A non-virtual distance education course in software engineering

Tadej Tuma, Iztok Fajfar, Janez Puhan, Arpad Bűrmen
University of Ljubljana
Faculty of electrical Engineering
Tržaška 25, 1000 Ljubljana, Slovenia
tadej.tuma@fe.uni-lj.si

Abstract

We have developed a training board based on the popular M68HC11A1 microcontroller. The described system is robust, inexpensive and nothing particularly innovative. The point of difference compared to other systems is our software: the Integrated Development Tools (IDT) running on the PC platform (Windows 95/98/NT). In summary, our training system works without expensive EPROM and processor emulators, yet providing the student with almost professional performances. There is one more advantage of our development system: it can be downloaded from the web free of charge.

And just how can this combination of minimal hardware and maximal software be exploited in distance learning? The basic idea is to supply each student with a hardware package including a processor board, a power supply unit and an RS232 cable. If produced in sufficiently large numbers, the cost including shipment should be comparable to a medium size textbook.

1. Introduction

There seems to be a contradiction in the tittle of this paper. How can a distance education course be based on a non-virtual concept? And why should we attempt a thing like that? Let us answer the latter question first.

There are numerous well-known reasons speaking in favor of distance education courses. Such courses are inexpensive, time efficient, campus and even country independent, to name just a few. But there is a price to pay: all courses must be delivered virtually. This means there is no personal contact between the students and the teacher, and furthermore, the students have no or very limited opportunities to do practical, laboratory work. Both drawbacks affect the quality of distance education and are being compensated by modern web communications and sophisticated courseware, simulating practical experiments on the student's personal computer.

We believe that even the best simulation cannot give the students the sensation and confidence of manipulating with real experiments. This is why we started looking for a specific course design enabling our off-campus students to get some hands-on experience in developing and testing software for real-time control problems [2]. Of course we want to improve the quality of distance education without compromising its benefits. So we have come up with a solution which meets the following requirements:

1. The target micro-controller system has integrated on-board development hardware.
2. The target / development board is connectable to any standard Windows PC (without additional plug-in boards).
3. The development system has semiprofessional performances.
4. The total cost should not exceed $100.
5. The system supports hardware and software courses as well as different levels thereof.

2. The development system

2.1. Hardware

The board is based on the popular single-chip microcontroller family MC68HC11. We have selected the basic model MC68HC11A1 which includes the HC11 CPU, 256 bytes of internal RAM, 512 bytes of internal EEPROM, 8 programmable timers, an 8 channel 8-bit A/D converter, a serial peripheral interface, and a serial communication interface.

We have kept the additional hardware on the board to a minimum without limiting the functionality. There is 8KB of EPROM, supporting the development system on the PC, 8KB of program memory (RAM or pin compatible EPROM), 8KB of data RAM and the 16 bit parallel interface MC6821. The board has a 50-pin I/O connector for direct controlling purposes and a 40-pin expansion connector giving access to the system bus and a standard serial RS323 connector attaching the system to a personal computer. There is also a relatively strong power supply unit, which is capable of driving peripheral modules over the I/O connector. Figure 1 shows all the hardware features of the basic controller board.

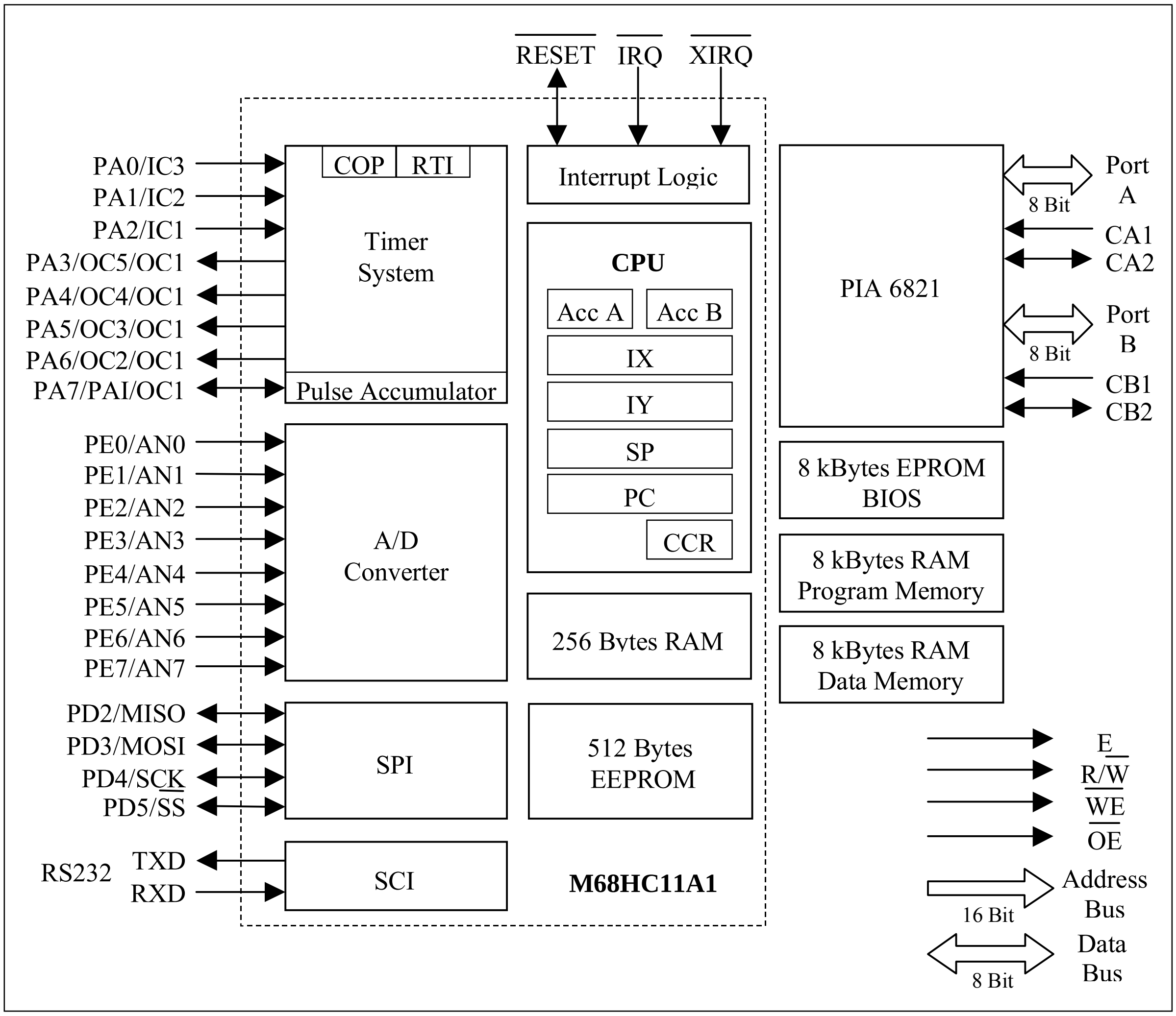

Figure 1. Hardware features of the HC11 training board

The board can be used as an independent module in which case the BIOS EPROM is replaced with a program EPROM, and the 8 Kbytes of program RAM is removed. This operation mode makes the board suitable as a general target system. However, the real value in education is the on-board development system residing in the BIOS EPROM.

After reset the BIOS (Basic Input Output System) takes over and establishes a serial (standard RS232) connection with a PC, running the IDT (Integrated Development Tools) under Windows 95/98/NT. The BIOS is capable of receiving machine code from the PC and placing it into the program memory (RAM). Upon request from the PC, the BIOS will run the downloaded code or execute just one instruction and report back the status of all registers. A running program can be interrupted at any time via the RS232 link, the BIOS gaining control and reporting the board status to the PC. In this way the user can single step his/her program directly on the training board with all peripheral devices also working in "slow motion". The communication between the IDT on the PC and the BIOS on the training board over the serial line is kept at the very minimum. The BIOS is providing only the infrastructure for the development software on the PC.

There is only one obvious disadvantage compared to the professional EPROM and processor emulator approach: you can't do real-time tracing. You either single-step your code or you run it in real-time (or a combination of the two). This is a logical consequence of our approach since the microcontroller cannot run the application at normal speed and record the tracing data at the same time. However, this drawback is not very painful in education and even many professional applications can be debugged and tested without real-time tracing capabilities.

The microcontroller board in figure 1 is the central and most crucial piece of hardware in our laboratory work, but it is by no means the only module. You have noticed that there are no I/O devices on the basic board. The students plug different peripheral modules to the expansion connector or develop their own modules, depending on the course type.

2.2. Software

The IDT (Integrated Development Tools) is a Windows based application coded in Visual Basic with a friendly graphical user interface, freely available from http://www.fe.uni-lj.si/tuma/hc11doc/hc11idt.zip. It includes just about every feature you would expect from a professional tool. The screenshot in figure 2 is showing off a flashy debugging session.

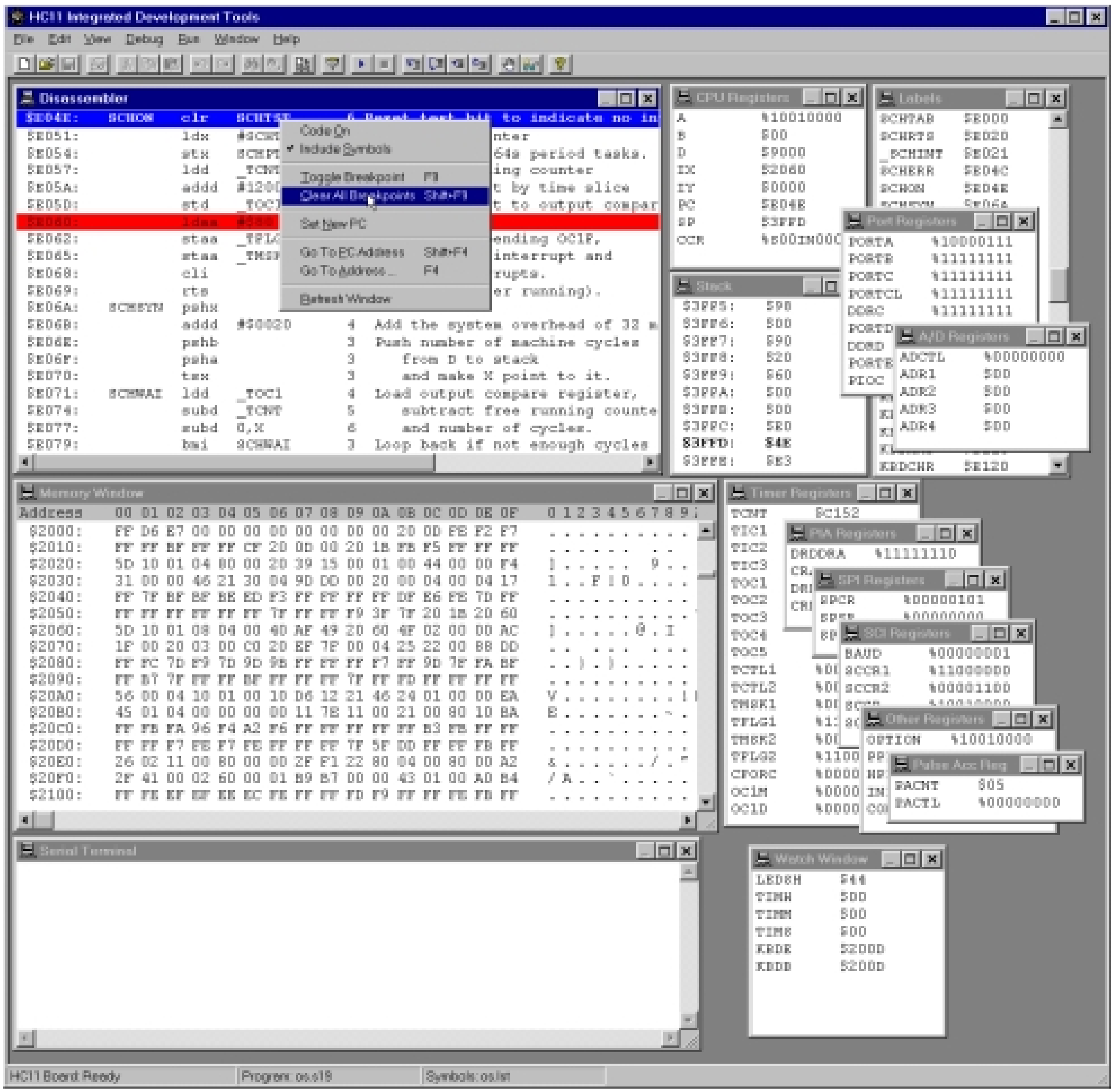

Figure 2. Screenshot of a debugging session with the IDT

There is a source editor, a macro assembler, a download utility, a symbolic disassembler, a serial terminal window, online help, and a complete debugging system. The debugger is capable of single-stepping the source, you can set breakpoints, watches, dump the memory, survey all registers of the system, and inspect the stack. All values can be displayed in hexadecimal, decimal, octal, or binary format. You can also edit any parameter at any time.

A very interesting option is the serial terminal window, which works just like a virtual terminal on the PC, communicating with the user applications on the microcontroller board. This sounds impossible, since the RS232 line is already occupied by the communication between the IDT and the BIOS. We have found a way to temporary withdraw the development communications from the line and let the application use the PC as a VT100 terminal. It goes without saying that a terminal communication routine cannot be single-stepped. As soon as the serial terminal window is active all other windows are frozen. The serial terminal is very important since our basic training board has no other I/O devices!

2.3. Cost considerations

As we have seen in the previous sections, our training system works without expensive EPROM and processor emulators, yet providing the student with semi professional performances.

Until recently we have been using the described microcontroller system in our campus-based courses with occasional students assembling their own board for additional studies or personal use. We thus required only several dozens training systems, which we produced ourselves at a relatively low cost. Lately however we have been planning to employ the same development system in a distance education course. In this case we would have to equip each student with his own hardware, so a very important question arises: exactly how much is “low cost”? We have prepared a calculation in table 1, which is valid for Slovenia and a serial production of at least 300 boards.

Table 1. A cost estimation

Item	Retail price in USD
Printed circuit board	65.00 USD
RS232 cable	3.00 USD
Power supply unit	8.00 USD
Total	**76.00 USD**

We can expect every distance education student to have a standard PC with excess to the Internet. Also there are no software costs since the IDT software is freely available on the net. A virtual engineering course based on our training board would have to cope only with additional hardware costs of about $76, which is unfortunately still too much, even if we consider the students benefit of having a functional development system for future use. However if we could pool at least two courses, for instance a peripheral design course and a software engineering course [4], the additional hardware costs per course would drop under $40, which is just about bearable.

Having estimated a realistic cost frame, we can focus on the most interesting part: how to design a virtual course based on non-virtual courseware?

3. Course design

Slovenia, being a small country, has a very much campus based higher education system. Only lately we are venturing in the world of distance education.

Based on the described development system, we are currently running three courses in different semesters: “Computer Architecture”, a second year course at the Faculty of Computer and Information Science [5, 8], “Basic Microprocessor Systems” and “Microprocessors in electronics”, a second and fifth year course, at the Faculty of Electrical Engineering. We have also introduced our system to a freshmen crash-course on basic programming skills.

Although everything has been carefully prepared for distance delivery, the four courses are officially run in the traditional form. The students are however encouraged to buy their own development board and to take the course at their home computer. Particularly the fifth year generations are very fond of this offer. In recent years we had over 10% of the students built (and funded) their own development board. Therefore let us first talk about our on campus experiences.

3.1. On campus

In the freshmen course the development board is first used as a portable visualization aid when explaining the basics of microcontrollers. The lecturer is attaching a development board to his laptop and using a screen projector to demonstrate typical procedures inside a microprocessor. For instance, you can single-step most illustratively through the role of the stack during procedure calls and their returns. Also the games of setting and employing processor flags can be visualized elegantly. Later, during the laboratory session the students get their first hands-on experiences while developing and testing simple programs to control some LEDs and read some switches. This software crash-course relies on the main development board from figure 1 with a simple keypad & LED display module attached to the parallel expansion connector. The students are given several weeks to become familiar with single stepping and other debugging techniques.

The second year course we are offering at the Faculty of Electrical Engineering is hardware based. The students are designing microcontroller systems, again with emphasis on practical work. Our development system is used as a basic hardware platform to connect and test different modules. In this course we make extensive use of the 50-pin I/O expansion as well the 40-pin system expansion. Especially the latter is very useful, when you want your students to design decoding circuitry without building an entire microprocessor system from scratch.

In the fifth year we have an extensive software-engineering course, concentrating on multitasking and real-time programming techniques. By that time our students are familiar with the HC11 training board, having already taken two courses. They have enough hardware experience and they know how to develop assembler language programs and how to debug them [1]. We introduce a small multitasking operating system evolving around a simple task scheduler. After going through some basic real-time problems, the students are grouped into teams of 2 to 4 and are assigned projects. The projects are carefully selected and highly motivational [6]. A project would typically involve a complete application well known to every lay person and not just parts of some sophisticated application [3]. We all know what a remote control, a credit card reader, a railway crossing, a code lock or an elevator do. It took us quite some time to design small toy-like models for each application. Although this may seem a little childish the students soon discover that controlling a toy robot requires exactly the same approach as does controlling a professional one.

Our way of teaching microprocessor systems is not only rather effective but also very popular with the students. Quite a respectable number of students also take microcontroller systems as their diploma or master thesis subject [7]. In fact an outstanding master's student has developed the presented training board and coded the IDT. Most of our expansion boards and control models have also been designed by graduating students. Also many undergraduates decide to build their own board during one of the courses, which gave us the idea to design a distance education course based on our development board.

3.2. Distance

The design of the HC11 based development board including the IDT software package is actually the result of a small evolution. We have had similar systems ten years ago, based on the MC6803 microcontroller. The predecessor systems were also capable of downloading the code form the PC and single stepping it on-board, but they had a far inferior user interface and a much more complex hardware structure. In other words, these systems were much more expensive and less effective to use. In spite of these fact the earlier development boards were very popular with the students and many of them have asked for the specifications in order to build (and fund) their own boards.

These experiences have directed us to the five design criteria from the introduction. Our final aim is to deliver a virtual course in software engineering with "real" hands-on training. As we have shown in section 2.3, the additional costs in case of two courses relying on the same development system are less than \$40 per course and student. At our Faculty this is just about acceptable.

We will provide the students of the second year on-campus course with the PCBs and all the elements and will require them to assemble their own HC11 development systems as a final course project. This will further reduce the costs, hopefully. After this, the students will be able to take the following software-engineering course at home, using online literature, discussing problems over the Usenet and handing in reports in HTML form.

3.3. Professional and personal use

Our experience from the past decade has shown that enthusiastic students have used even the inferior predecessor development systems for personal purposes. Some of our ex-students have even continued to use the development system in their professional environment.

We are confident that the new improved HC11 based development system will become even more popular as a personal and (to a limited extent) professional tool.

4. Conclusions

We have developed a semiprofessional development system for the HC11 microcontroller. Our design works without expensive processor and EPROM emulators. Apart from the inability of real-time tracing we have spared no pains writing professional development software for the PC platform. So far the system has been successful in our on-campus courses. In future we gradually plan to deliver more and more off-campus courses.

At the same time we are looking for teachers in other institutions with more experience in distance education, who would like to use our development system in their courses. We believe strongly that a cooperation would be beneficial for both parties. At the very least the production cost of PCBs would drop if more teachers were to use the same training boards.

5. Acknowledgements

The presented HC11 based development system has been single-handedly designed and tested by Robert Rostohar a remarkable master's student in 1999. He has well understood our exact requirement.

At this opportunity we would also like to explicitly acknowledge the work of Tomaž Rehar (1995) and Marijan Franc Zaletel (1993), both graduating students who have designed the MC6803 based predecessor systems. Without these two fine students we would have spent a lot of money during all those years.

6. References

[1] M. C. Loui, "The Case for Assembly Language Programming", *IEEE Transactions on Educatio*n, Vol. 31, No. 3, 1988.

[2] D. M. Auslander, C. H. Tham, *Real-time Software for Control: Program Examples in* C, Prentice Hall, Englewood Cliffs, NJ 1990.

[3] P. I. Lin, "Microcomputer Hardware/Software Education in Electrical Engineering Technology: A Practical Approach", *Proceedings ASEE-9*1, New Orleans, LA, 1991, pp. 791-794.

[4] T. F. Leibfried, R. B. MacDonald, "Where is Software Engineering in the Technical Spectrum *International Journal of Electrical Engineering Education*, 1992, Vol. 8, No. 6. pp. 419-426,.

[5] Michael E. Woodward, *Communication and computer networks modeling with discrete-time queue*s, Pentech Press, 1993.

[6] T. Tuma, F. Bratkovič, I. Fajfar and J. Puhan, "A Microcontroller Laboratory for Electrical Engineering, *International Journal of Electrical Engineering Education*, Manchester University Press, Oxford Road, Manchester, 1998, Vol. 14, No. 4, pp. 289-293, (http://www.fe.uni-lj.si/tuma/ijee1028.pdf).

[7] T. Tuma, I. Fajfar, J. Puhan, "The Principle of Vaccination in Teaching", *Frontiers in Education Conference '99*, San Juan, Puerto Rico, 1999, pp. 11b4/16-11b4/20, (http://www.fe.uni-lj.si/tuma/1115.pdf).

[8] T. Tuma, I. Fajfar, M. Perko, F. Bratkovič, J. Puhan, "A Hands-on Approach to Teaching Basic OSI Reference Model", *International Journal of Electrical Engineering Education*, Manchester University Press, Oxford Road, Manchester, 2000, Vol. 37/2, pp. 157-166, (http://www.fe.uni-lj.si/tuma/teachlan.pdf).