Sistemas Electrónicos Digitales

Máquina Expendedora

2023/2024 **Trabajo VHDL**

Fernando Moreno Santacruz ----- 56000

David Pinto Llorente ----- 56033

Miguel Ángel Pascual Collar ----- 55584

Contenidos:

Introducción

El trabajo a desarrollar por este equipo consiste en una máquina expendedora de refrescos. El usuario podrá introducir monedas de 10, 20, 50 o 100 céntimos y la maquina devolverá el dinero o permitirá seleccionar productos de diferente valor. El producto solo se entregará si se introduce el importe exacto, en el caso contrario, la maquina devolverá el dinero o esperará a que el usuario introduzca más monedas.

Para el producto 1, el importe a introducir serán 100 céntimos, para el producto 2: 120 céntimos, para el producto 3: 150 céntimos y para el producto 4 serán 200 céntimos.

La maquina cuenta con un temporizador de inactividad, si no se toca ninguna entrada de la máquina, esta detectara que hay inactividad y pasara a un estado de reposo. En caso de haber introducido dinero, la maquina devolverá el dinero.

Una vez se detecta actividad, la maquina dejara escoger uno de los 4 productos mediante los interruptores disponibles en la placa. Hecho esto, esperara a que el usuario introduzca el dinero justo. Finalmente, si se han seguido los pasos correctamente, a forma de señalizado de entrega, se encenderá un LED en la placa.

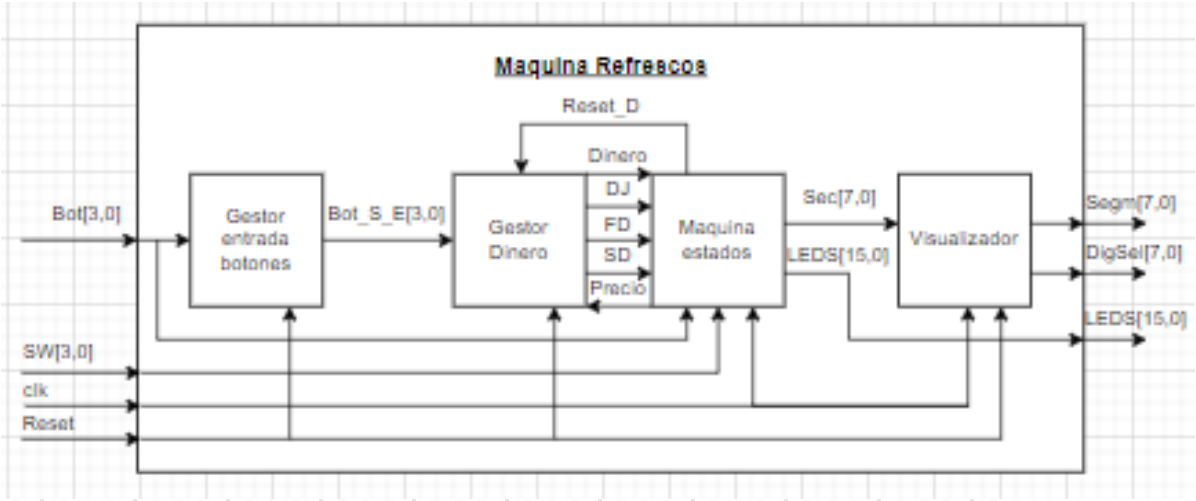
Tiempos:

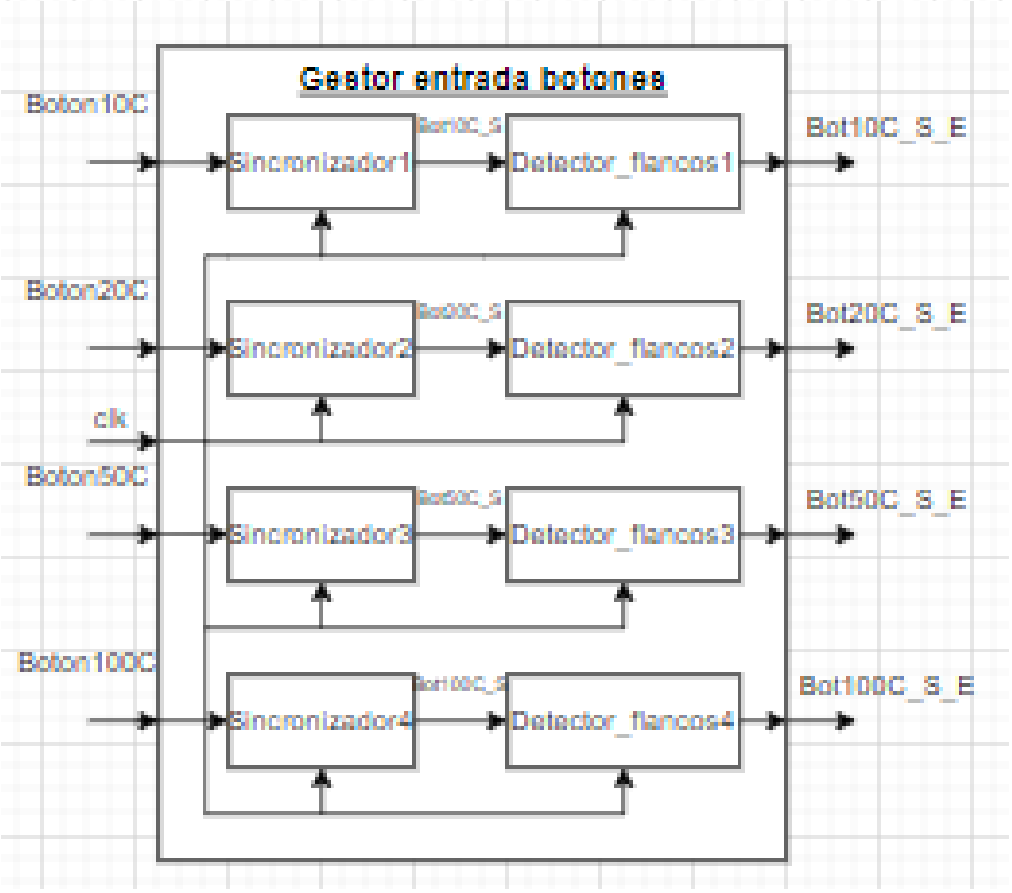
	Tiempo de inactividad	Tiempo de error de dinero	Tiempo entrega de producto
Tiempos (s)	30	2	5

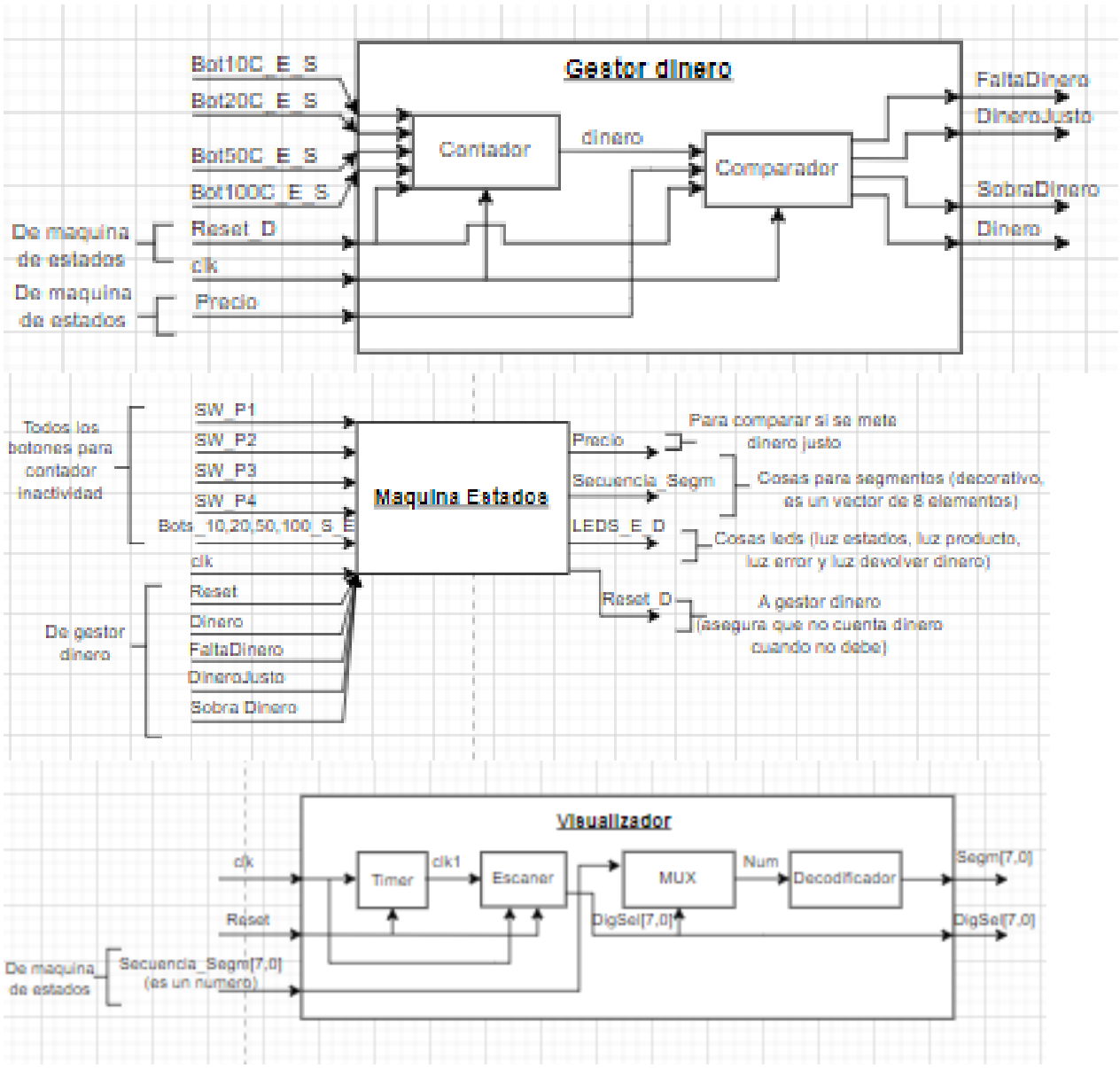
Para las simulaciones, se han utilizado tiempos muy pequeños, ya que no tiene sentido simular los 30 segundos, por ejemplo, y lleva mucho tiempo obtener los resultados de dicha simulación.

Las acciones están simuladas por LEDS, y como ayuda interactiva se hace uso de los displays de 7 segmentos. La devolución de dinero se hace con un LED, el estado actual de la maquina se hace con unos LEDS, incluso el RESET o la inactividad usan LEDS.

Diagrama de componentes






Gestor dinero
Bot10C_E_S
Bot20C_E_S
Bot50C_E_S
Bot100C_E_S
De maquina de estados
Reset_D
clk
De maquina de estados
Precio
Contador
dinero
Comparador
FaltaDinero
DineroJusto
SobraDinero
Dinero
Todos los botones para contador inactividad
SW_P1
SW_P2
SW_P3
SW_P4
Bots_10,20,50,100_S_E
clk
De gestor dinero
Reset
Dinero
FaltaDinero
DineroJusto
Sobra Dinero
Maquina Estados
Precio
Para comparar si se mete dinero justo
Secuencia_Segm
Cosas para segmentos (decorativo, es un vector de 8 elementos)
LEDS_E_D
Cosas leds (luz estados, luz producto, luz error y luz devolver dinero)
Reset_D
A gestor dinero (asegura que no cuenta dinero cuando no debe)
Visualizador
clk
Timer
clk1
Escaner
MUX
Num
Decodificador
Segm[7,0]
Reset
DigSel[7,0]
DigSel[7,0]
De maquina de estados
Secuencia_Segm[7,0] (es un numero)

Gestor entrada botones

Como entrada tiene los 4 botones de monedas y el reloj de la placa, sus salidas son los 4 botones, pero sincronizados con el reloj de la placa y tras pasar por un detector de flancos. Todas las señales de entrada y salida son del tipo std_logic.

La maquina tiene instanciado 4 sincronizadores y 4 detectores de flancos, uno para cada botón.

Los sincronizadores tienen un “filtro anti-rebotes”, este baja la frecuencia a la que la salida del botón se actualiza, de forma que se evitan los rebotes instantes después de un cambio en el botón.

Sincronizador

Las entradas de este componente son un botón y el reloj de la placa, su salida es el botón sincronizado. Se actualiza la salida del sincronizador cada cierto número de ciclos, definido por la constante SIZE de tipo entero y valor 10000.

El componente va contando ciclos de reloj y cuando la cuenta se pone a SIZE-1, se actualiza el registro de entradas concatenándolo con la entrada asíncrona y reinicia el contador. Por último, en cada ciclo, se asigna a la salida sincronizada el elemento 1 del registro de entradas.

Detector de flancos

Como entradas tiene la entrada sincronizada proveniente del sincronizador y el reloj de la placa, como salidas tiene el botón con flanco detectado. Todas las señales son de tipo std_logic.

El funcionamiento de este componente se basa en que cuando en un registro auxiliar, que es un vector de tipo std_logic de 3 elementos, se obtiene la combinación “100”, se asigna a la salida de flanco un 1 lógico y en el resto de los casos un 0 lógico. Cada ciclo de reloj se actualiza el registro auxiliar.

Gestor dinero

Este componente se encarga de actualizar y tratar las entradas o variables relacionadas con el dinero. Como entradas tiene los 4 botones sincronizados y pasados por detector de flancos, así como el reloj de la placa, un precio de tipo entero, un reset de la placa y un reset de dinero. Como salidas tenemos el dinero contado de tipo entero y si sobra, falta o hay el dinero justo. Todo menos precio y dinero son de tipo std_logic.

El gestor de dinero aumentara el dinero actual dependiendo de que botón se haya pulsado, actualizando el dinero actual tras un ciclo de reloj. Tiene instanciado un contador y un comparador para ejecutar dichos objetivos de forma satisfactoria.

Contador

Sus entradas son los 4 botones tratados, el reset de la placa, el reloj de la placa y el reset de dinero. Su única salida es el dinero contado. Todo de tipo std_logic salvo por el dinero actual.

El funcionamiento del componente es el siguiente: Cuando hay un solo botón pulsado, se suma el valor asignado a ese botón a la cuenta de dinero, cuenta es un registro auxiliar, que mas tarde es pasado a la salida. Si se pulsa más de un botón a la vez no se suma nada. Esto se asimila a la realidad, ya que no puedes meter dos monedas a la vez.

Si se pulsa el botón de reset de la placa, o se recibe que se debe de hacer un reset del dinero, la cuenta se pondrá a 0 y por ello, la salida también. De forma que el dinero actual seria 0.

Comparador

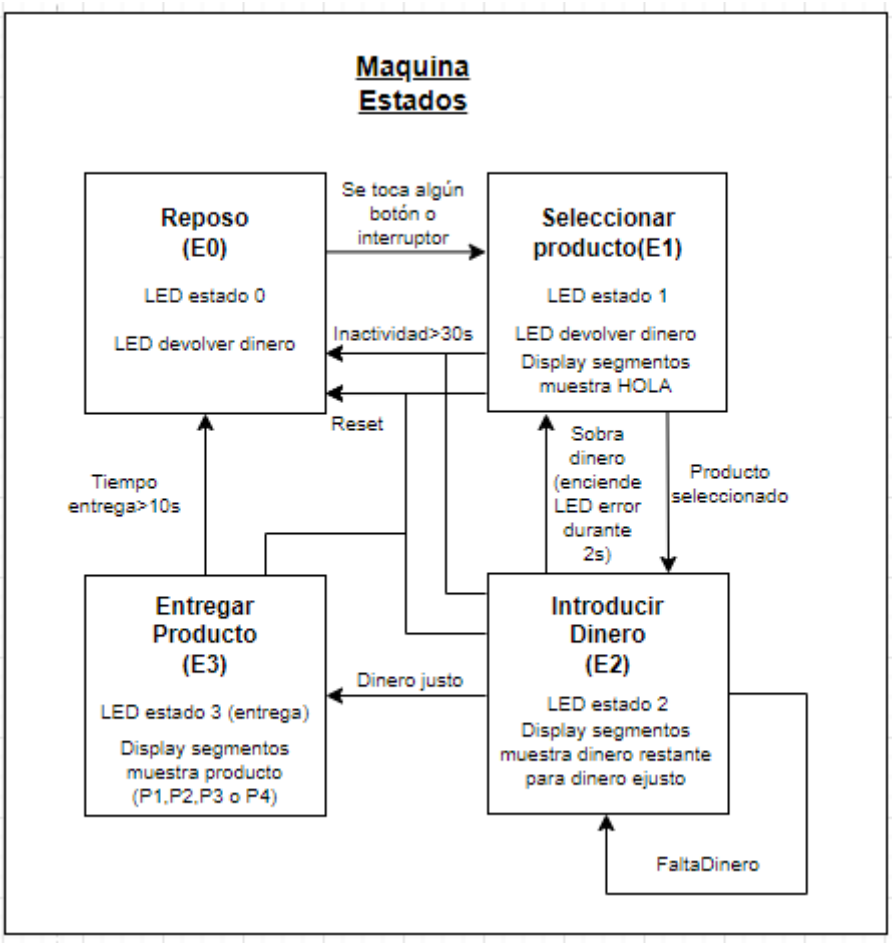
El comparador tiene como entradas el dinero actual, recibido del contador, el precio de tipo entero, el reloj de la placa, y los reset y reset de dinero. Como salidas, tenemos si falta de dinero, si se ha introducido el dinero justo o si sobra dinero. Todo de tipo std_logic salvo precio y dinero actual.

Se utiliza una señal de resultado, que es el precio – dinero actual, para evaluar la evolución de la máquina. Si esta variable vale 0, significaría que el dinero introducido esta justo, si da un numero negativo sobraría dinero y finalmente, si es mayor que 0 faltaría dinero. Dependiendo de su valor las tres salidas serán o 0 o 1 lógico.

Si se pulsa el botón de reset o se hace un reset del dinero, las tres salidas se pondrán a 0 lógico y el resultado se pondrá al valor de precio.

Maquina Estados

La maquina de estados sigue el siguiente diagrama de evolución o GRAFCET (nivel 1):



Consta de 4 estados:

1. Estado de reposo: En este estado, la maquina solo cambiara de estado si se detecta una actividad en la misma (tocar alguna entrada). No se ejecuta ninguna acción.

2. Selección de producto: En este estado, solo se transaccionará al siguiente si se acciona un solo interruptor, si se accionan varios la maquina no hace nada. Adicionalmente, si no se detecta actividad en 30 segundos, la maquina volverá al estado de reposo. Cuando se abandona este estado, dependiendo del interruptor accionado, se asigna un valor a la variable de tipo entero precio. Mientras se esta en este estado se podrá observar en los dos displays de la derecha el tiempo de inactividad contado.

3. Introducción de dinero: En este estado hay muchos caminos de transición. Todo el rato se evaluará la inactividad, de forma que si se exceden los 30 segundos se volvería a reposo. Por otro lado, se evalúa el dinero restante para alcanzar el importe del producto, de forma que si falta dinero se mantiene en este estado, si sobra dinero se activa un error y se vuelve a selección de producto, y si se introduce el importe exacto, se pasa al estado siguiente de entrega de producto. Durante este estado, en los displays de 7 segmentos encontraremos a la izquierda de los displays el tiempo de inactividad contado, así como en la derecha el dinero restante para alcanzar el importe del producto.

4. Entrega de producto: En este estado se ejecutarían las acciones de la maquina real, pero, a falta de actuadores en la placa, se simboliza con la activación de un LED y diferentes combinaciones de luces en los displays de 7 segmentos. Tras 5 segundos de alcanzar este estado, se pasa al estado de reposo, asumiendo que el cliente ha recibido su producto de forma satisfactoria.

Tiene como entradas los 4 botones tratados, los 4 interruptores, el reloj y el reset de la placa, el dinero actual de tipo entero y los 3 indicadores sobre si sobra, falta o hay dinero justo. Como salidas tenemos el precio, un vector de enteros con los valores a decodificador por el visualizar, un vector de tipo std_logic que indica los LEDs a activar y un reset de dinero.

El reset de dinero se activa en todos los estados menos en el de conteo de dinero. Esto se hace ya que la maquina de estados devuelve el dinero en todos los estados menos en ese. El precio se asigna dependiendo del producto seleccionado con los interruptores. Los botones y demás entradas las recibe para poder actualizar el temporizador de inactividad. El vector de secuencia de segmentos, dependiendo del estado en el que se encuentre la máquina, muestra diferentes informaciones. Normalmente, los displays de la izquierda muestran el tiempo de inactividad transcurrido, y los de la derecha muestran el dinero restante para importe justo, o el producto comprado.

Para la gestión de cada una de las tareas, hay un process especializado en tratar dicha tarea. Esto ayuda a organizar y tener el código más limpio.

La maquina de estados utiliza dos relojes reducidos. No confundir con el timer del visualizador, ya que ese es mas un temporizador que un reductor de reloj. Estos relojes reducidos aseguran que la cuenta de inactividad sea en segundos, así como que la cuenta de error de dinero también se haga en segundos.

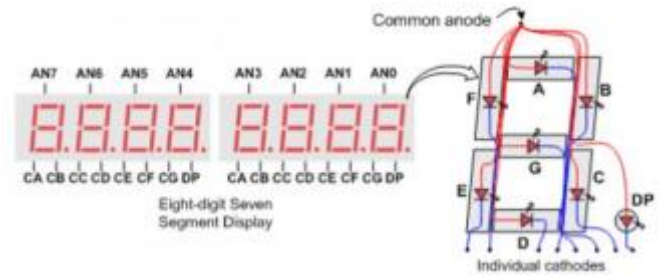
Visualizador

Por último, el visualizador es la entidad encargada de gestionar los segmentos de los displays, así como de seleccionar en cada instante un display y poner en el los elementos deseados.

Como entradas tiene el reloj de la placa, el reset de la placa y un vector de enteros que es la información por mostrar antes de ser decodificada. Como salidas tiene un vector de tipo std_logic para los segmentos y otro vector de tipo std_logic para la selección de display.

Para realizar las tareas pertinentes, se necesita instanciar un timer, un generador de secuencias o escaner, un MUX (multiplexor) y un decodificador.

Cabe destacar que se ha tenido que trabajar en algunos casos en lógica negada por la placa a usar. También se adjunta la siguiente imagen, que mas o menos explica como funciona el display de 7 segmentos a utilizar y las señales que necesitas para poder mostrar lo que se desee con el.



Timer

La entidad timer se encarga de reducir la frecuencia del reloj para no desperdiciar tanto el procesador interno de la placa, ya que solo necesitamos unos 60 hz para ver de forma clara los segmentos. El componente tiene como entradas un divider de tipo int, que será cuanto queremos dividir el reloj inicial, el reloj de la placa y el reset de la placa. Como salidas tiene un reloj reducido. Todas son de tipo std_logic salvo por el divider o divisor.

Respecto a su funcionamiento, se hace una cuenta que, al alcanzar el valor de 0, reinicia el contador y activa durante un ciclo de reloj el reloj reducido. Mas que un reductor de reloj es un contador.

Escaner

El escaner o generador de secuencias tiene como entradas un enable (en este caso el reloj reducido), el reloj de la placa y el reset de la placa. Como salidas tiene un vector de tamaño modificable.

El funcionamiento es el siguiente: Cuando se hace un reset, se selecciona el primer elemento de un registro auxiliar. Cada vez que hay un flanco de reloj se selecciona el siguiente elemento y los demás no. Cada ciclo se pasa a la salida el contenido del registro auxiliar. En este componente se trabaja con lógica negada, ya que la placa en la que se aplica así lo requiere.

MUX

El multiplexor se encarga de pasar al elemento seleccionado la información correspondiente a ese elemento. Tiene como entradas un vector de enteros y un vector de tipo std_logic del mismo tamaño. Como salidas tiene el valor seleccionado del vector de enteros.

Funciona de la siguiente manera: Dependiendo de la combinación recibida del generador de secuencias (vector std_logic), se selecciona el elemento correspondiente del vector de enteros y el valor de ese elemento a la salida. Solo puede haber un elemento seleccionado, en caso contrario, se pasa a la salida el elemento número 0 del vector.

Decodificador

El decodificador se encarga de transformar el entero a la salida del multiplexor a segmentos. Como entradas tenemos un entero y como salidas un vector de tipo std_logic.

Dependiendo de la casuística del valor entero a la entrada, se activan unos u otros segmentos. Hemos puesto que, en casos no contemplados, se muestre en el display un -. En este componente se ha vuelto a trabajar con lógica negada ya que la placa lo precisa.

Maquina Refrescos

La maquina de refrescos, o componente TOP, es donde se instancian todos los componentes citados anteriormente. El gestor de entradas, el gestor de dinero, la maquina de estados y el visualizador. En sí, es un componente que solo se encarga de contener a los demás y comunicarlos entre ellos, puesto que no realiza ninguna otra función.

Sus entradas son 4 botones, 4 interruptores, el boton de reset y el reloj de la placa. Sus salidas son los LEDs y displays de 7 segmentos de la propia placa.

Si se quisiese implementar la maquina para un proyecto real, se tendría que asignar las entradas y salidas a algún puerto de la placa para que activase los actuadores deseados y se activasen con otras fuentes que no sean de la propia placa.

Cabe destacar que se ha probado en el laboratorio a día 20/12/2023 y el proyecto funciona como se deseaba. No hemos detectado ningún fallo al instanciar todos nuestros componentes.

Anotaciones adicionales:

En este informe no se han incluido los testbenchs respectivos. Estos se pueden encontrar en la pagina de GitHub del equipo, así como capturas de pantalla de las correspondientes simulaciones. No se ha hecho testbench de todos los componentes ya que hay unos cuantos, y en algunos casos, como el trabajo a desarrollar por cada componente era poca cosa, se ha considerado innecesario. Adicionalmente, en la página de GitHub también se pueden encontrar anotaciones sobre el código, así como comentarios y el propio código y el proyecto de vivado funcional y probado en el laboratorio.