ENGINYERIA INFORMÀTICA
Assignatura: DISSENY DIGITAL BASIC
Avaluació única 20/1/2022
BLOC 1

COGNOM(S) I NOM:	**DNI o NIE:**	**Signat:**

1) Volem fer l'operació aritmètica següent **en Ca2 amb 12 bits** de la manera com la faria un sistema digital (en complement a 2):

$$(A – B) + C$$

on **A** = **FA_{16}** (hexadecimal), **B** = **110101010_{SM}** (representat en signe-mòdul amb 9 bits), **C** = **101100000_{Ca1}** (representat en Complement a 1 amb 9 bits). Expressa el resultat de l'operació en **Ca2** i en **signe-mòdul** i representa el valor absolut del resultat en **BCD**.
Fes les operacions tal com s'indica, fent tots els canvis necessaris per convertir els números a 12 bits, i indica els diferents resultats obtinguts a les caselles aquí sota.

A – B = Ca2	**(A – B) + C =** SM
(A – B) + C = Ca2	**|(A – B) + C| =** BCD

EXPLICA TOTS ELS PASSOS I TRANSFORMACIONS QUE FAS

(3.5 punts)

SOLUCIÓ

Volem fer l'operació de 3 números: **(A – B) + C**. Hem de recordar, que per poder fer totes les operacions, millor primer de tot transformar cada número en **Ca2** de 12 bits.

Número A=FA_{16}: com que està en hexadecimal, primer de tot el convertim en binari natural, cosa que podem fer fàcilment convertint cada dígit hexadecimal en 4 dígits binaris, ja que la base hexadecimal és 2^4. Així en resulta que **$A_{16}=1010_2$** (el dígit hexadecimal A és 10) i **$F_{16}=1111_2$** (el dígit hexadecimal F és 15). Per tant **$A=11111010_2$** (per comprovar les operacions, mirem quant val aquest número **A** en decimal: 1*1+1*2+1*4+1*8+1*16+1*32+1*128=**250_{10}**; també ho podem mirar directament a partir del número en hexadecimal: 10*1+15*16=250). En binari natural (i en hexadecimal) només podem representar números positius, per la qual cosa **A** és positiu, representat en aquest cas amb 8 bits. A continuació convertim aquest número en **Ca2**, (que és mateix que signe-mòdul positiu). Per tant només caldrà afegir-hi 4 **0**s a l'esquerra i el número resultant serà **$A=000011111010_{Ca2}$**.

Número B=110101010_{SM}: està en signe-mòdul amb 9 bits i veiem que el 1r bit és un **1**, cosa que ens indica que és un **número negatiu**. Ja que la primera operació és una resta, en compte del número **B**, transformem el número (**–B**). En signe-mòdul de 9 bits **–B** és el número **(–B)=010101010_{SM}**. Com que aquest número és positiu, el convertim primer a signe-mòdul amb 12 bits, que consisteix només en afegir-hi 3 **0** a l'esquerra, és a dir, **(–B)=000010101010_{SM}**. (de cara a fer comprovacions, deduïm quin és el valor decimal de **(–B)**, que és 1*2+1*8+1*32+1*128=**170_{10}**; per tant, el número **B** serà $–170_{10}$). Aquest mateix número, al ser positiu, tindrà la mateixa representació en **Ca2**, és a dir, és **(–B)= 000010101010_{Ca2}**.

Número C=101100000$_{Ca1}$: es tracta d'un número en **Ca1** amb 9 bits. De forma anàloga al que hem fet amb el número **B**, primer identifiquem que es tracta d'un número negatiu, ja que el bit de més pes és **1**. Per tant, podem transformar-lo en la seva representació en **Ca2** de 12 bits, només fa falta afegir-li uns 3 1s a la esquerra, i sumar-li 1. El resultat dóna **C=111101100001$_{Ca2}$**. Per trobar el corresponen número en base 10, calculem –C en Ca2, invertint bit a bit a partir del primer 1 de la dreta. **–C=000010011111$_{Ca2}$**. Sent positiu, la seva representació en binari és la mateixa, i convertint-lo en decimal dóna –C= 1*1+1*2+1*4+1*8+1*16+1*128= 159. Llavors **C = –159$_{10}$**.

Ara ja podem fer les diferents operacions. Comencem per **(A–B) = A+(–B)** i sumem bit a bit, generant l'arrossegament en cada cas:

```
             11111 1
A            000011111010Ca2
–B           000010101010Ca2
---------------------------------------
(A–B)        000110100100Ca2
```

Aquest resultat és positiu, indicat per el valor del primer bit i el número que en resulta és 1*4+1*32+1*128+1*256=420$_{10}$ (que coincideix amb 250-(-170)$_{10}$). Aquest resultat ja és correcte en la representació en Ca2.

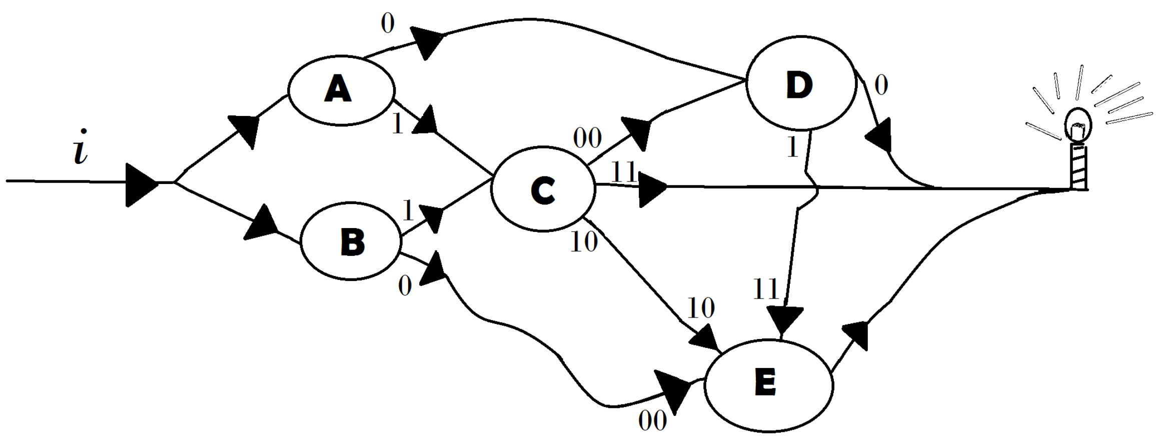
Ara podem fer la següent operació: **((A – B) + C)** Procedim igual que abans.

```
                      111111
A–B                   000110100100Ca2
C                     111101100001Ca2
-----------------------------------------------
(A–B)+ -C            1000100000101Ca2
```

Descartem el bit 13 (en vermell), ja que la representació de la suma s'ha de trencar al bit 12, per tal que el resultat és el número **000100000101$_{Ca2}$**. El primer bit ens indica que el resultat torna a ser un número positiu, per tal que la seva representació en **signe-mòdul** coincideix amb la binària i amb la en **Ca$_2$**. Calculem el seu valor decimal: 1*1+1*4+1*256 = **261$_{10}$** = 420 – 159, comprovant que l'operació és correcta a partir de l'operació en decimal.

Falta només la seva representació en **BCD**. El codi **BCD** codifica cada dígit decimal amb 4 bits, així que agafem cada xifra decimal i la convertim en un bloc de 4 dígits binaris. Així 2$_{10}$=**0010$_{BCD}$**, 6$_{10}$=**0110$_{BCD}$** i 1$_{10}$=**0001$_{BCD}$**. Per tant, el número resultant és el **00101100001$_{BCD}$**.

2) La figura següent representa un circuit elèctric, on en cada branca el pas de corrent es controla mitjançant una sèrie d'interruptors digitals. Al gràfic s'indiquen amb fletxes els sentits de circulació del corrent, i també l'estat de cada interruptor. Per exemple, si l'interruptor **A** està en estat 0, està dirigint el corrent cap a la branca superior que el connecta amb l'interruptor **D**; si, en canvi, el seu estat és 1, el corrent circula per la branca inferior cap a l'interruptor **C**. O si l'interruptor **E** està en estat 00, només li entra el corrent que prové de l'interruptor **B**, mentre que en l'estat 10 només prové de l'interruptor **C**, i en l'estat 11, de l'interruptor **D**.



Alguns interruptors no tenen un funcionament independent, com és el cas dels interruptors **A**, **B** i **D**. El control dels interruptors **B** i **D** és el mateix (si **B** està en 1, **D** també; si **B** està en 0, **D** també), mentre el dels interruptors **A** i **D** és complementari (si **A** està en 1, **D** està en 0 i viceversa). Els senyals associats als diferents interruptors s'han d'anomenar **A**, **B**, $\mathbf{C_1C_0}$, **D** i $\mathbf{E_1E_0}$, respectivament.
A més, tal com es veu de l'esquema, els casos $\mathbf{C_1C_0}$=01 o $\mathbf{E_1E_0}$=01 no es poden donar, ja que no corresponen a cap posició d'aquests interruptors.

Plantegis en quines condicions el corrent i arribarà a encendre la bombeta. Per a això:

a) Planteja el problema, raona i indica quantes variables caldrà per resoldre'l.
b) Representa la funció de sortida en minterms i maxterms i escriu algebraicament un dels maxterms (indica quin és).
c) Simplifica la funció EXCLUSIVAMENT a través de mapes de Karnaugh.
d) Fes l'esquemàtic del circuit resultant.

EXPLICA TOTS ELS PASSOS I TRANSFORMACIONS QUE FAS

(6.5 punts)

SOLUCIÓ

En funció de les variables de control ***B***, ***C*1**, ***C*0**, ***E1***, ***E*0** dels interruptors, escrivim una taula de possible camins de corrent que permeten encendre la bombeta. Agafem la funció ***f*** =1 si el corrent arriba a la bombeta. S'han descartats tots els casos on la variable de control de l'interruptor ***A*** tenia el mateix valor que el de ***B*** o de ***D***, ja que no és una condició vàlida.

Fent servir la condició que imposa la igualtat de les variables de control dels interruptors ***C*** i ***D***, el problema es redueix a tenir només 5 variables d'entrada independents.

B	***C1 C0***	***E1E0***	***f***
1	(00,10,11)	11	1
0	00	(00,10,11)	1
0	11	(00,10,11)	1
0	10	10	1
1	00	11	1
1	11	(00,10,11)	1
1	10	10	1
0	(00,10,11)	00	1

Així les primeres 4 línies (en groc) de la taula indiquen tots els camins possibles que passen pel interruptor ***A***, mentre les altres 4 (en blau) aquells que passen pel interruptor ***B***. Hi ha també casos que no poden sortir mai, els que corresponen a totes les combinacions que tenen ***C1C0***=01, o ***E1E0***=01. Tenim llavors tots els ingredients per escriure una taula de la veritat de la funció *f* en funció de les 32 combinacions possibles de les 5 variables d'entrada escollides.

#	***B***	***C1***	***C0***	***E1***	***E0***	***f***
0	0	0	0	0	0	1
1	0	0	0	0	1	X
2	0	0	0	1	0	1
3	0	0	0	1	1	1
4	0	0	1	0	0	X
5	0	0	1	0	1	X
6	0	0	1	1	0	X
7	0	0	1	1	1	X
8	0	1	0	0	0	1
9	0	1	0	0	1	X
10	0	1	0	1	0	1
11	0	1	0	1	1	0
12	0	1	1	0	0	1
13	0	1	1	0	1	X
14	0	1	1	1	0	1
15	0	1	1	1	1	1
16	1	0	0	0	0	0
17	1	0	0	0	1	X
18	1	0	0	1	0	0
19	1	0	0	1	1	1
20	1	0	1	0	0	X
21	1	0	1	0	1	X
22	1	0	1	1	0	X
23	1	0	1	1	1	X
24	1	1	0	0	0	0
25	1	1	0	0	1	X
26	1	1	0	1	0	1
27	1	1	0	1	1	1
28	1	1	1	0	0	1
29	1	1	1	0	1	X
30	1	1	1	1	0	1
31	1	1	1	1	1	1

Ara ja podem representar, tal com ens demanen, la funció en minterms i maxterms. En suma de minterms serà:

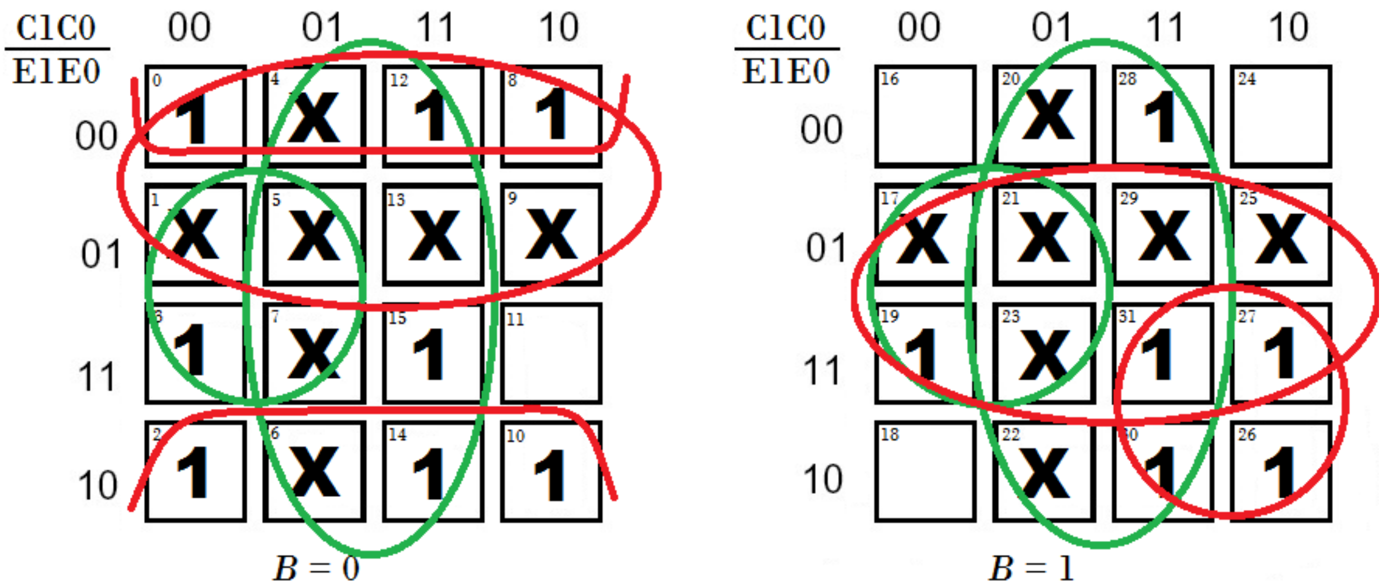
$f(B,C_1,C_0,E_1,E_0) =$
$\Sigma(0,2,3,8,10,12,14,15,19,26,27,28,30,31)+\Phi(1,4,5,6,7,9,13,17,20,21,22,23,25,29)$

I en producte de maxterms:

$f(B,C_1,C_0,E_1,E_0) = \Pi(11,16,18,24)\cdot\Phi(1,4,5,6,7,9,13,17,20,21,22,23,25,29)$

El maxterm 11 és la combinació **(/*B* + *C*1 + /*C*0 + *E*1 + *E*0).**

Per tal d'expressar la funció ***f*** de manera compacta, cal fes ús dels diagrames de Karnaugh i simplificar-los. He escollit els minterms. Amb cinc variables tinc dos sub-diagrames de Karnaugh superposats, un per a cada valor de la variable ***B***.



Un cop disposats els 1s i els Xs als 2 sub-diagrames 4x4, que corresponen al de ***B***=0 i ***B***=1, procedeixo a fer els agrupaments. En verd he indicat els comuns als dos diagrames, en vermell els que corresponen a cadascun d'ell. El agrupament verd del costat d'esquerra correspon a $/C_1 * E_0$, i el central a C_0. El vermell de dalt del sub-diagrama de l'esquerra correspon a $/B * /E_1$, i el de la primera i última fila correspon a $/B * /E_0$, mentre al sub-diagrama de a dreta falta afegir els termes $B * E_0$ (central, en vermell) i $B * C_1 * E_1$ (lateral, en vermell). Aleshores, en definitiva la funció ***f*** simplificada té la forma:

$$f(B,C_1,C_0,E_1,E_0) = C_0 + /C_1*E_0 + /B*E_1 + /B*/E_0 + B*E_0 + B*C_1*E_1$$

Surt una solució anàloga si s'escull la variable ***A*** en lloc de la ***B***, amb
$f(A,C_1,C_0,E_1,E_0)=\Sigma(3,10,11,12,14,15,16,18,19,24,26,28,30,31)+\Phi(1,4,5,6,7,9,13,17,20,21,$
$22,23,25,29) =\Pi(0,2,8,27)\cdot\Phi(1,4,5,6,7,9,13,17,20,21,22,23,25,29)$

EXPLICA TOTS ELS PASSOS I TRANSFORMACIONS QUE FAS

BLOC 2

Dissenya una màquina d'estats que serveixi per a introduir caramels en una bossa i per tancar-la quan estigui plena i preparar la bossa següent. Hi ha dos tipus de caramels, simple i doble, que surten aleatòriament per un tub. Un sistema de sensors detecten si hi ha o no caramel, i si aquest és simple o si és doble. Quan la bossa s'hagi omplert amb 5 caramels, tenint en compte que un de doble val per 2 i que no poden entrar mai més de 5 caramels a la bossa, el sistema ha de tancar la bossa i col·locar-ne una de buida sota el tub.

Realitza:

1. Descripció d'entrades i sortides ***(0,5 punts)***
2. Descripció d'estats ***(0,5 punts)***
3. Diagrama d'estats ***(3 punts)***
4. Taula d'estats ***(1 punts)***
5. Minimització d'estats ***(no cal, no puntua)***
6. Assignació d'estats ***(0,5 punts)***
7. Taula de transicions ***(1 punts)***
8. Elecció de FFs ***(no puntua)***
9. Mapes de Karnaugh per resoldre les lògiques dels senyals d'entrada dels FFs i dels de sortida del circuït ***(3 punts)***
10. Resolució gràfica del circuït ***(0,5 punts)***

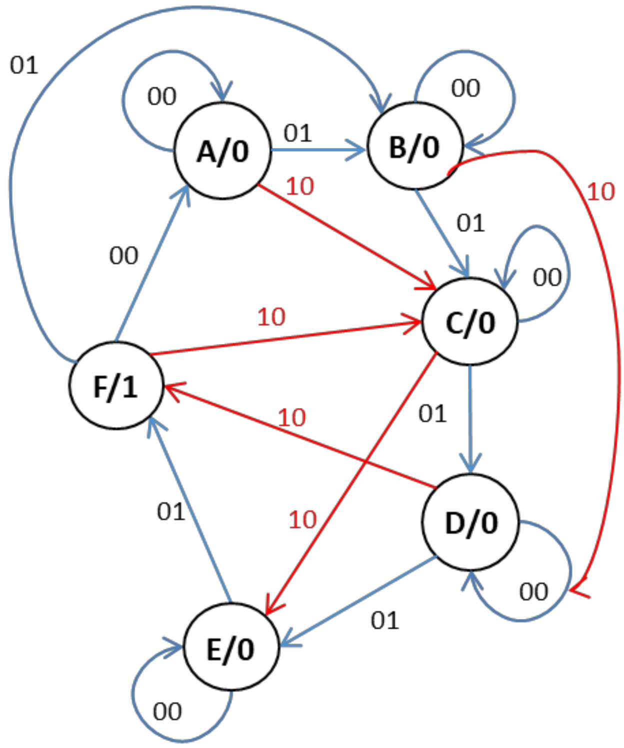
A cadascun dels 10 apartats anteriors, explica com el fas.

(10 punts)

1. **Definición de entradas y salidas**
   2 Entradas X1= caramelo doble, X0= caramelo simple (CASO 11 NO EXISTE)
   1 Salida Z=1 mueve bolsa

2. **Definición de estados**
   - A estado inicial sin caramelos y por tanto la salida es 0 (no mueve la bolsa)
   - B estado en que ha almacenado 1 caramelo y por tanto la salida es 0 (no mueve la bolsa)
   - C estado en que ha almacenado 2 caramelos y por tanto la salida es 0 (no mueve la bolsa)
   - D estado en que ha almacenado 3 caramelos y por tanto la salida es 0 (no mueve la bolsa)
   - E estado en que ha almacenado 4 caramelos y por tanto la salida es 0 (no mueve la bolsa)
   - F estado en que ha almacenado 5 caramelos y por tanto la salida es 1 (mueve la bolsa)

3. **Diagrama de estados**



4. **Tabla de estados**

Estado Presente Y	Estado Futuro Y^+ X=00	X=01	X=10	X=11	Salida Z
A	A	B	C	X	0
B	B	C	D	X	0
C	C	D	E	X	0
D	D	E	F	X	0
E	E	F	X	X	0
F	A	B	C	X	1

5. **Minimización de estados**
   Son todos diferentes

6. **Asignación de estados.**
   Hay 6 estados, se necesitan 3 FF's: Y2, Y1, Y0. Asignaremos arbitrariamente: A=000, B=001, C=010, D=011, E=100 y F=101

7. Tabla de transiciones

D2 D1 D0

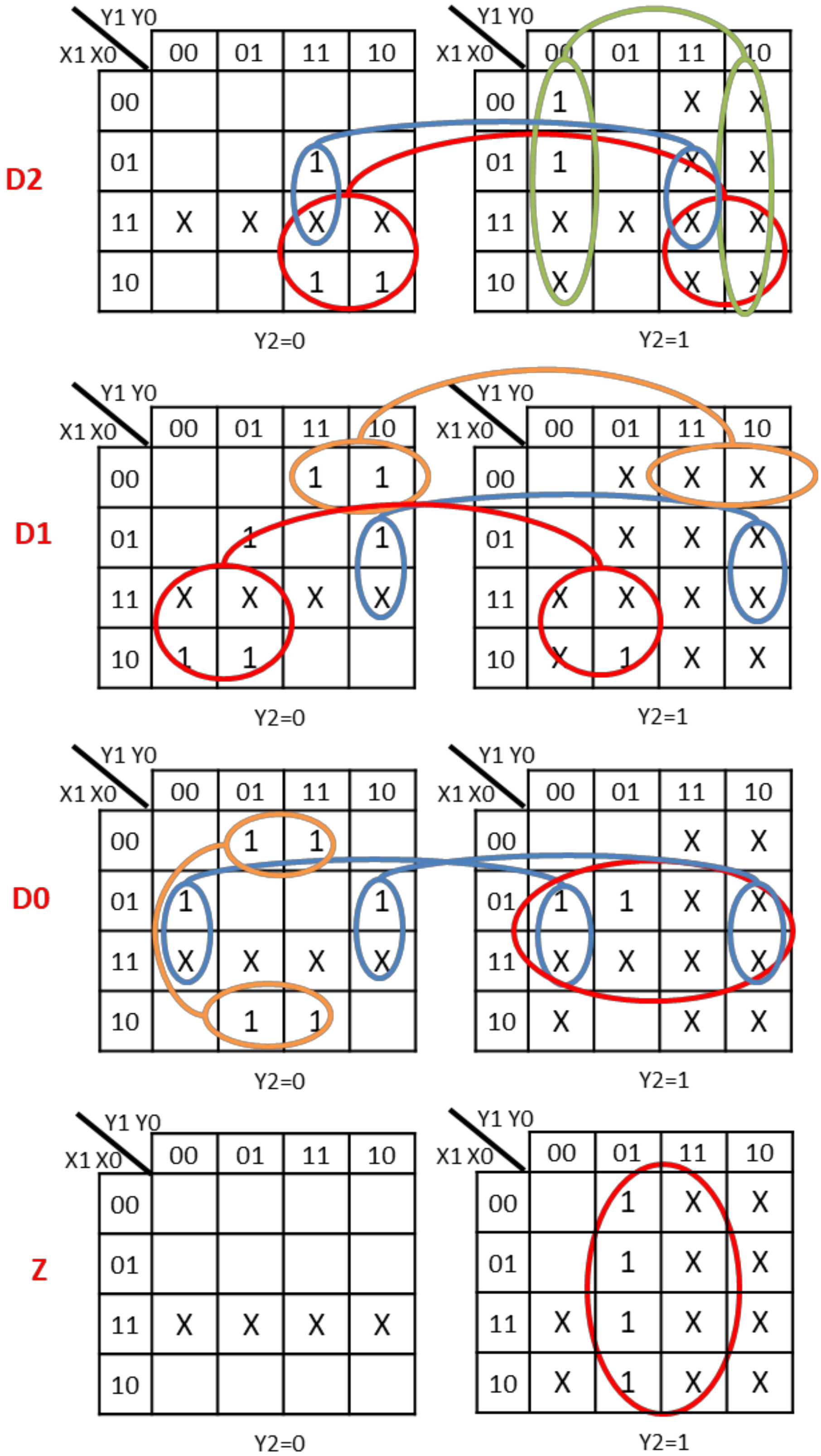
||| ||| |||

Presente	Y2	Y1	Y0	X1	X0	Futuro	$Y2^+$	$Y1^+$	$Y0^+$	Z
A	0	0	0	0	0	A	0	0	0	0
A	0	0	0	0	1	B	0	0	1	0
A	0	0	0	1	0	C	0	1	0	0
A	0	0	0	1	1	-	X	X	X	X
B	0	0	1	0	0	B	0	0	1	0
B	0	0	1	0	1	C	0	1	0	0
B	0	0	1	1	0	D	0	1	1	0
B	0	0	1	1	1	-	X	X	X	X
C	0	1	0	0	0	C	0	1	0	0
C	0	1	0	0	1	D	0	1	1	0
C	0	1	0	1	0	E	1	0	0	0
C	0	1	0	1	1	-	X	X	X	X
D	0	1	1	0	0	D	0	1	1	0
D	0	1	1	0	1	E	1	0	0	0
D	0	1	1	1	0	F	1	0	1	0
D	0	1	1	1	1	-	X	X	X	X
E	1	0	0	0	0	E	1	0	0	0
E	1	0	0	0	1	F	1	0	1	0
E	1	0	0	1	0	-	X	X	X	X
E	1	0	0	1	1	-	X	X	X	X
F	1	0	1	0	0	A	0	0	0	1
F	1	0	1	0	1	B	0	0	1	1
F	1	0	1	1	0	C	0	1	0	1
F	1	0	1	1	1	-	X	X	X	1
-	1	1	0	0	0	-	X	X	X	X
-	1	1	0	0	1	-	X	X	X	X
-	1	1	0	1	0	-	X	X	X	X
-	1	1	0	1	1	-	X	X	X	X
-	1	1	1	0	0	-	X	X	X	X
-	1	1	1	0	1	-	X	X	X	X
-	1	1	1	1	0	-	X	X	X	X
-	1	1	1	1	1	-	X	X	X	X

8. Elección de FFs

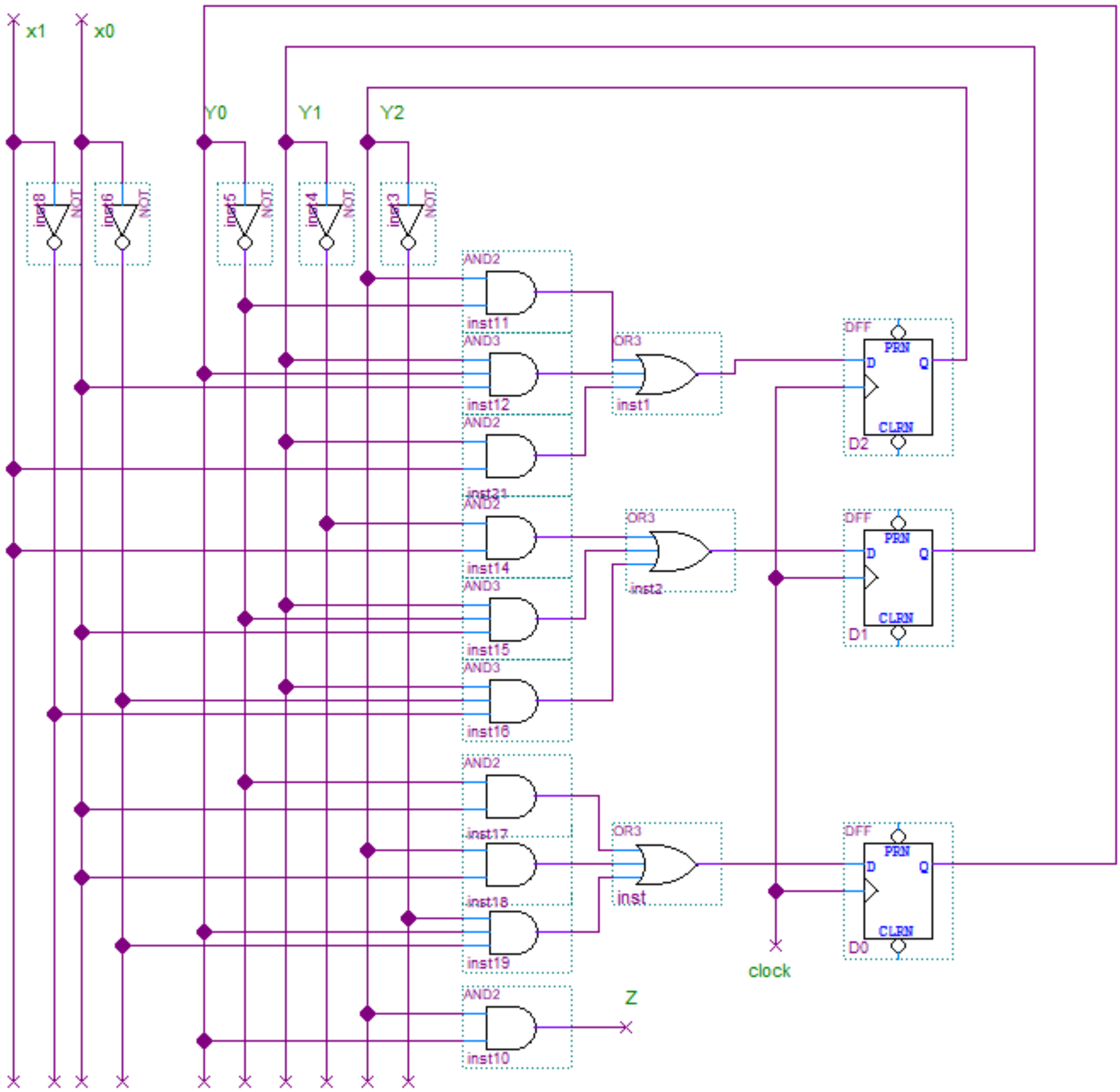
Tipo D por flanco de subida. **Q^+=D,** por tanto **$Y2^+$=D2, $Y1^+$=D1, $Y0^+$=D0**,

9. Resolución de Karnaughs



D2 = (Y2·/Y0) + (Y1·Y0·X0) + (Y1·X1)
D1 = (/Y1·X1) + (Y1·/Y0·X0) + (Y1·/X1·/X0)
D0 = (/Y0·X0) + (/Y2·Y0·/X0) + (Y2·X0)
Z = Y2·Y0

Esquema

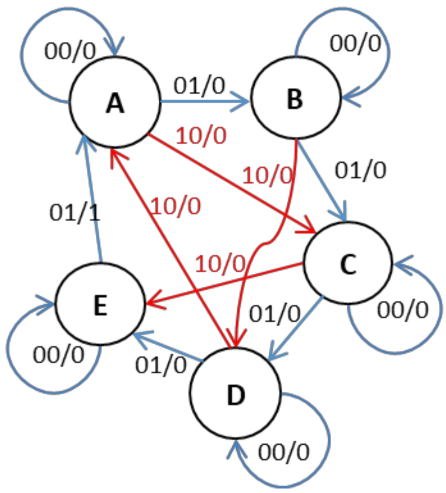

x1
x0
Y0
Y1
Y2
inst8
inst6
inst5
inst4
inst3
NOT
AND2
inst11
AND3
inst12
AND2
inst21
AND2
inst14
AND3
inst15
AND3
inst16
AND2
inst17
inst18
inst19
AND2
inst10
OR3
inst1
OR3
inst2
OR3
inst
DFF
PRN
D
Q
CLRN
D2
D1
D0
clock
Z

1. **Definición de entradas y salidas**
   2 Entradas X1= caramelo doble, X0= caramelo simple (CASO 11 NO EXISTE)
   1 Salida Z=1 mueve bolsa

2. **Definición de estados**

- A estado inicial sin caramelos
- B estado en que ha almacenado 1 caramelo
- C estado en que ha almacenado 2 caramelos
- D estado en que ha almacenado 3 caramelos
- E estado en que ha almacenado 4 caramelos

3. **Diagrama de estados**



4. **Tabla de estados**

Estado Presente Y	Estado Futuro Y^+ / Salida Z X=00	X=01	X=10	X=11
A	A / 0	B/ 0	C/0	X/X
B	B / 0	C / 0	D/0	X/X
C	C / 0	D / 0	E/0	X/X
D	D / 0	E / 0	A/1	X/X
E	E / 0	A / 1	X/X	X/X

5. **Minimización de estados**
   Son todos diferentes

6. **Asignación de estados.**
   Hay 5 estados, se necesitan 3 FF's: Y2, Y1, Y0. Asignaremos arbitrariamente: A=000, B=001, C=010, D=011 y E=100

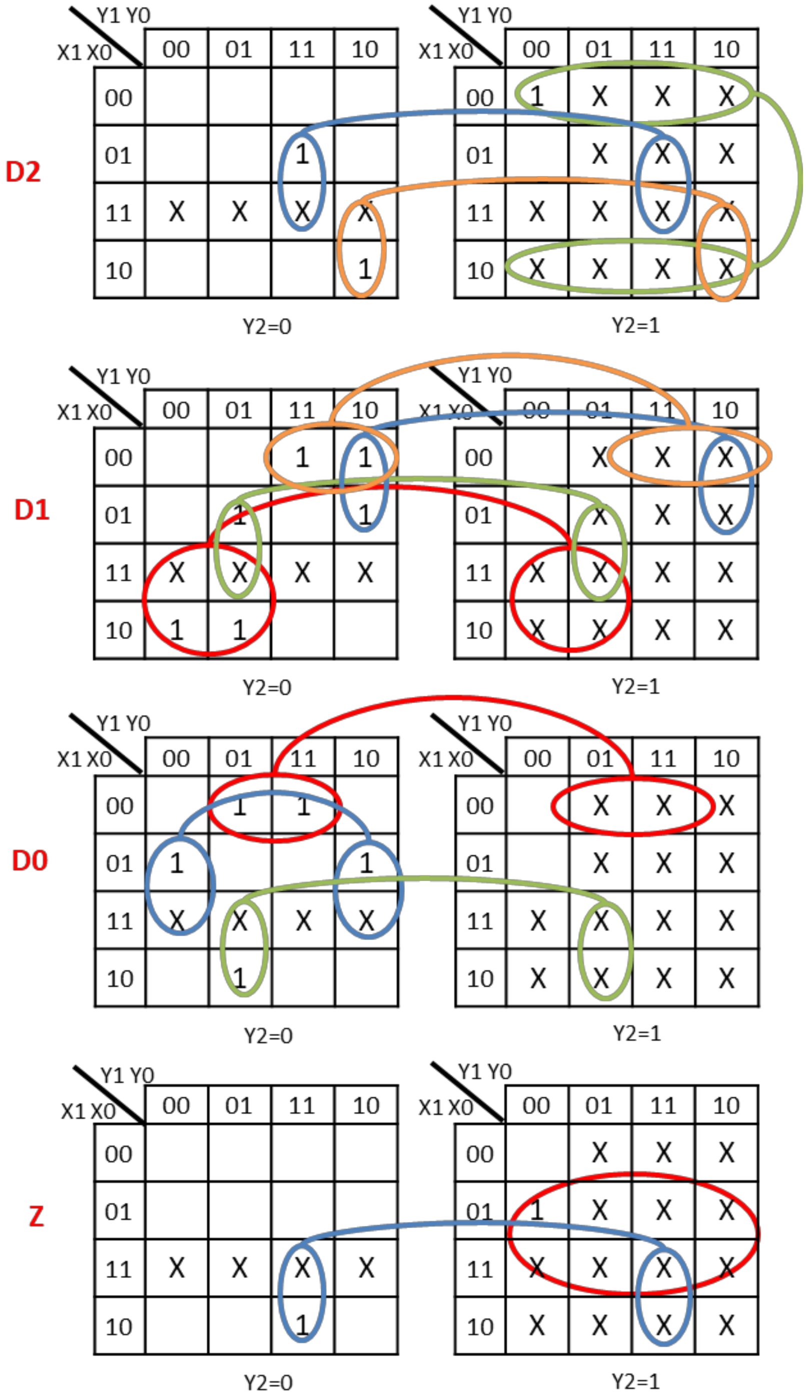
7. Tabla de transiciones

							D2	D1	D0	
Presente	**Y2**	**Y1**	**Y0**	**X1**	**X0**	**Futuro**	**$Y2^+$**	**$Y1^+$**	**$Y0^+$**	**Z**
A	0	0	0	0	0	A	0	0	0	0
A	0	0	0	0	1	B	0	0	1	0
A	0	0	0	1	0	C	0	1	0	0
A	0	0	0	1	1	-	X	X	X	X
B	0	0	1	0	0	B	0	0	1	0
B	0	0	1	0	1	C	0	1	0	0
B	0	0	1	1	0	D	0	1	1	0
B	0	0	1	1	1	-	X	X	X	X
C	0	1	0	0	0	C	0	1	0	0
C	0	1	0	0	1	D	0	1	1	0
C	0	1	0	1	0	E	1	0	0	0
C	0	1	0	1	1	-	X	X	X	X
D	0	1	1	0	0	D	0	1	1	0
D	0	1	1	0	1	E	1	0	0	0
D	0	1	1	1	0	A	0	0	0	1
D	0	1	1	1	1	-	X	X	X	X
E	1	0	0	0	0	E	1	0	0	0
E	1	0	0	0	1	A	0	0	0	1
E	1	0	0	1	0	-	X	X	X	X
E	1	0	0	1	1	-	X	X	X	X
-	1	0	1	0	0	-	X	X	X	X
-	1	0	1	0	1	-	X	X	X	X
-	1	0	1	1	0	-	X	X	X	X
-	1	0	1	1	1	-	X	X	X	X
-	1	1	0	0	0	-	X	X	X	X
-	1	1	0	0	1	-	X	X	X	X
-	1	1	0	1	0	-	X	X	X	X
-	1	1	0	1	1	-	X	X	X	X
-	1	1	1	0	0	-	X	X	X	X
-	1	1	1	0	1	-	X	X	X	X
-	1	1	1	1	0	-	X	X	X	X
-	1	1	1	1	1	-	X	X	X	X

8. Elección de FFs

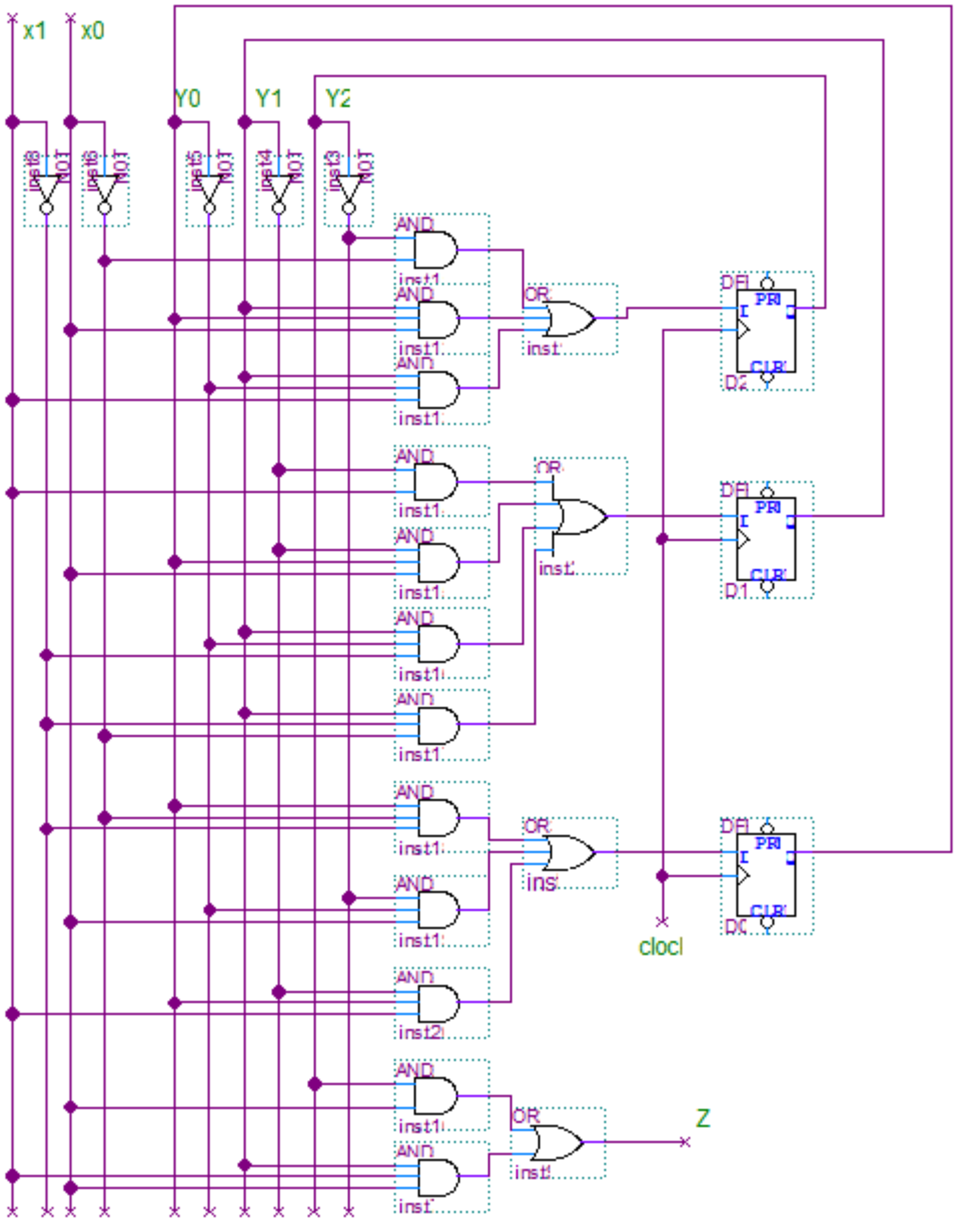
Tipo D por flanco de subida. **Q^+=D,** por tanto **$Y2^+$=D2, $Y1^+$=D1, $Y0^+$=D0**,

9. Resolución de Karnaughs



D2 = (Y2·/X0) + (Y1·Y0·X0) + (Y1·/Y0·X1)
D1 = (/Y1·X1) + (/Y1·Y0·X0) + (Y1·/Y0·/X1) + (Y1·/X1·/X0)
D0 = (Y0·/X1·/X0) + (/Y2·/Y0·X0) + (/Y1·Y0·X1)
Z = (Y2·X0) + (Y1·Y0·X1)

10. Esquema



BLOC 3

Dissenya un **ALU** que pugui realitzar les següents funcions

e) **A-B-1 (operació aritmètica)**

f) **A+B (operació aritmètica)**

g) **B-A (operació aritmètica)**

h) **$(\overline{A}+B) \cdot (A+\overline{B}) \cdot (\overline{A}+\overline{B})$ (operació lògica)**

i) **$\overline{A}+B$ (operació lògica)**

on A i B son nombres binaris de 6 bits. Utilitza com a nucli del ALU un **Sumador Complert Modificat.** Dissenya els **circuits que modifiquen les paraules A i B NOMÉS amb portes lògiques** i els **circuits selectors entre les diferents sortides NOMÉS amb MUX (cap porta lògica)**.

Explica com fas les diferents transformacions i fes l'esquemàtic, indicant quants bits tens a cada línia i quantes portes/MUXs fas servir en cada cas.

(10 punts)

1. Dissenya un **ALU** que pugui realitzar les següents funcions
   a) A-B-1 (operació aritmètica)
   b) A+B (operació aritmètica)
   c) B-A (operació aritmètica)
   d) (/A+B) · (A+/B) · (/A+/B) (operació lògica)
   e) /A+B (operació lògica)

Hem de fer 3 operacions aritmètiques i 2 operacions lògiques. Primer de tot, anem a assignar un número binari a cadascuna de les operacions: farem servir binari natural. Degut a que tenim un total de 5 operacions, necessitarem 3 bits per descriure-les (f2, f1, f0). L'operació "a)" tindrà associat el 000, la "b)" el 001 ... i la "e)" el 100, tal i com s'indica a la taula de baix.

En segon lloc, anem a analitzar les diferents operacions a fer:

a) Ho podem aconseguir amb la sortida suma del sumador i complementant B bit a bit (seria fer el complement a 1). El número A es quedaria igual.
b) Simplement, hem de deixar A i B tal com estan.
c) En aquest cas, B no es modifica i A el complementem bit a bit (Ca1 A) i li sumem '1'. Per tal de sumar-li aquesta unitat, farem servir el bit Cin = 1.
d) Necessitem simplificar l'operació lògica "d)" per tenir una expressió fàcilment manejable. Ho podem fer amb àlgebra de Boole, desenvolupant o amb la taula de la veritat. Aquí us presento el desenvolupament:

(/A+B) · (A+/B) · (/A+/B) = [~~(/A·A)~~+(A·B)+(/A·/B)+~~(B·/B)~~] · (/A+/B) = [(A·B) + (/A·/B)] · (/A+/B) =
~~(A·B·/A)~~ + (/A·/A·/B) + ~~(A·B·/B)~~ + (/A·/B·/B) = (/A·/B) + (/A·/B) = (/A·/B)

   Per tant, la funció lògica que hem de fer és una AND de /A i /B. Per poder-la implementar, hem de complementar bit a bit A i B (Ca1 A i Ca1 B) i triem la sortida lògica AND del sumador complet.
e) En aquest cas, hem d'agafar la sortida lògica OR del sumador complet, complementant bit a bit A i B es quedaria sense modificar.

Una vegada fet aquest anàlisi, veiem que les transformacions que hem d'aplicar tant A com B són simplement deixar-les igual o bé complementar-les bit a bit, segons l'operació (a la taula ho teniu com a MA o FA. Per tant, amb una porta XOR aplicada a tots els bits i fent entrar un '0' o un '1', segons sigui l'entrada directa o bé el seu complement bit a bit.

Per tal de triar les sortides, farem servir dos multiplexors 2 a 1, de forma que el primer seleccioni entre sortida suma i sortida lògia i el segon, entre la sortida "AND" i la sortida "OR".

Tota aquesta informació, es pot resumir a la següent taula:

f2 f1 f0	**Operació**	**MA**	**FA (xor)**	**MB**	**FB (xor)**	**Fcin**	**Sortida**	**FM1**	**FM2**
0 0 0	A-B-1	A	0	/B	1	0	Suma	0	X
0 0 1	A+B	A	0	B	0	0	Suma	0	X
0 1 0	B-A	/A	1	B	0	1	Suma	0	X
0 1 1	/A AND /B	/A	1	/B	1	X	Lògica	1	0
1 0 0	/A OR B	/A	1	B	0	X	Lògica	1	1
1 0 1	---	---	X	---	X	X	X	X	X
1 1 0	---	---	X	---	X	X	X	X	X
1 1 1	---	---	X	---	X	X	X	X	X

Ara només quedaria trobar les expressions de FA, FB, Fcin, FM1 i FM2, en funció de f2, f1, f0.
Per això, simplificarem per Karnaugh cadascuna d'elles:

FA

f0 \ f2f1	00	01	11	10
0		1	X	1
1		1	X	X

FA = S2 + S1

FB

f0 \ f2f1	00	01	11	10
0	1		X	
1		1	X	X

FB = (/S2·/S1·/S0) + (S1·S0)

FCin

f0 \ f2f1	00	01	11	10
0		1	X	X
1		X	X	X

FCin = S1

FM1

f0 \ f2f1	00	01	11	10
0			X	1
1		1	X	X

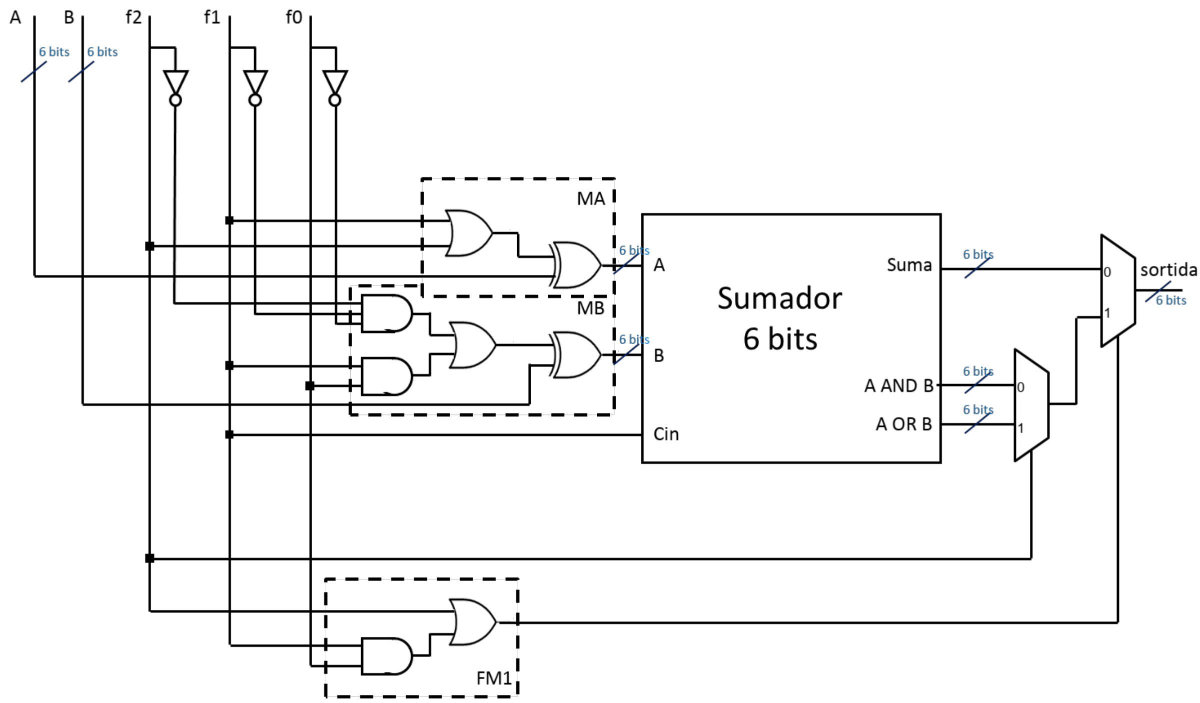
FM1 = S2 + (S1·S0)

FM2

f0 \ f2f1	00	01	11	10
0	X	X	X	1
1	X		X	X

FM2 = S2

I aquí tenim el circuit resultant:



BLOC 4

Dissenya un circuit, emprant les eines de disseny global, que realitzi la seqüència següent de **números de 5 bits** representats en Ca2: **-16, -8, -4, -2, -1, 0, -16, -8, ...**

Pots utilitzar: **registres de 5 bits, comparadors (del nombre de bits que vulguis), sumadors de 5 bits, multiplexors**, però no pots utilitzar portes lògiques.

Explica, primer, com fas el disseny i en què et bases per fer-ho. Després, fes l'esquemàtic i explica quina funció fan els diferents components utilitzats.

Indicació 1: el registre genèric NO disposa de la funció CLEAR. Les funcions que el registre fa són les següents:

Variables S1 S0	Funció
0 0	Manteniment (Hold)
0 1	Desplaçament a la dreta (Shift Right)
1 0	Desplaçament a l'esquerra (Shift Left)
1 1	Càrrega en paral·lel

Indicació 2: Dibuixa i identifica TOTS els terminals dels components que facis servir, indicant si són d'entrada o de sortida de dades.

(10 punts)

Per dissenyar el circuit que ens demanen en aquest problema, el primer que hem de fer és representar-nos en Ca2 els valors per els quals passa el sistema. Primer, recordem que amb Ca2 amb n bits podem representar del número -2^{n-1} fins el $2^{n-1}-1$ (si comparem amb signe-mòdul i amb Ca1, que només poden representar números entre $-2^{n-1}-1$ i $2^{n-1}-1$, perquè el 0 té dues representacions, podem representar un número més, que és, precisament, el -2^{n-1}, número al qual no se li pot aplicar el procediment per calcular el seu negatiu). Per tant, com que fem servir Ca2 amb 5 bits, podem representar del -16_{10} al $+15_{10}$.

Ara mirem quins són els números per els quals passa el problema. Són:

-16_{10}: 10000_{Ca2}

-8_{10}: 11000_{Ca2}

-4_{10}: 11100_{Ca2}

-2_{10}: 11110_{Ca2}

-1_{10}: 11111_{Ca2}

0_{10}: 00000_{Ca2}

Fixem-nos que entre el -16_{10} i el -1_{10}, per passar d'un número al següent, el que fem és afegir un **"1"** a l'esquerra. Per fer aquesta funció podem aprofitar el mode "Desplaçament a la dreta (Shift Right)" del registre, posant els senyals de control S_1=0 i S_0=1 i introduint un **"1"** per l'entrada en sèrie de l'esquerra, D_L.

Per a carregar el 0_{10}, com que el registre no disposa de la funció "Clear", podem fer servir la función "Càrrega en paral·lel", posant els senyals de control S_1=1 i S_0=1, i posant **"00000"** a l'entrada en paral·lel del registre. Així, doncs, el canvi dels senyals de control del registre, es realitzarà un cop hagi arribat el número **"11111"** al registre, és a dir, just abans de passar a **"00000"**. Podem fer servir diverses opcions:

a) identificar aquest canvi fent servir un comparador de 5 bits, que compari el número que surt del registre amb **”11111”** i quan siguin iguals, que la sortida del comparador actuï sobre el senyal de control S_1, fent-lo passar de **“0”** a **“1”**. Tinguem present que el comparador sempre compara en binari natural.

b) identificar que el bit de menys pes és un **“1”** i que aquest bit actuï sobre el senyal de control S_1 igual que en el cas a).

c) Igual que el b) però fent servir un MUX i que el bit de menys pes sigui la variable de selecció del MUX, posant com a entrada de dades del MUX **“0”** en el canal de dades 0 i **“1”** en el canal de dades 1.

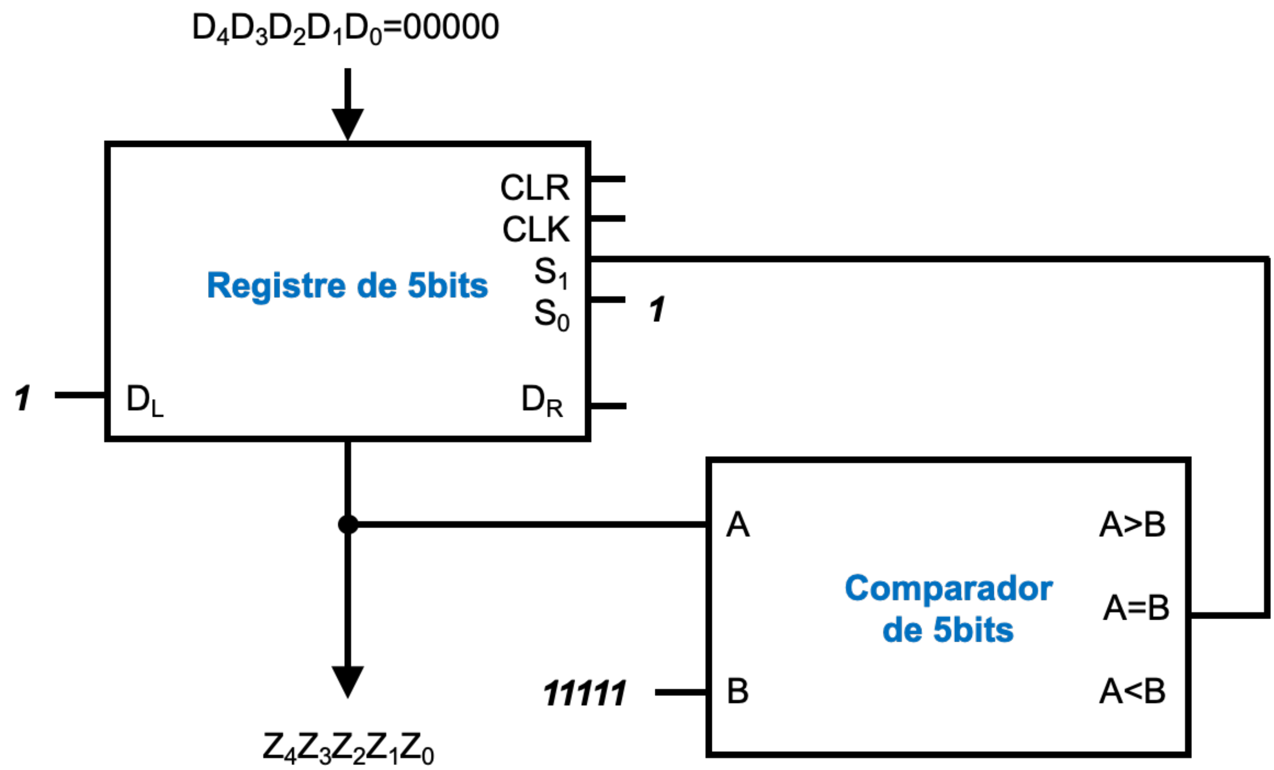
Per tornar a la seqüència, és a dir, per passar de 0_{10} a -16_{10}, és a dir, per passar de **”00000”** a **”10000”**, fem sevir el funcionament del registre en mode “Desplaçament a la dreta (Shift Right)”, entrant un **“1”** per l’entrada de l’esquerra, és a dir, les mateixes condicions que per anar passant per els diferents números negatius.

Fixem-nos que en qualsevol dels casos a), b) i c) descrits més amunt, un cop hem fet entrar en paral·lel el número **“00000”**, el senyal que controla el senyal de control S_1 del registre ha passat de 1 a 0 i, per tant, reprèn l’entrada en sèrie per l’esquerra, introduint un **“1”**. Per tant, el registre passa de **“00000”** a **“10000”**.

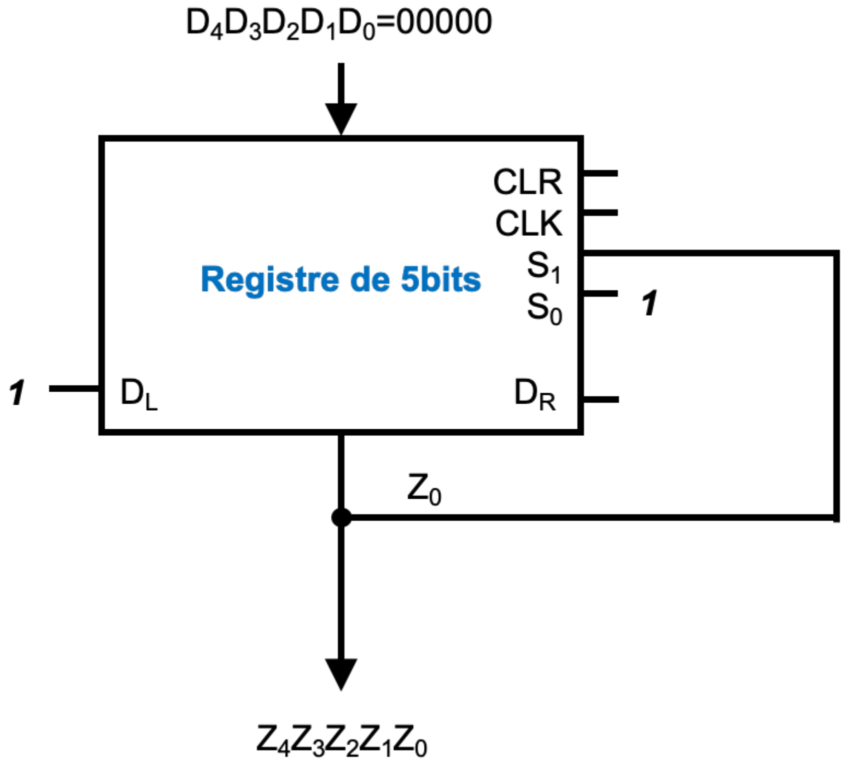
Fixem-nos, també, que podem deixar sempre el senyal D_L a **“1”** per que quan el registre realitzi la funció “Càrrega en paral·lel”, el senyal D_L no té cap impacte en el funcionament. Anàlogament, podem deixar sempre el senyal **“00000”** present a l’entrada en paral·lel del registre per que, quan el registre funciona en mode “Desplaçament a la dreta (Shift Right)”, l’entrada en paral·lel no afecta al funcionament.

Els esquemes corresponents a les 3 possibilitats es detallen a continuació.

a)

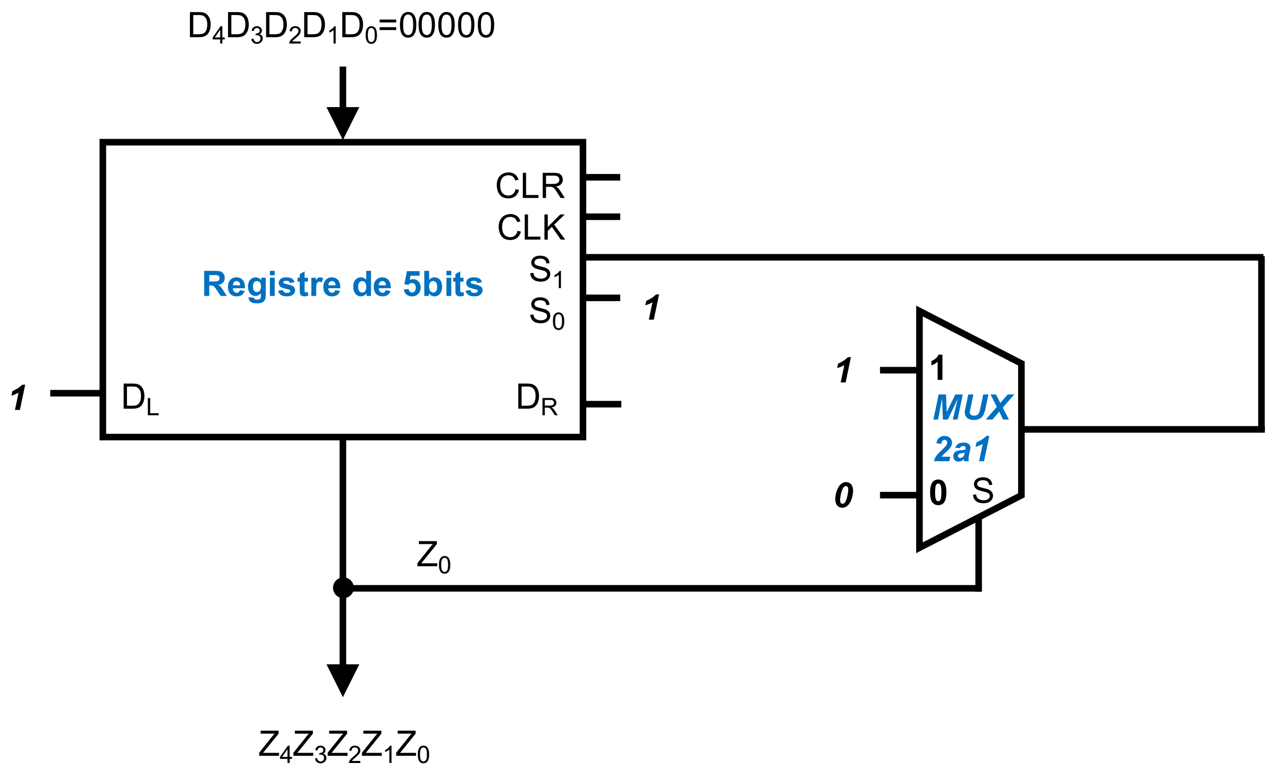


b)



D4D3D2D1D0=00000
CLR
CLK
Registre de 5bits
S1
S0
1
1
DL
DR
Z0
Z4Z3Z2Z1Z0

c)



D4D3D2D1D0=00000
CLR
CLK
Registre de 5bits
S1
S0
1
1
DL
DR
1
1
MUX
2a1
0
0
S
Z0
Z4Z3Z2Z1Z0