实验二　32 位 MIPS 多周期处理器的设计

一、【设计原理】

多周期处理器与单周期处理器相比有以下改进：
(1) 使用一个指令和数据组合的存储器；
(2) 主译码器采用有限状态机的形式，产生相关控制信号；
(3) 只需要一个加法器。

二、【实验方案与部分关键代码】

相对于教材与 ppt 中的多周期处理器原理图，我扩展了 andi 路径以完成上板任务，扩展代码如下：
1.在 maindec.sv 中增加两个状态 ANDIEX 和 ANDIWB：
即 State 12 和 State 13。

```
localparam   ADDIEX  = 4'b1001; //State 9
localparam   ADDIWB  = 4'b1010; //State 10
localparam   JEX     = 4'b1011; //State 11
localparam   ANDIEX  = 4'b1100; //State 12
localparam   ANDIWB  = 4'b1101; //State 13

localparam   LW      = 6'b100011; //Opcode for lw
localparam   SW      = 6'b101011; //Opcode for sw
```

并且为这两个新状态分别赋予相应的 controls 值：

```
ADDIEX:  controls = 15'h0420;
ADDIWB:  controls = 15'h0800;
ANDIEX:  controls = 15'h0423;
ANDIWB:  controls = 15'h0800;
JEX:     controls = 15'h4008;
default: controls = 15'hxxxx;
```

另外增加操作码 ANDI：

```
localparam   SW      = 6'b101011; //Opcode for sw
localparam   RTYPE   = 6'b000000; //Opcode for R-type
localparam   BEQ     = 6'b000100; //Opcode for beq
localparam   ADDI    = 6'b001000; //Opcode for addi
localparam   J       = 6'b000010; //Opcode for j
localparam   ANDI    = 6'b001100; //Opcode for andi

logic [3:0]  state, nextstate;
logic [14:0] controls;
```

2.在 aludec.sv 中新增 alucontrol 指令 andi 3'b000：

```
always_comb
  case(aluop)
      2'b00: alucontrol <= 3'b010;              // add
      2'b01: alucontrol <= 3'b110;              // sub
      2'b11: alucontrol <= 3'b000;              // andi
      default:
          case(funct)                           // RTYPE
              6'b100000: alucontrol <= 3'b010; // ADD
              6'b100010: alucontrol <= 3'b110; // SUB
              6'b100100: alucontrol <= 3'b000; // AND
              6'b100101: alucontrol <= 3'b001; // OR
              6'b101010: alucontrol <= 3'b111; // SLT
```

3.数据与指令存储器 mem:

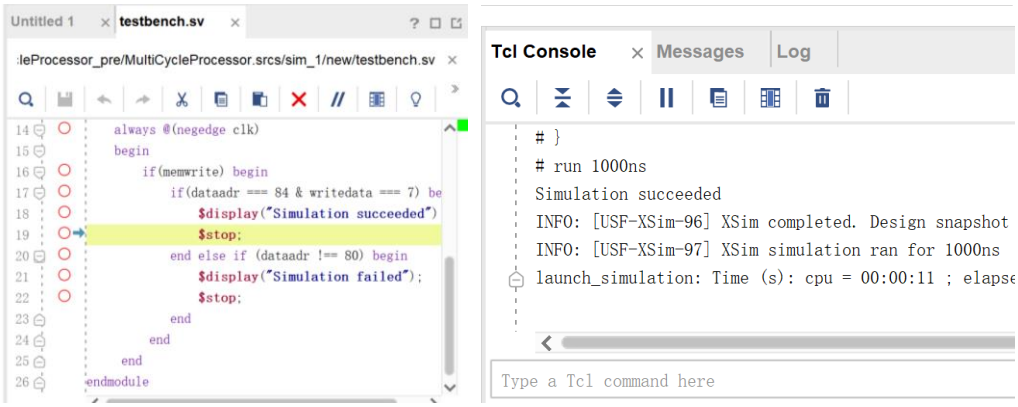
```
module mem(input  logic        clk, we,
           input  logic [5:0]  a,
           input  logic [31:0] wd,
           output logic [31:0] rd);
    logic [31:0] RAM[63:0];
    initial
       $readmemh("memfile1.dat", RAM);
    assign rd = RAM[a];
    always_ff @(posedge clk)
      if(we)
          RAM[a] <= wd;
endmodule
```

三、【仿真截图】

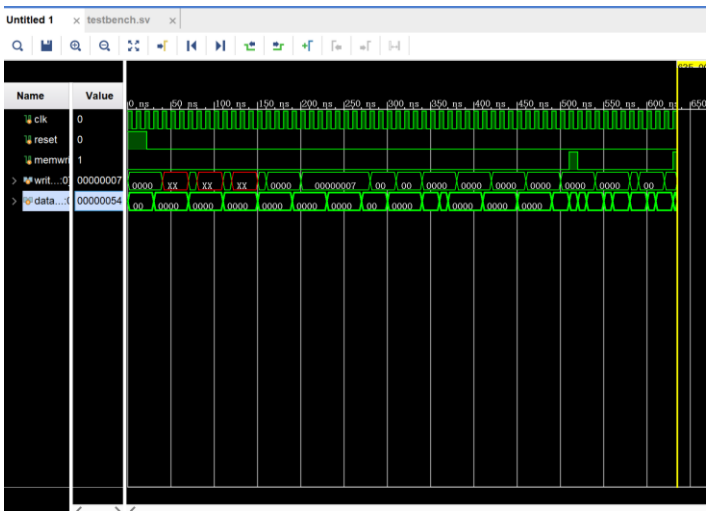
1.不添加 io 接口的代码仿真:
测试用汇编代码 memfile.dat 如下:
20020005
2003000c
2067fff7
00e22025
00642824
00a42820
10a7000a
0064202a
10800001
20050000
00e2202a
00853820
00e23822
ac670044
8c020050
08000011
20020001

ac020054

仿真的控制台输出结果：



仿真波形图：

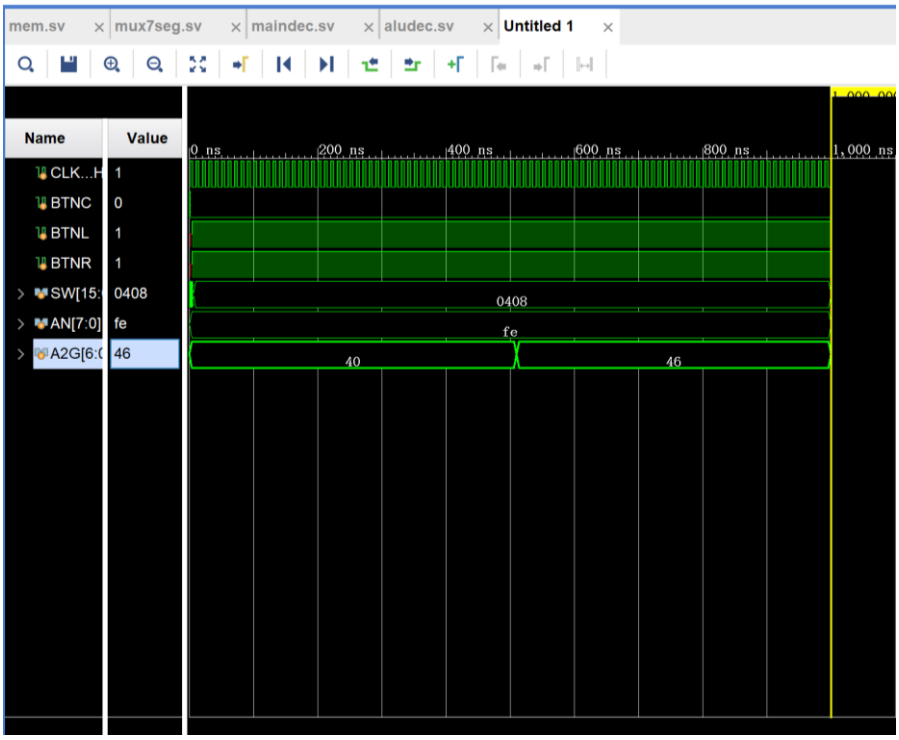


2.添加 io 接口的代码仿真：

测试用汇编代码 memfile1.dat 如下：

20100000
ac100080
8c110080
32320002
1240fffd
8c130088
8c14008c
0293a820
8c110080
32320001
1240fffd
ac150084
08000002

仿真波形图：



四、【实验开发板照片】

计算 12+34=46：

输入 SW = 16'b0001001000110100，按 BTNR 和 BTNL 计算：

按下 BTNC 和 BTNL，结果清零：