Designing and Implementing an Embedded Microcontroller System: Tetris Game

Tyler W. Gilbert, Barry E. Mullins, and Daniel J. Pack
Department of Electrical Engineering
US Air Force Academy

Abstract

In this paper we present the software and hardware design experience of a junior cadet majoring in electrical engineering at the U.S. Air Force Academy as he completed an embedded system project in a second microcontroller course. The paper also includes the corresponding observations made by his instructors. Some of the topics of this semester-long course are programming microcontrollers using C, software and hardware design techniques, interfacing devices to buses including fan-out issues, as well as buffering and timing analysis for interfacing devices. As a course requirement, an embedded system project must be completed by each student using the hardware kernel they had developed throughout the semester. The Tetris game was an ideal final project for this course due to its demanding software and hardware requirements, which required the student to apply and incorporate hardware and software skills and to perform tradeoff analyses using what he had learned throughout the course: (1) memory expansion, (2) input/output interfacing, (3) embedded software development, and (4) stand-alone system development. This paper provides the reader with the step-by-step procedure used at the Academy for the completion of the embedded system project and shares the lessons learned from both student and instructors' perspectives in the context of completing a successful and challenging Tetris game project.

Introduction

Second class cadet (junior year) electrical engineering (EE) majors at the United States Air Force Academy are given the opportunity to take a second course in microcontroller design—EE 383, Microcomputer System Design. The goals of the course are "Cadets shall develop the ability to design, build, program, and debug a stand-alone hardware kernel consisting of a microcontroller, memory, input/output (I/O) ports, and standard "glue" logic for use in embedded microcontroller applications." [1] The course helps students learn the skills required to implement and program microcontrollers, using the Motorola 68HC912B32 as the platform, to construct embedded systems. Each cadet in the course is required to complete a final project that ties all concepts together in an interesting microcontroller-embedded project. The final project described in this paper was accomplished by the lead author and was identified by the instructor as one of the outstanding example projects in the course. The student chose to build the game of Tetris using the 68HC12 along with an AND1371 graphical liquid crystal display (LCD) and a Nintendo Entertainment System (NES) controller. Figure 1 illustrates the various components of the completed project. The student chose this project because it was challenging and stimulated his interests. He was given three weeks to complete the final project.

The experience was invaluable in motivating the student to become a better digital electrical engineer for three reasons: (1) the project stimulated his personal interests; (2) the project enriched his abilities to gain new knowledge as he researched the requirements and capabilities of the AND1371, an NES controller and the game of Tetris; and (3) programming video game graphics equipped the student with skills to tackle more complex microcontroller applications. All courses within the digital sequence at the Air Force Academy strive to provide theses experiences to the students. We have found allowing the students the freedom to select their final projects provide students with extremely rewarding experience of dealing with microcontroller/microprocessor systems.

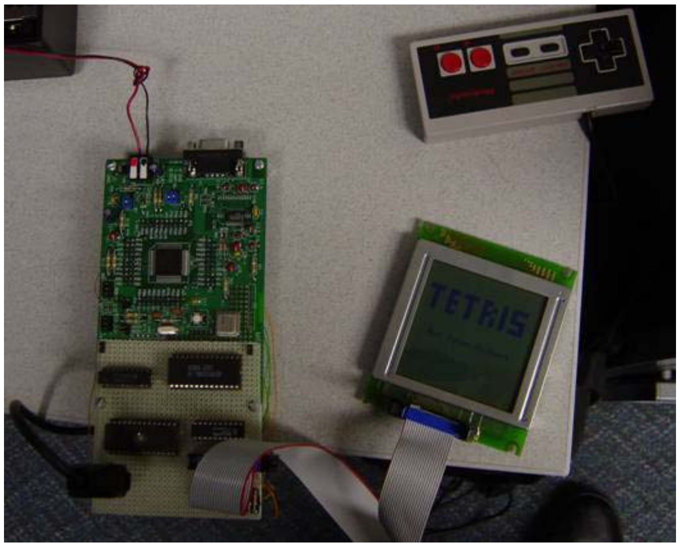


Figure 1. Picture of Tetris Project

The design of this project involved both hardware and software. The hardware portion of the design was divided into the following three sections: (1) memory expansion, (2) an AND1371 graphical LCD interface and 3) an NES controller interface. The memory expansion was completed as a part of the normal course work. Although the 68HC912B32 microcontroller possessed sufficient memory for the project, the memory expansion was required as a pedagogical exercise. The LCD and the controller were exclusive to the final project as was the majority of the software.

The corresponding software design was broken down into several milestones in order to systematically create the Tetras game using sound engineering practices. The main programming challenges, listed in order of practical implementation and not necessarily difficulty of programming, were (1) drawing blocks on the LCD, (2) making the blocks fall and rotate on the screen, (3) saving the position of each fallen block, (4) making sure blocks do not mesh into other blocks or boundaries, (5) generating a random number to vary the shape of the new blocks, and finally (6) incorporating extra features to the game, such as a scoreboard, a next-piece window, a number-of-lines completed entry, and an introduction screen, background wallpaper, and record of high scores with players' initials.

Design

In this section we describe both the hardware and software designs of the project. The hardware design consisted of integrating the 68HC912B32 with the AND1371, external EPROM (to store the code) and an NES controller. Once again, the internal memory of the microcontroller was not used for pedagogical reasons. In this paper we focus on the challenges a student may face in integrating the hardware and software components of such a project.

Hardware Design The hardware design for the project is shown in Figure 2. Although the hardware documentation for the AND 1371 provided sufficient information to correctly connect the LCD to the microcontroller, the complexity involved in operating the AND1371 made it challenging for the student to implement. One of the main problems encountered was wiring errors found within the data, control, and address buses. The problem was fixed by analyzing the signals using a digital logic analyzer and by careful review of the application notes.

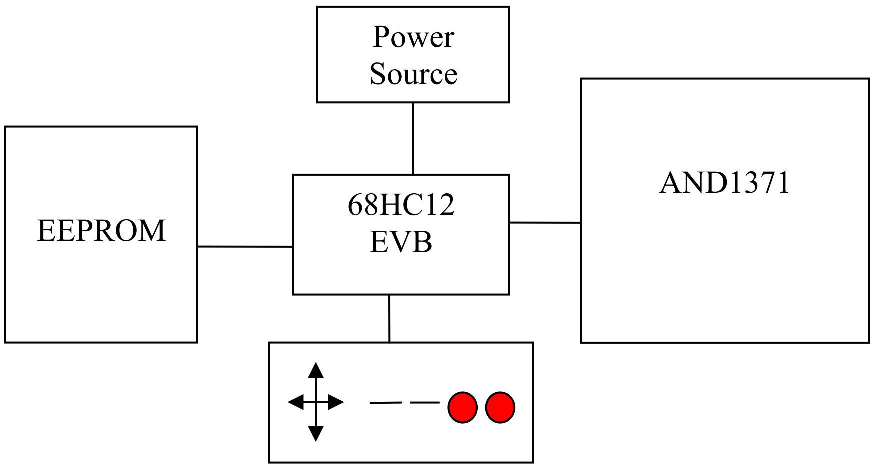


Figure 2. Block Diagram of Hardware Setup

Adding external memory required (1) performing time analysis for the data, control, and address signals between the 68HC12 and an EPROM memory chip, (2) programming a GAL chip to decode addresses, (3) incorporating an address buffer to demultiplex the address bus, and (4) wiring the components together. Figure 3 shows the hardware diagram for the memory expansion task. Learning the intricacies of booting a microcontroller from external memory was not a trivial task

The NES controller presented its own challenges to the student. Since the student was unable to obtain technical documentation, he performed reverse engineering to obtain and implement the specifications and requirements. After his analysis was complete, he discovered that others have performed the same reverse

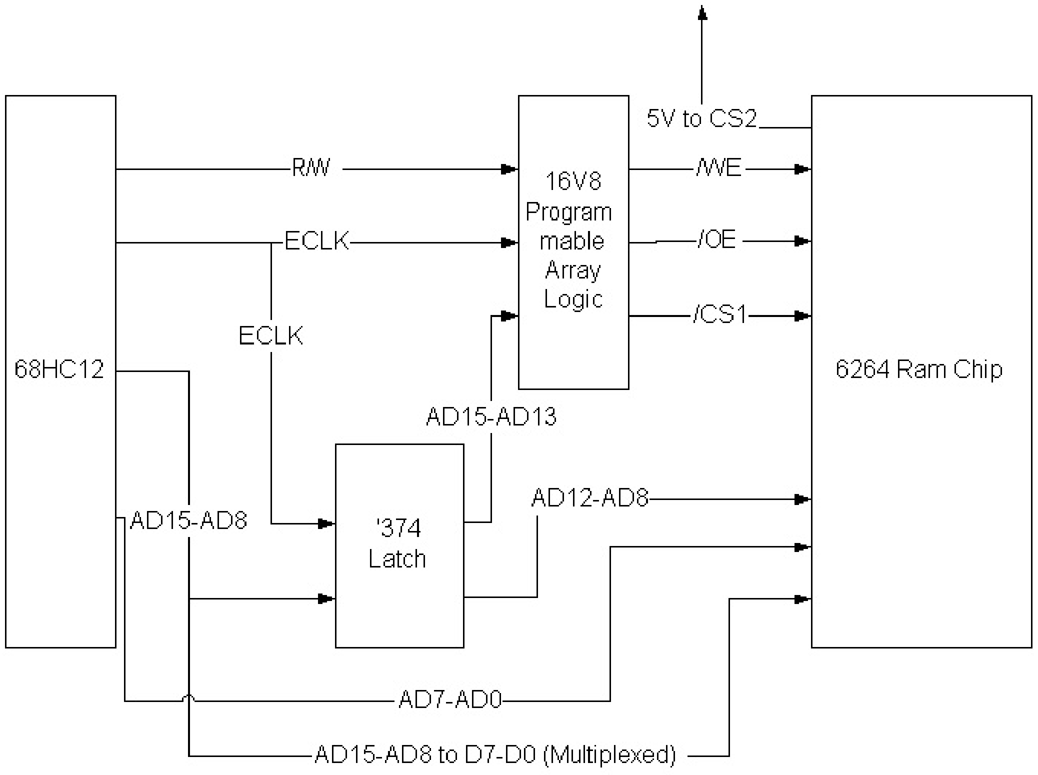
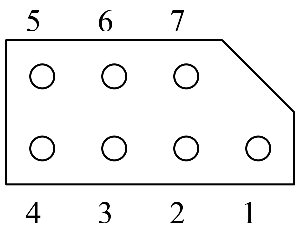


Figure 3 Memory Expansion Hardware Diagram

engineering task and published their findings online; regardless, the experience for the student was extremely valuable. The NES controller information can be found at http://www.mit.edu/~tarvizo/nes-controller.html. The student had to assume the voltage levels used with the 68HC12 were compatible with the NES controller and the microcontroller sourced sufficient current to drive the device. Fortunately, both



Pin	Function	Color
1	Ground	brown
2	Pulse	red
3	Latch	orange
4	Data	yellow
5	NC	NA
6	NC	NA
7	Power	white

Figure 4. NES Controller Pins with Picture of Connector (http://www.mit.edu/~tarvizo/nes-controller.html)

assumptions turned out to be true. Figure 4 shows the hardware connections for the controller. The latch and pulse pins should be wired to output ports on the microcontroller. The data line should be wired to an input on the microcontroller.

Software Design The software design involved programming the hardware components to work together and writing the code for the Tetris game. As the first step, the student identified major tasks and designed the required task flow using the flowchart shown in Figure 5.

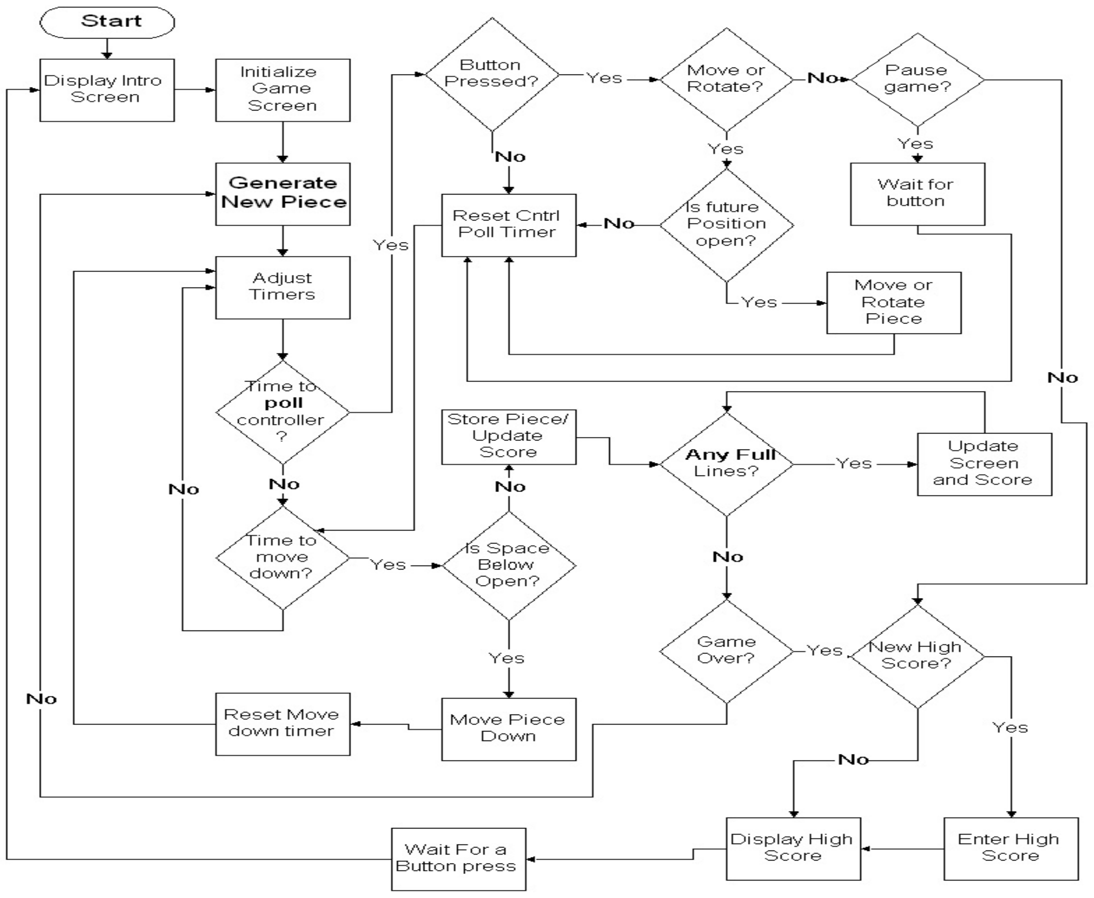


Figure 5 Software Flow Diagram

The NES controller proved much simpler to implement than first anticipated. It was important to test the controller on an NES console to verify that the controller was functioning properly. At first the student made the mistake of having the 68HC12 reading the pulse count from the device when it should have been writing to it. Figure 6 shows the waveforms that the controller must receive. The pulse period is given as 12 microseconds; however, it was found that the device was rather insensitive to timing.

Once the hardware devices were successfully implemented and tested using software, the time came to write the code for the Tetris game. The main hurdles in coding were (1) drawing blocks on the LCD, (2) making the blocks fall and rotate on the screen, (3) saving the position of each fallen block, (4) making sure blocks do not mesh into other blocks or boundaries, (5) generating a random number to vary the shape of the new blocks. This paper only offers a brief lead on how these hurdles were overcome. The

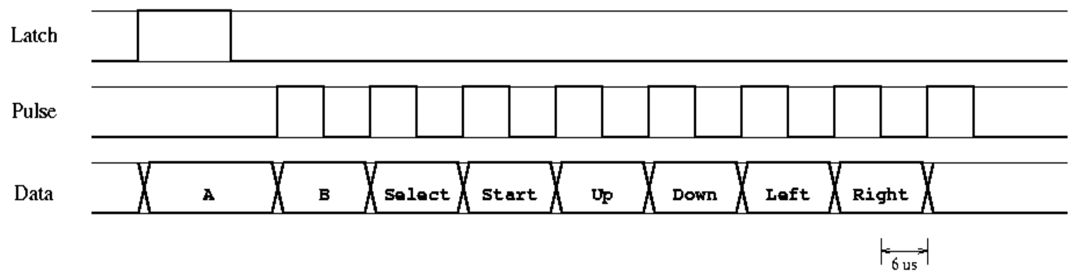


Figure 6. Waveform plots for the NES controller (http://www.mit.edu/~tarvizo/nes-controller.html)

easiest way to draw blocks on the screen was to fill an entire video memory location rather than trying to manipulate the bits in multiple memory locations. To rotate the blocks, the student created a simple Sine/Cosine lookup table to switch the coordinate system back and forth from polar to Cartesian and vice-versa—polar coordinates to rotate and Cartesian to move blocks left, right, and down. The student created a matrix that saved every fallen piece. To avoid meshing, the future position of any part of the moving block was checked for vacancy before allowing the move. Pulling a number off the 68HC12's free running timer generated a number with sufficient randomness for each new block.

Discussion

The final project of the second microcontroller course offers an opportunity for students to design and implement a microcontroller-based embedded system. The instructors see such projects as extremely valuable to student learning. Some of the more germane reasons are (1) an opportunity to reflect on the course materials and apply the knowledge learned, (2) experience to improve independent learning skills, (3) a chance to solve open-ended design problems, encouraging students to use their imaginations, and (4) an opportunity to exercise both software and hardware skills. The overall experience supports and enhances the students' probability of success in their senior capstone design course.

Conclusion

In this paper, we presented a case study of a student project, the Tetris game, in a second microcontroller/microprocessor course at the US Air Force Academy. The project provided the student with ample opportunities to improve his hardware and software engineering skills. Making the project open-ended made it fun for the student. Although the project time took away some lecture time from the course, the experience of the student demonstrated that the tradeoff was worthwhile. The detailed course information can be obtained by contacting the third author.

References
1. EE 383 Course Syllabus, United States Air Force Academy CO, 30 December 2003.

TYLER W. GILBERT graduated from Box Elder High School in Brigham City, Utah in June of 1999. Following his freshman year, he took two years off school and re-entered the Academy in the class of 2005. He will receive a B.S. on June 1st, 2005 in Electrical Engineering. He will go on to serve in the Air Force as a developmental engineer.

BARRY E. MULLINS is an Assistant Professor of Computer Engineering in the Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson AFB OH. He is a registered Professional Engineer in Colorado and a member of Eta Kappa Nu (Electrical Engineering), Tau Beta Pi (Engineering), IEEE (senior member), and ASEE.

DANIEL J. PACK is a Professor in the Department of Electrical Engineering at the United States Air Force Academy, CO. He has co-authored two textbooks on embedded systems (*68HC12 Microcontroller: Theory and Applications* and *Embedded Systems*) and published over 70 refereed journal and conference papers on control, robotics, pattern recognition, and engineering education.