Building a Real-Time Digital Signal Processing Course and Teaching Laboratory

Phillip L. De Leon
New Mexico State University
Klipsch School of Electrical and Computer Engineering
Las Cruces, New Mexico 88003-8001

Abstract

Electrical engineering graduates with course work in traditional digital signal processing theory and real-time digital signal processing (DSP) hardware are among the most sought after by potential employers. Training electrical engineering students on system design using this important technology presents several challenges. First, undergraduates typically obtain their first exposure to digital signal processing theory in their senior year thereby not allowing for a follow-up course in real-time digital signal processing (RTDSP). The first exposure usually does not include DSP hardware. Second, the cost of a RTDSP teaching laboratory can run as high as $5000/workstation when the costs for PC, DSP development tools, and test and measurement equipment are factored in. Third, complexity of these systems and long development time, even for simple projects, present major hurdles in determining what projects can or cannot be developed in a one-semester course. In this paper, we present our approach to developing a one-semester, stand-alone course in real-time digital signal processing. We recommend placement of the digital signal processing sequence in the curriculum, provide information on obtaining DSP hardware at little or no cost, information on faculty training, and a typical one-semester schedule with applications.

Introduction

The advent of digital signal processors is evidenced by wide application in consumer products such as cellular telephones and pagers; audio/video electronics such as CD, DVD, and DirectTV; Ethernet cards, hubs, and modems; and multimedia PCs. With an annual growth of over 40% since 1988, the DSP market has exceeded the growth of the overall semiconductor industry in recent years [1]. In fact digital signal processing has been called the “stealth technology” in that it is one of the unknown engines behind the information age [2]. A digital signal processor differs from a general-purpose microprocessor (such as a Pentium) in that the former is optimized for common DSP operations such as inner products (discrete-time convolutions) and fast Fourier transforms (FFTs). This optimization combined with tremendous advances in microelectronics has driven down cost of processing signals in the digital domain.

Electrical engineering graduates with course work in traditional digital signal processing theory and real-time DSP hardware are among the most sought after by potential employers. However, preparing

engineering students on system design using this important technology presents several challenges. First, undergraduates typically obtain their first exposure to digital signal processing theory in their senior year thereby not allowing for a follow-up course in real-time digital signal processing (RTDSP). The first exposure usually does not include real-time DSP hardware. Second, the cost of a RTDSP teaching laboratory can run as high as $5000/workstation when the costs for PC, DSP development tools, and test and measurement equipment are factored in. Third, complexity of these systems and long development time, even for simple projects, present major hurdles in determining what projects can or cannot be developed in a one-semester course by an undergraduate. In this paper, we present our approach to developing a one-semester, stand-alone course in real-time digital signal processing. We recommend placement of the digital signal processing sequence in the curriculum, provide information on obtaining DSP hardware at little or no cost, information on faculty training, and a typical one-semester schedule with applications.

Objectives in a Real-Time DSP Course

Our objective in the real-time digital signal processing course is simple. By the end of the course a student is capable of taking a description of a signal processing application from a textbook or journal article to algorithm and working code for real-time execution on a modern digital signal processor (DSP). This objective is certainly ambitious when one considers that most students have had only one semester of DSP experience, typically minor experience in real-time programming, and probably have never dealt with a processor as complicated as a DSP. However, in our limited experience, the vast majority of students demonstrate this capability through their final project which will be discussed in a later section. We believe the key to this success is the fact that after a few weeks of introduction, the students are in a constant state of programming. That is to say they are always writing code for projects and the projects increase in complexity as the semester goes on. We will discuss the semester schedule in a later section as well.

Building a Real-Time DSP Laboratory

Perhaps the most difficult part about building a RTDSP course and teaching laboratory is getting started. The basic DSP workstation consists of a PC, DSP board, and software development tools (editor, assembler or compiler, and debugger). Additionally, CD-ROM, amplifier and loudspeakers, and miscellaneous audio cables are required for digital audio projects. Finally, basic test and measurement equipment such as an oscilloscope and function generator are crucial to code verification. We have found that the ratio of students to DSP workstation should be no greater than three and that approximately one-quarter of DSP students take the RTDSP course.

The three primary DSP vendors (Texas Instruments, Motorola, Analog Devices) all have University support programs which provide a DSP board and software development tools for little (~$100) or no cost

to the department [3], [4], [5]. The Klipsch School has adopted the Motorola DSP as they provided twelve DSP boards and associated software tools at no cost. In addition, with the relatively low-cost hardware, we found many students optionally purchased the board from Motorola as they preferred working on their own PC at home. All of the above DSP vendors have 2-4 day courses available for hardware designers as well as faculty [6], [7]. Often the tuition to these courses can be waived (as was ours with Motorola). These courses provide perhaps the fastest way to get an instructor up to speed on the hardware. In addition, each processor has available textbooks from which to also learn and possibly adopt for the course [8], [9], [10], [11]. We have adopted El-Sharkawy's text but have extensively supplemented it with our own material and projects [10]. Finally, there is extensive reference material, sample source code, etc... available on the Internet. A good place to start is [12].

In order to minimize costs we have employed existing multimedia PCs (66MHz 486) within our department and fitted the DSP board inside a standard 5 1/4" bay. By fitting the DSP board inside the bay we solved a number of problems such as protecting the board from possible physical or electrical damage and eliminating the required external power supply (board is powered off the PC's internal power supply). This arrangement also provides security for the board. The total cost of supplies for this arrangement is less than $10 per DSP workstation.

RTDSP Course Outline

We focus the course on applications of DSP to digital audio and digital communications and not on system hardware issues such as interfacing the DSP to memory and/or analog-to-digital converters (A/D) and digital-to-analog converters (D/A) (an integrated A/D and D/A is often referred to as a codec). Therefore we assume code to initialize the codec, establish communications between the codec and DSP, and transfer samples to/from the codec and DSP. Code to perform these functions is often referred to as pass.asm (assembler code) or pass.c (C language code). These codes are typically included with professional development systems and are standard with student-targeted systems. With these codes students can focus on the processing of the digital signals themselves rather than with the support code.

We assume a semester long, stand-along course in real-time DSP taken immediately after a semester-long DSP course. Our experience has indicated that supplementing the traditional undergraduate DSP course with some hardware laboratory experience proves confusing and frustrating to many students since most of the DSP concepts are new to them. Even if the undergraduate DSP course is supplemented with RTDSP, it is unlikely the codes will be more sophisticated than digital filtering.

In our approach, we use the schedule given in Table 1 and point out the following.

- The first phase is composed of three weeks of familiarization with the development tools and DSP56002EVM operation. We end this phase with coding digital filters (FIR and IIR).
- The bulk of the course (second phase) involves individual coding projects. While coding of projects is taking place, lectures on the next project are taking place. This way the student is coding throughout the semester on progressively more difficult projects.
- The student is allowed to demonstrate creativity by selecting a final project of interest. The individual's final project is formally presented at the end of the semester.

We have chosen the projects to provide a good cross section of DSP applications in digital audio and digital communications. The projects demonstrate many features of the DSP and application to a wide range of engineering problems and are listed in Table 1. The quality of final projects has exceeded our expectations. Such final projects include modem design using various signaling schemes such as frequency-shift keying (FSK) and quadrature phase shift keying (QPSK), speech recognizer (single user, limited vocabulary), voice scrambler, advanced music/wavetable synthesizers, etc....

Table 1: Semester schedule for the real-time DSP course

Week	Lecture Topic	Current Project
1	Introduction, Software Tools, DSP56002EVM Operation	
2	DSP56002 Architecture, Addressing Modes, Instruction Set	
3	Instruction Set, CS4215 Audio Codec, Demo/Canned Code	
4-5	Project 1 Sound field simulator	Digital Filters
6-7	Project 2 Adaptive noise canceler	Project 1
8-9	Project 3 Wavetable synthesizer	Project 2
10-11	Project 4 Binary phase shift keying (BPSK) modem (300bps)	Project 3
12-13	Advanced Topics (finite precision effects)	Project 4
14-15	Advanced Topics (other)	Final Project
16	Final Project Presentations	

Measuring Success

Success throughout the course is measured by how well the student achieves our objective for the course. This is measured through the four projects and final project which count for 70% and 30% of the final grade respectively. If the student has demonstrated that the code performs as it is supposed to, full credit is awarded; otherwise, partial credit is awarded. We note, however, that non-working code will never score above 75/100 points. Our motto in the course is "anyone can write code that does not work." We expect the student to produce code that executes properly in much the same way that a future employer

might. The student must also design a test set to prove that their code functions properly. Proving that the code works can often times be as difficult as writing it.

In striving to meet our objective, we are periodically rewarded by student surveys and notes from former students. Many student surveys suggest that they really began to understand textbook DSP when they got involved in writing RTDSP code. Other students suggested that building solutions to engineering problems using DSP made them feel like real engineers. Perhaps most rewarding of all, are our students who pursue DSP hardware/software upon graduation either in industry or graduate school. In the first two offerings of this course, approximately one-third of our students have accepted DSP development positions at such companies as Hewlett-Packard, Motorola, and Texas Instruments.

Conclusions

The RTDSP teaching laboratory in the Klipsch School is based on low-cost DSP development tools and general-purpose multimedia PCs in order to keep costs to approximately $500 per DSP workstation (excluding PCs). Our course in RTDSP appears to consistently achieve its objective of enabling students to take a description of a signal processing application from a textbook or journal article and write code for real-time execution on a modern DSP. This capability allows our students to implement a wide-range of signal processing applications such as sound field simulators and modems digitally. We have been impressed with students' determination in project coding and the caliber of final projects our students undertake.

References

[1] T. Chen ed., "VLSI design and implementation fuels the signal-processing revolution," IEEE Signal Processing Magazine, Jan. 1998, p. 22-37.

[2] L. Jamieson., "President's message", IEEE Signal Processing Magazine, Jan. 1998, p. 8-14.

[3] http://www.ti.com/sc/docs/dsps/univprog/teachkit.htm, Jan. 1998.

[4] http://www.mot.com/SPS/DSP/university_relations/, Jan. 1998.

[5] http://www.analog.com/pdf/21x_tool.pdf, Jan. 1998.

[6] http://www.analog.com/support/dsp_workshops/dsp_work.html, Jan. 1998.

[7] http://mot-sps.com/training/catalog/dsp5600x.html, Jan. 1998.

[8] S. A. Tretter, Communication System Design using DSP Algoroithms with Laboratory Experiments for the TMS320C30, Plenus Press, 1995

[9] H. V. Sorensen, J. Chen, *A Digital Signal Processing Laboratory using the TMS32030*, Prentice-Hall, 1997.

[10] M. El-Sharkawy, Digital Signal Processing Applications with Motorola's DSP56002 Processor, Prentice-Hall, 1996

[11] Digital Signal Processing Laboratory Using The ADSP-2101 Microcomputer, V. K. Ingle and J. G. Proakis, Prentice-Hall, 1991.

[12] http://www.techonline.com/, Jan. 1998.