Assignment 2: MOSFETs and Logic Gates
Due date: Tues 09 Feb 2021, noon
ENGN/PHYS 208—Winter 2021
J. Erickson

Background

The invention of the MOSFET opened the flood gates (pun intended, ha!) to modern computer and electronic devices. Thus, one could fairly argue it is the single most important invention of the last century. As we've started to see in class, MOSFETs acting as digital switches are the core element in logic gates. As we will soon see, logic gates are the level of abstraction in building computers that count and remember, or perform other basic logic functions. for example, let's say a piece of hospital equipment is to alert a nurse of doctor if a patient's heart rate *or* temperature increases above a certain allowed limit. Notice, what we just said: the alarm bell must sound (output) if either the heart rate monitor OR the thermometer (inputs) tells it to. Imagine another scenario, where 2 buttons must be pressed to ignite the launch of a rocket (2 for redundancy and preventing accidental ignition). This is the AND function. So let's explore!

1 Song-writers workshop

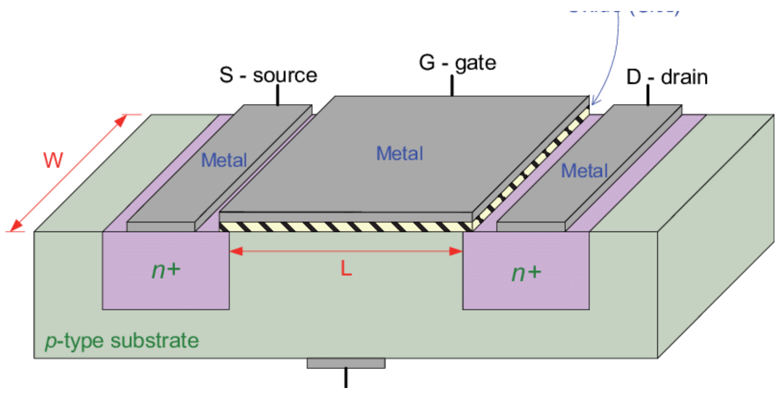

Figure 1: 3-D cross-section view of NMOS transistor. There's a lot going on inside. Writing a poem or song can help sort it all out. Image credit: researchgate.net

There's a lot going on in terms MOSFET semiconductor physics MOSFET and basic principles of operation. To help sort this all out, write lyrics for a poem or song that you perform. Your song or poem should (playfully) detail the basic operational principles of nmos, pmos, and cmos circuits from the electron and hole level up through basic logic gates. To this end, make sure to include and elucidate the following terminology plus any other semiconductor terms you deem appropriate.

- n-type and p-type doping, electrons, holes, semiconductor
- drain, gate, source
- depletion layer, electric field,
- nmos, pmos, cmos
- current flow, channel formation, enhancement mode
- digital switch, logic levels, high, low

Have fun with it and be creative. Post a video or audio file of your world premiere poetry or song performance. *Best performance wins a prize!*

2 Fire in the hole

Figure 2: Smoke/Fire detector. Can you build the circuitry that triggers the alarm? Hint: yes. Image credit: frontpointsecurity.com

In class, we watched a brilliant video describing the fundamental operation of MOSFETs. Toward the end, there is a brief mention that MOSFETs can be used to trigger a fire alarm. "Huh, how exactly would that work?" the curious 208 student asked. This is your chance to dive into some of the details just such a design using an N-channel MOSFET(we have plenty in the lab!) To sense fire danger, you can detect heat or smoke. Thus, you might use a thermistor (heat) or light sensitive resistor (smoke). For an alarm signaling danger, you could use either visual (LED) or audio cue (piezo speaker).

To help get you started, recall that NMOS transistors flow current when the gate-to-source voltage $V_{GS} = V_G - V_S$ is positive. Check out the data sheet here (link) for the ZVN2110A mosfet. Take particular note of page 2, top right plot showing I_{DS} vs V_{DS} for various values of V_{GS}.

For a power supply, you have at your fingertips either +5V and GND (Arduino uno R3), or +3.3V and GND (Adafruit Feather).

Carefully design your fire alarm using the parts listed above, plus any other standard components you might find lying around in the lab. Crucially, you need to pay attention to how you will vary V_{GS} such that the mosfet turns from off to on at the appropriate temperature or smoke level. Therefore, when justifying your design, make sure you refer to data sheets that will indicate resistance values as a function of temperature (thermistor) or light levels (CdS photoresistor). Be sure to show calculations as evidence your design proposal is a sensible one.

What to turn in:

1. Clearly labeled schematic
2. Brief description of circuit operation principle (1 paragraph max)
3. Relevant calculations supporting your design referencing data sheets, as needed
4. Bonus points (+10): Build and test your device in the lab! If you need to modify and adapt along the way, that's totally fine. In your write-up be sure to describe your final working device and justify with calculations how and and why it worked.

3 Digital Safe Key Code

Now it's time to integrate a bunch of logic gates and latches into a more complex device. The idea is to build the infamous Teddy KGB[1] a secure digital safe for all his chips and fortune-telling Oreos. The basic idea idea of the safe is that you must set digital bits correctly, then turn the key/push the open button. If the digital bits are set properly, then turning the key opens the safe. The safe should have a ***unique code***: It should open for one and only one code. (Otherwise, it remains a fortress for all those delicious Oreos.)

(a) Digital safe from Honeywell.

(b) Teddy KGB, from the movie *Rounders*.

Figure 3: Teddy KGB needs a digital safe. Can you help him?

Specifically, your task is to design a 4-bit (e.g., input = 1011) code based on logic gates, and implement an turn key/open button as well. The output should be a logic of 1 for *only* one set of digital bits, and 0 otherwise. Of course the fewer logic gates you use, the more obvious the solution. On the other hand, Teddy KGB greatly admires elegant solutions—so he places a hard constraint that you may use 6 logic chips maximum.

1. Carefully diagram your circuit schematic
2. Then build it to prove it works as advertise.
   (a) Again, the code should be 4 bits plus 1 "enter/open" button.
   (b) Your design must include green and red LED indicators: a green LED illuminates when the correct code has been entered, red lights up otherwise.
3. We will take turns attempting to crack each others' codes based on the hardware implementation. The one that takes the longest to crack wins a prize (a pack of oreos, of course!)

For all 74*xx* series chips listed above (see caption of Fig. 4), notice that pin outs of make them essentially interchangeable—the power connections, the inputs are on the pins, and the outputs are

[1]Check out the movie Rounders, in particular this famous scene

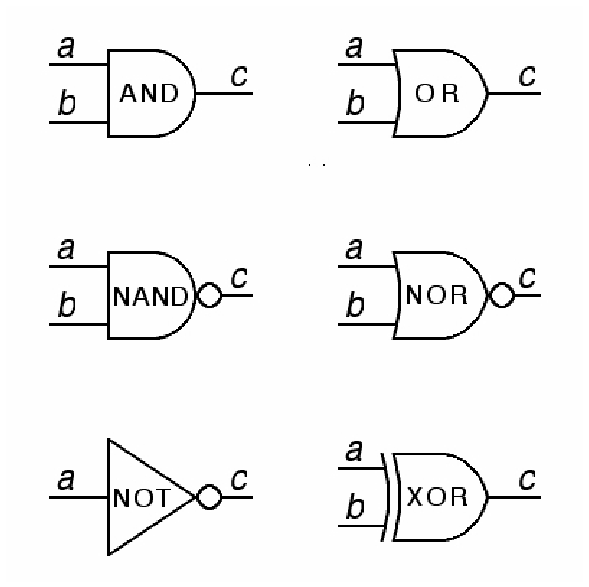

Figure 4: Collection of standard logic gates. AND = 7408. OR = 7423. NAND = 7400. NOR = 7402. NOT = 7404. XOR = 7486.

all on the same pin numbers regardless of logic operation. When implementing/testing logic gates, all inputs should be connected. Inputs left unconnected are termed ***floating***. Floating inputs (and outputs) typically wreak havoc in digital logic systems.

What to submit:

1. Circuit schematic. Use a software package such as Digikey schemeit or EagleCAD.
2. On your schematic, indicate the output for each logic gate, assuming the proper code to enter the safe has been set.
3. Now assume (at least) one of the bits has been set incorrectly. Indicate graphically/show logic gate output leading to safe staying locked (ultimately outputting a 0).
4. Make a brief video of your safe in action. You should walk the viewer through a few of the test cases (there are 16 in all, but something like 4 should suffice to get the point across). You should feel free to (read: are encouraged to) use a logic gate simulator to convince the reader that your system works as planned. One fairly easy and free (!) simulator is logic.ly. You'll be up in running in 20 min.