Parallel Port IO Control

PURPOSE

The purpose of this lab is to introduce you to the equipment and procedures used in the CEG 4330/6330 Laboratory, based on UNO I/O ports. This is a three-week project.

PRELAB (20%): Complete the prelab assignments as teams. Remove any syntax error before you turn in prelab codes. The code should be made easy to be modified for handling a different keypad.

(1) (Week 2: 10%) Draw a complete circuit diagram and then write the code for Experiment (4), assuming you are using the 4x4 matrix membrane keypad (refer to www.parallax.com/product/27899 for documentation). You must use as few digital I/O pins as possible.

(2) (Week 3: 10%) For Experiment (5), identify the function of each key on the keypad, specify how a new song segment can be input, draw a complete circuit diagram and then write the code. Clearly specify the keypad you use. Again you must use as few digital I/O pins as possible.

EXPERIMENT (80%)

General notes

In all lab experiment specifications, a check mark (√) indicates a question for which you are required to write an answer. Please keep neat, carefully labeled, and properly ordered notes. A double check (√√) indicates a point in the experiment where you should have the Lab Instructor check your progress before continuing. However, please feel free to ask your Lab Instructor for help at any time. For the lab projects, you may incorporate any source codes you find on the web with or without modification as long as you clearly identify the source.

Notes for this lab

In this lab, you are to be familiar with the usage of the Arduino IDE and the control of the parallel ports of a UNO board.

(1) (5%) Connect the UNO to a USB port of the PC. Create a project that turns on or off the on-board LED depending on whether a push-button connected to a digital input pin of the board is depressed or not. In addition, whenever the push-button is depressed and then released, a letter "A" should be displayed inside the IDE serial monitor window. Make sure the serial monitor and the code use the same baud rate. No need to demonstrate this code.

(2) (10%) Follow instructions on https://www.arduino.cc/en/Tutorial/Debounce so that the LED is toggled whenever the push-button is depressed and then released. (Toggling occurs when the button is released.) You need to handle the debouncing with software√√.

(3) (25%) Connect a speaker to the UNO board (through a resistor if too loud) and modify the previous program so as to play through a short song segment repetitively. The push-button allows the toggling between playing of the song and turning the song off. The program should be designed so that the song can be represented as a character string (or multiple character strings) and it should include tones of different duration√√. Note that a character string is the same as an array of characters. You may impose conditions on how long the push-button needs to be pressed down.

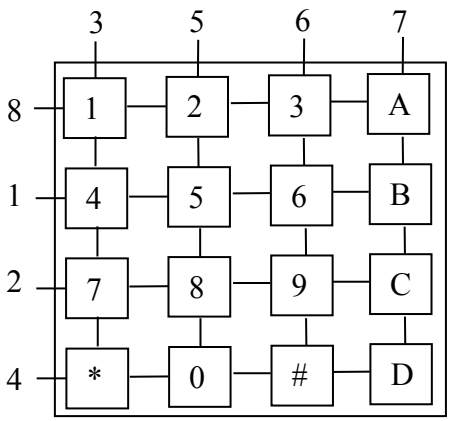
Background: To play a music note, you need to control both the frequency and the duration. In addition there should be pause after each tone.

- The note frequencies follow a logarithm scale, as shown in the following table. From C to high C, an octave, the frequency is doubled. Each octave is divided into 12 half tones. Using the note C as a starting point, the number of half tones to a note is called the pitch. So the frequency of a note = 261.2 x $2^{(pitch/12)}$ Hz.

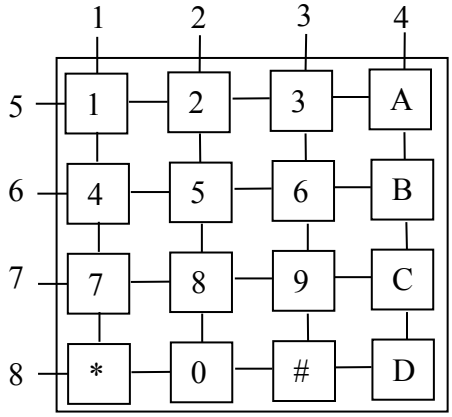
Note	C	D	E	F	G	A	B	High C
Pitch	0	2	4	5	7	9	11	12
Freq. (Hz)	261.6	293.6	329.6	349.2	392	440	493.8	523.2

- The duration is (70 x *length*) milliseconds where *length* is the duration in 1/16 measure that can be 1, 2, 4, 8, or 16. (e.g., the *length* is 2 for ♪) The pause after each note is (10 x *length*) milliseconds.

(4) (10%) Follow instructions on https://www.arduino.cc/en/Guide/Libraries to install a "keypad" library on the IDE. (You basically need to use the IDE menu Sketch->Include Library->Manage Libraries and then search for the keypad library to install.) Once the installation is completed, use the IDE menu File->Examples->Keypad->HelloKeypad to create an example sketch. Study the code, modify it slightly if necessary, and connect a keypad to the UNO board. Whenever any of the 12 numeric keys (i.e., including '*' and '#') is pressed down, the corresponding character is displayed inside the IDE serial monitor window. Assume no multiple keys get pressed at the same time. No need to demonstrate this code.
Hint: The keypad library source codes are inside the folder, My Documents\Arduino\libraries\keypad\src, in case you want to learn about the library internals.



This keypad has white letters on black keys. Viewed from above, Pin 1 is the left-most pin on the keypad circuit board.



This keypad has black letters on white keys. Viewed from above, Pin 1 is the left-most pin on the keypad circuit board.

(5) (30%) Combine the previous two projects. The keypad is used to enter a new song segment which replaces a previous song segment. You need to support the input of tones at least from C to high C. The push-button still controls whether to play the song. Demonstrate the ability to repetitively enter and play new song segments√√.

Keep all of your source codes. They are needed later on for system integration.

Extra Credit (10%): No partial credit for this part. Add a touch sensor (http://adafru.it/1374) to Experiment (5) so that the tones are played one octave higher whenever the sensor is being touched.