Design of a simple 8-bit Microprocessor

One of my key RTL projects was the **design of a simple 8-bit microprocessor** as part of my VLSI laboratory coursework. The microprocessor supported six registers (A–F) and implemented a small instruction set with five operations: **ADD, SUB, MOV, IN, and OUT**. The design used an 8-bit input bus, an 8-bit output bus, and 8-bit instruction codes.

The project involved building the datapath and control modules in Verilog, including:

- A **3-to-8 decoder** for register selection.
- An **adder/subtractor unit** for arithmetic operations.
- An **ALU module** to handle instruction decoding and execution.
- A **microprocessor top module** integrating registers, control logic, and data path.
- A **testbench** to verify correct functionality with various instruction sequences.

My role in the project included:

- Writing the Verilog RTL for the ALU, decoder, and microprocessor integration.
- Debugging issues related to data hazards and synchronization. Initially, the design latched onto incorrect values due to improper clocking, which I resolved by introducing a positive edge-triggered clocking scheme.
- Implementing pipelining in the microprocessor, ensuring data and control signals were properly staged across cycles.
- Creating the testbench, running simulations, and analyzing waveforms to validate the design using Vivado.

Through this project, I gained hands-on experience with RTL design, debugging hazards, and writing verification environments, as well as practical understanding of how a simple processor datapath is implemented in hardware.

Objective:

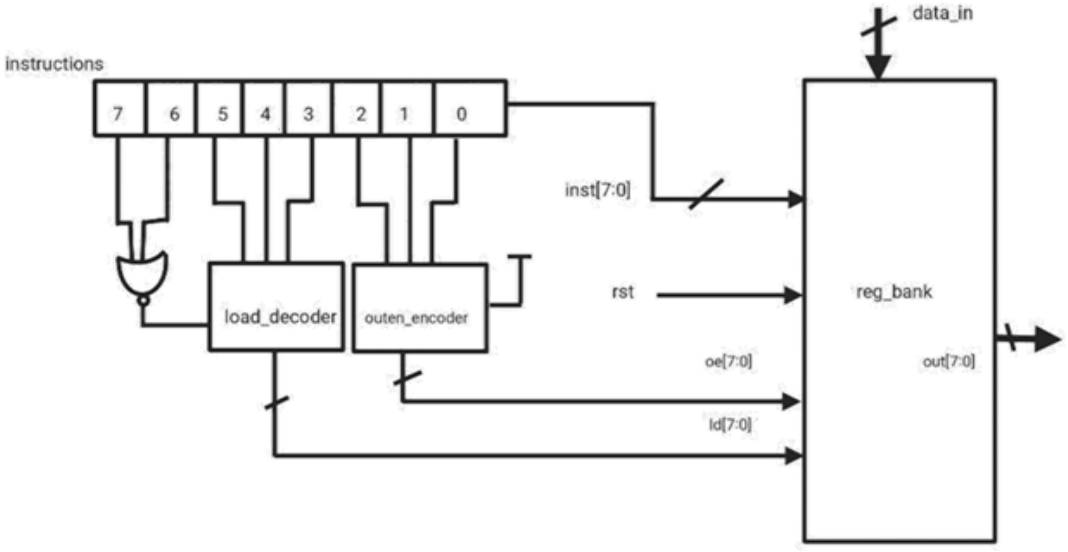
Design a sample microprocessor with 6 registers and 5 operations. Input bus and Output bus need to be of 8-bit width. The instruction set consists of 8-bit instructions. The allowed operations are ADD, SUB, MOV, IN and OUT.

Assume the 6 registers as A, B, C, D, E and F. Say R, R1 and R2 is used to represent one of the registers. The operations are defined as follows.

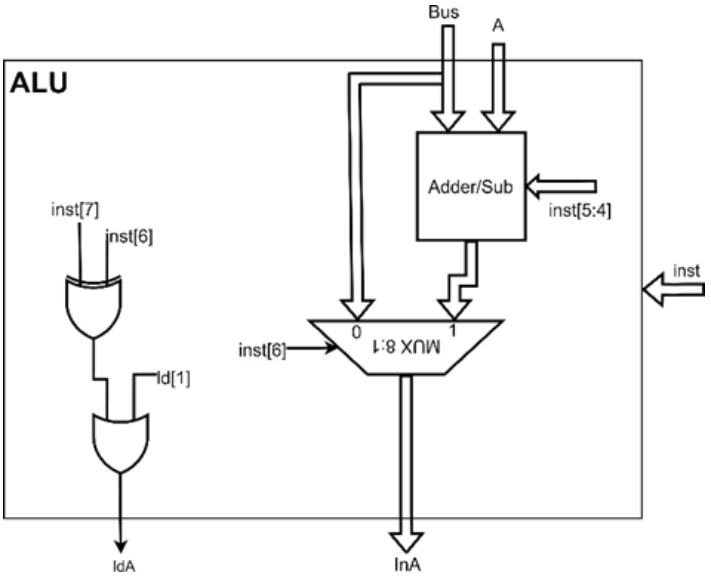
- **ADD R:** Add the value in R to value in A and update A to the obtained sum.
- **MOV R1 R2:** Copy the value of R2 into R1.
- **IN R:** Input an 8-bit number to register R.
- **OUT R:** Output the 8-bit number in register R.

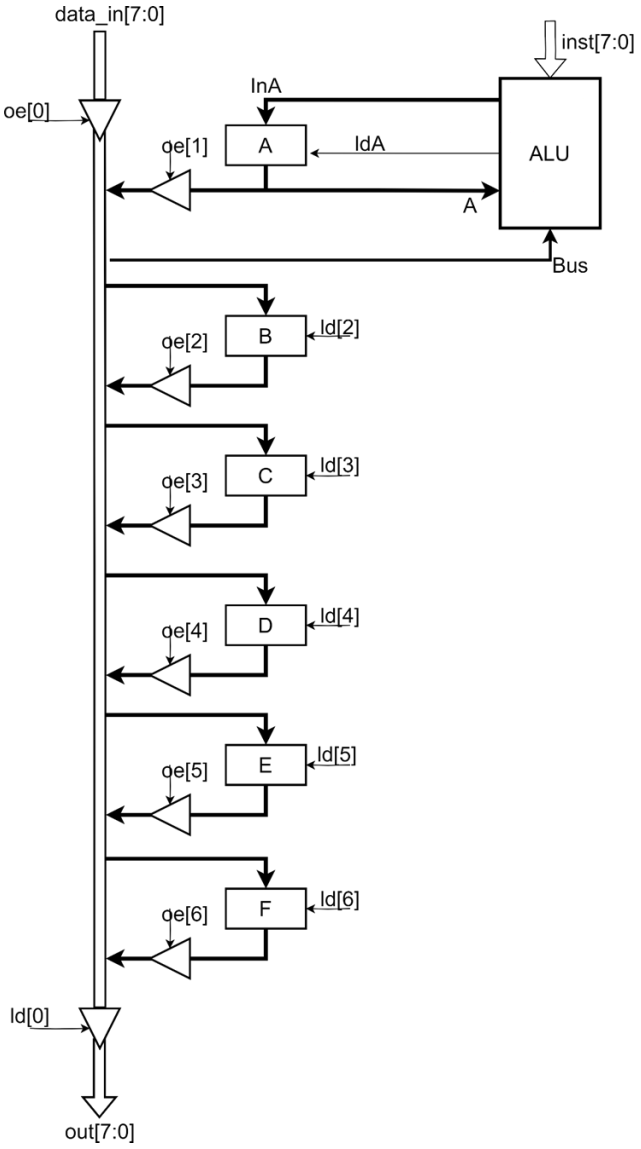
Assign 8-bit instructions to each of the above 5 operations.

Architecture



Main Architecture



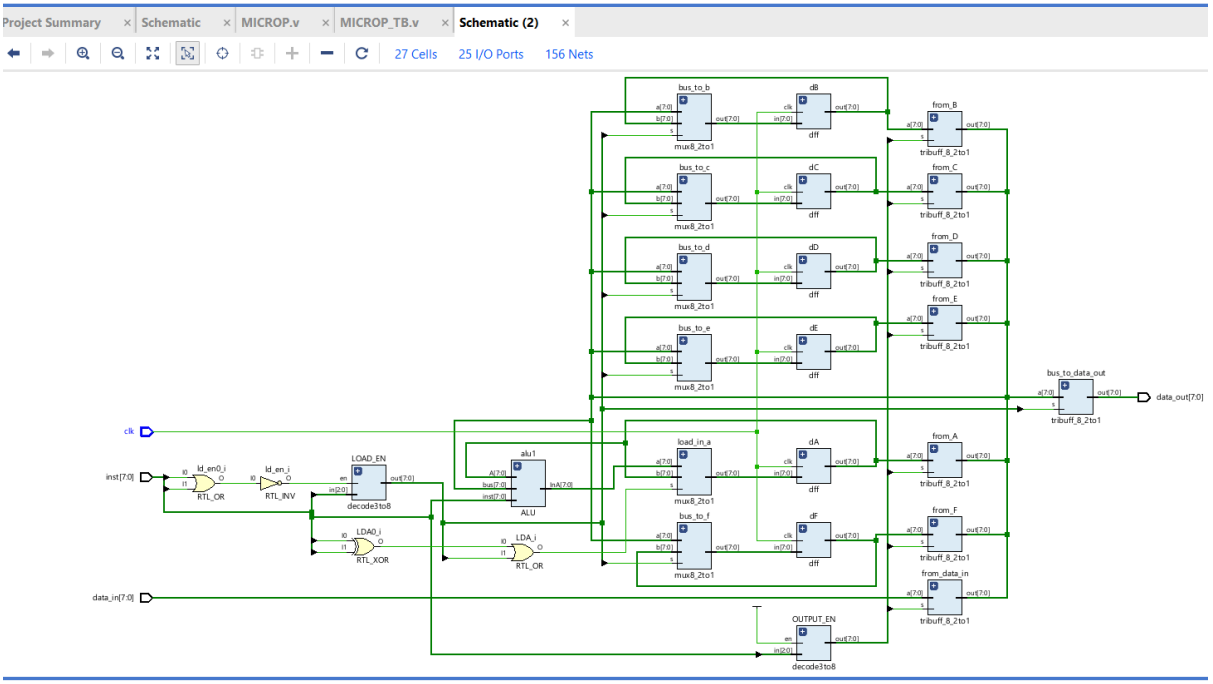


Reg bank architecture

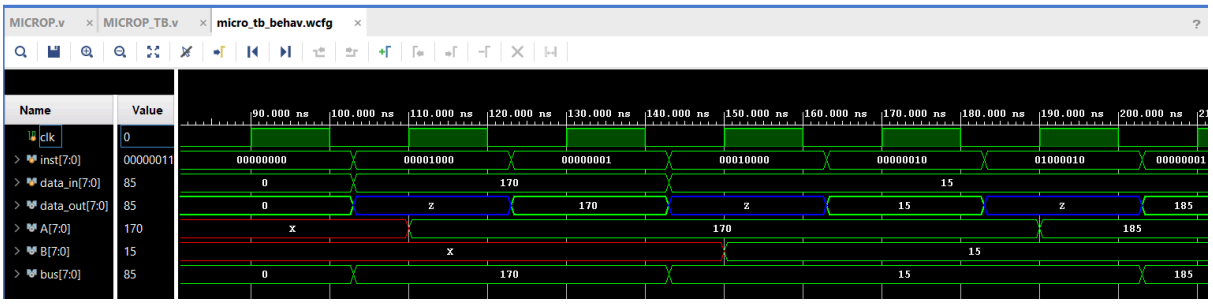
Instruction Set:

Inst[7:6]	Inst[5:3]	Inst[2:0]	Operation	
00	DDD	000	IN D	D ← input
00	DDD	SSS	Mov D,S	D ← S
00	000	SSS	OUT S	output ← S
01	0XX	S	ADD S	A ← A+S
01	1XX	S	SUB S	A ← A-S

➢ **Schematic diagram**



➢ Waveforms:


MICROP.v
MICROP_TB.v
micro_tb_behav.wcfg
Name
Value
clk
0
inst[7:0]
00000011
data_in[7:0]
85
data_out[7:0]
85
A[7:0]
170
B[7:0]
15
bus[7:0]
85
90.000 ns
100.000 ns
110.000 ns
120.000 ns
130.000 ns
140.000 ns
150.000 ns
160.000 ns
170.000 ns
180.000 ns
190.000 ns
200.000 ns
00000000
00001000
00000001
00010000
00000010
01000010
00000001
0
170
15
0
z
170
z
15
z
185
x
170
185
x
15
0
170
15
185