ESE680-002 (ESE534): Computer Organization

Day 13: February 26, 2007
Interconnect 1: Requirements

Last Time

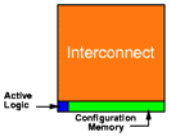
- Saw various compute blocks
- To **exploit structure** in typical designs we **need programmable interconnect**
- All reasonable, scalable structures:
  - small to moderate sized logic blocks
  - connected via programmable interconnect
- said delay across programmable interconnect is a big factor

Today

- Interconnect Design Space
- Dominance of Interconnect
- Interconnect Delay
- Simple things
  - and why they don't work

Dominant Area

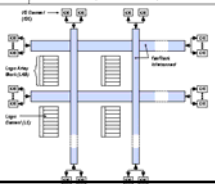
$$A_{bit\_elm} = A_{fixed} + \underbrace{N_{SW}(N_p, w, p) \cdot A_{SW}}_{\text{interconnect}} + \underbrace{\left(\frac{c}{w}\right) \cdot n_{ibits} \cdot A_{mem\_cell}}_{\text{instruction memory}} + \underbrace{d \cdot A_{mem\_cell}}_{\text{retiming memory}}$$



Function	Area (λ^2)
LUT MUX + ff	20K (generous, closer to 10K)
Programming Memory	80K (240K typical unencoded)
Interconnect	700K (for $N_p = 2048$)

Dominant Time

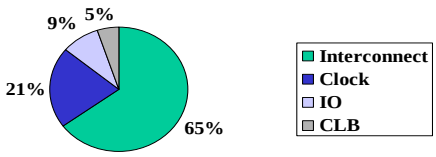
Design	Path	Total Delay	LUT Delay	Inter. %
Altera 10K130V-2	LUT-local-LUT	2.5 ns	2.1 ns	16%
	LUT-row-local-LUT	6.6 ns	2.1 ns	68%
	LUT-column-local-LUT	11.1 ns	2.1 ns	81%
	LUT-row-column-local-LUT	15.6 ns	2.1 ns	87%
	LUT-row-fanout-local-LUT (fanout)	28 ns	2.1 ns	90%



Dominant Time

Design	Path	Total Delay	LUT Delay	Inter. %
DPGA	LUT-LUT (in subarray)	3.5 ns	1.5 ns	60%
	LUT-xbar-LUT	7 ns	1.5 ns	80%
HSRA	LUT-LUT	8 ns	<2 ns	25%
	LUT-cascade	4 ns/4	<2 ns	0%
	LUT-4tree-LUT	8 ns	<2 ns	80%
	LUT-8tree-LUT	12 ns	<2 ns	83%
	LUT-16tree-LUT	16 ns	<2 ns	88%
	LUT-64tree-LUT	20 ns	<2 ns	90%

Dominant Power [Energy]



XC4003A data from Eric Kusse (UCB MS 1997)

For Spatial Architectures

- Interconnect dominant
  - area
  - energy, power
  - time
- …so need to understand in order to optimize architectures

Interconnect

- **Problem**
  - Thousands (100,000s) of independent (bit) operators producing results
    - true of FPGAs today
    - …true for *LIW, multi-uP, etc. in future
  - Each taking as inputs the results of other (bit) processing elements
  - Interconnect is **late bound**
    - don't know until after fabrication

Design Issues

- **Flexibility** -- route "anything"
  - (w/in reason?)
- **Area** -- wires, switches
- **Delay** -- switches in path, stubs, wire length
- **Energy** -- switch, wire capacitance
- **Routability** -- computational difficulty finding routes

Delay

Wiring Delay

- Delay on wire of length L_{seg}:

$$T_{seg} = T_{gate} + 0.4\ RC$$

- $C = L_{seg} \times C_{sq}$
- $R = L_{seg} \times R_{sq}$

$$T_{seg} = T_{gate} + 0.4\ C_{sq} \times R_{sq} \times L_{seg}^2$$

Wire Numbers

- R_{sq} = 0.17 Ω/sq. (@45nm)
  - from ITRS:Interconnect ρ=2.2μm-cm (Cu)
    - Conductor effective resistance
    - A/R (aspect ratio) ~ 1.8 [Table 80a, ITRS2005]
- $C_{sq} = 7 \times 10^{-18}$F/sq.
- $R_{sq} \times C_{sq} \approx 10^{-18}$ s
- T_{gate} = 30 ps, 5 ps ?
  - [ITRS2005 Table 40a, τ=0.4ps x 13.5➔5.4ps]
- Chip: 7mm side, 70nm sq. (45nm process)
  - 10^5 squares across chip

Wiring Delay

- Wire Delay

$$T_{seg} = T_{gate} + 0.4\ C_{sq} \times R_{sq} \times L_{seg}^2$$

$$T_{seg} = 30ps + 0.4\ 10^{-18}\ s \times 10^{10}$$

$$T_{seg} = 30ps + 4ns \approx 4ns$$

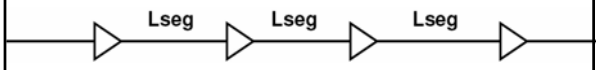
….even if 5ps

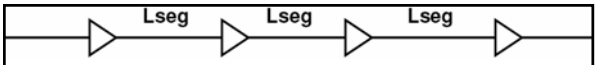
Buffer Wire

- Buffer every L_{seg}
- $T_{cross} = (L_{cross}/L_{seg})\ T_{seg}$

$$T_{cross} = (L_{cross}/L_{seg})\ (T_{gate} + 0.4\ C_{sq} \times R_{sq} \times L_{seg}^2)$$

$$= (L_{cross})\ (T_{gate}/L_{seg} + 0.4\ C_{sq} \times R_{sq} \times L_{seg})$$





Opt. Buffer Wire

- $T_{cross} = (L_{cross})\ (T_{gate}/L_{seg} + 0.4\ C_{sq} \times R_{sq} \times L_{seg})$
- Minimize: $d(T_{cross}) / d(L_{seg}) = 0$
  - $0 = (L_{cross})\ (-T_{gate}/L_{seg}^2 + 0.4\ C_{sq} \times R_{sq})$
  - $T_{gate} = 0.4\ C_{sq} \times R_{sq}\ L_{seg}^2$

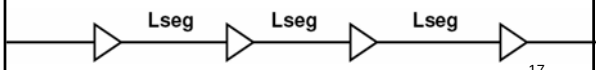
Optimization Point

- Optimized:

$$T_{cross} = (L_{cross}/L_{seg})\ (T_{gate} + 0.4\ C_{sq} \times R_{sq} \times L_{seg}^2)$$

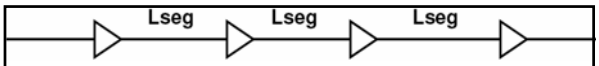
$$T_{gate} = 0.4\ C_{sq} \times R_{sq}\ L_{seg}^2$$

  - Says: equalize gate and wire delay



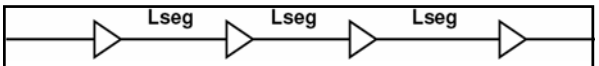
Optimal Segment Length

- $T_{gate} = 0.4\ C_{sq} \times R_{sq}\ L_{seg}^2$
- $L_{seg} = \sqrt{(T_{gate}/0.4\ C_{sq} \times R_{sq})}$
- $L_{seg} = \sqrt{(30 \times 10^{-12}\ s/0.4 \times 10^{-18}\ s)}$
  - Or $L_{seg} = \sqrt{(5 \times 10^{-12}\ s/0.4 \times 10^{-18}\ s)}$
- $L_{seg} \approx \sqrt{(10^8)} \approx 10^4$ sq.
  - Or $L_{seg} \approx \sqrt{(10^7)} \approx 3.5 \times 10^3$ sq.



Buffered Delay

- Chip: 7mm side, 70nm sq. (45nm process)
  - 10^5 squares across chip
- $L_{seg} \approx 10^4$ sq. (3.5×10^3 sq.)
- 10 segments:
  - Each of delay 2 T_{gate}
  - T_{cross} = 20×30ps = 600ps **Compare:** 4ns
  - T_{cross} = 2×30×5ps = 300ps



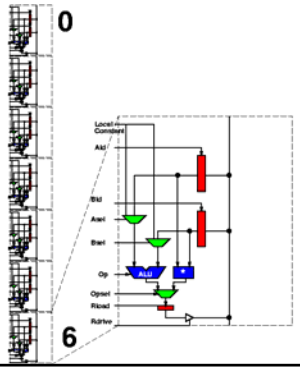
Implications

- 10 segments:
  - Each of delay 2 T_{gate}
  - T_{cross} = 20×30ps = 600ps **Compare:** 4ns
  - T_{cross} = 2×30×5ps = 300ps
- Chip crossing large compared to gate delay
  - 20× … 60×
  - Worse as gates get faster

First Attempts

(1) Shared Bus

- Familiar case
- Use single interconnect resource
- Reuse in Time
- Consequence?



Shared Bus

- Consider operation: $y=Ax^2 +Bx +C$
  - 3 mpys
  - 2 adds
  - ~5 values need to be routed from producer to consumer
- Performance lower bound if have design w/:
  - m multipliers
  - u madd units
  - a adders
  - i simultaneous interconnection busses

Resource Bounded Scheduling

- Scheduling in general NP-hard
  - (find optimum)
  - can approximate in O(E) time

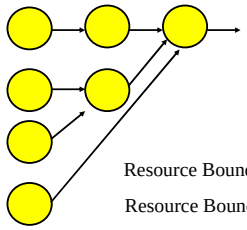
Lower Bound: Critical Path

- ASAP schedule ignoring resource constraints
  - (look at length of remaining critical path)
- Certainly cannot finish any faster than that

Lower Bound: Resource Capacity

- Sum up all capacity required per resource
- Divide by total resource (for type)
- Lower bound on remaining schedule time
  - (best can do is pack all use densely)

Example

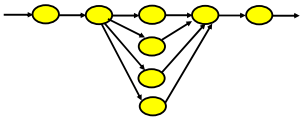


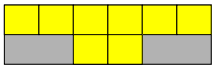
Critical Path

Resource Bound (2 resources)

Resource Bound (4 resources)

Example 2





RB = 8/2=4
LB = 5
best delay= 6

Shared Bus

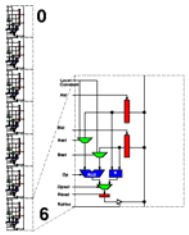
- Consider operation: $y=Ax^2+Bx+C$
  - 3 mpys
  - 2 adds
  - ~5 values need to be routed from producer to consumer
- Performance lower bound if have design w/:
  - m multipliers
  - u madd units
  - a adders
  - i simultaneous interconnection busses

Viewpoint

- Interconnect is a resource
- Bottleneck for design can be in availability of **any** resource
- Lower Bound on Delay:
  Logical Resource / Physical Resources
- May be worse
  - Dependencies (critical path bound)
  - ability to use resource

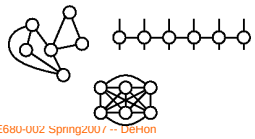
Shared Bus

- Flexibility (+)
  - routes everything (given enough time)
  - can be trick to schedule use optimally
- Delay (Power) (--)
  - wire length O(kn)
  - parasitic stubs: kn+n
  - series switch: 1
  - O(kn)
  - sequentialize I/B
- Area (++)
  - kn switches
  - O(n)



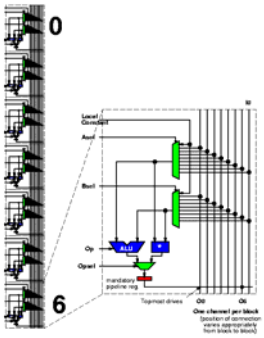
Term: Bisection Bandwidth

- Partition design into two equal size halves
- Minimize wires (nets) with ends in both halves
- Number of wires crossing is **bisection**

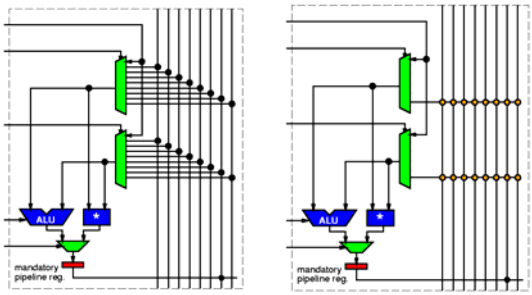


(2) Crossbar

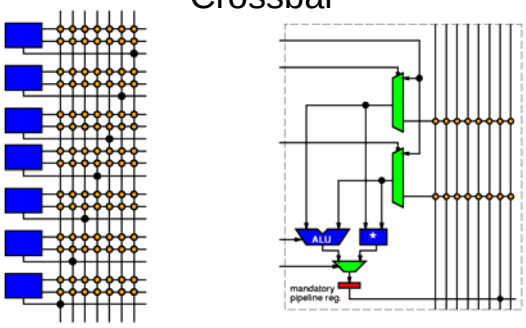
- Avoid bottleneck
- Every output gets its own interconnect channel



Crossbar

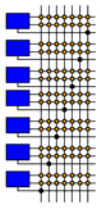


Crossbar



Crossbar

- Flexibility (++)
  - routes everything (guaranteed)
- Delay (Power) (-)
  - wire length O(kn)
  - parasitic stubs: kn+n
  - series switch: 1
  - O(kn)
- Area (-)
  - Bisection bandwidth n
  - kn^2 switches
  - $O(n^2)$



Crossbar

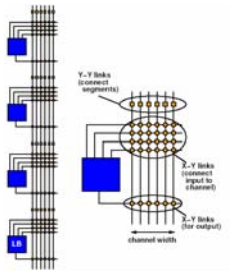
- Better than exponential
- Too expensive
  - Switch Area = $k*n^2*2.5K\lambda^2$
  - Switch Area/LUT = $k*n*2.5K\lambda^2$
  - n=1024, k=4 → 10M λ^2
- What can we do?

Avoiding Crossbar Costs

- Typical architecture trick:
  - exploit expected problem **structure**
- What structure/freedom do we have?
- We have **freedom** in operator placement
- Designs have spatial locality
- →place connected components "close" together
  - don't need full interconnect?

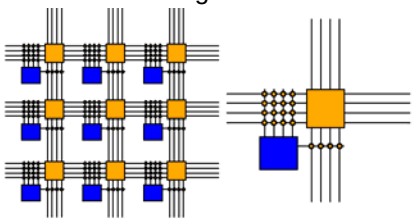
Exploit Locality

- Wires expensive
- Local interconnect cheap
- 1D versions
- What does this do to
  - Switches?
  - Delay?
- (quantify on hmwrk)



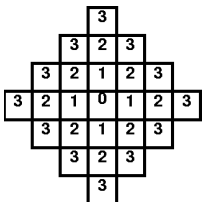
Exploit Locality

- Wires expensive
- Local interconnect cheap
- Use 2D to make more things closer
- Mesh?



Mesh Analysis

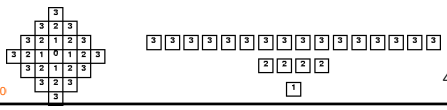
- Can we place everything close?



Mesh "Closeness"

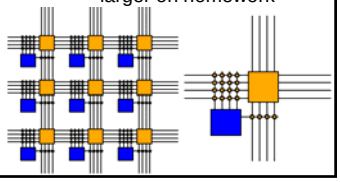
- Try placing "everything" close

Manhattan Distance	Places	Transitive Fanin
1	4	4
2	8	16
3	12	64
⋮	⋮	⋮
n	$4n$	4^n



Mesh Analysis

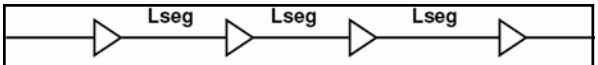
- Flexibility - ?
  - Ok w/ large w
- Delay (Power)
  - Series switches
    - 1--√n
  - Wire length
    - w--w√n
  - Stubs
    - O(w)--O(w√n)
- Area
  - Bisection BW -- w√n
  - Switches -- O(nw)
  - $O(w^2n)$
  - larger on homework



Mesh

- Plausible
- …but What's w
- …and how does it grow?

Returning to Delay

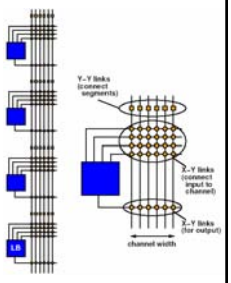


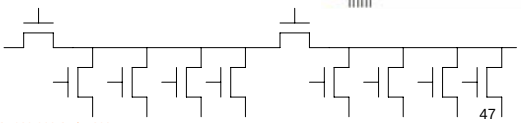
Buffered Delay

- Chip: 7mm side, 70nm sq. (45nm process)
  - 10^5 squares across chip
- $L_{seg} \approx 10^4$ sq.
- 10 segments:
  - Each of delay 2 T_{gate}
  - T_{cross} = 20×30ps = 600ps (300ps)
  - **Compare:** 4ns

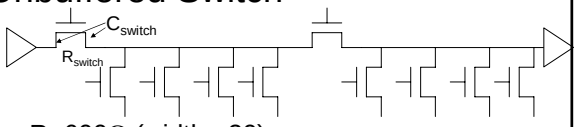
…But

- These aren't just wires
- What else?
- May go through switches
- Have switch loads



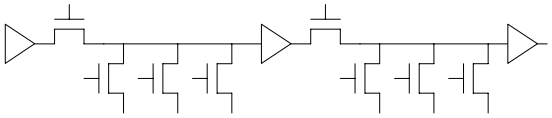


Unbuffered Switch



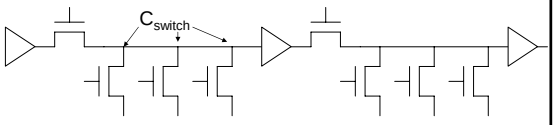
- R~600Ω (width ~20)
  - About 3600 squares? [0.17 Ω/sq.]
- $C \sim 5 \times 10^{-16}$F
  - About 100 squares?
- Not lumped ~2x worse
- Together contribute roughly 800 squares
  - $\sqrt{(2RC/(R_{sq}C_{sq}))}$
  - Vs. 10^4 or 3×10^3 sq. / rebuffer

Buffered Switch



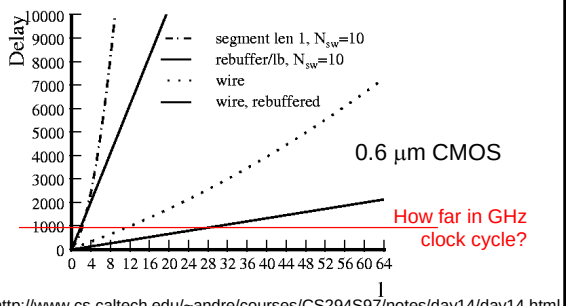
- Pay T_{gate} at each switch
- Slows down relative to
  - Optimally buffered wire
  - Unbuffered switch
- …when placed too often

Stub Capacitance



- Every untaken switch touching line
- C~2.5×10^{-16}F
  - About 50 squares
- …and lumped so 2×

Delay through Switching



http://www.cs.caltech.edu/~andre/courses/CS294S97/notes/day14/day14.html

Admin

- Assignment 4 due today
- Assignment 5 out today
  - Due after Spring Break
    - March 14 (Wed. after return)
- Reading for Wednesday
  - Rent's Rule paper (on web)

Big Ideas [MSB Ideas]

- Interconnect Dominant
  - power, delay, area
- Can be bottleneck for designs
- Can't afford full crossbar
- Need to exploit locality
- Can't have everything close