The First EDA MOOC: Teaching Design Automation to Planet Earth

Rob A. Rutenbar
Department of Computer Science, University of Illinois at Urbana-Champaign
Urbana, Illinois 61801 USA
rutenbar@illinois.edu

ABSTRACT
Massive Open Online Courses (MOOCs) can deliver advanced course material at planetary scale, combining internet-based video content delivery, and cloud-based assignments. From March to May 2013, I taught the world's first EDA MOOC, entitled VLSI CAD: Logic to Layout, based on roughly 20 years of experience teaching electronic design automation in a conventional face-to-face classroom setting. Over 17,000 participants registered for this MOOC. This paper summarizes my experience with teaching EDA at planetary scale: how we covered ASIC synthesis, verification, layout, and timing; how we built cloud resources to enable students to experiment with open-source tools; how we designed software projects and deployed cloud-based auto-graders to support realistic EDA tool projects. The paper also discusses what MOOCs could mean to the dynamism of the EDA community.

Categories and Subject Descriptors
B.7.2 [Integrated Circuits]: Design Aids
K.3.1 [Computer and Education]: Distance Learning

General Terms
Education, EDA

Keywords
VLSI, EDA, CAD, ASIC, MOOC, Logic, Layout

1. INTRODUCTION
Massive Open Online Courses (MOOCs) have made a significant splash in the academic world, since their debut in large experiments done in some computer science classes offered in late 2011 [1]. The basic organizing principles of a MOOC include: course design for massive scalability, to thousands, or even to 10,000 or 100,000 users; broad accessibility (i.e., “open” access); use of internet video technologies for content delivery; use of cloud technologies for large-scale assignments and computer-based assessment of educational progress.

The technology is not without controversy, of course. For example, there are robust arguments about “credentialing,” that is, whether these can be courses in the traditional sense of bearing college-level credit, whether we can create college degree programs from such courses, and what such courses and degrees might mean for typical university business models. In the few years since these early experiments, a dynamic ecosystem of MOOC platform providers (e.g., Coursera [2], Udacity [3], edX [4] among others), and a large number of new MOOC courses, have come into existence.

From March to May 2013, I taught the first-ever MOOC focused on Electronic Design Automation (EDA), to an audience that, at its peak, topped 17,000 registered participants. Entitled VLSI CAD: Logic to Layout [5], the MOOC was delivered on the Coursera MOOC platform [2], and focused on a classical Application Specific Integrated Circuit (ASIC) design flow, covering logic synthesis and formal verification, technology mapping, placement and routing, and both logical and electrical timing analysis. This paper is my attempt to describe the experience—what it is like to teach EDA at planetary scale.

First, let me justify the course name. “CAD” encompasses many forms of computer aid across many disciplines: mechanical, architectural, etc. So, why choose this older form of the label for our discipline, and not “EDA”? The answer speaks to one of the most important reasons I chose to do this MOOC: We want to attract new people into the field. “CAD” is easier to understand by persons not associated with our discipline. And “VLSI” remains in wide currency, especially among students.

The MOOC was based on over 20 years teaching traditional face-to-face classroom versions of this material, at Carnegie Mellon University. I estimate that I taught roughly 700 students this logic-to-layout course, during those years. The course exists as a large set of lecture notes—~1000 lecture slides—and numerous homework assignments and software projects. The challenge was to re-architect this material for a global audience with whom I would interact only remotely, yet do so in a manner that still conveys the elegance and power of these ideas, that underlie so much of modern IC design.

The remainder of this paper is organized as follows. Section 2. describes the development of this MOOC — what topics were included, and how these and a rigorous set of student assignments were transformed for this new delivery platform. Section 3. describes experiences from the first deployment of this material, in early 2013. Section 4. gives a wide range of statistics from this first offering of the MOOC: participants, participation rates, demographics, etc. Finally, I offer some concluding thoughts on what MOOC technology might mean more generally for the EDA community, in Section 5.

2. MOOC DESIGN
MOOC technology frees us from many of the constraints of a conventional university course. A MOOC need not adhere to a fixed academic calendar, and can be of any length. MOOCs on the Coursera platform are semi-synchronous, which means that they have defined start and end dates; participants can watch lecture video content at their convenience, but all assignments
have hard deadlines. The idea is that shared deadlines (1) encourage participants to progress through the class and (2) give the overall cohort of participants a sense of shared community, since everyone is working on similar issues at similar times. The key is to design compelling lecture content, and pair it with interesting and challenging assignments that can be graded at scale. We discuss both of these design problems in this section.

2.1 Lecture Content

Based on early evidence that shorter MOOCs were more popular and retained more participants (see [15] for a subsequent study), I chose a 10 week duration for VLSI CAD: Logic to Layout. The course featured 8 weeks of active instruction, and 2 additional weeks to allow for homeworks and projects to complete, and for a final exam.

My first task was to decide what topics to select from my conventional semester-long course, which runs 15-16 weeks. That course covers a robust subset of a full ASIC synthesis flow: computational Boolean algebra; logic synthesis (2-level and multi-level); formal verification (e.g., BDDs and SAT techniques); technology mapping; standard-cell ASIC layout style; partitioning; placement; routing; computational geometry for DRC/extraction. I usually cover a handful of essential algorithms for each of these topics. For a shorter 8-week version, I resolved that the core of the MOOC must still be “logic to layout,” but that I would reduce overall breadth and cover a more limited set.

MOOCs can be designed in many different styles, but a consistent element is use of many short video segments, which are each focused on one (or a few) critical concepts. There is evidence from those who study pedagogy that this works better than, for example, simply filling the totality of a standard 60 or 90 minute lecture [15]. But equally important is the infrastructure reality that in many parts of the planet, one cannot download a 60 minute video with any reliability. Thus, partitioning a lecture into many small parts, each independently downloaded, turns out to be an essential accommodation if one wishes to engage a planetary audience.

As a mechanical aid in this process, I built a “concept map” of my traditional class. I walked through each lecture slide in my full class—a total of 948 slides, to be exact—and partitioned these into unique concepts, with slide-counts. The motivation was two-fold: decide what topics to keep; decide what depth of coverage for each topic (with slide count as the metric). I ended up with 102 unique EDA concepts in the full version of the course. Figure 1 shows a snapshot of this analysis for topics related to BDDs.

![Figure 1. EDA concept map snapshot from traditional classroom version of VLSI CAD: Logic to Layout. Snapshot focuses on BDD topics.](image)

After developing the concept-map partition, I selected 8 core topics for the 8 weeks of class:

1. Computational Boolean Algebra (Shannon factors, quantification, unate recursive paradigm, etc.)
2. Formal Logic Verification: BDDs and SAT
3. Logic Synthesis I (2-level, introduction to multi-level)
4. Logic Synthesis II (algebraic model, factoring, don’t cares)
5. Technology Mapping (recursive tree covering)
6. Placement (annealing and recursive quadratic)
7. Routing (multi-layer maze)
8. Timing (logic-level static timing, Elmore wire timing)

These topics represent a good compromise between completeness and depth: there is a critical model, representation, or algorithm for every step from front-end to back-end EDA, for both synthesis and verification, for both logic and layout.

Of course some topics of great value had to be omitted, in the interests of fitting the schedule. There is solid coverage for logic, but not elemental topics. There is one, or at most two algorithms for some key step, and not several. There is no test, nor simulation, nor deep electricity-related content. This is not to minimize any of these vital topics for EDA, but simply an acknowledgement of time constraints.

The result of this effort was that my original 948 lecture slides were transformed into 615 new slides, arranged as 69 short, focused lectures. Figure 2 shows how these lectures became the video content for my MOOC. The result was 69 videos, with average length 15 minutes, comprising 17 total lecture hours. It is useful to note that my MOOC exhibited a common characteristic: enhanced lecture “efficiency”. The MOOC comprises roughly 50-60% of my traditional class material, but does so in just 1/3 of the time. One simply avoids many of the in-class distractions, and need to repeat material, in an environment where viewers can simply rewind and replay any topic, any time, anywhere.

![Figure 2. Week-by-week video lecture content of VLSI CAD: Logic to Layout in MOOC form. 69 total lecture videos, average length 15 minutes, comprising 17 total hours of video instruction.](image)
lecture. PDF versions of the completed lectures were available for students on the Coursera site, along with the videos. In addition, each video begin with a “talking head” interview with me, in which I talk about what the new material will cover, how it relates to the previous lecture, and how it fits with the overall arc of the course. Needless to say, building the lecture content for a MOOC is vastly more complex, and more time-consuming, than a traditional course. Figure 3 shows example frames from MOOC lectures, showing the personal introduction, and the style of write-on-slide development.

![Image](image_url)

Figure 3. Example frames from VLSI CAD: Logic to Layout video lecture content. (a) Introductory narration by the instructor. (b) Technical lecture slide with detailed mathematical derivation done as “live” writing.

### 2.2 Assignment Content

Perhaps the most significant difference in the design of a MOOC is the need to architect all grading and assessment activities so that they can be delivered at great scale. With 10,000 or more participants, one simply cannot employ human teaching assistants and ask them to grade anything, by hand. And some forms of assessment are yet beyond reliable mechanical means (i.e., software running in the cloud) to grade – notably detailed mathematical derivations. One can use sophisticated peer assessment methods here, in which the class itself is given a rubric, asked to grade others’ assignments, and then results are averaged-out in sensible statistical ways. I chose not to deploy any peer schemes. Our assignments are entirely graded in the cloud.

Assignments came in three forms: weekly homeworks (8 in all); biweekly software design projects (4 in all); a final exam (essentially a larger homework). The course actually allowed two paths to completion, an Accomplishment path that only required completion of homeworks and final exam; and a Mastery path that in addition required completion of the software projects.

They keys to designing effective MOOC homework problems are two-fold: intelligent partitioning, and aggressive randomization. One must decompose a large problem into a set of smaller, separated parts, each of which can support true/false, multiple choice, type-a-number, type-a-label sorts of questions. To combat cheating, and ensure that every participant gets a moderately “individualized” version of each assignment, one must over-supply problems and over-supply solutions to these problems. The Coursera platform supports a rich ecosystem that allows the MOOC to randomize each assignment at delivery time.

My initial skepticism was that a class based primarily on multiple-choice questions could never be rigorous. However, after this MOOC experience, I now believe that with sufficient hard work, and intelligent partitioning/randomization, one can build rich problems that challenge students to think, and to demonstrate their knowledge of the material. However, just as with the experience of building the video lecture content: this is significantly harder than building a standard set of homeworks.

Another challenge to be overcome was to devise ways to allow students to run real EDA tools. Although this was a class about building tools, I strongly believe that participants need to try other tools to see what they do. In addition, such experiments make excellent homework problems: give participants design problems, ask them to formulate a solution and run it using a real tool, then ask “What happened?” as a homework question. Again, dealing with this at scale proves to be the challenge.

Recall: this MOOC is open. So, arbitrary persons from around the planet can participate in the class, which is hosted on a commercial cloud provider (Amazon™ EC2 Elastic Cloud [6]). We are unable to offer commercial tools from any of the large, established vendors. Our solution was a “Back to the Future” strategy: we obtained copies of early foundational open-source codes from the 1980s and 1990s that harkened back to the dawn of the modern EDA flow, and deployed them (alongside some more modern codes) in the cloud. Figure 4 shows the overall architecture.

![Diagram](diagram_url)

Figure 4. Software architecture for EDA tools access, and auto-grading of software projects, for VLSI CAD: Logic to Layout.

We deployed the following tools in cloud-form for the MOOC:

- **kbdd** [7]: A software Boolean calculator with scripting language based on BDDs, from Carnegie Mellon.
- **miniSAT** [8]: A widely used engine for Boolean Satisfiability (SAT) solving, from Switzerland.
- **Espresso** [9,10]: First robust 2-level Boolean optimizer, originally developed for PLAs, from Berkeley.
- **Ax=b**: A simple custom solver for linear systems, to allow homeworks related to quadratic placement.

The architecture for all of these deployments is as a web-portal that consumes, and produces, ASCII text. So, to access these tools, participants upload a file with appropriate inputs to the web-portal for the tool. The tools are run in the cloud (with appropriate attention paid to terminating run-away tools), and the ASCII text output is delivered back to a private webpage, viewable by only that course participant. Older outputs are also available, simply by scrolling.

Our software projects use the same portal architecture, but the software running in the cloud is grading student results from their tool project. The major architectural decision here is not to try to execute code from class participants in the cloud – it is too difficult to compile/link/run code written in arbitrary languages, from arbitrary platforms, reliably. Also, there are some serious safety/security issues with allowing arbitrary code languages and platforms.
Instead, we specify what we want the tool to do, provide textfile inputs, and provide auto-graders that read textfile output files and score them. Course participants are free to use any language, any library, any platform, they choose.

We designed four challenging software projects for the MOOC, representing two logic and two layout tasks, respectively (see Figure 5). These were:

1. **Boolean Data Structures & Computation**: Using the Unate Recursive Paradigm (URP) and positional cube notation (PCN), build software to manipulate Boolean functions as data structures.
2. **BDD-Based Formal Network Repair**: Using the kbdd scripting format, formulate and solve a logic-repair problem using formal methods one can implement via BDDs.
3. **Quadratic Placement**: Using a quadratic wirelength formulation, and a recursive bipartition legalization strategy (as in [13]), do a global placement on industrial netlists.
4. **Maze Routing**: From pin/obstacle maps from reference placements of these same netlists, perform a 2-layer routing, with preferred layer directions, vias, etc.

As with the lectures, the projects represent a compromise between completeness and schedule. There are software projects involving both logic and layout, and both verification and synthesis.

One point worth emphasis is, just like with homeworks, one must decompose projects into “gradable-units”, so that (1) benchmarks can test individual aspects of the submitted solutions, and (2) partial credit is feasible to assign. Roughly speaking, this is exactly like building a large regression suite for a commercial EDA tool. One must ascertain that every performance-aspect of the tool is working correctly. This means: not only large benchmarks with coarse metrics, but also, many small unit tests.

Figure 6 shows one set of unit tests for the router project, to test whether a submitted solution can route: short wires in one layer; short vertical segments; short horizontal segments; wires with a few bends; wires around obstacles; etc.

A final interesting challenge to mention is how to provide graphical display for the layout-oriented projects. We cannot control what platform our participants have available. And, it is just impossible to build layout tools if one cannot see the layout results. We solved this problem by building a sophisticated graphics environment entirely in the browser, using modern HTML5 constructs. One can open a portal page on our MOOC, and then drag-and-drop a textfile representing the layout result from the client side; the full layout is rendered in the browser, and several powerful editing commands become available.

Figure 6 was actually drawn using our HTML5 browser graphics tools [16].

Figure 5. Software design projects for VLSI CAD: Logic to Layout.

3. **MOOC DEPLOYMENT**

The MOOC ran over ten weeks from March to May of 2013. Detailed data about demographics appears in Section 4. Here, I describe briefly some of our experiences in running the course.

The first observation is how enthusiastic our participants were. They were excited that this material was being offered. Some were happy to see any course on EDA, since this was relevant to their jobs. Some were excited because they liked VLSI topics. Some were excited because it was a “cool” application at the intersection of hardware and software.

The second observation is that many of our participants came without any preparation, and fell away quickly. I was clear in the overview materials about what was required to approach this class: basic digital design (combinational and sequential); undergraduate-level mathematical preparation (calculus, linear algebra); solid programming skills (in a modern language, with data structures). Nevertheless, for many students, their enthusiasm was abundant, but not their preparation.

There were several surprises. For example, some things that work “in the small” do not work “in the large” and one of these surprising things was my own hand-writing. Readable in-class, in-person, in a room in the USA; not so at planetary scale. In later lecture content, I animated some complex equations rather than writing them, to spare my viewers this problem. Another surprise was the range of languages in which our software projects were implemented. I was expecting JAVA, C, and C++, and likely Python. I was not expecting ML, or Haskell, or Ruby, or Racket. People who like to code often like to try new things in new languages they are learning for other purposes.

We were also impressed with the quality of the software projects submitted, and with the eagerness to try bigger designs. A particular surprise was how quickly a subset of the most sophisticated participants would arrive at working code and demand “more benchmarks”. On the fly, we added Extra Credit assignments, which featured bigger, more complex design scenarios. For the layout benchmarks, which were based on the classical MCNC benchmark suite [14], this meant bigger netlists. Figure 7 shows solutions to larger placement and routing benchmarks, to give some sense of the scale of these designs. By modern standards, these are, of course, tiny. But as class projects, they succeeded admirably in challenging our students to deliver working tools that pass a strict set of unit tests.

One final observation is how much the participants crave interaction with course staff. Despite being remote, they used the embedded Coursera Forums to interact with each other, and especially with the instructional staff. Tending these forums was a significant effort, which my teaching assistants mainly handled, with admirable speed and agility.
4. MOOC DATA

The first questions asked about all courses intended for “massive” scale are numerical: How many people registered? How many completed? The next questions are demographic: Who were they? Where were they? This section attempts to answer these and related questions about the experience of teaching VLSI CAD: Logic to Layout for the first time.

A particular point of controversy for all MOOCs is the so-called “funnel” effect: many enter, few complete. This MOOC was no exception. We note, to start, that given the open access of these offerings (i.e., no cost) and relatively frictionless signup (i.e., click on a link) it is common for many to register, fewer to show up to the class, and yet fewer to engage with the material. Thus, it is common to measure a variety of metrics for “engagement” to try to understand the funnel. Figure 8 shows the data.

The funnel is consistent with others’ experiences with technical MOOCs. About 1/2 of the registered participants never show up to the class at all. Around 1/5 of those who watched a video tried a homework. About 1/4 of those tried a software assignment; about 40% of these did the final exam.

While one would obviously like to be able to claim to have pulled 10,000 or more into the EDA universe, the numbers are nevertheless encouraging, and worth a closer look. Figure 9 shows the per-lecture viewers for each of the 69 videos. We can make a few observations:

- About 7000 people found it worthwhile to watch the course introduction videos—a number roughly equivalent the number of employees in the world’s largest EDA vendors.
- After a few weeks, the MOOC was still attracting as many viewers—5000—as the attendance at the 2013 50th Anniversary Design Automation Conference (DAC).
- Almost 2000 people watched all the videos – a number equivalent to my teaching a typical on-campus version of this class for another 40 years.

We have other demographic data about age and gender:
- Have a Bachelor’s Degree: 30%. Have MS/PhD: 29%.
- Male: 88%. Female: 12%.

The older demographic seems consistent with the fact that the course is rather advanced, and its on-campus version is taught as a Senior elective or first-year Graduate class. The gender split is, unfortunately, consistent with broad numbers for enrollment in Computer Science programs in the US.

Figure 10 gives a geographical view, showing participation by country. The MOOC had participation worldwide, and on (almost) every continent. Unsurprisingly, the US and India had the largest concentration, but there was also participation in South America (notably Brazil) and Africa (notably Egypt). For Africa, we note that ready availability of broadband (for video access) is still more prevalent near the coasts than the interior of the continent; for China, we note that in 2013, there were access issues for Chinese MOOCs, but there are now mirrors intended to resolve this situation [17].

One final survey taken as part of the course asked participants what topics not covered in the MOOC, or not covered sufficiently for their taste, they would ask us to do more of, in a subsequent (or extended, upgraded) offering of this course on EDA topics. Figure 11 shows a word cloud obtained from mining all of these survey responses.
There is a quite broad range of topics, from logic to layout, from synthesis to verification, as well as a number of positive words of affirmation in this sample.

5. CONCLUSIONS

From March to May 2013, I taught the world's first EDA MOOC, entitled VLSI CAD: Logic to Layout. The course attracted significant participation, which I can summarize in several different ways. In round numbers: 7000 people watched a lecture video; 2000 people watched all the lecture videos; 1000 people submitted a homework; 400-500 did software or the final exam. I believe the MOOC has added to the planet between 500 and 2000 persons with a serious level of EDA-competency. I think this qualifies as a success for this MOOC.

I also believe that MOOC technology offers some fresh opportunities for our community and our discipline. As some aspects of the discipline commodify, and others spin out widely into challenging new areas (e.g., nano, new forms of heterogeneous systems, etc.), it is imperative that somebody teaches the foundations. It is my experience that teaching of core EDA topics (and this MOOC is just one component of such a core) is on the wane. Yet, every vibrant discipline needs an "on ramp," a path for newcomers to understand who we are, where we came from, how we think, what problems we know how to solve.

Perhaps it is the case, instead of many offerings of exciting EDA classes at many universities, a more practical model is a few offerings of exciting EDA classes at planetary scale. Toward this end, I designed VLSI CAD: Logic to Layout as an experiment to see what could be done to draw some fresh talent into the community. I hope I succeeded. For the future health and dynamism of the discipline, I think design automation could use a thoughtful set of new MOOCS, to continue this experiment.

ACKNOWLEDGEMENTS

I owe an enormous debt of gratitude to my teaching staff, without whom this MOOC would not exist. Illinois Ph.D. students Nick Chen (cloud), Chen-Hsuan ("Adonis") Lin (logic) and Zigang ("Ivan") Xiao (layout) put in countless hours to make this MOOC a reality. I am also grateful to Deanna Raineri, Jason Mock, Liam Moran and Colleen Cook of the Illinois ATLAS organization for logistical, pedagogical, and media support for this effort. I also wish to thank Randy Bryant of CMU for his generous permission to use his kbd1 code. (Ed Clarke of CMU was similarly generous with his offer of early BDD software.) I would like to thank Daphne Koller of Coursera for her expeditious facilitation of the University of Illinois at Urbana-Champaign’s entry into the MOOC world. Finally, I wish to thank the several thousand various participants of this MOOC for their enthusiasm as we developed VLSI CAD: Logic to Layout in front of them.

REFERENCES