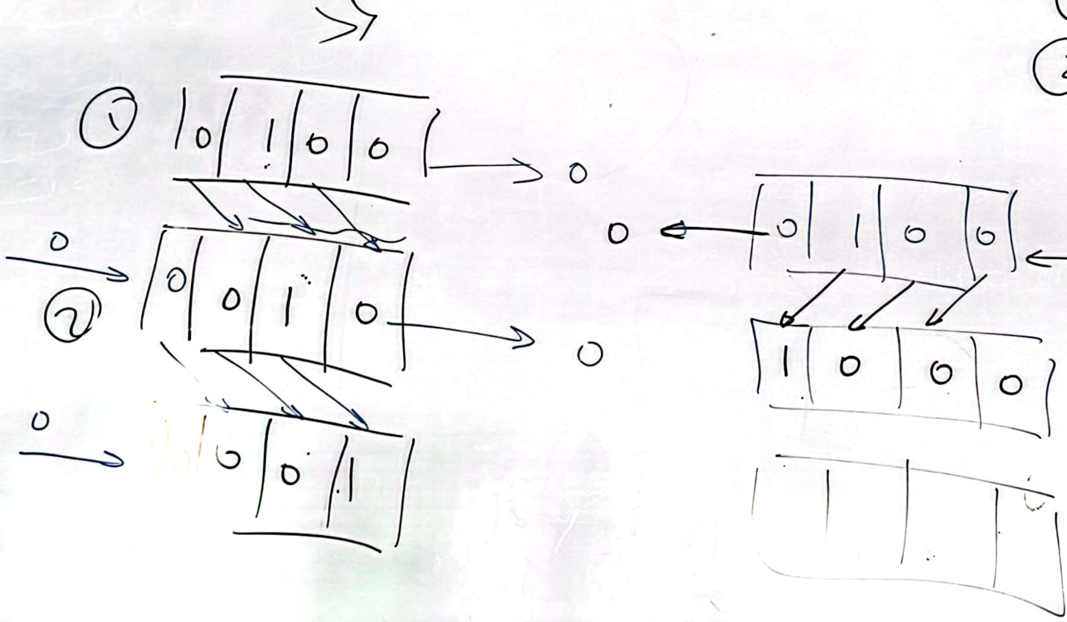
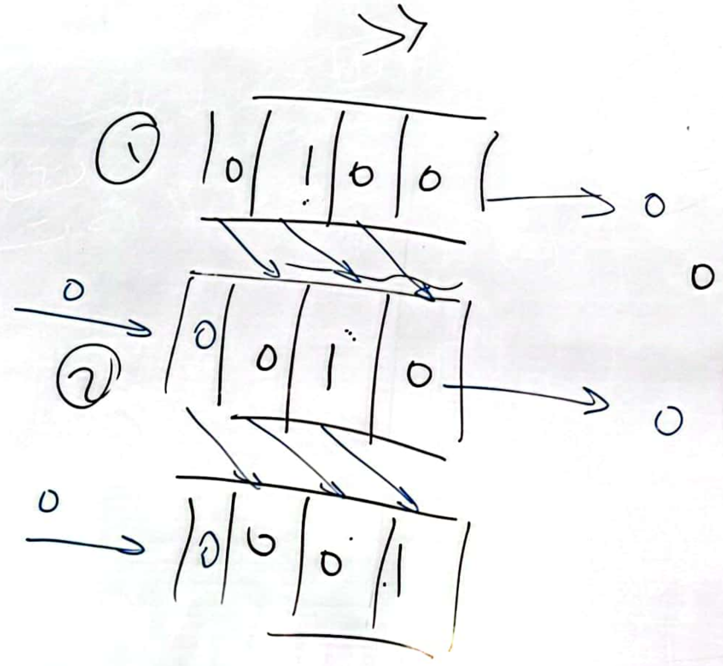
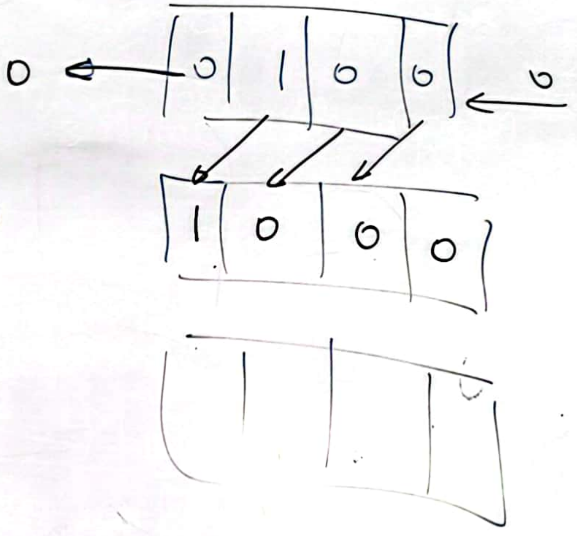
Shift register

1. load data
2. >> or <<



Shift register



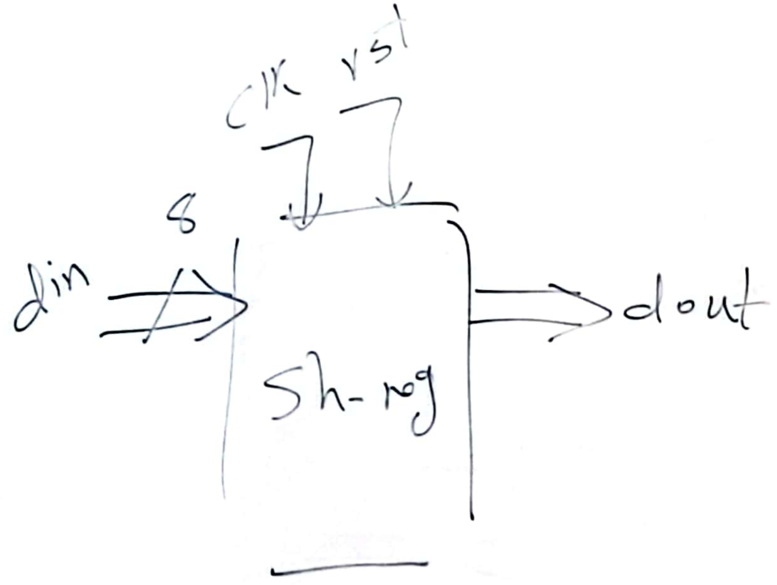


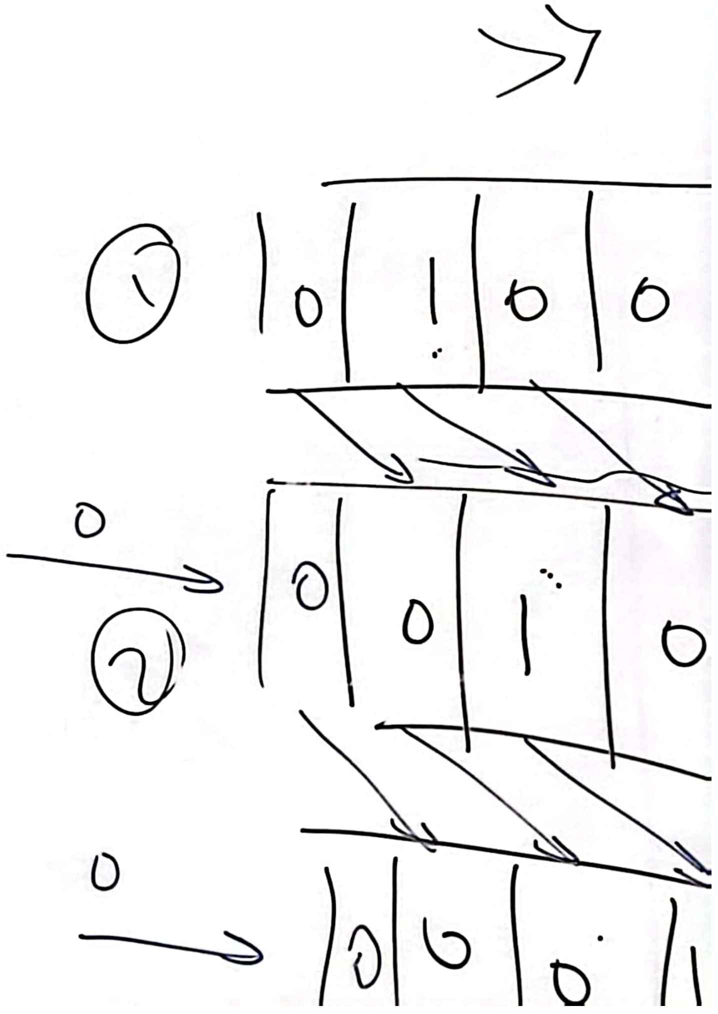
① load data

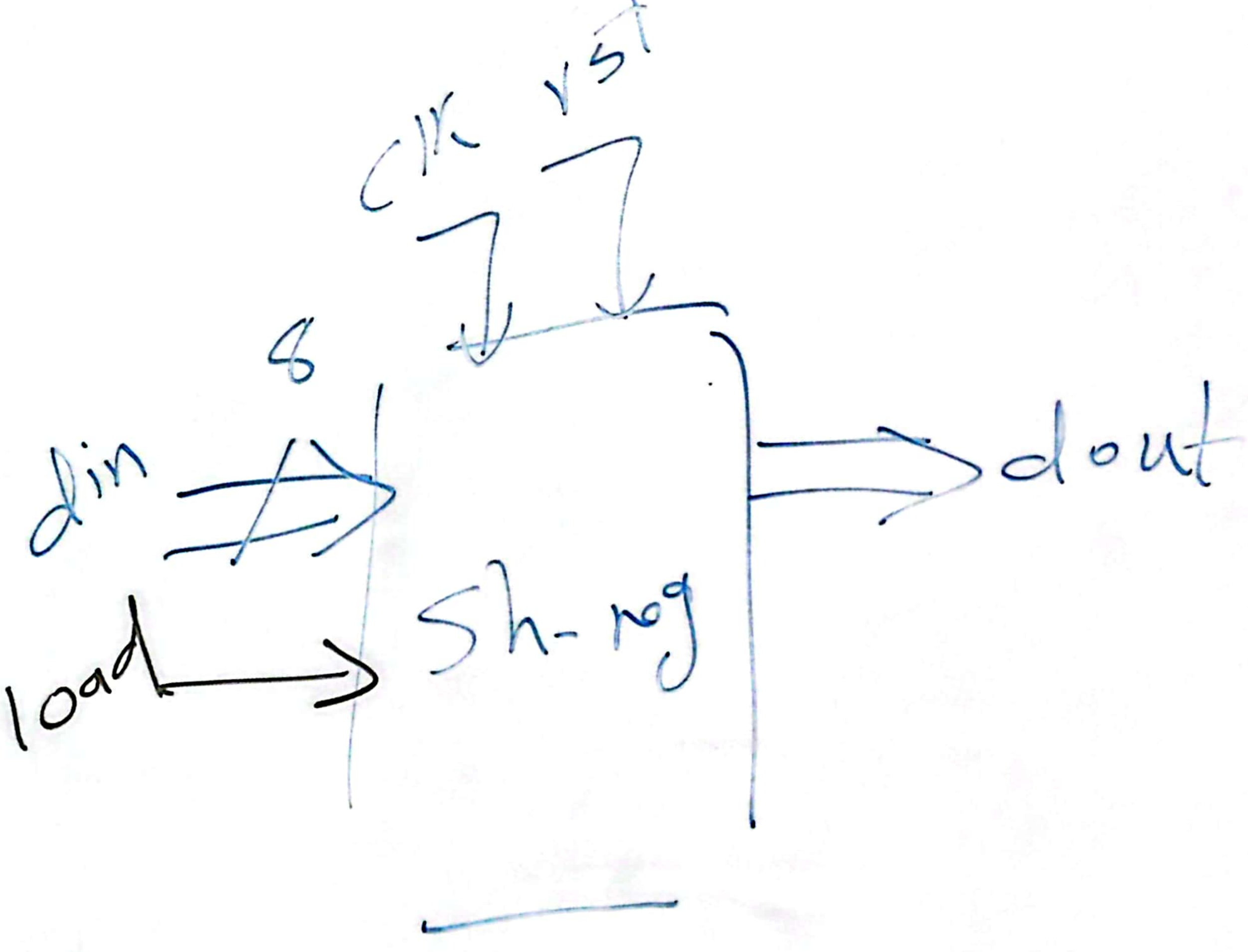
② >> or <<

```
reg_temp <= '0' & reg_temp(3 down 1)      >>
reg_temp <= reg_temp(2 downto 0) & '0'      <<
```

Shift regis

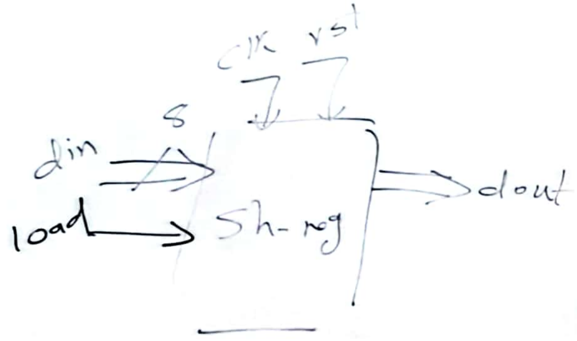


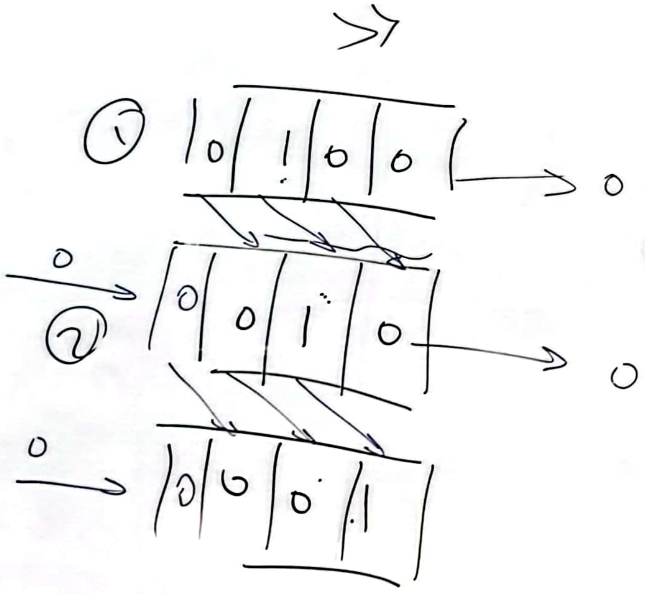



clk
rst
8
din
load
sh-reg
dout

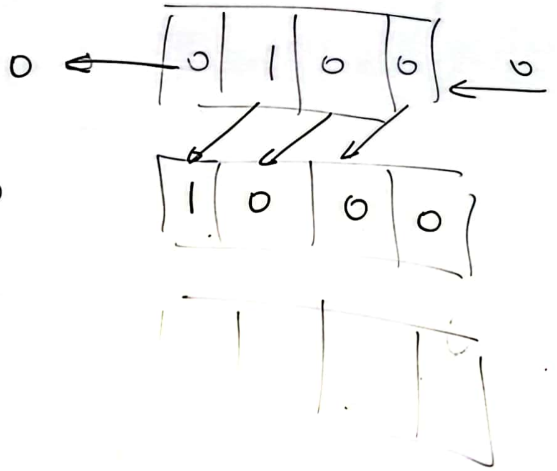
1
0
2

Shift register



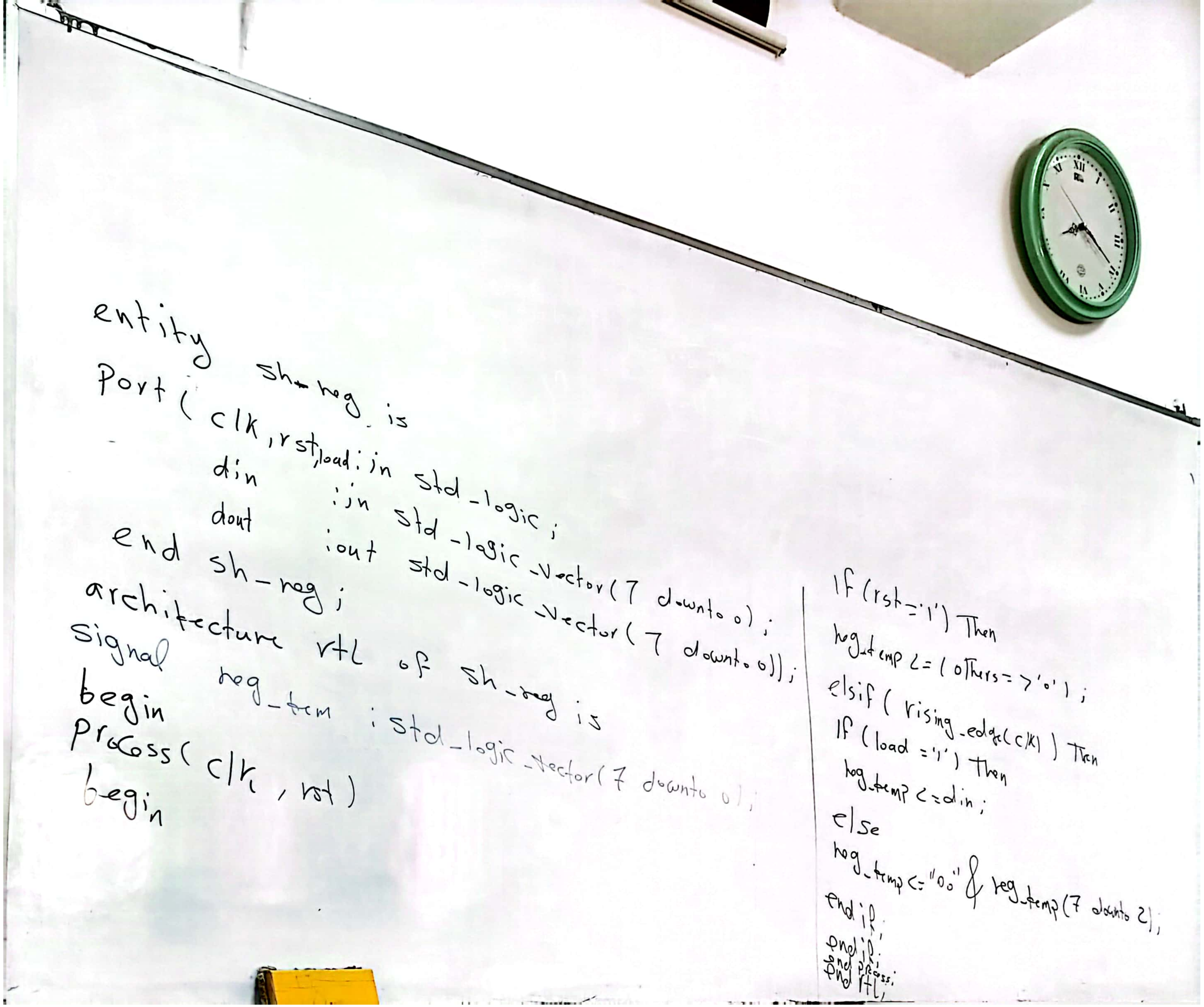


load —'1'→ ① load data
load —'0'→ ② >> or <<



```
entity sh_reg is
port ( clk, rst, load : in std_logic ;
        din       :in  std_logic_vector(7 downto 0);
        dout      :out std_logic_vector(7 downto 0));
 end sh_reg;
architecture rtl of sh_reg is
signal  reg_tem   : std_logic_vector(7 downto 0);
begin
process( clk , rst )
 begin
```

```
If (rst='1') Then
reg_temp <= ( others => '0' ) ;
elsif ( rising_edge(clk) ) Then
If ( load ='1' ) Then
 reg_temp <= din ;
else
reg_temp <= "00" & reg_temp(7 downto 2);
end if;
end if;
end process;
end rtl;
```



```
entity sh_reg is
Port ( clk, rst, load : in std_logic;
       din       : in std_logic_vector(7 downto 0);
       dout      : out std_logic_vector(7 downto 0));
 end sh_reg;
architecture rtl of sh_reg is
signal   reg_tem   : std_logic_vector(7 downto 0);
begin
Process( clk , rst )
begin
```

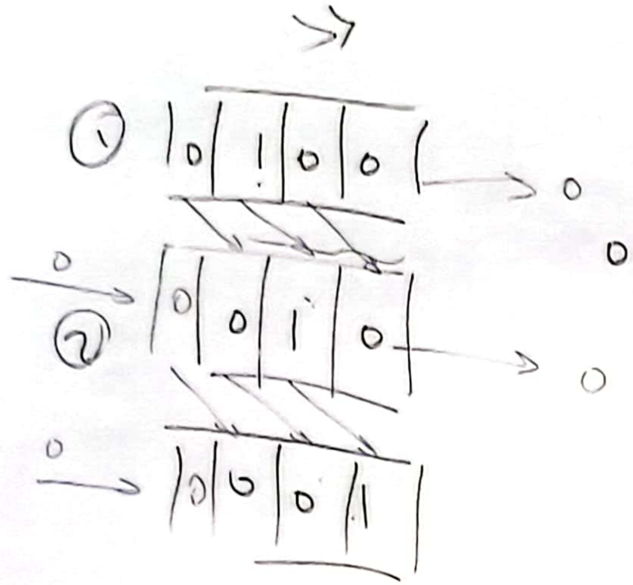
```
If (rst = '1') Then
reg_temp <= (others => '0');
elsif ( rising_edge(clk) ) Then
If ( load = '1' ) Then
 reg_temp <= din;
else
reg_temp <= "00" & reg_temp(7 downto 2);
end if;
end if;
end process;
end rtl;
```

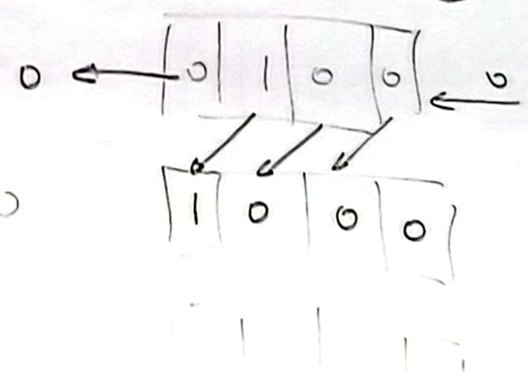
```
entity sh_reg is
port ( clk, rst, load : in std_logic ;
        din    : in  std_logic_vector( 7 downto 0) ;
        dout   : out std_logic_vector( 7 downto 0)) ;
  end sh_reg ;
architecture rtl of sh_reg is
signal  reg_tem   : std_logic_vector( 7 downto 0) ;
begin
process( clk , rst )
begin
```

```
If (rst = '1') Then
reg_temp <= ( others => '0' ) ;
elsif ( rising_edge( clk ) ) Then
If ( load = '1' ) Then
 reg_temp <= din ;
else
reg_temp <= "00" & reg_temp(7 downto 2) ;
end if ;
end if ;
end process ;
end rtl ;
```

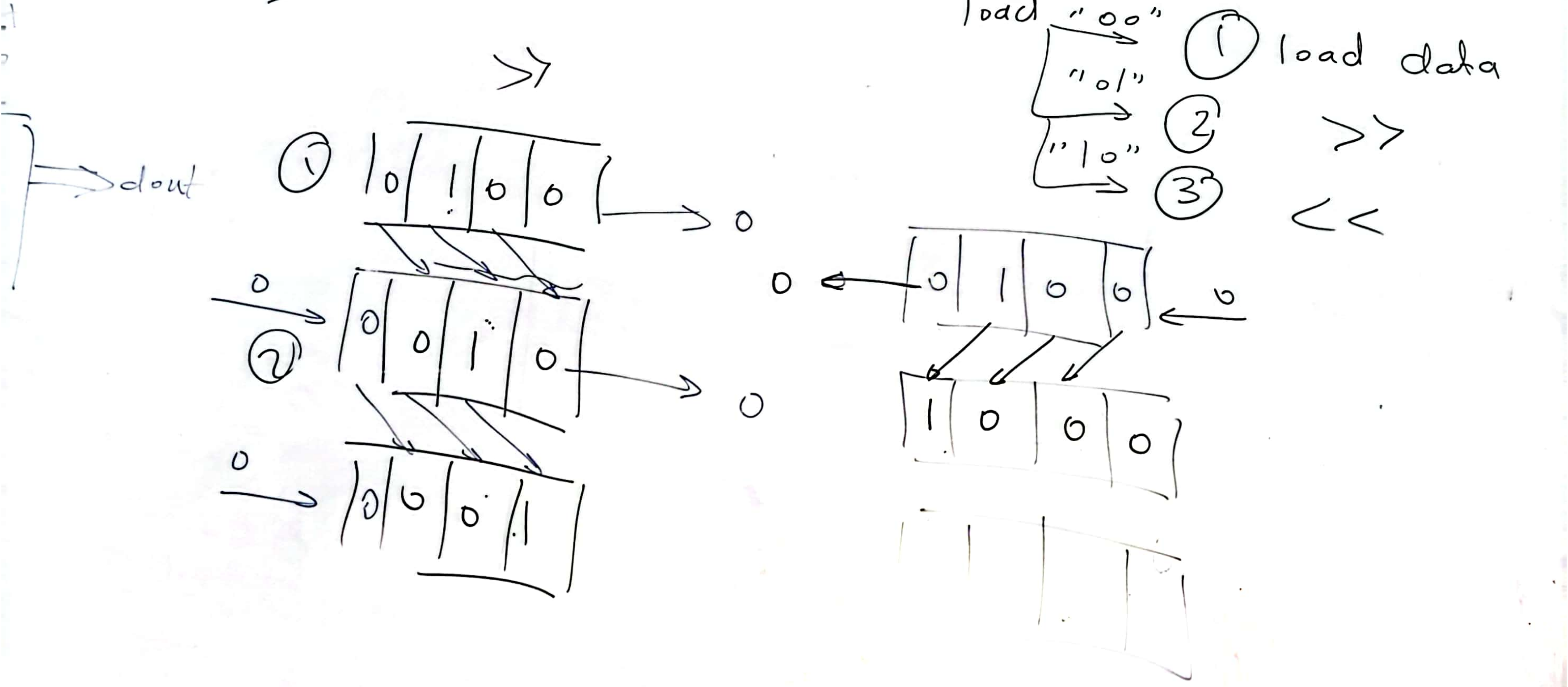
Shift register

load "00" → ① load data
"01" → ② >>
"10" → ③ <<





Shift register



sh_r
sh_L

load <= sh_r & sh_L;

'0' '1'

If // sh_r = '0' AND sh_L = '

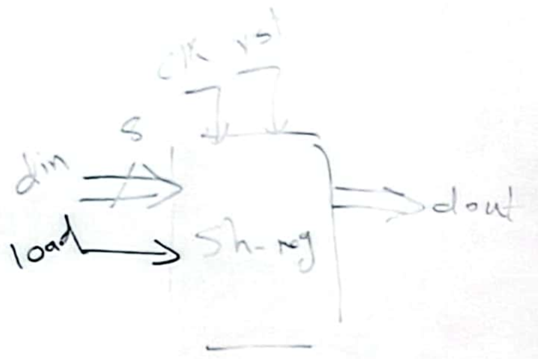
```
entity sh_reg is
port ( clk, rst   : in std_logic;
       din        : in std_logic_vector(7 downto 0);
       load       : in std_logic_vector(1 downto 0);
       dout       : out std_logic_vector(7 downto 0));
end sh_reg;
architecture rtl of sh_reg is
signal reg_tem : std_logic_vector(7 downto 0)
begin
Process(clk, rst)
begin
```

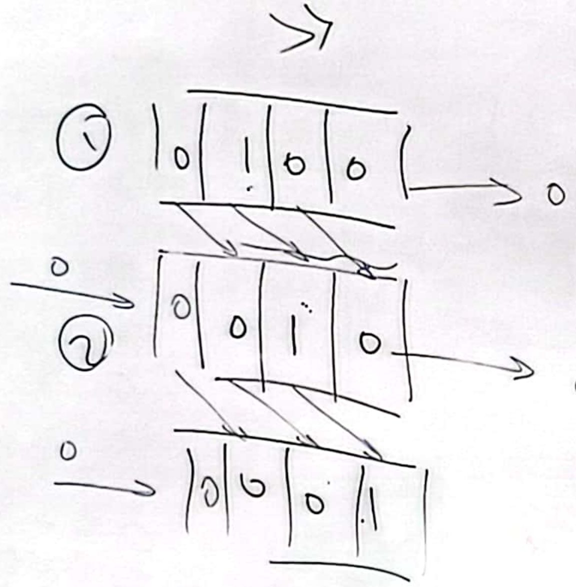
```
If (rst = '1') Then
reg_temp <= (others => '0');
elsif ( rising_edge(clk) ) Then
If (load = "00") then
reg_temp <= din;
elsif (load = "10") Then
reg_temp <= '0' & reg_temp(7 downto 1);
elsiF (load = "01") Then
reg_temp <= reg(6 downto 0) & '0';
end if;
```

```
end if;
end Process
dout <= reg_te
end rtl;
```

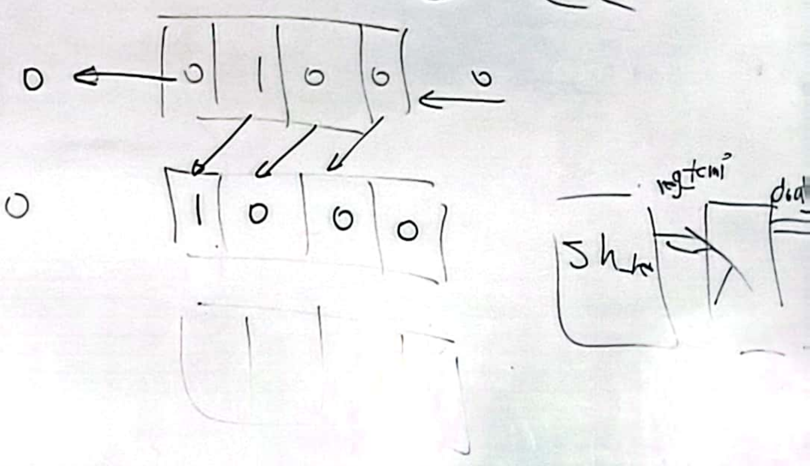
```
Sh-r
Sh-L

load <= Sh-r & Sh-L ;

If ( Sh-r = '0' AND Sh-L =

entity sh_reg is
port ( clk,rst   :in std_logic ;
       din       :in std_logic_vector(7 downto 0) ;
       load      :in std_logic_vector(1 downto 0)
       dout      :out std_logic_vector(7 downto 0)) ;
 end sh_reg ;
architecture rtl of sh_reg is
signal reg_tem : std_logic_vector(7 downto 0)
begin
process( clk , rst )
 begin

If (rst = '1') Then
reg_temp <= ( others => '0' ) ;
elsif ( rising_edge(clk) ) Then
If ( load = "00") Then
 reg_temp <= din ;
elsif( load = "10" ) Then
reg_temp <= '0' & reg_temp( 7 downto 1 ) ;
elsif ( load = "01" ) Then
reg_temp <= reg( 6 downto 0 ) & '0' ;
end if

End if ;
End process
dout <= reg_
End rtL ;
```

Shift register





load "00" → 1 load data
"01" → 2 >>
"10" → 3 <<



Counter

↓

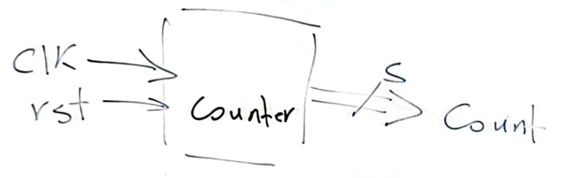
timer

↳ Clock → freq → 1 Hz   2 Hz

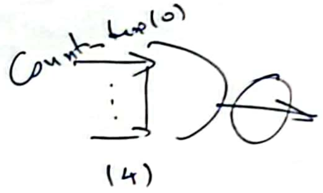
  ↳ T → 1 sec   0.5 sec

$$\frac{30\ \text{sec}}{0.5\ 1\ \text{sec}} = \overset{60}{30}\ \underline{\text{clock cycle}}$$

Count_temp <= step + Count_temp

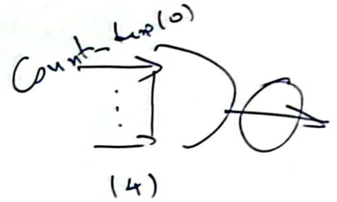


```
0 0 0 0 0
0 0 0 0 1
0 0 0 1 0
. . . 0 1 1
      1
      1
      1
      :
1 1 1 1 1
```



```
entity counter is
Port( clk, rst      : in  std_logic ;
      end_t         : out std_logic ;
      Count         : out std_logic_Vector ( 4 downto 0));
end counter ;
architecture rtL of counter is
Signal count_temp : unsigned (4 downto 0);
begin
Process ( clk, rst )
If( rst ='1') Then
Count_temp <= (others=>'0'
```

```
elsif (rising_edge ( clk)) then
Count_temp <= Count_temp + 1 ;
end if;
end process;
Count <= Std_logic_Vector(Count_temp);
end_t <= '1' When Count_temp = 31 else '0';
```



```
entity Counter is
  Port( clk, rst  : in  std_logic ;
        end_t     : out std_logic ;
        Count     : out Std_logic_Vector ( 4  downto 0));
 end counter ;
architecture rtL of counter is
 Signal  count_temp  : unsigned (4  downto 0);
 begin
Process ( clk, rst )
If ( rst = '1' ) Then
Count_temp <= (others => '0'

elsif ( rising_edge ( clk ) ) Then
  Count_temp <= Count_temp + 1 ;

end if;
end process;
Count <= Std_logic_Vector( Count_temp ) ;
end_t <= '1' When Count_temp = 31  else '0' ;
```

Counter

timer

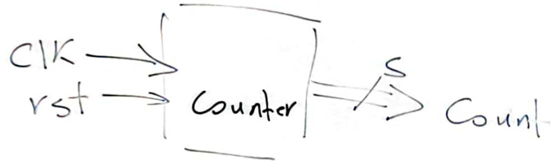
$$\begin{tikzcd}
\text{Clock} \arrow[r] & \text{freq} \arrow[r] \arrow[d] & 1\,\text{Hz} \quad 2\,\text{Hz} \\
& T \arrow[r] & 1\,\text{sec} \quad 0.5\,\text{sec}
\end{tikzcd}$$

$\frac{30\,\text{sec}}{0.5, 1\,\text{sec}} = \overset{60}{30}$ clock cycle

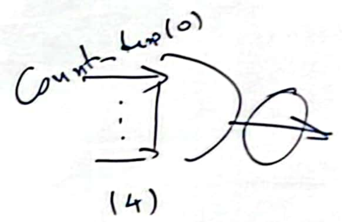
250

$0 \longrightarrow 29$

Count_temp <= step + Count_tem



```
0 0 0 0 0
0 0 0 0 1
0 0 0 1 0
. .   0 1 1
      .
      .
      .
1 1 1 1 1
```



```
entity Counter is
Port( clk, rst   : in std_logic ;
      end_t      : out std_logic ;
      Count      : out std_logic_Vector ( 4  downto 0));
end Counter ;
architecture rtL of Counter is
Signal Count_temp : unsigned (4 downto 0) ;
begin
Process ( clk , rst )
If( rst = '1' ) Then
Count_temp <= (others => '0' ) ;
```

```
→ elsif ( rising_edge ( clk ) ) Then
① Count_temp <= Count_temp + 1 ;
② If ( Count_temp = 29 ) Then
   Count_temp <= ( others => '0' ) ;
 end if ;
end if ;
end Process ;
Count <= Std_logic_Vector( Count_temp ) ;
end_t <= '1' When Count_temp = 31 else '0' ;
```

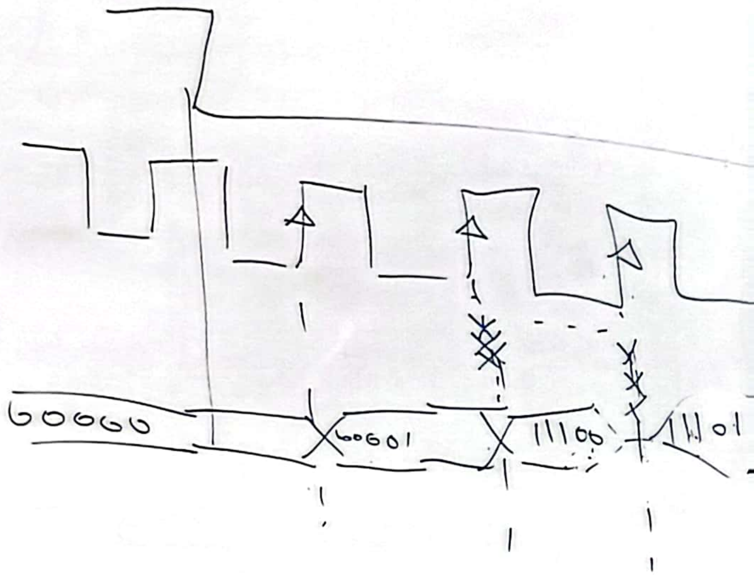
Counter

timer

Clock → ?

$0 \longrightarrow \overset{30}{29}$

$$\frac{30\text{ sec}}{0.5.1\text{ Sec}} = \overset{60}{3}$$


00000
00001
11100
11101