CSE141L

Week 1: "The Saga Begins"
Apologies to Weird Al Yankovic

The Saga Begins

- *www.youtube.com/watch?v=hEcjgJSqSRU*

Introductions

Introductions

- **Prof**
  - John Eldon
- **TAs** – 4 are experienced w/ CSE141L
  - Chao (Jack) Li Shashank Uppoor
  - Tu (Tobey) Thai Kareem Kamel
  - Sachin Bharadwaj Sundramurthy
- **Tutors** – former CSE141L A students
  - Darren Eck Moiz Qureshi

Office Hours

- Google calendar link on Piazza or TED
- TA and tutor office hours in or outside CSE B260/270 labs
  - Wear your hard hat ☺
- Prof office hours in CSE2122
  - **M,W 10-11**
  - **Also before class:**
    - **M starting @1300**
    - **W starting @1400**
  - **Plus by appointment M, W, F**

Extracurriculars

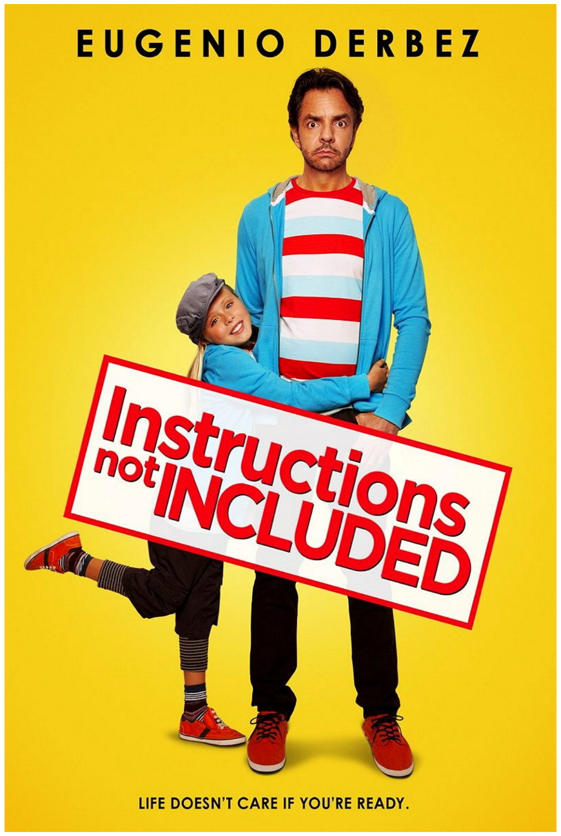
- First Lego League (FLL)
  - Saturday 20 May
  - Legoland Carlsbad
  - Project and robot design judges
- My boss’s TEDx talk about FLL
  - https://www.youtube.com/watch?v=MFS9-6xew6Q

Relevant websites

- TED
  - lots of links, incl. assignments, under Content
- Piazza
  - for class discussions, Q&A
- Altera.com model.com
- OpenCores.org
  - pipelined 5-stage MIPS
  - Google it!

Class Outline

- Lab 1 – design an ISA
  - Special purpose
  - Due 3rd wk of class
- Lab 2 – verify modules
  - ALU
  - Data memory
  - Instruction fetch
  - Due 5th week of class



Course Outline, C'td.

- Lab 3 – Write an Assembler
  - Assembly code => Machine code
  - Somewhat human-readable => Binary
- Lab 4 – Verify whole design

Course Outline, C’td.

- Lab 5 (opt.) – pipeline for faster clocking

Housekeeping

- All sections are podcast
- Lecture M, W 1500 in PcynH106
- Discussion
- M, W 1600
- Ctr 105,115
- Open attendance policy

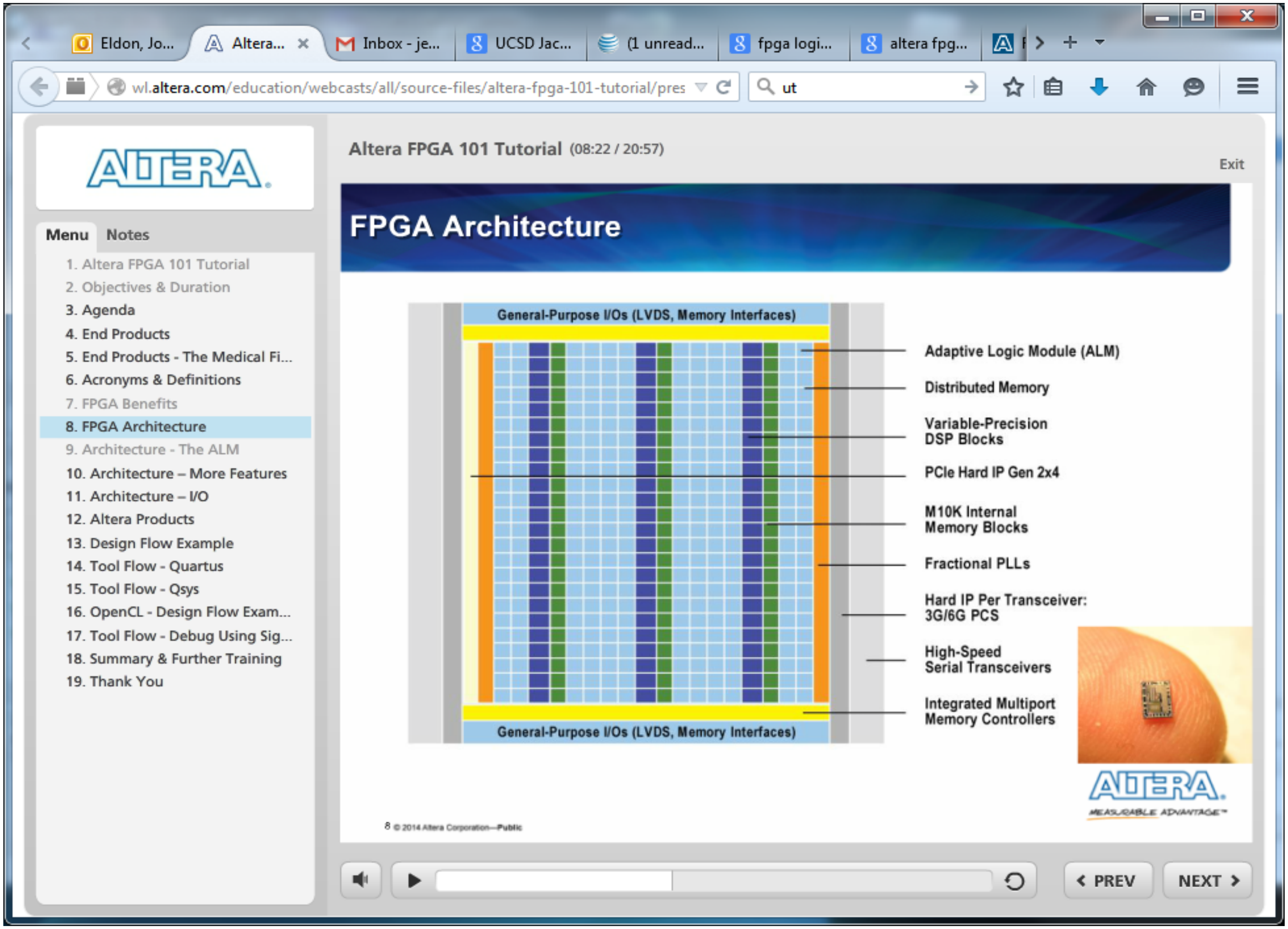
Housekeeping

- Grading
  - 4 lab writeups
  - Class & Piazza participation
  - Everyone can pass w/ a little effort
  - Large majority of students get at least a B-
  - A reserved for those who really work at it
  - A+ for Labs 1 – 4, plus 5

FPGA

- Field Progammable Gate Array
- Macro Architecture
  - array of logic cells, memory
- Micro Architecture
  - anatomy of a logic cell

Macro Architecture



Eldon, Jo... | Altera... | Inbox - je... | UCSD Jac... | (1 unread... | fpga logi... | altera fpg...

wl.altera.com/education/webcasts/all/source-files/altera-fpga-101-tutorial/pres

ut

Altera FPGA 101 Tutorial (08:22 / 20:57)

Exit

Menu Notes

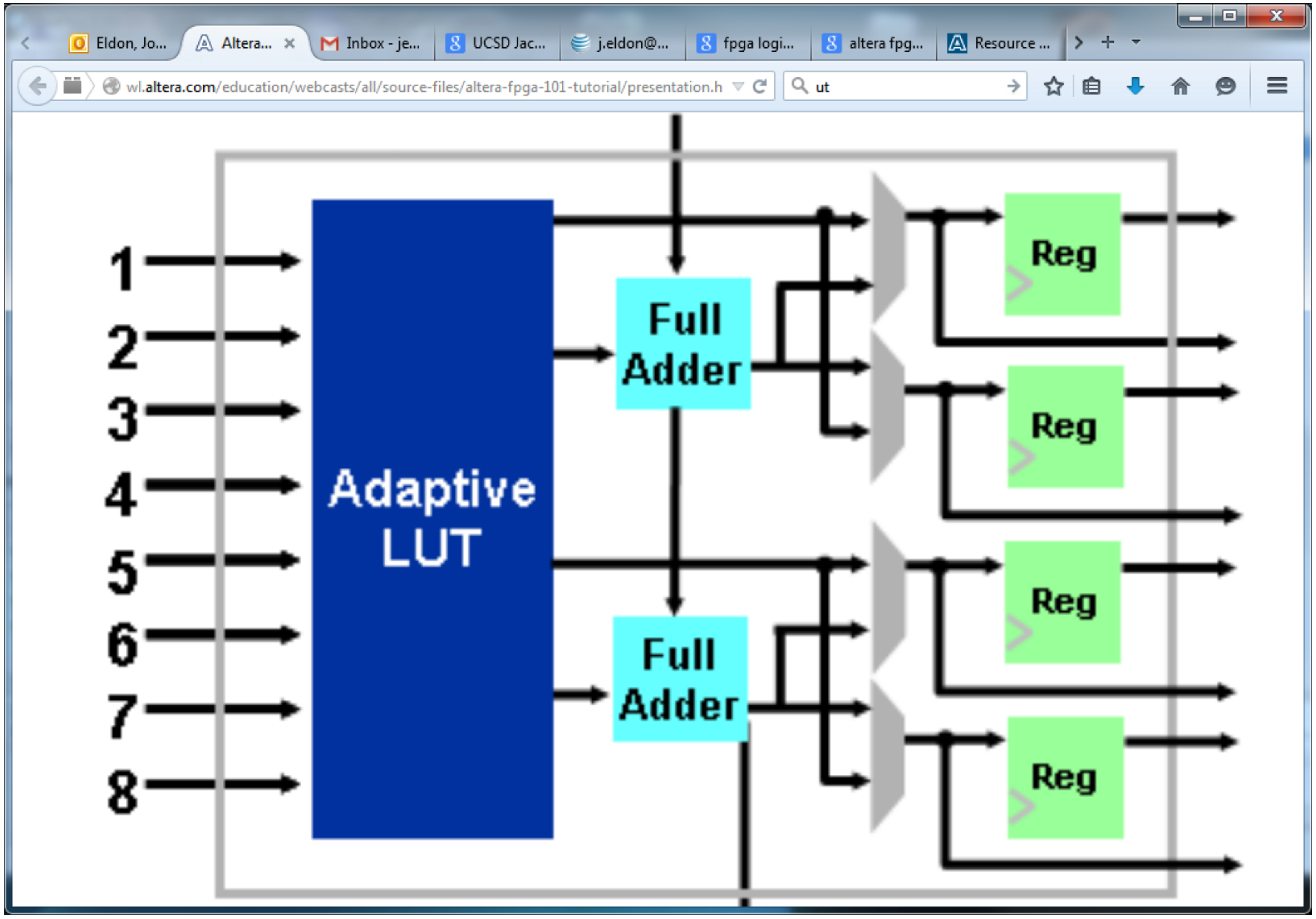
FPGA Architecture

General-Purpose I/Os (LVDS, Memory Interfaces)
Adaptive Logic Module (ALM)
Distributed Memory
Variable-Precision DSP Blocks
PCIe Hard IP Gen 2x4
M10K Internal Memory Blocks
Fractional PLLs
Hard IP Per Transceiver: 3G/6G PCS
High-Speed Serial Transceivers
Integrated Multiport Memory Controllers
General-Purpose I/Os (LVDS, Memory Interfaces)

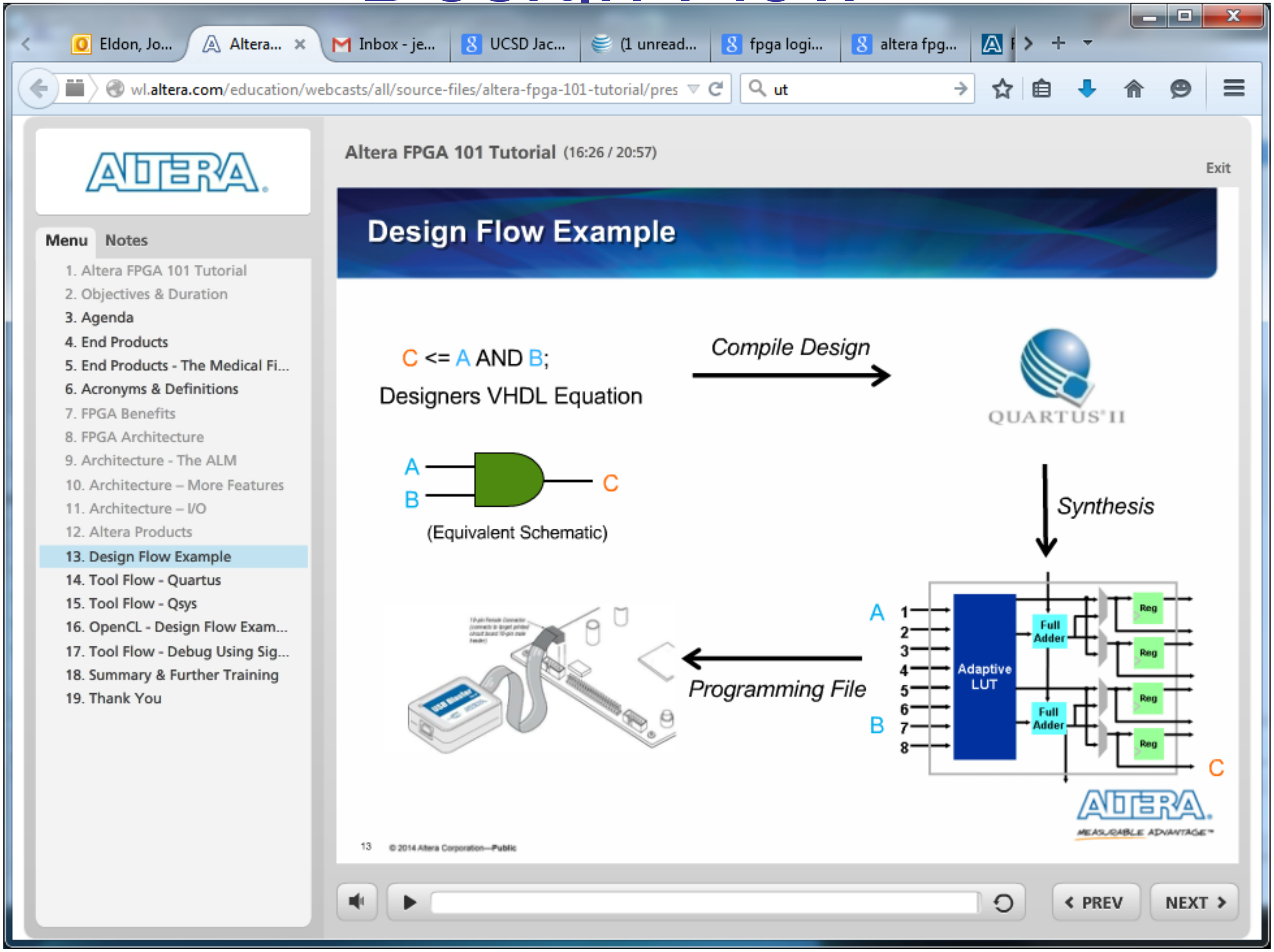
8 © 2014 Altera Corporation—Public

PREV NEXT

Anatomy of a Cell



Design Flow



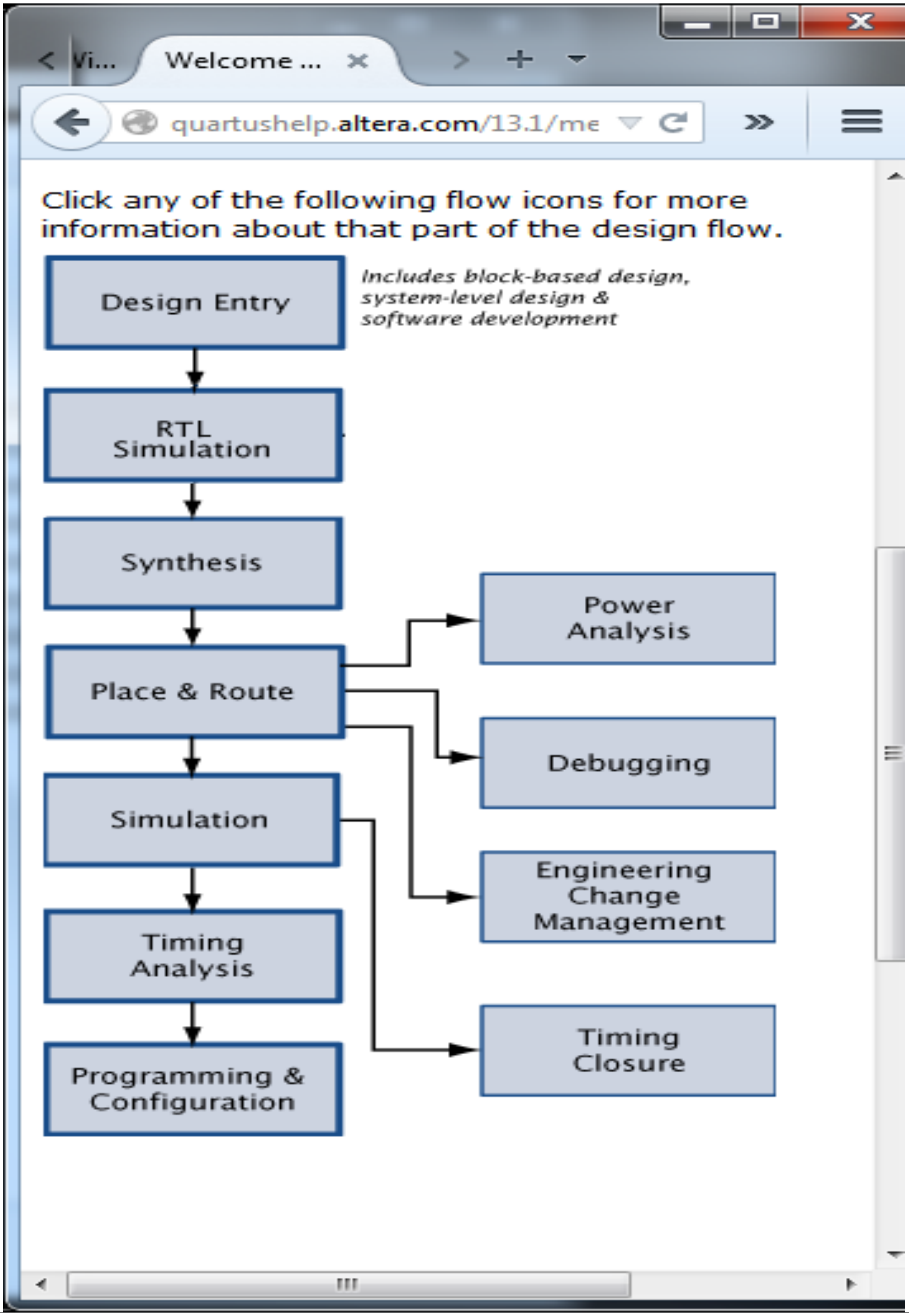
Eldon, Jo...
Altera...
Inbox - je...
UCSD Jac...
(1 unread...
fpga logi...
altera fpg...
wl.altera.com/education/webcasts/all/source-files/altera-fpga-101-tutorial/pres
ut
ALTERA
Menu
Notes
Altera FPGA 101 Tutorial (16:26 / 20:57)
Exit

1. Altera FPGA 101 Tutorial
2. Objectives & Duration
3. Agenda
4. End Products
5. End Products - The Medical Fi...
6. Acronyms & Definitions
7. FPGA Benefits
8. FPGA Architecture
9. Architecture - The ALM
10. Architecture – More Features
11. Architecture – I/O
12. Altera Products
13. Design Flow Example
14. Tool Flow - Quartus
15. Tool Flow - Qsys
16. OpenCL - Design Flow Exam...
17. Tool Flow - Debug Using Sig...
18. Summary & Further Training
19. Thank You

Design Flow Example

C <= A AND B;

Designers VHDL Equation


Vi...
Welcome ...
quartushelp.altera.com/13.1/me

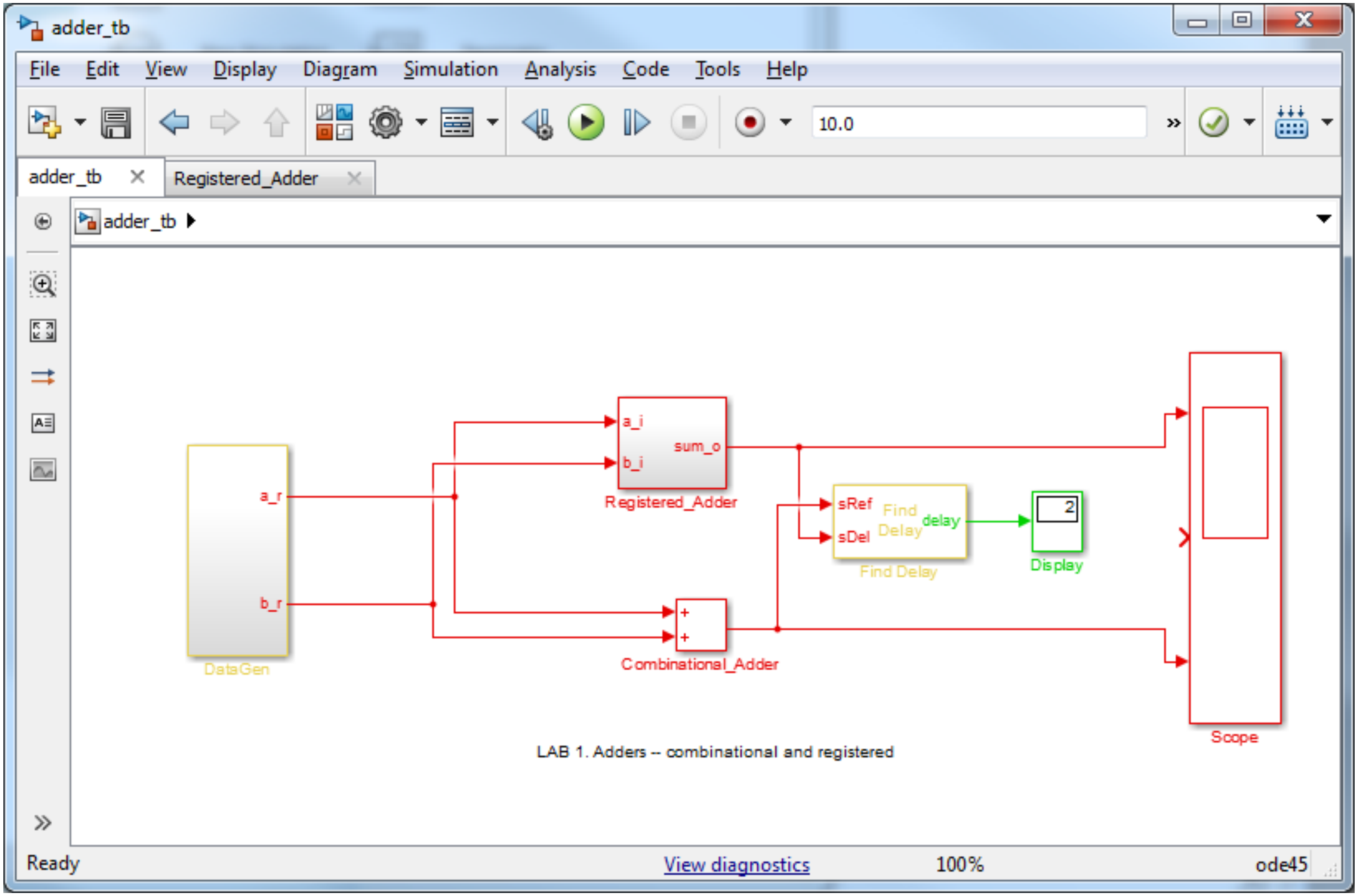
Click any of the following flow icons for more information about that part of the design flow.

Design Entry
Includes block-based design, system-level design & software development
RTL Simulation
Synthesis
Place & Route
Simulation
Timing Analysis
Programming & Configuration
Power Analysis
Debugging
Engineering Change Management
Timing Closure

Simulation vs. Synthesis

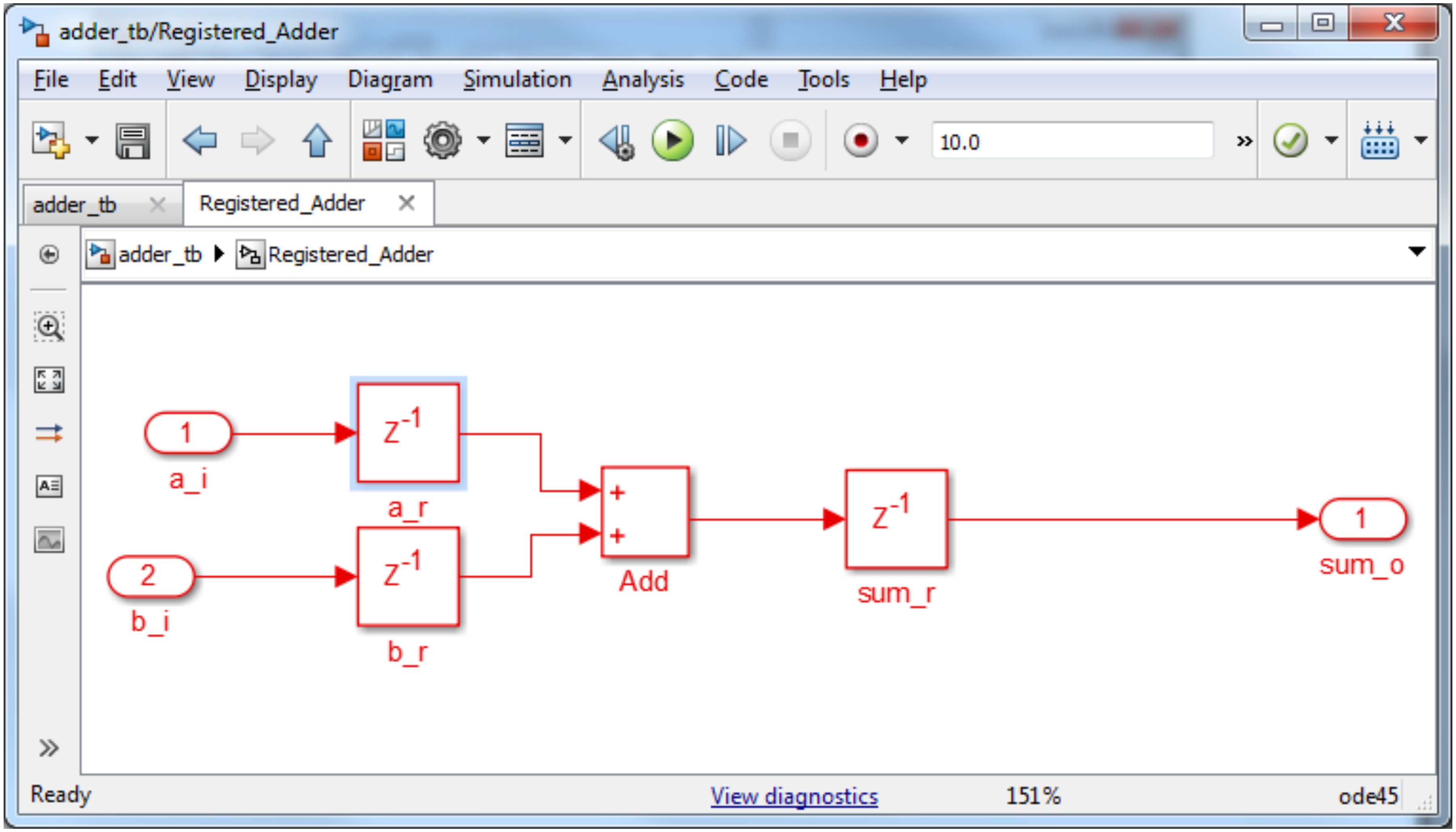
- We **simulate** the testbench and the device under test (DUT), using ModelSim logic simulator, to VERIfy LOGic
- We **synthesize** just the DUT, never the bench, using Quartus, to build real hardware
  - In this class we used this to make sure our design is realizable in hardware and to estimate how fast it could be clocked

Test Bench and DUT



LAB 1. Adders -- combinational and registered

Inside a DUT


adder_tb/Registered_Adder
File Edit View Display Diagram Simulation Analysis Code Tools Help
10.0
adder_tb
Registered_Adder
adder_tb ▸ Registered_Adder
1
a_i
2
b_i
z-1
a_r
z-1
b_r
+
+
Add
z-1
sum_r
1
sum_o
Ready
View diagnostics
151%
ode45

Verilog

- "Verify Logic“
- 1984, Phil Moorby
- Shift from schematic to text based design

SystemVerilog

- "Verify Logic" enhanced
- logic
- typedef struct
- typedef enum
- always_ff, always_comb
- array notation
- a**n and $clog2

ModelSim (Mentor Graphics)

- Simulation tool
- Behavior modeling of .sv files
  - no real sense of timing
  - verifies logical correctness only
- Post-synthesis timing simulation
  - uses .svo and .sdo files from Quartus
  - verifies logic & timing
  - beyond scope of this class ☺

Quartus (Altera)

- logic synthesis tool
- turns Verilog (.sv) into netlist (.svo)
- We need Quartus for three reasons:
  - 1. Verify that our design can be built in REAL hardware
  - 2. Obtain an estimate of how fast said hardware can be clocked/run (Lab 4)
  - 3. Estimate how much of our FPGA is needed