Review of MIPS ISA and ALU

Though many simple and intuitive ideas have been covered, they collectively have a large impact, enabling modern computers to run efficiently and reliably.

Basic Representations

What is this bit sequence?

1011 1000 0100 0001 0000 0000 0000 1100

Hexadecimal
Signed Integer
Unsigned Integer
Signed-magnitude Integer
Twos Complement Integer
Floating Point Number
Instruction
ASCII

Terminology

- Pseudo Instruction
- Stack Pointer
- Word Addressing
- Program Counter
- Opcode
- Significand
- Hi/Lo Registers
- "shamt"
- Overflow

Are there more positive numbers or negative numbers? Why?

ISA

Anatomy of a R-type Instruction

add $8, $9 $10 # no comment

op rs rt rd shamt funct
0 9 10 8 0 32

00000010010101010101010000000000100000

6 5 5 5 5

ISA

Anatomy of an I-type Instruction

addi $8, $9 100 # Add immediate

op rs rt immediate
8 9 8 100

00100001001010000000000001100100

ISA

Anatomy of a J-type Instruction

j 1000000 # Jump far away

op address
2 250000

00010000000000001111010100000001001000

 ISA

Anatomy of an I-type Instruction

addi $8, $9 100 # Add immediate

op rs rt immediate
8 9 8 100

00100001001010000000000001100100

ISA

Anatomy of a J-type Instruction

j 1000000 # Jump far away

op address
2 250000

00010000000000001111010100000001001000
### Addressing Modes

**Immediate**

\[ \text{op} \quad \text{rs} \quad \text{st} \quad \text{immediate} \]

**Register Addressing**

\[ \text{op} \quad \text{rs} \quad \text{rt} \quad \text{rd} \quad \text{shamt} \quad \text{funct} \]

**Base Addressing**

\[ \text{op} \quad \text{rs} \quad \text{rt} \quad \text{address} \]

### More Addressing Modes

**PC-relative Addressing**

\[ \text{op} \quad \text{rs} \quad \text{st} \quad \text{address} \]

**Pseudo-direct Addressing**

\[ \text{op} \quad \text{address} \]

### Register Usage

<table>
<thead>
<tr>
<th>Name</th>
<th>Reg No.</th>
<th>Usage</th>
<th>Preserved On Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>$zero</td>
<td>0</td>
<td>Constant value 0</td>
<td>N.A.</td>
</tr>
<tr>
<td>1</td>
<td>Reserved for Assm</td>
<td>N.A.</td>
<td></td>
</tr>
<tr>
<td>$t0-$t7</td>
<td>8-15</td>
<td>Temporaries</td>
<td>No</td>
</tr>
<tr>
<td>$s0-$s7</td>
<td>16-23</td>
<td>Saved Locals</td>
<td>Yes</td>
</tr>
<tr>
<td>$s8-$s15</td>
<td>24-25</td>
<td>More temporaries</td>
<td>No</td>
</tr>
<tr>
<td>$s16-$s23</td>
<td>26-27</td>
<td>Operating System</td>
<td>N.A.</td>
</tr>
<tr>
<td>$s24-$s27</td>
<td>28-29</td>
<td>Global Pointer</td>
<td>Yes</td>
</tr>
<tr>
<td>$s30</td>
<td>30</td>
<td>Frame Pointer</td>
<td>Yes</td>
</tr>
<tr>
<td>$s31</td>
<td>31</td>
<td>Return Address</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### 1-Bit ALU

![1-Bit ALU Diagram]

### Data Path Structure

![Data Path Structure Diagram]