# ESE535: Electronic Design Automation

Day 2: January 21, 2015

Heterogeneous Multicontext Computing Array

**Work Preclass**



## Today

- Motivation
  - Energy in programmable architectures
  - Gate Evaluation Model
  - Rent's Rule and VLSI Complexity
  - Central Processor
  - Spatial Locality
- Heterogeneous multicontext
- Issues and Concerns
- Project overview

Behavioral (C, MATLAB, …)
↓ Arch. Select Schedule
RTL
↓ FSM assign, Two-level, Multilevel opt., Covering, Retiming
Gate Netlist
↓ Placement, Routing
Layout
↓
Masks

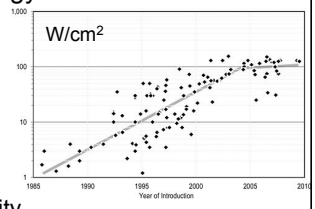


## Question

- What should an energy-efficient programmable substrate look like?

**Importance:**

- Computation/chip limited by energy-density **not** transistor capacity → dark silicon
- Energy efficiency determines battery life

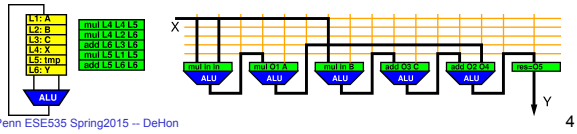



The Future of Computing Performance: Game Over or Next Level?
National Academy Press, 2011



## Question

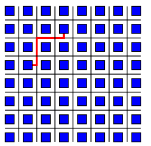
- Is it better to build compact computations
  - And read instructions and data from memory
  - Avoid sending data on long wires
- Or to build spatially distributed computations
  - And send data from producers to consumers
  - Avoid spending energy on large memories

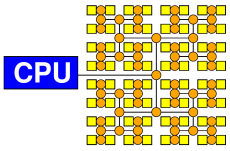





## Intuition

- Given a good spatial layout
  - It is cheaper to transmit the result of a gate to its well-placed consumers
    - average wire length $O(N^{p-0.5}) \leq O(N^{0.5})$
      - $O(1)$ for $p<0.5$
  - Than to
    - Fetch inputs from a large central memory
      - $O(N^{0.5})$

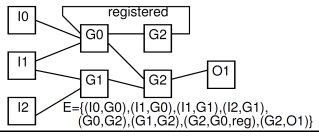






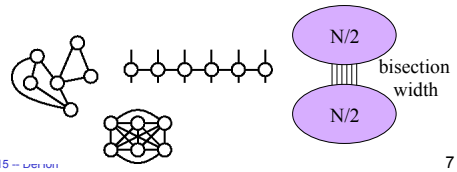

## Gate Array Evaluation Model

- For each "circuit"
  - Evaluate N k-input gates (k constant, e.g. 4)
- Assume
  - Must evaluate every gate every time
  - Every gate active (may switch)



## Bisection Width

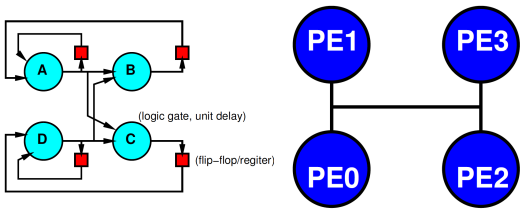
- Partition design into two equal size halves
  - Minimize wires (nets) with ends in both halves
- Number of wires crossing is **bisection width**
- lower bw = more locality






## Preclass

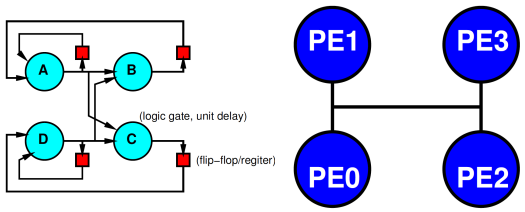
- 3a: What is bisection cut when:
  - 0:D, 1:A, 2: C, 3: B






## Preclass

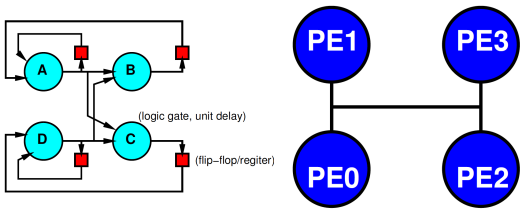
- 4a: What is bisection cut when:
  - 0:A, 1:B, 2: D, 3: C






## Minimum Bisection?

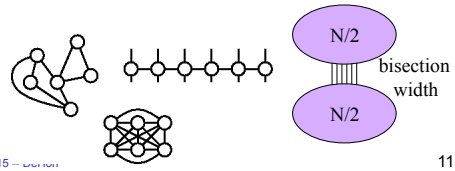
- Is there a better assignment?
  - One that achieves a smaller bisection cut?






## Bisection Width

- Partition design into two equal size halves
  - Minimize wires (nets) with ends in both halves
- Number of wires crossing is **bisection width**
- lower bw = more locality





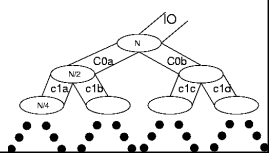

## Rent's Rule

- If we recursively bisect a graph, attempting to minimize the cut size, we typically get:

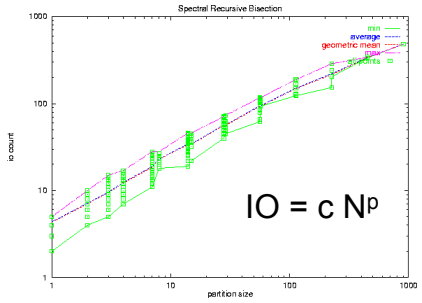
$$BW=IO = c\,N^{p}$$

  - $0 \le p \le 1$
  - $p \le 1$ means many inputs come from within a partition

[Landman and Russo, IEEE TR Computers p1469, 1971]



## Rent's Rule Bisection Data



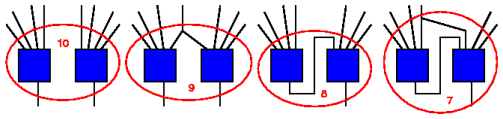

Log-log plot → straight lines represent geometric growth



## Rent and Locality

- Rent and IO quantifying locality
  - local consumption
  - local fanout

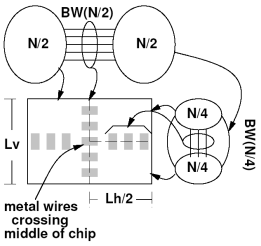
$IO = c N^p$






## VLSI Interconnect Area

- Bisection width is lower-bound on IC width
  - When wire dominated, may be tight bound
- (recursively)
- Rent's Rule tells us how big our chip must be





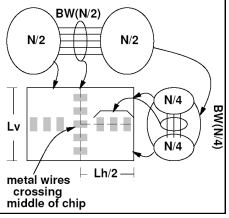

## As a function of Bisection

- $A_{chip} \geq N \times A_{gate}$
- $A_{chip} \geq N_{horizontal}\ W_{wire} \times N_{vertical}\ W_{wire}$
- $N_{horizontal} = N_{vertical} = BW = cN^p$
- $A_{chip} \geq (cN)^{2p}$
- If $p<0.5$

  $A_{chip} \propto N$
- If $p>0.5$

  $A_{chip} \propto N^{2p}$

Interconnect dominates





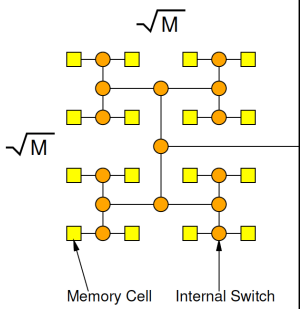

## Outline

- Gate Evaluation Model
- Rent's Rule and VLSI Complexity
- Central Processor
  - Instruction Sharing
- Spatial Locality
  - Description
  - Data
  - Instruction
    - Wire sharing



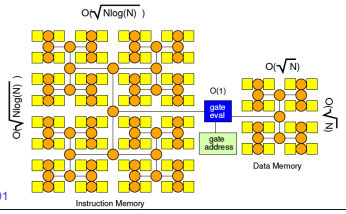
## Fully Banked Memory

- Only activate path to leaf
- $O(M^{0.5})$ wire length
- Random access must send address
  - $O(M^{0.5}\log(M))$
- Sequential access avoid address per bit
  - $O(M^{0.5})$ per bit



## Central Processor

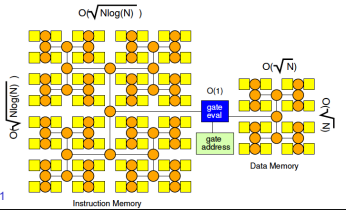
- Each instruction specifies source nodes so is O(log(N)) long
  - $O(N^{1.5}\log^{1.5}(N))$
- Read/write O(N) bits
  - $O(N^{1.5}\log(N))$






## Central Processor

- Each instruction specifies source nodes so is O(log(N)) long
  - $O(N^{1.5}\log^{1.5}(N))$
- Read/write O(N) bits
  - $O(N^{1.5}\log(N))$






## Problem 1: Description Locality

- **Assumption:** Each instruction specifies source nodes so is O(log(N)) long
- **Challenge:** Costs O(N log(N)) for description since can assume any input comes from anywhere
- **Opportunity:** exploit Rent locality

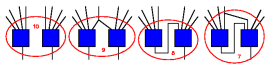


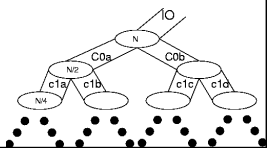
## Spatial Locality

- If we place a circuit well,
  - Most of the inputs can be "close"
- Use Rent's Rule recursive bisection

$$IO = c\,N^{p}$$

  - $p \le 1$ means many inputs come from within a partition

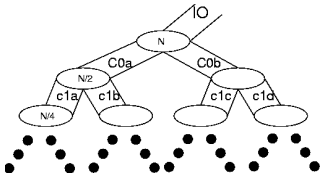







## Description Lemma

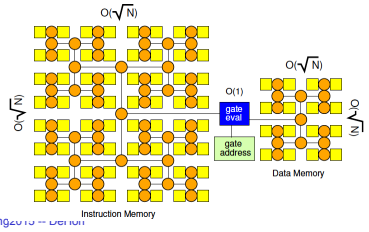
- If p<1.0, can describe computation with O(N) memory.
  - If something close by, only need to use bits proportional to subtree height






## Central Processor with Description Locality

- p<1.0, total instruction bits are O(N)
  - $O(N^{1.5})$
- Read/write O(N) bits
  - $O(N^{1.5}\log(N))$

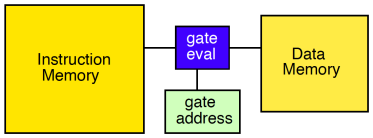


## Central Processor

- Total instruction bits
  - Read $I_{bits}$
  - From memory of size $I_{bits}$

$$I_{bits} = N\left(16 + \frac{5}{1-2^{p-1}}\right)$$

- Read/write bits
  - 5 N bits (for 4-LUT)
  - From memory of size N



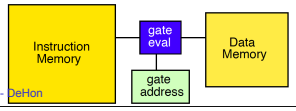



## Central Processor

- Total instruction bits
  - Read $I_{bits}$
  - $I_{bits}$-bit Memory

$$I_{bits} = N\left(16 + \frac{5}{1-2^{p-1}}\right)$$

  - p=0.7 ➔ ~N×(16+27)
- Read/write O(N) bits
  - 5 N
  - N-bit memory
  - Imem 43× larger
  - 43/5~8× more reads






## Problem 2: Instruction Sharing

- **Problem:** absolutely, instruction energy dominates data energy
- **Assumption:** every bit operator needs its own unique instruction
- **Opportunity:** share instructions across operators



## Looping

- Reuse—**share**—instructions across different data
  - Instructions not necessarily grow as N
- Example: Image or Graph
  - Fixed work per node
  - Variable number of nodes
- Let I = number of unique instructions

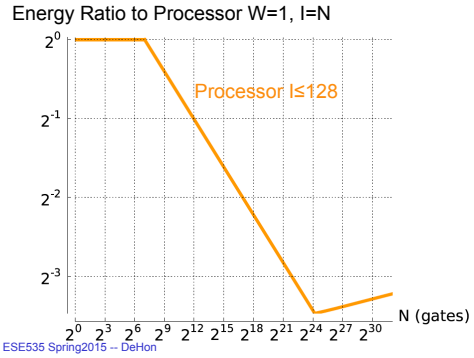


## Image Processing Example

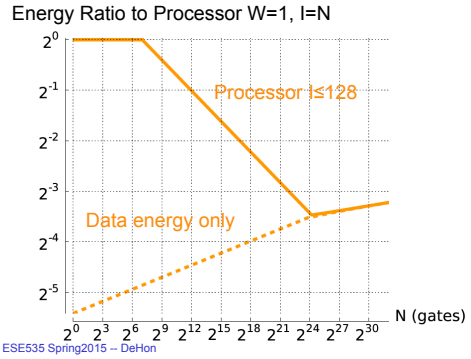
```
for (j=0;i<IMAGE_HEIGHT;j++)
  for (i=0;j<IMAGE_WIDTH;i++) {
    out[i][j]=0;
    for (wj=WIN_MIN_Y;
         wj<WIN_MAX_Y; wj++)
      for (wi=WIN_MIN_X;
           wi<WIN_MAX_X; wi++)
        out[i][j]+=window[wi+WIN_XOFF]
                         [wj+WIN_YOFF]
                  *in[i+wi][j+wj];
        // boundary condition omitted
        // for simplicity and brevity
    }
```



## Impact of Looping (I) (p=0.7)



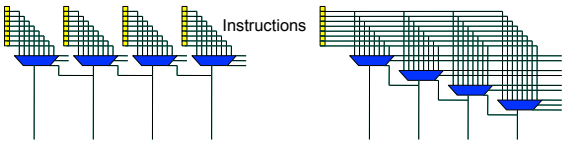
## Impact of Looping (I) (p=0.7)





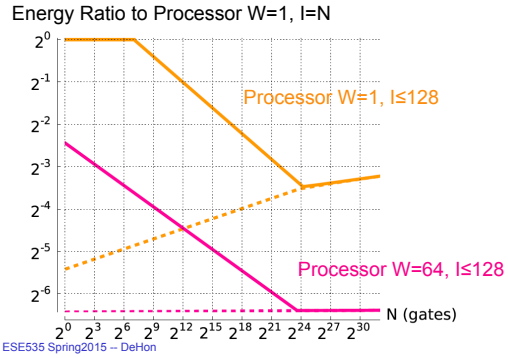

## Word Width

- Use same instruction across a set of bits
- Let W=SIMD Word width
  - E.g., 4b add
    - Share instruction bits
    - Share source/destination address bits





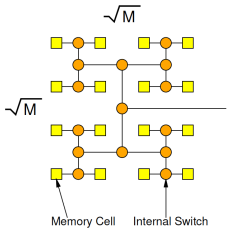

## Impact of I and W (p=0.7)






## Problem 3: Data Locality

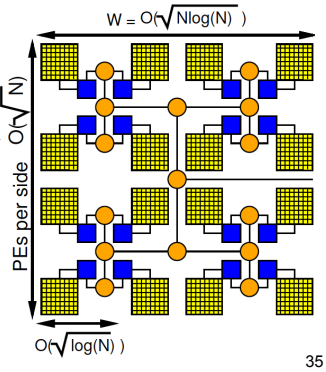
- **Problem:** Must pay $O(N^{0.5})$ for every read since data must be moved in and out of memory.
- **Opportunity:** compute local to data






## Sequential with Data Locality

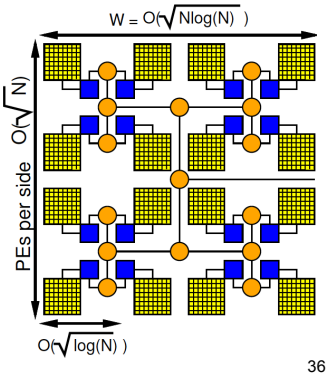
- Place for locality -- Rent Partitions
- Store data at endpoints
- Send through network from producer to consumer
- Store location at leaves – O(log(N))
- Build H-Tree to keep area to O(Nlog(N))



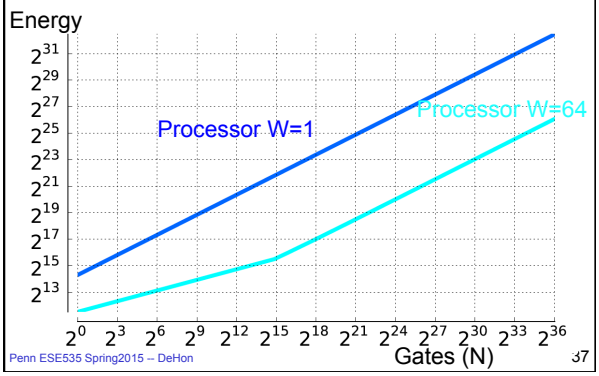



## Sequential with Data Locality

- Area = O(Nlog(N))
- Sending addresses log(N) bits at top
- Signals lower O(1)
- Only send a few over top $O(N^p)$
- $O((\log^{1.5}N)N^{p+0.5})$ for p>0.5
- **Cheaper to send where needed than to central location.**

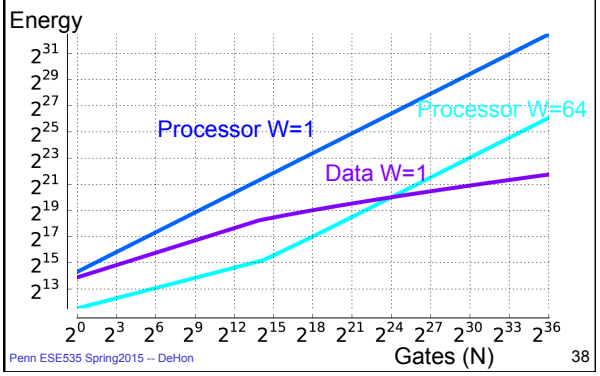


## Data Local Compare (p=0.7)



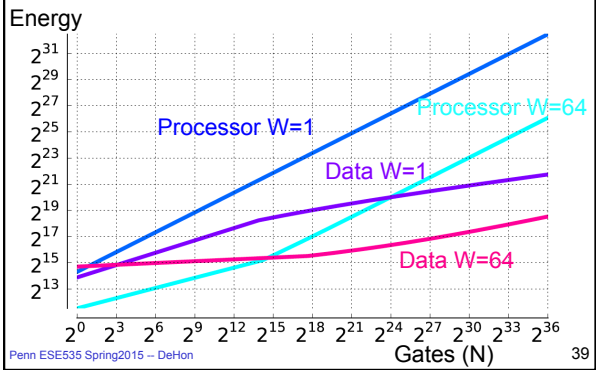



## Data Local Compare (p=0.7)





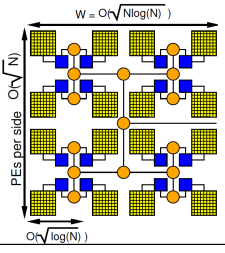

## Data Local Compare (p=0.7)






## Problem 4:

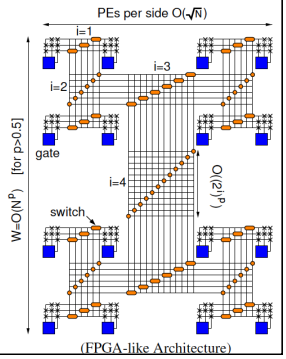
- **Problem:** Multiply energy by O(log(N)) to send an address up the tree
- **Opportunity:** store instructions local to switches
  - In tree






## Fully Spatial (FPGA)

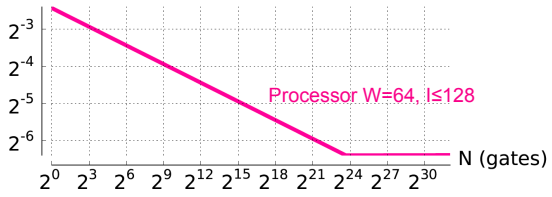
- Like an FPGA
- Each signal gets own wire
- No addresses
- Configuration local
- Area grows as $O(N^{2p})$ for p>0.5
- Energy $O(N^{2p})$ for p>0.5
  - Θ(N) for p<0.5





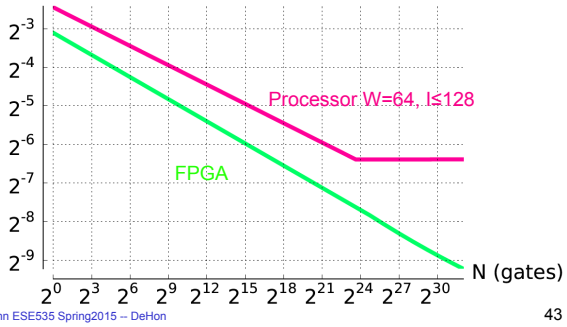

## p=0.7 Compare to Processor

Energy Ratio to Processor W=1, I=N



## p=0.7 Compare to Processor

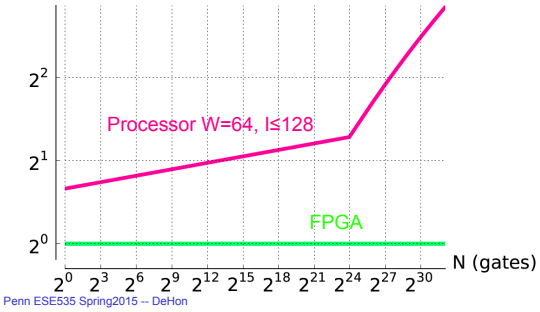
Energy Ratio to Processor W=1, I=N






## p=0.7 Compare to FPGA

Energy Processor/FPGA






## Asymptotic Energy

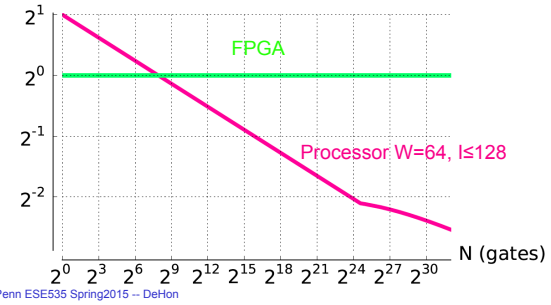
| Org | Any p | p<1.0 | 1>p>0.5 | p=0.5 | p<0.5 |
|---|---|---|---|---|---|
| Processor | $O(N^{1.5}\log^{1.5}N)$ | | $O(N^{1.5}\log N)$ Description Locality | | |
| FPGA 2-metal | | | $O(N^{2p})$ | $O(N\log^2 N)$ | $\Theta(N)$ |

Note break at p=0.75



## p=0.8 Compare to FPGA

Energy Processor/FPGA

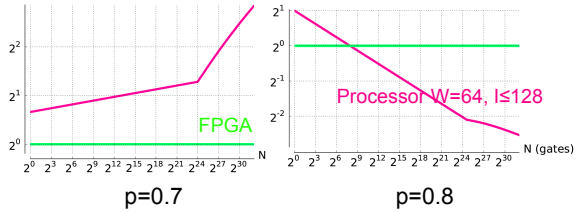





## Compare around p=0.75

- Breakpoint at p=0.75

Energy Processor/FPGA

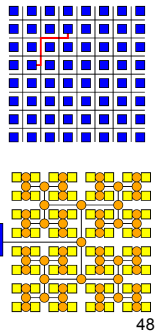



p=0.7 p=0.8



## Intuition

- Given a good spatial layout
  - It is cheaper to transmit the result of a gate to its well-placed consumers
    - Fixed metal layers: average wire length
      - p<0.5: O(1)
      - p<0.75: $O(N^{2p-1}) < O(N^{0.5})$
      - p>0.75: $O(N^{2p-1}) > O(N^{0.5})$
  - Than to
    - Fetch inputs from a large central memory
      - $O(N^{0.5})$



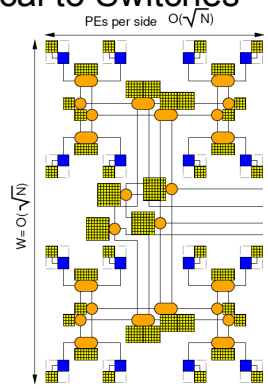
## Problem 5: Wiring Dominates

- **Problem:** When p>0.5 wiring dominates area
  → force longer wires
- **Opportunity:** Share wires
  But also, keep instructions local to switches



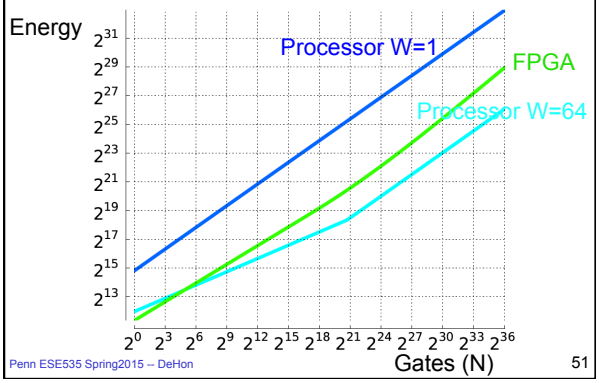
## Instructions Local to Switches

- Constant metal
- Build p<0.5 tree
- Store bits local to each tree level
- Read out of memory there
- Bits/switch differs with tree level
- Signal on wire dominates reads
- $O(N^{p+0.5})$ for p>0.5



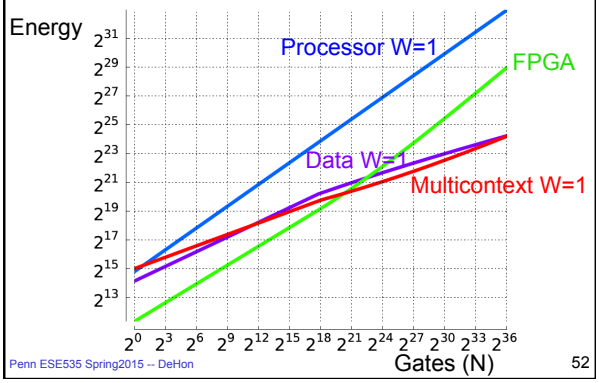



## Instructions Local (p=0.8)



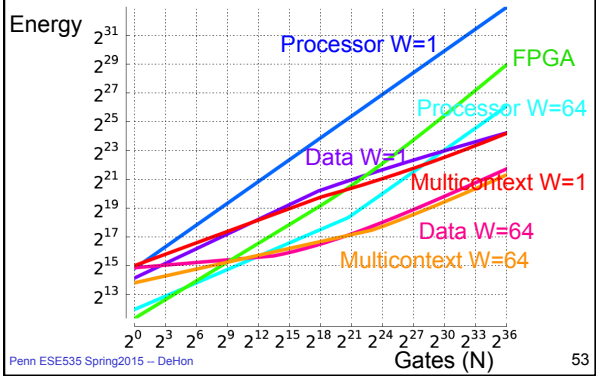



## Instructions Local (p=0.8)






## Instructions Local (p=0.8)

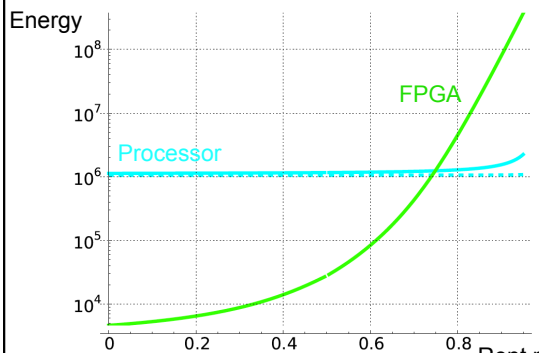





## Results: Energy

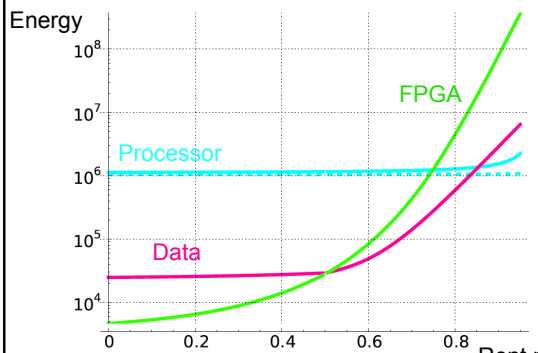
| Org | Any p | p<1.0 | 1>p>0.5 | p=0.5 | p<0.5 |
|---|---|---|---|---|---|
| Processor | $O(N^{1.5}\log^{1.5}N)$ | $O(N^{1.5}\log N)$ Description Locality | | | |
| Data Locality (Packet Switch) | | | $O(N^{p+0.5}\log^{1.5}N)$ | $O(N\log^{2.5}N)$ | $O(N\log^{1.5}N)$ |
| FPGA 2-metal | | | $O(N^{2p})$ | $O(N\log^{2}N)$ | $\Theta(N)$ |
| Multicontext | | | $O(N^{p+0.5})$ | $O(N\log N)$ | $\Theta(N)$ |

## Compare vs. p, W=64, N=$2^{24}$



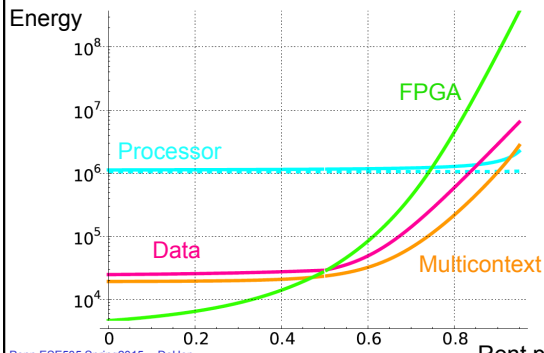



## Compare vs. p, W=64, N=$2^{24}$






## Compare vs. p, W=64, N=$2^{24}$






## Location[1], Location[2], Location[3]

- To minimize asymptotic energy, essential to exploit spatial locality to:
  1. Reduce size of **description**
  2. Minimize **data** movement energy
     - Argues against centralized processor
  3. Reduce or eliminate **instruction** energy
     - Argues for configuration
       - Local to the resources controlled



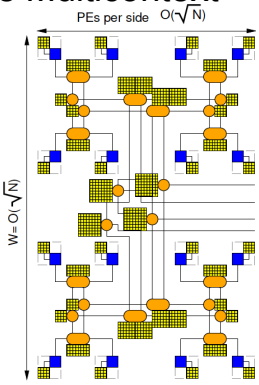
## Absolute Energy

- To minimize absolute energy, **essential to** share:
  1. Instructions to reduce instruction energy
  2. Wires to avoid size growth when p>0.5



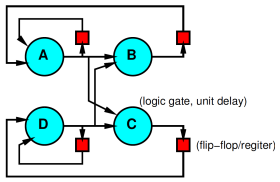
## Heterogeneous Multicontext

- Heterogeneous:
- Bits/switch differs with tree level
- More wire sharing higher in the tree



## Preclass

- 1: What are the levels on the wires?
  - (ASAP scheduling for when can route outputs)



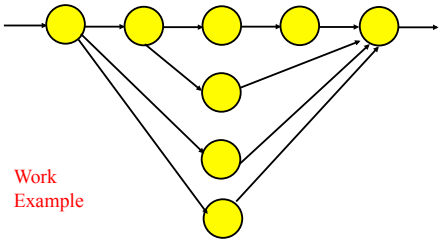



## ASAP Schedule
## As Soon As Possible (ASAP)

- For each input
  - mark input on successor
  - if successor has all inputs marked, put in visit queue
- While visit queue not empty
  - pick node
  - update time-slot based on latest input
    - Time-slot = max(time-slot-of-inputs)+1
  - mark inputs of all successors, adding to visit queue when all inputs marked



## ASAP Example

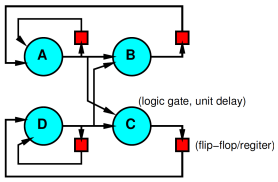


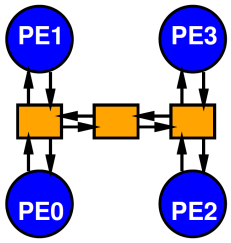
Work Example



## Interconnect and Levels

- Why relevant to interconnect?

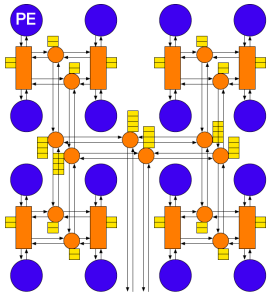





## Heterogeneous Multicontext

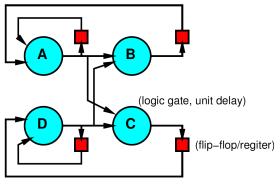
- PEs:
  - 4-LUT
  - Data memory
- Tree network
- Sequentially route by levels

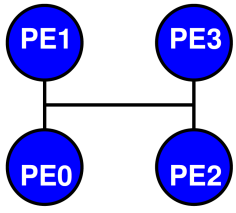





## Preclass

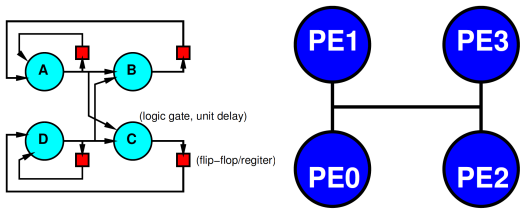
- 3b: What is bisection cut per level when:
  - 0:D, 1:A, 2: C, 3: B



## Preclass

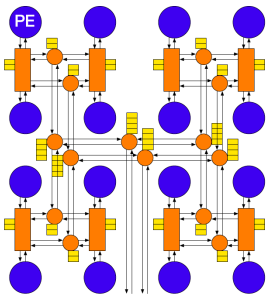
- 4b: What is bisection cut per level when:
  - 0:A, 1:B, 2: D, 3: C






## Challenges

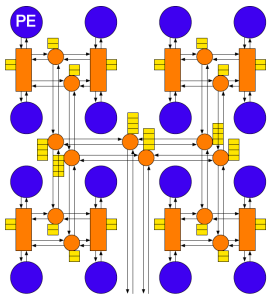
- Place to minimize wire lengths
  - Reduce bisection
- Track worst-case wiring across levels
  - Minimize bisection in worst-case level
- Schedule
  - Minimize bisection in worst-case level






## Project: Develop Mapping Tools

- Cost functions (2)
- Partitioning (3-6)
- Scheduling/ Routing (7-8)
- Open project to improve






## Today's Big Ideas

- Energy in communication
  - Wires and memories
- Save energy by exploiting locality
  - Exploit freedom in placement
- Must be careful about wire growth
  - Energy benefit to wire sharing for designs with low locality (p>0.5)
- Avoid bottlenecks:
  - Level with large bisection requirement
  - Placement, scheduling?



## Administrivia

- Assignment 1 due Thursday (tomorrow)
- Assignment 2 out
- Reading for Monday in canvas