Fecha de entrega 09 de 05 de 2023.

Simulación cajero automático

CRISTIAN GARCIA[1], JOAQUIN ROLDÁN[1]
[1]Pontificia Universidad Católica de Chile (e-mail: crisgarcia@uc., jeroldan@uc.cl)

Este proyecto ha sido desarrollado bajo el curso IEE2463: Sistemas Electrónicos Programables.

ABSTRACT El proyecto corresponde a una simulación de un cajero automático. Para realizar esto se utilizaron tres bloques principales. Primero se encuentra en el estado inicial 00, se aprieta un botón para comenzar y posteriormente se pone la clave correspondiendiente, de ser correcta se aprieta otro boton para pasar al estado siguiente. En el segundo estado uno pone mediante los switch cuando dinero quiere retirar y preciona un boton para pasar al siguiente estado. Finalmente en el tercer estado se compara la cantidad de dinero que se quiere retirar y cuanto tiene la cuenta. En caso de ser posible la transaccion se prende un led verde en caso contrario uno rojo. Para implementar esto se utilizaron varios ip que fueron cargados en un block desing. Uno de estos bloques fue configurado con axi para recibir datos mediante axi lite que van a corresponder a la cantidad que se desea retirar. Finalmente se logro interconectar los bloques y recibir datos mediante AXI, esto se podia evidenciar en el correcto funcionamiento del led RGB.

INDEX TERMS
Cajero, Axi lite, VHDL, maquina de estado.

I. ARQUITECTURA DE HARDWARE (1 PUNTO)

Para realizar este proyecto nos basamos en el funcionamiento de un cajero automático y tratamos de hacerlo de la manera mas realista posible dado nuestros conocimientos.

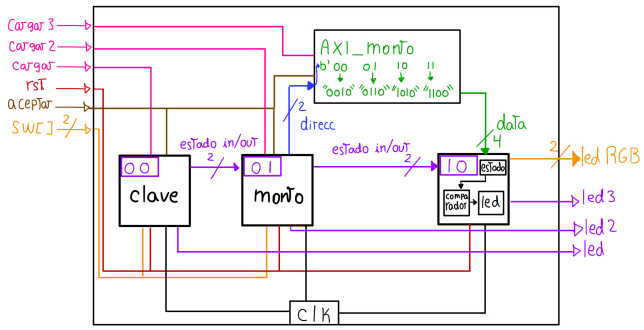

FIGURE 1: Este diagrama corresponde al funcionamiento del modelo del cajero

Los bloques negros corresponden a los estados de nuestro sistema los cuales comparten una variable estado que se va actualizando a medida que se cumplan ciertas condiciones. Cada uno de estos estados esta asociado con un led para poder saber visualmente en que estado nos encontramos, en caso de apretar reset volvemos al estado cero, además todos los bloques están funcionando bajo un clock general. El primer bloque corresponde a clave que es el estado 00, este comienza con el led 1 encendido, al momento que subimos el switch cargar el led 1 comienza a parpadear y esto significa que nuestra tarjeta esta esperando que pongamos la clave, luego de poner la clave si esta es correcta y se presiona el botón aceptar pasa al siguiente estado, en caso se de incorrecta vuelve al momento inicial. El segundo estado corresponde a monto (01), este bloque recibe el estado actual, en el caso de ser 01 actúa. Este bloque se comunica con nuestra ram para informarle cuanto dinero se desea retirar, para esto se le entrega 2bits mediante los switchs que van a corresponder a las direcciones del AXI que tienen asociados 4 bits predefinidos. El 00 vale 2, el 01 vale 6, el 01 vale 10, el 11 vale 13. Como en el caso anterior si el switch aceptar esta en 1 el led dos parpadea informándonos que esta listo para recibir información, luego de poner el monto se apreta aceptar y mande el monto deseado a la RAM.

En el caso del bloque AXI monto este tiene un botón para que el AXI traffic gen cargue la información de los archivos coe que contienen los montos a retirar en la RAM. La salida de este bloque corresponde a una data de 4 bits que sera el monto deseado.

Finalmente el ultimó bloque corresponde a comparación, el estado 10 este recibe cuanto se quiere retirar y compara con el monto que tiene la cuenta, este es un valor preestablecido. En caso de ser posible la transacción se prende el led verde, en caso contrario se prende el led rojo. Este bloque esta conformado por tres componentes dentro de esta entity.

II. ACTIVIDADES REALIZADAS (1.5 PUNTOS)

AO1 (100%): El código cumple con las funciones que habíamos planteado en nuestro proyecto las cuales son: El bloque que compara esta conformado por tres componentes, un comparador, un componente que enciende el led RGB y un componente que lee el estado actual. El código implementado ha compilado, se logra simular luego de la implementación, se logra generar bitstram y cargarlo en la ZYBOZ7.

AO2 (100%): Se logro generar generar 3 diferentes bloques que funcionan correctamente, estos corresponden a clave, monto, comparacion. Estos se relacionan entre ellos y funcionan y compilan correctamente.

AO3 (50%): Se logro una comunicacion AXI exitosa entre un IP esclavo y un ATG maestro en test mode este compila y funciona correctamente. La comunicacion mediante Advance Mode no fuimos capaces de implementarlo.

AC1(100%): Los bloques anteriormente mencionados estan todos relacionados a través de una maquina de estados que funciona correctamente.

AC2(85%): Utilizamos 4 botones, 3 switches, y 3 leds. Estos fueron suficientes para realizar nuestro proyecto y funcionaban de forma correcta.

AC3(50%): Se utilizaron mas de 5 operadores como: =>, <=, and, :=, =, <, +, $\neq$, -.

AC4(100%): En todos los bloques se implemento un divisor de clock para hacer parpadear una luz cada un segundo la cual se va actualizando instantaneamente.

AC5(100%): La mayor parte del código es secuencial, ya que, esta dentro de diferentes process y va recibiendo feedbacks del sistema para realizar acciones. En cambio tenemos código concurrente en el axi al obtener los valores de la RAM, ya que, estos son valores fijos que no dependen de otros inputs.

AC6(50%): Se utilizo reiteradas veces la función rising_edge para realizar las acciones cada vez que el clk tenia un flanco de subida, no se utilizo procedures que es muy similar a function pero que a diferencia de este puede retornar mas de un valor.

AC7(100%): Se implemento el led RGB el cual no fue visto en clases.

III. RESULTADOS DE SIMULACIÓN (3 PUNTOS)

Muestre su simulación post implementación de alguno de sus procesos con el fin de:

- Durante la simulación por bloques, específicamente en el bloque de monto, se pudo visualizar un retardo o problema a la hora de cargar un bus de 2 bits por las condiciones internas que actuaban en base al flanco de subida, pero fue corregido y se logró enviar en forma correcta el bus de estado y un bus con la dirección hacia nuestra ram.
- Utilizamos una variable dentro de un código secuencial process, en un componente de nuestro bloque comparador(main) ya que una señal no era óptima para mantener un valor fijo respecto al monto de la cuenta del usuario como información local durante la comparación de valores. Utilizamos señales cuando necesitábamos una actualización después de una conclusión post código secuencial como los estados actuales de cada uno.
- Respecto a AXI-Lite, primero cargamos los archivos data.coe con valores 2, 6, A y C los cuales eran los posibles montos a retirar de una cuenta corriente la cuál tenía 8 como monto interno. El package Monto se encargaba de entregar una dirección en base a la carga del valor de los switches en la Ram, el que luego era autorizado para escribirse y desde la Ram entregabamos un valor de 4 bits al comparador el cual representaba la autorización o negación de la transacción del monto en el color de salida del led rgb, que tenia como inicio color rojo y si es que se aprobaba daba el color verde. Su comprobación inicial del correcto envío fue evaluando el bus de bit que salía encendiendo los leds en la zybo.

IV. RESULTADOS IMPLEMENTACIÓN (0.1 PUNTOS)

Fue un constante debugeo respecto a los estados donde se encendían leds que no correspondían por saltos de fase pero fueron arreglados. No se logró implementar un reset que reiniciara todo el proceso desde 0 ya que la ram mantenía el constante envío de información, entonces solo se apagaban los leds al activarse pero no se podía volver a cargar la información para encontrar una respuesta ante un nuevo intento de extracción de monto.

V. CONCLUSIONES(0.2)

Enumere las conclusiones más importante de su proyecto. Recuerde que:

- Por bloques, Clave entrega un bus de 2 bits en un flanco de clock ante la activación. Monto entrega 2 buses de 2 bits ante activación y Axi monto entrega un bus de 4 bits ante activación con el cual se finaliza en el bloque Comparador nuestro resultado. Sin embargo, el bit de reset no reinicia la información cargada en un proceso debido a no poder reiniciar el bus recibido por Axi monto y cargar nueva respuesta.

VI. TRABAJOS FUTUROS (0.2)

Este proyecto si se trabaja mas a fondo es posible hacerlo mas realista, ya que esta pensado para ser aplicable para una sola tarjeta. Se podría implementar para varias claves y que tengan diferentes montos cada una, además se podría implementar un sistema que a medida que se saca plata esta se vaya descontando del monto que tiene la cuenta. Se podría implementar un monto total que tiene el cajero y a medida que la gente vaya sacando este vaya disminuyendo hasta que la maquina niega la transacción pero no porque la persona no tiene saldo si no porque la maquina no tiene mas dinero para imprimir. También se podría implementar un botón para retroceder en los estados.