AC 2009-2349: BALANCING VIRTUAL AND PHYSICAL PROTOTYPING ACROSS A MULTICOURSE VLSI/EMBEDDED-SYSTEMS/SOC DESIGN CURRICULUM

Mark McDermott, University of Texas, Austin

Jacob Abraham, University of Texas, Austin

Mihir Ravel, Franklin W. Olin College of Engineering

Balancing virtual and physical prototyping across a multi-course VLSI/Embedded-Systems/SoC Design curriculum

Introduction

With the advent of high performance computing platforms and design automation tools there has been a migration from physical prototyping of VLSI systems to virtual prototyping in both the industrial and educational environments. This move is attractive for many educational institutions as it is possible to have a "virtual" lab environment for a wide range of the curriculum that requires only computers and EDA software. This shift to virtual prototyping as the preferred method for teaching the design of integrated circuits and systems offers quicker iteration and exploration, but it leaves significant gaps in the intuitive and systematic design competencies gained from physically implementing and testing a complex electronic system. We have attempted to strike a balance between the two approaches, and this paper analyzes the lessons learned from our use of a common set of virtual and physical prototyping platforms for a four course graduate sequence in integrated circuit design and embedded system-on-chip (SoC) design.

Background and Motivation

A sequence of four graduate level courses was chosen for this analysis for three reasons: 1) the dependencies the courses have on laboratory based instruction, 2) applicability to the semiconductor industry and 3) each course builds upon the previous course culminating in a capstone course that unifies the systematic design competencies that are needed to build complex silicon systems. These silicon systems are composed of both hardware and software components that implement complex algorithms and functions, and these functions determine the competencies required by the student.

The four courses in the sequence are described in detail in the next section and include:

1) Basic VLSI Design
2) Advanced VLSI Design
3) Embedded Systems Architecture
4) System-on-Chip (SoC) Design

These courses were co-developed and are currently co-taught by full-time faculty and adjunct faculty from industry. There are a number of key benefits associated with using both full-time and adjunct faculty including timely access to state of the art teaching material, feedback on future directions in the design of complex silicon systems, support in developing new curriculum material and immediate feedback on the capabilities of the students. The course sequence has been taught in this format for a number of years. To support this approach with a maximum of efficiency yet allow teaching adaptability, the course sequence is being optimized to provide an "active learning" approach using a common set of platforms for both virtual and physical prototyping.

Course Sequence Organization

Course 1 - Basic VLSI Design
This course focuses on teaching the student the building blocks of VLSI systems. The students use the Weste & Harris book "CMOS VLSI Design: A Circuits and Systems Perspective" as reading material to support the lecture material. There are three laboratory assignments that provide the student with the necessary capabilities to design and layout CMOS integrated circuits using a virtual prototyping platform from Cadence Design Systems. There is a class project which requires the student to apply the knowledge of the three laboratory assignments to a real world application. These applications include generating functional elements and libraries that can be used as building blocks in a larger VLSI implementation. The project is reviewed by the course instructor(s).

Course 2 - Advanced VLSI Design
The second course focuses on the "Early Design Planning" of complex SoC platforms and feasibility analysis of critical circuits in the design. The students are required to do a class project in lieu of individual lab assignments. The class project is designed to be as "real-world" as possible utilizing a synthesizable open source Verilog model of a SoC as the design platform. The design platform undergoes detailed power and performance analysis, floorplanning and logic synthesis. The class project culminates in a full blown design review by the industry based adjunct faculty. In addition to providing excellent feedback for the students, it also provides the instructors feedback on how to augment the course material for subsequent semesters. This course recently received a Community Innovation Award from SUN Microsystems for "Best University level Computer Architecture and/or VLSI course".

Course 3 - Embedded Systems Architecture
The third course focuses on the HW/SW architectures of typical SoC platforms. These platforms are composed of hardware and software components which must be seamlessly integrated together to produce a working SoC. The course topics include: embedded processor architecture, hardware acceleration, embedded operating systems, driver development, power aware programming and testing of embedded systems. The classes are taught by both industry adjuncts and full-time faculty. There are three laboratory assignments and one class project assignment.

The laboratory assignments focus on developing an understanding of the physical prototyping platform (FPGA plus 32-bit processor) that the students will use to complete the class project. The TLL2020 platform (Figure 1) allows reconfiguration and interconnection of multiple processing fabrics for prototyping embedded digital logic machines (FPGA) and embedded software (ARM9 processor) [1]. The ARM9 based processor and large gate count FPGA communicate through a base system-board that also contains numerous multimedia and communications peripherals, interfaces, and system clocking/timing. The Freescale Semiconductor i.MX21 ARM9 processor was selected for its educational values of having 1) an easily understandable instruction set architecture, 2) broadly available Linux and GCC software, 3) an externally accessible 32-bit address and 32-bit data bus, and 4) a project relevant peripheral set including Ethernet, USB, serial communications and GPIO/LCD interfaces. The Xilinx Spartan-3 FPGA was selected for the educational values of 1) having a relatively lower architectural complexity coupled with reasonably large logic prototyping space, 2) freely downloadable design tools, 3) and a large community of open source IP cores and developers. The students decide which project to implement on the platform and are responsible for both the hardware and software design. The projects undergo both peer and instructor review at the end of the semester.

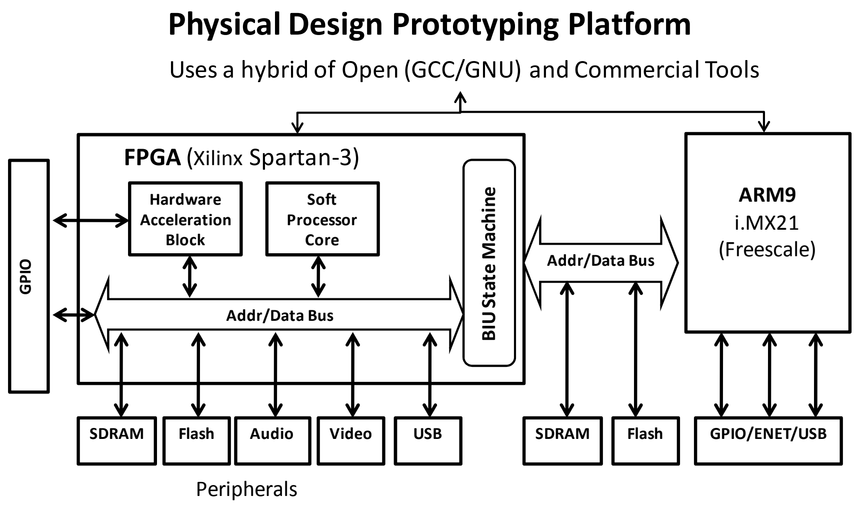

Figure 1.0 Physical Prototyping Platform

<u>Course 4 - System-on-Chip (SoC) Design</u>

The final course in the sequence is designed to give the student working knowledge of how to map a complex algorithm to a SoC implementation. The student analyzes hardware/software tradeoffs, algorithms, and architectures to optimize the SoC based on requirements and implementation constraints. The course material focuses on algorithmic mapping, transaction level modeling, performance analysis of HW/SW and hardware-software co-design and co-verification. There are three laboratory assignments that focus on a combination of virtual and physical prototyping techniques for building SoCs. The virtual prototyping environment consist of two distinct design methodologies: 1) algorithmic modeling in MATLAB/SPW and 2) virtual system prototyping using COMET from Vast Systems Technology (Figure 2.0). Both design methodologies require a fair amount of time to become proficient and special effort is taken to limit the amount of valuable class time relegated to "teaching the tools". This is accomplished by providing critical starting points for each lab exercise and limiting the amount of tool experimentation that the student can do while meeting the pedagogical goals of the courses.

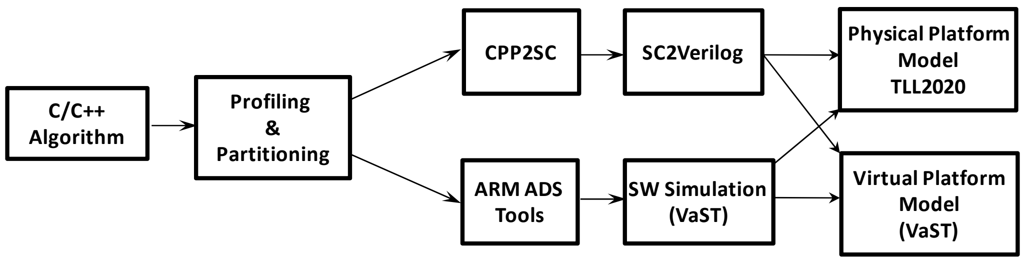

Figure 2.0 Virtual Prototyping Platform

Competencies

The course sequence described above has been designed to give the graduate student a number of competencies. These competencies are analyzed below according to the type of coursework and the optimal laboratory/project environment (virtual vs. physical). The list of competencies analyzed includes:

- Requirements gathering, design planning, cost modeling, energy modeling, robustness modeling, complexity analysis and reusability
- System partitioning, algorithmic mapping & transformation
- Models of Computation, Communication and Abstraction
- Software Architecture and Design including: Assembly Language Programming and High Level Language Programming
- Hardware Architecture and Design including: Register Transfer Level, Block Level and Component Level
- Physical planning and design
- Performance modeling and analysis
- System Integration
- System Verification and Validation

The type of coursework is determined depending on whether the student is required to have a conceptual understanding and/or a practicing knowledge of the subject material. The optimal balance across virtual and physical prototyping is determined by the desired competencies. Based on the goal of encouraging systematic engineering competencies, we analyzed the list above and generated an importance scorecard that is summarized in the table below.

Competencies	**Coursework**		**Prototyping Model**	
	Concepts	**Practice**	**Virtual**	**Physical**
Requirements Definition	8	6	8	1
System Partitioning & Tradeoffs	8	5	8	3
Complexity Analysis	7	6	8	1
Mapping & Transformation	8	6	7	7
Algorithmic Design	5	1	3	1
SW Architecture and Design	6	6	7	7
HW Architecture and Design	8	4	8	8
IC Layout Design	2	6	8	1
HW/SW Co-Verification	8	5	8	6
Multi-metric Modeling	8	4	9	2
Performance Modeling and Analysis	8	8	6	8
System Debug	4	8	7	5
System Integration	8	8	3	8
System Validation	5	8	3	8

SCALE: From 1 – minimal importance to 10 – very important

The coursework scoring is based on input from faculty members who teach the four course sequence while the prototyping model scoring is based on feedback from industry, students and faculty. To interpret the results of the scoring one needs to consider the two adjacent columns under each category. In general when both columns have similar values the relative importance is the same. A comparison between two rows is somewhat subjective. Feedback from students on the relative importance of each competency is dependent on their individual interests and course of studies. Some of this feedback is highlighted in the next section.

Analysis

The analysis of the coursework scoring shows a balance between learning concepts and gaining practical knowledge for those areas which are easy to accomplish in a single semester. There are often proposals to add "practical classes" to the curriculum in order to provide more active learning and augment the conceptual learning. This is in large part a response to a feedback from industrial managers who argue that students should know a particular tool or technique to be rapidly productive upon joining the work force. Balancing coursework between conceptual and practical knowledge is a topic for another paper and will be not discussed here. In this paper we would like to focus on whether there is particular merit between using a virtual prototype or a physical prototype for reinforcing practical knowledge of the competencies listed above.

The analysis of the scoring of the prototyping model indicates that "frontend" activities such as requirements, system partitioning and architecture design are best done using virtual models of the system. The opposite is true for "backend" activities like system integration, system verification, where physical prototypes are most useful. This is not a surprising outcome for a number of reasons:

- Virtual prototyping allows rapid exploration of complex scenarios while allowing "viewing" of internal design parameters that are difficult or impossible to access in physical prototyping.
- The economics of doing multiple physical prototypes as part of frontend tradeoff activities is expensive for industrial applications and even more so in the educational environment.
- As you would expect his is exactly what industry does from a methodological perspective and what it looks for when it hires new students. Frontend design teams look for skills in architectural modeling and partitioning, whereas backend teams look for students with practical hands-on knowledge.
- Virtual prototyping is an abstraction process and will inherently miss many physical realities such as system noise, nonlinearities, loading, power distribution errors, and interference that physical prototyping readily highlights. This is a powerful reminder to students that simulation and modeling are only preliminary steps to successful engineering systems.

The following diagram illustrates a typical prototyping environment as it transitions from virtual to physical as a function of the design abstraction level. The rate of the transition is a function of the available design automation tools and applicable prototyping hardware. [2] [3]

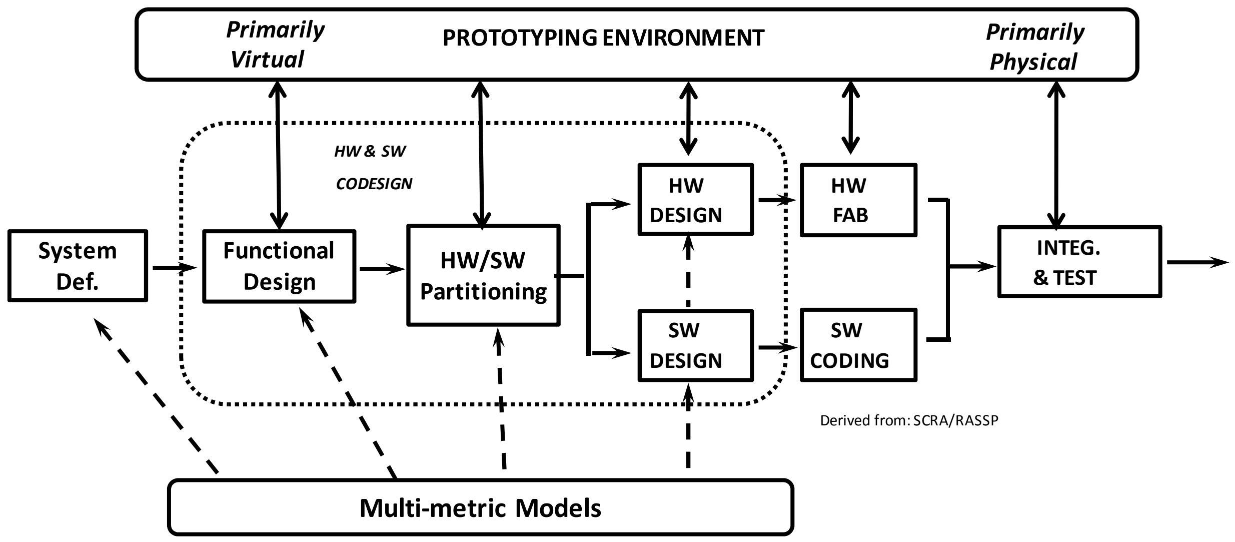

Figure 3.0 Typical Prototyping Platform

It should be noted that the embedded systems ecosystem is in the process of migrating from developing application specific integrated circuits to platform based design. [4] The availability of these architecture specific platforms for physically prototyping an embedded system will provide more opportunity to balance the mix between virtual and physical prototyping. [5]

Student Feedback

As with most academic institutions feedback is routinely solicited from the students at the end of each semester in the ongoing attempt to improve the curriculum. This feedback has been summarized below:

- Virtual prototyping is effective if the learning curve to use the design automation tools is limited to less than half a semester. The "models" used by the tools must also be error free.
- More students prefer virtual prototyping as it provides the ability to work from anywhere in the "world". Immediate access to the physical platforms is the number one complaint of most student teams.
- Debugging problems in the virtual environment is preferred.
- The design automation tools for physical prototyping are considerably behind in overall capability. Many students prefer "Visual" tools to command line and "makefile" based tools.
- Software development is much easier on a physical platform. Booting an OS on a virtual platform can take hours or even days.
- Physical prototyping is effective if the "bugs" that are detected are real world and not a side effect of the platform, i.e., many FPGAs have artifacts which are not seen in actual SoC implementations.

Industry Feedback

Industry feedback was obtained from two sources: 1) the adjunct faculty who in many cases will hire students taking their classes and 2) hiring managers who contact the faculty looking for specific competencies. Their feedback is summarized below:

- Students need to know the basics first. Lab assignments are good but team oriented projects are essential. The most important aspect of the student design experience is the practice of reflection which occurs when the class projects are reviewed by peers and industry based faculty.
- “Pre-silicon” hiring managers look for students with more virtual prototyping competencies specifically the area of RTL and ESL languages, synthesis and formal verification techniques.
- “Post silicon” hiring managers look for students with hands-on experience with logic analyzers, hardware bread-boarding, hardware/software co-design and co-verification.

Summary

The motivation for this analysis was in direct response to a report from The National Academy of Engineering "Training the Engineer of 2020" which calls for the engineer of the future to exhibit “practical ingenuity”. This will require an approach to learning in an engineering educational environment where the student learns a systematic approach to engineering design. This approach includes: establishing metric-based objectives, structure, planning, designing, prototyping, performance evaluation, and aligning outcomes to a desired objective. Sensible use of virtual prototyping allows rapid exploration of the design space while physical prototyping exposes the students to uncertainties of real world systems that are missing in abstracted virtual modeling. The balance of virtual and prototyping tools in a systematic learning environment has been found to be valuable in meeting the objective of “practical ingenuity” in graduating engineers.

References

[1] Mihir Ravel, Mark McDermott, "*An Electronic System Design Platform for SYSTEMatic Learning in ECE and ICT Curriculum*," MSE07, pp.145-146, 2007 IEEE International Conference on Microelectronic Systems Education (MSE'07), 2007

[2] Mark A. Richards, Anthony J. Gadient, Geoffrey A. Frank, “*Rapid Prototyping of Application Specific Signal Processors*”, Springer; 1st edition (February 28, 1997) ISBN-10: 0792398718

[3] Vijay K. Madisetti, Thomas W. Egolf, “*Virtual Prototyping of Microcontroller-Based Embedded DSP Systems*”, IEEE Micro, Volume 15, Issue 5 (October 1995), Pages 9-21

[4] Alberto Sangiovanni-Vincentelli, "*Defining Platform-based Design*". *EEDesign of EETimes*, February 2002.

[5] Alberto Sangiovanni-Vincentelli, et al, “*Benefits and challenges for platform-based design*”, Proceedings of the 41st annual conference on Design automation, (2004), San Diego, CA, USA Pages: 409 - 414