THE PEDAGOGICAL ADVANTAGES OF USING THE PONG VIDEO GAME FOR PROBLEM-BASED LEARNING IN AN INTRODUCTORY ASSEMBLY-LANGUAGE PROGRAMMING COURSE

Brinkley Sprunt
Electrical Engineering
Bucknell University

Abstract

Encouraging students to learn assembly-language programming is a difficult task in an introductory course. The traditional, lecture-style teaching approach can easily fail to address the varied learning styles of students or to promote deep learning of assembly-language programming concepts. This paper describes the Pong mid-term project that has been successfully used to overcome these problems. The Pong project, modeled after the original video arcade game, is an engaging programming project that augments the lecture-style approach while also promoting the deep learning of assembly-language programming concepts and encouraging the students to become independent learners. This paper provides a detailed description of the Pong assignment including the hardware and software requirements, functional description, suggested implementation steps, and the grading structure. Additionally, the overall structure of the introductory assembly-language programming course is discussed and its placement in the electrical engineering curriculum is explained. Assessment data are also presented that demonstrate the effectiveness of the Pong assignment for improving both the depth and level of learning as well as for fostering the development of independent learners. The paper also identifies several limitations of the Pong assignment and discusses how the assignment can be extended to introduce and/or demonstrate higher-level system design concepts.

Introduction

Students, even those with prior programming experience in a high-level language, often find assembly-language programming difficult to learn. This difficulty is primarily due to the precise requirements and low-level limitations typical of assembly-language programming. These seemingly unforgiving qualities can easily frustrate students and discourage them from making the effort to climb the learning curve associated with assembly-language programming. As such, it is important to devise a set of lectures and assignments that provide the necessary information to program in assembly while also engaging the students' interest and demonstrating to the students that they can be successful assembly-language programmers.

Pong Goals

This paper describes a key assignment, "Pong", that has been used as the mid-term project for the introductory assembly-language programming course for the last four years in the Electrical Engineering Department at Bucknell University. For this assignment, the students are asked to implement a version of the Pong video game in assembly language. Pong, a video-game version of ping-pong introduced in 1972 by Atari, is widely recognized as the progenitor of the hugely successful arcade and home video-game markets [8]. Although the simple features of the original Pong game pale in comparison to the many, full-featured video games available today, it is a good choice for a mid-term project because it is an interesting, engaging application that is not too difficult to implement.

The Pong assignment addresses a number of pedagogical goals this course. These goals are:

- To promote deep learning [5] of assembly-language programming concepts via a problem-based learning approach [15].
- To raise the level of learning of assembly-language programming concepts as characterized by Bloom's taxonomy [3].
- To augment the traditional, lecture-style teaching approach in order to accommodate the different learning styles of students [6].
- To encourage students who typically follow a strategic learning approach to take more of a deep-learning approach [4] by engaging them in the solution of an attractive and interesting problem. This also encourages students who may be characterized as surface or apathetic learners to move toward the deep-learning approach.
- To encourage students to become confident, independent learners by having them successfully solve a challenging problem characterized by easily understood functional requirements but vague or unspecified implementation requirements.

The Pong assignment promotes deep learning [5] of assembly-language programming concepts via a problem-based learning approach [15]. For the assignment, students work in groups of two to create a working program based upon requirements stated in the assignment as well as executable, reference examples of an incrementally-developed Pong program. To implement Pong successfully, the students must master the majority of the assembly-language concepts discussed in lectures and practiced in earlier lab assignments. As the students develop their Pong programs, they obtain immediate feedback by executing their programs and comparing their results with the behavior of the reference programs. Although the general requirements of Pong are clearly stated, some of the more advanced Pong features are not clearly defined and many different solution approaches exist. In addition to the feedback the students receive by executing their programs, four three-hour lab sessions (one per week) are allocated for the development of Pong during which the professor provides assistance. Because it is unlikely that the development of Pong can be completed using only these lab periods, the students are encouraged to work outside of the scheduled lab periods.

Students routinely demonstrate learning at the upper levels of Bloom's taxonomy of learning levels [3] while developing Pong. Although the students begin with a skeleton Pong program (discussed later in Section 3), they develop the majority of the code themselves. Such development not only requires demonstration of learning at the lower levels of Bloom's taxonomy (specifically *Knowledge* and *Comprehension* of assembly-language programming concepts), but also at the higher levels of *Application*, *Analysis*, and *Synthesis*. As the students write code, they demonstrate learning at the *Application* level. As the students debug their code, they demonstrate learning at the *Analysis* level. As the students develop the code for some of the more sophisticated features of Pong (e.g., full-motion paddles, generating beep tones, and adding "english" to the ball when it is hit), they demonstrate learning at the *Synthesis* level. Also, depending upon the approaches taken by the students, they may evaluate different implementation options for various Pong features (e.g., using the joystick position to indicate absolute position of the paddle or to indicate the relative direction for paddle movement), demonstrating learning at *Evaluation*, the highest level.

The traditional, lecture-style teaching approach can easily be ineffective for many students because it fails to address different learning styles [6]. This is especially true for teaching assembly-language programming, a subject area that is difficult to comprehend fully in a passive learning environment that does not provide sufficient practice or feedback. As such, additional teaching styles are necessary to engage the students and accommodate a variety of learning styles. The Pong assignment is an excellent way to augment lectures. It requires active involvement and cooperation from the students while providing extensive practice with feedback on key programming concepts and

skills. From the point of view of Felder's learning styles [6], the Pong assignment provides substantial practice in assembly-language programming concepts and requires cooperation between students. This practice and cooperation accommodate students with active, intuitive, reflective, and global learning styles. As such, the Pong assignment significantly increases the learning styles addressed in the course beyond the intuitive and deductive learning styles typically covered by lectures.

The Pong assignment was chosen primarily because it is an engaging project with easily understood functionality. These qualities inspire students who might otherwise tend to suffer through a more mundane assignment. Specifically, this goal of Pong is to encourage students who would be characterized by Entwistle [4] as surface/apathetic learners to move toward a more deep, strategic approach to learning. Entwistle defines surface/apathetic learners as students who desire only to meet the minimal course requirements. In contrast, he defines deep strategic learners as students who exhibit a genuine interest in understanding course concepts and an alertness to the course's assessment methods with the goal of attaining high grades. Although Entwistle states that a deep strategic approach to learning is generally associated with successful academic performance [5], not all students adopt this approach. As such, Pong's engaging qualities motivate students to employ a deep learning approach.

To foster independent learning, the early parts of the assignment are designed to provide students with significant structure and direction, while the later parts are less specific and more unstructured. This organization helps the students attain early success, giving them confidence to begin the implementation of the more complex features. By first providing explicit directions and structure and then moving to easily understood requirements with vague or unspecified implementation directions, the students are forced to learn independently. This approach also helps prepare the students for an end-of-semester term project, a microcontroller-based system of their own specification and design.

Paper Outline

The following is an outline of the remainder of this paper. First, a description of the introductory assembly-language programming course in the Electrical Engineering Department at Bucknell University is given, along with some background on the Pong video game. The next section explains the assignment, providing descriptions of the hardware and software tools needed, game functionality, interfacing requirements, suggested implementation steps, grading structure, and documentation requirements. An assessment of the effectiveness of the assignment is then presented, followed by a discussion of the opportunities and limitations of the Pong assignment. The paper concludes with a summary of the advantages of this assignment.

Background

This background section describes the introductory assembly-language programming course, ELEC 247 [13], in the Electrical Engineering Department at Bucknell University and also gives a brief description of the Pong video game.

The Microcontroller System Design Course

ELEC 247 is a sophomore-level course offered in the spring semester at Bucknell. Prior to ELEC 247, students are required to take an introductory electrical engineering course, a circuits course, and an introductory high-level-language programming course. As such, the students have some basic experience with lab instruments, circuits, and programming concepts prior to taking ELEC 247. ELEC 247 is a pre-requisite for the digital system design course and several elective courses including VLSI, computer architecture, and advanced digital design.

The primary course outcomes of ELEC 247 are related to the abilities to program and debug in assembly-language, to interface a microcontroller with simple devices, to use polling and interrupts effectively, and to evaluate microcontroller system design alternatives. The course is roughly divided into three sections. The first section introduces the key assembly-language programming concepts that underlie all of the course outcomes. The early lectures introduce processor operation, addressing modes, instruction classes, data structures, conditional branches, the stack, and subroutines. Students gain practice with these concepts along with I/O ports, interfacing, and interrupts in the early lab assignments. The second section of the course gives the students extensive practice with assembly-language programming concepts, while also introducing the higher-level microcontroller system design concepts. The extensive programming practice is accomplished via the Pong mid-term project, which includes a four-week lab and homework assignment. During the Pong project, the lectures introduce polling and interrupts, memory types, and memory-mapped I/O ports. The third section of the course gives the students practice with the topics covered in the lectures during the second section. Specifically, the students implement memory-mapped I/O ports, an interrupt-driven clock, and a term project. The lectures during the third section focus upon system design issues related to interrupts, polling, system state, and concurrency.

Structuring the course this way provides a number of advantages. First, the students are introduced to new concepts in lectures before encountering them in lab. This allows each concept to be presented in at least two different ways (passively in lectures and actively in labs). Second, while students "get their feet wet" with a gentle introduction to assembly-language programming early in the course, the Pong project throws the students into the "pool" of assembly-language programming and forces them to learn how to "swim". This approach solidifies the students' programming skills and enables them to understand and appreciate the higher-level concepts being introduced in the lectures. The third part of the course builds upon the momentum established with the Pong project and leads into the culminating term project assignment. The assignments early in the third section of the course give the students significant experience with interfacing and interrupts in preparation for the final term project.

For the final term project, the students specify, implement, and document a microcontroller system of their own choice. This project must interface the microcontroller with other devices and use both polling and interrupts. As the Pong assignment provided significant programming experience, the term project provides the opportunity to practice the additional and higher-level concepts of interrupts, interfacing, and system design. This three-part course structure enables the term project to be a capstone in which all of these concepts come into play, pushing the students to the highest levels of learning.

To date, ELEC 247 has used the Motorola M68HC11 processor [10] and the Axiom CME-11E9-EVBU development board [1]. Although this is a simple development system, it is sufficiently featured and powerful for all of the concepts covered in the course while also being relatively inexpensive (less than US $100.00). Note that none of the assignments and concepts covered in the course and discussed this paper are unique to the M68HC11 and all are easily applicable to other microprocessors.

The Pong Video Game

Pong, a video game version of the table-tennis game of ping-pong, was the first coin-operated video game ever produced [8]. Atari introduced Pong in 1972 and its popularity grew quickly, generating hefty profits and inspiring many other video games. The features of the original Pong game were quite simple. Pong had a monochrome display that depicted a paddle for each player, a ball, a net drawn down the middle of the display, and the score for each player as

shown in Figure 1. Each player attempted to keep the ball in play by hitting the ball toward their opponent using a knob to control their paddle's vertical position. When a player missed the ball, a point was scored for the player's opponent and a new point was begun. Pong also had a speaker that beeped each time the ball bounced and emitted a low-sounding tone when a point was lost.

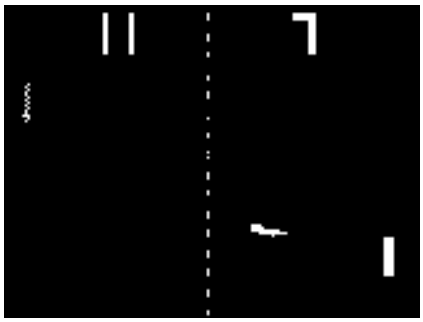

Figure 1: Atari Pong Screen Display.

The Pong Assignment

This section describes in detail the Pong mid-term project assignment. First, the Pong hardware and software setup is described. Next, a summary of the instructions and requirements for Pong along with a description of the Pong skeleton program are presented. The joysticks and buzzer, which must be interfaced with the microcontroller, are then described. Next, the grading structure along with an incentive bonus for Pong are explained. The homework assignment regarding the documentation for Pong is then described.

Pong Hardware and Software Setup

The hardware and software setup for the Pong assignment is as follows. A microcontroller development board and assembler tools are necessary. For ELEC 247, the Axiom CME-11E9-EVBU development board [1] is used along with the GNU development tools for the M68HC11 [14] running with the Cygwin [12] tools on either a Windows 2000 or Windows XP host PC. Students develop their assembly programs on the Windows PC and download them to the microcontroller board via a standard serial interface. Pong programs typically range in size between 1K to 4K bytes, so a microcontroller development board with at least 4K bytes of RAM is needed. To communicate with the microcontroller board from the Windows PC, the HyperTerminal communication program [7] (a standard Windows program) is used. HyperTerminal's VT52 terminal emulation mode is used to support the cursor addressing commands needed to implement Pong.

In addition to the above, the students need to interface two joysticks and a buzzer with the microcontroller board. For the Pong project, two dual-axis, spring-loaded, potentiometer joysticks are interfaced with the on-chip A/D converter on the M68HC11. If the microcontroller does not have an integrated A/D converter, then either an external converter or an alternative approach for controlling the paddles is needed. To support the Pong sound requirements, a simple piezoelectric buzzer is used.

The Pong Game Play Instructions

The Pong game play instructions are as follows. When the game starts, the screen is cleared and a new game message is printed. Next, the Pong court is displayed (as shown in Figure 2) and a ball is launched toward one side of the court. Players use the joysticks to position their paddles to hit the ball. When a player hits the ball, the ball is reflected toward the other side of the court and play on the current the point continues. When a player misses the ball, a point is scored for the player's opponent. When a point is scored, a message is displayed indicating who scored (left or right) and the ball is launched again to begin a new point. Once one player reaches the maximum score, a message is displayed indicating who won the game (left or right) and a new game is begun. As the game is being played, the system beeps each time the ball bounces and, when a point is lost, a low-frequency tone is sounded to indicate that the point is over.

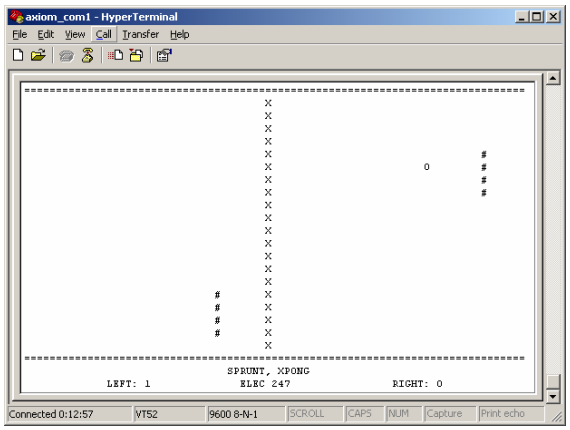

Figure 2: ELEC 247 Pong Display.

Suggested Implementation Steps for Pong

The students are given a set of suggested steps (shown below) for implementing Pong. These steps, along with the skeleton Pong program (discussed in Section 3.4), are meant to give the students a good starting point for thinking about and developing Pong. However, they are not strict requirements and different implementation approaches are acceptable and encouraged. Each step provides a brief description of the Pong functionality to be implemented. Some steps list the subroutines and data structures in the Pong skeleton program that need to be written or updated to provide the desired functionality. Additionally, the students are given a set of reference programs that implement the functionality of each of these steps. The students can download and execute these versions of Pong to get a visual example of the behavior of Pong for each of the steps. However, these reference programs can only be executed and their source code is not provided to the students.

Below is a summary of each suggested step for implementing Pong from the 2003- 2004 offering of ELEC 247:

- **PONG1**: **Setup the court**. Draw the court and score on the terminal screen and then loop forever, repeatedly drawing the paddles.
- **PONG2**: **Bounce the ball**. Continuously bounce the ball around the screen, always assuming the ball hits the paddles without detecting paddle hits or misses.
- **PONG3**: **Move the paddles vertically**. Move the paddles vertically on the extreme left and right sides of the court. Do not check for the paddles hits or misses.
- **PONG4**: **Hit or miss the ball**. On a hit, reverse the horizontal direction of the ball. On a miss, terminate the program.
- **PONG5**: **Score points**. When the ball misses a paddle, increment the appropriate score and start a new point. Do not check for a maximum score.
- **PONG6**: **Play a game**. Clear the screen and start a new game when one player scores five points.
- **PONG7**: **Print messages for new games, points scored, and games won**. Display these messages on the screen long enough for them to be read.
- **PONG8**: **Improve the game display**. Determine why the display for the PONG8 reference program is better than PONG7's display and implement the nicer display.
- **PONG9**: **Beep!** Obtain a buzzer and beep whenever the ball bounces and emit a low-sounding tone whenever a point is lost.
- **PONGFULL**: **Build a better Pong**. Enable the paddles to be moved horizontally as well as vertically and allow them to be placed anywhere on the appropriate side of the court.
- **XPONG**: **Build an eXtreme version of Pong**. Allow paddle hits to add "english" to the ball as is done in the XPONG reference program.

The descriptions for the suggested implementation steps are designed to move from being very detailed to being quite vague in order to force the students to become more independent in the design of Pong. For example,

the initial step, PONG1, describes the Pong court display for the terminal emulator, includes a picture (as shown in Figure 2), and details all of the Pong skeleton subroutines and data structures that need to be written to display the Pong court. In contrast, the specifications of the later steps are much more vague. For example, in the PONG9 step, the students are merely instructed to obtain a buzzer and have it beep when the ball bounces and emit a lower-sounding tone when a point is lost. No directions are given as to how the buzzer works, how to interface the buzzer with the microcontroller, or what subroutines or data structures need to be modified or created to make the buzzer work. Another example of vague specifications occurs with the PONG8 step, where the students are instructed to improve the display, but are not told what is wrong with the display or how to improve it. By comparing the behavior of the PONG7 and PONG8 reference programs, the students should determine that the PONG8 display is better because the paddles are only redrawn when their position should be changed. The baud rate for the serial link between the microcontroller and the terminal emulator is fairly low (9600 baud) and, at this rate, the display flickers and the game play is slowed when the paddles are redrawn constantly. Once the students determine why the PONG8 display is better, they must decide how to implement the improved display functionality.

The most difficult step in implementing Pong provides another example of how clear functionality specifications with vague implementation direction create a situation where students have to think independently. This step is PONGFULL and requires the students to support full-motion paddles. This step often surprises students because they assume that a simple modification of the code they use for vertical paddle motion will suffice for full-motion paddle support. After implementing this straightforward change to their code, they soon discover that, while their paddles can reach the top, bottom, and sides of the court, the paddles cannot be placed in the corners of the court. Eventually, they examine the joysticks and discover that the motion of the joystick is restricted to a circular hole on top of the joystick. The students must then recalibrate the acceptable ranges for the A/D converter readings from the joysticks and accommodate some "dead zones" where moving the joystick does not move the paddle. Alternatively, some students change their interpretations of the joystick positions at this point from an indicator of the absolute paddle position to an indicator of which direction the paddle should be moved toward. Using the joysticks to specify absolute paddle positions or to specify relative paddle motion are both acceptable solutions.

This well-structured approach for the Pong assignment yields a number of pedagogical advantages. First, Pong allows students to make visible progress early and fairly easily, giving them the confidence to pursue the more complex implementation steps. Second, the clear implementation directions of the early steps appeal to students who prefer a "serialist" learning approach (characterized by tight structure and a step-by-step approach) while the specific requirements but vague implementation directions of the later steps appeal to students who prefer a "holist" learning approach (characterized by impulsive work inspired by interest and examples) [4]. This movement from the "serialist" to "holist" learning approach also effectively encompasses both poles of Felder's sequential/global learning style dimension [6]. Third, the hands-on programming activity as well as the introspection necessary to resolve and debug the various implementation problems encountered cover both poles of Felder's active/reflective learning style [6]. Fourth, the experience and confidence the students gain prepare them to specify and implement the end-of-semester term project. By addressing a wide variety of learning styles and inspiring confidence in the students, the Pong assignment strives to provide an optimal learning environment for the students.

```
          ;; The GLOOP below manages the playing of a whole game.
        ;; Each time the loop is entered, a new game begins.
        ;;
GLOOP:  jsr     NEWGAME         ; setup for a new game

        ;; The PLOOP below manages the playing of a single point.
        ;; The PLOOP continues until either the left or right
        ;; player misses the ball.
        ;;
PLOOP:  jsr     MOVPADS         ; move the paddle locations
        jsr     PNTPADS         ; paint the paddles
        jsr     MOVBALL         ; move the ball location
        jsr     PNTBALL         ; paint the ball
        jsr     DELAY           ; delay for a little bit
        jsr     ERSBALL         ; erase the ball
        tst     PNTOVRF         ; is this point over?
        beq     PLOOP           ; keep playing this point

        jsr     UPSCORE         ; update the score
        jsr     NEWPNT          ; setup to play a new point
        jsr     GAMCHEK         ; see if the game is over
        tst     GAMOVRF         ; is this game over?
        beq     PLOOP           ; if game not over, play new point
        bra     GLOOP           ; begin a new game
```

Figure 3: The Pong Main Loop.

The Skeleton Pong Program

As a starting point for the development of Pong, the students are given a "skeleton" version of the full Pong program. The Pong skeleton program consists of the main Pong loop, empty subroutines and data structures with comments, a delay subroutine, and a set of constant definitions for drawing the court, paddles, and ball. Each of these main components in the skeleton program is briefly described below.

The Main Loop

The key part of the Pong skeleton program is the main loop shown in Figure 3. The loop is composed of an outer game-playing loop, GLOOP, and a nested, point-playing loop, PLOOP. These loops consist primarily of subroutine calls for the basic game functions: preparing for a new game or a new point, moving and drawing the paddles and the ball, and updating the score. Near the end of each loop, simple flag checks are made to see if the play for a point or a game is over. All of the subroutines defined in the Pong skeleton program are empty, except for the DELAY subroutine, which uses a nested loop to provide a delay of roughly a tenth of a second. As such, when the skeleton program is executed, it spins infinitely in the main loop, calling empty subroutines and the DELAY subroutine while displaying nothing on the terminal screen.

The Empty Subroutines

A brief set of comments and an entry point are defined in the skeleton program for each of the subroutines called from the main loop. The brief comment describes the high-level function of the subroutine. Examples for the NEWGAME and ERSBALL subroutines are shown in Figure 4. The students are free to follow the subroutine comments or implement their own approach. However, the students are expected to update the comments to match the code they write.

The Program Constants

A set of constants is defined in the Pong skeleton program. These constants define values that can be used to draw the court, paddles, and ball. For example, some of these constants are shown in Figure 5. The students are free to use or ignore these constants when creating their Pong programs.

```
;;; The NEWGAME subroutine sets up to play a new game.  First, the "new
;;; game" message is displayed on the screen.  Next, the scores are
;;; zeroed and the game over flag is cleared.  The NEWPNT subroutine
;;; is then called to setup for a new point.  Finally, the court
;;; and scores are painted on the screen.
;;;
NEWGAME:
        rts

;;; The ERSBALL (ERaSe BALL) subroutine below is used to erase the
;;; ball from the screen.  If the ball is on top of the "net" drawn
;;; down the middle of the court, the "net" is redrawn in the ball's
;;; place.  Otherwise, a SPACE character is used to erase the ball.
;;;
ERSBALL:
        rts
```

Figure 4: Example Subroutine Definitions in the Pong Skeleton Program.

```
.EQU    OFFSET,         31              ; VT52 cursor addressing offset

.EQU    CRTROWS,        20              ; number of rows in court
.EQU    CRTCOLS,        79              ; number of columns in court

.EQU    NETCOL,         OFFSET+39       ; the column for the "net"

.EQU    MINBROW,        OFFSET+2        ; minimum ball row value
.EQU    MAXBROW,        OFFSET+1+CRTROWS; maximum ball row value

.EQU    MINBCOL,        OFFSET+2        ; minimum ball column value
.EQU    MAXBCOL,        OFFSET+CRTCOLS-1; maximum ball column value

.EQU    MINLPCOL,       OFFSET+1        ; minimum left paddle column
.EQU    MAXLPCOL,       MINLPCOL+31     ; maximum left paddle column
.EQU    MAXRPCOL,       MAXBCOL+1       ; maximum right paddle column
.EQU    MINRPCOL,       MAXRPCOL-31     ; minimum right paddle column

.EQU    BALLCHR,        AS_O            ; the character for the ball, "O"
.EQU    PADCHR,         AS_PND          ; the character for the paddles, "#"
.EQU    NETCHR,         AS_X            ; the character for the "net", "X"
```

Figure 5: Pong Constant Definitions.

The Data Structures

The data structures defined in the Pong skeleton program consist of a few variables and empty strings. For example, the variables consist of definitions for the row and column increments used to change the ball position, the left and right player scores, and flags to indicate whether or not a point or a game is over. A number of empty string variables are defined for the various commands that will be sent to the terminal screen during game play, such as the commands to draw the court and erase or paint the ball and paddles along with the text messages that indicate which player scored and which player won the game. As with the other structures in the Pong skeleton program, the students are free to use or ignore them.

The ADC Program

The ADC program, a program that configures the on-chip A/D converter and displays its conversion results on the terminal, may be given to the students at the start of the Pong project if the A/D converter has not been introduced in prior lab assignments. The ADC program allows

the students to concentrate fully on developing Pong code rather than wrestling with the A/D converter. The ADC program also enables the students to determine quickly if their joysticks are operating as desired and to recalibrate their interpretation of the A/D converter results when supporting full-motion paddles as described above.

The Pong Grading Structure and Incentive Bonus

The grading structure for the Pong assignment (shown in Figure 6) encourages students to complete Pong successfully while also encouraging students to take more of a deep-learning approach. As Figure 6 shows, an average grade can be obtained by implementing the basic features of the Pong game (i.e., implementing the functionality specified in steps PONG1 through PONG7). Students can attain a high grade only by completing the more challenging features of Pong (PONG8 through XPONG). Additionally, the successful completion of XPONG provides a hefty bonus. The students who complete all of the specified Pong functionality, including XPONG, receive an additional five points added to their final grade for the entire course. Typically, a test in this course accounts for 15% of the final grade, making the Pong bonus equivalent to a 33-point bonus on a 100-point test. The Pong bonus also encourages students who may be taking a strategic approach to take the more desired deep-learning approach in order to complete the more difficult components of the Pong assignment.

The Pong Documentation Homework Assignment

In addition to the programming component of the Pong project, a documentation homework assignment is also given to the students after they have demonstrated their completed program. The students are given the following suggestions for documenting their program:

- Accurately comment the code. The comments must agree with the code.
- Use the comment field for every instruction in the program.
- The instruction comments should explain what the program is doing, not what the instruction is doing. For example, the comment "add 1 to X" for the INX instruction only explains what the instruction is doing. "Point X at the next

Pong Functionality	Grade	Bonus
PONG1 - PONG7: Full game play.	75%	none
PONG8: Improved game display.	80%	none
PONG9: Beep!	85%	none
PONGFULL: Full-motion paddles.	95%	none
XPONG	100%	+5 final course grade

Partial credit will be given as appropriate, based upon the functionality actually implemented.

Functionality demonstrated after the due date will be assessed a fixed late penalty. Functionality demonstrated two weeks after the due date will receive no additional credit.

Figure 6: The Pong Grading Structure.

element in the array" would be a better comment because it explains what the program is doing with the INX instruction.

- Use comments at the beginning of the main program and each subroutine.

- Use comments to describe the data structures.

The goals for this assignment are to encourage the students to follow a good commenting style for their code and to note the advantages of documenting the code as it is written, and not as an afterthought. This documentation assignment also provides the professor the opportunity to give the students feedback regarding their code documentation style prior to the term project, which must also be well documented.

Assessing The Effectiveness of The Pong Assignment

This section presents a summary of the assessments of the effectiveness of the Pong assignment with respect to achieving the objectives discussed in Section 1. First, the difficulties associated with assessing these pedagogical objectives for this course are discussed. Next, an assessment of the improvements in the depth and level of learning associated with the Pong assignment is presented. This section closes with an assessment of the role the Pong assignment played in encouraging students to become confident, independent learners.

Assessment Difficulties

Assessing the effectiveness of the Pong assignment upon the learning objectives is difficult for a number of reasons. As originally conceived, the Pong assignment was not designed to be an experiment to assess the benefits of one teaching style versus another or to assess the value of assigning Pong versus not assigning Pong. The genesis of the Pong assignment was the recognition by the professor that many students were not learning well the concepts discussed in class and practiced in the homework and lab assignments. Instead, many students were merely "fiddling" with their code until it worked and then moving on to the next problem, with the apparent goal of just finishing the assignment, not learning the concepts being practiced. Specifically, the professor felt that the students were not learning the concepts well enough to be able to apply them effectively to new problems and new situations. To address this concern, the professor sought an engaging, multi-week project that would require the students to attain a level of understanding much deeper than prior assignments. Also, such an in-depth mid-term project should prepare the students well for their end-of-semester term projects.

Another factor contributing to the difficulties in assessing the effectiveness of the Pong assignment is the small amount of student assessment data available. The amount of assessment data for the Pong assignment is small because ELEC 247 has only been taught four times by the author of this paper with enrollments ranging from thirty to thirty-six students (who were placed in two-student groups for the Pong assignment). This small number of students and lab groups would make conclusions drawn from control and experimental groups unreliable. Furthermore, it may well be unfair to offer different instruction to two separate groups of students while giving all the students credit for the same course.

Finally, it is not feasible to separate out all the factors other than the Pong assignment that affect student learning and performance. This limitation weakens conclusions drawn from comparisons of assessments before and after the Pong assignment or from comparisons of assessment data from different academic years. Assessments of student performance before and after the Pong assignment can be done, but Pong is nominally a four-week assignment and numerous topics and examples that are not directly associated with Pong are discussed in class during these weeks that could impact these assessments. Assessments of student performance from different academic years can

also be influenced by differences in the course content, the teaching style and experience of the professor, and the quality and experience of the students.

Improvements in the Depth and Level of Learning

Although the difficulties cited in the previous section limit the rigor of the conclusions drawn from the student assessment data related to the Pong assignment and its learning objectives, the assessment data are interesting and they do provide compelling evidence that the Pong assignment is meeting its educational objectives.

One way to investigate the impact of the Pong assignment is to measure how student performance changes as the assignment was introduced and subsequently changed. The author of this paper has taught the microcontroller course four times, but only used the Pong assignment in the last three times. The first time this course was taught (in the 1999-2000 academic year), a less complex and less challenging mid-term project was assigned which the professor felt did not meet the course objectives. Additionally, each time the Pong assignment has been given, its feature set and difficulty have increased. In the 2000-2001 academic year, the paddles were only moved vertically and the Pong bonus task was to improve the game display by only redrawing the paddles when they were moved. The author did not teach this course in the 2001-2002 academic year. In 2002-2003, the Pong sound effects were added and the task for the Pong bonus was to implement full-motion paddles. In 2003-2004, the beeping and full-motion paddles became standard requirements and adding "english" to the ball on paddle hits was the task for the Pong bonus (as discussed previously).

As a confirmation that the difficulty of the Pong assignment was increased, the average grades on the Pong assignment for each year it has been assigned are shown in Figure 7. The left chart axis and the white-colored bars in Figure 7 show the average percent grade on the Pong assignment and the right axis and the black-colored bars show the average Pong bonus points awarded. As can be seen from this chart, the average Pong grade has decreased only slightly (98% to 94%), but the average number of bonus points awarded has decreased significantly (4.5 points to 2.3 points) as the difficulty of the Pong assignment has increased.

To assess the impact of the inclusion and complexity of the Pong assignment upon student learning, student performance on three key course concepts as measured by questions on the final exam is shown in Figure 8. The three key course concepts tested were: addressing modes, programming, and system design. In the addressing mode questions, students were asked to interpret the behavior of small programs that employ various addressing modes and data structures. The students must indicate the results of each instruction in the programs as they execute. For the programming questions, the students are asked to write programs containing loops, conditional branches, subroutines, and simple data structures including arrays, strings, and flags. For the system design questions, the students are given descriptions of the functions of a microcontroller-based system and asked how they would implement the system using various devices along with interrupts and/or polling. These questions can only be answered successfully if the students have an in-depth understanding of these concepts. Furthermore, these questions involve the Application, Analysis, and Synthesis levels of learning in Bloom's taxonomy [3]. The system design questions also involve Bloom's highest level of learning, Evaluation. The grades shown for each of these concept areas in Figure 8 represent the average percent score on the exam questions related to the each of these concepts.

As shown in Figure 8, student performance in each of these concept areas has generally risen from year to year, with the best measured performance occurring in the most recent year. The largest improvements have been in the areas of addressing modes and programming, the concepts that must be understood well in order to complete the Pong project successfully.

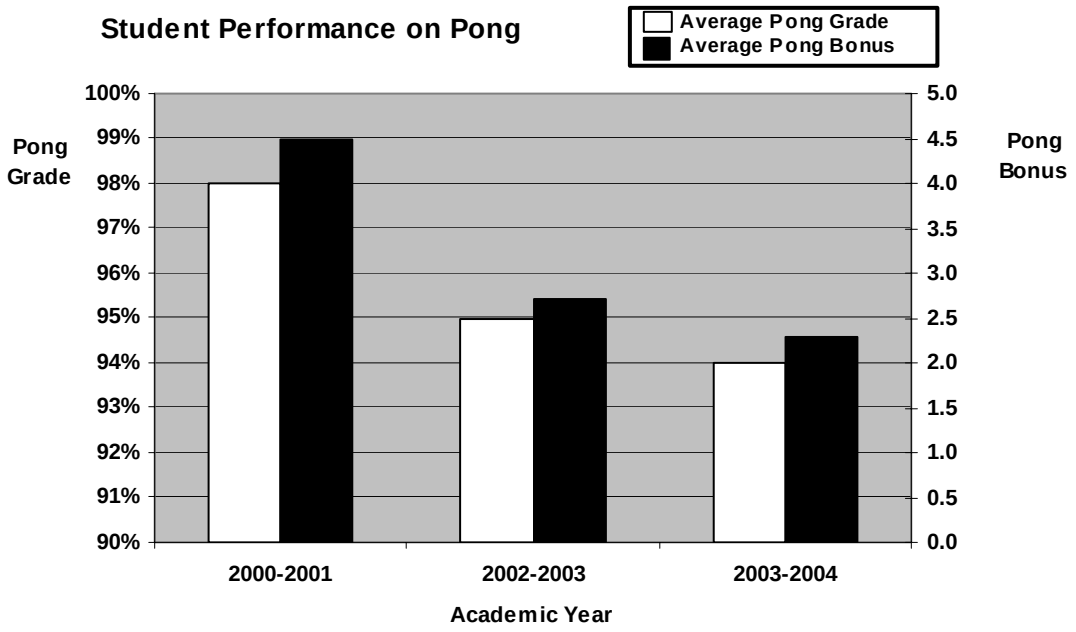

Figure 7: Average Pong Grade and Average Pong Bonus.

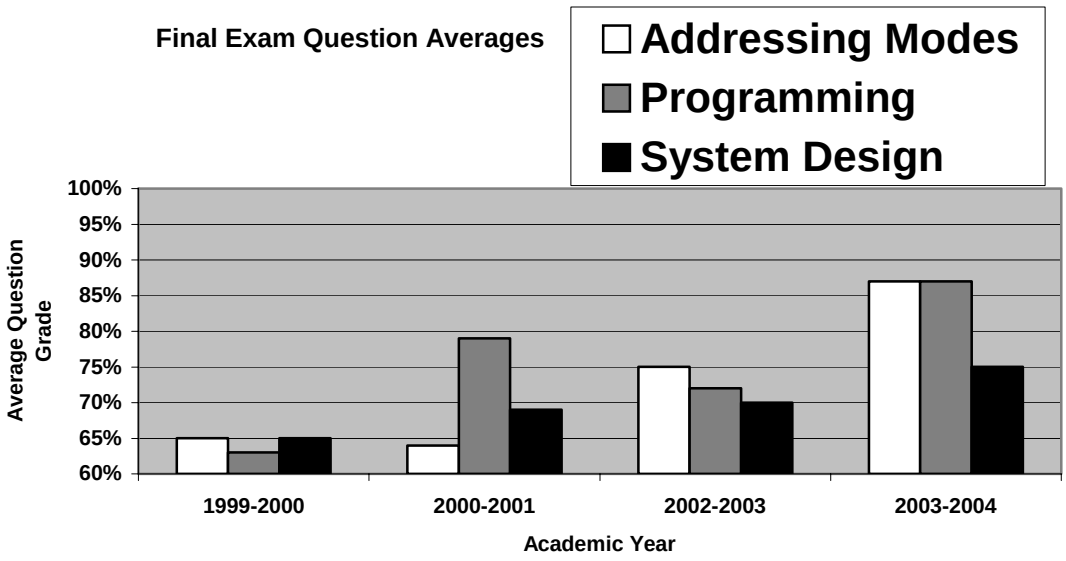

Figure 8: Final Exam Question Averages by Topic and Academic Year.

So, as the Pong project was introduced and its difficulty increased, student performance on key course concepts has improved significantly. This strongly suggests that the Pong assignment is a significant contributor to this increase in student performance and learning. As such, the Pong assignment is meeting its goals of improving both the depth and level of learning. Other factors may also contribute to this improvement, including improvements in the teaching style and experience of the professor and expected variations in the quality and experience of the students in the course.

Encouraging Students to Become Confident, Independent Learners

Near the end of the course, students implement a term project of their own choice and specification. The students are required to apply

skills emphasized in the course to a design project of interest to them. The main pedagogical goal for the term project is to serve as a capstone for the course, solidifying the learning of the course concepts. Because the students select their own projects, the term project can be used as an indicator of the students' confidence in their skills and their willingness to learn on their own.

In the most recent offering of the course, every term project employed devices that had not been used previously in the course. Examples of these devices include: LCD displays, keypads, temperature sensors, distance sensors, relays, accelerometers, motor controllers, and wireless serial links. Furthermore, the professor intentionally did not learn how to use the most popular of these devices (the LCD displays and the keypads) and informed the students that they were "on their own" with respect to using these devices.

Figure 9 shows that 92% of the term project demonstrations in the 2003-2004 offering of the course were successful, a notable success rate given the incorporation of devices new to the students in each project. This success rate is higher than it ever had been before and significantly higher than the 61% success rate when the Pong project was not used. In Figure 9, white and black bars represent the total number of projects and the number of successful projects respectively. A term project demonstration was deemed successful if it had the majority of its features working, receiving a grade of at least 85%. The percent displayed near the top of each black bar indicates the percentage of successful term projects. As with the student performance data presented in Section 4.2, this term project performance data does not rigorously prove that the Pong mid-term project made the key difference in improving term project performance. However, the use of new devices in the term projects and the term project performance data provide compelling evidence that the Pong project encouraged students to become confident, independent learners.

Pong Opportunities and Limitations

The Pong assignment has several additional attributes that can be used to demonstrate and/or

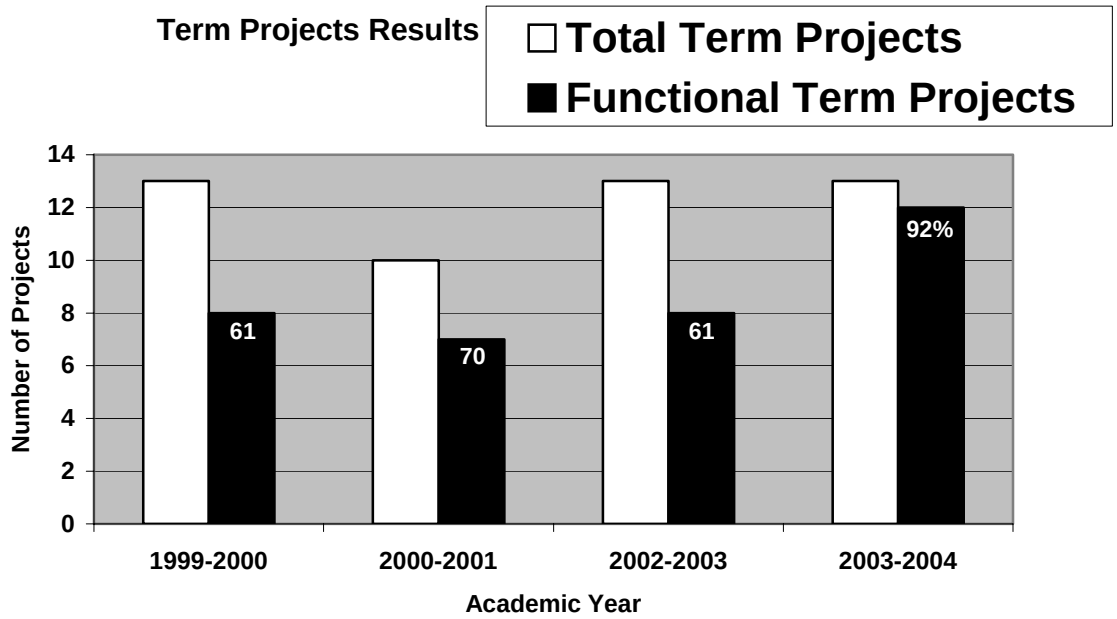

Figure 9: Term Project Success Rates.

apply other concepts. Pong is primarily a software project that only requires straightforward application of basic assembly-language programming concepts. This quality of the Pong program is due to the simple structure of the program's main loop as defined by the skeleton program (see Figure 3). This main loop structure constrains the complexity of Pong. By altering this structure, Pong could also be used to introduce and provide experience with higher-level system design concepts and choices. For example, the students could be asked to choose whether to use interrupts or a polling loop to monitor the joystick positions. Interrupts, rather than the variable delay loop provided in the Pong skeleton program, could also be used to provide the delays needed for managing paddle and ball movement and even provide different delays for each.

As an example of how Pong can be used to demonstrate high-level system design concepts, the author of this paper has used a modified version of Pong later in the course to demonstrate the concurrency problems that can arise with the use of interrupts and shared resources by using two different timer interrupts to manage ball and paddle movement on the screen. Without properly managing access to the screen by the interrupt service routines for the ball and paddle timers, this shared resource (the terminal screen) can be easily corrupted. Using Pong to provide practice with interrupts and concurrency issues may require shifting the assignment to a later point in the course in order to provide time to introduce and discuss these higher-level system design concepts in class.

Although Pong has been a very successful assignment, certain problems can limit its effectiveness. Using the Pong assignment year after year can pose some problems if the current students begin to use the solutions created from previous years. To address this concern, the professor has requested that previous students not share their Pong programs with newer students. The professor has also changed the Pong assignment and the Pong bonus each year. Additionally, many options exist for altering the functionality of Pong slightly that require significant changes to the Pong program. Some of these options are as follows:

- Show a trail of ball positions as the ball moves across the screen.
- Once both players have reached a certain score, place a small barrier on the net that reflects the ball when hit.
- Change the size of the paddles and/or the ball speed after a certain score is attained.
- Implement Pong without using a delay loop. This would require the use of timer interrupts as discussed above.
- Play a simple song in the background during the game.

To date, students in unstructured, two-person groups have done the Pong project. This lack of structure may limit the learning of the students. A more effective approach may be to require the students to practice pair programming [2]. With pair programming, the students assume two distinct roles: one enters the code and the other monitors the code, offering comments and feedback. These roles are then alternated at regular intervals. Pair programming has been shown to be effective in improving the quality of the programs being written as well as improving the learning of programming skills [11, 9]. Without imposing the practice of pair programming, some groups and individuals may exhibit substantially less learning than they would otherwise.

Conclusions

Although encouraging students to learn assembly-language programming well can be a difficult task, including the engaging, but challenging Pong mid-term project in an introductory assembly-language programming course promotes deep learning of assembly-language programming concepts while also inspiring the students to become independent learners. The Pong project promotes deep learning by using a problem-based learning approach where students work collaboratively with feedback on a complex problem that has many different solutions. The Pong project also

improves learning by consistently requiring students to demonstrate learning at the upper levels of Bloom's taxonomy. Additionally, the Pong project addresses the multiple learning styles of students by augmenting the more traditional, lecture-style teaching approach with an active, collaborative assignment that begins with clearly specified instructions regarding functionality and implementation requirements and moves to more vague implementation requirements while maintaining clear functional requirements. The more difficult parts of the Pong project also require students to learn without explicit, step-by-step guidance from the professor, which helps the students become independent learners and helps them recognize their potential.

The analysis of the data assessing the effectiveness of the Pong assignment supports the above conclusions. Assessment of student learning in the key areas of addressing modes, programming, and system design all show improvement as the Pong assignment was introduced and its difficulty increased. Furthermore, the demonstration success rate for the end-of-semester, student-specified term projects shows a similar improvement as the difficulty of Pong was increased. Students also demonstrated through their term project selections and the devices they chose to incorporate into those projects that they were confident in their abilities to be independent learners. This paper describes in detail the structure and goals of the Pong project and these should enable other educators to employ a similar project in their introductory assembly-language programming courses. Furthermore, the discussion of the opportunities and limitations of the Pong project provide guidance for managing the Pong project as well as suggestions for altering the Pong project to make it an effective vehicle for demonstrating and/or providing design practice with higher-level system design concepts such as polling, interrupts, and concurrency.

References

1. Axiom Manufacturing. CME-11E9-EVBU MC68HC11 Low Cost Development System. http://www.axman.com/Pages/neuprod8cme11e9.html.

2. Kent Beck. Extreme Programming Explained: Embrace Change. Addison-Wesley Professional, 1999.

3. B. S. Bloom, editor. Taxonomy of Educational Objectives: Handbook I: Cognitive Domain. New York: Longmans, Green, 1956.

4. Noel Entwistle. Approaches to Studying and Levels of Understanding: The Influences of Teaching and Assessment. Higher Education: Handbook of Theory and Research, XV:156–218, 2000.

5. Noel Entwistle. Promoting Deep Learning Through Teaching and Assessment: Conceptual Frameworks and Educational Contexts. In Proceedings of the Teaching and Learning Research Programme (TLRP) Conference, Leicester, U.K., November 2003.

6. Richard M. Felder and Linda K. Silverman. Learning and Teaching Styles in Engineering Education. Engineering Education, 78(7):674–681, 1988. http://www.ncsu.edu/felder-public/Papers/LS-1988.pdf.

7. Hilgraeve, Incorporated. HyperTerminal. http://www.hilgraeve.com/htpe/index.html.

8. Steve L. Kent. The Ultimate History of Video Games: from Pong to Pokemon and Beyond... Prima Publishing, 2001.

9. Charlie McDowell, Linda Werner, Heather Bullok, and Julian Fernald. The Effects of Pair-programming on Performance in an Introductory Programming Course. In Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education, pages 38–42. ACM Press, 2002.

10. Motorola. Motorola M68HC11 Reference Manual. http://e-www.motorola.com/files/ Microcontrollers/doc/ref manual/M68HC11 RM.pdf.

11. John T. Nosek. The Case for Collaborative Programming. Communications of the ACM, 41(3):105–108, 1998.

12. Red Hat. Cygwin: A Linux-like Environment for Windows. http://cygwinT. com/.

13. Brinkley Sprunt. ELEC 247: Microcontroller System Design. http://www. eg.bucknell.edu/~bsprunt/classes/elec_247_ spring_2004/elec_247.htm

14. Stephane Carrez. GNU Development Chain for the 68HC11 and 68HC12. http://stephane .carrez.free.fr/m68hc11 port.php.

15. Donald R. Woods. Problem-Based Learning: Helping Your Students Gain the Most from PBL. Donald R. Woods, 1995. http://chemeng.mcmaster.ca/pbl/pbl.htm.

Biographical Information

Brinkley Sprunt is an Assistant Professor of Electrical Engineering at Bucknell University. His research interests include computer performance modeling, measurement, and optimization. Prior to joining Bucknell, he worked at Intel Corporation for nine years where he was a member of the architecture teams for the 80960, Pentium Pro, and Pentium 4 projects. On the Pentium 4 project, he led the performance validation team and was the principal architect of the Pentium 4 performance monitoring capabilities.