VLSI Projects

Adnan Aziz
adnan@ece.utexas.edu
Electrical and Computer Engineering
The University of Texas at Austin

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 25% 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.

1 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 by **Wednesday, October 18.**

Intermediate reports: I would like you to turn in hardcopy of a timeline and a specifications document by Tuesday, October 31, and a design document by Tuesday November 14.

Final report: 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

The report is due in hardcopy on Friday, December 8, 2005 before 5:00 p.m. in my office.

2 Report details

2.1 Project report—details

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 the 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.

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.

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.

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.

[1] I like CVS, and have written a very brief tutorial on using CVS which you can read at `www.ece.utexas.edu/~adnan/cvs_notes.txt`

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.

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.

3 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 my notes on technical writing to avoid common mistakes that engineers regularly make in writing—`www.ece.utexas.edu/~adnan/writing.html`

4 Project list

Below, I give sketches of projects that I would like to see work done on.[2] Mentors are specified in boldface; their email addresses are in Table 1.

Bear in mind that the project description is not complete, and you are responsible for making reasonable assumptions and decisions about the project.

Name	Email
Adnan Aziz	adnan_REMOVE AT ece utexas edu
Diptendu Ghosh	diptendu_REMOVE AT mail.utexas.edu
Joonsoo Kim	turo _REMOVEat mail dot utexas dot edu
Hari Mony	harimony_REMOVE AT us.ibm.com
Jiseon Park	jpark_REMOVE AT ece.utexas.edu
Tung-yeh Wu	tywu _REMOVEat mail dot utexas dot edu
Fadi Zaraket	zaraket_REMOVE AT us.ibm.com
Chaoming Zhang	cmzhang _REMOVEat mail dot utexas dot edu

Table 1: Names and email addresses for mentors

4.1 Optimized MIPS

The text describes a simple implementation of a MIPS processor in Chapter 1 and also Appendix A.10. The goal of this project is to compare a baseline unoptimized implementation of the MIPS processor given in the book to an optimized implementation that uses architectural features such as pipelining, as well as circuit optimizations such as domino logic. If you are really ambitious, you can try to add multi-issue capability.

To ensure that your optimized design functions correctly, you will need to turn in a significant verification plan, including functional coverage checks, test-bench, and assertions.

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

Hari Mony

4.2 On-chip interconnection network

In this project, you will get an in-depth understanding of the 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 good basic introduction to VLSI implementation of crossbars, especially the mechanism for programming the cross-points.
- A 250-Mb/s CMOS Crosspoint LSI for ATM Switching. Akata et al., IEEE JSSC 25(6), December 1990, pp. 1433-1439.
  - Describes an implementation of the multiple shared-memory architecture.
- 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 more advanced version of the first paper, many details on the low-level circuits needed for high-performance.
- A 200Mhz CMOS Broad-Band Switching Chip. O'Neill et al., IEEE JSSC 28(3), March 1993, pp. 269-275.
  - A reasonable introduction to an alternative way of implementing a switch fabric. The main point of interest is the use of the SORT/EXPAND nodes.

Once the architecture is matured, you may employ the skills developed in three labs to implement a prototype. In view the limited time, you may put most of the effort on the core algorithm and structure and size down the whole system. Please consider how to test and benchmark your switch since the beginning. This is a good chance to get an idea of compromising among multiple design metrics—area, speed, and power.

Mentors: Chaoming Zhang

4.3 A sleepy SRAM

One of the major techniques used to control subthreshold is sleep transistors.

In essence, sleep transistors are used for power gating: Logic runs at low Vt, and the gates are faster and leaky; Sleep transistors are high Vt. They are switched off when idle (usually NMOS alone is used) and can save 2-1000$\times$ leakage power.

Your goal is to design a 32 kbit SRAM (128 rows, 256 columns, 8 bit words) which uses sleep transistors to reduce leakage power. There are a number of ways you can go: a single huge sleep transistor, a sleep transistor per cell, or a sleep transistor per 4 cells, etc. There are power–delay tradeoffs between these, which you are to explore.

Some papers that will help:

- "Leakage control with efficient use of transistor stacks in single threshold CMOS." M. Johnson, D. Somasekhar, L.-Y. Chiou, K. Roy. IEEE Transactions on VLSI Systems, February 2002.
- A Leakage Reduction Methodology for Distributed MTCMOS. B. H. Calhoun, F. A. Honore. IEEE JSC, 2004

Professor David Pan in our department has done some work in this area, and you may want to meet with him to get more ideas. Also take a look at his papers:

- "Sleep Transistor Sizing Using Timing Criticality and Temporal Currents", Proc. Asia South Pacific Design Automation Conference (ASPDAC), to appear, Jan. 2005.
- "Generic Voltage Island: CAD Flow and Design Experience", Austin Conference on Energy Efficient Design (ACEED), Austin, Texas, Feb. 2003.
- "5-GHz 32-bit integer execution core in 130-nm dual-Vt/CMOS", IEEE Journal of Solid State Circuits, Nov. 2002. Pages 1421–1432.
- "A shared-well dual-supply-voltage 64-bit ALU", IEEE Journal of Solid State Circuits. Mar. 2004. Pages 494–500.
- "A 2 load/store pipe for a low-power 1-GHz embedded processor," IEEE Journal of Solid State Circuits. Nov. 2003. Pages 1857–1865.
- "A Flexible Design Approach for the Use of Dual Supply Voltages and Level Conversion for Low-Power ASIC Design", Austin Conference on Energy Efficient Design (ACEED), Austin, Texas, Feb, 2003.

Jiseon Park

4.4 3 GHz 32-bit MAC in 0.18μ CMOS

The goal of this project is to design a very high-performance 32-bit multiply-accumulate unit in the technology used in this class. You are to used clocked dynamic circuits to get the performance of pipelining without paying the price of pipe latches.

- Wave-Pipelining: A Tutorial and Research Survey. Burleson, *et al.*. IEEE TVLSI, September 1999.
- Clock-delayed Domino for Dynamic Circuit Design. Yee and Sechen. IEEE TVLSI, August 2000.

Mentors: Jiseon Park

4.5 FFT

For the FFT project, you must create a hardware implementation of a 16-point FFT. The chip must intake time-sampled data at a set sampling frequency of your choice and output the correct bin counts for all the points in the FFT. You must write several test case inputs for all types of signals (sine waves, noise, dc) below your chosen Nyquist frequency. The hardware implementation may be derived from any FFT algorithm you wish, including or not including windows, radix-2, radix-4, and specialized FFT implementations. The final chip must exist in a layout form with physical implementations, so a pure code file is not sufficient.

The project's deliverables will include your FFT specification definitions, test files, your test outputs (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 code).

Below are several articles on FFT hardware implementations:

- A single chip radix-2 FFT butterfly architecture using parallel data distributed arithmetic Mactaggart, I.R.; Jack, M.A.; Solid-State Circuits, IEEE Journal of ,Volume: 19 , Issue: 3 , Jun 1984 Pages:368 - 373
- 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 VLSI array processor for 16-point FFT Lee, Moon-Key; Shin, Kyung-Wook; Lee, Jang-Kyu; Solid-State Circuits, IEEE Journal of ,Volume: 26 , Issue: 9 , Sept. 1991 Pages:1286 - 1292

- 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.
- `http://www.wright.edu/~patli.2/Thesis-SudhirPatli.pdf` (A thesis on a radix-4 hardware implementation of the FFT)

Tung-Yeh Wu

4.6 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 a pure digital PLL and compare its performance (measured in lock time and phase noise) and costs (in terms of area, power, delay) to a traditional analog PLL.

PLL design is discussed in an RF design book, e.g., Tom Lee's "Design of CMOS RF circuits." To be concrete, focus on a PLL for a standard such as GSM.

Jiseon Park

4.7 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 $\sum_{i=0} i = N(-1)^{a_i} \cdot \tan^{-1} 2^{-i})$. 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.

Joonsoo Kim

4.8 Equalizers for MIMO

MIMO technology has great promise for robust, high-bandwidth wireless communication. In order to achieve this MIMO includes a great deal of fairly complex digital signal processing. The goal of this project is to examine the cost of implementing MIMO. In particular, channel equalization for MIMO is computationally challenging, and I would like you to study the various equalization algorithms, and estimate their VLSI implementation cost.

References: Joseph Cavallaro at Rice University has a number of articles on equalization for MIMO.

Diptendu Ghosh

4.9 Applying hardware verification technologies to software

Hardware bugs are harder to find than those in software; they also have much higher costs, and are harder to rectify than bugs in software. For these reasons, hardware verification technology is more evolved than that for software.

The goal of this project is to see how to apply hardware verification techniques to software verification. The idea would be to model software using sequential circuits. This cannot be done for general programs, as they are infinite state—however, many bugs in software can be found via "finitization".

To be concrete, I would like you to take the 10 C programs from the following book:

- "Find the bug : a book of incorrect programs," by Adam Barr, available in PCL (Call number QA 76.9 D43 B37)

and model them as well as their correctness conditions in VHDL. (The mentor, Fadi Zaraket, works at IBM and has access to VHDL verification tools that he can use to then perform the checks.)

Fadi Zaraket

5 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.