Lógica Secuencial de Control

Jose Pablo Fuentes Apú, Jose David Sánchez Schnitzler
email: josepablofa@gmail.com
jose.david.san@gmail.com
Área Académica de Ingeniería en Computadores
Instituto Tecnológico de Costa Rica

***Resumen*—En este cuarto laboratorio se desarrolla un prototipo de máquina de café a base de los requerimientos principales brindados por el profesor. En este caso es necesario trabajar con elementos de laboratorios pasados y además de esto se necesita incluir en el funcionamiento del sistema una máquina de estados finitos para definir el flujo del proceso que se desea implementar para el correcto funcionamiento de dicho sistema.**

***Palabras clave*—prototipo, máquina de café, máquina de estados finitos, flujo.**

I. Introducción

Teniendo un conocimiento general del funcionamiento de los componentes electrónicos elaborados por medio de módulos en Quartus Prime se puede hacer un sistema controlado por medio de una serie flujos de procesos y requerimientos brindados por el profesor. En este caso para que el sistema completo funcione se necesita incorporar un componente nuevo que se llama máquina de estados finitos. Dicha máquina se tiene que programar de manera que la lógica contemple todos los casos para la solución del problema.

Las máquinas de estado finito se pueden clasificar en dos categorías o clases, las cuales son sincrónicas o asincrónicas. En el caso de las máquinas de estado sincrónicas podemos encontrar dos tipos, dichos tipos son las máquinas de Moore y por otro lado las máquinas de Mealy.

Empezando por la máquina Moore, se destaca que las salidas únicamente dependen del estado presente como se muestra en la figura 1.

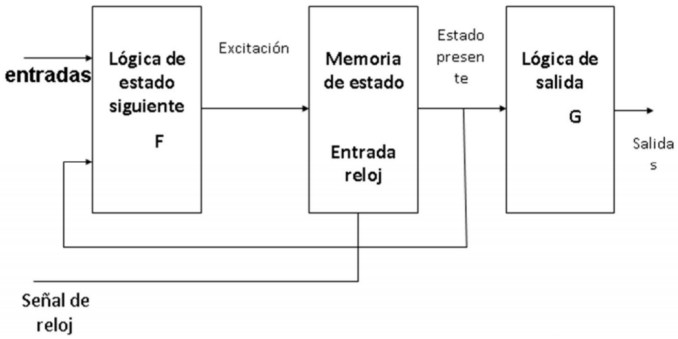

Figura 1. Diagrama de estado y señales de la máquina de Moore [1]

Seguidamente se presenta la máquina de Mealy la cual se destaca que la salida va a depender tanto del estado presente como de las entradas externas. Esto se aprecia en la figura 2.

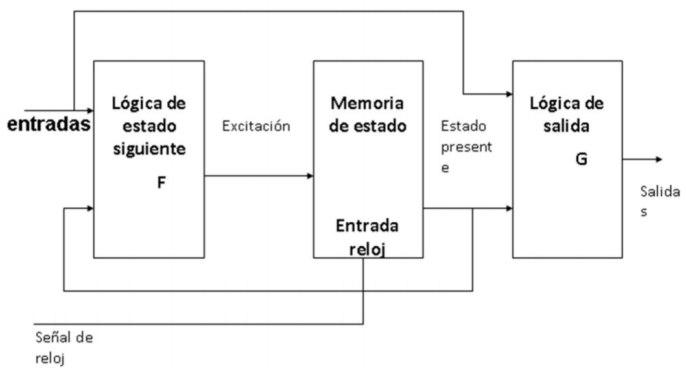

Figura 2. Diagrama de estado y señales de la máquina de Mealy [1]

Para el desarrollo de sistemas digitales secuenciales es necesario tener claro el concepto de setup time y hold time. Primeramente el setup time es cuando los datos de las señales de entrada son estables, exactamente antes de que se de un cambio en el reloj mientras que el hold time es cuando los datos de las señales de entrada son estables pero exactamente después de que ocurre un cambio en el reloj [2]. Esto se puede apreciar en la figura 3. La importancia de tener estos datos en cuenta al momento de diseñar un sistema digital es que se debe ser preciso con los tiempos ya que esta característica es la que va a definir un buen o mal funcionamiento.

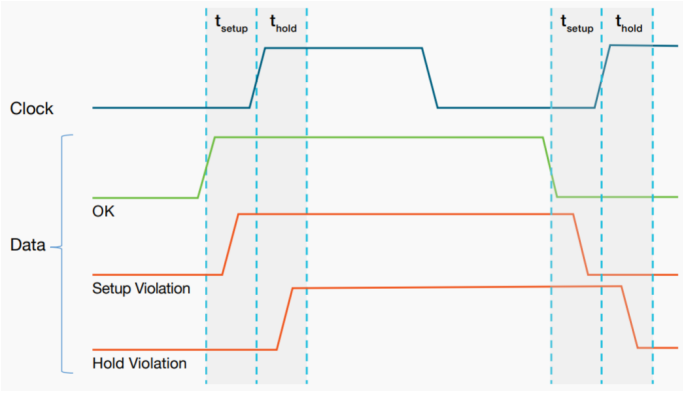

Figura 3. Setup Time y Hold Time en un sistema digital [2]

Para finalizar este apartado es importante tomar en cuenta un concepto llamado efecto rebote en señales digitales. Este efecto se produce ya que en los componentes mecánicos como lo son botones, interruptores, entre otros de este estilo existe una pequeña vibración al momento de ser utilizados

la cual puede interferir con la señal digital. Existen circuitos especializado para eliminar el efecto rebote como se muestra en la figura 4. Dicho circuito presenta un condensador el cual se va a descargar de manera constante al momento de accionar un componente mecánico, de esta manera existe un efecto de amortiguamiento en la señal. [3]

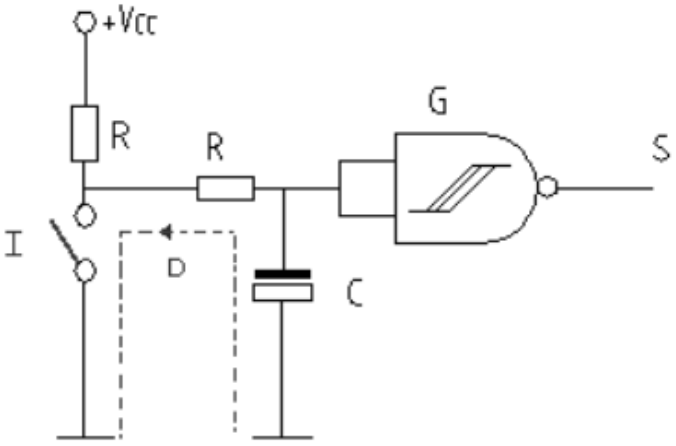

Figura 4. Circuito para eliminar el efecto rebote de componentes mecánicos [3]

II. Desarrollo

El desarrollo de este laboratorio consta en la elaboración de una máquina de café implementando lógica secuencial de control. Este sistema se va a dividir en dos grandes partes, primero va a procesar las monedas ingresadas, y en la segunda mitad se va elaborar la bebida comprada dependiendo en la selección de la primera mitad. Es de esta manera que el proceso de dicha bebida se ejecuta y una vez terminado el proceso se hace un reset del sistema dando el vuelto que le corresponde al cliente. El diagrama de segundo nivel de este sistema se muestra en la figura 5.

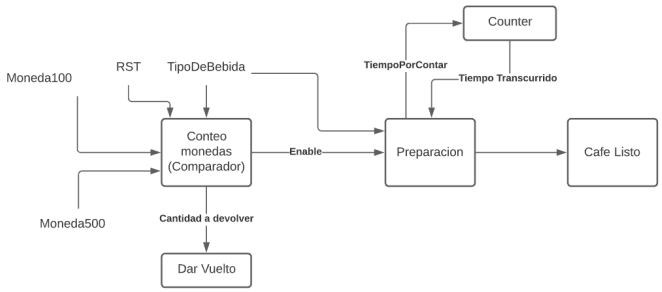

Figura 5. Diagrama de segundo nivel del sistema completo.

II-A. Parte 1: Procesamiento de monedas

Para esta primera parte se diseña un diagrama de flujo para tomar en cuenta todas las posibles decisiones del cliente y además contemplar todos los requerimientos presentes en el documento sobre el laboratorio 4 brindado por el profesor. En este caso es necesario que el sistema reciba monedas de 100 y de 500, además de esto el sistema debe ser capaz de cancelar la orden devolviendo el dinero total ingresado. Finalmente esta parte del sistema debe ser capaz de definir en sus salidas el vuelto total en caso de que haya, la activación del proceso por medio de un dato booleano y la bebida seleccionada. En la figura 6 se muestra el diagrama de flujo mencionado anteriormente.

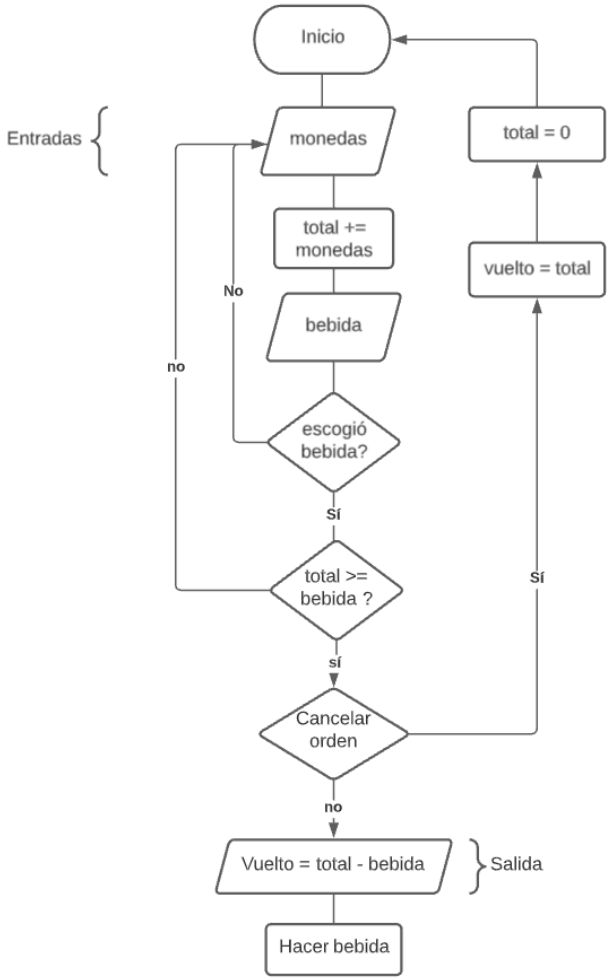

Figura 6. Diagrama de flujo que muestra el procesamiento del dinero y selección de bebidas.

Contemplando más a fondo las partes de este sistema de procesamiento de monedas se pueden contemplar 6 pequeños módulos necesarios para su funcionamiento. Dichos módulos son coinCounter diseñado específicamente para contar monedas sin tomar en cuenta los ciclos de reloj. Seguidamente se encuentran los módulos totalRegister y drinkRegister que se utilizan para almacenar el valor de las monedas ingresadas y la bebida seleccionada respectivamente. Luego se encuentra el módulo coinComparer cuyo dicho funcionamiento es el comparar el precio de la bebida seleccionada con el total de monedas que se han ingresado de esta manera saber si se necesitan más monedas o ya es posible hacer la bebida. Para finalizar se encuentran los módulos muxDrinks que define la bebida seleccionada con su precio y muxCoinsOutput que va devolver el vuelto respectivo si se hizo o no la compra de la

bebida. Los módulos mencionados anteriormente se muestran en un módulo llamado coinController encargado de conectar todos los módulos. Esto se muestra en el diseño de la figura 7.

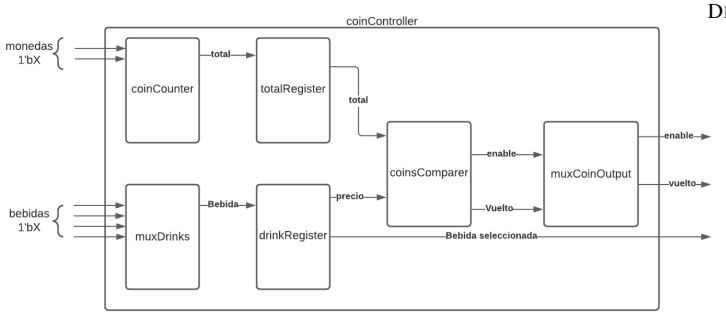

Figura 7. Diagrama para el controlador de módulos llamado CoinController

En pocas palabras este módulo llamado coinController se va a hacer cargo de enviar la señal para activar la elaboración del café dependiendo de la bebida seleccionada y las monedas ingresadas. También se va a encargar de devolver el dinero respectivo al cliente en caso de que haya.

II-B. Parte 2: Elaboración de bebidas

Para el módulo final y correspondiente a la preparación de café, se propone realizar una maquina de estados finitos donde el comportamiento se describe en el cuadro I, la cual tiene como entradas una señal que indica que el proceso de conteo de monedas y selección de bebida se ha completado con éxito (esta variable se llamará *enable*) y una señal llamada *timeChecked*, que indica que el tiempo necesario para el ingrediente o estado ha concluido. También cuenta con las señales de entrada correspondientes al reloj del sistema y al reinicio o reset.

A su vez, la máquina de estados finitos tiene una única salida y es la que indica si la bebida tiene el último ingrediente y el proceso de preparación ha terminado. Esta salida lleva el nombre de *ready*.

Cuadro I
DESCRIPCIÓN DEL COMPORTAMIENTO DE LA MÁQUINA DE ESTADOS DEL CICLO DE PREPARACIÓN DE UNA BEBIDA

Estado Actual	enable	timeChecked	Estado Siguiente	ready
000	0	x	000	0
000	1	x	001	0
001	x	0	001	0
001	x	1	010	0
010	x	0	010	0
010	x	1	011	0
011	x	0	011	0
011	x	1	100	0
100	x	0	100	0
100	x	1	101	0
101	x	0	101	0
101	x	1	110	0
110	x	0	110	1
110	x	1	000	0

Cada estado de la maquina de estado representa un ingrediente en la preparación de la bebida, como se muestra en el cuadro II

Cuadro II
DESCRIPCIÓN DEL COMPORTAMIENTO DE LA MÁQUINA DE ESTADOS DEL CICLO DE PREPARACIÓN DE UNA BEBIDA

Estado	Significado/Ingrediente
000	Estado inicial
001	Agua
010	Café
011	Leche
100	Chocolate
101	Azúcar
110	Estado final

Es importante resaltar que, las variables de entrada de la máquina de estados, son *enable* y *timeChecked*, esta última es una variable de salida de un módulo que se encarga de contar ciclos de reloj y activar una bandera cuando esos ciclos alcanzan un valor deseado, por ejemplo el valor perteneciente a un 3 segundos. El tiempo que debe de contar el módulo se calcula en base a la bebdida que esté preparando y al ingrediente o estado en el que se encuentre la máquina de estados. En el cuadro III se detallan los segundos necesarios para cada ingrediente dependiendo de la bebida escogida

Cuadro III
TIEMPO EN SEGUNDOS NECESARIO PARA CADA INGREDIENTE DEPENDIENDO DE LA BEBIDA

Bebida	Agua	Café	Leche	Chocolate	Azúcar
Expresso	2	3	0	0	1
Café con Leche	2	2	1	0	1
Capuccino	2	1	2	0	1
Mocaccino	1	1	1	2	1

III. RESULTADOS

En esta sección se muestran los resultados que se obtuvieron en la sección de Desarrollo, específicamente en las subsecciones referentes a los resultados simulados y presentes en la FPGA.

III-A. Resultados del módulo coinController

Al realizar los módulos de cada función requerida en el laboratorio, Quartus genera en circuito simulado. Esto se usa para verificar el buen funcionamiento de cada módulo y tomarlo de referencia para compararlo con los diseños mostrados en el apartado de Desarrollo.

Seguidamente se muestra el circuito del módulo coinController generado por el programa Quartus Prime una vez programado en el lenguaje de descripción de hardware llamado SystemVerilog. Este circuito contiene todos los módulos mencionados en el apartado de Desarrollo como lo son registros, multiplexores, comparador, sumador para el total de monedas y restador para el vuelto total del usuario. Dicho circuito se muestra en la figura 8.

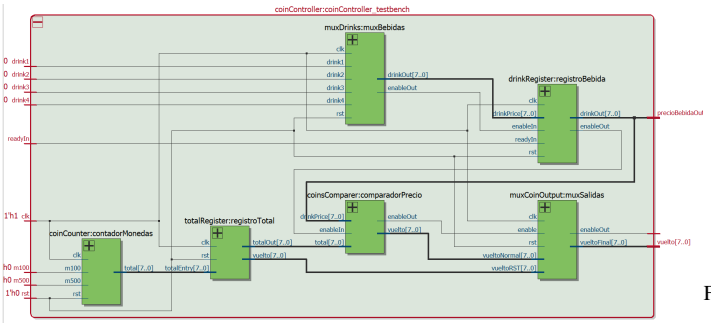

Figura 8. Diagrama para el controlador de módulos llamado CoinController

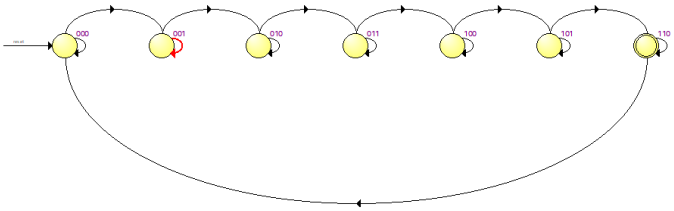

Figura 9. Máquina de estados finitos descrita en QuartusPrime

Con respecto al testbench que se diseñó para este módulo, se muestran los resultados en el cuadro IV. Se hacen pruebas para para dos grandes casos. Primero el caso en el que se ingresa una moneda de 100, otra moneda de 500, se selecciona la bebida valorada en 500 y se da el visto bueno con la entrada readyIn.

Cuadro IV
PRUEBAS A REALIZAR AL MÓDULO COINCONTROLLER PARA LA PRIMER PRUEBA

m100	m500	precio bebida	readyIn	monedero	enable	rst
1	0	0101	0	0001	0	0
0	1	0101	0	0110	0	0
0	0	0101	1	0001	1	0
0	0	0000	0	0000	0	1

Seguidamente se hace una segunda prueba para el caso en el que se seleccione una bebida y se active el readyIn pero no se tenga el saldo suficiente para comprar dicha bebida. Dichos resultados se muestran en el cuadro V.

Cuadro V
PRUEBAS A REALIZAR AL MÓDULO COINCONTROLLER PARA LA PRIMER PRUEBA

m100	m500	precio bebida	readyIn	monedero	enable	rst
1	0	0111	0	0001	0	0
0	1	0111	0	0110	0	0
0	0	0101	1	0110	0	0
0	0	0000	0	0000	0	1

III-B. Resultados del procesamiento de bebidas con la FSM

Para el módulo final que corresponde a la preparación de la bebida cuando el proceso de conteo de monedas y escogencia de bebida han sido existosos, se describió el comportamiento de la maquina de estados en el programa Quartus Prime descrita en el cuadro I.

En este programa se hace uso del lenguaje de descripción de hardware, SystemVerilog. El resultado al utilizar el programa para describir la máquina de estados se muestra en la figura 9.

El módulo de preparación requería de un sub-módulo que contara los ciclos de reloj, para indicar si el tiempo necesario para cada ingrediente/estado se ha cumplido. El resultado al describir el módulo en Quartus Prime utilizando SystemVerilog, se muestra en la figura 10. Este módulo da como salida la variable *timeChecked*. El tiempo que es necesario esperar, es indicado mediante el parámetro N, el cual es calculado tomando en cuenta el estado en el que se encuentra la máquina de estados y la bebida que se está preparando.

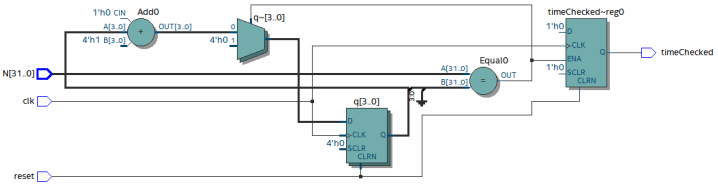

Figura 10. Módulo que cuenta los ciclos de reloj e indica si se ha cumplido el tiempo solicitado

Con respectos a los casos de prueba asociados al módulo de preparación, se pone a prueba el cada estado de la máquina de estados finitos cuando se está preparando la bebida *Expresso* el resultado de las pruebas se muestra en el cuadro VI. Igualmente, para las bebidas *café con leche*, *Capuccino* y *Mocaccino* los resultados se muestra en los cuadros VII, VIII y IX respectivamente.

Los tiempos de espera para cada ingrediente en cada bebida, se muestran en el cuadro III.

Cuadro VI
CASOS DE PRUEBA PARA LA PREPARACIÓN DE LA BEBIDA CAFÉ EXPRESSO

Actual	enable	tiempo	timeChecked	Siguiente	ready
000	1	0	1	001	0
001	1	2	1	010	0
010	1	3	1	011	0
011	1	0	1	100	0
100	1	0	1	100	0
101	1	1	1	110	0
110	1	0	1	000	1

Cuadro VII
CASOS DE PRUEBA PARA LA PREPARACIÓN DE LA BEBIDA CAFÉ CON LECHE

Actual	enable	tiempo	timeChecked	Siguiente	ready
000	1	0	1	001	0
001	1	2	1	010	0
010	1	2	1	011	0
011	1	1	1	100	0
100	1	0	1	100	0
101	1	1	1	110	0
110	1	0	1	000	1

Cuadro VIII
CASOS DE PRUEBA PARA LA PREPARACIÓN DE LA BEBIDA CAPUCCINO

Actual	enable	tiempo	timeChecked	Siguiente	ready
000	1	0	1	001	0
001	1	2	1	010	0
010	1	1	1	011	0
011	1	2	1	100	0
100	1	0	1	100	0
101	1	1	1	110	0
110	1	0	1	000	1

Cuadro IX
CASOS DE PRUEBA PARA LA PREPARACIÓN DE LA BEBIDA MOCACCINO

Actual	enable	tiempo	timeChecked	Siguiente	ready
000	1	0	1	001	0
001	1	1	1	010	0
010	1	1	1	011	0
011	1	1	1	100	0
100	1	2	1	100	0
101	1	1	1	110	0
110	1	0	1	000	1

IV. CONCLUSIONES

Las máquinas de estado finito tienen una serie de estados, que en conjunto al alfabeto y transiciones pueden ser utilizados para resolver y modelar un problema de la vida real con lógica secuencial. Es por eso que es importante definir los estados siguientes una vez las entradas cambian o mantienen su valor, así como definir las condiciones necesarias para que las salidas alteren su valor dependiendo del estado en el que se encuentra la máquina de estado. A su vez, es necesario plantear que la frecuencia del reloj utilizado se ajuste a lectura y escrituras de las diferentes entradas y salidas que se relacionan a la máquina de estado.

Es de suma importancia tomar en cuenta la frecuencia del reloj proporcionada por la FPGA ya que este es de 50 MHz quiere decir que los ciclos de reloj se ejecutan de manera excesivamente rápida lo que perjudica en los resultados reales mostrados por la FPGA aunque los datos de los testbench sean correctos y esperados. Implementando un divisor de frecuencia se arreglan estos problemas.

La implementación de un sistema digital se va a tornar excesivamente complicado en el caso de que su diseño se elabore de una manera poco optimizada en cuanto a componentes necesarios y lógica implementada. Es por esto que el diseño del sistema es de las partes más importantes al momento de elaborar un sistema digital de cualquier tipo.

REFERENCIAS

[1] ”Máquinas de estado Finito”, notas de clase del curso EL3307, Escuela de Electrónica, Instituto tecnológico de Costa Rica, Primer semestre del 2021.

[2] Pareles, D. (2019) *How to Track Down Setup and Hold Violations with a Mixed Signal Oscilloscope*. DesignNews. Recuperado en línea: https://www.designnews.com/electronics-test/how-track-down-setup-and-hold-violations-mixed-signal-oscilloscope

[3] Cardenas, R. (2020). *Eliminador de Rebotes en contactos Eléctricos Mecánicos*. Edublog Circuitos Eléctricos. Recuperado en línea: http://edublogcircuitosac.blogspot.com/2020/10/eliminador-de-rebotes-en-contactos.html