ESE 150 – Lab 02: Digital to Analog Conversion

LAB 02

In this lab we will do the following:

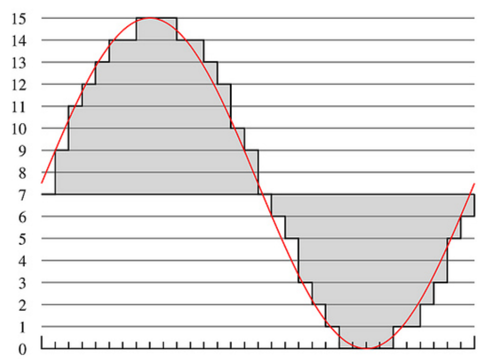
1. Take the samples you collected last week and reconstruct them in excel
2. Learn how to write Arduino code that outputs voltages between 0 and 5V
3. Learn how to use the Arduino's serial monitor to SEND data to the Arduino
4. Have the Arduino behave as a basic D2A to reconstruct your samples

Background:

Last week you used the Arduino to take an audio signal and turn it into a digital representation; you used an Arduino as an "A2D" converter (ADC). You should now have "samples" for the Arduino that you will use in today's lab.

Today we will import the data you captured last week from your A2D converter, into Excel. Next, we'll import the samples into the Arduino and attempt reconstruct the signal you sampled last week!

Recall the meaning of this picture from class:

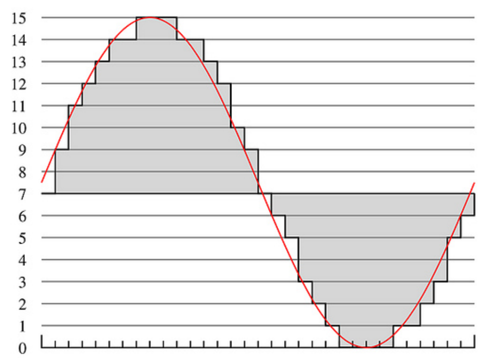


Lab Procedure:

Prelab: Reconstructing Original Signal in Excel

- You must have the output from your Arduino from Lab1 before starting this section
- You should have sampled a 2Vpp 300 Hz signal at 500 Hz, 1000 Hz, and 5000 Hz at 10b of precision.
- In this section, you'll use a spreadsheet (e.g., Excel, Numbers, Google Docs spreadsheet (docs.google.com)) to re-construct your original signal from the samples you collected in Lab 1

1. Open up your 10b sampled data in excel (you should have 3 rows: 500 Hz, 1kHz, 5kHz)
   a. Recall that you took 800 samples, so you should have 800 rows)
2. Use excel's plotting capability to produce a plot like this ***for each column*** of your data:
   a. The x-axis should just be the row # (aka the sample #)
   b. The y-axis should be the quantization level: 0->1023
   (since you are using the Arduino's *10-bit quantization data)*
3. Give each plot a title and label the axis with the appropriate units



4. Next, create & label 3 new plots:
   a. The x-axis should now be: **TIME**
   b. The y-axis should now be: **VOLTAGE**
   (your lowest voltage should be 0 due to the 1V offset that you used when you sampled the data)
5. How close does each plot look to a sine wave? Explain the difference between the three plots based on what we've covered so far in the course. [1 paragraph]
6. For the 5000Hz sample rate data
   a. Compute a column that quantizes the data to 8b of precision
   b. Compute a second column that quantizes the data to 2b of precision
   c. Create a pair of similar voltage-time plots (like 4) for the 8b and 2b quantized data sets you just computed

7. How close does each plot look to a sine wave? Explain the difference between the three quantization plots based on what we've covered so far in the course. [1 paragraph]
8. In your report, turn in the 8 labeled plots.

When you arrive in lab, compare your answers with your assigned partner. If your answers differ, discuss amongst yourselves and try to resolve your differences.

Early during the lab session, a TA will check you off on prelab. Go ahead and start working on the lab. It will take some time for the TAs to get around to all the groups.

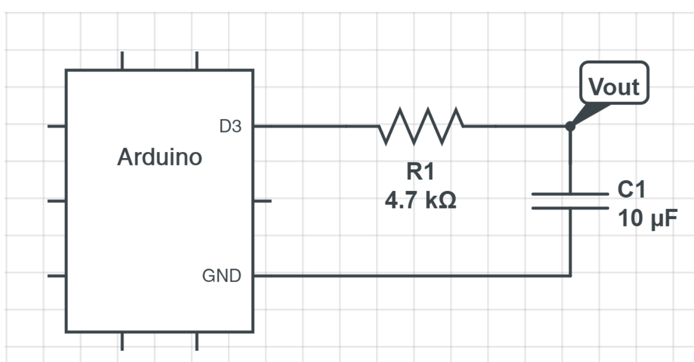
<u>Lab – Section 1: Setting up the Arduino as a D2A</u>

- In this section you'll calibrate your Arduino to output voltages between 0 and 5V

1. Obtain the following items from the lab bench stock (near the front of Detkin lab):

    (1) Arduino
    (1) Breadboard
    (1) 4.7 kΩ resistor
    (1) 10 uF Capacitor
    Wire to connect things

2. Using your Arduino and breadboard, connect the components above as shown in this schematic:

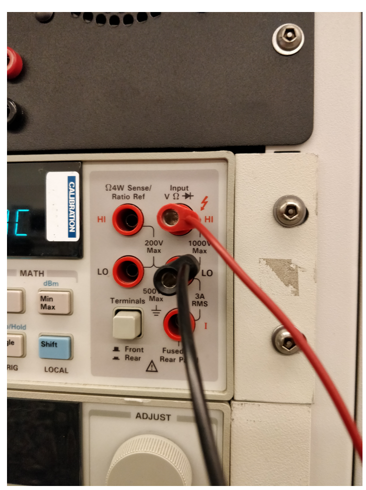


    a. Ensure that you connect the resistor to digital pin #3 on the Arduino
    b. Ensure that you connect the negative terminal of the capacitor to the Arduino's Ground terminal

3. Locate a "BNC -> Minigrabber" cable in the parts box at your workstation:

4. Connect one end to the oscilloscope, the red wire to "**VOUT**" in your circuit, and the black wire to "**GND**"
5. Turn on your oscilloscope
6. Locate (2) banana to minigrabber wire's, one RED, one BLACK:

7. Connect the red wire to the "VOUT" of your circuit, and the other to the "HI" terminal on your **DMM** (Digital Multimeter)
8. Connect the black wire to the "GND" of your circuit, and the other to the "LO" terminal on your **DMM** (Digital Multimeter)



9. Turn on the DMM

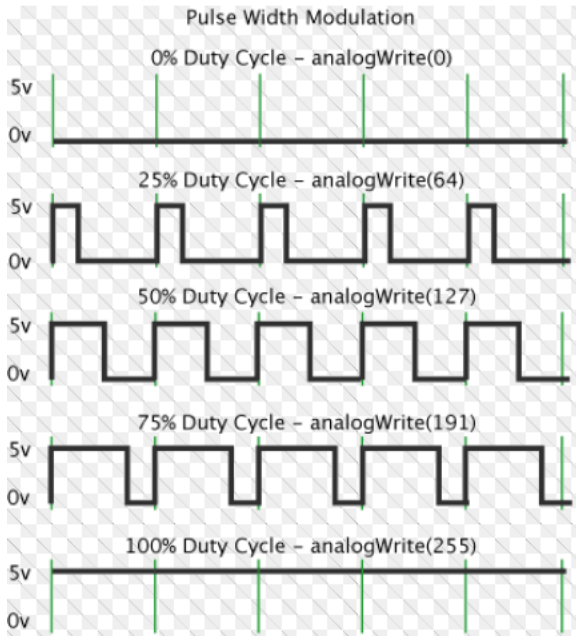
10. Connect your Arduino to your workstation's PC and open up the Arduino IDE software

11. Copy and paste the following code into your Arduino:

```
int pwmOut = 3 ;     // pin we'll output signal on
int holdTime  = 300 ;  // how long to hold output on PWM pin
void setup() {

  Serial.begin(9600);  // setup serial monitor speed
  pinMode(pwmOut, OUTPUT);  // configure pin for output only

}

void loop() {

  analogWrite(pwmOut, 0); // about zero volts
  delay(holdTime);
  analogWrite(pwmOut, 51); // about 1 volt
  delay(holdTime);
  analogWrite(pwmOut, 102); // about 2 volts
  delay(holdTime);
  analogWrite(pwmOut, 255);  // about 5.0 volts
  delay(holdTime);
}
```

Look carefully at the code and determine how it works:

1) Arduino's output pins can only put out voltages 0V or 5V, but nothing in between!
    a. It actually puts out a square pulse alternating between 0 and 5V every 2ms
2) What we can control is how long the output stays 5V during 1 cycle of that pulse
3) The line of code: analogWrite(3, 64) would hold 5V for about 25% of that cycle
4) The line of code: analogWrite(3, 255) will make it hold 5V for the entire cycle
5) This is referred to as "Pulse Width Modulation" or PWM, this chart helps visualize:



What is the effect of this code on the resistor-capacitor?

1) When the Arduino puts out 5V for 25% of its duty cycle, on pin 3, current flows through the capacitor and charges up the capacitor to about 1V
2) When the Arduino puts out 5V for 100% of its duty cycle, the capacitor and charges up the capacitor to about 5V
3) This happens because the resistor & capacitor have what's called an "RC time constant" you may know this from physics. Basically it takes time to charge up the capacitor all the way to 5V. We're taking advantage of this delay to produce voltage between 0V and 5V (e.g. – 0, 1, 2, 3V etc.)

YOUR FIRST JOB (calibrate Arduino output voltage):

1) Add to the provided Arduino code to produce voltages: 0V, 1V, 2V, 3V, 4V, 5V
2) The DMM is measuring the voltage across the capacitor, so you will know If your code is working!

YOUR SECOND JOB (produce a sine-wave):

1) Make your Arduino output the following voltages in this sequence:
   2V, 3V, 4V, 5V, 4V, 3V, 2V, 1V, 0V, 1V, 2V
2) Speed up the output to produce a sine-wave? If you do this right you should see a sine wave on your oscilloscope.
3) Make certain you understand how this actually works! It's the key to understanding a D2A converter
4) Capture an oscilloscope screenshot of the sine-wave.

Lab – Section 2: Communicating with the Arduino through the Serial Monitor

- We'd like to convert the samples you took last week, back to a voltage using the Arduino
- We need to find a way to send the sample BACK to the Arduino
- For this we'll use the Serial Monitor to actually send data to the Arduino (instead of just receive it)

1. Create a new sketch and copy and paste the following code in your Arduino IDE:

```
#define MAX_SAMPLES 801

// global variables
int samples [MAX_SAMPLES] ;
boolean samplesReceived = false ;

int outputPin = 3 ;     // PWM digital output pin
int holdTime  = 200 ;  // how long to hold output on PWM pin

void setup() {

  Serial.begin(9600);  // setup serial monitor speed
  pinMode(outputPin, OUTPUT);  // configure pin for output

}

void loop() {

  int i = 0 ;
  while (Serial.available() && i < MAX_SAMPLES ) {
      samples[i++] = Serial.parseInt() ;
      samplesReceived = true ;
  }

}
```

2. Upload the code to your Arduino

3. Open the Arduino's Serial Monitor:

4. In the top bar, type in the numbers: 100 200 300 400 1024, then press SEND
5. Those numbers have now been sent to the Arduino, "parsed" and stored in an array
6. Add to the Arduino code, if samples have been "received" have it print out all the possible samples that it received back to the Serial monitor (use Serial.prinln())
7. Try copying and pasting all 800 samples from your excel spreadsheet to the serial monitor and clicking SEND
    a. Note: In excel, select an entire column of data, press <ctrl> C, next, click on the top

Lab – Section 3: Putting it all together, Arduino as an A2D

- You now have a way of sending your samples to the Arduino
- You also have a way of putting voltages out to an Arudino output pin

1. Determine a conversion factor:
   a. Your samples are between 0 and 1023
   b. You can only output voltages been 0 and 5V
   c. Given what you know about section 2's code, can you determine a conversion factor to multiply the samples by to scale them between 0 and 5V, but more specifically 0 to 2V (since your original signal was 2Vpp with a 0V offset)

2. Create a new sketch that combines section 1 and section 2's Arduino code to output your samples as voltages
3. Your program should receive all 800 samples
4. After they are in the samples array, your program should convert each sample to PWM value
5. Lastly, it should output all 800 samples (now converted to voltages) to the output pin
6. This will take time. Once you have the output take many screenshots of the data on your oscilloscope. You should allow this code to loop to produce a continual sine wave
7. Before leaving lab, show your generated sine wave to a TA and answer a few questions. This is the Lab Exit Check-off.
8. Cleanup your lab station, leaving everything as you found it when you arrived.

Postlab: Synthesis

1. What equation could you use to create the sample data for the sine wave mathematically? (a function of sine-wave frequency, sample rate, and quantization)
2. Develop a spreadsheet that reproduces the 800 samples for the 300Hz wave sampled at a 1000 Hz sample rate with 8b of precision
3. With a small change from the first spreadsheet, create a second that that produces a 100Hz wave.

   With this technique, you can create (synthesize) sounds directly—no need to generate and sample the source.

HOW TO TURN IN THE LAB

- Upload a PDF document to canvas containing:
  - Prelab (8 plots, 2 answers)
  - Section 1 (revised Arduino code to produce specified voltages, sine-wave screen shot)
  - Section 2 (revised Arduino code)
  - Section 3 (final Arduino code and screenshots)
  - Postlab (answer to question 1 in PDF)
  - Please include adequate labels and text so it is clear where you have included each item requested in your report.
- Upload your postlab spreadsheets to the designated canvas lab assignment
- Due by Friday 5pm