The Art of Debugging Circuits

Or: How I Learned to Stop Worrying and Love Analog

D. Elliott Williams
6.101 TA
Spring 2016

Cheat Sheet: Common Circuit Problems to Look For

- **Always Check First**
  - Is the power supply on?
  - Is your oscilloscope settings correct? (AKA is there actually a problem?)
  - Do you understand the schematic?
  - Did you build it correctly?
  - *DO YOU UNDERSTAND THE SCHEMATIC?*
  - *DID YOU BUILD IT CORRECTLY?*

- **Oscilloscope / Test Equipment Issues**
  - Are you triggering properly?
  - Are you in in the correct AC/DC mode?
  - Are you at the appropriate zoom level in time and amplitude?
  - Is your trace offset centered at zero?
  - Is your probe ground clip connected to ground? (do this)
  - Are multiple probe ground clips tied to different voltages? (DON'T DO THIS)
  - Is your scope and probe on 10x mode?
  - Did you check/calibrate the scope probes?
  - Is your function generator in High Z mode?
  - Is the output button on the function generator green? (is it outputting a signal?)

- **Op-Amp Problems**
  - Is it powered?
  - Did you accidentally mix up the pins for V+ and V-?
  - Do you actually have feedback (is it connected properly)?
  - Is the op-amp in the desired feedback mode? (positive or negative)
  - If it is in negative feedback, does V+ = V-?
    - Remember that the output voltage will do whatever it needs to in order to ensure V+ = V-. If they aren't equal then it is a good sign that either your circuit is not connected correctly OR that your output pin is unable to reach the necessary output voltage (because it not between the supply rails or requires too much output current).
  - Did you read the data sheet?
  - Are you drawing too much current?
  - Are you expecting outputs voltages too close to power rails?
    - No op-amp is actually rail-to-rail.
  - Is the input impedance high enough?
  - Is the input bias/offset current low enough / compensated?
  - Is the input offset voltage low enough / compensated?
  - Is the slew rate high enough?
    - If not you will notice that the output voltage has a maximum slope
  - Is the gain-bandwidth product high enough?

- **Device/ Integrated Circuit (IC) Issues**
  - Are all of your IC's powered? (power pins tied to ground and/or supply voltages)
  - What is the quiescent operating point (ie with no inputs)?
  - Did you read the datasheet / understand how to use the device/ IC?
  - Can your device / IC even do the desired operation? (Is it fast enough?)
  - Is the device/ IC plugged in correctly? (It is easy to plug them in upside down)
  - Do you have any floating IC pins that should be tied to something
    - Only pins labeled "not connected" should be left floating; you must watch out for pesky enable pins that must be tied to the correct voltage.
  - Did you accidentally destroy the IC/Device?
    - Common for circuits involving a lot of power.
    - Check circuit to make sure replacement won't also be destroyed.
  - Did you use the recommended configuration for your application that is sometimes specified in the datasheet?
    - It is very common for sensor datasheets to specify a schematic, *use it.*

- **Unwanted Signal Issues (including Noise, Interference, and Oscillations)**
  - Do you have bypass capacitors?
  - Are your connecting to the power supply via twisted wires? (twisted pair)
  - Are your leads short?
  - Is your circuit neat?
  - Can you filter out undesired signals?
  - Are you operating within the bandwidth of all devices? (can cause oscillation)
  - Are you using a non-unity gain stable op-amp as a buffer? (will cause oscillation, check the data sheet)
  - Do you have unwanted feedback?
  - If your design requires feedback that involves more than an op-amp (as in phase-locked loop or an audio amplifier) have you considered loop's stability?
    - If not you should talk to the staff.
    - You can also try reducing the amount of feedback by attenuating the signal in the feedback path. This can often remove oscillations.
  - If the circuit has a lot of noise, is the first stage in your signal chain an amplifier?
    - Amplifying a signal as soon as possible reduces the effect of noise from the rest of your circuit and only the amplifier's noise matters.
  - It is also a good idea to filter out all signals that are outside of the desired frequency range. This will reduce noise.
  - If you are building a circuit above ~100 kHz, are your input and output nodes far apart from each other? (coupling through breadboard pins can cause undesired feedback)
  - Do you have any ground loops? (multiple paths for current to flow to ground)

- **Construction Issues**
    - Is there a bad breadboard connection? (a very short wire that goes into the hole but doesn't make contact)
    - Some breadboards have split power rails, make sure the entire rail is connected.
    - Is there a short between neighboring breadboard rails? (sometimes happens)
    - Do you have any broken connections / parts?
    - Do you have a bad solder connection?
    - Do you have solder bridge? (a short caused by solder connecting adjacent pins)
    - Are two wires touching that shouldn't be? (avoid by building neat circuits)

- **General Issues**
    - Are the power rails at the correct voltages?
        - If not you might be drawing too much power.
    - Check your DC voltages (bias and reference). Do they make sense?
    - Did you check for shorts using the multimeter's continuous mode?
    - Are you trying to drive the same node to multiple voltages?
        - This occurs when you connect the output of two buffers directly together.
    - Check the input-output relationship of your sub-modules independent of each other.
    - Are your components the correct size?
        - accidentally using a 100 Ohm instead of a 100k Ohm resistor is common and will certainly cause problems.
    - Are you using virtual grounds correctly?
        - Note that any bias voltage fed into an amplifier will ALSO be amplified. Thus the bias voltage should be **exactly** equal to virtual ground.
        - When using virtual grounds it is better to pretend that the virtual ground **IS** ground and that the actual ground (negative end of a single ended supply like a battery) is actually the **negative supply rail**. In this case the positive and negative supplies are equal to half the single ended supply voltage.

Introduction: Why Debugging Matters

Now that we have reached the final project portion of the class, the staff's ability to help you debug circuits is going to decrease dramatically. This is because you will be designing your very own circuits and systems (exciting!) that we have never seen before. Because labs have been designed, tested, and verified by the staff, you can be assured that the circuit will work if you follow the schematic. This is not necessarily true for your designs. In addition, labs have the benefit of being simultaneously tested by all of your fellow Analog Adventurers (6.101 students). Therefore when you have a problem, the odds are high that the staff have helped one of your peers with the exact same problem and know the solution. However it is pretty much guaranteed that we will not have seen any of the problems that you encounter while working on your final project.

The combination of these facts, (1) we don't know your design, (2) your design might not necessarily work, and (3) we haven't seen the problem before either, makes it much more difficult for us to help debug your circuits. This is not to say that you cannot ask us for help; we are here to help you and our years of experience working on circuits provides us with insight that could be very useful in solving tough issues. Rather, I wanted to stress the importance of developing your debug skills because **you** will be in the best position to solve your own problems. With this in mind I have written the following guide providing some advice on how to debug circuits. The thought process involved in debugging is an incredibly valuable skill no matter where you end up working or what you end up doing.

It is important to remember that debugging is a skill as useful to a circuit designer as one's understanding of mathematics, physics, and circuit theory or one's ability to come up with creative new topologies. It is almost impossible to do something perfect the first time. Therefore it is very important to be able to be able to figure out what went wrong and resolve the issue. This is useful for more than just getting a design to work. Debugging is actually an intimate part of the design process itself; good debugging skills can identify where a working design can be improved.

The purpose of this document is to give you advice on how to debug and maybe resolve some issues that you might run into with your project.

The Debugging State of Mind: Schematic vs. Construction

The most important thing to remember when debugging is to **THINK**. Trust me, it is harder than it sounds. Sometimes it can appear that the circuit is "magically" breaking for no reason, or that it has somehow violated the laws of physics to produce an "impossible result". At this point it is tempting to stop thinking and start changing things at random. Sometimes this method works, but it is extremely inefficient, might make things worse, and doesn't teach you anything. Although we use circuits to accomplish almost magical feats (long distance wireless communication, imaging atoms, sending each other emojis) remember that circuits are NOT MAGICAL.

A circuit appears magical when their is a conflict between your logic and physics. Remember: physics is never wrong (with respect to conventional circuits). Circuits always behave logically. Your circuit is doing exactly what it is supposed to do the way that you built it. Your *logic* is wrong. This can either be because **you do not understand what you built** or **you did not build what you understand**.

I find this distinction profound because it separates two very different types of issues that can show up when building circuits. Most circuit issues fall into the second category. Say you are trying to build an inverting amplifier and you accidentally connected the feedback resistor to the positive input. In this case the topology is correct and you understand it correctly, but you did not build it correctly. Hence, *you did not build what you understand*. Most circuit errors fall into this category and thus the bulk of this document is dedicated to it.

The first category, while less common, is typically more frustrating because the circuit is built correctly but it is not doing what you expect it to do. This is usually because there is an error in your schematic or your understanding of it. One example would be if you are trying to build an non-inverting amplifier so you deliberately connect the feedback resistor to the positive input thinking it will create a non-inverting amplifier. Clearly you do not understand how op-amp circuits work and thus *you do not understand what you built*. Therefore it is ***ABSOLUTELY IMPERATIVE*** that you understand how your schematic works before you start to build it. In fact I will put it bluntly, just to make myself extra clear.

IT IS *IMPOSSIBLE* TO EFFECTIVELY DEBUG A CIRCUIT IF YOU DO NOT UNDERSTAND ITS SCHEMATIC AND DO NOT KNOW HOW IT IS SUPPOSED TO WORK - *Socrates*

If you are having circuit problems be honest and ask yourself *"Do I understand how this topology works"* and *"how confident am I that this schematic does what I think it does"*. These questions are incredibly important and can save you a whole lot of time. You cannot logically reason about, and thus you cannot debug, a circuit that you do not understand. In addition, you can save yourself a lot of construction time if you verify that a topology does what you think it should do before building it. The ability to verify designs before construction is one of the biggest advantages of using LTSpice. It is much easier to find schematic errors in an ideal environment like LTSpice where there is no chance of building the circuit incorrectly. Another good resource for checking schematics is to ask your peers or the staff to look them over.

Now that we have verified that your schematic indeed does what you think it should (if you have not please do now... I will wait...), we can assume that the circuit you have built does not match the schematic. Now we just have to figure out why. The the very first thing to check is whether or not you actually built it correctly and by that I mean, **did you connect all of the components together as shown on the schematic.** There is a huge difference between an op-amp circuit where the feedback resistor is *accidentally* not connected to the negative input (because of a bad connection or something) and a circuit where the resistor was plugged into the *wrong* breadboard pin. While the end result is the same, one is significantly easier to find and resolve. Thus the very first thing you should do after verifying that the schematic is correct is check that everything is plugged in correctly and the components are the right sizes. Then when you are finished, check it again. I do not care if you think you did it right, check it again. If you get stuck later and you cannot figure out what is wrong, *check it again*. This problem is very easy to resolve yet easy to miss even if you are diligent. If I had a nickle for every time I had a student tell me that they checked their circuit only to have me find an incorrect connection 5 minutes later... well let's just say that I wouldn't be going to graduate school.

How to Debug: Reverse Engineering Your Own Circuit

Now that the common simple to find errors have been resolved, we can get our hands dirty with the tough stuff. Recall that the most important thing to do when debugging is to **think**. The circuit you have built is still a circuit that can be represented with some schematic. If I were to give you *that* schematic you could tell me what would happen. The problem is that we have to do the process in reverse because we are not given the schematic; we have to figure it out from what is happening. The trick is to remember that the schematic of the circuit you are debugging is likely *very close* to the schematic that you tried to build (of course it is possible that you made multiple errors, but it is usually possible to tease the issues out one by one). Knowing that the circuit is *almost* correct gives us a place to start our thinking.

The first thing that one needs to know when debugging is *what is happening*. Obviously the circuit is not doing what you want, but it is still doing **something**. It is doing something even if it's output voltage is zero volts. In that case the circuit is outputting zero volts! This information is critically important because it allows us to narrow the list of possible errors. The error in the circuit *must* make the output voltage zero. One option is that the output has been shorted to ground. On the other hand, we can clearly rule out the possibility that the output has been shorted to the power supply unless of course their is something wrong with the power supply. Don't just limit yourself to figuring out what is happening at the output! Probe other parts of the circuit as well and figure out what is happening there. For example, it is **always** a good idea to probe the power supply to make sure that it is behaving as expected. We might have to revisit our "not shorted to supply" theory if the supply is also at zero volts.

Now that we know what is happening, and we know what is supposed to be happening (because we checked and understand our schematic right?), we can try and reason why the thing that is happening might happen. Guesses are fine as long as they are *educated.* Look at the schematic and ask yourself all of the ways that this circuit could fail in order to give you the result that you are getting. No output? Well my power supply could be off, my op-amp could be dead, I could have a short to ground, I could be trying to pull too much current from my op-amp. Make a mental list, or even write them down if that helps. Then slowly cross them off one-by-one by testing if that is true. Measure the power supply, test/replace the op-amp, measure the resistance between the output and ground, calculate the output current you need and check the datasheet to see if the op-amp can handle it. Slowly you will uncover the truth.

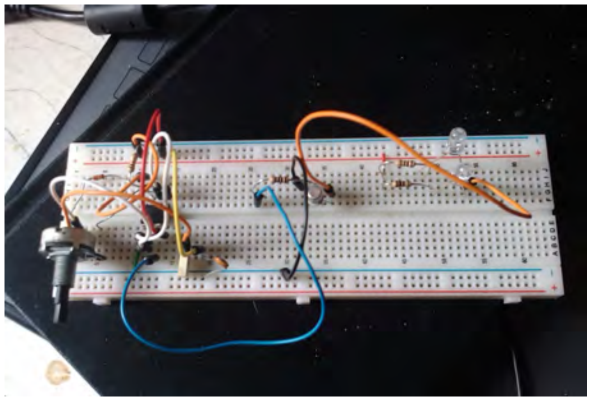
Best Practices: Follow Signals then Divide and Conquer!

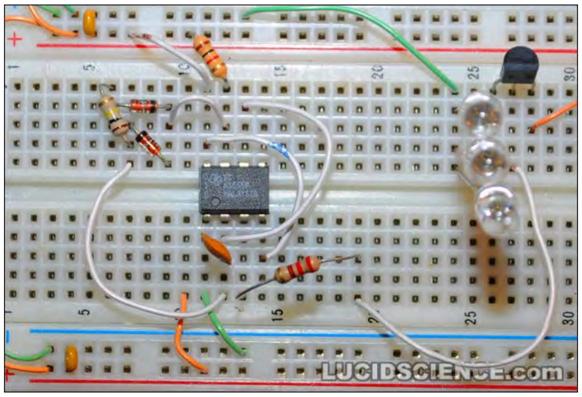
As you have learned from doing design, complex analog systems are comprised of sub modules that perform specialized tasks such as filter a signal or drive a motor. It is the interaction of these modules that performs the desired function. So when there is an error in the circuit, it is highly recommended that you test each module separately to isolate the source of the problem. In particular, many analog systems are signal processing pipelines that modify a signal in sequence. For example, a microphone amplifier might consist of a transducer, an amplifier, and then a filter. If there is an error at the output of the pipeline, it is good debugging practice to start from the beginning and work your way through the pipeline one step at a time. A working module will generate a junk signal when fed junk, whereas a block that produces a junk output from a good input is clearly not working. When working through the pipeline, you should always probe both the input and output of a block to verify that it is working properly. **I CANNOT STRESS ENOUGH** that it is a good habit to always use *at least* two scope probes while debugging in order to see how one affects the other. It is not only useful for checking in-out relationships, it is also useful to see whether signals are affect each other that should *not* be (such as a signal coupling onto a voltage reference).

Circuits can be more difficult to debug than software because there is no such thing as isolation. Changing one thing in a circuit, including probing it, changes the behaviour of every voltage and current in the circuit at least slightly. Circuits are typically designed to reduce the interdependence of different sections (usually by using op-amps). However, one cannot assume that the different sections are independent when something is wrong. The error could remove the isolation between sections so they interact in undesired ways (an op-amp error will usually cause this to happen). Therefore it is a very good idea when debugging to disconnect sub-circuits from each other and test them independently (with a function generator and a scope). As with testing down the signal processing chain, this will allow you to isolate the problem to a single stage. If all of the modules are working separately than you know that it is the interaction between two modules that is causing the failure. Adding modules back together one at a time can indicate the problem connection and allow you to figure out exactly what is wrong. This sort of error is typically caused by a schematic error and will likely require a slight design tweak.

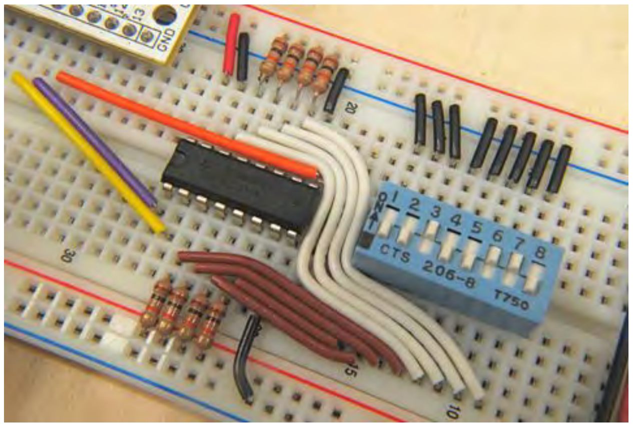
Neat Circuits Save Lives (or… Hours of Debugging...)

Now that I have given advice on the high level debugging thought process I am going to address some minor points that can help with debugging. The first point is that you should build your circuits as neat as possible and in a logical orientation. Use power and ground rails to route power (instead of wires jumping all over). Build stages in order so signals flow from left to right on the breadboard (unless you prefer right to left). Use neat wires that do not overlap a great deal. Color code your wires to signify what signal they carry; this is especially useful for power and ground connections (I recommend red for positive supply, blue for negative supply, and black for ground). Neat circuits will make your life easier in two very important ways: less parasitics and easier debugging. It is so much easier to work out what is going wrong in a circuit when you do not have to trace wires all over the place through a rat's nest to figure out what is connected where. Neat circuits clearly show what is connected where so it is very easy to find bad connections. In addition you are much less likely to make mistakes to begin with if you build a neat circuit because you must be more diligent and take your time. Finally, neat circuits implies short component leads which is important to reduce parasitics and reduce the chance of accidentally shorting something. That being said, your circuit doesn't have to be a work of art. You do not need to measure every wire to make them the perfect length. While that does look nice, it will take a lot longer to build and not provide that much additional clarity.


This is not a neat circuit. I have no idea what is going on.



Not perfect but probably good enough Notice how it is pretty easy to trace connections.

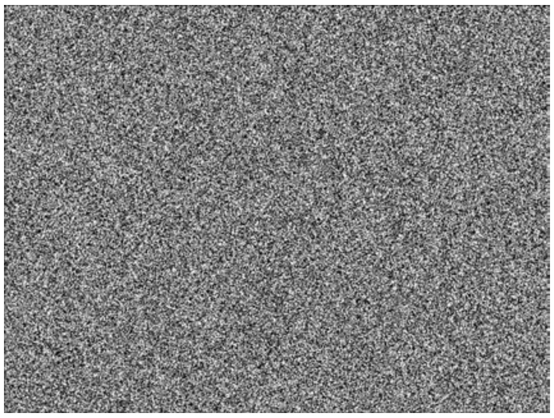

Beautiful, but almost certainly overkill.
Not necessary unless you have **a lot** of wires

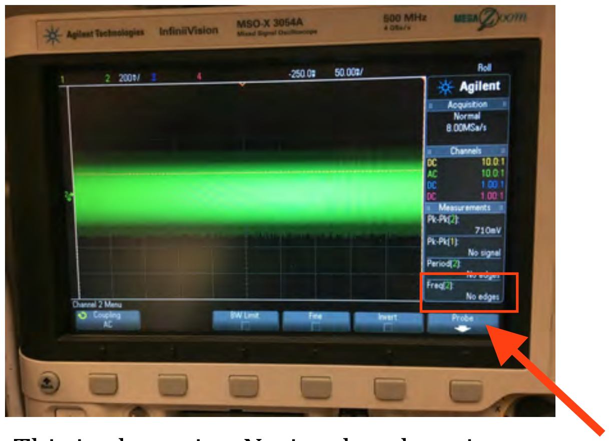
Datasheets Exist for a Reason

Another very good habit is to always read the data sheet **carefully**. They are easy to find on the internet. Every circuit element has limitations that should be avoided and are neatly summarized by the datasheet. If your circuit is not working, it is possible that you have bumped into one of these limitations and thus the device is not behaving as it should (limited op-amp output current and voltage is a common problem). So you should **always** check (preferably when designing the schematic and selecting parts) to make sure that you are using the device correctly. Does your device do what you think it should? Are you using it properly? Do you know what each pin does and have you connected it appropriately? Do not assume that you can leave a pin floating if you aren't using it. Did you supply power to the power pins? Are you within the expected operating regime? Did you temporarily leave the operating regime and thus destroy the device? All of these questions, and pretty much any question that falls under the category of "help my [name of integrated circuit and/or device] doesn't work" can usually be explained by reading the data sheet. Finally, for sensors it is **ESPECIALLY** important that you read the data sheet in order to use them correctly. Sensors are often designed to be driven in a very specific manner and if they are the datasheet will tell you how to drive them.

No, Your Circuit is not Noisy

Maybe it is the fact that I worked on reducing measurement noise as an intern, or maybe it was the fact that my current research focuses on generating noise, or maybe I am channeling the spirit of my 6.301 instructor (and general circuit wizard) Prof. Kent Lundberg, but for some reason my biggest pet-peeve is the following statement: "My circuit is really noisy." This might be pedantic, but I urge you to be very careful about what you call and think of as "noise". Pretty much every time a student claims that they have a lot of noise, they are **WRONG**. Noise is *by definition* non-deterministic. Therefore it is **not** periodic and cannot be clearly displayed on an oscilloscope. Most importantly, the random nature of noise means that very little information can be obtained. In fact the only information you can obtain from noise is how much of it there is. Noise will hide all smaller signals, set the bounds on how much information you can transfer, and generally ruin your day. Typically there is *almost nothing* you can do to get rid of noise except for putting your circuit in a refrigerator to cool it down or redesign it from scratch to be less noisy.

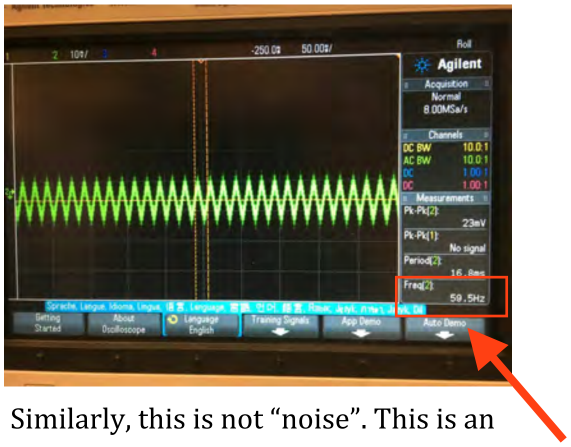

This is noise. The only information we can pull out is that it is noise.



This is also noise. Notice that there is no discernible pattern, just a randomly fluctuating signal. The only thing we do know is that there is a lot of it (~700mV).

The reason that I care about students incorrectly calling things noise is that you often ***can*** do something to resolve the issues that they are observing. If it is periodic then it isn't noise, it is an oscillation of sometype! This could be caused by instability in your circuit (which can be fixed), interfering signals from radio stations or the power grid (which can be fixed), or interfering signals from other parts of your chip (which can be fixed; noticing a trend?). By measuring the properties of the periodic signal you can learn something about what it is. For example if the signal is at 100 MHz then you know it is FM interference and if it is at 60 Hz you know that it is EMF from the power grid (if it is 50 Hz you know that it is EMF from the power grid and you are in Europe for some reason). So if you ever see unwanted signals while debugging, remember that it is not noise, figure out what it is, and resolve the problem.

Despite what philistines might say, this painting is not "noise". Notice how you can gain some information by looking at this painting; for example one can deduce that the yellow paint was added after most of the black paint.



Similarly, this is not "noise". This is an oscillation at 60 Hz which suggests that interference is being coupled in from the power traveling through the walls. Possible solutions include using shorter leads, removing ground loops, using twisted pairs, using differential signalling (requires circuit redesign), bypass caps, using a scope probe on 1.0x, making sure your signal is above or below 60 Hz and filtering out 60 Hz.

Unfortunately it would be technically incorrect to call this "Noise" despite the fact that no useful information can be obtained by listening to it.

Good Engineers Ask Good Questions

Sometimes the best way to debug a circuit is to have someone with a fresh perspective take a look at it. I cannot tell you how many times I have been stuck debugging something for 15 minutes or longer only to have a peer point out the problem after looking at the circuit for 15 seconds. This help does not have to be staff, your peers are on their way to become analog gurus themselves. However, when you do ask for someone's help it is important to ask **good questions**. Imagine that you are a 6.101 TA and a student is having trouble debugging their circuit and asks for some help. You know very little about their design and have no idea what sub-circuit they are currently working on. Which of the following statements do you think will allow you to help out the student most effectively:

(1) “Hey, my thing doesn’t work and I don’t know why.”

(2) “So here is the schematic of my receiver, this op-amp here [points to schematic] is supposed to amplify the signal but for some reason I am only getting 15V. I checked the [describes what they have tested thus far].”

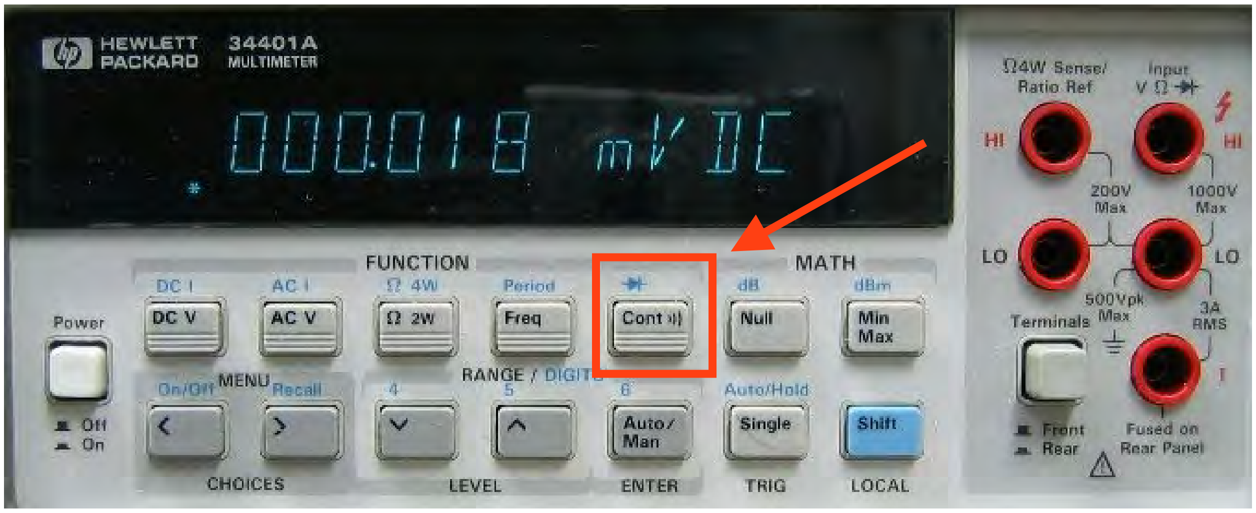
Notice that the second question gives you most of the information that you need to know, the context of the circuit (receiver), a schematic of the design, the problematic subcircuit, the problem, and lots of information about what could be causing the problem. The first question only tells you that the student is sad (which makes you sad).

Get Cozy with your Oscilloscope

Another fairly common problem students encounter is that their circuit is actually working *just fine* but they do not realize it because the oscilloscope is confusing them. Scopes are complicated and powerful pieces of equipment that can easily throw you off if you aren’t careful. Common errors include, not triggering, improper zooming, wrong coupling mode (AC vs. DC), and probe multiplication errors (10x vs. 1x). Finally, remember that **all probe ground clamp are shorted together** so you should (almost) never connect a probe ground clamp to anything that isn’t ground. Clearly it is very important that you understand how the scope works and always make sure that the scope has the correct settings. I am not going to get into the details of how oscilloscopes work here (that would be another 10+ page document), but if you have any questions at all please do not feel embarrassed to ask the staff. We are here to teach you such things!

The Best Multimeter Feature: Continuous Mode

Multi-meters have a **very** useful function called “continuous” for testing connections. Is this mode, the multimeter will beep if the two probes are shorted together. This is really helpful in finding shorts to ground (tie one probe to ground and tap away until you hear a beep), shorts between nodes (touch the two probes to the two nodes and hope it doesn’t beep) or testing if connections are good (touch the two probes to two wires that should be connected and hope for a beep). Obviously this technique is **extremely** helpful for testing solder connections. Finally this method is very fast because you do not need to look away from the board to determine if there is a short. It is one of my favorite tricks!



Continuous mode can be entered by pressing the third button from the right on the top row.

The Nuclear Option: Rebuilding

All of us run into bugs that we never quash. Sometimes the problem is just too subtle to figure out. So if you have tried everything and cannot figure it out, there is nothing wrong with rebuilding the circuit from scratch. Breadboards are beautiful but also evil. They are fraught with difficulties. Shorted internal connections, 1pF capacitors between rails, and bad wire connections are just some of the ways they will ruin a perfectly good circuit. So when you do rebuild get a completely fresh start and use a different breadboard and new components, the last thing you want is the same hidden dead capacitor or breadboard short following you to your new build.

Conclusion

Debugging is an important skill for all circuit designers. Not only does it help you get circuits working, it also shows you where designs could be improved and solidifies your understanding of the topology and the underlying physics and mathematics. Hopefully this document will guide you to be a more effective debugger and help you solve whatever issue you are facing. Please email me at gadgy42@gmail.com if you notice any typos, thought of any common circuit problems that I have missed, or want to share your thoughts on how to debug.