CONTADOR ARBITRARIO

GRADO EN INGENIERÍA INFOMÁTICA
CURSO 2021/2022

VNiVERSiDAD D SALAMANCA
CAMPUS OF INTERNATIONAL EXCELLENCE

Contenido

PRESENTACIÓN

El trabajo que se nos ha propuesto es un contador arbitrario de 4 bits en el que se debe seguir la siguiente secuencia de números:

3→5→0→11→11→11→2→8→12→5

Y vuelta a empezar

Los números 11 y 5 aparecen en repetidas ocasiones por lo que los sustituiremos por otros valores que no aparezcan en la serie, en nuestro caso hemos sustituido el 11→ 10, 11→15 y 5→13, hemos elegido esos números ya que en binario son los que menos números de bits cambian.

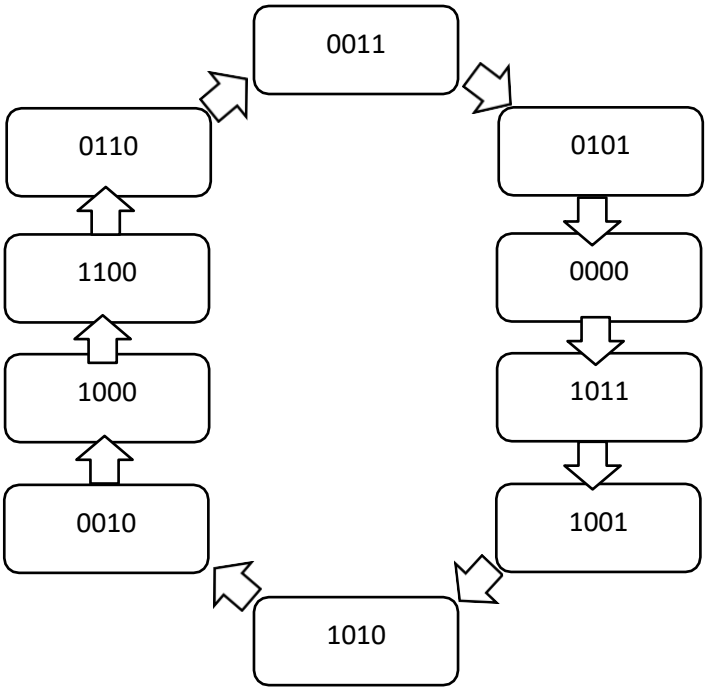
La serie quedaría así:

3→5→0→11→9→10→2→8→12→6

El circuito deberá de estar formado por 4 biestables JK, una señal de reloj y puertas lógicas AND, OR y XOR.

DESARROLLO DEL TRABAJO

Lo primero será, construir el diagrama de transición de estados que queda de la siguiente forma:



Para el siguiente paso nos será de gran ayuda la tabla de transiciones del biestable JK:

Actual	Siguiente	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

Una vez creado el diagrama de transiciones procedemos a crear la tabla de transiciones, en el caso de los valores que no se incluyen en la cadena principal se completaran con "X". Con la ayuda de la tabla de transiciones del biestable escribimos las "jotas" y las "kas" de los biestables en función de las variables.

Estas variables son las salidas de cada biestable: Q3, Q2,Q1,Q0 y K3,K2,K1,K0. Estas tablas reciben el nombre de tablas de Karnaugh y de ellas es de donde se extraen las ecuaciones para cada biestable para montar el circuito y que funcione el contador arbitrario y completar el mapa de transiciones y el diagrama final.

Actual	Siguiente	J3	K3	J2	K2	J1	K1	J0	K0
0	0000-1011	1	X	0	X	1	X	1	X
1	0001-XXXX	X	X	X	X	X	X	X	X
2	0010-1000	1	X	0	X	X	1	0	X
3	0011-0101	0	X	1	X	X	1	X	0
4	0100-XXXX	X	X	X	X	X	X	X	X
5	0101-0000	0	X	X	1	0	X	X	1
6	0110-XXXX	X	X	X	X	X	X	X	X
7	0111-XXXX	X	X	X	X	X	X	X	X
8	1000-1100	X	0	1	X	0	X	0	X
9	1001-XXXX	X	X	X	X	X	X	X	X
10	1010-1111	X	0	1	X	X	0	1	X
11	1011-1010	X	0	0	X	X	0	X	1
12	1100-1101	X	0	X	0	0	X	1	X
13	1101-0011	X	1	X	1	1	X	X	0
14	1110-XXXX	X	X	X	X	X	X	X	X
15	1111-0010	X	1	X	1	X	0	X	1

⇩ Q1 Q0 ⇨ Q3 Q2

J3	00	01	11	10
00	1	X	X	X
01	X	0	X	X
11	0	X	X	X
10	1	X	X	X

K3	00	01	11	10
00	X	X	0	0
01	X	X	1	X
11	X	X	1	0
10	X	X	X	0

J2	00	01	11	10
00	0	X	X	1
01	X	X	X	X
11	1	X	X	0
10	0	X	X	1

K2	00	01	11	10
00	X	X	0	X
01	X	1	1	X
11	X	X	1	X
10	X	X	X	X

J1	00	01	11	10
00	1	X	0	0
01	X	0	1	X
11	X	X	X	X
10	X	X	X	X

K1	00	01	11	10
00	X	X	X	X
01	X	X	X	X
11	1	X	0	0
10	1	X	X	0

J0	00	01	11	10
00	1	X	1	0
01	X	X	X	X
11	X	X	X	X
10	0	X	X	1

K0	00	01	11	10
00	X	X	X	X
01	X	1	0	X
11	0	X	1	1
10	X	X	X	X

Tras analizar estos mapas de Karnaugh, se llega a las siguientes ecuaciones:

J3->	nQ0
K3->	Q0*Q2

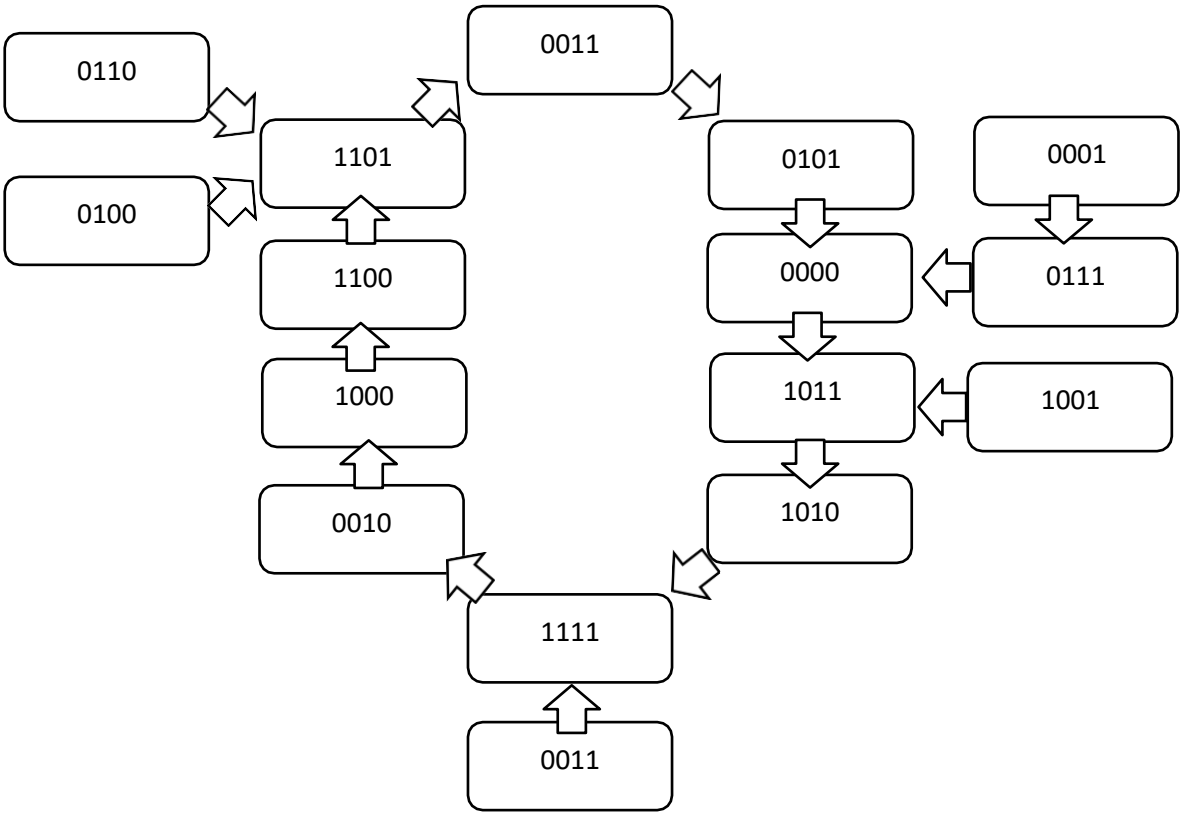
J2->	Q0*nQ3 + nQ0*Q3
K2->	Q0

J1->	nQ2*nQ3 + Q0*Q3
K1->	nQ3

J0->	nQ1*nQ3 + Q2 + Q1*Q3
K0->	nQ3*Q2 + Q1*Q3

Una vez obtenidas, obtenemos los valores de X en cada caso para obtener el diagrama de transición final:

	Actual	J3	K3	J2	K2	J1	K1	J0	K0	Siguiente	
0	0000	1	0	0	0	1	1	1	0	1011	11
1	0001	0	0	1	1	1	1	1	0	0111	7
2	0010	1	0	0	0	1	1	0	0	1000	8
3	0011	0	0	1	1	1	1	0	0	0101	5
4	0100	1	0	0	0	0	1	1	1	1101	13
5	0101	0	1	1	1	0	1	1	1	0000	0
6	0110	1	0	0	0	0	1	1	1	1101	13
7	0111	0	1	1	1	0	1	1	1	0000	0
8	1000	1	0	1	0	0	0	0	0	1100	12
9	1001	0	0	0	1	1	0	0	0	1011	11
10	1010	1	0	1	0	0	0	1	1	1111	15
11	1011	0	0	0	1	1	0	1	1	1010	10
12	1100	1	0	1	0	0	0	1	0	1101	13
13	1101	0	1	0	1	1	0	1	0	0011	3
14	1110	1	0	1	0	0	0	1	1	1111	15
15	1111	0	1	0	1	1	0	1	1	0010	2



Una vez construido el contador arbitrario, se procede a deshacer el cambio de los valores que estaban repetidos en la sucesión, para ello construimos una tabla y averiguamos sus ecuaciones.

Q	Q3	Q2	Q1	Q0	I3	I2	I1	I0	Q*
0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1	1
2	0	0	1	0	0	0	1	0	2
3	0	0	1	1	0	0	1	1	3
4	0	1	0	0	0	1	0	0	4
5	0	1	0	1	0	1	0	1	5
6	0	1	1	0	0	1	1	0	6
7	0	1	1	1	0	1	1	1	7
8	1	0	0	0	1	0	0	0	8
9	1	0	0	1	1	0	0	1	9
10	1	0	1	0	1	0	1	1	11
11	1	0	1	1	1	0	1	1	11
12	1	1	0	0	1	1	0	0	12
13	1	1	0	1	0	1	0	1	5
14	1	1	1	0	1	1	1	0	14
15	1	1	1	1	1	0	1	1	11

I3	00	01	11	10
00	0	0	1	1
01	0	0	0	1
11	0	0	1	1
10	0	0	1	1

I2	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	0	1	0	0
10	0	1	1	0

I1	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

I0	00	01	11	10
00	0	0	0	0
01	1	1	1	1
11	1	1	1	1
10	0	0	0	1

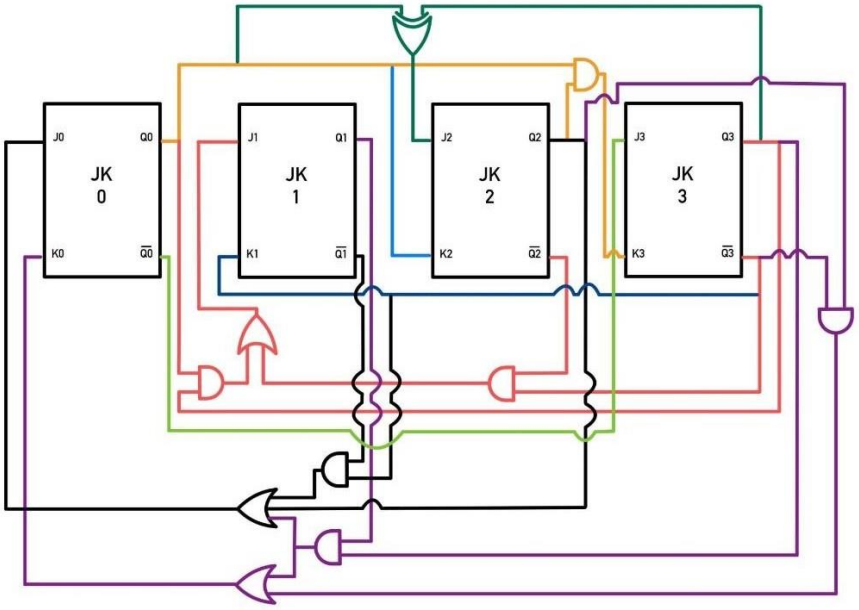
I3->	Q3 * (nQ0 + Q1 + nQ2)

I2->	Q2 * (nQ1 + nQ3 + nQ0)

I1->	Q1

I0->	Q0 + Q1*Q3*nQ2

Tras obtener las ecuaciones del circuito arbitrario y las ecuaciones para deshacer el cambio, podemos diseñar el circuito manualmente.



PROGRAMA FINAL

Después de todo lo explicado anteriormente llegamos a nuestro programan final, habiendo determinado las ecuaciones y programado sus diferentes módulos. Queda así el contador arbitrario en Verilog

```
//Modulo JK
module JK(output reg Q, output wire nQ, input wire J, input wire K, input
wire C);
not(nQ,Q);
initial
begin
Q=0;
end
always @(posedge C) // Se activa por cada flanco de subida
case ({J,K})
2'b10: Q=1;
2'b01: Q=0;
2'b11: Q=~Q;
endcase
endmodule
module ContadorArbitrario (output wire [3:0] Q, output wire [3:0] I, input
wire C);
wire [3:0] nQ;
wire J0,K0,J1,K1,J2,K2,J3,K3;
    wire nQ2nQ3,Q0Q3,nQ1nQ3,Q1Q3,nQ3Q2,nQ0Q3,Q0nQ3;



    //J0-K0
    and (nQ3Q2,nQ[3],Q[2]);
    or (K0,nQ3Q2,Q1Q3);//K0
    and (nQ1nQ3,nQ[1],nQ[3]);
    and (Q1Q3,Q[1],Q[3]);
    or (J0,nQ1nQ3,Q1Q3,Q[2]);//J0
    //J1-K1
    assign K1=nQ[3];//K1
```

```
    and (nQ2nQ3,nQ[2],nQ[3]);
    and  (Q0Q3,Q[0],Q[3]);
    or (J1,nQ2nQ3,Q0Q3);//J1
    //J2-K2
    assign  K2=Q[0];//K2
    and (nQ0Q3,nQ[0],Q[3]);
    and (Q0nQ3,Q[0],nQ[3]);
    or (J2,nQ0Q3,Q0nQ3);//J2
    //J3-K3
    and (K3,Q[0],Q[2]); //K3
    assign J3=nQ[0]; //J3

    JK JK0(Q[0],nQ[0],J0,K0,C);
    JK JK1(Q[1],nQ[1],J1,K1,C);
    JK JK2(Q[2],nQ[2],J2,K2,C);
    JK JK3(Q[3],nQ[3],J3,K3,C);
//Cambiar valores repetidos por sus respectivos
//La cuenta arbitraria definitiva con el cambio se guardara en un nuevo
registro I
wire or1, or2, and1, and2;

    //O0
    and (and2,Q[1],nQ[2],Q[3]);
    or (I[0],Q[0],and2);
    //O1
    assign I[1]=Q[1];
    //O2
    or (or2,nQ[1],nQ[3],nQ[0]);
    and (I[2],Q[2],or2);
    //O3
    or (or1, nQ[0],nQ[2],Q[1]);
    and (I[3],Q[3],or1);

endmodule
```

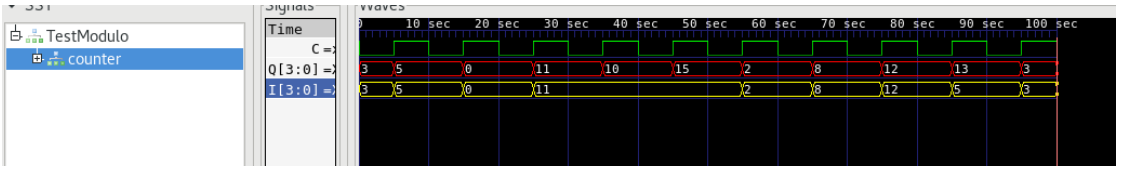
```
//Módulo para probar el circuito.
module TestModulo;
reg C;
wire [3:0] Q;
wire [3:0] I;
ContadorArbitrario counter (Q,I,C);
always #5 C=~C;

initial
begin
$dumpfile("ContadorArbitrarioGTK.dmp");
$dumpvars(2, counter, Q);

counter.JK0.Q<=1; //La serie empezara por el numero
counter.JK1.Q<=1; // 0011 es decir el 3 en decimal
counter.JK2.Q<=0;
counter.JK3.Q<=0;
C=0;
//Para comprobar el programa en el terminal
$monitor($time, " Reloj C(%b) Contador Q |%d| |%b| Contador I
|%d||%b|",C,Q,Q,I,I);
#100;
$dumpoff;
$finish;
end
endmodule
```

RESULTADOS

Por último, comprobamos los resultados con el GTKWave y el terminal.



En el GTKWave podemos comprobar como la progresión de los números es correcta y que funciona por flanco de subida, es decir, cambia el número cuando el valor de la señal de reloj (C), pasa a estar en alto.

En el terminal podemos comprobar igualmente que los valores son correctos.

```
VCD info: dumpfile ContadorArbitrarioGTK.dmp opened for output.
                   0 Reloj C(0) Contador Q | 3| |0011| Contador I | 3||0011|
                   5 Reloj C(1) Contador Q | 5| |0101| Contador I | 5||0101|
                  10 Reloj C(0) Contador Q | 5| |0101| Contador I | 5||0101|
                  15 Reloj C(1) Contador Q | 0| |0000| Contador I | 0||0000|
                  20 Reloj C(0) Contador Q | 0| |0000| Contador I | 0||0000|
                  25 Reloj C(1) Contador Q |11| |1011| Contador I |11||1011|
                  30 Reloj C(0) Contador Q |11| |1011| Contador I |11||1011|
                  35 Reloj C(1) Contador Q |10| |1010| Contador I |11||1011|
                  40 Reloj C(0) Contador Q |10| |1010| Contador I |11||1011|
                  45 Reloj C(1) Contador Q |15| |1111| Contador I |11||1011|
                  50 Reloj C(0) Contador Q |15| |1111| Contador I |11||1011|
                  55 Reloj C(1) Contador Q | 2| |0010| Contador I | 2||0010|
                  60 Reloj C(0) Contador Q | 2| |0010| Contador I | 2||0010|
                  65 Reloj C(1) Contador Q | 8| |1000| Contador I | 8||1000|
                  70 Reloj C(0) Contador Q | 8| |1000| Contador I | 8||1000|
                  75 Reloj C(1) Contador Q |12| |1100| Contador I |12||1100|
                  80 Reloj C(0) Contador Q |12| |1100| Contador I |12||1100|
                  85 Reloj C(1) Contador Q |13| |1101| Contador I | 5||0101|
                  90 Reloj C(0) Contador Q |13| |1101| Contador I | 5||0101|
                  95 Reloj C(1) Contador Q | 3| |0011| Contador I | 3||0011|
                 100 Reloj C(0) Contador Q | 3| |0011| Contador I | 3||0011|
i0958754@lipc17:~/Descargas$
```