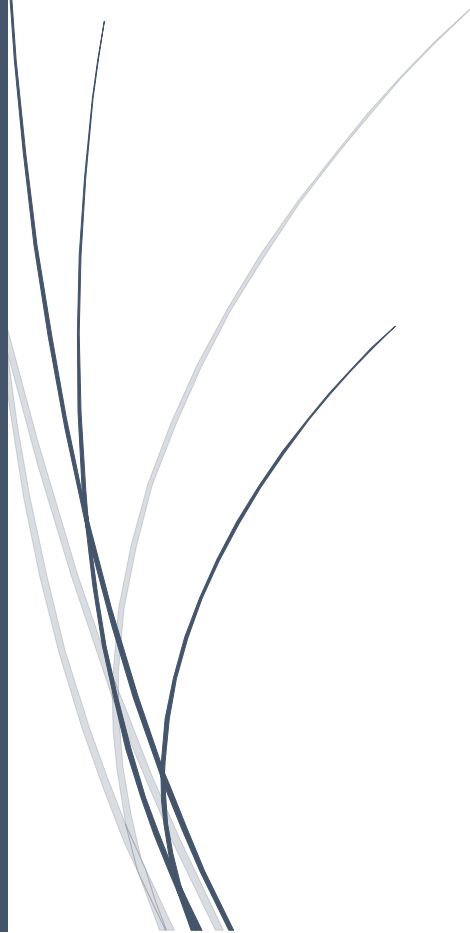


A dark blue vertical bar is on the left. A blue arrow points right from it, containing the date.

15-1-2014

Diseño en VHD: Máquina expendedora de refrescos

Trabajo en grupo de la asignatura:
“Sistemas electrónicos Digitales”

Several thin, curved lines in dark blue and light grey originate from the bottom left and sweep upwards and to the right.

Alonso Hernández, Francisco Javier.	49723.
López Benedí, Javier.	49462.
Lozano Catalán, Miguel Ángel.	49504.

0.1. Índice de contenidos

1.	Introducción – Máquina expendedora de refrescos.....
2.	Descripción de la estrategia y algoritmos desarrollados para realizar el trabajo, justificando las soluciones adoptadas.....
3.	Casos de uso del diseño.....
4.	Descripción del esquema de componentes del diseño.....
5.	Esquemas y diagramas del diseño.....
6.	Explicación detallada del funcionamiento de cada uno de los bloques funcionales y de su interfaz.....
6.1.	Sincronizador.....
6.2.	Codificador de monedas.....
6.3.	Sumador recursivo.....
6.4.	Codificador de producto.....
6.5.	Comparador.....
6.6.	Restador.....
6.7.	Devolución.....
6.8.	Máquina de estados.....
6.9.	Conversor Binario 8 bits – BCD 4 bits.....
6.10.	Prescaler (Divisor de frecuencia de reloj) de 50MHz a 100 Hz.....
6.11.	Decodificador BCD 4 bits – 7 segmentos para Display de Leds.....
6.12.	Multiplexor con cambio de señal de consigna automático por pulso de reloj, para control de los 4 displays de la FPGA de forma independiente.....
7.	Entradas / Salidas utilizadas de la placa de entrenamiento Spartan3.....
8.	Testbenches realizados.....
8.1.	Notas generales.....
8.2.	Sincronizador: SYNC_tb.vhd.....
8.3.	Estabilizador (Antirrebote): Estabilizador_tb.vhd.....
8.4.	Codificador de monedas: Codificador_monedas_tb.vhd.....
8.5.	Codificador de producto: Codificador_pdto_tb.vhd.....
8.6.	Comparador de tres salidas: Comparador_3outs_tb.vhd.....
8.7.	Sumador: Sumador_tb.vhd.....
8.8.	Restador: Restador_tb.vhd.....
8.9.	Devolución: devolución_tb.vhd.....
8.10.	Máquina de estados: máquina_tb.vhd.....
8.11.	Máquina expendedora: Top_Expendedora_tb.vhd.....

1. Introducción- Máquina expendedora de refrescos.

En este trabajo se ha realizado el diseño de una máquina expendedora de refrescos. En el enunciado se indicaban las siguientes condiciones que debe cumplir el diseño:

- Que admitiese monedas de 10c, 20c, 50c y 1€.
- Que solo tuviese un producto seleccionado automáticamente.
- Que si se introducía más dinero que el importe exacto diese un error y devolviese todo.
- Que en caso de ser el dinero justo entregase el producto.

Sobre esta base se han introducido las siguientes mejoras en funcionalidad:

- Acepta también monedas de 5c y de 2€.
- La máquina permite elegir entre más de un producto, tres exactamente, de distintos importes.
- En caso de introducirse una cantidad de dinero superior al importe del producto la máquina devolverá el cambio al entregar el producto.
- La máquina incluye la posibilidad de devolver todo el dinero introducido sin entregar ningún producto, en el caso de que el cliente cambiase de idea respecto a su compra.
- Si introdujese menos dinero que el necesario y eligiese un producto, la máquina simplemente se mantendría a la espera de que siguiese introduciendo dinero o eligiese otro.

2. Descripción de la estrategia y algoritmos desarrollados para realizar el trabajo, justificando las soluciones adoptadas.

Para realizar el diseño en componentes de esta máquina expendedora de refrescos se ha intentado dividir al máximo las funciones que realiza cada parte, de forma que sea posible la reutilización de los componentes.

Este modo de diseño permite la ampliación de funcionalidades de manera sencilla, como la incorporación de más productos, o posibles modificaciones, por ejemplo, si se cambiase un producto por otro con distinto precio.

Para permitir una mayor modularidad, se ha creado una máquina de estados central que va controlando el uso de la expendedora de refrescos a lo largo de los diferentes pasos de su funcionamiento, facilitando cambios o mejoras al diseño.

3. Casos de uso del diseño.

Los casos de uso que se han tenido en cuenta para realizar el diseño son:

- 1) Compra de producto con dinero introducido exacto: El cliente introduce el dinero y a continuación selecciona un producto con coste igual al dinero introducido. En este caso, el más simple, una vez la máquina los compara y ve que son iguales, entrega el producto.
- 2) Compra de producto con dinero introducido superior al importe: Tras introducir el dinero y seleccionar un producto con coste menor, la máquina procede a devolver el cambio y a continuación entrega el producto. Para esto existe una etapa que se encarga de calcular cuánto dinero ha de devolver y otra que se encarga de decir el tipo de monedas a devolver.
- 3) Selección de un producto con dinero introducido insuficiente: En este caso tras seleccionar un producto de coste superior, la máquina ignoraría la petición y permanecería a la espera de que siguiese introduciendo más dinero o eligiese un producto de coste inferior. Una vez se haga una de estas dos cosas seguiría como en los dos primeros casos de uso.
- 4) Devolución del dinero introducido: En este caso el cliente pulsaría el botón de devolución del dinero tras haber introducido dinero. La máquina internamente procesaría que el cliente ha elegido un producto especial de coste cero y que no entrega nada, con lo que pasaría al segundo caso de uso, devolviendo todo.
- 5) El cliente pulsa dos productos distintos seguidos o un producto mientras está devolviendo monedas: La máquina haría únicamente caso a la primera pulsación, ignorando la segunda.
- 6) El cliente pulsa devolución tras pulsar un producto: Igual que en el caso anterior la máquina ignora la pulsación de la devolución del dinero al tratarla como un producto más.

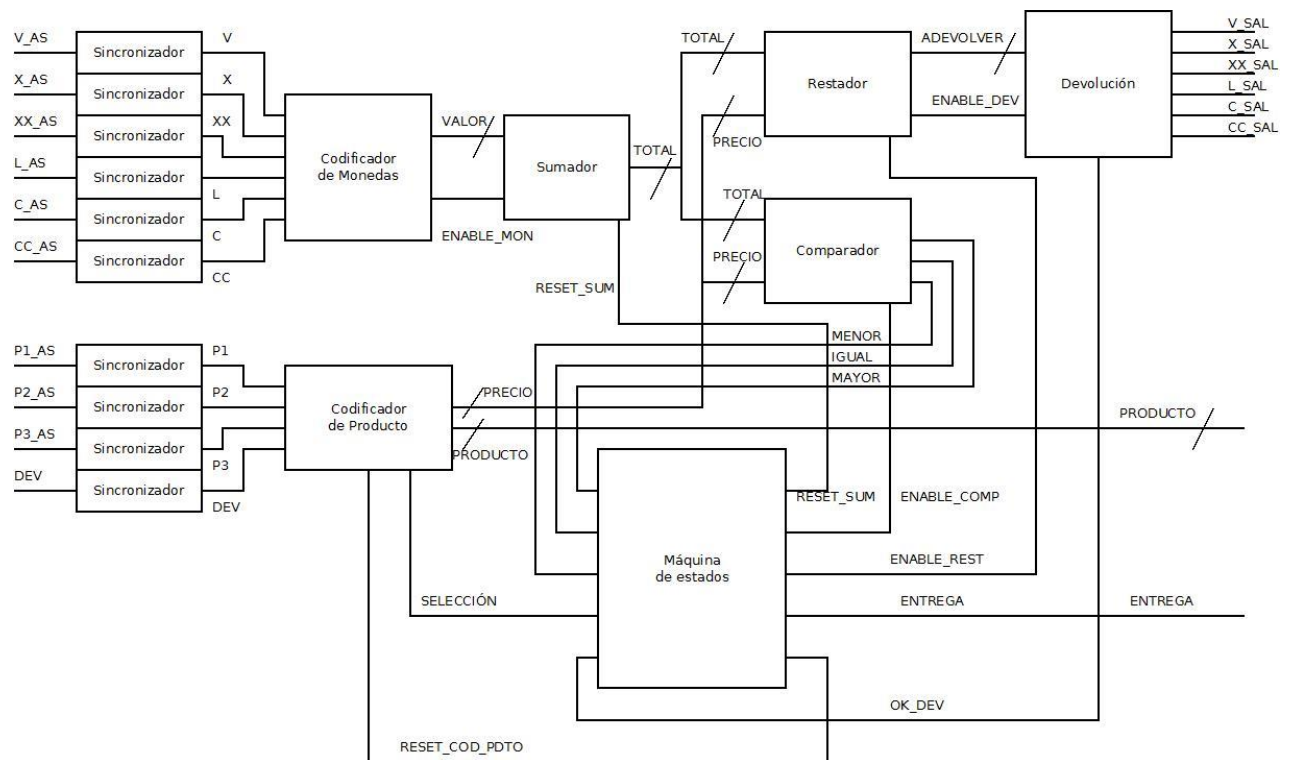
4. Descripción del esquema de componentes del diseño.

El funcionamiento del circuito es el siguiente:

- El cliente, al introducir monedas, introduce unas señales en el circuito (V_AS, X_AS, XX_AS...) que activan los sincronizadores.
- Los sincronizadores convierten las señales asíncronas en síncronas con el reloj del sistema (V, X, XX, L, C, CC) y los flancos de las señales ya sincronizadas son recibidos por el codificador de monedas.
- El codificador de monedas envía al sumador el valor de cada una de las monedas introducidas (VALOR) y éste va sumando el total introducido (TOTAL).
- Tras introducir las monedas, el cliente selecciona el producto que desea (P1_AS, P2_AS, P3_AS, DEV).
- Al igual que antes, un sincronizador convierte la señal en síncrona (P1, P2, P3, DEV) y la envía al codificador de producto, en cuyas salidas quedan enclavados el precio (PRECIO) y el código del producto (PRODUCTO) seleccionado.
- En el comparador se introducen las señales correspondientes al dinero introducido (TOTAL) y al precio del producto seleccionado (PRECIO).
- En el caso de que sea menor el dinero introducido al precio, se mantendría el sistema a la espera de que se introdujese más dinero y se seleccionase un nuevo producto.
- Si fuesen iguales se entregaría el producto y el sistema volvería al estado inicial.
- Por último en el caso de que el dinero sea mayor al importe, se calcula en el restador la cantidad de dinero a devolver (ADEVOLVER), y el módulo de devolución envía una señal por cada moneda a devolver (V_SAL, X_SAL, XX_SAL...), tras lo cual se entregaría el producto y el sistema volvería al estado inicial.

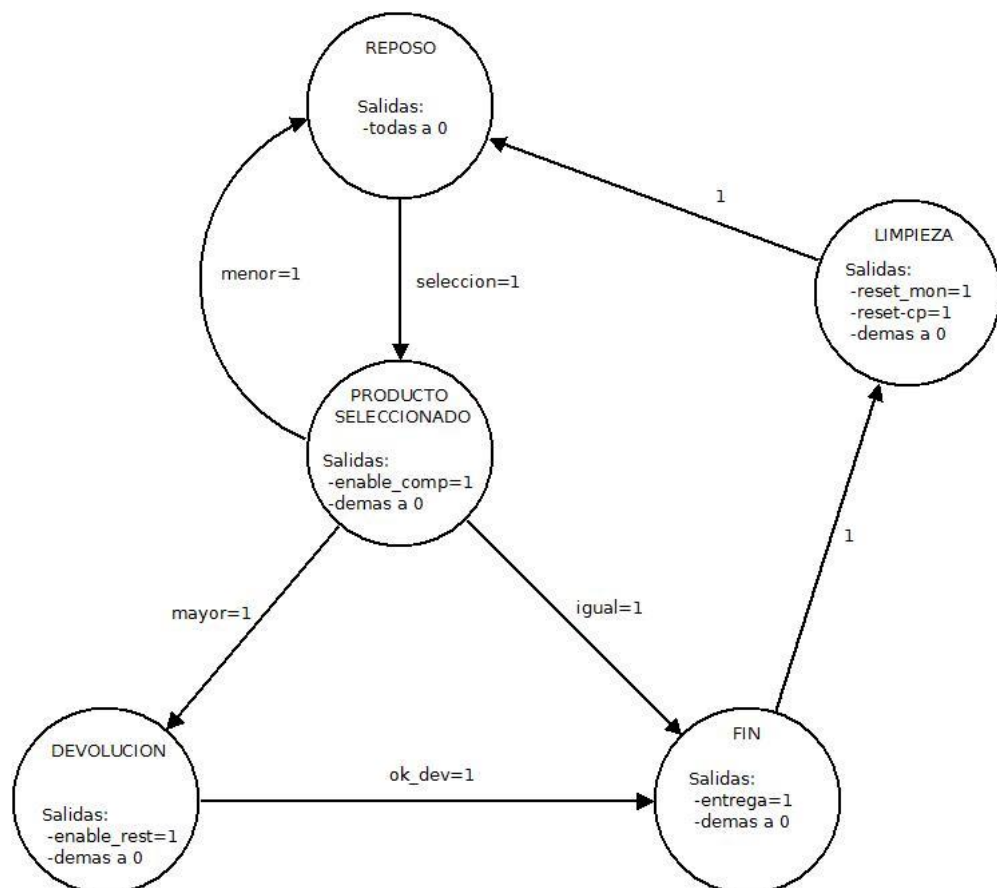
5. Esquemas y diagramas del diseño.

Diagrama de bloques: Nos indica la interconexión de los distintos componentes que se han diseñado para el proyecto, así como las señales utilizadas.



Se han obviado las conexiones del reloj, que irían a todos y cada uno de los componentes del circuito.

El circuito es gobernado por una máquina de estados tipo Moore (salida del sistema depende únicamente del estado en el que se encuentra dicho sistema). Dicha máquina de estados se representa a continuación.



- En cada estado, si se produce cualquier otra entrada a las indicadas, el sistema permanece en el estado actual.
- Todas las salidas que no estén indicadas que se activan a nivel alto, se activan a nivel bajo.

6. Explicación detallada del funcionamiento de cada uno de los bloques funcionales y de su interfaz.

6.1. Sincronizador.

El sincronizador antirrebote consiste simplemente en dos flip-flops de tipo D conectados en serie.

Lo que hace es ajustar la señal asíncrona que le entra por la patilla Async-in de acuerdo a la señal de reloj que le entra por CLK y sacar por Sync-in la señal sincronizada.

6.2. Codificador de monedas.

El codificador de monedas recibe por sus patillas V, X, XX, L, C y CC una señal de duración indefinida por cada moneda que se ha introducido en la máquina.

Por cada flanco que recibe de esta manera y de manera síncrona envía por su salida VALOR (8 bits) el mismo en céntimos de esa moneda y a la vez por su patilla ENABLE un pulso que señala que se está enviando un nuevo valor.

6.3. Sumador recursivo.

Este sumador recursivo recibe por su entrada VALOR (8 bits) cifras las cuales suma de manera síncrona a un número interno cuando la entrada ENABLE se pone a nivel alto.

Este número interno se pone a 0 cuando la patilla RESET se pone a nivel alto.
En su patilla TOTAL (8 bits) muestra en cada momento el número interno que se corresponde con el resultado de la suma.

6.4. Codificador de producto.

Este dispositivo recibe por las patillas P1, P2, P3 y DEV señales síncronas que se corresponden con los distintos productos que puede elegir el cliente.

Para cada uno de estos productos muestra en la salida PRECIO (8 bits) el importe del producto seleccionado y en la salida PRODUCTO (4 bits) el código del mismo, quedando ambas salidas enclavadas.

A su vez cuando se selecciona un producto se pone a nivel alto la patilla SELECCION, la cual se pone a nivel bajo de nuevo en cuanto se da un pulso de reloj en el que no esté seleccionado un producto.

Por último la patilla asíncrona RESET pone a cero tanto la patilla SELECCIÓN como las salidas PRECIO y PRODUCTO.

6.5. Comparador.

Este módulo muestra con los niveles lógicos de sus patillas MAYOR, MENOR e IGUAL la relación entre las entradas TOTAL (8 bits) y PRECIO (8 bits) mientras la patilla ENABLE permanezca a nivel alto, todo ello de manera síncrona con el reloj (CLK).

6.6. Restador.

El restador muestra síncronamente por su salida DEVOLVER (8 bits) la diferencia entre las entradas TOTAL (8 bits) y PRECIO (8 bits).

Su patilla OK se pone a nivel alto cuando se da un pulso de reloj y la patilla de entrada ENABLE está también a nivel alto.

Por otro lado cuando ENABLE está a nivel bajo pone a 0 la salida DEVOLVER además de a nivel bajo a la salida OK.

6.7. Devolución.

El componente devolución recibe la cantidad de dinero que tiene que devolver en monedas por la entrada DEVOLVER (8 bits) y cuando la patilla ENABLE está activa envía por las patillas V, X, XX, L, C, CC un pulso por cada moneda a devolver, por una patilla u otra en función de la moneda que se trate. Una vez ha enviado todos los pulsos pone la patilla OK a nivel alto.

6.8. Máquina.

Este componente encierra la máquina de estados que gobierna todo el sistema según el diagrama de estados mostrado anteriormente.

Las patillas SELEC, MAYOR, MENOR, IGUAL y DEV_OK se corresponden con las entradas del diagrama de estados mientras que RESET_MON, EN_COMP, EN_REST, ENTREGA y RESET_CP son las salidas del mismo.

6.9. Conversor Binario 8 bits – BCD 4 bits.

Este componente está compuesto por un vector de entradas de 8 bits, que contiene el importe total introducido hasta el momento en binario, y tres vectores de 4 bits que contienen cada una de las cifras del importe en base decimal, expresado en código BCD.

Para ello se guarda la señal de entrada en una señal “inicio”, a la cual se va a ir manipulando.

Si “inicio” contiene un valor mayor o igual a 100, se le resta 100 unidades y se suma una unidad en otra señal auxiliar “aux1” en cada ciclo de reloj, hasta que ya no cumple esa condición.

Se pasa a una segunda etapa en la que se resta a la señal “inicio” 10 unidades, de forma similar a lo anterior, pero sumando las unidades a una señal auxiliar “aux2”.

De igual manera después se pasa a una tercera etapa con las unidades de la señal “inicio”.

Este método permite pasar directamente a la etapa 2 ó 3 en caso de que la señal “inicio” no cumpla con las condiciones primeras.

Después cada señal auxiliar se conecta a cada una de las salidas del componente.

6.10. Prescaler (Divisor de frecuencia de reloj) de 50MHz a 100 Hz.

Se compone de una entrada en la que se conecta el reloj de la tarjeta, en este caso es de 50 MHz, y una salida que contiene la señal de reloj ralentizada.

El código se compone de un contador que llega hasta un número que es la frecuencia del reloj original entre la frecuencia del reloj deseado y entre 2. El contador, al llegar al final de su cuenta, invierte el valor actual de una señal auxiliar, que después se conecta a la señal de salida.

Se ha implementado con unas constantes genéricas de forma que al cambiar su valor, se cambian las frecuencias del reloj de inicio y del reloj resultado.

6.11. Decodificador BCD 4 bits – 7 segmentos para Display de Leds.

Este componente ya ha sido realizado para la práctica dos de la asignatura.

Se tiene un vector de 4 bits como entrada que representa un número decimal entre 0 y 9.

Mediante la instrucción “case <señal de entrada> is”, se asigna el valor adecuado a la señal de salida (vector de 7 bits), para encender o apagar los segmentos correspondientes del display de la tarjeta de prácticas.

Los segmentos son activos a nivel bajo.

En el apartado número 7 se dan más detalles sobre el display de 7 segmentos.

6.12. Multiplexor con cambio de señal de consigna automático por pulso de reloj, para control de los 4 displays de la FPGA de forma independiente.

Debido a que en principio no se pueden mostrar números independientes en cada uno de los cuatro displays de la tarjeta, se necesita mostrar el número requerido en cada display de forma intermitente, de uno en uno, en un ciclo de reloj cada vez.

Como la frecuencia del reloj es muy rápida para el ojo humano, se crea la ilusión de mostrar 4 números diferentes a la vez.

Para ello es necesario pasar cada una de las cuatro señales por un multiplexor, que da paso a una señal u otra según su señal de consigna.

Esta señal se debe ir cambiando para dar paso de forma cíclica cada una de las señales de entrada. Se podría conectar entonces un contador de anillo a esta entrada.

Sin embargo se ha optado por implementar en el propio multiplexor una pequeña máquina de estados, de forma que en cada ciclo de reloj se pasa al siguiente estado, en el que la señal de entrada de consigna da paso a la entrada de datos siguiente.

Como entradas de datos tenemos 4 vectores de 4 bits, correspondientes a los números codificados en BCD que indican el importe introducido y el producto seleccionado.

La salida por tanto es un vector de 4 bits.

7. Entradas/Salidas utilizadas de la placa de entrenamiento Spartan3.

En esta sección indicamos las correspondencias entre las entradas utilizadas de la tarjeta (interruptores y pulsadores) y sus correspondientes señales de entrada del sistema diseñado (introducción de monedas y selección de producto).

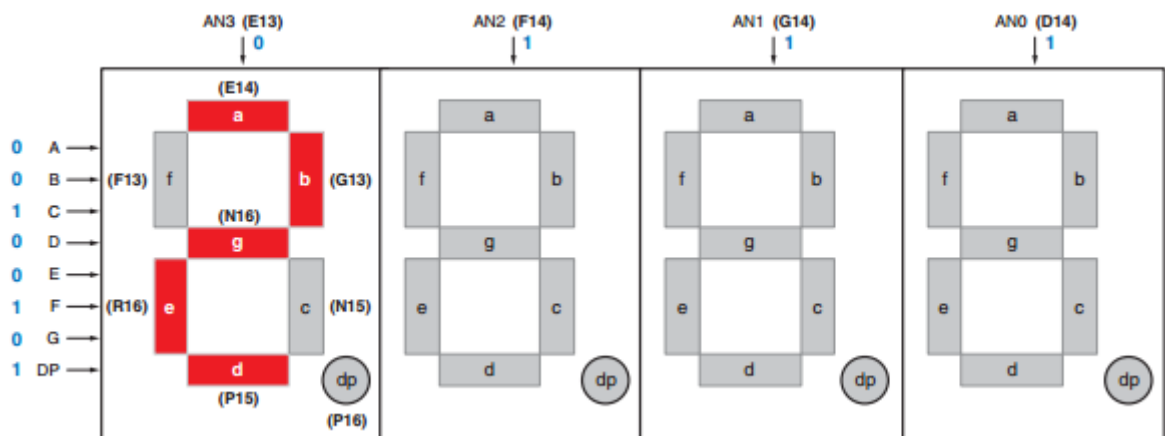
Asimismo se explica para las salidas (leds y displays), con las salidas del sistema (devolución de monedas, importe total introducido, indicador de producto seleccionado).

Entradas			
Monedas	Entrada sist.	Interruptor	Puerto FPGA
5 cent	V_as	SW-0	F12
10 cent	X_as	SW-1	G12
20 cent	XX_as	SW-2	H14
50 cent	L_as	SW-3	H13
1 e	C_as	SW-4	J14
2 e	CC_as	SW-5	J13

Entradas (continuación)			
Producto	Entrada sist.	Pulsador	Puerto FPGA
Devolución	DEV	BTN0	M13
P1	P1_as	BTN1	M14
P2	P2_as	BTN2	L13
P3	P3_as	BTN3	L14

Salidas			
Monedas	Salida sist.	Led	Puerto FPGA
5 cent	V_sal	LED-0	K12
10 cent	X_sal	LED-1	P14
20 cent	XX_sal	LED-2	L12
50 cent	L_sal	LED-3	N14
1 e	C_sal	LED-4	P13
2 e	CC_sal	LED-5	N12
Entrega pdto	Entrega	LED-7	P11

Las indicaciones del producto seleccionado y del total del importe introducido se realizan con los displays de la tarjeta. El del extremo derecho indica el producto, y los otros tres el importe. Las siguientes imágenes muestran los puertos de los elementos citados.



El vector de salida de bits del sistema llamado “digctrl” se encarga de encender los segmentos correspondientes para formar los números. El componente de menor peso corresponde al segmento ‘a’ (puerto E14). Se sigue el orden de las letras y el segmento de mayor peso es el correspondiente al punto decimal (P16).

El vector de salida de bits del sistema llamado “digselect” se encarga de seleccionar cuál display o displays se encienden.

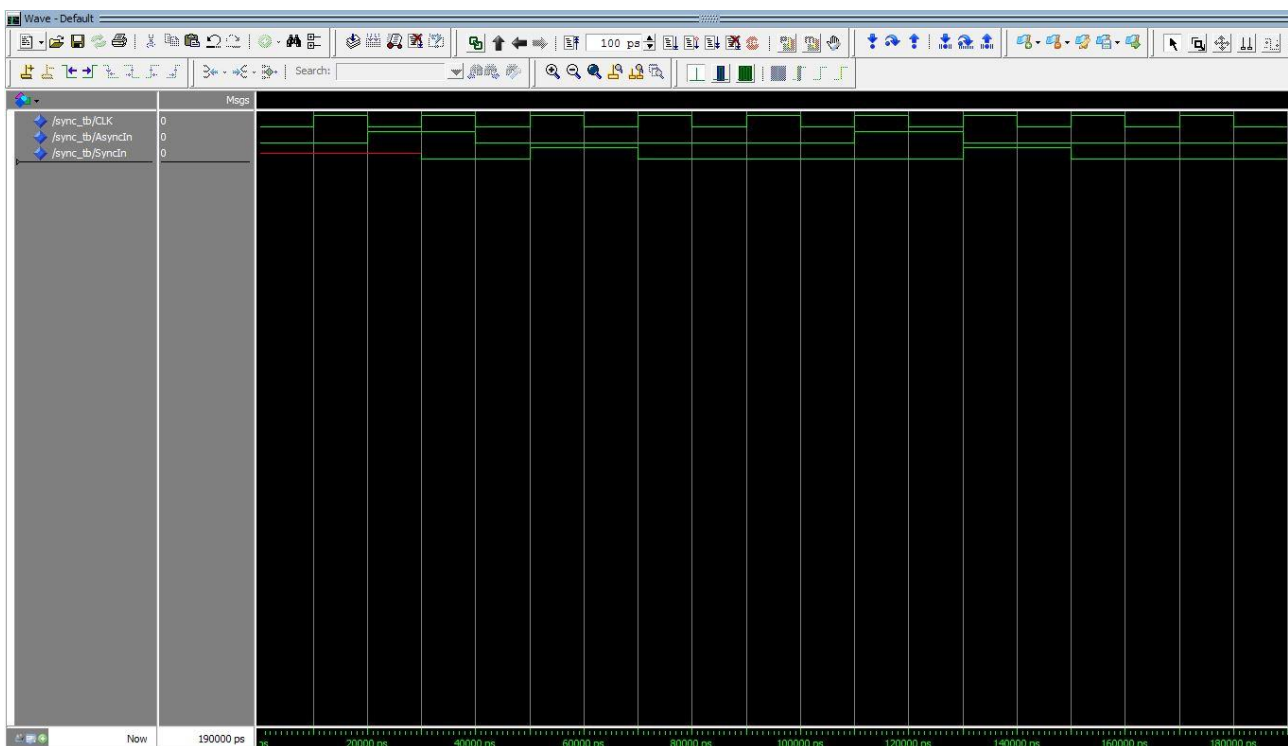
En la imagen, los puertos de la placa son los indicados entre paréntesis.

8. Testbenches realizados.

8.1. Notas generales:

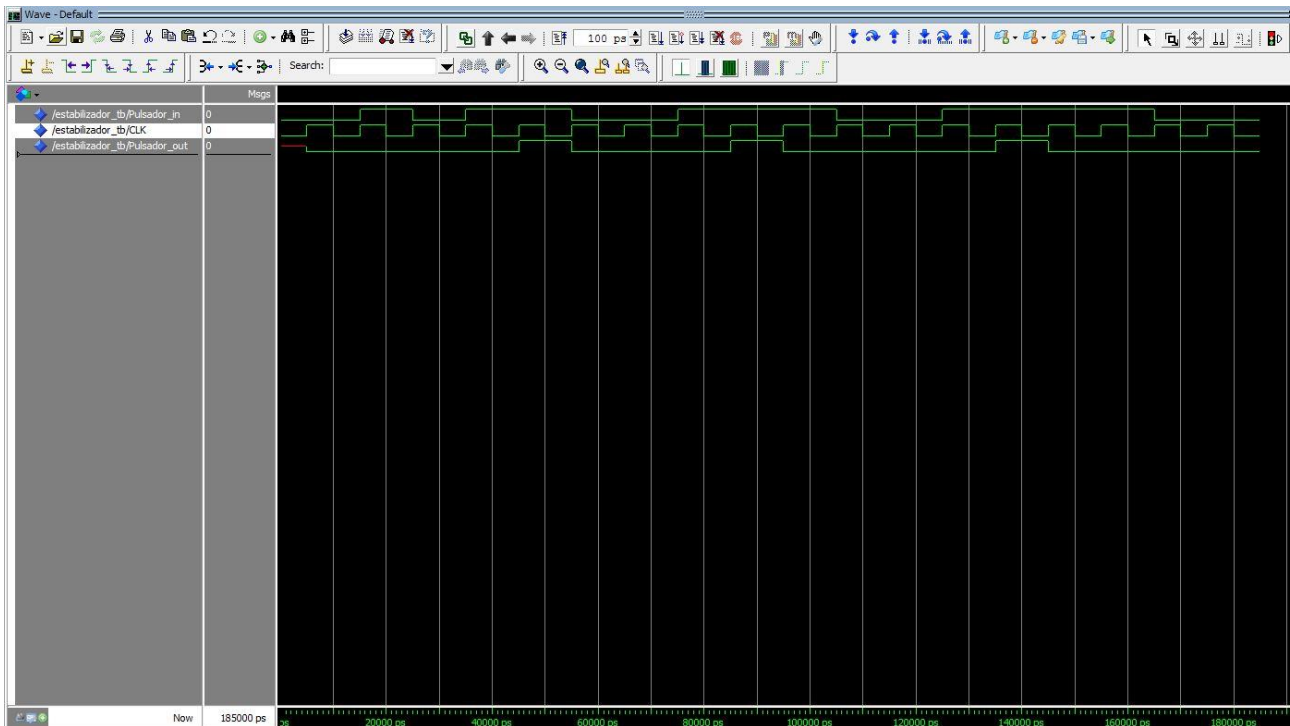
- Los testbenches de cada componente están hechos para comprobar su uso independientemente, pero teniendo en cuenta cómo serán las entradas de los anteriores componentes con los que irán conectados, es decir, semi-adaptados al uso particular de este diseño completo.
- Los decodificadores están hechos tal que además de comprobarse con las gráficas, disponen de tabla de verdad para comprobar automáticamente sus salidas.
- Todos tienen periodo de reloj flexible, es decir, permite cambiar el periodo de reloj sin modificar su código/funcionamiento.
- Excepto el top, se ha simulado con un periodo de reloj de 20 ns para una clara visualización.

8.2. Sincronizador: SYNC_tb.vhd



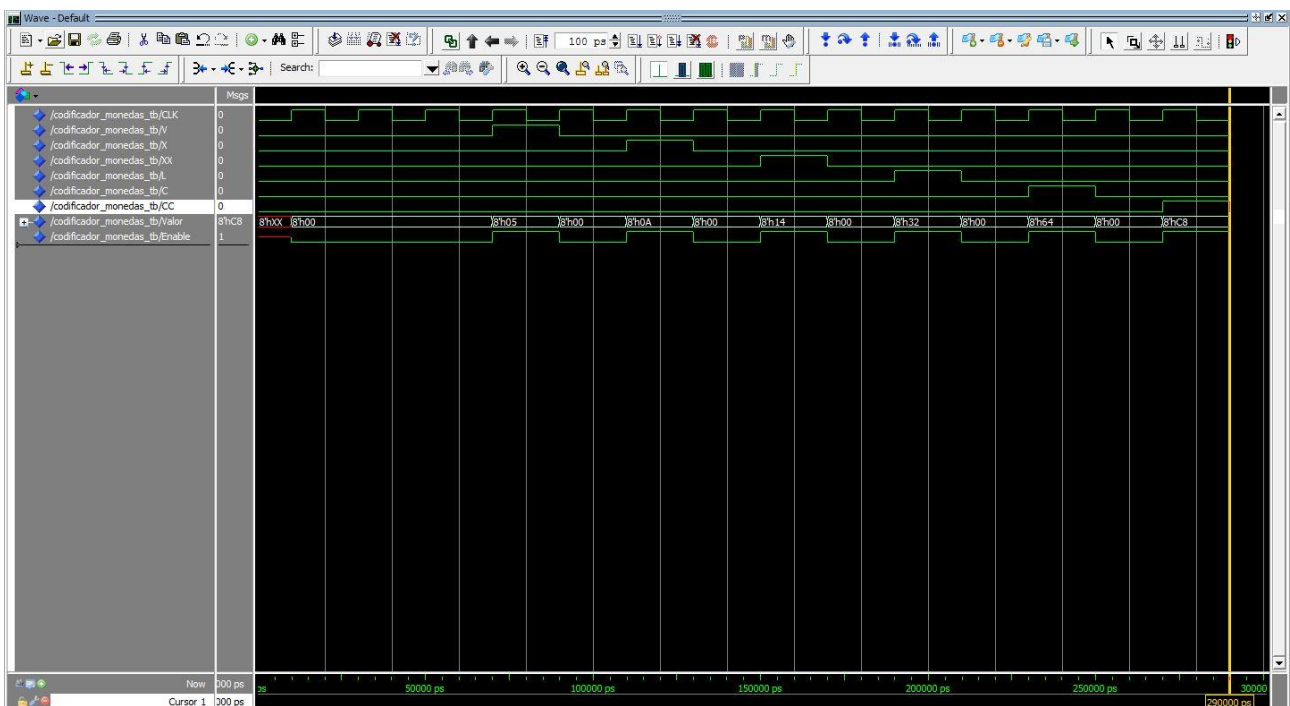
Se comprueba cómo sincroniza la entrada AsyncIn con el reloj CLK siendo la salida SyncIn.

8.3. Estabilizador (Antirrebote) : Estabilizador_tb.vhd



Como se observa, se va estimulando con entradas de duración desde un ciclo de reloj hasta varios, y vemos que ha de durar más de 2 ciclos por su funcionamiento ya que ha de comprobar el estado anterior (ver descripción del componente) y por tanto la salida tiene un ciclo de retraso, pero cumple el objetivo que es impedir que haya varias salidas cuando solo ha habido una entrada por mucho que dure.

8.4. Codificador de Monedas: Codificador_Monedas_tb.vhd



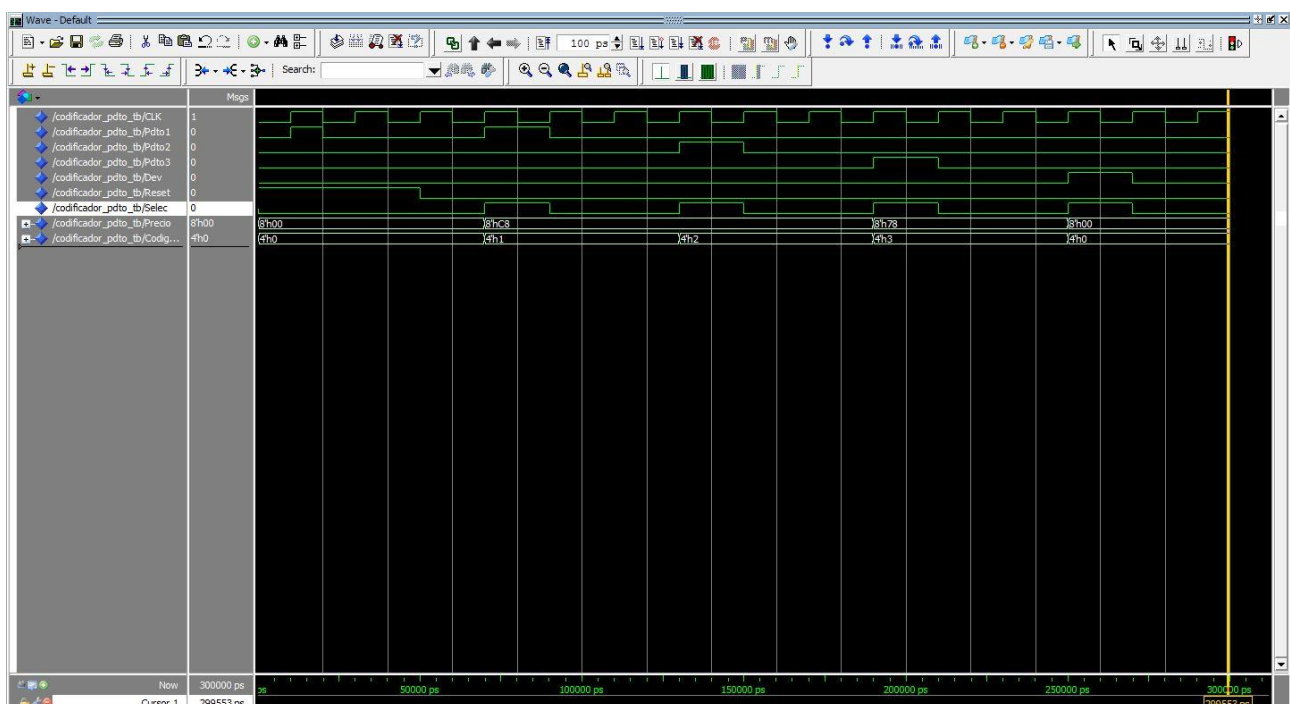
Como se puede ver en el código de este testbench y como se mencionó anteriormente, se ha simulado con una tabla de verdad donde están todos los valores de entrada/salida con la cual se han ido comprobando de manera que si alguna salida hubiera sido errónea hubiera saltado por pantalla: "Salida Valor incorrecto."

En caso de error en el valor del código o "Salida Enable incorrecto", en caso de salida enable. Como no es el caso, el Assert programado nos informa:

```
# ** Failure: Simulación finalizada. Test superado.  
# Time: 290 ns Iteration: 0 Process: /codificador_monedas_tb/stim_proc File:  
Codificador_Monedas_tb.vhd
```

El motivo de hacerlo de esta forma es agilizar el proceso pues se podrían comprobar todas las salidas una por una o por lo que muestran las gráficas, pero debido al gran número de ellas y a la necesidad de conocer el código del programador no es un método óptimo.

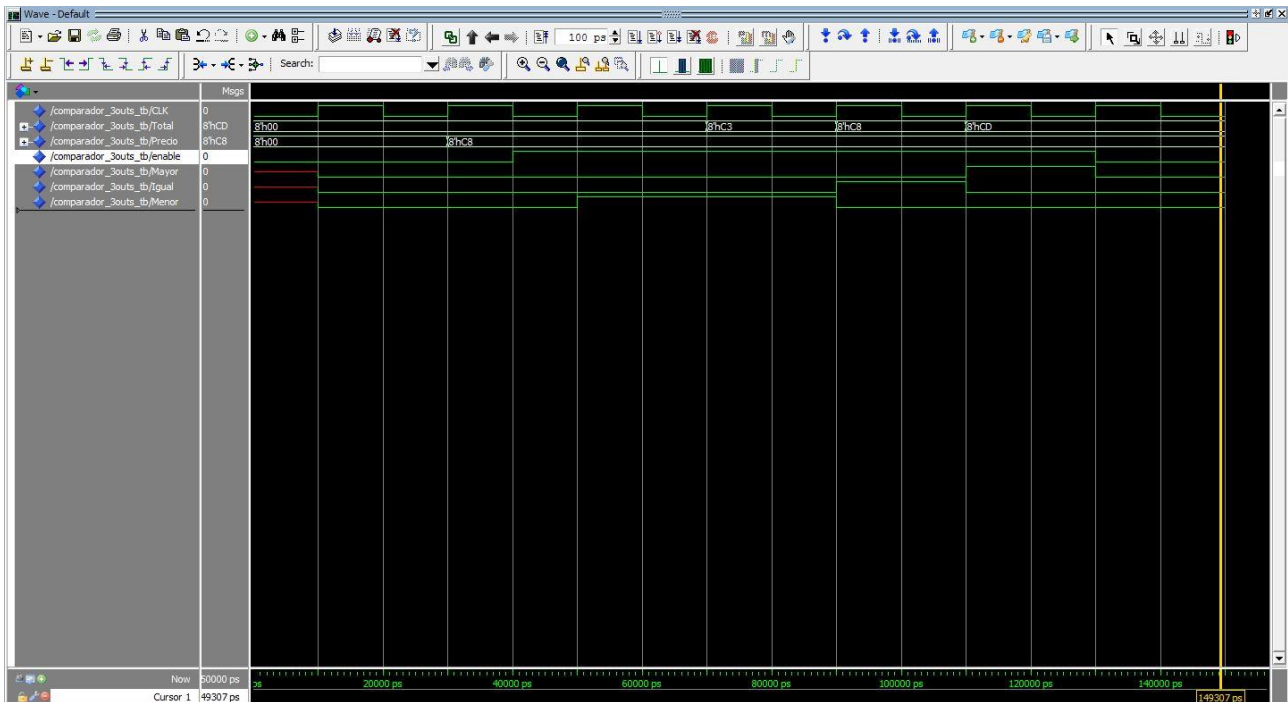
8.5. Codificador de Producto: Codificador_pdto_tb.vhd



Todo lo dicho para el codificador anterior es análogo para este.

Este componente incluye "reset" que se comprueba por gráficas, vemos que cuando está activo, las salidas son 0 aunque haya un producto seleccionado.

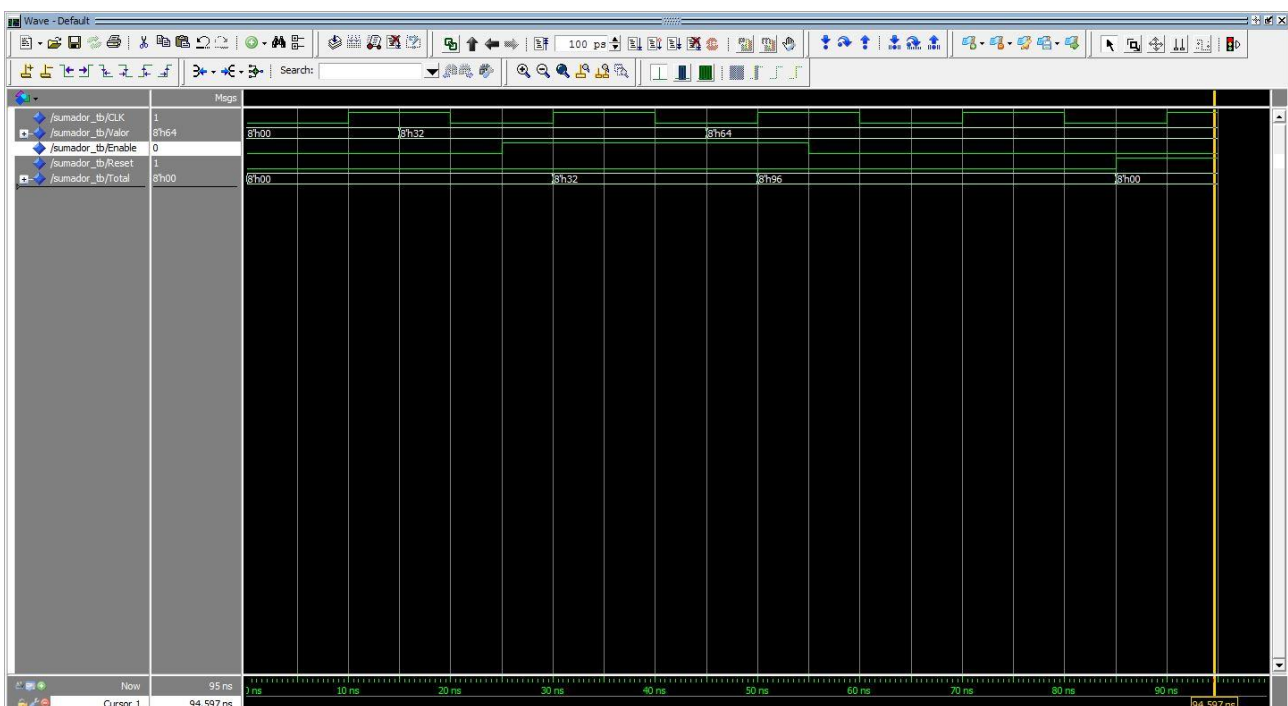
8.6. Comparador de tres salidas: Comparador_3outs_tb.vhd



Aquí se ha asignado “precio” a 200 y se ha ido comparando con “total” igual a 195, 200 y 205, vemos como la salida va cambiando correctamente.

Comprobamos al final cómo al desactivarse “enable” las salidas se ponen a '0' dejando de funcionar.

8.7. Sumador: Sumador_tb.vhd



Para comprobar el sumador se ha puesto “valor” a 50, se ha comprobado que hasta que no se active “enable” no funciona, y entonces empieza a sumar comprobando automáticamente las salidas correctas (con Assert), se actualiza “valor” a 100 y comprueba otra vez.

Vemos que las salidas son síncronas y que se mantienen hasta que se activa “reset”, reseteándolas.

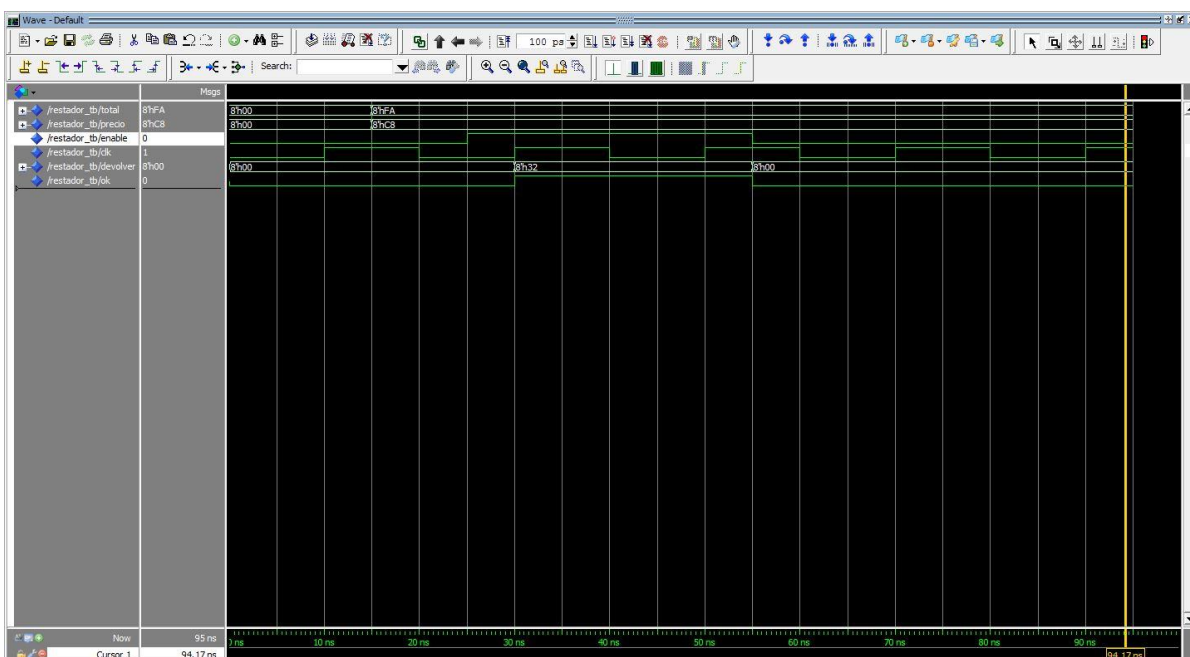
8.8. [Restador: Restador_tb.vhd](#)

Se procede de misma manera que con el sumador.

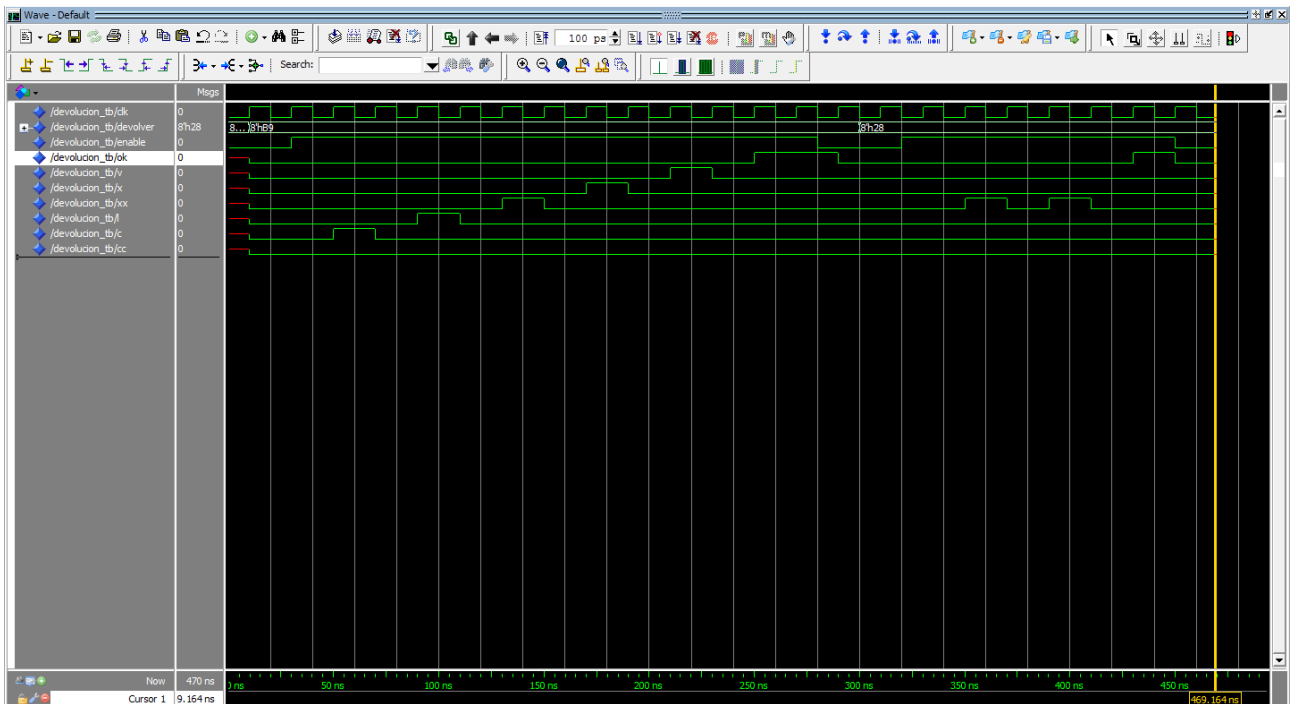
Las entradas se asignan a un valor conocido para hacer la prueba; “total” a 250 y “precio” a 200 para comprobar que “devolucion” es 50.

Aparte, se comprueban los tiempos y la salida “ok”.

Automáticamente, si alguna de estas salidas hubiera sido errónea se hubiera detenido la simulación y hubiera indicado donde se produjo el error al igual que en anteriores casos.



8.9. Devolución: devolucion_tb.vhd

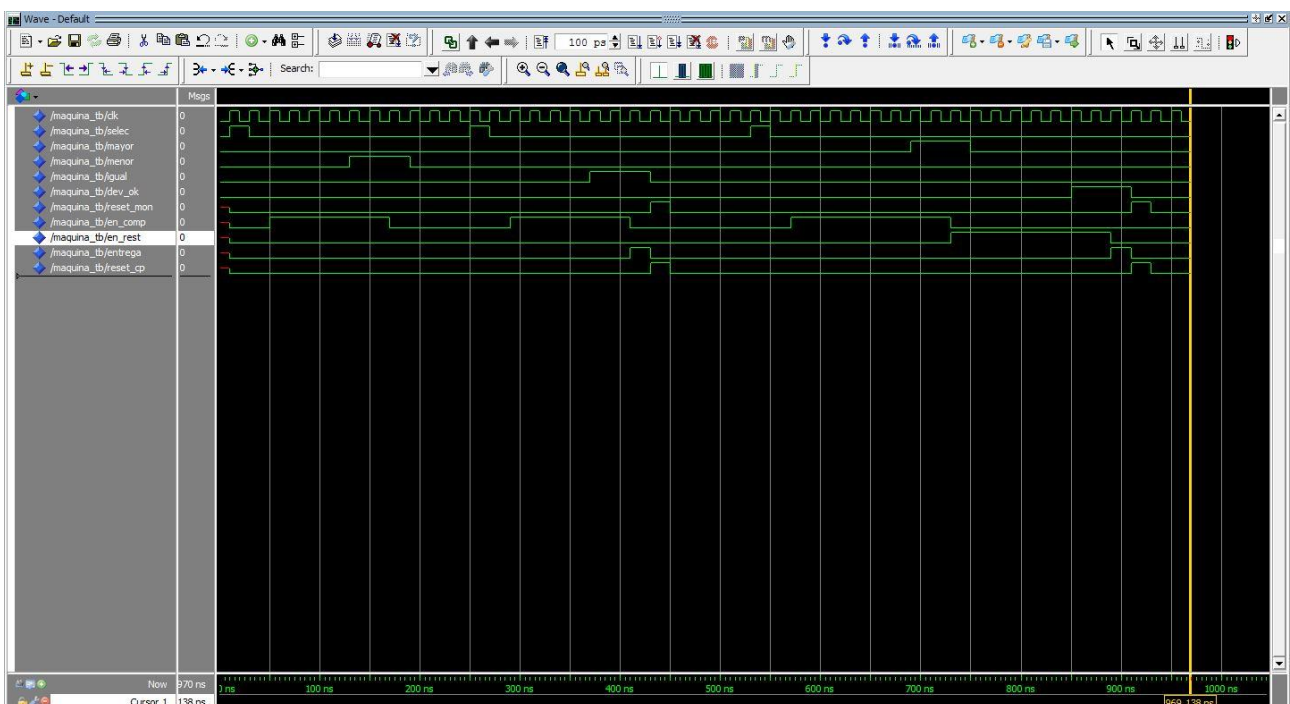


Se comprueba para la entrada “185” la salida “devolver” y se comprueban las salidas de monedas adecuadas.

Después se repite para 40, para comprobar cómo devuelve monedas del mismo tipo.

Todo ello y las salidas están comprobadas además con Assert, el “enable” lo comprobamos por la gráfica.

8.10. Máquina de estados: máquina_tb.vhd



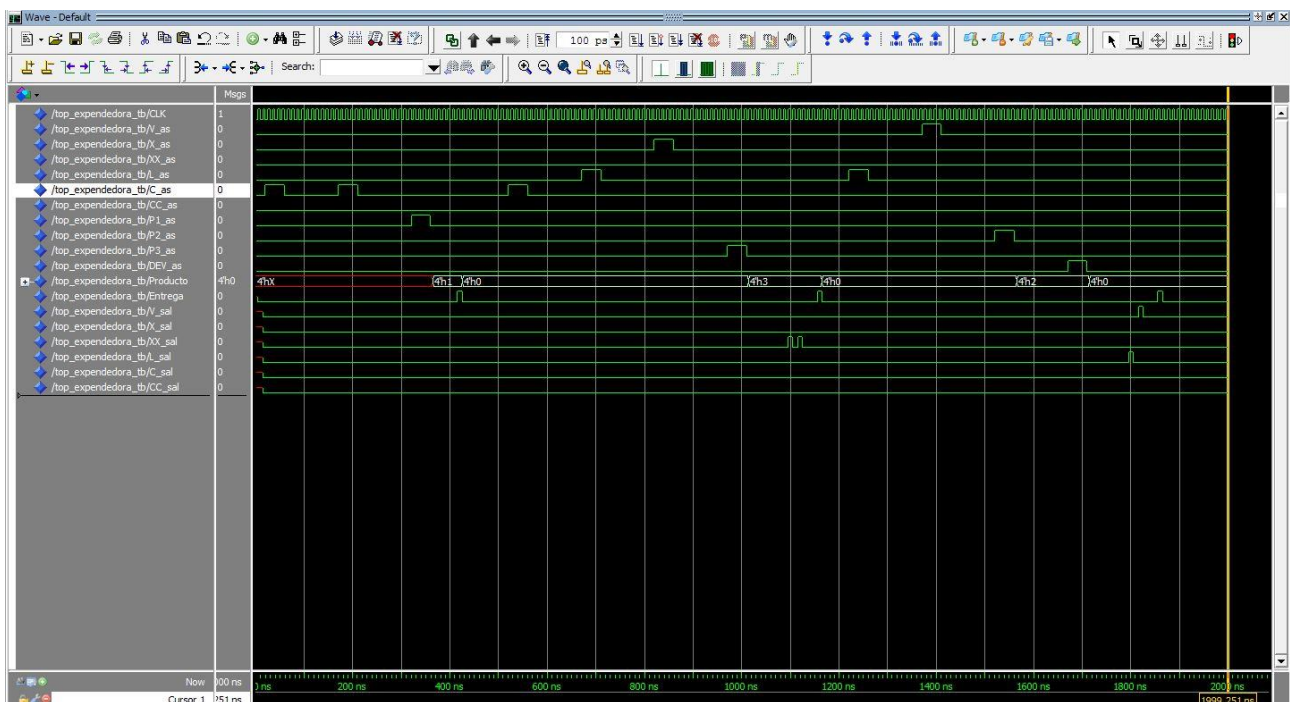
Este tb comprueba la evolución adecuada de la máquina de estados del diseño. Los estados se corresponden con las salidas que controlan otros componentes.

Se introducen las entradas y comprobamos con la gráfica; se indica que ha habido una selección con “selec”, por lo que pasa al estado comparación “en_comp” donde ya puede recibir la salida que en este caso se probamos “menor”, por lo que no se hace nada.

A continuación repetimos, pero se activa la señal del comparador “igual”. Entonces comprobamos que se activa “entrega” y se activan las salidas “reset” y se resetea (pasando al estado inicial).

Por último, repetimos y en este caso comprobamos que para “mayor” se activa después del estado de comparación el de resta “en_rest” y una vez activamos la entrada “dev_ok” ya activa la entrega y las salidas reset y se resetea (pasando al estado inicial).

8.11. Máquina Expendedora: Top_Expendedora_tb.vhd



Para el top se comprueban los casos de uso (uno a uno según corre el tb):

1. Se introducen 2 monedas de 1€ y se selecciona el producto 1, que vale ese importe exacto, por lo que comprobamos que se muestra el código de producto y activa la entrega.
2. Se introducen 1 moneda de 1€, otra de 50 cent y una última de 10 cent, lo que suma 1,60 € y selecciona el producto 3 que vale 1,20€. Comprobamos que muestra la salida adecuada y devuelve los 40 céntimos correctamente, vemos que realiza la devolución de varias monedas del mismo tipo (2 de 20 cent) y una vez hecho puede entregar el producto.

3. Se introducen monedas (por ejemplo 50 cent y 5 cent) y selecciona el producto 2, pero como no es suficiente decide pulsar devolución, por lo que se activan las salidas de las monedas correspondientes y se activa entrega por el funcionamiento de la máquina, pero como el producto es 0 al pulsar devolución, no entrega nada.