**CS523**

**Bottom UP Approach Understanding a Processor and Designing a Processor**

Dr. A. Sahu  
CSE IIT Guwahati

---

**Processor Design : Basic**

- Microprocessor Vs Processor
  - Fetch, Decode, Register Access, Execute, Write Back
  - Single Cycle Design: All state in one cycle
  - Multi Cycle Design: Each state in one cycle Multiple T State, 8085 Microprocessor
- 8085 Microprocessor
- RISC Processor
- Simple MIPS Processor with 9 Instructions
- One Instruction Set Computer

---

**8085 Microprocessor**

- 8 Bit CPU
- 3-6Mhz
- Simpler design: Single Cycle CPU
- ISA = Pre x86 design (Semi CISC)
- 40 Pin Dual line Package
- 16 bit address
- 6 registers: B, C, D, E, H, L
- Accumulator 8 bit

---

**Understanding a given Processor (Example 8085)**

---

**Processor Architecture**

- Data Path: ALU, REG, MEM, BUS, DECODER
- Control Path: Start/Activate, Status

---

**8085 Microprocessor Architecture**
Instruction to Micro-Instructions

- **ADD B** // ACC = ACC + B
- Things need to do to execute this:
  - Fetch Instruction from Memory and put into IR
  - Put Higher address to A8-15
  - Put Lower Address to A0-A7
  - Get back data/instruction from memory to BUS
  - Activate BUS to IR gate to store data in IR
- Addition need to be done in ALU
  - Operands should be in TempR and ACC, Result will put to BUS
- Execute:
  - Activate MUX to select Register B
  - Put value to B to BUS
  - Activate Gate of Temp Reg, so that data from BUS will go to TempR
  - Do the Operation ADD
- Again result from BUS put to ACC
  - Activate Gate for ALU, so that data from ALU come to BUS
  - Activate Gate for ACC, so that data from BUS go to ACC

MIPS subset for implementation

- **RISC and 32 bit instruction**
- Arithmetic - logic instructions
  - add, sub, and, or, sll
- Memory reference instructions
  - lw, sw
- Control flow instructions
  - beq, j

Designing a Processor

(Example Tiny MIPS (9 instructions))

A Processor Design Method

Build the datapath step by step as follows

- Start with R - class instructions
- Include other instructions one by one
- Identify control signals
- Interconnect datapath and controller
Datapath for add, sub, and, or, slt

- fetch instruction
- address the register file
- pass operands to ALU
- pass result to register file
- increment PC

Actions required:

Format: add $t0, $s1, $s2

000000 10001 10010 01000 00000 100000

op rs rt rd shamt funct

Fetching instruction

Addressing RF

Passing operands to ALU

Passing the result to RF

Incrementing PC
Load and Store instructions
• format: I

Example: lw $t0, 32($s2)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>9</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>op</td>
<td>rs</td>
<td>rt</td>
<td>32</td>
</tr>
</tbody>
</table>

Adding “sw” instruction
ALU is used to compute Address

Adding “lw” instruction

Format of beq instruction
• beq I/J - format

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>op</td>
<td>rs</td>
<td>rt</td>
<td>16 bit number</td>
</tr>
</tbody>
</table>

Adding “beq” instruction

MIPS components - bit manipulation circuits

A Sahu
Format of jump instruction

- j J - format
  op 26 bit number

Adding “j” instruction

Control signals

Datapath + Control