Escola Politècnica Superior de Castelldefels
UNIVERSITAT POLITÈCNICA DE CATALUNYA

TREBALL DE FI DE CARRERA

TÍTOL DEL TFC: Disseny d'un espectroradiòmetre

TITULACIÓ: Enginyeria Tècnica de Telecomunicació, especialitat Sistemes de Telecomunicació

AUTORS: Maribel Pérez González
Núria Pujol Vilanova

DIRECTOR: Jaume Piera Fernández

DATA: 5 de setembre de 2005

Títol: Disseny d'un espectroradiòmetre

Autor: Maribel Pérez González
Núria Pujol Vilanova

Director: Jaume Piera Fernández

Data: 5 de setembre de 2005

Resum

L'espectrometria arriba al moment de màxima expansió amb la miniaturització dels espectròmetres i s'utilitza actualment en infinitat de camps, sobretot a l'hora de caracteritzar components sense necessitat de realitzar un anàlisis químic. En els casos més recents s'utilitza aquest tipus de tecnologia per caracteritzar medis inhòspits a distància, com el sòl del planeta Mart. El nostre projecte segueix aquesta filosofia, ja que consisteix en la integració d'un espectròmetre dins d'una sonda oceanogràfica que permeti rebre informació del fons marí des de la superfície.

En el projecte s'han seguit els passos necessaris per tal de realitzar el disseny complert del sistema tractant tant software com hardware. Els passos principals en els que s'ha dividit la feina realitzada són la tria de components i l'estudi detallats dels datasheets, la realització del circuit i disseny de la placa utilitzant software específic, la programació del microcontrolador i la creació de la interfície gràfica.

S'ha finalitzat amb la plasmació del disseny en una placa de circuit integrat governada per una interfície gràfica en llenguatge Matlab obtenint així un primer prototip obert a futures modificacions.

Títle: Disseny d'un espectroradiòmetre

Author: Maribel Pérez González
Núria Pujol Vilanova

Director: Jaume Piera Fernández

Date: Setember, 5th 2005

Overview

The spectrometry suffered the highest expansion when the miniaturized spectrometers appeared and are currently used in lots of different disciplines, moreover in chemical composition characterisation without using chemical laboratory analyse. This kind of technology is recently used to characterise inhospitable environments from the distance, like Mars soil floor. This project follows this influence because it consists in integrate a spectrometer into an oceanographic probe allowing information reception from sea bed to the surface.

The project follows the necessary steps to complete the full design with software and hardware sections. The main steps the project is divided in are the electronic components choosing and data sheet studying, design the electronic circuit and the electric board using specific software, the microcontroller programming and the graphic interface construction.

Finally, the design has been assembled into a prototype board controlled by a Matlab graphic user interface, so future modifications and improvements can be applied easily.

Agraïm la paciència de tot l'equip de la UTM, en especial al Marc i al Javi pel seu assesorament, a la feina dels becaris del departament deTSC , en especial al Sergi, per la seva eficàcia i dedicació, a les nostres famílies per aguantar els nostres nervis i tenir sempre unes paraules de consell, als professors de l'EPSC que ens han aclarit els nostres dubtes i sobretot al nostre director per estar allà sempre que l'hem necessitat

ÍNDEX

ÍNDEX DE FIGURES I TAULES

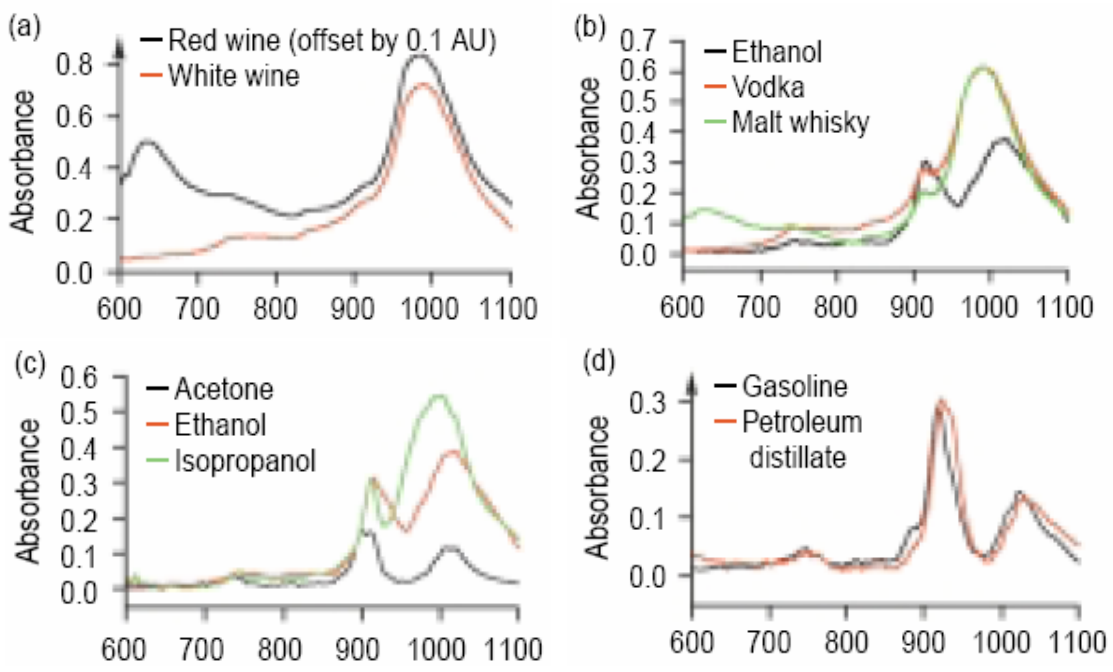
CAPÍTOL 1 INTRODUCCIÓ

1.1 Característiques i aplicacions dels espectròmetres

La tecnologia basada en l'espectroradiometria ha patit una considerable expansió i desenvolupament en els últims 30 anys. Per això actualment hi ha molt tipus d'espectròmetre. Definirem i classificarem les diferents tècniques que es fan servir als espectròmetres

Els espectròmetres utilitzats en els equips de mesura moderns són molt més petits que els clàssics i ofereixen una millor relació qualitat preu. Això ha fet que l'espectrometria sigui present en nombrosos camps, per exemple, anàlisis químics i de qualitat, anàlisis mèdics, per testejar la llum emesa per diferents cossos, mesurament de la pol·lució, etc., i actualment ha augmentat notòriament el seu ús en processos de control, plantes químiques i indústries de components semiconductors.

A continuació podem veure els diferents espectres de diferents elements, i de les diferents aplicacions dels espectròmetres



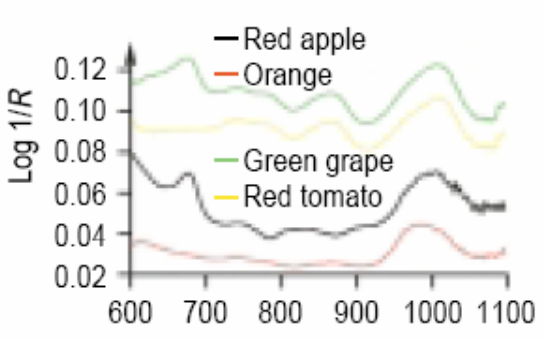


Fig. 1.1 Diferents espectres d'absorció característics de diferents elements

Els espectròmetres clàssics tenien com a elements principals, un raig incident, un element dispersiu rotatori (normalment una lent) , un raig de sortida i un únic detector. Els principals desavantatges que presentaven aquests espectròmetres (les mesures no es podien fer en paral·lel sinó seqüencialment, la mobilitat no desitjada que presentaven els diferents elements i les grans dimensions dels aparells) van provocar l'aparició dels espectròmetres basats en array de fotodíodes.

Aquest últim tipus d'espectròmetres utilitzen un detector format per una cadena de detectors simples capaços de fer la descomposició espectral de la llum incident de cop i per tant només necessitem elements fixes. Aquestes característiques han suposat una reducció notable de la mida d'aquests elements i han fet possible la seva integració en dispositius electrònics i poder ser governats per microcontroladors i fins i tot poder establir una comunicació amb ordenadors.

Els espectròmetres basats en array de fotodíodes estan formats per un raig incident, un element òptic i cadena de fotodetectors (fotodíodes). La seva principal avantatge és el fet de poder fer el mesurament de tot l'espectre en paral·lel, el que implica que el temps necessari per fer la mesura es veu reduït. A part, també solen ser dispositius robustos i de petita mida.

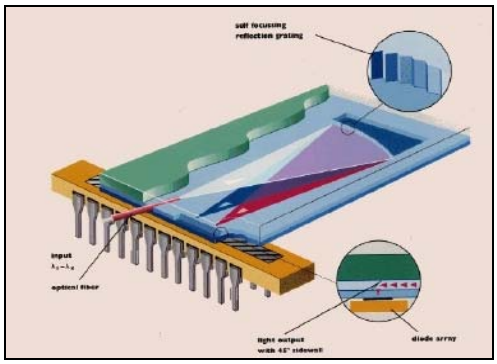


Fig. 1.2 Esquema intern d'un espectròmetre

Com més endavant explicarem quan definim el nostre sensor, dins els espectròmetres basats en array de fotodíodes en podem trobar de diferents tipus, segons les característiques dels elements òptics utilitzats.

1.1.1 Signatura espectral

La informació que rebem d'un espectròmetre ens permet obtenir la signatura espectral del cos al qual l'exposem.

Cada objecte o element emet un espectre electromagnètic específic, en funció de la seva naturalesa i de les radiacions que rep. La reflectància d'un objecte al llarg de tot l'espectre electromagnètic és el que anomenen signatura espectral.

Cada objecte o element està caracteritzat per un signatura espectral concreta, que el fa distingible dels altres.

1.2 Integració d'un espectròmetre dins d'una sonda oceanogràfica

Per entendre millor en que consisteix el projecte es necessari conèixer el fonaments i característiques dels seus elements.

Destaquem que l'aplicació final de l'espectròmetre realitzat es farà servir com a payload de la sonda oceanogràfica, però no només te aquesta aplicació, sinó que es pot fer servir per a diferents plataformes de mesura, com a vehicles marins amb una determinada autonomia anomenats ROV's i AUV's que més endavant explicarem.

1.2.1 Utilització de les sondes oceanogràfiques

La utilització de sondes oceanogràfiques s'ha fet molt popular ens el últims anys degut a la importància que té l'estudi de les profunditats marines.

Sempre hi ha hagut interès en l'estudi del medi marí, i l'aparició de les sondes oceanogràfiques ha ampliat les possibilitats d'obtenir informació a temps real d'aquest medi, la qual cosa permet fer patrons, anàlisis més acurats, etc. Les sondes ens ofereixen els avantatges que dóna l'electrònica vers altres mètodes d'adquisició de dades, com agafar mostres d'aigua i analitzar-les al laboratori, o fer immersions amb personal qualificat que prengui mostres "in situ". A més cal destacar el fet de que les sondes ens permeten obtenir major nombre de mesures per unitats de temps, mesures mes fiables, possibilitats de fer mesures en indrets inhòspits, etc. I tot això a un cost raonable

Dintre del camp de les sondes marines podem trobar-ne de molts tipus, segons la seva sofisticació i preu. Però l'estructura bàsica d'una sonda és la d'un recipient, més o menys aerodinàmic, totalment estanc i que en el seu interior pot contenir diferent nombre de sensors indicats per a la mesura de diferents paràmetres i que normalment són alimentats i governats des de la superfície.

Fig. 1.3 Sonda oceanogràfica

1.2.2 ROV'S i AUV'S

A partir d'aquest model molt senzill podem trobar la següent evolució de la sonda marina. Si incorporem motors, o hèlix que ens permetin controlar el moviment del nostre aparell, aconseguirem vehicles marins, que ens amplia la capacitat de les prospeccions marines que podem arribar a fer.

No obstant, el fet d'ampliar les possibilitats d'estudi marí, també amplia la complexitat del disseny, i ara en aquests vehicles seran necessaris sensors de posició, sensor òptics, sensor de moviment, motors, hèlix, microprocessadors més potents,...

Bàsicament, podem classificar aquests vehicles en dos tipus

- *ROV (Remotely Operated underwater Vehicle).* Aquest vehicle està controlat mitjançant un cable. Des d'un PC de bord, es controlen els moviments que volem que el vehicle realitzi. La principal avantatge és que no l'alimentació que consumeixi l'aparell no és gaire limitada, però no obstant el cable unit al vaixell, limita molt els moviments del vehicle.
- *AUV (Autonomus Underwater Vehicle)*. Aquest robot és totalment autònom, no necessita cable d'alimentació/control, i no es pot controlar a temps real, sinó que es programa per una funció específica. En aquests

aparells s'ha de tenir en compte el consum de potència, i el disseny hidrodinàmic, per tal que la resistència a l'aigua sigui mínima.

Fig. 1.4 Imatge esquerra ROV i imatge dreta AUV

1.3 Característiques del projecte

En el projecte integrem un espectròmetre dins d'una sonda marina per tal d'obtenir la signatura espectral de diferents trams de la mateixa columna d'aigua.

Posteriorment, amb l'estudi d'aquestes lectures podrem descobrir l'existència de diferents components anormals en la composició de l'aigua marina, com poden ser contaminants, concentracions extraordinàries d'algues, etc. Però com fer aquest reconeixement ja no forma part del nostre projecte.

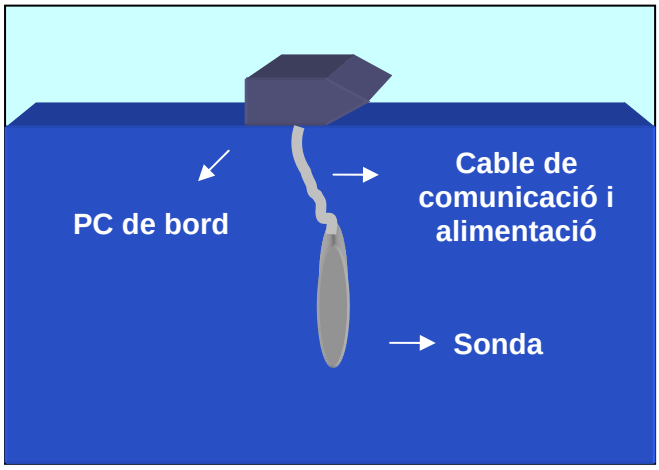


Fig. 1.5 Esquema del sistema complert

El nostre sistema a desenvolupar, es llançarà des d'un vaixell amb una sonda. Aquesta sonda anirà controlada per un PC de bord, i pel cable de comunicació i d'alimentació. Donat el protocol RS-232, la fondària màxima a la qual podrà

arribar la sonda serà de 50 metres (considerant un cable de baixa capacitat 50pF/m). A profunditats superiors, no es garanteix la correcta comunicació entre la sonda i el PC de bord. No obstant, destaquem que s'utilitzarà aquest protocol al prototipus perquè és més senzill de connectar al PC i no es necessitarà cap adaptador, però el sistema final, anirà implementat amb RS-422 o RS-485, que permet treballar a distàncies més grans (fins a una longitud de cable de 1200m).

La part a realitzar, és el disseny i la implementació de tot el sistema de control del sensor que realitza la descomposició freqüencial, el tractament i l'acondicionament de les dades i la posterior recuperació a través d'una interfície gràfica utilitzant les possibilitats que ens ofereix el llenguatge Matlab.

Tot el conjunt aconseguirà que l'adquisició de dades sigui totalment transparent per l'usuari.

Així doncs, en el projecte trobem dues part importants diferenciades:

HARDWARE

En la part de hardware ens preocuparem bàsicament de realitzar el disseny d'una placa que ens permeti integrar tot el sistema dins de la sonda marina tenint en compte les reduïdes dimensions d'aquesta i que el disseny permeti la connexió a l'alimentació i a la comunicació sèrie amb el PC.

SOFTWARE

En la part de software treballarem amb dos llenguatges.

En llenguatge C realitzarem la programació del microcontrolador (dsPIC), que serà l'encarregat de controlar el sensor, l'AD i guardar en memòria la informació rebuda.

En canvi la realització de la interfície es realitzarà en Matlab i serà en aquest entorn en el que realment controlarem del microcontrolador i la seva comunicació amb el PC. En entorn Matlab es realitzarà la representació gràfica de les dades obtingudes pel sensor i es presentaran de la manera més clara possible i amb la possibilitat de ser tractades i/o guardades.

Vegem a continuació un diagrama de blocs del sistema complert , on també s'especifiquen les connexions i el tipus de comunicació entre cada element.

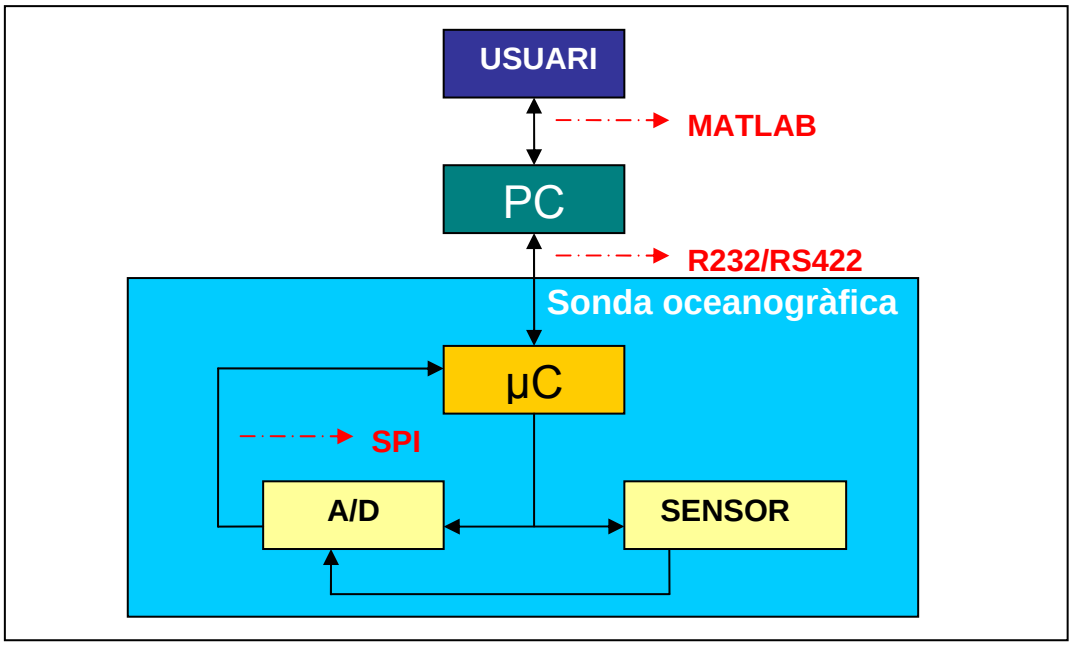


Fig. 1.6 Esquema del sistema final

1.4 Antecedents i objectius

El projecte vol ser la millora d'un projecte ja existent. L'objectiu és el mateix, però s'hi ha introduït nombroses millores:

- El sensor (espectròmetre) ha estat substituït respecte al projecte anterior. La senyal de sortida presenta les mateixes característiques, però el seu control és més senzill.

- S'ha substituït el microcontrolador de tipus PIC, per un dsPIC de menor mida i majors prestacions. Aquest nou microcontrolador, a diferència del projecte anterior serà programat mitjançant llenguatge C (cosa que també hagués estat possible implementar al PIC) i l'entorn de programació IAR Workbench per tal de tenir més flexibilitat i capacitat de reutilització de codi.

- En el projecte anterior es va utilitzar un AD intern de 10 bits del microcontrolador. Nosaltres, en canvi, a l'hora de codificar les dades rebudes des del sensor utilitzarem un A/D extern de 16 bits i implementarem la comunicació SPI entre el microcontrolador i aquest AD. Això ens permetrà obtenir una lectura amb més resolució.

- S'ha partit del codi Matlab inicial però se n'ha fet una reestructuració per tal de fer-lo més llegible i entenedor. També si han implementat algunes funcions noves, com el control automàtic del temps d'integració, que va adaptant el temps en que els fotosensors estan exposats al raig incident per tal que sempre s'ocupi el màxim marge dinàmic possible.

- A l'hora de realitzar la placa de circuit integrat s'han tingut en compte unes mides especifiques i s'ha considerat que els components fossin el més adequats possibles tenint en compte l'aplicació.

- El fet d'aconseguir el disseny d'aquest sistema complert, a més del desenvolupament Hardware i software implicarà la familiarització amb nous llenguatges de programació, noves eines de disseny, i nous entorns de treball.

CAPÍTOL 2 DESCRIPCIÓ HARDWARE

2.1 El dsPIC (Digital Signal Controller)

2.1.1 Característiques generals

Totes les funcionalitats de la placa estan controlades pel dsPIC 30F6012 de Microchip. També serà l'encarregat de la comunicació amb el PC. Podem dir llavors que actua com a interfície entre la placa i el PC.

El motiu pel qual vam escollir aquest dispositiu, i no un altre, és que els dsPIC son una nova família de microcontroladors . La novetat consisteix en considerar dos aspectes que fins ara no s'havien combinat, la tecnologia dels microcontroladors i la dels processadors digitals de senyals o DSP.

La principal avantatge de combinar aquests dos aspectes es que amb una arquitectura Harvard modificada, obtenim un alt rendiment en quant a control gràcies al microcontrolador (MCU), i també una gran velocitat de processat de senyals (DSP). Aquests dispositius poden arribar a velocitats de 30 MIPS.

La programació és més senzilla que en el cas dels microcontroladors, ja que gràcies als compiladors podem programar els dsPIC en llenguatge C.

Per tant, podem dir que farem servir dispositius dsPIC degut als grans recursos que ens proporciona aquest element i alhora això permetrà futures ampliacions, modificacions i integracions de diferents projectes del departament.

Fent un ràpid anàlisi de les prestacions del dsPIC 30F6012 vam veure que aquest dispositiu satisfà els requeriments necessaris per a la realització del nostre projecte, i per a futures ampliacions.

La mida del dispositiu és prou petita considerant els 64 pins del quals consta. A més de les capacitats d'entrada/sortida, tenim estàndards de protocols específics 2 ports CAN, 2 ports UART, 2 ports SPI i 1 port I2C, un conversor AD intern de 12 bits, i una gran potència de càlcul.

La memòria de programa és de 144Kbytes, també té una SRAM de 8192Bytes i una EEPROM de 4096 Bytes.

dsPIC30F6011/6012/6013/6014 Controller Families

Device	Pins	Program Memory		SRAM Bytes	EEPROM Bytes	Timer 16-bit	Input Cap	Output Comp/Std PWM	Codec Interface	A/D 12-bit 100 Ksps	UART	SPI™	I²C™	CAN
		Bytes	Instructions											
dsPIC30F6011	64	132K	44K	6144	2048	5	8	8	—	16 ch	2	2	1	2
dsPIC30F6012	64	144K	48K	8192	4096	5	8	8	AC'97, I²S	16 ch	2	2	1	2
dsPIC30F6013	80	132K	44K	6144	2048	5	8	8	—	16 ch	2	2	1	2
dsPIC30F6014	80	144K	48K	8192	4096	5	8	8	AC'97, I²S	16 ch	2	2	1	2

Taula. 2.1 Característiques generals dels dsPIC

Per últim dir que tenim un ampli rang de velocitats disponibles, que ens permet utilitzar velocitats adaptades per poder treballar correctament amb el sensor on tenim un compromís del temps vs sensibilitat.

2.1.1.1 Paràmetres tècnics

CPU RISC d'alt rendiment

- És capaç de processar fins a 30 MIPS (amb un rellotge de 40MHz), amb instruccions de 24 bits i un camí de dades de 16 bits. A més a més incorpora un registre w de 16x16 bits (de treball).
- La memòria de programa pot adreçar fins a 144Kbytes (FLASH), la memòria de dades fins a 8 Kbytes (RAM), i una EEPROM de 4Kbytes.
- Es disposen de 41 interrupcions, amb 8 nivells de prioritat seleccionables per l'usuari

DSP

- Dos busos de dades
- Totes les instruccions de la DSP s'executen en un sol cicle (MAC Multiply-Accumulate operation).
- Consta d'un multiplicador d'enters de 17 x 17 bits en Hardware d'un sol cicle.

Perifèrics integrats

- 5 fonts d'interrupcions externes
- Inclou 5 temporitzadors (timer) de 16 o 32 bits. Que també poden actuar com a comptadors
- Consta de dos mòduls UART amb buffers tipus FIFO, dos mòduls CAN que compleixen l'estàndard CAN2.0B, un mòdul I2C (tant de mestre com d'esclau) amb adreçament de 7-10 bits i dos mòduls SPI de 3 cables que suporta els 4 modes estàndard.

Característiques analògiques

- Conversor A/D integrat de 12 bits d'alta velocitat de mostreig, amb 100Ksps. És possible mostrejar durant SLEEP (estat de la dsPIC de baix consum)
- Detecció de baixa tensió programable (PLVD). En cas de detecció permet activar una interrupció.
- Reset programable de Brown-out (BOR), que activa un reset en determinades condicions d'alimentació

Característiques especials del microcontrolador

- Fins a 10.000 cicles d'esborrament / escriptura a la memòria de programa (FLASH)
- Fins a 100.000 cicles d'esborrament / escriptura a la memòria de dades EEPROM
- Auto-reprogramable sota control de software
- Reset d'alimentació (Power-on reset, POR), temporitzador d'alimentació (Power-up Timer, PWRT) i temporitzador d'inici d'oscil·lador (Oscillator Start-up Timer, OST)
- Protecció de codi programable
- Selecció mode SLEEP (baix consum de potència)
- Programació sèrie connectat al circuit (In-Circuit Serial Programming, ICSP)
- Opcions de selecció d'oscil·lador. Un PLL, oscil·lador de cristall, circuit RC, ressonador ceràmic o senyal de rellotge extern.

Tecnologia CMOS

- Tecnologia FLASH de baix consum i d'alta velocitat
- El rang d'operació de tensió es troba entre 2,5 i 5,5 V
- Els rangs de temperatura suportats comprenen els anomenats Industrial and Extended
- Baix consum de potència

2.1.1.2 Encapsulat

El dsPIC30F6012 està disponible amb un encapsulat TQFP de 64 pins. No obstant, està adaptat a un sòcol PLCC que protegeix els pins d'alimentació amb condensadors, i permet l'extracció del dispositiu en cas de que es faci malbé. Això ens facilita el disseny de la placa, i la manipulació del dispositiu alhora de reprogramar-lo, fer proves, i la reutilització d'aquest.

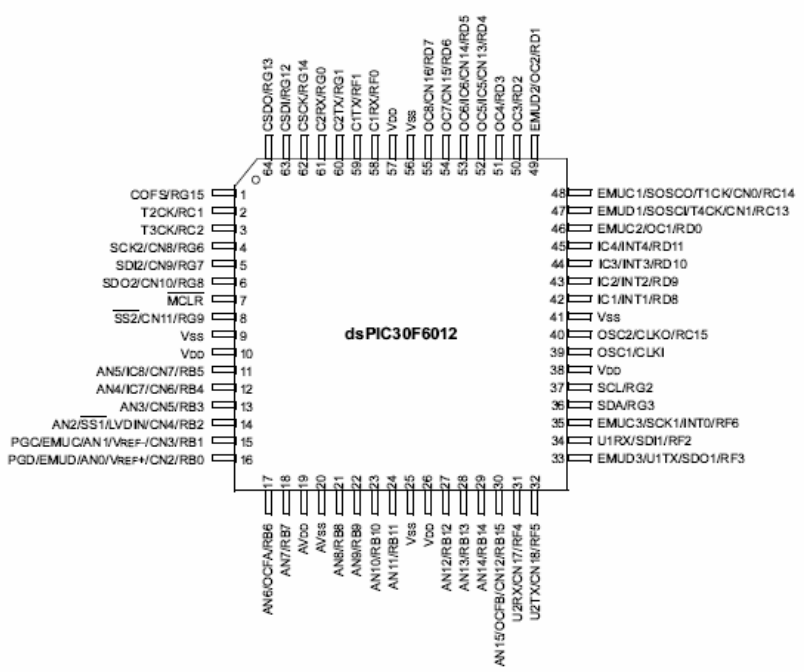


Fig. 2.1 Patillatge i encapsulat del dsPIC escollit

2.1.2 Organització de la memòria

Hi han tres blocs diferenciats de memòria:

- La memòria de programa (FLASH)
- La memòria de dades RAM
- La memòria de dades EEPROM

La memòria de programa i de dades utilitzen busos separats, la qual cosa permet un accés simultani a ambdós blocs. En el nostre projecte només utilitzarem la memòria de programa pel codi i la RAM per emmagatzemar-hi la seqüència a enviar pel sensor, i per emmagatzemar-hi la seqüència a enviar de la conversió de l'AD extern, ja que és més ràpida i eficient per escriure i llegir dades.

2.1.2.1 Organització de la memòria de programa

El dsPIC30F6012 consta de 4M x 24 bits d'espai d'adreces per la memòria de programa.. El mapa de memòria de programa es divideix en l'espai de programa d'usuari i l'espai de la configuració d'usuari. L'espai de programa d'usuari conté el vector de Reset, les taules de les interrupcions, la memòria de programa, i la memòria de dades EEPROM. L'espai de la configuració de l'usuari conté els bits de configuració no-volàtils, per tal de fixar les opcions i identificar cada dispositiu,

Hi ha tres maneres d'accedir-hi a l'espai de programa:

- Mitjançant els 23 bits del Comptador de Programa (Program Counter, PC)
- Mitjançant instruccions a taules de lectura (TBLRD) i taules d'escriptura (TBLWT)
- Mapejant un segment de 32Kbytes de memòria de programa dins de l'espai d'adreces de memòria de dades.

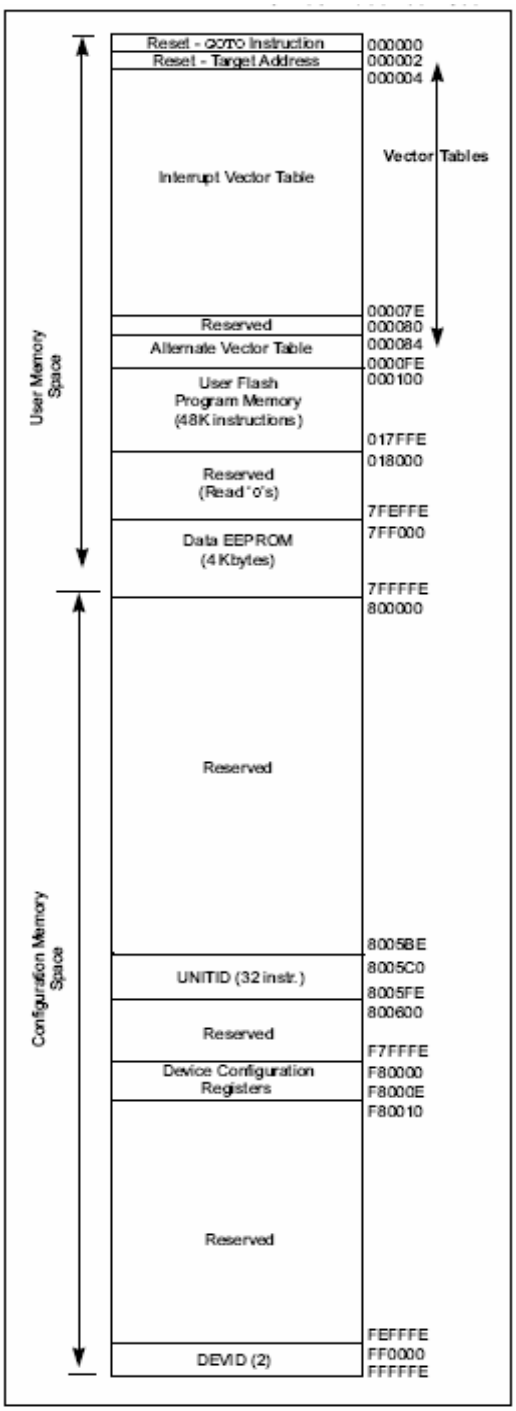


Fig. 2.2 Mapa de memòria de programa

Destaquem que l'adreça de l'espai de programa s'incrementa en dos entre les successives paraules de programa, per tal que hi hagi compatibilitat amb l'espai d'adreces de dades. L'accés a l'espai de programa es restringeix a les següents adreces 0x000000 fins 0x7FFFFE

2.1.2.2 Organització de la memòria de dades

Tots els registres interns i l'espai de memòria de dades està format per paraules de 16 bits. El dsPIC30F6012, consta de dos espais de dades als quals es pot accedir de manera paral·lela per algunes instruccions de la DSP, o amb mode simple amb un rang d'adreces de 64Kbyte per instruccions MCU.
Les adreces de l'espai de memòria de dades de 0x0000 fins 0x07FF estan reservats per registres de funcions especials (SFR, Special Function Registers), que contenen els bits de control i d'estat de la CPU i certs perifèrics que porta implementats.

La memòria RAM comença a l'adreça 0x0800 i està dividida en dos espais de dades, X i Y. Per escriure dades, l'accés a l'espai X i Y és simple, lineal, mentre que per llegir, l'accés a l'espai X i Y pot ser simultàniament, o linealment.

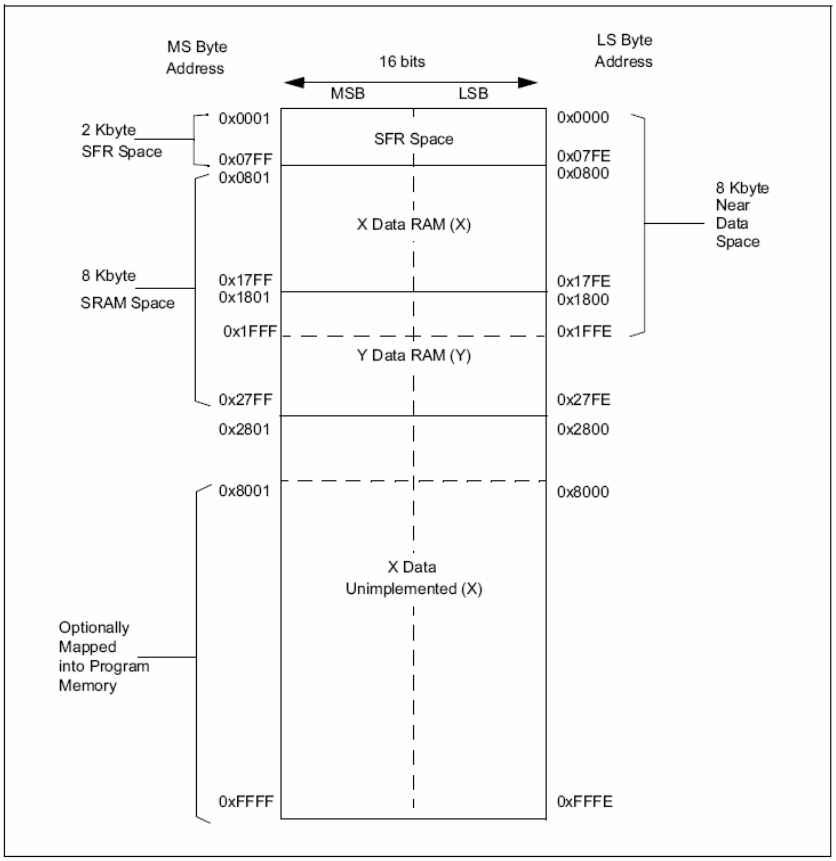


Fig. 2.3 Mapa de la memòria de dades

2.1.3 Ports d'E/S

El dsPIC30F6012 té cinc ports disponibles (és a dir, tots els pins possibles, excepte els Vdd, Vss, MCLR i OSC1/CLKI). Tots els ports d'E/S que configurem com a entrades, consten d'un trigger Schmitt (comparador amb histèresi), per tal d'aconseguir una certa immunitat davant del soroll. La majoria d'aquests pins estan multiplexats per tal que facin altres possibles funcions específiques d'alguns perifèrics. Es pot triar cadascuna de les funcions dels pins mitjançant software.

Cadascun dels ports, conté tres registres associats:

- El registre 'TRISx' que determina la direcció del port ('1' = entrada i '0' = sortida). Destaquem que per defecte, tots els ports estan configurats com a entrada.
- El registre 'PORTx' que estableix els nivells llegits als pins del port
- El registre 'LATx' com alternativa al registre anterior

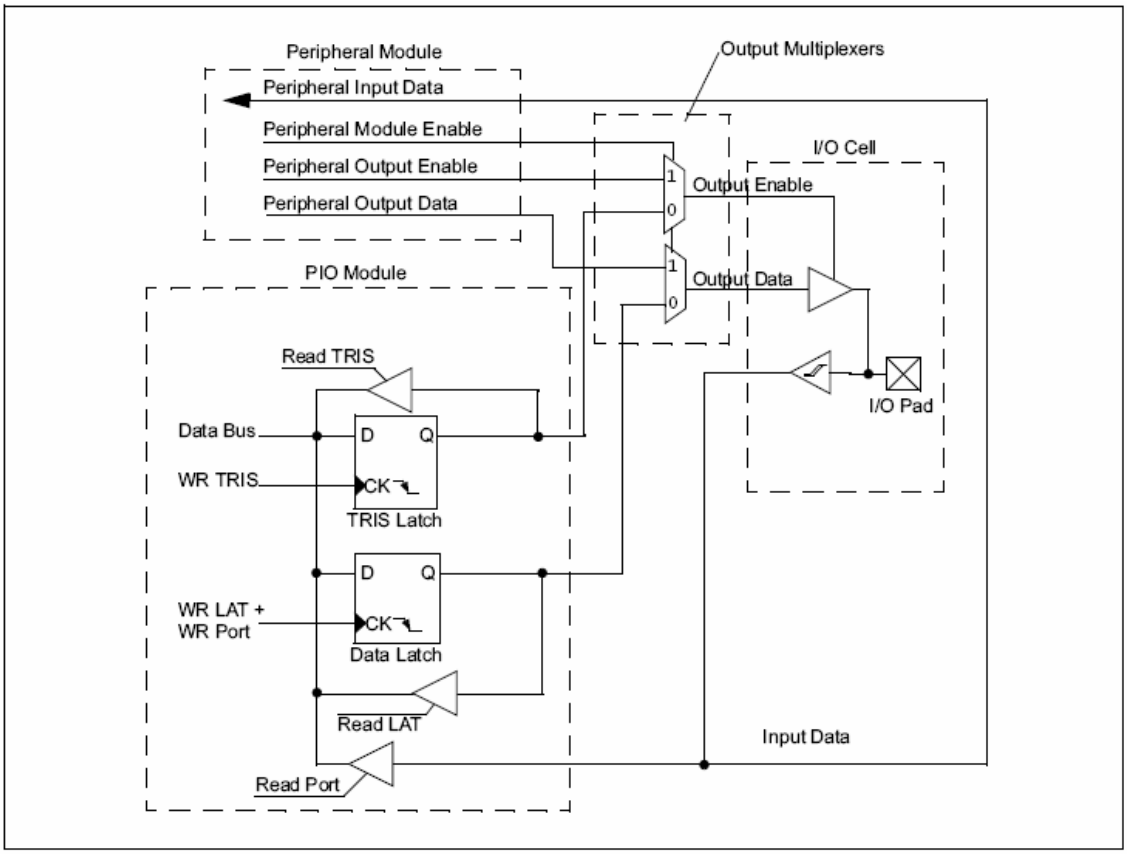


Fig. 2.4 Estructura dels PORTS E/S

2.1.4 Mòdul UART

El port USART (Universal Synchrounous Receiver Transmitter), també conegut com SCI (Serial Comunications Interface), ens servirà com a interfície entre la placa i l'ordinador. Mitjançant l'ordinador controlarem l'inici de totes les operacions possibles, com per exemple, l'adquisició de dades, l'enviament de les mostres, el càlcul del temps d'integració del sensor,...

Les principals opcions i característiques que ens ofereix el mòdul USART del dsPIC30F6012 són:

- Full-duplex de 8 o 9 bits
- Selecció de Paritat (9 bits), o sense paritat (8 bits)
- Un o dos bits de parada
- Generació de velocitat de transmissió amb un prescaler de 16 bits. Això ens permet obtenir un rang de 38bps fins a 1,875 Mbps a una velocitat de 30MHz per instrucció.
- Buffer de transmissió i de recepció de 4 paraules

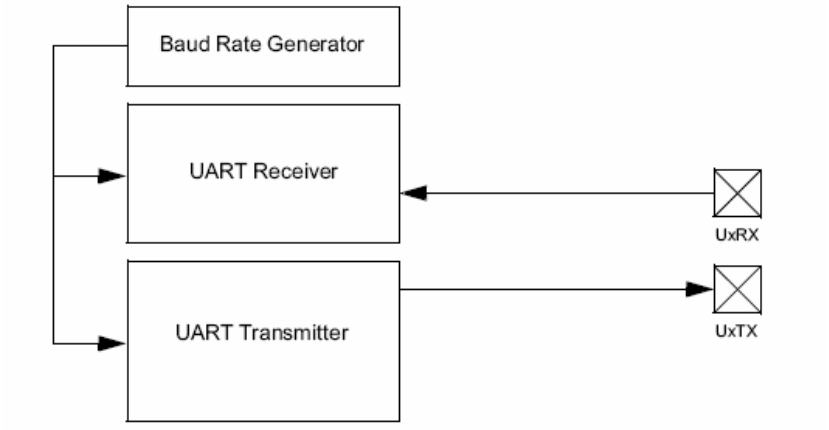


Fig. 2.5 Estructura simplificada del mòdul UART

La configuració la qual farem servir, serà de transmissió de 10 bits amb codificació NRZ, és a dir, tenim un bit d'inici a nivell baix, vuit bits de dades, i un bit de final de trama a nivell alt. El primer bit de la trama és sempre el menys significatiu.

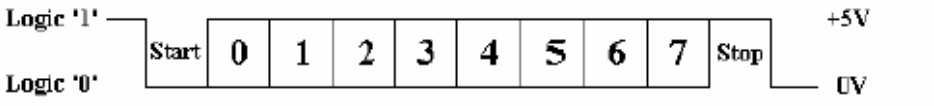


Fig. 2.6 Comunicació RS-232

Més endavant, a la part de programació explicarem la configuració realitzada.

2.1.4.1 Mòdul transmissor

El transmissor UART consisteix en un registre de desplaçament anomenat TSR (Transmit Shift Register). Les dades a Transmetre es carreguen al registre TXREG a través del bus de dades, i tot seguit són enviades cap al TSR. El registre TSR no es carrega totalment amb les dades fins que no es transmeti el bit de 'stop' d'aquest byte enviat. Un cop enviat el bit de 'stop', el registre TXREG es buida i s'activa el flag bit TXIF. Si tenim activat el bit UTXISEL, llavors també es produirà una interrupció per transmissió.

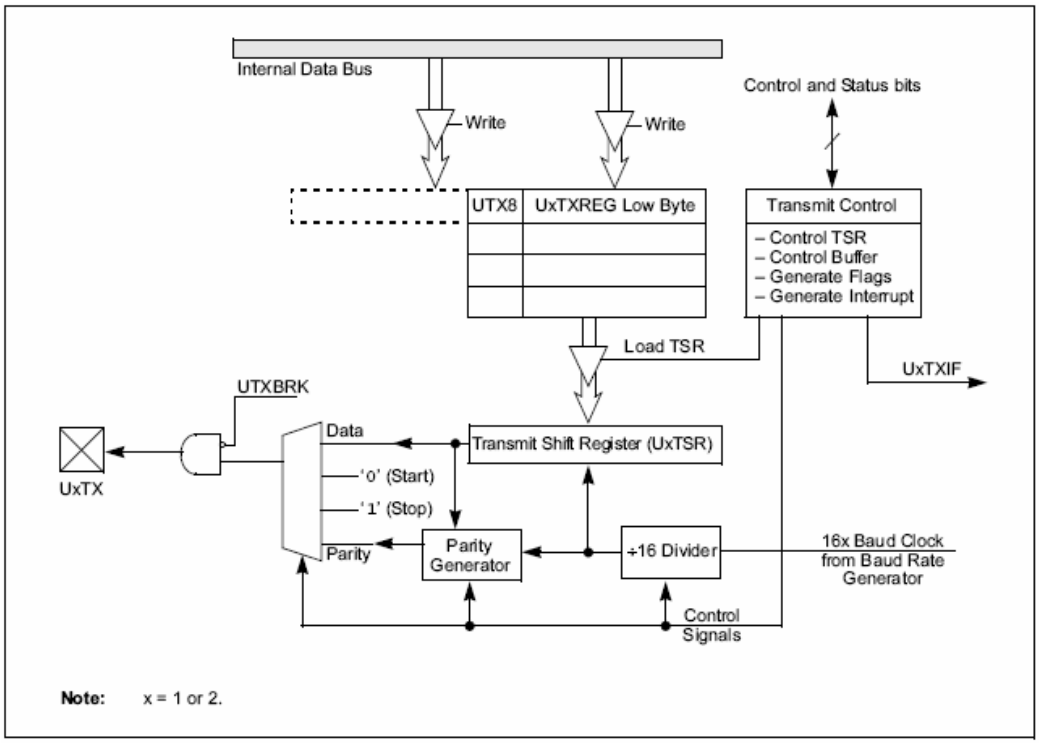


Fig. 2.7 Estructura interna del transmissor UART

El mòdul BRG (Baud Rate Generator), serveix per controlar la velocitat de transmissió del port UART, mitjançant bàsicament un comptador de 16 bits. Carregant un determinat valor al registre SPRBG, podem modificar la velocitat del port que òbviament ha de ser la mateixa que la programada pel port sèrie del PC.

2.1.4.2 Mòdul receptor

La recepció UART consisteix bàsicament en un registre de desplaçament que treballa a gran velocitat, aproximadament unes 16 vegades la tassa d'enviament. Les dades es reben als diferents pins possibles del port F,

depenent de la configuració i del port UART escollit (en disposem de dos) (veure figura 2.1)

Destaquem que per comunicar el dsPIC amb l'ordinador cal adaptar els nivells de tensió a la sortida del port UART del dsPIC ('0'=0V i '1' = 5V) als nivells de tensió que utilitza el port sèrie del PC ('0'=12V i '1' = -12V). Utilitzarem l'adaptador de ports MAX232A per fer-ho

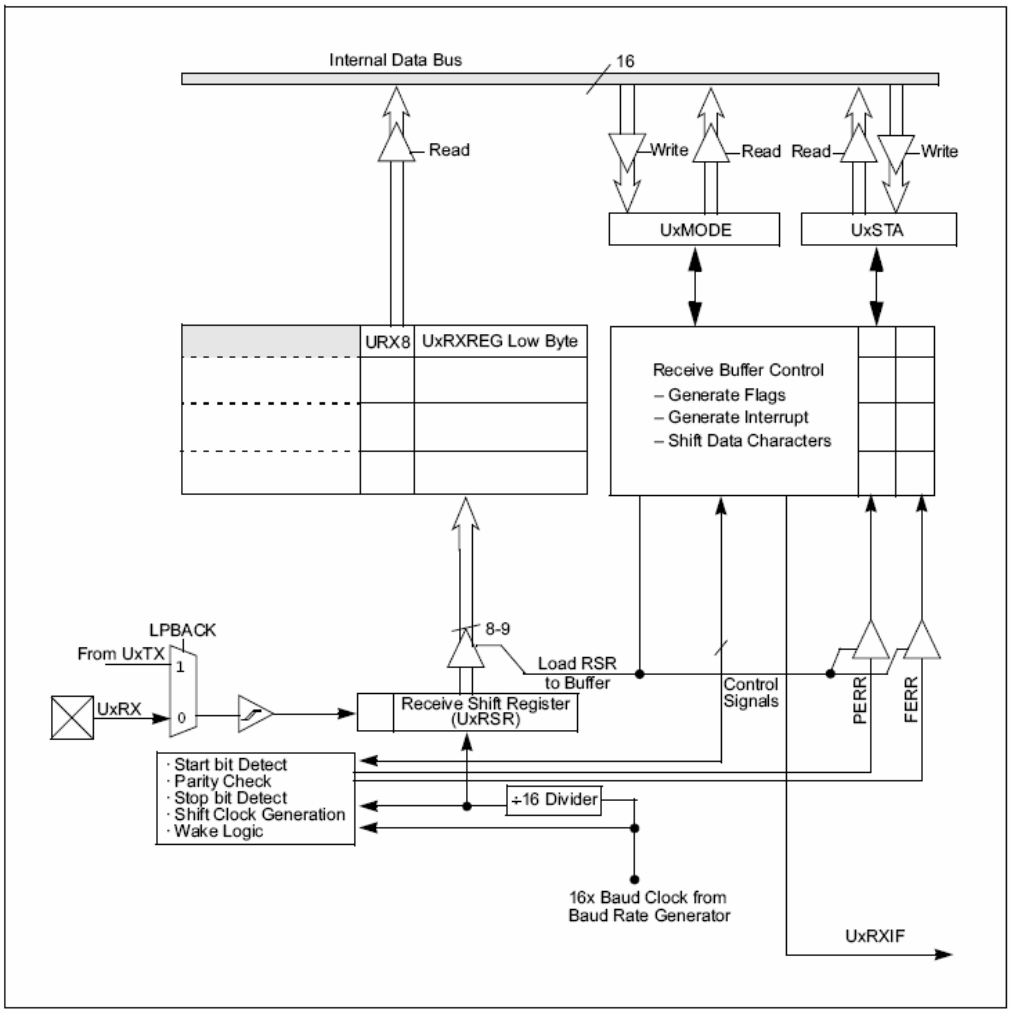


Fig. 2.8 Estructura interna del receptor UART

2.1.5 Mòdul SPI

El mòdul SPI (Serial Peripheral Interface), és un mòdul que es fa servir per comunicacions sèries síncrones. És un mode molt comú per comunicar-se amb perifèrics com EEPROM's registres de desplaçament, conversors A/D o altres microcontroladors

Al dsPIC30F6012 n'hi ha dos mòduls SPI. El port sèrie SPI consisteix en els següents registres SFR (Special Function Registers)

- SPIxBUF: Adreça a l'espai SFR que es fa servir per emmagatzemar les dades transmeses i rebudes al buffer
- SPIxCON: Registre de control que configura el mòdul per operar en diversos modes
- SPIxSTAT: Registre de estat que indica vàries condicions d'estat de la comunicació.

La comunicació SPI consta de quatre pins bàsics:

- SDIx: dades d'entrada sèrie
- SDOx: dades de sortida sèrie
- SCKx: senyal de rellotge
- SSx: activació de l'esclau o de sincronització de trama.

El mòdul SPI pot ser configurat per operar amb 3 o 4 pins. Si fem servir el mode de tres pins, és perquè el SSx no es fa servir.

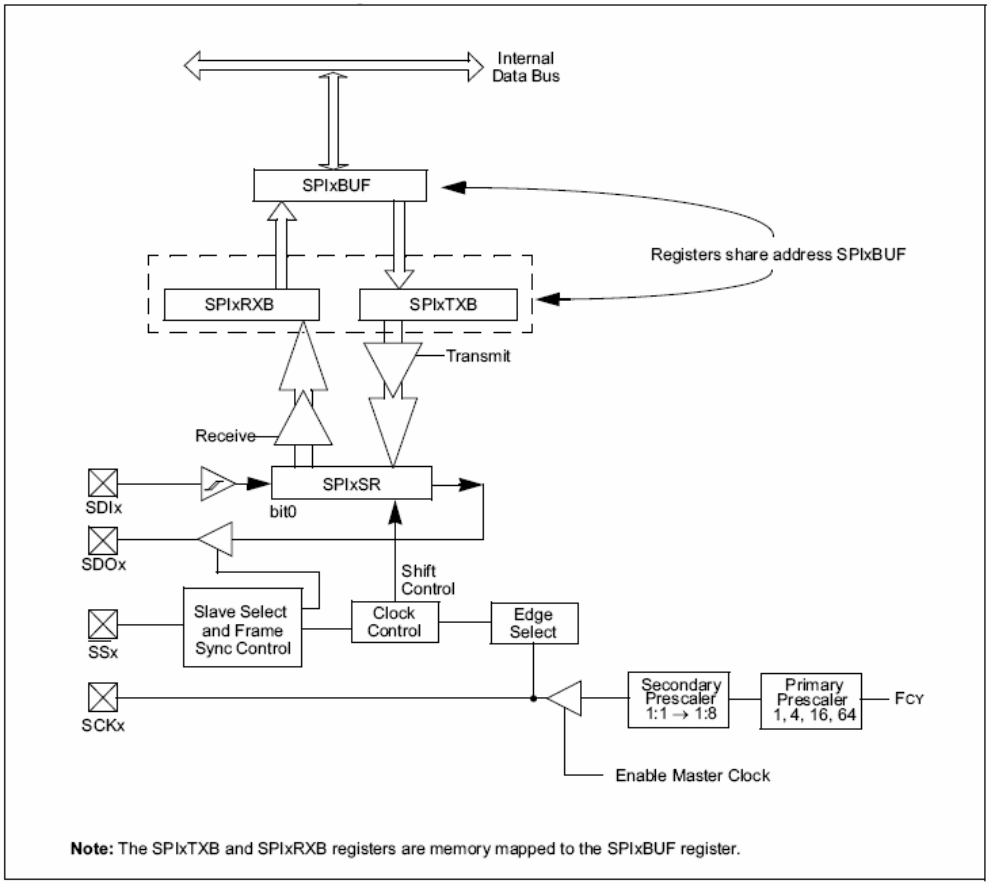


Fig. 2.9 Estructura del mòdul SPI

No obstant, destaquem que al nostre cas, farem servir el SPI per comunicar el conversor AD amb el dsPIC, per tant, només en farem servir dos pins, el SCKx, per on anirà el senyal de rellotge i el SDIx per on aniran les dades, ja que l'AD escollit no hi ha dades de sortida des de l'AD cap al dsPIC.

2.1.6 Mode ICD

El mode ICD (In-Circuit Debugging) es fa servir, per conèixer els valors que el microprocessador està guardant o fent servir al programa que em carregat en memòria a temps real.
Això ens permet localitzar molt més ràpidament on es produeixen els errors al programa que carreguem, i per tant, és molt més fàcil identificar les incoherències al codi de programació.

No obstant, per fer servir aquest mode, necessitarem el MPLAB ICD2, que com més endavant explicarem, ens permet programar i debugar mitjançant una connexió ràpida USB.

Quan estem en mode debug, no estan habilitades tots els pins d'entrada/sortida, ja que aquest mode fa servir qualsevol dels següents parells de pins anomenats EMUD/EMUC, EMUD1/EMUC1, EMUD2/EMUC2, EMUD3/EMUC3. On la línia de EMUD fa referència a la línia de dades (o PGD) i l'EMUC fa referència a la línia de rellotge (o PGC).

Aquests pins s'interconnectaran amb el mòdul ICD2 (in circuit Debugger), de manera que els pins seleccionats es faran servir per MPLAB ICD2 per enviar comandes i rebre respostes, i per enviar i rebre dades. Per fer servir la funció in circuit debugger és necessari implementar una connexió ICSP (In circuit Serial Programming), la qual requereix els següents pins bàsics al microcontrolador, que son MCLR, Vdd, Vss, PGC, PGD i els pins EMUDx/EMUCx.

Això ens dona dos possibilitats de configuració:

- Si seleccionem els EMUC/EMUD només necessitem 5 pins ja que els pins EMUC/EMUD estan multiplexats amb les funcions dels pins PGD i PGC

- Si seleccionem els EMUC1/EMUD1, EMUC2/EMUD2 o EMUC3/EMUD3 necessitem 7 pins ja que els pins EMUCx/EMUDx no estan multiplexats amb les funcions dels pins PGD i PGC

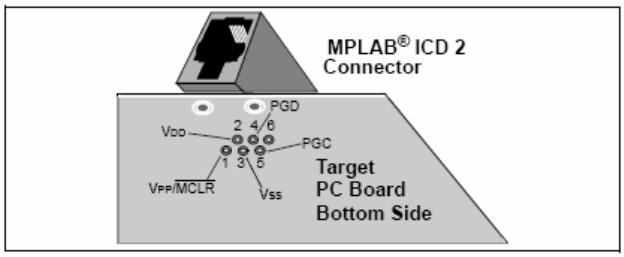


Fig. 2.10 Connector per l'ICD2

2.2 El sensor S8378 – 256N/SPL

2.2.1 Característiques generals

Un espectroradiòmetre capta un feix de llum recollit d'una mostra òptica. Mitjançant propietats de l'òptica es dispersa aquest feix de llum en longituds d'ona a través d'una fila de fotodíodes, i s'envien uns determinats valors de tensió proporcionals a la llum que ha arribat a cada díode. Cada espectròmetre està dissenyat per treballar en una determinada franja de l'espectre radioelèctric, en funció de la capacitat, del temps d'exposició i la quantitat de fotodetectors, de la geometria de les lents, dels filtres òptics,...

Hi ha tres tipus de mesures òptiques que pot fer un espectròmetre: transmissió (és la fracció de l'energia de la llum a cada longitud d'ona que passa a través d'una mostra), reflexió (és la fracció de l'energia de la llum que és reflexada de la superfície de la mostra) i radiació (és la quantitat d'energia lumínica a cada longitud d'ona procedent d'una mostra radiant).

El sensor escollit ha estat el sensor S8378-256N/SPL de la casa Hamamatsu Photonics. És un sensor d'imatge lineal que fa servir la tecnologia CMOS. Està format per 256 fotodíodes molt sensibles a les variacions de llum. Aquests 256 fotodíodes ens donen les diferents longituds d'ona.

La llum que arriba als fotodíodes, és guiada mitjançant un cable de fibra òptica. A l'entrada de la cavitat, es dirigeix el feix de llum cap a un cristall el qual mitjançant reflexió arriba la llum descomposada a cada fotodíode, per tant les mesures òptiques d'aquest sensor són de reflexió.

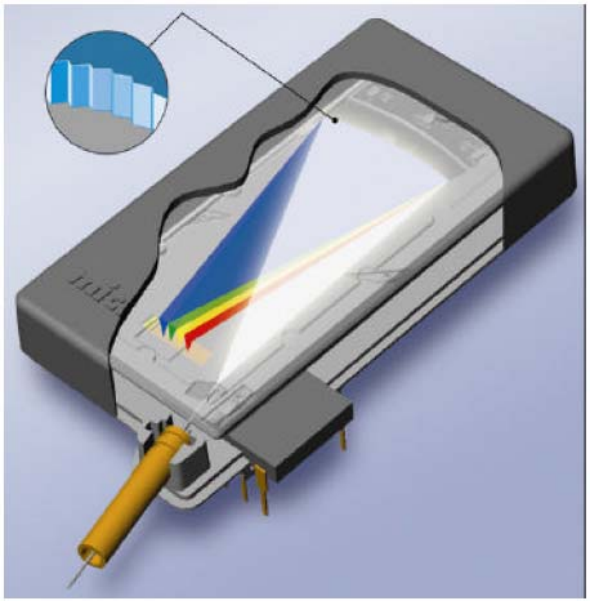


Fig. 2.11 Imatge del sensor

Els sensors d'imatges lineals amb tecnologia CMOS es basen en línies de fotodíodes, construïts amb amplificadors d'integració. Aquests amplificadors, a més de garantir un baix nivell de soroll, controlen el temps de lectura de cada mostra de llum (fins a un màxim de 500KHz) i el guany. Mitjançant el temps de lectura de cada mostra, controlem el temps d'exposició a la llum de cada fotodíode.

A continuació podem veure l'estructura interna del sensor:

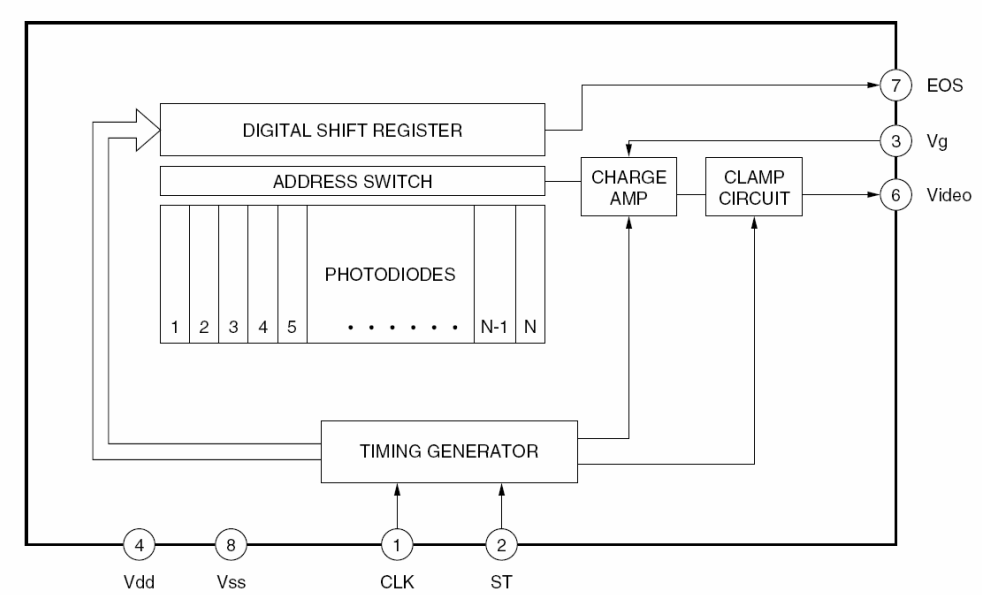


Fig. 2.12 Estructura interna del sensor Hamamatsu S8378-256N

On Vdd és l'alimentació de 5V, i Vss és la massa. Les entrades ST (Start Pulse) i CLK (Clock Pulse) serveixen per activar i començar a operar amb el registre de desplaçament respectivament. L'entrada Vg (Gain Selection Voltatge) ens determina el guany de l'amplificador, que serà baix si seleccionem 5V o alt si seleccionem 0V.

La sortida de Vídeo ens indica el valor obtingut per cada fotodíode, i EOS (End of Scan) ens indica el resultat de l'escanejada de l'últim fotodíode.

A continuació podem veure uns gràfics dels senyals que ens permetran una major comprensió del funcionament del sensor.

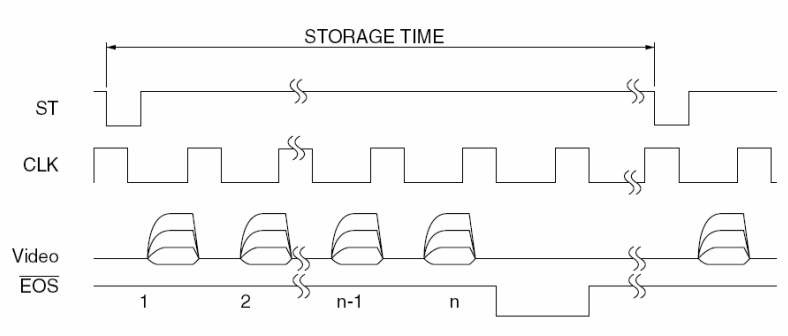


Fig. 2.13 Senyals d'entrada/sortida del sensor

El temps d'emmagatzemament és determinat pels intervals del senyal ST. Cal destacar que un cop ha començat la lectura del primer fotodíode s'ha d'esperar a que es facin 256 lectures abans de tornar a agafar una altre mostra de llum.

2.2.2 Característiques tècniques

Descrivim les especificacions mecàniques del sensor:

Nombre de fotodíodes	256
Amplada del fotodíode	25 um
Llargada del fotodíode	0.5 mm
Encapsulat	24 P in DIP
Llargada de l'encapsulat	31.75 mm
Pes	3 g

Taula. 2.2 Taula de les principals especificacions mecàniques

Podem veure l'encapsulat del sensor

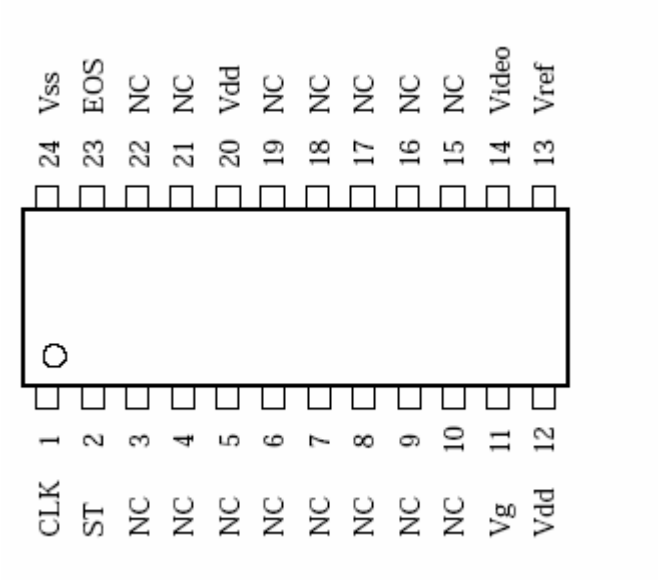


Fig. 2.14 Encapsulat del sensor Hamamatsu S8378-256N/SPL

Les principals característiques elèctriques del sensor són:

Freqüència del senyal CLK	0.1 kHz – 500 kHz
Impedància de sortida	1 KΩ
Consum de potència	25 mW

Taula. 2.3 Característiques elèctriques del sensor

A continuació mostrem la resposta espectral del sensor:

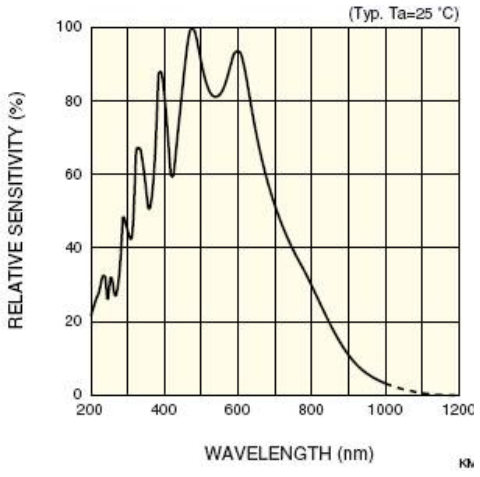


Fig. 2.15 Resposta espectral del sensor

Podem veure que el rang de longituds d'ona en el qual treballa aquest sensor va dels 200 nm fins als 1000 nm. Per tant aquest sensor ens permet veure la zona UV, visible i IR. La resposta del sensor és més sensible a les longituds que fan referència a l'espectre visible (380nm – 800nm).

El sensor té una resposta lineal en quant a la relació existent entre temps d'integració i nivell de lluminositat mesurats. Com ja em comentat, es pot triar el tipus de guany de l'integrat, la qual cosa determina el temps d'exposició de la mostra de llum, i per tant, també el nivell de saturació, per un temps elevat d'exposició. En aquesta aplicació es treballarà amb un guany alt, ja que amb la interfície Matlab, es programarà un autoajust, que modifica automàticament el temps d'exposició per tal que no hi hagi saturació ni baix nivell de lluminositat.

No obstant, per facilitar futures modificacions, deixem preparat el circuit per controlar des del dsPIC aquest guany.

2.2.3 Calibració

Per tal de poder establir una relació específica entre el número del fotodíode i la longitud d'ona que li correspon realitzem una calibració amb uns tubs d'espectre. La calibració es va fer amb fluorescents de gasos d'argó, hidrogen, heli, mercuri, neó i vapor d'aigua. Aquests gasos, tenen una signatura espectral amb uns pics molt localitzats a determinades longituds d'ona, que poden variar en funció del a pressió que té el gas dins de cada tub.

Realitzem la mesura de llum procedent de cada tub, i representem l'espectre, fixant-nos en els pics significatius. Així obtenim una relació directa entre el número del díode i la longitud d'ona corresponent.

El següent pas és fer una regressió polinòmica de tercer ordre amb les dades obtingudes, per tal de calibrar els punts no mesurats.

$$\lambda_P = I + C_1 p + C_2 p^2 + C_3 p^3 \qquad \textbf{(2.1)}$$

On 'λ' és la longitud d'ona del fotodíode p, 'I' és el punt d'intercepció, 'C' és el coeficient d'ordre n, i p és la mostra obtinguda (número de fotodíode)

Realitzem la calibració i obtenim els valors de coeficients que ens determinen la calibració dels punts no mesurats. Obtenim així la següent equació.

$$\lambda_p = 304.757 + 3.201p + 0.00352\,p^2 - 1.517 \cdot 10^{-5}\,p^3 \qquad \textbf{(2.2)}$$

Les estadístiques obtingudes de la regressió han estat prou adequades, ja que el coeficient de correlació proper a 1 ens indica la bona qualitat de la calibració. Em obtingut un valor de coeficient de correlació de 0.9992, i un error típic de 8.17nm.

Amb totes les dades obtingudes podem veure a la següent taula de valors, i al següent gràfic, les diferències entre els valors calculats i els calibrats.

Número de mostra	Longitud d'ona real (nm)	Longitud d'ona teòrica (nm)	Error (nm)
1	366.07	363.442761	2.62723929
2	397	390.119748	6.88025153
3	405.69	403.573594	2.11640571
4	434	434.088645	-0.08864514
5	486	485.544605	0.45539546
6	501	499.36006	1.63994038
7	546.81	540.953268	5.85673185
8	579.34	575.695761	3.64423907
9	587	586.118163	0.88183676
10	614.306	627.737913	-13.4319128
11	633.443	645.020697	-11.5776969
12	656	655.36649	0.6335104
13	668	669.127925	-1.1279253
14	706.519	703.330637	3.18836263
15	725	740.534567	-15.5345667
16	764.29	753.931668	10.3583318
17	812.56	800.157941	12.4020595

Taula. 2.4 Taula de punts de calibració

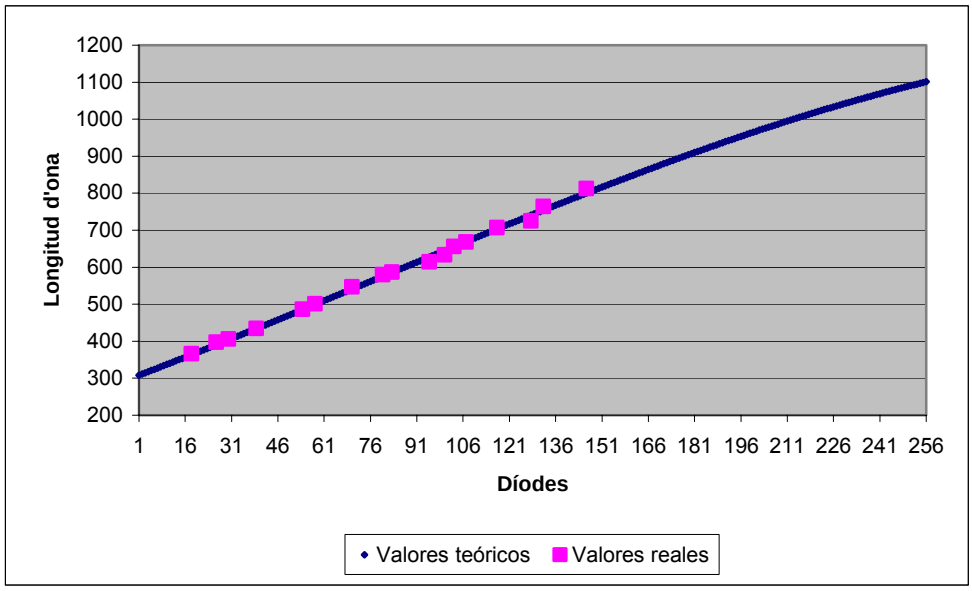


Fig. 2.16 Gràfic de calibració

2.3 Conversor AD 977

Per tal d'augmentar i millorar el sistema final, introduïm un conversor AD extern de 16 bits, la qual cosa ens permet una millor digitalització a la sortida de vídeo del sensor. L'AD escollit és l' AD977 d'Analog Devices.

2.3.1 Característiques generals

L'AD977 és un conversor A/D de 16 bits de baixa potència (màxim 100mW) que opera amb una alimentació unipolar de 5V. Les dades de sortida són tipus sèrie amb comunicació SPI.

La velocitat de sortida es de 100 kSPS

A continuació podem veure un diagrama de blocs que ens permet entendre millor el funcionament de l'AD.

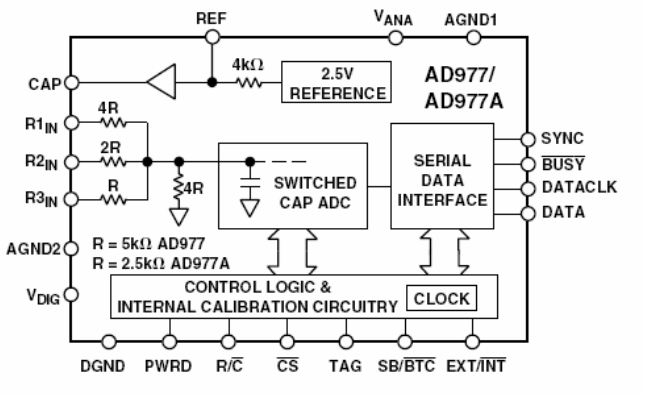


Fig. 2.17 Diagrama de blocs del funcionament intern de l'AD

L'encapsulat escollit és tipus PDIP amb 20 pins

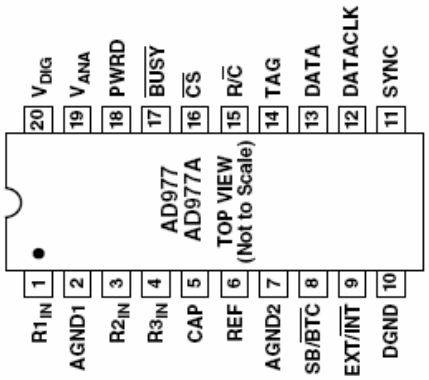


Fig. 2.18 Encapsulat i patillatge de l'AD escollit

A continuació observem el procés bàsic d'adquisició de dades i conversió que realitza el nostre AD, on el temps màxim de conversió és de 8 us

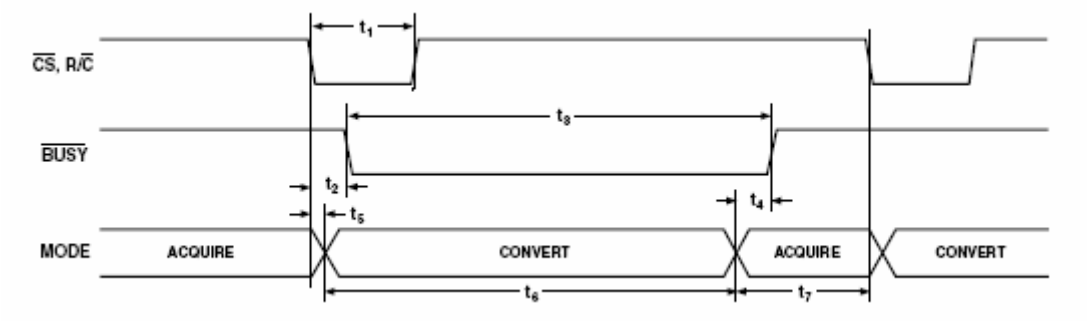


Fig. 2.19 Cronograma del procés de l'AD

On CS (Chip Select), R/C (Read/Convert) són les entrades que ens serveixen per controlar l'instant a partir del qual l'AD comença la conversió.

La sortida Busy, ens indica l'estat en el qual es troba l'AD. Si té nivell baix, vol dir que la conversió ha començat, i es manté en aquest nivell fins que s'hagi completat la conversió i les dades estiguin en el registre de desplaçament. Just en aquest precís instant, el nivell passa a prendre valor de nivell alt.

2.3.2 Configuració

A continuació especifiquem la configuració del nostre AD.

L'entrada digital SB/BTC ens especifica el format de les dades de sortida de l'AD. Si connectem a nivell baix, el format serà complement a 2 (Ca2), i si connectem a nivell alt el format serà binari natural. Pel nostre sistema, farem servir una sortida binària natural.

L'entrada digital EXT/INT ens especifica el funcionament de l'AD mitjançant una senyal de clock extern (nivell alt), o de clock intern (nivell baix). Donat que en el nostre projecte ens interessa controlar aquest senyal amb el dsPIC, treballarem amb un clock extern.

El circuit recomanat pel fabricant sense considerar tensió de offset, sense un ajust del guany i una alimentació unipolar de 0-4V, és el següent.

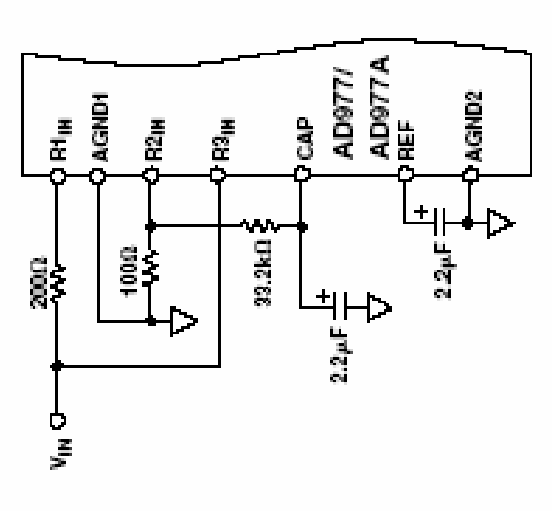


Fig. 2.20 Esquema del muntatge de 0-4 volts de senyal d'entrada

El fet d'escollir aquesta configuració és perquè el senyal que em de digitalitzar es troba dintre d'aquest rang. No obstant, aquest senyal, que anomenem sortida vídeo del sensor té un offset d'1Volt. Això ens implica una pèrdua de ½ bit el qual es tradueix en una pèrdua considerable de nivells. Per tant, la millor solució passa per realitzar una etapa diferencial, amb la qual obtindríem un aprofitament dels 16 bits de l'AD. Aquesta etapa diferencial, tractem el senyal de sortida de vídeo del sensor, considerant l'offset, de manera que a l'entrada de l'AD tenim tota la informació del sensor continguda dintre del marge dinàmic de l'AD (0-4V). Donada aquesta configuració tenim un pas de quantificació de 61,03uV.

Al següent esquema podem veure com interconnectar l'AD amb el microcontrolador amb un estàndard SPI.

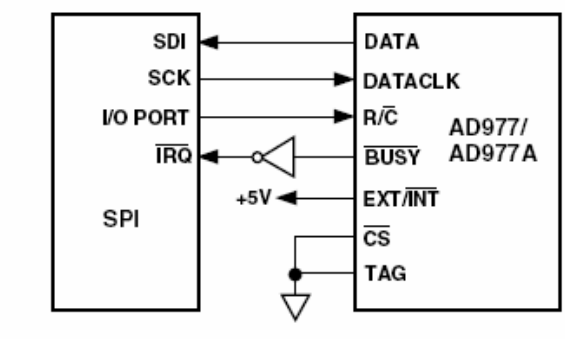


Fig. 2.21 Estàndard SPI amb l'AD977

Per tant, d'aquest esquema podem veure definitivament el comportament temporal de les diferents senyals de l'AD

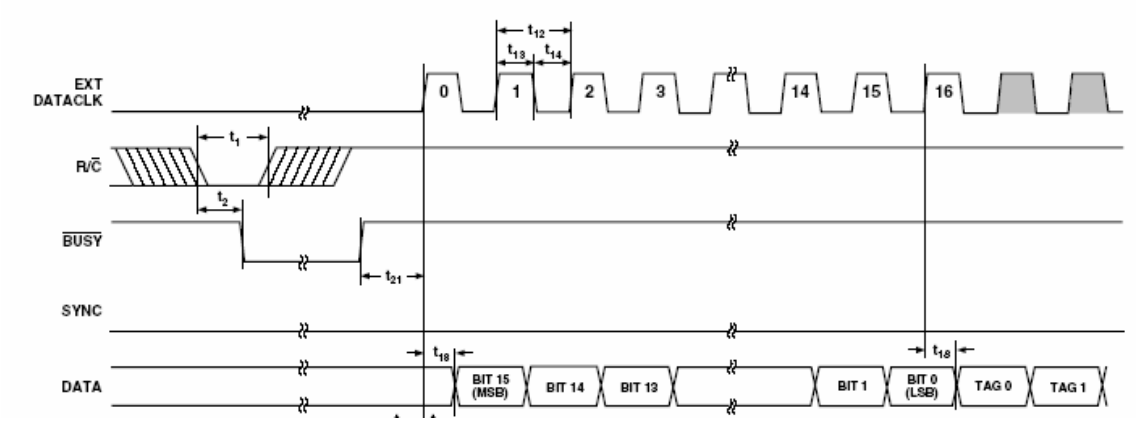


Fig. 2.22 Cronograma de l'AD977

Els principals temps que s'han de considerar a l'hora de programar els senyals de control d'aquest AD són els 8us que triga l'AD en fer la conversió i els 2 us que triga en fer l'adquisició.

2.3.3 Etapa d'acondicionament del senyal d'entrada

Tal i com em explicat anteriorment, tenim el sensor de llum Hamamatsu S8379-256N, com a senyal a digitalitzar, però la sortida del sensor de llum, presenta un marge d'entre 1 i 4,2V, i l'entrada de l'A/D té un rang d'entre 0V i 4V. Per no perdre qualitat, dissenyarem una etapa d'acondicionament que permeti fer aquests reajustaments de tensió i així aprofitar el marge dinàmic de l'AD, i aconseguir una mínima pèrdua de bits.

Per tant, dissenyarem un Amplificador diferencial, que situarem entre el senyal de sortida del sensor i l'entrada de l'AD.

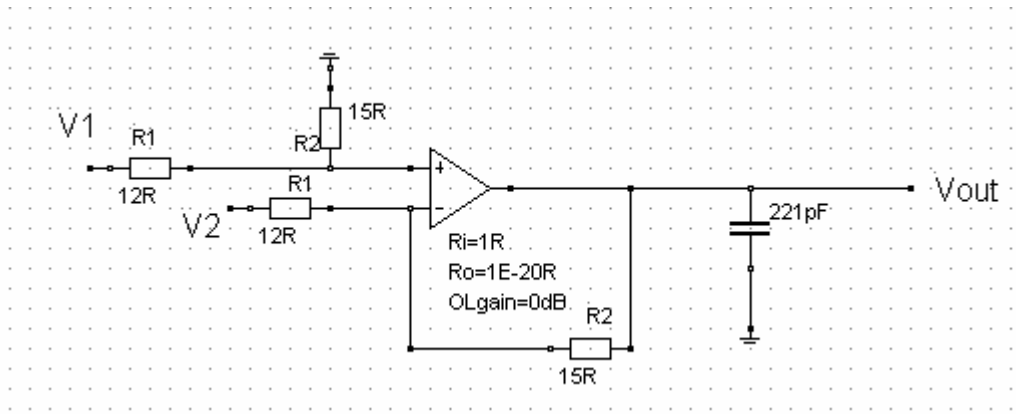


Fig. 2.23 Esquema del muntatge

Si atenem a l'equació del guany, en funció de les resistències i la tensió d'entrada obtenim:

$$V_{out} = \frac{R_2}{R_1}(V_2 - V_1) \quad \textbf{(2.3)}$$

On V2 fa referència al senyal d'offset del sensor, i V1 fa referència a la senyal de sortida de vídeo del sensor.

Observem, per tant que el guany desitjat és el següent:

$$G = \frac{R_2}{R_1} = \frac{V\max sensor}{V\max inAD} = \frac{4,2V}{4V} = 1,25 \quad \textbf{(2.4)}$$

Així, el valor comercial escollit de les resistències son de R1=12KΩ, i R2=15KΩ

Per tant, per implementar aquesta etapa d'acondicionament necessitem els següents components:

- Amplificador operacional TL084
- R1=12KΩ, i R2=15KΩ
- Condensador de 220pF

Aquesta etapa la tenim implementada mitjançant uns connectors a la placa final, que ens permeten tractar directament la sortida del sensor, amb la posterior pèrdua de bits degut a l'offset, o fer el pas de l'etapa d'acondicionament.

2.4 Altres components

2.4.1 MAX232

Per comunicar el dsPIC amb l'ordinador cal adaptar els nivells de tensió a la sortida del port UART del dsPIC ('0' = 0V i '1' = 5V) als nivells de tensió que utilitza el port sèrie del PC ('0' = 12V i '1'= -12V), podent fer servir el protocol RS232.

Utilitzarem l'adaptador de ports MAX232 per fer-ho, donat que es tracta d'un dispositiu alimentat amb 5 V com la resta d'elements de la placa que permet la comunicació sèrie 'full duplex'.

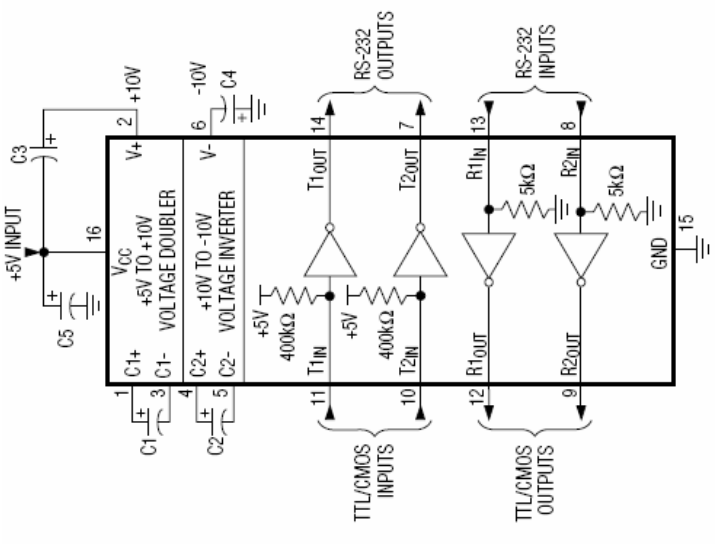


Fig. 2.24 Esquema intern del MAX232

2.4.2 Oscil·lador

Tots els dsPIC necessiten un rellotge per funcionar. Com ja sabem, necessitem diverses operacions per tal que una simple instrucció s'executi. Aquestes operacions han de ser controlades pel rellotge.

Els tipus d'oscil·lador més recomanables a fer servir segons el fabricant del dsPIC (Microchip) són els següents:

- HS (High Speed). Proporcionen oscil·ladors d'alta velocitat. Dissenyats per treballar a freqüències superiors a 4 MHz. Alt consum de corrent. Aquest mode és més utilitzat per ressonadors que per cristalls.
- XT (XTAL). Proporcionen oscil·ladors de velocitat mitja, ja que el rang d'operació és de 1 fins a 4MHz. És el mode estàndard de cristall més utilitzat. El consum de potència és de nivell mitjà.
- LP (Low Power). La característica bàsica d'aquest tipus d'oscil·lador és el baix consum de potència. Les principals freqüències disponibles són de 32 kHz fins a 200KHz. Aquest mode es fa servir quan necessitem oscil·ladors molt exactes i amb un grau elevat d'estabilitat.
- RC (External RC). Aquest oscil·lador es fa servir amb un circuit resistor-capacitor extern. S'obté només una aproximació a la freqüència desitjada, per tant, poca exactitud i molta inestabilitat. Té un baix cost, i un consum de potencia mitjà.

Donades les característiques de consum de potència, estabilitat i rang de freqüències, farem servir un oscil·lador XT de 4MHz, ja que tindrem un consum mitjà de potència, i el cost d'aquest oscil·lador no és elevat. El tipus de rellotge escollit és el HC49 del fabricant CMAC de 4MHz.

A la majoria d'oscil·ladors, per tal que funcionin correctament, necessitem un resistor i dos capacitors. Els valors d'aquests components, depèn de variables com la freqüència d'oscil·lació, l'estabilitat desitjada,...

En el cas que fem servir un cristall, generalment no és necessària la utilització de la resistència.

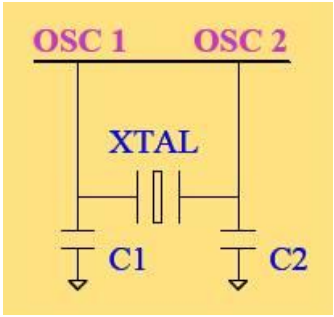


Fig. 2.25 Connexions entre l'oscil·lador i qualsevol microcontrolador

El condensador C2 controla el guany de l'oscil·lació, mentre que el condensador C1 serveix per controlar el desplaçament de fase que es pot produir. Normalment aquests dos valors són idèntics, però depèn de cada disseny. En el nostre cas, farem servir dos capacitors de 15 pF.

2.4.3 Reguladors de tensió

Per tal d'evitar que qualsevol component de la placa pateixi sobretensions, i alimentar correctament tots els components, inclourem un regulador de tensió, ja que l'alimentació del circuit serà a 5V.

El tipus de regulador escollit és el L7805CV de la casa STMicroelectronics, ja que el nivell de sortida és de 5V, i l'encapsulat és del tipus TO-220.

El circuit d'aplicació proposat pel fabricant és el següent:

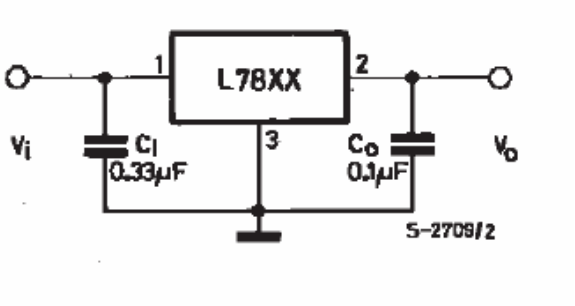


Fig. 2.26 Circuit d'aplicació del L7805CV

El màxim valor de tensió d'entrada que pot regular aquest component és de 35V DC.

2.5 Descripció del disseny PCB

Per tal de desenvolupar el disseny de la nostra placa, farem servir un programa per realitzar un esquemàtic, i després crearem un NETLIST, per tal de poder importar aquest arxiu i generar el layout.

Em seguit aquest procés ja que serà més fàcil fer futures modificacions del projecte, a partir d'un circuit esquemàtic.

El circuit esquemàtic l'hem realitzat amb el programa orcad capture i és el següent:

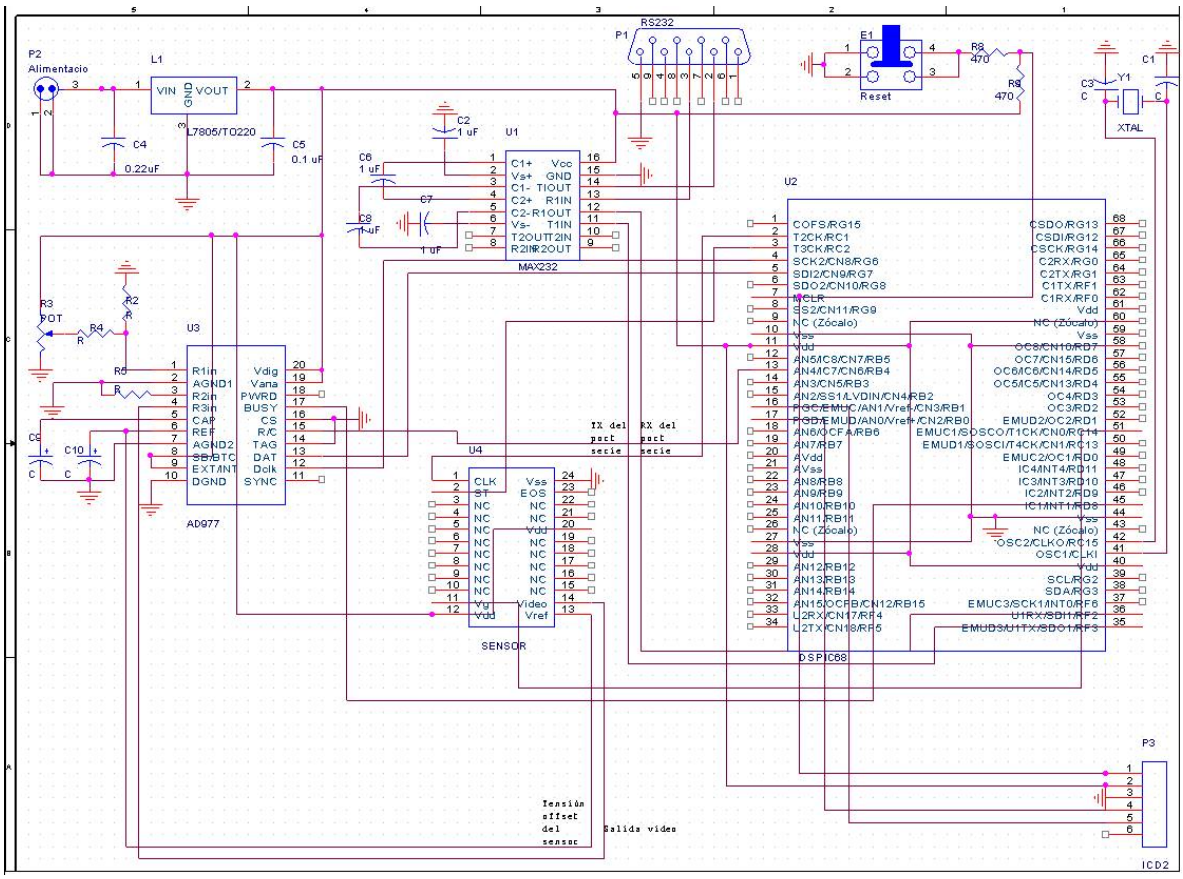


Fig. 2.27 Esquemàtic realitzat amb el programa orcad capture

Com podem veure en la imatge, la placa constarà de:

- Entrades
  - Alimentació (Vcc i massa)
- Sortides
  - Port sèrie per la comunicació RS232 de la placa amb el PC
  - ICD2, per tal de debuggar a temps real
- Components
  - Regulador de tensió
  - MAX232
  - dsPIC
  - Sensor Hamamatsu
  - AD extern
  - Oscil·lador
  - Resistències i condensadors

Un cop acabat aquest circuit, generem un fitxer NETLIST, per tal de poder importar aquest arxiu a un altre programa que ens generi el layout. El programa escollit per aquest procés serà el PCAD, ja que els operaris que realitzaran la placa han obtingut bons resultats amb aquest programa.

L'equip amb el qual es fabricarà el circuit imprès, es tracta d'una fresadora controlada sota un PC (Impressora fresadora LPKF PRotomat c-60). Partint des d'un disseny en P-CAD, aquestes màquines poden fressar pistes, foradant i tallant contorns, sense necessitat de fer servir sistemes químics per realitza la placa, podent fer-la directament des de l'ordinador. Això ens permetrà obtenir una placa de circuit imprès d'una o dues cares, tant analògiques com digitals de producció SMD Fine Pitch amb pistes de fins 0.1mm (4mils) i separacions de 0.1 mm (4mils).

Destaquem que la placa final serà adaptada per introduir-se a una sonda, per tant, tenim unes determinades mesures a complir.

Les dimensions de la placa són 24,5 cm llargada x 3,6 cm amplada. També s'han de tenir altres consideracions, com 1 cm que s'ha de deixar d'espai per els cargols que fixen la placa a l'habitacle de la sonda.
Una altre limitació és la posició del port sèrie i l'alimentació, ja que només tenen una possible ubicació dins de la sonda per tal que els cables puguin connectar-se des de l'exterior. El mateix problema el tenim amb les dimensions específiques del sensor i la posició i llargada del cable de fibra òptica.

Un cop que tenim les connexions definides, i posicionats els components, el següent pas és l'enroutament de les línies. Així, donades les limitacions de distàncies de la fresadora amb la qual es farà la placa, realitzem l'enroutat.

Les principals consideracions a l'hora de fer l'enroutament, són tenint en compte els condensadors de desacoblament, evitant llargues distàncies per tal que la seva funció sigui la correcta. També cal considerar l'enroutat dels components smd, evitar línies prou llargues de tal manera que ens facin d'antena,....

Els passos que realitzarà la fresadora, és realitzar tots els forats que hi hagi al disseny, després es fa una metal·lització d'aquests forats, per tal de tenir continuïtat si la placa està feta a dos cares, i per últim tallar els contorns definint les línies del nostre disseny.

No obstant, abans d'enviar aquest disseny cap a la fresadora, és molt pràctic definir-nos dos plans de massa, un a la capa de dalt, i un altre a la capa de sota. Em de tenir la precaució de connectar-los entre ells, per tal que no apareguin efectes capacitius a la placa final. El sentit d'aquests plans és facilitar el treball de la fresadora, ja que haurà de tallar menys contorns que si no introduíssim aquests plans.

Així obtenim el nostre disseny amb les dimensions determinades.

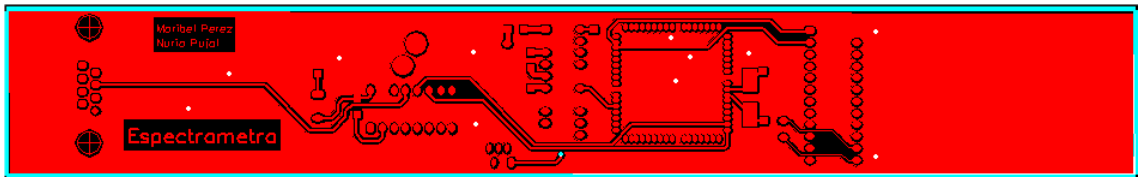


Fig. 2.28 Capa superior de la placa PCB

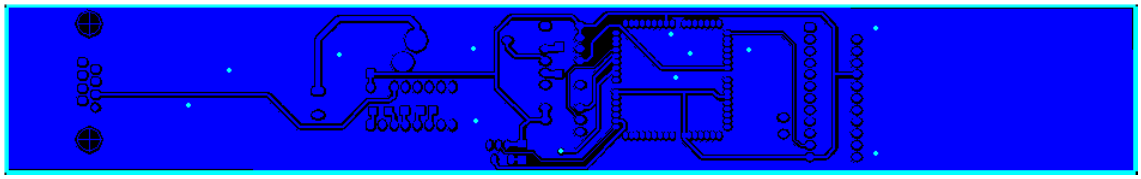


Fig. 2.29 Capa inferior de la placa PCB

Per últim destacar dos consideracions en el disseny de la placa pensades de cara a la realització de les soldadures.

La primera és que a l'hora de dissenyar els PAD's dels elements, em realitzat una forma ovalada, ja que així l'estany es distribueix sol per tot el PAD de coure i és més fàcil de realitzar una bona soldadura que faci bon contacte.

La segona consideració és que a l'hora de soldar cal realitzar amb l'estany forma de carpa de circ (com podem veure a la figura A). Mai s'ha de deixar una soldadura amb forma de bombolla, com la de la figura B, ja que pot ser lo que es diu una soldadura freda o falsa soldadura en la que no hi ha contacte elèctric entre la pota i l'estany perquè ha quedat una pel·lícula de resina que recobreix i aïlla elèctricament la pota de l'estany. Aquest error es produeix o bé perquè la pota no s'havia escalfat prou, de manera que la resina no s'ha volatilitzat o perquè s'ha ficat massa estany.

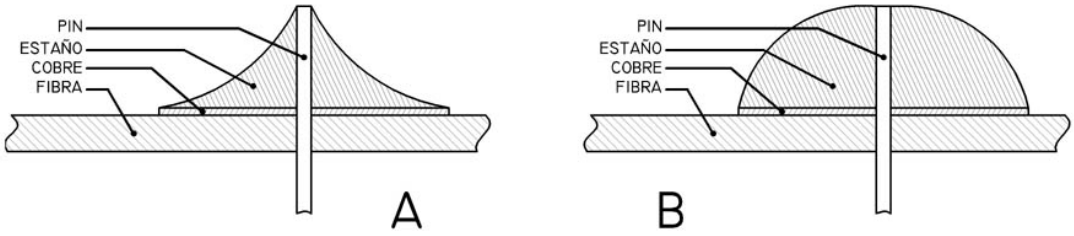


Fig. 2.30 Soldadura correcta (A) i soldadura incorrecta (B)

Si tenim en compte totes aquestes consideracions, no tindrem problemes d'efectes capacitius, discontinuïtat entre les pistes, o d'aïllament elèctric.

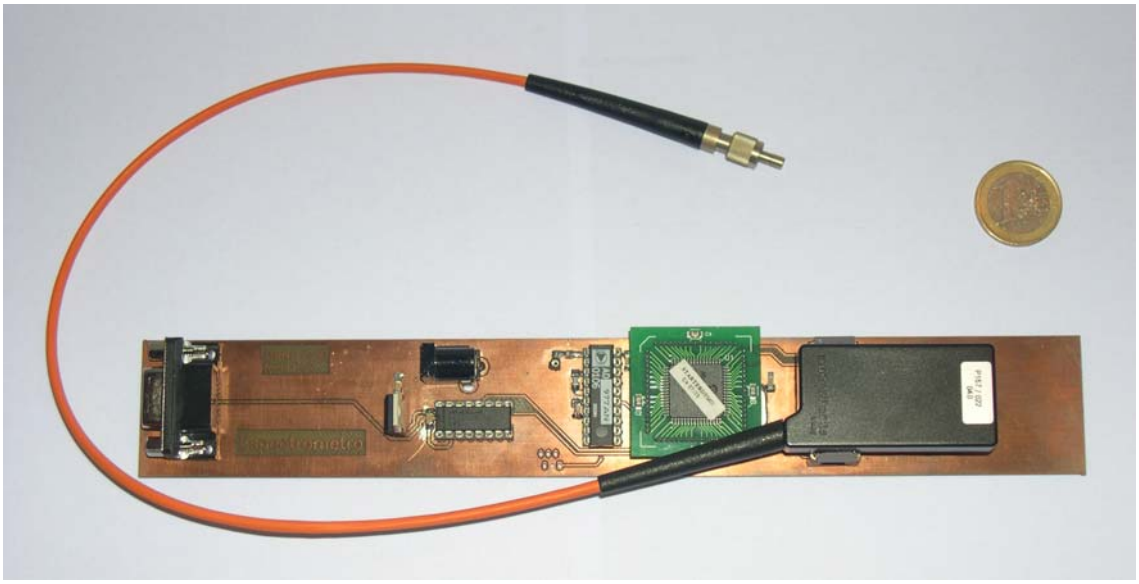


Fig. 2.31 Placa final

CAPÍTOL 3 DESCRIPCIÓ DEL SOFTWARE

3.1 Software utilitzat MPLAB i IAR Workbench Embedded Workbench

Un dels punts importants dins del projecte és la programació del microcontrolador (dsPIC), que és el que realment controlarà l'adquisició de les dades, encara que el seu tractament es faci posteriorment i el control global es faci des d'una interfície gràfica construïda amb llenguatge MATLAB.

Una de les principals avantatges que presenta el dsPIC vers altres microcontroladors és la possibilitat que programar en llenguatge C. Només hem de tenir en compte una sèrie de particularitats, com la declaració dels I/O ports i la utilització d'una sèrie de registres necessaris per la configuració de les comunicacions (SPI, UART, CAN...) que segueix un cert paral·lelisme amb el llenguatge ensamblador.

A l'hora de declarar les senyals de control d'elements externs que no tinguin una comunicació específica tenim els I/O ports que acostumen a utilitzar-se com a 'enables'. Tots aquests ports, pertanyent als diferents banks poden utilitzar-se com a senyals d'entrada o sortida, només s'ha d'especificar en la seva declaració.

```
PORTx=0;    Així accedim a la declaració del port x
TRISx=0xFF3F;    TRISx és un registre de 16 bits equivalent a cada un dels
                 pins del port, que poden tenir valor 1 o 0

                 0 – Declarem com a sortida
                 1 – Declarem com a entrada

0xFF3F           1 1 1 1   1 1 1 1   0 0 1 1   1 1 1 1
                                     ↳┬┘
                                      Bits 6 i 7 declarats com a sortida
```

També necessitem saber el nom dels registres utilitzats per la configuració dels ports utilitzats en les diferents tipus de comunicacions, que apareix en el fitxer *.h que s'ha d'incloure i que es característic per cada microcontrolador. En el nostre cas, hem d'incloure el fitxer 'io30f6012.h'.

```
SPI2CON_MODE16=1;    Comunicació SPI de trames de 16 bits
U1STA=0x0000;    Comunicació UART de 8 bits, sense paritat i 1 bit d'stop
```

Tota la resta segueix la estructura típica d'un programa C , amb la mateixa sintaxis i contempla també la possibilitat de crear fitxers *.h i *.c amb funcions que puguin ser cridades des del fitxer general.

Com editor de codi utilitzem el programa IAR Embedded Workbench, que ens permet depurar el codi i crear un fitxer .cof amb l'equivalència del nostre codi en ensamblador. Encara que nosaltres tinguem la opció d'escriure el codi en C, el microcontrolador realment es programa en aquest llenguatge.

El funcionament del programa IAR Embedded Workbench es força senzill. Un cop obert el programa, creem un nou projecte. Hem de seleccionar *File>New* i seleccionar *Project* i prémer *Acceptar*.

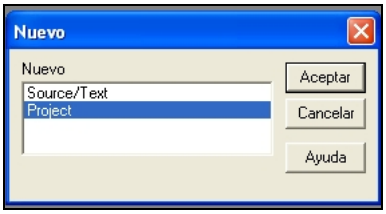


Fig. 3.1 Finestra de IAR per nou projecte

Posteriorment, hem de seleccionar el directori on volem crear el nou projecte. És un bon consell crear un carpeta nova amb tot el projecte, ja que si tenim dos projecte en una mateixa carpeta poden sorgir problemes.

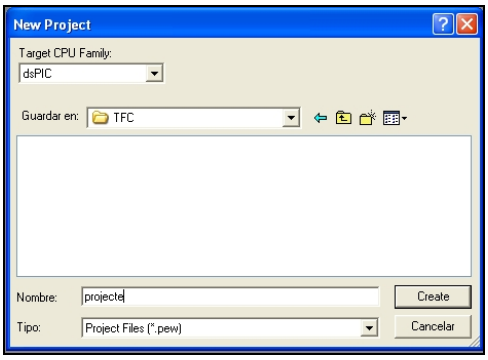


Fig. 3.2 Finestra de IAR per guardar nou projecte

Per incloure els arxius de text (codi) hem de seguir els mateixos passos que els necessaris per crear un projecte, però aquest cop hem de seleccionar la opció *Source/Text* de la finestra *Nuevo*.

Es recomanable posar aquests arxius de text creats dins el mateix directori que el propi projecte. Però tot i haver-los creat en falta agregar-los al projecte. Per tal de fer-ho us heu de posar dins la finestra amb el nom del vostre projecte,

clicar sobre la carpeta *Debug,* clicar amb el botó dret del ratolí sobre la carpeta i seleccionar l'opció *Add Files...*

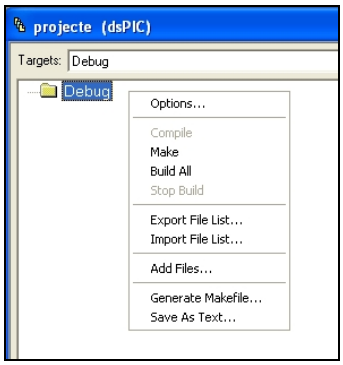


Fig. 3.3 Finestra de IAR per afegir arxius

Quan els haguem agregat la pantalla anterior tindrà un aspecte similar al la imatge que mostrem a continuació, on es poden veure tot els arxius agregat.

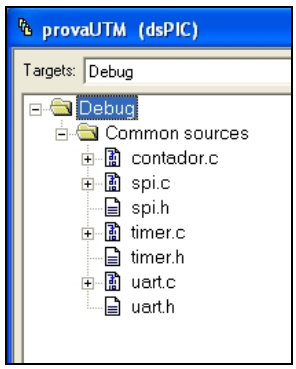


Fig. 3.4 Finestra de IAR amb el projecte creat

Com ja hem dit abans, necessitem crear un arxiu *.cof amb l'equivalència del nostre codi a ensamblador per tal que el programa el pròpiament programa el dsPIC entengui el nostre codi. Per obtenir aquest fitxer només hem seguir els següents passos. Seleccionem *Project>Options* i seleccionem XLINK dins de les opcions de *Category:* Seleccionem la opció *Other* dins de les opcions de *Format* i hem de seleccionar coff dins de la barra desplegable de la opció *Output* format dins de *Format* i finalment clicar el botó *OK*.

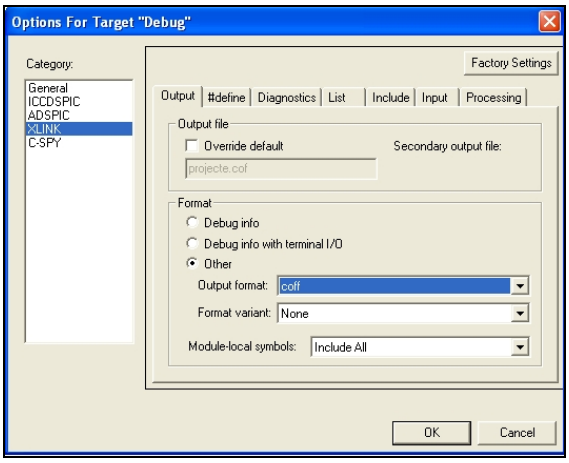


Fig. 3.5 Finestra de IAR per opcions de compilació

Fins al moment encara no s'ha creat l'arxiu .coff, només hem configurat el programa perquè ens el creï.

Un cop tenim tot el programa escrit, obrim el nostre programa principal i cliquem *Project>Compile*. Se'ns obrirà una finestra amb el errors sintàctics que té el codi i ens indicarà en quina línia es troba l'error i els hem d'anar depurant i tornar a *Compile* fins que no aparegui cap error. Un cop passi això podem ja podem clicar a *Project>Buid All*, ara si que crearem l'arxiu *.coff amb el mateix nom del nostre projecte i que quedarà guardat dins la carpeta creada pel nostre projecte dins del directori *Debug>Exe*.

Així doncs, necessitem un altre programa per fer la programació del microcontrolador pròpiament dita, i aquest és el paper del MPLAB IDE v7.10. Obrim el fitxer *.cof des d'aquest programa i hem de seleccionar un debugador (que posteriorment també utilitzarem com a programador). La funció del debugador es programar el microcontrolador però amb la possibilitat de fer un control de les variables i de les accions per verificar el correcte funcionament del codi. Aquesta programació es volàtil i només perdura mentre tinguem alimentat del dsPIC. Un cop tinguem el codi final, haurem de procedir a programar el microcontrolador, on aquest cop la programació serà 'indefinida'.

Per realitzar les accions de debugador i programador utilitzem un MPLAB ICD 2 de la casa Microchip, igual que el nostre dsPIC.

Per dur a terme aquestes accions primer hem d'Importar el nostre arxiu *.coff clicant la opció de menú *File>Import* i seleccionat el directori adequat. També hem de configurar quin es el microcontrolador que volem programar, per això seleccionem la opció del menú principal *Configure>Select Device* i s'obrirà una finestra on haurem de seleccionar el microcontrolador d'una barra desplegable on posa *Device*.

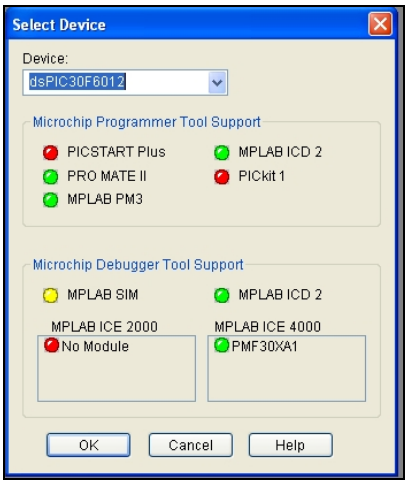


Fig. 3.6 Finestra de MPLAB per seleccionar el microcontrolador a programar

Un altre element que hem de configurar és la configuració del cristall que utilitza el nostre dsPIC. Com ja hem comentat anteriorment la configuració que utilitzem és XTL. Seleccionem *Configure>Configuration Bits...* i a *Category* despleguem el menú de *Primary Oscillator Mode* i seleccionem *XTL*, en el nostre cas.

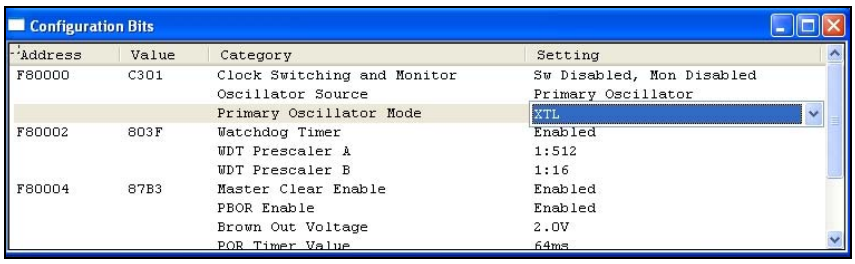


Fig. 3.7 Finestra de MPLAB per configurar l'oscil·lador

Per començar a 'debuggar' , amb el programa de proves, hem d'anar a la opció de menú principal *Debugger>Select Tool* i seleccionar MPLAB ICD 2 (prèviament l'hem hagut d'instal·lar adequadament). Arribat a aquest punt veiem que se'ns activen una sèrie d'icones a la barra de menú. Primer de tot hem de connectar amb l'ICD2 i posteriorment copiar el programa al dsPIC. Pel tal de fer això anem al menú principal *Debbuger>Connect*, veurem que surt una pantalla de diàleg que ens indicarà quan s'ha acabat de connectar. Un cop vegem aquest missatge podem procedir a carregar al programa al dsPIC *Debbuger>Program*. Un cop fet això, utilitzem les següents icones:

Fig. 3.8 Icones de MPLAB de funcionament general del programa

Que ens permetran que el programa s'executi pas a pas, fins a un punt determinat o tot seguit. Un cop premem l'stop podem veure el valor de les diferents variables *View>Watch* i introduint el nom de la variable.

Per programar definitivament el dsPIC hem de seguir els mateixos passos però canviar les opcions del menú *Debugger* per les mateixes pro del menú *Programmer*.

3.2 Software del dsPIC

3.2.1 Estructura general del programa

Abans d'entrar en detalls sobre la programació i el codi, mostrarem un esquema amb la idea general del programa que ha de contenir el dsPIC i posteriorment, en següents apartats, en comentarem els detalls.

Hem de dir que en l'esquema no si veuen reflexats les inicialitzacions de les variables, ni de la configuració de les comunicacions en detall.

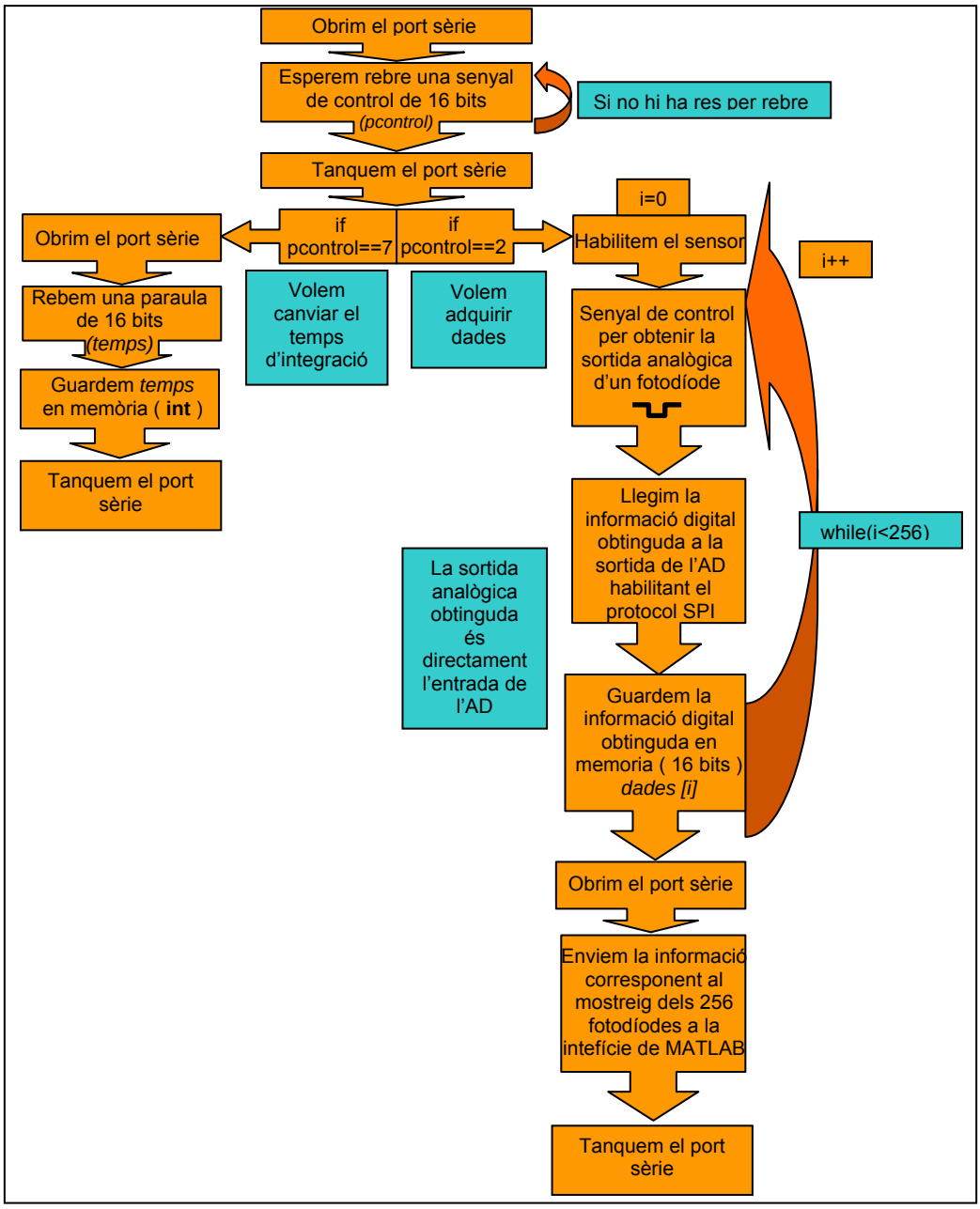


Fig. 3.9 Esquema general del programa carregat al dsPIC

3.2.2 Senyals de control

Tot i que el dsPIC es un híbrid entre un microcontrolador i una dsp, només utilitzarem les possibilitats que ens ofereix com a microcontrolador. Vegem doncs, les senyals que generem des del dsPIC que controlen els diferents perifèrics i les seves principals característiques.

3.2.2.1 Control del sensor

Pel control de l'AD són necessaris tres pins configurats com a sortida.

Nom del PIN	Descripció de la seva funció
RC2	Utilitzat com a senyal de 'enable' En els 'datasheets' apareix com a senyal ST
RC1	Utilitzat com a senyal de rellotge (CLK) Indicarà el número del fotodíode del que esperem rebre informació
RB14	En el data sheet apareix com a senyal Vg S'utilitza per indicar al sensor si ha treballar en guany màxim o mínim (el guany no es ajustable)

Taula. 3.1 Taula dels pins de control del sensor

Com que fem un control molt estricte de la senyal de CLK (sabem en tot moment en quin fotodíode demanem informació) no utilitzem la sortida EOF ('End of signal') que ens proporciona el sensor. Aquesta senyal indica que a acabat l'escombrada dels 256 fotodíodes i hem cregut totalment innecessari tenir en compte aquesta senyal.

En el programa inicialitzem els valors de ST i CLK a '1'. També hem d'inicialitzar el variable Vg a '0', ja que amb aquesta acció seleccionem que el sensor treballi en guany màxim.

Per habilitar el sensor només hem de posar ST a '0' abans de posar del primer temps de rellotge a '0' i tornar-lo al valor inicial abans de tornar CLK al valor inicial, en cas contrari el sensor no realitzarà la lectura.

Així doncs podem fer el control de la senyal de rellotge amb un simple bucle for de 255 iteracions, tenint en compte que s'ha de fer la primera iteració apart (fora del bucle) per la particularitat del 'enable'.

La duració del temps baix de la senyal de rellotge ens indica el temps d'integració de cada fotodíode, que equival al seu temps d'exposició a la llum incident. En aquest instant de temps hem de fer la lectura de l'amplitud tenint en compte que es faci durant la segona meitat de la seva duració, on en principi, haurà assolit el seu màxim.

```
RC2=1;        Inicialitzem la senyal de 'enable'
RC1=1;        Inicialitzem la senyal de rellotge

RC2=0;
retard(1);
RC1=0;
retard(1);
RC2=1;
retard(temps);
RC1=1;        Realització del primer temps baix de la senyal de rellotge,
              juntament amb la senyal de 'enable'

 for(i=0;i<255;i++)
{
        RC1=1;
        retard(1);
        RC1=0;
        retard(1);
        retard(temps);
        * Instant òptim de la realització de la lectura
        retard(1);
}                              Realització de les 255 senyals de rellotge posteriors

RB15=1;
```

3.2.2.2 Control de l'AD extern

El control de l'AD extern és fins i tot més simple que el del propi sensor, ja que consta d'una sola senyal de control configurada com a sortida del dsPIC i el control d'una senyal de sortida de l'AD, anomenada BUSY, que configurarem com a entrada del dsPIC

Nom del PIN	Descripció de la seva funció
RB4	Actua com a 'enable' de l'AD Indicarà el moment ha de ser convertida la senyal d'entrada (nosaltres haurem de fer que l'instant de mostreig sigui l'idoni)
RD8	Entrada provinent de l'AD anomenada BUSY Ens indica el temps en que està fent la conversió a digital de l'entrada

Taula. 3.2 Taula dels pins de control de l'AD extern

Configurem RB4 perquè per defecte, sempre es trobi a ‘1’ i només el posem a ‘0’ en el moment idoni de la lectura. El temps que dura aquest instant baix no cal que sigui molt gran (de l’ordre de us) però si que es imprescindible que durant la seva durada la senyal analògica sigui lo més estable possible.

Un cop l’AD ens indica que ha finalitzat la conversió (la senyal de BUSY ha tornat al seu valor original , a ‘1’) podem habilitar la comunicació SPI i guardar els 16 bits de la lectura en memòria.

3.2.2.3 Cronograma de l’adquisició

Per fer-nos una idea més clara del funcionament cronològic de l’adquisició i la variació del valor de les diferents senyals de control presentem a continuació el cronograma de l’adquisició, on podem veure en color negre les senyals que hem de generar nosaltres des del dsPIC i en verd, les que obtenim dels diferents perifèrics.

La senyal de VIDEO, és la senyal analògica obtinguda a la sortida del sensor i que serà l’entrada directa de l’AD. La senyal MOSTRES, també analògica, és el valor exacte que agafa l’AD com a senyal de fer-ne la conversió a digital. Finalment, trobem la senyal BUSY, que és una sortida proporcionada per l’AD que ens indica el moment en que ha acabat la conversió analògica (internament) i per tant és l’indicatiu del moment en que podem enviar les dades obtingudes a través del protocol SPI cap al dsPIC.

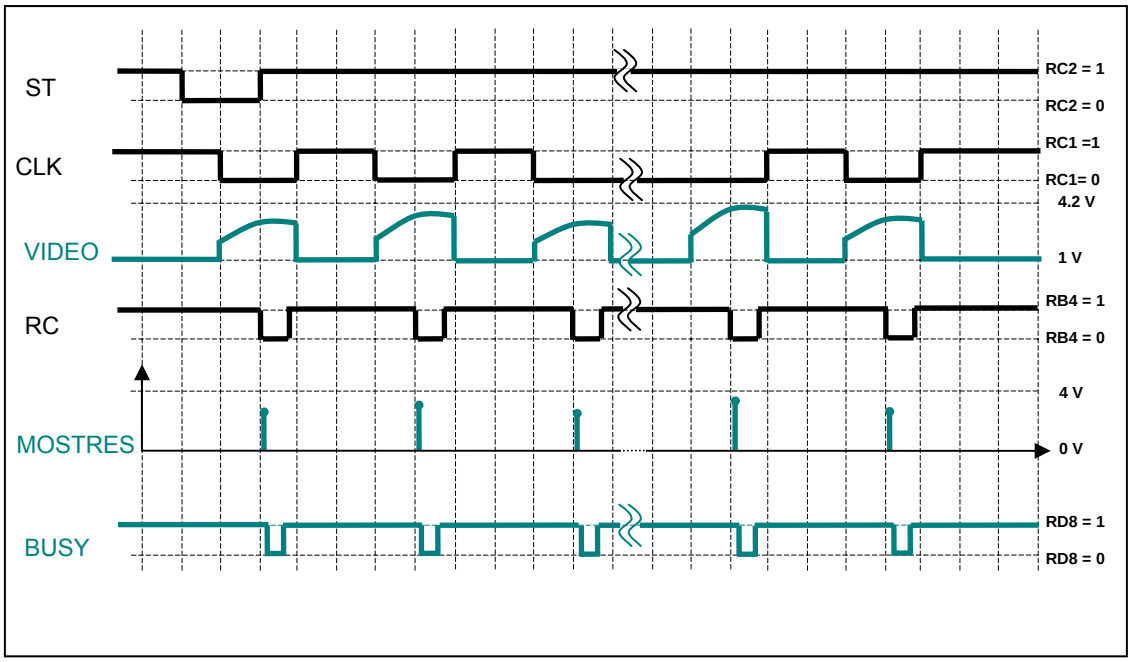


Fig. 3.10 Cronograma de l’etapa d’adquisició

3.2.3 Protocol SPI

El protocol SPI és l'utilitzat per enviar les dades que anem obtenint de l'AD cap al dsPIC i així poden ser emmagatzemades en memòria. L'AD utilitzat és l'idoni per realitzar aquest tipus de comunicació.

Com ja hem vist anteriorment, aquest protocol permet diferents configuracions, que nosaltres hem de triar la més idònia, a partir de la configuració d'una sèrie de registres interns del dsPIC a través del nostre codi.

En el dsPIC comptem amb dos ports SPI, nosaltres configurarem el SPI2, ja que creiem que SPI1 comparteix ports amb altres comunicacions que es podrien utilitzar en un futur.

Vegem amb un quadre les diferents configuracions dels registres que pertanyen a la comunicació SPI i la seva respectiva funció.

Instrucció	Efecte
SPI2CON_FRMEN=0;	Deshabilitem la opció de 'Framed SPI' Es una opció utilitzada pel sincronisme de trama quan es realitza la comunicació entre dos microcontroladors, en el nostre cas no és així
SPI2CON_SSEN=0;	Deshabilitem la sortida SS, ja que no l'utilitzem És la sortida utilitzada com a sincronisme de trama
SPI2CON_MODE16=1;	Configurem perquè la transmissió sigui de 16 bits D'aquesta manera podem guardar les dades rebudes com a **int** , sense necessitat de cap altre tractament
SPI2CON_MSTEN=1;	Configurem el dsPIC com a 'master' de la comunicació SPI Serà el dsPIC l'únic capaç d'iniciar i parar la comunicació SPI, com és lògic
SPI2CON_DISSDO=1;	Configurem el dsPIC només com a receptor Només hem de rebre dades des del AD, no n'hem d'enviar
SPI2CON_SMP=1; SPI2CON_CKP=0; SPI2CON_CKE=0;	Configuració de la forma d'ona del rellotge de dades i del moment idoni en que el dsPIC ha de fer la lectura de la informació digital

SPI2CON_SPRE0=1;
SPI2CON_SPRE1=1;
SPI2CON_SPRE2=1;//*1
 	
SPI2CON_PPRE0=0;
SPI2CON_PPRE1=1;//*16
	Configuració de la velocitat de la transmissió
	Partim de la velocitat de 1 MHz (en MIPS), característic del cristall utilitzat, per realitzar-ne un pre-escalat
	
	SPI rate = 1MHz/1*16 = 62.5 kHz.

Taula. 3.3 Instruccions per configuració SPI

Vegem doncs l'esquema de la comunicació SPI, segons la configuració anterior.

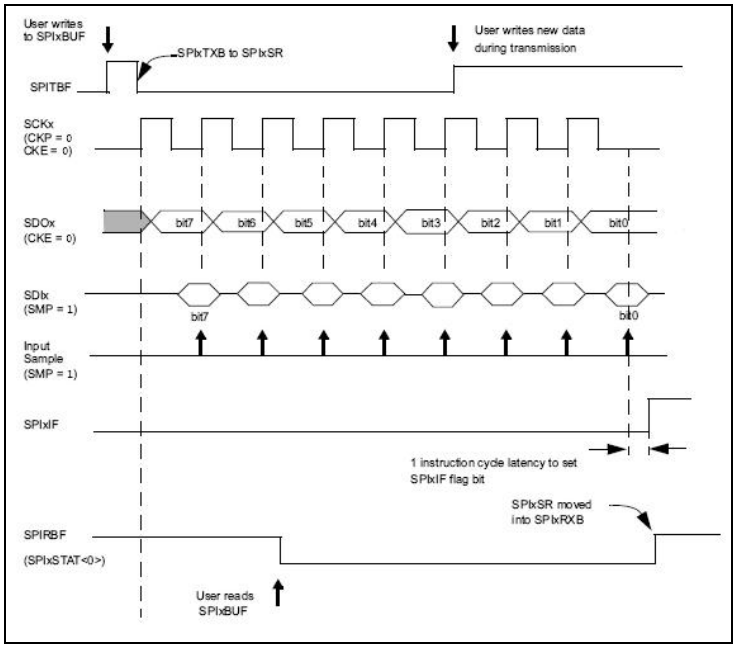


Fig. 3.11 Senyals de comunicació SPI

Cal remarcar que en la nostra configuració tenim 16 rellotges seguits i no 8 com apareix en el dibuix anterior i que anem llegint a la vegada que anem enviat, així que no hi ha solapament d'aquestes tasques. Només ens serveix per veure la forma del rellotge i de l'instant ens que es fa la lectura de les dades. Aquesta elecció ve donada per les característiques del sensor, ja que aquest clock es el que indica que pot anar enviant les dades digitals i seran llegides correctament.

Ja sabem que tots els pins del dsPIC poden tenir més d'una funció i comunicació associada. Així doncs, els pins que pertanyen a la comunicació SPI no en són una excepció.

Per tal que actuïn com a port SPI hem de realitzat la següent configuració:

SPI2STAT_SPIEN=1;

Si pel contrari, volem que actuïn com a I/O port hem de realitzar la següent configuració:

SPI2STAT_SPIEN=0;

Nosaltres no caldria que ho tinguéssim en compte, ja que disposem de ports suficients per les accions a realitzar. Però tot i així només configurem els ports SPI quan els hem d'utilitzar.

El protocol SPI té la característica que s'inicia la comunicació tant bon punt escrivim alguna cosa en el 'buffer' de i també espera rebre una informació de la mateixa mida de la enviada. Així doncs la transmissió i la recepció es realitza simultàniament.

En el nostre cas no hem de fer cap transmissió cap a l'AD, ni tant sols tenim el pin corresponent connectat enlloc, així que per iniciar la transmissió escrivim una paraula qualsevol (que no anirà enlloc) i esperem rebre'n una de la mateixa longitud a nostre buffer d'entrada.

while(RD8==0);	Mentre BUSY valgui 0 no podem realitzar la transmissió
SPI2BUF=0x0000;	Aquesta acció inicia la transmissió
while(SPI2IF==0);	Quan s'ha acabat la transmissió de 16 bits SPI2IF val 1
dades[i]=SPI2BUF;	Copiem les dades del buffer d'entrada en un vector de 256 elements (**dades**)
	El codi anterior s'ha de realitzar 256 vegades, fins a omplir tot el vector **dades**, per obtenir una 'escombrada' completa dels fotodíodes del sensor.

3.2.4 Protocol UART

El protocol UART és l'utilitzat per enviar les dades emmagatzemades en la memòria del dsPIC corresponent al valor digital de les mostres adquirides de la senyal analògica provinent dels 256 fotodíodes.

Com en el cas del protocol SPI, el protocol UART també ofereix diferents configuracions. Vegem en forma de quadre aquesta configuració:

Instrucció	Efecte
U1STA=0x0000;	Utilitzarem la configuració UART que existeix per defecte Consisteix en trames de 8 bits, sense paritat i 1 bit de stop
U1BRG=(FREQ/(16.0*BPS)); BPS 2400 Velocitat en bps FREQ 1000000 Freqüència en MIPS	Inicialitzem el registre U1BRG que anirà decrementant actuant com un timer Definim així el temps que ha de transcórrer entre la lectura d'un bit i el següent i en el nostre cas serà de 2400 bits per segon.
U1STA_UTXEN=1;	Habilitem la transmissió

Taula. 3.4 Instruccions per configuració UART

Després d'aquests passos només ens queda copiar lo que ens interessa enviar al registre U1TXREG o llegir el que ens arriba del registre U1RXREG.

Ja sabem que tots els pin's del dsPIC poden tenir més d'una funció i comunicació associada. Així doncs, els pin's que pertanyen a la comunicació UART actuen com els pertanyent a la comunicació SPI.
Per tal que actuïn com a port UART hem de realitzat la següent configuració:

U1MODE_UARTEN=1;

Si pel contrari, volem que actuïn com a I/O port hem de realitzar la següent configuració:

U1MODE_UARTEN=0;

En el protocol UART només podem enviar trames de 8 bits, així que hem de fer un truncament de les dades per obtenir a cada dada de 16 bits, dues de 8 bits. Aquest truncament es realitza en el moment de ser enviades, així doncs aquestes noves 512 dades no es guardaran en memòria.

```
void TxUART16(int bytes) {
U1TXREG=bytes;   Enviem la informació de menys pes

bytes>>=8;
U1TXREG=bytes;   Enviem la informació de més pes
return;
}
```

3.3 Software del PC

3.3.1 Funcionament del programa

A nivell d'usuari el funcionament del programa realitzat en llenguatge Matlab és molt senzill. Primer de tot hem d'obrir qualsevol versió de Matlab (nosaltres hem utilitzat la 6.5.1 per realitzar-lo) i escriure a *Command Window* la comanda següent i prémer la tecla ENTER del teclat.

>>projecte('inici')

Veurem que sobre una finestra amb el següent aspecte:

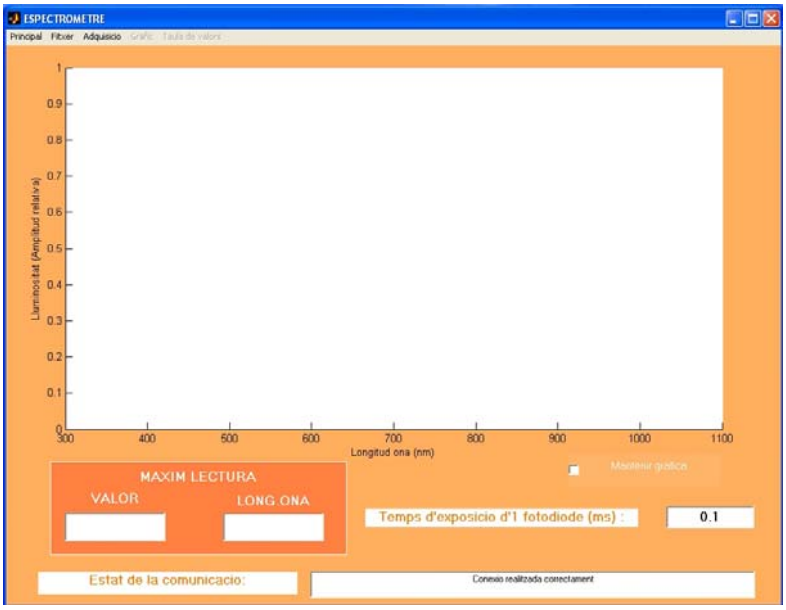


Fig. 3.12 Finestra del programa principal fet amb Matlab

La interactuació amb l'usuari és bàsicament la de les opcions e menú de la part superior de la finestra, on es van activant o desactivant les opcions segons les accions que es poden dur a terme.

Fig. 3.13 Finestra del menú del programa principal

Per iniciar el programa hem de seleccionar del menú principal *Principal>Connectar* i veurem que s'habiliten algunes opcions de menú que es trobaven deshabilitades. A partir d'aquest moment podem realitzar diferents accions:

- Adquirir dades: *Adquirir>Adquirir dades*
  Fem una adquisició simple dels 256 fotodíodes.

  Un cop realitzem aquesta acció queden habilitades altres opcions relacionades:

  - Mostrar els valors de la adquisició en forma de taula: *Taula de valors*
    Apareixen els valors de la captura, relacionant amplitud relativa i la longitud d'ona associada.

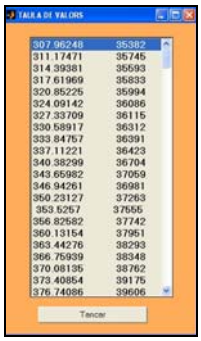


Fig. 3.14 Finestra de taula de dades del programa principal

  - Guardar les dades obtingudes: *Fitxer>Guardar fitxer*
    Guarda un fitxer *.dat en el mateix directori on es troba el programa en format ASCII i que només es pot obrir a través del programa.

  - Canviar el temps d'integració: Adquisició>Canviar temps d'integració
    Podem canviar el temps d'integració (temps en que cada fotodíode està exposat a la llum incident) per tal d'ajustar la senyal obtinguda a les nostres necessitats. Hem d'introduir el valor desitjat en una finestra, tenint em compte uns límits ja indicats.

- Adquisició continua: *Adquirir>Adquisició continua>Iniciar*
  Iniciem una adquisició continua de dades on es va regulant automàticament el temps d'integració. L'adquisició no es deté fins que no seleccionem *Adquirir>Adquisició continua>Parar*.

- Obrir un fitxer ja existent: *Fitxer>Obrir fitxer*
  Prèviament hem hagut de guardar un arxiu d'una captura amb el programa per després poder-la obrir.

- Tancar el programa: Principal>Desconnectar i posteriorment Principal>Tancar.

És la manera de finalitzar el programa.

3.3.2 Esquema general de la interfície gràfica

Com ja hem comentat abans, la interfície gràfica s'ha realitzat utilitzant Matlab. L'estructura bàsica de la interfície es basa en un *figure* general (.fig) en el que tenim diferents elements que li pertanyen i al ser premuts o seleccionats criden a una funció (.m) anomenada 'projecte'. Els elements que criden a aquesta funció són els apartats que pertanyen al menú superior del *figure*.

A continuació es mostra un esquema amb la jerarquia que segueixen els diferents elements del *figure* principal, així com des de quins elements podem interactuar amb els altres.

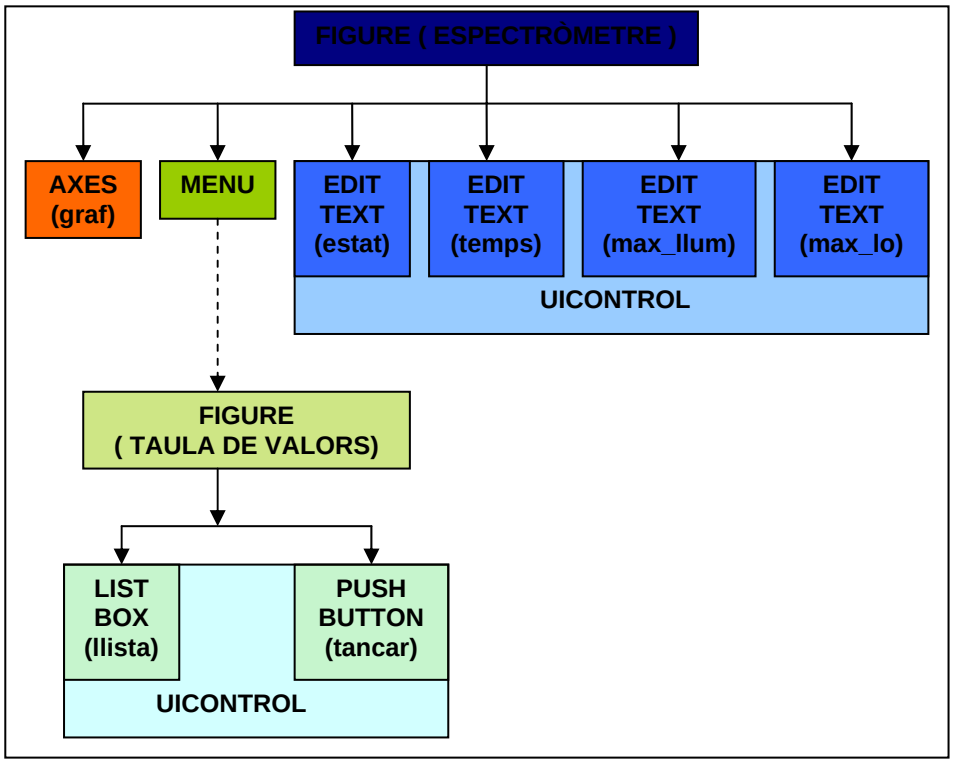


Fig. 3.15 Esquema de la interfície gràfica

La interfície gràfica està bàsicament governada pel menú principal. Aquest menú, segons l'opció seleccionada, realitza les diferents accions.

Veiem un esquema desglossat del menú principal i les funcions que crida en cada apartat .

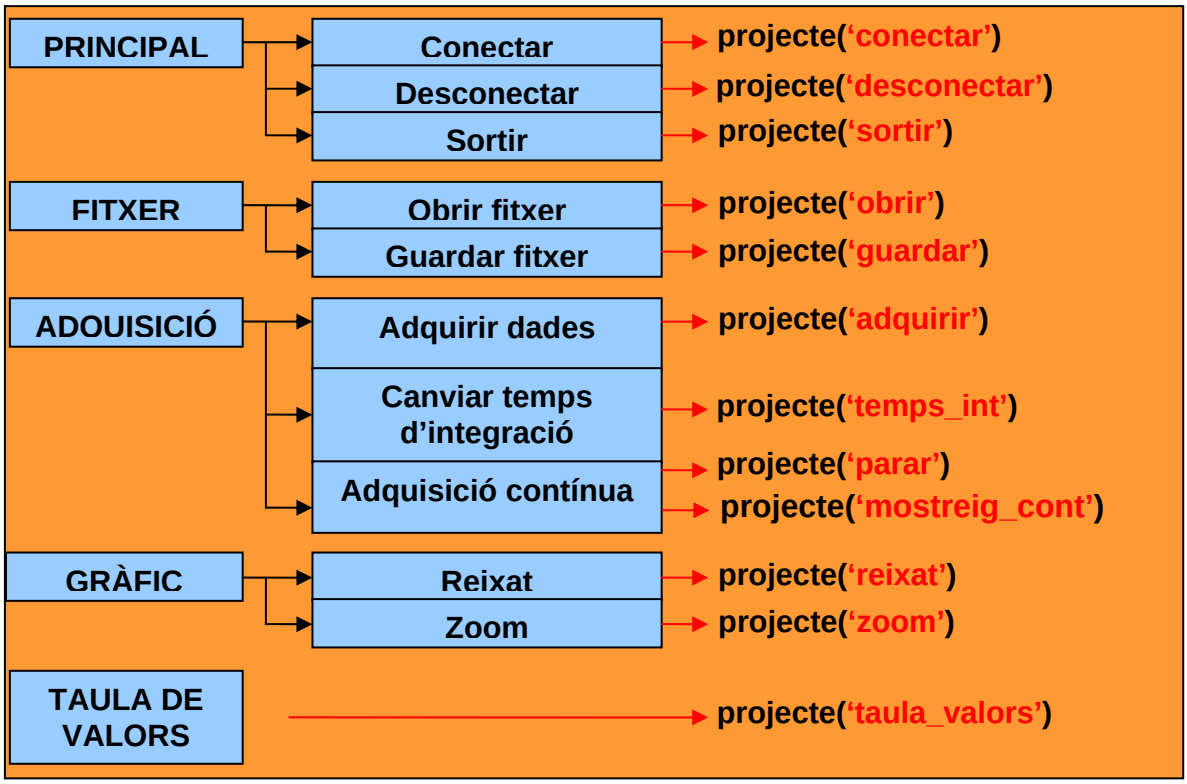


Fig. 3.16 Esquema del programa final fet amb Matlab

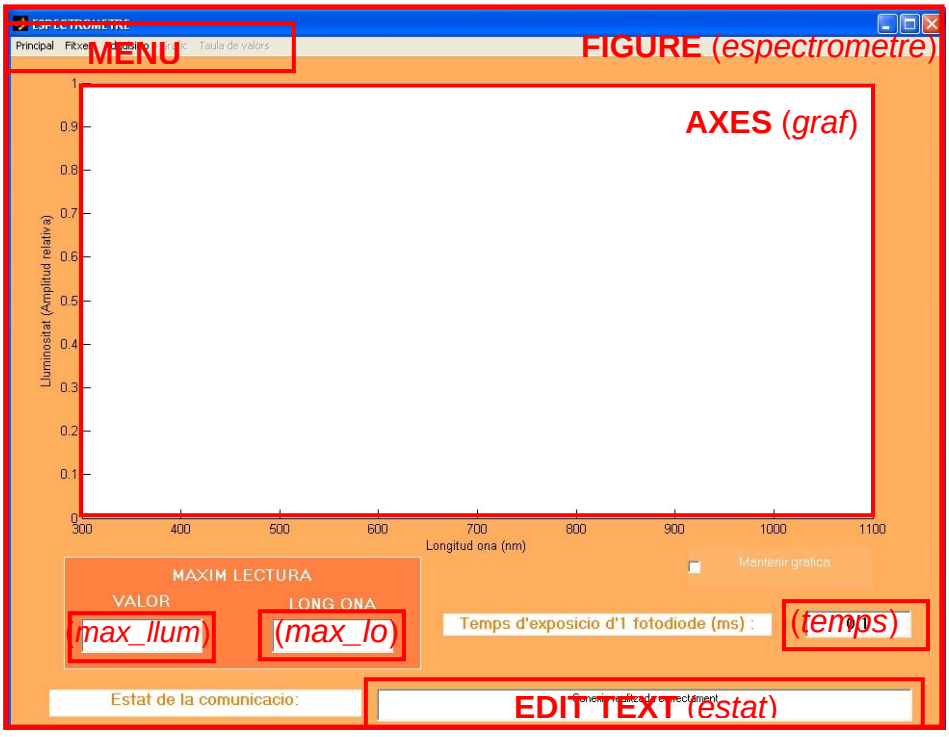


Fig. 3.17 Esquema de la interfície gràfica feta amb Matlab

Podem dir que tot el programa està governat per una sola funció anomenada 'projecte' i només hem de cridar la funció **projecte**('inici') des de la consola de Matlab perquè es posi en marxa la interfície gràfica.

3.3.3 Principals funcions

Així doncs, durant l'execució del programa es va cridant sempre la mateixa funció però amb un paràmetre diferent ,en forma de cadena de caràcters, ja que la declaració de la funció és la següent :

```
function projecte(funcio)
```

Vegem amb detall quines accions duu a terme el programa pels diferents valors de 'funcio'.

projecte('inici')
Com ja hem dit abans, aquesta és la funció principal i de fet és la única crida 'visible' per l'usuari.

- Obre el *figure* principal
- Configura els marges de *axes* (representem longituds d'ona de 300 – 1100 nm)
- Configura el port sèrie ('COM1') : BaudRate=2400 i InputBufferSize=512
- Inicialitzem el temps d'integració amb el valor mínim

projecte ('conectar')

Habilitem les opcions de menú que es poden dur a terme a partir d'aquest moment (adquisició de dades, desconnectar de la placa i adquirir dades).

projecte('adquirir')

Aquesta és la funció encarregada de comunicar-se amb la placa per tal d'adquirir les dades captades pel sensor, tractar-les i representar-les per pantalla.

- Obrim la comunicació amb el port sèrie
- Enviem un byte de control (pcontrol= 2) pel port sèrie al microcontrolador per indicar que volem adquirir
- Esperem a rebre 512 bytes (mida de la lectura dels 256 fotodíodes)
- Tanquem el port sèrie.
- Realitzem el tractament de les dades
- Representem els resultats obtinguts per pantalla
- Busquem el valor màxim, l'indiquem per pantalla i mostrem el seu valor relacionant longitud d'ona i amplitud relativa.

El tractament de les dades consisteix a passar de mostres de 8 bits (mida dels paquets de la comunicació sèrie) per mostres de 16 bits que formen els 256

valors corresponents a cada fotodíode. En definitiva, hem de tornar als valors originals que ens dona l'AD a la seva sortida.

```
j=1;
for i=1:2:512,
        dades(j)=rebut(i)+(rebut(i+1)*256);
        j=j+1;
end
```

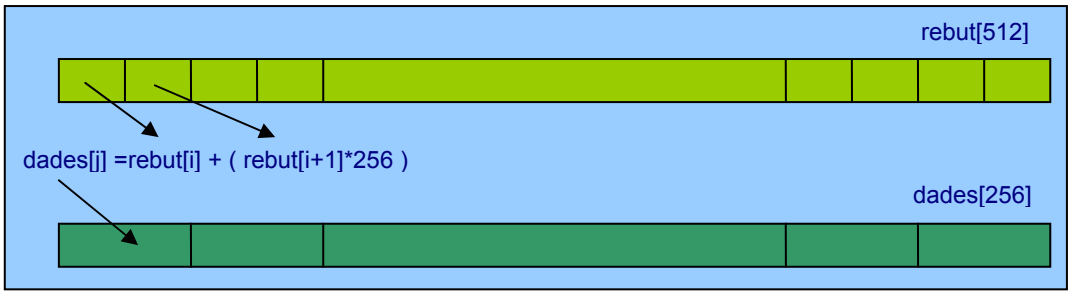


Fig. 3.17 Procés de passar de 16 bits de dades fins 8 bits

projecte('zoom') i projecte('reixat')

Aquests dos projectes, fan servir funcions bàsiques de Matlab, per tal d'ampliar la captura, o visualitzar les adquisicions obtingudes amb els eixos corresponents.

projecte('desconnectar')

Tornem a enviar el temps d'integració mínim, per tal que quedi emmagatzemat al dsPIC i quan el tornem a fer servir aquest tingui en memòria guardat el valor inicial de temps d'integració, i tanquem el port sèrie.

projecte('guardar')

Quan realitzem una adquisició simple tenim la opció de guardar les dades obtingudes per recuperar-les posteriorment.

- Obre una pantalla per tal d'indicar el nom i el directori on volem guardar les dades.
- Passem les dades obtingudes a codi ASCII
- Ho guardem en format *.dat

projecte('obrir')

Ens permet visualitzar la informació guardada corresponent a un arxiu *.dat prèviament guardat pel mateix programa.

projecte('sortir')

Simplement consisteix en tancar la interfície gràfica i esborrar totes les variables utilitzades durant l'execució del programa.

projecte('temps_int')

Variant el temps d'integració podem fer que la lectura sigui més clara i s'aprofiti millor el marge dinàmic. Així doncs, si el valor màxim obtingut a la lectura es molt baix, podem augmentar manualment el temps d'integració (temps d'exposició de cada fotodíode) per obtenir una amplitud major. De la mateixa manera, podem disminuir el temps d'integració quan apareguin moltes mostres saturades.

- S'introdueix el valor en ms del temps d'integració desitjat en un quadre de diàleg.
- Passa el valor de cadena de caràcters a valor numèric.
- Realitza la conversió de milisegons a cicles d'instrucció del microcontrolador.
- Fa un truncament del número de cicles d'instrucció necessaris i ho converteix en 2 valors de 8 bits, equivalents als bits de major i menor pes
- Envia una paraula de control (pcontrol=7) al microcontrolador per indicar que es vol canviar el temps d'integració.
- Envia els dos paquets corresponents al temps d'integració pel port sèrie.
- Actualitza el nou temps d'integració per pantalla

projecte('taula de valors')

Després d'haver fet una adquisició simple podem veure els valors que configuren la gràfica.

- Es guarden els valors que configuren la gràfica en un arxiu temporal de dades (*'temporal.dat'*).
- S'obre una nova finestra (*'taula.fig'*) on apareixen la relació d'amplituds amb el valor corresponent d'amplitud.
- Al prémer el botó *'Tancar'* la finestra es tanca i l'arxiu temporal es eliminat.

projecte ('tanca_taula')

Aquesta funció es cridava el prémer el botó *'Tancar'*. Llavors la finestra *'taula.fig'* es tanca i l'arxiu temporal creat per la funció anterior s'elimina.

projecte ('mostreig_cont')

En el mostreig continu realitzem les mateixes accions que en la funció projecte('adquirir') però amb la peculiaritat que després de cada mostreig calculem el temps d'integració òptim per la pròxima mostra.

Aquesta funció inicialitza a 1 una variable global anomenada connectar que ens servirà per detenir el mostreig continu, ja que el programa segueix mostrejant indefinidament mentre aquesta variable tingui com a valor 1.

projecte('parar')

Es varia el valor de la variable connectar inicialitzada per la funció anterior i això permet detenir la funció.

projecte('mantenir')

Aquesta funció conserva en pantalla la gràfica anterior, per tal de poder realitzar comparacions entre les diferents adquisicions que fem. Si es selecciona s'activa aquesta opció, i si no, manté el mostreig fet a temps real

3.3.3.1 Descripció general de funcionament

A nivell d'usuari el programa té un funcionament totalment transparent. Només cal escriure des de la consola de matlab la següent comanda:

projecte('inici')

Posteriorment s'obrirà una finestra amb la interfície gràfica del programa, on a partir del menú podrem accedir a diferents opcions, però prèviament haurem de prémer la opció 'connectar' del menú principal.

Podem triar entre dos tipus d'adquisició, l'adquisició simple i l'adquisició contínua. Un cop premem la opció d'adquisició simple només realitzem un sola adquisició i la representem per pantalla, tenint en compte que per defecte el temps d'integració és el més petit possible (el seu valor apareix en un requadre de la interfície gràfica), però que es pot variar amb l'opció 'Canviar temps d'integració' pel valor que entrem per teclat, dintre d'uns marges establerts. L'altre opció es escollir una adquisició contínua on el programa va adquirint mostres indefinidament fins que seleccionem la opció 'Parar' i on ell mateix fa una autoregulació del temps d'integració a partir de la informació rebuda fins el moment i fa una estimació de la informació futura.

També existeixen altres opcions menys important, com la possibilitat de guardar una lectura i poder-la recuperar posteriorment, fer un zoom de la imatge obtinguda i afegir-hi un reixat. També tenim la opció de mostrar per pantalla la informació representada gràficament en forma de taula numèrica on es relacionen les diferents longituds d'ona amb el valor d'amplitud corresponent.

Per finalitzar el programa, simplement hem de seleccionar la opció 'Desconnectar' i després la de 'Sortir'. Veiem que la opció de sortir no està habilitada si no hem seleccionat la opció de 'Desconnectar' prèviament. Això s'ha fet perquè no es tanqui el programa deixant el port sèrie 'obert', el que

faria que si volem tornar iniciar el programa apareixeria un missatge d'error per pantalla i hauríem de reiniciar el programa per tornar-lo a utilitzar.

3.3.3.2 Ajust del temps d'integració

L'algoritme utilitzat es basa en l'algoritme PID adaptat a les característiques del nostre projecte.

L'algoritme PID per a temps continu queda descrit per la següent fórmula, on els paràmetres Kp, Ti i Td caracteritzen l'evolució dels valors de sortida u(t) en funció de l'error obtingut en el moment t i tenint en compte també l'error acumulat. Variant els paràmetres podem variar l'estabilitat del sistema, i per cada cas s'hauran d'ajustar pels valors més adequats, segons la rapidesa en que la informació varia i la sensibilitat del sistema.

$$u(t) = Kp\left[e(t) + \frac{1}{Ti}\int^{t} e(t)dt + Td\,\frac{de(t)}{dt} \right] \qquad \textbf{(3.1)}$$

En el nostre cas, aquesta fórmula perd el significat, ja que nosaltres treballarem en temps discrets (en mostres). Així que una possible traducció de la fórmula anterior aplicada al nostre sistema seria la següent:

$$u[n] = \underbrace{Kp(e[n])}_{\text{Proporcional}} + \underbrace{Ki\left(\sum_{1}^{n} e[n]\right)}_{\text{Integral}} + \underbrace{Kd(e[n] - e[n-1])}_{\text{Derivat}} \qquad \textbf{(3.2)}$$

Pretenem anar regulant el temps d'integració per aconseguir un temps òptim que ens permeti aprofitar al màxim el marge dinàmic. En el nostre cas posem com a valor límit 60.000 (el valor màxim representable es de 65.535) i aquest valor serà el que el programa ha d'aconseguir que sempre sigui el màxim evitant que es saturi. Així doncs, el valor de l'error (e[n]) serà la diferència entre el màxim absolut obtingut de la lectura i el límit marcat per nosaltres i que el seu valor ha d'anar tendint a 0.

L'evolució teòrica de l'amplitud relativa en funció del temps d'integració desitjat segueix la següent gràfica.

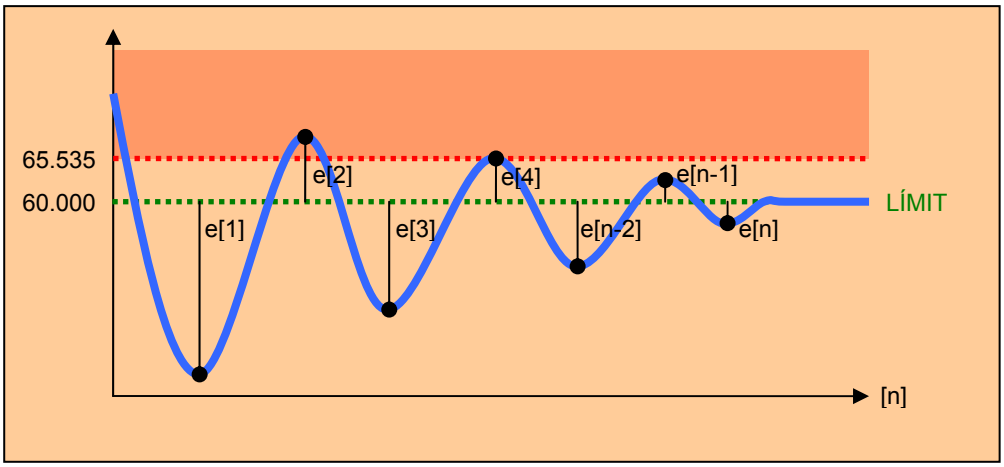


Fig. 3.18 Funcionament de l'algoritme PID

Hem de tenir en compte que l'algoritme PID pur és realment fàcil d'implementar quan es té caracteritzat el sistema al llarg del temps i la seva aplicació passa per implementar un filtre a l'entrada dels sistema. Això no es dóna en el nostre cas, ja que l'únic que hem pogut caracteritzar és la proporció entre el temps d'integració i el valor màxim de la mostra equivalent al terme proporcional. Per tal d'aconseguir un millor funcionament s'ha introduït un nou terme corresponent al temps anterior i la resta de termes s'han anat regulant a mesura que s'han anat fent proves i realment es calcula mitjançant PID la diferència que s'ha de sumar o restar al temps anterior. Cal remarcar que és tracta d'una primera aproximació de coeficients i és millorable.

3.3.3.3 Càlcul de màxims

Si llegim atentament el funcionament de l'algoritme d'ajust automàtic del temps d'integració veiem que contem amb un problema, la pèrdua d'informació del màxim real quan apareixen valors saturats, ja que el màxim valor del qual podem obtenir mostres correspon a 65.535.

Quan l'algoritme doni un temps d'integració per sobre de l'òptim obtindrem un cert nombre de mostres saturades, així que per tal que l'algoritme anterior doni una resposta lo millor possible haurem de fer una estimació del màxim 'real' a partir de la forma de les dades obtingudes.

S'han provat diferents mètodes, però el més efectiu ha resultat ser el més senzill. La variança no resulta una informació rellevant quan obtenim més d'un pic, ja que conjunt de dades amb la mateixa variança poden tenir valors molt dispars segons el nombre de pics.

Així doncs hem utilitzat un mètode de triangles semblants. A partir de calcular el nombre de sensors que tenen un valor superior a dos marges podem configurar un triangle, com es mostra a continuació.

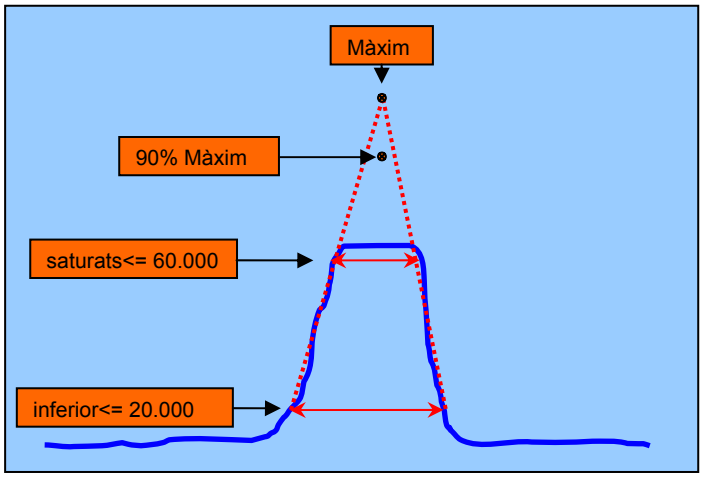


Fig. 3.19 Càlcul de màxim a partir dels nivells establerts

Per tal d'obtenir el màxim calculem el número de fotodíodes amb un valor major o igual a 20.000 i també els majors o iguals a 60.000. Així dos per trigonometria podem relacionar les bases dels dos triangles amb la seva alçada en funció del valor de màxim obtenint així la següent expressió:

$$\frac{màx - 20000}{màx - 60000} = \frac{n^o \inf}{n^o\, sat} \qquad \textbf{(3.3)}$$

Com hem pogut comprovar, el màxim en situacions de no saturació no apareix mai en forma de punxa, sinó més arrodonit, així el valor que agafarem finalment com a màxim és el 90% del màxim que apareix en l'expressió i tenint em compte que volem el valor del màxim en relació a 0 (en valor absolut). Deixant així com expressió final la següent :

$$\max = 18000 \bullet \left[\frac{\left(3 \bullet n^o \inf - n^o\, sat\right)}{\left(n^o \inf - n^o\, sat\right)} \right] \qquad \textbf{(3.4)}$$

CAPÍTOL 4 CONCLUSIONS

4.1 Aspectes mediambientals

A continuació analitzarem els diferents aspectes mediambientals relacionats amb el projecte. Cal recordar, no obstant, que la principal finalitat d'aquest sistema és l'estudi del medi, per tant el disseny del sistema, també s'ha construït respectant el medi ambient.

Un fet a destacar és que en la realització de la placa, tot i ser un prototip ajustat a les mides específiques del disseny final, em tingut en compte la introducció de components auxiliars, com els connectors per implementar l'etapa d'acondicionament del senyal del sensor. D'aquesta manera, no és necessària la implementació d'una nova placa per incorporar o provar aquesta etapa.

A l'hora d'escollir els components, s'ha tingut en compte, baix consum, i que tots els components estiguin alimentats amb una única font.

En quant a la construcció, s'ha intentat fer servir sòcols, per tal que es puguin reciclar posteriorment els components, així es poden compartir recursos amb altres projectes. Per tal de minimitzar el nombre de plaques construïdes en la realització i modificació en el disseny final, es va comprovar el correcte funcionament en una protoboard i en una placa de proves del dsPIC fabricada per Microchip.

Si parlem de l'ús de la placa, l'impacte mediambiental, serà el de la sonda on el circuit va integrat. No obstant, no emet residus, ni radiacions importants a alta freqüència

4.2 Situació i futures millores

Tractarem tant els objectius complerts, com les possibles millores a realitzar del projecte.

Tenint en compte els objectius del projecte esmentats a l'apartat 1.4, podem dir que s'han assolit satisfactòriament.

A continuació explicarem les futures millores.

La principal millora a destacar és canviar la interfície gràfica, perquè està feta amb Matlab, i aquesta alenteix tot el sistema de representació gràfica. Matlab tampoc permet un control d'errors prou fiable, per tant aconsellem canviar-la.

Alhora que canviem la interfície gràfica, podem canviar el protocol de comunicació entre la placa i el PC, ja que l'utilitzat actualment (RS-232) s'ha fet

servir, donat que és un prototip, i és mes senzill establir la comunicació entre ordinador i placa. En el disseny final, aquest protocol es canviarà per un RS-422 o RS-485, que tal i com ja em comentat anteriorment ens permet una correcta comunicació amb distàncies de cable de fins 1200m. Aquest canvi implica la modificació del MAX232 per un altre adaptador (MAX422 o MAX485)

Un altre pas és la integració de més sensors de fondària, temperatura i pressió per caracteritzar completament la columna d'aigua sobre la qual s'està fent l'estudi amb la sonda final. Tots aquests sensors són controlables des d'un sol dsPIC, per això s'ha tingut en compte deixar accessibles els ports I2C i CAN lliures per tal de reutilitzar el codi ja fet. Si volem obtenir més velocitat de les tasques realitzades pel dsPIC per tal de controlar els diferents sensors, es pot canviar l'oscil·lador fins a un màxim de 40MHz.

També destaquem com a millora la integració dins la mateixa placa de l'etapa d'acondicionament, ja que actualment està comprovat el correcte funcionament amb una protoboard. No obstant, el següent pas per integrar-ho és buscar una etapa d'acondicionament nova, per tal que l'alimentació sigui unipolar i no sigui necessària cap altre font d'alimentació.

Com a altre millora possible tenim l'algoritme PID de control automàtic de guany del sensor, ja que l'estudi realitzat en aquest projecte és una primera aproximació, i el càlcul d'aquests coeficients és millorable amb un estudi del comportament més exhaustiu.

A continuació comentarem els aspectes i conclusions finals del projecte.

En quant a la programació del dsPIC, destaquem que programar en C és molt més intuïtiu, donat el nivell de coneixement d'aquest llenguatge que ja teníem, no obstant, l'entorn és bastant nou, i existeixen poques guies o manuals entenedors sobre aquests compiladors. Per tant, ens em vist obligades a utilitzar suport tècnic via e-mail de Microchip, degut a l'absència de coneixements per poder importar els programes del software IAR Embedded a MPLAB.

També dir que al passar el compilador de codi C a ensamblador, no és suficientment depurat en quant a eficiència que si ho programéssim directament en ensamblador, però les possibilitats i aplicacions que ens proporciona programar en C són molt més àmplies, ja que ara tenim capacitat per programar qualsevol dsPIC, tenint en compte les especificacions de cada registre dins de cada família de dsPIC.

Una part molt important del projecte ha estat el coneixement del software necessari per crear el projecte. Tots els programes emprats han estat molt diferents entre si: MPLAB; IAR Embedded, Matlab, PCAD i Orcad Capture. El fet de que tots 5 programes eren pràcticament desconegut pels dos membres del grup, i degut al temps de realització del projecte, ha suposat l'especialització de cadascun dels membres del grup. Per tant, un membre de l'equip s'ha especialitzat més en la part Hardware, i un altre en la part Software,

però el coneixement del projecte final assolit per ambdues parts és pràcticament el mateix.

Per últim destacar que em pogut anar a laboratoris especialitzats, amb equips de soldadura professionals, amb un bon assessorament per part de l'equip tècnic, la qual cosa ens ha permès a la part final del projecte refer i millorar el sistema final.

BIBLIOGRAFIA

[1] Microchip Technology Inc. [en linea]. Pàgina web, URL <http://www.microchip.com>

[2] Applications [en linea]. Pàgina web URL <http://www.oceanoptics.com/applications.asp> [Consulta agost 2005]

[3] Brian Niece - Spectral Images [en linea]. Pàgina web URL <http://www.assumption.edu/users/bniece/CHE131/LineSpectra/Index.html> [Consulta març 2005]

[4] IAR Systems [en linea]. Pàgina web URL < http://www.iar.com> [Consulta febrer 2005]

[5] The Mathworks – MATLAB – The lenguaje of Technical Computing [en linea]. Pàgina web URL < http://www.mathworks.com> [Consulta març 2005]

[6] Manuel Merchán i Sergi Pons (2004).Director Gabriel Montoso i Jaume Piera. *Disseny d'un sistema de mesures espectroradiomètriques*Treball final de carrera,Departament de Teoria de Senyals i de comunicacions, Universitat Politècnica de Catalunya

[7] P-CAD 2004 (2004) *P-CAD 2004. PCB Design.* Docuent en Acrobat [.pdf]. P-CAD_2004_PCB_Users_Guide

[8] Microchip (2003).*dsPICDEM Starter Demo Board User's Guide*. Document en Acrobat [.pdf] Ds51425A

[9] Microchip (2004).*dsPIC30F Family Reference Manual*. Document en Acrobat [.pdf] Ds70046C

[10] Microchip (2004).*dsPIC30F6012 Datasheet*. Document en Acrobat [.pdf] Ds70117E

[11] Microchip (2004).*dsPIC30F6012 Datasheet*. Document en Acrobat [.pdf] Ds70117E

[12] Maxim Integrated Products (2003). *MAX232* [En linia]. Document en Acrobat [.pdf]. 19-4323.

[13] Analog Devices (2000). *AD977* [En linia]. Document en Acrobat [.pdf]. 57120676AD977_A_d

[14] STMicroelectronics (2004). *L7805CV* [En linia]. Document en Acrobat [.pdf]. 2143

[15] Hamamatsu Photonics (2004). *S8378-256* [En linia]. Document en Acrobat [.pdf]. KMPD1066E03.

[16] *Digital Control Applications with the TMS320 Family*. Selected Applications Notes. Texas Instrument. USA 1991

ANNEX

A.1 Captures

A continuació mostrem algunes captures de pantalla obtingudes amb el sistema final.

Primer de tot observem les signatures espectrals obtingudes dels tubs fluorescents emprats per la calibració del sensor.

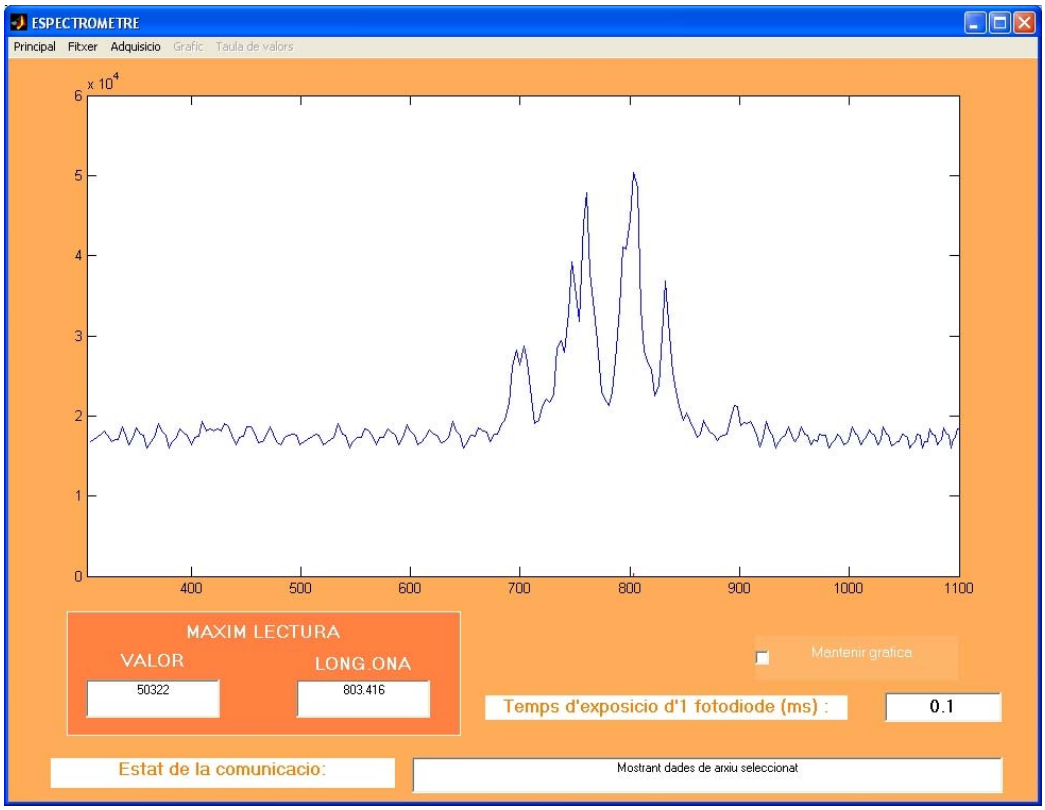


Fig A.1 Fluorescent d'argó

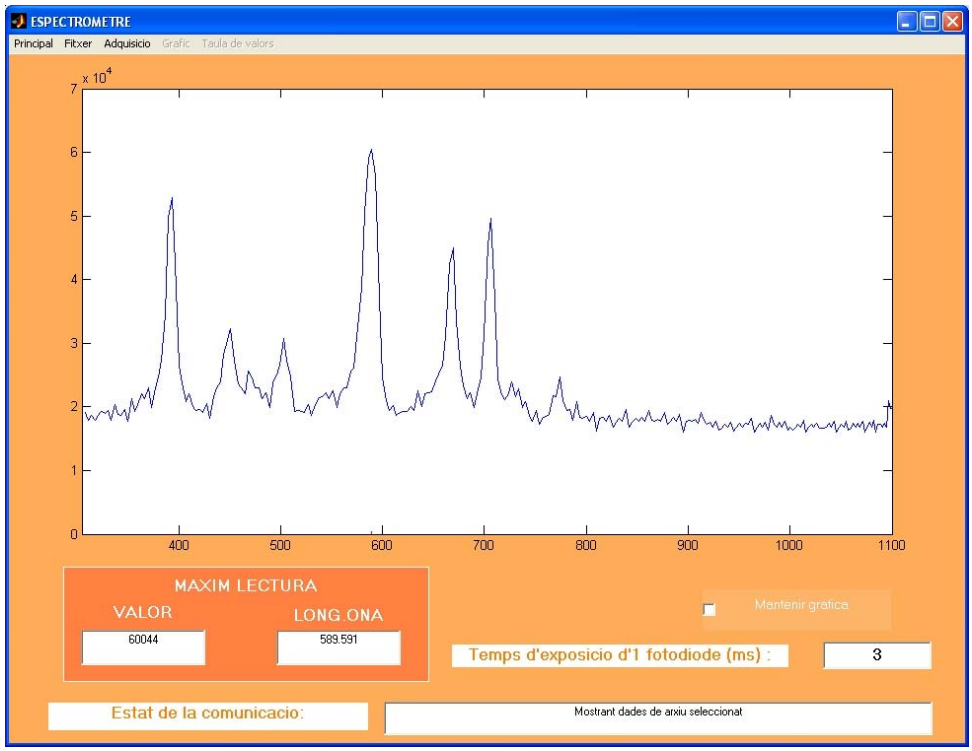


Fig A.2 Fluorescent d'heli

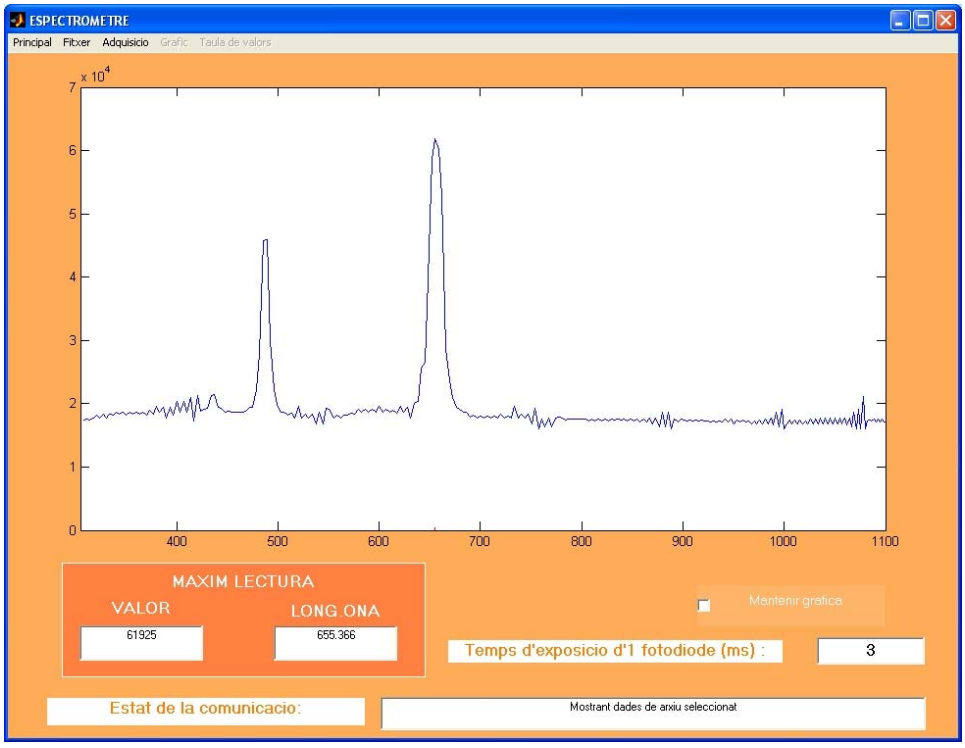


Fig A.3 Fluorescent d'hidrogen

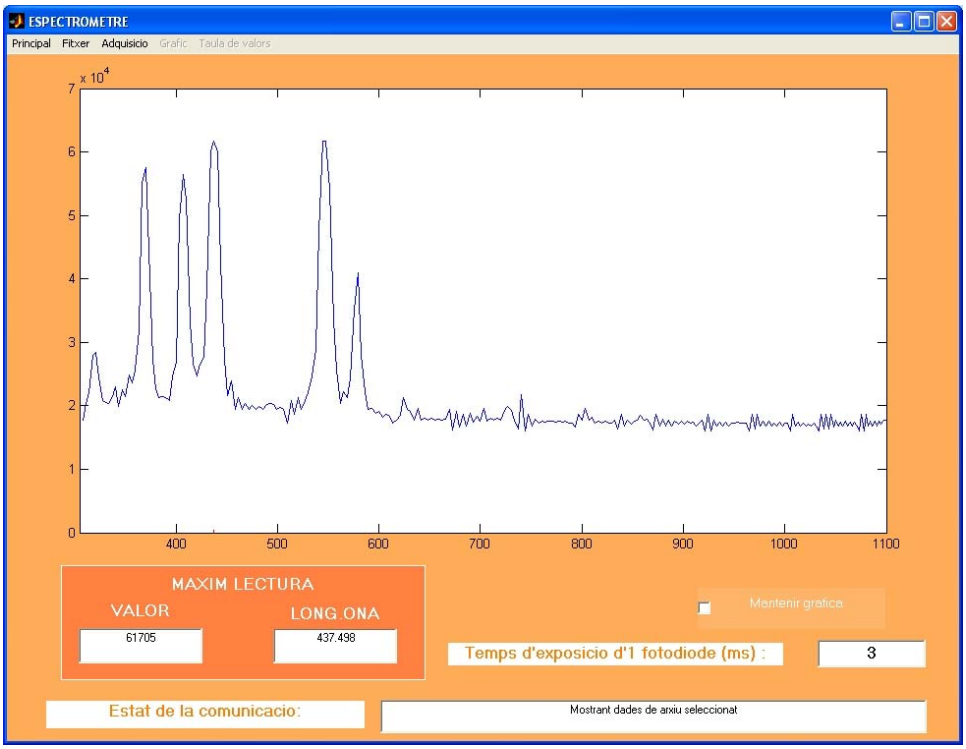


Fig A.4 Fluorescent de mercuri

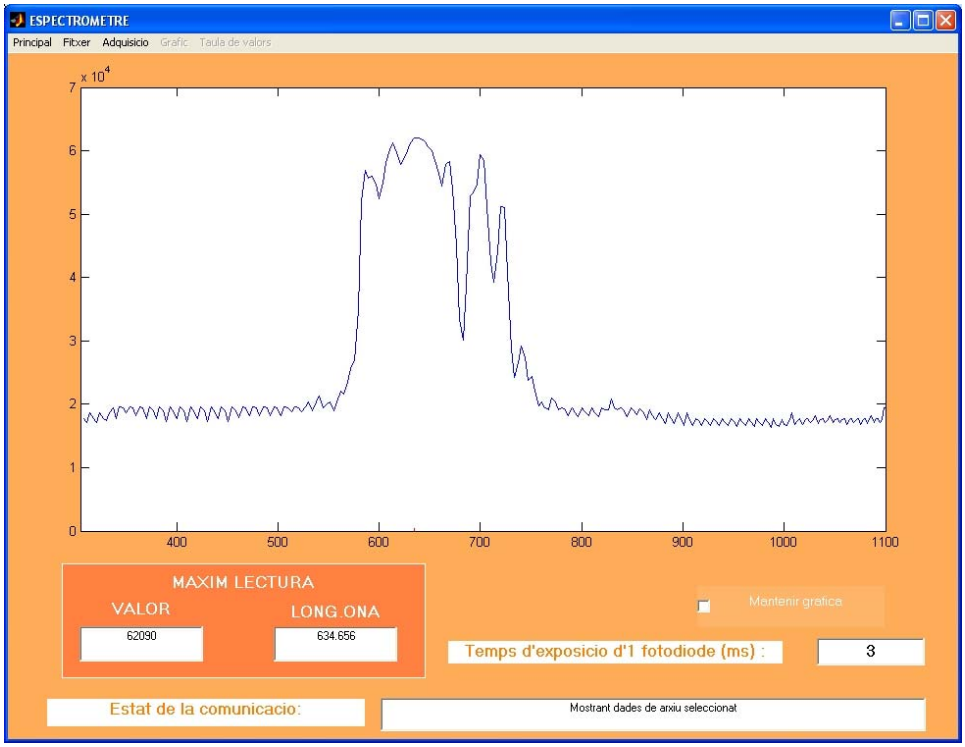


Fig A.5 Fluorescent de neó

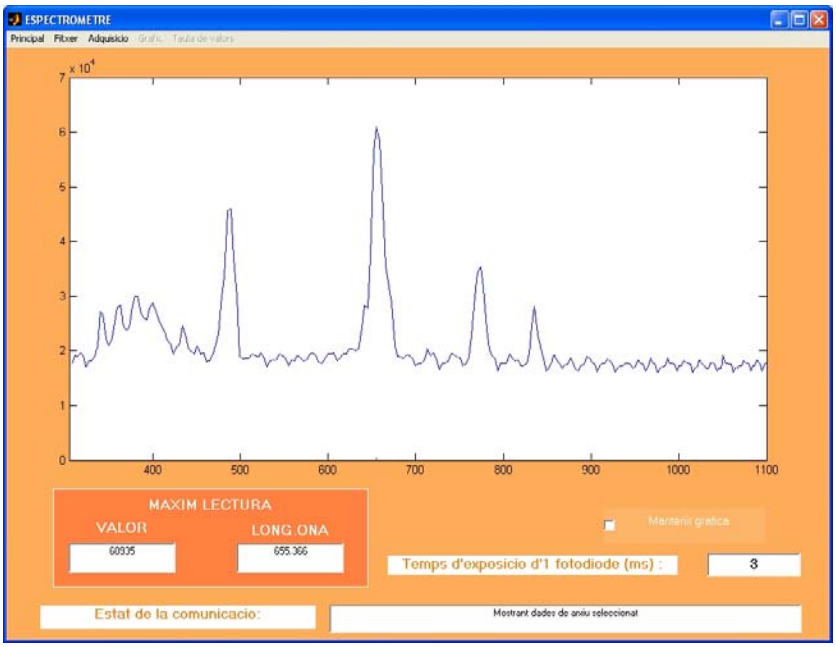


Fig A.6 Fluorescent de vapor d'aigua

A continuació podem veure dos captures representades simultàniament, per observar les diferències entre el temps d'exposició de cada fotodíode. La captura està realitzada amb un led vermell. A la gràfica podem veure l'adquisició obtinguda amb un temps d'exposició de 6ms en color blau, on el màxim arriba a 65535 (saturat). Podem comparar-lo amb l'adquisició obtinguda amb el mateix led però amb un temps d'exposició de 0.1ms en color vermell. En aquest cas el màxim només arriba fins a 36893.

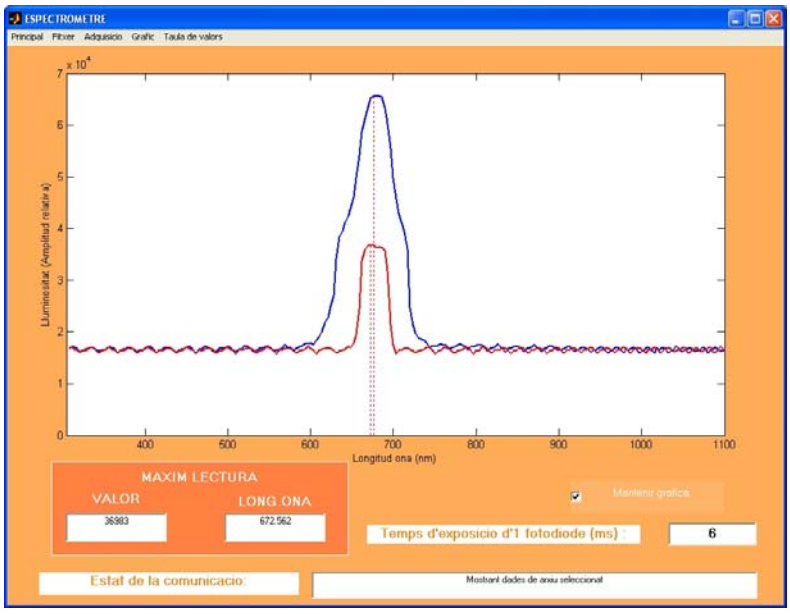


Fig A.7 Led vermell amb temps d'exposició de 0,1ms (vermell) i de 6 ms (blau)

Com ja em comentat anteriorment, em d'introduir una etapa d'acondicionament, per tal de no perdre bits. Vegem-ne un exemple gràfic. En verd veiem l'espectre obtingut d'un fluorescent sense l'etapa d'acondicionament, i en color blau veiem el mateix espectre, amb el mateix temps d'exposició però sense tractar la sortida de vídeo del sensor. Podem comprovar que l'espectre verd està saturat, per tant, hi ha una pèrdua d'informació respecte l'espectre blau.

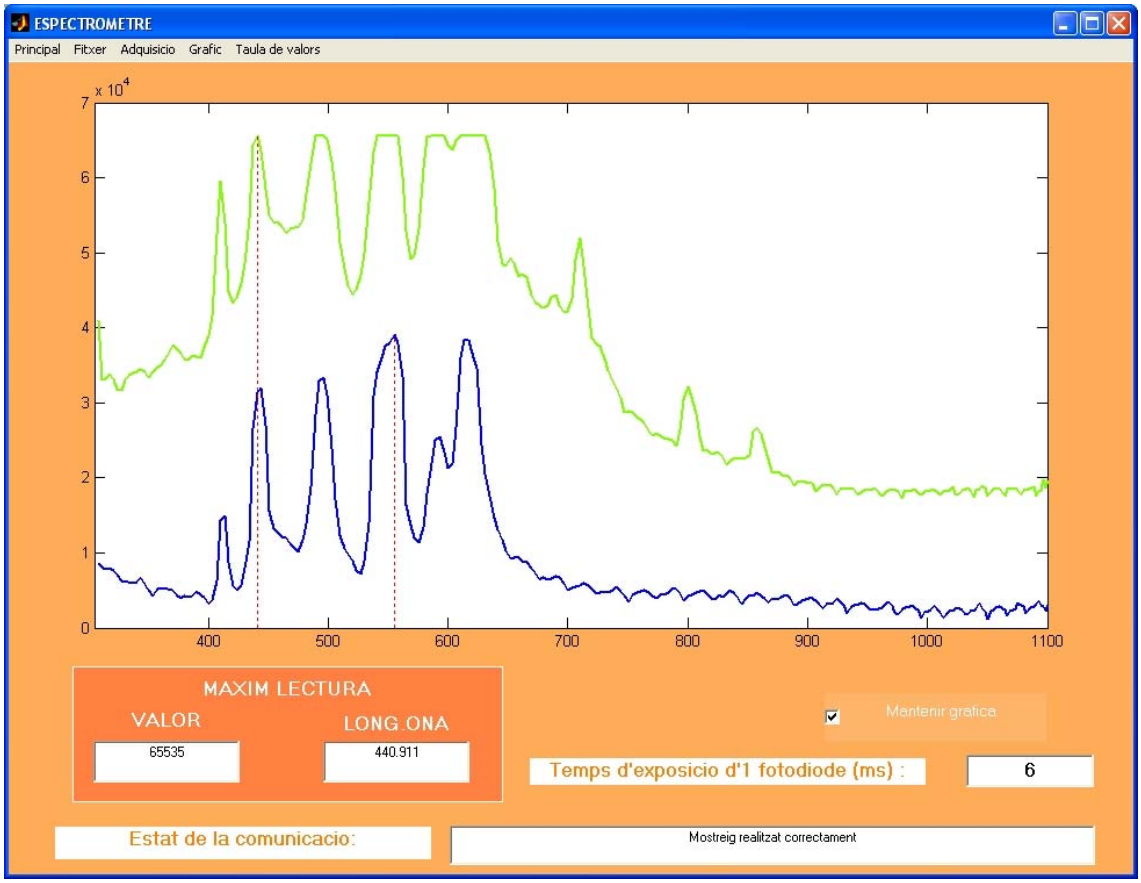


Fig A.8 Fluorescent amb etapa d'acondicionament (blau) i sense (verd) amb el mateix temps d'exposició.

A.2 Gràfiques

Per tal d'obtenir el temps d'exposició de cada fotodíode vam implementar una funció per obtenir el temps que es triga a fer un determinat número d'instruccions. El sentit d'aquesta funció és que la funció *"retard"* que està implementada al codi, no té prou exactitud comparada amb el nivell de precisió que necessitem.

Número de cicles d'instruccions	Període (ms)	Temps baix (ms)
1	0,132	0,121
2	0,138	0,126
5	0,154	0,142
10	0,178	0,166
30	0,278	0,266
100	0,728	0,716
250	1,63	1,618
500	3,14	3,128
1000	6,12	6,108

Taula A.1 Valors de retard

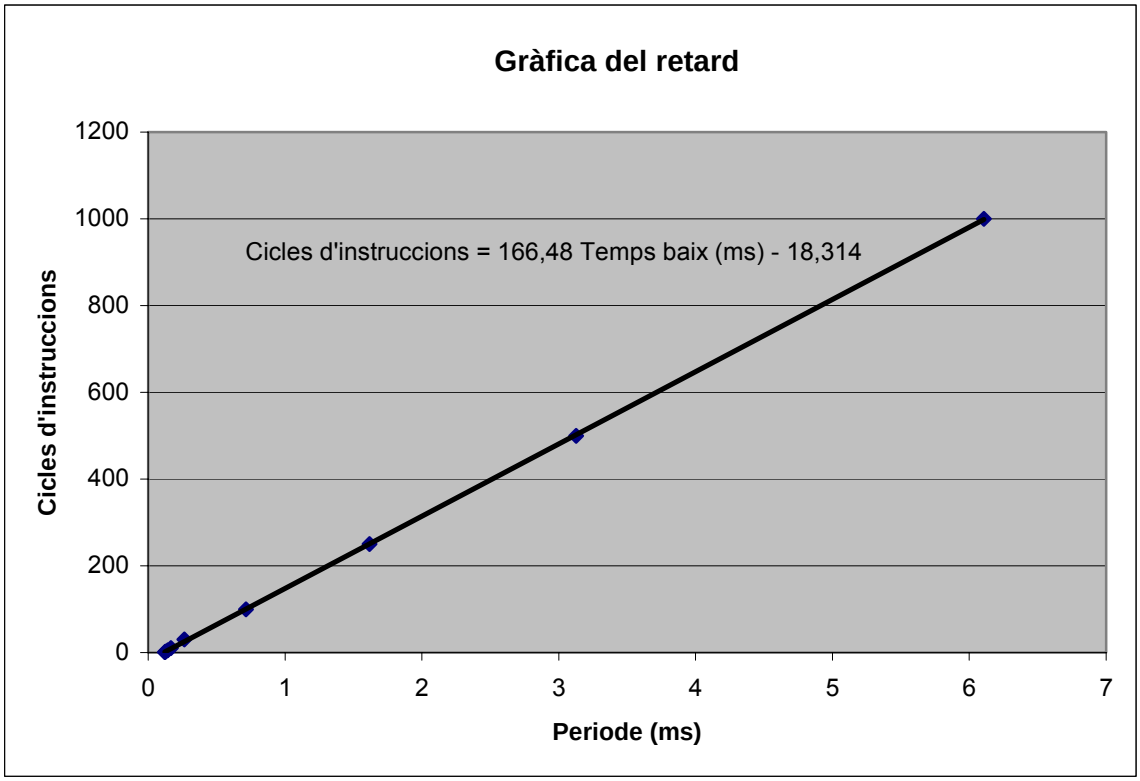


Fig. A.9 Gràfica del retard

A continuació podem veure la gràfica de linealitat del sensor, en funció del temps d'exposició de llum de cada fotodíode.

A.3 Llista material

DESCRIPCIÓ COMPONENT	MODEL	UNITATS	PREU
Sensor hiperespectral	Hamamtsu S8378-256N	1	420 €
dsPIC, digital signal controller	Microchip dsPIC30F6012	1	30 €
Conversor analògic digital de 16 bits	Analog devices AD977AN	1	15 €
Adaptador de nivells RS232	Màxim MAX232A	1	1€
Regulador de tensió	STMicroelectronics L7805CV	1	0,70€
Rellotge de 4MHz	CMAC model HC49	1	1€
Connector d'alimentació		1	0,20€
Connector RS232		1	1€
Connector RJ11-6		1	0,40€
Resistències	SMD 1206	5	2€
Condensadors	SMD 1206	11	2€
Sòcols		5	3€
Tira de pas 1,27mm		4	2€
Placa de circuit imprès		1	-
		TOTAL	**480 €**

Taula A.10 Taula del cost aproximat de la placa final

A.4 Codi dsPIC

```
*****************************************************************************************
*       contador.c                                                                      *
*****************************************************************************************

#include <io30f6012.h>
#include "timer.h"
#include "spi.h"
#include "uart.h"

int dades[256];
int control=1;
int temps=1;

void main() {

  int i;
  int j;

  PORTB= 0;
  TRISB=0x0000;

  PORTC=0;
  TRISC=0x0000;

  PORTD=0;
  TRISD=0X0100; Posem RD8 (BUSY) com a input

  SENSOR
  RC2=1;//ST
  RC1=1;//CLOCK
  RB14=0;//CONTROL DE GUANY Vg

  AD
  RB4=1;//RC

  while(1){

     InitUART();

     while(!U1STA_URXDA); Si val 0 el no hi ha res per rebre
     control=RxUART();

     EndUART();

     switch(control){

       case(2):
```

```
InitUART();
configurar_SPI();

mostreig_buit(); Mostrejat buit per "descarregar" els fotodiodes

 RC2=0;
 retard(1);
 RC1=0;
 retard(temps); Aquí variem el temps d'integració
 RC2=1;
 retard(1);

 RB4=0;
 retard(1);
 RB4=1;

 RC1=1;

 while(RD8==0);BUSY
 SPI2BUF=0x0000;

 while(SPI2IF==0);Quan s'ha acabat la transmissió SPI2IF val 1

 for(i=0;i<255;i++)
   {
    dades[i]=SPI2BUF;
    SPI2IF=0;
    RC1=1;
    retard(1);
    RC1=0;
    retard(temps);Variem el temps d'integració
  RB4=0;
  retard(1);
  RB4=1;

  while(RD8==0);BUSY
  SPI2BUF=0x0000;

  while(SPI2IF==0);Quan s'ha acabat la transmissió SPI2IF val 1
    retard(1);

 }

 dades[i]=SPI2BUF;
   no_ports_SPI();

   SPI2IF=0;

 RC1=1;
```

```
        ENVIAMENT DE LES DADES

        for(i=0;i<256;i++)
        {
          TxUART16(dades[i]);
          retardms(8);

        }

        EndUART();
        no_ports_SPI();

        break;

        case(7):

        InitUART();
        temps=RxUART16();
        EndUART();

        break;

      }

  }

}
```

```
*****************************************************************************************
*       spi.h                                                                           *
*****************************************************************************************

#include <io30f6012.h>

void configurar_SPI(void);
void no_ports_SPI(void);

*****************************************************************************************
*       spi.c                                                                           *
*****************************************************************************************

#include <io30f6012.h>

void configurar_SPI(void){

   Habilitem les interrupcions
   SPI2IF=0;
   SPI2IE=1;
   SPI2IP=000;Sense prioritat

   Configurar SPI2CON
   SPI2CON_FRMEN=0; Framed SPI deshabilitada
   SPI2CON_DISSDO=1; Només com a receptor
   SPI2CON_MODE16=1; Transmissió de 16 bits

   SPI2CON_SMP=1; Configuració de l'instant de mostreig
   SPI2CON_CKP=0;
   SPI2CON_CKE=0; Configuració de la forma del 'dada clock'

   SPI2CON_SSEN=0; Deshabilitem el pin SS_negat. No l'utilitzarem

   SPI2CON_MSTEN=1; Configurem la comunicació SPI com a master

   Per configurar el rellotge partim de la base que FCY = 1MHz ( en MIPS )

   SPI2CON_SPRE0=1;
   SPI2CON_SPRE1=1;
   SPI2CON_SPRE2=1; *1

   SPI2CON_PPRE0=0;
   SPI2CON_PPRE1=1; *16  SPI rate = 1MHz/1*16

   SPI2STAT_SPIEN=1; Configurem com a ports SPI
}
void no_ports_SPI(void){
   SPI2STAT_SPIEN=0; Configurem com a I/O ports

}
```

```
*********************************************************************************************
*       uart.h                                                                              *
*********************************************************************************************

#include <io30f6012.h>

#define BAUDRATE 9600 Velocitat en bps
#define FREQ        1000000 Freqüència en MIPS


Inicialització del perifèric UART
void InitUART(void);

Enviament d'un byte
void TxUART(int byte);

Enviament de 2 bytes
void TxUART16(int bytes);


Recepció d'un byte
char RxUART(void);

Recepció de 2 bytes
int RxUART16(void);

Desactivació del perifèric UART
void EndUART(void);


*********************************************************************************************
*       uart.c                                                                              *
*********************************************************************************************

#include <io30f6012.h>

#define BAUDRATE 2400 Velocitat en bps
#define FREQ        1000000 Freqüència en MIPS


Inicialització del perifèric UART
void InitUART(void) {
U1MODE=0x8000;  Habilitem UART
U1STA=0x0000;   Configuració per defecte: 8 bits, sense paritat, 1 bit stop
U1BRG=(FREQ/(16.0*BAUDRATE)); Calcul del BaudRate (optimitzat per 9600)
NSTDIS=1; Disable nested interrupts
U1STA_UTXEN=1; Habilitar transmissió
return;
}
```

Enviament d'un byte

```
void TxUART(int byte) {
U1TXREG=byte;
return;
}
```

Enviament de dos bytes

```
void TxUART16(int bytes) {
U1TXREG=bytes;

bytes>>=8;
U1TXREG=bytes;
return;

}

char RxUART(void) {
char byte;
byte=U1RXREG;
return(byte);
}

int RxUART16(void){

int temps;
char byte1;
char byte2;

while(!U1STA_URXDA); Si val 0 el no hi ha res per rebre
byte1=RxUART(); Més pes

while(!U1STA_URXDA); Si val 0 el no hi ha res per rebre
byte2=RxUART(); Menys pes

temps=(byte1*256)+byte2;

return(temps);

}

void EndUART(void) {
U1MODE_UARTEN=0; Deshabilitem UART
return;
}
```

```
*************************************************************************************************
*       timer.h                                                                                 *
*************************************************************************************************

#include <io30f6012.h>

void mostreig_buit();

void retard(unsigned int iteracions);

Genera un retard de X ms
void retardms(int ms);

Genera un retard de X us
void retardus(int us);

*************************************************************************************************
*       timer.c                                                                                 *
*************************************************************************************************

#include <io30f6012.h>

int j;
int i;

void retard(unsigned int iteracions){

  for(j=0;j<iteracions;j++);

}

void mostreig_buit(){

        RC2=0;
        retard(1);
        RC1=0;
        retard(1);
        RC2=1;
        retard(1);
        RC1=1;

        for(i=0;i<255;i++)
          {
            RC1=1;
            retard(1);
            RC1=0;
            retard(1);
         }

        RC1=1;}
```

```
void retardms(int ms) {

double prescaler= 0.008; Calculem el valor del període necessari
int periode;                per tenir els ms desitjats amb un preescaler
periode=ms/prescaler;              de 1/8 a 1 MIPS

TMR1 = 0;            Clear Timer 1
PR1 = periode;       Configurem el període (temps d'espera fins que salta la
interrupció)
T1IF = 0;      Clear interrupt flag
T1CON = 0x8010;  Fosc/4, 1:8 prescale, Start TMR1
while(T1IF == 0); Esperem que salti la interrupció
return;
}

void retardus(int us) {

double prescaler= 0.001; Calculem el valor del període necessari
int periode;                per tenir els ms desitjats amb un preescaler
periode=us/prescaler;       de 1/8 a 1 MIPS

TMR1 = 0;    Clear Timer 1
PR1 = periode; Configurem el període (temps d'espera fins que salta la
interrupció)
T1IF = 0;  Clear interrupt flag
T1CON = 0x8000;  Fosc/4, 1:1 prescale, Start TMR1
while(T1IF == 0); Esperem que salti la interrupció
return;
}
```

A.5 Codi Matlab

```
function projecte(funcio)

global ctrl;
global f;
global s;

if strcmp(funcio,'inici')

    format
    acccio=0;
    f=openfig('interficie.fig');
    ctrl=guihandles(f);

    set(ctrl.guardar_fitxer, 'Enable', 'off');
    set(ctrl.desconectar, 'Enable', 'off');
    set(ctrl.adquisicio, 'Enable', 'off');
    set(ctrl.taula, 'Enable', 'off');
    set(ctrl.grafic, 'Enable', 'off');

    ctrl.conectar=0;
    ctrl.dades=0;
    ctrl.llista=0;
    guidata(f,ctrl);

    xlim([300 1100]);
    axes(ctrl.graf);
    xlabel('Longitud ona (nm)');
    ylabel('Lluminositat (Amplitud relativa)');


    port=serial('COM1');
    port.BaudRate=2400;
    port.InputBufferSize=512;
    ctrl.sp=port;
    guidata(f,ctrl);

    retardms=0.1;

end

if strcmp(funcio,'conectar')

    set(ctrl.adquisicio, 'Enable', 'on');
    set(ctrl.desconectar, 'Enable', 'on');
    set(ctrl.conect, 'Enable', 'off');
    set(ctrl.sortir, 'Enable', 'off');
    set(ctrl.adquirir_dades, 'Enable', 'on');
```

```
    set(ctrl.estat, 'String', 'Conexio realitzada correctament');

    ctrl.dades=0;
    guidata(f,ctrl);

end

if strcmp(funcio,'adquirir')

        projecte('mantenir');
        set(ctrl.estat,'String','Mostrejant, si us plau espera...')
        temp=adquirir(ctrl);
        ctrl.dades=temp;

        x=temp(:,1);
        y=temp(:,2);

        plot(x,y);
        xlim([300 1100]);
        pinta_max(temp,ctrl);

        set(ctrl.guardar_fitxer, 'Enable', 'on');
        set(ctrl.grafic, 'Enable', 'on');
        set(ctrl.taula, 'Enable', 'on');

        set(ctrl.estat,'String','Mostreig realitzat correctament');

end

if strcmp(funcio,'zoom')
    zoom;
    check_onoff;
end

if strcmp(funcio,'reixat')

    switch get(ctrl.graf, 'XGrid');
    case 'on'
        set(ctrl.graf, 'XGrid', 'off');
        set(ctrl.graf, 'YGrid', 'off');
    case 'off'
        set(ctrl.graf, 'XGrid', 'on');
        set(ctrl.graf, 'YGrid', 'on');
    end

    switch get(hObject, 'Checked')
    case 'off'
    set(hObject, 'Checked', 'on');

    case 'on'
```

```
    set(hObject, 'Checked', 'off');

    end

end

if strcmp(funcio,'desconectar')

    set(ctrl.adquisicio, 'Enable', 'off');
    set(ctrl.desconectar, 'Enable', 'off');
    set(ctrl.conect, 'Enable', 'on');
    set(ctrl.sortir, 'Enable', 'on');

    set(ctrl.estat, 'String', 'Desconectat de la placa');

    s=ctrl.sp;
    fopen(s);
    fwrite(s,6);
    fclose(s);

end

if strcmp(funcio,'guardar')

   info=ctrl.dades;
   dades=info(:,2);

   arxiu=uiputfile('*.dat', 'Introdueix el nom de l''arxiu a guardar');
       if arxiu~=0
          arxiu=[arxiu '.dat'];
          save(arxiu,'dades','-ASCII');
       end
end

if strcmp(funcio,'obrir')

       arxiu=uigetfile('*.dat', 'Selecciona l''arxiu amb les dades');
       if arxiu~=0
          temp(:,1)=load('long_ona.dat');
          temp(:,2)=load(arxiu);

          ctrl.dades=temp;

          axes(ctrl.graf);
          xlabel('Longitud ona (nm)');
          ylabel('Lluminositat (Amplitud relativa)');

       end

        set(ctrl.estat, 'String', 'Mostrant dades de arxiu seleccionat');
```

```
end
if strcmp(funcio,'sortir')
    clear;
    clc;
    close all;
end

if strcmp(funcio,'temps_int')

    format
    prompt = {'Introdueix el nou temps d''integracio :'};
    dlg_title = 'Canviar temps d''integracio';
    num_lines = 1;
    def = {'0.1'};

    retardms = inputdlg(prompt,dlg_title,num_lines,def);
    retardms=cell2mat(retardms);
    retardmsnum=str2num(retardms);
    retard=uint16((166.48*retardmsnum-18.314)+0.5);%Fer truncament del
nombre d'iteracions necessari

    retard1=uint8(bitshift(retard,-8));
    retard2=uint8(bitand(retard,255));

    set(ctrl.estat, 'String', 'Modificant temps d''integracio...');
    s=ctrl.sp;
    fopen(s);
    pcontrol=7;
    fwrite(s,pcontrol);
    fwrite(s,retard1);%Mes pes
    fwrite(s,retard2);%Menys pes
    fclose(s);

    set(ctrl.estat, 'String', 'Modificant temps d''integracio...');

    missatge=['El nou temps d''adquisicio es de ', retardms, ' ms.'];
    msgbox(missatge);
    set(ctrl.temps, 'String', retardms);
    set(ctrl.estat, 'String', 'Temps d''integracio modificat correctament');

end

if strcmp(funcio,'taula_valors')

    f2=openfig('taula.fig');
    ctrl2=guihandles(f2);

    temp=ctrl.dades;
    save dades.tmp temp -ascii;
```

```
    taula=load('dades.tmp');
    taula=num2str(taula);
    set(ctrl2.llista,'String',taula);

end

if strcmp(funcio,'tanca_taula')

    !del dades.tmp;
    close
end

if strcmp(funcio,'mostreig_cont')

global conectar
conectar=1;
[ctrl]=parar_iniciar(ctrl,conectar);
end

if strcmp(funcio,'parar')

        global conectar
          conectar=0;
end

if strcmp(funcio,'mantenir')

    if get(ctrl.mantenir_graf, 'Value')   %Comprovem si hi ha marcada la opció de
mantenir la gràfica actual
        hold on;
    else
        hold off

    end

end

function [maxim,index]=obtenir_maxim(dades)

dades=dades(:,2);

maxim=max(dades);

index=find(dades==maxim);
index=index(1);
saturats=(find(dades>=60000));
inferior=(find(dades>=20000));

if(length(saturats)~=0)
```

```
    maxim=18000*(((3*length(inferior)-length(saturats)))/(length(inferior)-
length(saturats)));
end

function temps = nou_temps (error,integral, derivada, temps_max, temps_min,
temps_anterior)

if(error >= 10)
    temps=temps_anterior+(0.03*(error)+0.008*(integral)+0.002*(derivada));

elseif(error<=-10)
    temps=temps_anterior-(0.03*(error)+0.008*(integral)+0.002*(derivada));
else
    temps=temps_anterior;
end

if(temps<temps_min)
    temps=temps_min; %No podem donar un temps menor a aquest
elseif(temps>temps_max)
    temps=temps_max; %No pot donar un temps major a aquest
end

function [ctrl]=parar_iniciar(ctrl,conectar)

        s=ctrl.sp;

        projecte('mantenir')

        global conectar;
        error_anterior=0;
        temps=1;
        temps_min=1;
        temps_anterior=0;
        limit=60000;
        integral=0;

        while (conectar==1)

            [dades]=adquirir(ctrl);
            ctrl.dades=dades;

            x=dades(:,1);
            y=dades(:,2);

            plot(x,y);
            xlim([300 1100]);

            pinta_max(dades,ctrl);

            [maxim]=obtenir_maxim(dades);
```

```
        maxim
        error=limit-maxim;
        integral=integral+error;
        derivada=error_anterior-error;

        temps_min=1; %iteracions
        temps_max=1000; %iteracions


temps=nou_temps(error,integral,derivada,temps_max,temps_min,temps_anteri
or)

        set(ctrl.temps, 'String', temps);

        error_anterior=error;
        temps_anterior=temps;

        retard=uint16(temps);

        retard1=uint8(bitshift(retard,-8));
        retard2=uint8(bitand(retard,255));

        temps=((((temps-0.5)+18.314)/166.48));
        set(ctrl.temps, 'String', temps);

        fopen(s);
        pcontrol=7;
        fwrite(s,pcontrol);

        fwrite(s,retard1);%Mes pes
        fwrite(s,retard2);

        fclose(s);

    end

        set(ctrl.estat,'String','Sistema aturat')
        set(ctrl.max_llum, 'String','-');
        set(ctrl.max_lo, 'String', '-');

        retard=temps;

        set(ctrl.temps, 'String', temps_min);
        retard=temps_min;
        fopen(s);
        pcontrol=7;
        fwrite(s,pcontrol);
        fwrite(s,retard);
        fclose(s);
```

```
function [temp]=adquirir(ctrl)

    s=ctrl.sp;
    fopen(s);
    pcontrol=2;
    N_Bytes=0;

    fwrite(s, pcontrol);

    while(s.BytesAvailable<512)
    end

    [rebut,N_Bytes]=fread(s,512);
    fclose(s);

    j=1;
    for i=1:2:512,
       dades(j)=rebut(i)+(rebut(i+1)*256);%Passem els valors rebuts de 2
elements de 8 bits a un de 16.
       j=j+1;
    end

    temp(:,1)=load('long_ona.dat');
    temp(:,2)=dades';

function [ctrl]=pinta_max(temp,ctrl)

    maxim=find(temp(:,2)==max(temp(:,2)));
    maxim=maxim(1,:);
    set(ctrl.max_llum, 'String', temp(maxim,2));
    set(ctrl.max_lo, 'String', temp(maxim,1));

    linia_max_x=zeros(1,temp(maxim,2));
    linia_max_x=linia_max_x+temp(maxim,1);
    linia_max_y=[1:temp(maxim,2)];
    hold on;
    plot(linia_max_x,linia_max_y,':r');
    hold off;
```