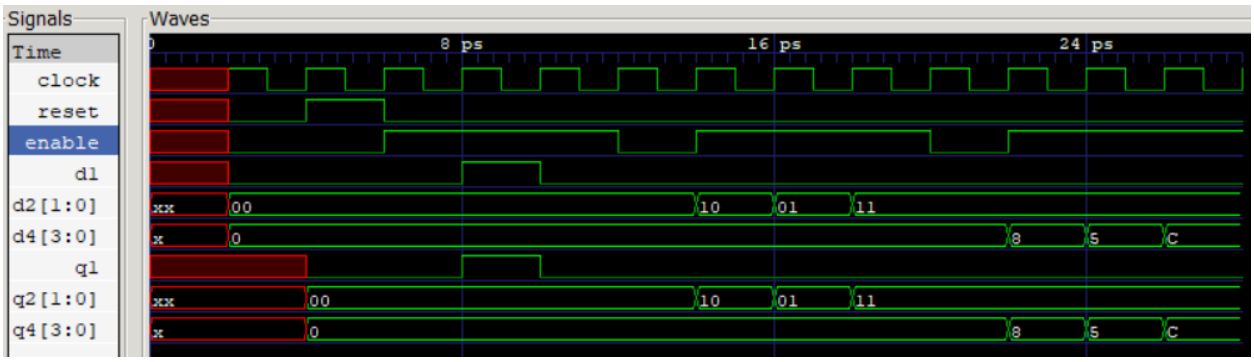
Guatemala, 31/Octubre/2020 Electrónica Digital
Ricardo Pellecer Orellana Carné 19072

Laboratorio 9

Ejercicio 1:

Para este ejercicio se creó un flip flop tipo D con el uso de un bloque always y condicionales. Además, haciendo uso de este primer módulo, se crearon módulos de flip flops tipo D de 2 y 4 bits. Para esto se crearon inputs y outputs de 2 y 4 bits y se instanciaron 2 y 4 flip flops respectivamente para asignarle valores a cada una de las salidas según cada una de las entradas. A estos módulos se les incluyó reset y enable.

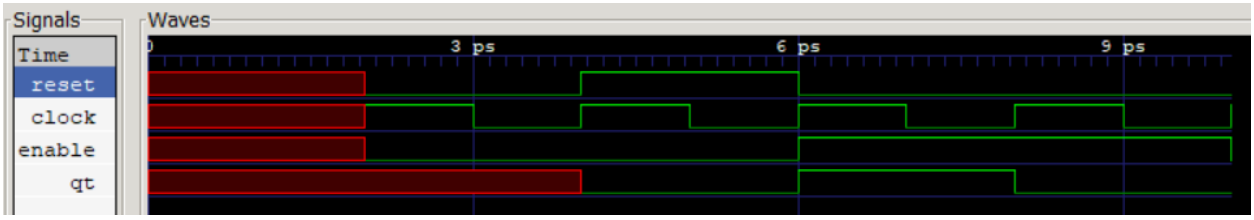
```
Un Bit
clock reset enable    D   |   Q
-------------------------|--------
1       0     0       0   |   x
0       0     0       0   |   x
1       1     0       0   |   0
0       1     0       0   |   0
1       0     1       0   |   0
0       0     1       0   |   0
1       0     1       1   |   1
0       0     1       1   |   1
1       0     1       0   |   0
0       0     1       0   |   0

Dos Bits
clock reset enable  D   |   Q
----------------------|--------
1       0      0    00  |   00
0       0      0    00  |   00
1       0      1    10  |   10
0       0      1    10  |   10
1       0      1    01  |   01
0       0      1    01  |   01
1       0      1    11  |   11
0       0      1    11  |   11

Cuatro Bits
clock reset enable     D   |   Q
--------------------------|--------
1       0      0     0000  |  0000
0       0      0     0000  |  0000
1       0      1     1000  |  1000
0       0      1     1000  |  1000
1       0      1     0101  |  0101
0       0      1     0101  |  0101
1       0      1     1100  |  1100
0       0      1     1100  |  1100
1       0      1     1100  |  1100
gtkwave code_tb.vcd code_tb.gtkw
```

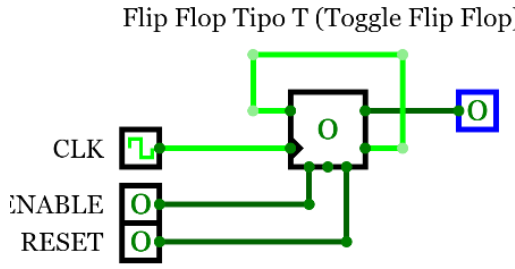


Ejercicio 2:

En este ejercicio se creó un flip flop tipo T o Toggle Flip Flop en base a un flip flop tipo D. Para esto se conectó el negado de la salida del flip flop a su entrada. Este módulo también contó con enable y reset.

```
Toggle Flip Flop
clock reset enable |   Q
-------------------|--------
1       0      0   |   x
0       0      0   |   x
1       1      0   |   0
0       1      0   |   0
1       0      1   |   1
0       0      1   |   1
1       0      1   |   0
0       0      1   |   0
1       0      0   |   0
gtkwave code_tb.vcd code_tb.gtkw
```

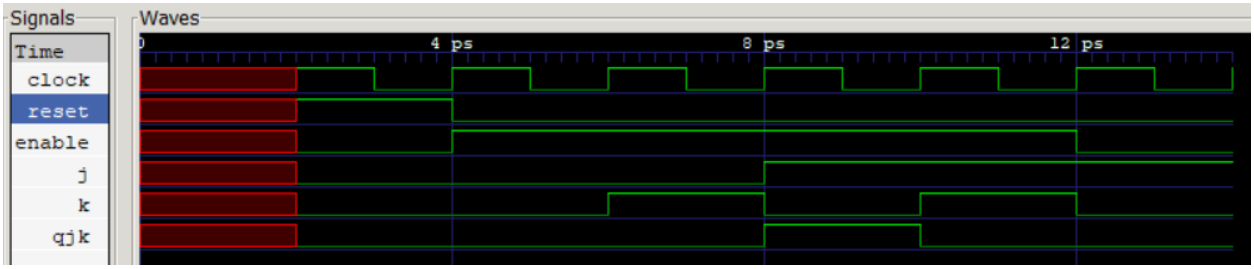


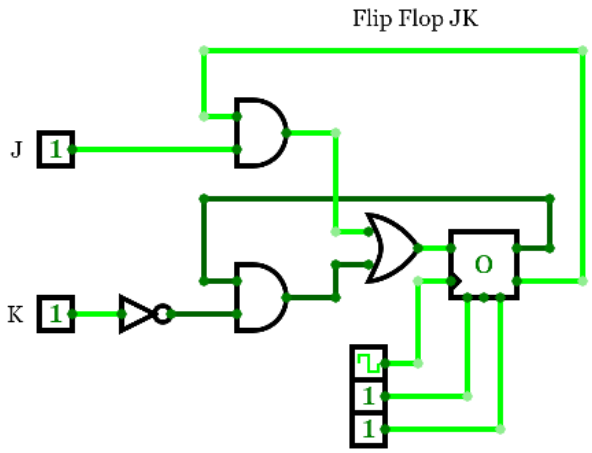


Ejercicio 3:

En este módulo se creó un Flip Flop JK con enable y reset. Este muestra un valor de Q previo cuando JK son 0, de 0 cuando J es 0 y K es 1, de 1 en el caso contrario y del negado de Q previo cuando ambos son 1.

```
Flip Flop JK
clock reset enable J   K |   Q
-------------------------|--------
1       1     0    0   0 |   0
0       1     0    0   0 |   0
1       0     1    0   0 |   0
0       0     1    0   0 |   0
1       0     1    0   1 |   0
0       0     1    0   1 |   0
1       0     1    1   0 |   1
0       0     1    1   0 |   1
1       0     1    1   1 |   0
0       0     1    1   1 |   0
1       0     0    1   0 |   0
0       0     0    1   0 |   0
1       0     0    1   0 |   0
gtkwave code_tb.vcd code_tb.gtkw
```

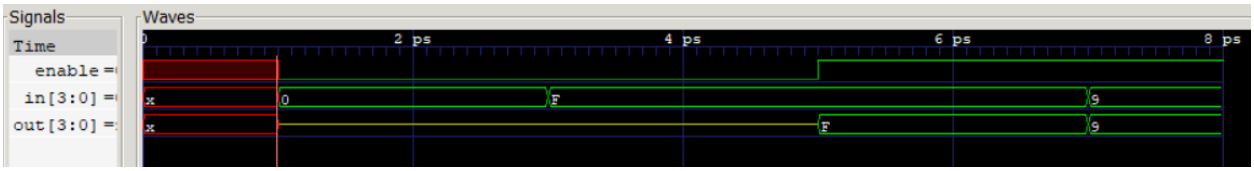




Ejercicio 4:

Se realizó un Buffer Tri-estado en un módulo utilizando un operador ternario, el cual indica que siempre que el enable está apagado, el resultado sea alta impedancia.

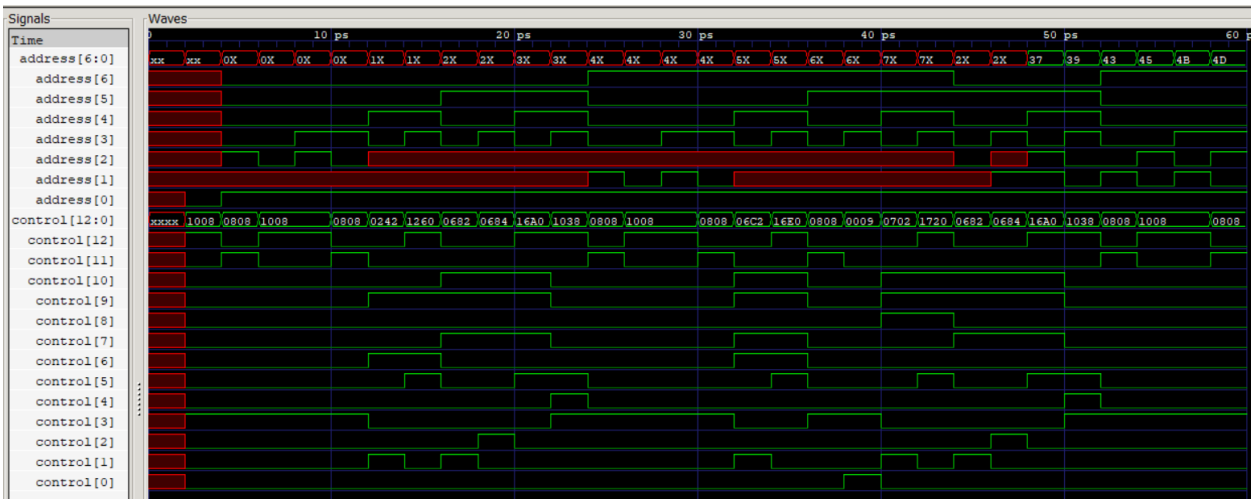
```
Buffer Tri-estado
enable   IN   |  OUT
-------------|------
0       0000 | zzzz
0       1111 | zzzz
1       1111 | 1111
1       1001 | 1001
gtkwave code_tb.vcd code_tb.gtkw
```



Ejercicio 5:

Se creó una look up table en un módulo usando los bloques casex, el cual toma en cuenta los don't cares de las tablas, lo cual fue útil para implementar las 128 posibilidades deseadas para crear todas las instrucciones de control para todos los opcodes posibles cuando se ancle al microcontrolador.

```
Microcode
address  |    control
---------|-------------
xxxxxx0  | 1000000001000
00001x1  | 0100000001000
00000x1  | 1000000001000
00011x1  | 1000000001000
00010x1  | 0100000001000
0010xx1  | 0001001000010
0011xx1  | 1001001100000
0100xx1  | 0011010000010
0101xx1  | 0011010000100
0110xx1  | 1011010100000
0111xx1  | 1000000111000
1000x11  | 0100000001000
1000x01  | 1000000001000
1001x11  | 1000000001000
1001x01  | 0100000001000
1010xx1  | 0011011000010
1011xx1  | 1011011100000
1100xx1  | 0100000001000
1101xx1  | 0000000001001
1110xx1  | 0011100000010
1111xx1  | 1011100100000
01000x1  | 0011010000010
0101x11  | 0011010000100
0110111  | 1011010100000
0111001  | 1000000111000
1000011  | 0100000001000
1000101  | 1000000001000
1001011  | 1000000001000
1001101  | 0100000001000
gtkwave code_tb.vcd code_tb.gtkw
```


Signals
Waves
Time
address[6:0]
address[6]
address[5]
address[4]
address[3]
address[2]
address[1]
address[0]
control[12:0]
control[12]
control[11]
control[10]
control[9]
control[8]
control[7]
control[6]
control[5]
control[4]
control[3]
control[2]
control[1]
control[0]
10 ps
20 ps
30 ps
40 ps
50 ps
xx xx 0X 0X 0X 0X 1X 1X 2X 2X 3X 3X 4X 4X 4X 4X 5X 5X 6X 6X 7X 7X 2X 2X 37 39 43 45 4B 4D
xxxx 1008 0808 1008 0808 0242 1260 0682 0684 16A0 1038 0808 1008 0808 06C2 16E0 0808 0009 0702 1720 0682 0684 16A0 1038 0808 1008 0808