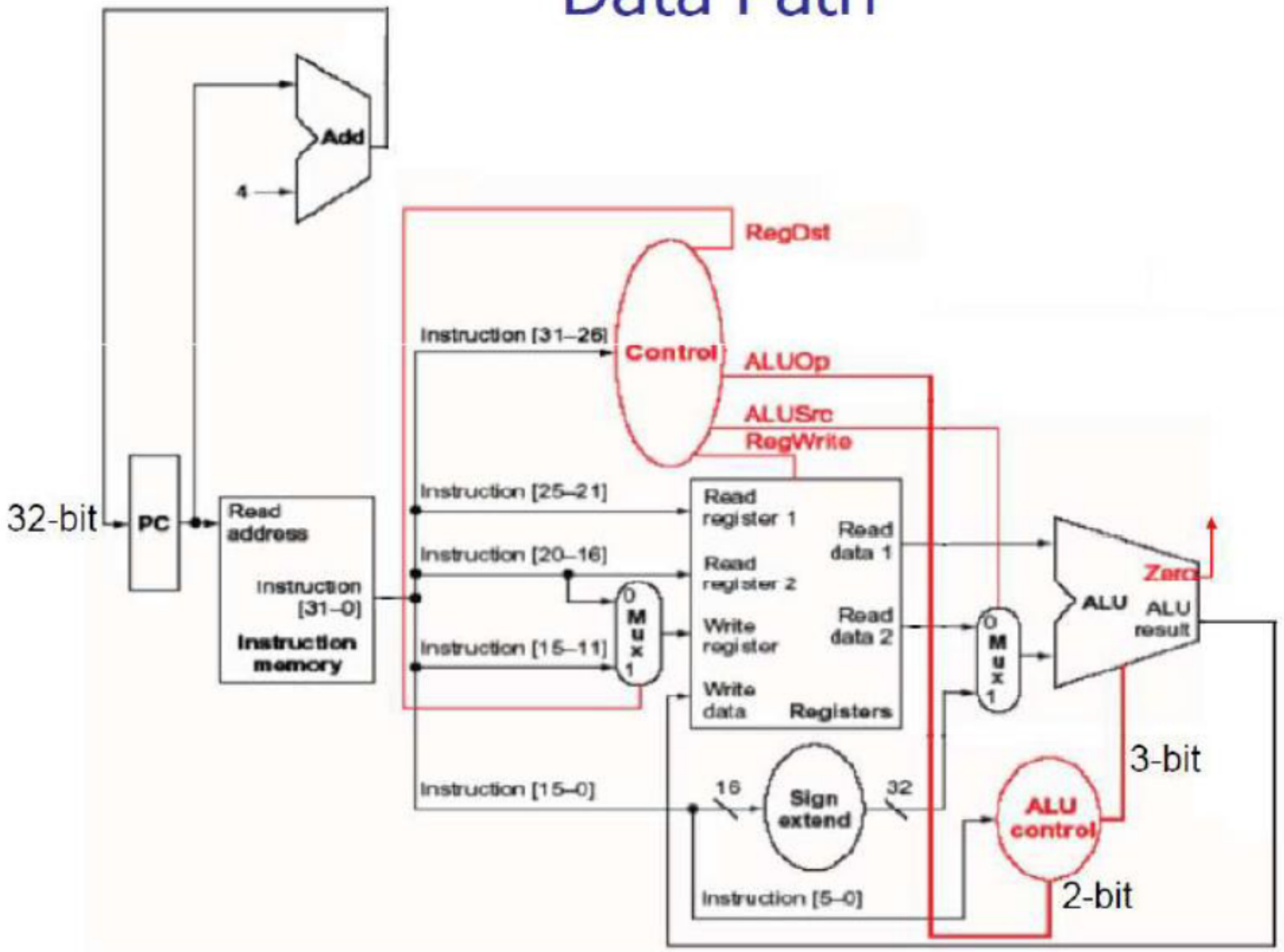
Homework 4

Homework 4: A Single Cycle CPU by Verilog

2017/11/08

Data Path

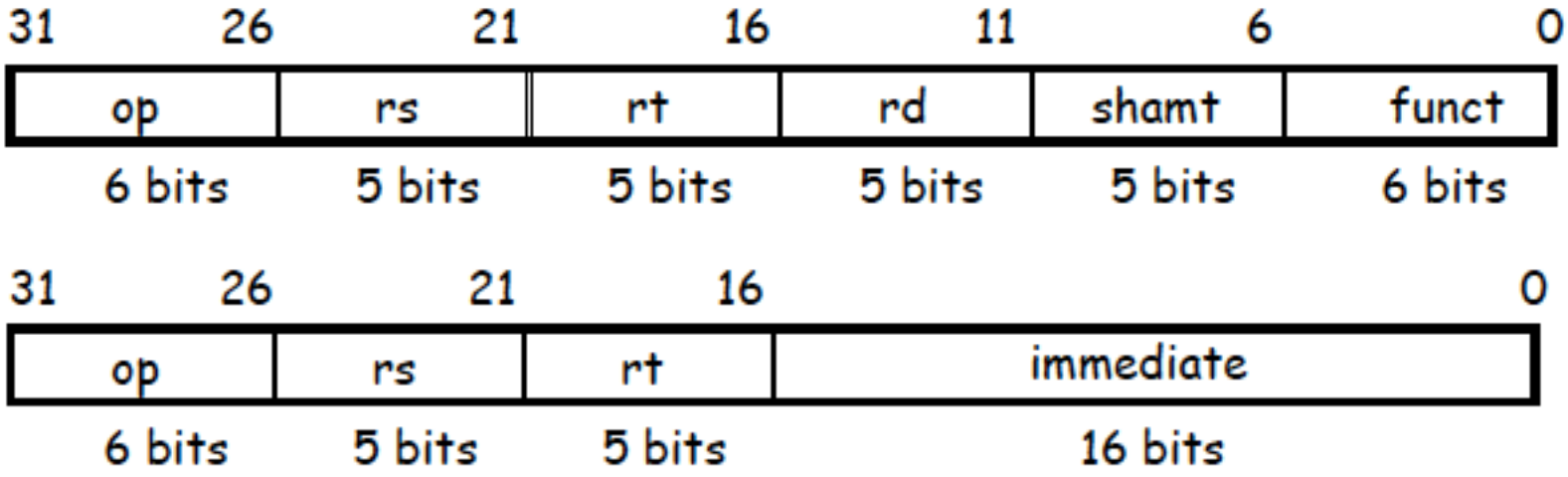

Add
4
RegDst
Instruction [31–26]
Control
ALUOp
ALUSrc
RegWrite
32-bit
PC
Read address
Instruction [31–0]
Instruction memory
Instruction [25–21]
Instruction [20–16]
Instruction [15–11]
0
M
u
x
1
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
ALU
Zero
ALU result
Instruction [15–0]
16
Sign extend
32
ALU control
3-bit
2-bit
Instruction [5–0]

Requirement #1

- Required Instruction Set
  - and
  - or
  - add
  - sub
  - mul
  - addi

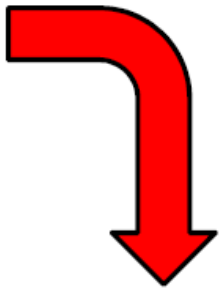
Requirement #2

- Translate the assembly code to machine code (next page)
- Register file: 32 registers
- Instruction Memory 1KBytes
- Machine code:

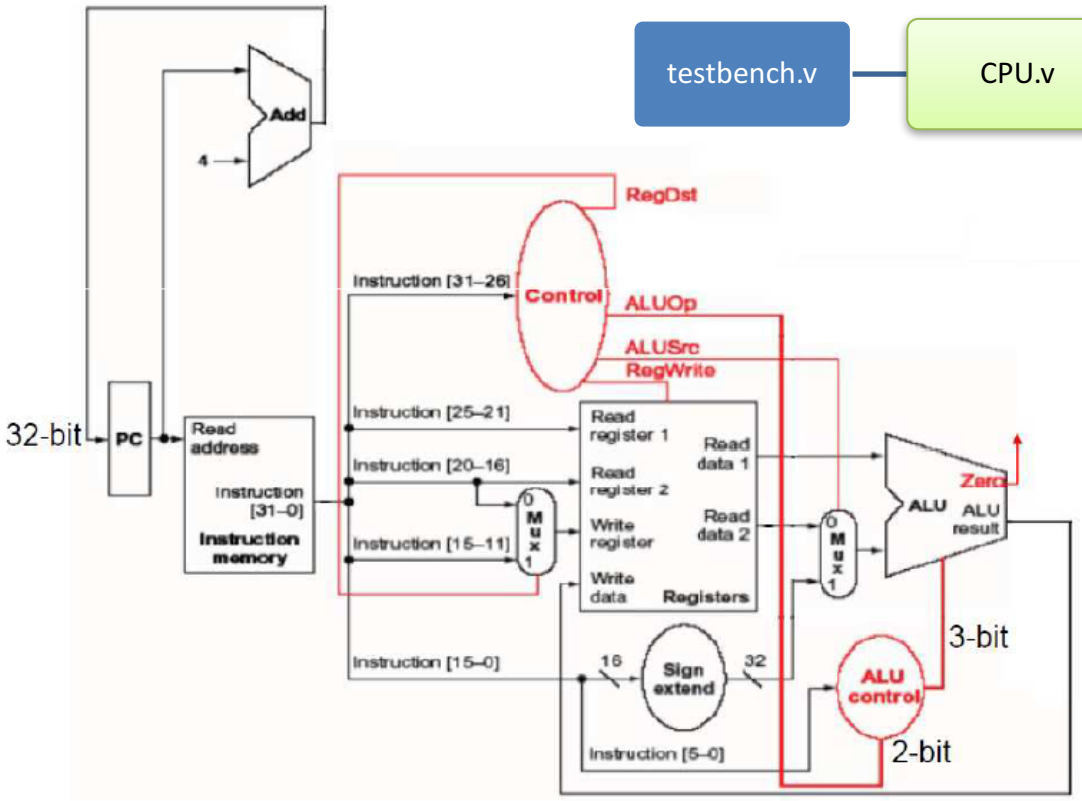


Instruction Translation

```
add $t0,$0,$0
addi $t1,$0,10
addi $t2,$0,13
mul $t3,$t1,$t1
addi $t1,$t1,1
sub $t2,$t2,$t1
and $t3,$t1,$t2
or $t4,$t2,$t3
```



```
000000_00000_00000_01000_00000_100000 //add $t0,$0,$0
001000_00000_01001_0000000000001010   //addi $t1,$0,10
001000_00000_01010_0000000000001101   //addi $t2,$0,13
000000_01001_01001_01011_00000_011000 //mul $t3,$t1,$t1
001000_01001_01001_0000000000000001   //addi $t1,$t1,1
000000_01010_01001_01010_00000_100010 //sub $t2,$t2,$t1
000000_01001_01010_01011_00000_100100 //and $t3,$t1,$t2
000000_01010_01011_01100_00000_100101 //or $t4,$t2,$t3
```



testbench.v

CPU.v

- Control.v
- Adder.v
- PC.v
- Instruction_Memory.v
- Registers.v
- MUX5.v
- MUX32.v
- Sign_Extend.v
- ALU.v
- ALU_Control.v

testbench.v

```
CPU CPU(
    .clk_i   (Clk),
    .rst_i   (Reset),
    .start_i(Start)
);

initial begin
    counter = 0;

    // initialize instruction memory
    for(i=0; i<128; i=i+1) begin
        CPU.Instruction_Memory.memory[i] = 32'b0;
    end


    // initialize Register File
    for(i=0; i<32; i=i+1) begin
        CPU.Registers.register[i] = 32'b0;
    end

    // Load instructions into instruction memory
    $readmemb("instruction.txt", CPU.Instruction_Memory.memory);

    // Open output file
    outfile = $fopen("output.txt") | 1;

    Clk = 0;
    Reset = 0;
    Start = 0;

    #(`CYCLE_TIME/4)
    Reset = 1;
    Start = 1;


end
```

CPU.v

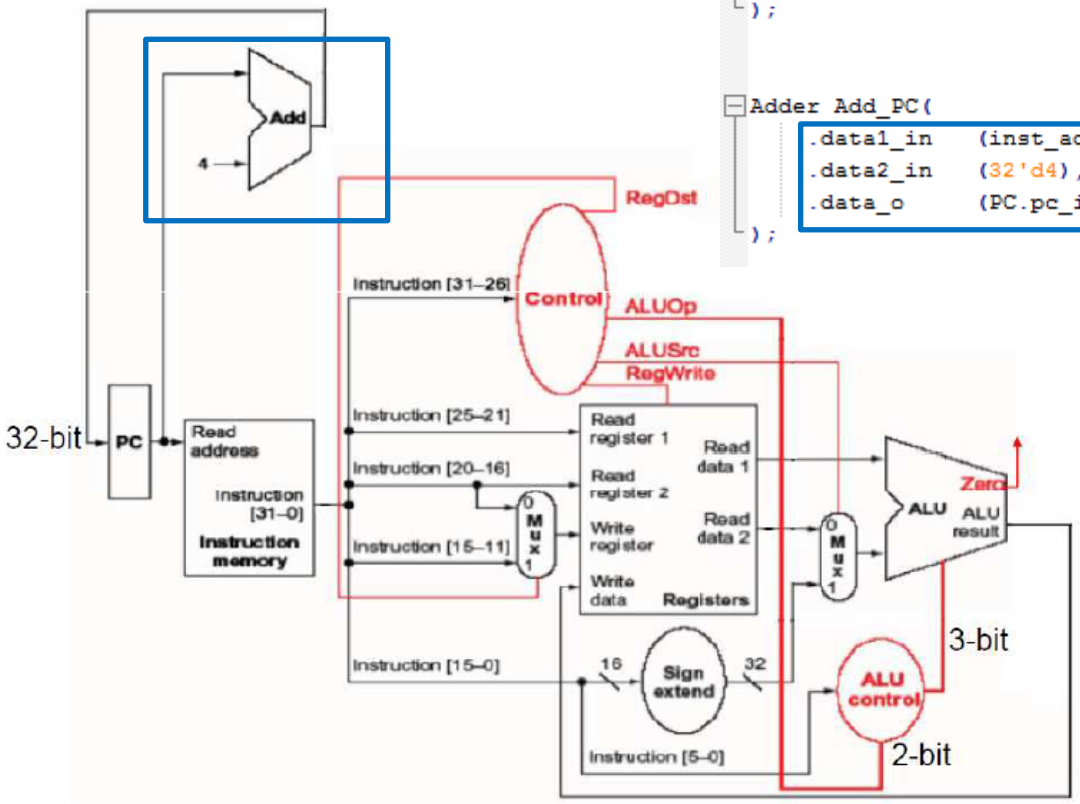
```
module CPU
(
    clk_i,
    rst_i,
    start_i
);

// Ports
input               clk_i;
input               rst_i;
input               start_i;

/*
Control Control(
    .Op_i       (),
    .RegDst_o   (),
    .ALUOp_o    (),
    .ALUSrc_o   (),
    .RegWrite_o ()
);
*/

/*
Adder Add_PC(
    .data1_in   (),
    .data2_in   (),
    .data_o     ()
);
*/

PC PC(
    .clk_i      (),
    .rst_i      (),
    .start_i    (),
    .pc_i       (),
    .pc_o       ()
);
```

CPU.v

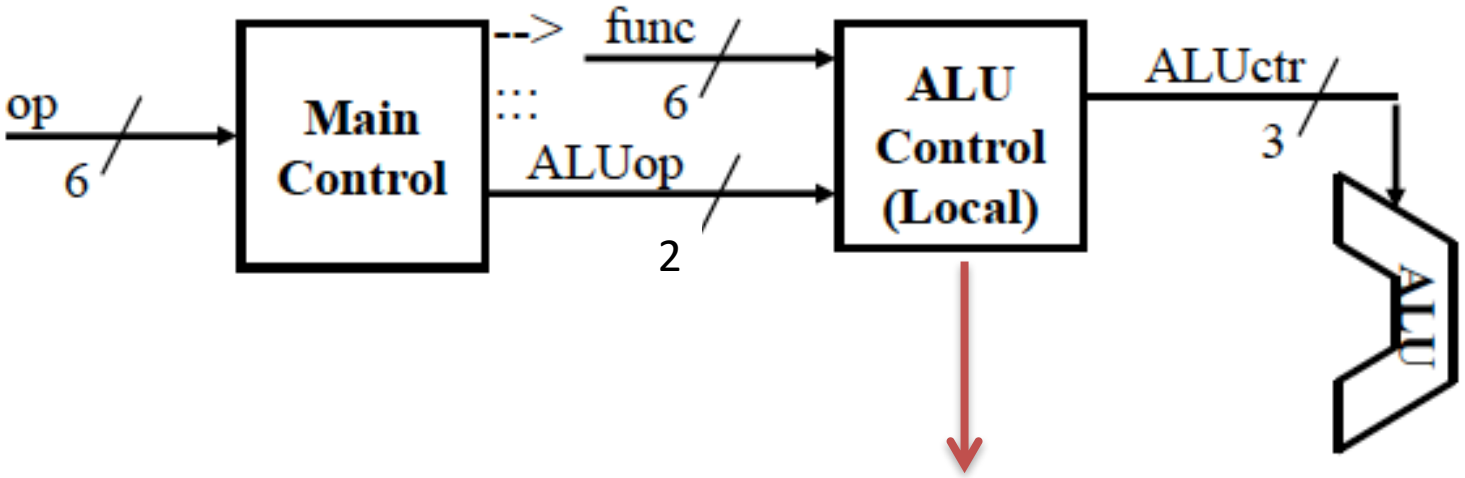
```
// Ports
input               clk_i;
input               rst_i;
input               start_i;

wire    [31:0]  inst_addr, inst;

Control Control(
    .Op_i       (inst[31:26]),
    .RegDst_o   (MUX_RegDst.select_i),
    .ALUOp_o    (ALU_Control.ALUOp_i),
    .ALUSrc_o   (/*???*/),
    .RegWrite_o (/*???*/)
);


Adder Add_PC(
    .data1_in   (inst_addr),
    .data2_in   (32'd4),
    .data_o     (PC.pc_i)
);
```

Adder.v

```
module Adder
(
    data1_in,
    data2_in,
    data_o
);

input   [31:0]  data1_in, data2_in;
output  [31:0]  data_o;

assign data_o = data1_in + data2_in;

endmodule
```



Control.v / ALU_Control.v



funct<5:0>	Instruction Op.
100000	add
100010	subtract
100100	and
100101	or
011000	mul

Homework 4

- Requirements (1)
  - Source codes (*.v files)
    - testbench.v
    - PC.v
    - Registers.v
    - Instruction_Memory.v
    - CPU.v
    - Adder.v
    - Control.v
    - ALU_Control.v
    - Sign_Extend.v
    - ALU.v
    - MUX32.v
    - MUX5.v
- Requirements (2)
  - Machine Code text file
    - Instruction.txt (no need to modify)
    - There is no need to submit "output.txt".
  - Report (hw4_b03902xxx.pdf)
    - Coding Environment
    - Module implementation explanation
    - Either English or Chinese is fine
    - **（No more than 2 pages）**

Homework 4

- Submission format
  - [dir] hw4_b03902xxx_v0
  - hw4_b03902xxx_v0 / hw4_b03902xxx.pdf
  - hw4_b03902xxx_v0 / src / *.v files
- Deadline: 2017/11/20 23:59

- Upload to CEIBA

Homework 4

- Evaluation criteria
  - Report : 15%
  - Code : 85 %
    - Correctness: 36%
    - module correct implementation: 49%
  - Wrong Format: -10%
  - Compile error: coding 0 %
    - Please make sure your code can compile before submitting.