Economy Sized DSP: Signal Processing Instruction on a Budget

Gregory M. Dick
University of Pittsburgh at Johnstown

Abstract

Instruction in Digital Signal Processing can be a very expensive proposition. The cost of a single PC based DSP laboratory station can easily exceed $4,000. Financing a complete lab can be a financial impossibility for some institutions. The benefits of a strong laboratory component to a DSP course are obvious. However, the use of several generally available resources can fulfill some of the functions of the laboratory at greatly reduced costs.

This paper outlines the history of DSP instruction at the University of Pittsburgh at Johnstown (UPJ). This case study discusses how relatively ubiquitous tools (spreadsheets, advanced mathematical languages (e.g., MATLAB), conventional programming languages, and inexpensive additions to PCs) can be used to support undergraduate DSP education. This approach allows quality DSP instruction - with a modest "laboratory" component to be carried out while resources for more elaborate facilities is being sought.

Introduction

Undergraduate instruction in Digital Signal Processing has become increasingly popular over the last decade. This is particularly appropriate in light of the significant advances in DSP technology. However, the cost of laboratory equipment to support signal processing instruction can be an impediment to the institution of a new course. The cost of a single, stand-alone DSP laboratory station can easily exceed $4,000. This can be a particularly acute problem in these days of shrinking budgets throughout the higher education community.

"Introduction to Digital Signal Processing" has been taught at the University of Pittsburgh at Johnstown (UPJ) for ten years. As of this date, all practical exercises have been conducted using existing resources. Thus, the department has not been required to make major investments to support this course. This paper documents the history of the development of this course. It is hoped that it may help others to initiate DSP instruction "on a budget".

Course Chronology

Fortran - the Early Years

Instruction in Digital Signal Processing at UPJ was initiated in 1987. Then, as now, equipment budgets were less than extravagant. The expenditure of scarce resources to support a new course, whose success was not established, was not viewed as prudent. A decision was made to proceed with the initiation of a lecture based course which would be supported by software based practical exercises.

The computing environment at Pitt in the late 1980's was centered on a time-shared system. The University operated a cluster of several Digital Equipment VAX processors. These were available across the University's five campuses, including Johnstown. No specific signal processing applications were available.

Software based DSP "laboratory" exercises were conducted using Fortran. Using this approach, the students were required to:

- Construct "Laboratory Equipment" - The students wrote Fortran programs which would perform the functions of signal generator and oscilloscope. Sinusoidal, random, and modulated signals were typical. General purpose plotting routines, based on Tektronix 40xx technology, were available. These were adapted for their specific needs.
- Construct Digital Filter Design Tools - Standard design techniques were implemented with Fortran programs. Finite Impulse Response (FIR) filters were designed using Fourier techniques and Windowing methods. The design of Infinite Impulse Response (IIR) filters based on analog prototypes was investigated.
- Construct Signal Processing Elements - Digital filters were implemented in Fortan functions and used to filter a variety of sampled signals. Algorithms which implement both the Discrete Fourier Transform (DFT) and it's computationally superior version, the Fast Fourier Transform (FFT), were coded.

The construction of tools, "laboratory equipment", and signal processing elements takes a considerable amount of the students' time. Nevertheless, several simple, but illuminating, exercises were conducted each term. A typical project might involve:

- The generation of a signal which is corrupted with random noise
- The design of a filter (or several) to remove the noise from the signal
- The analysis of the filtered signal to quantify the effectiveness of the signal processing system (filter).

MATLAB - a Big Step Forward

Students benefited significantly from the programming exercises outlined above. Student resources (i.e., time) are finite. The investment of time to build tools in Fortran diminished the amount of time they could devote to experimenting with different signal processing techniques. This problem was solved in the early 1990's when MATLAB was installed on the University's

time-share system. MATLAB, and it's associated Signal Processing Toolbox, includes powerful tools to support experimentation in DSP. In many cases a single line of MATLAB code replaces an entire Fortan program. e.g.,

```
[b,a] = butter(5, 30/(100/2));
```

designs a fifth order Butterworth low pass filter with a cut-off frequency of 30 Hz for signals sampled at 100 Hz. The addition of a few more lines of code can implement a simple project which might:

- Generate a signal corrupted with random noise
- Design a filter and apply it to the corrupted signal
- Display both the original and filtered signal.

The availability of these powerful primitives allows the students to concentrate their efforts on signal processing, not tool building. Therefore, they can be expected to approach more sophisticated projects. The set of MATLAB based projects which have been completed include:

- The design and implementation of the receiver subsystem of a Bell 202 compatible (FSK) modem. This subsystem is then used to decode messages in composite signals which contain both information and noise.
- The analysis of temperature data which describes the cooling and solidification of a particular alloy. These data were first filtered to remove noise, then analyzed to identify the time when phase change occurred. Real data from a local metallurgical laboratory was used.

Hardware - "it really works"

Student understanding of DSP principles was measurably improved by software simulation exercises (both Fortran and MATLAB based). Lecture discussions were reinforced and interest in the material was increased. However, a certain (appropriate) level of skepticism was expressed by the class. "The simulation results agree with theory, but ... does this REALLY work? Like, with signals in the electronics lab?" The connections between theory, simulation, and "real hardware" are not obvious to the neophyte signal processor.

In 1991 the opportunity to address this problem presented itself. The key elements were: (1) the availability of several PC analog interface cards (MetraByte DASH-16 cards purchased to support a Measurements course) and (2) the appearance of three bright and enthusiastic students eager to do a special project. An appropriate project was defined. It was the design and implementation of a software "shell" to make the analog interface hardware (A/D and D/A converters) and associated DSP processes available to users in an easy to use environment. The function of the resulting system (PC, analog interface, student software) can be summarized:

- Inputs:
    - Parameters:
        - sampling frequency
        - filter coefficients
    - Electrical Signal - 5.0 volts maximum
- Outputs:
    - Screen error messages (e.g., sampling frequency too high)
    - Electrical Signal - the filtered input signal.

The system was designed, implemented and tested. It met it's design criteria and could successfully filter analog signals. The performance of the system was limited by the relatively slow speed of the A/D and D/A channels, but this did not reduce the system's ability to support demonstrations.

This system, titled "Real Time DSP" by the project team, made a significant contribution to the students understanding. It was deliberately constructed with neither anti-aliasing nor reconstruction filters. The resulting "stairstep" output graphically displayed the effect of sampling. Dual trace oscilloscope displays of input and output signals clearly exposed the delay (phase shift) inherent in real sampled data systems. Students observations of a 240 Hz input signal, and the resulting 40 Hz output sinusoid cleared up many misunderstandings about the concept of aliasing.

Spreadsheets - Another Perspective

Throughout the 1990's, the University constructed several well equipped PC labs. Another "free" tool for DSP instruction became available. The ubiquitous spreadsheet can be used to implement digital filters. The spreadsheet is not the most effective tool for digital filtering. However, its cellular nature is useful in examining the structure of DSP algorithms. Many texts explain the sum-of-products algorithm with the help of signal flow diagrams. The use of a spreadsheet is simply one more pedagogical tool to help the students grasp this facet of the topic.

Modern DSP Hardware - the Next Step

"Introduction to Digital Signal Processing" has been a successful and valuable course. However, it lacks exposure to modern signal processing hardware (i.e., the family of special purpose microprocessors which have come to be known as DSP chips). The next step in the development of this course will be the incorporation of real DSP hardware into a set of laboratory exercises. In the 1980's, when this course was first developed, the cost to do this was prohibitive. Now, fortunately, that is not the case. Evaluation modules based on a DSP microprocessor are available at modest prices. Such systems typically include:

- DSP microprocessor
- modest memory
- analog I/O capability
- assembler - linker - loader
- debugger
- example application software.

These evaluation modules are designed to be hosted by a Windows/Intel based PC, thus the cost of establishing a "Signal Procession Workstation" is the incremental cost of the evaluation module. Currently, such systems are very affordable (e.g., the TMS320C30 DSP Starter Kit is available from Texas Instruments for about $100, a more sophisticated Evaluation Module which includes more development tools and enhanced hardware is available for less about $1000). We are currently investigating the logistics required to acquire a set of evaluation modules and install them in an existing instructional computing laboratory.

Conclusions and Recommendations

Instruction in Digital Signal Processing can be a valuable addition to an undergraduate curriculum. The absence of major funding need not stand in the way of establishing such a course. The litany of (low cost or "free") tools above suggests that practical exercises can be conducted with a minimal investment of resources. It is our observation that each tool has particular strengths and weaknesses.

- Traditional Fortran (or C) Programming Exercises - Such projects give the students valuable insight in to the internal structure and operation of DSP algorithms and limited exposure to real applications. The time devoted to tool building detracts from time available to implement applications.
- MATLAB Projects - The powerful primitives and built-in DSP operations allow for more concentration on applications. It has been observed, not surprisingly, that the students understanding of the "internals" of the algorithms suffers. We have found that a combination of traditional programming exercises and MATLAB exercises seems to be the most effective approach to building a strong understanding of both the algorithmic approach and applications.
- Hardware - Some exposure to real-time processing is extremely valuable. The students' observation of "real" signals on an oscilloscope goes a long way toward reinforcing fundamental concepts.
- Spreadsheet - This tool is helpful in reinforcing the sum-of-products structure of DSP algorithms. We have found it's use in implementing signal processing applications somewhat limited.
- Modern DSP Hardware - Inexpensive evaluation modules make the power of real DSP systems affordable for programs with modest resources. They carry the promise of integrating several of the tools above. A project might include:
    - MATLAB based design and testing (simulation) of a digital filter. The power of MATLAB's Signal Processing Toolbox supports a rapid prototype approach.
    - Implementation of the filter in the C language. Knowing that the "correct" filter has been designed, it could be coded in C and tested using the same data used to exercise the MATLAB implementation. Results which agree verify the C implementation.
    - Real-time implementation on the DSP evaluation module. Minor changes to the C program should adapt it for execution on the DSP hardware. Proper operation in a real-time environment validates the entire process.

If the institution of DSP instruction is being considered, the process should proceed. A survey of "free" tools - as outlined above - may indicate that a course, with some practical exercises, can be instituted without the need for new funds. Later, the success of the course may well justify the expenditure of future resources to further develop the laboratory component of the course.

GREGORY M. DICK is Associate Professor of Electrical Engineering Technology at the University of Pittsburgh at Johnstown (UPJ). His academic interests include Digital Signal Processing, Digital Control Systems and Computer Applications to Engineering Education. Dr. Dick is a registered Professional Engineer and holds degrees from the University of Pittsburgh, The Pennsylvania State University, and Stanford University.