**SECOND PROJECT JONAS JIJIRI**

**#1 (HDL based) - Implement a simple microprocessor**

**12 March 2025**

**Lockdown of design specifications**

**The above document must contain all information on registers, instruction set, timing diagrams. Errors can always be fixed. Minor information can be absent.**

Simple HDL-Based Microprocessor Specifications

**1. Design Lockdown Statement**

This document serves as the final locked-down specification for the simple HDL-based microprocessor. It contains all necessary details on registers, the instruction set, and timing diagrams. Errors can be corrected, but no major modifications to the architecture will be made. Minor details may still be refined as needed.

**2. Block Diagram with Input/Output Pins**

The microprocessor consists of the following main components:

Control Unit (CU): Decodes instructions and controls data flow.

Arithmetic Logic Unit (ALU): Performs arithmetic and logical operations.

Registers: Stores temporary data and instruction-related information.

Program Counter (PC): Holds the address of the next instruction.

Instruction Register (IR): Holds the current instruction being executed.

Memory Interface: Reads and writes data to memory.

I/O Interface: Handles communication with external devices.

**Input/Output Pins:**

* **Inputs:**

clk: Clock signal

rst: Reset signal

instr\_in [7:0]: Instruction input bus

data\_in [7:0]: Data input bus

addr\_in [7:0]: Address input bus

rw: Read/Write control

* **Outputs:**

data out [7:0]: Data output bus

addr out [7:0]: Address output bus

status [3:0]: Status/flags output

**3. Number of Registers**

* **General-Purpose Registers (GPRs)**: 4 registers (R0 - R3), each 8-bit wide
* **Special-Purpose Registers:**

**Program Counter (PC):** 8-bit

**Instruction Register (IR):** 8-bit

**Accumulator (ACC):** 8-bit

**Stack Pointer (SP):** 8-bit

**Control Status Register (CSR):** Stores system control flags

**4. Flags**

The processor includes a 4-bit status register containing:

**Z (Zero Flag)**: Set when the result of an operation is zero.

* **C (Carry Flag)**: Set when an arithmetic operation generates a carry.
* **N (Negative Flag)**: Set when the result of an operation is negative.
* **O (Overflow Flag)**: Set when signed arithmetic overflow occur

5. **INSTRUCTION SET**

|  |  |  |  |
| --- | --- | --- | --- |
| **Opcode** | **Mnemonic** | **Description** | **Example** |
| 0000 | NOP | No Operation | NOP |
| 0001 | LDA Rn | Load ACC from register n | LDA R1 |
| 0010 | STA Rn | Store ACC in register n | STA R2 |
| 0011 | ADD Rn | Add register n to ACC | ADD R3 |
| 0100 | SUB Rn | Subtract register n from ACC | SUB R0 |
| 0101 | Logical AND with ACC | AND R2 | AND Rn |
| 0110 | OR Rn | Logical OR with ACC | OR R1 |
| 0111 | XOR Rn | Logical XOR with ACC | XOR R3 |
| 1001 | JZ addr | Jump if Zero flag is set | JZ 0x20 |
| 1010 | JC addr | Jump if Carry flag is set | JC 0x30 |
| 1011 | HALT | Stop execution | HALT |

**6**

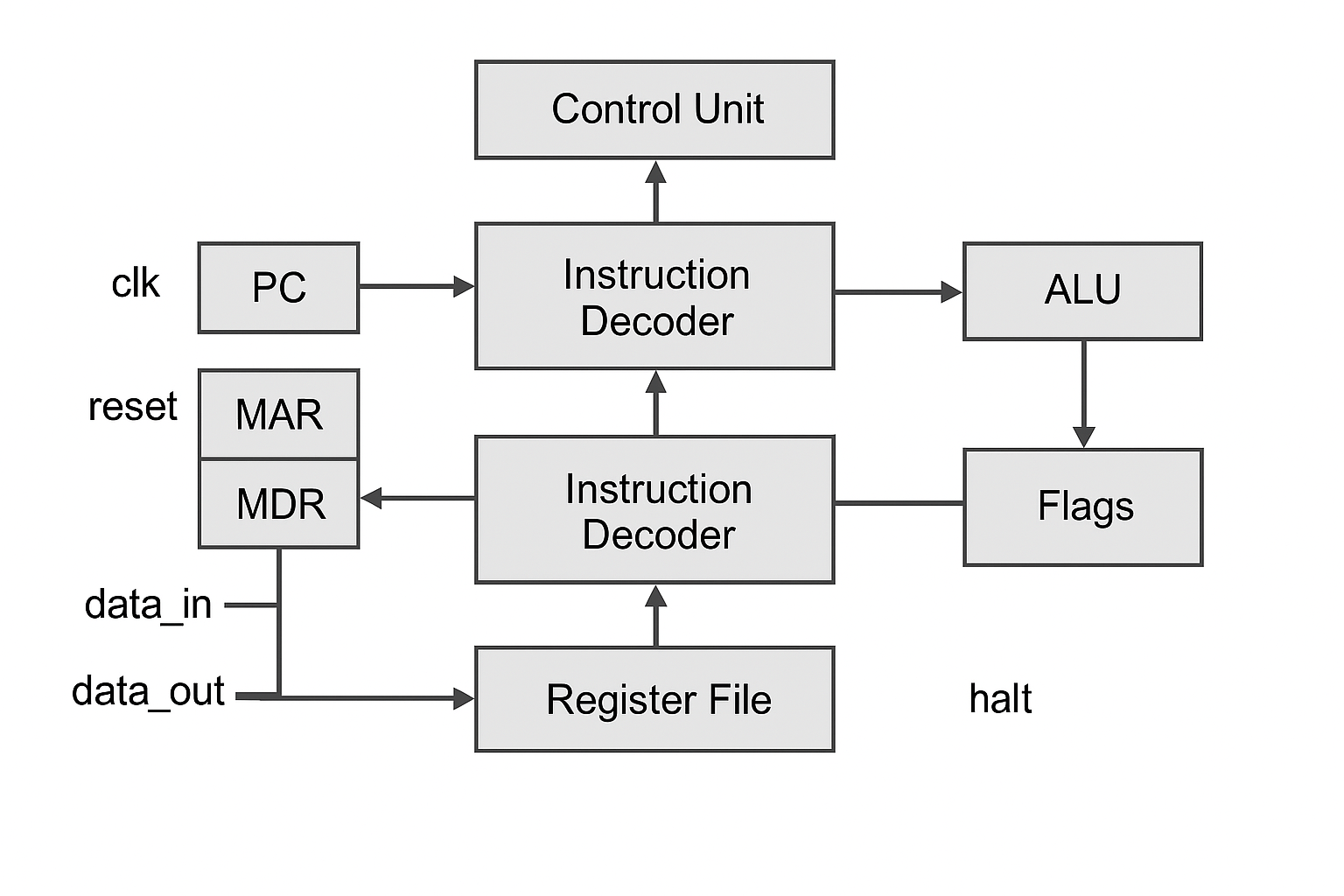
**Timing Diagram for a Sample Instruction**

**Example: LDA R1 (Load Accumulator from Register 1)**

|  |  |  |
| --- | --- | --- |
| Clock Cycle | Operation | Description |

|  |  |  |
| --- | --- | --- |
| T1 | Fetch | PC -> MAR (Memory Address Register) |
| T2 | Memory Read | IR <- Memory [PC] (Fetch Instruction) |
| T3 | Decode | Decode instruction LDA R1 |
| T4 | Execute | ACC <- R1 (Load value from Register 1) |
| T5 | Update PC | PC <- PC + 1 (Next Instruction) |

This document serves as the final locked-down specification for the simple HDL-based microprocessor. Minor refinements can be made, and errors will be corrected as necessary, but the core design will not change.

****

**20th March. Simple HDL-Based Microprocessor Specifications**

**1. Design Lockdown Statement**

This document serves as the final locked-down specification for the simple HDL-based microprocessor. It contains all necessary details on registers, the instruction set, and timing diagrams. Errors can be corrected, but no major modifications to the architecture will be made. Minor details may still be refined as needed.

**2. Block Diagram with Input/Output Pins**

The microprocessor consists of the following main components:

* **Control Unit (CU)**: Decodes instructions and controls data flow.
* **Arithmetic Logic Unit (ALU)**: Performs arithmetic and logical operations.
* **Registers**: Stores temporary data and instruction-related information.
* **Program Counter (PC)**: Holds the address of the next instruction.
* **Instruction Register (IR)**: Holds the current instruction being executed.
* **Memory Interface**: Reads and writes data to memory.
* **I/O Interface**: Handles communication with external devices.

**Input/Output Pins:**

* **Inputs:**
  + clk: Clock signal
  + rst: Reset signal
  + instr\_in Instruction input bus
  + data\_in Data input bus
  + addr\_in Address input bus
  + rw: Read/Write control
* **Outputs:**

data\_out Data output bus

addr\_out[7:0]: Address output bus

status [3:0]: Status/flags output

**BLOCK DIAGRAM WITH INPUT OUTPUT PINS**

**A black background with white rectangles

AI-generated content may be incorrect.**

**3. Number of Registers**

**General-Purpose Registers (GPRs)**: 4 registers (R0 - R3), each 8-bit wide

**Special-Purpose Registers:**

**Program Counter (PC):** 8-bit

**Instruction Register (IR):** 8-bit

**Accumulator (ACC):** 8-bit

**Stack Pointer (SP):** 8-bit

**Control Status Register (CSR):** Stores system control flags

**4. Flags**

The processor includes a 4-bit status register containing:

* **Z (Zero Flag)**: Set when the result of an operation is zero.
* **C (Carry Flag)**: Set when an arithmetic operation generates a carry.
* **N (Negative Flag)**: Set when the result of an operation is negative.
* **O (Overflow Flag)**: Set when signed arithmetic overflow occurs.

**5 Instruction Set**

|  |  |  |  |
| --- | --- | --- | --- |
| **Opcode** | **Mnemonic** | **Description** | **Example** |
| 0000 | NOP | No Operation | NOP |
| 0001 | LDA Rn | Load ACC from register n | LDA R1 |
| 0010 | STA Rn | Store ACC in register n | STA R2 |
| 0011 | ADD Rn | Add register n to ACC | ADD R3 |
| 0100 | SUB Rn | Subtract register n from ACC | SUB R0 |
| 0101 | AND Rn | Logical AND with ACC | AND R2 |
| 0110 | OR Rn | Logical OR with ACC | OR R1 |
| 0111 | XOR Rn | Logical XOR with ACC | XOR R3 |
| 1000 | JMP addr | Jump to address | JMP 0x10 |
| 1001 | JZ addr | Jump if Zero flag is set | JZ 0x20 |
| 1010 | JC addr | Jump if Carry flag is set | JC 0x30 |
| 1011 | HALT | Stop execution | HALT |

**6. Timing Diagram for a Sample Instruction**

**Example: LDA R1 (Load Accumulator from Register 1)**

**Stages:**

1. **IF – Instruction Fetch**
2. **ID – Instruction Decode & Register Fetch**
3. **EX – Execution / Effective Address Calculation**
4. **MEM – Memory Access**
5. **WB – Write Back**

| **Clock Cycle** | **Instruction** | **Control Signals Active** |
| --- | --- | --- |

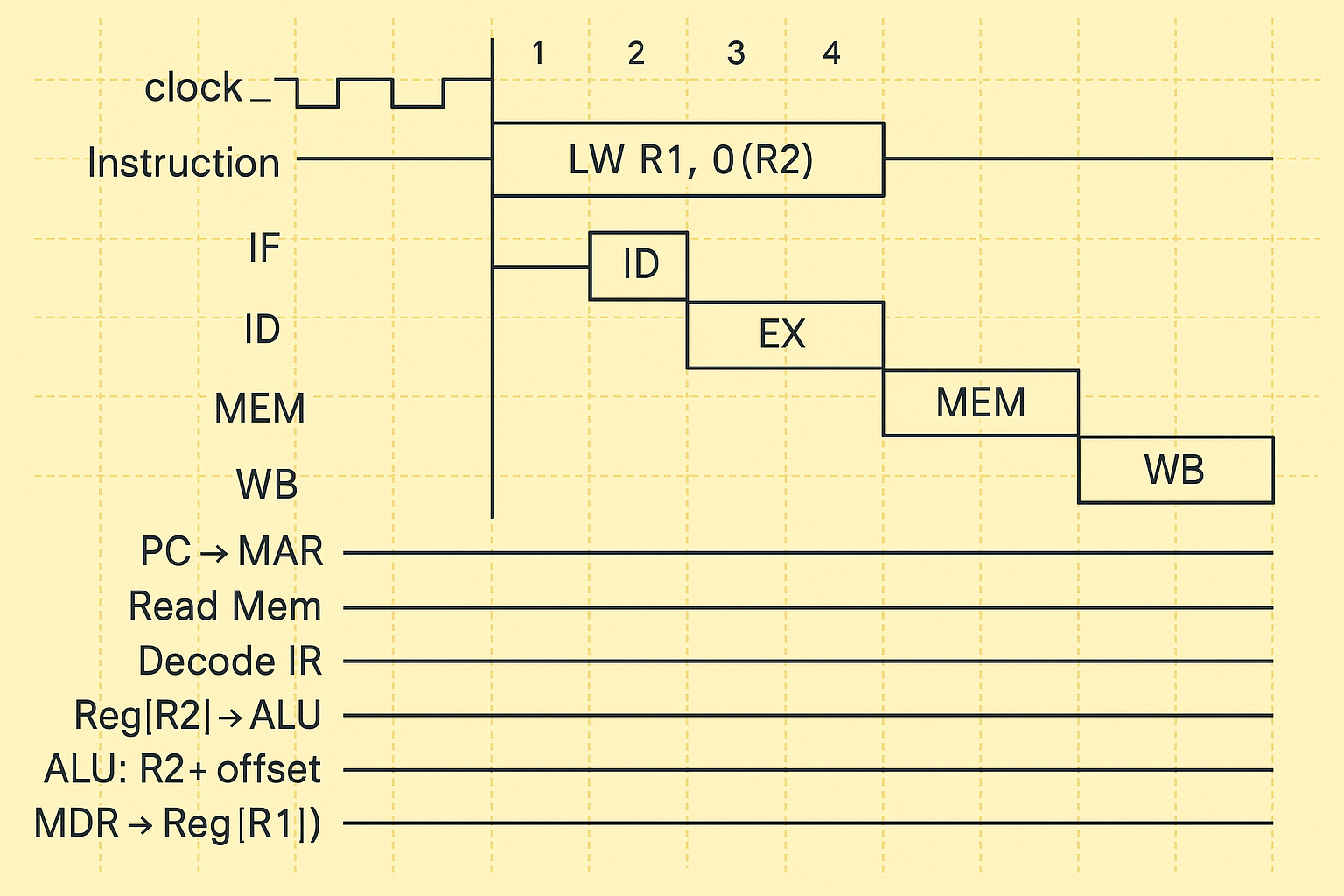
|  |  |  |
| --- | --- | --- |
| **A** | **IF** | **PC → MAR, Read Mem, IR ← MDR** |

|  |  |  |
| --- | --- | --- |
| **Cycle 2** | **ID** | **Decode IR, Reg[R2] → ALU** |

|  |  |  |
| --- | --- | --- |
| **Cycle 3** | **EX** | **ALU: R2 + offset, Address → MAR** |

|  |  |  |
| --- | --- | --- |
| **Cycle 4** | **MEM** | **Read Mem, MDR ← Mem[MAR]** |

|  |  |  |
| --- | --- | --- |
| **Cycle 5** | **WB** | **MDR → Reg[R1]** |

****