Migration from the MC68HC11 to the MC68HC12 within an Electrical & Computer Engineering Curriculum

Barry E. Mullins, Daniel J. Pack

Department of Electrical Engineering
United States Air Force Academy, CO

Abstract

For many educators, microcontrollers are becoming the pedagogical tool of choice for teaching fundamentals of microprocessor and microcontroller architectures and programming. The choice is mainly due to a variety of built-in functional components and easy input/output capabilities of microcontrollers. For several years, microcontrollers have been an integral part of the electrical and computer engineering curriculum at the U.S. Air Force Academy. In fact, fifty percent of our junior/senior level computer systems courses have used Motorola's 68HC11 exclusively either in conjunction with an evaluation board or as a single, stand-alone controller. We are currently in the process of migrating from the 68HC11 to the 68HC12 within our computer systems courses. This paper describes the rationale for the transition, the preparation steps required, the implementation issues we faced, the results of the transition as viewed by faculty members and cadets, and lessons learned from the experience.

Introduction

After the advent of the Motorola 68HC11 microcontroller in 1986, a large number of engineering educators in universities eagerly embraced and used the controller in digital systems courses. The chief reason behind the enthusiastic acceptance and the continuing use of the controller is due to a variety of built-in functional units such as I/O ports, timer units, and an analog-to-digital converter that allow the educators to easily teach students the fundamental knowledge on how a computer works while readily incorporating those units in homework and laboratory exercises to enhance student learning[2]. Thus, the use of built-in units (as opposed to building and assembling individual components) made it possible for educators to concentrate on central issues rather than solving problems associated with assembling functional parts. Reflecting the importance of teaching the fundamental computer knowledge, engineering accreditation organization ABET currently requires all accredited electrical and computer engineering programs to include a microcontroller/microprocessor course.

The two most popular microcontroller modules used in universities are the 68HC11EVB and the 68HC11EVBU. The first one is designed to work along with external memory components and additional ports (expanded mode) while the latter one was developed for embedded applications where a microcontroller contains all necessary resources within the controller (single chip mode). At the Air Force Academy, we have used the 68HC11EVB in two microcontroller/

microprocessor courses, which are required to be taken by most cadets majoring in electrical and computer engineering. The objective of the first course is to teach cadets (1) microcontroller-based assembly language programming skills, (2) knowledge of the functional units of a microcontroller, and (3) techniques to interface a microcontroller with external devices[3]. The 68HC11EVB is used for this course. During the second microcontroller/ microprocessor course, cadets learn how to program microcontrollers using high level programming languages and design and implement an embedded computer for a large project. The 68HC11EVBU has been the platform for this course.

Both modules have provided educators with necessary tools to fulfill microcontroller course objectives. If so, why change? The primary reason for switching from the 68HC11 to 68HC12 microcontroller is that Motorola is no longer producing both evaluation boards due to old technology-based components on the boards. Rather than creating new boards based on the 68HC11, Motorola is pushing universities to use the next generation microcontroller, the 68HC12.

The 68HC12 improved the 68HC11 system performance by incorporating an instruction queuing system, similar to a parallel-pipe instruction execution system found in most of high performance microprocessors, sophisticated mathematics operations, expanded timer functions, and fuzzy logic operations. By expanding the data bus to 16 bits from 8 bits (as found on the 68HC11) and using a higher speed clock, the 68HC12 readily meets the challenge presented by highly computational, time critical tasks. Furthermore, the paging scheme and expanded input and output ports allow one to implement small and large programs that access multiple I/O ports with ease.

Our decision to switch to the 68HC12 has affected a number of electrical and computer engineering courses. The two microcontroller/microprocessor courses had to be changed to take the advantages offered by the 68HC12. We have completed the transition in the first course and are currently in the process of switching the controller in the second course. We plan to present the feedback from the second course along with the ones from the first course at the upcoming conference. In addition, we will also receive feedback from faculty and cadets in two other spring 2002 electrical engineering major courses: Senior Design (EE 464) and Introduction to Robotic Systems (EE 387). Our focus of this paper is based on our experience in the first microcontroller/ microprocessor course. The rest of the paper is organized as follows. In the next section, we show the process of implementation followed by the outcomes of the implementation. We present lessons we learned as we administered the change. A few concluding remarks complete the paper.

Implementation

The course goals of our first microcontroller/microprocessor course, EE 382, include “Cadets shall develop the skills to design, implement, test, and debug microcontroller-based systems by developing operational assembly language programs that incorporate the built-in microcontroller functions, and by successfully interfacing the microcontroller to the external world.”[1] To this end, cadets in EE 382 learn the basics of computer architecture, microcontroller hardware,

assembly language programming, and interfacing external devices to the microcontroller's internal resources.

Portable Laboratory (Portalab)

We use two devices to teach these concepts—a 68HC12A4EVB (hereafter referred to as A4EVB) mounted within a portable laboratory (portalab) and a mobile robot. Figure 1 illustrates the layout and construction of both. The portalabs are issued to cadets at the beginning of the semester thus allowing them to work on assignments in their rooms or in the laboratory during evenings and weekends. Later in the semester the robots are also issued. The robots consist of two DC motors driving two 3.5" diameter wheels. The motors are mounted on an 8" diameter aluminum platter. Another platter of the same dimension is used to mount three infrared sensors. The remainder of this section discusses the modifications made to the portalab, the laboratory exercises, and course materials to migrate from the 68HC11 to the 68HC12.

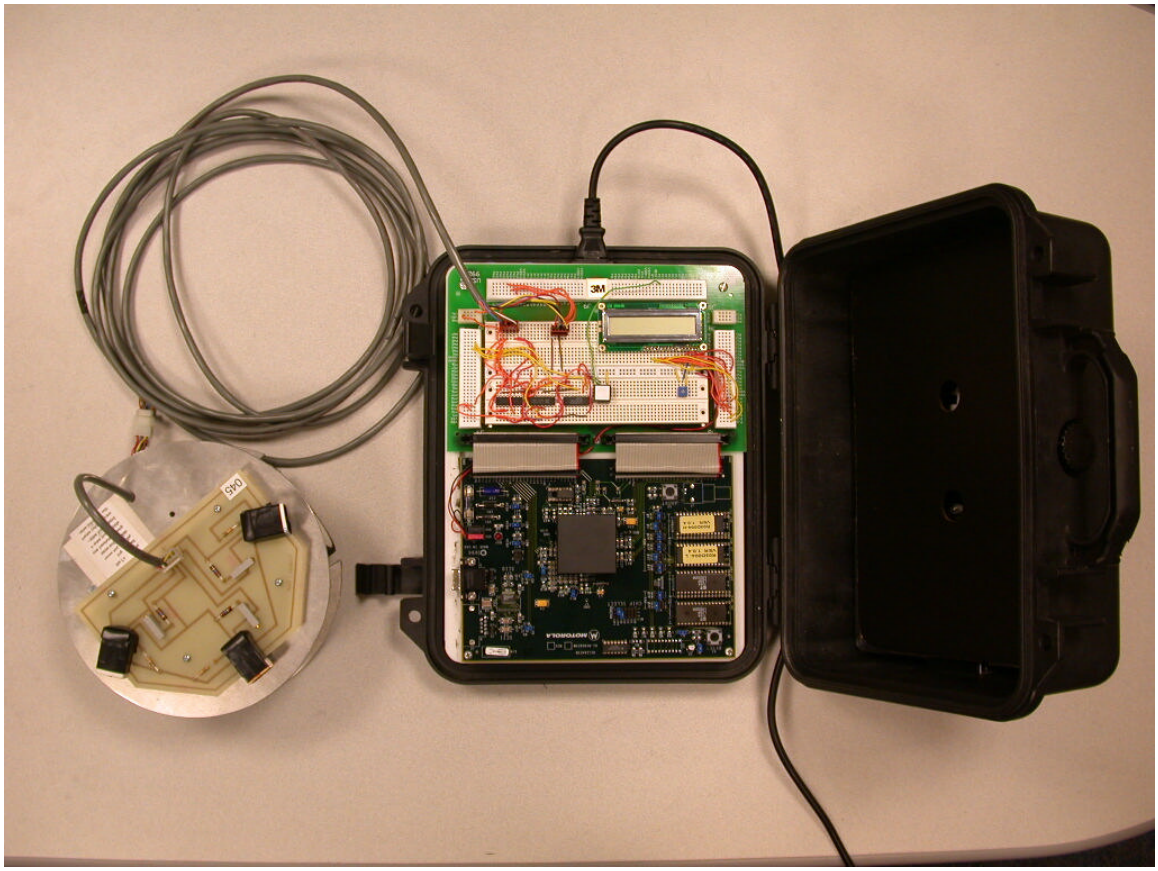

Figure 1. Portalab and Mobile Robot

As can be seen in Figure 1, the portalab is housed inside a hardened plastic case to endure everyday use of a college student. When we first designed and built the portalabs in 1995 for the 68HC11EVB, we constructed a platform box out of 0.25" plastic. This platform provided us a flat surface on which to mount the A4EVB board and a custom printed circuit board. Since we had to replace the 68HC11EVB board used in previous years with the A4EVB, we had to make

some design decisions. Since we already had the black cases, we decided to adapt the A4EVB to the case. This meant we had to cut off the A4EVB prototyping area which is well suited for wire-wrapped prototyping. Since we use breadboards in this course, the loss of this area was inconsequential. The custom printed circuit board transfers all available signals using ribbon cables from the A4EVB's J8 and J9 header connectors to a breadboarding area. Based on past experience, we designed the portalab such that the breadboard is fastened to the circuit board using Velcro. This allowed us to transfer a cadet's work between portalabs in the event an A4EVB is damaged. We also opted to replace the slower random access memory (RAM) integrated circuits supplied with the board (8K x 8, 200 ns) with integrated circuits recommended in the user's manual—IDT7164L25P (8K x 8, 25 ns)[4]. Replacing the RAM allowed the portalabs to operate with no E clock (system clock) stretching which simplified the explanation of timing analysis of delay loops. All RAM access times were the same regardless of the location of the RAM (on-chip vs. off-chip). We modified a total of 50 portalabs at a cost of $650 each; total upgrade cost was $32,500.

Labs

The labs within EE 382 are designed to build on one another, culminating in a robot maze competition at the end of the semester[3]. All cadets are exposed to most of the 68HC12's resources throughout the semester using nine labs that were designed to encourage software reuse and modular programming. All labs used assembly language exclusively. The nine labs are discussed here to highlight the concepts taught as well as the 68HC12 devices used. Additionally, the modifications required to each lab are presented as we changed from the 68HC11 to the 68HC12. Naturally, since the memory map for the A4EVB is different than the 68HC11EVB, adjustments were required when allocating memory in all labs.

Lab 1

The first lab is designed to introduce the cadets to the computing environment. This includes the 68HC12A4EVB, the terminal program that interfaces the EVB to the PC called WinIDE (Windows Integrated Development Environment created by P & E Microcomputer Systems), and the EPROM-resident monitor/communications program within the 68HC12 called D-Bug12. The lab guides the cadets through interfacing the portalab to the PC via a serial cable, establishing communication between the two using WinIDE (includes the CASM12Z assembler and communications software), and practicing with the commonly used D-Bug12 commands. The lab requires the cadets to copy a small assembly language program into WinIDE, assemble it, download the resulting S19 file to the portalab, and trace through the program. As they trace through the code, they are prompted for memory contents to verify the program is operating correctly. The 68HC11EVB monitor program (BUFFALO—Bit User Fast Friendly Aid to Logical Operations) implements some command instructions slightly different than D-Bug12.

The authors decided not to utilize the background debug mode (BDM) capabilities of the HC12 although it was definitely seen as a valuable debugging tool. The primary reason for not using BDM involved finances and logistics. We did not have the necessary funds to purchase the requisite hardware and cables, and we did not have time to incorporate the BDM into our labs.

This lab required minor rewrites of the lab handout and the sample program since the CPU core was the only resource used.

Lab 2

This lab requires the cadets to write their first complete assembly language program using the basic instruction set and assembler directives. The lab helps them practice using branch instructions in "if-then-else" situations. The resulting program is used as a subroutine in their final lab (robot maze competition) to make navigational decisions for their robot based on infrared sensor values. This lab emphasizes structured programming and memory management.

Since the CPU core is the only resource used, no modifications to the lab were required. However, since the 68HC12 instruction set is more robust than the 68HC11, the resulting programs were more readable and easier to maintain and to adapt for future applications.

Lab 3

The third lab is designed to exercise programming skills using subroutines to promote software reuse and modularity. At this point, we insist on structured programming. Cadets are required to pass data to subroutines using the call-by-value and call-by-reference techniques. The lab reads a velocity value and a time value from memory, calculates a trapezoidal speed profile (to accelerate and decelerate linearly) based on the two read values, and stores the speed profile table into memory. The techniques learned creating the speed profile are used in labs 6 through 9 to control the speed of the robot's two motors.

Although modifications were not absolutely necessary, the added index addressing modes of the 68HC12 made the speed profile calculations and saving tables to memory more straightforward.

Lab 4

Introducing cadets to interfacing components to the A4EVB was the objective of lab 4. They connected a simple pushbutton and a 16 character LCD to ports on the HC12 and wrote a program to interface the devices with the HC12. The program polled the pushbutton, debounced the button using software, printed messages to the WinIDE terminal window on the PC, called various subroutines written during lab 3 and displayed a short message to the LCD. This lab required the cadets to completely understand the programming and operation of HC12 ports, create precise delay loops by calculating delay times for various instructions, satisfy timing requirements for external devices (e.g., LCD), and utilize D-Bug12 routines.

This lab required four modifications. First, a different means of pulsing the enable (E) line on the LCD was required. The LCD requires data be placed on its data lines followed by a positive pulse on its E line with a pulse width of at least 450 ns. The 68HC11-implementation used the STRB line (1000 ns pulse width) to pulse E. The 68HC11 automatically generated this pulse after writing a byte to port B (attached to LCD data lines). Since the 68HC12 doesn't have strobe lines, the cadets were required to implement a strobe using a bit on one of the output ports. This required the cadets to consider the timing of data and signal arrival at the LCD,

which forced them to design their software appropriately (it is possible to pulse E but still violate the pulse width specification of the LCD). The cadets had to carefully calculate the time it took to bring a single output port bit from low to high and back to low.

Second, instead of feeding the pushbutton input into the STRA pin on the HC11, we fed the input into a bit on an input port on the HC12. This was necessary since the HC12 doesn't have an input strobe. Therefore, the input bit (attached to the switch) is polled instead of the STAF bit in the PIOC register (HC11 implementation).

Third, the LCD display required several delay periods to operate properly. These delays were implemented using simple delay loops. All delay loop values used for the HC11 had to recalculated using an 8 MHz clock versus a 2 MHz clock; the HC12 instruction queue also had to be factored into the delay calculations.

Fourth, lab 4 required extensive use of D-Bug12 routines to print messages and data values to the WinIDE terminal window. The 68HC11 used BUFFALO routines. Although minor modifications to the code were necessary, it was critical for the cadets to understand the mechanism by which the data values are passed to the called routines.

Lab 5

This lab introduced the cadets to the interrupt subsystem on the 68HC12. The objective of the lab is to program the 68HC12 to accept an external interrupt request via a pushbutton wired to the IRQ line and display appropriate messages to the WinIDE terminal window and a LCD display using the 68HC12 I/O ports. The lab requirements stated that the interrupt must occur within the first three seconds of the program's execution. The external interrupt capability can be used as an emergency stop button in Lab 9, and the LCD display interface can be used to communicate the robot's status.

This lab required three modifications. First, instead of connecting the switch to STRA (configured to generate an interrupt) on the HC11, the switch is connected to IRQ on the HC12. Functionally, both configurations yield the same result—an external interrupt.

Second, the method by which the interrupt vector table was updated was also modified. Instead of storing a jump instruction followed by the address of the interrupt service routine (ISR) in the vector table, the actual ISR address was loaded (without the jump instruction). This requirement is imposed by D-Bug12.

Third, once again the values used by the delay loop had to be recalculated.

Lab 6

This lab was designed to provide the cadets with experience using the input-capture and output-compare features of the 68HC12. Specifically, they programmed the 68HC12 to generate pulse-width-modulated (PWM) waveforms to control their robot's motor speed. This lab only requires cadets to program forward motion of the robot.

Due to an enhanced input-capture (IC) / output-compare (OC) subsystem, several modifications were necessary. First, the timer system had to be enabled by writing a one to the TEN bit in the TSCR register.

Second, since the pins of port T are configurable to act as either an IC or OC, the cadets had to specify which pins were IC and OC by setting the corresponding bits in the TIOS register.

Third, using the default timer prescaler values in the TMSK2 register, the HC12 created a PWM waveform with a period of 8.192 ms. Using the same default settings, the HC11 generated a PWM waveform with a period of 32.768 ms. Although the HC12 was capable of generating a much shorter period, the DC motors on the robot did not perform as well with this shorter period. The cadets had to increase the PWM period to 32.768 ms.

Lab 7

This lab completed the entire range of motion for the robot. Additional hardware was added to the system along with software modifications from lab 6 to make the robot go backward, turn right, and turn left in addition to the forward motion mastered in lab 6.

No further modifications were required to this lab.

Lab 8

This lab was designed to introduce the cadets to the built-in 68HC12 Analog-to-Digital (ATD) converter. A set of three infrared (IR) emitter and detector pairs on their robots (as seen in Figure 1) were used to generate analog voltages proportional to the distance between the robot and an obstacle. The cadets programmed the 68HC12 to interpret these incoming voltages to determine whether a mobile robot was approaching a wall in front or on one of its sides.

The HC12 built in more versatility in the ATD conversion subsystem. As such, an additional register must be addressed. The ATDCTL4 register essentially controls the conversion rate. The default settings for ATDCTL4 were used.

Lab 9

This lab requires the cadets to pull together everything they have learned during the course and program their robots to navigate through a maze. On the last day of classes, each section held a competition to see who could navigate the maze in the shortest amount of time without colliding into the maze walls.

Since this lab was a culmination of all previous labs, no further modifications were required to migrate from the HC11 to the HC12.

Administrative Changes

A small change to a course requires small adjustments to the course syllabus, instructor notes, presentation material, etc. Changing microcontrollers is not a small change. The course director put in countless hours selecting a new text (*Embedded Microcontrollers*, ©2001, Prentice-Hall, by Todd Morton); modifying the syllabus to incorporate the text change; rewriting the lesson plans, presentation material as well as the assignments (labs); and reaccomplishing the labs using the new portalabs and programming environment. The instructors had to reaccomplish the labs.

Outcomes

Changing the content of a course should not be taken lightly. The consequences need to be considered not only from the initial added workload on the instructors, but also from the educational payoffs for the cadets. From the instructor's perspective, the change from the 68HC11 to the 68HC12 was well worth the investment in time and money. The 68HC12's instruction set, with its enhanced indexed addressing modes, provided excellent means to write modular, compact code. The increased clock speed increased the resolution of timing projects. The amount of time required to complete the labs did not increase with the change.

From the cadet's perspective, the change from the 68HC11 to the 68HC12 was seamless. Since none of the cadets had used the 68HC11, they did not have a baseline to compare the 68HC12 against. However, when the instructors demonstrated 68HC11 projects to the cadets and the DOS-based programming environment used with the 68HC11, they were appreciative of WinIDE and the power of the 68HC12. Overall, the cadets enjoyed the course as in previous years.

Conclusions

In this paper we described the 68HC11 to the 68HC12 migration process that took place at the U.S. Air Force Academy during the fall 2001 semester. We presented the rationale for the change, critical implementation issues, required administrative changes, and the outcomes of the change. The migration from the 68HC11 to the 68HC12 has gone smoothly albeit with a lot of work! The migration of the labs and course material took the bulk of our efforts. Some thought was required to ensure the pedagogical outcomes of the labs were not altered from the 68HC11 implementation to the 68HC12. We are currently migrating the follow-on course to the 68HC12 as well. This course, Microcomputer Systems Design I, will use the 68HC12 in the expanded stand-alone mode. The cadets will use the MC68HC912B32 as the foundation to build a small computer system kernel consisting of RAM, ROM, and I/O. They will program their systems using C to perform various functions (mainly to test their system hardware). We plan to report our findings from the follow-on course at the upcoming conference.

References

1. EE 382 Course Assessment Plan Version 3, United States Air Force Academy CO, 10 December 2001.

2. Daniel Pack and Steven Barrett, "The 68HC12 Microcontroller: Theory and Applications," Prentice Hall, New York, 2001.

3. D.J. Pack, A.R. Klayton, A.L. Clark, and J.P. Trudeau, "Incorporating Mobile Robots in an EE Microcomputer Programming Course," *Proceedings of the 1998 American Society for Engineering Educators Conference*, Seattle, Washington, June 1998.

4. Motorola, M68HC12A4EVB Evaluation Board User's Manual—Rev 1, M68HC12A4EVBUM/D, October 1999.

BARRY E. MULLINS is an Assistant Professor and Chief of the Computer Systems Division within the Department of Electrical Engineering at the U.S. Air Force Academy. He received the B.S. in Computer Engineering from the University of Evansville, the M.S. in Computer Engineering from the Air Force Institute of Technology, and the Ph.D. from Virginia Tech. Lt Colonel Mullins is a member of ASEE, IEEE (Senior Member), Tau Beta Pi, and Eta Kappa Nu. His research interests include mobile robots, microprocessor/microcontroller-based systems and applications, and wireless local area networks. Email: barry.mullins@usafa.edu

Daniel Pack is Professor of Electrical Engineering at the United Air Force Academy, CO. He received the Bachelor of Science degree in Electrical Engineering in 1988, the Master of Science degree in Engineering Sciences in 1990, and the Ph.D. degree in Electrical Engineering in 1995 from Arizona State University, Harvard University, and Purdue University, respectively. He is a member of Eta Kappa Nu, Tau Beta Pi, IEEE, and ASEE. His research interests include intelligent control, automatic target recognition, robot vision, and walking robots.