ENGR 270 Lab #4 Online – Interrupts

Objectives

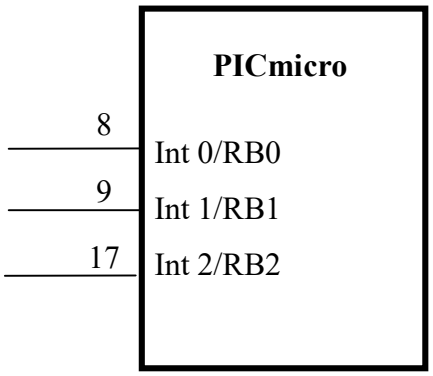
Apply interrupts to service unscheduled events.

Preparation

Complete the following steps before starting to work on the experiments in this lab:

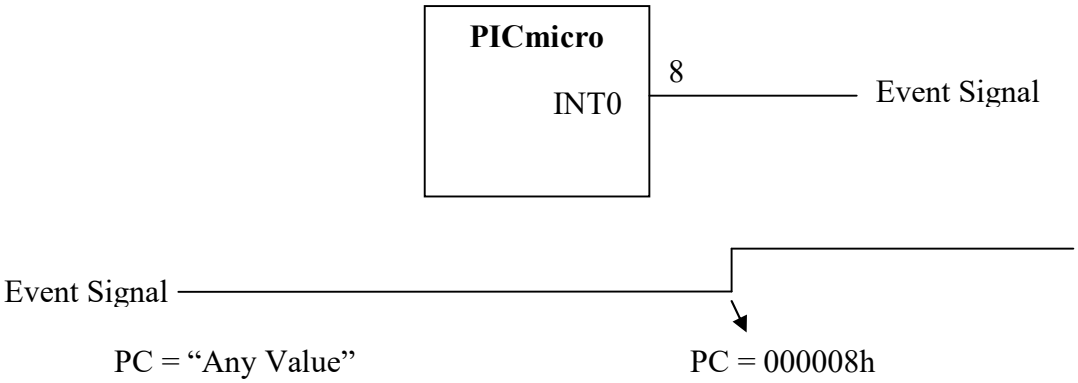
1) Complete Lab3 and associated report.
2) Complete lecture, homework and videos in Sections 4.1 through 4.4 "Interrupts & Oscillators"

The remainder of this section provides a brief overview of PICmicro's three external or peripheral interrupt pins which are needed to complete the lab.



The high priority interrupt vector is at 000008h program memory location and the low priority interrupt vector is at 000018h program memory location. Interrupt vector is the location that PC will be set to after an interrupt has occurred and has been acknowledged.

There are three external interrupts available on PICmicro (INT0-Pin 8, INT1-Pin9 and INT2-Pin 17). Below is an example of connecting interrupt INT0 to Event Signal. Anytime, Event Signal goes from low to high which causes a high priority interrupt and sets PC to 000008h.



In general, each interrupt source has three bits to control its operation. The functions of these bits are:

- Flag bit to indicate that an interrupt event occurred.
- Enable bit that allows program execution to branch to the interrupt vector address when the flag bit is set.
- Priority bit to select high priority or low priority (INT0 has no priority bit and is always high priority)

The following four SFR registers are used to control interrupt operations:

- RCON Register

Bit 7							Bit0	
IPEN	—	—	RI'	TO'	PD'	POR'	BOR'	RCON

bit 7 **IPEN:** Interrupt Priority Enable bit
1 = Enable priority levels on interrupts
0 = Disable priority levels on interrupts (PIC16CXXX Compatibility mode)

bit 6-5 **Unimplemented:** Read as '0'

bit 4 **$\overline{\text{RI}}$:** RESET Instruction Flag bit
1 = The RESET instruction was not executed (set by firmware only)
0 = The RESET instruction was executed causing a device Reset
(must be set in software after a Brown-out Reset occurs)

bit 3 **$\overline{\text{TO}}$:** Watchdog Time-out Flag bit
1 = Set by power-up, CLRWDT instruction or SLEEP instruction
0 = A WDT time-out occurred

bit 2 **$\overline{\text{PD}}$:** Power-down Detection Flag bit
1 = Set by power-up or by the CLRWDT instruction
0 = Cleared by execution of the SLEEP instruction

bit 1 **$\overline{\text{POR}}$:** Power-on Reset Status bit
1 = A Power-on Reset has not occurred (set by firmware only)
0 = A Power-on Reset occurred (must be set in software after a Power-on Reset occurs)

bit 0 **$\overline{\text{BOR}}$:** Brown-out Reset Status bit
1 = A Brown-out Reset has not occurred (set by firmware only)
0 = A Brown-out Reset occurred (must be set in software after a Brown-out Reset occurs)

- INTCON Register

Bit 7

GIE/ GIEH	PEIE/ GIEL	TMR0 IE	INT0 IE	RBIE	TMR0 IF	INT0 IF	RB IF

INTCO

bit 7 **GIE/GIEH:** Global Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked interrupts
0 = Disables all interrupts
When IPEN = 1:
1 = Enables all high priority interrupts
0 = Disables all interrupts

bit 6 **PEIE/GIEL:** Peripheral Interrupt Enable bit
When IPEN = 0:
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts
When IPEN = 1:
1 = Enables all low priority peripheral interrupts
0 = Disables all low priority peripheral interrupts

bit 5 **TMR0IE:** TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 overflow interrupt
0 = Disables the TMR0 overflow interrupt

bit 4 **INT0IE:** INT0 External Interrupt Enable bit
1 = Enables the INT0 external interrupt
0 = Disables the INT0 external interrupt

bit 3 **RBIE:** RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF:** TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INT0IF:** INT0 External Interrupt Flag bit
1 = The INT0 external interrupt occurred (must be cleared in software)
0 = The INT0 external interrupt did not occur

bit 0 **RBIF:** RB Port Change Interrupt Flag bit
1 = At least one of the RB7:RB4 pins changed state (must be cleared in software)
0 = None of the RB7:RB4 pins have changed state

Note: A mismatch condition will continue to set this bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared.

- INTCON2 Register

Bit 7 Bit0

RBPU'	INTE DG0	INTE DG1	INTE DG2	—	TMR0 IP	—	RBIP

INTCON2

bit 7 **$\overline{\text{RBPU}}$:** PORTB Pull-up Enable bit
1 = All PORTB pull-ups are disabled
0 = PORTB pull-ups are enabled by individual port latch values

bit 6 **INTEDG0:** External Interrupt 0 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 5 **INTEDG1:** External Interrupt 1 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 4 **INTEDG2:** External Interrupt 2 Edge Select bit
1 = Interrupt on rising edge
0 = Interrupt on falling edge

bit 3 **Unimplemented:** Read as '0'

bit 2 **TMR0IP:** TMR0 Overflow Interrupt Priority bit
1 = High priority
0 = Low priority

bit 1 **Unimplemented:** Read as '0'

bit 0 **RBIP:** RB Port Change Interrupt Priority bit
1 = High priority
0 = Low priority

- INTCON3

Bit 7						Bit0		
INT2 IP	INT1 IP	—	INT2 IE	INT1 IE	—	INT2 IF	INT1 IF	INTCON3

bit 7 **INT2IP:** INT2 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 6 **INT1IP:** INT1 External Interrupt Priority bit
1 = High priority
0 = Low priority

bit 5 **Unimplemented:** Read as '0'

bit 4 **INT2IE:** INT2 External Interrupt Enable bit
1 = Enables the INT2 external interrupt
0 = Disables the INT2 external interrupt

bit 3 **INT1IE:** INT1 External Interrupt Enable bit
1 = Enables the INT1 external interrupt
0 = Disables the INT1 external interrupt

bit 2 **Unimplemented:** Read as '0'

bit 1 **INT2IF:** INT2 External Interrupt Flag bit
1 = The INT2 external interrupt occurred (must be cleared in software)
0 = The INT2 external interrupt did not occur

bit 0 **INT1IF:** INT1 External Interrupt Flag bit
1 = The INT1 external interrupt occurred (must be cleared in software)
0 = The INT1 external interrupt did not occur

The interrupt priority feature is enabled by setting the IPEN bit (RCON<7>). When interrupt priority is enabled, there are two bits that enable interrupts globally. Setting the GIEH bit (INTCON<7>) enables all interrupts that have the priority bit set (high priority). Setting the GIEL bit (INTCON<6>) enables all interrupts that have the priority bit cleared (low priority). When the interrupt flag enable bit and

appropriate global interrupt enable bit are set, the interrupt will vector immediately to address 000008h or 000018h, depending on the priority bit setting. Individual interrupts can be disabled through their corresponding enable bits.

When the IPEN bit is cleared (default state), the interrupt priority feature is disabled. With this setting, the interrupts are compatible with PICmicro mid-range devices. In compatibility mode, the interrupt priority bits for each source have no effect. INTCON<6> is the PEIE bit, which enables/disables all peripheral interrupt sources. INTCON<7> is the GIE bit, which enables/disables all interrupt sources. All interrupts branch to address 000008h in compatibility mode.

When an interrupt is responded to, the global interrupt enable bit is cleared to disable further interrupts. If the IPEN bit is cleared, this is the GIE bit. If interrupt priority levels are used, this will be either the GIEH or GIEL bit. High priority interrupt sources can interrupt a low priority interrupt. Low priority interrupts are not processed while high priority interrupts are in progress.

Upon interrupt, the return address is pushed onto the stack and the PC is loaded with the interrupt vector address (000008h or 000018h). Once in the interrupt service routine, the source(s) of the interrupt can be determined by polling the interrupt flag bits. The interrupt flag bits must be cleared in software before re-enabling interrupts to avoid recursive interrupts. The "return from interrupt" instruction, RETFIE, exits the interrupt routine and sets the GIE bit (GIEH or GIEL, if priority levels are used), which re-enables interrupts.

For external interrupt events, such as the INT pins or the PORTB input change interrupt, the interrupt latency may be three to four instruction cycles. The exact latency is the same for one or two-cycle instructions. Individual interrupt flag bits are set, regardless of the status of their corresponding enable bit or the GIE bit.

Note: Do not use the MOVFF instruction to modify any of the interrupt control registers while any interrupt is enabled. Doing so may cause erratic microcontroller behavior.

Returning from interrupt handling code

Upon interrupt, the value of PC+2 (pointer to the next instruction) is pushed on the stack. This allows the interrupt handling code to return to the next instruction before interrupt by popping the stack and using the top of stack value as the PC.

The Instruction RETFIE when executed will automatically return the instruction execution back to the next instruction before the interrupt.

RETFIE	Return from Interrupt
Syntax:	[*label*] RETFIE [s]
Operands:	s ∈ [0,1]
Operation:	(TOS) → PC, 1 → GIE/GIEH or PEIE/GIEL, if s = 1 (WS) → W, (STATUSS) → Status, (BSRS) → BSR, PCLATU, PCLATH are unchanged.
Status Affected:	GIE/GIEH, PEIE/GIEL.
Encoding:	0000 0000 0001 000s
Description:	Return from interrupt. Stack is popped and Top-of-Stack (TOS) is loaded into the PC. Interrupts are enabled by setting either the high or low priority global interrupt enable bit. If 's' = 1, the contents of the shadow registers, WS, STATUSS and BSRS, are loaded into their corresponding registers, W, Status and BSR. If 's' = 0, no update of these registers occurs (default).
Words:	1
Cycles:	2

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	No operation	No operation	Pop PC from stack Set GIEH or GIEL
No operation	No operation	No operation	No operation

Example: `RETFIE 1`

After Interrupt

PC	=	TOS
W	=	WS
BSR	=	BSRS
Status	=	STATUSS
GIE/GIEH, PEIE/GIEL	=	1

Notes:

- Example – High priority interrupts and returns code.

Solution:

Address	**Content**
0x008	MVLW 23
0x00A	ADDWF 0x90, 1, 0
0x00C	CLRF 0x89
0x00E	RETFIE
…	
0x126	MVLW 23
0x128	ADDWF 0x90, 1, 0
0x12A	CLRF 0x89

A high Priority Interrupt occurs when instruction at location 0x128 is being executed. Where PC is equal to 0x12A.

❖

Interrupt Usage Example

The following code is written to demonstrate the use of low and high priority interrupts.

```
; File: Main.asm
; Desc: Demonstrates use of interrupts
;  * Main code toggles RA0 every second
;  * High priority INTO occurrence toggles RA1
;  * Low priority INT1 occurrence toggles RA2
;
; Last Update: June, 2020
; Auth: Class
;-------------------------------------------------------------------------------

    list    p=18f1220   ;processor type
    radix   hex         ;default radix for data
    ; Disable Watchdog timer, Low V. Prog, and RA6 as a clock
    config  WDT=OFF,LVP=OFF,OSC=INTIO2

    #include    p18f1220.inc    ;header file Includes SFR definitions
    #define      dCount      0x80

    org         0x000      ; Executes after rest
    GOTO    main

    org         0x008      ; Executes after high priority interrupt
    GOTO    HPRIO

    org         0x018      ; Executes after low priority interrupt
    GOTO    LPRIO

    org         0x20       ; Code start here
main:
    ; initialize all I/O ports
    CLRF    PORTA        ; Initialize PORTA
    CLRF    PORTB        ; Initialize PORTB
    MOVLW   0x7F
    MOVWF   ADCON1       ; Configure I/O as digital
    MOVLW   0xF8
    MOVWF   TRISA        ; Set RA<0:2> as output

    ; Enable INT0 and INT1
    BSF    INTCON, PEIE      ; enable all peripheral interrupts
    BSF    INTCON3, INT1IE   ; enable interrupt INT1
    BSF    INTCON, INT0IE    ; enable interrupt INT0
    BCF    INTCON3, INT1IP   ; set INT1 to low priority
    BSF    RCON, IPEN        ; enable priority levels on interrupts
    BCF    INTCON3, INT1IF   ; clear INT1 flag
    BCF    INTCON, INT0IF    ; clear INT0 flag
    BSF    INTCON, GIE       ; enable interrupts globally

mainL:    ; Main loop
    BTG     PORTA,0          ; Toggle RA0
    MOVLW   .10
    MOVWF   dCount
    CALL    delay            ; Delay for one second
    BRA     mainL

; Delay function waits for (dcount/10) seconds
; Assuming PICmicro is set to default RC Clock Oscillator
```

```
delay:
    MOVLW   .195     ; WREG = 195
; requires 195*4*128=99840 usec to complete the loop or Approx. 0.1 sec.
interDelay:
    DECF    WREG       ; 1 Cycle
    NOP                ; 1 Cycle
    BNZ    interDelay  ; 2 Cycles except the last time
    DECF    dCount,1
    BNZ    delay
    RETURN

; Interrupt Handling Section
HPRIO:     ; High priority interrupt INT0
    BTG     PORTA,1            ; Toggle RA1
    BCF     INTCON, INT0IF     ; Clear Interrupt 0 Flag
    RETFIE                     ; Return from interrupt

LPRIO:    ; Low Priority Interrupt INT 1
    BTG    PORTA,2            ; Toggle RA2
    BCF    INTCON3, INT1IF    ; Clear Interrupt 1 Flag
    RETFIE     ; Return from interrupt

    end        ; end program
```

Experiment 1. Interrupt

Write PICmicro Assembly code that performs the following:

1) In the main part of code RB7 is toggled every 10 seconds until execution is terminated
2) Each time, INTO occurs, RB6 value toggles once
3) Each time, high priority INT1 occurs, RB5 value toggles once
4) Each time, low priority INT2 occurs, RB4 value toggles once

Your test plan is expected to test when one interrupts occurs only and when multiple interrupts occur simultaneously.

Report Requirements

This lab and associated report must be completed individually. All reports must be computer printed (formulas and diagrams may be hand drawn) and at minimum:

For each experiment include:

- Clear problem statement in your words.
- Answer to any specific experiment questions (if any)
- Pseudo code which may be written in C-like syntax
- Disassembled code available after successful assembling
- Test plan which describes the input values and expect output/memory values for a successful design.
- Simulator output which shows the stimuli, relevant memory locations values showing validation based on test plan.

For the whole report include:

- A Cover sheet with your name, class, lab and completion date.
- A Lessons Learned section which summarizes your learning from this lab.
- A New Experiment section that has description of a new experiment and the experiment's results. Experiment should be related to material covered in class but not similar to one of the experiments in this lab.