Today

- Course details
  - Policies
  - Grading
  - Schedule (web page)
  - Course objective and strategies
- My background
Teaching Assistants

• Arthur Hlaing
• Brent Bohnenstiehl
• Filipe Borges
• Molly Smith
My Teaching Philosophy

• Primary goal (mine and yours):

  Learn digital system design well

• Achieve this through:
  – Reading textbook
    • Chapter material, Example design problems
    • Book is comprehensive and quite broad
  – Lectures
  – Solving problems on paper (homework)
  – Solving problems and building things in lab
  – Discussions with other students, TAs, myself
Textbook

- Digital Design, A Systems Approach
  William Dally & R. Curtis Harting
  Cambridge University Press
  2012 edition

- Begin reading:
  - Chapter 1
    - *The Digital Abstraction*
    - Prerequisite material
  - Chapter 2
    - *The Practice of Digital System Design*
    - High-level background
  - Chapter 3
    - *Boolean Algebra*
    - Prerequisite material
Grading Philosophy

- Grading serves two main purposes:
  1. Motivate you to do the work required to learn
     - Reading textbook (quizzes)
     - Lectures (quizzes)
     - Solving problems in homework (exams)
     - Solving problems in labs (lab grading, exams)
     - 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
• 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

© B. Baas

(not actual grade data)
Lab Grades

• Each lab section is graded by the TA assigned to that section
• For many reasons, the same lab work will be given slightly different grades for students in different sections
  – We will do whatever is possible to minimize this such as having detailed rubrics for problems
• To eliminate much of this bias, at the end of the quarter, the average for each lab section will be calculated and the averages for the N–1 lower sections will be increased so all sections have the same average
• This issue is not present for all other graded work
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
  – Time-critical announcements only
Questions

- In class
- In lab
- Office hours
  - Tentatively Wednesdays 3-4pm. Please talk to me or email me today if that is not convenient.
- After both lectures
- See me in person rather than through email
- TAs
  - Lab
  - Office hours
  - 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
  – One to three quarter suspension from the university
  – Permanent dismissal from all ten campuses of the University of California. Permanent notation on your transcript.

• The purpose of the penalties and me mentioning them is so that no one will get a penalty!!! 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

- 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
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 demo-ing 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
  - One student did last quarter and got in big BIG trouble!!!
Cheating Websites
chegg, coursehero, etc.

• Key take-away messages:
  – Do not post assignments
  – Of course do not use any unpermitted outside material in work you submit
  – Of course do not post solutions
  – Two students did last year and got caught!!!
Course Workload

• **5 unit course**

• **New ways of thinking of things requires effort**
  – Algebra: use variables
  – Calculus: no concrete solutions for indefinite integrals
  – Digital Design (EEC 180)
    • HDL—a new way of writing “code”
    • Autonomous hardware: datapaths, memory, control
    • Sophisticated controllers

• **Passing this course requires significant time and effort**
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.
Laboratory

• Each lab period contains three main sub-periods:
  1) Give your Prelab to your TA as soon as you arrive and they will be checked-off and returned in the first ~10 minutes of the period
  2) Next, any final checkoffs for the previous week's lab can be done and then the Lab Reports for the previous week's lab are due. Lab Reports submitted after this deadline are considered late
  3) Lastly, work on the lab to be done that week
Laboratory: DE10-Lite FPGA Board

- Altera/Intel FPGA
- 50K programmable logic elements
- 1.6 Mbits M9K memory
  8192 bits/block (9216 incl. parity)
- 144 18x18 multipliers
- 4 PLLs
- On-Board
  - 10 LEDs
  - 10 slide switches
  - 2 push buttons
  - 6 7-segment displays
  - USB
  - 64 MB SDRAM
  - Accelerometers
  - low-resolution VGA
Laboratory: DE10-Lite FPGA Board
• The main body of material is presented in the book, lectures, and handouts

• Generally speaking, the labs complement the main material
  – They go into a much greater depth on specific topics
  – They give design experience
  – They give significant practical application of theory

• The Quizzes and Final Exam generally focus on the main body of material
Book + Lectures + Handouts + Laboratory

- Breadth and Depth
Handouts

- The handouts are copies of most of my lecture notes
- 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
- There is a link on the canvas course web page
AsAP2
167-Processor Chip

- 65 nm CMOS, 1.2 GHz (fastest processor designed in any university)
- 3 accelerators + 3 shared memories
- New on-chip networks
- Processors choose own supply voltage and clock freq.
- Apps: JPEG, Wi-Fi TX & RX, H.264 video encoder, ultrasound
- Tools: compiler, mapping, simulators
- Undergrad research opportunities
Advancing CMOS Technologies

- Moore’s “Law” (Observation) was made in 1965 and notes that transistor density ~doubles every year (every 1.5 years now)
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp
New data added by B. Baas
Number of Processors on a Single Die vs. Year

Note: Each processor capable of independent program execution

© B. Baas
Processor Eras

- **Transistor Era:** The Intel 4004 was the first commercial single-chip microprocessor and it contained 2300 hand-drawn transistors.

- **Single/Multi-Processor Era:** Focus on components of single processors and multi-processors, which generally scale well to only small numbers of processors.

- **1000-Processor Era:** Focus on making systems scalable and working with processors as building blocks. The 32 nm 1000-processor KiloCore chip would contain approximately 2300-3700 processors if its area were the same as a 32 nm Intel Core i7 processor, or 11,000 processors if its area were the same as an Nvidia GP100.
Some Benefits of Fine-Grain Many-Core

- Die drawn approximately to scale

<table>
<thead>
<tr>
<th>Single Processor</th>
<th>Cores per area of ARM A9</th>
<th>22 mW/GHz per 0.055 mm^2 area</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM Cortex-A9</td>
<td>1</td>
<td>1643 mW/GHz</td>
</tr>
<tr>
<td>Intel Atom Clover Trail</td>
<td>1.5</td>
<td>1120 mW/GHz</td>
</tr>
<tr>
<td>ARM Cortex-A15</td>
<td>7.8</td>
<td>212 mW/GHz</td>
</tr>
<tr>
<td>MIT RAW</td>
<td>8.3</td>
<td>198 mW/GHz</td>
</tr>
<tr>
<td>UC Davis KiloCore</td>
<td>74.7</td>
<td>22 mW/GHz</td>
</tr>
</tbody>
</table>
DIGITAL SYSTEMS
Digital Systems

- Datapath
- Memory
- Control
- Misc
Key Design Metrics
(Means to Compare Multiple Designs)

1) Performance
   a) Throughput (high); e.g., 250 MSamples/sec
   b) Latency (low); e.g., 2.7 µsec from first sample in -> first out
   c) Numerical precision

2) Chip area (cost); e.g., mm$^2$ die area, area of standard cell netlist

3) Energy dissipation per workload, e.g., Joules per JPEG image

4) Design complexity
   – Design time = lower performance
   – Software more important as systems become more complex

5) Suitability for future fabrication technologies
   – Many transistors
   – Faulty devices
     i) During manufacturing process
     ii) Device wear out due to effects such as NBTI