Panel 1

Prior to Le17

Pulse Width Modulation

Panel 2

Pulse Width Modulation (PWM)

What is Pulse Width Modulation? Why?

Two parts of the PWM:
- Frequency - 1/(time between rising edges)
- Duty Cycle - % of time PWM is high

Panel 3

Piezoelectric speakers

Cheap
Easy to use

Panel 4

Speed Control for a DC motor

Pulse Width Modulation

Figure 1. PWM signals of varying duty cycles
TABLE 2-9: PWM FUNCTIONS

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenPWMx</td>
<td>Configure PWM channel x</td>
</tr>
<tr>
<td>SetDCPWMx</td>
<td>Write a new duty cycle value to PWM channel x</td>
</tr>
</tbody>
</table>

Panel 5

Panel 6

Duty cycle range

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetDCPWM1</td>
<td>Write new duty cycle value to the specified PWM channel duty-cycle register.</td>
</tr>
<tr>
<td>SetDCPWM2</td>
<td>Write new duty cycle value to the specified PWM channel duty-cycle register.</td>
</tr>
<tr>
<td>SetDCPWM3</td>
<td>Write new duty cycle value to the specified PWM channel duty-cycle register.</td>
</tr>
<tr>
<td>SetDCPWM4</td>
<td>Write new duty cycle value to the specified PWM channel duty-cycle register.</td>
</tr>
<tr>
<td>SetDCPWM5</td>
<td>Write new duty cycle value to the specified PWM channel duty-cycle register.</td>
</tr>
<tr>
<td>SetDCPWM6</td>
<td>Write new duty cycle value to the specified PWM channel duty-cycle register.</td>
</tr>
<tr>
<td>SetDCPWM7</td>
<td>Write new duty cycle value to the specified PWM channel duty-cycle register.</td>
</tr>
<tr>
<td>SetDCPWM8</td>
<td>Write new duty cycle value to the specified PWM channel duty-cycle register.</td>
</tr>
</tbody>
</table>

Panel 7

Simple Open function even

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpenPWM1</td>
<td>Configures PWM channel.</td>
</tr>
<tr>
<td>OpenPWM2</td>
<td>Configures PWM channel.</td>
</tr>
<tr>
<td>OpenPWM3</td>
<td>Configures PWM channel.</td>
</tr>
<tr>
<td>OpenPWM4</td>
<td>Configures PWM channel.</td>
</tr>
<tr>
<td>OpenPWM5</td>
<td>Configures PWM channel.</td>
</tr>
<tr>
<td>OpenPWM6</td>
<td>Configures PWM channel.</td>
</tr>
</tbody>
</table>

Panel 8

Example code for PWM

OpenTimer2(TIMER_INT_OFF & T2_PS_1_4);
   //PWM period = (timer ticks) * 1 / (TOSC * TMR2 prescaler)
   // Fosc = 1 MHz (the default internal oscillator)
   // TMR2 prescaler = 4

OpenPWM1(141);
   // Resulting PWM Period = (141 * 1 / 4 / 1/1E6 * 4) = 0.00227 seconds → 440 Hz
   SetDCPWM1(300);
   // Set the duty cycle

   // Later in your code if you want it off
   SetDCPWM1(0);

   // Later in your code if you want it always high
   SetDCPWM1(1023);

   // Later in your code if you want it at 50%
   SetDCPWM1(512);
Panel 9

PWM shortcut formula:

\[
\text{Desired PWM period } = \left( \frac{\text{period ticks}}{2} + 1 \right) \times T_{\text{OSC}} \times T_{\text{M2 prescaler}}
\]

Limitations:

- **timer ticks**
- **Tosc**
- **TMR2 prescaler**

Panel 10

PWM also depends on the OSCCON setting

bit 6-4  **IRCF2:IRCF0**: Internal Oscillator Frequency Select bits

111 = 8 MHz (INTOSC drives clock directly)
110 = 4 MHz
101 = 2 MHz
100 = 1 MHz
011 = 500 kHz
010 = 250 kHz
001 = 125 kHz
000 = 31 kHz (from either INTOSC/256 or INTRC directly)

Panel 11

Open Timer 2

**OpenTimer2**

Function: Configure and enable timer2
Include: timers.h
Prototype: void openTimer2(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 timers.h.
Enable Timer2 Interrupt:

- **TIMER_INT_ON**: Interrupt enabled
- **TIMER_INT_OFF**: Interrupt disabled

Prescale Value:

- **T2_PSC_1**: 1 prescale
- **T2_PSC_1_4**: 1/4 prescale
- **T2_PSC_1_16**: 1/16 prescale

Panel 12

Sample calculation already done for you

```c
/** Define Constants Here ******************************************/
// Formula from the library function:
// PWM period = (period + 1) * 4 * TOSC * TMR2 prescaler
//
// We have setup:
// Tosc = 1 MHz
// TMR2 = 1/1 069 000
//
// Want a frequency for Middle A (440Hz)
// 1/440 = (period + 1) * 4 / 1/1000000
//
// Therefore A_m = 141
//
// It's 1 equation and 1 unknown. It's not that hard.
// I used excel and setup the period for basic notes:
```

Page 3 of 4
Example problem: Say we want 3 Hz. What is the Timer 2 prescaler, PWM period, and clock frequency?

Panel 14

Where does the PWM output?

Panel 15

Sample code:

Using the ADC program play me a nice middle F (349 Hz) when RB0 is pressed

Note to keep the LCD you must use the 4 MHz clock frequency (so use EC)

It works out nicely. What Timer 2 prescaler and Period did you use?

Write the C code needed to perform these tasks