ESE534: Computer Organization

Day 14: March 12, 2012
Empirical Comparisons

Previously
- Instruction Space Modeling

Before Break
- Programmable compute blocks
  - LUTs, ALUs, PLAs
  - …still working on PLA assignment

Today
- What if we just built a completely custom circuit?
- What cost are we paying for programmability?
- Can we afford to build completely custom circuits?
  – Can we afford not to?

Today
- Empirical Data
  - Custom
    - Gate Array
    - Std. Cell (ASIC)
    - Full
  - FPGAs
  - Processors
  - Tasks

Preclass 1
- How big?
  - 2-LUT?
  - 2-LUT w/ Flip-flop?
  - 2-LUT w/ 4 input sources?
  - 2-LUT w/ 1024 input sources?
Empirical Comparisons

• Ground modeling in some concretes
• Start sorting out
  – custom vs. configurable
  – spatial configurable vs. temporal

Full Custom

• Get to define all layers
• Use any geometry you like
• Only rules are process design rules
• ESE570

Identify the full custom and standard cell regions on 386DX die
http://microscope.fsu.edu/chipshots/intel/386dxlarge.html

Standard Cell Area

All cells uniform height

Width of channel determined by routing

What freedom have we removed? Impact?
MPGA

- Metal Programmable Gate Array
  - Resurrected as “Structured ASICs”
- Gates pre-placed (poly, diffusion)
- Only get to define metal connections
  - Cheap (low NRE)
    - only have to pay for metal mask(s)

Structured ASIC: eASIC

- 2011
- 45nm
- 1.4M LUTs
- 500MHz?

Structured ASIC

- Maybe think about it as an FPGA with vias instead of configurable switches?
- Ratio of SRAM to via design?

What do we expect?

- Comparing density/delay/energy
  - Full custom
  - Standard Cell (ASIC)
  - MPGA / Structured ASIC
  - FPGA
  - Processor

Why it isn’t trivial?

- Different logic forms
- Interconnect
- Logic Balance
- Mix of Requirements

MPGA/SA vs. Custom?

- AMI CICC’83
  - MPGA 1.0
  - Std-Cell 0.7
  - Custom 0.5
- AMI CICC’04
  - Custom 0.6 (DSP)
  - Custom 0.8 (DPath)
- Toshiba DSP
  - Custom 0.3
- Mosaic RAM
  - Custom 0.2
- GE CICC’86
  - MPGA 1.0
  - Std-Cell 0.4–0.7
    - FF/counter 0.7
    - FullAdder 0.4
    - RAM 0.2
**Metal Programmable Gate Arrays**

- Area per Gate ($A_0$)
- Feature Size ($L$)
- Channelled Sea of Gates

**MPGAs**
- Modern -- "Sea of Gates"
- Yield 35–70%
- Maybe $5kA_0^2/gate = 1.25F_0^2/gate$?
  - (Quite a bit of variance)

**Conventional FPGA Tile**
- K-LUT (typical $k=4$)
- With optional output Flip-Flop
- Interconnect

**FPGA Table**

<table>
<thead>
<tr>
<th>Year</th>
<th>Design</th>
<th>Organization</th>
<th>Max</th>
<th>$A_0$</th>
<th>Area</th>
<th>Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>1986</td>
<td>XILINX 2K</td>
<td>CLB (4-LUT)</td>
<td>100</td>
<td>1</td>
<td>500K</td>
<td>20 ns</td>
</tr>
<tr>
<td>1988</td>
<td>XILINX 3K</td>
<td>CLB (2-4-LUT)</td>
<td>320</td>
<td>0.6</td>
<td>1.3M</td>
<td>13 ns</td>
</tr>
<tr>
<td>1992</td>
<td>XILINX 4K</td>
<td>CLB (2-4-LUT)</td>
<td>1024</td>
<td>0.6</td>
<td>1.25M</td>
<td>7 ns</td>
</tr>
<tr>
<td>1995</td>
<td>XILINX 5K</td>
<td>CLB (4-LUT)</td>
<td>484</td>
<td>0.3</td>
<td>2.25M</td>
<td>6 ns</td>
</tr>
<tr>
<td>1995</td>
<td>ALTERA 9K</td>
<td>LE (4-LUT)</td>
<td>1296</td>
<td>0.3</td>
<td>920K</td>
<td>7.5 ns</td>
</tr>
<tr>
<td>1998</td>
<td>QRC 2C</td>
<td>PLD (4-LUT)</td>
<td>900</td>
<td>0.3</td>
<td>4.3M</td>
<td>7 ns</td>
</tr>
<tr>
<td>1998</td>
<td>HSRA</td>
<td>BLB (5-LUT/2x4-LUT)</td>
<td>900</td>
<td>0.2</td>
<td>4M</td>
<td>4 ns</td>
</tr>
<tr>
<td></td>
<td>Model 4-LUT</td>
<td>2K</td>
<td>1M</td>
<td>1 ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Model 4-LUT</td>
<td>1K</td>
<td>1M</td>
<td>1 ns</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Toronto FPGA Model**

**Semi-Modern FPGAs**
- APEX 20K1500E
  - 52K LEs
  - 0.18μm
  - 24mm x 22mm
  - 1.25M$A_0^2$/LE
- XC2V1000
  - 10.44mm x 9.90mm
  - 0.15μm
  - 11,520 4-LUTs
  - 1.5M$A_0^2$/4-LUT
  - (~375K$A_0^2$/4-LUT)

[Both also have RAM in cited area]
How many gates? (Prelcass 2)

“gates” in 2-LUT

Now how many?

Which gives:
More usable gates?

More gates/unit area?

Gates Required?

Gate metric for FPGAs?
• Day11: several components for computations
  – compute element
  – interconnect:
    • space
    • time
    – instructions
• Not all applications need in same balance
• Assigning a single “capacity” number to device is an oversimplification
MPGA vs. FPGA

- **Xilinx XC4K**
  - 1.25M×²/CLB
  - 17–48 gates (26?)
  - 26–73K×²/gate net

- **Ratio**: 2–10 (5)

Adding ~2x Custom/MPGA, Custom/FPGA ~10x

FPGA vs. Structure ASIC

- **Virtex 6**
- 40nm
- 470K 6-LUTs
- Largest device
- eASIC
- 45nm
- 580K eCells
- Probably smaller die

FPGA vs. Std Cell

- **90nm**
- FPGA: Stratix II
- STMicro CMOS090
  - Standard Cell
    - Full custom layout
    - ...but by tool

MPGA vs. FPGA (Delay)

- **MPGA (SOG GA)**
  - 5K×²/gate
  - 35–70% usable (50%)  
  - 7–17K×²/gate net

- **Ratio**: 2–10 (5)

  - Altera claiming 2×
  - LSI claiming 3×  
  - For their Structured ASIC [2007]
  - 2005

- **Xilinx XC4K**
  - λ=0.6µ
  - τgd~1ns

  - 1-7 gates in 7ns
  - 2-3 gates typical
**FPGA vs. Std Cell Energy**

- 90nm
- FPGA: Stratix II
- STMicro CMOS090
- eASIC (MPGA) claim
  - 20% of FPGA power
  - (best case)

[Kuon/Rose TRCADv26n2p203–215 2007]

### Component Example

<table>
<thead>
<tr>
<th>Metric: 4 input gate-evaluations $\lambda^2/s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor: $2N_{ALU \times ALU} / \lambda^{Proc \times T_{cycle}}$</td>
</tr>
<tr>
<td>FPGA: $N_{ALU} / \lambda^{Array \times T_{cycle}}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metric: 4 input gate-evaluations $\lambda^2/s$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processor: $2N_{ALU \times ALU} / \lambda^{Proc \times T_{cycle}}$</td>
</tr>
<tr>
<td>FPGA: $N_{ALU} / \lambda^{Array \times T_{cycle}}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Year</th>
<th>Design</th>
<th>Organization</th>
<th>$\lambda$</th>
<th>$\lambda^{area}$</th>
<th>cycle</th>
<th>$90^* \mu$m</th>
</tr>
</thead>
<tbody>
<tr>
<td>1984</td>
<td>MPS</td>
<td>1 x 32</td>
<td>1.5</td>
<td>1.5</td>
<td>458</td>
<td>1.7</td>
</tr>
<tr>
<td>1987</td>
<td>MPS-V</td>
<td>1 x 32</td>
<td>1.0</td>
<td>1.0</td>
<td>458</td>
<td>1.7</td>
</tr>
<tr>
<td>1994</td>
<td>MPS</td>
<td>1 x 32</td>
<td>0.28</td>
<td>0.28</td>
<td>458</td>
<td>1.7</td>
</tr>
<tr>
<td>1992</td>
<td>Alpha</td>
<td>1 x 64</td>
<td>0.39</td>
<td>0.39</td>
<td>458</td>
<td>1.7</td>
</tr>
<tr>
<td>1995</td>
<td>Alpha</td>
<td>2 x 64</td>
<td>0.25</td>
<td>0.25</td>
<td>458</td>
<td>1.7</td>
</tr>
<tr>
<td>1996</td>
<td>Alpha</td>
<td>4 x 64</td>
<td>0.15</td>
<td>0.15</td>
<td>458</td>
<td>1.7</td>
</tr>
</tbody>
</table>

### Raw Density Summary

- Area
  - MPG 2-3x Custom
  - FPGA 5x FPGA
    - FPGA: std-cell custom ~ 15-30x
- Area-Time
  - Gate Array 6-10x Custom
  - FPGA 15-20x Gate Array
    - FPGA: std-cell custom ~ 100x
  - Processor 10x FPGA
Raw Density Caveats

- Processor/FPGA may solve more specialized problem
- Problems have different resource balance requirements
  - ...can lead to low yield of raw density

Task Comparisons

Broadening Picture

- Compare larger computations
- For comparison
  - throughput density metric: results/area-time
    - normalize out area-time point selection
    - high throughput density
      - most in fixed area
      - least area to satisfy fixed throughput target

Preclass 4

- Efficiency of 8×8 multiply on 16×16 multiplier?
Example: FIR Filtering

\[ Y_i = w_1 x_i + w_2 x_{i+1} + \ldots \]

| Architecture       | Feature          | Size (\(\mu m\)) | TAPs
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>32b RSC</td>
<td></td>
<td>0.75(\mu m)</td>
<td>0.020</td>
</tr>
<tr>
<td>16b DSP</td>
<td></td>
<td>0.65(\mu m)</td>
<td>0.057</td>
</tr>
<tr>
<td>32b RSC/DSP</td>
<td></td>
<td>0.25(\mu m)</td>
<td>0.021</td>
</tr>
<tr>
<td>64b RSC</td>
<td></td>
<td>0.18(\mu m)</td>
<td>0.064</td>
</tr>
<tr>
<td>FPGA (XCAK)</td>
<td>(Altera 8K)</td>
<td>0.30(\mu m)</td>
<td>1.9</td>
</tr>
<tr>
<td>Full Custom</td>
<td></td>
<td>0.75(\mu m)</td>
<td>3.6</td>
</tr>
<tr>
<td>(fixed coefficient)</td>
<td></td>
<td>0.60(\mu m)</td>
<td>56</td>
</tr>
</tbody>
</table>

Application metric: TAPs = filter taps multiply accumulate

Mixed Designs

- Modern FPGAs include hardwired multipliers (Virtex 25x18)

<table>
<thead>
<tr>
<th>Device</th>
<th>Configurable Logic Core</th>
<th>Configurable Logic Block</th>
<th>Block RAM</th>
<th>Look-Up Table Block</th>
<th>DSP Slice</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIRTEX4</td>
<td>15.05</td>
<td>324</td>
<td>70</td>
<td>150</td>
<td>23</td>
</tr>
<tr>
<td>VIRTEX5</td>
<td>25.05</td>
<td>500</td>
<td>100</td>
<td>250</td>
<td>33</td>
</tr>
<tr>
<td>VIRTEX6</td>
<td>35.05</td>
<td>695</td>
<td>130</td>
<td>450</td>
<td>39</td>
</tr>
</tbody>
</table>

FPGA vs. Std Cell

(revisit)

- 90nm
- FPGA: Stratix II
- STMicro CMOS090

Pleides includes hardwire multiply accumulator

Energy


FPGA vs. Std Cell

Energy (revisit)

- 90nm
- FPGA: Stratix II
- STMicro CMOS090

Simplest IIR: \( Y_i = A \times X_i + B \times Y_{i-1} \)
DES Keysearch

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Feature Size (L)</th>
<th>Area</th>
<th>Keys/Second</th>
<th>Keys</th>
</tr>
</thead>
<tbody>
<tr>
<td>DES IC</td>
<td>1.5m²</td>
<td>310k</td>
<td>0.108K</td>
<td></td>
</tr>
<tr>
<td>FPGA (Altera BK)</td>
<td>0.30m²</td>
<td>8118B (0.30m²)</td>
<td>900k</td>
<td>0.00086</td>
</tr>
<tr>
<td>RISC</td>
<td>0.30m²</td>
<td>1.853²</td>
<td>4.1K</td>
<td>0.00023</td>
</tr>
</tbody>
</table>

<http://www.cs.berkeley.edu/~iang/isaac/hardware/>

DNA Sequence Match

- **Problem**: “cost” of transform $S_1 \rightarrow S_2$
- **Given**: cost of insertion, deletion, substitution
- **Relevance**: similarity of DNA sequences
  - evolutionary similarity
  - structure predict function
- **Typically**: new sequence compared to large database

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Feature Size (L)</th>
<th>Area</th>
<th>Cell Updates per Second</th>
<th>cu</th>
</tr>
</thead>
<tbody>
<tr>
<td>Custom</td>
<td>2.0m²</td>
<td>270M²</td>
<td>500M 1.9</td>
<td></td>
</tr>
<tr>
<td>FPGA (SPLASH 2)</td>
<td>0.60m²</td>
<td>43G²</td>
<td>3,000M 0.070</td>
<td></td>
</tr>
<tr>
<td>(SPLASH)</td>
<td>0.60m²</td>
<td>33G²</td>
<td>370M 0.012</td>
<td></td>
</tr>
<tr>
<td>RISC (SparStation 1)</td>
<td>0.75m²</td>
<td>273M²</td>
<td>0.87M 0.0032</td>
<td></td>
</tr>
<tr>
<td>(SparStation 10)</td>
<td>0.40m²</td>
<td>1.6G²</td>
<td>1.2M 0.00075</td>
<td></td>
</tr>
</tbody>
</table>

N.B. includes memory area for SPLASH

How do various architecture degrade from peak?

- FPGA?
- Processor?
- Custom?

Degrade from Peak

- Long path length $\rightarrow$ not run at cycle
- Limited throughput requirement
  - bottlenecks elsewhere limit throughput req.
- Insufficient interconnect
- Insufficient retiming resources
  (bandwidth)

Degrade from Peak: FPGAs
Degrade from Peak: Processors

- Ops w/ no gate evaluations (interconnect)
- Ops use limited word width
- Stalls waiting for retimed data

Degrade from Peak: Custom/MPGA

- Solve more general problem than required
  - (more gates than really need)
- Long path length
- Limited throughput requirement
- Not needed or applicable to a problem

Degrade Notes

- We’ll cover these issues in more detail as we get into them later in the course

Admin

- Grading through 6.1-2 done during break
  - Check feedback
- HW6.3-4 due Wednesday
- HW7 out
- Wednesday Reading on web
  - Classic Paper

Big Ideas
[MSB Ideas]

- Raw densities:
  custom:ga:fpga:processor
  - 1:5:100:1000
  - close gap with specialization