RCET 3375 Experiment 5

Tone Generation (Funky Muzak)

Goals: The student will be able to:

Program input and output (I/O) ports.
Explain the purpose and operation of registers and I/O ports.
Send data to and get data from I/O ports.
Use a software look-up table.
Play some funky muzak.

Background Information:

The I/O ports of a microprocessor system are used any time there is a need to transfer data in or out of the system. For example, disk drives, keyboard, CRT, line printers, and modems could all be interfaced to the system via I/O ports.
A port is simply a group of pins that the CPU can either read data from or write data to. We can connect a device to the port's pins then easily transfer data to/from the device by using instructions and hardware that are already setup in the system. This makes it easier to interface hardware to the system.

Tasks:

1. Connect Port C bit 0 to Port B bit 0. Write a program that will send toggling (xor) data out Port C and read the data back from Port B then xor B and put it out C. Compare the data read with the data send. Display an xor of the read and written data. No using xor ICs!

2. Show the Port B bit 0 waveform in your lab notebook. Include the assembly code for this step. Use the XORWF command

3. Interface a speaker using a class D amplifier as shown in figure 1, to Porta bit 0. Write a program to cause a 1KHz square wave to be driven into the speaker when a button is pressed. The waveform must be exactly square.

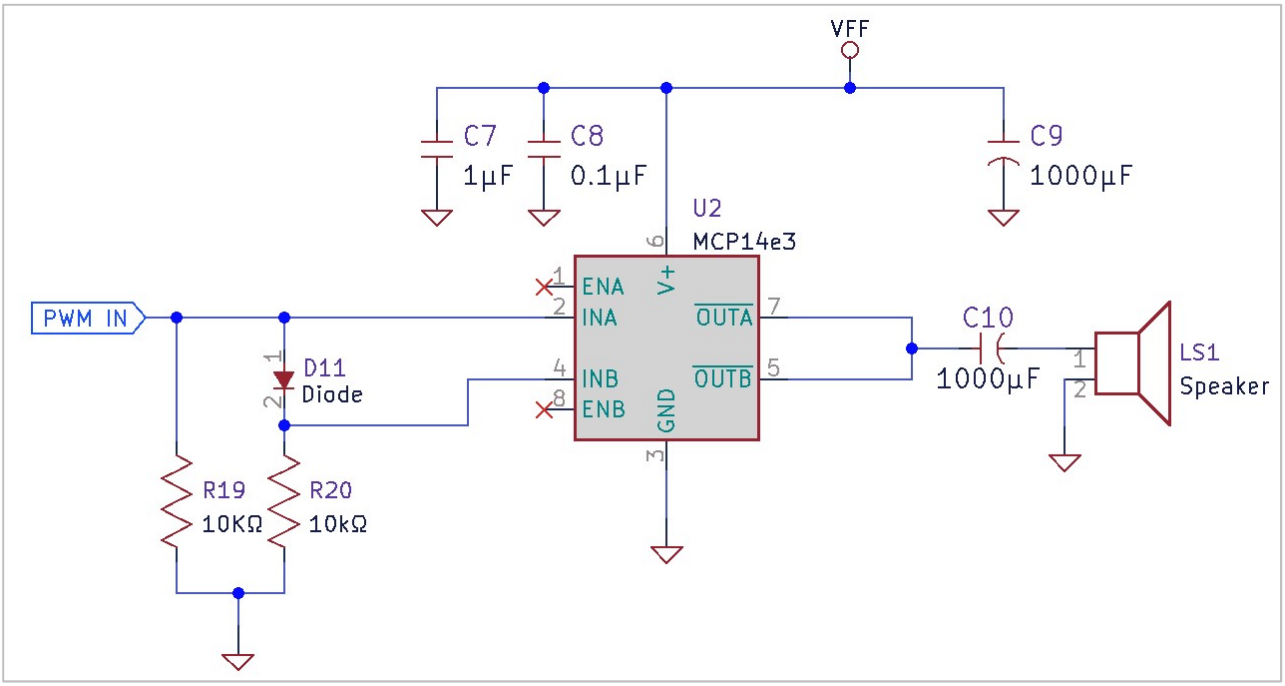

Figure 1: Class D Amplifier

4. Show the measured waveform in lab notebook. Include the assembly code for this step.

5. **Instructor Check point** (lab book – machine program, assembly program, flow chart)

6. Write a detailed description on how the Class D amplifier above works. (Consider the answers to the following; What does the MCP 14E5 do? What is the function of the diode? How do the speaker and cap effect the circuit? Ect.....)

7. Write a program that will cause the PIC to act like an electronic piano using look up tables. When key 0 is struck a continuous low tone will be driven from the speaker. When key 1,2,3 etc. are struck, successively higher tones will be played.

   One way to accomplish this is to have a table of delay values loaded into memory. One delay value for each key that can be struck. The main program will move the speaker in, wait some delay then move the speaker out and wait some more delay. The delay value will come from the lookup table. If key 0 is struck the main program will get the bottom value in the delay table, key 1 will

cause the next delay value to be used etc. This can be accomplished by loading a register with the bottom address of the delay table, reading the value of the key from the input buffer, adding this value to the bottom address which gives the address of the delay value. See Table below.

Button	Note	Frequency (Hz)	Period (s)	Table Value (Hex)	Table Value (Dec)
1	C_3	130.81	7.645E-3	EE	238
2	D_3	146.83	6.811E-3	D4	212
3	E_3	164.81	6.068E-3	BD	189
4	F_3	174.61	5.727E-3	B2	178
5	G_3	196	5.102E-3	9F	159
6	A_3	220	4.545E-3	8E	142
7	B_3	246.94	4.050E-3	7E	126
8	C_4	261.63	3.822E-3	77	119
9	D_4	293.66	3.405E-3	6A	106
10	E_4	329.63	3.034E-3	5E	94
11	F_4	349.23	2.863E-3	59	89
12	G_4	392	2.551E-3	4F	79
13	A_4	440	2.273E-3	47	71
14	B_4	493.88	2.025E-3	3F	63
15	C_5	523.25	1.911E-3	3B	59
16	D_5	587.33	1.703E-3	35	53

Table 1: Whole Note Suggested Delay Values

Your program may follow this general flow:

Step 1. Read Keypad
Step 2. Go to table and read value
Step 3. Push or Pull Speaker (Opposite action each time through loop)
Step 4. Call Delay
Step 5. Repeat.

8. **Instructor Check Point:** ♫ *PLAY SOME FUNKY MUZAK* ♫