WE-GIRLS Summer Camp (June 19-22 2017)

Project 4 – Tune player

Cristinel Ababei, Dept. of Electrical and Computer Engineering, Marquette University

1. Objective

In this project, we'll play a series of musical notes. We'll use a digital-to-analog converter (DAC) to approximate a sine wave.

2. Introduction

We can generate sounds by turning one of Arduino's pins on and off at the right frequency and drive a buzzer with it. Such a sound is rough because the waveform generated is a "square waveform". To produce a more pleasing tone, we need a signal that is more like a "sine waveform".

A simple way to generate an "approximate" sine wave is to use a digital-to-analog converter (DAC). A DAC has a number of ***digital inputs*** and produces an ***analog output*** voltage proportional to the digital input value. We can build a simple DAC using an R-2R resistor ladder like the one shown in Figure 1.

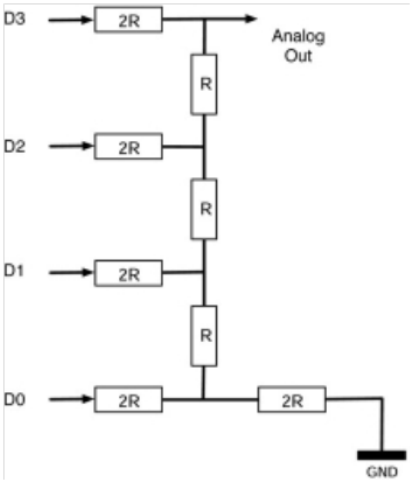

Figure 1: Example of an DAC constructed using an R-2R resistor ladder.

It uses resistors with values R=5K and 2R=10K. Each of the digital inputs will be connected to an Arduino digital output. The four binary digits represent the four bits of the digital number. In this way we can generate 16 different analog output values listed below:

D3	D2	D1	D0	Output
0	0	0	0	**0**
0	0	0	1	**1**
0	0	1	0	**2**
0	0	1	1	**3**
0	1	0	0	**4**
0	1	0	1	**5**
0	1	1	0	**6**
0	1	1	1	**7**
1	0	0	0	**8**
1	0	0	1	**9**
1	0	1	0	**10**
1	0	1	1	**11**
1	1	0	0	**12**
1	1	0	1	**13**
1	1	1	0	**14**
1	1	1	1	**15**

3. Project 4: Tune player

We will play a series of musical notes to create or generate the sound of a jingle. To do that we'll use a simple DAC controlled by the Arduino board. To keep it simple, we drive the buzzer directly with the signal from the ladder (a better design should use an audio amplifier to pick up the signal from the ladder and a speaker). The schematic diagram of this project is shown in Figure 2.

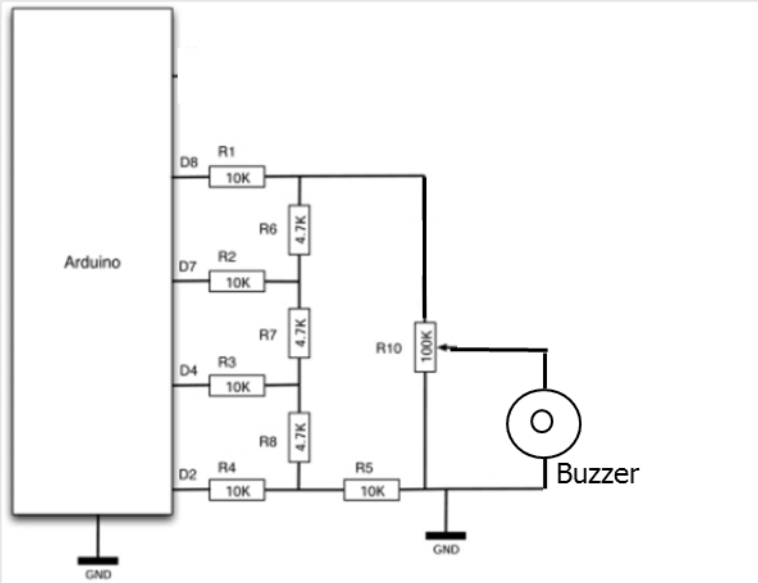

Figure 2: Schematic diagram.

The protoboard layout is shown in Figure 3.

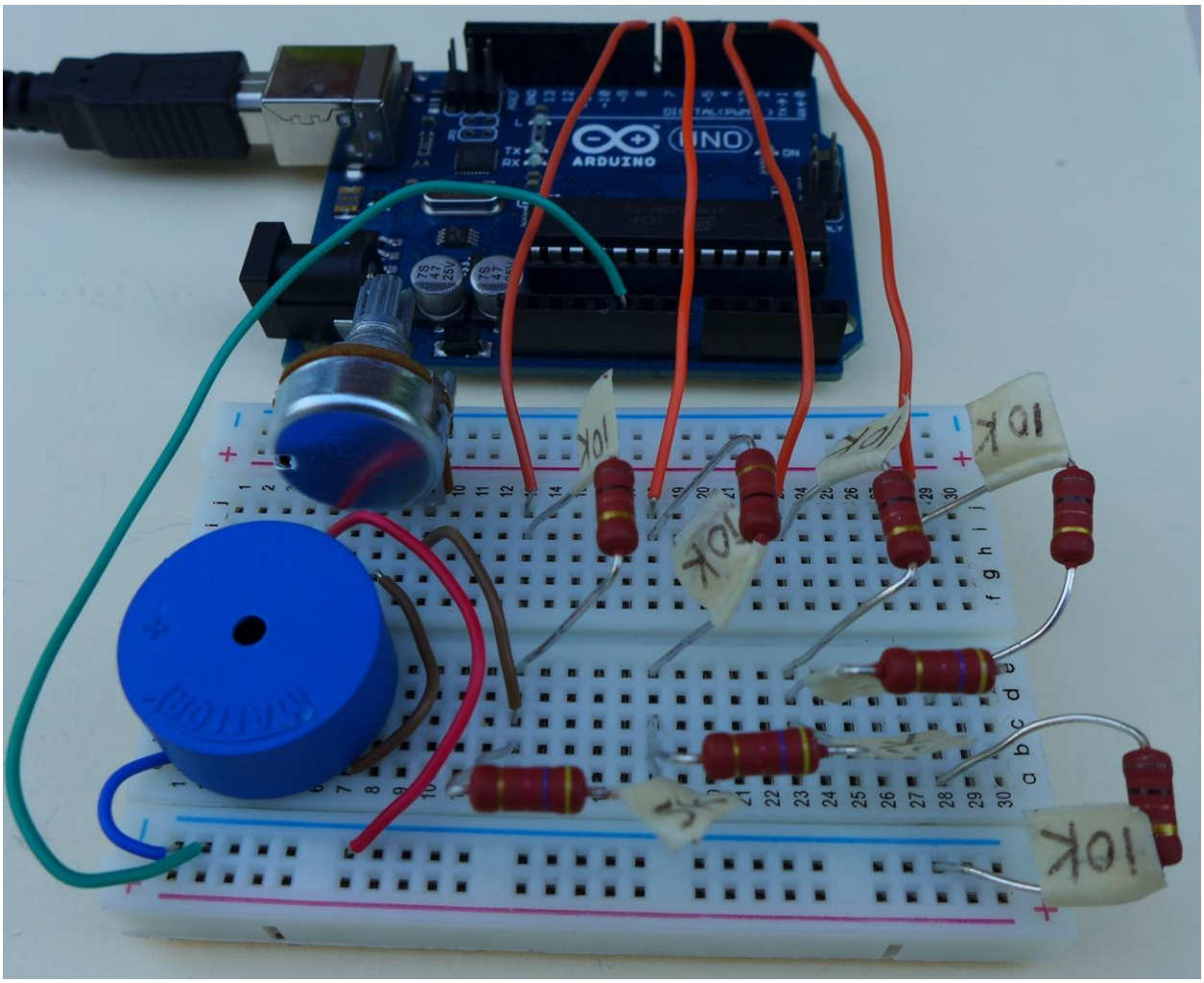

Figure 3: Protoboard layout.

Tunes are played from an array of characters. Each character corresponds to a note, and a space corresponds to the silence between notes. The main loop of the ***sketch*** looks at each letter in the song variable and plays it. When the whole song is played, there is a pause of five seconds and then the song is played again.

The ***sketch*** for this project is shown below:

```
// Project 4: Tune Player
// Plays a jingle

// Arduino's pins 2,4,7,8 control the R-2R ladder;
int dacPins[] = {2, 4, 7, 8};

// this is an array that stores a "hard-coded" rudimentary sine wave;
// to generate a sine wave these values are "stepped" through;
int sin16[] = {7, 8, 10, 11, 12, 13, 14, 14, 15, 14, 14, 13, 12, 11,
               10, 8, 7, 6, 4, 3, 2, 1, 0, 0, 0, 0, 0, 1, 2, 3, 4, 6};

int lowToneDurations[] = {120, 105, 98, 89, 78, 74, 62};
//                         A    B   C   D    E  F   G
int highToneDurations[] = { 54, 45, 42, 36, 28, 26, 22 };
//                          a   b   c   d   e   f   g

// Jingle Bells
char* song = "E E EE E E EE E G C D EEEE F F F F F E E E E D D E DD GG E E EE
              E E EE E G C D EEEE F F F F F E E E G G F D CCCC";

// Jingle Bells - Higher
//char* song = "e e ee e e ee e g c d eeee f f f f f e e e e d d e dd gg e e
//              ee e e ee e g c d eeee f f f f f e e e g g f d cccc";


void setup()
{
  for (int i = 0; i < 4; i++)
  {
    pinMode(dacPins[i], OUTPUT);
  }
}

void loop()
{
  int i = 0;
  char ch = song[0];
  while (ch != 0)
  {
    if (ch == ' ')
    {
      delay(75);
    }
```

```
    else if (ch >= 'A' and ch <= 'G')
    {
      playNote(lowToneDurations[ch - 'A']); // function defined below;
    }
    else if (ch >= 'a' and ch <= 'g')
    {
      playNote(highToneDurations[ch - 'a']);
    }
    i++;
    ch = song[i];
  }

  delay(5000);
}
// next function generates the note; the pitch of the note generated is
// controlled by the delay after each step of the signal;
// the loop to write out the waveform is inside a loop that writes out
// a number of cycles sufficient to make each note about the same duration;
void playNote(int pitchDelay)
{
  long numCycles = 5000 / pitchDelay + (pitchDelay / 4);
  for (int c = 0; c < numCycles; c++)
  {
    for (int i = 0; i < 32; i++)
    {
      setOutput(sin16[i]); // another function defined below;
      delayMicroseconds(pitchDelay);
    }
  }
}
// this function is called by playNote();
// it sets the value of the four output pins of the Arduino board
// that are connected to the R-2R ladder; the "&" operator is used to
// mask the "value" such that we see only the value of the bit we are
// interested in;
void setOutput(byte value)
{
    digitalWrite(dacPins[3], ((value & 8) > 0));
    digitalWrite(dacPins[2], ((value & 4) > 0));
    digitalWrite(dacPins[1], ((value & 2) > 0));
    digitalWrite(dacPins[0], ((value & 1) > 0));
}
```

Launch Arduino software and type-in or open the existing file with the above sketch. Once you have the sketch loaded press the “Verify” button. Once the sketch is verified, press the “Upload” button. Observe the operation and comment on it.

4. Assignment

Modify the sketch to play the jingle with higher notes.
Modify the sketch to play a different song – compose your own song.