2006-1313: DSP ON GENERIC MACHINES

Dick Blandford, University of Evansville

Dr. Dick K. Blandford is the Chair of the Electrical Engineering and Computer Science Department at the University of Evansville.

DSP on Generic Machines

Abstract

Many electrical engineering classes which introduce digital signal processing at the undergraduate level include a laboratory component in which students implement systems on dedicated DSP boards. Many such boards are programmed in an unfamiliar assembly language or they require cumbersome I/O drivers. Instructors must decide whether to spend class time teaching arcane programming techniques or to give the students the templates as "magic" pieces that make it all work. In this paper, another alternative is considered: can a generic processor with which the students are familiar from a previous embedded systems class be used to teach DSP, and if so, what is lost by not using the dedicated DSP systems? The pros and cons of both sides are considered, and results from the use of three different generic processors are presented.

Introduction

Over the past 20 years, digital signal processing (DSP) has been introduced into the undergraduate electrical engineering curriculum. Today many programs either require an introduction to DSP, or introduce the topic as a significant part of one or more courses related to linear systems. An introductory DSP class typically follows a first course on linear systems and is most often taken as a second semester junior level class or at the senior level. However, some schools have begun introducing DSP at the sophomore level as a vehicle for teaching linear systems in place of, or as a supplement to the traditional circuits sequence.

Many DSP introductory classes include a lab where students implement digital algorithms on real-time signals. The audio band of frequencies provides suitable applications. Many introductory courses are limited to this band; however, this seems to be slowly changing as processors become more capable and applications such as wireless technology are moved into the undergraduate curriculum.

Typical labs feature experiments carried out on one of the many DSP development boards produced for this purpose, such as those from Analog Devices, Motorola, Texas Instruments and others. Matlab, or one of its clones,[1, 2] is used for almost all DSP systems as the vehicle to design algorithms, which are then implemented in either assembly language or in a compiled language which is almost always C.

For students, learning a new assembly language is sometimes difficult. Even for algorithms implemented in C, the low level interfacing software is often provided without much in the way of explanation.

This paper examines an alternative strategy of using a generic processor to implement DSP algorithms with the inherent advantage that, in many cases, students coming into the class have a

working familiarity with such a processor. Therefore, little instruction is needed to learn a new language or to understand the hardware.

Generic processors

For this paper, a generic processor will be defined as one where the intended use is not for digital signal processing. For example, a microcontroller used in an embedded systems class would be classified as a generic processor.

An examination of 37 EE programs (via the Internet) shows that in addition to teaching courses in logic design and programmable logic, EE programs include significant instruction on microcontrollers. Numerous microcontrollers are in use with 8-bit microcontrollers dominant[3]. For the 37 EE programs examined in the survey, nearly 85% used an 8-bit microcontroller with about equal numbers using the Microchip PIC processor, the Motorola (now Freescale) 68HC11, an Intel 8051 variant, or an Atmel AVR processor. The remaining 15% of programs used a 16-bit microcontroller (such as a Motorola 68HC12 or a Texas Instruments MSP430) with just two programs using the 32-bit ARM processor variant.

All of the processors in use in microcontroller based courses are available with software development systems that include an assembler and a C compiler and usually include a simulator and a debugger. Many of the processors have on board an analog-to-digital converter and these are most often 10-bit converters. Digital-to-analog converters on board are much less common but on board hardware for pulse width modulation (PWM) is usually available and makes a convenient D to A converter. A few of the processors have multiply and accumulate (MAC) units on board for implementing difference equations but the MAC unit is unusual and almost non-existent on 8-bit processors. None of the processors have circular buffers but all have sufficient program and data storage memory suitable for most DSP algorithms.

To simplify the process of evaluating processors for DSP applications the author has chosen a single processor from the 8-bit, 16-bit, and 32-bit processors for more detailed evaluation: an Intel 8051 variant, a Texas Instrument MSP 430 processor, and an ARM7 processor. These three were chosen for their convenience and availability (to the author). Although they may not be typical, they represent the wide variety of processors that are in current use as generic processors in microcontroller courses.

Processing speed is an important issue with regard to DSP. The generic processors examined present a wide variation in processing speed. With the exception of the 8-bit Intel 8051 variant, the author has not attempted to find the fastest processor for a given type but reasonable linear extraction can be made by comparing multiply times and A to D conversion times.

Processor details

The Dallas Semiconductor DS89C450,[5] which is an Intel 8051 variant, is an 8-bit machine with a very fast instruction cycle. The original 8051 had a 12MHz crystal and used 12 clock cycles per instruction, allowing it to carry out 1 instruction per microsecond at best. The DS89C450 has a maximum clock speed of 33 MHz and does one instruction per clock cycle. This allows to

do about 1 instruction every 30 nanoseconds. Thus it is more than 30 times faster than the original 8051. The processor has 64K of flash program storage and 1K of static RAM. It does not have an A to D converter, a D to A converter, or pulse width modulation in hardware. All data paths and the ALU are just 8-bits wide. There are three 16-bit timers and a fast interrupt system. Software for this machine was developed using the Keil C compiler and assembler.

The Texas Instruments MSP430F169 processor[6] was chosen as a 16-bit test processor for DSP. This processor has 60K of flash memory for program space and an additional 256 byte data flash. It has 2K bytes of RAM memory and runs with an 8MHz crystal. All of the data paths, registers, and ALU have a 16-bit width. The RISC instruction set executes register-to-register instructions in a single 125 nsec instruction cycle. There is a hardware multiplier and the assembly language has a MAC instruction. There are two 12-bit A to D converters and one 12-bit D to A converter. An unusual feature of this processor is that it supports three DMA channels. This processor is currently available in several versions with some running at 16MHz. Software for this machine was developed on the IAR Embedded Workbench.

The Philips Semiconductor LPC2138 ARM7 processor[7] was chosen as the 32-bit test processor. It uses a phase locked loop to drive the processor at 60MHz from a 12MHz crystal. The LPC2138 has 512 K or flash program memory and 32 K of static RAM. All of the data paths and ALU are 32-bits wide, although the processor has a "thumb" mode that allows it to operate on a 16-bit wide thumb instruction to save program memory space. There are two 8-channel 10-bit A to D converters and one 10-bit D to A converter that provide an onboard analog interface. The A to D can complete a 10-bit conversion in 2.44 μseconds. There are seven onboard PWM registers that allow for 6 single-ended or 3 double-ended PWM signals generated in hardware. Other DSP-like hardware features include an onboard 32-bit barrel shifter and a MAC unit. The processor has a 3-stage pipeline. Software for this machine was developed using the Keil CARM compiler and assembler.

DSP requirements

To determine DSP requirements the author surveyed the teaching materials and lab requirements for introductory courses in DSP which include a lab where a real-time DSP system is in use[4]. All courses considered were undergraduate courses taken as a first course in digital signal processing. Table 1 summarizes the results of this survey.

While many of the experiments and projects were done at frequencies beyond the audio band, it is clear that all could be done at lower frequencies without degrading the concept being taught. For example, aliasing can be illustrated using a 1 KHz sampling frequency as well as it can be using a 44.1KHz sampling frequency. For a few experiments such as "audio effects and audio noise filtering", a sampling frequency of at least 20 KHz is desirable but for others a minimum sampling frequency of 11.025 KHz is adequate.

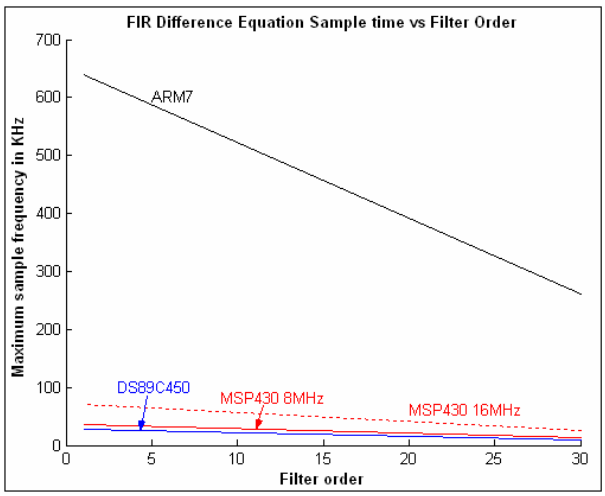
Topic	Description
Lab introduction	Students are given an overview of the lab hardware and a "canned" experiment to complete.
FIR filters	Students are asked to design and implement a real time FIR filter from specifications
IIR filters	Students are asked to design and implement a real time IIR filter from specifications
Oscillator	Students are asked to design a sinusoidal oscillator using various algorithms
Correlation	Students are asked to use correlation for filter design and experiments in filtering. Not done in real time.
Aliasing	Students are asked to input from and A to D and output the same signal to a D to A as input sinusoids roll past the sample frequency
PWM	Students are asked to produce PWM signals and apply analog filters to the output.
PI, PD, or PID controller	Students implement a PI, PD, or PID controller for a dc motor with a flywheel.
System identification	Students use a DSP system to measure the response of a system to inputs created by the DSP. The response is used to identify the system.
Inverted pendulum controller	Students use a DSP system as the controller on an inverted pendulum which moves in one dimension.
FFT	Students are asked to use a DSP system to calculate the FFT on a non real-time sequence.
Audio effects and audio noise filtering	Students are asked to use a DSP system and an input audio signal to demonstrate such audio effects as time delay, echo, flanging, reverberation, noise reduction.
Over sampling	Students use a DSP system with a fast A to D to over sample, filter, and down sample a signal to improve the SNR. A signal with a 1.25KHz bandwidth is sampled at 20KHz, sent through a low pass filter, and sent to a decimation filter.
Spectral analysis	Students are asked to use a DSP system to analyze the spectrum of typical EEG and ECG signals. (not in real time)

Table 1

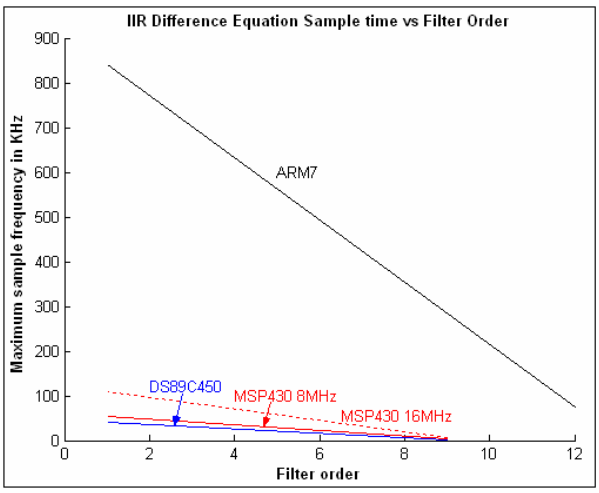
This table shows typical assignments found in introductory undergraduate DSP classes.

Summary of results

To determine the maximum evaluation time for difference equations for various processors, the author calculated the computation time based on multiply and add times published in specifications for each processor. To verify these calculations, three FIR and three IIR filters were created and implemented on each processor. The actual execution time was measured and found to fit the computed evaluation time with a small error.[8] Figure 1 shows the results for the three processors. The 32-bit ARM7 processor running at 60MHz is a clear stand-out from the others and Figure 2 shows a blow up of the graphs in Figure 1 with the ARM7 omitted.



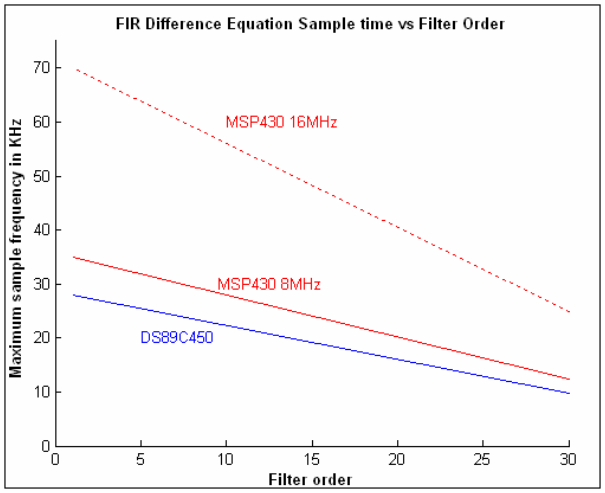
a) FIR Filters



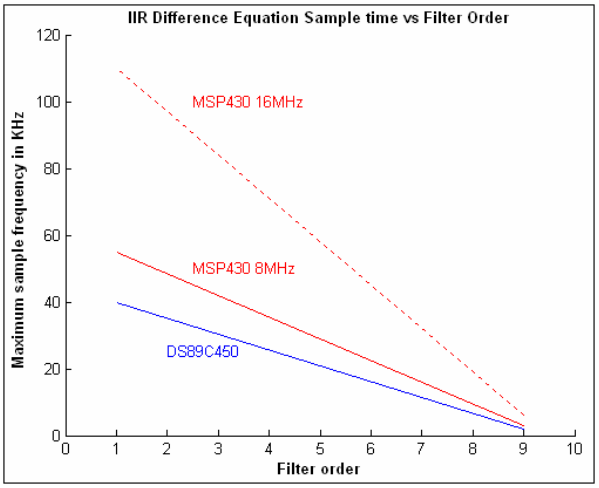
b) IIR Filters

Figure 1

Maximum sample frequency vs. filter order for a) FIR filters and b) IIR filters. The FIR filters were low pass with linear phase and the IIR were low pass Butterworth filters with symmetric numerator polynomials. The graphs show sampling frequencies for evaluation of the difference equations only and does not include computation time for variable shifting and analog conversion. All difference equations used 16-bit integer arithmetic.



a) FIR filters



b) IIR filters

Figure 2

Maximum sample frequency vs. filter order for a) FIR filters and b) IIR filters. This is an exploded scale view of Figure 1 with the much faster ARM 7 processor omitted.

The graphs show that if you are willing to restrict computation to 16-bit integer arithmetic then even the 8-bit DS89C450 is adequate for illustrating many of the concepts in an introductory DSP class. Using a fast 16-bit processor has clear advantages and provides more flexibility. The 32-bit ARM7 processor would clearly meet all needs and in addition, 32-bit integers could be used with no penalty in time since all data operations use 32-bit arithmetic on this processor.

Note that none of the three processors considered have any built-in floating point capability, and only the MSP430 has a hardware multiplier for integer arithmetic. Floating point arithmetic is an

option in software and can be successfully used on even the slower 8-bit DS89C450. Measurements of floating point implementations on the DS89C450 show that it runs 3 to 5 times slower. For example, a third order Butterworth filter could implement an integer version using 16-bit arithmetic at about 33 KHz (maximum), but when floating point arithmetic is used in C, the speed drops to about 10KHz. The MSP 430 and the ARM7 similarly slows down by a factor of 3 to 5.[9]

Conclusions

The author has introduced generic processors into the introductory DSP class in the spring of 2005. Follow up focus groups with students and assessment of work done by students by independent group of faculty show favorable results. The assessment is an ongoing process and follow up work is being done to see how students incorporate DSP into their senior projects.

The author notes several advantages of this approach as well as some clear negative side-effects. These are listed below.

Benefits of using generic processors in a DSP class:

- Students can begin implementations very quickly without additional labs aimed at introducing and using new software and hardware.
- Students who have implemented DSP algorithms on generic processors should be more likely to consider DSP as a design alternative in embedded systems projects.
- This approach provides some "mental glue" between classes so that an embedded systems class and a DSP class are not isolated incidents of the use of computers.
- There is a wider range of hardware and software available for generic processors than there is for dedicated DSP systems. This provides flexibility for different and possibly more interesting applications.

Disadvantages of using generic processors in a DSP class:

- Students will be less familiar with dedicated DSP processors. Such familiarity may be of value in the workplace.
- DSP chips are inherently faster than most generic processors so that a wider range of signals can be dealt with. This gives a bit more flexibility in course design.
- Many DSP systems have a direct link to design software such as Matlab, which is not generally available for generic processors.
- Generic processors do not provide anti-aliasing and anti-imaging filters which are typically available as part of DSP system boards.

It is suggested that perhaps the best approach is to use generic processors as an implementation vehicle in the first half of the course and move to a dedicated processor in the last half of the course. Such an approach is currently being explored.

References and notes

[1] Scilab at http://www.scilab.org/

[2] Octave at http://www.octave.org/

[3] Of the 37 programs examined no program used a 4-bit processor, 33 used an 8-bit processor, 5 used a 16-bit processor, and 2 used a 32-bit microcontroller. Some programs used more than one processor.

[4] All of the data in this survey was taken from published lab manuals and experiments on the internet as well as the author's own introductory lab.

[5] Maxim Ultra High Speed Flash Microcontroller's User Guide at http://www.maximsemiconductor.com, Revision 6, December, 2004

[6] Texas Instruments MSP430x1xx Family User's Guide Extract from Texas Instruments, 2005 see http://focus.ti.com/lit/ug/slau169/slau169.pdf

[7] LPC213x User's Manual, Philips Semiconductor, June, 2005 see http://www.semiconductors.philips.com/acrobat_download/usermanuals/UM10120_1.pdf

[8] The actual error was about ±7%. The true curve is not a straight line but zigzags around a straight line depending on whether the filter order is odd or even because of symmetry of the coefficients.

[9] Morton, Greg, "MSP430 Competitive Benchmarking", Application Report SLAA205A, Texas Instruments.