Previously

- Instruction Space Modeling
  - huge range of densities
  - huge range of efficiencies
  - large architecture space
  - modeling to understand design space

Today

- Look at Programmable Compute Blocks
- Specifically LUTs
- Introduce recurring theme (methodology):
  - define parameterized space
  - identify costs and benefits
  - look at typical application requirements
  - compose results, try to find best point

Compute Function

- What do we use for “compute” function?
  - Any Universal
    - NAND
    - ALU
    - LUT

Lookup Table

- Load bits into table
  - $2^N$ bits to describe
  - $\rightarrow 2^N$ different functions

- Table translation
  - performs logic transform
We could...

• Just build a large memory = large LUT
• Put our function in there
• What’s wrong with that?

How big is a k-LUT?

• k-input, 1-output?
• k-input, m-output?

FPGA = Many small LUTs

Alternative to one big LUT

What’s best to use?

• Small LUTs
• Large Memories

• ...small LUTs or large LUTs
• Continuum question: how big should our memory blocks used to perform computation be?

Start to Sort Out: Big vs. Small Luts

• Establish equivalence
  – how many small LUTs equal one big LUT?
"gates" in 2-LUT?

How Much Logic in a LUT?

• Lower Bound?
  – Concrete: 4-LUTs to implement M-LUT?
• Not use all inputs?
  – 0 ... maybe 1
• Use all inputs?
  – \((M-1)/3\)

(M-1)/(k-1) for K-lut

How Much logic in a LUT?

• Upper Upper Bound?:
  – M-LUT implemented w/ 4-LUTs
  – M-LUT \(\leq 2^{M-4+(2^{M-4}-1)} \leq 2^{M-3}\) 4-LUTs

How Much?

• Combine
  – Lower Upper Bound
  – Upper Lower Bound
  – (number of 4-LUTs in M-LUT)
  \[2^{M-4} \leq n \leq 2^{M-3}\]

Memories and 4-LUTs

• For the most complex functions
  – an M-LUT has \(\sim 2^{M-4}\) 4-LUTs

  ◊ SRAM 32Kx8 \(\lambda=0.6\mu m\)
    – 170M\(\lambda^2\) (21ns latency)
    – 8\(\times 2^{11}\) =16K 4-LUTs

  ◊ XC3042 \(\lambda=0.6\mu m\)
    – 180M\(\lambda^2\) (13ns delay per CLB)
    – 288 4-LUTs

  • Memory is 50+x denser than FPGA
  • Memory is 50+x denser than FPGA
    ... and faster
Memory and 4-LUTs

- For “regular” functions?
  - 15-bit parity
    - entire 32Kx8 SRAM
    - How many 4-LUTs?
    - 5 4-LUTs
      - (2% of XC3042 ~ 3.2M$\lambda^2$ ~ 1/50th Memory)

Preclass: 16-bit Adder from Memory and 3-LUTs

- How many inputs? outputs?
- Area for single large LUT?
- How many 3-LUTs?
- Area per 3-LUT?
- LUT area to implement adder with 3-LUTs?
  - Not include interconnect
- Ratio?

Memory and 4-LUTs

- Same 32Kx8 SRAM
  - 7b Add
    - entire 32Kx8 SRAM (largest will support)
    - 14 4-LUTs
      - (5% of XC3042, 8.8M$\lambda^2$ ~ 1/20th Memory)

LUT + Interconnect

- Interconnect allows us to exploit structure in computation
- Consider addition:
  - N-input add takes
    - 2N 3-LUTs
      - one N-output (2N)-LUT
    - N$\times$2$^{2N}$ >> 2N$\times$2$^3$
    - N=16: 16$\times$2$^{32}$ >> 2$\times$16$\times$2$^3$
    - 2$^{36}$ >> 2$^8$ \rightarrow factor of 2$^{28}$ = 256 Million

- Structure exploitation to avoid exponential costs is worth it!

LUT + Interconnect

- Interconnect allows us to exploit structure in computation
- Even if Interconnect was 99% of the area (100$\times$ logic area)
  - Would still be worth paying!
  - Add: N$\times$2$^{2N}$ >> 2N$\times$(2$^3$ $\times$ 128)
  - N=16: 16$\times$2$^{32}$ >> 2$\times$16$\times$2$^{10}$=2$^{15}$
  - \rightarrow factor of 2$^{15}$ = 2 Million
- Structure exploitation to avoid exponential costs is worth it!

Different Instance of a Familiar Concept

- The most general functions are huge
- Applications exhibit structure
  - Typical functions not so complex
- Exploit structure to optimize “common” case
LUT Count vs. base LUT size

Simple: \( \frac{M-1}{K-1} \)

Complex: \( 2^{(M-K)} \)

LUT vs. K

- DES MCNC Benchmark
  - moderately irregular

Gross Scaling Trend

Simple: \( \frac{1}{K} \)

Complex: \( \frac{1}{2^K} \)

Toronto Experiments

- Want to determine best K for LUTs
- Bigger LUTs
  - handle complicated functions efficiently
  - less interconnect overhead
- Smaller LUTs
  - handle regular functions efficiently
  - interconnect allows exploitation of compute structure
- What’s the typical complexity/structure?

Standard Systematization

1. Define a design/optimization space
   - pick key parameters
   - e.g. K = number of LUT inputs
2. Build a cost model
3. Map designs
4. Look at resource costs at each point
5. Compose:
   - Logical Resources ⊕ Resource Cost
6. Look for best design points
Toronto LUT Size

- Map to K-LUT
  - use Chortle
- Route to determine wiring tracks
  - global route
  - different channel width $W$ for each benchmark
- Area Model for K and W
  - $A_{\text{int}}$ exponential in K
  - Interconnect area based on switch count

LUT Area vs. K

Interconnect ~ 20x logic

LUT Area vs. K

- Logic Area vs. K
- Routing Area vs. K
- Total Area vs. K

Maped LUT Area

- Compose Mapped LUTs and Area Model

Total Area = \#k-LUTs \times \text{Area/k-LUT}

N.B. unusual case minimum area at $K=3$
Area vs. K (different tools)

Toronto Result
- Minimum LUT Area
  - at K=4
  - robust for different switch sizes
    * (wire widths)
    * [see graphs in paper]

Implications
Can we make more general conclusions?
- More restricted logic functions than LUTs?

Implications (Deep)
In the range the minimizes area:
- LUT area negligible compared to interconnect
- Anything less flexible than LUT will require more interconnect

Implications
Can we make more general conclusions?
- Custom? / Gate Arrays?

Delay
Delay?

- Circuit Depth in LUTs?
- Lower bound?
  - (M-input fun using K-LUTs)
- "Simple Function" \(\rightarrow\) M-input AND

1 table lookup in M-LUT
\(\log_2(M)\) lookups in K-LUT

Delay?

- M-input "Complex" function
  - Upper Bound:
    - use each k-lut as a k- \(\log_2(k)\) input mux
    - Upper Bound: \([\frac{(M-k)}{\log_2(k)}]+1\)

Will not cover in class, here if want to see additional details.

Delay?

- M-input "Complex" function
  - 1 table lookup for M-LUT
  - Lower Upper bound: \([\log_2(2^{(M-k)})]+1\)
  - \([\log_2(2^{(M-k)})]=(M-k)\log_2(2)\)

Will not cover in class, here if want to see additional details.

Delay?

- M-input "Complex" function
  - Lower Upper bound: \([\log_2(2^{(M-k)})]+1\)
  - \([\log_2(2^{(M-k)})]=(M-k)\log_2(2)\)
  - Lower Upper Bound: \([\frac{(M-k)}{\log_2(k)}]+1\)

Will not cover in class, here if want to see additional details.

Delay?

- M-input "Complex" function
  - 1 table lookup for M-LUT
  - between: \([\frac{(M-k)}{\log_2(k)}]+1\)
  - and \([\frac{(M-k)}{\log_2(k-\log_2(k))}]+1\)

Will not cover in class, here if want to see additional details.

Some Math

- \(Y=\log_2(2)\)
- \(k^Y = 2\)
- \(Y\log_2(k) = 1\)
- \(Y=1/\log_2(k)\)
- \(\log_2(2)=1/\log_2(k)\)
Delay

- **Simple**: \( \log M \)
- **Complex**: linear in \( M \)

- Both scale with \( k \) as \( 1/\log(k) \)

Circuit Depth vs. \( K \)

[Lee et al., JSSC 27(3):281—287, 1992]

LUT Delay vs. \( K \)

- How LUT delay scale with \( k \) for small LUTs?
  - \( t_{\text{LUT}} = c_0 + c_1 \times K \)

- Large LUTs:
  - add length term
  - \( c_2 \times \sqrt{2^K} \)

- Plus Wire Delay
  - \( \sim \text{area} \)

Delay vs. \( K \)

[Why not satisfied with this model?]

Delay vs. \( K \) (different tools)

[Why not satisfied with this model?]

Delay vs. \( K \) (proper critical path interconnect)

[Yan et al., FPGA 2002]

[Luu et al., FPGA 2009]
Energy

Observation

• General interconnect is expensive
• "Larger" logic blocks
  ➢ fewer interconnect crossings
  ➢ reduces interconnect delay
  ➢ get larger
  ➢ less area efficient
  • don’t match structure in computation
  ➢ get slower
  • Happens faster than modeled here due to area

Admin

• Reading
  – Today’s: classic paper…**definitely read**
  – Wed. → no **required** reading
    • Are some suggestions
• Office hours Tuesday
  – Especially if still confused about HW6
• HW6.1-2 due on Friday

Big Ideas

[MSB Ideas]

• Memory most dense programmable structure for the **most complex** functions
• Memory inefficient (scales poorly) for structured compute tasks
• Most tasks have structure
• Programmable interconnect allows us to exploit that structure

Big Ideas

[MSB-1 Ideas]

• Area
  – LUT count decrease w/ K, but slower than exponential
  – LUT size increase w/ K
    • exponential LUT function
    • empirically linear routing area
  – Minimum area around K=4

Big Ideas

[MSB-1 Ideas]

• Delay
  – LUT depth decreases with K
    • in practice closer to log(K)
  – Delay increases with K
    • small K linear + large fixed term