Pràctiques de Sistemes Digitals i Microprocessadors Curs 2023-2024

Pràctica 1

LSBattleShip

	Login	Nom
Alumnes	oriol.aparicio	Oriol Aparicio Casanovas
	hicham.naf	Hicham Naf

	Placa	Memòria	Nota
Entrega			

Data	29/10/23

Index

1. Síntesi de l'enunciat

Per a realitzar la pràctica de LSBattleShip implementarem digitalment el joc d'enfonsar vaixells BattleShip entre dues persones.
El funcionament consistirà a fer que, els dos usuaris introdueixin les coordenades dels seus 5 vaixells per torns. Un cop distribuïts comença el joc, llavors per torns els jugadors hauran d'atacar les caselles del taulell i el sistema comprovarà si hi havia un vaixell contrincant. Finalment, el joc finalitzarà quan un dels jugadors hagi encertat la posició de tots els vaixells contrincants.

Per a realitzar la pràctica la dividirem en dues fases, i en aquesta memòria només explicarem la fase 1. En aquesta primera fase ens centrarem en l'obtenció de les dades del taulell dels dos usuaris per a preparar la partida.

Primer de tot generarem una mida de vaixell aleatòria, de mida 1 a 5, i seguidament demanarem les coordenades a un jugador, mitjançant el teclat matriu de 3x4, i introduirem tantes coordenades com posicions que ocupa el vaixell generat. La conversió de les tecles del teclat matriu les farem amb una EEPROM.
Un cop el primer jugador hagi acabat d'introduir totes les coordenades del vaixell, el segon jugador introduirà les coordenades d'un vaixell d'igual mida.
Finalment, un cop hagin introduït 5 vaixells els dos jugadors, es podrà començar la partida.

Altrament, per comunicar la fase 1 amb la fase 2, ens ajudarem de 5 senyals per traspassar les dades d'una fase a l'altre. Els senyals seran: *BoatCoords[7..0], NewCoord, NewBoat, StartGame i CurrentPlayer*.

2. Plantejament

Per a realitzar aquesta pràctica hem dividit el projecte en diferents mòduls, fent que a l'hora de solar i debugar la placa sigues més senzill trobar errors i comprovar el seu correcte funcionament.

En aquesta fase es demana la implementació de la màquina d'estats amb el PIC18F4321 programat en llenguatge d'assemblador, fent que controli i gestioni els diversos blocs de la pràctica. També hem creat un bloc que ens genera un clock a freqüència 1000Hz.

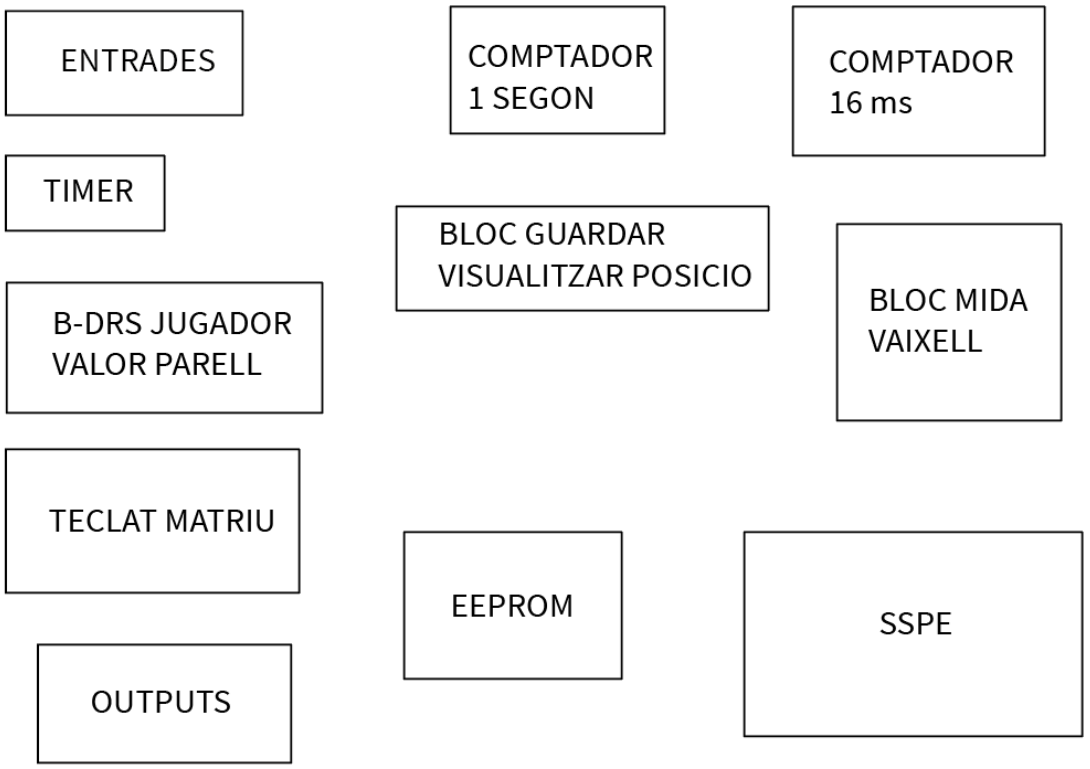
Per començar tenim el bloc del teclat matriu, en el que juntament amb l'EEPROM i un seguit de registres i comptadors, obteníem el número del teclat traduït a binari natural. Per a fer l'escombrat del teclat, també hem utilitzat la màquina d'estats i el bloc del comptador de 16 ms.

Tot seguit, tenim el bloc de visualització, que ens permet veure mitjançant dos displays 7-Segments els números introduïts pel teclat, acompanyats per uns biestables per a controlar la lògica de quan s'han de mostrar.

Finalment, tenim un bloc on generem el nombre aleatori i el mostrem amb tres led's, també guardem el número generat en un registre i el carreguem en un comptador per a controlar la mida del vaixell.

3. Diagrama de mòduls

A continuació tenim l'esquema dels blocs implementats per a resoldre la pràctica



A continuació explicarem més detalladament el funcionament de cada un dels blocs:

1. ENTRADES

En aquest bloc ens encarregarem de generar, mitjançant dos polsadors, els senyals de PCI i ACK (per a implementar la fase 1 de moment hem utilitzat un polsador, de cara a la fase 2, aquest senyal vindrà del bloc de la fase 2), això ens permetrà reiniciar el sistema en qualsevol moment i indicar que tenim una nova coordenada al sistema.

2. TIMER

En aquest bloc generarem el clock del sistema. Aquest anirà a una freqüència de 1000 Hz i un duty-cycle del 50%. En l'apartat d'Unitat de procés: Clock del sistema, explicarem amb més detall el seu esquema i el valor de resistències escollit.

3. TECLAT MATRIU

El bloc de teclat matriu serà l'encarregat de rebre les tecles premudes al teclat matriu, gestionar l'escombrat d'aquest i guardar la combinació introduïda al teclat (convertida a binari per l'EEPROM) en un registre.

4. EEPROM

Aquest bloc consisteix només de l'EEPROM, que rebrà la combinació generada pel teclat matriu i la convertirà en el número equivalent en binari natural.

5. COMPTADOR 1 SEGON

En aquest bloc generarem un comptador d'un segon, mitjançant tres comptadors conectats en serie i el senyal del clock.

6. COMPTADOR 16 ms

Aquest bloc generarà un comptador de 16 milisegons, mitjançant el clock.

7. B-DRS JUGADOR I VALOR PARELL

Aquest bloc generarà el senyal del current player, que indicarà si és el torn del jugador 0 o del jugador 1. També s'encarrega de generar un senyal cada vegada que introduïm un nombre parell de valors, és a dir, cada vegada que introduïm una tecla si l'hem premuda un nombre parell de vegades, aquest senyal s'activarà.

8. BLOC GUARDAR I VISUALITZAR POSICIÓ

En aquest bloc ens encarregarem de mostrar els valors convertits del teclat matriu mitjançant displays 7-Segments, que aniran shiftant els seus valors a mesura que introduïm més números. També, amb l'ajuda de dos biestables, controlarem la visualització dels números pels displays, cada vegada que introduïm una nova coordenada.

9. BLOC MIDA VAIXELL

En aquest bloc ens encarregarem de generar el número aleatori, de guardar-lo i gestionar que introduïm el mateix nombre de coordenades que el nombre aleatori generat. També generarem un senyal que indicarà que totes les coordenades d'un vaixell han estat introduïdes.

10. SSPE

Aquest bloc és el de la màquina d'estats, on tindrem el PIC18F4321 i un registre que guardarà l'estat actual.

11. OUTPUTS

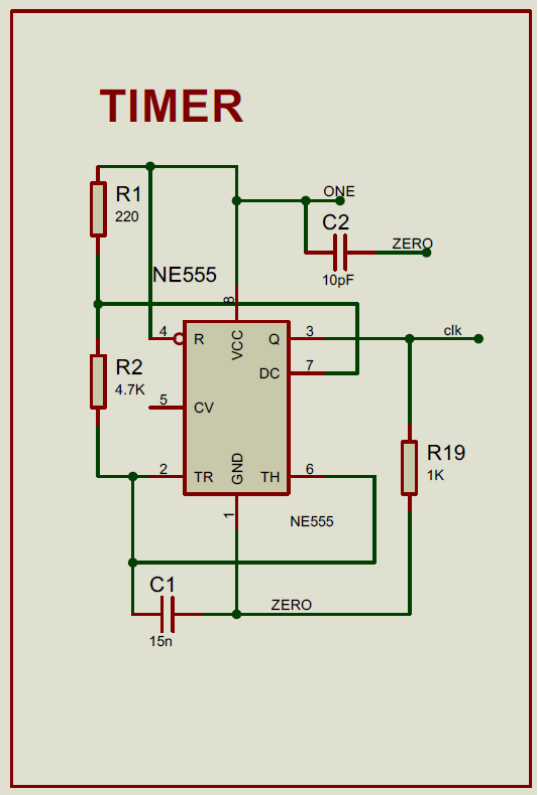
Aquest bloc mostrarà mitjançant led's el current player, start game, i el nombre aleatori (mostrat per un conjunt de tres leds).

4. Disseny de la màquina d'estats

4.1. Unitat de control

Disseny de la unitat de sincronització

La unitat de sincronització consta d'un xip NE555, aquest xip genera un pols rectangular que variarà depenent dels valors de les resistències i condensadors que hi connectem.
Per al disseny d'aquesta fase hem necessitat una sortida quadrada d'1 ms de període en què estigués a 1 la primera meitat del període i a 0 la segona meitat, és a dir un duty cicle del 50%.
A continuació tenim el diagrama de mòduls de la unitat de sincronització:



Tenim el NE555 que ens genera el senyal de sincronisme a 1kHz d'aquesta manera les 2 fases aniran a la mateixa velocitat.

Llavors tenim diverses sortides de la Fase 1 que van a la Fase 2.

BoatCoords[7..0] és el bus de dades de la coordenada, quan en la fase 1 tenim ja estable aquesta sortida, l'hi indiquem a la Fase 2 mitjançant un flanc de baixada de la sortida

NewCoord, aquest ens respondrà amb un ACK quan hagi acabat de gestionar aquesta informació.

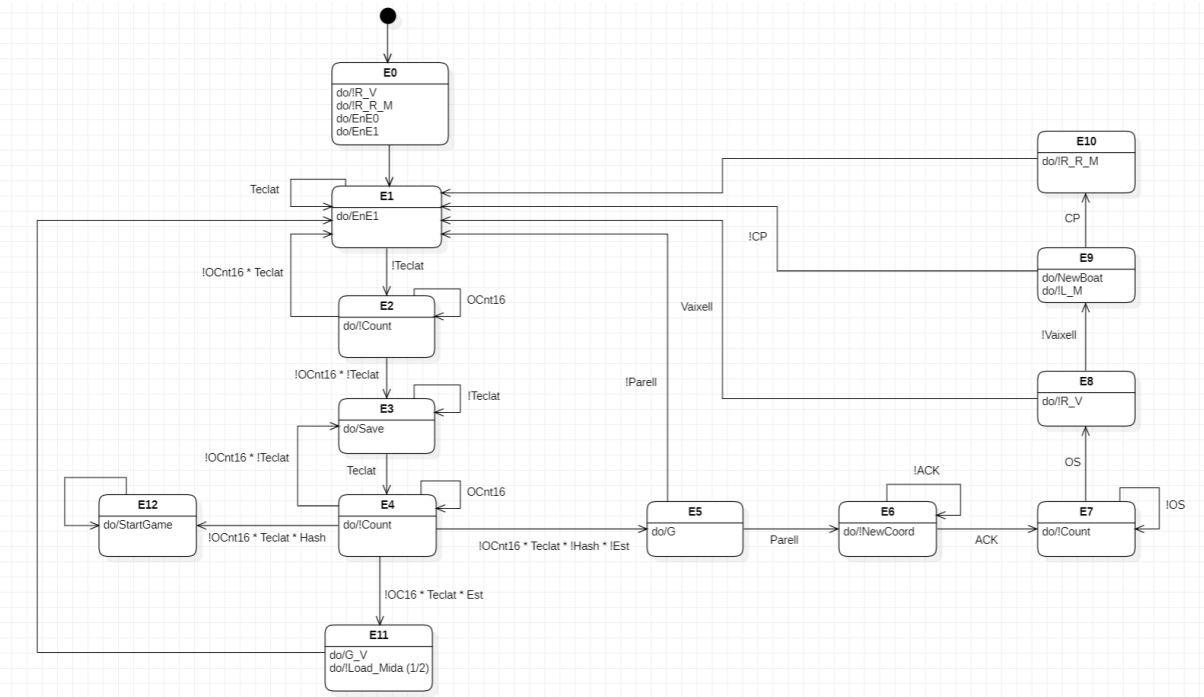
Llavors tenim el senyal NewBoat que ens serveix per indicar-li a la fase 2 que ja tenim una mida d'un vaixell, aquest senyal enviem després de rebre l'ACK, no l'enviem conjuntament amb el NewCoord per a poder gestionar aquestes dos informacions per separat.

El senyal Current Player l'hi indicarà a la fase 2 de quin jugador és la coordenada que l'hi arriba.

I finalment el Start Game que l'hi indicarà a la Fase 2 que ja hem acabat d'introduir les coordenades i pot començar amb la Fase 2.

Disseny del SSS (Sistema Seqüencial Síncron)

La màquina d'estats consta de 12 estats.



A continuació explicarem el funcionament de cada un dels estats:

Iniciem en l'estat E0 al fer un PCI, en aquest estat fem un reset general de tot el hardware i llavors anem a l'estat E1 sense cap condició.

En l'estat E1 habilitem l'escombrat, l'E1 és l'únic estat on activem l'escombrat i en tots els altres estarà parat. Esperem que es premi una tecla que va amb lògica negativa, una vegada premuda anem a l'estat E2.

En l'estat E2 habilitem el comptador de 16ms per evitar els rebots, una vegada passats aquests 16ms comprovem si tecla val 1 vol dir que ha estat un rebot i tornem a l'E1, però si, en canvi, val 0 vol dir que realment s'ha premut una tecla i anem a l'estat E3.

En l'estat E3 guardem la sortida de l'EEPROM a un registre i mantindre'l guardat, ens mantenim en aquest estat mentre tecla segueixi valent 0, quan valgui 1 anem a l'estat E4.

En l'estat E4 filtrem els rebots de sortida esperant 16ms, una vegada passats aquests 16ms comprovem quina tecla ens han premut, pot passar 4 coses, la primera és que hagi sigut un rebot de sortida així que tornem a l'estat E3, la segona és que s'hagi premut el Hashtag i llavors anirem a l'estat E12, la tercera opció és que s'hagi premut l'Estèric i anirem a l'estat E11 i finalment que no sigui cap d'aquestes dues tecles que voldria dir que la tecla premuda és la d'un número i llavors procedim a anar a l'estat E5.

En l'estat E5 activem la sortida Guardar que fa diverses coses, donar un flanc als registres dels valors a visualitzar per guardar el nou valor introduït, un altre flanc en els biestables que s'encarreguen de la visualització dels valors i per últim shiftar el valor del biestable que ens indica si el numero es parell. Llavors comprovem si el número introduït és el primer o el segon amb l'entrada Parell.
Si Parell val 0 vol dir que es el primer número i llavors tornem a l'estat E1 per esperar a que s'introdueix un segon número. Si val 1 vol dir que ja s'han introduït 2 valors i llavors ja tenim una coordenada per enviar i procedim a anar a l'estat E6.

En l'estat E6 enviem posem a 0 New Cord per generar un flanc negatiu i esperem a rebre ACK de la fase 2, New Cord també l'utilitzem per descomptar el valor del comptador de la mida del vaixell. Un cop ACK valgui 1 passem a l'estat E7.

En l'estat E7 després de rebre ACK comptem 1 segon per després apagar els displays, com que la senyal d'ACK que rebrem de la fase 2 sera molt rapida, és més logic esperar primer l'ACK i després comptar 1 segon que no pas comptar 1 segon primer i perdre el senyal d'ACK en el procés. Quan ja s'ha acabat de comptar 1 segon anem a l'estat E8.

En l'estat E8 activem el reset dels visualitzadors que van en lógica negativa, aqui comprovem si ja hem arribat a la mida del vaixell o no amb la senyal Vaixell, si val 0 vol dir que no hem arribat a la mida i llavors tornem a l'estat E1, pero si en canvi val 1 anem a l'estat E9.

En l'estat E9 activem New Boat i fem un load al comptador de la mida del vaixell per el segon jugador,
New Boat també l'utilitzem per shiftar el valor del biestable del Current Player. Finalment si el Current Player val 0 tornem a l'estat E1 pero si val 1 anem a l'estat E10.

En l'estat E10 fem un reset al registre de la mida del vaixell per apagar els LEDs de la mida i tornem a l'estat E1.

En l'estat E11 generem un flanc positiu en el registre que guarda la mida del vaixell per tindre una nova mida i en mig estat carregar aquesta mida al comptador que anirà descomptant cada cop que introduïm una coordenada. Llavors tornem a l'estat E1.

En l'estat E12 generem el senyal Start Game i ens quedem en aquest estat fins que tornem a fer PCI per iniciar nova partida.

En resum, l'estat E0 fem el reset de tot el hardware, l'estat E1 es el nostre estat principal on hi fem l'escombrat i esperem a que ens premin una tecla, els estats E2, E3 i E4 serveixen per controlar els rebots d'entrada i de sortida.
En l'estat E4 comprovem si la tecla és un hashtag anem a l'estat E12 i ens cadem enclavats alla, si és un Asterisc anem a l'estat E11 i generem un número aleatori.
Si no es cap d'aquests dos anem a l'E5 i es procedeix a fer tota la gestió del número introduït.

Disseny de la interfície

Aquestes són les equacions de la nostra màquina d'estats.

Primer tenim les equacions dels estats, com que tenim 12 estats utilitzem 4 bits.
Després tenim les equacions per a anar a cada un dels estats.
Finalment, tenim les sortides que s'activen en l'estat que toca.

/* Estats */

E0 = !EA3 & !EA2 & !EA1 & !EA0;
E1 = !EA3 & !EA2 & !EA1 & EA0;
E2 = !EA3 & !EA2 & EA1 & !EA0;
E3 = !EA3 & !EA2 & EA1 & EA0;
E4 = !EA3 & EA2 & !EA1 & !EA0;
E5 = !EA3 & EA2 & !EA1 & EA0;
E6 = !EA3 &!EA2 & EA1 & !EA0;
E7 = !EA3 & EA2 & EA1 & EA0;

```
E8 = EA3 & !EA2 & !EA1 & !EA0;
E9 = EA3 & !EA2 & !EA1 & EA0;
E10 = EA3 & !EA2 & EA1 & !EA0;
E11 = EA3 & !EA2 & EA1 & EA0;
E12 = EA3 & EA2 & !EA1 & !EA0;

/* Condicions */

C1 = E0 # (E1 & Teclat) # (E5 & !Parell) # (E8 & Vaixell) # (E9 & !CP) # E10 # E11 # (E2 &
!OCnt16 & Teclat);
C2 = (E2 & OCnt16) # (E1 & !Teclat);
C3 = (E2 & !OCnt16 & !Teclat) # (E3 & !Teclat) # (E4 & !OCnt16 & !Teclat);
C4 = (E3 & Teclat) # (E4 & OCnt16);
C5 = (E4 & !OCnt16 & Teclat & !Hash & !Est);
C6 = (E5 & Parell) # (E6 & !ACK);
C7 = (E6 & ACK) # (E7 & !OS);
C8= (E7 & OS);
C9 = (E8 & !Vaixell);
C10 = (E9 & CP);
C11 = (E4 & !OCnt16 & Teclat & Est);
C12 = (E4 & !OCnt16 & Teclat & Hash) # E12;

/*Estats Futurs*/
EF0 = (C1 # C3 # C5 # C7 # C9 # C11);
EF1 = (C2 # C3 # C6 # C7 # C10 # C11);
EF2 = (C4 # C5 # C6 # C7 # C12);
EF3 = (C8 # C9 # C10 # C11 # C12);

/* Sortides */
R_R_M = !(E10 # E0);
Reset_Vis = !(E8 # E0);
EnEsc0 = E0 # E1;
EnEsc1 = E0;
EnCnt16 = (!E2 & !E4 & !E7);
Save = E3;
G = E5;
NewCoord = !E6;
NewBoat = E9;
G_V = E11;
LoadMida = !((E11 & !clk) # E9);
StartGame = E12;
```

Un cop hem fet les equacions de la màquina d'estats, hem fet una taula de la veritat per poder implementar-la en llenguatge assembler.

ºEA	EA3	EA2	EA1	EA0	T	OC16	H	E	P	ACK	OS	V	CP	ºEF	EF3	EF2	EF1	EF0
E0	0	0	0	0	X	X	X	X	X	X	X	X	X	E1	0	0	0	1
E1	0	0	0	1	1	X	X	X	X	X	X	X	X	E1	0	0	0	1
E1	0	0	0	1	0	X	X	X	X	X	X	X	X	E2	0	0	1	0
E2	0	0	1	0	X	1	X	X	X	X	X	X	X	E2	0	0	1	0
E2	0	0	1	0	1	0	X	X	X	X	X	X	X	E1	0	0	0	1
E2	0	0	1	0	0	0	X	X	X	X	X	X	X	E3	0	0	1	1
E3	0	0	1	1	0	X	X	X	X	X	X	X	X	E3	0	0	1	1
E3	0	0	1	1	1	X	X	X	X	X	X	X	X	E4	0	1	0	0
E4	0	1	0	0	X	1	X	X	X	X	X	X	X	E4	0	1	0	0
E4	0	1	0	0	0	0	X	X	X	X	X	X	X	E3	0	0	1	1
E4	0	1	0	0	1	0	0	1	X	X	X	X	X	E11	1	0	1	1
E4	0	1	0	0	1	0	1	0	X	X	X	X	X	E12	1	1	0	0
E4	0	1	0	0	1	0	0	0	X	X	X	X	X	E5	0	1	0	1
E5	0	1	0	1	X	X	X	X	0	X	X	X	X	E1	0	0	0	1
E5	0	1	0	1	X	X	X	X	1	X	X	X	X	E6	0	1	1	0
E6	0	1	1	0	X	X	X	X	X	0	X	X	X	E6	0	1	1	0
E6	0	1	1	0	X	X	X	X	X	1	X	X	X	E7	0	1	1	1
E7	0	1	1	1	X	X	X	X	X	X	0	X	X	E7	0	1	1	1
E7	0	1	1	1	X	X	X	X	X	X	1	X	X	E8	1	0	0	0
E8	1	0	0	0	X	X	X	X	X	X	X	1	X	E1	0	0	0	1
E8	1	0	0	0	X	X	X	X	X	X	X	0	X	E9	1	0	0	1
E9	1	0	0	1	X	X	X	X	X	X	X	X	0	E1	0	0	0	1
E9	1	0	0	1	X	X	X	X	X	X	X	X	1	E10	1	0	1	0
E10	1	0	1	0	X	X	X	X	X	X	X	X	X	E1	0	0	0	1
E11	1	0	1	1	X	X	X	X	X	X	X	X	X	E1	0	0	0	1
E12	1	1	0	0	X	X	X	X	X	X	X	X	X	E12	1	1	0	0

LLavors en assembler lo primer que fem és declarar les nostres entrades i sortides, llavors fem la nostre maquina d'estats implementant la taula anterior.
Lo que hem de fer es determinar quin estat volem anar depenent de l'estat actual i les entrades que estiguin actives.

Exemple:

Si estem en l'estat E4, volem anar a l'estat E11 si es compleix que Tecla val 0, Comptador de 16ms val 0, Hashtag val 0 i l'Asterisc val 1.
Si es compleix aquesta condició anomenada minterm ja que fem les equacions de les sortides que volem que valguin 1.
El que fem és posar a 1 els bits de l'estat futur que volem anar, com que volem anar a l'estat E11, hem d'activar el EF3, EF1 i EF0 per tenir '1011'.

Així que el següent minterm el repetim per a aquests 3 bits.

```
;-------------------------------------------------------------------------;
;        Minterms per EF3                                                 ;
;-------------------------------------------------------------------------;

CLRF RESULT, 0 ;Inicialitzzem RESULT a 0

;Minterm (!EA3 & EA2 & !EA1 & !EA0 & T & !OC16 & !H & E)
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
BSF WREG, 0, 0 ;Posar a 0 el WREG si Tecla = 0;
BTFSC PORTA, 1, 0 ;Comprovar que OC16 = 0;
BSF WREG, 0, 0 ;Posar a 0 el WREG;
BTFSC PORTA, 2, 0 ;Comprovar que H = 0;
BSF WREG, 0, 0 ;Posar a 0 el WREG;
BTFSS PORTA, 3, 0 ;Comprovar que E = 1;
BSF WREG, 0, 0 ;Posar a 0 el WREG;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat
```

L'estructura de cada bit és sempre la mateixa, utilitzem un file anomenat RESULT que el posem a 0 al començament de tot, llavors posem tots els minterms d'aquest bit, l'estructura del minterm és primer mirar quin estat éstem.

Nosaltres utilitzem el port C per els estats actuals i futurs així que carreguem aquest port en el WREG, l'estat actual el tenim en els 4 LSB així que aquest port el multipliquem per una mascara '00001111', d'aquesta manera els altres 4 bits els posem a 0 i no ens afecten.

Després l'hi restem l'estat del minterm que volem comprovar, si la resta val 0 vol dir que estem en aquest estat, sino el WREG valdra un altre valor diferente a 0.

Llavors procedim a comprovar les condicions d'aquest estat per a activar aquest bit.
Això ho fem amb la comanda BTFSS o BTFSC depenent de la condició. En l'exemple anterior volia mirar que Tecla i E valguin 1 per aquest motiu hem utilitzar BTFSS que el que fa és comprovar el vit que l'hi indiquem i salta la següent linea si val 1, El BTFSC salta la següent linea si val 0.

La linea que tenim despres de cada BTFSS i BTFSC és BSF WREG, que és posar a 1 el WREG[0].
És a dir, que si arribem al final del minterm i el WREG segueix valent 0 haurem complert totes les condicions.

Llavors lo que hem de fer és restar-li un 0 al WREG perquè hi ha un registre anomenat STATUS que te un bit Z que es posa a 1 per defecte si després de fer una resta en el WREG és 0.

Finalment activem el bit 0 del RESULT si Z val 1,i al final de tot comprovem si aquest bit val 1 què voldrà dir que s'ha complert un minterm i llavors posem a 1 el bit de l'estat futur que toca.

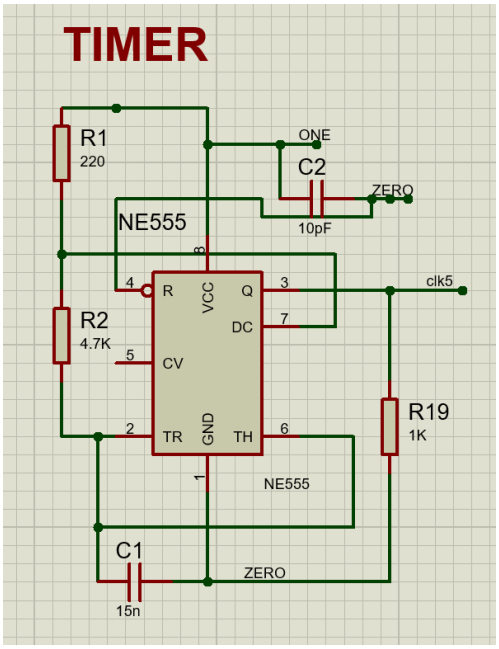
Per a activar les sortides que toquin és senzillament comprovar en quin estat estem i activar la sortida que toqui.

De portes lògiques només hem programat una not per obtenir el clock negat, es simplement comprovar si la entra clk val 0 posar la sortida a 1, i si en canvi val 1 posar-la a 0.

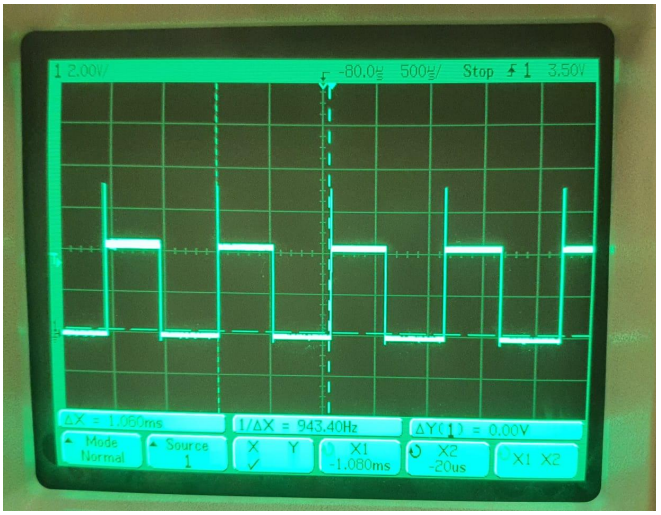
4.2. Unitat de procés

Clock del sistema

Per al clock del sistema utilitzem el N555, amb freqüència de 1000 Hz i duty-cicle del 50%, amb valors de resistències de 4,7KΩ i 220Ω i un condensador de 15nF, connectats de la següent manera:



Un cop vàrem realitzar la implementació física el valor real del clock ha resultat ser de 940 Hz, degut als valors de resistències triats i les seves toleràncies (que han estat seleccionats segons els valors comercials més comuns que s'assemblaven més al nostre càlcul), però per a la nostra implementació no ha suposat cap problema, ja que aquesta diferència no ha impactat negativament en el nostre funcionament.
En la imatge a continuació, es pot observar el clock un cop implementat ja a la placa i en funcionament:



ALU (Unitat Aritmètico-Lògica)

Teclat matricial 3x4

És un teclat matricial de 12 tecles, té 7 connexions de les quals 3 són les columnes i 4 són files, formant una graella de 3x4, on cada tecla és un polsador que quan és premut curtcircuita la sortida de la fila amb la columna.
Per saber quina tecla s'ha premut hem fet un escombrat en les files amb un comptador.

74LS194

Es un registre shifter de 4 bits, te una càrrega en paral·lel i una sortida que podem shiftar el seu valor, el que fem és carregar un '1000' i anem shiftant aquest valor, és d'aquesta manera com fem l'escombrat. Mitjançant les entrades S0 i S1 controlem el registre, quan els 2 valen 1 es fa una càrrega en paral·lel, quan nomes un d'ells val 1, la sortida es shifta cap a un sentit o cap a l'altre, i quan els dos valen 0, la sortida para de shiftar-se. Tot això ho controlem mitjançant la màquina d'estats

74LS273

Es un registre de 8 bits que utilitzem per a guardar el valor traduït de la memòria EEPROM. Mitjançant el senyal Save (de la màquina d'estats) connectat al clock del xip, indicarem quan s'ha de guardar el valor que ens dona l'EEPROM.

74LS193

És un comptador de 4 bits que utilitzarem per a generar el comptador d'1 segon, el comptador de 16 mil·lisegons, el nombre aleatori i comptar la mida del vaixell.

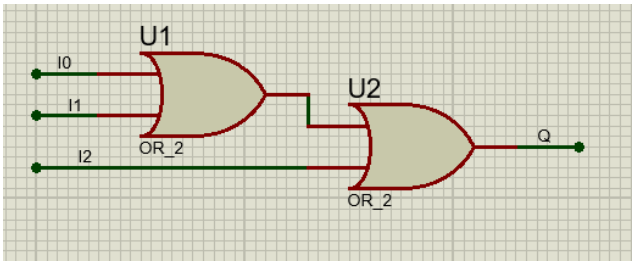
Per a generar el comptador d'un segon hem concatenat 3 comptadors en sèrie, per a poder arribar al número 1000, però realment comptarem fins al 1024, ja que si no hauríem d'utilitzar una porta lògica o un comparador per a comptar fins al nombre exacte, en comptes d'aixo agafem directament la 3ra sortida de l'ulitm comptador que és el desè (10) bit que equival a 2^10=1024 . Nosaltres ho hem ignorat, ja que no passarà res per a comptar 24 mil·lisegons extra, ja que visualment no percebem aquesta diferència.

74LS48

Aquest xip serveix per convertir de binari natural a 7-Segments.
El fem servir per mostrar els números introduïts pel teclat matriu, i les sortides d'aquests aniran connectats a displays 7-Segments.

74LS32

Son 4 portes lògiques ORs de 2 entrades. Hem necesitat 2 portes or de 3 entrades, ja que teniem a la nostre disposició aquest xip l'hem aprofitat. Per obtenir una porta or de 3 entrades hem conectat la sortida d'una or de 2 en l'entrada d'un altre, d'aquesta manera obtenim una or de 3.



Unitat de memorització

74LS74

És un biestable, que utilitzarem per a activar o desactivar els displays i generar els senyals de Parell i Current player.

Per al Parell i Current Player utilitzem un Biestable cadascun on agafem la sortida negada i la conectem a l'entrada, d'aquesta manera cada vegada que l'hi donem un flanc al clk, la sortida s'inverteix el seu valor.

Per al control dels visualitzadors hem connectat 2 biestables en serie, a l 'entrada del primer biestable hi tenim un 1 logic, la sortida Q del primer va a l'entrada del segon, i les dos entrades clk es controlen amb la mateixa senyal.

Primer estan reiniciats, quan generem un flanc, la sortida del primer biestable es posa a 1 i la del segon es manté a 0, i quan generem un segon flanc les 2 sortides es posen a 1.

74LS175

Es un registre, que utilitzarem per a guardar el nombre aleatori generat i l'estat actual de la màquina d'estats.

EEPROM 27C256

Utilitzem l'EEPROM per a convertir els valors que obtenim quan premem una tecla en el teclat matricial per a convertir-los a binari natural.
Això ho fem connectant els 7 pins del teclat matricial a les entrades de l'EEPROM i per aconseguir les sortides en binari natural fem servir una taula amb les combinacions que s'obtenen en prémer cada tecla, on hi assignem el seu equivalent en binari natural.
Per a les combinacions d'Asterisc i Hashtag els hi hem assignat un bit independent per a que sigues mes fàcil d'identificar i així facilitar la detecció d'aquestes tecles a la màquina d'estats.

Numeros Decimals	R[1]	R[2]	C[2]	R[3]	C[0]	R[0]	C[1]	Q5-*	Q4-#	Q3	Q2	Q1	Q0
1	0	0	0	0	1	1	0	0	0	0	0	0	1
2	0	0	0	0	0	1	1	0	0	0	0	1	0
3	0	0	1	0	0	1	0	0	0	0	0	1	1
4	1	0	0	0	1	0	0	0	0	0	1	0	0
5	1	0	0	0	0	0	1	0	0	0	1	0	1
6	1	0	1	0	0	0	0	0	0	0	1	1	0
7	0	1	0	0	1	0	0	0	0	0	1	1	1
8	0	1	0	0	0	0	1	0	0	1	0	0	0
9	0	1	1	0	0	0	0	0	0	1	0	0	1
0	0	0	0	1	0	0	1	0	0	0	0	0	0
*	0	0	0	1	1	0	0	1	0	0	0	0	0
#	0	0	1	1	0	0	0	0	1	0	0	0	0

Implementació del SSS (Sistema Seqüencial Síncron)

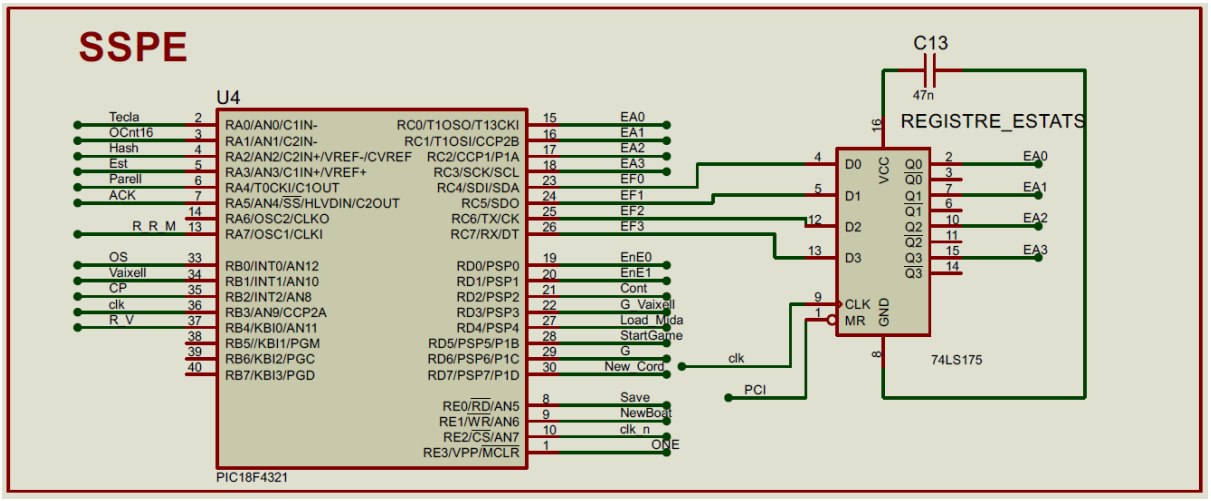
Per implementar la màquina d'estat hem fet ús del PIC18F4321. És un microcontrolador programable amb entrades i sortides.

El primer pas que hem fet és l'assignació dels pins d'entrada i sortida, tinguent en compte que tenim un nombre limitat de pins a utilitzar.

8 pins els hem utilitzat per indicar l'estat actual i l'estat futur, 4 bits d'entrada per indicar l'estat actual i 4 bits de sortida per a l'estat futur.

Per mantindre el nostre estat actual hem utilitzat un registre de 4 bits, la seva sortida ens indica l'estat actual i a la seva entrada la PIC l'hi indicara a quin estat volem anar.

Els estats del registre s'aniran actualitzant cada 1ms, ja que la seva senyal de clock va a 1000Hz.



Fitxer .asm del PIC18F4321

```
;-------------------------------------------------------------------------;
;                    Configuracions Generals                              ;
;-------------------------------------------------------------------------;
#include <p18f4321.inc>

CONFIG OSC = INTIO1
CONFIG WDT = OFF
CONFIG PBADEN = DIG
CONFIG LVP = OFF
RESULT EQU 0x00 ;Variable on guardarem el resultat de la equacio

ORG 0x0000
GOTO MAIN
ORG 0x0008
RETFIE FAST
ORG 0x0018
RETFIE FAST

CONFIG_OSC
MOVLW b'01110000'
MOVWF OSCCON, 0
MOVLW b'01000000'
MOVWF OSCTUNE, 0
RETURN
```

```
MAIN
   movlw 0x0F
   movwf ADCON1,0
   CALL CONFIG_OSC ;Cridem la funcio que configura l'oscil·lador intern


   ;Primer hem d'assignar les diferents entrades i sortides als pins del PIC
   ;EA[3..0] = PORTC[3..0] ;EF[1..0] = PORTC[7..4] ;Tecla = PORTA[0] ;OC16 = PORTA[1]
   ;Hash = PORTA[2] ;Est = PORTA[3] ;Parell = PORTA[4] ;ACK = PORTA[5]
   ;OS = PORTB[0] ;Vaixell = PORTB[1] ;C_P = PORTB[2] ;clk = PORTB[3]
   ;R_V = PORTB[4] ;R_R_M = PORTA[7] ;EnE0 = PORTD[0] ;EnE1 = PORTD[1]
   ;OC16 = PORTD[2] ;G_V = PORTD[3] ;L_M = PORTD[4] ;S_G = PORTD[5]
   ;C1S = PORTE[0] ;N_B = PORTE[1] ;clk_n = PORTE[2]

   ;1 = INPUT, 0 = OUTPUT
   MOVLW 0x0F
   MOVWF TRISC, 0
   ;Input for EA0 ;Input for EA1 ;Input for EA2 ;Input for EA3
   ;Output for EF0 ;Output for EF1 ;Output for EF2 ;Output for EF3

   BSF TRISA, 0, 0 ;Input for Tecla
   BSF TRISA, 1, 0 ;Input for OC16
   BSF TRISA, 2, 0 ;Input for Hash
   BSF TRISA, 3, 0 ;Input for Est
   BSF TRISA, 4, 0 ;Input for Parell
   BSF TRISA, 5, 0 ;Input for ACK

   BCF TRISA, 7, 0 ;Output for R_R_M

   BSF TRISB, 0, 0 ;Input for OS
   BSF TRISB, 1, 0 ;Input for Vaixell
   BSF TRISB, 2, 0 ;Input for CurrentPlayer
   BSF TRISB, 3, 0 ;Input for clk

   BCF TRISB, 4, 0 ;Output for R_V

   CLRF TRISD, 0
   ;Output for EnE0
   ;Output for EnE1
   ;Output for OC16
   ;Output for G_V
   ;Output for L_M
   ;Output for S_G
   ;Output for G
   ;Output for N_C
```

```
BCF TRISE, 0, 0 ;Output for C1S
BCF TRISE, 1, 0 ;Output for New_Boat
BCF TRISE, 2, 0 ;Output for !clk

CLRF LATD,0
BCF LATC,0,0
BCF LATC,1,0
BCF LATC,2,0
BCF LATC,3,0
BCF LATC,4,0
BCF LATC,5,0

BCF LATE,0,0
BCF LATE,1,0
BCF LATE,2,0

;Farem tot el sistema combinacional i qualsevol porta logica aqui al LOOP
LOOP
;-------------------------------------------------------------------------;
;       Minterms per EF3                                                  ;
;-------------------------------------------------------------------------;

CLRF RESULT, 0 ;Inicialitzzem RESULT a 0

;Minterm (!EA3 & EA2 & !EA1 & !EA0 & T & !OC16 & !H & E)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
BTFSC PORTA, 1, 0 ;Comprovar que 0C16 = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG;
BTFSC PORTA, 2, 0 ;Comprovar que H = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG;
BTFSS PORTA, 3, 0 ;Comprovar que E = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & !EA0 & T & !OC16 & H & !E)
```

```
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x04 ;Ara restem 0, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
BTFSC PORTA, 1, 0 ;Comprovar que 0C16 = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG;
BTFSS PORTA, 2, 0 ;Comprovar que H = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG;
BTFSC PORTA, 3, 0 ;Comprovar que E = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & EA1 & EA0 & OS)
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x07 ;Ara restem 7, si el resultat es 0 (EA[3..0] - 7 = 0) significa que EA[3..0] = 7
BTFSS PORTB, 0, 0 ;Comprovar que OS = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (EA3 & !EA2 & !EA1 & !EA0 & V)
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x08 ;Ara restem 8, si el resultat es 0 (EA[3..0] - 8 = 0) significa que EA[3..0] = 8
BTFSC PORTB, 1, 0 ;Comprovar que V = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (EA3 & !EA2 & !EA1 & EA0 & CP)
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
```

```
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els quatre ultims bits, apliquem la mÃ scara [00001111]
   SUBLW 0x09 ;Ara restem 9, si el resultat es 0 (EA[3..0] - 9 = 0) significa que EA[3..0] = 9
   BTFSS PORTB, 2, 0 ;Comprovar que CP = 1;
   BSF WREG, 0, 0 ;Posar a 1 el WREG si CP = 1;
   SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
   BTFSC STATUS, Z, 0
   BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

   ;Minterm (EA3 & EA2 & !EA1 & !EA0)
    MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
   SUBLW 0x0C ;Ara restem 12, si el resultat es 0 (EA[3..0] - 12 = 0) significa que EA[3..0] = 12
   SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
   BTFSC STATUS, Z, 0
   BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

   ;Ara nomes hem de comprovar si RESULT es un 1 o un 0, si es un 1, significa que s'han completat un o mes minterms, si es 0, no s'ha completat cap minterm
   BTFSC RESULT, 0, 0
   BSF LATC, 7, 0 ;Si RESULT = 1 llavors EF3 = 1
   BTFSS RESULT, 0, 0
   BCF LATC, 7, 0 ;Si RESULT = 0 llavors EF3 = 0

   ;-------------------------------------------------------------------------;
   ;              Minterms per EF2                ;
   ;-------------------------------------------------------------------------;

   CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

   ;Minterm (!EA3 & !EA2 & EA1 & EA0 & T)
    MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
   SUBLW 0x03 ;Ara restem 3, si el resultat es 0 (EA[3..3] - 0 = 0) significa que EA[3..0] = 3
   BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
   BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
   SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
   BTFSC STATUS, Z, 0
   BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat
```

```
;Minterm (!EA3 & EA2 & !EA1 & !EA0 & OC16)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
  SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
  BTFSS PORTA, 1, 0 ;Comprovar que OC16 = 1;
  BSF WREG, 0, 0 ;Posar a 1 el WREG si OC16 = 0;
  SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
  BTFSC STATUS, Z, 0
  BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

  ;Minterm (!EA3 & EA2 & !EA1 & !EA0 & T & !OC16 & H & !E)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
  SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
  BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
  BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
  BTFSC PORTA, 1, 0 ;Comprovar que OC16 = 0;
  BSF WREG, 0, 0
  BTFSS PORTA, 2, 0 ;Comprovar que H = 1;
  BSF WREG, 0, 0
  BTFSC PORTA, 3, 0 ;Comprovar que E = 0;
  BSF WREG, 0, 0
  SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
  BTFSC STATUS, Z, 0
  BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat


  ;Minterm (!EA3 & EA2 & !EA1 & !EA0 & T & !OC16 & !H & !E)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
  SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
  BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
  BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
  BTFSC PORTA, 1, 0 ;Comprovar que OC16 = 0;
  BSF WREG, 0, 0
  BTFSC PORTA, 2, 0 ;Comprovar que H = 0;
  BSF WREG, 0, 0
  BTFSC PORTA, 3, 0 ;Comprovar que E = 0;
  BSF WREG, 0, 0
```

```
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat


;Minterm (!EA3 & EA2 & !EA1 & EA0 & P)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x05 ;Ara restem 5, si el resultat es 0 (EA[3..0] - 5 = 0) significa que EA[3..0] = 5
BTFSS PORTA, 4, 0 ;Comprovar que P = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si P = 0;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & EA1 & !EA0)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x06 ;Ara restem 6, si el resultat es 0 (EA[3..0] - 6 = 0) significa que EA[3..0] = 6
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & EA1 & EA0 & !OS)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x07 ;Ara restem 7, si el resultat es 0 (EA[3..0] - 7 = 0) significa que EA[3..0] = 7
BTFSC PORTB, 0, 0 ;Comprovar que OS = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si OS = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat


;Minterm (EA3 & EA2 & !EA1 & !EA0)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
```

```
SUBLW 0x0C ;Ara restem 12, si el resultat es 0 (EA[3..0] - 12 = 0) significa que EA[3..0] = 12
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

BTFSC RESULT, 0, 0
BSF LATC, 6, 0 ;Si RESULT = 1 llavors EF1 = 1
BTFSS RESULT, 0, 0
BCF LATC, 6, 0 ;Si RESULT = 0 llavors EF1 = 0

;-------------------------------------------------------------------------;
;       Minterms per EF1          ;
;-------------------------------------------------------------------------;

CLRF RESULT, 0 ;Inicialitzem RESULT a 0

;Minterm (!EA3 & !EA2 & !EA1 & EA0 & !T)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x01 ;Ara restem 1, si el resultat es 0 (EA[3..0] - 1 = 0) significa que EA[3..0] = 1
BTFSC PORTA, 0, 0 ;Comprovar que Tecla = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & !EA2 & EA1 & !EA0 & !OC16)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x02 ;Ara restem 2, si el resultat es 0 (EA[3..0] - 2 = 0) significa que EA[3..0] = 2
BTFSS PORTA, 1, 0 ;Comprovar que OC16 = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si OC16 = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & !EA2 & EA1 & !EA0 & !T & !OC16)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
```

```
SUBLW 0x02 ;Ara restem 2, si el resultat es 0 (EA[3..0] - 2 = 0) significa que EA[3..0] = 2
BTFSC PORTA, 0, 0 ;Comprovar que Tecla = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
BTFSC PORTA, 1, 0 ;Comprovar que OC16 = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si OC16 = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & !EA2 & EA1 & EA0 & !T)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x03 ;Ara restem 3, si el resultat es 0 (EA[3..0] - 3 = 0) significa que EA[3..0] = 3
BTFSC PORTA, 0, 0 ;Comprovar que Tecla = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & !EA0 & !T & !OC16)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
BTFSC PORTA, 0, 0 ;Comprovar que Tecla = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
BTFSC PORTA, 1, 0 ;Comprovar que OC16 = 0;
BSF WREG, 0, 0  ;Posar a 1 el WREG si OC16 = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & !EA0 & T & !OC16 & !H & E)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
BTFSC PORTA, 1, 0 ;Comprovar que OC16 = 0;
BSF WREG, 0, 0
```

```
BTFSC PORTA, 2, 0 ;Comprovar que H = 0;
BSF WREG, 0, 0
BTFSS PORTA, 3, 0 ;Comprovar que E = 1;
BSF WREG, 0, 0
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & EA0 & P)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els dos ultims bits, apliquem la mascara [00001111]
SUBLW 0x05 ;Ara restem 5, si el resultat es 0 (EA[3..0] - 5 = 0) significa que EA[3..0] = 5
BTFSS PORTA, 4, 0 ;Comprovar que P = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si P = 0;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & EA1 & !EA0) & (ACK + !ACK)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els dos ultims bits, apliquem la mascara [00001111]
SUBLW 0x06 ;Ara restem 6, si el resultat es 0 (EA[3..0] - 6 = 0) significa que EA[3..0] = 6
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & EA0 & !OS)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x07 ;Ara restem 7, si el resultat es 0 (EA[3..0] - 7 = 0) significa que EA[3..0] = 7
BTFSC PORTB, 0, 0 ;Comprovar que OS = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si OS = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (EA3 & !EA2 & !EA1 & EA0 & CP)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
```

```
SUBLW 0x09 ;Ara restem 9, si el resultat es 0 (EA[3..0] - 9 = 0) significa que EA[3..0] = 9
BTFSS PORTB, 2, 0 ;Comprovar que CP = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si CP = 0;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

BTFSC RESULT, 0, 0
BSF LATC, 5, 0 ;Si RESULT = 1 llavors EF1 = 1
BTFSS RESULT, 0, 0
BCF LATC, 5, 0 ;Si RESULT = 0 llavors EF1 = 0
;-------------------------------------------------------------------------;
;       Minterms per EF0 = (!EA1 & !EA0 & X) | (!EA1 & EA0 & !Y)        ;
;-------------------------------------------------------------------------;

CLRF RESULT, 0 ;Inicialitzzem RESULT a 0

;Minterm (!EA3 & !EA2 & !EA1 & !EA0)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x00 ;Ara restem 0, si el resultat es 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & !EA2 & !EA1 & !EA0 & T)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x01 ;Ara restem 1, si el resultat es 0 (EA[3..0] - 1 = 0) significa que EA[3..0] = 1
BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & !EA2 & EA1 & !EA0 & T & !OC16)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x02 ;Ara restem 2, si el resultat es 0 (EA[3..0] - 2 = 0) significa que EA[3..0] = 2
BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
```

```
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
BTFSC PORTA, 1, 0
BSF WREG, 0, 0
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & !EA2 & EA1 & !EA0 & !T & !OC16)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x02 ;Ara restem 2, si el resultat es 0 (EA[3..0] - 2 = 0) significa que EA[3..0] = 2
BTFSC PORTA, 0, 0 ;Comprovar que Tecla = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
BTFSC PORTA, 1, 0
BSF WREG, 0, 0
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & !EA2 & EA1 & EA0 & !T)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x03 ;Ara restem 3, si el resultat es 0 (EA[3..0] - 3 = 0) significa que EA[3..0] = 3
BTFSC PORTA, 0, 0 ;Comprovar que Tecla = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & !EA0 & !T & !OC16)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
BTFSC PORTA, 0, 0 ;Comprovar que Tecla = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 1;
BTFSC PORTA, 1, 0
BSF WREG, 0, 0
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
```

```
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & !EA0 & T & !OC16 & !H & E)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x04 ;Ara restem 0, si el resultat es 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
BTFSC PORTA, 1, 0
BSF WREG, 0, 0
BTFSC PORTA, 2, 0
BSF WREG, 0, 0
BTFSS PORTA, 3, 0
BSF WREG, 0, 0
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & !EA0 & T & !OC16 & !H & !E)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x04 ;Ara restem 4, si el resultat es 0 (EA[3..0] - 4 = 0) significa que EA[3..0] = 4
BTFSS PORTA, 0, 0 ;Comprovar que Tecla = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si Tecla = 0;
BTFSC PORTA, 1, 0
BSF WREG, 0, 0
BTFSC PORTA, 2, 0
BSF WREG, 0, 0
BTFSC PORTA, 3, 0
BSF WREG, 0, 0
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & !EA1 & EA0 & !P)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x05 ;Ara restem 5, si el resultat es 0 (EA[3..0] - 5 = 0) significa que EA[3..0] = 5
BTFSC PORTA, 4, 0 ;Comprovar que P = 0;
```

```
BSF WREG, 0, 0 ;Posar a 1 el WREG si P = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & EA1 & !EA0 & ACK)
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x06 ;Ara restem 6, si el resultat es 0 (EA[3..0] - 6 = 0) significa que EA[3..0] = 6
BTFSS PORTA, 5, 0 ;Comprovar que ACK = 1;
BSF WREG, 0, 0 ;Posar a 1 el WREG si ACK = 0;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (!EA3 & EA2 & EA1 & EA0 & !OS)
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x07 ;Ara restem 7, si el resultat es 0 (EA[3..0] - 7 = 0) significa que EA[3..0] = 7
BTFSC PORTB, 0, 0 ;Comprovar que OS = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si OS = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (EA3 & !EA2 & !EA1 & !EA0)
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x08 ;Ara restem 8, si el resultat es 0 (EA[3..0] - 8 = 0) significa que EA[3..0] = 8
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (EA3 & !EA2 & !EA1 & EA0 & !CP)
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x09 ;Ara restem 9, si el resultat es 0 (EA[3..0] - 9 = 0) significa que EA[3..0] = 9
```

```
BTFSC PORTB, 2, 0 ;Comprovar que CP = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si CP = 1;
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (EA3 & !EA2 & EA1 & !EA0)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x0A ;Ara restem 10, si el resultat es 0 (EA[3..0] - 10 = 0) significa que EA[3..0] = 10
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

;Minterm (EA3 & !EA2 & EA1 & EA0)
 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomes treballar amb els quatre ultims bits, apliquem la mascara [00001111]
SUBLW 0x0B ;Ara restem 11, si el resultat es 0 (EA[3..0] - 11 = 0) significa que EA[3..0] = 11
SUBLW 0x00 ;si WREG - 0 = 0 llavors STATUS, Z = 1
BTFSC STATUS, Z, 0
BSF RESULT, 0, 0 ;si STATUS, Z = 1 posem RESULT a 1, per indicar que el midterm s'ha completat

BTFSC RESULT, 0, 0
BSF LATC, 4, 0 ;Si RESULT = 1 llavors EF1 = 1
BTFSS RESULT, 0, 0
BCF LATC, 4, 0 ;Si RESULT = 0 llavors EF1 = 0



;-------------------------------------------------------------------------;
;          Equacio de sortida per R_V = (!EA1 & !EA0) | (EA1 & !EA0)      ;
;-------------------------------------------------------------------------;
CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
   ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x00 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1
```

```
MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en ell
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00000011]
SUBLW 0x08 ;Ara restem 2, si el resultat Ã©s 0 (EA[1..0] - 2 = 0) significa que EA[1..0] = 2
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 2 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

 ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
BTFSS RESULT, 0, 0
BSF LATB, 4, 0 ;Si RESULT = 0 llavors A = 1
BTFSC RESULT, 0, 0
BCF LATB, 4, 0 ;Si RESULT = 1 llavors A = 0

;-----------------------------------------------------------------------;
;           Equacio de sortida per R_R_M = (!EA1 & !EA0) | (EA1 & !EA0)         ;
;-----------------------------------------------------------------------;
CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x00 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en ell
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00000011]
SUBLW 0x0A ;Ara restem 2, si el resultat Ã©s 0 (EA[1..0] - 2 = 0) significa que EA[1..0] = 2
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 2 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

 ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
BTFSS RESULT, 0, 0
BSF LATA, 7, 0 ;Si RESULT = 0 llavors A = 1
BTFSC RESULT, 0, 0
BCF LATA, 7, 0 ;Si RESULT = 1 llavors A = 0

;-----------------------------------------------------------------------;
;           Equacio de sortida per En_E0 = (!EA1 & !EA0) | (EA1 & !EA0)     ;
;-----------------------------------------------------------------------;
```

```
    CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

     MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest
registre
        ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara
[00001111]
    SUBLW 0x00 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
    BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
    BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

    MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en ell
        ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara
[00000011]
    SUBLW 0x01 ;Ara restem 2, si el resultat Ã©s 0 (EA[1..0] - 2 = 0) significa que EA[1..0] = 2
    BTFSC STATUS, Z, 0 ;Si EA[1..0] - 2 = 0, llavors STATUS, Z = 1
    BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

     ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o
mÃ©s minterms, si Ã©s 0, no s'ha completat cap
    BTFSC RESULT, 0, 0
    BSF LATD, 0, 0 ;Si RESULT = 0 llavors A = 1
    BTFSS RESULT, 0, 0
    BCF LATD, 0, 0 ;Si RESULT = 1 llavors A = 0

       ;-------------------------------------------------------------------------;
    ;           Equacio de sortida per En_E1 = (!EA1 & !EA0) | (EA1 & !EA0)     ;
    ;-------------------------------------------------------------------------;
    CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

     MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest
registre
        ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara
[00001111]
    SUBLW 0x00 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
    BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
    BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

     ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o
mÃ©s minterms, si Ã©s 0, no s'ha completat cap
    BTFSC RESULT, 0, 0
    BSF LATD, 1, 0 ;Si RESULT = 0 llavors A = 1
    BTFSS RESULT, 0, 0
    BCF LATD, 1, 0 ;Si RESULT = 1 llavors A = 0

    ;-------------------------------------------------------------------------;
```

```
;           Equacio de sortida per Cont = (!EA1 & !EA0) | (EA1 & !EA0)     ;
;-------------------------------------------------------------------------;
CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x02 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x04 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x07 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

 ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
BTFSS RESULT, 0, 0
BSF LATD, 2, 0 ;Si RESULT = 0 llavors A = 1
BTFSC RESULT, 0, 0
BCF LATD, 2, 0 ;Si RESULT = 1 llavors A = 0

   ;-------------------------------------------------------------------------;
;           Equacio de sortida per G_V = (!EA1 & !EA0) | (EA1 & !EA0)     ;
;-------------------------------------------------------------------------;
CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

 MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x0B ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
```

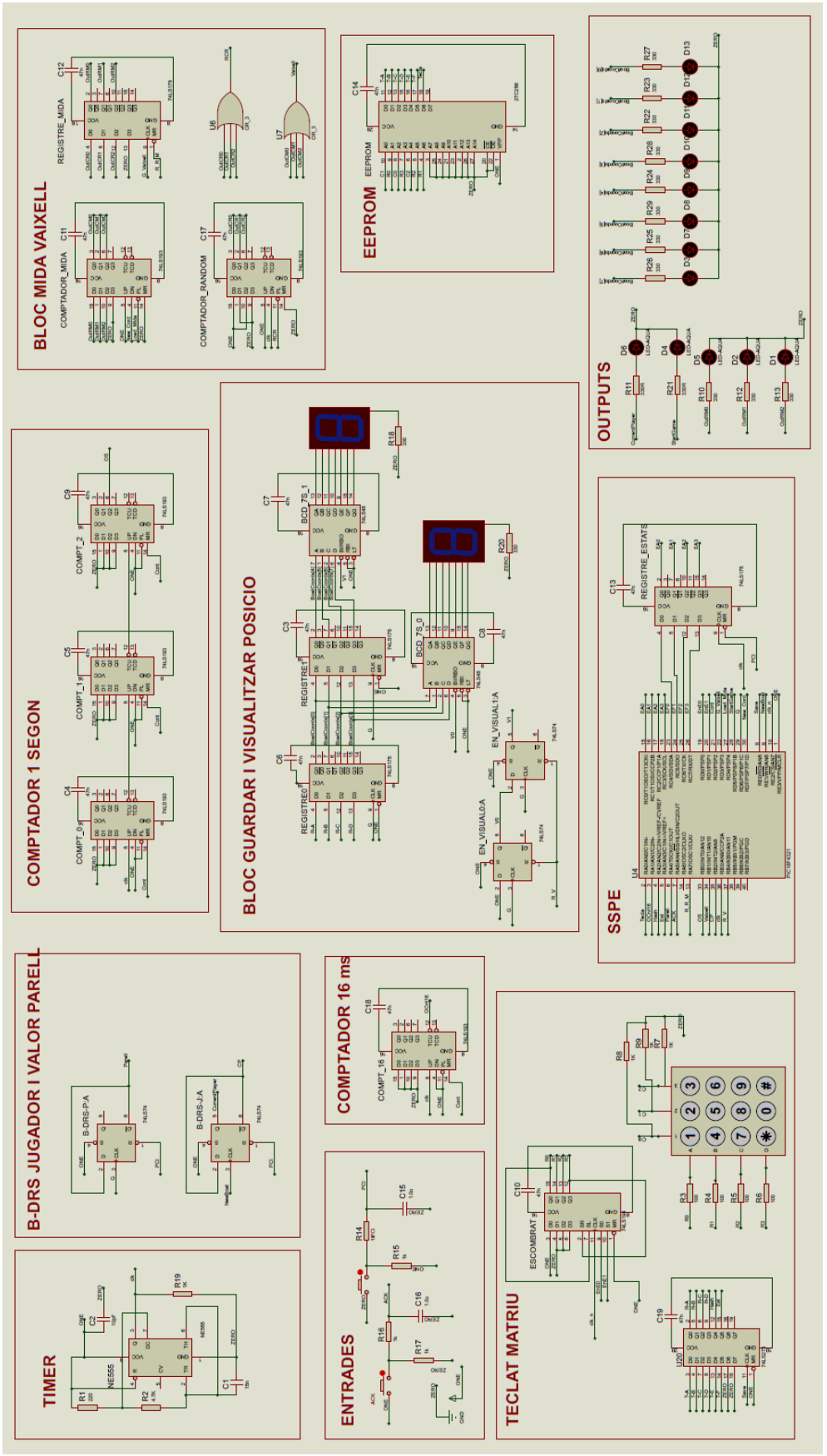
```
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
BTFSC RESULT, 0, 0
BSF LATD, 3, 0 ;Si RESULT = 0 llavors A = 1
BTFSS RESULT, 0, 0
BCF LATD, 3, 0 ;Si RESULT = 1 llavors A = 0

;-------------------------------------------------------------------------;
;           Equacio de sortida per L_M =    ;
;-------------------------------------------------------------------------;
CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x0B ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSC PORTB, 3, 0 ;Comprovar que clk = 0;
BSF WREG, 0, 0 ;Posar a 1 el WREG si clk = 1;
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
SUBLW 0x09 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

;Ara nomes hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
BTFSS RESULT, 0, 0
BSF LATD, 4, 0 ;Si RESULT = 0 llavors A = 1
BTFSC RESULT, 0, 0
BCF LATD, 4, 0 ;Si RESULT = 1 llavors A = 0


;-------------------------------------------------------------------------;
;           Equacio de sortida per S_G = (!EA1 & !EA0) | (EA1 & !EA0)     ;
;-------------------------------------------------------------------------;
CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions
```

```
    MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
      ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
   SUBLW 0x0C ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
   BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
   BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

    ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
   BTFSC RESULT, 0, 0
   BSF LATD, 5, 0 ;Si RESULT = 0 llavors A = 1
   BTFSS RESULT, 0, 0
   BCF LATD, 5, 0 ;Si RESULT = 1 llavors A = 0


   ;-------------------------------------------------------------------------;
   ;           Equacio de sortida per G = (!EA1 & !EA0) | (EA1 & !EA0)     ;
   ;-------------------------------------------------------------------------;
   CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

    MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
      ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
   SUBLW 0x05 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
   BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
   BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

    ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
   BTFSC RESULT, 0, 0
   BSF LATD, 6, 0 ;Si RESULT = 0 llavors A = 1
   BTFSS RESULT, 0, 0
   BCF LATD, 6, 0 ;Si RESULT = 1 llavors A = 0


   ;-------------------------------------------------------------------------;
   ;           Equacio de sortida per N_C = (!EA1 & !EA0) | (EA1 & !EA0)     ;
   ;-------------------------------------------------------------------------;
   CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

    MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
```

```
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
   SUBLW 0x06 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
   BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
   BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

    ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
   BTFSS RESULT, 0, 0
   BSF LATD, 7, 0 ;Si RESULT = 0 llavors A = 1
   BTFSC RESULT, 0, 0
   BCF LATD, 7, 0 ;Si RESULT = 1 llavors A = 0



   ;-------------------------------------------------------------------------;
   ;           Equacio de sortida per Save = (!EA1 & !EA0) | (EA1 & !EA0)     ;
   ;-------------------------------------------------------------------------;
   CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

    MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
   SUBLW 0x03 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
   BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
   BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

    ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o mÃ©s minterms, si Ã©s 0, no s'ha completat cap
   BTFSC RESULT, 0, 0
   BSF LATE, 0, 0 ;Si RESULT = 0 llavors A = 1
   BTFSS RESULT, 0, 0
   BCF LATE, 0, 0 ;Si RESULT = 1 llavors A = 0



   ;-------------------------------------------------------------------------;
   ;           Equacio de sortida per NewBoat = (!EA1 & !EA0) | (EA1 & !EA0)     ;
   ;-------------------------------------------------------------------------;
   CLRF RESULT, 0 ;Resetejem RESULT per les futures operacions

    MOVF PORTC, 0, 0 ;Movem els 8 bits del PORTC al WREG per poder realitzar operacions en aquest registre
    ANDLW 0x0F ;Per tal de nomÃ¨s treballar amb els dos ultims bits, apliquem la mÃ scara [00001111]
   SUBLW 0x09 ;Ara restem 0, si el resultat Ã©s 0 (EA[3..0] - 0 = 0) significa que EA[3..0] = 0
```

```
    BTFSC STATUS, Z, 0 ;Si EA[1..0] - 0 = 0, llavors STATUS, Z = 1
    BSF RESULT, 0, 0 ;Si STATUS, Z = 1 llavors RESULT = 1

     ;Ara nomÃ©s hem de comprovar si el RESULT Ã©s un 1 o un 0, si Ã©s un 1, s'han completat un o
mÃ©s minterms, si Ã©s 0, no s'ha completat cap
    BTFSC RESULT, 0, 0
    BSF LATE, 1, 0 ;Si RESULT = 0 llavors A = 1
    BTFSS RESULT, 0, 0
    BCF LATE, 1, 0 ;Si RESULT = 1 llavors A = 0



    ;-------------------------------------------------------------------------;
    ;                              Porta logica Not                           ;
    ;-------------------------------------------------------------------------;

    ;Implementem una porta NOT --> n_clk = !clk
    BTFSS PORTB, 3, 0
    BSF LATE, 2, 0 ;Si B = 1, saltem la linea, si B = 0 llavors Z = 1
    BTFSC PORTB, 3, 0
    BCF LATE, 2, 0 ;Si B = 0, saltem la linea, si B = 1 llavors Z = 0

    GOTO LOOP
    END
```

5. Esquema elèctric



6. Problemes observats

En aquesta pràctica els únics problemes que heu tingut han estat a l'hora d'implementar la placa físicament i alguna cosa en la part de l'assembler, tota la resta ens ha anat funcionat a mesura que anàvem avançant amb la pràctica. En fer les simulacions en els proteus, no ens varem, trobar cap problema, ja que ens simulava correctament.

El primer problema va estar amb els cables, ja que en diverses ocasions ens generaven un mal contacte entre els pins connectats per cable, fent que no arribes bé el senyal o bé provoques un curtcircuit sense que es pogués apreciar visualment.

També vàrem tenir un problema amb un pin nombre aleatori, ja que no actuava llevat que punxéssim amb l'oscil·loscopi, i tot i provar de canviar el cable, la soldadura i altres mètodes, vàrem optar per posar un pulldown en aquell pin i llavors vàrem aconseguir que funciones amb normalitat.

Un altre problema que vàrem tenir amb l'assembler, va ser que els pins RA2 i RA3 actuaven com Voltatge de referència, és per això que vàrem afegir unes línies de codi (línia 27) per a forçar que funcionin com a pins normals.
Les línies són:

```
movlw 0x0F
movwf ADCON1, 0
```

7. Conclusions

Un cop finalitzada la practica, hem estat capaços d'implementar la nostra solució de la pràctica 1 fase 1, essent capaços de programar amb assembler una màquina d'estats i fent que pugui funcionar amb la resta de hardware soldat a la placa.

8. Planificació

Hores estimades i planificació:

	Hores estimades	18/09/23	25/09/23	2/10/23	9/10/23	16/10/23	23/10/23
Comprensió de l'enunciat	2						
Disseny sobre paper	4						
Volcat en proteus	6						
Soldar	20						
Debugar	6						
Programació Maquina d'estats	8						
Realització de la memòria	8						

Hores totals dedicades:

	Hores estimades	18/09/23	25/09/23	2/10/23	9/10/23	16/10/23	23/10/23
Comprensió de l'enunciat	2						
Disseny sobre paper	2						
Volcat en proteus	3						
Soldar	30						
Debugar	8						
Programació Maquina d'estats	4						
Realització de la memòria	8						

Per a realitzar la pràctica vàrem estimar unes hores per a la part de comprensió, disseny i volcat al proteus, que a l'hora de dissenyar-ho va resultar ser menys temps del planificat.
En canvi, per a la part de la soldadura i debugar, vàrem trigar més de l'esperat, degut a que al debugar per blocs trigàvem més, ja que ens asseguràvem que cada bloc funciones correctament abans de passar a soldar el següent.
En canvi, per a la màquina d'estats va ser el contrari, va ser més ràpid del que ens vàrem imaginar, ja que a l'inici va costar entendre com funcionava, però un cop ho vàrem entendre, ja tot era repetitiu i com que ja teníem les equacions, era més senzill de programar.
Finalment, a la memòria, li hem dedicat les hores que ja vàrem estimar.