Universitat Autònoma de Barcelona

escola d'enginyeria

2037. SISTEMA ENCASTRAT DE MONITORITZACIÓ D'INFORMACIÓ MULTIPLE
- IMPLEMENTACIÓ PER A VEHICLES -

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per
Antonio Flores Ortega
i dirigit per
Leonardo Portaña Elettrico
Bellaterra, 9 de setembre de 2010.

Universitat Autònoma de Barcelona

etse
Escola Tècnica Superior d'Enginyeria

El sotasignat, **Leonardo Portaña Elettrico**

Professor/a de l'Escola Tècnica Superior d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en **Antonio Flores Ortega**

I per tal que consti firma la present.

Signat: .....

Bellaterra, ...de.....de 2010.

Agraïments

M'agradaria donar les gràcies a totes les persones que m'han ajudat i recolzat als moments necessaris en la realització de la meva carrera. Aquest projecte va dedicat a tots ells.

Gracies al meu director de projecte Leonardo Portaña Elettrico pel temps i esforç dedicat, i per haver-m'he ajudat a resoldre els dubtes que han anat sorgint. Sense els seus consells i suggeriments aquest projecte no hauria estat el mateix.

Gracies a la meva família pels ànims i el recolzament que m'han impulsat a continuar i acabar en aquest any ple de dificultats.

Gracies a la meva parella Raquel per aguantar-me a casa al dia a dia, i per recolzar-me i animar-me mentre durant la carrera i en la realització d'aquest projecte.

Gracies als meus companys i amics, els quals d'alguna manera també han contribuït a que aquest projecte sigui possible.

Índex

Índex de figures

1. Introducció

El projecte presentat a continuació es situa al marc de l'automoció, en concret en el monitoratge de sensors i presentació d'informació, realitzant les funcions bàsiques d'un ordinador d'abord convencional juntament amb característiques addicionals que permeten una explotació posterior de dades en ordinadors personals.

Aquest projecte neix de la necessitat d'estendre la informació que el fabricant proporciona de sèrie al conductor del vehicle. Partint d'aquesta necessitat, s'han realitzat sondeigs a les principals comunitats d'usuaris d'aquests vehicles a Internet per tal de determinar l'acceptació, i en conseqüència la viabilitat de comercialització.
La proposta inicial va ser la implementació d'un velocímetre digital reutilitzant sistemes propietaris del vehicle. Es va realitzar una primera versió en llenguatge assemblador per a PIC que es comunicava amb el display d'origen i implementava el velocímetre. Un cop assolit això es va pensar en estendre les funcionalitats del mateix creant un sistema complet al qual es barreja programació en assemblador de PIC i codi C propietari de Microchip.

A continuació es descriuen breument els capítols que conformen aquesta memòria per poder obtenir una visió global del que es tractarà a continuació:

En primer lloc es parla dels objectius del projecte, on s'exposen les idees a desenvolupar, i es defineixen uns objectius vers el desenvolupament del sistema.
A continuació es planteja el projecte, on s'analitzen les funcions del sistema com a punt de partida i els recursos necessaris i de que disposem per tal de poder desenvolupar-lo.
Seguidament es mostra la planificació, en la qual s'analitza per etapes i dintre d'un marc temporal totes les tasques que s'han dut a terme per a la realització d'aquest projecte, donant una breu explicació de cadascuna. Aquesta part incorpora un apartat que parla dels problemes trobats que han anat trencant la planificació prèvia.
A continuació ve la memòria tècnica, a on es profunditza en totes les parts i/o funcionalitats del sistema. S'explica a nivell tècnic com funciona cadascuna de les funcions que realitza el sistema, juntament amb diagrames de flux que acompanyen les explicacions.
Per últim, es detallen les conclusions de la realització del projecte. També es parla dels objectius que no s'han pogut assolir, juntament amb les possibles millores i/o noves funcionalitats que es tenien en ment i que es podrien implementar si es disposés de més temps.

2. Objectius

Al finalitzar aquest projecte es pretén realitzar un ordinador d'abord amb funcions esteses per a vehicles que sigui capaç de suplir totes les necessitats d'informació addicional que els conductors puguin requerir. Aquests tipus de projectes poden ser molt ambiciosos ja que es difícil posar un límit als objectius que es volen aconseguir, i es el temps qui marca al final les fites que es poden arribar a aconseguir per tancar una primera versió. No obstant, tot el que s'ha desenvolupat s'ha realitzat amb la idea de continuar ampliant aquest projecte, per tant els objectius subratllats que s'expliquen continuació marquen una primera versió del sistema, i els no subratllats són aquells que per motius de temps no s'han pogut aconseguir però que es plantejaran per a versions posteriors.

- Mostrar missatges i aconseguir compatibilitat amb els dos tipus de display que integren la sèrie de models de vehicles escollits.

  Degut a que els displays no utilitzaven la implementació estàndard del bus I^2C, aquesta fase va consistir en trobar informació sobre el funcionament dels displays i obtenir una primera orientació per començar. Un cop amb les especificacions estàndards del bus I^2C i una idea de les diferencies amb l'estàndard es va implementar les subrutines necessàries per enviar missatges i acte seguit es va començar a afinar els timings experimentalment per tal de poder enviar les dades als displays amb la màxima velocitat possible.

- Mostrar la velocitat instantània del vehicle

  Consisteix en localitzar el cable provinent de l'encoder del velocímetre, interpretat el senyal i mostrar la velocitat lineal instantània del vehicle.

- Implementar canvi de funció amb la maneta

  Es tracta bàsicament d'entendre el sistema d'interrupcions del microcontrolador, veure el funcionament de la maneta original i muntar el sistema de canvi de funció.

- Mostrar el valor de la velocitat angular del volant motor

  Al igual que el velocímetre, es tracta de trobar el cable portador de la senyal de l'encoder del volant motor, interpretar-lo i mostrar-lo al display.

- Funcionalitat de velocitat màxima

  Es tracta de poder mantenir actualitzat un valor de la velocitat lineal màxima assolida pel vehicle des de l'última vegada que es va posar a 0 i que es mostra com una funció al sistema.

- Funcionalitat de distància recorreguda parcial

  Es tracta de anar actualitzat un valor de distància recorreguda en funció de la velocitat des de l'última vegada que es va posar a 0 i que es mostra com una funció al sistema. Aquesta funció era necessària per a una posterior implementació de la funcionalitat de velocitat mitja.

- Funcionalitat de cronòmetre

  Es tracta de crear un comptador de temps que es posa a 0 a la vegada que la distància parcial recorreguda, i que juntament amb aquesta donaria lloc a una altra funcionalitat, la velocitat mitja.

- Funcionalitat de velocitat mitja

  Tracta de poder tenir un valor actualitzat que mostri a l'usuari la velocitat mitja del seu recorregut des de l'última vegada que es va posar a 0. La posta a 0 es realitza a la vegada per a la distància parcial, el temps de recorregut i la velocitat mitja.

- Consum instantani de combustible.

  Consisteix en poder mostrar al display el consum instantani de combustible del vehicle, el litres/h si està parat o en litres/100Km si es troba en moviment. Per a realitzar això es necessari localitzar el cable que porta la senyal d'obertura d'injector des de la ECU (Engine Control Unit) fins als injectors i interpretar-la en conseqüència per estimar el consum. D'aquesta funció es deriven altres com per exemple el consum mitjà.

- Nivell de combustible al tanc.

  Es basa en trobar el cable provinent del sensor o boia del tanc de combustible, interpretar-lo i mostrar al display la quantitat de litres de combustible que resten. D'aquesta funció, juntament amb la funció de consum instantani es deriven altres funcions, com per exemple la quantitat de quilòmetres que podem realitzar amb el nivell de combustible actual (autonomia).

- Hora i data.

  Tracta de incorporar al sistema un RTC (Real Time Clock) per tal de tenir una forma fiable de conèixer l'hora i la data per a poder incloure-la al datalogger.

- Datalogger.

  Tracta de guardar dades rellevants al trajecte realitzat en un suport digital amb informació sobre el dia i l'hora per a donar la possibilitat de poder explotar aquestes dades a posteriori.

3. Plantejament

3.1. Plantejament del sistema

El sistema esta dissenyat per a funcionar en vehicles de la marca Opel, concretament en els models que han coexistit en el mercat durant un període de temps concret, aproximadament entre 1998 i 2004. Desenvolupa funcions bàsiques d'ordinador d'abord com per exemple velocitat del vehicle, voltes del motor, velocitat màxima, distància recorreguda parcial, hora i data.

Figura 1. Esquema general del sistema

Els usuaris poden governar totes les funcions del sistema mitjançant la maneta de control, tal com es descriu a la següent màquina d'estats:

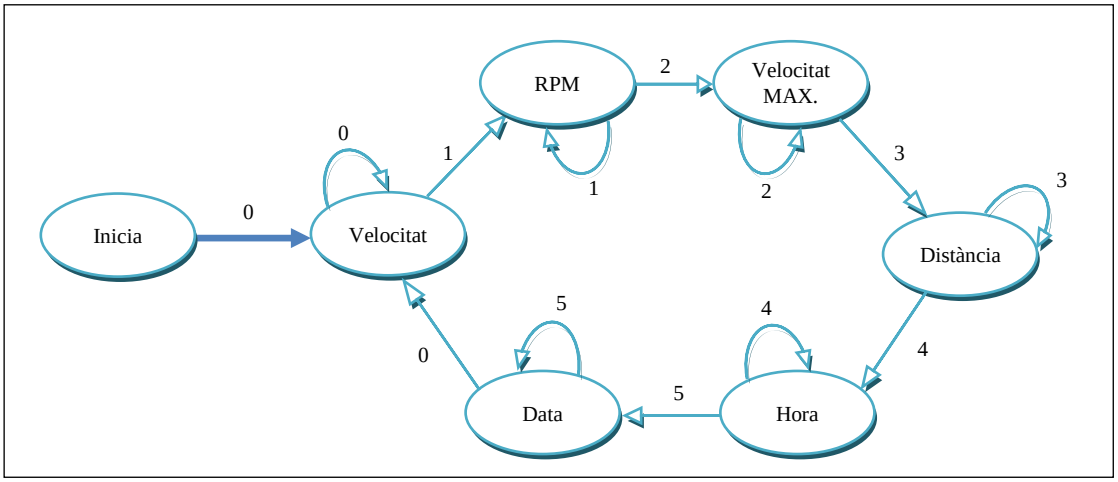


Figura 2. Màquina d'estats de les funcions del sistema

Mostra dades a un display integrat al vehicle i enregistra dades a un suport extern de memòria flash per a una posterior explotació.

3.2. Hardware

3.2.1. Microcontrolador PIC18F4620

El PIC18F4620 és un microcontrolador de la família PIC18 corresponent a la casa Microchip. És un microcontrolador de 8 bits amb prestacions mitjes-altes. A aquest projecte hem triat l'encapsulat PDIP de 40 pins.

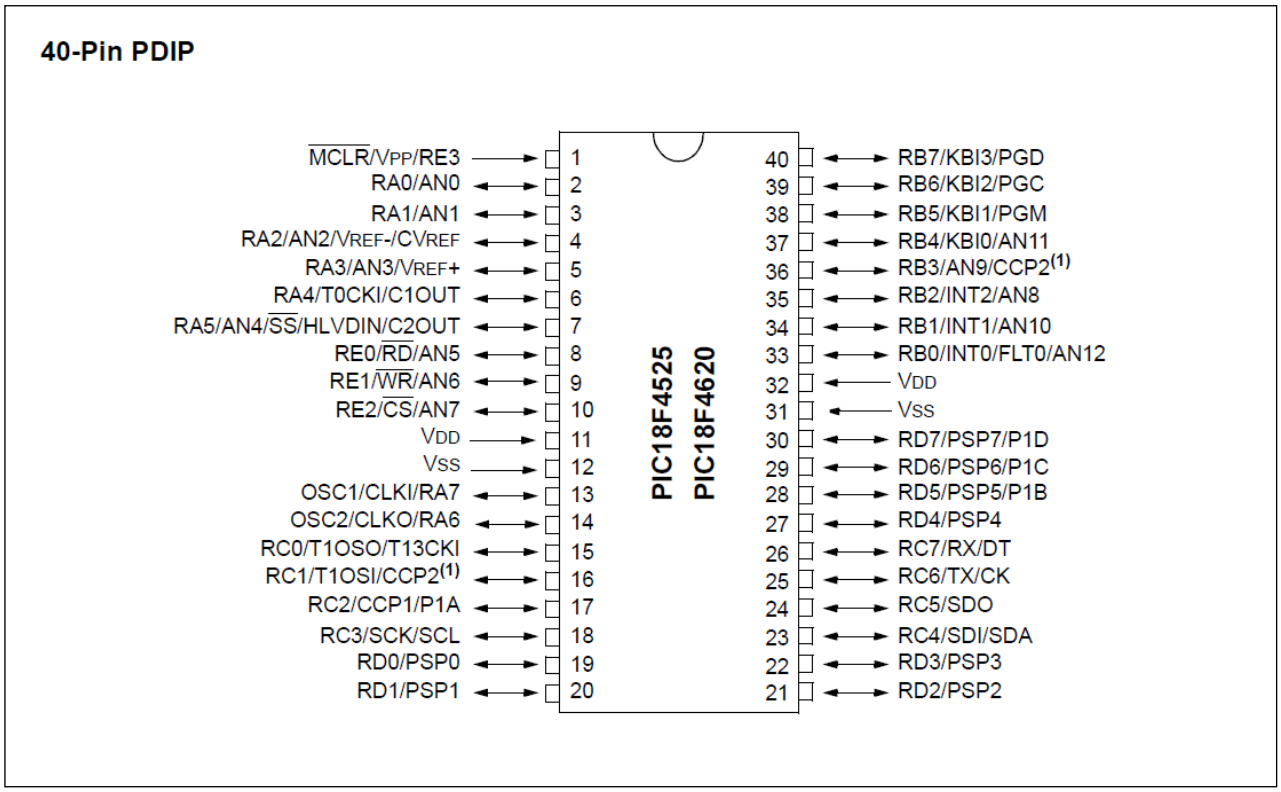


Figura 3. Diagrama de pins del PIC18F4620

Disposa de 3 modes de funcionament bàsic i modes de baix consum:

- Mode Run: És el mode de funcionament normal, al qual funciona tant la CPU com els perifèrics.
- Mode Idle: És un mode al qual la CPU està parada però els perifèrics continuen en funcionament. Per sortir d'aquest mode es necessari que es produeixi una interrupció, s'esgoti el temps del temporitzador del gos guardià (WDT) o es produeixi un Reset.
- Mode Sleep: Tan la CPU com els perifèrics es troben aturats. Es tornarà al funcionament normal un cop s'hagi produït una interrupció, s'esgoti el temps del temporitzador del gos guardià (WDT) o es produeixi un Reset.

Aquest microcontrolador posseeix una gran versatilitat vers el rellotge principal, des d'un rellotge intern a 8 MHz a externs i arribar a una freqüència de rellotge principal de fins a 48MHz. Posseeix divisors preescalars i un multiplicador x4 (PLL) per poder adaptar la freqüència del rellotge principal a les necessitats. En el nostre cas hem fet servir un cristall de quars de freqüència 20 MHz (HS). Per tal de calcular les subrutines de espera necessàries a qualsevol aplicació s'ha de tenir en compte que el cicle de rellotge no equival al cicle de instrucció, sinó que el per a cada 4 cicles de rellotge tenim 1 cicle d'instrucció, per tant tots els càlculs s'hauran de realitzar amb el valor FOSC/4. Gracies al pipeline que utilitza, totes les instruccions, excepte salts, son de cicle únic.

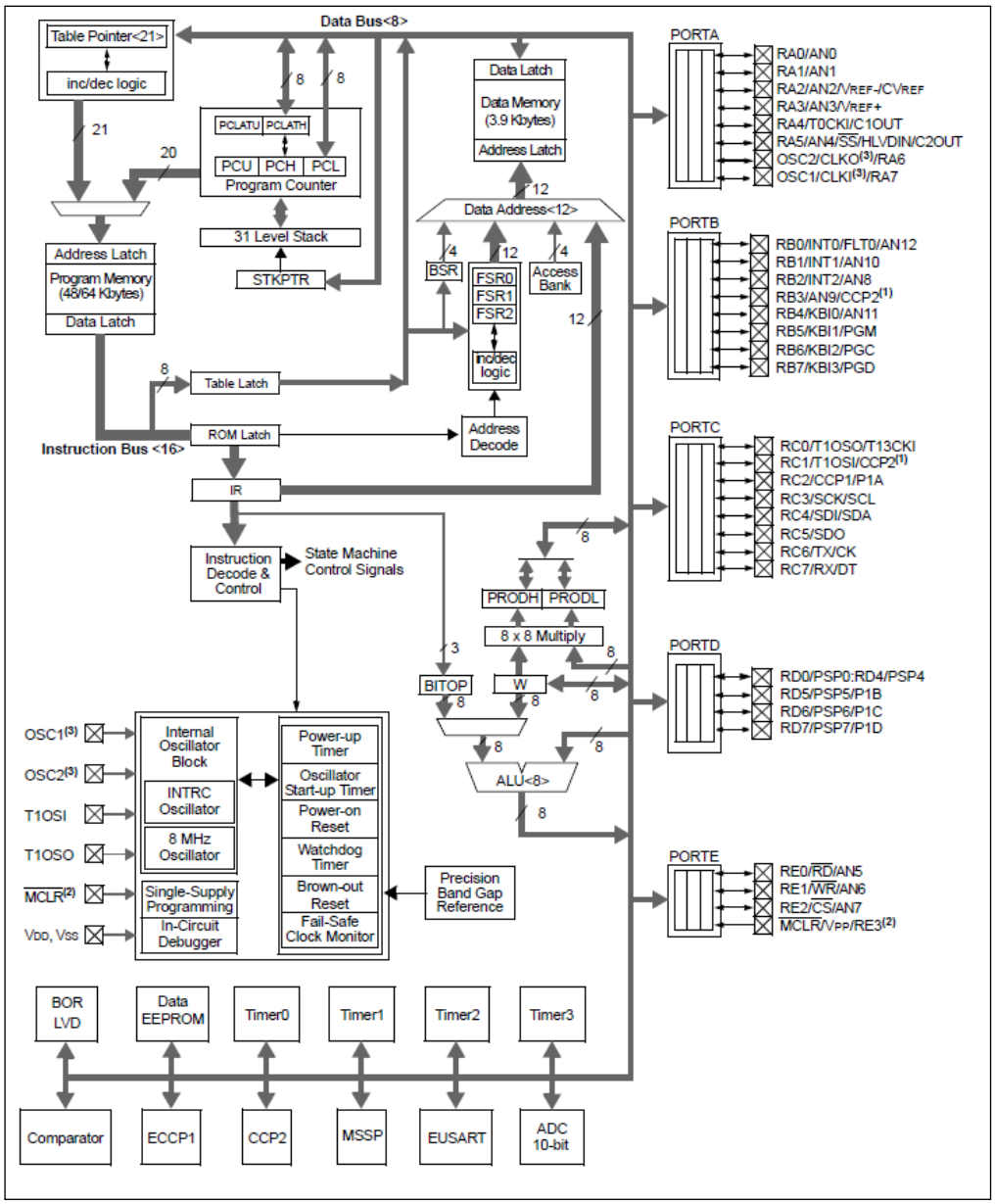


Figura 4. Diagrama de blocs del PIC18F4620 (40 PIN)

Respecte a les característiques de memòria disposa de 64KB de flash per a instruccions, 3986 bytes de memòria RAM estàtica y 1024 bytes de EEPROM per a dades y una pila de direccions de 31 paraules de 21 bits.

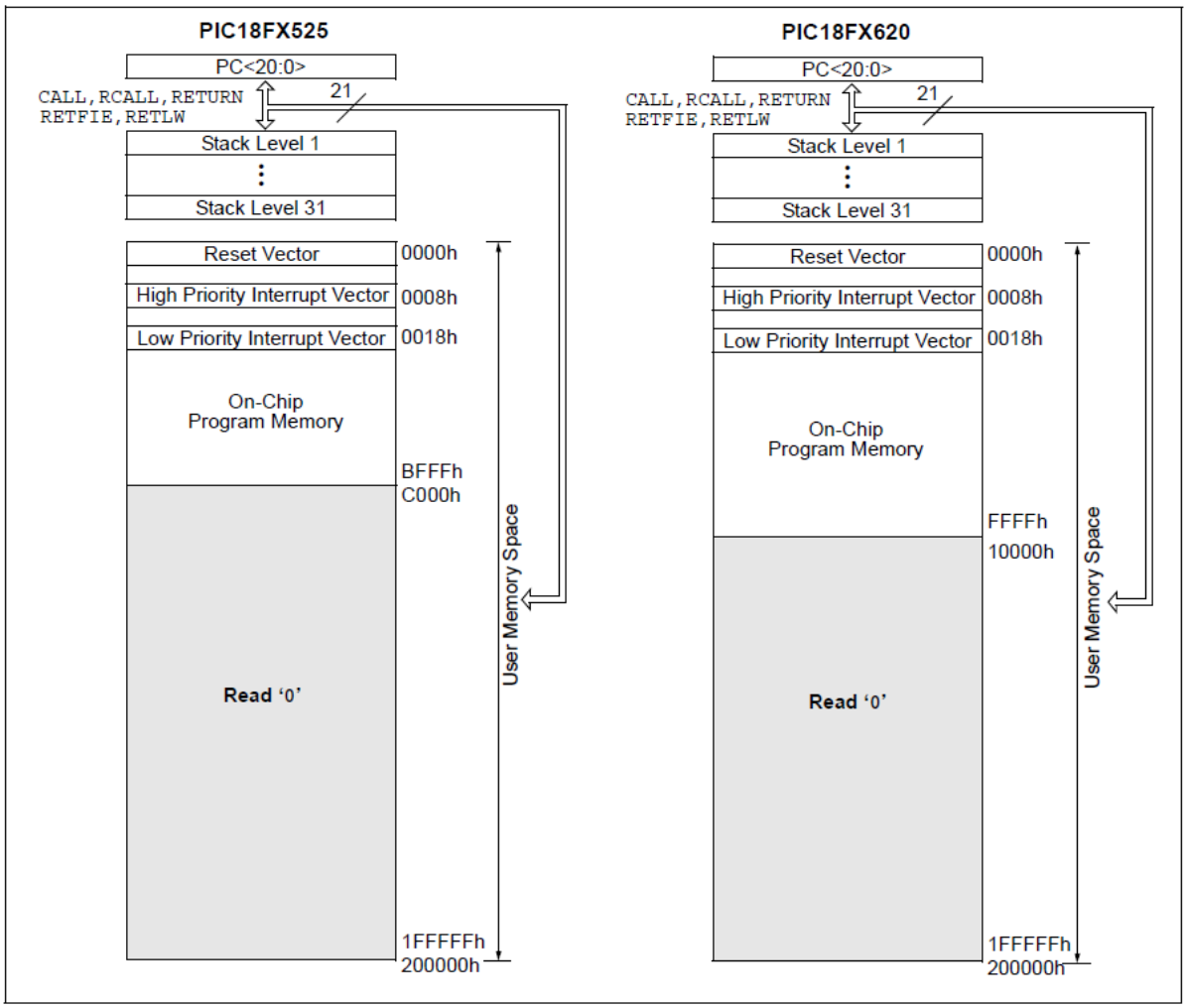


Figura 5. Mapa de memòria de programa i pila del PIC18F4620

Posseeix diferents característiques especials i unitats funcionals:

- Multiplicador hardware 8 x 8 de cicle únic d'instrucció com a part de l'ALU.
- Timer WDT estès programable 4ms a 131s.
- In-Circuit Serial Programming.
- In-Circuit Debug.
- Ampli rang de tensió d'alimentació (2.0V a 5.5V).
- Brown-out reset (BOR) programable.
- Convertidor A/D de 10 bits.
- Mòdul CCP (Capture/Compare/PWM).
- Mòdul ECCP (Enhaced CCP).
- MSSP (Master Synchronous Serial Port).
- EUSART (Enhaced Universal Synchronous Receiver Transmitter).
- 4 Comptadors/Timers de 8 y 16 bits.

A continuació es detallen les característiques i/o mòduls que s'han fet servir a aquest projecte.

3.2.1.1. Mòdul CCP

Es tracta d'un mòdul que configurable com a capturador, comparador o PWM. En aquest projecte s'ha utilitzat la configuració de capturador per tal d'interpretar la senyal provinent dels encoders de celeritat del vehicle i la velocitat angular del volant motor.
El capturador captura una seqüència de polsos d'entrada segons una condició i generar una interrupció. En aquest moment copia el valor de 16 bits del timer associat al capturador segons la seva configuració en dos registres propis de 8 bits per tal de mesurar la senyal d'entrada.
Tant el timer associat com el capturador admeten diferents configuracions. Les configuracions possibles respecte a la condició de captura son les següents:

- Cada flanc ascendent.
- Cada flanc descendent.
- Al 4rt flanc ascendent.
- Al 16é flanc ascendent.

És possible fer servir el Timer 1 o el Timer 3 com a timer associat a cada mòdul CCP (CCP1 i CCP2). La configuració d'aquest forma part de la configuració del Timer 3 i es veurà al apartat pertinent.

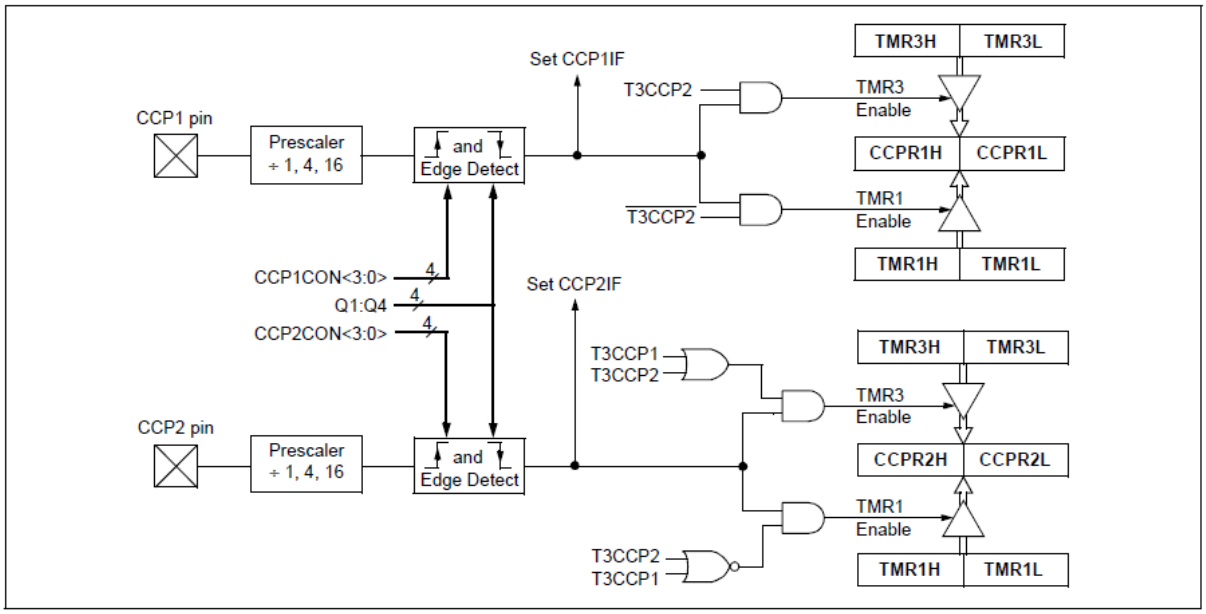


Figura 6. Diagrama de blocs de la operació de captura al PIC18F4620

En aquest projecte s'ha optat per generar una interrupció al 16é flanc ascendent com a condició de captura del mòdul CCP1 que es el que s'ha fet servir.

3.2.1.2. Master Synchronous Serial Port (MSSP)

El MSSP es una interfase sèrie útil per a comunicar-se amb altres perifèrics com per exemple EEPROMs sèrie, registres de desplaçament, controladors de pantalla, etc., o altres microcontroladors.

El MSSP pot funcionar en dos modes, el mode Serial Peripheral Interface (SPI) i el mode Inter-Integrated Circuit (I^2C). Malgrat que en aquest projecte s'ha fet servir I^2C, aquest s'ha implementat per software i el mòdul MSSP s'ha utilitzat en mode SPI per a la comunicació amb targetes de memòria SD/MMC.

Gràcies a la configuració de pins del microcontrolador, el mòdul MSSP en mode SPI admet la transmissió sincronitzada de 8 bits enviant i rebent simultàniament, ja que s'utilitza una línia per enviar dades i una altra independent per a rebre. El MSSP com a SPI admet diferents configuracions respecte a mode de funcionament. En aquest cas es necessita governar les transferències de dades i el conseqüència la freqüència de rellotge de les mateixes, per tant es treballa en mode mestre. Las possibilitats per a escollir la freqüència que governarà les transferències es basen en divisors sobre la freqüència del oscil·lador principal del microcontrolador, i les opcions són FOSC/64, FOSC/16 i FOSC/4.

Degut a que la freqüència del oscil·lador principal es de 20MHz i la freqüència del SPI no pot superar els 10MHz, s'ha triat FOSC/4 = 5MHz. Com a configuracions addicionals s'ha configurat com a estat inactiu del rellotge quan aquest es troba a nivell alt, i les transferències de dades es fan quan el rellotge passa de mode inactiu a mode actiu o en aquest cas al flanc de baixada.

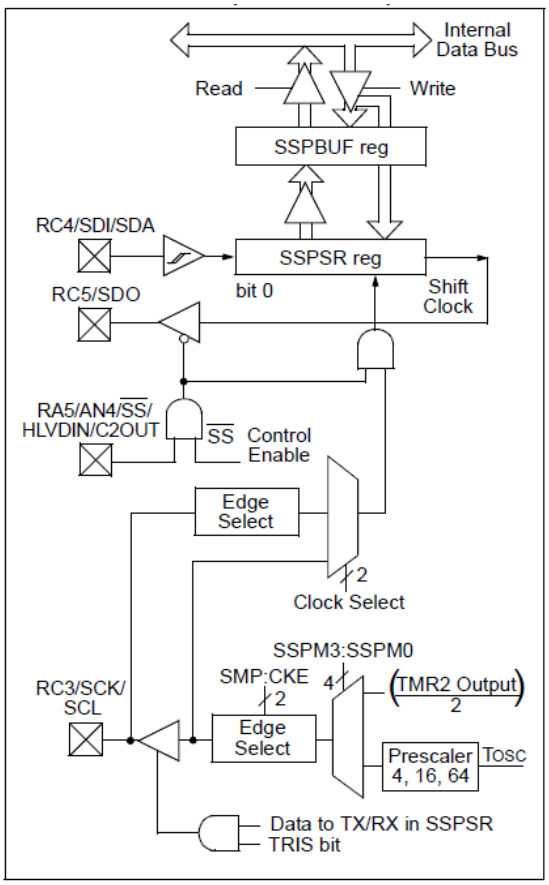


Figura 7. Diagrama de blocs del mòdul MSSP en mode SPI

3.2.1.3. Enhaced Universal Synchronous Receiver Transmitter (EUSART)

L'EUSART és un canal de comunicació sèrie que pot ser configurat en mode asíncron per enviar i rebre simultàniament i pot comunicar al terminals CRT o PC's (port sèrie). En el nostre cas utilitzem el canal USART bàsic, sense les característiques addicionals, en l'entorn virtual (simulació amb Proteus) per comunicar el microcontrolador amb un terminal virtual i poder realitzar debug amb més facilitat. La configuració utilitzada es en mode asíncron, amplada de transmissió de 8 bits, recepció contínua i baix baud rate. Existeix una fórmula per obtenir un valor que serveix per configurar la velocitat de comunicació, i es la següent (en mode asíncron i baixa velocitat):

$$Fosc / (64 * (spbrg + 1)) = Velocitat\ de\ comunicació$$

En el nostre cas utilitzem una velocitat de comunicació d'aproximadament 9600 bauds, per tant:

$$20\ MHz / (64 * (32 + 1)) = 9469{,}69 \approx 9600\ bauds$$

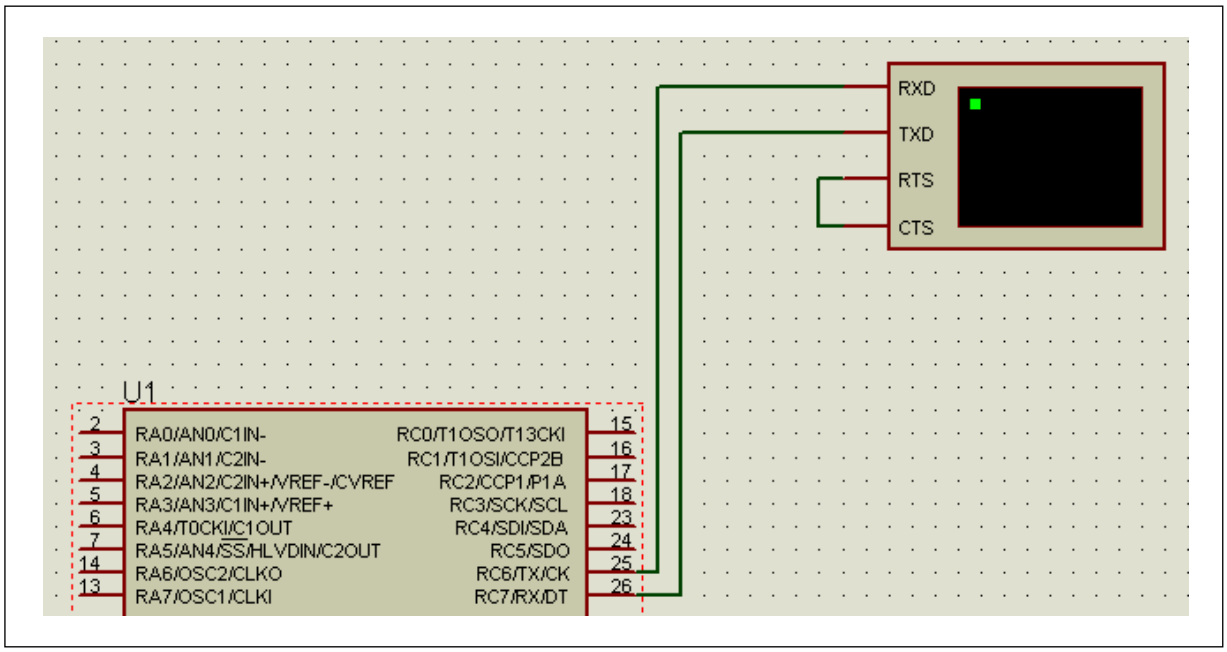


Figura 8. Connexió de terminal virtual a PIC18F4620 en l'entorn ISIS de Proteus

3.2.1.4. Mòdul Timer 0

El mòdul Timer 0 es un comptador configurable com a comptador de 8 bits o 16 bits. També es pot seleccionar la font de rellotge entre dues opcions, rellotge intern o extern. Si s'utilitza el rellotge intern, aquest funciona com a temporitzador, mentre que amb un rellotge intern pot fer-se servir de temporitzador o comptador de polsos.

A l'entrada de rellotge porta un divisor preescalar, el qual es configura per dividir la freqüència de la font de rellotge entre un valor escollit als bits de configuració. El divisor preescalar es de 8 bits, per tant els valors seleccionables són els següents: 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, 1:128 i 1:256. Un altre paràmetre configurable es la transició del pols on s'ha d'incrementar el comptador, al flanc ascendent o al flanc descendent. Aquest timer pot generar una interrupció (si s'ha habilitat tal interrupció prèviament) si es desborda.

A la versió final d'aquest projecte no s'ha fet servir aquest timer, malgrat això, en primeres versions s'utilitzava, configurat amb la font de rellotge externa, per mesurar la quantitat de polsos al port corresponent del microcontrolador, i poder així mesurar el senyal provinent dels encoders de la celeritat o el de la velocitat de rotació del volant motor del vehicle, es per això que s'ha explicat el funcionament del mateix.

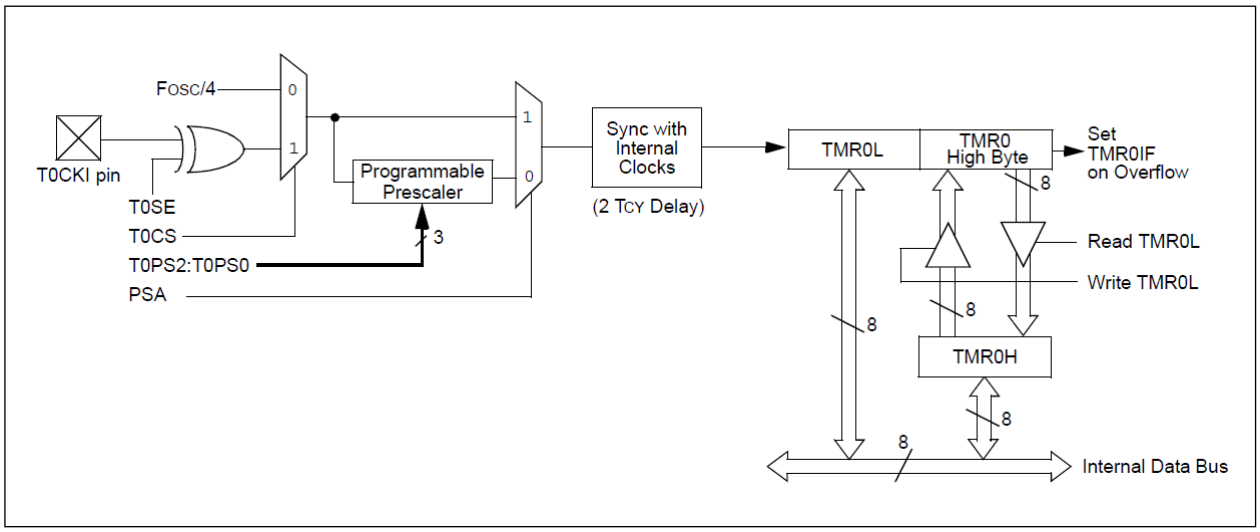


Figura 9. Diagrama de blocs del mòdul Timer 0 del PIC18F4620 en mode 16 bits

3.2.1.5. Mòdul Timer 1

El mòdul Timer 1 es un comptador especial que pot treballar únicament com a comptador de 16 bits. Al igual que el Timer 0, aquest també admet triar entre incrementar segons el rellotge intern del microcontrolador, o bé triar una font externa, amb la peculiaritat de que dóna la possibilitat de sincronitzar amb aquesta senyal d'entrada de manera que sense sincronitzar, aquest actua com a comptador, i al sincronitzar, aquest actua com a temporitzador. Dóna la possibilitat també d'utilitzar el rellotge extern del Timer 1 com a rellotge principal del microcontrolador per a modes de baix consum. Aquest mòdul també pot generar interrupcions si aquestes s'han habilitat prèviament. En el nostre cas, hem associat aquest mòdul al CCP1 per tal de mesurar la llargada dels polsos capturats.

La configuració del Timer 1 associat al comptador CCP1 es realitza al registre de control del Timer 3.

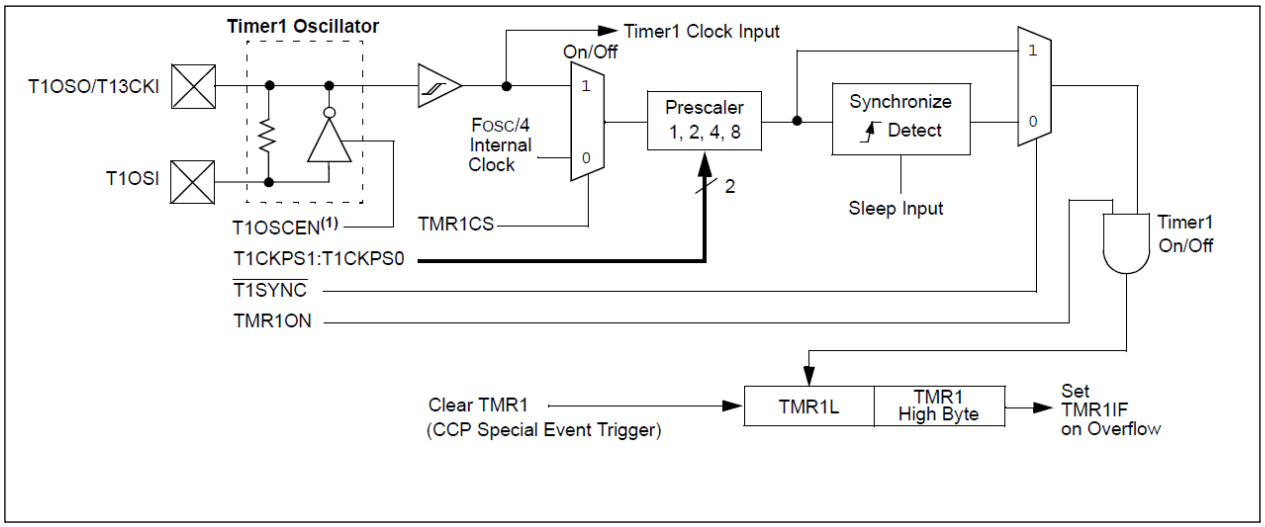


Figura 10. Diagrama de blocs del mòdul Timer 1 del PIC18F4620

3.2.1.6. Mòdul Timer 2

El Mòdul Timer 2 es un temporitzador de 8 bits que només pot funcionar amb la senyal de rellotge intern del microcontrolador, i té característiques especials. A diferència de la resta de temporitzadors, aquest disposa de dos divisors, un preescalar i un postescalar. El funcionament bàsic del temporitzador incrementa segons la senyal del rellotge intern del microcontrolador i el divisor preescalar. El valor del temporitzador es compara contínuament amb un registre anomenat PR2, i en el moment en que es produeix coincidència produeix una sortida al divisor preescalar que és qui activa la interrupció per overflow del Timer 2.

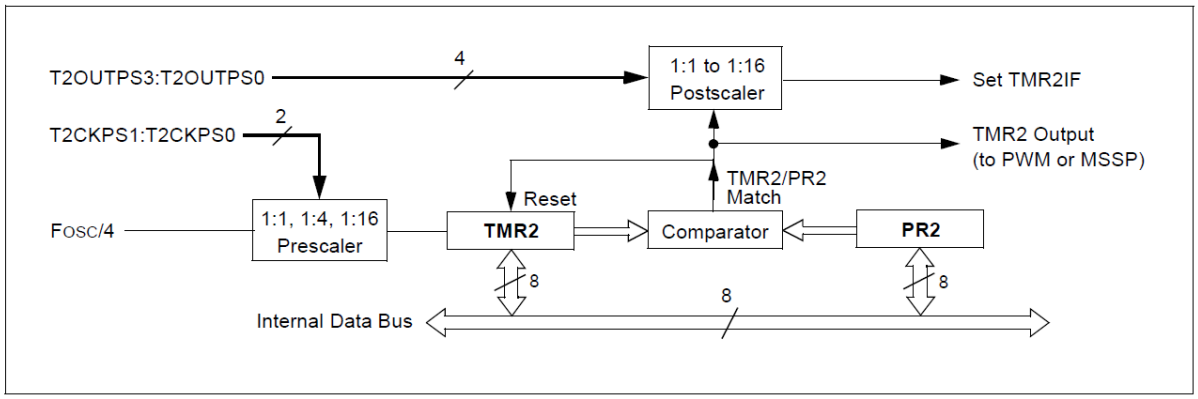


Figura 11. Diagrama de blocs del mòdul Timer 2 del PIC18F4620

En aquest projecte s'ha fet servir el valor màxim tan per al preescalar com per al postescalar, el valor 1:16. Per tal de realitzar una interrupció temporitzada es pot fer un càlcul per obtenir el valor necessari per a precarregar el registre PR2. El càlcul es el següent:

$$PR2 = \frac{\frac{Temps\ (seg.)}{FOSC/4}}{Preescalar \times Postescalar} - 1$$

Mitjançant el següent càlcul es pot deduir segons la configuració de rellotge a aquest projecte que la temporització més gran que es pot aconseguir es de aproximadament 13ms i per tant, si volem temporitzacions superiors (com és el cas) podem fer servir una variable de 16 bits que s'anirà incrementant en cada overflow del Timer 2 fins a aconseguir la temporització desitjada.

3.2.1.7. Interrupcions

El microcontrolador PIC18F4620 disposa de 21 fonts d'interrupció organitzades en dos grups, les interrupcions generals i les interrupcions de perifèrics.

Interrupcions generals:

- Interrupció del Timer 0
- Canvi d'estat al PORTB (RB7:RB4)
- Interrupcions externes (INT0, INT1, INT2)

Interrupcions de perifèrics:

- Lectura/escriptura al port esclau paral·lel.
- Interrupció del conversor A/D
- Interrupció del CCP1
- Interrupció del CCP2
- Interrupció del comparador
- Interrupció del SSP
- Interrupció del MSSP
- Recepció de la EUSART
- Transmissió de la EUSART
- Interrupció del Timer 1
- Interrupció del Timer 2
- Interrupció del Timer 3
- Fallida del oscil·lador
- Escriptura Flash/EEPROM
- Col·lisió al bus
- Interrupció del detector de tensió alta/baixa .

En el moment que es produeix una interrupció es desactiven a nivell global, per la qual cosa no es poden produir fins que no s'acabi d'atendre la interrupció en curs.

Disposa de dos nivells de prioritat d'interrupció configurable per a cada tipus d'interrupció mitjançant uns flags dedicats a aquest fi. Les interrupcions d'alta prioritat poden interrompre les subrutines d'atenció a les interrupcions de baixa prioritat. Existeixen uns flags que s'activen segons la interrupció que s'hagi produït per tal de poder reconèixer la font de la interrupció, i aquest han de ser desactivats per software.

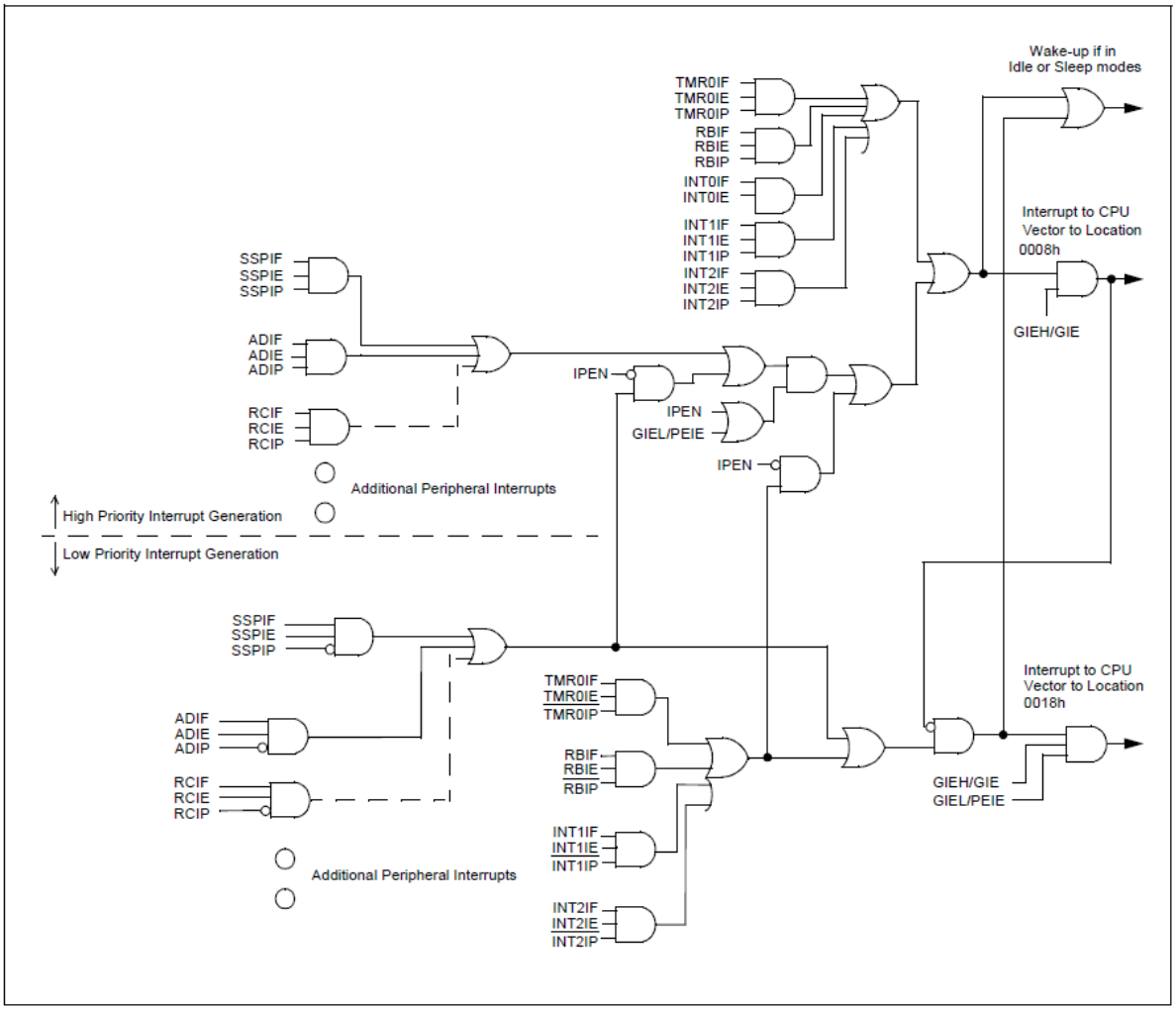


Figura 12. Diagrama lògic del sistema d'interrupcions del PIC18F4620

3.2.2. PICKit2 Clone programmer

El programador utilitzat per a gravar el microcontrolador del sistema es un programador basat en el PICKit2 de l'empresa Microchip. Internament consta d'un microcontrolador PIC18F2550, el qual porta gravat el firmware del PICKit2 original. Malgrat que existeixen les especificacions per muntar-lo a Internet, es va optar per adquirir-lo a la botiga http://www.piccircuit.com/ degut al baix preu.

Les característiques d'aquest programador són:

- Low-cost, petit i d'alt rendiment.
- USB plug and play, no necessita alimentació externa.
- Apte per a ordinador d'escriptori i portàtils.
- Programació d'alta velocitat.

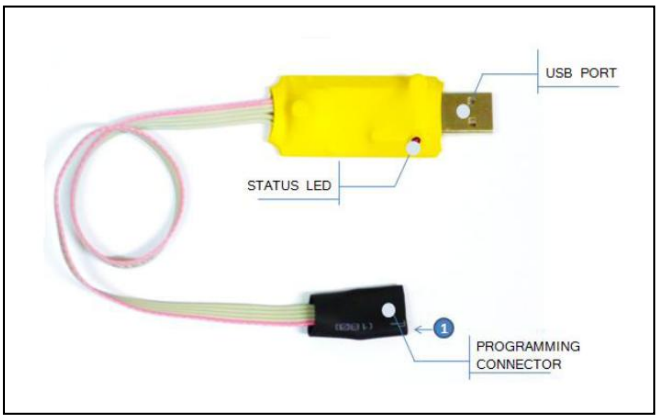


Figura 13. Programador PICKit2 Clone (iCP01)

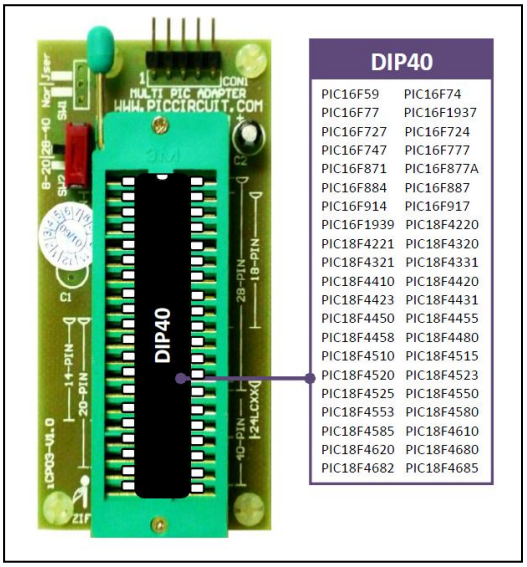


Figura 14. Multi PIC Adapter (iCP03)

3.2.3. Real Time Clock (RTC DS1307)

El DS1307 es un circuit integrat de Dallas Semiconductor molt popular degut a la seva senzillesa d'utilització. Aquest circuit es capaç de mesurar el temps amb gran exactitud i contar anys, mesos, dies del mes, dies de la setmana, hores, minuts i segons. Té en compte la durada dels mesos i els anys bixests. Pot operar en mode 24h o 12h amb indicador AM/PM. Esta codificat en BCD, per tant es summament senzill treballar amb els seus registres, sobretot si s'han de mostrar en un display. Aquest CI està preparat per a treballar sense problemes fins l'any 2010 i a més, ofereix la possibilitat de connexió amb una bateria de liti de 3V la qual garantitzarà que es mantingui la data i l'hora durant 10 anys. El DS1307 té 7 registres per accedir als valors de any, mes, dia, etc..., un registre de control i 56 bytes de memòria SRAM. El pinout del CI es el següent:

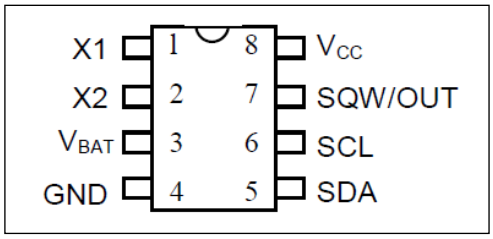


Figura 15. Pins del RTC DS1307

La sortida SQW/OUT es por configurar perquè generi senyals a diferents freqüències. En aquest projecte s'ha configurat perquè generi una senyal 1Hz, de manera que aquest senyal interrompi per la interrupció externa 2 del nostre microcontrolador cada segon, i així realitzar diferents tasques, com per exemple la actualització de la hora al display.

La comunicació amb aquest CI es realitza mitjançant un bus I^2C com a esclau.

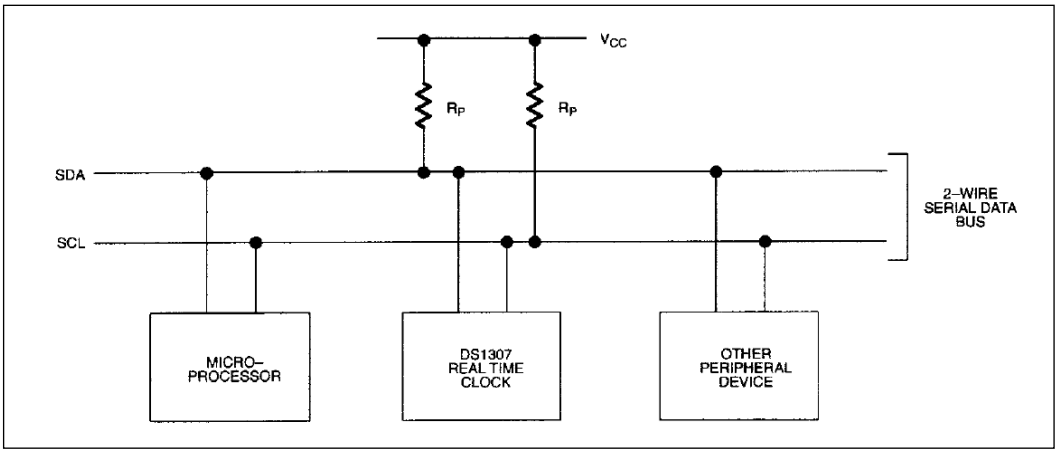


Figura 16. Diagrama de connexió del RTC DS1307

3.3. Software

3.3.1. MPLAB IDE v.8.50

El MPLAB es un entorn gràfic de treball creat per la casa Microchip específicament per a treballar amb els seus microcontroladors. Per defecte porta compilador per a assemblador de PIC específic per a cada família de microcontroladors i el linker.

Permet la compilació de fitxers separats i la creació de projectes compostos per múltiples fitxers de codi i headers.

Porta la possibilitat de seleccionar diferents eines de debug, entre les quals està el MPLAB SIM, que es una eina que permet execució pas a pas, breakpoints, etc.

Entre les seves utilitats per a fer debug, porta per exemple la possibilitat d'explorar variables, registres, memòria de programa, pila, etc. I a més utilitats com el generador d'estímuls per al microcontrolador, un analitzador lògic, etc.

Aquest entorn porta la possibilitat d'afegir extres, com per exemple compiladors propietaris de C (C18, CCS...) i la eina de debugger VSM de la suite Proteus, el qual permet la simulació amb el microcontrolador muntat en un circuit complet. Ambdós complements s'han fet servir en aquest projecte.

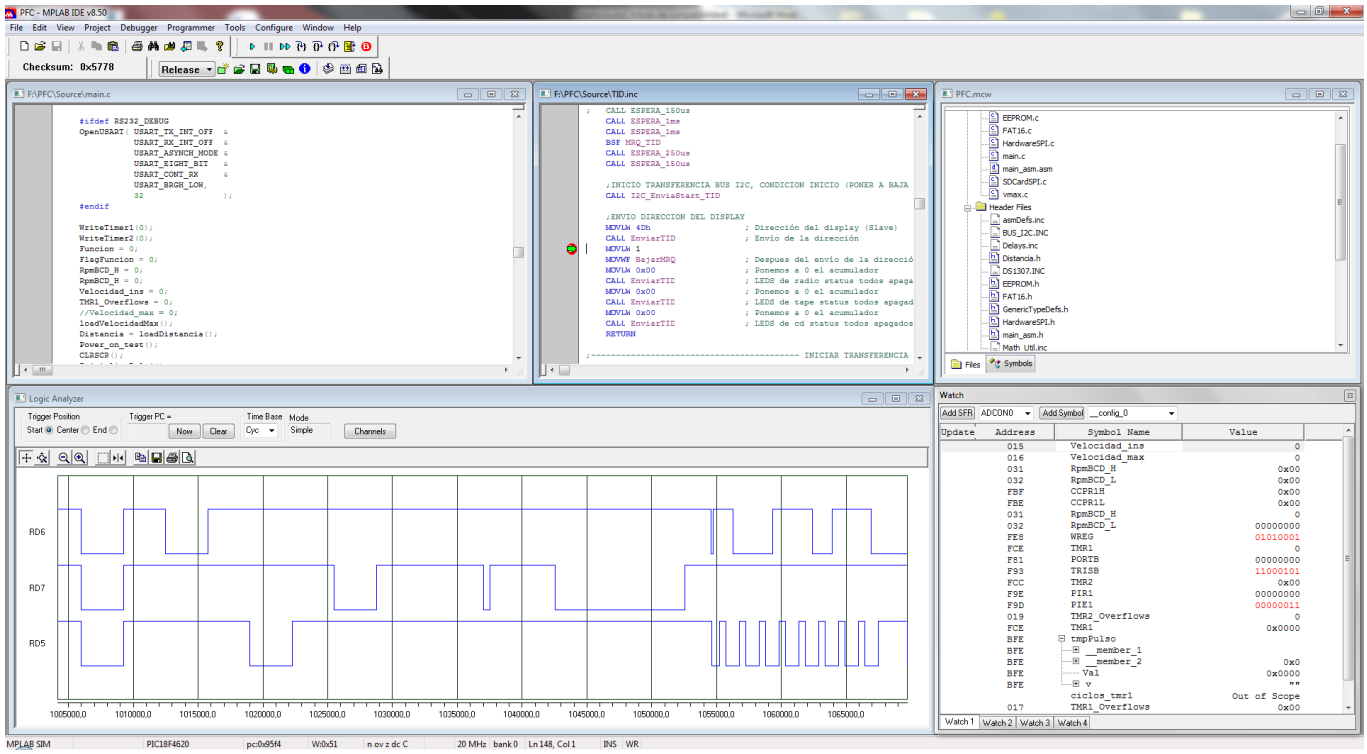


Figura 17. Entorn gràfic MPLAB v8.50

3.3.1.1. Assembler PIC

El llenguatge assemblador de PIC es força semblant entre les diferents famílies de microcontroladors de Microchip, però no es exactament igual. En aquest projecte s'ha utilitzat un PIC18F4620, i per tant serà d'aquest microcontrolador del que es parlarà únicament. Aquest microcontrolador conta amb el grup estàndard de 75 instruccions de la família PIC18 i a més conta també amb 8 instruccions extra per tal d'optimitzar codis recursius o que utilitzin una pila implementada per software. La majoria d'instruccions requereixen 16 bits per emmagatzemar-les, tret de 4 instruccions que necessiten el doble.

Les instruccions de 16 bits es divideixen en un codi d'operació que especifica el tipus d'instrucció i un o més operands. El repertori d'instruccions es classifica en quatre tipus:

- Operacions orientades a byte.
- Operacions orientades a bit.
- Operacions de literals.
- Operacions de control.

La majoria d'operacions orientades a byte tenen 3 operands: el registre utilitzat, el destí de l'operació i la memòria accedida.

En el cas de les operacions orientades a bit, la majoria disposen també de 3 operands que són: el registre utilitzat, el bit del registre utilitzat i la memòria accedida.

En el cas de les operacions de literals poden utilitzar alguns d'aquests operands: un valor literal per a carregar a un registre, el registre a on es vol carregar el valor o cap operand.

En el cas de les operacions de control han d'utilitzar alguns d'aquests operands: Adreça de memòria de programa, el mode de les instruccions CALL o RETURN, el mode de lectura o escriptura de taules o cap operand.

3.3.1.2. Compilador C per a PIC18F (C18)

El llenguatge C18 específic per a treballar amb microcontroladors de la família PIC18 de l'empresa Microchip. Es tracta d'un llenguatge propietari del qual es necessari l'adquisició de llicència per tal de poder-ne fer us. En aquest projecte s'ha fet servir la versió per a estudiants, la qual es gratuïta.

L'ús del llenguatge C18 respecte ASM sobre PIC aporta molts avantatges de valor afegit, com es per exemple el gran increment en la productivitat donat que pot treballar amb nombres reals de fins a 32 bits i incorpora diferents llibreries com per exemple les de I^2C, SPI, UART, USART, generació PWM, strings i funcions matemàtiques de punt flotant.

C18 suporta tots els tipus de dades de l'estàndard ANSI, els quals es mostren a continuació:

Tipus	Mida (bits)	Rang	
char	8	-128	127
Signed char	8	-128	127
Unsigned char	8	0	255
int	16	-32.768	32.767
Unsigned int	16	0	65.535
short	16	-32.768	32.767
Unsigned short	16	0	65.535
Short long	24	-8.388.608	8.388.607
Unsigned short long	24	0	16.777.215
long	32	-2.147.483.648	2.147.483.647
Unsigned long	32	0	4.294.967.295

Figura 18. Tipus enters

Tipus	Mida (bits)	Exponent		Rang	
float	32	-126	128	$2_{-126} \approx 1.17549435e-38$	$2_{128} * (2-2_{-15}) \approx 6.80564693e+38$
double	32	-126	128	$2_{-126} \approx 1.17549435e-38$	$2_{128} * (2-2_{-15}) \approx 6.80564693e+38$

Figura 19. Tipus en punt flotant

L'emmagatzemament de dades a una variable de més d'un byte es realitza en format little-endian, es a dir, el byte menys significatiu ocupa la posició més baixa.
El compilador C18 té divergències respecte l'estàndard ISO. L'estàndard ISO requereix que les operacions aritmètiques es realitzin amb precisió d'int o superior, però C18 realitza les operacions amb la mida de l'operand més gran, fins i tot quan aquest es més petit que un int. Malgrat això es pot forçar a que es comporti d'acord amb l'estàndard ISO especificant-ho a la compilació.

```
unsigned char a, b;
unsigned i;
a = b = 0x80;
i = a + b; /* ISO requereix que i == 0x100, però a C18 i == 0 */
```

Respecte a les constants numèriques, C18 suporta els prefixos estàndard per definir hexadecimals (0x), octals (0) i afegeix suport per a definir binaris (0b).

3.3.2. ISIS 7.5 SP3 (Proteus Suite)

Intelligent Schematic Input System (ISIS) es una eina gràfica que permet dissenyar esquemes de circuits electrònics, i que compta amb una amplia varietat de components reals per a realitzar-los. A més a més dels components, ISIS també compta amb una gran varietat de eines útils per a comprovar el funcionament de qualsevol circuit, com per exemple comprovadors de bus, generador de funcions, oscil·loscopi, generador lògic, etc.

ISIS està associat a un mòdul de Proteus anomenat VSM, el qual permet la simulació en temps real del nostre circuit. És especialment útil per a treballar amb microcontroladors, ja que podem carregar el nostre codi al microcontrolador del circuit dissenyat i fer un debug del circuit i del microcontrolador molt acurat.

En aquest projecte s'ha fet servir el programa ISIS per tal de crear un circuit amb el PIC18F4620 amb el codi compilat i poder fer debug de les diferents parts del projecte. Un exemple es la comprovació del funcionament del datalogger, pel qual s'ha connectat un terminal virtual, un debugger de SPI i una targeta SD a la qual se li ha muntat una imatge d'una targeta SD real. El mòdul VSM queda també associat al IDE MPLAB, de manera que seleccionant com a debugger el Proteus VSM podem carregar el circuit muntat amb el programa ISIS i fer una simulació pas a pas amb el circuit.

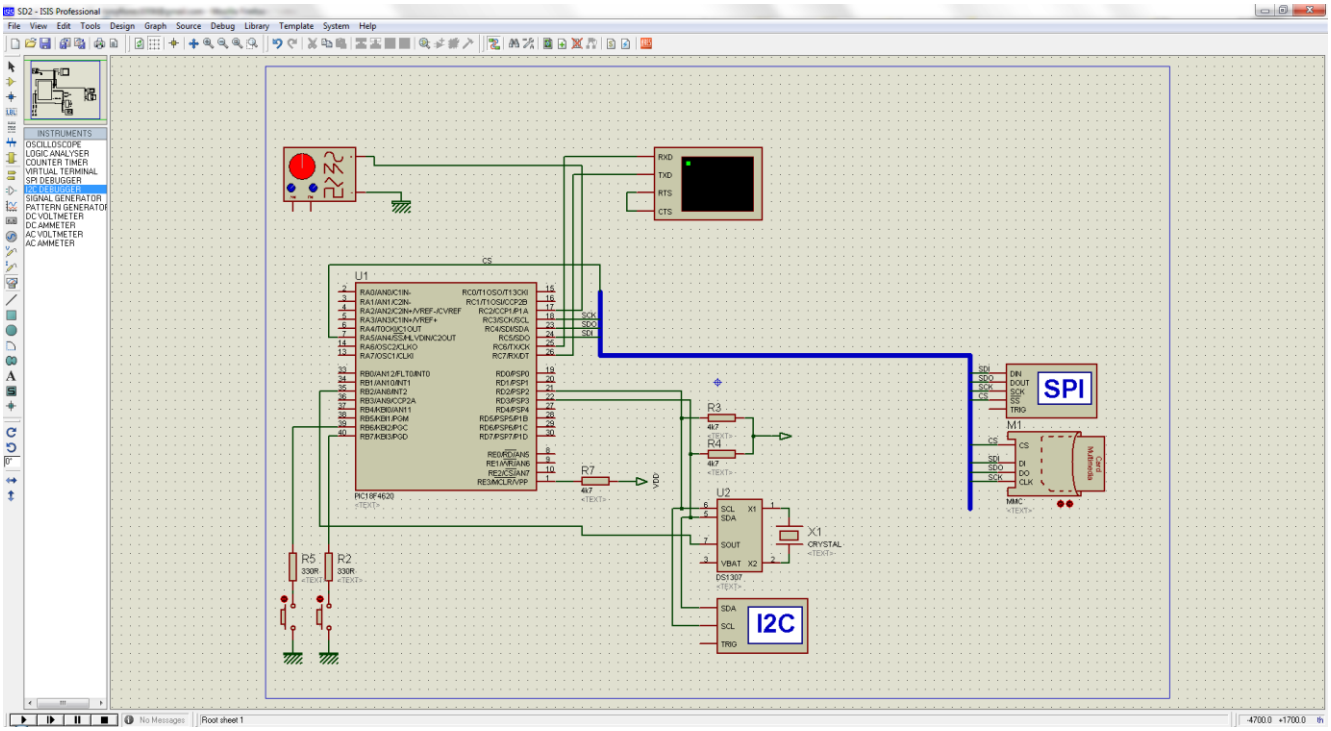


Figura 20. Entorn gràfic del programa ISIS (Proteus Suite)

3.4. Busos

3.4.1. Serial Peripheral Interface (SPI)

El bus SPI es un bus sèrie de 3 línies a través del qual s'envien paquets de 8 bits i és utilitzat majoritàriament per a la comunicació entre diferents circuits integrats en sistemes electrònics. Cada dispositiu dintre del bus pot rebre i enviar informació a l'hora (full duplex), degut a que 2 de les 3 línies son de dades, una per a enviar i l'altra per a rebre, i la tercera es el rellotge que governa les transferències de dades. Dintre del bus SPI els dispositius es defineixen com a mestres i esclaus, però només pot existir un mestre en cada transferència de dades, degut a que el dispositiu mestre es aquell que inicia la transferència de dades. El dispositiu mestre selecciona l'esclau al que enviarà les dades mitjançant una línia de chip select (CS), la qual cada esclau tindrà la seva, per tant, tots aquells dispositius esclaus que no hagin estat seleccionat hauran d'estar deshabilitats.

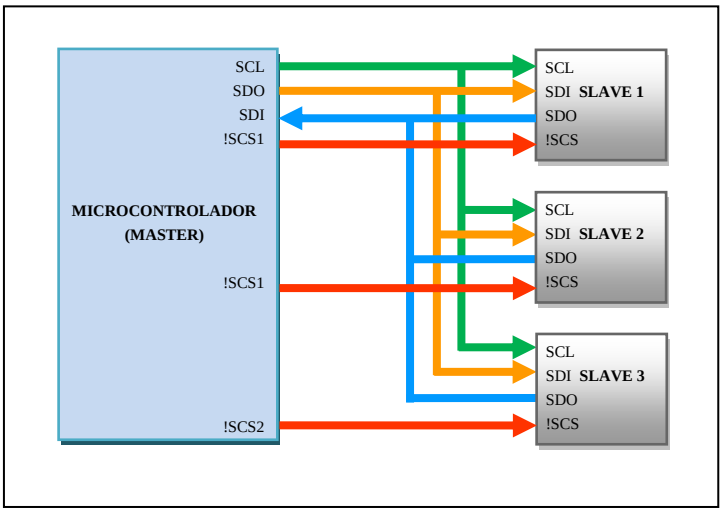


Figura 21. Estructura d'un bus SPI

La majoria d'interfases SPI porten 2 bits de configuració que són *Clock Polarity* o polaritat del rellotge i *Clock Phase* o fase del rellotge. La polaritat de rellotge configura l'estat inactiu o de repòs d'aquest, el qual pot ser a nivell baix o bé a nivell alt. La fase del rellotge es el moment en que s'envien les dades, que pot ser quan el rellotge passa de mode inactiu a mode actiu o viceversa. Aquests bits de configuració respecte el rellotge donen lloc als 4 modes del bus SPI. El mode del bus SPI ve donat per l'esclau, per tant, es imprescindible que el mestre estigui configurat en el ma mateix mode per tal de que aquests es puguin comunicar.

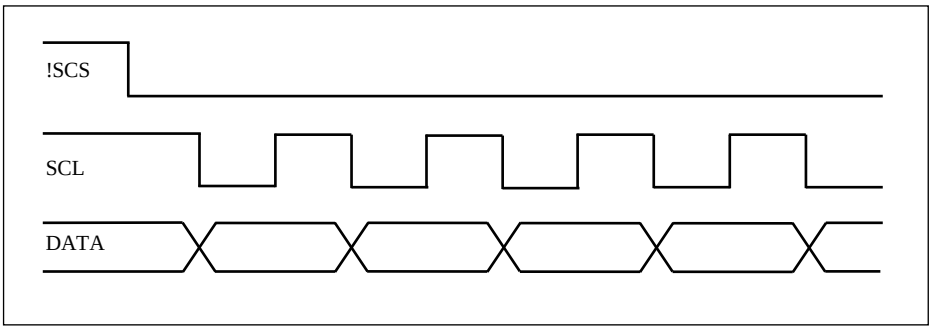


Figura 22. SPI Mode 1

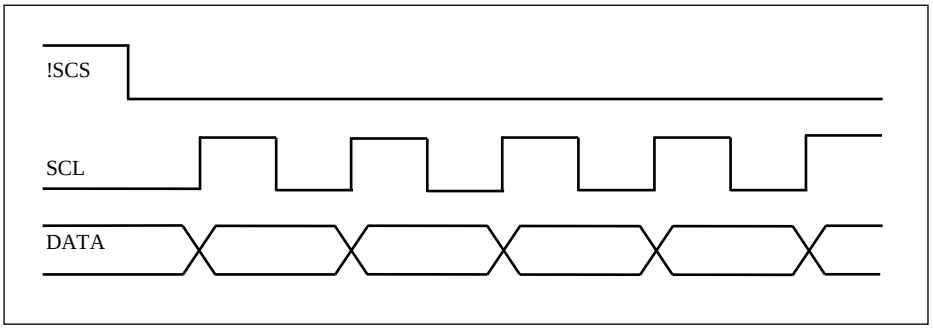


Figura 23. SPI Mode 2

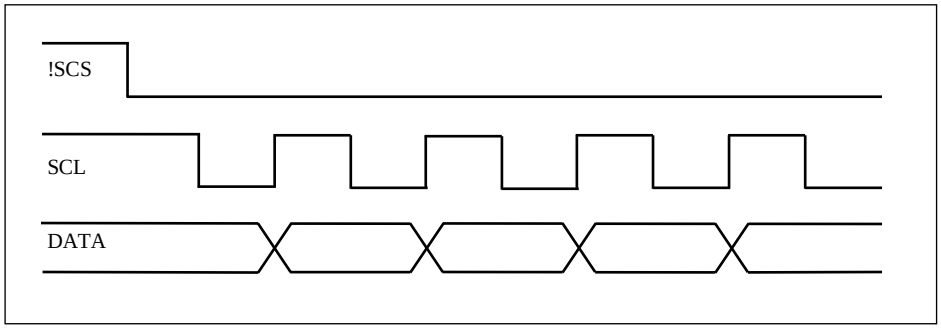


Figura 24. SPI Mode 3

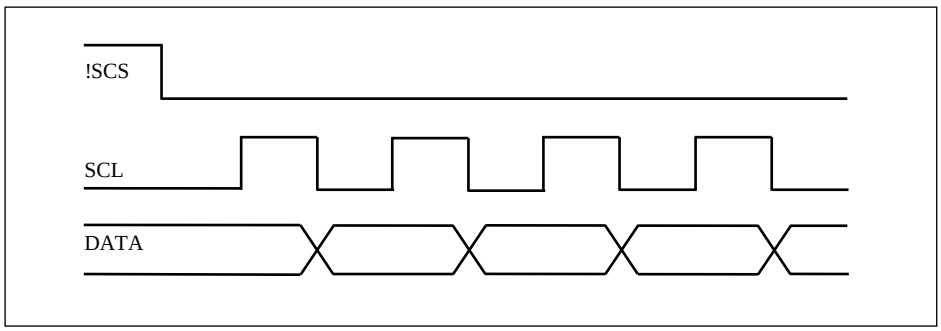


Figura 25. SPI Mode 4

3.4.2. Inter-Integrated Circuit (I^2C)

El bus I^2C es un bus sèrie ideat per Phillips Semiconductors i dissenyat per simplificar al màxim la interconnexió de microcontroladors amb els seus perifèrics i/o altres microcontroladors. Es ideal si no es requereix velocitats superiors als 100Kbits per segon, que és la seva taxa de transferència en mode estàndard.

El bus I^2C consta de dues línies bidireccionals, una per a dades (SDA) i una altra per al rellotge (SCL), que poden connectar diversos dispositius alhora mitjançant un hardware molt simple. Les línies SDA i SCL estan polaritzades a positiu amb unes resistències de pull-up, el valor de les quals depèn del nombre de dispositius connectats i la tensió d'alimentació.

Les comunicacions al bus I^2C es fan mitjançant comunicació mestre-esclau, podent existir alhora més d'un mestre dintre del bus, malgrat això, lo més típic es que només existeixi un únic mestre. El mestre es qui controla la senyal de rellotge i qui inicia i finalitza les transferències de dades, el qual generalment es un microcontrolador.

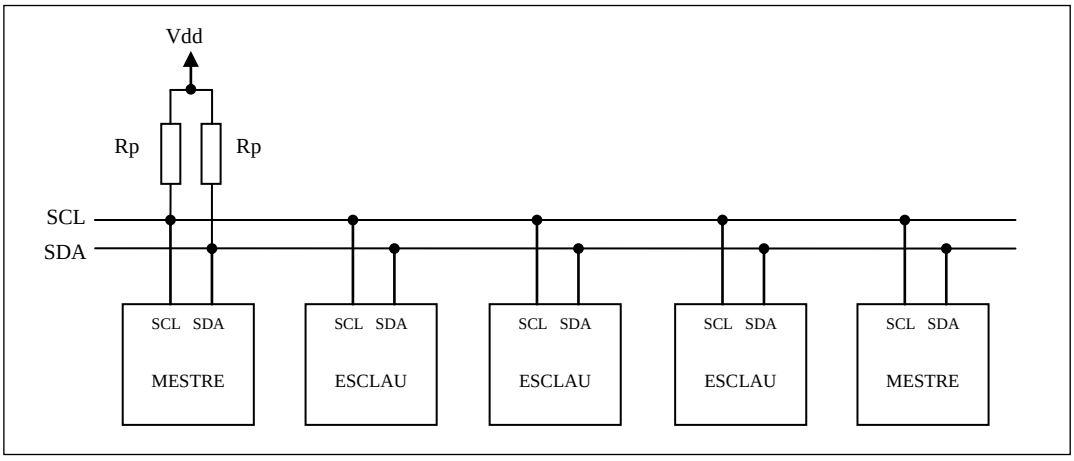


Figura 26. Estructura d'un bus I^2C

Cada esclau al bus I^2C es identificat per una direcció única que el diferència de la resta d'esclaus. Els dispositius compatibles amb I^2C solen acceptar la modificació d'aquesta direcció mitjançant 2 o 3 pins destinats a aquesta configuració.

Per iniciar i finalitzar una transferència de dades existeixen dos patrons que marquen aquestes condicions. La condició d'inici consisteix en el pas de nivell alt a nivell baix mentre SCL es troba a nivell alt, i un pas de nivell baix a nivell alt amb SCL a nivell alt marca la condició de final. Qualsevol altre canvi de la línia de dades SDA haurà de ser mentre SCL es troba a nivell baix.

A continuació es mostra un exemple de les condicions d'inici i finalització d'una transferència de dades a un bus I^2C:

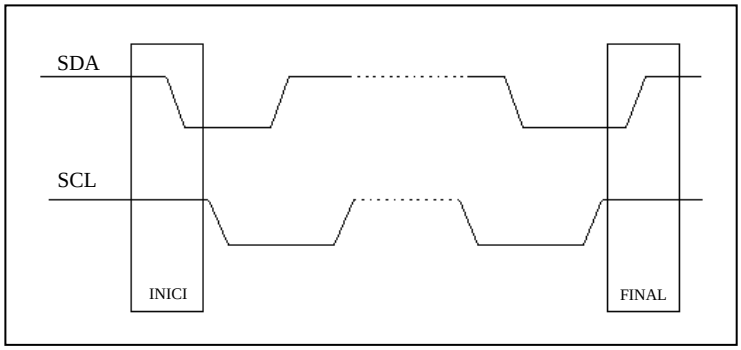


Figura 27. Condicions d'inici i final d'un bus I^2C

En cada transferència es transmet 1 bit cada pols de rellotge fins a completar els 8 bits que corresponen al byte enviat, i al novè pols el mestre deixa la línia SDA a alta impedància, per tal de que el receptor la baixi i la torni a deixar a alta impedància. Aquesta acció de l'esclau serà entesa com un acknowledge (ACK). Si l'esclau no allibera la línia SDA llavors forçarà al mestre a mantenir-se en un estat d'espera, i per tant no li enviarà més dades, mentre l'esclau acabi les seves operacions internes i alliberi la línia.

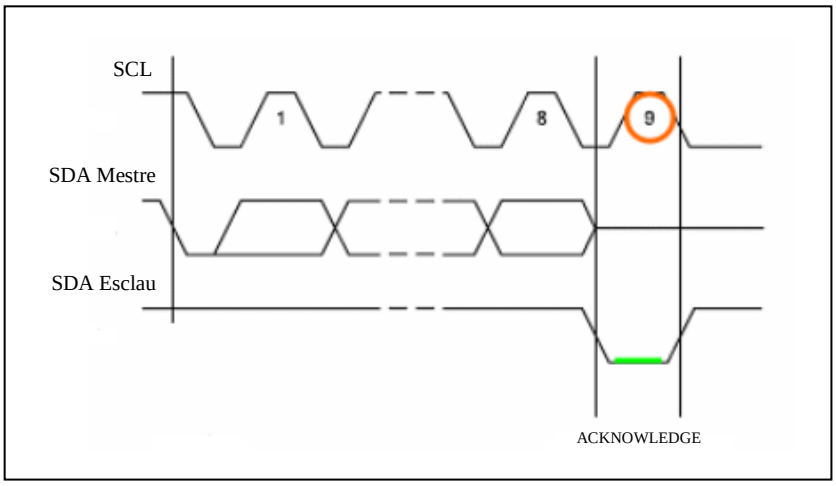


Figura 28. Confirmació o ACK enviat del esclau al mestre en un bus I^2C

3.4.3. Inter-Integrated Circuit (I2C) específic del display

El display utilitzat canvia respecte l'estàndard del bus. Un dels canvis més significatius d'aquest es la incorporació d'una tercera línia de control anomenada MRQ la qual serveix per a consultar si el esclau, en aquest cas el display, està disponible per a rebre dades, i per marcar a l'esclau quan se li enviaran dades per mostrar al display (l'enviament de la direcció I^2C del display no està inclosa).
Un altre canvi important són els temps, que són superiors als que marca l'estàndard, causant amb aquests una disminució en la velocitat d'enviament a les dades.

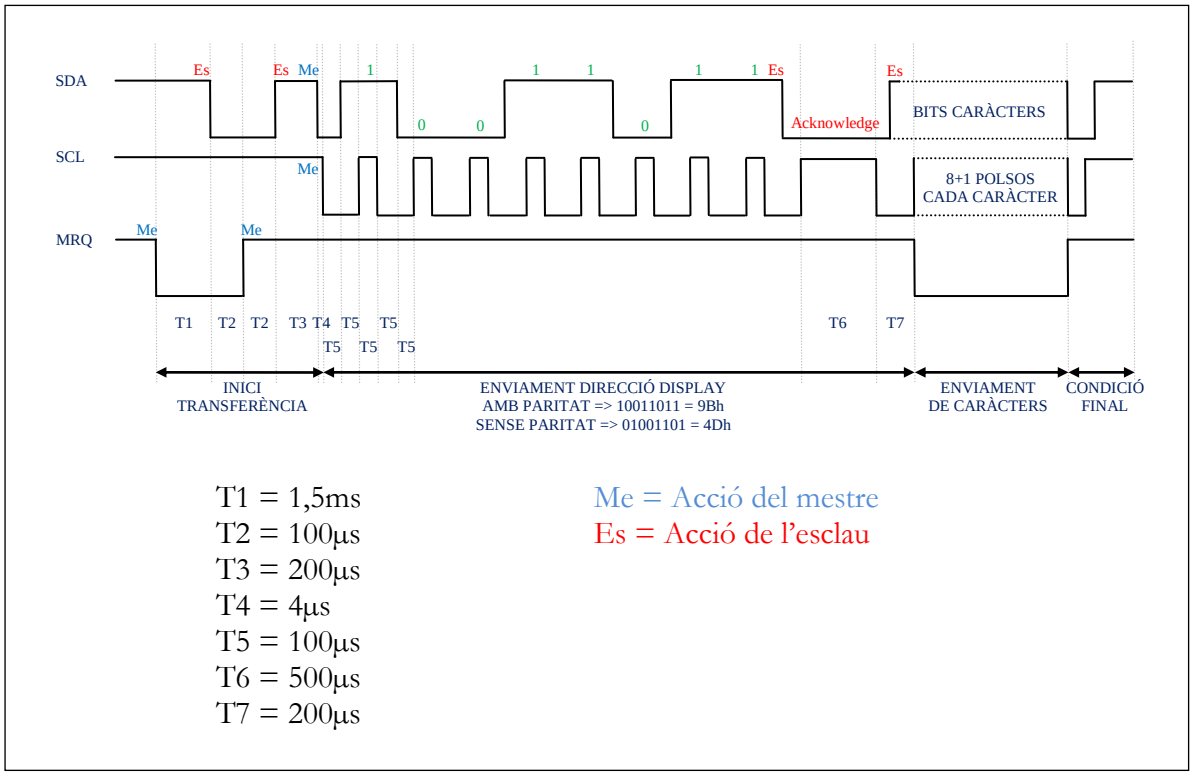


Figura 29. Enviament de dades al display mitjançant I^2C

Un detall que cal destacar del gràfic es que la duració del 9é pols (el destinat a que el esclau enviï l'acknowldge) dura 5 vegades més que l'estat alt de la resta de polsos. De no respectar la duració d'aquest, no funcionarà. A més dels canvis a nivell de timing, una de les especificacions del display és que requereix que se li enviï una seqüència anomenada "Power On Test" que consisteix en enviar alhora un pols baix de cadascuna de les 3 línies i a continuació un pols de cadascuna seguits. La seqüència sencera per a enviar dades consisteix en enviar primer la direcció del display, a continuació 3 bytes que determinen la configuració de les icones corresponents al radiocasset que s'il·luminaran (ràdio, cinta i CD), com per exemple la de "Stereo", "RDS", etc., i seguidament 10 bytes que es corresponen amb els deu caràcters del display.

3.5. Sistema de fitxers FAT16 de Microsoft

El sistema de fitxers FAT16 (File Allocation Table o Taula d'assignació d'arxius) es un sistema de fitxers basat en una llista que descriu l'estat dels clústers del disc. El clúster es una assignació de sectors de 512 bytes contigus i es la unitat més petita utilitzada per sistema operatiu.

Cada entrada de la taula d'assignació d'arxius es correspon amb un sòl clúster, en aquest es de 16 bits (FAT16), i conté en cas d'estar ocupat la direcció del següent clúster en un arxiu. En qualsevol altre cas, l'entrada pot contenir valors que indiquin l'estat específic del clúster, com ara els valors 000 (no utilitzat), FFF7 (clúster defectuós), FFF8 i FFFF (EOF).

Va ser el primer sistema de fitxers utilitzat en un sistema operatiu de Microsoft (la primera versió va ser la FAT12 destinada a disquets).

El sistema de fitxers FAT manté dues còpies de la taula d'assignació de fitxers per si es donés el cas que la primera quedés corrupta.

La partició més gran que pot suportar aquest sistema de fitxers es de 2GB, degut a que la mida màxima de clúster es de 32KB i les direccions a clúster són de 16 bits, per tant només podem direccionar:

$$Clusters\ direccionables = 2^{16} = 65536$$

I d'aquí deduïm la mida de partició màxima:

$$Espai\ màxim\ de\ partició = 32KB \cdot 65536 = 2097152KB = 2GB$$

El nombre màxim d'entrades que es pot tenir al directori arrel es de 512. La FAT emmagatzema els fitxers utilitzant 8 caràcters per al nom i 3 per a l'extensió. Emmagatzema les dates de creació, modificació i últim accés i atributs: Només lectura, ocult, sistema, arxiu i nom del volum. La FAT no emmagatzema permisos sobre els fitxers.

Un disc amb FAT està estructurat en quatre zones: zona reservada, zona de FAT, zona del directori arrel i zona de dades de directori i d'arxius.

Al primer sector del disc sempre es troba el BS (Boot Sector) el qual ocupa 11 bytes i dóna informació sobre l'inici del codi de boot. A continuació es troba el BPB (BIOS Parameter Block) que ens proporciona informació sobre l'estructura física del volum d'emmagatzemament.

4. Planificació

En aquest apartat s'explica tot el procés necessari per a realitzar el projecte. El projecte s'ha dividit en diverses etapes, les quals s'expliquen per separat i es situen en un marc temporal mitjançant un diagrama de Gantt.

4.1. Diagrama de Gantt

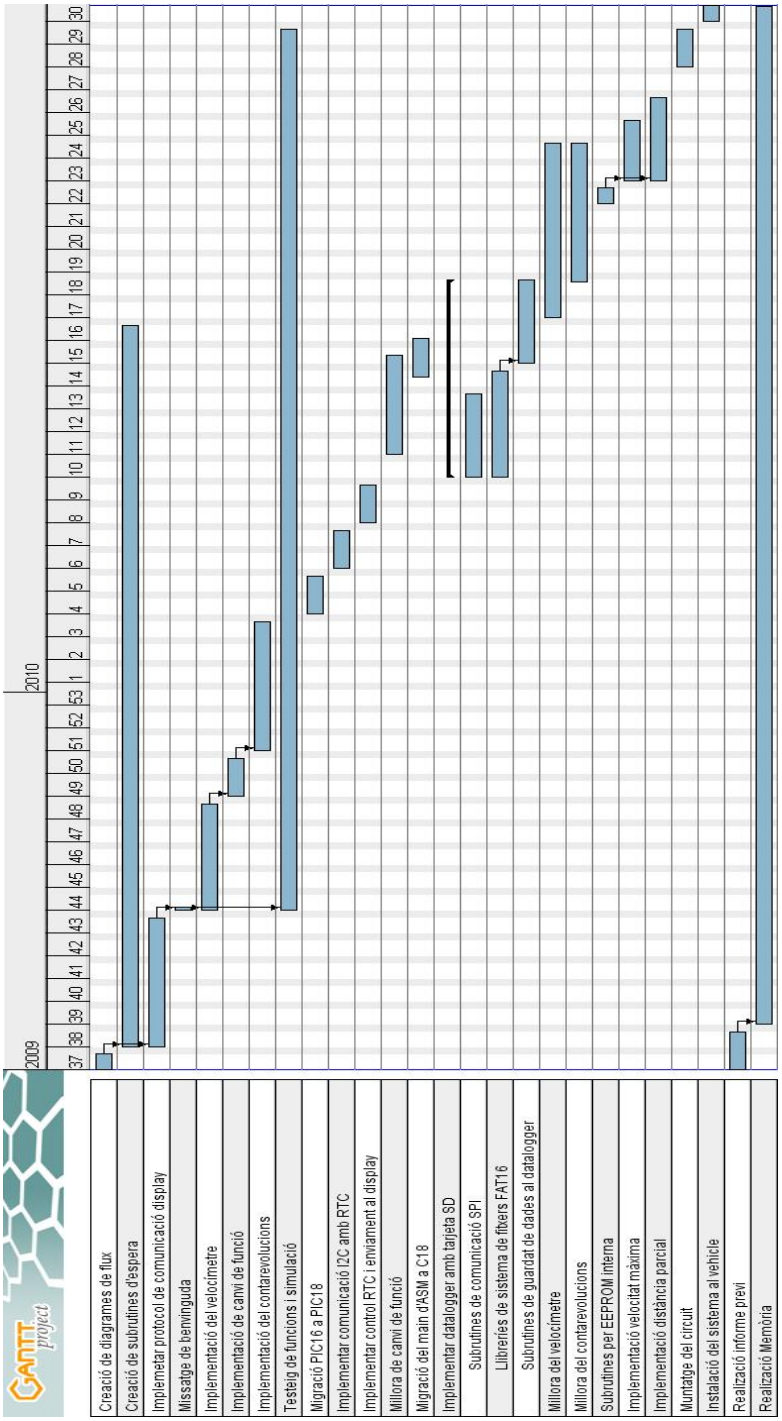


Figura 30. Diagrama de Gantt

4.2. Tasques realitzades

- Creació de diagrames de flux: Creació de diagrames de flux en paper per a basar el desenvolupament del codi en ells.
- Creació de subrutines d'espera: Creació de subrutines que introdueixen un temps d'espera de no operació dintre del flux del programa. Inclou la adaptació de les mateixes quan es va canviar el rellotge del sistema de 4MHz a 20MHz.
- Implementar protocol de comunicació display: Creació de les rutines necessàries per a poder enviar dades al display mitjançant el bus I^2C propietari que utilitza. Inclou el temps de investigació sobre el seu funcionament.
- Missatge de benvinguda: Creació de subrutines per enviar missatges utilitzant efecte scroll i subrutina d'enviament d'un missatge genèric quan s'inicia l'aplicació.
- Implementació del velocímetre: Implementació inicial del velocímetre mitjançant una subrutina d'espera i conteig de polsos amb un timer i conversió a ASCII per a enviar al display.
- Implementació de canvi de funció: Implementació inicial del canvi de funció basat en una interrupció externa que crea un flanc descendent en una pulsació del botó.
- Implementació del contarevolucions: Implementació inicial mitjançant una subrutina d'espera i conteig de polsos amb un timer, conversió a BCD i després a ASCII per a enviar al display.
- Testeig de funcions i simulació: Tasques de testeig mitjançant les eines disponibles a MPLAB com per exemple execucions pas a pas, analitzador lògic, etc. i simulacions amb la eina ISIS de la suite Proteus realitzades al llarg de tot el desenvolupament del codi.
- Migració de PIC16 a PIC18: Migració de tot el codi del PIC16F628A al PIC18F4620 a causa de la necessitat de més ports d'E/S, major capacitat, més memòria, etc. de cara a la implementació de les funcions del datalogger.
- Subrutines de comunicació SPI, subrutines bàsiques per a la utilització de la implementació hardware del SPI que incorpora el PIC18F4620 per a la comunicació amb la targeta de memòria SD.
- Llibreries de sistema de fitxers FAT16: Modificació i adaptació d'una llibreria FAT16 implementada en C18 per a PIC.
- Subrutines de guardat de dades al datalogger: Creació de subrutines que s'encarreguen d'utilitzar la llibreria FAT16 i crear el contingut del fitxer de LOG.

- Millora del velocímetre: Implementació del velocímetre amb el capturador que incorpora el PIC18 per tal de suprimir el temps de NOP que implicava l'anterior implementació. Inclou la gestió de timeouts amb les interrupcions de TMR1 i TMR2 per comprovar l'existència de senyal dels encoders i habilitacions i deshabilitacions de les interrupcions del capturador per a no saturar el microcontrolador.
- Millora del contarevolucions: Implementació del contarevolucions amb el capturador que incorpora el PIC18 per tal de suprimir el temps de NOP que implicava l'anterior implementació. Inclou la gestió de timeouts amb les interrupcions de TMR1 i TMR2 per comprovar l'existència de senyal dels encoders i habilitacions i deshabilitacions de les interrupcions del capturador per a no saturar el microcontrolador.
- Subrutines per EEPROM interna: Creació de subrutines per llegir, escriure i esborrar a la EEPROM interna del microcontrolador.
- Implementació velocitat màxima: Creació de codi necessari per a poder mantenir un registre de la velocitat lineal màxima assolida pel vehicle. Inclou la creació de subrutines de reset.
- Implementació de la distància parcial: Creació del codi necessari per a gestionar la distància parcial acumulada. Inclou la creació de subrutines de reset
- Muntatge del circuit: Muntatge del circuit en una placa de fibra de vidre i dibuixat amb soldador i estany de totes les pistes del circuit.
- Instal·lació del sistema al vehicle: Localització de les línies dels encoders a la centraleta (ECU) del vehicle i muntatge de la consola central amb el display original, maneta original i circuit amagat.
- Realització informe previ: Realització de l'informe previ a la realització del projecte.
- Realització de la memòria: Anotacions durant tota la realització del projecte i posterior redacció de la memòria.

5. Memòria Tècnica

5.1. Descripció general i requisits del sistema

El sistema desenvolupat ha de realitzar funcions de monitoratge de certs sistemes d'un vehicle a motor i utilitzar un display propietari integrat al vehicle per a presentar les dades, a més, també ha de ser capaç de realitzar un datalogger de les dades recollides dels encoders en una targeta SD amb sistema de fitxers FAT16 tenint en compte la data i l'hora. La creació d'un fitxer de dades en un sistema extraïble i amb format FAT16 facilitarà la tasca de portar aquestes dades a un PC convencional per tal de poder-les explotar.

En aquesta versió es disposa de dos enconders corresponents a la celeritat i a la velocitat del volant motor, els quals serviran per a mesurar aquests paràmetres, mostrar-los al display, i guardar-los a la targeta SD. L'encoder corresponent a la celeritat disposa d'una resolució de 32 divisions, i el de la velocitat del volant motor de 4 divisions.

La comunicació respecte el display propietari del vehicle es realitza mitjançant un bus I^2C amb diferencies respecte el bus I^2C estàndard, com per exemple per la incorporació d'una línia addicional anomenada MQR i o la variació dels timings del bus. El nostre microcontrolador ocuparà el lloc del equip de so original del vehicle, el qual utilitzava el display com a propi per a mostrar informació.

Com que tindrem diverses funcions, haurem de disposar d'una forma per poder interactuar amb el sistema, i això s'ha aconseguit mitjançant una maneta original de la marca del vehicle que incorpora botons per a interactuar amb l'ordinador opcional original de la marca, adaptant-la al nostre sistema.

Haurà de ser capaç de comunicar-se amb una targeta SD mitjançant Serial Peripheral Interface (SPI) implementat per hardware al microcontrolador, per tal de poder fer servir una llibreria del sistema de fitxers FAT16.

Per tal de poder tenir referència temporal a l'hora de crear dades al datalogger, haurem de disposar d'un RTC (DS1307), el qual es comunica mitjançant un bus I^2C estàndard, i així disposar de la data i l'hora en qualsevol moment.

Degut a que es tracta d'un sistema pensat per a ampliar les funcionalitats, haurà de ser fàcilment reprogramable, per aquest motiu disposa de una interfase ICSP (In circuit serial programming) mitjançant la qual es podrà reprogramar el firmware del microcontrolador sense extreure'l del circuit.

5.2. Hardware del sistema

El circuit final s'ha muntat sobre una placa de tops, reduint al màxim l'ús de cables. S'han fet servir circuits integrats, cristalls de quars, condensadors electrolítics i ceràmics, resistències, reguladors de tensió, diodes zener, diodes led, sockets per CI, socket per pila de liti i connectors.

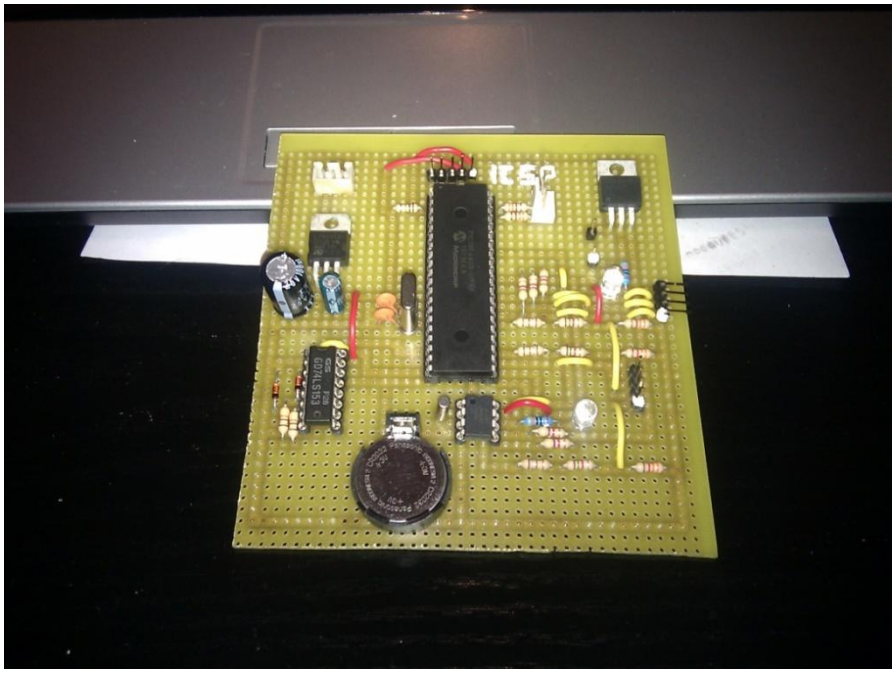


Figura 31. Vista principal del circuit definitiu

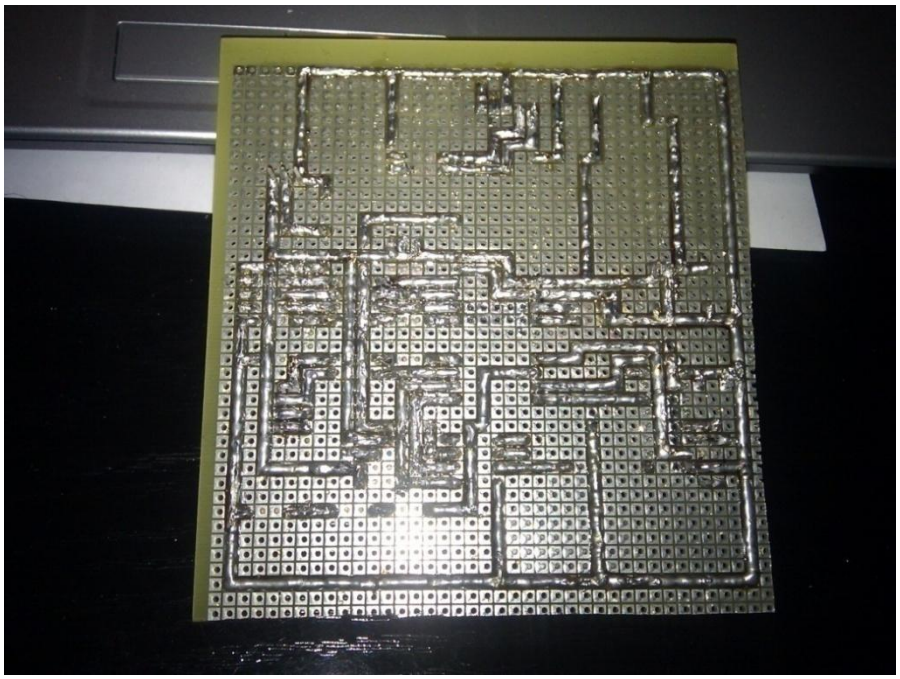
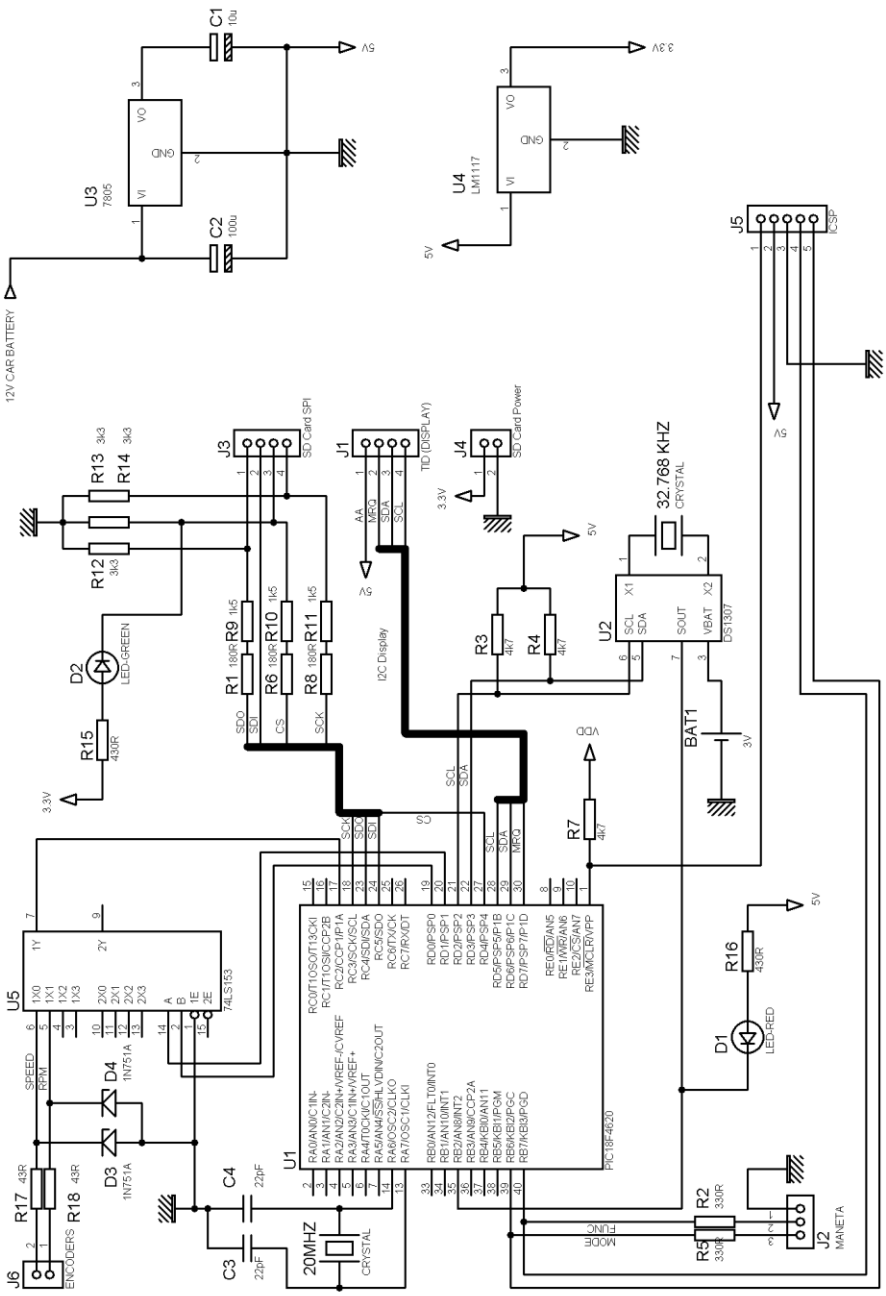


Figura 32. Vista posterior del circuit definitiu



PROJECTE: Sistema encastat de monitorització d'informació múltiple. Implementació per a vehicles.	
DATA: 05/07/2010	
AUTOR: Antonio Flores Ortega	
DESCRIPCIÓ: Esquema elèctric del circuit definitiu.	

Figura 33. Esquema elèctric de la placa

5.2.1. Alimentació del sistema

Aquest sistema està destinat a funcionar en un vehicle, per tant, alimentat per un alternador i espatllat per una bateria de 12V. Els circuits integrats que s'han fet servir en aquest sistema funcionen a una tensió mitja de 5V, tot i que poden treballar a tensions superiors i inferiors dintre d'un marge, i la targeta SD funciona amb una alimentació de 3,3V. Els valors de tensió necessaris per al correcte funcionament del circuit obliguen a transformar la tensió d'alimentació que subministra el vehicle als valors correctes, i per això s'han triat dos components del tipus estabilitzador de tensió, els quals admeten una tensió d'entrada de fins a 25V. Aquests components són respectivament el LM7805 i els LM1117, estabilitzadors de tensió de 5V i 3,3V respectivament. Degut a que la tensió d'alimentació que subministra el vehicle variarà entre els 11,5V i els 13,5V, i els microcontroladors són molt sensibles a les variacions en la tensió d'alimentació, s'han col·locat dos condensadors electrolítics, un a l'entrada del LM7805 i un altre a la sortida amb valors de càrrega 100μF i 10μF respectivament, per tal de absorbir les possibles variacions de tensió que es puguin produir i garantir així una estabilitat en la tensió d'alimentació del circuit.

Com ja s'ha comentat, la targeta SD treballa a uns valors de tensió de 3,3V, però el microcontrolador ho fa a 5V, això implica que les sortides d'aquest tenen una tensió de 5V que no pot arribar directament a la targeta SD, per tant s'ha muntat un divisor de tensió que garanteix el nivell correcte de tensió a les línies d'entrada de la targeta SD.

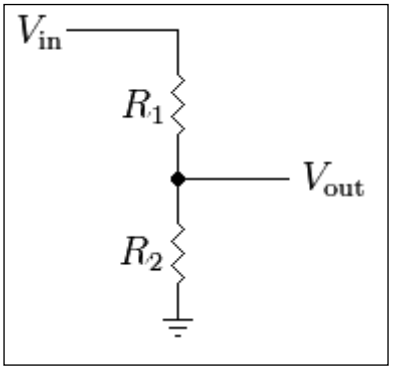


Figura 34. Divisor de tensió

R1 està compost per 2 resistències en sèrie de 1K5 Ω i 180 Ω respectivament, i R2 3K3 Ω, donant com a tensió de sortida:

$$V_{out} = \frac{R2}{R1 + R2} \cdot V_{in} = \frac{3300\Omega}{1680\Omega + 3300\Omega} \cdot 5V = 3{,}31V$$

Pel que fa l'alimentació del circuit integrat DS1307, s'ha afegit una bateria de liti de 3V com a alimentació de recolzament, per tal de que aquest pugui mantenir els valors de data i hora per un període de fins a 10 anys segons les especificacions del datasheet.

Referent a les entrades dels encoders, s'ha muntat una resistència i un diode zener de 5,1V en paral·lel i polaritzat invers per tal d'aconseguir un regulador senzill de tensió que pugui absorbir el s possibles pics que es puguin produir. Segons la intensitat màxima als ports del microcontrolador, les especificacions del diode zener i la tensió màxima teòrica d'entrada s'ha calculat el valor aproximat de la resistència:

$$R_s = \frac{13V - 5{,}1V}{0{,}2A + 0{,}005A} = 38{,}53\Omega \cong 42\Omega$$

5.2.2. Lector de targetes SD

Degut a la poca disponibilitat de lectors de targetes tipus SMD per soldar a placa que hi ha a les botigues convencionals d'electrònica es va decidir fabricar-ne un utilitzant un connector de disquetera de 5¼, que casualment té els contactes amb la mateixa separació que es de una targeta SD, tret dels pins 8 i 9 que no es fan servir en mode SPI (només es fan servir en mode SD).

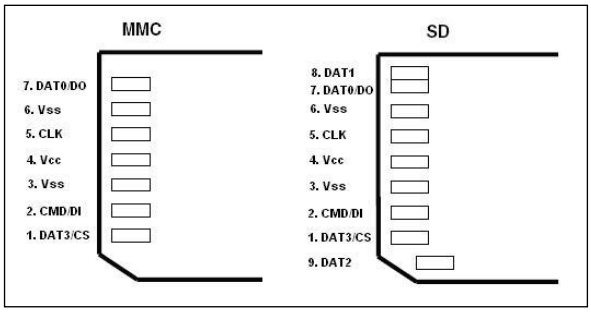


Figura 35. Esquema de targeta MMC/SD

Els cables del lector de targetes s'han unit de manera que la targeta SD es pot col·locar indiferentment de les dues posicions possibles que permet el connector.

Figura 36. Lector de targetes SD/MMC sense targeta

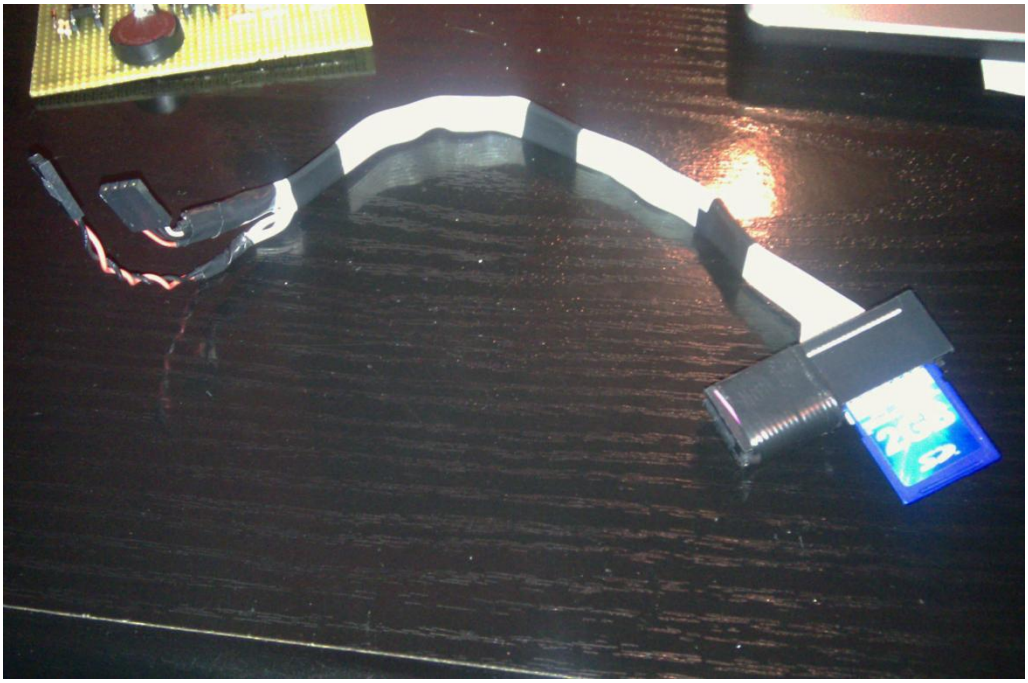


Figura 37. Lector de targetes SD/MMC amb una targeta SD

5.2.3. In Circuit Serial Programming (ICSP)

Aprofitant aquesta capacitat del microcontrolador PIC18F4620, s'ha afegit al circuit 5 pins per a poder programar el microcontrolador directament a la placa definitiva. El ICSP consta de 2 pins per a rellotge i dades i 3 pins per a alimentació, massa i voltatge de programació.

Una de les característiques més importants del ICSP es que dóna la possibilitat, en cas de ser un circuit destinat a comercialitzar, de poder muntar circuits i no programar els microcontroladors fins a la venta dels mateixos, donant així la possibilitat de distribuir els circuits sempre amb l'última versió del firmware. Dóna a més la possibilitat de actualitzar ràpidament el firmware en cas de ser circuits ja programats.

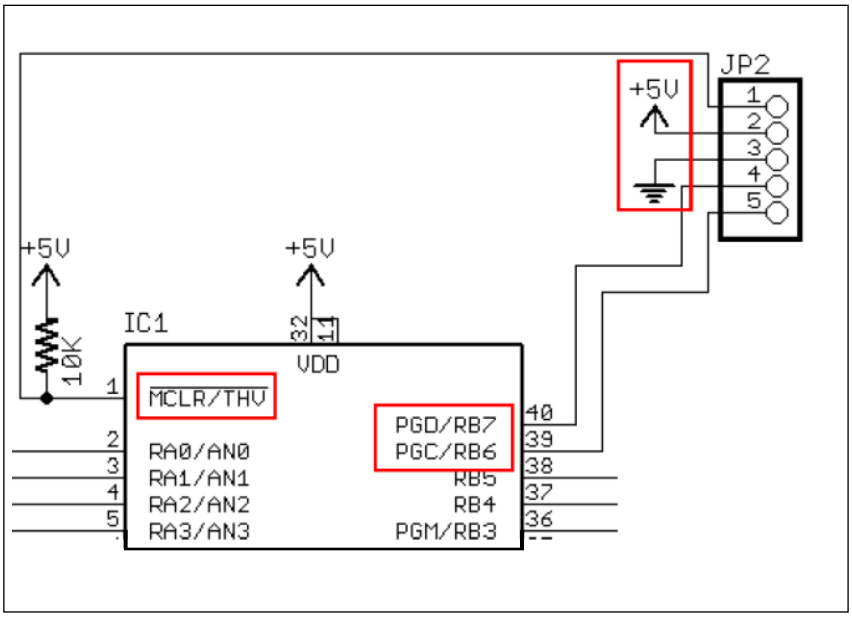


Figura 38. Esquema de connexió ICSP amb PIC18F4620

5.3. Estructura general del programa

El programa consta de diverses parts diferenciades per diferents fitxers de codi font. Les parts del programa son les següents:

- Rutina principal de l'aplicació (C18)

  En aquesta rutina es criden totes les subrutines d'inicialització pertinents, inicialització de registres per a configurar el comportament del microcontrolador, etc. Un cop s'ha inicialitzat tot, es passa a un bucle infinit el qual mitjançant una variable que marca la funció actual a la que ens trobem crida les subrutines pertinents. Aquesta variable s'incrementa segons les pulsacions de botó FUNC de la maneta utilitzada.

  Abans d'incorporar C18 al projecte, aquesta rutina estava creada en assemblador, i realitzava les mateixes funcions que en la versió definitiva.

- Subrutines d'enviament de dades al display (ASM)

  Aquestes subrutines s'encarreguen d'enviar les dades desitjades al display per tal de mostrar informació a l'usuari mitjançant el bus I^2C i tenint en compte totes les peculiaritats d'aquest display. A les versions abans d'incorporar el datalogger en C18

- Subrutines de Celeritat (ASM)

  En una primera instància, aquestes subrutines s'encarregaven de mesurar la celeritat segons els polsos rebuts, convertir aquest valor a codi ASCII, suprimir els ceros a l'esquerra i cridar les funcions necessàries per a mostrar al display la velocitat instantània. Posteriorment es va procedir a millorar l'obtenció de la velocitat instantània amb la utilització del mòdul de captura del microcontrolador, i per tant, es va suprimir la subrutina de càlcul de la celeritat del vehicle i es va derivar aquesta funcionalitat a les subrutines de tractament d'interrupcions.

- Subrutines de velocitat del volant motor (ASM)

  Al igual que les subrutines de celeritat, aquestes també es van millorar amb la utilització de capturador. Les subrutines de càlcul de velocitat del volant motor tenien alguna funció respecte les de la celeritat, ja que les revolucions per minut són de l'ordre dels milers (la celeritat es de l'ordre dels centenars). A la velocitat del volant motor per a facilitar la tasca de mostrar al display, el valor es passa a BCD i després a ASCII. En la versió definitiva aquestes últimes dues tasques i el càlcul de la velocitat del volant motor es realitzen també a les subrutines d'atenció a interrupcions.

- Subrutines de rellotge (ASM)

  Aquestes subrutines s'encarreguen de tractar amb el Real Time Clock, i realitzar la funcionalitat de canvi d'hora, de data i diferents funcionalitats específiques del rellotge.

- Subrutines DS1307 (ASM)

  S'encarreguen de comunicar-se amb el Real Time Clock. Realitza funcions d'inicialització, lectura i escriptura de registres.

- Subrutines I^2C (ASM)

  Subrutines de bus I^2C estàndard utilitzades per a la comunicació amb el Real Time Clock.

- Subrutines d'espera (ASM)

  Subrutines de espera amb temps específic per a la implementació del diferents bussos implementats per software, com per exemple el bus I^2C del display o el bus I^2C que utilitza el Real Time Clock.

- Subrutines de comunicació SPI (C18)

  Aquestes subrutines s'encarreguen de la inicialització del SPI i implementen les funcions bàsiques de lectura i escriptura.

- Subrutines comunicació amb la targeta SD (C18)

  Aquestes subrutines s'encarreguen de la inicialització específica de la targeta SD, comandes específiques i lectura i escriptura de blocs.

- Subrutines FAT16 (C18)

  Incorporen totes les funcions específiques del sistema de fitxers per realitzar operacions amb fitxers i directoris o crear-ne de nous.

- Subrutines EEPROM (C18)

  Implementa funcions bàsiques per llegir i escriure sobre la EEPROM interna de dades que incorpora el microcontrolador utilitzat.

- Subrutines de Velocitat Màxima del vehicle (C18)

  Aquests subrutines s'encarreguen de mantenir sempre actualitzada la celeritat màxima enregistrada i guardada a la EEPROM del microcontrolador. També s'encarreguen de realitzar reset del valor a petició de l'usuari. Per mostrar la velocitat màxima del vehicle s'han reutilitzat les subrutines encarregades de mostrar la velocitat instantània del vehicle al display.

- Subrutines de Distància (C18)

  S'encarreguen de mantenir sempre actualitzada i guardada la distància recorreguda, i de convertir-la a ASCII per tal de poder-la mostrar al display. La crida de la funció d'increment de la distància es fa desde les subrutines d'atenció a interrupció.

- Subrutines d'atenció a interrupció (C18)

  Inicialment es tractava d'un fitxer de codi en assemblador separat del codi principal. Amb el pas del codi principal a C18, també es van passar les subrutines d'atenció a interrupció, passant a formar part del mateix fitxer que conté el codi principal. Les funcions de les subrutines de atenció a interrupció son diverses, i van des de la crida de les funció que crea una entrada al datalogger, fins a la crida de subrutines d'actualització del rellotge.

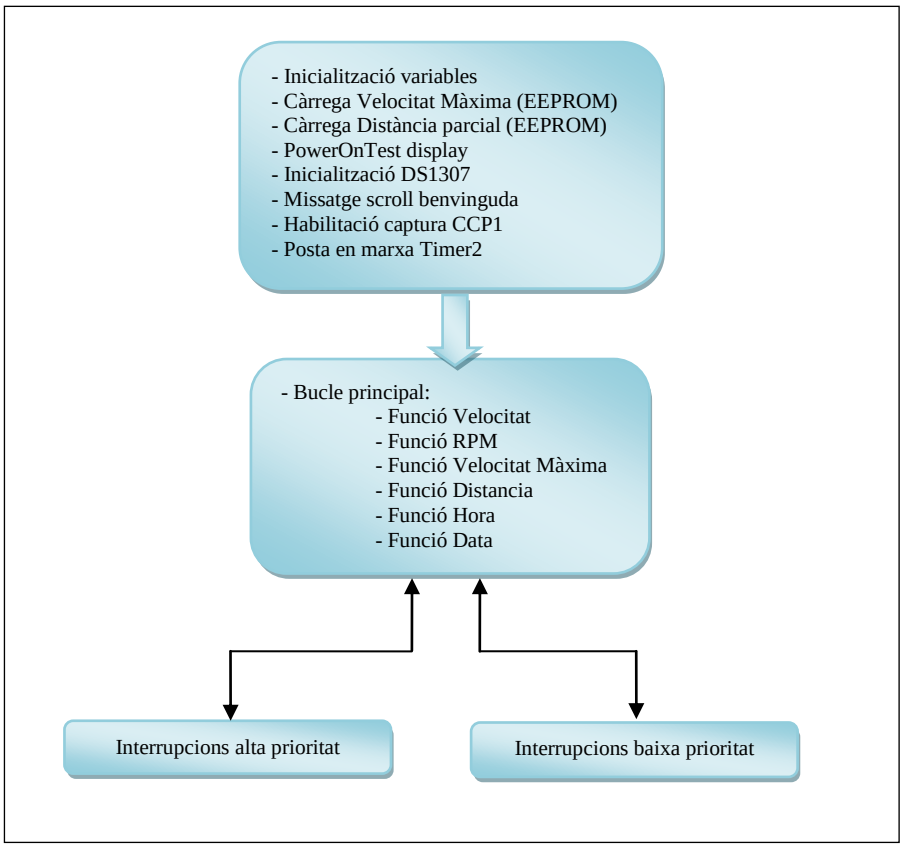


Figura 39. Diagrama general del programa

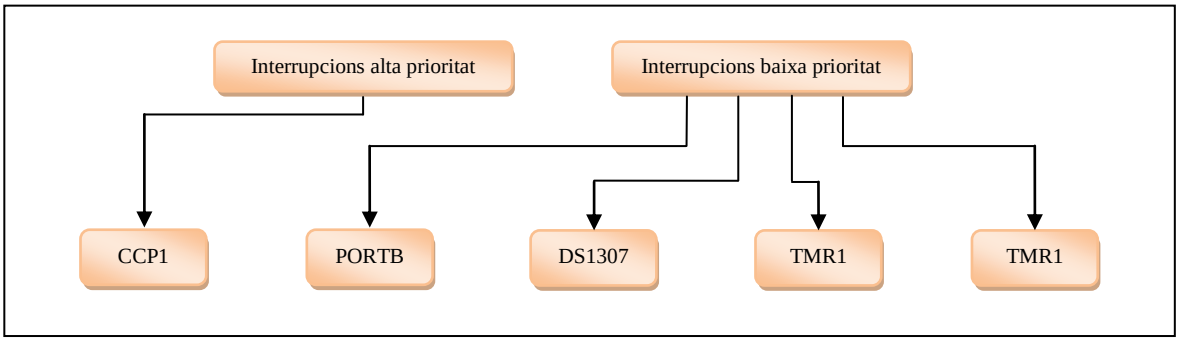


Figura 40. Diagrama general d'atenció a interrupcions

5.4. Inicialització

La inicialització es la primera secció de codi en ser executada. En aquesta part del codi es realitzen les inicialitzacions pertinents de registres, variables i dispositius per a poder passar al funcionament normal del sistema.

Primer de tot s'inicialitzen tots els registres que afecten a la configuració del microcontrolador, com per exemple els que habiliten les interrupcions que necessitem, les prioritats de les mateixes, la configuració dels ports com a entrada o sortida segons necessitat, etc.

A continuació s'inicialitzen les variables globals com la de funció actual, la de velocitat del volant motor, la de celeritat, els comptadors d'overflows i els Timers 1 i 2. En quest moment es llegeix de la EEPROM l'últim valor corresponent a la celeritat màxima (expressada com a velocitat màxima) i la distància parcial, i es carrega aquests valors a les variables globals pertinents.

Un cop s'ha inicialitzat s'habiliten les interrupcions del capturador i es passa a l'execució normal dintre del bucle principal de l'aplicació.

5.5. Enviament de text al display

El display consta de 10 caràcters al quals podem mostrar qualsevol caràcter del codi ASCII. Per tal de poder mostrar missatges amb o sense efecte scroll, s'han creat 10 variables de tipus byte, un per a cada caràcter del display, d'aquesta manera, per mostrar un missatge de com a màxim 10 caràcters serà suficient amb carregar amb el valor ASCII corresponent cadascuna de les variables associades al display. En aquest projecte els texts en scroll apareixen per la part dreta de la pantalla i desapareixen per l'esquerra. Per realitzar aquest efecte es disposa d'una funció que cada cop que s'afegeix un caràcter, copia tots els caràcters al lloc del caràcter anterior, deixant lliure l'últim, al que es copia l'últim caràcter afegit. Un cop carregades totes les variables de caràcter s'envia el text al display. Aquesta funció precisa que es carreguin prèviament totes les variables del display amb el valor ASCII del espai en blanc, i si es vol que el missatge desaparegui totalment, s'hauran d'afegir els caràcters d'espai en blanc pertinents per a poder treure tots els caràcters mostrats al display per la part esquerra. El funcionament es idèntic a realitzar un desplaçament a l'esquerra sobre un registre sense utilitzar el bit de carreteig.

5.6. Rellotge de temps real

El rellotge s'actualitza al display cada segon degut a que el CI genera una interrupció cada segon, i cada cop que es produeix aquesta es fa una crida a la funció que envia les dades al display.
El funcionament normal del rellotge simplement llegeix els registres pertinents del DS1307 i col·loca els valors a les variables destinades al rellotge. Fet això, segons la funció seleccionada (hora o data) es transformen els valors pertinents de BCD a ASCII i s'envien al display.
El rellotge pot funcionar en mode ajustar, i en aquest cas, cada cop que es crida la funció per mostrar hora o data s'alterna el flanc de la interrupció del rellotge (ascendent o descendent), fent així que la interrupció sigui cada 500 ms. En aquest mode assignem cada dígit del rellotge o de la data a un bit d'una variable de 8 bits que rotem cada cop que acabem d'ajustar un dígit per tal de passar al següent, sortint del mode ajustar quan la mateixa arriba a valor 0. Quan finalitza l'ajust escriu els nous valors al DS1307, quedant el rellotge establert a l'hora desitjada. A cada interrupció alternem el dígit que estem ajustant entre el valor i un espai en blanc, aconseguint així el parpelleig del dígit en ajust.

5.7. Càlcul de la velocitat (celeritat) màxima

El càlcul de la velocitat es realitza a partir de varies mostres. Es guarden dues mostres de la celeritat instantània, i a la tercera es comparen les tres. Com aquests tres valors es prenen en un interval de temps molt curt, seleccionem com a velocitat màxima el valor més petit de les tres, evitant així pics falsos de velocitat. Cada cert temps, aquest valor de velocitat màxima es compara amb la velocitat màxima emmagatzemada a la EEPROM interna del microcontrolador, substituint aquesta en cas de que el valor a la EEPROM sigui inferior. Juntament amb les funcions de tractament de la velocitat màxima existeix una funció que substitueix directament el valor de la EEPROM per un cero.

5.8. Estimació de la distància parcial

La distància parcial recorreguda no es calcula amb gran precisió, ja que la idea es utilitzar-la per a calcular un valor de velocitat mitjana del recorregut. Al final per motius de temps no s'ha pogut implementar la velocitat mitjana.

La estimació de la distància es realitza guardant cada segon la velocitat instantània cada segon, assumint que durant tot el segon anterior a la mostra de velocitat instantània la celeritat no ha variat. El valor de la distància en metres es guarda en una variable de 4 bytes, i al igual que la velocitat màxima es va guardant a la EEPROM interna, amb la diferència de que en aquest cas, com que la distància es incremental, no s'han de realitzar comparacions, simplement es substitueix el valor anterior. La distància parcial també disposa d'una funció per a resetejar-la, substituint el valor guardat a la EEPROM per 0.

5.9. Datalogger

El datalogger s'encarrega de guardar les dades necessàries a la targeta SD. Per realitzar aquesta tasca es necessari que les variables que contenen els valors a guardar estiguin actualitzades. Degut a que guardar les dades a la targeta SD es una operació costosa, mantenim un buffer a la memòria RAM del microcontrolador i desem les dades a la targeta quan aquest buffer està ple al màxim. Mantenir aquest buffer implica a més a més poder omplir al màxim els sectors de la targeta SD que son de 512 bytes.

La cadena de text que es guarda al log es munta byte a byte en comptes de utilitzar funcions d'impressió a streams que porta el llenguatge C18, ja que avaluant les dues opcions en cicles d'instrucció a la simulació, la opció d'impressió era una gran quantitat de vegades més costosa.

Els valors que finalment van a parar al log són la data, la hora, velocitat instantània o celeritat i revolucions per minut. El format és el següent: "HH:mm:ss dd/mm/yy /t ddd /t dddd". Les dades estan separades entre sí pel valor "/t" que correspon a una tabulació, de manera que es poden interpretar com a diferents columnes a un full de càlcul.

10/06/2010 13:15:00	0	900
10/06/2010 13:15:01	20	2500
10/06/2010 13:15:02	33	1200
10/06/2010 13:15:03	45	1500
10/06/2010 13:15:04	57	1100
10/06/2010 13:15:05	68	1400
10/06/2010 13:15:06	79	1900
10/06/2010 13:15:07	87	2400
10/06/2010 13:15:08	94	2900
10/06/2010 13:15:09	101	1300
10/06/2010 13:15:10	108	1450
10/06/2010 13:15:11	113	1570
10/06/2010 13:15:12	119	1720
10/06/2010 13:15:13	125	1800
10/06/2010 13:15:14	120	1750
10/06/2010 13:15:15	122	1770
10/06/2010 13:15:16	123	1780
10/06/2010 13:15:17	120	1750
10/06/2010 13:15:18	122	1770
10/06/2010 13:15:19	125	1800
10/06/2010 13:15:20	127	1820
10/06/2010 13:15:21	123	1780
10/06/2010 13:15:22	118	1710
10/06/2010 13:15:23	121	1760
10/06/2010 13:15:24	124	1790
10/06/2010 13:15:25	122	1770
10/06/2010 13:15:26	120	1750
10/06/2010 13:15:27	119	1720
10/06/2010 13:15:28	122	1770
10/06/2010 13:15:29	120	1750
10/06/2010 13:15:30	123	1780

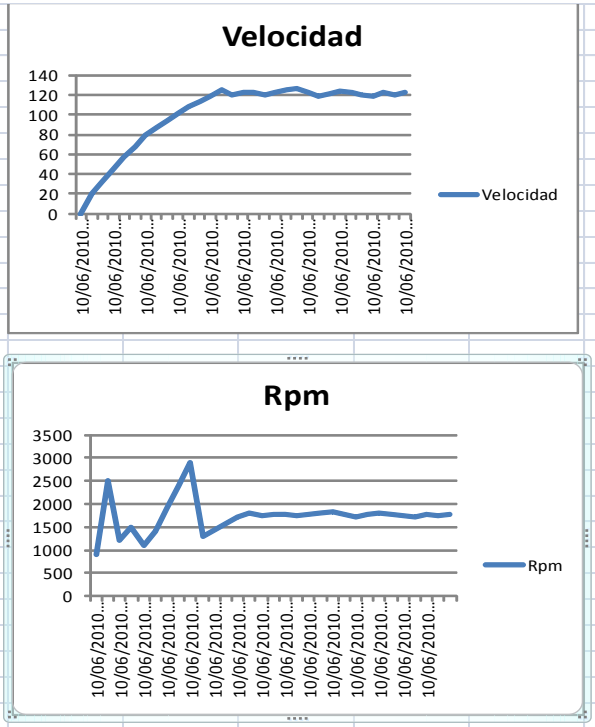


Figura 41. Exemple 1 d'explotació de dades del datalogger

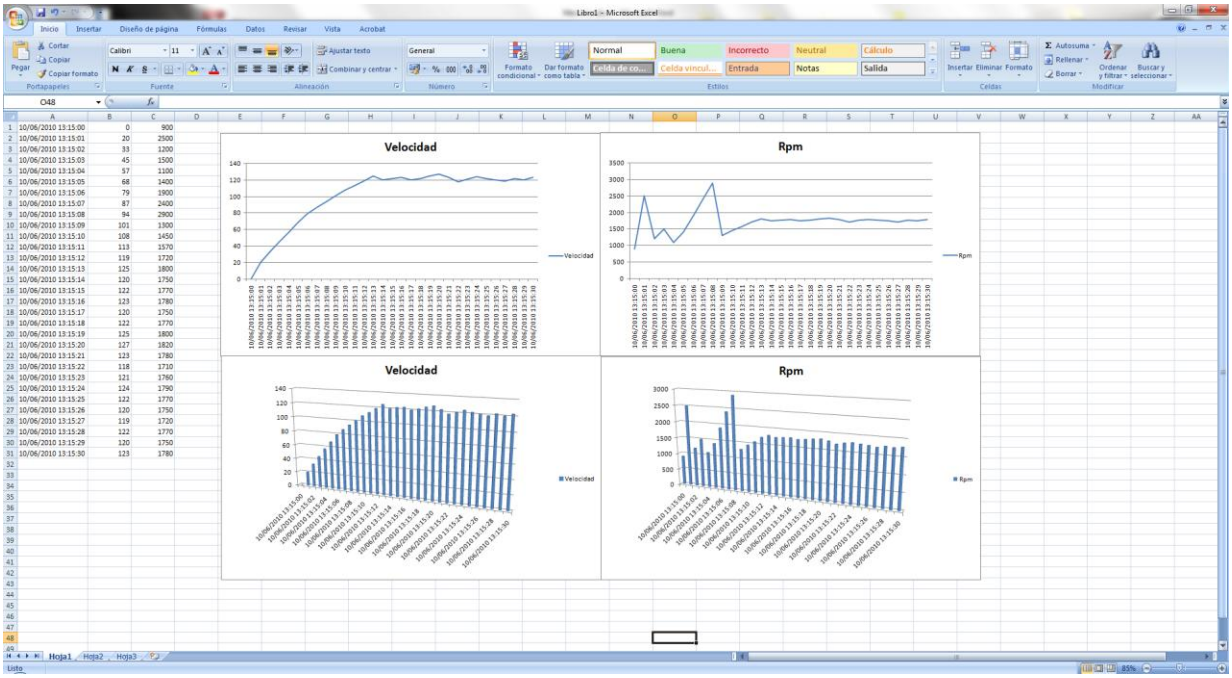


Figura 42. Exemple 2 d'explotació de dades del datalogger

5.10. Interrupció del CCP1

El CCP1 o "Capture/Compare/PWM 1" està associat al Timer 1, i produeix una interrupció al microcontrolador cada cop que rep 16 flancs de pujada al port corresponent (Pin núm. 17).
Amb la intenció de utilitzar el mínim possible de recursos del microcontrolador, aquest mòdul s'ha multiplexat per obtenir el valor de la velocitat instantània i la velocitat del volant motor amb la incorporació d'un multiplexor a l'entrada. Com que el càlcul de aquests valors es molt important s'ha donat prioritat alta a l'atenció d'aquesta interrupció.
Al final de cada operació s'alterna l'entrada del multiplexor i es desactiven les interrupcions del CCP1. Les interrupcions del CCP1 es tornen a habilitar en el tractament de la interrupció del overflow al Timer2 amb una freqüència de 200ms aproximadament, aconseguint així que s'obtinguin mostres de velocitat instantània i de velocitat del volant motor cada 400ms, i per tant evitant excessives interrupcions que acabarien per executar-se una rere l'altre evitant que es pugui executar el flux principal amb normalitat.

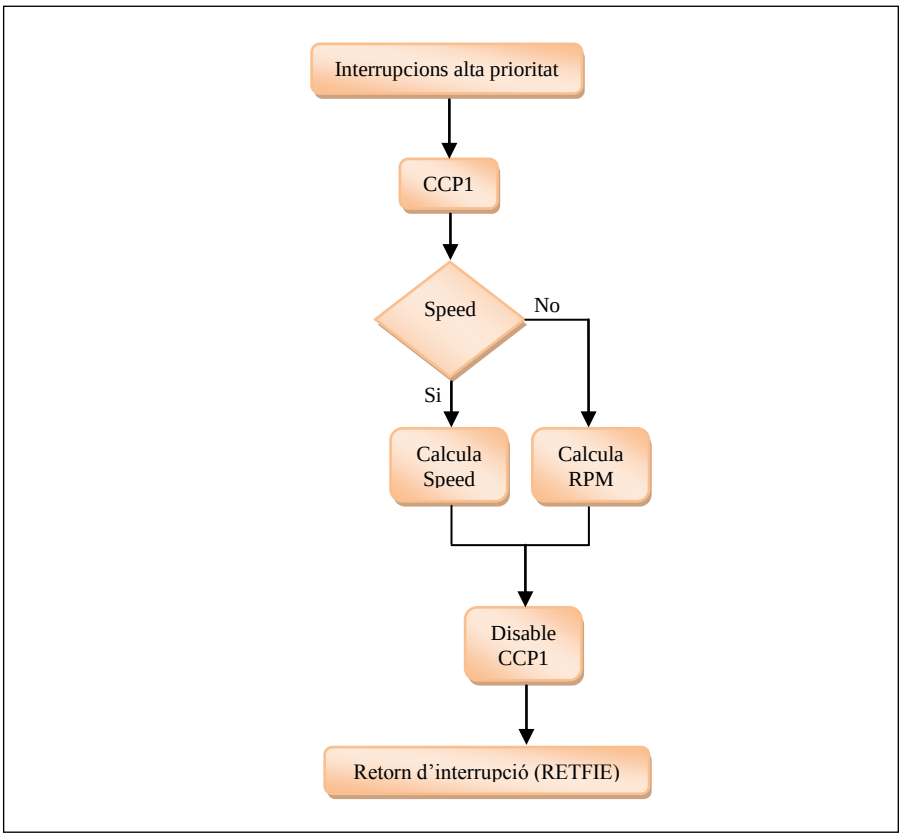


Figura 43. Diagrama d'atenció a interrupcions del CCP1

5.10.1. Càlcul de la velocitat lineal instantània

Per tal de calcular la velocitat lineal instantània del vehicle, el primer pas es mesurar la longitud de la circumferència de la roda:

$$Roda = 205\ 50R16$$

$$Diametre\ llanta\ (mm) = Llanta\ (pulsades) \cdot \frac{25{,}4mm}{1\ pulsada} = 16 \cdot 25{,}4 = 406{,}4mm$$

$$Diametre\ goma\ (mm) = Amplada(mm) \cdot \frac{Perfil}{100} = 205 \cdot \frac{50}{100} = 102{,}5mm$$

$$Diametre\ roda(mm) = Diametre\ llanta(mm) + [2 \cdot Diametre\ goma(mm)]$$

$$Diametre\ roda = 406{,}4 + 2 \cdot 102{,}5 = 611{,}4mm$$

$$Long.circumf.roda = 2\pi \cdot {}^{611{,}4}/_{2} = 1920{,}77mm = 1{,}92m$$

Un cop tenim la longitud de la circumferència de la roda podem continuar amb els càlculs. Considerem:

$$\varphi_0 = 0 \ \rightarrow \ \omega = \frac{\varphi}{t}$$

Sabent que la velocitat lineal es $v = \omega \cdot r$ deduïm:

$$\frac{v}{r} = \frac{\varphi}{t} \ \rightarrow \ v = \frac{r \cdot \varphi}{t}$$

La posició angular en el sistema internacional es mesura en radians i en aquest cas es necessita mesurar el número de voltes de la roda apliquem un factor de conversió:

$$\varphi = Num.voltes \cdot 2\pi$$

I substituïm:

$$v = \frac{r \cdot Num.voltes \cdot 2\pi}{t}$$ i com que $$Long.circumf. = 2\pi r$$

Ens queda:

$$v = \frac{Num.voltes \cdot Long.circumf.}{t}$$

Com que la resolució de l'encoder de la velocitat lineal instantània es de 32 divisions:

$$Num.voltes = \frac{Num.polsos}{32}$$

Substituïm:

$$v = \frac{\frac{Num.polsos}{32} \cdot Long.circumf.}{t}$$

Degut a que en les fórmules anteriors la velocitat estan en m/s, apliquem un factor de conversió per a passar-la a Km/h i imposem que la velocitat en Km/h es igual al nombre de polsos, aconseguint així simplificar el càlcul de la velocitat lineal instantània, i obtenint el temps necessari perquè aquesta condició es compleixi:

$$Num.polsos = v(^{m}/_{s}) \cdot \frac{1Km}{1000m} \cdot \frac{3600s}{1h}$$

$$v = \frac{Num.polsos}{3{,}6}$$

Igualem:

$$\frac{Num.polsos}{3{,}6} = \frac{\frac{Num.polsos}{32} \cdot Long.circumf.}{t}$$

I simplifiquem:

$$t = \frac{Long.cirmcumf \cdot 3{,}6}{32} = \frac{1{,}92 \cdot 3{,}6}{32} = 0{,}216s$$

Per tal de calcular la velocitat lineal instantània, si es compten els polsos generats per l'encoder durant un període de 0.216s el valor obtingut coincideix exactament amb la velocitat instantània mesurada en Km/h, per tant si multipliquem aquest temps d'espera per la freqüència dels cicles d'instrucció del microcontrolador dividida pel valor del divisor preescalar i multiplicat per 16 a causa de la condició de captura, obtenim:

$$FOSC/_4 = 20MHz/_4 = 5000000\ Hz$$

$$Freq.Instruc.(amb\ preesc.) = 5000000/_8 = 625000\ Hz$$

$$Temps \cdot Freq.Instruc. \cdot 16 = 0{,}216 \cdot 625000 \cdot 16 = 2160000$$

I a continuació, per a obtenir la velocitat instantània realitzem el següent càlcul:

$$Vel.inst. = \frac{2160000}{Cicles\ TMR1 + (65536 \cdot Overflows\ TMR1)}$$

5.10.2. Càlcul de la velocitat del volant motor (RPM)

Per a calcular la velocitat del volant motor hem de tenir en compte que la resolució del encoder està marcada pel nombre de cilindres del motor. En aquest cas el nombre de cilindres es de 4, i per tant sabem que cada 4 polsos tenim una rotació del volant motor.
Per tal d'obtenir la velocitat del volant motor en revolucions per minut (rpm), i tenint en compte la resolució de l'encoder, utilitzem aquest càlcul:

$$Rpm = \frac{Duració\ pols \cdot 60s}{4} = Duració\ pols \cdot 15$$

La freqüència mínima que tindrà el senyal de l'encoder del volant motor es de 56,64Hz que correspon a 849,6 rpm, i és quan el motor es troba en ralentí.

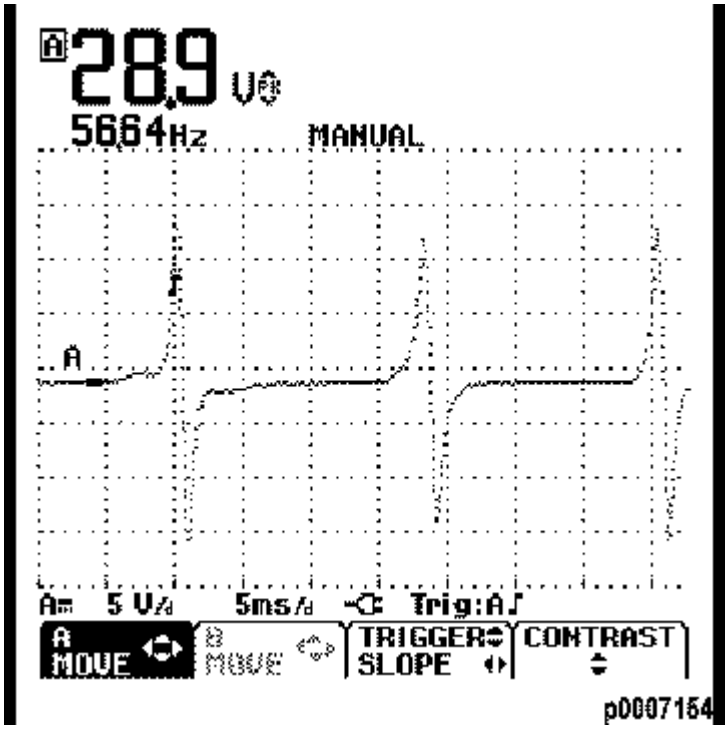


Figura 44. Senyal de l'encoder del volant motor

Establim 100ms com a temps per a contar els polsos (al ralentí tindríem 5,664 polsos), i el valor resultant anirà multiplicat per 10 a la formula descrita anteriorment. Si multipliquem el temps d'espera (100ms) per la freqüència dels cicles d'instrucció del microcontrolador dividida pel valor del divisor preescalar i multiplicat per 16 a causa de la condició de captura, obtenim:

$$FOSC/_4 = 20MHz/_4 = 5000000\ Hz$$

$$Freq.Instruc.(amb\ preesc.) = 5000000/_8 = 625000\ Hz$$

$$Temps \cdot Freq.Instruc. \cdot 16 = 0{,}1s \cdot 625000Hz \cdot 16 = 10000000$$

I aplicant a la fórmula per a calcular la velocitat del volant motor:

$$Rpm = \frac{10000000}{Cicles\ TMR1 + (65536 \cdot Overflows\ TMR1)} \cdot 15$$

$$Rpm = \frac{150000000}{Cicles\ TMR1 + (65536 \cdot Overflows\ TMR1)}$$

5.11. Interrupció per canvi d'estat al PORTB

Les interrupcions per canvi d'estat es produeixen quan es detecta un canvi d'estat en els pins RB4:RB7. En aquest projecte només tenim en compte el canvi d'estat en els pins RB6 i RB7 que es corresponen respectivament amb els polsador MODE i FUNC, els quals comuniquen a massa amb una resistència per tal de canviar l'estat de alt a baix durant la pulsació.

Sempre que es detecta un canvi d'estat s'ha de comprovar les dues entrades per tal de conèixer quin es el polsador causant de la interrupció.

Per tal de evitar falses interrupcions, després de detectar un canvi d'estat esperem 20ms per a que s'estabilitzi el nivell de tensió abans de comprovar quin polsador ha generat la interrupció. Aquest fenomen es coneix com a rebot del senyal, i sempre està present als contactors mecànics. Una altra forma de absorbir els rebots de senyal seria per hardware en comptes de per software, amb la incorporació d'un condensador.

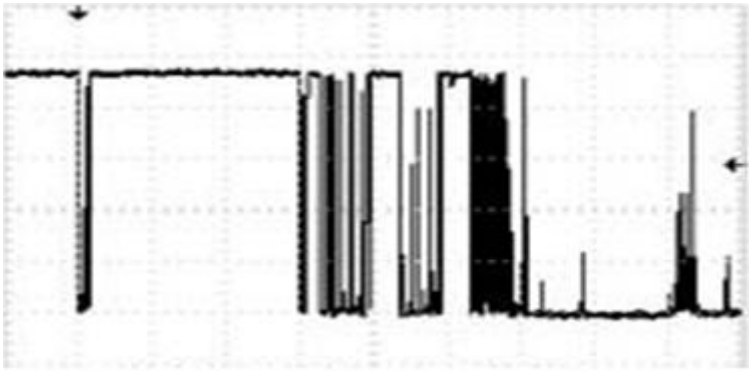


Figura 45. Rebots del senyal a un polsador

Quan es detecta una interrupció del polsador FUNC, es comprova si el microcontrolador es troba en estat d'ajust del rellotge, en cas afirmatiu es cridarà a la subrutina encarregada d'incrementar el dígit que s'estigui ajustant en aquest moment. Si el microcontrolador no es troba en mode ajust, la funció per defecte del polsador FUNC es canviar la funció que l'usuari està utilitzant en aquest moment mitjançant l'increment d'una variable destinada a indicar la funció actual. Cada funció té un número concret i aquests estan definits al principi del codi mitjançant les directives #define.

El polsador MODE està destinat als modes d'ajust i reset. Si el microcontrolador es troba a la funció de rellotge o de data, la pulsació del polsador mode causarà que el microcontrolador entri en mode ajust, fer parpellejar el primer dígit a ajustar. En cada pulsació posterior de MODE, aquest canviarà al següent dígit a ajustar, excepte quan el dígit que s'estava ajustant fos l'últim, moment en el que sortirà del mode ajust. Si la funció del microcontrolador es la de velocitat màxima o la de distància, una pulsació de MODE causarà la crida de les subrutines que posen a 0 el valor corresponent segons la funció i escriuen el valor 0 per a aquestes a la EEPROM interna del microcontrolador.

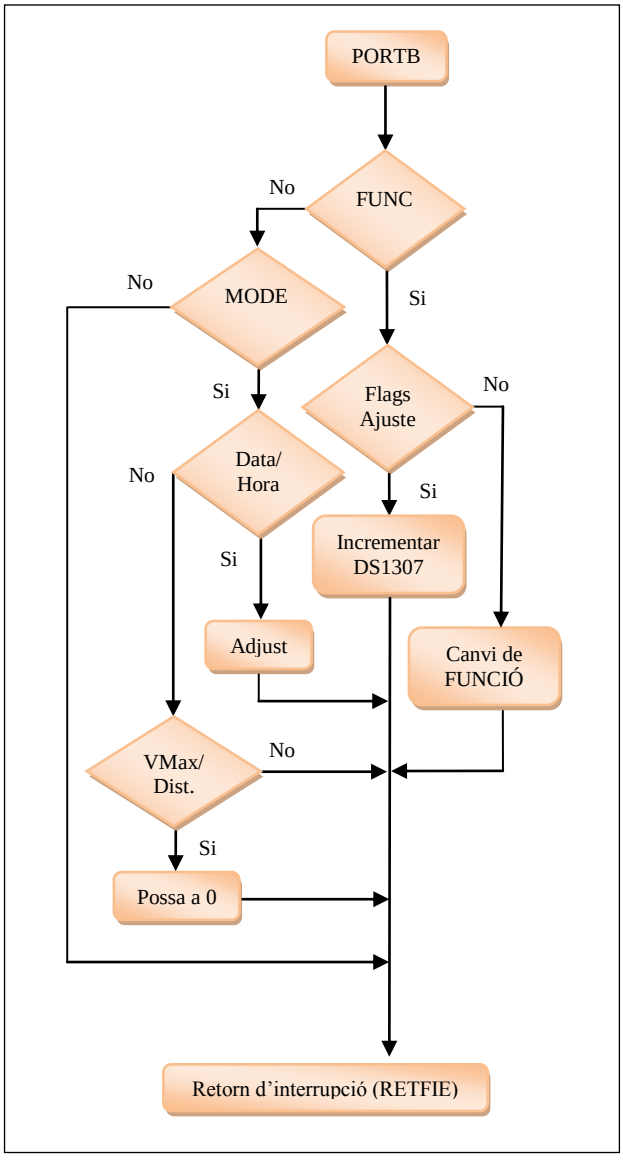


Figura 46. Diagrama d'atenció a interrupcions per canvi d'estat al PORTB

5.12. Interrupció del DS1307

Les interrupcions del DS1307 es un pols de alt a baix que genera el DS1307 amb una freqüència configurada prèviament. En aquest projecte la interrupció està configurada amb una freqüència de 1Hz, aconseguint una interrupció cada segon. La subrutina d'atenció a la interrupció llegeix el valor del rellotge en cada interrupció, i comprova si el microcontrolador es troba en mode rellotge o mode data. En cas afirmatiu crida les subrutines encarregades d'enviar al display l'hora o la data. Aquesta interrupció està configurada per succeir si es produeix un flanc descendent, però en mode ajust es reconfigura en cada atenció a interrupció per tal de que interrompi a cada flanc i generi interrupció cada 500ms. Un cop realitzades les funcions del rellotge, aquesta subrutina actualitza la velocitat lineal màxima i incrementa la distància recorreguda.

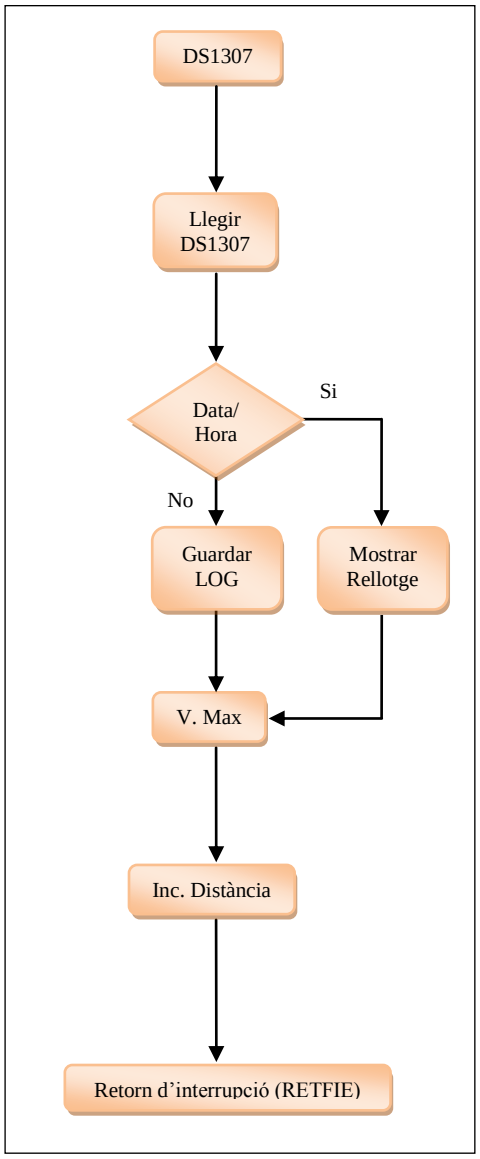


Figura 47. Diagrama d'atenció a interrupcions del DS1307

5.13. Interrupció per overflow del Timer1

La funció principal del Timer1 es comptar la duració dels polsos dels enconders, però a més a més, es controla si existeix senyal en algun dels encoders monitorats. En l'atenció a les interrupcions de CCP1 alternem l'entrada del multiplexor per a poder interpretar dos encoders diferents amb el mateix mòdul CCP, però si algun d'aquests encoders no produeix un senyal, no es produeixen interrupcions de CCP1 i per tant no s'alterna la entrada del MUX. En aquest cas, i segons l'encoder, el Timer1 té un timeout, alternant les entrades del MUX i posant el valor que s'està mesurant a 0 si es supera un temps màxim. En cas de no superar el temps màxim, simplement s'incrementa el comptador d'overflows.

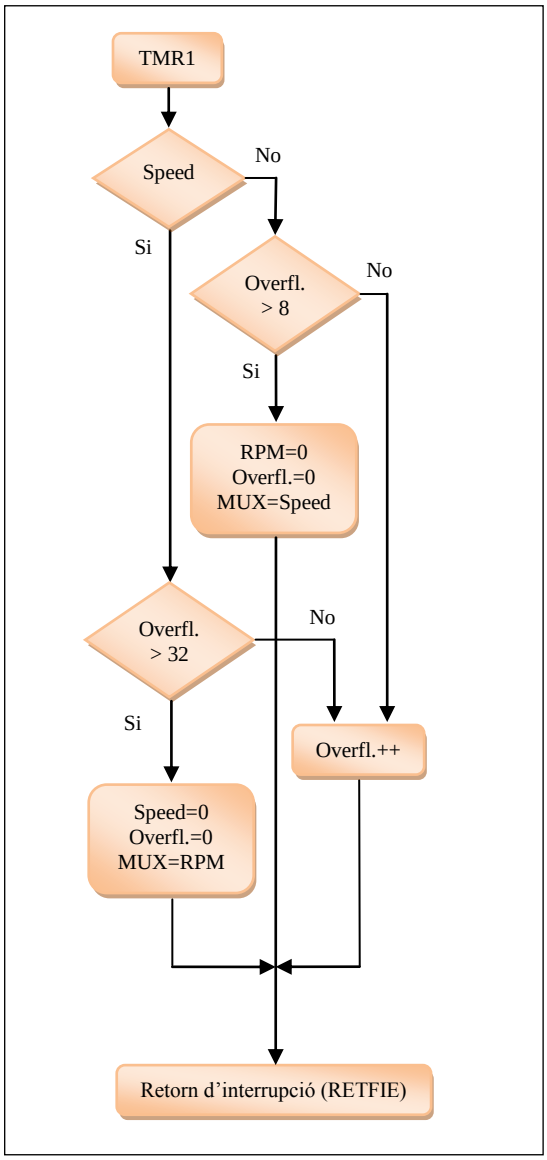


Figura 48. Diagrama d'atenció a interrupcions per overflow del TMR1

5.13.1. Timeout de l'encoder de velocitat lineal

Pel cas de la velocitat, el temps màxim es igual a desbordar el Timer1 32 vegades. En càlculs anteriors hem calculat que el temps necessari per a que la velocitat en Km/h coincideixi amb els polsos el temps d'espera era exactament 210ms, per tant, la freqüència mínima que es rebrà d'aquest enconder es un pols cada 210ms, temps que suposa els següents cicles d'instrucció:

$$Cicles\ instruc. = \frac{Delay(s)}{4/_{FOSC}} = \frac{0{,}21s}{4/_{20MHz}} = 1050000\ cicles$$

Degut a que el Timer1 està configurat per a utilitzar el divisor preescalar amb valor 8, i el capturador produeix una interrupció cada 16 polsos, el valor màxim del Timer1 serà:

$$Valor\ màxim\ TMR1 = \frac{Cicles\ instruc. \cdot 16}{8} = 1050000 \cdot 2 = 2100000$$

El Timer1 es de 16 bits, per tant el valor màxim es de 65536 cicles, i en conseqüència, el número d'overflows serà:

$$Overflows\ màxims = \frac{2100000}{65536} = 32{,}04 \approx 32$$

5.13.2. Timeout de l'encoder de velocitat del volant motor

En el cas de l'encoder de la velocitat del volant motor, el nombre màxim d'overflows que s'haurà de mesurar es de 8. Com hem calculat en apartats anteriors, el nombre mínim de polsos que es rebrà d'aquest enconder es de 5,664 polsos cada 100ms, per tant, tindrem 1 pols cada 17,66ms, que en cicles d'instrucció suposa:

$$Cicles\ instruc. = \frac{Delay(s)}{4/_{FOSC}} = \frac{0{,}01766s}{4/_{20MHz}} = 88300\ cicles$$

Degut a que el Timer1 està configurat per a utilitzar el divisor preescalar amb valor 8, i el capturador produeix una interrupció cada 16 polsos, el valor màxim del Timer1 serà:

$$Valor\ màxim\ TMR1 = \frac{Cicles\ instruc. \cdot 16}{8} = 88300 \cdot 2 = 176600$$

El Timer1 es de 16 bits, per tant el valor màxim es de 65536 cicles, i en conseqüència, el número d'overflows serà:

$$Overflows\ màxims = \frac{176600}{65536} = 2{,}69 \approx 3$$

5.14. Interrupció per overflow del Timer2

La funció del Timer2 es per a realitzar dos timeouts per a diferents tasques. La primera tasca es la de reactivar el Timer1 i habilitar les interrupcions per CCP1. Com ja s'ha comentat, cada vegada que s'atén a una interrupció del CCP1 s'atura el Timer1 i es deshabiliten les interrupcions del CCP1 per tal de que no es produeixi una quantitat d'interrupcions tan alta que el temps atenció a la interrupció sigui igual al temps que triga en tornar a interrompre el CCP1, causant d'aquesta manera que no es pugui executar el fil d'execució principal del programa amb normalitat. La temporització per tornar a habilitar les interrupcions per CCP1 es de aproximadament 200ms. El càlcul dels overflows necessaris del Timer2 es el següent:

$$Cicles\ TMR2 = \frac{\frac{Delay(s)}{4/FOSC}}{Preescaler \cdot Postescaler} - 1 = \frac{\frac{0{,}2s}{4/20MHz}}{16 \cdot 16} = 3906{,}25\ cicles$$

Com que el registre PR2 s'ha carregat prèviament amb el valor 255, el nombre d'overflows necessaris per a mesurar 200ms es:

$$Overflows\ TMR2 = \frac{Cicles\ TMR2}{PR2} = \frac{3906{,}25}{255} = 15{,}32 \approx 15$$

Sempre que el nombre d'overflows es múltiple de 15 habilitem les interrupcions del mòdul CCP1 i habilitem el Timer1, d'aquesta forma ho realitzem aproximadament cada 200ms.
La segona temporització del Timer2 es la de cridar les subrutines per a guardar la Velocitat màxima i la distància a la EEPROM interna del microcontrolador. Aquesta tasca s'ha de realitzar cada 5 segons, per tant:

$$Cicles\ TMR2 = \frac{\frac{Delay(s)}{4/FOSC}}{Preescaler \cdot Postescaler} - 1 = \frac{\frac{5s}{4/20MHz}}{16 \cdot 16} = 97656{,}25\ cicles$$

Com ja s'ha comentat en l'altra temporització, el registre PR2 té el valor 255, i per tant els overflows necessaris per a la temporització de 5 segons són:

$$Overflows\ TMR2 = \frac{Cicles\ TMR2}{PR2} = \frac{97656{,}25}{255} = 382{,}96 \approx 383$$

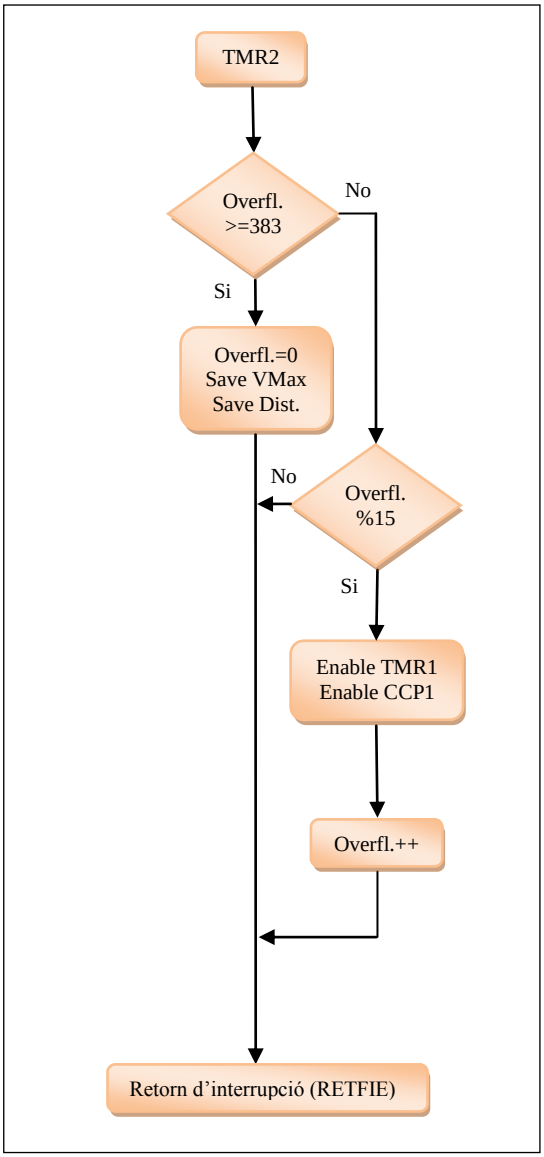


Figura 49. Diagrama d'atenció a interrupcions per overflow del TMR2

6. Conclusions

Acabat el projecte la conclusió principal que es pot extreure es satisfactòria, ja que malgrat que no s'han aconseguit tots els objectius del brainstorming inicial degut als problemes trobats i la falta de temps, si que s'han aconseguit els principals, i s'ha desenvolupat un sistema fàcilment ampliable de manera que més endavant es podran afegir la resta d'objectius i les noves idees que puguin sorgir com a actualitzacions. A més, el sistema creat utilitza els perifèrics propietaris del vehicle com ara display i maneta de control aconseguint així un dels objectius primordials, que el sistema quedés totalment integrat al vehicle.

Durant la realització del projecte han sorgit molts problemes, des de la falta de certs materials que s'han anat adquirint, problemes personals i de feina, i o canvi de requeriments. Un exemple de canvi de requeriments va ser la necessitat de memòria, ports, etc., que va obligar a canviar de microcontrolador, i sumat amb la addició de les llibreries de FAT, que van obligar a canviar tota la base del codi de llenguatge ASM a C18, van causar un endarreriment substancial.

Analitzant tots els problemes trobats s'arriba a la conclusió de que en tot projecte relacionat amb la informàtica i l'electrònica sempre apareixen nombrosos problemes que causen un endarreriment de la planificació original i que moltes vegades són difícils de predir, per tant es especialment important tenir cura d'aquestes coses al realitzar la planificació.

6.1 Propostes de millora

Un cop realitzat la primera versió del projecte, els objectius no realitzats passen a formar part de les propostes de millora, les quals es detallen a continuació:

- Realitzar un cronòmetre. Cronòmetre parcial que es posa a 0 juntament amb la distància recorreguda parcial. La forma més precisa de realitzar aquesta funció consistiria en canviar la font de rellotge del TMR1 i emprar un cristall de quars extern de freqüència 32.768Hz tal com utilitza el DS1307, i recalcular els valors de les condicions de número d'verflows. També es podria fer servir la interrupció que causa el DS1307, però aquest interromp 2 vegades per segon en mode ajust causant el parpelleig, per tant s'hauria de canviar la configuració per tal de que es mantingui sempre la interrupció cada segon i ralentir en conseqüència el parpelleig.

- Realitzar la funció de velocitat mitja. Un cop realitzat el cronòmetre i en conjunció amb la distància parcial recorreguda, es realitza el càlcul de la velocitat mitja. Aquesta funció consisteix simplement en dividir la distància parcial entre el temps parcial (cronòmetre) i mostrar el resultat al display.
- Funcions de consum. Per tal de realitzar aquesta funció serà necessari la realització de múltiples proves, ja que un cop identificada la senyal dels injectors del vehicle, i segons les especificacions d'aquests en conjunció amb el temps d'injecció es podrà estimar el consum instantani. La senyal es podrà mesurar igual que es fa amb la velocitat instantània o les revolucions del motor.
- Funció de nivell de combustible. S'haurà d'interpretar el senyal de la boia del dipòsit de combustible i utilitzant com a referència l'agulla del nivell de combustible del quadre d'instruments, obtenir la representació del senyal i mostrar al display.
- Funcions complementaries de combustible. Són les funcions que es deriven del consum instantani i/o del nivell de combustible i la distància parcial, com per exemple consum mitjà i autonomia. Per tal d'estimar el consum mitjà de combustible es podria utilitzar una variable de 32 bits en la qual es suma el valor del consum instantani cada segon. Al calcular el consum mitjà s'agafa el valor acumulat i dividit per 3600 (representaria els litres de combustible consumits) es divideix per la (distància parcial/100.000), obtenint d'aquesta manera el consum mitjà en litres/100Km. Per tal d'estimar l'autonomia simplement s'utilitza el valor dels litres restants al dipòsit de combustible i es divideix entre el consum mitjà, obtenint així l'estimació de la distància que es podrà recórrer amb el combustible restant.
- Millora de la distància parcial recorreguda. S'hauria de duplicar aquest senyal i fer servir un comptador dels que incorpora el microcontrolador, amb la senyal de l'encoder com a font. D'aquesta forma es pot saber sempre en número total de voltes de la roda, que en conjunció amb la longitud de la circumferència d'aquesta ens dóna la distància parcial exacta.
- Realització d'un software específic per a l'explotació de les dades del datalogger.
- Creació d'un PCB insolat, per possibilitar la distribució i/o comercialització del sistema.

7. Bibliografia

[1] Enrique Palacios, Fernando Remiro y Lucas J. López. *Microcontrolador PIC16F84, Desarrollo de proyectos*. Editorial Ra-Ma.

[2] Microchip: 16F628A Datasheet.

[3] Microchip: 18F4620 Datasheet.

[4] Microchip: PIC18 Configuration settings addendum

[5] Microchip: MPLAB® C18 C Compiler getting started

[6] Microchip: MPLAB® C18 C Compiler user guide

[7] Microchip: MPLAB® C18 C Compiler libraries

[8] Casanova, Alejandro. Tutorial MPLAB C18. http://www.infopic.comlu.com

[9] Universidad politécnica de Valéncia. Departamento de ingeniería electrónica. Tema 3: Microcontrolador PIC18F4550.

[10] Dallas Semiconductor: DS1307 Datasheet.

[11] National Semiconductor: LM1117 Datasheet.

[12] Fairchild Semiconductor: LM78XX Datasheet.

[13] Motorola: SN74LS153 Datasheet.

[14] Philips Semiconductors: The I2C-bus and how to use it.

[15] Serial Interconnect Buses I2C (SMB) and SPI. Embedded Systems and Software. University of Iowa. 2010.

[16] Federico Miyara. Diseño óptimo de un regulador de tensión en paralelo. Escuela de ingeniería electrónica. Facultad de ciencias exactas, Universidad Nacional de Rosario.

[17] Darío Carluccio. Protokoll TID. 2002. http://www.carluccio.de

[18] Opel TIS2000 Wiring diagrams. Software propietari intern d'Opel.

[19] http://www.wikipedia.org/

[20] http://www.ucontrol.com.ar

[21] http://www.todopic.com.ar/foros/

[22] http://www.infopic.comlu.com

8. Annexes: Contingut del CD-ROM

Amb aquest projecte s'adjunta un CD-ROM que conté la documentació complementaria del mateix inclosa a la bibliografia juntament amb el codi font del sistema. Tots els fitxers estan classificats per carpetes tal com es detalla a continuació:

- C18

  Conté tota la del compilador de C18. Manuals, guies i llibreries disponibles.

- Datasheets

  Conté els datasheets dels components principals, com per exemple el microcontrolador, el RTC, el multiplexor, etc.

- Misc

  Conté el document sobre la construcció de reguladors de tensió paral·lels.

- PIC18

  Conté un manual del PIC18F4550, del qual el 90% es aplicable al microcontrolador emprat en aquest projecte i l'especificació dels bits de configuració per a tota la família de microcontroladors PIC18

- Planning

  Conté el fitxer del programa Gantt Project amb el diagrama de Gantt de la planificació del projecte i una exportació a imatge JPEG del mateix.

- Schematic

  Conté l'esquema del circuit en format ISIS versió 7.5 de la suite Proteus.

- Serial buses

  Conté les especificacions del bus I^2C, transparències sobre busos sèrie (I^2C i SPI) i les especificacions del protocol per al control del display TID.

- Source

  Projecte de Microchip MPLAB que comprèn tots els fitxers.

Resum

Aquest projecte es centra en l'àmbit de l'automoció, i la seva finalitat es ampliar la informació que proporciona el fabricant d'un vehicle als usuaris. Un dels punts claus del sistema es que aquest quedi completament integrat, per tant, utilitza dispositius externs inclosos al vehicle com ara el display que anteriorment s'utilitzava per a mostrar informació del radiocasset, i una maneta amb dos botons utilitzada per al control del sistema. El sistema mostra diferents tipus d'informació com ara la velocitat instantània del vehicle, revolucions del motor, velocitat màxima, distància parcial recorreguda, hora i data. A més de mostrar informació, aquest es capaç d'emmagatzemar-la en un suport extraïble per a una posterior explotació a un ordinador convencional.
El sistema no està pensat per a ser un projecte tancat, sinó que esta dissenyat per a ser ampliat en quant a funcionalitats, i per tant aquest projecte es considera una primera versió.

Resumen

Este proyecto se centra en el ámbito de la automoción, y su finalidad es ampliar la información que proporciona el fabricante de un vehículo a los usuarios. Uno de los puntos clave del sistema es que quede completamente integrado, por tanto, utiliza dispositivos externos incluidos en el vehículo como son un display que anteriormente se utilizaba para mostrar información del radiocasete, i una maneta con dos botones para el control del sistema. El sistema muestra diferentes tipos de información como la velocidad instantánea del vehículo, revoluciones del motor, velocidad máxima, distancia parcial recorrida, hora y fecha. Además de mostrar información, éste es capaz de almacenarla en un soporte extraíble para su posterior explotación en un ordenador convencional.
El sistema no está pensado para ser un proyecto cerrado, sino que está diseñado para ser ampliado en cuanto a funcionalidades, y por tanto este proyecto se considera una primera versión.

Abstract

This project focuses on the automotive sector, and aims to extend the information provided by the manufacturer of a vehicle to users. One of the key points of the system is that is fully integrated, therefore, it uses external devices provided in the vehicle such as a display that was previously used to display information from the radio cassette, and a handle with two buttons to control the system. The system displays different types of information as the instantaneous speed, engine speed, maximum speed, partial distance traveled, time and date. In addition to displaying information, it is able to store it in a removable media for later use in a conventional computer.
The system is not intended to be a final draft, but is designed to be expanded in terms of functionality, and therefore this project is considered a first version.