Interrupt Priorities
PIC 18 Series
by
Bruce Misner

There is little doubt, that in this day and age we can’t stand pat with what we know and how we accomplish our tasks. So there is little question that microcontrollers have to evolve as well. With the 18 series of microcontrollers from Microchip things have stayed the same and changed as well. In the past there was one interrupt to service all changing states and it was tedious to find out what happened to generate the interrupt. With the newer 18 series we can set priorities on interrupts to give us better control over what generates interrupts and finding out what caused the interrupt. Hence, Microchip has modified their chip to give you better interrupt support.

The interrupt priority management can be turned on or off. If the priorities are turned off, that is called compatibility mode. This means that interrupts function the way they used to under the 16 series. The interrupt vector is different so beware of that. When priorities are enabled a host of new registers and some old are used.

To turn on the interrupt priorities you must set bit IPEN in register RCON. Once this bit is set, by setting the GIEH bit in the INTCON register you will enable all interrupts that have been set to high priority. There are two IPR registers that allow you to set the priorities of the interrupts you wish. Then old regular registers are used to enable and flag the interrupts. In our example we will set priority interrupt for the A/D converter and use the A/D to generate interrupts that we must service. This is example is the now infamous PWMSET code, this time with interrupts. There are no interrupts associated with the PWM itself, so we just focus on the A/D.

The registers and bits required are

- register IPR1, set bit ADIP
- register PIE1, set bit ADIE
- register PIR1, flag bit is ADIF
- register RCON, set bit IPEN
- register INTCON, set bit GIEH

Now, we will have to go through all the IPR registers and INTCON registers and make sure all interrupts have low priority that we do not want to service. So, by setting ADIP bit we are enabling the A/D to generate a high level interrupt. By setting ADIE we enable the A/D to generate interrupts. We set the IPEN bit high to enable priority interrupts and the GIEH bit enables all high priority interrupts.
here’s the code

```assembly
Pwmset program
with
Priority interrupt support
written by Bruce Miser
May 2007

list p=18f442
include "p18f442.inc"
declare registers for our data
advalue set 0x26
adstate set 0x27
define program origin
org 0x0
goto start
define interrupt vector
org 0x08
goto intvect
set initial register values
start:

bcf INTCON2,RBIP ; low priority port b change int
bcf INTCON2,TMR0IP ; low priority timer 0 int
bcf INTCON3,INT2IP ; low priority internal int 2
bcf INTCON3,INT1IP ; low priority internal int 1
movlw 0x40 ; turn on A/D high priority
movwf IPR1 ; low priority all others
movlw 0 ; low priority all others
bsf RCON,IPEN ; enable priority interrupts
bsf PIE1,ADIE ; enable A/D interrupt
movlw 0x81 ; A/D clock = Fosc/32, A/D on
movwf ADCON0 ; A/D channel 0
movlw 0xff ; Port A input
movwf TRISA
movlw 0 ; left justified result, analog input
movwf ADCON1 ; Vdd and Vss reference
movwf TRISC ; Port C output
movwf advalue ; initial value 0
movwf adstate ; initial value 0
bsf INTCON,GIEH ; enable high priority interrupts
```
call initpwm ;initialize PWM output
;
; start of the main program loop
;
mainloop:
    movlw 0x81 ;value for ADCON for channel 0
    call channels ;call select channel routine
    bsf ADCON0,GO ;start A/D conversion

wait4ad0: ;wait for A/D int routine to change state
    btfss adstate,0 ;if adstate=0 not done
    goto wait4ad0 ;if adstate=1 conversion completed

    movf advalue,W ;update PWM with new value for duty cycle
    movwf CCPR1L
    bcf adstate,0 ;reset adstate to 0

    movlw 0x89 ;do same for A/D channel 1
    call channels
    bsf ADCON0,GO

wait4ad1:
    btfss adstate,0
    goto wait4ad1

    movf advalue,W
    movwf CCPR2L
    bcf adstate,0

    goto mainloop ;repeat forever
;
; Initialize PWM outputs
; duty cycle = 0
; frequency = 610 Hz
;
initpwm:
    movlw 0xff
    movwf PR2

    movf advalue,W
    movwf CCPR1L
    bcf CCP1CON,5
    bcf CCP1CON,4
    bsf T2CON,1
    bsf CCP1CON,3
    bsf CCP1CON,2

    movf advalue,W
    movwf CCPR2L
    bcf CCP2CON,5
    bcf CCP2CON,4
    bsf T2CON,1
    bsf CCP2CON,3
    bsf CCP2CON,2
    bsf T2CON,2
return

; A/D selection subroutine
; written to add delay to allow voltage to settle
; W register must have ADCON0 value before call
chanse:
    movwf ADCON0
    movlw 0x10
deloop:
    decfsz W
    goto deloop
return

; Interrupt Service Routine
intvect:
    movf ADRESH,W    ; read A/D value to W
    movwf advalue   ; save in advalue
    bsf adstate,0    ; change state to 1
    bcf PIR1,ADIF    ; reset A/D flag
    retfie           ; return from interrupt

end