Heimdall

A Remotely Controlled Inspection Workbench For Debugging Microcontroller Projects

Mitchell Karchemsky
UC Berkeley
Berkeley, California
mkarch@berkeley.edu

J.D. Zamfirescu-Pereira
Cornell Tech
New York, New York
jdz32@cornell.edu

Kuan-Ju Wu
UC Berkeley
Berkeley, California
kuanju@berkeley.edu

François Guimbretière
Cornell University
Ithaca, New York
fvg3@cornell.edu

Bjoern Hartmann
UC Berkeley
Berkeley, California
bjoern@eecs.berkeley.edu

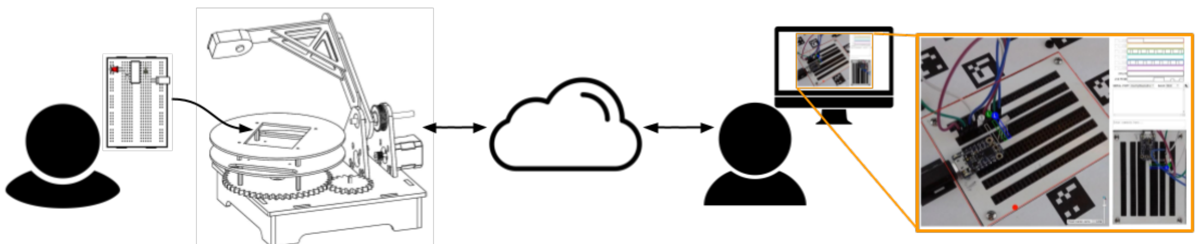


Figure 1: A student brings their breadboarded circuit to the Heimdall debugging station, where measurement and injection circuitry automatically connect to the board, and an image set is captured by a robotic gantry. A remote expert then uses a web-based interface to visually inspect the circuit, take measurements, reconfigure connectivity and inject signals to debug the problem.

ABSTRACT

Students and hobbyists build embedded systems that combine sensing, actuation and microcontrollers on solderless breadboards. To help students debug such circuits, experienced teachers apply visual inspection, targeted measurements, and circuit modifications to diagnose and localize the problem(s). However, experienced helpers may not always be available to review student projects in person. To enable remote debugging of circuit problems, we introduce Heimdall, a remote electronics workbench that allows experts to visually inspect a student's circuit; perform measurements; and to re-wire and inject test signals. These interactions are enabled by an actuated inspection camera; an augmented breadboard that enables flexible configuration of row connectivity and measurement/injection lines; and a web-based UI that teachers can use to perform measurements through interaction with the captured images. We demonstrate that common issues arising in embedded electronics classes can be successfully diagnosed remotely and report on preliminary user feedback from teaching assistants who frequently debug circuits.

CCS CONCEPTS

• **Human-centered computing** *Interactive systems and tools*.

KEYWORDS

electronics; embedded systems; remote debugging

ACM Reference Format:
Mitchell Karchemsky, J.D. Zamfirescu-Pereira, Kuan-Ju Wu, François Guimbretière, and Bjoern Hartmann. 2019. Heimdall: A Remotely Controlled Inspection Workbench For Debugging Microcontroller Projects. In *CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019), May 4–9, 2019, Glasgow, Scotland UK*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3290605.3300728

CHI 2019, May 4–9, 2019, Glasgow, Scotland UK

ACM ISBN 978-1-4503-5970-2/19/05.
https://doi.org/10.1145/3290605.3300728

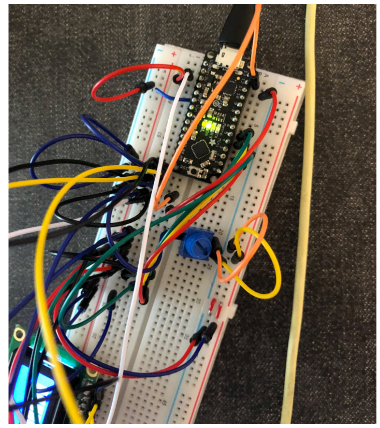
1 INTRODUCTION

As the Internet of Things (IoT), Cyber-Physical Systems (CPS) and related technologies grow in importance, educating students in core topics in electronics, circuits, and embedded system design is becoming increasingly relevant. Traditionally, such electronics courses are taught in special hands-on laboratories dedicated to this purpose, with teaching staff available to provide assistance in real-time during lab sessions. Recently, several factors are driving electronics teaching away from this centralized, real-time assistance model. First, the growth of the maker movement and makerspaces has led to an increase in informal learning, where students and hobbyists build projects outside of a formal classroom setting, and often after regular hours. Second, the growth of online education (such as MOOCs) is characterized by geographical separation of students and teachers. Taken together, these trends highlight a need for online, remote electronics teaching and mentoring. While lectures and reference materials can be easily delivered online, a large gulf remains for teachers: it is difficult to view and understand a student's remote work, in order to provide feedback and debugging advice.

Our research goal is to allow instructors to remotely and accurately inspect a student's circuit both visually and electrically. We focus on the setting of debugging breadboarded circuits, since the solderless breadboard is a ubiquitous substrate for quickly prototyping electronic devices. Recently, the HCI research community has introduced several "smart" breadboards that deliver additional functionality to aid debugging, such as voltage sensing [10], current visualization [30] and automatic component detection [31]. These are aimed at users who have physical access to the circuit in question. While they could also be utilized to perform remote measurements, they don't address a number of additional important challenges posed by remote debugging:

Difficulty of Visual Inspection: Circuits with many components and wires are complex and difficult to understand visually; a single photograph (e.g. Fig. 2) is almost always insufficient to determine whether components are misplaced or misaligned, due to occlusion, orientation, or parallax.
Inability to Isolate: Testing a bug hypothesis often requires isolating and testing components individually, which requires changing the topology of the circuit (e.g. removing wires).
Inability to Actively Test: Testing often requires driving a circuit in a certain way; providing input signals or protocol messages to test behavior on demand.

Each of these challenges is due to the remote nature of the task. In person, an instructor can rotate the breadboard fully in her hands, place measurement probes at any point, and physically remove and replace electronic components. Today, in debugging remotely (via forum posts or video calls), all these actions are mediated by a student who is only just learning enough to be able to follow the directions of the instructor. This leads to significant errors and miscommunication that increases debugging time and decreases debugging success. An alternative is to simulate circuits and use virtual tools (e.g., Tinkercad [4]). However, simulations may be limited in fidelity and do not teach the embodied skills of working with electronics. This is especially true in classes that focus on hands-on, open ended projects with sensors and actuators. In these classes, physical prototyping on breadboards remains the method of choice.



From: student@
To: teaching-team@
Subject: Help!
Date: Mon, 11:48pm

Today/tonight is crunch time on the lab. But unfortunately I can't get the hello world to print out....

I wondered if it's related to my inputs?

Figure 2: A real and representative example of a request from a student seeking debugging help with a lab assignment.

To address the challenges of remote breadboard debugging, we introduce Heimdall, a remotely controlled inspection workbench for debugging microcontroller projects. Heimdall enables effective remote work through three core functions:

Visual Inspection through Robotic Gantry: A two-degree of freedom robotic gantry pre-captures a static image data set of a circuit that a remote expert can quickly navigate; any perspective can also be viewed as a live video feed.
On-Image Measurement via Instrumented Breadboard: Our augmented breadboard allows instructors to measure digital or analog voltages on rows by selecting any breadboard row in an image of the remote circuit.
Active Testing through Rewiring and Injection: The breadboard enables remote experts to break the usual breadboard connectivity at any row, and to inject analog or digital signals into any selected row.

We demonstrate how these functions can be used to debug common problems faced by students and report on initial feedback from four expert users who debugged two circuits each with Heimdall.

2 RELATED WORK

Our work is informed by prior research on troubleshooting and debugging in the learning sciences, and extends prior technical contributions in circuit debugging tools and remote laboratories.

Troubleshooting and Debugging from an Educational and Psychological Perspective

How students learn to debug and troubleshoot has been investigated most thoroughly for software, e.g., in [1, 2, 18]. Debugging is "both difficult for novice programmers to learn and challenging for computer science educators to teach." [18] Several existing classifications of types of bugs are particular to software [25], though Ko and Myers contribute a more general framework of cognitive breakdowns that can transcend domain-specific details [15]. One important insight is that novices have trouble forming correct hypotheses about the causes of unexpected behavior — better tools can help by supporting them asking "why" and "why not" questions [16].

Some studies specifically focus on electronics debugging by novices. Gitomer investigated differences in troubleshooting strategies, finding that novices' mental models of their device were misaligned with the actual functionality of the device [11]. This implies a need for effective tools to communicate the proper model of a device to a student. Booth et al. conducted a study of a physical computing task that involved both circuit construction and programming. A key result was that "most fatal faults were due to incorrect circuit construction, and that often problems were wrongly diagnosed as program bugs." [5] Mellis et al. report on workshops that engaged amateurs in circuit board design, finding — similarly to software — that participants had difficulty formulating hypotheses, but with the added complication that it was unclear whether a problem was mechanical (loose wire), related to a component (defective, wrong polarity), or related to their circuit design [20]. This implies that tools should help students narrow the type of problem they are experiencing. Park and Gittelman conducted experiments on different instructional techniques to help students with electronics troubleshooting, finding that animated visual displays and feedback can be effective strategies [7]. Jonassen and Hung describe a domain-agnostic approach to developing troubleshooting skills in learners. Notably, a core component of their approach is to review what experienced troubleshooters would do; this can be achieved either through a case library of existing examples, or, as in our research, by facilitating an asynchronous exchange with an expert [14].

Debugging Tools for Electronic Circuits

Recently, several technologies that focus on supporting users with learning and debugging electronic circuits have emerged. One core distinction is between self-contained toolkits that teach electronics concepts with a limited set of modular parts (e.g., LightUp [3], Flow of Electrons [8]) and tools that work with standard electronic circuits and components. A key direction has been to develop augmented solderless breadboards, as many students start building circuits on breadboards. Augmented breadboards can measure and visualize voltages on each row [10], current flow [30], and can partially detect what components have been inserted through active probing [31]. Finally, many student projects in embedded and physical computing live at the intersection between software and hardware. Therefore, debugging tools that enable students to understand interactions between these realms are important. The Bifröst system combines code instrumentation and logic analyzer circuit tracing for this purpose [19]. With Heimdall, we extend the smart breadboard approach to allow for remote troubleshooting. A key novelty is Heimdall's ability to enable teachers to change circuit topology to narrow measurements to sub-circuits without having physical access to the board.

Remote Electronics Laboratories and Simulations

Two important developments in enabling online education in electronics have been the use of circuit simulations and remote electronics laboratories. Both enable students with a computer but no direct physical access to electronics equipment to learn important concepts and skills. Simulation environments go back at least to the 1970s [6]. Simulation and visualization are now available in commercial products aimed at novices and students, e.g., Autodesk Tinkercad [4]. While simulation can teach concepts effectively, it cannot provide the embodied skills; in addition, simulation is limited by the expressivity of the underlying models and cannot easily handle systems where complex real-time sensor input is needed to test functionality. To overcome the second limitation, remote laboratories provide internet access to actual hardware on which students can perform experiments [12, 17, 24, 27]. Cooper discusses challenges in their adoption in practice [9]. Some implementations only provide pre-designed circuits for specific lab exercises, while more ambitious projects include the ability for students to re-wire circuits remotely using a relay switching matrix, as in VISIR [27]. Heimdall is complementary and orthogonal to this body of literature, focusing on the case where students build functional physical prototype devices, but the teacher is remote, instead of the student.

3 DESIGN RATIONALE

The design of our system was informed by our experiences teaching introductory physical computing and Internet of Things microcontroller courses in our respective institutions. Student projects typically consist of connecting a microcontroller (MCU) to basic electronic components (e.g., LEDs, servos, etc.), sensors (buttons, potentiometers, photo cell, etc.), and other ICs with digital interfaces (I2C/SPI/UART). Throughout class time, instructors are asked to debug circuits of different complexity and familiarity, often under time pressure (e.g., debugging embedded systems circuits during a lab section). The typical steps for circuit debugging are:

(1) Visually inspect to identify simple errors. Is the power properly connected, or are the wires off by one row? Are parts with similar package types confused, e.g., was the voltage regulator mistaken for a power transistor? This step identifies a surprisingly large number of errors, but requires the ability to quickly observe the breadboard closely and from different angles.
(2) Use a multi-meter to quickly measure key voltages across the board. Is *Vcc* at the expected level? Are the pull-up resistors doing their job? Requiring only a multi-meter, this step can be performed quickly with minimum circuit modification.
(3) Analysis of each part of the circuit in depth. This step often requires isolating some part of a circuit, modifying the circuit, or using sophisticated tools such as a logic analyzer to identify the activity of a serial bus or to inject arbitrary signals. Experienced debuggers may also use the serial console, available on many modern MCUs, to understand the MCU's internal state.

To translate these practices into a remote debugging setting, we designed a system that:

(1) Enables the remote visual inspection of the board, using a robotic gantry system that captures hundreds of unique views of the board at different angles and elevations before a remote debugging session starts. A custom UI allows the remote expert to quickly navigate through these points of view as if they were manipulating a remote camera directly, enabling a smooth and efficient visual inspection. The remote user can switch to a live video feed at the given perspective to observe real-time events, such as the frequency of a blinking light.
(2) Lets the remote expert perform voltage measurements directly on the image of the breadboard, emulating the multi-meter inspection they would perform in person. This enables measurements in the context of the circuit and overall awareness of the current state of the circuit.
(3) Supports remote modification of the circuit for deeper analysis. The unpredictable structure and layout of students' breadboarded circuits make remote modification (without literally moving wires) challenging; instead, our system enables modifications through isolating a portion of a breadboard row and injecting a new signal in the isolated section. The isolation is implemented using a row-splitting feature that disconnects the two peripheral connections from the three central connections of each breadboard row (see Fig. 5a).
(4) Lets the remote instructor interact directly with the serial console of the microcontroller, helping the remote expert understand the internal computational state of the system.

To minimize cost, we centralize all electronics for measurement, circuit reconfiguration and signal injection in a single debugging workbench. Students build their projects on breadboards that use the CircuitStack [28] system with two layers: one with headers to receive components, and a second layer that defines the usual breadboard connectivity. When students want to debug a circuit, they remove their backplane and place their circuit in the Heimdall station, where a different backplane connects their circuit to our sensing infrastructure.

Table 1: A Summary of Features and Use Examples of Heimdall

Feature	Usage Example
Split connectivity of a single row	Isolate a sensor output from a microcontroller input pin
Inject digital signal to a row	Simulate a HIGH or LOW signal as from a button
Inject analog signal to a row	Input arbitrary 0-3V signal to a row to simulate a sensor
Voltage measurement of 7 simultaneous rows	Verify output voltage at pin to ensure an LED is getting power
Fast visual inspection of breadboard, 360°	Identify misplaced components/wires; identify resistor values and ICs
Live visual updates	Determine whether split rows or injected voltages change LED state

4 REMOTE DEBUGGING WITH HEIMDALL

Given the design rationale described (summarized in Table 1) above, Heimdall is well suited for debugging problems typical of introductory physical computing and Internet of Things courses. Typical problems in those courses that Heimdall can identify include:

(1) Incorrect component selection, e.g., resistor values
(2) Mispositioned components
(3) Electrically broken wiring that cannot be seen
(4) Misbehaving analog or digital (SPI/I2C/UART) sensors
(5) Incorrect MCU behavior

In this section, we describe the user experience of debugging with Heimdall and illustrate how the technical contributions of Heimdall enable the various debugging techniques described previously. We introduce a running example to ground the description in concrete, common problems encountered by students: *Joyce is teaching an embedded systems prototyping class but is currently traveling to attend UIST. Students Annabel, Bella, and Chris are working on an assignment which requires a multicolor LED to pulse on and off at a frequency set by a potentiometer, using a color configured by serial messages. They each run into different issues that they cannot resolve, and ask Joyce for help.*

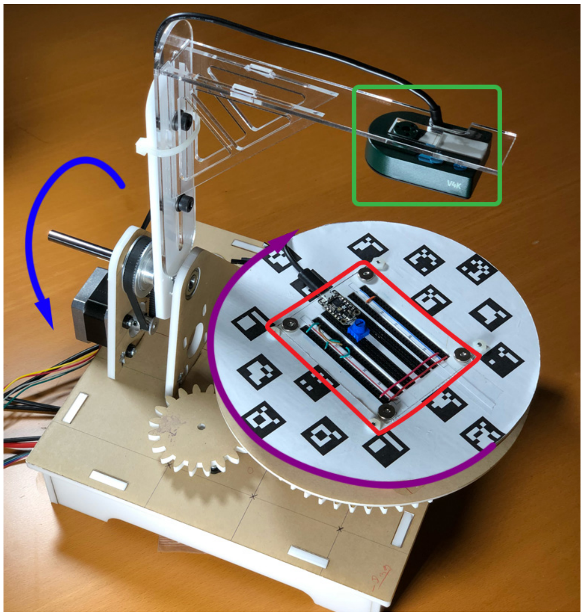


Figure 3: Two rotational axes provided by the debugging station marked in purple and blue. This allows for the camera(green) to have full rotation view of the student breadboard(red).

Capture and Visual Inspection

Annabel prototyped her circuit on a Heimdall-compatible breadboard with a removable backplane. Her LED is not turning on, even though the code and circuit look correct to her. To ask her teacher for help, she removes the backplane and places her breadboard into the Heimdall debugging station.

Heimdall makes electrical connections to the breadboard and then automatically captures and caches a set of 156 still images of the student circuit . These photos comprise a 360° view of the circuit board at four elevations.

Joyce, on a break between conference sessions, can open up the remote, web-based user interface, and quickly explore the circuit's wiring and topology visually by panning and orbiting through the image set (see Fig. 4). She can also zoom in for reading small details such as resistor color codes and part numbers on capacitors, transistors, and integrated circuits. For further investigation, she can toggle from the pre-captured set to a live view to check the current state of the LED in the circuit. As Joyce pans Annabel's breadboard she notices that the wire which connects the output of the microcontroller's built-in voltage regulator to the breadboard's power rail has been misplaced by one row and does not connect to the Vcc pin on the microcontroller. She sends this hint back to Annabel.

Measurement

Bella has verified that her wiring is correct, but also has problems getting the intended behavior from her circuit. She simplified her code to turn all colors of the RGB LED on, but even this does not work. After placing her project into the

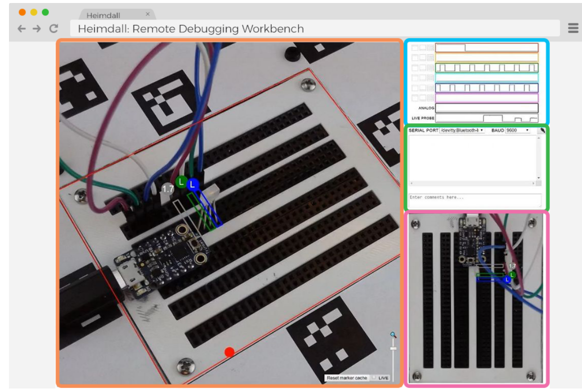


Figure 4: The Heimdall User Interface. Orange: Pre-cached Image explorer with three rows instrumented for signal measurement. Blue: Signal-Time graphs for all inspection probes. Green: Serial Monitor. Pink: Fixed overhead view.

Heimdall station, Joyce uses the remote interface to perform different measurements to characterize the problem.

Voltage levels on each breadboard row can be read via a configurable logic analyzer which provides both digital or analog values. The remote interface is based on a direct manipulation metaphor [13]: Joyce can directly interact with the image of the remote breadboard circuit in order to configure and perform measurements. Hovering over a row configures the augmented breadboard to connect that row to a programmable logic analyzer and display the resulting measurements at the same location the teacher hovered (see Fig. 5) as well as in the LIVE PROBE "scope" view (see Fig. 6); clicking on the row lets Joyce select a dedicated measuring channel to be displayed permanently next to the row and in the scope view.

Approximately 15 seconds of samples are displayed on the scope views, allowing Joyce to see the recent state of all 8 inspected pins: high or low for the digital pins, and voltage values on the analog pins (see Fig. 6).

Joyce selects the three signal lines going to the RGB LED from the microcontroller by directly clicking on the respective breadboard rows in the current image as shown in Fig. 5(b). In the scope view, she notices that the microcontroller is sending HIGH signals to the LED. She suspects that Annabel was assuming that the component she had was a common cathode LED (where sending HIGH would turn the LED on), when it might be a common anode LED (where LOW would turn the LED on).

Electrical Isolation and Injection

Chris has a problem with the blink rate of his LED. When he turns the potentiometer, the LED jumps from not illuminating to blinking too quickly to see. Joyce has the hypothesis that the student may have picked a log-scale potentiometer (often used for audio applications) instead of a linear scale

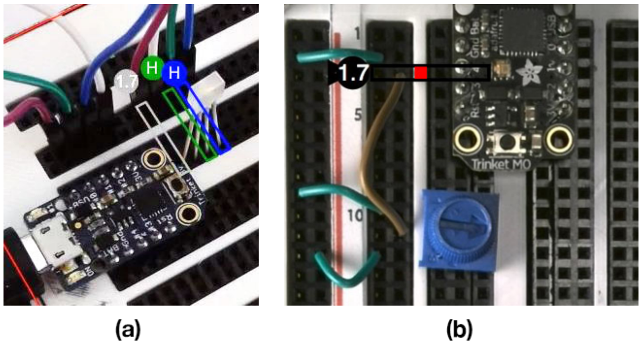


Figure 5: (a) Instantaneous measurements are displayed directly adjacent to the breadboard row; "H" is digital HIGH, numbers are analog voltages. (b) Left column row 3 is split as indicated by a red marker, isolating the blue potentiometer (row 11) from the microcontroller.

potentiometer. To test if the student circuit would work correctly with a linear analog input, she would like to replace the potentiometer input with an input under her control, while observing the LED in the live video stream mode.

A key contribution of Heimdall is the ability to remotely manipulate the topology of the student circuit. The inner three columns of each row of the breadboard can be programmatically connected or disconnected from the outer two columns. This is made possible by using relays underneath the breadboard to connect or isolate the columns (see Fig. 7)

In Chris's circuit, the microcontroller is on an inner column of the breadboard, while the wire from the potentiometer is on an outer column. Joyce clicks on the space between inner and outer column on the analog input to disconnect them as shown in Figure 5; the UI shows a red square to indicate this status.

Joyce then configures this same row to receive an injected signal. To do this, she clicks on the row she wants to inject with the analog signal, and selects the analog scope connection. Then, in the scope interface, she moves her mouse between the top (3.3V) and bottom (0V) of the injection button to select a voltage, and clicks to enable injection as shown in Figure 6. The selected voltage is injected by (1) configuring the analog crossbar underneath the breadboard to link the selected row to a DAC (digital-to-analog converter) on the same microcontroller used to coordinate row splitting and crossbar configuration, and (2) sending that DAC the voltage selected in the user interface. Joyce can now see that the blink rate does change smoothly in the video view and sends a message to Chris to replace the potentiometer or convert the readings in code from log to linear scale.

5 IMPLEMENTATION

Heimdall incorporates both software and hardware instrumentation in order to capture visual and electrical information from a student's circuit. In this section, we describe the

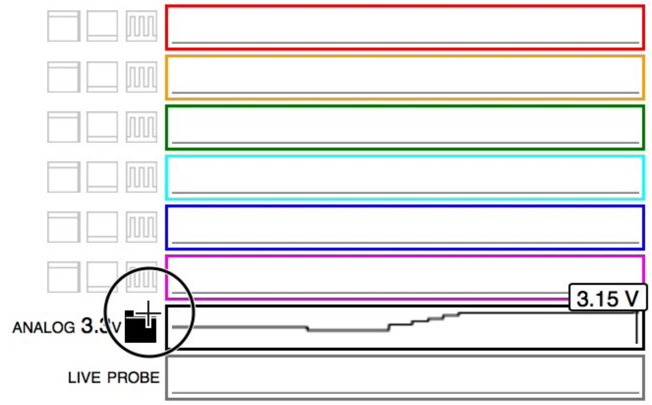


Figure 6: Scope Interface. The first six rows show digital signals over time; To the left of the digital scopes are options to inject signals (HIGH, LOW) or a 1hz square wave. The black "ANALOG" row can either show or inject analog voltages (here, an analog voltage of 3.3 is being injected). The last row reflects the live probe measurements over time.

high-level design of the system (Fig. 7) as well as concrete implementation choices for our prototype.

Visual Inspection

Our inspection station is comprised of laser-cut acrylic, stepper motors, and a camera. The platform consists of two rotary axes (Fig. 3): one allows the instrumented breadboard to rotate; the other pans the camera arm about the center of the platform. We perform an automated capture of high resolution photos at 4 different heights as well as complete rotation around the board. To capture the photos, we use an IPEVO Document Camera which captures 8-megapixel still images of the board and 1440x1080 resolution in live-viewing.

Locating the Breadboard in Images

The system needs to precisely locate the board in captured images at millimeter resolution, since adjacent breadboard rows are spaced 0.1" (2.54mm) apart. We place printed ArUco markers [23] on the same plane as the top of the student's board. We then locate the markers at the corners of the breadboard via OpenCV's ArUco library and compute a perspective transformation to convert pixel coordinates to breadboard coordinates. This transform enables the expert to directly select breadboard rows in both overhead and angled images.

We found that in many different perspectives, wires and other components often occlude part of the four corner markers indicating the corners of the breadboard. To address this, we place additional ArUco markers at greater distances from the board. If any or all of the four immediate-corner markers are not visible, we use other nearby markers to construct a set of four fully-visible markers and extrapolate the breadboard location from those.

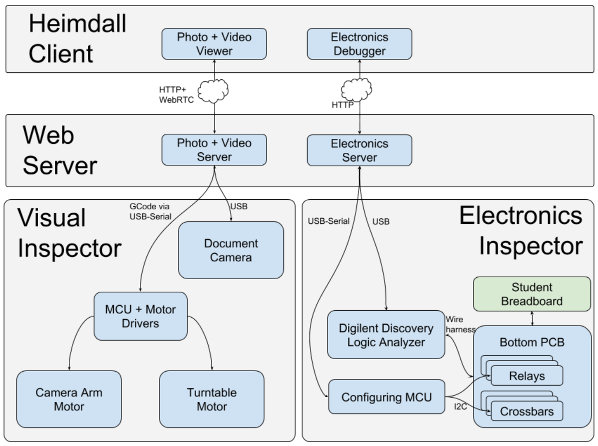


Figure 7: A high level diagram of the Heimdall system

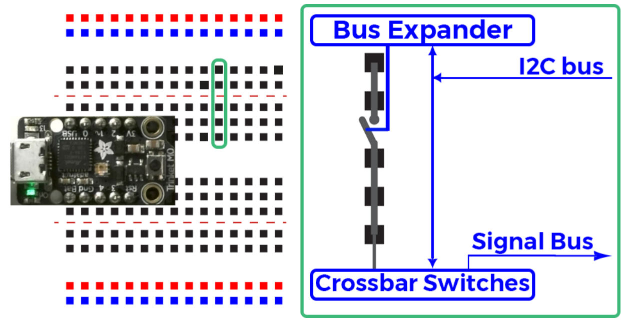


Figure 8: Backplane design. Computer controlled relays let us isolate the central part of the breadboard rows. Crossbar switches are used to perform measurements or inject signals. Both switches and crossbars can be controlled through via I2C.

Student Breadboard

Our approach for designing an affordable, scalable system was influenced by the CircuitStack breadboard system [28]. The main difference in Heimdall is that each breadboard row is split into two (both electrically and visually as shown Fig. 5). Like in CircuitStack, each student receives two halves that are assembled to form a standard breadboard, a "top" and a "bottom". The top half functions as the insertion points for a breadboard; The bottom half is a passive PCB which connects the rows of the top part in a standard breadboard configuration. When the student wants to use the Heimdall system, they can remove the top portion of their breadboard and insert it into the debugging station. The debugging station houses all of the electronics for instrumenting the students board.

Instrumentation PCB

We designed our breadboard instrumentation as a separate PCB that remains in the inspection station that replaces the student passive bottom PCB. Figure 8 shows the concept and Figure 9 shows our implementation.

We use one Omron G3VM-61VR normally open MOSFET relay for each row to implement the isolation mechanism. We picked this component for its high current capability, its low ON resistance (0.25Ω), and low driving current (3mA trigger for a total of 180mA). We also considered a normally closed alternative to avoid having to drive the relays all the time, but the high ON resistance was judged unacceptable for our application. Each relay is in turn controlled by one of our four MAX7314 I2C-controlled port extenders.

To implement our measurement system, we used 5 Analog Devices ADG2128 I2C-controlled 8x12 analog switch arrays to connect the inner three columns of each row to one of our 8 analog bus lines. The same system can be used in reverse to inject signals to the board as needed. While the system could also be used to connect any row to any other row, the high resistance of the connections and the power dissipation limit of the chip means that this functionality would have to be used with care. The nine chips (4 MAX7314 bus extenders and 5 ADG2128 analog switches) are all connected to the same I2C bus controlled by a RedBear STM32 microcontroller connected to a host computer via a USB connection. While our PCB design is complete for an entire breadboard with 30 rows, in our prototype we only hand-soldered components for the first 12 rows of the board. An entire board could be quickly produced using an industrial pick-and-place process.

Electrical Signal Inspection and Injection

The Instrumentation PCB described above is controlled by an Arduino-like microcontroller running a purpose-built command parser which interfaces with the web server backend described later. The command set includes commands to connect any specific row on the breadboard to one of the 8 analog bus lines described above, and commands to split individual rows' middle three and peripheral two pins. Communication between the instrumentation PCB and the microcontroller occurs over an I2C bus.

To perform digital and analog signal inspection and digital signal injection, Heimdall relies on a Digilent Analog Discovery 2 (AD2). The AD2 is a logic analyzer with 16 digital I/O pins and 2 analog input pins backed by a 14-bit ADC. The eight analog bus lines on the Instrumentation PCB are individually connected to the AD2 on six of the digital I/O pins and the two analog input pins. This allows software-controlled sampling of up to eight arbitrary breadboard rows at a time (six digital, two analog).

For signal injection, Heimdall uses two systems. The AD2's digital pins are used to inject logic-level signals into the bus lines, and via the crossbar switches, into any row on the instrumented breadboard. To inject arbitrary analog voltages, Heimdall uses the 10-bit DAC integrated into the Instrumentation PCB control microcontroller.

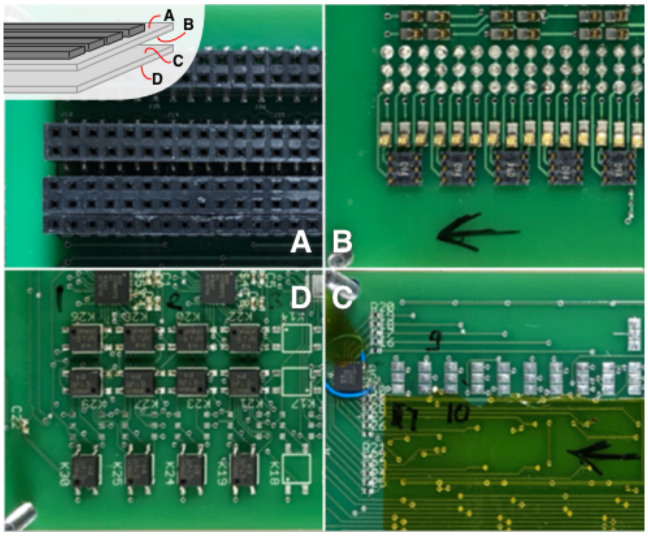


Figure 9: The two PCBs in Heimdall: student breadboard (top) and measuring board (bottom). A: top of student board featuring female headers; B: bottom of the student board has spring-loaded contacts to connect to the measuring board; C: top of the measuring board showing the bus extender; D: bottom of the measuring board showing 2 ADG2128 crossbar switches and Omron G3VM-61VR relays. Our prototype used only 12 rows.

Web-based User Interface

Remote instructors interact with Heimdall entirely through a web-based interface, which is implemented as a single-page app built using the React web framework, and served via a custom Node.js-based server. The system uses WebSockets for real-time communication between the server components and the browser. For performance, the 156 images of the breadboard at various angles are pre-loaded into browser memory and displayed on demand, resulting in lag-free panning. Samples from the logic analyzer and analog scope are collected at 100Hz by a python server and transmitted via WebSocket connection as they are received.

The Node.js server additionally mediates communication between the browser-based UI and the command parser running on the Instrumentation PCB control microcontroller, using a REST interface.

The Node.js server also serves a special local web page that connects with the document camera and makes it available via WebRTC to enable the live view mechanism. The remote instructor page connects to the local page to create a low-latency WebRTC connection and its live video is displayed on demand.

The system supports multiple users viewing the Heimdall web interface simultaneously. Multiple users can see synchronized views of probed, scoped, and split rows. Changes made by one user are reflected in the interface shown to all other users, enabling Heimdall's simultaneous use by instructor and student with an out-of-band audio signal for more effective debugging.

Heimdall can be used asynchronously. This means that a student can leave their circuit in the Heimdall system and allow for the instructor to independently inspect the circuit when ready. We provide the mechanism for synchronized and asynchronized viewing; but we leave instructors to decide on the particular policy. Future work can address how to seamlessly include findings into comprehensive feedback and hints delivered to the students. Ideas to build on for such functionality include Electrotutor [29] and TraceDiff [26].

Additional Functionality

In addition to digital and analog measurements, Heimdall also captures and relays serial output from the student's MCU, which is shown in the remote UI underneath the scope interface. Serial messages can also be sent from the remote UI to the student's MCU, which enables remote control and testing of code that interprets serial messages. Serial input can also be used to control other measurement devices. We have successfully integrated a BusPirate device to our system in this way. To do so we connect the BusPirate inputs (which are normally in high impedance state) to two of our digital measuring lines. By remotely connecting these lines to the proper pins of the micro-controller, we are able to observe messages being transmitted on the bus.

6 USER EXPERIENCES WITH HEIMDALL

We recruited four participants with prior experience in teaching embedded electronics for a limited-scope formative study to evaluate whether Heimdall can effectively enable instructors to remotely debug circuits. After an introduction to the system's functionality, participants were presented with two circuit debugging tasks in which they used the Heimdall interface to remotely inspect a circuit over a network connection.

Methodology

Participants were first asked to complete a pre-survey to establish their experience and familiarity with offering debugging assistance. They were then introduced to the system in the context of a simple example circuit. The experimenter walked them through Heimdall's interface, first for manipulating the view of the remote circuit board, and then for probing, disconnecting, and injecting signals into the breadboard rows.

Both tasks were presented as student circuits containing an error in either the software or hardware. The experimenter provided the participant with a description of the circuit's intended functionality and a printout of the embedded code running on the student's microcontroller for each task. The participant was instructed to debug the circuit solely using the Heimdall web interface while explaining their thought process aloud. Debugging continued until the participant could give an accurate description of the issue and a potential fix or until ten minutes had elapsed for each task.

The first task was to debug a simple microcontroller system that monitored an infrared distance sensor and lit an LED whenever an object was close enough to it. The challenge was that while two of the wires going to the the distance sensor were connected correctly (Signal and GND), the Vcc wire was "off by one" and connected to an empty row, leaving the sensor unpowered. This resulted in the sensor output never changing and the LED remaining on.

The second task was to debug a more complex system that read numbers over serial communications and outputted the corresponding Morse Code as a series of dots and dashes as red and blue flashes of an RGB LED. However, the semantic definition of the wires in the code are reversed, creating the inverted lighting pattern. Participants were able to send characters to the remote device through the Heimdall interface and view the series of flashes on the system's live view.

After the two tasks were finished, participants were given a second survey asking them to evaluate their experience.

Findings

All users were able to successfully debug the example circuits. Users made use of Heimdall's key features largely as intended to progress from inspection for building hypotheses, to measurement in order to gather additional information, and finally to isolation and injection to evaluate potential issues. We were also encouraged that the users organically combined the features, particularly the probing, injection, and isolation, in complex ways to solve the tasks.

Visual inspection. Although the users encountered some initial minor difficulty with the orbit and zoom controls, they all became adept at using the pre-captured view to quickly inspect the circuit. All users used this view extensively and remarked on its utility. One user even managed to read the pin information printed in tiny font on the distance sensor's silk screen from the first task by finding the right angle and zoom combination. Users also made effective use of the live camera view to monitor the dynamic behavior of the example circuits, but noted that the lower streaming resolution and need to switch between the views encouraged them to spend more time on the pre-captured view.

Measurements. Participants all used the measurement feature extensively and effectively to probe and understand the circuits in the tasks. They used it to verify that components were receiving power and that outputs were functioning correctly. One mentioned that, "[The point measurements] felt pretty close to using a true multimeter, which is I think the most essential tool for debugging circuits physically that I use." Users also made effective use of multiple color-coded digital probes simultaneously during the second task to keep track of the signals on the RGB LED's color channels.

Signal Injection and Component Isolation. Heimdall provides signal injection and component isolation in lieu of physically moving wires to make connections, removing components, or triggering sensors. Users encountered some frustration and difficulty mapping their debugging workflow onto the new primitives of signal injection and component isolation. Despite the initial learning curve, all but one of the users ultimately used signal injection and all users used component isolation during the tasks. We were pleased to see that two of the users were able to use this functionality without prompting to remotely "brute force" the pinout of the RGB LED in the second example by disconnecting all four of its pins, grounding all of them, and then injecting a digital high signal onto each line one at a time and observing the color on the live view. One user also mentioned that the component isolation was reassuring and wrote, "The ability to break row connectivity before signal injection made me much more confident that my changes would not negatively affect the rest of the user circuit." Overall, the participants were excited with the potential of the system and multiple users mentioned that with time, they felt they would be able to more robustly utilize the debugging primitives Heimdall offers.

7 LIMITATIONS

Our system as realized has several limitations which stem from engineering constraints and design choices.

Instrumentation Impact on Students' Circuits

While we designed our system to have a similar behavior as a regular breadboard, the use of the CircuitStack layout and relay between the inner and outer columns will introduce a small amount of parasitic resistance(0.6Ω) and extra capacitence (100pF between output terminals in the open state). Similarly the crossbar-switch's high resistance (up to 85Ω) limits our ability to inject signal or create inter-row connections. This switch resistance has a limited impact for measurement given the high impedance of the measuring equipment.

Our board also has a greater capacitance between two adjacent rows: we measured about 3pF on a standard breadboard and 8pF for our system (with the crossbar switches open) and up to 14pF if both lines are connected through the crossbar switches to a measuring line. It should also be expected that the added length of traces will add some cross-talk to the system, but this could be addressed with a better PCB design.

Limited Instrumentation

We have 6 digital and 2 analog inputs; this number is sufficient for many class projects. The low instrumentation bandwidth (100Hz) limits the kinds of analyses Heimdall can perform compared to full logic analyzers, oscilloscopes, and arbitrary waveform generators. Such limitations could be solved with additional engineering effort to implement sampling and buffering at higher rates as well as additional circuitry or instrumentation devices for more robust signal generation.

Signal Injection Characterizations

We use the Analog Discovery 2 (AD2) for digital signal injection. It supplies 10mA of current, similar to an I/O pin of a common MCU (enough to drive an LED). To inject analog signals, we use a STM32F205 DAC, which can drive a load as low as 5kΩ. The DAC's maximum resolution is 1MS/s, more than enough to generate low frequency signals which would simulate human inputs, e.g., on a potentiometer.

No Physical Manipulation

Heimdall cannot physically manipulate a user's circuit (i.e., shake it to test an accelerometer, change lighting to test a photo cell, or actuate joysticks). The analog and digital signal injection mechanisms can replace these components with "virtual sensors" to test other parts of the system that rely on sensor input, but we cannot fully test the sensors themselves.

Physical Circuit Constraints

The size of the capture platform and the requirement to access the underside of the student's breadboard currently limits students projects in size. For larger projects, handheld 3D reconstruction and tracking systems might be used (e.g., KinectFusion [21]) and might overcome these limitations when combined with portable instrumentation boards.

Rewiring

The extent to which circuits can be rewired by the remote instructor is limited, however, using the row disconnect and signal injection functionality, Heimdall supports a subset of common rewiring tasks, including:

(1) Connecting a given pin to V_{CC} or *GND*.
(2) Replacing a physically-placed component/sensor input with a simulated signal, e.g., testing MCU firmware.
(3) Diagnosing a sensor by isolating it from the MCU, injecting voltages, and monitoring output, e.g., a light-dependent resistor which is shorted.

Heimdall can isolate any Dual In-line Package (ICs, MCUs) component as long as modules are placed centrally and leads to other components are on the outermost periphery of the breadboard. This design choice is also required in traditional breadboards.

Microcontroller code changes

Heimdall does not include remote code editing, but there are several cloud-based IDEs for MCUs which would allow the remote instructor to make edits to student code. One example is the Particle "Build" [22] Web-based IDE. These existing systems could be used in conjunction with Heimdall.

Unsuitable Classes of Problems

Heimdall's limitations can require a degree of unconventional thinking on the part of instructors in debugging, compared with working in-person; however, there are certain classes of problems that are not well-suited to Heimdall. In particular, Heimdall cannot connect components that are not physically present on the board, such as adding a pull-up resistor; our system also cannot split two or more connections within a row, or move wires on the student breadboard. Many problems can still be debugged despite these limitations, however, testing correct behavior of the "debugged circuit" is not always possible.

Lastly, some classes of circuit are not well-suited to Heimdall due to limitations imposed by the instrumentation hardware. In particular, circuits that rely on high-frequency signals may exhibit different behavior when instrumentation is active.

Unknown Student Experience

Our primary goal was to show that knowledgeable instructors can successfully use our tool to remotely inspect circuits.Thus far, we have only collected first-use feedback from a small number of experts. We do not yet know what impact Heimdall would have from a student's perspective, and how it would change the expectations of students and teachers in a class setting. An evaluation of a field deployment, such as in a makerspace or a class, would shed light on this question; we leave it for future work.

8 CONCLUSIONS AND FUTURE WORK

This paper presents Heimdall, a remotely controlled inspection workbench for debugging microcontroller projects frequently encountered in universities and makerspaces. We analyzed common in-person debugging techniques (visual inspection, measurement, and circuit modification) and introduce hardware and software to enable similar functionality remotely. Initial feedback from users suggests that our tool could be effectively used by teachers in electronics debugging courses to evaluate and debug student circuits.

In the future, we are interested in integrating remote code debugging with circuit debugging; and to fabricate enough student breadboards to deploy Heimdall for a full semester in the makerspaces and labs that support embedded systems courses at our universities.

Additionally, we have not yet investigated how instructors should best communicate feedback back to students. Instructors may not want to give away the answer but rather provide hints that teach students how to perform the right measurements and analyses themselves so they become more self-sufficient debuggers.

Acknowledgements

This work was supported in part by NSF awards CNS 1505728 and IIS 1149799.

REFERENCES

[1] Marzieh Ahmadzadeh, Dave Elliman, and Colin Higgins. 2005. An Analysis of Patterns of Debugging Among Novice Computer Science Students. In *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE '05)*. ACM, New York, NY, USA, 84–88. https://doi.org/10.1145/1067445.1067472

[2] Jennifer M. Allen, Leo Gugerty, Eric R. Muth, and Jenna L. Scisco. 2013. Remote Technical Support Requires Diagnosing the End User (Customer) as well as the Computer. *Human–Computer Interaction* 28, 5 (sep 2013), 442–477. https://doi.org/10.1080/07370024.2013.770360

[3] Zain Asgar, Joshua Chan, Chang Liu, and Paulo Blikstein. 2011. LightUp: A Low-cost, Multi-age Toolkit for Learning and Prototyping Electronics. In *Proceedings of the 10th International Conference on Interaction Design and Children (IDC '11)*. ACM, New York, NY, USA, 225–226. https://doi.org/10.1145/1999030.1999067

[4] Inc. Autodesk. [n. d.]. Circuits on Tinkercad. https://www.tinkercad.com/circuits Retrieved September 20, 2018.

[5] Tracey Booth, Simone Stumpf, Jon Bird, and Sara Jones. 2016. Crossed Wires: Investigating the Problems of End-User Developers in a Physical Computing Task. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 3485–3497. https://doi.org/10.1145/2858036.2858533

[6] John S Brown, Alan G Bell, and Richard R Burton. 1974. Sophisticated Instructional Environment for Teaching Electronic Troubleshooting. http://www.dtic.mil/docs/citations/ADA002148

[7] Ok choon Park and Stuart S. Gittelman. 1992. Selective use of animation and feedback in computer-based instruction. *Educational Technology Research and Development* 40, 4 (dec 1992), 27–38. https://doi.org/10.1007/bf02296897

[8] Bettina Conradi, Verena Lerch, Martin Hommer, Robert Kowalski, Ioanna Vletsou, and Heinrich Hussmann. 2011. Flow of Electrons: An Augmented Workspace for Learning Physical Computing Experientially. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS '11)*. ACM, New York, NY, USA, 182–191. https://doi.org/10.1145/2076354.2076389

[9] Martyn Cooper. 2005. Remote laboratories in teaching and learning – issues impinging on widespread adoption in science and engineering education. http://online-journals.org/index.php/i-joe/article/view/298

[10] Daniel Drew, Julie L. Newcomb, William McGrath, Filip Maksimovic, David Mellis, and Björn Hartmann. 2016. The Toastboard: Ubiquitous Instrumentation and Automated Checking of Breadboarded Circuits. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 677–686. https://doi.org/10.1145/2984511.2984566

[11] Drew H. Gitomer. 1988. Individual Differences in Technical Troubleshooting. *Human Performance* 1, 2 (jun 1988), 111–131. https://doi.org/10.1207/s15327043hup0102_3

[12] Ian Grout, J. Murphy, J. Walsh, and T. O'Shea. 2005. Local and Remote Laboratory User Experimentation Access using Digital Programmable Logic. http://online-journals.org/index.php/i-joe/article/view/294

[13] Edwin L. Hutchins, James D. Hollan, and Donald A. Norman. 1985. Direct Manipulation Interfaces. *Hum.-Comput. Interact.* 1, 4 (Dec. 1985), 311–338. https://doi.org/10.1207/s15327051hci0104_2

[14] David H. Jonassen and Woei Hung. 2006. Learning to Troubleshoot: A New Theory-Based Design Architecture. *Educational Psychology Review* 18, 1 (mar 2006), 77–114. https://doi.org/10.1007/s10648-006-9001-8

[15] Andrew J. Ko and Brad A. Myers. 2005. A framework and methodology for studying the causes of software errors in programming systems. *Journal of Visual Languages & Computing* 16, 1-2 (feb 2005), 41–84. https://doi.org/10.1016/j.jvlc.2004.08.003

[16] Andrew J. Ko and Brad A. Myers. 2008. Debugging Reinvented: Asking and Answering Why and Why Not Questions About Program Behavior. In *Proceedings of the 30th International Conference on Software Engineering (ICSE '08)*. ACM, New York, NY, USA, 301–310. https://doi.org/10.1145/1368088.1368130

[17] Siu Cheung Kong, Yau Yuen Yeung, and Xian Qiu Wu. 2009. An experience of teaching for learning by observation: Remote-controlled experiments on electrical circuits. *Computers & Education* 52, 3 (apr 2009), 702–717. https://doi.org/10.1016/j.compedu.2008.11.011

[18] Renée McCauley, Sue Fitzgerald, Gary Lewandowski, Laurie Murphy, Beth Simon, Lynda Thomas, and Carol Zander. 2008. Debugging: a review of the literature from an educational perspective. *Computer Science Education* 18, 2 (jun 2008), 67–92. https://doi.org/10.1080/08993400802114581

[19] Will McGrath, Daniel Drew, Jeremy Warner, Majeed Kazemitabaar, Mitchell Karchemsky, David Mellis, and Björn Hartmann. 2017. Bifröst: Visualizing and Checking Behavior of Embedded Systems Across Hardware and Software. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 299–310. https://doi.org/10.1145/3126594.3126658

[20] David A. Mellis, Leah Buechley, Mitchel Resnick, and Björn Hartmann. 2016. Engaging Amateurs in the Design, Fabrication, and Assembly of Electronic Devices. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16)*. ACM, New York, NY, USA, 1270–1281. https://doi.org/10.1145/2901790.2901833

[21] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. 2011. KinectFusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. 127–136. https://doi.org/10.1109/ISMAR.2011.6092378

[22] Particle. 2018. Particle Cloud. https://www.particle.io/ Retrieved December 20, 2018.

[23] Francisco J. Romero-Ramirez, Rafael MuÃśoz-Salinas, and Rafael Medina-Carnicer. 2018. Speeded up detection of squared fiducial markers. *Image and Vision Computing* 76 (2018), 38 – 47. https://doi.org/10.1016/j.imavis.2018.05.004

[24] N Sousa, G R Alves, and M G Gericota. 2010. An Integrated Reusable Remote Laboratory to Complement Electronics Teaching. *IEEE Transactions on Learning Technologies* 3, 3 (jul 2010), 265–271. https://doi.org/10.1109/tlt.2009.51

[25] James G. Spohrer and Elliot Soloway. 1986. Analyzing the High Frequency Bugs in Novice Programs. In *Papers Presented at the First Workshop on Empirical Studies of Programmers on Empirical Studies of Programmers*. Ablex Publishing Corp., Norwood, NJ, USA, 230–251. http://dl.acm.org/citation.cfm?id=21842.28897

[26] Ryo Suzuki, Gustavo Soares, Andrew Head, Elena Glassman, Ruan Reis, Melina Mongiovi, Loris D'Antoni, and Bjorn Hartmann. 2017. TraceDiff: Debugging unexpected code behavior using trace divergences. In *2017 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE. https://doi.org/10.1109/vlhcc.2017.8103457

[27] M. Tawfik, E. Sancristobal, S. Martin, R. Gil, G. Diaz, A. Colmenar, J. Peire, M. Castro, K. Nilsson, J. Zackrisson, L. Hkansson, and I. Gustavsson. 2013. Virtual Instrument Systems in Reality (VISIR) for Remote Wiring and Measurement of Electronic Circuits on Breadboard. *IEEE Transactions on Learning Technologies* 6, 1 (jan 2013), 60–72.

https://doi.org/10.1109/tlt.2012.20

[28] Chiuan Wang, Hsuan-Ming Yeh, Bryan Wang, Te-Yen Wu, Hsin-Ruey Tsai, Rong-Hao Liang, Yi-Ping Hung, and Mike Y. Chen. 2016. CircuitStack: Supporting Rapid Prototyping and Evolution of Electronic Circuits. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 687–695. https://doi.org/10.1145/2984511.2984527

[29] Jeremy Warner, Ben Lafreniere, George Fitzmaurice, and Tovi Grossman. 2018. ElectroTutor: Test-Driven Physical Computing Tutorials. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. ACM, New York, NY, USA, 435–446. https://doi.org/10.1145/3242587.3242591

[30] Te-Yen Wu, Hao-Ping Shen, Yu-Chian Wu, Yu-An Chen, Pin-Sung Ku, Ming-Wei Hsu, Jun-You Liu, Yu-Chih Lin, and Mike Y. Chen. 2017. CurrentViz: Sensing and Visualizing Electric Current Flows of Breadboarded Circuits. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 343–349. https://doi.org/10.1145/3126594.3126646

[31] Te-Yen Wu, Bryan Wang, Jiun-Yu Lee, Hao-Ping Shen, Yu-Chian Wu, Yu-An Chen, Pin-Sung Ku, Ming-Wei Hsu, Yu-Chih Lin, and Mike Y. Chen. 2017. CircuitSense: Automatic Sensing of Physical Circuits and Generation of Virtual Circuits to Support Software Tools.. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 311–319. https://doi.org/10.1145/3126594.3126634