Name : salma soliman
Id : 900182325
Lab8 Monday 3:30pm

Experiment (1):

In this experiment, we updated and Simulated the Data Memory Module to accommodate the cache addition to the system.
Here is the list of differences between both modules :

The old module :

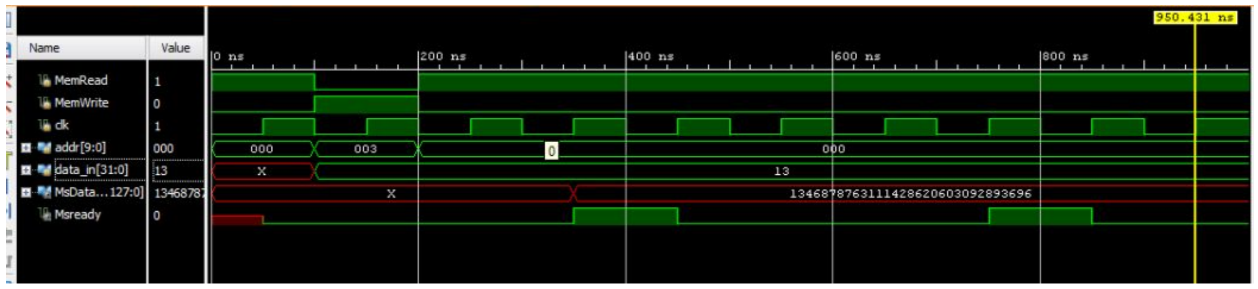
```
module DataMem (input clk, input MemRead, input MemWrite,
input [5:0] addr //representing the ALU result
,input [31:0] data_in, output [31:0] data_out);

reg [31:0] mem [0:63]; // here the mem size is smaller

always @(posedge clk)

Begin
if (MemWrite)  //accessing mem directly

mem[addr] <= data_in;

End

assign data_out = MemRead?mem[addr]:0; //reading from mem
directly

initial begin
mem[0]=32'd17;
mem[1]=32'd9;
mem[2]=32'd25;
endendmodule
```

The updated one :

```
module DataMem (input clk, input MemRead, input MemWrite,
input [9:0] addr // larger in size because it contains the
word offset and the index

, input [31:0] data_in, output reg [127:0] MsDataOut,//32*4
cause the mem sends a whole block that contains 4 words

 output reg Msready // represents that the mem finished
reading or writing);

reg [31:0] mem [0:1023];//1KB memory , larger mem size
```

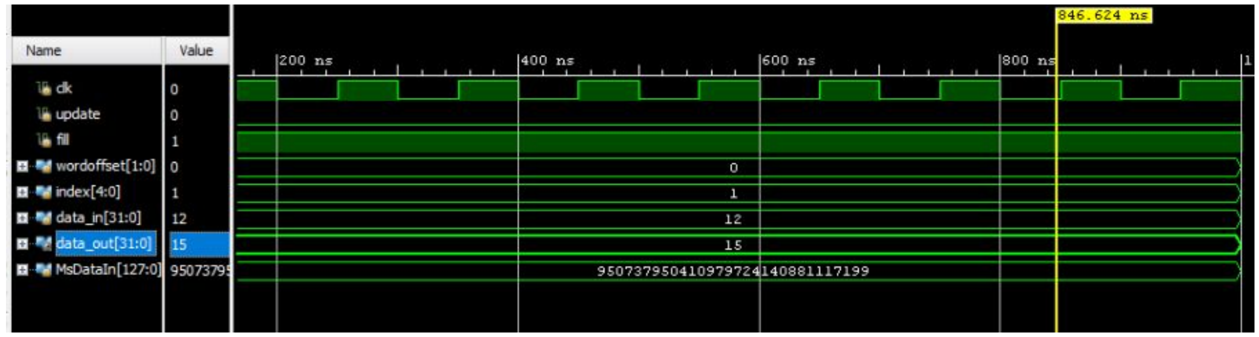
```
wire[9:0] bAddr;

assign bAddr = {addr[9:2],2'b00}; // neglecting the first 2
bits cause we don’t need the word offset, we need the block
index

always @(posedgeclk)

Begin
Msready <= 0; // initializing it with zero to be changed
afterwards

if (MemWrite)

 begin
repeat (3) begin// Wait for 3 clock cycles
@ (posedge clk);
End
mem[addr] <= data_in;

Msready <= 1; // mem finished writing

End
else if (MemRead)
 Begin repeat (3) begin@ (posedge clk);
End
//reading a whole block,4 words
MsDataOut<={mem[bAddr],mem[bAddr+1],mem[bAddr+2],mem[bAddr+3]
};
Msready <= 1; // mem finished reading
End
End
initial begin
mem[0]=32'd17;
mem[1]=32'd9;
mem[2]=32'd25;
endendmodul
```

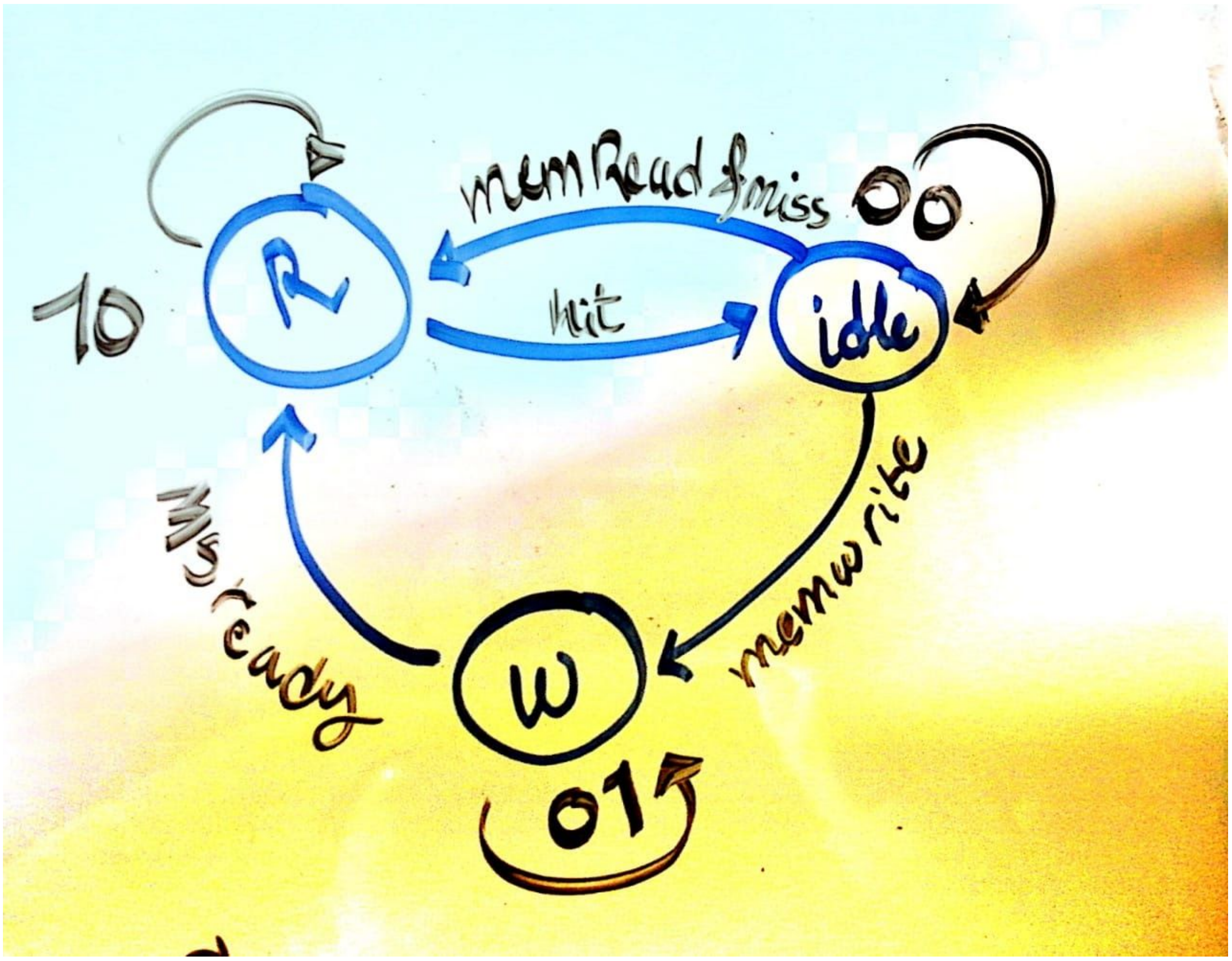
We did a testbench for it.
Here is the simulation :



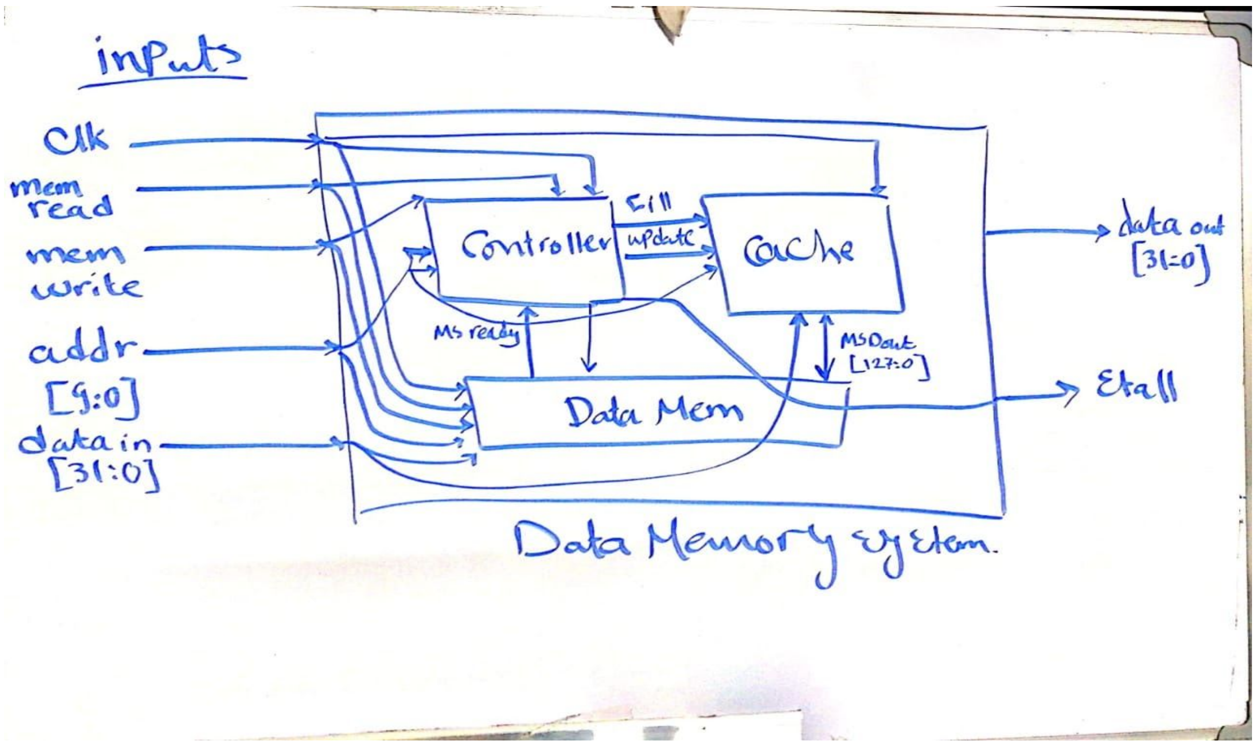
Experiment (2):

In this experiment, we Designed and implemented the Cache Memory Module.The cache memory module is essentially a small memory. It contains a small portion of the memory data to facilitate its access. It’s a 2D array of words. Vertically, representing blocks and horizontally representing words.





An FSM diagram for experiment 3 showing the state transitions and the input and output values



Since the drawing may be confusing , here is the code i followed :

```
`timescale 1ns / 1ps
module DataMemSys(
input clk, input MemRead, input MemWrite,
input [9:0] addr, input[31:0] data_in,
output [31:0] data_out, output stall);

wire wordoffset = addr [1:0];
wire [127:0] MsDataIn;
wire fill;
wire update;
wire index = addr [6:2];
wire tag = addr[9:7];

DataCache cachhe (clk,update, fill, wordoffset,index,
                    data_in,data_out,MsDataIn );

DataMem mm (clk,MemRead,MemWrite,addr,data_in, MsDataOut, Msready);

DataCacheController cont( clk,MemRead, MemWrite, index, tag, MsReady,stall,fill, update,
MsRead );


endmodule
```