AC 2007-2290: INCORPORATING SYSTEM-LEVEL DESIGN TOOLS INTO UPPER-LEVEL DIGITAL DESIGN AND CAPSTONE COURSES

Wagdy Mahmoud, University of the District of Columbia
IEEE Senior Member

Incorporating System-Level Design Tools into upper-Level Digital Design and Capstone Courses

Abstract

This paper describes the efforts to incorporate system-level digital design tools and state-of-the FPGA boards in the capstone design course sequence. This paper provides the details of two capstone projects in the areas of digital communications and image processing. This paper also details the challenges involved in using continually-evolving system-level design tools and the efforts made to reduce their learning times.

Introduction

ABET 2000 requires providing students with a significant hands-on design experience. Graduating electrical engineering students should have the ability to develop system-level designs for a variety of applications, implement these designs in functional hardware, and test the hardware in real-life operating conditions. To achieve such professional competence, students should be required to participate in a sequence of hardware design experiments and projects. These laboratory exercises aim at: a) sharpening students' abilities to design complex digital circuits and systems, and to interface these designs to peripheral devices, b) exposes students to contemporary digital design tools, and c) provide students with lessons that can help them become long-life learners and successful professionals.

This paper describes the efforts to incorporate system-level digital design tools and state-of-the FPGA boards in the capstone design course sequence. This is part of the Electrical Engineering Department efforts to enhance the quality of upper division design courses through the introduction of computer-aided design and system-level design tools. These tools are vertically integrated with many of junior, upper-level, and capstone courses. The capstone design is a two-semester sequence courses. The cross-disciplinary capstone design projects emphasize the hardware/software implementation of algorithms, the design and use of intellectual property (IP) cores, and system-on-a-chip (SoC) concept.

The system-level software packages used include Matlab, Simulink, assortment of Matlab toolboxes and blocksets, Xilinx System Generator (SysGen), Xilinx embedded design kit (EDK), and Extreme DSP. Matlab blocksets provide commonly used design components needed to model the majority of digital control, DSP and digital communication systems. The Xilinx SysGen allows performing FPGA in the loop simulation with Simulink. A complete reconfigurable system can be developed and simulated in Simulink environment. The use of IP cores allow designers to concentrate on the overall system design and performance without having to spend time to verify the correctness or performance of the system's components.

The FPGA boards include the Spartan 3 starter kit, the XUP-V2Pro, and the DO-ML403-EDK-ISE-PC4-US. The reconfigurable resources of these boards are used to implement the combinational and sequential logic needed to implement digital designs. It can also be used to

implement soft processor cores. The V2Pro board contains one V2pro FPGA chip with two PowerPC processors, Audio Codec, and connectors to Gigabit serial input/output. The DO-ML403 board contains a Virtex-4 FX chip that is optimized for embedded processing and supports both the PowerPC™ hard and MicroBlaze™ soft processors. These boards can be used to implement a Configurable system-on-a-chip (CSoC).

This paper provides the details of two capstone projects in the areas of digital communications and image processing. The digital communication project involves the hardware implementation of the various components of a communication systems including QAM Modulators, Raised Cosine Filter, Reed-Solomon and hamming code decoders, parity checkers, feedback and convolutional encoders, and soft and hard decision decoders. The image processing project involves the design of major components used in image compression algorithms such as JPEG and MPEG. This paper also details the challenges involved in using continually-evolving system-level design tools and the efforts made to reduce their learning times.

Challenges

Introducing system-level design tools into undergraduate courses pose many challenges to the instructors of these courses. These challenges include:

a) Number and functionalities of these tools: A large number of system-level tools are used. These tools are produced by Mathworks, Xilinx, and Mentor Graphics. These tools are used for design creation, functional and timing simulation, debugging, design synthesis, and to map, route, and download the design onto the implementation FPGA. Some of these tools are used to test the physical hardware produced.
b) Documentation sizes: As shown in Table 1, these tools have lengthy user guides, references, and getting started manuals.
c) Frequency of tools updates and modifications: currently, most of the software tools are being updated twice a year. FPGA chips and boards become obsolete in a few years.

	User Guide	Reference	Getting Started
Communication Toolbox	824		
Filter Design Toolbox	1786		
Fixed-point Toolbox	169	339	
Signal processing Toolbox	1043		71
Communication Blockset	246	626	94
Signal processing Blockset	1745		126
Xilinx System Generator	890		

Table 1: PDF Documentation size in pages

The above mentioned challenges reflect some of the problems facing the instructor of courses that use these design tools. It is extremely time consuming to develop good tutorials and meaningful laboratory exercises. Moreover, due to the high frequency of software updates and hardware advances, the developed tutorials and exercises need to be updated each semester. The only feasible solution to these problems is a regional or national collaboration between faculties to develop and update course materials for these classes.

Student Backgrounds

In the past few years, the EE department has started using Matlab, Simulink, and some of their associated toolboxes and blocksets in many junior and senior-level lecture and laboratory courses. These courses include signal and systems, control, and communication courses. Xilinx and Mentor Graphics software packages are being used in the computer-aided digital design course.

In the signal and system class (ECE 370), Matlab programs and Simulink models are used expensively to model, simulate and analyze analog and digital signals and systems, to perform a variety of operations and transformations on signals, and to design a variety of subsystems including FIR and IIR filters.

In the introduction to control systems and applications lecture and laboratory courses (EE470/477) students use Matlab programming, and Simulink models to analyze and plot complex signals, to solve the state and differential equations of a system, to find the a system's response to a particular input, to model sampled-data systems, to convert continuous-time systems to discrete-time systems and vice versa, and to design proportional, integral, and derivative (PID) controller for a system and use it to obtain a desired response. Student also use built-in functions, the response analysis GUI (the LTI viewer), and the interactive design tools of the Control System toolbox to analyze the response of a controlled system to pulse, step, and arbitrary inputs, and to view the root locus and the Bode plot of a system.

In the introduction to computer-aided designs class and laboratory (ECE 480/483) students learn a) the hardware description language (VHDL) and use it efficiently to model the design of a variety of applications, b) The development environment used to create and implement the design, and c) The main characteristics of the FPGA chip and board used for the physical implementation of the design. Students learn how to use the integrated software environment (ISE) from Xilinx to a) create the design, b) define the target FPGA chip and the operating frequency requirement, c) perform a functional simulation of the design, d) define the pin allocation of the FPGA chip, e) syntheses the design, f) find the amount of used hardware resources, g) create the bit map for the design and download it onto the FPGA chip, and g) test the design. In addition, students learn how to use the Core Generator, to instantiate IP cores in their designs, how to use simulation tools from Xilinx and Mentor Graphics for the functional and timing simulation of their designs, and how to use ChipScope to debug their designs. In its last offering of this course, students implemented their designs using the Spartan 3 FPGA boards.

In the introduction to communication systems lecture and laboratory classes (EE467/476), Matlab programs, Simulink models, Communication Toolbox, Communication Blockset, Filter Design Toolbox, Signal Processing Toolbox and Signal Processing Blockset are used extensively in the class assignments, laboratory experiments and projects. Matlab programs and built-in functions are used to: a) generate, analyze and visualize signals of varying probabilistic characteristics including signal sources, antipodal signal sets, random signals, and white and Gaussian noise signals, and to study the effects of noise on the performances of communication systems, b) analyze the responses of communication systems to the noise inherent in its real-world components, c) perform modulation, filtering, and demodulation operations d) study the

effect of sampling on both bit and symbol error rates and error statistics of communication systems, and e) investigate the effects of error detection and correction codes such as Hamming and cyclic codes on the overall performance of digital communication systems. The communication toolbox provides additional built-in functions and parameterized GUI needed for evaluating the performance of communication systems. It also provide behavioral analysis tools using graphical means that can produce error rate plots, scatter plots, and eye diagrams. Students also use Simulink, Communication Blockset and DSP block set to model, analyze and test the performance of a variety of communication systems and components. The class projects included the design of binary phase-shift keying (BPSK), Quadri-Phase shift keying (QPSK), and M-ary phase shift keying noisy signal modulators and BPSK, QPSK, and M-PSK and detectors to produce the transmitted signal (message).

Introductory Assignments

As shown in the previous section, students enroll in the capstone design course with different design experiences. All students know how to create Matlab programs and how to use Simulink in modeling systems. However, they may have different experiences in using Matlab toolboxes and blocksets. In addition, system-level design requires additional toolboxes such as the Xilinx SysGen. Therefore, to prepare student for using system-level tools in their capstone designs, the instructor needs to develop a set of detailed tutorials aimed at: a) familiarize students with the use and main characteristics of each individual tool, b) teach students how do these tools interface with each other?, c) teach students the overall advantages and disadvantages of these tools, and d) familiarize students with concepts and techniques relevant to fixed-point design such as sample rate, precision, dynamic range, quantization methods, overflow options, design latency, and hardware/software co-simulation. Students need to be able to evaluate the impacts of there design decisions on the quality of the produced design and on the cost of these decisions in terms of hardware resource utilization. Students need to know that hardware resources cost money and lowering the design costs is essential for companies to stay competitive in the market. They also need to know the extreme importance of hardware in the loop verification in complex systems design.

The set of introductory assignments consists of four laboratory assignments. These assignments are written in the form of detailed tutorials to accelerate students' learning curve. These assignments are implemented on both the Spartan-3 and the Virtex II-pro FPGA boards. This enables students to investigate the effects of using different FPGA architectures on the overall performance of the produced design. The details of theses experiments are as follows:

a) Lab #1: Fixed-point designs
   The adder/subtractor is a simple and basic circuit that is used in a wide variety of applications. In pervious classes, students have studied various techniques for designing such a circuit and have implemented it in many of their previous laboratory assignments using digital design tools. In this experiment, students use Xilinx blockset (part of the SysGen) and Simulink blocks to design an adder/subtractor circuit. The main goals of this experiment are: a) teach students how to use Simulink and the System Generator to design a circuit, subsystem, or a system, b) teach students how to interface Simulink blocks, which uses floating point arithmetic (double) to System generator blocks that use fixed-point format, c)

teach students how to find the dynamic range of the application and how use this information to determine the optimal fixed-point format in terms of number of bits used and the position of the binary point, d) help students understand the effects of quantization methods (truncate and round) and overflow options (saturate, wrap, and error flag) and evaluate their effects on both the performance of the circuit and hardware recourses utilization, e) Teach students how to use the resource estimator to calculate the cost of various hardware implementations of the circuit in terms of number of FPGA slices, number of flip flops, and number of FPGA look-up-tables used, f) how to generate synthesizable VHDL code for the design and how to use the ISE tools to simulate, synthesize, and implement the design on the target FPGA chip, and g) teach students how to use the RTL viewer (part of the ISE tool) to view the logic produced for each design option used.

b) Lab #2: Multiply-Accumulate (MAC) design
Multiply –accumulate circuits are used in a wide variety of DSP applications including the design of Finite Impulse Response (FIR) and Infinite Impulse Response (IIR) filters. In this experiment, students use the knowledge acquired in lab1 #1 design to create a more complex circuit such as a 10 x 8 MAC. In this design, the data input to the multipliers have widths of 10-bits and 8-bits of signed (tow's complement) data. In this case, the output of the multiplier is 18-bit wide. The accumulator can add up to the results of 256 multiplications, i.e., the width of the accumulator is 25 bit. Xilinx and Simulink blocks are used to implement the design. The outputs of analog sources are sampled and the sampled data are used as inputs to the multiplier. This allows students to experiment with various quantization methods and overflow options and to use the resource estimator to compute the hardware costs for each method and option used. For the Virtex board, the effects of using the embedded multipliers, instead of using look-up table multipliers, on both performance and resource utilization can also be investigated.

c) Lab3: MAC FIR Filter design

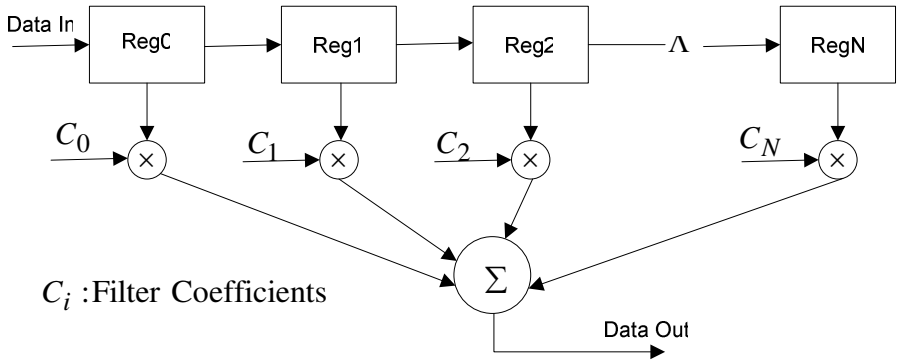


Figure 1: A Schematic of FIR filters

FIR filters are widely used in a wide variety of signal processing applications. A schematic of the structure of FIR is shown in Figure 1. In this experiment, students design and implement a MAC FIR bandpass filter, design using the SysGen filter design and analysis tool (FDATool). The Xilinx FDATool, which is a part of the Xilinx SysGen, provides a GUI to the FDATool package, which is part of the Signal Processing Toolbox). The FDATool

software allows user to design all common types of filters. This experiment will help students understand and utilize the capabilities of this tool including its quantization capabilities.
The FDATool accepts the filter specification, and type, and optionally the filter order. The tool produces the filter coefficients and optionally the minimum filter order needed to meet the specification. The generated coefficients are associated to a filter block in the Simulink model. The user defines the quantization and overflow options of the filter and simulates the FIR block using the true-bit simulation capabilities of the Simulink. The user can use the resource estimator to find out the amount of recourses needed to implement the filter using the specified FPGA chip. The SysGen generates the VHDL code needed for the ISE tools. The ISE tools synthesize the design, map it to the FPGA chip and generate the implementation netlist. The FPGA board can be connected to computer running the Simulink model to perform hardware-in-the loop verification. Alternatively, the functionality of both the simulated FIR filter (this is referred to as HDL co-simulation) and the hardware-implemented filter can be verified (this is referred to as hardware co-simulation) and compared to each other using the Simulink verification capabilities.
The user can experiment with a variety of quantization and overflow options in the design and compares the hardware resources needed and the corresponding clock rate when each of these options is employed.

d) Lab 4: hardware design verification

In this experiment, students design a MAC FIR filter with the same specifications as the one designed in lab 3. In this case, students use the Core Generator software system, which is a part of the ISE environment. They also use the filter coefficients produced in the previous lab after converting then to integer values. After generating the VHDL code for the FIR core, the user modifies its wrapper file to allow it to interface the SysGen blockset.
In order to incorporate a hardware description language block in the Simulink model, the Sysgen uses a black box block. In this case, the black box block is associated with developed FIR core. An HDL co-simulation and hardware co-simulation can be performed simultaneously to verify the functionality of the design.

The capstone course projects

The above assignments gives student teams the know-how and experience needed to design and implement more complex projects. Details of two of these projects follow.

The Image Processing Project
The image processing project involves the design of major components used in image compression algorithms such as JPEG and MPEG. These components include 1D and 2D Discrete-Cosine Transform (DCT), and 1D and 2D Inverse Discrete-Cosine Transform (IDCT).

a) 1-D DCT: This transform is a lossy compression scheme that decomposes converts the spatial domain waveform (audio input or one row/column of a Video input) into spatial components called DCT coefficients. These coefficients can be rounded or truncated to achieve the desired precision. The 1-D DCT transform can be defined using the following equations [1]:

$$X(k) = \alpha(k) \sum_{n=0}^{N-1} x(n) \cos\left(\frac{\pi(2n+1)k}{2N} \right) \qquad 0 \le k \le N-1 \tag{1}$$

Where

$$\alpha(0) = \sqrt{\frac{1}{N}}, \qquad \alpha(k) = \sqrt{\frac{2}{N}} \quad for\ 1 \le k \le N-1 \tag{2}$$

The DCT core can efficiently be designed using a technique called Distributed Arithmetic (DA) [2]. The values of the cosine terms are pre-computed for each number of point k. The DCT core can be designed using k FIR filters, each of which is used to generate a single DCT coefficient. The core can also be designed using the Core Generator software.

b) 1-D IDCT: The IDCT is the inverse of the DCT function. It converts the DCT coefficients into a signal. The 1-D IDCT can be defined using the following equations

$$x(n) = \sum_{k=0}^{N-1} \alpha(k) X(k) \cos\left(\frac{\pi(2n+1)k}{2N} \right) \qquad 0 \le k \le N-1 \tag{3}$$

Where

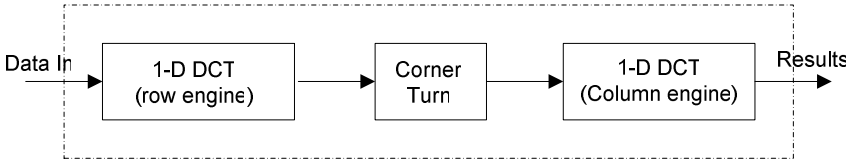
$$\alpha(0) = \sqrt{\frac{1}{N}}, \qquad \alpha(k) = \sqrt{\frac{2}{N}} \quad for\ 1 \le k \le N-1 \tag{4}$$

The 1-D IDCT core can be designed using the same techniques used for designing the 1D-DCT core.

c) 2-D DCT: The 2-D DCT is used to compute the DCT coefficients of a $N \times M$ sub-image. The algorithm for computing the coefficients is based on the following equation [3]

$$XC_{pq} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X N_{mn} \cdot \frac{c(p)c(q)}{4} \cdot \cos\left(\frac{(2m+1)\pi p}{2M} \right) \cdot \cos\left(\frac{(2n+1)\pi q}{2N} \right) \tag{5}$$

The algorithms can be implemented using two 1-D DCT cores as shown in Figure 2.



Block diagram to implement 2-D DCT

The 1-D DCT of each row of the sub-image is calculated and the computed 2-D matrix is fed into another 1-D DCT column wise. The 1-D DCT coefficients for the rows and columns can be calculated using the following equations

$$C = K \cdot \cos \frac{(2 \times (col\#+1)) \times row\# \times \pi}{2M} \tag{6}$$

Where

$$K = \sqrt{\frac{1}{N}} \quad for \quad row\# = 0, \qquad \alpha(k) = \sqrt{\frac{2}{N}} \quad for\, row\# \neq 0$$

And

$$C^t = K \cdot \cos \frac{(2 \times (row\#+1)) \times col\# \times \pi}{2N} \tag{7}$$

Where

$$K = \sqrt{\frac{1}{M}} \quad for \quad col\# = 0, \qquad \alpha(k) = \sqrt{\frac{2}{M}} \quad for\, col\# \neq 0$$

The results of 2-D DCT for an input $\mathbf{X}$ is given by $\mathbf{Y} = \mathbf{C} \cdot \mathbf{X} . \mathbf{C}^t$.

d) 2-D IDCT: This algorithm is used to decompress 2-D DCT compressed data. The 2-D IDCT core can be implemented using 1-D IDCT followed by a double buffer memory, followed by another 1D-IDCT [4] in a similar fashion to the implementation of 2-D DCT circuit.

The Communication Project

The digital communication project involves the hardware implementation of some of the components of communication systems including Raised Cosine Filter, Reed-Solomon decoders and encoders, Transposed Form FIR filters, and soft and hard decision decoders. The project also involves the hardware implementation of these system using FPGA boards.

a) Reed Solomon Encoder and Decoder: This decoder is used for forward error correction of transmitted data. This code is also designated as (n,k) codes, where k is the number of data symbols in the original message and $(n-k)$ is the number of check symbols that are added to the original message. The Reed-Solomon decoder samples n symbols of data and attempts to correct errors in the received message. The algorithm is guaranteed to correct a maximum of $\frac{(n-k)}{2}$ errors.

b) Pulse shaping filters: There is a wide-variety of pulse shaping filters including the Raised Cosine (RC) Pulse. The RC shaping technique eliminates some of the problems of the sinc function. In the frequency domain, the RC pulse can be defined as follows:

$$V(f) = \begin{cases} \sqrt{T} & 0 \le |f| \le \frac{1-\alpha}{2T} \\ \sqrt{T} \cos^2 \left(\frac{\pi T}{2\alpha} \left(f - \frac{1-\alpha}{2T} \right) \right) & \frac{1-\alpha}{2T} < |f| < \frac{1+\alpha}{2T} \\ 0 & otherwise \end{cases} \tag{8}$$

In the time domain, the RC pulse is defined by

$$v(t) = \frac{1}{\sqrt{T}} \frac{\sin\left(\pi t / T\right)}{\pi t / T} \frac{\cos(\alpha \pi t / T)}{1 - 4\alpha^2 t^2 / T^2} \tag{9}$$

Where α represent the excess band width needed to implement the pulse shaper.

c) Transposed Form FIR filters: In this structure, data samples are applied in parallel to all the tap multipliers through pipeline registers. The outputs of the multipliers can then be added using a pipeline adders. This approach is more beneficial for filters with large number of taps.

Comments and Observations

Our Introduction of system-level tools into upper division courses is still in its experimental phase. We are still developing more projects and refining existing ones. In addition, our Electrical Engineering department is small and our institution is a minority-serving university.

Student evaluations and feedback provided mixed reactions to the design projects. In one hand, students recognize the need for system-level designers, and the extreme values and future career rewards for such experiences. Many students were extremely enthusiastic to participate in such design projects and their evaluation comments reflected their appreciation of the opportunity given to them to gain such a valuable design experience. On the other hand other students felt overwhelmed by the sheer volume of documents that they may need to consult in order to experiment of options available in these tools. Detailed tutorials, extensive coaching, and clearly-defined objectives have helped reduce the effort needed for the successful implementation of these projects. However, some students felt uncomfortable about having to spend too much time working to meet the demands of **one** of their courses.

Student mixed comments and feedback were as expected. It is a fact that not every engineering student would like to have a career as a professional system designer, which is a very demanding job. Such a job require talented and highly-competent professional with a leadership quality and vision. The author will be very satisfied if these system-level design projects help train a few of such professional. To ensure achieving such results, we are planning to continue developing a diversified set of quality system-level projects, refining the contents of existing ones, and to improving the tutorials and handouts of used design tools. We are also working on improving the evaluation and assessment tools of these project courses.

References

1. N. Ahmed, T. Natarajan, and k. R. Rao, “Discrete Cosine Transform,” IEEE Trans. Computers, Vol. C-23, pp.90-94, 1974
2. S. A. White, “Application of Distributed Arithmetic to Digital Signal Processing,” IEEE ASSP magazine, Vol. 6(3), pp. 4-19, July 1989.
3. Application Note XAPP610: Video Compression Using DCT, v1.3, March 2005 at http://www.xilinx.com/bvdoc/appnotes/xapp610.pdf
4. Application Note XAPP611: Video Compression Using IDCT, v1.2, March 2005 at http://www.xilinx.com/bvdoc/appnotes/xapp611.pdf