UNIVERSITAT OBERTA DE CATALUNYA

Enginyeria Tècnica de Telecomunicació esp.

Telemàtica

DISSENY D'UN EXPANSOR D'ENTRADES I SORTIDES ANALÒGIQUES PER MODBUS RTU SOBRE RS485.

Alumne/a: Iban Garcia Del Sol

Dirigit per: Carlos Alberto Pacheco Navas

CURS 2012-13 (Febrer)

Índex de continguts.

Índex de figures.

Índex de taules.

1. Introducció.

Vivim en un món rodejat de màquines, ja siguin petites o grans, les quals necessitem per dur a terme moltíssimes tasques tant siguin quotidianes, d'oci, de treball.., per comunicar-nos amb elles necessita d'algun sistema(interfície) que ens permeti interactuar amb elles.

En aquest TFC el que veurem és com fer un adaptador d'interfície, en aquest cas per processos industrials on moltes vegades fa falta connectar diferents sistemes entre ells, per aconseguir-ho utilitzarem el bus RS-485 juntament amb el protocol ModBus que són dels més utilitzats en processos industrials.

Tot seguit veurem una mica d'introducció a dues parts del TFC com són el bus RS-485 i el protocol Modbus.

1.0.1 RS-485.

El bus RS485 està definit com un sistema de transmissió que es fa servir bastant al món industrial ja que ofereix alta velocitat per transmetre a llarga distància a més a més de tenir molta tolerància al soroll.

La longitud màxima a la que pot funcionar son 1,2km tot i que a aquesta distància no és tan ràpid, ja que la seva velocitat baixa als 100kps, mentre que en distàncies curtes pot arribar als 35Mbps.

El secret de la seva tolerància al soroll radica en el fet que per transmetre les dades utilitza mínim dos fils, i per cadascun passa la mateixa informació però amb la polaritat invertida d'un cable respecte de l'altre, es sol utilitzat una tensió de 6V, llavors totes les interferències que hi hagi a la transmissió afectaran de igual manera tant a un cable com a l'altre i un cop arribi la senyal al seu destí al processar el senyal el soroll es cancel·la,

La seva instal·lació pot ser tant en 2 fils com en 4 fils i d'aquesta dependrà si és *half dúplex*, quan siguin 2, o *full dúplex* quan utilitzi els 4 i està basat en una arquitectura de mestre/esclau.

1.0.2 Modbus.

El protocol de comunicació Modbus està basat en una arquitectura mestre/esclau com el bus RS485 i és molt utilitzat per la comunicació en dispositius electrònics industrials ja que va ser dissenyat per aparells PLC.

Aquest protocol ha obtingut bona part del seu èxit degut a que permet controlar una xarxa de dispositius sense gaires maldecaps ja que gaudeix d'una implementació fàcil, fa bona parella amb el bus RS485 i és *obert*, per tant qualsevol el pot utilitzar i modificar per les seves necessitats sense haver de pagar *royalties*.

Hi ha dos tipus de Modbus, un és l'ASCII i l'altre que utilitzarem nosaltres l'RTU, aquest últim té un funcionament força senzill ja que per la durada d'un missatge l'envia tot seguit sense espais i quan troba un espai esta indicant que es tracta d'un missatge nou, el nombre de bits que utilitza, és 10, s'envia com una cadena de 8bits amb un bit a davant i un al darrera que fan de start/stop, a més a més a cada missatge incorpora un Codi de Redundància Cíclic que serveix per comprovar que el missatge s'ha rebut correctament sense modificacions.

1.1 Presentació.

El motiu de l'elecció d'aquesta àrea de Treball Final de Carrera ha sigut per diferents factors.

Un d'ells és degut a que es tractava d'un projecte més aviat pràctic, tot i que de vegades implica més feina sol resultar més distret a la vegada que satisfactori, en el meu cas.

Un altre és al fet que sempre he tingut curiositat per saber com funcionen les coses i aquest TFC em permetia aprendre una mica més del tot el tema d'electrònica.

També necessitava un repte, sortir de la meva zona de confort i espavilar-me pròpiament dit fent alguna cosa de la qual tenia més aviat poc coneixement en vers altres disciplines.

1.1.1 Definició del projecte.

El projecte tracta de dissenyar un adaptador d'interfície en la qual es connectaran quatre entrades analògiques que es podran llegir, les arribaran en un rang de 0-10V i s'haurà de fer una conversió A/D de 10 bits per poder-la llegir. També disposarà de dues sortides les quals generaran senyals analògics amb un rang de sortida de 0-10V oferint 10bits de precisió i que puguin donar un màxim de corrent de 200mA cadascuna.

També s'ha de definir el sistema d'alimentació que utilitzarà el dispositiu i tot això es controlarà mitjançant la interfície RS-485 juntament amb el procotol ModBus RTU, amb la qual cosa farà falta programar un microxip per què controli l'adaptador.

El diagrama de blocs del projecte amb les diferents parts que el composen seria aquest:

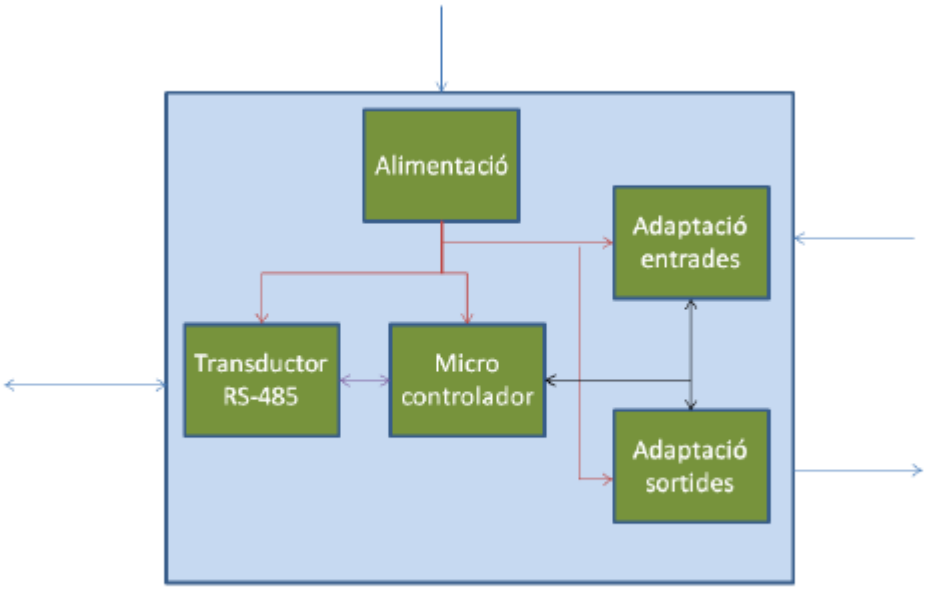


Figura 1. Diagrama de blocs de l'adaptador

Figura 1. Diagrama de blocs de l'adaptador

1.1.2 Objectius.

Els objectius que componen el TFC són:

- Analitzar una problemàtica concreta que podem trobar-nos com Enginyer Tèc. de Telecomunicació i donar solució a aquesta.

- Conèixer en profunditat el protocol Modbus i d'interfície RS-485, que són de les més utilitzades en l'àmbit industrial.

- Dissenyar circuits d'adaptació d'entrades i sortides analògiques/digitals.

- Dissenyar fonts d'alimentació.

- Programació de microxips PIC.

- Disseny de plaques de circuits PCB.

1.2 Planificació.

Per la realització del projecte, es planificaran les tasques a realitzar amb la seva durada aproximada i a més a més marcarem uns objectius com a fites.

1.2.1 Tasques.

La planificació de les tasques on indicarem tot el que hem de fer per cadascuna, dividit en activitats, a més a més de decidir si tenen dependència d'altres tasques.

1. Pla de treball	***Predecessores***	***Dedicació en Hores***
1.1 Lectura de l'enunciat del projecte		2h
1.2 Buscar informació, recopilar-la i fer una primera lectura sobre protocol Modbus i interfície RS-485 per tenir idea del que haurem d'aconseguir fer.		4h
1.3 Esborrany del Pla de treball	1.1, 1.2	6h
1.4 Revisió de l'Esborrany i modificacions si calen.	1.3	2h
1.5 Entrega del Pla de treball	1.4	
2. Disseny dels circuits d'adaptació d'entrades/sortides analògiques/digitals.		
2.1 Buscar informació sobre l'adaptació de senyals analògics a digitals mitjançant microxip PIC, tant de volts com de mA.		4h
2.2 Buscar informació sobre el funcionament del programa TINA-TI		2h
2.3 Disseny del circuit d'adaptació d'entrada analògica de tensions entre 0 i 10V.	2.1	6h
2.4 Proves del disseny amb TINA-TI	2.2	2h

2.5 Disseny del circuit d'adaptació d'entrada analògica del bucle de corrent 4-20mA passiu.	2.4	6h
2.6 Proves del disseny amb TINA-TI	2.5	2h
2.7 Disseny de les sortides digitals a analògiques amb tensió entre 0 i 10V i màxim de 200mA de corrent.	2.6	6h
2.8 Proves del disseny amb TINA-TI	2.7	2h
3. Disseny de la font d'alimentació		
3.1 Buscar informació sobre fonts d'alimentació		4h
3.2 Disseny de la font d'alimentació	3.1	6h
3.3 Proves del disseny amb TINA-TI	3.2	2h
4. Implementació del processador		
4.1 Buscar informació sobre diferents processadors per triar-ne un segons les especificacions que necessitem.		2h
4.2 Definició del mapa de memòria Modbus	4.1	6h
4.3 Documentació sobre el mapa de memòria que hem definit	4.2	4h
4.4 Entregar l'esborrany de la PAC2	2,3,4.3	4h
4.5 Revisió/modificació de l'esborrany	4.4	2h
4.6 Entrega de la PAC2	4.5	
4.7 Buscar informació/tutorials pel funcionament del programari MPLAB	4.1	2h
4.8 Disseny del programari de control del microxip	4.1, 4.4	25h
4.9 Proves del disseny del programari amb MPLAB	4.5	5h
5. Implementació de la interfície de sortida del microxip a RS-485.	***1.2***	
5.1 Proves de la interfície port sèrie a RS-485	5	1h

6. Disseny layout de la placa PCB		
6.1 Buscar informació sobre disseny de *layouts* PCB.		2h
6.2 Buscar informació/tutorials sobre el programari Eagle de Cadsoft		2h
6.3 Implementació del disseny del nostre projecte.	6.1	15h
6.4 Proves del disseny *layout* de la placa PCB.	6.3	2h
6.5 Lliurar l'esborrany de la PAC3	6.4	4h
6.6 Revisió/modificació de l'esborrany	6.5	2h
6.7 Lliurament PAC3	6.6	
7. Entrega del Projecte	***1,2,3,4,5,6***	
7.1 Realització de la memòria del projecte		6h
7.2 Realització del vídeo presentació (20min màxim)	7.1	10h
7.3 Lliurament de l'esborrany de la memòria i el vídeo	7.2, 7.3	
7.4 Revisió/modificació de la memòria i el vídeo	7.3	4h
7.5 Lliurament de la memòria i el vídeo	7.4	
7.6 Debat		

Taula 1. Taula de Tasques.

En total em surten unes 154 hores, per la feina de la PAC2 em surten 60 hores i contant que hi ha aproximadament uns 30 dies, serien 15 sessions de 4 hores durant el mes, això em deixaria temps per les altres assignatures.

Per la PAC3 també surten unes 60 hores de feina i coincideix com la PAC2 que hi han uns 30 dies per l'entrega, per tant també 15 sessions de 4 hores.

Finament per la darrera entrega quedarien 20 hores de feina per la qual hi han 15 dies per tant hi ha una mica més de marge que per les altres entregues, serien 5 sessions de 4 hores així també queda temps per preparar exàmens i proves validació.

1.2.2 Fites.

Per tal de dur a terme el projecte marcarem les dates clau dels lliuraments que hem de fer per l'avaluació continuada, ja que les fites solen ser parts lliurables del projecte.

Entrega del pla del treball	02/10/2012
Entrega de la PAC2.	06/11/2012
Entrega de la PAC3.	11/12/2012
Entrega de la memòria i el vídeo.	07/01/2013
Debat Virtual	25/01/2013

1.2.3 Diagrama de Gantt.

En aquest apartat veurem el Diagrama amb les tasques definides de l'apartat anterior amb una aproximació d'hores necessàries per fer-les i la durada aproximada en dies.

	Nombre de tarea	Duración	Comienzo	Fin	Predec
1	**Pla de treball**	**10 días**	**mié 19/09/12**	**mar 02/10/12**	
2	Lectura de l'enunciat del projecte	1 día	jue 20/09/12	jue 20/09/12	
3	Buscar informació, recopilar-la i fer una primera lectura sobre protocc	2 días	sáb 22/09/12	lun 24/09/12	2
4	Esborrany del Pla de treball	3 días	mar 25/09/12	jue 27/09/12	2;3
5	Revisió de l'Esborrany i modificacions si calen.	1 día	sáb 29/09/12	sáb 29/09/12	4
6	Entrega del Pla de treball	1 día	mar 02/10/12	mar 02/10/12	5
7	**Disseny dels circuits d'adaptació d'entrades/sortides analògiques/digitals.**	**10 días**	**mié 03/10/12**	**mar 16/10/12**	**1**
8	Buscar informació sobre l'adaptació de senyals analògics a digitals m	1 día	mié 03/10/12	mié 03/10/12	
9	Buscar informació sobre el funcionament del programa TINA-TI	1 día	jue 04/10/12	jue 04/10/12	
10	Disseny del circuit d'adaptació d'entrada analògica de tensions entre (	2 días	jue 04/10/12	vie 05/10/12	8
11	Proves del disseny amb TINA-TI	1 día	lun 08/10/12	lun 08/10/12	10
12	Disseny del circuit d'adaptació d'entrada analògica del bucle de correi	2 días	mar 09/10/12	mié 10/10/12	11
13	Proves del disseny amb TINA-TI	1 día	jue 11/10/12	jue 11/10/12	12
14	Disseny de les sortides digitals a analògiques amb tensió entre 0 i 10V	2 días	vie 12/10/12	lun 15/10/12	13
15	Proves del disseny amb TINA-TI	1 día	mar 16/10/12	mar 16/10/12	14
16	**Disseny de la font d'alimentació**	**4 días**	**mié 17/10/12**	**lun 22/10/12**	**7**
17	Buscar informació sobre fonts d'alimentació	1 día	mié 17/10/12	mié 17/10/12	
18	Disseny de la font d'alimentació	2 días	jue 18/10/12	vie 19/10/12	17
19	Proves del disseny amb TINA-TI	1 día	lun 22/10/12	lun 22/10/12	18

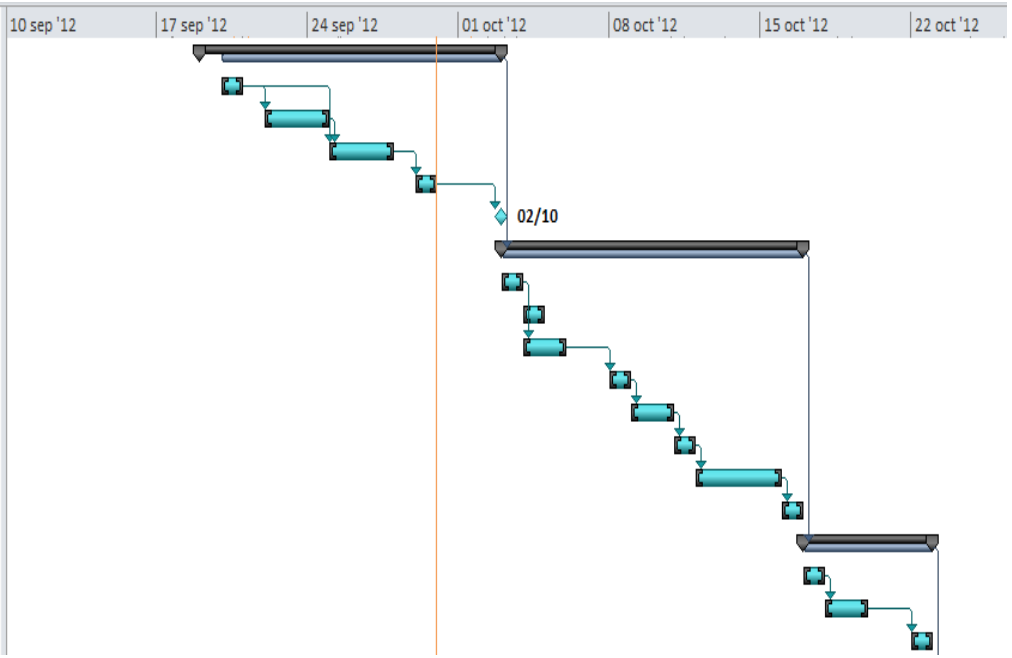


Figura 2. Diagrama de Gantt 1

20	Implementació del processador	23 días	mar 23/10/12	jue 22/11/12	16
21	Buscar informació sobre diferents processadors per triar-ne un segons	1 día	mar 23/10/12	mar 23/10/12	
22	Definició del mapa de memòria Modbus	4 días	mié 24/10/12	lun 29/10/12	21
23	Documentació sobre el mapa de memòria que hem definit	2 días	mar 30/10/12	mié 31/10/12	22
24	Entregar l'esborrany de la PAC2	2 días	jue 01/11/12	vie 02/11/12	23
25	Revisió/modificació de l'esborrany	1 día	lun 05/11/12	lun 05/11/12	24
26	Entrega de la PAC2	1 día	mar 06/11/12	mar 06/11/12	25
27	Buscar informació/tutorials pel funcionament del programari MPLAB	1 día	mié 07/11/12	mié 07/11/12	26
28	Disseny del programari de control del microxip	10 días	jue 08/11/12	mié 21/11/12	21;27
29	Proves del disseny del programari amb MPLAB	1 día	jue 22/11/12	jue 22/11/12	28
30	Implementació de la interfície de sortida del microxip a RS-485.	3 días	vie 23/11/12	mar 27/11/12	20
31	Implementació de la interfície de sortida del microxip a RS-485.	2 días	vie 23/11/12	lun 26/11/12	
32	Proves de la interfície port sèrie a RS-485	1 día	mar 27/11/12	mar 27/11/12	31

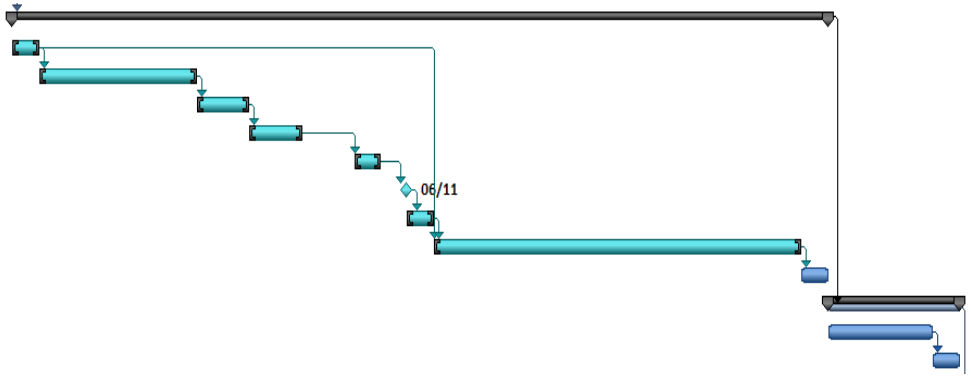


Figura 3. Diagrama de Gantt 2

33	Disseny layout de la placa PCB	10 días	mié 28/11/12	mar 11/12/12	30
34	Buscar informació sobre disseny de layouts PCB.	1 día	mié 28/11/12	mié 28/11/12	
35	Buscar informació/tutorials sobre el programari Eagle de Cadsoft	1 día	mié 28/11/12	mié 28/11/12	
36	Implementació del disseny del nostre projecte.	4 días	jue 29/11/12	mar 04/12/12	34
37	Proves del disseny layout de la placa PCB.	1 día	mié 05/12/12	mié 05/12/12	36
38	Lliurar l'esborrany de la PAC3	2 días	jue 06/12/12	vie 07/12/12	37
39	Revisió/modificació de l'esborrany	1 día	lun 10/12/12	lun 10/12/12	38
40	Lliurament PAC3	1 día	mar 11/12/12	mar 11/12/12	39
41	Entrega del Projecte	20 días	mar 11/12/12	lun 07/01/13	40
42	Realització de la memòria del projecte	7 días	mar 11/12/12	mié 19/12/12	
43	Realització del vídeo presentació (20min màxim)	7 días	jue 20/12/12	vie 28/12/12	42
44	Lliurament de l'esborrany de la memòria i el vídeo	1 día	lun 31/12/12	lun 31/12/12	42;43
45	Revisió/modificació de la memòria i el vídeo	4 días	mar 01/01/13	vie 04/01/13	44
46	Lliurament de la memòria i el vídeo	1 día	lun 07/01/13	lun 07/01/13	45
47	Debat	3 días	mié 23/01/13	vie 25/01/13	

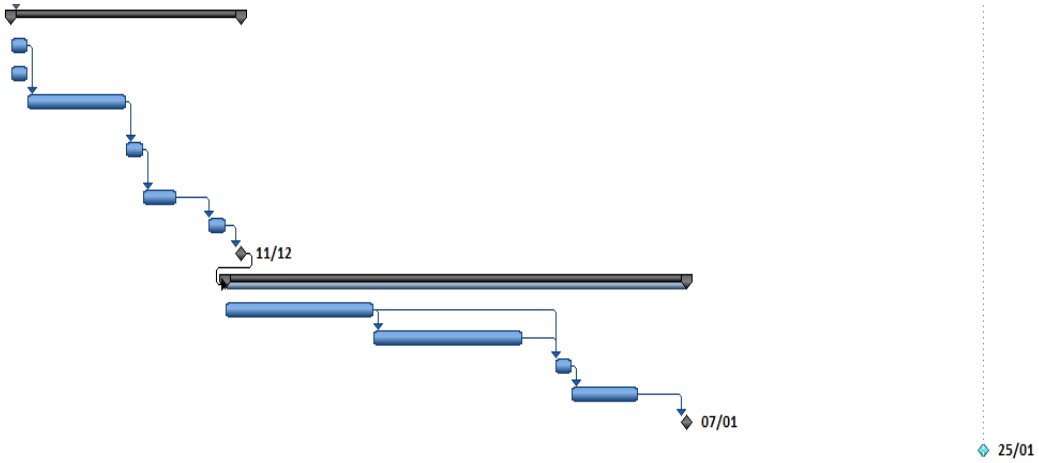


Figura 4. Diagrama de Gantt 3

1.3 Incidències i Riscos.

Els possibles incidents i riscos que ens podem trobar al llarg del projecte i una possible pla de contingències per solucionar-ho.

Avaria del punt de treball.

Actualment puc disposar d'un PC de sobretaula amb S.O Windows 7/Linux que és l'ordinador principal i un ordinador portàtil amb S.O. Mac OSX/Windows7, es fan còpies de seguretat cada dia mitjançant un disc dur extern que tinc connectat en xarxa.

En cas que tingui alguna averia amb el PC de sobretaula o el portàtil, sempre em quedarà l'altre per continuar treballant i a més a més tinc còpia de seguretat diària, en el disc dur extern per tant en principi no es perdria gaire feina.

Acumulació de PACS.

A més a més del TFC, estic cursant unes altres tres assignatures, a priori segons el calendari em coincideix l'entrega de la PAC2 del TFC amb una altra entrega i la PAC3 també coincideix amb una.

Com he comentat abans, puc disposar de 15 dies de vacances de la feina per poder avançar feina si fos necessari.

Possible viatge al desembre.

Podria ser que al desembre hagués de marxar de viatge una setmana, no està clar que hagi de marxar però és una possibilitat.

Per no perdre tants dies puc fer dues coses, podria agafar vacances la setmana abans per avançar la feina i a més a més al viatge em puc emportar l'ordinador portàtil i així poder fer alguna coseta dura la setmana.

1.4 Material.

En aquest apartat s'estableix el material que serà necessari per dur a terme el projecte, tant a nivell de Hardware com de Software.

Hardware:

Punt de treball estàndard de la UOC.

Software:

- Windows 7, serà el sistema operatiu que utilitzaré bàsicament per la compatibilitat dels programes, ja que puc trobar-me que algun no tingués versió de Mac OSX.

- Microsoft Word, per redactar els arxius de text, pla de treball, PACS, memòria...

- TINA-TI que farem servir per dissenyar els circuits i fer les proves pertinents.

- MPLAB amb el que programarem el microxip i farem les proves.

- Eagle de Cadsoft que servirà per dissenyar el *layout* de la placa PCB.

- Camparia amb el que farem el vídeo per la presentació.

2. Disseny de les entrades i sortides analògiques.

Per tal de fer la conversió analògica/digital al microxip hem d'adaptar les senyals perquè aquest les pugui processar, en aquest bloc veurem els diferents tipus d'adaptacions que necessitarem.

2.1 Entrades.

Al microxip arribaran 2 tipus diferents de senyals, una en volts i l'altra en ampers, la primera caldrà adaptar-la al rang que accepti el microxip, que n'hi ha de diferents tipus, el més comuns serien els de 0 a 3,3v i els de 0 a 5v, tot i que molts microxips incorporen una entrada de referència per tal de fer les mesures.

2.1.1 Entrades 0-10V.

En aquesta entrada el que ens caldrà serà fer un divisor de tensió de la senyal que ens arriba a 0-10v i fer que aquesta sigui de 0-5v, però hem de tenir en compte que no sabem com ens arriba aquesta senyal i per tant l'haurem d'aïllar per evitar així que funcioni malament el divisor de tensió a més d'evitar cremar el microxip per un excés de corrent.

Per aïllar la impedància utilitzarem dos Amplificadors Operacionals en mode buffer, el seu disseny és el següent:

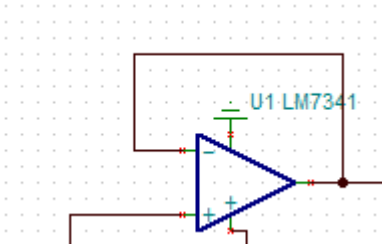


Figura 5. Amplificador Operacional mode buffer

Es tracta d'un amplificador no inversor el qual tindrà una impedància d'entrada molt gran i una impedància de sortida molt petita, de tal manera que aïllarà el nostre divisor de tensió, el model en concret triat és LM7341 que ja que es un amplificador operacional de propòsit general, serveix per tot i és *rail-to-rail*, que vol dir que la sortida de tensió que proporcionen es pràcticament igual a la seva entrada d'alimentació.

El voltatge de sortida el calcularem mitjançant el divisor de tensió utilitzant la fórmula següent:

$$Vo = \frac{R_2}{R_1 + R_2} * Vi$$

Per tant necessitem que:

$$\frac{R_2}{R_1 + R_2} = 0{,}5$$

Amb la qual cosa podem agafar unes resistències de 1kΩ, així doncs si a l'entrada tenim 10v a la sortida tindrem 5v.

$$Vo = \frac{1000}{1000 + 1000} * 10$$

$$Vo = 5v$$

El disseny del circuit d'adaptació serà:

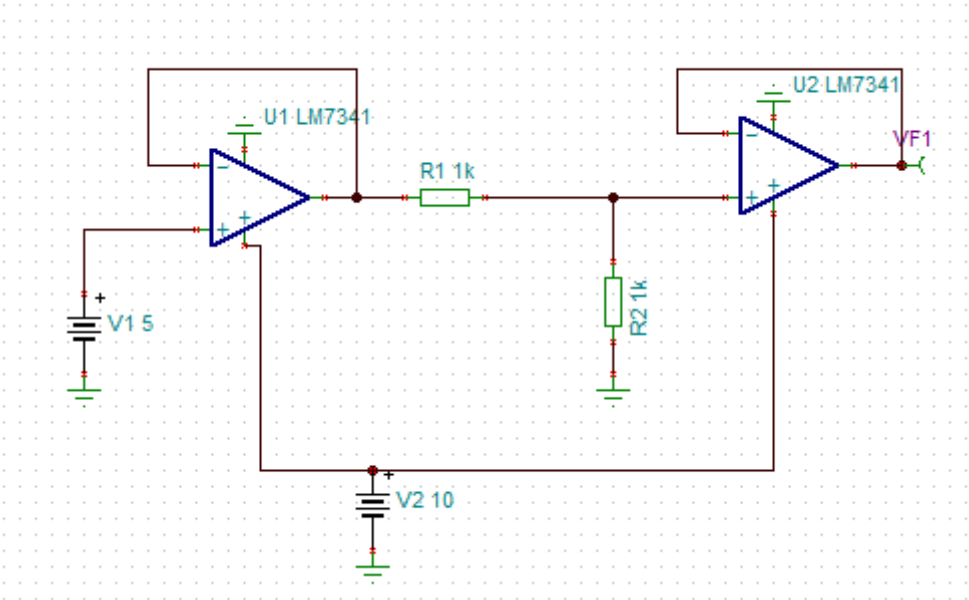


Figura 6. Circuit d'adaptació 0-10v a 0-5v

Veiem a la simulació que com ens queda una recta on el valor màxim son els 5V a la sortida quan a l'entrada tenim 10V.

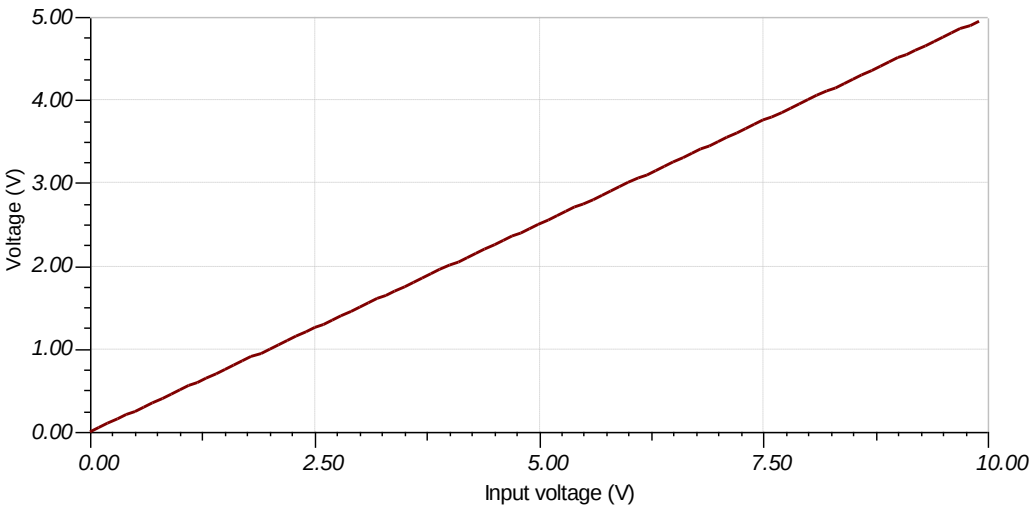


Figura 7. Simulació Circuit d'adaptació 0-10v a 0-5v.

2.1.2 Entrades 4-20mA.

En aquest cas ja que els microxips només treballen amb volts el que cal fer és convertir els Ampers en volts i llavors adaptar els volts a un rang de 0-5v, per tant aquest adaptador constarà de dues parts, a la primera utilitzarem un amplificador operacional per convertir el corrent de 4-20mA a 1-5V i a la segona part utilitzarem un amplificador operacional restador el qual restarà un volt als 1-5V i llavors amplificarà per obtenir el rang 0-5V.

El disseny del conversor de corrent a voltatge és el següent:

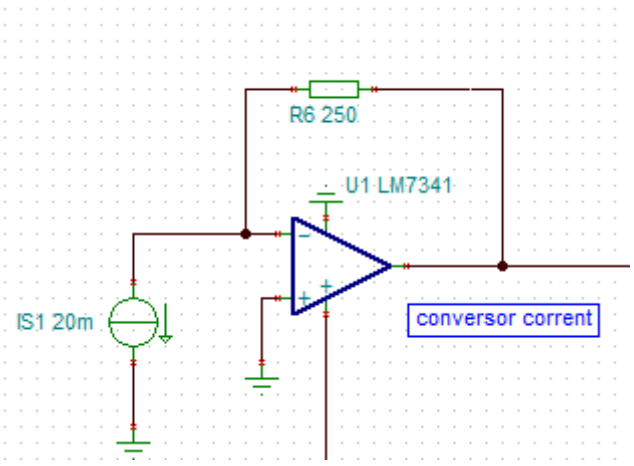


Figura 8. Conversor corrent a voltatge

El sistema per calcular la sortida ens el dóna la llei d'Ohm:

$$V = I * R$$

En el nostre cas volem 5v a la sortida quan a l'entrada tenim 20mA, per tant:

$$5 = 0{,}02 * R$$

$$R = \frac{5}{0{,}02} = 250\Omega$$

Quan a l'entrada tenim 4mA a la sortida tindre 1v:

$$V = 0{,}004 * 250$$

$$V = 1v$$

La segona part d'aquest circuit d'adaptació és la que ens restarà 1v al resultat anterior i amplificarà per arribar al rang 0-5V, utilitzant l'amplificador operacional restador.

El disseny dels amplificadors restadors és:

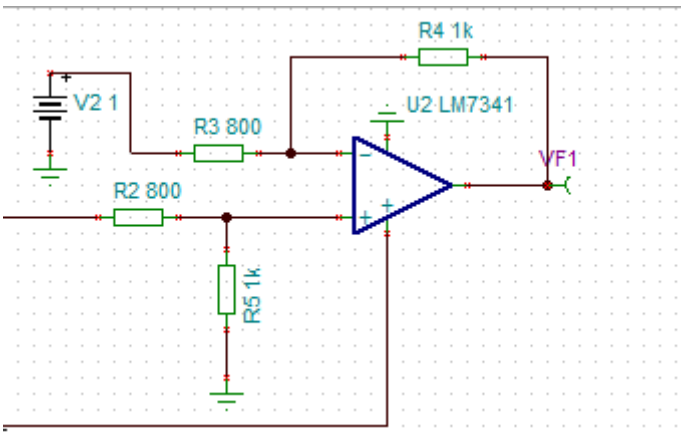


Figura 9. Amplificador Operacional Restador

Aquest circuit el que fa és restar l'entrada del pol positiu amb el negatiu i amplificar per la diferència de R2/R1, per aquest motiu al pol negatiu l'alimentarem amb una tensió de 1v.

$$Vo = (V_1 - V_2) * \frac{R_2}{R_1}$$

Per tant la divisió de les resistències hauran de tenir un valor de 1,25 per tal de convertir el rang 0-4v en 0-5v, podem agafar R_1=1000Ω i R_2=800Ω.

Comprovem en el cas que a l'entrada arribin 5v:

$$Vo = (5-1) * \frac{1000}{800}$$
$$Vo = 5v$$

Comprovem en el cas que a l'entrada arribi 1v:

$$Vo = (1-1) * \frac{1000}{800}$$
$$Vo = 0v$$

La simulació del circuit al TINA-TI alimentant l'amplificador operacional LM7341 amb 10v, l'amplificador és el mateix utilitzat a l'anterior circuit d'adaptació.

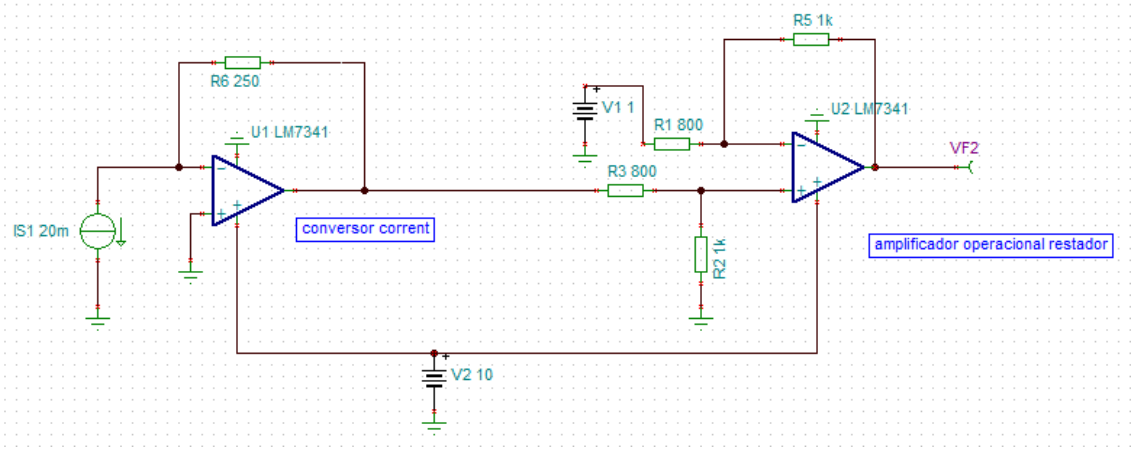


Figura 10. Adaptador 4-20mA a 0-5V

Veiem com ens quedaria el circuit muntat i podem comprovar a més la simulació quan a l'entrada hi tenim els 20mA a la sortida ens dóna 5v.

Si fem la simulació veiem com comencem a tenir corrent apartir dels 4mA fins els 20mA que és on agafa el valor màxim de 5V.

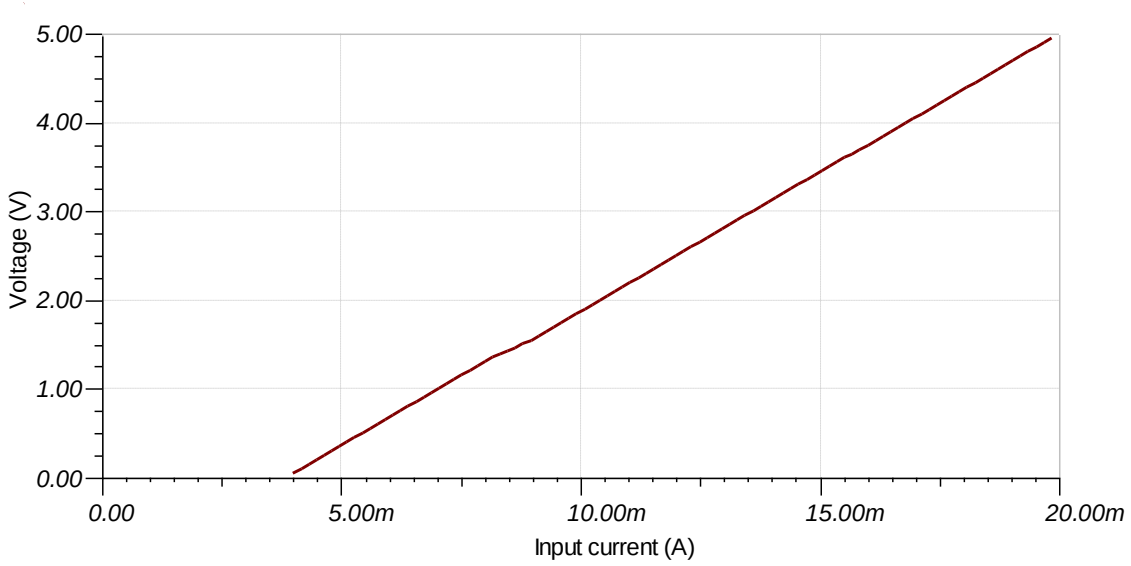


Figura 11. Simulació del circuit d'adaptació 4-20mA

2.2 Sortides.

Per l'expansor faran falta dues sortides analògiques, totes dues amb les mateixes especificacions les quals han de generar una tensió de 0-10v i suportar 200mA de corrent.

2.2.1 Sortida 0-10V.

Per dissenyar aquest circuit utilitzaré les sortides PWM del microxip les quals generen un pols amb la freqüència que li indiquem, en el nostre cas li indicarem que volem 10bits.

Els pols PWM per ell mateix no genera una sortida de tensió, per generar-la utilitzaré un circuit RC, el qual consisteix en una resistència en sèrie amb un condensador el qual generarà la tensió.

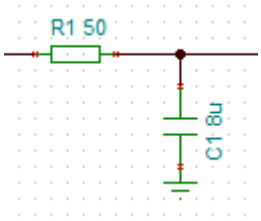


Figura 12. Circuit RC

Per donar el valor al condensador utilitzarem la següent equació:

$$RC = \frac{1}{2 * \pi * f}$$

On el valor de la freqüència el triem en funció de que la senyal que ens generi sigui estable i no tingui canvis bruscs o que no trigui molt a arribar al valor desitjat, en el nostre projecte no es comenta i he optat per una freqüència petita perquè la senyal sigui estable ja que la sortida el que si ha de tenir són 10bits de precisió, per tant si la senyal oscil.la gaire perdrem precisió, per la resistència podem agafar un valor estàndard com pot ser 5kΩ i llavors buscar el valor del condensador amb la fórmula anterior, a una freqüència de 10Hz.

$$5000C = \frac{1}{2 * \pi * 10}$$
$$C = 3{,}183\mu$$

Si veiem que pel valor del condensador que obtenim no n'hi han al mercat, amb la mateixa fórmula podem buscar la resistència apartir d'un valor en concret de condensador.

Un cop tenim el circuit RC i donat que el microxip treballa amb una tensió de 5V aquesta serà la que tindrem a la sortida, per tant haurem d'afegir al circuit un amplificador operacional no inversor per augmentar per 2 els valors de la sortida, amb la qual cosa el nostre circuit complert quedarà de la següent manera.

En aquest cas l'amplificador operacional que utilitzem és el AD8397 de Analog Devices, ja que per la sortida necessitem 200mA i el LM7341 no arriba a donar aquest valor segons el seu *datasheet*, tot i que en les simulacions si que funciona correctament.

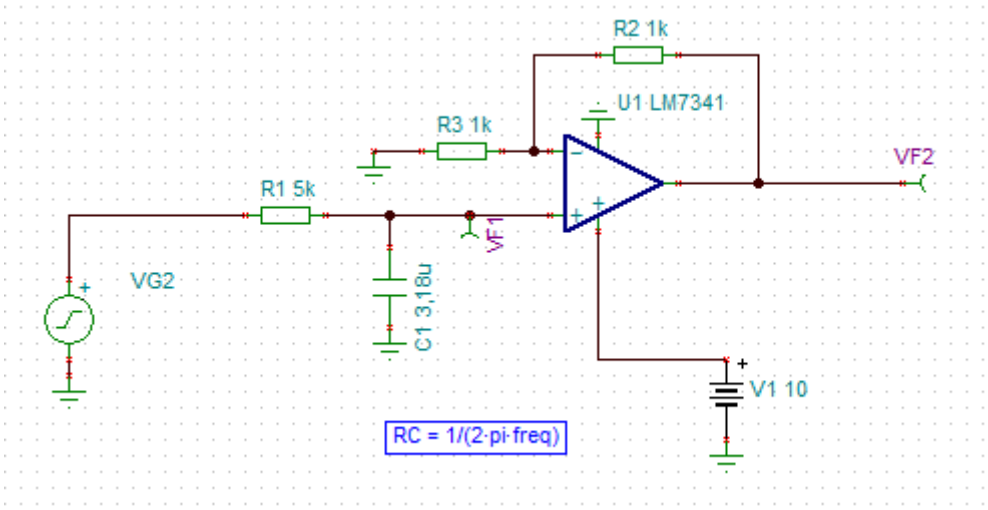


Figura 13. Sortida analògica 0-10V

La simulació del circuit treballant al 99% del PWM seria aquesta:

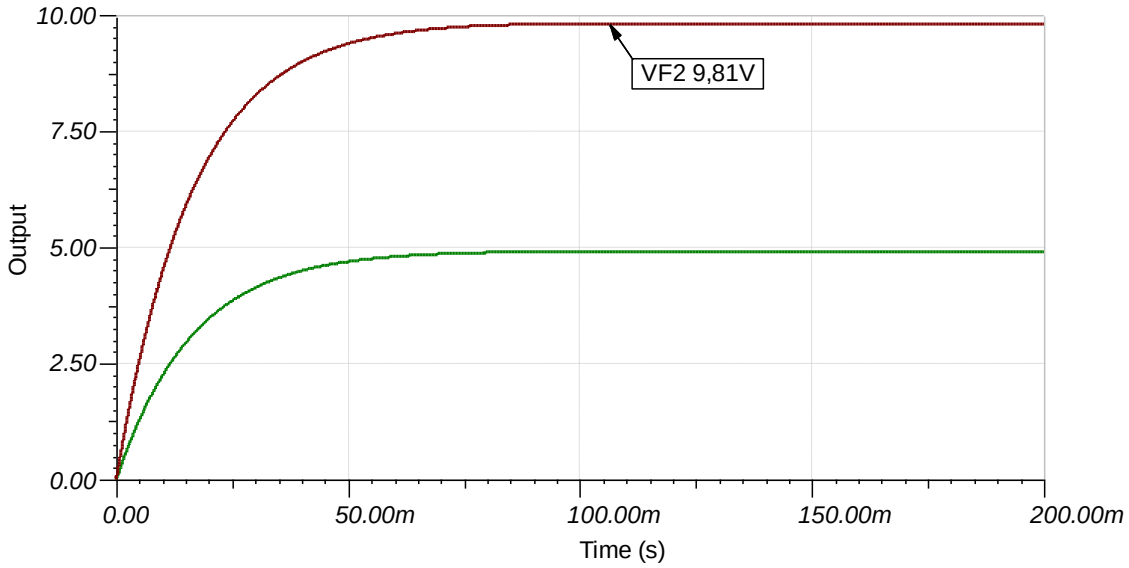


Figura 14. Simulació Circuit PWM al 99% cicle treball.

La simulació del circuit treballant al 50% del PWM seria aquesta:

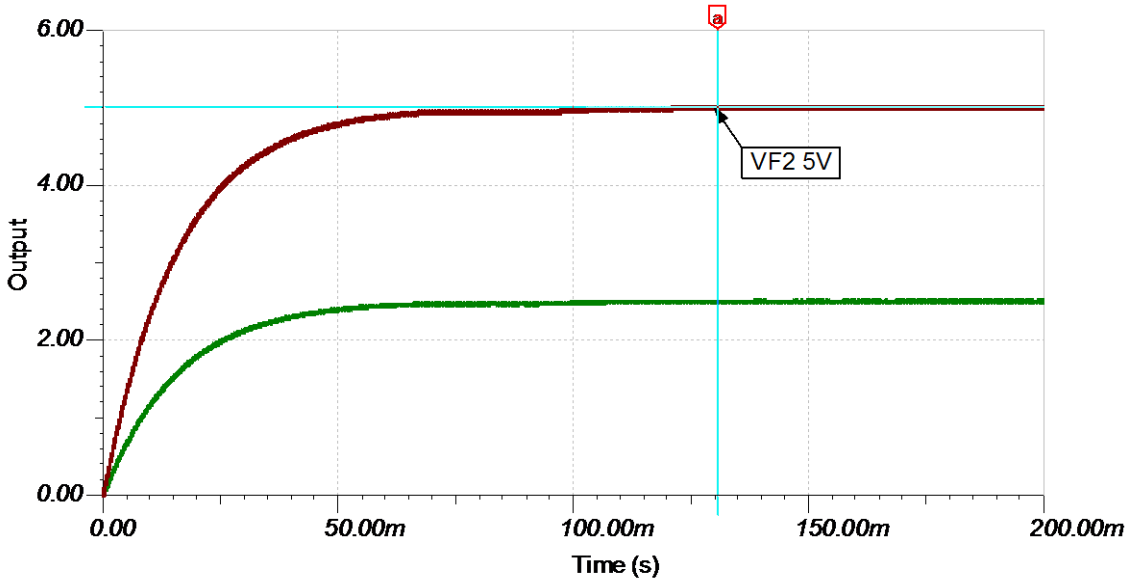


Figura 15. Simulació Circuit PWM al 50% cicle treball.

Si faig la simulació afegint una resistència de 50Ω a la sortida de l'amplificador, per un cicle de treball del 99%, veiem que la gràfica és pràcticament la mateixa, per tant podem dir que suporta una sortida de 200mA.

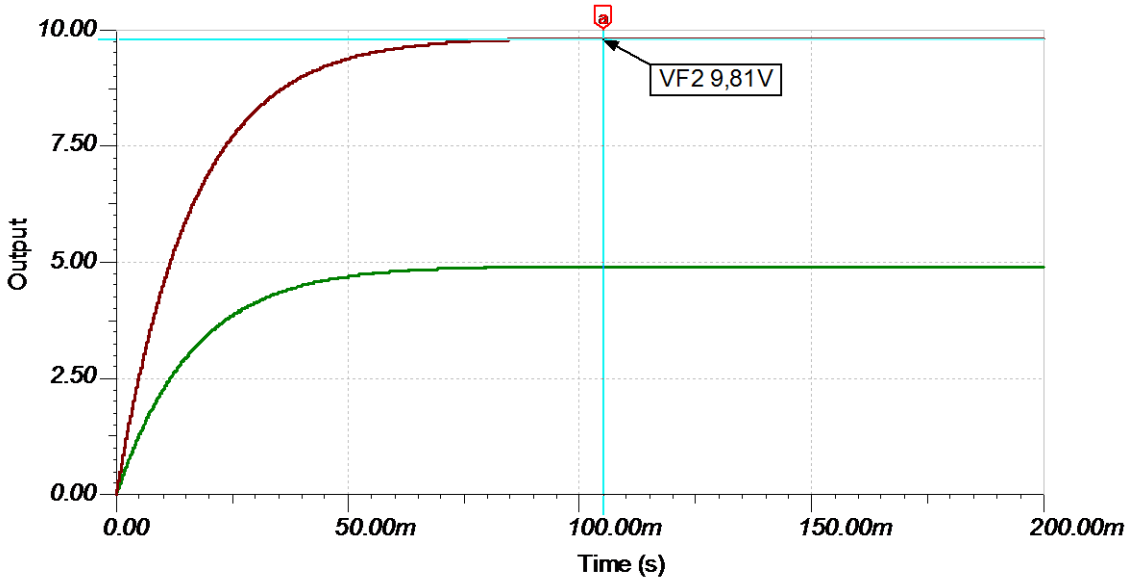


Figura 16. Simulació Circuit PWM al 99% cicle treball amb sortida 200mA.

Tenim exactament el mateix resultat.

Per fer la simulació he tingut en compte que per tenir un PWM de 10bits de precisió, utilitzarà 1024Tosc amb la qual cosa el període del senyal serà de 64us, per tant la freqüència PWM serà de 1/Període en aquest cas 15.625kHz.

3. Disseny de la Font d'Alimentació.

La font d'alimentació com a característica té que s'ha de poder alimentar amb qualsevol tensió continua que vagi de 12-48V això vol dir que haurem d'utilitzar un conversor de tensió continua a tensió continua, d'aquests n'hi ha de diferents tipus segons si volem més tensió a la sortida que no pas ens ofereix l'entrada, menys a la sortida que a l'entrada o volem una tensió dins el rang.

En el meu disseny he optat per un conversor *Buck* que és com s'anomenen els que obtenen una tensió reduïda en vers l'entrada, ja que segons les especificacions del microxip i els amplificadors operacionals que utilitzaré amb 10V a la sortida és suficient.

El model en concret que utilitzaré serà el TPS54060 de Texas Instruments, el qual pot treballar en el rang 3,5-60V.

Per fer el disseny m'he ajudat de l'eina que proporciona Texas Instruments anomenada Webench, amb la qual segons les especificacions indicades et mostra disseny simplificat, adjunt al *datasheet*, amb els valors d'entrada/sortida establerts, així com gràfiques de consum, i diferents alternatives al conversor de dc escollit.

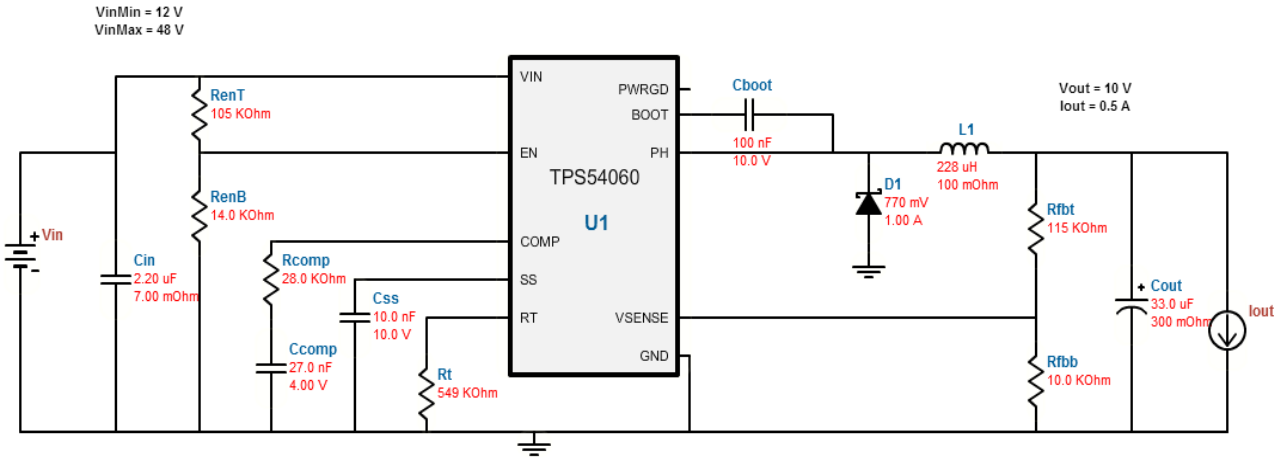


Figura 17. Disseny Font d'Alimentació.

La part més destacada del conversor és la del sistema que utilitza per graduar el voltatge de sortida a les nostres necessitats, la resta forma part del muntatge estàndard del conversor.

El voltatge de sortida es gradua amb el divisor de tensió de la sortida VSENSE, s'utilitza per defecte que una resistència sigui de 10k, a la imatge és Rfbb, i per calcular l'altre resistència s'utilitza la fórmula següent:

$$R1 = R2 \times \left(\frac{Vout - 0.8V}{0.8\,V} \right)$$

En el nostre cas com que volem 10V a la sortida, si resolem l'equació tenim:

$$R1 = 10000 * \left(\frac{10V - 0{,}8V}{0{,}8V} \right)$$
$$R1 = 10000 * 11{,}5$$
$$R1 = 115000\Omega$$

Que si mirem detalladament la sortida VSENSE, veiem el divisor de tensió amb els valors comentats, una resistència valor 10kΩ i l'altre 115kΩ.

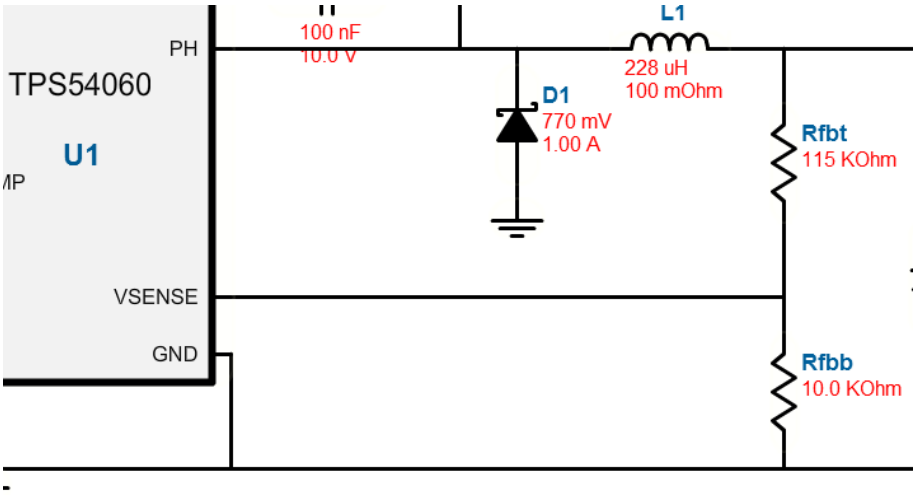


Figura 18. Detall divisor tensió del conversor DC.

Com he comentat tindrem 10V a la sortida que ens serviran per subministrar els amplificadors operacionals de l'expansor ja que aquests son el mateix model LM7341 *rail-to-rail* i ja ens subministraran 10v a la sortida analògica, ja que com he comentat abans els amplificadors amb aquesta característica poden proporcionar a la sortida una tensió igual a la seva entrada d'alimentació, pel microxip que necessita 5V el que farem serà utilitzar un altre conversor de corrent dc/dc en aquest cas de 10V a 5V.

El model triat per la conversió de 10V a 5V és el TPS62112, el qual porta de sèrie una configuració per sortida fixe de 5V, el disseny estàndard segons el *datasheet* per aquesta sortida de tensió és el següent:

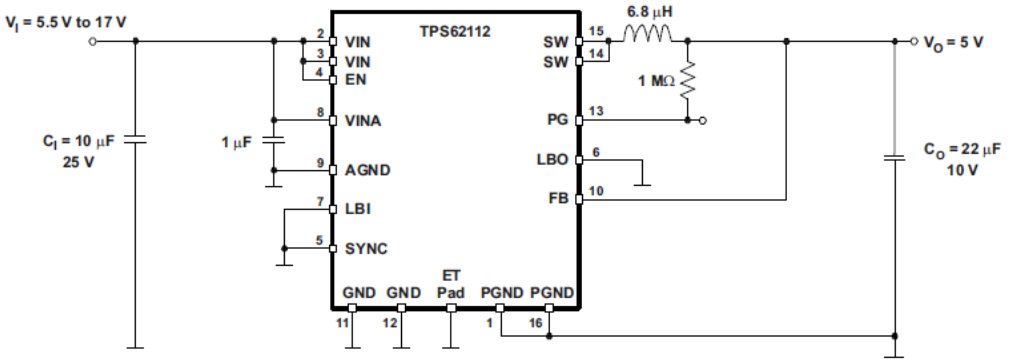


Figura 19. Disseny conversor de DC de la font al microxip.

Per acabar amb la font d'alimentació, donat que pel circuit amplificador operacional restador necessitem una alimentació de 1V, utilitzaré la sortida de 5V del conversor pel microxip i un divisor de tensió.

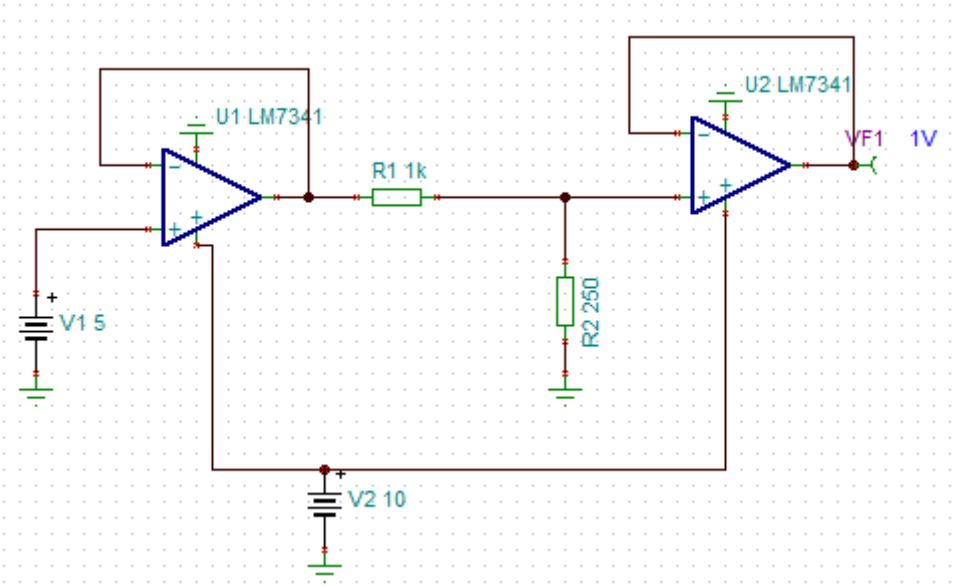


Figura 20. Disseny adaptador de 1V.

4. Implementació del processador.

En aquest bloc veurem tota la implementació del microxip, tant la conversió A/D com el funcionament amb el protocol Modbus, per donar les funcionalitats que necessitem al microxip primer haurem d'establir el mapa de memòria que utilitzarem.

4.1 Mapa de memòria Modbus.

Per definir el mapa de memòria hem de tenir en compte una sèrie de característiques de Modbus i en concret del que utilitzarem nosaltres l'RTU, les trames solen tenir el següent format.

- Numero d'esclau, en les trames que envia el mestre s'indica la direcció del destinatari de la trama, hi ha des de la 1 a la 247, com el Modbus RTU és hexadecimal serà des de l'adreça 0x00 fins la 0xF7, tot i que la 0x00 està reservada per fer *broadcasting*, o sigui un enviament a tots els esclaus.
- Codi de funció, aquí s'indica el tipus d'operació que es vol realitzar, lectura/escriptura o alguna ordre de control de l'esclau, aquest codi té un valor entre 0 i 127, moltes de les funcions son estàndard per tots els dispositius compatibles amb Modbus tot i que es poden afegir funcionalitats personalitzades.
- Direcció/dades, en aquest camp s'inclou la informació necessària per dur a terme el codi de funció especificat anteriorment.
- Codi *CRC*, les trames de Modbus RTU finalitzen amb un *Check Redundanci Code*, el qual s'utilitza per comprovar que la trama rebuda és correcte i no ha sofert canvis en cap dels seus bits.

El protocol Modbus segons el tipus de dades, aquestes es guarden en un rang de memòria en concret, generalment és així.

- 0-9999, s'utilitza per sortides digitals
- 10000-19999, s'utilitza per entrades digitals
- 20000-29999, sol estar reservat pel mateix protocol
- 30000-39999, s'utilitza per entrades analògiques.

- 40000-49999, s'utilitza per les sortides analògiques.

  En el nostre cas per identificar cada entrada sortida/utilitzarem.

Adreça	Dispositiu
0x0001	Entrada analògica 1
0x0002	Entrada analògica 2
0x0003	Entrada analògica 3
0x0004	Entrada analògica 4
0x0010	Sortida analògica 1
0x0011	Sortida analògica 2

Taula 2. Mapa d'entrades/sortides.

Com que el projecte tracta d'un expansor d'entrades/sortides, les funcions que utilitzarem seran les de lectura/escriptura, ja sigui en blocs de registres o registres en concret.

Funció	Descripció
0x03	*Read Holding Registers*. Rebrem l'adreça de memòria del primer registre a llegir i llavors el nombre de registres a llegir.
0x04	*Read Input Registers*. Rebrem l'adreça i el nombre de registres a llegir.
0x06	*Write Single Register*. Escriurem a l'adreça de memòria el valor que ens especificaran a la petició.

Taula 3. Mapa de Funcions Modbus.

Aquestes dues son les funcions mínimes que haurem d'utilitzar ja que les entrades i sortides dels esclaus de l'expansor són analògiques i suposem que el mestre és qui fa anar l'ordinador i el que "vigilarà" els dispositius que connectem a l'expansor.

Llavors també implementarem pel control d'errors una trama de resposta com la següent:

Adreça Dispositiu	Codi de funció + 0x80	Codi d'error	*CRC*

Taula 4. Trama d'error Modbus.

Els codis d'error que podem tenir seran els següents:

Codi	Tipus d'error
01	La funció rebuda no està suportada.
02	L'adreça de memòria no és vàlida.
03	El valor indicat al camp data no és vàlid.

Taula 5. Taula d'errors de l'expansor.

En cas que la trama contingui algun error en el *CRC* es retornarà al mestre.

4.2 Microxip.

Per l'elecció del microxip hem hagut de tenir en compte les necessitats que tenia el projecte sobre el nombre d'entrades/sortides i el fet que havia d'incorporar un conversor A/D.

Un dels microxips més utilitzats per fer conversions A/D és el PIC 16F877A el qual ja ens aniria bé pel nostre projecte, però es un microxip que fa temps que es comercialitza i la mateixa empresa que el fabrica a la seva web ens ofereix podríem dir, el seu substitut, el qual farem servir que es tracta del PIC16F887.

El PIC16F887 cobreix les nostres necessitats ja que disposa d'un conversor A/D de 10bits de precisió i té 14 canals, s'alimenta amb una tensió de 5V tal com havíem previst en els dissenys dels circuits d'adaptació i disposa de dues sortides PWM, les quals farem servir per les sortides analògiques.

El *pinout* del microxip és el següent:

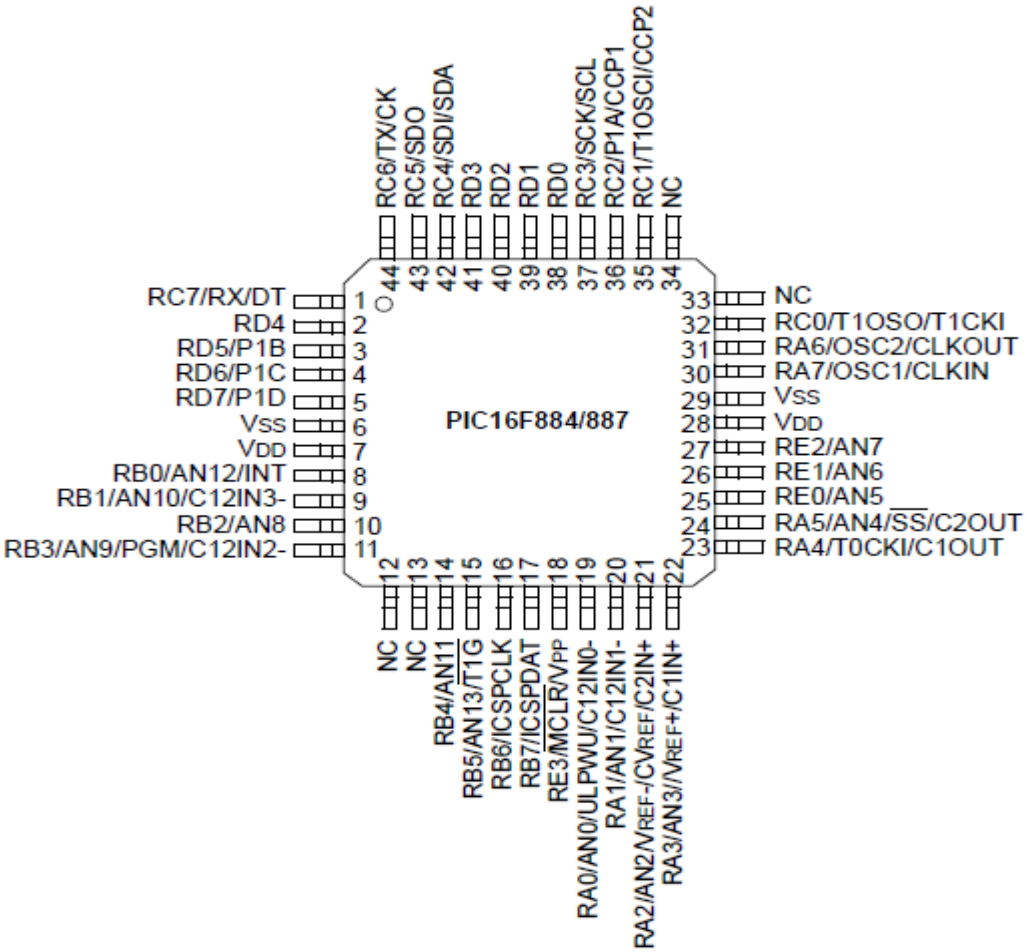


Figura 21. *Pinout* del microxip.

El seu diagrama de blocs.

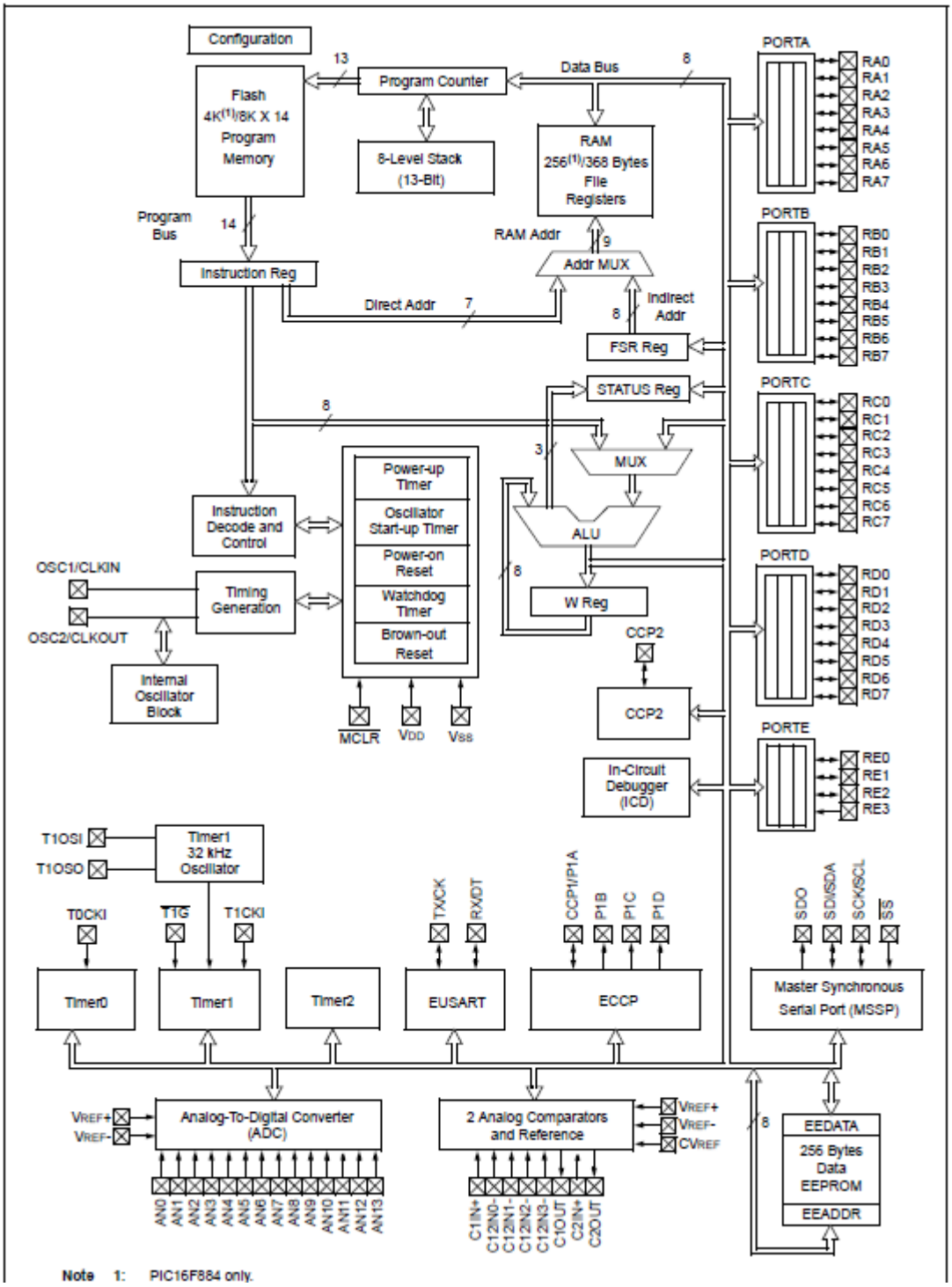


Figura 22. Diagrama de blocs del microxip.

4.3 Programació del microxip.

Per programar el microxip en principi volia fer ús del programari MPLAB però degut al microxip triat el compilador d'aquest (XC8) no disposava de llibreries per facilitar la programació, com les que inclou l'XC16 que s'utilitza en altres microxips, per aquest motiu al final he utilitzat CCS C Compilator (www.ccsinfo.com).

El programa consta de dos fitxers tfc.h que és el fitxer de capçalera típic de C i tfc.c que és on hi ha pròpiament la programació.

En el fitxer tfc.h hi tenim una part de la configuració del microxip la qual es genera fàcilment amb el compilador mitjançant un assistent.

```
#include <16F887.h>
#device adc=10

#FUSES NOWDT                    //No Watch Dog Timer
#FUSES HS                       //High speed Osc (> 4mhz for PCM/PCH)
(>10mhz for PCD)
#FUSES NOBROWNOUT               //No brownout reset
#FUSES NOLVP                    //No low voltage prgming, B3(PIC16) or
B5(PIC18) used for I/O

#use delay(clock=20000000)

#use rs232(baud=9600,parity=N, xmit=PIN_C6, rcv=PIN_C7, ENABLE=PIN_C5,
stream=PC, errors)
```

El més destacat és que definim que el conversor A/D utilitzi 10bits amb #device adc=10 i llavors hi tenim la configuració del rs232 per utilitzar la UART, aquí apareix com a rs232 però nosaltres hi tenim llavors el conversor MAX que serà el que transforma el senyal en rs485, en aquesta configuració li indiquem els PINS del microxip que utilitzarem per transferir, rebre i el pin que fem servir per indicar si enviem o rebem, així com la velocitat.

A la part pròpiament del codi la tenim a tfc.c, he creat diferents funcions per fer el codi més clar.

```
int16 llegeixAD(int direccio)
{
   switch(direccio)
   {
   case 0x01:
      set_adc_channel(sAN0);
      break;
   case 0x02:
      set_adc_channel(sAN1);
      break;
   case 0x03:
      set_adc_channel(sAN2);
      break;
   case 0x04:
      set_adc_channel(sAN3);
      break;
   }
   delay_us(10);
   int16 value = read_adc();

   return value;
}
```

En aquesta funció el que faig és la conversió A/D de l'entrada analògica que passo com a paràmetre i retorno el valor en una variable int16, ja que la conversió és de 10bits.

Després hi ha una funció semblant però en aquest cas per generar el pols a les sortides per mitjà del PWM.

```
void enviaPWM(int8 port,int16 valor)
{
   switch(port)
   {
      case 0x10:
         set_pwm2_duty(valor);
         break;
      case 0x11:
         set_pwm1_duty(valor);
         break;
   }

}
```

A la funció li passo dos paràmetres, el port de sortida i el valor que ha de tenir, recordem que el valor estarà entre 0-1023 ja que ha de tenir 10bits de resolució, aquesta funció com veiem no retorna res.

També he definit una estructura per guardar-hi la trama ordenada mitjançant un struct i pel *crc* el que he fet és una variable union amb dos int8, per tenir els 2 bits disponibles per manipular.

```
typedef struct {
   int8 direccio;
   int8 funcio;
   int8 data[4];
   } trama;

union
{
   int8 b[2];
   int16 d;
} crc;
```

Per calcular el *CRC* he fet servir una funció disponible, juntament amb les variables constants que utilitza, la API modbus que incorpora el compilador.

També he creat una funció per crear la trama en cas d'algun dels errors especificats al mapa de memòria, és aquesta:

```
void errorTrama(trama rebre, int error)
{
            char c;
            crc.d = 0xFFFF;
            c = rebre.direccio;
            modbus_calc_crc(c);
            fputc(c);
            c = rebre.funcio+0x80;
            modbus_calc_crc(c);
            fputc(c);
            c = error; // error de funció no vàlida
            modbus_calc_crc(c);
            fputc(c);
            fputc(crc.b[0]);
            fputc(crc.b[1]);

}
```

En aquest cas es crea la trama definida al mapa de memòria on primer estableixo el valor de la variable *CRC* a 0xFFFF per poder calcular el valor que tindrà la trama, tot seguit es van agafant els valors de la trama, es calcula el *CRC* i s'envien per la UART, un cop estan tots s'envien els valors del *CRC* i es dóna per finalitzada la trama.

Abans de començar el bucle del programa hi ha definides una sèrie de variables les quals també les codifica l'assistent que incorpora el CCS Compilator.

```
setup_adc_ports(sAN0|sAN1|sAN2|sAN3);
setup_adc(ADC_CLOCK_DIV_2);
setup_timer_0(RTCC_INTERNAL|RTCC_DIV_32|RTCC_8_bit
setup_timer_2(T2_DIV_BY_16,249,5);

setup_ccp1(CCP_PWM|CCP_SHUTDOWN_AC_L|CCP_SHUTDOWN_BD_L);
setup_ccp2(CCP_PWM);
set_pwm1_duty((int16)0);
set_pwm2_duty((int16)0);
```

Primer es configuren els ports per la conversió A/D, en el meu cas com que només utilitzo les 4 primeres entrades, aquestes són les que defineixo, tot seguit hi trobem el rellotge que utilitzarà el conversor A/D, llavors ha configurat el timer0 i el timer2, el 2 és el que s'utilitza per generar el pols PWM i els pins cpp1 i cpp2 que faran funció de PWM, per últim indica que la sortida PWM sigui 0 per defecte.

Tot seguit hi tenim la part del programa amb un bucle infinit while(*TRUE*) i a dins la part codificada que comença quan kbhit() és *TRUE* que voldrà dir que tenim un caràcter a la UART.

```
while(TRUE)
{

/* si ha arribat caràcter comencem */
if(kbhit())
{
crc.d = 0xFFFF;
trama rebre;

char c;
c = getc();
modbus_calc_crc(c);
rebre.direccio = c;

c = getc();
modbus_calc_crc(c);
rebre.funcio = c;

c = getc();
modbus_calc_crc(c);
rebre.data[0] = c;

c = getc();
modbus_calc_crc(c);
rebre.data[1] = c;

c = getc();
modbus_calc_crc(c);
rebre.data[2] = c;

c = getc();
modbus_calc_crc(c);
rebre.data[3] = c;
```

```
int8 crcHi = getc();
int8 crcLo = getc();
```

Al trobar un caràcter primer poso a "0" el *CRC* per poder calcular que la trama sigui correcte, es crea una variable *struct* trama i llavors es va guardant cada caràcter que arriba a la trama, tot seguit guardem el *CRC* de la trama que hem rebut i el comparem amb el *CRC* que es genera apartir del que hem rebut.

```
if(rebre.direccio==0x01 || rebre.direccio == 0x00)
```

Primer trobem un condicional on comprovem que la trama sigui pel nostre dispositiu o sigui *broadcast*, si és així la processem i sino la ignorem.

```
if( crc.b[0] == crcHi && crc.b[1] == crcLo)
          {
            int16 adress = make16(rebre.data[0],rebre.data[1]);
            /* Funció   Descripció
             0x03/04   Lectura del bloc de registres analògics.
             0x06   Permet l'escriptura als registres de sortida
analògica. */
             switch(rebre.funcio)
             {
             case 0x03:
             case 0x04:
                int16 lectures = make16(rebre.data[2], rebre.data[3]);
                if(adress>0x04 || (adress+lectures) > 0x04)
                {
                   errorTrama(rebre,0x02);
                }
                else
                {
                    //faig un "reset" al crc per calcular el nou CRC
                   crc.d = 0xFFFF;
                   c = rebre.direccio;
                   modbus_calc_crc(c);
                   fputc(c);
                   c = rebre.funcio;
                   modbus_calc_crc(c);
                   fputc(c);
                   c = lectures * 2;
                   modbus_calc_crc(c);
                   fputc(c);
                   for (int i = 0; i < lectures; i++)
                   {
                      int16 conv = llegeixAD(adress+i);
                      c = conv>>8; //enviem els 2 primers bytes de la
conversió
                      modbus_calc_crc(c);
                      fputc(c);
                      c = conv; //els 2 bytes restants de la conversió
                      modbus_calc_crc(c);
                      fputc(c);
                   }
                   //Finalment enviem el CRC
                   fputc(crc.b[0]);
```

```
            fputc(crc.b[1]);
        }
        break;
    break;
```

Si el *CRC* és correcte llavors entrem al switch on segons el valor de la funció que ens arriba farem una o altre, si ens arriba la funció 0x03 o 0x04 que tenen el mateix funcionament, primer es comprova que l'adreça o adreces de memòria que es vol llegir existeixin, si es així es comença a crear la trama de resposta i hi tenim un bucle for, on es faran tantes lectures i conversions A/D com ens hagin indicat i per últim enviarem els bits de *CRC*.

Si la funció que demanen és la 0x06 es comprova que el valor a escriure estigui entre 0 i 1023(0 i 0x03FF) i que l'adreça de sortida sigui correcte, sinó es fa la corresponent trama d'error.

```
case 0x06:
            int16 valor = make16(rebre.data[2], rebre.data[3]);
            if(valor>0x3FF)
            {
               errorTrama(rebre, 0x03);
            }
            else if(adress<0x10 || adress>0x11)
            {
               errorTrama(rebre, 0x02);
            }
            else
            {
```

Si tot és correcte llavors es genera el pols PWM amb el valor rebut i es genera una trama de resposta.

```
               enviaPWM(adress, valor);
               crc.d = 0xFFFF; //faig un "reset" al crc per
               calcular el nou
               c = rebre.direccio;
               modbus_calc_crc(c);
               fputc(c);
               c = rebre.funcio;
               modbus_calc_crc(c);
               fputc(c);
               c = rebre.data[0];
               modbus_calc_crc(c);
               fputc(c);
               c = rebre.data[1];
               modbus_calc_crc(c);
               fputc(c);
               c = rebre.data[2];
               modbus_calc_crc(c);
               fputc(c);
               c = rebre.data[3];
               modbus_calc_crc(c);
```

```
                fputc(c);
                fputc(crc.b[0]);
                fputc(crc.b[1]);
            }
            break;
```

En cas que la funció que arriba no sigui cap de les suportades pel dispositiu o el *CRC* de la trama que hem rebut no sigui correcte, es generen trames d'error.

```
            default:
                //Si no ens envien una funció definida enviem missatge
d'error
                errorTrama(rebre,0x01);
            }

        }
        else
        {
            putc(rebre.direccio);
            putc(rebre.funcio+0x80);
            putc(rebre.data[0]);
            putc(rebre.data[1]);
            putc(rebre.data[2]);
            putc(rebre.data[3]);
            putc(crcHi);
            putc(crcLo);
        }
```

Per calcular el *CRC* he fet servir una funció disponible, juntament amb les variables constants que utilitza, de la API modbus que incorpora el compilador, és la següent:

```
   void modbus_calc_crc(char data)
   {
   int8 uIndex ; // És l'índex per adreçar-nos a les taules de CRC
(lookup table)
   //Càlcul del CRC
   uIndex = (crc.b[1]) ^ data;
   crc.b[1] = (crc.b[0]) ^ modbus_CRCHi[uIndex]; //byte   major pes
   crc.b[0] = modbus_CRCLo[uIndex]; //byte   menor pes
   }
```

4.4 Simulació del microxip.

Per acabar el tema de la programació del PIC he fet unes simulacions amb el programa Real Pic Simulator, en la primera demano una lectura de l'adreça de memòria 0x0001 que seria AN0 (Al simulador Analog1), i que volem llegir 2 registres, per tant farem la lectura de Analog1 i Analog2 que tenen com a valor 1023.

01 03 00 01 00 02 CB 95

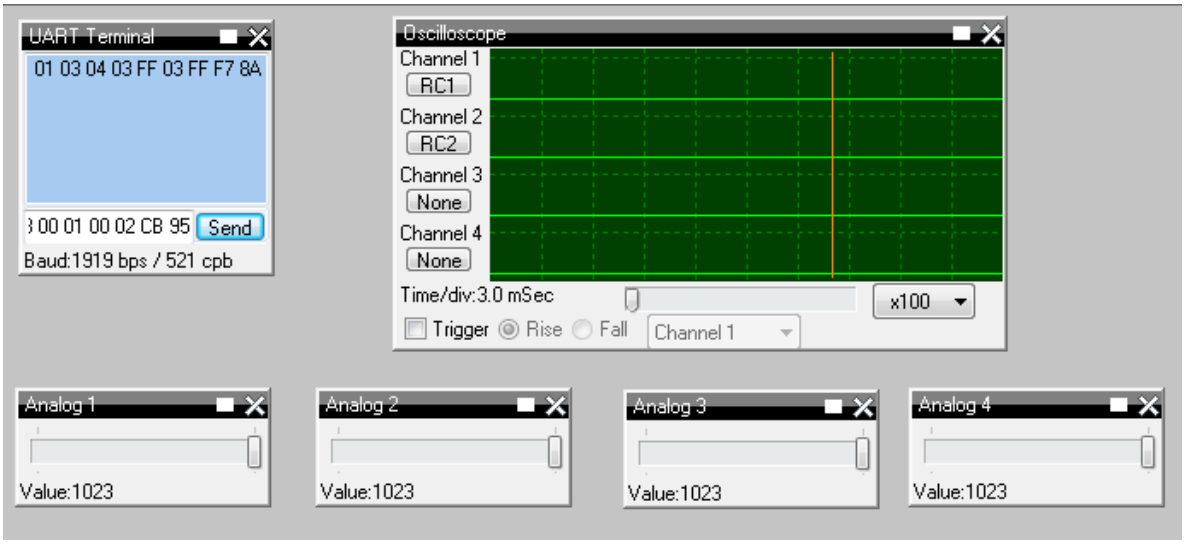


Figura 23. Simulació de la funció 0x03.

Podem veure com ens respon amb l'adreça del dispositiu 0x01, la funció 0x03, llavors 0x04 indica que rebrem 2 registres de 2 bytes, tot seguit hi tenim 0x03FF i 0x03FF que corresponen als valor de la lectura, seguit del *CRC* F7 8A.

Si fem una petició demanant la lectura de l'adreça 0x05 amb la trama 01 03 00 05 00 01 0B 94 com que no esta disponible rebrem una trama d'error.

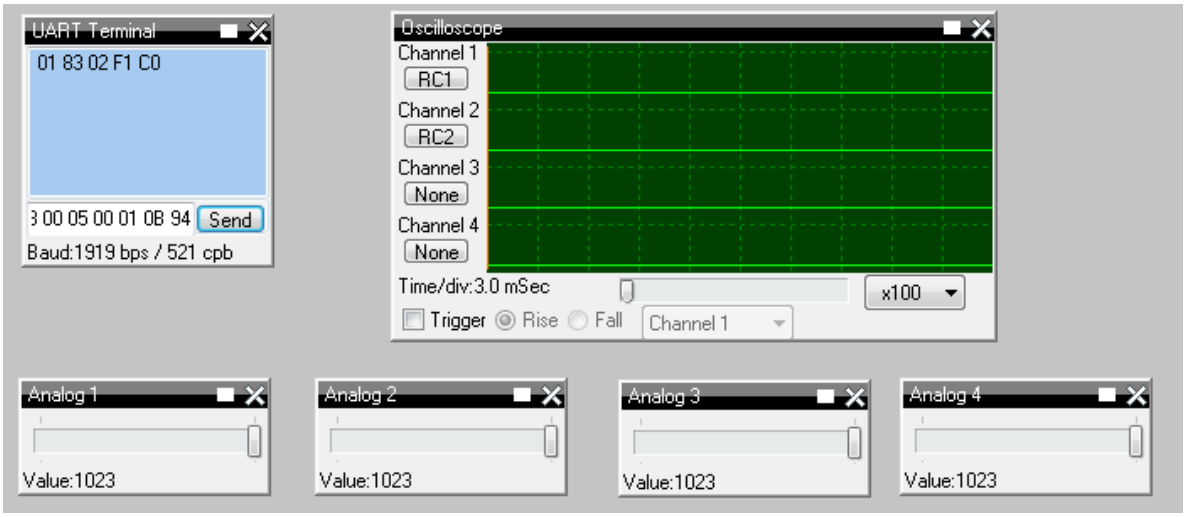


Figura 24. Simulació de la funció 0x03 amb error.

Veiem que la resposta hi tenim l'adreça del nostre dispositiu 0x01, llavors la funció que hem demanat + 0x80 per tant 0x83, tot seguit el codi d'error 0x02 que segons el mapa de memòria indica adreça de memòria no vàlida i finalment el codi *CRC*.

Ara simularem una escriptura per activar el PWM i generar la sortida analògica enviant la trama 01 06 00 10 02 00 6F 89, amb la qual indiquem que al registre 0x0010 volem donar-li valor 0x0200.

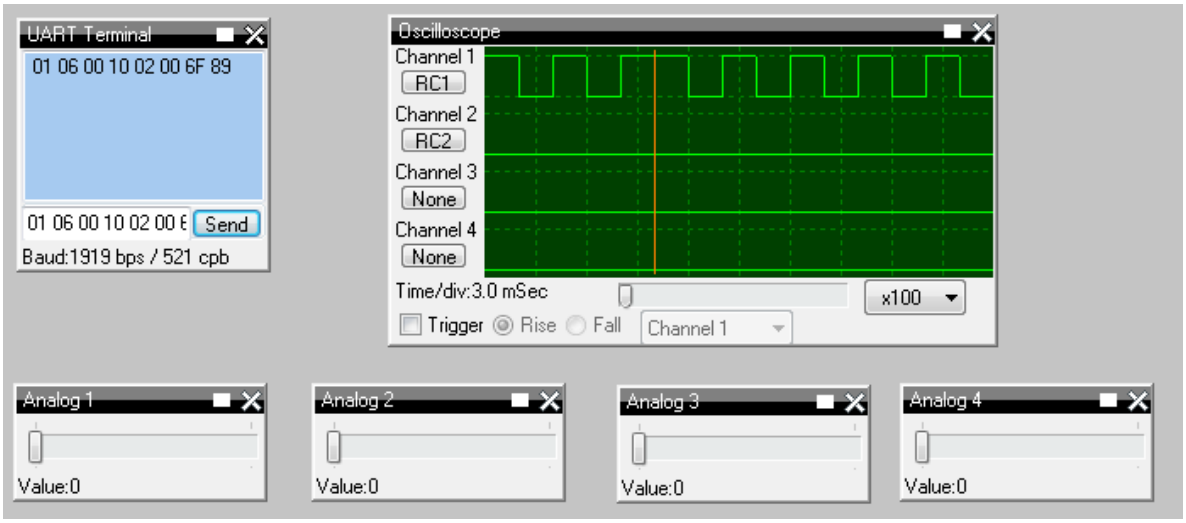


Figura 25. Simulació de la funció 0x06 amb valor 0x200.

En aquest cas al enviar el valor 0x200 (512) es veu com el pols que generem és del 50%. Si enviem el valor 0x3AA (938) que es comença a apropar al 0x3FF que és el valor màxim hauriem de veure que el pols ha augmentat.

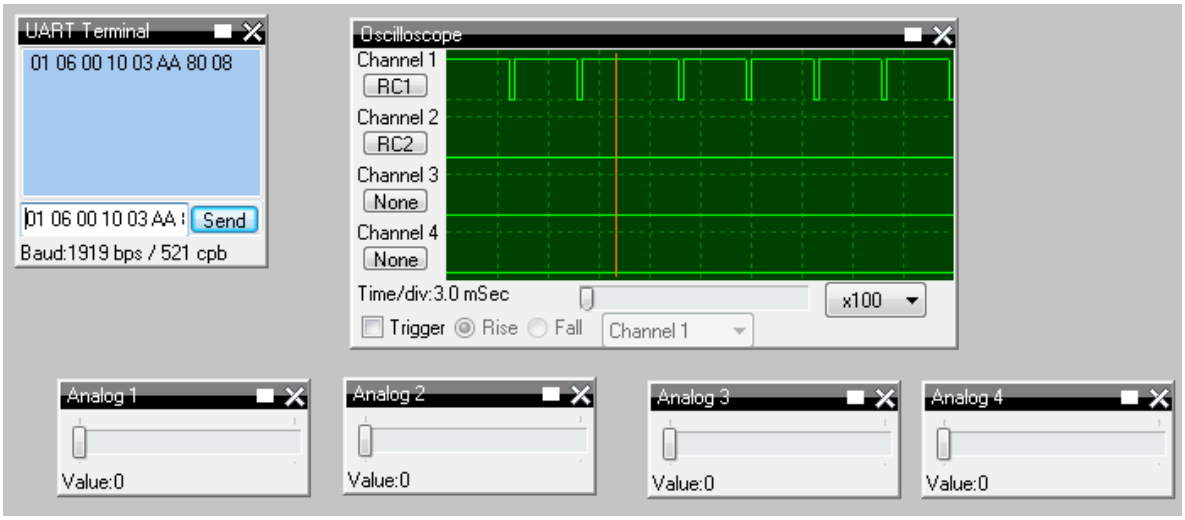
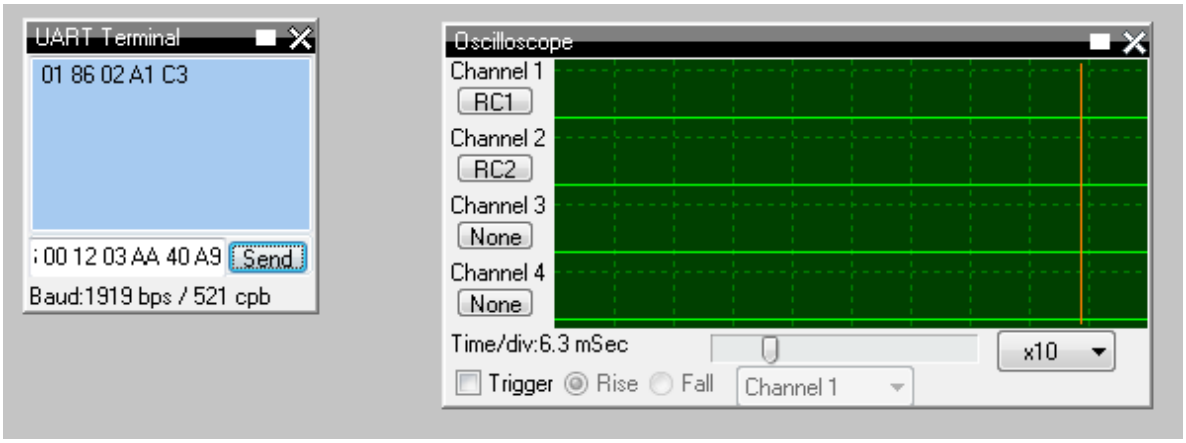


Figura 26. Simulació de la funció 0x06 amb valor 0x3AA.

Mirant la imatge efectivament podem comprovar que el pols ha augmentat el seu període.

Si enviem una trama on volem escriure a alguna adreça de memòria que no sigui la 0x0010 o la 0x0011 llavors hem de rebre una trama d'error.



Hem enviat la trama 01 06 00 12 03 AA 40 A9, amb la qual volem executar la funció 0x06 d'escriure i com adreça de memòria el registre 0x0012, que està fora de rang i per tant hem rebut la trama d'error 01 86 (0x06+0x80) 02 (error fora rang adreces) i els 2 bytes de *CRC*.

5. Implementació de la interfície de sortida del microxip a RS-485.

El que veurem en aquesta part és la última peça que ens falta al circuit per tal de completar la placa PCB.

Per poder comunicar la placa amb un terminal o PC i poder fer les crides modbus al nostre projecte hem d'afegir la interfície amb la qual ens puguem comunicar amb el microxip, aquesta la farem apartir de la sortida USART del processador.

Com hem comentat a l'apartat d'introducció del RS-485, aquest fa servir dos cables amb els quals per un viatja un senyal i per l'altre hi viatja la mateixa senyal però invertida així els possibles senyals de soroll que es trobi quedaran anul·lats, per aconseguir això el que utilitzarem serà un dispositiu de la casa Maxim Integrated, el MAX481 RS-485 *transceiver* que ens convertirà les dades de la sortida USART del microxip en dos cables per implementar la interfície RS-485.

Aquest dispositiu s'alimenta amb una tensió de 5V com el microxip, per tant el subministrarem amb el mateix circuit del processador i la resta de connexions seran les RX i TX del microxip al MAX481.

Els MAX481 pot arribar a una velocitat de 2.5Mbps, més que suficient per les nostres necessitats i ens ofereix una connexió *Half Duplex* la qual cosa també ens va bé ja que en el nostre projecte considerem que el terminal/PC connectat amb l'RS-485 serà el que faci de mestre i demani la informació de les trames modbus per tant al ser el mateix dispositiu qui demani/enviï informació no considero necessari el *Full Duplex*.

El circuit implementat seria el següent:

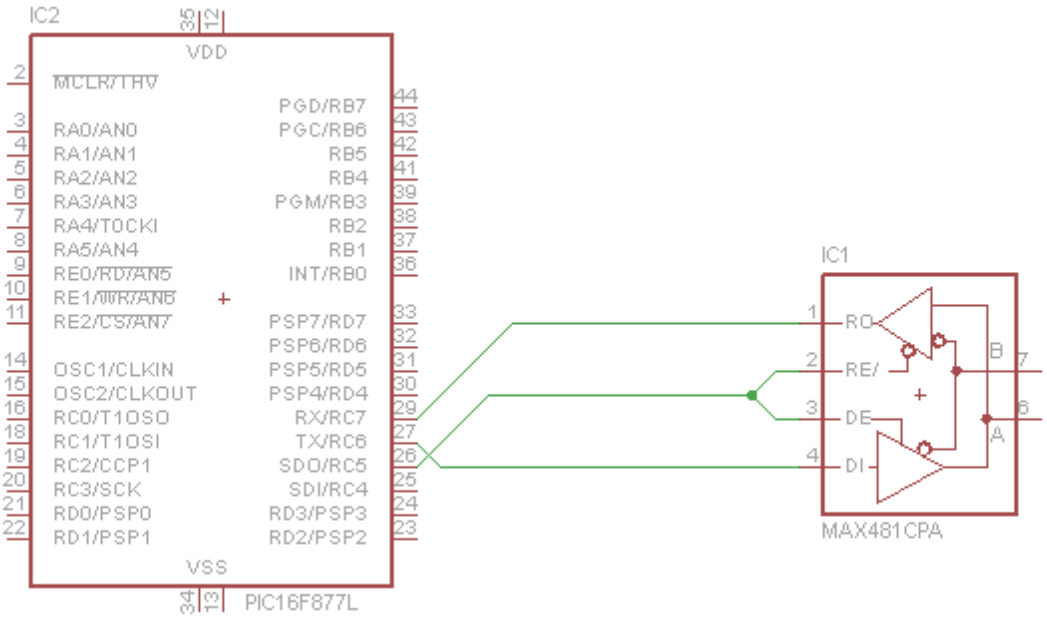


Figura 27. Circuit interfície del microxip a RS-485.

Pel disseny he hagut d'agafar el PIC16F877 ja que el que faig servir, el PIC16F887 no hi tenia el disseny, però pràcticament és el mateix, es connecten les sortides del microxip TX i RX a les corresponents del *transceiver* i apart utilitzem el PIN_C5 del microxip per regular l'enviament/recepció.

6. Disseny layout de la placa PCB.

En aquest apartat veurem el disseny de la placa on aniran col·locats tots els dispositius amb els seus circuits pertinents que hem anat veient anteriorment.

Per construir la nostra placa PCB utilitzaré el programari Eagle 6.3 Light de Cadsoft, donat que aquesta és la versió gratuïta i no té totes les prestacions de la normal, només permet el disseny de plaques petites és per aquest motiu i perquè els circuits es vegin més clars que he optat per dividir la placa en 2 trossos, en un apareixeran el disseny de la font d'alimentació amb els dos conversos de DC i d'altra banda tindrem tot el que serien els circuits d'adaptació amb el microxip.

Les plaques segons si es fan casolanes o enviem els fitxers a una empresa perquè ens la imprimeixi tindran unes limitacions o unes altres, com el gruix de la pista, que és bo que la part d'alimentació sigui més aviat gruixuda i que farà que com més petites puguin ser les pistes més en puguem posar o més les podrem acostar a dispositius sense que hi facin interferència.

Una altra limitació serà el nombre de capes que tingui la placa, si ens la fem nosaltres podríem fer que tingui 2 pistes com a molt, una a la part superior i una altra a la part inferior, si ens fa la placa una empresa podríem arribar a tenir més capes i comunicar-les entre elles mitjançant vies, que aquestes el que fan és comunicar una capa amb una altra.

Tot i això una de les consideracions més importants que hem de tenir en compte alhora de dissenyar una placa PCB és mirar que les pistes no facin angles quadrats de 90º, per poc que puguem els hauríem de fer de 45º.

6.1 Placa PCB font d'alimentació.

Com he comentat he dividit la placa en dos trossos per veure més clar el disseny i que no quedi tant atapeït, primerament veurem el disseny de la PCB de la font d'alimentació, per fer les plaques PCB primerament el programari ens fa dibuixar el disseny com si treballéssim al TINA-TI, un cop tenim aquest disseny ja podem fer el de la placa.

Els disseny ens quedaria així:

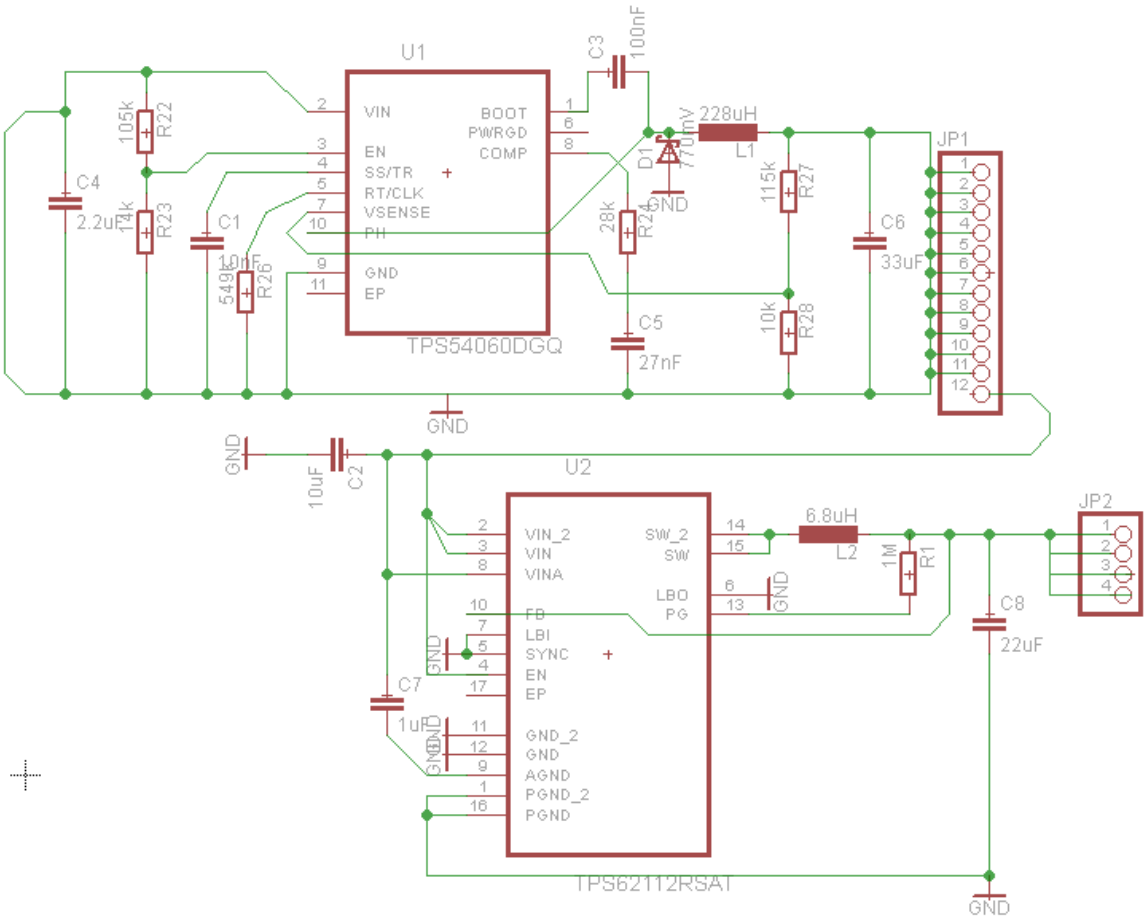


Figura 28. Disseny font d'alimentació per la placa PCB.

Per aquest disseny el fet que hi hagi angles de 90º a les pistes no té massa importància, ja que és alhora de fabricara la placa PCB on hi podríen haver problemes.

Podem veure a la imatge que he col.locat un pin de 8 posicions el qual co.locaré a l'altre disseny com si continuès i fos la mateixa placa.

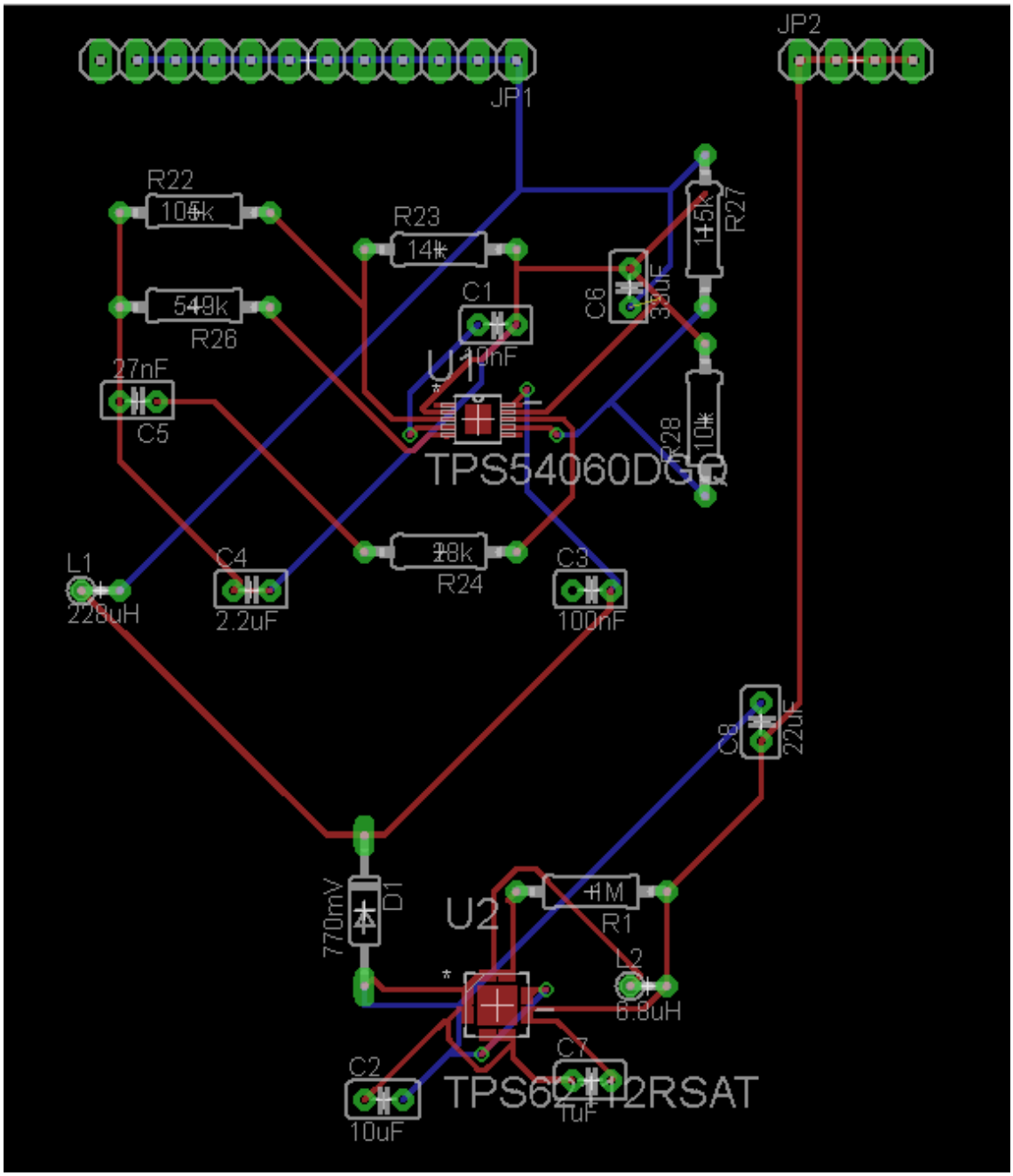


Figura 29. Placa PCB de la font d'alimentació.

Donada la meva poca experiència amb plaques PCB, he utilitzat l'optimitzador de l'eagle per fer les pistes i després he retocat les pistes que m'ha semblat estaven mal posades o feien angles extranys i també he intentat utilitzar la capa superior de la placa tant com fos possible però donada la quantitat de dispositius connectats entre ells he utilitzat la capa inferior també així com alguna vía.

6.2 Placa PCB circuit d'adaptació.

La resta de la placa amb els circuits d'adaptació, el microxip i l'adaptador RS-485 seria aquesta:

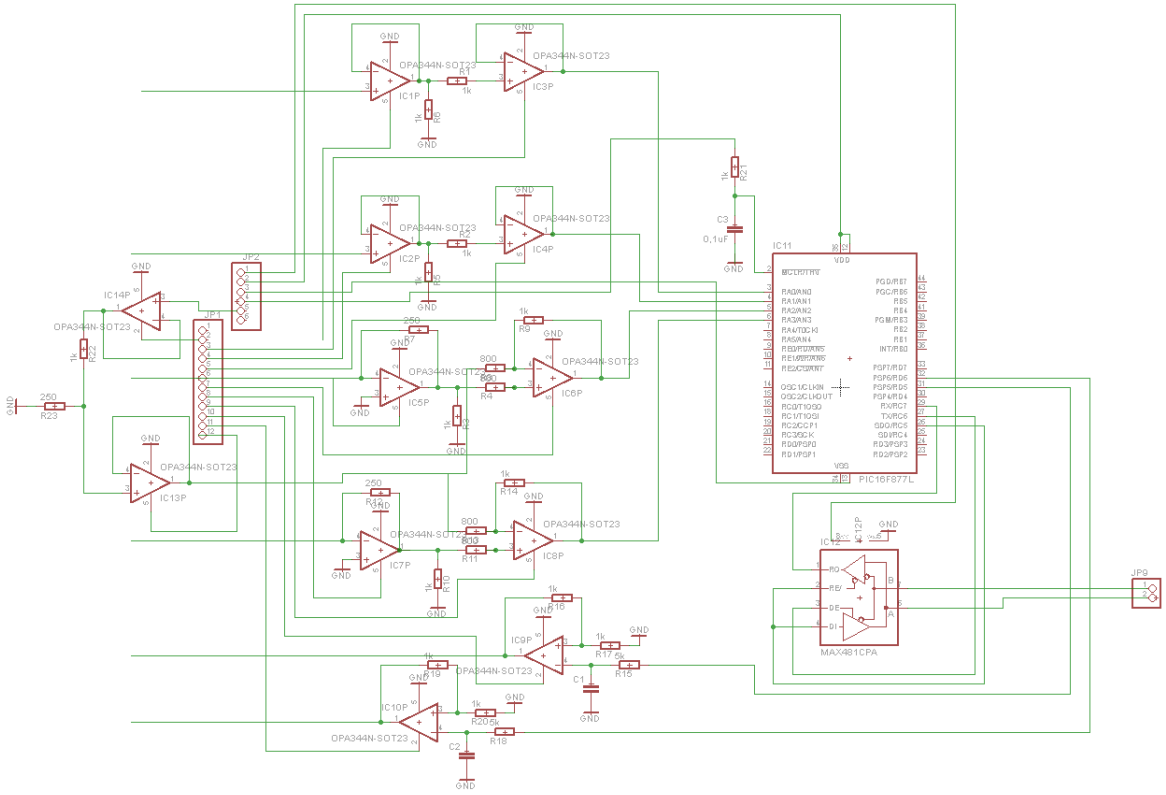


Figura 30. Disseny circuit d'adaptació per la placa PCB.

Per la placa també he utilitzat l'optimitzador i he refet les pistes extranyes que m'havia creat.

Com veurem ens queda un circuit bastant més complexe que l'anterior degut a la gran quantitat de dispositius i connexions que hi apareixen.

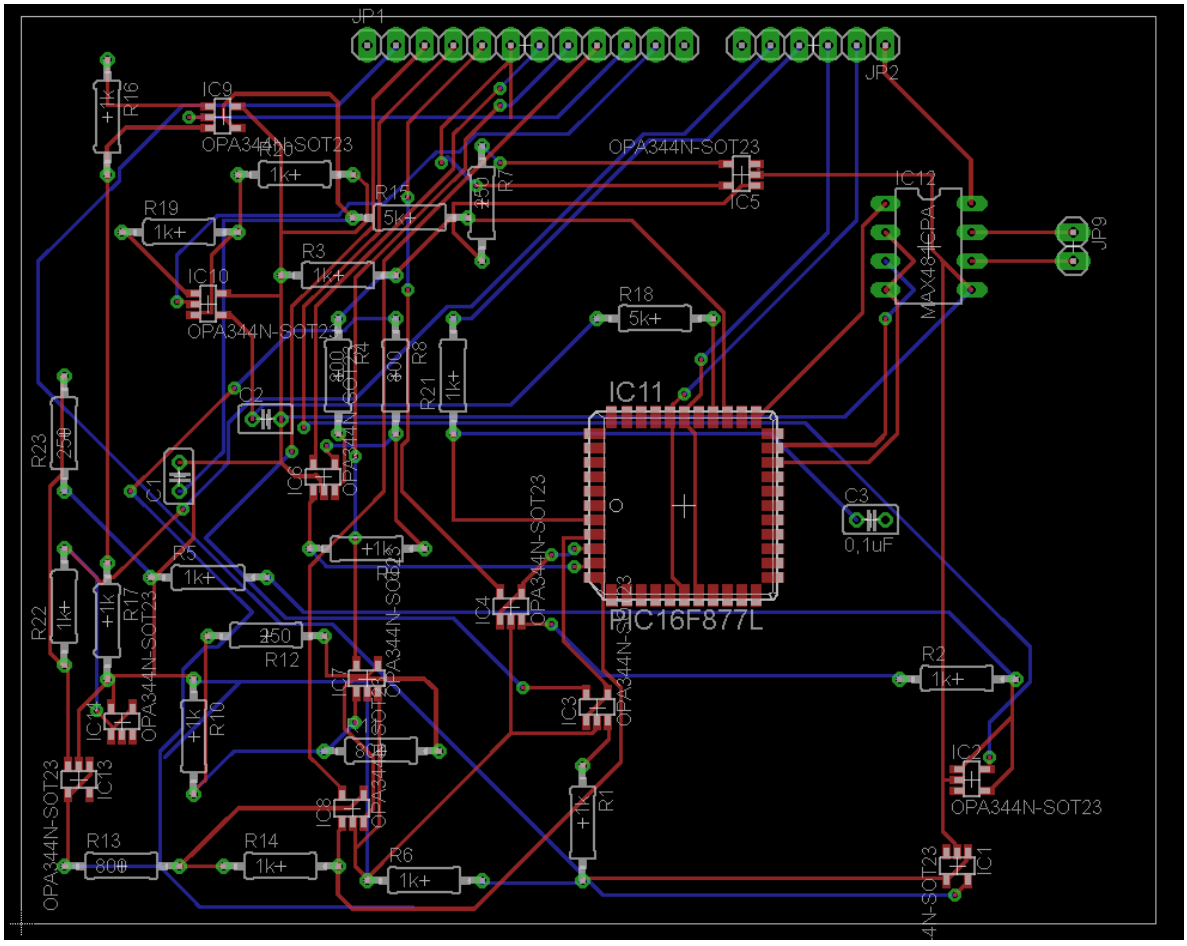


Figura 31. Placa PCB del circuit d'adaptació.

7. Conclusions i ampliacions.

Finalment un cop unides totes les peces que conformen el projecte ja el podem donar per acabat.

En projectes com aquest és on queda clar la constància que s'ha de tenir per poder dur-lo a terme ja que s'ha d'anar treballant cada part fins a poder ajuntar-les i és un clar exponent de la frase "divideix i guanyaràs", ja que recordo la primer lectura l'enunciat del projecte i que no sabia ni que havia de fer, però poc a poc, pas a pas ha anat agafant forma.

Per mi ha estat una experiència molt enriquidora, sobre electrònica no en sabia res pràcticament abans de començar el projecte, ha sigut com si descobrís una tecnologia nova i l'hagués de fer servir per implementar un nou projecte, he hagut de buscar molta informació sobre cada part del projecte per saber com funcionava i poder-la implementar.

Tot i que segurament la meva solució tindrà mancances, coses que es podrien fer millor, estic molt orgullós de mi mateix i no hi ha millor premi que aquest.

Les possibles ampliacions del projecte son diverses, primer és podria augmentar el nombre de connexions d'entrada, amb el mateix microxip ja es podria implementar.

També es podria augmentar el nombre de sortides de l'expansor però per fer això ja hi hauria més feina.

Llavors també es podria canviar el funcionament de l'expansor i fer-lo que donat un interval de temps recopilés informació de les entrades i les fes arribar al terminal connectat al RS-485, d'aquesta manera podríem monitoritzar tota una sèrie de màquines o una cadena de muntatge per exemple.

Bibliografia.

- http://www.ti.com/
- http://www.modbus.org/
- http://www.simplymodbus.ca/
- http://www.uhu.es/adoracion.hermoso/Documentos/tema-5.pdf
- http://www.electronicafacil.net/tutoriales/
- http://provideyourown.com/2011/analogwrite-convert-pwm-to-voltage/
- http://www.tutoelectro.com
- http://www.analog.com
- http://www.mikroe.com/chapters/view/1/introduction-world-of-microcontrollers/
- http://www.ccsinfo.com/downloads/ccs_c_manual.pdf
- http://www.maximintegrated.com