ELECTRÓNICA DIGITAL **Primera convocatoria 2023-2024**

1	2	3	4

Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

TEORÍA (Cada pregunta vale 1 punto)

1.- Defina breve y claramente los siguientes términos:

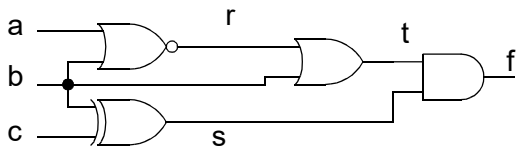
- I_{OH}
- Azar
- Codificador de prioridad
- Contador incompleto
- Full adder
- Microoperación
- Arquitectura Harvard
- Registro de instrucción

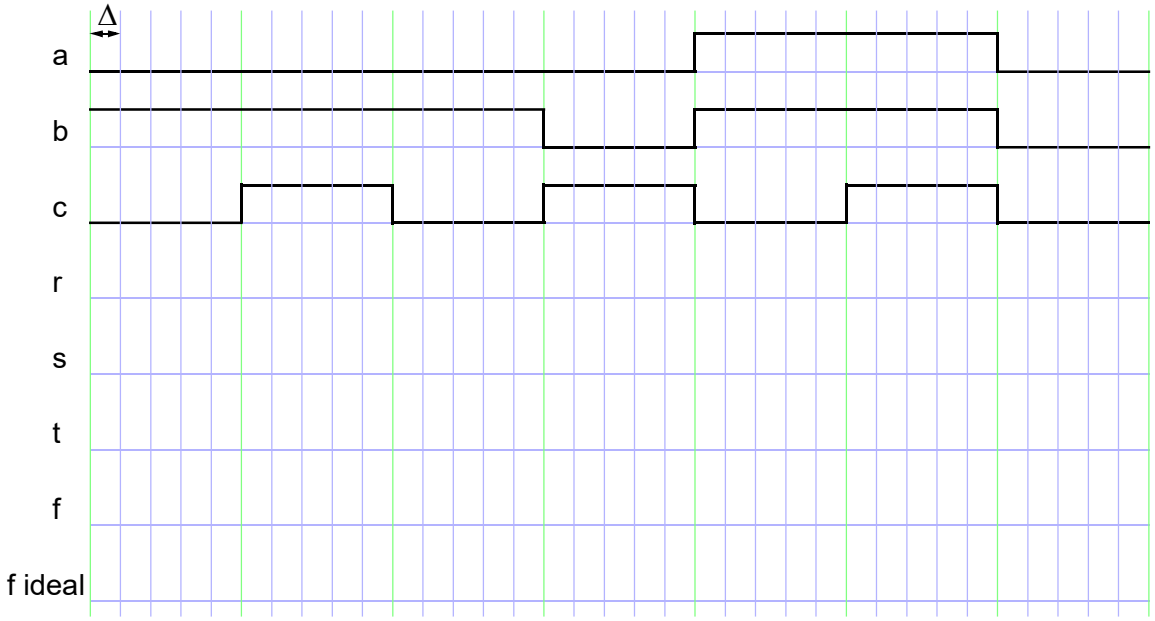
2.- Describa la función que realiza un demultiplexor, sus entradas y salidas y su tabla de funcionamiento. Ponga un ejemplo de cómo construir un demultiplexor mayor con otros más pequeños.

3.- Analice el circuito siguiente y complete el cronograma:

a) La salida y todos los nodos del circuito cuando las puertas tienen el mismo retraso Δ. Tenga en cuenta qué pasa con las señales para t=0.

b) La salida en el caso ideal.





ELECTRÓNICA DIGITAL **Primera convocatoria 2023-2024**

1	2	3	4

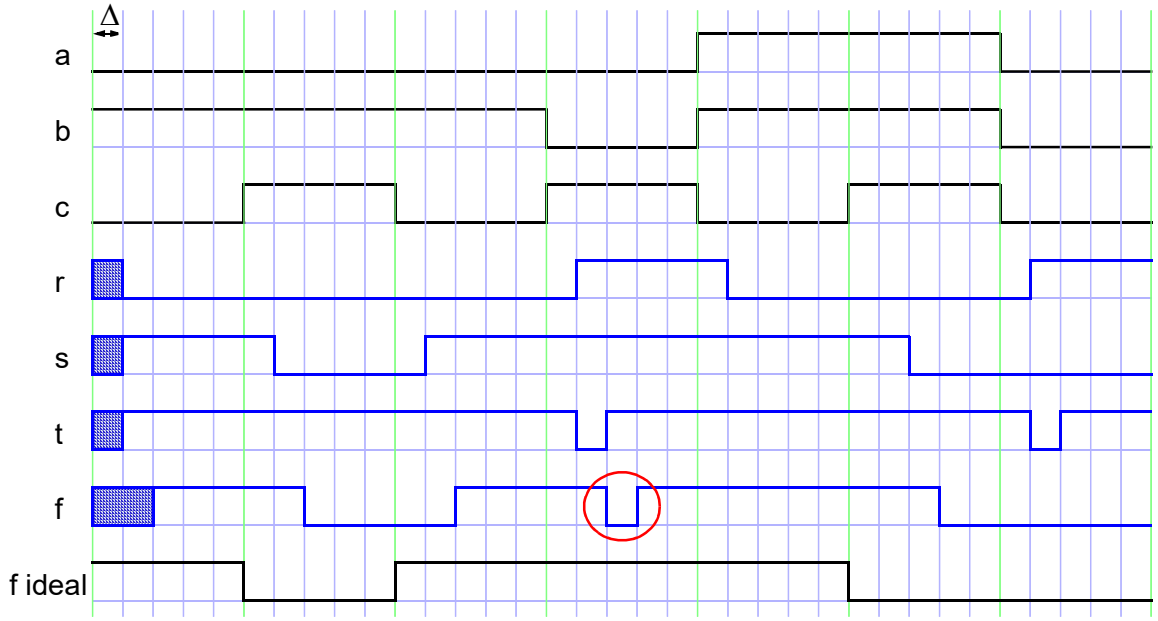
Apellidos:.....SOLUCIÓN.....

Nombre:.....

SOLUCIÓN

$$r = \overline{a+b} = \bar{a}\bar{b} \qquad s = b \oplus c = (b+c)(\bar{b}+\bar{c}) \qquad t = r+b$$

$$f = ts = (r+b)(b\bar{c}+\bar{b}c) = (\bar{a}\bar{b}+b)(b+c)(\bar{b}+\bar{c}) = (\bar{a}+b)(b+c)(\bar{b}+\bar{c}) = \bar{a}\bar{b}c+b\bar{c}$$



CRITERIO DE CORRECCIÓN

- Análisis y obtención de f mínima: 4
- Cada señal del cronograma: 1 (5 en total)
- Análisis en t=0: 1

4.- Para un procesador, explique los tipos de direccionamiento de memoria que conozca.

SOLUCIÓN

Hay que explicar cómo funcionan los siguientes modos de direccionamiento:

- Directo de registro, inmediato, directo de memoria, indirecto de registro, indirecto de registro con post-incremento, indirecto de registro con pre-decremento, indirecto de registro con offset y relativo a PCo SP.

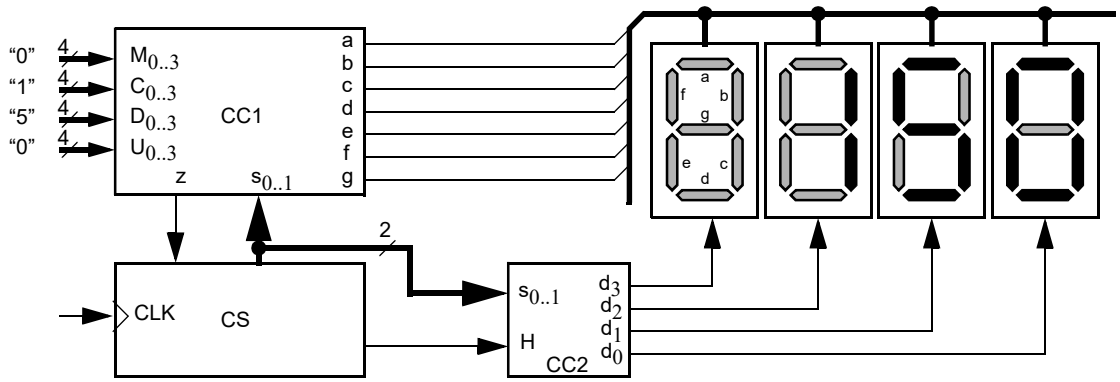
Se puede conseguir la máxima puntuación explicando únicamente 5 de ellos.

CRITERIO CORRECCIÓN

2 puntos por cada modo. Si sólo pone el nombre un punto.

PROBLEMAS (Cada pregunta vale 3 puntos)

1.- En los dispositivos electrónicos, con frecuencia se desea poder visualizar una información numérica. Una solución bastante habitual consiste en usar visualizadores de 7 segmentos. Cada uno de dichos elementos es capaz de representar un dígito decimal. Para simplificar la circuitería en la representación de cantidades con muchos dígitos, se suele usar un diseño multiplexado. El procedimiento consiste en visualizar los dígitos, de uno en uno, alternativamente y en rápida sucesión. Así se consigue la ilusión de que están encendidos todos simultáneamente. Además, y para mayor claridad, se eliminan de la visualización los ceros no significativos (por ejemplo, se visualiza "__50" y no "0050"; se visualiza "___0" y no "0000"; ver figura). El circuito se corresponde con el siguiente esquema:



El circuito se compone de tres subcircuitos: CS, CC1 y CC2. CS es el controlador del sistema, CC1 es, básicamente un decodificador de siete segmentos y CC2 un seleccionador de visualizador (los visualizadores disponen, además de las entradas correspondientes a cada segmento, una línea de habilitación).

CC1 es un circuito combinacional que tiene por entradas cuatro dígitos BCD ($M_{0..3}$, $C_{0..3}$, $D_{0..3}$ y $U_{0..3}$) y un par de líneas de selección de dígito ($s_{0..1}$). Las salidas corresponden con la decodificación a siete segmentos (a, b, c, d, e, f y g) del BCD seleccionado y una salida que se activa cuando dicho dígito es cero (z = 1).

CC2 dispone de las entradas $s_{0..1}$ y H y cuatro salidas. Se adjunta la tabla de verdad del funcionamiento de este dispositivo.

Hs_1s_0	$d_3d_2d_1d_0$
0xx	0000
100	0001
101	0010
110	0100
111	1000

Por último, CS es un Circuito Secuencial y se encarga de llevar el control del visualizador. Su misión es activar cada uno de los dígitos por turnos. Dotado de una entrada de reloj de 50 Hz, debe seleccionar inicialmente el dígito BCD de mayor peso, M, (actuando sobre s_1 y s_0). En el siguiente ciclo de reloj seleccionará el dígito C, en el siguiente el D, después el U, después el M, etc. Este módulo dispone de una entrada, z, que le informa de si el dígito actual vale 0 (z=1) y de una salida H que controla a CC2 (para eliminar los ceros no significativos cuando H = 0).

Sabiendo que se disponde de cuatro multiplexores de 4 a 1, una ROM de 16 bytes y un decodificador de 2 a 4 con entrada de habilitación, se pide:

a) **[Punt 40%]** Obtener un diseño de CC1.
b) **[Punt 10%]** Obtener un diseño de CC2.
c) **[Punt 50%]** Diagrama de estados mínimo de una máquina de Mealy para CS.

SOLUCIÓN

a) Las misiones encomendadas a CC1 son:

- Realizar la multiplexión de los cuatro dígitos BCD, seleccionados por $s_{0..1}$.

ELECTRÓNICA DIGITAL **Primera convocatoria 2023-2024**

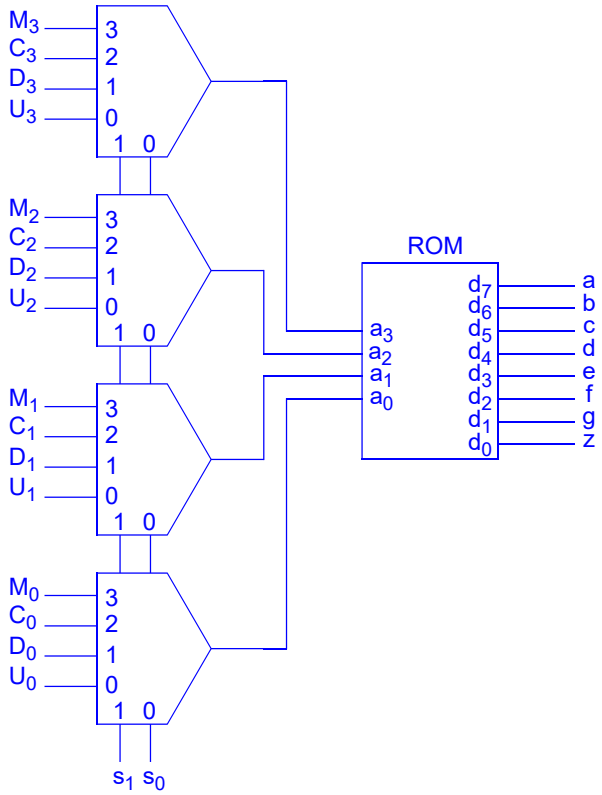
1	2	3	4

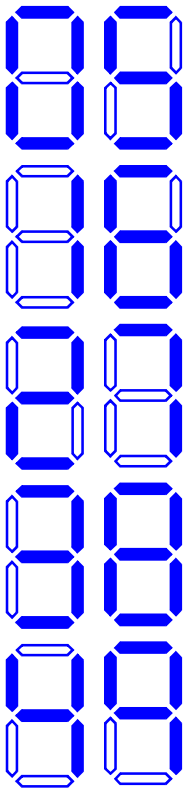
Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

- Realizar la decodificación BCD a siete segmentos y la detección del valor 0.

Con los componentes que nos dan, lo mejor es usar la ROM para decodificación de 7 segmentos y la detección del cero (la memoria dispone de 4 entradas y 8 salidas), y los multiplexores para la multiplexión.



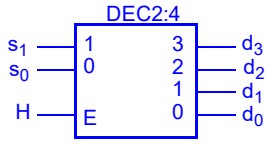


El contenido de la ROM será (en función de los dibujos de los dígitos en el visualizador):

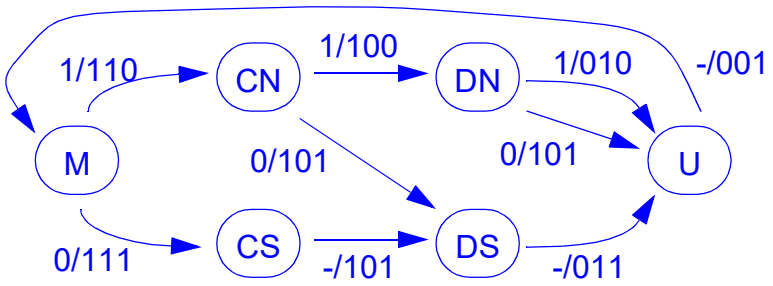
$a_3a_2a_1a_0$	$d_7d_6d_5d_4d_3d_2d_1d_0$
0000	11111101
0001	01100000
0010	11011010
0011	11110010
0100	01100110
0101	10110110
0110	10111110
0111	11100000
1000	11111110
1001	11100110
1010	00000000
1011	00000000
1100	00000000
1101	00000000
1110	00000000
1111	00000000

b) El diseño de CC2 coincide con el de un decodificador de 2 a 4 con salidas activas en alto y entrada de habilitación activa en alto. Usaremos el decodificador que nos

proporcionan y ya que no nos dicen sus características, hagámoslas iguales a las mencionadas.



c) La labor de CS es la de seleccionar secuencialmente los distintos dígitos BCD (por la posición ocupada en el MUX, desde la 0 a la 3) y, en función de si los dígitos más significativos son 0 o no, activar H o no. Hay que notar que se comienza por el dígito más significativo, el M. Si éste es cero, no se activa H. Si, en cambio, es no nulo, se debe activar H. El proceso se repite para los demás dígitos, teniendo en cuenta que si se ha visualizado un determinado dígito, se visualizarán todos los de menor peso independientemente de z. Por último, la excepción es el dígito de las unidades que se visualiza siempre. Así se puede notar que la máquina hace ciclos de cuatro estados, recorriendo desde el dígito M hasta el U. Los dígitos C y D tienen dos estados cada uno, para poder recordar si se están visualizando estos dígitos (CS y DS) o no (CN y DN). Notar también, que si se desactiva la visualización (H=0), daría igual qué dígito se seleccione a efectos del display; sin embargo, hay que mantener su código, ya que la salida z también depende de s_1 y s_0 para generar z.



Si intentamos minimizar al máquina de estados, podemos ver, por simple inspección que la máquina es mínima, ya que no hay dos estados con las mismas salidas.

CRITERIO DE CORRECCIÓN

a) 50% (MUX 4, ROM 6)
b) 10%
c) 40%

2.- Sea la operación SHUFFLE (x, y) que mezcla los bits de x e y, dejando los bit impares de x y los pares de y. Es decir, si x e y tienen 4 bits, SHUFFLE (x, y) devuelve $x_3y_2x_1y_0$, mientras que SHUFFLE (y, x) devuelve $y_3x_2y_1x_0$.

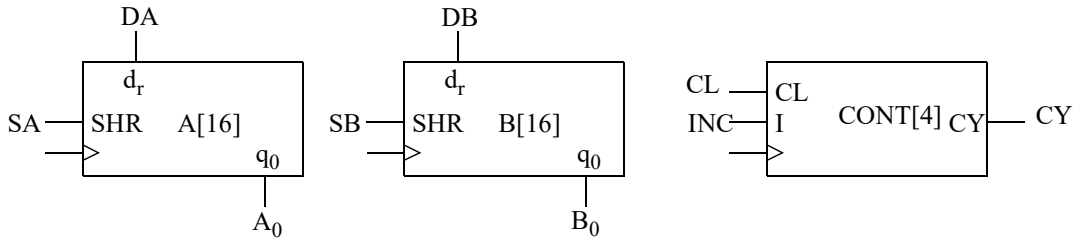
a) Obtenga las cartas ASM de datos y control de una UC para la UD de la figura que haga la instrucción A↔B (A y B intercambian sus contenidos) cuando I=0 y A←SHUFFLE

1	2	3	4

Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

(A, B) y B←SHUFFLE (B, A) cuando I=1.



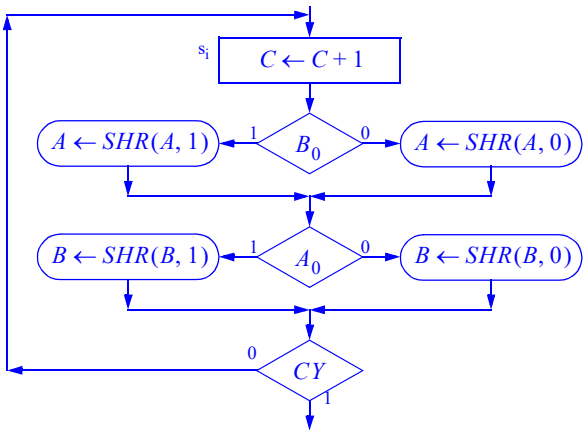
b) Escriba la subrutina swap en ensamblador del CS3 que realiza la operación R0 ↔ R1. Para hacer una versión lo más óptima posible, se recomienda no usar las instrucciones de desplazamiento.

c) Escriba la subrutina shuffle en ensamblador del CS3 que devuelve en R2 la operación SHUFFLE (R0, R1).

SOLUCIÓN

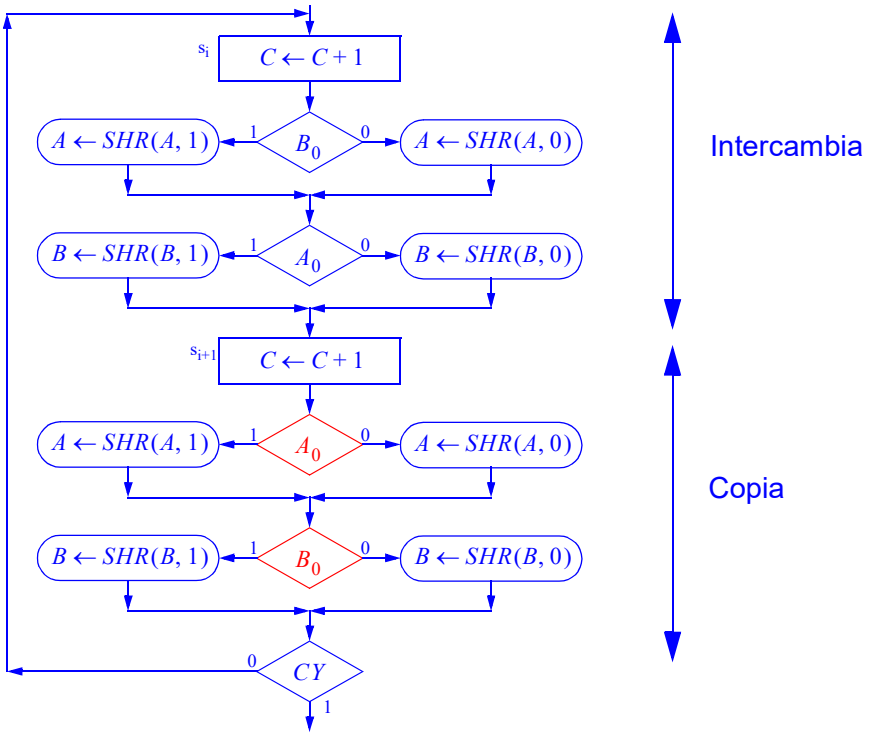
a) $\bar{I}$: A↔B; I: A ← SHUFFLE(A,B); I: B ← SHUFFLE(B,A)

Para intercambiar el contenido de los dos registros, basta preguntar por lsb del otro registro e introducirlo por la izquierda al hacer el desplazamiento a la derecha. Dicho desplazamiento deja expuesto el siguiente bit en el lsb para la siguiente iteración.



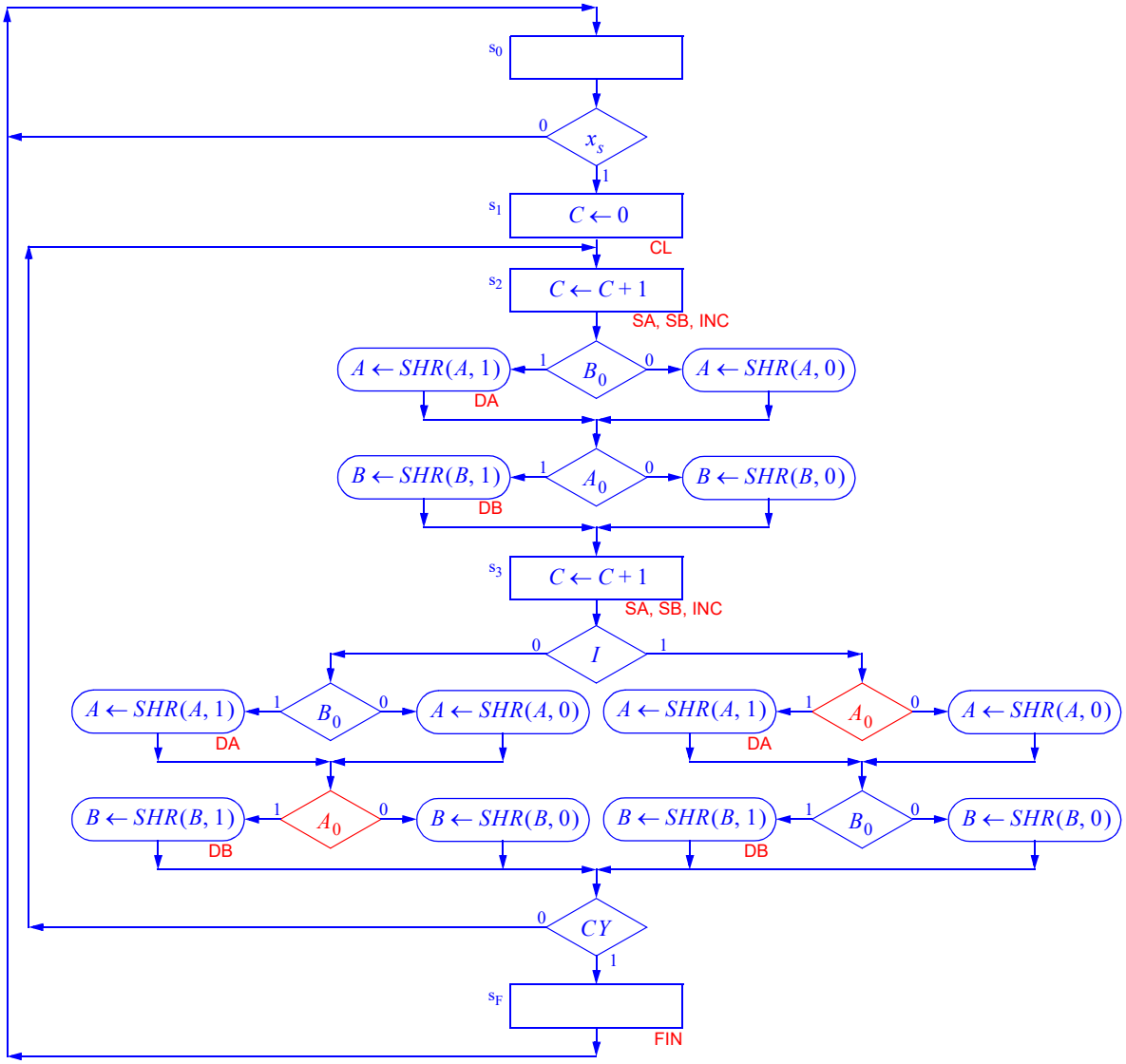
La operación de mezcla es similar, ya que intercambia los bits pares, pero para los bits impares los deja sin alterar. Eso supone un cambio en el bucle de la carta ASM: habrá un

estado que intercambie (s_i) y otro que copie (s_{i+1}). Se han resaltado en rojo los cambios entre los dos estados:



La versión final de la carta ASM fusiona esos dos comportamientos en función de la entrada I. Si I=0, hay que hacer 16 iteraciones del bucle mostrado en la primera figura. SI I=1, 8 iteraciones del bucle de la segunda figura, ya que procesa 2 bits en cada vuelta, uno para

intercambiarlo y otro para copiarlo. El diseño optimiza el número de estados para simplificar la UC, llevando el intercambio de A y B al segundo modelo (2 bits por iteración).



b) La opción más eficiente es usar las instrucciones de movimiento de registros (MOV). Se necesita un registro auxiliar (R2) para hacer el intercambio de R0 y R1. A efectos de la explicación, supongamos que R0=A y R1=B.

```
;-------------------------------------------------------------------------------
; swap                                                                  v1.0
;
; Intercambia el contenido de R0 y R1.
; Entradas:
;   R0 y R1
; Salidas:
;   R0 y R1
;-------------------------------------------------------------------------------
swap:       mov         r2, r1      ;R2=B
            mov         r1, r0      ;R1=A
            mov         r0, r2      ;R0=B
            ret
```

c) Para mezclar los registros R0 y R1 en R2 tal como se indica, se procederá a hacer un bucle que dé 4 vueltas. En cada una de ellas, se copiará un bit de R1 y uno de R0 mediante desplazamientos a la derecha tanto de los registros fuentes como del destino.

ELECTRÓNICA DIGITAL **Primera convocatoria 2023-2024**

1	2	3	4

Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

Todos los registros son desplazados 2 veces en cada iteración. Se usará el registro R2 simultáneamente como registro destino y como variable de control de bucle que cuenta las iteraciones. Para ello se carga con el número 128 (10000000_2), de tal forma que, después de 8 desplazamientos a la derecha, el msb sale y se coloca en el carry, evento que es detectado por la instrucción de salida.

```
;-------------------------------------------------------------------------------------
; shuffle                                                                      v1.0
;
; Baraja R0 y R1
; Entradas:
;   R0 y R1
; Salidas:
;   R2 = SHUFFLE (R0, R1)
;-------------------------------------------------------------------------------------
shuffle:    ldi       r2,128     ;R2=1000 0000. Contador del bucle y registro salida
shuffleBuc: ror       r0         ;Desplazar a la derecha R0. LSB de R0 a C
            ror       r1         ;Desplazar a la derecha R1. LSB de R1 a C
            ror       r2         ;Desplazar a la derecha R2. Entra C por izquierda
            ror       r1         ;Desplazar a la derecha R1. LSB de R1 a C
            ror       r0         ;Desplazar a la derecha R0. LSB de R0 a C
            ror       r2         ;Desplazar a la derecha R2. Entra C por izquierda
            brcs      shuffleFin;C=1?, hemos terminado
            jmp       shuffleBuc;...no, otra vuelta más
shuffleFin: ret
```

CRITERIO DE CORRECCIÓN

a) 50%
b) 20%
c) 30%