Lab 1 – TV Jammer

*** Please read and understand the “Lab Readiness Prerequisites” and “Lab Etiquette and Procedures” documents before starting this lab ***

1. Introduction

In this lab you will build the core pieces of a TV jammer. By “jammer,” we mean a near-universal remote that can be used to turn on or off most televisions (this is based upon the TV-B-Gone device: http://en.wikipedia.org/wiki/TV-B-Gone). This lab will involve designing a MOSFET circuit to power an infrared LED bright enough to be picked up by an infrared photodiode in a TV. It will also involve programming your Arduino to control that circuit and pulse the LED. As you complete the lab, you will also be introduced to the primary lab equipment you will be using and learn how to take measurements.

This lab and others that follow require you to insert screen shots of waveforms captured on the oscilloscope into your lab notebook. This requires that you first save the captured waveform to a flash drive (from the scope) so you can transfer it to your computer for printing on the lab’s networked printer. As such, **please always bring a flash drive with you to every lab session.**

Note any instructions within that are **highlighted in red**. You will be expected to confer with a TA on these tasks, observations or problems in order to earn the points available for that section.

You must complete all Pre-Lab work before your scheduled session. Your TAs will not answer questions about your lab until you have completed your Pre-Lab assignment. This policy applies to this lab and all future lab sessions.

2. Recommended Courses

If you’re interested in the contents of this lab, you may also wish to take:

- 18-310 (Fundamentals of Semiconductor Devices): in-depth discussion of the physics and operating principles of the semiconductor devices you will be using today (diodes and MOSFETs)
- 18-320 (Microelectronics): introductory course on microelectronic circuits
- 18-422 (Analysis and Design of Digital Integrated Circuits): taking the concept of a MOSFET as a switch and using it to build large systems
- Build-18 (not a course): hacking!

3. The Arduino Uno Microcontroller

Although this isn’t a programming course, you will need to be able to capture data and do basic signal processing in some labs. For this task you will use an Arduino Uno, an open-source prototyping platform based on the ATmega328 microcontroller. This board has several input and output ports that you may use to read and write digital signals and read analog voltages. You may preview information about the board at http://arduino.cc/en/Main/arduinoBoardUno (pay particular attention to the various types of ports).

To program this board we will help you write C (or C++) code within an integrated development environment (IDE). This environment is accessed from a simple Graphical User Interface (GUI) and allows you to develop “sketches”. A *sketch* is a project containing one or more code files that may be compiled and uploaded to the Arduino board. This IDE, drivers and initial code libraries are free and available at http://arduino.cc/en/Guide/windows (Windows) and http://arduino.cc/en/Guide/MacOSX (Mac). These links explain how to connect the Arduino, download and install the software and drivers, and run some simple test programs.

The software has been installed and tested on the lab computers, but you may download and install it on your own computer in order to complete the pre-lab assignment. When installing the code on your own computer, make sure you install the drivers as documented on the installation website. Also make sure you associate your Arduino with the correct USB port (see 'Tools → Serial Port' from the Arduino IDE main menu). If one setting doesn't work, then try others.

Compiled code that is uploaded to the Arduino is stored in its nonvolatile memory on the board (so your program is stored even when you remove the power supply). You can reset the Arduino at any time by pushing the small button on the corner of the board. This does not wipe the memory; it only restarts the program stored in nonvolatile memory. On the board, you will see several connectors to which you can connect the following types of signals:

- **Power**: this section is for inputting and outputting voltages for power supplies. The Arduino can be powered through USB, but if you want to use it unplugged you need to supply 5-9 V to the *Vin* terminal. The *5V* and *3V* terminals output regulated DC voltages at their respective levels. **<u>Do not supply power to the *5V* or *3V* terminal</u>** – they are outputs only. The GND pins are, as you guessed it, the reference ground for the entire board.
- **Analog in**: this section is for analog inputs. Anything that isn't a digital signal (high and low voltages) should be plugged into A0-A5. For example, if you want to sample a sine wave from an RLC circuit, you would input the voltage here.
- **Digital**: this section is for digital input or output; in the code, you will designate the pins you use as inputs or outputs. The terminals marked with "~" can be used for pulse width modulation (PWM), which involves switching between high and low voltages with a certain duty cycle (the percentage of time the signal is high). You can read more about PWM online, but you won't directly implement it for the labs in this course.

The ZIP file downloaded for this assignment includes a directory named *tutorial* that includes a commented Arduino code file (*tutorial.ino*) that walks you through how to read digital values (high's/1's and low's/0's) and analog values and how to write digital values to the proper output pins. Read through this program to make sure you understand how everything fits together. If you are already familiar with programming the Arduino, then you may skip ahead to the pre-lab work (next section). Otherwise, breadboard the circuit in Figure 1 and compile and upload the 'tutorial' sketch so you can see it in action.

You will need the following components for the tutorial:

- Your Arduino Uno with USB cable
- Your breadboard with hookup wire
- (2) momentary N.O. push-button switches – return to the drawer, NOT the used parts bin.
- (2) 100 Ω resistors (LED current limiting)
- (2) 1kΩ resistors (pull downs)
- (2) LEDs (any color)
- Speaker (you might need to solder on the wires) – return to the drawer, NOT the used parts bin.
- Potentiometer (500 Ω or greater)

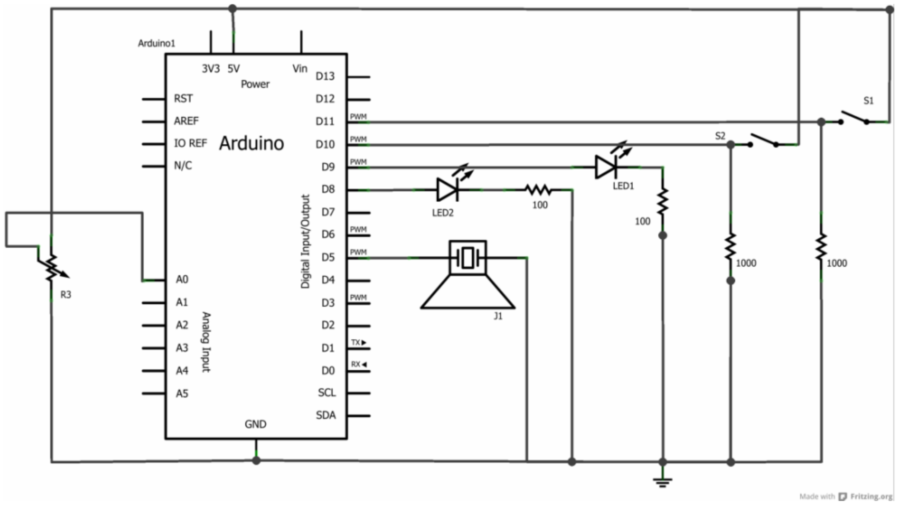

Figure 1 – Schematic of circuit to be used for the Arduino 'tutorial' sketch.

If you have questions, try to find the answers online. There is a large community of Arduino users, and information can be found on http://arduino.cc/en in addition to other sites you can find with a quick internet search. Many example programs (called *sketches*) come loaded with the Arduino IDE (under 'File → Examples', from the main Arduino IDE menu). Please take some time to become familiar with them.

NOTE: When opening an Arduino *sketch*, you must first open the file by the name that corresponds to the sketch. For example, if there is a sketch named *FooBar* that contains several files, then you would open the sketch by navigating to the *FooBar* directory and then opening the file *FooBar.ino*.

4. Pre-Lab Work (15 points)

Before starting any lab work, familiarize yourself with the Measurement Guide in the Appendix to this document. Make sure you understand how each of the instruments work, and ensure that they are working properly. If you need help to operate any of the instruments, ask a TA for help during this lab. **You will be expected to know how to use the instruments without needing help for all the following labs.**

After you have reviewed the information on the Arduino (presented in the previous section) and downloaded and installed the Arduino IDE, complete the following tasks **prior to your lab session**:

1. Using your Arduino Uno and your breadboard, construct the circuit shown in Figure 2. You will also need an LED (any visible color), current limiting resistor (100Ω), momentary pushbutton switch and a pull up resistor (10 kΩ -100 kΩ).

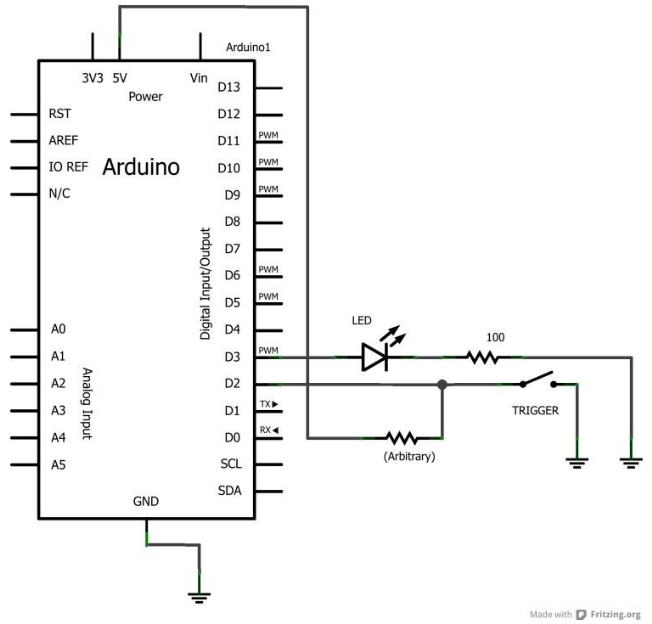

Figure 2 – Pre-lab circuit with trigger switch on digital I/O pin D2 and LED on digital I/O pin D3.

2. Open a new sketch from the Arduino IDE. Next, write code to determine if the button (named TRIGGER, between digital I/O pin D2 and GND) is pushed, and if it is, blink the LED on digital I/O pin D3 (named LED, in series with a current-liming resistor) for one second.

   Use the following pseudo code to guide you in developing your Arduino program. If you are unsure how to convert this to real code, consult the tutorial program presented in the previous section.

```
declare variable LED equal to 3           //  LED  pin  number
declare variable TRIGGER equal to 2       // TRIGGER pin number
void setup() {
    declare  LED  pin  as  output
    declare TRIGGER pin as input
}

void loop() {
    if trigger state is a digital low {                  // button was pushed down
        delay 500 milliseconds                           // wait a bit
        if trigger state has returned to digital high {  // button was let go, ready to blink
            write a digital high to the LED pin          // turn on LED
            delay 1 second
            write a digital low to the LED pin           // turn off LED
        }
    }
    delay 30 milliseconds
}
```

3. Compile and upload your completed program. **A TA will check that this is working during the Pre-Lab checkoff.**

4. Review the IR LED and STQ1NK80ZR datasheets. Use the TO-92 package for the MOSFET.

(a) **What is the maximum amount of current the IR LED can handle?**

(b) **Based on this maximum IR LED current, what is an approximate "ON" voltage for the diode?**

(c) **What is the maximum amount of current the STQ1NK80ZR MOSFET can handle?**

(d) **What is its nominal threshold voltage of the STQ1NK80ZR MOSFET?**

(e) **For a series combination of one resistor and the IR LED, what is the value of series resistance needed to limit the current to the value calculated in part (a)? Be sure to explain**

your choice of resistance. Use V_{dd} = 5V.

5. To illuminate an LED you must apply a voltage across it that is larger than the "turn-on" voltage (remember the current flow through a diode is an exponential relation to the voltage, so what we really mean is the voltage at which an appreciable current flows).

   You could do this by hooking up a battery or power supply to the LED along with a series resistor to limit the current (preventing the diode from burning out). If you built a circuit like this, the LED would constantly be emitting light.

   For the TV jammer, however, we want to pulse the LED so it flashes light. Intuitively, you can see the solution is to insert a switch in series with the LED and resistor. When you want the LED to emit, you close the switch; when you want it to stop, you open the switch. By opening and closing the switch, you get the LED to pulse. What we need is a circuit element we can control with a voltage signal from the Arduino that will behave as a switch.

   In lecture you learned how MOSFETs can be used as digital switches. This transistor can be viewed (at a high level) as a switch for which the voltage at the *gate* terminal creates conduction between the *drain* and *source* terminals as described in lecture.

   [You may be wondering why you can't just use the signal from the Arduino directly to light the LED and why you need a transistor at all. In general, the microcontroller may not be able to source enough current to light the LED brightly (or drive whatever load you may have connected).]

   Consider the circuit shown in Figure 3. The transistor is a STQ1NK80ZR MOSFET. To transmit TV codes over the largest possible distance, we must guarantee that the maximum possible current flows through the IR LED. Assuming the IR LED acts as an ideal diode with the ON voltage found in 4b and that R = 0, **what is the current through the LED?** (Hint: use the MOSFET datasheet)

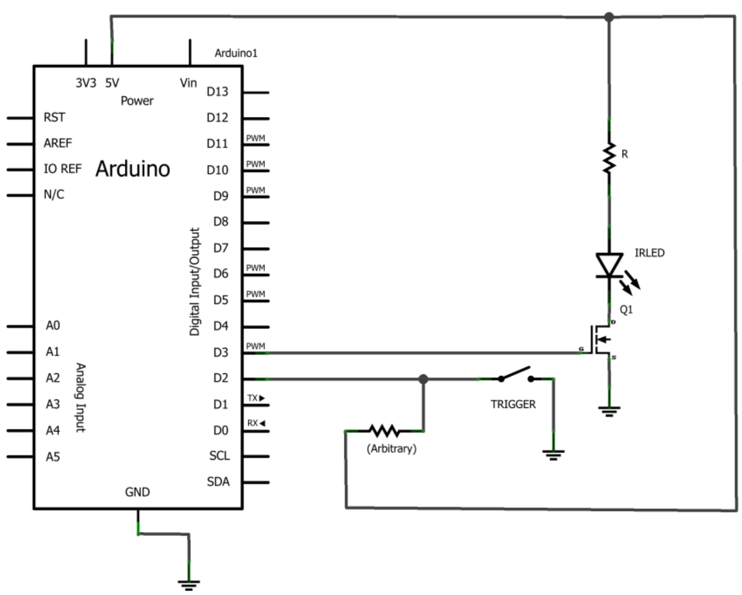

Figure 3 – An Arduino directly driving a MOSFET in series with an IR LED and current limiting resistor.

6. Now consider the circuit in Figure 4. Both MOSFET A and B are STQ1NK80ZR MOSFETs. Assuming ideal switch model for MOSFET A, **what is VGS of MOSFET B when the gate of MOSFET A is driven by a 5V pulse from the Arduino? What is VGS of MOSFET B when the gate of MOSFET A is 0V?** Use your estimated value of ON voltage for the IR LED and the STQ1NK80ZR MOSFET datasheet to **solve for the current through MOSFET B when its VGS is 10V and R = 0 Ω.**

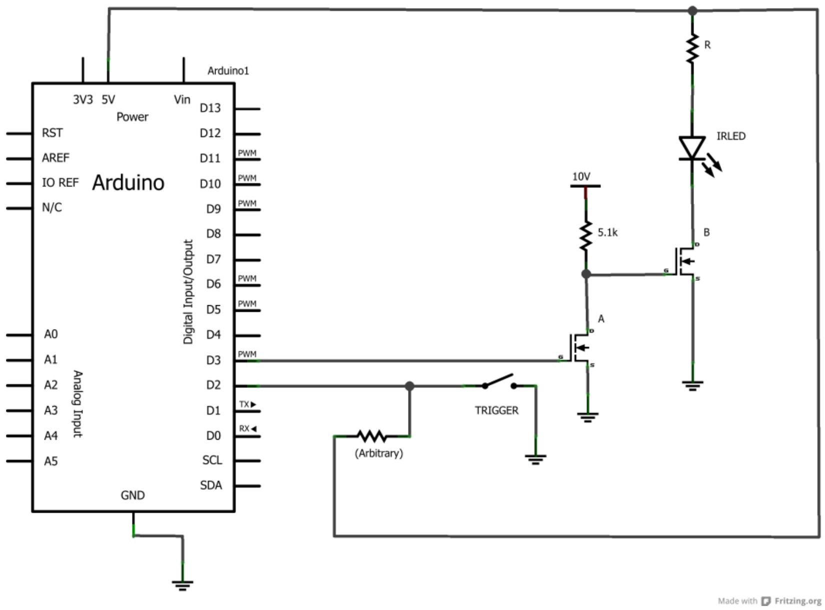

Figure 4 – Two-stage MOSFET driver circuit.

That concludes the pre-lab work. Save the trigger/button circuit for the lab. The remainder of the sections should be completed in your regularly scheduled lab session.

5. MOSFET Characterization (15 points)

The files downloaded for this lab include a datasheet for the STQ1NK80ZR MOSFET. Locate the datasheet and refer to it for hooking it up and using it in the following steps.

1. Connect one output of your DC power supply (e.g., the +5 V terminal) to the gate terminal of the transistor you selected. Ground the source terminal and connect your ammeter between a 5 V supply (from the other terminal of the power supply) and the drain. **Draw the schematic of this circuit in your lab notebook and indicate which pins of the MOSFET correspond to the gate, drain, and source of the transistor.**

   Sweep the gate voltage from 0V in steps of 0.2 V, leaving the drain voltage at 5V. At each voltage step, measure the drain current with your ammeter. At each step, also calculate the power being dissipated by the transistor. It is also a good idea to check that the voltage display on your power supply is accurate by using your voltmeter to check the output voltage at both the drain and gate.

2. **Using MATLAB or Excel, construct a plot of the current as a function of gate-to-source voltage and attach it your lab notebook. Is the result what you expected from the lectures?** Although in your circuit the source is held at a constant potential, in general it is the difference in the gate and source voltages that controls the drain current. **From your plot, what voltage level do**

you observe to be the threshold?

3. Now, swap the voltage sources – apply 5V to the gate and sweep the voltage source on one side of the ammeter from 0 to 5V. Use fine-grained steps at small values of VDS (the drain voltage minus the source voltage) and larger steps when you observe the drain current to level off. Repeat this measurement for a gate-to-source voltage of 4.5 V and 1 V. **Using MATLAB or Excel, plot the results for drain current (I_D) vs. drain-to-source voltage (V_{DS}) for each value of gate-to-source voltage (V_{GS}) on the same set of axes. Using this data and the data from the previous step, discuss why the gate of the MOSFET is considered the controlling terminal of the device.**

4. **Using the plot constructed in the previous step, estimate R_{on} for the STQ1NK80ZR MOSFET.** (Hint: what is the slope of the I-V curve?)

5. **Compare your I-V curve from step 4 to the I-V curve of the STQ1NK80ZR MOSFET given in its datasheet. Are they similar? If not, give a plausible explanation as to why they are different.**

You now know how your MOSFET will respond when you use it in your driver circuit. The gate of the MOSFET is the control of your switch.

Please have a TA check your results and lab notebook before proceeding to the next section.

6. TV Code Transmission (20 points)

Some TV codes are transmitted simply by turning on an LED for an "ontime" and turning it off for an "offtime." Others are more complicated, requiring the LED to be pulsed at a certain carrier frequency during the "ontime" periods. We won't bog you down with the details of structuring, writing, and reading the codes because it's not important for this course; instead, we will provide you with a mostly complete program written by Ken Shirriff that does exactly this. It includes the 100 most popular TV codes for North America, Europe, and Asia. A detailed explanation of how this program works can be found in the comments and on Ken's website (http://www.arcfn.com/2009/12/tv-b-gone-for-arduino.html). All we will ask you to do is fill some blanks in the code based on the previous exercises and simple Arduino know-how.

1. To open the TVB_220 sketch, start by opening the TVB_220.ino file in the TVB_220 directory found in the lab's ZIP file. You should see the following files after opening the sketch:
   - main.h
   - WORLDcodes.cpp
   - TVB_220.ino

   The *TVB_220.ino* file contains the main program while the other files contain supporting code. The TV we are using in lab is turned on by NA code 20. To save time, the macro 'QUICK' in *main.h* is set to 1, which instructs the program to only send the code corresponding to our TV. You can set this macro to 0 to send all the codes defined in *WORLDcodes.cpp*.

2. In *TVB_220.ino*, search (using Ctrl + F) for the string "220_TODO" to find comments describing what we want you to write. Write the required code as instructed by the comments. You'll note that it is similar to the pre-lab assignment; you will read the digital state of pin D2, write the digital state to pin D3, and fill in other small sections of code. Compile and upload your program to your

Arduino.

The next two steps should give you an idea of what these codes actually look like before testing it on a TV.

3. Use the switch circuit you built in the pre-lab with your new jammer program. Make sure whatever supplies you are using (function generator, DC power supply, etc.) share the same ground; the Arduino ground (GND) pin should also be tied there. Push the button; do you see the orange indicator LED on the Arduino flashing? If so, the program is sending codes.

4. Use your oscilloscope to display at least one code coming out of pin D3; measure the amplitude of one code. In order to do this, you will need to know the following scope functions:
   a. Probe compensation
   b. Bandwidth limiting
   c. Voltage and time scales
   d. Triggering (you will need to demonstrate use of triggering to a TA)
   e. Continuous run vs. single capture (you will need the latter to only capture the first few codes)
   f. Quick measurements and cursors (you will need the latter if you only wish to measure the amplitude in a specific region, which you will if you've captured more than one code)

Include a screenshot of a captured code with an amplitude measurement. Attach it to your lab notebook. To do this, insert a flash drive, hit the button for "Save/Recall" in the File subsection of the front panel, hit the tab under the display for "Save," make sure your flash drive is selected as the destination with the second tab, and then hit "Press to Save").

5. Breadboard the circuit shown in Figure 3 with R=0 Ω. **Measure the current going through the LED. How does it compare to the target current from the pre-lab? If it is significantly different, give a possible explanation.**

6. **Demo your working TV jammer to the TA using the TV(s) we have provided. What is the approximate maximum distance over which you can control the TV?**

7. Breadboard the circuit in Figure 4 using a value of R such that the total series resistance (including R_{on} of the MOSFET) is the value you calculated in the pre-lab (question 4(e)). To ensure the TV jammer is safe and fully functional, measure the current going through the IR LED by putting the ammeter in series with the LED. If the current you measure is >200 mA, immediately turn off your power supply and choose a value for R until you are under 200 mA, or you risk burning out the IR LED. If the current measured is less than 180 mA, decrease the value of R to increase the range of your TV jammer. **With this resistor value, measure the current through the IR LED and estimate the actual R_{on} value of your MOSFET.**

8. **Demo your working TV jammer to the TA using the TV(s) we have provided. What is your new approximate maximum distance?**

9. Compare the two TV jammers you constructed. **Which one had a greater range and why?**

You now have a working TV jammer. You may experiment with how the codes are stored and transmitted, and try adding codes that aren't already included.

Please have a TA check your notebook before clearing your workstation.

After signoff by a TA, properly stow all equipment, cables, tools and reusable parts. Return other parts to appropriate discard bins and make sure your work area is free of all clutter and left clean for the next group.

7. Expanding the TV Jammer Project (Extra Credit)

Now that you have a working prototype of a TV jammer, here are a few ideas to expand on this project for extra credit at the discretion of the professors or PhD TAs. Please do not work on the extra credit before completing the main sections of the lab. If you choose to pursue a project, please present it to a PhD TA or professor before the next lab. We will evaluate based on creativity and hard work (no Wikipedia regurgitations!). No extra credit is guaranteed.

Here are some thoughts to get you started:

- The long-range version of the TV Jammer designed in this lab requires the bench power supply to drive the transistors. This means that it would be very difficult to take into a bar. What can you do to make the system portable? Make a portable version of your TV jammer, thinking through what would be needed to drive the entire system off of (a reasonable number of) batteries. Here are some points to consider:
    - Why do we need the 10V source in our application? Could a change in the properties of a component remove the need for this device?
    - Ideally, the diode should be off at all times unless actually trying to turn off a TV – is this the case in our final design? What could you change to minimize the power consumption?
    - Could you make the system more robust by removing the breadboard and soldering the components in a smaller form-factor?
    - Can you think of a way to increase the directionality of the jammer? (That is, make it work on a wider range of angles instead of having to align the device precisely)
- We only considered two current levels on the diode when measuring the functional distance of the TV jammer. It could be useful, however, to understand exactly how the power and distance trades off. For example, if you double the power through the LED, how much additional distance do you get? Is there a point of diminishing returns? Design an experimental set up that allows you to accurately control the current through the LED and then plot functional distance versus diode power.

Appendix: Measurement Guide

1. DC Power Supply

This tool is used to generate steady DC voltages. There are three output terminals, +6V, +20V, and -20V, all of which are referenced to the *common* terminal. The terminal labeled “ground” is meant to be a connection to earth ground, and is generally shorted to the common terminal for noise rejection. The +6V terminal is capable of supplying 0 to +6 VDC; you can tune the output value by turning the labeled knob. The +20V terminal can deliver 0-20 VDC; the -20V terminal *tracks* this +20 V terminal, outputting the negative of whatever is coming out of the +20 V terminal. You can, however, change this by adjusting the “tracking” knob, which can set a ratio between the outputs of the two terminals.

<u>Note</u>: Be sure to set the display to show you the voltage of whatever terminal you are controlling; this is selectable with buttons. The displayed value is not terribly accurate and is only meant to give you a rough idea of the voltage the supply is outputting.

2. Digital Multimeter (DMM)

The digital multimeter combines voltmeter, ammeter, and ohmmeter functions (as well as some others) into a convenient package. Below are basic instructions on using these functions.

2.1. Ohmmeter

1. Connect one lead of your DMM to each side of the resistor. Be sure to have the positive lead of the DMM plugged into the terminal to measure resistance (it’s the same one used to measure voltage).
2. Hit the button on the front panel for resistance (Ω).

2.2. Voltmeter

1. To measure the voltage across an element, place the DMM leads in parallel with the element (see Figure 5(a) for an example). Be sure to have the positive lead of the DMM plugged into the terminal to measure voltage.
2. Hit the button on the front panel for voltage (DC V for DC voltages or AC V for AC voltages, which displays the RMS value).

2.3. Ammeter

1. To measure the current through an element, place the DMM in series with the element (see Figure 5(b) for an example). Be sure to have the positive lead of the DMM plugged into the terminal to current (in general, you will use the terminal rated for smaller currents [1.2 A vs. 12 A]).
2. Hit the button on the front panel for current (DC I for DC currents or AC I for AC currents, which displays the RMS value).

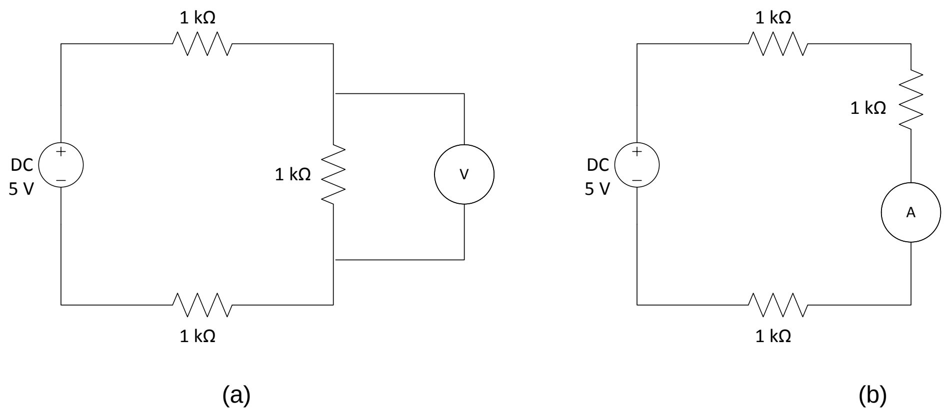

Figure 5 – (a) Configuration to measure voltage across an element; (b) configuration to measure current through an element.

3. Function Generator

A function generator, at its most basic level, is used to output various waveforms, like sine, square, or sawtooth waves. It is also capable of generating more complicated functions, like amplitude-modulated signals. The front panel has buttons to allow you to select between the various waveforms. At a minimum, you will set the amplitude and frequency of each; you can also adjust things like a DC offset level or duty cycle. Doing this is fairly intuitive; you select the button for the waveform you want, then use the buttons on the front panel to switch between frequency/amplitude/duty cycle/etc. and use the dial or the number pad to set the desired value.

The two most common issues students have with function generators are:

- **Forgetting to push the "output" button**. If you don't push this button, the function generator will not output your signal. When you are ready to apply the waveform to your circuit, be sure to push this button.
- **Using the wrong impedance setting**. Unless we tell you otherwise, navigate to **Utility→OUTPUT SETUP → LOAD → HIGH-Z → DONE** every time you turn on the function generator. This will tell the function generator to expect loads much larger than 50 **Ω**.

3.1. Outputting a DC Voltage with your Function Generator

1. Select sine wave.
2. Turn down the frequency to something small (e.g., 1 mHz).
3. Turn down the amplitude to something small (e.g., 20 mVpp).
4. Set the DC offset to whatever DC level you are trying to generate.

OR

Navigate to Utility → OUTPUT SETUP → DC.
Set the desired voltage and turn the output on.

4. Digital Storage Oscilloscope (DSO)

The scopes we use have two channels, each of which can measure a voltage signal with independent probes. The channels are labeled "1" and "2;" there are buttons on the front panel that let you select between them. If a channel is being displayed on the screen, the button for that channel should glow. You can enable/disable a channel by pushing its label button.

For your experiments, **always** connect the second lead of your scope probe to ground unless we instruct you to do otherwise. **Do this for both probes**.

4.1. Probe Compensation

Oscilloscope probes introduce capacitance in parallel with the 1 MΩ resistance of the probe. This is meant to compensate for the capacitance across the scope input terminals to equalize the frequency response. There is a small screw at the base of the probe that plugs into the scope which can be turned to adjust the value of the capacitor. If the screw is plastic, you can use a metal screwdriver to adjust it; otherwise, use a nylon (or otherwise insulated) screwdriver.

Your oscilloscope has a built-in square wave output that can be used to adjust the compensation capacitor. A sine wave, as you'll learn in 18-290, is a "pure tone," consisting of a single frequency, but a square wave is more complicated, containing high frequency harmonics. If the probes are not adjusted properly, some of these high frequency components will be lost, causing the edges of the square wave to become rounded. Other times the high frequency response of the probes is enhanced and the leading edges spike and then decay. The probe capacitor must be adjusted to make the frequency response correct and show a true square wave.

1. Connect a scope probe to the channel 1 terminal of the scope. You'll have to push the probe in and rotate to lock it.
2. Ground the "pigtail" alligator clip lead to the metal tab at the bottom of the scope labeled with an earth ground symbol.
3. Connect the main probe lead to the "Probe Comp" tab at the bottom of the scope.
4. Hit the button for "Autoscale." If the waveform is fuzzy, press the button for channel 1, then hit the tab under the display for "BW Limit" (see section 3.2 in the appendix for a discussion about what this does).
5. Use an insulated screwdriver to adjust the probe compensation by turning the small screw located on the scope probe. Take the probe to undercompensated and overcompensated states so you see what they look like. Tip: there should be a small probe screwdriver in your toolkit.
6. Return the probe to a compensated state.

Always check probe compensation before using a scope probe. If you find a probe that cannot be compensated, DO NOT continue to use it and DO NOT put it back on the rack. Notify a TA and request a new cable.

4.2. Bandwidth Limiting

You will frequently see that the signal you are trying to observe is noisy. Noise on a signal means you are picking up frequency components you don't actually want to look at; for example, if you have a 10 kHz sine wave, and you see the waveform is a bit "fuzzy," it's because you're seeing higher-frequency signals riding on it. In Hamerschlag Hall, high-frequency noise is a problem because of an FM radio station in close proximity.

Your scope has a built-in function to filter out the noise above 100 MHz. To use it:

1. Enter the menu for channel 1 (assuming that is the channel you are using). At the bottom of the display, you should see a tab that says "BW Limit."
2. Push the button underneath the tab. You should see the wave get less fuzzy.

4.3. Voltage and Time Scales

In order to be able to see a waveform, you'll need to adjust the scale of the axes on the display. As you might imagine, the x-axis represents time and the y-axis represents voltage; there are knobs for each of these on your scope. You can see what the scale values are set to on the top of the display.

1. Adjust the knob that controls time (sec/div, which is the amount of time each division of the grid in the x-direction represents) to something reasonable to view a wave with a given frequency. In other words, don't set the time scale such that each division represents 10 ns or something absurdly small if you're trying to see a 1 kHz sine wave. The knob is at the top of the front panel in the box labeled "Horizontal".
2. Adjust the knob for channel 1 that controls the voltage per division of the grid in the y-direction. Again, set it to something reasonable to see a signal of a given magnitude. The knob is located near the button for channel 1, in the box labeled "Vertical". Channel 2 has an independent control for this setting, but both channels use the same time scale.

4.4. Triggering

Sometimes you won't be able to capture a clear picture of your signal due to the scope's inability to accurately control when it begins capturing and displaying the input signal. This is common for both repetitive/periodic waveforms, which may appear unstable on your display, as well as single shot pulses. To fix this issue, we adjust the triggering parameters of the scope. Once a captured signal fills the display, the scope pauses its tracing. The signal will not continue to be traced until a specified event occurs. The occurrence of this event *triggers* the scope to continue drawing the trace. Most commonly, the trigger event is the signal passing a threshold voltage on a positive or negative edge (edge triggering).

1. Press the "Trigger" button on the front panel of your oscilloscope in the subsection for triggering. You'll see a tabbed menu appear on the bottom of the display. Select the tab that says "Source," and select the option for channel 1, which will tell the scope to trigger based on what it measures on channel 1. Use the "Edge" tab to set the trigger event to a positive edge.
2. Adjust the knob labeled "Level" on the front panel of the scope in the subsection for triggering. You'll see a cursor move on the display; this cursor indicates the threshold voltage level at which the scope will trigger tracing. Generally, if you set the trigger level around the midpoint of your signal, the scope will capture a steady wave. Play with the trigger level to get a feel for what it does; note that if you take the trigger level above the maximum of your signal or below the minimum, the scope can't get a steady picture.

A good way to test whether or not you are comfortable with triggering is by using the probe compensation square wave and adjusting the scales and trigger level manually. Note what happens to the wave if the trigger level is not set correctly. Do this in Lab 1 and demonstrate to a TA you know what you are doing.

4.5. Continuous Run vs. Single Capture

In some cases, you won't want the scope to continually display what it reads at its input. Say you have a circuit that emits a pulse when a button is pushed and you want to observe the pulse; if the scope continued to trace the signal, you'd lose it. That's where single capture comes in.

1. At the top right of your scope front panel, you'll see a subsection for "Run Control." Push the button for "Single." When the scope begins to pick up a signal, it will trace it until it reaches the end

of the display and then stop (ignoring whatever is going on afterwards). Sometimes, your scope will miss the signal. You can keep hitting “Single” until the display captures what you want.

2. To go back to continuous capture mode, hit the “Run/Stop” button.

Note: you can also pause the tracing of a signal by hitting “Run/Stop” in continuous capture mode. When you push the button again, the scope will continue tracing the signal. This is useful if you want to perform a cursor measurement.

4.6. Measurement Functions

4.6.1. Quick Measurement Functions

There are many functions the scope provides to measure different quantities. Here, we include only a few of the more common ones we will expect you to use this semester. You can consult the manual if you’re curious about the other functions.

To use the quick measurement functions, hit the button on the front panel that says “Meas” under the “Measure” subsection of the front panel. At the bottom of the screen, you will see a tabbed menu appear; the buttons are:

- 1: controls the source, or the channel the oscilloscope will perform a measurement on.
- 2: selects the type of measurement being performed; if you push this button, a list will appear. You can navigate through this list using the knob on your front panel in the gray box, with a circular arrow next to it. Once you have highlighted the measurement you want, push the knob to select it.
- 3: triggers the measurement.
- 4: various settings for the quick measurement (you probably won’t use this much).
- 5: clear the measurements displayed at the bottom of the screen above the tabbed menu. Every time you take a measurement, it is added to a box in this location. Using the clear button will remove the measurements from the screen. You can clear individual measurements or clear them all.
- 6: lets you choose if you want to display the statistics of your measurements (e.g., mean, std. dev.) on the screen.

4.6.2. Cursors

Sometimes the quick measurement functions don’t cut it and you have to do things manually. The cursors are what you turn to when this is the case. With the cursors, you can accurately measure voltage amplitudes, time delays, phase, etc.

To activate the cursors, press the button that says “Cursors” in the “Measure” subsection of the front panel. You’ll see four dashed lines appear on the screen, two in the x-direction (representing voltage levels) and two in the y-direction (representing time values). By adjusting the positions of these lines, you’ll be able to accurately measure signal amplitudes, the time between two points of your choice, and anything else you can extract from that information.

To select which cursor line to adjust, hit the button for the bottom tab labeled “Cursors.” You’ll see your options pop up; you can move X1 and X2 (the cursors measuring time) independently or together, and likewise for your Y1 and Y2 (voltage level) cursors.