CPE 300L
DIGITAL SYSTEM ARCHITECTURE AND DESIGN LABORATORY

FINAL PROJECT
8 BIT MICROPROCESSOR DESIGN

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
UNIVERSITY OF NEVADA, LAS VEGAS

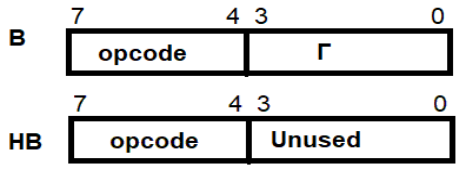
OBJECTIVE

Design and test a simple 8-bit Microprocessor (SMP8).

INTRODUCTION

The SMP8 is an 8-bit processor (instructions and data are 8-bit length). SMP8 has one 8-bit general purpose register, R, and an 8-bit Accumulator Register, AC. The result of arithmetic operation is to be automatically loaded into AC register. One of the operands in two-operand arithmetic/logic instructions (ADD, SUB, XOR, OR, AND) is always supplied from AC, and another comes from R register. There is 1-bit flip-flop, Zero flag, Z. It is set to 1, if the result of any arithmetic or logic instruction is 0. Otherwise, it is 0. This is to be done automatically by hardware. There is also PC-program counter to form the instruction address. Both Data and Instruction memories are 16x 1 byte memories.
Table 1 provides the description of the instruction set. It can be seen that instruction opcode is 4-bit long; there are 16 instructions in the set. In many instructions the low order 4 bits are unused. Here, Γ indicates a four-bit memory address of either data or instruction memory.

Two formats are used:



Conditional Branch instructions are JMPZ (if FlagZ=1) and JMPNZ (if Flag Z=0). Unconditional branch is JUMP. Memory Load and Store are implemented with Accumulator: LDAC instruction reads from Data Memory into AC, and STAC writes the contents of AC into the Data Memory. There is only one type of memory addressing: direct (absolute). There are two instructions to move data from and to Accumulator: MVAC (From AC to R) and MOVR (from R to AC). Increment(INAC) and NOT are performed on the operand that comes from AC register and the result goes back to Accumulator. Clear Accumulator (CLAC) sets Accumulator to 0. NOP does nothing (void).

Table 1: Instruction description

Instruction	Instr Code	Operation
NOP	0000	No operation
LDAC	0001 Γ	$AC = M[\Gamma]$
STAC	0010 Γ	$M[\Gamma] = AC$
MVAC	0011	$R = AC$
MOVR	0100	$AC = R$
JUMP	0101 Γ	GOTO Γ
JMPZ	0110 Γ	IF (Z=1) THEN GOTO Γ
JPNZ	0111 Γ	IF (Z=0) THEN GOTO Γ
ADD	1000	$AC = AC + R$, If $(AC + R = 0)$ Then $Z = 1$ Else $Z = 0$
SUB	1001	$AC = AC - R$, If $(AC - R = 0)$ Then $Z = 1$ Else $Z = 0$
INAC	1010	$AC = AC + 1$, If $(AC + 1 = 0)$ Then $Z = 1$ Else $Z = 0$
CLAC	1011	$AC = 0$, $Z = 1$
AND	1100	$AC = AC \wedge R$, If $(AC \wedge R = 0)$ Then $Z = 1$ Else $Z = 0$
OR	1101	$AC = AC \vee R$, If $(AC \vee R = 0)$ Then $Z = 1$ Else $Z = 0$
XOR	1110	$AC = AC \oplus R$, If $(AC \oplus R = 0)$ Then $Z = 1$ Else $Z = 0$
NOT	1111	$AC = AC'$, If $(AC' = 0)$ Then $Z = 1$ Else $Z = 0$

NOTES:

- The simplicity of the processor suggests a single cycle implementation. So, follow that path of design.
- PC is to be incremented using an increment circuit. You can merge PC register and up-counter. At Start and Reset, PC is to be set to 0.
- Start with the datapath design: design ALU, and then add all other components.
- Designate control signals required for Datapath operation. That includes multiplexers for selecting inputs or outputs, Clock gating signals for loading data into registers. Z flip-flop is to be updated every time AC register is getting data in.
- Create the control word table.
- Design the control unit. Opcode decoder (4-to-16) is a major part of it. The control unit is purely combinational circuit.
- Design on-chip memories of indicated depth/width.

DELIVERIES

1. **Design the block diagram of the SMP8 (similar to the Fig 1 of MIPS Lab)**

2. **Design and test**

a. Write a Verilog code to design SMP8 and test your microprocessor with the following test codes:

Test Code 1	Test Code 2
0: LDAC 0 1: INAC 2: JPNZ 4 3: JUMP 0 4: INAC 5: MVAC 6: ADD 7: STAC 2 8: NOP Data Memory 0: 55 1: 29 //Initialize all other locations (2-15) as 0	0: CLAC 1: INAC 2: MVAC 3: NOT 4: XOR 5: JMPZ 10 6: STAC 4 7: NOP 8: NOP Data Memory //All Locations initialize to 0

b. Write a testbench and test your design in VCS.

c. Implement the design in DE2 Board. You can apply reset and clock via pushbutton or a toggle switch, implement PC, INSTR and AC in a 7-seg display (you can display in HEX).

d. Record the video of (c) explaining the operation of each instruction from two test codes.

3. **Timing Requirement**

Perform the Timequest analysis to find the optimal clock and provide the total time to complete the code for each test codes: Cycle time x number of instructions

FINAL REPORT

Include the following elements in your report:

1. Include the control word table for the control unit.
2. Include the machine code for the test code 1 and 2.
3. Deliveries

Experiment	Delivery
1	a. Block Diagram of SMP8
2	a. Verilog code of the design b. Testbench code for test code 1 and 2 c. VCS waveform showing the result of the test code 1 and 2 (should be in hex and clear operation of each executed instructions) d. DE2 video delivery
3	a. Screenshot of the timing report before and after setting up timing constraints. b. Calculate how long it takes to complete Test Code1 and Test Code 2.

4. Conclusions
    a. Write down your conclusions, things learned, problems encountered during the lab and how they were solved, etc.
5. Submit the zip file of the DE2 project.