CO224 – Computer Architecture

Lab 05 – Part 05

Documentation

Group 07

- **E/18/028 – P.H.J.U. Ariyawansha**
- **E/18/285 – S.M.T.S.C. Ranasinghe**

In this part, we have extended our processor to support following instructions

- Logical Left Shift
- Logical Right Shift

Above instructions are encoded in the format shown below

OP-CODE	RD	RT	IMM
(bits 31-24)	(bits 23-16)	(bits 15-8)	(bits 7-0)

Fig 01

OP-CODE: To identify the instruction's operation

RD: A Register to be written in the register file

RT: A Register to be read from register file

IMM: Immediate value (How many times to be shifted)

Logic gate implementation

Basic element of the gate level circuit is the shifter comprising two AND gates and OR gate.

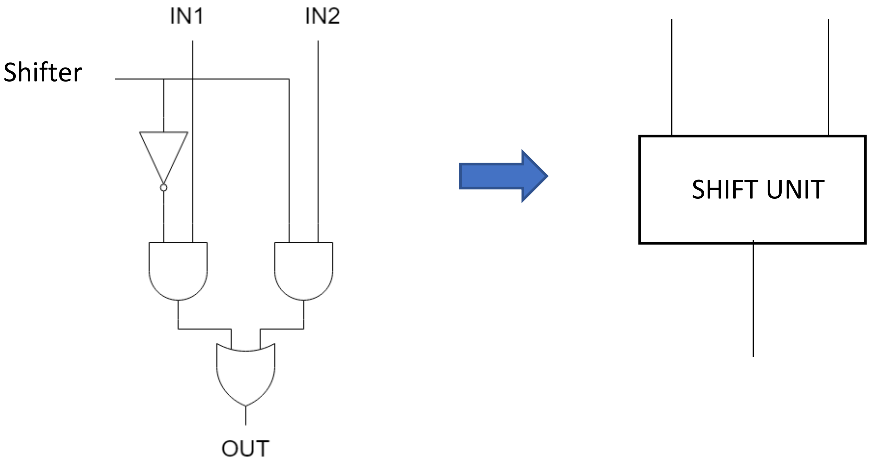


Fig 02

For **left shift** 8-bit value by 1, we have used 8 elements as below,

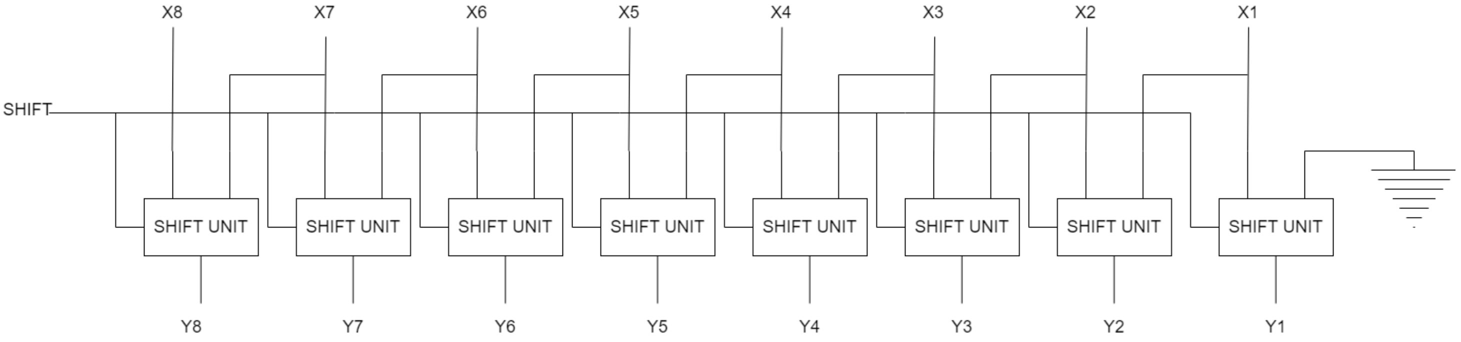


Fig 03

In above circuit diagram,

Input is $X_8X_7X_6X_5X_4X_3X_2X_1$, while the output is $Y_8Y_7Y_6Y_5Y_4Y_3Y_2Y_1$

When SHIFT equals 1, the shifted by 1 bit value is given as output.

When SHIFT equals 0, the Input is given as the output.

For shift multiple times we have to use multiple units as in Fig 03.

Since we use 8-bit register values, we use 4 units as in Fig 03. So that, we can use 4-bit SHIFT signal. Then we can shift the values by **up-to 8 bits**.

To **right shift** we used almost a mirror-image of Fig 03.

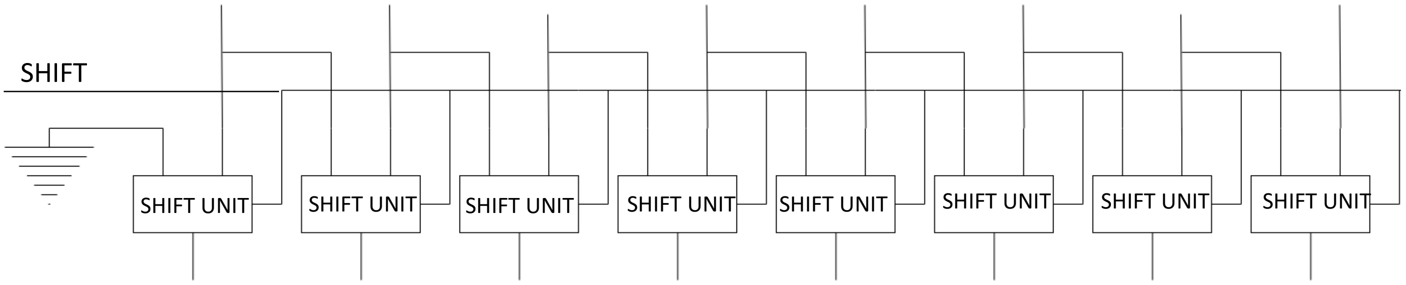


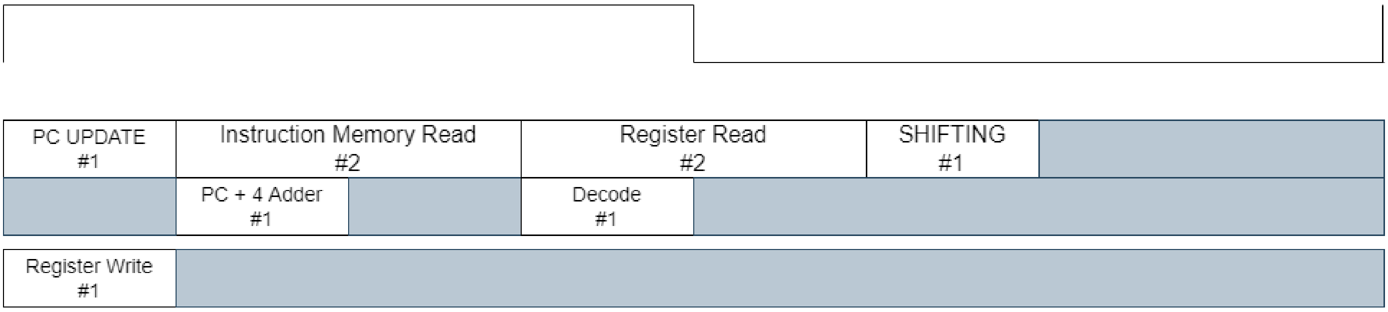
Fig 03

Logical Shift Left

```
sll <destination register> <register to read> <shifting amount>
```

- In `<shifting amount>,` we should give values in range 0 to 8 in **Hexadecimal.**
- Opcode for `sll` is `00001010.` ALUOP for `sll` is `100.`
- To realistically simulate the latencies of hardware we have included **1 time unit** artificial delay.
- This instruction module is designed by Verilog gate level modeling. For that we have used 32 NOT gates, 32 OR gates & 64 AND gates.

Timing Diagram for `sll`



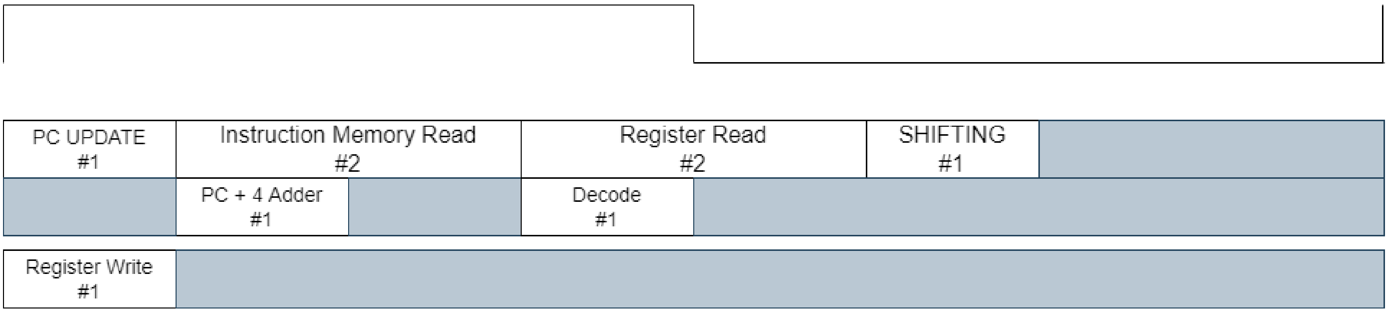
Note that this timing diagram is almost same as `OR` & `AND` instruction's timing diagrams.

Logical Shift Right

```
srl <destination register> <register to read> <shifting amount>
```

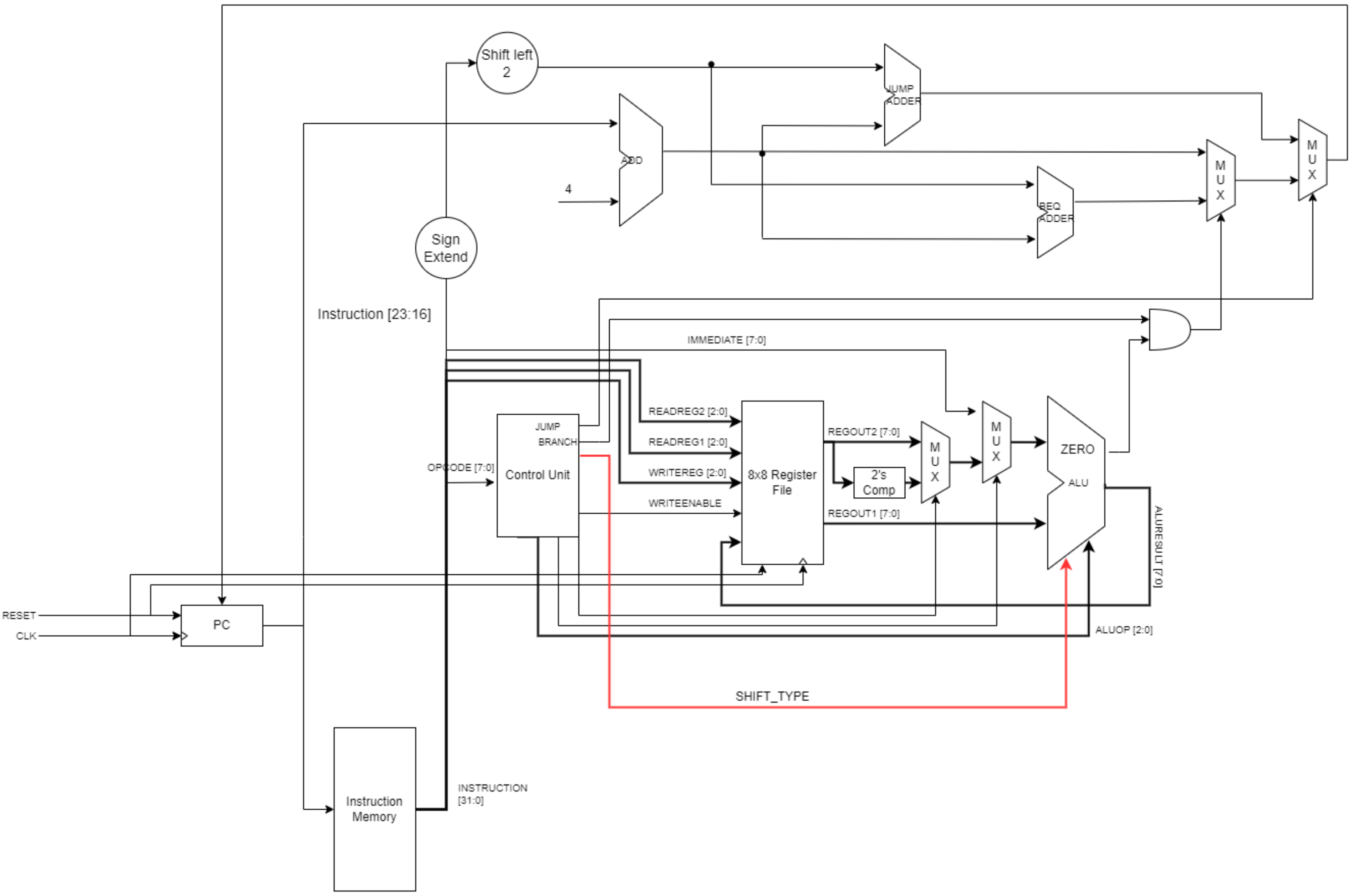
- In `<shifting amount>,` we should give values in range 0 to 8 in **Hexadecimal.**
- Opcode for `srl` is `00001011.` ALUOP for `srl` is `100.`
- To realistically simulate the latencies of hardware we have included **1 time unit** artificial delay.
- This instruction module is designed by Verilog gate level modeling. For that we have used 32 NOT gates, 32 OR gates & 64 AND gates.

Timing Diagram for `srl`



Note that this timing diagram is almost same as `OR` & `AND` instruction’s timing diagrams.

Changes made to Datapath



- We have created new control signal to ALU for sake of selecting whether operation is shift right or shift left.
- In ALU, we have instantiated the shifting module for both shift-left and shift-right.