ECE 505 Computer Architecture

Branch Prediction

Exceptions

Floating Point Instructions

Berk Sunar and Thomas Eisenbarth
Branch Prediction

• Static: Always predict as taken or not taken.
  • Examples: First MIPS and SPARC architectures would predict not taken always.
  • Forward branches as taken and backwards as not taken.

• Dynamic: Predict branch based on its history.
  • Local: use only history of the analyzed branch.
  • General: use history of all branches.

• Static prediction is still used when the branch is not in the history.
Advanced Branch Prediction

• We have a long code with many branches. How can we predict the future behavior (T or UT) of each branch?

1\textsuperscript{st} try: Use the $k$-LSB of branch address to select 1-bit predictor.
Advanced Branch Prediction

- 2\textsuperscript{st} try: Use the $k$-LSB of branch address to select 2-bit predictor. Total size = $2 \times 2^k$ bits

Generalized to: $n$-bit predictor using $n$-bit saturating counter
Total Size = $n \times 2^k$ bits
Advanced Branch Prediction

• 3rd try: Correlating Branch Predictor
  Track the last $m$ branches. For each sequence, use a separate predictor (Global Information).
  Why?

  ```
  if (aa==2)
      aa=0;
  if (bb==2)
      bb=0;
  if (aa!=bb) {
  ```

  Branch behavior depends on previous branches
Advanced Branch Prediction

- 3rd try: Correlating Branch Predictor
  Use the $k$-LSB of branch address
  Also, use the result of $m$-last branches
  Select n-bit predictor.

\[
\begin{array}{c}
\text{Pattern History Table} \\
\begin{array}{cccc}
00 & 01 & 10 & 11 \\
\end{array}
\end{array}
\]

\begin{align*}
& \text{if (aa==2)} \\
& \quad \text{aa=0;} \\
& \text{if (bb==2)} \\
& \quad \text{bb=0;} \\
& \text{if (aa!=bb) }
\end{align*}

$(m,n)$ predictor of $k$ entries:
Total size = $2^m \times 2^k \times n$ bits
Advanced Branch Prediction

• But, local information also follow patterns
• 4\textsuperscript{th} try: Tournament Predictors
  Use 2 different predictors:
  Local, following the pattern of the individual branch
  Global, following the pattern of previous branches
  Use a 2-bit saturating counter to select between them.
Advanced Branch Prediction

- 4th try: Tournament Predictors

Branch Address

\( K \)-bits

Global History Table

Local History Table

Selector

Total Size?

Prediction
Advanced Branch Prediction

• Performance:

Tournament Predictors can use 3 different predictors: Local, Global and a Special (Loops)
Further Problems with Pipelining

• Handling Exceptions
  \[ \text{SW } R2, 0(R4) \]

• Handling Floating-Point Operations
  \[ \text{MUL.D } F2, F0, F8 \]

• Case Example (MIPS R4000)
Exceptions

• What is an exception?

A problem has been detected and windows has been shut down to prevent damage to your computer.

If this is the first time you've seen this stop error screen, restart your computer. If this screen appears again, follow these steps:

Check to be sure you have adequate disk space. If a driver is identified in the Stop message, disable the driver or check with the manufacturer for driver updates. Try changing video adapters.

Check with your hardware vendor for any BIOS updates. Disable BIOS memory options such as caching or shadowing. If you need to use Safe Mode to remove or disable components, restart your computer, press F8 to select Advanced Startup options, and then select Safe Mode.

Technical Information:
*** STOP: 0x0000001E (0xFFFFFFC0000094, 0xFFFFF8000C074D1E, 0x0000000000000000, 0x0000000000000000)
Exceptions

• Characteristics of exceptions:
  • Synchronous Vs Asynchronous
    Cause by the code itself?
  • User Requested Vs Coerced
    Is it predictable?
  • User Maskable VS User Nonmaskable
    The hardware response can be disabled?
  • Within Vs Between Instructions
    During the execution of an instruction?
  • Resume Vs Terminate
    Try to resume or terminate the program?
## Exceptions

<table>
<thead>
<tr>
<th>Exception type</th>
<th>Synchronous?</th>
<th>User request?</th>
<th>Maskable?</th>
<th>Within?</th>
<th>Resume or Terminate?</th>
</tr>
</thead>
<tbody>
<tr>
<td>I/O device request</td>
<td>Asynchronous</td>
<td>Coerced</td>
<td>Non-maskable</td>
<td>Between</td>
<td>Resume</td>
</tr>
<tr>
<td>Invoke operating system</td>
<td>Synchronous</td>
<td>User request</td>
<td>Non-maskable</td>
<td>Between</td>
<td>Resume</td>
</tr>
<tr>
<td>Breakpoint</td>
<td>Synchronous</td>
<td>User request</td>
<td>maskable</td>
<td>Between</td>
<td>Resume</td>
</tr>
<tr>
<td>arithmetic overflow</td>
<td>Synchronous</td>
<td>Coerced</td>
<td>maskable</td>
<td>Within</td>
<td>Resume</td>
</tr>
<tr>
<td>Page fault</td>
<td>Synchronous</td>
<td>Coerced</td>
<td>Non-maskable</td>
<td>Within</td>
<td>Resume</td>
</tr>
<tr>
<td>Using undefined instructions</td>
<td>Synchronous</td>
<td>Coerced</td>
<td>Non-maskable</td>
<td>Within</td>
<td>Terminate</td>
</tr>
<tr>
<td>Hardware malfunctions / Power failure</td>
<td>Asynchronous</td>
<td>Coerced</td>
<td>Non-maskable</td>
<td>Within</td>
<td>Terminate</td>
</tr>
</tbody>
</table>
Exceptions

• How can we “resume” after “within” exception?
  
  • Force a trap instruction into the IF
  • Turn-off all writes into memory or registers
    (Never commit any further instruction)
  • Saves the PC to return later
    If out-of-order execution (e.g. delay branch):
    Save PC of all instructions
Exceptions

- Precise Exception
  If the pipeline can be stopped so that the instructions just before the faulting instruction are completed and those after it can be restarted from scratch.

- Problems?
  Multiple exceptions, out-of-order completion
Exceptions

• Handling Multiple Exceptions

<table>
<thead>
<tr>
<th></th>
<th>IF</th>
<th>ID</th>
<th>EX</th>
<th>MEM</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD</td>
<td></td>
<td></td>
<td></td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>DADD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Resolve one-by-one

<table>
<thead>
<tr>
<th></th>
<th>IF</th>
<th>ID</th>
<th>EX</th>
<th>MEM</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD</td>
<td></td>
<td></td>
<td></td>
<td>✗</td>
<td></td>
</tr>
<tr>
<td>DADD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Exception Status Vector

Page Fault
Arithmetic Exception
Page Fault
Page Fault
Page Fault
Page Fault
Exceptions

Possible exceptions in each stage

• **IF**: Page fault; misaligned memory access; memory protection violation
• **ID**: Undefined or illegal opcode
• **EX**: Arithmetic exception
• **MEM**: Page fault; misaligned memory access; memory protection violation
• **WB**: None
Floating-Point Operations

\[ \text{MUL.D F2, F0, F8} \]

• Problems?
  • FP unit require longer latency
  • Do all FP operations require same latency?
  • Complications on Pipelining?
  • Do we have to stall?
  • Can we commit a later but faster instruction?
  • What happen if we have an arithmetic exception?
Floating-Point Operations

• Assume four separate EX units

  ![Diagram of a processor pipeline](image)

• Latency: The number of cycles between an instruction that produces a result and an instruction that uses the result.

• Initiation Interval: The number of cycles that must elapse between issuing two operations of the same type.
Floating-Point Operations

<table>
<thead>
<tr>
<th>Functional unit</th>
<th>Latency</th>
<th>Initiation interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer ALU</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Data memory (integer and FP loads)</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>FP add</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>FP multiply (also integer multiply)</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>FP divide (also integer divide)</td>
<td>24</td>
<td>25</td>
</tr>
</tbody>
</table>

ECE 505 Computer Architecture
Floating-Point Operations

<table>
<thead>
<tr>
<th>Instruction</th>
<th>IF</th>
<th>ID</th>
<th>EX</th>
<th>MEM</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td>L.D</td>
<td>F4,0(R2)</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
</tr>
<tr>
<td>MUL.D</td>
<td>F0,F4,F6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD.D</td>
<td>F2,F0,F8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S.D</td>
<td>F2,0(R2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Floating-Point Operations

Clock cycle number

<table>
<thead>
<tr>
<th>Instruction</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>L.D F4,0(R2)</td>
<td>IF</td>
<td>ID</td>
<td>EX</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MUL.D F0,F4,F6</td>
<td>IF</td>
<td>ID</td>
<td>stall</td>
<td>M1</td>
<td>M2</td>
<td>M3</td>
<td>M4</td>
<td>M5</td>
<td>M6</td>
<td>M7</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADD.D F2,F0,F8</td>
<td>IF</td>
<td>stall</td>
<td>ID</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>A1</td>
<td>A2</td>
<td>A3</td>
<td>A4</td>
<td>MEM</td>
<td>WB</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S.D F2,0(R2)</td>
<td>IF</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>ID</td>
<td>EX</td>
<td>stall</td>
<td>stall</td>
<td>stall</td>
<td>MEM</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ECE 505 Computer Architecture
Floating-Point Operations

• Precise Exception?

```
DIV.D F0,F2,F4
ADD.D F10,F10,F8
SUB.D F12,F12,F14
```

Out-of-order completion. If DIV raised an exception, the value of F10 may be lost.

Solutions?
• Give up (I am imprecise exception)
• Two modes of operation
• Buffer all the results
• Mark committed instructions back to software