

A wide-angle photograph of the Stanford University Main Quad. The central focus is the redwood building, a large, ornate structure with a prominent stained-glass window and a gabled roof. The building is surrounded by a paved plaza with circular landscaped areas containing palm trees and other greenery. The sky is clear and blue.

# EDA in the Age of Machine Learning Teaching a New Elective Course at Stanford

Patrick Groeneveld, PhD

Cadence Design Systems

[prgr@stanford.edu](mailto:prgr@stanford.edu), [prgr@cadence.com](mailto:prgr@cadence.com)

EDPS 2018 Milpitas

Copyright ©2018 by Patrick Groeneveld

# Stanford EE292A: “Electronic Design Automation (EDA) and Machine Learning Hardware”

---

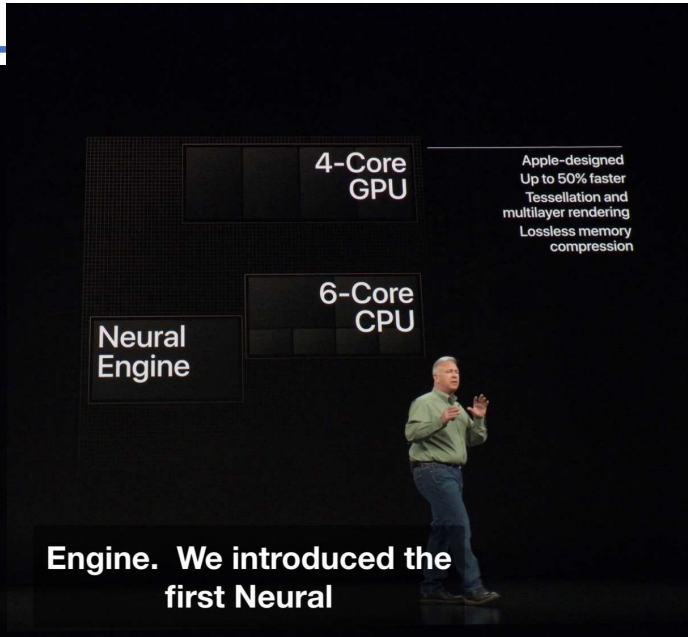
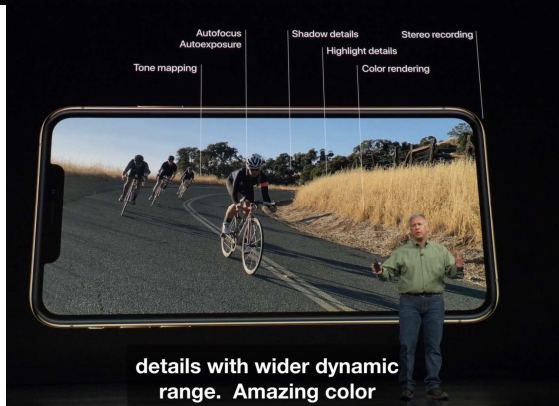
- Background & Motivation
- Curriculum
  - Choices made in teaching the class.
  - Some perspectives
- Some representative details of topics
  - Machine Learning running example
- Experiences:
  - What worked
  - What we plan to change in the next iteration

# 2017: Peak Machine Learning, Post-Peak EDA

---

- EDA is not taught at universities as much as before
- ...Yet custom hardware is more relevant than ever for Machine Learning
- Stanford Professors **Mark Horowitz** and **Subhashish Mitra** recognized a demand for an undergraduate course
- Chance meeting at the 2017 Design Automation Conference


# Mobile System-on-Chip with Inference Engine



# 5 trillion

A12 Neural Engine operations per second

second. Well, the A12 Bionic is able



At 1/10th the energy

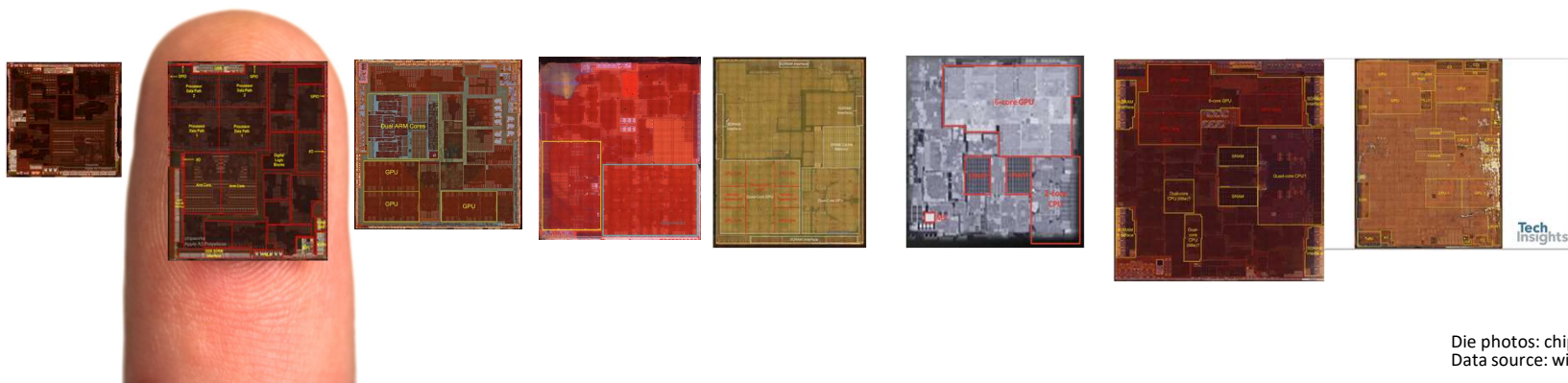
But it's going to run with as little as

A12 'Bionic' SoC:  
7nm, 6.9 Billion transistors  
Big-little: 2 HP cores, 4 LP cores  
8 core 'Next Generation Neural Engine'

© Patrick Groeneveld  
Chipitas, Sept 2018

# 9 Generations of Apple Mobile System-on-Chips

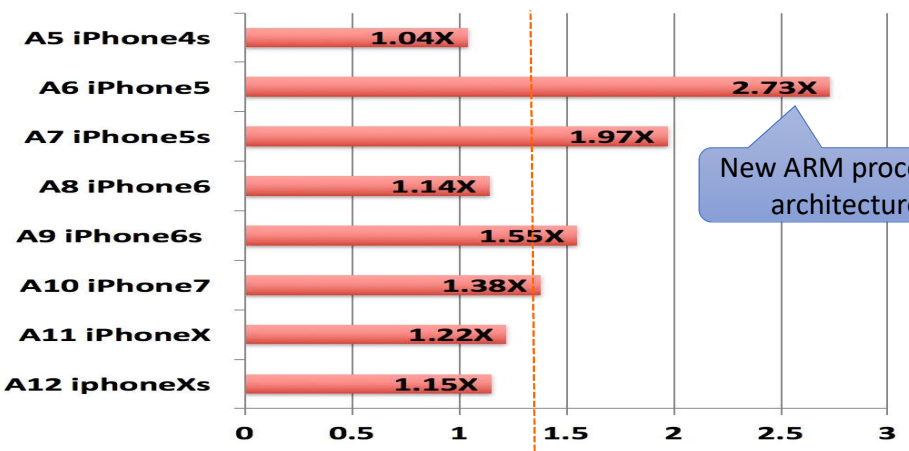
Chip	A4	A5	A6	A7	A8	A9	A10	A11	A12
Year	2010	2011	2012	2013	2014	2015	2016	2017	2018
Device	iPhone 4	iPhone 4s	iPhone 5	iPhone 5s	iPhone 6	iPhone 6s	iPhone 7	iPhone 8 & X	iPhone Xs & Xr
Node	45nm Samsung	45nm Samsung	32nm Samsung	28nm Samsung	20nm TSMC	16nm/14nm TSMC/Samsung	16nm TSMC	10nm TSMC	7nm TSMC
Area [cm <sup>2</sup> ]	0.52	1.25	0.96	1.03	0.89	1.05/0.96	1.25	0.88	1.0??



Die photos: chipworks/TechInsights  
Data source: wikipedia

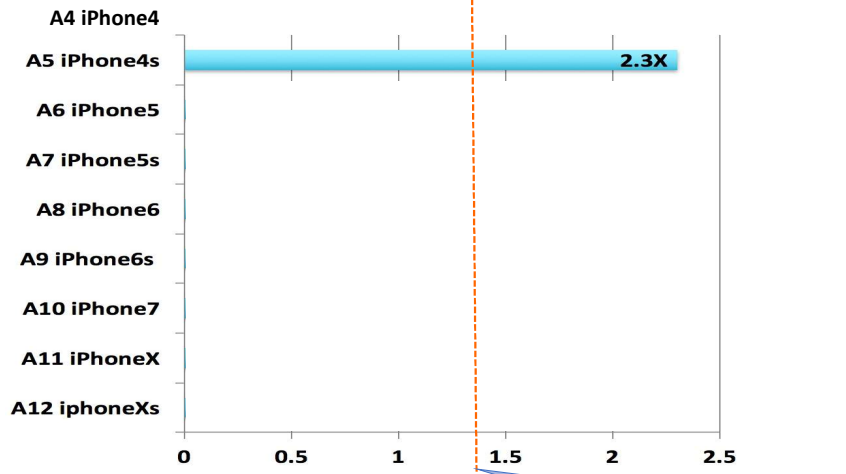
# Relative Scaling vs Predecessor

Geekbench4 Improvement

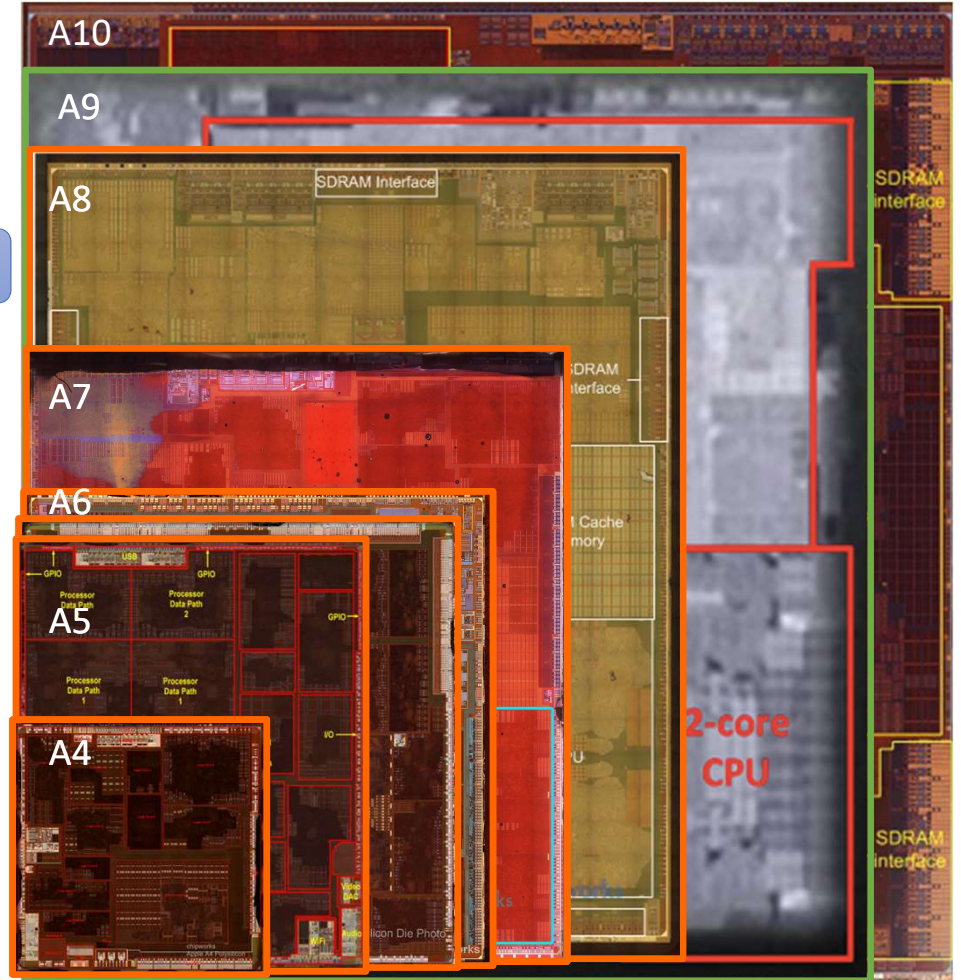


New ARM processor architecture

Transistor Count increase



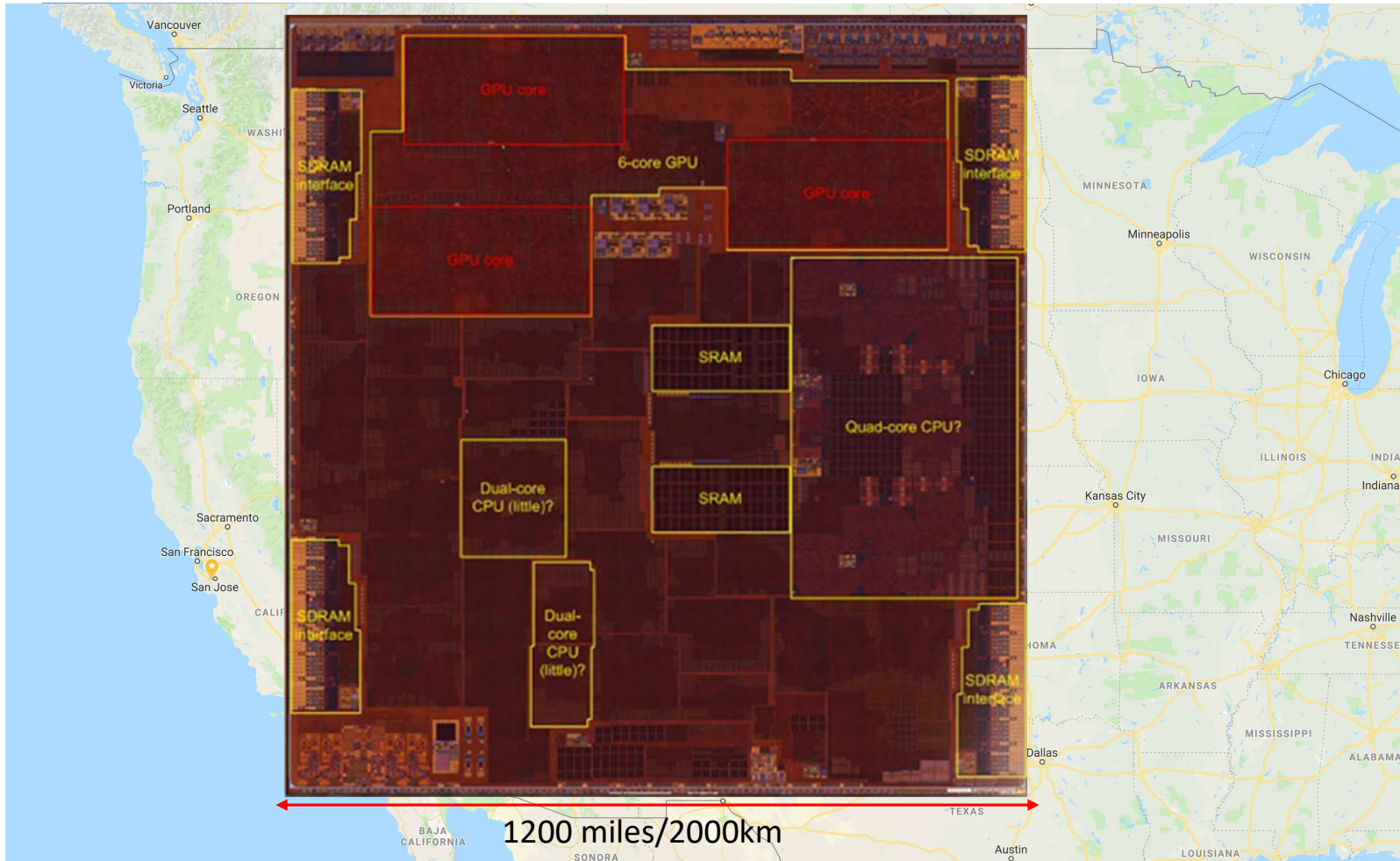
Moore's law = 1.38x



Relative SoC die size (normalized to 45nm)

Die photos: chipworks

# If the Wires on a 10nm Mobile SoC Were as Wide as Roads...



A Chip contains  
~10 million km  
wires in 10+ layers.  
Connecting  
4.4 Billion transistors  
in  
0.2 Billion cells

The USA contains  
~4.3 million km  
paved roads in 1 layer.  
Connecting  
0.3 Billion people  
in  
0.14 Billion homes

# Lecturers from Industry

---



- Raúl Camposano,
  - Sage CEO, previously Synopsys GM, Professor at GMD



- Antun Domic
  - Synopsys CTO, previously Cadence, Digital Equipment



- Patrick Groeneveld
  - Cadence R&D, previously Synopsys, Magma, professor at TU Eindhoven



# EE292A: Choices, choices, choices

---

- Depths vs Breadth
  - We chose breadth: to support the resurgence of the ‘tall thin engineer’
  - Cover all relevant steps between System Level and Layout
- Synthesis vs Analysis
  - Synthesis (design) only
- Flow vs Algorithm:
  - We do both: explain key algorithms but put in flow context
- Machine Learning aspect:
  - We focus on hardware design of a Convolutional Neural Network
    - Performance/cost trade-offs
- Lab: ASIC vs FPGA
  - We chose OpenCL -> FPGA for practical reasons

# ASIC Lab: from Idea to Chip

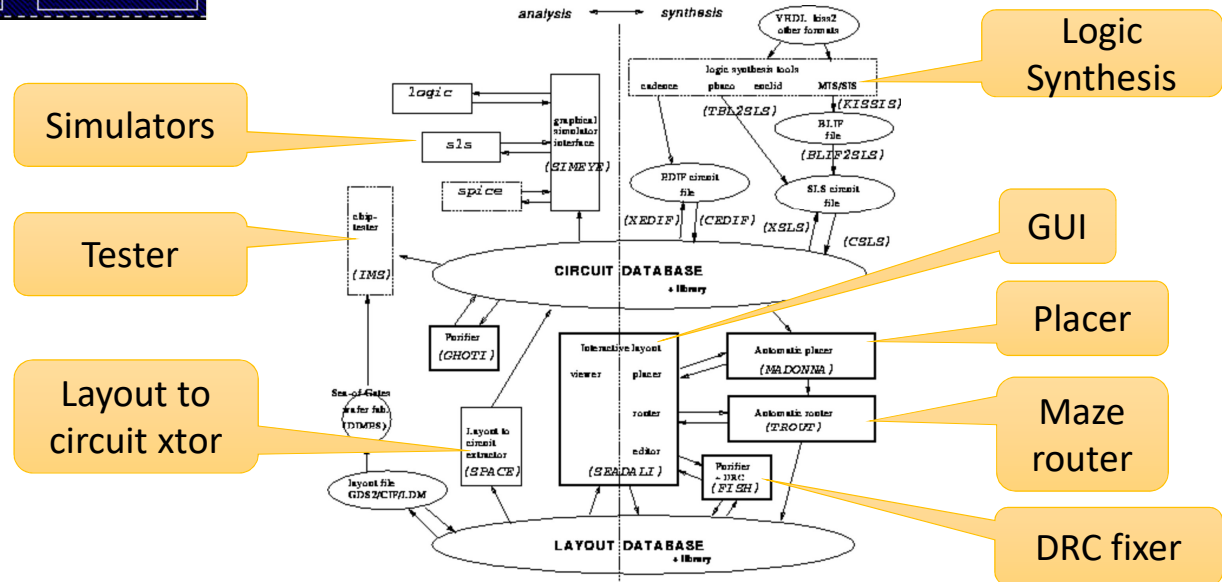
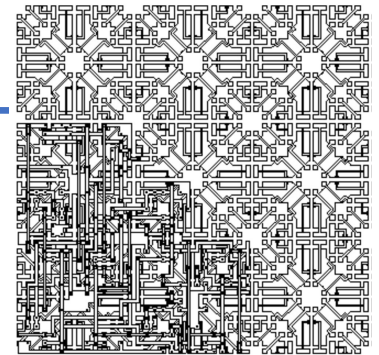
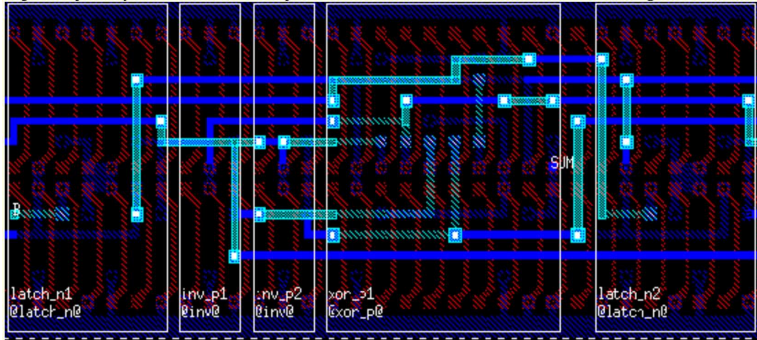
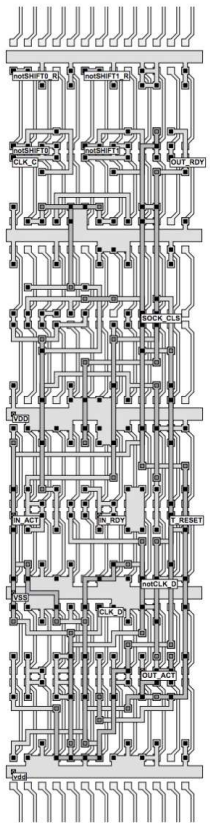


Figure 4.1. The main programs in the OCEAN Sea-of-Gates Design System. The arrows indicate the design flow.

# EE292A: Syllabus Outline

---

The class teaches cutting-edge **algorithms for the design of complex digital integrated circuits**. It provides working knowledge of the key technologies in EDA, focusing on synthesis algorithms that perform the major transformations between levels of abstraction and get a design ready to be fabricated.

These programs manipulate massive data today's most advanced integrated circuits contain over 10 billion transistors and trillions of polygons. You will become familiar with design flows and the internal algorithms of design automation software tools.

As running example, we will use the design of a **convolutional neural network** (CNN) for basic image recognition, which is a pertinent example of the interaction between hardware and software for machine learning.

You will learn how to apply machine learning to optimize certain aspects of EDA and you will design a convolutional neural network (CNN) You will implemented and test the design on a state-of-the-art FPGA board

# Inspiration

---

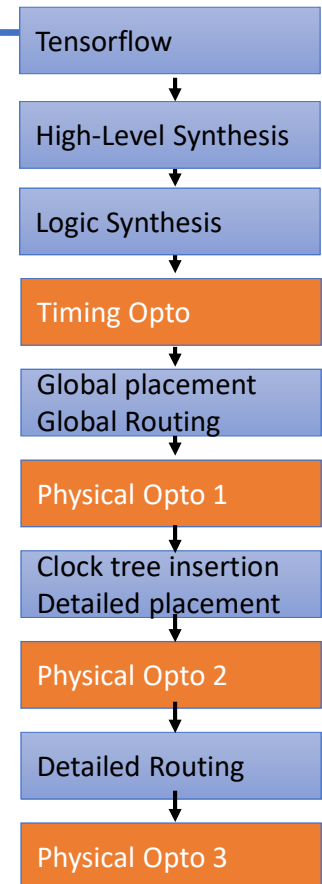
- Rob Rutenbar's MOOC at Coursera:
  - "From Logic to Layout"
- CAD classes at UT Austin, San Diego, Wisconsin, others.
- Several (older) books



# Basic Ideas, Practical Considerations

---

- Expose students to the immensely complex EDA design flow:
  - With many abstraction levels
  - With complex interactions
- The core synthesis algorithms that make or break the flow
  - Engineering trade-offs that need to be made
  - State of the art: what works?, what remains hard work?
- Hands-on lab: Synthesize Linear Classifier Hardware
  - On Intel/Altera DE-10 FPGA board, & FPGA synthesis tools: remote login
  - Trade-offs: speed vs accuracy vs area, Fixed-Point32, 16, 8 and 4 bit



GDS2

© Patrick Groeneveld  
EDPS Milpitas, Sept 2018

# A few more Relevant Logistics

---

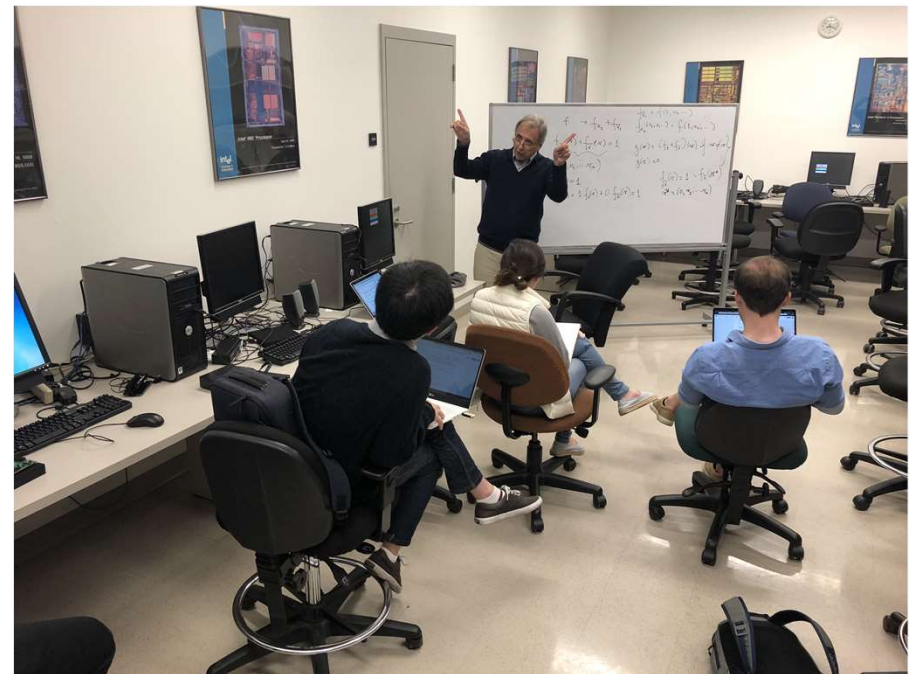
- What we expected the students to know:
  - Basic circuit design, Digital systems design (EE 108/109), programming skills (Software Projects, C/C++) and Machine Learning.
- Spring semester:
  - 18 classes of 80 minutes each, twice weekly (9 weeks)
  - Weekly Homework & lab assignments
  - Weekly office hours
  - Cap of 38 students
- Assistance:
  - 2 Teacher-Assistants
  - Lab coordinator



# E292A: Teaching the class

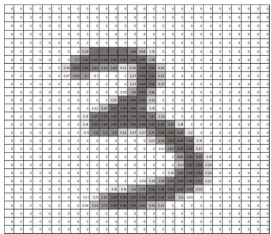
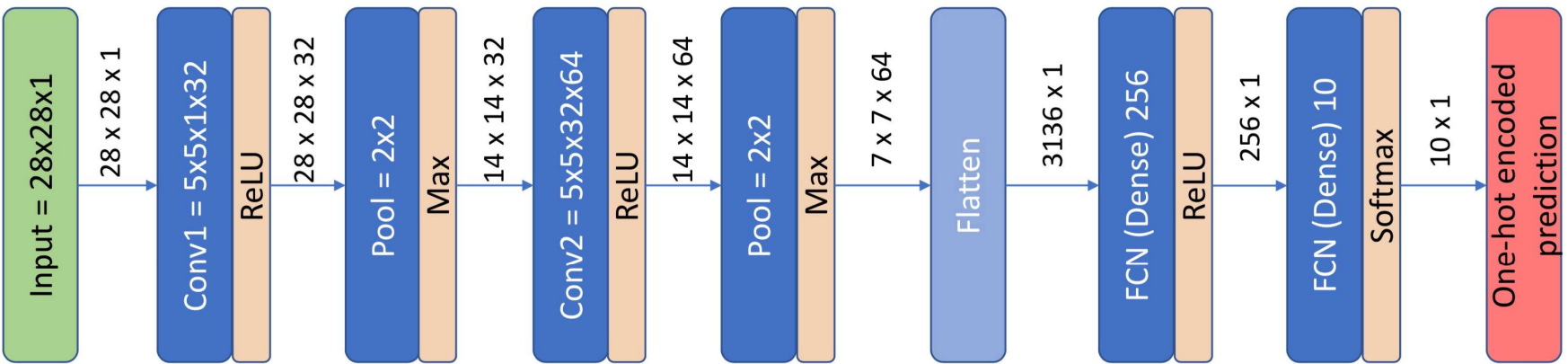


- 80 minutes, no break
- ~ 55 slides per session.

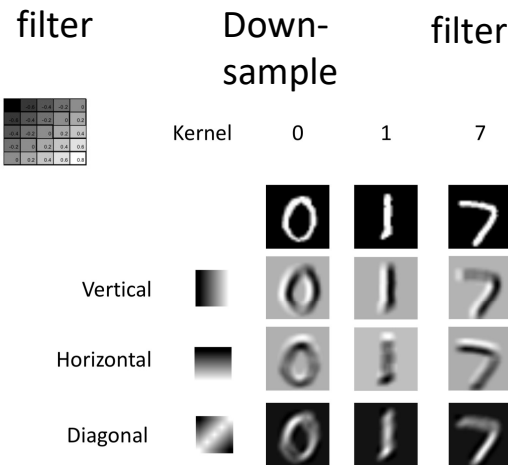


- Weekly office hours, discussing homework

# CNN Lab: Classify the numbers 0-9



28x28 image



Fully Connected Fully Connected

0
1
2
3
4
5
6
7
8
9



# Lab: Linear Classifier

$$\text{scores} = W x_i + b$$

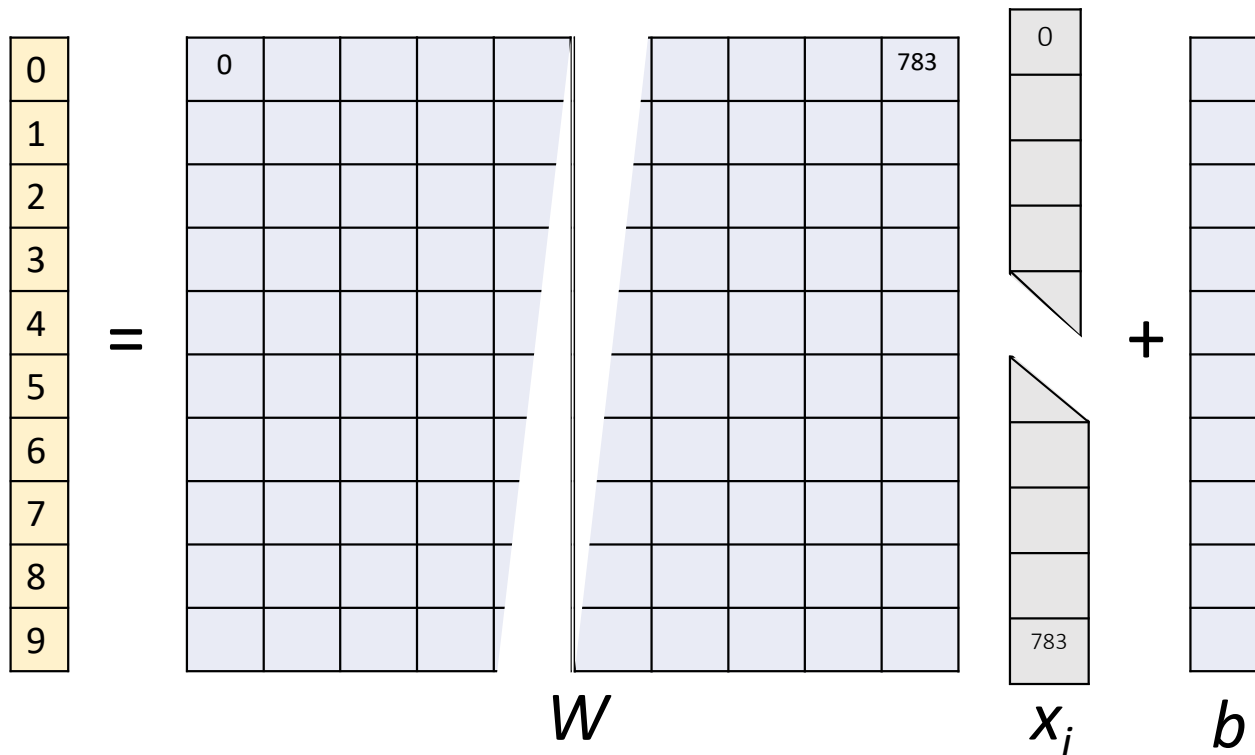
$W$  = weight matrix

$x_i$  = image  $i$  pixels  
stretched into  
single column

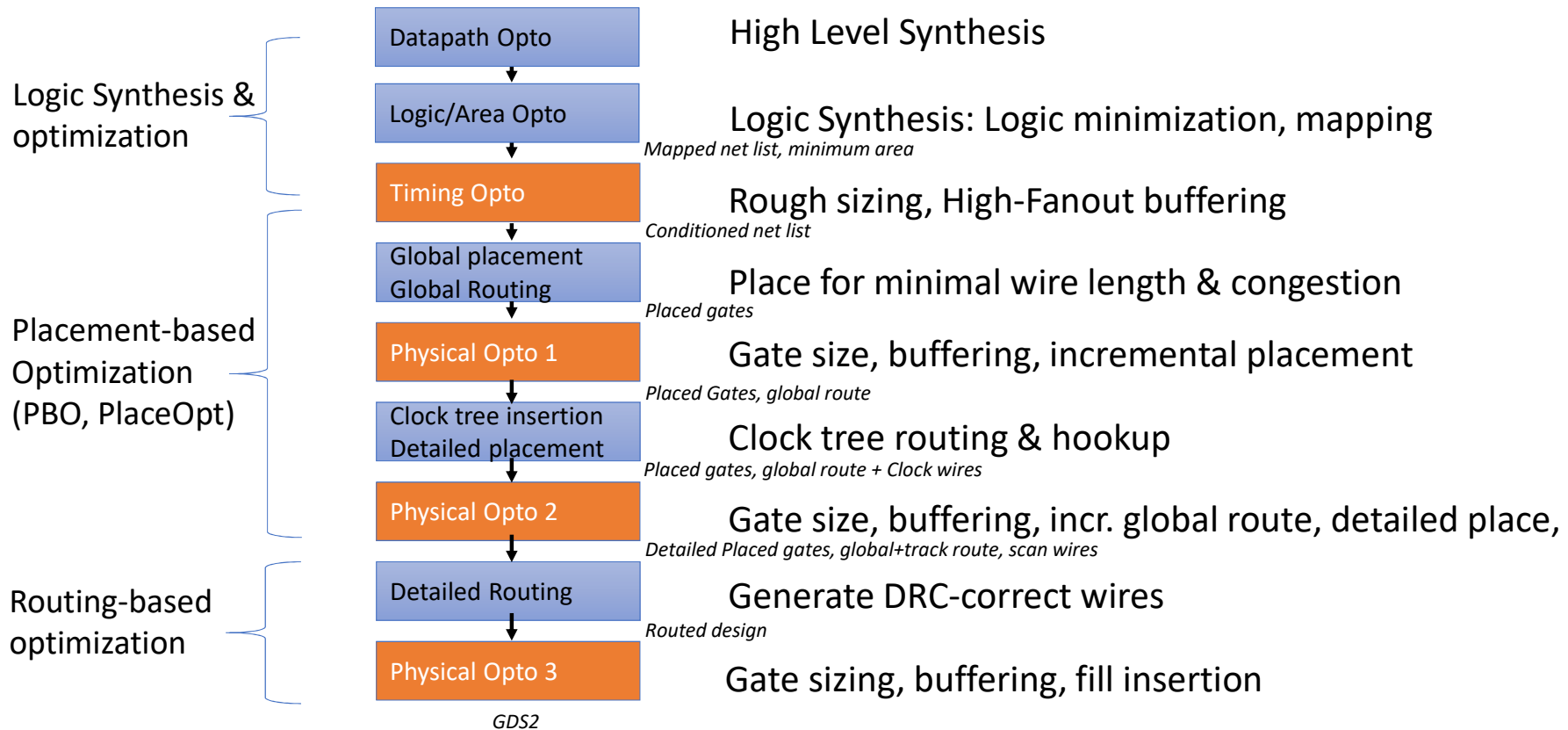
$b$  = bias vector

Images 28 x 28

$0 \leq i \leq 783$



# Compare CNN to EDA flow



# Generic Optimization: Code Skeleton

```
void OPT::simulatedAnnealing(string & current) {  
    for(double temp = _hot; temp > _cold; temp *= 0.95) {  
        for(int i = 0; i < _maxIter; i++) {  
            string attempt = perturb(current);  
            float delta = cost(attempt) - cost(current);  
            if(delta < 0.0 ||  
                random(1.0) < exp(-delta/(K*temp)) ) {  
                current = attempt;           // Accept  
            }  
        }  
    }  
}
```

Cool from high temperature  
to cold temperature

Run many attempts at each  
temperature.

Generate a random move

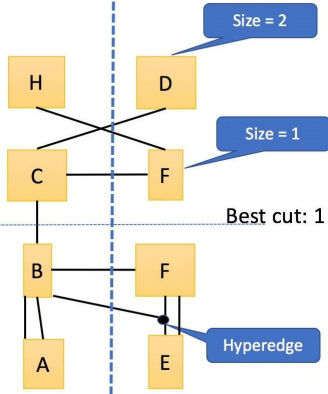
Calculate the change in cost  
of the perturb

Accept if cost decreases

Accept cost deterioration  
with probability  $e^{-\Delta E/kT}$

# F-M Code Skeleton

Initial cut: 5



Calculates the gain of all vertices.  
Only remembers best move of unlocked and movable gates

Moves the best gain

Remembers the configuration with the lowest cut cost

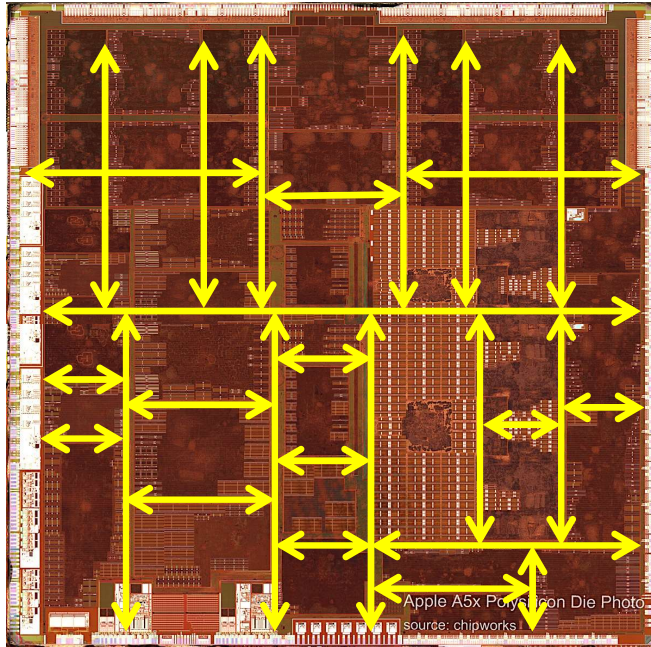
Backs up results until the one with the lowest cut count

```
void fiducciaMattheysesPartitioning(graph G, graph & L, graph & R) {
    randomPartition(G, L, R);
    while(true) {
        int cutCountDelta = 0, bestCutCountIndex = -1;
        for(int pass = 0; /* empty */ ; pass++) {
            // try until there are no unlocked moves left
            int bestGain = 0, bestGainIndex = -1;
            for(int i=0; i < G.size(); i++) {
                int gain = retentionForce(G[i]) - moveForce(G[i]);

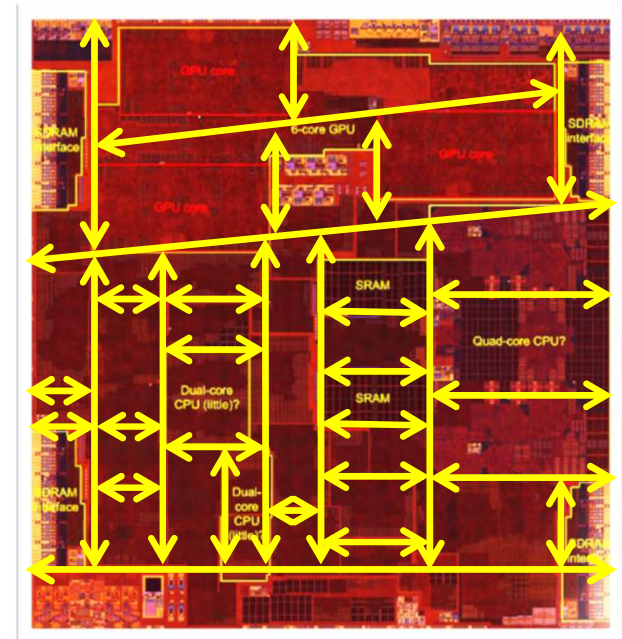
                if(areaBalanceOK(G[i]) && // obeys area balance
                    isLocked(G[i]) == false && // is not moved before
                    gain > bestGain) { // is the best we've seen
                    bestGain = gain; bestGainIndex = i;
                }
            }
            if(bestGainIndex < 0) {
                break; // Stop: no improvement found
            }
            // Execute the best move we've seen
            moveToOtherPartitionAndLock(bestGainIndex, L, R);
            cutCountDelta -= bestGain; // maintain current cut count
            if(cutCountDelta < bestCurCountDelta) {
                // we have a new winner! Remember it
                bestCutCountDelta = cutCountDelta;
                bestCutCountIndex = i;
            }
        }
        if(bestCutCountIndex < 0) {
            break; // stop: there were no more improvements
        }
        undoMovesUntil(bestCutCountIndex, L, R); // keep the best cut
        // and try again
    }
}
```

# Floorplans are 'Near-Slicing' in Practice

- Slicing tree: a rectangular dissection that can be obtained by repetitively subdividing rectangles horizontally or vertically



Apple A5: Slicing (with a few corners cut)



Apple A10: Non-slicing, but likely started out as slicing

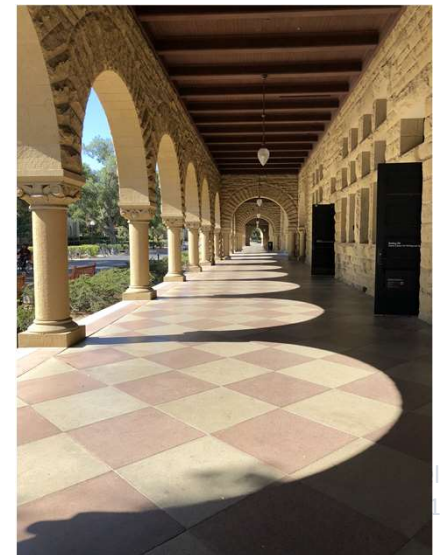
# Stanford Environment



The building where EE292A classes were taught



<sup>22</sup> Prepping in the library before class



# Summary: EDA & Machine Learning Hardware

---

- What worked:
  - Class was popular, students excellent (100% pass)
  - Great assistance from TA's
  - Overall really uplifting experience for all involved
- What we plan to improve:
  - Intel FPGA board flow did not work smooth
  - Students request running 'real' EDA tools

