18-643 Lecture 1:
Welcome, why are you here?

James C. Hoe
Department of ECE
Carnegie Mellon University
Housekeeping

• Your goal today: decide if you are coming back . . .

• Notices (all handouts on Canvas)
  – Handout #1: syllabus
  – Handout #2: lab 0, due noon, Friday 9/11
  – Complete survey on Canvas, due noon, Wed 9/9

• Readings
  – Ch 1, Reconfigurable Computing
Field Programmable Gate Arrays: in the beginning

- I/O pins
- Programmable lookup tables (LUT) and flip-flops (FF)
  aka “soft logic” or “fabric”
- Programmable routing

Interconnect
Original Xilinx FPGA

There was another alternative . . .
A Quite Wondrous Device

• Make an ASIC from your desk all by yourself
  – no manufacturing NRE (non-recurring eng.) cost
  – faster design time: try out increments as you go
  – less validation time: debug as you go at full speed / can also patch after shipping

• But
  – high unit cost (not for high-volume products)
  – “~10x” overhead in area/speed/power/....
  – RTL-level design abstraction

• Somewhere between ASICs and processors
FPGA “Growing Pains”

• Real designers make ASICs
• It is not programmable if it is not “C”
  – until 2005, CPUs were fast and getting faster
  – after 2005, GPGPU happened
• Where are the killer apps?
  – performance demanding but not too demanding
  – enough volume but not too high
  – high-concurrency but not totally regular
  – functionalities evolve quickly but not too quickly
FPGA Killer Apps Over Time

• 1990s: glue logic, embedded cntrl, interface logic
  – reduce chip-count, increase reliability
  – rapid roll-out of “new” products

• 2000s: DSP and HPC
  – strong need for performance
  – abundant parallelism and regularity
  – low-volume, high-valued

• 2010s: communications and networking
  – require high-throughput and low-latency
  – fast-changing designs and standards
  – price insensitive
  – $value in field updates and upgrades
“Age of Expansion”

8. Growth of the FPGA addressable market.

[Fig 8, S. M. Trimberger, “Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology.”]
Fast-forward through Moore’s Law

Fig. 1. Xilinx FPGA attributes relative to 1988. Capacity is logic cell count. Speed is same-function performance in programmable fabric. Price is per logic cell. Power is per logic cell. Price and power are scaled up by 10,000×. Data: Xilinx published data.

[Fig 1, S. M. Trimberger, “Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology.”]
“Age of Accumulation”

Fig. 11. Shrinking growth of the FPGA addressable market.

[Fig 11, S. M. Trimberger, “Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology.”]
New Age in FPGA Computing

• Every Microsoft datacenter server has an FPGA
  – Bing, Brainwave, . . .
  – try googling also
    “<<big-cloud-company-X>> FPGA datacenter”
• AWS will rent you servers with FPGAs (EC2-F1)
• IBM and Intel want to sell you CPUs with cache-coherent FPGA accelerators
• You can buy FPGA fabric IP to add to your own chip

Why the new interest from computing?
Moore’s Law without Dennard Scaling

2013 Intl. Technology Roadmap for Semiconductors

- logic density
- VDD

Under fixed power ceiling, more ops/second only achievable if less Joules/op?
Future is about
Performance/Watt and Ops/Joule

This is a sign of desperation . . . .
Why is HW/FPGA better?

no overhead

• A processor spends a lot of transistors & energy
  – to present von Neumann ISA abstraction
  – to support a broad application base (e.g., caches, superscalar out-of-order, prefetching, . . . )

• In fact, processor is mostly overhead
  – ~90% energy [Hameed, ISCA 2010, Tensilica core]
  – ~95% energy [Balfour, CAL 2007, embedded RISC ]
  – even worse on a high-perf superscalar-OoO proc

Computing directly in application-specific hardware can be 10x to 100x more energy efficient
Why is HW/FPGA better? efficiency of parallelism

- For a given functionality, non-linear tradeoff between power and performance
  - slower design is simpler
  - lower frequency needs lower voltage

⇒ For the same throughput, replacing 1 module by 2 half-as-fast reduces total power and energy

*Good hardware designs derive performance from parallelism*
Digression: Other Looming Dooms

- **Memory Wall**
  - Moore’s Law scaled DRAM in capacity not speed
  - relative to logic, DRAM looks slower and slower
  - dark-silicon from data starvation?

- **Complexity Wall**
  - designer productivity grew slower than Moore’s Law on logic capacity
  - design team grew exponentially to make up
  - more transistors than a team can make all work?
  (see *Mythical Man-Month*)
Differing Tradeoffs and Sweetspots

Efficiency
("good" per "cost")

Commitment:
- data type
- operations
- exploitable parallelism

Ease

Versatility

CGRA/GPU

FPGA

CPU
Software to Hardware Spectrum

- **CPU**: highest-level abstraction / most general-purpose support
- **GPU**: explicitly parallel programs / best for SIMD, regular
- **FPGA**: ASIC-like abstraction / overhead for reprogrammability
- **ASIC**: lowest-level abstraction / fixed application and tuning
Heterogeneity for Efficiency

Don’t think of modern FPGAs as RTL targets!
Why this course

• Will FPGAs continue to gain importance in computing?

• If so, what will computing FPGAs (separate from ASIC-type FPGAs) look like in the future?

These are not questions to be asked passively . . .
Go Over Canvas and Syllabus
Be Forewarned

• This is still a “young” course
  – no course out there exactly like this one
  – the topic area is “unsettled” and in transition
• This is a hard course
  – 5 “training” labs; 1 large open-ended project
  – 1 midterm (1st half)
  – weekly paper reviews (2nd half)
  – you must speak up in class
• I am assuming
  – you are into computer hardware
  – you know RTL
  – you are willing to suffer for performance