ESE5320: System-on-a-Chip Architecture

Day 2: September 7, 2022

Analysis, Metrics, and Bottlenecks

Remember masks.
Day 1 Daily Quiz due.
Work Preclass.
Lecture start 10:20am

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?

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)

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

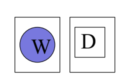
4

Latency vs. Throughput

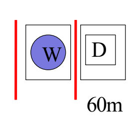
- **Latency:** Delay from inputs to output(s)
- **Throughput:** Rate at which can produce new set of outputs
  - (alternately, can introduce new set of inputs)

5

Preclass Washer/Dryer Example

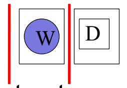


- 10 shirt capacity
- 1 Washer Takes 30 minutes
- 1 Dryer Takes 60 minutes
- How long to do one load of wash?
  - → Wash latency
- Cleaning Throughput?



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

7

Escalator

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

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 ?

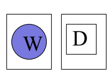
9

Bottleneck

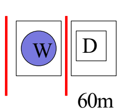
- What is the rate limiting item?
  - Resource, computation, ….

10

Preclass Washer/Dryer Example

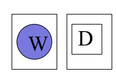


- 1 Washer Takes 30 minutes
  - Isolated throughput 20 shirts/hour
- 1 Dryer Takes 60 minutes
  - Isolated throughput 10 shirts/hour
- Where is bottleneck in our cleaning system?

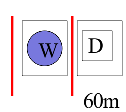


11

Preclass Washer/Dryer Example



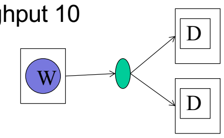
- 1 Washer $500
  - Isolated throughput 20 shirts/hour
- 1 Dryer $500
  - Isolated throughput 10 shirts/hour
- How do we increase throughput with $500 investment



12

Preclass
Washer/Dryer Example

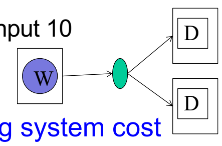
- 1 Washer $500
  - Isolated throughput 20 shirts/hour
- 2 Dryers $500
  - Isolated single dryer throughput 10 shirts/hour
- Latency?
- Throughput?



13

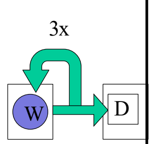
Preclass
Washer/Dryer Example

- 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

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
- Latency?
- Throughput?
  - (assuming reuse single washer)

15

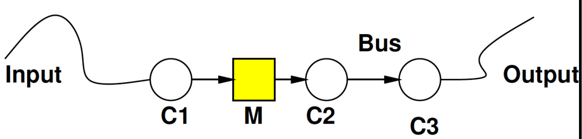
Beyond Computation

(Part 2: Broader View)

16

Bottleneck

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

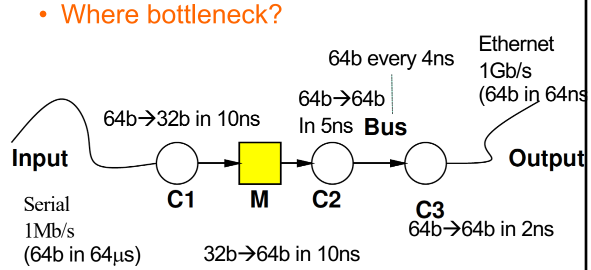


(M for Memory)

17

Bottleneck

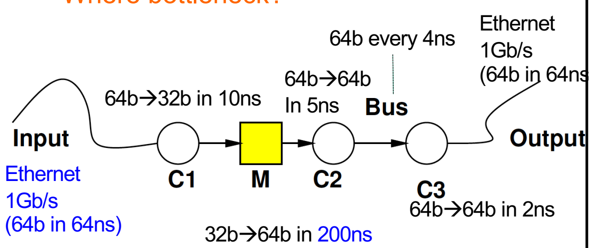
- Where bottleneck?



18

Bottleneck

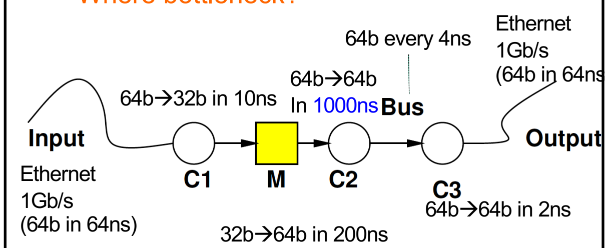
- Where bottleneck?



19

Bottleneck

- Where bottleneck?



20

Feasibility / Limits

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

21

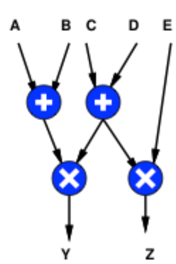
Generalizing

(to more general task graphs)

22

Computation as Graph

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

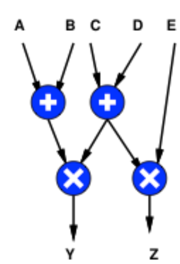


Note: HW2 ask you to draw a dataflow graph.
Here's an example…more to come.

23

Computation as Graph

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

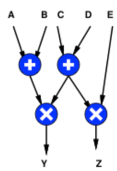


24

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

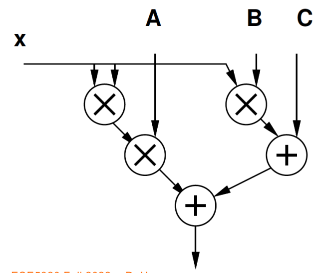


25

Computation as Graph

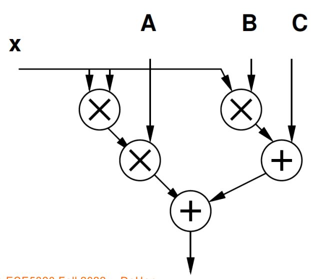
- Y=Ax²+Bx+C

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



26

Computation as Graph



- Latency multiply = 1
- Latency add = 1/3
- Latency from B to output?
- Latency from x to output?
  - Through Ax² ?
  - Through Bx ?

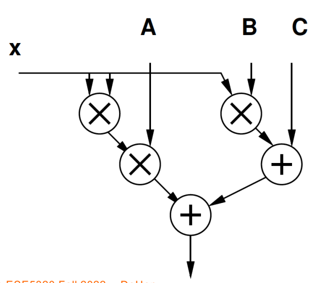
27

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

28

Computation as Graph

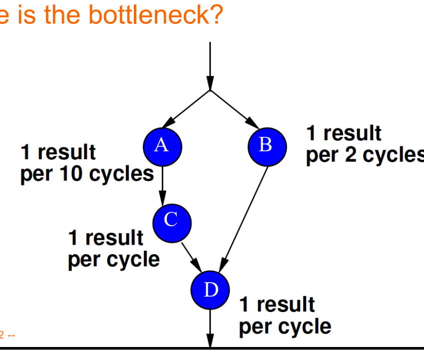


- Latency multiply = 1
- Latency add = 1/3
- Critical Path?

29

Bottleneck

- Where is the bottleneck?



30

Time and Space

(Part 3)

Penn ESE5320 Fall 2022 -- DeHon

31

31

Space

- "Space" is an abstract term for physical resources
  - On VLSI chip: Area – mm^2 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

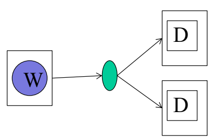
Penn ESE5320 Fall 2022 -- DeHon

32

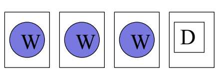
32

Space-Time

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



Three wash stain removal case



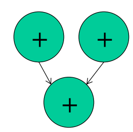
Penn ESE5320 Fall 2022 -- DeHon

33

33

Space Time

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



Penn ESE5320 Fall 2022 -- DeHon

34

34

Space Time

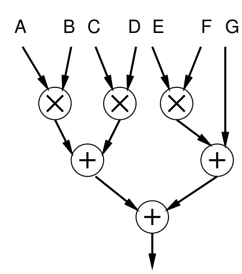
- Computation
  - A=x0+x1
  - B=x2+x3
  - C=A+B
- Adder takes one cycle
- Could perform on one adder
  - (like one washer)
  - Reuse adder in time
  - Let cycle time be one adder delay
- Latency on one adder?

Penn ESE5320 Fall 2022 -- DeHon

35

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

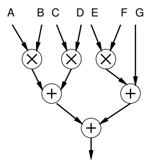
Penn ESE5320 Fall 2022 -- DeHon

36

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)	

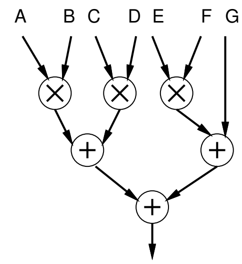


Penn ESE5320 Fall 2022 -- DeHon

37

37

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 2022 -- DeHon

38

38

Schedule 2 mul, 1 add

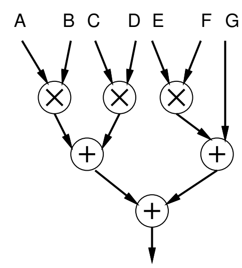
Cycle	Mul	Mul	Add
0	A*B	C*D	
1	E*F		(A*B+C*D)
2			E*F+G
3			(A*B+C*D)+ (E*F+G)

Penn ESE5320 Fall 2022 -- DeHon

39

39

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 2022 -- DeHon

40

40

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)

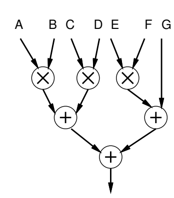
Penn ESE5320 Fall 2022 -- DeHon

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

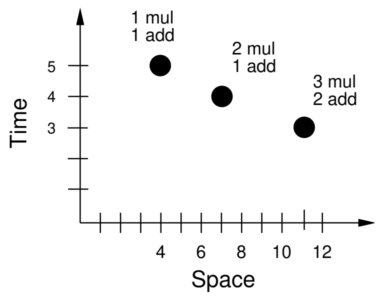


Penn ESE5320 Fall 2022 -- DeHon

42

42

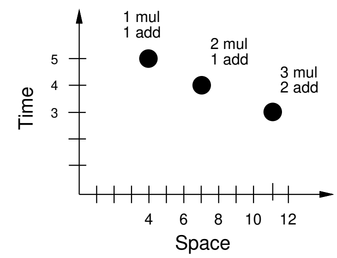
Space-Time Graph



43

Space-Time Graph

Depending on goals, *time* could be throughput or latency (may need to look at both)



44

Two Bounds

Part 4: Limits

(still in Time and Space)

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.

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

47

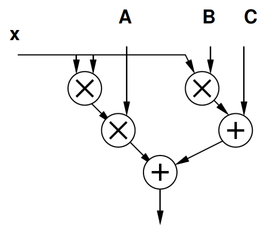
Bounds

- 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"

48

Critical Path Lower Bound

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



Resource Capacity Lower Bound

- Sum up all capacity required per resource: $\text{TotalOps} = \sum Ops$
  - 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

Multiple Resource Types

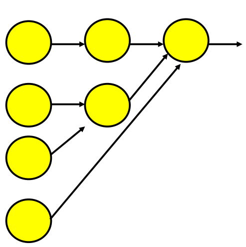
- $RB = Max(\lceil TotalOps_1/Operators_1 \rceil, \lceil TotalOps_2/Operators_2 \rceil, \ldots) \leq ActualTime$
- Combine Critical Path Lower Bound

$$Max(CP, \lceil TotalOps_1/Operators_1 \rceil, \lceil TotalOps_2/Operators_2 \rceil, \ldots) \leq ActualTime$$

For Single Resource Type

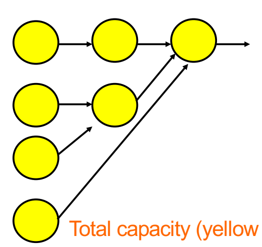
- (and no communication time…)
- Can use to get upper bound:
- $ActualTime \leq CP + RB$
- Together:
- $Max(CP, RB) \leq ActualTime \leq CP + RB$

Example



Critical Path

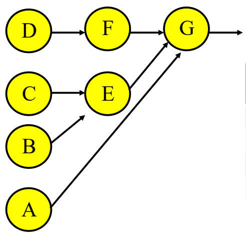
Example



Total capacity (yellow circle evaluations) needed?

Resource Bound (2 resources)?

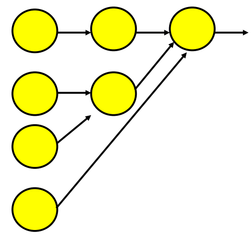
Example



Cycle	Resource 1	Resource 2
0	A	B
1	C	D
2	E	F
3	G	

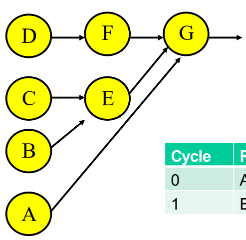
Resource Bound (2 resources)?

Example



Resource Bound (4 resources)?

Example



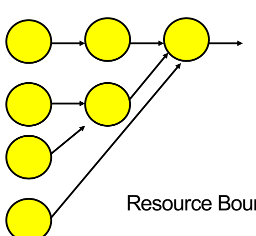
Cycle	R1	R2	R3	R4
0	A	B	C	D
1	E	F	G	

Legal Schedule?

Resource Capacity Lower Bound

- Sum up all capacity required per resource: $\text{TotalOps} = \sum Ops$
  - 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

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 → relation depends on task.

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

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 2022 -- DeHon

61

61

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 2022 -- DeHon

62

62

Admin

- Diagnostic Assessment due today!
- Reading for Day 3 on web
- HW1 due Friday
- HW2 out today
  - Individual assignment
- Remember feedback
- Remaining Questions?

Penn ESE5320 Fall 2022 -- DeHon

63

63