ESE680-002 (ESE534): Computer Organization

Day 9: February 7, 2007
Instruction Space Modeling

Last Time

• Instruction Requirements
• Instruction Space

Architecture Instruction Taxonomy

<table>
<thead>
<tr>
<th>Control Threads (PCs)</th>
<th>pin/0 per Control Thread</th>
<th>Instruction Depth</th>
<th>Granularity</th>
<th>Architecture/Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>n/a</td>
<td>0</td>
<td>n/a</td>
<td>FPGA</td>
</tr>
<tr>
<td>n</td>
<td>1</td>
<td>1</td>
<td>Reconfigurable ALUs</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>e</td>
<td>1</td>
<td>Traditional Processors</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>1</td>
<td>1</td>
<td>Vector Processors</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>x</td>
<td>1</td>
<td>DPRGA</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td>0</td>
<td>8</td>
<td>TACED</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>HIRASCORE</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>MMID</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VEGA</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VEGA</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VEGA</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>VEGA</td>
<td></td>
</tr>
</tbody>
</table>

Today

• Instructions
  – Model Architecture
    • 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 (big) is a solution
  – 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
• Deal with
  – estimation from components
  – technology differences
  – few area-time points

Modeling Instruction Effects

• Restrictions from “ideal” save area
• Restriction from “ideal” limits usability (yield) of PE
• Want to understand effects
  – area model
  – utilization/yield model

Efficiency/Yield Intuition

• What happens when
  – Datapath is too wide?
  – Datapath is too narrow?
  – Instruction memory is too deep?
  – Instruction memory is too shallow?
Relative Sizes

- Bit Operator: 10-20K\,\lambda^2
- Bit Operator Interconnect: 500K-1M\,\lambda^2
- Instruction (w/ interconnect): 80K\,\lambda^2
- Memory bit (SRAM): 1-2K\,\lambda^2

Model Area

\[ A_{bit\_elm} = A_{fixed} + N_{SW}(N_p, w, p) \cdot A_{SW \ interconnect} \]
\[ + \frac{(c)}{w} \cdot n_{bits} \cdot A_{mem\_cell} \cdot \frac{\text{instruction memory}}{d} \cdot A_{mem\_cell} \cdot \text{retiming memory} \]

Architectures Fall in Space

Calibrate Model

- FPGA model \( w = 1, d = 1, c = 1, k = 4 \) - 880K\,\lambda^2
- Xilinx 4K - 630K\,\lambda^2
- Altera 8K - 930K\,\lambda^2

- SIMD model \( w = 1000, d = 0, c = 0, k = 3 \) - 170K\,\lambda^2
- Abacus - 190K\,\lambda^2

- Processor model \( w = 32, d = 64, c = 1024, k = 2 \) - 2.6M\,\lambda^2
- MIPS-X - 2.1M\,\lambda^2

Peak Densities from Model

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

- Large difference in peak densities
  - large design space!
Peak Densities from Model

Efficiency

• What do we want to maximize?
  – Useful work per unit silicon
  – (not potential/peak work)

• Yield Fraction / Area
  • (or minimize (Area/Yield))

Efficiency

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

Width Mismatch Efficiency Calculation

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

$$E = \frac{W_{task} \times A_{bitelm}|w=w_{task}}{W_{arch} \times \frac{W_{task}}{W_{arch}} \times A_{bitelm}|w=w_{arch}}$$

Efficiency: Width Mismatch

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)?

Context Depth

Efficiency with fixed Width

Path Length

Context Depth

w=1, 16K PEs

Ideal Efficiency (different model)

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

Applications need in different proportions.

Robust point: \( c \cdot A_{ctx} = A_{base} \)

Robust Point depend on Width

w=1  w=8  w=64

Processors and FPGAs

FPGA

c=d=1, w=1, k=4

“Processor”

FPGA

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 which are important
  – interconnect (day 13–18)
  – control (day 24)
  – specialized functional units (next time)
• 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

• Assignment 4 out today
  – Did push back due dates for 4 and 5
• Reading for Monday on web
  – Supplemental from this month TRCAD

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