

Homework # 3 Solutions

1). 2-to-1 multiplexer:-

Code:-

```
entity mux2to1 is
port(
  inp1 : in bit;
  inp2 : in bit;
  sel  : in bit;
  outp : out bit);
end mux2to1;

architecture mux of mux2to1 is

begin

outp <= inp1 after 1 ns when sel = '0' else
  inp2 after 1 ns;

end;
```

Simulation results in LIST format:-

ps	delta	/mux2to1/inp1	/mux2to1/inp2	/mux2to1/sel	/mux2to1/outp	
0	+0	1	0	0	0	
1000	+0	1	0	0	1	---- outp gets inp1 when sel='0'
10000	+0	1	0	1	1	
11000	+0	1	0	1	0	---- outp gets inp2 when sel='1'

2). D flip-flop:-**Code:-**

```
entity dff is
port(
  d : in bit;
  clk : in bit;
  clr : in bit;
  q : out bit);
end dff;
```

```
architecture dff_arch of dff is
```

```
begin
process(clr,clk)
begin
```

```
if(clr = '0') then
  q <= '0' after 1 ns;
elsif(clk'event and clk = '1') then
  q <= d after 1 ns;
end if;
```

```
end process;
end;
```

Simulation results in LIST format:-

ps	delta	/dff/d	/dff/clk	/dff/clr	/dff/q	
0	+0	1	0	1	0	
5000	+0	1	1	1	0	
6000	+0	1	1	1	1	--- q gets d
10000	+0	1	0	0	1	
11000	+0	1	0	0	0	--- flip-flop is cleared
15000	+0	1	1	1	0	
16000	+0	1	1	1	1	--- q gets d
20000	+0	0	0	1	1	
25000	+0	0	1	1	1	
26000	+0	0	1	1	0	--- q gets d

3). 4-bit binary shift-register:-**Code:-**

```
entity shft_reg is
port(
  Pin      : in bit_vector(3 downto 0);
  Sin      : in bit;
  clr,clk,load : in bit;
  Pout     : out bit_vector(3 downto 0);
  Sout     : out bit);
end shft_reg;
```

architecture sr of shft_reg is

```
component dff
port(
  d : in bit;
  clk : in bit;
  clr : in bit;
  q : out bit);
end component;
```

```
component mux2to1
port(
  inp1 : in bit;
  inp2 : in bit;
  sel : in bit;
  outp : out bit);
end component;
```

```
signal int1, int2 : bit_vector(3 downto 0);
```

```
begin
```

```
M3 : mux2to1 port map(Sin,Pin(3),load,int1(3));
M2 : mux2to1 port map(int2(3),Pin(2),load,int1(2));
M1 : mux2to1 port map(int2(2),Pin(1),load,int1(1));
M0 : mux2to1 port map(int2(1),Pin(0),load,int1(0));
D3 : dff port map(int1(3),clk,clr,int2(3));
D2 : dff port map(int1(2),clk,clr,int2(2));
D1 : dff port map(int1(1),clk,clr,int2(1));
D0 : dff port map(int1(0),clk,clr,int2(0));
Pout <= int2;
Sout <= int2(0);
end;
```

Simulation results in LIST format:-

ps	delta	pin	sin	clr	clk	load	pout	sout
0	+0	0000	1	1	0	0	0000	0
10000	+0	0000	1	1	1	0	0000	0
11000	+1	0000	1	1	1	0	1000	0
20000	+0	0000	1	1	0	0	1000	0
30000	+0	0000	1	1	1	0	1000	0
31000	+1	0000	1	1	1	0	1100	0
40000	+0	0000	1	1	0	0	1100	0
50000	+0	0000	1	1	1	0	1100	0
51000	+1	0000	1	1	1	0	1110	0
60000	+0	0000	1	1	0	0	1110	0
70000	+0	0000	1	1	1	0	1110	0
71000	+1	0000	1	1	1	0	1111	1
80000	+0	0000	1	1	0	0	1111	1
90000	+0	0000	1	1	1	0	1111	1
100000	+0	0000	0	1	0	0	1111	1
110000	+0	0000	0	1	1	0	1111	1
111000	+1	0000	0	1	1	0	0111	1
120000	+0	1010	0	1	0	1	0111	1
130000	+0	1010	0	1	1	1	0111	1
131000	+1	1010	0	1	1	1	1010	0
140000	+0	1010	0	0	0	1	1010	0
141000	+1	1010	0	0	0	1	0000	0

First four steps shift a '1' into the shift register serially. Next step shifts in a '0' serially. Next step loads a "1010" pattern through the parallel load. Last step clears all the flip-flops.