CONTADORES

Santillán Atilio Emanuel
Estudiante ingenieria electronica
Facultad de Ciencias Exactas y Tecnología

RESUMEN

El presente trabajo aborda el diseño y análisis de contadores digitales aplicando dos enfoques complementarios: el método basado en **máquinas de estado finito (FSM)** y la técnica de **datapath**. En primer lugar, se desarrolla un contador binario de tres bits con **reset sincrónico**, utilizando el procedimiento clásico de definición de estados, asignación de códigos, elaboración de tablas de transición y síntesis de la lógica combinacional correspondiente. Posteriormente, se diseña un contador binario de cuatro bits con **reset, carga y habilitación**, implementado mediante un **datapath** que incorpora un sumador de cuatro bits con propagación de acarreo como núcleo de procesamiento. Ambos diseños se analizaron teóricamente y se comprobaron sus comportamientos mediante diagramas de transición y de tiempo. Finalmente, se comparan las metodologías empleadas, destacando la escalabilidad y flexibilidad del enfoque por datapath frente a la simplicidad y claridad del método por estados y transiciones.

INTRODUCCIÓN

Un contador es, en su definición más elemental, la combinación de flip-flops conectados para realizar funciones de recuento; el número de flip-flops y la forma en que éstos se interconectan determinan el módulo del contador, es decir, el número de estados distintos por los que el contador puede pasar durante un ciclo completo. El funcionamiento de un contador se implementa finalmente como una máquina de estados finita: la información almacenada en los flip-flops (estado actual) y las entradas del sistema determinan el estado siguiente y las salidas correspondientes. (Floyd, 2006).

Los contadores se clasifican, de forma práctica, según cómo se distribuye la señal de reloj entre sus etapas. En un contador asíncrono (ripple counter) los flip-flops no cambian de estado exactamente al mismo tiempo, puesto que la salida de una etapa dispara la siguiente; este comportamiento origina retardos de propagación acumulados que limitan la velocidad máxima y pueden causar problemas de decodificación. En un contador síncrono, en cambio, la entrada de reloj llega a todos los flip-flops simultáneamente y las transiciones de estado se coordinan en el mismo instante de reloj, lo que facilita la temporización en sistemas de mayor velocidad. (Floyd, 2006).

Desde la perspectiva de diseño estructurado, un contador puede entenderse como una máquina de estados cuya transición entre estados puede realizarse mediante rutas de datos explícitas: la máquina de estados finita que implementa el conteo se puede considerar un contador cuando, por ejemplo, una condición de reinicio fuerza el circuito a un estado inicial y la lógica siguiente incrementa o modifica el registro de estado en cada pulso de reloj. En el enfoque de datapath, la lógica que calcula el siguiente estado se organiza como un flujo de datos que atraviesa bloques combinacionales y registros; de forma general, un circuito secuencial con datapath consta de un registro de estado, la lógica del siguiente estado y la lógica de salida, que conjuntamente definen la operación del contador. (Dally, 2016).

Ver el contador como un elemento del datapath facilita la reutilización y la verificación: el datapath es la parte del sistema encargada de manipular datos (registros, sumadores/ALU, multiplexores y buses) y la unidad de control genera las señales que seleccionan las operaciones y sincronizan las actualizaciones. Esta separación entre datapath y unidad de control permite emplear bloques estándar —por ejemplo, un sumador de n bits para obtener el siguiente valor— y reduce la complejidad al diseñar y verificar contadores dentro de sistemas más grandes. (Harris & Harris, 2018).

En la práctica de implementación, los contadores incluidos en un datapath suelen seguir idiomáticas sencillas: el siguiente estado se obtiene incrementando o decrementando el registro que contiene el estado actual, usando un

sumador/incrementador y un multiplexor para seleccionar entre diversas fuentes de actualización (p. ej. carga paralela, reinicio, conteo habilitado). Esta estructura (registro + lógica combinacional de incremento + lógica de control) se refleja de forma directa en descripciones HDL y en diagramas de datapath típicos. Además, la metodología recomendada es priorizar el diseño síncrono para minimizar problemas de sincronización y facilitar la escalabilidad del diseño dentro de sistemas complejos. (Dally, 2016; Wakerly, 2001).

MATERIALES Y MÉTODOS

A continuación, se detallarán los pasos que permitieron obtener la representación lógica y síntesis de cada diseño.

MÁQUINA DE ESTADO FINITO

Los pasos a realizar para realizar una máquina de estado finito sincrónica, serán los siguientes:

1- Definir estados y salidas

2- Definir la estructura a utilizar.

3- Construir la tabla de estado/salida

4- Asignación de estados y salidas

5- Elaborar una tabla de transición

6- Realizar la síntesis lógica de los bloques combinacionales de estado siguiente y salida

7- Implementar las funciones sintetizadas en 6, mediante circuitos logicos.

DATAPATH

1- Definir arquitectura
2- Seleccionar códigos
3- Determinar las expresiones aritmeticas y logicas para la evaluación del estado y la salida
4- Implemento las funciones lógicas mediante bloques combinacionales:

RESULTADOS

2. MÁQUINA DE ESTADO FINITO

Contador de 3 bits

Especificaciones:

Reset sincrónico

Enfoque basado en estados y transiciones

Si reset activo, en el siguiente flanco de clk la salida será 0

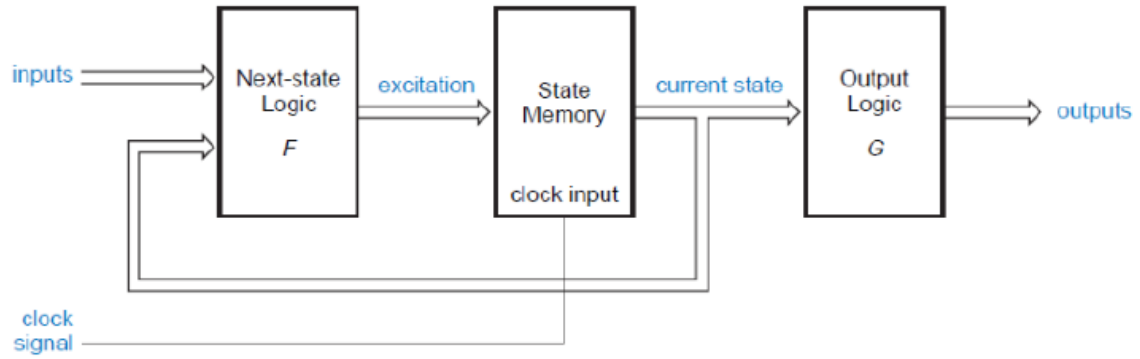
En otro caso: En cada flanco de clk la salida se incrementa en uno.

1- Definir estados estados y salidas:

Estado	E0	E1	E2	E3	E4	E5	E6	E7
Salida	0	1	2	3	4	5	6	7

2- Definir estructura

Máquina de Moore



3- Tabla de estados y salidas

Estado actual	Reset		Salida
	0	1	
E0	E1	E0	0
E1	E2	E0	1
E2	E3	E0	2
E3	E4	E0	3
E4	E5	E0	4
E5	E6	E0	5
E6	E7	E0	6
E7	E0	E0	7
	Estado siguiente		

4- Asignación de estados y salidas

Elijo para la salida el código binario natural, por simplicidad.

Salida	0	1	2	3	4	5	6	7
Código	000	001	010	011	100	101	110	111

Para los estados elijo también binario natural, para que la lógica de salida sea solamente un buffer.

Estado	E0	E1	E2	E3	E4	E5	E6	E7
Código	000	001	010	011	100	101	110	111

5- Tabla de transición

Estado actual			Reset		Salida		
Q2	Q1	Q0	0	1	S2	S1	S0
0	0	0	001	000	0	0	0
0	0	1	010	000	0	0	1
0	1	0	011	000	0	1	0
0	1	1	100	000	0	1	1
1	0	0	101	000	1	0	0
1	0	1	110	000	1	0	1
1	1	0	111	000	1	1	0
1	1	1	000	000	1	1	1
			Estado siguiente				

6- Síntesis de lógica de transición

R	Q2	Q1	Q0	Q2+	Q1+	Q0+
1	ind	ind	ind	0	0	0
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	1
0	0	1	1	1	0	0
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	0	1	1	1
0	1	1	1	0	0	0

Síntesis Q0+ :

Mapa de Karnaugh

		Q1	0	0	1	1
R	Q2	Q0	0	1	1	0
0	0		1	0	0	1
0	1		1	0	0	1
1	1		0	0	0	0
1	0		0	0	0	0
			Q0+			

$$Q0^{+} = \overline{R}\,\overline{Q0}$$

Síntesis

		Q1	0	0	1	1
R	Q2	Q0	0	1	1	0
0	0		0	1	0	1
0	1		0	1	0	1
1	1		0	0	0	0
1	0		0	0	0	0
			Q1+			

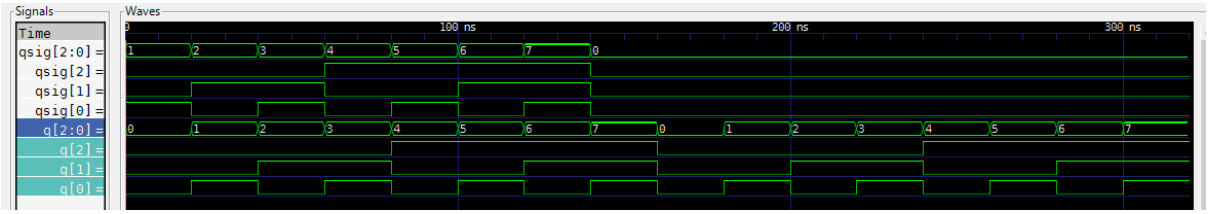
$$Q1^{+} = \overline{R}\,Q0\,\overline{Q1} + \overline{R}\,\overline{Q0}\,Q1 = \overline{R}\,(Q0\,\overline{Q1} + \overline{Q0}\,Q1) = \overline{R}\,(Q0 \oplus Q1)$$

Sintesis Q2+:

		Q1	0	0	1	1
R	Q2	Q0	0	1	1	0
0	0		0	0	1	0
0	1		1	1	0	1
1	1		0	0	0	0
1	0		0	0	0	0
			Q2+			

$$Q2^{+} = \overline{R}\,\overline{Q1}\,Q2 + \overline{R}\;Q0\,Q1\,\overline{Q2} + \overline{R}\;\overline{Q0}\;Q1\,Q2 = \overline{R}\,(\overline{Q1}\,Q2 + Q0\,Q1\,\overline{Q2} + \overline{Q0}\;Q1\,Q2)$$

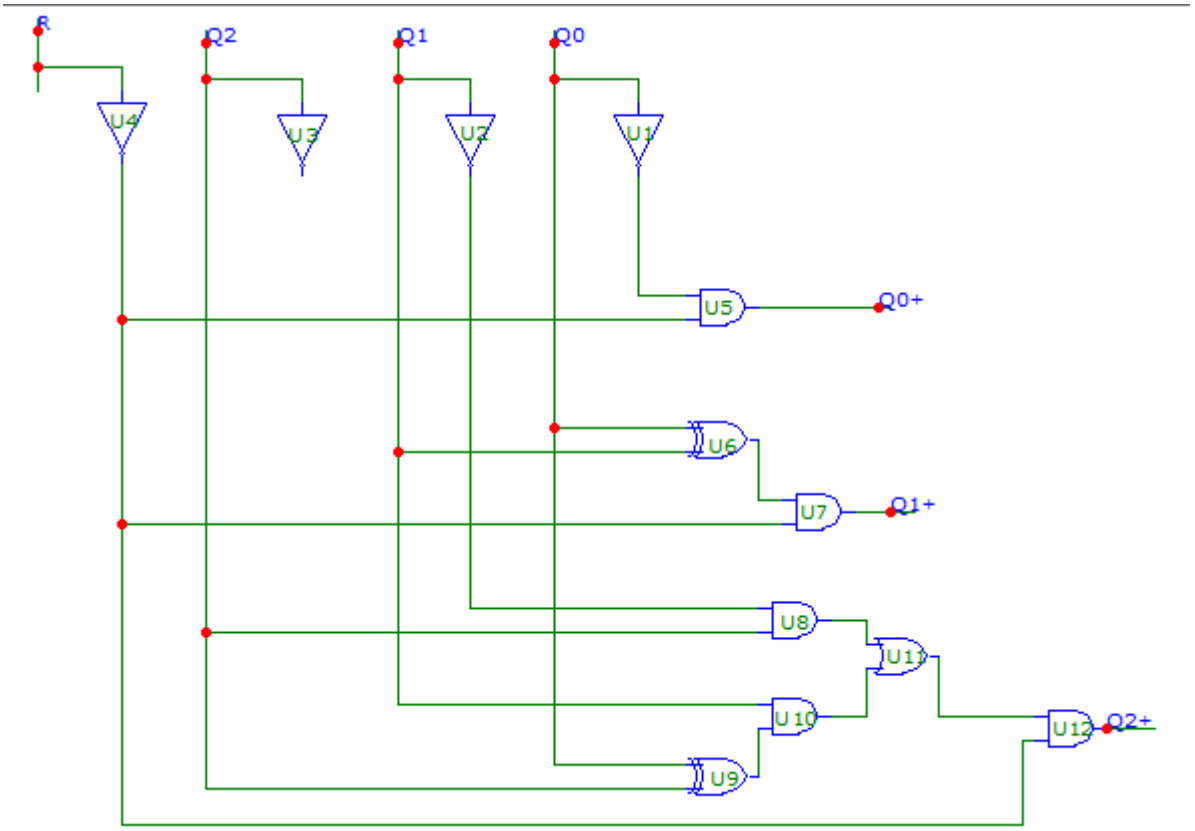
$$Q2^{+} = \overline{R}\,(\overline{Q1}\,Q2 + Q1\,(Q0 \oplus Q2))$$



Compruebo que la lógica de estado siguiente, sea correcta con VHDL.

7- Implementación mediante circuitos lógicos.

Bloque combinacional de estado siguiente:



El bloque combinacional de salida es un cable.

3. DATAPATH

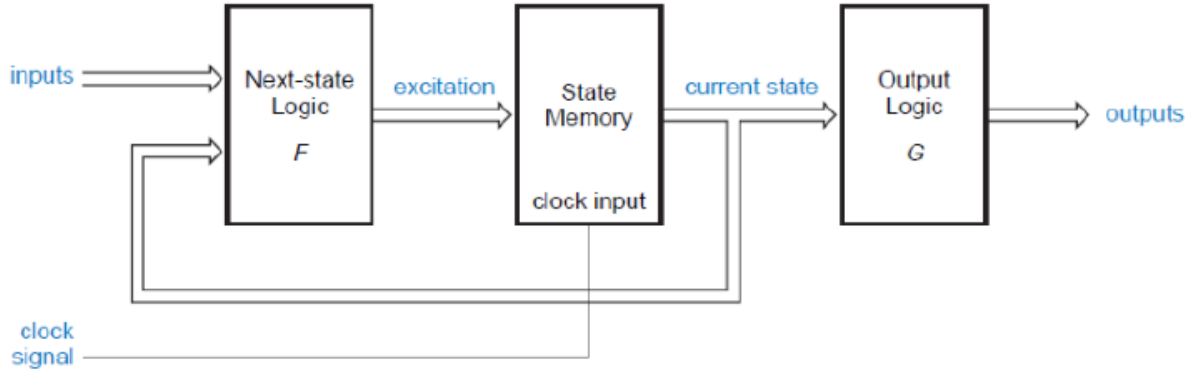
Contador de 4 bits

Especificaciones:

- 4 bits
- Reset
- Habilitación
- Carga
- Entradas sincrónicas
- Para la suma, usar un sumador de 4 bits con propagación de acarreo.

1- Definir arquitectura

Moore



2- Seleccionar códigos

Salida: binario 4 bits
Entrada: salida (de esta forma, la lógica de salida se simplifica)

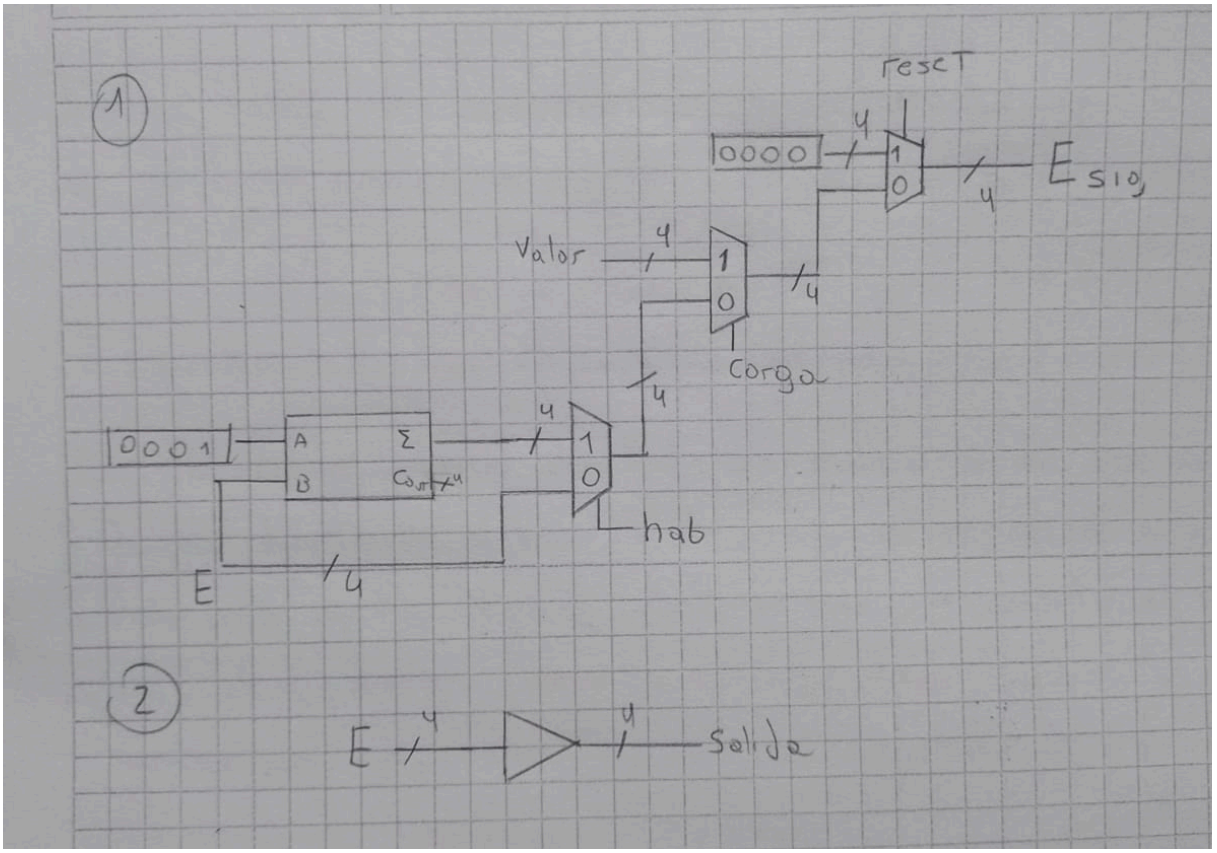
3- Determinar las expresiones aritmeticas y logicas para la evaluación del estado y la salida
Describo con VHDL, la lógica combinacional del estado siguiente:

Esig <= “0000” when reset else
valor when carga else
E + 1 when hab else
E;

Describo con VHDL, la lógica combinacional de salida:

Salida <= E;

4- Implemento las funciones lógicas mediante bloques combinacionales:


1
resct
0000
4
1
0
4
E sio
Valor
4
1
0
4
Corga
4
4
0001
A
Σ
B
Cout
4
1
0
hab
E
4
2
E
4
4
Salida

CONCLUSIONES

El desarrollo permitió afianzar los conceptos fundamentales de los **circuitos secuenciales síncronos** y su aplicación en el diseño de contadores digitales. El método basado en **máquinas de estado finito** resultó adecuado para circuitos de baja complejidad, al brindar una representación explícita del comportamiento del sistema y facilitar la obtención de las ecuaciones lógicas mediante técnicas de simplificación. Por otro lado, el enfoque basado en **datapath** demostró ser más apropiado para estructuras de mayor escala, ya que permite integrar bloques combinacionales reutilizables —como sumadores o registros— bajo el control de señales sincronizadas. En síntesis, la técnica de máquina de estados ofrece un diseño más didáctico y controlado, mientras que el datapath aporta una organización modular y eficiente, favoreciendo la implementación de sistemas más complejos y fácilmente expandibles.

REFERENCIAS

Floyd, T. L. (2006). *Fundamentos de sistemas digitales* (9.ª ed.). Pearson Education.

Wakerly, J. F. (2001). *Diseño digital: principios y prácticas* (3.ª ed.). Pearson Education.

Harris, D. M., & Harris, S. L. (2019). *Digital design and computer architecture, RISC-V edition*. Morgan Kaufmann.

Dally, W. J., Harting, R. C., & Aamodt, T. M. (2016). *Digital design using VHDL: A systems approach.* Cambridge University Press.