Adapting the eBlock Platform for Middle School STEM Projects: Initial Platform Usability Testing

Anuradha Phalke and Susan Lysecky, Member, IEEE

Abstract—The benefits of project-based learning environments are well documented; however, setting up and maintaining these environments can be challenging due to the high cost and expertise associated with these platforms. To alleviate some of these roadblocks, the existing eBlock platform which is composed of fixed function building blocks targeted to enable nonexperts users to easily build a variety of interactive electronic systems is expanded to incorporate newly defined integer-based building blocks to enable a wider range of project possibilities for middle school STEM projects. We discuss various interface possibilities, including initial usability experiments, and summarize our overall experiences and observations in working with local middle school students utilizing the eBlock platform.

Index Terms—Devices for learning, learning environments, learning via discovery.

1 INTRODUCTION

The need to substantially increase the number of science, technology, engineering, and mathematics (STEM) professionals in the coming decades is one of the greatest challenges facing the US today [4], [30], [31]. Meeting that need requires innovations that increase children’s awareness, interest, and participation in scientific and technological fields. Paramount in this effort will be helping all students develop dispositions for engaging in the basic processes of scientific inquiry—questioning, observing, designing, investigating, and explaining. Beyond secondary schools, we see that engineering enrollment and retention at the college and university level is adversely effected [53], fewer students are pursuing science and engineering degrees with only about 6 percent of undergraduates majoring in engineering [13]. The US Bureau of Labor Statistics projects 15 of the 20 fastest growing occupations through 2010 will require significant math and science backgrounds [70]. These challenge are not unique to the United States, similar concerns are found in (but not limited to) Europe, Latin America, and Australia [2], [3], [20], [21]. Interestingly, students’ lack of interest in science [57] or attitude differences based on gender [25] are more prevalent in developed countries, driving numerous initiatives to change the way STEM education is approached.

The benefits of project-based learning (PBL) have been studied for decades [35], [40], [63], where students participate in the learning process rather than passively listening to a lecture, reading a book, or watching a video. Studies have shown that students who are able to investigate the world around them with topics that are relevant to their everyday surroundings, not only achieve higher scores than students in traditional classrooms [23], [69], but are also more engaged [11], [49]. PBL typically consists of a subset of the following features [7], [36], [37]: a driving question or real-world problem that motivates activities and discussion, opportunities for student to become active investigators by applying information or investigating possible solutions, collaboration among students and teachers to promote participation and develop the sense of a learning community, the use of cognitive tools such as computer-based laboratories, and opportunities for self-assessment and revision. One tool that can be used is a Driving Question Board (DQB) [36], [81] which organizes ideas and address problems experienced while working through the lessons. As students work on the posed problem, subquestions and/or central ideas are generated, branching off the main question. The DQB continually reminds students of the main goal they are working toward, evolution of their understanding, as well as what’s left to be figured out. Furthermore, by working together to address questions posed on the DQB, or reviewing solutions provided, students build a sense of community.

While these learning models have been shown to be effective, adopting these strategies into a real-world classroom often fails. Resources to integrate new advanced learning technologies into the curriculum typically do not exist, as test equipment is too expensive and school budgets are already spread thin [74]. Additionally, teachers face challenges in moving toward a PBL environment because significant changes are required in classroom organization, curriculum, and assessment while still having to balance existing district requirements, growing class sizes, and the diversity of student needs [7], [37]. Furthermore, many teachers are asked to teach subjects in which they lack proper training and self-confidence with inadequate resources available for professional development [12], [20]. Incorporating new platforms and instructional techniques on top of the existing core curriculum can quickly become overwhelming. Additionally, teachers struggle with motivating and sustaining students’ interest, promoting collaboration within groups, resolving problems with group dynamics, and helping student to develop well thought out designs [6].
We are faced with a challenge to develop a platform that is usable by a wide range of teachers and students, cost-effective for schools, and support a variety of STEM learning goals. We propose to utilize and extend the eBlocks platform \[16\] to provide an interactive and customizable PBL environment that allows students to see how the topics studied in class relate to real-world applications by enabling students to create their own customized systems. While the long-term goal is to develop a complete solution for the integration of the eBlocks platform into schools, in this paper, we focus on the extension of the eBlock platform for STEM education within middle schools.

2 EBLOCK OVERVIEW

The existing eBlocks platform \[16\], \[19\] emerged from a desire to empower regular people, having no programming or electronics experience, to build custom electronic systems. Rather than creating specialized systems for each application scenario, the eBlock platform is composed of commonly used building blocks that users connect together like Legos to form an application. The key to the eBlocks platform is to integrate compute intelligence with previously “dumb” sensors and actuators, as shown in Fig. 1a, to hide low-level interfacing details and communication between nodes so that users simply connect components together to specify the functionality of the application.

A variety of building blocks have been identified, with approximately 20 different blocks physically implemented. Individual blocks fall into two categories—fixed function or programmable. Fixed function blocks have a specific predefined function that may have slight configurability (i.e., which logical operation to perform, the length of time a signal is prolonged). A programmable block is a generic two-input, two-output block that a user can specify custom functionality through a graphical interface \[42\], \[43\]. As this block is targeted for more experience users, we omit further discussion and usage of the programmable block. The fixed function blocks can be further subdivided as follows:

- Sensor blocks—monitor the environment, including motion sensors, light sensors, buttons, contact switches, etc.
- Compute blocks—perform basic logic transformations (e.g., AND, OR, and NOT) or basic state functions (e.g., prolong, toggle, trip, and pulse).
- Output blocks—provide stimuli and include light-emitting diodes (LEDs), beepers, electric relays, etc.
- Communication blocks—provide wireless point-to-point communication or replicate a signal to send to multiple blocks.

eBlocks operate on Boolean data, meaning that blocks send and operate on “yes” or “no” packets. Applications are built by snapping blocks together and the order in which blocks are connected specifies the functionality of the system. Fig. 1c illustrates a garage-door-open-at-night system. The magnetic contact sensor indicates that if “yes,” the garage door is open or “no,” the garage door is not open. The light sensor indicates that if “yes,” there is light (daytime) or “no,” there is no light (nighttime). These values are fed into a two-input logic block that is configured to look for the combination “yes” the garage door is open AND “no” light. If this combination is detected, a red light is illuminated.

Extensive usability testing has been performed on Boolean blocks involving close to 500 participants \[14\], \[15\], \[16\], \[42\]. Findings indicate that over 50 percent of
participants, with little or no training, are able to successfully create a variety of applications within 10 minutes of being introduced to the platform. Boolean values, however, limit the types of projects that can be built, in particular STEM projects require building blocks that operate within the integer domain. Thus, to enable the use of the eBlock platform within STEM related courses, we must extend the current platform to incorporate integer-based eBlocks.

3 RELATED WORK

A central goal of our proposed work is to develop a set of building blocks that empower educators and students, who have no prior programming or electronics experience, to build custom interactive electronic systems. The underlying platform is transparent to users; they can simply connect blocks together like Legos to form an application. Over the past several decades, construction kits like electronic building blocks have gained increasing attention as powerful tools for learning. Logiblocs [38] and Electronic Blocks [83] consisting of small plastic blocks that young children snap together to build various systems that consist of light sensors, buttons, speakers, bleeps, LEDs, etc. Users connect blocks to build systems that produce the desired output based on various stimuli. Logidules [39] and MagicBlocks [33] are similarly block-based platforms but are intended for students who either already have some electronics background or are studying electronics. Thus, abstracting the underlying implementation is difficult and users must first understand the underlying platform before they are able to create customized systems.

At the top end of the K-12 spectrum, a technology-based construction kit is the centerpiece of The Infinity Project [14] that enables secondary school students to engage in engineering design and experimentation as they use hardware and software to design cell phones, internet applications, and audio, video, and graphics programs using computer-based graphical software design environment provides a workspace for students.

Perhaps the best known upper elementary and middle school construction kits currently available are MIT Crickets, based on the early pioneering work of Seymour Papert and the more recent work of Mitchel Resnick and colleagues at MIT’s Media Lab [45], [46], The Cricket construction kit is a collection of Lego bricks and electronics that include a tiny microprocessor block called a Cricket, motors, and sensors that can be used to build a variety of small robots. A key principle in Crickets is that people program them to perform a variety of functions. To make programming simple, the Logo language is used [64], [65]—a simple, graphical, highly intuitive language. Crickets provided the foundation for the Lego Mindstorm product [79], consisting of numerous sensor and actuator Lego blocks that can be connected to a central microprocessor block to build a variety of small robots, again programmed using a simple graphical language. Crickets provide a general programmable computer, requiring use of a programming language by users.

Programmable platforms that target the commercial domain have also appeared. One such platform is Phidgets [24], [44], [61] which consists of numerous electronic components such as force sensors, vibration sensors, RFID readers, LEDs, and servo motors that are connected to a central board. Users are able to connect Phidgets to a PC using the USB interface and program the desired application in a variety of programming languages using the provided Application Program Interface (API) library. The Motes and [17], [28] Smart Dust [72], [80] platforms similarly target commercial domains and consist of miniature wireless sensing devices incorporating sensing, communication, and I/O capabilities. While both platforms can implement an endless number of applications, implementation and optimization of these systems require extensive programming and electronic knowledge and currently remain a challenge to most engineers.

Robotics platforms are also a popular choice to get students involved in STEM. Researchers in Chile have explored using several different robotics platforms to increase technology literacy [68] including the BEAM platform [27], the Parallax Board of Education platform [58], and the LEGO platform [79]. iRobot’s Roomba vacuum has been utilized to enhance introductory programming courses at the university level [79] as well proposed for use in a variety of K-12 programs [47]. Another emerging platform is Eddy [10] in which the authors propose an open hardware paradigm where schematics, datasheets, and drivers for hardware are freely available, enabling participants to build their own robots by using components commercially available in electronic stores. Probably, the most popular platform is the Lego Mindstoms kits, with projects targeting a wide range of audiences [5], [34], [51], [56], [82].

The eBlock platform has several distinguishing features. First, users will design and build interactive systems that will not require knowledge of a programming language or any underlying platform-specific details. By eliminating these requirements, we can quickly engage students and teachers by considering new application scenarios and developing solutions within minutes, eliminating the need for tedious and time-consuming training. Second, this platform is approachable to students who may be reluctant to learning a programming language or find building custom electronics platforms intimidating. While a few plug-and-play platforms exist, the types of sensor, control logic, and actuators limit the type and scope of application possibilities, whereas our approach will integrate more robust and complex components while ensuring usage by nonexperts. Third, the platform will be low-cost and well suited for resource-strapped educational settings, especially classrooms where computer access is limited or nonexistent, because the platform will not require communication with a computer.

4 INTEGER-BASED EBLOCK PLATFORM

eBlocks have the potential to enhance STEM education by providing students with the opportunity to build customiz-able and interactive projects that complement the specific topic studied without the burden of first having extensive programming or electronics experience to learn how to utilize the platform itself. Studies have shown that in PBL environments, students’ interest, motivation, and engagement are increased [11], [20], [49] and that projects which integrate real-life problems affecting students’ everyday lives can help to grasp the value and meaning behind the
topics investigated [26], [41], [50]. Thus, we set out to define an eBlock Education Kit to target STEM education by providing the underlying platform to enable students to build a variety of interactive systems. We first consider middle school STEM education with future extensions reaching to the high school and university level.

To begin, we needed to identify which building blocks to include in the eBlock Education Kit so that students are able to build a wide variety of applications. At the same time, we must balance the number of components included as not to overwhelm students with a gigantic catalog of eBlocks. By looking at the curriculum listed on the local school district’s Webpage [76] as well as several other websites that provide descriptions of middle school science fair projects [55], [62], [67], we were able to identify the math and science topics covered at this level. We then tried to take an existing project and determine how an electronic system could be integrated. For example, instead of manually measuring temperature, we utilized the eBlock platform to automate the sampling and logging process. Alternatively, we may start with a general lesson plan and try to come up with an interactive project that could complement the lesson plan. For example, when studying topics related to motion and forces, an interactive lab may have students set up a system to automatically start and stop a timer to record how long an object takes to traverse different surfaces. As these applications are defined, we are able to identify a new set of integer-based eBlocks to integrate into the existing Boolean platform. As new applications are developed, we must then reevaluate the existing set of building blocks to see if additional blocks are needed.

4.1 Design and Development of Integer eBlocks

A key design criterion for all eBlocks is that each block should be self-explanatory. Users should be able to easily understand what each block does merely by reading the name of the block or possibly by reading a very short (about 10 word) description included on the front of the block. All of our experiments have confirmed to us that participants often do not read or understand instructions, even when written on the back of each block. Instead, participants ignore the documentation provided in favor of a trial-and-error methodology to understand the functionality of each building block. These experiences are also consistent with other studies indicating that users disliked reading even the shortest instructions and would rather work things out for themselves [22], [71].

The initial attempt to expand the existing Boolean eBlock platform has yielded the addition of six new integer-based eBlocks, as shown in Fig. 2. The temperature sensor block, depicted in Fig. 2a, senses and outputs the current ambient temperature. The constant block, as represented in the Fig. 2b, consists of two rotary switches that the user can adjust to select a constant value output by the block. The threshold block, as illustrated in Fig. 2c, accepts two external inputs values A and B. This block additionally includes a three-position slide switch to enable a user to configure the block to determine if \( A > B \), \( A = B \), or \( A < B \) depending on the position of the switch. The inputs to the threshold block are integers, while the output of the threshold block is a Boolean “yes” or “no” value indicating the outcome of the comparison performed. The range block, as shown in Fig. 2d, is a multifunction block which accepts three external inputs, A, B, and C. Additionally, the range block contains a three-position slide switch to enable a user to configure the range block to detect whether \( A < B \), \( B < C \), or \( A < B < C \). The inputs to the range block are integer values, while the output of the range block is a Boolean “yes” or “no” value indicating the result of the range operation. The timer block shown in Fig. 2e has two inputs, start and reset, and one output value referred to as output. The timer block activates an internal counter when the start input receives a “yes” value and deactivates the internal counter when the start input receives a “no” value. The timer is reset to an internal value of zero when the reset input receives a “yes” value. If both the reset and start values receive a “yes” value, the reset input has priority. Additionally, the frequency in which the timer block updates the internal counter is specified by the user and can be configured to track milliseconds, seconds, or minutes. Inputs to the timer block are Boolean values, while the output of the timer block is an integer value. Lastly, the integer display depicted in Fig. 2f receives an integer value on the input port and displays the corresponding value on an LCD screen integrated within the block. The range of the display block is \(-999 \text{ to } +999\). Usability testing of these newly added integer blocks is provided in Section 5.

4.2 Project Booklet

As we discover the project possibilities, we are also compiling a project guide booklet which serves several purposes. First, this booklet provides introductory materials illustrating how to power on blocks, connect blocks together, and simple experiments that direct students to investigate what various fixed function blocks do. The idea of scaffolding [8] is utilized in which student gradually learn how to utilize the eBlock platform through a series of smaller tasks, with the hope of building to more complex tasks. Second, this booklet is intended to provide teachers and students with sample systems that can be implemented with the eBlock Education Kit. Each project begins by posing a question. For example, why don’t oceans freeze as quickly as lakes? Next, steps are outlined to help students connect

\[
\begin{align*}
A &< B \\
A &> B \\
A &\leq B \\
A &\geq B \\
A &\neq B \\
A &\equiv B
\end{align*}
\]

Fig. 2. Newly integrated integer-based eBlocks include a (a) temperature sensor, (b) constant block, (c) threshold block, (d) multifunction range block, (e) timer block, and (f) integer display block.
various eBlock components together to build a project that tests the posed question. Fig. 3a illustrates how temperature sensor and integer display eBlocks can be snapped together to observe the freezing point between containers of fresh and salt water. Students are then asked questions that require interaction with the platform, such as placing both containers into a freezer and recording the temperature of each container every 10 minutes. Additionally, a series of follow-up questions are provided, requiring students to reflect on the experiment. Fig. 3b illustrates the setup for another experiment intended to monitor the effects of temperature and moisture on mold. While the project guide booklet provides numerous projects, it serves only as a starting point for students and teachers to create their own projects. Lastly, these project booklets can provide additional support materials such as a frequently asked questions section, debugging, and troubleshooting methods, as well as background materials relating to STEM topics that can provide motivation for a series of experiments.

Currently, the project booklet contains introductory materials explaining how to utilize the eBlocks platform, ideas on how to present and organize project information such as utilizing a DBQ or engineering notebook, a small number of sample projects, and a general description of what engineering is and what are the everyday tasks of an engineer. Support materials are being added as we begin to observe common problems encountered by users. We also plan to add more background materials as we increase the number of sample projects.

5 EXPERIMENTS

The addition of integer-based eBlocks increases not only the number of application possibilities, but also increases the complexity of the applications themselves. In the following sections, we present usability experiments with both university and middle school students to evaluate: 1) if the functionality of individual building blocks is easily understood from the interface, 2) if users can determine which blocks are needed, as well as how to connect blocks, to implement the desired application functionality, and 3) whether user can come up with their own application ideas using eBlocks.

5.1 Usability Interface Testing for Integer eBlocks at the University Level

Utilizing the new integer-based eBlock prototypes, we initially conducted a series of informal experiments with university students to see what aspects of the eBlock Education Kit were readily understood and what aspects were confusing [59], [60]. Participants were categorized into two groups: nonexpert and expert. Nonexpert participants are students at the university who have no engineering background, and include majors such as pharmacy, accounting, or political science. Expert participants are students who have had at least one semester in an engineering major such as computer engineering, chemical engineering, or hydrology. All participants worked individually and were provided with a handout containing a brief introduction to eBlocks, as shown in Fig. 4. The flyer quickly illustrates how blocks are powered on, connected to one another, and communicates with one another. Additionally, participants were provided with a set of physical blocks and an eBlock catalog containing descriptions of each block’s functionality and interface. These handouts were the only materials provided to participants; we did not provide an oral introduction or answer any questions during testing.

Participants were easily able to construct simple systems composed of sensor and output blocks, indicating a basic understanding of the eBlock platform. More complex

![Fig. 3. Sample eBlock projects. (a) eBlocks are used to observe the freezing of fresh and salt water. (b) eBlock temperature sensors are used to observe the effects of temperature and moisture on mold.](image)

![Fig. 4. A one-page starter guide is provided to participants to quickly illustrate how eBlocks are powered on, connected to one another, and how to determine what the blocks are communicating to one another.](image)
systems, which mixed both Boolean and integer blocks, yielded slightly lower success rates. Forty percent of the nonexpert participants were able to successfully construct a system which tracks the amount of time a light is left on, with another 40 percent close to correct with a minor error in the connection of blocks. Sixty seven percent of the expert participants were able to successfully build the same systems, with another 33 percent close to correct. On average, experts finished constructing both systems in 8 minutes, while nonexperts required 20 minutes.

5.2 Usability Interface Testing for Integer eBlocks at the Middle School Level

While the initial experiments with university students provided a glimpse into the usability of the newly expanded eBlock platform, these participants were not the target audience. Thus, to truly ensure successful use of the platform, we needed to get eBlocks into the hands of middle school students. In the Fall of 2008, two local middle schools became involved in the eBlock project with participants ranging from the sixth to eighth grade, and unlike the university testing, students were able to work with the eBlock platform for three to five 1-hour sessions. Due to the limited number of kits available, participants worked in five-six groups, with a maximum of four students per group. In addition to the introductory flyer shown in Fig. 4, the first sessions with students provided a 30 minute introduction to eBlocks, specifically demonstrating how to power on blocks, how to connect blocks together, the meaning of tiny status lights located on the front of each block, and the functionality of all the blocks. The following sections provide a breakdown of the testing performed at both middle school locations, the resulting usability of various blocks, as well as insights gained from interacting with these students.

5.2.1 Boolean Blocks versus Integer Blocks

The integration of integer-based blocks is intended to expand the class of applications that can be implemented utilizing the eBlocks platform. However, with the addition of integer blocks, the construction of eBlock systems become more complex. Not only are there a larger number of building blocks to choose from, but the underlying communication changes. In Boolean eBlock systems, all blocks communicate using “yes” and “no” packets, while integer blocks need to handle numerical values. Instead of limiting eBlock systems to strictly Boolean blocks or strictly integer blocks, users should have the flexibility to create heterogeneous systems composed of both Boolean and integer blocks. Thus, the existing Boolean blocks have been updated to handle integer values. When a zero is received, the packet is interpreted as a “no,” and when values greater than zero are received, the packet is interpreted as a “yes.” Integer blocks similarly accept Boolean values, “yes” packets are interpreted as a value of one, and “no” packets are interpreted as a value of zero.

To determine if users are able to comprehend the updated protocol, participants were first asked to build a simple system that sounds a buzzer then a button is pressed. This system is composed strictly of Boolean blocks and 100 percent of users are able to successfully create the desired application, as shown in Table 1. Next, participants were asked to build an alarm to detect when a nonzero number is detected. The resulting system is a heterogeneous application composed of a constant block (integer-based block) connected to a buzzer block (Boolean-based block). Again, all users are able to successfully create the desired application. Additionally, when students were asked why the beeper sounded for certain inputs but remained silent for other inputs, participants were able to explain that nonzero values were converted by the beeper block as a “yes” and the beeper block turned on. However, a zero value is converted by the beeper block to a “no” and the beeper block turns off. Participates were additionally asked to construct a system to sound an alarm when a value of zero is detected (instead of a nonzero value as specified by the second system). In this system implementation, an inverter block is needed between the constant and beeper blocks. While 60 percent of participants were able to successfully build the desired system, many groups struggled with how to modify the system to meet the new specification. From discussions with participants, it seems that the difficulty with the third application centered on the identification and use of the intermediate block rather than the combination of Boolean and integer packets utilized by the application.

5.2.2 Timer Block

Many of the applications surveyed in Section 4.2 contained the concept to time. One such application may determine the speed at which a ball travels given different inclines. The system would need to track the time taken for the ball to travel between designated starting and ending points, and given the distance traveled a student could then calculate the average ball speed. Other such applications may include measuring velocity, acceleration, time required for different liquids to boil, hours of sunlight versus dark, and so on. Thus, an important addition to the eBlock platform is the timer block, as shown in Fig. 3e.

Aside from specifying the functionality of the timer block, we are also interested in determining how best to design the timer block interface. Fig. 5 illustrates two interface possibilities. In Fig. 5a, a sentence-like structure is utilized. In previous usability experiments, we found that students were better able to specify Boolean expressions with sentence-like interfaces versus table-like interfaces [14]. An alternative interface is also provided, as shown in Fig. 5b, with a slight change in wording.

One group of students utilized the first interface to construct a system that monitored how long a magnet is

<table>
<thead>
<tr>
<th>Application Scenario</th>
<th>% Success</th>
<th>Total Participants</th>
</tr>
</thead>
<tbody>
<tr>
<td>System 1 - Activate beeper when button is pressed</td>
<td>100</td>
<td>24</td>
</tr>
<tr>
<td>System 2 - Activate beeper when the number is not 0</td>
<td>100</td>
<td>24</td>
</tr>
<tr>
<td>System 3 - Activate beeper when the number is 0</td>
<td>60</td>
<td>24</td>
</tr>
</tbody>
</table>
near a proximity sensor by connecting the magnet sensor eBlock to the start input of the timer eBlock. When the magnet is within a predefined range, the timer block is activated and counts up. When the magnet is farther away, the timer stops counting. An integer display block is connected to the output of the timer block to view the internal value held by the timer block. Additionally, a button block is connected to the reset input of the timer block to zero out or restart, the internal time value held by the timer block. As shown in Table 2, 100 percent of the participants were able to successfully create the desired application. A second set of participants was provided with an alternative timer interface, as shown in Fig. 5b, and asked to construct a system to detect the length of time a button is pressed. The underlying implementation of this system is identical to the first scenario, with the exception that the magnetic sensor block connected to the start input of the timer block is replaced by a second button block. The change in application scenario was an attempt to make the desired system functionality more clear, as well as have students use button blocks instead of magnetic sensors as some students needed clarification on the use of proximity sensors. Again, 100 percent of these users were able to successfully implement the desired system functionality.

In speaking with participants, they readily understood both timer block interfaces. Participants were able to figure out that the “start” input controlled whether the timer counted or not. Additionally, participants were easily able to adjust the granularity of time tracked by utilizing the slide switch to select the desired option. Several groups utilizing the second timer block interface initially omitted the button block connected to the reset input of the timer block, but were able to figure out the missing connection required to reset the count back to zero. Moreover, one group was able to optimize the system implementation by utilizing a single button. The button was directly connected to the “start” input, while the same button was inverted before connecting to the “reset” input, thereby eliminating the need for a secondary button.

Based on the usability results, the middle school participants were able to easily utilize the timer block, regardless of the interface provided. We observed that the actual wording of the interface as well as the instructions provided on the back of blocks were generally ignored. Instead, participants looked for keywords, such as the name of the block or the name of the interface signals, to understand the functionality of the block. Furthermore, by eliminating an additional “stop” input from the timer block interface utilized with university students, the average success rate increased from 55 to 100 percent.

5.2.3 Threshold Block
Another common operation found in a variety of applications is the comparison of two values. In Boolean-based systems, two values can be compared utilizing the two-input logic (or combine) block configured to detect the desired relationship. For example, to detect if two Boolean values are equivalent, logic blocks are configured to detect $A \oplus B$ (i.e., XNOR gate), to determine if $A < B$ the logic block is configured to detect $A \land B$, or if the user wants to detect $A \geq B$, the logic block is instead configured to detect $A \lor B$. However, once the eBlock platform is extended to the integer domain, a new block capable of performing integer comparisons is required. It is not feasible to utilize a logic block to detect if ambient temperature is greater than 85 degrees, or sound an alarm when the number of people in room is greater than 50. Thus, the threshold block was introduced as a possible block to perform integer comparisons.

Two interface possibilities are examined, as shown in Fig. 6. The first threshold block interface utilizes a sentence-like structure to indicate the functionality of the block. The second interface, as shown in Fig. 6b, is updated slightly to indicate that a “yes” is only transmitted when the input conditions are true. Participants were asked to construct three systems: one system to detect if the temperature in a room was greater than 60 degrees $F$, another to detect when temperature was less than 90 degrees $F$, and the last system to detect when the temperature is equal to 76 degrees $F$. One group of participants utilized the first threshold block interface, while another group utilized the second interface. In each of the systems, participants needed to attach the temperature sensor block to one of the inputs of the threshold block and a constant block to the other input. Then, the slide switch needed to be adjusted to the “<,” “>,” or “=” setting depending on which comparison was

<table>
<thead>
<tr>
<th>Application Scenario</th>
<th>Interface 1 % Success</th>
<th>Interface 2 % Success</th>
<th>Number of Participants</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build a system that will count time when the magnet is near</td>
<td>100</td>
<td>100</td>
<td>22</td>
</tr>
<tr>
<td>Create a timer system that will show how long you are pressing a button</td>
<td></td>
<td></td>
<td>12</td>
</tr>
</tbody>
</table>
needed. Lastly, the output of the threshold block was connected to an output device to alert user when the specified scenario occurred, a beeper, LED, or integer display can be utilized.

Again, we observed that most students ignored the sentences provided on the face of the threshold block, and instead, tried to determine the block functionality by utilizing the switch and input labels. On average, 87 percent of users were able to successfully identify which block was needed to perform a comparison operation as well as select the appropriate setting to configure the threshold block to. However, as detailed in Table 3, interface 1 clearly outperformed interface two yielding success rates of 100 and 80 percent, respectively. We observed that the groups who struggled with the second interface often interchanged the A and B inputs (i.e., instead of detecting $\text{temperature} > 60$, the system was configured to detect $60 > \text{temperature}$). At this point, it remains unclear whether the interface itself posed the problem, or if determining how to setup the comparison based on the system description was difficult. Further investigation is needed. Lastly, based on the feedback provided from teachers and students, the naming of the threshold block will be updated to comparison block, to more readily impart the block’s functionality.

### 5.2.4 Range Block

An alternative type of comparison operation is range detection. For example, a user may want to see if the temperature in a fish tank or greenhouse is within a recommended range. If the temperature fluctuates beyond the specified limits, an alarm is activated or a thermostat is automatically adjusted. Thus, the range block depicted in Fig. 3d is introduced. The range block can be configured to detect $B < C$, $A < B$, or $A < B < C$. Instead of creating single purpose block that only detects if an input is within a predefined range, we decided to incorporate additional comparison possibilities in hopes of reducing the total number of blocks needed within the kit.

Fig. 7 illustrates two possible user interfaces. Similar to the threshold block design, a sentence structure is utilized, with a small change between the two interfaces wherein the first interface points the user to the status lights to determine the outcome of the comparison performed, while the second interface simply reminds users that a “yes” is transmitted with the comparison is true. Participants were asked to develop an application to determine if temperature detected was greater than 60 degrees Fahrenheit but less than 90 degrees Fahrenheit. One group utilized the interface shown in Fig. 7a, while the second group utilized the interface in Fig. 7b.

While the successes rates presented in Table 4 indicate that the first range block interface is superior, this is not necessarily the case. Most participants were readily able to select which block was needed to perform the range comparison. However, determining how to connect the various input blocks proved problematic. To create the desired system, two constant blocks are required. One block is connected to input A and specifies the low bound (60 in this example), while a secondary constant block is connected to input C to specify the high bound (90 in this example). Additionally, a temperature sensor is needed to detect the current temperature of the room and connected to input B of the range block. Many students tried to utilize two temperature sensors and one constant block. Others were unsure of which blocks should be connected to which inputs. We believe that the discrepancy is more closely related to the students’ experience levels than difference between the interfaces provided. The first group of participants had a stronger knowledge of math and was
able to translate the textual system description into an equation. While the second group of participants struggled to convert the problem statement into an equation and was unsure of even which input blocks were needed to implement the desired functionality. Additional testing is required to determine how best to reduce the ambiguity of the range block. Future iterations could update the interface to include arrows to correlating inputs to the comparison performed, updating the relative placement of inputs and switch, removing the additional comparison options, or by simply renaming the block.

5.2.5 Comparison of Threshold and Range Blocks
Currently, comparison operations can be performed using either the threshold or range block. Fig. 8 demonstrates how both types of comparison can be accomplished by using only threshold blocks or using only range blocks. A comparison can be implemented using the threshold block, as depicted in Fig. 8a, by simply connecting both inputs to the desired values and configuring the block to either detect greater than, less than, or equal. Alternatively, a comparison can also be implemented by utilizing the range block, as demonstrated in Fig. 8b, by connecting two out of the three inputs of the range block. Depending on which two inputs are connected, a less than or a greater than operations is performed. The second type of comparison operation of interest is range detection. Again, this function can be implemented utilizing either the threshold or range block. If a user is limited to threshold blocks, the range operations can be implemented by utilizing two threshold blocks and a two-input logic block, as shown in Fig. 8c. The first threshold block evaluates whether $2 < 5$, while the second threshold block evaluates $5 < 7$. To determine if both comparisons are true, the outputs of both threshold blocks are connected to a two-input logic block, outputting a true value only when $AB$ is true (both input comparisons are true). Similarly, a range operation can be performed by using only range blocks, and results in a much simpler system as indicated in Fig. 8d. To detect range utilizing a range block, all three inputs are simply connected, with the block configured to detected $A < B < C$.

Because the threshold and range blocks have overlapping functionality, providing that a single block to perform both a comparison and range operation will not only reduce the number of blocks users need to choose from, but reduce the cost of the overall kit. However, the usability of the eBlock platform as a whole remains the top priority. From the usability experiments conducted in Sections 5.2.3 and 5.3.4, it seems that the threshold block is a more user friendly choice, while the range block seems more difficult for users. However, we must not only consider the usability of an individual block, but how well these blocks are utilized in a variety of applications. Fig. 8 clearly demonstrates that range detection requires a more complex setup when using threshold blocks versus the correspond setup using a range block. While the comparison operation yields similar complexity when either a threshold or range block is utilized, the connections to the threshold block are less ambiguous. Previous simulator-based testing [42] indicates that users preferred blocks that most closely matches the desired functionality. Overall, participants who were asked to construct a range operation experienced higher success rates when range blocks were utilized instead of threshold blocks, 53-13 percent, respectively. Similarly, participants who were asked to construct a comparison operation experienced higher success rates when threshold blocks were utilized instead of range blocks, 84-51 percent, respectively. Preliminary results indicate that providing a larger set of straightforward fixed-function blocks, even when overlapping functionality exists, is preferred to provide a smaller set of blocks that require clever uses of these blocks. As the physical platform continues to expand, we will see whether this trend continues.

---

**TABLE 4**

<table>
<thead>
<tr>
<th>Application Scenario</th>
<th>Interface 1</th>
<th>Interface 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build a system that will alert you when the room temperature is more than 80°F and less than 90°F</td>
<td>75</td>
<td>12</td>
</tr>
</tbody>
</table>
5.2.6 Open-Ended Project

In the previous usability experiments, participants were always provided with application scenarios and their task was to construct the corresponding eBlock system to meet the desired functionality. After participants had a chance to acquaint themselves with the eBlock platform, and see potential uses for these blocks, they were provided with an opportunity to define their own application. Participants were provided with a kit consisting of 11 blocks, including but not limited to a temperature sensor, button, timer, two-input logic, threshold, range, beeper, and LED blocks and given 45 minutes to design systems for home or school use. However, participants were encouraged to develop any system they thought would be useful, and did not need to strictly adhere to the aforementioned specifications. Table 5 indicates a subset of systems created by participants, including a breakdown of system characteristics. Specifically, the number of sensors utilized, the complexity of the system created in terms of the longest path between a sensor and output node, the number of integer and Boolean blocks integrated within the system, and whether the timer, threshold, or range block was incorporated into the end application. Initially, when presented with the open-ended project, most users began with simpler systems consisting of a single input and output block. Such systems included a light detector that connected a light sensor to an integer display, a weather station composed of a temperature sensor and integer display, or a magnet detector that libraries could utilize to detect whether a book was checked out before leaving the premises. Several groups graduated to more complex systems involving both integer and Boolean blocks. These groups were extremely creative and able to invent a wide range of applications, including a missile system which would deploy the rocket at the end of a count down tracked by the timer block. Another group developed an alarm system which would sound a beeper when a prespecified temperature was exceeded for longer than 60 seconds. In observing these participants, we found that they were able to determine their own uses for the building blocks provided, as well as develop an assortment of application possibilities. Among the integer blocks, the threshold and timer block were commonly utilized. While no groups utilized the range block, this is not surprising as the initial usability results showed many users struggled with usage of this block.

5.2.7 Observations and Experiences

While platform development and testing is still early, we can see an increase in the confidence level among the student participants and remain optimistic that the eBlock platform has the potential to enhance STEM education. Initially, students were extremely timid in utilizing the building blocks and many students thought because the blocks looked similar, they all had the same functionality. All blocks, with the exception of the timer and integer display, were prototyped to fit into small 3.6" x 2.6" x 1.1" black plastic boxes. While each building block was labeled and a description of the functionality provided on the back of each box, students tended to ignore these. Instead, in order to determine what each block did, students connected blocks together and observed the resulting action given different input conditions. Furthermore, while block labels were color coded to indicate an input, output, or intermediate block, this difference was not noticed by students who instead tried to connect two inputs or two outputs together. While the connectors were specifically chosen such that these connections were not physically possible, student still tried to force the connectors together. As students worked with blocks, they eventually understood the difference between these blocks types and could explain the purpose of input, intermediate, and output blocks. Additionally, while the small debugging lights (green, yellow, and red status lights) on each block were initially provided to help us test and debug blocks during the development stage, we notice that students also used these lights to determine the status of individual blocks as
well as trace the system behavior to determine at what point the functional derivates from the desired behavior.

After observing and speaking with student and teachers, we found that students were very enthusiastic about creating their own interactive systems with eBlocks. After several sessions, timid users grew more comfortable with the eBlock platform and indicated that they especially enjoyed the open-ended project where they were able to create self-directed systems with varied complexity and functionality. Several students asked how to obtain their own kit to take home and utilize. One participant talked about these blocks so much at home, we later found out that their parents obtained a commercially available electronic kit for this participant to use. While we have not yet conducted formal surveys with student after using the eBlock platform, most students were able to identify tasks related to engineering and a few students even expressed interest in pursuing engineering.

While the initial usability experiments focused on evaluating various block interfaces, we remain cognizant of the need to provide support materials for teachers and their involvement in developing project modules. As we approached several afterschool programs, we found that while teachers embraced the idea of using new interactive technologies within their programs, they were extremely wary of presenting the materials themselves. Most teachers asked if there would be someone from the university to present the materials and oversee students while utilizing the eBlock platform in case any problems arose. Additionally, as we explained the long-term goal of utilizing the eBlock platform within classrooms, teachers were unclear in how these building blocks related to existing course materials. Thus, we have found the importance of providing a comprehensive project booklet to facilitate the introduction and use of the eBlock platform as well as the need to work with teachers to determine the feasibility of using these project ideas in a real-world classroom setting.

Lastly, while students were successfully able to determine the functional identity of individual blocks and construct larger systems through trial-and-error, in many instances, there did not appear to be an organized plan to approach the problem proposed. Integrating a section on the basic process of scientific inquiry [52], which includes making observations, planning experiments, and how to collect and interpret data, would be a helpful addition to the project booklet. Students utilizing the scientific inquiry method have shown improved retention [1], better performance, and positive attitudes about learning science [18], and asked more reflective questions [9].

6 CONCLUSIONS AND FUTURE WORK

To stimulate interest and improve performance in STEM education, we have proposed the use of the eBlock platform to enable the construction of interactive projects. The usability experiments have demonstrated the feasibility of the eBlocks platform within middle schools. On average, 89 percent were able to successfully utilize the newly incorporated integer-based eBlocks. While the temperature sensor, timer, threshold, and integer display blocks were easily understood by most participants, users struggled with the range block. Participants obtained an average success rate of 51 percent for range-based applications, mandating the need for future updates.

Future iterations of the platform will include expanding the number of sensors available, as well as incorporating additional building blocks to enable users to further manipulate integer values. We are also working with colleagues specializing in education to determine how best to overcome the difficulties associated with PBL technologies, incorporating the eBlock platform in existing STEM courses, as well as developing mechanisms to monitor long-terms goals such as measuring changes in attitudes or determining any increases in computational thinking skills. As the platform and support materials mature, we also plan to test the use of the eBlocks platform within a classroom setting.

References


B. Gammon, “Everything We Currently Know about Making


---

Anuradha Phalke received the BE and ME degrees in electronics from Shivaji University, Kolhapur, India, in 2004 and 2006, respectively, and the MS degree in electrical and computer engineering from the University of Arizona in 2008. Her research interest includes the development of interactive platforms for use within middle school settings, to expose students to STEM-related topics, and increase interest in engineering. She is currently working as a systems engineer for LSI Corporation in Colorado Springs.

Susan Lysecky received the MS and PhD degrees in computer science from the University of California, Riverside, in 2003 and 2006, respectively. She is currently an assistant professor in the Department of Electrical and Computer Engineering at the University of Arizona. She coordinates research efforts for the Ubiquitous and Embedded Computing Lab, and her current research interests include embedded system design with emphasis on self-configuring architectures, human-computer interaction, and facilitating the design and use of complex sensor-based system by nonengineers. She is a member of the IEEE and ACM.