DSP LAB COURSE EXPERIMENTS BASED ON THE MOTOROLA DSP56K PROCESSOR

David Báez-López[1], Eduardo Trueba-Marcos[2], Juan Manuel Ramírez[1], and Eduardo Jiménez-López[1]

Abstract—*This paper describes a set of real time experiments for a Digital Signal Processing undergraduate course. These experiments are based upon the Motorola 56K DSP processor. The main purpose of the experiments describes is to allow students to appreciate the potential of DSP processors to implement digital filters. The experiments described range from function generation to IIR and FIR filter design. The code is available to interested instructors as shareware files. Data for the digital filters designed and implemented in these experiments is obtained from the MATLAB Signals Toolbox and from other design packages available.*

Index Terms—Digital signal processing, DSP 56K, filter design,

INTRODUCTION

Nowadays, digital signal processors have been used in a wide range of applications, from sophisticated communication systems, instrumentation, uses in control, entertainment, to some home equipment such as speaker-phones.

Because of low-cost development tools for DSP processors are available, in the near future they will be used in many more applications. Furthermore, DSP-based systems can be readily reprogrammed for a different application.

For these reasons, among others, the undergraduate electrical engineering curriculum must include a Digital Signal Processing course [1,2] with a heavy emphasis on hands-on DSP processor usage. Unfortunately, the already loaded EE curriculum does not allow students to learn the fundamentals of digital filter design and their limitations and circuit realizations and to grasp every detail of a DSP processor in a single semester course with its many advantages. Thus, we have implemented a set of 17 experiments in filter design and realization that allow students to see the many advantages of such devices [3]. Some examples will show students some features (advantages and disadvantages) of the digital signal processing experiments.

TABLE 1
BENCHMARKS FOR THE DSP56K

Benchmark	No. of Cycles	No. of Algorithm Multiplies
Real Multiply	3	1
N Real Multiplies	2N	N
Real Update	4	1
N Real Update	2	1
N Term Real Convolution (FIR)	N	N
N Term Real *Complex Convolution	2N	N
Complex Multiply	6	4
N Complex Multiplies	4N	N
Complex Update	7	4
N Complex Updates	4N	4N
N Term Complex Convolution (FIR)	4N	4N
Nth Order Power Series	2N	2N
2^{nd} Order Real Biquad Filter	7	4
N Cascaded 2^{nd} Order Biquads	4N	4N
N Radix Two FFT Butterflies	6N	4N

THE DSP56K PROCESSOR FAMILY

The DSP 56K family as Motorola's series of 24 bit general-purpose digital signal processors (DSP´s) [4]. The family architecture features a central processing module that is common to all family members. The DSP 56K family is designed to execute commonly used DSP benchmarks in a minimum time for a single-multiplier architecture. Table 1 shows a list of benchmarks with the number of instruction cycles a DSP56K uses compared to the number of multiplies the algorithm requires. Further information on the DSP56k family can be obtained from Motorola's website at www.motorola.com.

[1]Departamento de Ingeniería Electrónica, Universidad de las Américas-Puebla, Cholula, Puebla, México, dbaeziec@mail.udlap.mx.
[2]Xerox Corp, Puebla, México.

The 24 bits word length data path provides 144 dB of dynamic range, while intermediate results are held in 56-bit accumulators thus giving a 336 dB dynamic range. These facts give the DSP 56K a high precision. In addition, there is high degree of parallelism that allows a second order IIR filter to be executed in only four cycles, the theoretical minimum for single-multiplier architecture. Finally as a CMOS part, the DSP56K is inherently a very low power device.

The next section will show how the DSP56K has been used in a set of DSP experiments for a Digital signal processing course. The experiments are related to IIR and FIR filter design and some related functions such as function generation and Nyquist theorem. There are 17 experiments available.

The list of experiments is the following:

1. Sum of two numbers.
2. Receive and transmit signals.
3. Aliasing (Nyquist theorem)
4. Sine function generation
5. Square function generation
6. Ramp function generation
7. Derivative of a signal
8. Second order IIR low pass filter (Direct form realization)
9. Fourth order IIR low pass filter (as a cascade of two second order stages)
10. Second order IIR high pass filter
11. Second order IIR band reject filter
12. Second order IIR band pass filter
13. Fourth order IIR band pass filter (cascade of two second order stages)
14. Third order IIR high pass filter
15. Fifth order FIR low pass filter
16. Twentieth order FIR band pass filter
17. Twenty fifth order FIR low pass filter

In addition, subroutines to initialize the DSP56K SDK are included.

The DSP56K used forms part of an SDK which is available from Motorola and several other OEM's. The used DSP56K SDK is shown in Fig. 1 next to an oscilloscope.

FILTER DESIGN

There exist several tools available for digital filter design. Among the most popular ones available is the MATLAB Signals toolbox [5]. Another tools that we have used are winfilters and Mfilters both developed by Baez group at Universidad de las Americas. As described in Reference 7, Mfilters is a friendly interface for the MATLAB Signals Toolbox. Regardless of the package used, the resulting filter coefficients will be the same.

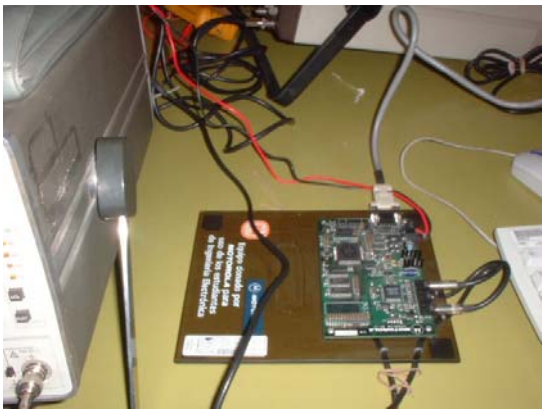


FIGURE 1
DSP56K SDK USED IN THE EXPERIMENTS.

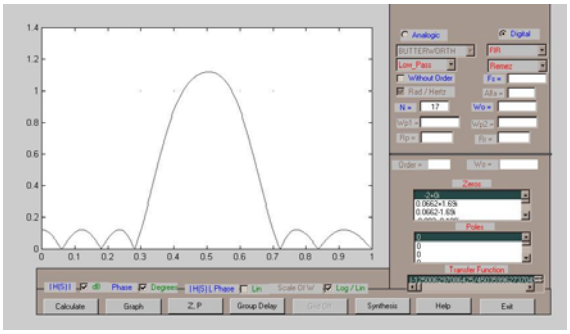
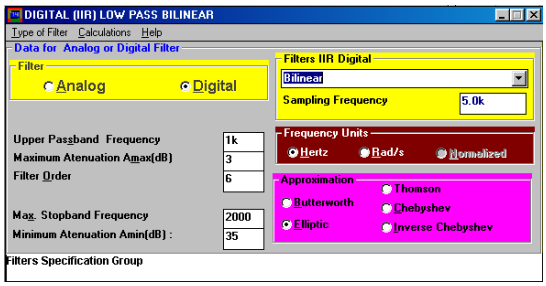


FIGURE 2
INPUT/OUTPUT WINDOW FOR MFILTERS. A BANDPASS FIR FILTER IS DESIGNED HERE.

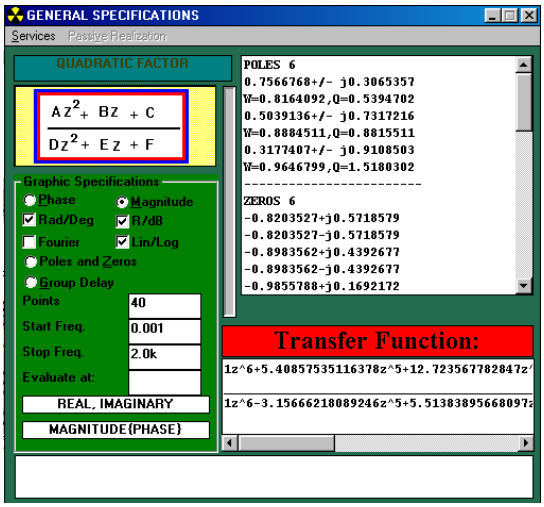
Figs. 2 and 3 show typical input/output windows for Mfilters and Winfilters, respectively [6,7]. While Mfilters allows the input and output data to be displayed in the same window, Winfilters does requires separate windows for input and output data.

EXAMPLES

In this section we present some examples of the results obtained in the lab experiments designed and based in the DSP56K processor. Our experiments include Nyquist theorem (sampling theorem) where students can check the problem of aliasing and how to avoid it, an IIR lowpass Butterworth filter, an IIR bandreject filter, and an FIR lowpass filter designed using the windowing technique.



(a)



(b)

FIGURE 3
(a)INPUT AND (b) OUTPUT WINDOWS FOR WINFILTERS.

Aliasing (Nyquist theorem)

In order to emphasize the importance of an adequate sampling, we present an example of Nyquist theorem and observe aliasing because we did not comply with it.
We apply a sinewave signal to the SDK and make no processing on it. We just pass it through the SDK. The sampling frequency in this example has been fixed at 2 KHz whereas the input signal frequency is 3125 Hz. The input signal frequency is higher than the Nyquist frequency. Fig. 4 shows the input and output waveforms. As can be seen, the output signal frequency appears to be 176.1 Hz. By changing the input waveform frequency, students can see how aliasing is not present at frequencies below 1 KHz but it starts to appear at frequencies above 1 KHz.

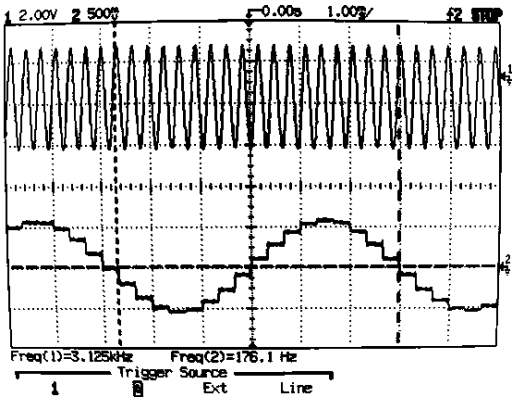


FIGURE 4
INPUT AND OUTPUT WAVEFORMS FOR THE ALIASING EXPERIMENT.

IIR lowpass filter with the DSP56K

As an example of an IIR filter we show a second order lowpass filter. The magnitude response is shown in Fig. 5 where we have used the HP3561A spectrum analyzer. We can clearly see the lowpass nature of our filter. With this example students can rapidly appreciate the response characteristic of a filter.

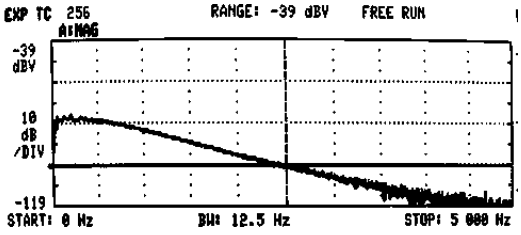


FIGURE 5
MAGNITUDE RESPONSE OF THE IIR LOWPASS FILTER.

IIR bandreject (notch) filter with the DSP56K

Our third example shows an IIR bandreject filter. The magnitude response is shown in Fig. 6. We can clearly see how the bandreject characteristic is shown in the plot. A further variation on this project could be changing the sampling frequency to observe how the notch frequency varies.

FIR Filter with the DSP56K

As our example for FIR filters, we have a 25th order low pass filter with 2 KHz cutoff frequency. Fig. 7 shows the resulting frequency response. A purpose of this experiment is that students can compare the order of a lowpass IIR filter, designed above, with the much higher order of an FIR filter.

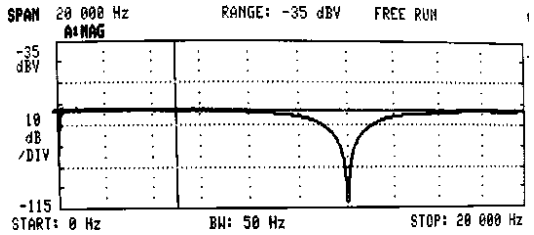


FIGURE 6
MAGNITUDE RESPONSE FOR THE IIR BANDREJECT FILTER.

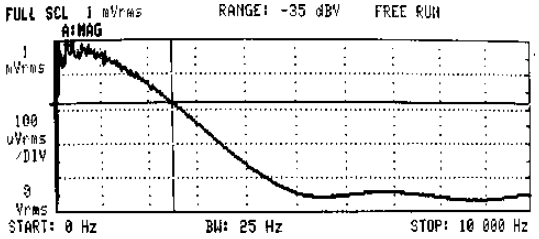


FIGURE 7
OUTPUT SIGNAL FOR THE FIR LOWPASS FILTER.

CONCLUSIONS

We have presented a set of experiments which will help students to quickly understand some of the topics in a typical Digital Signal Processing course. The experiments are based in the Motorola DSP56K processor. Several examples show how easy is to obtain meaningful results for these experiments. There is a set of 17 experiments available to students and they cover basic material on filter design, both IIR and FIR filters.

One important point in using these experiments in a course is that students gain a better understanding of the theoretical concepts covered in the lectures. This goal has been achieved when compared to the time when no lab experiments were avilable. In addition, students learn to program a DSP chip and we have chosen the Motorola DSP56K among the many DSP processors available.

The set of experiments is available to interested instructors at no charge.

REFERENCES

[1] Antoniou, A. “Digital Filters, Analysis, Design, and Applications”, McGraw-Hill Book Co., New York, 1993.
[2] Chassaing, R., “Digital Signal Processing, Laboratory Experiments Using C and the TMS320C31 SDK”, Wiley Interscience, New York, 1998.
[3] E. Trueba-Marcos, Lab experiments in the DSP56K processor, B.S. Thesis, Universidad de las Americas-Puebla, México, 1998.
[4] DSP56K Manual, Motorola, Mesa, 1998.
[5] MATLAB, The language of Technical Computation, The MathWorks Inc., Natick, MA, 1996.
[6] Baez-Lopez, D et al., Multimedia Based Analog and Digital Filter Design, Computer Applications in Engineering Education New York, pp.1-8, 1998.
[7] D. Baez-Lopez et al., MATLAB based analog and digital filter design, Electrosoft, Lemnos, Greece, 2001.