EEDG/CE 6302

Microprocessors and Embedded Systems

Electrical and Computer Engineering

Erik Jonsson School of Engineering and Computer Science

The University of Texas at Dallas

Dr. Tooraj Nikoubin

Project 1: 8 bit Microcontroller Unit with 4 pipeline stages and Hazard Control

Submitted By:

Muripa Uppaluri (MXU220008)

Chandanam Sai Nived (SXC210186)

Table of Contents

This report describes about the fully functional 8-bit microcontroller unit with 4 pipeline stages, its components, their functionality, and program execution. This MCU model has 3 main submodules – data memory, instruction memory and CPU.

Data Memory –

The Data Memory in a microcontroller refers to the memory used to store data and variables that change dynamically during execution. In this project, we've used Verilog HDL to design a data memory which is a RAM with 8-bit data_in input bus that can be used to write data into the memory and 8-bit data_out output bus that is used to read the value out from the given memory location. We also have a 1-bit wr_rd_en input signal which helps to understand whether we want a write operation (write data into the specified address location) or a read operation (read data out from the specified address location).

Instruction Memory –

The instruction memory stores the instructions to be implemented by the processor. It is usually ROM or flash memory and ensures the instructions are not lost or overwritten during runtime. In this project, we've used Verilog HDL to design an instruction memory with 8-bit addr_in bus input and 17-bit data_out bus. It has a memory capacity of 256*17 bits. It gets address from program counter and outputs the corresponding 17-bit instruction from memory for decoding.

Central Processing Unit –

The CPU several submodules that complete the ALU operation. The CPU has the following submodules –

- **Arithmetic Logic Unit (ALU) –**
  ALU performs the arithmetic and logic operations in MCU. We have a pure combinational ALU in this design. The instruction is decoded based on the opcode and the operands are used for data interpretation. We have 5 status signals that output from ALU and provide the status of carry, negative result, zero result, etc.

- **Multiplexors –**
  In this project, we have 4 multiplexers, MUXA, MUXB, MUXC, MUXD. MUX A and MUX B have 2 input ports each and each input port is 8 bits wide. MUXC has 4 input ports with 8 bits each and MUXD has 3 input ports and 2 select lines. In an MCU, the multiplexer selects one of several input signals and routes it to a single output line, which can then be processed by other components in the system.

- **Registers –**
  In this project we have 8 8bit registers that store the data required for MCU operation. Data can be read and stored into these registers.

- **Instruction Decoder –**
  The instruction decoder decodes the instruction received from instruction register and generates the necessary control signals to other components of MCU. Based on the type of instruction received, the instruction decoder asserts the conditional outputs that perform the required operation. We have 17-bit instructions as input to the decoder.

- **Constant Unit –**
  A constant unit is responsible for generating constant values that are necessary for program execution. In our design, constant unit performs the extension of bits from 6 to 8 for immediate values in 2 register type instructions.

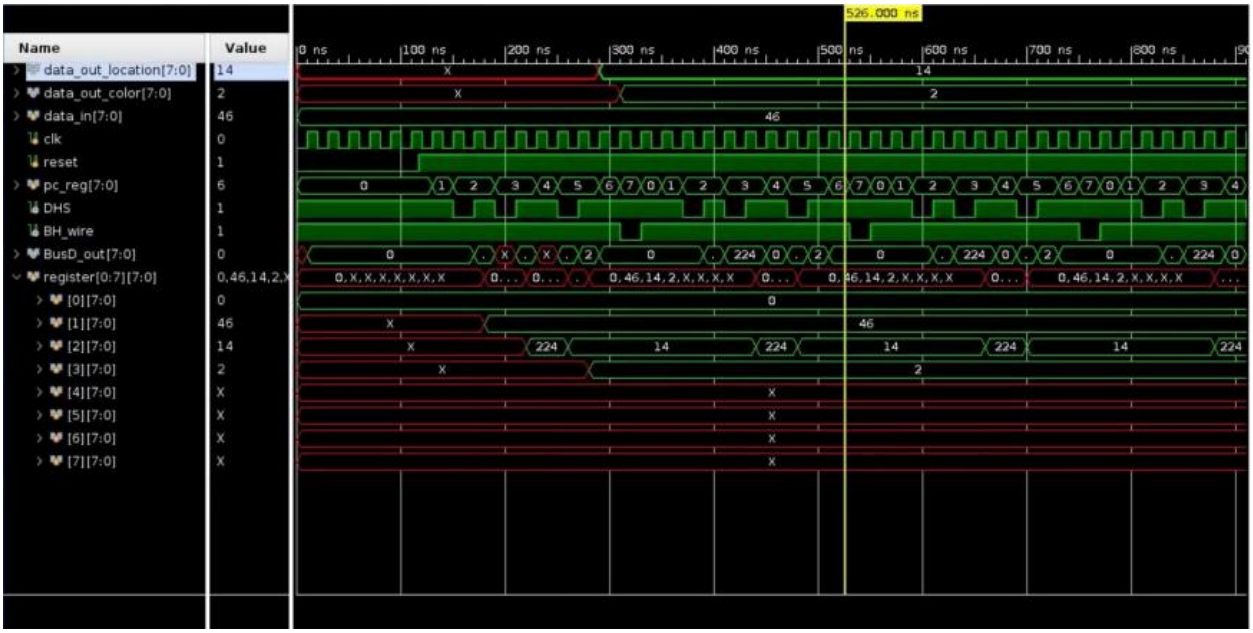
- **A set of pipeline registers –**
  We have 4 pipeline stages in our MCU design. And each of these pipeline stages require the data to be stored for the execution of next steps.

- **Data Hazard –**
  Data hazard is an error which occurs due to the pipeline mode of execution. When different stages of pipeline want to work on the same data entity, data hazard occurs.

- **Branch Hazards –**
  A branch hazard occurs when an instruction has jump and branch operations associated with it.

Testing the MCU –

We've created a Verilog testbench to check the working of the MCU module. We've simulated our Verilog design using Xilinx ISE and synthesized to generate the bit file to program the FPGA. This bit file contains the configuration information necessary to program an FPGA.



Testing the Verilog module using FPGA evaluation board –

In this project, we've used Digilent Nexys 3 evaluation board which contains Xilinx Spartan 6 FPGA. To connect to the FPGA, we've used a set of drivers. We've created a top level module called minsys to connect all the drivers to the MCU.

Programming the FPGA and Observing the output –

The FPGA is connected to the monitor using a VGA cable and FPGA is connected to laptop using Adept USB cable. The bit is programmed into the FPGA using Digilent Adept tool. Once programmed, we observe the colors based on slide switches on the FPGA.