A VLIW Processor for Multimedia Applications

E. Holmann     T. Yoshida     A. Yamada     Y. Shimazu

Mitsubishi Electric Corporation, System LSI Laboratory
4-1 Mizuhara, Itami, Hyogo 664, Japan
Outline

• Objective
• System Architecture
• Performance
• Conclusions
System Architecture

MPEG2 bitstream

Video Audio

Display & Audio output

Block loader

Bus cnt.

D30V core

Inst RAM (32KB)

Data RAM (32KB)

System Bus I/F

DD bus

DA bus

External ROM RAM I/O etc.

Serial input

VLD process

ED bus

EA bus

DRAM I/F

External DRAM (2MB)
Instruction Formats

- Two types of instructions
  - Two short RISC sub-instructions (28 bits each)
    - Short sub-instruction L
    - Short sub-instruction R
  - One long RISC sub-instruction (54 bits)
    - Long sub-instruction
Instruction Issuing

<table>
<thead>
<tr>
<th>FM</th>
<th>Short sub-instruction L</th>
<th>Short sub-instruction R</th>
<th>Issue format</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td></td>
<td></td>
<td>parallel</td>
</tr>
<tr>
<td>01</td>
<td>Short sub-instruction L</td>
<td>Short sub-instruction R</td>
<td>serial</td>
</tr>
<tr>
<td>10</td>
<td>Short sub-instruction R</td>
<td>Short sub-instruction L</td>
<td>serial</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>Long sub-instruction</td>
<td>long inst</td>
</tr>
</tbody>
</table>

CC

<table>
<thead>
<tr>
<th></th>
<th>L-container</th>
<th></th>
<th>R-container</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4</td>
<td>32</td>
<td>36</td>
</tr>
</tbody>
</table>
Speculative Execution

- Every sub-instruction is speculatively executed
- 3 bits define condition for execution
- Conditions are based on status of user flags
- PSW has 8 user flags
- 2 user flags used for speculative execution
ALU Special Operations

Added video operations

- **Variable length saturation instruction:**
  - SAT, SATZ, SATHL, SATHH
    - SAT ra, rb, 24 -> ra = saturate (rb, 24)
    - SATHH ra, rb, 12 -> raH = saturate (rb, 12)

- **Flexible join instruction:**
  - JOINLL, JOINLH, JOINHL, JOINHH
    - JOINLH ra, rb, rc -> ra = rbL || rcH

- **Add sign instruction**
  - ADDS
    - ADDS ra, rb, rc -> ra = rb + sign(rc)
ALU Special Operations (cont.)

- Added sub-word operations

- ALU operations on dual half-word data:
  - ADD2H, SUB2H, ADDS2H, AVG2H, SAT2H, SATZ2H
  - MUL2H, MULX2H
    » ADD2H ra, rb, rc -> raH = rbH + rcH
    « raL = rbL + rcL

- Shifter operations on dual half-word data:
  - SRA2H, SRL2H, ROT2H
    » SRA2H ra, rb, 3 -> raH = rbH >> 3
    « raL = rbL >> 3
ALU Special Operations (cont.)

→ Added single half-word operations

• ALU operations on single half-word operands
  – ADDHppp, SUBHppp, JOINpp, MULHXpp, SATHp
    » ADDHLHH ra, rb, rc  ->  raL = rbH + rcH
    » SUBHHLH ra, rb, rc  ->  raH = rbL - rcH
    » JOINLH    ra, rb, rc  ->  ra = rbL || rcH
Memory Unit Special Features

- **Flexible operand types:**
  - byte (signed, unsigned)
  - half-word (signed, unsigned)
  - word
  - double word

- **Multiple operand accessing:**
  - four byte data load with packing
  - four byte data store with unpacking
  - two half-word data load with packing
  - two half-word data store with unpacking

- **Post-increment/decrement register indexed**
- **Modulo addressing**
Branch Unit Special Features

- Destination address calculated in second pipe stage
- Variable number of delay slots
- Block repeat with zero delay penalty
- Additional conditional branches
  - Test zero and branch instruction
  - Test not-zero and branch instruction
Instruction Examples

**Short_M**

<table>
<thead>
<tr>
<th>opcode</th>
<th>X</th>
<th>Ra</th>
<th>Rb</th>
<th>src</th>
</tr>
</thead>
</table>

- LDBU  R7, @(R6, 20)
- LDW   R4, @(R5, R7)
- LD2W  R8, @(R7+, R22)

**Short_A**

<table>
<thead>
<tr>
<th>opcode</th>
<th>Y</th>
<th>0</th>
<th>Ra</th>
<th>Rb</th>
<th>src</th>
</tr>
</thead>
</table>

- ADD   R7, R6, R8
- SUB   R10, R6, 20
- ADD2H R4, R5, R7

**Long**

<table>
<thead>
<tr>
<th>opcode</th>
<th>1</th>
<th>0</th>
<th>Ra</th>
<th>Rb</th>
<th>imm:32</th>
</tr>
</thead>
</table>

- LD2H  R7, @(R6, 0x00001000)
- AVG2H R8, R9, 0x00010001
### Instruction Examples (Branches)

<table>
<thead>
<tr>
<th>Opcode</th>
<th>Register</th>
<th>Immediate</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>02 779</td>
<td>00 00 0</td>
<td>00</td>
<td>BRA R3</td>
</tr>
<tr>
<td>02 779</td>
<td>10 disp:18</td>
<td></td>
<td>BRA 0x1234</td>
</tr>
<tr>
<td>02 779</td>
<td>W0 Ra src</td>
<td></td>
<td>BSRTZR R2, R5</td>
</tr>
<tr>
<td>02 779</td>
<td>W0 d:6 src</td>
<td></td>
<td>DJMP 3, R50</td>
</tr>
<tr>
<td>0x00</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x10000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x200</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x300</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0x39A</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Short_B1**

- **Op Code**: 02 779
- **Immediate**: 00 00 0
- **Description**: BRA R3

**Short_B2**

- **Op Code**: 02 779 10 disp:18
- **Description**: BRA 0x1234

**Short_B3**

- **Op Code**: 02 779 W0 Ra src
- **Description**: BSRTZR R2, R5

**Short_D1**

- **Op Code**: 02 779 W0 Ra src
- **Description**: DBRA R3, R17

**Short_D2**

- **Op Code**: 02 779 W0 d:6 src
- **Description**: DBSR 41, 0x300
# Pipeline Specification

<table>
<thead>
<tr>
<th>ALU</th>
<th>IF</th>
<th>D</th>
<th>EX</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IF</td>
<td>D</td>
<td>EX</td>
<td>WB</td>
</tr>
</tbody>
</table>

**IF**: Instruction Fetch  
**D**: Decode  
**EX**: Execute  
**WB**: Write Back

<table>
<thead>
<tr>
<th>LD/ST</th>
<th>IF</th>
<th>DA</th>
<th>M</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IF</td>
<td>DA</td>
<td>M</td>
<td>WB</td>
</tr>
</tbody>
</table>

**IF**: Instruction Fetch  
**DA**: Decode & Address  
**M**: Memory  
**WB**: Write Back

<table>
<thead>
<tr>
<th>BRA</th>
<th>IF</th>
<th>DA</th>
<th>EX</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IF</td>
<td>DA</td>
<td>EX</td>
<td>WB</td>
</tr>
</tbody>
</table>

**IF**: Instruction Fetch  
**DA**: Decode & Address  
**EX**: Execute (delayed branch)  
**WB**: Write Back

<table>
<thead>
<tr>
<th>MUL16</th>
<th>IF</th>
<th>D</th>
<th>EX</th>
<th>WB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IF</td>
<td>D</td>
<td>EX</td>
<td>WB</td>
</tr>
</tbody>
</table>

**IF**: Instruction Fetch  
**D**: Decode  
**EX**: Execute  
**WB**: Write Back
Conditional Branch Instructions

Instructions: BRAT, BSRT, JMPT, JSRT

- Decode Instruction
- Calculate newPC
- Speculative Execution (CC bits and user flags)
- Test register for zero/not zero (conditional execution)
Delayed Branch Instructions

Instructions: DBRA, DBSR, DJMP, DJSR

PC → IF DA EX WB → (DBRA DELAY, OFFSET)

PC + 8 → IF D EX WB

PC + 16 → IF D EX WB

PC + 24 → IF D EX WB

PC + OFFSET → IF D EX WB

Decode Instruction
Calculate PC + offset
Speculative Execution

Calculate PC + delay
### Processor Parameters and Figures of Merit

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock Frequency</td>
<td>250 MHz</td>
</tr>
<tr>
<td>Parallelism</td>
<td>2 way VLIW, 2 way SIMD</td>
</tr>
<tr>
<td>Peak Performance</td>
<td>1000 MIPS</td>
</tr>
<tr>
<td>Register File</td>
<td>64 x 32bits</td>
</tr>
<tr>
<td>RAM</td>
<td>32KB DRAM, 32KB IRAM</td>
</tr>
<tr>
<td>8x8 IDCT</td>
<td>&lt; 2 µseconds</td>
</tr>
<tr>
<td>256 point complex IFFT</td>
<td>~ 40 µseconds</td>
</tr>
<tr>
<td>MPEG-2 macroblock</td>
<td>&lt; 800 cycles (real time)</td>
</tr>
</tbody>
</table>
Conclusions

• High performance dual-issue RISC system
  – zero delay branches
  – zero delay repeat loops
  – speculative execution

• Multimedia Processor
  – sub-word operations
  – half-word operations
  – special video operations

• Single chip system for DSP applications
  – D30V serves functions of DSP and MCU chip
Application areas for D30V

- 2D/3D graphics
- AC-3 decode
- modem V.34 (28.8kbps)
- H.263 codec
- MPEG-1 decode
- MPEG-2 decode