GRASPING THE POTENTIAL OF DIGITAL SIGNAL PROCESSING THROUGH REAL-TIME DSP LABORATORY EXPERIMENTS

Aníbal J. S. Ferreira, Francisco J. O. Restivo

Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia Electrotécnica e Computadores
Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
ajf@inescporto.pt, fjr@fe.up.pt

ABSTRACT

A new DSP laboratory course has been included in the Electrical and Computer Engineering curriculum at the Faculdade de Engenharia da Universidade do Porto, in Portugal, since the school year of 1999/2000. This paper addresses the context and motivation underlying this new course, outlines its structure and methodology, highlights the design and goals of all DSP experiments currently proposed for the 13 weeks of the semester, and reports on the receptivity students have expressed to this elective course. The course is based on the TI C31 Starter Kit and tries to combine full use of its resources with a representative diversity of efficient digital signal processing techniques and associated application scenarios. A perspective is also given on current plans to reinforce DSP expertise at the graduate level.

1. CONTEXT AND MOTIVATION

At the Faculdade de Engenharia da Universidade do Porto (http://www.fe.up.pt) the Portuguese *Licenciado* degree is earned at the end of a 5-year course program organized in 10 semesters. This degree corresponds to a level between the American Bachelor degree and the Master degree.

During the first three years of the Electrical and Computer Engineering program (ECE), students are exposed to courses that are basic to digital signal processing such as digital systems, signal theory, circuit theory, microprocessors, probability and statistics, systems theory, algorithms and data structures.

In particular, these courses expose students to the concepts of Laplace transform, Fourier analysis, modulation sampling and filtering, Z transform, random processes and noise, and programming.

Students progressing to the 3rd. year of the ECE program have to chose a specialization branch that constrains the mandatory and elective courses they can take during the last three years of the ECE program. However, it is only during the 1st. semester of the 4th. year of the ECE program that students choosing the Electronics, Computer and Telecommunication Systems (TEC) specialization branch[1], are exposed to a mandatory course (Digital Signal Processing -EEC4162) that focuses on basic concepts of digital signal processing, namely:

- discrete signals and systems,
- sampling and reconstruction of analog signals,
- linear time invariant systems,
- structures for the realization of LTI systems,
- FIR and IIR filter design,
- finite word length effects,
- decimation and interpolation,
- the discrete Fourier transform,
- overlap-add and overlap-save methods for efficient FIR filtering,
- response of LTI systems to random discrete signals,
- the FFT and its implementation.

EEC4162 is mostly based on the Oppenheim, Schafer and Buck text book [1] and is structured on a 3-hour class per week, where theory is explained and illustrative problems are solved, and also on a 2-hour class per week where students are invited to solve problems and to discuss them with

THIS WORK WAS SUPPORTED BY THE PORTUGUESE FCT UNDER PROJECT POSI/CPS/34178, BY THE DEPARTMENT OF ECE AT THE FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO, PORTUGAL, AND HAS ALSO RECEIVED THE SPECIAL SUPPORT OF THE T.I. EUROPEAN UNIVERSITY PROGRAMME

The first author is also affiliated with INESC Porto (http://www.inescporto.pt, http://www.inescporto.pt/gaudio).

[1]The other two possible specialization branches are Energy Systems, and Industrial Automation, Production and Electronics.

the teacher. It happens frequently in this type of class that students are active only when compelled to by the teacher.

Individual homework assignments are also given by the middle and by the end of the semester. Although EEC4162 illustrates frequently practical examples and applications using Matlab, in order to motivate students to a clear understanding of essential DSP concepts such as aliasing, it happens also that some students do not capture the substantive meaning of formulae such as the autocorrelation and the spectral density Fourier pair, the implications and usefulness of important signal modifications such as digital interpolation, or the practical interpretation of the relation between the normalized frequency axis describing the frequency response of a digital filter, and the associated analog frequency when the digital filter replaces the analog filter.

Less often, we face spontaneous and almost chocking confessions of graduating students saying that they never understood very well what DSP was all about.

Despite the clear increasing importance of digital signal processing in all application areas, namely telecommunications, biomedical, control, multimedia, and consumer electronics, the conclusion arosed that a pencil and paper based approach to digital signal processing learning, together with a few simulation examples on Matlab, was not enough for students to capture the advantage and potential of digital signal processing in various engineering applications, and further are not inspired to explore DSP based solutions to specific problems.

It was in this context that the idea came about of a new course taking EEC4162 as a pre-requisite and focusing on hands-on practical DSP laboratory experiments using real world signals and placing the emphasis on real-time processing constraints. The new course was named Real-Time Digital Signal Processing (EEC5274) and was admitted in the ECE curriculum as an elective course of the 2nd. semester (4th. ECE year) to all TEC students.

The enrollment per semester in the course has been 16 students in 1999, 32 students in 2000 and 22 students in 2001, which represents about 20-30% of all TEC students (each TEC student has to pick 2 electives from an universe of about 10 different electives).

2. RATIONALE, METHODOLOGY AND STRUCTURE

The rationale of the EEC5274 course (Real-Time Digital Signal Processing) is that the intuition of DSP concepts, the understanding and mastering of DSP realization issues, and the capability of solving problems using DSP technology, are better understood by means of practical experiments configured in such as way as to

- demonstrate by example the advantages of DSP,
- face the student with challenges in DSP design and implementation,
- illustrate the pertinence and simplicity of a DSP based approach to a representative diversity of application scenarios.

The Texas Instruments *C31 Starter Kit* has been chosen as the basic DSP platform for all laboratory experiments, essentially for four reasons:

1. the platform has the necessary hardware resources making possible reasonably complex laboratory experiments and is based on the T.I. TMS320C31 floating point processor. In turn, this DSP is very convenient because its assembly instruction set is rich and yet simple enought to understand, which makes manual assembly programming viable after a short learning period. On the other hand, the instruction set also permits fixed point implementation as well as floating point implementation of the same algorithms which eases a practical evaluation of the issues involved,

2. the platform includes an assembler environment permitting assembly programming, and includes a windows oriented debugger that we have replaced by the very convenient and truly windows based Code Explorer (essentially because it allows graphic visualization of memory contents), originally created by the Go-DSP company (but that is unfortunately not available any more since this company has been acquired by T.I. in 1998 and became its subsidiary),

3. the platform is well supported by many demonstration code examples and from the point of view of the teacher, a significant help is provided by the T.I. *C31 Teaching Kit* [2] that is based on the same platform, as well as by existing literature dedicated to the topic of C31 based laboratory experiments [3, 4].

During the 13 weeks of the semester, students enrolled in the EEC5274 course attend two classes per week, one of them with a duration of 1h:30m and devoted to the theoretical presentation and multimedia demonstration of concepts, and the other class, with a duration of 2h:30m, takes place in a laboratory equipped with PCs, power suplies, signal generators, multimeters, frequencymeters, oscilloscopes, microphones and active loudspeakers, and is devoted to the design, realization, demonstration and assessment of algorithms running in real-time on the *C31 Starter Kit.*

Since the *C31 Starter Kit* represents a complete development environment, students are allowed and encouraged to keep the *Kit* with them between classes so that they can prepare at home for their laboratory work. It is obviously

not assumed that students can replicate the laboratory conditions at home (namely specific equipment such as oscilloscope and signal generator), however, laboratory experiments are designed such that most of the work can be effectively prepared at home and, in most cases, even assessed by means of a microphone, a loudspeaker or a (easily available) software application able to generate and analyzing signals and running of the host PC.

Not only students prepare their lab work at home but often they also explore beyond the strict realization goals proposed for each laboratory experiment, which reveals that motivation and even a sense of self-learning can build-up around DSP and this is perfectly in-line with the rationale of EEC5274.

From the point of view of the concepts covered in both theoretical and laboratory classes, the course is divided into two parts, each one lasting approximately six weeks. The first part focuses on the understanding and use of the development environment of the *C31 Starter Kit*, namely the assembler and debugger, the C31 architecture, peripherals and instruction set. The second part is targeted to the realization of algorithms for FIR, IIR, and FIR-adaptive filtering, multirate processing using polyphase decomposition, FFT and spectral analysis, and single side band modulation using a Hilbert transformer.

An additional aspect that is believed to contribute to a more regular commitment of the students is that at the end of each laboratory class, each group of two students must hand over a short report to the teacher including the significant results and main conclusions of each laboratory exercise. All the questions to be answered in each report are known trough the Web site (http://www.fe.up.pt/∼ajf/pdstr) of the course one week before the laboratory class takes place. Each report is corrected, graded, and returned back to the students one week later. This way, students have an early feed-back of their performance which is very helpful in subsequent laboratory work. In fact, a good performance encourages students to excel in subsequent classes, and a poor performance encourages students to strive for better grades in subsequent classes since they feel each week counts to the final grade. Both of these positive reactions are naturally highly dependent on a vigilant and diligent attitude of the teacher.

3. DSP LABORATORY EXPERIMENTS

Most of the theoretical background for all laboratory experiments has already been presented in the pre-requisite course (EEC4162), as it is the case of structures for the realization of discrete systems. However, new topics are also presented in theory oriented classes of EEC5274, at least one week before a given topic could possibly be related to a laboratory work, as it is the case of polyphase decomposition in multirate filtering.

The purpose of those new topics, that may receive a different emphasis on different editions of EEC5274, is to give students not only a broader perspective of DSP solutions and applications, but also to create room for innovation and initiative in the context of the laboratory experiments. Examples of topics that currently fall into this category are the following ones:

- filter banks, uniform filter banks and their connection to the DFT,
- half-band filters, M-band filters, power complementary filters and their design,
- the QMF filter bank, design and implementation issues, multiresolution analysis using QMFs,
- efficient realization of interpolation or decimation filters using polyphase decomposition,
- cepstral analysis, concept and applications.

The first five laboratory experiments are designed to get the student familiar with the development environment of the *C31 Starter Kit*, with the writing of C31 assembly code, and with examples of basic realization structures. The realization objectives and challenges associated to each one of this initial set of laboratory experiments are detailed next.

1. **Verification of the Aliasing in Sampling** The A/D and D/A converters of the *C31 Starter Kit* are configured to operate without the anti-aliasing filter but with anti-imaging filter (that can not be removed anyway). Students are asked to estimate the sampling frequency of the converters just by listening to the audio output of the DSK, and by varying the known frequency of a pure sinusoid that is injected to the audio input of the DSK. To most students, the subjective effect of the aliasing phenomenon represents a mixture of surprise and fascination. Students are also asked to analyze the assembly code in order to understand the initialization of a timer, the serial port and the Analog Interface Circuit (AIC) containing both A/D and D/A converters (namely the configuration of the sampling frequency and the activation and setting of the cut-off frequency of switched-capacitor analog filters), as well as the interrupt mechanism handling arriving and departing audio samples. Students are finally asked to activate the anti-aliasing filter and to report the differences.

2. **Waveform Generation and Converter Testing** Taking as a reference a simple example of waveform generation using a look-up table, students are asked to program other waveforms and to verify the resulting

analog waveform. As the AIC of the DSK can be configured in a loop-back mode (*i.e.*, the output of the D/A converter is internally connected to the input of the A/D converter) students are asked to synthesize an appropriate waveform in order to evaluate the accumulated quality of D/A and A/D conversion, namely noise floor, offset, delay and differential nonlinearity.

3. **Fixed-point processing versus floating-point processing** Students are first asked to interpret the C31 internal format of floating-point representation (that differs from the standard IEEE754). Afterwards, students are invited to analyze the assembly code that generates a sine and a cosine function, in a recursive way, using fixed-point representation, and to appreciate the result of multiplying both functions and the result of point-to-point sum of the squares of both functions. Students are then asked to modify scale factors and to conclude on the improvement or degradation of results. Finally, students are asked to modify the assembly code to floating-point processing (and to eliminate scale-factor control) and to draw conclusions on the difference to the previous results.

4. **FIR Filtering** In the first part of this laboratory work students have to analyze the assembly code that implements the discrete filter $H(z) = 1 - z^{-15}$ and are encouraged to derive the theoretical frequency response of this filter. Then, using the audio input and output of the DSK, students are invited to obtain experimentally the frequency response of this filter after setting the sampling frequency of the converters to $20kHz$. To a significant number of students, this is an opportunity to clarify (for good) the relation between normalized frequency and analog frequency. In the second part of this laboratory work, students are asked to implement and test an FIR equiripple filter of length 32, whose coefficients are obtained by means of a Matlab command file that is provided. Finally, students are asked to modulate the discrete filter to $\omega = \pi/2$ and to $\omega = \pi$, and to determine experimentally the frequency response of the resulting filter.

5. **IIR Filtering** An assembly code is provided that implements a 6th order band-pass filter suggested in [3], using a type 2 direct realization structure. This same filter is also illustrated in a Matlab command file that is also provided. Students are asked to clarify the relation between the coefficients of the transfer function of the filter, and those of the realization structure. To many students, this is an opportunity to recall the minus sign affecting all the coefficients involved in the realization of the poles. Students are then asked to draw the frequency response they expect to derive experimentally, and then to compare that to the practical results. As a result of this laboratory work (and of the previous one as well), students gain a practical understanding on the correct use of pointers, the indirect and circular addressing modes, and parallel instructions of the C31 DSP.

After this initial set of laboratory experiments, students feel already reasonably comfortable with C31 assembly programming, debugging and testing. The next set of five laboratory experiments targets an increased autonomy of each group (of two) students in creating an efficient DSP solution to some specified design problem. Therefore, only in two (out of five) of the following laboratory classes, an example of assembly code is provided. The objective is that students learn to follow the design chain:

- characterization of the problem and identification of a suitable DSP realization approach,
- validation of the conceptual approach in a simulation environment such as Matlab, and evaluation of an efficient implementation solution and associated performance,
- writing of the assembly code, debugging and assessment of the robustness and quality of the implementation.

The remaining laboratory experiments are as follows.

6. **Five Vowel Synthesizer** A Matlab command file is provided to the students that illustrates the generation of a single (synthetic) voiced sound by exciting a 6th order all-pole IIR filter with a $200Hz$ train of impulses. This IIR filter allows the definition of three formants that are enough to distinguish between the most common vowels found in the Portuguese language: /à/, /é/, /i/, /ó/ and /u/. Students are first asked to extend the Matlab simulation so as to synthesize the five vowels. A didactic application conceived by the first author, running on a PC and allowing the identification, in real-time, of the pitch, the first three formants of a voiced sound, and recognizing vowels, is also made available (through the Web site www.inescporto.pt/cienciaviva) to the students. This application encourages students to give a personal flavor to each implementation by using the parameters derived from their own voices. Experience has revealed that this 'personalization' has a high impact on the motivation to the laboratory work. Some student present very realistic and convincing vowel synthesizers by modulating the rate of the impulse train like in a normal voice. Following this simulation stage, students are then asked to adapt the assembly code tested in the previous laboratory class

(and that implements a 6th order IIR filter) in order to synthesize, in loop, the above vowels, with a similar loudness. Experience has shown that this is one of the most successful experiments since students engage in a very committed attitude, while in a relaxed laboratory atmosphere.

7. **Interpolation Using Polyphase Filters** The illustration of the polyphase decomposition of a 64-tap FIR interpolating filter is provided to the students in a Matlab command file. As the interpolation factor is four, students are first asked to demonstrate the analytical expression denoting the ideal phase response of each one of the four polyphase components. Besides verifying that the magnitude response of each polyphase component approaches very closely the all-pass response, they are also invited to draw conclusions on the relation between ideal and actual phase response of each component (which match very well). As the AIC of the DSK allows asynchronous conversion rate between A/D and D/A, students are asked to set the former to $5kHz$ and the latter to $20kHz$, using an assembly code that eases the task of programming the four polyphase components. They are finally asked to implement 4-fold interpolation in an efficient way, and to assess by practical experimentation the difference of the $\sin(x)/x$ effect for frequencies between $200Hz$ and $2kHz$, when 4-times interpolation is used, relative to the case when it is not.

8. **Adaptive Filtering** The implementation of a 32-tap FIR adaptive is illustrated by means of a Matlab file and C31 assembly code file that are provided to the students. The adaptation criterion is the standard LMS and the adaptive filter is configured to operate as system identification or as interference cancellation. Students are first invited to evaluate on the Matlab environment the influence of the μ factor (step-size adaptation parameter), and to confirm afterwards the same findings using the DSK and analog signals. Most students reveal to become quite impressed when observing the system to "learn" by it self "on-the-fly" as a consequence of the fact that the filter coefficients of the FIR adaptive filter converge to some expected or unexpected shape. Students are finally asked to set the step-size parameter so that the adaptive filter implemented in the DSK effectively cancels, in real-time, the sinusoidal interference arising from a microphone that feeds back the sound from an active loudspeaker (*i.e.*, a sort of 'feed-back killer').

   Given that the assembly code structure associated to this realization is very simple, students are invited to profile the code by hand and to estimate the maximum filter length that allows real-time operation of the adaptive filter.

9. **FFT Implementation** Two versions of an FFT implementation on the Matlab environment are provided to the students. The first version is meant to be readable and easily reminiscent of the 'Radix-2' structure of an FFT. The second version is an adaptation by the author from the previous version but targeting an efficient C-like or assembly-like implementation. Students are asked to answer a number of questions testing their understanding of the FFT in relation to the Matlab code, and are then asked to implement a 64-point FFT for the C31 taking as a reference the 'optimized' code provided. This code is important to facilitate subsequent evaluation by the teacher of the solution presented by each group of students. Students are finally asked to draw conclusions on the real-time operation of the FFT implementation as a spectrum analyzer (namely frequency resolution and selectivity), using two different windows: rectangular and Hanning. This is the laboratory work that is more time consuming to students and that better reveals their accumulated knowledge and programming skills. For this reason, they are allowed to have a little more than two weeks to complete this work.

10. **Single Side Band Modulation** This laboratory experiment is a little bit less complex than the (previously described) FFT experiment but has the advantage to motivate students to the complete design-simulation-realization chain of a DSP project. For this reason, students take only one of these two laboratory experiments (FFT or SSB) as their final laboratory assignment.

    The SSB laboratory experiment requires that students implement a band-pass filter followed by a single side-band modulator based on a Hilbert transformer. The whole system must operate in real-time and perform a spectral transformation on a speech signal. Only a few mandatory specifications are given:

    - the sampling frequency should be $20kHz$,
    - the band-pass filter must reject frequencies outside the range $200Hz$ - $4.8kHz$,
    - the system must be able to perform the modulation of a single side-band of the speech spectrum to a carrier frequency that is a multiple integer of $200Hz$.

    Each group of (two) students must present a realization of the SSB system using the C31 DSK and making evidence at least of the following spectral transformations on a speech signal:

    - spectral up-shift by $200Hz$,

- spectral down-shift by $200Hz$,
- spectral inversion.

Despite the fact that students are given a Matlab command file and an audio file illustrating the effect of a "pitch shifter" (*i.e.*, up-shift of the upper side-band of the speech signal), they are expected to design, simulate and implement all system blocks (namely the Hilbert transformer and the sin and cos modulators) so that real-time operation is achieved.

Experience has shown that although some students do not succeed in completing their realizations, it is generally true that students curiosity (and even anxiety) to know how a specific spectral transformation sounds like using their own voices (or a music signal), acts as a strong motivation factor.

4. EEC5274 RECEPTIVITY

At the end of the semester students are asked to express their opinions in an anonymous way, by commenting on the following topics:

- the aspects they have appreciated the most during the course,
- the aspects they have appreciated the least during the course,
- their personal recommendations to improve the course in future editions,
- their opinion on an additional course following the same format of EEC5274 but in a more ambitious and professional perspective combining both C and assembly programming,
- any free comment.

Concerning the first question, most students emphasize the fact that the course establishes an intelligible and even intuitive linkage between theory and practice, and that the methodology of the course helps to establish a discipline that elicits insight and application. Regarding the negative aspects of the discipline, many complaints arise that more time should be allowed to complete and explore further ('play' as they frequently say) each laboratory work, and that it takes too long to learn the C31 assembly language and code development. Regarding recommendations, students generally point out that the objectives for each laboratory work should be known earlier than just one week before, that laboratory classes should be longer than 2h:30m, and that all laboratory experiments should be open to student creativity and personalization. Concerning the pertinence of another course on applied DSP combining C and assembly programming, more than 50% of students express that they would possibly take that course, as long as it unveils a larger diversity of application scenarios, possibly by combining video and audio. The last question normally leads students to repeat their perspectives already expressed in previous answers, but sometimes unexpected thoughs (or feelings) are found. For example, one student stated that he already felt like missing the university.

5. CONCLUSIONS

The structure and methodology of a DSP laboratory course focusing on hands-on experience has been described. All the course pedagogical material including Power Point slides, exams, laboratory work description and accompanying Matlab code and/or assembly code is in Portuguese and is available on the course Web site (http://www.fe.up.pt/∼ajf/pdstr). This course in integrated in the ECE curriculum at the Faculdade de Engenharia da Universidade do Porto, in Portugal. Students feed-back indicates that their understanding of DSP concepts and techniques is in fact consolidated and stimulated by hands-on laboratory experience. This perspective is in line with the rationale and objectives of the course: by facing and overcoming practical DSP laboratory challenges, students acquire a higher retention of knowledge, develop a rewarding sense of achievement, and are more motivated to be involved in final-year projects using DSP technology.

Future plans include the extension of the DSP hands-on laboratory concept to the graduate level where a more professional development environment will be used. This environment will probably be based on the Texas Instruments C6711 floating-point processor and the Code Composer Studio development/debugging environment. The objective here is to focus on complex algorithm implementation combining high-level languages such C or C++ and assembly code development.

6. REFERENCES

[1] Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck, *Discrete-time Signal Processing*, Prentice-Hall Inc., 1998, 2nd ed.

[2] T.I. Instructer's Guide, *TMS320C3x Digital Signal Processing Teaching Kit*, Texas Instruments, 1998.

[3] Rulph Chassaing, *Digital Signal Processing - laboratory experiments using C and the TMS320C31 DSK*, John Wiley & Sons Inc., 1999.

[4] Henrik V. Sorensen and Jianping Chen, *A Digital Signal Processing Laboratory using the TMS320C30*, Prentice-Hall, Inc., 1997.