Digital Systems Design Project

EEC 181A/B

Lecture 1

Bevan M. Baas

Tuesday, January 7, 2025

Today

- Course details
  - Policies
  - Grading
  - Schedule (web page)
  - Course objective and strategies
- My background & some recent work in many-core processor arrays
- Digital Systems Overview
- Basic Units
- 7 Basic Diagrams
- HDL to Hardware

A Few Announcements

- No lab today
- Start meeting Thursday in Kemper 2107
  - Tue/Th 5:10 – 6:00pm
- FPGA board pick up Thursday in lab
- One board per student
- Probably cameras will need to be shared
- You may need to purchase a microSD card

Project Teams

- The second-quarter EEC 181B project will be done in a small group
  - Normal size: 3 students
  - A group of 2 or 4 students may be possible if necessary (needs to be verified)
- Start talking to classmates you would like to work with
- Be honest to your potential teammates about the level of effort you want to put into the class
- Wait until the enrollment stabilizes to finalize your group—near or after the drop deadline

Teaching Assistant

- Derek Li

Course Workload

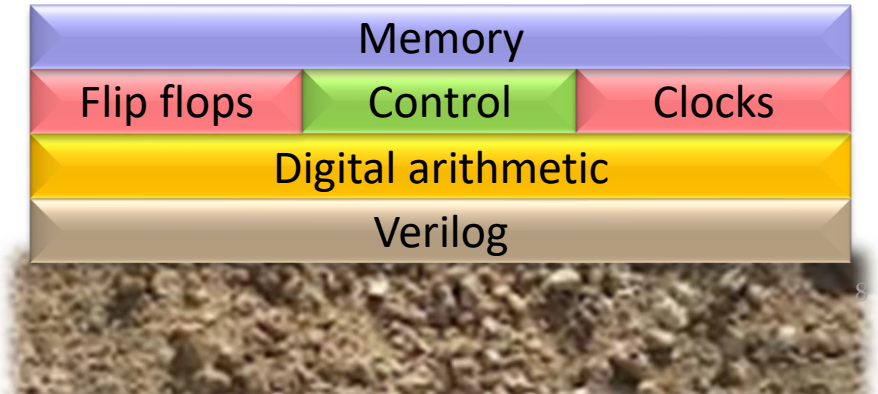
- 6 units (3+3)
- Upper division design capstone
- Complete complex real-time digital system
  - datapath
  - control
  - memory
  - peripheral components
  - hardware pipelining
  - digital arithmetic
  - timing/clocking
- Passing this course requires significant time and effort

Course Overview

- First quarter (181A)
  - Individual projects to build domain knowledge
- Second quarter (181B)
  - Creative open-ended design
  - Implement
  - Test and verify
  - Measure
  - Document
  - Present
  - Write (brief report)

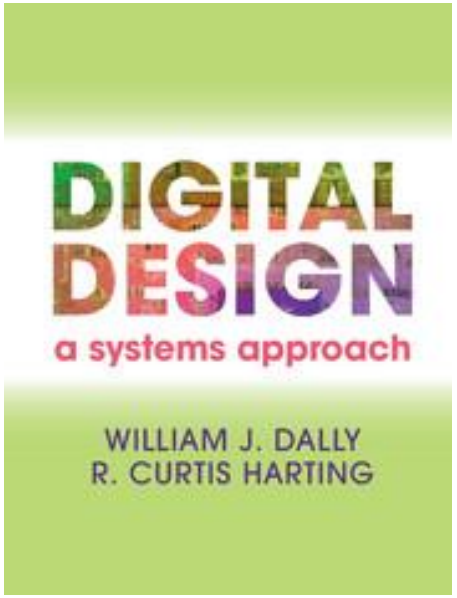
Course Plan: Tools and Methodologies for Large-Scale Digital Designs

- Introduction
- The Verilog hardware description language (HDL)
- Digital arithmetic
  - Number formats
  - Addition / Subtraction
  - Later in quarter: Multiplication, saturation, and rounding
- Flip-Flops
- Control circuits: counters and general FSMs
- Clocks
- Interface circuits
- Memories



Textbook

- None required
- Recommended reference:
  Digital Design, A Systems Approach
  William Dally & R. Curtis Harting
  Cambridge University Press
  2012 edition (not Draft or VHDL version)

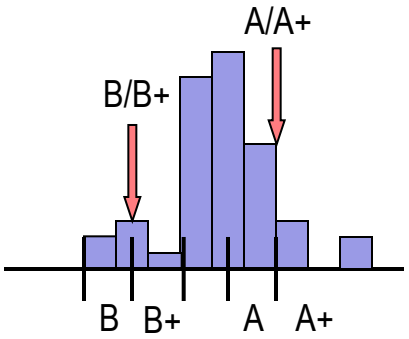


Grading Philosophy

- Grading serves two main purposes:
  1. Motivate you to do the work required to learn
     - Lectures
     - Solving problems in labs
     - Discussions with others
  2. Give others an indication of how well you know the material

Letter Grade Assignments

- I assign a letter grade for only the final course grade
- One grade for both 181A and 181B
- "IP" (In Progress) given at the end of 181A
- You can see score statistics for each graded item on Canvas
- I look at the final exams and course record of the class and assign two key dividing points: the A/A+ and D+/C- boundaries, and assign course grades from there using equally-sized intervals
  - No required numbers of any particular letter grades
  - Absolute scores are not important; the boundaries shift according to the difficulty of the exams in any quarter
  - Ignore any letter grades you might see on canvas



(not actual grade data)

Lectures

- Ask questions at any time
  - Please raise your hand
- Be respectful of others
  - Hold conversations outside of class
  - Silence phones
  - Sit near the door if you come in late or need to leave early

Course Announcements

- In class
- Web
  - Assignments, etc.
- Email via Canvas Announcements
  - Time-critical announcements only

Questions

- In class
- In lab
- Office hours
  - Me: after lecture
  - TA will also have an office hour
- TA
  - Lab
  - Office hours
  - Email
- Very likely: slack channel
- Please avoid email

Working With Others

- Collaboration
  - Asking questions and explaining principles produces better work and dramatically increases learning
  - Working with others
    - Do homework and prelabs with classmates nearby
    - Ask each other questions, help each other—regarding **principles,** and **approaches to solving** only
  - See *Course Collaboration Policy* on web page
- Dishonesty
  - Copying produces similar work, stunts learning, is not fair to honest students, and is not allowed in this course
    - Students engaged in dishonest work will be referred to Student Judicial Affairs
    - I will try to keep in-class exams honest
    - Steps will be taken to keep out of class work honest

Penalties for Violating the *Policy on Student Conduct and Discipline*

- Penalties
  - Minimum penalty: meetings with SJA officer, zero grade on work, record with SJA
  - Permanent **F** grade on your transcript, no credit for the class
  - One to three quarter suspension from the university
  - Permanent dismissal from all ten campuses of the University of California. Permanent notation on your transcript.

Penalties for Violating the *Policy on Student Conduct and Discipline*

- Several perspectives
  - Personal — obvious reasons
  - ECE and UCD — (especially for those inclined to share work with someone doing poorly in class) Cheating harms our major and university's reputation among employers who interview our graduates.
- In summary: The purpose of the penalties and me mentioning them is so that no one will get one!!! Don't do anything that violates the Policy on Student Conduct!

Penalties for Violating the *Policy on Student Conduct and Discipline*

- Typical scenario:
  - Someone shares code/design with another
  - They get caught
  - The "Copier" feels terrible guilt for causing a friend to get a zero
  - The "Sharer" deeply regrets sharing resulting in a zero when he/she should have had a full score

MOSS

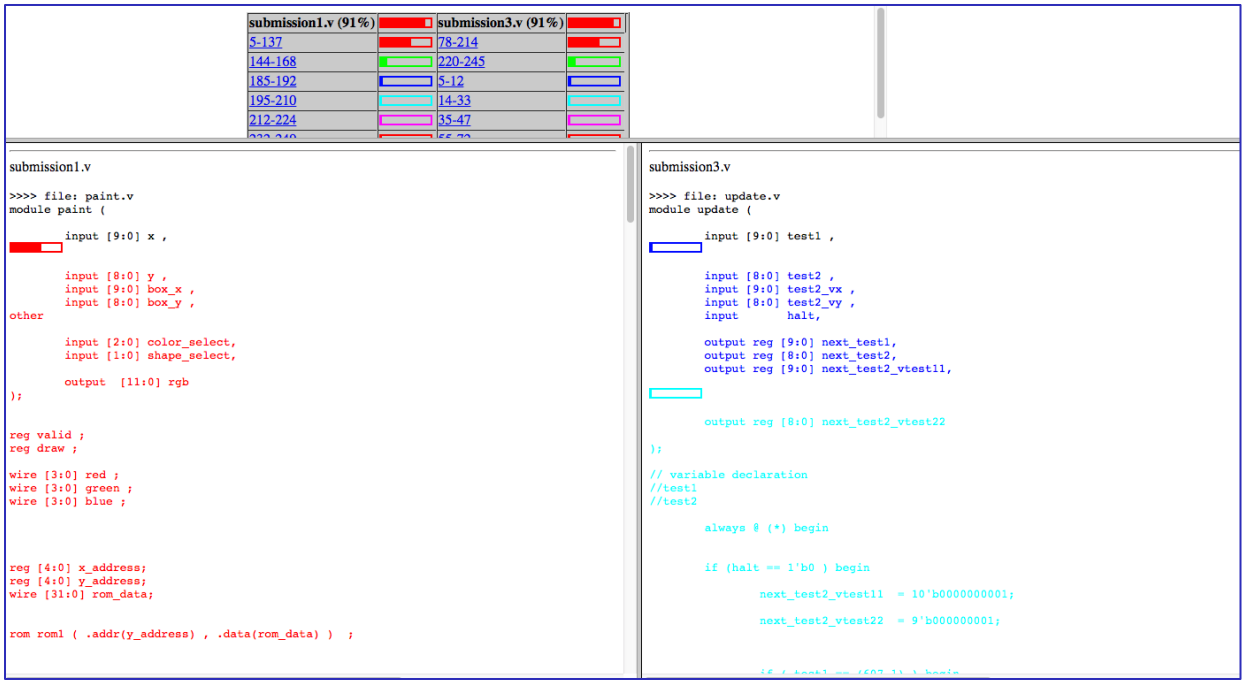
- "Measure Of Software Similarity" tool
- Utilizes very sophisticated and fast algorithms
- Processes all *Order*(*N*^2/2) = *N*(*N*–1)/2 pairings
  - Ex: examination of 300 submissions includes 44,850 pairwise comparisons
- Runs are done for each lab of this year's work combined with work from past years

MOSS Demonstration Case

Modified version

Original version

- Added, deleted, changed all comments
- Changed all variable names
- Reordered modules
- Reordered lines of code within modules
- Changed equivalent logic
- 91% similarity for submission 1
- 91% similarity for submission 2



submission1.v (91%)		submission3.v (91%)	
5-137		78-214	
144-168		220-245	
185-192		5-12	
195-210		14-33	
212-224		35-47	

submission1.v

```
>>>> file: paint.v
module paint (

        input [9:0] x ,

        input [8:0] y ,
        input [9:0] box_x ,
        input [8:0] box_y ,
other

        input [2:0] color_select,
        input [1:0] shape_select,

        output  [11:0] rgb
);


reg valid ;
reg draw ;

wire [3:0] red ;
wire [3:0] green ;
wire [3:0] blue ;



reg [4:0] x_address;
reg [4:0] y_address;
wire [31:0] rom_data;


rom rom1 ( .addr(y_address) , .data(rom_data) )  ;
```

submission3.v

```
>>>> file: update.v
module update (

        input [9:0] test1 ,

        input [8:0] test2 ,
        input [9:0] test2_vx ,
        input [8:0] test2_vy ,
        input       halt,

        output reg [9:0] next_test1,
        output reg [8:0] next_test2,
        output reg [9:0] next_test2_vtest11,

        output reg [8:0] next_test2_vtest22

);

// variable declaration
//test1
//test2

        always @ (*) begin

        if (halt == 1'b0 ) begin

                next_test2_vtest11  = 10'b0000000001;

                next_test2_vtest22  = 9'b000000001;
```

MOSS

- Key take-away messages:
  1) MOSS is amazingly good at spotting pairs of submissions that share a common design
     - This meshes very well with the course collaboration policy
  2) Follow the course collaboration policy and you have nothing to worry about
  3) Violate the course collaboration policy and you *will* have something to worry about

Demo Your Own Design

- A hash code of your verilog will be recorded during your demo
- Do not modify your code at all before uploading to canvas—your hash codes must match exactly to receive credit
- Yes, unfortunately a few students tried demoing someone else's code in past quarters

Exam and Quiz Regrades

- Some number of exams and quizzes will be scanned before being returned
- Key take-away messages:
  - Do not change anything on your work if you request a regrade
  - Past students have tried this and got in big BIG trouble!!!

Cheating Websites
chegg, coursehero, slader, etc.

- The university has recently taken a very strong stand against paying for work (2-quarter suspension for first offense recently)
- Key take-away messages:
  - Do not post any course material
  - Of course do not use any unpermitted outside material in work you submit
  - Of course do not post solutions
  - Multiple students have and got caught!!!

Hwk/Proj Deadline Extensions

- Extending submission deadlines is problematic
  - Quarters are short and we have a lot of material to cover
  - When I have granted extensions, a student will frequently tell me, "I wish the extension had been announced a long time ago—I sacrificed one of my other classes to meet the deadline and this is not fair to me". I am sympathetic to that reasoning.
- So an extension will be given in only a very unusual circumstance

Advice From Last Year's 180 Students

- *What advice do you have for future 180 students?*
- 29% — Designing process
  - Spend more time designing a circuit before trying to implement it in Verilog.
  - Make sure & diagram everything above a design before writing any verilog code.
  - Design intensively before coding (draw diagrams), use a testbench because it's much more efficient than testing on the board. Review EEC180A concepts!
  - DRAW diagrams. Ask questions about syntax (be paranoid), actually write testbenches while writing the code. Think like circuits, not programs.
  - The big thing with Verilog/HDL coding is you NEED to think a lot about the problem before starting code. Draw the circuit, pipeline diagram, and timing diagram before writing your code.
  - Work on testbench while doing the design. It helps.
  - ModelSim is your best friend for debugging.
  - Start thinking about the design of lab early so you can break the whole lab into parts. Schedule time to finish each part accordingly.

Advice From Last Year's 180 Students

- *What advice do you have for future 180 students?*
- 29% — Lab-related
  - Start labs early and divide modules into submodules for easy testbenching.
  - Spend a lot of time on labs: learn A LOT from them.
  - Start labs early.
  - Organize time for the labs.
  - Work hard on the labs.
  - Give yourself more time for labs, don't look at prelab right before lab.
  - Start labs early before the lab section because a lot of questions come up during the process. It's best to have questions in the beginning than in the end when lab section ends and your ways of getting answers are harder now.
  - Start the labs as early as possible!!!

Advice From Last Year's 180 Students

- *What advice do you have for future 180 students?*
- 14% — Book related
  - BUY THE BOOK AND READ IT! Also, make a design first then start coding. Make submodules and write testbenches for each (as you go).
  - Read the book!!
  - Read the book and get your lab work done ASAP!
  - Read the book, start labs early
- 28% — Misc
  - Learn the differences between a reg and a wire and where/how to use them right away.
  - Understand timing very well.
  - Take notes in class and read lecture notes after. Start labs quickly so more questions can be asked during lab time.
  - etc.

Advice From Past TAs: Finishing Labs

- "some students leave their lab work until the last minute because they tend to think that the code will work without any issues in the span of four hours."
- "some students do not test their designs using ModelSim waveforms; instead, they either ask me what could possibly be wrong with their code or they will attempt the trial and error method by changing the code in small increments and hope that it all works out."
- "many students are spending very little time on paper design"
- "When they ask me for help, I notice that their bugs are usually caused by bad planning, not having a plan or not understanding the lab in the first place."

Advice From Past TAs: Finishing Labs

- "...students don't test their designs in modelsim using test benches, rather they try to verify their designs on the board itself. I have repeatedly stressed that it would be better if they did the former first."
- "...students in my section probably left a lot of implementation for the last moment. Some of the students in the other sections who were present in my lab today had very little work done as well."

Advice From Past TAs: Lab Checkoffs

- *"There is too little time in lab for questions because checkoffs take too long"*
- "I've made it clear that I expect check off sheets and lab reports to be printed before check offs begin, but in every lab I have many students who do not do this before lab."
- "Students also spend a lot of time setting up their simulations so I've asked students to have their simulations already open and ran but still the majority of students will not."
- "I stayed an extra hour yesterday after my lab section but the majority of my students leave right after checkoff. I mostly end up helping students from other sections after checkoff."

Handouts Mostly from EEC 180

- The material in the Handouts posted on the course web page includes some of the most foundational material
- There is a **pretty high expectation** that you understand this material and are proficient in applying it
- When the handouts or lectures and the textbook disagree on something, follow the handouts and lectures

Course Web Page

- Everything except grades will be posted on the main course web page
- http://www.ece.ucdavis.edu/~bbaas/181/
- There is a link on the canvas course web page