CurrentViz: Sensing and Visualizing Electric Current Flows of Breadboarded Circuits

Te-Yen Wu* **Hao-Ping Shen*** **Yu-Chian Wu*** **Yu-An Chen*** **Pin-Sung Ku***
Ming-Wei Hsu* **Jun-You Liu*** **Yu-Chih Lin*** **Mike Y. Chen†**
National Taiwan University
*{r04922078, r04922060, r05922103, r05922105,r05922095}@ntu.edu.tw
*{r05944023, b03502040, r04922170}@ntu.edu.tw †mikechen@csie.ntu.edu.tw

ABSTRACT
Electric current and voltage are fundamental to learning, understanding, and debugging circuits. Although both can be measured using tools such as multimeters and oscilloscopes, electric current is much more difficult to measure because users have to unplug parts of a circuit and then insert the measuring tools in serial. Furthermore, users need to restore the circuits back to its original state after measurements have been taken. In practice, this cumbersome process poses a formidable barrier to understanding how current flows throughout a circuit. We present *CurrentViz*, a system that can sense and visualize the electric current flowing through a circuit, which helps users quickly understand otherwise invisible circuit behavior. It supports fully automatic, ubiquitous, and real-time collection of amperage information of breadboarded circuits. It also supports visualization of the amperage data on a circuit schematic to provide an intuitive view into the current state of a circuit.

ACM Classification Keywords
H.5.2. Prototyping

Author Keywords
Electric Current; Breadboarded Circuit; Debugging; Visualization;

INTRODUCTION
Electric current and voltage are essential to learning, understanding, and debugging circuits. In an actual circuit, both can be measured using tools such as multimeters and oscilloscopes. To measure voltage, users simply place two probes at any two points in a circuit. Measuring current, however, is much more difficult. Users need to break the circuit at the point they would like to measure, and then insert a measurement tool in serial to complete the circuit. Users also need to restore the circuit back to its original state after measurements has been taken. This time-consuming and error-prone process makes measuring current throughout a circuit impractical.

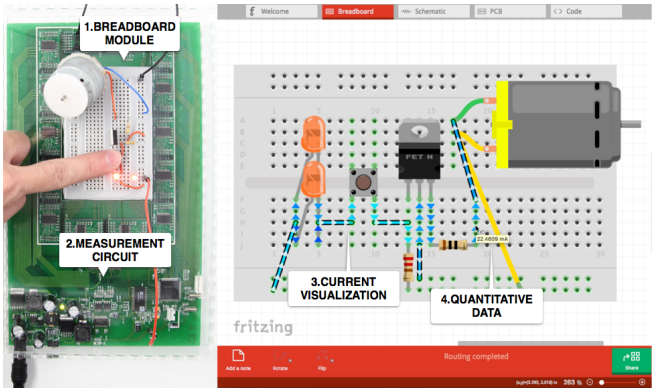

Figure 1. CurrentViz device and accompanying software. (1) The breadboard module, on which the user circuit can be built. (2) The underlying circuit to measure currents. (3) The current visualization, with arrows indicating flow direction and color representing intensity. (4) The exact intensity is displayed in a popup.

With the rise of Maker movement and education, there is significant growing interest in learning, understanding, and building electronic circuits. Breadboard, in particular, is the most widely used approach to build circuits for education and for maker projects. Recent efforts [11, 13, 8, 7, 10, 12] have explored ways to make the invisible circuit behavior more understandable. In particular, Toastboard[11] has improved upon existing tools by providing ubiquitous measurement and visualization of voltage on a breadboard. However, ubiquitous measurement and visualization of electric current, which is critical to understanding and debugging breadboard circuits, remains to be addressed.

We present CurrentViz, a system that offers ubiquitous measurement and visualization of electric current of breadboarded circuits. As shown in Figure 1, CurrentViz has a breadboard module that is the same form factor and appearance as ordinary breadboards, enabling full compatibility with exisiting breadboard circuits, components, and projects. Our current prototype collects measurements at 240 points on the breadboard, and carries out analysis and computation to produce continuous, real-time updates. To visualize the data, we use ar-

UIST 2017, October 22–25, 2017, Quebec City, QC, Canada
 ISBN 978-1-4503-4981-9/17/10…$15.00
DOI: https://doi.org/10.1145/3126594.3126646

rows to represent current flow direction and color to represent amperage level, and present the results using the schematic view in Fritzing [3], as illustrated in Figure 1. Furthermore, we offer a convenient API for users to process current measurement and to customize visualization.

The rest of the paper is organized as follows: We first discuss related work, and present the design and implementation of our system. We discuss example cases of debugging, then present limitations and future work, and finally summarize our contributions.

RELATED WORK

Measurement and Visualization Tools for Circuits

The most common debugging tools in use with breadboards are multimeters and oscilloscopes. These tools provide numeric measurement of a circuit, but can only measure a single point at a time, and users need to take multiple measurements to acquire a more holistic view of the circuit. Visible Board[13] partially simplifies the process by directly displaying voltage information in corresponding software. It also allows users to connect two points by touching them, instead of physically placing wires. However, only voltage information is available, and the distance between holes is 3.2cm – more than 12 times larger than the 2.54mm of standard breadboards – which would demand fully customized components.

The Toastboard[11] provides ubiquitous measurement and visualization of voltage on a breadboard, and also offers functionality similar to oscilloscopes by enabling users to observe voltage waveforms over time. In addition, Bifrost[12] visualizes and compares the electrical activity and the variable values at the interface between the processor and the circuit. It helps users better understand the behavior of embedded systems across hardware and software. These tools both improve the debugging environment by visualizing the information in circuits, but leave current information alone. CurrentViz provides ubiquitous measurement of current on a breadboard, which when combined with Toastboard could provide complete visibility into a breadboarded circuit. Besides, having amperage measurement is needed to identify several classes of common circuit errors, such as an LED connected in reverse or a motor not functioning due to insufficient current.

Virtual Circuit Design and Simulation

Nowadays, there is an abundance of auxiliary software to help users design and simulate circuits. Fritzing is a software program to design printed circuit boards (PCBs), and one of the favorites among circuit makers. Fritzing has a simple and easy-to-use user interface, and supports prototype design in the breadboard view, the schematic view, and the PCB view. All views are automatically updated when a new component is dragged onto the virtual breadboard, truly visualizing the entire workflow of circuit prototyping. Furthermore, some have found success using Fritzing in an electronic class, or designing an Arduino, etc.

AutoDesk [1] Circuits allows the user to design a circuit on top of a virtual breadboard, and additionally supports simulation. PSpice [5] is a circuit simulation and analysis program that supports simulation of circuits with both analog and digital components. Users can design and edit a circuit in the circuit design program OrCAD [4] Capture, select an analysis method and specify input parameters, and then use PSpice to perform analysis on the circuit. CircuitLab [2] provides an environment for circuit simulation and drawing schematics on the browser, which enables users to construct and test virtual circuits schematics. However, a simulated circuit could differ from an actual, physical circuit. This might be due to users connecting wires incorrectly, misusing components, or supplying insufficient power, etc. CurrentViz provide real-time updates on the electric current flows in a physical breadboard, giving users a clear view and accelerating debugging efforts.

Education Tools for Electronic circuit

Electronic education is an increasingly popular topic in HCI research. Many studies have discussed how to help newcomers quickly become more familiar with circuit prototyping. Kowalski *et al.* supposed that the learning experience of physical computing can be fostered by interactive surfaces offering the following advantages: (1) Collaborative learning experience, (2) Augmentation of physical components and (3) Continuous guidance from digital draft to physical prototype. [9] BitBlox[10] uncovers the underlying connections within the modules of a breadboard, thereby decreasing the cognitive load on users.

Some other toolkits[8][6] additionally focus on augmented workspaces that track customized components and provide relevant information in place. Having a representation of the invisible behavior of the circuit, users would be able to more deeply understand important concepts in electronics.

SYSTEM DESIGN AND IMPLEMENTATION

To support continuous measurement of electric current across an entire breadboard, our system consists of the following 3 key modules: 1) Breadboard, 2) Multiplexer Cascades, and 3) Data Processor, as shown in Figure 2.

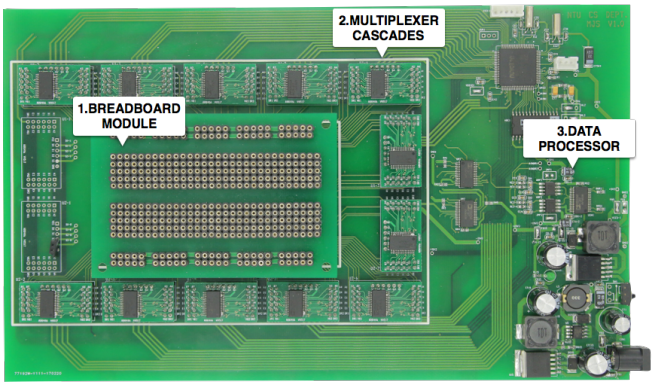

Figure 2. CurrentViz hardware highlighting the 3 key modules: 1) Breadboard, 2) Multiplexer Cascades, and 3) Data Processor. The board includes multiplexers, amplifiers, and an ADC, which are all controlled by a microcontroller. There is also a UART wire that connects the board to the computer, and a power adapter module to supply different required voltages.

We leverage Ohm's law ($V = IR$) to convert the measurement of current into the measurement of voltage. We start by designing the breadboard module with a fixed resistor between each pair of adjacent points in a row of 5 points. This provides the fixed resistance needed in Ohm's law. To get the precise voltage difference, we then insert an instrumentation amplifier (In-Amp) with differential inputs, connecting points on the board to the In-Amp, and taking its output to the input of an Analog to Digital Converter (ADC). With this setup, we can use the formula $I = V/R$ to calculate the current. In order to sample such a large number of inputs (192 points), we use two two-stage cascades of multiplexers controlled by ARM Cortex-M453AE6VG to perform round-robin selection on the inputs at a rate of around $1000Hz$ per set of inputs. Figure 3 illustrates the system described above.

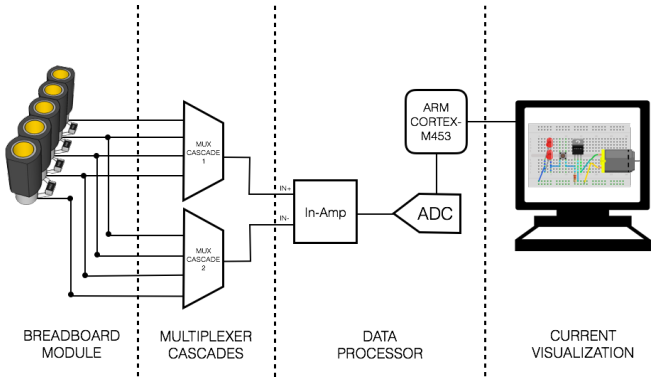

Figure 3. A simple diagram of the CurrentViz system. Each row in the breadboard module is connected to two multiplexer cascades, which performs round-robin selection on pairs of adjacent points to forward signals into the data processor. Finally, the microcontroller processes the data read by the ADC and visualizes it in circuit design software.

Breadboard Module

The breadboard module has the same appearance and form factor as a standard breadboard, such that any breadboard-compatible component can be used. In the middle are 60 horizontal rows of 5 points each, out of which 48 rows are usable. To capture the electric current profile of the entire circuit board, we consider the nature of breadboards that one point has at most one component plugged in, which means that the current at one point has at most 3 directions of flow: to either of the two adjacent points in the same row and to the component currently plugged in (Figure 4(a)). By Kirchhoff's current law (KCL), we can calculate the current flowing into the component as long as we can measure the current flowing to the adjacent points. To that end, we use a 0402 resistor to connect each pair of points in a row (4 resistors in total), in order to form a measurable voltage between adjacent points and calculate the current ($I = V/R$) with the microcontroller. We choose a very low resistor (0.1Ω) such that the impact on the normal behavior of the circuit is minimized, shown as Figure 4(b).

Multiplexer Cascades

To sample all 240 points on 48 rows, we employ the 16:1 MUX ADG426 to design two two-stage cascades, enabling the MCU

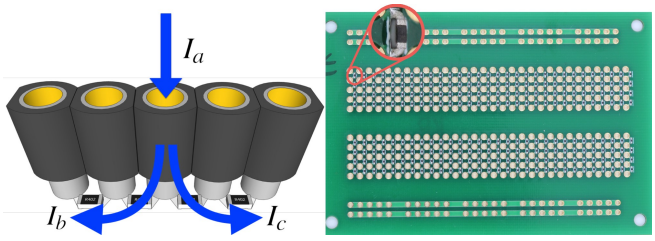

Figure 4. (a) The current flow in either direction (I_b, I_c) can be calculated from the voltage difference between the point under test and either adjacent point. By KCL, the sum of currents at a single point must be zero, so the outflowing current I_a is $I_b + I_c$. (b) In the breadboard module, the 5 points in each row are connected with 4 0.1Ω 0402 resistors, the largest resistors in size that can fit between two holes ($1mmX0.5mm$).

to control which two points to test through General-Purpose Input/Output (GPIO) pins. In an earlier design, we tried connecting every point to the multiplexer as shown in Figure 5(a), similar to the Multiplexer Cascades in Toastboard[11]. The resulting circuit requires 32 multiplexers and 128 GPIO pins, which is unacceptable.

To improve on this design, we observe that the two points under test at once must be two adjacent points in the same row, and one test point being fixed would cause the other one to also be fixed. This helped us make key improvements to our design as illustrated in Figure 5(b). Of the 5 points in a row, we connect the first 4 points to MUX 1, the last 4 points to MUX 2, and connect the control pins of both MUXs. Thus, we can use a pair of 16-channel MUXs to fully cover 4 rows. With this improvement, the number of MUX in the first stage becomes 24 MUXs (48 rows / 4 row $\times$ 2); together with the 2 MUXs needed in the second stage, only 26 MUXs are required in total. Accordingly, 52 lines ((26 / 2) $\times$ 4=) will be drawn on the PCB. The final design is as illustrated in Figure 6.

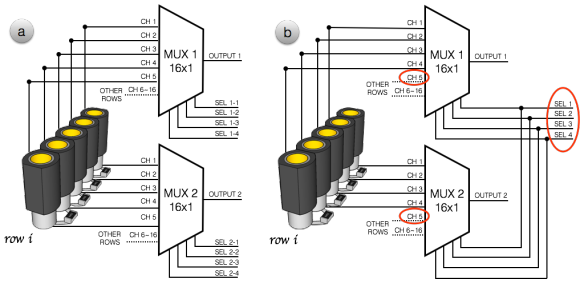

Figure 5. The original design of the Multiplexer Cascades and later improvements.

Data Processor

Depending on the user-created circuit, the input voltage magnitude of the In-Amp can vary. Most In-Amps offering precise gain selection have a fixed gain, which means that one In-Amp might not be enough to amplify the voltage to a suitable range.

We designed an automatic mechanism to select a suitable gain in different situations, in order to make sure that the input voltage always falls within the measurement range. The mechanism consists of a multiplexer, an MCU, an ADC, and multiple In-Amps with different gains. We first connect the points

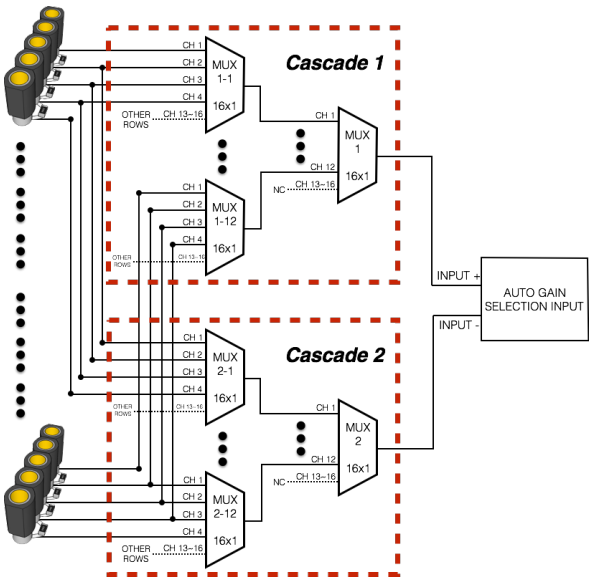

Figure 6. The first 4 points (a,b,c,d) are connected to 4 channels of a MUX in the first stage of Cascade 1, while the last 4 points (b,c,d,e) are connected to 4 channels of a MUX in first stage of Cascade 2. In either cascade, the first-stage MUX's are then connected to one MUX (the second stage). Finally, the outputs of the two MUXs in the second stage are connected to the In-Amp set.

to test to all In-Amps simultaneously, and connect the outputs of these In-Amps to the multiplexer. When amplification of voltage is desired, the MCU first switches the multiplexer to the In-Amp with the lowest gain, and inspects the voltage amplified by the ADC; if the result is lower than a minimum accuracy threshold, the MCU will switch the multiplexer to the In-Amp with the next lowest gain, and so forth until the MCU receives a voltage within the suitable range.

We chose to use 2 AD8228 In-Amps, whose gain can be set to either 10 or 100 by short-circuiting or open-circuiting two of its pins. For the ADC we chose AD1674, which has a sample rate of 100kHz, a measurement range of $-5V$ to $+5V$, and accuracy of 12bits. This means that the maximum and minimum of the voltage we can measure are 500mV and $50\mu V$ respectively, and the sample rate can be as high as 1kHz. The procedure is illustrated in Figure 7.

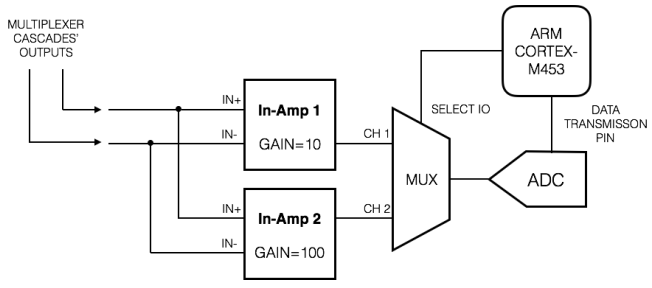

Figure 7. The block diagram of Auto Gain Selection. Signals are simultaneously connected to multiple In-Amps with different gains. The MUX selects an In-Amp in increasing gain order. Based on the ADC's measurement, the MCU decides whether to switch to a higher-gain In-Amp.

Specification of Hardware

In this section, we will discuss the measurement range of currents, the sample rate of measurement, and the impact of CurrentViz on the behavior of the circuit. The measurement range of currents are mainly limited by three factors: the type of resistors used in the breadboard module, the gain of the In-Amp set, and the accuracy of the ADC. The 0402 resistors used in the breadboard module have a power rating of 1/16W. By Ohm's law $P = I^2R$, we know that their maximum current is $790mA$. On the other hand, the voltage difference between two adjacent points, amplified by the In-Amp set, must fall within the measurement range of the ADC. The ADC used in CurrentViz has a resolution of 12 bits and a working voltage of 5V, which means its minimum accuracy is $2.4mV$ (the first bit is the signal bit). The gain of the In-Amp set is configurable to either 10 or 100. Thus the lowest measurable voltage difference is $2.4mV/100 = 24\mu V$; and the highest voltage difference is $5V/10 = 0.5V$. In other words, we can calculate the lowest and highest measurable currents are $24\mu V/0.1\Omega = 240\mu A$ and $0.5V/0.1 = 5A$, respectively. Finally, we combine the current limits of the breadboard module, the ADC, and the In-Amp to conclude that the current measurement range of CurrentViz is $240\mu A$ to $790mA$.

In order to achieve real-time data collection, we employed an ADC with a high sample rate and low-value resistors. The sample rate of the ADC can be as high as 100kHz theoretically, but due to the need to switch MUX's and wait for the output of the In-Amp to stabilize, the realistic sample rate is approximately 1kHz. Consequently, it takes approximately 0.2s to perform measurement at all points on the breadboard. With regards to the resistor selection, we chose 0.1Ω resistors to minimize the impact to the breadboard module. As a result, if the lowest resistor used in the user circuit is 100Ω, CurrentViz would incur an error of approximately 0.1%.

Current Visualization

Typically, makers and hardware developers surfed the web for schematics or designed their circuit diagrams on popular platforms, such as Fritzing [3]. In order to support users debugging in their circuits, we designed and implemented the visualization of CurrentViz based on the virtual circuits in Fritzing. In contrast to the available visualizations of current only on circuit schematic, we present the current on corresponding breadboard diagram for the convenience to examine with physical circuits. We focused on the ubiquitous presence of current flow and the intuitive clue of amperage value to enable users to quickly understand circuit behaviour and rapidly find the potential problems. We proposed and implemented the following guidelines of visualization when presenting current on a virtual breadboard:

Showing the current flow on each connection: In order to present ubiquitous current flow on a virtual breadboard without the obstruction of viewing user-created circuits, we used the animation of arrows to show the current flow only on the connections such as the wires and rows of the breadboard (Figure 8.1).

Providing the detail information if needing: When users encountered current-related problems, they would require more

detail information of the current. We provided the functionality for retrieving the precise value of the current intensity when users were hovering on it (Figure 8.2).

Cluing the intensity of current: To provide users the clue of current value on the components, we mapped the intensity of the current to the color from red to blue (Figure 8.3), which illustrated the current range from 500mA to 5mA.

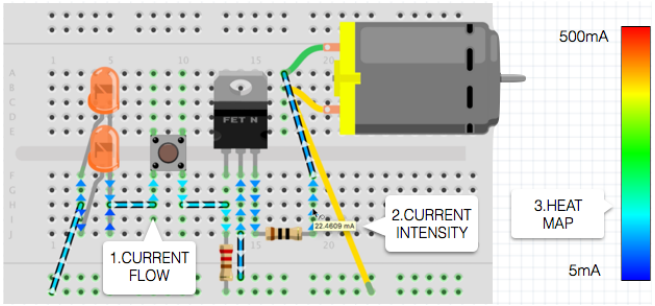

Figure 8. Software view of CurrentViz. (1) The current flow is shown by moving arrows on the virtual breadboard. (2) The popup shows up and gives the precise value when a user hovers on the current. (3) The color of the arrows and wires illustrates the current intensity from red (high) to blue (low).

DEBUGGING EXAMPLES

In this section, we show several examples of the debugging scenarios that current information is uniquely useful to.

Parallel Circuits

When a component malfunctions in a parallel circuit, locating the problem can be quite a challenge with existing debugging tools. As Figure 9 shows, in a parallel circuit with 3 infrared LEDs, one LED is inserted with the incorrect polarity and is thus not working properly. The user cannot visually determine which LED is broken, and voltage information is not helpful either as the voltages on all 3 LEDs are the same. However, with CurrentViz, the user can easily see the lack of current on one LED and locate the problem.

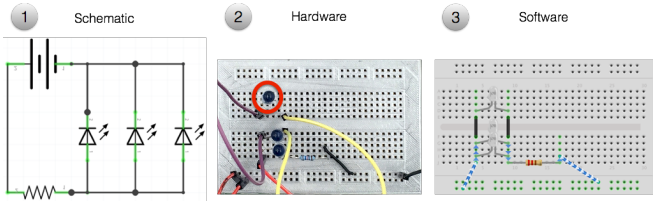

Figure 9. (1) The schematic for infrared LED circuits. (2) As the LEDs emit invisible infrared light, the user doesn't realize that they plugged the top LED in the wrong polarity. (3) With CurrentViz the user can quickly realize that there is a problem with the top LED, as there is no current flowing through it.

Power Management

When there are multiple high-power components coexisting in a circuit, power management will become a serious issue. As shown in Figure 10, the circuit contains two speakers which already exceed the output limit of the Arduino providing power, causing both speakers to malfunction. Traditional debugging tools fail to provide power measurements, and simply measuring the voltage is not helpful either (as the voltages are correct). But with CurrentViz, the user can perceive the abnormal current flow, and calculate the power consumed by each component, thus deducing that the problem results from insufficient power. Alternatively, CurrentViz provides all current information on the breadboard such that the user can quickly get the amperage on each component and further compute the power consumption based on the voltages and the power equation ($P = I \times V$).

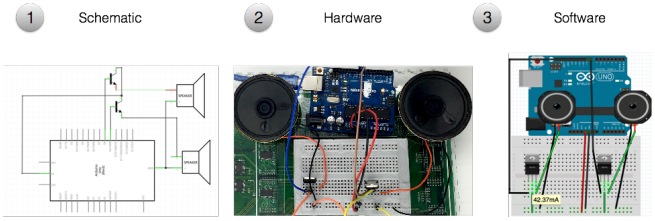

Figure 10. (1) A circuit with two speakers controlled by an Arduino. (2) The speaker receives power from the GPIO pin of the Arduino, but a single GPIO has a maximum output current of 40mA, which is insufficient for the speaker with a working power of 0.5W. (3) With the amperage of 42mA, provided by CurrentViz, and the voltage of 5V, the user can deduce that the speaker is only receiving 210mW, lower than the working power.

Current Amplifier/Inverter/Divider circuits

Current Amplifier/Inverter/Divider circuits are the basic circuits to control the current flowing into further circuits. A current amplifier circuit in Figure 11, for example, amplifies the input current by a fixed factor and feeds it to other circuits. The circuit is a typical circuit in electronic classes, but it is difficult to understand and verify the current gains with the voltage and resistance. With CurrentViz, the visualized and detailed currents are shown on the software. Users can simply use this information to confirm the current gains.

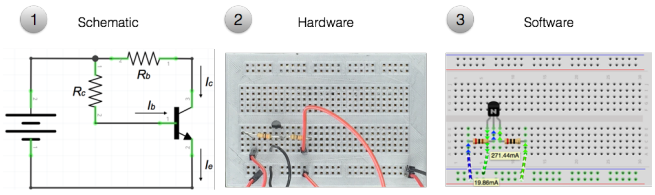

Figure 11. (1) A simple BJT amplifier circuit which amplifies I_b by a fixed factor(β). (2) In order to verify the value of β, users have to measure lot of points and perform a lot of calculation. (3) User can easily find out that $\beta = I_c/I_b = 271.44/19.86 = 13.67$ by the amperage shown in software.

INFORMAL USER FEEDBACK

To verify the usability of CurrentViz we recruited 9 participants. Experience levels ranged from novices with only a little formal circuit experience (3 participants), to moderately experienced builders (4 participants), to experts who have constructed many circuits (2 participants). We prepared the following 3 types of problematic circuits for the participants

to debug: 1) Misplacement: polarized components plugged in reverse, or components plugged in the wrong holes; 2) Unexpected sensor readings: short-circuited points of measurement, floating pins, or wrong usage of sensors; 3) Power management: insufficient voltage or currents causing components to have too low power. We've also provided 3 types of information for the participants to consult: "current only", "voltage only", and "both", so participants need 9 circuits in total, and the order in which they appear is counter-balanced. In addition, we also provide the schematics of the circuits and the source code of the firmware. We asked participants to think aloud during the experiment and note down the problems they thought about. We performed post-interviews with them, and got the result that 6 out of 9 participants preferred "both", 2 preferred "current only" (1 novice and 1 expert), and 1 preferred "voltage only". We also had the following findings:

- 4 out of 9 participants expressed in the "power management" section that only being able to view either voltage or current is not enough to spot the problem, but having both would expose the problem of insufficient power.

- 3 out of 9 participants noted that in the more complex circuits in the "unexpected sensor readings" section, the direction of current flow is very helpful for locating the general region of the bug (usually the broken component has no current flow), and then they could use the voltages to precisely locate which component is malfunctioning.

- The expert participant who preferred "current only" commented that "the direction of current flow helped deduce the behavior of the circuit, and even when I looked at voltage, I converted it to current in my head."

DISCUSSION

Measurement Range

As the circuit grows in size and complexity, some components might require too high or too low current, causing CurrentVizto fail to measure this.

We can solve the problem of the current being too low via adding an In-Amp with a bigger gain into the set of In-Amps. For example, the ADG426 MUX has a normal current of $5\mu A$, far lower than CurrentViz's accuracy $0.5mA$. We could simply add an In-Amp with a gain of 10000, and the voltage difference in the breadboard module would be amplified from $0.5\mu V$ to $5mV$, which falls within the ADC's accuracy range.

However, when the current is too high, besides the addition of an In-Amp with a smaller gain, we'd also need to replace the resistors in the breadboard module. The 0402 resistors we are using have a max wattage of 1/16W. By Ohm's law, we can deduce that the voltage can only reach $79mV$ before the resistor is damaged. The simplest solution is just to use resistors with a higher wattage, but generally such resistors are also larger in physical size. Naively increasing the max wattage by using larger resistors could result in the resistors not fitting in the space between holes in the breadboard module, so extra design work would be required to place the resistors.

Combining with Voltage Measurement

CurrentViz gives the user an intuitive view into the behavior of the circuit by display the current. One of our further goals is to also integrate voltage information into the system. Doing so not only brings one more metric to the user, but also enables them to combine the current and the voltage to derive more useful information. For example, we could calculate the internal resistance of a component, and use it as an indicator whether the component conforms to specs or whether it has been damaged. We could also look for points with a voltage difference but no current between them, to detect and locate components with a faulty connection or a pin plugged in a floating row. The Toastboard provides a floating detection circuit to detect which rows on the board are unused, but it fails to detect floating rows with a voltage difference but no current. To achieve this, we can simply add an ADC with a higher range of measurement (for example $-15V$ to $+15V$) to the hardware of CurrentViz, and connect its input to one of the MUX outputs in the second stage of the multiplexer cascade. This ADC can then measure the voltage at any given point on the board, and with a round-robin selection mechanism, report voltage information of the whole board.

Debugging Assistant on Software

In order to further assist the user with circuit debugging, we hope to provide more suggestions in our software. We plan to use a rule-based system, where the user can define the expected behavior of each component, and CurrentViz can detect discrepancies in the actual state and provide suggestions and warnings [11]. For example, under a working voltage of $12V$, the DC Motor Crouzet 89850007 should have a working current of $260mA$. If CurrentViz detects a far lower current flowing through this component, it could alarm the user in the software. Furthermore, if CurrentViz has integrated voltage information, we could support much more powerful rule definitions with both working voltage and current.

Furthermore, combining voltage and current, we could develop a power optimization software that automatically visualizes the power consumption of each component, which will be especially important for circuits designed for mobile and wearable usage. In addition, we could explore smart zooming that shows the current (and power) for a section of the circuit when zoomed out, and shows individual components when zoomed in.

CONCLUSION

This paper presents CurrentViz, a system that provides ubiquitous measurement and visualization of electric current flow for breadboard circuits. It exposes the invisible behavior of circuits, and helps users learn, understand, and debug circuits. We designed hardware that can measure amperage values at 240 locations in real-time. Additionally, we implemented software to visualize the current information, facilitating observation of the behavior of the circuit. We trialed CurrentViz for circuit prototyping in practice, and found that the current information did help users quickly locate and solve problems. Finally, we discussed some current limitations of CurrentViz and presented plans for further improvements.

REFERENCES

1. AutoDesk Circuits. **https://circuits.io**
2. CircuitLab. **https://www.circuitlab.com**
3. Fritzing. **http://fritzing.org/**
4. OrCAD. **http://www.orcad.com**
5. PSpice. **http://www.pspice.com**
6. Yoh Akiyama and Homei Miyashita. 2014. Projectron Mapping: The Exercise and Extension of Augmented Workspaces for Learning Electronic Modeling Through Projection Mapping. In *Proceedings of the Adjunct Publication of the 27th Annual ACM Symposium on User Interface Software and Technology (UIST'14 Adjunct)*. ACM, New York, NY, USA, 57–58. DOI: **http://dx.doi.org/10.1145/2658779.2659113**
7. Ayah Bdeir and Paul Rothman. 2012. Electronics As Material: LittleBits. In *Proc. of TEI '12*. ACM, New York, NY, USA, 371–374. DOI: **http://dx.doi.org/10.1145/2148131.2148220**
8. Joshua Chan, Tarun Pondicherry, and Paulo Blikstein. 2013. LightUp: An Augmented, Learning Platform for Electronics. In *Proceedings of the 12th International Conference on Interaction Design and Children (IDC '13)*. ACM, New York, NY, USA, 491–494. DOI: **http://dx.doi.org/10.1145/2485760.2485812**
9. Bettina Conradi, Martin Hommer, and Robert Kowalski. 2010. From Digital to Physical: Learning Physical Computing on Interactive Surfaces. In *ACM International Conference on Interactive Tabletops and Surfaces (ITS '10)*. ACM, New York, NY, USA, 249–250. DOI: **http://dx.doi.org/10.1145/1936652.1936700**
10. Kayla DesPortes, Aditya Anupam, Neeti Pathak, and Betsy DiSalvo. 2016. BitBlox: A Redesign of the Breadboard. In *Proceedings of the The 15th International Conference on Interaction Design and Children (IDC '16)*. ACM, New York, NY, USA, 255–261. DOI: **http://dx.doi.org/10.1145/2930674.2930708**
11. 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. DOI: **http://dx.doi.org/10.1145/2984511.2984566**
12. William 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 Symposium on User Interface Software and Technology (UIST âĂŹ17)*. ACM, New York, NY, USA.
13. Yoichi Ochiai. 2014. Visible Breadboard: System for Dynamic, Programmable, and Tangible Circuit Prototyping with Visible Electricity. In *Virtual, Augmented and Mixed Reality. Applications of Virtual and Augmented Reality*. Lecture Notes in Computer Science, Vol. 8526. 73–84.