CALIFORNIA STATE POLYTECHNIC UNIVERSITY, POMONA
COLLEGE OF ENGINEERING

ECE 3301L Spring 2019-2 Microcontroller Lab Felix Pinai

LAB 5X: Sound Generation

We will need to generate a sound to supplement the ohm meter designed in Lab #5. A beeper sound will be produced when a given resistor is measured to be close to 0 ohm. In fact, to make the experiment more interesting, two types of sounds will be generated based on the value of an unknown resistor is measured.

Assuming that RL is an unknown resistor. If RL is measured to be greater than 100 ohms, then no sound is produced. If RL is less than 100 ohms and greater than 10 ohms, then a sound with a frequency of 1 Khz and a duty cycle of 33% is generated. When RL is less than 10 ohms, a sound with a frequency of 2 Khz and a duty cycle of 50% is produced.

We will use PWM (Pulse-Width-Modulation) to generate the pulse needed to excite the buzzer. If you are not familiar with the setting of the PWM register, use the following link:

http://www.micro-examples.com/public/microex-navig/doc/097-pwm-calculator.html

to determine the settings of the required registers.

Based on the provided requirements, we will need to write two routines. The first one is to generate the 1Khz tone while the second one is for the 2Khz tone.

Use the value of 4 Mhz for the speed of the processor then the required frequency of your PWM followed by the duty cycle into the link above, the software will provide the value of several registers. Use these registers to write the two routines:

```
void gen_1Khz_beep (void)
{
    PR2        = ???;                    // values for 1Khz beep
    CCPR1L     = ???;
    CCP1CON = ???'
    T2CON      = ???;                    // Turn T2 on here
}

void gen_2Khz_beep (void)
{
    PR2        = ???;                    // values for 2Khz beep
    CCPR1L     = ???;
    CCP1CON = ???’
    T2CON      = ???;                              // Turn T2 on here
}
```

In addition, to turn off the buzzer, write the following routine:

```
void turn_off_buzzer()
{
    CCP1CON = 0;
}
```

Normally, that should be enough to get the buzzer to function as required. However, the code above uses the CCP1 as the pin for the output to the buzzer. Since this CCP1 is on PORTC bit 2 and this bit is also used to drive the 7-segment, we need to stay away from PORTC. The alternate is to use CCP2. The PIC18F4620 provides two alternate pins for CCP2. They can be at PORTC bit 1 or PORTB bit 3. Since PORTC cannot be used, then PORTB is the only choice. To force CCP2 to be on PORTB, we have to set the configuration properly. We will need to add the following line at the beginning of the code:

```
#pragma config CCP2MX = PORTBE
```

The next step is to change everything from CCP1 to CCP2. On the three routines above, change CCPR1L to CCPR2L and CCP1CON to CCP2CON.

Build the circuit shown on the added box on the schematics. Use a NPN transistor to drive the buzzer (2N2222 or 2N3904). To adjust the volume of the sound, the 100-resistor can be adjusted.

On the software side, insert some codes after you have measured the value of the unknown resistor to test whether you should turn the buzzer on. Call to either one of the three routines will depend on the value of the unknown resistor. Keep in mind that the value of the measured resistor is provided in Kohms and not in ohms.