Lab building a DAC with arduino

source: https://app.ph.qmul.ac.uk/wiki/ajm:teaching:arduino-pi:projects:arduino:dac-audio:dac

materials:
- 20K (9) and 10K (8) resistors 1% if possible
- arduino Uno
- voltmeter, oscilloscope
- 150 ohms
- IC 741
- jumper wires

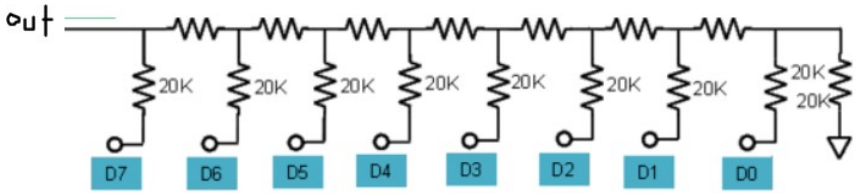
@ veronique.lankar 2020

Part I : simple DAC – 7 10K and 9 2)

Arduino does not have a read digital to analog converter (DAC). It uses the PWM method to simulate analog values. We will build a simple 8-bits DAC with a simple circuit called R-2R.

It uses only 2 values of resistors. 10K and 20K (could be any value as long as the raio is 2.

Here is the set-up:



The set up up converts a 8 bits number D7 D6 D5 .. D0 to an analog value at the output.
D0 is the least significant bit and D7 the most significant bit.

We apply HIGH or LOW at the digital input D0 to D7
The number 35 means 00100011 with D0=0. D1=1. D5=1.

A 1 is a HIGH (5V for arduino) and 0 is a LOW (less than 2.5V).

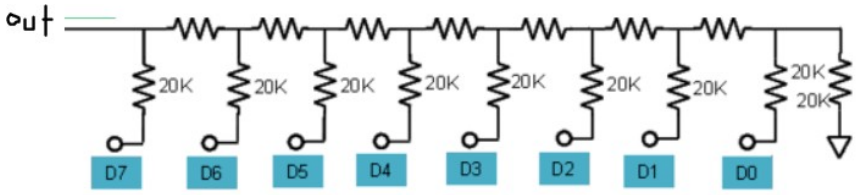
Vout= 5* (D0/256+ D1/128+D2/64+D3/32+D4/16+D5/8+D6/4+D7/2)

so Vout = 5 (1/256+1/128+1/8) = 5x 0.1367 = 0.68 V or **5x 35/256 = 0.68V**

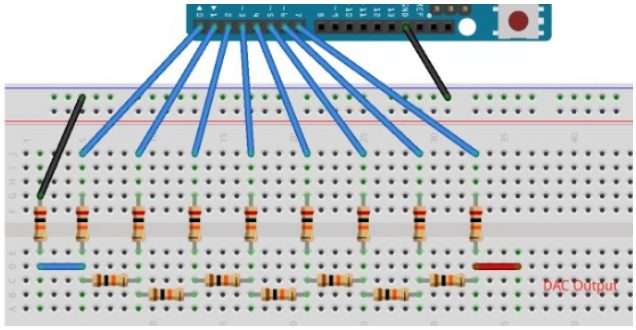
see : https://www.tek.com/blog/tutorial-digital-analog-conversion-r-2r-dac

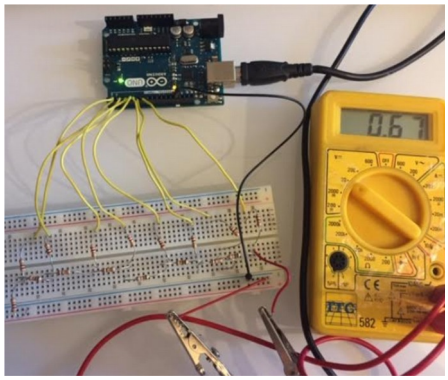
PART I : basics 8-bits DAC

Use resistors 20K and 10K 1% if possible. Here our set up:



D0 is connected to digital 0 of arduino D1 to digital 1, D2 to digital 2 .. D7 to digital 7
See next page





After building the set up upload the code ***DAC_arduino_1***.
The arduino will write the number 35 in binary form or 00100011. So D0=1, D1=1, D5=1
Use a voltmeter to measure the output
Vout = 5 x (1/256+1/128+1/8)= 0.68V or **35/256 = 0.68V**

Vout = _____ Is it close?

Modify the code so the binary number is stored in an array before setup(). In the loop() write the binary number at the digital pins using a for loop. Use a voltmeter to see if you get the write answer.
Hint. Something like: int bin[8] = {1,1,0,0,0,1,0,0};

Write a code so you write all those values consecutively using arrays and for loop inside loop(). Write a delay between each values so you have time to write down the number.

Decimal form	Binary form	Voltmeter value
35		
255		
127		
55		
70		

Show your code to your instructor

PART II hardware registers

See appendix B. The digital pins 0 to 7 are attached to PORT D of the AVR chip.
We can manipulate the PORT D without using the IDE functions pinMode() and digitalWrite() and it takes less computation time.

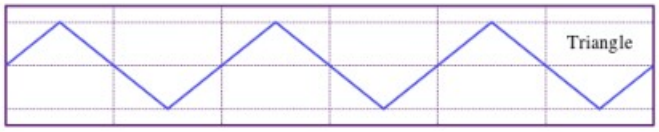
DDRD is the register that defines if the pins are output or input. Here they are output and we are using all of them so we can write in the setup() DDRD = B11111111 or better DDRD=0xff

In the loop() we can just write PORT D = 35 (decimal form of 00100011) instead of using digitalWrite()

Modify the code to implement this improvement. Store the binary numbers 35, 250, 127, 55, 70 in an array before setup(). Remember to set register PORTD as outputs in setup(). In the loop() write each of the number of array at the pins using PORT D (using for loop) but wait a 3000ms between each output so you have to time to measure the analog value. PORTD=bin[i]; // in a for loop.

PART II waveform ramp

A) Try to make a triangle form that you can see on the oscilloscope



see below

Use two loops like:

```
for (int a =0; a <256; a++)
{
PORTD =a;
delayMicroseconds(50);

}

for (int a =255; a>=0 ; a--)
{
PORTD =a;
delayMicroseconds(50);

}
```

Visualize the result on the oscilloscope.

B) We can do the same with the serial plotter of the arduino. To plot the analog value. For that plug the output of the circuit into the pin A0 which is an analog input connected to an ADC. Arduino will turn the analog value into a binary number between 0 and 1023.

Upload the code DAC_arduino_2 and try it. See if you can display the analog values.

Then modify the code so you get a triangle wave form.

PART II waveform sine

we can make a sine wave. Here the clode ***DAC_arduino_3*** make a sine wave with 100 samples and we wait 50us between each one. So 100x50us = 200Hz. Read the maximum of the sine wave.
What is it ? _____. The voltage varies between 0 and 5V.

Upload the code and display the sine wave on the oscilloscope. See if the oscilloscope find the right frequency. Try to understand the code.

The values for the samples were computed with a python program sine.py

```
import math
for x in range(0, 100):
   print str(int(127+127*math.sin(2*math.pi*x*0.01)),)+str(","),
```

Make your own python code so you get 500Hz instead of 200 Hz.

PART III buffer

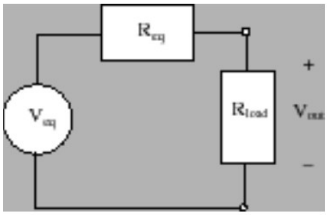
reference:
https://app.ph.qmul.ac.uk/wiki/ajm:teaching:arduino-pi:projects:arduino:dac-audio:buffer-circuits

Let's place place a load at the output of the DAC. The load is a LED and a resistor of 150 ohms.

Output of 2R-R–> resistor –> Gnd

And run the code sine. Connect the oscilloscope across the load.
What is the max voltage you get ? _____ is it less? What is the frequency ?

With a small load, the current through the circuit is large and most of the voltage drops is across the equivalent resistor of the circuit. We get a distorted sine wave. Use a 10K resistor instead of the 220ohms and see if the sine wave is better. What is the amplitude now? (the max) _____

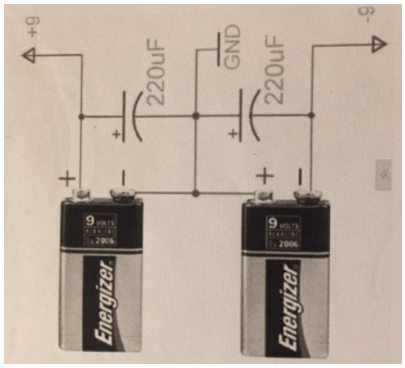


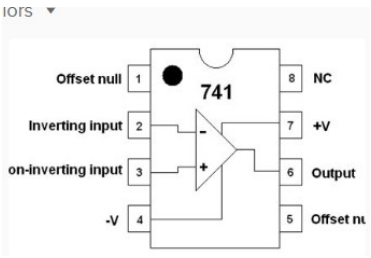
We can show (Thevenin's theorem) that the circuit with load is equivalent to the simple circuit with Req = 10K (for a R-2R) and Vout = 5(D0/256+D1/128+D2/64 + …)
If the load (150ohms) is small compared to 10K then all the voltage drop is across the 10K since V= R x I (ohms law). In a circuit in series the voltage drops is the largest across across the largest R. Here the ratio between 10K and 150 is about 60.

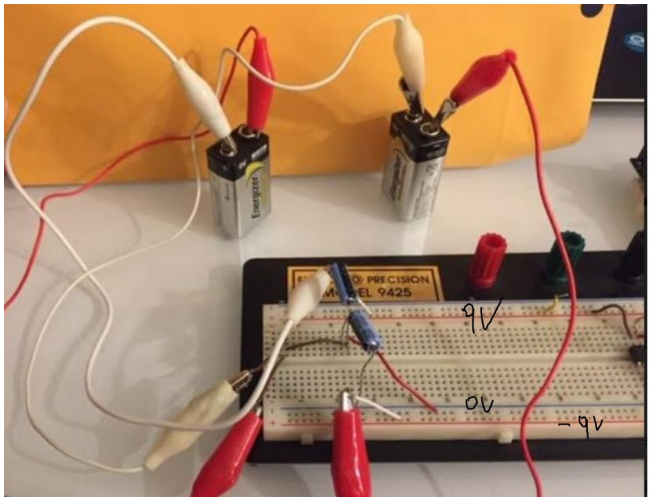
Some loads (like speakers, light bulbs) have a small resistance. So we need to fix this problem.

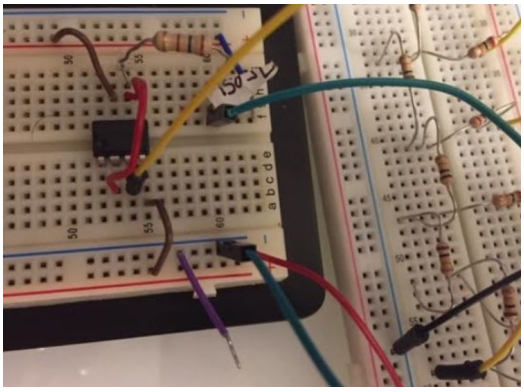
To solve this problem we use a buffer like the IC 741 (see lab buffer). The output is not distorted and the output of the circuit sees an " infinite" resistance. The load sees a small resistance and enough current to run a 3V light bulb for example.

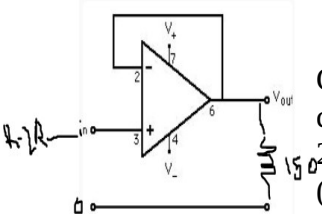
Have your dual power supply ready on the same breadboard as the 741 op amp. First make sure you get 9V on a red line and -9V on the opposite one and the ground on a blue line. Check the voltages with a voltmeter. See below. Then you can power the op amp. +V is connected to 9V and -V to -9V











Connect the inverting input to the output. Connect the output of the 2-2R circuit to the non-inverting input of the op-amp. And Connect the ground of the 2 boards together. The Gnd is connected to the Arduino Gnd. Connect the load (150 ohms) at the output of the op-amp and to the Gnd. With the oscilloscope see if now you get a sine wave with a larger amplitude (compare to when the load is directely connected to the output of the R-2R circuit).

Conclusion: