New Orleans JAZZED about Engineering Education

Paper ID #16453

Attracting Students to Programming via Physical Computing

Prof. Alka R Harriger, Purdue University, West Lafayette

Alka Harriger joined the faculty of the Computer and Information Technology Department (CIT) in 1982 and is currently a Professor of CIT. For the majority of that time, she has been actively involved in teaching software development courses. From 2008-2014, she led the NSF-ITEST funded SPIRIT (Surprising Possibilities Imagined and Realized through Information Technology) project. Since October 2013, she has been co-leading with Prof. Brad Harriger the NSF-ITEST funded TECHFIT (Teaching Engineering Concepts to Harness Future Innovators and Technologists) project. Professor Harriger's current interests include application development, outreach to K-12 to interest more students to pursue computing careers, applying IT skills to innovating fitness tools, and wearable computing.

Mrs. Mayari Illarij Serrano Anazco P.E., Purdue University, West Lafayette

MAYARI SERRANO is currently a graduate research assistant in the College of Engineering at Purdue University. She earned her B.S. degree from the Army Polytechnic School, Quito, Ecuador. She completed her M.S. in Computer and Information Technology at Purdue University. Mayari is currently a PhD student at Purdue University and is working in for the Women in Engineering Program. Her interests include foster STEM enthusiasm, and technology innovation.

Attracting Students to Programming via Physical Computing

Abstract:

College computing programs sometimes use hands-on outreach activities to introduce pre-college students to their discipline. These programs should ensure that the activities employed will engage the interest of participants and spark a desire to learn more. Programming is an essential skill of most, if not all, computing programs, yet it is also a subject that tends to make students shy away from the discipline. By selecting tools that make the value of programming obvious, and the process of learning more straightforward, the chance of retaining student interest is increased. Given that the outreach sessions are typically less than one hour, communicating the value of programming to various application areas becomes a challenge.

After employing a number of different programming tools in different 40-50 minute outreach sessions for K-12 students, experience has shown that using tools that demonstrate programming's application to physical devices enables participants to immediately grasp the benefit of this skill set. The authors share how a specific programming tool for controlling behavior of a physical microcontroller system with input sensors and output devices has been used successfully in outreach programs. The tool allows the user to create flowcharts to depict program logic, has an integrated simulator to test the behavior of the program, and can be downloaded directly to the physical microcontroller to control the physical system.

Background

The Diversity office of the Purdue Polytechnic Institute at Purdue University offers numerous outreach programs to different target groups that include girl scouts (4^{th}-12^{th} grades), minority students (7^{th}-12 grades), and female students (7^{th}-12^{th} grades). Each program may host a group between 20-60 students. These college-run programs offer a collection of 40-50 minute hands-on activities created and implemented by each participating unit in the college. The goal is to spark interest in the programs offered by the presenting unit. The outreach programs also include social activities to show that college can be fun, too.

The Computer and Information Technology department actively participates in these outreach programs. A variety of activities designed by faculty, undergraduate, and graduate students have been used in these sessions over the past ten years. Some example sessions included the use of the following tools:

1. Web page development (static HTML to dynamic ASP.NET)
2. Social media tools through a Twitter-enabled game
3. Programming languages
    a. Alice
    b. Scratch
    c. Scratch 4 Arduino
    d. C#
4. Physical computing
    a. Arduino board
    b. Phoenix Contact Nanoline microcontroller

The remainder of this paper will present the design and implementation of the physical computing session and share participant feedback.

Outreach Session Timetable

The purpose of each session is to inform students about the wide range of career opportunities in computing and provide a sample activity that sparks interest. Most tools are selected with the goal of enabling the students to perform the activity during the session and then build on it once they are home. For this reason, most tools that are selected are downloadable for free. Each student is provided a handout that includes the step-by-step instructions for the activity as well as links to the tool's download site.

The department's sessions are usually held in a computer lab that accommodates up to 25 students. Therefore, for each outreach program the department delivers two or three back-to-back sessions, depending on the number of registered students. Each session usually lasts 50 minutes; however, the time allotted to complete the activities might vary. The presenter must be able to adjust the schedule if the students arrive late or must leave early in order to reach their next session. The presenter must also be prepared to adjust in the event of unexpected technical difficulties.

Table 1 illustrates a generic schedule of activities for the physical computing outreach session. In general, every session attempts always to share career information and the end-of-session survey, so any needed time adjustments are made for the other activities. The survey is administered using the Qualtrics online tool.[6] Each student's computer is setup in advance to facilitate students accessing the survey. The survey is designed so that it can be completed in under five minutes.

Table 1: Outreach session schedule

Duration (minutes)	**Description of Activity**
10	Presenter introduction & IT careers and background
20	Guided programming activity (using Scratch or nanoNavigator)
10	Demonstration (using Arduino or Nanoline microcontroller)
5	Session review Questions & answers
5	Session survey

If a few students have technical difficulties with their assigned computer, they are paired with other students who have functioning computers. If all of the students are experiencing technical problems or time is lost due to late arrivals, they are asked to watch the presenter complete the activity and encouraged to download the tool and try the activity at home using the provided instructions.

The remainder of this paper will describe the physical computing activity that employs flowchart programming for the Phoenix Contact Nanoline microcontroller.

The Nanoline Microcontroller

Phoenix Contact is a German-based company that manufactures products and solutions for all aspects of electrical engineering and automation. Their US operations are based in Harrisburg, Pennsylvania.[3] The Nanoline is one of their programmable logic modules.[4] Although the Nanoline is a compact microcontroller, it may be used to create simple to complex physical systems for a variety of application areas.

For the activity presented in the outreach sessions, the controller is used to develop a game that incorporates exercise, also known as an exergame.[7] While the player completes a set of exercises, the controller detects the successful performance of the exercise and provides visual feedback such as lighting an LED or audio feedback such as sounding a buzzer.

The Nanoline may be programmed using a ladder diagram or a flowchart. Flowcharts are more intuitive for novices to read and understand, so the activity was designed to have students create a flowchart program for the Nanoline.

Figure 1 shows the 24-volt Nanoline base unit. This model incorporates eight digital inputs, two analog inputs, and four relay digital output channels. Additional digital and analog channels may be incorporated using extension modules, if required. Optionally, GSM (Global System for Mobile) to enable text messaging via smartphones and Ethernet modules could be incorporated for more advanced applications.

Figure 1 Nanoline 24 volts base unit

The versatile nature of this technology makes the creation and reengineering of demonstration examples easier for the presenter. The exergame demonstration products are compact and easy to assemble and transport. Additionally, the components are sufficiently robust to support student usage during the outreach activities. This robustness is important because participants sometimes get very excited and more competitive as they attempt to get the best scores.

Flowchart Programming

The flowchart is a visual representation of the program's logic. To create flowchart programs for the Nanoline, the nanoNavigator software is downloaded to each student's computer. NanoNavigator provides an integrated development environment (IDE) that includes an editor with related tools that enable the user to construct elements of a program, and a built-in simulator.[4] The simulator allows the user to observe and test the behavior of the program throughout execution by watching memory elements change in response to the environment and/or user input/actions. Once the program has been thoroughly tested with the simulator, the IDE is used to download the program to the actual controller.

Figure 2 shows the eight key symbols that may be used in a flowchart. As with traditional flowcharts, the shape of a particular block denotes its function. For example, the two diamond-shaped symbols, compare and decision, reflect branch points in the program logic from which there are two exits.

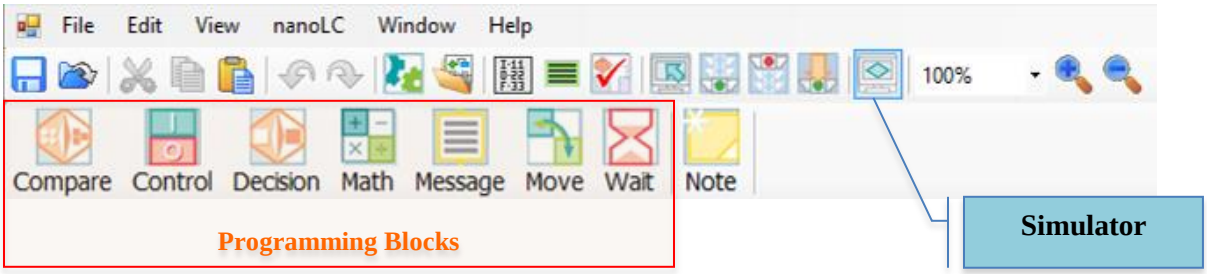

Figure 2 nanoNavigator menu.

Figure 3 shows the flowchart program that students develop in the outreach activity using the written instructions provided by the presenter. The use of a visual programming tool in the session removes the need to teach the programming of lexical commands.[1] Participants are able to easily read, understand and configure the programming blocks after a brief explanation. The reader who wishes to see how to develop a NanoNavigator flowchart should review a YouTube video tutorial on getting started with flowchart programming of the Nanoline controller.[5]

Built-in Simulator

When using a more conventional programming language, students are advised to test their programs with available tools before delivering them as a final product. Testing a NanoNavigator program is even more important because the final product is physical in nature. Depending upon the actual product, program errors may result in damage to the product, its environment, or people in its path. Fortunately, the built-in simulator does an excellent job in allowing the developer to ensure everything works as it should before deploying to the hardware.

The simulator contains various symbols that represent inputs and outputs of the product/system. It also contains a table that shows the values of different memory locations used by the program. Once the user clicks the 'start' button, the program begins execution. The user can watch values of the memory locations change in response to user input or environment changes. Additionally,

the user can watch the flow of control travel through the flowchart simultaneously via color changes. The programming block being executed changes its color to green as shown in Figure 4. The color changes allow the user to see exactly which block is active when a particular input is provided to the program or a specific condition is detected. The visual nature of the simulator allows for rapid debugging, in addition to providing instant feedback on overall correctness of the application being tested.

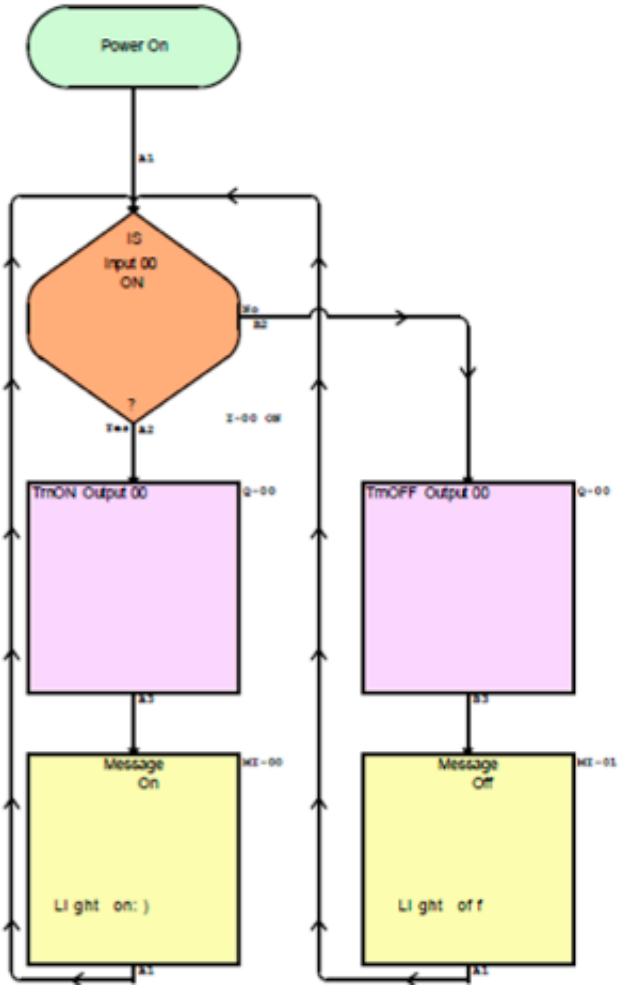

Figure 3 Flowchart program created in outreach session

Once the developer is satisfied that the program is functioning correctly based on testing with the simulator, nanoNavigator is used to download the program directly to the microcontroller. The program must be both syntactically and logically correct in order for the microcontroller to operate correctly. The input and output components must also be wired correctly, to match the program, in order for the program to run as expected on the controller. For the purposes of the outreach session, one controller that is prewired and tested for specific input/output slots is used, and during programming, students are instructed to use only those specific input/output slots.

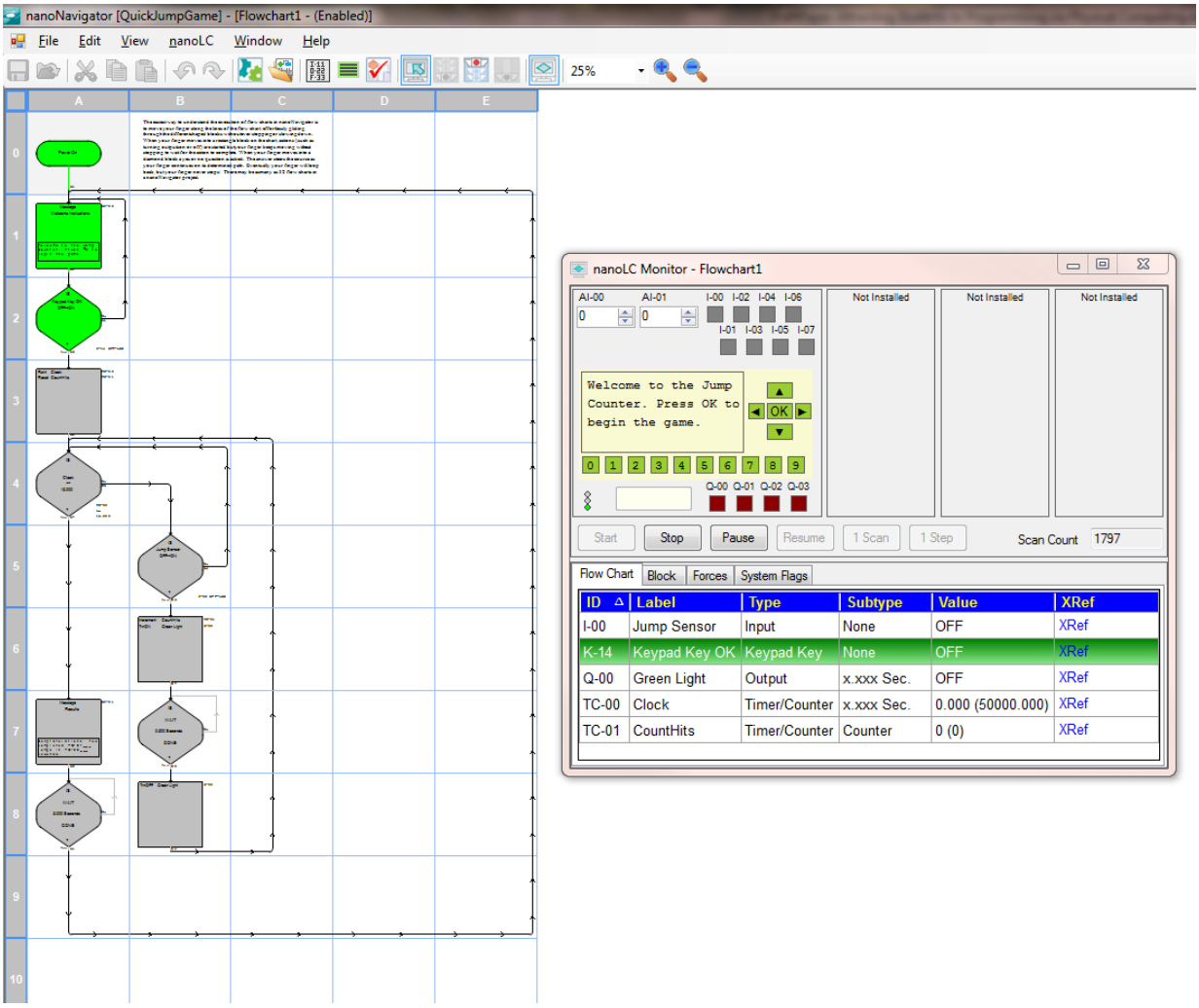

Figure 4 NanoNavigator showing flowchart execution with simulator

Conclusion

The US Bureau of Labor Statistics predicts that nearly 1.2 million jobs will be added to the IT sector; however, computing college graduates will cover only 39% of these jobs.[2] Therefore, it is increasingly more important to spark interest in precollege students to pursue computing majors. Programming is one of the most important skills needed to succeed in computer-related programs. By making the process of learning programming easy and straightforward, it should be possible to increase retention rates in computing majors. Because students tend be visual learners, developing programs that control physical devices/systems can help students understand the intricacies of programming more immediately.

The nanoNavigator software and nanoLine microcontroller accomplish this task by providing students with a tool that caters to their visual and experiential learning nature. Several sessions were delivered using this technology to develop a variety of demonstration devices that were easily manipulated and adapted by student-instructors and pre-college students during the sessions. Additionally, the implementations were both time and cost effective. Based upon the

observations from these multiple outreach sessions, the researchers believe that when students can easily touch and feel their creations, the experience will lead them to not just improved understanding but also continued interest in computing-related topics.

References

[1] Phoenix Contact, [Online]. Available: https://www.phoenixcontact.com/online/portal/us. [Accessed 1 February 2015].

[2] Qualtrics, [Online]. Available: http://www.qualtrics.com/. [Accessed 1 February 2015].

[3] S. Y. Lye and J. H. L. Koh, "Review on teaching and learning of computational thinking through programming: What is next for K-12?," vol. 41, pp. 51-61, 2014.

[4] Phoenix Contact, [Online]. Available: https://www.phoenixcontact.com/nanoline. [Accessed 1 February 2015].

[5] Phoenix Contact, "How to program nanoLine with flow chart programming - Phoenix Contact," 15 October 2009. [Online]. Available: https://www.youtube.com/watch?v=qXA2O47rqqw. [Accessed 1 February 2015].

[6] National Center for Women in Computing, "By the Numbers," National Center for Women in Computing, Boulder, CO, 2014.

[7] Y. Oh and S. Yang, "Defining exergames and exergaming," in *Proceedings of Meaningful Play*, East Lansing, MI, 2010.