Hand-Held Video Games Using a PIC Microcontroller and Graphic LCD Module: A Capstone Design Project

James S. McDonald

Kettering University

Abstract

This paper describes a capstone design project carried out by several groups of senior undergraduate students in the author's computer engineering capstone design course during the Summer 1998 term at Kettering University. The basic project requirement was to design and build a hand-held video game using a Microchip PIC 16C74A microcontroller, a 128×128-pixel graphic LCD module based on the industry-standard Toshiba T6963C graphic LCD controller, and only incidental additional parts. The paper describes the students' backgrounds, the project assignment and the motivation for choosing it, and several video games that students designed and built. It concludes with an informal assessment of the project's success and preliminary ideas for related projects based on this experience.

I. Introduction

ECE 403 *Computer Engineering Capstone Design* is the capstone design course in the Computer Engineering program in the Department of Electrical and Computer Engineering at Kettering University (formerly GMI Engineering and Management Institute). Computer Engineering is a new program in the Department, and this was the first time the capstone course was offered. It is a four credit-hour course lasting twelve weeks, and eight students were enrolled for the Summer 1998 term. Students worked in groups of two, and completion of the project described herein along with appropriate written and oral reporting as the term progressed were the sole requirements of the course.

Student Background

The Computer Engineering curriculum puts a strong emphasis on microcontrollers and embedded systems, with a required three-course sequence in these areas leading up to the capstone course:

1. ECE 374 *Microcomputers I* gives an introduction to microcontrollers, including assembly-language programming, software design, and some interfacing. It covers Chapters 1 through 6 and parts of Chapter 7 in the text *Microcomputer Engineering* by Gene H. Miller[1]. The text, and therefore the course, is based on the Motorola 68HC11 family of 8-bit microcontrollers, and laboratory projects use Motorola's 6811 EVB evaluation board.

2. ECE 474 *Microcomputers II* introduces more advanced topics in assembly-language programming and software design through the study of a considerably more advanced microcontroller: the Motorola MC68332, a 32-bit processor and member of the large 68000 series of processors. The text used[2] is specific to this processor, and laboratory projects use Motorola's M68332EVK evaluation kit.
3. ECE 480 *Real-Time Embedded Computers* covers Chapters 8 and 9 in Miller's book[1], taking the students through the development of a simple real-time operating system and a weather station application for it. All software and hardware is developed using Motorola's M68HC11EVS evaluation system.

At the end of this course sequence, students have a good deal of experience in assembly-language programming and hands-on development and debugging using standard evaluation-type development systems (as opposed to more sophisticated in-circuit emulators).

Motivation for Project Assignment

A wide range of design projects might be considered "appropriate" in a capstone course following this sequence. For example, a project might be chosen that builds on and strengthens the students' already considerable expertise with Motorola microcontrollers and development systems, moving up to a higher-end in-circuit emulation type of development system and an application with fairly demanding real-time requirements. Alternatively, a project might be chosen to broaden the students' experience into different microcontrollers, development environments, and peripheral hardware.

The latter approach was taken in the choice of project in ECE 403, for several reasons:

- All the Motorola microcontrollers students were familiar with from previous courses share a unique Motorola design philosophy and many broad characteristics such as complex instruction set rich with addressing modes, multi-cycle instruction execution, a limitless call stack stored in memory, and common program and data memory. The PIC, on the other hand, has none of these features, and learning it gave students a much broader perspective on the microcontroller industry as a whole.
- Many embedded applications are characterized by the need to interconnect a microcontroller with quite complex peripherals, which the students had not seen in previous courses. The graphic LCD module is an excellent example of such a peripheral, and the project required students to learn its operation and apply it.
- Embedded applications are also typically characterized by the need to work within severe resource limitations. In order to capture this requirement using a higher-end microcontroller, the project would have to be far too involved for undergraduates to take on in the limited time available in the capstone course. The PIC, on the other hand, is quite limited in its program and data memories, its instruction set, and its call stack, amongst other things, so that a project of realistic scope can tax its resources.

Finally, choosing the design of a hand-held video game as the basic goal of the project was very motivational for the students, and they were able to be very creative in the preliminary design

phase because of their experience with commercially available games.

The remainder of the paper describes and discusses the project and its results in some detail. Section II describes the project assignment, discusses the motivation behind some of the particular choices made, and describes the components, development environment, and other resources available to the students in completing it. Section III describes briefly the projects completed by three student groups. Section IV gives an informal assessment of the project's success and some preliminary ideas for related projects based on this experience.

II. Project Assignment

The project was described to students in a single sentence: "The project for this term, in a nutshell, is to design, build, and test a hand-held video game." The precise game was left open (subject to approval of the instructor), but all games were to satisfy the following minimum requirements:

- The game shall incorporate a Microchip Technology PICDEM-2 demonstration board with a PIC16C74A microcontroller installed. No additional microcontroller-like device or other source of "processing power" may be used.
- The game's primary video display shall be a Hantronix HDM128GS12-1 graphic LCD module. Additional display elements (e.g., LEDs) may be used but should be only secondary (e.g., power-on indication).
- The game shall be housed securely in an enclosure that can be held in the hand(s).
- The game shall be operable for at least one hour without any external power source or other components.
- Each game shall function according to the minimum requirements established by the group in its project proposal.
- Total cost to construct the game is not to exceed $50.00 over and above the cost of any components provided by the instructor to all groups. The following guidelines apply:

Each group was provided their own PICDEM-2 demonstration board (available from Digi-Key Corporation at $99, with a substantial discount for educational customers), PIC 16C74A UV-erasable EPROMed microcontroller (also from Digi-Key at about $15), and Hantronix HDM 128GS12-1 graphic LCD module (from Hantronix Inc. at about $45).

Major Components

The PIC 16C74A is a relatively high-end member of Microchip's extensive PIC family of 8-bit microcontrollers, with 33 parallel I/O pins, 8 A/D channels, an I^2C serial peripheral interface, and several timers. It's packaged in a 40-pin DIP package. Like all PICs, it has separate data and program memories, with 4K words of program memory and 192 bytes of data memory on-chip. There is no external access to its address or data bus, so any required off-chip memory must be accessed serially via the I^2C bus. Also like all PICs, it has a minimal set of only 35 instructions, all of which (except branches) operate in a single cycle of the (up to) 20-MHz clock.

The PICDEM-2 is a simple demonstration board that supports a range of 28- and 40-pin PIC microcontrollers, including the 16C74A. It includes an RC clock (with provision for changing to a crystal clock), an eight-LED display, two pushbutton switch inputs, a serially connected 128-byte EEPROM, and a small (9×24 holes on 0.1-inch centers) prototyping area. It also has a 14-pin connector designed to interface to a standard text LCD module (not useful for the standard 20-pin graphics LCD module interface) and an 8-pin connector designed to interface to a standard 12- or 16-key switch-matrix keypad.

Developing a system using the PICDEM-2 involves removing and UV-erasing the 16C74A, re-programming it using a device programmer (in this case a Logical Devices ChipMaster 6000, although inexpensive PIC programmers are available from Microchip and various third parties), re-installing it on the board, and resetting it. There is no debugging interface, so programs under development cannot be controlled or traced and must use the eight LEDs to output any debug "messages" as they execute. Customarily, programs under development are thoroughly tested first in simulation before in-system testing is attempted.

A block diagram of the HDM 128GS12-1 graphic LCD module is shown in Figure 1. It consists of a 128×128-pixel black-and-white LCD display, an industry-standard Toshiba T6963C graphics controller, an 8K×8-bit graphics memory, and LCD bias circuit. The standard 20-pin interface includes eight data lines, control signals, and power, and is accessible via a surface-mount 1-mm-center connector. The module is quite compact (70×72×13 mm), and the LCD display supports LED backlighting, although this was not used in the project because of high power requirements. The T6963C controller supports both graphics and text (in a 16×16 array of 5×8-pixel characters) in separate memory areas and can display graphics only, text only, or both graphics and text overlayed in various ways.

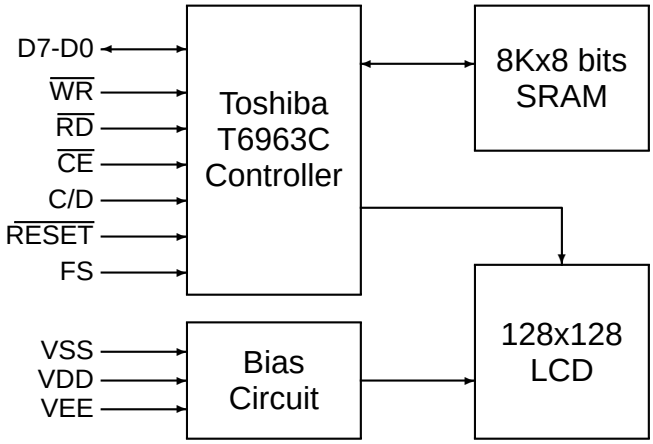

Figure 1: Block diagram of Hantronix HDM 128GS12-1 graphic LCD module

Additional Resources

In order for the students to have a reasonable chance of success, considerable support and information resources had to be made available.

A text, *Design with PIC Microcontrollers* by John B. Peatman[3], was required for the course and several lectures were given based on the first two or three chapters. Students were left on their own to read other parts of the book as the need arose in their particular projects. This book is an *excellent* introduction to PIC microcontrollers, discussing the architecture and instruction set, assembly-language programming and development techniques, subroutines, internal and external peripherals, and system issues such as clocking and low-power operation. It places particular emphasis on the 16C74A and the PICDEM-2 demonstration board. Several very instructive demo programs are discussed in detail in the book and are available in electronic form from the author (at URL `http://users.ece.gatech.edu/~peatman/picbook/`).

MPLAB, Microchip's complete integrated development environment for the PIC family, was made available on Department computers and several lectures, demonstrations, and an introductory lab assignment were given to get students started using it. MPLAB is a powerful and complete suite of development tools and is very well maintained and robust, enjoying an excellent reputation in the PIC user community. It includes the MPASM macro assembler, the MPLINK linker, MPLAB Editor, and MPLAB Project Manager. Most importantly, it includes the excellent MPLAB-SIM simulator, which is essential when developing using a simple reset-and-go demonstration board like the PICDEM-2. It runs on Microsoft Windows 9x/NT platforms and is available free of charge from Microchip (at URL `http://www.microchip.com`), so most students installed and used their own copies on their home computers and/or laptops. MPLAB v3.99.06 was used in the course, and v4.0 is now available.

In part because of the popularity of PIC microcontrollers and graphic LCD modules with hobbyists, a wide range of resources are available on the Internet. A course web page (URL `http://www.kettering.edu/~mcdonald/class/ece403/`) was constructed to point students to some of the more useful of these, including:

- Microchip's web page, containing PIC-family technical manuals and a wide range of application notes in PDF format in addition to the MPLAB software
- A collection of hobbyist web pages describing PIC projects
- The LCD manufacturer's web page (`http://www.hantronix.com/`), with technical manuals and application notes on the LCD module in PDF format
- A collection of hobbyist pages describing graphic LCD applications using various microcontrollers

Supplying power to the LCD display, which requires +5 V and −15 V for its bias circuit, from a single battery source turns out not to be a trivial task, and an LCD bias-generator chip (Maxim MAX 766) and detailed application notes on its use were provided to all groups. The small-geometry (1-mm centers) connectors on the LCD modules were problematic as well, and a converter board to the 0.1-in-center ribbon cable required for connection to the PICDEM-2 was developed cooperatively and used by all groups.

III. Student Projects

Because the major components of the project were common to all groups, the hardware block-diagrammed in Figure 2 is quite similar to that used by all groups. An entire 8-bit PIC I/O port (port D in the figure) is required for data to the LCD module, and five outputs from another port (port C in the figure) are needed for control signals to the LCD. User inputs to control the game, most often generated by simple SPST switches, require from one to five inputs depending on the project (port B in the figure). Altogether about fourteen to eighteen of the 16C74A's 33 total I/O pins are needed. Power to both the PIC and LCD module are supplied by the MAX 766 bias-supply circuit powered by a 9-V battery.

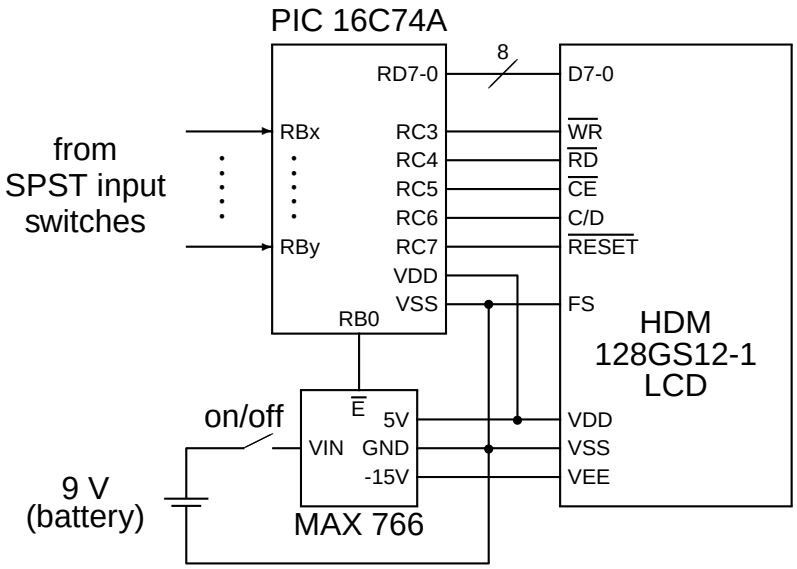

Figure 2: Hardware block diagram of "typical" project

The typical game program was a relatively simple loop structure in which player inputs are polled and the game and LCD state updated. Most groups used only a single timer interrupt to regulate the speed of loop execution and therefore of any animated motions on the LCD. This approach is in line with the basic PIC philosophy of program simplicity as well as the requirements of the project.

Battleship

Game Description. This game was inspired by the familiar naval search-and-destroy game sometimes played using pencil and paper, and also available in at least one commercial version. This version differs only in that it is single-player, as are most hand-held video games. The object of the game is to find and sink all ships hidden in a 10×10-square grid (representing the ocean) using as few total shots as possible. There are a total of five ships, ranging from five (a battleship) down to two (a PT boat) grid squares in length and placed (but hidden from the player) either horizontally, vertically, or diagonally, in the grid as the game begins. The player targets a square by moving a cursor through the grid using *Left*, *Right*, *Up*, and *Down* control buttons and fires on

the square using a *Fire* control button. The outcome of each shot, either "hit" or "miss," is indicated on the targeted grid square, and if a ship has been sunk as the result of a hit its identity is indicated briefly (for several seconds) on the screen. Total hits, misses, and shots fired so far are indicated on the screen at all times.

Implementation. This game uses only the text mode of the LCD module, as this is sufficient to support game play and considerably simpler than the graphics mode. The 10×10 ocean grid is located in the lower right-hand corner of the overall 16×16 text mode character grid. Initially all squares contain tildes (`~`) to represent ocean waves; once fired upon they're marked with either `H` or `M` as appropriate. Hit, miss, and shot totals are displayed in the upper 16×6-character rectangle of the character grid, and indications of sunken ships in the lower left-hand 6×10-character rectangle. Player input is from five standard momentary SPST buttons. An introductory screen lets the player choose one of five pre-selected ship patterns stored in the PIC program memory; there was insufficient time for the students to implement a random ship-placement algorithm. The game program requires 1476 of the 4096 available words of PIC program memory.

Breakout

Game Description. This game is based on a familiar arcade game of the same name in which a pattern of blocks are displayed on the screen and the player attempts to break through the blocks by striking them with a ball. The block pattern takes up the upper portion of the screen and a ball is "served" upwards toward them to begin a turn. A block is destroyed (it disappears) when struck with the ball, and the ball bounces off the boundary of the (former) block back toward the bottom of the screen. The player controls a paddle that can be moved back and forth across the bottom of the screen and can bounce the ball back up toward the block pattern. The ball reflects off the side and top walls in addition to the blocks and paddle, but is lost if it reaches the bottom of the screen without striking the paddle. If all the blocks are destroyed, a new "level" is reached and a new pattern of blocks appears. At each new level the speed of the ball increases, making play more difficult.

The block pattern is a regular array of five rows of seven blocks each on each level. Ten points is scored for each block destroyed on each level, and the object of the game is to score the most points before three balls are lost. The paddle is controlled by a knob, and an *Action* button causes a new ball to be served at the beginning of the game and when a ball has been lost. The game begins by displaying a welcome screen, which is exited by pressing *Action*.

Implementation. This game uses both the text and graphics mode of the LCD. The text mode is used to display, in a 16×1 row of characters across the top of the screen, the current level and score and the number of balls remaining. The remainder of the screen displays the regular array of blocks, the ball, and the paddle using the LCD's graphic mode. The paddle is controlled using a potentiometer whose wiper voltage is read via one of the PICs A/D inputs on input port pin `RA1` and the *Action* button is a momentary SPST switch whose state is read via `RA3`. To enhance playability, the game generates a tone by enabling an NE555 timer driving a piezo buzzer via output pin `RC1` whenever the ball strikes a block, a wall, or the paddle. The game program used 2010 of the 4096 available words of PIC program memory.

Street Fighter

Game Description. This game was inspired by the various punch'n'kick-style arcade and computer games in which the player controls a fighting figure who attempt to defeat an opponent by punching and/or kicking them. In this version, a player-controlled fighter faces a computer-controlled opponent. Each fighter can move left-to-right on the game screen, but cannot pass the other fighter. Each can either punch or kick, and each can block either "high" (effective against punches) or "low" (effective against kicks). A punch or kick "lands" if it is thrown while the fighters are adjacent and the opponent is not blocking the punch or kick. Each fighter also has a "life" level that is decreased each time the opponent lands a punch or kick. Fifteen total punches or kicks depletes a fighter's life to zero, and the first to have its life depleted is the loser.

Implementation. This was the most ambitious of the projects, and its implementation is considerably more involved than the previous two. Player input to control the fighter is done using a standard Super Nintendo Entertainment System (SNES) gamepad. The direction pad controls the fighter's motion (the up and down directions on the pad are not used). The *Start* button starts the game, the *Y* button punches, the *B* button kicks, the *X* button blocks high and the *A* button blocks low. The SNES gamepad uses a synchronous three-wire interface, which was connected to input port pins RC2-0 on the PIC. Details of the interface's operation were found by the students on a Nintendo-related web site. The two fighters are animated on the LCD screen using its graphics mode, and the images of the fighters in various positions are stored as bitmaps. There are thirteen separate bitmaps altogether, requiring a total of 5,973 bytes of data. This far exceeds what can be stored in the PIC's program memory, so external storage was required. The students used a Microchip 24LC65 8K$\times$8-bit serial EEPROM accessed via the PIC's I^2C interface. (The EEPROM on the PICDEM-2 board was simply replace by the pin-compatible 24LC65.) The computer-controlled fighter uses a simplistic, somewhat random algorithm for determining its actions as this was all that the available development time allowed. The game program requires 1446 of the 4096 available words of PIC program memory.

IV. Conclusion

The author considers the project as described herein to have been largely very successful. Three of four student groups successfully completed the project, and all were able, using the considerable information resources provided for them, to become adept in the operation and programming of both the PIC and LCD module. The course was evaluated very highly, indicating that the students did enjoy the opportunity to design their own video games.

The only serious problems in the course had to do with unanticipated difficulties in mechanically connecting to the LCD display (quite a lot of time and frustration went into the development of the cable adapter from the 1-mm-center LCD connector to the 0.1-in-center PICDEM-2 board) and supplying power to bias the LCD (there was considerable difficulty in debugging the MAX 766 application circuit for supplying this).

There will likely be a very similar project in the next edition of ECE 403 in which the above problems will be avoided. LCD modules have already been obtained (larger-geometry variants of the 128GS12-1) that have a standard 0.1-in-center 20-pin connector for easy cabling, and LCD

bias power modules will likely be pre-fabricated. This will allow students to focus on more sophisticated game designs, and sound (using a piezo buzzer or possibly more sophisticated sound chips) is likely to be a required feature of all games.

A relatively minor shortcoming of the project was that PIC resources were not as serious a limitation as had been intended. No program approached needing the full 4K program words, and the 192 bytes of data memory were at least adequate for all projects. This problem may be alleviated if more sophisticated games are attempted, or a bonus could be awarded to groups successful in using lower-end chips than the 16C74A. (These chips, which have smaller program and data memories and fewer I/O pins and on-board peripherals, are not supported by the PICDEM-2, but the bonus could be awarded to groups whose designs *could* use them.)

An additional consideration that will likely be brought into play is power consumption, as this is a battery-operated application. The LCD module is inherently very low power and the PIC has extensive support for low-power operation (e.g., sleep mode with “wake” on keystroke), so it ought to be possible to get very long battery life if the PIC is used properly.

Acknowledgements

The author wishes to acknowledge the contributions of the students in ECE 403, Summer 1998, whose projects have been described in this paper:

- Aaron Tiedje and Paul Fisher, designers of *Battleship*
- Eric Baron and Paul Morgan, designers of *Breakout*
- Jaremy Pyle and Jason Paskvan, designers of *Street Fighter*

Without their engineering abilities and determination in completing their projects, this paper obviously could not have been written.

Bibliography

1. Gene H. Miller. *Microcomputer Engineering*. Prentice-Hall, 2nd edition, 1999.
2. Thomas L. Harman. *The Motorola MC68332 Microcontroller: Product Design, Assembly Language Programming, and Interfacing*. Prentice-Hall, 1991.
3. John B. Peatman. *Design with PIC Microcontrollers*. Prentice-Hall, 1998.

JAMES S. MCDONALD

James S. McDonald is Associate Professor of Computer Engineering at Kettering University. He received a Ph.D. in Electrical and Computer Engineering from Rice University in 1992 and S.B.E.E. and S.M.E.E. degrees from the Massachusetts Institute of Technology in 1980. Dr. McDonald has seven years of industrial experience with Lawrence Livermore National Laboratory and General Electric Company Corporate Research and Development Center. He also has seven years of faculty experience during which he has focused on capstone design project, course, and laboratory development in various areas of electrical and computer engineering.