INTERFACING JAVA-DSP WITH A TI DSK FOR USE IN A SIGNAL PROCESSING CLASS

Andreas Spanias, Chih-Wei Huang, Ashwin Natarajan, Rony Ferzli, Homin Kwon, Venkataraman Atti, Visar Berisha, Leonidas Iasemidis, Harish Krishnamoorthi, Photini Spanias, Shibani Misra, Mahesh Banavar, Kostas Tsakalis, Susan Haag

Arizona State University
Tempe, AZ 85287-5706

Abstract

In this paper, we describe the development of a Java-DSP (J-DSP) interface with DSP hardware for use in undergraduate signals and systems and DSP classes. The interface enables undergraduate students to design and implement algorithms real time on DSP hardware using the user-friendly graphical interface of J-DSP. Simulations involving digital filters and FFTs are first established in the object oriented J-DSP environment. Through the use of a clever software interface, real-time implementation of select algorithms become possible on the TI DSP Starter Kit C6713. These real-time implementations enable students to examine the properties of various signal processing algorithms using real-life signals. A simple audio compression scheme that uses the Fast Fourier Transform (FFT) is described in detail. The algorithm exposes students to the application of the FFT in a simplified MPEG-like audio compression scheme. The hardware–software interaction of J-DSP with the TI DSK is also covered in the class; an introduction to the architecture and its peripherals is also part of the learning experience. Pre- and Post-assessment instruments have been developed and administered.

Introduction

An effective course in Digital Signal Processing (DSP) must convey theoretical and practical knowledge of key concepts associated with the subject. While simulation tools such as MATLAB, Simulink, and J-DSP are valuable, running DSP algorithms on real-time hardware can further enhance the understanding of these concepts. With real-time DSP labs, students learn about important implementation issues associated with signal processing algorithms and DSP chips [1-6]. They also gain an appreciation for several compelling applications that use DSP chips, such as digital cellular phones and MP3 players. These days, several segments of the industry require students that have been exposed to implementation issues and DSP hardware. In this paper, we choose the Texas Instruments (TI) TMS320C6713™ DSP Starter Kit (DSK), which is based on the C6713™ floating-point processor, as a platform for real-time DSP experiments. This choice was motivated by the availability of user-friendly development tools and the popularity of the TI DSK in industry and academic training circles. Although fixed-point processors are less expensive and more power efficient relative to floating-point processors, they are often more difficult to use in an undergraduate learning environment and pose certain limitations in terms of handling natural data computations. Even with specialized training, programming fixed-point processors is difficult. Floating-point processors are easier to program than their fixed-point counterparts, but developing code for them is still challenging. The learning curve is steep for undergraduates, especially if the DSP platform has to be programmed in assembly language. TI has developed the Code Composer Studio™ (CCS™) that provides an integrated development environment (IDE) for users. CCS is a powerful tool for writing and debugging code and has an inbuilt compiler that generates the assembly level code required to run the DSP. However, even CCS can be

overwhelming for undergraduates because it requires knowledge of tedious programming structures. All these issues highlight the need for a front end that allows students to access and exploit the capabilities of real-time DSP hardware with a user-friendly GUI-based software.

In this paper, we describe a GUI-based approach to teach undergraduate students the concepts of real-time signal processing using J-DSP [7]. Combining the ease of use of J-DSP and the powerful capabilities of the TI DSK allows students to explore the concepts of real-time signal processing in a friendly environment. Before using J-DSP to run real-time DSP algorithms on the DSK, students are given a brief overview of the interface between J-DSP and the TI DSK. The process of connecting J-DSP to the hardware via the RS-232 and the USB ports is explained. The role of CCS in this process is also described. Students are then asked to select various DSP functions to examine the differences between real-time and offline signal processing. Hands-on exercises that use this J-DSP interface to the DSK have been developed and disseminated to undergraduate students in the ASU DSP class. A laboratory session was organized where students programmed select real-time DSP tasks using J-DSP. Pre- and post-lab quizzes were given to assess their understanding of real time DSP.

The rest of the paper is organized as follows. The next section discusses the hardware aspect of this educational DSP DSK setup. We then explain the software and interfacing aspects of this project. The next section outlines some of the functions available to students, while the final section provides concluding remarks.

Hardware Overview

The TI DSKC6713 Board

The TI DSKC6713 board [8-12] is based on the TMS3206713 processor, which is a floating-point DSP chip operating at 225 MHz. The board also includes the 32-bit stereo codec TLV320AIC23 (AIC23) to access and produce analog input and output signals. Sampling rates are programmable and can be varied between 8 kHz and 96 kHz. The board has 16MB of SDRAM and 256 kB of flash memory. Input and output functions are provided by several jacks (MIC IN, LINE IN, etc). The MIC IN and HEADPHONE ports constitute the McASP (Multichannel Audio Serial Ports) and the LINE IN and LINE OUT ports comprise the McBSP (Multichannel Buffered Serial Ports.) Also available on the board are the EMIP (External memory interface), two inter-Integrated Circuit buses, two timers, one GPIO (General Purpose Input and Output), one HPI (Host-Port Interface), and one EDMA (Enhanced Direct Memory Access) with 16 independent channels.

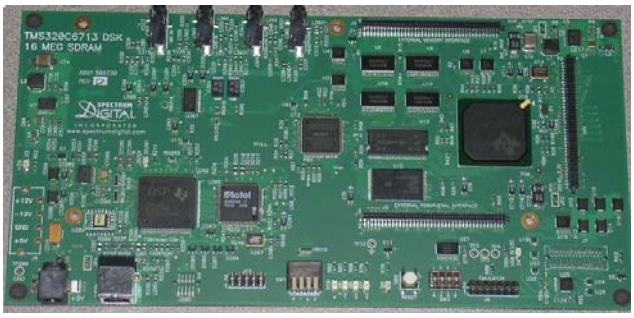

Figure 1. The TMS320C6713 DSK board.

The TMS320C6713 DSP

The TMS C6713 DSP chip can fetch 8 instructions per cycle and its performance is rated at 1800 millions of instructions per second

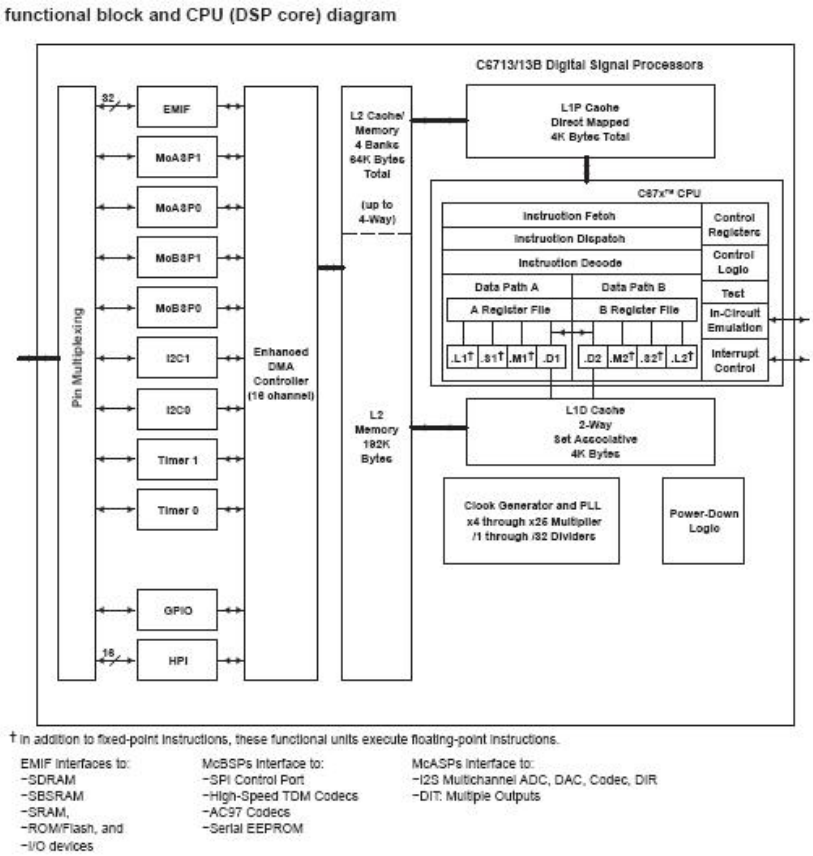

Figure 2. The functional block diagram of TMS320C6713 (Courtesy of Texas Instruments).

(MIPS). The C6713 can perform 2 multiply and accumulate (MAC) instructions per cycle which are very important in digital filtering implementations. The processor can access 264 kB of internal memory including 8 kB allocated to program and data cache and the remaining 256 kB shared by program and data space.

The Importance of the McBSP Ports

Since the McBSP ports are important in this J-DSP interface project, we discuss them in some detail. The McBSP ports provide the interface between the input and output facilities of the TI chip family. There are two McBSP ports; McBSP0 is used for control and McBSP1 is used to send and receive data. Both ports support full duplex communication. Double buffered data registers are available for a continuous signal stream. Multichannel transmit and receive is also available for up to 128 channels with choice of data size (8, 12, 16, 20, 24 and 32 bits). Most importantly, the McBSP provides conversion functions (ADC and DAC).

The DR (Data Receive pin) receives data from the external input (such as a microphone). The received data is first stored in the RSR (Receive Shift Register). Once the RSR receives the complete data stream, it is moved into the RBR (Receive Buffer Register). While the DRR (Data Receive Register) is not ready to be read by the CPU or the DMA controller, the RBR copies the data into the DRR. Data transmission is similar to data reception except the DX (Data Transmit pin), DXR (Data Transmit Register) and XSR (Transmit Shift Register) are employed.

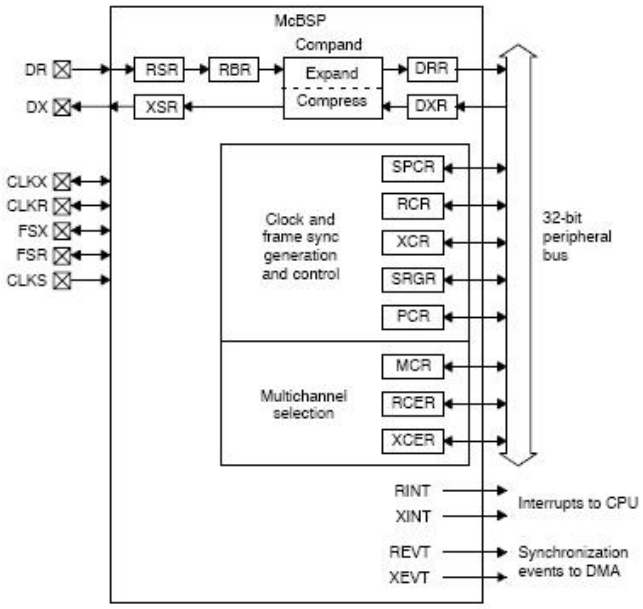

Figure 3. The block diagram of McBSP (Courtesy of Texas Instruments).

Software and Interface overview

The Code Composer Studio

The code composer studio (CCS) [10] includes a C/C++ editor, compiler, assembler, linker, and debugger. The code for the DSK is programmed in CCS and then loaded to the board. Support for DSP/BIOS is included using a GUI that allows users to configure interrupt handlers, multithreading, etc. Additionally, CCS optimizes the instructions and memory for efficiency.

Interface with J-DSP

Although CCS is a powerful tool, it can be difficult for first time users. We have developed an interface to enable students to load and run DSP functions on the DSK through the object oriented J-DSP software[15-16]. We have also developed a simple custom GUI in J-DSP to facilitate working with the real-time DSK hardware. This is shown in Figure 4. Basically, J-DSP acts as an extra layer between the end-user and the DSK.

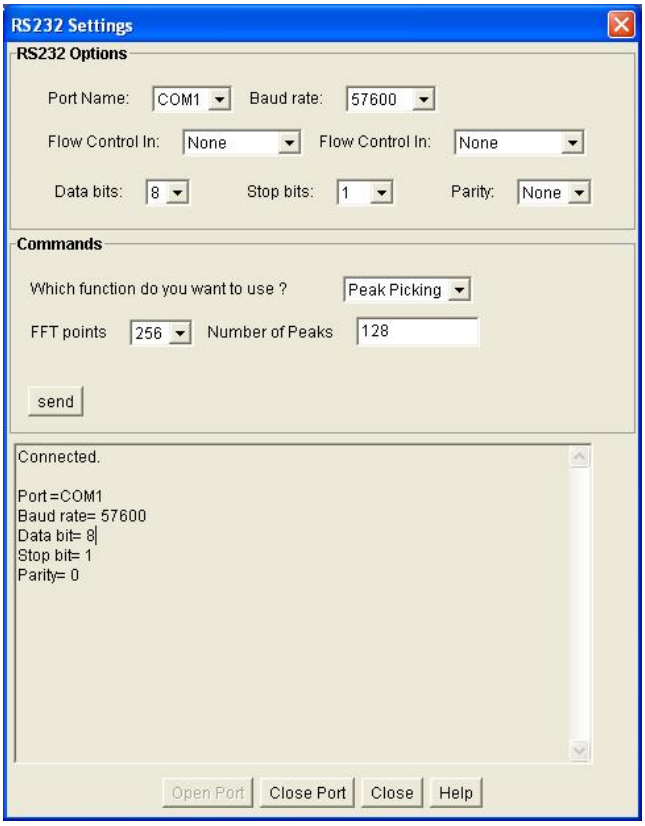

Figure 4: The real-time DSP block in J-DSP.

For communication via the RS-232 port, Java requires a *signed applet* to permit resource access[13]. Java also needs the Communication API to access the RS-232 port. The end user must copy the win32com.dll, comm.jar, and javax.comm.properties to the specific directories detailed in the Communication API manual.

Modifying the DSK to Resolve Hardware Conflicts

Successful interfacing requires both CCS and J-DSP to communicate with the DSK as shown in Figure 5.

The CCS communicates with the DSK via the USB port, while J-DSP employs the RS-232 port available on the daughter card used with the DSK. However, a hardware conflict arises because the daughter card and the AIC23 codec both employ the McBSP1 port. The solution was to rewire some of the hardware to enable the use of the McBSP0 port for communication

via the RS-232 port, and allow McBSP1 to be used by the codec, as shown in Figure 6.

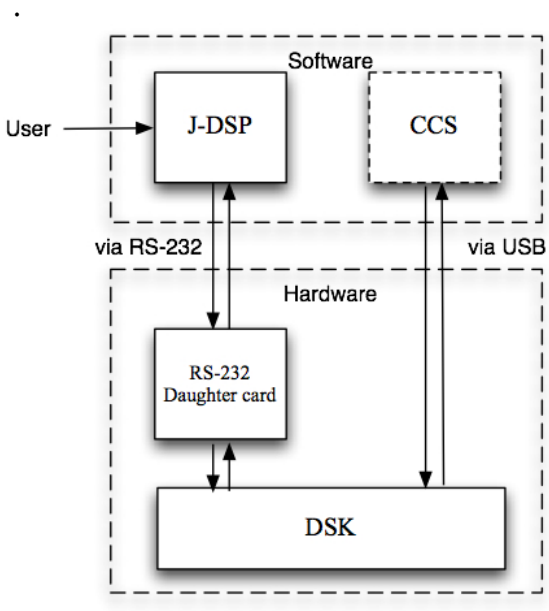

Figure 5: Layers involved in interfacing J-DSP with the DSK.

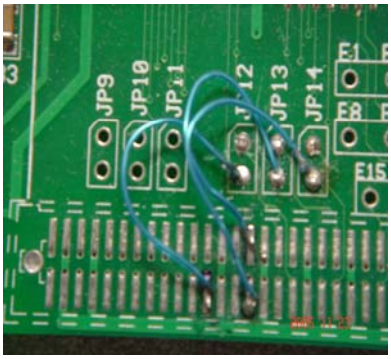

Figure 6: Re-wiring the daughter card.

The user has to connect the J-DSP block shown in Figure 4 to the DSK via the RS-232 port by clicking the [Open Port] button. USB connectivity is also possible through adaptors. The default settings are: baud rate 57600bps, 8 data bits with 1 stop bit and no parity. The status of the connection is updated in the textbox at the bottom of the dialog box. Once the connection is made, various DSP functions, shown in Figure 7, can be selected from the drop down box.

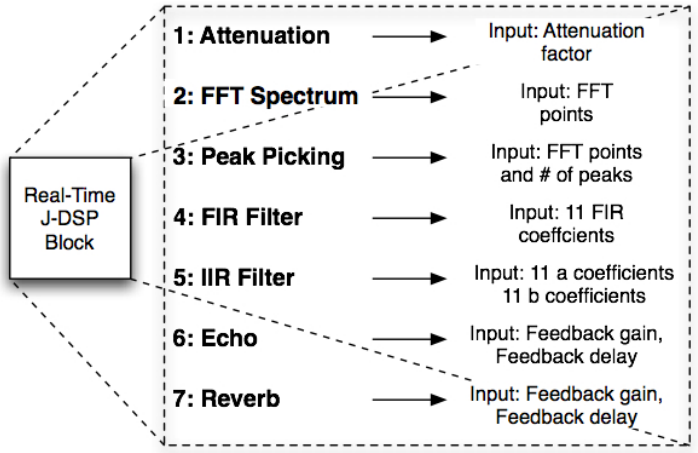

Figure 7: DSP functions available to the user.

Demo/Lab Exercise

Among the various applications implemented in real-time via J-DSP, the peak-picking algorithm is of particular interest. A music file was played on a continuous loop and the peak-picking algorithm is initiated. Students can change various parameters and observe their effects on the reconstructed signal.

FFT Peak-Picking and Parseval's Theorem

The peak-picking algorithm is used to implement a lossy audio compression scheme. Its educational value comes from its association with Parseval's theorem and the fact that transform domain analysis-synthesis [16] is used in JPEG, MPEG, and MP3 [17-22] algorithms. The signal is compressed by using a reduced set of the chosen transform parameters. An assumption that a signal follows certain mathematical and statistical properties is vital in choosing signal parameters. After redundancies are removed in the compression step, the synthesis step involves reconstructing the signal with the reduced parameter set. The peak-picking compression scheme is a simple FFT-based algorithm depicted in Figure 8.

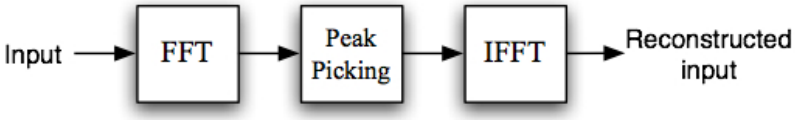

Figure 8: The peak-picking block algorithm.

The input signal vector is transformed using the FFT. The next step involves redundancy removal or compression where only a specific number of peaks of the FFT magnitude spectrum are selected (peak-picking). The peak picking implies that the maximum power is maintained in the compressed signal as implied by the Parseval's theorem. These select frequency components are then transformed back to the time domain using the IFFT. For proper reconstruction of the original signal, frequency symmetries have to be preserved. Since this compression scheme is lossy, there is a certain error associated with the reconstruction. The key is to choose the components such that the error is not perceptible.

Deciding the number of peaks to be selected is crucial to the quality of the reconstructed signal. Selecting more peaks ensures high quality reconstruction at the expense of a higher rate.

Students from the undergraduate DSP class first played the original music file. They were then asked to process this audio signal using the peak-picking algorithm that was implemented real time on the DSK board. The number of FFT components selected was varied. Two methods for selection were implemented, namely peak-picking and initial low-frequency component selection. The students were asked to subjectively rate the quality of the reconstructed signal in each case. They were also asked to assess the differences between real time and offline implementations.

An assessment quiz was administered before (pre-lab) and after (post-lab) the hands-on laboratory exercise. The questions posed are itemized below:

1. Peak-picking of the DFT is typically used for:
   a. Filter design
   b. Speech/Audio compression
   c. JPEG compression
   d. None of the above
2. Peak-picking is equivalent to downsampling. (T/F)
3. Picking the first components bears similarities to lowpass filtering. (T/F)
4. In the peak-picking algorithm all the phase components are set to zero. (T/F)
5. Running DSP algorithms on a generic processor is faster than running the same algorithm on real time DSP hardware. (T/F)
6. The SNRs obtained with peak-picking are better / worse (circle one) than the SNRs obtained by choosing the same number of the initial low frequency DFT components.
7. Arrange in order the following functions that are involved in the A/D conversion of the input signal at the codec embedded in the TI-DSK board:
   a. Sampling
   b. Pre-filtering
   c. Quantization
8. The real time implementation of the peak picking algorithm implies that the output is delivered
   a. with precisely 0 delay.
   b. with delay approximately equal to the frame size.
   c. with delay equal to the number of spectral components selected times the sampling period.

9. Choose those that are true. DSP chips are
   a. embedded in PCs to assist the main processor to run software.
   b. used in cell phones.
   c. used in Hi-def TV.
   d. used in typical digital wrist watches.
10. DSP chips are optimized
   a. for FFTs.
   b. for high order digital filters.
   c. to manage peripheral devices on the PC such as the mouse and keyboard.
11. Circle the statements that are correct.
   a. A DSP chip does a Mulitply-Accumulate in one cycle.
   b. A Pentium III chip does a Multiply-Accumulate in one cycle.
12. Circle the correct statement
   a. Fixed-point processors consume less power than floating-point processors.
   b. Floating-point processors are easier to program than fixed-point processors.
   c. Floating-point processors are more expensive than fixed-point processors.

In summary, students showed an improvement in terms of knowledge of general topics in real-time processing. They also became familiar with real-time compression techniques that utilize the FFT.

Conclusion

This paper described the basic hardware architecture of the TMS320C6713 DSK board along with some of its functions. Interfacing the real-time DSK hardware with the software such as J-DSP and the CCS was explained. Pre- and post assessment quizzes were administered and improvements were demonstrated. This new real-time capability of J-DSP enabled instructors to provide a valuable introduction to real-time DSP without having to cover low level assembly programming. Through this laboratory experience, students gained knowledge on the following topics:

- The association of Parseval's theorem with real-time transform-domain compression schemes.
- Differences between offline and real-time signal processing in terms of execution time and software complexity.
- Capabilities of DSP chips in terms of real time processing of signals.
- Association of FFT-based compression schemes with compelling JPEG and MP3 applications.
- Exposure to DSP hardware issues.

Acknowledgment

This work has been sponsored in part by the NSF CRCD EI Project (award 0417604), the NSF award 0443137, and the ASU FSE SenSIP Center.

References

1. T. B. Welch, C. H. G. Wright, and M. G. Morrow, "Experiences in Offering A DSP-based Communication Laboratory," Digital Signal Proc. Workshop, 2004 and the 3rd IEEE Sig. Proc. Education Workshop, pp. 68-72, Aug 2004.

2. W.-S. Gan, "Teaching and Learning the Hows and Whys of Real-Time Digital Signal Processing," IEEE Trans. on Educ., vol. 45, no. 4, pp. 336-343, Nov. 2002.

3. M. D. Galanis, A. Papazacharias, and E. Zigouris, "A DSP Course for Real-Time Systems Design and Implementation Based on the TMS320C6211 DSK," 14th International Conf. on Dig. Sig. Proc., vol. 2, pp. 853-856, July 2002.

4. S. L. Wood, G. C. Orsak, J. R. Treichler, D. C. Munson, S. C. Douglas, R. Athale, and M. A. Yoder, "DSP Concepts and Experiments in a High School Curriculum," 37th Conf. on Sig., Systems, and Computers, vol. 2, pp. 1365-1369, Nov. 2003.

5. C. H. G. Wright, T. B. Welch, and W. J. Gomes III, "Teaching DSP Concepts using MATLAB and the TMS320C31 DSK," Proc. of International Conf. on Acous., Speech, Sig. Proc., vol. 6, pp. 3573-3576, March 1999.

6. M. G. Morrow, T. B. Welch, and C. H. G. Wright, "A Tool for Real-Time DSP Demonstration and Experimentation," Proc. of 10th IEEE Digital Signal Proc. Workshop, pp. 162-167, Oct. 2002.

7. JDSP http://jdsp.asu.edu

8. Texas Instruments, Inc., http://www.ti.com

9. Texas Instruments, Inc., "TMS320C6713 DSP Starter Kit", http://focus.ti.com/docs/toolsw/folders/print/tmdsdsk6713.html

10. Texas Instruments, Inc., "Code Composer Studio Features and Demos", http://focus.ti.com/dsp/docs/dspsupporto.tsp?sectionId=3&tabId=432.

11. R. Chassaing, Digital Signal Processing and Applications with the C6713 and C6416 DSK, John Wiley & Sons, Inc., Hoboken, New Jersey, 2005.

12. DSPGLOBAL, "DSPG-IC1-232 Rs232 daughtercard", http://www.dspglobal.com/Int%20Card%20SERIAL.htm.

13. Java Technology, http://java.sun.com.

14. A. Spanias and V. Atti, "Interactive On-line Undergraduate Laboratories Using J-DSP," IEEE Trans. on Education Special Issue on Web-based Instruction, vol. 48, no. 4, pp. 735-749, Nov. 2005.

15. A. Spanias, A. Venkataraman, K. Ahmed, A. Papandreou-Suppappola, M. Zaman, and T. Thrassyvoulou "On-line signal processing using J-DSP," IEEE Signal Processing Letters, Volume: 11 , Issue: 10 , pp. 821 – 825, Sept. 2004.

16. S. Ahmadi and A. Spanias, "Algorithms for Low-bit rate sinusoidal coding," Speech Communications, 34(2001), pp. 369-390, June 2001.

17. T. Painter, A. Spanias, "Perceptual segmentation and component selection for sinusoidal representations of audio," IEEE Transactions on Speech and Audio Processing," Volume 13, Issue 2, pp. 149-162, March 2005.

18. Y. Song, A. Spanias, V. Atti, and V. Berisha, "Interactive Java Modules for the MPEG-1 Psychoacoustic Model," Proc. ICASSP '05, Vol. 5, pp.581-584, Philadelphia, March 2005.

19. R. Ramapriya and A. Spanias, "A Simulation Tool for introducing MPEG - Audio (MP3) concepts in a DSP course, Proc. of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-2002), Orlando, May 2002.

20. T. Painter and A. S. Spanias, "Perceptual Coding of Digital Audio," Proceedings of the IEEE, pp. 451-513, Vol. 88, No.4, April 2000 (winner of field series 2002 IEEE Donald G. Fink Prize Paper Award).

21. A. Spanias, T. Painter, V. Atti, Audio Signal Processing and Coding, 464 pages, ISBN: 0-471-79147-4, Wiley, New Jersey, February 2007.

22. A. Spanias, Digital Signal Processing; An Interactive Approach, 388 pages, Textbook with theory, problems, and JAVA-DSP computer exercises, ISBN 978-1-4243-2524-5, Tempe, January 2007.

Biographical Information

Andreas Spanias is a Professor in Electrical Engineering and Director of the SenSIP Consortium at Arizona State University (ASU). He is the author of two text books. He and his student team developed the computer simulation software Java-DSP (J-DSP-ISBN 0-9724984-0-0) which is being used in the ASU DSP courses. He is co-recipient of the 2002 IEEE Donald G. Fink paper prize award and he is a Fellow of the IEEE. He served as Distinguished Lecturer of the IEEE SPS in 2004.

Chih-Wei Huang is a Masters student in Electrical Engineering at ASU. He worked on DSP and Java implementations of the MP3 decoder.

Ashwin Natarajan is a Doctoral student in Electrical Engineering at ASU. He worked on adaptive filter simulations in J-DSP.

Rony Ferzli is a Doctoral student in Electrical Engineering at ASU. He worked on DSP boards and his research is in image processing.

Homin Kwon is a Doctoral student in Electrical Engineering at ASU. His research interests are in sensor networks.

Venkataraman Atti received his Doctoral Degree at ASU in 2006. He is currently with Acoustic Technologies working on echo cancellers.

Visar Berisha is a Doctoral student in Electrical Engineering at ASU. His research interests are in bandwidth extension of speech. He has done work in DSP, sensor networks, radar, and biomedical signal processing. He held internships at General Dynamics, Raytheon, MIT Lincoln Labs and Medronic.

Leonidas Iasemidis is Associate Professor in Biomedical Engineering at ASU. His research interests are in biomedical signal processing.

Photini Spanias has a Doctoral degree from the College of Education at ASU. She is currently a lecturer at ASU teaching math classes. She participated in the assessment of J-DSP software.

Shibani Misra is a Masters student in Electrical Engineering at ASU. She worked on J-DSP implementations of genomic signal processing algorithms.

Mahesh Banavar is a Doctoral student in Electrical Engineering at ASU. He works on sensor networks and multi-carrier systems.

Kostas Tsakalis is a Professor in Electrical Engineering at ASU. He works on adaptive control systems. He co-developed J-DSPs controls simulator. He published in automatic control and co-authored a monograph in adaptive control.

Harish Krishnamoorthi is a Doctoral student in Electrical Engineering at ASU. He is the Teaching Assistant of the DSP class at ASU and works on audio processing systems.

Susan Haag is an Associate Research Professor with the Ira A. Fulton School of Engineering. She specializes in assessment and evaluation.