

Panel 1

Prior to Le15

# Interrupts

Day 2 of 2

ME430 Mechatronics

1

Panel 2

Interrupt Priority Feature			
RCONbits.IPEN	All Interrupts High Priority	High and Low Interrupts	

Global / High Priority Bit			
Global Interrupt	INTCONbits.GIE	All interrupts off	Globally turn on interrupts
High Priority Interrupt	INTCONbits.GIEH	All interrupts off	High Priority on

Peripheral / Low Priority Bit			
Peripheral Interrupt	INTCONbits.PEIE	Peripherals off	Peripheral interrupts on
Low Priority Interrupt	INTCONbits.GIEL	Low Priority off	Low Priority on

Interrupt on Change			
- On RB4,RB5,RB6,RB7 (KB0,KB1,KB2,KB3)			
Enable	INTCONbits.RBIE	off	Turn on RB4:7 interrupts
Flag	INTCONbits.RBIF	-	Interrupt occurred!
Priority	INTCON2bits.RBIP	Low Priority	High Priority

2

Panel 3

Interrupt 0 - On RB0 (INT0)			
Enable	INTCONbits.INT0IE	off	Turn on INT 0 interrupts
Flag	INTCONbits.INT0IF	-	Interrupt occurred!
Edge	INTCON2bits.INTEDG0	Falling Edge	Rising Edge

Interrupt 1 - On RB1 (INT1)			
Enable	INTCON3bits.INT1IE	off	Turn on INT 1 interrupts
Flag	INTCON3bits.INT1IF	-	Interrupt occurred!
Priority	INTCON3bits.INT1IP	Low Priority	High Priority
Edge	INTCON2bits.INTEDG1	Falling Edge	Rising Edge

Interrupt 2 - On RB2 (INT2)			
Enable	INTCON3bits.INT2IE	off	Turn on INT 2 interrupts
Flag	INTCON3bits.INT2IF	-	Interrupt occurred!
Priority	INTCON3bits.INT2IP	Low Priority	High Priority
Edge	INTCON2bits.INTEDG2	Falling Edge	Rising Edge

3

Panel 4

Timer 0 Interrupt			
Enable	INTCONbits.TMR0IE	off	Turn on Timer 0 interrupts
Flag	INTCONbits.TMR0IF	-	Interrupt occurred!
Priority	INTCON2bits.TMR0IP	Low Priority	High Priority

Timer 1 Interrupt			
Enable	PIE1bits.TMR1IE	off	Turn on Timer 1 interrupts
Flag	PIR1bits.TMR1IF	-	Interrupt occurred!
Priority	IPR1bits.TMR1IP	Low Priority	High Priority

Timer 2 Interrupt			
Enable	PIE1bits.TMR2IE	off	Turn on Timer 2 interrupts
Flag	PIR1bits.TMR2IF	-	Interrupt occurred!
Priority	IPR1bits.TMR2IP	Low Priority	High Priority

Timer 3 Interrupt			
Enable	PIE2bits.TMR3IE	off	Turn on Timer 3 interrupts
Flag	PIR2bits.TMR3IF	-	Interrupt occurred!
Priority	IPR2bits.TMR3IP	Low Priority	High Priority

4

## Panel 5

Please tell me there are some library functions to help me out...

## 2.5 I/O PORT FUNCTIONS

PORTB is supported with the following functions:

TABLE 2-6: I/O PORT FUNCTIONS

Function	Description
ClosePORTB	Disable the interrupts and internal pull-up resistors for PORTB.
CloseRBxINT	Disable interrupts for PORTB pin <i>x</i> .
DisablePullups	Disable the internal pull-up resistors on PORTB.
EnablePullups	Enable the internal pull-up resistors on PORTB.
OpenPORTB	Configure the interrupts and internal pull-up resistors on PORTB.
OpenRBxINT	Enable interrupts for PORTB pin <i>x</i> .

Only help you set the individual enable bits.  
 Don't help with global setting (GIE or IPEN)  
 Don't help with flag within ISR

5

## Panel 6

## The PORTB Interrupt on change function

### OpenPORTB

**Function:** Configure the interrupts and internal pull-up resistors on PORTB.

**Include:** portb.h

**Prototype:** void OpenPORTB( unsigned char *config* );

**Arguments:** *config*  
 A bitmask that is created by performing a bitwise AND operation ('&') with a value from each of the categories listed below. These values are defined in the file portb.h.

**Interrupt-on-change:**

PORTB_CHANGE_INT_ON	Interrupt enabled
PORTB_CHANGE_INT_OFF	Interrupt disabled

**Enable Pullups:**

PORTB_PULLUPS_ON	pull-up resistors enabled
PORTB_PULLUPS_OFF	pull-up resistors disabled

**Remarks:** This function configures the interrupts and internal pull-up resistors on PORTB.

**File Name:** pbopen.c

**Code Example:** OpenPORTB( PORTB\_CHANGE\_INT\_ON & PORTB\_PULLUPS\_ON );

6

Panel 7

<b>OpenRB0INT</b> <b>OpenRB1INT</b> <b>OpenRB2INT</b>	
<b>Function:</b>	Enable interrupts for the specified PORTB pin.
<b>Include:</b>	portb.h
<b>Prototype:</b>	<pre>void OpenRB0INT( unsigned char <i>config</i> ); void OpenRB1INT( unsigned char <i>config</i> ); void OpenRB2INT( unsigned char <i>config</i> );</pre>
<b>Arguments:</b>	<p><b><i>config</i></b>  A bitmask that is created by performing a bitwise AND operation ('&amp;') with a value from each of the categories listed below. These values are defined in the file portb.h.</p> <p><b>Interrupt-on-change:</b>  PORTB_CHANGE_INT_ON    Interrupt enabled  PORTB_CHANGE_INT_OFF    Interrupt disabled</p> <p><b>Interrupt-on-edge:</b>  RISING_EDGE_INT        Interrupt on rising edge  FALLING_EDGE_INT        Interrupt on falling edge</p> <p><b>Enable Pullups:</b>  PORTE_PULLUPS_ON        pull-up resistors enabled  PORTE_PULLUPS_OFF       pull-up resistors disabled</p>
<b>Remarks:</b>	This function configures the interrupts and internal pull-up resistors on PORTB.
<b>File Name:</b>	rb0open.c rb1open.c rb2open.c
<b>Code Example:</b>	<pre>OpenRB0INT( PORTB_CHANGE_INT_ON &amp; RISING_EDGE_INT &amp; PORTB_PULLUPS_ON );</pre>

7

Panel 8

<b>OpenTimer0</b>	
<b>Function:</b>	Configure and enable timer0.
<b>Include:</b>	timers.h
<b>Prototype:</b>	<pre>void OpenTimer0( unsigned char <i>config</i> );</pre>
<b>Arguments:</b>	<p><b><i>config</i></b>  A bitmask that is created by performing a bitwise AND operation ('&amp;') with a value from each of the categories listed below. These values are defined in the file timers.h.</p> <p><b>Enable Timer0 Interrupt:</b>  TIMER_INT_ON    Interrupt enabled  TIMER_INT_OFF    Interrupt disabled</p>

8

## Panel 9

Let's do a big crazy example to make sure you get the idea with interrupts!

Let's make a program with 3 interrupts using the Priority Mode  
(That's about as hard as we'll make any interrupt problems)

### Timer 3 interrupt

- Low priority
- 16 bit timer with 1:4 Prescaler

### INT2 interrupt

- High priority
- Rising edge

### PORTB RB4:RB7 interrupt on change

- Low priority

9

## Panel 10

What should this crazy program do?

```
/** Global Variables *****/
int RBinterrupts = 0;
int timerOverflows = 0;
int resets = 0;
int RBmagnitude = 0;
```

### Timer 3 interrupt

- Increments timerOverflows

### INT2 interrupt

- Resets RBinterrupts, timerOverflows, and RBmagnitude
- Increments resets

### PORTB RB4:RB7 interrupt on change

- Increases RBmagnitude based on PORTB
- Increments RBinterrupts

10

## Panel 11

Start by making a new project using the "template with interrupts.c"  
(located on website under Courseware)

```
/******  
* Function:      void main(void)  
*****/  
#pragma code  
void main (void)  
{  
    // Setup pins to be digital  
  
    // Setup PORTB to be inputs  
  
    // Put the interrupts into Priority Mode  
  
    // Turn on the RB2 interrupt INT2 use the library  
    // Make it rising edge and high priority  
  
    // Turn on the Change on RB4:RB7 interrupt  
    // Make it low priority  
  
    // Start up Timer 3 with low priority interrupts  
    // Use bit mode with a 1:4 Prescaler  
  
    // Turn on High Priority interrupts  
  
    // Turn on Low Priority interrupts  
  
    while (1)  
    {  
        // This area loops forever but does nothing at all  
    }  
}
```

11

## Panel 12

Start trying to write the code to perform the comments

```
// Setup pins to be digital  
ADCON1 = 0x0F;  
  
// Setup PORTB to be inputs  
TRISB = 0xFF;  
  
...  

```

Make sure to declare the variables and  
include the appropriate library files

12

Panel 13

### Copy in the comments for the High Interrupt Service Routine

```

/*****
* Function:      void high_isr(void)
* Purpose:
*****/
#pragma interrupt high_isr
void high_isr(void)
{
    // High Priority Interrupt Service Routine (High ISR)
    // See if it was due to Interrupt 2 (it should be)
    // Reset the flag and counters, increment resets
}

```

Try to write the code for these

13

Panel 14

### Copy in the comments for the Low Interrupt Service Routine

```

/*****
* Function:      void low_isr(void)
* Purpose:
*****/
#pragma interruptlow low_isr
void low_isr(void)
{
    // Low Priority Interrupt Service Routine (Low ISR)
    // See if it's due to the timer overflow
    // If it is reset the flag and increment timerOverflows

    // See if it's due the change on RB4:RB7
    // If it is increase RBmagnitude, increment RBinterrupts, reset flag
    // Wierd note: Can only clear this flag AFTER reading PORTE
}

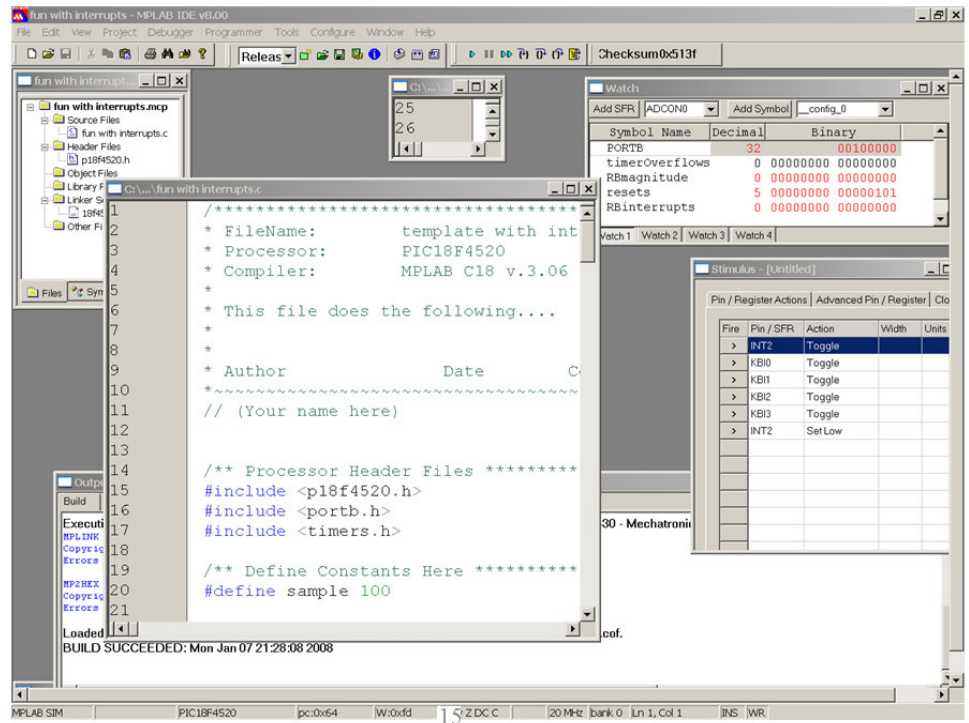
```

Try to write the code for these

14

Panel 15

## Add the Watch window and Stimulus



Panel 16

```

/** Header Files *****
#include <p18f4520.h>
#include <portb.h>
#include <timers.h>

/** Global Variables *****
int RBinterrupts = 0;
int timerOverflows = 0;
int resets = 0;
int RBmagnitude = 0;

// Setup pins to be digital
ADCON1 = 0x0F;

// Setup PORTB to be inputs
TRISB = 0xFF;

```

Panel 17

```

// Put the interrupts into Priority Mode
RCONbits.IPEN = 1;

// Turn on the RB2 interrupt INT2 use the library
// Make it rising edge and high priority
OpenRB2INT( PORTE_CHANGE_INT_ON & RISING_EDGE_INT & PORTE_PULLUPS_OFF);
INTCON3bits.INT2IP = 1;

// Turn on the Change on RB4:RB7 interrupt
// Make it low priority
OpenPORTB( PORTE_CHANGE_INT_ON & PORTE_PULLUPS_OFF);
INTCON2bits.RBIP = 0;

// Start up Timer 3 with low priority interrupts
// Use bit mode with a 1:4 Prescaler
OpenTimer3( TIMER_INT_ON & T3_16BIT_RW & T3_SOURCE_INT & T3_PS_1_4);
IPR2bits.TMR3IP = 0;

// Turn on High Priority interrupts
INTCONbits.GIEH = 1;

// Turn on Low Priority interrupts
INTCONbits.GIEL = 1;

```

17

Panel 18

```

/*****
* Function:      void high_isr(void)
* Purpose:
*****/
#pragma interrupt high_isr
void high_isr(void)
{
    // High Priority Interrupt Service Routine (High ISR)
    // See if it was due to Interrupt 2 (it should be)
    // Reset the flag and counters, increment resets
    if (INTCON3bits.INT2IF)
    {
        INTCON3bits.INT2IF = 0;
        RBinterrupts = 0;
        timerOverflows = 0;
        RBmagnitude = 0;
        resets++;
    }
}

```

18

Panel 19

```
#pragma interruptlow low_isr
void low_isr(void)
{
    // Low Priority Interrupt Service Routine (Low ISR)
    // See if it's due to the timer overflow
    // If it is reset the flag and increment timerOverflows
    if (PIR2bits.TMR3IF)
    {
        PIR2bits.TMR3IF = 0;
        timerOverflows++;
    }

    // See if it's due the change on RB4:RB7
    // If it is increase RBmagnitude, increment RBinterrupts, reset flag
    // Wierd note: Can only clear this flag AFTER reading PORTB
    if (INTCONbits.RBIF)
    {
        if (PORTBbits.KBI3)
            RBmagnitude = RBmagnitude + 3;
        if (PORTBbits.KBI2)
            RBmagnitude = RBmagnitude + 2;
        if (PORTBbits.KBI1)
            RBmagnitude = RBmagnitude + 1;
        if (PORTBbits.KBI0)
            RBmagnitude = RBmagnitude + 0;

        RBinterrupts++;
        INTCONbits.RBIF = 0;
    }
}
```