Reporte Tarea # 2

Brandon Jiménez Campos C33972

Resumen

En este reporte se presentará la descripción estructural de un controlador de cajero automático, el cual realiza todas las funciones necesarias de un cajero, recibir una tarjeta, digitar el pin, realizar retiros, verificar que se puede entregar dinero o si no se cuenta con los fondos suficientes, además de que este controlador tiene la peculiaridad de que permite depósitos, función que muchos cajeros actualmente no contienen. Para verificar el correcto funcionamiento del diseño propuesto este pasó por varias pruebas, tales como, realizar un retiro, realizar un depósito, digitar el pin incorrecto, entre otras. Se logró poder diseñar correctamente un controlador, además de poder realizar la descripción estructural de manera que contenga solamente los componentes que se tienen en la librería *cmos_cells*, además de poder implementar esta novedad la cual permite generar depósitos, las pruebas demostraron que el controlador responde correctamente ante las situaciones planteadas para su debida prueba, por ende al responder correctamente y que con la descripción estructural se haya tenido el mismo comportamiento que la descripción conductual, se podría decir que este controlador está listo para poder implementarse en cajeros automáticos.

1. Descripción Arquitectónica

El diagrama de bloques utilizado para la creación de este diseño es el otorgado en el enunciado de la tarea al igual que la descripción de cada señal y cómo funciona. Por otro lado, el funcionamiento de la máquina de estados se encuentra en la figura 1, en la cual se detalla las condiciones para poder pasar de estados, el nombre de los estados y las salidas que se llegarán a activar en cada estado. A la hora de diseñar el controlador una de las partes que más dieron trabajo fue la lógica sobre cómo introducir los dígitos al pin ingresado, la solución a esto, fue colocarlos en la lógica secuencial utilizando asignaciones no bloqueantes, permitiendo así el poder colocar dígito por dígito. Además, cosas relevantes de este diseño es que en el código se colocó una parte de lógica combinacional, la cual trata sólamente sobre las condiciones para poder pasar de estado, mientras que por otro lado se tiene una parte de lógica secuencial, la cual lo que realiza es todo lo que se debe de realizar en cada estado. También, cada vez que se entra al estado de elegir trámite y de esperando tarjeta se realiza una limpieza en los contadores, la señal de pin incorrecto, la de advertencia, entre otras. Esto con el fin de limpiar las señales cuando ya se colocó el pin correcto ó se está esperando una nueva tarjeta.

Nombre del estado	Número asignado al estado
Esperando Tarjeta	0
Digitar número	1
Se digitaron 4 números	2
Elegir trámite	3
Digitar monto retiro	4
Digitar monto depósito	5
Actualización balance	6
Actualización balance retiro	7
Pin incorrecto 1 vez	8
Pin incorrecto 2 veces	9
Pin incorrecto 3 veces	10

Cuadro 1: Estados y su número asignado en el código de verilog.

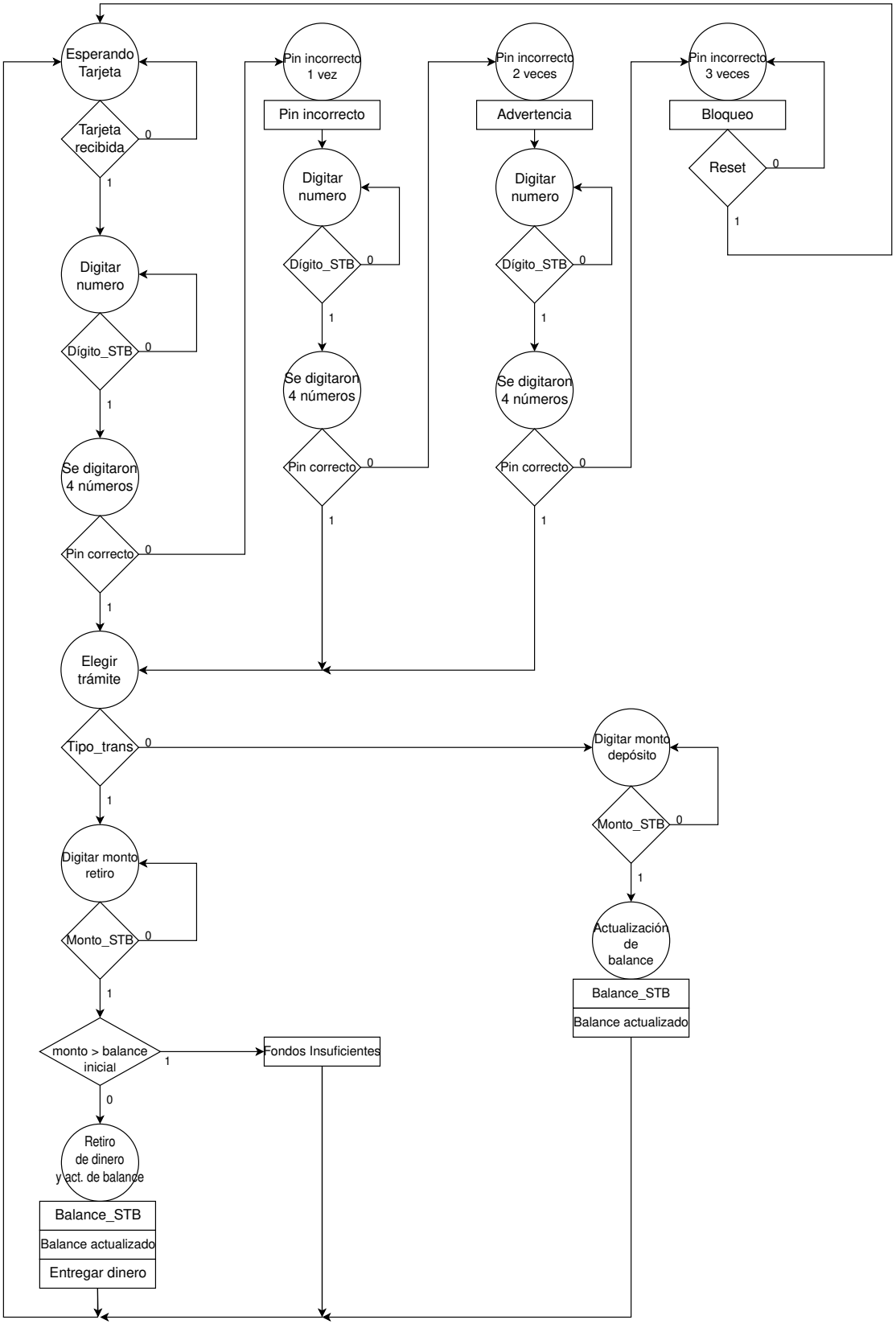

Figura 1: Diagrama ASM utilizado para el controlador del cajero automático.

Además, se colocó en la librería *cmos_cells* que las compuertas tengan cierto retardo, esto para acercarlo más a la realidad, ya que debido a los transistores se tiene tiempos de retardo, estos se encuentran presentados en el cuadro 2

Elemento	**Retardo (unidades de tiempo)**
Compuerta NAND	6
Compuerta NOR	2
Compuerta NOT	4
Flip FLop	2
Buffer	1

Cuadro 2: Retardos en las compuertas lógicas y flip flops

Con estos retardos se necesita un reloj de mínimo un ciclo de reloj de 100 unidades de tiempo para que el diseño funcione correctamente.

2. Plan de Pruebas

El controlador fue puesto bajo las siguientes pruebas con el fin de poder verificar su correcto funcionamiento ante situaciones para las cuales está diseñado, como lo es un retiro o un depósito, tanto como situaciones que podría sufrir en el día a día, como podría ser el que una persona olvide su clave y falle el pin 3 veces o también que recuerde el pin en el segundo intento. A continuación se presentan las pruebas a las cuales fue sometido el diseño, además se planteó una situación muy común, la cual es que el usuario no disponga de los fondos suficientes, entonces también se realizó la prueba para observar cómo reacciona el controlador en un caso como este.

Prueba I: Realización de un depósito básico.

Esta prueba consiste en una situación a la cual se va a enfrentar el controlador, ya que esta es una de sus funciones básicas, la cual es que el cajero recibe la tarjeta, el usuario coloca su pin correctamente, se selecciona el monto deseado para el depósito y se debe de actualizar el balance de la cuenta, esta prueba su finalidad es demostrar que el diseño cumple con lo solicitado correctamente.
Estado de la prueba: ✔ Aprobado.

Prueba II: Realización de un retiro básico.

Esta prueba consiste en una situación a la que se va a enfrentar el controlador de manera muy común, la cual es la de realizar un retiro, esta prueba consiste en que el cajero recibe una tarjeta, se ingresa el pin correcto y se selecciona el monto que se desea retirar y en este caso el monto va a ser menor que el balance inicial, logrando así que se actualice el balance y se active la salida de entregar dinero, esta prueba su finalidad es verificar que el diseño funciona correctamente y activa la señal de entregar el dinero.
Estado de la prueba: ✔ Aprobado.

Prueba III: Caso en donde la persona no tenga los fondos suficientes

Esta prueba consiste en una situación a la que muy probablemente vaya a enfrentar el controlador y esta es una prueba para verificar la correcta implementación de seguridad para que el cajero no otorgue más dinero del que contiene el usuario en su cuenta bancaria, ya que si el cajero otorga más dinero del que contiene el usuario sería un problema que perjudicaría al banco directamente.
Estado de la prueba: ✔ Aprobado.

Prueba IV: Se ingresa 3 veces un pin incorrecto

En esta prueba se verifica que el controlador sea capaz de poder actuar cuando un usuario falla el pin 3 veces, en este caso debería de bloquearse.
Estado de la prueba: ✔ Aprobado.

Prueba V: Un usuario realiza un trámite y luego llega otro usuario a realizar otro trámite

En esta prueba se verifica el funcionamiento del controlador cuando se tiene 2 trámites seguidos, esto ya que el controlador no solamente va a ser de un uso, este va a estar en un uso constante, por ende debe de funcionar correctamente ante dos trámites seguidos.
Estado de la prueba: ✔ Aprobado.

Prueba VI: Un usuario obtiene fondos insuficientes y luego otro usuario realiza un retiro

En esta prueba se verifica que el funcionamiento del controlador después de que a un usuario se le otorgó la señal de fondos insuficientes sea el correcto, este funcionamiento se verifica a través de cómo se comporta en la transacción posterior a que se activó la señal de fondos insuficientes.
Estado de la prueba: ✔ Aprobado.

Prueba VII: Un usuario bloquea el cajero y el siguiente usuario realiza un depósito.

A través de esta prueba se busca verificar el correcto funcionamiento del controlador en la transacción posterior a que se tuvo un bloqueo debido al introducir tres veces el pin incorrecto, luego de que se bloquee el controlador y se active la señal de reset para poder sacarlo de este estado de bloqueo y poder utilizarlo nuevamente se introducirá otra tarjeta en la cual se realizará un depósito.
Estado de la prueba: ✔ Aprobado.

Prueba VIII: Un usuario falla el pin 2 veces y en la tercera coloca correctamente el pin y el usuario que sigue también falla el pin 1 vez

Esta prueba consiste en que el primer usuario falla el pin 2 veces y en el tercer intento logra ingresar el pin correcto y realiza la transferencia que desea, seguido a esto el usuario que sigue también falla el pin 1 vez y en el segundo intento ya ingresa el pin correctamente y realizar el trámite deseado, esto es con el fin de poder observar que el controlador no acumule la cantidad de errores y se demuestre que cada vez que se ingresa una tarjeta el contador de fallos no arrastra números de usuarios anteriores.
Estado de la prueba: ✔ Aprobado.

3. Instrucciones de utilización de la simulación

Para poder ejecutar la simulación se presentará a continuación la manera de poder ejecutarlo desde la terminal en linux. Para esto primero se debe de descargar los archivos nombrados como controlador.v, tester.v, testbench.v, *cmos_cells.lib*, *cmos_cells.v*, controlador.ys y Makefile, esto debido a que se necesitan los 7 archivos para poder ejecutar la simulación, se facilita la colocación de comandos ya que el archivo.ys contiene todo lo necesario para la síntesis de la descripción conductual y también el colocar sólamente con las compuertas que se contienen en la biblioteca *cmos_cells.lib*, además de que se facilita un makefile el cual realizará todo, tanto realizar la

síntesis, como la compilación y abrir el gtkwave, por ende se necesita solamente utilizar el comando presentado en el listado 1

Listado 1: Colocación del comando make

```
user@ubuntu:~$ make
```

Si se desea realizar el proceso de síntesis solamente, sin la compilación ni la simulación, se utiliza el comando presentado en el listado 2, para esto es necesario tener los archivos *cmos_cells.lib cmos_cells.v controlador.v controlador.ys*

Listado 2: Colocación del comando para síntesis

```
user@ubuntu:~$ yosys -s controlador.ys
```

Este genera un archivo llamado *contr_synth.v*, en el cual es donde se encuentra la descripción estructural.

Cuando se desee eliminar los archivos generados por el make se utiliza el comando mostrado en el listado 3

Listado 3: Comando para limpieza de archivos generados por el make.

```
user@ubuntu:~$ make clear
```

4. Ejemplos de resultados

Al realizar el proceso de síntesis, se obtuvo que el diseño cuenta con la siguiente cantidad de compuertas lógicas NAND, NOR, NOT y flip flops.

Elemento	**Cantidad**
Compuerta NAND	788
Compuerta NOR	714
Compuerta NOT	234
Flip FLop	96

Cuadro 3: Cantidad de compuertas lógicas y flip flops

Para comprobar que funciona la descripción estructural las pruebas que se realizaron son las mismas que las que tuvo la descripcion conductual, solamente que esta vez con los retrasos mencionados en el cuadro 2, ya que se busca que las pruebas se asemejen a la realidad. Sin embargo, se presenta de ejemplo, la prueba de un depósito simple y un retiro con el código sintetizado, pero sin retardos, como referencia para observar que funciona sin retardos.

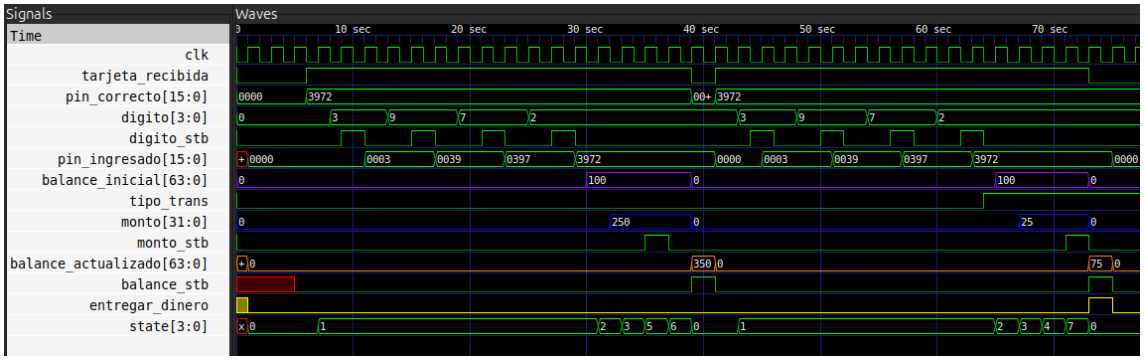

Figura 2: Prueba depósito y retiro con código sin retardos.

En la figura 2, se puede observar el cómo se comporta la descripción estructural ante 1 depósito y un retiro simples ambos, donde completa de manera correcta su función, ya sea en el depósito que actualiza el balance como en el retiro (señal naranja) y activa la salida de entregar dinero (señal amarilla).

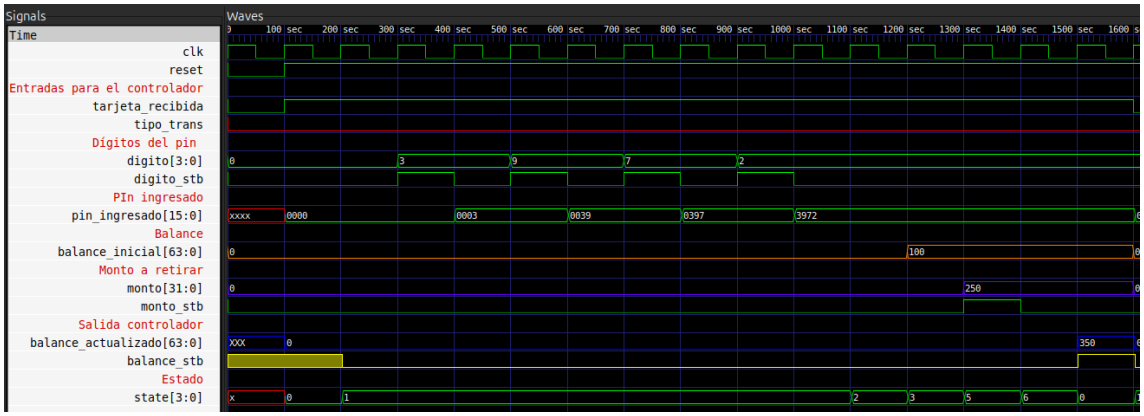

Figura 3: Prueba depósito con retardo

En la figura 3 se puede observar el cómo se realiza un depósito simple el cual ahora presenta retardos, en donde luego de ingresar el pin correcto se pasa de estado en donde se elije el tipo de transferencia y como está en 0, es un depósito, seguido a esto se digita el monto y finalmente se obtiene el balance actualizado (señal azul) y se activa el $Balance\_STB$ que es la señal amarilla.

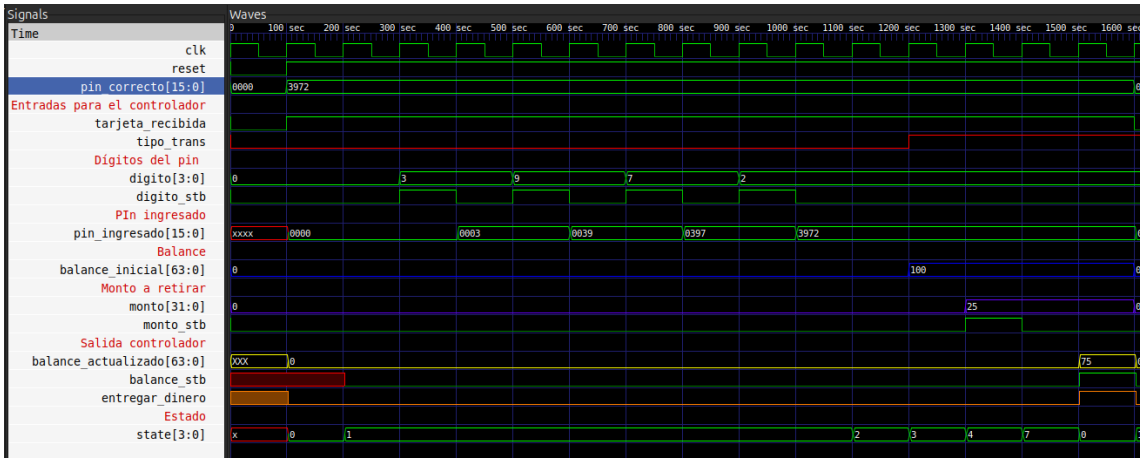

Figura 4: Prueba retiro con retardo.

En la figura 4 se puede observar el cómo se realiza un retiro, en donde luego de colocar el pin correcto y seleccionar que el tipo de transferencia sea un retiro, en este caso el monto es menor al balance inicial, por lo que sí se permite el retiro, y se puede observar en la señal la amarilla, la actualización del balance, además se observa el cómo la señal de $Balance\_STB$ también se enciende, al igual que la señal de entregar dinero también se enciende, por lo que se presenta el comportamiento esperado.

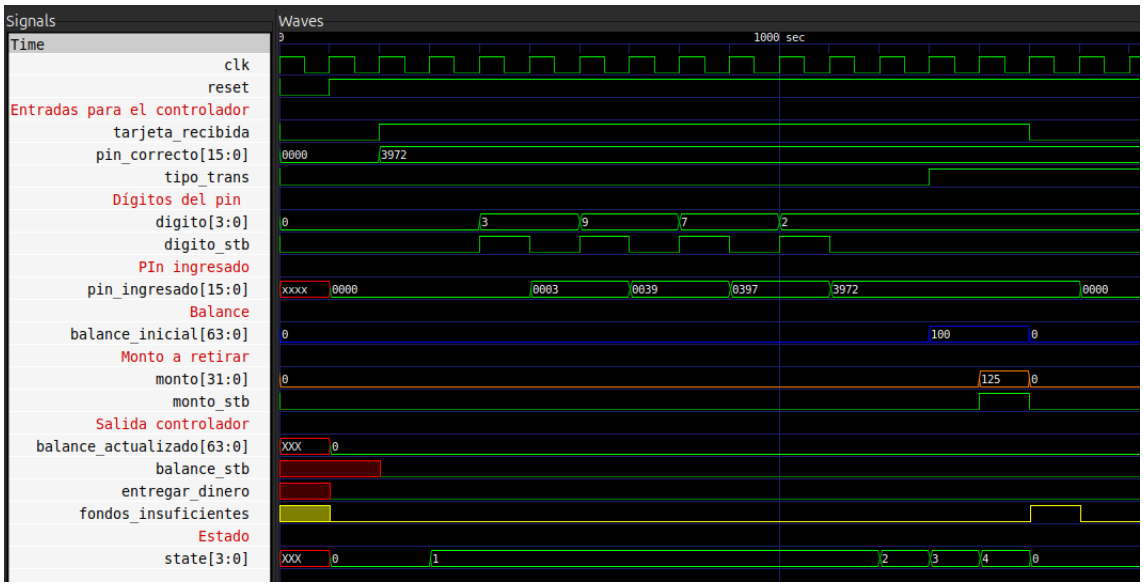

Figura 5: Prueba fondos insuficientes con retardo.

En la figura 5 se puede observar que el monto que se requiere retirar (la señal naranja) es mayor al balance inicial del usuario (la señal azul), por ende el balance no se debe de activar, sino la señal de salida que se debe de activar es la de fondos insuficientes, que efectivamente en la figura se observa como la señal amarilla, la cual es la de fondos insuficientes, es la que se enciende.

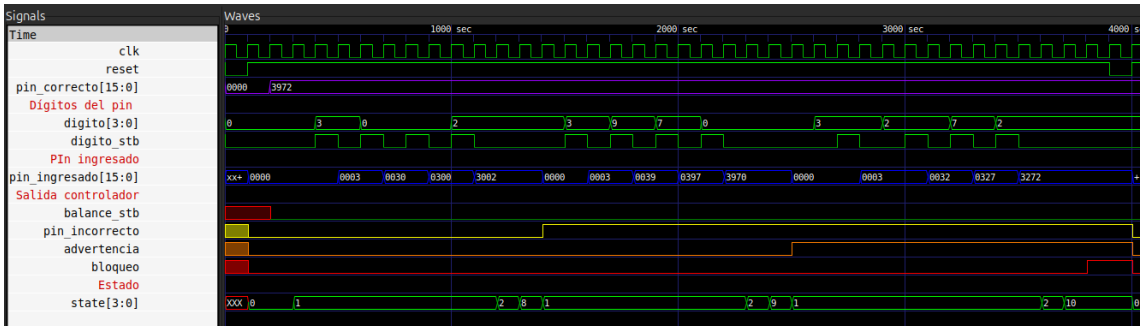

Figura 6: Prueba de pines incorrectos con retardo.

En la figura 6 se puede observar el comportamiento del controlador en descripción estructural cuando se colocan pines incorrectos, en este caso se colocaron 3 pines incorrectos, llevando al controlador a bloquearse, esto se puede observar siguiendo las señales amarilla, naranja y roja, en donde la amarilla se enciende cuando se falla el pin por primera vez, la naranja es la advertencia cuando se falla el pin 2 veces y finalmente cuando se falla el pin por tercera vez se bloquea el cajero.

Si se desea realizar más pruebas o observar otras pruebas, se invita a descargar el proyecto y a ejecutarlo con el makefile, en donde aquí se contienen más pruebas y se observa el controlador funcionando con trámites de manera continua.

5. Conclusiones y recomendaciones

A través de los resultados obtenidos en las pruebas se puede observar que realmente el diseño estructural funciona correctamente y que a pesar de los retardos, que es una característica que en la vida real van a tener debido a las capacitancias de los transistores, funciona correctamente y cumple con todas las funcionalidades que se necesitan, además de que se mantiene las funcionalidades de seguridad, como lo es el verificar los pines, y activar las señales de pin correcto, advertencia y de bloqueo, además de verificar si se cuenta con los fondos necesarios para poder realizar un retiro, para así asegurar que el banco no tenga inconvenientes, a través de los retardos, se reafirma la importancia que tiene el reloj, ya que al colocar los retardos con el reloj que se tenía para la descripción conductual, ya que debido a los retardos dejó de funcionar correctamente, hasta el punto de ni si quiera poder introducirse el pin, ya que puede que debido al retardo no se logre tener un adecuado flanco de reloj. Es por esto que esta parte se complicó un poco, por el hecho de que el agregar retardos implica modificar completamente el tester, pero lo que se aprendió es que se debe de comenzar a probar con valores de reloj que sea mayores al retardo más grande que contenga cualquier compuerta o flip flop, por ejemplo si, en los valores de retardo asignados en este diseño el mayor es el de la NAND, el cual es 6 unidades de tiempo, entonces, se debe de comenzar a probar valores de reloj mayores a 6, por ejemplo 8.

Como recomendación, se tiene que se realice el trabajo con un tiempo prudente, ya que se complica a la hora de comenzar a probar ciclos de reloj, ya que no solamente es cambiar el número de los ciclos de reloj, sino tambíen es modificar la entrada $Digito\_STB$ de cada dígito que se introduce, entonces, esto en todas las pruebas, por lo que el estar cambiando los valores y verificar si funciona se lleva bastante tiempo, además de que se pueden presentar problemas como que el código contenga latches, además de que se recomienda crear un archivo .ys, el cual podrá ayudar bastante a no colocar tantos comandos en yosys.