18-643 Lecture 1: Not Your Parents’ FPGAs

James C. Hoe
Department of ECE
Carnegie Mellon University
FPGAs as we knew them
(FPGA=Field Programmable Gate Array)

Traditionally, FPGAs have been the \textit{bastard step-brother of ASICs}. They have been forced to act like ASICs and fit themselves into the ASIC development model. . . . . . .

. . . . . . This has meant ignoring their unique strengths: \textit{reprogrammability, late binding} and \textit{runtime reconfiguration}.

Andre DeHon, ISFPGA 2004 Panel
https://dl.acm.org/doi/10.1145/968280.968281
New FPGAs not RTL targets

Are they FPGAs?
• spatial data/compute
• highly concurrent
• finely controllable
• reprogrammable

[Xilinx Versal]
[Achronix Speedster]
[Intel Agilex]
What you will be using:
Xilinx Zynq SoC FPGA

Housekeeping

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

• Notices (all handouts on Canvas)
  – Handout #1: syllabus
  – Handout #2: lab 0, due noon, Mon 9/11
  – Complete survey on Canvas, due noon, Wed 9/6

• Readings (see lecture schedule online)
Field Programmable Gate Arrays: in the beginning

programmable routing

programmable lookup tables (LUT) and flip-flops (FF)
aka “soft logic” or “fabric”
Original Xilinx FPGA

The other alternative to ASIC . . .

https://www.computerhistory.org/revolution/digital-logic/12/278/1445
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 software
Early FPGA “Growing Pains”

• Real HW designers tapeout 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

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

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

• ~2010: communications and networking
  – require high-throughput and low-latency
  – fast-changing designs and standards
  – price insensitive
  – $value in field updates and upgrades

What is in store in 2020s?
“Age of Expansion”

[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, 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, Azure, Brainwave, . . .
  – try googling also
    “<<big-cloud-company-X>> FPGA datacenter”
• You can rent AWS servers with FPGAs (EC2-F1)
• You can buy CPUs with cache-coherent FPGA accelerators or buy FPGA with embedded CPUs
  Either way, you buy them from Intel or AMD
• Google is building FPGA tools (SymbiFlow)

Why the new interest from computing players?
Power Wall: 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
What will (or can) you do with all those transistors?

Future is about **Performance/Watt** and **Ops/Joule**

This is a sign of desperation . . . .
Why HW/FPGA better than SW: 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 HW/FPGA better than SW: 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**
Why future of computing need FPGAs in addition to “real” HW?
Past: Ingrained Formula of HW vs SW

- Do as much as possible in SW
- Pay for HW where SW not good enough
Present: Third Option Wanted

- More things SW not good enough
- Neither is HW when
  - functionalities not fixable at deployment
  - require many functionalities but never all at once
Differing Tradeoffs and Sweetspots

Efficiency
(“good” per “cost”)

ASIC

CGRA/GPU

FPGA

 Ease

Versatility

committed:
- data type
- operations
- exploitable parallelism

CPU
Heterogeneity for Efficiency

https://www.xilinx.com/versal

https://www.intel.com/FPGA/agilex
All roads lead to heterogeneous systems

[Figure 2: Orin System-on-Chip (SoC) Block Diagram]

[Orin SoC, Nvidia.com] [M1 “chip”, Apple.com]
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 . . ._
Check out https://crossroadsfpga.org

(sign up for seminar announcements)
Be Forewarned

• The topic area is “unsettled” and in transition
• This is a hard course
  – 4 “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
Go Over Canvas and Syllabus