



Práctica

Presentación

Por fin llegamos al final del curso. Esta práctica servirá para sintetizar todos los conocimientos del curso y ampliarlos con el diseño de circuitos más complejos. En esta práctica se os pedirá el diseño de un circuito a partir de un grafo de estados y el diseño de una máquina de estados en concreto. Para la primera parte, tendréis que aplicar los conocimientos que habéis adquirido en este curso, como por ejemplo el de los sistemas de representación. Por lo tanto, se recomienda que le deis un vistazo.

Competencias

Conocer la organización general de un computador como circuito digital y conocer las características distintivas de la arquitectura de Von Neumann.

Objetivos

- Conocer varios modelos de máquinas de estados y las arquitecturas de controlador con camino de datos.
- Haber adquirido una experiencia básica en la elección del modelo de máquina de estados más adecuado para la resolución de un problema concreto.
- Ser capaz de diseñar circuitos secuenciales a partir de grafos de transiciones de estados.

Recursos

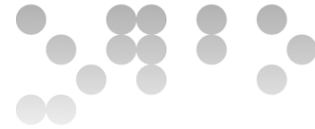
Los recursos que se recomienda usar para esta práctica son los siguientes:

Básicos: El módulo 5 de los materiales.

Complementarios: VerilCIRC, VerilCHART y el Wiki de la asignatura.

Criterios de valoración

- Razonad la respuesta en todos los ejercicios. Las respuestas que no estén justificadas no recibirán puntuación.
- La valoración de cada apartado está indicada en los enunciados correspondientes.



Formato y fecha de entrega

- Para dudas y aclaraciones sobre el enunciado, dirigiós al consultor responsable de vuestra aula.
- Hay que entregar la solución en un fichero PDF usando una de las plantillas entregadas conjuntamente con este enunciado.
- Se tiene que entregar a través de la aplicación de **Entrega y registro de EC** del apartado Evaluación de vuestra aula.
- La fecha límite de entrega es el **29 de Mayo** (a las 24 horas).

Ejemplo de solución

PRIMERA PARTE [65%]

Una de las cuestiones más importantes en las redes de comunicación es la seguridad, que se consigue, entre otras cosas, *encriptando* los mensajes que se transmiten. Esto dificulta que nadie que no tenga la clave para descifrarlos pueda acceder al contenido y que, además, se pueda determinar la autoría.

En un sistema criptográfico sencillo, los mensajes pueden ser textos que se codifican con una determinada clave de forma que resulten ilegibles para quién no tenga la clave. Para hacerlo, se pueden transformar los mensajes de texto en secuencias de códigos que sólo representen a las letras del alfabeto y, después, cifrar estas secuencias con una clave para esconder el mensaje a quien no la tenga.

La EFSM siguiente se corresponde con la de una máquina que recibe caracteres codificados en ASCII de 7 bits y los transforma en códigos de 5 bits de forma que sólo se codifican las letras del alfabeto inglés. (Los arcos no etiquetados indican transiciones incondicionales, que se efectúan siempre.)

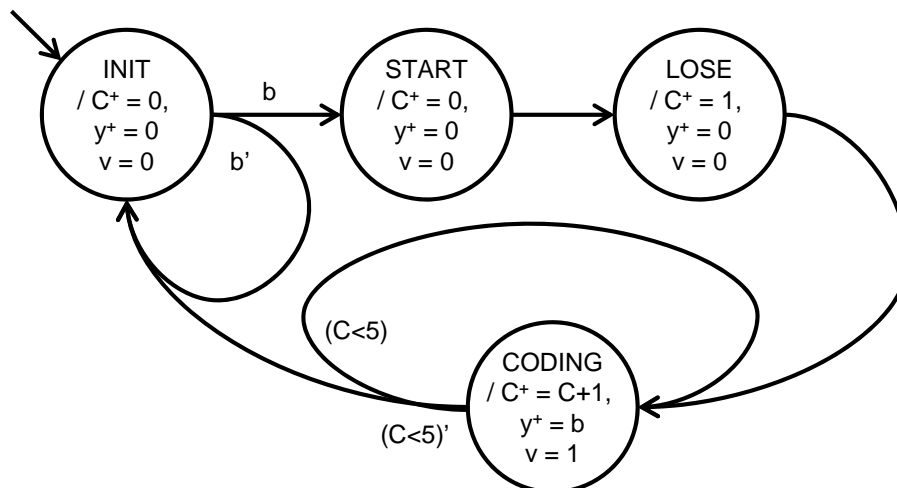
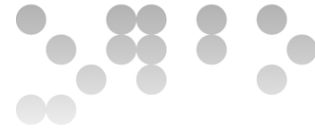


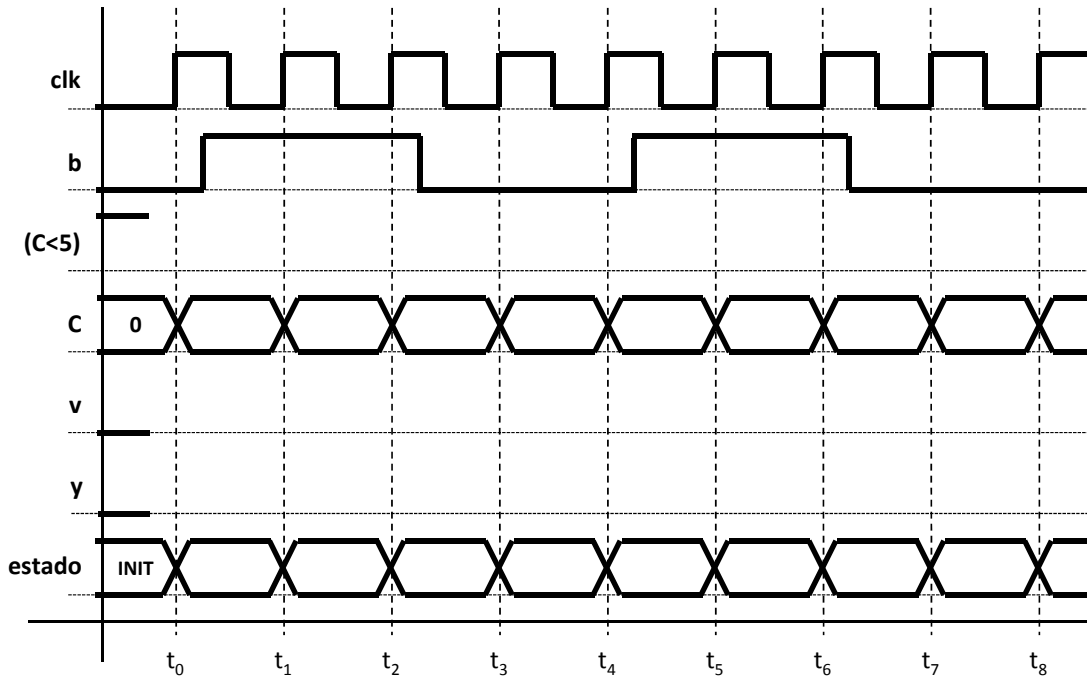
Fig. 1. EFSM del módulo de reducción de bits de la codificación ASCII.



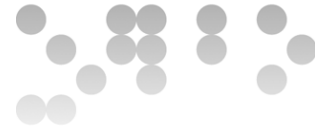
Se pide lo siguiente:

- a) [15%] A partir del grafo de la EFSM de la fig. 1 completad, en el cronograma siguiente, la evolución de las señales y , C y v , y del estado en que se encuentra la máquina en cada ciclo de reloj.

Nota: Tenéis el ejercicio disponible en VerilChart donde los valores de la señal C se representan en hexadecimal y las señales ($C < 5$) y *estado* están representadas por $C5$ y E .



- b) [15%] Obtened las tablas de transiciones y de salidas de la EFSM de la fig. 1, así como las codificaciones binarias de estados y salidas correspondientes.



La EFSM siguiente se corresponde con el funcionamiento de un encriptador sencillo que usa una clave entera de 5 bits, C , para codificar la secuencia de bits de entrada, y , siempre que sea válida, cosa que viene indicada por la señal de entrada v . La salida del módulo es una secuencia de bits z que cifra la secuencia de entrada y . Los bits de la secuencia z sólo serán correctos si la salida k es 1. Si no, indicará que no se está cifrando ningún mensaje

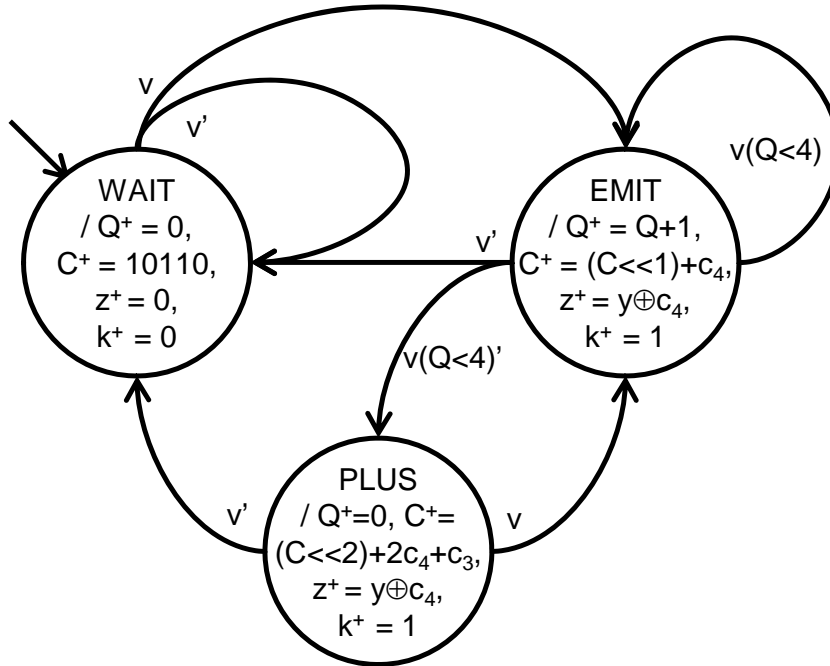


Fig. 2. EFSM de un módulo de encriptación con clave (C) de 5 bits.

En el grafo anterior, las sumas para el cálculo de C^+ son aritméticas.

Se pide lo siguiente:

- c) [15%] A partir de las tablas de transiciones y de salidas del grafo de la fig. 2 que se dan a continuación, implementad la unidad de control usando ROM y los registros, los bloques combinacionales y las puertas lógicas que creáis convenientes. Especificad y justificad las dimensiones de la/las ROM que uséis e indicad el contenido.

Estado actual	Entradas		Estado ⁺	Estado		Salidas			
	v	$(Q<4)$		Símbolo	q_1q_0	s_Q	s_C	s_z	s_k
WAIT	0	x	WAIT	WAIT	00	0	00	0	0
WAIT	1	x	EMIT	EMIT	01	1	01	1	1
EMIT	0	x	WAIT	PLUS	10	0	10	1	1
EMIT	1	0	PLUS						
EMIT	1	1	EMIT						
PLUS	0	x	WAIT						
PLUS	1	x	EMIT						

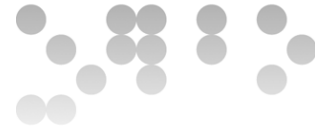


Hay que tener presente que las señales s_Q , s_C , s_z y s_k se usan como selectores para elegir qué resultados se asignan a las variables Q , C , z , y k respectivamente.

Nota: Tenéis el ejercicio disponible en VerilUOC y, si queréis comprobar previamente la ROM, la tenéis disponible en VerilChart. En ese entorno, las señales ($Q < 4$) y *Estado* son $Q4$ y E , respectivamente.

- d) [20%] Implementad el circuito completo de la EFSM de la fig. 2: Construid el camino de datos usando las puertas lógicas y los bloques necesarios e incorporad la unidad de control obtenida en el apartado anterior. Indicad y razonad, para cada bus, su dimensión en bits teniendo en cuenta que las señales de entrada ocupan el mínimo número de bits posible según su rango.

Nota: Tenéis el ejercicio disponible en VerilUOC. En el circuito, se han introducido las señales Q , C , $C4$ y E del apartado anterior como señales de salida para poder comprobar más fácilmente el correcto funcionamiento. Tened en cuenta que VerilUOC no soporta la verificación de subcircuitos diseñados por el usuario. Por lo tanto, deberéis copiar en vuestra solución el circuito de la unidad de control diseñada en el anterior ejercicio.



SEGUNDA PARTE [25%]

Un método para encriptar textos se basa en codificar cada carácter en otro según una operación con un carácter de una palabra clave. Una opción sencilla para ello es que el código correspondiente sea el carácter que esté en la posición que resulte de sumar las posiciones del carácter original y la del carácter de la palabra clave.

Por ejemplo, si el carácter a codificar es 'C' y el carácter de la palabra clave que se usa es 'K', según la tabla de caracteres de más abajo, el código resultante es 'N' porque ocupa la posición 15, que es el resultado de sumar la posición de la 'C' (3) y la de la 'K' (12).

Tabla 1. Ejemplo de tabla de codificación de caracteres.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
-	A	B	C	Ç	D	E	F	G	H	I	J	K	L	M
15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	.

Si el código resultante es mayor que 29, entonces se vuelve a empezar por el 0. Es decir, se suman las posiciones de los caracteres del texto y de la palabra clave en módulo 30.

Para cada nuevo carácter a codificar se usa un nuevo carácter de la palabra clave y, al llegar al final de la palabra clave, se vuelve al principio.

Un módulo encriptador que usara este método tendría que recibir el texto codificado tal como se muestra a la tabla 1 y emitir el texto cifrado con una palabra clave prefijada. El módulo empezaría a funcionar cada vez que recibiera la secuencia "---" y volvería al modo de espera cuando recibiera un punto o, si se quiere, cuando recibiera un 29. La salida sería la secuencia de caracteres cifrados o bien '.' (29).

Por ejemplo, el mensaje "UN-EJEMPLO." se vería codificado con la tabla como <0, 0, 0, 23, 15, 0, 6, 11, 6, 14, 18, 13, 17, 29>, de usar "CODIGO", es decir <3, 17, 5, 10, 8, 17>, como palabra clave, se obtendría el mensaje cifrado que se ve en la última fila de la tabla siguiente.

Tabla 2. Ejemplo de cifrado.

-	-	-	U	N	-	E	J	E	M	P	L	O	.	.
0	0	0	23	15	0	6	11	6	14	18	13	17	29	29
			3	17	5	10	8	17	3	17	5	10	8	
29	29	29	26	2	5	16	19	23	17	5	18	27	7	29
			X	B	D	Ñ	Q	U	O	D	P	Y	F	



Fijaos que también se codifica el punto final. Hay que tener en cuenta que, en la tabla anterior se muestra el cifrado en la misma columna, pero que, en la realidad, habría un desfase temporal entre la entrada y la salida equivalente.

Se pide que diseñéis una máquina de estados algorítmica que implemente este método de encriptación. Para ello, tened presente que:

- Recibe una entrada D de 5 bits que contiene el valor de la posición en la tabla 1 del carácter que representa;
- Tiene una memoria ROM de 8×5 bits con la palabra clave, que siempre es de ocho caracteres;
- Dispone de un registro de direcciones de memoria A y de una entrada de los datos de la memoria K , y
- Genera una salida W de 5 bits que representa cada uno de los caracteres cifrados o bien es '.' (posición 29) en caso contrario.

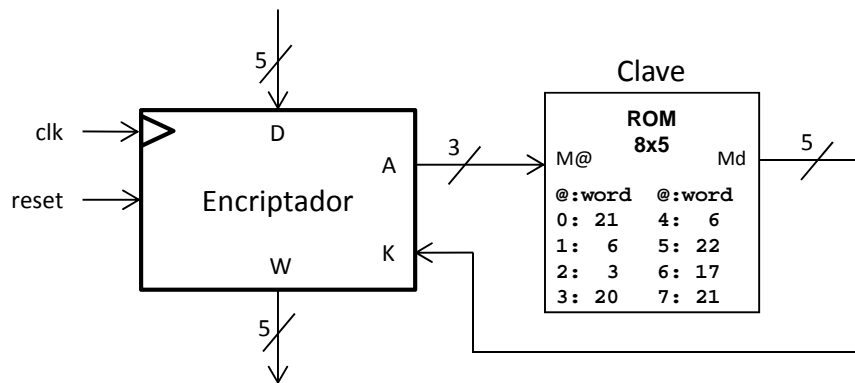


Fig. 3. Esquema de bloques de un encriptador de mensajes de texto.



TERCERA PARTE [10%]

Nota importante: Las dos preguntas de este apartado las encontraréis en el aula en la actividad Práctica. Hay que responderlas en los enlaces que se habilitarán en el aula.

Práctica

 Baixa en PDF
 Imprimeix
 Completada!

Inici: 05/05/17	Lliurament: 29/05/17	Solució: 03/06/17	Qualificació: 04/06/17
---------------------------	--------------------------------	-----------------------------	----------------------------------

[> Presentación](#)

[> Objetivos y competencias](#)

[> Contenidos y recursos](#)

[> Materiales y fuentes de información](#)

▼ Tercera Parte

En esta sección, durante el transcurso de la Práctica se mostrarán dos preguntas. Estas preguntas se tendrán que responder en el enlace que se proporcionará en esta sección.

NOTA IMPORTANTE: No se permite resolver estas preguntas en el documento de la Práctica. En el caso de no no firmar el consentimiento informado o no hacer las actividades de registro (enrollment), realizaréis las actividades sin ningún instrumento TeSLA activado.