Data Memory, Load and Store Instructions

To develop a data memory module and test it for reading and writing. Combine data memory module with RF + ALU to fetch memory data and load it to registers (memory read operation) and to store register data to memory (memory write operation). Below are the steps involved:

a. Design Data Memory unit using inputs addr 5 bit wide, wr_data 64 bit wide, clk, mem_wr, mem_rd witch are I bit wide and output rd_data 64 bit wide. We can load or store 64 bit data in the data memory unit at a time.
b. Design combine Data memory, RF with ALU. Where register content will be loaded from data memory unit using load instruction.
c. Design combine Data memory, RF with ALU. Where register content will be stored in data memory unit using store instruction.

- Simulation of Data Memory Unit

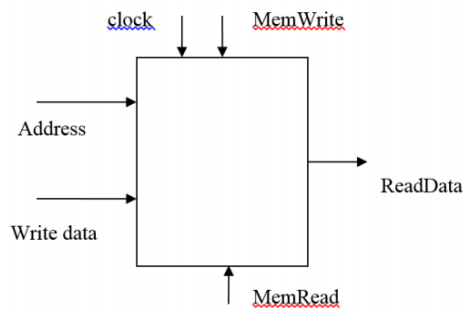


Figure 4.3 Memory module

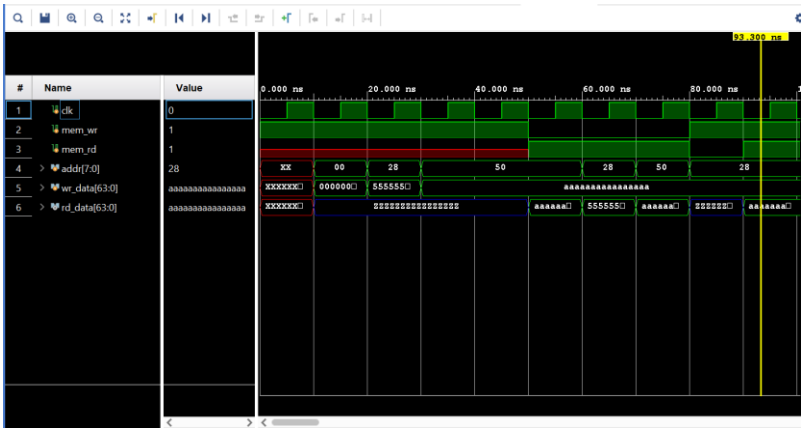
Data memory unit operation parameters are:

- Input clk, mem_wr, mem_rd 1 bit
- Input addr 8 bit
- Input wr_data 64 bit
- Output rd_data 64 bit

Module Name: DataMem

Test Module Name: DataMem_tb

Below screen shot show the waveform for input and output parameters of data memory unit.



Here the data memory unit is Byte addressable hence the input address is 8 bit. First we will store location one (i.e DM[0] to DM[7]) with value 64'h0000000000000000. Then will store Location 40 decimal (28 in hex) with 64'h5555555555555555 and Location 80 (50 in hex) with 64'haaaaaaaaaaaaaaaa. To store the data in memory we will set input mem_wr. Then will test location 40 and 80 by setting input mem_rd and observing rd_data. Then will override location 40 with 64'haaaaaaaaaaaaaaaa. And will check location by reading the same.

- Simulation of Data Memory Unit, Register File and ALU for load instruction

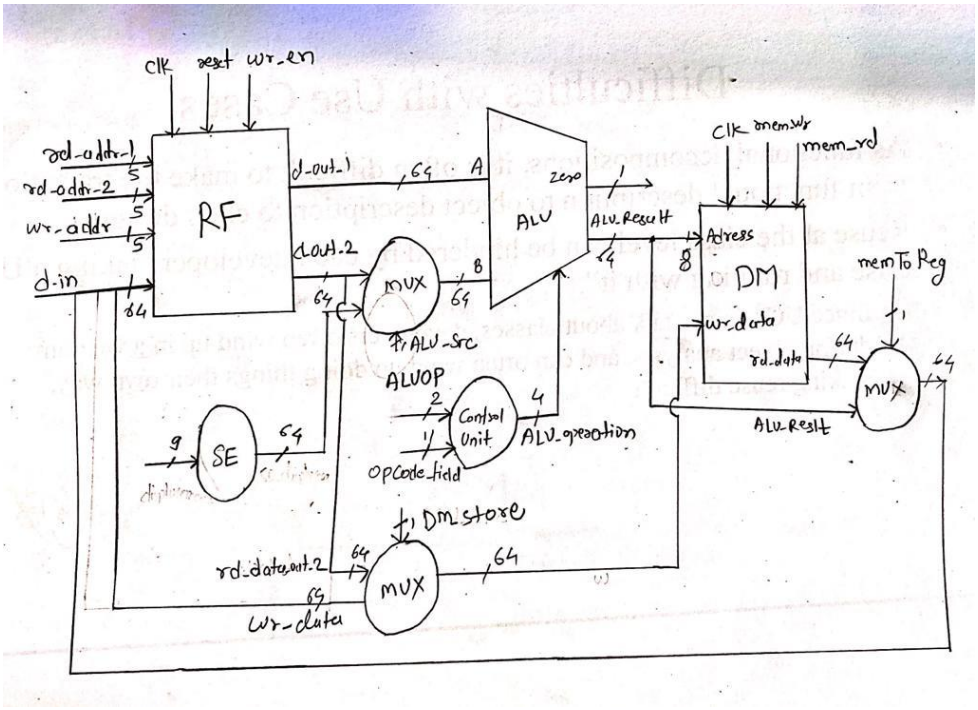


Fig: Load and store instruction with DM, RF and ALU.

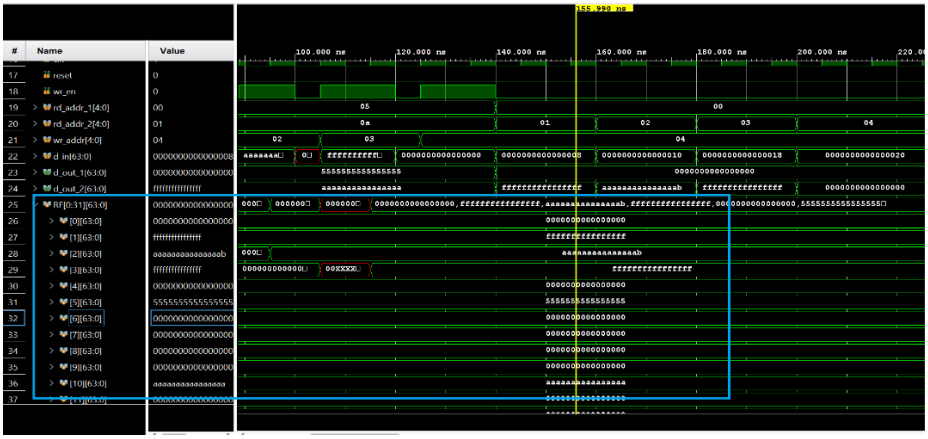
Data Memory Unit, Register File and ALU for load instruction parameters are:

- Input clk, reset, wr_en, memToReg, mem_rd, mem_wr, ALU_src I bit
- Input wr_addr, rd_addr_1, rd_addr_2 5 bit
- Input wr_data 64 bit
- Input displacement 9 bit
- Input ALU_op 2 bit
- Input Opcode_field 11 bit

Module Name: RFALUDM

Test Module Name: RFALUDM_tb

Below screen shot show the waveform for input and output parameters for load instruction.



Here we initialize Data memory unit with location 40 with 64'h5555555555555555 and location 80 with 64'haaaaaaaaaaaaaaaa. We reset the register file. With the help of sign extend module and ALU we calculated the memory location and load the data from memory to register file. We load register 5 with data from location 40 and register 10 with data from location 80. Then we perform ALU operation and store the ALU result back to register file which are as follows:

- Write back result of addition of register 5 and 10 to register 1
- Write back result of addition of register 5 and 10 to register 2
- Write back result of addition of register 5 and 10 to register 3
- Write back result of addition of register 5 and 10 to register 4

We have added one MUX unit to select input B for ALU unit with ALU_src as a selection input. Also one more Mux unit to select input wr_data in register file with memToReg as a selection input. Refer Fig: Load and store instruction with DM, RF and ALU for same.

- Simulation of Data Memory Unit, Register File and ALU for store instruction

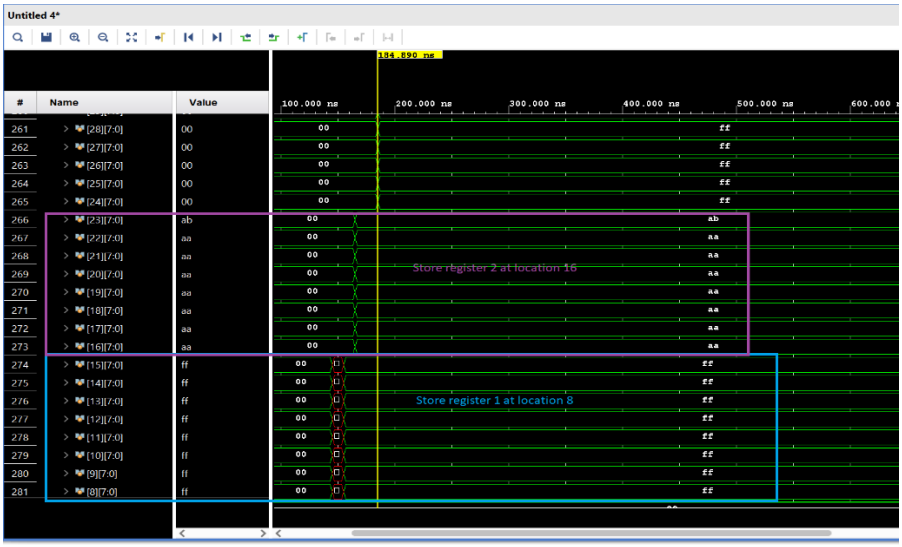
  Data Memory Unit, Register File and ALU for load instruction parameters are:

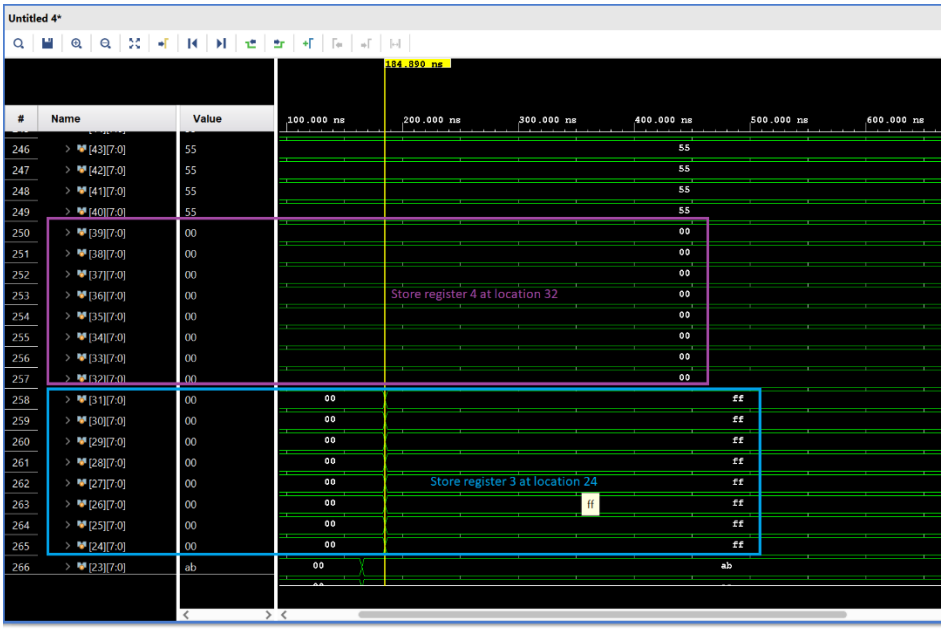
  - Input clk, reset, wr_en, memToReg, mem_rd, mem_wr, ALU_src, DM_store I bit
  - Input wr_addr, rd_addr_1, rd_addr_2 5 bit
  - Input wr_data 64 bit
  - Input displacement 9 bit
  - Input ALU_op 2 bit
  - Input Opcode_field 11 bit

  Module Name: RFALUDM

  Test Module Name: RFALUDM_tb

Below screen shot show the waveform for input and output parameters for store instruction.





Here we use the register R1, R2, R3, R4 values calculated in part B and store these to location 8, 16,24,32 respectively. We added one Mux unit to select input wr_data in data memory unit with DM_store as a selection input. This is to support both R and D type instruction. Refer Fig: Load and store instruction with DM, RF and ALU for same.