STANFORD CIRCUIT DEBUGGING SIMULATOR: A NEW TOOL FOR TEACHING

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Amy Verkler

December 2021

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Mark Horowitz, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Roger Howe

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Carl Wieman

Approved for the Stanford University Committee on Graduate Studies.

Stacey F. Bent, Vice Provost for Graduate Education

Preface

This thesis details the creation of a virtual circuit debugging simulator for use in an introductory circuit class. The simulator was designed to create an authentic debugging experience for students. Students could virtually take measurements on a breadboard circuit with a digital multimeter. The circuit debugging simulator allowed researchers to study debugging in a controlled setting. The simulator also shows potential as a curricular intervention to help students practice debugging.

Acknowledgments

I would like to thank my four parents, my brother, and my husband for supporting me through many stressful times in grad school. I would also like to thank my labmates for their community and support throughout the years. I would like to thank Argenta Price for her mentorship around all things related to education research. This thesis would not have been possible without the hard work of Atindra Jha writing Javascript and PHP code for the website. I also want to thank Mary McDevitt for her technical writing expertise and editing my thesis. And finally I would like to thank my advisor Mark Horowitz for always believing in me, even when I did not believe in myself.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Debugging physical circuits is a difficult and frustrating problem for novices and experts alike. In introductory lab classes, we have seen students brought to tears and fits of rage when their circuit is not working as expected. Although debugging is such an inherent part of building circuits, university curriculum addressing the circuits debugging process is almost nonexistent. Debugging circuits is seen as an art that students learn through experience in lab, rather than something that can be effectively taught. This disconnect is particularly problematic, considering that the majority of college instructors said that the primary objective of the lab portion of introductory electronics was learning debugging when interviewed [1]. To address this gap, we developed a computerized debugging simulator that functions both as an intervention and measurement tool for studying how students debug circuits.

The work presented in this thesis focuses specifically on debugging physical circuits, but many other forms of troubleshooting exist and require similar skills. Troubleshooting is an activity that people engage in when an engineered system is broken or not working as expected. The troubleshooter interacts with the system to diagnose and ultimately fix the problem. We will use the words debugging and troubleshooting interchangeably throughout this thesis.

Troubleshooting requires many types of knowledge and consists of a number of cognitive steps. To successfully debug a circuit, one must know what the circuit is supposed to do and be able to calculate the voltages at various points in the circuit. One must be able to look at the physical circuit and relate it back to the theoretical diagram. One must also know to use tools to take measurements of the circuit and strategize where and when to take the measurements. One must then compare the physical circuit and measurements to the theoretical models to determine whether the two agree or not. If there is a discrepancy, one must troubleshoot and bring these two into agreement.

Throughout the process, the troubleshooter must make decisions about what features are important and what course of action to pursue. There are nearly infinitely many things that could be wrong with the circuit, so narrowing the search space and isolating the problem is a key part of debugging strategy. As more information is gathered, one must interpret the results and incorporate it into their mental model. With so many moving parts, it is no wonder that students struggle with the complex cognitive task of troubleshooting.

The goal of the research presented here is to explore how students debug physical circuits. Although we considered using traditional methods to study students, like observing students directly in lab or having debugging journals and test problems, we concluded that both methods had shortcomings. Observing students in lab was time-intensive and was thus not expandable to large numbers of students. Furthermore, researchers at University of Colorado at Boulder had already published such a study [2]. We wanted to improve upon their results by expanding the research to larger numbers of students, so we looked into debugging journals and paper-and-pencil test questions that could be given to hundreds of students at once. Unfortunately, students sometimes hide their previous mistakes, so both journals and test questions were plagued by hindsight bias. In debugging journals, a number of students could not accurately reflect on their own debugging experiences. On tests, students would erase previous incorrect work or not fully explain their reasoning even when explicitly asked. Therefore, we created an effective circuit debugging simulator to measure debugging for large numbers of students.

This thesis details the creation of this new tool for studying and measuring debugging. Our online debugging simulator emulates taking measurements on a physical circuit in lab. While such simulators already existed, none allowed researchers to record all the actions taken by students for later analysis. Therefore, we designed our own debugging simulator that fulfilled both these frontend and backend needs. Not only did this allow us to test large numbers of students quickly and effectively, but it also gave us access to students' measurements and actions in real time, instead of their interpretations after the fact. This allowed us to study measurement strategy and see how it correlated with performance.

To better understand what parts of debugging students struggled with the most, we broke the debugging process down into smaller pieces. We wanted to identify what types of knowledge students were missing to successfully troubleshoot circuits, so we created a series of questions that addressed the different types of knowledge. This series of questions allowed us to compare performance on individual subtasks to overall performance.

Over the course of five quarters, we iterated and improved our simulator in an introductory circuits class at Stanford University. Through this pilot testing, we discovered that most students did not successfully debug the circuit on the first try. We realized that we needed additional features in the simulator in order to keep students on track. First of all, we decided to make the activity iterative, so that students had to keep guessing until they found the correct fix for the circuit. Not only was this a more accurate portrayal of real-world debugging, where one has to keep going until you find the circuit fault, but it also gave students numerous chances, which students told us in feedback that they greatly appreciated. In order to make the activity iterative, we had to include immediate feedback, which told students each step of the way whether their answers were right or wrong. These additional features not only gave us a more rich dataset for each student but also acted as a form of intervention to help struggling students.

Our efforts culminated in the creation of a full curriculum of six different choose-your-own-adventure debugging activities through which students could choose when and where to take measurements and keep trying until they found the correct fix for the circuit. We then used the multiple terms of data from these activities to 1) show what parts of debugging students struggle with the most, 2) create a holistic view of how students approach debugging, and 3) show preliminary results about how our interventions affected student performance.

1.1 Thesis Organization

This section provides a basic overview of the remaining thesis chapters.

Without a basic understanding of the complex cognitive task of troubleshooting, it would be hard for us to create debugging tool that effectively measures and differentiates various steps of debugging. Since only limited research has been done specifically on debugging physical circuits, we first turned to the more general field of problem solving to learn more about the key cognitive aspects of troubleshooting. Chapter 2 starts with a review of prior cognitive science and education research pertaining to problem solving. We broke troubleshooting down into a series of cognitive steps or decisions one must make. We then discuss how people create knowledge structures when they learn and how people make decisions to better understand the cognitive aspects of troubleshooting. Next the chapter discusses research focused specifically on troubleshooting, which includes common troubleshooting strategies and the different types of knowledge needed to successfully troubleshoot. Finally, this chapter concludes with a look at current research on studying circuit debugging through observation in lab and show how our own online circuit debugging simulator extends and expands that research.

Chapter 3 discusses the shortfalls of paper-and-pencil tests and in-lab observation, and shows why these shortfalls necessitate the need for an interactive debugging simulator. It also discusses the difficulties of getting students to accurately share their thought processes in open-ended responses. In order to address this problem, we used multiple choice answers in an effort to get meaningful answers from students, but this brought its own potential issues. We then ran a controlled experiment to see how open-ended questions versus closed-form drag-and-drop questions affected student performance on idea generation questions. Even though the students were directly provided with the possible answers instead of having to generate the ideas themselves, we did not see a significant difference between the two groups. Based on these results, we were able to confidently move forward with closed-form questions, which allowed us to create an iterative simulator where students could keep guessing until they correctly isolated the circuit fault. Due to the iterative nature of the assignment, students were no longer measured based on the correctness of their answer but rather on the number of attempts.

Chapter 4 describes how we created our novel online circuit debugging simulator. We needed a circuit simulator that 1) included physical images of the circuit, 2) allowed students to take measurements on the circuit, and 3) recorded all student actions on the backend for researchers to analyze later. While it would have been easier to use a preexisting circuit simulator, we were not able to find any that fit all three of our requirements. Therefore, we created our own circuit debugging simulator using HTML, Javascript, PHP and mySQL. We show how the simulator was iterated and improved over time, starting with a Qualtrics quiz where students could ask for the nodal voltage in one of six places and culminating in a complete simulator where students can place digital multimeter (DMM) probes in any hole on the breadboard and receive the DMM voltage reading from the computer. In this chapter we also introduce the twelve different debugging scenarios that were created and the introductory circuits class where we debuted the debugging simulator. The full activities can be found in the appendix.

The next two chapters present our results. Chapter 5 starts with a discussion of student feedback and how well students received the debugging simulator and activities. Student feedback from the large majority of students was overwhelmingly positive with students commenting that they learned something about debugging, that they liked the iterative aspect, and that the platform was intuitive to use. We also received feedback that will allow us to improve our simulator in the future; the most common complaint was that the breadboard holes were not large enough to allow students to click on cell phones or other small touchpad devices (a use case we had not anticipated). The second half the chapter then discusses the difficulties of analyzing the massive dataset from this new type of debugging activity. It details how to create flowcharts of student actions and measurement timelines for students. Also addressed at the end of the chapter are issues we ran into with the dataset–including incomplete submissions, double submissions, missing data, and insufficient measurements–and how we resolved them in hopes of helping future researchers.

Although the brunt of the effort on this thesis was spent creating an effective debugging simulator and a full set of debugging activities, chapter 6 presents efforts made to analyze data from an education research perspective. We look at what steps of circuit debugging students struggle with the most. We then share our attempts to create interventions to improve performance on these most difficult steps, which included giving students metacognitive guidance, giving students feedback when they made wrong conclusions, and pushing students to measure all parts of the circuit. It also discusses preliminary indicators that our full debugging curriculum of six activities may have changed the way students approach debugging. Based on our datasets, we describe debugging like a building a puzzle. Each piece of the puzzle is a different type of knowledge needed to successfully debug the circuit. Missing one piece of the puzzle is not enough to obscure the final picture, but if a student is missing enough pieces, they are not going to be able to discern the final picture, namely, how to fix the circuit.

Chapter 7 concludes the thesis with a discussion of the results and next steps. After summarizing the main points of the thesis, we suggest ways we would like to improve the simulator given more time and resources. In addition, it outlines a more controlled experiment that could be conducted to more conclusively show how our full debugging curriculum is helping students improve at debugging. It also discusses how one might validate the circuit debugging simulator activities as a proper assessment that measures debugging ability in the future.

1.2 Broader Impacts

Based on the promising results of the research presented in this thesis, we hope that both educators and instructors see the utility of a circuit debugging simulator such as ours for the classroom setting. We hope the preliminary results further emphasize the importance of teaching hands-on knowledge about physical circuits and measurement techniques for effectively interacting with circuits. Traditionally tests in introductory circuits classes have only tested one domain of knowledge–theoretical knowledge. However, theoretical knowledge is but one aspect of the broader knowledge students must utilize in circuits lab. The ability to recognize a capacitor in lab or compare a breadboard wiring with a circuit schematic is equally important. We have created activities that probe these more hands-on types of circuit knowledge and show that a sizeable fraction of students struggle with questions regarding physical and measurement knowledge about circuits. Moreover, we show that students struggle with interpreting data to isolate the cause of circuit faults. We will show that guided, computerized feedback can help students successfully acquire these skills. Based on the findings of this research, we challenge the electrical engineering education community to expand beyond paper-and-pencil tests based on theoretical knowledge to include other types of domain knowledge in their curriculum and assessments.

Chapter 2

Background

In this chapter, we start by establishing that troubleshooting is a specialized form of problem-solving. This is followed by a discussion of why troubleshooting circuits is important. This discussion is based on our previous research that consisted of interviews of circuit instructors at a variety of institutions across America. Given the close relationship between problem solving and troubleshooting, we then delve into the wealth of cognitive science research and education research on problem-solving for a better understanding of what troubleshooting is and how to better teach it.

Troubleshooting circuits is an essential but inherently complex process because it requires many steps and various forms of knowledge. It also requires the troubleshooter to make decisions about what to test first and generate ideas about what can go wrong with a circuit. These various tasks can compete for limited space in the working memory, leading students who are learning circuit troubleshooting to feel frustrated and overwhelmed. There have been many previous attempts to break the troubleshooting process down into smaller, discernible steps. In this chapter we look at three different ways of characterizing problem-solving. First, we look at previous research about the cognitive steps of problem-solving (also known as general problem-solving approaches) to show how the steps identified in previous work compare with the list we created based on our direct experience working with students. Then we look at the many different types of knowledge required to troubleshoot circuits, and discuss the lack of assessment of these different types of knowledge in a traditional introductory circuits course. Finally, we will share previous research describing troubleshooting as a series of decisions and look at what decision-making theory tells us about how people approach troubleshooting.

After that we will look at techniques previously used to study problem-solving and how they might apply in our own work. Finally, we will discuss similar research on circuit debugging being conducted by a group at the University of Colorado–Boulder and how it compares with our work.

2.1 Instructor Perceptions about Importance of Troubleshooting in Lab

While we believe troubleshooting is the most important skill students learn in introductory circuits lab, we want to know whether this is a generally held belief across the teaching community. To answer this, we turned to research at the University of Colorado Boulder where they interviewed 20 instructors across the US at 18 different physics programs about their experiences teaching circuit labs [1]. Almost all instructors said that learning to troubleshoot was a major objective of circuits lab. The general consensus of the instructors was that troubleshooting "involves isolating and fixing errors using a logical, iterative, and step-by-step process." When asked what makes a student a good troubleshooter, a number of the instructors pointed to the independence and initiative by the student. They said weak troubleshooters "immediately ask for help" or "sit on their hands and wait." Around half of the instructors also said they believed that "some aspects of troubleshooting proficiency are innate."

The belief that troubleshooting is a fixed trait like IQ corresponds with multiple comments we received when discussing our research with professors of electrical engineering. Several even stated that "One-third of students will never get it," with "it" referring to the reflective practices of problem-solving. This one-third guestimate corresponds with percentages found by Maier in regard to people's ability to reflect on how they invented a solution to a problem [3]. Even when explicitly confronted and asked if actions by the researcher helped them come up with the solution, one-third of his subjects denied that the hidden hints helped.

When the circuit instructors interviewed by the University of Colorado Boulder researchers were asked about the cognitive traits of a proficient troubleshooter, many instructors emphasized the importance of having a strategy. They used words like "logical," "methodological," "organized," and "step-by-step" to describe a solid strategy. They also noted non-cognitive traits important to troubleshooting, especially the ability to regulate emotions. Good debuggers were described as patient, confident, independent, and able to overcome frustration.

The instructors also described actions by students that hindered effective troubleshooting. Standard problems included messy wires and students trying to build the entire circuit all at once instead of step-by-step with testing at each stage. When asked how they taught students troubleshooting, instructors referred to small-group interactions in lab and notes in the lab manuals about common pitfalls. With lab groups, instructors normally educated students about troubleshooting through "asking questions, coaching, verbalizing reasoning, modeling how to troubleshoot, and fading support." However, none of these instructors mentioned having a standard, formal curriculum around troubleshooting. In sum, despite the fact that troubleshooting is a key learning objective, most instructors did not assess troubleshooting ability or give homework on the topic. As we studied how students learn to troubleshoot for this thesis, one of the problems we had to solve was finding meaningful ways to measure troubleshooting ability.

2.2 Defining Troubleshooting

Troubleshooting is, simply put, problem-solving that is traditionally mechanical or electrical in nature and that occurs when a physical item is broken and needs to be fixed. In the field of computer science, such a search for problems in the code is known as debugging. Given that computer science is one of the driving forces in society, the word "debugging" has become synonymous with troubleshooting. Therefore, throughout this thesis the words will be used interchangeably.

The theoretical concept of problem-solving is described as the act of finding a path between the starting state and goal state, especially when that path is not previously known. For example, if your car breaks down in the middle of the desert, the current state is that your car is broken and the goal state is that your car is working and you are in civilization again. There may be many routes to the goal state and some may be more successful or efficient than others. Hayes described problem-solving as bridging an unknown gap:

> Solving a problem means finding an appropriate way to cross a gap. The process of finding a solution has two major parts: 1. Representing the gap–that is understanding the nature of the problem, and 2. Searching for a means to cross it. [4]

Since troubleshooting is essentially a specialized form of problem-solving, we turn to problem-solving research to better understand the thought processes involved in troubleshooting.

2.3 Troubleshooting Steps

Many researchers have attempted to divide the act of problem-solving into bite-sized cognitive steps. Their observations generally break the task of problem-solving into a planning step that outlines the scope of the problem, an execution step where the plan is carried out, and finally a reflection step. George Polya created one of the better known list of steps, breaking problem-solving down into 4 discrete steps [5]. According to Polya, problem solvers do the following to successfully solve a problem:

1. Understand the Problem
2. Devise a Plan
3. Carry out the Plan
4. Look Back

Bransford and Stein formulated a similar list a decade later, naming it *The IDEAL Problem Solver* after the Mnemonic acronym for their cognitive steps: [6]

Identify the problem
Define and represent the problem
Explore the strategies
Act on the strategies
Look back + evaluate effects of your activities

Bransford and Stein's problem-solving structure emphasizes the centrality of a careful and thorough problem representation given that the definition of the problem can influence the solution space. They also discuss working memory limitations and the importance of field-specific external representations to keep track of information in a coherent and simplified manner.

More recently frameworks similar to Polya's and Bransford and Stein's have been applied to troubleshooting-specific contexts. For example, the DESCAR method for troubleshooting follows a similar structure to Bransford and Stein's framework, but splits exploration of the strategies into two steps: search for causes and consider the solutions [7]. Just like the previous frameworks, the DESCAR method emphasizes the importance of defining the problem and search space beforehand along with reflecting afterwards.

Define the problem
Examine the environment
Search for causes
Consider the solutions
Act + Test
Reflect + develop more complete understanding

As seen in previous general problem-solving steps, reflection is central to problem solving. One study of students' problem-solving skills in the context of a circuits problem and found reflection was key to success on the problem [8]. In the experiments, participants used an interactive cartoon circuit simulator to experiment and figure out what circuit component(s) were hidden behind a black box. As part of the work, the researcher created the following diagram to represent the cognitive processes associated with solving the problem (see Figure 2.1).

Note that in this work reflection is considered so important that it makes up half the steps of their problem-solving framework. This framework also highlights how there are many types of reflection that need to occur in each step of the way when a person is debugging. For example, it is not enough for an individual to simply reflect on whether their execution of the math is right, but, rather, they must also reflect on whether the numeric answer makes sense in their general understanding of the world, as well as reflect on what they have learned in order to make it easier for them to solve problems of this type in the future.

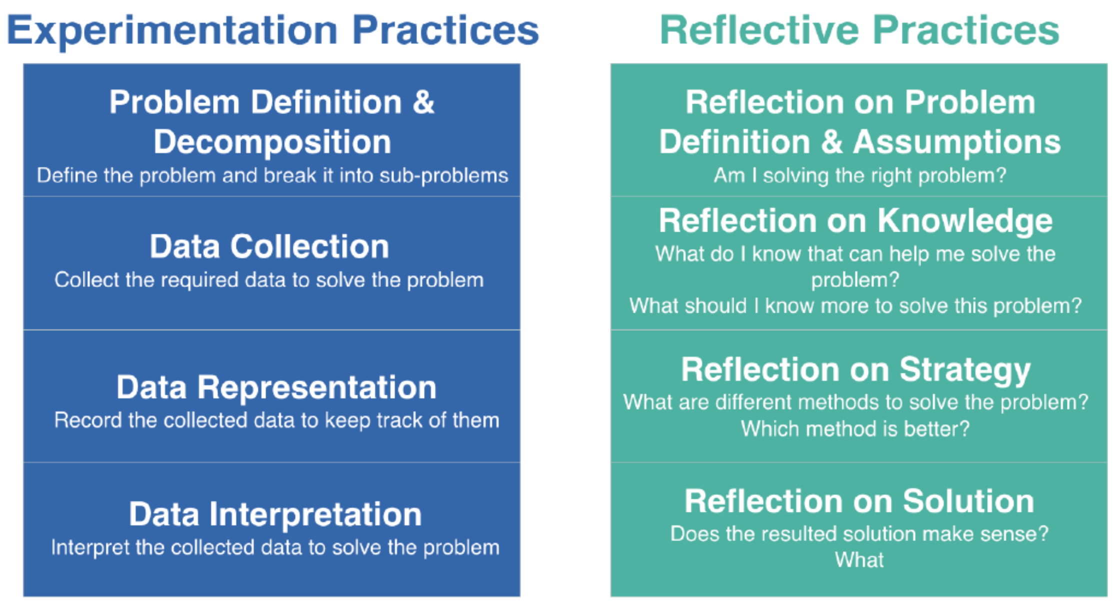

Experimentation Practices
Problem Definition & Decomposition
Define the problem and break it into sub-problems
Data Collection
Collect the required data to solve the problem
Data Representation
Record the collected data to keep track of them
Data Interpretation
Interpret the collected data to solve the problem
Reflective Practices
Reflection on Problem Definition & Assumptions
Am I solving the right problem?
Reflection on Knowledge
What do I know that can help me solve the problem?
What should I know more to solve this problem?
Reflection on Strategy
What are different methods to solve the problem?
Which method is better?
Reflection on Solution
Does the resulted solution make sense?
What

Figure 2.1: Problem-solving framework emphasizing reflection from [8].

Based our own experience watching students debug circuits in lab, we have created our own list of steps for debugging, which we used in TA training. Our list was created independently from the lists above, but it has many similarities with previous work on problem solving and troubleshooting, lending it credibility and relevance.

Troubleshooting steps:

1. Determine there is something that needs troubleshooting.
   (a) Have an expectation for what should happen.
   (b) Have a metric to decide whether the outcome is reasonable.
2. Decompose the problem and refine models.
3. Compare prediction/model and experimental result.
4. Brainstorm possible explanations for discrepancies and prioritize.
5. Test your guess.
6. Iterate until solution found.

These troubleshooting steps essentially help create a strategy for approaching the problem. For a better understanding of the strategies specifically involved in troubleshooting circuits, we referred to Jonassen's, the father of engineering problem-solving research, review article, which summarizes common debugging strategies [9]. He identified five different common strategies, shown in the next box.

1. Trial and error
2. Exhaustive
3. Topographic
4. Split-Half
5. Functional/discrepancy detection

Exhaustive refers to having a list of all possible problems with a circuit and going through one-by-one to evaluate each possible cause. *Topographic* refers to following the signal through the system either forward–from input to output–or backward–from output to input. *Split-Half* method refers to breaking the circuit into subcircuits and then checking each part. Lastly, *functional/discrepancy detection* refers to a higher order search function where knowledge of what the theoretical value should be at various points is compared against what is actually measured in order to isolate the problem in an organized fashion.

We have witnessed all of these approaches when students debug in lab and in our controlled online debugging assessments. We will later look for hints of such strategies in our own students' measurement actions.

2.4 Types of Knowledge Needed to Debug

Another way of looking at debugging is through a cognitive task analysis lens. There are many types of knowledge that one must call upon to successfully debug a circuit. Jonassen describes five types of knowledge states that novices must build competency in to become experts [9]:

1. Domain Knowledge
2. System/Device Knowledge
3. Performance/Procedural Knowledge
4. Strategic Knowledge
5. Experiential Knowledge

Domain knowledge refers to the knowledge traditionally tested in pen-and-paper tests, like whether a student can apply Ohm's Law to find the voltage across a resistor. *System/Device Knowledge* is knowing how components interact with each other. This includes the ability to read schematic diagrams and understand power flow in a circuit. It also includes understanding how a change at one place in the circuit affects other parts of the circuit. *Performance/Procedural Knowledge* refers to understanding how to use measurement tools like a DMM or oscilloscope to make measurements of what is going on in the physical circuit. *Strategic knowledge* refers to knowing what can go wrong in a circuit and knowledge of how to isolate and test the problem. Lastly, *experiential knowledge* allows one to quickly recognize and prioritize possible failures based on previous experience. All these types of knowledge are required for successful debugging.

To represent the different types of domain knowledge needed for debugging, we independently created our own knowledge domains, which we called *theoretical knowledge*, *physical knowledge*, and *measurement knowledge*. These three knowledge types align closely with domain knowledge, system knowledge, and performance knowledge, respectively. Theoretical knowledge refers to principles and calculations used to make predictions about the circuit. Physical knowledge includes recognizing physical circuit components, knowing how a breadboard works, being able to map physical circuits to theoretical diagrams, etc. Finally, measurement knowledge refers to the ability to take proper measurements with DMMs or oscilloscopes and be able to interpret that data to compare to the theoretical knowledge. It is important that we recognize all these different types of knowledge, so that we make sure to measure them all when testing debugging skills.

2.5 Decision Models

We should note that many of the previous problem-solving methods are lists of actions that people make that require judgements and decisions based on one's knowledge. Therefore, cognitive scientists have expanded the idea of technical knowledge in a field to be more than facts. According to this expanded view, each piece of knowledge in our head is stored as *schema* [10]. This means a piece of knowledge includes not only the fact, but also the context in which it is used and how it relates to other facts in an individual's head. Thus, individuals have a complex knowledge structure of many schemata in their heads.

We should note that not all researchers use the phrase schemata to describe knowledge structures. As we move away from the cognitive science community, researchers in fields such as industrial engineering and engineering management commonly refer to models or predictive frameworks to represent technical knowledge structures. They use decision trees and concept maps to show the relations of knowledge to one another. For example, in industry, expert decisions are often captured as operational procedural flowcharts in the industrial world for technicians troubleshooting on the production line. [11]

It turns out that a great amount of the troubleshooting literature currently available focuses around training technicians to fix broken products coming off the production line. One such book provides the following flowchart describing the general troubleshooting process (see Figure 2.2) [11]. This diagram, unlike most of the general problem-solving steps mentioned in Section 2.3, show the cyclical nature of the troubleshooting process. An individual might have to iterate multiple times through the process to reach the desired state.

Another technician-oriented book describes troubleshooting as composed of three different actions: (1) situational analysis, (2) problem solving, and (3) decision making [12]. The situational analysis is similar to the "define the problem" step mentioned in the general problem-solving steps in Section 2.3.

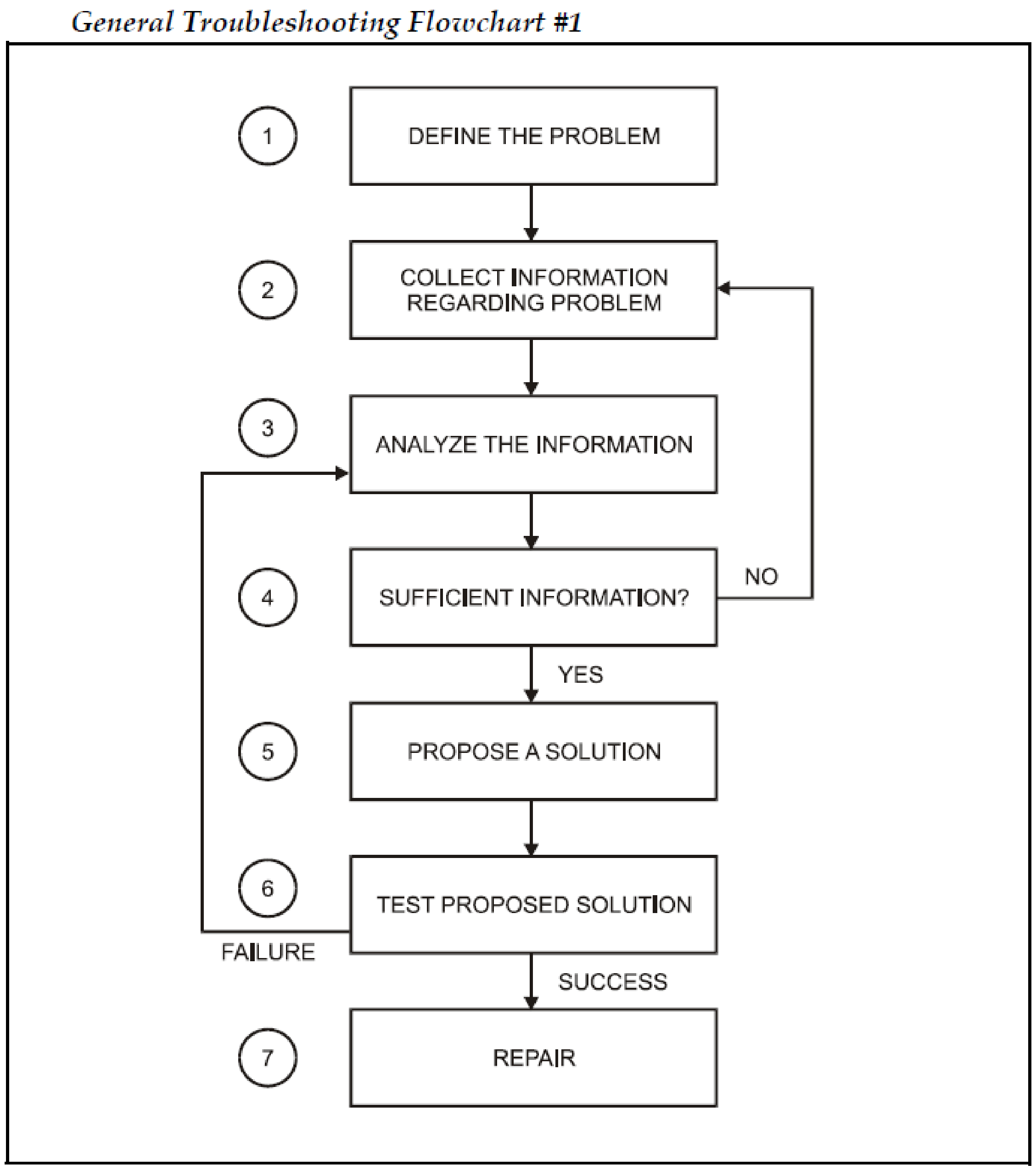


Figure 2.2: General troubleshooting flowchart from [11].

Another way of viewing problem-solving is that it is a string of decisions that must successfully be made to reach the desired goal. If we look at problem-solving as a series of decisions that need to be made, it is useful to collect a list of the common decisions scientists and engineers need to make when solving problems. Carl Wieman's research group has done just that, collecting over 50 interviews across the STEM fields [13]. Experts from academia and industry were asked to talk through the decisions they made when working on a recent project. The group then analyzed the interviews to find 29 decisions which were mentioned in the majority of all interviews (see Figure 2.3). The decisions are grouped into five main categories, which roughly match the general problem-solving steps: 1) Goals, 2) Frame Problem, 3) Plan, 4) Interpret Information + Choose Solutions, 5) Reflect, and 6) Implications and Communication of Results.

They found that these universal decisions correspond with many of choices we see students struggling with while troubleshooting in circuits lab. For example, determining important features, breaking the problem into sub-parts, deciding what additional information is needed, and comparing information to predictions are all decisions with which electrical engineering students in intro circuits also struggle.

If we view troubleshooting as a series of decisions, then it is important to know how people make decisions. Since students are time-pressured and often stressed in lab, Klein's studies on decision making under high pressure are relevant and insightful. Klein studied firefighters, EMTS, and military personal making decisions on the job [14]. This field of study grew out of an attempt to challenge the theory that people compare possible courses of action and choose one using a utilitarian cost-benefit analyses. What Klein found was that people in a stressful, time-sensitive situation usually consider only one option at a time and then determine whether it will be good enough for the situation at hand before moving on to another idea. The approach could be described as "sufficing" instead of comparing options. He also noted that a key feature of this approach is that people map every situation onto a prototype in their head. In other words, they recognize the situation as matching a general prototype in their head, leading Klein to name the model the *Recognition-Primed Decision Model*.

The idea of prototypes has many similarities with schemata. It is a form of knowledge that stores not only the facts needed, but also the context in which they appear. The general prototype includes what features and cues generally come together in that situation. For example, the speed, altitude, time and location of an object approaching can help tell a naval officer whether it is a friendly aircraft or foe missile. Similarly, wildly varying readings on an oscilloscope and loose wires might indicate a bad connection in a circuit a student is trying to debug. Learning what cues go together and being alert to a discrepancy is key to solving problems.

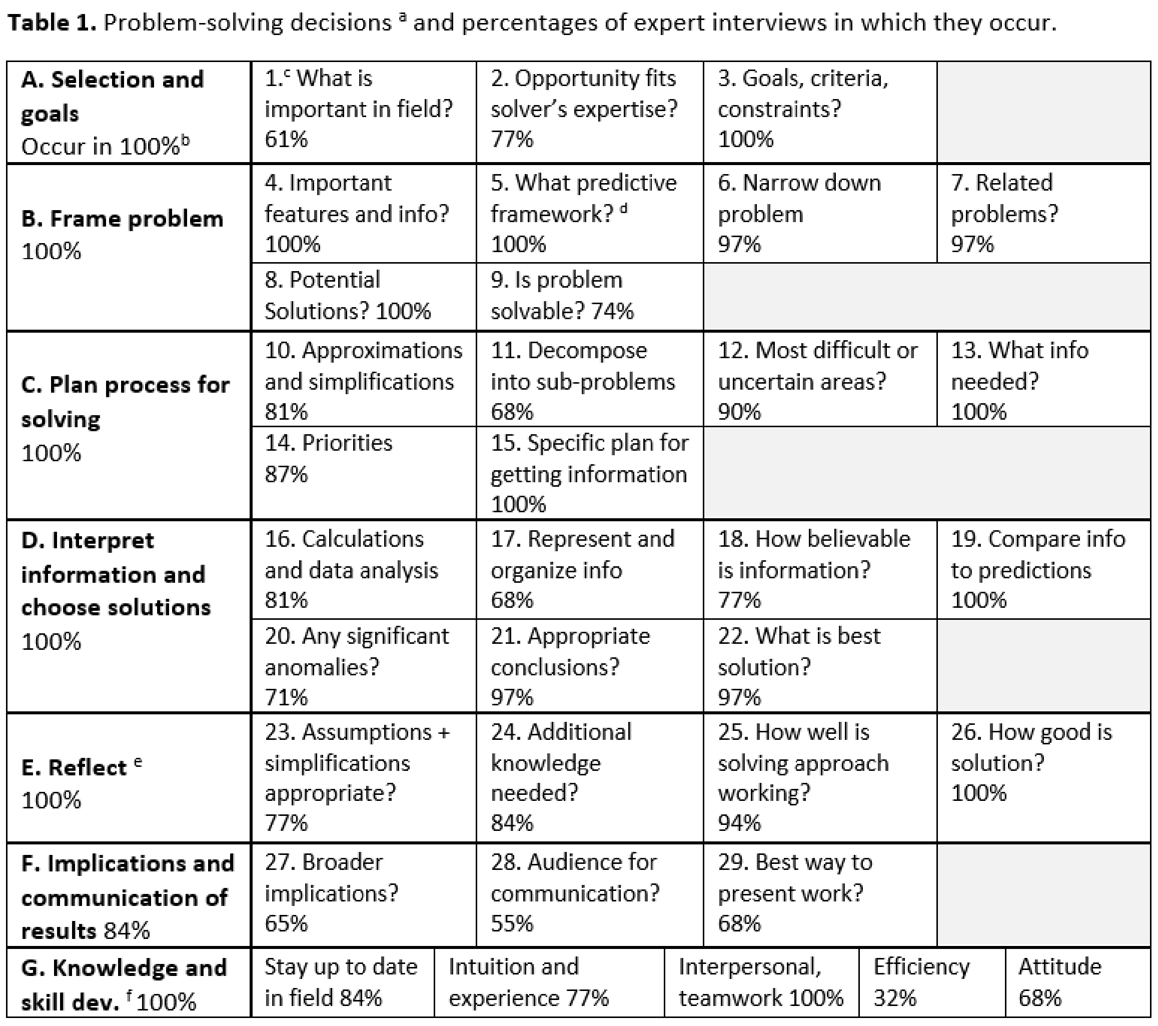


Table 1. Problem-solving decisions [a] and percentages of expert interviews in which they occur.

A. Selection and goals Occur in 100%[b]	1.[c] What is important in field? 61%	2. Opportunity fits solver's expertise? 77%	3. Goals, criteria, constraints? 100%		
B. Frame problem 100%	4. Important features and info? 100%	5. What predictive framework? [d] 100%	6. Narrow down problem 97%	7. Related problems? 97%	
	8. Potential Solutions? 100%	9. Is problem solvable? 74%			
C. Plan process for solving 100%	10. Approximations and simplifications 81%	11. Decompose into sub-problems 68%	12. Most difficult or uncertain areas? 90%	13. What info needed? 100%	
	14. Priorities 87%	15. Specific plan for getting information 100%			
D. Interpret information and choose solutions 100%	16. Calculations and data analysis 81%	17. Represent and organize info 68%	18. How believable is information? 77%	19. Compare info to predictions 100%	
	20. Any significant anomalies? 71%	21. Appropriate conclusions? 97%	22. What is best solution? 97%		
E. Reflect [e] 100%	23. Assumptions + simplifications appropriate? 77%	24. Additional knowledge needed? 84%	25. How well is solving approach working? 94%	26. How good is solution? 100%	
F. Implications and communication of results 84%	27. Broader implications? 65%	28. Audience for communication? 55%	29. Best way to present work? 68%		
G. Knowledge and skill dev. [f] 100%	Stay up to date in field 84%	Intuition and experience 77%	Interpersonal, teamwork 100%	Efficiency 32%	Attitude 68%

Figure 2.3: Problem-solving decisions universal across the STEM fields gathered by [13].

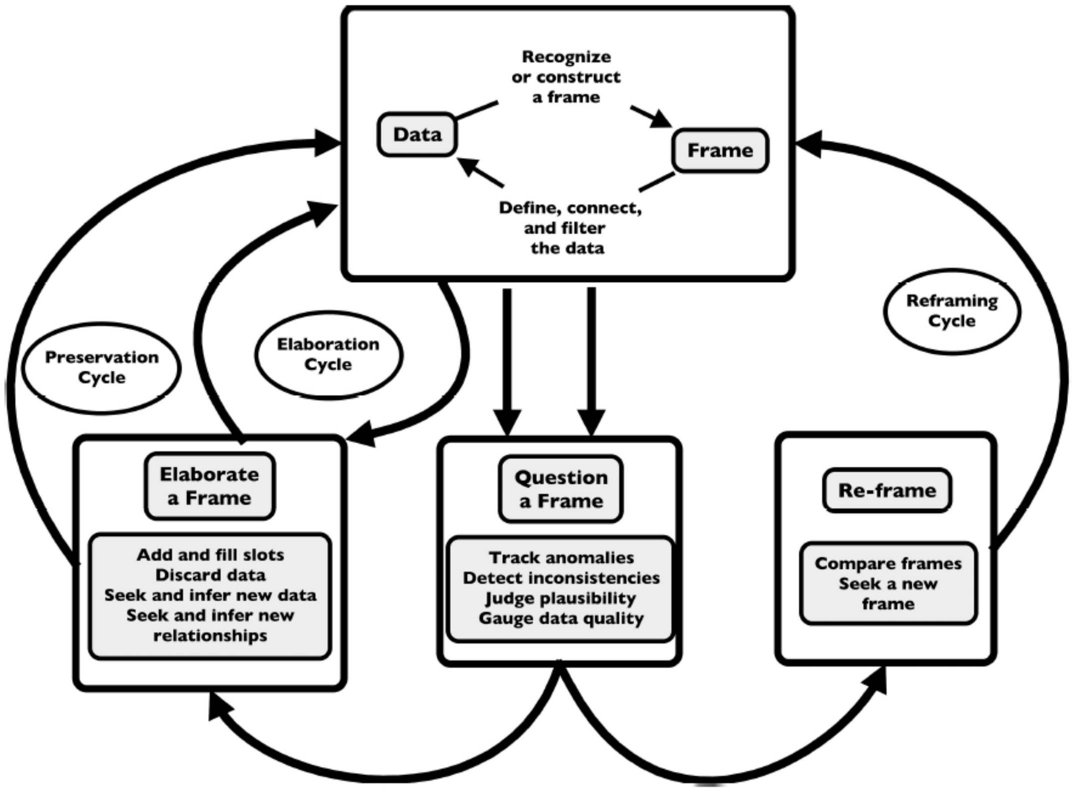


Figure 2.4: Data Frame Model from [15].

Also from these studies emerged the data-frame theory of sensemaking (see Figure 2.4) [15]. The *Data-Frame Model* looks at decision making as a comparison of two realms–the data from the physical world around us and the frame explaining it in our minds. The model also includes a cyclic element showing that when an anomaly is found, either the data or the frame needs to change to bring these in agreement. A graphical representation of this model can be found in Figure 2.4.

Combining the idea of debugging as a decision process with the idea of different types of knowledge needed to debug a circuit from the previous section (2.4), we arrive at *model-based reasoning.* This branch of research developed out of the world of artificial intelligence where researchers study how humans use models to understand the world around them in order to better help computers utilize these models and decisions [16]. In model-based reasoning, the physical and model realms are compared for discrepancies. This looks surprisingly similar to the Data-Frame model from Klein's decision-making research, only the data is now called the physical realm and the frame is now called

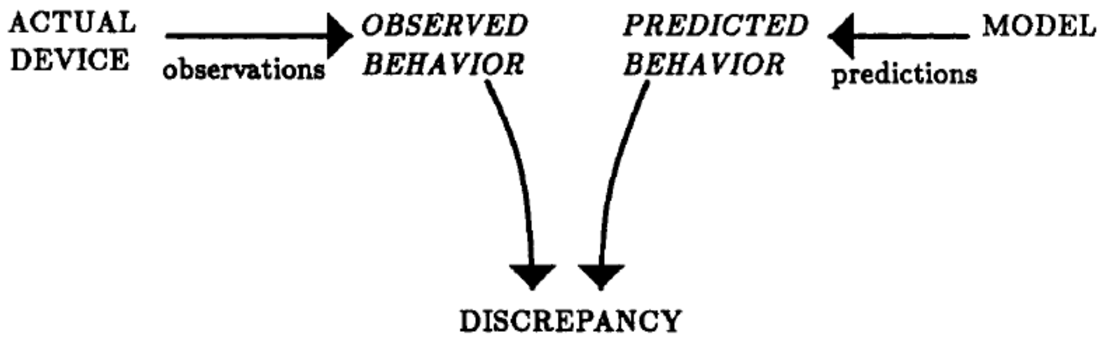


Figure 2.5: Visual for model-based reasoning from [16].

the theoretical realm. Just as with the recognition-primed decision model, we find that discrepancies are the key to solving the problem. Thus troubleshooting can be viewed as the iterative act of finding and resolving discrepancies between the theoretical model and physical world (see Figure 2.5). In the case of circuits, we use measurements of the circuit to compare these two realms.

2.6 Our Cognitive Framework

Based on the work described in the previous three sections, we created a diagram to visually summarize the cognitive processes of debugging (see Figure 2.6). We have synthesized the cognitive steps of problem-solving with model-based reasoning and embedded the various types of knowledge needed during the debugging process. On the horizontal axis in green (left-to-right), we captured the idea that debugging is essentially a search for discrepancies between the physical world and theoretical expectations. During debugging, these discrepancies are constantly being compared, and the vertical axis (top-to-bottom) contains important steps where this comparison is done. For example, one must compare theoretical diagrams with physical circuits and voltage measurements with theoretical calculations. Not only is collection of data important, but one must also have a knowledge structure of what can possibly go wrong with a circuit, which one can gain through experience or lessons. If this realm of possible problems is not large enough, then one may fail to imagine what could be wrong with the circuit and thus fail to test and find that particular circuit fault. Thus both comparing models and exploring the space of what can go wrong include using a variety of knowledge beyond theoretical knowledge. As we move on to creating our own tools for measuring debugging, it will be important to keep all these aspects in mind and include measures for them.

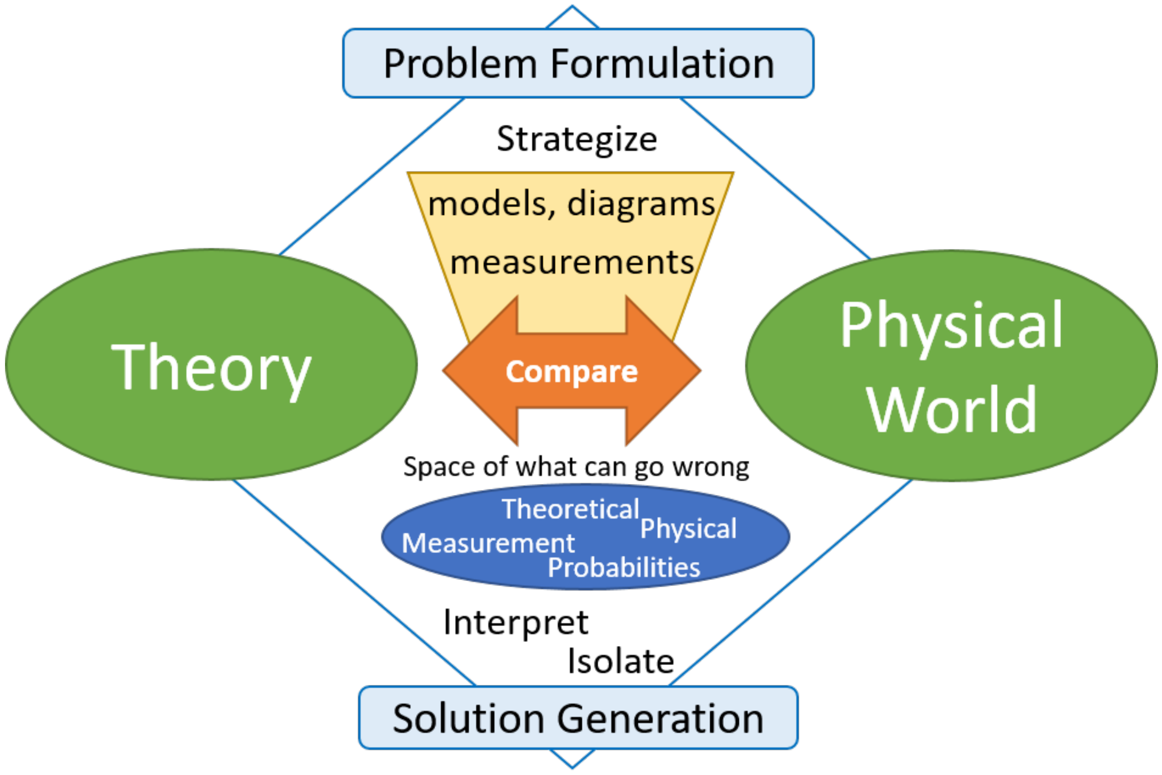


Figure 2.6: Our troubleshooting cognitive framework diagram captures the model-based reasoning of comparing the theoretical and physical worlds along the x-axis and includes the main problem-solving actions like problem formulation and solution generation along the y-axis.

2.7 Techniques to Study and Teach Troubleshooting

Having laid the groundwork for what troubleshooting is and the cognitive tasks involved, we now turn to looking at previous research into measuring troubleshooting and other forms of problem-solving. Measuring what is going on in a person's head when they are solving a complex problem is a difficult task. Researchers have used a variety of research methods to attempt to find what people are thinking about during the problem-solving process. In general, these techniques can be called *Cognitive Task Analysis* [17]. Researchers observe a subject while problem-solving, noting the actions they take. They may also collect written work from the problem-solver or ask the subject to think out loud while they work on the problem. Finally, the researcher may interview the subject and directly ask questions about their thought process while they solved the problem.

Some researchers have concluded that problem-solving in the lab setting is not authentic because the problem often has to be constrained for research purposes. Therefore, these researchers have searched for other ways to study problem-solving, which has led to an entire field of study known as *naturalistic decision making.* A primary tool of this technique is the retroactive interview about a real-world situation. Techniques have been developed to try and elicit a problem solver's decisions without over-guiding and thus biasing the subject [18]. This technique was developed based on Klein et al.'s studies of firefighters and military personal that led to the recognition-primed decision model.

Other researchers find the retrospective interviews too biased for studying problem-solving. They present evidence that people misattribute their actions when reflecting on them [3]. Given this possible limitation, we have tried to avoid the retrospective form of research when studying students' troubleshooting, although we did initially conduct some informal retrospective interviews to get a better idea of how to guide students through the process.

Another way that researchers look to measure and define good problem-solving is by looking for differences between experts and novices. Researchers have found a number of differences between how novices and experts characterize and approach the problem. For example, a seminal study by Chi et al. showed that beginning students sorted physics problems by surface features, e.g., whether the problem included a ramp or pulley, whereas the instructors given the same task sorted the problem by what physics concept was needed to solve the problem [19]. A more recent paper by Atman et al. looked at differences between experts and novices in engineering design and found that experts spent significantly more time gathering information than did the novices [20].

Experts also store spatial information in compressed form in their mind. When chess experts were shown configurations of chess boards from actual world chess tournaments, they were able to memorize the board configuration in a matter of seconds [21]. In contrast, when they were shown a board with a random configuration of pieces, they took as long as the novices to memorize the board configuration. The experts had meaningful chess configurations stored in their head as the relation of pieces to one another, but when pieces were randomly configured on the board, they had to memorize the location of each individual piece.

A brief foray into cognitive science will help illuminate this phenomenon of information compression in experts. It turns out that humans can only hold roughly seven items in their head at once [22], so as we gain more knowledge and work on more complex tasks, we start combining pieces of information into chunks in our memory. Cognitive scientists call this ability to hold only a small number of things in one's mind simultaneously *limited working memory*, and they call the combining of smaller bits of information into bigger bytes *chunking*. A good example of limited working memory in action is people strategizing solutions during decision making. In his studies of firefighters and military personnel, Klein found that people made plans by mentally simulating their course of action [14]. When people described their plans, they were usually limited to around six steps, which closely matched the seven-piece limit previously discovered. Applying these ideas to circuits, we can see how a novice might progress to a more advanced level by chunking circuit components into functional blocks. For example, they could calculate the output of an amplifier each time they came upon it using nodal analysis and the golden rules, or they could over time come to recognize that the configuration of two resistors making a voltage divider in the negative feedback path of an op amp creates an amplifier. Thus, the chunked knowledge makes a shortcut in the process of understanding the circuit, leaving space in the working memory to think about more complicated things, like multi-stage amplifiers.

Now that researchers had an idea about how memory works and how experts categorize information, they used this information to analyze the problem of how to best teach problem-solving. From studies on novices and experts, we know the importance of external representations to help deal with limited working memory. We also know that experts are adept at recognizing key features, which they then use to guide their approach to the problem. In contrast, the most common approach used by students on physics problems, for example, is to look at the desired quantity and try to work backward to find an equation (or string of equations) that will lead them back to the initial quantities they are given [23]. Such an approach is called a *means-ends analysis*, and cognitive scientists argue that this novice approach is suboptimal because students' limited working memory is simultaneously being split between cognitive processes for problem-solving and cognitive processes for schema acquisition [24]. The view of *cognitive load theory* is that instructors can manage external cognitive load and thus help students through the choice of problems, the use of non-cluttered slides, and the use of effective visual diagrams.

Other techniques for teaching problem-solving that have come out of the field of recognition-primed decision making include teaching students what factors to focus on when assessing the situation, providing practice scenarios for experience, and offering guided reflection periods [14]. Since students experience much stress and time-pressure in lab surrounding their troubleshooting, the training methods that came out of these studies may be helpful to them in troubleshooting.

The power of reflection in learning to problem solve has also been the subject of much research. In the world of cognitive science, reflection is called *metacognition*, which basically means thinking about thinking. However, a difficulty in research focused on teaching students how to problem-solve is determining how to get students to effectively reflect. As Shima Salehi noted in her thesis about problem solving and reflection, she had to make several attempts to figure out the timing and nature of the reflection questions so that students would actually engage with the material in a meaningful way [**?**]. Despite the finding that students were better problem solvers when prompted to reflect, it is interesting to note that the overall correctness of students answers did not statistically improve–only their problem-solving approach improved.

It is important to take a moment to note that the attempts above to improve student strategies and reflections always included guidance. Such guidance is called *scaffolding* in the cognitive science world. The analogy is that students' thought processes are like a building we are trying to construct. As the walls of the structure go up, scaffolds are needed to support the structure, but once the structure is built, the scaffolding can be removed. The scaffolding helps prevent cognitive overload during the problem-solving process by breaking the complex cognitive processes into bite-sized pieces that students can do. Over time, students will be able to the bridge the gap on their own and the scaffolding can be gradually removed.

A final key feature of problem-solving that is important to keep in mind when teaching and measuring troubleshooting is that an individual needs to be able to apply their knowledge in new situations. Cognitive science calls this ability *transfer*, because a person transfers their knowledge to new situations. Instructors tend to assume that students can do this implicitly, but it turns out that transfer is very difficult for people in general and even harder to teach [25]. A seminal study in the 1980's told subjects the story of how a general stormed a castle by sending troops in from three different paths to converge on the castle [26]. The participants were then asked to solve a problem about how to radiate a cancer tumor inside a human without destroying the surrounding healthy tissue. The thought was that subjects would transfer the idea of converging troops to converging beams of light in the cancer radiation case. It turns out very few people were able to do this spontaneously, but when a second analogous story about converging paths was given beforehand along with the castle storming example, subjects were better able to understand and use the underlying principles to solve the problem. Such analogs are called *contrasting cases* and are one of the best ways we know to help students transfer their knowledge [27].

2.8 Previous Circuit Debugging Research

Having surveyed the cognitive science behind problem-solving and how researchers measure problem-solving, we next looked to other circuit-specific debugging research for guidance and inspiration. It turns out that we are not the only ones studying the cognitive aspects of students debugging circuits in introductory courses. The Lewandowski group at the University of Colorado at Boulder has also studied students troubleshooting multistage op amp circuits in pairs in lab [2]. In the Boulder group's study, eight pairs of students debugged the circuit over the course of 50 minutes with all their actions being video recorded. The researchers were specifically looking at what type of cognitive tasks students were engaged in throughout the debugging session. They used the following 4-step cognitive framework to represent the debugging process:

1. Formulate Problem Description
2. Generate Causes
3. Test
4. Repair and Evaluate

All groups engaged in all four cognitive activities throughout the debugging session. The first half of sessions tended to be dominated by formulating the problem description and testing; the second half tended to be dominated by generating causes and repairing and evaluating. A separate paper has been published by the group specifically discussing how they analyzed the data to turn open-ended statements from students thinking out loud into scientifically sound categories of cognitive thinking [28].

The Lewandowski group has also looked at students' use of model-based reasoning during the debugging process. They created a visual flowchart to explain how students approach debugging (see Figure 2.7. This diagram accurately captures the iterative process of troubleshooting and how it focuses around the comparison of physical and theoretical models. The bottom of the chart also shows how many different types of problems can lead to the need to troubleshoot, which hints at the different types of knowledge from cognitive task analysis and indicates the importance of reflection.

Students were found to use model-based reasoning in two primary cases when troubleshooting the two-stage amplifier circuit. The first case was the choice to split the circuit into two sub-systems–the first amplifying stage and the second amplifying stage. The other point at which students commonly used model-based reasoning was in their decision to try replacing the op-amp to see if it would fix the problem.

These results from the Lewandowski group correspond with our understanding of debugging, but, as detailed in subsequent chapters, we have expanded on these results and shown they also apply to large numbers of students by using computerized assessments. It turns out the Lewandowski group had a similar idea, and over the past three years they have been creating an assessment called MAPLE, which stands for Modeling Assessment for Physics Laboratory Experiments [29]. The group

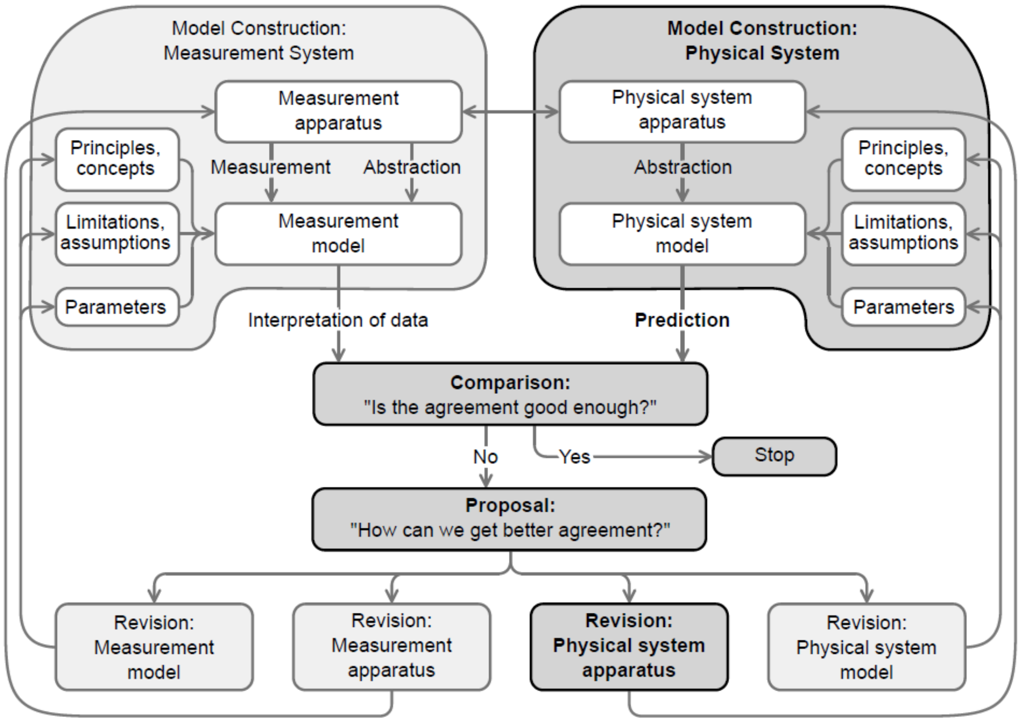


Figure 2.7: Modelling framework for experimental physics by the Lewandowski Group at University of Colorado at Boulder [2].

finished creating the assessment in spring 2021 and are now working to validate the assessment. The complete assessment and validation is not yet published, but conversations with the researchers confirm that their debugging simulator shares several similarities with our own. The Lewandowski group has also created a computerized assessment that works like a choose-your-own-adventure story. Students can choose different actions to perform on the circuit, and the computer gives back the results in an interactive, text-based adventure in Qualtrics. They have tested the system on hundreds of students at multiple different institutions and are preparing to release their results. However, there are several key differences between our work and MAPLE. MAPLE has worked on perfecting a single assessment on a multi-stage amplifier, whereas we have focused on creating a a multi-activity curriculum that also functions as an intervention. And while MAPLE is attempting to create a validated assessment, our primary objective was to create a new debugging platform. Moreover, while MAPLE is an entirely text-based adventure, we have a graphical interface that allows students to virtually place DMM probes and take their own measurements on a breadboard circuit. More information about our circuit debugging platform can be found in chapter 4, but first, the next chapter outlines why we designed our interactive circuit debugging simulator the way we did.

Chapter 3

Measuring Complex Decisions with Computers

Now that we have defined troubleshooting and discussed the cognitive processes involved, in this chapter, we discuss a meaningful way to measure circuit debugging. To create the best tool for the job, we first needed to understand the specific students we would study. For this work, we were given permission by instructors to study the students in the introductory circuits lab course at Stanford University, and the first section of this chapter goes into greater depth about that class.

After that, we will outline our attempts to elicit insights into the debugging process using debugging journals. Based on the shortfalls of students' reflection on their own debugging in their journals, we moved to online, interactive activities where we could use their choice of measurements as a proxy for their thought process. We also ensured that our debugging activity addressed all three types of knowledge we identified–theoretical knowledge, physical knowledge, and measurement knowledge. From these initial online pilot studies, we discovered that most students struggle with debugging, which necessitated a mechanism to keep them on track during the debugging process. Therefore, we created iterative, choose-your-own-adventure type debugging activities where students had to keep trying until they got to the correct fix for the circuit.

Lastly, this chapter discusses how we searched for an effective way to give immediate feedback. We needed a way for the computer to quickly and easily parse the students' answers, so we used multiple choice answers based on previous open-ended student responses. Because multiple choice questions involve analysis rather than idea generation, we were concerned that the multiple choice questions might affect student responses. Therefore, we conducted experiments to compare performance on brainstorming what could be wrong with a circuit when subjects were given open-ended responses versus choosing causes from a drag-and-drop list. The null result of this experiment allowed us to create our interactive circuit debugging simulator with confidence that our closed-form solutions were not affecting student responses, at least not in the short term.

3.1 Students studied

Our studies focused on an introductory electrical engineering class at Stanford University. The class, *What is EE?: An Intro to Making*, is known by its class number ENGR40M to students. Because Stanford is on the quarter system, the students get a compact 10-week overview of electrical engineering from nodal analysis to frequency domain representations and digital logic. The full course syllabus can be found in Appendix C. The lab-intensive class is centered around four lab projects that students get to keep at the end of term: a solar charger, a useless box [30], a programmable 8x8 LED display, and and an EKG (see Figure 3.1). Motivation around and details about the design of the labs can be found in a previous publication [31]. Students spend 3 hours a week in lab with 10 students per TA, though many come to open lab hours to finish. The class is very popular among undergrads and is offered every term, with around 400 students taking the class each year.

Note that throughout the remainder of the thesis, we will refer to specific terms during which we gave students various debugging assignments. Because Stanford is on the quarter system, these terms are referred to as fall, winter, and spring terms. We have been giving students debugging assignments since Fall 2019, and this thesis includes data from five different terms. In Spring 2020, we did not collect data when classes were moved online abruptly due to COVID-19 and the lab component of the class was cut. Starting in Fall 2020, labs were included in remote learning. Each student received a lab kit in the mail including a soldering iron, digital multimeter (DMM), and Analog Discovery 2 so they could do all the labs at home. For those readers who are not familiar with these materials, the Analog Discovery 2 (AD2) is a device that can connect to a computer and function as a power supply, waveform generator, and oscilloscope [32]. Students were allowed to keep the projects they built, as well as the soldering iron and the DMM at the end of term, but they were required to return the AD2 due to its cost.

The remote conditions ended up providing a perfect environment for us to study debugging. The only interactions with TAs was through a Zoom screen, so students had to fix the problems with the circuits themselves because a TA could no longer step in for a few minutes to fix something. Therefore, the teaching staff turned to our team, which had already been studying debugging, to discuss how to best teach debugging. An electronic lab journal helped students articulate the problem they were observing with their circuits to their TA. If they were stuck, the journal also guided them a checklist of common problems with circuits. This list included information like checking that their power supply and oscilloscope were on, along with checking whether components were hot.

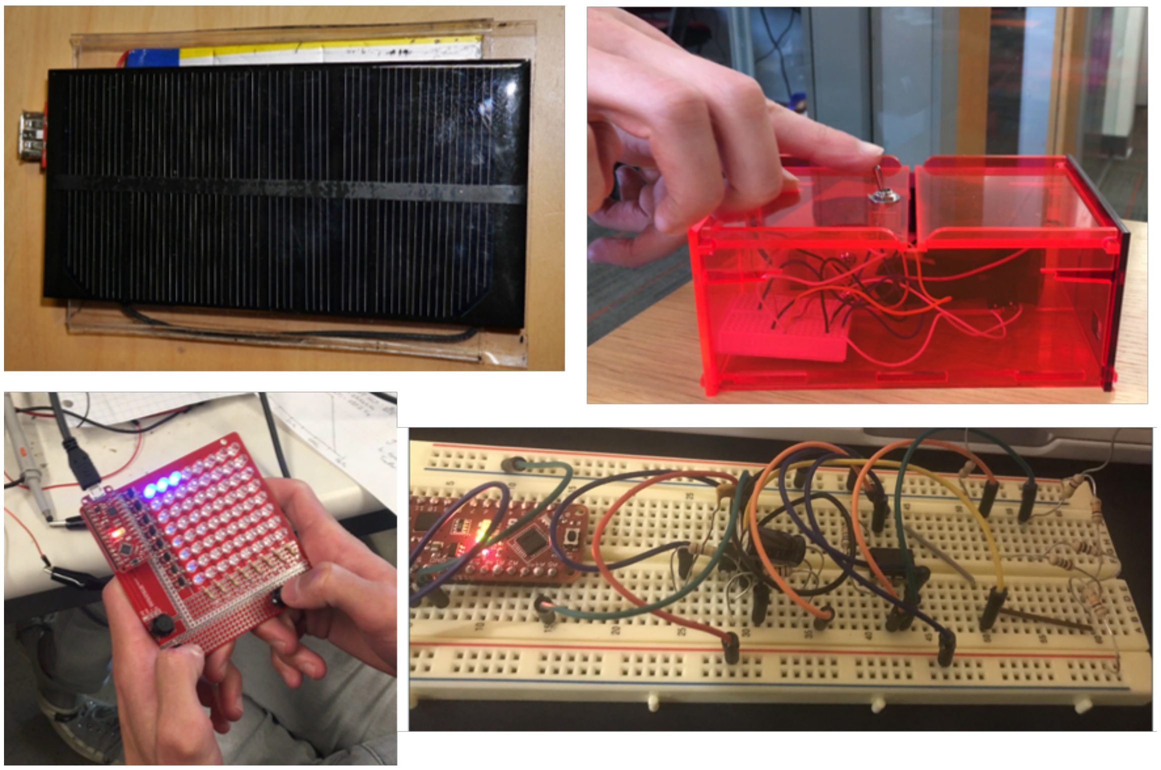


Figure 3.1: Lab projects in Engr40m. **Top Left**: Solar Charger, Lab 1. **Top Right**: Useless Box, Labs 2a and 2b. **Bottom Left**: LED display, Labs 3a and 3b. **Bottom Right**: EKG, Labs 4a and 4b.

Due to the limited interaction students could have with TAs during remote labs, we expanded our debugging curriculum in winter 2020 to be a weekly assignment, so that students would get practice and exposure each week to debugging. By choosing some common problems we had seen in lab for these debugging assignments, we hoped to help students before they completed a lab. Students responded positively to these debugging assignments, which we will discuss in more detail in chapter 5.

3.2 Need for an interactive simulator

Ideally, we would study students physically troubleshooting circuits in lab, but such a study tends to be more qualitative in nature and is hard to scale to large numbers. As a test run on the feasibility of such an approach, we recorded the TAs debugging circuits in small groups during TA training multiple times, but the data was tedious and time consuming to analyze. New ideas were interjected only very sporadically, and most of the time was spent with the TAs "fiddling" with the circuit in an unclear way on the video-recordings. Since the University of Colorado Boulder group had already conducted such studies on a small-scale, we decided to move to methods where we could quantitatively and quickly analyze large numbers of students.

The standard college class currently assess large numbers of students through paper-and-pencil assignments and exams, so we attempted to adapt such a medium to measure troubleshooting. To test the feasibility of student written work, we collected debugging journals from hundreds of students over multiple terms. While the journals were enlightening, we found they did not work well for quantitative analysis for two main reasons. First, many students completed the debugging journals after the fact, even when encouraged to do them in lab, so their description of the problem was often hindsight-biased. Second, there was a large variation in how much students wrote and the effort they put in to their journal, so it was hard to tell the difference between a student who knew what they were doing and just didn't write much and a student who had no idea what they were doing.

Nevertheless, we were able to glean some insights from these debugging journals, which influenced later experiments. For example, a number of the students expressed in their debugging journals that one thing they learned was that external factors, and not incorrect wiring, can create the need to debug. For example, a bad batch of op-amps may take you hours to debug if you assume that all the individual components are working. That is, our students appeared to many times assume that if a circuit was not working, they must have wired the circuit incorrectly. This assumption narrowed the search space in a way that the student might never identify the problem because they were searching in the wrong place.

These results gleaned from looking at the debugging journals are not surprising. While paper-and-pencil may be the expected form of assignments, this medium has several major limitations if we are attempting to measure more complex cognitive process like troubleshooting. Researchers have studied the shortfalls of measuring student thought processes through paper-and-pencil assignments by both video-taping and collecting the written work of students while problem-solving [33]. Using a problem-solving skills rubric, the researchers compared the score of a student based solely on the student's written work versus also using the video footage for data. They found that many higher-order problem solving skills, such as construction of models, assumptions, expansion of equations, and checking work, were not visible in the written work. Another problem with paper and pencil is that the students can erase previous answers, and we cannot determine how they changed plans in response to new information.

Since observing hundreds of students debugging in lab is infeasible, and written reports from students are unreliable, we needed to create an alternate way of recording students' interactions with the circuit. We decided to create a interactive online debugging activity that captures the most important aspects of the troubleshooting process. We specifically wanted to look at students' measurement strategy and how they responded to and incorporated new information. By having the activity on the computer, we could prevent students from erasing their answers later. We could also give them more information after their initial open-ended answer. This allowed us to know what they naturally thought about before we potentially biased them with directed questions or more information.

3.3 Simulator Goals

When we began this research, we started with just a few requirements for our interactive, online debugging activity. We wanted the simulator to accurately model the real world, which meant including physical representations and measurement representations, which are not traditionally included in assignments. We also wanted the debugging simulator to be easy-to-navigate, so students would not spend their limited cognitive load on figuring out how to control the simulation. By capturing both students' written work and actions, we hoped to create a more complete picture of students' thought processes. Our hope was that through this realistic simulator, we would discover which domains of knowledge students struggled with the most while debugging, so we could create modified activities in the future that honed in on those specific aspects and provided a space for deliberate practice.

Just as Jonassen and Hung identified knowledge states [9], we aimed to design the debugging simulator to probe students' debugging knowledge in four different states, which we call theoretical, physical, measurement, and debugging heuristics knowledge. Theoretical knowledge represents the theoretical knowledge that students must have to solve the debugging problem. This includes understanding how to calculate gain or knowing whether a transistor is on or off in a digital logic circuit based on its gate voltage. Physical knowledge refers to the ability to identify physical circuit components and being able to map physical circuits to the theoretical diagram. Measurement knowledge refers to the the ability to correctly connect and take measurements with traditional electrical engineering lab bench tools like a DMM and oscilloscope. Finally, debugging heuristics refers to one's knowledge of what can go wrong with a circuit and the frequencies at which these problems occur, along with having methodical strategies for isolating the problem in the circuit. The final category is often gained through debugging experience.

To provide a realistic simulation, it was of utmost importance that the activity include aspects from all four of these knowledge domains. Including physical and measurement knowledge was different from current homework and test problems, which usually test theoretical knowledge. For physical knowledge, we included actual pictures of breadboards. Physical representations of the circuit allowed us to test students' physical knowledge about circuits and their ability to map the physical circuit back to the theoretical schematic. For measurement knowledge, we tried to replicate the process of using a DMM, where students had to place the red and black probes to get measurements from the circuit. The measurement tools in our debugging simulator required students to show that they knew how to properly use measurement tools and interpret the data from them.

We concluded the best way to make the system easy-to-navigate was to make the measurement process as natural as possible. We wanted to allow students to click directly on the physical breadboard representation to place the red and black probes of their DMM. The probe wires would then appear, graphically attaching the DMM to the point of measurement and the reading would appear on the DMM.

To create a more complete picture of the debugging process, we collected both written work and measurement data. In an attempt to elicit more insightful written responses, we tried asking students for their tentative conclusions in the middle of the assignment. This ended up giving us more insight into how students were drawing conclusions, because their thought process was not yet complete. Therefore, they shared observations they made in "real time" instead of the whitewashed version of their conclusions they gave at the end, which ignored any irregularities and focused only on the data that matched their final conclusions.

Finally, from a research perspective, we wanted to find a better way to collect information about students' thought processes than just written work, which can be vague and inaccurate. We used inspiration from model-based reasoning, which was mentioned in Section 2.5. Model-based reasoning describes debugging as the act of repeatedly comparing measurements and other data from the physical circuit with theoretical models and looking for discrepancies. Since measurements are such an integral part of this process, we wanted to sequentially record students' actions each step along the way. Furthermore, because students are often reluctant to write down beginnings of ideas or things they are unsure about, we found that recording students' actions could serve as a proxy for their thought process. By recording what actions students took in what order, we opened up a entirely new dimension for analysis. We were also able to record their mouse clicks and the order in which they took measurements to get a more quantitative idea of how students interacted with the circuit. The measurement order and timing provided us with a quantitative indicator of strategy.

3.4 Using Computers to Interpret Student Work

After running our preliminary studies, we found that most students struggled with debugging and were not able to correctly diagnose the circuit fault when it was not a visible miswiring. Therefore, we realized that our debugging simulator also needed to give students feedback. We needed a way to keep students on track when they made mistakes, so we could still get meaningful data about their debugging process even after a mistake was made. This requirement meant that the system needed to "understand" the student's written work so that it could provide useful feedback to the user.

Therefore, we thought of three different ways to potentially automate the process of parsing student answers, which would allow the computer to give responses in real time. The first idea was to use a natural language processor to parse student written work. The second idea was to use questions that were not open-ended, like multiple choice questions and ranking questions, to allow the computer to react to student responses. Third, we investigated using measurements and other actions as proxy for thought process and ignoring students' written work completely. Each solution has its own complications and trade-offs, which are discussed below.

First of all, we want to acknowledge that interpreting written work is a difficult problem, even for humans. As any instructor knows from grading, deciding what a student meant based on a vague answer is often difficult. To fairly compare all answers as a researcher, one must read each written response and "code" it, which means categorizing the responses into a short list of common answer types. This process is time-consuming, qualitative, and not scalable to hundreds of students.

Natural language processors have shown promise in the last decade at parsing student written work due to improvements in machine learning and neural networks. For example, researchers have successfully used neural networks to auto-grade open-ended questions on citizenship tests [34]. We were able to successfully use basic text parsing in Qualtrics to parse students' relatively straightforward answers about the purpose of an op amp circuit, but students' paragraphs about final reasoning were too complicated for that system. Since making a natural language processor for students' debugging reasoning could be a thesis in itself, and no one on our team had the skillset to implement cutting-edge natural language processors at the time, we chose a different approach for analyzing student data.

Therefore, we turned to our second idea, which was to replace student open-ended essays with closed-form answers to make it easier to analyze the large data sets. Closed-form answers include multiple choice questions along with drag-and-drop questions. Creating good multiple choice answers took several academic quarters of effort on our part. We would start with an open-ended question one term and then categorized student reasoning to create a multiple choice version the next term. This technique is similar to the techniques used to make concept inventories [35].

Many of our multiple choice questions differed somewhat from standard multiple choice questions, because our questions often did not include a “correct” answer, but rather presented a variety of actions or decisions one could make, some of which were more or less helpful to a student trying to identify what was wrong with the circuit. We tried to constrain the cognitive load by limiting multiple choice questions to six answers or fewer. If more options were available, we would include a first question by category and then have sub-questions to drill down further. The downside of this approach was that some students were uncertain about which category their answer fit in and may have clicked the wrong category and ended up in the wrong sub-question group.

We often used drag-and-drop questions when students were asked to brainstorm possible causes, so they could sort ideas into “possible” and “not possible” and then prioritize the possible ideas into most important to least important. These kinds of problems are similar to Parsons problems in computer science, where students drag and drop lines of code into coherent structures instead of writing their own code. Studies have shown that Parsons problems are an effective alternative to written work on exams [36]. We ran experiments giving students the drag-and-drop questions versus open-ended brainstorming questions and found no difference in performance between students given open-ended questions versus drag-and-drop questions (see Appendix A). Therefore, we felt comfortable using drag-and-drop and multiple choice questions in the final versions of our debugging activities.

The final idea we looked into was using students’ actions to help us better understand their thought process while debugging. This allowed us to observe if students were taking measurements randomly or if they were using one of the standard systematic strategies, like checking the input and output first or checking the power supply first. For circuits where the problem was not a miswiring, we could also seek confirmation of whether a student isolated the cause by whether they took multiple measurements around the problem point in the circuit. We could also observe what action they next took after gaining new information to determine whether they interpreted the new information in a meaningful way. These examples are just a few ways in which students’ actions complement their written reasoning, allowing us a fuller picture of the students’ thought process during debugging.

We have just shown three different ways that we use computers to parse student responses in real-time. For simpler open-ended questions, we can have the computer search for key words to determine whether a student has correctly answered the question. For more complex open-ended answers, we can convert them to multiple choice questions and other closed-form answers, which allows the computer to immediately parse the answer. Finally, we can gather other information besides written work, such as capturing the sequence of measurements a student took, as an indicator of a student's strategy and thought process. Together all three of these tools allow us to quickly and effectively measure data for hundreds of students.

3.5 Choose-Your-Own-Adventure Activities

One other innovation we made to our debugging simulator based on preliminary data was to make to our debugging simulator iterative. In other words, students were required to keep guessing until they found the correct fix for the circuit. We call this version of the activity choose-your-own-adventure, because everyone ends up at the correct answer at the end, but each takes their own path. We found that formatting the activities to repeat until the student successfully debugged the circuit was helpful for three main reasons.

First of all, students liked the iterative aspect. In order to make students keep guessing until they got to correct fix, we had to give them immediate feedback that their answers were incorrect. Students appreciated the immediate feedback and said it kept them motivated. They liked knowing when they had reached the correct answer and they liked getting graded on completion rather than correctness the first time round. Chapter 5 discusses further the positive responses we received from students about the iterative aspect of the debugging simulator.

Secondly, the iterative debugging simulator is more authentic. In the real world, debugging is not complete until an engineer finds the cause of the circuit fault. Since most students were not identifying the circuit fault on the first try when the problem was not a visible miswiring, we felt that the original structure of our activities was not realistic enough. Therefore, we created the iterative version of the activity to better mimic real debugging.

Finally, the iterative, choose-your-own adventure style allowed us to collect more complete data from struggling students. As mentioned previously, we faced a major hurdle about how to collect useful data for struggling students. If a student made a mistake early in their work, like not knowing how to calculate the theoretical gain for the amplifier, then the rest of their measurements and reasoning often varied greatly from their peers and was hard to analyze. Furthermore, an incorrect conclusion earlier often affected a student's performance on other, later parts of the activity. In this case it was hard to differentiate whether the student did not have the domain knowledge for the later part of the activity or whether their mistake on the later part was just carry-over from an earlier mistake. It would be more effective if we could note when a student makes a mistake and then get them back on track, so that they start from the same place as other students later in the activity. In this way, we could see how they perform in later cognitive tasks without their reasoning and work being obscured by previous mistakes.

To illustrate how the new iterative (or looping) system allows us to obtain more information about a student's thought process after the initial error, let us look at two examples. The first is that many students stop taking measurements after measuring the input and output of an amplifier circuit and state that they know the amplifier is not working because the gain is not as expected. While this is an accurate observation, no effort has been made by the student to figure out how to fix the circuit. Therefore, we can give feedback and force the student to keep looking for the problem with the circuit. The second example is that students will say the circuit is working even when it is not and give all correct evidence in favor of their explanation–perhaps they say they checked the wiring and that the amplifier successfully amplifies without doing a quantitative comparison of of the theoretical and empirical gain. Again, with the new iterative system, they are given feedback that their answer is incorrect, and we get to see what they do next in response. Previously, we had no data on what these students thought was wrong with the circuit, because they never got past realizing nothing was wrong with the circuit. With the iterative version of the circuit debugging simulator, we get comparable information for **every** student because each student has to keep trying until they realize what is wrong with the circuit. We call the feedback that their answer is incorrect *guardrails* because it keeps students on track.

Although we obtained more useful data from the iterative, choose-your-own adventure activities, creating these activities posed some significant technical challenges. First, we had to find a sensible way to break down the looping. We could have students work the whole problem through to the end and then, if they got the wrong cause for the circuit fault, we could send them back to measurements and other testing until they had another idea. However, while we have used that method in various assignments, we decided that more feedback would be helpful for students, so we broke the debugging task down into cognitive tasks. Each subtask would have a "checkpoint" at the end and give students feedback that their answer was incorrect so that they would have to keep trying at that checkpoint until they got it right. We used the following standard checkpoints: what they expected the circuit should do, whether they could use the measurement tool, whether the circuit was working, what measurements indicated it was not working, and how to fix the problem. Implementing the checkpoints with the looping structure was a difficult task, and we will discuss the technical details of this in chapter 4.

Not only did making the debugging activities iterative make them technically more difficult to build, but it also changed how we analyzed and interpreted data. Because all students get to the correct fix for the circuit in the end, we could no longer use correctness of final answer as a metric for performance on the debugging activity. Instead, we used the number of attempts at each of these checkpoints as a measure of how well a student was doing on the assignment. A student who needed multiple attempts at each checkpoint would be labeled a struggling student, and a student who breezed through each checkpoint on the first try represented a high performing debugging student. Because such an activity is defined by the path a student takes, rather than the destination (the correct answer), we had to create new techniques for analyzing and comparing student performance, which we discuss further in chapter 5.

3.6 Summary

As we have shown in this chapter, it is difficult to get hundreds of students to share their inner thoughts and reasoning on assignments in a way that is feasible for researchers to quickly analyze. Without the guidance of cognitive science, which allows us to break down the complex process of decision making into smaller steps, and advancements in user-friendly web interfaces, this endeavor would not have been possible. We believe we have provided three main novel and effective approaches to measuring complex debugging decisions by means of computers: 1) recording the order of students' actions and measurements as a complement to student reasoning, 2) breaking debugging down into cognitive domains to allow for more fine-grained measurement, and 3) creating guardrails that allowed students multiple chances and differentiated students based on journey rather than final answer. Together, these three approaches allowed us to create a rich data set for each student that gave us important information about their thought processes during debugging. We were able to observe what effect mistakes on individual steps can have on the whole process and what smaller decisions best correlated with success on the debugging assignments.

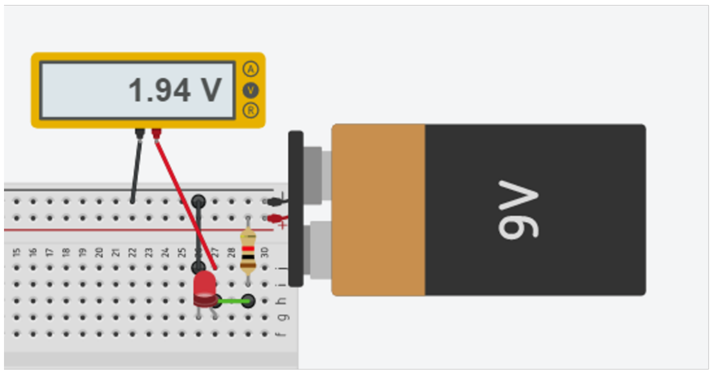
Since our innovation of guardrails (immediate feedback to keep students on track) was implemented through the use of multiple choice and other closed-form answers, we decided it was necessary to test whether or not the closed-form answers were artificially changing students' answers. We made sure there was no significant difference in performance when students were given closed-form questions versus open-ended questions before converting our debugging simulator completely to the guardrails and choose-your-own-adventure style. The details of the experiments on the effects of closed-form answers for idea-generation questions can be found in Appendix A. The results showed that students were no more likely to correctly identify the cause of the circuit fault when given a drag-and-drop list of ideas than when they had to brainstorm the causes themselves. At this point we were ready to build our circuit debugging simulator, the details of which we discuss in the next chapter.

Chapter 4

Designing the Circuit Debugging Simulator

In the last chapter we outlined the requirements of our online circuit debugging simulator. This chapter focuses on the technical details of fulfilling those requirements. The simulator needs to be easy-to-use and allow students to intuitively interact with the circuit. It needs to record all actions students took for later analysis by the researchers. It needs to provide various representations of the circuit to students, including the schematic, physical breadboard, and measurement representations. The system also needs to give students immediate feedback, both to help struggling students and to provide a more complete data for each student from beginning to end of the problem. By capturing the journey rather than the destination, the system will also open up new possibilities for better understanding the thought processes of students as they debug.

Having outlined the type of computerized system we aimed to design, we now describe our design of it. First, we looked at preexisting systems to see if we could modify any of them to meet our needs, the details of which are discussed in Section 4.1. Since none of the systems fulfilled all of our requirements, we ended up building our own circuit debugging environment. Our simulator underwent several iterations before reaching its final version. We began building our simulator in Qualtrics during our pilot studies. Due to limitations with Qualtrics, we moved the activity to our own website, so that there could be a more intuitive interface for taking measurements. After that, we added feedback and a looping structure that required students to keep guessing until they found the correct fix for the circuit fault. These efforts culminated in a full curriculum of six debugging activities, which were tested in two different quarters in ENGR40M, the introductory circuits class at Stanford University. Each activity is outlined later in this chapter.



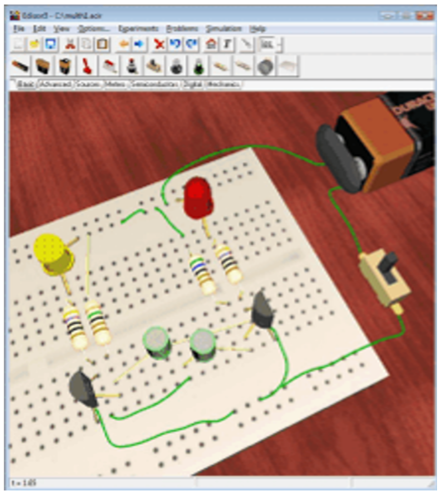


Figure 4.1: Alternative Breadboard Circuit Simulators. **Left:** Circuits for Tinkercad by Autodesk. *Image Courtesy of* `www.tinkercad.com`. **Right:** Edison – Photorealistic Multimedia Lab for Exploring Electricity and Electronics. *Image Courtesy of* `http://www.designwareinc.com/edison.htm`.

4.1 Preexisting Circuits Platforms

We evaluated Spice, Fritzing, Tinkercad, Qualtrics, and Edison Electronics Educational Software for use in our debugging simulator. We assessed them for the following three requirements: 1) inclusion of a physical view of the circuit, like a breadboard, 2) option for students to make changes to the circuit, and 3) ability to record students' interaction with the circuit. Here are our findings on each and their suitability to our circuit debugging activities.

At first glance, the closest fits were Tinkercad and Edison Electronics Education Software. Circuits for Tinkercad is a free educational platform offered online by Autodesk [37], which has a sleek modern interface with easy to drag-and-drop components (see Figure 4.1). When the simulator is started, motors will turn and LEDs will light up, which provides a good visual feedback for students. Edison Electronics Education Software is desktop computer program that offers similar drag-and-drop functionality with slightly outdated graphics and interfaces [38] (see Figure 4.1).

Both Tinkercad and Edison provide physical representations and have a circuit simulator back end, which fulfills two of the three requirements. Unfortunately, both are proprietary, which would have led to a data collection difficulty. Because they are propriety, we did not have access to the back end code to collect data, and even if we did, we would not be legally allowed to make changes for data collection purposes. Since both programs were designed with education instead of research in mind, they do not collect information on the actions students take. We reached out to both companies asking whether it would be possible to get access to backend data about how our students used the platform, but both said it would not be possible in the current infrastructure without major changes.

Therefore, we looked at several more versatile options to see if we could adjust them to our needs. Below we list the best option we found for each of the required traits. The best system for graphics was Fritzing. The best system for allowing real-time changes to the circuit was a backend server running Spice. Lastly, the best system for data collection was Qualtrics, which we had been previously using for debugging activities.

Fritzing is a European-based open-source system that creates beautiful physical representations of circuits [39]. The problem with this system is that it does not include a circuit simulator, so all nodal voltages need to be calculated and recorded on a separate system. Since this system was free and had good graphics, we used the software to create the physical representations for our own website.

Spice is an open-source circuit simulator that is widely used in the education world [40]. There are a variety of user-friendly spin-offs available like the freeware LTspice from Analog Devices [41]. Circuits can be defined in code, which would allow for real-time changes on the circuit with a server running Spice on the backend to simulate the new experimental results. While the open-source nature was appealing to us, the current systems did not support physical representations of circuits like breadboards. Although this was a deal breaker for us, we may use Spice in the future to run a circuit simulator backend for a breadboard representation.

Lastly, Qualtrics is a paid subscription service aimed at surveys and other customer satisfaction metrics [42]. It has a graphical interface for building surveys and requires very little computer science knowledge from the test developer. Qualtrics takes care of browser compatibility issues and all data management, so the test developer can just download a spreadsheet of results at the end. This greatly reduced our work when building and designing a survey, making it our platform of choice. We created both our pilot activities and a variety of other debugging activities in Qualtrics, which we discuss in greater detail in the next section. Unfortunately, the platform does not currently offer ways for students to click and add multiple measurement probes to a picture, so we we had to build our own system for students to place and take measurements on a breadboard.

4.2 Simulator 1.0

In the next several sections we detail the evolution of our debugging simulator from its initial pilot in Fall 2019 to its final version deployed in Spring 2021. We started out with a single debugging activity based on a non-inverting amplifier circuit, so we describe the evolution of this single activity, though we should note that we later added other debugging activities to the simulator.

We chose a non-inverting amplifier for the first debugging activity for a variety of reasons. First, we wanted the debugging activity to come at the end of term when students had had more debugging experience, so we chose the last topic from the course, which is op amps. In addition, op amps are easy to breadboard and show a clean picture for the wiring. We specifically chose a non-inverting amplifier because it matches one of the stages they build in their EKG in their last lab. Therefore, the activity was supposed to be a simplified version of something they had seen in lab.

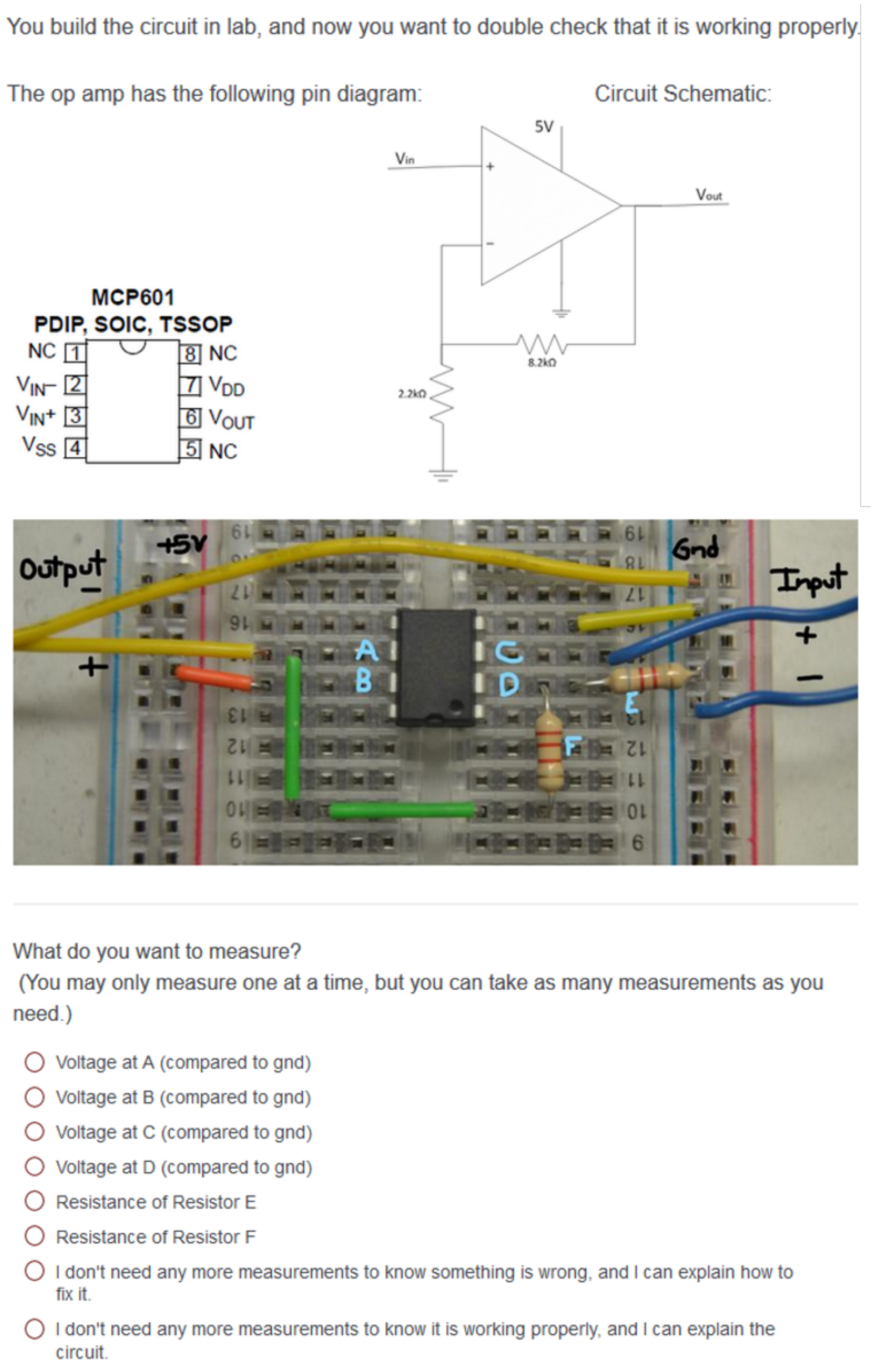
For the specific problem with the circuit, we chose to have the resistors switched, because we wanted a straightforward issue that was not too hard to find. Furthermore, if students measured the empirical gain, they could potentially deduce what was wrong from the specific value of the gain (and some of the stronger students did).

We created the initial debugging activities in Qualtrics because we did not want to invest significant amounts of time into making our own website unless we knew the activities would be fruitful–both in terms of reception by students and usefulness to research. To simulate taking measurements in lab, we took a picture of a real circuit and labeled various locations on the circuit with letters. Students could choose the letter location where they wanted to take a measurement and clicked next to receive the measurement of the nodal voltage at that location (see Figure 4.2).

On the backend, the system was structured using the loop-and-merge feature, which allowed us to repeat a block of questions a fixed number of times for students. Each measurement was given logic to appear only when a student requested that specific measurement. The measurement question could also be given conditional logic to keep appearing until the student said they were done taking measurements. An example of the interface is shown in Figure 4.3.

We learned a number of lessons from the pilot studies on this original simulator, which improved future versions of the simulator. First, we learned that it was important to put the op amp at a 90 degree rotation from the pin out diagram in order to force students to show whether they knew how to interpret the dot representing pin 1 on the op amp. We also learned that many students naturally wanted to use an oscilloscope to measure this circuit, even though it had a constant input voltage. We suspect that students were falsely pattern matching op amps and oscilloscopes, which we will discuss more in Section 4.5.

We also realized that some students were not taking any measurements at all, because they noticed by visible inspection that the two resistors were switched. Since we wanted students to have to take and interpret measurements in order to isolate the cause of the circuit fault, we changed the problem in subsequent terms to have different issues that were invisible in the picture–like a broken op amp or a loose connection. Changing the problem with the circuit to a non-visible problem greatly increased the difficulty of the problem. We will discuss the repercussions of making the problem more difficult in Section 4.7.


You build the circuit in lab, and now you want to double check that it is working properly.

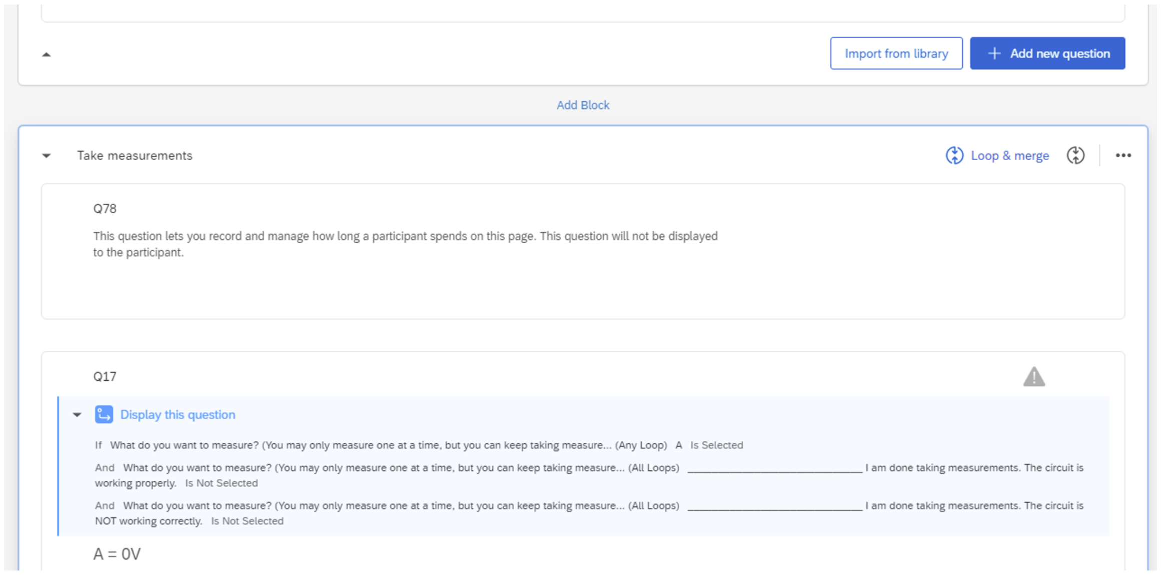
The op amp has the following pin diagram:

Circuit Schematic:

What do you want to measure?
(You may only measure one at a time, but you can take as many measurements as you need.)

- ○ Voltage at A (compared to gnd)
- ○ Voltage at B (compared to gnd)
- ○ Voltage at C (compared to gnd)
- ○ Voltage at D (compared to gnd)
- ○ Resistance of Resistor E
- ○ Resistance of Resistor F
- ○ I don't need any more measurements to know something is wrong, and I can explain how to fix it.
- ○ I don't need any more measurements to know it is working properly, and I can explain the circuit.

Figure 4.2: Very first version of the debugging simulator ever given to students (simulator 1.0). The circuit is a non-inverting op amp with its resistors reversed. Students could measure the voltage at six locations or measure the resistance of the two resistors.



Q78

This question lets you record and manage how long a participant spends on this page. This question will not be displayed to the participant.

Q17

Display this question

If What do you want to measure? (You may only measure one at a time, but you can keep taking measure... (Any Loop) A Is Selected

And What do you want to measure? (You may only measure one at a time, but you can keep taking measure... (All Loops) ______________________ I am done taking measurements. The circuit is working properly. Is Not Selected

And What do you want to measure? (You may only measure one at a time, but you can keep taking measure... (All Loops) ______________________ I am done taking measurements. The circuit is NOT working correctly. Is Not Selected

A = 0V

Figure 4.3: Qualtrics interface showing the loop and merge feature and how to program logic to make measurements appear only after a student asks for them. The settings for the loop and merge feature can be adjusted by clicking on the loop and merge button in the upper right. The display logic for the question is shown in the blue box in the middle bottom.

We decided to develop our own debugging simulator on our own platform for several reasons. While the Qualtrics platform worked in practice for this simple system with six measurement points, it was both clunky and limited. The multiple clicks required for each measurement made the system tedious and non-intuitive. In feedback, multiple students complained that they were frustrated with how many times they had to click "next" just to get a measurement. Another limitation was that students were only able to take measurements in a finite number of places, which made the system unrealistic. In a real-world debugging scenario, a student can choose to place their probes anywhere, not just at one of a half dozen lettered locations, so we had effectively reduced the search space for students. Furthermore, there was no way to test whether students understood where to place the ground probe, which made us concerned that we were not fully measuring students' measurement knowledge because they only had to place the positive probe. We wanted to to be able to measure how students chose to place both measurements probes when given the choice, because we knew from lab that many students forget to place the ground probe or placed it in strange locations.

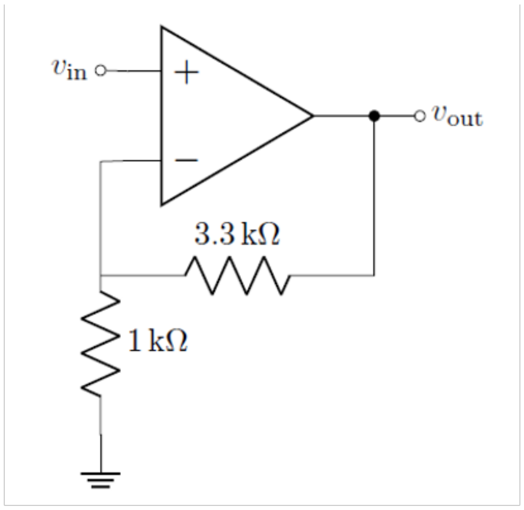
Note that although Qualtrics expanded its features to include a multi-click heat map question format, these new features still did not suffice for our needs, and we concluded that our own debugging platform is a better tool for simulating debugging. The heat map question allows for a user to click on an image and for the computer to give a response based on where they clicked, which would remove the need for letters and make the graphical interface more intuitive. Unfortunately, the multi-click heat map does not differentiate between the two clicks, so we cannot assign one to the red probe and one to the black probe. Furthermore, Qualtrics also does not allow for logical operations like addition and subtraction, which means we would have to create an entry for every possible combination of locations where the two probes could go. Finally, students would still have to click “next” to get their measurements after placing the dots representing the probes with the picture format, so we are still bounded by the superfluous clicking problem.

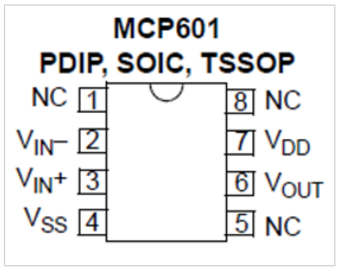
4.3 Simulator 2.0

In order to make the user interface for placing measurement probes more intuitive, we developed our own web-based debugging simulator. Our new simulator allowed students to graphically place probes on a breadboard and take measurements while we recorded the order and timing of measurements. Measurements would appear on the screen of the DMM once both the red and the black probes were placed at a given location (see Figure 4.4).

Stanford student Atindra Jha developed both the front and back end of the website for an undergraduate research project in summer 2020. The website was stored on Stanford’s class website server space. The basic pipeline is as follows. Images of the physical circuit were created in Fritzing and placed on an HTML Canvas. Javascript was then used to draw wires when students place probes on the circuit and make a number appear on the DMM (see Figure 4.4). Since students do not learn to use an oscilloscope until the final lab in class, we decided to simplify out simulator and only include a DMM in the first version. If need be, it would be relatively straight forward to replace the DMM with an oscilloscope in the future. Since we have no circuit simulator running backend, voltages are experimentally taken from a physical circuit in lab. The nodal voltages for each breadboard row are then recorded in a JSON file, which is then accessed to calculate the value that should be displayed on the DMM. This nodal voltage system allows us to model unique debugging situations like broken op amps that one could not normally simulate with a circuit simulator.

Every time a student clicked to place a probe, the color probe, time, and location data was stored backend in a mySQL database. This allowed us to completely reconstruct each students’ measurement attempts. We used mouse click events to record the location of the click, which were then translated into breadboard hole pixel locations referenced in a JSON file. To keep students’ data separate while students simultaneously used the debugging simulator, we identified each student by their student ID, which they had to use to authenticate at the beginning of the activity. This ID was stored, so that submissions on subsequent pages would still be linked to that student. This also allowed us to save students’ submissions across multiple browser sessions and store it in the same entry in the mySQL database.





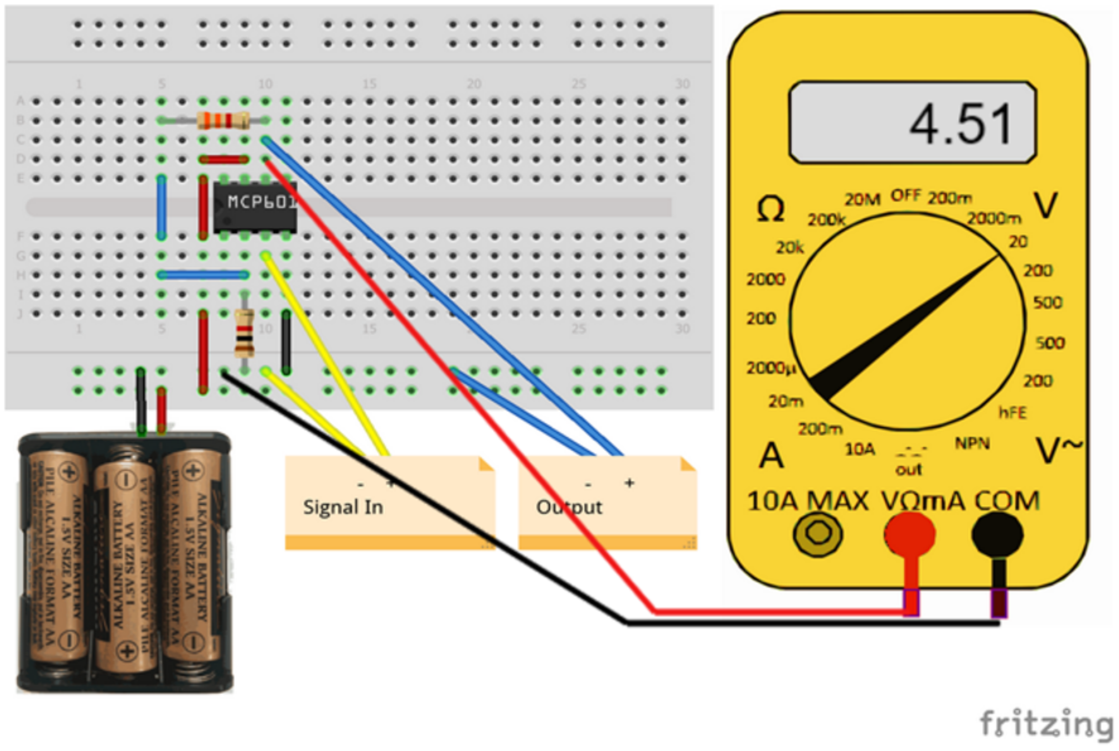


Figure 4.4: **Simulator 2.0**. Students click to place the red and black probes on the breadboard. When both probes are placed, the voltage is displayed on the screen of the DMM. This particular debugging circuit is a noninverting amplifier with a loose wire in the feedback.

The debugging simulator was designed for assignments to be easily interchangeable. To put a new circuit in the debugging simulator only two files need to be changed–the JSON file with the nodal voltages for each row of the breadboard and the Fritzing picture of the breadboard. The only limitation on the Fritzing pictures is that the same breadboard type must be used each time, and it must be placed in the upper left corner of the setup. This causes the pixel values of the breadboard holes to be the same each time.

While we originally intended to include all multiple choice and open-ended questions as part of our own webpage, travel and visa restrictions on our international undergraduate researcher during COVID prevented us from being able to upgrade our system from the original pilot system. As a result of this and the sizeable time needed to maintain the backend data collection long term, we pushed many of the response questions onto Qualtrics, leaving only the measurement tool on our webpage. The Qualtrics page was then embedded in our own webpage below the tool. This way we were able to use our specially designed circuit measurement system, while still having the benefits of Qualtric's data management services. Although we appeared to have some database issues on our webpage resulting in a failure to register some students' measurement data, we were still able to capture full data for at least 80% of students.

4.4 Simulator 2.1

Now that we had made the measurement system more realistic, we needed to solve our next problem, which was how to gather more meaningful data for students who were off track. For circuits without the fault visible in the wiring, the majority of students were not successfully finding the problem with the circuit. Therefore, we wanted to get these struggling students to the point that they solved the problem, so we could see what measurements and other realizations were key to solving the problem. Therefore, we decided to make our simulator iterative, so that students had to keep guessing until they successfully found the circuit fault. Not only did this approach more closely resemble authentic debugging, where one is not done until the problem is found, but it also provided scaffolding for students who were struggling.

In order to implement this iterative version of our simulator, we had to create multiple choice answers, so that responses could be immediately parsed by the computer (answers were previously free-response essays). In this way, we could make students keep trying until they found the correct fix for the circuit. The multiple response choices were created from the open-ended answers given by students in previous versions of the simulator.

Not only did multiple choice allow the computer to give immediate feedback, but it also fixed a recurring problem we had in previous versions of the simulator where students would not fully answer the open-ended questions. On various pilot studies, a good number of students explained how they knew the circuit was not working, but they did not offer any ideas of how to fix it. Therefore, we modified the prompt repeatedly to emphasize that we wanted not only what measurements told them something was wrong with the circuit, but also how to fix the circuit. Despite these changes, in Fall 2020, 17% of students who said the circuit was not working still did not provide any information about how to fix the circuit. We believe a good number of these students simply had no idea what was wrong with the circuit, so they left that part of the question blank. This provided one more reason to move to a multiple choice system that would force students to make a decision about what was wrong with the circuit.

Since we did not want students simply guessing about the correct fix for the circuit, we created a long list of choices of what could be wrong with the circuit. Since long lists can cause cognitive overload for students, we made a two-step branching series of questions, so that no question had more than six choices. The first question asked them which part of the circuit was the problem, for example,the power supply, the op amp, the resistors, the breadboard, a miswiring, or loose connections. Each category led to a second question with several possible specific causes related to that category. Once students selected the correct fix to the circuit, they were sent to the end of the survey and told, "Congratulations! You debugged the circuit!". If students had not found the problem with the circuit after 15 tries, they were thanked for their hard work and also allowed to end the assignment. Full details of the multiple choice answers can be found in Appendix B.9.

For less reasoning-intensive answers, we also tested out the text-parsing feature of Qualtrics to see if that would suffice for computer grading instead of resorting to multiple choice. We tested this on a straightforward question at the beginning of the debugging activity where students were asked what the purpose of the circuit was. We looked for variants of the words "non-inverting amplifier" in the text answer. If students gave an incorrect answer or a non-standard answer format, then we followed up with a multiple choice question which listed seven standard op amp circuits they learned in class. Students were given 7 chances to get this question correct (one for each possible answer) before they were automatically moved to the next section. The majority of students got the answer right on the first try, and only one student in winter term and two students in spring term needed all seven attempts.

In the first term of testing the text-parsing feature in Qualtrics, 29% of students had open-ended expectation responses that were successfully parsed by the computer and allowed to skip to the next section. When reviewed by hand, none of these students were incorrectly allowed to skip to the next section, which is promising. Another 19% of students should have been allowed to skip the multiple choice expectation follow-up question but the computer could not parse their answer. We believe as we get more sample data, we can keep improving this text parser in Qualtrics to accurately catch 90% of cases. Despite this success, we do not believe the Qualtrics text-parsing system is sophisticated enough to handle more complicated reasoning for what is wrong with the circuit.

4.5 Testing Domains of Knowledge

Now that we had a working simulator that authentically modeled interacting with a physical circuit and the iterative process of debugging until the issue was solved, we had to create a set of questions to accompany the simulator. We wanted questions that would probe the various cognitive steps and types of knowledge needed to successfully troubleshoot, which we discussed in our survey of the field in Chapter 2. More specifically, we chose four types of knowledge we wanted to test when we defined our simulator goals in Section 3.3: theoretical knowledge, physical knowledge, measurement knowledge, and debugging heuristics knowledge. Therefore, we based our questions on these various knowledge domains.

During our pilot studies, we tried a variety of different knowledge questions in a variety of orders. Our goal was to figure out which domains students struggled with the most and provide scaffolding and extra questions as necessary at those points. For testing theoretical knowledge, we used questions about the purpose of the non-inverting amplifier circuit and its gain. These are fairly standard homework and test questions, so we were more interested in whether explicitly asking students to calculate gain ahead of time affected their debugging process.

To test how our knowledge domain questions might artificially influence performance, we looked at differences in performance when students were asked theoretical knowledge questions before or after they debugged the circuit. This way we could see if explicitly asking students beforehand about theory improved student performance. We tested this in version 1.0 of the simulator on the non-inverting amplifier circuit. At the beginning of the assignment, we gave students an open-ended question asking what the circuit is supposed to do. Then after they debugged the circuit, we followed up with a question where they explicitly had to calculate gain. From this, we learned that at least 55% of students each term did not mention the gain at the beginning, which indicates they did not calculate it beforehand or they did not feel it was relevant information for their answer. When prompted to calculate gain at the end, over a quarter of the students each term incorrectly calculated the gain. This indicates to us that roughly 20% of students knew how to calculate the gain, but potentially did not calculate it beforehand.

Therefore, if we ask the students to calculate the gain beforehand, we might expect those 20% to perform better. When the gain question was moved to the beginning of the debugging activity, the same percentage of students were incorrectly calculating gain. Furthermore, the percentage of students correctly diagnosing the circuit did not statistically significantly increase. Thus, our explicitly asking about the gain does not appeared to have changed students' chances of solving the problem.

For physical knowledge, we put pictures of circuit components used in lab on students' final exams in Summer and Fall 2019. The most commonly mislabeled component was the voltage converter, which was a component students used in their first lab to step up the 3.7V from their lithium polymer battery to the 5V they needed to power a USB port (see item B in Figure 4.5). Over a fifth

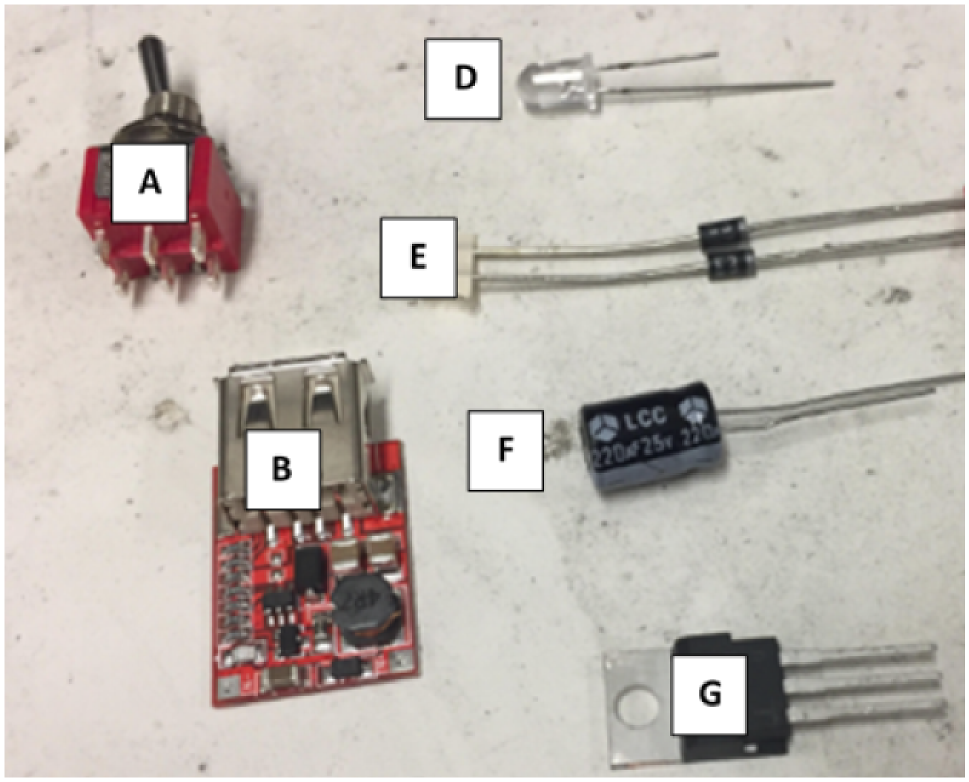


Figure 4.5: Final exam question about physical circuit components. Students were asked to label each component. Components from A to G are DPDT switch, voltage converter, LED, diode, capacitor, and transistor.

of the students mislabeled it each quarter, with the most common wrong answers being "Arduino" and "USB port". Over 10% of students mislabeled the diodes each term with the most common mislabels being "inductor" and "capacitor." Finally, 10% of the students in fall mislabeled the capacitor, with the most common incorrect answer being "inductor". It was clear from these tests that students were not ingraining basic physical knowledge we assumed they were in lab. This was why it was of utmost importance to us that our circuit simulator contain physical images of the circuit.

To test students' measurement knowledge, we asked students in Winter 2020 where to place the red and black probes to measure the voltage at a starred location on the theoretical diagram. Their answers were constrained by labeling different locations on the physical circuit diagram with letters and asking them at what letter location the red probe should be and at what location the black probe should be. 29% of students incorrectly placed the red and/or black probe. This is a big reason why we moved to our own debugging platform, so we could force students to take their own measurements and practice placing both probes themselves on the circuit.

Another question we asked related to measurement knowledge was whether a DMM or oscilloscope would be the best tool for measuring the circuit, considering the circuit had a constant input voltage. 26% of students said they would use an oscilloscope, most likely incorrectly pattern matching op amps and oscilloscopes because they only used the oscilloscope in their lab with op amps.

Finally, for a measure of debugging heuristics, we looked at the order in which students chose to take measurements. We wanted to know if students were just taking measurements randomly or if they had a strategy. For example, with a non-inverting amplifier a good starting point might be measuring the input and the output to see if the gain is as expected. Even in the very first term of our simulator 1.0 where we only have 6 distinct places that students could measure, there are 15 possible combinations of first two measurements. We found that 73% of students started with one of just six of those combinations. We saw these same six starting combinations in the same frequencies even when we expanded to 12 locations students could measure, which meant there were 51 more possible starting combinations. Furthermore, we saw the students who started with one of the six common combinations were more likely to correctly diagnose what was wrong with the circuit than students who started off with a more abnormal set of measurements. Because of the consistency of these six combinations, we called them starting strategies, and we discuss them in more detail in the results chapter (see section 6.2).

Since one of the starting combinations was taking measurements in alphabetical order, we wanted to eliminate that as an option, since it is not a choice of strategy in a real debugging lab. This was another good reason to upgrade to simulator 2.0 where students could measure at any place they wanted on the breadboard. Even after eliminating that option and opening up a seemingly infinite choice of probe placement locations, we still found the majority of students starting with one of the five other starting strategies.

From these results, it is clear that students are lacking knowledge in a variety of domains needed for successfully debugging circuits. Based on the results above where a large number of students incorrectly answered physical and measurement knowledge questions, we hope we have made clear the importance of testing all types of knowledge for beginning students.

We also see these results as justifying several of our previous design decisions. Since so many students struggled with mapping the circuit picture and theoretical diagram, our decision to keep a physical diagram of the circuit, instead of resorting to a pre-existing platform like SPICE (which would not have allowed for a physical diagram) was good. Secondly, the number of students who incorrectly placed their red and black probes demonstrates the importance of our choice to make our own platform where students had to place their own probes instead of simply asking for letter-labeled nodal voltages in Qualtrics.

4.6 Checkpoints

Once we knew that a number of students struggled with each of the four different types of domain knowledge, we wanted to implement scaffolding for each to help keep struggling students on track. Therefore, when we moved to an iterative system in simulator 2.1, we not only made the final answer about the cause of the circuit fault iterative, but we also made previous knowledge domain questions iterative. We called these previous knowledge domain questions **checkpoints**, because students could not move on to the next part until they had correctly answered the previous part. This served both as a way for us to measure students' performance on the different knowledge domains and also as a way for us to give further guidance to struggling students.

Students were simply told their answer was wrong each time and asked to try again. After a fixed number of failed attempts, students were then told the correct answer and moved on to next checkpoint. The number of attempts allowed at each checkpoint ranged from 4-15 and depended on the number of multiple choice answers for the question and/or the difficulty of the checkpoint. The following section details the various checkpoints that were placed in the 2.1 simulator for the non-inverting amplifier circuits.

First, students were asked two questions regarding their **theoretical knowledge** about the circuit. The first question asked what the purpose of the circuit was, which was the open-ended question that utilized text parsing described previously in Section 4.4. If a student's answer did not satisfy the computer parser, they would be given the same question in multiple-choice format. They were then given 7 attempts to answer the 7 possible answer multiple choice question.

The second theory question asked for the gain of the circuit. The computer would only accept numeric answers up to one decimal precision (and would give students feedback if they tried to type in something else). Students were given five attempts to correctly calculate gain, with the only feedback each time being that their answer was incorrect. After the fifth attempt, students were shown a explanation where the gain was calculated, and they were then moved on to the next section. While the majority of students got the gain right on the first try, 12% of students still had not calculated the gain correctly after 5 tries. This gain question, along with the purpose of the circuit question above, comprised the theoretical knowledge section of the assignment.

The next checkpoint regarded **measurement knowledge**. Students were asked whether a DMM or oscilloscope was a more appropriate tool for making measurements, given that a constant input voltage was being applied to the circuit. Students who answered "oscilloscope" were given feedback that oscilloscopes are designed for use with signals that vary in time.

Next, students faced a checkpoint that incorporated both measurement and **physical knowledge**. Students were asked to measure the input voltage and report it back to us. The computer would only accept numeric answers. This tested not only whether students knew how to use the measurement tool, but also if they could translate the theoretical and pinout diagrams to the physical circuit. Students were given five chances to answer this question. They could take as many measurements as they wanted on the circuit before answering this question. Therefore, this is not a measure of how many tries it took them to correctly take a measurement on the circuit, but rather a measurement of how many times they thought they took the correct measurement when they had not. During the first term of testing this new question, 10 students (11%) still had not managed to properly measure the input voltage after 5 tries. These 10 students then received feedback on where to place the red and black probes to take the measurement and that the result they should have found is 0.5V.

At this point, the students were invited to take as many measurements on the circuit as they needed to determine whether the circuit was working properly or not. We looked at their measurement patterns along with a previous non-checkpoint question about their plan for debugging the circuit to determine a student's heuristic knowledge about debugging. They were told to click next when they were done investigating.

The next set of questions focused around other cognitive tasks of debugging not initially included in the four knowledge domains. These other categories were drawn from the decision framework for problem solving discussed in Section 2.5 and chosen because we noticed students struggling with them during the pilot studies. The first difficulty was with **interpreting the data** and realizing there was a discrepancy between the actual measurements (physical world) and the expected measurements (theory). The second difficulty involved using the data to generate possible explanations for this discrepancy, which we call **isolating the cause**.

We combined two questions into a single checkpoint for interpretation of data. First, students had to decide whether the circuit was working properly or not. Students who said the circuit was working (around 20% each term), were asked the follow-up question of what measurements indicated the circuit was working. Students could select as many of the statements as they wanted. The most common answers were "the circuit amplifies" and the "I checked the wiring". It is interesting to note that both of these are true statements, but not the complete story, which is a result we will discuss further in Section . Regardless of the explanation students gave for the circuit working, students were then told that one or more of their conclusions were incorrect and sent back to the question of whether the circuit was working or not. Students were given four chances to say whether the circuit was working properly or not. By the fourth try, all students had realized the circuit was not working.

After a student realized the circuit was not working, they were asked what measurements indicated the circuit was not working. This is similar to the question asked of students who said the circuit was working, but the multiple choice answers are very different for this situation. Five of the seven answers were true statements related to different aspects a student might have noticed, like the input pins of the op amp being unequal or the gain being off. The other two answers were common wrong answers we had previously seen in open-ended responses–that the circuit was miswired or that the power supply was the incorrect voltage. Again, students could select as many answers as they wanted. Students were allowed to move on to the next question only if they had not chosen one of the untrue answers.

Finally, students were asked to **isolate the cause** of the circuit fault. This was the main question showcased when we described simulator version 2.1 because it is the ultimate goal of the debugging process. If a student got this answer incorrect, they were told to take more measurements to try and figure out what is wrong. Then, when they are ready, they could try answering the question again about how to fix the circuit. After 15 tries, students were thanked for their effort and allowed to leave the assignment with full credit for their attempt.

4.7 Isolating the Cause

The checkpoints various pilot versions of the simulator helped us uncover the major difficulties students were having interpreting measurements in order to isolate what was wrong with the circuit. This instigated us to add another checkpoint in an attempt the bridge the gap for students between knowing something was wrong with the circuit and interpreting that information to find a cause for the circuit fault.

The first indicator of how hard isolating the cause is came when we switched the problem with the circuit from a visible miswiring (the resistors reversed) to a non-visible problem (loose wire in feedback loop) in the pilot studies. The percentage of students correctly diagnosing the circuit dropped drastically from 60% to 20% when we made this switch. This indicated that students were not successfully interpreting measurements to figure out what was wrong with the circuit.

The next indicator came when we switched to simulator 2.1, which allowed students to keep guessing until they found the correct fix for the circuit. We were surprised to discover that half the students took 7 or more guesses to find the correct fix for the circuit and almost a quarter of them never figured out what was wrong with the circuit, but rather timed out after 15 attempts. From this, we knew that a significant portion of students were completely lost about how to isolate the circuit fault. Because so many students needed so many attempts, we decided we needed to give students more cognitive guidance on how to interpret measurements and generate possible causes. This decision also aligned with some student feedback that they knew something was wrong with the circuit, but they had no idea how to pinpoint where the problem was.

Therefore, in spring 2021, we included an extra checkpoint in the assignment between the question about what measurements indicated something was wrong and the final answer of how to fix the circuit. Students were given an identical picture of the circuit marked with boxes circling various parts of the circuit, like the feedback loop, or the power rails. Students were asked to highlight each box green for parts of the circuit that were working and highlight each box red for parts of the circuit where measurements indicated something was wrong (see Figure B.27). This checkpoint where students had to mark different regions of the circuit as working or not was our attempt to help students isolate the cause. Students who did not want extra guidance were given a chance to skip that portion, if they already thought they knew what was wrong with the circuit. The intervention was a success, with many more students correctly diagnosing the fix for the circuit on the first try, which we will discuss further in Section 6.3.3.

4.8 Limitations of Final Simulator

As we improved our simulator and checkpoints, we also introduced new problems. Multiple choice questions have their limitations because it is hard to convey complex reasoning in a single line answer, especially when the audience is not fluent in the jargon for the field. Furthermore, our breaking of the possible causes for the circuit fault into six categories to reduce the number of multiple choice answers listed at any one time was confusing for some students. Based on student and TA feedback, we learned that some students have an idea of what was wrong with the circuit, but were not sure which of the six categories their idea would fit in. Therefore, in the future, we will experiment with better ways to present the final question about how to fix the circuit. Perhaps we will limit it to ten choices that are the most common answers or we will try to use text parsing, as we did for the first question about the purpose of the circuit.

Another problem that arose was that a small, but noticeable, percentage of the class (around 5-10%) appear to have moved to a mindless guess-and-check strategy when we introduced the checkpoints in simulator 2.1. Instead of taking more measurements when they incorrectly diagnosed the circuit fault as prompted, these students stopped taking measurements completely and just started guessing in rapid succession about what might be wrong with the circuit. Cognitive effort is energy consuming, so it is not surprising that these students chose to take the path of least cognitive resistance and just start blindly guessing.

Further contributing to this problem was our reward structure. Students were graded for participation only because we were trying to get honest answers from students. We wanted to encourage students to give their initial thoughts even if they were wrong instead of asking their friends for the answers. Unfortunately, a downside of this appears be that many students just click through without much thought in an attempt to get done with the least effort possible. Several students in Spring 2020 even commented on this practice in the feedback. They said they were not sure how much they were actually learning from the assignment because they were just guessing and checking to get done with the assignment instead of really understanding what was going on.

A comparison of the data from before and after we introduced the iterative questions in simulator 2.1 shows that having the chance to keep guessing did appear to have an effect on performance. In fall 2020, students received an open-ended version of an op amp debugging problem, and in winter 2021 we gave students the same problem but with multiple choice responses. In the open-ended version, 33% of students who said the circuit was not working correctly identified what was wrong with the circuit. This percentage went down to 13% of students when they were given multiple choice and could keep guessing until they got the correct answer. The p-value via the Fischer Exact test is 0.0246, indicating this difference in performance is significant and not likely due to sampling variance from term to term.

We believe it is reasonable to attribute this difference to the open-ended versus multiple choice solution response types. If students know there is little cost to their guess, they may start guessing prematurely, whereas the students in the open-ended answer will keep exploring until they are fairly confident and then write down their final answer. Therefore, we see that while multiple choice responses allow for easier analysis and allow us to create looping structures to keep students participating until they get to the correct answer, there is a trade-off. Namely, there is less at stake for each guess, so students can invoke less mental effort each step of the way. We were comfortable with this compromise because we would rather know students' gut responses rather than a whitewashed version with just their final thoughts. In the future, it may be worth studying the effects of this decision to see if the guess-and-check mentality of the multiple choice is changing the way students approach debugging long term.

4.9 Debugging Assignments

Once we had our simulator up and running, we created a whole debugging curriculum for winter and spring terms 2021. Students were given one debugging assignment per homework, starting on the second homework. On some weeks the students completed text-based choose-your-own-adventure debugging activities in Qualtrics and other weeks they completed assignments where they could take their own measurements on our own debugging simulator. Some assignments in Qualtrics allowed students to take voltage readings and others let them take more generic actions, such as looking at the settings of the DMM or inspecting solder joints on the back of a PCB.

We tried to make the assignments match what they were doing week to week in lab. In the next table, we give a summary of the order of the activities. The activities marked with a double asterisk used our own website with the measurement tool and the other choose-your-own-adventure assignments were created in Qualtrics. While most of the Qualtrics activities were text-based adventures, several of them were based on the version 1.0 of our simulator and allowed students to take sequential voltage measurements (denoted with a single asterisk in the table).

	Winter 2021	**Spring 2021**
HW 2	Basic measurements with DMM (Section 4.9.1)	Basic measurements with DMM (Section 4.9.1)
HW 3	Miswired useless box (Section 4.9.2)	**LED circuit with bad connection (Section 4.9.6)
HW 4	*Miswired inverter in computerized useless box (Section 4.9.3)	*Miswired inverter (Section 4.9.3)
HW 5	Row not working on LED display (Section 4.9.4)	Row not working on LED display (Section 4.9.4)
HW 6	Incorrect resistor in LED circuit (Section 4.9.5)	*Non-inverting amplifier with broken op amp (Section 4.9.7)
HW 7	**Non-inverting amplifier with loose wire in feedback loop (Sections 4.2 to 4.7)	**Non-inverting amplifier with loose wire in feedback loop (Sections 4.2 to 4.7)

**Our own measurement platform *Simulator 1.0

The basic flow of the choose-your-own-adventure debugging activities was as follows. Each activity started with an initial description of the problem. Then students entered a looping structure. Each round, students had to answer what they suspected was wrong with the circuit and then choose a test to perform on the circuit. We used their updating hypotheses to get a gauge for what they were thinking and how they were interpreting new information they were gathering. When students felt they had gathered enough information to fix the circuit, they could check a box to exit the loop and answer a final question about what was wrong with the circuit. If students got this question incorrect, they were sent back to continue performing tests on the circuit again.

The looping structure of each activity was generally broken down into the following checkpoints, where students had to get each part correct before being able to move to the next part. Below is a list of general checkpoints we measured:

1. **Theoretical knowledge** (function of circuit, expected gain, etc.)
2. Plan
3. **Measurement knowledge** (how to place DMM probes, etc.)
4. Enough data
5. **Diagnosis** (is circuit working or not?)
6. **Problem indicators**
7. **Isolate Cause** (correct fix for circuit)

Items in bold represent steps that were checkpoints and students had to repeat until they got the correct answer, and the other steps simply were recorded and analyzed later, but no correct answer was required. For example, the plan question was not a checkpoint because there is no right or wrong answer. Sometimes we made the plan question an open-ended response, and other times we made it a list of possible measurements that students could order. *Enough data* simply looked at whether students collected sufficient data. In other words, did the student measure any of the parts of the circuit where the anomalous behavior was occurring? While one might think it obvious that students would take basic measurements, not all students did this! Finally, some checkpoints had multiple knowledge components; for example, how to place DMM probes might involve both measurement knowledge and physical knowledge. We will look at general trends in performance at each of the checkpoints in Chapter 6.

The subsections below give a short description for each of the different debugging activities. A full description of each can be found in Appendix B.

4.9.1 Basic measurements with DMM

The first debugging activity we had students complete in our debugging curriculum was based on a common mistake students make when first learning how to use a DMM. In the very first lab session, students were asked to measure the short circuit current and then the leakage current of their solar cell sequentially. Many students forgot to switch back the red probe from the 10A port to the standard port on DMM when they switched from measuring high current to lower current. In the debugging activity, students were told that they are trying to measure the leakage current in the same way they had in the lab earlier that week, but no matter what they did, they always got "0" on their DMM. In each iteration of the debugging simulator, the students were required to say what they think might be wrong with the circuit, and then choose a test to perform on the circuit to better diagnose the problem. Students could take a variety of actions, like looking at a picture of the experimental setup, looking at the settings and probes on the DMM, replacing batteries, or checking for loose connections. When students felt they knew what was wrong with the circuit, they could click a button to jump to answering a question about what was wrong with the circuit. They were asked whether the problem was the battery pack, the solar cell, the wiring, or the DMM. A second follow up question was asked for the DMM with five possible choices of what could be wrong–ranging from a broken fuse to the hold button (freeze the display) being on. If students chose the wrong answer on either question, they were sent back to keep exploring and testing the circuit. Details of the full activity can be found in the Appendix B.1.

4.9.2 Miswired Useless Box

In the second lab, students were asked to build a useless box using a DPDT toggle switch and SPDT limit switch for logic. As mentioned in the introduction, the only purpose of the box is to send out an arm when the toggle switch is flipped on to turn it back off again [30]. Students were told in the initial problem statement that the box was working when they first built it, but when they came back several days later, nothing happened when they flipped the switch. They were then asked a looping question where for each iteration, they were asked what they suspected was wrong with the circuit and what they would do to test/check the circuit.

Nothing would happen until the student replaced the batteries. When they replaced the batteries, they were told the useless box worked temporarily, but it stopped working several days later. They were also told they could assume they put fresh batteries in to test the circuit each time from then on.

Our aim was for students to discover that the most informative test is when one tests what happens when each of the switches is pushed. In that case, if a student chose to check the wiring, they would be shown a picture of the wires (which were a giant mess), and told that the TA suggests they try to isolate the problem instead. When students finally tested what happened when they pushed each of the switches, they should have noticed that the limit switch was turning the motor off at the wrong time, indicating that it was in the wrong path in the circuit. To test whether students had the theoretical knowledge to correctly understand the results of this test, we had students fill out a finite state diagram of the switch states and the resulting motor direction before getting the results of the test.

When students felt they had collected enough information, they checked a box which led them to a question to confirm their final diagnosis. Students were given a list of seven possible causes, and if they got this question incorrect, they were sent back to keep investigating, as in the previous assignment. Details of the full activity can be found in the Appendix B.2.

4.9.3 Miswired inverters

The third debugging assignment students encountered both terms related to the computerized useless box, which used an Arduino and two inverters to control the motor. We tested out two different versions of this debugging activity. For the first term we had a choose-your-adventure where the drain and source were switched on one of the pMOS. The key observation students were supposed to uncover was that some of the transistors were hot to the touch (caused by the pMOS being on all the time). If students did not discover that from the myriad of actions they could choose, students could also take measurements of the circuit or check wiring in a text-based fashion. To incentivize measurement actions to isolate the cause over the mistake-prone activity of checking wiring, we allowed students to take up to four measurements in one go, but they had to check each wiring connection one-by-one. For example, they could ask for the voltage at the gate, drain, and source of the second pMOS all at once. Or, they could check the wire connecting the gate of the first nMOS to the Arduino first and then next check the wire from the drain of the first pMOS to the first nMOS and so forth for each wire.

We believe this debugging problem was too difficult for students, so we did not pursue it for a second term. Instead, we created a debugging problem involving a single inverter where two nMOS were accidentally used for the circuit instead of one nMOS and one pMOS. Not realizing there are different types of transistors differentiated by the label on the outside of their packaging is a fairly common mistake we see during the first day of building transistor circuits in class. Therefore, we felt this debugging situation was authentic and useful preparation for lab because it made students think about the unexpected or odd things that occur when transistors are miswired.

While the nature of the problem with the circuit was designed to help students learn and prepare for lab, we additionally ran an A-B test to see how physical representations of the circuit affect student performance. Half the group was given a schematic and allowed to take measurements at various points on the schematic. The other half were given both the schematic and a physical picture of the breadboard. They were allowed to take measurements at various points on the breadboard, which meant that they then had to mentally map the voltage readings back to locations on the schematic. We believed that the students who had to map the physical breadboard to the schematic would perform worse than their counterparts that obtained the measurements directly marked on the schematic, because they had one more cognitive step in the debugging process to execute. It turns out there was no difference in performance isolating the cause, though students who received the measurements directly on the schematic took one less measurement on average (9 measurements versus 10 for the group who took measurements on the physical diagram). With a standard error of roughly 0.3 for each group, this result is significant, and is likely a reflection on the relative ease of taking measurements directly on the schematic versus having to translate locations to the physical circuit.

Like other activities with checkpoints, the inverter activity required students to correctly guess what the circuit was supposed to do before allowing them to move on to making measurements on the circuit. After they were done taking measurements, students were asked what measurements indicated something was wrong, and then asked how to fix the circuit. Again, if students incorrectly diagnosed the problem with the circuit, they were sent back to take measurements again. Because students struggled so much to find the correct fix on various activities in previous terms, we created specific feedback for each wrong answer chosen. In each case, we presented them with a piece of information that refuted their conclusion and then told them to continue investigating. Details of the full activity for both inverter problems can be found in the Appendices B.3 and B.4.

4.9.4 Row not working on LED display

During the fourth debugging assignment each term, students were asked to debug an 8x8 LED display on a PCB board, like the one they had built in lab that week. Instead of focusing on students taking measurements, we focused on what subparts of the board students chose to inspect. Students were told that the 7th row of the display was not working. We presumed most students would choose to inspect the pMOS first, since that was the most straightforward explanation for why a single row was not working and we often observe students put in a transistor backwards in lab. After students realized the transistor was not the problem, we were curious about what alternative explanations they would look at next–such as a typo in the Arduino code or a missing solder joint on the back of the PCB. We were also curious about how many students would look at highly unlikely possibilities based on the symptoms, like inspecting the resistors or inspecting LEDs. We also gave students an option to inspect or replace each component to get an idea of how many students choose the more difficult route of unsoldering and replacing components before inspecting them. The full activity can be found in Appendix B.5. We will discuss some results from this debugging activity in Section 5.6.2.

4.9.5 Incorrect resistor in LED circuit

While students in both winter and spring 2021 terms encountered an LED debugging activity, they were different problems in different weeks. In winter 2021, students debugged the LED circuit in the second-to-last homework. The circuit had 3 LEDs, each in parallel with a resistor, but one of the resistors was replaced with a $1\,\mathrm{M}\Omega$ resistor, so that the LED did not light up. The main actions students could take were to measure the voltage across various components (like the battery pack or a specific LED), measure the resistance of the resistors, check the connections between components using a DMM, or inspect the circuit visually, where they were shown a picture of the circuit on the breadboard. The most frequent feedback we got on this assignment was that it was hard to tell the color bands of the resistors in the picture, which was true due to the contrast difficulties of taking a photo of brightly lit LEDs. Interestingly, these students could also have gathered the same information by asking for the resistance of the resistor, but their individual choose-your-own adventure path must not have led them to see that test option. The full activity can be found in Appendix B.6.

4.9.6 LED circuit with bad connection

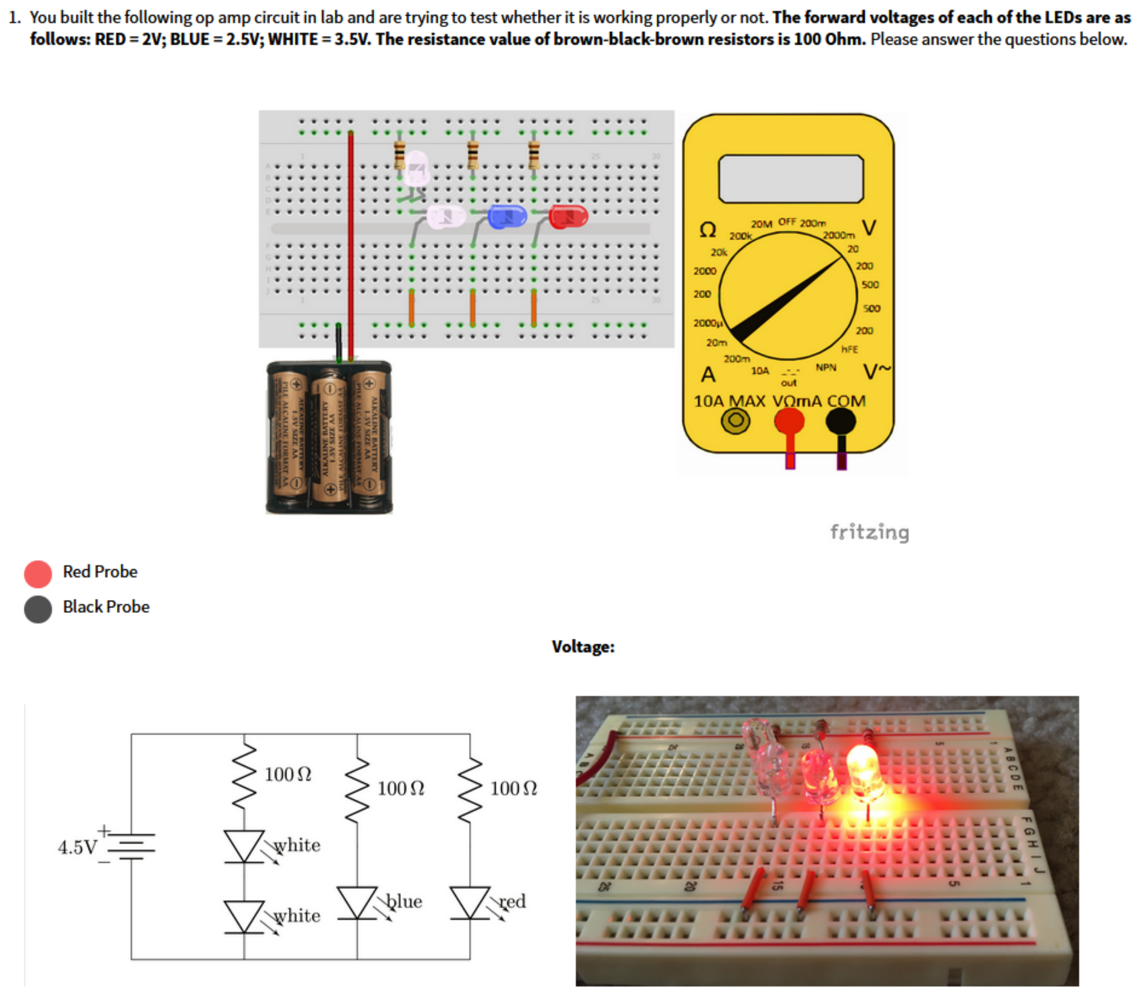
In spring 2021, students were given the LED debugging activity in week 3, and it was done on our own debugging simulator that we described in the last section. We wanted to run our debugging simulator at both the beginning and the end of the term to see how students developed over time on the tool. Since the assignment would be in the third week of term, we needed a topic that was covered at the beginning of term, but was still interesting to build on a breadboard, hence our choice of LEDs. We also wanted to have a circuit without a visible mistake, to parallel the debugging assignment at the end of the term. Therefore, the problem with the circuit was chosen to be a resistor not firmly pushed into the breadboard. A picture of the circuit simulator for this LED debugging activity is shown in Figure 4.6.

Like the op amp activity on our debugging simulator that was previously described, students could take voltage readings anywhere on the circuit by placing the red and the black probes of the DMM. The checkpoint questions were separately created in Qualtrics and embedded in the website, as was done in the previous activity on our debugging simulator.

Since our measurement tool only allowed for students to take voltage readings, we had to allow text-based options in the embedded Qualtrics page where students could take other types of measurements. Specifically, we allowed them to check the functionality of the LED with the DMM in diode setting and also measure the resistance of the resistors. While it would be easy to add a way to change the knob on the DMM to move to either diode or resistance settings, both of these tests required pulling components out of the circuit for testing, which would have made the graphics much more complicated. Therefore, we left measuring resistance and LED functionality with the DMM as text-based options for the initial trial of the simulator. Some students lamented in the feedback that they wished they could take resistance readings with the DMM, and a few others even tried taking resistance readings and said they “couldn’t figure out how to do it.” Incorporating these suggestions is an important consideration for future implementations of the simulator.

The structure of the checkpoints was the same as the op amp problem at the end of term. First, students had to show correct theoretical knowledge. In this case, that meant correctly identifying which LEDs should be on. We chose a circuit with four LEDs, where only two of the LEDs should be on, to make this theoretical knowledge question non-trivial. Next, students were asked to make a plan for what they wanted to test in what order. We left this task open-ended, since we did not have any previous data about how students approached this problem. After that, students were asked to measure the voltage of the battery pack to test their measurement skills, and, again, they could not move on until their answer was correct. Then, when students were done taking measurements, they were asked to decide whether the circuit was working properly or not. Since students previously had to show in the theory knowledge question that only the red and blue LEDs should be on, it was fairly straightforward to see a problem, namely, that the blue LED was not lighting up. Next, students answered an open-ended question about what measurements indicated something was wrong with the circuit. Note that these answers will be used in future terms to create a closed-form version of the question that can be made into a checkpoint, where students have to give a correct answer to move on. Finally, students were asked how to fix the circuit and given 15 chances to pass this checkpoint before being thanked for their hard work and allowed to exit. A detailed description of the branching structure for the how-to-fix-it checkpoint can be found in Appendix B.7.

Since students had previously struggled so much with finding the correct fix, we created personalized feedback based on their responses. We also allowed students to exit the activity if they picked other solutions that isolated the problem to the same part of the circuit. For example, if students said that the breadboard was not electrically connected between the resistor and blue LED, they were allowed to exit, because realistically, if they had moved the circuit to a different row of



1. You built the following op amp circuit in lab and are trying to test whether it is working properly or not. **The forward voltages of each of the LEDs are as follows: RED = 2V; BLUE = 2.5V; WHITE = 3.5V. The resistance value of brown-black-brown resistors is 100 Ohm.** Please answer the questions below.

● Red Probe

● Black Probe

Voltage:

Figure 4.6: LED debugging problem on version 2.1 of the simulator. Students can take measurements by clicking to place the red and black probes on the breadboard. When both probes are placed the voltage appears on the screen of the DMM. The white LEDs should never light up and the blue LED should be lit up, but is not due to the resistor not being firmly pushed into the breadboard.

the breadboard, they would likely have fixed the loose resistor wire. To account for this choice, we let them exit upon making it, but first they were given the feedback that moving the components to another row of the breadboard actually fixed the problem, but the breadboard was not actually broken. They were then prompted for an alternate explanation of what might be wrong. A number of students said they were not sure when asked this follow-up, which is concerning. This could be a sign that students are just clicking and not thinking too much about these activities, or it could point to difficulties interpreting data and brainstorming plausible explanations that fit the data. A full recounting of the personalized feedback and explanations that allowed students to exit can be found in Appendix B.7.

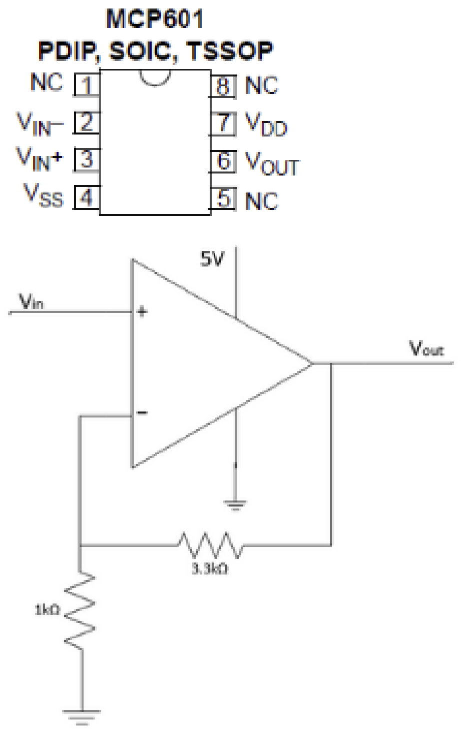
4.9.7 Non-inverting amplifier with broken op amp

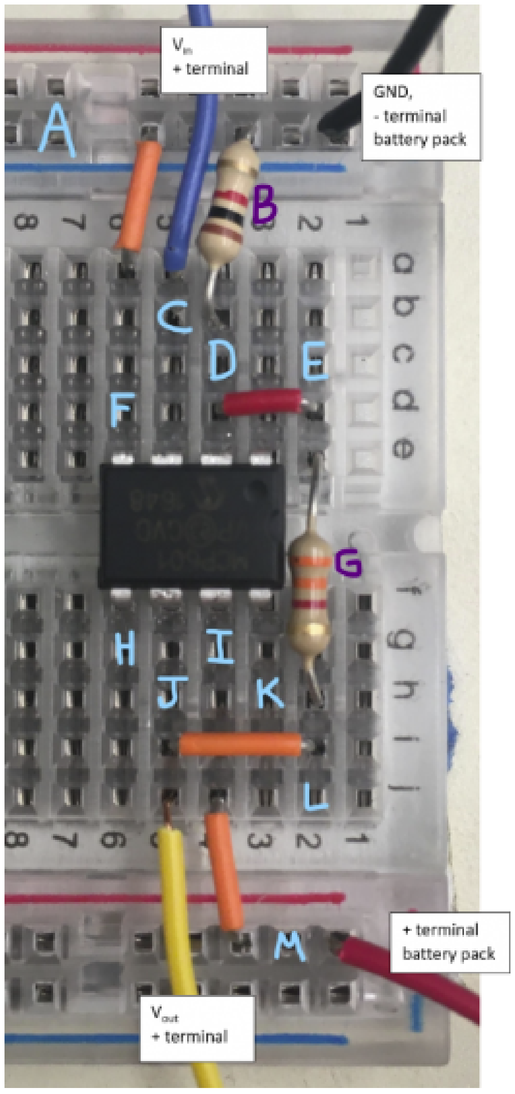
This problem was designed to test whether reflective guidance improves student performance. Half the class was given reflective guidance while making measurements (intervention group), and the other half of the class received no guidance (control group). This assignment was built on simulator 1.0 and first given in winter 2020, and an updated version was given in spring 2021. The reflective guidance experiment will be described in more detail in the results chapter in Section 6.3.2, so we will simply describe the activity here.

This activity was based on the EKG lab, where students build a non-inverting amplifier for one stage of the circuit. We chose a broken op amp as the problem because it is not visible in the picture of the circuit. Additionally, we chose this specific problem because we had had a bad batch of op amps the previous term in class, and many students had indignantly expressed their surprise that a part could be broken in their debugging journals.

In order to allow for reflective questions between each measurement, we left this debugging activity on simulator 1.0. Measurement locations were denoted by letters labeled at various locations on the picture of the physical circuit. Measurements could be taken by choosing a letter in a multiple choice question and clicking "next" (see Figure 4.7). Because the students had to click "next" between each measurement, it allowed us to easily insert the reflective questions after each measurement. In addition, this assignment did not have computer-regulated checkpoints for each step. Although the assignment could have been upgraded to an iterative assignment where students had to keep trying at each checkpoint until they chose the correct answer, the A-B testing format made it more difficult in Qualtrics to create subloops for two different paths. We also felt it important to leave the activity as similar as possible to the pilot activity for comparison.

Although the measurement system was different from our 2.0 debugging simulator and the activity did not allow students keep guessing until they got the correct answer at each checkpoint like the 2.1 debugging simulator, this activity did contain the same breadth of knowledge questions in the same order as the op amp question in our version 2.0 debugging simulator. First, students answered theoretical knowledge questions regarding the function of the circuit and the gain. Then they made





What do you want to measure? (You may only measure one at a time, but you can keep taking measurements.)

A B C D E F G H I J K L M	I am done taking measurements. The circuit is working properly.	I am done taking measurements. The circuit is NOT working correctly.
○○○○○○○○○○○○○	○	○

Figure 4.7: Non-inverting op amp circuit with a broken op amp. Simulator version 1.0

a plan using a ranking question type where they could choose and order the measurements they wanted to take. The choices of measurements were based on the most common starting strategies we had seen in the pilot studies, such as checking the power supply first or measuring the input and output first. Next, students were asked two questions about their measurement knowledge. The second of these questions was especially useful, because it was a multiple choice question about how to place DMM probes. Our debugging simulator logs the sequence of nodes that are probed by students, but it does not tell us when they think they had the correct measurement. Consequently, this question was especially useful in assessing what percentage of students knew how to compare a location marked with a star on the theoretical diagram with the location for the DMM probes on the physical circuit. It turns out over a third of the students incorrectly placed the probes both times we ran this debugging activity.

After students completed the debugging activity, we recorded measurement strategy by looking at the order of measurements students took. When they felt they had enough information, students had to choose a final diagnosis–either the circuit was working properly or it was not. They were then asked an open-ended follow-up question about their reasoning. For students who said the circuit was not working properly, they were asked what measurements indicated there was a problem and how to fix the circuit. Finally, at the end, since we did not have an iterative loop that made students guess until they found the correct fix, we asked students how confident they were in their answer. This allowed us to see whether confidence correlated with correctness. A full list of the questions can be found in Appendix B.8.

Chapter 5

Student Response & Analyzing Data

Once the choose-your-own adventure debugging simulator was complete, we needed to analyze the results from testing it on students. First and foremost, it was important that the simulator was well-received by the students. If students hated the assignments and found them useless, it would be hard to make the assignments a regular part of the curriculum. Second, we needed to find novel ways to analyze the debugging assignments, because the system no longer separated students according to right and wrong answers. Now students were separated by journey rather than destination, and thus we had to parse the data for common paths taken, using techniques like Markov matrices.

The first half of this chapter will focus on student feedback. Student feedback to the debugging activities was overwhelmingly positive. We had a number of students eagerly asking when the next assignment would come, which is far more excitement than we see for homework problems. Not only did students describe the debugging activities as fun and enjoyable, but they also said it was helpful for lab and that they learned something about debugging. Students also gave us feedback about ways to improve the navigation of the activities, which we will discuss at the end of section 5.2.

The second half of this chapter will be devoted to how we analyzed the data and made sense of students' actions. It turns out that students not taking measurements at all or submitting the assignment multiple times in order to make their performance appear better were not-so-rare phenomena, and we had to account for this accordingly in data analysis. After dealing with missing and double submissions, we also had to figure out how to make sense of the complex datasets. Since students were allowed to place both the red and black probes, and they could place them in any breadboard hole, there were around ten thousand possible unique probe locations that could be made in the system. Furthermore, for a text-based adventure with 10 possible actions on the circuit, there are millions of different sequences to pursue those 10 actions. With so many possible sequences of actions, no two students had the same exact path, which makes comparing students difficult. Therefore, we parsed the measurements and actions and grouped them in sensible ways in an attempt to better understand how students approach debugging. We created flowcharts of actions for text-based activities and timelines of measurements, which we discuss in Section 5.6.

5.1 Feedback on Text-Based Adventures

After their initial choose-your-own-adventure text-based debugging activity in Qualtrics, students were asked how much they enjoyed the assignment, how easy the interface was to navigate, and whether they learned something about debugging. Their responses were recording on a five-point Likert scale from strongly disagree to strongly agree. Students resoundingly said that they enjoyed the assignment, that the interface was easy to navigate, and that they learned something about debugging (see Figure 5.1), which indicate that students value these activities.

The open-ended responses were also overwhelming positive. 57 students gave open-ended feedback on this assignment over the two terms. 60% included positive feedback, 37% included complaints, and 25% included suggestions for improvements. Figure 5.2 shows a breakdown of the most common compliments and complaints. Two-thirds of the compliments included mentions of how enjoyable and fun the activity was, and almost half of them (44%) included exclamation marks in their praise! Another third commented how they liked the iterative aspect, describing it as a game, an electrical engineering choose-your-own-adventure, and even an escape room. Other compliments included six students who said the activity was helpful and/or realistic, and three students who wanted more assignments like this. On the other hand, we had four students describe the activity as unintuitive or not realistic enough, and four students describe the activity as frustrating or stressful. About half of the complaints suggested that a physical picture of the circuit be included on the first page or that a clearer picture of the physical circuit be included. A few students also complained that the sub menu options were confusing and that they wished there was more clues (four each).

5.2 Feedback on Simulator 2.1

We also included a feedback section on our final version of the simulator. We had two terms of students debug an op amp circuit on the 2.1 simulator, and in one of those terms, students additionally debugged an LED circuit on the simulator. At the end of each activity, we asked students to rate how difficult the problem was, how intuitive and easy to use the interface was, and how helpful they found the iterative aspect of the debugging activity that allowed them to keep trying until they got the answer right. The question about problem difficulty was on a five-point Likert scale, but the other two questions were on a four-point Likert scale, which forced students to choose either a positive or negative response. In terms of difficulty, all three assignments were

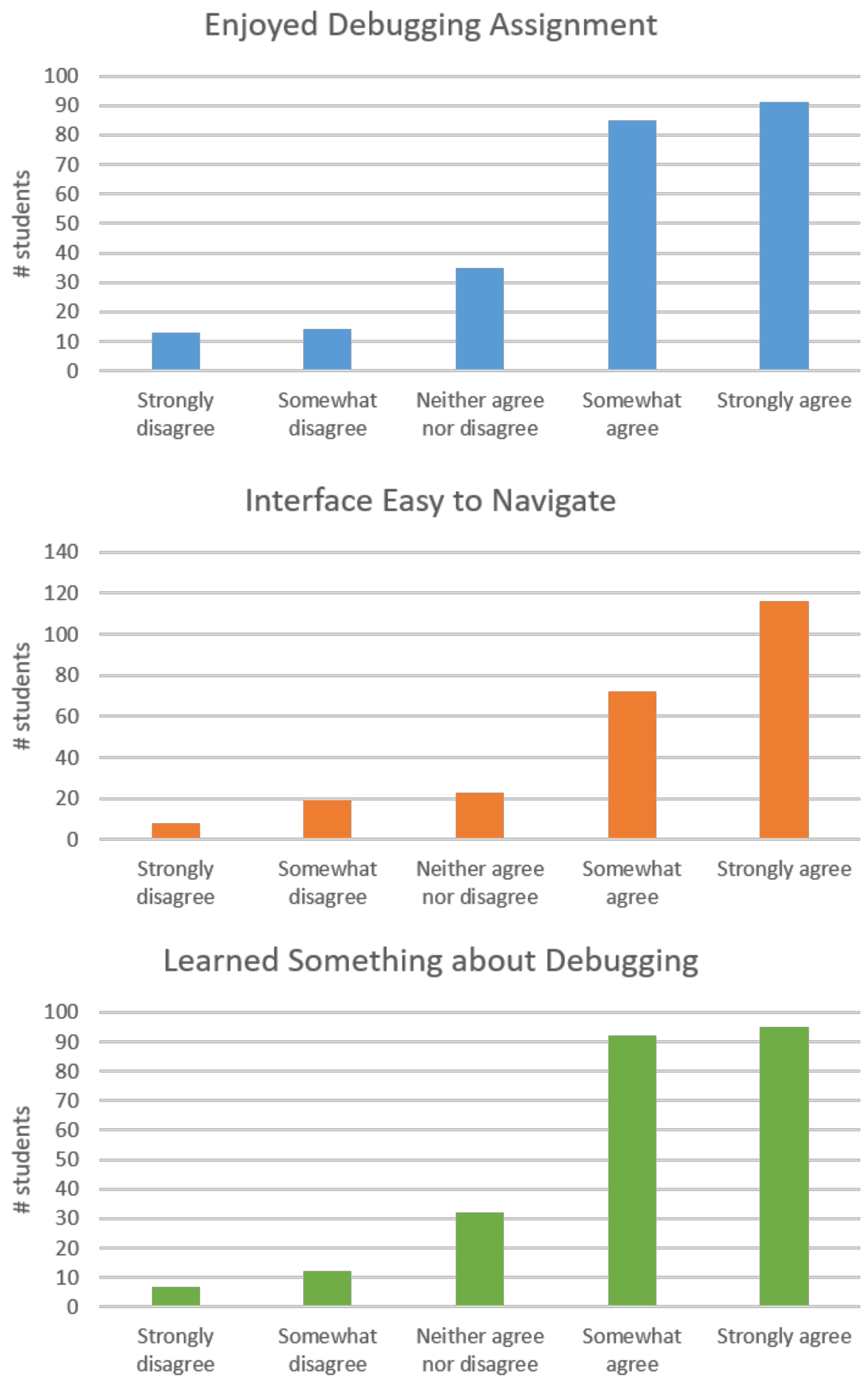


Figure 5.1: Student feedback on the debugging simulator.

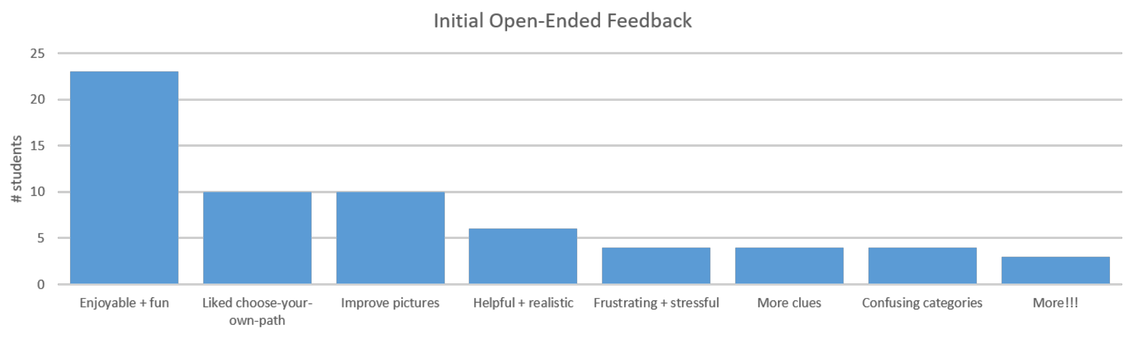


Figure 5.2: Most common open-ended feedback in order of frequency.

rated to be moderately difficult (see Figure 5.3). In general, students found the op amp problem to be more difficult than the LED problem. The students in winter rated the op amp problem a little more difficult. This result is most likely just variance from different terms, but it also makes sense in light of the spring cohort having already seen the debugging platform on a previous assignment. Their familiarity with the platform might have made the task seen a bit easier.

While the difficulty varied from problem to problem, the responses to the other two feedback questions from assignment to assignment and term to term was nearly identical, so we combined all the results together. Students overwhelmingly indicated that the iterative aspect that allowed them to keep trying until they got the correct answer was very helpful (see figure 5.4). While it requires significantly more work to make an iterative assignment, it appears that students respond positively to such a interactive assignment with immediate feedback.

Results for "How intuitive was the measurement tool?" were less positive. Students generally rated the tool as somewhat intuitive (see Figure 5.4). While this was a decrease from the text-based activities in Qualtrics that students overwhelming agreed were easy to navigate, the response still falls in the positive category of intuitive rather than unintuitive. This dip in intuitiveness could reflect both the increased complexity of the tool and areas that need improvement in the tool, which we discuss in the next two sections.

In addition to these multiple choice feedback questions, we also asked students for open-ended feedback. 130 students gave open-ended feedback on simulator 2.1 over the two terms. Equal amounts of praise and criticism were given (49% and 51%, respectively) along with about one-third of the students giving tangible suggestions about how to improve the tool (see Figure 5.5).

The most common compliments were that the tool was enjoyable, well thought out, and helpful for labs. Many students also expressed gratitude for multiple chances and feedback. 31% of the compliments described the debugging simulator activity as enjoyable, fun, and interesting. 25% of the compliments described the tool as well thought out, intuitive, easy to use, straightforward, a good

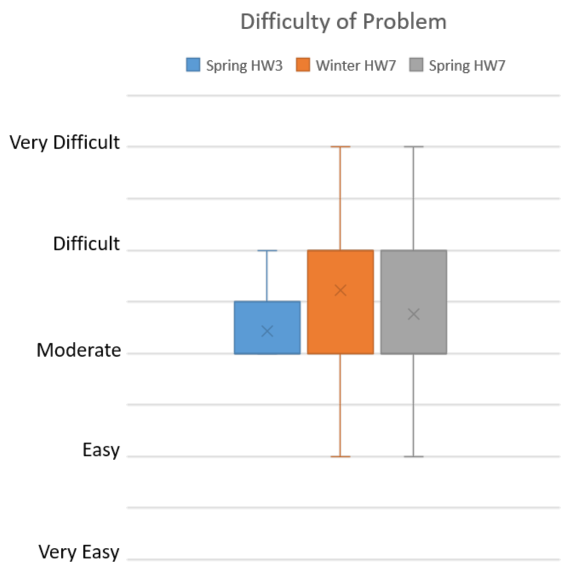


Figure 5.3: Perceived difficulty of the debugging problems on simulator version 2.1. The orange and grey box show the results for the op amp with a loose feedback wire in winter and spring respectively. The blue box shows the results for the LED debugging problem.

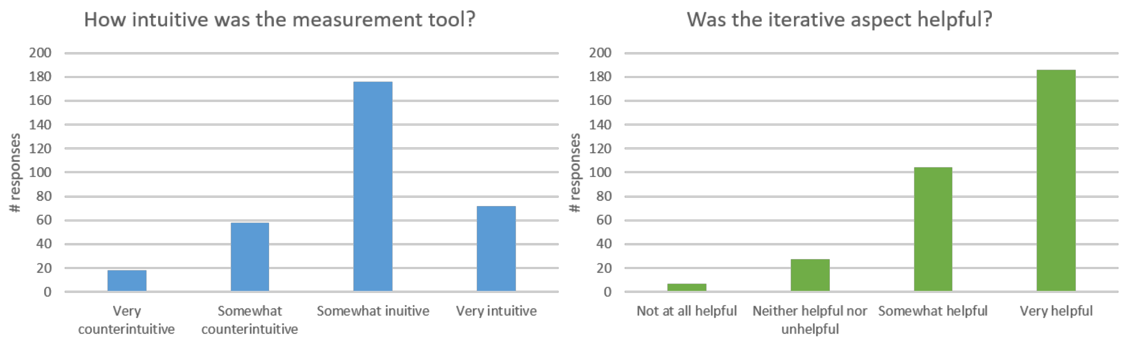


Figure 5.4: Feedback on simulator 2.1. **Left:** The blue graph shows how intuitive students found the simulator. **Right:** The green graph shows how helpful students found the iterative aspect of the simulator.

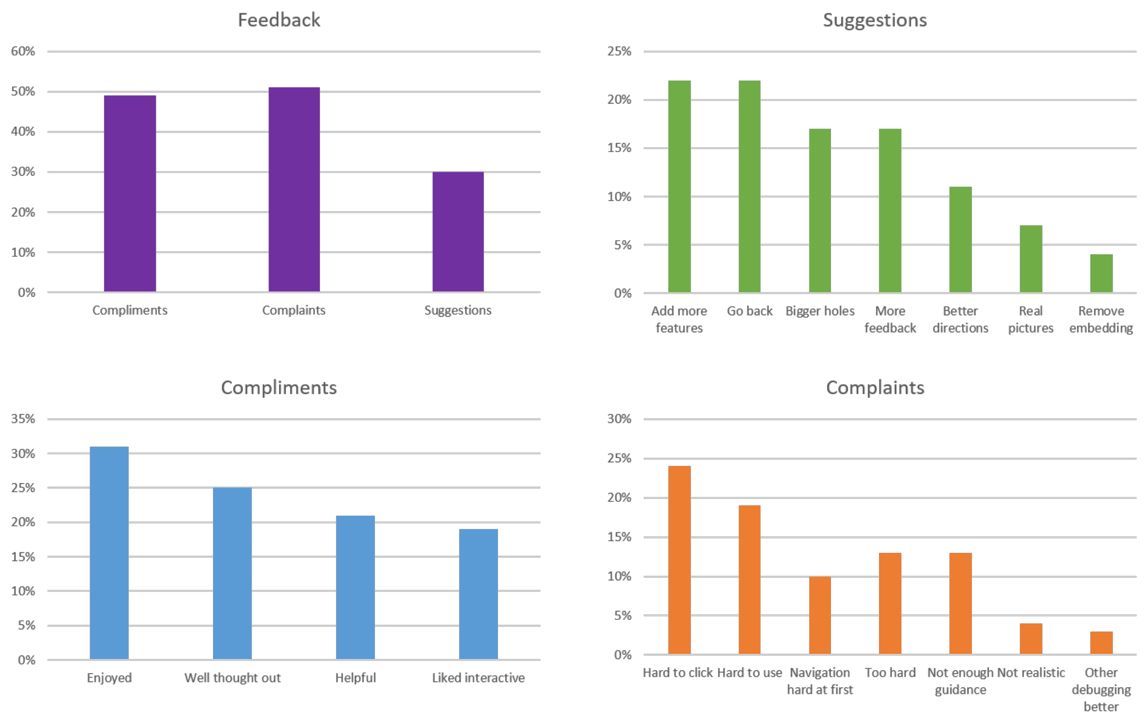


Figure 5.5: Open-ended feedback on simulator 2.1. **Top left:** Percentage of open-ended feedback that were compliments, complaints, and suggestions, respectively. **Top right:** Most common suggestions sorted by frequency. **Bottom left:** Most common compliments sorted by frequency. **Bottom right:** Most common complaints sorted by frequency.

concept, or clearer than previous debugging assignments. Another 21% of compliments described the tool as helpful or useful for lab often and included comments about how similar to their real lab experiences it was. Finally, 19% of students said they liked the iterative aspect and or feedback, with many expressing thanks for the multiple chances.

The most common complaints were that the interface was hard to navigate (53%), the problem was too hard (26%), that there were bugs (2%), and that the problem was not authentic enough or worse than previous debugging activities (7%). The main complaint related to the interface was that the simulator breadboard holes were hard to click (24% of complaints). Others simply said the interface was hard to navigate or the tool was hard to use (19%). Finally, some described the tool as difficult to use at first, but that they eventually figured it out (10%). For those who said the problem was too hard, they split equally into two categories: 1) that the problem was too hard or tedious (13%) and 2) that the problem did not give enough information or guidance (13%). For students who reported bugs, some had clear problems like their responses not being recorded (getting a field error) or not being able to see the Qualtrics page embedded in our website, but with others it was not as clear whether the issues they were complaining about were actually a bug. For example, some students said the debugging simulator was not working on their touchpad or mobile phone, but this may just have been a problem with the small breadboard holes on a small screen or a touchscreen with low clicking resolution. Finally, 4% of students complained the simulator was not realistic enough and 3% lamented the change from the Qualtrics choose-your-own-adventure text adventures to this interactive breadboard simulator.

About one-third of the students gave suggestions for how to improve the tool. The two most common suggestions were to add more features like measuring current. resistance, and connectivity (22%) and to allow students to go back and change answers (22% of suggestions). One student even suggested including an oscilloscope. Unfortunately, due to the nature of Qualtrics, we cannot allow students go back and change their answers without it changing their answer in our database. A large number of students also suggested increasing the size of the clickable area for the breadboard holes, so that the probe placement experience would be less frustrating (17%). An equal number also asked for more guided feedback and/or a summary of the correct conclusions and answers at the end (17%). 11% of suggestions also asked for better directions on how to use the tool beforehand, but the majority of these came in the winter term, where the directions page was accidentally placed a page after they had to start taking measurements. Finally, a few students asked for a more realistic image of the breadboard (3) and a better computer interface that did not require them to enter their credentials twice (2).

We can see that many of the compliments and complaints are in direct contradiction to one another, which is often the case with feedback. Some students strongly liked one aspect, while another group strongly disliked that same aspect. Regardless, we are encouraged enough by the positive feedback that we have concluded that we should add more features like measuring resistance and connectivity. We also learned from this feedback that the clickable area for the breadboard holes should be made larger, and that we should continue to strive to include more feedback and guidance in the activities for those who are struggling.

5.3 Student Preferences for Simulator Types

Our results show that students enjoyed the choose-your-adventure text-based debugging activities more than the system where they could place their own DMM probes. Students were asked this question after the third assignment in spring term, which was the inverter debugging problem built in Qualtrics. Students were able to request measurements in the text-based adventure by asking for the nodal voltage at different labeled letters in the picture. Students were asked in the feedback to

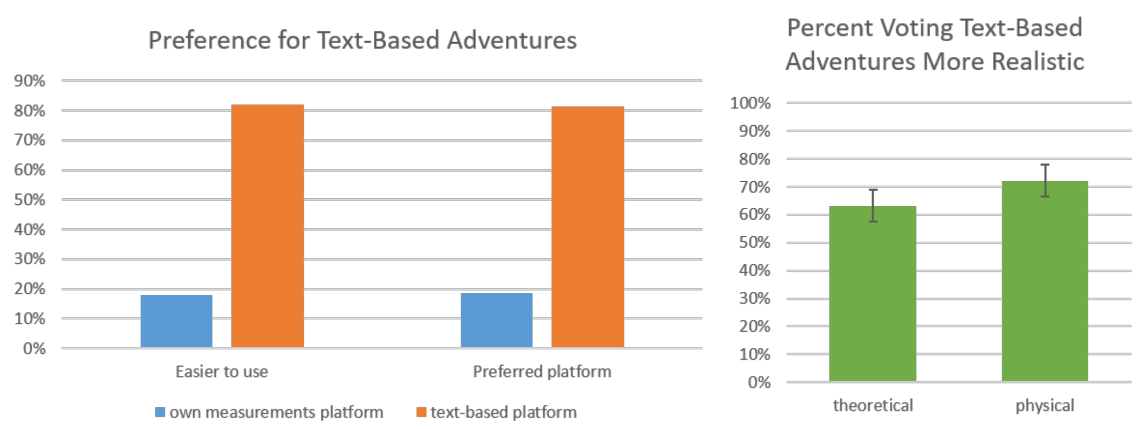


Figure 5.6: HW3 was an LED debugging assignment in our platform where students can place probes and take their own DMM measurements. HW4 was a text-based assignment in Qualtrics about debugging an inverter where students could take measurements by requesting nodal voltages at labeled letters on a picture. Half of the students in HW4 had a physical picture (the group referenced as HW4 physical), and the other half had a theoretical picture (HW theoretical).

compare this assignment to the previous week's assignment where they used our website with the usable DMM to debug an LED circuit. We asked students to compare three different aspects: 1) which platform was easier to use, 2) which platform is more realistic, and 3) which platform they preferred.

While we thought students would appreciate the authenticity of the DMM system, it turns out they favored the text-based adventures in all cases (see Figure 5.6). Almost 80% of the students said that the inverter debugging problem in Qualtrics was easier to use and that they preferred it over the system where they could take their own debugging measurements. For opinions on which assignment was more realistic, the response was more dependent on which version of the Qualtrics inverter problem students got. Since half the students simply got schematics, and the other half had physical pictures, it makes sense that the two groups might rate how realistic the assignment was differently. Therefore, we split the responses for which platform was more realistic into those who received the physical pictures of the circuit and those who only received the schematic. The students who had the inverter problem with only schematics were more likely to say that the previous week's assignment with the DMM and their own placement of the probes was more realistic, but with only 73 students in each condition (physical diagram versus theoretical diagram), the difference in results is not significant (p-value 0.28 by Fisher-Exact test). Therefore, we must conclude that there is no difference in student responses about how realistic the platform is, and that any difference we did see was likely just sampling variance.

Since only 10% of students provided open-ended feedback, it is hard for us to know exactly why they preferred one platform over the other. However, we speculate that there are several reasons that students preferred the text-based adventures over our DMM measurement platform. Based on previous cognitive science research, we believe it is possible that students are encountering cognitive overload during these debugging activities. They must think about measurement knowledge, and theoretical knowledge, and interpreting the physical circuit all at once, which can be overwhelming for novices. We also see comments related to this cognitive overload in several students' open-ended feedback. One good summary by a student was, "Find it generally easier to just retrieve the measurements rather than having to drag the connections myself as it is not representative of the movement in real life."

The other reason we have considered is that students liked the text-based adventures because they did not have the bugs and interface issues our own new platform did. As mentioned in the previous section, students struggled with clicking the breadboard holes, and several could not see the Qualtrics text embedded or complained that the embedded Qualtrics text was clunky. Therefore, it is conceivable as we improve the place-your-own-probes platform that students will start to prefer that more often.

These results have several potential implications for future debugging assignments. It is easier for us to create assignments in Qualtrics, since Qualtrics maintains the backend data management and makes sure the activities are compatible in every browser. Therefore, it appears it could be reasonable to use only text-based adventures. The only reason we are somewhat hesitant to do this is that we believe it is important to test students' measurement knowledge and their ability to properly place measurement probes. As we will show in the next chapter, a large number of students cannot place probes in the correct place on a physical circuit when shown a marked node on the schematic, which is concerning. Considering potential cognitive overload issues, it may also make sense in the future to have different activities focus on different aspects of debugging. This could help students more deliberately practice sub-steps of debugging in order to eventually master the entire process.

The other feedback we received about the different types of simulators was that once students were exposed to the iterative version where they could keep trying until they got to the correct answer, they did not want to go back to any other type of simulator. In spring 2021, we gave students a non-iterative assignment after four iterative assignments, and we received critical comments from many students in the open-ended feedback. Of 136 responses, 58 students gave optional written feedback. 43% complained about the lack of feedback on whether they were right or not. This lack of guidance led to comments expressing frustration about how difficult the activity was, with 38% asserting that the activity was difficult or was hard to follow. Not surprisingly, 22% further commented that they preferred the previous activities where they got much more guidance. In contrast, only 2 students called this assignment fun, and only one said they preferred this form over previous activities. From this feedback, it is clear that students appreciate the iterative assignments where they can keep guessing until they get the right answer. Several students even mentioned how the immediate feedback built confidence and how they felt confused without it.

Another frequent comment was that students do not appreciate having their thought process interrupted for reflection questions. This assignment had an intervention group given reflection questions after each measurement, and the control group was given random questions after each measurement (see section 4.9.7 for a full description). For students in the control group, who were asked the random questions between each measurement, 16% expressed frustration that the questions interrupted their thought process and seemed irrelevant to the problem at hand. One student even acutely commented that the questions between made them not want to test as much as possible, so they avoided testing extra components. This feedback is not surprising, since those questions were meant to be tedious and create a cost for each measurement. As we will see in the next chapter, the weighting of the control group with irrelevant questions between each measurement made the measurement patterns for both groups equivalent, so we could accurately measure the effect of the reflection activities on performance.

5.4 Improvements to Simulator

Based on this feedback, the most urgent change needed is to fix the Javascript code for the breadboard holes, such that they are easier to click on a touchscreen or small device such as a cell phone. Given more funds and resources, we would like to expand the tool to allow for resistance measurements and current measurements on the DMM along with oscilloscope measurements. We would ideally have a simulator running backend, such that students could change the input and see the response or move/change components and see the response. While adding those features, it would also be helpful to add the checkpoint questions to our own website using dynamic HTML content. This would allow us to include a back button, which was another feature students asked for in the open-ended feedback. We could then also have all the student data stored in one database system, which would make the data analysis process easier for researchers. Although these upgrades would also make the system harder to maintain, we believe the expanded capabilities of the debugging simulator are worth it.

While the student feedback discussed in the sections above shows there is room for improvement in our debugging activities, we want to focus on the overwhelmingly positive response to this new type of iterative debugging problem. When prompted to rate the activities, the large majority of students said that they enjoyed the assignments, that the assignments were easy to navigate, that they found the iterative aspect of the assignments helpful, and that they learned something about debugging, all which point to the success of our new debugging activities.

5.5 Data Analysis Issues Insights

In the process of implementing the iterative debugging activities, we ran into a number of student behaviors that hindered quick and easy analysis of the large datasets. We also had to develop new ideas for how to effectively analyze measurement data. We present both in the section below in hopes that the detailed discussion will be of use to others trying to create similar assignments.

5.5.1 Dealing with abnormal submissions

We ran into four main issues when trying to analyze debugging data: 1) Students not finishing the assignment 2) Students redoing assignments 3) Students not taking measurements, and 4) Missing database entries. We will discuss each in detail below. All of these issues lead to a significant reduction in usable data. For example, for the final assignment of spring 2021, there were 147 initial submissions, but after we removed incomplete submissions, multiple submissions, submissions missing measurement data, and submissions missing SQL entries, only 111 responses remained. In quarters where there were only 100 students to begin with and only 90 do the assignment, having only 75% of the total submissions be usable could lead to insufficient responses for statistical significance.

Incomplete submissions

Each assignment had a number of incomplete submissions. While incomplete submissions are fairly easy to catch with the Assignment Progress column in spreadsheets exported from Qualtrics, it is not always in our interest to delete all this data for several reasons. First, there may be a significant number of entries in this incomplete category, and we do not want to lose the partial data contained in them. For example, in the final assignment of spring 201, there were 26 incomplete submissions (18% of all submissions). Some submissions simply consisted of a student looking at the first page of the assignment; other submissions students answered only part of the questions; and yet other submissions students appear to have done the complete assignment and simply forgotten to do the feedback page or hit submit at the very end. Some of these partial submissions were also a second or third submission for a single student, which we will discuss in the next section. Therefore, we had to go through by hand and select which parts of which submissions to include.

Over time, we created a system to retain as much student data as possible. After deleting or merging multiple responses from students, we ended up keeping a tally of the number of students who completed each checkpoint and allowing this number to vary from checkpoint to checkpoint. Therefore, if we had responses for the theory questions at the beginning for 113 students, we would analyze all 113 responses, but if there were only 109 students who answered the next checkpoints about measurement knowledge, we would do any fractional accounting based on the 109 responses for that step. While this approach made the accounting a little more complex, it did allow us to keep the maximal amount of data possible.

Multiple submissions

Another repeated problem we had involved students submitting the assignment twice. We consciously made the choice for students to be able to submit multiple times for ease of access and to lower the stakes on the problem, so students would be less inclined to ask friends for answers. Over the two terms (12 assignments total with 1471 different student submissions), we had 90 instances of double submissions on activities. By instances, we mean number of students submitting at least two responses on each assignment (a few submitted more than two times). This corresponds to 6% of students on average submitting multiple times on each assignment. While this represents only a small fraction of students submitting double responses, these double responses required meticulous curating by us to combine their answers in a meaningful way. This led to the question of why students respond multiple times and what this can tell us about their approach to the debugging problem.

We should first note that it was usually not the same students week after week submitting multiple times. Of the 90 instances, there were 63 students involved over the two terms. In other words only one-third of the instances were from "repeat offenders." Of the 13 students who submitted multiple times on multiple assignments, only two submitted multiple times on three assignments, and two submitted multiple times on four assignments (out of a possible six assignments).

There appears to be three main types of multiple submissions, representing 80% of all double submissions. The most common (36%) is what we will refer to as the *double submission.* In this case the student did the **entire** activity twice. Oftentimes the second submission would be more efficient with less attempts before the correct answer. Sometimes, for the the second submission, the student even immediately chose the best test and then immediately chose the right answer. It appears that the student may be attempting to cover up their mistakes by submitting a second version of the assignment with no mistakes in it. This attempt to cover up mistakes by submitting a second version of the assignment is also apparent in the second-most common group, which we will call the *fixing mistakes* group (25%). The fixing mistakes group is defined by a partial attempt followed immediately by a complete response. Oftentimes the mistakes in earlier checkpoints, like calculating gain, were corrected in the second attempt, so students appear to have gotten it correct on the first try in their final submission. Once again, it seems fairly clear that students are trying to cover up their mistakes by submitting again. It is interesting to note that this small percentage of students show perfectionist tendencies to the point that even when they are told they are being graded purely based on completion, they still take time to meticulously fix all their mistakes and resubmit a "perfect" attempt.

Since these double submissions are such a small fraction of the class, it is not necessarily something we need to change, though it was time consuming for us to recombine these students' responses into one. We wanted to create an authentic retelling of the students' decisions and actions, so we therefore painstakingly went through and determined where the student left off after the first attempt and pasted those actions into their final submission.

The third most common type of double submission was the *take a peek beforehand* group (19%). This group would look at just the first page or two and then come back hours or even days later to do the full assignment. We also had 9% of students come back days later and just look at the first page or two, which we suspect was students double-checking that they actually did the assignment immediately before it was due. Lastly, we also had a few students partially redo a response later (but leave it incomplete), or a couple students who redid the entire assignment days later, perhaps forgetting they had done it the first time.

While studying the double submissions, we only found three instances of students potentially getting help from others. All three instances were on the fourth assignment in spring 2021, which was also the assignment with the largest percentage of duplicate responses (10.5%). These students took many attempts at guessing what was wrong with the circuit and then exited before getting to the intended solution. Then, several days later, they logged in and chose the correct answer on the first guess, but the only measurement they took that day was a test that would not show the intended answer. For example, one student looked at the pMOS and then said that they knew what was wrong with the circuit, immediately choosing the correct solution that solder was missing on an Arduino socket pin, which is completely unrelated. While they may have spent the previous days thinking about their first attempt and figured out the correct solution, it seems questionable that they would choose a measurement not related to finding that issue when redoing the problem.

Ideally, we would not allow students to retake the assignment, but we also had several difficulties with this approach when we tried it in Fall 2020. In this case, each student got an individualized link for their assignment. To make sure we collected responses from students who forgot to click the final submit button, we collected responses after 24 hours of inaction (which allowed us to be able to grade the assignment 24 hours after it was due). Although we told students to do the assignment in one sitting, over a dozen students needed a new link because they glanced at the assignment and then tried to come back days later. Due to the extra work this personalized link system made for the TAs and students, we decided to move to the repeatable link, which moved the brunt of the problem onto the researchers instead of the TAs.

The main take-away from these double submissions is that some students like to hide their mistakes. This corresponds with the common complaint in feedback that students wished there was a back button (which we disabled so that we can get a students' true first thoughts instead of their revised thoughts). We must keep this habit in mind when designing new forms of debugging activities, because without a carefully designed activity and incentive structure, students will try to cover up earlier incorrect work with later corrected work.

Incomplete Reasoning

Although the lack of expertise on our team stopped us from pursuing machine learning for parsing open-ended answers, we later realized this approach would not have worked due to students' tendency to leave reasoning out of their short responses. The most commonly avoided question was how to fix the circuit fault. Many students failed to give a detailed enough description of the problem with the circuit such that someone could actually fix the circuit fault. Many students seem to think that saying the gain was off and thus the circuit was not working was sufficient reasoning.

Therefore, we added a second question to the open-ended response, so the question now read, "What is wrong with the circuit? And how do you fix the circuit?". Despite this change, we still had a large number of students not answering the question how do you fix the circuit. In winter 2020 students debugging an op amp circuit with a broken op amp. Of the students who said the circuit was not working, 33% did not give any suggestions about how to fix the circuit. In Fall 2020, students received four different op amp circuits to debug over two weeks, and three of the circuits were not working. One circuit had the resistors switched, one circuit had the + and - input terminals of the op amp switched and final one had a loose wire in the feedback loop. Respectively, 10%, 16%, and 21% of students who said the circuit was not working did not suggest a fix for the circuit (see Figure 5.7).

The rates of incomplete responses goes up when the problem with the circuit is more difficult to find, such as when the circuit fault is not visible in the physical picture. Thus, we suspect students are not answering the second question because they do not know how to fix the circuit, so they just answered what they could and moved on. This issue motivated us to use multiple-choice responses, so that we could force students to decide what was wrong with the circuit and how to fix it.

Students not taking measurements

Due to some bugs on our own debugging platform, we also lost some students' measurement data. Therefore, it was sometimes unclear whether a student didn't take any measurements or if their data was lost. To better answer this question, we looked at data for another op-amp debugging assignment offered in Winter 2020 and Spring 2021. This assignment was purely in Qualtrics, so we are fairly confident no data was lost. In both terms 20% of students took no measurements or a single measurement before moving onto to answer the final questions about what was wrong with

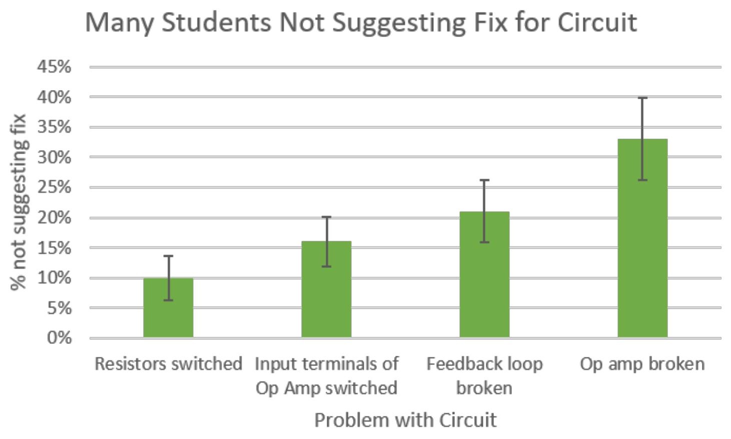


Figure 5.7: Percentage of students not suggesting fix for circuit increased when circuit fault was not a visible miswiring.

the circuit. Not surprisingly, none of the students who took less than two measurements in Winter 2020 correctly diagnosed the circuit, and only one correctly diagnosed the circuit in Spring 2021. If students were copying answers from friends, we would expect more cases of students without sufficient measurements getting the answers correct, but this is not the case. This indicates that, by-and-large, students were not copying answers from their friends, which was the goal of having a no-stakes activity based purely on participation.

On the flip side, we see that 20% of students were not sufficiently engaged in the measurement process, which is something we would like to change. While we might be inclined to attribute this to lack of effort due to the lack of points associated with performance on the activity, this also reflects a general trend we have seen with students in lab over the years. Oftentimes, as a teaching staff member, one leaves a group that is debugging to help others and comes back 10 minutes later only to discover that the first group had not managed to take any measurements. Whether this is a result of students' being apprehensive about touching the circuit, or they were not sure how to start, or they got distracted talking about spring break, we do not know, but it would be interesting to study what students were thinking in those minutes and why they chose not to investigate further. Unfortunately, that question is outside the scope of this research; nevertheless, knowing as we do that lack of measurements is a problem, we can help teach good habits by not allowing students to exit the measurement process early. For example, in future debugging activities, we can make sure not to allow students to exit until they have taken at least several measurements. It is also feasible to prevent students from exiting until they have have measured certain regions of the circuit, as was done in the final assignment of Spring 2021.

Missing Data

We ran into a few problems with data collection on our website. Around 10-20% of students who completed the activity had data on the Qualtrics side, but not on our measurement tool mySQL database. After talking with individual students when this problem initially came up, we realized that some students were redoing the assignment; and we found that on the second time they were not taking any measurements (since they had taken them the previous time), and our system was overwriting their measurements with the "new" lack of measurements. We also believe that internet connectivity or browser issues may have been causing the lost data for a few of the other students. We were able to fix our system to not overwrite old measurements for spring term 2021, which helped retention of the data of a few of the students, but there were still a number of students missing measurements, so we investigated further.

On the final assignment of Spring 2021, we had 24 students out of 135 (18%) missing either the SQL database part of their submission or the Qualtrics part of their submission. 13 students were missing entries completely in the mySQL database, which indicates to us there might have been a problem with our backend code, since a student who did the assignment and took no measurements should still have a blank entry in the database. There were also two students with measurements in the SQL database but no Qualtrics data. It appears that for the two students missing Qualtrics entries, one had only cursorily started the activity and never reached the Qualtrics part. The other student gave feedback at the end and never mentioned having problems with the Qualtrics part of the survey, so perhaps their data was lost by Qualtrics, though that seems unlikely.

For the rest of the nine students, they had a mySQL database entry, but no measurement data. Looking on the Qualtrics side, we see that two of the students quit toward the beginning of the activity, which indicates that they probably did not take any measurements. Ironically, one of those two gave feedback ratings for the activity, but since we can see exactly which part of the activity they stopped at in Qualtrics, their ratings are probably suspect. Of the seven remaining, four of them also have feedback in the database, which indicates that they made it to the end of the activity. Three of those four even gave open-ended feedback, indicating they were putting in effort even at the end of the activity. We were thus not sure how to differentiate between students who actually did not take any measurements and times when our backend data management might have made errors. As we noted in the previous chapter, 20% of students simply did not take more than a single measurement during a debugging activity. Therefore, it is plausible that each of these students actually did not attempt to take measurements.

Based on these idiosyncrasies we identified, we had to go through and appropriately combine multiple submissions and separate out students with incomplete submissions to create a clean dataset ready for analysis.

5.6 Parsing measurement and action data

Since our final version of the simulator is based on number of attempts at each checkpoint rather than correctness of answer, we had to come up with novel ways to analyze the data. The emphasis on journey led us to create graphs of the path taken. In the case of measurements, this meant timelines of measurements, and in the case of text-based adventures, this meant flowcharts of actions taken. We showcase both below.

5.6.1 Timelines of measurements

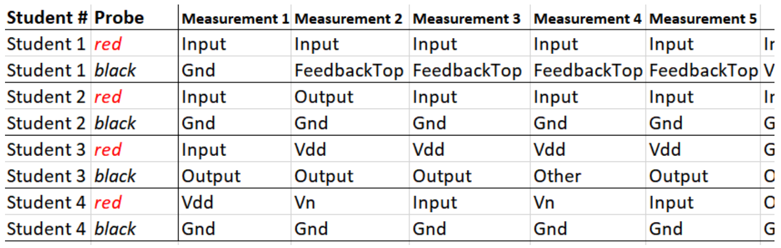
While the text-based adventures in Qualtrics allow for a straightforward analysis of what measurement was taken first and what measurement was taken second in the confined space of 10-12 possible locations, the DMM programmed on our own web page allows for a seemingly infinite variety of measurement patterns. About one-third of students do not just place the black probe in ground and keep it there, and even those who do make dozens of clicks in dozens of places, so no two students have the same measurement path.

Data was stored as an entry for each student in a mySQL database. One column contained a list of clicks. For clicks on the breadboard, the coordinate of the click location was saved. For selecting of the red or black probe buttons, the words "black" or "red" were listed accordingly. Each entry was separated by commas. The next column had a list of timestamps for when each of these clicks occurred.

Using this information, we could reconstruct each student's measurements using a python script. In order to parse the measurements, we had to create states to keep track of where the black and red probes were. A measurement is only counted when a student had placed both the black and red probe somewhere. This eliminated a lot of extraneous clicks at the beginning as students figured out how to use the tool. We then called upon a mapping function to find what row of the breadboard the coordinate was located in. This function was a constant for all activities on our simulator, since the breadboard is designed to be always in the same pixel location. We then called a second mapping function that was specific to that particular activity to turn the row of the breadboard into names representing the specific nodal voltage, like input or ground.

Since there were two probes, we kept track of both probes separately. This resulted in a 2-row table that has a descriptive location for both the red and black probe listed for each measurement (see Figure 5.8).

To better interpret the data, we then grouped various combinations of red and black probe placements into categories of measurements. For example, in the op amp circuit with the loose feedback wire, if you place the DMM probes in the output row and any row sitting at ground, we would categorize that as an "Output" measurement even if the student had flipped the probes and had the red probe in ground. Other categories included the input row, the negative input pin row,



Student #	Probe	Measurement 1	Measurement 2	Measurement 3	Measurement 4	Measurement 5	
Student 1	*red*	Input	Input	Input	Input	Input	Ir
Student 1	*black*	Gnd	FeedbackTop	FeedbackTop	FeedbackTop	FeedbackTop	V
Student 2	*red*	Input	Output	Input	Input	Input	Ir
Student 2	*black*	Gnd	Gnd	Gnd	Gnd	Gnd	G
Student 3	*red*	Input	Vdd	Vdd	Vdd	Vdd	G
Student 3	*black*	Output	Output	Output	Other	Output	O
Student 4	*red*	Vdd	Vn	Input	Vn	Input	O
Student 4	*black*	Gnd	Gnd	Gnd	Gnd	Gnd	G

Figure 5.8: Example of parsed measurement data for simulator version 2.0. Records were kept of the location of both the red and black probes for each measurement.

the power rail, anywhere on the feedback loop, and any empty row. We also checked if students measured the voltage across a resistor or between the two input pins–measurements where one probe would not normally be in ground. Any measurement that did not fit in one of these categories would be marked as "Other."

These categories then helped us determine whether students ever looked at various regions of the circuit. For example, we could determine if students ever put a measurement probe in the feedback loop on the amplifier circuit with the loose feedback wire. To isolate the circuit fault, it was key to measure the voltage drop across a wire in the feedback loop. It turned out that in winter 2021, 39% of students never placed a measurement probe in the feedback loop. We concluded that if students were failing to collect key information, our interactive simulator should provide feedback to prompt students to measure other parts of the circuit. This insight was a major reason we created the extra checkpoint where students were guided to check different subsections of the circuit and mark them as working or not. We will save the discussion for whether prompting students to look at more parts of the circuit actually worked for Section 6.3.3.

Using the python scripts outlined above, we can create matrices stating whether students measured different areas of the circuits or not and how many times they did. While this was useful in the case of measuring whether students looked at the feedback loop in the op amp circuit with the loose feedback wire, it was not very helpful for measurements at other locations or other circuit scenarios. A somewhat more meaningful metric was the measurement number of the first time a student measured a given pertinent location. For example, perhaps a student first measured the output voltage 4th, the power supply 10th, and the input voltage 50th. This knowledge helps us get a relative sense for each student of the order in which they measured key things and whether they did that relatively early or relatively late in their measurement sequence.

We also tried creating correlation tables with the number of times and the first time students measured different key places and compared against how many tries it took them to isolate the circuit fault. Unfortunately, the correlations were weak. The strongest correlation was how many times a student measured the feedback loop at -0.21, meaning that the more times a student measured the feedback loop, the fewer number of attempts they needed on average to isolate the cause of the

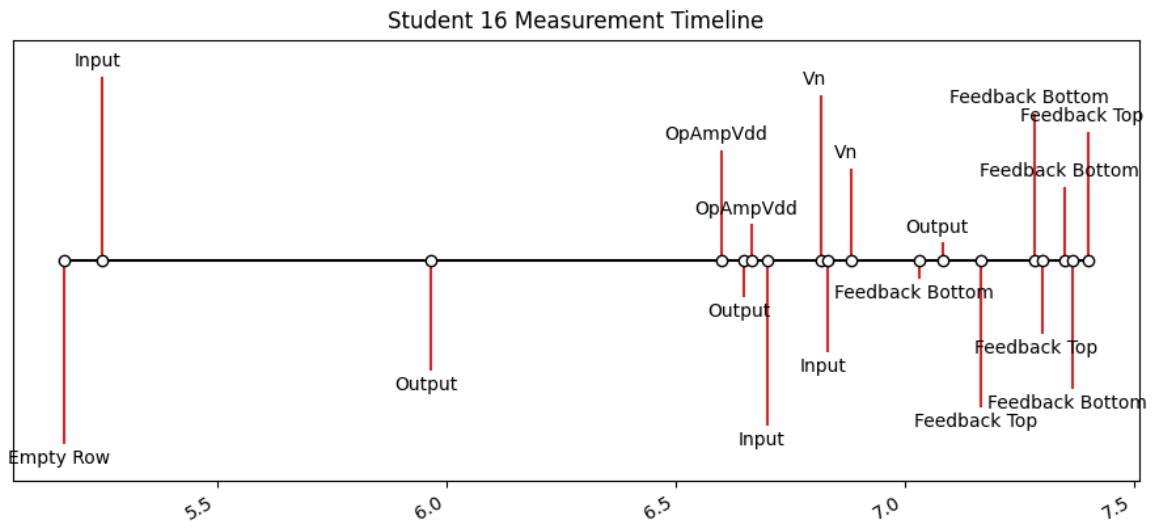


Figure 5.9: Measurement timeline for student 16

circuit fault. This makes sense because a student who notices the discrepancy in the measurements on the feedback loop will likely keep investigating in an effort to isolate the cause. Therefore, it should be no surprise that a student taking many measurements of the feedback loop correlated with their more quickly isolating the cause.

Another interesting metric we pursued was *when* students took measurements. We wanted to know if students took many measurements all at once or if they had periods of thinking and periods of investigating, which would appear as small clumps of measurements interspersed with breaks. In order to test this, we used the timestamps to create timelines of students' measurements.

Figure 5.9 shows a timeline for a student that correctly found the loose wire in the feedback loop of the op amp debugging activity on their first guess. This particular student was a strong student who correctly answered almost all the checkpoints on the first try (but did take two tries on gain). Notice they did not start taking measurements until around five minutes in because they were likely reading the problem statement and answering the theory questions first. We see the student first measured the input and then the output. They then moved on to checking that the power was correctly supplied to the op amp. After that, they checked both input pins of the op amp, and, finally, they started looking at the feedback loop. They checked the nodal voltage above and below the loose feedback wire multiple times, which likely represented their trying to make sense of the voltage drop across the wire. At this point, they successfully isolated what was wrong with the circuit, and they correctly answered how to fix it on their first try and exited the activity.

We can also look at several of the timelines together and look for trends. Figure 5.10 shows the measurement timelines for 35 random students tiled on one page. For a handful of the students we can see periods of rapid measurement where the density of measurements was so strong that the timeline of measurements just looks solid red blur interspersed with periods of no measurements. This cyclic alternation of rapid measurements and then pauses was something we also observed when making video observations of our TAs debugging circuits in lab.

We also wanted to add the data about when students completed various checkpoints to the timelines. Unfortunately, we were not able to precisely line up measurement timings between the Qualtrics pages and our website, since Qualtrics gave duration timestamps instead of absolute timestamps. Some information-only pages in Qualtrics were also missing timers, so if students spent a lot of time paused on these pages, it messed up the integration of the the two timings. Despite this, we were able to obtain reasonable matched timelines for some of the students.

We show a sample timeline in Figure 5.11. Student 23 is another strong student who answered all the checkpoints correctly on the first try. Similar to student 16 discussed above, they did not start taking measurement until a number of minutes into the assignment. Using this integrated timeline, we can see that this time gap corresponds with the time the student spent answering the theory checkpoints, creating a plan, and answering the first measurement knowledge question. Only then, when asked to measure the input voltage in the second measurement knowledge checkpoint, did they start taking measurements. It appears that they were able to diagnose the circuit as not working merely from comparing the gain of the circuit to theory, since they had measured only the input and output at this point. Then they set to work trying to find where the problem was in the circuit. First they checked the power supply, then there was a rapid flurry of measurements that includes checking the power is properly supplied to the correct op amp pin, checking both input pins of the op amp, and taking 5 measurements on the feedback loop. We presume this was the point when the student realized what was wrong with the circuit, and they immediately chose the correct fix from the list of choices and exited the activity.

In contrast to these students, we can look at a struggling student and analyze their approach (see Figure 5.12). Unlike student 23, whose timeline was just described, this student struggled with measurement knowledge and isolating the cause of the circuit fault. A look at their measurements sheds light on why they struggled to isolate the circuit fault. It looks like they successfully determined that the circuit was not working based on calculating gain from the measurements of input and output and seeing that it differed from theory. After that, the student only measured a single other place on the circuit–the negative input pin of the op amp. At this point, they gave up taking measurements and rapidly started guessing what is wrong with the circuit to try and get past

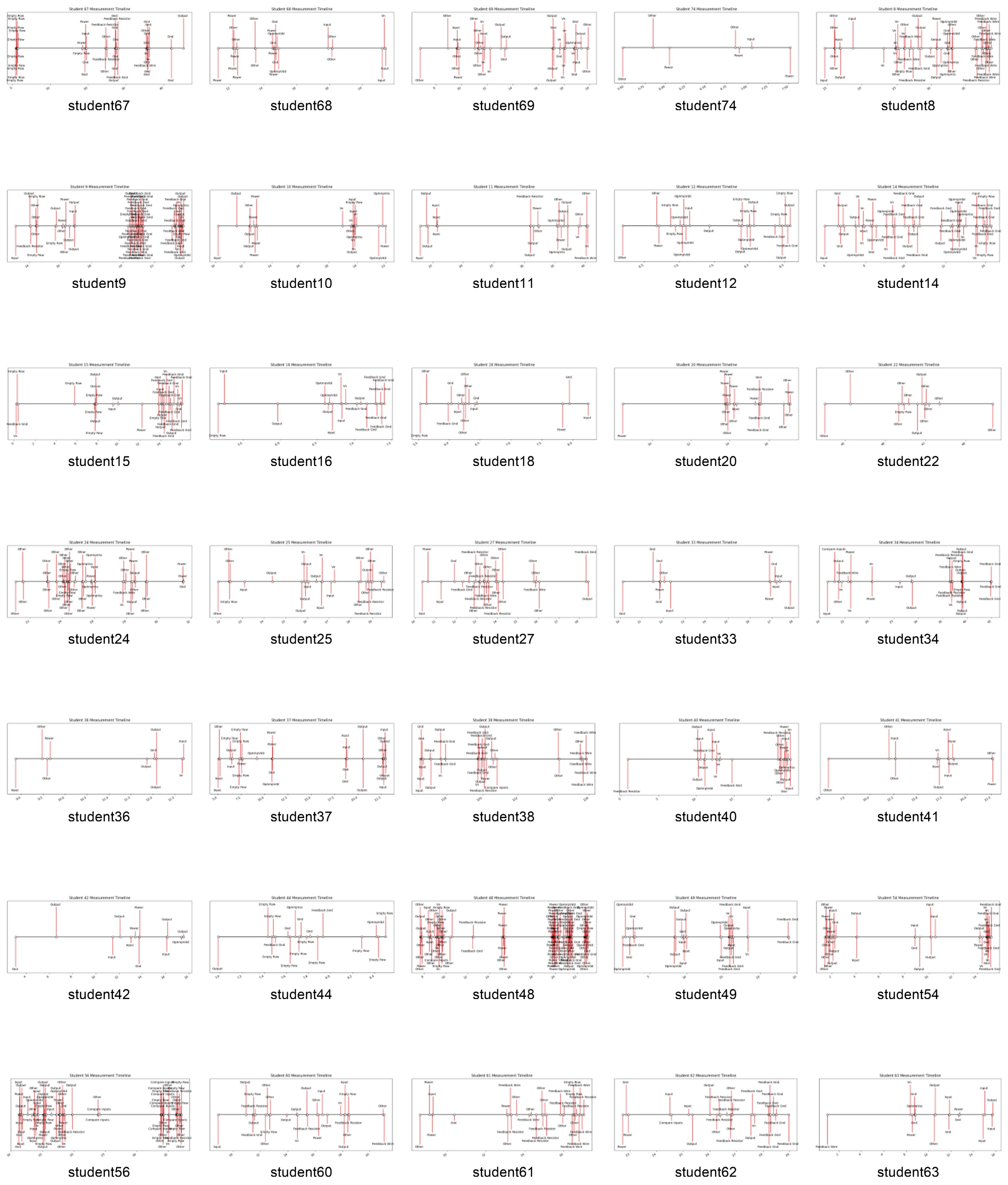


Figure 5.10: Measurement timelines for 35 random students

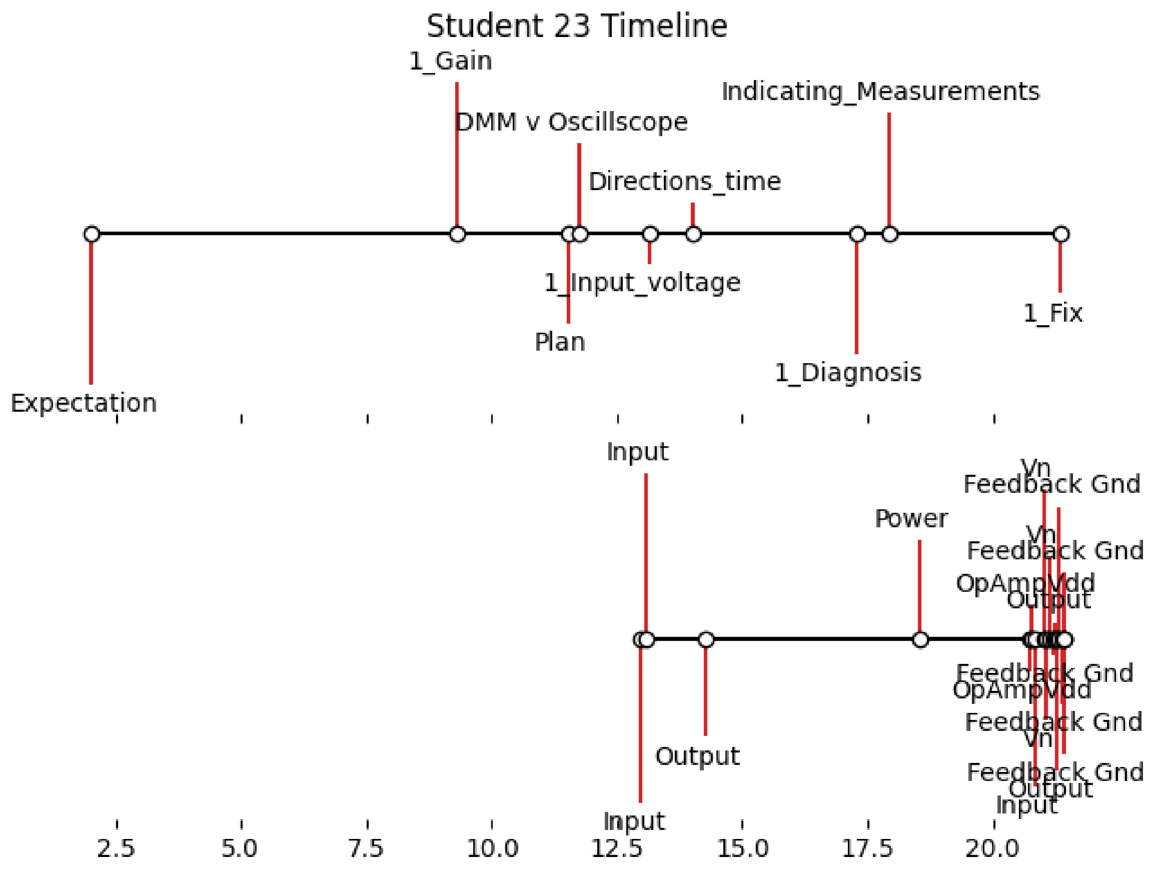


Figure 5.11: Combined measurement timeline for student 23. The top timeline shows when the student answered each of the questions in Qualtrics. The bottom timeline shows when the student took each of the measurements. The timelines are lined up, so they can be compared.

the final checkpoint. After 15 tries, they maxed out on responses and were allowed to quit the activity. It is not surprising that they never guessed what was wrong with the circuit, because as the measurements show, they never looked at the feedback loop. Even when their guesses were wrong, they never decided to take more measurements.

5.6.2 Action flowcharts

Since not all our activities had students take voltage measurements, we had to find other ways to make sense of students' actions for the text-based adventures. For these activities, we used several techniques to look at aggregate data and individual data. For aggregate data, we created matrices with each action represented by a state. We then looked for common transitions between states. While we were able to find common threads at the beginning and end of students' journeys, the large number of possible paths and variance in path length between students made it hard to characterize the aggregate data further. For analyzing individual students, we created visual flowcharts of their actions using the graph description language DOT [43]. This created a visual story of each students' debugging journey, which gave us further insight into how students approached debugging.

A good debugging activity to showcase the value of such flowcharts is the activity where students debug their 8x8 LED display where one row is not working. Students were given a variety of places on the board where they could check if something was wrong, such as the Arduino, transistors, resistors, or LEDs. Therefore, their choice of where to check next is an indicator of how they were interpreting information to try and isolate the location of the fault. We anticipated that some struggling students would randomly choose where to investigate, but other stronger students would have a strategy where they choose locations that were consistent with the clues. To better differentiate between students who were choosing randomly and students approaching the problem strategically, we often put in "red herring" actions that involved replacing parts that were very unlikely to cause the symptom listed.

When the only information students had is that one row of their display is not working, most students (68%) started by checking the transistor that turns on that row. This demonstrates that students were correctly interpreting the symptoms (a row not working) to isolate a probable single cause, i.e., the pMOS in that row was not working correctly. When students investigated the pMOS, they learned that it was connected correctly and working, so they had to brainstorm a new possible cause and use that idea to decide upon the next thing to check. Just as most students had a common starting point, most students also had a common ending point to their investigation. Almost all students (92%) correctly ended their investigation right after they looked at the solder joints on the back of the PCB (and noticed the solder missing on one Arduino socket pin).

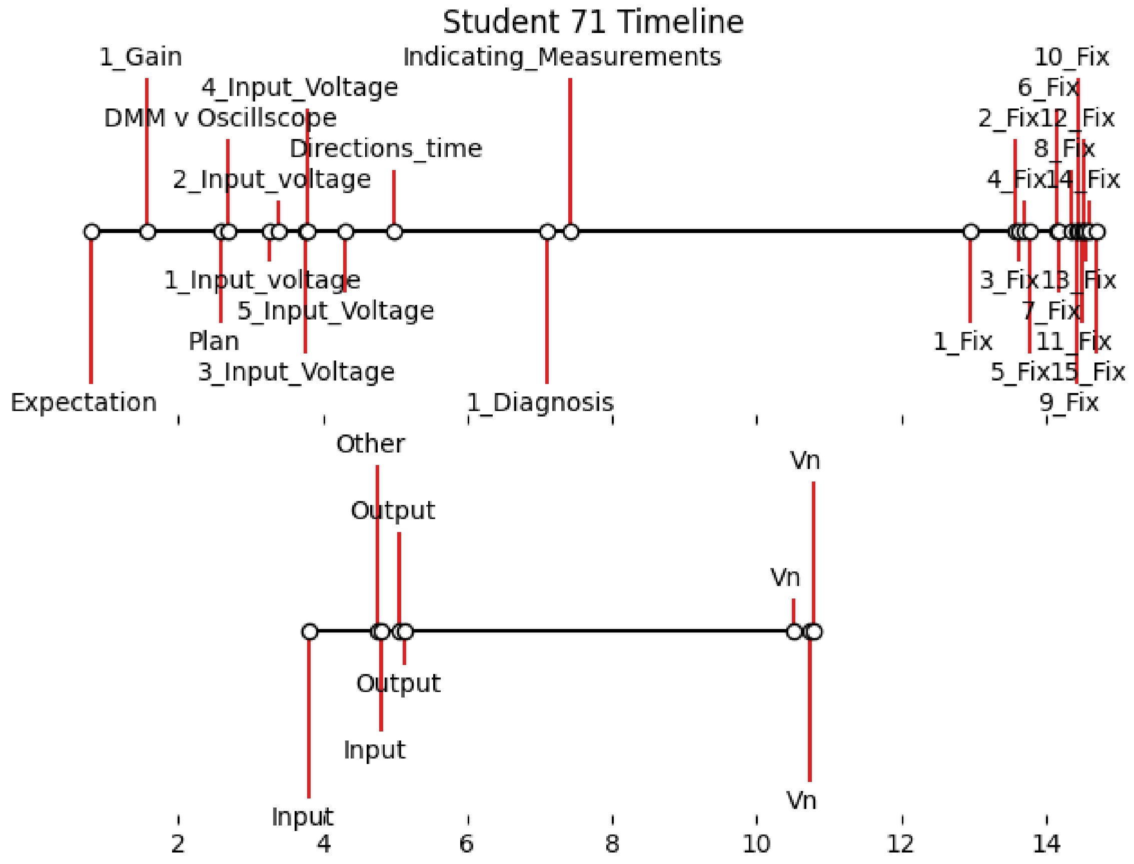


Figure 5.12: Combined measurement timeline for student 71. The top timeline shows when the student answered each of the questions in Qualtrics. The bottom timeline shows when the student took each of the measurements. The timelines are lined up, so they can be compared. In this case, the student did not take sufficient measurements, gave up taking measurements, and then tried guessing 15 times what was wrong with the circuit before timing out and being allowed to exit the assignment.

In terms of the red herrings, roughly 40% of students checked the resistors, despite the fact that it would be impossible for broken resistors to be the issue based on these symptoms, because problems with the resistors would have manifested in an entire column, not a row. While we cannot know for certain why students chose to check the resistors, the large number of students indicates that either students were falsely interpreting data or that most students were just randomly guessing, both which are concerning. We also found that 14% of students who looked at the pMOS in the first round chose to replace the pMOS instead of inspecting it. We hope that with improved training students can learn to check components first instead of just blindly replacing them.

Another way of analyzing the data is to look at individual students' actions. To do this, we constructed directional graphs showing each student's sequence of actions (see Figure 5.13). We created python scripts to parse the data and output DOT files (a graph description language). The DOT files were then visually rendered as flowcharts.

Oftentimes, the order of a student's actions painted a clear picture of their shifting hypotheses of what is wrong with the circuit as they gathered more information about the problem. In the flowchart shown in Figure 5.13, the student looked at the pMOS first, then tried replacing the pMOS, and then decided to check the Arduino code. When the Arduino code looked fine, the student went back to their starting point to see if they had missed something. This is a good example of a student circling back to where they started when they were stuck, which was something we commonly saw in the action flowcharts. Next, the student checked the soldering on the path from the Arduino to the pMOS. They must have noticed that there is solder missing on one of the socket pins, because they immediately say they are done taking measurements and correctly diagnose what is wrong with the circuit.

While that student kept investigating until they knew what was wrong with the circuit, others would try to exit early. Figure 5.14 shows a different student who seemed fairly convinced at the beginning of the process that the problem was the pMOS. They started by checking the pMOS, and after checking, they clicked that they had identified what was wrong with the circuit, but they were incorrect, so they were sent back to investigate. They then again checked the pMOS and again said they knew what is wrong with the circuit. At this point, they appeared to give up on the pMOS as the problem and moved on to check other ideas such as the Arduino code, the LEDs, and the solder joints on the back of the PCB. Like the previous student, as soon as they saw the back of the PCB board, they successfully finished the debugging process.

The flowcharts aided us in our efforts to reconstruct what a student was thinking about while debugging. Since students do not freely write their first guesses down, we can use their actions as a proxy. With the necessary tools and python scripts to parse the data, we were then able to analyze the data and what it means from a education research perspective. The results are discussed in the next chapter.

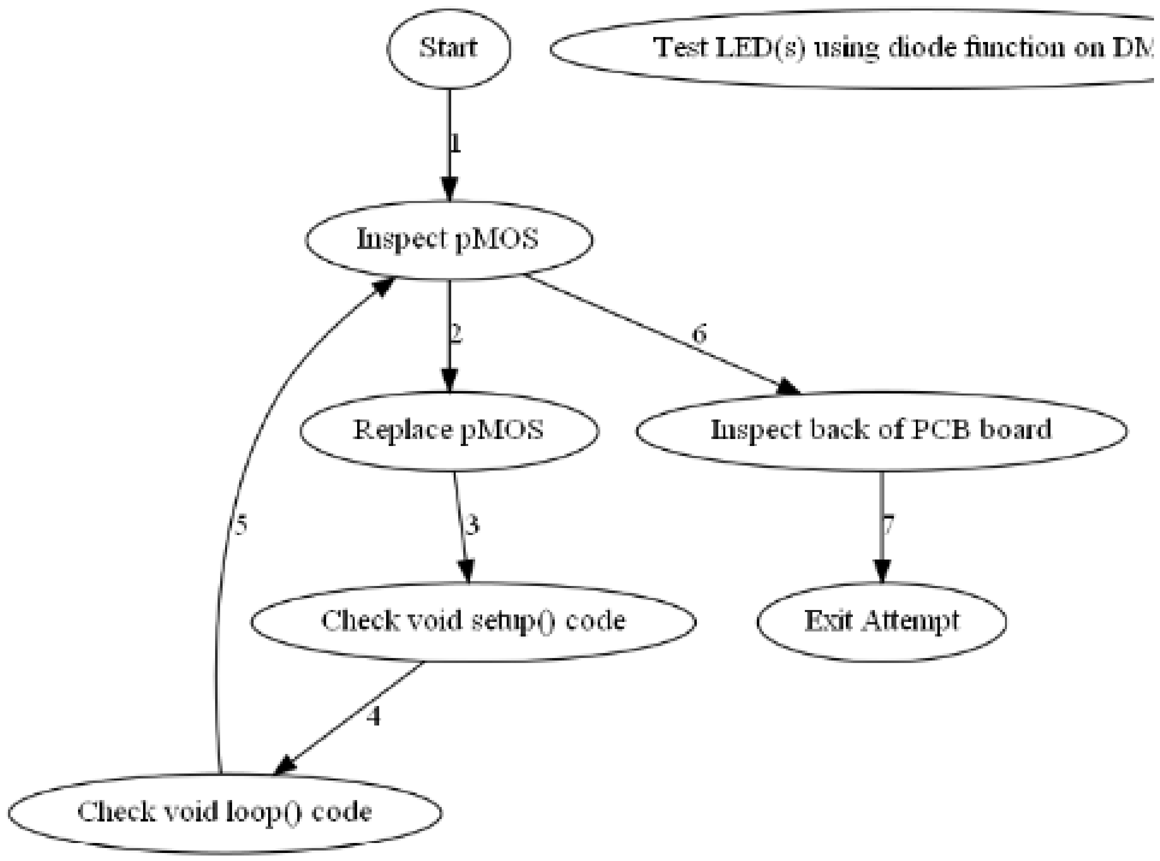


Figure 5.13: Student action flowchart for the 8x8 LED display with a row not working. This particular student checked the pMOS first, then checked the Arduino code, then checked the pMOS again, before finally inspecting the back of the PCB board and finally discovering the missing solder on one of the Arduino socket pins.

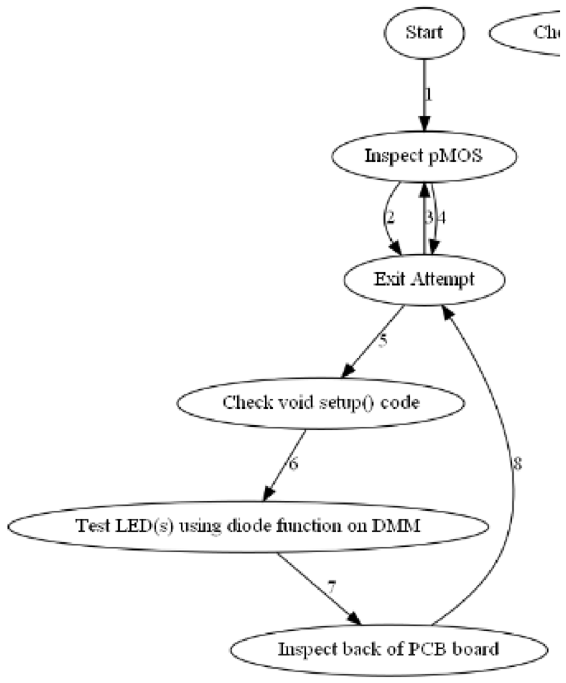

Start
Ch
1
Inspect pMOS
2
3
4
Exit Attempt
5
Check void setup() code
6
Test LED(s) using diode function on DMM
7
8
Inspect back of PCB board

Figure 5.14: Student action flowchart for the 8x8 LED display with a row not working. This particular student inspected the pMOS first, then incorrectly thought they knew what was wrong with the circuit, then checked the pMOS again, then incorrectly thought they knew what was wrong with the circuit again, then finally moved on to checking other parts of the circuit like the Arduino code and the LEDs before looking at the back of the PCB and finding the missing solder on one of the Arduino socket pins and successfully exiting the activity.

Chapter 6

Studying Debugging Behavior (Results)

Up to this point, this thesis has focused primarily on the design and implementation of a functioning circuit simulator that would allow researchers to effectively study and teach circuit debugging. The final chapter of this thesis interpret the results of the data gathered from over 500 students. We look at how students approached debugging, what parts they struggled with the most, and how our interventions improved student performance.

By breaking down the complex activity of troubleshooting into smaller cognitive steps, we were able to study how performance on these smaller parts affected the whole. We had two theories about the cognitive subtasks of debugging coming into these experiments. According to the first theory, there would be certain subtasks (demarked by checkpoints in our simulator) with which all students struggled. If this were the case, we would be able to create activities to specifically target these subtasks in the future and help them deliberately practice. According to the second theory, a few students would "fall off the bandwagon," so to speak, each step of the way. In this latter case, there would be no common barrier for students, and we could treat each cognitive subtask as a piece of a puzzle. If the student had enough pieces of the puzzle, they could still figure out what the picture was, but if they were missing too many pieces, they would be unlikely to solve the puzzle (i.e., correctly diagnose what is wrong with the circuit).

It turns out that both of these theories are supported by our results. There was a single step that caused most students trouble, which was the final step of isolating the cause. The vast majority of students (80%) struggled with this final step, even if they performed well on every previous step. Beyond this, there were a number of steps that at least one third of students struggled with, but for the most part performance on any of these other checkpoints did not dictate whether a student correctly diagnosed what was wrong with the circuit or not.

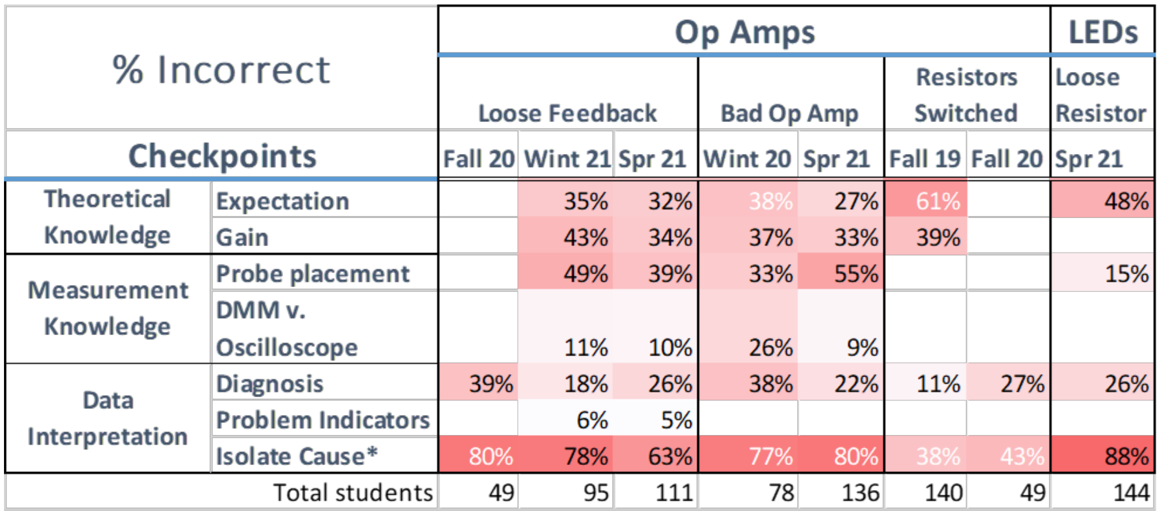
An analysis of the checkpoints leading up to the final one of isolating the cause reveals that the average student would get most the checkpoints correct and struggle with only one or two. The specific checkpoint that each student struggled with varied from student to student. The more checkpoints a student got wrong, the more likely they were to not successfully diagnose the circuit. On the other hand, getting all the checkpoints correct did not necessarily mean the student correctly isolated the cause of the circuit fault in the end, because the main step students struggled with was interpreting anomalous measurements and drawing conclusions from that data to isolate the cause.

Since so many students struggled with interpreting the data to isolate a possible cause, we tried several different interventions to help students, only one of which actually affected final performance. Our first intervention was to make the checkpoints have immediate feedback, whereby students were told their answer was wrong and were then given more chances to keep trying until they got the answer correct. The second intervention was to give students reflective guidance and see if that helped students better diagnose the circuit. Finally, we created another checkpoint where students highlighted various regions of the circuit as working or not working to help them isolate the cause, and this intervention appears to have been somewhat successful, unlike the first two.

6.1 Checkpoint data

As we discussed in section 2.4, students need many types of knowledge to successfully troubleshoot a circuit, including theoretical knowledge, physical knowledge, and measurement knowledge. To better understand what parts of debugging students struggled with the most, we analyzed checkpoint data. See sections 4.6 and 4.9 for a review of the general checkpoints. Figure 6.1 shows the percentage of students who got each checkpoint wrong on the first try for four different assignments. We can see that the difficulty of the problem varied depending on what was wrong with the circuit. For the case where the non-inverting amplifier had the two resistors switched, significantly more students correctly diagnosed the circuit than they did for circuits where the problem was not a visible miswiring.

The more red a box is, the greater number of students who struggled with that checkpoint. We see that the most difficult step for students is isolating the cause for circuits where the the circuit fault is not a visible miswiring. For activities where the circuit fault was not a visible miswiring, 75-80% of students were not correctly isolating the cause of the circuit fault in the first two tries. Furthermore, a good number of students still had not correctly diagnosed the circuit fault after 15 tries. Figure 6.2 shows a histogram of the number of tries it took students to correctly identify the cause of the circuit fault on the op amp circuit with a loose feedback wire from Winter 2021. Due to the difficulties students had with isolating the cause, we made this step the focus of our interventions, which we will talk about later in this chapter.



% Incorrect		Op Amps							LEDs
		Loose Feedback			Bad Op Amp		Resistors Switched		Loose Resistor
Checkpoints		Fall 20	Wint 21	Spr 21	Wint 20	Spr 21	Fall 19	Fall 20	Spr 21
Theoretical Knowledge	Expectation		35%	32%	38%	27%	61%		48%
	Gain		43%	34%	37%	33%	39%		
Measurement Knowledge	Probe placement		49%	39%	33%	55%			15%
	DMM v. Oscilloscope		11%	10%	26%	9%			
Data Interpretation	Diagnosis	39%	18%	26%	38%	22%	11%	27%	26%
	Problem Indicators		6%	5%					
	Isolate Cause*	80%	78%	63%	77%	80%	38%	43%	88%
	Total students	49	95	111	78	136	140	49	144

Figure 6.1: Percentage of students needing multiple attempts at various checkpoints. The percentages in white represent questions that were open-ended responses instead of multiple choice. The asterisk on Isolate Cause indicates that students were allowed 2 attempts for a more equitable comparison to the terms without multiple choice, where students often gave two suggestions for what might be wrong with the circuit.

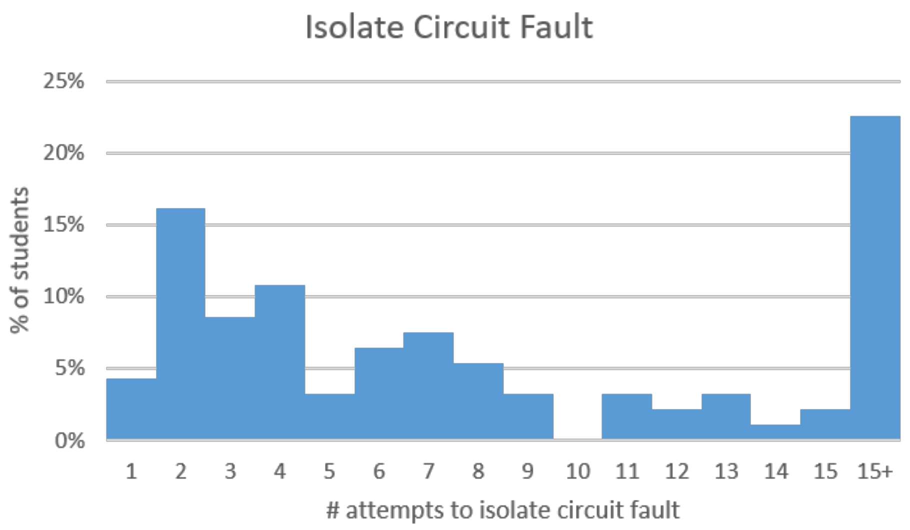


Figure 6.2: Histogram of number of attempts to correctly identify the cause of the circuit fault. Over 20% of students never isolated the cause of the circuit fault and were allowed to exit after 15 tries.

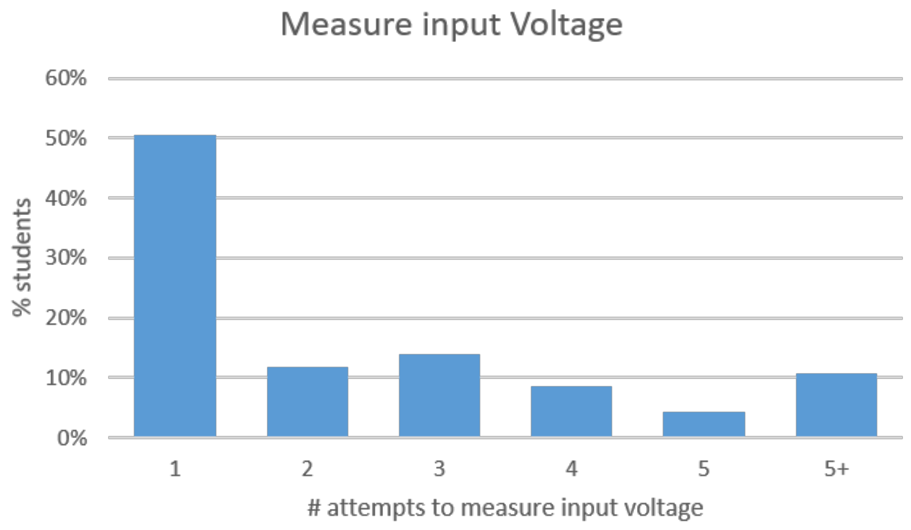


Figure 6.3: Histogram of number of attempts it took students to correctly identify input voltage. The majority of students correctly identified the input voltage on the first try, but 10% of students still had not figured out how to measure the input voltage after 5 tries.

The next most difficult step for students averaged out over the terms was measurement knowledge about probe placement when working with op amps. Nearly half the students (45%) incorrectly placed measurement probes on the first try in the debugging activities with op amps. For the debugging activity with the bad op amp, students were given a starred location on the circuit schematic and told to choose the correct locations for the red and black probes on the physical circuit (which was marked with letters to designate possible locations for the probes). For the other two terms of op amp data (loose feedback wire), students were told to measure the input voltage to the op amp circuit. They were then allowed to interact with the DMM and take their own measurements until they thought they had the correct answer, and they would then type in the voltage for verification at the checkpoint. Figure 6.3 shows a histogram of the number of tries it took students to correctly measure the input voltage in Winter 2021. Although students were allowed to freely interact with the circuit and correct their own mistakes before answering, the percentage of students answering correctly on the first try did not improve compared to the multiple choice version.

It turns out measurement knowledge data changed drastically depending on what students were asked to measure. We see that the very same group of students who struggled to place DMM probes on the op amp circuit with a loose feedback wire in Spring 2021 performed much better when having to perform the same task on an LED circuit (see Table 6.1). Only 15% of students incorrectly measured the battery voltage on the first try in the case of the LED circuit, whereas 39% incorrectly

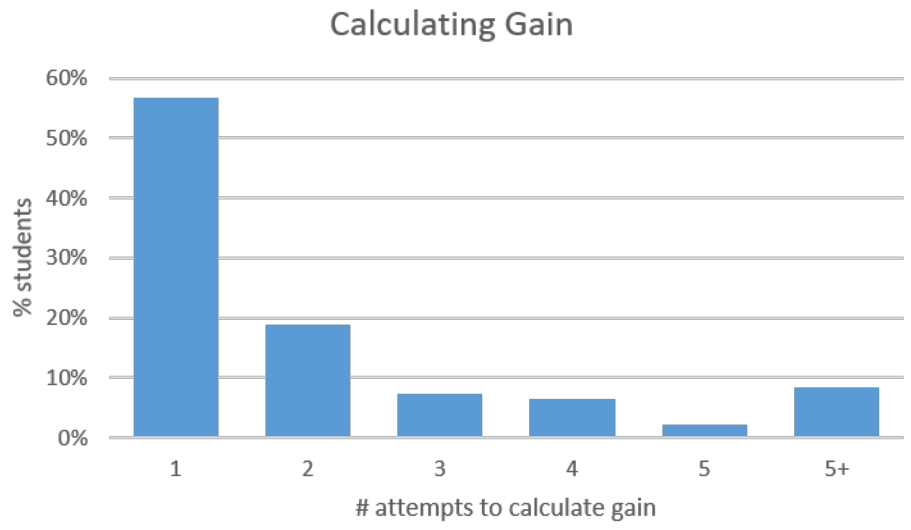


Figure 6.4: Histogram of number of attempts to calculate gain. The majority of students correctly calculated gain on the first try, but 8% still had not successfully calculated gain after five tries.

measured the input voltage to the op amp circuit on the first try. This result is interesting, because the LED circuit was the first circuit students saw on our DMM simulator, and the op amp circuit was presented in a later week when they had mastered the DMM simulator. Because the students struggled more with measuring a given voltage after they were already familiar with the simulator, we believe that the difference in performance actually indicates a misunderstanding of the abstract concept of an input signal rather than a misunderstanding of how to take measurements. Further testing will have to be done to confirm this hypothesis.

Not only did we find students struggling with measurement and physical knowledge, but we also found them struggling regularly with theoretical knowledge. Our results show that 30-40% gave incorrect answers to theoretical knowledge questions on their first try. Since theoretical knowledge, like how to calculate gain or what LEDs should be on, is something already widely tested in homework, we were a bit surprised by the number of students struggling with these steps. Perhaps this is a sign that students are not checking answers with their classmates as they do on other homework problems (as we hoped for by giving participation points, not correctness points). More encouraging is that 80% of the students who were incorrect on the first try were able to self-correct when given feedback that their answer was incorrect (see the histogram in Figure 6.4).

We further wanted to know how all these pieces of knowledge affect a student's debugging performance, so we looked at how performance on each correlated with successfully isolating the cause of the circuit fault. We highlight the data from winter 2020 here as an example, but similar trends were found for the other debugging assignments in other terms. In winter 2020, the debugging activity involved a non-inverting amplifier that had a broken op amp. This activity occurred before the final version of our simulator, so students had only one chance to answer each question. We asked the following questions to individually test each of these knowledge realms. For theoretical knowledge, students were asked to calculate gain. For measurement knowledge, students were asked whether a DMM or oscilloscope would be more appropriate for the task at hand. Finally, for physical circuit knowledge, the students were asked where on the breadboard to put the red and black probes of their DMM to measure the voltage at a marked point on the circuit schematic. The expectation was that students who missed any of these were more likely to come to incorrect conclusions about the circuit, and the results agree with this expectation.

Based on these three different domain knowledge questions, we gave each student a knowledge domains score, with a maximum score of 3 if a student answered the questions in each of the three domains correctly and a minimum score of 0 if a student answered none of the questions in the three domains correctly. Similarly, we gave a diagnosis score, with 1 point for each of the following correct answers:

1. determined circuit is not working properly
2. isolated the problem to the input of the op amps being unequal
3. suggested the fix should be replacing the op amp

As with the knowledge domain score, students could have a maximum score of 3 and a minimum score of 0 on the diagnosis score. The results are shown in the table below (Table 6.1).

	Diagnosis Score				
Knowledge Score	3	2	1	0	Avg Dx score
3	12	6	6	1	2.16
2	3	6	5	11	1.04
1	0	4	4	2	1.2
0	0	0	0	2	0

Table 6.1: Correlation knowledge domains score to final diagnosis score. The more knowledge domain questions a student got wrong, the less likely they were to correctly diagnose the circuit fault in the end. Average diagnosis score for each given knowledge domain score is shown in the far right column.

Students who displayed correct knowledge in all three domains represented the vast majority of students who also got full diagnosis points. In contrast, no students missing at least two domains of knowledge got full diagnosis points. Furthermore, the two students who were not able to show domain knowledge in any of the three domains received no diagnosis points because they said the circuit was working correctly. In general, the average diagnosis score decreased as the average knowledge score decreased. These results correspond with our view of debugging as building a puzzle, where the knowledge domains are the puzzle pieces. A student can still figure out what the final picture should be even if one or two pieces of the puzzle are missing (i.e., missing one or two pieces of domain knowledge), but, on the other extreme, if most of the puzzle pieces are missing (checkpoints incorrect), then it is hard to tell what the final picture is.

While the example data above is from a single term, we can construct similar tables for other terms with a few minor adjustments. We next focus on spring 2021, where students were assigned to debug a non-inverting amplifier with a loose feedback wire. In that term, we required students to keep trying until they correctly diagnosed the circuit fault. Therefore, the students' diagnosis score is a number between 0 and 15, representing the number of attempts it took to correctly isolate the cause of the circuit fault. The knowledge score in this case was based on the number of checkpoints correct on the first try. The six checkpoints were as follows:

1. purpose of the circuit (non-inverting amplifier)
2. gain
3. best measurement tool (DMM vs. Oscilloscope)
4. probe placement
5. diagnosis (circuit working or not)
6. problem indicators

Unlike the winter 2020, this term included immediate feedback for students at the checkpoints. This scaffolding allowed students to correct their mistakes and helped keep them on track. Therefore, we see that even students getting the majority of the checkpoints incorrect could still correctly diagnose the circuit in the end, given enough attempts (see Figure 6.5). Furthermore, unlike winter 2020, we see that getting a single checkpoint incorrect had barely any effect on the final isolation of the cause. This makes sense because in such a case, a student was able to immediately correct their mistake. In the end, students who got all the checkpoints right on the first try took half the number of attempts to correctly isolate the cause of the circuit fault than those who got half the checkpoints incorrect.

Although these trends seem telling, we must remember that they are averages. Even if a student got all the checkpoints correct, this did not guarantee that a given student correctly isolated the cause of the circuit fault. For example, in the group with zero incorrect checkpoints, there were still students who never correctly isolated the cause of the circuit fault (and these two students with 15 tries greatly pulled up the average number of attempts for their group). On the flipside, for the group of students who got at least 4 checkpoints incorrect, the minimum number of attempts it took a student to isolate the cause was five tries. That is, knowledge of these checkpoints is necessary to successfully isolate the cause of the circuit fault, but it is not sufficient.

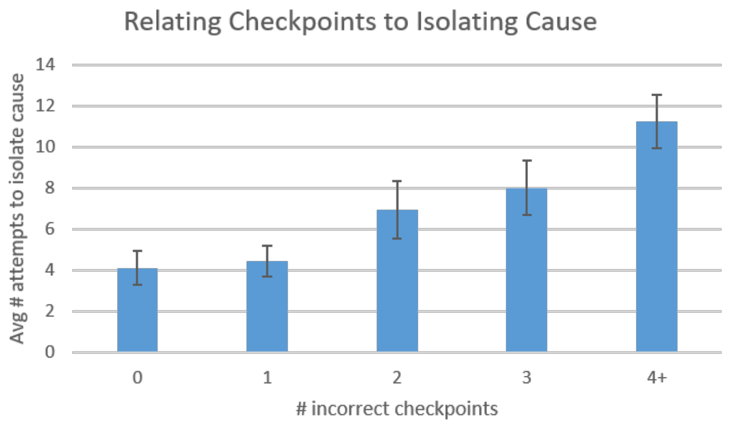


Figure 6.5: Performance on checkpoints directly related to final performance on isolating the cause. As the number of incorrect checkpoints goes up, the number of attempts needed to isolate the cause also goes up.

6.2 Debugging Strategies

We noticed a number of interesting behaviors that students had when debugging, which affected either their ability to figure out what was wrong with the circuit and/or our ability as researchers to interpret their results. The most striking relates to our assumption going in that we would find certain measurement patterns that indicated a good debugger. While it is true that we found students using a number of strategies mentioned by Jonassen, like topographic (e.g.,following the signal from left to right in the circuit schematic) or functional/discrepancy detection (checking the input and output first to compare against gain) [9], none of these strategies were as significant as whether or not the students simply took measurements in the area of the circuit where the problem was.

For insight into students' debugging strategies, we looked at the order in which they took measurements. We were curious if students taking certain measurement paths would be more likely to correctly find the cause of the circuit fault than others. To measure this, we specifically looked at which part of the circuit students chose to measure first. We chose to focus on this part because students were so diverse in the measurement paths they took that only the starting point was shared by students.

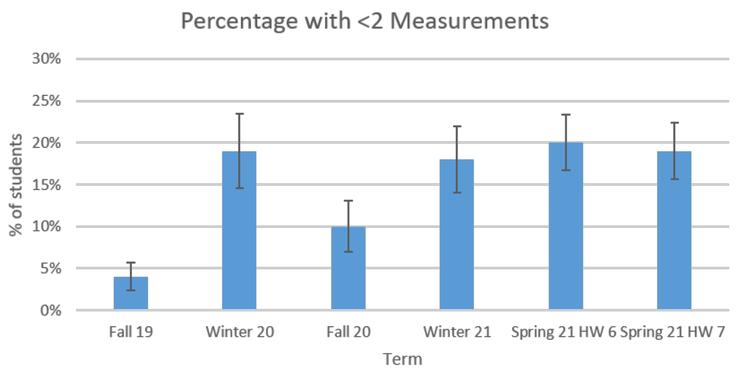


Figure 6.6: Percentage of students taking less than two measurements by term. Note for unusual term: Fall 2019 the assignment was worth 10% of students' final exam score, which may have made students take assignment more seriously.

In order to properly analyze the measurement data, we first had to remove the students who did take measurements. As we discussed in the last chapter in 5.5.1, many students did not take sufficient measurements, and that was a problem that we faced most terms. Not only did we see upwards of 20% of students not taking any measurements or only a single measurement in any given term, but we also had students failing to measure relevant parts of the circuit. It should come as no surprise that virtually all of the students who did not take measurements did not correctly diagnose the cause of the circuit fault.

Figure 6.6 shows the percentage of students each term taking less than two measurements. Percentages range from 4% to 20% each term. We believe the rate is very low in Fall 2019 because the problem was worth 10% of their final exam score, which was a big incentive for students to put in hard work on the problem. Winter 2020 was plagued by the start of COVID, so half the class did not turn in the assignment in the midst of being sent home from campus, and those who did were stressed and often rushed (as indicated in open-ended feedback), which may have contributed to the uptick in incomplete measurements. Winter and Spring 2021 have similar high levels of insufficient measurements, which may have been caused by fatigue from weekly debugging assignments (this was the first terms they had debugging assignments each week) or disengagement by a some students when we switched to the iterative format with checkpoints. If a student was allowed to keep guessing until they got the correct answer, they might have avoided thinking about what to measure and just have randomly started guessing answers until they got it right to minimize mental effort. We could try to address these students taking insufficient measurements either by requiring a minimum number of measurements or taking points off if they do not take enough measurements.

After removing the students with insufficient measurement data, we then looked at what the most common first two measurements were as a measure of starting strategy. Since so many students were taking atypical measurements on the interactive DMM (e.g., black probe not connected to ground) in simulator 2.0, we decided to look at earlier simulator 1.0 where the data was a little cleaner. The pilot studies all involved op amp circuits, and students were given a picture with a limited number of labeled locations where they could measure the nodal voltage. Due to the smaller number of measurement possibilities, the space of measurement patterns was much smaller. Furthermore, in these older version simulators the final circuit diagnosis was a binary of correct or not correct because students were not given multiple tries, which made comparison of starting strategy and chances at successfully isolating the circuit fault a little more straightforward.

For the op amp circuits on simulator 1.0, we found the majority of students (around 70%) used one of 6 strategies:

1. check input and output first
2. check resistances first
3. check power rails first
4. check the input terminals of the op amp first
5. check voltage divider first
6. check voltages alphabetically (each measurement location was labeled with a letter)

Checking resistances appears to be a form of checking for miswirings first. Checking the input terminals appears to be a strategy of starting at the left side of the circuit schematic and moving left to right like reading. Checking voltage divider first means students checked the output and the negative input pin first, which we call voltage divider because it is essentially measuring the voltage across both resistors and then the voltage across one resistor. Students employed these same six basic strategies in surprisingly similar frequencies over three terms. The results are shown in Figure 6.7.

Fall 2019 and Winter 2020 (blue and orange) are a good comparison showing baseline variance for different terms and different issues with the circuit (one had the issue that the resistors were switched, and the other had a broken op amp). Despite these differences, we see relative consistency across terms. In contrast, in Winter 2020 and Spring 2021 (orange and grey) the same problem was given out, the only difference being that in Spring 2021 students had done 4 previous debugging assignments. Therefore, the extreme differences in percentage of students using each strategy are notable. Specifically, we see there are two statistically significant changes: 1) the number of students checking resistors first decreases and 2) the number of students checking the input terminals of the op amp first rises significantly. (We also see a decrease in the number of students checking input-output first, but this is not statistically significant). Since the only difference between the terms was

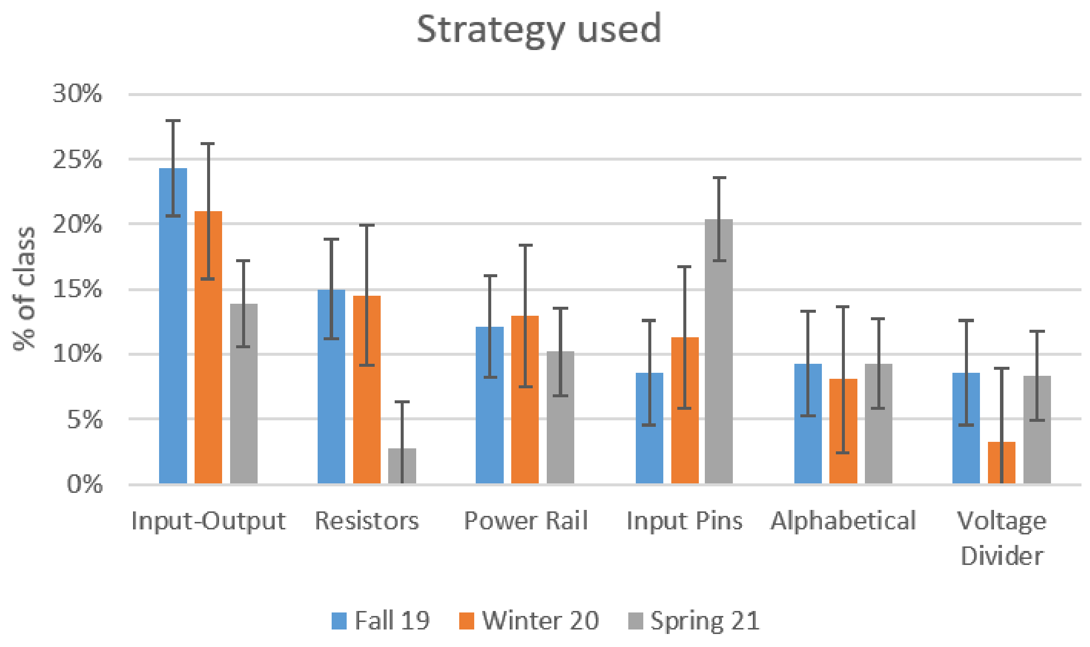


Figure 6.7: Starting measurement strategy used when debugging op amp circuits for various terms. Blue represents Fall 2019 term (amplifier with resistors switched), orange represents Winter 2020 (amplifier with bad op amp), and grey represents Spring 2021 (amplifier with bad op amp).

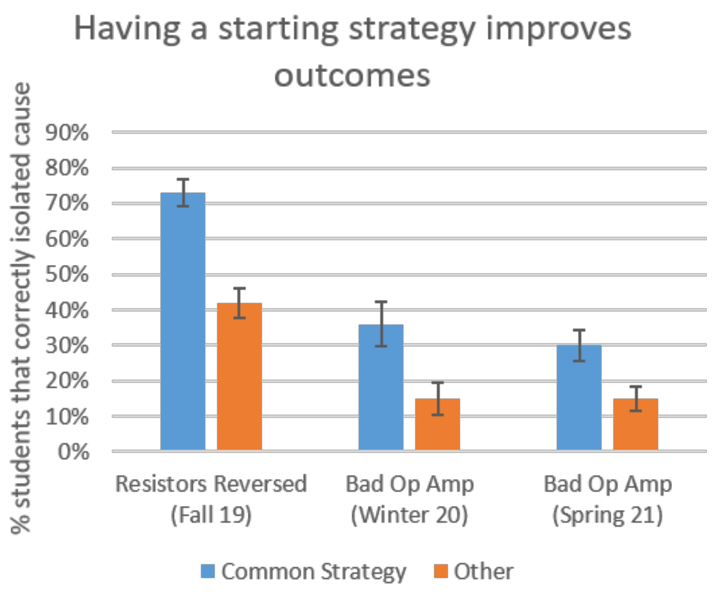


Figure 6.8: Percentage of students who correctly diagnosed the circuit fault based on their starting strategy. Students using standard starting strategies perform better. Blue bars represent students who used one of the six common starting measurement strategies and orange bars represent all other students.

the number of debugging activities done before this one, this change in strategy from checking the wiring first to instead making measurements following the schematic from left to right could be a result of the previous debugging practice. A follow-up study should be conducted to ascertain what effect participating repeatedly in our scaffolded debugging activities has on student strategy.

We also found that students who used one of the six common strategies were better at finding the circuit fault than their counterparts (see Figure 6.8). In Fall 2019, the problem with the circuit was that the resistors were switched. In winter 2020 and spring 2021, the problem with the circuit was that the op amp was broken. Because the problem with the circuit was different in Fall 2019, we see vastly different absolute percentages, but the difference in performance between the group using one of the six common strategies and all other students is the same from term to term. (Please note that any students who did not take at least two measurements were removed from this analysis). While the students who did not use one of the common 6 strategies may indeed have had a strategy, the wide variance in measurement order in this group indicates that many were probably clicking randomly. Based on previous troubleshooting research indicating that strategic knowledge is needed to successfully debug a circuit, it should come as no surprise that those who were likely clicking randomly performed worse.

Finally, in an effort to understand the most common strategies of struggling students, we looked at what the most common reasons were that students gave for why they thought the the circuit as working. We intended to learn common wrong answers from this, but instead we learned that students by-and-large were not giving incorrect data or interpretations, but rather giving accurate but incomplete data. They were choosing to ignore any discrepancies they found and instead focus on the reasons they found that would indicate the circuit was working. In other words, they were cherry-picking the most favorable data.

To further analyze this, we looked at the submissions on the op amp debugging activity in winter 2020. In this term, the circuit had a broken op amp, which was causing the input pins of the op amp to have unequal voltages and the output to be saturated. We looked very closely at the reasoning of the 14 students who said the circuit was working properly. We noticed 5 different actions/arguments from the students who diagnosed the circuit as working, which may have led to their incorrect conclusion that the circuit was working. The reasons are shown in Table 6.2.

Inappropriate action taken	**# students**
Argues that ratio of Vin- and Output are correct	5
Notes gain is off after 2 measurements, but later ignores	3
Never calculated gain	2
Considered checking wires sufficient	2
Didn't take sufficient measurements	2

Table 6.2: Reasoning of students who said circuit was working for broken op amp circuit in Winter 2020.

Most of these reasons involve arguing that the gain is correct. There appears to be several paths of reasoning that lead students to believe that gain is correct. The first is a simple misapprehension of the pin out diagram. If a student mixes up the negative and positive input terminals of the op amp, they may accidentally compare the negative input terminal to the output, which gives the correct gain ratio. The other incorrect paths of reasoning revolve around a student trying to make sense of discrepancies in the measurements. Some students are not confident in their ability to interpret locations on the physical circuit to the circuit schematic, so they point to the location on the circuit that gives the correct ratio of voltages they were expecting for gain, even if they suspect that is not the correct place to measure. In a similar line of reasoning, other students may not have confidence in their theoretical gain calculation, and thus defer to the empirical gain they find and say it is the correct gain. We see this in the three students who noted the gain was off after two measurements, but then later ignored this fact when they made their final conclusions. It was as if students were trying to fit the all the information together into the most coherent story, and if a few details did not fit, they simply ignored them in favor of the majority of the data. This makes sense in the worldview of a novice student who knows they make mistakes, and so they have trained themselves on traditional homework assignments to choose the answer that fits best. Unfortunately, in the world of debugging, this is the wrong approach, because debugging needs to occur whenever there is a SINGLE discrepancy in the data.

After scrutinizing winter 2020 in detail, we noticed similar reasoning in larger numbers in other terms. The most common reasoning given for why the circuit is working was:

1. the wiring was double checked and correct (34%)
2. the circuit amplifies (32%)
3. the voltages are as expected and/or the gain is correct (30%)

Other common, but less frequent reasons, given included that voltages were as expected, the input terminals were equal as they should be according to the Golden Rules, and the circuit correctly inverts the signal (output is negative). While the first one shows an incorrect calculation of gain or how to measure it in a physical circuit, the second two are both true statements that are accurate but incomplete pictures of the circuit. A full list of reasons given by at least 10% of students on three different op amp debugging assignments in Fall 2020 are shown in Table 6.3.

Top Reasons Why Circuit Working in Order of Prevalence (all cases >10%)		
Inverting amplifier (input terminals switched)	Non-inverting amplifier (feedback wire loose)	Non-inverting amplifier (resistors switched)
16 students said working	27 students	34 students
Mistook output voltage as gain (31%) Checked wiring (25%) Amplifies (19%) Voltages as expected (13%) Incorrect gain (13%)	Amplifies (38%) Checked wiring (29%)	Checked wiring (44%) Amplifies (33%) Voltages as expected (15%) Golden Rules (V+=V- holds true) (11%) Incorrect gain (resistors reversed) (11%)

Table 6.3: Aggregate data from three terms showing the top reasons given for the incorrect conclusion that the circuit was working.

There were 16 students who said the circuit was not working on the inverting amplifier with the input terminals switched, 27 students who said the circuit was not working on the circuit with the loose wire in the feedback loop, and 34 students who said the circuit was not working on the circuit with the resistors switched. Therefore, we cannot read too much into any one assignment, but from the aggregate, we see that two reasons are commonly shared across all assignments–namely, that the circuit amplifies and that the wiring is correct.

Based on these results, we realize the importance of emphasizing debugging as discrepancy analysis when teaching students debugging in introductory circuits. Debugging is warranted whenever a SINGLE discrepancy is found, and no discrepancies between data and theory should be ignored without further investigation. Furthermore, we must emphasize that checking the entire wiring of the circuit is not enough. The increase in number of students incorrectly diagnosing circuits when the circuit fault is not a visible miswiring, the lack of sufficient measurements taken by many students, and the reasoning given by students who say the circuit is working, all indicate that checking wiring appears to be the primary method of debugging a circuit for novice students. This makes sense, since in our experience, the primary mistake made by novice students in lab is miswiring.

Our goal, therefore, should be to help students move toward more advanced forms of debugging. We must teach students the relevance and importance of taking measurements to isolate circuit faults and that this method can be more efficient than checking all the wiring to find a circuit fault. Even if they do take measurements, as the results discussed above show, students are good at interpreting data that fits a picture they want and ignoring information that does not match the story they are building in their head (confirmation bias). We will further discuss data interpretation in the section below, where we show that teaching students to take measurements in itself is not enough, since students also need to correctly interpret the data.

6.3 Interventions to Improve Isolating the Cause

Since students struggled so much with isolating the cause of the circuit fault, we wanted to find ways to improve their performance on this step. We tried three different interventions, but only the third one appears to have had any effect on how effectively students were able to isolate the cause of the circuit fault. The first intervention was to include immediate feedback at checkpoints that let students know when their answer was incorrect. While this kept students on track throughout the assignment, it did not ultimately help their ability to isolate the cause on the first two tries. The second intervention we attempted was giving students reflective guidance while they were making measurements. We thought this might help students interpret the data and notice anomalies better, but we saw no change in performance. Finally, we gave students an additional optional checkpoint where they could mark subparts of the circuit as working or not working and get immediate feedback. This final intervention appears to have played a role in helping students successfully isolate the cause, with 20% more students identifying the correct cause/fix on the first try.

6.3.1 Effects of Immediate Feedback

Because we believed that students would be able to fix their own mistakes when told their answer is incorrect, we created immediate, computerized feedback which told the students when their answer was incorrect and told them to try again. We have data for three different activities run on our own interactive DMM website in winter and spring terms 2021. Two of the activities were the op amp with the loose feedback, and the other activity was a LED circuit with a loose resistor.

Most students got checkpoints right on the first try or were able to self-correct after a few tries, and the results are shown in Figure 6.9. We broke the students into 3 categories: correct first try, self-corrected, or struggling. Self-corrected refers to the fact that our immediate feedback of right or wrong led the student to reflect on their mistakes and find the correct answer on their second to fourth try. Struggling refers to anyone who took 5 or more tries and includes anyone who never got the correct answer. Students were given 7 tries on expectation, 5 tries on gain, 5 tries on probe placement, 4 tries at diagnosis, and 15 tries at isolate the cause. After that, students were given an explanation and the answer and moved onto the next step.

From these graphs, we can see that 20-30% of students on average fall into the self-corrected category at each checkpoint, meaning we successfully helped these students stay on track. We also can see that at three of the checkpoints–gain, probe placement, and isolate the cause–at least 10% of students on average were tagged as struggling, meaning that the immediate feedback intervention did not help them find the correct answer. The checkpoint *isolate the cause* is particularly notable, because half the students needed five or more guesses to figure out what was wrong with the op amp circuit.

We then compared the op amp circuit in winter term with the previous term (Fall 20), when students were not given the checkpoints and were not required to keep trying until they found the correct diagnosis for the circuit. Before the checkpoints, 20% of students correctly diagnosed the loose wire in the feedback loop; after the checkpoints only 22% correctly diagnosed the loose wire. This difference is not remotely statistically significant with a sample size of 100 each term. Therefore, while the checkpoints appeared to help students stay on track throughout the debugging process, it was not enough to help them in the final and most difficult step of debugging, which appears to be isolating the cause. Therefore, we had to look into other ways to help students.

6.3.2 Effects of Reflective Guidance

We speculated that if we gave students reflective guidance around interpreting data, it might help them better isolate the cause of the circuit fault. Therefore, we created an experiment where half the group was given reflective guidance when taking measurements and other other half was not. We hypothesized that the students receiving reflective guidance about what to think about when taking measurements would more likely isolate what was wrong with the circuit than those who did not receive the reflective guidance.

We structured the guidance in the following way. We used the simulator 1.0 where students could ask for nodal voltage measurements at lettered locations. Students were required to enter an answer about what they expected the measurement to be before they received the measurement they wanted. Then, after they received the measurement, they were required to answer an open-ended question about what they had learned from the measurement. While we saw no statistically significant difference in performance between the intervention and control groups based on correctness of their

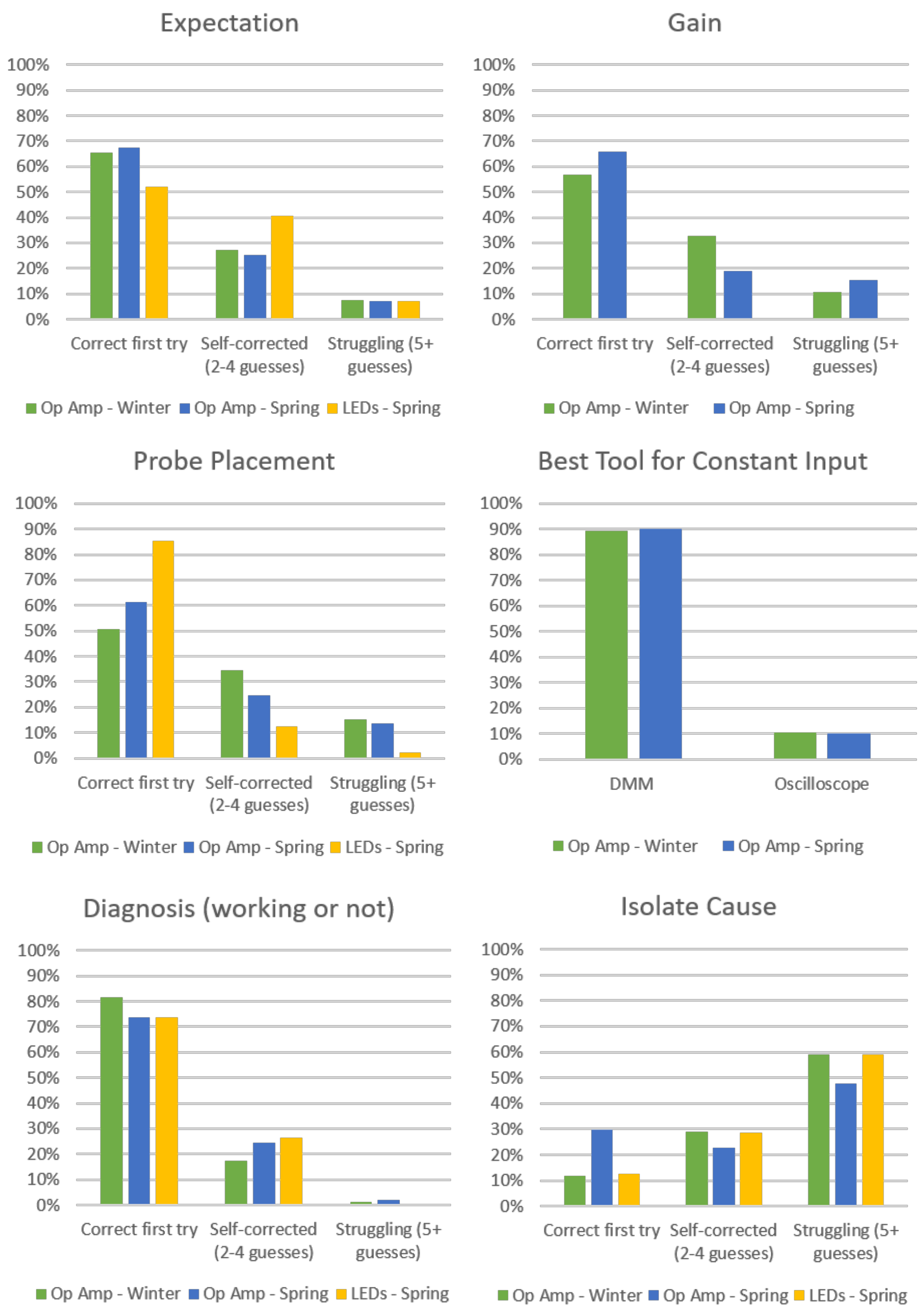


Figure 6.9: Shows how many students got each checkpoint correct on the first try, how many self-corrected after 2-4 tries, and how many took more than 5 tries to answer the checkpoint. The different color bars represent different terms. Green and blue represent the op amp circuit with the loose feedback wire, given in winter and spring 2021 terms, respectively. Yellow represents the LED circuit with loose component given in spring 2021. Checkpoint graphs from top left to bottom right: Expectation, Gain, Probe Placement, Best Measurement Tool, Diagnosis (working or not), and Isolate Cause.

final answers, we did notice a statistically significant difference in measurement strategy. Namely, the intervention group on average took two measurements less than their counterparts in the control group, or, in other words, they took 27% fewer measurements on average (7.3 vs. 5.5). An unpaired t-test confirms the significance of this difference with a p-value of p=0.03.

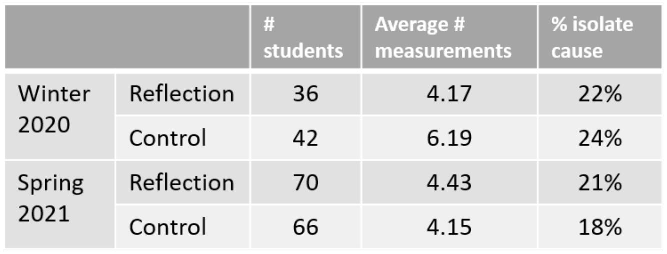
While these results may indicate that students in the reflective guidance group were better at prioritizing their measurements, they could also simply be a reflection of the cost of the measurement. Instead of just quickly clicking and getting measurements, students in the reflective guidance group had to spend time thinking about and answering two extra questions each time, which was a potentially significant mental cost. One plausible alternative hypothesis is that the students were better prioritizing their measurements simply because each measurement now had a higher cost associated with it.

Therefore, we continued this experiment in a later term, both to gather more numbers and to resolve this unanswered question about whether it was the reflective guidance or the cost of each measurement that changed measurement behavior. To better differentiate between the two hypotheses, we made both the control group and intervention group answer a question after each measurement. The control group had to answer open-ended questions related to op amps, but not necessarily relevant to the debugging task at hand, and the intervention group answered the same set of questions about expectation and what they learned before and after each measurement. This meant that the only difference between the control and intervention groups was whether the students were doing relevant cognitive reflection or not.

After creating a cost to each measurement, both control and intervention groups on average took the same number of measurements (see Figure 6.10). The control group was now taking fewer measurements, similar to the reflection group in both terms. Furthermore, there was still no difference in performance between the intervention and control groups on isolating the cause. Therefore, we concluded that simply providing reflection questions without feedback is not enough to help students better interpret the measurement data and isolate the cause. Consequently, our next attempted intervention included real-time, personalized feedback around isolating the part of the circuit where the problem was.

6.3.3 Breaking Circuit into Subparts

We have discussed several unsuccessful attempts to help students better isolate the cause of the circuit fault. To try to address this gap, we dug deeper into the measurement data to identify indicators of why some students quickly isolated the cause of the circuit fault and others did not. We noticed that for the op amp circuit with the loose wire in the feedback loop, a large number of students were never even looking at the feedback loop when taking measurements (see Section 5.6.1). Therefore, we decided to create a checkpoint that broke the circuit into subsections and



		# students	Average # measurements	% isolate cause
Winter 2020	Reflection	36	4.17	22%
	Control	42	6.19	24%
Spring 2021	Reflection	70	4.43	21%
	Control	66	4.15	18%

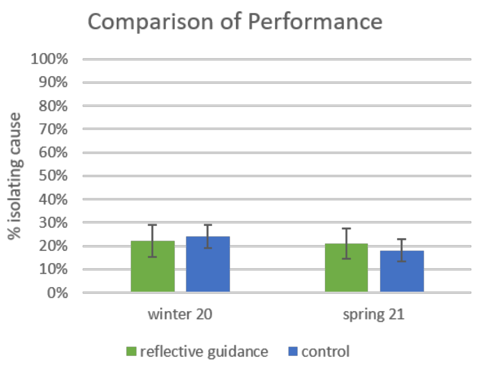


Figure 6.10: Effects of guided reflection on performance. **Left:** Table compares reflection and control groups on three parameters–number of students, average number of measurements, and percentage of students who isolated the cause. **Right:** Graph shows percentage of students who successfully isolated the cause by term and by experimental group (reflection or control group). No difference was found in performance on isolating the cause.

required students to check whether each subsection was working as expected (see Figure B.27). We believed that by requiring students to give responses about different regions of the circuit, we would get them to take measurements in all the regions, including, most importantly, the feedback loop where the issue was.

While we believed simply getting students to measure the feedback loop was key, we also wanted to ensure that they interpreted the data correctly. Therefore, we also gave them personalized feedback based on their responses at this checkpoint (see Appendix B.9 for the full list of feedback). The feedback was designed to help them figure out what they should have expected for a measurement at that location and challenge them to take a measurement to compare with that expectation.

This extra checkpoint was integrated into the op amp debugging activity with a loose wire in the feedback loop in spring 2021. The checkpoint was made optional, and about 2/3 of the students chose to use the extra guidance step for isolating the cause. As a result, 20% more of the class measured the feedback loop, and 20% more of the class correctly isolated the cause of the circuit fault on the first try (see Figures 6.11 and 6.12). We also saw a 15% decrease in the number of students who never figured out what was wrong with the circuit. According to a Fisher's Exact Test, the groups of students isolating the cause correctly on the first try, after 2-4 guesses, and after 5+ guesses are statistically different between winter and spring terms ($p\text{-value} < 0.01$). Therefore, we attribute this change to the one modification we made to the assignment, namely, the addition of this checkpoint that isolated parts of the circuit.

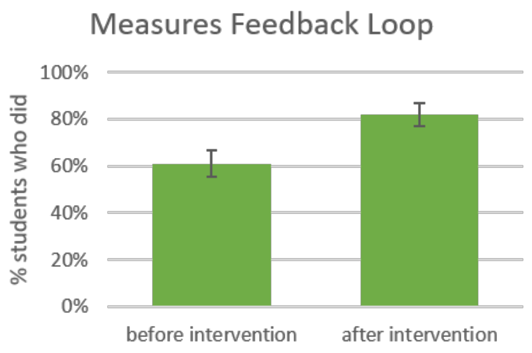


Figure 6.11: Percentage of students measuring feedback loop before and after intervention.

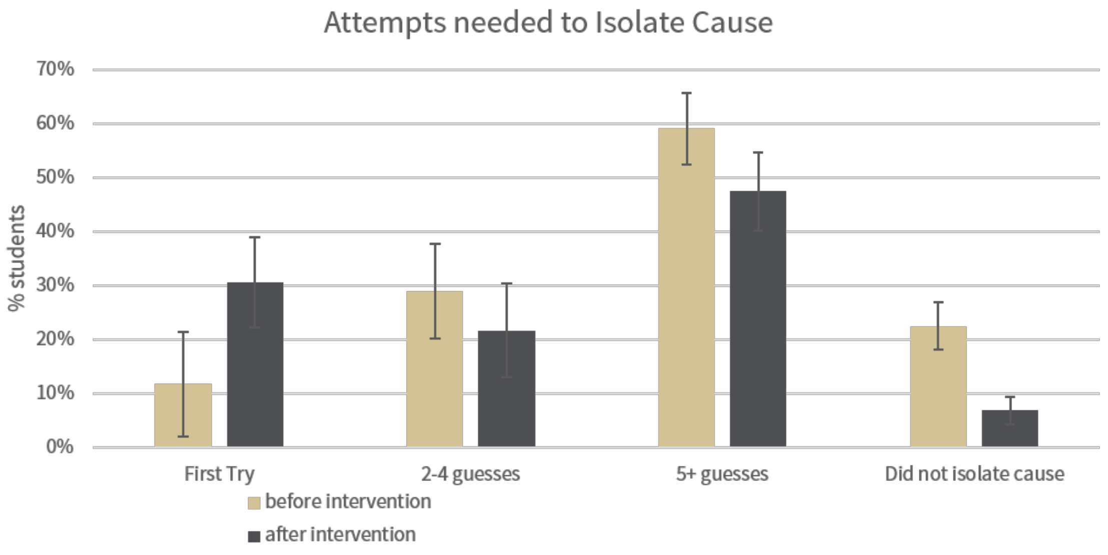


Figure 6.12: Histogram comparing number of attempts needed to isolate cause for students before and after the intervention.

Limitations of study

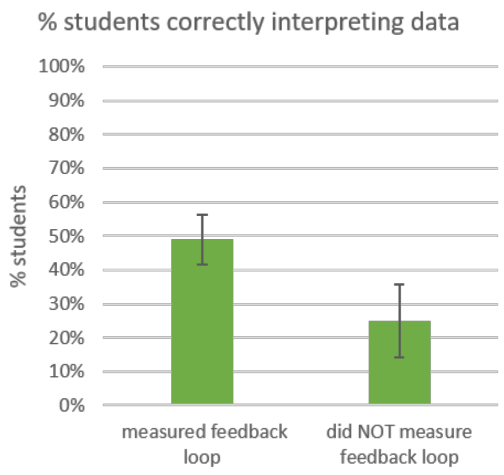
It is difficult to tease apart how much of this improvement was caused by forcing students to measure the feedback loop in the first place and how much was caused by the personalized feedback that corrected their interpretations of the data. We know that 20% more students measured the feedback loop, but not all students who measured the feedback loop correctly interpreted the results.

When we look at the two-thirds of students who used the extra checkpoint in spring 2021, we find that 25% of them never measured the feedback loop. Of those who did measure the feedback loop, we find that 49% of them correctly indicated that the feedback loop was not working, which indicates they were correctly interpreting measurements (see Figure 6.13 a). Not surprisingly, when we look at students who did not measure the feedback loop, we see they have half the chance of correctly noticing the feedback loop was not working compared to their counterparts who measured the feedback loop.

Finally, we looked at how many students who initially said the feedback loop was working correctly changed their interpretations after being given the personalized feedback. For this analysis, we only looked at the students who had actually measured the feedback loop. It turns out that 67% of students correctly updated their interpretations based on the feedback (see Figure 6.13 b). Only a single student incorrectly switched their answer from the feedback loop not working to the feedback loop working. The 67% of students who correctly updated their interpretations based on feedback represents 16% of the entire class.

Comparing this result with the previously presented results, we see that 20% more students measured the feedback loop and 16% more students correctly interpreted data based on the computerized feedback. Together, these interventions allowed 20% more students to correctly isolate the circuit fault on the first try.

We also looked at a debugging activity students did earlier in spring 2021 to determine whether simply knowing where to measure in the circuit was enough for a correct response or whether the issue lies with students' interpretation of data. In this other assignment, students were debugging an LED circuit where there were a number of LEDs (with current-limiting resistors) in parallel. The blue LED was not turning on because the resistor in series with it was not firmly pushed into the breadboard. Like the op amp with a loose feedback wire, this circuit had a loose wire, but unlike the op amp circuit, it also had a clear indication of which subpart of the circuit has the problem–namely, the row with the red LED and its current-limiting resistor. Despite this clear indicator of what subpart of the circuit had the problem, we see no difference in performance on isolating the cause than on the op amp debugging problem with the loose feedback wire (see the lower right image of Figure 6.9). This would indicate that students' data interpretation is the bigger problem.



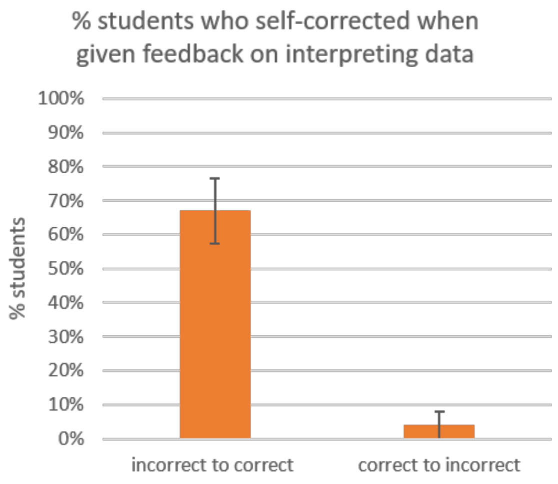


Figure 6.13: **a)** (left) Percentage of students correctly interpreting data on the first try. Students who actually measured the feedback loop were twice as likely to have correct interpretations of the data. **b** (right) Percentage of students who changed their answer after getting feedback on interpreting data (only looking at students who measured the feedback loop). Most students correctly updated their interpretations, though one student did change from correct to incorrect.

The similarities of the LED problem to the op amp problem also pose a possible third reason for better performance on isolating the cause in spring 2021. It is possible that the students remembered the loose component from the earlier assignment and carried this idea with them, making them more likely to subsequently diagnose a loose wire. Further studies will have to be carried out to differentiate these various aspects.

In sum, our results show that students need help knowing where to measure in the circuit when the problem is not a visible circuit fault. They also need help interpreting data when they take relevant measurements, including creating expectations for sub-parts of the circuit. When we take these findings into account and provide scaffolding for students around where to measure and how to interpret that data, we can help students more quickly isolate the cause of the circuit fault. Finally,our results may also indicate that previous exposure to types of faults with a circuit may contribute to improved performance on isolating the cause.

6.4 Summary

In this chapter, we have shown that the more domains of knowledge at which students excel, the more likely they are to successfully debug a problem. We have also shown that the vast majority of students struggle with circuit debugging when the circuit fault is not a visible miswiring. Students whose first measurements match one of the common starting strategies are more likely to successfully debug the circuit than a counterpart taking more random measurements. We have also showed that students who respond that a circuit is working when it is not tend to view checking the wiring as sufficient and tend to ignore discrepancies. Finally, we showed that giving immediate, but non-specific, feedback helps students at specific checkpoints. Generally 20-30% of students were able to self-correct when told their answers were wrong at checkpoints and get back on track. For the hardest step, isolating the cause, we found that non-specific feedback was not enough to help the average student. Therefore, we created an additional intervention that gave students personalized feedback on where to measure and how to interpret data, which improved their performance. These findings are significant for informing more effective pedagogical activities for teaching circuit debugging.

Chapter 7

Conclusions

This thesis described the creation of an increasingly realistic debugging simulator and the debut of a number of debugging activities on the platform in an introductory circuits course, which were well-received by students. We also showed preliminary results about students' measurement patterns and what subparts of debugging students struggle with the most. We hope these results make clear the value of such debugging simulations in a classroom and encourages others to actively include debugging in their curriculum.

We drew inspiration from previous work that breaks down the complex task of problem-solving into cognitive substeps and taught a version of these steps in our curriculum to our TAs and students. In short, the iterative process of debugging starts with defining the problem and resolves by comparing models and experimental results and brainstorming possible explanations for the discrepancies. Based on previous research, we also see debugging inherently as a decision-making process, meaning that each step of the debugging process can be viewed as a decision one must make. Debugging also involves many types of knowledge, including theoretical knowledge, physical knowledge, and measurement knowledge. All of these different lenses for analyzing problem-solving were helpful in our efforts to understand novice debuggers and influenced the ultimate design of our debugging simulator.

While other researchers have studied students troubleshooting circuits in lab, our research differed in several ways. First of all, our computerized debugging simulator logged backend data about all actions students took, allowing us to capture comprehensive data for hundreds of students at a time. Our simulator created an authentic virtual debugging experience where students placed DMM probes on a virtual breadboard to take measurements. Students were free to take as many measurements as they wanted, wherever they wanted, which prompted us to call the debugging scenarios choose-your-own-adventure activities. We also created an iterative system where students had to keep testing until they figured out how to fix the circuit. Not only did this better mimic the real debugging process, but it also helped keep struggling students on track and allowed us to collect more data from struggling students.

We found that checkpoints were useful. They both acted as guardrails to keep struggling students on track and gave us better insight on which aspect of the exercise were most difficult. Since the assignment was iterative, students had to correctly answer each checkpoint before moving onto the next one. This gave us information about what knowledge areas students were struggling with before the debugging activity culminated in the final question requiring students to isolate the circuit fault. The iterative nature of the simulator also changed the way we measured performance. A greater emphasis was placed on journey instead of destination, since all students got to the correct answer in the end.

Tracking the journey provided a rich dataset of student responses to our debugging activities, which allowed us to better characterize novice debuggers and determine a number of indicators related to successfully debugging the circuit, including measurement patterns and checkpoint performance. We saw that gaps in theoretical, physical, and measurement knowledge adversely contributed to the debugging process. We also found that most novice debuggers did not have the measurement and data interpretation skills necessary to debug issues with a circuit that were not a visible miswiring. Not only did many students lack knowledge on how to place the red and black probes of a DMM for a basic measurement, but we also found that many students, when left to their own devices to take measurements, failed to take sufficient measurements in relevant places. This result led us to increase scaffolding in our debugging activities, including questions that prompted students to measure more parts of the circuit and personalized feedback to help students with data interpretation.

In the end, we created 10 different debugging scenarios. Four scenarios involved amplifiers, two involved LEDs, one involved transistors, one involved taking basic readings with a DMM, one involved LED displays, and one involved the circuitry in a useless box. While not every scenario was used every term, five different terms of students from the introductory circuits class at Stanford University interacted with the debugging simulator. The final two terms of students completed an entire debugging curriculum including two activities with the interactive DMM on our own platform and four text-based adventures in Qualtrics.

Student feedback about the debugging activities was overwhelming positive. Students said the activities were fun, intuitive to use, and helped them learn something about debugging. Their open-ended responses helped us find bugs in our system and suggested some practical fixes to make the user-interface (UI) more user friendly.

7.1 Next Steps

Creating the debugging simulator and developing the debugging activities required significant effort and yielded a very useful debugging training for students in the introductory circuits course. The debugging simulator and activities also opened up a number of interesting new research opportunities. We have some preliminary evidence that the full sequence of debugging activities changed the way students approached taking measurements, and we would like to further investigate whether the guardrails used to help keep students on track also helped them learn a better approach to debugging. To test whether the guardrails improved students' approaches to debugging, we would like to remove the scaffolding from the activities at the end of term, to evaluate which techniques students internalized.

We would also like to collect data for a broader range of students and ability levels. This would allow us to work toward validating our debugging activities. By collecting data for more advanced debuggers, we can look for characteristics that distinguish an advanced debugger from a novice debugger. We also want to test the debugging activities on a wider variety of students including students of different demographics and ability levels. Apple is working with Stanford to share the material taught in Engr40M with many historically black colleges and universities (HBCUs). This program provides an exciting opportunity to collect data from a wider range of students.

Finally, we would like to use machine learning to better study the complex measurement patterns of students. Because the search space was so large, we had a hard time identifying patterns in the data using traditional means. Atindra Jha, the student who designed the code for the interactive DMM, plans to apply machine learning to the measurement data in the coming year to find better ways to characterize and group students. Taken together, these additional research opportunities will allow us to better assess the effectiveness of debugging activities and complex data they provide.

We also believe that the principles outlined in this thesis that guided the creation of these circuit debugging activities can apply to other classes within electrical engineering or even classes in other engineering fields. Taking examples from optics, one could create a debugging simulator for an interferometer or a 4f lens system. Instead of asking students to take voltage measurements, students could ask to the look at the beam profile at various locations within the experimental setup. Similar to the circuit debugging activities, emphasis would be placed around comparing theory and the physical world and looking for discrepancies. The activity could also be broken into similar checkpoints covering the various types of knowledge. For knowledge questions, we could ask students what they expected at the output. For physical knowledge, we could ask students to match locations on the ray diagram to physical locations in the setup. Or for measurement knowledge, we could ask whether a photodiode, piece of paper, or beam profiler is best for looking at the output. We suspect that even in this very different application, we would find similar difficulties for students in isolating the cause of the problem.

This thesis holds a wealth of knowledge about how to create and successfully execute circuit debugging activities in a classroom. We shared the complications that we ran into when creating the activities and when analyzing the data in hopes that it will assist future instructors and researchers who make or use similar materials. We also included the full list of questions and answers for each debugging activity in the appendix, so that any of our activities can be fully reproduced. Our dearest hope is that this publication along with others will cause a revolution in how circuit debugging is taught and assessed in introductory circuits courses.

Appendix A

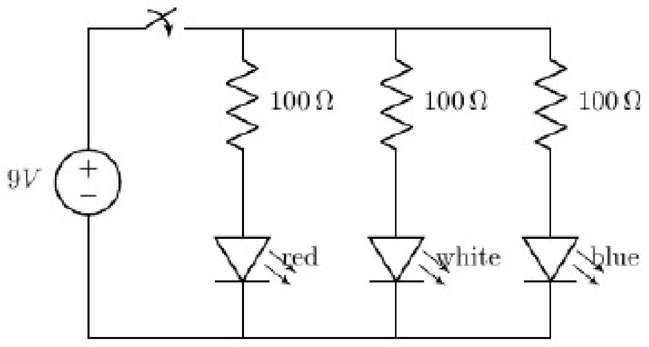
Studying Effects of Closed-Form Answers on Brainstorming

As we moved our debugging problems to an online platform, we also started moving away from open-ended answers to make it quicker and easier to analyze data for hundreds of students. While some questions were easily moved to multiple-choice format, we were concerned whether we could actually probe students' brainstorming of what could be wrong with the circuit with closed-form answers. If a list of possible ideas is given to the students, it is no longer an idea generation activity, but rather an idea analysis activity. We thought that we might be making the question easier by giving students a number of ideas to sort versus making them generate the ideas themselves.

Therefore, we designed the following experiment to test whether students were affected by the answers being listed instead of having to generate the ideas themselves. Half the class received debugging problems where they had to brainstorm their own ideas about what might be wrong with the circuit based on a list of symptoms. The other half of the class received the same circuit to debug, but instead of brainstorming their own ideas, they were asked rank a list of possible causes (see Figure A.1).

We expected that there would be a group of struggling students who would not be able to come up with ideas on their own, but, given a list of ideas, they could analyze them and come up with possible answers. Therefore, we expected the ranking group to perform better than the open-ended group on average. Contrary to our expectations, it turns out both groups performed the same on both assignments, indicating the ranking problems were not actually making the problem easier for students. Details of the experiment and results are shared below.

You are designing a multi-colored light strip. The circuit is shown below.



The white and blue LEDs turn on, but the red one does not.

Think about the reasons listed below and whether they might be the cause of the problem with the circuit. Sort the possible causes into possible or not possible and rank the possible causes in the order you think they are most likely (1 being most likely).

Items

battery dead

battery backward

battery shorted

LED flipped

LED broken

resistor backward

resistor broken

wrong resistance value

miswired-- connections between individual components incorrect

breadboard broken

bad circuit design

Possible cause

Not the cause

Figure A.1: Problem statement for brainstorming experiment. The intervention group received the drag and drop question shown and the control group had to answer an open-ended version of the question.

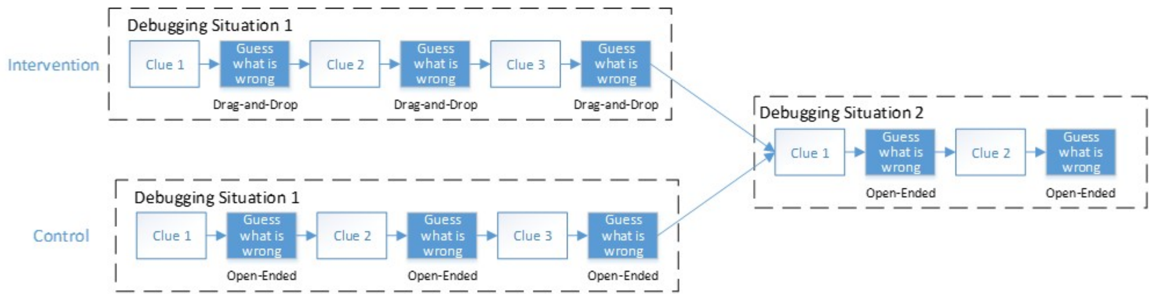


Figure A.2: **Experimental flow for the brainstorming experiment.** After each clue, students are asked what they think is wrong with the circuit. Half the class was given open-ended questions (control) and half were given closed-form drag-and-drop questions (intervention). For the second activity both groups were given open-ended questions.

A.1 Experimental Design

We conducted this experiment twice in Fall 2020. Students were given two debugging problems a week over the course of two weeks on their homework. The assignments were available online and built on the Qualtrics platform. Students were told they would get participation points for completion, not correctness. We also reminded them that debugging is like a mystery and encouraged them not to talk with their classmates until after they had completed the problem.

Students were given two or three symptoms (or “clues”) for each debugging situation. After each clue, they were asked to give possible explanations for what was wrong with the circuit. By asking the same question multiple times, we were able to see how students incorporated each new piece of information into their model of what was wrong with the circuit. The general flow of the experiment can be found in Figure A.2.

The first debugging problem on each homework was open-ended for half the class and a ranking question for half the class, but everyone received the same open-ended question for the second debugging problem that week. We had everyone do open-ended responses on the second problem each week to see if seeing the list of possible causes on the previous problem for the ranking group helped them brainstorm on the subsequent open-ended problem. Students were randomly sorted into the open-ended or ranking group each week.

The open-ended group was always encouraged to list multiple ideas if possible. The ranking group was given a question format called drag-and-drop in Qualtrics, which allowed us to give the students a list of possible causes, which they could then drag-and-drop into different category boxes on the right (see Fig. A.1). We gave them two possible categories: possible and not possible. We told them to rank items in the possible category in order they thought them most likely to to be the cause. Students could also choose to leave the item in the original list if they were not sure what category it belonged in.

In the first homework, both debugging problems involved issues with the DMM. In the first situation, students were told they were trying to measure resistance of a resistor, but they kept getting a 1 on the DMM. After brainstorming what might be wrong, they were told that when they touched the two probes together, the DMM measured 0. Many students blow fuses when they are first learning to use their DMM, so we assumed many students would suggest this after the first system. This second clue was supposed to indicate that it was not a blown fuse as they suspected. There were then asked, “Given this new measurement, what do you think is wrong with the circuit?” The final clue they were given is a picture of the DMM and a close-up of the banana clips attached to the resistor. While the color code of the resistor was visible, we did not give them a numerical value for the resistor in order to not unfairly draw attention to the resistance. Students now had enough information to solve the issue, which was the resistor was a $220\,\mathrm{k}\Omega$ resistor, but the DMM was set for a $2\,\mathrm{k}\Omega$ resistance reading.

The second problem emulated a measurement of leakage current of a solar cell that students had to perform during their first lab. We noticed that many students in lab forgot to switch their probes back from the 10A port to the standard port when they went to measuring smaller currents again, so we decided to simulate that situation. The first clue was that the DMM always read 0 whenever they tried to measure the leakage current. The second clue was a picture of the experimental setup where it could be seen that the DMM was set to the 2mA current reading but the red probe was in the 10A port.

The next week, students were given two debugging problems about LEDs. In the first problem, students were given a circuit with a three different colored LEDs, one of which was not lighting up (see Fig. A.1). The students were given a circuit schematic to start so that they could eliminate whether it was a bad circuit design preventing the circuit from lighting. For example, if each parallel LED had not had its own current-limiting resistor, but instead the LEDs all shared one resistor, then some of the LEDs would not have been on. In this specific debugging activity, the students should have found that all the LEDs should be on in theory.

We assumed most students would first guess that the unlit LED was backwards. Then students were given a second clue that when they switched one of the working LEDs with the nonworking one, the nonworking one now worked and the previously working LED no longer lit. This allowed students to eliminate that the LED was broken or backwards. Finally, students were shown a picture of the physical setup, where they could notice that one of the resistors has a different color code than the other two. The intended problem with the circuit was that a $1\,\mathrm{M}\Omega$ resistor was mistakenly put in for one of the $100\,\Omega$ resistors.

The second problem involved a bad circuit design that never should have lit up. Students were given both the theoretical schematic and a physical picture of the circuit all at once, and no further clues were given.

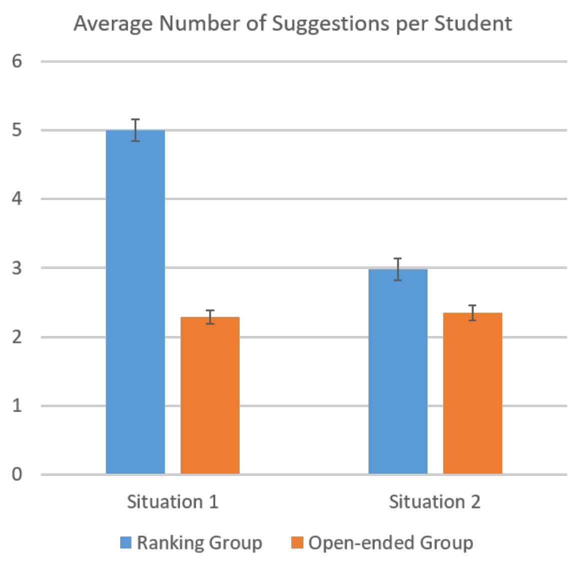


Figure A.3: Average number of suggestions per student. Students who received the open-ended version of the question suggested fewer possibilities of what could be wrong with the circuit than their counterparts who chose from a drag-and-drop list.

A.2 Results + Analysis

There were 110 students in the class in Fall 2020, and about 100 students completed the debugging problem on each homework, leaving about 50 students in each group–the ranking group and the open-ended group–on each assignment. Students were sorted randomly each week into the ranking group or open-ended group.

Open-ended answers were read and categorized based on the list of choices presented to the ranking group. Any answers that did not match anything in the ranking group's list were categorized as "other." For the ranking group, anything a student put in the "possible list" was considered a suggestion, which was then compared against the list of ideas from the open-ended group.

Not surprisingly, the ranking group suggested 2.75 more ideas on average during the first debugging clue than the open-ended group (see Figure A.3). This makes sense because they simply had to drag and drop ideas from a list, so it took less effort to generate ideas. With a t-test, this represents a p-value of less than 0.0001. Even on the second problem each week, where both groups got open-ended questions, the group that had a ranking question for the previous problem produced 0.64 more ideas on average than their counterparts in the open-ended group, which is statistically significant via t-test (p-value 0.0013).

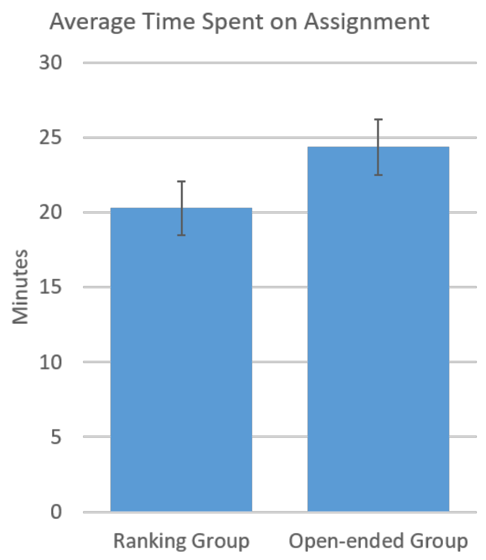


Figure A.4: Average time spent on assignment increased when students had to generate their own answers in the open-ended version.

Next, we looked at time spent on the debugging activity, because we expected it would take students in the open-ended group longer, since they had to generate their own ideas. Students in the open-ended group spent over four minutes longer on average than their counterparts on the assignment each week (see Figure A.4). Considering students spent 22 minutes on average on the entire assignment, including the second part where everyone had to answer the open-ended version, this 20% change appears large. But considering the standard deviation between students was about 18 minutes, it turns out the variation between students is the dominating factor over the small four minute difference in average time between students. But due to high variance, the change is not actually significant (p-value 0.11 via t-test).

We believe the large variance is due to the difficulties in assessing time spent on the assignment accurately. Qualtrics simply tells us when the students started and stopped the assignment, which means if a student stepped away for a break and came back later, their time would be artificially inflated. To account for this, for any outlier student who took over 60 minutes on the assignment, we assumed they had taken a break and capped their data point at 60 minutes for the t-test calculation.

What we were more interested in was whether the students in the ranking group performed better than the students in the open-ended group. Our expectation was that the ranking list would artificially help students generate ideas, so the ranking group would perform better. We measured this by comparing the percentage of students who suggested the correct issue with the circuit (see Figure A.5). To make this comparison, we had to make sure that there were enough clues to eliminate

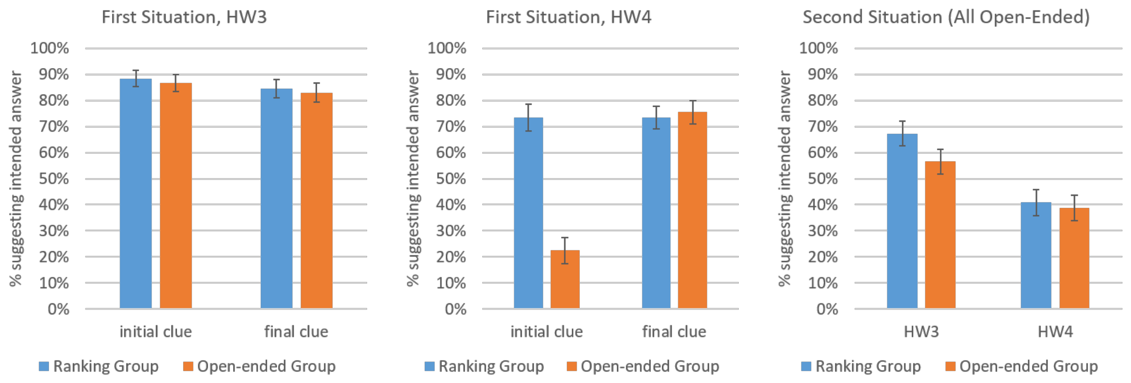


Figure A.5: Percentage of students suggesting the correct answer after the initial clue and after the final clue for three different problems. The group receiving open-ended questions is shown in orange and the group receiving drag-and-drop questions is shown in blue.

the other potential causes in the ranking list. We also noticed after the first homework, over 80% of the students suggested the correct problem immediately from the first clue. This made it harder to study idea generation, so we made sure in the second homework to choose an issue with the circuit that would not be the first guess of students.

In the second week assignment, we saw that only roughly a quarter of students in the open-ended group came up with the intended answer after the first clue, whereas almost three-quarters of the students in the ranking group recognized that might be the cause after the first clue (see Figure A.5). Therefore, we had successfully picked a problem with the circuit that students would not naturally and immediately guess based on the symptoms. We then had two further steps where we gave them more symptoms. After these additional clues, the three-quarters of the open-ended group was able to brainstorm the correct problem with the circuit, and they had effectively caught up to their peers in the ranking group.

The final question we looked into was whether the ideas from the ranking activity carried over to future assignments–namely the second situation on each homework. While the ranking group did perform better on average on both homework assignments for this second activity, it is not statistically significant. Therefore, based on hypothesis testing and p-values (p=0.3164 on HW3 by Fisher Exact test), we must reject the hypothesis that the ranking group performed better on the follow-up assignments.

A.3 Conclusions

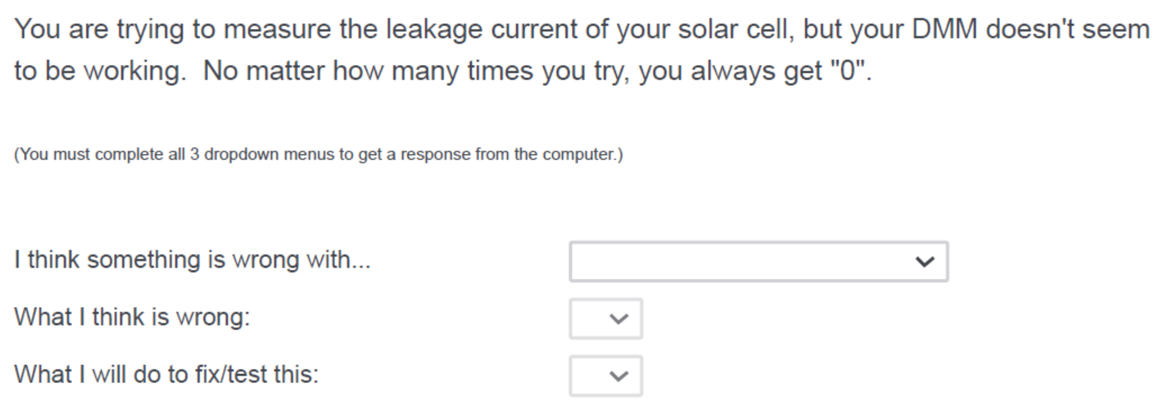
This analysis shows that while students suggest more ideas when using a ranking question versus an open-ended question for brainstorming possible causes, they do not get to the correct answer more frequently than their counterparts. This indicates that using closed-form questions like ranking questions in place of open-ended questions appears to have no significant effect on the final answers given by students. Therefore, we were able to move forward with confidence that our ranking questions were not changing student performance, at least not in the short term.

Appendix B

Full Questions from Debugging Activities

B.1 Basic measurements with DMM

The first debugging assignment in winter and spring 2021 was about a problem measuring leakage current of the solar cell due to the red probe being in the wrong port on the DMM. The questions are shown in chronological order below. We used a drill down question type in Qualtrics as the base question for the repeating structure. Each round students would say what they suspect was wrong with the circuit and choose a corresponding test/fix to perform on the circuit (see Figure B.1). The list of all possible combinations of what the student suspects is wrong with the circuit and the fixes/tests they can perform in each case in shown in table B.1. The computer would then return the result of the test or a picture of the setup or DMM (see Figure B.2). After receiving the info, the students were asked if they had enough information to fix the circuit or if they wanted to continue investigating. If students said they were done investigating, they were given the final diagnosis screen where they were asked for a final definitive answer of what was wrong with the circuit (see Figure B.3). If students choose the incorrect fix for the circuit, they were sent back to the original drill down question about what they suspect is wrong with the circuit and what fix/test they want to do. This cycle repeats until the student successfully chooses the correct fix for the debugging problem.



You are trying to measure the leakage current of your solar cell, but your DMM doesn't seem to be working. No matter how many times you try, you always get "0".

(You must complete all 3 dropdown menus to get a response from the computer.)

I think something is wrong with...

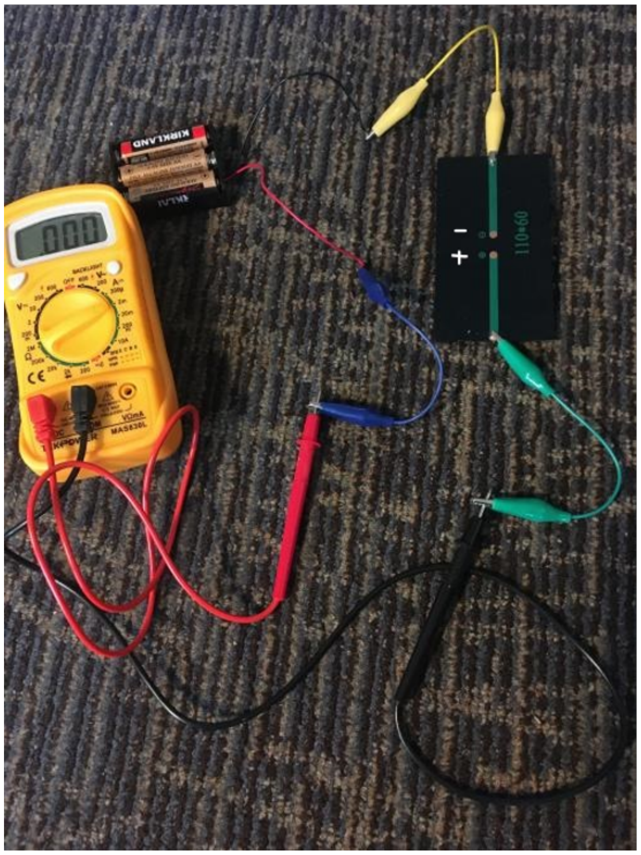
What I think is wrong:

What I will do to fix/test this:

Figure B.1: The repeating problem statement for the first debugging assignment. The question format is called drill down in Qualtrics.

I think something is wrong with:	What I think is wrong:	What I will do to fix/test this:
the battery	the battery pack is dead	Replace batteries.
the battery	the battery pack is dead	Measure voltage of battery pack.
the battery	the battery pack is dead	Measure current of battery pack.
the battery	the battery pack is backwards	Switch red and black wires of battery pack in circuit.
the solar cell	solar cell broken	Replace solar cell.
the solar cell	solar cell broken	Measure short circuit current again.
the solar cell	solar cell backwards	Replace solar cell.
the solar cell	solar cell backwards	Flip connections to solar cell.
the DMM	scale on DMM wrong	Check settings of DMM.
the DMM	scale on DMM wrong	Test DMM by measuring current through 5 kΩ resistor connected to battery pack.
the DMM	probes in wrong port of DMM	Check settings of DMM.
the DMM	probes in wrong port of DMM	Test DMM by measuring current through 5 kΩ resistor connected to battery pack.
the DMM	freeze button on DMM accidentally on	Check settings of DMM.
the DMM	freeze button on DMM accidentally on	Push freeze button.
the DMM	freeze button on DMM accidentally on	Test DMM by measuring current through 5 kΩ resistor connected to battery pack.
the DMM	the fuse is blown in DMM	Replace fuse.
the DMM	the fuse is blown in DMM	Check settings of DMM.
the DMM	the fuse in blown in DMM	Test DMM by measuring current through 5 kΩ resistor connected to battery pack.
the wiring	loose connections	Reconnect each wire to make sure there are no loose connections.
the wiring	loose connections	Use DMM in resistance mode to check connections.
the wiring	loose connections	Look more closely at experimental setup and wiring.
the wiring	miswired	Look more closely at experimental setup and wiring.
the wiring	miswired	Place DMM, solar cell, and battery pack in series not parallel.
the wiring	miswired	Switch the location of the battery and solar cell.
the wiring	miswired	Connect the DMM in parallel with the battery.
the wiring	miswired	Connect the DMM in parallel with the solar cell.
nothing is wrong	the leakage current should be 0	Nothing needs to be done.
nothing is wrong	the solar cell needs to be moved to the sun	Move solar cell into sun.
I have no idea what is wrong	I will investigate further	Measure voltage of battery pack.
I have no idea what is wrong	I will investigate further	Test DMM by measuring current through 5 kΩ resistor connected to battery pack.
I have no idea what is wrong	I will investigate further	Check settings of DMM.
I have no idea what is wrong	I will investigate further	Look at setup.
I have no idea what is wrong	I will investigate further	Use DMM in resistance mode to check connections.
something else	I will specify on the next page.	I will specify on the next page

Table B.1: All possible submenu combinations for the main repeating question in the first debugging assignment.



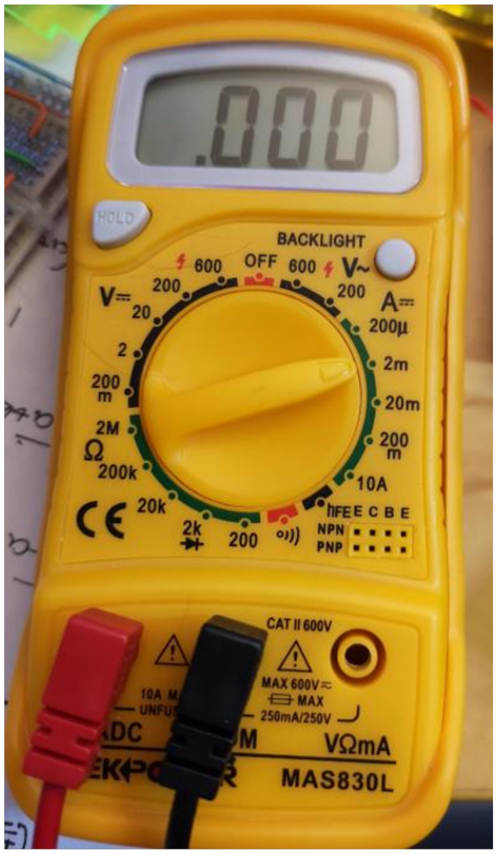
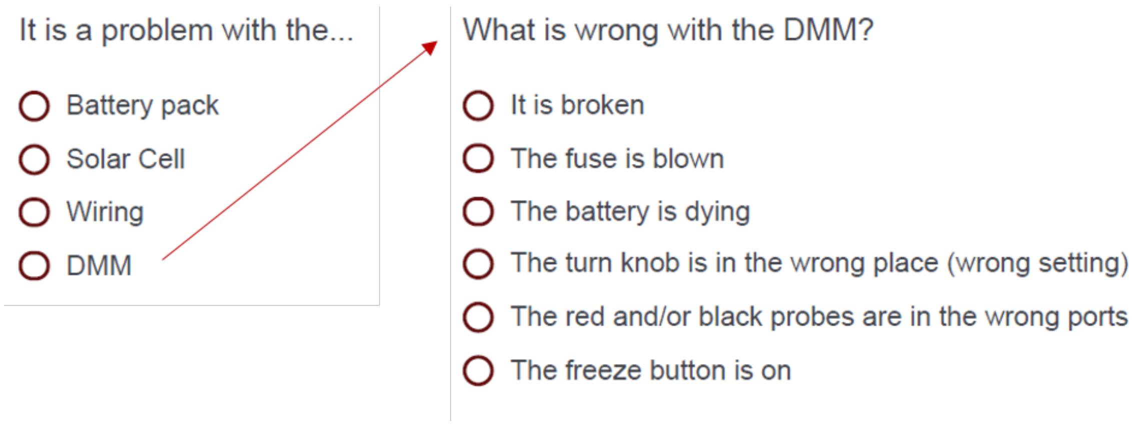


Figure B.2: **Left:** Picture of the experimental setup. **Right:** Close-up image of the DMM and its settings.



It is a problem with the...

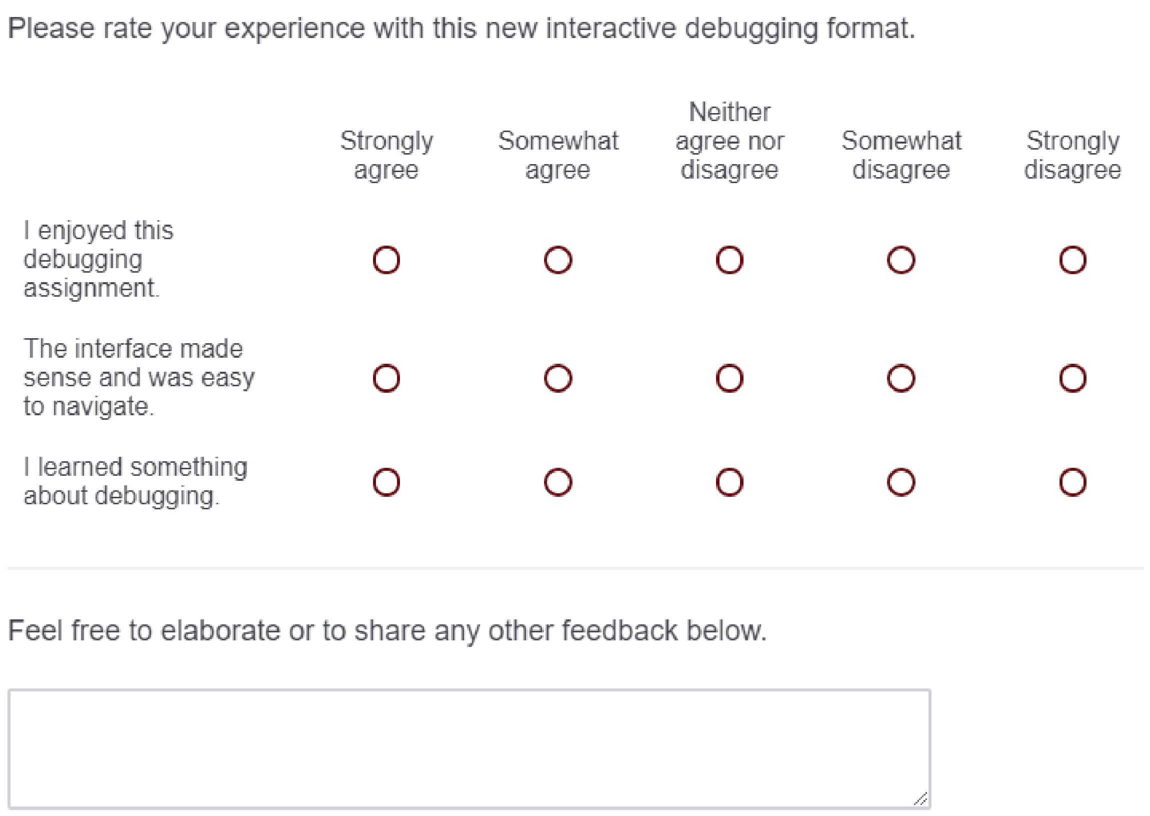
- ○ Battery pack
- ○ Solar Cell
- ○ Wiring
- ○ DMM

What is wrong with the DMM?

- ○ It is broken
- ○ The fuse is blown
- ○ The battery is dying
- ○ The turn knob is in the wrong place (wrong setting)
- ○ The red and/or black probes are in the wrong ports
- ○ The freeze button is on

Figure B.3: **The final diagnosis questions for the first debugging assignment.** If students correctly choose DMM on the first question, they were given a follow-up question where they had to pick the correct problem with the DMM from a list of six possibilities. If students answered either of these questions incorrectly they were sent back to the original drill down question to continue investigating.

Afterwards, students were asked for their feedback. They rated on a 5 point scale how much they enjoyed the debugging activity, how easy the interface was to navigate, and whether they learned something about debugging (see Figure B.4).



Please rate your experience with this new interactive debugging format.

	Strongly agree	Somewhat agree	Neither agree nor disagree	Somewhat disagree	Strongly disagree
I enjoyed this debugging assignment.	○	○	○	○	○
The interface made sense and was easy to navigate.	○	○	○	○	○
I learned something about debugging.	○	○	○	○	○

Feel free to elaborate or to share any other feedback below.

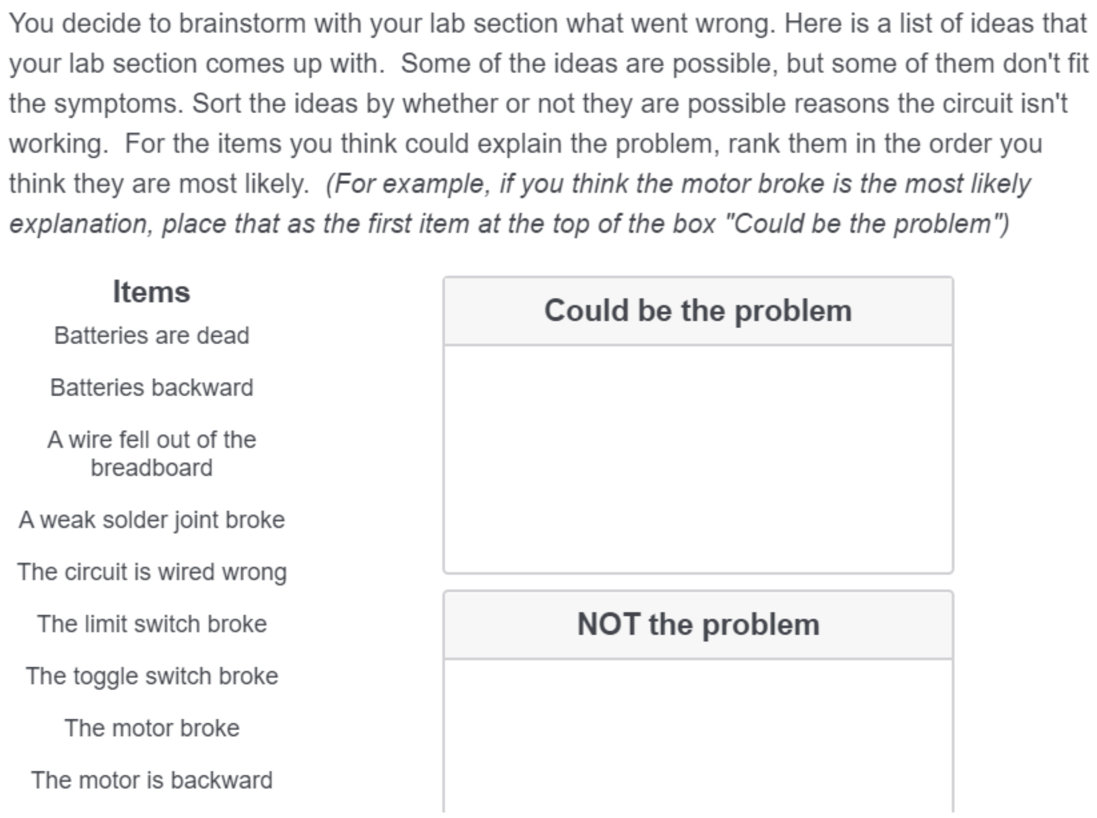
Figure B.4: Feedback Questions for the text-based Qualtrics debugging activities.

B.2 Miswired useless box

Students were given the following initial problem description:

> *During lab, you got your basic useless box working. It works perfectly. You flip the switch, the finger comes out and turns off the switch, and then the finger goes back into the box and stops against the limit switch. You leave the box sitting on your desk. Several days later you go to show it to your friend, but when you flip the switch, nothing happens.*

Next students were asked to brainstorm what they suspect is wrong with the circuit based on the problem description. The brainstorming question was made using the drag-and-drop question type in Qualtrics and is shown in Figure B.5.


You decide to brainstorm with your lab section what went wrong. Here is a list of ideas that your lab section comes up with. Some of the ideas are possible, but some of them don't fit the symptoms. Sort the ideas by whether or not they are possible reasons the circuit isn't working. For the items you think could explain the problem, rank them in the order you think they are most likely. *(For example, if you think the motor broke is the most likely explanation, place that as the first item at the top of the box "Could be the problem")*

Items

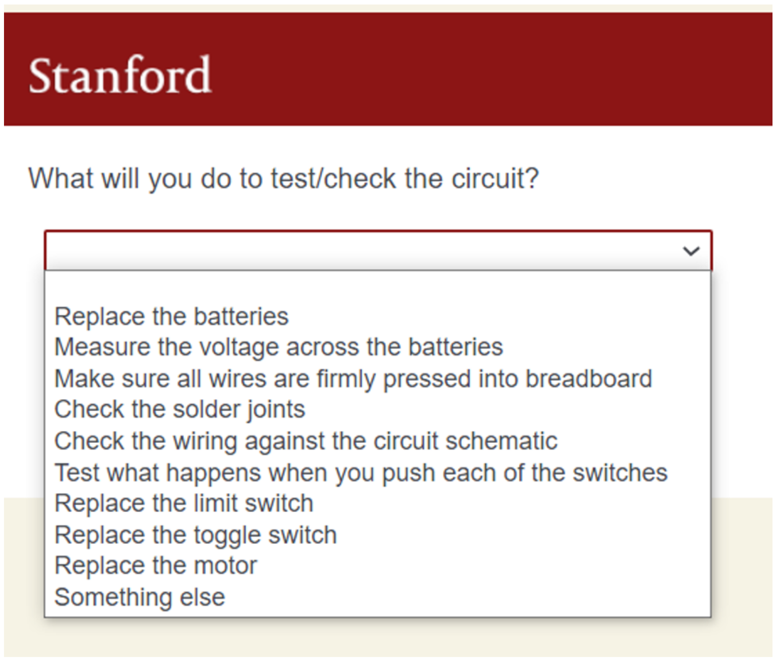
- Batteries are dead
- Batteries backward
- A wire fell out of the breadboard
- A weak solder joint broke
- The circuit is wired wrong
- The limit switch broke
- The toggle switch broke
- The motor broke
- The motor is backward

Could be the problem

NOT the problem

Figure B.5: First brainstorm causes question for miswired useless box activity

Then students got to keep performing tests on the circuit until they thought they knew what was wrong with the circuit. They were then given a dropdown menu with 9 choices (shown in Figure B.6).

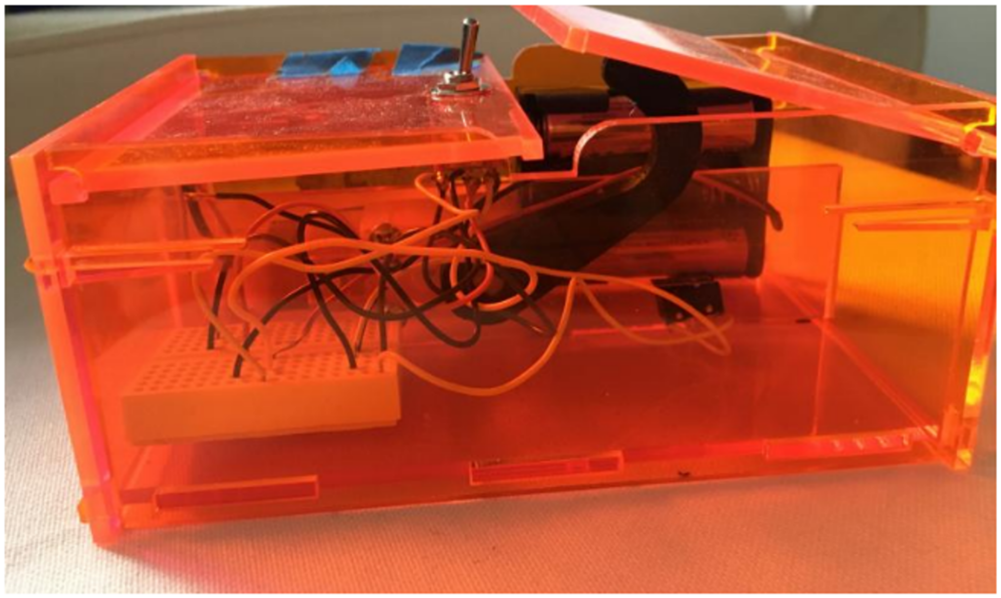

Stanford

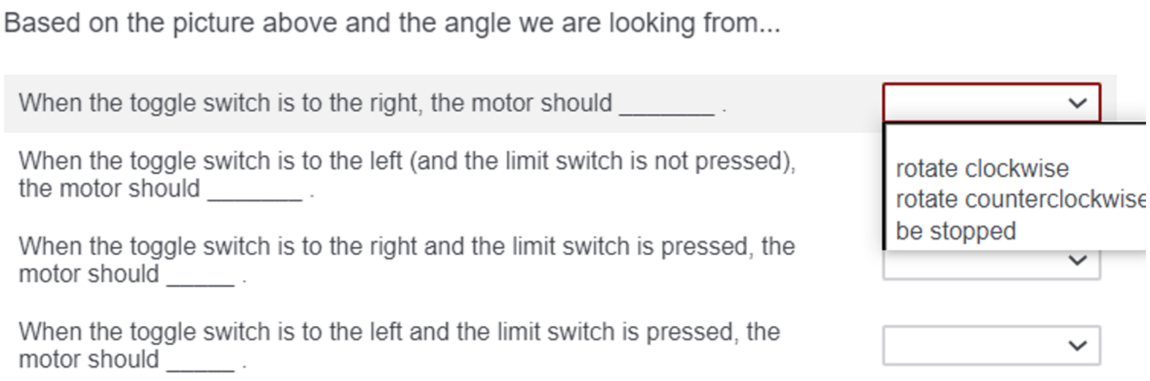
What will you do to test/check the circuit?

Replace the batteries
Measure the voltage across the batteries
Make sure all wires are firmly pressed into breadboard
Check the solder joints
Check the wiring against the circuit schematic
Test what happens when you push each of the switches
Replace the limit switch
Replace the toggle switch
Replace the motor
Something else

Figure B.6: Tests that can be performed on the miswired useless box circuit.

Replacing the limit switch, toggle switch, or motor did nothing. If a student checked the solder joints or that all wires were firmly pressed into the breadboard, they were told that everything looks good. To encourage students to isolate the problem instead of redoing every wire, students were shown a very messy picture of the jumbled wires if they asked to check the wiring and were told their TA suggests a different route. Hopefully, students eventually realized the most informative test was to test what happens when you push each of the switches. To test students' theoretical knowledge, students were given a finite state diagram to fill out before getting the results of their switches test (see Figure B.7).





Based on the picture above and the angle we are looking from...

When the toggle switch is to the right, the motor should ________ .

When the toggle switch is to the left (and the limit switch is not pressed), the motor should ________ .

When the toggle switch is to the right and the limit switch is pressed, the motor should ______ .

When the toggle switch is to the left and the limit switch is pressed, the motor should ______ .

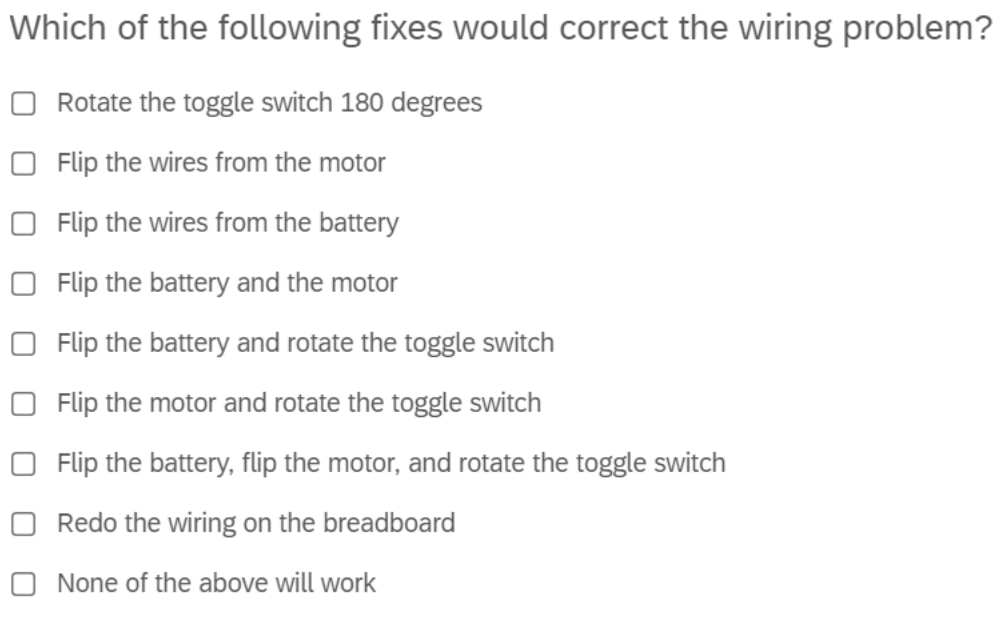
rotate clockwise
rotate counterclockwise
be stopped

Figure B.7: Test of theoretical knowledge for the miswired useless box.

Once a student said they were done performing tests on the circuit, they were taken to the final diagnosis screen. They were given a dropdown menu with the following choices of what could be wrong with the circuit:

(a) Batteries are dead
(b) A wire fell out of the breadboard
(c) A weak solder joint
(d) The circuit is wired wrong
(e) The limit switch broke
(f) The toggle switch broke
(g) The motor broke

Students who answered incorrectly were sent back to investigate further. Students who correctly answered that the circuit was wired wrong were asked a follow-up question about how to fix the miswiring (question shown in Figure B.8). Students were allowed to choose multiple answers. Since understanding the easiest way to fix the miswiring was beyond what we reasonably expected from students, we allowed the generic answer of "redo the wiring on the breadboard" as a sufficient answer to exit the loop and thus be finished.

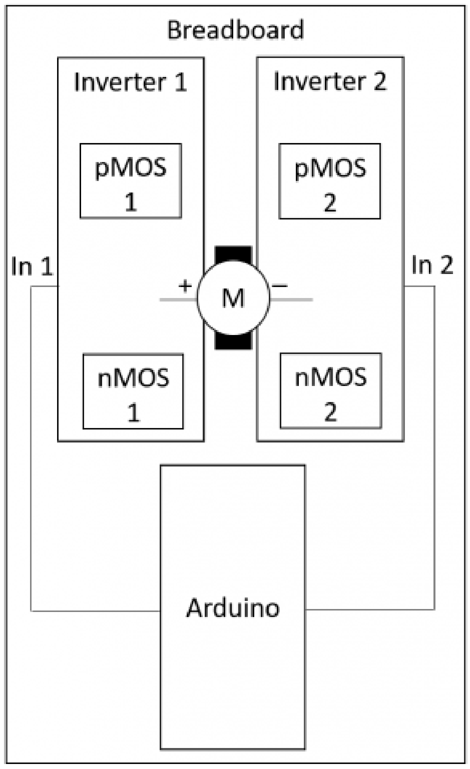

Which of the following fixes would correct the wiring problem?

- [ ] Rotate the toggle switch 180 degrees
- [ ] Flip the wires from the motor
- [ ] Flip the wires from the battery
- [ ] Flip the battery and the motor
- [ ] Flip the battery and rotate the toggle switch
- [ ] Flip the motor and rotate the toggle switch
- [ ] Flip the battery, flip the motor, and rotate the toggle switch
- [ ] Redo the wiring on the breadboard
- [ ] None of the above will work

Figure B.8: Final question about how to fix the circuit for the miswired useless box.

You go to test your motor driver (made out of two inverters). You notice that when you run the motor forwards it runs much slower than when you run the motor backwards.

Below is a sketch of the layout of your breadboard and a copy of the circuit schematic.



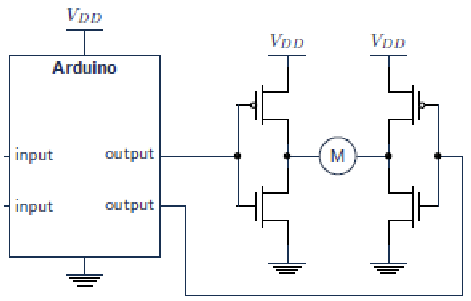


Figure B.9: Problem statement for miswired computerized useless box.

B.3 Miswired Computerized Useless Box

Students were given an initial problem description stating that the motor was moving faster in one direction than the other direction. They were given a cartoon schematic of the layout of the breadboard (but they could not see individual wires) and a theoretical circuit diagram laid out in a similar fashion, so the two would be easy to compare (see Figure B.9).

Each round students were asked what they suspected was wrong with the circuit and what test they wanted to perform to check the circuit. They were giving the following choices of what could be wrong with the circuit:

(a) A component is broken
(b) A component is backward or connected wrong
(c) The circuit is wired wrong
(d) The wrong signal was applied
(e) Short circuit
(f) Loose connections
(g) Arduino code incorrect
(h) I don't know
(i) Something else that I will specify below

For the possible tests of the circuit, students could choose between replace a component, compare circuit schematic and breadboard, measure voltages, check inputs, check Arduino code, push wires firmly into breadboard, or feel if components are hot.

If students chose to check inputs, they were first asked what voltage should be applied to each input to move the motor in the forward direction. If students answered this theory question incorrectly, they were given personalized feedback based on their answer. If they said both inputs should be HIGH or both inputs should be LOW, an explanation was given pointing out that for the motor to turn, there needs to be a voltage difference between both sides of the motor, so one side of the motor should be set to 5V and the other side to 0V. If a student got the voltages reversed, they were reminded that inverters output the inverse of the input voltage, so both their input voltages were reversed.

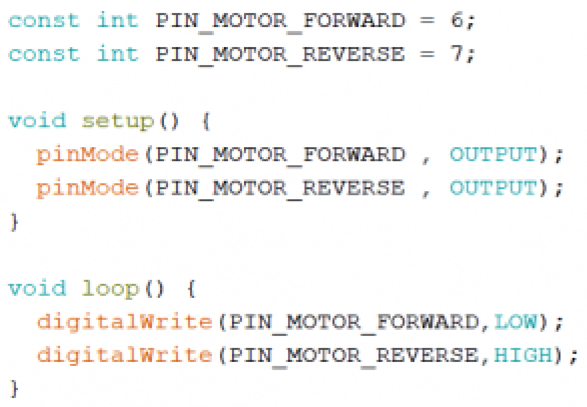
Comparing the circuit schematic and breadboard allowed students to check each connection one-by-one. To encourage taking measurements over checking each wire, we purposefully made this step tedious. For measuring voltages, students were given a list of each of the locations where they could take measurements and they were told they could choose up to four at once. See Table B.2 for a list of all the locations where students could check the wiring and voltage.

Checking the Arduino code revealed the code shown in Figure B.10.

What connection would you like to check?
Check that gate of nMOS 1 is connected to gate of pMOS 1
Check that gate of nMOS 2 is connected to gate of pMOS 2
Check that gate of nMOS 1 and pMOS 1 connects to Arduino
Check that gate of nMOS 2 and pMOS 2 connects to Arduino
Check that drain of pMOS 1 is connected to drain of nMOS 1
Check that drain of pMOS 2 is connected to drain of nMOS 2
Check that drain of pMOS 1 and nMOS 1 is connected to red wire of motor
Check that drain of pMOS 2 and nMOS 2 is connected to black wire of motor
Check that source of pMOS 1 is connected to source of pMOS 2
Check that source of nMOS 1 is connected to source of nMOS 2
Check that the source of both pMOS are connected to the 5V supply on the Arduino
Check that the source of both nMOS are connected to the Gnd on the Arduino

What voltages would you like to measure? (choose up to 4)
Voltage across motor
5V supply on Arduino
Gate of nMOS 1
Drain of nMOS 1
Source of nMOS 1
Gate of nMOS 2
Drain of nMOS 2
Source of nMOS 2
Gate of pMOS 1
Drain of pMOS 1
Source of pMOS 1
Gate of pMOS 2
Drain of pMOS 2
Source of pMOS 2

Table B.2: **Left:** List of all connections students could check for the miswired computerized useless box. **Right:** List of all voltages students could measure. They could measure up to 4 voltages at once.



```
const int PIN_MOTOR_FORWARD = 6;
const int PIN_MOTOR_REVERSE = 7;

void setup() {
  pinMode(PIN_MOTOR_FORWARD , OUTPUT);
  pinMode(PIN_MOTOR_REVERSE , OUTPUT);
}

void loop() {
  digitalWrite(PIN_MOTOR_FORWARD,LOW);
  digitalWrite(PIN_MOTOR_REVERSE,HIGH);
}
```

Figure B.10: Arduino code for the miswired computerized useless box. Students should find nothing wrong with the code.

After students indicated they are done performing tests on the circuit, the final diagnosis question asked what is wrong with the circuit and gave students six choices:

(a) The circuit is wired wrong
(b) The wrong signal was applied
(c) A component is broken
(d) A component is backwards or connected wrong
(e) Loose connections
(f) Arduino code incorrect

There was a follow-up question with six choices for what could be miswired in the circuit:

(a) The pMOS sources are not properly connected to the 5V supply on the Arduino.
(b) The nMOS sources are not properly connected to the Gnd on the Arduino.
(c) The gate and drain are mixed up on one transistor, the source and drain are mixed up on one transistor.
(d) A pMOS and nMOS are switched.
(e) Missing wire from Arduino output to inverter.

Students were given three choices for what could be wrong with the Arduino code in a follow-up question:

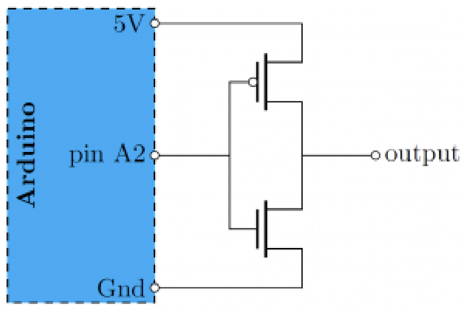
(a) Arduino code instantiates output pins wrong
(b) The code incorrectly uses digitalWrite()
(c) The forward pin should be set to HIGH and the reverse pin to LOW.

For students who said a component was backward or connected wrong, students were asked a follow-up question about which component was having problems. Students who said transistor had to go a step further and state which transistor they thought was acting up. This same follow-up question about the specific transistor was asked to people who said the circuit was miswired with the source and drain mixed up on a transistor. When students chose that pMOS 2 was connected incorrectly, they were told congratulations on debugging the circuit and allowed to exit. Any other answer was given the feedback that their answer was incorrect and sent back to keep taking measurements.

B.4 Miswired Inverter

This assignment was created with multiple checkpoints where students could not move on to the next section until they correctly answered the previous step. First, students were asked what output they expected from the inverter for each possible input (see Figure B.11). If students incorrectly filled out the truth table, then they would be told that their expectation was incorrect and asked the same question again. Students were given four chances to answer correctly. Once students correctly filled out the truth table, they moved on to the next section, where they could take measurements on the circuit.

You built the following circuit in lab. You are using the Arduino to power and control the transistor circuit. Your task is to figure out whether the circuit is working properly or not.

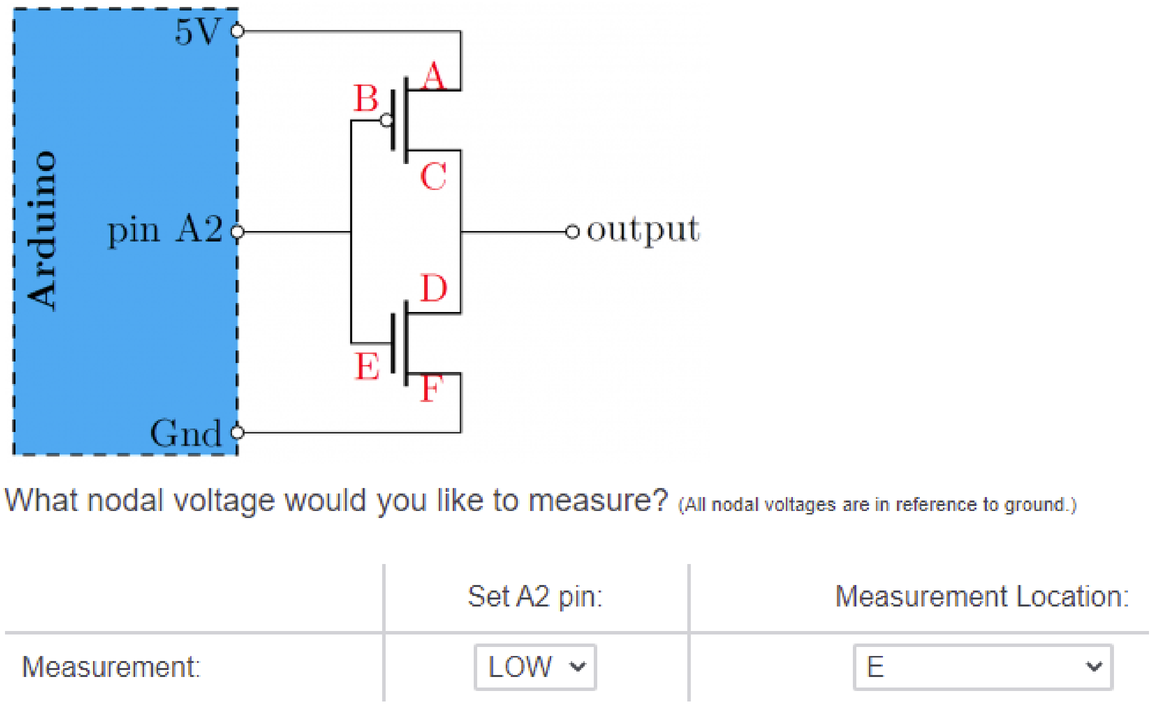


Before you take measurements, what output do you expect for each of the possible inputs?

	Output	
	LOW	HIGH
Pin A2 LOW	○	○
Pin A2 HIGH	○	○

Figure B.11: Expectations question for inverter activity.

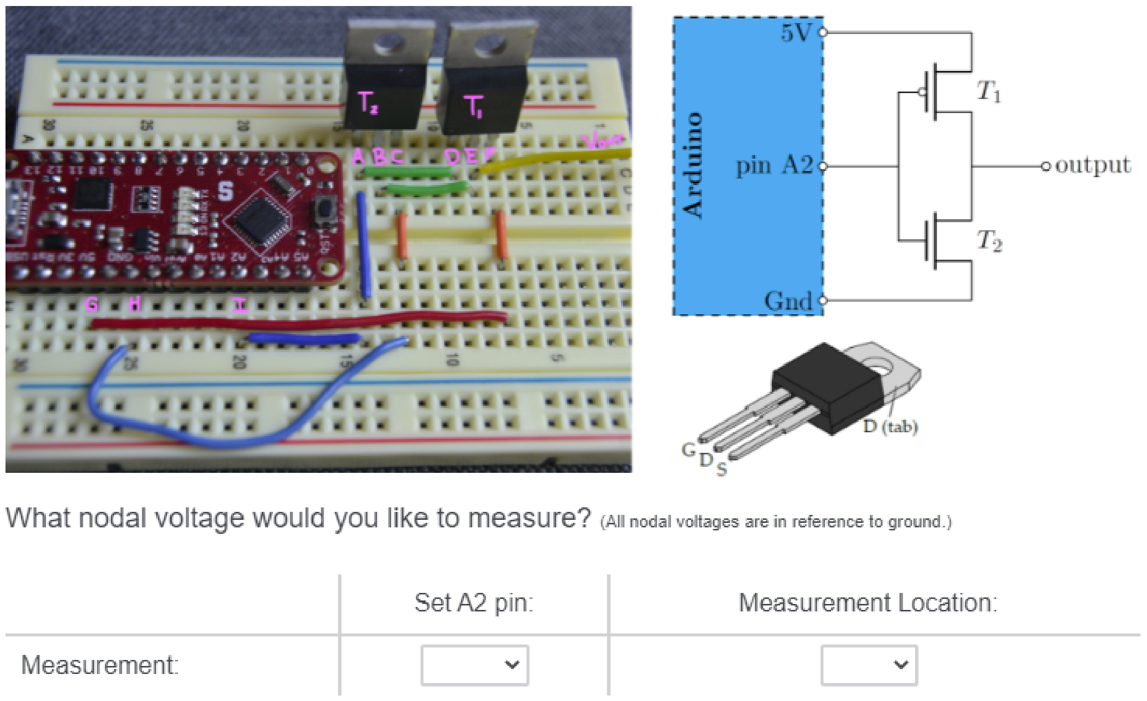
After that, students were split randomly into two groups, with half of them receiving a theoretical diagram where they could take measurements (see Figure B.12), the other half receiving a picture of a breadboard where they could take measurements (see Figure B.13). Each location where they could ask for a nodal voltage was labeled with a letter. Students could take as many measurements as they wanted until they thought they knew what was wrong with the circuit.



What nodal voltage would you like to measure? (All nodal voltages are in reference to ground.)

	Set A2 pin:	Measurement Location:
Measurement:	LOW	E

Figure B.12: Measurement page for theoretical version of miswired inverter.

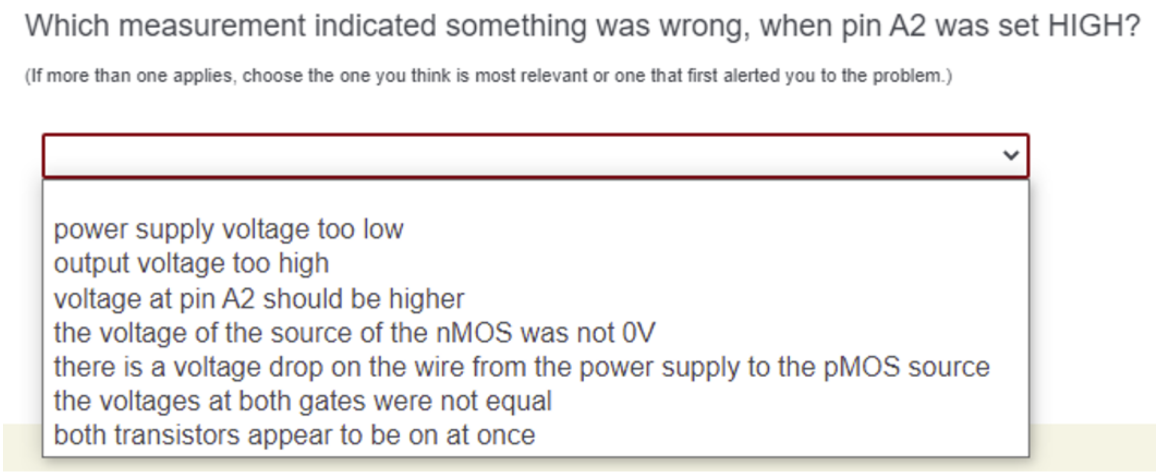


What nodal voltage would you like to measure? (All nodal voltages are in reference to ground.)

	Set A2 pin:	Measurement Location:
Measurement:		

Figure B.13: Measurements page for physical version of miswired inverter.

After students received each measurement, they were asked whether they wanted to take another measurement or whether they were done investigating. They were then prompted to decide whether the circuit was working properly or not. If students said it was working, we asked them an open-ended question about what measurements indicated it was working properly. We will use these responses to create a closed-form question in the future. Students were then told their conclusion was incorrect, and were given an extra clue–that when the input is set HIGH, both transistors are hot to the touch. Students were then sent back to take measurements.


Which measurement indicated something was wrong, when pin A2 was set HIGH?

(If more than one applies, choose the one you think is most relevant or one that first alerted you to the problem.)

power supply voltage too low
output voltage too high
voltage at pin A2 should be higher
the voltage of the source of the nMOS was not 0V
there is a voltage drop on the wire from the power supply to the pMOS source
the voltages at both gates were not equal
both transistors appear to be on at once

Figure B.14: Choices of what is wrong with the circuit for inverter activity.

If students said the circuit was not working, they were asked follow-up questions about what measurements indicated something was wrong with the circuit. First they were asked if they noticed problems with the measurements when the input was HIGH or LOW or both. We presumed most students would just notice the large irregularities when the input was set HIGH. For those who said they noticed problems when the input voltage was HIGH, they were asked to specify which of the following specific problem they saw:

(a) Power supply voltage too low
(b) Output voltage too high
(c) Voltage at pin A2 should be higher
(d) The voltage of the source of the nMOS was not 0V
(e) There is a voltage drop on the wire from the power supply to the pMOS source
(f) The voltages at both gates were not equal
(g) Both transistors appear to be on at once

If multiple statements were true, students were told to choose the one they thought was most relevant. Note that the voltages at both gates were equal, so that answer was a red herring. See Figure B.14 for the exact wording of the problem as it was displayed to students. For students who said they noticed problems when the input voltage was LOW, they were asked to specify which of the following specific problem they saw: the output voltage was lower than expected, the output voltage was higher than expected, the output voltage should have been near 0V, the power supply voltage should have been 5V. Again, there were several red herring answers–in this case the output voltage was higher than expected and the output voltage should have been near 0V.

Finally, students were asked what was wrong with the circuit and given a dropdown menu with the following choices: a miswiring–something is shorting, transistor is wired wrong, transistor is broken, wrong type of transistor used, pin A2 is not outputting correct voltages, or 5V pin on the Arduino is broken. To help students who get the wrong answer, we gave them feedback based on their response. For students who chose one of the first two answers, we responded:

You check the wiring and everything looks fine. While doing this, you notice the transistors are hot when pin A2 is set to HIGH.

For the last two choices, we gave students the feedback:

The Arduino outputs the correct voltages on the 5V and A2 pins when disconnected from the breadboard.

If students said the transistors were broken, they were told:

You replace both transistors, and the circuit works afterward. You think you have solved the problem, but the TA points out that the original transistors were actually working.

They were also shown a picture of the transistors used in both cases with the labels on each transistor clearly visible (see Figure B.15). If students chose the intended answer–that the wrong transistor was used–they were given a follow-up question asking which transistor was incorrect. If they chose the wrong transistor, then they were told:

You have correctly isolated the problem to the type of transistors used, but you have not isolated the correct transistor(s).

Like the previous problems, students were required to keep guessing until they got to the correct fix.

B.5 Row not working on LED display

Students started out with a video of the 8x8 LED display with the third-to-top row not lighting up (see Figure B.16). With this video came a looping drilldown question which asked students to indicate their current best guess as to what was wrong with the circuit and what they wanted to check next on the circuit.

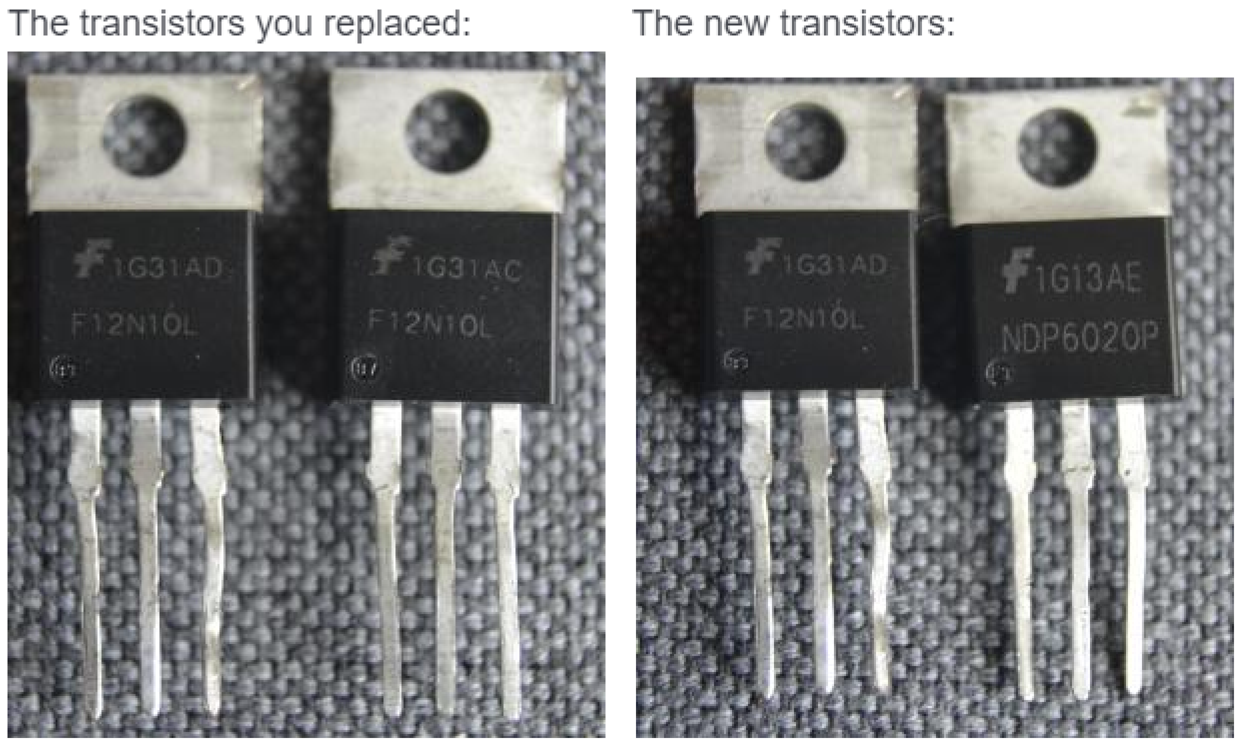
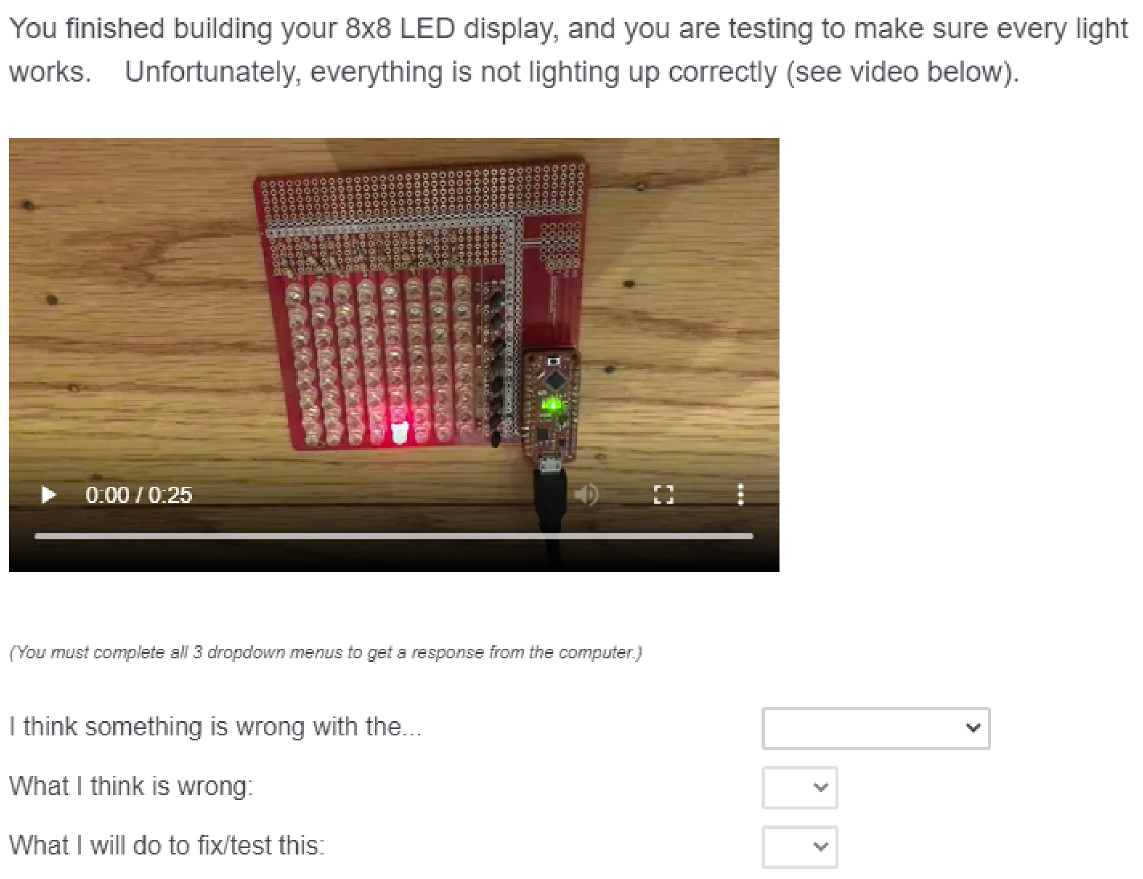


Figure B.15: Comparison of labels of transistors for inverter activity.



You finished building your 8x8 LED display, and you are testing to make sure every light works. Unfortunately, everything is not lighting up correctly (see video below).

(You must complete all 3 dropdown menus to get a response from the computer.)

I think something is wrong with the...

What I think is wrong:

What I will do to fix/test this:

Figure B.16: Initial problem description and looping drill down question for LED display debugging activity.

Students could choose to replace or inspect each part. Inspecting the part meant they got to see a close-up image of the component and its arrangement on the board (see Figure B.17). Choosing replace components returned the following message:

> *It is very hard to de-solder components from the board. This action should not be taken until you are absolutely certain what is wrong with the circuit, and there is no alternative course of action. If you end up needing to de-solder during lab, your TA will help you walk through the process, and you may have to destroy the component in the process.*

Students could also check the resistance of the resistors and look at the Arduino code (see Figure B.18). See Table B.19 for the full list of guess and action combinations that were available to students.

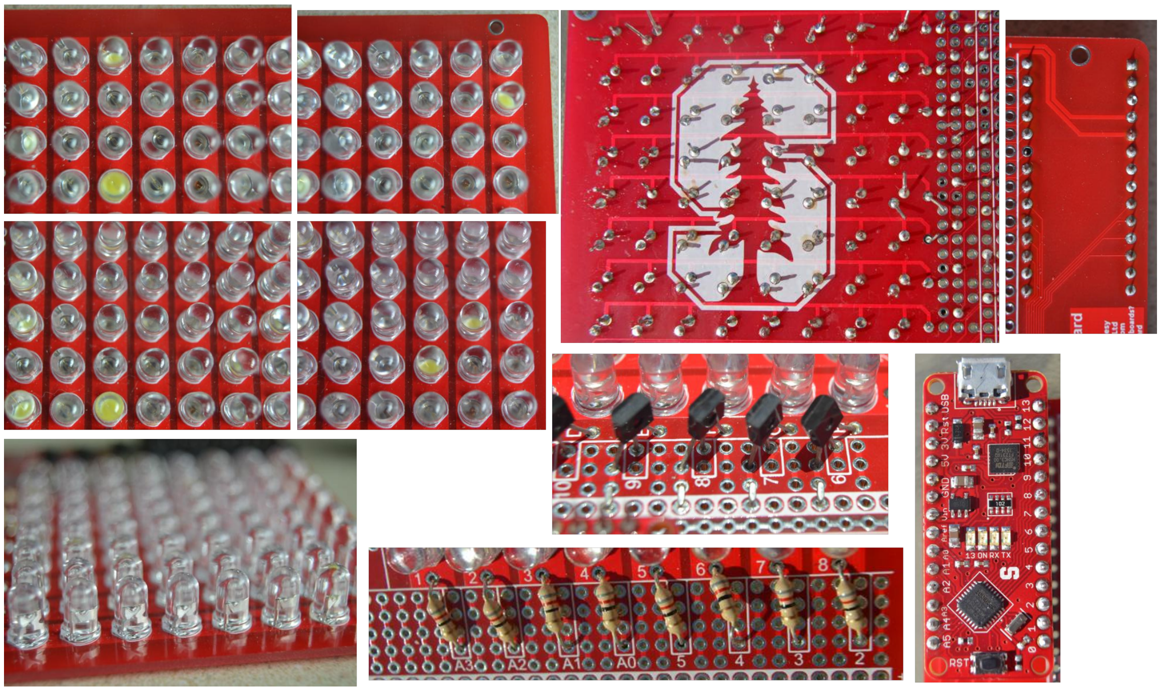
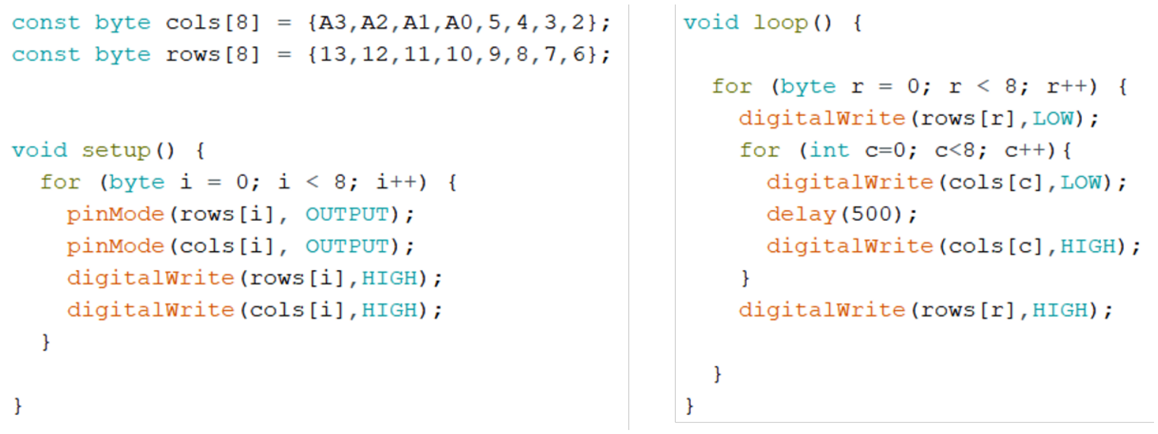


Figure B.17: **Pictures for LED display debugging activity.** Students could ask to inspect various parts of the circuit board to try and find where the problem was. **Upper and lower left:** inspection of the LEDs, **Upper right:** two pictures show back of PCB board, **Middle:** inspection of transistors, **Bottom middle:** inspection of resistors, and **Bottom right:** inspection of Arduino. In the upper right picture showing the back of the circuit board where the Arduino socket connects to the board, you will find the 6th pin down on the left is missing solder.



```
const byte cols[8] = {A3,A2,A1,A0,5,4,3,2};
const byte rows[8] = {13,12,11,10,9,8,7,6};


void setup() {
  for (byte i = 0; i < 8; i++) {
    pinMode(rows[i], OUTPUT);
    pinMode(cols[i], OUTPUT);
    digitalWrite(rows[i],HIGH);
    digitalWrite(cols[i],HIGH);
  }

}
```

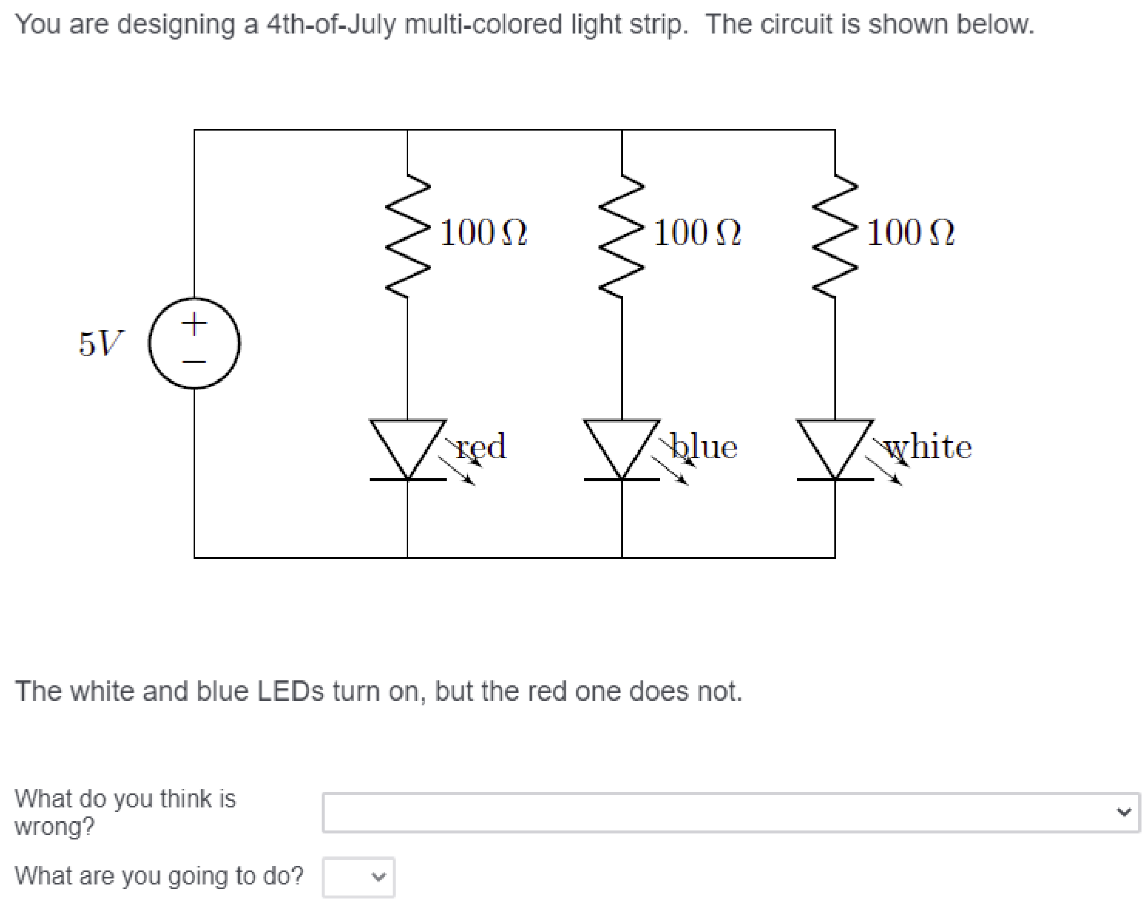
```
void loop() {

  for (byte r = 0; r < 8; r++) {
    digitalWrite(rows[r],LOW);
    for (int c=0; c<8; c++){
      digitalWrite(cols[c],LOW);
      delay(500);
      digitalWrite(cols[c],HIGH);
    }
    digitalWrite(rows[r],HIGH);

  }
}
```

Figure B.18: Arduino code for the LED display debugging activity. Students should find nothing wrong with the code.

When students indicated they are done exploring, they were given a branching set of questions regarding what was wrong with the circuit. First they were asked whether the problem was in the code or in the physical circuit. If the student chose code, they were asked whether the problem was that pin 8 was not initialized or that pin 8 was incorrectly set HIGH instead of LOW to turn pMOS on. If students indicated something was wrong with the physical circuit, then they were asked to isolate the problem to a specific component or area of the circuit: pMOS, LED(s), resistor(s), Arduino, or connections. If the student chose pMOS, they were asked whether the pMOS in the row labeled 8 was broken, its drain and source switched, its gate and drain switched, or not soldered solidly. If the student chose LED(s), they were asked whether the LEDs in the row labeled 8 were broken, backward, the wrong color, or not soldered solidly. If the student said the resistors had a problem, they were asked whether they were the wrong resistance, connected wrong, or not connected solidly. If they said the Arduino had a problem, they were asked to specify whether the Arduino was not powered properly, the pin was broken pin 8, the pins not soldered solidly, or solder missing on a socket pin. Finally, if the student said the connections had a problem, they had a follow up question where they could specify wither the resistors should connect to ground, the pMOS in the row labeled 8 was missing a connection, the first LED in the row labeled 8 was missing solder on the lead, pin 8 on the Arduino was not connected on the back of the board, or a connection was broken internally in the PCB. Both the Arduino was missing solder on a socket pin or pin 8 on the Arduino was not connected on the back of the board would allow students to exit the cycle with a *Congratulations! You debugged the circuit!* message. All other choices were told *Not quite. Please keep investigating*, and sent back to the drill down question where you can inspect the circuit.

Something is wrong with the	What I think is wrong	What I will do to fix/test this
Code	Pin initialized wrong	Check void setup() code
Code	Incorrectly set row/column high/low	Check void loop() code
Code	Something else. I will specify below.	Check void setup() code
Code	Something else. I will specify below.	Check void loop() code
pMOS	pMOS broken	Replace pMOS
pMOS	pMOS broken	Inspect pMOS
pMOS	pMOS backwards	Replace pMOS
pMOS	pMOS backwards	Inspect pMOS
LED(s)	LED(s) broken	Replace LED(s)
LED(s)	LED(s) broken	Inspect LED(s)
LED(s)	LED(s) broken	Test LED(s) using diode function on DMM
LED(s)	LED(s) backward	Replace LED(s)
LED(s)	LED(s) backward	Inspect LED(s)
LED(s)	LED(s) backward	Test LED(s) using diode function on DMM
Resistor(s)	Backward resistor(s)	Replace resistor(s)
Resistor(s)	Backward resistor(s)	Inspect resistor(s)
Resistor(s)	Broken resistor(s)	Replace resistor(s)
Resistor(s)	Broken resistor(s)	Inspect resistor(s)
Resistor(s)	Broken resistor(s)	Measure resistance
Resistor(s)	Wrong resistance	Replace resistor(s)
Resistor(s)	Wrong resistance	Inspect resistor(s)
Resistor(s)	Wrong resistance	Measure resistance
Connections	Missing connection	Inspect back of PCB board
Connections	Missing connection	Inspect Arduino pins
Connections	Missing connection	Inspect resistor placement
Connections	Missing connection	Inspect pMOS connections
Connections	Missing connection	Something else. I will specify on next page.
Connections	Bad solder joint	Inspect back of PCB board
Connections	Bad solder joint	Inspect Arduino pins
Connections	Bad solder joint	Inspect resistor placement
Connections	Bad solder joint	Inspect pMOS connections
Connections	Bad solder joint	Something else. I will specify on next page.
I don't know	I don't know	Check void setup() code
I don't know	I don't know	Check void loop() code
I don't know	I don't know	Replace pMOS
I don't know	I don't know	Inspect pMOS
I don't know	I don't know	Replace LED(s)
I don't know	I don't know	Inspect LED(s)
I don't know	I don't know	Test LED(s) using diode function on DMM
I don't know	I don't know	Replace resistor(s)
I don't know	I don't know	Inspect resistor(s)
I don't know	I don't know	Measure resistance
I don't know	I don't know	Inspect back of PCB board
I don't know	I don't know	Inspect Arduino pins
I don't know	I don't know	Something else. I will specify on next page.
Something Else	I will specify on next page.	Check void setup() code
Something Else	I will specify on next page.	Check void loop() code
Something Else	I will specify on next page.	Replace pMOS
Something Else	I will specify on next page.	Inspect pMOS
Something Else	I will specify on next page.	Replace LED(s)
Something Else	I will specify on next page.	Inspect LED(s)
Something Else	I will specify on next page.	Inspect back of PCB board
Something Else	I will specify on next page.	Inspect Arduino pins
Something Else	I will specify on next page.	Inspect resistor placement

Figure B.19: Spreadsheet containing the data for the Qualtrics 3-level drill-down question used to let students navigate the LED display debugging activity. The first menu item asked students to identify which part of the circuit they suspected had the problem. The second menu item allowed for a more specific answer on that guess and the third menu item asked them to take an action to test the circuit (or fix it).



You are designing a 4th-of-July multi-colored light strip. The circuit is shown below.

The white and blue LEDs turn on, but the red one does not.

What do you think is wrong?

What are you going to do?

Figure B.20: Problem description for the LED circuit with an incorrect resistor.

B.6 Incorrect resistor in LED circuit

Students were told that they are designing a Fourth of July light strip, but the red LED was not lighting up. The looping question showed them the circuit schematic, asked what they thought was wrong and asked them what they are going to do to test the problem using a drill down question (see Figure B.20). Examples of actions students could take included flip or replace the LED, flip or replace the resistor, measure the resistance, use the diode function on the DMM to test the LED, measure voltage across the LED, resistor or battery, or inspect the wiring. A complete list of actions the students could take is listed in Table B.21.

What do you think is wrong?	What are you going to do?
I have no idea what is wrong.	Check the red LED.
I have no idea what is wrong.	Look at circuit to check wiring.
I have no idea what is wrong.	Investigate by measuring voltages in circuit.
red LED backward	Flip red LED
red LED backward	Replace red LED
red LED backward	Measure voltage across red LED
red LED backward	Use diode function of DMM to test whether red LED works.
red LED backward	Something else
LED broken	Replace red LED
LED broken	Measure voltage across red LED
LED broken	Use diode function of DMM to test whether red LED works.
LED broken	Exchange location of red and blue LEDs
LED broken	Something else
resistor backward	Replace resistor
resistor backward	Flip resistor
resistor backward	Exchange location of red and blue LEDs
resistor backward	Measure voltage across resistor (that is in series with red LED)
resistor backward	Something else
resistor broken	Replace resistor
resistor broken	Flip resistor
resistor broken	Exchange location of red and blue LEDs
resistor broken	Measure voltage across resistor (that is in series with red LED)
resistor broken	Measure resistance of resistor (that is in series with red LED) with DMM
resistor broken	Something else
wrong resistance value	Check resistor color code
wrong resistance value	Measure resistance of resistor (that is in series with red LED) with DMM
wrong resistance value	Something else
miswired– connections between individual components incorrect	Rewire circuit to fix any mistakes
miswired– connections between individual components incorrect	Look at circuit to check wiring.
miswired– connections between individual components incorrect	Something else
bad wires or connections	Push wires and components firmly into board.
bad wires or connections	Look at circuit to check wiring.
bad wires or connections	Rewire circuit to fix any problems
bad wires or connections	Measure voltage across red LED
bad wires or connections	Measure voltage across resistor (that is in series with red LED)
bad wires or connections	Use DMM in resistance mode to check for negligible resistance across connections.
bad wires or connections	Something else
breadboard broken	Transfer red LED and series resistor to new row of breadboard.
breadboard broken	Use DMM in resistance mode to check for negligible resistance between pins in same row.
breadboard broken	Something else
bad circuit design– the LED should never light up based on this design	Fix circuit diagram and rebuild circuit.
something else	I will specify on next page.

Figure B.21: Spreadsheet used to create list of possible answers and actions for the Qualtrics two-level drill-down question for the LED circuit with an incorrect resistor.

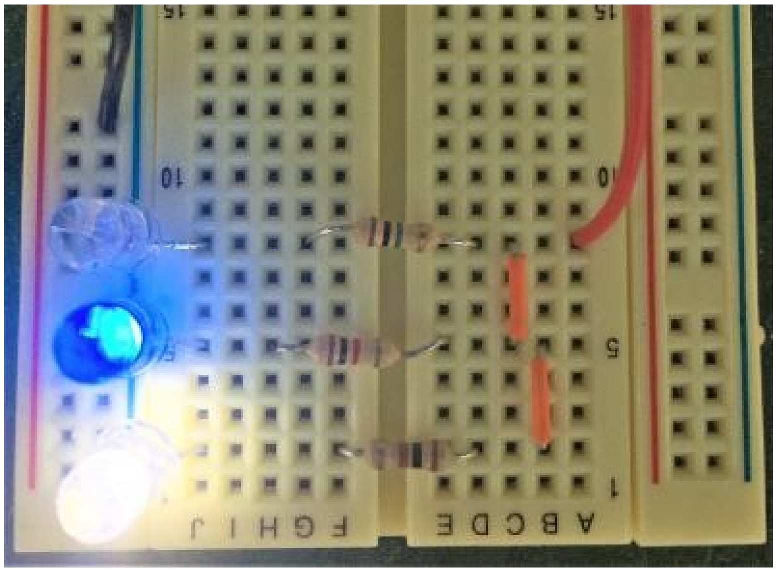


Figure B.22: Picture of LED circuit with the red LED not properly lighting up.

Many actions, like flipping or replacing the LED or flipping the resistor did not change the functionality of the circuit. Testing the LEDs with the DMM showed that they were working properly. To uncover the wrong resistor, there were a number of measurements students could take. Students could exchange the location of the red and blue LEDs and then notice that the blue LED was now the one working, which would isolate the problem to the branch of the circuit where the red LED originally was. If they went the voltage-measuring route, they might notice that there was a voltage drop across both the red LED and resistor in series with it, indicating some current is flowing through it (so it is not backward). Alternatively, they could inspect the picture of the breadboard and notice the difference in color bands between the resistor (see Figure B.22). Finally, students could check the resistance of the resistor using their DMM to determine the resistance was much higher than indicated in the theoretical diagram.

When students stated they had enough information to solve the problem, they were sent to the final diagnosis question where they were asked to choose between a list of 10 possible things that could be wrong with the circuit. The choices were:

(a) LED was broken
(b) LED was backward
(c) Resistor was broken
(d) Resistor was backward
(e) Resistor was wrong resistance
(f) Problem with battery pack
(g) Circuit was wired wrong
(h) Loose wire or bad connection
(i) Broken breadboard
(j) Bad circuit design

Notice that some of these options are red herrings that tested the students' theoretical knowledge. For example, resistors do not have directionality, so resistor backward makes no sense. Similarly, bad circuit design indicates that the student thinks that not all the LEDs should have lit up in this case. For those who correctly indicate the resistor is the wrong resistance, a follow-up question is asked about what resistance was accidentally put in. If students do not correctly answer $1\,\mathrm{M}\Omega$, students are sent back to keep investigating.

B.7 LED circuit with bad connection

The following branching question structure was used for the final checkpoint to check whether students correctly diagnosed the loose resistor in the LED circuit in our own debugging simulator. First students had to say whether the problem was something wrong with the battery pack, LED(s), resistors, breadboard, bad connections or loose wires, or the circuit was miswired. If the student said something was wrong with the battery pack, they were given the following choices to choose between:

(a) The batteries are dying. Replace the batteries.
(b) One battery is in the battery pack backward.
(c) The red and black wires from the battery pack should be switched.
(d) One of the wires from the battery pack is not firmly connected to the breadboard.

None of these answers are realistic, but some students chose them.

If the student said something was wrong with the LED(s), they were asked a follow-up question about which LED is not working properly. The choices were the red LED, the blue LED, one of the white LEDs, or both the blue and a white LED. If they correctly said the blue LED, then they were given another follow-up question asking whether the blue LED was broken, backwards, or not pushed well into the breadboard. If they said the LED was not pushed well into the breadboard, they were given the following feedback:

> *It looks like you have correctly isolated the problem to the blue row of the circuit. If the blue LED is not firmly pushed in, but all other connections are good, we would expect the voltage of the row where the blue LED and resistor connect to be the same as the voltage of the power supply due to the connection of the resistor (with no current flowing through it) to the power supply.*

This was supposed to guide them that they were on the right track but chose the wrong component as being loose. Alternatively, if they said that the blue LED was broken or backwards, they were given the following feedback:

> *It looks like you have correctly isolated the problem to the blue row of the circuit. When you use the diode function on your DMM to test the LED you find it is working and correctly oriented in the circuit.*

This new piece of information hopefully allowed them to eliminate that the LED is broken or backward.

If the student said something was wrong with the resistors, then they were asked which resistor was having problems. If they said the one in the row with the blue LED, then they were asked the follow-up question of what was wrong with the resistor. They could choose that the resistor was broken, backward, the wrong resistance, or not firmly connected to the breadboard. Not firmly connected to the breadboard was the intended answer, so students were allowed to exit the activity at that point. All other answer led to the message, “Not quite. Please investigate further. Click next when you are prepared to answer again.”

If the student said something was wrong with the breadboard, they were given the following choices:

(a) The breadboard is broken and not electrically connected on the ground row.
(b) The breadboard is broken and not electrically connected on the row where the resistor and blue LED are connected.
(c) The breadboard is electrically connected horizontally, not vertically, in the middle section, so the components are connected incorrectly.

If students chose the breadboard was not electrically connected between the resistor and blue LED, then they were allowed to exit the activity, but first they were given one final feedback and open-ended response. They were told:

It sounds like you correctly isolated where the problem is. You move the wires to a different row of the breadboard and it works! But, you used this breadboard last week and it worked fine (and it turns out the breadboard is still fine). What is an alternate explanation for what caused this problem?

If the student said there were bad connections, they were given the following choices about where the connections could be bad:

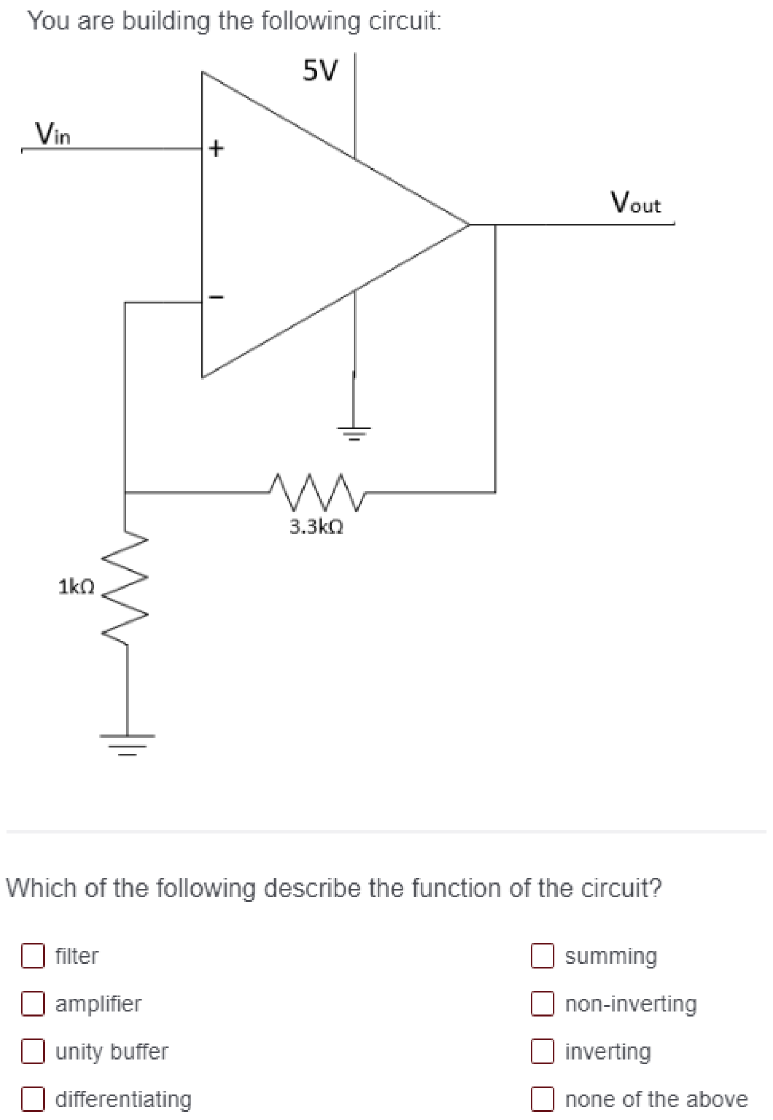
(a) The red wire from the battery pack is not firmly connected to the breadboard.
(b) The resistor in the row with the blue LED is not firmly connected to the breadboard.
(c) The blue LED is not firmly connected to the breadboard.
(d) The top white LED is not firmly connected to the breadboard.
(e) Something is short-circuiting in the circuit.

The first choice and last two choices were not consistent with the data. The second choice was the intended answer, and allowed students to exit the activity. The third answer led students to the personalized feedback mentioned in the LED paragraph (3 paragraphs before) where they were told that they had correctly isolated the problem to the path with the blue LED, but pointed out why the symptoms were inconsistent with a loose LED.

For those who said the circuit was miswired, they were given an open-ended essay blank where they could explain what they thought was wrong. Regardless of what they wrote, the computer would give them the feedback that their answer was incorrect and tell them to investigate further.

B.8 Non-inverting amplifier with broken op amp

First students were shown a theoretical diagram of the op amp circuit they were supposedly building in lab and asked two theoretical knowledge questions. The first was a multiple choice question where you could choose multiple descriptor words that described the function of the op amp circuit (see Figure B.23). The second question asked for the gain of the circuit.



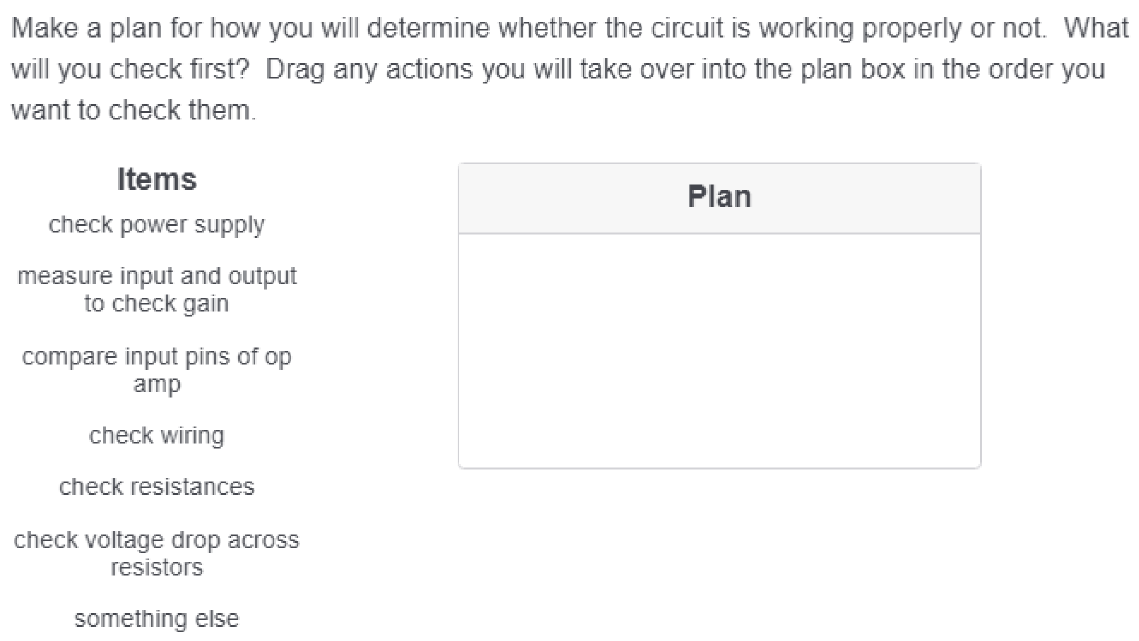
You are building the following circuit:

Which of the following describe the function of the circuit?

- ☐ filter
- ☐ amplifier
- ☐ unity buffer
- ☐ differentiating
- ☐ summing
- ☐ non-inverting
- ☐ inverting
- ☐ none of the above

Figure B.23: Expectation knowledge question for the non-inverting op amp with broken op amp. Students should pick amplifier and non-inverting for the correct words describing the function of the circuit.

Next students were asked to make a plan. A drag-and-drop question format was used to allow students to choose what measurements they would like to take from a list and prioritize them in a box labeled “Plan” (see Figure B.24).



Make a plan for how you will determine whether the circuit is working properly or not. What will you check first? Drag any actions you will take over into the plan box in the order you want to check them.

Items

check power supply

measure input and output to check gain

compare input pins of op amp

check wiring

check resistances

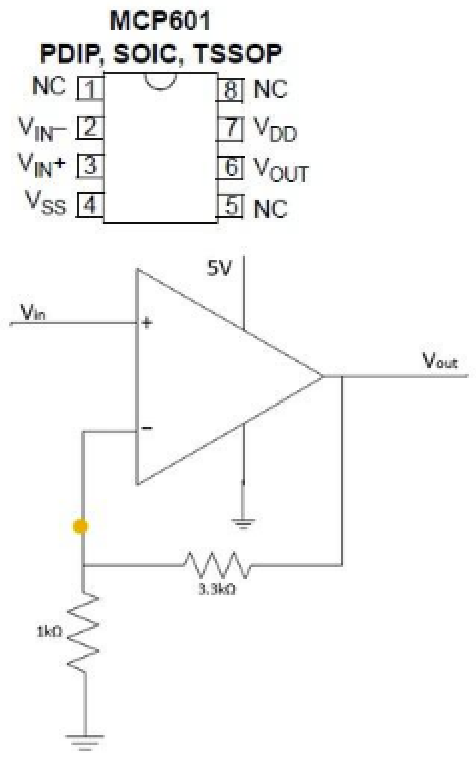
check voltage drop across resistors

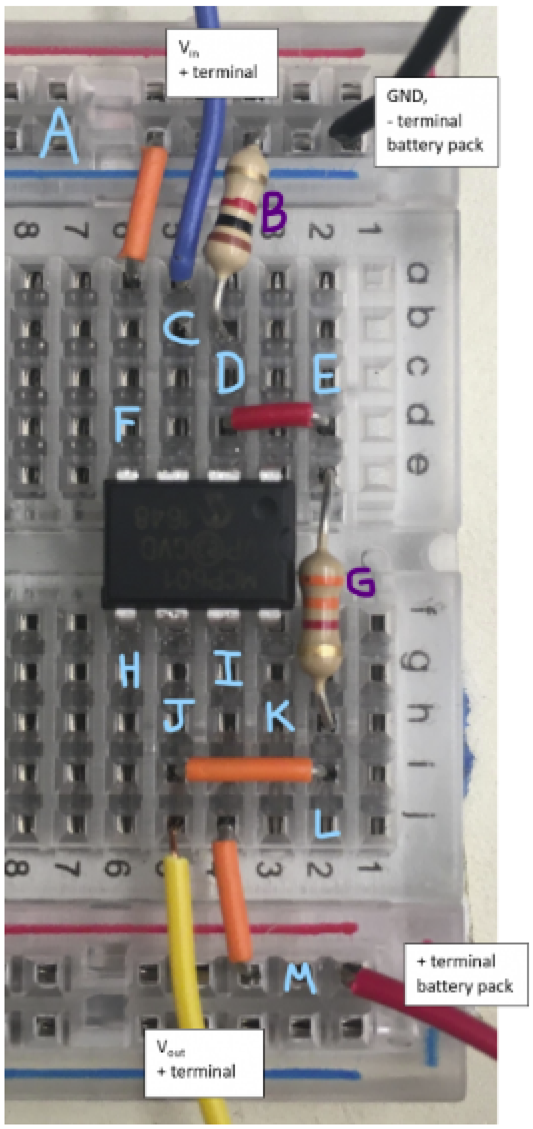
something else

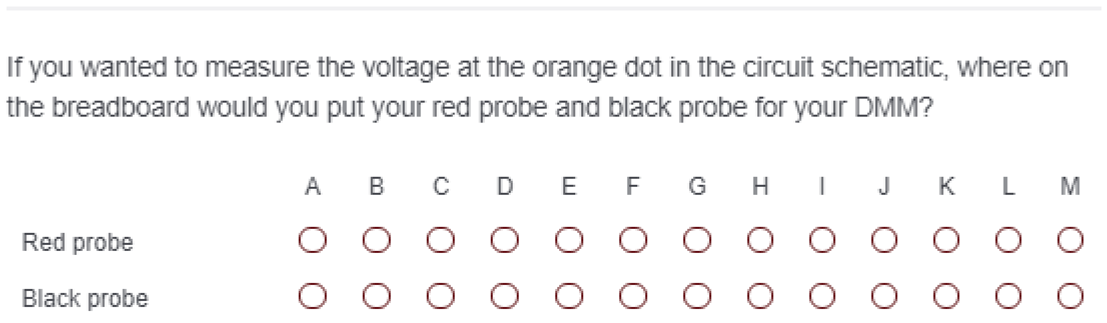
Plan

Figure B.24: Make a plan question for the non-inverting amplifier with a broken op amp. Students drag and drop the actions into the plan box in the order they want to do them.

After that, students were asked two measurement knowledge questions. First they were asked whether the DMM or oscilloscope would be a better tool to use in this situation with a constant input voltage. Then they were given a star on the theoretical diagram and asked to choose the letter locations on the physical circuit picture where the red and black probe should go to measure the starred location (see Figure B.25).






If you wanted to measure the voltage at the orange dot in the circuit schematic, where on the breadboard would you put your red probe and black probe for your DMM?

	A	B	C	D	E	F	G	H	I	J	K	L	M
Red probe	○	○	○	○	○	○	○	○	○	○	○	○	○
Black probe	○	○	○	○	○	○	○	○	○	○	○	○	○

Figure B.25: Probe placement question. Students are asked to identify the letter locations for the red and black probe to measure the voltage at the location of the yellow dot on the circuit schematic.

Students could take up to 14 measurements at any of 13 locations labeled A-M on the physical picture (see Figure B.25). For students in the reflective guidance group (intervention), they required to answer what they expected the measurement to be before receiving each measurement and they had to answer an open-ended question about what they learned from the measurement afterwards before continuing to the next measurement. In contrast, the control group was asked an open-ended question about op amps in general in between each measurement. The questions were in the following order:

1. What are the golden rules?
2. What are KVL and KCL?
3. What is the definition of gain?
4. What is the difference between an inverting and non-inverting amplifier?
5. What does it mean for an op amp to be an "active" component (versus a "passive" component like a resistor)?
6. What is the importance of negative feedback in op amp circuits?
7. In what situations would you want to use an oscilloscope instead of a DMM to measure voltages?
8. Why is it important to pull a resistor out of the circuit before measuring its resistance?
9. How would you improve this interface?
10. What does it mean for an op amp to be saturated?
11. Name at least one important piece of information you can get from an op amp's datasheet.

For students who wanted to take more measurements, we just let them take the final few measurements at no cost (the cost being a question afterwards).

When students were ready, they could choose I am done taking measurements. The circuit is working properly, or *I am done taking measurements. The circuit is NOT working correctly*, instead of taking another measurement. If students did not choose working or not working by the final cycle of the measurement loop, then a question appeared forcing them to choose. If the student said the circuit was working, then they were asked the open-ended response: *How do you know the circuit is working properly?* If the student said the circuit was not working properly, then they were asked the open-ended response:

> *How did you conclude the circuit was not working? What is your best guess as to what is wrong with the circuit? How could you fix the circuit?*

Finally, students were asked how confident they were about their answer on the following 5 point scale:

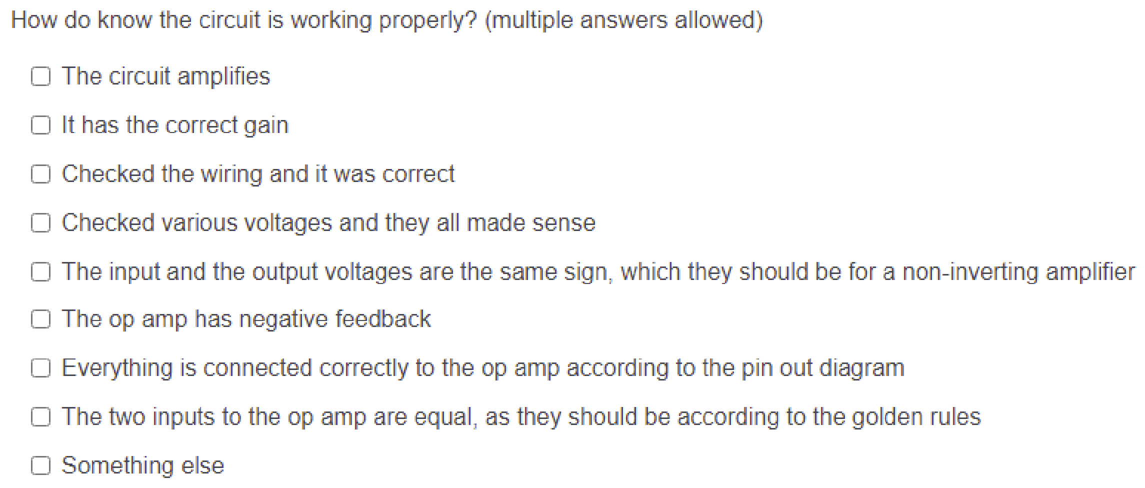
I think my answer is incorrect – I doubt my answer – I am not sure – Pretty sure I am right – Definitely got 100%.

B.9 Non-inverting amplifier with loose wire in feedback

If the computer could not find words related to non-inverting and amplifier in the students' submission they were given the following multiple choice answers for the function of the circuit: summing amplifier, differential amplifier, inverting amplifier, non-inverting amplifier, unity buffer, low pass filter, and high pass filter.

For the make a plan section, students were given an identically worded problem to the other op amp debugging activity (see Figure B.24). The only difference was that students had an additional option of "check the voltage at each op amp pin".

If a student said the circuit was working properly, they were given a multiple-choice follow-up question about how they knew it was working properly. Students could choose multiple answers, and most the answers were true statements about the circuit, like the wiring matched the theoretical diagram or the output voltage is greater than the input voltage (see Figure B.26). While these answers were true, they were also not the full picture, and so when students gave these true, but incomplete answers, we encouraged them with the following feedback: "While these are true statements, debugging is needed whenever at least a single piece of information does not match expectation. Was there any measurement that didn't match your expectations?"



How do know the circuit is working properly? (multiple answers allowed)

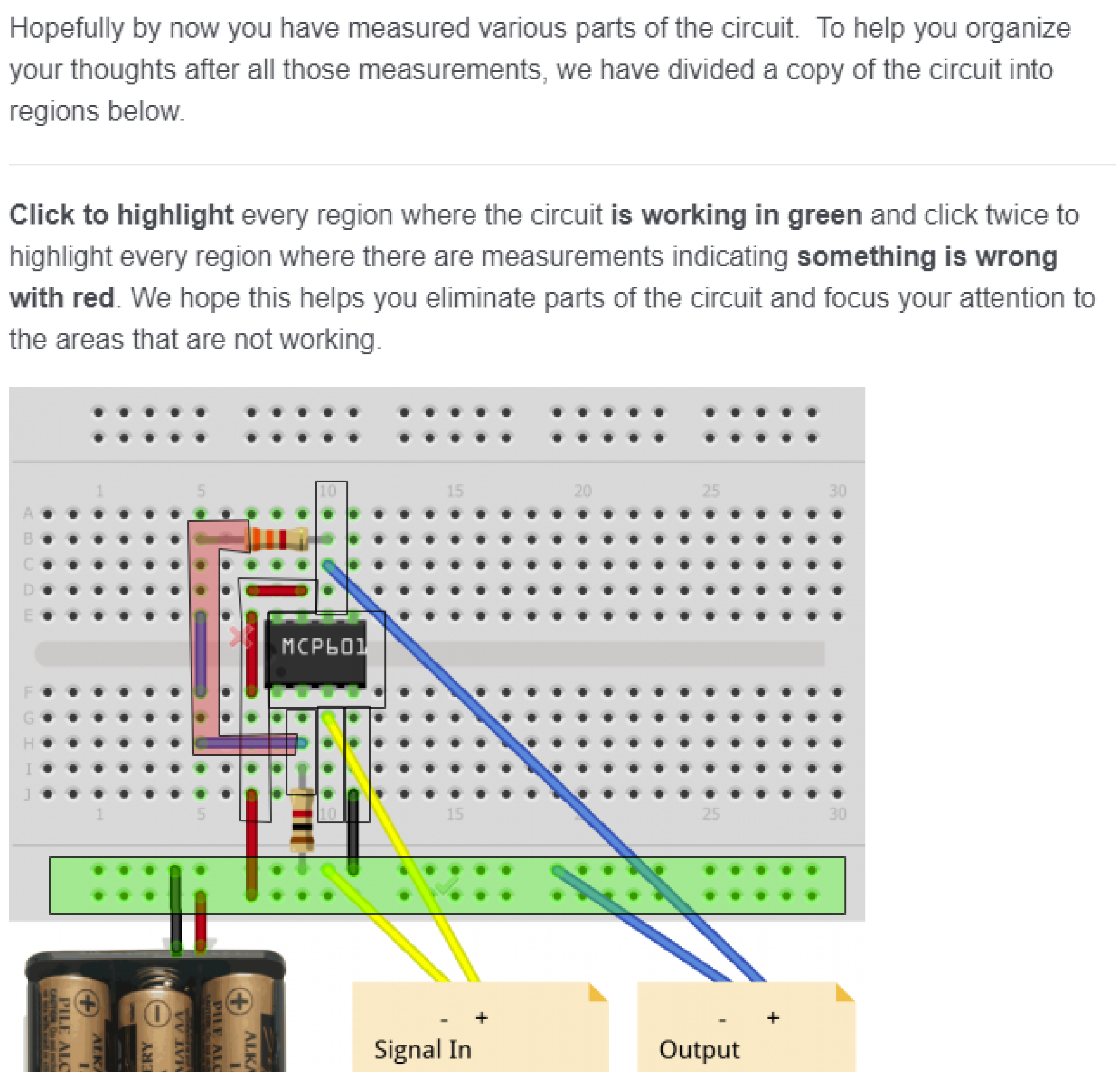
- ☐ The circuit amplifies
- ☐ It has the correct gain
- ☐ Checked the wiring and it was correct
- ☐ Checked various voltages and they all made sense
- ☐ The input and the output voltages are the same sign, which they should be for a non-inverting amplifier
- ☐ The op amp has negative feedback
- ☐ Everything is connected correctly to the op amp according to the pin out diagram
- ☐ The two inputs to the op amp are equal, as they should be according to the golden rules
- ☐ Something else

Figure B.26: Question about reasoning for students who said circuit was working for the non-inverting amplifier with a loose feedback wire. Answers are based off what students previously said in open-ended versions of debugging activity.

Once students figured out the circuit was not working properly, they were asked what measurements indicated there was a problem with the circuit. The choices were 1) the gain is wrong, 2) the output voltage is too large, 3) I don't need a measurement. I can see the circuit is wired wrong, 4) the voltages of the input terminals should be equal according to the Golden rules, 5) there is no voltage drop across the resistors, 6) the voltage supply is incorrect, or 7) something else.

In spring 2021, additional feedback was added for incorrect answers. If they chose a correct, but vague statement, they were given feedback that encouraged them to continue down this path, but get more details first. If they said the gain was wrong or the output voltage too large, they were given the feedback: "You are correct that the gain/output voltage is off, but now let's try to isolate the problem more specifically." If they noticed that the input terminals to the op amp were not equal, then they were given the feedback: "Yes, the input voltages to the op amp are unequal, meaning the Golden Rules are being violated and something is wrong with the circuit. Let's investigate a bit further to try and figure out what is causing this problem." If students chose a statement that was blatantly false, they were given feedback and told to try again. If they said the circuit was miswired, they were told, "You consult the TA and they also check the wiring. They say nothing looks miswired in the circuit and that you should keep searching for the problem." If they said the voltage supply was incorrect, they were given the following feedback: "The voltage for the 3-battery battery pack should be around 4.5-4.7V (1.5V for each battery). It is true that we used a power supply with +/-2.5V in lab 3, but according to the datasheet, you are also allowed to power the op amp with +5V and 0V. And actually, before we moved to remote learning and everyone getting their own AD2, the op amp used to be powered this way, so you could power it from your Arduino."

Besides the additional feedback about what measurements indicated something was wrong, the students in spring 2021 had an additional checkpoint to help them isolate the cause. If a student was already confident that they knew what was wrong with the circuit, they could skip this section. For the rest, they were shown a picture of the circuit with different regions boxed and asked to mark each region as either working or not working. We used the Qualtrics Heat Map question type for this (see Figure B.27).



Hopefully by now you have measured various parts of the circuit. To help you organize your thoughts after all those measurements, we have divided a copy of the circuit into regions below.

Click to highlight every region where the circuit **is working in green** and click twice to highlight every region where there are measurements indicating **something is wrong with red**. We hope this helps you eliminate parts of the circuit and focus your attention to the areas that are not working.

Figure B.27: **Isolate cause checkpoint for the amplifier with a loose feedback wire.** Students were asked to highlight different regions green if the measurements indicated they were working correctly and to mark the regions red if measurements indicated something was not right in that area of the circuit. Students had to highlight at least 5 regions to click "next". The computer would give immediate personalized feedback based on a student's answer. Students could only move on to the next question if they correctly highlighted the feedback loop as not working and did not mark any of the power supplies or input as not working.

Students were required to mark at least five areas as working or not working. If areas were incorrectly marked, they were given feedback, and if they failed to check the feedback loop, they were also given feedback. If they said the power supply rails had something wrong, then they were given the following feedback:

> *The bottom two rows of the breadboard are the power rails. All holes in the same row should have the same voltage. The top of those two rows should be Gnd and the bottom rail should be around 4.5V. Check whether this is true.*

If they said the paths connecting Vdd or Gnd to the op amp were incorrect, then they were given the following feedback:

> *The Vdd pin of the op amp should be connected to 4.5V. The Vss pin should be connected to 0V. Check whether this is true.*

If they said the connection from signal in to the op amp was wrong, we gave them the following feedback:

> *The voltage at the input pin Vin+ of the op amp should be same same as the voltage of the input wire, which is 0.5V. Check whether this is true.*

If they said the row of the negative input terminal of the op amp was working properly, they were given the feedback:

> *If the circuit was working properly, we would expect both input pins to the op amp to be 0.5V (because the input voltage is 5V and by the Golden Rules both input pins should be the same voltage). Check whether this is true.*

If they incorrectly marked the feedback loop as good, we told them:

> *The part of the feedback loop on the far left that you highlighted as working is all wires. What should the voltage difference be between any two points along a wire be? Check whether that holds true.*

If they did not mark the feedback loop either working or not, they were given the following feedback and told to keep investigating:

> *It looks like you haven't checked all the regions yet.*

Finally, if they said something was wrong with the op amp, we allowed them to move on, but they were given the feedback:

> *You clicked on the op amp indicating it is not working properly. While it is possible you were shipped a broken op amp, that is highly unlikely. A more likely possibility is that something is miswired (missing connections, wrong connections, etc.).*

As long as students marked that the feedback loop was not working and they had not incorrectly marked any other region, they were allowed to move on.

The final checkpoint required students to identify what was wrong with the circuit and how to fix it. Choosing whether the problem was with the power supply, the op amp, the resistors, the breadboard, a miswiring, or loose connections, led to sub-questions with more options.

If the student said the problem was with the power supply, they were given the following 3 options:

(a) The op amp is saturated. A larger power supply is needed.
(b) The op amp is powered incorrectly. (It needs +/-5V).
(c) The batteries need to be replaced.

If they said something was wrong with the op amp, they were given the following options:

(a) The op amp is broken.
(b) The op amp is connected incorrectly according to the pin out diagram.
(c) To properly build an amplifier, one input of the op amp should be connected to gnd.
(d) There is positive feedback instead of negative feedback.

If they said something is wrong with the resistors, they were given the following options:

(a) One or more resistors is broken.
(b) One or more resistors is the wrong resistance.
(c) One or more resistors is backward.
(d) The resistors are switched.

If they said the problem was with the breadboard, they were given the following options:

(a) The breadboard is broken and not electrically connected on the path to supply power to the op amp.
(b) The breadboard is broken and not electrically connected on the feedback path.
(c) The breadboard is broken and not electrically connected on the row of an input pin to the op amp.
(d) The breadboard is electrically connected horizontally, not vertically, in the middle section, so the components are connected incorrectly.

If they said the problem was loose wires, then they were given the following options:

(a) There is a loose wire at the input.
(b) There is a loose wire at gnd.
(c) There is a loose wire in the feedback loop.
(d) There is a missing wire in the circuit.
(e) Something is short-circuiting in the circuit.

If the student said the circuit was miswired, they were given the following options:

(a) One resistance is incorrect.

(b) The resistors are switched.

(c) The op amp is powered incorrectly. (It needs +/-5V).

(d) The op amp is saturated. A larger power supply is needed.

(e) The resistors currently aren't making a voltage divider.

(f) There is positive feedback instead of negative feedback.

(g) The op amp is wired incorrectly according to the pin out diagram.

(h) To properly build an amplifier, one input of the op amp should be connected to gnd.

(i) Something else, which I will specify on the next page.

The two answers that would let students exit were *there is a loose wire in the feedback loop* and *the breadboard was not electrically connected on the feedback loop.* If students chose the breadboard was not electrically connected on the feedback path, then they were allowed to exit the activity, but first they were given one final feedback and open-ended response. They were told:

> *It sounds like you correctly isolated where the problem is. You move the wires to a different row of the breadboard and it works! But, you used this breadboard last week and it worked fine (and it turns out the breadboard is still fine). What is an alternate explanation for what was wrong with the circuit?*

All other students were forced to iterate again and given the following message:

> *Not quite. Please investigate further. Click next when you are prepared to answer again.*

Appendix C

Course Syllabus

ENGR40M An Introduction to Making: What is EE?
Winter 2020
MWF 3:30-4:50 Lathrop 282

Welcome to ENGR 40M! This course provides an introduction to the broad field of electrical engineering through a series of hands-on projects. Countless devices use electronics, from cars to clocks to cameras to cell phones, but the way they work is usually hidden and often mysterious. Our objective is to demystify the world of electronics by tearing things apart (both literally and figuratively) so that you can understand how they work, and give you the skills to construct electronic devices of your own. We'll find that not only do many devices contain some electronics inside, but most of them contain small processors as well. Once you grasp the power of adding computing to physical devices and understand how a processor interfaces with other circuitry, you can use this knowledge to construct programmable electronic devices on your own. To exercise this building skill, during the quarter, you will build:

- A solar-powered cell phone charger, while learning about batteries, solar cells, power, and efficiency.
- A programmable "useless box", which is a silly toy to play with on your desk. While building this project, you will use switches, motors, transistors, digital logic, and learn to control physical things with software.
- An LED display, which uses the idea of multiplexing to control more lights than your microcontroller has outputs.
- An electrocardiogram (ECG) to measure your heartbeat. You will learn how to build an amplifier capable of magnifying the tiny electrical signal from your heart into something your microcontroller can measure.

One of first facts you will discover is that circuit rather work as expected immediately after you finish building them. Thus the class will focus on teaching you how to debug the circuits you build, so you can figure out what is wrong, and move to a working final result. To accomplish this, the class will both teach you the basic debugging approach, which requires you have a model of what should happen, and the basic tools you need to create these models. You will practice ways of modeling and reasoning about circuits in the homework and prelab assignments. By the end of the course, you will have models for simple analog and digital circuits, the practical skills for constructing, programming, and debugging electronic devices of your own, and the ability to explain some of the countless ways electronic circuits are used in the modern world. Specifically, you will be able to:

- Predict the behavior of electrical circuits containing resistors, capacitors, inductors, transistors, diodes, switches, and motors.
- Construct such circuits in the lab, and control or monitor them with software running on a microcontroller.
- Use good electronics construction skills to build circuits that are robust and easy to debug.
- Use lab equipment and a logical reasoning process to debug your circuits and code when they aren't working.
- Give examples of how the circuit elements and techniques from the course are used in real products.

E40M Schedule

				Topics	Assignments	Lab	Reader
	1	Monday	Jan 06	Course introduction; learning to look inside stuff; Basic physical properties in circuits: voltage, current and energy flow; KCL	Survey out		1-1.2
Week 1	2	Wednesday	Jan 08	Communicating circuits: devices, nodes and schematics; series parallel devices, equivalent circuits; Energy conservation: KVL			1.3-1.8
	3	Friday	Jan 10	Electrical device models: voltage and current sources, resistors; sign convention; introduction to a DMM; open and short circuits; solar cell device	Introductory survey due HW 1 released; Lab 1 released		2-2.4
	4	Monday	Jan 13	Visualizing device properties: i-V curves; power(sign convention review) building model of a solar cell: diodes +			2.5-2.6
Week 2	5	Wednesday	Jan 15	Solving circuits; nodal voltage and gnd; graphically; Thevenin equivalent circuit; solving diode circuits			3-3.1
	6	Friday	Jan 17	Solving more complex circuits: Nodal analysis; Circuit simplifications: series/parallel resistors	HW 1 due HW 2 released		3.2-3.3
		Monday	Jan 20	MLK day - Holiday			
Week 3	7	Wednesday	Jan 22	Circuit patterns:voltage/current dividers; solving by resistor combining; simulating circuit with everycircuit	Lab 2a released	Solar-powered cell phone charger	3.4-3.5
	8	Friday	Jan 24	Digital machines: FSM; Input/output devices: switches and motors; how to wire to your switches	HW 2 due HW 3 released		4-4.1
	9	Monday	Jan 27	Using breadboards; Intro to boolean logic; MOS transistors: electrically controllable switches			4.2-4.4
Week 4	10	Wednesday	Jan 29	MOS logic gates	Lab 2b released	Useless box	4.5
	11	Friday	Jan 31	Interfacing software and hardware: Building a smart useless box, Arduinos and inputs and outputs	HW3 due; Midterm Practice released		5-5.3.1
	12	Monday	Feb 03	Debugging: common software problems and overflow, isolating problems; Codes and coding			5.3.2-5.5
Week 5	13	Wednesday	Feb 05	Time multiplexing: How to control more LEDs than you have pins; driving your LED display			5.6
	14	Friday	Feb 07	Midterm	HW 4 released	No Lab	
	15	Monday	Feb 10	Sound: how speakers work, storing music on your computer; Transforms: what do we mean by tones			
Week 6	16	Wednesday	Feb 12	Capacitors: that take time to change voltage; RC time constant; delay in MOS circuits;	Lab 3a released	Computerized useless box	6-6.7
	17	Friday	Feb 14	Impedance: frequency dependent resistance, series and parallel capacitors; RC circuits; do prelab 3a	HW 4 due HW 5 released		7.3
		Monday	Feb 17	President's day - Holiday			
Week 7	18	Wednesday	Feb 19	Lab equipment: power supply, oscilloscope, signal generator; Filters: transfer function, how to approximate and guess which kind, inputs and outputs;	Lab 3b released	LED array: electronics and software	7.4
	19	Friday	Feb 21	Bode Plots: log/log plots, corner freq;dB; audio amp; prelab 3b	HW 5 due HW 6 released		7-7.2, 7.4
	20	Monday	Feb 24	Bioelectric signals; op-amps: intro, symbol; using feedback			8-8.1
Week 8	21	Wednesday	Feb 26	Op-amp circuits:inverting, non-inverting and differential amplifiers	Lab 4a released	LED array: displaying cool stuff	8.2-8.3.2
	22	Friday	Feb 28	Instrumentation Amps; Noise; Op-amp filters; prelab 4a	HW 6 due HW 7 released		8.3.3-8.5
	23	Monday	Mar 02	Inductors: take time to change current; Interesting circuits with inductors; L/R time constant			9.3
Week 9	24	Wednesday	Mar 04	Buck DC-DC converter	Lab 4b released	ECG: Analog front-end	9.4-9.5
	25	Friday	Mar 06	Boost DC-DC converter	HW 7 due Final Review released		9.6
	26	Monday	Mar 09	Class Review 1			
Week 10	27	Wednesday	Mar 11	Class Review 2		ECG: digital backend	
	28	Wednesday	Mar 13	Class Review 3			
			Mar 18	**Final Exam**, 7:00-10:00 PM			

Bibliography

[1] Dimitri Dounas-Frazer and Heather Lewandowski. Electronics lab instructors' approaches to troubleshooting instruction. *Phys. Rev. Phys. Educ. Res.*, 13, 2017.

[2] Dimitri Dounas-Frazer et al. Investigating the role of model-based reasoning while troubleshooting an electric circuit. *Phys. Rev. Phys. Educ. Res.*, 12, 2016.

[3] Norman Maier. Reasoning in humans. the solution of a problem and its appearance in consciousness. *Journal of Comparative Psychology*, 12(2):181–194, 1931.

[4] John Hayes. *The Complete Problem Solver.* Franklin Institute Press, 1981.

[5] George Polya. *How to Solve It: A New Aspect of Mathematical Method.* Princeton Science Library, 2nd edition, 1973.

[6] John Bransford and Barry Stein. *The ideal problem solver : a guide for improving thinking, learning, and creativity.* W.H. Freeman, 1984.

[7] Craig Ross and Robert Orr. Teaching structured troubleshooting: Integrating a standard methodology into an information technology program. *Educational Technology Research and Development*, 57(2):251–265, 2009.

[8] Shima Salehi. *Improving Problem-Solving through Reflection.* PhD thesis, Stanford University, 2018.

[9] David Jonassen and Woei Hung. Learning to troubleshoot: A new theory-based design architecture. *Educational Psychology Review*, 18(1):77–114, 2006.

[10] David Rumelhart. *Theoretical Issues in Reading Comprehension*, chapter Schemata: The Building Blocks of Cognition. Routledge, 1980.

[11] William Mostia. *Troubleshooting: A Technician's Guide.* ISA Technician Series. 2nd edition, 2006.

[12] Daniel Tomal and Aram Agajanian. *Electronic Troubleshooting.* McGraw-Hill, 4th edition, 2014.

[13] Argenta Price et al. A detailed characterization of the expert problem-solving process in science and engineering; guidance for teaching and assessment. *CBE–Life Sciences Education.* Submitted April 26, 2021. Under Review.

[14] Gary Klein. *Sources of Power: How People Make Decisions.* MIT Press, 1998.

[15] Kathleen Mosier et al. *The Cambridge Handbook of Expertise and Expert Performance*, 2018.

[16] Randall Davis and Walter Hamscher. *Exploring Artificial Intelligence*, chapter Model-Based Reasoning: Troubleshooting. Morgan Kaufman Publishers, 1988.

[17] Beth Crandall et al. *Working Minds: A Practitioner's Guide to Cognitive Task Analysis.* MIT Press, 2006.

[18] Caroline Zsambok and Gary Klein, editors. *Naturalistic Decision Making.* Expertise: Research and Applications. Lawrence Erlbaum Associates, 1996.

[19] Michelene Chi et al. Characterization and representations of physics problems by experts and novices. *Cognitive Science*, 5:121–152, 1981.

[20] Cynthia Atman et al. Engineering design processes: A comparison of students and expert practitioners. *Journal of Engineering Education*, 96:359–379, 2007.

[21] Adriann de Groot. *Thought and Choice in Chess.* 1965.

[22] George Miller. The magical number seven plus or minus two: some limits on our capacity for processing information. *Psychological Review*, 63(2):81–97, 1956.

[23] John Sweller. Models of competence in solving physics problems. *Cognitive Science*, 4:317–345, 1980.

[24] John Sweller. Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12:257–285, 1988.

[25] John Bransford and Daniel Schwartz. Rethinking transfer: A simple proposal with multiple implications. *Review of Research in Education*, 24:61–100, 1999.

[26] Mary Gick and Keith Holyoak. Schema induction and analogical transfer. *Cognitive Science*, 15:1–38, 1983.

[27] Dan Schwartz et al. *The ABCs of How We Learn.* W. W. Norton Company, 2016.

[28] Laura Rios et al. Using think-aloud interviews to characterize model-based reasoning in electronics for a laboratory course assessment. *Phys. Rev. Phys. Educ. Res.*, 15, 2019.

[29] Heather Lewandowski at University of Colorado-Boulder. https://jila.colorado.edu/lewandowski/research/maple-instructors.

[30] Video of a Useless Box by Think Geek. `https://www.youtube.com/watch?v=aqAUmgE3WyM`.

[31] Steven Bell and Mark Horowitz. Rethinking non-major circuits pedagogy for improved motivation. *2018 ASEE Annual Conference Exposition*, Salt Lake City, UT June 24-27, 2018.

[32] Analog Discovery 2 by Digilent. `https://store.digilentinc.com/analog-discovery-2-100msps-usb-oscilloscope-logic-analyzer-and-variable-power-supply/`.

[33] Roman Taraban et al. Using paper-and-pencil solutions to assess problem solving skill. *Journal of Engineering Education*, 100(3):498–519, 2011.

[34] Ali Malik et al. Generative grading: Neural approximate parsing for automated student feedback. *33rd Conference on Neural Information Processing Systems*, Vancouver, Canada December 8-14, 2019.

[35] Wendy Adams and Carl Wieman. Development and validation of instruments to measure learning of expert-like thinking. *International Journal of Science Education*, 33(9):1289–1312, 2011.

[36] Paul Denny et al. Evaluating a new exam question: Parsons problems. *ICER '08*, Sydney, Australia September 6-7, 2008.

[37] Circuits on Tinkercad® by Autodesk®. `https://www.tinkercad.com/learn/circuits`.

[38] Edison: Photorealistic Multimedia Lab for Exploring Electricity and Inc. Electronics by DesignWare. `http://www.designwareinc.com/edison.htm`.

[39] Fritzing. `https://fritzing.org/`.

[40] Spice general purpose circuit-simulator program by UC Berkeley. `http://bwrcs.eecs.berkeley.edu/Classes/IcBook/SPICE/`.

[41] LTspice Simulator by Analog Devices. `https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html`.

[42] Qualtrics®. `https://www.qualtrics.com/`.

[43] The Graphviz package for Python based on DOT language. `https://www.graphviz.org/doc/info/lang.html`.