ES204 Digital Systems
Lab Exam

Indian Institute of Technology, Gandhinagar

220 marks

Dt. 13.04.2024

Final Demo:

FPGA implementation of the Complete “Tiny” Processor Design which runs one program.

Submit:
- Verilog Codes
- Testbench
- XDC Files
- Block diagram showing all the Verilog modules and how then interact
- Simulation results
- FPGA implementation Video any one program with at least 4 instructions.

(Max 30 marks for report.)

The following processor has a register file consisting of 16 registers each of 8 bit. The processor can execute the following instructions. The instructions that need 2 operands will take one of the operand from the Register file and another from the accumulator. The result will be transferred to Accumulator. There is an 8-bit extended (EXT) register used only during multiplication and division operation. This register stores the higher order bits during multiplication and quotient during division. The C/B register holds the carry and borrow during addition and subtraction, respectively.

Note:
- Each instruction takes 1 clock cycle.
- Division operation can never have the 0 as divisor.
- Branch instruction can only branch within the program.

Instruction format:

Direct instruction

Operation code	Register address

Branch Instruction

Operation code	4-bit address (label)

Instruction set:

Instruction Opcode	Operation	Explanation
0000 0000	NOP	No operation
0001 xxxx	ADD Ri	Add ACC with Register contents and store the result in ACC. Updates C/B
0010 xxxx	SUB Ri	Subtract ACC with Register contents and store the result in ACC. Updates C/B
0011 xxxx	MUL Ri	Multiple ACC with Register contents and store the result in ACC. Updates EXT
0100 xxxx	DIV Ri	Divides ACC with Register contents and store the Quotient in ACC. Updates EXT with remainder.

0000 0001	LSL ACC	Left shift left logical the contents of ACC. Does not update C/B
0000 0010	LSR ACC	Left shift right logical the contents of ACC. Does not update C/B
0000 0011	CIR ACC	Circuit right shift ACC contents. Does not update C/B
0000 0100	CIL ACC	Circuit left shift ACC contents. Does not update C/B
0000 0101	ASR ACC	Arithmetic Shift Right ACC contents
0101 xxxx	AND Ri	AND ACC with Register contents (bitwise) and store the result in ACC. C/B is not updated
0110 xxxx	XRA Ri	XRA ACC with Register contents (bitwise) and store the result in ACC. C/B is not updated
0111 xxxx	CMP Ri	CMP ACC with Register contents (ACC-Reg) and update C/B. If ACC>=Reg, C/B=0, else C/B=1
0000 0110	INC ACC	Increments ACC, updates C/B when overflows
0000 0111	DEC ACC	Decrements ACC, updates C/B when underflows
1000 xxxx	Br <4-bit address>	PC is updated and the program Branches to 4-bit address if C/B=1
1001 xxxx	MOV ACC, Ri	Moves the contents of Ri to ACC
1010 xxxx	MOV Ri, ACC	Moves the contents of ACC to Ri
1011 xxxx	Ret <4-bit address>	PC is updated, and the program returns to the called program.
1111 1111	HLT	Stop the program (last instruction)

Note:

- Avoid trying to design one large monolithic code. See how you can partition the design into smaller modules and then implement them, test them using detailed test bench codes, and synthesize them.
- Combine them together after detailed testing and synthesis is done.
- Finally write a meaningful program based on this set and show its running using FPGA.

The final implementation should be shown on FPGA.

Sample code:

Add the contents of R5 and R6, and store the result in R7.

```
MOV ACC, R1        ; Load R1 in ACC
XRA R1             ; clears ACC
ADD R5             ; ACC+R5
ADD R6             ; ACC + R6 (which is R5+R6)
MOV R7, ACC
HLT
```