UART Adapter (*Mini-Project*)

UCB EECS150 Spring 2010

Lab Lecture #5

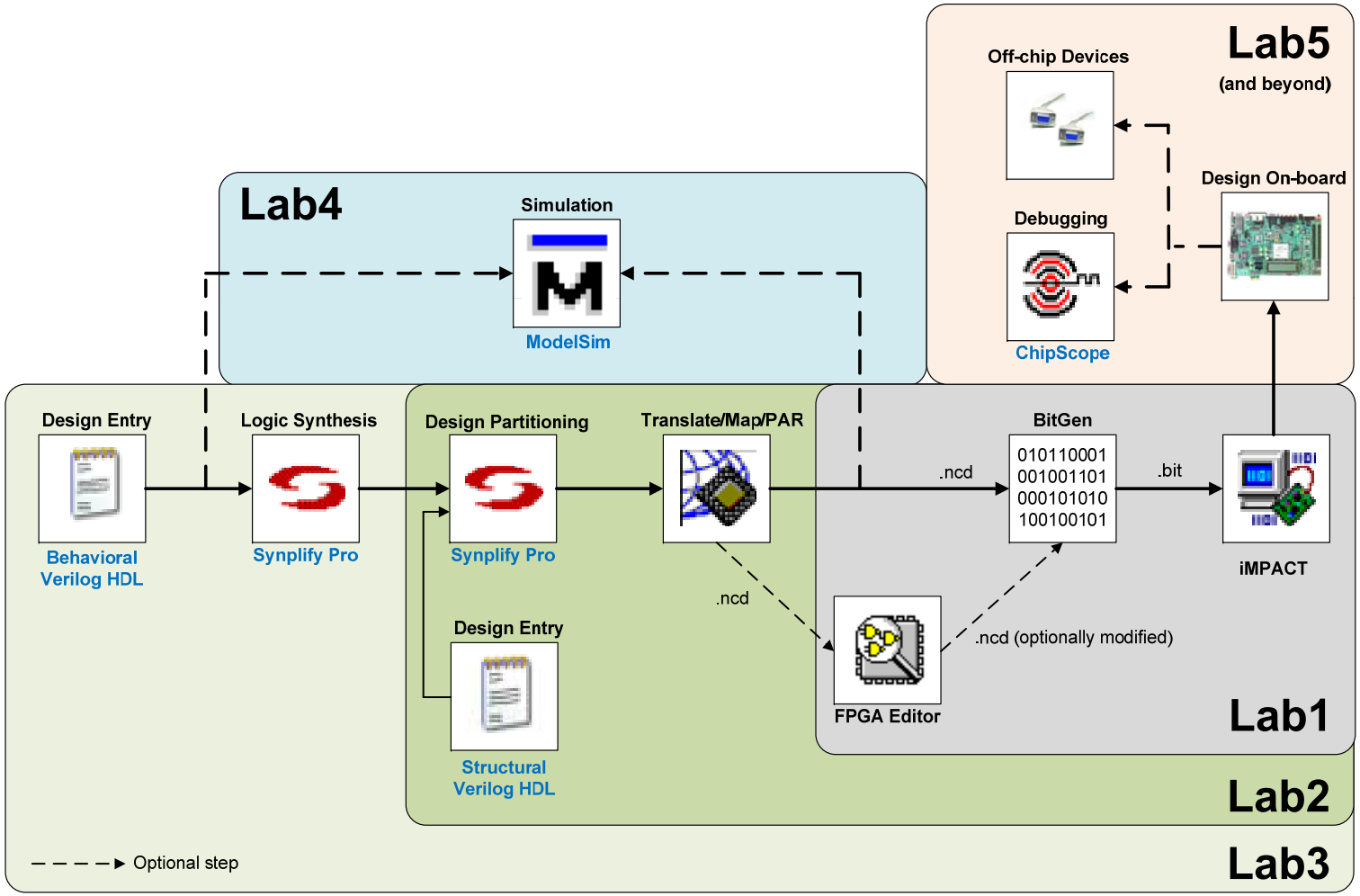
Agenda

- The **entire** CS150 CAD flow
- A new debugging tool (ChipScope)
- Lab 5 is to be done in pairs

Questions?

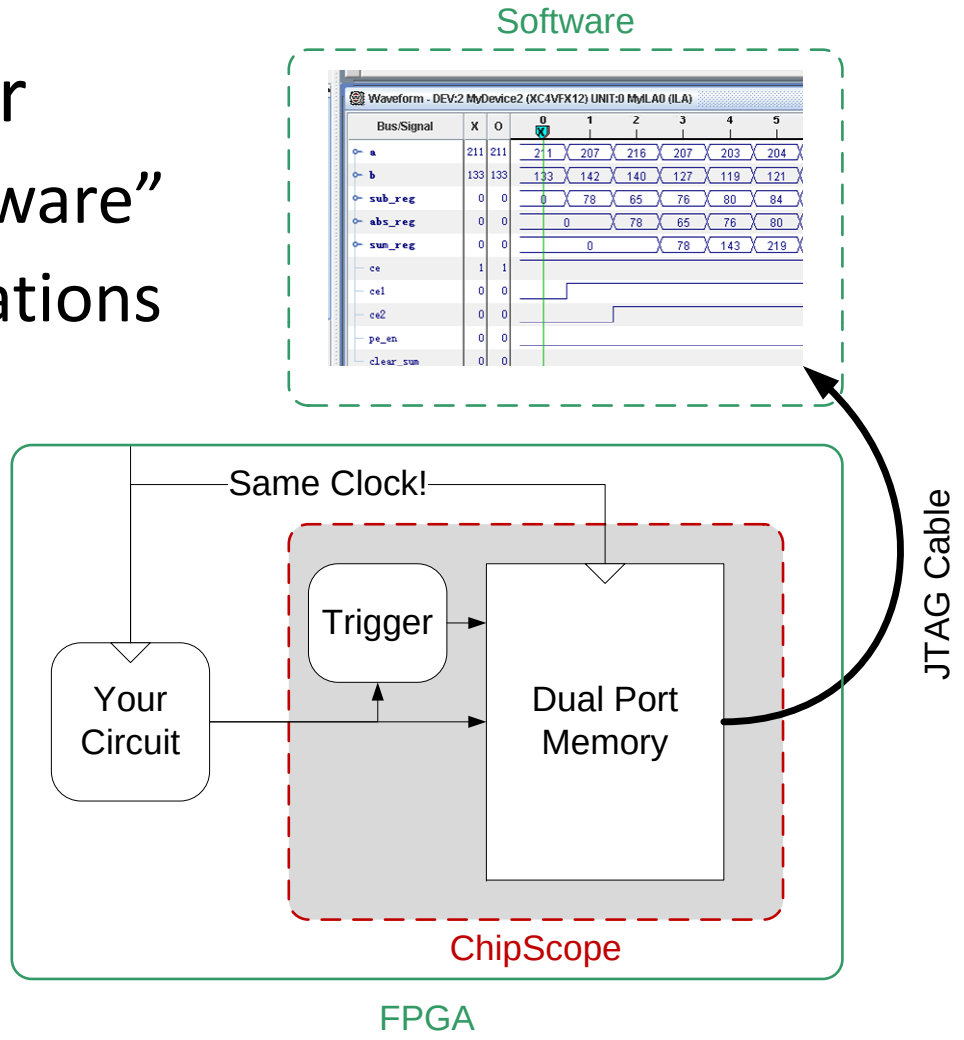
- A very awkward picture of Chris asleep in a car
- Lab 5 overview
- Design reviews

Where we are and why



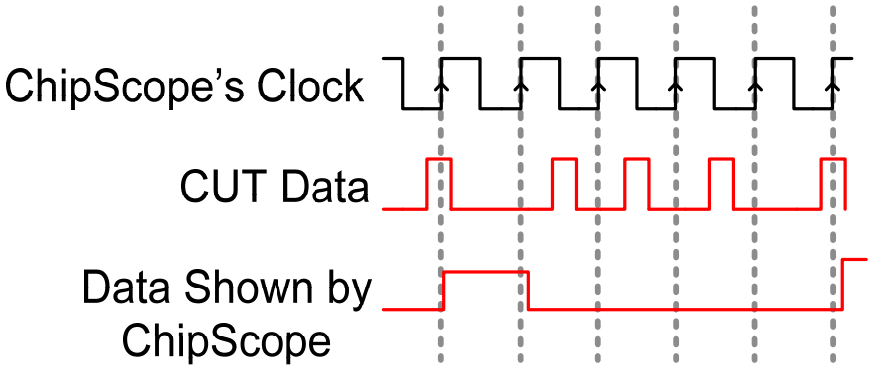
New tool : ChipScope (1)

- In-System Debugger
  - "ModelSim in hardware"
  - But has many limitations
- **Samples** signals on clock edge
- Shows only a few cycles
- Trigger-based



ChipScope (2)

- Is not “Magic”
  - Uses block memories on the FPGA to save the value of a signal. Saves several cycles after triggered (a pre-determined input pattern occurs)
  - Software reads and displays the saved trace
- Know its limitations!
  - Expensive
  - Can affect timing
  - Gives limited visibility



ChipScope (3)

Compared to ModelSim:

- ModelSim
  - High visibility (shows any, or every signal in the design).
  - Quick turnaround for debugging
  - **Only a simulation** (not guaranteed to work in hardware)
  - Will not show all bugs
- ChipScope
  - Shows values observed in hardware (the real deal)
  - **Samples** the data using a clock
  - Requires a complete tool cycle for debugging
  - Low visibility (shows only a small number of signals)

USE BOTH!

New Policy (Lab 5 and Project)

- A design document must be shown **1 week before check-off** in your lab section.
  - Both partners must be present.
  - Be prepared to defend your design
  - This is a part of your grade
  - Stay tuned (detail in a few slides)

- Pick a partner for Lab 5!

Lab 5 is a Partner Lab!

- Find someone to work with!
- **Must** pick a partner by Friday
  - Newsgroup can be used for match-making
  - Can pick a different partner for the project

Questions?

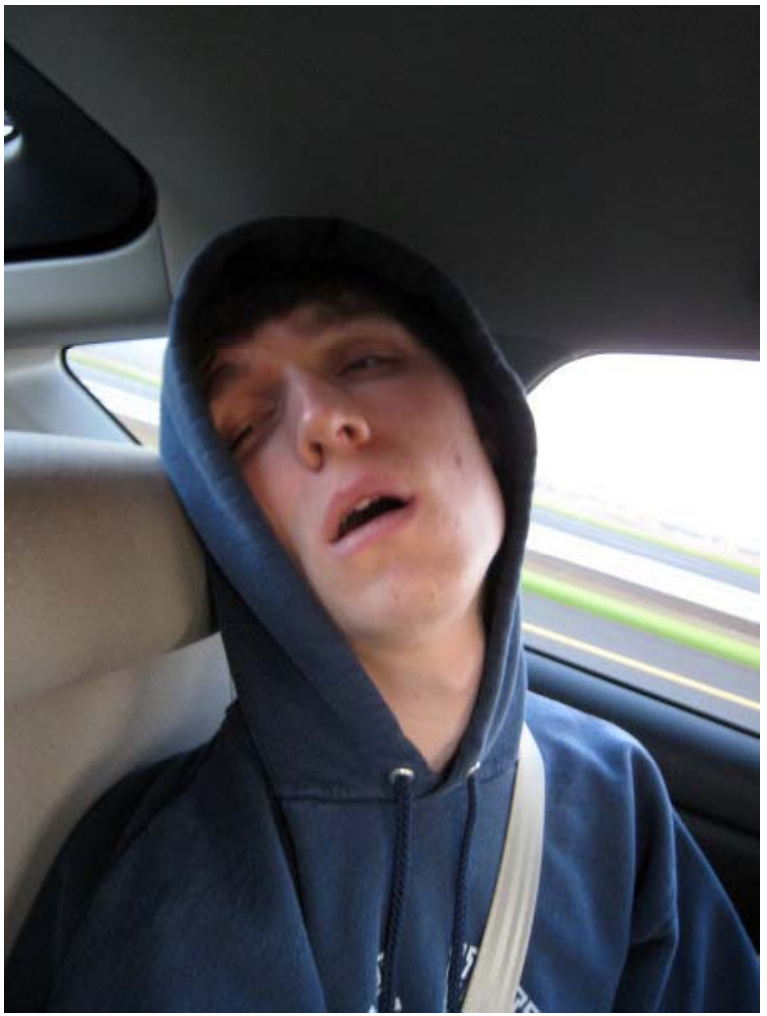
- CS150 CAD Toolflow
- ChipScope, ModelSim
- Partnerships for Lab 5, Project

- Anything else?

Please Don’t Sleep in the Lab

- It’s uncomfortable
- It’s embarrassing
- 4 chairs ≠ bed
- It’s just plain bad for you

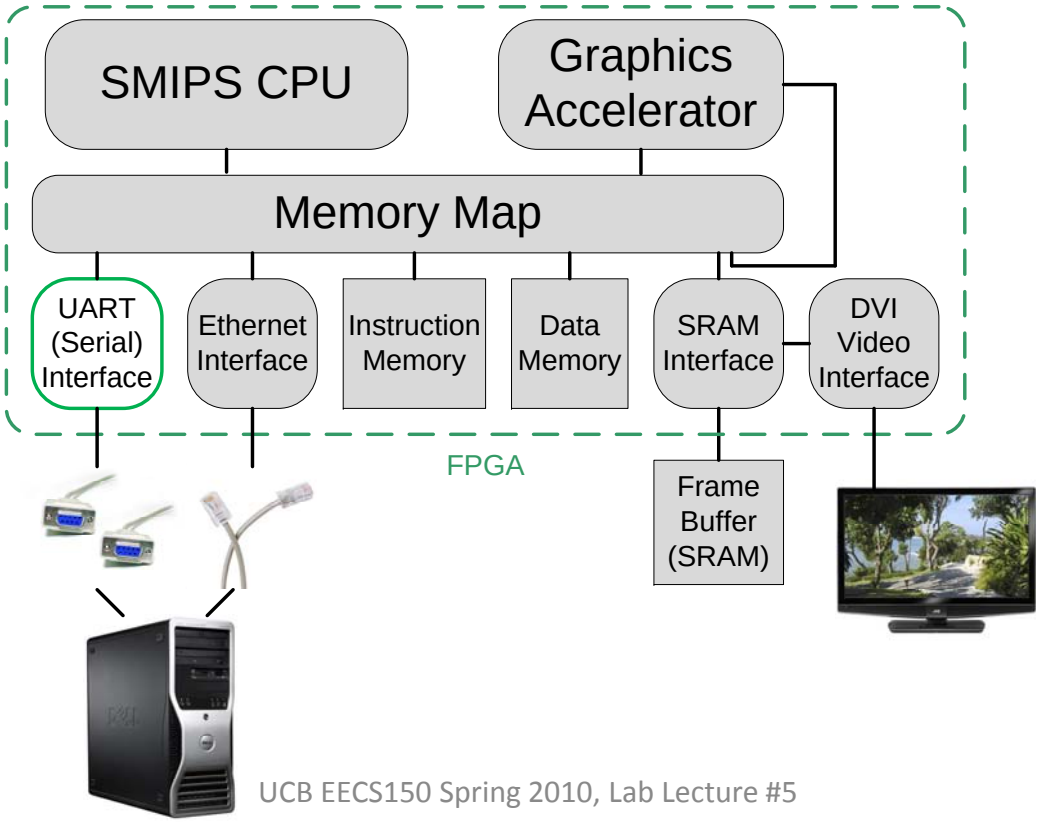
Go home to sleep!



Lab 5 (Mini-Project) (1)

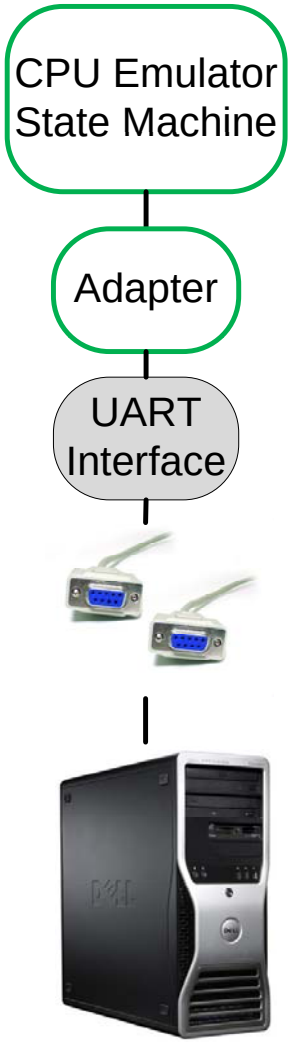
Small part of **the** project

UART interface and a little something to test it

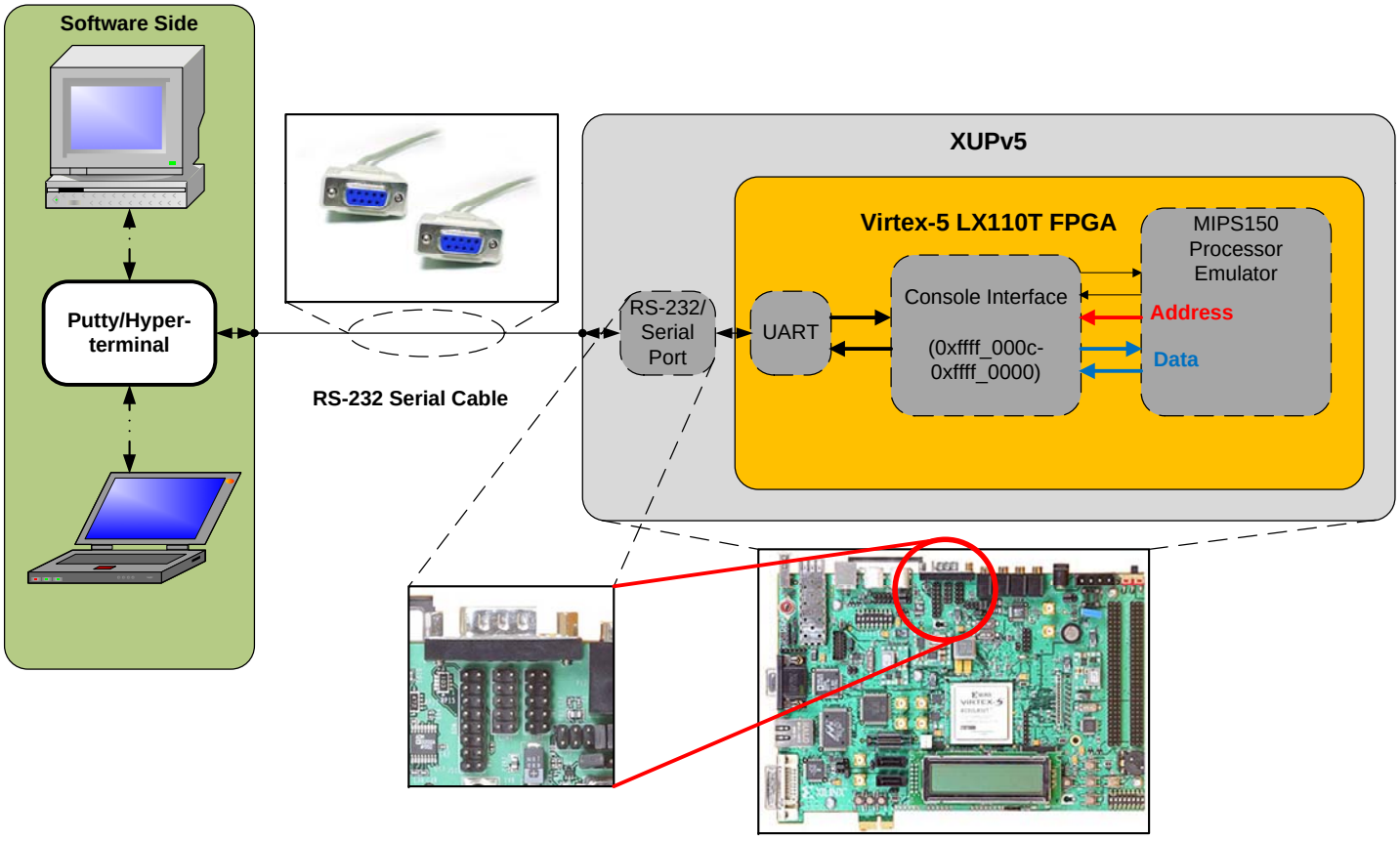


Lab 5 (Mini-Project) (2)

- Use the entire toolflow
  - Labs 1-4 taught you how
- Learn to use a handshake
- Build a state machine to mimic a CPU
  - To test the UART Interface
- We give you a UART module
  - Understand it intimately
- Practice creating design documents

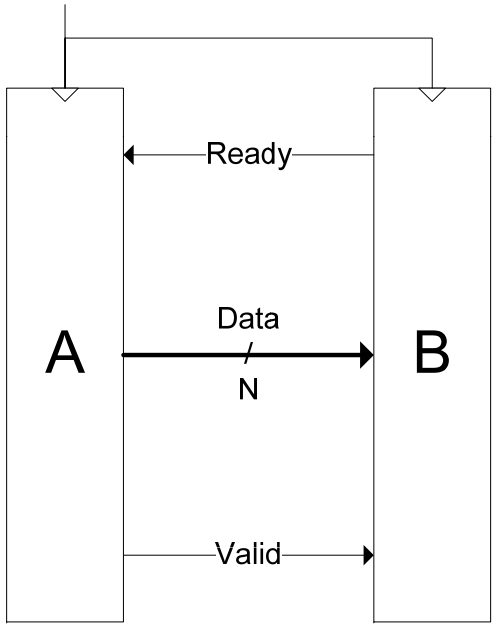


Lab 5 (Mini-Project) (3)



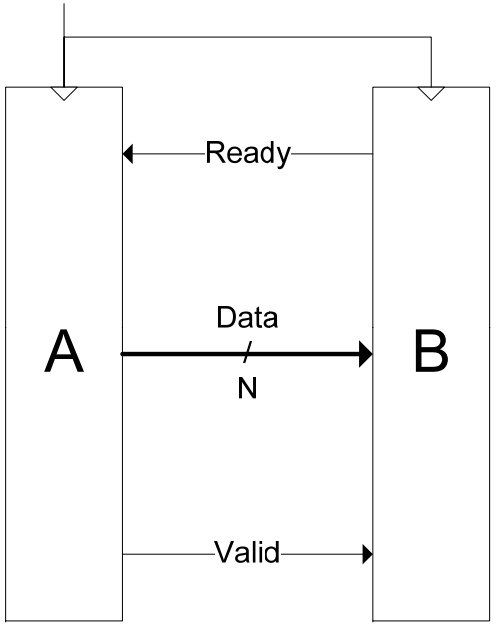
Ready-Valid Handshake (1)

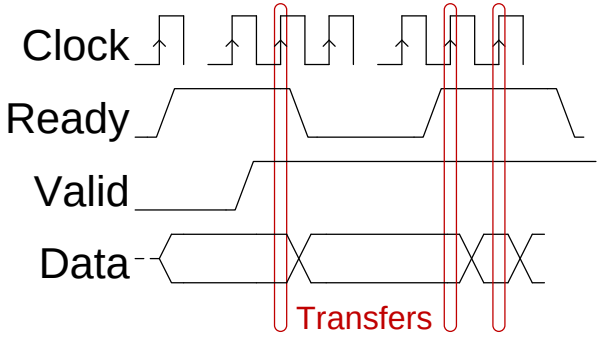
- Synchronous flow control
  - Synchronizes the flow of data (within one clock domain)
- **Creates a stream abstraction**
- Other handshakes exist
  - But we like this one



Ready-Valid Handshake (2)

- A *transfer* from A to B occurs when:
  - A positive edge of the clock arrives
  - and B is asserting Ready
  - and A is asserting Valid
- **No sequence requirements**
- Upon a transfer:
  - B may look at the Data (save, etc.)
  - A must either:
    - de-assert valid
    - Expose the next Datum





Procedure

1) Read the specification
2) DO NOT WRITE ANY VERILOG YET!
3) Draw a ***very*** high-level block diagram (be neat and name everything)
4) Expand blocks into new diagrams until you understand all details.
5) Find design flaws and repeat steps 1-4.
6) Think of ways to verify (test) the design.
7) Show your design to the TAs. Be prepared to defend it.

8) Now implement and verify the design

Design Documents (1)

- Spend 2+ hours on this
  - Detailed enough for someone else to implement
  - Show structure and function (no screenshots)
  - Use hierarchy and omit detail (no mess o' wires)
  - Xfig, OmniGraffle, Visio, etc. (*no MSPaint please*)
  - Document all optimizations and hacks thoroughly
- A good design document will make implementation and debugging easy
  - Else you **will** pull an all-nighter.

Design Documents (2)

- Graded out of 3 points:

Clear	Understandable	What?
2	1	0
Solid idea	An idea	No idea

Past CS150 designs deserve a 1.
Many commercial datasheets deserve a 0.

Acknowledgements & Contributors

Slides developed by Ilia Lebedev & John Wawrzynek (2/2010).

This work is based in part on slides by:

Ilia Lebedev, Chris Fletcher (2008-2009)

Greg Gibeling (2003-2005)

This work has been used by the following courses:

– UC Berkeley CS150 (Spring 2010): Components and Design Techniques for Digital Systems