

**Department of Electrical and Computer Engineering**

**Course Name:**  **Computer Organization and Architecture**

**Course Code: CSE-332L**

**Section: 2**

**Topic:**

16 Bit Single Cycle CPU

Submitted By:

* Md. Abdullah Al Mamun -1713062642
* Md. Sazidur Rahman -1632826042

Types of Instructions and their Format:

**opcode:** Operation Code

**rs:** Source Register

**rt:** Temporary Register

**rd:** Destination Register

**R-Type:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Opcode | rd | rs | rt | Function |
| 4 | 3 | 3 | 3 | 3 |

**I-Type:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Opcode** | **rd** | **rs** | **immediate** |
| **4** | **3** | **3** | **6** |

**J-Type:**

|  |  |
| --- | --- |
| **opcode** | **Address** |
| **4** | **12** |

**Data bits:** 16 bits throughout the project.

15 0

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Tables:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Type | Format | Details | Op Code | Function |
| ADD | Arithmetic | Register | add $S0, $S1, $S2  $S0 = $S1 + $S2 | 0000 | 000 |
| SUB | Arithmetic | Register | sub $S0, $S1, $S2  $S0 = $S1 - $S2 | 0000 | 001 |
| AND | Logical | Register | and $S0,$S1,$S2  $S0 = $S1 & $S2 | 0000 | 010 |
| OR | Logical | Register | or $S0,$S1,$S2  $S0 = $S1 || $S2 | 0000 | 011 |
| SLT | Conditional | Register | slt $S0, $S1,$S2  if ($S1 < $S2) $S0 = 1  else $S0 = 0 | 0000 | 100 |
| ADDi | Arithmetic | Immediate | addi $S0,$S1, 20  $S0 = $S1 + 20 | 0110 | XXXX |
| LW | Data Transfer | Immediate | lw $S0,20($S1)  $S0 = mem[$S1 + 20] | 0001 | XXXX |
| SW | Data Transfer | Immediate | sw $S0,20($S1)  mem[$S1 + 20]= $S0 | 0010 | XXXX |
| BEQ | Conditional | Immediate | beq $S0,$S1,25 if(S0 == S1)then goto 25th line  else proceed as usual | 0011 | XXXX |
| BNE | Conditional | Immediate | bne $S0,$S1,25 if(S0 != S1)then goto 25th line  else proceed as usual | 0100 | XXXX |
| J | Unconditional | Target | j address  Go to address | 0101 | XXXX |
|  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| Register Number | Conventional name | Value of register (3bit) |
| $1 | $s0 | 000 |
| $2 | $s1 | 001 |
| $3 | $s2 | 010 |
| $4 | $s3 | 011 |
| $ 5 | $t0 | 100 |
| $6 | $t1 | 101 |
| $7 | $t2 | 110 |
| $8 | $t3 | 111 |

**Instructions List:**

**R-Type**

|  |  |  |  |
| --- | --- | --- | --- |
| Instruction | Name | Action | Function |
| Add rd,rs,rt | Addition | rd=rs+rt | 000 |
| Sub rd,rs,rt | Subtraction | rd = rs-rt | 001 |
| And rd,rs,rt | And | rd = rs & rt | 010 |
| OR rd,rs,rt | Or | rd = rs |rt | 011 |
| SLT rd,rs,rt | Set less than | If (rs<rt) rd=1 else rd=0 | 100 |

**I type**

|  |  |  |  |
| --- | --- | --- | --- |
| Instruction | Name | Action | Function |
| LW rt,rs | Load | M[rs]= rt | \*\*\* |
| SW rt,rs | Store | Rt = M[rs] | \*\*\* |
| BEQ rt,rs | Branch on equal | If(rs==rt) then pc=pc+rs | \*\*\* |
| BNE rt,rs | Branch on not equal | If(rs!=rt) then pc=pc+rs | \*\*\* |
| Add i rt,rs,imm | Add immediate | rt = rs+imm | \*\*\* |

**J Type**

|  |  |  |  |
| --- | --- | --- | --- |
| **Instruction** | **Name** | **Action** | **Function** |
| Jump | **Jump** | j address  Go to address | **\*\*\*** |

**Control Signals Table:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Type | opcode | Reg  Dst | Alu  src | Mem  To  Reg | Reg  write | Mem  Read | Mem  write | Beq | BNe | Jump | Alu  Op1 | Alu  Op0 |
| R type | 0000 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| LW | 0001 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| SW | 0010 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Beq | 0011 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| Bne | 0100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| Jump | 0101 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| addi | 0110 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**ALU OP**

**LW/SW - 00**

**BEQ/BEN - 01**

**R-Type - 10**

**J- Type - 11**

**ALU CONTROL OPERATION**

|  |  |  |  |
| --- | --- | --- | --- |
| **Bin** | **A** | **B** | **Operation** |
| **0** | **0** | **0** | **and** |
| **0** | **0** | **1** | **or** |
| **0** | **1** | **0** | **add** |
| **1** | **1** | **0** | **sub** |
| **1** | **1** | **1** | **slt** |

**ALU Control Signals Table:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Alu op1 | Alu op0 | Function2 | Function1 | Functioon0 | Alu op  control | Operation |
| 0 | 0 | \* | \* | \* | 010 | LW |
| 0 | 0 | \* | \* | \* | 010 | SW |
| 0 | 1 | \* | \* | \* | 110 | Beq |
| 0 | 1 | \* | \* | \* | 110 | Ben |
| 1 | 0 | 0 | 0 | 0 | 010 | R(add) |
| 1 | 0 | 0 | 0 | 1 | 110 | R(sub) |
| 1 | 0 | 0 | 1 | 0 | 000 | R(and) |
| 1 | 0 | 0 | 1 | 1 | 001 | R(or) |
| 1 | 0 | 1 | 0 | 0 | 111 | R(SLT) |
| 1 | 1 | \* | \* | \* | 010 | Jump |
| 0 | 0 | \* | \* | \* | 010 | I -addi |

**Instruction table for ALU control:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Operation | Control  Unit  Opcode | Alu op | Instruction  operation | Func field | Desired  ALU  action | ALU  Control  Input |
| R-Type | 0000 | 10 | Add | 000 | Add | 010 |
| R-Type | 0000 | 10 | Sub | 001 | Sub | 110 |
| R-Type | 0000 | 10 | And | 010 | And | 000 |
| R-Type | 0000 | 10 | Or | 011 | Or | 001 |
| R-Type | 0000 | 10 | Slt | 100 | Set on  Less than | 111 |
| LW | 0001 | 00 | Load | \*\*\* | Add | 010 |
| SW | 0010 | 00 | Store | \*\*\* | Add | 010 |
| Beq | 0011 | 01 | Branch  Equal | \*\*\* | Sub | 110 |
| Bne | 0100 | 01 | Branch not  Equal | \*\*\* | Sub | 110 |
| Jump | 0101 | 10 | Jump  unconditional | \*\*\* | \*\* | 010 |
| Add i | 0110 | 00 |  | \*\*\* |  | 010 |