2016

A Spiral Computer Engineering Lab Framework

Pong P. Chu
Cleveland State University, p.chu@csuohio.edu

Chansu Yu
Cleveland State University, c.yu91@csuohio.edu

Karla Hamlen-Mansour
Cleveland State University

A Spiral Computer Engineering Lab Framework

Pong P. Chu and Chansu Yu
Dept. of Electrical Engineering and Computer Science

Karla Hamlen-Mansour
Dept. of Curriculum & Foundations
Cleveland State University, Cleveland, OH 44115

Abstract

This development establishes a "spiral" lab framework, in which the key concepts are revisited at increasing levels of sophistication and interconnection, for Computer Engineering curriculum. The development is addressing a serious deficiency – lack of integration skill – in engineering curriculum. The framework connects and integrates the individual courses through a coherent sequence of lab experiments and projects across the entire curriculum. These lab experiments and projects are designed to illustrate and reinforce key theoretical concepts and their complexities and abstraction levels gradually grow as students progress through the curriculum. The framework is based in three cohesive themes: video, audio, and touch sensor, and individual projects are eventually evolved into a complete set of IP (intellectual property) cores that form customizable I/O subsystems and can be incorporated into any FPGA (field programmable gate array) based computer system. The development uses low-cost prototyping board and can be easily incorporated into existing curriculum. The improved curriculum will enhance students' integration skill, make them aware of the big context, and keep them interested and motivated. This project is currently implemented and evaluated in two institutions in parallel and two new book manuscripts with the developed materials are under preparation.

1. Need and Goal

Recent reports on engineering education call for change to enhance the learning of students and to prepare graduates to meet the challenge of the new century [1,2,3,4]. A good engineer is an expert who has a deep understanding of a domain and can apply the knowledge to solve novel problems. Becoming an expert requires two types of practices – the "component skill," which is the knowledge in specific areas, and the "integration skill," which applies and integrates component skill to address complex and realistic problems [3]. The Carnegie Foundation for the Advancement of Teaching conducted a five-year study of engineering education and reported the results in a book titled "*Educating Engineer: Designing for the Future of the Field"* [4]. It points out that one deficiency of the traditional engineering curricula is that they mainly focus on the component skill, in which each subject is taught in isolation and without proper context, and do not adequately prepare students for the integration skill. One recommendation of the study is to use a "spiral model" to provide more effective learning experiences:

"… the ideal learning trajectory is a spiral, with all components revisited at increasing levels of sophistication and interconnection. Learning in one area supports learning in another."

Incorporating hands-on exercises can greatly enhance students' learning experiences. It is particularly true for engineering since developing and designing new components and systems are the basic characteristics of this field. Many courses in computer engineering contain lab experiments and projects. However, the labs are generally treated as adjuncts that follow the learning of the theories and are presented in a limited "component context." The Educating Engineer study calls the lab a missed opportunity and states that [4]:

"…[The labs] can be more effectively used in the curriculum to support integration and synthesis of knowledge, development of persistence, skills in formulating and solving problems, and skills of collaboration. Design projects offer opportunities to approximate professional practice, with its concerns for social implications; integrate and synthesize knowledge; and develop skills of persistence, creativity, and teamwork."

This development is motivated by the study and follows the recommended "spiral model" to provide more effective learning experiences. It develops a "spiral" lab framework that contains a series of connected, theme-based experiments and projects weaving through the entire curriculum, from freshman engineering to senior capstone design. The goal is to help students to see the "big picture" of the discipline and enhance their "integration skills."

2. Project Description

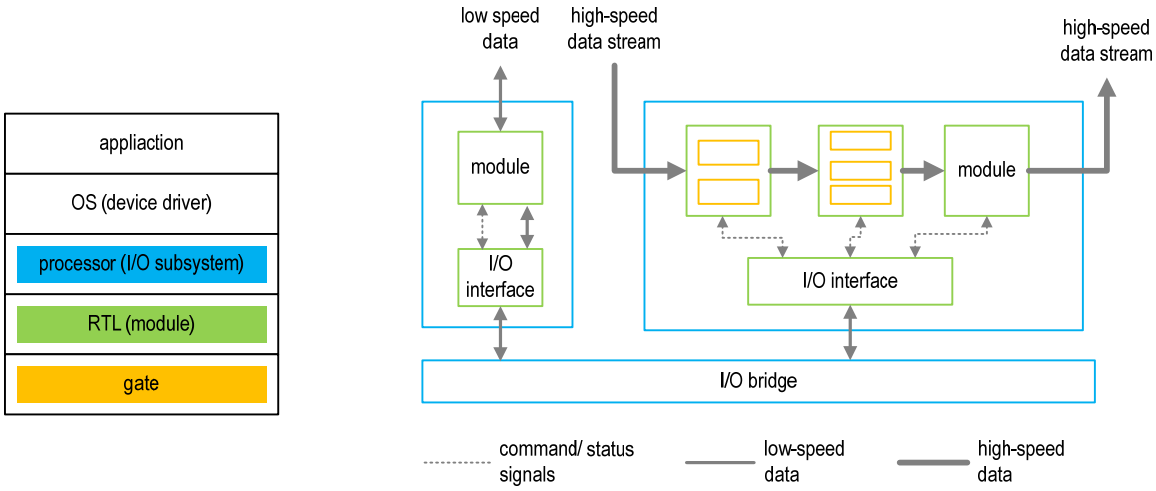
2.1. Layered model of computer engineering

To manage complexity, a computer system is described by a layered model. A simple model is shown in Figure 1(a). The computer engineering curriculum is basically organized according to the abstraction layers of the model. A course is usually focused in a single layer and studies the materials in great detail. These courses are taught as independent topics and emphasize the "component skill." Thus, the courses are mostly presented as isolated and disconnected pieces. While students learn the specific details of each layer, they do not have a comprehension of the relationships among the layers. This leads to the lack of integration skill observed in [4].

This project connects and integrates the individual courses through a cohesive lab framework. The framework is to construct a complete I/O subsystem that contains a collection of IP cores of general-purpose peripherals and hardware accelerators. Combining these cores with a processor and memory forms a complete system, which resembles a high-end embedded system. An I/O IP core is constructed according to the layered model in Figure 1(a) and built in a bottom-up fashion. In the gate layer, the circuits are constructed with small and medium sized logic components. In the RTL layer, the circuits are assembled to form a larger module. In the processor layer, an I/O module is augmented with additional decoding circuit and buffering registers to interface with the system bus and to communicate with the processor. The three layers and their corresponding parts are depicted in Figure 1(b). In the OS layer, the software driver routines are derived to access the IP cores.

The two general I/O IP core architectures are shown in Figure 1(b). The block diagram on the left represents a simple subsystem, such as the LED matrix control

core. It consists of a basic module, which accesses and controls an external device, and a bus interface module, which communicates with the processor. The block diagram on the right represents the sound and video cores. It can be treated as a hardware accelerator with separated data flow and control flow. The high-speed data stream is processed with customized computation modules and routed through the dedicated paths. The control is connected via a similar bus interface module.



(a) abstraction layer (b) basic I/O core model

Figure 1. Layered I/O core model

The lab framework keeps lecture content intact but updates the experiments and projects to make students aware of the big picture, help them relate the individual subjects, and apply and integrate the previous learning in a new context.

2.2. Theme based lab framework

The lab framework's IP cores are built around the "themes" of *video*, *sound*, and *touch sensor*. Each theme grows in two dimensions: *component complexity* and *abstraction level*. The component dimension represents the I/O devices and peripherals with increasing complexity. The abstraction level follows the layered computer system model in Figure 1(a), which includes gate, register transfer, processor, operating system, and application layers. The experiments and projects are designed to illustrate and reinforce key theoretical concepts in various courses. Their complexities and abstraction levels gradually grow as students progress through the curriculum.

These themes are spread over the entire curriculum and woven together from an introductory freshman engineering to senior capstone design course. Key concepts are repeated in different courses with increasing complexity and sophistication and studied from different aspects and contexts, such as software implementation versus hardware implementation, gate-level design versus system-level integration, etc. Components in the themes are designed from scratch at the gate level and eventually evolve into customized IP cores that can be physically implemented and verified in an FPGA

prototyping board. Because of the page limitation, only the video theme is discussed in the following subsections. The other two themes are developed in a similar manner.

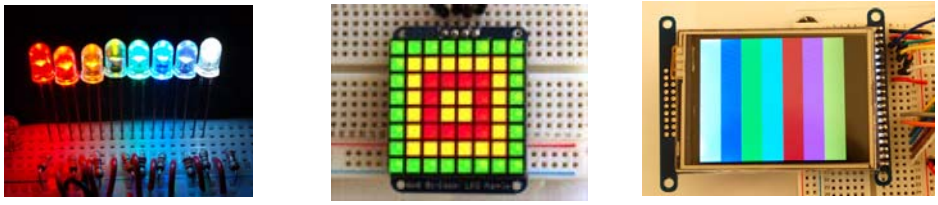
2.3. Video theme components

The video theme involves the display and acquisition of an image or video. A computer image is composed of a matrix of pixels. A pixel contains three primary colors (red, green, and blue) and different colors can be obtained by adjusting the intensities of the primary colors. For example, in the "VGA mode," an image is composed of a 640-by-480 matrix (i.e., 640 pixels in a row and 480 pixels in a column) and has about 300 thousand (i.e., 640*480) pixels.

The devices in the theme gradually increase the number of pixels within the display:

- One-pixel devices: a tri-color LED and a photo resistor.
- 64-pixel device: an 8-by-8 LED matrix representing a display with 64 pixels.
- Low-resolution devices: a low-resolution (e.g., 320-by-240 pixels) TFT (thin film transistor) LCD display module and a low-resolution camera module.
- High-resolution devices: a high-resolution LCD display (VGA resolution or better).

Some devices are shown in Figure 2.



(a) Discrete LEDs (b) LED matrix (c) TFT display

Figure 2. Some video theme devices

2.4. Video theme IP cores

The block diagram of the video theme portion is shown in Figure 3. It contains a PWM core to control the tri-color LED, a multiplexing module to control the LED matrix, an SPI core to interface the TFT, and a complete video core for the LCD display and camera. The video core is more sophisticated and its diagram is shown in the bottom.

2.5. Video theme experiments and projects

We divide the lab into four levels. The video theme experiments and lab projects in each level are

- **Level 1: Freshman engineering**. Develop software for microcontroller to set the color of LED with PWM, to measure the light intensity via photo resistor, and to implement a time-multiplexing scheme to control the LED matrix.
- **Level 2: Basic digital system.**
  - Repeat the experiments in Level 1 but implement them with digital circuit (the PWM core and LED multiplexing core).
  - Design synchronization, line buffer, and test pattern modules of video core.
  - Design the SPI core to control the TFT display.

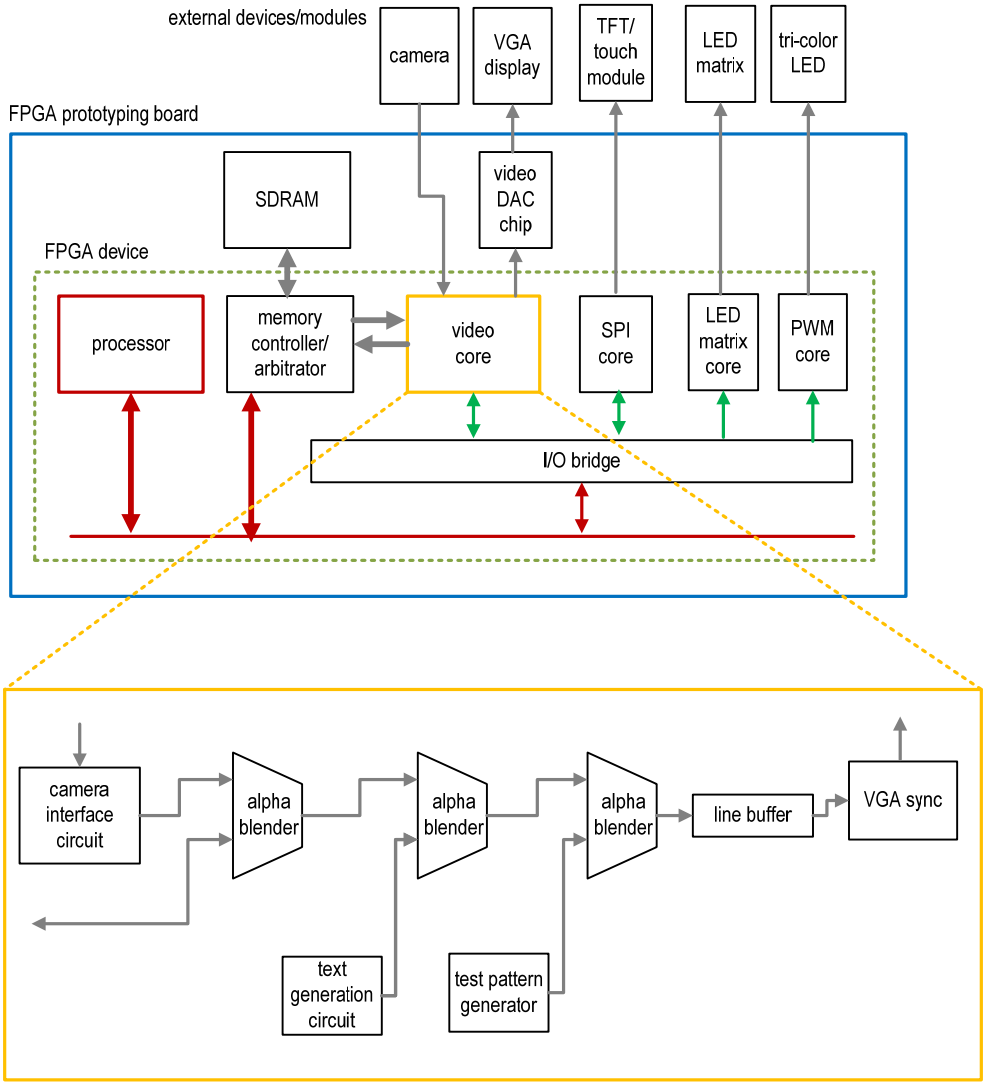


Figure 3. Block diagram of video theme system

- **Level 3: Advanced digital system (hardware-software co-design).**
  - Design alpha blender, text generation, camera interface, and frame buffer modules of the video core.
  - Develop I/O bridge and core interface circuits.
  - Complete all I/O IP cores.
  - Integrate the I/O cores with a processor module to form a complete embedded system.
  - Develop software drivers for I/O cores.
- **Level 4: Term or senior capstone projects**. Implement a video based project, such as video games with light and/or force sensor, “sprite” based graphics system, 2D graphic accelerator, video-based motion detector, and hardware implementation of image processing and machine vision algorithms.

3. Outcomes and Evaluation

The project overhauls the lab portion of the computer engineering curriculum and replaces the isolated and scattered projects with a single cohesive theme-based framework. A series of lecture notes, tutorials, experiments, and projects are being developed and will be made available to other institutions. Two new book manuscripts with the developed materials are under preparation.

The project is currently implemented in two institutions in parallel and its effectiveness is evaluated by an array of assessment instruments, as shown in the table below.

	Content tests	Student survey	Interviews (qualitative)	Student lab work	Course materials
Student knowledge	x			x	
Student interest		x	x		
Student perception		x	x		
Instructor perception			x		
External review (at CBU)					x

The data collection is in progress. After the completion, comparisons will be made between gain scores of each class, survey and interview differences, as well as any differences in available formative course assessments, such as student homework and participation. If the sample is diverse enough, we will also examine which curriculum is more effective for high- vs. low-achieving students, as well as differences in the effectiveness of the curriculum based on gender and other demographic factors if available.

4. Broader Impacts

Because of the flexible adoption path and low cost, the proposed lab work can be easily incorporated into any existing computer engineering curriculum. The improvement will enhance students' integration skill, make them aware of the big context, and keep them interested and motivated. The PI's previous NSF grant led to two books that use a project-based approach to introduce hardware design. They are well received and used in many institutions. It is likely that these institutions will adopt some themes for their curricula and there will be immediate impacts. The new book should attract a larger audience since the lab framework constitutes a complete computer system with customizable I/O cores and hardware accelerators. Many interesting hardware and software experiments and projects can be developed.

Bibliography

[1]. S. A. Ambrose et al., How Learning Works: Seven Research-Based Principles for Smart Teaching. Jossey-Bass, 2010.

[2]. C. J. Atman, et al., *Enabling Engineering Student Success: The Final Report for the Center for the Advancement of Engineering Education*, 2010.

[3]. J. E. Froyd and M. W. Ohland, “Integrated Engineering Curricula,” *Journal of Engineering Education*, pp.147-164, January 2005.

[4]. S. Sheppard, et al., *Educating Engineers: Designing for the Future of the Field*. Jossey-Bass, 2009.