CONTENT

In this part of the project, you will build an 8-bit microprocessor that supports the given instruction set. The following table summarizes the supported instructions and their OPCODE. More detailed information on the instruction set is introduced in lecture.

Syntax	**Description**	**Synopsis**	**Opcode**
ADD *rs*	ADD	Acc ← Acc + Reg[rs]	0001
SUB *rs*	SUB	Acc ← Acc – Reg[rs]	0010
NOR *rs*	NOR	Acc ← Acc | Reg[rs]	0011
MOVR *rs*	Move Register value to Acc	Acc ← Reg[rs]	0100
MOVA *rd*	Move Acc to Register	Reg[rd] ← Acc	0101
JZ *rs*	Jump if Zero to Reg Value	PC ← Reg[rs] if Zero	0110
JZ *Imm*	Jump if Zero to Immediate	PC ← Imm if Zero	0111
JC *rs*	Jump if Carry to Reg Value	PC ← Reg[rs] if Carry	1000
JC *Imm*	Jump if Carry to Immediate	PC ← Imm if Carry	1010
SHL	Shift Left Acc	Acc ← Acc << 1	1011
SHR	Shift Right Acc	Acc ← Acc >> 1	1100
LD *Imm*	Load Immediate to Acc	Acc ← Imm	1101
NOP	NOP	Do Nothing	0000
HALT	Halt	STOP Execution	1111

1. Modules and I/Os

The top level block diagram of the 8-bit microprocessor is given in the following figure. They are partitioned into several modules. The RTL Verilog code for the modules in blue boxes are provided to you. The ALU module should be taken from your lab 2. In this project, you need to complete the design of the controller, the top level, and the test bench. You also need to verify the behavior of the microprocessor at RT level.

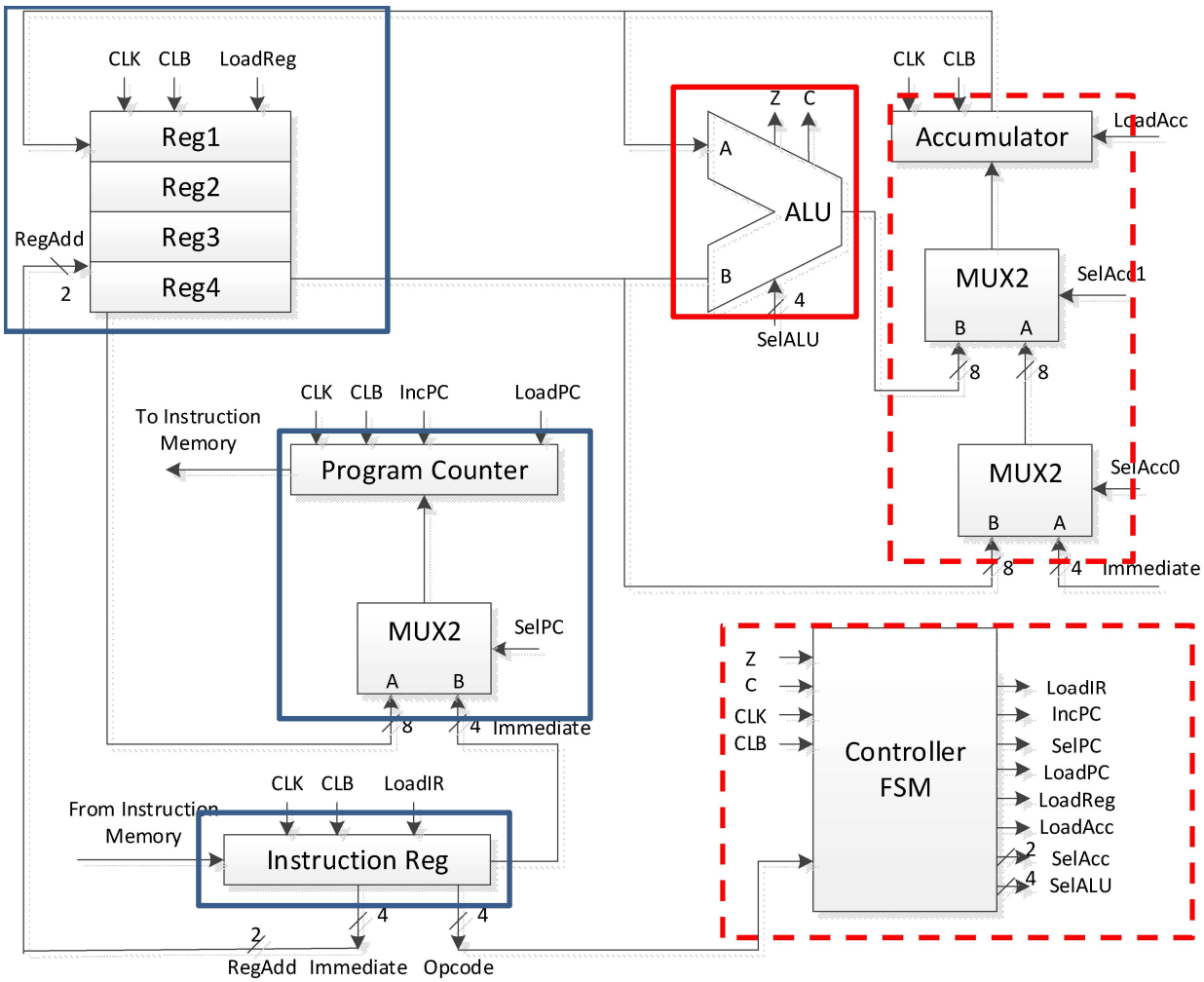

Figure 1: Reference Block Diagram of the 8-bit Microprocessor Architecture

The outputs from the controller are mainly the select and control signals to the datapath (select lines of multiplexers, de-multiplexers, address decoder for register bank, load/increment PC and so on).

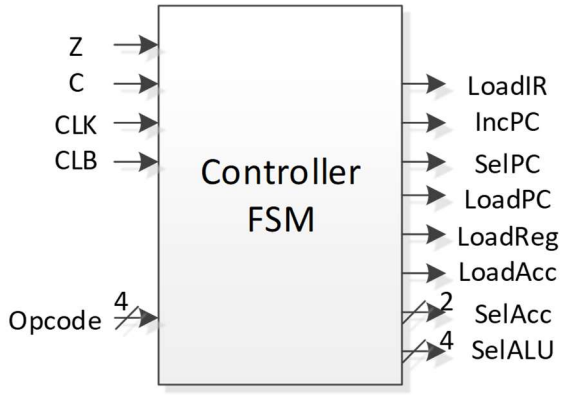

Figure 2: Block Diagram of Controller (Input/Output)

The input/output of the 8-bit microprocessor is specified in the following Table:

Name	Description	I/O	Width
CLK	Clock input	I	1 bit
CLB	Reset input (low active)	I	1 bit
INST[7:0]	Instruction bus coming from instruction memory	I	8 bit
PC[7:0]	Address bus goes into instruction memory	O	8 bit
ACC[7:0]	ACC register value (used for debugging purpose)	O	8 bit

2. Test Bench Design

Your test bench should consist of two memory banks. one of them emulate the instruction memory. It should be initialized with the binary (hexadecimal) code of your test program. The other bank of memory stores the cycle accuracy expected value of the output (i.e. PC and ACC). The test bench should be able to report an error when there is a mismatch between the actual and expected output.

Your test program should cover different execution scenarios and all instructions in the instruction set. An example testing program is given below:

0.RESET = 1

1.LD ACC IMM(0100) /*ACC = 4*/ ----PC = 0
2.MOV REG01 ACC /* R1 = 4 */ --- PC = 1 and will increment at each instruction afterwards

3.LD ACC IMM(1010) /*ACC = 10 */
4.MOV REG02 ACC /* R2 = 10 */

5.LD ACC IMM(0010)
6.MOV REG03 ACC /* R3 = 2 */

7.SUB REG01 ACC /*ACC = ACC - REG01 = 2 - 4 */
8.JC REG02 /*check if PCOUT is 1010*/ ----- PC = 7

9.LD ACC IMM(1001) ---- PC = 10 (PC jumped from 7 to 10)
10.SHL ACC ---- PC = 11
11.SHL ACC ---- PC = 12

12.SHL ACC
13.SHL ACC
14.MOV REG00 ACC /*R0 = 10010000*/

15.MOV ACC REG03 /* ACC = 2 */
16.ADD ACC REG03 /*ACC = 4 */
17.SUB REG01 ACC /* ACC = ACC - R1 = 4 - 4 */

18.JZ IMM(1000) /*check if PCOUT is 1000*/ ---- PC = 19

19.LD ACC IMM(1101) /* ACC = 1101 */ ---- PC = 8 (PC jumped from 19 to 5)
20. Halt --- PC = 9

3. What to Turn In

Complete the Verilog design for the controller, the ACC, the top level system, and the test bench of the system. Create a test vector to check the function your processor. The data path of this simple processor is not pipelined. Each instruction takes two clock cycles. Please hand in the following items:

1. The zipped file of your working directory (including all Verilog source code, and the test vectors)
2. A report, which includes
    a. The FSM diagram of your controller
    b. The assembly code of your test program
    c. The simulation waveform, in which the important steps (that indicate the correct function of the microprocessor) are labeled and explained.

I will first run your design using your own test bench, then copy your ALU.v, controller.v and processor.v to my working directory and test it using my testbench and test vector. It is extremely important that you follow the given I/O specifications, make sure that the name of the top level module is called "**processor**", and do not change the provided Verilog code, otherwise, your design will not work in my environment.