Su22-ENGR-40M-01 Prelab 2b

Jannah Sabic El-Rayess

TOTAL POINTS

10.3 / 12

QUESTION 1

1 P2 1 / 1

✓ **- 0 pts Correct-- should be around 30 Ohm (accept 20-40 Ohm)**

- 0.5 pts Value too small

- 0.5 pts Value too large

- 1 pts No work/Incorrect

QUESTION 2

2 P3 1 / 1

✓ **- 0 pts Correct-- should be around 30 Ohm (accept 20-40 Ohm)**

- 0.5 pts Value too small

- 0.5 pts Value too large

- 1 pts No work/Incorrect

QUESTION 3

3 P4 1 / 1

✓ **- 0 pts Correct**

- 0.5 pts Model should include internal resistance of arduino

- 0.1 pts Need to label resistors

- 1 pts No work/Incorrect/Late

- 0.5 pts Motor should be modeled with a resistor

- 0.5 pts Resistors need to be connected to Vdd and GND

QUESTION 4

4 P5 0.8 / 1

- 0 pts Correct-- should be around 60mA (accept 50-80mA), student also correctly notes that motor needs more current when load is greater (like starting or pushing button)

- 0 pts Carried error from resistances in P2 + P3

- 0.2 pts motor current >100mA when you try to stop it (so not enough current to flip toggle switch)

✓ **- 0.2 pts Correct calculation, but unclear/incorrect conclusion**

- 0.4 pts incorrect current calculation-- forgot to add all 3 resistances

- 0.4 pts incorrect current calculation--other

- 0.4 pts incorrect reasoning-- references low resistances causing high currents (short circuit) or current too much for motor or no reference to motor current requirements

- 0.4 pts Incorrectly concluded that the design works, but correct current calculation

- 1 pts No work/incorrect

- 0.2 pts Small calculation mistake, but correct conclusion.

QUESTION 5

5 P6 0.5 / 1

- 0 pts Correct--a) and b) should be less than an Ohm and c) and d) should be high resistance (order MOhm)

- 0.1 pts forgot to subtract probe resistance

- 0.2 pts Measurement error

- 0.3 pts c) and d) said ohmmeter can't measure the resistance but did not specify that this is because the resistance is very large (> MOhms)

- 0.5 pts a) and b) should be near 0 Ohm

✓ **- 0.5 pts c) and d) should be high resistance (order MOhm or max out DMM)**

- 0.5 pts b) and d) should mimic the measurement before them

- 0.3 pts measured around 500 Ohm instead of very high resistance in off state

- 1 pts no work/incorrect

- 0.25 pts d) should be high resistance (order MOhm or max out DMM)

- **0.25 pts** b) should be low resistance (< 1ohm)

QUESTION 6

6 P7 0 / 1

- **0 pts** Correct---- a) and b) should be high resistance (order MOhm) and c) and d) should be around 0.2 Ohm
- **0.1 pts** forgot to subtract probe resistance
- ✓ **- 0.5 pts a) and b) should be high resistance (order MOhm)**
- ✓ **- 0.5 pts c) and d) should be near 0 Ohm**
- **0.5 pts** b) and d) should mimic the measurement before them
- **1 pts** no work/incorrect
- **0.3 pts** Measured near 500 ohm for a and b
- **0.2 pts** Slight measurement error

QUESTION 7

7 P8 2 / 2

- ✓ **- 0 pts Correct**
- **0.3 pts** Error(s) see comments
- **0.2 pts** lots of crossing wires makes layout hard to follow
- **1 pts** Switched PMOS and NMOS
- **1 pts** nMOS and pMOS with same gate should connect to same side of motor
- **1 pts** Gates connected to motor
- **1 pts** all gates need to attach to an input
- **1.5 pts** incomplete- partially attempted
- **0.3 pts** did not complete in EveryCircuit
- **2 pts** No work/Incorrect
- **1 pts** missing motor
- **0.4 pts** Power source not connected properly

QUESTION 8

8 P9 2 / 2

- ✓ **- 0 pts Correct**
- **0.1 pts** Minor error
- **0.2 pts** only need 1 half of toggle switch
- **0.2 pts** use pull-up function of input pins (then switches can be connected to ground at other end instead of 5V)
- **0.2 pts** switches inappropriately connected to two different input pins -- switches should have one end connected to ground
- **0.5 pts** switches connected to input incorrectly
- **1 pts** Motor still driven directly by switches instead of by arduino
- **1 pts** Arduino and switch missing
- **1 pts** need transistor driver for motor
- **0.5 pts** Switches connected incorrectly
- **0.3 pts** Motor incorrectly attached to transistors
- **0.5 pts** Switches are missing
- **1 pts** Arduino not implemented
- **0.3 pts** Diagram is difficult to read
- **2 pts** no work or incorrect
- **0.2 pts** Limit Switch Missing
- **0.2 pts** Missing output pin connection(s) to motor driver circuit
- **0 pts** Motor missing
- **0.3 pts** Diagram missing pin labels; some connections missing

QUESTION 9

9 P10 1 / 1

- ✓ **- 0 pts Correct**
- **0.1 pts** Error(s) see comments
- **0.1 pts** code hard to read
- **0.3 pts** did not use digitalWrite correctly
- **0.3 pts** missing pin initiation in void setup()
- **0.3 pts** need to set both output pins when changing state of motor
- **0.2 pts** pins should be set to output mode
- **0.5 pts** code does not control motor (need to turn motor on and off)
- **0.5 pts** Reverse not implemented correctly
- **0.5 pts** Forward/stop not implemented correctly
- **1 pts** No work or incorrect
- **0.2 pts** Uses Digital Pins 0 or 1. These pins are the serial pins so using them in your circuit will break serial communication.
- **0.3 pts** Setup loop not shown

QUESTION 10

10 P11 **1 / 1**

✓ **- 0 pts Correct**

- 0.1 pts minor error see comments

- 0.1 pts Incomplete

- 0.1 pts inefficient code/hard to read

- 0.3 pts Either toggle switch or limit switch missing

- 0.2 pts Uses input instead of Input_pullup

- 0.4 pts Setup missing / Incorrect

- 1 pts No response/Incorrect

- 0.2 pts One switch has incorrect state at input

- 0.2 pts Uses Digital Pins 0 or 1. These pins are the serial pins so using them in your circuit will break serial communication.

- 0.5 pts No use of Serial Monitor to read state

- 0.25 pts Used output instead of input_pullup

- 0.2 pts Used input instead of input_pullup

- 1 pts Code missing

To get started, connect your Arduino to a USB port, and try loading one of the sample programs to make sure everything is in order.

P1: Choose your two favorite I/O pins, but don't use pins 0, 1 or 13. Write a simple program that configures both of them to be outputs using `pinMode()`, then drives one of those pins to `HIGH` and the other to `LOW`. Load the program into your Arduino.

This program should be four lines long. You don't need to submit it with your prelab.

P2: Measure the output resistance of the I/O pin that you drove high.

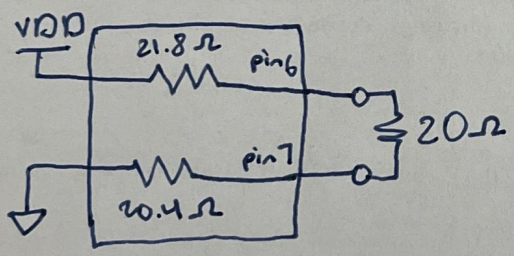
21.8 Ω

P3: Measure the output resistance of the I/O pin that you drove low.

20.4 Ω

A naïve way to control the direction of your useless box's motor would be to connect it directly between two I/O pins. To drive the motor one way, you set one pin to `HIGH` and the other to `LOW`; to drive it the other way, you set the pins the other way round.

P4: Draw a model of the circuit that would be formed if you connected the motor directly between two I/O pins, one of which was driven to high and the other of which was driven to low. Model the motor as a 20 Ω resistor.[a]



[a] It's not actually a resistor, but this gives us some idea of the load it provides when it's running continuously.

P5: Use the model you just drew to estimate the maximum current that this design would be able to provide to the motor. Compare this to your measurement of the highest current you found the

$R_T = 21.8\,\Omega + 20.4\,\Omega + 20\,\Omega$

$R_T = 62.2\,\Omega$

$V = iR$

$5V = i\,62.2\,\Omega$

$i = 80.4\,mA$

80.4mA

1 P2 **1 / 1**

✓ **- 0 pts** **Correct-- should be around 30 Ohm (accept 20-40 Ohm)**

- 0.5 pts Value too small

- 0.5 pts Value too large

- 1 pts No work/Incorrect

To get started, connect your Arduino to a USB port, and try loading one of the sample programs to make sure everything is in order.

P1: Choose your two favorite I/O pins, but don't use pins 0, 1 or 13. Write a simple program that configures both of them to be outputs using `pinMode()`, then drives one of those pins to `HIGH` and the other to `LOW`. Load the program into your Arduino.

This program should be four lines long. You don't need to submit it with your prelab.

P2: Measure the output resistance of the I/O pin that you drove high.

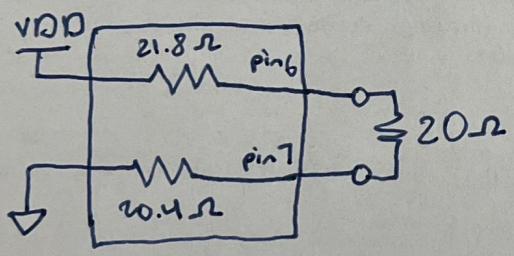
21.8 Ω

P3: Measure the output resistance of the I/O pin that you drove low.

20.4 Ω

A naïve way to control the direction of your useless box's motor would be to connect it directly between two I/O pins. To drive the motor one way, you set one pin to `HIGH` and the other to `LOW`; to drive it the other way, you set the pins the other way round.

P4: Draw a model of the circuit that would be formed if you connected the motor directly between two I/O pins, one of which was driven to high and the other of which was driven to low. Model the motor as a 20 Ω resistor.[a]



[a] It's not actually a resistor, but this gives us some idea of the load it provides when it's running continuously.

P5: Use the model you just drew to estimate the maximum current that this design would be able to provide to the motor. Compare this to your measurement of the highest current you found the

$R_T = 21.8\,\Omega + 20.4\,\Omega + 20\,\Omega$

$R_T = 62.2\,\Omega$

$V = iR$

$5V = i\,62.2\,\Omega$

$i = 80.4\,mA$

80.4mA

2 P3 **1 / 1**

✓ **- 0 pts** **Correct-- should be around 30 Ohm (accept 20-40 Ohm)**

- 0.5 pts Value too small

- 0.5 pts Value too large

- 1 pts No work/Incorrect

To get started, connect your Arduino to a USB port, and try loading one of the sample programs to make sure everything is in order.

P1: Choose your two favorite I/O pins, but don't use pins 0, 1 or 13. Write a simple program that configures both of them to be outputs using `pinMode()`, then drives one of those pins to `HIGH` and the other to `LOW`. Load the program into your Arduino.

This program should be four lines long. You don't need to submit it with your prelab.

P2: Measure the output resistance of the I/O pin that you drove high.

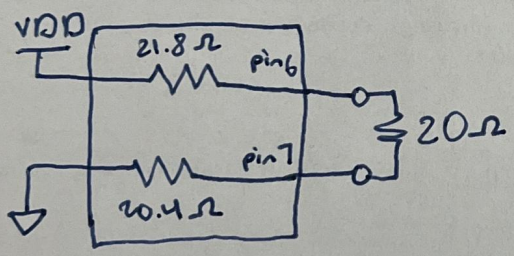
21.8 Ω

P3: Measure the output resistance of the I/O pin that you drove low.

20.4 Ω

A naïve way to control the direction of your useless box's motor would be to connect it directly between two I/O pins. To drive the motor one way, you set one pin to `HIGH` and the other to `LOW`; to drive it the other way, you set the pins the other way round.

P4: Draw a model of the circuit that would be formed if you connected the motor directly between two I/O pins, one of which was driven to high and the other of which was driven to low. Model the motor as a 20 Ω resistor.[a]



[a] It's not actually a resistor, but this gives us some idea of the load it provides when it's running continuously.

P5: Use the model you just drew to estimate the maximum current that this design would be able to provide to the motor. Compare this to your measurement of the highest current you found the

$R_T = 21.8\,\Omega + 20.4\,\Omega + 20\,\Omega$

$R_T = 62.2\,\Omega$

$V = iR$

$5V = i\,62.2\,\Omega$

$i = 80.4\,mA$

80.4mA

3 P4 1 / 1

✓ **- 0 pts** **Correct**

- 0.5 pts Model should include internal resistance of arduino

- 0.1 pts Need to label resistors

- 1 pts No work/Incorrect/Late

- 0.5 pts Motor should be modeled with a resistor

- 0.5 pts Resistors need to be connected to Vdd and GND

To get started, connect your Arduino to a USB port, and try loading one of the sample programs to make sure everything is in order.

P1: Choose your two favorite I/O pins, but don't use pins 0, 1 or 13. Write a simple program that configures both of them to be outputs using `pinMode()`, then drives one of those pins to `HIGH` and the other to `LOW`. Load the program into your Arduino.

This program should be four lines long. You don't need to submit it with your prelab.

P2: Measure the output resistance of the I/O pin that you drove high.

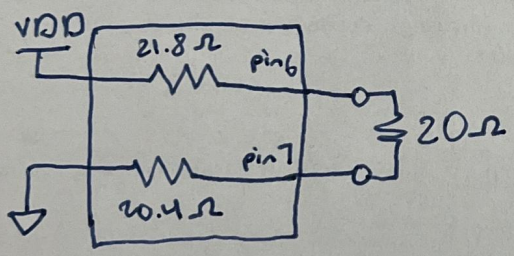
21.8 Ω

P3: Measure the output resistance of the I/O pin that you drove low.

20.4 Ω

A naïve way to control the direction of your useless box's motor would be to connect it directly between two I/O pins. To drive the motor one way, you set one pin to `HIGH` and the other to `LOW`; to drive it the other way, you set the pins the other way round.

P4: Draw a model of the circuit that would be formed if you connected the motor directly between two I/O pins, one of which was driven to high and the other of which was driven to low. Model the motor as a 20 Ω resistor.[a]



[a] It's not actually a resistor, but this gives us some idea of the load it provides when it's running continuously.

P5: Use the model you just drew to estimate the maximum current that this design would be able to provide to the motor. Compare this to your measurement of the highest current you found the

$R_T = 21.8\,\Omega + 20.4\,\Omega + 20\,\Omega$

$R_T = 62.2\,\Omega$

$V = iR$

$5V = i\,62.2\,\Omega$

$i = 80.4\,mA$

80.4mA

motor could draw, from prelab 2a. Would this design work? Why, or why not?

No because 80.4mA is way too low as it is much less than the max current measured in 2a.

2.2 MOS transistor resistance

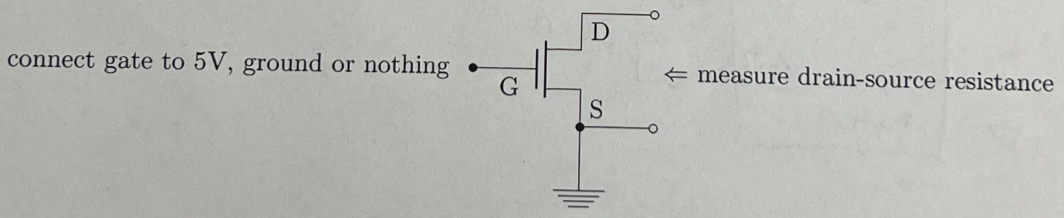
Our goal in this section is to measure the *on resistance* and *off resistance* of your power MOS transistors.

To do this, you'll need a 5 V power supply. The easiest way to do this is to use the USB cable you cut in half during lab 1 to power the breadboard directly from any USB charging port (*e.g.* on your computer, a phone charger or your solar charger from lab 1). Alternatively, you could plug your Arduino into the breadboard, connect the Arduino to your computer, and connect the VDD and GND pins of the Arduino to the side rails on the breadboard.

Some measurement tips:

- All wires have some resistance, including the leads of your multimeter. This means that even when you touch the two probes together, you won't measure zero resistance. Since this is typically less than 1 Ω, it can usually be ignored. But if you are measuring a small resistance, you should measure this baseline resistance first, and subtract it from the resistance you measure.
- Remember, your multimeter needs *firm* connections to measure accurately. You'll find this a lot easier if you plug your transistor into one of your breadboards, and use the flexible jumper wires and alligator clips in your lab kit. Make sure you plug your transistor in so that its legs are in *different* rows!

Plug your nMOS power transistor (RFP12N10L) into the breadboard. To measure the on and off resistances of the nMOS transistor, we'll want to connect the source to ground, so that we can control the gate-source voltage. The gate connects to 5V, GND or nothing (floating), depending on the measurement. Do **not** connect the drain to anything other than the ohmmeter.



P6: What is the drain-source resistance of the nMOS transistor when the gate is, measured *in this order*: (a) connected to 5V, (b) floating, (c) connected to ground, and (d) floating?

(a) 0.8V

(b) 2.6V

(c) 153.5V

(d) 147.6V

Now, do the same thing with the pMOS (NDP6020P). Recall that pMOS sources connect to V_{DD}.

4 P5 **0.8 / 1**

- **0 pts** Correct-- should be around 60mA (accept 50-80mA), student also correctly notes that motor needs more current when load is greater (like starting or pushing button)
- **0 pts** Carried error from resistances in P2 + P3
- **0.2 pts** motor current >100mA when you try to stop it (so not enough current to flip toggle switch)

✓ - **0.2 pts** **Correct calculation, but unclear/incorrect conclusion**

- **0.4 pts** incorrect current calculation-- forgot to add all 3 resistances
- **0.4 pts** incorrect current calculation--other
- **0.4 pts** incorrect reasoning-- references low resistances causing high currents (short circuit) or current too much for motor or no reference to motor current requirements
- **0.4 pts** Incorrectly concluded that the design works, but correct current calculation
- **1 pts** No work/incorrect
- **0.2 pts** Small calculation mistake, but correct conclusion.

motor could draw, from prelab 2a. Would this design work? Why, or why not?

No because 80.4mA is way too low as it is much less than the max current measured in 2a.

2.2 MOS transistor resistance

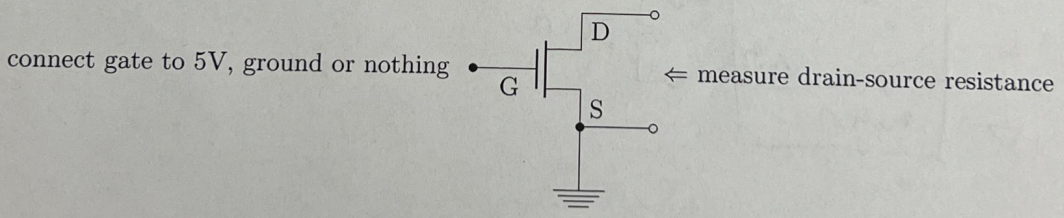
Our goal in this section is to measure the *on resistance* and *off resistance* of your power MOS transistors.

To do this, you'll need a 5 V power supply. The easiest way to do this is to use the USB cable you cut in half during lab 1 to power the breadboard directly from any USB charging port (*e.g.* on your computer, a phone charger or your solar charger from lab 1). Alternatively, you could plug your Arduino into the breadboard, connect the Arduino to your computer, and connect the VDD and GND pins of the Arduino to the side rails on the breadboard.

Some measurement tips:

- All wires have some resistance, including the leads of your multimeter. This means that even when you touch the two probes together, you won't measure zero resistance. Since this is typically less than 1 Ω, it can usually be ignored. But if you are measuring a small resistance, you should measure this baseline resistance first, and subtract it from the resistance you measure.
- Remember, your multimeter needs *firm* connections to measure accurately. You'll find this a lot easier if you plug your transistor into one of your breadboards, and use the flexible jumper wires and alligator clips in your lab kit. Make sure you plug your transistor in so that its legs are in *different* rows!

Plug your nMOS power transistor (RFP12N10L) into the breadboard. To measure the on and off resistances of the nMOS transistor, we'll want to connect the source to ground, so that we can control the gate-source voltage. The gate connects to 5V, GND or nothing (floating), depending on the measurement. Do **not** connect the drain to anything other than the ohmmeter.



P6: What is the drain-source resistance of the nMOS transistor when the gate is, measured *in this order*: (a) connected to 5V, (b) floating, (c) connected to ground, and (d) floating?

(a) 0.8V

(b) 2.6V

(c) 153.5V

(d) 147.6V

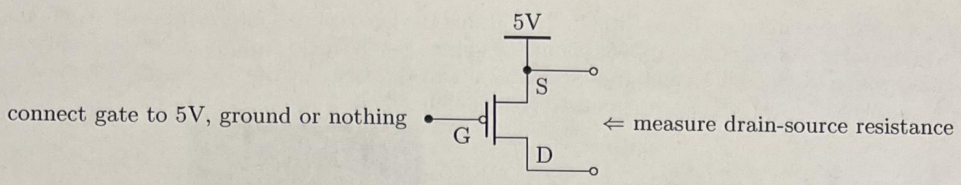
Now, do the same thing with the pMOS (NDP6020P). Recall that pMOS sources connect to V_{DD}.

5 P6 **0.5 / 1**

- **0 pts** Correct--a) and b) should be less than an Ohm and c) and d) should be high resistance (order MOhm)
- **0.1 pts** forgot to subtract probe resistance
- **0.2 pts** Measurement error
- **0.3 pts** c) and d) said ohmmeter can't measure the resistance but did not specify that this is because the resistance is very large (> MOhms)
- **0.5 pts** a) and b) should be near 0 Ohm

✓ - **0.5 pts** **c) and d) should be high resistance (order MOhm or max out DMM)**

- **0.5 pts** b) and d) should mimic the measurement before them
- **0.3 pts** measured around 500 Ohm instead of very high resistance in off state
- **1 pts** no work/incorrect
- **0.25 pts** d) should be high resistance (order MOhm or max out DMM)
- **0.25 pts** b) should be low resistance (< 1ohm)



P7: What is the drain-source resistance of the pMOS transistor when the gate is, measured *in this order*: (a) connected to 5V, (b) floating, (c) connected to ground and (d) floating?

(a) 167.6 V (c) 168.1 V

(b) 168.0 V (d) 168.8 V

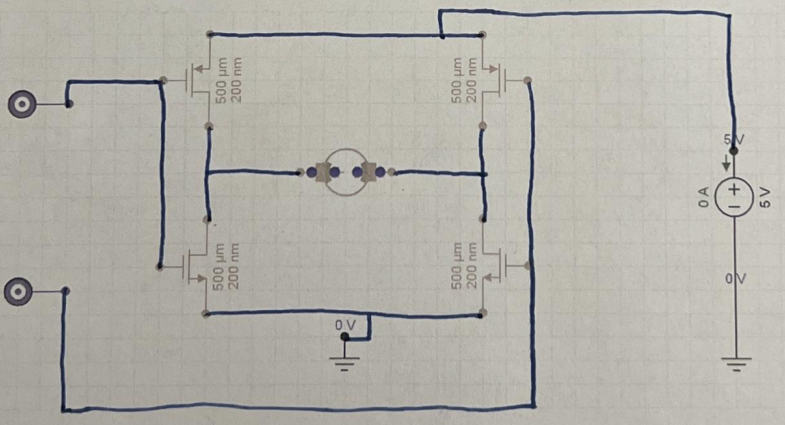
2.3 Designing the circuit

Having established that we need these "power" transistors to interface between the Arduino and the motor, let's design the circuit that drives the motor. The motor needs to be able to do *three* things: go forward, go backward, and stop. With the help of a motor driver circuit, by changing the state of two Arduino output pins, you should be able get the motor to do each of those three actions.

To get started, search for this starter circuit on EveryCircuit using the search term "ENGR 40M prelab 2b starter" (or just "ENGR 40M"), or use this URL: `http://everycircuit.com/circuit/5069979055816704`.

pMOS

nMOS



- The two circles on the left are "logic sources", which represent your Arduino output pins. If you click on them, they'll change from 0 (low) to 1 (high) and vice versa.
- The circular item in the middle is the motor. When current goes through it, it animates either clockwise or counter-clockwise.
- The 5 V source on the right represents V_{DD}.

6 P7 **0 / 1**

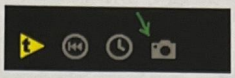
- **0 pts** Correct---- a) and b) should be high resistance (order MOhm) and c) and d) should be around 0.2 Ohm
- **0.1 pts** forgot to subtract probe resistance

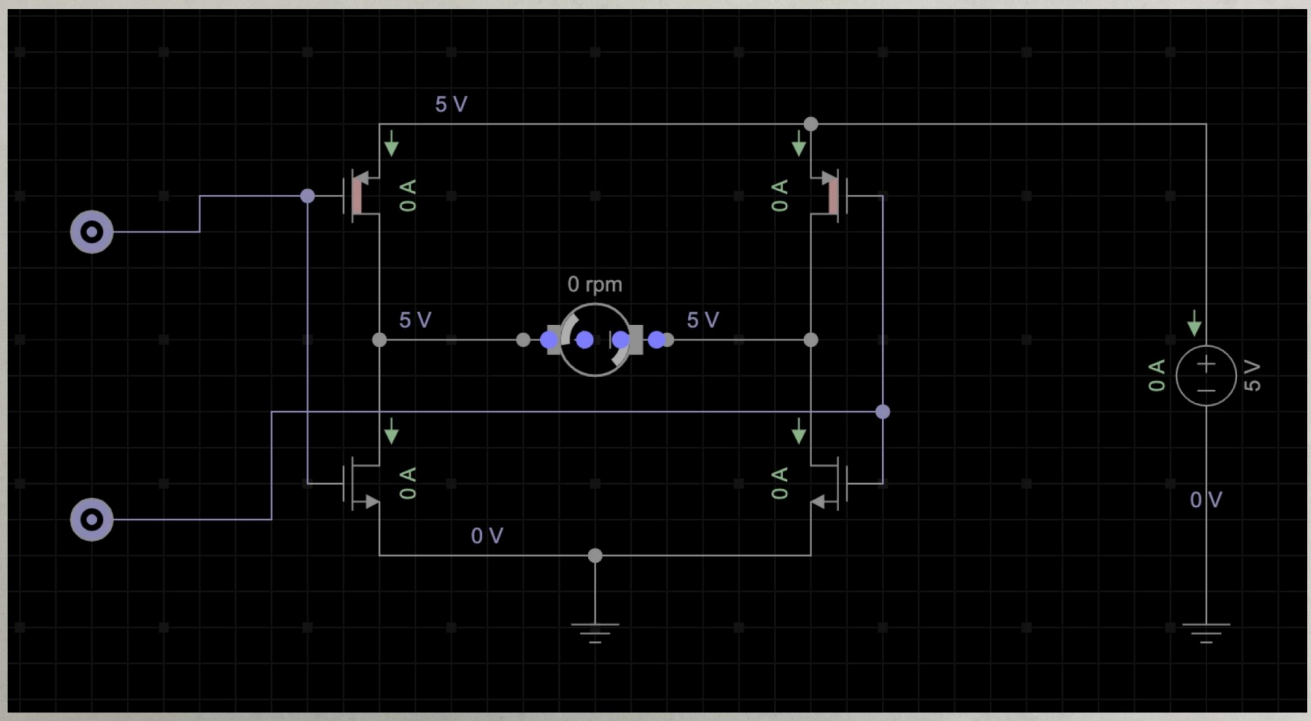
✓ **- 0.5 pts a) and b) should be high resistance (order MOhm)**

✓ **- 0.5 pts c) and d) should be near 0 Ohm**

- **0.5 pts** b) and d) should mimic the measurement before them
- **1 pts** no work/incorrect
- **0.3 pts** Measured near 500 ohm for a and b
- **0.2 pts** Slight measurement error

P8: Complete the circuit on EveryCircuit. By clicking the logic sources into appropriate combinations (in the middle of a simulation), you should be able to get the motor to spin clockwise, spin counter-clockwise and stop. It should also be impossible to short-circuit the 5 V source. Please attach a screenshot of your completed circuit to your prelab submission.

Tip: This button saves a screenshot of the circuit: 



7 P8 2 / 2

✓ **- 0 pts Correct**

- 0.3 pts Error(s) see comments

- 0.2 pts lots of crossing wires makes layout hard to follow

- 1 pts Switched PMOS and NMOS

- 1 pts nMOS and pMOS with same gate should connect to same side of motor

- 1 pts Gates connected to motor

- 1 pts all gates need to attach to an input

- 1.5 pts incomplete- partially attempted

- 0.3 pts did not complete in EveryCircuit

- 2 pts No work/Incorrect

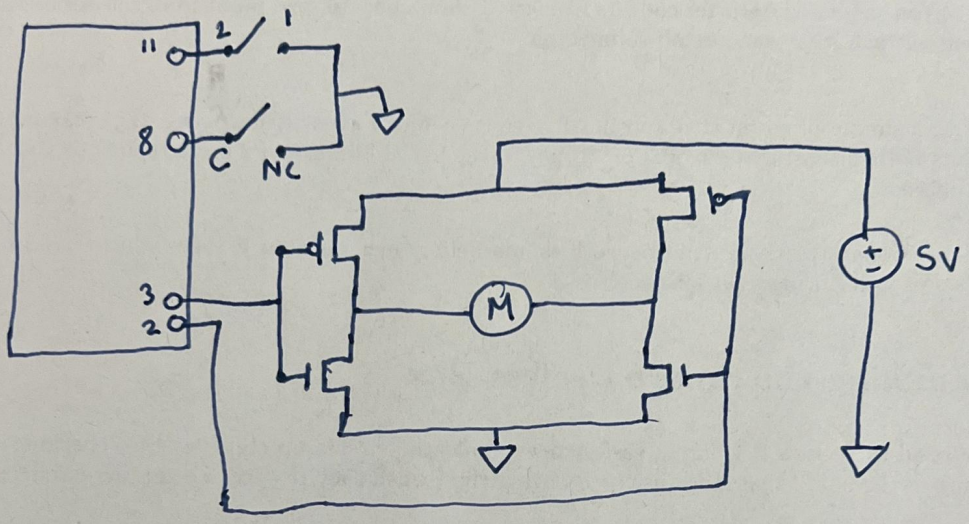
- 1 pts missing motor

- 0.4 pts Power source not connected properly

That's most of the work needed to design the circuit. Now we just need to add the switches and be a bit more explicit about how the Arduino fits in.

P9: Draw a complete schematic of the entire circuit you need to build in the lab to make your More Awesome Useless Box. This should include the motor driver, the two switches in your basic useless box, the Arduino, batteries and it should indicate which Arduino pins the motor driver and switches connect to.

Please submit this with your prelab, and bring a copy to lab, since it will guide you as you make your More Awesome Useless Box a reality.



2.4 Programming the Arduino

It makes your life easier as an engineer—and as a maker—if you build things up step by step. This makes debugging much easier: if something doesn't work, you know it should be the thing you just added. As far as the Arduino goes, here's our plan of attack:

1. First, we'll verify that we can read our switches correctly (without touching the motor).
2. Then, we'll verify that we can drive our motor correctly (without touching the switches).
3. Then, we'll write a program that replicates our original useless box, incorporating both the switches and the motor.
4. Finally, we'll extend the program to make the useless box more awesome.

Before you come to lab, we ask you to write two simple programs, for steps 1 and 2 respectively. Once you come to lab and have built the improved box, we'll continue with step 3. Your homework for the week after the lab is to do step 4.

8 P9 2 / 2

✓ **- 0 pts Correct**

- 0.1 pts Minor error

- 0.2 pts only need 1 half of toggle switch

- 0.2 pts use pull-up function of input pins (then switches can be connected to ground at other end instead of 5V)

- 0.2 pts switches inappropriately connected to two different input pins -- switches should have one end connected to ground

- 0.5 pts switches connected to input incorrectly

- 1 pts Motor still driven directly by switches instead of by arduino

- 1 pts Arduino and switch missing

- 1 pts need transistor driver for motor

- 0.5 pts Switches connected incorrectly

- 0.3 pts Motor incorrectly attached to transistors

- 0.5 pts Switches are missing

- 1 pts Arduino not implemented

- 0.3 pts Diagram is difficult to read

- 2 pts no work or incorrect

- 0.2 pts Limit Switch Missing

- 0.2 pts Missing output pin connection(s) to motor driver circuit

- 0 pts Motor missing

- 0.3 pts Diagram missing pin labels; some connections missing

```
void setup() {
  Serial.begin(115200);
  Serial.println("Setup completed");
  pinMode(6, OUTPUT);
  pinMode(7, OUTPUT);
}

/*
 * Function: loop
 * ---------------
 * This will run continuously after setup() finishes running.
 */
void loop() {
  byte user_input = 0;

  if (Serial.available() > 0) {
    user_input = Serial.read();

    if (user_input == 'f') {
      Serial.println("Forward");
      digitalWrite(6, LOW);
      digitalWrite(7, HIGH);
    } else if (user_input == 'r') {
      Serial.println("Reverse");
      digitalWrite(6, HIGH);
      digitalWrite(7, LOW);
    } else if (user_input == 's') {
      Serial.println("Stop");
      digitalWrite(6, LOW);
      digitalWrite(7, LOW);
    } else {
      Serial.println("Invalid character");
    }
  }
}
```

9 P10 1 / 1

✓ **- 0 pts Correct**

- 0.1 pts Error(s) see comments

- 0.1 pts code hard to read

- 0.3 pts did not use digitalWrite correctly

- 0.3 pts missing pin initiation in void setup()

- 0.3 pts need to set both output pins when changing state of motor

- 0.2 pts pins should be set to output mode

- 0.5 pts code does not control motor (need to turn motor on and off)

- 0.5 pts Reverse not implemented correctly

- 0.5 pts Forward/stop not implemented correctly

- 1 pts No work or incorrect

- 0.2 pts Uses Digital Pins 0 or 1. These pins are the serial pins so using them in your circuit will break serial communication.

- 0.3 pts Setup loop not shown

```
void setup() {
  Serial.begin(115200);
  Serial.println("Setup completed");
  pinMode(11, INPUT_PULLUP);
  pinMode(8, INPUT_PULLUP);
}

/*
 * Function: loop
 * ---------------
 * This will run continuously after setup() finishes running.
 */
void loop() {
  if (Serial.available() > 0) {
    if (digitalRead(11) == HIGH) {
      Serial.println("Toggle: high");
    }
    if (digitalRead(11) == LOW) {
      Serial.println("Toggle: low");
    }
    if (digitalRead(8) == HIGH) {
      Serial.println("Limit: high");
    }
    if (digitalRead(8) == LOW) {
      Serial.println("Limit: low");
    }
  }
}
```

10 P11 1 / 1

✓ - 0 pts Correct

- **0.1 pts** minor error see comments
- **0.1 pts** Incomplete
- **0.1 pts** inefficient code/hard to read
- **0.3 pts** Either toggle switch or limit switch missing
- **0.2 pts** Uses input instead of Input_pullup
- **0.4 pts** Setup missing / Incorrect
- **1 pts** No response/Incorrect
- **0.2 pts** One switch has incorrect state at input
- **0.2 pts** Uses Digital Pins 0 or 1. These pins are the serial pins so using them in your circuit will break serial communication.
- **0.5 pts** No use of Serial Monitor to read state
- **0.25 pts** Used output instead of input_pullup
- **0.2 pts** Used input instead of input_pullup
- **1 pts** Code missing