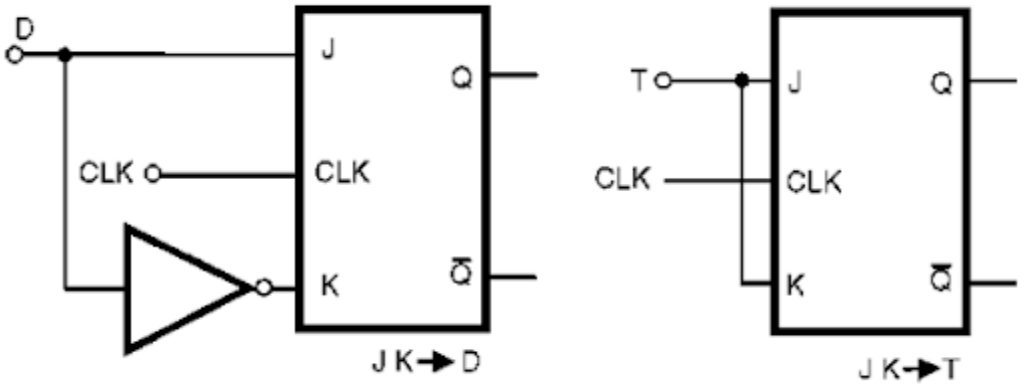
Practica 11

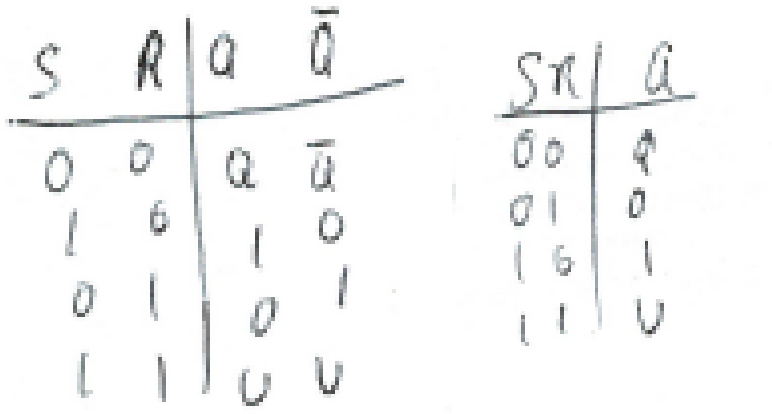
Flip-flops

Profesor: Barrón Vera José Emanuel
Materia: Fundamentos de diseño digital
Grupo: 3CV6
Alumno: Cazares Cruz Jeremy Sajid
Boleta: 2021630179

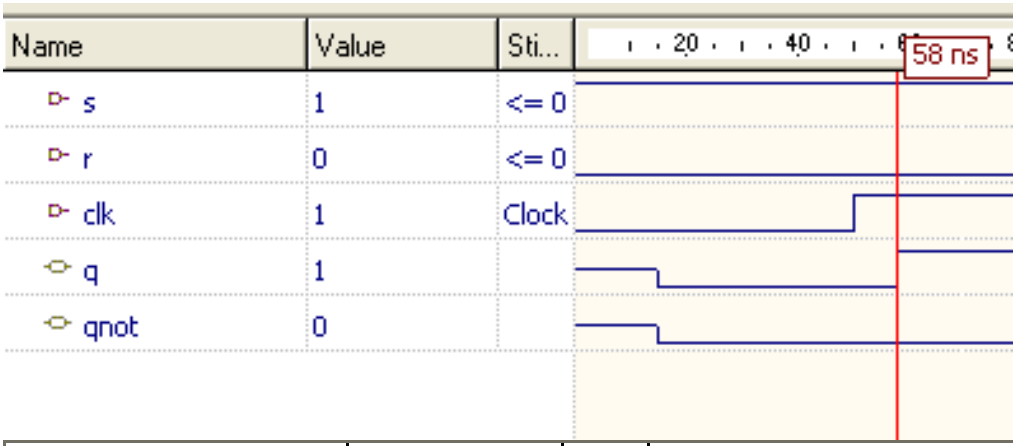


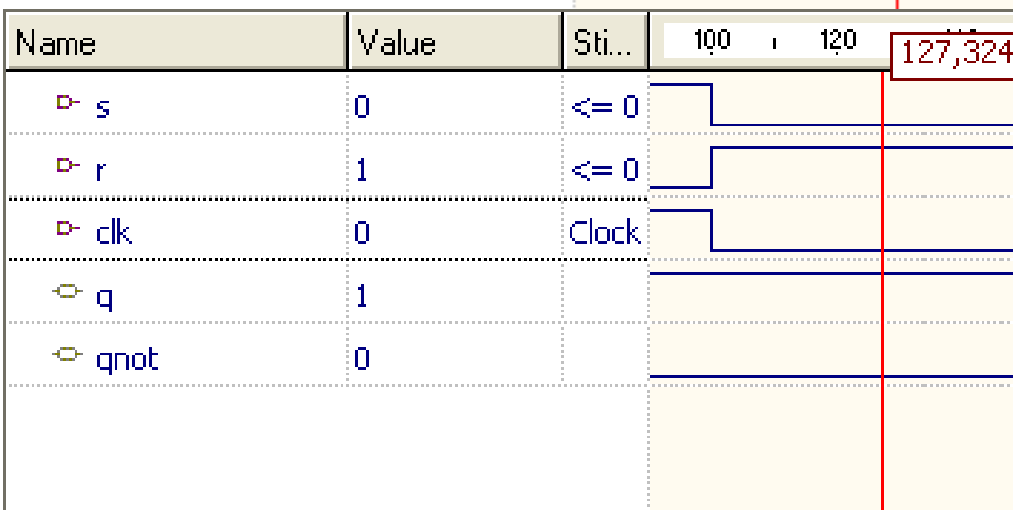
Flip-flop SR

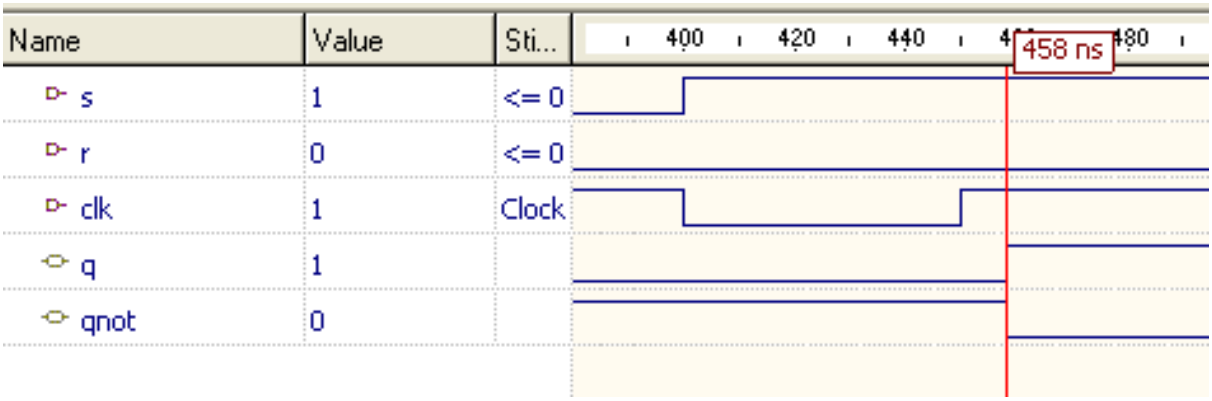
Para el Flip-flop sr se tiene tanto la utilización de las compuertas retroalimentadas que son implementadas mediante el uso de “signals” para después mediante “ifs” realizar la tabla de verdad de la misma, en caso de no existir ningún caso dentro de los “ifs” se pone como estado prohibido en esta caso se hace uso de “-“mediante la librería “std_ulogic”



```
12  signal t1,t2 : std_logic;
13   begin
14    t1 <= r nor t2;
15    t2 <= s nor t1;
16      process (clk,r,s)
17      begin
18       if(clk'event and clk='1' ) then
19             if(r='0' and s='0') then
20              Q <=t1;
21              Qnot <= t2;
22             elsif(r='0' and s='1') then
23              Q <='1';
24              Qnot <='0';
25             elsif(r='1' and s='0') then
26              Q <='0';
27              Qnot <='1';
28             elsif(r='1' and s='1') then
29              Q <='-';
30              Qnot <='-';
31             end if;
32           end if;
33      end process;
```

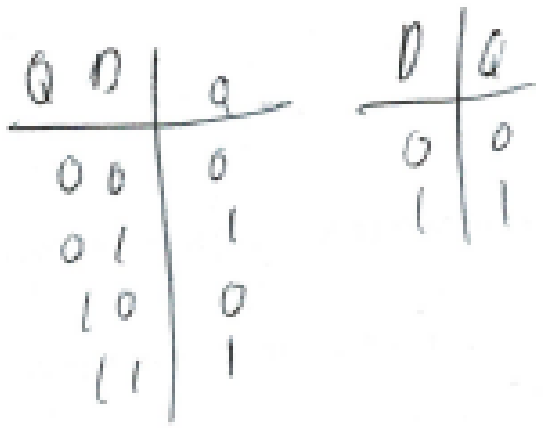

Name
Value
Sti...
20
40
58 ns
s
1
<= 0
r
0
<= 0
clk
1
Clock
q
1
qnot
0


Name
Value
Sti...
100
120
127,324
s
0
<= 0
r
1
<= 0
clk
0
Clock
q
1
qnot
0


Name
Value
Sti...
400
420
440
458 ns
s
1
<= 0
r
0
<= 0
clk
1
Clock
q
1
qnot
0

Flip-flop D

Para el Flip-flop tipo D se utiliza la forma sencilla sin retroalimentar el circuito tal como se puede ver en la segunda tabla de verdad



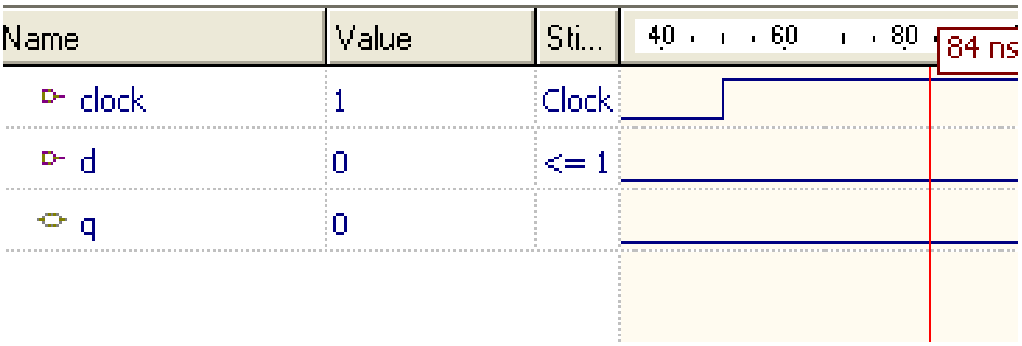
Q	D	q
0	0	0
0	1	1
1	0	0
1	1	1

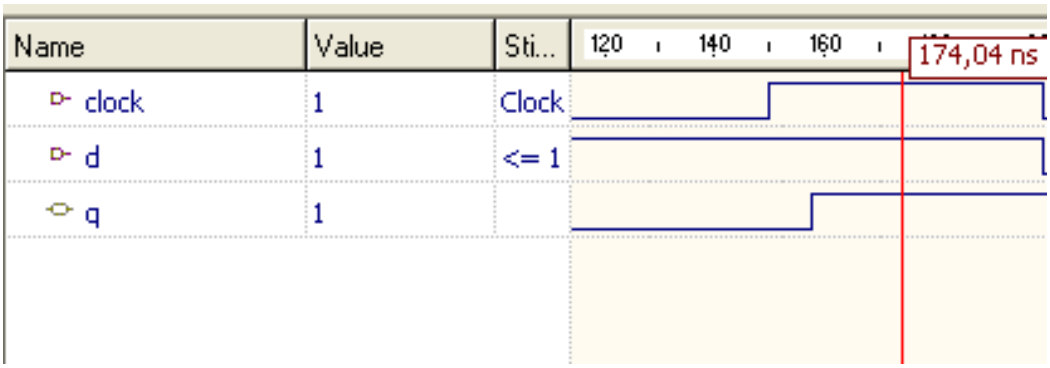
D	Q
0	0
1	1

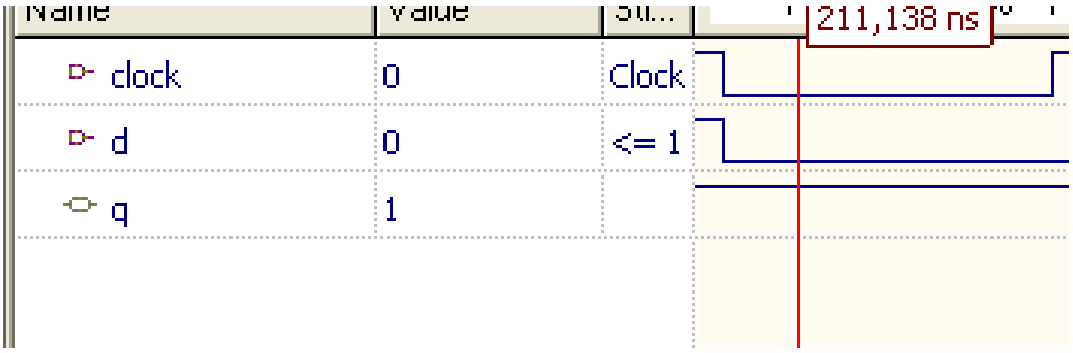
El código es el siguiente:

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.all;
3
4  entity flipflop is
5  port(
6      q:out std_logic;
7      clk,rst,D:in std_logic
8  );
9  end flipflop;
10
11 architecture codigo of flipflop is
12     begin
13     process (clk,rst)
14     begin
15         if rst='1' then
16         q <= '0';
17         elsif clk'event and clk='1' then
18         q <= d;
19         end if;
20     end process;
21 end codigo;
```

Las simulaciones se verán a continuación;



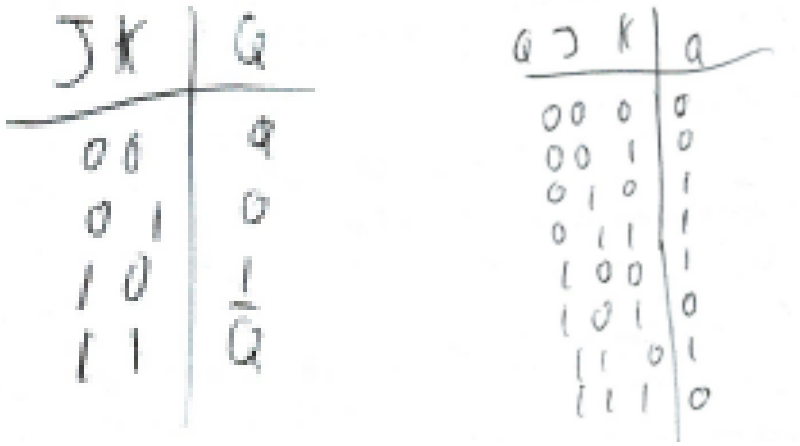




En la ultima imagen se puede ver un dato “guardado” debido a que el cambio no se ha hecho hasta que el reloj pase de estar de “0” a “1”

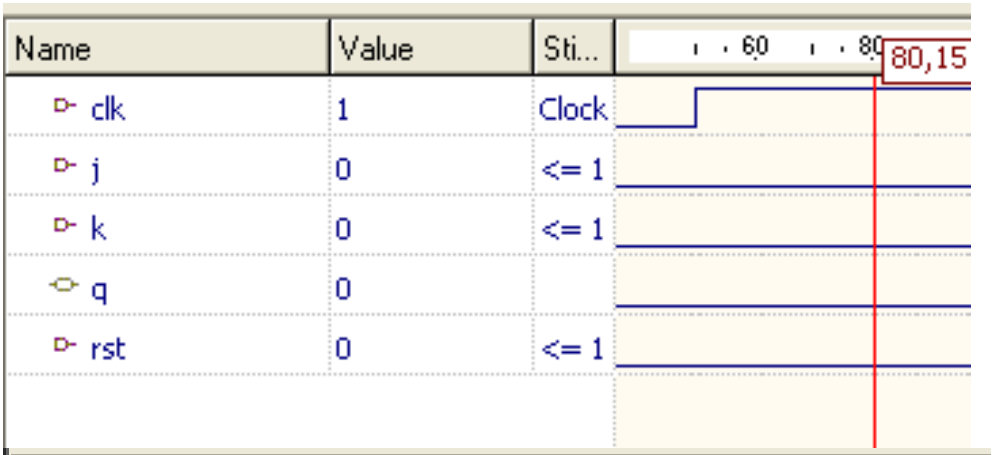
Flip-flop JK

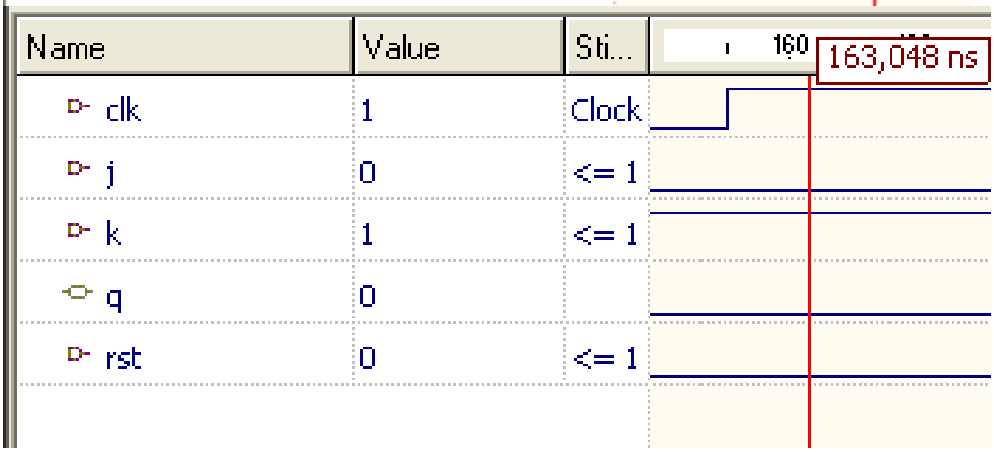
Para la implementación del Flip-flop jk se hace uso de entradas retroalimentadas de tal manera que se debe hacer uso de “signal” para poder realizar esta conexión lógica y de igual manera de una “q auxiliar” y de igual manera el estado lógico de la salida solo cambiara con el flanco de subida del reloj
Finalmente, para cumplir la tabla de verdad es necesario usar un ”when-case”

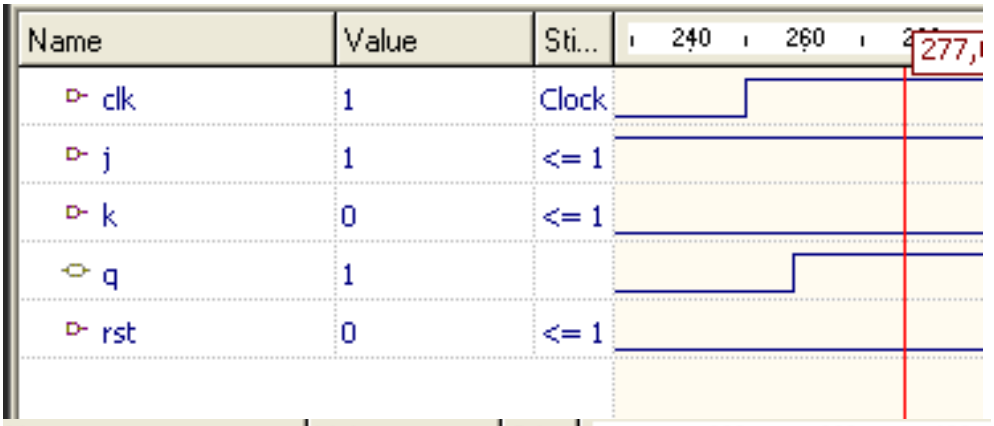


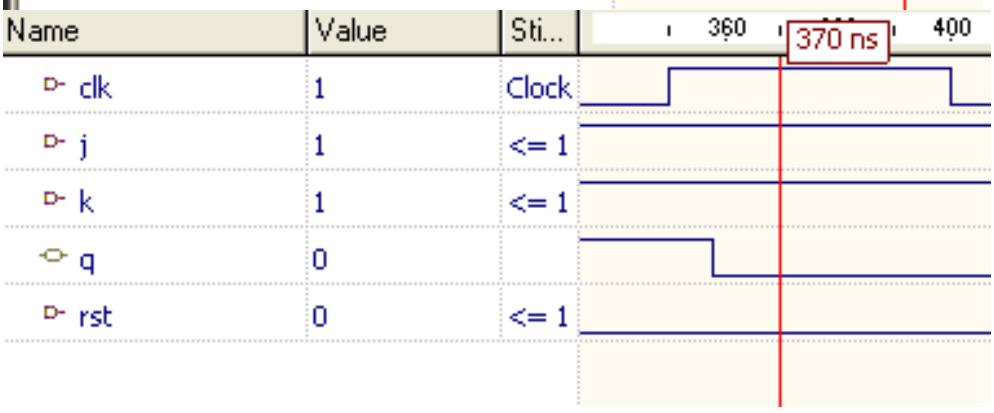
El código es el siguiente:

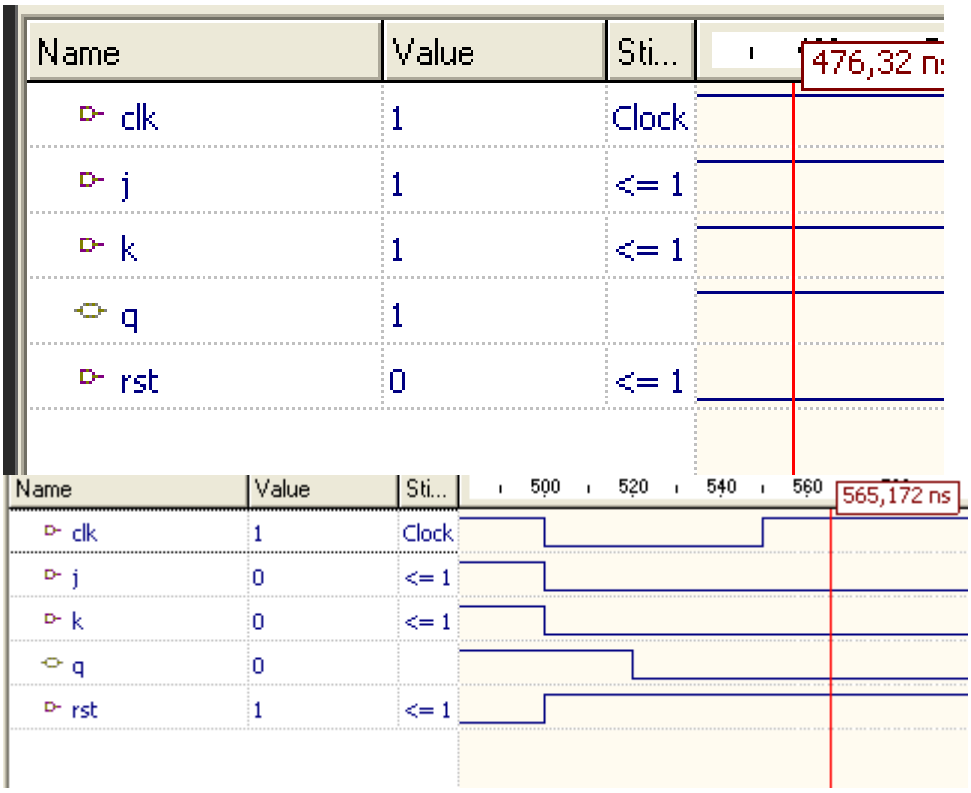
```
10 architecture codigo of FFJK is
11
12     signal JK: std_logic_vector(1 downto 0);
13     signal Q_aux: std_logic;
14     begin
15     JK <= J&K;
16     process(clk, rst)
17     begin
18         if rst = '1' then
19         Q_aux <= '0';
20         elsif rising_edge(clk) then
21             case JK is
22             when "00" => Q_aux <= Q_aux;
23             when "01" => Q_aux <= '0';
24             when "10" => Q_aux <= '1';
25             when others => Q_aux <= not Q_aux;
26             end case;
27         end if;
28     end process;
29     Q <= Q_aux;
30
31 end codigo;
```


Name
Value
Sti...
60
80
80,15
clk
1
Clock
j
0
<= 1
k
0
<= 1
q
0
rst
0
<= 1


Name
Value
Sti...
160
163,048 ns
clk
1
Clock
j
0
<= 1
k
1
<= 1
q
0
rst
0
<= 1

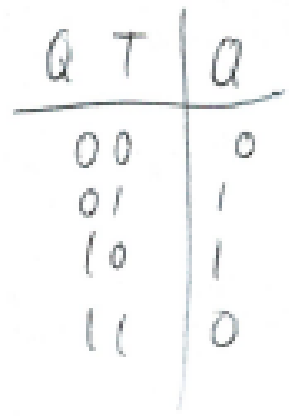

Name
Value
Sti...
240
260
277,
clk
1
Clock
j
1
<= 1
k
0
<= 1
q
1
rst
0
<= 1


Name
Value
Sti...
360
370 ns
400
clk
1
Clock
j
1
<= 1
k
1
<= 1
q
0
rst
0
<= 1

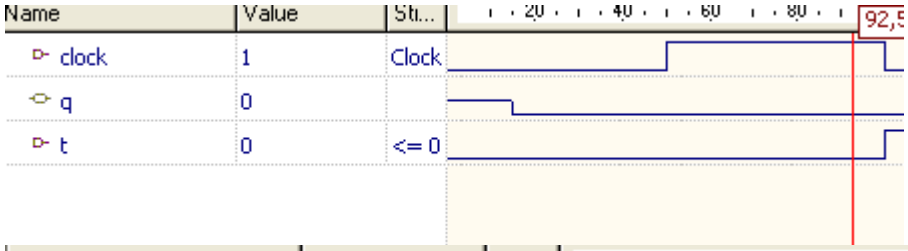


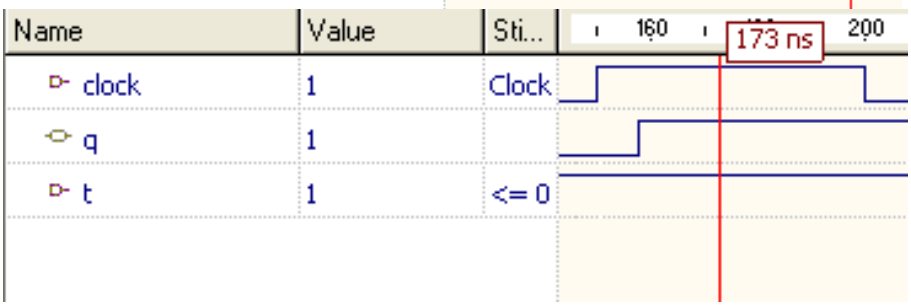
Flip-flop T

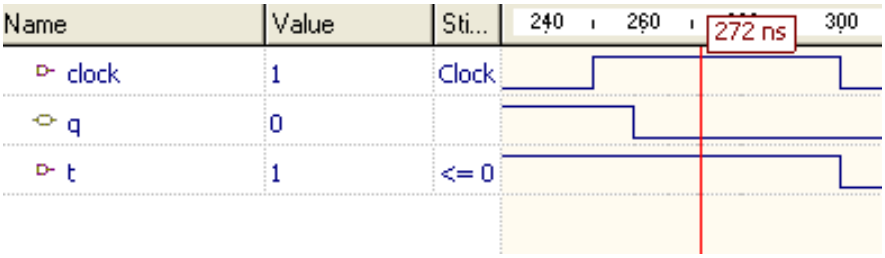
Se cumple la tabla de verdad y no se utiliza un método “signal” al no tener entradas retroalimentadas



```
10
11 architecture codigo of FFT is
12
13     signal tmp: std_logic;
14     begin
15         process (Clock)
16         begin
17         if Clock'event and Clock='1' then
18             if T='0' then
19             tmp <= tmp;
20             elsif T='1' then
21             tmp <= not (tmp);
22             end if;
23         end if;
24         end process;
25     Q <= tmp;
26
27 end codigo;
28
```







Conclusiones

Por medio de la realización de la practica se pudo comprender el método de implementación de los flip-flops mediante el uso de clock event y rising Edge dejando en claro la funcionalidad de cada uno así como la herramienta clock que se encuentra dentro del programa de simulación