

ESE5320: System-on-a-Chip Architecture

Day 2: September 4, 2024
Analysis, Metrics, and Bottlenecks

Day 1 Daily Quiz due.
Work Preclass (on syllabus)
Lecture start 10:20am



Penn ESE5320 Fall 2024 -- DeHon

1

Today: Analysis

- How do we quickly estimate what's possible?
 - Before developing a complete solution
 - less effort than developing complete solution
- How should we attack the problem?
 - Achieve the performance, energy goals?
- When we don't like the performance we're getting, how do we understand it?
- Where should we spend our time?

Penn ESE5320 Fall 2024 -- DeHon

2

2

Today: Analysis

- Part 1: Key Terms and Concepts
 - Throughput
 - Latency
 - Bottleneck
- Part 2: Broader view
 - Bottleneck
 - Computation as a Graph, Sequence
 - Critical Path
- Part 3: Time and Space
- Part 4: Limits
 - Resource Bound
 - And Critical Path Bound
 - 90/10 Rule (time permitting)

Penn ESE5320 Fall 2024 -- DeHon

3

3

Message for Day

- Identify the **Bottleneck**
 - May be in compute, I/O, memory, data movement
- Focus and reduce/remove bottleneck
 - More resources
 - More efficient use of resources
- Repeat

Penn ESE5320 Fall 2024 -- DeHon

4

4

Latency vs. Throughput

- **Latency:** Delay from inputs to output(s)
- **Throughput:** Rate at which can produce new set of outputs
 - (alternately, can introduce new set of inputs)

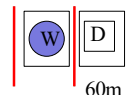
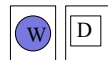
Penn ESE5320 Fall 2024 -- DeHon

5

5

Preclass Washer/Dryer Example

- 10 shirt capacity
- 1 Washer Takes 30 minutes
- 1 Dryer Takes 60 minutes
- How long to do one load of wash?
 - → Wash latency
- Cleaning Throughput?



Penn ESE5320 Fall 2024 -- DeHon

6

6

Pipeline Concurrency

- Break up the computation graph into stages
 - Allowing us to
 - reuse resources for new inputs (data),
 - while older data is still working its way through the graph
 - Before it has exited graph
 - Throughput > (1/Latency)
- Relate liquid in pipe
 - Doesn't wait for first drop of liquid to exit far end of pipe before accepting second drop

Penn ESE5320 Fall 2024 -- DeHon 7

7

Escalator




Image Source: https://commons.wikimedia.org/wiki/File:Tanforan_Target_escalator_1.JPG

Penn ESE5320 Fall 2024 -- DeHon 8

8

Escalator

- Moves 2 ft/second
- Assume for simplicity one person can step on escalator each second
- Escalator travels 30 feet (vertical and horizontal)
- Latency of escalator trip?
- Throughput of escalator: people/hour ?

Penn ESE5320 Fall 2024 -- DeHon 9

9

Bottleneck

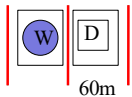
- What is the rate limiting item?
 - Resource, computation,

Penn ESE5320 Fall 2024 -- DeHon 10

10

Preclass Washer/Dryer Example

- 1 Washer Takes 30 minutes
 - Isolated throughput 20 shirts/hour
- 1 Dryer Takes 60 minutes
 - Isolated throughput 10 shirts/hour
- Where is bottleneck in our cleaning cycle?

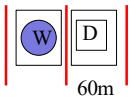


Penn ESE5320 Fall 2024 -- DeHon 11

11

Preclass Washer/Dryer Example

- 1 Washer \$500
 - Isolated throughput 20 shirts/hour
- 1 Dryer \$500
 - Isolated throughput 10 shirts/hour
- How do we increase throughput with \$500 investment?



Penn ESE5320 Fall 2024 -- DeHon 12

12

Preclass Washer/Dryer Example

- 1 Washer \$500
 - Isolated throughput 20 shirts/hour
- 2 Dryers \$500
 - Isolated single dryer throughput 10 shirts/hour

• Latency?
• Throughput?

13

Penn ESE5320 Fall 2024 -- DeHon

13

Preclass Washer/Dryer Example

- 1 Washer \$500
 - Isolated throughput 20 shirts/hour
- 2 Dryers \$500
 - Isolated single dryer throughput 10 shirts/hour

• Able to double the throughput without doubling system cost

14

Penn ESE5320 Fall 2024 -- DeHon

14

Preclass Stain Example

- 1 Washer Takes 30 minutes
 - Isolated throughput 20 shirts/hour
- 1 Dryer Takes 60 minutes
 - Isolated throughput 10 shirts/hour
- Shirt need 3 wash cycles
 - (assuming reuse single washer)

• Latency?
• Throughput?

15

Penn ESE5320 Fall 2024 -- DeHon

15

Beyond Computation

(Part 2: Broader View)

16

Penn ESE5320 Fall 2024 -- DeHon

16

Bottleneck

- May be anywhere in path
 - I/O, compute, memory, data movement

(M for Memory)

17

Penn ESE5320 Fall 2024 -- DeHon

17

Bottleneck

- Where bottleneck?

18

Penn ESE5320 Fall 2024 -- DeHon

18

Bottleneck

- Where bottleneck?

Input: Ethernet 1Gb/s (64b in 64ns)

64b → 32b in 10ns

32b → 64b in 200ns

64b → 64b In 5ns

64b every 4ns

Bus

Output: Ethernet 1Gb/s (64b in 64ns)

64b → 64b in 2ns

Penn ESE5320 Fall 2024 -- DeHon

19

Bottleneck

- Where bottleneck?

Input: Ethernet 1Gb/s (64b in 64ns)

64b → 32b in 10ns

32b → 64b in 200ns

64b → 64b In 1000ns

64b every 4ns

Bus

Output: Ethernet 1Gb/s (64b in 64ns)

64b → 64b in 2ns

Penn ESE5320 Fall 2024 -- DeHon

20

Feasibility / Limits

- First things to understand
 - Obvious limits in system?
- Impossible?
- Which aspects will demand efficient mapping?
- Where might there be spare capacity?

Penn ESE5320 Fall 2024 -- DeHon

21

Generalizing

(to more general task graphs)

Penn ESE5320 Fall 2024 -- DeHon

22

Computation as Graph

- Shown “simple” graphs (pipelines) so far
- $Y = (A+B) * (C+D)$
- $Z = (C+D) * E$

Note: HW2 ask you to draw a dataflow graph. Here's an example...more to come.

Penn ESE5320 Fall 2024 -- DeHon

23

Computation as Graph

- Nodes have multiple input/output edges
- Edges may fanout
 - Results go to multiple successors

Penn ESE5320 Fall 2024 -- DeHon

24

Computation as Sequence

- Shown “simple” graphs (pipelines) so far
- $Y=(A+B)*(C+D)$
- $Z=(C+D)*E$

$T1=A+B$
 $T2=C+D$
 $Y=T1*T2$
 $Z=T2*E$

Penn ESE5320 Fall 2024 -- DeHon 25

25

Computation as Graph

- $Y=Ax^2+Bx+C$

$T1=x*x$
 $T2=A*T1$
 $T3=B*x$
 $T4=T3+C$
 $Y=T2+T4$

Penn ESE5320 Fall 2024 -- DeHon 26

26

Computation as Graph

- Latency multiply = 1
- Latency add = 1/3
- Latency from B to output?
- Latency from x to output?
 - Through Ax^2 ?
 - Through Bx ?

Penn ESE5320 Fall 2024 -- DeHon 27

27

Delay in Graphs

- **Observe:** There are multiple paths from inputs to outputs
- Need to complete all of them to produce outputs
- Limited by longest path
- **Critical path:** longest path in the graph

Penn ESE5320 Fall 2024 -- DeHon 28

28

Computation as Graph

- Latency multiply = 1
- Latency add = 1/3
- Critical Path?

Penn ESE5320 Fall 2024 -- DeHon 29

29

Bottleneck

- Where is the bottleneck?

Penn ESE5320 Fall 2024 -- 30

30

Time and Space

(Part 3)

31

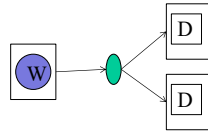
Space

- “Space” is an abstract term for physical resources
 - On VLSI (IC) chip: Area – mm² of silicon
 - On our FPGA: # of LUTs used
 - LUT = Lookup Table = Programmable Gate
 - More abstractly: # of Adders, multipliers
 - Laundry example
 - \$\$ to spend on laundry equipment
 - Physical space (sq. ft) in laundry room

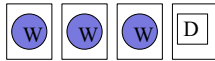
32

Space-Time

- In general, we can spend resources to reduce time
 - Increase throughput



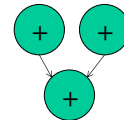
Three wash stain removal case



33

Space Time

- Computation
 - $A=x0+x1$
 - $B=x2+x3$
 - $C=A+B$
- Adder takes one cycle
- Latency on 3 adders?



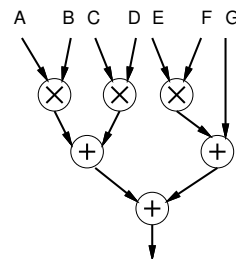
34

Space Time

- Computation
 - $A=x0+x1$
 - $B=x2+x3$
 - $C=A+B$
- Adder takes one cycle
- Could perform on one adder
 - (like one washer)
 - Reuse adder in time
 - Let cycle time be one adder delay
- Latency on one adder?

35

Computation as Graph

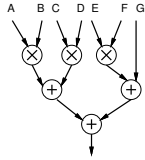


- Latency multiply = 1
- Space multiply = 3
- Latency add = 1
- Space add = 1
- (can perform add or multiple in one cycle)
- Latency and Space
 - 3 mul, 2 add

36

Schedule 3 mul, 2 add

Cycle	Mul	Mul	Mul	Add	Add
0	A*B	C*D	E*F		
1				A*B+C*D	E*F+G
2				(A*B+C*D) +(E*F+G)	

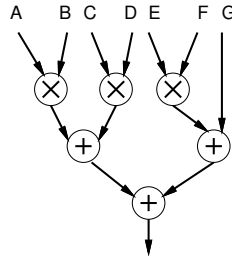


Penn ESE5320 Fall 2024 -- DeHon

37

37

Computation as Graph



- Latency multiply = 1
- Space multiply = 3
- Latency add = 1
- Space add = 1
- Latency and Space
– 1 mul, 1 add

Penn ESE5320 Fall 2024 -- DeHon

38

38

Schedule 1 mul, 1 add

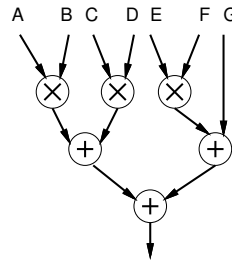
Cycle	Mul	Add
0	A*B	
1	C*D	
2	E*F	A*B+C*D
3		E*F+G
4		(A*B+C*D)+(E*F+G)

Penn ESE5320 Fall 2024 -- DeHon

39

39

Computation as Graph



- Latency multiply = 1
- Space multiply = 3
- Latency add = 1
- Space add = 1
- Latency and Space
– 2 mul, 1 add

Penn ESE5320 Fall 2024 -- DeHon

40

40

Schedule 2 mul, 1 add

Cycle	Mul	Mul	Add
0	A*B	C*D	
1	E*F		(A*B+C*D)
2			E*F+G
3			(A*B+C*D)+ (E*F+G)

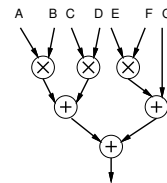
Penn ESE5320 Fall 2024 -- DeHon

41

41

A*B+C*D+E*F+G Design Points

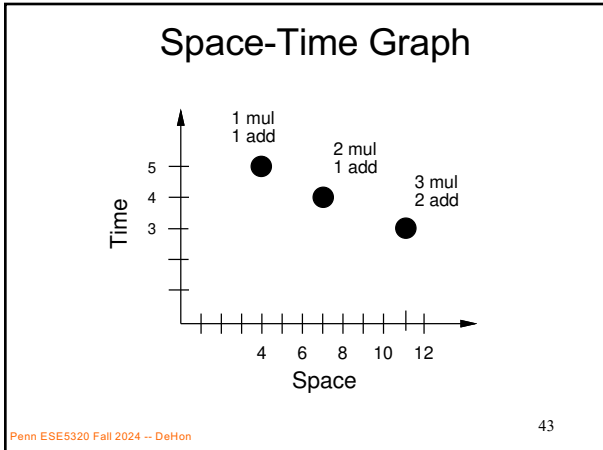
mul	add	space	latency
3	2	3*3+2*1=11	3
2	1	2*3+1*1=7	4
1	1	1*3+1*1=4	5



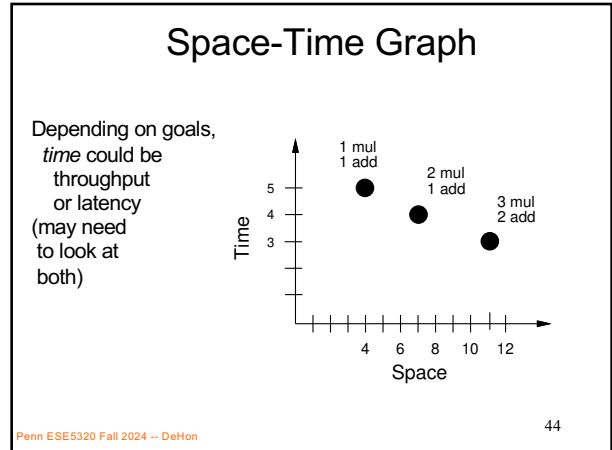
Penn ESE5320 Fall 2024 -- DeHon

42

42



43



44

Two Bounds

Part 4: Limits
 (still in Time and Space)

Penn ESE5320 Fall 2024 -- DeHon

45

Problem

- Coming up with an exact time count can be hard (human/computer time consuming)
 - Technically a hard problem
 - NP-Complete: no known non-exponential solution
- Requires reasoning about structure of graph
- Would be nice to have a quick (easy) answer on what is feasible
 - ...and what is not feasible → impossible.

Penn ESE5320 Fall 2024 -- DeHon

46

Bounds

- Establish the feasible range
 - Must be larger (or equal) than LB (lower bound)
 - Must be smaller (or equal) than UB (upper bound)
 - Solution will be between LB and UB
 - $LB \leq ActualTime \leq UB$
- Bounds in sports
 - Ball landing in-bounds or out-of bounds

Penn ESE5320 Fall 2024 -- DeHon

47

Bounds

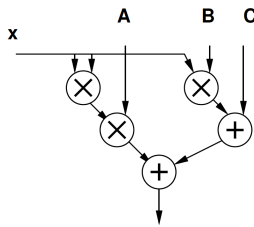
- Quick **lower** bounds (LB) can estimate
 - $LB \leq ActualTime$
- Two:
 - CP: Critical Path
 - Sometimes call it “Latency Bound”
 - RB: Resource Capacity Bound
 - Sometimes call it “Throughput Bound” or “Compute Bound”

Penn ESE5320 Fall 2024 -- DeHon

48

Critical Path Lower Bound

- Critical path assuming infinite resources
- Certainly cannot finish any faster than that
- $CP \leq ActualTime$
- Ignores resource limits



Penn ESE5320 Fall 2024 -- DeHon

49

49

Resource Bound: Single resource

- Ignore precedence (graph)
- If adds take one cycle,
 - How many additions can perform on 3 adders in 3 cycles?
- How many additions can perform in C cycles on M adders?
- If need to perform N operations, and have M adders, how many cycles?

Penn ESE5320 Fall 2024 -- DeHon

50

50

Resource Bound: Single resource

- Ignore precedence (graph)
- N operations (calculations to make)
- M operators (resource can perform calculation)
- Perform operation in one time step (cycle)
- Need at least $\lceil N/M \rceil$ time steps

Penn ESE5320 Fall 2024 -- DeHon

51

51

Resource Capacity Lower Bound

- Sum up all capacity required per resource: $TotalOps = \sum Ops$
 - E.g. number of multiplications, additions, memory lookups
- Divide by total resource (for type)
 - E.g., number of multipliers, adders, memory ports
 - $RB = \lceil TotalOps / Operators \rceil \leq ActualTime$
- Lower bound on compute
 - (best can do is pack all use densely)
 - Ignores data dependency constraints

Penn ESE5320 Fall 2024 -- DeHon

52

52

RB: Multiple Resource Types

- $RB = \text{Max}(\lceil TotalOps_1 / Operators_1 \rceil, \lceil TotalOps_2 / Operators_2 \rceil, \dots) \leq ActualTime$
- Combine Critical Path Lower Bound $\text{Max}(CP, \lceil TotalOps_1 / Operators_1 \rceil, \lceil TotalOps_2 / Operators_2 \rceil, \dots) \leq ActualTime$

Penn ESE5320 Fall 2024 -- DeHon

53

53

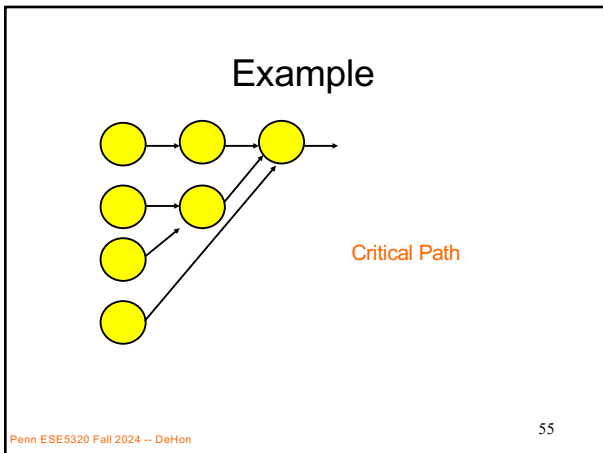
For Single Resource Type

- (and no communication time...)
- Can use to get upper bound:
 - $ActualTime \leq CP + RB$
 - (not showing here, just telling)
 - Together:
 - $\text{Max}(CP, RB) \leq ActualTime \leq CP + RB$

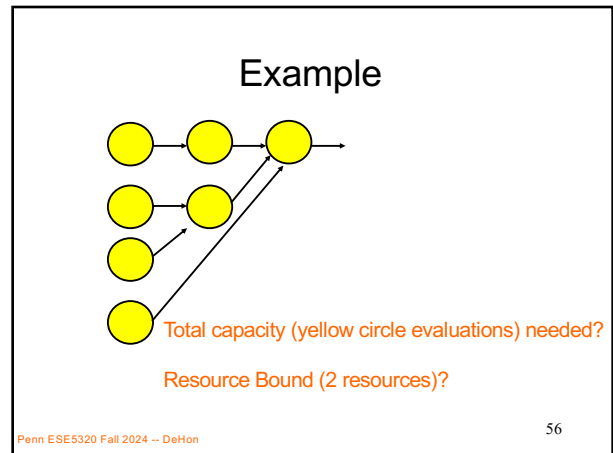
Penn ESE5320 Fall 2024 -- DeHon

54

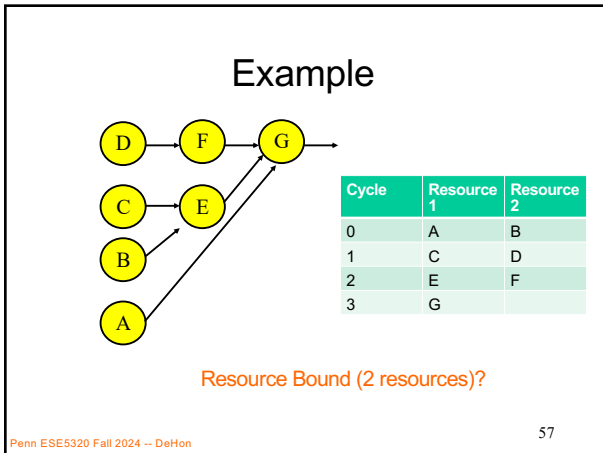
54



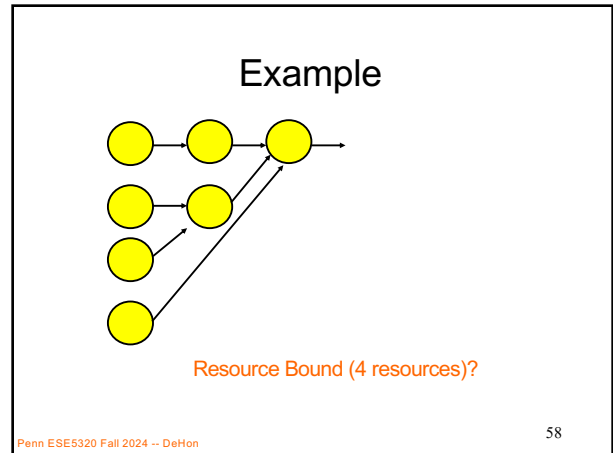
55



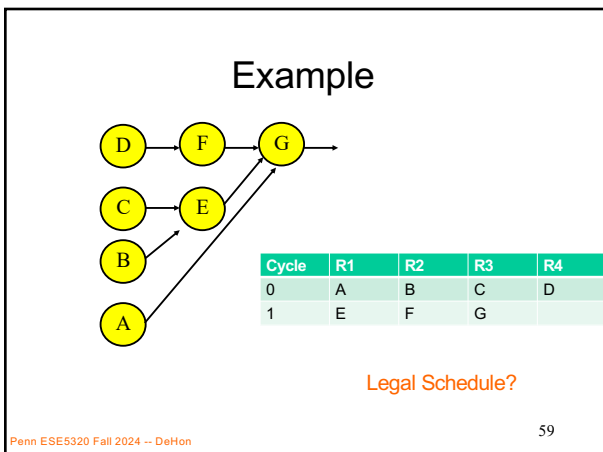
56



57



58



59

Resource Capacity Lower Bound

- Sum up all capacity required per resource: $TotalOps = \sum Ops$
 - E.g. number of multiplications, additions, memory lookups
- Divide by total resource (for type)
 - E.g., number of multipliers, adders, memory ports
 - $RB = \lceil TotalOps / Operators \rceil \leq ActualTime$
- Lower bound on compute
 - (best can do is pack all use densely)
 - Ignores data dependency constraints

Penn ESE5320 Fall 2024 -- DeHon

60

Example

Critical Path 3

Resource Bound (2 resources) $7/2=4$

Resource Bound (4 resources) $7/4=2$

Either one (CP, RB) can be limit. Check both.
In general, independent \rightarrow relation depends on task.

Penn ESE5320 Fall 2024 -- DeHon 61

61

What are the telling us

- If $CP < RB$
 - Adding resources (space) may be effective at reducing latency
- If $RB < CP$
 - Adding resources (space) will not reduce latency

Penn ESE5320 Fall 2024 -- DeHon 62

62

90/10 Rule (of Thumb)

- Observation that code is not used uniformly
- 90% of the time is spent in 10% of the code
- Knuth: 50% of the time in 2% of the code
- Implications
 - There will typically be a bottleneck
 - We don't need to optimize everything
 - We don't need to uniformly replicate space to achieve speedup
 - Not everything needs to be accelerated

Penn ESE5320 Fall 2024 -- DeHon 63

63

Big Ideas

- Identify the Bottleneck
 - May be in compute, I/O, memory, data movement
- Focus and reduce/remove bottleneck
 - More resources
 - More efficient use of resources

Penn ESE5320 Fall 2024 -- DeHon 64

64

Admin

- Diagnostic Assessment due today!
- C-Review tonight
- Reading for Day 3 on web
- HW1 due Friday
- HW2 should be out today (or tomorrow)
 - Individual assignment
- Remember feedback
- Remaining Questions?

Penn ESE5320 Fall 2024 -- DeHon 65

65