PH435 Lab 2

Sankalp Gambhir
180260032

September 13, 2020

A. Simple DAC

A.1) To generate a DAC from the circuit:

1) **2.** A low value and a high value.
2) $R_x, R_x, 2 \cdot R_x$ **from the left.** Actual value does not matter, unless considering a load connected to this circuit. Generally I would use very large resistors and connect a buffer after to minimize power draw.
3) $V_{out} = V_{ref} \cdot \frac{B_0}{2}$. The resistors simply act as a divider.
4) **2.5V.**
5) **0V.**

A.2) $V_{out} = V_{ref} \cdot \left(\frac{B_3}{2} + \frac{B_2}{4} + \frac{B_1}{8} + \frac{B_0}{16} \right)$. See Figure 1 for the circuit.

It is a good idea to restrict to only two (proportional) values as multiple independent values would generate a much higher error. It is relatively easier to manufacture two matched values to a good accuracy.

The impedances are:

- $\frac{128}{43} \cdot R_x$ looking in from B_0.
- R_x looking in from V_{out}.

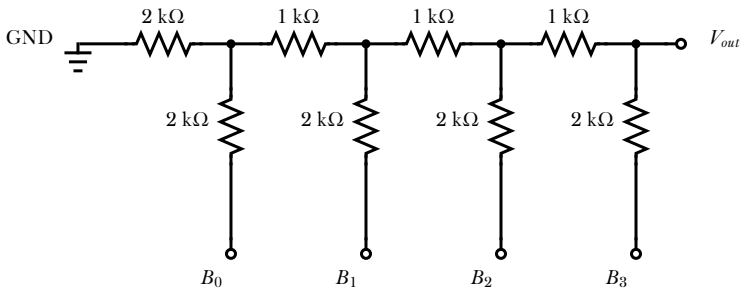

Figure 1: 4-bit DAC design.

B. Design a 4-bit DAC

1) Setup done.
2) $R_x = 1k\Omega$. The resistors simply need to be large enough to keep the current under the Arduino limits and be lower than the load voltage for Part E.
3) See Figure 2.

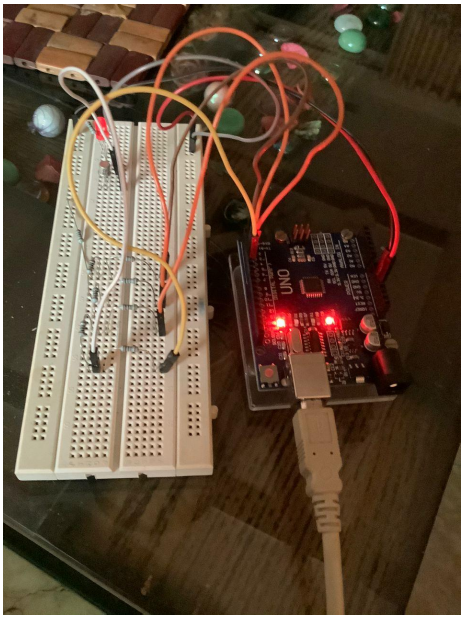

Figure 2: Connected DAC.

C. Test DAC with Arduino

The code uploaded to the Arduino may be found in the Appendix.

C.1) The prediction is as in Figure 3.

C.2) The circuit was compeleted, as previously attached in Figure 2. A video may be found in the folder linked in Part E.

D. Improve DAC control from Arduino

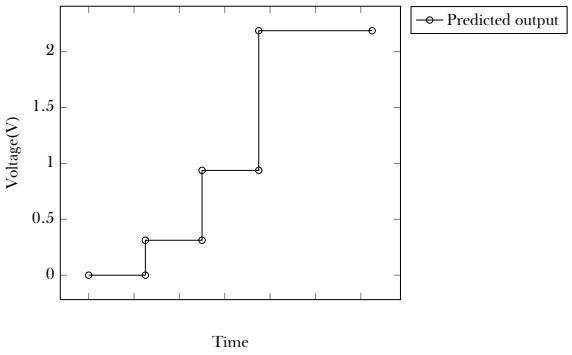

Figure 3: Predicted output

Checked documentation, updated code to use `PORTD`.

E. Demonstrate DAC analog output voltage with LED

E.1) The circuit design for a VCCS, acting here as a buffer, is as in Figure 4.

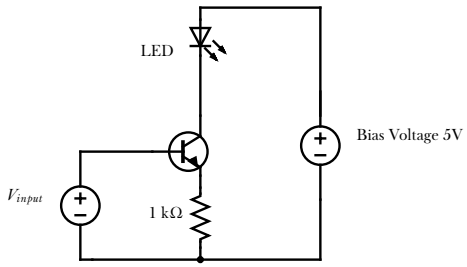

Figure 4: Voltage Controlled Current Source Circuit.

E.2) The simulation and demonstration videos may be found in the Google Drive folder linked here.

Appendix

DAC

The code uploaded to the Arduino for testing the DAC alone. Delays were introduced manually to see the changes in the output clearly. It can also be seen without the delay, by keeping the print there, but this is just clearer.

```
void setup(){
    Serial.begin(1000000); // why not
    for(int i = 2; i < 6; i++){
        pinMode(i, OUTPUT);
    }
}

void loop(){
    for(int i = 2; i < 5; i++){
        digitalWrite(i, HIGH);
        delay(500); // manually add delay
        Serial.println(5 * (analogRead() / 1023.0)); // measured Vout in Volts
    }
}
```

VCCS

The code uploaded to the Arduino for testing the DAC and the VCCS. We iterate from a value corresponding to a value below threshold to one well above it, while making sure to not modify `Rx/Tx` pins (0 and 1).

```
/*
Lab 3
DAC
*/

// variables
uint8_t temp;

// functions

void setup(){
    Serial.begin(1000000); // why not
    DDRD = 0b11111110; // pins 1-7 as output. 0 is Tx
    temp = 0b00000000; // initialise as low
    PORTD = temp;
}

void loop(){
    // go from 0.9 to 2.2V
    // for temp from 12 to 28 adding 4, in binary to be explicit about pins
    for(temp = 0b00001100; temp <= 0b00011100; temp += 0b100){
      PORTD = temp;
      delay(500);
    }
}
```