Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE-0523 Circuitos Digitales II
II ciclo 2022

Tarea 1

Descripción conductual de un contador binario sincrónico

Javier Solera Bolaños B66963
Grupo 1

Profesor: Enrique Coen

13/09/2022

Índice

1. Resumen

La tarea consiste en crear un sumador de 4 bits y 16 bits, con 4 distintos modos, una cuenta ascendente, una cuenta descendente, una cuenta descendente de 3 en 3 y una carga paralela, mediante el programa de descripción de hardware verilog. Se realizaron 4 archivos que serian el contador, el cual contiene un modulo con la lógica para el correcto funcionamiento del contador de 4 bits y contiene otro modulo para el contador de 16 bits, este modulo instancia 4 veces al contador de 4 bits, de manera que se pueda crear un contador de 16 bits. Un tester, el cual se encarga de inicializar y determinar las pruebas que se le hacen al contador. Un flipflop, el cual se encarga de que el contador funcione siempre que el enable este en alto. Y un Testbench, el cual se encarga de conectar el contador al tester y también de generar el gtkwave.

Se realizaron pruebas para los distintos modos y para ambos contadores, se obtuvo el correcto funcionamiento de los contadores, el contador funcionara siempre y cuando el enable estuviera en alto, que cuando era necesario un llevo, el ripple carry out se ponía en alto y que el contador cambiara de función dependiendo del modo en que se encontrara.

Como conclusión, Se comprendio el uso del programa verilog, mediante la creacion de modulos, la logica implementada y el conectar los diferentes modulos que se encotraban en los archivos del contador, tester, testbench y flipflop. Se analizo el funcionamiento del llevo, y se implemento en la lógica del contador para que este funcionara correctamente siempre que hubiera un llevo.

2. Descripción Arquitectónica

Es un posible diseño para el contador de 4 bits y el contador de 16 bits, ambos contienen logica combinacional y secuencial separados por bloques. Estos bloques se juntan en un banco de pruebas, que seria el testbench, como se observa en la figura 1

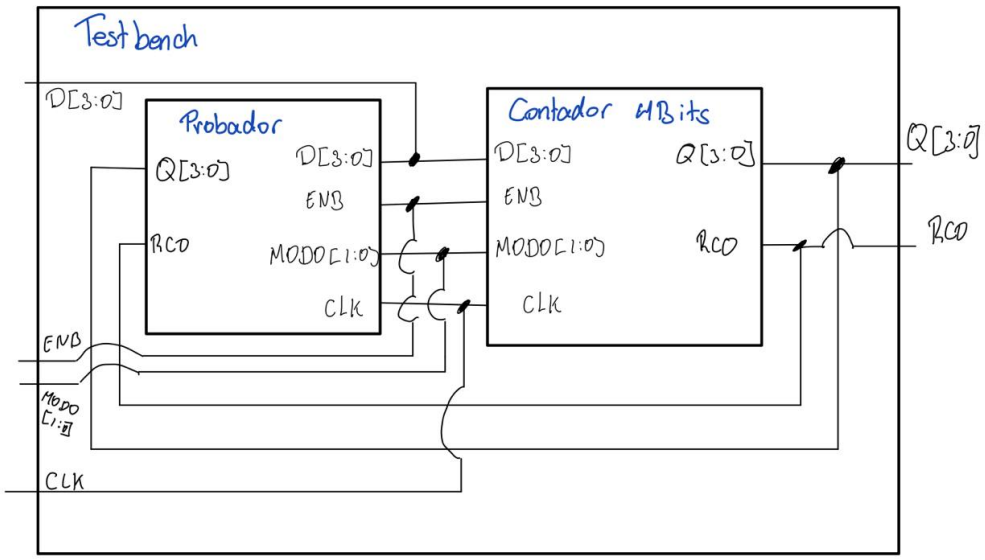

Figura 1: Diagrama de Bloques, contador 4 bits

Para formar un contador de 16 bits, se pensó en utilizar 4 veces el contador de 4 bits e ir acomodando las salidas de los contadores hasta formar una salida de 16 bits, tal y como se muestra en la figura 2, el contador 1 va a funcionar normalmente y los 4 bits de este contador

van a ser LSB de la salida Q, para contador 2 va a tener como reloj el Ripple carry out del contador 1 y estos 4 bits se van a colocar a la par de los del contador 1, para el contador 3 el reloj va a ser el ripple carry out del contador 2 y estos 4 bits se colocan a la par de los 4 bits del contador 2 y por últimos para el contador 4, el reloj va a ser el ripple carry out del contador 3 y estos 4 bits van a ser los MSB de la salida Q.

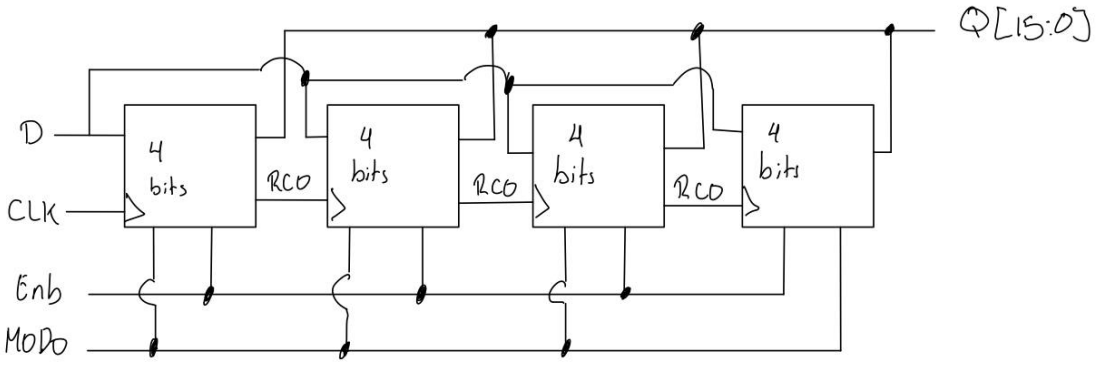

Figura 2: Diagrama de Bloques para la parte del contador de 16 bits

.

Al juntar todos los bloques en el banco de pruebas (testbench), se obtiene el contador de 16 bits con pruebas de funcionamiento, como se puede observar en el diagrama de la figura 3

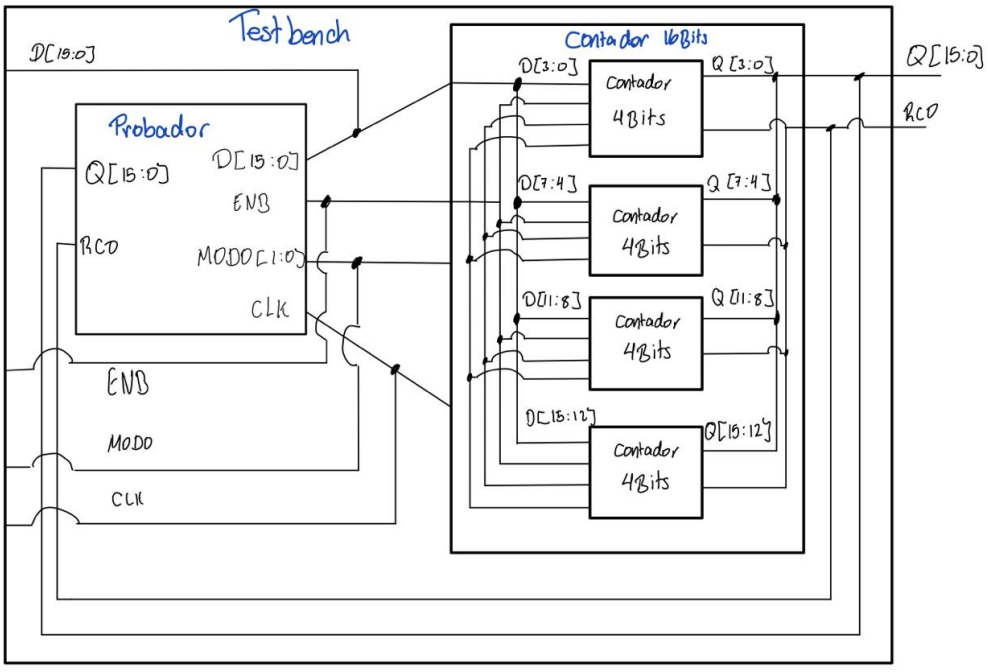

Figura 3: Diagrama de Bloques, contador 16 bits

3. Plan de Pruebas

3.1. Contador de 4 Bits

Se realizaron 4 pruebas para cada modo, cuenta ascendente, cuenta descendente, cuenta descendente de 3 en 3 y carga en paralelo:

1. Cuenta ascendente MODO[1 : 0] = 00: Se desea realizar una cuenta ascendente de 1 en 1 y el Ripple carry out (RCO) de va a activar cada vez que hay un llevo

   1.1) El registro inicia en 0000, para lograr esto se toma el MODO[1 : 0] = 11 y D[3 : 0] = 0000

   1.2) Cada flanco positivo del reloj, me activa la suma, entonces la salida Q se vería de esta manera: Q[3 : 0] =0000 − > Q[3 : 0] =0001 − > Q[3 : 0] =0010 − > ... − > Q[3 : 0] =1111.

   1.3) Cuando pasan 15 flanco positivos del reloj, el contador llega a Q[3 : 0] =1111, el RCO se activa y se pone en 1, dando a entender que hay un llevo.

2. Cuenta descendente MODO[1 : 0] = 01: Se desea realizar una cuenta descendente de 1 en 1 y el Ripple carry out (RCO) de va a activar cada vez que hay un llevo

   2.1) El registro inicia en 1111, para lograr esto se toma el MODO[1 : 0] = 11 y D[3 : 0]= 1111

   2.2) Cada flanco positivo del reloj, me activa la resta, entonces la salida Q se vería de esta manera: Q[3 : 0] =1111 − > Q[3 : 0] =1110 − > Q[3 : 0] =1110 − > ... − > Q[3 : 0] =0000.

   2.3) Cuando pasan 15 flanco positivos del reloj, el contador llega a Q[3 : 0] =0000, el RCO se activa y se pone en 1, dando a entender que hay un llevo.

3. Cuenta descendente 3 en 3 MODO[1 : 0] = 10: Se desea realizar una cuenta descendente de 3 en 3 y el Ripple carry out (RCO) de va a activar cada vez que hay un llevo

   3.1) El registro inicia en 1111, para lograr esto se toma el MODO[1 : 0] = 11 y D[3 : 0]= 1111

   3.2) Cada flanco positivo del reloj, me activa la resta, entonces la salida Q se vería de esta manera: Q[3 : 0] =1111 − > Q[3 : 0] =1100 − > Q[3 : 0] =1001 − > ... − > Q[3 : 0] =0000.

   3.3) Cuando pasan 5 flanco positivos del reloj, el contador llega a Q[3 : 0] =0000, el RCO se activa y se pone en 1, dando a entender que hay un llevo. En esta prueba, dependiendo de donde la resta empieza, el ultimo valor de Q antes de que se encienda el RCO puede ser Q[3 : 0] =0001 o Q[3 : 0] =0010

4. Carga en Paralela MODO[1 : 0] = 11:

   4.1) El registro inicia en XXXX, para lograr esto se toma el MODO[1 : 0] = 11 y D[3 : 0] =XXXX

   4.2) Cada flanco positivo del reloj, a la entrada D, se le va a sumar 1 y la salida Q va a copiar el valor que haya en D, por ejemplo si en D[3 : 0]= 1010, la salida Q en el siguiente flanco positivo del reloj va a ser Q[3 : 0]= 1010

   4.3) En esta prueba el RCO se va a encender siempre que D[3 : 0]= 1111, porque como D va aumentando, ahi es cuando se genera un llevo.

3.2. Contador de 16 Bits

Se realizaron 4 pruebas para cada modo, cuenta ascendente, cuenta descendente, cuenta descendente de 3 en 3 y carga en paralelo, de manera que se usa 4 contadores de 4 bits crear un contador de 16 bits:

1. Cuenta ascendente MODO[1 : 0] = 00: Se desea realizar una cuenta ascendente de 1 en 1 y el Ripple carry out (RCO) se va a activar cada vez que hay un llevo y cuando el RCO se activa, el contador 2 entraba en funcionamiento, luego cuando el contador 2 activa el RCO, el contador 3 entra en funcionamiento y asi con el contador 4.

   1.1) El registro inicia en 0000000000000000, para lograr esto se toma el MODO[1 : 0] = 11 y D[15 : 0] = 0000000000000000

   1.2) Cada flanco positivo del reloj, me activa la suma, entonces la salida Q se vería de esta manera: Q[15 : 0] =0000000000000000 − > Q[15 : 0] =0000000000000001 − > Q[15 : 0] =0000000000000010 − > ... − > Q[15 : 0] =1111111111111111.

   1.3) En esta prueba se trato de analizar que el RCO funcionara correctamente, es decir, que cada vez que el RCO se activara, el contador 2 empezara a funcionar.

2. Cuenta descendente MODO[1 : 0] = 01: Se desea realizar una cuenta descendente de 1 en 1 y el Ripple carry out (RCO) de va a activar cada vez que hay un llevo, al igual que el modo anterior, cuando el RCO se activa, activa el contador 2 y cuando el contador 2 activa el RCO, el contador 3 se activa y asi sucesivamente.

   2.1) El registro inicia en 1111111111111111, para lograr esto se toma el MODO[1 : 0] = 11 y D[15 : 0]= 1111111111111111

   2.2) Cada flanco positivo del reloj, me activa la resta, entonces la salida Q se vería de esta manera: Q[15 : 0] =1111111111111111 − > Q[15 : 0] =1111111111111110 − > Q[15 : 0] =1111111111111110 − > ... − > Q[15 : 0] =0000000000000000.

   2.3) En esta prueba se trato de analizar que el RCO funcionara correctamente, es decir, que cada vez que el RCO se activara, el contador 2 empezara a restar y asi sucesivamente.

3. Cuenta descendente 3 en 3 MODO[1 : 0] = 10: Se desea realizar una cuenta descendente de 3 en 3 y el Ripple carry out (RCO) de va a activar cada vez que hay un llevo

   3.1) El registro inicia en 1111111111111111, para lograr esto se toma el MODO[1 : 0] = 11 y D[15 : 0]= 1111111111111111

   3.2) Cada flanco positivo del reloj, me activa la resta, entonces la salida Q se vería de esta manera: Q[3 : 0] =1111111111111111 − > Q[3 : 0] =1111111111111100 − > Q[3 : 0] =1111111111111001 − > ... − > Q[3 : 0] =0000000000000000.

   3.3) Al igual que en las pruebas anteriores, RCO se va a activar siempre que alguno de los contadores llegue a estos valores: 0000, 0001 o 0010, y asi activara al siguiente contador.

4. Carga en Paralela MODO[1 : 0] = 11:

   4.1) El registro inicia en XXXXXXXXXXXXXXXX, para lograr esto se toma el MODO[1 : 0] = 11 y D[3 : 0] =XXXXXXXXXXXXXXXX

4.2) Cada flanco positivo del reloj, a la entrada D, se le va a sumar 1 y la salida Q va a copiar el valor que haya en D, por ejemplo si en D[15 : 0]= 1010000011001010, la salida Q en el siguiente flanco positivo del reloj va a ser Q[15 : 0]= 1010000011001010

4.3) En esta prueba el RCO se va a encender siempre que D[3 : 0]= 1111 de alguno de los 4 contadores de 4 bits.

4. Instrucciones de utilización de la simulación

Es importante que los archivos flipflop.v, tb_Counter.v, teste.v, Counter.v y el Makefile se encuentren en la misma carpeta.

Entonces puede abrir la terminal y ubicarse en donde están los archivos mencionados anteriormente y correr el comando make, este ejecutará el archivo Makefile y complilará y correrá los archivos.

También puede seguir las instrucciones en la terminal son: iverilog -o tb.vvp tb_Counter.v, esta linea lo que hace es compilar los archivos. Luego tienen que escribir en la terminal: vvp tb.vvp, esto lo que hace es correr la simulación y deesplegarlo en la terminal. Y por ultimo, se escribe en la terminal: gtkwave tb.vcd, esta lo que hace es abrir el contador en la aplicacion gtkwave.

5. Ejemplos de los resultados

Los resultados se muestran en la siguientes figuras,

5.1. Contador de 4 Bits

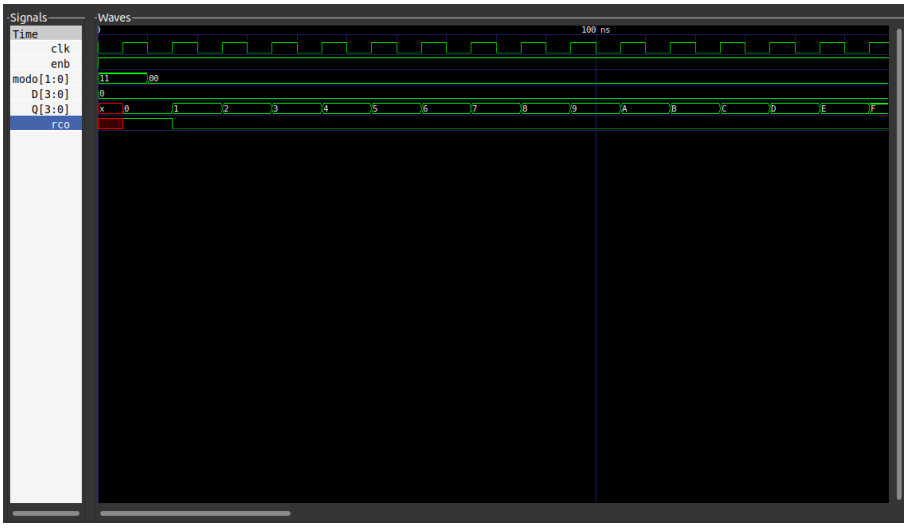

Figura 4: Cuenta Ascendente, Contador de 4 Bits

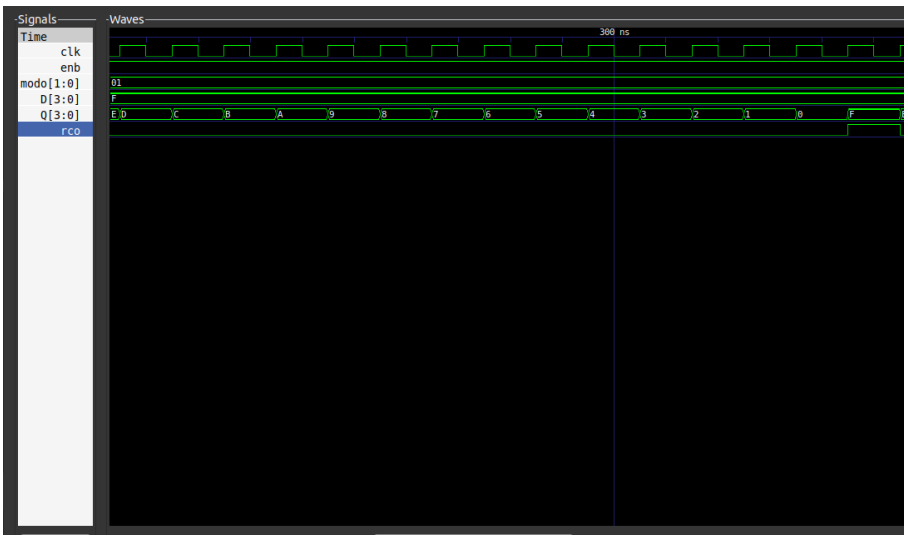

Figura 5: Cuenta Descendente, Contador de 4 Bits

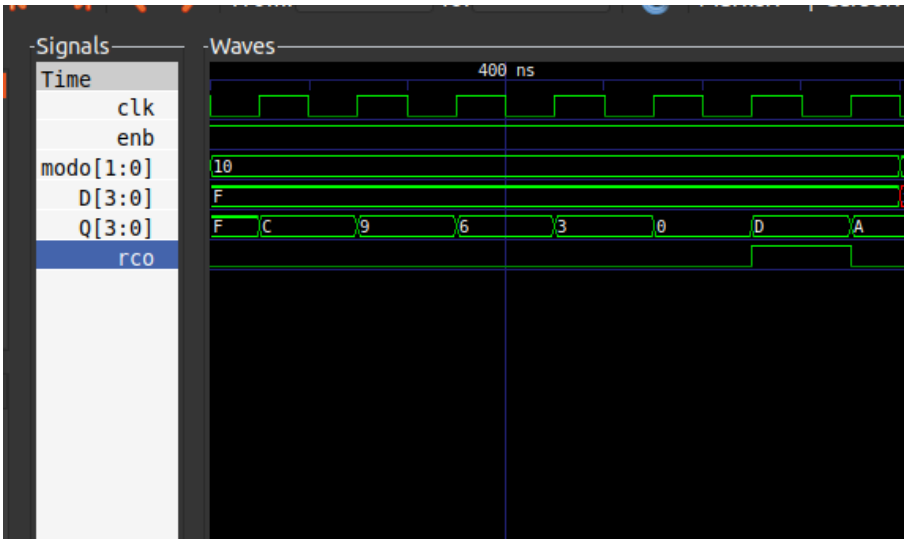

Figura 6: Cuenta Descendente de 3 en 3, Contador de 4 Bits

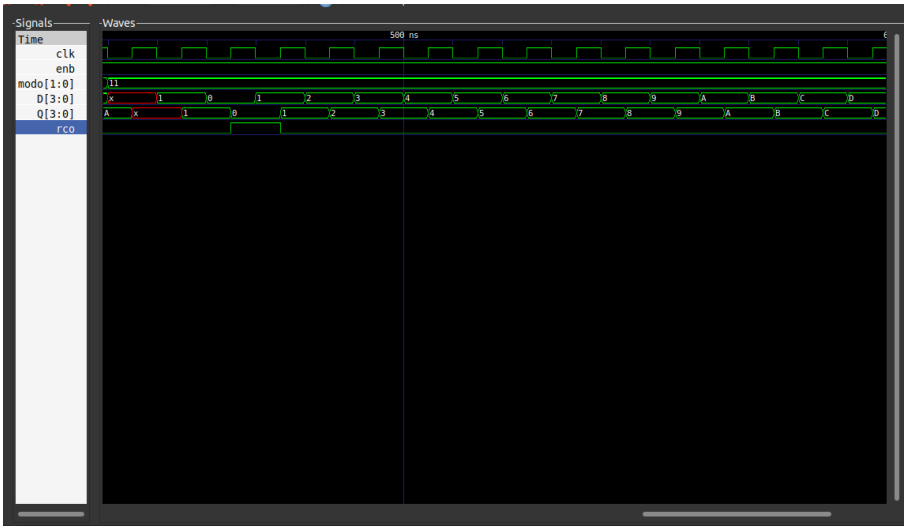

Figura 7: Carga en Paralela, Contador de 4 Bits

5.2. Contador de 16 Bits

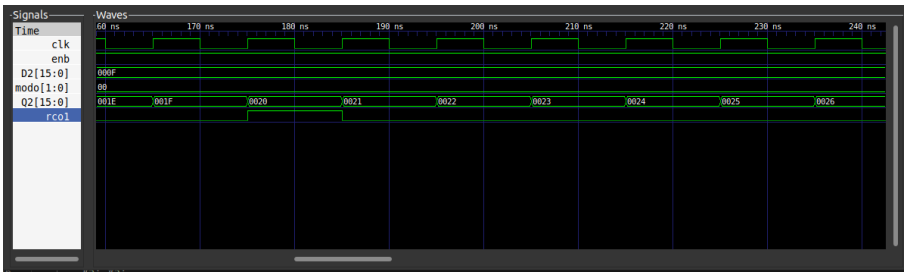

Figura 8: Cuenta Ascendente, Contador de 16 Bits

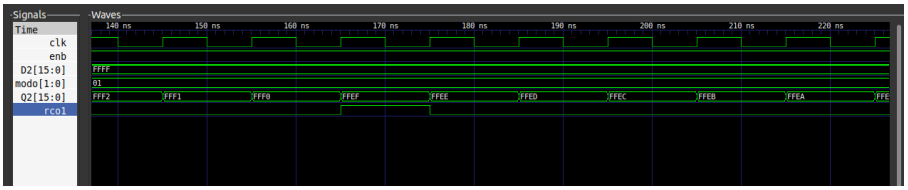

Figura 9: Cuenta Descendente, Contador de 16 Bits

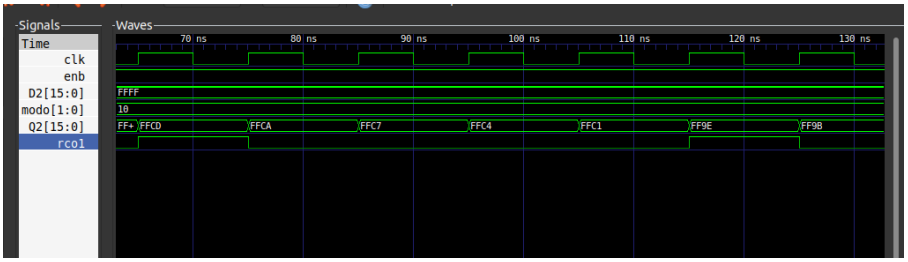

Figura 10: Cuenta Descendente de 3 en 3, Contador de 16 Bits

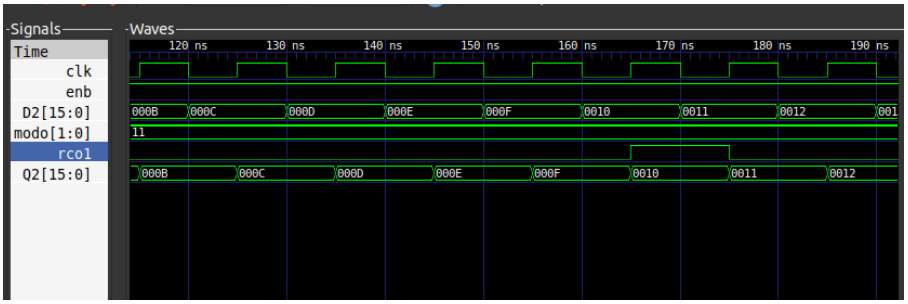

Figura 11: Carga en Paralela, Contador de 16 Bits

6. Conclusiones y recomendaciones

Se concluye que, las pruebas realizadas para los 2 contadores funcionaron como se esperaba, ya que se cumplen con los requerimientos descritos en la tarea, se determino que el ripple carry out se activara siempre que en la suma o en la resta se obtuviera un llevo, se logro implementar el diseño que se comento en la sección de descripción arquitectónico, separando los archivos en un probador, contador y un testbench.

Se recomienda siempre iniciar con la parte de diseño arquitectónico, ya que esto puede facilitar y dar un visualización de como es que se desea el trabajo y en caso de que haya algun error o problema, es mas sencillo solucionarlo en el diagrama de bloques que en el codigo fuente. También se recomienda hacer una verificación de funcionamiento, de manera que se cumplan todas las especificaciones dadas.