Low-Cost Personal DSP Training Station based on the TI C3x DSK

Armando B. Barreto[1] and Cesar D. Aguilar
Electrical and Computer Engineering
Florida International University, CEAS-3942
Miami, FL, 33199
armando@dsplab1.eng.fiu.edu
(305) 348-3711

Abstract: This paper outlines the configuration of a personal DSP Training Station around a TMS320C3x DSK Starting Kit and a personal computer running Windows ® and fitted with a sound card. The substitution of expensive laboratory equipment, such as signal and noise generators, oscilloscope and spectrum analyzer makes this a cost-effective alternative for the setup of a learning station to explore the development of real-time DSP implementations.

I. INTRODUCTION: THE NEED FOR A VIABLE REAL-TIME DSP TRAINING STATION

The last few years have brought a precipitated expansion of the application of Digital Signal Processing (DSP). Theories such as digital filtering, modulation, data compression , etc., that used to be utilized only for highly sophisticated systems, have found their way into an increasing number of industrial and end-consumer applications. Accordingly, there is an increasing demand for engineering professionals capable of understanding DSP algorithms and, furthermore, of implementing them on specialized DSP processors to offer effective real-time solutions to the design problems found in the diverse areas mentioned before. It should be noted that an individual capable of efficiently implementing a DSP algorithm for the real-time solution of a practical problem must posses skills in at least three different areas:
- Continuous and Discrete Time Systems Theory
- Microprocessor Programming and Hardware Interfacing
- Software Engineering Concepts

Interestingly, the only way to effectively acquire these skills and develop an understanding of the interplay between them towards the actual implementation of a real-time solution is by interaction with the developing tools used for real-time DSP implementations.

The need for providing hands-on experience to prospective DSP system designers in industry has been addressed by Texas Instruments since the last decade through the DSP Evaluation Modules (EVMs). The availability of these EVMs was appreciated by some colleges and universities that used them to provide real-time DSP implementation instruction to their students as part of their curricula.

Unfortunately, in both scenarios: the practicing engineer learning how to program and interface DSP chips and the college student being taught about DSP implementations, a number of additional costly devices used to be needed to complete the learning experience.

From the perspective of instrumental requirements, means of generating inputs with certain controlled characteristics for the DSP system under development were needed. Additionally, testing instruments were required to verify the expected performance of the DSP system under development.

One other factor that used to make real-time DSP training appealing to a fairly limited audience was the level of “specialization” required to operate and interact with both the development system and the associated instrumentation. Just a few years ago only some electrical engineers were familiar enough with the workstations that were used as hosts for these development systems. The current familiarity of virtually everyone with personal computer systems is, clearly, a much better context for the broad development of real-time DSP expertise among practicing engineers and college students.

II. FACTORS THAT MAKE POSSIBLE THE PERSONAL DSP TRAINING STATION

With the recent development by Texas Instruments of the economic Digital Signal Processing Kits (DSK’s) for the TMS320C50 and TMS320C30 processors, experimenting with real-time DSP applications is virtually within anyone’s reach. However, the significance of this has been magnified by two other concurrent developments: The incorporation of computer systems in our culture and the emergence of the internet as a powerful medium of information exchange, available to virtually all engineering students and practicing engineers.

[1] This work was made possible by US Dept. of Education grant MSIP-P120A60016 and matching support from Texas Instruments, Inc.

As mentioned before, computers have shifted from the exclusive domain of engineers and scientist to become more of a household item. This has enabled the assimilation of personal computer systems and the capability to interact with them into contemporary culture. So, at this point it can safely be expected that most upper-division students in electrical and computer engineering will have enough familiarity with some of the concepts involved in setting up and getting started programming a DSP development system. It is reasonable, in fact, to assume that the engineering student or the practicing engineer will have access to a personal computer system where he or she will be able to set up the DSP development system.

The popularity of the internet as a means of information exchange is significant to the development of a personal DSP training station in that it provides (instantaneous) access to a continuously growing body of knowledge and resources. Specifically, the internet has become the prime vehicle for the dissemination of "freeware" or "shareware" for a wide variety of applications. Some of those applications, in fact, can help substitute the acquisition of expensive equipment towards the development of the personal DSP training station. Furthermore, the internet has magnified the sharing of practical knowledge among individuals interested in real-time DSP that was started through the Bulletin Board Services (BBS's) and it has complemented that interaction with related news groups and World Wide Web pages.

It is in this context that the setup of an economical, personal DSP training station is proposed. In summary, the setup of the training station requires the following components:

- A TMS320C3x DSK kit
- A reasonably powerful personal computer (e.g., Pentium 90 MHz, or better).
- A sound card in the computer (e. g., SoundBlaster 16)
- The Windows® operating system
- Internet access (direct or indirect).

It should be noted that is very likely that the interested individuals may already have availability of most of the items listed above.

For some more advance analysis of the results obtained experimenting with the Real-Time DSP training station the student edition of analysis packages such as Matlab ®, may be a valuable addition.

III. THE C3xDSK

The Texas Instruments TMS320C3x DSP Starter Kit (DSK) comprises all the basic elements to learn and practice programming the TMS320C31 DSP.

The C3x DSK board includes a 50 MHz TMS320C31 DSP and a TLC32040 Analog Interface Chip (AIC), which provides 14-bit Analog-to-Digital (A/D) and Digital-to-Analog (D/A) conversions for one signal, at adjustable sampling rates. The RCA jacks for analog input and analog output in the board connect to the AIC, which , in turn, is connected to the serial port of the C31 DSP. In addition, The C3x DSK has a host interface port that connects to a parallel port in the host PC. A "communications kernel" running in the C31 DSP manages the host interface, enabling the host to inspect and modify memory and start or stop execution of program segments (through the debugger).

The software provided in the C3x DSK includes a DOS assembler that will convert a source ASCII file written using the appropriate mnemonics for the C3x DSP processors into a loadable/executable file of type ".dsk"

The ".dsk" files can be loaded to DSP memory through the DSK3D debugger included with the kit. In addition the debugger will allow the user to inspect and modify all the registers in the DSP, as well as the DSP memory. DSK3D also implements most of the functions found in microprocessor or DSP debuggers: Start and stop of execution, establishment of breakpoints and watches, single-step tracing, etc.

The complete description of the hardware and software included in the C3x DSK is found in the "TMS320C3x DSP Starter Kit User's Guide" [1], from Texas Instruments (Literature number SPRU163).

IV. INTERRUPT-DRIVEN PROCESSING IN THE C3xDSK.

Real-time processing of an analog signal implies that such signal will be sampled at a certain sampling frequency, f_s, and that the DSP system will be able to perform all the necessary computations to provide an upgrade in the output signal before the next input signal is collected. This means that the total time required for the computation of the next output sample must be less than $T = 1/f_s$, the sampling interval.

In the C3x DSK the timing for sampling the signal is set by programming parameters in the AIC and setting the corresponding DSP serial port to cause a periodic interrupt, every T units of time. So, the DSK will ordinarily be sharing its processing time between (at least) two (user) programs: a "main" program and the Interrupt

Service Routine (ISR) that responds to the interrupt request issued periodically for the AIC. Typically the real-time processing tasks, as explained in the previous paragraph, will be carried out in the ISR, while the main program will usually involve an initialization phase, followed by some type of endless loop where some non-real time tasks, such as user interface, etc. take place, having to yield periodically to the execution of the ISR. Figure 1 illustrates this concept. Much more detailed information about this concept can be found in the TMS320C3x User's Guide [2] (Lit. no. SPRU031D) or in the book by Chassaing [3].

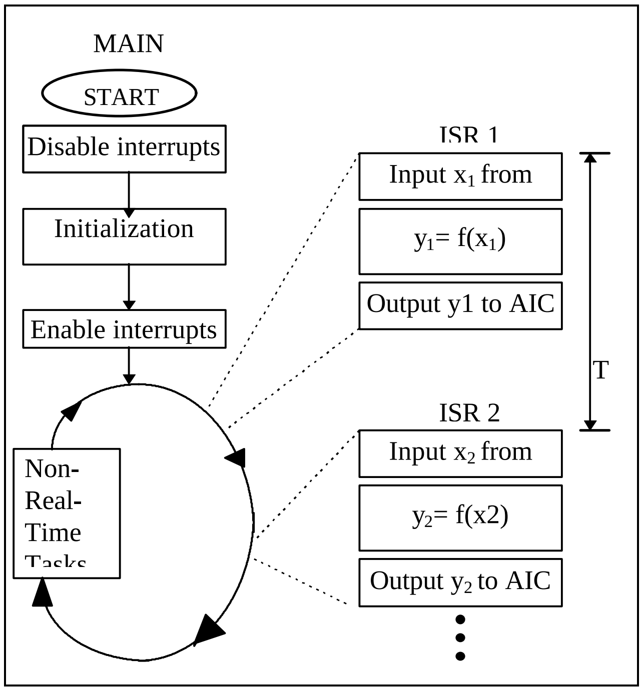


Figure 1. *Interrupt -driven real-time processing*

This brief explanation of interrupt-driven processing has been include here just to help understand the input substitutions that are proposed in one of the following sections.

V. EXPERIMENTING IN REAL-TIME WITH THE C3x DSK

While the C3x DSK package itself provides the necessary elements to program the C31 DSP in the DSK board, the effective impact that a given algorithm programmed in the DSP will have on a signal processed by it requires the application of an input signal with known, controlled characteristics and the observation of the characteristics in the corresponding output signal.

Ordinarily, the generation of the an input signal with controlled properties and the monitoring of the output signal with respect to several of its characteristics calls for the use of testing equipment specifically designed for those purposes. Whenever available, specialized testing equipment will continue to be the best choice for signal generation and monitoring. On the other hand, the following sections outline ways in which these needs may be met, at least at the level required for verification of algorithm performance for instructional purposes, with much less expensive and widely available means.

VI. INPUT SUBSTITUTION

In a standard laboratory setting test signals would be produced with signal and function generators of various classes. In some cases a random noise generator may be available to provide an input with approximately flat magnitude spectrum. In this way, observation of the output spectrum will directly display the magnitude response of the DSP implementation under test. A number of alternatives exist for the substitution of those pieces of equipment:

VI.1 Internal input substitution

A variety of periodic signals (sinusoidals, trains of pulses or impulses) can be substituted internally by the DSP if a Look-Up Table (LUT) is set up with the values of one cycle of the signal in question. If such set of values is downloaded with the program to a buffer aligned to a boundary for a block size larger than the actual length of the LUT, then circular addressing can be employed to read one sample from the LUT instead of reading the input sample from the AIC, in each ISR. This will also require that the block size register (BK) be set to the length of the LUT. For a LUT of length N, the simulated frequency of the periodic signal effectively used as input will be $f_p = f_s/N$.

The generation of a signal with flat magnitude spectrum to directly test the magnitude response of an algorithm would require a random noise generator. A Pseudo-Random Binary Noise (PRBN) generator may be built into the implementation under test, so that the DSP will create one PRBN sample and use it as the most current input to the algorithm, instead of retrieving a real Analog-to-Digital converter result from the AIC.

A simple algorithm for PRBN generation consists of the bit-wise (modulo 2) addition of pre-selected bit locations (e.g., b_{17}, b_{28}, b_{30}, b_{31})in a 32-bit word that is iteratively shifted left, one bit at a time. The resulting bit determines if the PRBN value for the iteration is a "high" or a "low" level and it is copied into the LSB location of the generator word, opened up by the left shift applied to it. The 32-bit word loaded to start the process is the "seed" value for the PRBN iterations. Figure 2 shows the basic concept. Actual C3x code for implementing this process can be found in the book by Chassaing [3].

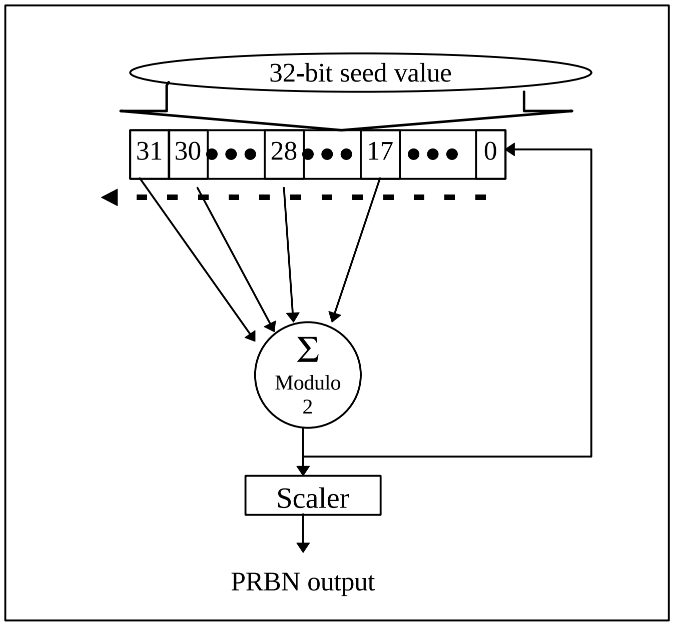


Figure 2. *Block diagram for the generation of Pseudo-Random Binary Noise (PRBN).*

VI.2 External input substitution

For several applications, such as the verification of digital filters, a number of external devices, other than signal generators may be used as signal sources. Typically, the signal from the headphone output of an inexpensive personal tape playback unit ("Walkman" type of device), or CD player may be suitable for the verification of signal processing algorithms for audio and voice applications. This mechanism enables the student to really appreciate the effect of the processing, (e.g., filtering) on familiar sounds (e.g., songs) which can, on the other hand, be played again and again, as required. This form of verification requires that the output of the DSK be played as a sound through magnified speakers (Section VII.2).

Another form of external input substitution which takes advantage of the assumed host computer capabilities is the use of the sound card for playing a digital audio file. To use this form of signal generation the "output" connector of the sound card is patched to the "input'' connector of the DSK board. This approach may provide additional advantages if there is a mechanism to build digital audio files with specific signals. For example, Matlab® is capable of converting an array of numbers determined by assignment or computation into a "Wave" (.wav) file, which can then be played out through the soundcard, using standard applications for that purpose. The "Sound Recorder"® application in Windows® can be used for this. Other more elaborate programs, such as "GoldWave"®, which can be retrieved through the World-Wide Web as shareware (http://www.goldwave.com), are capable of playing this files in a loop, thus providing a continuous signal source.

VII. MONITOR SUBSTITUTION

The ultimate verification of a real-time DSP implementation takes place by matching the expected characteristics of an output signal with those measured from the actual DSK board. In most cases the verification calls for the evaluation of certain performance measures through instrumentation capable of revealing the effective values of those figures of merit, in real time. In some specific cases, particularly audio signals, the ultimate verification is also subjected to the appreciation of the observer, which is obtained by "playing out" the output through speakers, allowing a qualitative assessment of the algorithm performance. Fortunately there are also affordable means to perform both the quantitative and the qualitative types of verification of real-time DSP implementations in the C3x DSK.

VII.1 Substitution of quantitative monitoring

Typically performance of the algorithm can be measured by the degree of observed modification in the time and frequency characteristics of the output signal, for a known input signal. In a traditional laboratory setting these features of the output signal would be measured with an oscilloscope and a spectrum analyzer, respectively. While the oscilloscopes were initially analog in nature, the most recent types, as well as the spectrum analyzers, are essentially digital systems. Just a few years ago the functions involved in the implementation of digital oscilloscopes and spectrum analyzers required the use of dedicated processors. However, most of the computers used these days as DSK hosts have the capability of carrying out the computations and functions required for the display of the output signal in time or in frequency, simultaneously with the performance of the DSP processing by the DSK board.

The missing link that prevents ordinary computers from serving as monitors in the time or frequency domain is the necessary Analog-to-Digital and Digital-to-Analog interfaces to convert the continuous time inputs and outputs to / from the DSK board. With the recent advent of multimedia systems for personal computers the "sound boards", such as Creative Labs' SoundBlaster 16 ® have become an almost standard module in the ubiquitous PCs . This sudden popularity has prompted several groups of programmers to develop software applications for the PC that take advantage of the sound board as a two-channel data acquisition board that enables the PC to capture one or two analog signals as discrete-time sequences. Once the signals have been digitized using the sound board, the corresponding sequences may be displayed to the PC user, emulating a digital oscilloscope. Alternatively, these sequences may be further processed in the PC, by means of an FFT routine, displaying the resulting magnitude spectrum, as in a spectrum analyzer.

There are two free software applications that we have found useful in the context of verification of DSP algorithms implemented in the DSK:

“The Scoper”, by Mike Ferris, for real-time display of one analog signal connected to the input of the sound board. This is a freeware program that can be downloaded without charge from
http:// www.mulberry.com/~mferris/

The Scoper is a Windows® application that provides simple display of the signal with six levels of attenuation and seven different time scales. Although the display capabilities are much more limited than those of a real scope, the display is normally of sufficient quality to verify the performance of the algorithm (e. g., using a manual sinusoidal sweep as input).

“The Spectrum Analyzer” by Philip VanBaren is a fairly complete program capable of using several types of sound boards (SoundBlaster, ProAudio Spectrum, etc.) to acquire one or two analog channels and process them to display a single spectrum (individual channel or left-right difference). This program provides numerous options (sampling rate, FFT size, windowing function, number and type of FFT averaged, etc.) The program also has a number of display management functions that make it very useful. For example, the spectrum display can be frozen and printed or saved as an ASCII file. In this way the spectrum values can be imported into other analysis and display programs, such as Matlab®, where improved formatting is possible. Both the DOS executable and the source code can be retrieved from the URL below, under the name “freq51”. (The copying policy for this program is the GNU GPL, General Program License):
http://bul.eecs.umich.edu/~phillipv/signal/

Other sound board commercial programs exist that can be used for monitoring DSK performance in both the time and the frequency domain. For example “GoldWave” by Chris Craig is offered for trial as shareware from:
http://www.goldwave.com/

This program is more versatile than the two above, being capable of displaying both the time and frequency domain representation of both channels sampled by the soundcard. The main aim of this program is the use of the sound bord for the recording and reproduction of sound (wave) files. A small registration fee is requested from the user.

In some instances, as mentioned for the Spectrum Analyzer, it is desirable to perform a more detailed analysis of the DSK output. That would be the case in which the response of the algorithm in the DSK to an impulse or rectangular pulse (step) needs to be observed in detail. In these cases the desired input signal could be generated internally to the DSK, by accessing an appropriate Look-up Table with circular addressing and the output of the DSK could be connected to the input of the soundboard. Then, using applications such as “Sound Recorder”® the response can be recorded as a wave file. This wave file, in turn, can be converted by programs such as Matlab® into a vector of samples vs. time which allows detailed measurements on the output waveform.

VII.2 Qualitative monitoring

In the specific case of audio or speech processing applications the validity of the DSP algorithm implemented in the DSK can be qualitatively evaluated by the assessment of the output audio signal played through amplified speakers commonly included in the PC multimedia packages.

In addition, availability of a sound card and an associated application, such as Window’s “Sound Recorder” enables the storage and playback of results obtained from the same input, for different parameters of the DSP algorithm, for qualitative comparison purposes.

VIII. THE PERSONAL DSP TRAINING STATION

Figure 3 shows a block diagram indicating how all the options that have been outlined for signal generation and monitoring can be combined to provide a full set of resources for real-time DSP experimentation. The block representing Matlab® is drawn with dashed lines because the basic functionality of the DSP training station can still be implemented without it, although its incorporation certainly adds to the capabilities of the training station. Furthermore, its use provides very good support to the algorithm design phase of the DSP experimentation, including commands for one-step digital filter design and extensive features for interactive algorithm simulation and off-line verification.

IX. EXAMPLE

Figure 4 is a screen capture obtained from the use of “The Spectrum Analyzer”, with the output of a C3x DSK connected to the microphone input of the SoundBlaster16® card. The parameters of the sound card have been adjusted to f_s = 5000 Hz, FFT length = 1024 samples, and the display corresponds to the (uniform) average of 420 FFTs. The DSK was programmed to implement a 6th order Chebyshev-I Low-Pass Filter with a cutoff frequency at 1KHz (DSK Sampling rate was 20 KHz), with 1 dB ripple. The filter was fed with Pseudo-Random Binary Noise samples generated in every ISR by the algorithm outlined in Section VI.1 . The display of averaged spectra from “The Spectrum Analyzer” allows the verification of the cutoff frequency and the passband ripple characteristic of the Chebyshev type I approximation function.

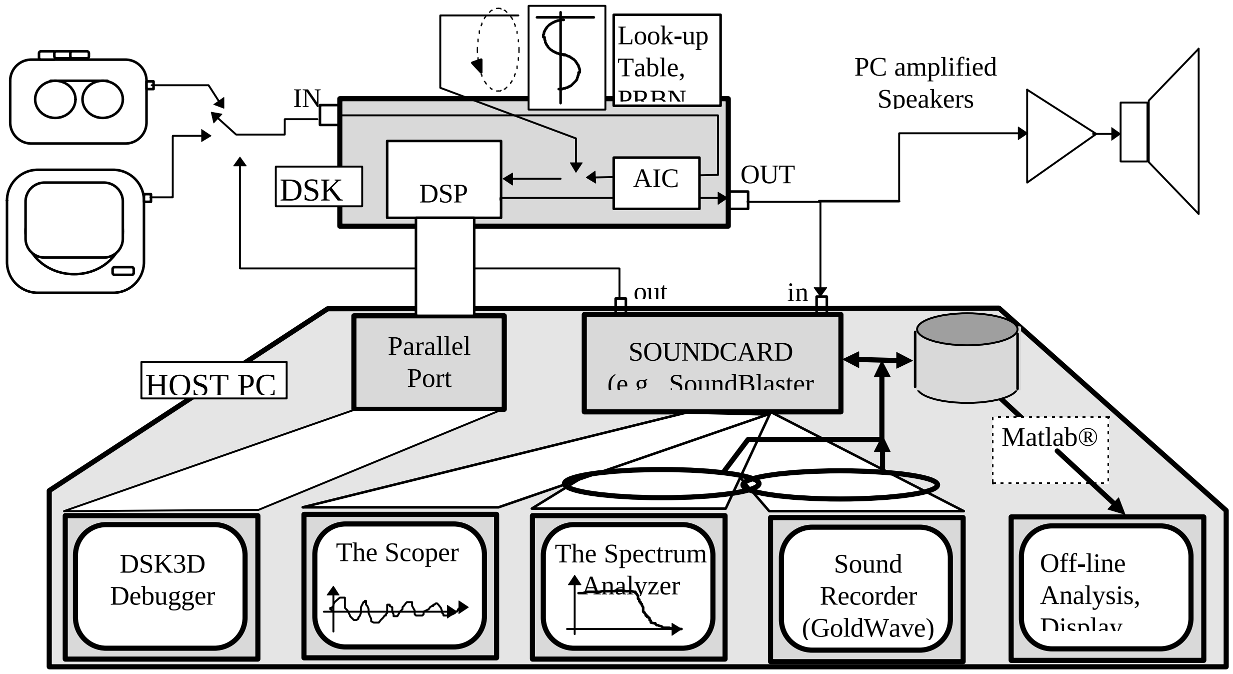
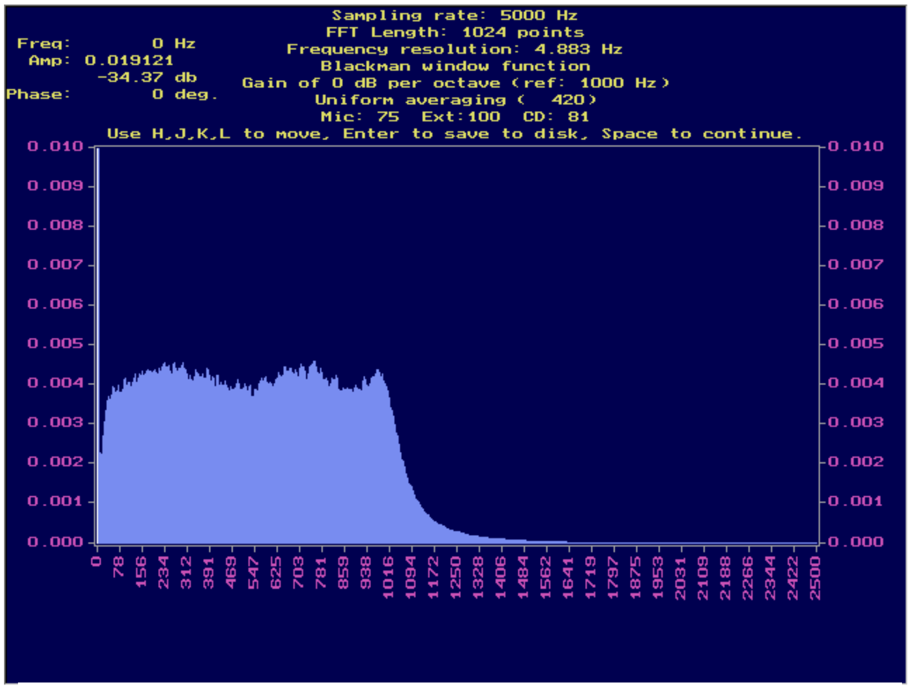


Figure 3. *Block Diagram illustrating the interconnections of the modules in the Personal DSP Training Station*



```
                         Sampling rate: 5000 Hz
                        FFT Length: 1024 points
Freq:        0 Hz    Frequency resolution: 4.883 Hz
 Amp: 0.019121          Blackman window function
       -34.37 db    Gain of 0 dB per octave (ref: 1000 Hz)
Phase:       0 deg.      Uniform averaging (   420)
                         Mic: 75  Ext:100  CD: 81
      Use H,J,K,L to move, Enter to save to disk, Space to continue.
```

Figure 4. *Sample result of the spectrum displayed by "The Spectrum Analyzer". This figure shows the averaged spectrum for a 6th order Low-Pass Chebyshev Filter implementation using the PRBN input substitution (Section VI. 2)*

X. REFERENCES

[1] "TMS320C3x DSP Starter Kit Users Guide", Texas Instruments Inc., 1996.

[2] "TMS320C3x Users Guide", Texas Instruments Inc., 1994.

[3] " Digital Signal Processing with C and the TMS320C30", Rulph Chassaing, Wiley Interscience, 1992.