Building a datapath

The basic design

Single-cycle edge-triggered execution.

- Every instruction begins execution one clock edge and completes execution on the next.

Instruction Fetch

1. Fetch instruction from memory.
2. Increment the program counter (PC) to point to the next instruction.

add $t0, $s1, $s2

R-type instructions include `add`, `sub`, `and`, `or`, and `slt`. 

*Design decision: Instructions are four bytes long.
Control unit signals

<table>
<thead>
<tr>
<th>ALU control lines</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>AND</td>
</tr>
<tr>
<td>0001</td>
<td>OR</td>
</tr>
<tr>
<td>0010</td>
<td>add</td>
</tr>
<tr>
<td>0110</td>
<td>subtract</td>
</tr>
<tr>
<td>0111</td>
<td>set on less than</td>
</tr>
<tr>
<td>1100</td>
<td>NOR</td>
</tr>
</tbody>
</table>

\[ A_{\text{invert}}, B_{\text{negate}}, \text{Op} \]

ALU operation: Zero, Result, Overflow

Datapath

Register file (32 words) and ALU

*An explicit RegWrite line is shown as data is not necessarily written back to the register file on every operation.

1w $t0, 1200($t1)

Instruction operation or opcode

<table>
<thead>
<tr>
<th>base register</th>
<th>destination register</th>
<th>offset</th>
</tr>
</thead>
<tbody>
<tr>
<td>100011</td>
<td>01001</td>
<td>01000</td>
</tr>
<tr>
<td>10100</td>
<td>10111</td>
<td>0000</td>
</tr>
</tbody>
</table>

6 bits 5 bits 5 bits 16 bits

still 32 bits

*And, yes there is a lw reg1, offset_value(reg2).
Uh oh.

An R-format instruction like \texttt{add \$t1, \$t2, \$t3} wants...

![Datapath Diagram](image1)

While...

an I-format instruction like \texttt{lw \$t0, 1200(\$t2)} wants...

![Datapath Diagram](image2)

MUXs to the rescue

(guided by the munificent control unit)

![Datapath Diagram](image3)

\texttt{beq \$s1, \$s2, 100}

instruction operation or opcode \hspace{6.5cm} second operand \hspace{6.5cm} location

\begin{tabular}{cccc}
000100 & 10001 & 10010 & 0000 0000 0010 0101 \\
6 bits & 5 bits & 5 bits & 16 bits
\end{tabular}

New costume idea: MUX ox

![Datapath Diagram](image4)
beq $s1, $s2, 100

Fetch + data paths: the instrument

Next: control