

ESE532: System-on-a-Chip Architecture

Day 1: September 2, 2020
Introduction and Overview
(lecture start 10:35am)

Note: both linked to web (also slides)
www.seas.upenn.edu/~ese532/fall2020/fall2020.html

- Preclass
- Feedback form

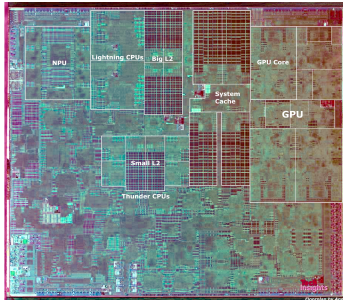


Today

- Case for Programmable SoC
- Course Goals
- Outcomes
- Evolving Course, Risks, Tools
- Sample Optimization
- This course (incl. policies, logistics)

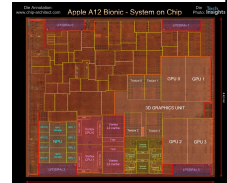
Apple A13 Bionic

- 98mm², 7nm
- 8.5 Billion Tr.
- iPhone 11 +
- 6 ARM cores
 - 2 fast (2.6GHz)
 - 4 low energy
- 4 custom GPUs
- Neural Engine
 - 5 Trillion ops/s?



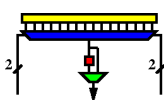
Questions

- Why do today's SoC look like they do?
- How approach programming modern SoCs?
- How design a custom SoC?
- When building a System-on-a-Chip (SoC)
 - How much area should go into:
 - Processor cores, GPUs, FPGA logic, memory, interconnect, custom functions (which) ?

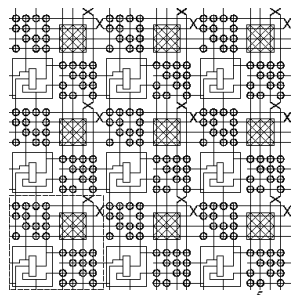


FPGA Field-Programmable Gate Array

K-LUT (typical k=4 or 6)
Compute block
w/ optional
output Flip-Flop



ESE171, ESE150, CIS371



Case for Programmable SoC

End of uProcessor Scaling

Old

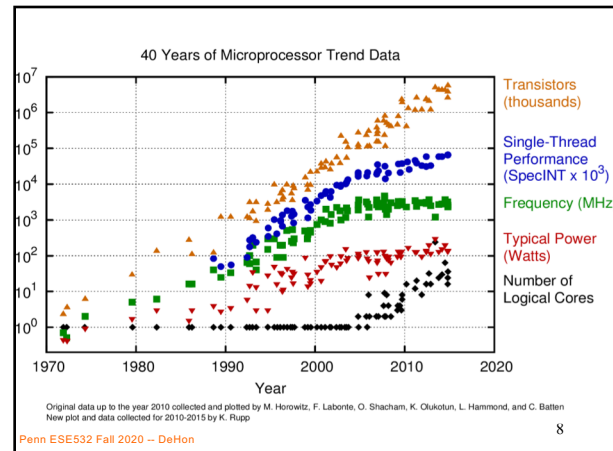
- Moore's Law scaling delivered faster transistors
- Processors rode Moore's Law
 - Turning transistors into performance
- Could wait and ride technology curve

Now

- Dennard's Law kicked in
- uP were burning more power
- Lost ability to scale down voltage
- Processor performance stalled

Penn ESE532 Fall 2020 -- DeHon

7



The Way things Were

30 years ago

- Wanted programmability
 - used a processor
- Wanted it a little faster
 - Next year's processor would run faster...
- Wanted high-throughput
 - used a custom IC
- Wanted product differentiation
 - Got it at the board level
 - Select which ICs and how wired
- Build a custom IC
 - It was about gates and logic

Penn ESE532 Fall 2020 -- DeHon

9

Today

- Microprocessor may not be fast enough
 - (but often it is)
 - Or low enough energy
 - Single core processor scaling has ended
 - Time and Cost of a custom IC is too high
 - \$100M's of dollars for development, Years
 - FPGAs promising
 - But build everything from prog. gates?
 - Premium for small part count
 - And avoid chip crossing
- ICs with Billions of Transistors

Penn ESE532 Fall 2020 -- DeHon

10

Non-Recurring Engineering (NRE) Costs

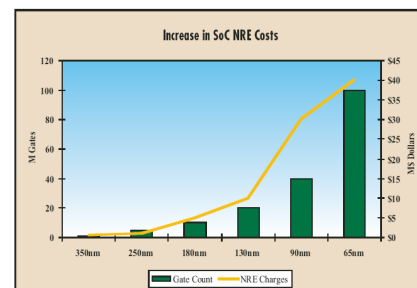
- Costs spent up front on development
 - Engineering Design Time
 - Prototypes
 - Mask costs
- Recurring Engineering
 - Costs to produce each chip

$$Cost(N_{chips}) = Cost_{NRE} + N_{chips} \times Cost_{perchip}$$

Penn ESE532 Fall 2020 -- DeHon

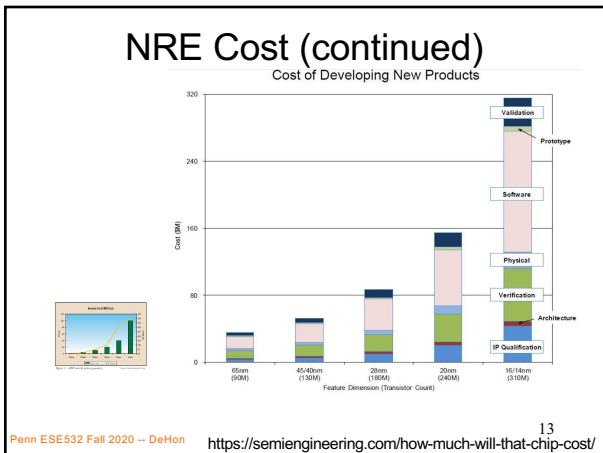
11

NRE Costs



Penn ESE532 Fall 2020 -- DeHon

12

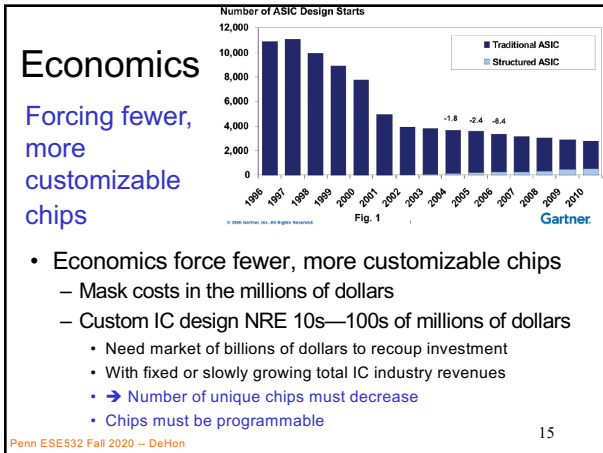


Amortize NRE with Volume

$$Cost(N_{chips}) = Cost_{NRE} + N_{chips} \times Cost_{perchip}$$

$$Cost = \frac{Cost_{NRE}}{N_{chips}} + Cost_{perchip}$$

Penn ESE532 Fall 2020 – DeHon



- ### Large ICs
- Now contain significant software
 - Almost all have embedded processors
 - Must co-design SW and HW
 - Must solve complete computing task
 - Tasks has components with variety of needs
 - Some don't need custom circuit
 - 90/10 Rule
- Penn ESE532 Fall 2020 – DeHon

Given Demand for Programmable

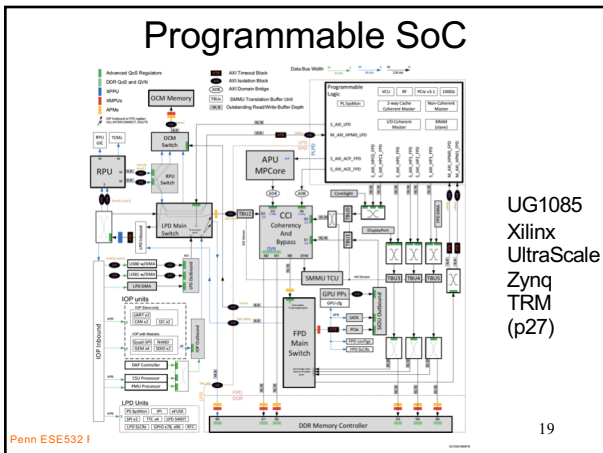
- How do we get higher performance than a processor, while retaining programmability?

17

Programmable SoC

- Implementation Platform for innovation
 - This is what you target (avoid NRE)
 - Implementation vehicle

Penn ESE532 Fall 2020 – DeHon



Then and Now

30 years ago

- Programmability?
 - use a processor
- Faster
 - Processors scaled
- High-throughput
 - used a custom IC
- Wanted product differentiation
 - board level
 - Select & wired IC
- Build a custom IC
 - It was about gates and logic

Today

- Programmability?
 - uP, FPGA, GPU, PSoC
- Faster
 - Can't get with single core
- High-throughput
 - FPGA, GPU, PSoC, custom
- Wanted product differentiation
 - Program FPGAs, PSoC
- Build a custom IC
 - System and software

Penn ESE532 Fall 2020 – DeHon 20

Course Goals, Outcomes

Penn ESE532 Fall 2020 – DeHon

21

Goals

- Create Computer Engineers
 - SW/HW divide is wrong, outdated
 - **Computer engineers understand computation**
 - HW and SW are just tools and design options
 - Parallelism, data movement, resource management, abstractions
 - Cannot build a chip without software
- SoC user – know how to exploit
- SoC designer – architecture space, hw/sw codesign
- Project experience – design and optimization

Penn ESE532 Fall 2020 – DeHon

Roles

- PhD Qualifier
 - One broad Computer Engineering
- CMPE Concurrency
- Hands-on Project course

Penn ESE532 Fall 2020 – DeHon

23

Outcomes

- Design, optimize, and program a modern System-on-a-Chip.
- Analyze, identify bottlenecks, design-space
 - Modeling → write equations to estimate
- Decompose into parallel components
- Characterize and develop real-time solutions
- Implement both hardware and software solutions
- Formulate hardware/software tradeoffs, and perform hardware/software codesign

Penn ESE532 Fall 2020 – DeHon

24

Outcomes

- Understand the system on a chip from gates to application software, including:
 - on-chip memories and communication networks, I/O interfacing, design of accelerators, processors, firmware and OS/infrastructure software.
- Understand and *estimate* key design metrics and requirements including:
 - area, latency, throughput, energy, power, predictability, and reliability.

Penn ESE532 Fall 2020 -- DeHon

25

Evolving Course

- Spring 2017 – first offering
 - Raw, all assignments new, buggy, too tedious, long
- Fall 2017 – second offering
 - Refine assignments, project; increased explicit modeling emphasis
 - Hard, not insane
- Fall 2018 – third offering (similar 2017)
 - Added real-time ethernet data handling; project groups of 3
 - Many students challenged with C and software engineering
 - Stream debug and performance challenging
- Fall 2019 – fourth (structure same)
 - Try front-load more C, better introduce Stream optimization and debug
 - Group writeup on projects
- Fall 2020 – now
 - Move to Vitis (from SDSoC)
 - Use Amazon cloud for first half; F1 instance for FPGA access HW
 - Then transition to Ultra96 (SoC FPGA) for projects

Penn ESE532 Fall 2020 -- DeHon

26

Tools

- Are complex
- Will be challenging, but good for you to build confidence can understand and master
- Tool runtimes can be long
- Learning and sharing experience will be part of assignments

Penn ESE532 Fall 2020 -- DeHon

27

Distinction

CIS240, 371, 471, 571

- Best Effort Computing
 - Run as fast as you can
- Binary compatible
- ISA separation
- Shared memory parallelism

ESE532

- Hardware-Software codesign
 - Willing to recompile, maybe rewrite code
 - Define/refine hardware
- Real-Time
 - Guarantee meet deadline
- Non shared-memory parallelism models

Penn ESE532 Fall 2020 -- DeHon

28

Distinction

ESE680:

Hardware/Software Co-Design for Machine Learning

- Deep on Application (ML)
 - Less previous experience with circuits and architecture
- Won't be as deep on understanding HW and optimization
- Program in Pytorch, OpenCL

Penn ESE532 Fall 2020 -- DeHon

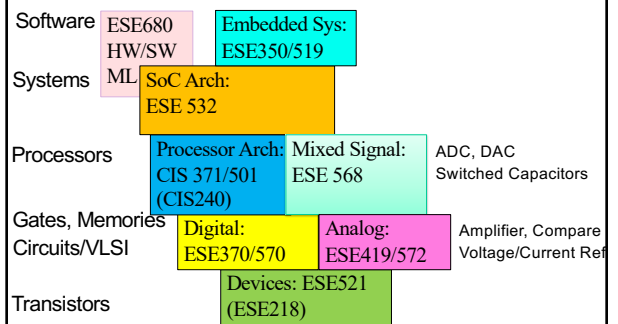
- First offering

29

ESE532:

- Deep computer engineering
- Broad application
- Program in C
- Suitable followup if want to dig deeper
- 5th offering

Abstraction Stack



Penn ESE532 Fall 2020 -- DeHon

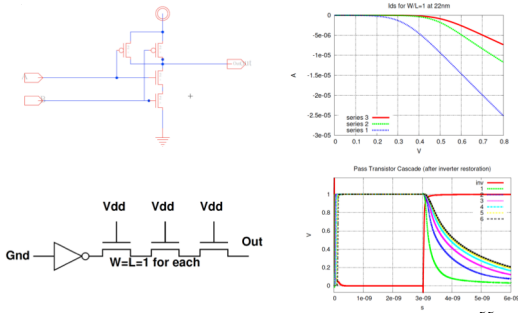
30

Approach -- Example

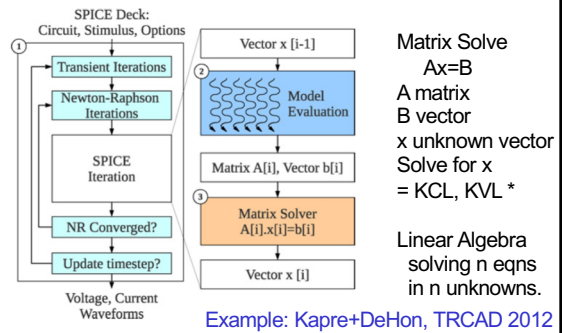
Abstract Approach

- Identify requirements, bottlenecks
- Decompose Parallel Opportunities
 - At extreme, how parallel could make it?
 - What forms of parallelism exist?
 - Thread-level, data parallel, instruction-level
- Design space of mapping
 - Choices of where to map, area-time tradeoffs
- Map, analyze, refine
 - Write equations to understand, predict

Example SPICE Circuit Simulator



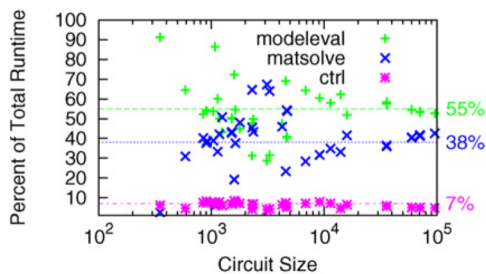
Example: SPICE Circuit Simulator



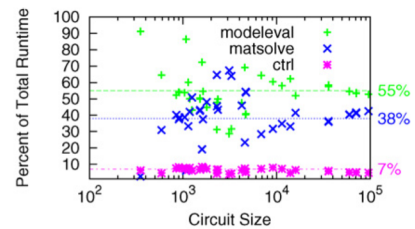
Example: Kapre+DeHon, TRCAD 2012

* Kirchoff {Current, Voltage} Laws³⁴

Analyze

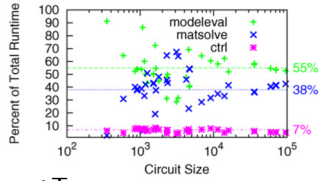


Analyze



$$T = T_{\text{modeval}} + T_{\text{matsolve}} + T_{\text{ctrl}}$$

Speedup



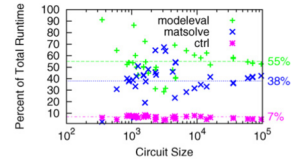
- $T = T_{\text{modeval}} + T_{\text{matsolve}} + T_{\text{ctrl}}$
 - What should we speedup first?
 - What happens if only speedup modeval?
- $T = T_{\text{matsolve}} + (T_{\text{modeval}})/S + T_{\text{ctrl}}$

Penn ESE532 Fall 2020 – DeHon

37

Analyze

- If only accelerated model evaluation only about 2x speedup
- If want better than 14x speed, must also attack control



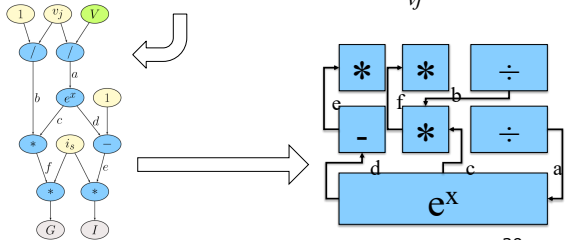
Penn ESE532 Fall 2020 – DeHon

38

Model Evaluation: Trivial Hardware Implementation

$$I_{D1} = I_s \times (e^{V_{D1}/V_j} - 1)$$

$$G_{D1} = \frac{d}{dV_{D1}}(I_{D1}) = I_s \times e^{V_{D1}/V_j} \times \frac{1}{V_j}$$

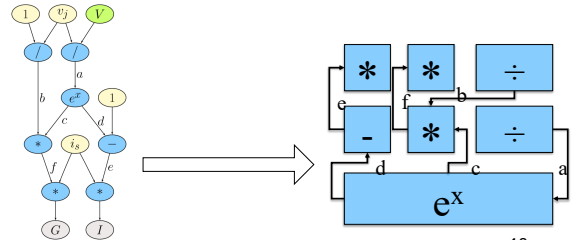


Penn ESE532 Fall 2020 – DeHon Verilog-AMS as Domain-Specific Language

39

Spatial Parallelism

- Every operation (*, + /) gets dedicated hardware.
- Implement task in space → use additional area for each operator.
- Parallel – all operations occur simultaneously.

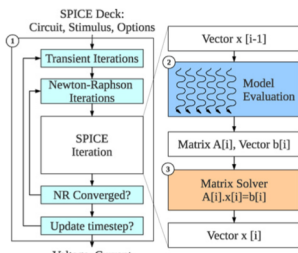


Penn ESE532 Fall 2020 – DeHon

40

Parallelism: Model Evaluation Data Parallel

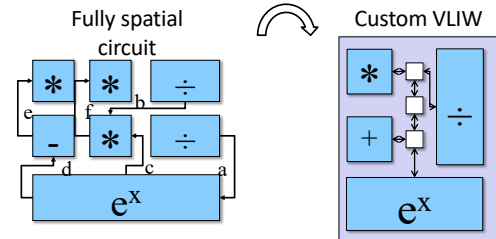
- Every device independent
- Many of each type of device
- Can evaluate in parallel
 - $T = T_{\text{seq}}/N_{\text{proc}}$
- Build pipelined circuit for model
 - $T_{\text{seq}} = N_{\text{comp}} * T_{\text{cycle}}$
 - vs. $T_{\text{pipe}} = T_{\text{cycle}}$



Penn ESE532 Fall 2020 – DeHon

41

Spatial Too Big?



~100x Speedup
Multiple FPGAs

~10x Speedup
1 FPGA

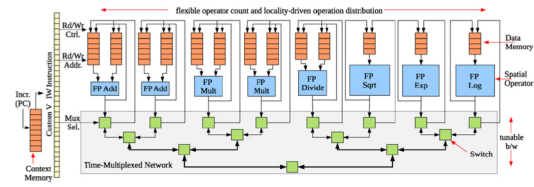
VLIW=Very Long Instruction Word
exploits Instruction-Level Parallelism

Penn ESE532 Fall 2020 – DeHon

42

Parallelism: Model Evaluation

- Spatial end up bottlenecked by other components
- Use custom evaluation engines
- ...or GPUs

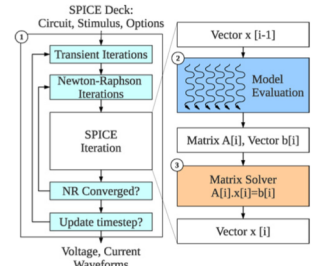


Penn ESE532 Fall 2020 -- DeHon

43

Parallelism: Matrix Solve

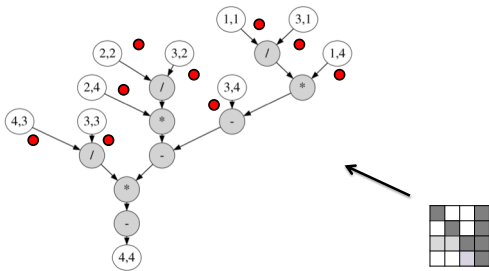
- Needed direct solver?
- E.g. Gaussian elimination
- Data dependence on previous reduce
 - Limited data parallelism
- Parallelism in subtracts
- Some row independence



Penn ESE532 Fall 2020 -- DeHon

44

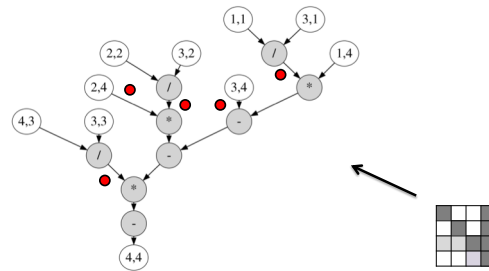
Example Matrix



Penn ESE532 Fall 2020 -- DeHon

45

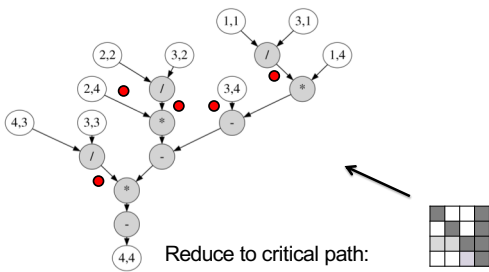
Example Matrix



Penn ESE532 Fall 2020 -- DeHon

46

Example Matrix

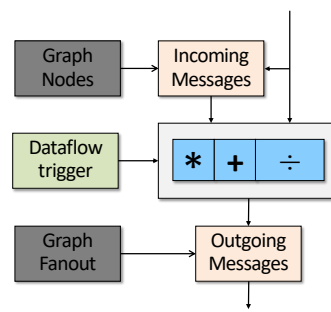


Reduce to critical path:
from 9 sequential operations
to path of 5 operations.

Penn ESE532 Fall 2020 -- DeHon

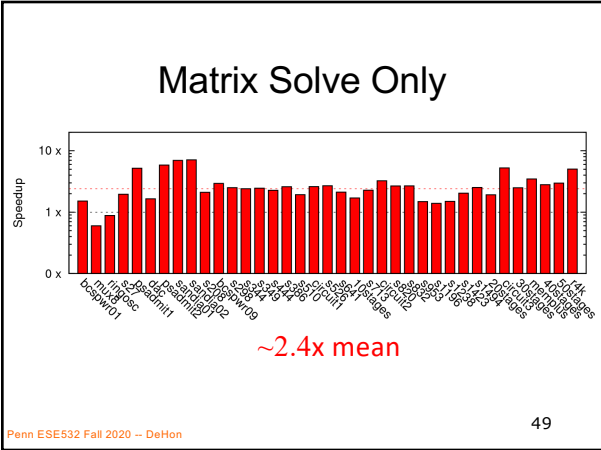
47

Dataflow Processing Element (PE)



Penn ESE532 Fall 2020 -- DeHon

48

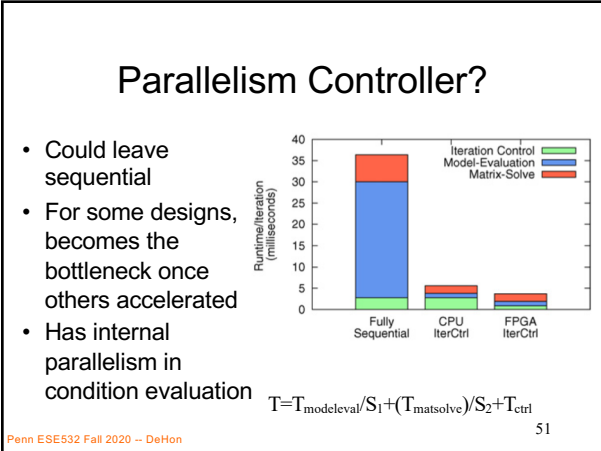


Parallelism: Matrix Solve

- Settled on constructing dataflow graph
- Graph can be iteration independent
 - Statically scheduled
 - (cheaper)
- This is bottleneck to further acceleration

Penn ESE532 Fall 2020 -- DeHon

50



Parallelism Controller

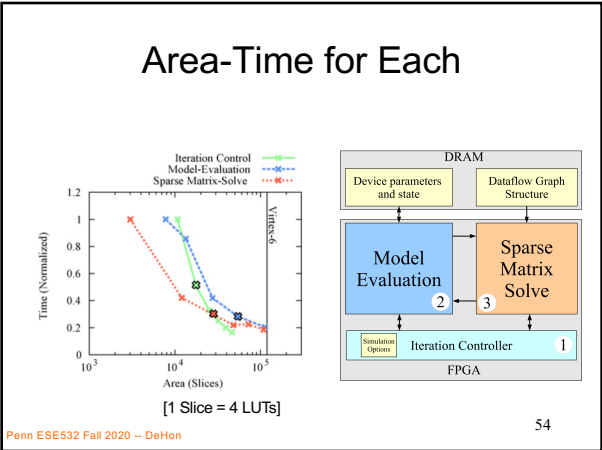
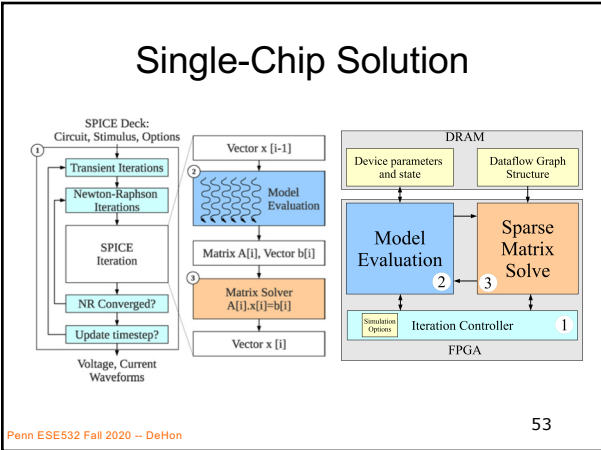
- Customized datapath controller

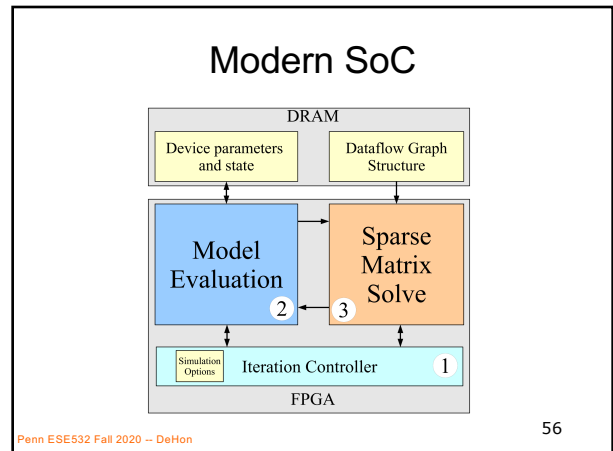
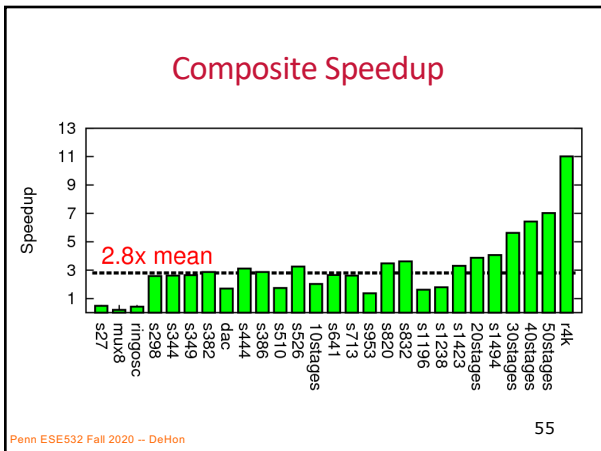
$$T_{\text{seqctrl}} = N_{\text{add}} + N_{\text{mul}} + 10 * N_{\text{divide}}$$

$$T_{\text{vlwctrl}} = \text{Max}(N_{\text{add}}/2, N_{\text{mul}}, 10 * N_{\text{divide}})$$

Penn ESE532 Fall 2020 -- DeHon

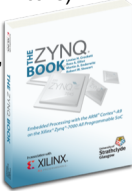
52





Class Components

Penn ESE532 Fall 2020 -- DeHon 57

- ### Class Components
- Lecture (incl. preclass exercise)
 - Slides on web before class
 - (you can print if want a follow-along copy)
 - N.B. I encourage class participation
 - Attend Synch recording; Questions (“warm” calls)
 - Reading [~1 required paper/lecture]
 - online: Canvas, IEEE, ACM, also ZynqBook, Parallel Programming for FPGAs
 - Homework
 - (1 per week due F5pm Eastern)
 - Project – open-ended (~6 weeks)
 - [Note syllabus, course admin online](#)
- 
- Penn ESE532 Fall 2020 -- DeHon 58

- ### First Half
- Quickly cover breadth
 - Metrics, bottlenecks
 - Memory
 - Parallel models
 - SIMD/Data Parallel
 - Thread-level parallelism
 - Spatial, C-to-gates
 - Line up with homeworks
- Penn ESE532 Fall 2020 -- DeHon 59

- ### Second Half
- Use everything on project
 - Schedule more tentative
 - Adjust as experience and project demands
 - Going deeper
 - Real-time
 - Reactive
 - Memory
 - Networking
 - Energy
 - Scaling
 - Chip Cost
 - Verification
- Penn ESE532 Fall 2020 -- DeHon 60

Teaming

- HW in Groups of 2
- HW: we assign
- Individual assignment writeup
- Project in Groups of 3
- Project: you propose, we review
 - Most portions group writeup
 - Few components individual writeup

Penn ESE532 Fall 2020 -- DeHon

61

Office & Lab Hours

- Andre: T 4:15pm—5:30pm Zoom
 - See canvas
- Syed:
 - First: Thursday (tomorrow) 10am
 - Make sure learning-from up to date
 - Make sure on piazza will poll

Penn ESE532 Fall 2020 -- DeHon

62

C Review

- Course will rely heavily on C
 - Program both hardware and software in C
- HW1 has some C warmup problems
- Syed will hold C review
 - on Sept. 8th, 5:30pm
 - (before our next class meeting since Monday 9/7 is Labor day)

Penn ESE532 Fall 2020 -- DeHon

63

Preclass Exercise

- Motivate the topic of the day
 - Introduce a problem
 - Introduce a design space, tradeoff, transform
- Available before lecture
 - (only available for 24-48 hours; download)
 - Should work before lecture starts
- Do use calculator
 - Will be numerical examples

Penn ESE532 Fall 2020 -- DeHon

64

Synchronous Recording Timeline

- Start Zoom before 10:30am
- Start lecture at 10:35am
- Lecture until 11:55am
- (most days) stay for remaining questions
- Post lecture to canvas later in day

Penn ESE532 Fall 2020 -- DeHon

65

Feedback

- Will have anonymous feedback google forms for each lecture
 - Clarity?
 - Speed?
 - Vocabulary?
 - General comments
- Linked on syllabus
- Today's:
<https://forms.gle/qMuup5pLkwtP187x5>

Penn ESE532 Fall 2020 -- DeHon

66

Policies

- Canvas turn-in of assignments
- No handwritten work
- Due on time
 - Individual assignments only
 - 3 free late days total
- Collaboration
 - Tools – allowed
 - Designs – limited to project teams as specified on assignments
- See web page

Penn ESE532 Fall 2020 – DeHon

67

Admin

- Your action:
 - Find course web page
 - Read it, including the policies
 - Find Syllabus
 - Find homework 1
 - Find lecture slides
 - » Will try to post before lecture
 - Find reading assignments
 - Find reading for lecture 2 on canvas and web
 - ...for this lecture if you haven't already
 - Find/join piazza group for course

Penn ESE532 Fall 2020 – DeHon

68

Logistics

- Will need SD Card writer for HW7+
 - (can get \$<10 on amazon.com)

Penn ESE532 Fall 2020 – DeHon

69

Big Ideas

- Programmable Platforms
 - Key delivery vehicle for innovative computing applications
 - Reduce TTM (Time-to-Market), risk
 - More than a microprocessor
 - Heterogeneous, parallel
- Demand hardware-software codesign
 - Soft view of hardware
 - Resource-aware view of parallelism

Penn ESE532 Fall 2020 – DeHon

70

Questions?

Penn ESE532 Fall 2020 – DeHon

71