ELECTRÓNICA DIGITAL **Segunda convocatoria 2024-2025**

1	2	3	4

Apellidos:.....**SOLUCIÓN**.....

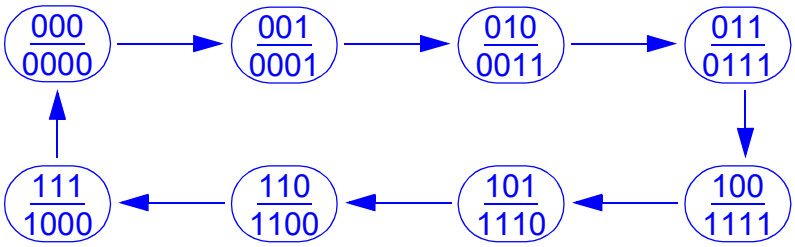
Nombre:.....

TEORÍA (Cada pregunta vale 1 punto)

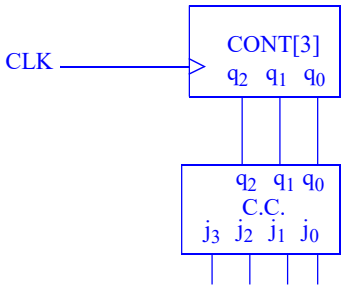
1.- Diseñe un contador Johnson módulo 8 (que sigue la secuencia 0000, 0001, 0011, 0111, 1111, 1110, 1100, 1000) a partir de un contador ascendente módulo 8 y puertas NAND.

SOLUCIÓN

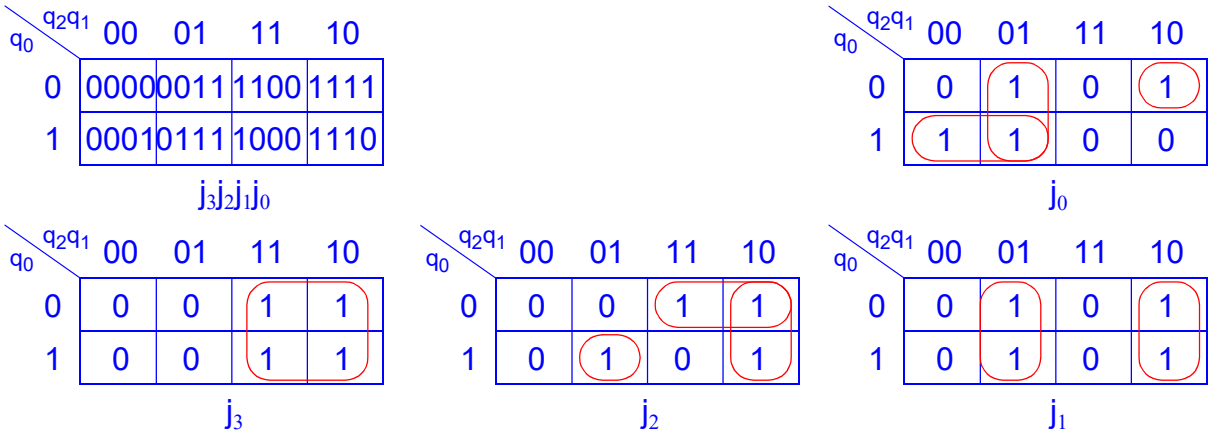
Aunque el contador Johnson tiene 4 salidas, su módulo es 8, por lo que se puede realizar con un contador ascendente natural módulo 8. A las salidas de dicho contador se acoplará un circuito combinacional que se encargue de hacer el cambio de código natural/Johnson.



La estructura del circuito sería:



Los mapas-K del circuito combinacional son:



Minimizando por los unos, complementando dos veces y aplicando DeMorgan:

$$j_3 = q_2$$

$$j_2 = q_2\overline{q_1} + q_2\overline{q_0} + \overline{q_2}q_1q_0 = \overline{\overline{q_2\overline{q_1} + q_2\overline{q_0} + \overline{q_2}q_1q_0}} = \overline{\overline{q_2\overline{q_1}}\;\overline{q_2\overline{q_0}}\;\overline{\overline{q_2}q_1q_0}}$$

$$j_1 = \overline{q_2}q_1 + q_2\overline{q_1} = \overline{\overline{q_2}q_1 + q_2\overline{q_1}} = \overline{\overline{\overline{q_2}q_1}\;\overline{q_2\overline{q_1}}}$$

$$j_0 = \overline{q_2}q_1 + \overline{q_2}q_0 + q_2\overline{q_1}\,\overline{q_0} = \overline{\overline{q_2}q_1 + \overline{q_2}q_0 + q_2\overline{q_1}\,\overline{q_0}} = \overline{\overline{\overline{q_2}q_1}\;\overline{\overline{q_2}q_0}\;\overline{q_2\overline{q_1}\,\overline{q_0}}}$$

ELECTRÓNICA DIGITAL **Segunda convocatoria 2024-2025**

1	2	3	4

Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

Sólo queda dibujar el circuito.

CRITERIO DE CORRECCIÓN

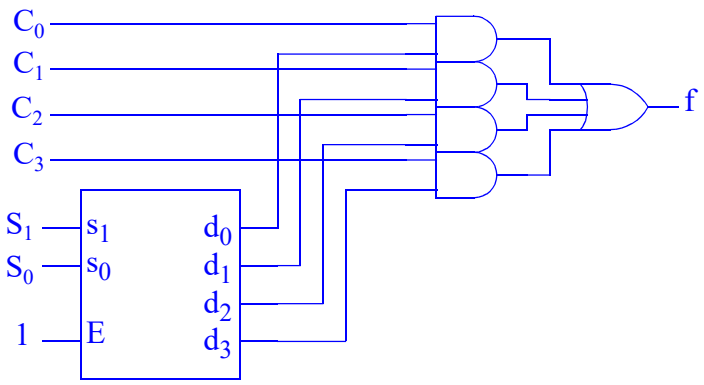
2.- Explique de forma concisa las similitudes y diferencias de las memorias ROM, PROM, EPROM y EEPROM.

SOLUCIÓN

CRITERIO DE CORRECCIÓN

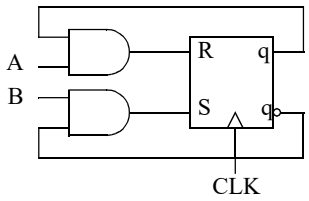
3.- Realice un MUX4:1 si dispone de un DEC2:4, puertas AND de 2 entradas y OR de 4.

SOLUCIÓN



CRITERIO DE CORRECCIÓN

4.- Analice el circuito siguiente y obtenga el diagrama de estados. ¿Qué hace el circuito?



SOLUCIÓN

Ecuaciones de excitación/salida:

$$R = Aq \qquad S = B\bar{q}$$

Tablas de excitación/salida:

q \ AB	00	01	11	10
0	00	10	10	00
1	00	00	01	01

SR

Tabla de transición/salida:

q \ AB	00	01	11	10
0	0	1	1	0
1	1	1	0	0

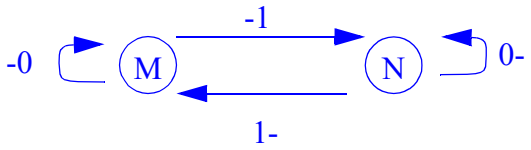
Q

Tabla de estados:

S \ AB	00	01	11	10
M	M	N	N	M
N	N	N	M	M

NS

Diagrama de estados:

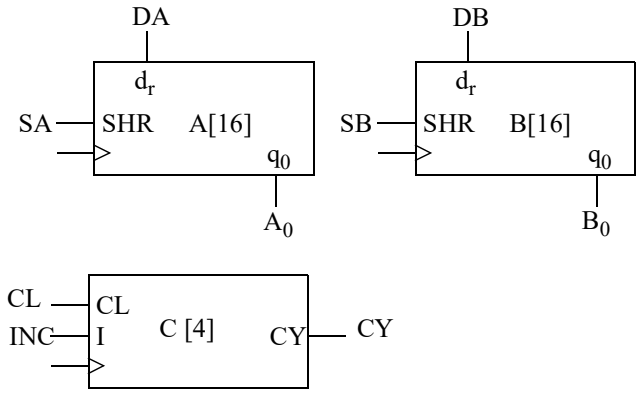


Se trata de un biestable JK en el que J=B y K=A.

CRITERIO DE CORRECCIÓN

PROBLEMAS (Cada pregunta vale 3 puntos)

1.- Para la unidad de datos de la figura:



Restador completo

Ci A B	Co R
0 0 0	0 0
0 0 1	1 1
0 1 0	0 1
0 1 1	0 0
1 0 0	1 1
1 0 1	1 0
1 1 0	0 0
1 1 1	1 1

a) Obtenga las cartas ASM de datos y control de una UC que, dados dos números sin signo en A y B, deja en A el valor absoluto de la diferencia entre A y B. B queda inalterado.

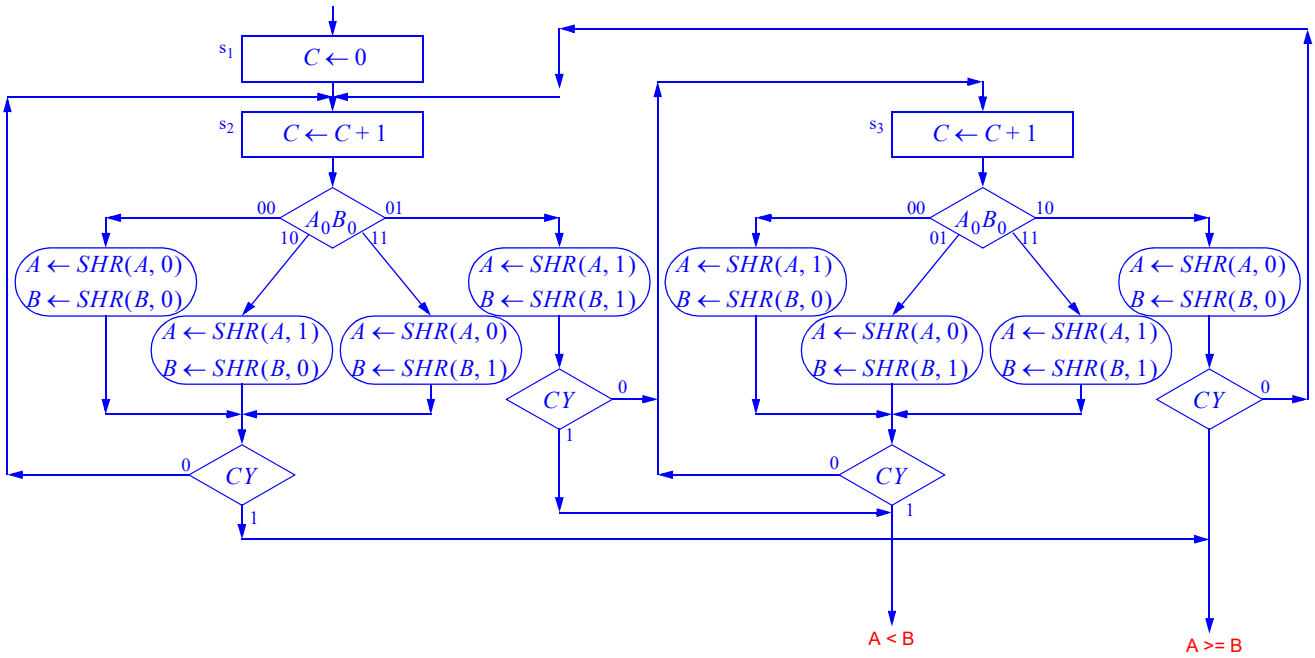
b) Realice la implementación de la UC usando la técnica de un biestable por estado.

SOLUCIÓN

a) Para calcular el valor absoluto de la resta se puede proceder de dos formas. La primera restar A-B y evalúar el último carry de salida. Si es 0, el resultado es positivo y no hay que hacer más. Si el último carry es 1, el resultado es negativo y hay que cambiar el signo al mismo. La segunda requiere averiguar cuál de los dos números es mayor y después restar el menor al mayor. Dicha comparación no es trivial en la UD

proporcionada, ya que no se tiene acceso al msb. Por ello, la comparación debe hacerse mediante una resta A-B (o B-A), que sí se puede hacer accediendo al lsb, y que no modifique los registros. Observando el último carry de salida, sabemos qué número es mayor (si c=0, A; en caso contrario, B). Una vez que sabemos cuál es mayor, sólo queda hacer la resta A-B o B-A según el carry de salida. Parece mejor la primera opción, ya que en un 50% de las veces no hay que hacer nada después de la resta y en el otro 50% sólo hay que cambiar el signo a A con una operación Ca2.

La resta se hace con un par de estados (s_2 y s_3) que implementan cada uno una mitad de la tabla de verdad del restador completo. s_2 recuerda que el carry de entrada es 0 y s_3 que es 1. En cada estado se calcula la resta (que se inserta en A) y el carry de salida (que determina el próximo estado) y se reinserta B_0 para conservar el registro. Después de 16 iteraciones el último carry indica si $A \geq B$ (c_O=0) o $A < B$ (c_O=1) y el contador vuelve a estar a 0, listo para cambiar el signo si hace falta.



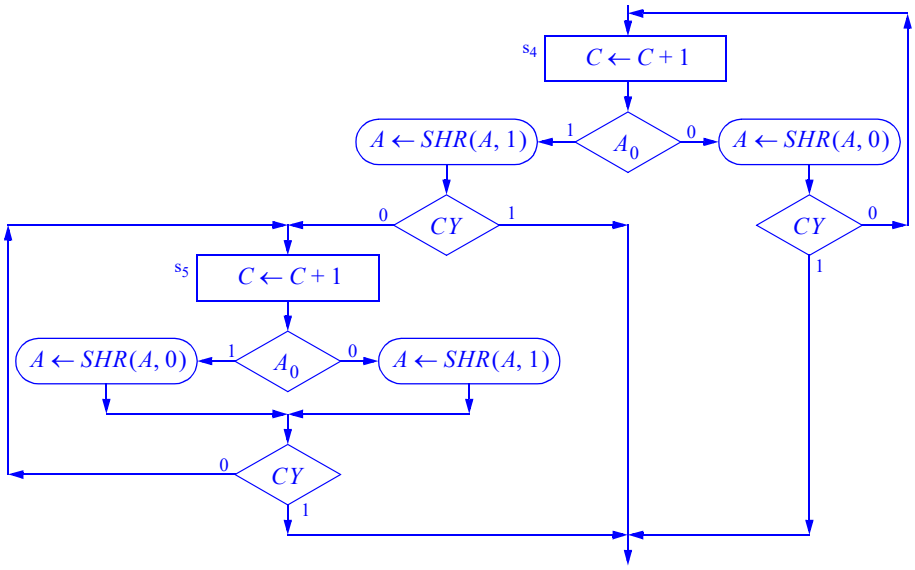
Para cambiar el signo de A, se calcula el Ca2 del mismo. La forma más sencilla de hacerlo en un registro serie-serie en el que sólo se accede al lsb, es copiando los bits que salgan por la

1	2	3	4

Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

derecha hasta que se copie el primer 1 (estado s_4). Después invertir todos los demás (estado s_5).



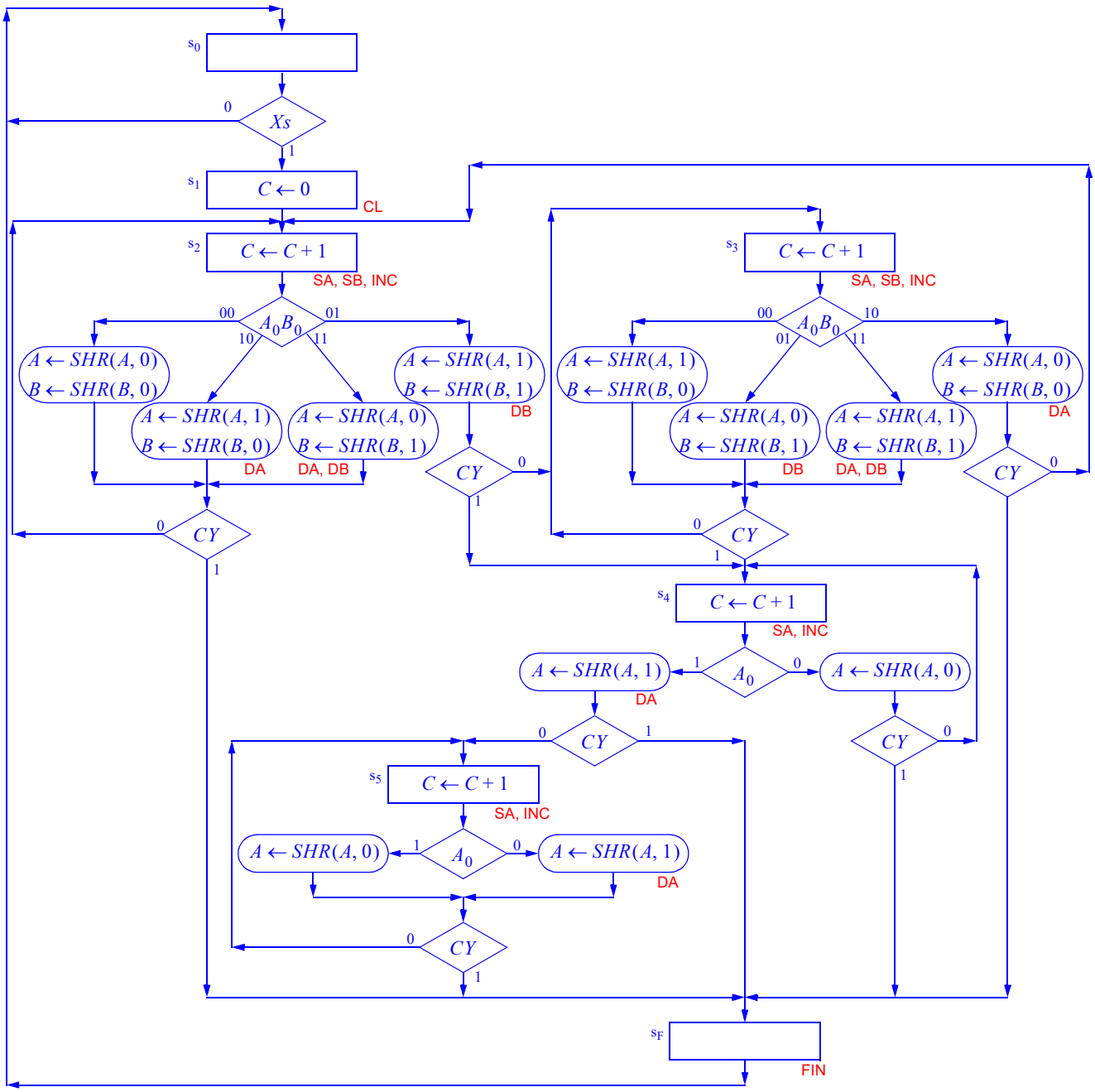
ELECTRÓNICA DIGITAL **Segunda convocatoria 2024-2025**

1	2	3	4

Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

Uniéndolo todo nos queda (en rojo los datos de la carta ASM de la UC):



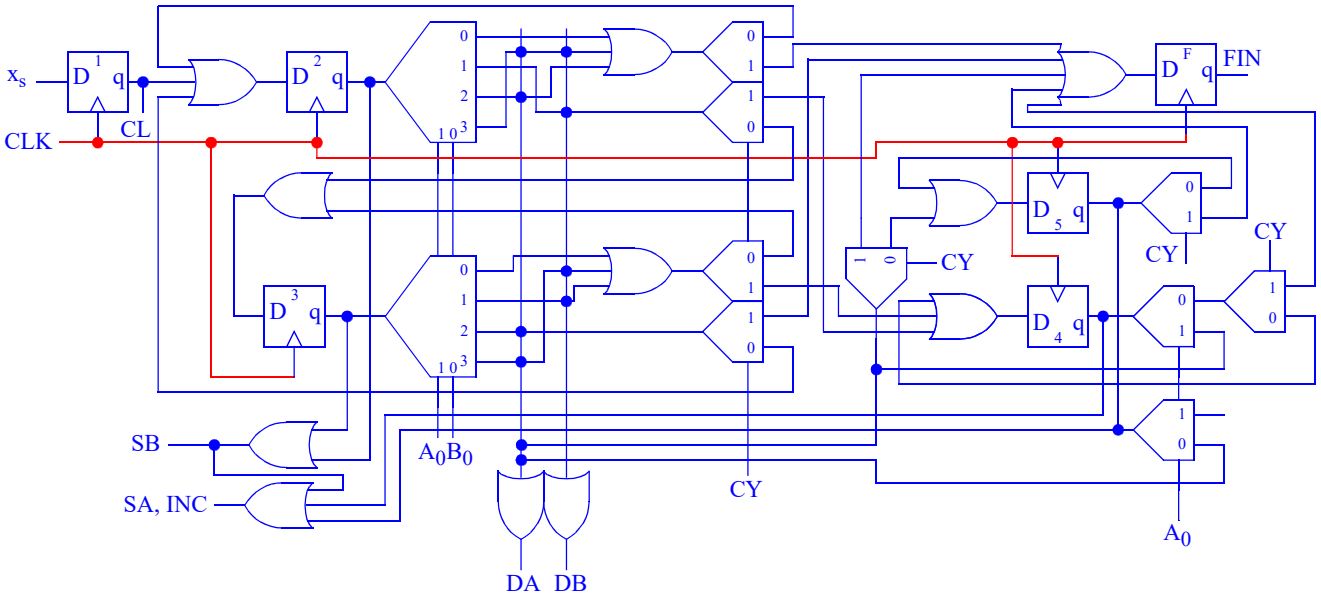
ELECTRÓNICA DIGITAL **Segunda convocatoria 2024-2025**

1	2	3	4

Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

Implementación de la unidad de control:



CRITERIO DE CORRECCIÓN

a) 40% Resta
b) 20% Complemento a 2
c) 40% Implementación

ELECTRÓNICA DIGITAL **Segunda convocatoria 2024-2025**

1	2	3	4

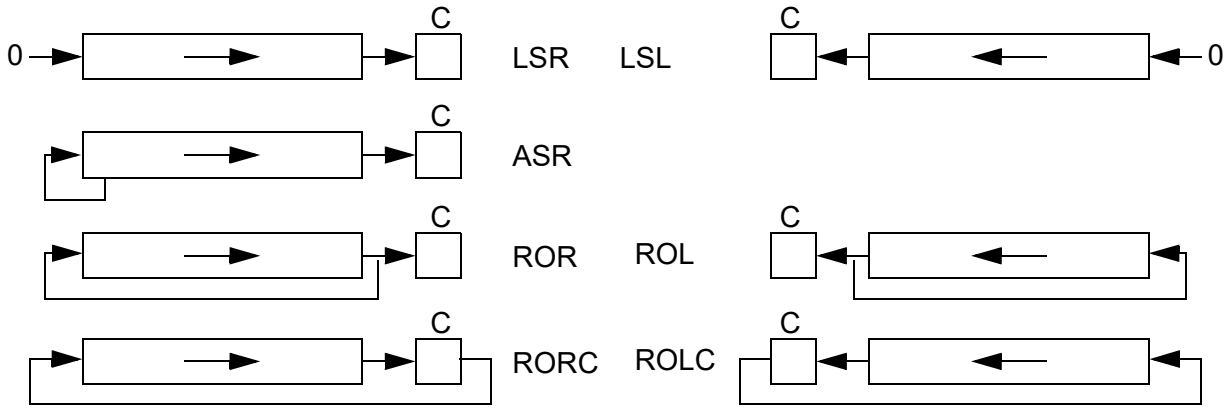
Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

2.- En CS3 existen dos instrucciones lógicas que realizan desplazamientos a derecha (ROR) e izquierda (ROL) basándose en las operaciones 0100 (SHR) y 0101 (SHL) de la ALU. Se desea completar el repertorio de desplazamientos de la ALU a las siguientes (las ya implementadas cambian de nombre y código: ROR es ahora RORC y ROL, ROLC). Sólo se diferencian en qué bit se utiliza para rellenar la salida después del desplazamiento (0, A_7, A_0 y C_{IN} para los desplazamientos a la derecha y 0, -, A_7 y C_{IN} para los desplazamientos a la izquierda).

Operación	A la derecha			A la izquierda		
	$OP_{3:0}$	F=	Nemónico	$OP_{3:0}$	F=	Nemónico
Desplazamiento lógico	0000	SHR (A, 0)	LSR	0100	SHL (A, 0)	LSL
Desplazamiento aritmético	0001	SHR (A, A_7)	ASR	0101	-	-
Rotación	0010	SHR (A, A_0)	ROR	0110	SHL (A, A_7)	ROL
Rotación con carry	0011	SHR (A, C_{IN})	RORC	0111	SHL (A, C_{IN})	ROLC

Gráficamente podemos describir las operaciones de la siguiente forma:



a) Diseñe la nueva unidad de desplazamientos de la ALU usando puertas y subsistemas combinacionales no programables. En todos los casos, el carry de salida se carga con el bit saliente.

b) Dado que las operaciones de la ALU no están disponibles actualmente en el CS3 y que no existen las correspondientes instrucciones, escriba las subrutinas LSR y LSL en ensamblador que realizan, respectivamente, las operaciones LSR y LSL sobre el registro R0.

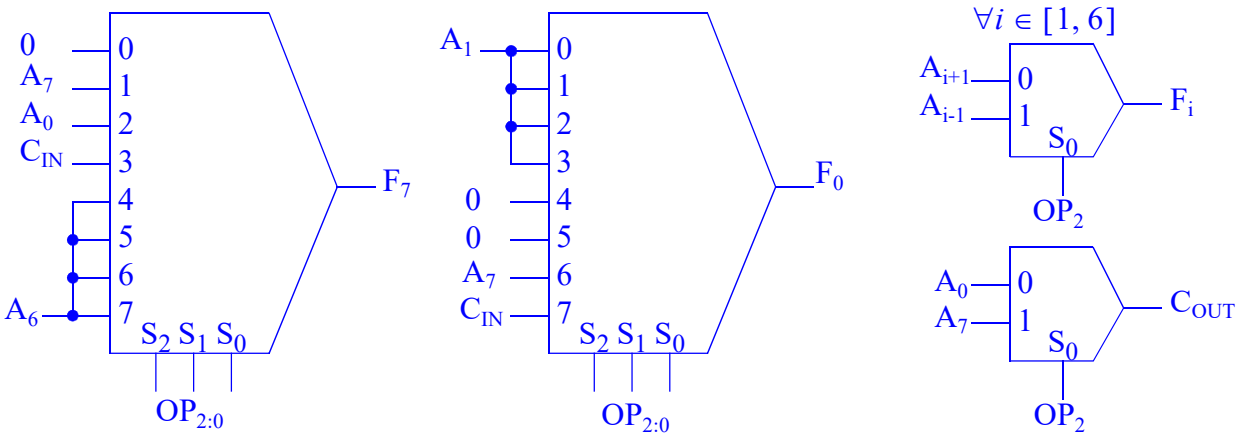
c) Idem para ASR.

d) Ídem para ROR y ROL.

SOLUCIÓN

a) Los bits centrales de las salida, del 1 al 6, sólo dependen de la dirección del desplazamiento (OP_2) y se puede implementar con un simple MUX2:1 para cada salida. Los bits de los extremos son más complejos al necesitar un MUX8:1 cada uno. El de F_7 se conecta a las entradas que aparecen en la macro SHR de la tabla del enunciado para los códigos 0 a 3 (que corresponden a desplazamientos a la derecha) y a A_6 para los desplazamientos a la izquierda. De forma análoga, las entradas del MUX de F_0 corresponden con A_1 para los códigos 0 a 3 (derecha) y las entradas de la tabla del enunciado para los códigos 4 a 7 (izquierda). La entrada 5 no está definida

(desplazamiento aritmético a la izquierda), por lo que se ha dejado a 0. Por último, el carry de salida C_{OUT}, se carga con el lsb en los desplazamientos a la derecha (A_0) y con el msb en los desplazamientos a la izquierda (A_7)



b)

```
;-------------------------------------------------------------------------------
; LSR                                                                   v1.0
;-------------------------------------------------------------------------------
LSR:        clc                 ;Carry a 0
            ror       r0        ;Desplazar a la derecha insertando C=0 por la izqda
            ret
;-------------------------------------------------------------------------------
; LSL                                                                   v1.0
;-------------------------------------------------------------------------------
LSL:        clc                 ;Carry a 0
            rol       r0        ;Desplazar a la izqda insertando C=0 por la dcha
            ret
```

c)

```
;-------------------------------------------------------------------------------
; ASR                                                                   v1.0
;-------------------------------------------------------------------------------
ASR:        mov       r1, r0    ;R1=R0
            rol       r1        ;msb al carry
            ror       r0        ;Desplazar a la derecha insertando C=msb por la izqda
            ret
```

d)

```
;-------------------------------------------------------------------------------
; ROR                                                                   v1.0
;-------------------------------------------------------------------------------
ROR:        mov       r1, r0    ;R1=R0
            ror       r1        ;lsb al carry
            ror       r0        ;Desplazar a la dcha insertando C=lsb por la izqda
            ret
;-------------------------------------------------------------------------------
; ROL                                                                   v1.0
;-------------------------------------------------------------------------------
ROL:        mov       r1, r0    ;R1=R0
            rol       r1        ;msb al carry
            rol       r0        ;Desplazar a la izqda insertando C=msb por la dcha
            ret
```

CRITERIO DE CORRECCIÓN

a) 50%
b) 20%

Apellidos:.....**SOLUCIÓN**.....

Nombre:.....

1	2	3	4

c) 10%

d) 20%