Design of the MIPS Processor

We will study the design of a simple version of MIPS that can support the following instructions:

- I-type instructions LW, SW
- R-type instructions, like ADD, SUB
- Conditional branch instruction BEQ
- J-type branch instruction J

The instruction formats

<table>
<thead>
<tr>
<th></th>
<th>6-bit</th>
<th>5-bit</th>
<th>5-bit</th>
<th>5-bit</th>
<th>5-bit</th>
<th>5-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>LW</td>
<td>op</td>
<td>rs</td>
<td>rt</td>
<td></td>
<td></td>
<td>immediate</td>
</tr>
<tr>
<td>SW</td>
<td>op</td>
<td>rs</td>
<td>rt</td>
<td></td>
<td></td>
<td>immediate</td>
</tr>
<tr>
<td>ADD</td>
<td>op</td>
<td>rs</td>
<td>rt</td>
<td>rd</td>
<td>0</td>
<td>func</td>
</tr>
<tr>
<td>SUB</td>
<td>op</td>
<td>rs</td>
<td>rt</td>
<td>rd</td>
<td>0</td>
<td>func</td>
</tr>
<tr>
<td>BEQ</td>
<td>op</td>
<td>rs</td>
<td>rt</td>
<td></td>
<td></td>
<td>immediate</td>
</tr>
<tr>
<td>J</td>
<td>op</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>address</td>
</tr>
</tbody>
</table>
How to generate the ALU control input? The control unit first generates this from the opcode of the instruction.
A **single-cycle MIPS**

We consider a simple version of MIPS that uses **Harvard architecture**. Harvard architecture uses separate memory for instruction and data.

Instruction memory is *read-only* – a programmer cannot write into the instruction memory.
To read from the data memory, set $\text{Memory read} = 1$
To write into the data memory, set $\text{Memory write} = 1$
Instruction fetching

Each clock cycle fetches the instruction from the address specified by the PC, and increments PC by 4 at the same time.
Executing R-type instructions

<table>
<thead>
<tr>
<th>op</th>
<th>rs</th>
<th>rt</th>
<th>rd</th>
<th>shamt</th>
<th>func</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 bits</td>
<td>5 bits</td>
<td>5 bits</td>
<td>5 bits</td>
<td>5 bits</td>
<td>6 bits</td>
</tr>
</tbody>
</table>

```
add $s4, $t1, $t2      000000 | 0101  | 01010 | 10100 | 00000 | 100000
```

This is the instruction format for the R-type instructions.
Here are the steps in the execution of an R-type instruction:

- Read instruction
- Read source registers \( rs \) and \( rt \)
- ALU performs the desired operation
- Store result in the destination register \( rd \).

Q. Why should all these be completed in a single cycle?
Executing **lw, sw instructions**

These are I-type instructions.

\[ \text{lw} \quad \mathit{\text{St0, -4(Sp)}} \quad \begin{array}{cccccc}
100011 & 11101 & 01000 & 1111 & 1111 & 1111 & 1100 \\
31 & 26 & 25 & 21 & 20 & 16 & 15 & 0
\end{array} \]

\[ \text{sw} \quad \mathit{\text{S0, 16(Sp)}} \quad \begin{array}{cccccc}
101011 & 11101 & 00100 & 0000 & 0000 & 0001 & 0000 \\
\end{array} \]

Try to recognize the steps in the execution of **lw** and **sw**.