Anusha Nagesh (AXN220008)

Nishanth Raghavendra (NXR210063)

FINAL REPORT – MICROPROCESSOR DESIGN

➢ MCU Top Module:

Main module which connected the MCU to the MCU_IO module. We input a value to the MCU_Top module and the output is fed into the FPGA to produce a corresponding color output.

Inputs:

- Fpga_in
- Clk
- Reset

Output:

- Fpga_out

Modules instantiated:

➢ Microcontroller module:

I. Modules instantiated:

a. **Init Logic**: Initial logic implementation for PS, Z and 2-bit BS. Gives branch selection output

b. **mux c**: Inputs provided by BrA, RAA, and PC of next stage (PC + 1). Outputs to PC

c. **Program Memory**: The module where instructions are fetched and updated.

d. **Instruction Decoder**: Opcode initialization and updating of various select register implementation.

e. **Constant Unit**: Sections IM and SH initialized in accordance with the 6-bit IR register input.

f. **Register File**: Stores the register values of instructions

g. **mux a**: For analyzing data passed along the pipeline along Bus A and passed on as input to ALU.

h. **mux b**: For analyzing data passed along the pipeline along Bus B and passed on as input to ALU.

i: **Data Hazard Module**: DHS register implementation for management of data hazard and stalling.

j: **Alu**: Arithmetic Logic Unit which contains the implementations for all the main logics.

k: **Data Memory**: Operations on memory block based on enable input.

j: **Adder**: Addition operator unit for PC and B; propagates output BrA to mux c.

k: **mux d**: Module implemented for writeback stage selection. Outputs to Mux E

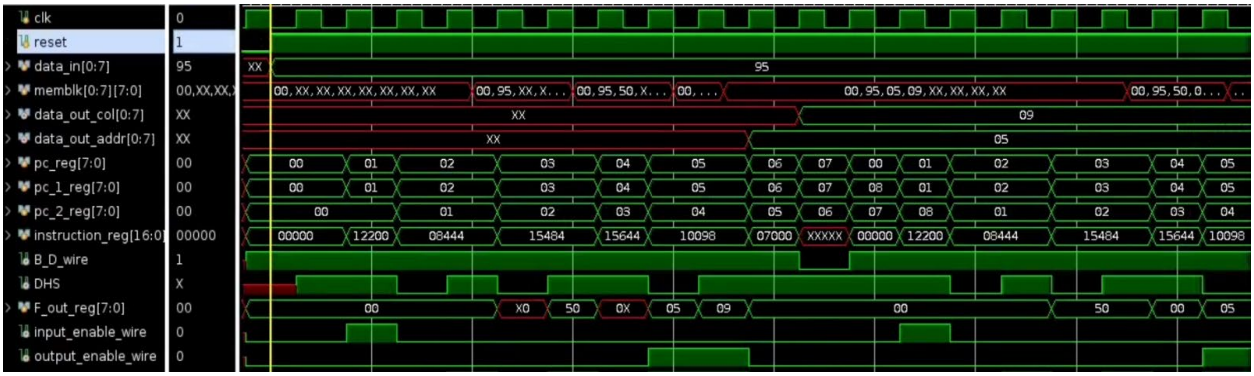
l: **mux e**: Register write in writeback cycle. Output of Mux E is input to register file

m: **Branch Detector**: Data management module for each stage of pipelining, performed by BD_wire.

II. Inputs and outputs:

a. **Clock:** Data updation set on posedge.
b. **Reset:** Data updation set on negedge.
c. **Data_in**: 9 bit.
d. **data_out_col:** 8 bit.
e. **data_out_addr:** 8 bit

III. Simulation Results:



Instructions fed as 17-bit binary inputs in program memory module.

```
begin
instruction_memory[0] = 17'b10010001000000000;
instruction_memory[1] = 17'b01000010001000100;
instruction_memory[2] = 17'b10101010010000100;
instruction_memory[3] = 17'b10101011001000100;
instruction_memory[4] = 17'b10000000010011000;
instruction_memory[5] = 17'b00111000000000000;

end
```

- **Mcu.io module**

Connects Mcu to input and output peripherals from the MCU main module. Instantiated by

Anusha Nagesh (AXN220008)

Nishanth Raghavendra (NXR210063)

VGA blocks for data propagation into the FPGA module.

I. Modules instantiated:

- VGA_out:

  a. VGA_driver
  b. VGA_grid
  c. Clock_100to25
  d. Color_converter

CHALLENGES:

- Understanding how to use Vivado
- Understanding how to read a circuit diagram and implement it as Verilog code
- Comprehending how the 17-bit binary instruction we input through program memory is propagated and computed by the MCU
- Pipelining of instructions
- Understanding how Data Hazard and Branch detection stalls work

LEARNINGS:

- How to implement a microcontroller unit with instruction pipelining
- How to implement Data hazard stall
- Workings of how binary instruction is processed by the MCU