ESE534: Computer Organization

Day 11: March 1, 2010
Instruction Space Modeling

Last Time
• Instruction Requirements
• Instruction Space

Architecture Instruction Taxonomy

<table>
<thead>
<tr>
<th>Control Threads (PCs)</th>
<th>Pinsts/PC</th>
<th>depth</th>
<th>width</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>FPGA</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1024</td>
<td>32</td>
<td>Scalar Processor (RISC)</td>
</tr>
<tr>
<td>1</td>
<td>N</td>
<td>D</td>
<td>W</td>
<td>VLIW (superscalar)</td>
</tr>
<tr>
<td>1</td>
<td>W</td>
<td>N</td>
<td>M</td>
<td>SIMD, GPU, Vector</td>
</tr>
<tr>
<td>4</td>
<td>N</td>
<td>D</td>
<td>W</td>
<td>MIMD</td>
</tr>
</tbody>
</table>

Architecture Taxonomy

<table>
<thead>
<tr>
<th>PCs</th>
<th>Pinsts/PC</th>
<th>depth</th>
<th>width</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>FPGA</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1024</td>
<td>32</td>
<td>Scalar Processor (RISC)</td>
</tr>
<tr>
<td>1</td>
<td>N</td>
<td>D</td>
<td>W</td>
<td>VLIW (superscalar)</td>
</tr>
<tr>
<td>1</td>
<td>W</td>
<td>N</td>
<td>M</td>
<td>SIMD, GPU, Vector</td>
</tr>
<tr>
<td>4</td>
<td>N</td>
<td>D</td>
<td>W</td>
<td>MIMD</td>
</tr>
</tbody>
</table>

Today
• Model Architecture from Instruction Parameters
  – implied costs
  – gross application characteristics

Quotes
• *If it can’t be expressed in figures, it is not science; it is opinion.*  -- Lazarus Long
Modeling

• Why do we model?

Motivation

• Need to understand
  – How costly is a solution
    • Big, slow, hot, energy hungry….
  – How compare to alternatives
  – Cost and benefit of flexibility

What we really want:

• Complete implementation of our application
• For each architectural alternatives
  – In same implementation technology
  – w/ multiple area-time points

Reality

• Seldom get it packaged that nicely
  – much work to do so
  – technology keeps moving
• We must deal with
  – estimation from components
  – technology differences
  – few area-time points

Modeling Instruction Effects

• Restrictions from “ideal”
  + save area and energy
    – limit usability (yield) of PE
      • May cost more energy, area in the end…
  • Want to understand effects
    – area model [today] (energy model on HW5)
    – utilization/yield model

Preclass

• Energies?
• 16-bit on 32-bit?
  – Sources of inefficiency?
• 8-bit operations per 16-bit operation?
• 16-bit on 8-bit?
  – Sources of inefficiency?
Efficiency/Yield Intuition

- What happens when
  - Datapath is too wide?
  - Datapath is too narrow?
  - Instruction memory is too deep?
  - Instruction memory is too shallow?

Computing Device

- Composition
  - Bit Processing elements
  - Interconnect: space
  - Interconnect: time
  - Instruction Memory

Tile together to build device

Relative Sizes

- Bit Operator 10-20Kλ²
- Bit Operator Interconnect 500K-1Mλ²
- Instruction (w/ interconnect) 80Kλ²
- Memory bit (SRAM) 1-2Kλ²

Model Area

\[
A_{bit.elm} = A_{fixed} + N_{SW}(N_p,w,p) \cdot A_{SW} \cdot \frac{A_{mem.cell}}{n_{bits}} + \frac{c}{d} \cdot A_{mem.cell} \cdot \text{retiming memory}
\]
Architectures Fall in Space

Calibrate Model

FPGA model \( w = 1, d = c = 1, k = 4 \) 880K\( \chi \)
Xilinx 4K 630K\( \chi \)
Altera 8K 930K\( \chi \)

SIMD model \( w = 1000, c = 0, d = 64, k = 3 \) 170K\( \chi \)
Abacus 190K\( \chi \)

Processor model \( w = 32, d = 32, c = 1024, k = 2 \) 2.6M\( \chi \)
MIPS-X 2.1M\( \chi \)

Peak Densities from Model

• Only 2 of 4 parameters
  – small slice of space
  – 100x density across

• Large difference in peak densities
  – large design space!

Architectural parameters \( \rightarrow \) Peak Densities

Efficiency

• What do we really want to maximize?
  – Not peak, “guaranteed not to exceed” performance, but…
  – Useful work per unit silicon [per Joule]

• Yield Fraction / Area
• (or minimize (Area/Yielded performance))
Efficiency

- For comparison, look at relative efficiency to ideal.
- Ideal = architecture exactly matched to application requirements
- Efficiency = $A_{\text{ideal}}/A_{\text{arch}}$
- $A_{\text{arch}} = \text{Area Op/Yield}$

Width Mismatch Efficiency Calculation

$$E = \frac{\text{Area(Task – on – matched – Architecture)}}{\text{Area(Task – on – this – Architecture)}}$$

$$E = \frac{W_{\text{task}} \times A_{\text{bitlm|w=w_{\text{task}}}}}{W_{\text{arch}} \times \left[ \frac{W_{\text{task}}}{W_{\text{arch}}} \right] \times A_{\text{bitlm|w=w_{\text{arch}}}}}$$

Efficiency: Width Mismatch

c=1, 16K PEs

Efficiency for Preclass

$$E = \frac{\text{Energy(Task – on – matched – Architecture)}}{\text{Energy(Task – on – this – Architecture)}}$$

- Efficiency of 16-bit on 32-bit arch?
- Efficiency of 16-bit on 8-bit arch?

Application vs. Architecture

- $W_{\text{task}}$ vs. $W_{\text{arch}}$
- Path Length vs. Context Depth

Path Length

- How many primitive-operator delays before can perform next operation?
  - Reuse the resource
Reuse

Pipeline and reuse at primitive-operator delay level.

How many times can I reuse each primitive operator?

Path Length: How much sequentialization is allowed (required)?

E.g. Want meet 30ns real time rate with 1.5ns cycle time, can afford to issue 15 sequential ops.

Context (Instruction) Depth

Efficiency with fixed Width

Path Length: How much sequentialization is allowed (required)?

Context Depth

w=1, 16K PEs

Ideal Efficiency (different model)

Two resources here:
• active processing elements
• operation description/state

Applications need in different proportions.

Breaking News (3/1/10)


Note release from startup Tabula today:

Robust Point depend on Width

w=1

w=8

w=64
Processors and FPGAs
(architecture vs. two application axes)

- FPGA: c=d=1, w=1, k=4
- "Processor": c=d=1024, w=64, k=2

Intermediate Architecture

- w=8
- c=64
- 16K PEs

Hard to be robust across entire space...

Caveats
- Model abstracts away many details that are important
  - interconnect (day 17–20)
  - control (day 23)
  - specialized functional units (day 14)
- Applications are a heterogeneous mix of characteristics

Modeling Message
- Architecture space is huge
- Easy to be very inefficient
- Hard to pick one point robust across entire space
- Why we have so many architectures?

General Message
- Parameterize architectures
- Look at continuum
  - costs
  - benefits
- Often have competing effects
  - leads to maxima/minima

Admin
- Should now have all background for HW5
  - Problem 2 similar (looking for robust point)
  - Different
    - Interconnect parameter
    - Energy efficiency
- Reading for Wednesday on Blackboard
Big Ideas
[MSB Ideas]

• Applications typically have structure
• Exploit this structure to reduce resource requirements
• Architecture is about understanding and exploiting structure and costs to reduce requirements

Big Ideas
[MSB Ideas]

• Instruction organization induces a design space (taxonomy) for programmable architectures
• Arch. structure and application requirements mismatch ⇒ inefficiencies
• Model ⇒ visualize efficiency trends
• Architecture space is huge
  – can be very inefficient
  – need to learn to navigate