BitBlox: A Redesign of the Breadboard

Kayla DesPortes
Georgia Institute of Technology
Atlanta, USA
ksdesportes@gatech.edu

Aditya Anupam
Georgia Institute of Technology
Atlanta, USA
aanupam3@gatech.edu

Neeti Pathak
Georgia Institute of Technology
Atlanta, USA
pathak.neetin@gatech.edu

Betsy DiSalvo
Georgia Institute of Technology
Atlanta, USA
bdisalvo@cc.gatech.edu

ABSTRACT
Building physical computing projects can enable learners to integrate computing into a range of interests and disciplines. However, the electronic portion of these projects can be difficult. Students are learning new concepts as well as how to work with new tools. This influx of information can be difficult for students to retain in their working memory as they construct their circuits. In this paper, we introduce BitBlox, a set of modular, solderless Breadboards for prototyping circuits. BitBlox attempts to decrease the cognitive load on the user by reducing the complexity found in the standard Breadboard by bringing visibility to the underlying connections within its modules. We present a comparative classroom study integrating the Breadboard and BitBlox into two different high school classes. Our qualitative analysis focuses on student errors, strategies, and collaborative practices, highlighting important dynamics for designing hardware tools.

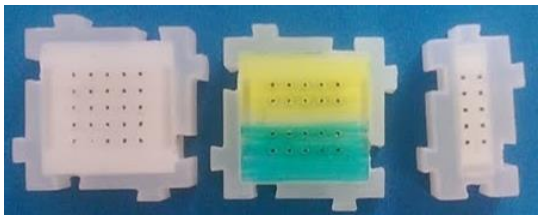
Author Keywords
Electrical circuits; microcontroller; design; education.

ACM Classification Keywords
K.3.0 Computers and Education: General.

INTRODUCTION
Enabling students to create personally meaningful projects can play a pivotal role in encouraging a diversity of students to participate in computing disciplines [9]. One way this can be accomplished is by empowering students to build physical computing projects that weave into their other interests [5]. With the surge in popularity of the maker movement, there have been numerous efforts and tools developed for helping middle and high school students create personally relevant electronics projects [2]. However, many of the tools do not support the flexibility needed to integrate into a vast array of interests while providing the necessary visibility to learn electronics concepts.

IDC '16, June 21 - 24, 2016, Manchester, United Kingdom

ACM 978-1-4503-4313-8/16/06…$15.00
DOI: http://dx.doi.org/10.1145/2930674.2930708



(a) (b) (c)

Figure 1. BitBlox modules: (a) 5x5 block, (b) split block with two 2x5 sections, and (c) 2x5 block.

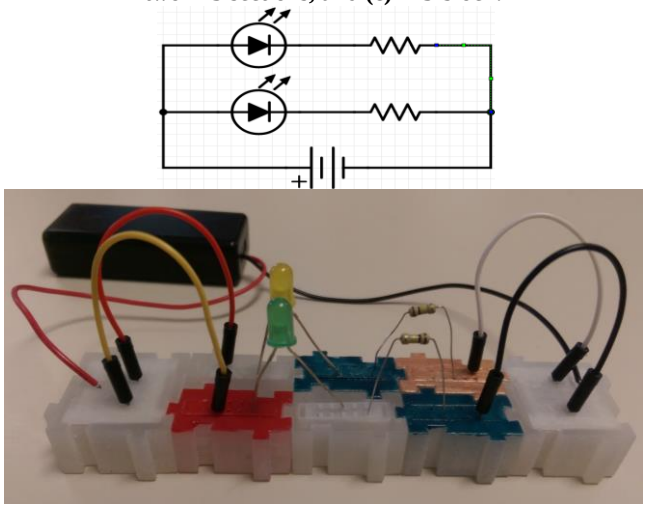


Figure 2. Circuit diagram (top) of circuit built using BitBlox (bottom), illustrating one organizational option.

Blikstein and Sipitakiat categorized physical computing tools into two models for design: the *Cricket model*—tools with a robust design, plug and play features, and hidden component complexities (ex. PicoCricket, LEGO Mindstorms, etc.); and the *Breakout model*—tools with a flexible, extendable design with the circuit complexities offloaded to the Breadboard and shields (ex. Arduino, Raspberry Pi, etc.) [3]. The authors emphasize the importance of creating cross-over tools that can capitalize on the advantages of these two models. Within this paper we discuss BitBlox, a redesigned model of the solderless Breadboard. It was created to encompass the approachability and ease of Cricket models while integrating seamlessly with the flexible Breakout models.

The standard Breadboard is a compact tool for efficiently building circuits, saving the user both time and space. Unfortunately, the design incorporates hidden connections, which can prove challenging for novices [3]. We hypothesize that having to learn the Breadboard connection

scheme, which is irrelevant for learning the electronics concepts, places a higher cognitive load [14] on the student (*see* Background). This would make learning the concepts more difficult than necessary; thus, BitBlox aims to alleviate some of the extraneous load by bringing visibility to the underlying connections (*see* Figure 1 and 2).

We piloted BitBlox with high school juniors in a comparative classroom study with the Breadboard. We ran an identical curriculum in two separate classes—one with BitBlox and one with the Breadboard. Conducting this pilot *in the wild* enabled us to understand the tools in an actual learning environment such that we could highlight some of the important variables for design and future explorations.

BACKGROUND

Educational Tools for Electronics Education

Educational tools for building physical computing projects have been around for over 30 years (*see* [2] for a review). While the two persistent models (*Cricket* and *Breakout)* identified by Blikstein and Sipitakiat, each have their benefits, they also have drawbacks. The Cricket models can hide important circuitry for learning electronics, while the Breakout models can drown novices in their complexity [3].

Chan et al. raise concerns about the gap between the low-threshold, accessible platforms and high-ceiling, expert platforms [6]. A lack of middle ground means there is no bridge to advance students from the Cricket model into the complexities found in the Breakout model such that students can learn the underlying concepts. LightUp is one tool constructed specifically to bridge this gap. It enables quick construction of circuits using magnetic connectors on single component blocks. LightUp brings further visibility to the circuit through an augmented reality overlay which visualizes the underlying circuit behavior [6].

Similar to their platform we have devised a tool to enable users to more easily construct circuits. However, our design allows users to interface their circuits with Breakout models, such as the Arduino. Previous work has incorporated this tactic to enable a greater usability for the software and hardware aspects of the Breakout models. Both Modkit [10] and Splish [8] enable students to use a graphical programming language while interfacing with the Arduino. Blocks-based programming languages have shown a decreased level of misconceptions over text-based languages [17] indicating this as a logical step forward.

The LilyPad Arduino [4] and Chibitronics [12] have both made headway in the hardware tools. They enable students to create projects incorporating microcontrollers through decreasing the complexity of making connections. The LilyPad lets users sew together connections using conductive thread, while Chibitronics enables its users to lay down conductive tape to make connections. BitBlox takes the student one step closer to the more expert tools. It keeps the visibility of the circuit high, while enabling the students to flexibly integrate into a range of physical computing projects more easily. The BitBlox modules were designed as an alternative to the Breadboard to decrease the cognitive load on the learner.

Cognitive Load Theory

Cognitive load theory has been applied in several domains to "provide guidelines intended to assist in the presentation of information in a manner that encourages learning activities that optimize intellectual performance" [16]. Some of these domains include mathematics education [14], computer science education [11], and designing tangibles for learning [1]. We propose to apply cognitive load theory to the educational tools for building circuits.

Cognitive load theory understands learning as it relates to the available working memory of the learner. Due to the limited capacity of working memory, any irrelevant information can interfere with a student's ability to learn [16]. Cognitive load theory makes a distinction between the amount of working memory needed to learn the material—the intrinsic load—and the amount of working memory used on information in the learning environment that is irrelevant to the material—the extraneous load [15]. We hypothesize that the standard Breadboard imposes extraneous cognitive load on the learner, through requiring the learner to understand its underlying connection scheme. BitBlox was designed specifically to decrease the extraneous cognitive load on the user by simplifying the connections. We do not however, attempt to support or refute this hypothesis because we are piloting this in a real-world environment where other variables would impede the control necessary to test this. Instead, we gather information to understand if this is a path worth future investigations.

PROTOTYPING TOOLS

Breadboard

A Breadboard is a tool for easily connecting wires and other electronic elements when prototyping circuits. The Breadboard has two main sections—the outside rows and the middle columns. The holes or contact points in the row sections are only electrically connected to contact points in the same row. The contact points in the column sections are only electrically connected to the contact points in the same column (*see* Figure 3).

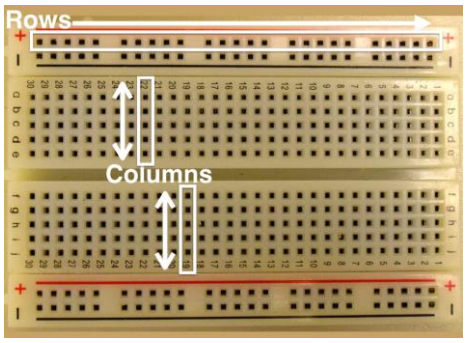


Figure 3. Every hole within each row is interconnected, and every hole within each column is interconnected

BitBlox

BitBlox are similar to Breadboards in that they also have holes or contact points into which wires can be plugged in. Each of these contact points is electrically connected to other contact points. In most variations, all contact points in the board are electrically connected (*see* Figure 1(a) and (c)). The exception is that some BitBlox modules are color coded with two sections of contact points (*see* Figure 1(b)). This split block was created specifically for connecting buttons. To learn how to use BitBlox students need to understand that connections are made by simply spanning a wire across two modules. This connection scheme requires less memorization than the Breadboard's.

The individual BitBlox modules can snap together like puzzle pieces. The flexibility of snapping together the BitBlox allows the user to modify the structure of their breadboard giving them the ability to organize and adapt the tool for their needs. For example, if they wanted to, they could mirror a circuit diagram more closely, like in Figure 2. We hypothesized that this would also allow the students to break up the circuit in various ways for either conceptual or collaborative purposes.

METHODS

The pilot study was comprised of two class periods designed to compare the use of BitBlox to the standard Breadboard. While there are imperfections with comparative studies *in the wild*, the design of this study allowed us to observe how the various tools affected learning and collaboration, bringing to light future directions for further investigation.

Students	Gender		Ethnicity		Age		
	M	F	White	Black	16	17	18
Breadboard (22 total)	9	13	20	2	14	7	1
BitBlox (22 total)	14	8	21	1	11	11	0

Table 1. Student demographic information.

Students	Had Experience	No Experience	Unreported
Breadboard (22 total)	6	15	1
BitBlox (22 total)	3	18	1

Table 2. Student response when asked about prior experience building circuits.

Selection and Participation of Children

The participants that completed the pre-study survey and class activities comprised of 44 high school juniors enrolled in a rural, southern US school. The majority of the students were 16 or 17 years old identifying as White or Caucasian (*see* Table 1). The study took place during a regularly scheduled Drafting class designed to introduce students to topics in engineering. All participants had Parent/Guardian Permission forms signed, and we walked them through the Assent form that explained the research study to them. They were then given the choice to opt out of the experiment or to sign the Assent form and participate. The majority of the students had never worked circuits before (*see* Table 2).

Class Structure

The study incorporated two classes—one class used BitBlox and one used the Breadboard. The students worked in pairs with a computer and a basic Arduino kit. There were 11 pairs on each day. Each class lasted 1hr and 40min, following an identical curriculum: surveys → presentation → simple circuit exercise → challenge circuit exercise. The presentation introduced the Arduino, explained either the BitBlox or the Breadboard, and walked through building a basic circuit for blinking an LED (Figure 4 (a)). In the final step, the pairs of students worked without direct guidance to build the challenge circuit (Figure 4(b)).

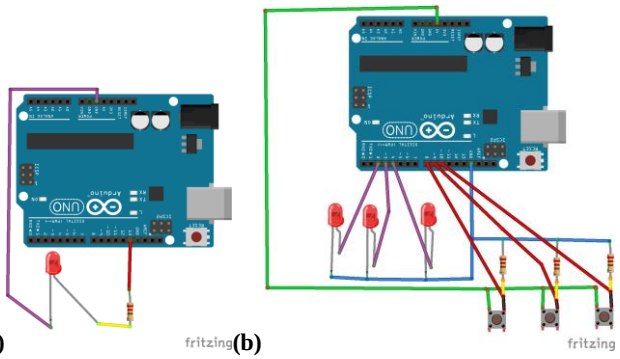


Figure 4. (a) The schematic for the introductory LED circuit. (b) The schematic for the Simon memory game circuit.

The challenge circuit involved building the circuit needed to create the Simon memory game. The game involves the computer randomly generating an ever-increasing pattern of blinking LEDs for the player to replicate by pressing buttons connected to the LEDs. The students had to construct the circuit and then upload the pre-programmed code that they received.

Data Collection

Data was gathered through pre-study surveys administered at the beginning of the class and through participant observations during the class periods. The pre-study survey asked students their age, ethnicity, gender, and if they had experience with building circuits, working with the breadboard, or working with the Arduino. Three researchers taught the lesson material, distributed the supplies in the class, and gathered the observations. This enabled them to be alerted when there were issues, and listen to each group explain their circuit. They agreed to split their attention on different sections in the classroom to ensure no group was left unnoticed. The researchers agreed to focus their observations within the broad categories of: mistakes,

misunderstandings, questions, collaboration patterns, moments when a feature in the tool seemed helpful or problematic, and patterns in tool use. They didn't pre-define a list of codes for the observations because this was a pilot study and sticking to a rigid list would run the risk of missing key observations. Each of the researchers took hand-written notes during the class, recording a diversity of observations throughout the class because they each worked with a variety of pairs. After each class, the researchers audio recorded a conversation amongst themselves discussing their written observations. They first talked about general observations of the class, and then addressed each group to discuss what each one of them found in their personal notes. This strategy ensured that the audio recording incorporated all of the data from each researcher.

Category	Description	Example
Tool Usage	How they used the Breadboard or BitBlox	Not connecting any of the BitBlox during their circuit construction
Student Hesitancy	Students demonstrating doubt and uncertainty	Wanting to be told if their approach was correct before building it
Student Agency	Students demonstrating self-efficacy/autonomy	Debugging circuit on their own
Concept Errors	Errors made based on not understanding basic concepts	Not being able to transfer a signal from one point in a circuit to another
Component Errors	Using components (ex. LED) incorrectly	Mixing up the button orientation
Tool Error	Errors with using or understanding the tool	Plugging both ends of an LED into one BitBlox module
Collaboration Within Pair	How students worked together in their pair	One student directing while the other was plugging in the wires
Collaboration Between Pairs	How different pairs interacted	One pair using a finished group's circuit for comparison
Work Strategies	Strategies students used to get the circuit working	Color coding of their circuit to the diagram

Table 3. Code Book with descriptions and examples.

Analysis

The audio recording of the researchers' conversation was transcribed and used for analysis because it contained all of the data from the observation notes and reflections made throughout the classes. The researchers independently coded the transcription for patterns in the categories defined above. They then met to agree on a final code book with appropriate labels, and settle discrepancies regarding transcribed text with different labels. Reaching a consensus, they solidified the codes (*see* Table 3). Observation notes for each code was then cross-referenced for patterns across and between conditions.

FINDINGS

Below we present the findings that are most relevant to how the tools affected the students and learning environment.

Circuit Layout

The students using the Breadboard only had small variations between the circuit layouts. The variations primarily encompassed using different combinations of rows or columns (*see* Figure 5 for an example circuit). The compact nature of the Breadboard made debugging the circuit difficult for the students. We observed students having difficulty tracing out their circuit for the instructors.

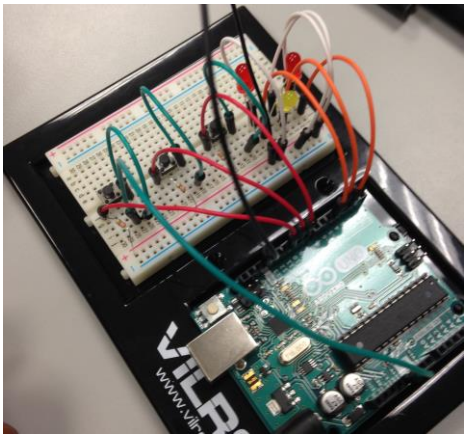


Figure 5. An example Breadboard circuit

The BitBlox allowed for more variation since there were different modules the students could use and different ways they could connect them. Figures 6, 7, and 8 show some of the various layouts from the different groups. Group F used many blocks and only connected a few (Figure 6). Two of the groups made an island of BitBlox for the buttons and an island for the LEDs (Figure 7). Other groups used many blocks but snapped them together making a circuit that was more compact and organized (Figure 8).

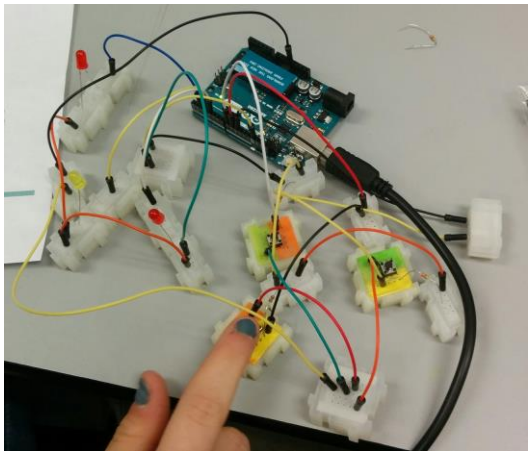


Figure 6. Group F's spread out circuit

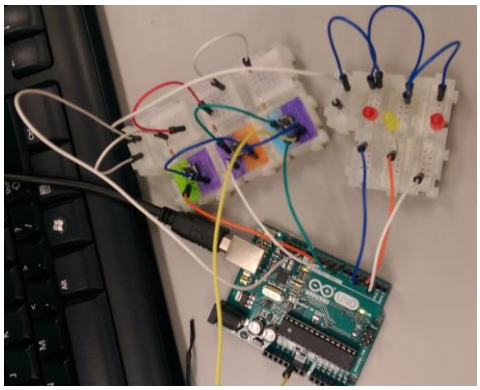


Figure 7. Group D's large organized circuit

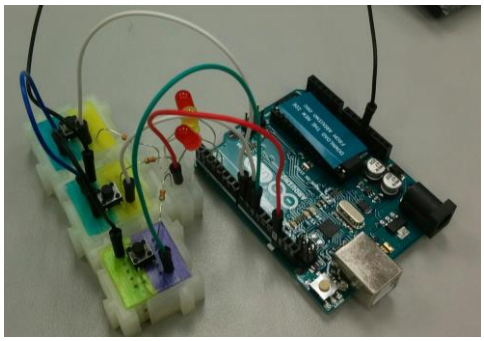


Figure 8. Groups A's small compact circuit

Tool Issues

Confusing the Breadboard row and column connections caused issues with all of the students. The researchers observed all 11 pairs mixing up the underlying connection schemes between the row and column. This error persisted throughout the length of the class, despite students hearing repeated explanations. For example, after running out of contact points to place wires in a column, group B began to group the wires in the columns next to it, as if being close together meant that they were connected.

After the Breadboard rows and columns were explained to them, they corrected this error and continued to construct their circuit using the columns properly, but incorrectly extended the column connection scheme to the outer rows of the Breadboard.

Observations showed the BitBlox connections caused confusion in four groups. In all four cases, they tried connecting a component with both sides into one block rather than connecting it across blocks or across colors.

The BitBlox enabled many block layout options, which led to decision paralysis in some of the students. For example, Group F was stuck for 10 minutes trying to decide between multiple ways of connecting the ground wire for the components. *"These are all connected, but should they be in one block or separate blocks?"* they asked. Even though they could build it either way, they wanted a straightforward solution. They finally decided to use separate blocks and ended up with the circuit in Figure 6.

Component Errors

The buttons were a pain point for almost all of the pairs in both classes. Because the buttons had four pins, students had trouble understanding their orientation. The students using the Breadboard struggled with connecting the button and then connecting wires to the button. All of the students had the opportunity to hold and examine an example button on the Breadboard enabling them to see how it bridges across two columns. Yet when applying that to their own circuit, many of them confused the *rows* and *columns* and had difficulties comprehending what happened to the connections when the button bridged the two columns.

Students using the BitBlox also had issues with connecting the button to the block because they confused the orientation. However, the students used the visual color cues on the block to fit it into their circuit properly.

Terminology

The majority of the students were novices, and learning new circuit terminology. We observed that in both conditions, students were initially confused with terms such as *resistor*, *LED*, and what *positive* and *negative* meant in this context. We observed additional confusion during the Breadboard class with the added terms of *row* and *column*. Further, students initially confused the '+' and '-' annotations on the Breadboard with the positive and negative sides of the LEDs. While these annotations did lead to confusion, we also observed that the annotations were a useful way for students to reference parts of their circuit as they were explaining them to researchers.

Other Circuit Building Issues

The researchers observed students struggling to transfer connections, such as power and ground, from one point in the circuit to another. At least six groups faced this issue with the Breadboard. This problem occurred at times when students exhausted all holes in a column and needed to connect more wires to it. Only two groups were observed struggling to transfer connections with the BitBlox. No students were observed running out of holes while using the BitBlox, which may be related.

When some students encountered an issue, instead of debugging their circuit, they took it apart and rebuilt it. Of the Breadboard groups four were observed using this technique, as opposed to one Bitblox group. Roughly the same number of groups in both conditions built their circuit color coded to the diagram—i.e. they matched the wire color to the diagram. Observations showed two Breadboard groups tried to map the number of wires to the schematic. This caused them to leave out wires because of the lack of a one-to-one correspondence between what they considered a separate wire in the diagram versus their actual circuit.

Collaboration

Some students disengaged from the circuit building activity, either not paying attention at all or merely observing and not contributing anything. However, this was more pronounced amongst students using the Breadboard. Five

groups using the Breadboard had at least one disengaged student as opposed to only one group using the BitBlox.

The Breadboard was more compact, so it was not placed in between the members, but instead resided in front of one person. This made it difficult for one student to see it as the other worked on it. A few groups were struggling with the close connections on the Breadboard. In one pair, the male student said to the female student, *"your fingers are smaller, maybe you can make this work"*.

Students using the BitBlox often had the modules physically placed in front of both people as they worked together. Each member of the pair was not necessarily plugging in wires at the same time, but they could clearly see and engage with what the other student was working on.

DISCUSSION

The findings from the comparative study between the Breadboard and BitBlox led to interesting design considerations for creating tools for building circuits.

Simplicity in Design

While neither connection scheme was flawless, we observed that the BitBlox's simpler connection scheme resulted in fewer user errors. The simpler scheme also meant that fewer terms were needed to describe the tool to the students. This proved beneficial for decreasing the complexity of the learning environment as the students had many other terms to learn and new concepts to understand.

Several groups were observed struggling with the underlying connections of the Breadboard, but far fewer had similar problems with the BitBlox. We attribute this to the simplicity of BitBlox design as it set a lower level of entry for novice students, allowing them to build the same complex circuits that are built on breadboards.

Affordances for Organization

Circuitry, even at a basic level, can be difficult to understand. Intermediates and experts become trained over time at understanding complex circuits and are able to use this skill when building and debugging them. To an untrained novice, these tasks can lead to frustrations. We noticed groups took apart their circuit instead of trying to debug it more often in the Breadboard groups than in the BitBlox groups. On both days, some students used the wires to color code their circuits; however, students using the BitBlox could snap together their BitBlox in different ways resulting in different organizations of their circuits. The differences in organization may help instructors discern students' misconceptions. These findings suggest that affordances for flexible organization within these tools can help students decrease the complexity of their circuits, and serve as a physical manifestation of their knowledge about their circuit that instructors can use for providing personalized assistance.

Designing for Appropriation

We observed that the BitBlox gave the students opportunities to create their circuit in many different ways. Instead of using a *one size fits all* model, the flexibility in the tool allows for greater diversity in learning approaches. This can empower learners by giving them greater control over their learning environment [7]. However, the limitations of useful flexibility should be further explored. It is possible that some students may end up more confused and hesitant about building circuits due to the extra degree of freedom. Without guidance, students may construct circuits that they themselves find difficult to debug making it harder for them to get assistance (*see* Figure 6).

Including Identifiers

We observed that annotations on the Breadboard caused confusion for some students, but it also served as a useful tool for students to describe sections of their circuit. This suggests that having identifiers on the tool that allow students to reference their circuit may be useful for collaborating with other students and the instructor.

Size of Tool

One of the main differences between the BitBlox and the Breadboard is the size of the circuit that students produce. This affected how easily students could work on their circuit and how they collaborated within their pairs. Consistent with research in collaborative learning, creating a larger tool allows *multiple access points* for students to work more seamlessly together [13]. Therefore, the number of students to be working together should be considered when determining the size for a tool in this space.

CONCLUSION AND FUTURE WORK

We presented a pilot study of BitBlox, a new tool for building circuits in a classroom. Through a qualitative comparative study we found it useful to consider the simplicity of the design, affordances for organization, affordances for appropriation, identifiers for referencing, and the size of the tool. Within our future work we intend to iterate on the design including modifications such as including identifiers on the BitBlox such that the individual blocks can be referenced more easily and more kinds of blocks, as the diversity in blocks seemed to help students.

We hypothesized that the increased complexity of the Breadboard causes a higher strain on the students' working memory. Throughout our investigation we found a greater confusion with the underlying design and terminology associated with the Breadboard in comparison to the BitBlox. This suggests that a further investigation of the cognitive load theory within this domain could prove useful. Through decreasing the complexity of the learning environment we hope to improve the learning experience for all students as they engage in physical computing.

ACKNOWLEDGMENTS

We thank the teacher and students who worked with us on this investigation. We also thank the NSF for grant DUE 1431984 that made this work possible.

REFERENCES

1. Alissa N. Antle, Alyssa F. Wise, and Kristine Nielsen. 2011. Towards Utopia: designing tangibles for learning. *Proceedings of the 10th International Conference on Interaction Design and Children*, ACM, 11–20.
2. Paulo Blikstein. 2013. Gears of our childhood: constructionist toolkits, robotics, and physical computing, past and future. *Proceedings of the 12th International Conference on Interaction Design and Children*, ACM, 173–182.
3. Paulo Blikstein and Arnan Sipitakiat. 2011. QWERTY and the art of designing microcontrollers for children. *Proceedings of the 10th International Conference on Interaction Design and Children*, ACM, 234–237.
4. Leah Buechley and Michael Eisenberg. 2008. The LilyPad Arduino: Toward wearable engineering for everyone. *Pervasive Computing, IEEE* 7, 2: 12–15.
5. Leah Buechley, Mike Eisenberg, Jaime Catchen, and Ali Crockett. 2008. The LilyPad Arduino: using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 423–432.
6. Joshua Chan, Tarun Pondicherry, and Paulo Blikstein. 2013. LightUp: an augmented, learning platform for electronics. *Proceedings of the 12th International Conference on Interaction Design and Children*, ACM, 491–494.
7. Gerhard Fischer. 2013. Meta-Design: Empowering All Stakeholder as Co-Designers. *Handbook of Design in Educational Technology*: 135.
8. Yoshiharu Kato. 2010. Splish: A Visual Programming Environment for Arduino to Accelerate Physical Computing Experiences. IEEE, 3–10.
9. Jane Margolis and Allan Fisher. 2003. *Unlocking the clubhouse: Women in computing*. MIT Press.
10. Amon Millner and Edward Baafi. 2011. Modkit: blending and extending approachable platforms for creating computer programs and interactive objects. ACM Press, 250–253.
11. Briana B. Morrison, Brian Dorn, and Mark Guzdial. 2014. Measuring cognitive load in introductory CS: adaptation of an instrument. ACM Press, 131–138.
12. Jie Qi, Andrew "bunnie" Huang, and Joseph Paradiso. 2015. Crafting technology with circuit stickers. ACM Press, 438–441.
13. Jochen Rick. 2012. Proportion: a tablet app for collaborative learning. ACM Press, 316.
14. John Sweller. 1988. Cognitive Load During Problem Solving: Effects on Learning. *Cognitive Science* 12, 2: 257–285.
15. John Sweller and Paul Chandler. 1994. Why Some Material Is Difficult to Learn. *Cognition and instruction* 12, 3: 185–233.
16. John Sweller, Jeroen JG Van Merrienboer, and Fred GWC Paas. 1998. Cognitive architecture and instructional design. *Educational psychology review* 10, 3: 251–296.
17. David Weintrop and Uri Wilensky. 2015. Using Commutative Assessments to Compare Conceptual Understanding in Blocks-based and Text-based Programs. ACM Press, 101–110.