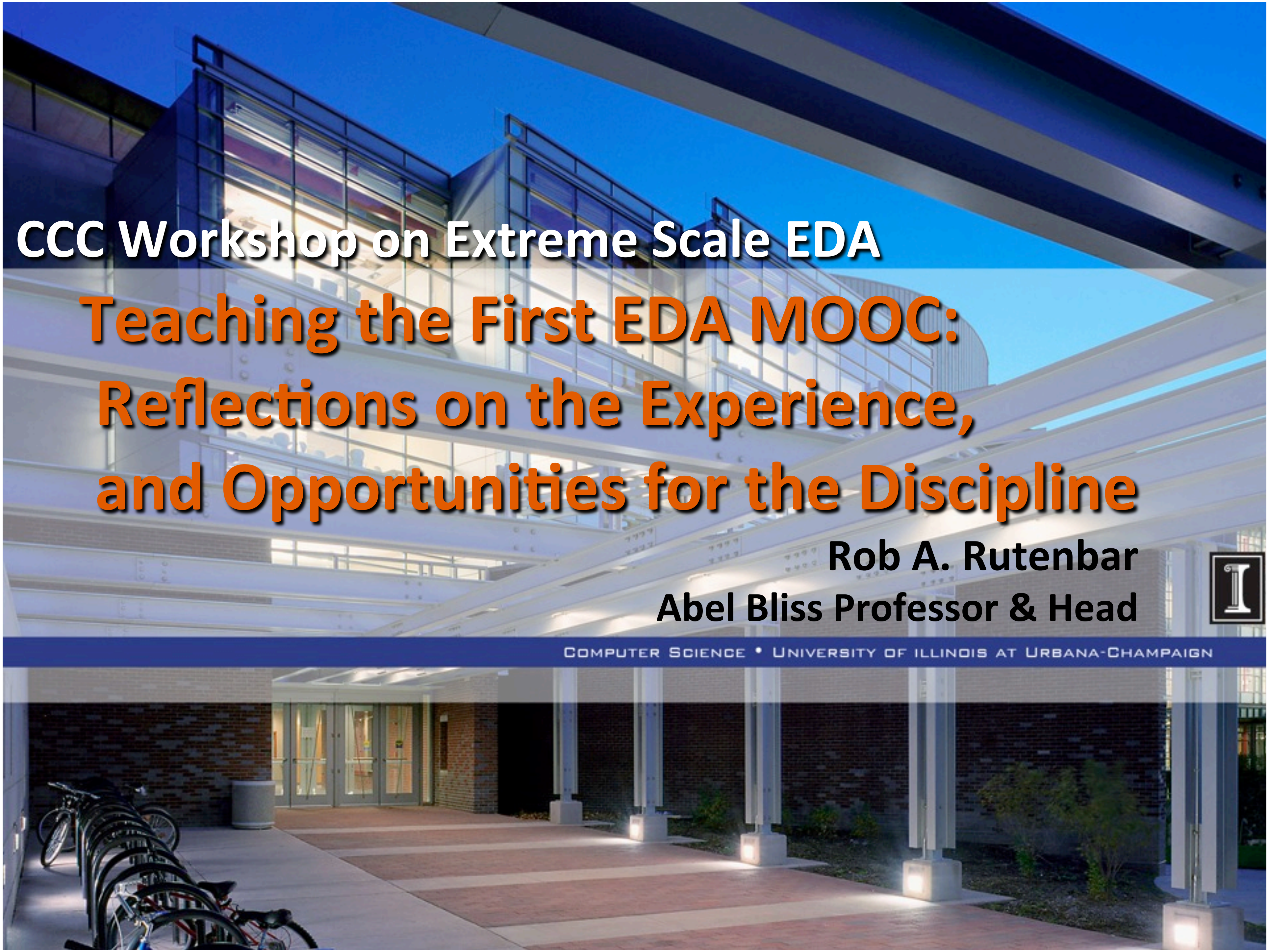


CCC Workshop on Extreme Scale EDA

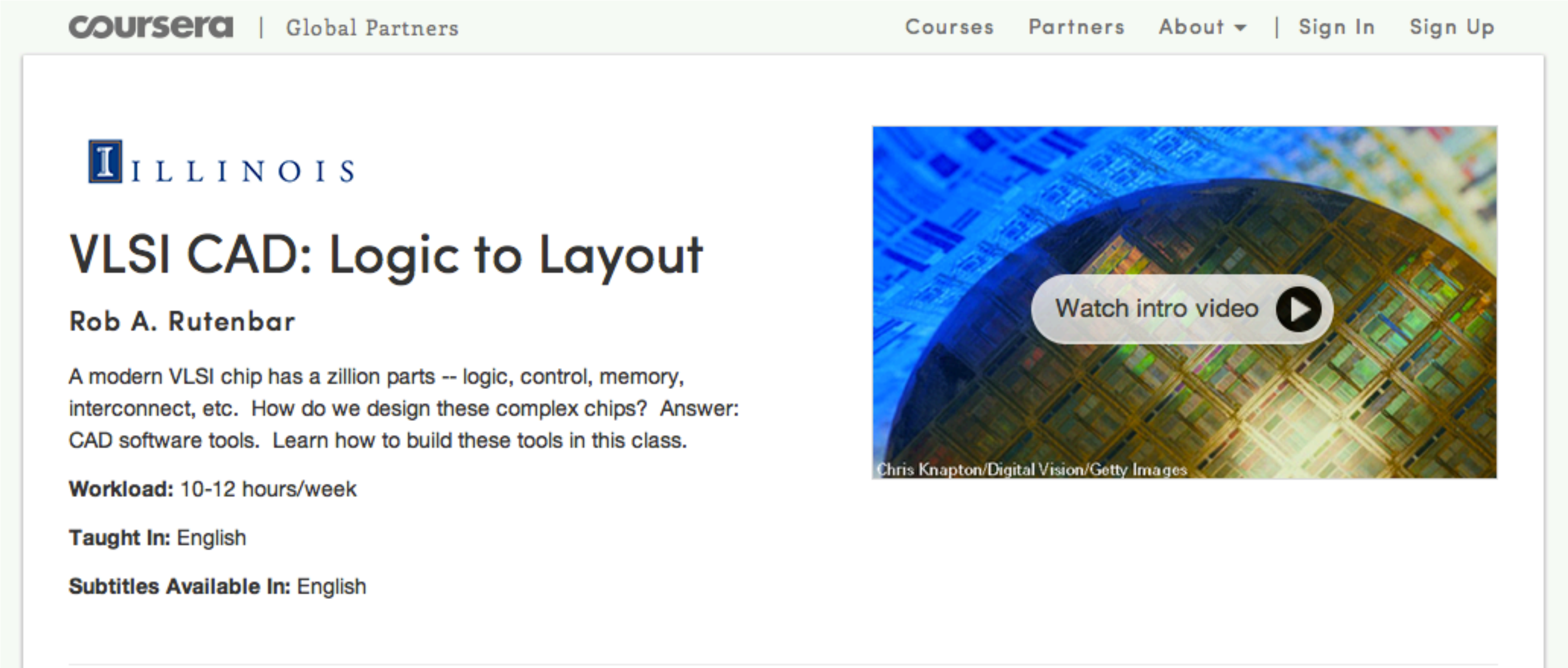
Teaching the First EDA MOOC: Reflections on the Experience, and Opportunities for the Discipline

Rob A. Rutenbar
Abel Bliss Professor & Head

COMPUTER SCIENCE • UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Context

- **From March – May 2013, I did the first-ever MOOC on how we build EDA tools for chip design, on Coursera**
  - In contrast to lots of MOOCS, explicitly a “grad level” course


coursera | Global Partners
Courses
Partners
About
Sign In
Sign Up
Watch intro video
Chris Knapton/Digital Vision/Getty Images

ILLINOIS

VLSI CAD: Logic to Layout

Rob A. Rutenbar

A modern VLSI chip has a zillion parts -- logic, control, memory, interconnect, etc. How do we design these complex chips? Answer: CAD software tools. Learn how to build these tools in this class.

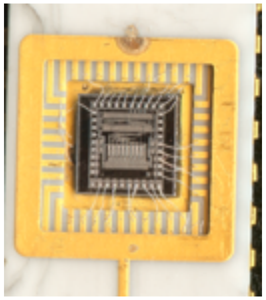
Workload: 10-12 hours/week

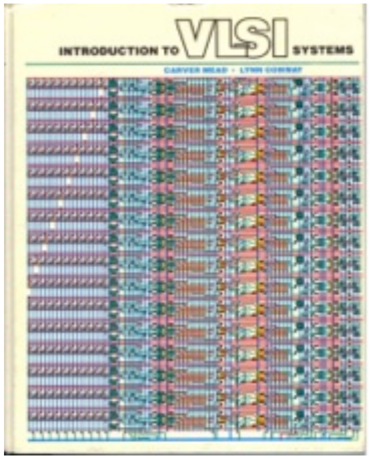
Taught In: English

Subtitles Available In: English

More Context

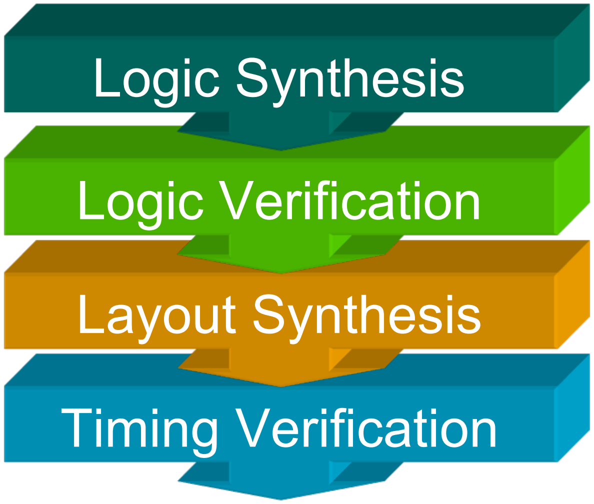
- **I spent 25 years on faculty in ECE at Carnegie Mellon**
- **And I taught this class…**
  - About **15** times over the years
  - To about **700-750** students over these years
  - Class called: **VLSI CAD: Logic to Layout**
    - I "came of age" in Mead&Conway era, so it's "**VLSI**" and "**CAD**" for me…
    - …even tho in course I explained "yeah, it's really **EDA**"





What Does the "Regular" Class Teach?

- **Classical ASIC Flow**
  - Some models
  - Some synthesis
  - Some verification
  - Some logic, some layout
  - Some timing



Logic Synthesis

Logic Verification

Layout Synthesis

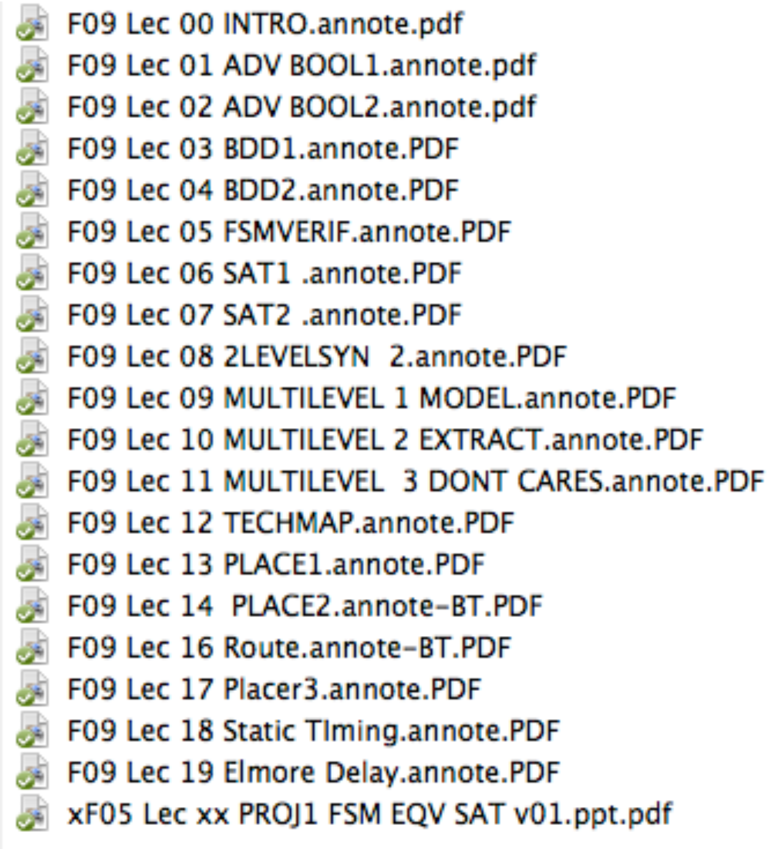
Timing Verification

- **Topics**
  - Computational Boolean Alg
  - Logic verification: BDDs, SAT
  - FSM verification (no model check)
  - Logic synthesis: 2-level, multi-level
  - Technology mapping
  - Timing analysis: static + electrical
  - Placement (Iterative, mincut, analytical)
  - Routing: maze
  - Geometric data structures
  - Extras: eg, Litho simulation

Physical On-Campus Class → MOOC

- **My physical class is**
  - 15 weeks
  - 948 PPT slides
  - 20 separate slide decks
  - Delivered as roughly 14*3 hours (est) = 52 hours of lecs

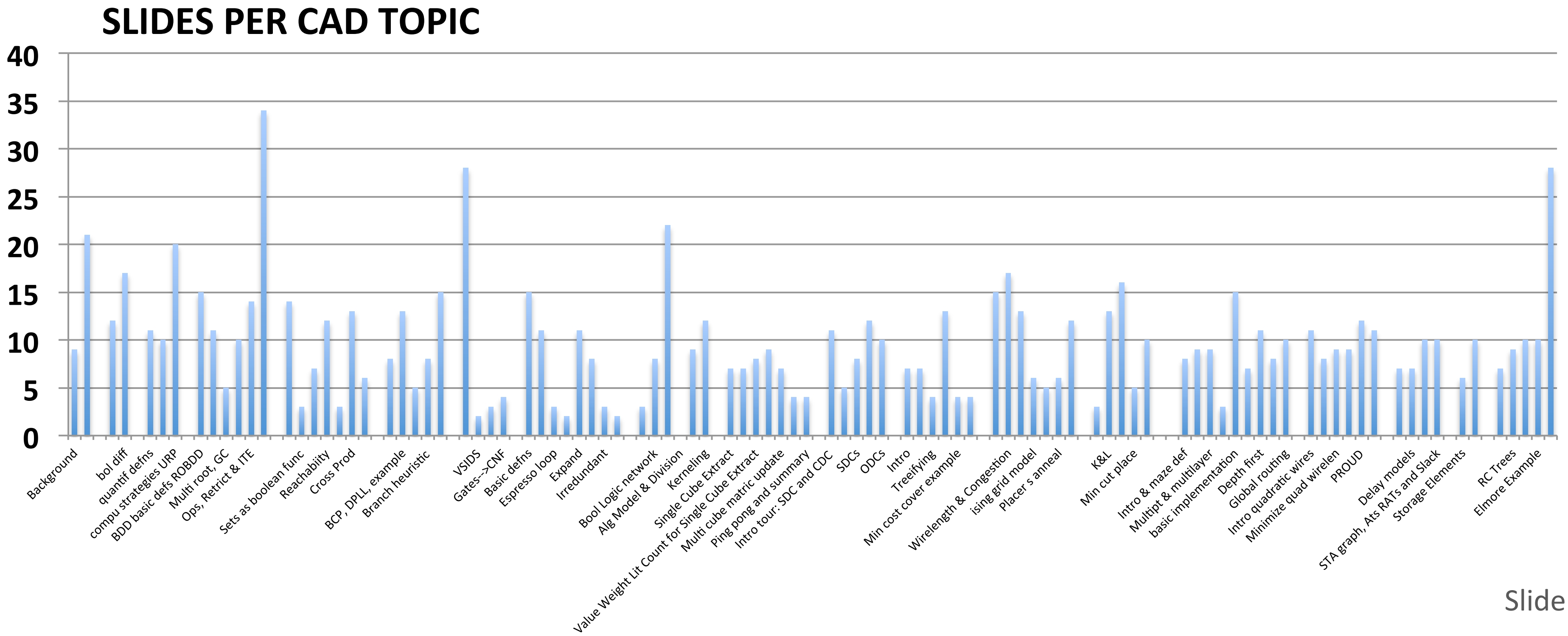
- **But, typical MOOC not same length & intensity as a standard semester**



F09 Lec 00 INTRO.annote.pdf
F09 Lec 01 ADV BOOL1.annote.pdf
F09 Lec 02 ADV BOOL2.annote.pdf
F09 Lec 03 BDD1.annote.PDF
F09 Lec 04 BDD2.annote.PDF
F09 Lec 05 FSMVERIF.annote.PDF
F09 Lec 06 SAT1 .annote.PDF
F09 Lec 07 SAT2 .annote.PDF
F09 Lec 08 2LEVELSYN 2.annote.PDF
F09 Lec 09 MULTILEVEL 1 MODEL.annote.PDF
F09 Lec 10 MULTILEVEL 2 EXTRACT.annote.PDF
F09 Lec 11 MULTILEVEL 3 DONT CARES.annote.PDF
F09 Lec 12 TECHMAP.annote.PDF
F09 Lec 13 PLACE1.annote.PDF
F09 Lec 14 PLACE2.annote-BT.PDF
F09 Lec 16 Route.annote-BT.PDF
F09 Lec 17 Placer3.annote.PDF
F09 Lec 18 Static TIming.annote.PDF
F09 Lec 19 Elmore Delay.annote.PDF
xF05 Lec xx PROJ1 FSM EQV SAT v01.ppt.pdf

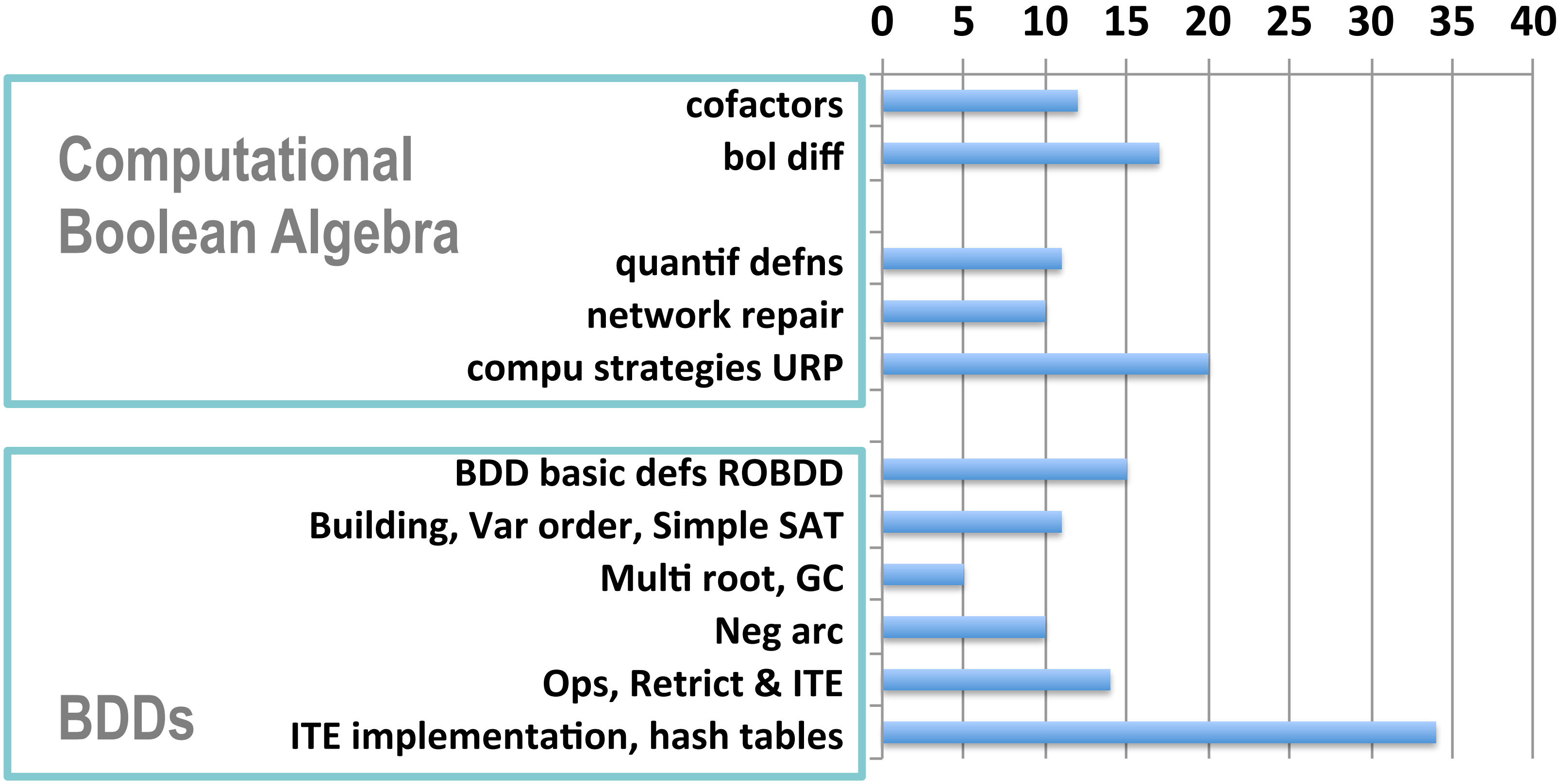
Designing Coverage for my MOOC

- **Walking thru each lec slide by slide, and chunking into individual essential topics, was for me a very useful exercise**
  - I have **~102 separate topics**, with per-topic slides counts below



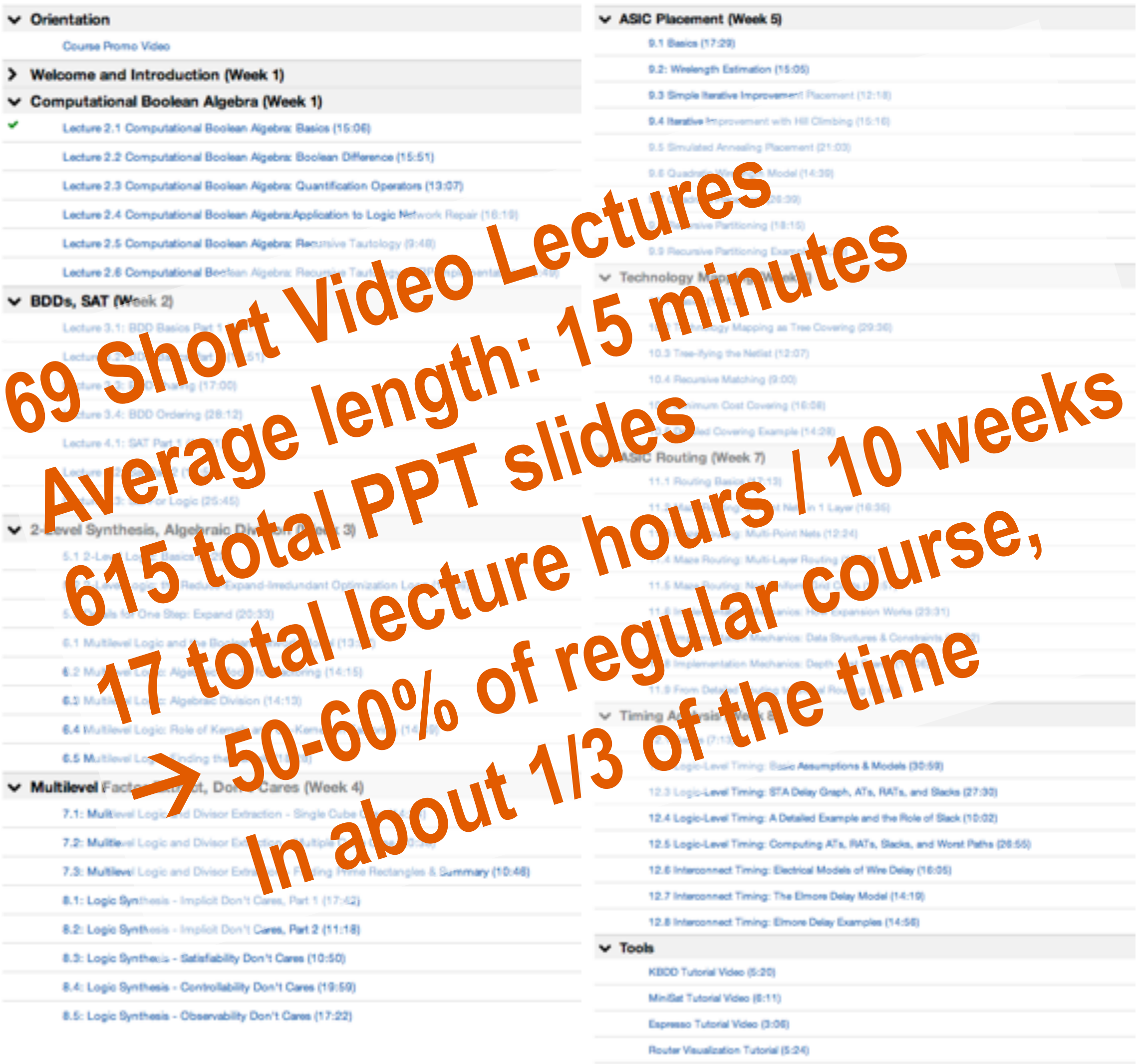
Closeup on Topical Coverage

- **Example of 2 topics, up close, PPT lecture slide counts**



Result: From Regular Course to MOOC

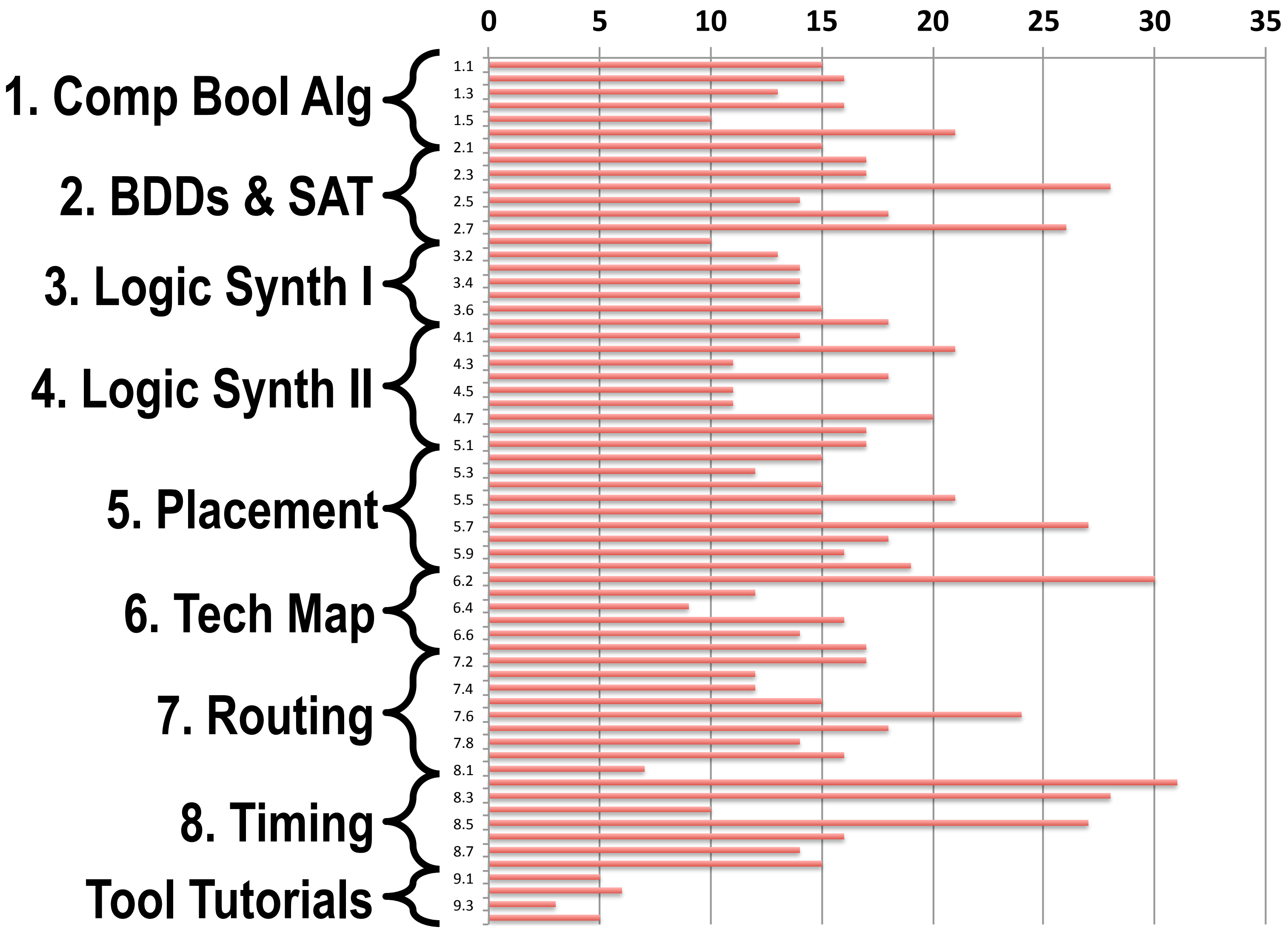
Regular class:
20 Lectures
948 PPT slides



- Orientation
  - Course Promo Video
- Welcome and Introduction (Week 1)
- Computational Boolean Algebra (Week 1)
  - ✓ Lecture 2.1 Computational Boolean Algebra: Basics (15:06)
  - Lecture 2.2 Computational Boolean Algebra: Boolean Difference (15:51)
  - Lecture 2.3 Computational Boolean Algebra: Quantification Operators (13:07)
  - Lecture 2.4 Computational Boolean Algebra:Application to Logic Network Repair (16:19)
  - Lecture 2.5 Computational Boolean Algebra: Recursive Tautology (9:48)
  - Lecture 2.6 Computational Boolean Algebra: Recursive Tautology [illegible] Implementation ([illegible])
- BDDs, SAT (Week 2)
  - Lecture 3.1: BDD Basics Part 1 ([illegible])
  - Lecture 3.2: BDD Basics Part 2 ([illegible]:51)
  - Lecture 3.3: BDD Sharing (17:00)
  - Lecture 3.4: BDD Ordering (28:12)
  - Lecture 4.1: SAT Part 1 ([illegible])
  - Lecture 4.2: SAT Part 2 ([illegible])
  - Lecture 4.3: SAT for Logic (25:45)
- 2-Level Synthesis, Algebraic Division (Week 3)
  - 5.1 2-Level Logic Basics ([illegible])
  - 5.2 2-Level Logic: the Reduce-Expand-Irredundant Optimization Loop ([illegible])
  - 5.3 Details for One Step: Expand (20:33)
  - 6.1 Multilevel Logic and the Boolean Network Model (13:[illegible])
  - 6.2 Multilevel Logic: Algebraic Model for Factoring (14:15)
  - 6.3 Multilevel Logic: Algebraic Division (14:13)
  - 6.4 Multilevel Logic: Role of Kernels and Co-Kernels [illegible] (14:[illegible])
  - 6.5 Multilevel Logic: Finding the Kernels ([illegible])
- Multilevel Factor Extract, Don't Cares (Week 4)
  - 7.1: Multilevel Logic and Divisor Extraction - Single Cube [illegible] ([illegible])
  - 7.2: Multilevel Logic and Divisor Extraction - Multiple Cube [illegible] ([illegible])
  - 7.3: Multilevel Logic and Divisor Extraction - Finding Prime Rectangles & Summary (10:48)
  - 8.1: Logic Synthesis - Implicit Don't Cares, Part 1 (17:42)
  - 8.2: Logic Synthesis - Implicit Don't Cares, Part 2 (11:18)
  - 8.3: Logic Synthesis - Satisfiability Don't Cares (10:50)
  - 8.4: Logic Synthesis - Controllability Don't Cares (19:59)
  - 8.5: Logic Synthesis - Observability Don't Cares (17:22)
- ASIC Placement (Week 5)
  - 9.1 Basics (17:29)
  - 9.2: Wirelength Estimation (15:05)
  - 9.3 Simple Iterative Improvement Placement (12:18)
  - 9.4 Iterative Improvement with Hill Climbing (15:16)
  - 9.5 Simulated Annealing Placement (21:03)
  - 9.6 Quadratic Wirelength Model (14:39)
  - 9.7 Quadratic Placement (26:39)
  - 9.8 Recursive Partitioning (18:15)
  - 9.9 Recursive Partitioning Example ([illegible])
- Technology Mapping (Week 6)
  - 10.1 Basics ([illegible])
  - 10.2 Technology Mapping as Tree Covering (29:36)
  - 10.3 Tree-ifying the Netlist (12:07)
  - 10.4 Recursive Matching (9:00)
  - 10.5 Minimum Cost Covering (16:08)
  - 10.6 Detailed Covering Example (14:28)
- ASIC Routing (Week 7)
  - 11.1 Routing Basics (17:13)
  - 11.2 Maze Routing: 2-Point Nets in 1 Layer (16:35)
  - 11.3 Maze Routing: Multi-Point Nets (12:24)
  - 11.4 Maze Routing: Multi-Layer Routing ([illegible])
  - 11.5 Maze Routing: Non-Uniform Grid Costs ([illegible])
  - 11.6 Implementation Mechanics: How Expansion Works (23:31)
  - 11.7 Implementation Mechanics: Data Structures & Constraints ([illegible])
  - 11.8 Implementation Mechanics: Depth-First Search ([illegible])
  - 11.9 From Detailed Routing to Global Routing ([illegible])
- Timing Analysis (Week 8)
  - 12.1 Basics (7:1[illegible])
  - 12.2 Logic-Level Timing: Basic Assumptions & Models (30:59)
  - 12.3 Logic-Level Timing: STA Delay Graph, ATs, RATs, and Slacks (27:30)
  - 12.4 Logic-Level Timing: A Detailed Example and the Role of Slack (10:02)
  - 12.5 Logic-Level Timing: Computing ATs, RATs, Slacks, and Worst Paths (26:55)
  - 12.6 Interconnect Timing: Electrical Models of Wire Delay (16:05)
  - 12.7 Interconnect Timing: The Elmore Delay Model (14:19)
  - 12.8 Interconnect Timing: Elmore Delay Examples (14:56)
- Tools
  - KBDD Tutorial Video (5:20)
  - MiniSat Tutorial Video (6:11)
  - Espresso Tutorial Video (3:06)
  - Router Visualization Tutorial (5:24)

69 Short Video Lectures
Average length: 15 minutes
615 total PPT slides
17 total lecture hours / 10 weeks
→ 50-60% of regular course,
In about 1/3 of the time

MOOC Video Content: Minutes/Lecture, by Week



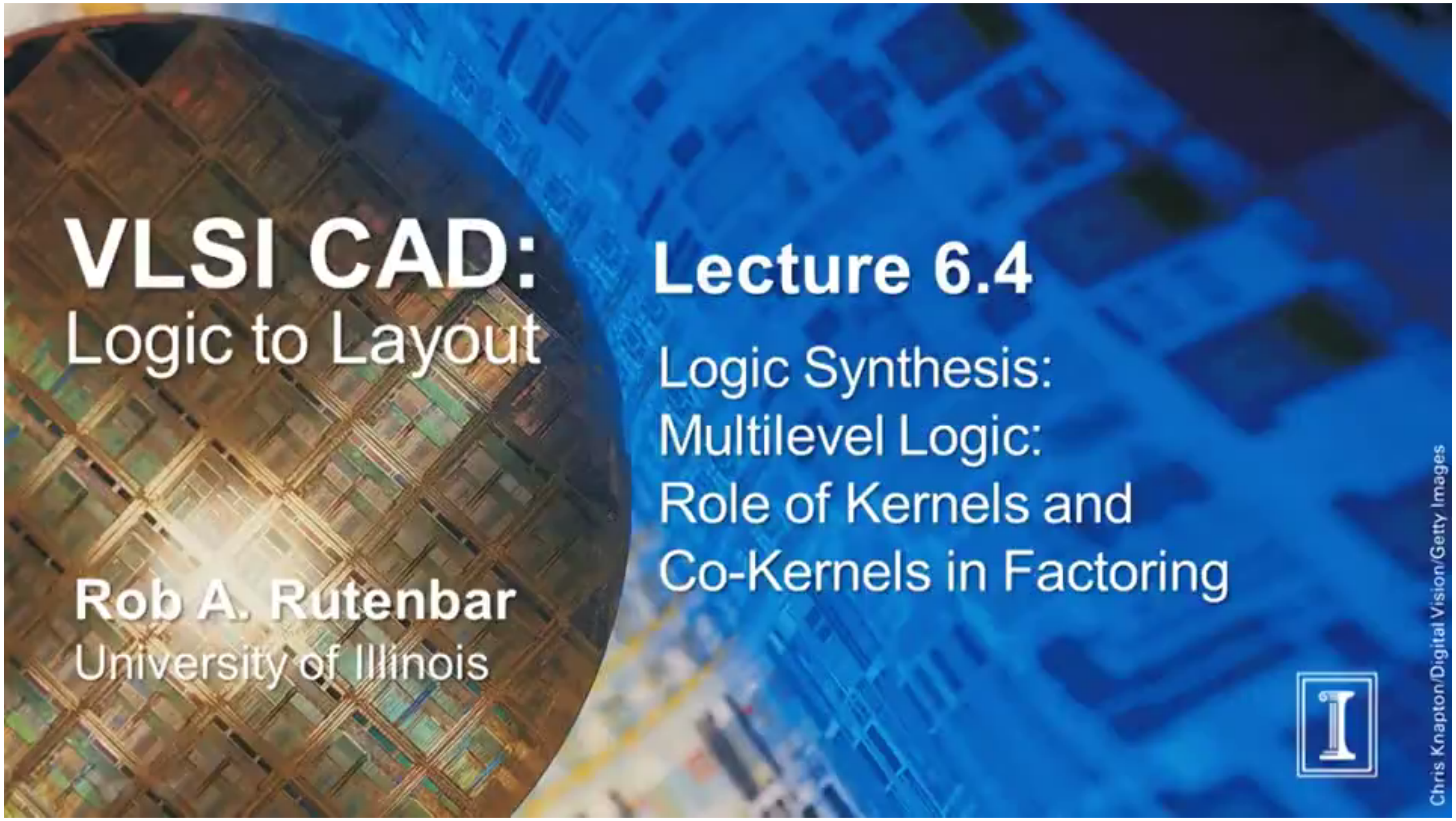
Why Many Short Video Segments?

- **Pedagogy**
  - Better retention with short bursts of focused instruction
  - Emphasize one (or few) core topics

- **Bandwidth**
  - In much of planet, cannot download 2-hour video as 1 chunk
  - So– keep vids short. Don’t stream “live”. Download, view later
  - Critical to respect infrastructure realities for successful MOOC

What’s In A Video Lecture?

- **Start with “talking Rob head” intro of lecture topic**



VLSI CAD:
Logic to Layout

Lecture 6.4

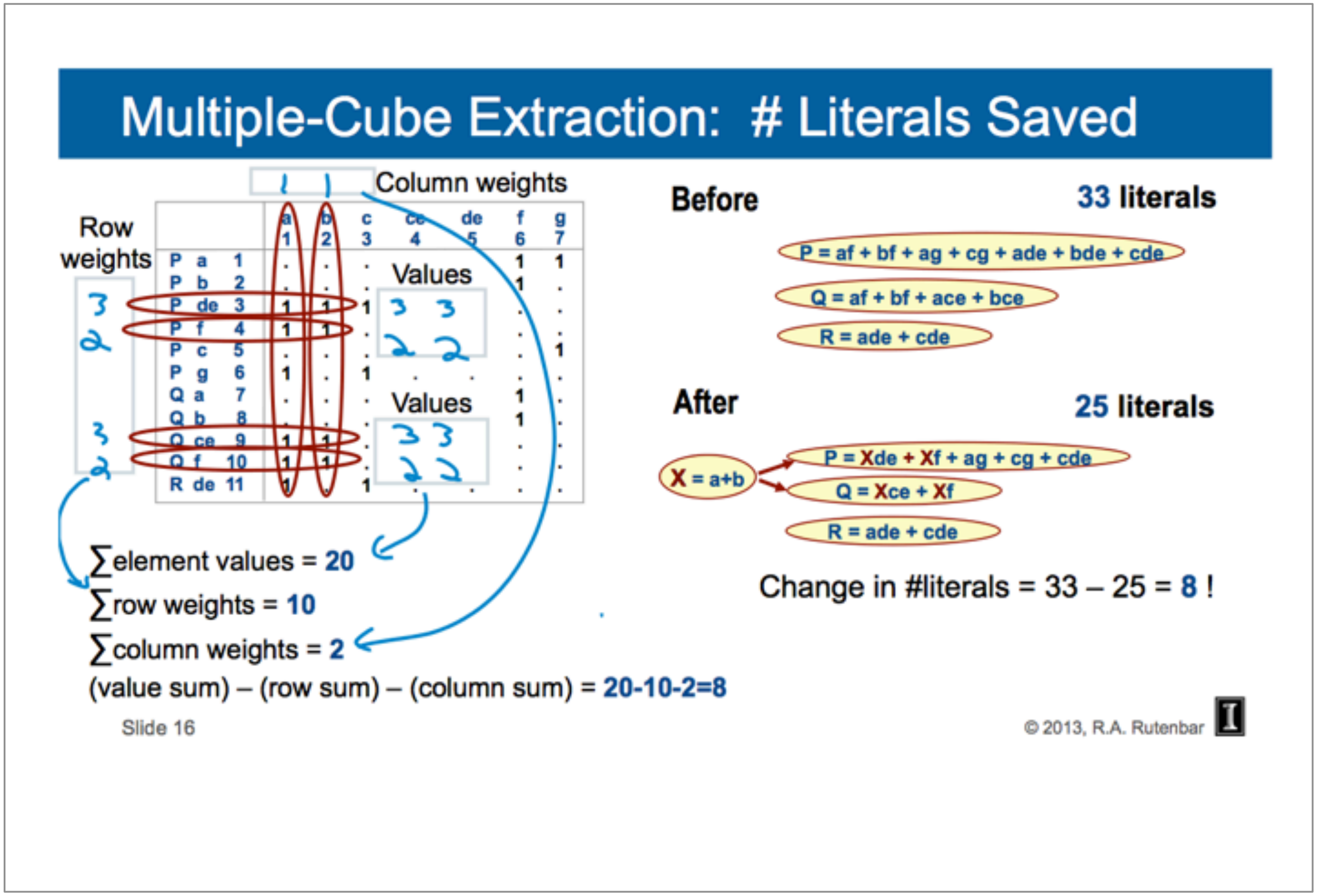
Logic Synthesis:
Multilevel Logic:
Role of Kernels and
Co-Kernels in Factoring

Rob A. Rutenbar
University of Illinois

Chris Knapton/Digital Vision/Getty Images

What’s In A Video Lecture?

- **Most content is me writing-on-the-slides of lec + voiceover**



Multiple-Cube Extraction: # Literals Saved

				a	b	c	ce	de	f	g
Column weights				1	1					
Row weights				1	2	3	4	5	6	7
	P	a	1	.	.	.			1	1
	P	b	2	.	.	.			1	.
3	P	de	3	1	1	1	3	3	.	.
2	P	f	4	1	1	.	2	2	.	.
	P	c	5	.	.	.			.	1
	P	g	6	1	.	1	.	.	.	.
	Q	a	7	.	.	.			1	.
	Q	b	8	.	.	.			1	.
3	Q	ce	9	1	1	.	3	3	.	.
2	Q	f	10	1	1	.	2	2	.	.
	R	de	11	1	.	1	.	.	.	.

Values

Values

∑element values = **20**

∑row weights = **10**

∑column weights = **2**

(value sum) – (row sum) – (column sum) = **20-10-2=8**

Before **33 literals**

P = af + bf + ag + cg + ade + bde + cde

Q = af + bf + ace + bce

R = ade + cde

After **25 literals**

X = a+b

P = Xde + Xf + ag + cg + cde

Q = Xce + Xf

R = ade + cde

Change in #literals = 33 – 25 = **8** !

Slide 16

© 2013, R.A. Rutenbar

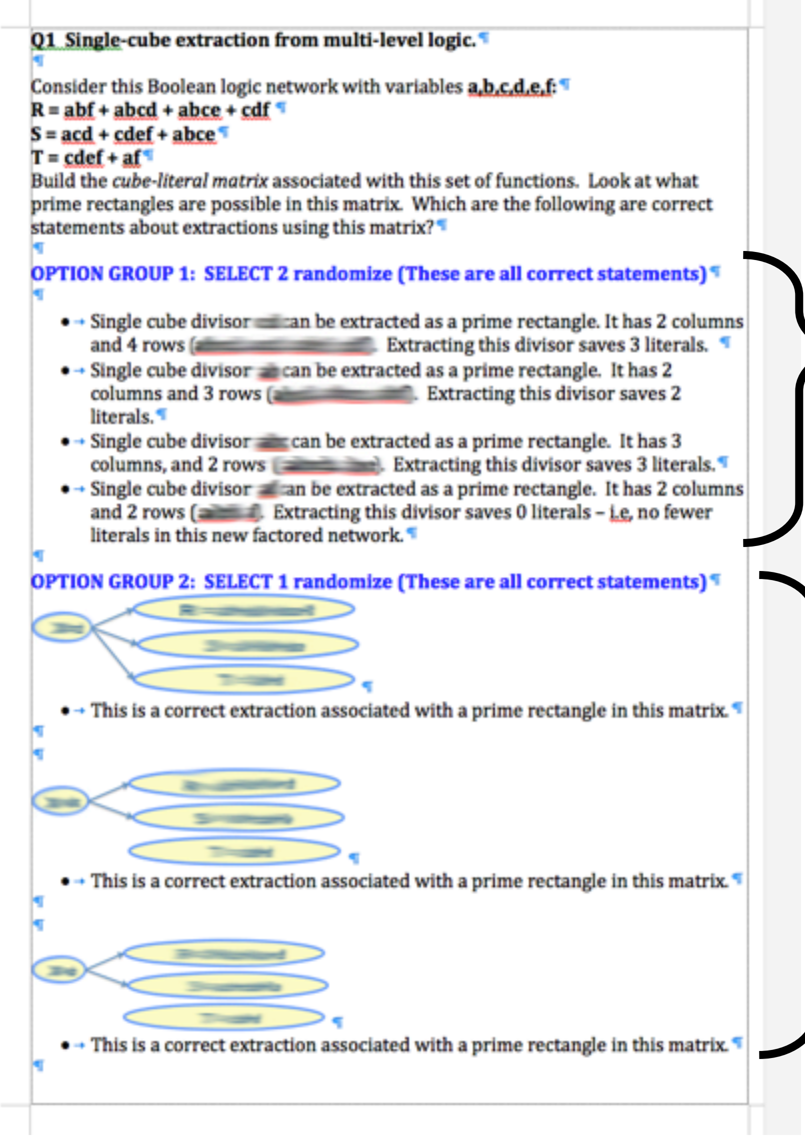
MOOC Assignments

- **8 weekly homeworks, aka, Problem Sets**
  - True/False
  - Multiple Choice
  - Type a number in a box, etc

- **What's *different* than in-class versions?**
  - Can't do math derivations
  - Must "over supply" sub-problems, and randomize (cheating)
  - Partition big problems into smaller, step-by-step parts

- **Took about 3 HW sets to "get this right", style-wise**

Example: One HW Problem Spec



Q1 Single-cube extraction from multi-level logic.

Consider this Boolean logic network with variables a,b,c,d,e,f:
R = abf + abcd + abce + cdf
S = acd + cdef + abce
T = cdef + af
Build the *cube-literal matrix* associated with this set of functions. Look at what prime rectangles are possible in this matrix. Which are the following are correct statements about extractions using this matrix?

OPTION GROUP 1: SELECT 2 randomize (These are all correct statements)

- Single cube divisor [redacted]an be extracted as a prime rectangle. It has 2 columns and 4 rows ([redacted]. Extracting this divisor saves 3 literals.
- Single cube divisor [redacted] can be extracted as a prime rectangle. It has 2 columns and 3 rows ([redacted]. Extracting this divisor saves 2 literals.
- Single cube divisor [redacted] can be extracted as a prime rectangle. It has 3 columns, and 2 rows ([redacted]. Extracting this divisor saves 3 literals.
- Single cube divisor [redacted]an be extracted as a prime rectangle. It has 2 columns and 2 rows ([redacted]. Extracting this divisor saves 0 literals – i.e, no fewer literals in this new factored network.

OPTION GROUP 2: SELECT 1 randomize (These are all correct statements)

[redacted]

- This is a correct extraction associated with a prime rectangle in this matrix.

[redacted]

- This is a correct extraction associated with a prime rectangle in this matrix.

[redacted]

- This is a correct extraction associated with a prime rectangle in this matrix.

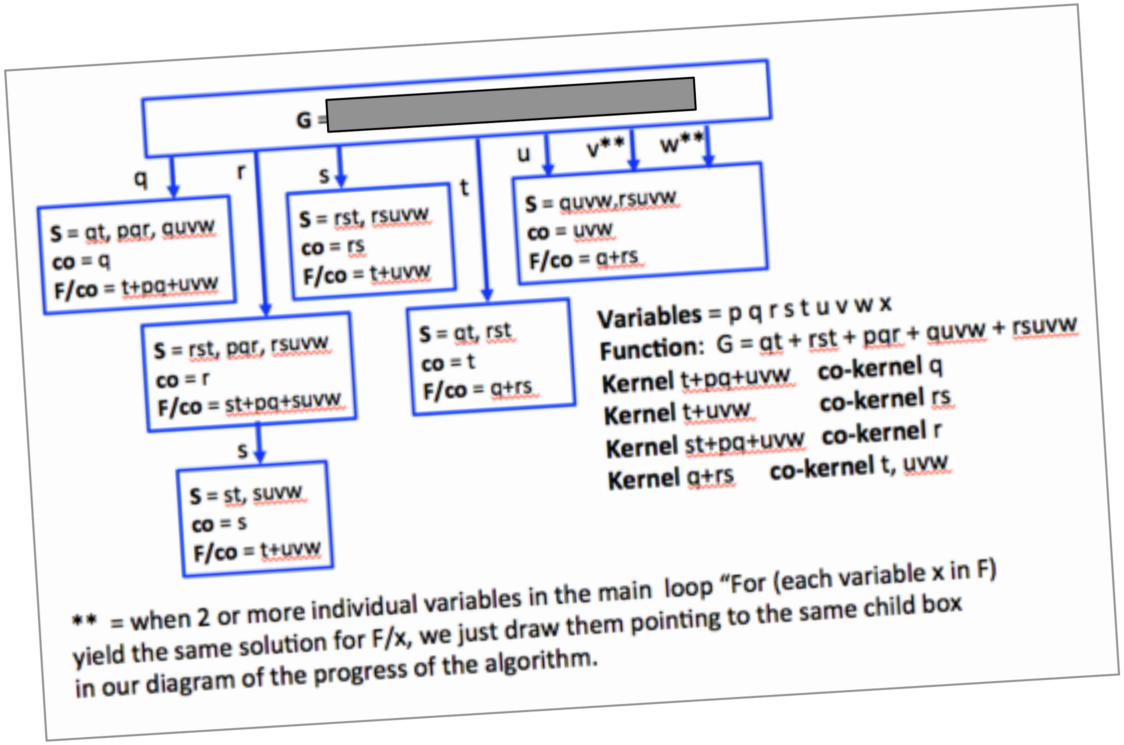
Randomize selection from this 1st group of sub-questions

Randomize selection from this 2nd group of sub-questions

Etc…

MOOC Assignments

- **Lots of criticism of MOOCs being "dumbed down"**
  - Yes, they are a bit simpler. No, not a lot "dumber"
  - Mostly, burden on instructor to **design smart assignments**
  - Ex: this is the "macro answer" to **one** factoring HW problem



G = [redacted]

S = qt, pqr, quvw
co = q
F/co = t+pq+uvw

S = rst, rsuvw
co = rs
F/co = t+uvw

S = quvw,rsuvw
co = uvw
F/co = q+rs

S = rst, pqr, rsuvw
co = r
F/co = st+pq+suvw

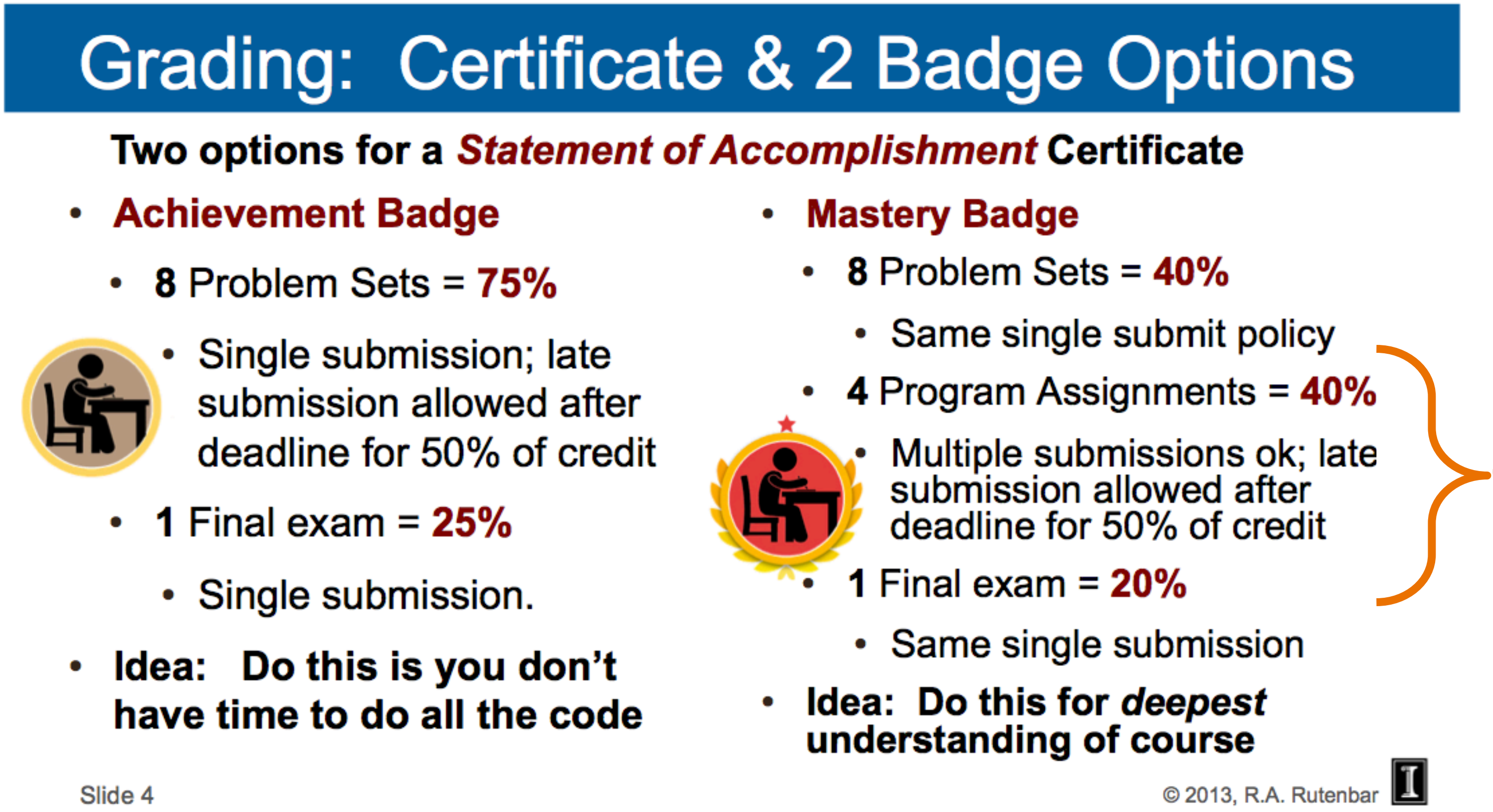
S = qt, rst
co = t
F/co = q+rs

S = st, suvw
co = s
F/co = t+uvw

Variables = p q r s t u v w x
Function: G = qt + rst + pqr + quvw + rsuvw
Kernel t+pq+uvw **co-kernel** q
Kernel t+uvw **co-kernel** rs
Kernel st+pq+uvw **co-kernel** r
Kernel q+rs **co-kernel** t, uvw

** = when 2 or more individual variables in the main loop "For (each variable x in F) yield the same solution for F/x, we just draw them pointing to the same child box in our diagram of the progress of the algorithm.

MOOC Assignments: 2 Paths Thru Class



Grading: Certificate & 2 Badge Options

Two options for a *Statement of Accomplishment* Certificate

- **Achievement Badge**
  - **8** Problem Sets = **75%**
    - Single submission; late submission allowed after deadline for 50% of credit
  - **1** Final exam = **25%**
    - Single submission.
- **Idea: Do this is you don't have time to do all the code**

- **Mastery Badge**
  - **8** Problem Sets = **40%**
    - Same single submit policy
  - **4** Program Assignments = **40%**
    - Multiple submissions ok; late submission allowed after deadline for 50% of credit
  - **1** Final exam = **20%**
    - Same single submission
- **Idea: Do this for *deepest* understanding of course**

Slide 4

© 2013, R.A. Rutenbar

CODE!

EDA MOOC: Some Philosophy

- **I don't believe you can teach a serious EDA course without:**
  - **Experimenting** with some existing EDA tools
  - **Designing software** to build some EDA tools

- **How to do this at scale? To 10,000 students?**
  - CDNS, SNPS, etc, are **not** going to give us free stuff
  - Lots of IP, Licensing, etc, landmines: you get to know your university's top lawyers on a first name basis, in your MOOC

- **Answers**
  - Carefully select **open source software**
  - Write everything else **yourself**

VLSI CAD MOOC: Software Ecosystem

VLSI CAD: Logic to Layout

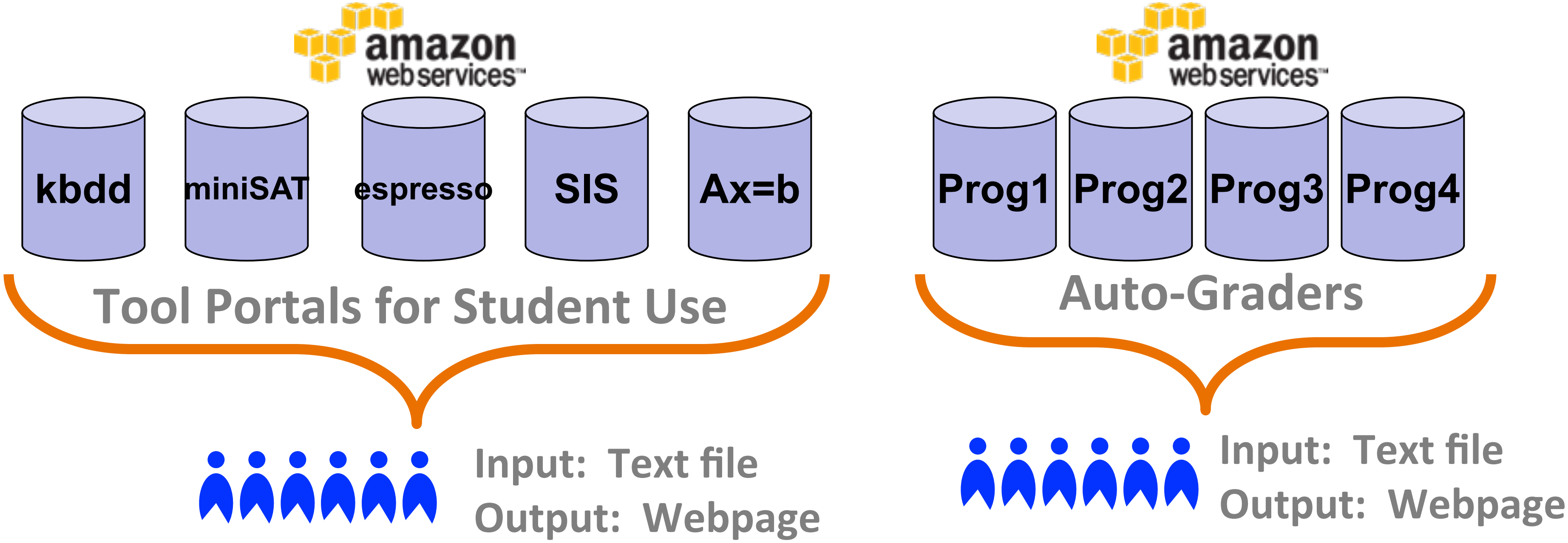
Lecture 1

Welcome & Introduction

Rob A. Rutenbar
University of Illinois

coursera

amazon web services™



amazon web services™

kbdd

miniSAT

espresso

SIS

Ax=b

Tool Portals for Student Use

Input: Text file
Output: Webpage

amazon web services™

Prog1

Prog2

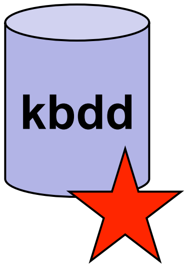
Prog3

Prog4

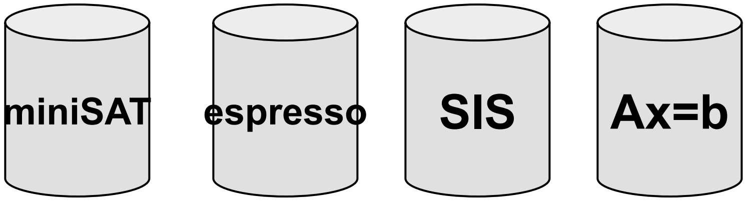
Auto-Graders

Input: Text file
Output: Webpage

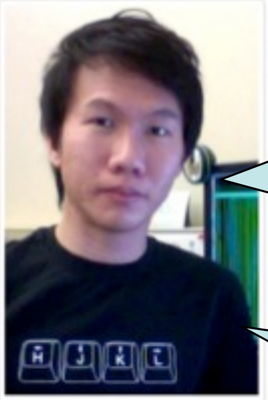
Aside: Building this Software Ecosystem



kbdd



miniSAT espresso SIS Ax=b



Nick Chen

Teaching Assistant

Rob – what is this stuff? Is this **K&R C** or what??

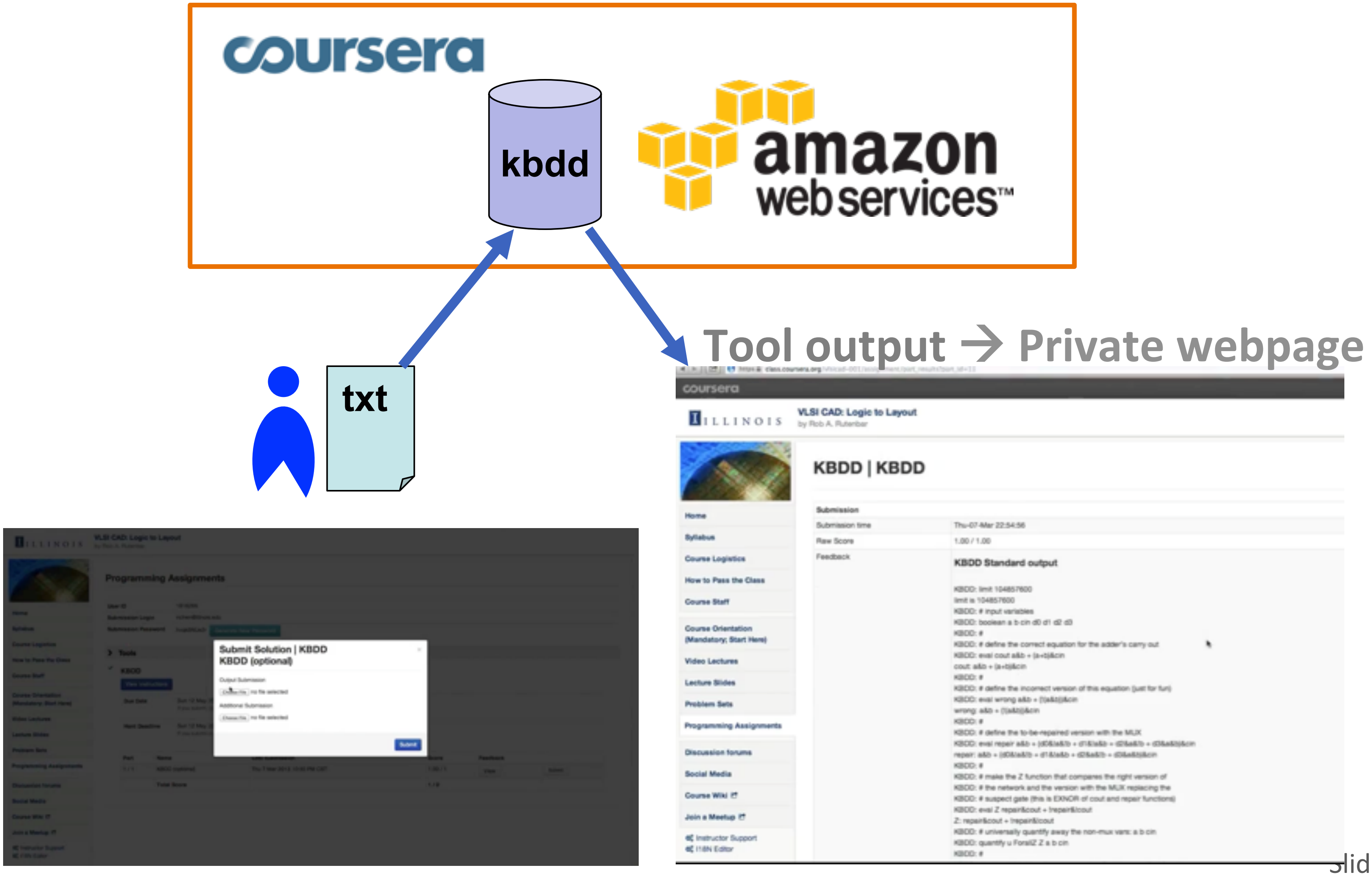
Ohhhh….

Yes.

Rob A. Rutenbar

Instructor

VLSI CAD MOOC: Software Ecosystem



Tool output → Private webpage

ILLINOIS VLSI CAD: Logic to Layout

Programming Assignments

Tools

KBDD

Submit Solution | KBDD
KBDD (optional)

Output Submission

Choose File no file selected

Additional Submission

Choose File no file selected

Submit

class.coursera.org

coursera

ILLINOIS **VLSI CAD: Logic to Layout**
by Rob A. Rutenbar

Home
Syllabus
Course Logistics
How to Pass the Class
Course Staff
Course Orientation (Mandatory; Start Here)
Video Lectures
Lecture Slides
Problem Sets
Programming Assignments
Discussion forums
Social Media
Course Wiki
Join a Meetup
Instructor Support
I18N Editor

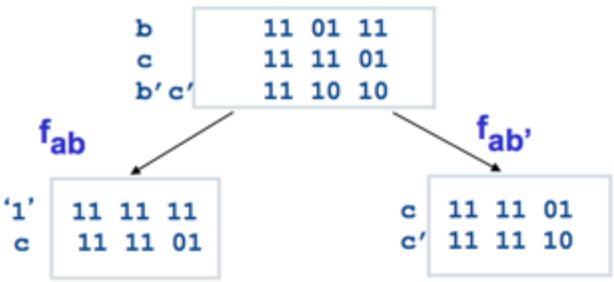
KBDD | KBDD

Submission	
Submission time	Thu-07-Mar 22:54:56
Raw Score	1.00 / 1.00
Feedback	**KBDD Standard output**

```
KBDD: limit 104857600
limit is 104857600
KBDD: # input variables
KBDD: boolean a b cin d0 d1 d2 d3
KBDD: #
KBDD: # define the correct equation for the adder's carry out
KBDD: eval cout a&b + (a+b)&cin
cout: a&b + (a+b)&cin
KBDD: #
KBDD: # define the incorrect version of this equation (just for fun)
KBDD: eval wrong a&b + (!(a&b))&cin
wrong: a&b + (!(a&b))&cin
KBDD: #
KBDD: # define the to-be-repaired version with the MUX
KBDD: eval repair a&b + (d0&!a&!b + d1&!a&b + d2&a&!b + d3&a&b)&cin
repair: a&b + (d0&!a&!b + d1&!a&b + d2&a&!b + d3&a&b)&cin
KBDD: #
KBDD: # make the Z function that compares the right version of
KBDD: # the network and the version with the MUX replacing the
KBDD: # suspect gate (this is EXNOR of cout and repair functions)
KBDD: eval Z repair&cout + !repair&!cout
Z: repair&cout + !repair&!cout
KBDD: # universally quantify away the non-mux vars: a b cin
KBDD: quantify u ForallZ Z a b cin
KBDD: #
```

VLSI CAD MOOC: 4 Software Projects

1. Boolean Data Structures & Computation (URP,PCN)



b 11 01 11
c 11 11 01
b' c' 11 10 10

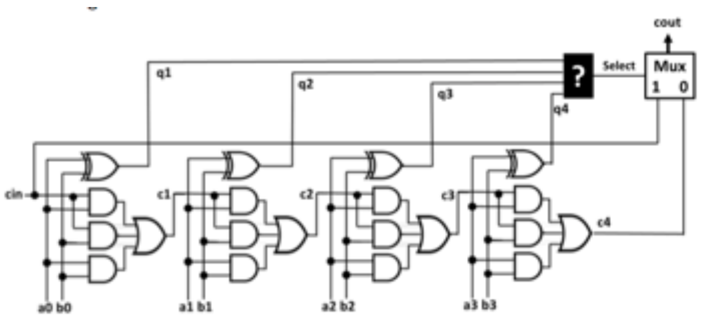
f_{ab}

'1' 11 11 11
c 11 11 01

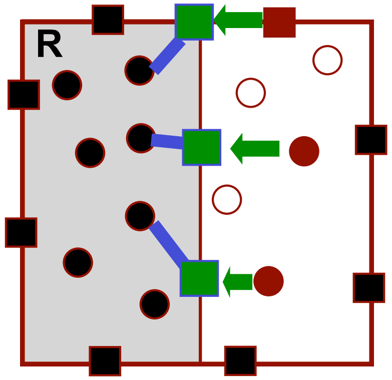
$f_{ab'}$

c 11 11 01
c' 11 11 10

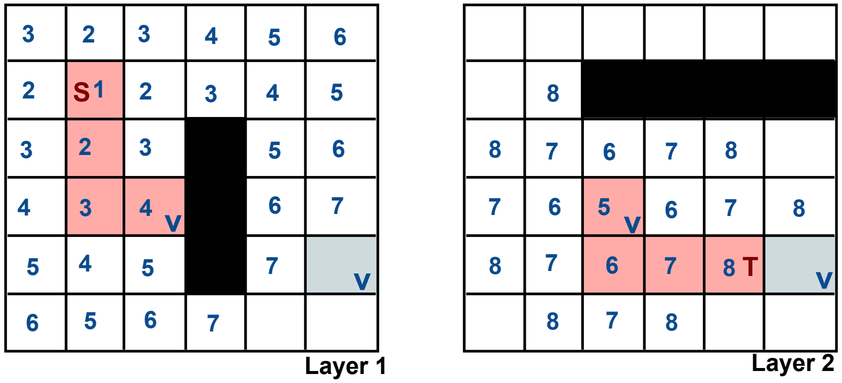
2. BDD-based Logic Network Repair



3. Quadratic Placement



4. Maze Routing



3	2	3	4	5	6
2	S 1	2	3	4	5
3	2	3	■	5	6
4	3	4 V	■	6	7
5	4	5	■	7	V
6	5	6	7		

Layer 1

	8	■	■	■	■
8	7	6	7	8	
7	6	5 V	6	7	8
8	7	6	7	8 T	V
	8	7	8		

Layer 2

About MOOC Software Projects

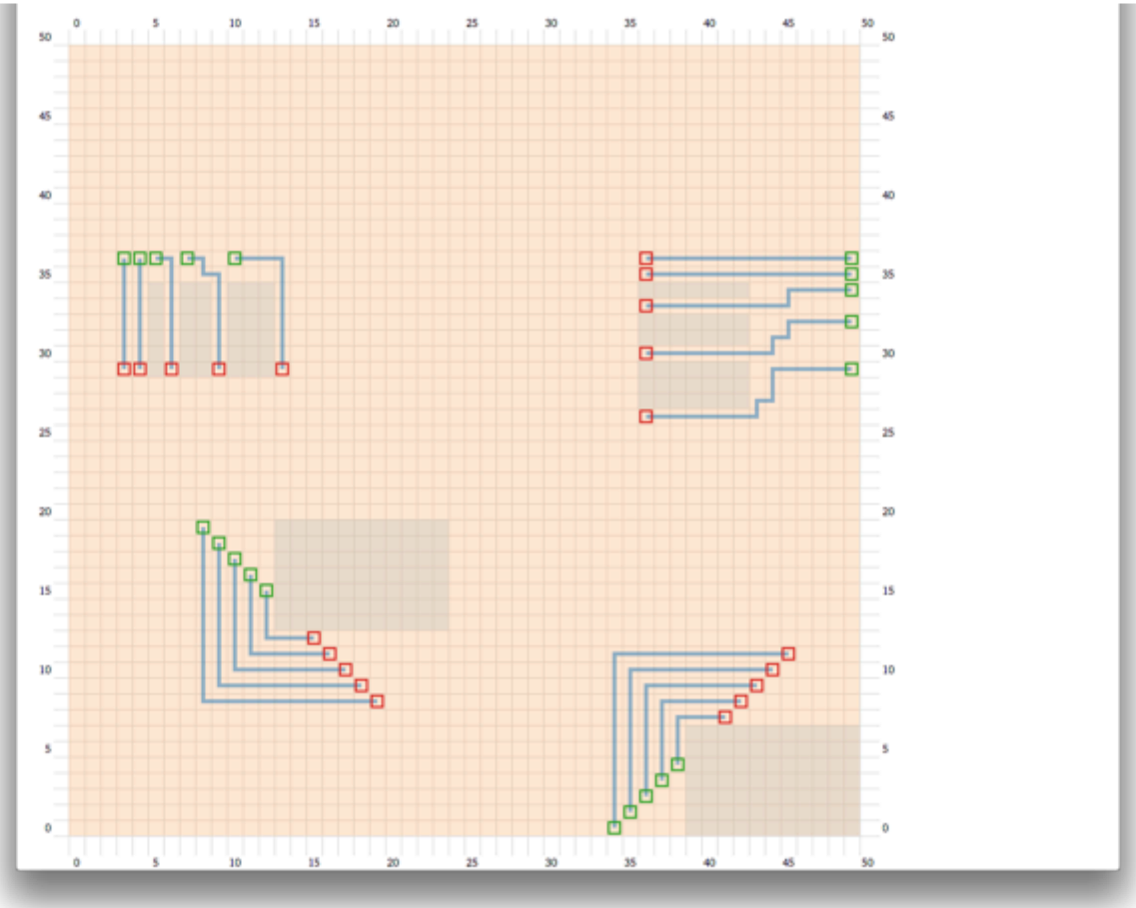
- **We provide the spec and input data; we grade output file**

- **Too hard to compile/run other people's code**
  - Too many platform issues; too many language issues
  - ASCII textfile in; ASCII textfile out

- **Architect like a real "regression test suite"**
  - Several benchmarks, from tiny to big
  - Partial credit for each part (as much as possible)
  - Lots of feedback in the webpage portal about how it went

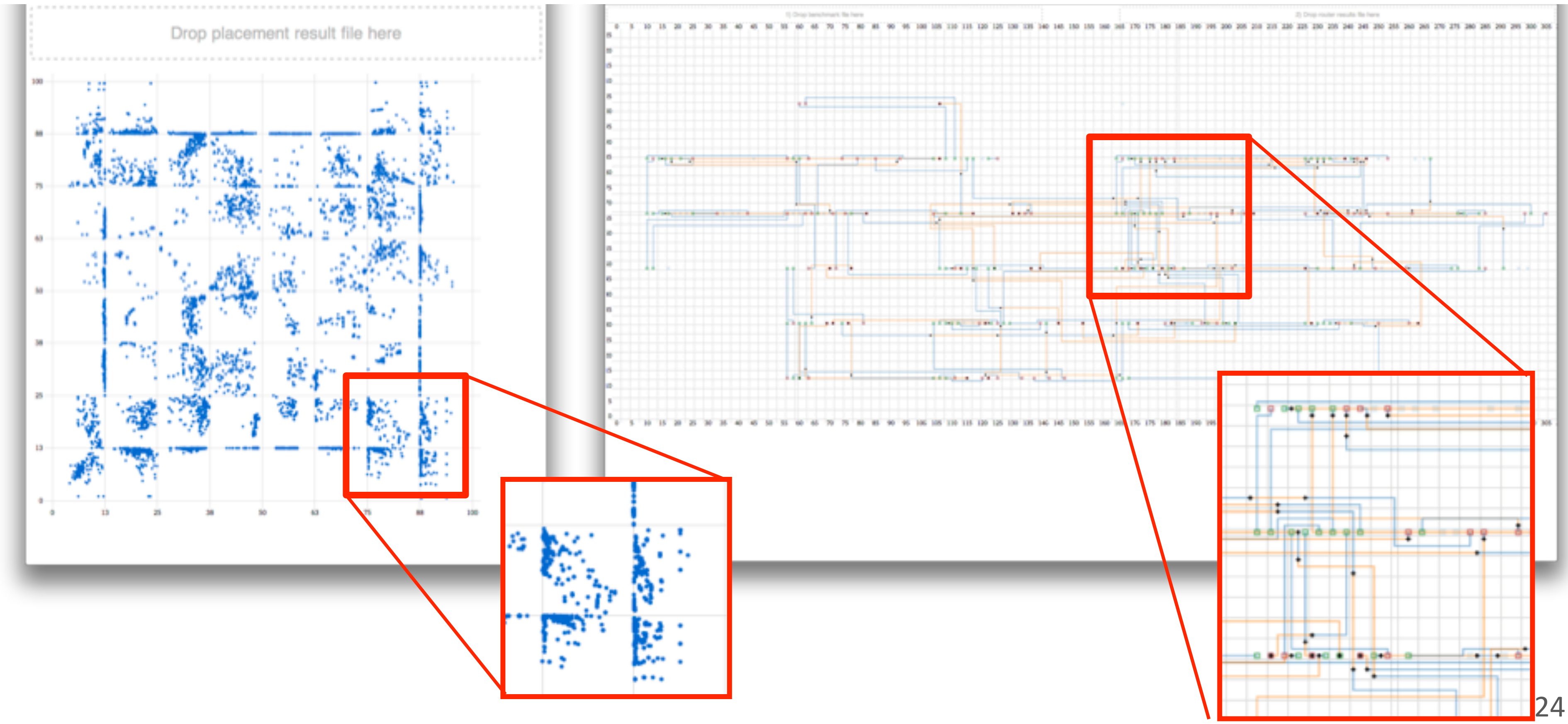
Concrete Example: Regression-Like Benchmarks

- Can you route a 2-point wire in 1 lqyer?? Horizontal? Vertical? Around obstacle? Straight line? With a bend? etc



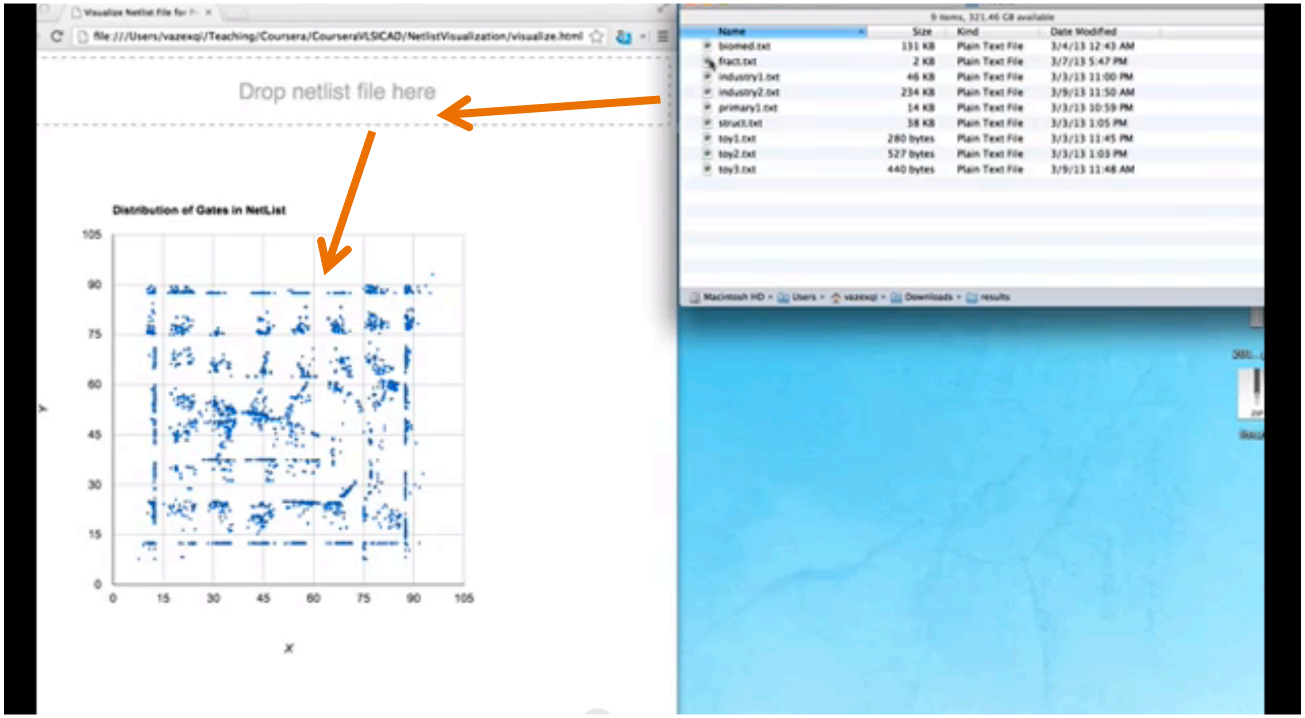
Software Project Examples: Layout

Recursive Quadratic Placer **2-Layer ASIC-style Maze Router**



New Problem: How Can Students *See* Layout?

- **Can't control what platform they use!**
- **Answer: custom HTML5 geometry web-based environment**
  - If they have a modern browser, they can **drag/drop text file**



Visualize Netlist File for P
file:///Users/vazexqi/Teaching/Coursera/CourseraVLSICAD/NetlistVisualization/visualize.html
Drop netlist file here
Distribution of Gates in NetList
105
90
75
60
45
30
15
0
y
0
15
30
45
60
75
90
105
x

9 items, 321.46 GB available

Name	Size	Kind	Date Modified
biomed.txt	131 KB	Plain Text File	3/4/13 12:43 AM
fract.txt	2 KB	Plain Text File	3/7/13 5:47 PM
industry1.txt	46 KB	Plain Text File	3/3/13 11:00 PM
industry2.txt	234 KB	Plain Text File	3/9/13 11:50 AM
primary1.txt	14 KB	Plain Text File	3/3/13 10:59 PM
struct.txt	38 KB	Plain Text File	3/3/13 1:05 PM
toy1.txt	280 bytes	Plain Text File	3/3/13 11:45 PM
toy2.txt	527 bytes	Plain Text File	3/3/13 1:03 PM
toy3.txt	440 bytes	Plain Text File	3/9/13 11:48 AM

(HTML5 design and pic by Nicholas Chen)

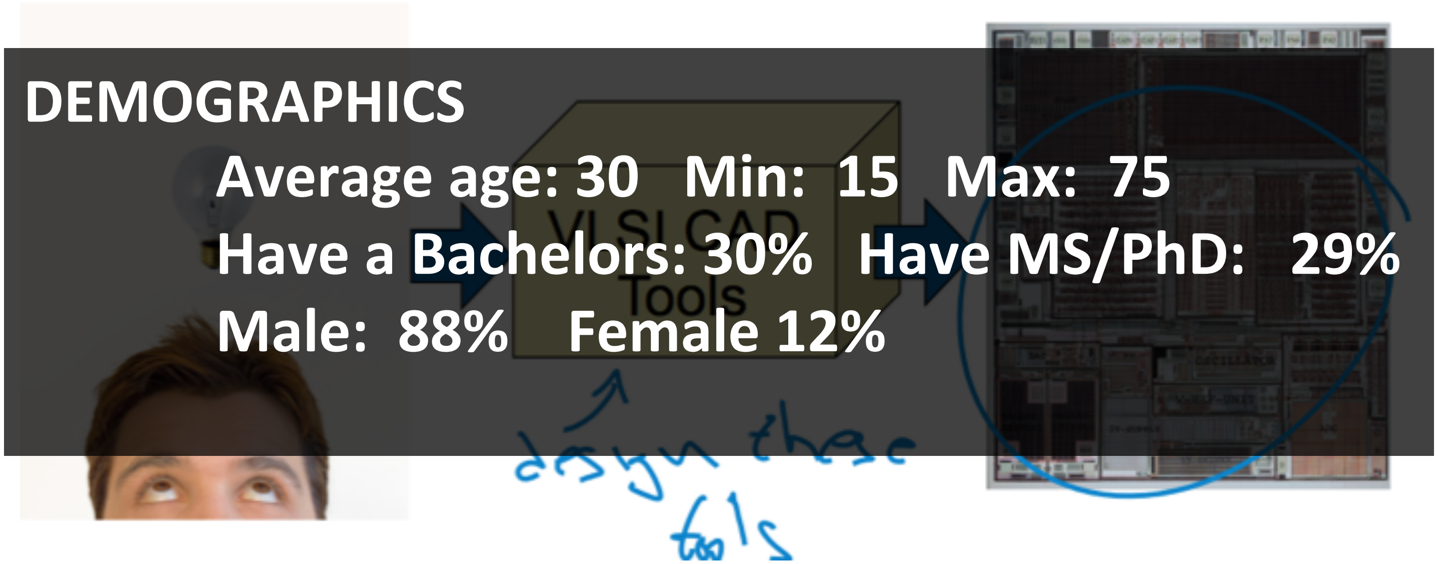
The Elephant In The (MOOC) Room…

LOTS of people start…

NOT so many people finish

VLSI CAD MOOC: The Participant Landscape

- **17,500** participants at peak
- **7,000** people watched a video
- **1,300** people did a homework
- **400** people tried a software assignment
- **500** people took the Final Exam
- **386** ***Statement of Accomplishment* Cert's**



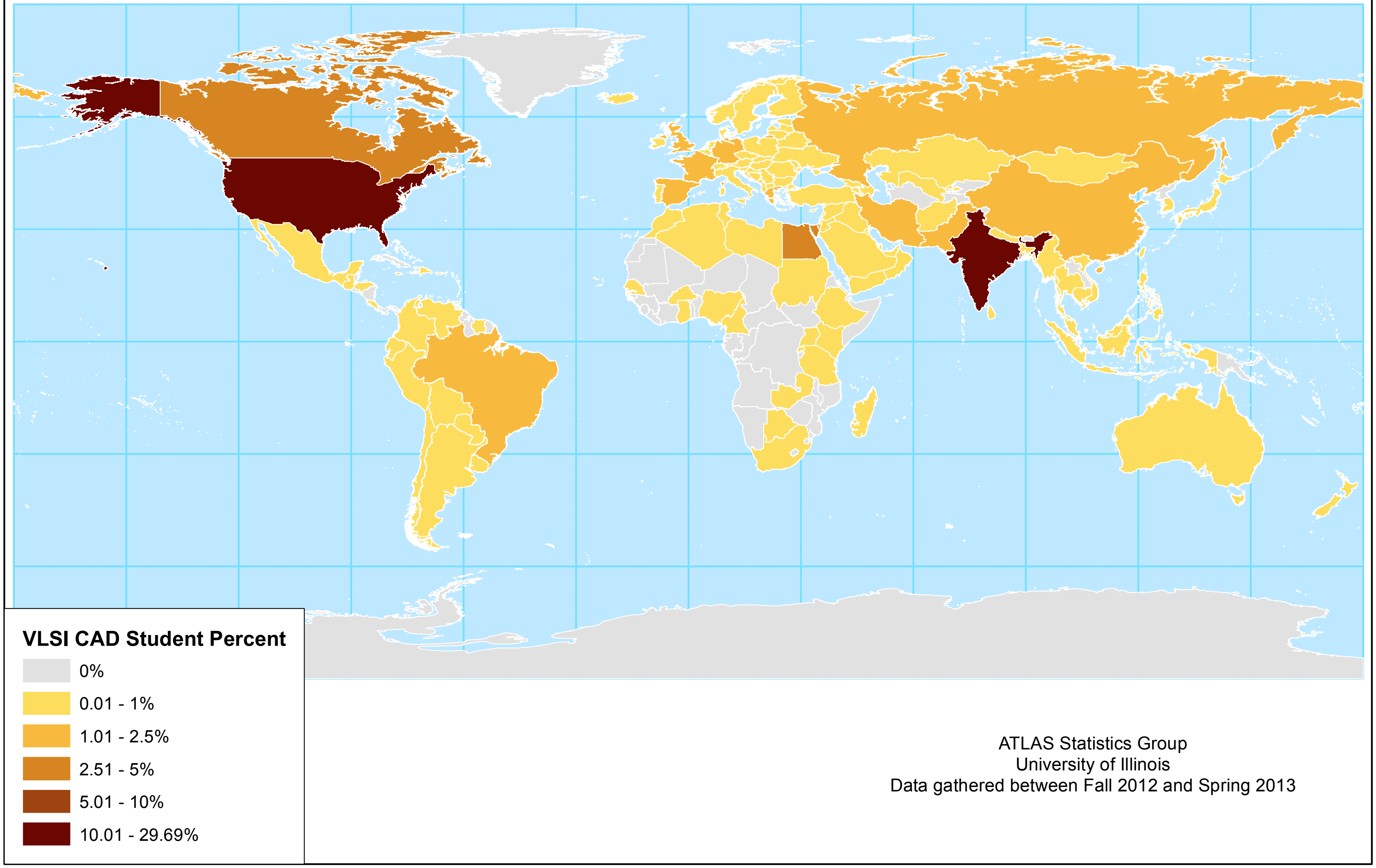
DEMOGRAPHICS

Average age: 30 Min: 15 Max: 75

Have a Bachelors: 30% Have MS/PhD: 29%

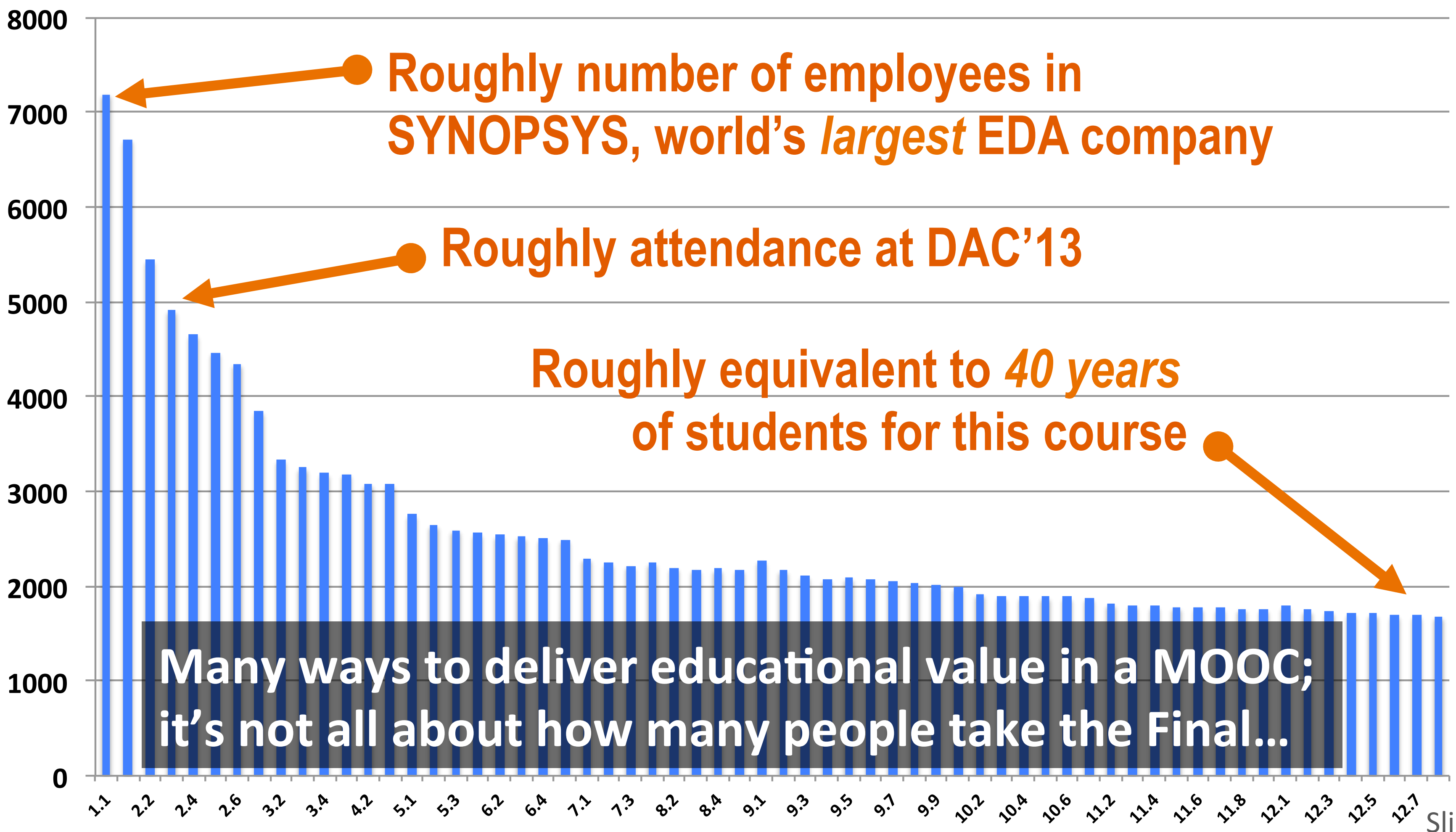
Male: 88% Female 12%

University of Illinois Coursera Students: VLSI CAD
Percent of Students by Country



ATLAS Statistics Group
University of Illinois
Data gathered between Fall 2012 and Spring 2013

Details: Views Across All My MOOC Videos



Roughly number of employees in SYNOPSYS, world's *largest* EDA company

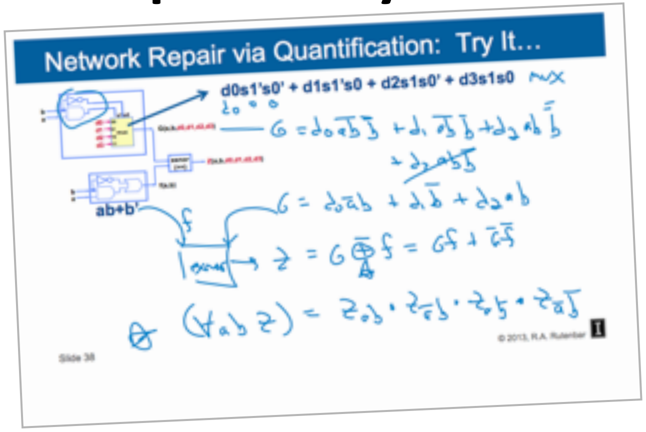
Roughly attendance at DAC'13

Roughly equivalent to *40 years* of students for this course

Many ways to deliver educational value in a MOOC; it's not all about how many people take the Final…

Things One Learns in MOOC-Land..

My handwriting *unreadable* for a planetary audience



Network Repair via Quantification: Try It…

d0s1's0' + d1s1's0 + d2s1s0' + d3s1s0 ~x

$d_0 \circ \circ$

$G = d_0 \bar{a}\bar{b}\bar{f} + d_1 \bar{a}\bar{b} f + d_2 a b \bar{f}$

$+ \cancel{d_3 a \bar{b} \bar{f}}$

$G = d_0 \bar{a} b + d_1 \bar{b} + d_2 a b$

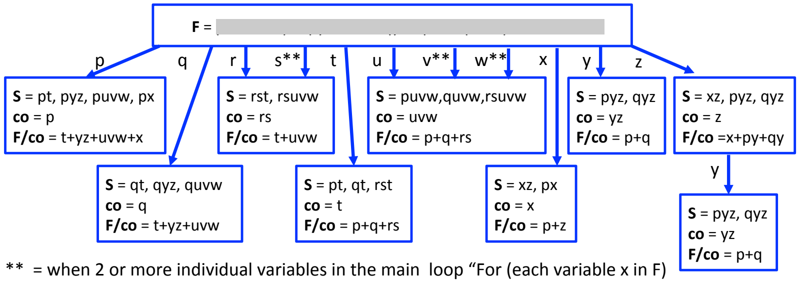
$Z = G \overline{\oplus} f = Gf + \bar{G}\bar{f}$

$\forall (\forall a b \, Z) = Z_{ab} \cdot Z_{\bar{a}b} \cdot Z_{a\bar{b}} \cdot Z_{\bar{a}\bar{b}}$

Slide 38

© 2013, R.A. Rutenbar

Creativity to make homeworks in multiple-choice format



F = [redacted]

S = pt, pyz, puvw, px
co = p
F/co = t+yz+uvw+x

S = qt, qyz, quvw
co = q
F/co = t+yz+uvw

S = rst, rsuvw
co = rs
F/co = t+uvw

S = pt, qt, rst
co = t
F/co = p+q+rs

S = puvw,quvw,rsuvw
co = uvw
F/co = p+q+rs

S = xz, px
co = x
F/co = p+z

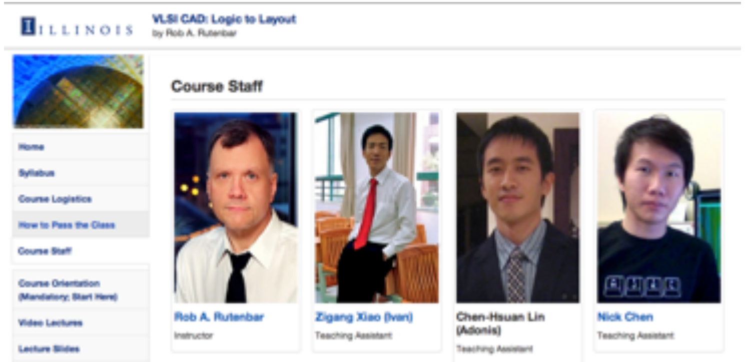
S = pyz, qyz
co = yz
F/co = p+q

S = xz, pyz, qyz
co = z
F/co =x+py+qy

S = pyz, qyz
co = yz
F/co = p+q

** = when 2 or more individual variables in the main loop "For (each variable x in F) yield the same solution for F/x, we just draw them pointing to the same child box in our diagram of the progress of the algorithm.

MOOCers crave *interaction* with instructors, 24x7!



ILLINOIS

VLSI CAD: Logic to Layout
by Rob A. Rutenbar

Course Staff

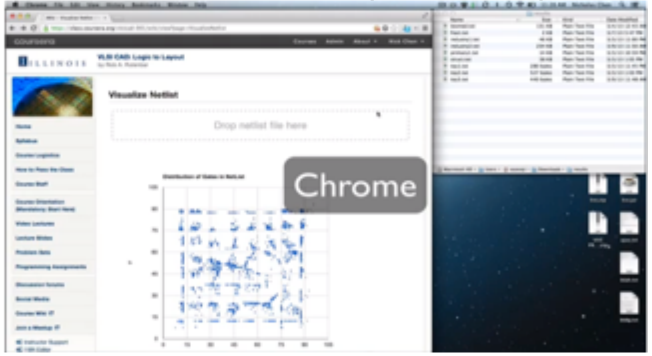
Rob A. Rutenbar
Instructor

Zigang Xiao (Ivan)
Teaching Assistant

Chen-Hsuan Lin (Adonis)
Teaching Assistant

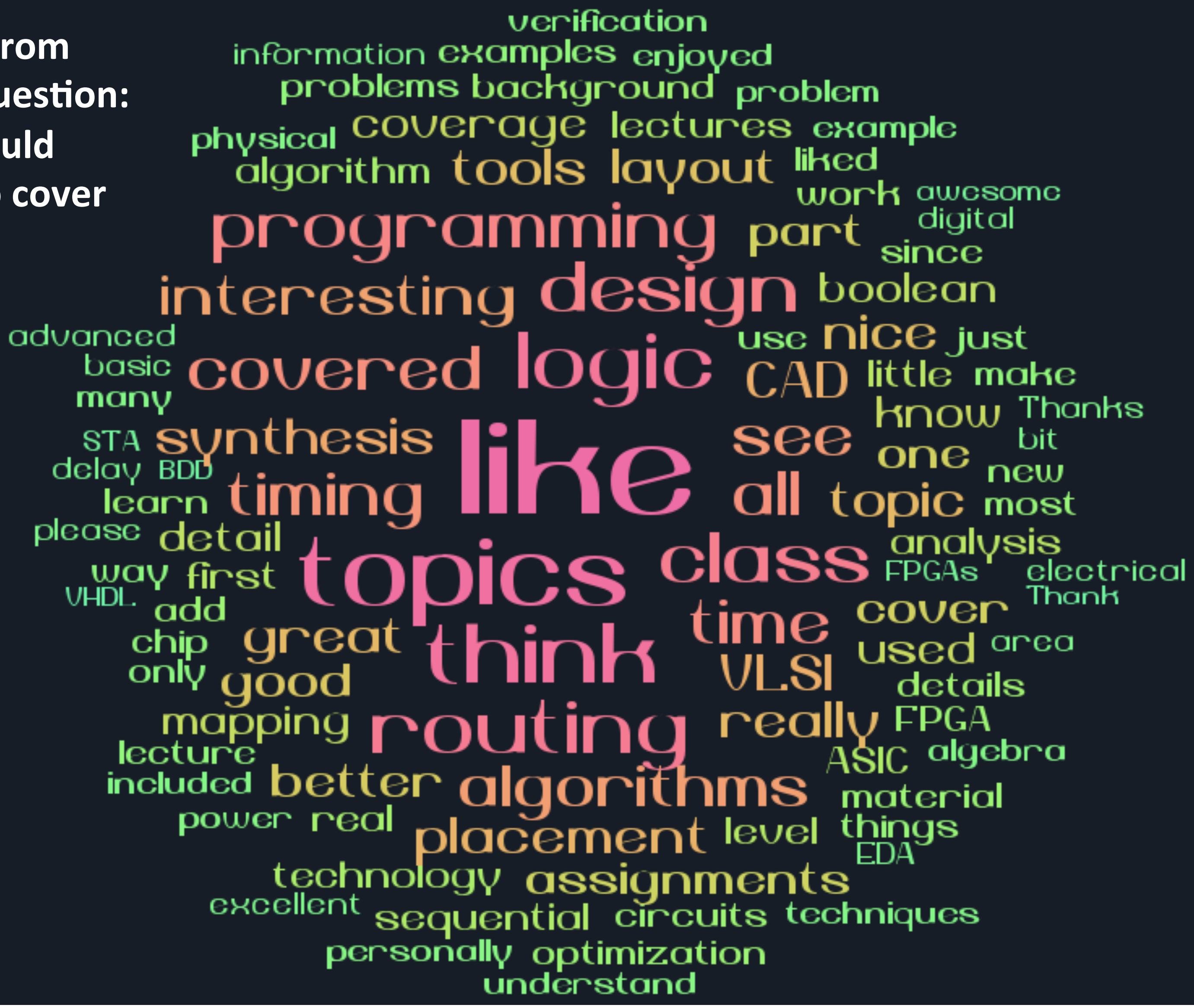
Nick Chen
Teaching Assistant

Be sensitive to diversity of IT resources of participants



My MOOCers: Want *More* EDA…

Word-cloud from
Final Exam question:
what *else* would
you like us to cover



Word It Out

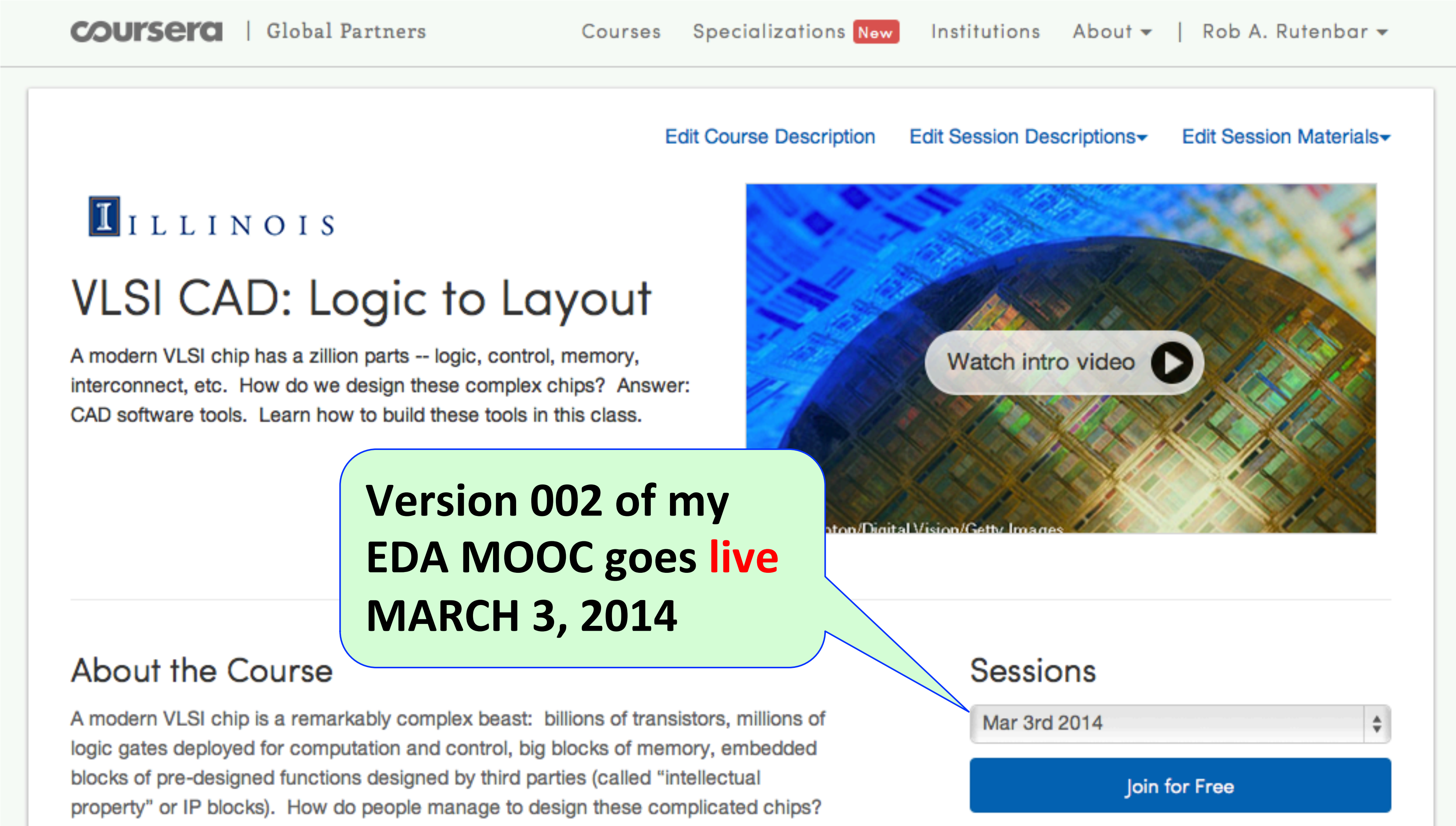
Why Did I *Do* This…?

Because every vibrant discipline needs a solid on-ramp, and teaching of core-EDA was vanishing, rapidly…

Reflections…

- **If we want to energize a new generation of EDA**
  - Moving EDA into **other** areas or kinds of systems
  - Moving EDA into **new** technology platforms (eg, post-Moore)
  - Translating EDA **"sideways"** into adjacent opportunity areas
- **If we want to do any of these things, somebody has to be teaching the foundational topics**
  - Maybe it is the case that planet can only support a **handful** of these large, global-scale courses on EDA foundations
  - Maybe this is only way to reach broadest, global audience
  - OK – so be it. **Here's my shot at "regenerating the excitement"**

Me: Personally Positive on MOOC Potential



ILLINOIS

VLSI CAD: Logic to Layout

A modern VLSI chip has a zillion parts -- logic, control, memory, interconnect, etc. How do we design these complex chips? Answer: CAD software tools. Learn how to build these tools in this class.

About the Course

A modern VLSI chip is a remarkably complex beast: billions of transistors, millions of logic gates deployed for computation and control, big blocks of memory, embedded blocks of pre-designed functions designed by third parties (called "intellectual property" or IP blocks). How do people manage to design these complicated chips?

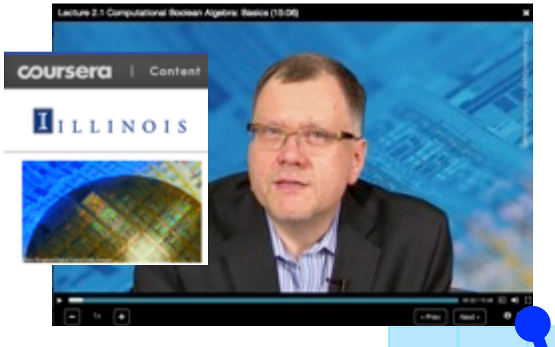
Sessions

Mar 3rd 2014

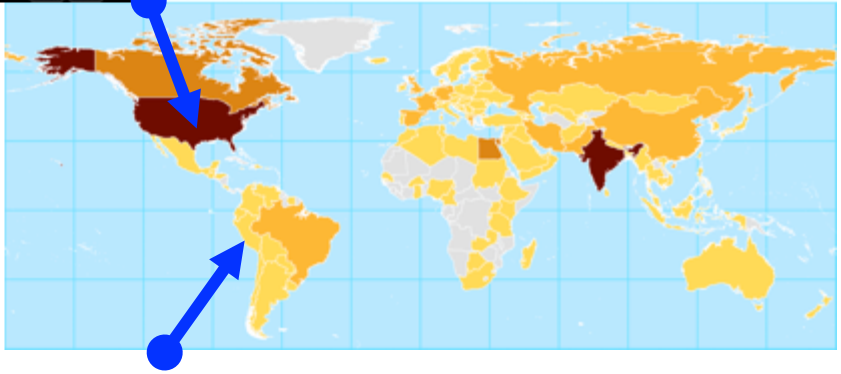
Version 002 of my EDA MOOC goes live MARCH 3, 2014

Me: Personally Positive on MOOC Potential

Me, March-May 2013



Lecture 2.1 Computational Boolean Algebra: Basics (15:06)



Me, Aug 2013

INABIF, Villa El Salvadore, Peru

Summary

- **I taught the first-ever EDA MOOC last year**
  - Exhilarating. Amazing. (Terrifying). Satisfying.
  - **Why? Because *somebody* has teach this stuff *at scale*.**
  - **Why? Because the material is *important* & *beautiful*.**
  - **Why? Because *need* excitement for vibrancy of discipline.**

- **I'm doing it again this year (in about 2 weeks)**
  - **Ask me at DAC how Round #2 goes….**