Designing a Paintball Gun Chronograph in a Computer Systems Course Incorporating Intel Microprocessors and PICmicro® Microcontrollers

William Dillard
200 Broun Hall
ECE Department
Auburn University, AL 36849
Voice: (334) 844-1840
Fax: (334) 844-1809
dillard@eng.auburn.edu

Abstract

Most ECE departments teach computer systems using microcontrollers with microtrainers systems. This approach has two deficiencies. First, students must be in the laboratory to debug real-time code and physically connect peripherals, leaving little opportunity for extracurricular experimentation. Second, treating only the microcontroller distances the student from the ubiquitous PC and its standards.

A new approach to teaching computer systems and assembly language for sophomore electrical engineering students is being investigated at Auburn University. Due to curriculum restrictions, the sophomore level course has no formal hardware laboratory. From the outset, four issues were addressed: treating PC-related issues via the 8086 microprocessor (our traditional approach), introducing embedded systems with simple microcontrollers, including a project to add a meaningful hardware experience and providing a means for students to inexpensively program their MCU's at home.

We selected the PIC12F675 microcontroller and the PICkit™ 1 FLASH Start Kit development board from Microchip Technology, Inc. for our microcontroller studies. Teams of students construct, code, debug and test complete design solutions at home and verify their implementation by real-time execution in class. The paintball chronograph project requires hardware and coding for both the PC and PIC12F675, focusing most of the pertinent course material into a single effort.

Course assessments show that the chronograph project was very successful and highly motivational. Hardware construction was relatively simple and easy to debug. Conducting field tests with "live ammo" in front of the entire class provided both excitement and extra motivation for the work. Also, the concepts of serial protocols and the PC serial port operation, particularly the importance of transfer timing and MCU clocking, were well appreciated.

Microcontrollers vs. Microprocessors

Currently, most ECE departments teach computer systems using microcontrollers with microtrainers systems in formal laboratory settings[1 - 4]. Compared to a course based entirely on a microprocessor such as the 8086, this approach has obvious advantages for electrical engineering students. A generic list includes:

1. On-chip memory eliminates the need to interface RAM and ROM to data/address buses or decoding hardware.
2. A rich assortment of on-chip peripherals such as ADC, comparators, timers, serial bus interfaces (UART, CAN, I^2C, etc.) PWM and DAC are available. There is no need for bus interfacing, I/O decoding or for interrupt number transfers.
3. Clocking a microcontroller requires at most a crystal and a few capacitors rather than a clock generator chip. Many microcontrollers have an internal clocking option that eliminates the crystal altogether.
4. A smaller instruction set requires fewer lectures on coding and allows more time for system-level concerns.
5. Since the microcontroller is a complete computing system, all major CPU concepts can be covered.
6. The amount of lab time needed to construct an embedded system design can be much shorter since bus interfacing to peripherals may not be necessary.
7. Given the popularity of microcontrollers in commercial embedded systems, a proper education should include exposure to microcontrollers.

All of these features make embedded systems via the microcontroller more attractive to the student. Such a course/laboratory structure, excluding a treatment of microprocessors (particularly the Intel microprocessors and the PC) and holding laboratories at a fixed location, does have deficiencies.

1. Avoiding the Intel processor family distances students from their most familiar point of reference for computing, the ubiquitous PC and its standards.
2. Students must be in the laboratory to debug real-time code, leaving little opportunity for extracurricular experimentation.
3. Unlike PC's, microcontrollers are not designed specifically to interface to keyboards and monitors. As a result, students have no exposure to DOS and BIOS function calls for keyboard, monitor and port manipulations.

Course Evolution at Auburn University

At Auburn University, our undergraduate computer systems course was based entirely on the 8086 processor with a dedicated laboratory where interfacing and coding were verified on custom microtrainers. In 1999, the laboratory structure was changed significantly[5]. Course specific laboratories were replaced with four standalone labs – two in the sophomore year and two in the junior year – none of which included any exposure to the 8086 hardware. Coding migrated from the microtrainers to Pentium based PC's where the only supported

hardware was DOS and BIOS function calls to the keyboard and monitor. Although this allowed students to write, debug and verify code in open computer facilities and at home, there was no hardware content that could be consider an "embedded system". Furthermore, an introduction to microcontrollers was available only as an elective.

In spring semester 2002, a new course structure was attempted that added interface experiments to the microprocessor content using the parallel port. These experiments were conducted by teams of 4 to 6 students as homeworks, constructed on breadboards at home and verified in class. A similar approach was used successfully to add hardware content to a digital electronics service course for computer science majors at Auburn University[6]. Although analog-to-digital converters could have been accessed through the PC ports, we felt this approach was too far removed from a true microprocessor-based embedded system approach and limited all experiments to purely digital designs. While the new course reclaimed some hardware exposure, we were still far from an "embedded system" treatment.

To rectify the situation, the course was again modified in spring semester 2003 to retain the microprocessor content but include microcontroller architecture, coding and projects. Time constraints meant that the microcontroller had to be relatively simple with a small instruction set and easy to debug and program. The PIC16F72 microcontroller and MPLAB assembler/debugger from Microchip Technology, Inc. were selected for this task[7]. The salient features of each product are listed in Table 1. Borrowing from an industrial application, the spring 2003 semester project was a fan speed monitor with alarm. The PICmicro® MCU monitored the speeds of three fans and transmitted the data to the PC via its serial port. 8086-compatible coding displayed the speeds and alarms on the monitor screen. The major difficulty was providing PIC programmers to the student teams. The least expensive solution was the PICSTART® Plus Programmer at $200 each. At the end of the semester, each team owned a PIC16F72, fans and assorted IC's, but they did not own a programmer. Since one of our goals was to leave the students with the capacity to pursue their own embedded system projects, an alternative to the PICSTART® Plus was needed.

Table 1. Key Features of the PIC16F72

Core Architecture	*Peripherals*
8-bit data bus	Three timers
Two 8-bit and one 6-bit I/O port	8-bit SA-ADC
Harvard bus structure	PWM module
Orthogonal RAM	I^2C and SSP serial interfaces
Direct, indirect and relative addressing	Dc – 20 MHz operation
35 instructions	No internal clock option

During the summer of 2003, Microchip released the PICkit™ 1 FLASH Start Kit programming and evaluation board, shown in Figure 1. Although the product can program only 4 different microcontollers, a price of only $36.00 made it an adequate replacement for the PICSTART® Plus. In fall semester 2003, we migrated to the PICkit™ and the 8-pin PIC12F675 microcontroller. (Table 2 contains the critical features of each product.) With the PICkit™ 1 FLASH Start Kit, coding could be done at home or in open laboratories in the MPLAB environment and programming via an USB port to the PICkit™ board. A collection

of LED's, switches with potentiometers facilitated simple experiments and the PICkit™ User's Guide provided programming details and seven coding tutorials. An important feature of the PICkit™ was the unpopulated serial port communication snapoff board on the right side in Figure 1, which, after population, was used to communicate with a PC. Although the PIC12F675 lacked serial communication modules, coding for a simple half-duplex UART was not difficult. Armed with their own PICkit™ 1 FLASH Start Kit, each team OWNED a programmer and could execute all phases of their design at home.

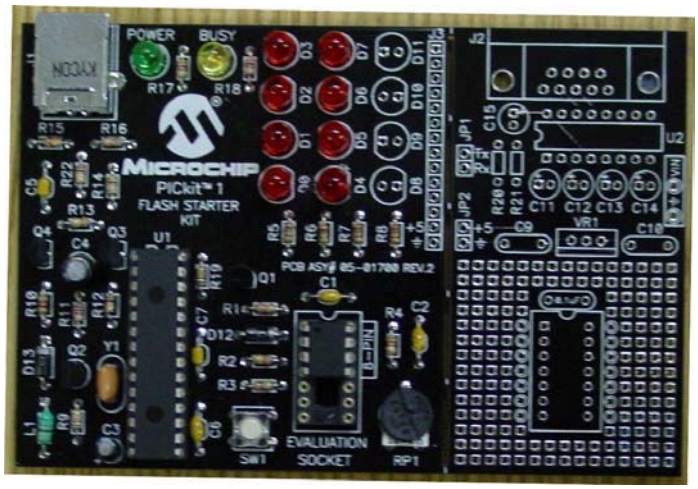


Figure 1. The PICKIT programmer/evaluation board from Microchip Technology, Inc.. The right side of the board is the unpopulated serial communication circuit.

Table 2. Features of the PICkit™ 1 FLASH Start Kit and the PIC12F675 Microcontroller

PICKIT	*PIC12F675*
Programs 4 different PIC MCU's	Internal 4 MHz clock
Compatible with MPLAB	8 pins
Self powered from USB	10-bit ADC
Unpopulated serial comm. circuit	35 instructions
Mutiplexed LED's	Configurable comparator
Pushbutton	EEPROM
Potentiometer	One 8-bit timer
External access to all μC pins	One 16-bit timer
User's guide with 7 coding tutorials	6 programmable I/O

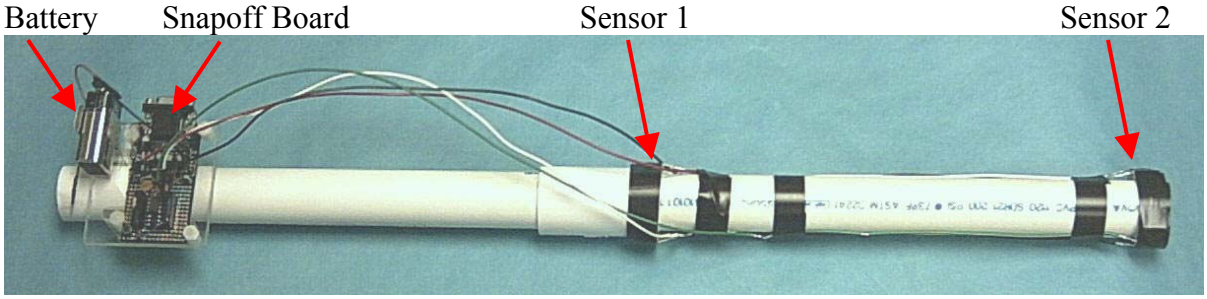
The Paintball Chronograph Project

For fall semester 2003, a paintball chronograph was chosen for the semester project. A chronograph measures the muzzle velocity of the paintball, usually between 100 and 500 fps, as it exits the barrel of the marker (paintball guns are called markers). Unlike rifles, marker muzzle velocities are adjustable for safe play in close quarters. Unfortunately, the adjustment mechanism on most markers is an uncalibrated screw. Using a chronograph, a player can calibrate his marker more precisely, gaining an edge in competition.

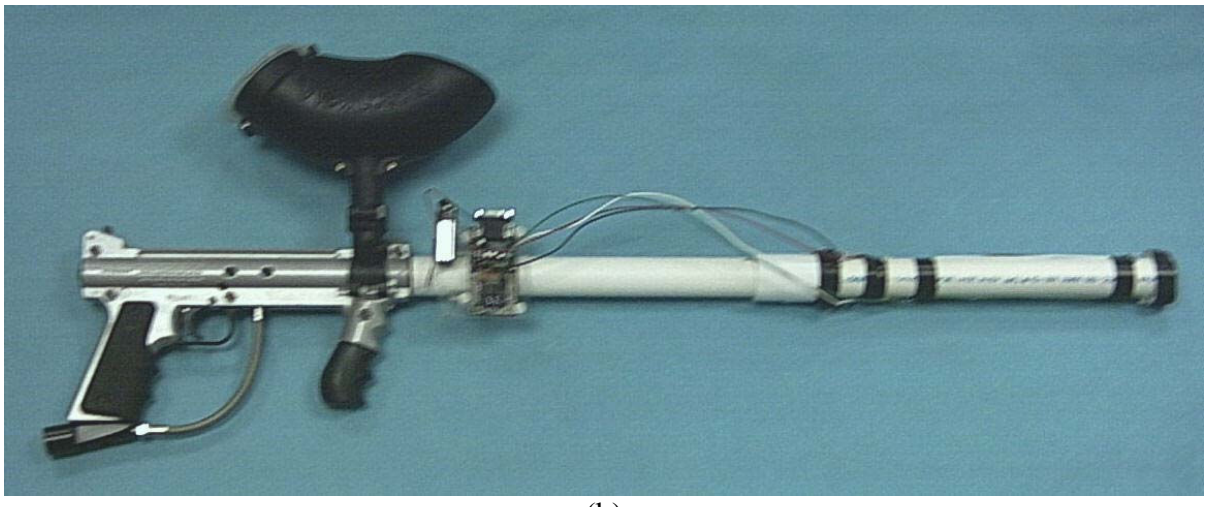
To incorporate 8086-compatible software and the serial port, the specifications of the chronograph project were expanded beyond the standard commercial versions that simply display the most recent velocity on an LCD or LED screen. Our chronograph system was designed to allow multiple firings at a single velocity setting for statistical analysis and, if desired, the results could be saved to file on a PC.

1. Using IR optics, detect the paintball as it passes two points one foot apart.
2. Measure the time elapsed in traversing said distance.
3. Convert elapsed time to velocity.
4. Display velocity and updated average on a PC monitor in a table format.
5. When the ESCAPE key is pressed, save the screen contents to a user-specified file.

To satisfy specification 1, two matched photodiode/phototransistor pairs were attached to a PVC pipe. Each team populated their serial communication board, securing it to a Plexiglass platform affixed to the pipe. The entire chronograph was then mounted to the marker barrel. Figure 2 shows the marker/chronograph assembly. Most chronographs are not barrel mounted, resting instead on a table, and have no walls. We chose the barrel-mounted option to improve the reliability of the optics regardless of the ambient conditions.



(a)



(b)

Figure 2. (a) The chronograph tube showing the snapoff serial communications board with battery. (b) The marker with chronograph attached and read for use.

The PIC12F675 was used to measure the elapsed time between the paintball passing the optical sensor pairs. Three critical hardware features were used: the internal 4 MHz clock, the 16-bit timer and the comparator. Since the PIC12F675 has a four clock instruction cycle, the timer count is exactly the elapsed time in microseconds. For muzzle velocities between 100 and 500 fps, the elapsed times in microseconds and so the timer count will be 10,000 to 2000, respectively. These values fit easily in the 16-bit timer with a worst-case velocity resolution of 3 inches/second. Also, the 16-bit timer can be enabled/disabled in one instruction cycle. To provide more consistent sensing of the paintball, the outputs of the optics were input to the on-chip comparator whose reference was set internally at $V_{DD}/2$. In this way, paintball detection is not a function of the logic high threshold voltage of a digital input pin. During initialization of the microcontroller, the comparator was configured as shown in Figure 3. After the first sensor detects the paintball, the analog switch is "thrown" via software to the second sensor.

After the paintball passes both sensors, the 16-bit data is passed to the PC serial port by the serial communication board. This requires two 8-bit transfers. Serial port reads are done via BIOS function calls. Converting elapsed times to muzzle velocities, calculating average velocities, updating the table on the monitor and saving the file to disk are done by the PC in 8086-compatible code.

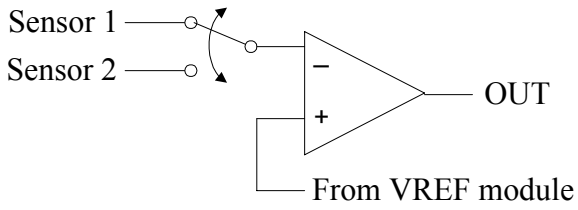


Figure 3. The comparator in the PIC12F675 is configured during initialization to accept one of two possible inputs. The analog switch position is controllable in real-time.

Project Details

Flowcharts for the microcontroller and the 8086 code are shown in Figure 4. A pushbutton has been added to the snapoff board for resetting the microcontroller. This button, pressed before each shot is fired, resets and initializes the MCU. A complete circuit diagram for the chronograph is shown in Figure 5. A baud rate of 9600 bps was selected for the project. This is the maximum 8086-compatible rate supported by BIOS function calls and corresponds to a bit cell duration of 104 μs. The precision of the 4 MHz internal oscillator is sufficient for transfers containing 8-bit data with one start and one stop bit. Since the PIC code is linear with few conditional jumps, the comparator output is polled directly rather than used as an interrupt source. This approach reduces the latency inherent in entering and exiting the interrupt service routine. Finally, an example of the on-screen firing table is shown in Figure 6.

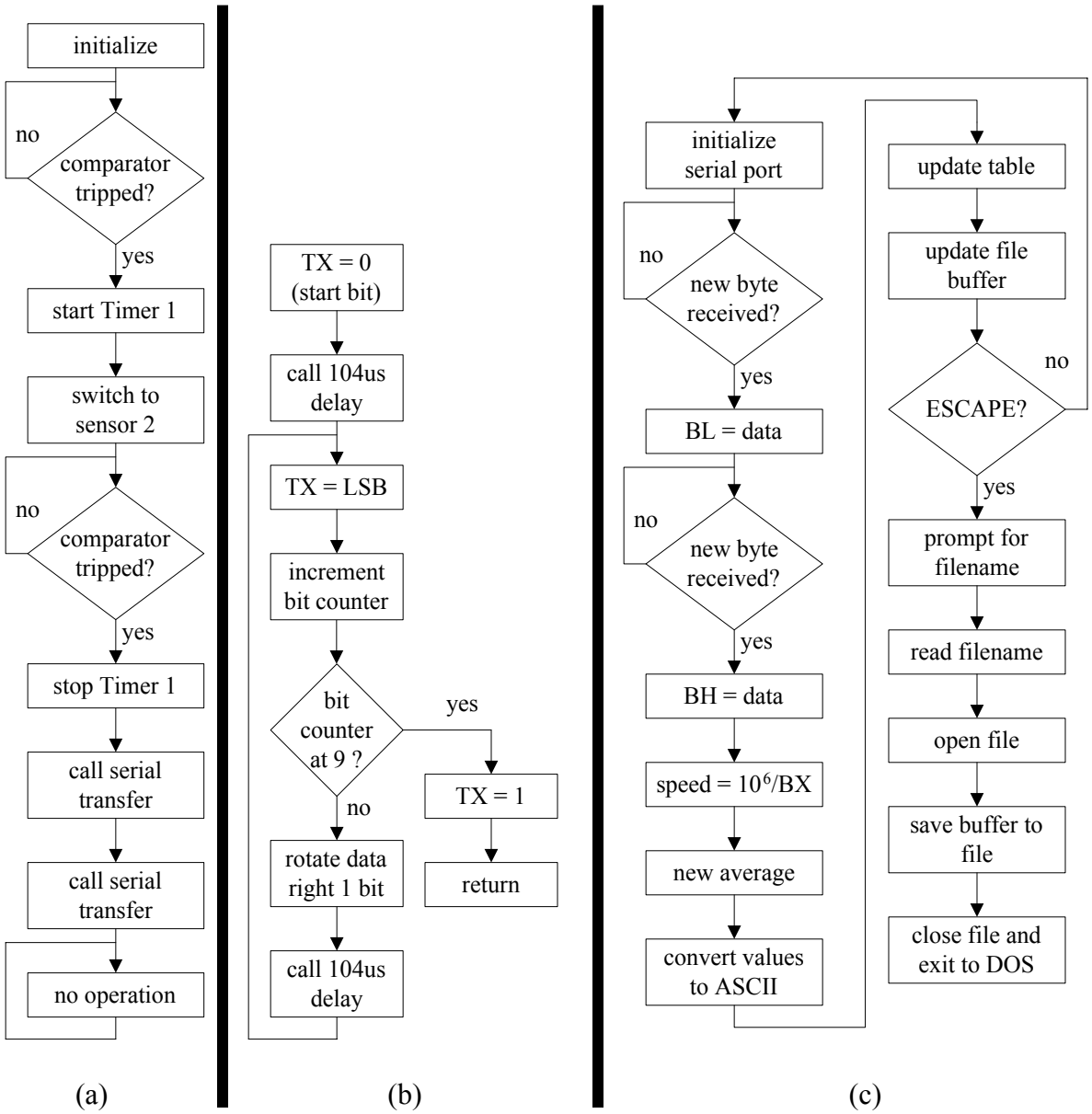


Figure 4. Flowcharts for the software components of the project. (a) Main microcontroller code monitors sensors, determines time of flight and calls a procedure to serially download data to PC. (b) A serial communication procedure outputs the start bit, then 8 bits of data in a shift/delay loop and finally the stop bit. (c) The PC code monitors the serial port for newly read data, saves it, calculates the muzzle velocity, updates the average velocity and displays the results on the screen. Finally, when prompted by the ESCAPE key, the table is saved in a user-specified file.

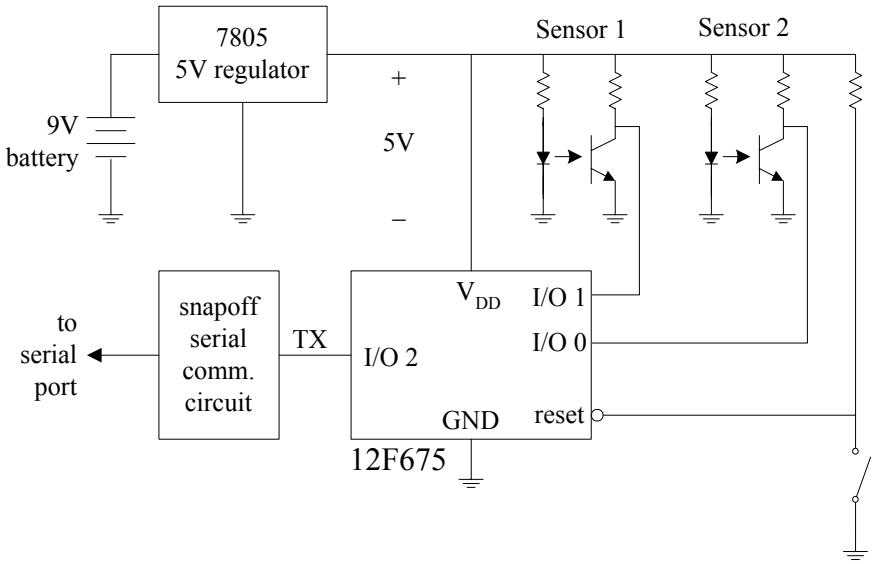


Figure 5. The complete circuit diagram including the reset pushbutton. All optics and associated resistors are mounted on the chronograph. All other components are on the snapoff serial communications board. Parts for the snapoff board are detailed in the PICkit™ User's Guide documentation.

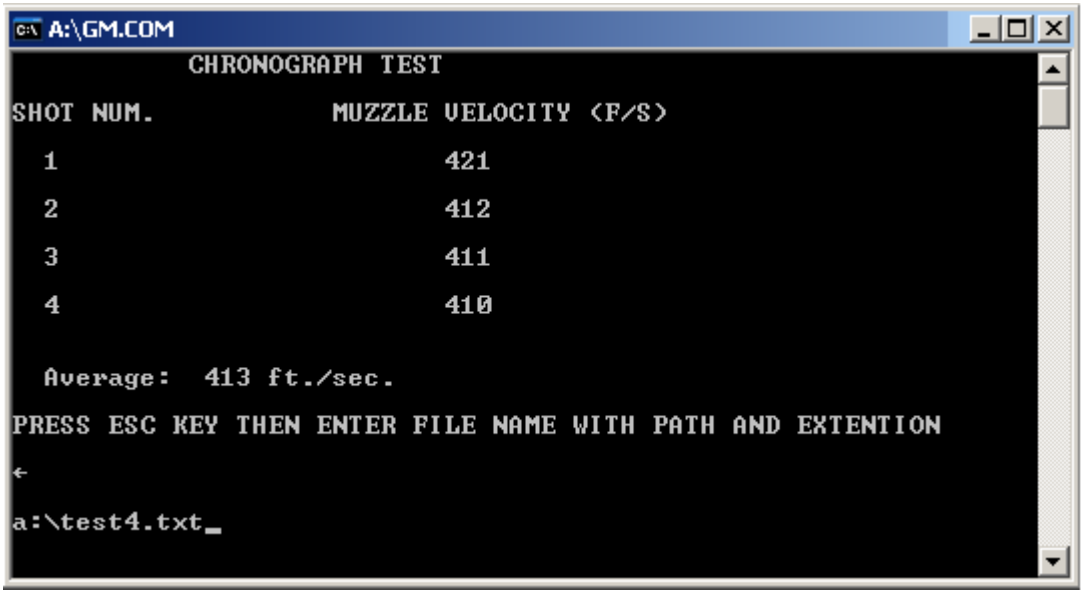


Figure 6. An example of a firing table showing the shot numbers, corresponding velocities and the current average velocity. When the ESCAPE key is pressed, the table will be saved to a user-defined file on the A: drive.

Verifications and Field Tests

A vital part of any undergraduate project is creating meaningful yet simple verification procedures for each major component of the work. Here, six verifications were used during the life of the project.

1. The chronograph was constructed with sensor outputs driving LEDs on the PICkit™ board and the microcontroller programmed as a simple buffer. Students used this circuitry to verify the performance of the optical circuitry by rolling a paintball through the chronograph tube.
2. The complete microcontroller code was used with the timer prescaled by a factor of 64 and the ball was rolled down the tube. The 16-bit timer content was displayed on the PICkit™ LED's 8-bits at a time at one second intervals, verifying the comparator and timer performance.
3. Dummy data was used to verify 8086 code for screen manipulations and file storage.
4. A serial cable loop termination was provided to allow students to test serial port/ cable hardware and BIOS functions calls to initialize, write and read the serial port.
5. The chronograph was connected to the PC serial port and the entire system was tested with a rolling paintball.
6. The timer prescale was removed, the chronograph was mounted to the marker, and the live field tests conducted.

Assessments

During the semester, two formal assessments were conducted as surveys. The first survey, administered early in the semester, targeted the students' preparation for the modified course structure, particularly their previous exposure to computer hardware. From the survey statistics listed in Table 3, students were prepared for the course with the vast majority having PC's at home and some confidence in circuit construction.

Essay questions were included to gauge student opinions on the importance of computer systems and particularly embedded systems in their future careers. All students felt they should have at least an introduction to the fundamentals of embedded systems and half felt their education should include details of theory and implementations. A full 2/3 of the

Table 3. Selected Statistics from Survey 1.

Question	*Mean Response*
Do you own a PC?	yes = 93%
Experience with assembly language?	1.4 out of 5
Experience with high-level languages?	2.6 / 5
Confident in constructing circuits?	3.6 / 5
Confident in designing your own circuits	2.4 / 5
Have you installed PC hardware?	yes = 93%
Written code for serial/parallel port?	yes = 6%

students were excited about the prospect of a hands-on approach to both the microcontroller and PC in a functional project. Specifically, they felt that the course structure would benefit them significantly when interviewing for employment.

At the end of the semester, the second survey was completed. The quantifiable statistics in Table 4 show that while the embedded systems approach was not overwhelmingly endorsed, it did find favor. Of particular interest are the responses to the third question, which suggest that owning a PICkit™ programmer and stressing physical design content produced better motivated students.

These results are in agreement with the essay responses on survey 2 where 80% of the students expressed a keen interest in design and construction of systems as the primary reason for choosing electrical engineering. The remaining students were mostly drawn to careers in management. Not surprising, 40% expressed a dislike for coding in general. Opinions on the effectiveness of utilizing the PICkit™ programmers ranged from highly enthusiastic to mild disinterest. However, these opinions were highly correlated to the individual students interest in programming and embedded systems design. Interestingly, since the PICkit™ programmers were introduced, two senior design groups are using the programmers to develop subsystems in their capstone projects.

Table 4. Selected Statistics from Survey 2

Question	*Response*
I have a clear idea of the skills I must master to have the career I want.	3.7 / 5
I know which classes in my curriculum will provide those skills.	3.8 / 5
I prefer working with the PICkit™ to a tradition course structure.	4.3 / 5
Owning a PICkit™, I can envision projects I would like to pursue.	3.4 / 5
Creating a functional prototype changes my perspective from academic to career.	3.3 / 5

Conclusions

A method for teaching computer systems for both the Intel 8086 microprocessor and the microcontrollers (the PIC12F675) with a comprehensive project has been discussed. The course does not require a dedicated laboratory or microtrainers and leans heavily on the new PICkit™ 1 FLASH Start Kit programmer/evaluation board from Microchip Technology, Inc. to free student teams from a fixed laboratory site, allowing them to code, debug, program and verify hardware at home. By treating both the 8086 and the PIC12F675, significant projects can be conducted with the MCU connected to the serial and parallel ports, replacing traditional timers, ADC and UART's normally interfaced on the 8086 data and address bus.

One goal of the course structure is to encourage lifelong learning by leaving students with the resources to implement their own designs. One such resource is a complete embedded system kit. The PICkti™ meets this goal at a reasonable cost of $36.00 per team. A second "resource" example of this occurred during the population of the snapoff board when students were taught proper soldering techniques. The majority of the students had never

used a soldering iron and they were very excited to master such a simple, low-tech task. They are hungry for the very skills that are directly applicable to extracurricular projects and lifelong learning.

By stressing a well-conceived verification schedule, every team assembled a working chronograph with very little hardware rework. Some teams had coding difficulties in the screen and file coding, but all hardware and MCU coding was functional. In retrospect, we recommend that instructors using our course structure be very careful to create projects that can be verified without advanced diagnostic equipment such as oscilloscopes and logic analyzers. Also, we suggest adding custom tutorials to those in the PICkit™ software as your assignments require.

Acknowledgements

I must mention Mr. Nathan Loden, a graduate student at Auburn University, who suggested the chronograph as a suitable project and provided the paintball marker used in our field tests. Also, my sincere thanks to each of my students enrolled in ELEC 2220 during the spring, summer and fall semesters of 2003. Your enthusiasm, hard work and constructive feedback have made this document and the course advancements reported herein possible.

References

[1] Carroll, C., Rocio, A., Fernando R., 1999, "New life for the MC68HC11 evaluation board," *2001ASEE Annual Conference Proceedings,* Montreal: American Society Engineering Education.
[2] Beenfeldt, D., Beenfelt, E., Field, J., Williams, E., "An ECE Freshman Microcontroller Course at the University Of Maine," *2001ASEE Annual Conference Proceedings,* Montreal: American Society Engineering Education.
[3] Neilsen, M., Lenhart, D., Mizuno, M., Singh, G., Zhang, N., Gross, A., "An interdisciplinary curriculum on real-time embedded systems," *2001ASEE Annual Conference Proceedings,* Montreal: American Society Engineering Education.
[4] Jeon, J., "A microprocessor course: designing and implementing personal computers," IEEE Transactions on Education, vol. 43, no. 4, pp. 426 – 433, November 2000.
[5] Roppel, T.A.; Hung, J.Y.; Wentworth, S.W.; Hodel, A.S., "An interdisciplinary laboratory sequence in electrical and computer engineering: curriculum design and assessment results," IEEE Transactions on Education, vol. 43, no. 2, pp. 143 – 152, May 2000.
[6] Dillard, W.C., 2001, "Effectively incorporating a hardware experience into a digital electronics service course," *2001 ASEE Annual Conference Proceedings*, Montreal: American Society Engineering Education.
[7] Microchip Technology, Inc., Chandler, Arizona, website at http://www.microchip.com

William C. Dillard -
William Dillard is a Ph.D. candidate in the Electrical and Computer Engineering Department at Auburn University. He holds M.S. and B.S.E.E degrees from Auburn University. His current educational interests are teaching strategies that promote professionalism and career development in students. Technical interests are in the area of power electronic systems and devices.