IE-0523 Circuitos Digitales II
II Ciclo 2023
Prof. E.Coen
Prof. Luis Li

**Tarea #3**

Máquina de estados
Diseño de un controlador para cajero automático

1. Diseñar una máquina de estados para un controlador de cajero automático de acuerdo con las especificaciones dadas en este enunciado.

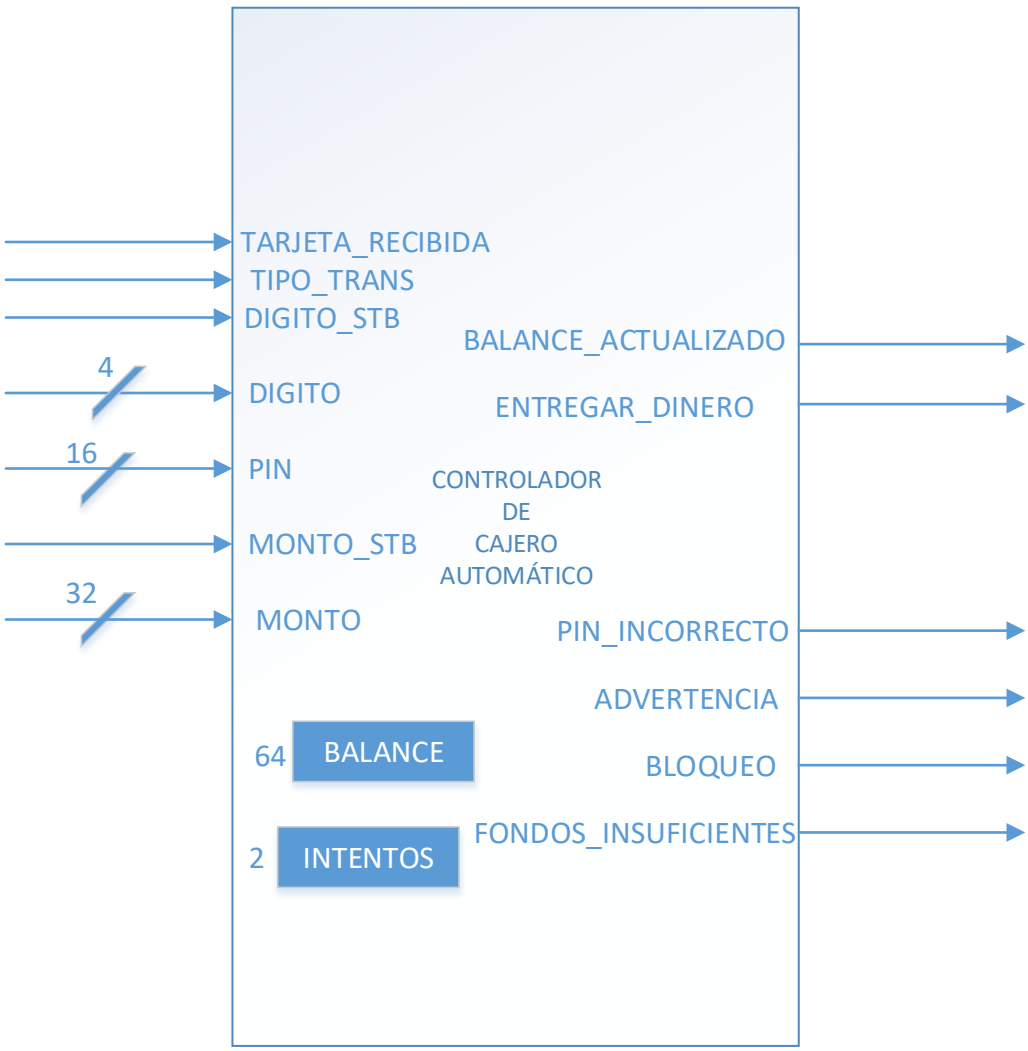


Figura #1: Controlador para cajero automático

**Interfaces del controlador:**

a) ***CLK (no se muestra en la figura) –*** Es una entrada que llega al controlador desde el CPU con una frecuencia determinada. El flanco activo de CLK es el flanco creciente.

b) ***RESET (no se muestra en la figura)*** – Entrada de reinicio del controlador. Si *RESET*=1 el generador funciona normalmente. En caso contrario, el generador vuelve a su estado inicial y todas las salidas toman el valor de cero.

c) ***TARJETA_RECIBIDA*** – Señal de entrada binaria que indica que se ha introducido una tarjeta válida.

d) ***PIN*** – Entrada de 16 bits donde cada grupo de 4 bits representa un dígito del PIN de la tarjeta que se recibió.

e) ***DIGITO*** – Entrada de 4 bits que representa el último dígito tecleado por el usuario. Mantiene su valor anterior hasta que se presiona la siguiente tecla.

f) ***DIGITO_STB*** – Señal de entrada que se pone en 1 durante un solo ciclo de reloj (STB) cuando se presiona un botón en el teclado. La lectura del PIN debe compararse con los valores de la entrada DIGITO cuando DIGITO_STB=1.

g) ***TIPO_TRANS*** – Entrada binaria que indica el tipo de transacción. Cuando TIPO_TRANS=0 se trata de un depósito, cuando TIPO_TRANS=1 se trata de un retiro.

h) ***MONTO*** – Entrada de 32 bits que representa el monto de la transacción expresado en binario.

i) ***MONTO_STB*** – Señal de entrada que se pone en 1 durante un solo ciclo de reloj (STB) cuando se ha actualizado el valor del MONTO a través del teclado.

j) ***BALANCE*** – Registro interno de 64 bits que representa el balance existente en la cuenta del usuario.

k) ***BALANCE_ACTUALIZADO*** – Salida de un bit para indicar que una transacción fue exitosa y que se actualizó el balance en la cuenta, tanto para depósitos como para retiros.

l) ***ENTREGAR_DINERO*** – Salida de un bit para indicarle al cajero que entregue el MONTO durante una transacción de retiro.

m) ***FONDOS_INSUFICIENTES*** – Señal de alerta cuando, durante un retiro, el valor de BALANCE es menor que el valor de MONTO.

n) ***PIN_INCORRECTO*** – Señal que indica que el valor del PIN recibido a través del teclado (la entrada DIGITO) es distinto del PIN de la tarjeta especificado en la entrada PIN.

o) ***ADVERTENCIA*** – Salida binaria que se enciende cuando el usuario ha introducido el PIN de forma incorrecta dos veces.

p) ***BLOQUEO*** – Salida binaria que se enciende cuando el usuario ha introducido el pin de forma incorrecta 3 veces.

**Casos de uso:**

El usuario introduce una tarjeta en el cajero automático. En ese momento la señal TARJETA_RECIBIDA toma el valor de 1 y se actualiza el valor de la entrada PIN.
Posteriormente, el usuario introduce, uno por uno, los cuatro dígitos de su número de PIN.
Cada vez que el usuario presiona un dígito, la señal DIGITO_STB se pone en 1 durante un ciclo de reloj y se actualiza el valor de la entrada DIGITO.
Una vez introducidos los 4 dígitos del número de PIN, el resultado se compara con el de la entrada PIN.
Si el PIN es incorrecto, se enciende la señal de PIN_INCORRECTO y se incrementa el contador de intentos fallidos. Si el usuario introduce un pin incorrecto dos veces seguidas se enciende la señal de ADVERTENCIA. Si introduce un pin incorrecto tres veces seguidas, se enciende la señal de BLOQUEO.
Una vez que se alcanza el estado de bloqueo, solo se puede restablecer el funcionamiento del cajero reiniciando la máquina de estados con una secuencia de RESET=0 y RESET=1.
Si el PIN es correcto se procede a leer la entrada TIPO_TRANS para saber si se trata de un depósito o un retiro.
En caso de un depósito simplemente se suma el valor de MONTO al valor de BALANCE y cuando se completa la operación se enciende BALANCE_ACTUALIZADO.
En el caso de un retiro se verifica que el MONTO sea menor que el BALANCE. De ser así, se actualiza el balance y se enciende la indicación de BALANCE_ACTUALIZADO y la de ENTREGAR_DINERO.
Si el MONTO es mayor que el BALANCE se debe encender la indicación de FONDOS_INSUFICIENTES.

2. Escribir una descripción conductual del controlador de cajero automático usando Verilog. Esta descripción servirá como una especificación detallada y formal del funcionamiento del dispositivo diseñado.

3. La descripción en Verilog deberá tener al menos un módulo de banco de pruebas, un módulo probador, y un módulo con la descripción del controlador.

4. Definir un plan de pruebas para garantizar el funcionamiento del diseño. El plan de pruebas debe cubrir todos los modos de operación del controlador, es decir, cubrir todos los casos de uso descritos en este enunciado. El módulo probador debe suministrar las señales necesarias para que las pruebas se realicen.

5. Realizar la síntesis del controlador del cajero automático usando Yosys y la biblioteca cmos_cells.lib. Indicar en el reporte la cantidad de compuertas (por tipo) y flip flops requeridos por el diseño.

6. Realizar el mismo plan de pruebas sobre el netlist sintetizado y verificar su correcto funcionamiento.

**Entregables:**

Con el fin de facilitar el proceso de revisión, se le solicita entregar el proyecto en EDA playground o bien organizar los entregables de la tarea de la siguiente manera:

a) Entregar un solo archivo comprimido, y nombrado según el patrón <# de carné>.<formato de compresión>, por ejemplo B41047.zip

b) El archivo comprimido descrito en el rubro a) deberá contener específicamente los siguientes archivos:

- Reporte en formato PDF cuyo nombre debe seguir el patrón <# de carné>.pdf, por ejemplo, B41047.pdf
- Uno o varios archivos de Verilog con el formato <nombre de archivo>.v que construyan la solución que se solicita en la tarea.
- Un solo archivo probador llamado tester.v
- Un solo archivo de banco de pruebas llamado testbench.v
- Un archivo Makefile que permita correr todos los pasos de simulación con una sola línea de comando.

c) El banco de pruebas, testbench.v, deberá incluir a todos los demás archivos *.v, de modo que la compilación y simulación del testbench implique la compilación y simulación de todos los archivos internos.

d) El probador, tester.v, debe escribirse de forma tal que una sola simulación contenga los resultados de una escritura, una lectura y un proceso de RESET, que ejemplifiquen el funcionamiento esperado del módulo.

e) El Makefile debe contener, como mínimo, los comandos necesarios para correr la compilación y simulación de los módulos de la tarea. La figura #2 muestra un ejemplo general como guía.

```
tarea: testbench.v mdio.ys #Archivos requeridos
       yosys -s mdio.ys    #Corre sìntesis
       iverilog -o salida testbench.v #Corre Icarus
       vvp salida #Corre la simulación
       gtkwave resultados.vcd #Abre las formas de onda
```

Figura #2: Ejemplo de Makefile

## Rúbrica de Calificación

| Tarea #3: Controlador para cajero automático | Categoría | % Categoría | % Rubro | % Total |
|---|---|---|---|---|
| Existe una descripción del comportamiento de la máquina de estados del controlador para cajero automático, ya sea como diagrama de estados o como tabla de estados. La descripción incluye el comportamiento correcto de todas las señales relevantes. | Código | 10% | 20% | 2% |
| Existe una descripción conductual en Verilog del controlador para cajero automático. Esta descripción incluye al menos un módulo de banco de pruebas (testbench.v), un módulo probador (tester.v) y un módulo para el dispositivo bajo prueba (DUT). | Código | 10% | 20% | 2% |
| Las descripciones de Verilog se entregan en archivos distintos al reporte, listos para ser simulados, e incluyen un archivo de Makefile, de modo que la simulación se corre con una sola línea de comando. Alternativamente se entrega un proyecto en EDA playground. | Código | 10% | 10% | 1% |
| Las descripciones en Verilog están comentadas adecuadamente para que otras personas entiendan la lógica de la descripción. | Código | 10% | 10% | 1% |
| Las descripciones en Verilog compilan sin producir errores. | Código | 10% | 20% | 2% |
| Las descripciones en Verilog ejecutan correctamente. Es decir, corren, entregan algunos resultados y finalizan. | Código | 10% | 20% | 2% |
| El dispositivo completa correctamente un depósito, incluyendo revisar el pin correcto. | Pruebas | 40% | 40% | 16% |
| El dispositivo completa correctamente un retiro, incluyendo revisar el pin correcto e indicar fondos insuficientes cuando corresponde. | Pruebas | 40% | 40% | 16% |
| El dispositivo reacciona ante el ingreso de pines incorrectos de acuerdo con la especificación dada. | Pruebas | 40% | 20% | 8% |
| La descripción por comportamiento completa correctamente el proceso de síntesis y produce un netlist equivalente según la biblioteca utilizada. | Síntesis | 40% | 20% | 8% |
| El netlist sintetizado completa correctamente un depósito, incluyendo revisar el pin correcto. | Síntesis | 40% | 30% | 12% |
| El netlist sintetizado completa correctamente un retiro, incluyendo revisar el pin correcto e indicar fondos insuficientes cuando corresponde. | Síntesis | 40% | 30% | 12% |
| El netlist sintetizado reacciona ante el ingreso de pines incorrectos de acuerdo con la especificación dada. | Síntesis | 40% | 20% | 8% |
| El reporte contiene las seguientes secciones: Resumen, descripción arquitectónica, plan de pruebas, instrucciones de utilización de la simulación para quien califica, ejemplos de resultados, conclusiones y recomendaciones. | Reporte | 10% | 40% | 4% |
| que se hizo, las partes del diseño que dieron más trabajo para completar y por qué, una explicación de los problemas que se presentaron y cómo se solucionaron. | Reporte | 10% | 40% | 4% |
| La longitud del reporte no excede 10 páginas. | Reporte | 10% | 20% | 2% |

**Guía para el reporte** (Sigue los mismo lineamientos del reporte de los proyectos)

Se debe entregar en forma electrónica un documento, a lo sumo de 10 páginas de longitud, que incluya los siguientes puntos:

1. **Resumen**: Breve (Media página máximo) descripción de **todo** el proyecto. Esta sección es fundamental pues puede determinar si el lector se interesa o no en leer los detalles del proyecto. Un resumen mal hecho puede esconder un excelente proyecto. El resumen debería incluir:

    a) Descripción breve del sistema, es decir, qué hace. Incluya alguna característica que considere que distingue este diseño en particular.
    b) Las pruebas que se realizaron y qué resultados se obtuvieron. Indique problemas que se tuvieron que considere importante resaltar.
    **c)** Conclusiones más importantes y recomendaciones para un diseño posterior.

2. **Descripción Arquitectónica**: Incluye un diagrama de bloques con las señales más importantes que sirve como base para describir el funcionamiento del sistema. La descripción va en términos de lo que se espera que el sistema haga. Es decir, se debe detallar la funcionalidad del sistema, el protocolo de las señales que se usan para que funcionen cada una de las partes y las secuencias de eventos que se deben dar. Esta descripción podría ir acompañada de tablas de verdad, tablas de transición de estados, diagramas de estados, diagramas temporales, etc.

3. **Plan de Pruebas**: Aquí se deben enumerar, esto es, se debe presentar una **lista detallada** de las pruebas que se le van a hacer al diseño para verificar que está funcionando de acuerdo a las especificaciones dadas. La lista debe contener por lo menos los siguientes elementos i) Nombre/número de prueba, ii) Descripción de la prueba, y iii) Una indicación de si el diseño la falló o la pasó. Estas pruebas podrían incluir la generación de vectores de entrada para probar en forma exhaustiva todas las líneas de una tabla de verdad o tabla de estados, patrones aleatorios de entradas para tratar de causar errores en la respuesta del diseño, o patrones específicos que ejerciten un cierto modo de funcionamiento. Cada prueba debería ser claramente enumerada en el plan para que también se pueda hacer referencia a ella en el código del banco de pruebas del diseño.

4. **Instrucciones de utilización de la simulación**: Esta sección debe mostrar los comandos necesarios para hacer funcionar la simulación en todos los casos que especifica el plan de pruebas. Hay que suponer que el diseño de un grupo puede ser utilizado por otro grupo o el profesor. Si los resultados no se pueden repetir porque no se conocen los comandos para hacer funcionar la simulación entonces es como si el diseño no funcionara del todo. Se recomienda crear un Makefile de modo que se pueda correr todas las pruebas del caso con un solo comando en Icarus Verilog y GTKwave.

5. **Ejemplos de los resultados**: Una descripción de los resultados más importantes acompañados de los diagramas temporales de la simulación (GTkWave) o cualquier otra salida que demuestre claramente el comportamiento descrito. No es necesario incluir una muestra exhaustiva de resultados, sino que los más representativos del diseño. El punto es mostrarle al lector los comportamientos más sobresalientes para formarle una idea clara

del funcionamiento del diseño. Ya verá el lector si desea más detalles, entonces podrá correr una simulación.

6. **Conclusiones y recomendaciones**: Basado en los resultados obtenidos se indica aquí qué se logró con el proyecto. Puede ser que se concluya que con el diseño propuesto se tiene una limitación en la velocidad de respuesta de… etc. O que con ciertas combinaciones de entradas el diseño se vuelve inestable o los resultados no son los esperados. También se puede concluir qué ventajas o problemas encontraron al seguir el plan de trabajo. A raíz de las conclusiones se puede también recomendar cómo se podría mejorar el diseño o qué otras pruebas se le podrían hacer para garantizar su funcionamiento en otras condiciones que al principio no se consideraron, o también cómo se debería planear el siguiente proyecto para poder cumplirlo a tiempo.