Optimising microcontroller based educational packages for educational purposes: A case study using the Parallax Stamps In Class Kit.

Benjamin Babcock

Project report submitted in part fulfilment of the requirements for the degree of Master of Science (HF/HCI) in the Faculty of Science, University of London, 1999.

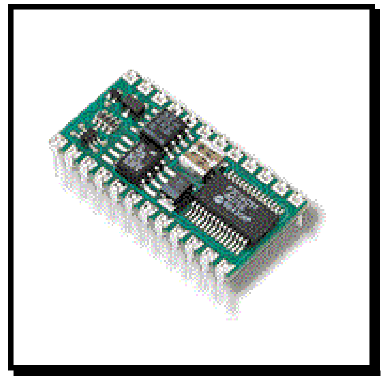
Abstract

This project explores a curriculum's ability to provide insight into the functionality of a microcontroller based educational system. As a case study, the project worked exclusively with the Stamps in Class Educational Kit. Human factors evaluation methods were used to assess the curriculum's ability to provide insight into the functionality of the system. The factors preventing users from gaining insight into the functionality of the system were isolated and discussed. Finally, solutions were provided in an attempt to eliminate or reduce the effect of the factors preventing users from gaining insight into the functionality of the system.

Naturalistic observation and objective textual analysis revealed that four factors prevented users from gaining insight into the functionality of the system. First, frequent errors prevented users from completing the lessons. Second, a lack of text devoted to discussing the functionality of the system. Third, the invisibility of the system prevented users from observing the system in operation. Lastly, the limitations of a paper based curriculum prevented users from understanding the complex functionality of the system.

Based on knowledge gained from observational analysis, objective textual analysis, and existing research, solutions were provided in an attempt to eliminate or reduce the effect of the factors preventing users from gaining insight into the functionality of the system. The project concludes with a discussion of potential future work, the implications of the findings, and solutions.

Table of Contents



Basic Stamp II
Microcontroller

Section 1: INTRODUCTION

1.1 Background

Few would have thought there would ever be a day when Human Factors (HF) would reach the mysterious, elusive, and powerful microcontroller. Microcontrollers are essentially miniature computers. In fact, many are about the size of a coin. Using input and output lines, they are capable of sensing and interacting with the environment. For example, a microcontroller responsible for regulating lighting levels in a diverse, dynamic office building may be used to collect information from various photo sensors. In turn, based on the photo sensor input and the program stored in the controller's memory, the microcontroller may output current that reduces or increases lighting levels respectively. Appendix A illustrates this application.

Recent trends in education have brought microcontrollers to the classroom. Over the past few years, many schools have begun using microcontrollers as educational tools in science and mathematics teaching. Various researchers have studied the effectiveness of implementing microcontrollers as educational tools. Likewise, various companies have begun developing microcontrollers suitable for educational purposes. However, little HF input has been offered to the design of such devices. Thus, this research will be concerned with the application of HF substantive and methodological knowledge to support the design of microcontroller based educational packages. In particular, Parallax's Stamps in Class materials will be used as a case study. As more and more microcontrollers are implemented in our schools, a need has grown to understand the role human factors may play in their development.

1.2 What is a Microcontroller?

Essentially, microcontrollers consist of a processor (ALU), a memory store (ROM), input and output lines, and a program. The ALU, or arithmetic logic unit, is the portion of the controller responsible for making decisions based upon input and outputs and how they are handled by the stored program. ROM, or read only memory, stores program information to be run by the ALU. As previously mentioned, input and output channels are responsible for sending and receiving current through various external components. Programs, stored in ROM, are responsible for controlling the behaviour of the controller and how it interacts with its environment. Given their size, processing capabilities, and cost, they are used in many situations. They are used in alarm clocks, video cassette recorders, microwaves, radios, factory control operations, telephones, toys, robotics, and a host of other devices. In contrast to desktop PC systems, microcontrollers have no built-in human interface components. That is, they have no keyboard, mouse, monitor, or other device facilitating human interaction. There has been little need for human interface components as microcontrollers are generally embedded within some form of process control, void of human interaction. In fact, only during program development and transfer, is human interaction required or necessary. Besides this, only one copy of the program need be developed because the same program can then be mass copied onto as many controllers as necessary.

In the past, working with microcontrollers required advanced programming skills and a solid understanding of electronics and electricity. They were usually programmed in Assembly language, C, or other low-level, complex languages. Given the skill and knowledge required to work with microcontrollers, successfully programming and implementing a microcontroller usually required a degree in Electrical, Mechanical, or Robotics engineering. Desktop tinkerers and hobbyists were left unable, without intense self-training, to develop the skills necessary to effectively interact with microcontrollers. Thus, the diversity of the users of such technology was extremely limited. Further, until recently, microcontrollers were one-time programmable. That is, a program could only be written to memory once during the life span of the microcontroller. These factors limited the amount of time users spent interfacing with microcontrollers.

Recently, companies have made attempts to redesign microcontrollers to facilitate interactions with a more diverse user population. One company in particular, Parallax Inc, has made great strides in reducing the problems associated with interacting with microcontrollers of the past. Parallax developed a family of microcontrollers called Basic Stamps. Basic Stamps are low cost, high performance microcontrollers with rewritable memory. They do not require knowledge of traditional programming languages. Instead, they can be programmed using a language very similar to the popular programming language, BASIC. Aware of the benefits of using a language similar to BASIC, Parallax developed Parallax Basic (PBASIC). Similar to the original BASIC, it has been considered a fairly easy to learn natural language programming language. It is considered ideal for beginners and untrained programmers. This effort to use a language already widely used and accepted reduces the learning curve for new users.

Secondly, the Parallax family of microcontrollers is designed with a re-writable memory system, allowing the same controller to be programmed over and over again. This feature supports increased interaction with users as it encourages users to spend more time programming and reprogramming. Parallax also developed an external device designed to reduce the effort required during program development, download, and component interface, called the Board of Education. The Board of Education serves as a platform through which the Basic Stamp may be interfaced. It contains a small breadboard for projects and prototyping. The breadboard provides a surface in which components (resistors, capacitors, motors, et cetera) may be connected to the input and output lines of the Basic Stamp. The Board of Education also contains a serial adapter which allows easy connection to the computer's serial port for program download. Also, Parallax has developed a version of the Stamp Editor to be run in the Windows 95 environment. Likewise, the PBASIC language has been continually refined based on technological advances and improved features.

In an attempt to make their product available for educational purposes, Parallax has developed the Stamps in Class program. The Stamps in Class program was developed to promote the use of Basic Stamps in the classroom to teach students about microcontrollers, electricity, and electronics. It has been designed for users of fifteen years of age and above. The Stamps in Class program distributes educational kits including the following items:

- Software
  - Stamp Editor (Win 95 or DOS based)
- Hardware
- Basic Stamp I or II
- Board of Education
- Components
  - Resistors
  - LEDs
  - Potentiometers
  - Capacitors
  - Timer

Photo Resistor
Switches
Servo Motor
Jumper Wires

Curriculum

What's a MICROCONTROLLER? Student Workbook V.1.4

The software, Stamp Editor for Windows 95 or DOS, bears no difference from other versions of the stamp editor software. The Board of Education is the same as the version available to regular customers. Though, regular customers, with greater expertise may find the Board of Education too limited in its ability to support prototyping aside from the projects listed in the What's a MICROCONTROLLER curriculum. The curriculum can be found in Appendix B. The design of the Board of Education supports the curriculum. Next, the components have been specially selected to support the curriculum. The components can easily be plugged into the prototyping area of the Board of Education. The components are standard electronic components purchasable through any major electronics vendor. Lastly, the curriculum consists of an introduction to Microcontrollers, a brief introduction to their architecture, and finally a series of experiments. Each experiment features circuit diagrams, representational diagrams, discussions, code listings, as well as learning questions. The curriculum is linear in that it must be read in succession to complete the experiment. However, as reference, it may be used in a non-linear fashion. The language level may not be appropriate for users younger than the specified age.

Recent investigations into the appropriateness of using microcontrollers in science and mathematics in grades K-12 (ages 5 – 18), has revealed important insight into the age at which microcontrollers may be effectively implemented (Smith 1997). Dr. Mike Smith and his research assistants explored how science education can benefit from the role of the Basic Stamp in the classroom. He and his researchers concluded that the ideal grades to introduce such a learning tool would be in Division II (grades 3 - 6) for mechanical systems and simple data entry, Division III (grades 7 - 9) for understanding primary electronic components and systems, and division IV (grades 10 - 12) for more advanced and long term applications (Smith 1997). The scope of this research will be confined to the application of microcontrollers in grades 4 (age 10) and above. In particular, it may be used to facilitate the teaching of basic electronic principles. For example, circuits, resistance, electricity, magnetism, and so forth.

A wealth of other advanced microcontroller based educational tools exist. For example, researchers at MIT have been developing a new family of microcontroller based educational tools called crickets (Resnick 1997-1999). Crickets are fairly similar to the popular “programmable Lego bricks” previously developed at the MIT Media Lab (Martin 1994 and Sargeant et al 1996). However, crickets are much smaller and more powerful than their predecessors. At about the size of a 9 volt battery, they can control motors, receive information from sensors, and communicate with one another using infrared communications. Like Basic Stamps, they are fully programmable from a desktop computer. Crickets have been developed with the goal of making it easier for students to write and understand control and sensory oriented programs. Like other microcontroller based educational tools, little HF input has been considered in the design of Crickets. Crickets, still in the development process, have been successfully implemented in grades 4 and above.

Furthermore, Programmable Bricks, BitBalls, Digital Beads, and Thinking Tags are all examples of recent attempts to bring microcontrollers into the educational setting. Many are aimed at users of 10 years and above. These, and other recent pieces of research associated with implementing microcontrollers in the classroom, justifies the need for HF consideration in the design of such educational devices. As the diversity of the user population, in terms of age, gender, and so forth increases in variety, the need for HF intervention increases.

As technologies in the classroom become more and more complex, the role of teaching curricula has become increasingly important. Often, technologies come with curricula that teach users how to use the device. However, the curriculum does not teach users how the device actually works. That is, they do not provide users with insight into the functionality of the system. If such curricula gain widespread favour, then implementing such technologies will result in masses of students who know how to use advanced technologies but don’t understand how they work. With this in mind, it is hoped that implementing such technologies in the classroom would provide users with insight into how the technologies work, not simply how to use them. Because they are small, complex, and their behaviour is invisible during operation, Basic Stamps create a formidable challenge in providing insight into their functionality.

1.3 How do users develop insight into how a system functions?

Psychological research has generated a wealth of knowledge about how people learn. To understand how learners develop an understanding of how to use technologies as well as understand how they work, we must reference current views related to learning. In particular, a wealth of knowledge exists about how people develop mental models. However, before discussing mental models, we must consider the important concept of schemas and scripts.

Information that is organised around a central concept or topic is called a schema. In long term memory, schemas form a knowledge structure about a particular topic. People store schemas for most of the items in their world, including the systems and equipment they use. Schemas that describe the steps required to complete a task are called scripts. For example, a person may have a script that defines how to use a cash machine, or ride a bus. Scripts are often procedural in that they simply contain information pertaining to how to complete a task not necessarily how the task works. For instance, a person may have a script for how to use a cash machine without any understanding of how the cash machine actually works. In terms of the Stamps in Class curriculum, students acquire scripts from the experiments. That is, they learn how to perform a sequence of actions to achieve a desired result. In the first experiment, users are provided with a script about how to perform a task using the microcontroller, Board of Education, a program, and an LED to create a blink sequence. The curriculum provides a detailed script allowing users to easily complete the task. As you can see, completion of script does not require much insight into how a system functions. Instead, a student could, essentially, thoughtlessly follow the presented sequence to complete the task.

Schemas that describe how things work are called mental models. Mental models are basically schemas about equipment or systems. They consist of the knowledge about how a system or piece of equipment functions. Research has revealed that “mental models typically include our understanding of system components, how the system works, and how to use it.” (Wickens 1997). Thus, developing a mental model of a system implies a deeper level of learning. Scripts can be developed strictly based on memorisation of a sequence of steps to complete a task or use a piece of equipment. In contrast, developing a mental model requires a deeper level of understanding with a combined knowledge of how the system functions as well as how to use it and its components. However, the importance of scripts cannot be underestimated. As Wickens pointed out, part of developing a mental model, or understanding of how a system functions, involves learning how to use a system. The lessons in the curriculum can be considered scripts. They provide users with detailed information about how to use the equipment. Thus, the curriculum’s ability to provide users with insight into functionality involves careful consideration of its ability to aid users in correctly completing the experiment.

Thus, in terms of the Basic Stamp kit, it is hoped that the curriculum would provide students with the opportunity to develop script-based knowledge about how to use the system as well as develop mental models of how the system actually functions. In other words, it is hoped that the curriculum would teach users not only how to use the system (script) but also how it works (mental models).

1.4 Research objectives

Motivated by a lack of Human Factors (HF) involvement and an increasing amount of microcontroller based technologies in the classroom, this research will focus on evaluating the curriculum responsible for providing users with insight into how the system functions. HF knowledge will be used to evaluate an existing microcontroller based educational system as well as support design changes aimed at improving the existing system for use in educational settings. To achieve this goal, Parallax's "What's a Microcontroller?" curriculum will be used as a case study. Thus, this research will focus on exploring the "What's a Microcontroller?" curriculum and its ability to provide users with insight into the functionality of the Basic Stamp educational kit.

In order to achieve this goal, the existing curriculum will be evaluated using naturalistic observation and textual analysis of the content of the curriculum. From these evaluations, the factors inhibiting the curriculum from providing users with insight into the functionality of the system will be isolated. Upon isolation and analysis of these factors, solutions will be provided for each of the factors. Solutions will be aimed at directly eliminating the inhibiting factors or at least reducing their effect. From these solutions, design recommendations will be developed and discussed. Thus, this research will focus on establishing and improving the curriculum's ability to provide insight into the functionality of the system.

The following steps will be taken to evaluate and improve the curriculum's ability to provide insight into the functionality of the system.

1) Evaluate existing curriculum
2) Assess curriculum' ability to provide insight into the system
3) Isolate factors decreasing curriculum's ability to provide insight into the functionality of the system.
4) Provide solutions for factors decreasing curriculum's ability to provide insight into the functionality of the system
5) Generate redesign recommendations based on established solutions

1.5 Goal of the Curriculum

The goal of the "What's a Microcontroller?" curriculum is clearly stated in the preface of the curriculum. That is, "the Stamps in Class curriculum is designed to introduce students and teachers to microcontrollers using software basics and simple hardware" (Parallax, Inc. 1999). Further, it has a "purpose of providing an entry-level background to microcontroller programming and interfacing . . . combining software and hardware, first showing the student how to build the circuit, then program the microcontroller, and finally challenging them to improve the design" (Parallax, Inc. 1999).

Essentially, the kit seeks to provide users with insight into how microcontrollers function. Interfacing microcontrollers requires combined insight into how the program and hardware function. Thus, to achieve this goal, students must learn how microcontrollers, programs, circuits, and components function. Thus, the body of the curriculum should support students in acquiring insight into the functionality of the system. The following section describes the methods that will be used to explore the current curriculum's ability to provide insight into the functionality of the system.

1.6 The experiment

The experiment has been designed to assess the curriculum's ability to provide insight into the functionality of the system. That is, assess the curriculum's ability to provide insight into how the system works. It must be noted that, normally, the curriculum would not be solely responsible for providing insight into functionality. Under normal circumstances, the hardware as well as the software would be considered vital to the task of providing insight into functionality. However, one of the constraints of the research is that the hardware and software cannot be altered to increase functional transparency. This is due to the fact that it would be terribly expensive to alter the physical appearance of the hardware. Likewise, it would be terribly expensive and disruptive to change the existing programming environment used by the Basic Stamp. Thus, because of the ease with which the curriculum may be altered to support the revelation of insight into the functionality of the system, the experiment will focus exclusively upon the curriculum.

Understanding the functionality of the system requires that users develop an understanding of the microcontroller, program, circuit, and components. The experiment has been designed to explore the effectiveness of the current curriculum in terms of its ability to provide insight into how the system functions. The system may be defined as the microcontroller, program, circuit, and components (resistors, LEDs, and Board of Education). Thus, an effective curriculum should reveal insight into how the system (microcontroller, program, circuit, and components) functions.

Section 2: PROCEDURE

2.1 Review of analyses

First off, an objective textual analysis will be undertaken. The textual analysis will focus on locating portions of text that directly discuss functionality pertaining to Microcontrollers, Programs, Circuits, and Components. The total number of words devoted to discussing functionality pertaining to the aforementioned categories will be tallied and compared to the total word count to generate data that presents the distribution of text devoted to each category. The introduction and experiment one found in the "What's a Microcontroller?" (Appendix B) curriculum will be used to generate the objective textual analysis data.

Secondly, an observational evaluation of users interacting with the Stamps in Class Curriculum will be performed. Video analysis as well as verbal protocols were used to obtain data. Observations were task based in an attempt to determine how users tackled the tasks given, and where the major difficulties lie. Observations lasted approximately one hour, allowing users to complete the first lesson in the curriculum. This portion of the research will facilitate in the process assessing the curriculum's ability to provide insight into the functionality of the system. To achieve this goal, the evaluation focused on locating problems with the current curriculum that prevented users from completing the experiments and gaining insight into the functionality of the system. Secondly, by means of the chapter-end and experimenter generated questions, the evaluation assessed the current curriculum's ability to provide insight into the functionality of the system.

2.2 Procedure for the observational evaluation

For the observational evaluation, participants were asked to complete experiment 1 found in What's a Microcontroller? Student Guide Version 1.4, pages 5-21. Prior to commencement of the experiment, students received a brief introduction to the research at hand as well as a brief introduction to microcontrollers. The following text details the dialogue of the introductions.

2.2.1 Research introduction

"Hello. I will be exploring how kids work with microcontrollers. In a moment, I will explain what a microcontroller is and how it can be used in the classroom. I'm interested in understanding what sort of problems you may have while you complete an experiment designed to help you understand what microcontrollers are and how they work. I will be here to answer any questions you may have while completing the experiment. It would be most beneficial if you could share your thoughts with me aloud as you work through the experiment. I am most interested in what you have to say about the microcontroller and lesson not whether or not you complete the experiment."

2.2.2 Microcontroller introduction

"Microcontrollers are basically tiny computers. Though, they do not have a key board or monitor as you would expect. They are used in robots and other things to control motors and lights and things like that. They have special pins or channels that allow the microcontroller to send and receive electricity to and from the world. Reading the "What's a Microcontroller? lesson should help you understand a little better what microcontrollers are and how they work."

Participants were then asked to read pages 5 and 6 for an introduction to microcontrollers. Then, they were asked to complete experiment 1 found on pages 7 through 17 of the curriculum. The framework for the experiment is clearly outlined in Appendix C

Using the curriculum (pages 7-17), participants were asked to follow the curriculum to complete the experiment. For purposes of evaluation, experiments were viewed as consisting of three parts: Hardware construction, Program construction, and Program alteration. Successful completion of the experiment were considered in terms of the participant's ability to accurately complete and comprehend the three tasks.

The first task, Hardware construction, required users to make use of the curriculum to arrange the components (resistors, LEDs, and leads) on the breadboard as shown in the pictorials and described in the text. The second task, Program construction, required users to make use of the curriculum to enter code to create a program capable of illuminating an LED from pin 0. At this point, the curriculum's ability to provide users with insight into the functionality of the system was assessed. The following questions were provided by the curriculum as a set of chapter-end questions. The questions were used because they shed light upon how well users understood the functionality of the system.

How does a microcontroller differ from a computer?
What is the difference between hardware and software?
Why is a microcontroller like your brain?

2.3 Program alteration

The third and final task, Program alteration, was designed to reveal insight into how well the children understood how the system works. Program alteration required users to make use of the curriculum as well as their understanding of the hardware and program to successfully alter the code to create a blink sequence in which pin 0 illuminates an LED for 1 second and pin 1 illuminates an LED for 1 second. This portion of the evaluation was carried out as specified in the following text:

"Now that you've successfully written a program to run one of the LEDs for 1 second, try to write a program that will run both LEDs. Make the LEDs blink like a police car with one of the LEDs going on for 1 second and then the other going on for 1 second. Feel free to use the curriculum or ask questions at any time."

Upon completion of the third task, evaluations were brought to an end and the nature of the experiment was disclosed. The following text presents the final dialogue of the evaluation.

"Thank you for participating in my research. The goal of my research is to look at how kids use microcontrollers. I want to know what kind of problems kids have when they use microcontrollers. I'm interested in making microcontrollers easier to use so that they can be used in schools like yours to help kids learn about electricity, computers, electronics, robotics, and other fun stuff."

Section 3: RESULTS and DISCUSSION

3.1 Interpretation of objective textual analysis

The goal of the objective textual analysis was to explore the ratio of text that provides insight into functionality versus text that does not provide insight into functionality. Text that provides insight into functionality has been called functional text. Such text may discuss how a particular aspect of the system functions or may provide insight into how the entire system functions. In contrast, non-functional text may be defined as text that does not directly render insight into how the system functions.

Mentioned in the introduction in section 1.3, this portion of the evaluation was guided by existing knowledge about schemas, scripts, and mental models. In this portion of the evaluation, text that was related to the development of scripts was called non-functional text, while text related to the development of mental models was called functional text. Text related to script development was referred to as non-functional because such text does not directly discuss an aspect of the functionality of the system, thus it is non-functional. In contrast, text that directly discussed the functionality of the system, or an aspect of the system, was deemed functional text.

Examples of functional and non-functional text have been provided in the following excerpts. An example of functional text pertaining to the functionality of the Board of Education was presented on page 9, "What's a Microcontroller?"

"It's important to understand how a breadboard works. The breadboard has many metal strips which run underneath in rows. These strips connect the sockets to each other. This makes it easy to connect components together to build an electrical circuit.

To use the breadboard, the legs of the LED and resistor will be placed in the sockets. These sockets are made so that they will hold the component in place. Each hole is connected to one of the metal strips running underneath the board. You can connect different components by plugging them into common nodes."

Clearly, this portion of functional text reveals insight into how the breadboard actually functions. The text helps users develop a mental model of how the breadboard works. Thus, this portion of text serves as an example of functional text. In contrast, the following excerpt demonstrates non-functional text. An example of non-functional text pertaining to LED's was presented on page 8, "What's a Microcontroller?"

"There are two very important things to remember when connecting LED's to the Basic Stamp. The first is always be sure that there is a resistor connected, as shown in Figure 1.3 below. In this experiment the resistors should be rated at 470 ohms, ¼ watt.

Secondly, be certain that the polarity of the LED is correct. There is a flat spot on the side of the LED that should be connected as shown in Figure 1.3. If the polarity is reversed, the LED will not work. The flat side also has the shortest LED lead."

The non-functional text listed above discusses two important components (LED's and resistors) used in the experiment. The text was deemed non-functional because it did not discuss the functionality of LED's and resistors. Rather, it simply provided factual information that did not directly contribute insight into how the components function. Rather than contribute to mental model development, the text helped users to develop a script for how to correctly construct the circuit.

Objective textual analysis revealed that 82% (3012 words) of the text was deemed non-functional, or text that was not directly related to discussing functionality. This fact, in itself, does not necessarily indicate an insufficient amount of words spent discussing functionality. However, it does provide insight into the nature of the text. That is, an overwhelming portion of the text does not directly reveal insight into how the system functions. Instead, it spends much of its time focusing on teaching users how to use the equipment rather than how it works.

In contrast, 28% (1188 words), or roughly a quarter, of the text directly discussed how the system functions. From these data it is clear to see that much of the text is not directly related to discussing functionality. The following chart presents the distribution of the 28% (1188 words) of the text devoted to discussing functionality. The chart presents the distribution of text discussing functionality grouped by four categories. The first category, Microcontrollers, presents the amount of text devoted to discussing how microcontrollers function. The second category, Program, presents the amount of text devoted to discussing how programs function. The third category, Circuit, presents the amount of text devoted to discussing how Circuits function. The fourth and final category, Components, presents the amount of text devoted to discussing how the components function. The Components section was further dissected into LED, Resistor, and Board of Education. Each subsection presents the amount of text devoted to discussing how each component works.

Categorical Breakdown of Text Discussing Functionality

Microcontrollers

Words	199
Percent of Total	4.7%

Program

Words	774
Percent of Total	18.4%

Circuit

Words	0
Percent of Total	0%

Components

Resistor

Words	0
Percent of Total	0%

LED

Words	7
Percent of Total	.004%

Board of Education

Words	208
Percent of Total	4.9%

*Percentage of total was calculated from the total word count for Experiment 1, "What's a Microcontroller?," 4200 words.

Only 4.7% (199 words) of the text was devoted to discussing how microcontrollers function. This relatively small amount of text devoted to discussing microcontroller function is quite surprising given the fact that one of the central goals of the curriculum is to provide insight into how microcontrollers function. Surprisingly, 18.4% (774 words) of the text was devoted to discussing how programs function. Again, this is quite surprising given the fact that one of the central goals of the curriculum is to provide insight into how microcontrollers function. This suggests the notion that understanding microcontroller functionality means understanding program functionality. None of the text was devoted to discussing circuit functionality. Circuits are mentioned on many occasions through out the text. However, there is no independent discussion of circuit functionality. In terms of components, none of the text was devoted to discussing how resistors function. Instead, the text simply discusses factual information related to resistors. For example, the text discusses the importance of using an appropriate resistor to prevent damage to the LED. However, it does not discuss how a resistor actually functions. Nor does it provide the information elsewhere in the text. Next, .004% (7 words) of the text was devoted to discussing how LEDs function. It must be noted that additional information was provided in an information box (What's a . . . Page 8). However, the information was not included because users were unable to comprehend the terminology of the given text. Interestingly, 5% (208 words) of the text was devoted to discussing how the Board of Education functions. Much of the text in this section was devoted to discussing how the breadboard functions.

The objective textual analysis demonstrated the fact that much of the text does not directly contribute to helping users develop an understanding of how the system functions. Further, a large majority of the 25% of functional text is devoted to providing insight into only of the aspects of the system, Program functionality.

3.1.2 Implications of findings from objective textual analysis

It cannot be safely concluded that simply because of the unequal proportion of functional and non-functional text, that the non-functional text must be eliminated. The non-functional text is vital to completing the experiment safely and correctly while the functional text is vital to providing insight into how the system actually functions.

In terms of scripts and mental models, the non-functional text facilitates script development teaching users how to use the kit. The functional text provides users with mental models helping users understand how the system functions. The problem is not that there is too much non-functional text. Rather, that there is too little functional text. To illustrate this point, we shall consider two extreme-case curricula - One in which all of the text is functional and another in which all of the text is non-functional.

First off, consider a "What's a Microcontroller?" curriculum in which all of the text is functional. The curriculum would provide page after page of information about microcontrollers and how they work. In careful detail, it would describe how LED's, resistors, and et cetera work. However, being strictly devoted to providing insight into how the system functions, it would never teach users how to actually use microcontrollers. Users would be able to describe how microcontrollers, programs, circuits, and components work but would be unable to actually use them to complete the experiments.

In contrast, consider a "What's a Microcontroller?" curriculum in which all of the text is non-functional. The curriculum would provide page after page of instructions and details that do not render insight into how the parts work. Following the curriculum, users would be able to successfully complete the experiments, however, they would be unable to understand how the system actually functions. Such a curriculum would be more of an operating manual than a teaching curriculum. Using the Stamps in Class Kit would be much like using any other household electronic device that carefully hides its function in a black box and detailed script. Essentially, using the kit would provide little more educational benefit than teaching our next generation of young people to do what we never could, program a VCR to record our favourite sitcom.

Thus, considering the two extreme-case curricula, it is clear to see that there is a distinct need for both functional and non-functional text in a teaching curriculum. Further, in cases such as the Stamps in Class curriculum, in which the curriculum acts as a teaching tool, the distribution of functional and non-functional text is a serious concern. That is, too much functional text renders a curriculum that does not facilitate completion of experiments but does render insight into how the system functions. Or, too much non-functional text that renders a curriculum capable facilitating completion of experiments, but does not render insight into how the system actually functions. The "What's a Microcontroller?" curriculum has unequally distributed the amount of functional and non-functional text. The curriculum is closer to an operating manual than it is to a teaching curriculum. Thus, improving the current kit requires a more equal distribution of functional and non-functional text.

Next, careful examination of the extant functional text was performed. Analysis of the distribution of functional text pertaining to the individual categories (Microcontrollers, Programs, Circuits, and Components) revealed an unequal distribution of functional text favouring program functionality. Nearly all of the functional text provided in the curriculum is devoted to revealing insight into program functionality. On the other hand, less than 5% of the text is devoted to revealing insight into Microcontroller functionality. It seems that the curriculum is catered to users with an interest in understanding program functionality. Completing experiments requires users to work with components, circuits, the Basic Stamp, and programs. Different users have different interests and talents. The current curriculum has catered its teaching to support users with an interest in programming and program functionality. The unequal distribution of functional text leaves users without programming experience or without an interest in programming at a disadvantage to completing and enjoying the experiment. A user without an interest or talent for programming may have tremendous talent for working with and understanding circuits, components, or microcontroller functionality. Such users would not share equal benefit from the curriculum. In order to be of equal advantage to a diverse user population with differing talents and interests, the curriculum should distribute the amount of functional text equally amongst the four categories - Microcontrollers, Programs, Circuits, and Components.

3.2 Findings from naturalistic observation

3.2.1 Classification of observed errors

Errors and mistakes were recorded throughout the duration of the naturalistic observation. Careful review of the video recorded sessions also contributed to the observed errors. Before continuing, it is important to establish a classification system for the observed errors. In order to best interpret the captured errors, we shall consider Donald Norman's approach to error classification. Donald Norman described errors as the distinction between the operator's intentions and his or her actual behaviour, where the result is undesirable. He described errors as resulting from slips and mistakes. He defined slips as errors in which the user's intention is correct for the situation, but the execution is incorrect. Slips are commonly caused by memory lapses which may cause the user to forget an important step. Or, a slip may be something as simple as literally a "slip" in which the user presses an incorrect key due to a slip of the finger.

Next, he defined mistakes as errors in which a user carries out an inappropriate intention. Mistakes are often the result of misunderstanding or confusion. For example, a mistake may result when a user has an inappropriate mental model of how a system works. Operating on this inappropriate mental model, the user then carries out an inappropriate action resulting in an error (Wickens 1999). Mistakes suggest a deeper problem within the design of an interface while slips often times suggest a more superficial problem. For example, in this experiment, many participants made errors resulting from keyboard slips. Considering the fact that many of the users have extremely limited experience with computers, such slips should be expected. However, when users made errors when attempting to alter the existing code to create a new behaviour, a deeper, more serious problem came to light. That is, users did not understand how the system works and thus made mistakes due to their confusion and misunderstanding of system functionality. Thus, in order to gain insight into the nature of the actual observed errors, errors will be classified as either slips or mistakes.

3.2.2 Connecting Slips and Mistakes to Scripts and Mental Models

Slips and mistakes parallel the concepts of scripts and mental models. Slips are often the result of failing to correctly perform a script. Mistakes are often times the result of inappropriate mental models. Correctly completing an experiment requires users to perfectly complete the sequence of steps (script) presented in the curriculum. Likewise, developing an understanding of the functionality of the system requires users to develop appropriate mental models of how the system functions. When users develop inappropriate mental models of the system, they make mistakes. Likewise, when users fail to complete a script correctly, they make slips. Thus, the following errors represent instances in which users were unable to complete the experiment (script) and instances in which users were unable to gain insight into how an aspect of the system functions.

3.3 Users were observed making the following errors

3.3.1 Participants were unable to construct hardware

Participants were unable to identify and accurately connect components to the breadboard to construct the hardware necessary to complete the experiment. In particular, users experienced the following errors.

Users had difficulty in locating proper components when relying solely upon curriculum.

a) Determining resistor types from black and white pictorials (mistake)

Many participants were unable to determine which resistor type to use for the experiment. The text does not provide sufficient aid in helping participants select the appropriate resistor type. The following excerpt presents the only supporting text for selecting resistor types for the experiment.

"There are two very important things to remember when connecting LED's to the BASIC Stamp. The first is always be sure that there is a resistor connected, as shown in figure 1.3 below. In this experiment the resistors should be rated at 470 ohms, ¼ watt."

The text does not take advantage of the available colour coding scheme to aid resistor selection. Resistors are always colour coded. The colour codes indicate the corresponding ohm and watt characteristics of the particular resistor. The colour coding scheme is presented in Appendix C, and in colour, on the back cover for reference. The errors resulting from inappropriate resistor type selection were classified as mistakes as they were the result of confusion and misunderstanding. That is, the text presents the resistor information in a way that resulted in mistakes leading to errors in resistor selection. Thus, the fact that this problem was classified as a mistake means that this particular problem threatened the user's ability to develop a mental model about this particular aspect of the system. Without an understanding of the important function of resistors in the circuit, users frequently committed this mistake.

b) Participants failed to correctly connect LED’s to breadboard (slip)

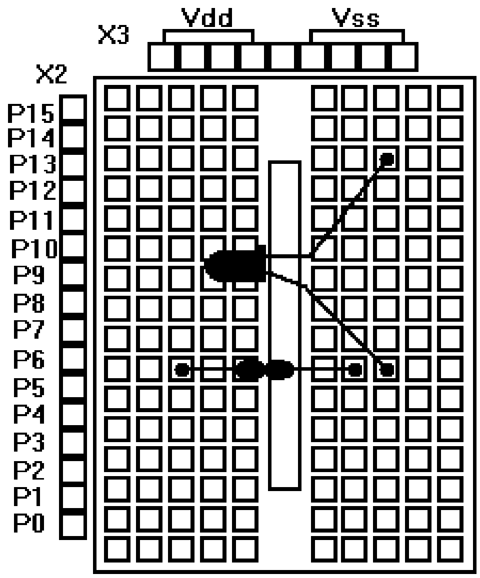
Next, Participants failed to correctly connect LED’s to the breadboard. The LED’s used in the experiment are manufactured with a flat spot to indicate polarity (direction of current flow). The LED must be positioned correctly, relative to the power source, to allow illumination. The curriculum clearly discusses the importance of this fact in the following excerpt.

“Secondly, be certain that the polarity of the LED is correct. There is a flat spot on the side of the LED that should be connected as shown in [the figure]. If the polarity is reversed, the LED will not work. The flat side also has the shortest LED lead.”

Despite this set of instructions, users incorrectly inserted the LED into the breadboard. Upon questioning, many participants claimed that they forgot to pay attention once they began inserting the component. Thus, the fact that the users had appropriate intentions, but merely forgot to pay attention to the polarity of the LED means that this error can safely be classified as a slip. Thus, this problem was due to the user’s inability to correctly complete a script. That is, users were unable to correctly complete the sequence of steps required to correctly connect the LED to the breadboard.

c) Users were unable to insert components into breadboard slots as shown in the pictorial (slip)

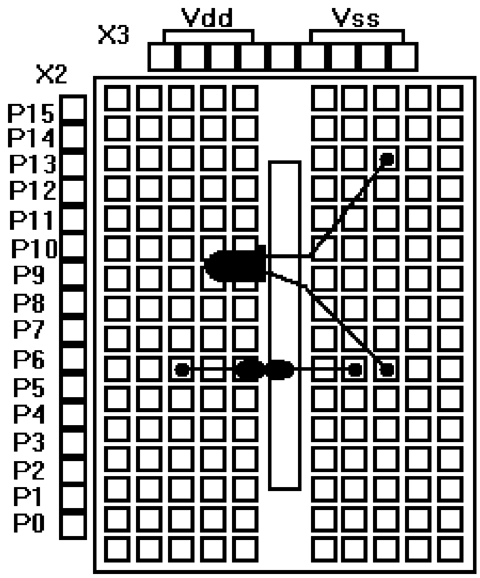
Users made errors when connecting the components to the breadboard. First off, users made errors when trying to insert the components into the specified breadboard slots. The breadboard consists of two 5 X 17 slot matrices. Each slot is horizontally connected to create a series of 5 connected, horizontal slots. To complete the experiment, users must plug the components into the correct slots on the breadboard. Pictorials are used to indicate slot positions for the components. The curriculum provides the following pictorial to indicate correct component-slot positions.



The pictorial clearly indicates the appropriate positions for insertion of components, however, users were unable to insert the components into the proper slot positions. Upon questioning, participants complained that the pictorial was too small and that it was hard to count the boxes to determine slot locations. Further, many participants simply made insertion errors due to the difficulties of handling the small components with limited manual dexterity. This error was classified as a slip. Users had appropriate intentions, but were unable to complete the task due to manual dexterity problems or simply misreading the pictorial.

d) LED's were placed on side as shown in pictorial (mistake)

Next, over half of the participants made errors when connecting the LED's to the breadboard. Many users placed the LED flat on the breadboard surface as depicted in the supporting pictorial.



Due to the difficulties of representing the LED from a 'birds-eye-view', the author represented the LED from a side profile. This representation gave users an inappropriate understanding of how to connect the LED to the breadboard. This error was classified as a mistake because the error resulted from a misunderstanding of the representation rather than a slip.

e) Participants were physically unable to connect components to breadboard (slip)

Next, many participants were physically unable to connect the components to the breadboard. Given their small size and rigidity, it was difficult for many of the participants to connect the components to the breadboard. Many participants partially inserted, damaged, or incorrectly inserted components. The breadboard slots are quite narrow and require a relatively high degree of manual strength and co-ordination to insert components. Thus, errors resulting from an inability to insert components into the breadboard were deemed slips because participants fully understood their goal, however they were unable to complete the task due to a lack of manual dexterity. Experimenter intervention was utilised to aid in circuit construction where necessary.

3.3.2 Participants were unable to correctly construct the program

The next set of findings are related to errors made during program construction. Program construction involved referring to the text to enter pre-written code segments. The program is responsible for directing the flow of current through the circuit that was constructed during the first phase of the experiment. If correctly entered, the code will instruct the microcontroller to output current through channel 0 to illuminate the connected LED every 1000 milliseconds.

Users were directed by the curriculum to copy the following code segment to the code window of the Stamp Editor software.

```
output 0
reblink:
  out0 = 0
pause 1000
  out0 = 1
  pause 1000
goto reblink
```

The participants all made similar errors. The comments next to each line of code summarise the frequently occurring, combined errors demonstrated by users.

```
Output 0
reblink       (users were unable to locate colon key or entered the “pipe” key)
out0=0        (users were unable to determine the presence of a space or relevance of a space)
pause1000     (users were unable to determine the presence of a space or relevance of a space)
out0=1        (users  were unable to determine the presence of a space or relevance of a space)
pause1000     (users were unable to determine the presence of a space or relevance of a space)
gotoreblink   (users were unable to determine the presence of a space or relevance of a space)
```

It seems that most of the code entry errors can be classified as slips. For example, in the case of the ‘reblink’ command, almost every user forgot to include the colon at the end of the command. This simple error creates a bug rendering the code unusable. However, more serious errors occurred throughout. Many users did not include spaces between commands. The omission of spaces is believed to be the result of two factors. The first factor is that the font used to present the code segments made it extremely difficult to detect the presence or absence of spaces. Secondly, the fact that much of the terminology of the command language is new to users prevents them from being aware of individual words. For example, in the last line of code, participants often omitted the space between ‘goto’ and ‘reblink.’ It is believed that this error is due to the influence of the poor font selection as well as the fact that users are unfamiliar with the terminology of the command language. Further, users omitted indentations intended to functionally group code segments. The omission of indentations intended to functionally group the code did not result in unusable code, however, it later contributed to poor code comprehension as users were unable to determine and alter the functionality of the code. Though, it must be noted that an error was discovered in the intended functional grouping of the code. The author intended to functionally group the output segments of the code like this:

```
output 0
reblink:
   out0=0
pause 1000
   out0=1
pause 1000
goto reblink
```

However, he failed to continue with his effort and produced the following ambiguous functional grouping:

```
output 0
reblink:
   out0=0
pause 1000
   out0=1
   pause 1000
goto reblink
```

Functional grouping refers to strategically placing items to render insight into their function. For example, a programmer may group code segments with similar or related functions strategically to increase the readability or his or her code. The author of the code segment found in the curriculum failed to functionally group the code in a way that reveals insight into the functionality of the code.

The lack of functional grouping resulted in poor code comprehension as users were unable to locate the portions of code responsible for setting pin 0 to high and low. Likewise, users were unable to locate the portion of code responsible for setting the duration of the blink sequence, pause 1000.

On an individual basis, program construction was fraught with typical slips. As in any computing task requiring perfect transfer of text from one location to another (ie curriculum to computer), such errors should be expected. During the experiment, users were provided with immediate feedback and help when such errors were committed.

3.3.3 Participants were unable to correctly alter the behaviour of the system

Participants were unable to successfully alter the system. Upon completion of the experiment detailed in the curriculum, participants were asked to alter the system to change the behaviour of the LED's. Participants were asked to change the blink pattern of the LED to create a blink sequence in which the LED begins 'off' for ½ second and on for ½ second. Successfully altering the system requires only a small change to the code segment. The following code excerpts demonstrate the necessary changes.

Current code	Correctly altered code
output 0	output 0
reblink:	reblink:
out0=0	out0=**1**
pause 1000	pause **500**
out0=1	out0=**0**
pause 1000	pause **500**
goto reblink	goto reblink

Correctly altering the code required participants to demonstrate insight into the functionality of the program. In particular, alteration required that the participants understand the function of the 'out' command as well as the 'pause' command. Without experimenter intervention, users were unable to successfully make the correct alterations to the code segments. Before help was provided, users were directed to use the curriculum for reference to gain insight into program functionality. Reference to the curriculum did not facilitate completion of the alteration and thus did not provide insight into the functionality of the program. Participants were encouraged to alter the code experimentally to achieve their goal. When provided with clues and the opportunity to freely alter the code, participants soon achieved success.

It seems that the curriculum itself did not provide enough insight into the functionality of the system to support alteration of the defined experiment. Thus, the knowledge gained from successful completion of the experiment did not necessarily transfer to novel alteration of the defined experiment. More insight and the opportunity to freely, experimentally alter the code led to successful completion of the alteration task.

3.3.4 Experimenter and chapter end questions

Children were unable to answer questions related to the functionality of Microcontrollers, Programs, Circuits, and Components. Upon completion of the experiment, users were asked the following questions. The number of correct responses (7 possible correct responses) is indicated to the right of each question.

F Can you tell me how a Microcontroller works? 0

NF	Can you tell me what a Microcontroller is? 3

F	Can you tell me how a Program works? 2

NF	Can you tell me what a Program is? 4

F	Can you tell me how this Circuit works? 0

NF	Can you tell me what a Circuit is? 0

F	Can you tell me what a Resistor is? (Its function defines what it is) 0

F	Can you tell me how an LED works? 0

NF	Can you tell me what an LED is? 6

F	Can you tell me how the Board of Education works? 0

Total functional questions: 6
Total non-functional questions: 4

Total correct responses to functional questions: 2
Total correct responses to non-functional questions: 13

Upon completion of the experiment, users were unable to accurately answer questions related to the functionality of Microcontrollers, Programs, Circuits, and Components. Only two correct responses were recorded for questions related to functionality. Not only that, but the two correct responses were related to Program functionality. The curriculum did not provide users with enough insight to answer simple questions related to the functionality of the aspects of the system. In contrast, users were able to accurately provide 13 correct responses to questions not directly related to an aspect of functionality.

Next, users failed to provide correct responses to the Chapter End questions featured on page 18, “What’s a Microcontroller?” Only the first three questions, were posed to participants. The first three questions were selected because they questioned the user’s understanding of the functionality of the system.

How does a Microcontroller differ from a computer?

What is the difference between hardware and software?

Why is a Microcontroller like your brain?

All users were able to provide a correct response to the first question, “How does a Microcontroller differ from a computer?” However, nearly all of the responses mentioned only visible, physical differences between microcontrollers and computers. For example, 6 out of 7 users replied, “[microcontrollers] are smaller than computers.” Other popular answers mentioned the absence of a mouse and or monitor being a major difference. None of the participants mentioned functional differences between microcontrollers and computers. The functional differences between microcontrollers and computers are actually quite complex for younger users. To answer this question, users must have a mental model of a computer as well as a microcontroller. Thus, it was expected that the participants would be unable to mention the functional differences between microcontrollers and computers.

4 of the 7 users were able to provide a correct response to the second question, “What is the difference between hardware and software?” Users commonly mentioned the fact that software is invisible (cannot be seen) and that hardware is visible (can be seen). Such an answer was considered acceptable because, although fairly obvious, software cannot be seen and hardware can. Again, responses were centred around the visible, physical differences between hardware and software. It is my opinion that the participants understood the deeper, functional difference between hardware and software, however, they were unable to articulate their understanding given the limitations of an eleven year old vocabulary and knowledge base. This opinion is based on the fact that many of the participants attempted to discuss functional differences but soon found themselves at a loss for words to describe their understanding of how they functionally differ.

None of the users were able to provide a correct response to the third question, “Why is a microcontroller like your brain?” Given the fact the users were hardly able to describe what a microcontroller is and how it functions, users were unable to comprehend the metaphorical relationship of the human brain and a microcontroller. It’s possible that the text did not sufficiently illustrate the microcontroller-brain metaphor presented on page 6 of the curriculum. On page 6, the following metaphor is presented in an attempt to reveal insight into the relationship between the human brain and microcontrollers.

“When we create devices that have a microcontroller acting as a ‘brain’, in many ways we are attempting to mimic how our own bodies operate.

Your brain relies on certain information in order to make decisions. That information is gathered through various senses such as, sight, hearing, touch, etc. These senses detect what we'll call the 'real world', & send that information to your brain for 'processing'. Conversely, when your brain makes a decision, it sends signals throughout your body to do something to the 'real world'. Utilising the 'inputs' from your senses, and the 'output' from your legs, arms, hands, etc., your brain is interfaced & interacting with the real world."

To determine if the metaphor failed to establish a metaphorical relationship between microcontrollers and the brain, participants were asked to re-read the excerpt and were again asked the same question. When the question was posed a second time, immediately after reading the excerpt, 5 out of 7 of users were able to provide correct responses to the question. Thus, it was concluded that the metaphor was sufficient in providing a metaphorical relationship, however the remainder of the text did not carry the metaphorical relationship forward. Thus, the metaphorical relationship was soon forgotten as other knowledge was acquired. This question was different from the previous because correct responses required participants to demonstrate functional knowledge. That is, it directly asked users to discuss how the functionality of a microcontroller is similar to the functionality of a brain.

All in all, the data collected from the objective textual analysis and naturalistic observation suggest that there are several factors contributing to the curriculum's inability to provide users with insight into the functionality of the system. The objective textual analysis demonstrated a lack of functional text and an unequal distribution of functional text. This prevented users from having the opportunity to learn about the functionality of the system. The naturalistic observation revealed a number of errors experienced by users during performance of the lesson. These errors are an important consideration in light of the previously established fact that the ability to learn how to use a system contributes to a user's understanding of how a system functions. Thus, if the errors prevent the user from completing the experiment, then the errors will also prevent the user from developing an understanding of how the system functions. Further, the fact that users were unable to alter the system suggests that users did not acquire enough insight into the functionality of the system to make minor alterations to the existing code (script) to create a desired behaviour. Lastly, the experimenter and chapter end questions clearly established that users were unable to answer questions related to functionality. Thus, evidenced by the findings, it has been concluded that the curriculum fails to provide users with insight into the functionality of the system.

Section4: SOLUTIONS to errors captured during naturalistic observation

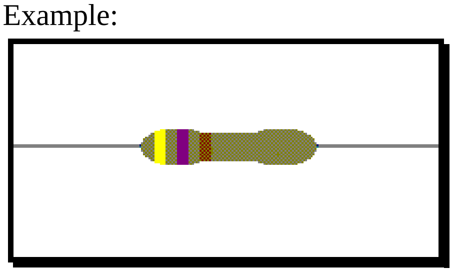
The findings presented in the previous section detailed problems that inhibited users from learning how to use the system. As Wickens pointed out, learning how to use a system is a vital part of learning how a system functions. If users are unable to complete the lesson and learn how to use the system, then they will be unable to learn about how the system functions. The following section seeks to provide solutions to the previously discussed problems that prevented users from completing the lesson and learning how to use the system. By solving these problems, users gain the ability to learn how to use the system. The solutions will allow users to learn how to use the system, thereby gaining insight into the functionality of the system.

Problem: Users made mistakes in selecting resistor types

Cause: Curriculum refers to resistors in terms of their ohm and watt characteristics resulting in confusion.

Solution: The goal of this portion of the curriculum is to help users locate the proper resistor for the experiment. Because users are unfamiliar with ohms and watts, users become confused when presented with these concepts for the selection of resistors. However, users are familiar with colour. As all resistors are designed with a colour code which represents watt and ohm characteristics, the colour code is best suited to help users locate the appropriate resistor for the experiment. Thus, the curriculum should refer to resistors using their respective colour codes, rather than solely relying upon ohm and watt characteristics. Further, use of the colour code could be used to prompt an introduction to how the colour code represents the properties of the resistor. Naturally, discussing the properties of resistors could then lead to a discussion about watts and ohms and how they effect resistance.

Example:



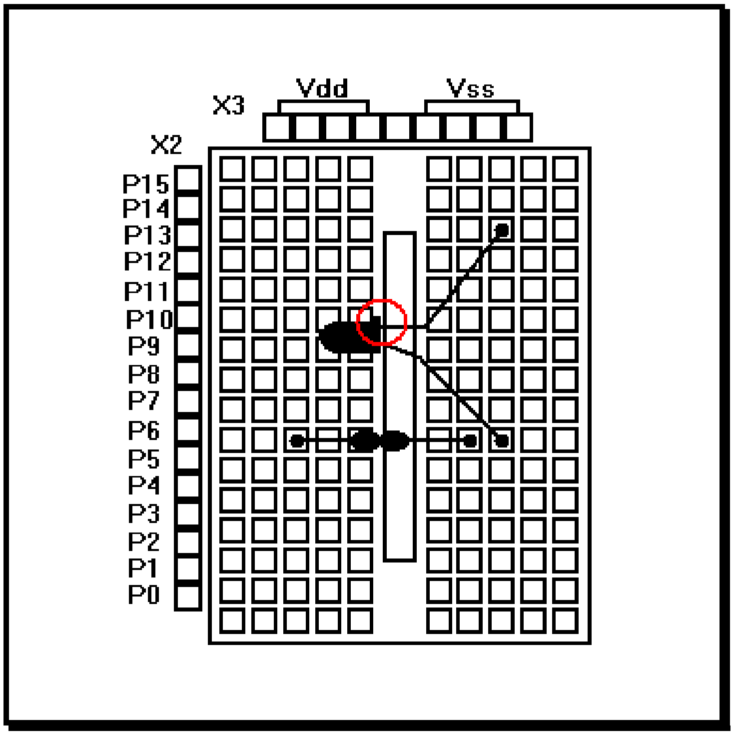
Considerations: One of the major problems with using this solution is the resulting necessity to use colour in the curriculum. At this point, only the front and back covers of the curriculum make use of colour. The body of the curriculum is represented entirely in black and white. Using colour in the curriculum would raise the production cost of the curriculum requiring an increase in cost to the customer. Thus, in order to apply this solution, the curriculum must support colour representation.

Secondly, exclusively using the colour code as the criteria for resistor selection may be counter productive to the intention of the redesign of the curriculum. The intention of the redesign suggestions is to improve the curriculum's ability to provide insight into how the system, as well as its parts, function. Strictly relying upon the colour code and not the watt and ohm characteristics of the resistor, does not improve the curriculum's ability to provide insight into how the system functions. Instead, it encourages the use of the curriculum as an operator's manual rather than an teaching curriculum.

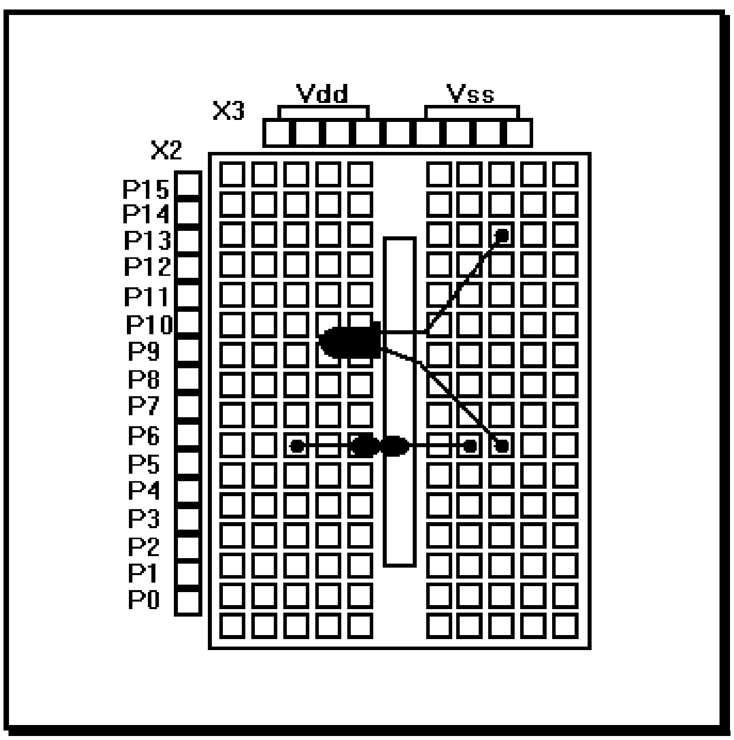
Problem: Participants failed to correctly connect LED's to breadboard. In particular, users failed to connect LED's to breadboard in proper polar position.

Cause: Users forgot to pay attention to the polarity of the LED once they began inserting it into the breadboard. The task of inserting the LED consists of numerous sub-tasks. The first task is locating the appropriate bread board slot positions to which the LED must be inserted. This involves carefully counting the row and column positions of the bread board to determine where the legs of the LED will be inserted. The second task is locating the actual LED. The third task is where the problem arises. The third task involves attending to the polarity of the LED prior to insertion. Due to the difficulty and degree of attention required to locate the correct bread board slot position to which the LED will be inserted, users forget to attend to the polarity of the LED. Thus, it seems that the primary task of locating the insertion point for the LED overrides the secondary task of attending to the polarity of the LED.

Further, users experienced difficulty in determining the correct polarity of the LED from the pictorial. That is, the pictorial did not sufficiently aid users in determining the proper polar orientation of the LED. The manufacturer of the LED implemented two forms of polarity detection. The small lip, barely visible in the pictorial and barely visible on the LED itself, indicates the polarity of the LED. Secondly, the shorter leg on the LED corresponds to the flat side of the LED. By including two forms of polarity detection, it seems that the manufacturer also recognises the difficulties of detecting the polarity of the LED. In the original pictorial, the lip is barely visible and the short leg of the LED is undetectable. Thus, the pictorial is of little use if users are unable to detect the lip highlighted below.



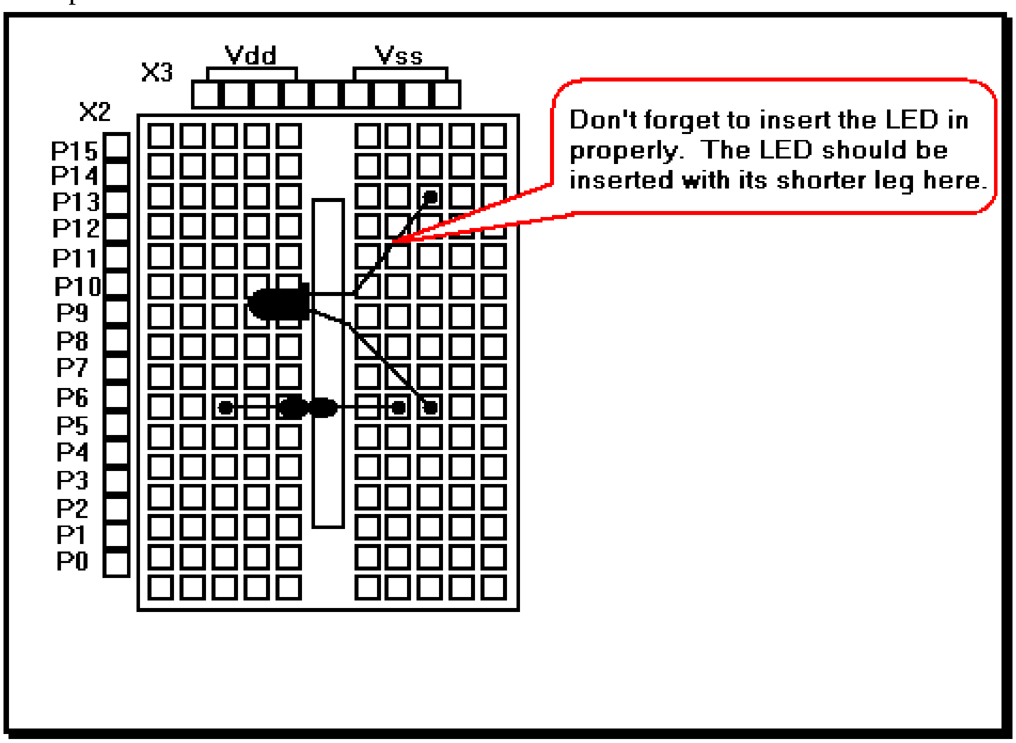
Solution: Thus, the solution shall require a reminder for users to attend to the polarity of the LED prior to insertion. When determining the polarity as well as the insertion points for the LED, users currently employ the following diagram:



The first goal of the solution is to redesign the pictorial so that users are reminded of the importance of attending to the polarity of the LED prior to insertion. Secondly, the goal of the solution is to improve the ease with which users may detect the polarity of the LED.

In providing a solution, it is important to capitalise on the most obvious hardware features that indicate the polarity of the LED. With this in mind, the lip can be disregarded as an obvious feature indicating polarity because it is nearly impossible to detect without a magnifying glass. In fact, none of the users were able to detect the lip, whereas all of the users were able to detect the shorter leg indicating the polarity of the LED. Thus, the shorter leg shall be emphasised to indicate polarity. However, the lip shall be referred to for redundancy. It's possible that the student will be working with used LED's in which the legs may have been altered in some way. The following diagram represents a suitable solution to the problem of aiding users in attending to the polarity of the LED as well as aiding users in detecting the polarity of the LED and its orientation on the breadboard.

Example:



Considerations: The solution again involves colour. Colour was used to attract the attention of the user. The solution achieves both of its aims. First, it reminds users to attend to the polarity of the LED prior to insertion. Second, it helps users to determine the proper polar position by pointing out the shorter leg of the LED.

Problem: Users were unable to insert components into the breadboard slots as show in the pictorial.

Cause: The pictorial which indicates slot positions for the components is too small. Participants complained that the breadboard was too small, making it difficult to count the slots to determine where to place the components on the breadboard.

Solution: Increase the size of the pictorial. The current pictorial is about ¼ the size of the pictorial presented in the previous solution. Thus, the redesigned pictorial should be increased to roughly the size of the pictorial featured in the previous solution.

Example: Refer to previous solution example.

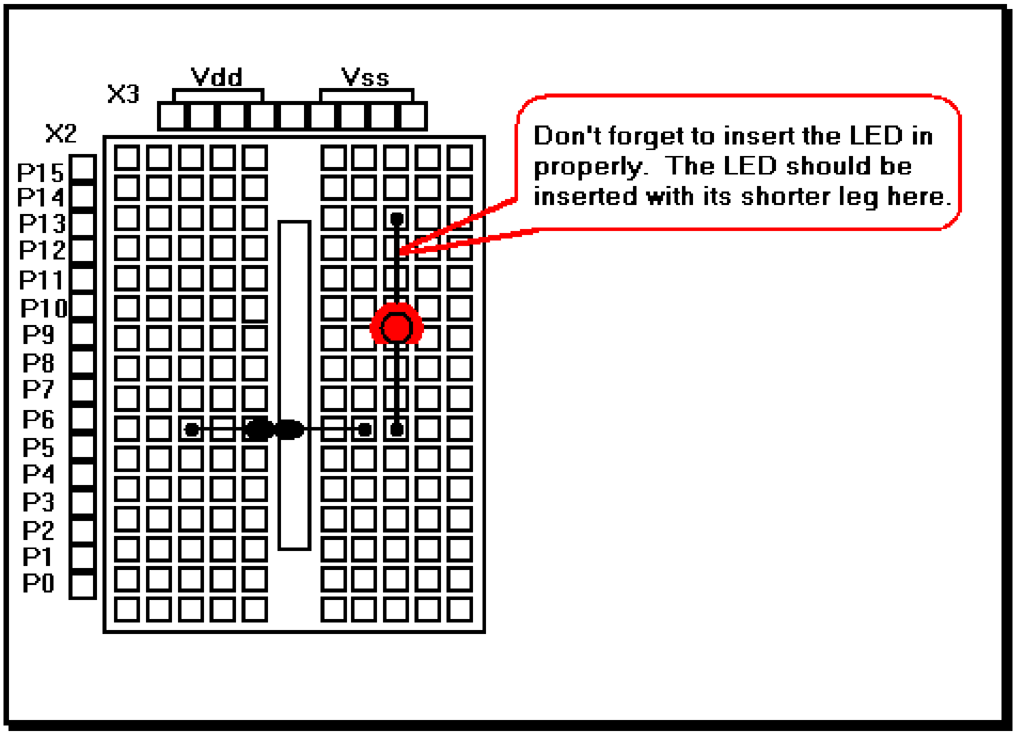
Considerations: Increasing the size of the pictorial requires more space in the curriculum. The result of this change means that the curriculum may have to increase in length to accommodate for the larger pictorial. This is a small price to pay for improved performance.

Problem: Users placed the LED flat on the breadboard surface as depicted in the supporting pictorial.

Cause: The pictorial represents the LED on its side. The author represented the LED on its side to help users recognise the component as an LED and also to emphasise the polarity of the LED. From a top perspective, the LED and its polarity are difficult to recognise.

Solution: Represent LED from a top-down perspective.

Example:



Considerations: Again, the solution requires the use of colour. Represented from a top view, without colour, the LED is extremely difficult to recognise. Thus, colour was incorporated to ensure recognition of the component as an LED. However, the clarity of the lip, indicating the polarity, is lost. Thus, the previously inserted reminder box was carried forward to provide additional support for polarity detection.

Problem: Many users were unable to physically connect components to the breadboard.

Cause: Users complained that the components were too small and fragile to be handled easily. This led to difficulties in inserting the components to the breadboard.

Solution: As previously established, the hardware may not be altered to solve problems. Thus, a more innovative solution must be generated. The following problem can be solved by moving to a computer based curriculum in which users virtually interact with the hardware. In such a solution, users would be able to construct circuits on a simulated system. A computer based system could provide an environment in which users would be able to freely interact with the hardware uninhibited by manual dexterity problems.

Considerations: Before implementing such a solution, one must consider the implications of moving to a computer based curriculum. Further, a computer based solution creates new hardware requirements for users. The current kit may be used with nearly any DOS or Windows based system. However, the solution may require an increase in hardware requirements to support simulation of the system.

Problems encountered during program construction

Problem: Users were unable to correctly copy code from the curriculum to the Stamp Editor program. In particular, users forgot to include the colon following the reblink statement, users were unable to detect the presence or absence of spaces within the code, and made frequent typos. All of these errors rendered the code dysfunctional.

Cause: Users complained that the code was difficult to read because of the font used.

Solution: In order to solve this problem, the font properties of the code segment were altered in a way that improved readability. The solution involved utilising the existing font but altering its properties. For example, the font size was increased to 13. Character spacing between commands was expanded to 2.5 point. This change improves the ease with which users may detect the presence or absence of spaces. This in turn improves the users ability to detect different code commands, improving the likelihood of code comprehension. Commands (Output, Reblink:, Reblink, Pause, and Goto) were capitalised to aid users in word recognition. Observation revealed that not only was the problem related to user's inability to detect the presence or absence of spaces within the text, but user's also had problems detecting commands. For example, originally, user's were unable to recognise the fact that goto and reblink were two different commands. Instead, users thought that the command was gotoreblink. Capitalising goto (Goto) and reblink (Reblink) helps users to detect the ending and beginning of the two commands. Capitalisation of the aforementioned words coupled with alteration of the font properties of the text, will enable users to read, comprehend, and rewrite the code segment to the Stamp Editor.

Example:
Current Code and Font

```
output 0
reblink:
   out0=0
pause 1000
   out0=1
   pause 1000
goto reblink
```

Redesigned Code and Font

```
Output  0
Reblink:
   out0  =  0
     Pause  1000
   out0  =  0
     Pause  1000
Goto  Reblink
```

Considerations: The changes made would require minor alteration to the current curriculum.

Problem: The author failed to functionally group code segments.

Cause: It appears as though the author began to functionally group the code as indicated by the indentations present within the main body of the code. However, the functional grouping is inconsistent and incomplete.

Solution: An effective solution requires functionally grouping the code with consistency and careful examination of the function of the code. Thus, the code shall be functionally grouped to emphasise the difference in function between the Out and Pause commands.

Example:

```
Output  0

Reblink:

   out0  =  0
     Pause  1000
   out0  =  0
     Pause  1000

Goto  Reblink
```

Considerations: The out commands have been consistently indented to indicate that they share similar functions. Likewise, the Pause commands have been consistently indented to indicate that they share similar functions. Further, the out and Pause commands have been indented differently to suggest that they have different functions. Line spaces have been inserted between the Reblink: and out commands to reduce the clutter and suggest that reblink is not exactly part of the functionality of the program. Likewise, a line space has been inserted between the Output and Reblink: commands to indicate that the Output command is not part of the behaviour of the Reblink procedure.

Section 4.1 Review of observed errors

Frequent errors prevented users from learning how to use the system. As Wickens pointed out, an important part of learning about how a system functions, is learning how to use the system. If users are prevented from learning how to use the system due to frequent errors, then they will be unable to learn about how the system functions. Thus, the solutions are aimed at reducing the amount of errors committed by users. It is hoped that by reducing the amount of errors experienced by users, users will be able to more efficiently learn how to use the system, thereby gain insight into the functionality of the system.

Section 5: SOLUTIONS to factors inhibiting insight into functionality

Having provided solutions to the errors captured during naturalistic observation, it is now possible to consider the other factors preventing users from gaining insight into the functionality of the system. Evaluation of the curriculum, the system, and the existing research suggest that there are two other factors preventing users from understanding how the system functions. First off, the curriculum lacks text pertaining to functionality. This was revealed by the objective textual analysis. The textual analysis uncovered the fact that much of the text is non-functional and does not directly discuss how the system functions. As a result, users were unable to gain insight into the functionality of the system. Second, the functionality of the hardware is invisible. The system relies entirely on the behaviour of electricity as it flows through the system. Electricity is invisible. Thus, the functionality of the system is invisible. Also, much of the hardware components are extremely small, almost invisible to most users. The final problem with the visibility of the system is related to the design of the components. That is, the physical appearance of the hardware does not afford insight into its function. The following section discusses and provides solutions for the aforementioned factors preventing users from gaining insight into the functionality of the system.

5.1 Causes of curriculum's inability to reveal insight into the functionality of the system.

5.1.2 Functionality of the hardware is invisible

Ordinarily, the functionality of modern hardware is kept at a distance from users. Most technology is designed in such a way that the system operates invisibly within its environment. This is because users are not normally required to understand the functionality of a system in order to use it. Consider a Video Cassette Recorder (VCR). Users do not seek to understand how VCR's function. Instead, users merely seek to understand how to use them. VCR designers design their products with this in mind. They hide the electronics, the motors, the microcontroller(s) within a plain black box. They essentially make the functionality of the VCR invisible to the user. The end result is a complex technology that is easy to use with invisible functionality. Recent attempts to make the functionality of technologies invisible has created a new area of research, ubiquitous computing.

The father of ubiquitous computing, Mark Weiser, stated that the goal of ubiquitous computing is "to make a computer so embedded, so fitting, so natural, that we use it without even thinking about it. Achieving ubiquity is usually a matter of hardware design. That is, designing hardware that hides its functionality.

Unfortunately, the Stamps in Class kit is extremely ubiquitous. That is, its functionality is invisible to users. The hardware is miniature, does not afford its function, and its behaviour is invisible.

At first glance, one might suggest that because the hardware creates the ubiquity, the solution may be found in altering the hardware. However, one of the requirements of the redesign is that the hardware may not be altered to solve problems. The reasons for this requirement are quite clear considering the enormous costs associated with changing the physical properties of the hardware.

Because the hardware is ubiquitous and cannot be altered, it puts more responsibility on the curriculum to reveal insight into how the system works. In some situations, hardware may facilitate the process of revealing insight into how a system functions. For example, consider a curriculum teaching a user how to fix a car. Because the hardware in a car is considerably larger, and has moving, mechanical parts, the hardware may actually help reveal insight into how the car and its part function. In this case, the responsibility of providing insight into how the system works has been divided between the hardware and the curriculum. In this situation, a user may read from a curriculum about how a fan belt works, then actually witness a moving, functioning fan belt. The complimentary interaction of the curriculum and the visibility of the hardware can be used to support one another. However, in the case of the Basic Stamp, the situation is quite different. Despite the fact that the Basic Stamp and its hardware are exposed and visible, users are unable to actually witness the parts in action. Yes, users may see an LED illuminate in response to carefully entered code. However, they are unable to see how the current flows through the Basic Stamp, Board of Education, components, and circuit to cause such behaviours.

The cause of the ubiquity of the hardware is three-fold. The first problem is related to the nature of the equipment. The hardware operates on electricity. Electricity is invisible when flowing through the hardware. Thus, the hardware becomes ubiquitous. The second problem is related to the size of the hardware. Although the hardware is exposed and visible, it is still too small for younger users. The Basic Stamp itself is composed of many different parts. The parts of the Basic Stamp are all visible to the naked eye, yet they are extremely difficult to detect at their current size. Thirdly, the design of the components does not afford their functionality. The Basic Stamp kit makes use of components designed not for teaching, but for application. That is, the components are designed in a way that supports integration into circuits. As a result, they are often similar in appearance, and very small. For example, resistors of varying resistance share the same appearance aside from subtle differences in their colour code. Younger users would expect that if a resistor has a higher resistance than another, that it would be larger or appear physically different. This is not the case in modern electronic component design. Thus, because the hardware cannot be designed to afford its function, the curriculum must reveal insight into the functionality of the hardware.

Thus, in order to overcome the ubiquity of the hardware, the curriculum must find a way to increase the size of the hardware, reveal how the current flows through the hardware, and must accommodate for the lack of affordance present in the design of modern electronics.

5.1.2.1 Solutions to Three Ubiquity Problems

Size: Although the hardware is exposed and visible, it is still too small for younger users.

Solution: The curriculum must provide close-up views of the hardware and multiple viewing angles. Although this may provide improved viewing of the actual hardware, merely improving a user's ability to view the hardware will not necessarily reveal insight into the functionality of the system. Improving the user's ability to view the hardware coupled with the following solutions to the other problems will contribute to improving the curriculum's ability to provide insight into the functionality of the system.

Invisibility of Current: As current is invisible, the curriculum must provide insight into the behaviour of the current as it flows through the hardware.

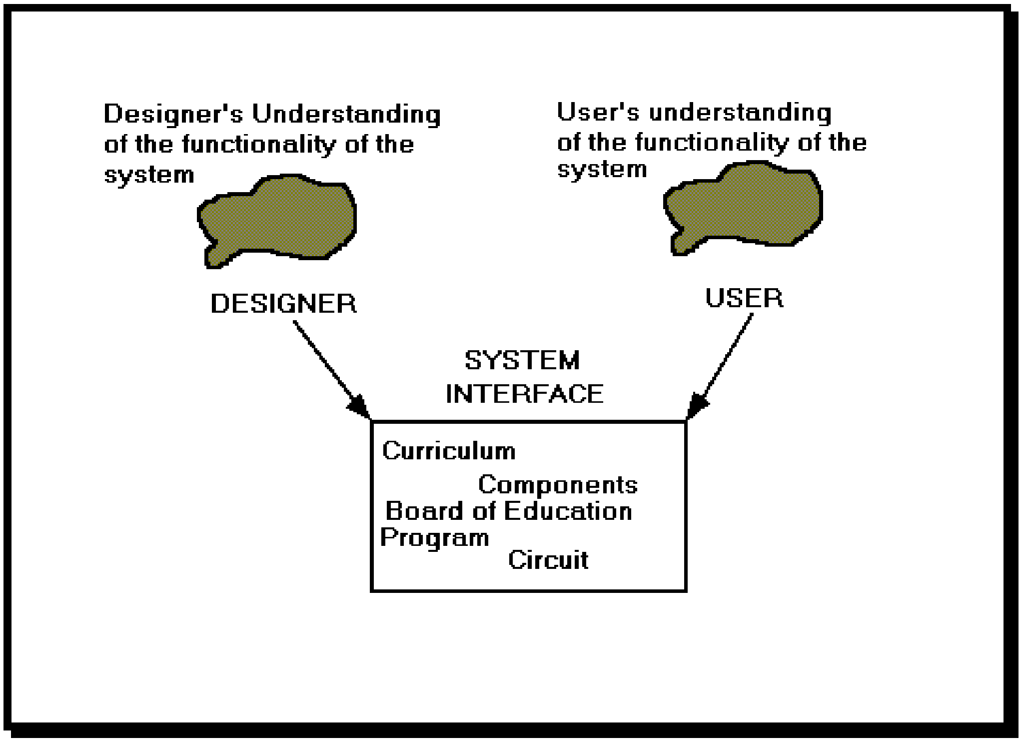
Solution: The curriculum must provide visual, dynamic representation of the behaviour of the current flowing through the hardware. The behaviour of the current can be extremely difficult for users to imagine. Thus, by providing users with a window into the system, this problem could be alleviated. However, this solution might be quite difficult to implement within the limitations of a black and white, paper based curriculum.

Lack of Affordance: There is a lack of affordance in the design of the hardware. The curriculum should provide insight into the functionality of the hardware where its design does not afford its function.

Solution: The curriculum must provide insight into the functionality of the hardware. Where the design of hardware does not afford its function the curriculum must provide insight into the function of the hardware. To achieve this goal metaphorical representation will be used.

The aforementioned problems associated with the ubiquity of the system and the curriculum's failure to compensate for the ubiquity of the hardware, make it difficult for users to develop conceptual models of the system. Conceptual models is a generic term that describes the various ways in which systems are understood by users (Preece 1994). Essentially, developing a conceptual model is the same as developing an understanding of how a system functions. With this in mind, the aim for designers is to help users develop accurate conceptual models of the system.

Preece states that Conceptual models are the result of the integration of the way users conceptualise and understand the system and the way designers conceptualise and view the system. Thus, in the case of the Basic Stamp, this means that conceptual models consist of the users understanding of the functionality of the Basic Stamp and the designers understanding of the functionality of the Basic Stamp. The users understanding of the functionality of the Basic Stamp is developed from previous knowledge as well as the designers representation of the functionality of the system presented via the curriculum. It is a highly successful approach in interface design to capitalise on users' existing knowledge (Preece 1994). The curriculum acts as the medium through which the designers understanding of the functionality of the system is communicated. Likewise, it is the curriculum through which the designer's understanding of the system and the designer's understanding of the user's understanding of the system is communicated. Clearly, the designer's understanding of the user's understanding of the system must match the user's actual understanding of the system. Confusion is created when the designer's understanding of the system or the designer's understanding of the user's understanding of the system fail to match. Where the user lacks an understanding of the system, the designer's goal should be to help the user develop an understanding of the system.



The use of metaphors may be necessary to achieve this goal. Metaphors can be used to describe a concept in a more accessible and familiar form. They are employed to help users develop an understanding of complex or difficult concepts. They can be used to present a coherent image of the whole system or to deal with specific functions or parts of the system (Preece). For example, consider the invisible, complex concept of electricity. It is invisible, yet it causes activity within electronic equipment. With its invisibility and complexity, children have a difficult time developing an understanding of how it works. Thus, implementing a metaphor to represent electricity and electronic equipment is essential. Commonly, a water metaphor is adopted to teach students about electricity. Using water as a metaphor to represent electricity, other metaphors may be developed to represent the electronic components that interact with electricity. For example, with water as a metaphor for electricity, a resistor may be represented as a funnel, limiting the amount of water (current) flowing through a circuit. It is hoped that the use of such metaphors would provide a gate-way through which users may more easily come to understand the behaviour of electricity and electronics.

The curriculum already takes advantage of the teaching power of metaphors. In the following excerpt, presented on page 6 of the curriculum, the author presents the following metaphor to aid users in understanding how microcontrollers work.

"When we create devices that have a microcontroller acting as a "brain", in many ways we are attempting to mimic how our own bodies operate.

Your brain relies on certain information in order to make decisions. That information is gathered through various senses such as sight, hearing, touch, etc. These senses detect what we'll call the "real world", & send that information to your brain for "processing". Conversely, when your brain makes a decision, it sends signals throughout your body to do something to the "real world". Utilising the "inputs" from your senses, and the "outputs" from your legs, arms, hands, etc., your brain is interfaced & interacting with the real world."

This metaphor makes use of users' existing knowledge about the brain and nervous system, to reveal insight into how the Basic Stamp works. This metaphor was well constructed and provided users with a better understanding of the Basic Stamp. However, the metaphor was soon forgotten as it was presented early in the text and was not carried forward later in the experiment. The fact that is was forgotten was demonstrated by the fact that users were unable to answer the chapter end question asking users about how a microcontroller is like a brain. However, upon re-reading the metaphor, users were able to correctly demonstrate an understanding of the metaphorical relationship between microcontrollers and the brain.

Thus, to solve the aforementioned problems, the following changes must be made to the curriculum. First, the curriculum must provide close-up views of the equipment. Namely, the curriculum must provide detailed, visible images of the Basic Stamp and any other hardware deemed too small for easy viewing.

Second, to solve the problem of the invisibility of current, the behaviour of the current must be presented in the curriculum. To achieve this goal, the use of metaphors will be required. Thus, current and the objects with which it interacts shall be represented metaphorically. The metaphors shall provide users with the opportunity to make use of existing knowledge and newly acquired knowledge to obtain insight into the behaviour of the current and the hardware it acts upon. Likewise, obtaining insight into the behaviour of the current and hardware will result in users developing an understanding of how the system functions.

Thirdly, the problem of a lack of affordance in the design of hardware shall be remedied through the previous solution. The lack of affordance essentially makes the functionality of the hardware invisible, thus transforming this problem into the same problem as the second. Thus to provide insight into the functionality of the hardware, metaphorical representation must be implemented. As the hardware cannot be redesigned to render insight into its functionality, metaphorical representation of the functionality of components shall be implemented.

5.1.3 Solutions to lack of functional text

The objective textual analysis revealed that there is an insufficient amount of functional text as well as an unequal distribution of functional text and non-functional text. That is, the curriculum lacks a sufficient amount of text related to the functionality of the system. Further, there is an unequal distribution of functional text amongst the individual aspects (Microcontroller, Program, Components, and Circuit) of the system.

The impact of the lack of functional text was revealed by the chapter-end and experimenter questions, and the users' inability to alter the system. First off, users were unable to correctly respond to chapter-end questions related to the functionality of aspects of the system. Likewise, they were also unable to correctly respond to experimenter questions related to the functionality of aspects of the system. Unable to correctly answer questions, users demonstrated the fact that the curriculum failed to provide them with sufficient insight into the functionality of the system.

Next, users were unable to alter the system. Discussed in the previous section, this portion of the experiment was based on the premise that slightly altering the behaviour of the completed experiment required insight into how the system functions. In order to make the necessary alterations to the program to achieve the new behaviour, users' had to demonstrate an understanding of the functionality of the different aspects of the system. In particular, users had to understand the functionality of the program and the circuitry on the breadboard. All participants failed at this portion of the experiment. Thus, the curriculum did not provide users with sufficient insight into the functionality of the system to facilitate alteration of the completed experiment.

To solve this problem, the following steps must be taken. First off, the raw number of functional text must be increased. Secondly, the distribution of text devoted to each of the categories (Microcontroller, Program, Circuit, and Components) must be increased and equal in distribution. By with-holding functional text from users, the curriculum supports an operator based interaction rather than a learner based interaction. Likewise, improving the quantity and distribution of text devoted to each category will provide users with different talents to share an equal opportunity to benefit from the system. Currently, the system favours users interested in program functionality and nearly ignores the other aspects of the system.

5.1.4 Limitations of a paper based curriculum

In determining the feasibility of implementing the established solutions, one must consider the limitations of a black and white, paper based curriculum. Many of the solutions support the use of colour, high quality, or moving images. The solutions require a multimedia environment. The current paper based curriculum cannot provide an appropriate medium through which the solutions may be implemented. Thus, a paper based curriculum may not provide a sufficient medium through which the complexity of microcontroller functionality may be taught.

The previously discussed problems and solutions all suggest the implementation of an, interactive, multimedia, computer based curriculum. The complexity of learning about how microcontrollers, programs, electricity, electronics, and circuits function moves beyond the scope of paper based, linear curriculum.

A wealth of research exists pertaining to the efficacy of implementing multimedia systems in the teaching of complex concepts. Overall, the research suggests "that a [multimedia based curriculum] with conceptually indexed case-based materials and modelling support can help students acquire and flexibly use complex knowledge." In other words, multimedia based curricula with structured tasks, conceptual organisation and dynamic modelling of concepts can help students learn complex concepts. It is important to note that the research suggests that not only should information be readily available, but it should also be organised. This supports the conclusions from the objective textual analysis. That is, the curriculum should provide a sense of structure using the non-functional text, and an opportunity to gain insight into how the system functions through the functional text.

A computer based curriculum could provide an excellent medium through which additional functional text could be inserted as links within the body of the structured experiment. Likewise, additional non-functional text could be inserted into the text available via links. In accordance with the problem of hardware ubiquity, a computer based solution could provide an excellent medium through which users could obtain insightful, close-up views of the hardware. Likewise, dynamic, interactive, simulations of the system could be supported. Further, these interactive simulations could implement metaphorical representation to increase the likelihood that users may gain insight into the functionality of the system. When attempting to alter the system, users could immediately view the impact of changing the design. A simulation environment would provide users with a window into the system in which they could freely and safely alter the system to learn about its functionality.

Section 6: Conclusion

The project explored the ability of a curriculum to provide insight into the functionality of a microcontroller based educational system. As a case study, the project worked exclusively with the Stamps in Class Educational Kit. Human factors evaluation methods were used too assess the curriculum's ability to provide insight into the functionality of the system. The factors preventing users from gaining insight into the functionality of the system were isolated and discussed. Finally, solutions were provided in an attempt to eliminate or reduce the effect of the factors preventing users from gaining insight into the functionality of the system.

Naturalistic observation and objective textual analysis revealed that four factors prevented users from gaining insight into the functionality of the system. First, frequent errors prevented users from completing the lessons. Second, a lack of text devoted to discussing the functionality of the system. Third, the invisibility of the system prevented users from observing the system in operation. Lastly, the limitations of a paper based curriculum prevented users from understanding the complex functionality of the system.

Based on knowledge gained from the observational analysis, objective textual analysis, and existing research, solutions were provided in an attempt to eliminate or reduce the effect of the factors preventing users from gaining insight into the functionality of the system. Where this research fails is in validating its findings. Unfortunately, carrying out an entire investigation into the curriculum was nearly impossible given the time required to conduct observational and textual analyses. Provided more time, this research could have attempted to experimentally verify the recommended solutions and redesign suggestions. Further, a more diverse sample of users could have contributed more insight into how users interact with and learn from the kit. Future research may attempt to experimentally verify the recommended solutions and redesign suggestions. Such research may also seek a more diverse sample of users.

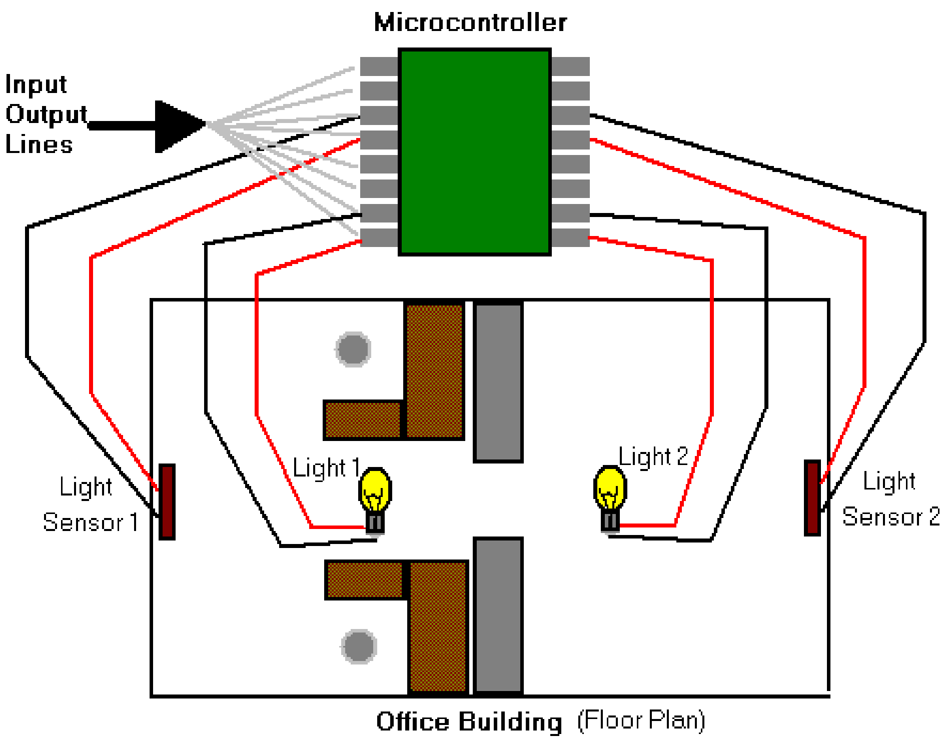
The are many implications for this research. First, off it provides insight into how users develop an understanding of how microcontrollers function. More generally, it provides insight into how users develop an understanding of how systems function. Although much of the work is problem specific, the findings can be generalised to aid in the development of other functionality based teaching curricula.

The research suggests the need for more progressive medium through which complex concepts may be taught. The research drew attention to the limitations of black and white, paper based curricula. The findings suggest the need for a more flexible medium through which dynamic, interactive simulations, and images may be implemented to aid in the teaching of functionality. It seems that new technologies demand new teaching methods. Future work may focus on developing new media through which advanced technologies may be taught.

Future work may focus on establishing metaphors for microcontroller based systems. One of the conclusions of this research was the need to make use of metaphors to aid in the teaching of complex electronic principles and electricity. Not only that, but establishing metaphors that are appropriate for younger learners. Because learning about microcontrollers requires insight into the basic principles of electricity, electronics, programming, and et cetera, researchers may focus on the feasibility of creating metaphors that consistently relate to one another. That is, the creation of metaphors that clearly integrate the concepts of microcontroller functionality, program functionality, electricity, electronics, et cetera. Further, how these metaphors may be used in curricula to facilitate the learning of complex systems.

Other future work may explore the feasibility of implementing technologically complex science kits in the classroom. Such work may identify the factors that contribute to the success or failure of such technologies in the classroom. Relatively new to the classroom, the effectiveness of implementing microcontroller based systems has not yet been the subject of rigorous, scientific scrutiny.

Appendix Item A: Regulating Lighting Levels In An Office Building With A Microcontroller Based System.



Light sensors 1 and 2 detect the lighting levels on both sides of the office. Based on the lighting environment measured by sensors 1 and 2, the microcontroller adjusts the lighting levels appropriately. Lights 1 and 2 are dimmed or brightened in response.

Appendix Item B: See What's a Microcontroller? V.1.4

Appendix Item C: Outline of experiment 1, "What's a Microncontroller?"

The experiment found on pages 7 through 17 consists of the following:

Listing of parts required for the experiment (page 7)
- Introduction to hardware used in experiment (page 7)
  - Board of Education (page 7)
  - Microcontroller (page 7)
  - LEDs (page 7)
  - Resistors (page 7)
- Figure 1.3 Pictorial (page 8)
  1) Breadboard representation
  2) Top view of components attached to breadboard

Understanding the breadboard (page 8)
- Introduction to breadboards (page 8)
  - Figure 1.4 (page 9)
    - Breadboard representation
    - Breadboard circuitry
      - Figure 1.5 Schematic (page 10)
- Electrical diagram (symbolic representation of circuit)
- Component representation

Stamp interface
- Figure 1.6 Pictorial (page 10)
- Breadboard representation
- Top view of components attached to breadboard
- Program entry and download (page 14 - 17)

First code segment (page 14)

```
output 0
reblink:
  out0=0
pause 1000
  out0=1
  pause 1000
goto reblink
```

"Dissect and look at our program" (page 14-17)
- Breakdown of commands used
- Figure 1.9 "LED off" pictorial (page 15)
  - Visual representation of code running on hardware
  - Circuit representation
- Figure 1.19 "LED on" pictorial (page 15)
  - Visual representation of code running on hardware
  - Circuit representation

References

Endsley, M. (1998). Design and evaluation for situation awareness enhancement. In Proceedings of the Human Factors Society 32nd Annual Meeting (pp. 97 - 101). Santa Monica, CA: Human Factors Society.

Gleick, J. (1992). Genius: The Life and Science of Richard Feynman. New York: Pantheon.

Lincoln, B. (1992). Constructivism, Technology, and the Future of Classroom Learning. ILTweb.com.

Martin, F. (1994). Circuits to Control: Learning Engineering by Designing LEGO Robots. Doctoral Dissertation. Cambridge, MA: MIT Laboratory.

Norman, D.A. (1981). Categorization of action slips. Psychological Review, 88, 1-15.

Preece, Jenny (1998). Human-Computer Interaction. Harlow, England: Addison Wesley.

Resnick, M. (1994). Turtles, Termites, and Traffic Jams. Cambridge, MA: MIT Press.

Simon, H.A. (1987). Decision making and problem solving. Interfaces, 17, 11-31.

Wickens, D., Christopher (1998). Human Factors Engineering. Harlow, England: Addison Wesley.