2CS504CC23

COMPUTER ARCHITECTURE

B. Tech Semester IV

INNOVATIVE ASSIGNMENT

REPORT

Prepared by:

Param Desai 23BCE208

Institute of Technology

Nirma University

Ahmedabad

March 2025

1. INTRODUCTION

This report analyzes the Verilog implementation of a simplified Register Reference Instruction Processor, which mimics the operation of basic control logic within a processor. The module demonstrates how certain micro-operations are executed based on 4-bit opcode inputs, affecting internal registers such as the Accumulator (AC) and a single-bit flag (E). This processor simulates control unit behavior, focusing specifically on register-reference instructions, as found in early computing models.

2. DESIGN OVERVIEW

The Verilog design comprises two main modules:
- register_reference_processor: Simulates processor operations based on opcode instructions.
- register_reference_processor_tb: A testbench used to verify and visualize behavior under various instruction inputs.

3. MODULE: register_reference_processor

3.1 Inputs and Outputs

Inputs:
- clk : Clock signal.
- reset : Active-high reset to initialize the processor.
- opcode : 4-bit code representing the operation to execute.

Outputs:
- AC : 8-bit Accumulator register.
- E : Single-bit flag used for circular shift and logical operations.
- halt : Processor halt indicator.

3.2 Internal Operations

The processor supports the following instruction set, defined by parameterized opcodes:

Opcode	Mnemonic	Operation Description
0000	CLA	Clear Accumulator (AC ← 0)

0001 | CLE | Clear E flag (E ← 0)
0010 | CMA | Complement AC
0011 | CME | Complement E
0100 | CIR | Circular right shift of AC & E
0101 | CIL | Circular left shift of AC & E
0110 | INC | Increment AC
0111 | SPA | Skip if AC is positive (MSB = 0)
1000 | SZA | Skip if AC is zero
1001 | SNA | Skip if AC is negative (MSB = 1)
1010 | HLT | Halt processor

3.3 Behavior

The module uses a case statement inside a clocked always block to perform operations. It logs operations using $display, allowing simulation outputs to explain state changes (e.g., values of AC and E before and after execution).

The reset condition initializes AC to 8'b10100101 (0xA5) and E to 1.

For each opcode:
- Registers are updated as per operation logic.
- Skip instructions (SPA, SZA, SNA) check AC flags but do not implement actual program counter control (can be extended in future versions).
- The HLT instruction sets the halt signal to disable further instruction processing.

4. MODULE: register_reference_processor_tb (Testbench)

4.1 Features
- Initializes signals and toggles clock every 5 time units.
- Uses $dumpfile and $dumpvars for waveform visualization.
- Tests each instruction opcode in a controlled manner.
- Interleaves opcode = 4'b1111 as a no operation (NOP) to give time between executions.

4.2 Simulation Sequence
- Reset the processor.

- Sequentially apply each opcode to observe behavior.
- Use comments or waveform viewers (GTKWave) to interpret signal transitions.

5. OUTPUT AND SIMULATION ANALYSIS

The output from $display statements in the simulation confirms:
- Correct initialization of registers.
- Successful execution of operations like clearing, complementing, shifting, incrementing, and condition checking.
- Correct state transitions and expected halting behavior.

The use of conditional logs helps understand how data flows internally—important for students and learners simulating hardware logic for the first time.

6. FUTURE SCOPE AND EXTENSIONS

- PC and MAR integration: To allow address-based instruction skipping and jumps.
- Instruction memory decoder: To automate opcode loading from memory.
- More operations: Such as load/store from memory, branching, and interrupts.
- Pipelining simulation: For advanced architectural learning.

7. CONCLUSION

This Verilog-based implementation successfully simulates a basic register reference processor with elementary micro-operations. It helps in visualizing the inner workings of control logic and instruction execution in digital processors. The modular design and logging make it an excellent learning tool for computer architecture concepts.