        Call Delay(1.0)
    Loop

End Sub
'-------------------------------------------------------
```

```
'-------------------------------------------------------
Private Sub PulseCountTask()

    Const LEDpin As Byte = 25    ' Red LED
    Const LEDon  As Byte = 0
    Const LEDoff As Byte = 1

    Do
        Call WaitForInterrupt(bxPinRisingEdge)
        PulseCount = PulseCount + 1

        Call PutPin(LEDpin, LEDon)

        ' Debounce.
        Call Sleep(0.1)

        Call PutPin(LEDpin, LEDoff)
    Loop

End Sub
'-------------------------------------------------------
Private Sub InitializeBX24()

' BX-24 only -- configure INT0 as input-pullup in order
' to prevent it from causing an unwanted interrupt.
' INT0 is (internal) pin 11 on the 8535 chip, and is not
' connected to an external pin.

    Register.DDRD  = Register.DDRD And bx1111_1011
    Register.PORTD = Register.PORTD Or bx0000_0100

End Sub
'-------------------------------------------------------
```

Although the main task transmits the pulse count at a leisurely rate, the PulseCountTask, running in the background, is able to respond quickly to a rising edge on the interrupt pin.

Depending on how often pulses occur, the PulseCountTask may spend most of its time in WaitForInterrupt without consuming the higher CPU time of alternative approaches such as polling loops. This usually increases the amount of CPU time available for other tasks, unless the pulse rate is so high that PulseCountTask monopolizes the CPU.

Note that the WaitForInterrupt parameter can specify 3 different types of interrupt triggers -- falling edge, rising edge or logic low. The BasicX system library supplies the predefined enumerations bxPinFallingEdge, bxPinRisingEdge and bxPinLow for these parameters.