ACTIVITY 4B: SHIFT REGISTER

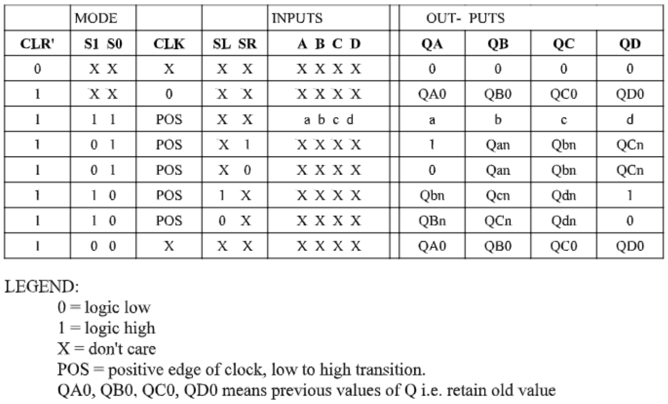
Jason Harvey Lorenzo, March 19, 2024
ENGG 125.03, Department of Electronics, Computer, and Communications Engineering
School of Science and Engineering, Ateneo de Manila University
Quezon City, 1108, Philippines
jason.lorenzo@obf.ateneo.edu

***Abstract*—This paper details the construction and functional emulation of a 4-bit shift register, aimed at replicating the behavior of a 74194 IC as depicted. The Verilog HDL code provided defines the operation of the shift register, with the output Q being determined by either a clear operation or the contents of the register nextQ. The sequential block, governed by positive clock edges and control signals, orchestrates shift operations and data loading, while the combinational block processes these inputs to generate the output Q. Through simulation, the functionality of the shift register is validated, showcasing operations such as left and right shifts, data loading, and clearing, in line with the expected behavior.**

***Index Terms*—Verilog HDL, shift register**

I. Introduction

This activity involves the construction of a 4-bit shift register, specifically the task is to imitate the function of a 74194 IC as outlined in Fig. 1.



	MODE			INPUTS	OUT- PUTS			
CLR'	**S1 S0**	**CLK**	**SL SR**	**A B C D**	**QA**	**QB**	**QC**	**QD**
0	X X	X	X X	X X X X	0	0	0	0
1	X X	0	X X	X X X X	QA0	QB0	QC0	QD0
1	1 1	POS	X X	a b c d	a	b	c	d
1	0 1	POS	X 1	X X X X	1	Qan	Qbn	QCn
1	0 1	POS	X 0	X X X X	0	Qan	Qbn	QCn
1	1 0	POS	1 X	X X X X	Qbn	Qcn	Qdn	1
1	1 0	POS	0 X	X X X X	QBn	QCn	Qdn	0
1	0 0	X	X X	X X X X	QA0	QB0	QC0	QD0

LEGEND:
0 = logic low
1 = logic high
X = don't care
POS = positive edge of clock, low to high transition.
QA0, QB0, QC0, QD0 means previous values of Q i.e. retain old value

Fig. 1. 74194 Simplified State Table

The 74194, or 74LS194, is a 4-bit bidirectional universal shift register. Its output (4-bit Q) changes based on its inputs: CLR' (clear), CLK (clock), S1 and S0 (shift control inputs), and SL and SR (serial data inputs for left and right shift, respectively). When CLR' is low, the register is cleared, setting all bits of Q to 0. On each rising edge of CLK, the register shifts its contents based on the values of S1 and S0. If S1 and S0 are both low, Q retains its value, if S0 is low and S1 is high, it shifts left (SL), if S0 is high and S1 is low, it shifts right, and if both are high, it loads the 4-bit input ABCD (LDINPUT). During shifting, the input from SL or SR is loaded into the first or last bit, respectively, depending on the shift direction. The remaining bits shift accordingly, with the last bit shifted out being captured in the output Q. This process repeats on each clock pulse, enabling the register to effectively shift its contents bidirectionally.

II. HDL Code

Listing 1 shows the HDL code implementation of the function defined above.

Listing 1. 74XX138 Decoder HDL Code

```
module shift_reg(CLK, CLRn, S1,
   S0, SL, SR, LDINPUT, Q);
input CLK, CLRn, S1, S0, SL, SR;
input [3:0] LDINPUT;
output [3:0] Q;

wire [3:0] Q;
reg [3:0] nextQ;

assign Q = (CLRn) ? (nextQ) : (4'b0000);

always @(posedge CLK)
  if (!CLRn) nextQ <= 4'b0000;
  else
    if (S1)
      if (S0) nextQ <= LDINPUT;
      else nextQ <= {Q[2:0], SL};
    else
      if(S0) nextQ <= {SR, Q[3:1]};

endmodule
```

The inputs CLK, CLRn, S1, S0, SL, SR, and LDINPUT represent the clock signal, clear signal (active low), shift control inputs, and serial data inputs for left and right shift, respectively. The output Q represents the 4-bit output of the shift register. Inside the module, there are two main components: a combinational block and a sequential block.

In the combinational block, the assign statement sets the output Q to either the internal register nextQ or 4'b0000 based on the CLRn input. When CLRn is low, indicating a clear operation, Q is set to 4'b0000, effectively clearing the register.

In the sequential block, the always block is sensitive to the positive edge of the clock (posedge CLK). When CLRn is high, indicating that the clear operation is not active, the if statements within the always block determine the behavior of the shift register based on the values of S1 and S0. If both S1 and S0 are high, the register loads the data from LDINPUT into nextQ. If S1 is high and S0 is low, indicating a shift left operation, the register shifts its contents to the left, with SL being loaded into the rightmost bit. If S1 is low and S0 is high, indicating a shift right operation, the register shifts its contents to the right, with SR being loaded into the leftmost bit, and the remaining bits shifted to the right. If both are low, the sequential block just passes, effectively retaining nextQ's value.

III. Output Circuit

Fig. 2 shows the output circuit of the code above.

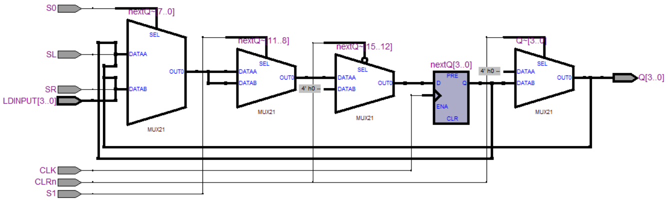


Fig. 2. 74194 Shift Register Circuit Implementation RTL Netlist

The diagram illustrates the functional architecture of the shift register module. The rightmost section depicts the combinational block responsible for determining the output Q. This output is contingent upon the state of the clear signal (CLR'), which dictates whether Q adopts the value 0000 or the contents of the register nextQ. In contrast, the generation of nextQ is managed by the sequential block. This block comprises a series of multiplexers, nested in order of significance, controlled by signals S0, S1, and CLR'.

IV. Simulation

Fig. 3 shows respective output waveforms of the device given test inputs.

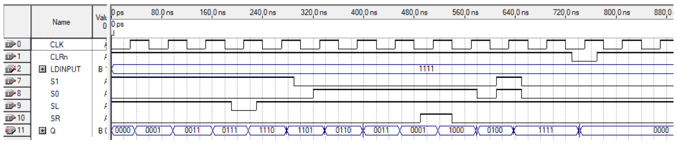


Fig. 3. 74194 Shift Register Implementation Simulation Waveforms

In this test simulation, the shift register undergoes various operations to validate its functionality. Initially, a left shift operation is performed, with the serial input set to 1 ($SL = 1$) for the first three positive clock pulses, 0 for the fourth, then back to 1 for the fifth, resulting in the sequence 1101. Subsequently, the shift direction is reversed, shifting the register right by setting the serial input to 0 ($SR = 0$) for three clock pulses, followed by a transition to 1 ($SR = 1$) on the fourth pulse, then back to 0 for the last, yielding the sequence 0100. Following this, the register is loaded with the value 1111 before being cleared to 0000. This observed behavior is consistent with the anticipated functionalities outlined in Fig. 1.