VLSI-1 Project Ideas

Fall 2008

1.0 Introduction

Your task in this project is to use the skills you have been acquiring through the lectures and labs to design a fairly sophisticated module—an intellectual property (IP) core.

The purpose of the project is threefold:

1. It is worth a large fraction of your grade (but this should be the least important item);
2. working on this project should be training on how to go about approaching a design project; and
3. the project should yield results:
    - Experimental results on the efficacy of proposed VLSI architectures, and
    - Suggestions for improving these architectures.

Projects can be done individually, or in groups of 2–3; naturally, I will expect more from group projects.

2.0 Timeline

I do not want a student going off on a tangent, only to learn at the end of the semester that this happened. On the other hand, I don't want to stifle creativity by monitoring things too closely. Most of all, **I do not want rush jobs**, where everything is crammed into a few days at the end of the semester. You need time to develop these projects.

Project selection: You should make a decision as to the projects you are interested in working on and turn in hardcopy in my office of your specifications document and a timeline by **Friday, 10.17.2008, 11:59am**.

Intermediate reports: I would like you to turn in hardcopy of a design document by **Thursday, 11.6.2008**, in class.

Final report: The report is due in hardcopy on **Friday, 12.5.2008 by NOON (11:59.59AM)** in my office.

3.0 Final Report details

The intention of a report to generate a document that would normally be given to a customer (or your boss) as part of your project deliverables. The final report should consist of the following, which you think of as the individual chapters:

- Specifications document
- (Optional) Marketing document
- Design document
- User document
- Testing strategy and results
- Optimization strategy and results
- Source code and layout

3.1 Specifications

The specifications document should include a high-level overview of the IP block you are implementing; a description based on a diagram or set of diagrams is the best way to do this. It should also include a summary of the logical interface the block presents to its environment.

In addition, the document should include the area, power, and performance numbers you are targeting. If you base your work on an existing design, you should be able to come up with estimates on these parameters; otherwise, back-of-the-envelope calculations are fine. It's not imperative that you meet the numbers in the specification document.

The specifications document should not discuss the implementation; its focus is the functionality that you will implement, and the cost of this functionality.

3.2 Marketing

Optionally, you can include a marketing document. This should include an estimate of how many chips/cores you expect to sell, what price people will pay for them, how much they will cost to design and build, and how you will get customers to know about your product.

The text has a good discussion of design economics in Chapter 8, especially Section 8.5. EEtimes and Dataquest are other standard places to get information from.

3.3 Design

The design document should include a description of how you will implement the specification—a set of figures is the best way to convey this. The implementation discussion should include the basic architecture and algorithms, as well as the floorplan, and circuit technology, etc.

You should also make notes on the optimization techniques you expect to use and their implications to your design, and the trade-offs they will entail. For example, if you have long interconnects, you may want to state that you intend to overcome problems resulting from crosstalk by shielding, and hence all long nets should have enough space between them for such shielding lines.

All choices should be justified, on the basis of references to portions of the book/research papers, and by logical arguments.

The design document should also include an overview of the tool suite you will be using, the naming conventions for variables/modules/files, the regression control strategy,[1] and an issue tracking mechanism (which could be just entries in a text file).

Think of the design document as something you would give to an engineer just joining the project to help him/her come up to speed. (Design documents also spell out a regular system of "code reviews," where designers have to explain what they have done to their colleagues, at a very detailed level, e.g., a walk-through of RTL code. We won't have review process is probably too involved for a class project.)

The specifications and design documents do not have to be exactly what you turned in; indeed I would expect the design document to evolve as you discover problems and find improvements with your approach.

3.4 User document

The user document describes how end-users are to integrate the IP block into their designs—think of it as being like the datasheet you get with a chip.

In particular, the user document should include detailed information on interfacing to the block, i.e., the timing on the different signals. It should describe the power, area, delay numbers at various operating points, and the loading capacitance and drive strengths on the input-output signals.

3.5 Testing

In this chapter, you are to describe the set of tests you applied to your design to check for logical errors, and your coverage metrics. Classify the bugs you encountered, and how you corrected for them. In addition, discuss the traces you applied to determine the critical path, and compute the delays.

For some projects it may make sense to write a high-level model in C or C++ and do performance simulations (e.g., determine the average latency and drop rate through the Benes fabric as a function of load, and buffering). If this is the case, include results from these simulations.

---

[1] CVS is the tool of choice. There is a very brief tutorial on using CVS which you can read at http://users.ece.utexas.edu/~adnan/cvs_notes.txt

3.6 Optimization

Include a discussion of all the steps you took to improve performance, and the magnitude of improvements that you saw. I am particularly interested in novel techniques that gave your better performance that the descriptions that you based your approach on.

4 Evaluation

Your grade on the project will be based on a number of factors, particularly the originality and quality of the work. Other considerations include clarity of the written report, attention to detail.

You may want to read Prof. Adnan's notes on technical writing to avoid common mistakes that engineers regularly make in writing:
www.ece.utexas.edu/~adnan/writing.html

5.0 Project ideas

Below, is a list of projects that may be of interest.[2] Bear in mind that the project description is not complete, and you are responsible for making reasonable assumptions and decisions about the project.

5.1 Digital PLL

Phase-lock loops (PLLs) are used to recover timing information from a signal—they are ubiquitous in communications, and are also used for timing recovery on boards and chips. Analog PLLs are very hard to design because they use feedback, and are very sensitive to noise and operating parameters.

The goal of this project is to design an "all digital PLL" which is an implementation of the PLL with all digital components and compare its performance (measured in lock time and phase noise) and costs (in terms of area, power, delay) to a traditional analog PLL.

References:

- www.cs.wright.edu/~jstephen/ee737/ResearchPapers/DeLong.doc
- CMOS Circuit Design, Layout, and Simulation, by R. Jacob Baker, Harry W.Li and David E. Boyce, Published by IEEE
- R. B. Staszewski, C. Hung, K. Maggio, J. Wallberg D. Leipold, P. T. Balsara, "All-Digital Phase-Domain TX Frequency Synthesizer for Bluetooth Radios in 0.13um CMOS

[2]You are welcome to suggest your own project; however it must meet with my approval.

- R. B. Staszewski et al., “A First Digitally-Controlled Oscillator in a Deep-Submicron CMOS Process for Multi-GHz Wireless Applications,” Dig. RFIC Symp., pp. 81–84, June 2003.
- R. B. Staszewski et al., “Digitally-Controlled Oscillator (DCO)-Based Architecture for RF Frequency Synthesis in a Deep-Submicron CMOS Pprocess,” Trans. on Circuits and Systems II, vol. 50, no. 11, pp. 815-828, Nov. 2003.

5.2 On-silicon delay characterization

As variability increases, there is growing interesting in making adaptive chips, where parameters such as supply voltage and body biases can be set post-manufacturing to overcome the effects of parametric variation.

The goal of this project is to study the cost and accuracy of on-chip delay characterization structures. I’d like you to survey the state-of-the-art, as well as perform your own experiments.

For example, Dhar *et al.* introduce an adaptive voltage scaling controller that uses an inexpensive ring oscillator to measure speed. There could be multiple ring oscillators placed throughout the design. The gate delay would be approximated based on the delay of the nearest ring oscillator.

Another promising approach would be to implement delay characterization based on Razor by Ernst *et al.* By using a shadow latch and comparator logic, Razor has mechanisms to monitor when a delay error has taken place. In the context of an FPGA, a test input could run through the CLB in successively faster clock cycles until there is a delay error. Additionally, neighboring CLBs could perform the shadow latching and comparator logic need for Razor testing using existing CLB resources.

- S. Dhar, D. Maksimović, and B. Kranzen. Closed-loop adaptive voltage scaling controller for standard-cell ASICs. *International Symposium on Low Power Electronics and Design*, pages 103–107, 2002.
- D. Ernst, S. Das, S. Lee, D. Blaauw, T. Austin, T. Mudge, N. Kim, and K. Flautner. Razor: Circuit-Level Correction of Timing Errors for Low-Power Operation. *IEEE Micro*, 24(6):10–20, 2004.

The test literature (International Test Conference, Fault-Tolerant Computing) would also be a good place review.

5.3 Pattern matching

In this project, you will get an in-depth understanding of the VLSI design of modern on-chip interconnection network. To begin with, the following article serves as a good introduction: “Architectural Choices in Large Scale ATM Switches,” J. Turner and N. Yamanaka, IEICE Transactions, 1998.

The major task of this project is to select and implement a switching architecture. For instance, in the article above, a Batcher-Banyan based, self-routing network is chosen.

Many new techniques have been proposed; please spend proportional time on selecting among them. You are encouraged to invent new architectures and algorithms and analyze their strength and drawback.

Here are some more articles that may be useful:

- A 250-Mbit/s CMOS Crosspoint Switch. Shin and Hodges, IEEE JSSC 24(2), April 1989, pp. 478-486.
- A 250-Mb/s CMOS Crosspoint LSI for ATM Switching. Akata et al., IEEE JSSC 25(6), December 1990, pp. 1433-1439.
- A High-speed CMOS Circuit for 1.2-Gb/s 16x16 ATM Switching. Chemarin et al., IEEE JSSC 27(7), July 1992, pp. 1116-1120.
- A 200Mhz CMOS Broad-Band Switching Chip. O'Neill et al., IEEE JSSC 28(3), March 1993, pp. 269-275.
- Please actively search/Google for new ideas and build upon them!

Once the architecture is matured, you may employ the skills developed in our labs to implement a prototype (physical level). In view the limited time, you may put most of the efforts on core algorithm and structure and size down the whole system. Please consider how to establish your testing benchmark of your switch from the very beginning. Again, your testing benchmark should have fairly good coverage. As to the benchmark setup, you may use C/C++, or scripture languages like TCL/TK, PERL, etc. As this is more in the flavor of an open topic, your final grade will be based on your ideas, implementation workload, and testing mechanisms, etc. Especially, your implementation should demonstrate fair workload worthy of a serious project in our graduate class.

5.4 Sub-threshold PLA generator

You can operate a design with supply voltages below the device V_T. This can save power, since it reduces leakage as well as the $C \cdot V^2$ switching energy.

The goal of this project is to design a sub-threshold PLA generator using output from Synopsys Design Compiler and/or Espresso. The generator will generate a placement file for the Cadence layout tool. You should develop a power modeling tool which can be used to predict how much energy will be needed for various configurations of the PLA.

To test out the PLA generator you will need to synthesize various control logic blocks from the SUN Niagara core. The Verilog models for the SUN Niagara core can be found in:

http://projects.ece.utexas.edu/courses/spring_08/ee382m-16615/vlsi/main/project/s1_core/hdl/rtl/sparc_core/

- A 180-mV Subthreshold FFT Processor Using a Minimum Energy Design Methodology. A. Wang and A. Chandrakasan. IEEE Journal of Solid State Circuits, Jan. 2005.
- Computing with subthreshold leakage: device/circuit/architecture co-design for ultralow-power subthreshold operation. A. Raychowdhury, B. C. Paul, S. Bhunia, and K. Roy. IEEE Transactions on VLSI Systems 13(11).

5.5 TuneFPGA

For this project, you will implement a dual-Vdd FPGA in hardware. Each FPGA CLB will have a mechanism to select either a high-Vdd power supply or a low-Vdd power supply. First, you will need to run schematic level simulations to determine the best implementation of the tuneable FPGA CLB. Then, you will implement the resulting FPGA in hardware. Depending on the size of the group, this project should also implement a complete FPGA architecture, including a routing network, configuration programming infrastructure, a clock network, and reset logic.

- TuneFPGA: Post-Silicon Tuning for FPGAs. Submitted to ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, www.ece.utexas.edu/~bijansky/fpga.pdf
- Field Programmability of Supply Voltages for FPGA Power Reduction. F. Li, Y. Lin, and L. He. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 26(4):752764, 2007.
- The Design of a SRAM-based Field-Programmable Gate Array-Part II: Circuit Design and Layout, P. Chow, S. Seo, J. Rose, K. Chung, G. Paez-Monzon, and I. Rahardja„ IEEE Transactions on Very Large Scale Integration Systems, vol. 7, no. 3, pp. 321330, 1999.

5.6 Higher arithmetic

We'll discuss the implementation of adders and multipliers in detail in this class. However, we won't talk about how complex functions are implemented.

So if you've ever wondered how a 2$ pocket calculator computes sines, cosines, logs, etc. in the blink of an eye, this is the project for you.

It would be grossly inefficient to use Taylor series expansions for computing transcendental functions. Instead there are much better representations, and CORDIC makes use of one particular representation, which allows sines and cosines to be computed with nothing more than additions, shifts, and a single multiplication (by a constant); it very high precision with very little computational cost ($O(n)$ work for n bits).

The text book talks about computing CORDIC in Chapter 8. You can also read a short, easy to follow account of CORDIC here:

http://www.worldserver.com/turk/computergraphics/FixedPointTrigonometry.pdf

Note that the article sidesteps the issue of approximability of angles by the sum $\sum_{i=0} i=N(-1)^{\sigma_i}\cdot\tan^{-1}2^{-i})$. The approach works because $\tan^{-1}2^{-i}<2\cdot\tan^{-1}2^{-(i+1)})$, so each successive iteration yields an angle that's less than half of what it was before.

5.7 Implementation of a sub-threshold library

A standard cell library (SCL) contains the basic building blocks for designing an integrated circuit. It has a fixed set of well-characterized logic blocks. Once an integrated circuit is built using the library, the behavior of circuit will depend on information within the individual cells from the library. This information includes parasitic capacitance, area, and delay. In order to qualify as a standard cell library, it has to include NAND, NOR, inverter, and D flip-flops. SCL is commonly employed by Application Specific Integrated Circuit (ASIC) designers due to robustness and flexibility of the library, resulting in quick turnaround times.

This purpose of this project is to build a low power standard cell library using sub-threshold voltage in 45nm technology. You should have your report explaining in detail how you created the library.

Reference:

- http://www.cerc.utexas.edu/~tywu/library
- B. H. Calhoun and A. Chandrakasan, "Ultra-Dynamic Voltage Scaling Using Subthreshold Operation and Local Voltage Dithering in 90nm CMOS," in *Proc. IEEE International Solid-State Circuits Conference*,Feb. 2005.
- H. Soeleman and K. Roy, "Ultra Low Power Digital Sub-Threshold Logic", *International Symp. on Low-Power Electronics and Design*, pp. 94-96, August 1999.
- H. Soeleman and K. Roy, "Digital CMOS Logic Operation in the Sub-Threshold Region", *IEEE Great Lakes Symposium on VLSI*, pp. 107-112, March 2000.

5.8 Hardware accelerated Monte Carlo simulation

In its simplest form, an option gives the purchaser the right to buy an object (which could be a stock, or a commodity, we'll assume stock for simplicity) for a fixed price at a given time in the future. More generally, options exist wherein the purchaser can buy the commodity for a fixed price at any point up to a given time, or at the lowest price up to the given time, etc.

When the purchase time is fixed, interest rates are constant, and the object price follows Brownian motion, the Black-Scholes formula gives an analytical way to determine the fair

price of the option. This situation is rare, and analytical techniques do not exist for general option pricing.

Monte Carlo simulation can be used to get an idea of the fair price; it is computationally challenging, and the goal of this project is to use hardware acceleration for pricing. It is most natural to use a finite time step for the simulation.

One approach is to derive the exact distribution of the stock price. Given a distribution for a discrete random variable *X* (the stock price), and a distribution for a discrete random variable *Y* (its change), the distribution for *X*+*Y* is derived by convolving the two distributions—direct convolution can get expensive (quadratic in the range of the two variables), and FFT-based convolution may be a good way to proceed. You may want to consider various distributions for the increment, not just binomial, but something with a heavy tail.

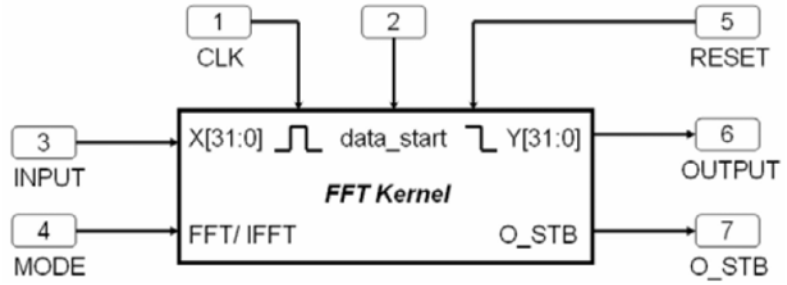
Another approach is to simulate a large number of trials, and determine a distribution based on the trial outcomes.

- en.wikipedia.org/wiki/Binomial_options_pricing_model
- Approximate Option Pricing. P. Chalasani, S. Jha, and I. Saias. Algorithmica. 1999.
- Mathematics for Finance: An Introduction to Financial Engineering. Capinski and Zastawniak.
- Randomized Algorithms. Motwani and Raghavan.

5.9 Fast Fourier Transform Kernel

For the FFT project, you are to finish a VLSI implementation of a 16-point FFT. The chip shall take the time-sampled data input at a set sampling frequency of your choice and output the correct bin counts for all the points in the FFT, within the range of error tolerance. For real-world applications, you are encouraged to aim for 64/128-point high precision FFT kernel which are compatible to wireless industry's protocols.

Example Specifications (16 bit precision for both real and img parts of inputs):



signal name	direction	description
CLK	input	system clock
RST	input	reset signal(1 effective)
data_start	input	input start (1 effective)
data_in	input [31:0]	serial input data
mode	input	FFT(0) / IFFT(1)
control counter	output [4:0]	controlling counter strobe
start_count	output	FFT_start strobe(1 effective)
data_ready	output	output ready (1 effective)
data_out	output [31:0]	serial output data

Testing: You must establish your own testing benchmark structure to test your FFT core with reasonably good coverage using all types of signals (sine waves, noise, dc) and their random combinations, below your chosen Nyquist frequency.

On algorithm level, you may choose from radix-2, radix-4, and specialized FFT implementations, etc. Final chip must be presented in layout level after synthesis, a code file alone is not sufficient.

For establishing the benchmark, you may need C/C++, TCL/TK, PERL, MATLAB, etc.

The project's deliverables will include your FFT specification definitions, test files, testing benchmark, test outputs reports (both simulated and physical), a Cadence layout of the FFT hardware, code or a schematic abstracting your layout, and a report of your algorithm (in the form of a paper or pseudo-code).

Below are several references on FFT hardware implementations:

- A radix 4 delay commutator for fast Fourier transform processor implementation Swartzlander, E.E.; Young, W.K.W.; Joseph, S.J.; Solid-State Circuits, IEEE Journal of ,Volume: 19 , Issue: 5 , Oct 1984 Pages:702 - 709
- A 64-point Fourier transform chip for high-speed wireless LAN application using OFDM Maharatna, K.; Grass, E.; Jagdhold, U.; Solid-State Circuits, IEEE Journal of ,Volume: 39 , Issue: 3 , March 2004 Pages:484 - 493
- "Design Considerations and Implementation of a DSP-Based Car-Radio IF Processor", IEEE Journal of Solid State Circuits. Jul. 2004. Pages 1110–1118.
- "A single-chip MPEG-2 codec based on customizable media embedded processor", IEEE Journal of Solid State Circuits. Mar. 2003. Pages 530–540.

- Please actively search/Google for newest ideas and build upon them!

6.0 Miscellaneous

A project of this nature will naturally build upon existing work. You are encouraged to build upon existing code/results, and in your report you may copy/adapt from others papers. However, you must explicitly make it clear that you have done so; failure to report this will be considered **plagiarism** and will be dealt with severely.