Our ultimate goal: building the datapath
Arithmetic Logic Unit (ALU)

ALU operation:
- 000 = and
- 001 = or
- 010 = add
- 110 = subtract
- 111 = slt
What about subtraction \((a - b)\)?

- Two's complement approach: just negate \(b\) and add.
- How do we negate?

- The solution:
Tailoring the ALU to the MIPS datapath

• Need to support the set-on-less-than instruction
  
  \texttt{slt rd, rs, rt}
  
  – slt is an arithmetic instruction
  – produces a 1 if rs < rt and 0 otherwise
  – use subtraction: \((a-b) < 0\) implies \(a < b\)

• Need to support test for equality (beq \$t5, \$t6, label)
  
  – use subtraction: \((a-b) = 0\) implies \(a = b\)
Supporting slt

- Binvert
- CarryIn
- Operation

- a0
- b0

- ALU0
- Less
- CarryOut

- a1
- b1
- 0

- ALU1
- Less
- CarryOut

- a2
- b2
- 0

- ALU2
- Less
- CarryOut

- a31
- b31
- 0

- ALU31
- Less

- Result0
- CarryIn
- Result1
- CarryIn
- Result2
- CarryIn
- Result31
- CarryIn

- Set
- Overflow

- Overflow detection
- Overflow

- Operation
- CarryIn
- Operation
- CarryIn

- Less
- CarryOut
- Result
- CarryOut
- Result
- CarryOut
- Result

- b0

- a0
Test for equality and complete ALU

ALU operation:
000 = and
001 = or
010 = add
110 = subtract
111 = slt

Control lines

<table>
<thead>
<tr>
<th>Bnegate</th>
<th>Operation</th>
<th>Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>00</td>
<td>and</td>
</tr>
<tr>
<td>0</td>
<td>01</td>
<td>or</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
<td>add</td>
</tr>
<tr>
<td>1</td>
<td>10</td>
<td>sub</td>
</tr>
<tr>
<td>1</td>
<td>11</td>
<td>slt</td>
</tr>
</tbody>
</table>

• Note: zero is a 1 when the result is zero!