Escola Universitària Politècnica de Mataró

Centre adscrit a:

UNIVERSITAT POLITÈCNICA DE CATALUNYA

Grau en Enginyeria Electrònica Industrial i Automàtica

Mòdul de vigilància i alarma per a vehicles

Memòria

Jordi Bolet Castellà
PONENT: Julián Horrillo

TARDOR 2013

TecnoCampus Mataró-Maresme

Resum

En aquest projecte s'ha realitzat un mòdul de vigilància i alarma per a vehicles incorporant les tecnologies GPS i GSM. Per a fer-ho s'ha adquirit un prototip existent i s'ha programat perquè tingui la possibilitat d'enviar les coordenades GPS, tant en una pàgina web com en un mòbil. A partir del hardware del prototip, s'ha dissenyat de manera teòrica un nou dispositiu amb una simplificació de components.

Resumen

En este proyecto se ha realizado un módulo de vigilancia y alarma para vehículos incorporando las tecnologías GPS y GSM. Para hacerlo se ha adquirido un prototipo existente y se ha programado para que tenga la posibilidad de enviar las coordenadas GPS en una página web o en un móvil. Basándose en el hardware del prototipo, se ha diseñado de forma teórica un nuevo dispositivo con una simplificación de componentes.

Abstract

In this project has been developed a vehicle alarm system incorporating GPS and GSM technologies. To do this, it has acquired an existing prototype that has been programed to have the ability to send GPS information on a website or mobile. Based on prototype hardware, has theoretically designed a new device with a simplification of components.

Índex

Índex de figures

1. Objecte del projecte

En aquest projecte es pretén aconseguir un sistema que serveixi pel control de la posició d'un vehicle amb la utilització d'una placa base, un mòdul GPS i un mòdul GPRS/GSM. Es desenvoluparà un sistema real a partir d'un model existent i es realitzarà un nou disseny del sistema per tal de minimitzar-ne el cost, els components electrònics, el tamany físic i poder encabir-lo en una carcassa. Es farà la programació del sistema per tal de poder establir comunicacions entre el sistema i un telèfon mòbil o un ordinador. I finalment es disposarà d'una memòria amb l'explicació del funcionament del conjunt i l'estudi de les seves possibilitats d'aplicació, contemplant un disseny final per a la seva producció.

A partir de l'avantprojecte realitzat es va acotar el projecte i es va especificar què és el que es duria a terme exactament. En aquest avantprojecte es va decidir adquirir, al proveïdor Cooking Hacks, la placa de desenvolupament Arduino Uno Rev3 amb el mòdul SIM908 que es tracte d'un mòdul GPS+GPRS integrat i les antenes GPS i GSM per utilitzar com a prototip del sistema. El següent pas va ser revisar el pressupost de l'avantprojecte, per si hi havia alguna modificació, i seguidament fer la comanda de materials necessaris que quedaven pendents. Els quals eren una targeta de la companyia telefònica Movistar i un cable USB amb una entrada Tipus A-B (és un cable conegut vulgarment com el cable de la impressora). Un cop es va tenir tot en mà, es va fer una revisió de material així com la comprovació del seu correcte funcionament, i un cop comprovat es van realitzar les connexions físiques del mòdul SIM908 amb les antenes i el Arduino. Amb això enllestit, el següent pas va ser fer la programació a través de la IDE del Arduino. El primer pas dins de la programació va ser procurar les comunicacions entre el dispositiu mòbil i el prototip. S'havia de garantir que el prototip rebés missatges SMS d'un dispositiu mòbil, que llegís les coordenades GPS i que enviés missatges SMS amb les coordenades GPS. Un cop amb això aconseguit es va programar que el prototip es pogués comunicar també a través d'un ordinador, o sigui que les coordenades de posició es poguessin obtenir a través del mòbil o a través del PC. Un cop aconseguit el funcionament i les comunicacions a través de mòbil i ordinador, es va començar a dissenyar de manera teòrica un nou model del prototip. Abans de començar amb el disseny del nou model es va fer un profund anàlisi del funcionament del Arduino i del mòdul SIM908, així com del seu muntatge físic i dels esquemes electrònics, ja que el nou model es basa en el prototip amb el que es van realitzar les

proves reals. Aquest prototip existent s'ha de tenir en compte que comporta la connexió d'una placa de desenvolupament amb un mòdul GPS+GPRS i unes antenes, tot de manera independent. Per tant un dels objectius del nou disseny teòric era incloure la placa amb el mòdul i les antenes com un de sol i així poder encabir-lo en una carcassa, d'aquesta manera obtenir un sistema més robust ja que seria d'una sola peça i se'n reduiria el tamany físic i els components electrònics. També es va intentar reduir-ne el preu per tal d'aconseguir un preu competitiu de mercat. És per això que quan es va tenir el nou model dissenyat se'n va fer un petit càlcul del seu cost en el cas que es volgués fabricar. Finalment, es va fer referència a les possibles lleis que poden afectar degut al impacte ambiental i es van treure conclusions finals del treball realitzat, així com possibles millores i futures línies de treball del projecte. Tots els passos esmentats es troben detallats en aquesta memòria del projecte.

2. Antecedents i necessitats d'informació

Els robatoris de cotxes han anat disminuint durant els últims 20 anys. Segons l'estudi (publicat el 20 de febrer del 2013) realitzat per l'empresa Línia directa asseguradora S.A. amb l'ajut del Ministeri de l'interior i el grup de trànsit il·lícit de la brigada central de crim organitzat de la policia nacional, l'any 1990, es van robar al voltant de 100.000 cotxes en tot l'estat espanyol, la mitjana de cotxes robats a l'any s'ha anat mantenint fins l'any 2007 quan hi va haver un total de 94.875 robatoris. Però a partir de llavors hi ha hagut un descens molt important ja que els últims anys, des del 2010 fins el 2012 s'han robat una mitja de 50.000 vehicles l'any a tot l'estat. Un dels factors més importants que ha contribuït en aquest descens dels robatoris és el fet que els cotxes tenen més sistemes de seguretat. Tot i aquest descens, Espanya és un dels països més afectats pel robatori de cotxes a la Unió Europea, situant-se en el cinquè lloc, per darrere de Itàlia, Regne Unit, França i Alemanya. Cal destacar que en el conjunt de països de la Unió Europea es van robar un total de 1,3 milions de cotxes.

Si es centra l'atenció en el cas de les motocicletes s'observa que les dades són similars. Segons l'estudi (publicat el setembre de 2012) realitzat per l'empresa Pont Grup Correduría de Seguros S.A., l'any 2011, es va robar, a l'estat espanyol, una motocicleta cada 11,8 minuts. Al voltant de 45.000 motocicletes l'any. La diferencia entre els robatoris de cotxes i els de motocicletes, és que mentre els de cotxes disminueixen cada any, la freqüència de robatoris de motocicletes han augmentat des de l'any 2010. Del gener al juliol de 2010 de cada 100 motocicletes se'n robava un 1,08% mentre que el mateix període de 2012 la freqüència va ser de 1,57%. Per tant es pot constatar que hi ha hagut un increment del 45% de la freqüència de robatoris de motocicletes a l'estat espanyol en un període de 2 anys.

Com es pot comprovar, a l'estat espanyol es roben gairebé 100.000 vehicles si es fa la suma entre els robatoris de cotxes i motocicletes. Un dels problemes més greus en el cas de les motocicletes és que el robatori d'aquestes ha augmentat. I si es fa referència als cotxes, un dels problemes que hi ha és que encara que els robatoris d'aquests han disminuït, Espanya ocupa el 5è lloc a la llista de països de la UE més afectats pel robatori de cotxes. És per això que el mòdul de vigilància i alarma per a vehicles que es desenvolupa en

aquest projecte pot ajudar a reduir aquests problemes. Per tant és una alternativa que permet com a finalitat oferir al mercat una solució.

Per a la realització d'aquest projecte és necessari adquirir coneixement previ dels components que s'utilitzen així com del funcionament i tecnologies del sistema conjunt que es dissenya. És per això que al capítol 4 del projecte (estat de l'art i viabilitat tècnica) es fa una breu explicació de les tecnologies emprades en aquest projecte i al capítol 6 una explicació extensa del Hardware i Software emprats.

El sistema que s'implementa incorpora la possibilitat d'informar en tot moment de la ubicació geogràfica del vehicle o element a controlar, és per això que en el sistema que s'ha dissenyat s'incorpora un dispositiu GPS a la placa base. D'aquesta manera s'obté un sistema de posicionament global capaç de rebre la ubicació geogràfica del vehicle o element a controlar. Però s'ha de tenir en compte que un dispositiu GPS no envia cap senyal, és un element receptor i és per això que es necessita una segona tecnologia per enviar aquesta informació al telèfon mòbil. És per això que per enviar la informació des del sistema s'incorpora un mòdem GPRS/GSM. Aquest es pot programar com un mòbil amb el seu propi número i d'aquesta manera des del telèfon on es vol rebre la ubicació geogràfica del vehicle es pot enviar un SMS. Quan el mòdem GPRS/GSM incorporat en el sistema rep el SMS, agafa la informació de les coordenades que rep el GPS i les envia amb un altre SMS al telèfon mòbil on s'ha demanat la informació. D'aquesta manera s'obté la posició geogràfica del vehicle des de qualsevol lloc del món amb cobertura GPS i GPRS/GSM.

Al mercat es poden trobar varies empreses que desenvolupen i utilitzen mòduls de vigilància per a vehicles basats en tecnologia GPS. La majoria però, no ofereixen al mercat pel públic general la venta dels seus sistemes; sinó que el que ofereixen és un servei a empreses, sobretot de transport, pel control de flotes d'aquestes, amb un cost per la instal·lació del sistema als vehicles, així com la seva posada en marxa i l'explicació del funcionament d'un software propi, i després cobren quotes mensuals de manteniment. Aquest és el negoci de moltes empreses que desenvolupament els hardwares i software dels seus propis mòduls de vigilància i alarma per a vehicles com és el cas de l'empresa Mobilefleet. Una altra empresa que desenvolupa i ven mòduls de vigilància i alarma per a vehicles és la xinesa Xexun, la qual ofereix diferents dispositius basats en tecnologia GSM

i GPS, però a diferència de la primera, aquesta ofereix els productes pel públic general i no a empreses.

Per a realitzar el disseny teòric propi del dispositiu i fer-ne la programació s'han utilitzat de manera constant les informacions de les pàgines web de Arduino (www.arduino.cc) i Cooking Hacks (www.cooking-hacks.com). La primera ha servit per entendre de manera detallada la placa de desenvolupament Arduino, tant en la basant de Hardware com en la de Software d'aquest. Al oferir un producte amb llicència Creative Commons ShareAlike, que permet copiar, distribuir, construir i transformar el material, entre d'altres coses, ha estat fàcil obtenir molta informació detallada del producte. La segona és la pàgina web d'on s'ha obtingut el mòdul GPS+GPRS/GSM. Aquesta pàgina també conté informació detallada del Hardware que incorpora i de la seva integració amb Arduino. També s'hi poden trobar molts programes d'exemple per a la funcionalitat del mòdul desenvolupats amb l'entorn de desenvolupament Arduino i han estat de gran ajuda per a poder desenvolupar el programa del projecte que permet la funcionalitat del dispositiu GPS+GPRS/GSM.

3. Abast

L'abast d'aquest projecte engloba el disseny i construcció exitosa d'un mòdul d'alarma i vigilància per a vehicles prenent com a base un mòdul ja existent programant-lo de manera que obtingui la posició GPS i aquesta es pugui enviar tant en una pàgina web com en un mòbil. A partir d'aquest model existent també es contempla obtenir un disseny teòric nou optimitzat i amb cost reduït respecte el model existent. A continuació es detalla en diferents punts la totalitat de l'abast del projecte amb el que es pretén aconseguir tot el que s'ha esmentat.

- Elecció de components. El primer punt, que es va desenvolupar durant l'avantprojecte, va ser estudiar les diferents possibilitats i components que ofereix el mercat actual i a partir d'aquí triar els components que millor s'ajustaven als criteris de cost, facilitat tècniques de muntatge, programació i característiques tècniques (tamany, durabilitat, efectivitat, robustesa, fiabilitat, benefici, rendibilitat).

- Muntatge del sistema. Un cop es va fer l'elecció dels components pel desenvolupament del mòdul de vigilància per a vehicles, el següent pas ha estat integrar i connectar els components obtinguts seguint les instruccions donades pels fabricants de manera que s'aconsegueixi una correcte connexió i que la part de Hardware funcioni. També s'ha tingut present que el sistema pot tenir la possibilitat d'incorporar una bateria per disposar d'autonomia, ja que ha de ser un aparell que tingui la possibilitat de posar-se en algun lloc discret del vehicle. De totes maneres també s'ha tingut en compte que el dispositiu existent es pugui alimentar utilitzant la sortida d'alimentació de 12V de que disposen normalment els vehicles. Ja que l'objectiu és que el dispositiu pugui funcionar dins el cotxe i tant la bateria com l'alimentació 12V donen aquesta possibilitat.

El sistema pot informar de la posició geogràfica i enviar-la tant a un mòbil com a una pàgina web. Però, per a fer possibles aquestes possibilitats el dispositiu s'ha hagut de programar. A través de la programació que se'n ha fet, el sistema obté la posició GPS i espera. Quan rep un SMS d'un mòbil, depenent del text del missatge, el dispositiu farà una cosa o una altra. Per tant el següent pas ha estat la programació del sistema.

- Programació del sistema. Permet la funcionalitat següent. Quan s'envia un SMS al dispositiu des del telèfon mòbil des d'on es vol rebre la informació, la programació permet rebre aquest SMS i seguidament agafa les dades de posició geogràfica en NMEA que proporciona el GPS i enviar aquestes a través d'un altre SMS al telèfon o ordinador. Si se li envia un missatge de text amb la paraula -pos-, el sistema envia les dades al mòbil i si se li envia la paraula -ser- s'envien a una pàgina web que es pot consultar des de l'ordinador. Si s'envia qualsevol altra paraula, el dispositiu no fa res i el SMS s'esborra.

- Proves reals del sistema. Un cop el sistema prototip ha estat muntat, durant la programació, i també un cop finalitzada, s'han anat realitzant proves experimentals amb el mòdul de vigilància i alarma per a vehicles muntat i programat per a verificar el seu correcte funcionament.

- Nou disseny teòric. Un cop fet i testat el muntatge i la programació del sistema prototip, s'ha dissenyat el propi sistema optimitzant i suprimint, de manera teòrica, la utilització d'alguns components que incorpora el prototip original i se'n podria prescindir de manera que es pogués reduir al màxim la mida física de l'aparell per tal d'encabir-lo en una carcassa, ja que l'objectiu seria poder tenir el sistema dins el vehicle sense que se'n noti la seva presencia, per tant hauria de ser un aparell no gaire voluminós. S'ha incorporat una bateria que ofereixi autonomia al sistema, i amb això els components i circuit necessari per implementar-la al nou disseny. Un dels objectius de disseny han estat permetre que el sistema sigui fàcil d'integrar en un vehicle, és per això que a l'hora de dissenyar-lo s'ha tingut molt en compte aquest fet. També s'ha remarcat que el nou disseny no sigui un producte totalment tancat, o sigui que permeti ser ampliable i es pugui actualitzar tant a nivell de Hardware com de Software.

- Cost i possibilitats de fabricació. Un cop el nou sistema ha estat dissenyat, de manera teòrica, s'han considerat els costos de tots els components utilitzats per si es tingués la possibilitat de fabricar el dispositiu.

4. Estat de l'art i viabilitat tècnica

En aquest capítol s'expliquen de manera general la placa de desenvolupament (Arduino) utilitzada com a base del projecte, que esdevé un element important per garantir la viabilitat tècnica ja que permet el control de l'aplicació que es desenvolupa, així com les tecnologies imprescindibles utilitzades que sense aquestes el mòdul de vigilància i alarma per a vehicles no seria possible.

4.1. Arduino

Arduino és una placa de circuit imprès simple basada en un microcontrolador de codi obert provinent de la plataforma de codi obert Wiring, basat en llenguatge de programació C, amb l'objectiu de fer més simple i accessible el disseny de circuits electrònics amb microcontroladors.

Arduino consisteix en dissenys simples de Hardware lliure amb processadors Atmel AVR en una placa amb pins E/S (entrades i sortides). L'entorn de desenvolupament implementa el llenguatge Processing de Wiring, el qual com s'ha comentat està basat en C. Arduino es pot utilitzar per desenvolupar objectes interactius autònoms o pot ser connectat a programari de l'ordinador (com per exemple Macromèdia Flash, Processing, Max/MSP, Pure Data). Les plaques es poden muntar a mà o adquirir-se i la IDE (entorn de desenvolupament) de font oberta es pot descarregar de franc.

Aquesta placa de desenvolupament és una plataforma oberta d'electrònica que es fa servir per a la creació de prototips basats en software i hardware flexibles i fàcils d'utilitzar. Es va crear per a artistes, dissenyadors, aficionats, en general, per a qualsevol persona interessada en crear entorns o objectes interactius.

El Arduino pot agafar informació de l'entorn a través dels seus pins d'entrada de tota una gamma de sensors i pot afectar a tot el que l'envolta controlant llums, motors i altres actuadors, en el cas del projecte per controlar el xip SIM908 responsable d'obtenir les dades GPS i enviar-les amb tecnologia GPRS/GSM. Com s'ha dit el microcontrolador de la placa Arduino es pot programar mitjançant el llenguatge de programació Arduino (basat en Wiring) i l'entorn de desenvolupament Arduino (basat en processing). Els programes

En aquest capítol s'utilitzen les referències [5]-[7]

fets amb Arduino es poden executar sense necessitat de connectar-se a un ordinador, tot i que tenen la possibilitat de fer-ho i comunicar-se amb altres tipus de software, tal i com ja s'ha comentat.

Els fitxers de disseny de Hardware de les plaques (CAD) estan disponibles sota una llicència oberta Creative Commons, així doncs s'és lliure d'adaptar-los a les necessitats que es demanen. Dins de Arduino existeixen un munt de plaques diferents. En el projecte que s'ha realitzat s'ha buscat informació referent a la placa Arduino Uno que és la que es fa servir.

El Arduino Uno és una placa electrònica basada en el microprocessador ATmega328. Té 14 pins digitals d'entrada/sortida (dels quals 6 es poden utilitzar com a sortides PWM), 6 entrades analògiques, un ressonador ceràmic de 16 MHz, una connexió USB, un connector d'alimentació, una capçalera ICSP i un botó de reinici. La placa conté tot el necessari per donar suport al microcontrolador, només cal connectar-la a un ordinador amb un cable USB o a una font d'alimentació amb un adaptador AC/DC o bateria per començar. Aquest fet ha estat de vital importància per a decantar-se a utilitzar aquesta placa de desenvolupament en el projecte, ja que al comptar amb l'entrada d'alimentació i la possibilitat d'inserir una bateria, la fan fàcilment integrable a un vehicle, podent funcionar de forma autònoma dins d'aquest o endollat a la sortida d'alimentació que solen incorporar els vehicles.

Hi ha varies versions de la placa Arduino Uno, per tant s'ha d'esmentar que la que s'ha fet servir al projecte és la revisió 3, també anomenada Rev3 o R3. Aquesta compta amb un altre microcontrolador, el ATMEGA16U2 programat com a convertidor USB a sèrie.

El Arduino Uno R3 pot ser alimentat a través de la connexió USB o amb una font d'alimentació externa. La font d'alimentació es selecciona automàticament. Si es vol prescindir de la connexió USB, la font d'alimentació externa pot venir amb un adaptador AC/DC o una bateria com ja s'ha comentat. L'adaptador es pot utilitzar al endollar una entrada de 2,1 mm de centre positiu al connector d'alimentació de la placa. Els cables de la bateria es poden inserir en els capçals dels pins GND i Vin del connector d'alimentació.

La placa pot funcionar amb un subministrament extern de 6 a 20 volts. Si es proporcionen menys de 7V, però, el pin de 5V en pot subministrar menys i la placa pot no tenir suficient

voltatge per a funcionar o pot fer-ho de manera inestable. Si s'utilitzen més de 12V, el regulador de voltatge que incorpora la placa es pot sobreescalfar i danyar la placa. És per això que el rang recomanat és de 7 a 12 volts.

El preu de la placa Arduino Uno bàsica és de 20 Euros amb IVA no inclòs. Per a la programació d'aquesta placa com ja s'ha comentat es disposa de software lliure, per tant també d'un munt de biblioteques de codi que es poden descarregar lliurament ja que els propis usuaris/clients de Arduino poden oferir codi lliurament i no s'ha de pagar cap tipus de llicencia per obtenir-lo, per tant, això ofereix un avantatge molt important a l'hora de programar ja que és una placa que ofereix molta facilitat per a usuaris que comencen a familiaritzar-se en la programació de plaques de desenvolupament amb microcontroladors.

4.2. Tecnologia GPS

El NAVSTAR GPS (*Navigation Signal Timing and Ranging Global Positioning System*), conegut simplement com a GPS (*Global Positioning System*) és un sistema que permet calcular, mitjançant un receptor adequat, les coordenades (longitud i latitud) de qualsevol punt de la superfície terrestre a partir de la recepció de senyals emesos per una constel·lació de satèl·lits en òrbita. La precisió habitual és de fins a uns pocs metres, encara que amb tècniques com el GPS diferencial pot arribar fins al centímetre. El sistema va ser desenvolupat i encara és mantingut pel Departament de Defensa dels EUA, però es permet la seva lliure utilització a la resta de la societat civil del món. És l'únic sistema de posicionament global totalment operatiu i funcional a dia d'avui i està format per tres conjunts de components anomenats segments.

4.2.1. Segment espacial

Està constituït d'una banda pels satèl·lits que suporten el sistema i d'una altra pels senyals de comunicació que aquests emeten. Els primers conformen la constel·lació NAVSTAR que es compon amb 21 satèl·lits operatius més 3 de reserva. La seva distribució és tal que en qualsevol moment es pot llegir el senyal d'almenys quatre satèl·lits amb una elevació major de 15º sobre l'horitzó. Aquest número de satèl·lits és el mínim necessari per a determinar la posició de l'observador, però la major part del temps n'hi ha un número major de disponibles. Els segons, o sigui, els senyals de comunicació, estableixen connexió entre els satèl·lits i la superfície de la Terra mitjançant senyals situats a la regió de

freqüències de ràdio de l'espectre electromagnètic. Sobre un senyal principal s'emeten diversos senyals portadors (*carriers*) amb freqüències diferents dins una banda anomenada L de l'espectre radioelèctric, que és la que presenta una major transparència atmosfèrica per a realitzar aquesta comunicació.

Els principals senyals portadors s'anomenen L1 i L2, amb freqüències diferents. Sobre ells es modula, codificada en binari, la informació necessària per al funcionament del sistema i per al càlcul de posicions.

4.2.2. Segment de control

Està format per totes les infraestructures situades a la Terra necessàries per a mantenir un control homogeni de la constel·lació de satèl·lits. Se'n distingeixen tres tipus:

- Estació Principal de Control: ubicada a la base aèria Schriever, a Colorado Springs (EUA), és la instal·lació de processament central.
- Estacions de control: són cinc, ubicades a Colorado Springs (EUA), Ascensión (Atlàntic Sud), Hawai (Pacífic Oriental), Kwajalein (Pacífic Occidental) i Diego García (Índic). Només realitzen captació de dades, que envien a l'Estació Principal de Control per al seu processament en temps real.
- Antenes de terra: N'hi ha tres, a Ascensión, Diego García i Kwajalein, ubicades a les estacions de control. Són totalment automàtiques i estan sota el control directe de l'estació principal de control. La seva funció és permetre el control i l'enviament de correccions als satèl·lits de la xarxa NAVSTAR.

L'estació principal de control efectua una predicció precisa de les òrbites dels satèl·lits, conegudes com a almanacs. No obstant, les alteracions en la força de la gravetat provoquen certes degradacions en les òrbites que podrien reduir la precisió del sistema GPS. Les estacions de control realitzen un seguiment continuat dels satèl·lits per tal de detectar desviaments de les seves òrbites.

Les dades són transmeses a l'estació principal de control, que calcula les correccions necessàries als almanacs. Les òrbites corregides i amb informació temporal són anomenades efemèrides, que són enviades a les antenes de terra per a la seva retransmissió

als satèl·lits, per tal que les puguin incorporar als missatges de navegació que reben els usuaris del sistema.

4.2.3. Segment d'usuari

Està constituït pel maquinari (els equips de recepció, coneguts com a unitats GPS) i pel seu programari, utilitzats per a captar i processar els senyals dels satèl·lits, mostrar la localització, indicar els punts de destinació i determinar les rutes. El tipus de receptor va lligat al mètode de mesura i als requeriments de precisió.

4.2.4. Descripció del funcionament GPS

Inicialment un receptor recull els almanacs i les efemèrides de tots els satèl·lits (és a dir, les posicions corregides dins la seva òrbita), de forma que quan es desitja determinar una ubicació, l'aparell ja coneix la posició exacta dels satèl·lits als quals pregunta.

El receptor GPS localitza com a mínim quatre satèl·lits de la xarxa, i per un mètode matemàtic anomenat trilateració el receptor trobarà la seva posició exacta. A partir del temps de viatge del senyal entre el satèl·lit i el receptor (obtinguda a partir de l'hora de satèl·lit), i considerant que aquell es mou a la velocitat de la llum es calcula la distància respecte el satèl·lit.

Amb cada satèl·lit enquestat s'obté que el receptor es troba sobre una esfera amb centre el propi satèl·lit i radi la distància calculada. Amb la informació de dos satèl·lits només es pot determinar que el receptor es troba a la circumferència que resulta de la intersecció de les dues esferes. Amb tres satèl·lits s'assegura que el receptor només pot ser a dos punts possibles, un dels quals és incorrecte. La Fig. 4.1. mostra el procés de trilateració suposant bidimensionalitat.

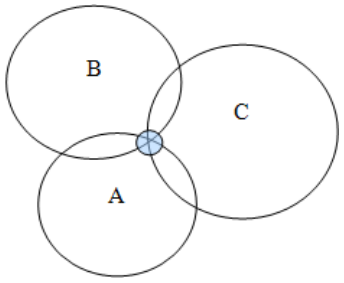

Fig. 4.1. Sistema de trilateració on la petita esfera blava del mig és el receptor

En aquest capítol s'utilitzen les referències [5]-[7]

En teoria amb això ja es disposaria de la posició exacta del receptor, però a la pràctica l'hora dels satèl·lits i del receptor no estan sincronitzades, de forma que en lloc d'obtenir un sol punt es disposaria d'un petit volum. Per aquesta raó s'utilitza la mesura del quart satèl·lit, que reduiria el volum a un sol punt. Com que l'aparell coneix les òrbites teòriques dels satèl·lits i les correccions pot calcular les seves efemèrides. Gràcies a elles i les distàncies s'obté la posició tridimensional exacta (és a dir, longitud, latitud i alçada) del receptor. Com a efecte col·lateral de posicionament GPS, s'obté una mesura del temps molt precisa, propera a la dels rellotges atòmics que incorporen els satèl·lits i als que els sincronitzen des de terra.

4.2.5. Protocol NMEA

Molts receptors GPS permeten l'intercanvi d'informació amb altres dispositius o fins i tot un ordinador mitjançant el protocol NMEA. És un protocol de comunicacions sèrie amb format ASCII que defineix com es transmeten sentències de dades de fins a 80 caràcters des d'un emissor a un o varis receptors.

Cada missatge comença amb un caràcter de dòlar. Els següents dos caràcters són el prefix d'identificació de l'emissor, que són GP per a aparells GPS (tot i que hi ha un gran número de prefixos). Els tres caràcters següents identifiquen el tipus de missatge. A continuació segueix una sèrie de camps que depenen del missatge, delimitats per comes. Per a tancar el missatge hi ha un caràcter asterisc i dos dígits de suma de verificació, encara que en alguns tipus de missatges poden no aparèixer. El missatge acaba amb dos caràcters especials: retorn de carro i canvi de línia (<CR><LF>). A la Taula 4.1. es resumeix el protocol NMEA, on els signes d'interrogació indiquen un caràcter alfanumèric.

Inici	**ID**	**Tipus**	**Camps**	**Suma**	**Final**
$	??	???	,?...?,	*??	<CR><LF>

Taula 4.1. Dades del protocol NMEA

D'entre els possibles missatges NMEA destaca el que comunica una posició en latitud i longitud i el temps de la mesura, identificat com GLL. Té el format que es mostra a continuació.

$GPGLL,llll.ll,a,yyyyy.yy,b,hhmmss.ss,c,d*zz<CR><LF>

llll.ll és la latitud, segons el format indicat més avall.

a pot valer N (Nord) o S (Sud).

yyyyy.yy és la longitud, segons el format indicat més avall.

b pot valer E (Est) o W (Oest).

hhmmss.ss és l'hora UTC de la mesura, segons el format indicat més avall.

c és el valor de l'indicador d'estat. Pot valer A (dades vàlides) o V (invàlides).

d és el valor de l'indicador de mode. Normalment serà A (dades automàtiques).

zz és el valor de la suma de control.

La longitud i la latitud s'expressen amb un format com *zzzzz.zz*, on els dos dígits a l'esquerra del punt decimal són els minuts, la resta de dígits a l'esquerra són els graus de la mesura i els decimals a la dreta del punt decimal són les dècimes de minut. En resum, *aaabb.bb*, on *aaa* són els graus de la mesura i *bb.bb* els minuts. El temps s'expressa com *hhmmss.ss*, on *hh* són les hores, *mm* els minuts i *ss.ss* els segons (amb precisió de dos dígits) en temps UTC.

Aquest missatge NMEA que s'ha descrit és només un exemple estàndard, però hi pot haver altres missatges diferents que comuniquen el número de satèl·lits detectats pel receptor a l'hora de captar la posició i/o l'altitud, com és el cas del receptor GPS que s'utilitza en el projecte.

4.2. Tecnologia GSM

El sistema global per a les comunicacions mòbils, GSM, és un estàndard mundial de segona generació (2G) per a telefonia mòbil. Un client GSM pot connectar-se a través del seu telèfon amb el seu ordinador i enviar i rebre missatges per correu electrònic, faxos, navegar per Internet, accedir amb seguretat a la xarxa informàtica d'una companyia (xarxa local/Intranet), així com utilitzar altres funcions digitals de transmissió de dades, incloent el servei de missatges curts (SMS) o missatges de text. És aquest el sistema, junt amb el

En aquest capítol s'utilitzen les referències [5]-[7]

GPRS, que s'inclou en l'aparell dissenyat en el projecte que servirà per a enviar les dades rebudes del GPS al telèfon mòbil o ordinador que les requereixi.

Aquest sistema va ser desenvolupat pel grup "Group Special Mobile" (les sigles GSM provenen originàriament d'aquí) de l'institut d'enginyeria de telecomunicacions europeu (ETSI), la principal diferència amb els seus predecessors radica en que tant els canals de veu com els de senyalització són digitals i per tant, de manera natural, el sistema suporta, com ja s'ha comentat, transmissió de dades. També hi ha una millora en la seguretat de les comunicacions i en la reducció d'interferències aconseguides, entre d'altres motius, gràcies a la tècnica del salt de freqüència.

GSM disposa de quatre versions principals en funció de les bandes utilitzades: GSM-850, GSM-900, GSM-1800 i GSM-1900. GSM-900 (900 MHz) i GSM-1800 (1,8 GHz) i totes elles fan de GSM l'estàndard majoritari al món. Tot i que inicialment en el continent Americà i part d'Àsia es va fer servir l'estàndard IS-95 diferents operadors l'han acabat substituint per GSM, en concret GSM-850 (850 MHz) i GSM-1900 (1,9 GHz). El canvi de banda es deu a que a EUA les bandes de 900 i 1800 MHz són usades per a comunicacions militars. En funció del nombre de bandes que pot fer servir un terminal, rep la denominació dual, tribanda o quadribanda.

4.3. Tecnologia GPRS

El sistema GPRS (*General Packet Radio Service*), és la xarxa cel·lular considerada la generació 2,5, entre la segona generació (GSM) i la tercera (UMTS). El sistema GPRS es basa en l'arquitectura del GSM, però modificant-ne certes parts que permeten que es pugui utilitzar el protocol de xarxa IP, així com proporcionar amples de banda molt més grans i utilitzar aplicacions com, Web, e-mail, ftp, etc.

Gràcies a que només s'han de realitzar petites modificacions al sistema GSM, per tal de poder utilitzar el sistema GPRS i que s'utilitzen els mateixos recursos que GSM, fan que el sistema GPRS sigui una manera viable de millorar el sistema GSM i alhora donar el primer pas cap a l'evolució a UMTS.

Amb la xarxa cel·lular de GPRS es poden suportar tràfics a ràfegues o asimètrics que són més característics de les xarxes IP. A més a més ofereix diferents velocitats, que varien en

funció de la distancia a la que ens trobem respecte a l'estació base. Com ja s'ha comentat suporta el protocol IP, però també d'altres com pot ser X.25. Com les dades van en paquets, es poden aplicar diferents esquemes de tarifació i es millora la transmissió de dades respecte GSM.

Tal com ja s'ha comentat l'arquitectura del sistema GPRS es basa en la de GSM, però amb petites modificacions. Per tal que aquestes modificacions no suposessin una gran modificació a tota la xarxa, es va procurar no modificar la xarxa d'accés i fer aquests canvis a la xarxa troncal.

Per aconseguir no modificar la xarxa d'accés el que es va fer va ser afegir un sistema de commutació de paquets. Amb aquest canvi la xarxa passava a tenir dues parts diferenciades, la de commutació de circuits per GSM i la de commutació de paquets per GPRS.

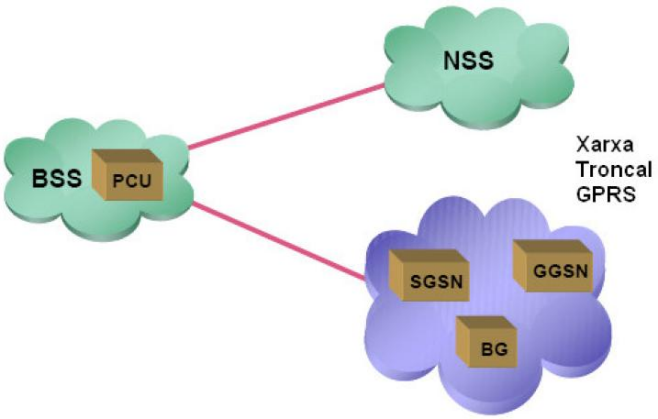

Fig. 4.2. Modificacions a la xarxa GSM

Al sistema de commutació de paquets, la xarxa troncal de GPRS, hi haurà tres nous elements, que són els següents:

- SGSN (*Serving GPRS Support Node*). És el node encarregat de realitzar la commutació de paquets a la xarxa GPRS i des del punt de vista del terminal mòbil suposa el punt d'accés a la xarxa GPRS. Les funcions del SGSN són les següents:
    - Commutació de paquets.
    - Compressió de dades.
    - Xifrat de dades.
    - Autenticació, Gestió de sessions.

  - Proporciona gestió de la mobilitat, on segons l'estat del terminal informa de l'àrea d'encaminament (*routing area*) o de la informació de la BTS.
  - Tarifació: Recull informació sobre la utilització de la xarxa a la interfície radio.
  - Traspàs: S'encarrega de l'enviament del paquets no confirmats pel mòbil, segons el tipus de traspàs al nou SGSN o al mòbil.
- GGSN (*Gateway GPRS Support Node*). És el node encarregat d'interconnectar les xarxes de dades externes amb la xarxa troncal GPRS. Les funcions del GGSN són les següents:
  - Connexió a les xarxes de paquets externes.
  - Tarifació, recull la informació per tarifar l'ús de xarxes externes.
  - Des de les xarxes externes es veu com un *router*, amb el qual intercanvien informació d'encaminament.
  - Pot funcionar com un dispositiu RADIUS (*Remote Access Dial-In User Service*) per connectar amb altres ISPs.
  - Permet associar a l'usuari amb el seu SGSN corresponent.
- BG (*Border Gateway*). És l'element que actua com a punt de connexió amb altres PLMNs (*Public Line Mobile Network*) o Xarxes Mòbils Terrestres Públiques i possibilita l'intercanvi de dades de forma segura sense haver d'utilitzar Internet per interconnectar diferents xarxes. Permet monitoritzar els missatges i acostuma a tenir implementada la funció de *firewall*.

Un cop es té la xarxa dividida en dues parts, cal un element que diferenciï els tipus de paquets que s'envien, es a dir, si és per commutació de circuits o bé per commutació de paquets. L'element que s'encarregarà de diferenciar els paquets serà el *Packet Control Unit* (PCU), que controlarà i gestionarà les funcions relacionades amb la part radio del sistema GPRS.

La PCU pot estar col·locada en diferents ubicacions, la primera d'elles pot ser a la BTS, fet que provocaria que s'haguessin d'instal·lar tantes PCU com BTS hi hagi, fet que suposaria un alt cost econòmic. La segona ubicació i que és la que està adoptada actualment és la de col·locar la PCU a la BSC, així d'aquesta manera la PCU agruparia a varies, BTS. Per últim es podria col·locar abans del SGSN, per tal d'agrupar encara més BTSs en una sola PCU. La solució utilitzada és la segona ja que d'aquesta manera hi ha

una distribució més equilibrada i no es confia tant tota la carrega a una única PCU. A l'apartat de connectivitat es detalla el significat tant de la BTS com de la BSC.

La xarxa GPRS queda distribuïda tal i com es pot visualitzar a la Fig. 4 .3.

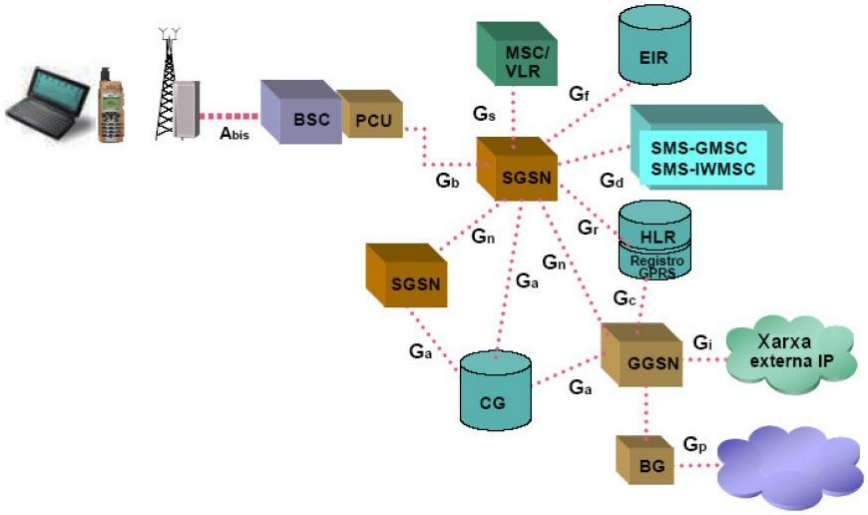

Fig. 4.3. Xarxa GPRS

4.4. Connectivitat GPRS/GSM

Per a poder comunicar el sistema dissenyat amb un dispositiu mòbil o l'ordinador, el sistema ha de tenir una connectivitat. Hi ha moltes i diverses maneres de connectar diferents dispositius entre ells però, com ja s'ha comentat, aquesta connectivitat es farà a través del sistema GPRS/GSM.

Per a explicar com funciona aquesta connectivitat, s'ha d'entendre l'arquitectura d'una xarxa GSM, que és en la que es basa GPRS. Totes les xarxes GSM es poden dividir en quatre parts fonamentals i ben diferenciades:

1. L'Estació Mòbil o *Mobile Station* (MS): Consta de dos elements bàsics que s'han de conèixer, per un costat el terminal o equip mòbil i per l'altre el SIM o *Subscriber Identify Module*. En el cas dels terminals, s'ha de dir que n'hi ha de molts i diversos tipus, i la diferència entre uns i altres radica fonamentalment en la potència que tenen, que va des dels 2 Watts fins els 20 Watts.

En aquest capítol s'utilitzen les referències [5]-[7]

La SIM és una petita targeta intel·ligent que serveix per identificar característiques del nostre terminal. Aquesta targeta s'insereix a l'interior del mòbil i permet a l'usuari accedir a tots els serveis que hi ha disponibles pel seu operador. Sense la targeta SIM el terminal no ens serveix per res ja que no es pot fer ús de la xarxa. La SIM està protegida per un número de quatre xifres que rep el nom de PIN o *Personal Identification Number*. La gran avantatge de les targetes SIM és que proporcionen mobilitat a l'usuari ja que es pot canviar de terminal i utilitzar la mateix SIM. Una vegada introduït el PIN en el terminal, aquest es posarà a buscar xarxes GSM que estiguin disponibles i intentarà validar-se en elles, una vegada que la xarxa (generalment la que ha estat contractada) ha validat el terminal, el telèfon queda registrat a la cèl·lula que l'ha validat.

2. L'Estació Base o *Base Station Subsystem* (BSS): És l'encarregada de la transmissió i recepció i també serveix per a connectar les estacions mòbils amb els NSS (*Network and Switching Subsystem*), els quals aquests últims s'expliquen en el següent punt.

   Com els MS els BSS també consten de dos elements diferenciats: La *Base Transceiver Station* (BTS) o *Base Station* i la *Base Station Controller* (BSC). La BTS consta de transceptors i antenes usades en cada cèl·lula de la xarxa i que solen estar situades en el centre de la cèl·lula, normalment la seva potència de transmissió determina el tamany de la cèl·lula.

   Els BSC s'utilitzen com a controladors dels BTS i tenen com a funcions principals les d'estar a càrrec dels *handovers,* els salts de freqüència i els controls de freqüència de ràdio dels BTS.

3. El Subsistema de Commutació i Xarxa o *Network and Switching Subsystem* (NSS): Aquest sistema s'encarrega d'administrar les comunicacions que es realitzen entre els diferents usuaris de la xarxa; per a poder realitzar aquesta tasca, la NSS es divideix en set sistemes diferents, cadascun amb una missió dins de la xarxa:

   - *Mobile Services Switching Center* (MSC): És el component central del NSS i s'encarrega de realitzar les tasques de commutació dins de la xarxa, així com de proporcionar connexió amb altres xarxes

- *Gateway Mobile Sercies Switching Center* (GMSC): Un *gateway* és un dispositiu traductor (pot ser software o hardware) que s'encarrega d'interconnectar dues xarxes fent que els protocols de comunicació que existeixen entre ambdues xarxes s'entenguin. L'objectiu del GMSC és aquesta mateixa, servir de mediador entre les xarxes de telefonia fixes (PSTN, ISDN PSDN, entre d'altres) i la xarxa GSM.
- *Home Location Registrer* (HLR): El HLR és una base de dades que conté informació sobre els usuaris connectats a un determinat MSC. Entre la informació que emmagatzema el HLR tenim fonamentalment la localització de l'usuari i els serveis als que té accés. El HRL funciona en unió amb el VLR que s'explica a continuació.
- *Visitor Location Registrer* (VLR): Conté tota la informació necessària sobre un usuari perquè aquest accedeixi a tot els serveis de la xarxa. Forma part del HLR amb el que comparteix funcionalitat.
- *Authentication Center* (AuC): Proporciona els paràmetres necessaris per a l'autentificació d'usuaris dins de la xarxa; també s'encarrega de suportar funcions d'encriptació.
- *Equipment Identy Registrer* (EIR): també s'utilitza per a proporcionar seguretat a les xarxes GSM però a nivell d'equips vàlids. La EIR conté una base de dades amb tots els terminals que són vàlids per a ser usats a la xarxa. Aquesta base de dades conté els *International Mobile Equipment Identy* o IMEI de cada terminal, de manera que si un determinat mòbil intenta fer ús de la xarxa i el seu IMEI no es troba localitzat a la base de dades del EIR no podrà fer ús de la xarxa.
- *GSM Interworking Unit* (GIWU): Serveix com a interfície de comunicació entre diferents xarxes per a la comunicació de dades.

4. Els Subsistemes de Suport i Operació o *Operation and Support Subsystem* (OSS): Els OSS es connecten a diferents NSS i BSC per a controlar i monitoritzar tota la xarxa GSM. La tendència actual en aquests sistemes és que, donat que el número de BSS s'està incrementant, es pretén delegar funcions que actualment realitza el subsistema OSS en els BTS de manera que es redueixin els costos de manteniment del sistema.

En aquest capítol s'utilitzen les referències [5]-[7]

A la figura següent es mostra un petit esquema del que s'acaba d'explicar.

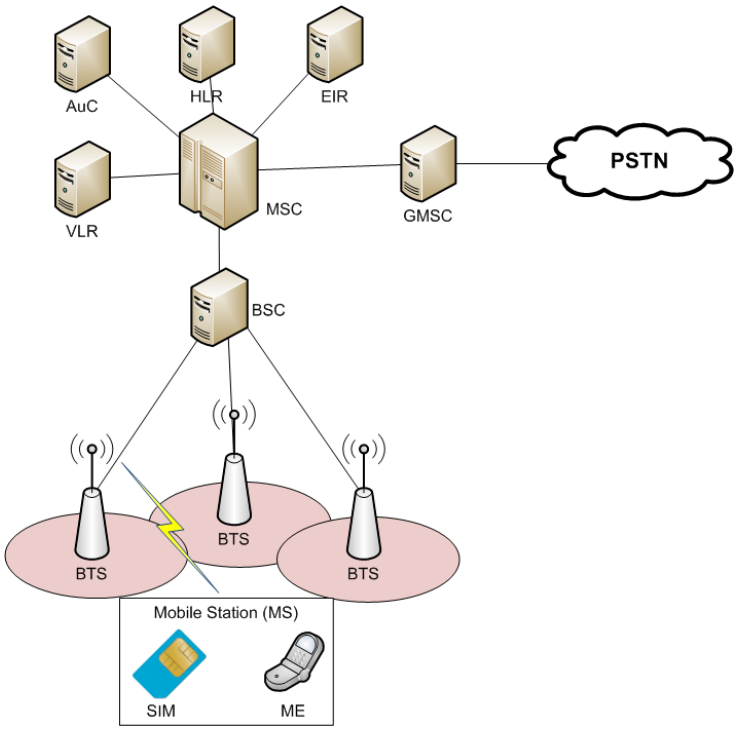

Fig. 4.4. Representació gràfica del funcionament de la connectivitat GSM

Aquesta informació que s'ha detallat serveix per a explicar d'una forma simple com actua el sistema desenvolupat en el projecte. Per a la utilització d'ambdós tecnologies GPS i GPRS/GSM s'ha triat un model existent que ofereix l'empresa Libelium Cooking-Hacks que presenta un mòdul integral constituït pels mòduls GPS i GPRS/GSM de molt fàcil assemblatge amb la placa base (Arduino). Aquest model existent utilitza un sol mòdul que integra GSM+GPS dissenyat per l'empresa SIMCom. Fent servir el GPS Receiver A1080-B de l'empresa Vincotech, en el cas del GPS. I en el cas del GSM, una actualització del SIM900 dissenyat per la mateixa empresa SIMCom però que va ser fet a partir del mòdul HiLO de l'empresa Sagem Communications. El preu d'aquest mòdul GPRS/GSM+GPS és de 99 Euros amb IVA no inclòs.

Al mateix distribuïdor s'hi han trobat les antenes GPS i GPRS/GSM que tenen una integració molt fàcil amb els mòduls. I tenen uns costos de 8 Euros la primera i 6 Euros la segona ambdues amb IVA no inclòs.

4.5. Aplicacions

La plataforma Arduino és d'ús popular i és una de les més utilitzades a nivell mundial, aquest fet ja mostra que permet moltes possibilitats. Les aplicacions que ofereix Arduino són múltiples, i el que es pot aconseguir dependrà de la imaginació i coneixements que es tinguin. Proveïts de sensors es poden crear aplicacions senzilles enfocades a la docència per a estudiants d'electrònica, projectes més elaborats per a la indústria o fins i tot sistemes adreçats simplement a l'oci. Cercant per internet es poden trobar un munt de projectes desenvolupats en Arduino, dels quals se'n mostren alguns a continuació.

El primer exemple del que es pot fer amb Arduino és la creació de robots; es poden fer tota mena de robots, prenent com a base l'Arduino i connectant alguns perifèrics més es poden trobar exemples de robots rastrejadors que són robots que van seguint un camí senyalitzat per exemple amb una línia blanca, robots lluitadors de sumo, robots per jugar a escacs, robots que actuen com a braços mecànics per a agafar objectes, així es podria seguir, ja que la plataforma Arduino té un ventall de possibilitats gràcies, sobretot, a la diversitat de plaques de desenvolupament que ofereix, el qual provoca que es pugui elegir la placa que més s'adapti a les necessitats i objectius que es volen aconseguir.

Com a exemples més concrets es podria trobar un hivernacle controlat per Arduino. Es tracte d'un sistema automatitzat de reg i de control de temperatura. Incloent sensors que activen el sistema només quan sigui necessari.

Una altre aplicació interessant és la domòtica. Amb la plataforma Arduino es podria fer un projecte que consistís en desenvolupar una aplicació per a controlar enllumenat, temperatura, pujada i baixada de persianes, registres de consum i tot el que es vulgui d'una casa.

El Arduino també permet ser controlat fàcilment a través de quasi qualsevol sistema de comunicació sense fils, és per això que incorporant diferents perifèrics (*shields*) es poden tenir molts tipus de connectivitat per a poder interactuar amb el Arduino a distància. Per exemple i depenent dels perifèrics que s'afegeixin es pot tenir comunicació a través de Wi-fi, bluetooth, Xbee, infrarojos, o en el cas del projecte que s'ha desenvolupat GPRS/GSM. Aquests serien alguns exemples de comunicació sense fils que permet Arduino, el qual li dóna unes possibilitats enormes.

En aquest capítol s'utilitzen les referències [5]-[7]

5. Entorn de desenvolupament

Com ja s'ha comentat en anterioritat, el microcontrolador de la placa Arduino es programa mitjançant el llenguatge de programació Arduino (basat en Wiring) i l'entorn de desenvolupament Arduino (basat en Processing).

L'entorn de Desenvolupament (IDE) Arduino està constituït per un editor de textos per a escriure el codi, una àrea de missatges, una consola de text, una barra d'eines amb botons per a les funcions comuns, i una sèrie de menús. Permet la connexió amb el hardware de Arduino per a carregar programes i comunicar-se amb ells.

Per a escriure el software, Arduino utilitza el que es denomina "*sketch*" (programa). Aquests programes són escrits a l'editor de textos. Existeix la possibilitat de tallar/enganxar i buscar/substituir text. A l'àrea de missatges es mostra informació mentre es carreguen els programes i també s'hi mostren possibles errors. La consola mostra el text de sortida a l'entorn de Arduino incloent els missatges d'error complets i altres informacions. La barra d'eines permet verificar el procés de càrrega, creació, obertura i desat de programes, i seguidament es mostra el que permet la monitorització sèrie:

 *Verify/Compile* (comprova el codi en busca d'errors).

 *Stop* (finalitza la monitorització sèrie i amaga altres botons).

 *New* (crea un nou sketch).

 *Open* (presenta un menú de tots els programes (*sketch*) del seu *sketch book* (llibreria de *skecth*), i fent un clic sobre un d'ells, l'obrirà a la finestra actual).

 *Save* (guarda el programa sketch).

 *Upload to I/O Board* (compila el codi i el volca a la placa E/S d'Arduino).

 *Serial Monitor* (Inicia la monitorització sèrie).

En aquest capítol s'utilitza la referència [5]

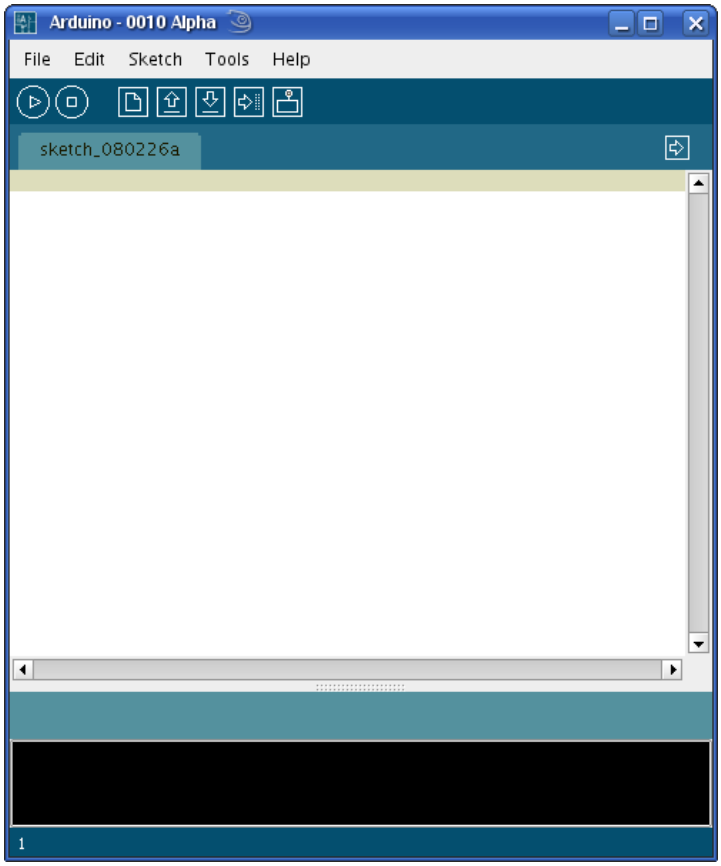

Fig. 5.1. IDE de Arduino

Es poden trobar altres ordres en els cinc menús: *File*, *Edit*, *Skecth*, *Tools*, *Help*. Els menús són sensibles al context, el que significa que només estaran disponibles els elements rellevants per la tasca que s'estigui realitzant en aquell moment concret.

- *Edit*

Copy for Discourse: Copia el codi del seu *skecth* al portapapers per a publicar-lo en un fòrum amb el format adequat, incloent sintaxis colorejada.

Copy as HTML: Copia el codi d'un programa (skecth) al portapapers en format HTML, adequant-lo així, per a poder incrustar-lo en una pàgina web.

- *Skecth*

Verify/Compile: Verifica els errors del programa (sketch).

Import Library: Afegeix una llibreria al seu programa (sketch) inserint la sentència *#include* en el codi.

Show Skecth Folder: Obre la carpeta de programes (*skecth*) a l'escriptori.

Add File...: Afegeix un fitxer font al programa. El fitxer apareix en una nova pestanya a la finestra del programa. Els fitxers es poden treure del programa (*sketch*) utilitzant el menú "tab".

- *Tools*

Auto Format: Dóna format al codi proporcionant estètica, per exemple realitza tabulacions entre l'obertura i el tancament de claus. I les sentències que hagin de ser tabulades ho estaran.

Board: Selecciona la placa que s'està usant.

Serial Port: Aquest menú conté tots els dispositius sèrie (reals o virtuals) de l'equip que s'utilitza. S'actualitzarà automàticament cada vegada que s'obre el menú tools.

Burn Bootloader: Aquest element del menú permet gravar un gestor d'arrancada (*bootloader*) dins del microcontrolador de la placa Arduino. Encara que no és un requisit imprescindible pel funcionament de la placa, pot ser útil si es fa servir un microcontrolador ATmega (el qual normalment no té gestor d'arrancada). S'ha d'assegurar que s'ha seleccionat la placa correcta en el menú *boards* abans de gravar el *bootloader*.

En aquest capítol es comenta només la funcionalitat del programa que es troba instal·lat a l'ordinador amb el qual s'escriu el codi de programació per a posteriorment enviar-lo a la placa de desenvolupament. A l'apartat 6.2. relacionat amb el llenguatge de programació Arduino s'explica de manera més detallada el codi per a programar el microcontrolador ATmega328, fent referència sobretot a les instruccions que es fan servir al projecte, així com les ordres AT que es fan servir pel control del xip (SIM908) del mòdul GPS+GPRS/GSM.

6. Estudi pel desenvolupament de l'aplicació

En aquest capítol s'explica de manera detallada el hardware que s'ha utilitzat per a l'aplicació real del sistema. També es comenta el llenguatge de programació Arduino fent especial referència a les instruccions emprades per a la realització del projecte.

6.1. Hardware

Com ja s'ha comentat en anterioritat, el hardware principal utilitzat en aquest projecte serà la placa de desenvolupament Arduino Uno Rev 3 i el mòdul GPS+GPRS/GSM. De manera general ja s'han comentat algunes característiques del hardware; però a continuació s'explica de manera detallada les parts i el funcionament del hardware que s'utilitza.

6.1.1. Arduino Uno Rev 3

Com a recordatori; cal comentar que Arduino és una placa de circuit imprès simple basada en microcontrolador de codi obert i disposa d'una IDE (entorn integrat de desenvolupament) de font oberta que es pot descarregar de franc. Amb això, la placa que s'utilitza és la Arduino Uno Rev 3 de la qual se'n mostra una imatge a continuació.

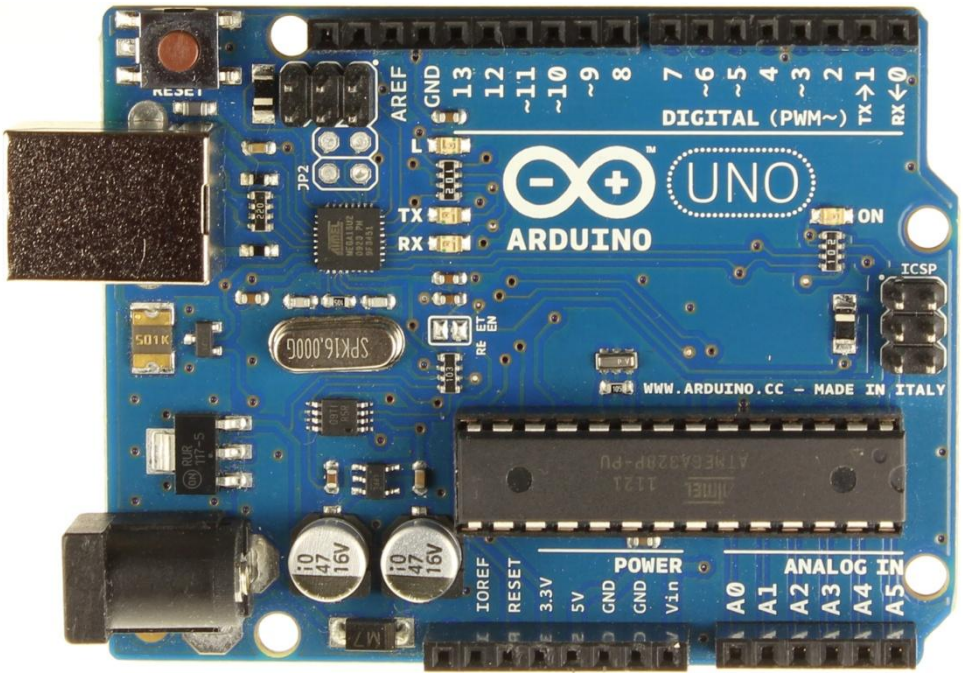

Fig. 6.1. Placa de desenvolupament Arduino Uno Rev 3

En aquest capítol s'utilitzen les referències [8]-[12]

Com es pot apreciar a la imatge, la part de potència es troba a l'esquerre de la placa, mentre que el microcontrolador amb les entrades i sortides es troba a la part dreta.

A continuació es mostra l'esquema de connexions de la placa de desenvolupament que es pot veure de manera més detallada al document de plànols del projecte.

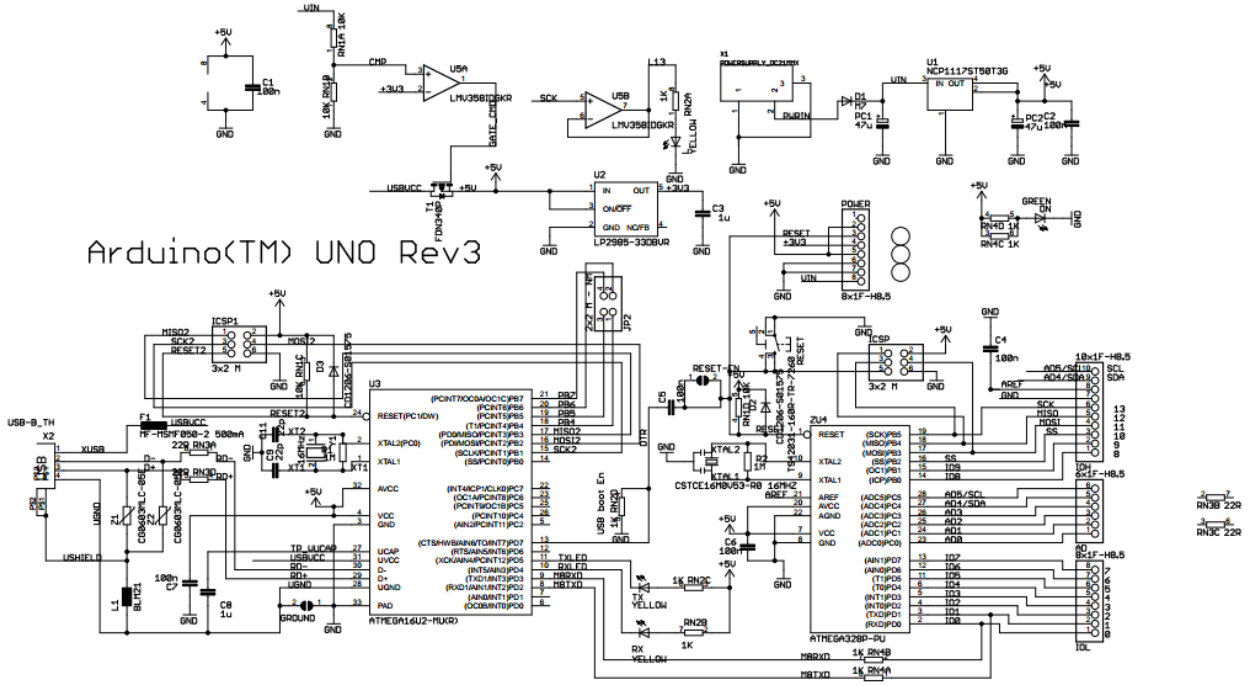

Fig. 6.2. Esquema de connexions Arduino Uno Rev 3

Per a començar amb l'explicació del funcionament i connexions de la placa primer de tot es comentarà com s'alimenta aquesta.

6.1.1.1. Alimentació

La placa pot ser alimentada de diferents maneres; la primera forma d'alimentació és a través del cable USB. Aquest cable està format, realment, per 4 fils més petits: dos per a transmetre dades i dos més per a l'electricitat.

Hi ha dos tipus de connectors USB estàndard. El de Tipus A (o Sèrie A) i el de Tipus B (o Sèrie B).

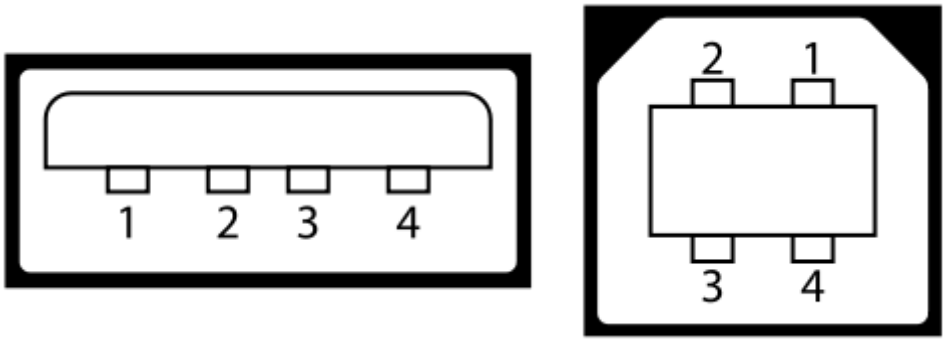

Fig. 6.3. Posició dels pins als connectors de tipus A (esquerra) i B (dreta).

El Arduino fa servir una connexió tipus B, i a part de servir com a alimentació, serveix per a transmetre i rebre dades d'un ordinador, els quals solen tenir una connexió tipus A; per tant el cable que s'utilitza com a alimentació i transmissió de dades pel Arduino és un cable tipus A-tipus B (mascles). Per a l'alimentació del Arduino es fan servir els pins 1 i 4 que s'observen a la figura 6.3. Els quals el pin 1 és el fil d'alimentació de 5 Volts i el pin 4 fa referència a la massa. Els senyals USB es transmeten en els altres dos cables entrellaçats de dades, el D- i el D+ (2 i 3 a la figura 6.3.). Seguidament es mostra una taula resum dels fils que incorpora el cable USB.

Fil	**Funció**
1	V_{BUS} (5 volts)
2	D−
3	D+
4	GND (massa)

Taula 6.1. Fils que incorpora el cable USB i funció de cadascun d'ells.

En resum es pot veure que connectant el cable USB del Arduino a un ordinador, això ofereix alimentació per la placa i a la vegada la possibilitat de poder enviar i/o rebre dades; o sigui una interacció entre el Arduino i un ordinador.

A continuació es mostra la part de l'esquema de connexions relacionat amb l'entrada USB.

En aquest capítol s'utilitzen les referències [8]-[12]

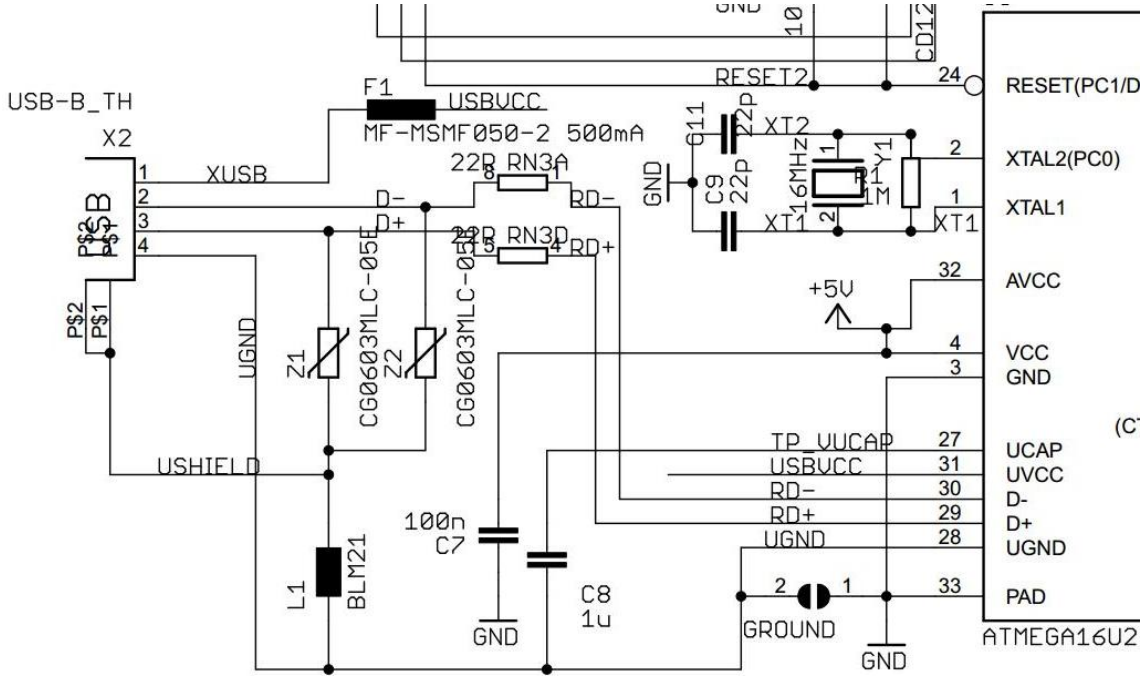

Fig. 6.4. Part de l'esquema de connexions del Arduino dedicat a l'entrada USB.

Si s'observa l'esquema de connexions de la Fig. 6.4., es veu que l'entrada USB, situada a l'esquerra de la figura, és el component X2 (USB-B_TH). En els fils que surten es pot observar com el fil XUSB, que surt del pin 1, correspon a l'entrada de 5 Volts que alimenta la placa, i el fil UGND, que surt del pin 4, correspon a la massa, 0 Volts. Els pins 2 i 3 corresponen als fils D- i D+ respectivament, que com ja s'ha comentat serveixen per a la transmissió de dades. Com es pot observar, l'entrada de voltatge XUSB incorpora un fusible reajustable F1 (MF-MSMF050-2 500mA). La funció del qual és protegir el Arduino contra sobrecorrents i té la capacitat de reaccionar al sobreescalfament produït per un excés de corrent, quan el corrent sobrepassa el corrent de dispar, el dispositiu pot limitar el corrent d'entrada, i quan es refreda torna al seu estat normal. Les especificacions dels ports USB marquen que han d'oferir 5 Volts amb una limitació de corrent de 500mA. És a dir, la sortida del USB de l'ordinador, si compleix les especificacions, ha d'estar protegida contra sobrecorrents de 500mA, és per això que el fusible F1 del Arduino és un component addicional, que serveix per protegir la placa en cas que l'ordinador amb el que es connecta no compleix les especificacions. Hi ha una característica sobre els fusibles reajustables que s'ha de considerar, i és que aquests són lents. El temps que triguen en obrir el circuit està relacionat amb el corrent que circula. Si el sobrecorrent és lleugerament superior al límit, en el cas del Arduino 500mA, pot trigar fins a un segon en obrir-se, tal i com es pot veure a

la fulla de dades del fabricant. Això pot ser un problema en alguns casos, perquè pot danyar l'electrònica de la placa abans que aquesta sigui protegida pel fusible. La connexió XUSB, després de passar pel F1 passar a anomenar-se USBVCC, aquesta connexió arriba fins el MOSFET de l'esquema de connexions, el qual la seva utilitat s'explica més endavant.

Els fils D- i D+ incorporen uns varistors Z1 i Z2 (CG0603MLC-05E), els quals la seva funció és evitar danys per possibles càrregues electrostàtiques. Davant d'una pujada de tensió brusca als cables D- i/o D+, els varistors Z1 i Z2 l'absorbeixen com a càrrega, i així es protegeix el xip ATMEGA 16U2 de possibles danys. Les connexions D- i D+ també incorporen una resistència de 22 Ohms a cada connexió, anomenades RN3A en el D- i RN3D en el D+. El valor d'aquestes resistències i la seva connexió ve donada pel fabricant del ATMEGA 16U2, i s'han d'incorporar per a poder complir amb les especificacions tècniques de càrrega USB i garantir la integritat del senyal que es podria veure afectada per a la impedància de 90 Ohms de la línia D- i D+.

Com es pot observar a l'esquema de la figura 6.4., seguit dels components Z1 i Z2 hi ha connectada una bobina L1 (BLM21), que està connectada al UGND del microcontrolador ATMEGA 16U2. Aquesta bobina fa la funció d'eliminar possibles sorolls externs del USB perquè no entrin al circuit de la placa Arduino a través del seu terra.

A la Figura 6.4. si s'observa l'entrada USB tipus B, o sigui el component X2 (USB-B_TH), s'hi troba unes línies P$1 i P$2, que s'ajunten en una línia (USHIELD) que es connecta a la línia de massa del circuit. Aquestes línies (P$1, P$2) són simplement els dos punts de soldadura del component X2 que es connecten a la línia de massa de la placa segons les directrius de disseny per esquemes d'interfície física USB de l'empresa fabricant de les PCBs de Arduino (Smart Projects Srl).

Si es vol prescindir de la connexió USB, l'alimentació de la placa pot provenir d'un adaptador d'AC-DC. Aquest es pot connectar a l'endoll de 2,1 mm de centre positiu que es troba al Arduino. La placa pot funcionar teòricament, amb un subministrament de l'adaptador de 6 a 20 volts, aquests valors en realitat vénen donats pel fabricant del regulador de tensió que incorpora la placa per a l'entrada d'alimentació externa. Si es proporcionen menys de 7V, però, el circuit de control que incorpora el Arduino podria ser que no funcionés correctament, aquest fet s'explica més avall. I com a contrapartida les

especificacions tècniques del regulador de voltatge de la placa, que serveix per a subministrar 5V (que és el voltatge amb el qual opera la placa) marquen que amb més de 12 Volts a l'entrada es podria superar la màxima potència que pot dissipar, i per tant, es podria sobreescalfar i danyar la placa. És per això que és necessari treballar amb un rang de 7 a 12 volts, quan l'alimentació es fa a través de l'adaptador o qualsevol altre alimentació externa.

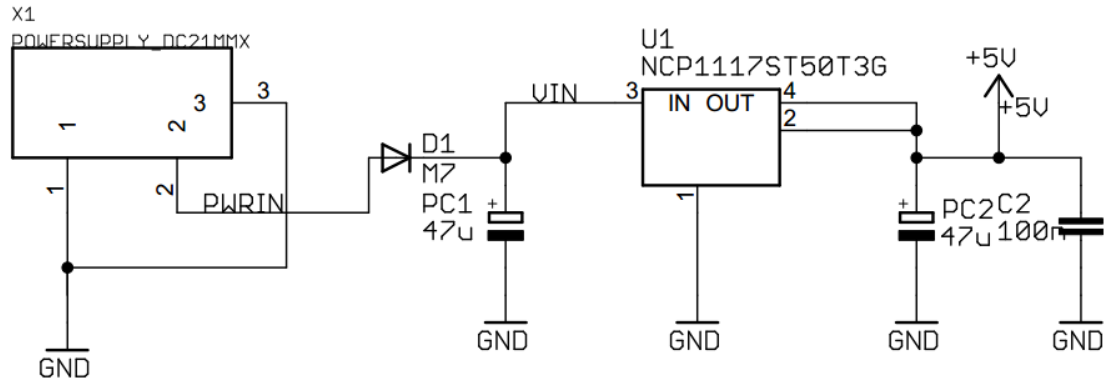

Fig. 6.5. Part de l'esquema Arduino dedicat a l'entrada d'alimentació del adaptador.

A la figura de sobre, es pot veure el component X1 (POWERSUPPLY Y DC21MMX) que correspon a l'entrada per un adaptador AC/DC que incorpora el Arduino. Les línies 1 i 3 del X1 estan connectades a massa, i la corrent de l'adaptador arriba per la línia PWRIN. Aquesta línia incorpora un díode D1 (M7), el qual la seva funció és assegurar que la polaritat del corrent sigui unidireccional entre l'adaptador i el regulador de tensió, i d'aquesta manera protegir el circuit. Seguidament s'hi troba el regulador de tensió U1 (NCP1117ST50T3G) el qual serveix per a rebaixar la tensió d'entrada de l'adaptador fins a 5V, que és el voltatge amb el qual opera la placa. Aquest regulador de tensió incorpora dos condensadors electrolítics connectats, un a l'entrada i l'altre a la sortida del regulador, PC1 i PC2 respectivament, els valors dels quals els recomana el fabricant del regulador. Aquest fet és per a formar un sistema de filtrat, per disminuir possibles efectes de salts o pics de voltatge. El condensador ceràmic C2 que també es pot veure a l'esquema de la figura de sobre, fa la funció de desacoblament, que és explicada més endavant.

Una altra manera amb la qual es pot alimentar la placa Arduino és connectant una alimentació, normalment una bateria, inserint els capçals d'aquesta en els pins GND i Vin del connector d'alimentació.

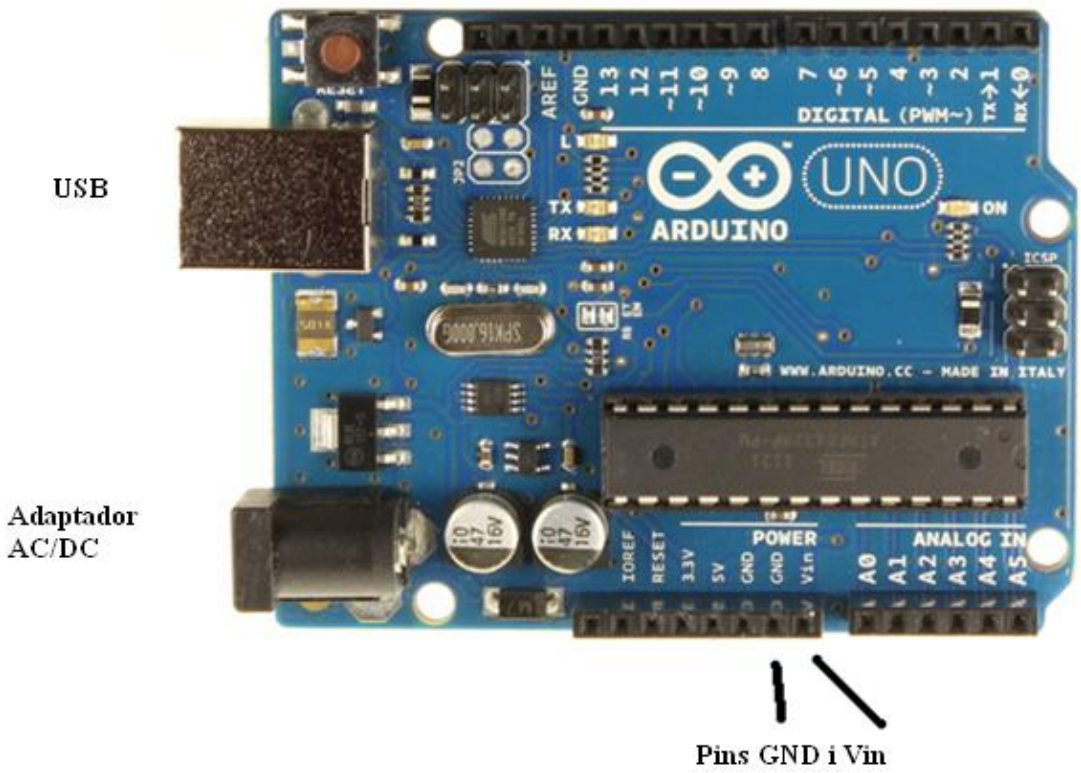

Fig. 6.6. Parts de la placa Arduino relacionades amb l'alimentació d'aquesta.

La font d'alimentació de la placa es selecciona automàticament, això es produeix gràcies a un circuit de control que ofereix la placa, que consisteix en un MOSFET i un xip que incorpora dos amplificadors operacionals.

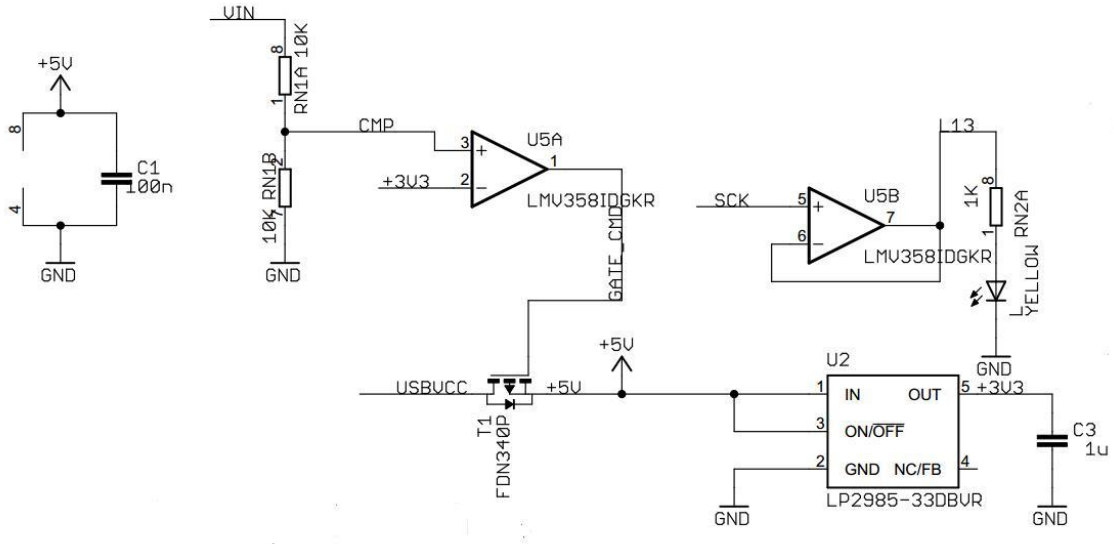

Fig. 6.7. Part de l'esquema de connexions del Arduino dedicat al circuit de control.

En aquest capítol s'utilitzen les referències [8]-[12]

En aquesta part de l'esquema del Arduino Uno Rev 3 es pot observar que si el corrent ens arriba només des de la connexió USB, els 5 volts que proporciona el cable passen pel fil USBVCC i passen pel drenador d'un transistor MOSFET tipus P (component T1). El transistor MOSFET (FDN340P), conduirà deixant passar la corrent d'alimentació del connector USB ja que al no haver-hi alimentació per Vin la base del transistor (GATE_CMD) estarà a massa (GND o 0V) i d'aquesta manera es proporciona la sortida de +5V a través de l'USB. El fet pel qual GATE_CMD proporciona 0V s'explica més avall. Ara bé, si l'alimentació arriba des de la connexió USB i l'adaptador AC/DC o una bateria a la vegada, el circuit de control seleccionarà l'entrada d'alimentació. L'entrada de l'adaptador o de la bateria vindran pel cable Vin. Aquest passarà pel comparador U5A (aquest comparador junt amb el U5B són els dos operacionals del xip U5 de la placa Arduino) que compara el valor de Vin/2 amb 3,3V que esdevenen de la sortida del regulador de tensió U2. Si el valor de Vin/2 és més gran que la tensió de referència que és la del cable de 3V3, l'amplificador operacional U5A proporcionarà una sortida de +5V, que és la tensió positiva d'alimentació de l'operacional, això provocarà que es tingui una tensió positiva a la base del transistor (FDN340P) i significa que el MOSFET estarà obert i tallarà el corrent que passa per USBVCC si n'hi ha, per tant, l'alimentació de la placa Arduino esdevindrà a través de Vin passant pel regulador de tensió U1 obtenint els 5V d'alimentació de la placa, aquest component (U1) s'explica més endavant. En cas contrari, o sigui si la tensió de referència és més gran que Vin/2 o directament no hi ha alimentació externa la tensió de GATE_CMP serà GND. Ja que la resistència RN1B fa la funció "pull down", o sigui quan el circuit està en repòs la caiguda de tensió en aquesta resistència és pràcticament 0V, ja que tenim un nivell lògic baix en el comparador, i això provoca que la tensió de sortida del comparador (GATE_CMP) sigui de 0V. El voltatge de referència es compara amb Vin/2 ja que abans del comparador U5A es té un divisor de tensió format per les resistències RN1A i RN1B de 10K ohms cadascuna. Com que el voltatge de referència és de 3,3V s'ha de tenir en compte que si es vol alimentar la placa per Vin, perquè es detecti que efectivament s'està alimentant el Arduino des d'una font externa el voltatge d'aquesta entrada (Vin) ha de ser com a mínim de 7V ja que es compara 3,3V amb Vin/2.

Com es pot veure a la Figura 6.7. a part del circuit de control amb el U5A i el MOSFET, s'hi troba un altre operacional el U5B, que és el segon operacional del xip U5. Aquest segon operacional és un seguidor de tensió, la tensió del qual arriba des de la pota 19 del

microcontrolador ATMEGA 328P-PU, que correspon al pin digital 13 del Arduino. A la sortida del U5B hi ha connectada una resistència i un LED, que serveixen d'ajuda per a fer proves més visuals de funcionament del Arduino amb el pin 13. Aquest circuit seguidor de voltatge fa la funció de separar la connexió del LED de funcionament amb la sortida digital del Arduino. D'aquesta manera s'evita que qualsevol cosa que es connecti a la sortida digital 13 del Arduino formi un divisor de tensió amb el LED, que està connectat a la mateixa sortida. Ja que la impedància d'entrada del seguidor de tensió és molt alta i fa de separador per no carregar el pin 13 de la placa i el corrent que demandi el LED la proporcionarà l'operacional.

Com es pot observar a la Figura 6.7. El xip U5 està alimentat pel pin 8 amb +5V que provenen de l'alimentació de la placa i pel pin 4 amb 0V o sigui, GND, entre ambdós connexions hi ha connectat un condensador (C1) que fa la funció de desacoblament. Els condensadors de desacoblament es posen en els circuits digitals ja que la seva absència podria provocar un funcionament erràtic del circuit. Les causes d'aquest mal funcionament són, en primer terme, que els circuits digitals commuten en temps molt breus, generant una gran velocitat de canvi en el corrent consumit per la font, i en segon terme, que les línies de connexió de la font i de terra contenen inductàncies paràsites, en les quals es poden generar caigudes de tensió significatives com a conseqüència de la gran velocitat de canvi del corrent. Això podria desencadenar en que el microcontrolador canviés d'estat si el voltatge fluctua massa. Els circuits digitals poden funcionar sense condensadors de desacoblament, però al no col·locar-los s'assumeix aquest risc de generar falles, i el programa del microcontrolador podria executar-se de forma incorrecte, creant la impressió de que existeix un error lògic en el programa, quan en realitat es tracte d'un problema de hardware.

El component U2 (LP2985-22DBVR) de la Figura 6.7. és un regulador de tensió amb un voltatge de sortida de 3,3V el qual es fa servir com a tensió de referència a l'amplificador operacional U5A. El fabricant del regulador recomana incorporar un condensador a l'entrada i l'altre a la sortida d'aquest, amb uns valors determinats, per a tenir un sistema de filtratge de soroll. Els 5V de l'entrada ja han estats filtrats és per això que s'incorpora només un condensador C3 a la sortida.

La placa Arduino UNO Rev 3 incorpora uns pins d'alimentació que s'utilitzen normalment com a sortida per a connectar-hi perifèrics.

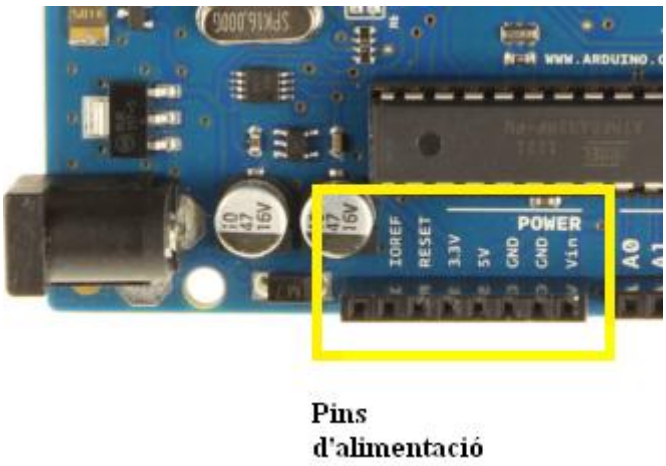

Fig. 6.8. Pins d'alimentació de la placa Arduino UNO Rev 3

Amb l'ajuda de la imatge de sobre, seguidament s'expliquen els diferents pins que incorpora la placa. D'esquerra a dreta:

- Pin 1. El primer pin no es fa servir. Segons explica Arduino, servirà per a futures aplicacions.
- IOREF. El pin IOREF, està connectat a la línia de 5V amb la que opera el microcontrolador ATMEGA 328 P-PU. Aquest pin està pensat per a actuar com a voltatge de referència per a perifèrics connectats a Arduino. I d'aquesta manera el perifèric (shield), sap amb quina tensió treballa el Arduino. Ja que, per exemple, el Arduino UNO Rev 3 opera amb 5V però altres versions de Arduino treballen amb 3,3V. Això està pensat per a què els shields, es fabriquin de manera que estiguin preparats per a operar tant amb 5V com amb 3,3V i d'aquesta manera, sabent el voltatge amb el qual opera la placa Arduino puguin seleccionar el voltatge que convingui.
- RESET. Aquest pin està connectat al reset del microcontrolador principal (ATMEGA 328 P-PU), i serveix per a ser utilitzat com a reset extern de la placa Arduino.
- 3V3. Aquest pin està connectat a la sortida del regulador de tensió U2, per tant, es tracte d'una sortida d'alimentació de 3,3Volts. Les especificacions tècniques indiquen que el consum màxim de corrent en aquest pin no pot superar els 50mA.

Aquest pin també es pot fer servir com a entrada d'alimentació. Però s'ha de tenir en compte que no hi ha cap regulador de tensió; per tant si s'hi connecten més de 3,3V es podria danyar la placa.

- 5V. Aquest pin està connectat a la línia de 5V amb la qual opera la placa Arduino. El voltatge d'aquest pin pot provenir de la connexió USB (5V), del connector d'entrada d'alimentació (7-12V, 5V després de passar pel regulador U1) o del pin Vin de la placa (7-12V, 5V després de passar pel regulador U1). Per tant es tracte d'una sortida d'alimentació de 5V. Igual que el pin 3V3, aquest pin es pot fer servir com a entrada d'alimentació, però s'han de fer les mateixes consideracions que en el pin 3V3, ja que no hi ha cap regulador que atenuï el voltatge que s'aplica a l'entrada d'aquest pin. Les especificacions tècniques indiquen que el consum màxim de corrent en aquest pin no pot superar els 300mA.
- GND. Els dos pins GND estan connectats a massa.
- Vin. Aquest pin està connectat amb la línia Vin de la Figura 6.5. i es pot utilitzar com a entrada per a connectar-hi una font d'alimentació externa (per exemple una bateria, com s'ha comentat anteriorment) entre aquest pin i el pin GND, tenint en compte les especificacions de voltatge mínim i màxim per a una alimentació externa de la placa, que ha de ser d'un rang comprès entre 7 i 12 Volts. O també es pot utilitzar com a sortida d'alimentació, el qual obtindrem el voltatge provinent del connector d'alimentació de corrent continu (X1).

El Arduino també incorpora un LED de funcionament GREEN ON (LEDCHIP-LED0805) que serveix com a referència visual per a saber si estan arribant els 5V d'alimentació correctament.

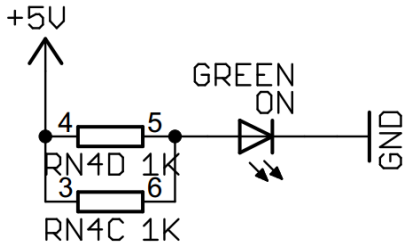

Fig. 6.9. LED de funcionament de la placa Arduino.

El fet que hi hagi dues resistències en paral·lel és perquè, segons els consums del fabricant del LED, la resistència recomanada pel funcionament d'aquest hauria de ser d'un valor al voltant de 500 Ohms, i les resistències de la placa són de 1K Ohm.

6.1.1.2. Comunicacions

Per a la comunicació entre Arduino i PC es fa servir una entrada USB com ja s'ha comentat anteriorment. Les dades de transmissió i recepció amb les quals es fa la comunicació venen per les línies D- i D+. Però s'ha de tenir en compte que la transmissió de dades per USB fa servir el seu propi protocol. Per exemple en el USB 1.1 fa servir polsos de 0 a 0,3V per a nivells baixos (zeros), i de 2,8 a 3,6 per a nivells alts (uns). En canvi el microcontrolador principal (ATMEGA 328) funciona amb polsos TTL que fan servir 0V per nivells baixos i 5V per nivell alts. És per aquest fet que la placa Arduino UNO Rev 3 incorpora un microcontrolador, ATMEGA 16U2-MU, que està programat com a convertidor de polsos USB a TTL.

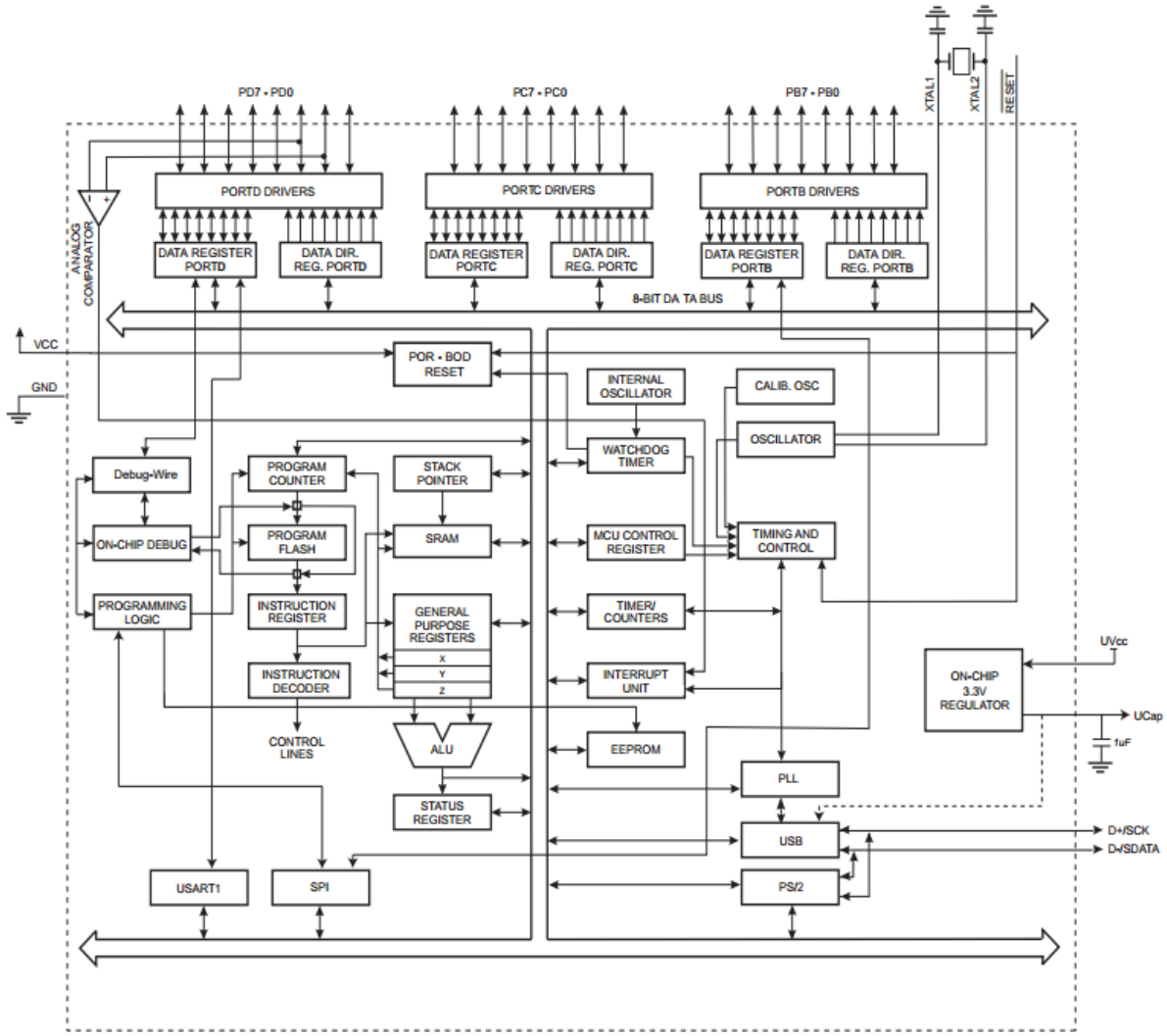

Fig. 6.10. Diagrama de blocs de les parts del microcontrolador ATMEGA 16U2

Observant la figura de sobre, es poden veure tots els pins que componen el microcontrolador:

- VCC. Aquest pin correspon a l'entrada d'alimentació del ATMEGA 16U2.
- GND. Correspon al pin de massa.
- AVCC. És el pin d'entrada d'alimentació per a totes les funcions analògiques del microcontrolador. Com per exemple el xip PLL (modulador de freqüències) que incorpora. Aquest modulador es fa servir pel controlador USB que hi ha al xip, ja que aquest últim requereix un rellotge de referència de 48MHz per a compilar els polsos de la transmissió de dades USB. El PLL, està programat per a obtenir aquesta freqüència interna de 48MHz, i per això, també ha de tenir un rellotge de referència que, com es pot veure a la Figura 6.10, és extern, però s'ha de tenir en

compte que el ATmega16U2 també incorpora un oscil·lador intern de 8MHz, que està desactivat per software, per a poder utilitzar l'extern de 16MHz.

- Port B (PB7..PB0) i Port C (PC7..PC0). Són 16 pins bidireccionals (8 del Port B i 8 del Port C), o sigui tant poden ser entrades com sortides.
- Port D (PD7..PD0). Aquest port de 8 pins es fa servir per a connectar-hi entrades analògiques. Però també es poden fer servir com a pins bidireccionals, o sigui tant per entrades com a sortides.
- D-. És l'entrada de la línia de transmissió negativa del USB.
- D+. És l'entrada de la línia de transmissió positiva del USB
- UGND. Aquí s'hi connecta la línia de massa del USB.
- UVCC. En aquest pin s'hi connecta l'entrada d'alimentació del USB.
- UCAP. El microcontrolador incorpora un regulador de voltatge de 3,3V en el seu interior. Aquest pin correspon al voltatge de sortida d'aquest regulador, i segons el fabricant s'hi ha de connectar un condensador de 1µF.
- RESET. Aquest pin correspon a l'entrada de reset. Si es detecta un nivell baix durant el temps indicat pel fabricant es generarà un reset encara que el rellotge no s'estigui executant.
- XTAL1 i XTAL2. Són els dos pins on es connecta el cristall que dóna els polsos de rellotge al microcontrolador.

Per a complir amb les especificacions elèctriques del USB, els pads (connexions) D- i D+ han de ser alimentats amb un rang de 3 a 3,6 Volts. El ATMEGA 16, pot ser alimentat fins a 5,5V; és per això que s'incorpora un regulador de tensió intern de 3,3V, per a poder alimentar les patilles USB amb el rang que s'especifica. En el cas de l'esquema de connexions del Arduino UNO Rev 3 (Figura 6.2.) es pot comprovar que el microcontrolador ATMEGA 16U2 s'alimenta amb una tensió de 5V. Les connexions d'alimentació vénen donades pel fabricant del microcontrolador que recomana el següent circuit per a l'alimentació a 5V d'aquest.

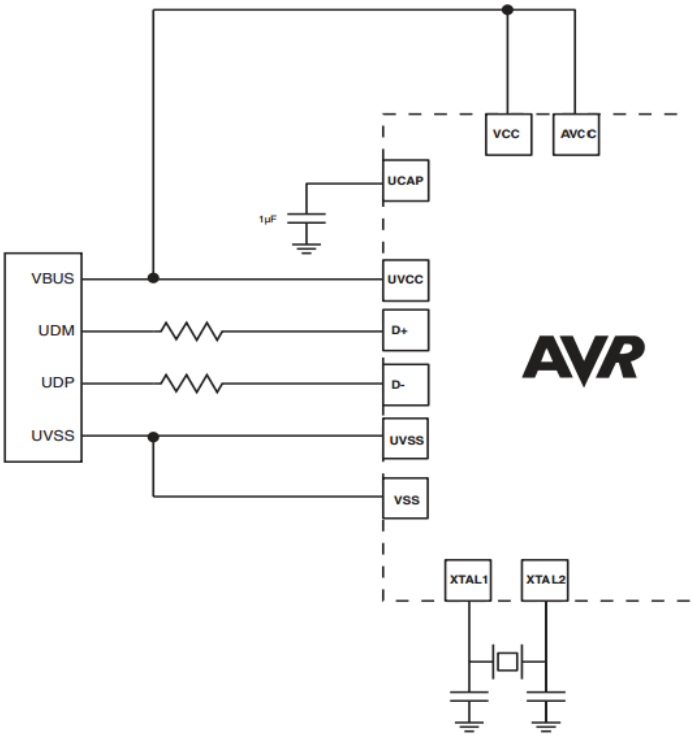

Fig. 6.11. Esquema de connexions recomanat per l'alimentació a 5V del Atmega 16

Com es pot observar; l'alimentació de 5V de VCC arriba des del cable de 5 Volts del USB. Seguidament es mostren les connexions tal i com estan posades al Arduino.

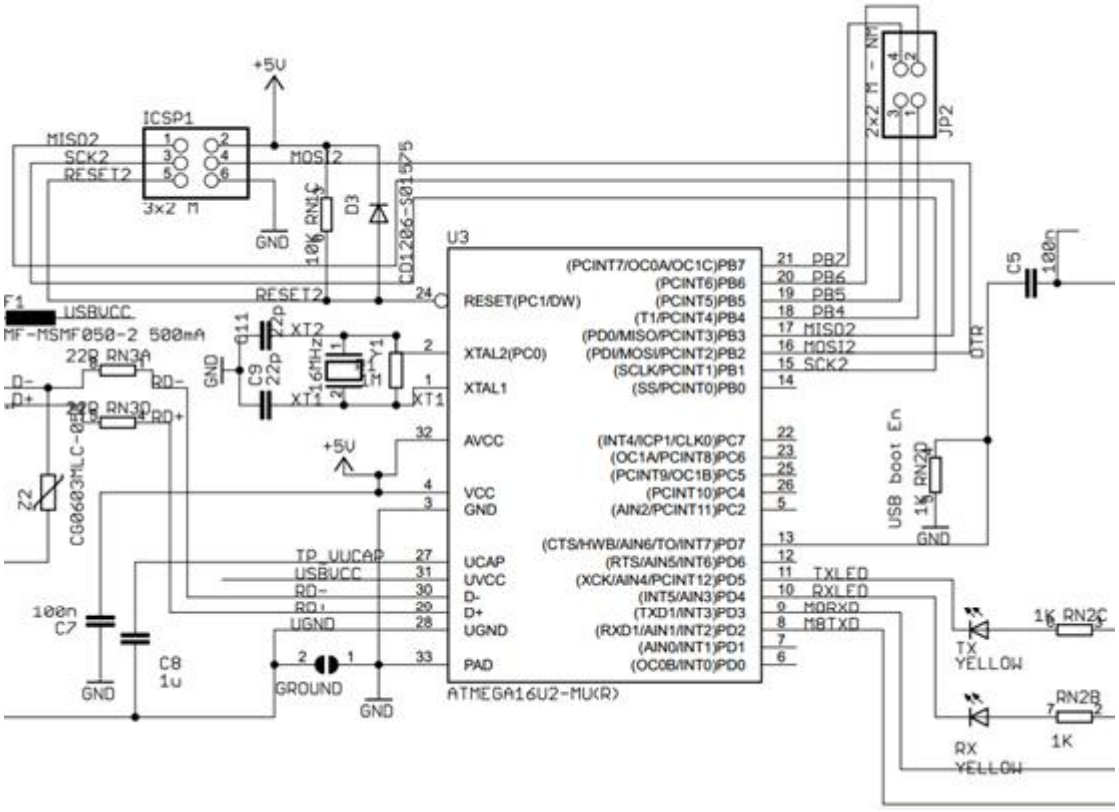

Fig. 6.12. Part de l'esquema de connexions Arduino, destinat al Atmega 16U2

Com es pot observar a la figura de sobre, el circuit d'alimentació és com el de la Figura 6.11. amb la diferència que la línia de 5 Volts de VCC i AVCC no està connectada directament a l'entrada del USB, sinó que està connectada després del MOSFET T1 que incorpora el Arduino. I també s'hi pot apreciar que hi ha connectat un condensador de desacoblament C7 a la mateixa línia.

També es pot apreciar l'oscil·lador Y1 de 16MHz que incorpora la placa, amb dos condensadors C9 i C11 de 22pF. Les connexions de l'oscil·lador i els condensadors venen donades pel fabricant del microcontrolador, així com els seus valors corresponents.

Com s'ha comentat anteriorment, la funció d'aquest microcontrolador és només la de convertir els polsos de USB a TTL, perquè el microcontrolador principal de la placa Arduino, el ATMEGA 328, pugui comunicar-se amb el PC. Aquests polsos TTL (0-5V) surten dels pins PD3 i PD2 del Atmega 16, i van connectats al PD1 i PD0 del Atmega 328. D'aquesta manera el que s'aconsegueix és que els polsos que vénen per el USB es transformen en el microcontrolador Atmega 16 i surten en format TTL perquè pugui interpretar-los el Atmega 328.

Per a fer aquesta conversió; l'equip de Arduino instal·la un programa (firmware) al microcontrolador ATMEGA 16U2 per a poder realitzar aquest canvi de polsos. Aquest firmware es pot descarregar lliurament i instal·lar en el cas de que hom vulgui obtenir aquest microcontrolador sense la placa Arduino i utilitzar-lo com a convertidor USB-TTL.

El pin PAD que es pot veure a la placa és la part física que serveix de suport per a connectar la placa a la PCB i que quedi fixe; el qual el fabricant recomana connectar amb el terra del circuit; és per això que hi ha aquesta connexió. La massa de la placa és una massa diferent que la que arriba des del USB. La placa Arduino incorpora un jumper de soldadura (un jumper és un element que permet interconnectar dos terminals de manera temporal, un Jumper de soldadura, significa que per a poder interconnectar ambdós terminals s'han de soldar de manera que quedin tocant un amb l'altre per a fer possible la connexió) per a poder connectar el terra provinent del USB (UGND) amb el terra general del circuit (GND).

La placa Arduino també incorpora uns LEDs (TX YELLOW i RX YELLOW), connectats a PD5 i PD4 del microcontrolador Atmega 16, que parpellegen quan es transmeten dades entre la connexió USB del ordinador i el xip Atmega 16. Per tant, serveixen de referència per a saber si hi ha transmissió entre l'ordinador i el microcontrolador esmentat; però s'ha de tenir en compte que no serveix com a prova visual de si hi ha transmissió de dades entre ambdós microcontroladors, el Atmega 328 i el Atmega 16.

La línia PD7 és una línia que connecta amb el reset del Atmega 328, la funció de la qual s'explica més endavant, així com l'explicació del circuit de reset general de la placa.

Com es pot observar a l'esquema de la Figura 6.12. hi ha unes línies del Atmega 16 que van a uns connectors anomenats ICSP1 on s'hi troben les ja conegudes línies de 5 Volts i GND, i també la línia de reset que s'explicarà més endavant. Però també incorpora tres connectors més el MISO, MOSI i el SCK, es pot observar que les línies en realitat es diuen MISO2, MOSI2 i SCK2, aquest dos al final és degut a que la placa Arduino incorpora dos ICSP, el ICSP1 pel Atmega 16 i el ICSP pel Atmega 328. Les línies MISO2, MOSI2 i SCK2 venen dels pins del Atmega 16 PB3, PB2 i PB1, respectivament. El ICSP (In-Circuit Serial Programming), és una forma que permet programar xips quan es troben en un circuit integrat. Al Arduino això permet programar els dos xips el Atmega 328P i el Atmega 16U2 (per això hi ha dos ICSP, un per a cada xip), directament amb les instruccions AVR, que

són les instruccions per defecte dels microcontroladors de la placa, sense la necessitat de que en el xip hi hagi instal·lat l'entorn de programació Arduino, en el xip Atmega 328P o el firmware per a convertir polsos USB-TTL en el cas del Atmega 16U2. D'aquesta manera l'equip Arduino pot instal·lar els gestors d'arrancada de manera senzilla, després de fabricar les plaques i abans de vendre-les, perquè el Atmega 328P pugui funcionar amb la IDE del Arduino i el Atmega16U2 faci la funció de convertidor de polsos, i també facilitin a l'usuari la possibilitat de poder actualitzar la última versió del bootloader del Arduino. En el cas que l'empresa Arduino tregui una nova versió i l'usuari hagi comprat la placa amb una versió del gestor d'arrancada antiga.

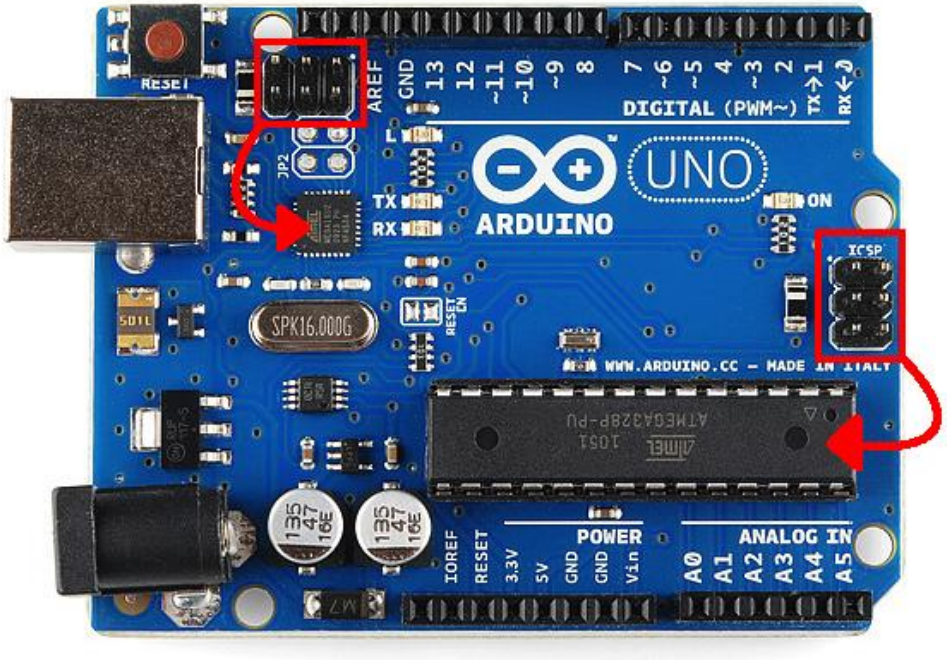

Fig. 6.13. Connectors ICSP de la placa Arduino UNO Rev 3.

A la imatge de sobre es pot veure la distribució dels ICSP a la placa i quin correspon a cadascun dels dos microcontroladors Atmega (ATMEGA 16U2 a l'esquerra de la Figura 6.13. i ATMEGA 328P a la dreta).

A la Figura 6.12. també s'hi poden veure uns jumpers anomenats JP2 amb els quals teòricament es podrien connectar els pins del Atmega 16 PB7 i PB5, amb un jumper, i PB6 i PB4 amb l'altre. Aquests jumpers estan preparats per a futures aplicacions del Atmega 16 que no s'esmenten.

6.1.1.3. ATmega 328P-PU

Aquest microcontrolador, és el xip principal de la placa Arduino UNO Rev3, i és on s'hi carreguen els programes per a treballar amb el Arduino. Seguidament es mostra una imatge amb els 28 pins que incorpora, la funció dels quals s'explica amb detall més endavant.

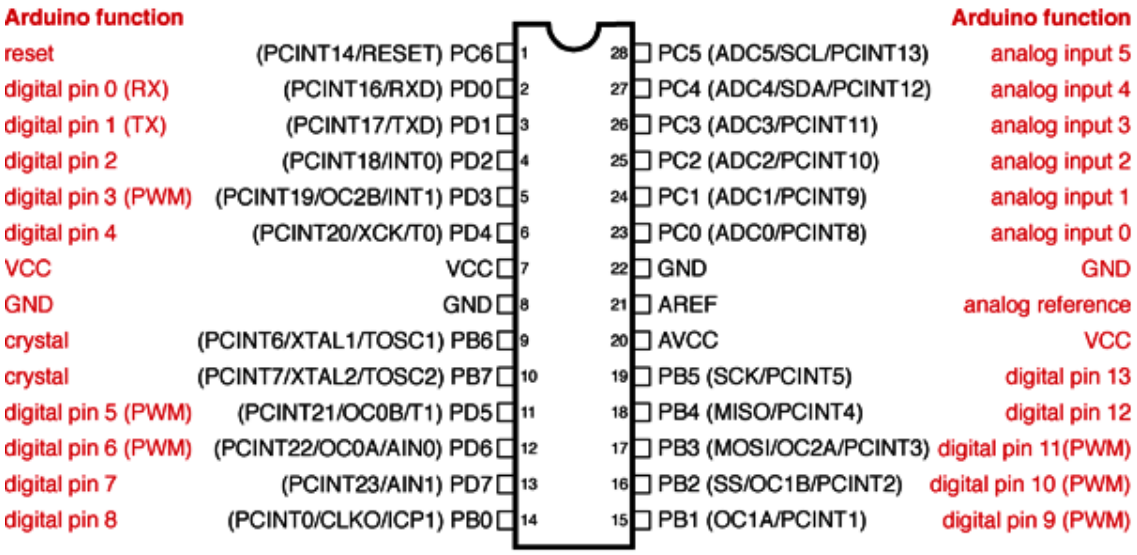

Fig. 6.14. Distribució dels pins del ATmega328P-PU i funció de cadascun d'ells.

El microcontrolador ATmega328 segueix l'estructura AVR. Els AVR són una família de microcontroladors RISC de Atmel. RISC (Computador amb conjunt d'instruccions reduït) és un tipus de microprocessador que reconeix un nombre típicament reduït d'instruccions de codi màquina, implementades per hardware directament en la CPU. Amb això s'aconsegueix eliminar microcodi i la necessitat de descodificar instruccions complexes. D'aquesta manera la CPU treballa més ràpid al utilitzar menys cicles de rellotge per executar instruccions.

El AVR és una CPU d'arquitectura Harvard. Aquesta arquitectura utilitza dispositius d'emmagatzematge físicament separat per les instruccions i per les dades. El AVR té 32 registres de 8 bits. Algunes instruccions només operen en un subconjunt d'aquests registres. La concatenació dels 32 registres, els registres d'entrada/sortida (I/O) i la memòria de dades conformen un espai de direccions unificat, al qual s'hi accedeix a través d'operacions de càrrega/emmagatzematge. A diferencia d'altres tipus de

En aquest capítol s'utilitzen les referències [8]-[12]

microcontroladors, com ara els PIC, la pila s'ubica en aquest espai de memòria unificat, i no està limitat a un tamany fixe.

El AVR va ser dissenyat des d'un principi per a l'execució eficient de codi C compilat. Com que aquest llenguatge utilitza, de manera repetida, punters per a la gestió de variables a la memòria, els tres últims parells de registres interns del processador són usats com a punters de 16 bits a l'espai de memòria externa, amb els noms de X, Y i Z. Això és un comportament que es fa servir en estructures de 8 bits ja que el tamany de paraula natiu de 8 bits (256 localitats accessibles) és pobre per direccionar (especificar un operant dins d'una instrucció). D'altra banda, fer que tot el banc superior de 16 registres de 8 bits tingui un comportament altern com un banc de 8 registres de 16 bits, complicaria molt el disseny i violaria la premissa original de simplicitat que caracteritza aquesta estructura.

El conjunt d'instruccions AVR està implementat físicament i disponible al mercat en diferents dispositius que comparteixen el mateix AVR però tenen diferents perifèrics i quantitat de memòria RAM i ROM (RAM memòria de lectura i escriptura i ROM memòria només de lectura).

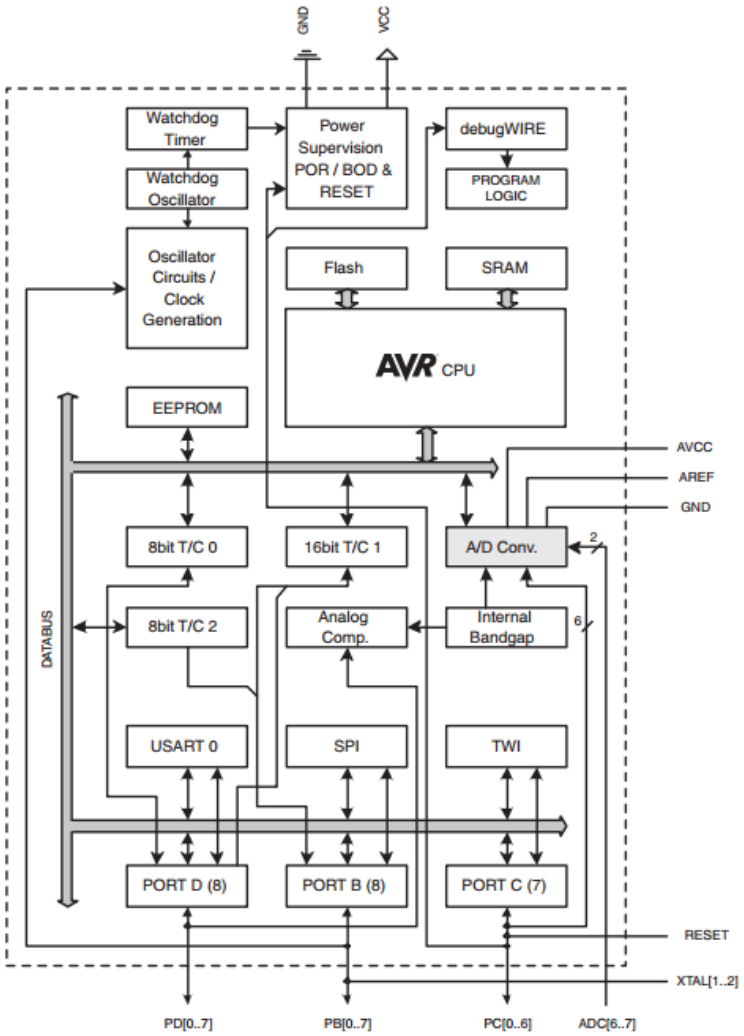

Fig. 6.15. Diagrama de blocs de l'estructura interna del ATmega328P-PU

Els microcontroladors AVR tenen un *pipeline* amb dues etapes (carregar/executar). Un *pipeline* de dues etapes vol dir que es podrà realitzar una instrucció completa per cada cicle de màquina. Cada etapa dura un cicle, per tant quan una instrucció entra, realment trigarà dos cicles a sortir, un cicle per carregar i l'altre per executar-se, però si quan una instrucció surt, simultàniament n'hi posem una altre, encara que la mateixa instrucció estigui en el *pipeline* dos cicles, com a mitjana sortirà una instrucció cada cicle de màquina. Aquest fet comporta que els microcontroladors AVR siguin relativament ràpids entre els microcontroladors de 8 bits.

El ATmega328P-PU té 32KB de memòria flash, 2KB de memòria SRAM i 1KB de memòria EEPROM. Com es pot comprovar són tres fonts de memòria diferents. La memòria flash és on es guarden els programes que es fan amb la IDE de Arduino. Es tracte

d'una memòria no volàtil regravable. Això significa que el seu contingut encara hi serà si s'apaga l'alimentació. Es podria dir que és com el disc dur de la placa Arduino. El seu programa s'emmagatzema aquí. El gestor d'arrencada que està instal·lat al ATmega328 perquè es pugui programar el microcontrolador amb la IDE de Arduino està instal·lat aquí i ocupa 0,5KB dels 32KB que disposa la memòria flash. Aquesta memòria, segons el fabricant, pot suportar fins a 10000 escriptures o cicles de càrrega, o sigui que s'hi poden carregar programes realitzats amb la IDE de Arduino fins a 10000 vegades. La memòria SRAM és com la memòria RAM de l'ordinador , el seu contingut desapareix quan s'apaga la unitat, però s'hi pot llegir i escriure molt ràpid. Cada variable del programa realitzat amb la IDE de Arduino i carregat al microcontrolador es manté a la memòria RAM d'aquest últim mentre el programa s'executa, però si es treu l'alimentació el seu contingut no queda guardat. La memòria EEPROM és una memòria no volàtil regravable que normalment s'utilitza per emmagatzemar informació de llarg termini. Aquesta memòria, segons el fabricant, pot suportar fins a 100000 escriptures.

S'ha de tenir en compte que les memòries són limitades. Per exemple en la memòria SRAM, és fàcil utilitzar-la tota al tenir moltes cadenes de caràcters (Strings) en el programa. Per exemple una declaració com:

Char missatge[ ] = "Projecte final de grau."

Posa 23 bytes a la SRAM (cada caràcter utilitza un byte). Això pot no semblar gaire, però no és tant difícil arribar fins a 2048, especialment si es té una llarga quantitat de text que s'ha d'enviar (com és el cas del programa que es desenvolupa per a fer l'aplicació que es realitza en el projecte, on s'hi llegeixen SMS i s'envia la posició GPS). En el cas que es superés la SRAM disponible, el programa podria executar-se de manera incorrecta. Com a solució a aquest problema hi ha unes llibreries disponibles que permeten guardar variables a la memòria flash si no s'han de modificar les dades de la variable guardada durant l'execució del programa o a la memòria EEPROM.

Com ja s'ha comentat anteriorment, el ATmega328P-PU és el microcontrolador principal del Arduino UNO Rev 3, i és on s'hi executen els programes. Seguidament es mostra l'esquema de connexions d'aquest microcontrolador a la placa.

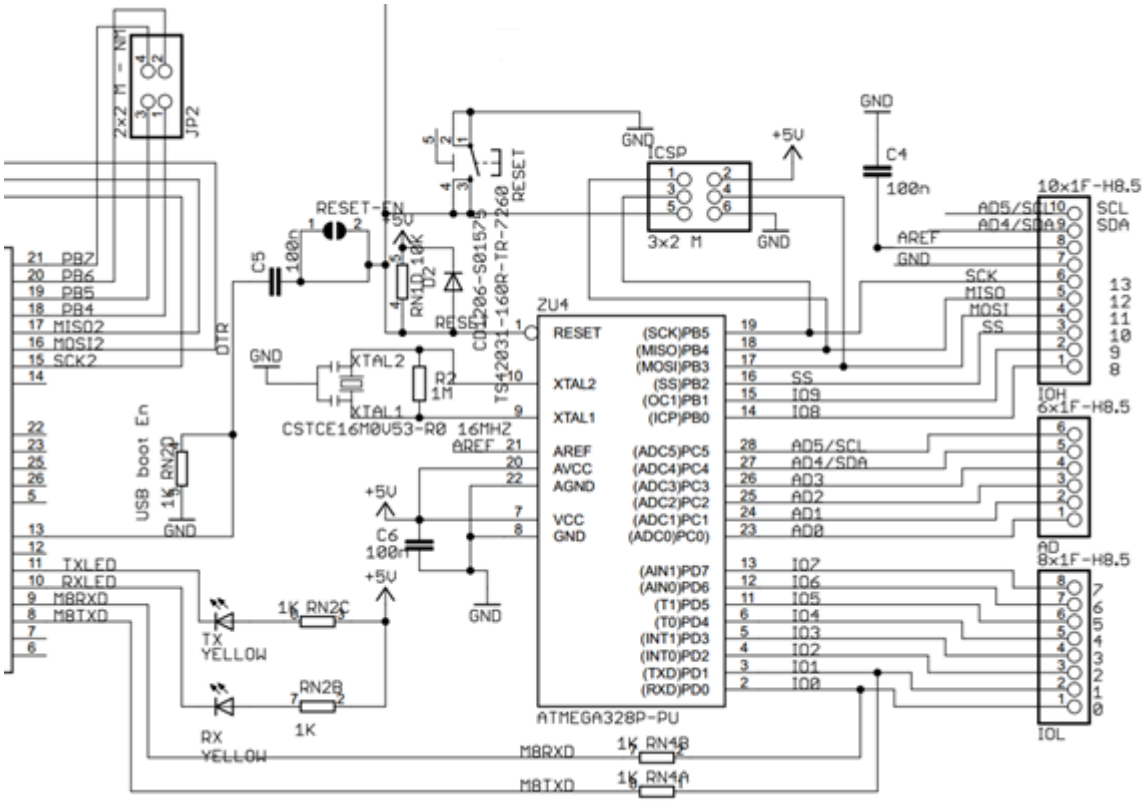

Fig. 6.16. Part de l'esquema de connexions Arduino destinat al ATMEGA328P

El component ZU4 correspon al ATMEGA328P-PU; es pot observar que la placa s'alimenta a través del pin VCC amb un voltatge de 5 Volts que pot arribar des de les diferents formes d'alimentació que permet el Arduino. El rang del voltatge d'entrada VCC d'aquest microcontrolador, especificat pel fabricant, ha d'estar comprès entre 1,8 i 5,5 Volts. A la línia de 5 Volts hi ha connectat un condensador, recomanat pel fabricant, que fa la funció de filtrar possible soroll. També es pot comprovar que el pin AVCC està connectat a la línia de 5 Volts. Aquest pin es connecta internament amb el pin AREF que serveix per a indicar quin és el voltatge de referència usat per a les entrades analògiques. Tenint en compte que AVCC està connectat per defecte a la línia de 5 Volts, el voltatge de referència serà de 5 Volts. Quan es llegeixi una entrada analògica, el valor d'aquesta serà de 0 a 1023. Tenint en compte que el voltatge de referència és 5 Volts per defecte; quan es tingui un valor de 1023 a l'entrada analògica, això es traduirà amb 5 Volts. El microcontrolador ATmega328, també incorpora una referència de tensió interna de 1,1 Volts, la qual es pot activar per software, per si es vol tenir una precisió més acurada dels valors d'una entrada analògica. En el pin AREF també s'hi pot configurar per software una tensió de referència externa, i d'aquesta manera introduir al connector 8 del IOH (10x1F-

H8.5) una tensió que es farà servir com a tensió de referència. D'aquesta manera, si s'hi introdueixen 4 Volts, quan es tingui un valor de 1023 en una entrada analògica, es traduirà amb 4 Volts, i de la mateixa manera, si a l'entrada s'hi llegeix un valor de 512, s'obtindran 2 Volts. Es pot apreciar que a la línia AREF que arriba al connector 8 hi ha connectat un condensador, recomanat pel fabricant del microcontrolador, perquè la línia sigui més immune al soroll.

Als pins XTAL1 i XTAL2 (pins 9 i 10 respectivament) del ATmega328 hi ha connectat un ressonador ceràmic CSTCE16M0V53-R0 de 16MHz per a donar els polsos de rellotge al microcontrolador. Aquest ressonador porta uns condensadors interns de 15pF, és per això que encara que els condensadors surten dibuixats a l'esquema de la Figura 6.16. no tenen cap referència associada. La resistència R2 de 1M Ohm connectada en paral·lel amb el ressonador és recomanada pel fabricant d'aquest. El ATmega328 també incorpora un oscil·lador intern de 8MHz, que es pot habilitar per software, però per defecte a les plaques Arduino es fa servir l'extern de 16MHz.

El microcontrolador ATmega328 també incorpora un pin de Reset per a poder reiniciar la placa externament en un moment donat. Els Reset a efectes pràctics del senyal és posar el comptador de programa (PC) en un valor predeterminat (per exemple, PC=0), provocant així que el microcontrolador comenci a executar les instruccions que hi ha a partir d'aquesta posició de memòria apuntada pel PC. Els Resets dels dos microcontroladors de la placa Arduino, ATmega328 i ATmega16U2, activen el Reset quan interpreten un nivell baix en aquest pin. El Reset té especificada pel fabricant una tensió llindar. Quan la tensió que s'aplica al pin de Reset és superior a la tensió llindar; el terminal interpretarà un valor lògic alt (RESET=1), d'aquesta manera el microcontrolador no es reinicia. En el cas contrari, si la tensió que s'aplica al pin de RESET és inferior que la tensió llindar, el terminal interpretarà un valor lògic baix (RESET=0) i el microcontrolador es reiniciarà.

Si s'observa la Figura 6.16. es pot comprovar que hi ha unes línies de connexió que van a parar al terminal de Reset i serveixen per a poder reiniciar el Arduino. El voltatge que s'aplica a la tensió de Reset, que és més gran que el de la tensió llindar del terminal, ve donat per la línia de 5 Volts passant per la resistència de 10K Ohms (RN1D) que es pot veure a la Figura esmentada. Si es vol fer un Reset manual; la placa incorpora un polsador normalment obert anomenat RESET (TS42031-160R-TR-7260) que al polsar-lo durant un

període mínim de 2,5µS, dades donades pel fabricant, provoca que la tensió del pin Reset es connecti directament a massa i d'aquesta manera sigui inferior a la tensió llindar i proporcioni un valor lògic baix al terminal que provocarà un Reset.

La placa també incorpora un circuit per l'anomenat Power-on Reset (Reset d'engegada). Com es pot observar, a l'esquema de connexions hi ha connectat un díode D2 (CD1206-S01575) en paral·lel amb la resistència. Quan la placa Arduino està alimentada a 5 Volts, el díode no deixa passar corrent i fa la funció de circuit obert ja que la tensió del pin Reset és menor a la tensió d'alimentació de la placa. En el cas que es deixi d'alimentar la placa; la tensió del pin Reset serà més gran que la tensió d'alimentació i el díode conduirà. D'aquesta manera s'aconsegueix que la tensió en el pin Reset sigui més baixa que la tensió llindar i això provocarà un Reset al microcontrolador. D'aquesta manera quan es deixa d'alimentar la placa Arduino es provoca un Reset, i així, quan es torni a alimentar, el microcontrolador començarà a executar les instruccions del programa des del principi marcat pel PC. Aquest mateix tipus de Reset (Power-on Reset) s'aplica en el microcontrolador ATmega16U2, perquè aquest es reinicïi quan es desconnecta l'alimentació del Arduino. Si s'observa la Figura 6.12. es pot veure la connexió del Power-on Reset. Des de la línia de 5 Volts i passant per una resistència de 10K Ohms (RN1C), es proporciona la tensió en el pin Reset, i quan es desconnecta l'alimentació, la tensió d'aquest pin disminueix a través del díode D3 (CD1206-S01575).

El microcontrolador ATmega328P-PU també es reseteja quan s'hi carrega un programa a través de l'ordinador, aquest Auto-Reset, és possible gràcies a la línia DTR que surt del ATmega16U2. Aquesta línia quan no hi ha Reset està a nivell alt; però quan hi ha un Auto-Reset, o sigui quan es carrega un programa al Arduino, el ATmega16U2 es reseteja automàticament per software i com a conseqüència, la línia DTR passa a nivell baix. Aquest nivell lògic es transmet a la línia de Reset del ATmega328 mitjançant el condensador C5 de 100nF, tal i com es pot veure a la Figura 6.16., i al tenir el nivell baix (RESET=0) en aquesta línia, el microcontrolador ATmega328 també es reseteja. Com es pot observar a la Figura esmentada, a les connexions del Arduino també hi ha un jumper de soldadura anomenat RESET-EN. Amb aquest jumper es pot prescindir del Auto-Reset soldant els dos terminals entre ells.

Un últim reset que es comenta, encara que si es vol utilitzar al Arduino s'ha d'activar per software, és el Reset per desbordament del gos guardià (Watchdog Reset). El Atmega328 incorpora un oscil·lador intern independent exclusiu pel watchdog que també hi ha integrat. El oscil·lador intern envia polsos periòdica i permanentment a l'entrada del rellotge del comptador (watchdog). Si el comptador arriba a comptar N pulsos, es desborda, la seva sortida s'activa i produeix un reset del microcontrolador. Si es possibilita aquest reset per software, l'objectiu és evitar el desbordament del watchdog. Quan aquest últim s'activa s'inicia el comptador, i l'única manera d'evitar el seu desbordament, i amb això el reset del microcontrolador, és posar a 0 el comptador del watchdog des del programa, i fer-ho en intervals de temps més curts que el temps que es triga en comptar els N polsos (les especificacions tècniques del ATmega328 indiquen en una taula el nombre de cicles del comptador i el temps que triga en realitzar-los, com a exemple, amb una alimentació del microcontrolador de 5 Volts el watchdog trigarà un segon a fer 131072 cicles) . Per aconseguir-ho, quan es realitza el codi de programa s'han de distribuir les instruccions que esborren el watchdog. Si el programa s'executa correctament, el gos guardià mai es desborda ja que abans de fer-ho s'esborra oportunament des del programa. En canvi, si el microcontrolador es perd i el programa deixa de ser executat en la seqüència correcta, el watchdog no s'esborra a temps, es desborda i produeix el reset del microcontrolador, amb la qual cosa és possible reconduir el programa pel camí correcte. L'aplicació d'aquest reset serveix per a garantir la seguretat de funcionament del microcontrolador.

A la Figura 6.16. es poden observar les línies M8TXD i M8RXD que surten des del ATmega16U2 i es connecten als pins PD0 i PD1 del ATmega328. Aquestes línies fan arribar els polsos TTL convertits de USB mitjançant el ATmega16U2, que es connecten al Atmega328 per establir comunicació entre el microcontrolador principal (ATmega328) i l'ordinador. Aquestes línies estan connectades al PD0 i PD1, ja que són els pins de transmissió i recepció que connecten amb la USART del microcontrolador. La USART (transmissió/recepció síncrona/asíncrona universal) permet una comunicació de tipus sèrie, és per això que fa servir dues línies de comunicació, una per a la recepció i l'altra per a la transmissió. Aquest tipus de transmissió, en el cas del Arduino, es fa en mode asíncron, ja que no hi ha cap rellotge que reguli la comunicació entre el ATmega16U2 i el microcontrolador principal. Si es volgués utilitzar comunicació síncrona s'hauria de

connectar un cable del pin PD5 del ATmega16U2 al PD4 del ATmega328, segons les indicacions del full de dades d'aquests dos microcontroladors.

En la comunicació sèrie, la informació en forma de bits viatgen un darrere de l'altre, per aquest motiu la quantitat de línies és mínima.

La USART que incorpora el ATmega328 es pot configurar de diferents maneres. La descripció detallada d'aquest mòdul es troba a l'apartat 20, pàgina 178 del manual de referència del ATmega328P. La configuració que es fa servir per defecte amb l'ordinador és en mode asíncron, amb un rellotge intern i sense habilitar el mode de multiprocessador. La velocitat a la que poden anar els bits, al igual que el format del missatge són paràmetres que es poden configurar tant per part del Arduino com per part de l'ordinador. La comunicació és possible si els dos dispositius tenen la mateixa configuració. Per defecte, aquesta comunicació queda definida per 8 bits de dades, 1 bit de parada i sense paritat. És per això que si no s'indica el contrari aquesta és la configuració que queda definida tant per l'ordinador com pel Arduino.

Tant a la línia M8TXD com a la M8RXD hi ha connectades unes resistències de 1K Ohm RN4A i RN4B respectivament. Aquestes resistències són utilitzades per a acoblar les dues USARTs, la del ATmega16U2 i la del ATmega328, de tal manera que la USART de la primera no es faci malbé si s'utilitzen tant el pin 0 com el pin 1 del ATmega328 per a altres coses, que no siguin comunicar amb la USART del ATmega16U2.

El Arduino també permet, un tipus de comunicació anomenat I2C, mitjançant els pins GND, PC5 i PC4 del ATmega328. I2C és un protocol dissenyat per l'empresa Phillips, la patent del qual ja està expirada i es pot utilitzar amb total llibertat. Aquest protocol s'executa sobre tres cables; un cable de base (GND), un cable amb senyal de rellotge (SCL) que correspon al PC5 del ATmega328, i un cable amb dades (SDA) que correspon al PC4 del ATmega328. Aquest protocol serveix perquè diferents dispositius interactuïn entre ells, de manera que un dispositiu dóna instruccions i els altres compleixen. La comunicació I2C és popular ja que amb dues línies SCL i SDA es poden connectar varis dispositius, els quals cadascun té una direcció diferent, per tant, el dispositiu principal pot donar instruccions només a un dels dispositius de forma independent sense pertorbar als altres. Per a fer servir aquest tipus de comunicació amb Arduino, s'ha d'utilitzar una llibreria de programació disponible a la web de Arduino (www.arduino.cc) anomenada

Wire. Els pins PC5 i PC4 del ATmega328, quan no es fa servir el I2C, normalment es fan servir com a entrades analògiques o entrades i/o sortides digitals.

Els pins del ATmega328 PD3, PD5, PD6, PB1, PB2 i PB3, són pins d'entrada o sortida digitals, però també es poden fer servir com a sortides PWM (modulació per ample de pols). El PWM és una tècnica per a simular una sortida analògica amb una sortida digital. El control digital s'usa per a crear una ona quadrada, un senyal que commuta constantment entre encès i apagat. Aquest patró d'encès-apagat pot simular voltatges des de 0 (sempre apagat) fins a 5 volts (sempre encès) simplement variant la proporció de temps entre encès i apagat. A la durada del temps d'encesa (ON) se l'anomena Ample de Pols (Pulse Width). Per variar el valor analògic es canvia aquest ample de pols amb la IDE de Arduino. La freqüència de la senyal PWM és pròxima a 500Hz, i els valors de sortida han d'estar compresos entre 0 i 255 sent 0 apagat (0 Volts) i 255 encès (5 Volts).

Els ATmega328 té 6 canals connectats al convertidor analògic a digital (A/D), corresponents als pins PC0, PC1, PC2, PC3, PC4 i PC5 de l'esquema de connexions de la Figura 6.16. El convertidor A/D té una resolució de 10 bits, la qual cosa permet que retorni un valor enter entre 0 i 1023 proporcional a la tensió llegida. Encara que la funció principal dels pins analògics serà la de llegir sensors analògics, aquests pins també podran ser usats com a pins d'entrada i sortida digitals.

Tots els pins del port D, port C i port B del ATmega328, poden configurar-se com a entrades o sortides digitals. Els pins del ATmega328 per defecte són d'entrada (INPUT), de manera que no cal configurar-los explícitament com entrades a la IDE. Es diu que els pins configurats com entrades estan en estat d'alta impedància. Una manera d'explicar això és que els terminals d'entrada fan demandes extremadament petites en el circuit que estan mostrejant, es diu que equival a una resistència en sèrie de 100M davant el pin. Els pins configurats com a sortida (OUTPUT) es diu que estan en un estat de baixa impedància. Això significa que poden proporcionar una quantitat substancial de corrent a altres circuits. Els pins del ATmega328 poden proporcionar corrent positiu o corrent negatiu de fins a 40 mA (miliampers) a altres dispositius o circuits.

6.1.2. Mòdul GPS+GPRS/GSM

Aquest mòdul, desenvolupat per l'empresa Cooking Hacks, és el que s'utilitza en el projecte per a realitzar les funcions GSM i GPS gràcies al xip que porta integrat, el SIM908. La idea d'aquest mòdul (Shield) és llegir les coordenades del GPS que porta integrat (longitud i latitud) i enviar-les d'una banda a través d'un SMS i per l'altre mitjançant una petició HTTP a un servidor web. Aquest Shield és totalment compatible amb Arduino i aquesta va ser una de les raons per la qual es va decidir utilitzar-lo en el projecte. S'ha de tenir en compte que aquest mòdul també té la possibilitat de ser utilitzat amb una altra placa de desenvolupament, la Raspberry Pi, és per això que el mòdul incorpora un Jumper que selecciona quina de les dues plaques s'està utilitzant.

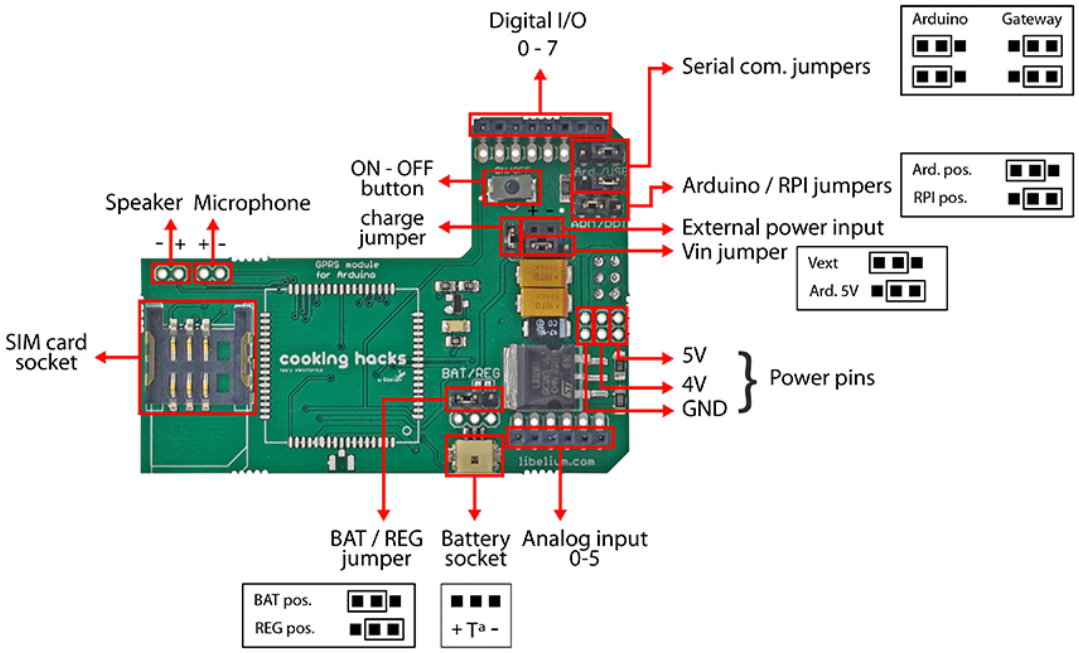

Fig. 6.17. Diagrama del mòdul GPRS+GSM

A la Figura de sobre s'hi pot veure un diagrama del Shield. El pont de selecció de la placa que s'acaba de comentar és el anomenat Arduino/RPI jumpers. Els connectors Digital I/O són extensions de les entrades i sortides Arduino, de les quals només es fan servir la 0 i la 1 per a la transmissió i recepció de dades i la 2 per a la funció de power del mòdul. Els Serial com. Jumpers són de selecció. Quan es carrega un programa al Arduino han d'estar en Gateway per a no interferir en les línies que connecten el Arduino amb el PC, que són les mateixes que connecten Arduino amb el xip SIM908. Un cop el xip està en funcionament,

En aquest capítol s'utilitzen les referències [8]-[12]

han d'estar en mode Arduino. Aquest mòdul també incorpora un botó de ON-OFF, com es pot veure a la Figura de sobre. El External power input, serveix per si s'hi vol connectar una entrada d'alimentació externa. És per això que també hi ha un pont de selecció anomenat Vin jumper que selecciona si l'alimentació del mòdul arriba des del Arduino o l'alimentació exterior. Per alimentar al mòdul també s'hi pot connectar directament una bateria mitjançant el battery socket. El BAT/REG jumper serveix per a seleccionar si l'alimentació arriba des de la bateria o el Arduino/alimentació externa. Si es l'alimentació arriba des de la bateria s'ha de connectar el charge jumper; en cas contrari s'ha de desconnectar. Els connectors Analog input, són extensions de les entrades analògiques del Arduino. Com es pot veure, la placa també incorpora unes sortides d'alimentació (power pins) i unes sortides per a connectar-hi un altaveu i un micròfon (Speaker Microphone). Per a introduir la targeta SIM amb la que es vol treballar, el mòdul també incorpora un sòcol, anomenat SIM card socket.

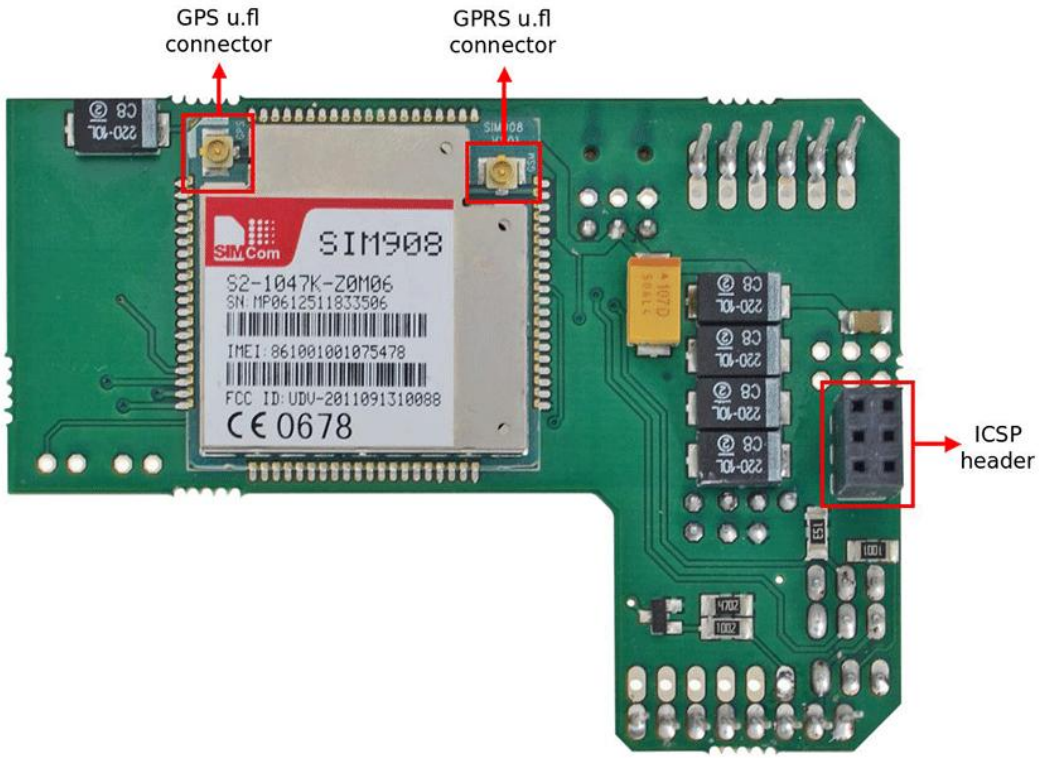

Fig. 6.18. Part inferior del mòdul GPRS+GPS

A la part inferior del mòdul, s'hi troben dos connectors, on s'hi han de posar la antena GPS i la antena GPRS, per a poder garantir un millor funcionament. El ICSP és simplement

l'extensió ICSP del microcontrolador ATmega328 del Arduino, i en el cas del mòdul, s'utilitza per a fer-hi arribar l'alimentació de 5 Volts i el GND. A la Figura de sobre també s'hi pot apreciar el xip principal del mòdul SIM908 que fa les funcions GPS i GPRS. Seguidament es mostra una imatge amb les antenes GPS i GPRS connectades correctament.

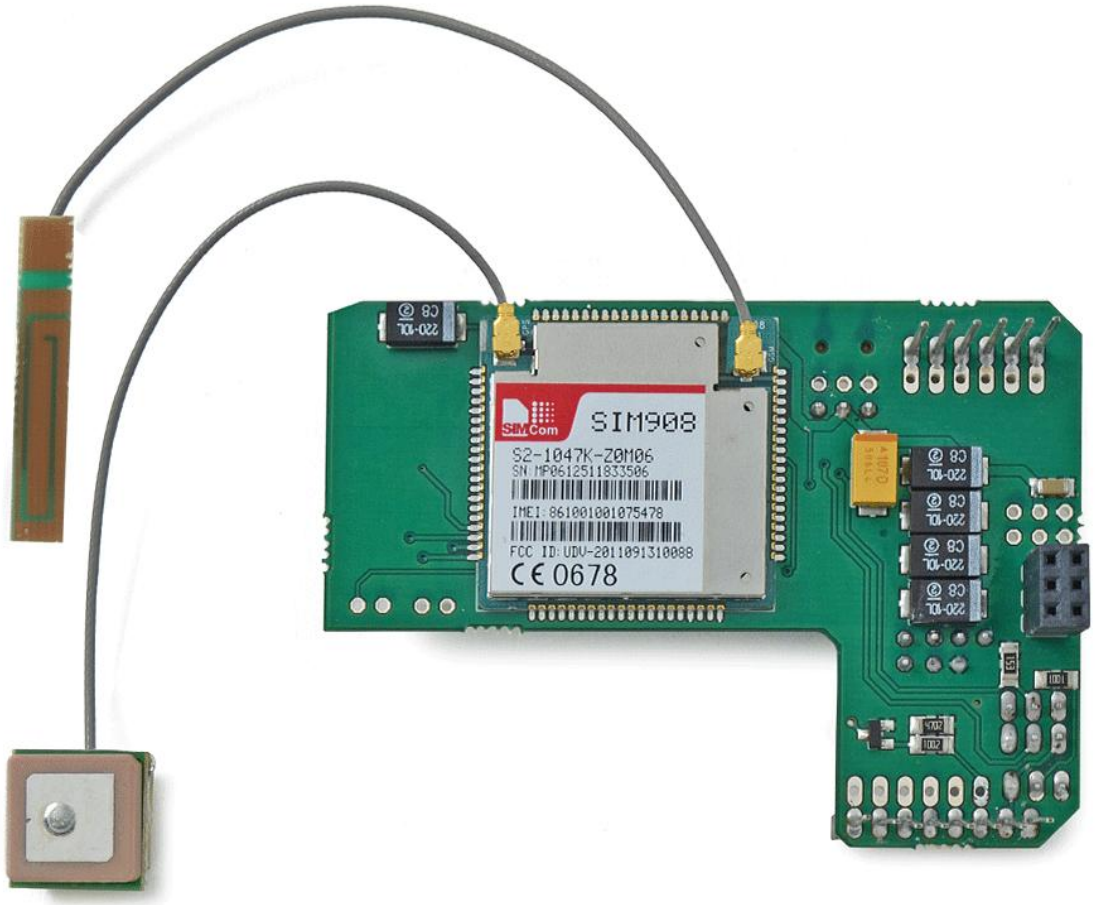

Fig. 6.19. Part inferior del mòdul GPRS+GSM amb les antenes connectades.

El xip SIM908 funciona amb instruccions AT. Aquest tipus d'instruccions s'expliquen a l'apartat de software del treball. Però com a apunts importants, s'ha de comentar que si es vol interactuar amb el mòdul, es pot utilitzar el monitor sèrie de la IDE de Arduino, per a enviar instruccions AT al xip SIM908. El monitor ha d'estar en mode CR (Carriage Return i LF (Line Feed) i amb una velocitat de transmissió (baud rate) de 115200bps (bits per segon) segons indica el fabricant del mòdul. S'ha de tenir en compte que per a fer funcionar el mòdul, un cop totes les connexions estan establertes, s'ha de polsar el botó de

En aquest capítol s'utilitzen les referències [8]-[12]

ON. La placa incorpora un LED que ajudar a identificar el comportament que té el mòdul. Els diferents significats d'aquest LED es mostren a la taula de sota.

Estat del LED	**Comportament SIM908**
Apagat	SIM908 no funciona
64ms Apagat/800ms encès	SIM908 no registrat a la xarxa GSM
64ms Apagat/3000ms encès	SIM908 registrat a la xarxa GSM

Taula 6.2. Comportament SIM908 en funció de l'estat del LED del mòdul.

Com ja s'ha comentat anteriorment, hi ha tres maneres d'alimentar el mòdul.

La primera és alimentar-lo directament amb els 5 Volts del Arduino, mitjançant el ICSP. Per a fer-ho, el pont BAT/REG ha d'estar a la posició REG, el pont Vin a la posició Ard 5V, i finalment s'ha de desconnectar el pont charge. Alguns ports USB dels ordinadors no són capaços de proporcionar el corrent necessari per a fer funcionar el mòdul. Si aquest s'apaga quan intenta connectar-se a la xarxa GSM, pot ser útil utilitzar l'entrada d'alimentació externa (12V-2A, segons el fabricant del mòdul) del Arduino.

La segona manera d'alimentar el mòdul és utilitzant l'entrada externa que incopora. En aquest cas, el pont BAT/REG també ha d'estar a la posició REG, el pont Vin a la posició Vext, i finalment s'ha de desconnectar el pont charge. La font d'alimentació externa pot ser capaç de subministrar pics de corrent de 2A. El fabricant recomana una alimentació externa de 12V-2A. S'ha de tenir en compte que aquesta alimentació externa només serveix pel mòdul i no pot alimentar el Arduino.

L'última manera d'alimentar el mòdul és mitjançant una bateria de Liti. Per fer-ho, el pont BAT/REG ha d'estar a la posició BAT, s'ha de desconnectar el pont Vin i finalment, connectar el pont charge. S'ha de tenir en compte que si s'utilitza la bateria per a alimentar el mòdul, aquesta no pot alimentar el Arduino. La bateria del mòdul es carrega utilitzant l'alimentació del Arduino, mitjançant el ICSP.

Seguidament es mostra una taula resum amb les posicions de connexió dels ponts segons des d'on arriba l'alimentació del mòdul.

Font d'alimentació	**Pont BAT/REG**	**Pont Vin**	**Pont charge**
Des del Arduino	REG	Ard	Desconnectar
Alimentació externa	REG	Vext	Desconnectar
Bateria	BAT	Desconnectar	Connectar

Taula 6.3. Connexió dels ponts del mòdul segons la font d'alimentació d'aquest.

El mòdul està dissenyat de manera que tingui una analogia amb el Arduino. Les dimensions d'aquest primer són molt similars a les de la placa de desenvolupament. Com s'ha pogut apreciar, les entrades i sortides del Arduino tenen unes extensions al mòdul; perquè d'aquesta manera no quedin desaprofitades i no es puguin utilitzar, ja que si no es tinguessin aquestes extensions; moltes entrades i sortides quedarien amb la impossibilitades sota el mòdul. Com es pot veure a la Figura de sota.

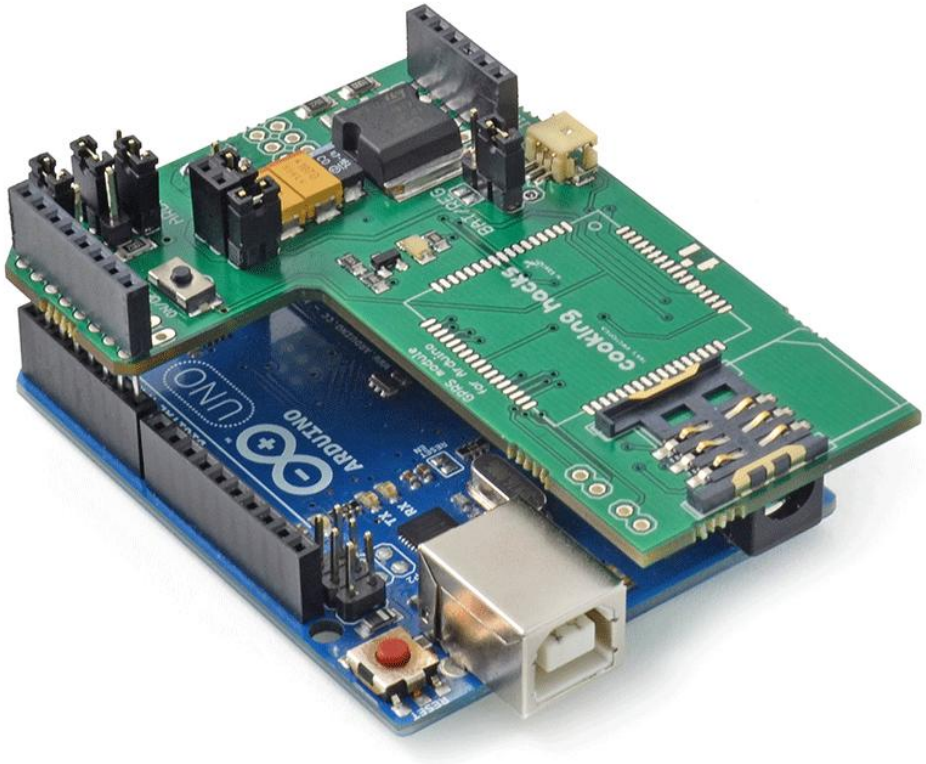

Fig. 6.20. Arduino amb el mòdul GPRS+GPS degudament muntat.

En aquesta Figura de sobre, també es pot apreciar com el mòdul està dissenyat de manera que no cobreix les entrades/sortides del Arduino 8-13; perquè s'hi puguin connectar altres components i no quedin inutilitzades. Aquest disseny també permet que el botó de Reset manual del Arduino quedi fàcilment accessible per si s'ha d'utilitzar.

Seguidament s'explica l'esquema elèctric del mòdul GPRS+GPS, del fabricant Cooking Hacks.

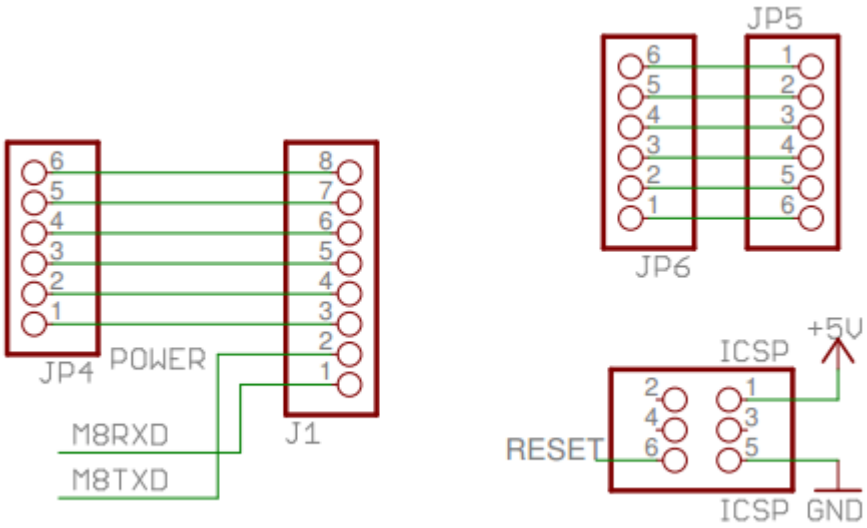

Fig. 6.21. Connectors entre Arduino i mòdul GPRS+GPS

En aquesta primera Figura que es veu a sobre, s'hi poden relacionar els connectors del Arduino que tenen extensió al mòdul. El JP6 correspon a les entrades analògiques del Arduino, que com es pot apreciar, connecten amb el JP5 que són les extensions que incorpora el perifèric d'aquestes entrades analògiques. El ICSP és una extensió del ICSP que incorpora el Arduino, com es pot apreciar, si l'alimentació arriba des de la placa de desenvolupament, ho fa a través d'aquest connector. El JP4 representen una part de les entrades/sortides digitals que incorpora el Arduino i que també tenen extensió en el mòdul. Com es pot comprovar les línies de transmissió i recepció de dades que comuniquen el mòdul amb el Arduino venen per les extensions 1 i 2 (Pins 0 i 1 del Arduino); i la línia de POWER, correspon al connector 3 (pin 2 del Arduino).

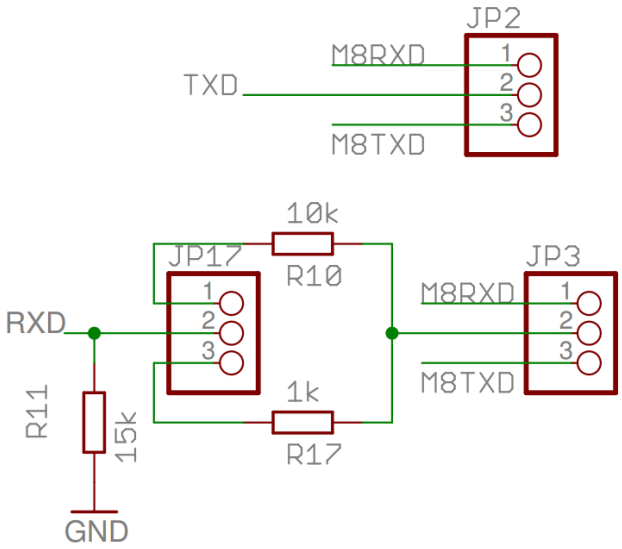

Fig. 6.22. Línies de transmissió i recepció del mòdul GPRS+GPS

Els Jumpers JP2 i JP3 són els ponts Serial Com. S'ha de tenir en compte que la línia TXD és la línia de transmissió del SIM908; i perquè interactuï amb el ATmega328 ha d'estar connectada a la línia de recepció del Arduino M8RXD i el mateix passa amb la línia RXD i M8TXD. Quan es vol carregar un programa al Arduino; per no interferir en les línies de transmissió i recepció; les connexions han d'estar posades a l'inversa del que s'ha comentat. El JP17 és un pont de selecció per si es fa servir el mòdul amb una placa de desenvolupament Raspberry Pi o una Arduino (Arduino/RPI jumper), en el cas d'utilitzar Arduino s'ha d'utlitzar la resistència R10 10K Ohms, per tant el pont estarà connectat entre 1 i 2. Les resistències R10 i R11 serveixen per a formar un divisor de tensió a la línia RXD, això és degut a que la transmissió i recepció de dades en el SIM908 es fan amb nivells CMOS (0-3,3V), i en canvi les comunicacions del ATmega328 es fan amb nivells TTL (0-5V), per això s'han d'adaptar els nivells; i en aquest cas, s'utilitza un divisor de tensió. En el cas de la línia de transmissió no cal adaptar els nivells CMOS que arriben al Arduino ja que els TTL detecten un nivell alt a partir dels 2V; i com que el CMOS dóna un nivell alt de 3,3, no cal incorporar cap component.

En aquest capítol s'utilitzen les referències [8]-[12]

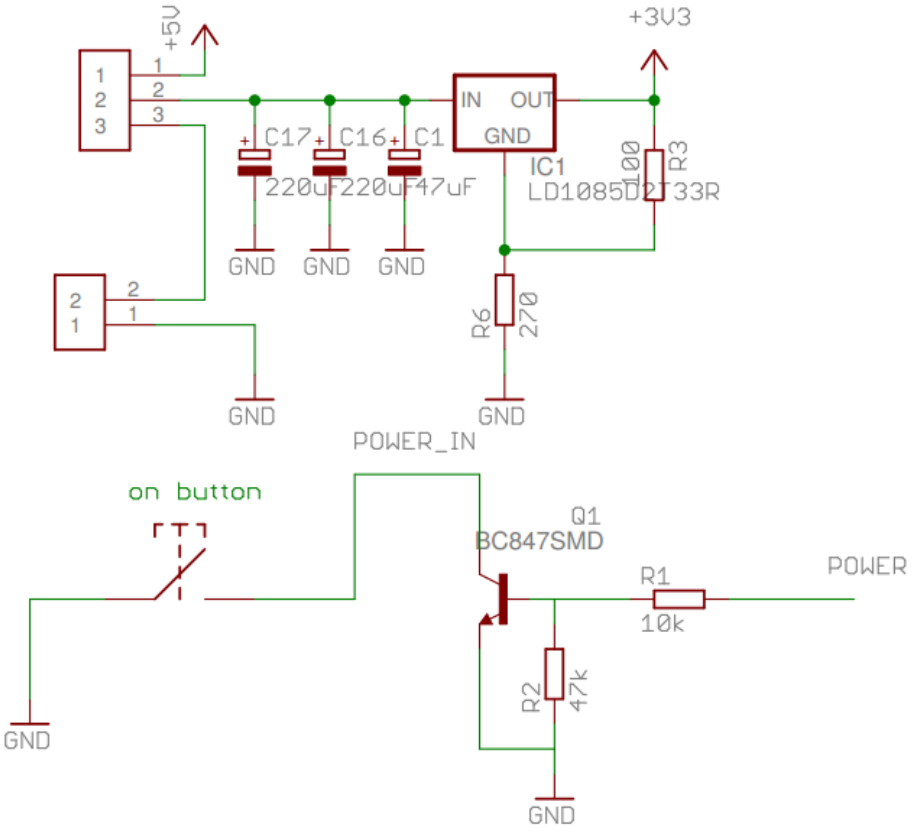

Fig. 6.23. Alimentació del mòdul des de Arduino/Vin

En aquesta Figura 6.23. es pot apreciar el Vin Jumper, que és el pont que en el seu interior té els números 1, 2 i 3 i que serveix per a seleccionar si l'alimentació arriba des del Arduino o externament. El connector que hi ha sota del Vin Jumper, que té els números 1 i 2 en el seu interior és l'entrada d'alimentació externa que incorpora el mòdul. La línia que surt del Vin Jumper va a parar a un regulador de tensió de 3,3V (IC1) que és la tensió amb la que opera el SIM908. Els condensadors i resistències que incorpora el circuit a banda i banda del regulador són recomanades pel fabricant d'aquest últim. A la part de baix de la Figura 6.23. s'hi troba el circuit, recomanat pel fabricant del xip SIM908, per incorporar un botó de Power. Aquest pin de Power ja està connectat internament a 3 Volts, per tant, aquest circuit de Power extern és prescindible. De totes maneres, segons el fabricant, si es fa servir aquest circuit de Power extern; és bo saber que per engegar el SIM908, el botó extern s'ha de prémer durant un temps mínim d'un segon.

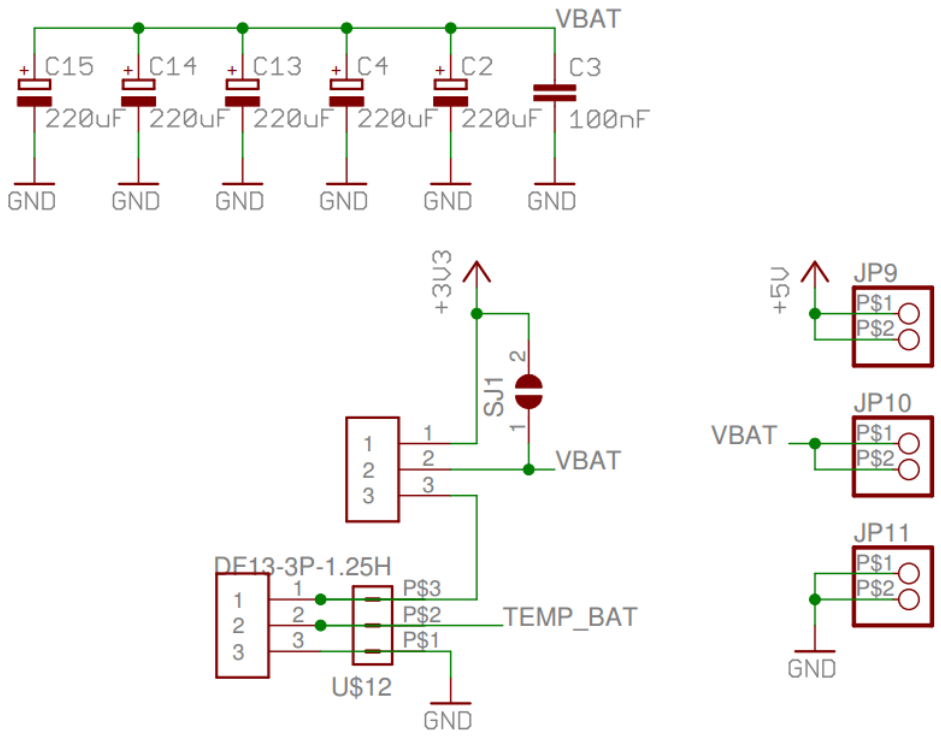

Fig. 6.24. Alimentació del mòdul des de la bateria.

Segons es pot veure a la Figura de sobre; el DF13-3P-1.25H és el connector per la bateria. La línia TEMP_BAT és una línia que va fins a una entrada del xip SIM908 i serveix per a mesurar el nivell de la bateria. El pont que es troba a sobre del connector d'entrada de la bateria, és el BAT/REG Jumper, i serveix per a seleccionar si es vol que l'alimentació arribi des de la bateria o des del Arduino/Vin, o sigui, des del regulador de tensió de 3,3V. El component SJ1 és un pont de soldadura que serveix per si es vol seleccionar que permanentment l'alimentació arribi des del regulador.

Els connectors JP9, JP10 i JP11, són les sortides d'alimentació que incorpora el mòdul. A la part superior de la figura, es pot veure que la línia VBAT incorpora una bateria de condensadors. Aquesta part del circuit és utilitzada sobretot per quan l'alimentació arriba des de una bateria, es fa servir perquè quan la bateria es va descarregant poden haver-hi fluctuacions en el corrent, i aquests condensadors ajuden a estabilitzar la senyal que arriba fins el SIM908.

En aquest capítol s'utilitzen les referències [8]-[12]

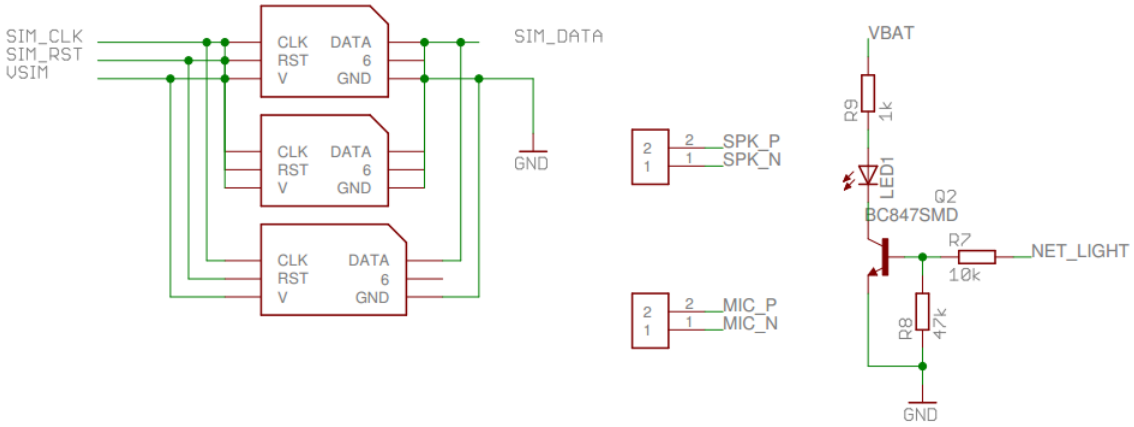

Fig. 6.25. LED de funcionament, targeta SIM, micròfon i altaveu.

A la figura de dalt, s'hi poden observar els connectors que incorpora el mòdul per a connectar la SIM, a l'esquerra de la imatge. Al centre d'aquesta hi ha els connectors per a incorporar un micròfon (MIC_P i MIC_N) i un altaveu (SPK_P i SPK_N), això és degut a que el SIM908 incorpora l'opció de programar-lo per a realitzar trucades. A la part dreta de la figura, hi ha el circuit, recomanat pel fabricant del SIM908, per a incorporar un LED de funcionament, l'alimentació del qual arriba per la línia VBAT i la línia NET_LIGHT prové directament del SIM908 i serveix per a donar al LED les diferents instruccions perquè tingui un estat o un altre segons el comportament que esdevé en el SIM908 (veure Taula 6.1.).

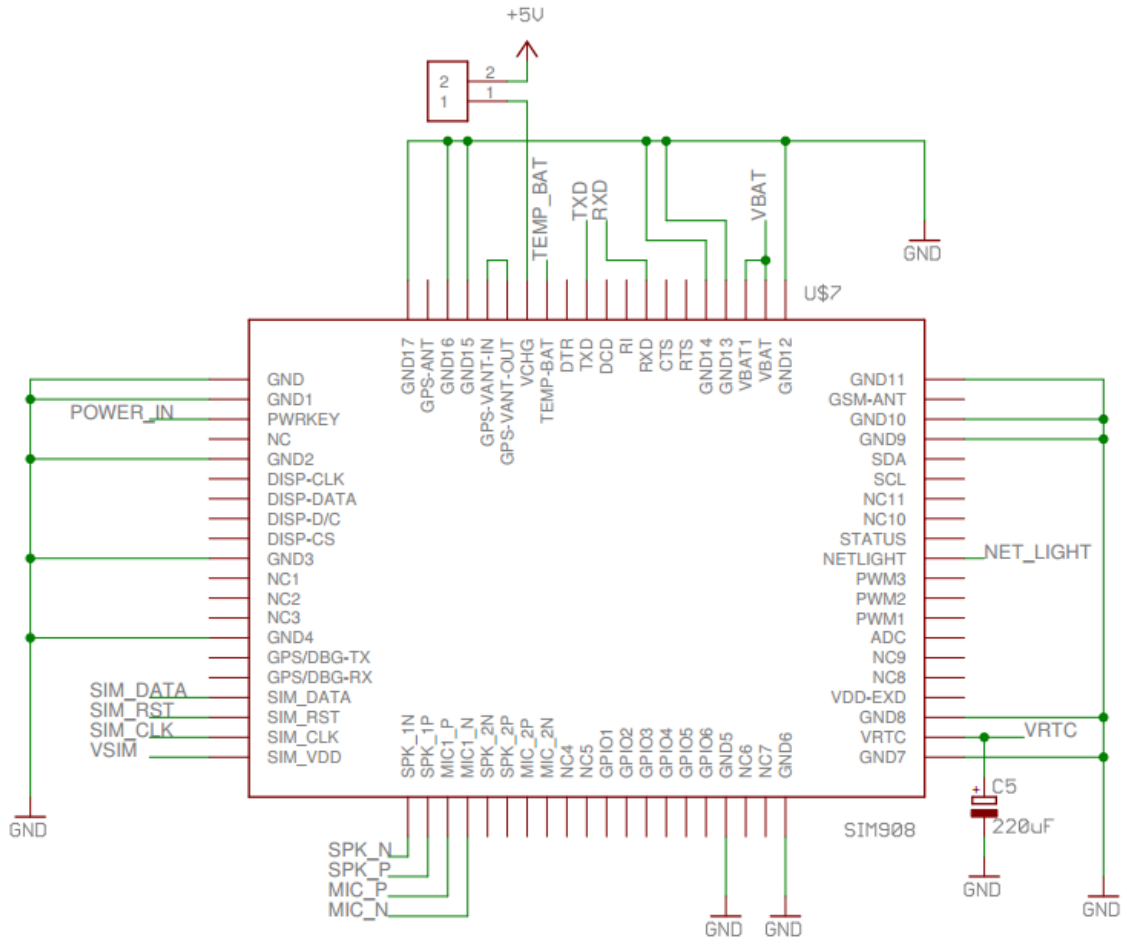

Fig. 6.26. Esquema de connexions del SIM908

A la figura de dalt, es poden observar les connexions del xip SIM908 que incorpora el mòdul GPRS+GSM. Els pins SIM_DATA, SIM_RST, SIM_CLK i SIM_VDD (aquest últim és l'alimentació que arriba a la targeta SIM mitjançant VSIM) són els que van connectats al sòcol on es col·loca la targeta SIM per operar amb el xip. Com ja s'ha comentat anteriorment els MIC_P i MIC_N serveixen pel micròfon i els SPK_P i SPK_N per l'altaveu. El pin PWRKEY va connectat a POWER_IN que és la línia d'entrada del botó d'engegada del xip extern, que s'ha de recordar que és prescindible ja que també està connectat internament. El pin NET_LIGHT serveix per a donar un estat al LED, i depenent de l'estat que tingui aquest, es pot saber quin comportament està tenint el SIM908. El pin VRTC va connectat a un condensador. El SIM908 incorpora un rellotge a temps real RTC (Real Time Clock) el qual funciona encara que el xip no tingui alimentació gràcies a aquest condensador. També s'hi pot connectar una bateria. Els pins TXD i RXD són els pins de transmissió i recepció de dades del SIM908, que s'ha de recordar que treballen amb polsos

En aquest capítol s'utilitzen les referències [8]-[12]

CMOS de 3,3 Volts. Els pins VBAT són els d'entrada d'alimentació, tant si l'alimentació arriba des de la bateria, Arduino o externament, aquesta arriba per la línia VBAT i entra pel pin amb aquest mateix nom. El TEMP_BAT és un pin que serveix per a saber la temperatura de la bateria. Finalment el pin VCHG és un pin d'alimentació externa que serveix per a quan l'alimentació de VBAT arriba des d'una bateria, poder carregar aquesta mitjançant el VCHG. Ja que el SIM908 incorpora un circuit intern de carrega de bateria. El pont que es troba sobre del pin VCHG de la Figura 6.26. és l'anomenat charge jumper. Quan l'alimentació arriba des d'una bateria es connecta aquest pont, i d'aquesta manera quan el nivell de bateria sigui baix, es pot introduir una altra font d'alimentació per a poder carregar-la a través del VCHG, que mitjançant el circuit intern del xip SIM908 carregarà la bateria per la línia VBAT. S'ha de tenir en compte que quan la bateria s'està carregant a través de VCHG, el SIM908 es posa en estat de càrrega amb la conseqüència que les seves funcions són limitades. Per exemple, quan s'està carregant la bateria, el xip es desconnecta de la xarxa GSM.

6.2. Llenguatge de programació

Com s'ha explicat anteriorment, el ATmega328, que és el microcontrolador principal de la placa de desenvolupament Arduino, té instal·lat un gestor de càrrega (bootloader), d'aquesta manera es pot programar mitjançant l'entorn de desenvolupament (IDE), dissenyat exclusivament per Arduino. El llenguatge de programació que es fa servir a la IDE del Arduino està basat en Wiring, el qual aquest és molt semblant al llenguatge C; per tant, es podria dir que realment Arduino es programa amb llenguatge C.

Una part molt important del projecte implica la programació, mitjançant el llenguatge Arduino, del microcontrolador ATmega328 perquè aquest pugui interactuar amb el xip SIM908. És per això que seguidament s'expliquen algunes de les funcions del llenguatge de programació Arduino.

6.2.1. Estructura bàsica d'un programa

Qualsevol programa que es realitza utilitzant la IDE de Arduino presenta tres parts.

La primera part és la de declaració de variables. Aquestes són valors que per defecte es guarden a la memòria SRAM del Arduino. També es poden guardar a la memòria EEPROM o la memòria FLASH d'aquest, utilitzant biblioteques especials per aquestes funcions. A l'hora de declarar una variable no cal assignar-li un valor inicial, es pot declarar i posteriorment durant el transcurs del programa donar-li un valor.

```
int x;
...
x=4;
```

Fig. 6.27. Declaració d'una variable

Si es declara una variable al començament del programa, aquesta es podrà utilitzar en qualsevol moment (dins de qualsevol funció o bloc de programa), però si es declara una variable dins d'una funció, només es podrà utilitzar en aquesta funció. S'ha de tenir en compte que = serveix per assignar, i no s'ha de confondre amb == que serveix per a comparar. Seguidament es mostren els diferents tipus de variables que hi poden haver.

- int: guarda un número enter entre -32769 i 32767 (2 bytes).

- unsigned int: guarda un número natural entre 0 i 65536 (2 bytes).
- char: guarda un caràcter, com pot ser una lletra o un símbol. També es pot utilitzar per un número enter entre -128 i 127 (1 byte).
- boolean: guarda un valor amb dues possibilitats: 0 o 1, o veritat o fals.
- byte: guarda un número natural entre 0 i 255 (1 byte).
- long: guarda un número enter entre -2147483648 i 2147483647 (4 bytes).
- unsigned long: guarda un número enter entre 0 i 4294967295 (4 bytes).
- float: guarda un número decimal amb un rang entre $-3.4028235 \cdot 10^{38}$ i $3.4028235 \cdot 10^{38}$ (4 bytes).
- double: En llenguatge C, es guardaria un número decimal amb molta precisió, amb un valor màxim de $1.7976931348623157 \cdot 10^{308}$. Però, en Arduino és el mateix que un float (4 bytes).

Si s'escriu la paraula const davant d'una variable, s'especifica que aquesta no podrà canviar-se durant el programa i serà un valor constant. Una variable que ja s'ha declarat, si davant no s'hi escriu la paraula const, pot canviar de valor i també pot canviar de tipus de variable durant el programa. En el cas que es declarin diverses variables del mateix tipus es pot escurçar el codi escrivint només una vegada el tipus de variable, i seguidament, el nom de les variables separant-les amb comes.

La segona part del programa és la que s'encarrega de configurar el Arduino. Va encapçalada per la funció void setup (), i aquí s'especifiquen quins pins són utilitzats com a entrades i quins com a sortides.

```
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
pinMode(8,INPUT);
```

Fig. 6.28. Configuració de les entrades i sortides dels pins del ATmega328

A la figura de dalt es pot observar que el pin 2 del Arduino es configura com a sortida digital. El pin 3 del Arduino com a sortida, com que aquest pin permet PWM, la sortida tant pot ser analògica com digital. Per últim el pin 8 es configura com a entrada digital. Les entrades analògiques no cal configurar-les al setup ja que aquests pins (A0, A1, A2, A3,

A4 i A5), només poden ser entrades i el programa té configurats els pins com a entrades per defecte.

La tercera part del programa és on s'hi especifiquen les instruccions que regiran el comportament del Arduino. Aquesta tercera part va encapçalada per la funció void loop (). Aquí s'escriuen totes les instruccions, ordres o funcions necessàries perquè el Arduino funcioni tal com es vulgui. Realment aquest bloc constitueix un bucle infinit, ja que el Arduino, mentre estigui alimentat, funcionarà fent el programa loop una i altra vegada.

6.2.2. Funcions de temps

El llenguatge Arduino també incorpora diferents funcions de temps, com és el cas de la funció delay(). Quan el programa es troba aquesta funció s'espera durant el temps indicat en milisegons que s'escrigui dins els parèntesi de la funció. S'ha de considerar que els valors que s'introdueixen a la funció delay, el Arduino els interpreta com a enters i si es volen fer pauses molt llargues (per exemple d'un minut) es pot sobrepassar el límit. Si es considera delay (60*1000); el programa no farà una pausa d'un minut ja que el resultat supera el límit de 32767 de la variable int. És possible que el Arduino interpreti un número com a long col·locant darrere el valor la lletra L, d'aquesta manera si s'escriu delay(60*1000L); el programa si que farà una pausa d'un minut.

Hi ha altres funcions de temps com la milis (); Per exemple al escriure x = milis (); el que fa el programa és assignar a x el número de milisegons que han passat des de que la placa Arduino ha tingut un reset. És una variable del tipus unsigned long i es produeix un desbordament als 50 dies, sinó hi ha cap reset del ATmega328. Aquesta funció de temps és utilitzada perquè el programa pugui fer una altra tasca mentre espera que es dugui a terme el retard cosa que amb la funció delay no passa. Per exemple a delay(1000); tot el programa s'espera un segon, en canvi utilitzant la funció milis, només s'espera un segon la tasca que es designi, i el programa pot continuar fent una altra tasca.

Exemple d'un programa bàsic.

Amb la IDE del Arduino es pot indicar que un pin determinat del ATmega328 estigui a nivell alt o nivell baix (0 o 1 lògic) que seran 5 Volts i 0 Volts respectivament.

```
digitalWrite(12,LOW);          digitalWrite(12,HIGH);
```

Fig. 6.29. Codi per a posar un pin del ATmega328 a nivell alt o nivell baix

Com es pot apreciar a la figura de dalt, a l'esquerra es posa el pin 12 del Arduino a nivell baix (0V) i a la dreta a nivell alt (5V). Per acabar d'entendre tot el que s'ha explicat fins ara i veure-ho d'una manera més visual, seguidament es mostra una imatge d'un programa molt simple que es realitza amb el Arduino, el qual es tracte de poder encendre i apagar un LED de manera continua.

```
int pinLed=3;
int t=1000;

void setup() {
  pinMode(pinLed,OUTPUT);

}

void loop() {
  digitalWrite(pinLed,HIGH);
  delay(t);
  digitalWrite(pinLed,LOW);
  delay(t);
}
```

Fig. 6.30. Programa bàsic realitzat amb codi Arduino

Com es pot veure a la Figura 6.30. el programa té les tres parts ben diferenciades. Primer de tot es declaren les variables, en segon terme es configura el Arduino de manera que s'estableix el pin 3 com a sortida. I a la tercera part s'hi indiquen les instruccions. En aquest cas les instruccions són donar un nivell alt (5V) al pin 3 del Arduino, esperar un segon, donar un nivell baix (0V) al pin 3 i esperar un segon. Quan acaba el loop el programa torna a començar-lo i es va executar una i altra vegada. D'aquesta manera si es connecta un LED al pin 3 del Arduino, es pot veure com aquest s'encén i s'apaga amb intervals d'un segon de manera indefinida. Aquest programa se sol utilitzar per a provar que el Arduino funciona correctament.

6.2.3. Comunicació Arduino-PC

En moltes ocasions és útil poder visualitzar a través del ordinador els valors de lectura en els pins d'entrada i sortida del Arduino. Així mateix, també pot ser necessari enviar informació a la placa de desenvolupament des del teclat del PC. Per a posar en contacte ambdós aparells, primer s'ha de configurar en el void setup () que s'establirà connexió utilitzant l'ordre Serial.begin (). Dins dels parèntesis s'especifica el valor en Bauds (1 Baud = 1 bit/segon). Un cop fet això; dins el void loop () es poden utilitzar diferents funcions com Serial.print (val); la qual imprimeix el valor de la variable val, Serial.println (val) la qual imprimeix el valor de la variable val i insereix un salt de línia o Serial.print ("hola Arduino"); la qual imprimeix el text hola Arduino. Per a visualitzar les dades per pantalla, s'ha d'obrir la pantalla d'impressió de la IDE de Arduino anomenada Serial Monitor.

6.2.4. Estructures

El Arduino permet programar diferents estructures com és el cas de la estructura condicional que es programa d'aquesta manera if () {...}. Dins els parèntesi s'escriu una condició, i en cas de complir-se, s'executen les instruccions que hi hagi dins les claus. Una variant d'aquesta estructura és la formada per if (condició) {...} else {...}, la qual permet que el programa agafi un dels dos camins; si es compleix la condició s'executaran les instruccions que s'englobin dins les claus del if, i si no es compleix s'executaran les del else.

Una altra estructura que s'utilitza en la programació Arduino es la while que s'especifica així while (condició) {...}. Mentre es compleix la condició expressada entre parèntesi, el Arduino realitzarà les instruccions que hi hagi entre claus. En cas de no complir-se aquesta condició, el programa es saltarà aquesta estructura. Una estructura molt semblant a la while és l'estructura for (inici; condició; increment) {...}. Una última estructura també semblant a les dues anteriors és la do la qual s'especifica do {...} while (condició); aquesta estructura es diferencia del while en que almenys s'executen un cop les instruccions entre claus i després les instruccions es seguiran repetint mentre es compleixi la condició especificada. Hi ha dues ordres que es poden incloure en aquestes estructures esmentades while, for i do, que són l'ordre break i l'ordre continue. L'ordre break; serveix per a trencar

el bucle i que el programa continuï amb el codi que hi ha darrere del bloc. En canvi l'ordre continue el que fa és saltar la resta d'instruccions que quedin per executar de l'estructura i començar-la novament des del principi.

6.2.5. Arrays

El llenguatge també permet guardar una llista de variables (array), accedint a cadascuna d'elles a través del seu índex, començant pel zero. Per exemple:

int dades [ ] = {4,7,9,12,18}

Aquesta instrucció fa referència a dades [0] = 4; dades [1] = 7; ... dades [4] = 18; que com es pot observar en aquest cas, no s'ha hagut d'especificar el tamany del array perquè s'ha inclòs el valor de totes les variables, però si no es declaren els valors inicials s'ha d'especificar el tamany del array; int dades [5];. També existeix una altra manera de cridar els diferents elements del array, aquesta és a través d'un punter. Seguint l'exemple anterior, el primer element és *dades, i per a cridar als següents només s'han de sumar les seves posicions relatives, o sigui que, *dades+1 serà dades [1], *dades+2 serà dades [2], i així successivament.

El llenguatge de programació també permet guardar una cadena de caràcters (strings), i s'utilitza per a guardar textos. Quan es guarda només un caràcter s'usa el tipus d'instrucció char lletra = 'a'. En canvi si es vol emmagatzemar un text es fa amb la inicialització char text [ ] = "Hola Tecnocampus"; Els 17 caràcters que componen el text (16 + 1 que es fa servir per a indicar la fi de cadena, que encara que no es vegi hi és amb la nomenclatura '\0') es guarden com a elements separats al array. S'ha de tenir en compte que per a guardar cadenes es fan servir cometes dobles (" ") i per a guardar caràcters individuals cometes simple (' ').

6.2.6. Definició de funcions

Segons les característiques del programa que es vulgui realitzar, pot ser útil, i en el cas del projecte que es realitza ho és, utilitzar funcions que es poden definir. Els llenguatges de programació incorporen aquesta possibilitat, i Arduino com a tal, no és una excepció.

Si de la funció que es vol definir no se n'espera cap valor de retorn, i es limita a realitzar una sèrie d'ordres que depenen (o no) de certes variables, llavors la funció (que és exactament una subrutina) es defineix de la manera void funciona (a,b) {...}, on a i b són variables que la funció hauria d'utilitzar (per descomptat, hi haurà funcions que no necessitin cap variable). Si aquestes variables no estaven declarades anteriorment, s'hauran de declarar en aquest moment: void funciona (int a, int b) {...}.

Si la funció que es definirà es vol que retorni un valor, s'ha de declarar com si es tractés d'una variable, especificant de quin tipus de variable és el valor (int, long, float...). Perquè la funció agafi un valor, s'ha d'utilitzar la funció return acompanyada de la variable, el valor de la qual es vol assignar a la funció. Seguidament es mostra un exemple bàsic de l'estructura d'una funció:

```
int funcion (a,b) { …
      int val=0;
      … ;
      val= … ;
      return val;
}
```

Fig. 6.31. Estructura bàsica d'una funció

Si en algun moment es desitja interrompre el codi de la funció i sortir-ne (de manera anàloga al break a les estructures), i seguir el programa per la següent línia després de la crida de la funció, s'utilitza la instrucció return; (sense acompanyar-la de cap variable o dada, amb la qual cosa retorna un valor buit).

6.2.7. Ordres AT

Les ordres AT es denominen així per l'abreviatura de la paraula atenció (*attention* en anglès). La telefonia mòbil GSM ha adoptat com estàndard aquest llenguatge per poder comunicar-se amb els seus terminals. D'aquesta manera, tots els telèfons mòbils GSM tenen un joc de comandaments AT específic que serveix d'interfície per a configurar i proporcionar instruccions als terminals, permeten accions com ara fer trucades de dades o de veu, llegir i escriure a l'agenda de contactes i enviar missatges SMS, a més de moltes altres opcions de configuració del terminal. Per a veure les ordres AT específiques del SIM908, que són les que es fan servir en el projecte, es pot veure el PDF del software del dispositiu, o visitar la pàgina web de Cooking Hacks on es detalla l'explicació del mòdul.

En aquest capítol s'utilitzen les referències [8]-[12]

7. Avaluació d'una solució propietària

En aquest capítol s'avalua el disseny d'una solució propietària de hardware, basada en el prototip utilitzat i també s'especifica i s'explica la programació que s'ha realitzat per a complir amb l'objectiu de funcionalitat del dispositiu.

7.1. Hardware de la solució propietària

Un dels objectius del projecte, és dissenyar un model propi basat en el hardware utilitzat en el projecte. El Arduino és una placa de desenvolupament que està adaptada per a poder realitzar molts i diferents projectes, és per això que el seu hardware està pensat perquè pugui utilitzar-se per un munt de coses. En el cas del projecte, el que s'ha volgut aconseguir és un mòdul de vigilància i alarma per a vehicles, per tant, hi ha moltes parts del Arduino que no s'utilitzen i són prescindibles en aquesta aplicació. El mateix passa amb el mòdul GPRS+GPS, aquest està preparat per a ser utilitzat en diverses plaques de desenvolupament i tenir diverses formes d'alimentació que es seleccionen mitjançant els ponts que incorpora. Per aquests motius, s'ha avaluat el Hardware de la solució existent i s'ha decidit re dissenyar-ne algunes parts.

En el disseny teòric, s'ha decidit mantenir el mòdul GPRS+GPS en la seva forma original, i no modificar-lo, canviant així només el disseny de la placa de desenvolupament Arduino. Aquest fet es deu a que el xip SIM908 és de difícil assemblatge ja que els seus punts de soldadura estan molt junts entre ells. D'aquesta manera, dissenyant un nou Arduino mantenint-ne l'analogia ja que les connexions del mòdul GPRS+GPS estan pensades per a connectar-lo amb aquest, s'aconsegueix un disseny propi amb una possibilitat de fabricació més senzilla que es pot utilitzar com a primer prototip.

Pel disseny d'aquest prototip basat en la placa Arduino, s'ha de tenir en compte de no tocar les connexions entre el mòdul i la placa de desenvolupament; per tant el ICSP i els pins 0, 1 i 2, s'han de mantenir en la mateixa posició que en el Arduino Uno R3, ja que són les línies de connexió entre aquestes dues plaques. Degut això s'ha decidit mantenir totes les sortides de pins del microcontrolador ja que així també es dóna la possibilitat a ampliar les possibilitats del dispositiu podent connectar més sensors, leds etc. Com a recordatori, es comenta que el ICSP serveix per a fer arribar l'alimentació del mòdul des de la placa de

desenvolupament, els pins 0 i 1 són els de transmissió i recepció i el pin 2 és el que inicia el SIM908.

En el cas que es decidís fabricar el prototip d'aquest primer disseny, les mides del circuit imprès serien les mateixes que les del Arduino Uno R3, 6,8cm de llargada per 5,3cm d'amplada. D'aquesta manera es manté una analogia que permet connectar el mòdul GPRS+GPS d'una manera simple, igual que quan s'utilitza el Arduino.

Un dels objectius del projecte és que aquest sistema disposi d'autonomia per a funcionar, és per això que en aquest nou disseny, un dels afegits important és incorporar una bateria en el nou circuit. Ja que la idea final seria que el mòdul vagi integrat al cotxe de manera que disposi d'autonomia mitjançant una bateria quan el cotxe està parat, i quan el cotxe engega carregui la bateria del mòdul amb la bateria del cotxe com a alimentació.

El mòdul també ha d'incorporar un circuit de càrrega per aquesta bateria, així com una entrada d'alimentació per a carregar aquesta, així s'emula l'entrada d'alimentació que vindria de la bateria del cotxe en una aplicació final del producte integrat al vehicle i també obrir una altre línia d'aplicació que seria la de que el dispositiu no ha d'anar integrat al vehicle i es pot utilitzar en més aplicacions i no només en vehicles. Òbviament, també s'utilitza un microcontrolador que permeti la instal·lació del gestor de càrrega Arduino per a poder programar amb el seu IDE, ja que el llenguatge de programació utilitzat al projecte és Arduino (que està basat en llenguatge C). També s'ha tingut en compte poder instal·lar els programes fets al microcontrolador. És per això que s'ha incorporat un convertidor de pulsos de USB a TTL (0-5V). Una altra cosa a considerar és poder fer un reset manual del microcontrolador amb un botó, i també garantir el reset d'engegada, per tant s'ha implementat el circuit necessari per a garantir aquests resets.

7.1.1. Comunicacions microcontrolador-PC

Per a les comunicacions entre PC i ATmega328, el Arduino fa servir un microcontrolador ATmega16 programat com a convertidor de polsos. Aquest microcontrolador és de difícil assemblatge ja que té els punts de soldadura molt junts, fins i tot més junts que el xip SIM908. En un primer disseny, s'ha descartat integrar el mòdul GPRS+GPS amb la placa de desenvolupament pel difícil assemblatge del xip, per aquest motiu, també s'ha descartat utilitzar el ATmega16 com a convertidor de polsos. Un altre motiu per a no utilitzar aquest

mètode per a la conversió de polsos, és pel fet de que aquest microcontrolador té instal·lat un programa (*firmware*) per a poder fer aquesta conversió. Encara que aquest programa és software lliure i es pot descarregar de franc mitjançant un enllaç de la pàgina web oficial de Arduino, no és trivial instal·lar-lo. Ja que per fer-ho s'han d'utilitzar els pins MISO, MOSI i el SCK del microcontrolador, per a poder fer la programació amb instruccions AVR que són les que per defecte incorpora aquest. La finalitat d'aquest projecte no és poder interactuar entre el ordinador i la placa de desenvolupament, i enviar programes constantment, ja que en principi, el codi de programa només s'ha d'enviar una vegada i un cop fet, la comunicació directe amb el PC no es necessita més. Per tant, es podria prescindir totalment d'aquesta part de circuit, ja que es podrien programar els microcontroladors a part (per exemple utilitzar un Arduino per carregar el programa al microcontrolador, i quan aquest es carrega treure'l i posar un altre microcontrolador i fer el mateix, i així successivament amb tots els microcontroladors que es vulgui) i després assemblar-los a la placa de circuit imprès. Tot i així, és bo deixar les portes obertes a possibles actualitzacions de software, en el cas que es facin millores, o s'afegeixin funcions al programa, ja que si es prescindeix de la part de comunicació amb el PC, el programa del microcontrolador no es podria modificar d'una forma senzilla ja que estaria integrat a la PCB (placa de circuit imprès). Seria bo els dispositius ja dissenyats i assemblats poguessin disposar d'aquestes actualitzacions, per tant, s'ha decidit no prescindir totalment d'aquesta part.

Mirant possibles solucions dins el mercat actual, s'ha decidit optar per la utilització d'un cable FTDI per a fer aquesta comunicació entre microcontrolador i ordinador. El cable FTDI és un cable que incorpora en el seu interior un xip que fa la transformació de polsos de USB a sèrie (0-5V). És un cable desenvolupat per l'empresa FTDI (*Future Technology Devices International*), d'aquí ve el nom del cable. Aquest cable té l'avantatge que incorpora el circuit per a la transformació de polsos en el seu interior, d'aquesta manera a la placa de desenvolupament només s'hi ha de fer una connexió d'entrada per a connectar aquest cable. A una banda del cable hi ha una entrada USB per a connectar a l'ordinador, i a l'altra hi ha un connector amb sis línies.

Fig. 7.1. Connectors del cable FTDI

A la Figura 7.1. es pot veure l'entrada USB i el connector amb sis línies. De baix a dalt de la imatge, la línia negre correspon al GND, la marró al CTS (que es connecta a terra), la vermella al VCC (5V), la taronja al TX (transmissió de dades) la groga al RX (recepció de dades) i la verda al RTS (es connecta al reset del microcontrolador en el cas que es vulgui fer un Reset per software). Com s'ha comentat, al disseny de la placa de desenvolupament s'hi afegeix una entrada per les 6 línies del connector del cable FTDI. La línia VCC, en un primer moment es va decidir utilitzar-la per a alimentar directament el microcontrolador, però com que en el disseny s'hi inclou una bateria que és la que dóna l'alimentació, la línia VCC del cable FTDI serveix pel circuit de càrrega de la bateria que s'especifica més endavant. Les línies GND i CTS es connecten a terra. La línia RTS (també anomenada DTR) no s'utilitza, ja que no es considera la possibilitat del reset per software, i les línies TX i RX es connecten directament als pins de la USART del microcontrolador, ja que aquestes línies són les dels polsos sèrie (0-5V) que s'han convertit des de USB dins el xip del cable FTDI. Amb aquesta simple entrada pel connector del cable FTDI es resol el tema de poder incorporar la comunicació microcontrolador-ordinador i poder fer actualitzacions de software sense dificultats, i es prescindeix de molts components que s'utilitzen en el cas del Arduino Uno, aconseguint així eliminar parts de circuit, tenir més espai físic a la placa, podent-ne fins i tot disminuir el tamany, i reduir la dificultat d'assemblatge i fabricació.

7.1.2. Alimentació del dispositiu

Com s'ha comentat anteriorment, s'ha considerat el fet d'incorporar una entrada d'alimentació per emular l'alimentació que arriba des de la bateria del cotxe. És per això que al disseny s'hi incorpora una entrada d'alimentació, la qual s'ha tingut en compte que hi pugui haver un voltatge d'entrada de 12V, que és el voltatge amb el que treballen les bateries dels turismes, i així s'emula al màxim el fet que l'entrada d'alimentació externa pel circuit de càrrega de la bateria del dispositiu pugui ser la mateixa bateria del cotxe. El fet de posar una entrada d'alimentació externa, a part d'emular l'alimentació del cotxe, també permet que el dispositiu es pugui carregar per exemple amb un adaptador AC/DC, com en el cas del Arduino. Aquest nou disseny està basat en la placa de desenvolupament Arduino, com ja s'ha esmentat, és per aquest motiu, i tenint en compte que l'entrada d'alimentació del Arduino Uno permet un rang de 7-12 Volts, que s'ha utilitzat el mateix circuit que el que incorpora aquesta entrada d'alimentació (veure Fig. 6.5.) prescindint del condensador C2 que és un condensador de desacoblament, i que no s'utilitza ja que la línia d'aquesta entrada d'alimentació no va connectada directament al microcontrolador, sinó que forma part del circuit de càrrega de la bateria que s'explica més endavant. En un principi, es va pensar en utilitzar el mateix circuit ja que la sortida de voltatge proporcionada pel regulador és de 5 Volts, i d'aquesta manera utilitzar-la per a alimentar el microcontrolador, però al incorporar la bateria per donar autonomia al dispositiu, aquesta línia passa a formar part del circuit de càrrega. De totes maneres s'ha mantingut el regulador ja que el circuit utilitzat per la càrrega té unes especificacions que fan compatibles els 5 Volts de sortida del regulador.

Com es comenta, el nou circuit implementa una bateria. Aquesta acció és una de les incorporacions més importants en el nou disseny, i no ha estat fàcil implementar-la. Les bateries es van descarregant i això provoca fluctuacions de voltatge que poden fer inestable el funcionament de la placa, és per això que el primer tema que es va tractar va ser com estabilitzar el voltatge de sortida de la bateria perquè sigui lineal. Una de les primeres solucions que es va proposar va ser la d'incorporar una línia de condensadors en paral·lel, disposada com en el mòdul GPRS+GPS (veure part superior Fig. 6.24.), de tal manera que els condensadors compensessin les possibles fluctuacions i d'aquesta manera obtenir una senyal lineal. Finalment però, s'ha optat per a fer servir un convertidor CC/CC (corrent continu/corrent continu), ja que d'aquesta manera s'aïlla el senyal de la bateria del que

alimenta el microcontrolador i es garanteix millor l'estabilitat de voltatge a la sortida del convertidor.

7.1.3. Pins del microcontrolador

El microcontrolador que s'utilitza és el ATmega328, per a mantenir l'analogia amb el prototip existent i mantenir les línies de connexió, ja que en aquest nou disseny el mòdul GPRS+GPS es conserva en la seva forma original, per tant la seva integració amb la nova placa de desenvolupament també ho ha de ser, però en un disseny final, es podria utilitzar un microcontrolador diferent amb menys pins, ja que, en el cas del mòdul existent només es fan servir els pins d'alimentació, els pins de la USART i el pin 2 per l'inici del SIM908, quedant lliures i sense utilitat la resta de pins. El número de pins necessaris al disseny final, quedaria marcat per la funcionalitat que es vol obtenir del dispositiu, ja que per exemple si es volgués que el dispositiu faci més coses a part d'obtenir i enviar la posició GPS, com obrir les finestres del cotxe, activar una alarma, activar un LED, etc. Es necessitarien utilitzar més o menys pins del microcontrolador.

7.1.4. Incorporació d'una bateria

En el cas de la bateria, s'ha pensat utilitzar una bateria de liti de 3,7V i que proporcioni al voltant de 2000mAh (miliampers/hora). Amb això, sabent la bateria i el microcontrolador que s'utilitzen, el convertidor CC/CC que s'ha elegit es tracte del MAX1674 (fabricat per l'empresa Maxim Integrated) que permet un voltatge d'entrada amb un rang entre 1,1-5,5V i un voltatge de sortida a elegir de 3,3V o 5V. En el cas del disseny propi, el voltatge d'entrada és el de la bateria 3,7V i pel de sortida es seleccionen 5V que serveixen per alimentar el microcontrolador ATmega328. Encara que aquest podria treballar amb el voltatge de 3,3V no es selecciona pel fet que el Mòdul GPRS+GSM incorpora una regulador de tensió de 3,3V, i si l'alimentació que arriba en aquest regulador no és superior, aquest no funcionarà correctament, ja que sempre s'ha de tenir en compte una caiguda de tensió al regulador anomenada drop-out. La sortida a 5V del MAX1674 proporciona un valor típic de 285mA. El ATmega328 té un consum de 200mA, i el consum en el mòdul GPRS+GSM varia depenent de les funcions que hi hagin actives. L'activació i captació de dades GPS requereix 77mA. La connexió GSM per a poder rebre i enviar SMS requereix 21mA. I quan hi ha transmissió de dades GPRS el SIM908

requereix un mínim de 80mA. El fabricant comenta que durant les transmissions, el SIM908 pot requerir pics de 2A. Això pot ser un problema, tenint en compte que l'amperatge de sortida del convertidor CC/CC és de 285mA, podria passar que el mòdul s'apagués. Per a solucionar-ho, es podria connectar una segona bateria mitjançant el sòcol específic per aquesta funció al mòdul GPRS+GPS i posicionar els ponts d'aquest perquè l'alimentació no arribi des de la placa sinó des de la segona bateria. Encara que en aquest disseny s'ha utilitzat el MAX1674, la solució permanent per aquest possible problema podria ser utilitzar un altre convertidor CC/CC que proporcionés un amperatge de sortida més elevat o utilitzar la primera solució proposada d'inserir els condensadors connectats en paral·lel i prescindir del convertidor CC/CC, i d'aquesta manera la demanda de corrent vindria directament des de la línia de la bateria, sense està aïllada.

Un altre punt important és el circuit de càrrega de bateria. La idea és que la bateria estigui integrada a la placa, i no s'hagi de treure per a poder carregar-la; és per això que el nou disseny incorpora la possibilitat de carregar la bateria mitjançant les entrades d'alimentació que hi ha a la placa. Aquesta incorpora el connector d'alimentació i l'entrada FTDI, encara que aquesta segona no serà d'utilització regular. Buscant solucions dins al mercat actual, s'ha decidit utilitzar el xip MAX1551 (fabricat per l'empresa Maxim Integrated) que incorpora un circuit intern que permet carregar la bateria de liti a partir de dues possibles entrades, pensades una per USB i l'altre per un adaptador AC/DC. La integració d'aquest xip encaixa perfectament amb les necessitats del disseny, ja que aquest disposa d'una entrada d'alimentació de 5V provinent del cable FTDI, que són els mateixos 5V que proporciona el cable USB i una entrada d'alimentació externa que emula la integració del mòdul a un vehicle. Segons el full de dades d'aquest xip, l'entrada d'alimentació USB ha de ser entre 3,7-6V, tenint en compte que el cable en proporciona 5V, el fa compatible. En el cas de l'entrada pensada per un adaptador AC/DC, o sigui l'alimentació externa, el rang ha de ser entre 3,7-7V. Com que anteriorment s'ha comentat que l'alimentació externa ha de suportar una entrada de 12V, i el circuit de càrrega n'accepta 7V màxim, es necessita un regulador de tensió per a rebaixar la tensió del connector d'alimentació externa, per aquest motiu s'ha decidit mantenir el regulador que incorpora el circuit d'alimentació externa del Arduino Uno. Ja que proporciona 5 Volts a la sortida i és un voltatge que està dins el rang de funcionament del MAX1551.

En aquest capítol s'utilitzen les referències [13]-[18]

7.1.5. Circuit de Reset

Al nou disseny també s'hi ha incorporat els circuits per a poder fer un reset manual i un reset d'engegada de la placa. Per ambdós tipus de reset, s'han utilitzat els circuits recomanats pel fabricant Atmel, que és el fabricant del microcontrolador que s'utilitza al projecte, el ATmega328.

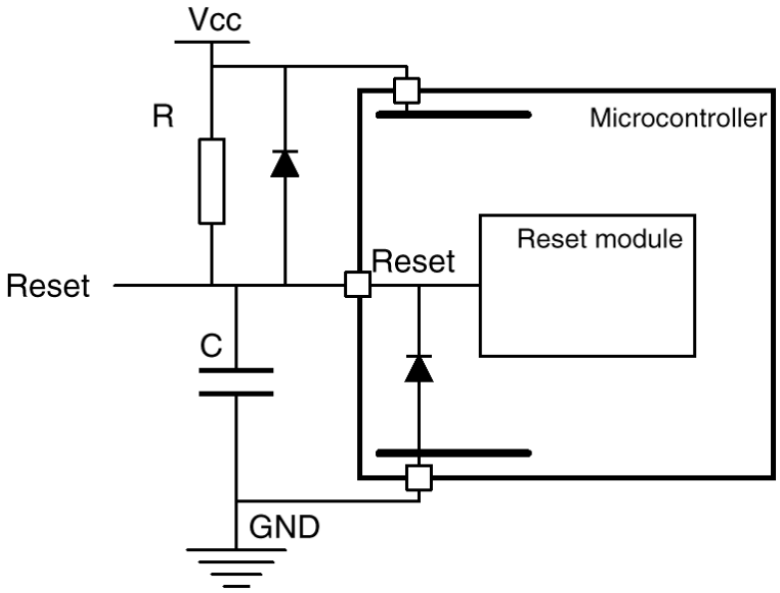

Fig. 7.2. Circuit de reset d'engegada recomanat per Atmel

En aquesta primera figura (Fig. 7.2.) es pot veure el circuit de reset d'engegada que recomana el fabricant Atmel. Segons especifica el fabricant, la resistència R es posa perquè no hi hagi la possibilitat de que hi hagi un reset involuntari. El díode connectat en paral·lel amb aquesta resistència, s'utilitza perquè el condensador C es descarregui ràpidament quan es va atenuant l'alimentació VCC, i d'aquesta manera obtenir un nivell baix al terminal de reset del microcontrolador.

El Atmega328 incorpora un filtre passa baixes per a eliminar possibles sorolls externs. De totes maneres, es recomana incorporar un condensador C, com a protecció addicional contra aquests possibles sorolls elèctrics.

Com s'ha comentat, pel circuit de reset manual, també es fa servir el que recomana el fabricant. Aquest segon conjunt de connexions, tal i com indica Atmel, van connectades amb el circuit que es veu a la Fig. 7.2. Tal i com es pot veure a la figura que es veu a sota (Fig. 7.3.) on la línia de RESET d'aquesta figura és la continuació de la línia amb el mateix nom de la Fig. 7.2. El que es fa en aquest segon circuit és incorporar un interruptor que va de la línia de reset fins a GND, d'aquesta manera quan aquest es prem, la tensió del pin de reset passa a nivell baix provocant un reset del microcontrolador. La resistència R

s'afegeix per a evitar possibles pics de corrent elevats que es poden generar quan es prem el botó de reset.

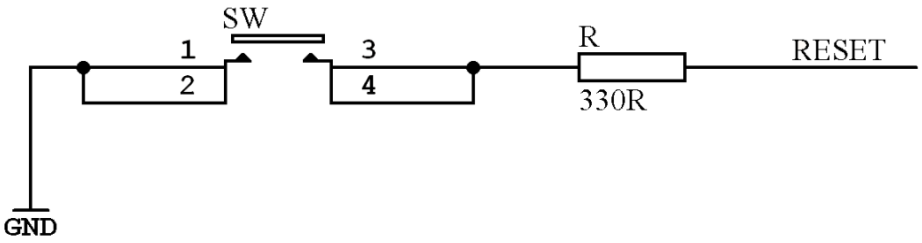

Fig. 7.3. Circuit de reset manual recomanat per Atmel

7.2. Software de la solució propietària

Amb l'entorn de desenvolupament Arduino, s'ha desenvolupat un programa que permet utilitzar la tecnologia GPRS i GPS del SIM908, i d'aquesta manera complir amb l'objectiu de tenir un mòdul de vigilància que pot ser utilitzat per a vehicles.

El programa realitzat, primer de tot, espera que el SIM908 estableixi connexió amb la xarxa GSM, a través d'una targeta SIM que s'hagi introduït. Un cop s'ha establit la connexió GSM, el programa fixa la posició GPS. Quan ha acabat aquest pas, el programa entre en un bucle, i espera que se li enviï un SMS al número de telèfon de la targeta SIM. Si rep un SMS amb la paraula –pos-, el programa respon al mòbil des d'on s'ha enviat aquesta paraula amb un altre missatge de text especificant la posició de latitud i longitud que capta el GPS, i al mateix SMS també hi envia un enllaç a google maps on es pot veure de manera visual on es troba el mòdul. Si el mòdul rep un SMS amb la paraula –ser-, el que fa el SIM908 és connectar-se a la xarxa GPRS i enviar les coordenades a un arxiu d'un servidor web. La web que incorpora aquest servidor està configurada de manera que mostra una finestra amb el google maps, i quan rep les coordenades GPS, es pot veure la posició del SIM908 des de la pàgina web. Un cop ha rebut un missatge o l'altre i ha realitzat les instruccions que s'especifiquen, el programa esborra l'últim SMS rebut, i segueix amb el bucle d'espera de recepció de missatge. En el cas de que el missatge de text, no contingui cap de les dues paraules que s'acaben d'especificar, el programa esborra el missatge rebut i segueix esperant. Si en el missatge que s'envia hi ha escrites les dues paraules –ser- i –pos-, el programa realitzara primer les instruccions que pertanyen a la paraula –pos- i després les de la paraula –ser-. Seguidament s'explica el codi de programa amb més detall i amb l'ajuda de figures.

Seguint l'estructura de software Arduino, primer de tot es fa la declaració de variables en el programa.

```
int8_t answer;
int onModulePin= 2;
char aux_string[30];
char aux_str[30];
char phone_number[12];
char data[100];
char ordre[5];
int data_size;
char aux;
int x = 0;
char N_S,W_E;

char url[] = "www.agotalo.com/modul_cotxe";
char frame[200];
char SMS[200];
char latitude[15];
char longitude[15];
char altitude[6];
char date[16];
char time[7];
char satellites[3];
char speedOTG[10];
char course[10];
```

Fig. 7.4. Declaració de variables

Com es pot apreciar a la Figura de sobre, la majoria de variables que es fan servir són del tipus char, ja que el programa llegeix SMS i coordenades GPS, i això comporta que s'hagin de guardar moltes cadenes de caràcters. Per exemple, les variables per a guardar les coordenades GPS del SIM908, engloben (de dalt a baix) des de la variable latitude, fins la variable course. La variable url conté part del nom web on es pot veure la posició GPS al google maps.

La segona part del programa és la que s'encarrega tant de configurar el Arduino com el SIM908.

```
void setup(){

    pinMode(onModulePin, OUTPUT);
    Serial.begin(115200);

    Serial.println("Engegant...");
    power_on();

    delay(3000);

    // estableix el codi pin (Aga 9959, jordi 4752)
    sendATcommand("AT+CPIN=9959", "OK", 2000);

    delay(3000);

    Serial.println("Connectant a la xarxa...");

    while( (sendATcommand("AT+CREG?", "+CREG: 0,1", 500) ||
            sendATcommand("AT+CREG?", "+CREG: 0,5", 500)) == 0 );

    Serial.print("Configurant mode SMS...");
    sendATcommand("AT+CMGF=1", "OK", 1000);    // sets the SMS mode to text

    Serial.println("Configuracio parametres apn");
        // Configuracio APN , usuari i contrasenya
    sendATcommand("AT+SAPBR=3,1,\"Contype\",\"GPRS\"", "OK", 2000);
    sendATcommand("AT+SAPBR=3,1,\"APN\",\"airtelwap.es\"", "OK", 2000);
    sendATcommand("AT+SAPBR=3,1,\"USER\",\"wap@wap\"", "OK", 2000);
    sendATcommand("AT+SAPBR=3,1,\"PWD\",\"wap125\"", "OK", 2000);
```

Fig. 7.5. Part de configuració setup del programa

Com es pot apreciar a la figura de dalt, el programa comença configurant el pin 2 del Arduino com a sortida per a poder fer servir el LED que incorpora el mòdul GPRS+GSM, seguidament es fixe una comunicació Arduino-PC amb 115200 bits/segon i d'aquesta manera poder veure el comportament del Arduino i del mòdul a través del monitor serial de la IDE de Arduino. La funció power_on el que fa és verificar que el mòdul s'hagi iniciat correctament. Seguidament utilitzant les ordres AT, s'envia el codi pin a la targeta SIM (depenent de la targeta SIM que s'utilitzi s'haurà de configurar en el programa un codi pin o un altre), per a posteriorment connectar el SIM908 a la xarxa GSM, configurar el mode SMS i l'enviament per GPRS. En el cas del projecte s'ha fet servir una targeta SIM de l'empresa Vodafone, la qual té una APN, usuari i contrasenya específics (que es poden trobar a la web de la companyia de manera gratuïta). En el cas que la SIM fos d'una altra companyia telefònica, s'hauria de configurar el GPRS amb un APN, usuari i contrasenya diferents.

```
        Serial.println("Inicialitzant GPS...");

        while ( start_GPS() == 0);

        memset(SMS, '\0', 200);
        // Activacio del GPRS
        while (sendATcommand("AT+SAPBR=1,1", "OK", 20000) == 0)
        {
            delay(5000);
        }

    }
```

Fig. 7.6. Seguiment de la part de configuració setup.

Aquesta segona part del setup que es mostra a la Fig. 7.6. inicialitza el GPS perquè fixi la posició on es troba el SIM908; seguidament es buida la variable SMS, per si hi hagués algun valor, i activa el GPRS.

La tercera part del programa és on s'hi posen les ordres que regiran el comportament tant del Arduino com del SIM908, i està incorporada dins del loop.

```
void loop(){
    Serial.println("Esperant un SMS");
    answer = sendATcommand("AT+CMGR=1", "+CMGR:", 2000);    // Llegeix el primer SMS
    if (answer == 1)
    {
        answer = 0;
        while(Serial.available() == 0);
        // Aquest bucle llegeix les dades del SMS
        do{
            // Si hi ha dades al buffer d'entrada UART, llegeix i comprova la resposta
            if(Serial.available() > 0){
                SMS[x] = Serial.read();
                x++;
```

Fig. 7.7. Primera part de la funció loop del programa

Aquesta tercera part, composta per la funció loop espera que arribi un missatge de text. En el cas que en rebi un, el llegeix i el guarda a la variable SMS.

```
        // Comprova si la resposta desitjada (OK) és la resposta del modul
        if (strstr(SMS, "OK") != NULL)
        {
            answer = 1;
            SMS[x] = '\0'; // Posa el caracter de fi de String
            char ch=SMS[0];
            int i=0;
            while ((ch != '+')&&(i<200)) {ch=SMS[i++];}  /*Busca el numero de telefon: sempre ens
                                                          arriba començat pel signe + amb 12 caracters*/
            if (i<200)
            {
              char *ini=SMS+i-1;
              strncpy(phone_number, ini, 12);   /*Copia els 12 caracters del numero a
                                                la variable phone_number*/
              phone_number[12]='\0';
            }
            Serial.println(phone_number);

        }
    }
}
```

Fig. 7.8. Segona part de la funció loop del programa

Si arriba la resposta OK, el programa, utilitza la variable i com a punter, per a buscar el caràcter + dins de la variable SMS; el fet de buscar el caràcter + és degut a que el format SMS envia una capçalera de caràcters per defecte amb un format especial, on hi ha el número de telèfon des d'on s'ha enviat el SMS, entre d'altres coses. Per exemple un número de mòbil real seria +34623456789, on +34 és el codi del país (en aquest cas Espanya), i els nou nombres restants pertanyen a la resta del número de mòbil. Per tant, per a saber on comença el número, s'utilitza el punter per a posicionar-se al caràcter + i seguidament es copien els 12 caràcters que componen el número de telèfon i es guarden a la variable phone_number. Mirem si "i" és més petit que 200 pel fet que la variable SMS té 200 caràcters, i si el punter en els 200 caràcters del SMS no troba el signe + del començament del número de mòbil, vol dir que s'ha passat.

```
        while(answer == 0);    // Espera a la resposta sense limit de temps

        Serial.println(SMS);
        if (strstr(SMS,"-pos-")>0)    //Busca si SMS conte la paraula -pos-
        {
          Serial.println("Enviant resposta");
          enviar(phone_number);      //Entra a la funcio enviar per enviar la resposta
        }
        if (strstr(SMS,"-ser-")>0)    //Busca si SMS conte la paraula -ser-
        {
          delay(1000);
          send_HTTP();     //Entra a la funcio send_HTTP per enviar la resposta al servidor
          delay(1000);
        }
        x=0;
        memset(SMS, '\0', 200);
        memset(phone_number,'\0', 12);
        answer = sendATcommand("AT+CMGD=1", "+CMGD:", 2000);
    }
    else
    {
        Serial.println("No SMS disponibles");

    }
    delay(3000);
}
```

Fig. 7.9. Tercera part de la funció loop del programa

Mitjançant la instrucció strstr primer es busca si dins SMS hi ha la cadena de caràcters -pos-. Si és així, el programa entra a la funció enviar amb la variable phone_number que té guardat el número de telèfon des d'on s'ha enviat el SMS. Seguidament, també amb la instrucció strstr es mira si la variable SMS conté –ser-, i si és així el programa entra a la funció send_HTTP, per a enviar la posició GPS al servidor web. Finalment es buiden de contingut les variables SMS i phone_number i el programa retorna al principi del loop.

La part de programa que s'ha explicat fins ara és la constituïda per les tres etapes principals, que sol tenir qualsevol programa realitzat amb la IDE de Arduino. Aquest programa també incorpora funcions que, com s'ha pogut comprovar, es criden durant l'execució. Els noms d'aquestes funcions són: power_on, enviar, send_HTTP, sendATcommand, start_GPS, get_GPS i convert2Degrees.

Les funcions convert2Degrees i sendATcommand, són funcions de software lliure recomanades pel fabricant del mòdul GPRS+GSM (Cooking Hacks). La qual s'inclouen com a codi d'exemple a la seva pàgina web. La primera funció esmentada, convert2Degrees, canvia la notació de les coordenades GPS de DD°MM.mmm' a

DD.dddddd°, ja que el google maps interpreta les coordenades en graus, i d'aquesta manera, si el programa les proporciona directament en aquest format, és més fàcil poder comprovar-les de manera visual. L'altre funció sendATcommand serveix per a poder enviar ordres AT de manera satisfactòria. Seguidament es mostren la resta de funcions començant per power_on.

```
void power_on(){

    uint8_t answer=0;

    // Comprova si el mòdul s'inicia correctament
    answer = sendATcommand("AT", "OK", 2000);
    if (answer == 0)
    {
        // Pols power on
        digitalWrite(onModulePin,HIGH);
        delay(3000);
        digitalWrite(onModulePin,LOW);

        // Espera una resposta des del mòdul
        while(answer == 0){     // Envia un AT cada dos segons i espera la resposta
            answer = sendATcommand("AT", "OK", 2000);
        }
    }

}
```

Fig. 7.10. codi de la funció power_on

Aquesta part de codi que es pot veure a la figura de sobre, correspon a la funció power_on, que es crida a la part de setup com es pot veure a la Fig. 7.5. Aquesta funció comprova si el mòdul s'inicia correctament i dóna un pols de tres segons al LED que incorpora el mòdul quan aquest s'inicia.

La segona funció que es representa a continuació és l'anomenada enviar, la qual envia el missatge de text mitjançant el codi que es mostra a la figura següent.

```
void enviar(char* phone){
Serial.println("Llegint coordenades GPS");

    get_GPS();

    Serial.print("Establint mode SMS...");
    sendATcommand("AT+CMGF=1", "OK", 1000);    // Estableix el mode SMS
    Serial.println("Enviant SMS");

    sprintf(aux_string,"AT+CMGS=\"%s\"", phone);
    answer = sendATcommand(aux_string, ">", 2000);    // Envia el numero del SMS
    if (answer == 1)
    {
```

Fig. 7.11. Primera part del codi de la funció enviar

Mirant la figura de sobre s'observa com primer, abans d'enviar el SMS, el programa llegeix les coordenades que proporciona el GPS. Seguidament estableix el mode SMS i prepara el número de mòbil on ha d'enviar el missatge de text.

```
sprintf(aux_string,"latitude=%s, longitude=%s\nhttps://maps.google.es/?q=%s,%s",

    latitude,longitude, latitude,longitude); //Text del SMS que s'envia
```

Fig. 7.12. Segona part del codi de la funció enviar, on es veu el text que s'envia

A la Figura 7.12. es pot veure la part de codi que segueix de manera seqüencial. Aquesta part de codi, realment està posada en una sola línia, però a la figura de dalt s'ha posat en dues línies per a inserir-la a la memòria del treball i que es visualitzi de forma correcta. En aquesta figura es mostra el text que s'envia com a resposta des del mòdul. Que com es pot veure són les coordenades de latitud i longitud, i seguidament l'enllaç per a introduir-lo a la pàgina web de google maps i poder veure la posició de manera més visual.

```
            Serial.println(aux_string);
            Serial.write(0x1A);
            answer = sendATcommand("", "OK", 20000);
            if (answer == 1)
            {
                Serial.print("Enviat ");
            }
            else
            {
                Serial.print("error ");
            }
        }
        else
        {
            Serial.print("error ");
            Serial.println(answer, DEC);
        }
    }
```

Fig. 7.13. Tercera part del codi de la funció enviar.

En aquesta figura de sobre, es pot veure la part final de la funció enviar, la qual, mitjançant les ordres AT, envia el missatge de text al número que li ha enviat el SMS. En el cas que no es pugui enviar, el programa escriu la paraula error, en el cas que s'estigui utilitzant el monitor sèrie.

Una altra funció que s'ha esmentat anteriorment és l'anomenada funció send_HTTP, la qual envia les coordenades GPS a un document php del servidor web. I d'aquesta manera quan es posa la url de la pàgina, es pot veure la posició des d'internet.

```
void send_HTTP(){

    // Inicialitza servei HTTP
    answer = sendATcommand("AT+HTTPINIT", "OK", 10000);
    if (answer == 1)
    {
        // Estableix el paràmetre CID
        answer = sendATcommand("AT+HTTPPARA=\"CID\",1", "OK", 5000);
        if (answer == 1)
        {
            // Estableix la url
            sprintf(aux_str, "AT+HTTPPARA=\"URL\",\"http://%s/demo_sim908.php?", url);
            Serial.print(aux_str);
```

Fig. 7.14. Primera part del codi de la funció send_HTTP

En aquesta primera part de codi, es pot apreciar com s'inicialitza la funció HTTP i tots els paràmetres necessaris. Seguidament el programa estableix la url on es podran visualitzar

les coordenades GPS, que es guardaran a un document del servidor. Aquest enllaç complet per a la pàgina web és, en el cas del projecte que s'ha realitzat, http://www.agotalo.com/modul_cotxe/demo_sim908.php.

```
sprintf(frame, "visor=false&latitude=%s&longitude=%s&altitude=%s&time=%s&satellites=%s&
speedOTG=%s&course=%s\0", latitude, longitude, altitude, date, satellites, speedOTG, course);
```

Fig. 7.15. Segona part de codi de la funció send_HTTP. Guardar coordenades GPS

A la figura 7.15. es pot veure la part de codi que segueix de manera seqüencial. El text d'aquesta part de codi, realment està disposada de manera diferent en el programa (hi ha part del codi de la línia de sota que va a la línia de sobre), però a la figura de dalt s'ha posat en aquest format per a inserir-la a la memòria del treball i que es visualitzi de forma correcta.

```
            Serial.print(frame);
            answer = sendATcommand("\"", "OK", 5000);
            if (answer == 1)
            {
                // Inicia la accio GET
                answer = sendATcommand("AT+HTTPACTION=0", "+HTTPACTION:0,200", 40000);
                if (answer == 1)    {

                    Serial.println(F("Fet!"));
                }
                else   {
                    Serial.println(F("Error en obtenir url"));
                }
            }
            else    {
                Serial.println(F("Error establint url"));
            }
        }
        else      {
            Serial.println(F("Error establint CID"));
        }
    }
    else   {
        Serial.println(F("Error inicialitzant"));
    }

    sendATcommand("AT+HTTPTERM", "OK", 5000);
}
```

Fig. 7.16. Tercera part del codi de la funció send_HTTP

En aquesta part de codi, el programa visualitza la variable frame en el monitor sèrie, i seguidament realitza una acció anomenada GET, per a enviar la posició GPS al servidor

web. Si la resposta que s'obté des del SIM908 és un 1, vol dir que s'ha enviat correctament; i utilitzant l'enllaç per a la pàgina web es podran visualitzar les coordenades GPS en un mapa de google maps. En el cas que no es pugui enviar, el programa visualitzarà un error. Finalment es finalitza el servei HTTP.

Una altra funció que s'ha esmentat i que es crida dins el setup com es pot veure a la Fig. 7.6., és l'anomenada start_GPS.

```
int8_t start_GPS(){

    unsigned long previous;

    previous = millis();
    // Inicia el GPS
    sendATcommand("AT+CGPSPWR=1", "OK", 2000);
    sendATcommand("AT+CGPSRST=0", "OK", 2000);

    // Espera que es fixi el GPS
    while(( (sendATcommand("AT+CGPSSTATUS?", "2D Fix", 5000) ||
        sendATcommand("AT+CGPSSTATUS?", "3D Fix", 5000)) == 0 ) &&
        ((millis() - previous) < 90000));

    if ((millis() - previous) < 90000)
    {
        return 1;
    }
    else
    {
        return 0;
    }
}
```

Fig. 7.17. Codi de la funció start_GPS

En aquesta funció el que fa el codi és iniciar el GPS del SIM908, i espera que es fixi la posició GPS, o sigui que aquest trobi els satèl·lits per a posicionar-se. Si en 90 segons no troba la posició, la funció torna a iniciar el GPS i torna a esperar que es fixi. Un cop el GPS ha fixat la posició on es troba, el programa surt del bucle.

L'altra funció relacionada amb el GPS que s'esmenta és l'anomenada get_GPS, la qual té la finalitat d'agafar les dades proporcionades pel GPS del SIM908 i guardar-les en diferents variables.

```
int8_t get_GPS(){

    int8_t counter, answer;
    long previous;
    // Primer agafa la cadena de caracters NMEA
    // Neteja el buffer d'entrada
    while( Serial.available() > 0) Serial.read();
    // Sol·licita una cadena de caràcters basica
    sendATcommand("AT+CGPSINF=0", "AT+CGPSINF=0\r\n\r\n", 2000);

    counter = 0;
    answer = 0;
    memset(frame, '\0', 100);    // Inicia la cadena de caràcters
    previous = millis();
    // Aquest bucle espera la cadena NMEA
    do{
        if(Serial.available() != 0){
            frame[counter] = Serial.read();
            counter++;
            // Comprova si la resposta desitjada es la resposta del modul
            if (strstr(frame, "OK") != NULL)
            {
                answer = 1;
            }
        }
        // Espera a la resposta sense limit de temps
    }
    while((answer == 0) && ((millis() - previous) < 2000));
    frame[counter-3] = '\0';
```

Fig. 7.18. Primera part del codi de la funció get_GPS

Com es pot observar a la figura de dalt, el que fa el programa en aquesta part de codi és esperar la informació de les coordenades GPS del SIM908, llegir-les i guardar-les a una variable.

```
    // Analitza la cadena
    strtok(frame, ",");
    strcpy(longitude,strtok(NULL, ",")); // Agafa longitud
    strcpy(latitude,strtok(NULL, ",")); // Agafa latitud
    strcpy(altitude,strtok(NULL, ".")); // Agafa altitud
    strtok(NULL, ",");
    strcpy(date,strtok(NULL, ".")); // Agafa la data
    strtok(NULL, ",");
    strtok(NULL, ",");
    strcpy(satellites,strtok(NULL, ",")); // Agafa els satelits
    strcpy(speedOTG,strtok(NULL, ",")); // Agafa velocitat sobre el fons. Unitats en nusos.
    strcpy(course,strtok(NULL, "\r")); // Agafa curs

    convert2Degrees(latitude);
    convert2Degrees(longitude);

    return answer;
}
```

Fig. 7.19. Segona part del codi de la funció get_GPS.

En aquesta figura que es mostra a sobre, es pot observar la part de codi dins de la funció get_GPS que s'encarrega de separar per parts les dades proporcionades pel GPS. Com es pot comprovar les dades estan en una sola variable i el programa les separa en longitud, latitud, altitud, data, satèl·lits que detecta el GPS, velocitat sobre el fons en nusos i direcció en la que es mou (en el cas que no hi hagi velocitat serà 0).

Per a la realització d'aquest programa s'han utilitzat codis de software lliure proporcionats per l'empresa Cooking Hacks, la qual és fabricant del mòdul GPRS+GPS que s'ha fet servir pel projecte. La tasca de realitzar aquest programa ha estat una de les més dificultoses durant el projecte ja que primer de tot s'han hagut d'entendre els codis d'exemple de software lliure, i un cop fet això s'han hagut d'agafar les diferents funcions i parts de codi i englobar-les en un sol programa. Durant la programació hi hagut diversos problemes per a fer funcionar el codi fins que s'ha aconseguit. Per exemple, per a fer funcionar el mode HTTP per enviar la posició GPS al servidor web, primer de tot s'ha d'iniciar aquest mode utilitzant una ordre AT. Un dels problemes que hi ha hagut és oblidar-se d'iniciar el mode HTTP, i sense això el programa sempre responia amb error a l'hora d'enviar la posició al servidor. Una altra de les grans dificultats ha estat la de quan arriba un SMS, poder guardar el número de telèfon des d'on s'ha enviat el SMS. En aquest cas va ser una ajuda poder veure el SMS rebut, en el monitor sèrie del ordinador, ja que d'aquesta manera es va poder comprovar com a la variable on es guarda el SMS rebut, abans del text del SMS hi ha una capçalera on es veu el número de telèfon des d'on s'ha

enviat el missatge de text, entre d'altres coses. D'aquesta manera la solució ha estat utilitzar una variable com a punter dins de la variable on es guarda el SMS i d'aquesta manera trobar el número de telèfon i guardar-lo en una altra variable, per així, poder enviar una resposta. Amb aquesta solució del punter per a trobar el número de telèfon, va sorgir la idea d'utilitzar un altre punter per a buscar paraules clau dins el SMS com –pos- o –ser-, i d'aquesta manera que el programa enviés una cosa o una altra depenent del missatge rebut. Al primer moment semblava senzill, però al utilitzar dos punters, aquests s'interferien un amb l'altre i la variable on s'havia de guardar només el número de telèfon es concatenava amb la paraula –pos- o –ser- darrera i llavors no es podia enviar el SMS de resposta. La solució a aquest conflicte ha estat utilitzar la instrucció de C strstr, la qual busca directament la paraula –pos- o –ser-, depenent el que es vulgui, dins la variable del SMS sense haver de crear una variable que s'utilitzi com a segon punter, i d'aquesta manera també es simplifica el codi i és més fàcil de poder ampliar les possibilitats del programa ja que afegint una altra instrucció strstr que busqui una paraula diferent que s'especifiqui, es pot programar molt fàcilment que hi hagi més possibilitats a part de la –pos- i la –ser-. El fet que aquestes paraules es posin entre guions és degut a que en el cas de que hom enviés un SMS al número de la targeta SIM del SIM908, el programa no li respongués per equivocació amb les coordenades GPS. Per exemple si el xip SIM908 rep un SMS amb el text: Servei de posicionament. El programa trobaria la paraula ser i la paraula pos dins el SMS i executaria la part de codi relacionada amb aquestes funcions. És per això que posant les paraules clau entre guions, s'intenta minimitzar aquests tipus de conflictes. S'ha de dir també, que s'ha perdut molt de temps a l'hora de fer proves amb el programa per culpa d'una part de hardware, l'antena GPS. Al fer la funció start_GPS, el GPS ha de fixar la posició perquè el programa surti del bucle de la funció, i en ocasions aquest ha trigat més de 30 minuts en fixar-la, sobretot quan s'han realitzat proves en ubicacions totalment interiors.

El codi del programa complet es pot trobar a la part d'annexos d'aquest treball.

En el programa realitzat el mòdul està configurat perquè quan s'enviï la paraula –ser-, el SIM908 envia les dades a través d'internet, mitjançant un scrpt php del ordinador. Per a fer possible aquesta part de codi i que funcioni, s'ha de poder disposar d'un servidor que suporti documents del tipus php. En aquest projecte s'ha pogut disposar ja d'un servidor, però en el cas que no es tingués, s'hauria de crear-ne un de nou. En aquest servidor s'hi ha

de tenir un domini web, que en el cas del projecte es fa servir el domini www.agotalo.com. Els documents d'exemple que s'han de pujar en el servidor dins del domini web vénen donats als exemples de software lliure de la pàgina web de l'empresa Cooking Hacks (www.cooking-hacks.com). Són tres documents. La finalitat d'aquests documents és quan reben una posició GPS del mòdul GPRS+GPS, posen un marcador de la posició al mapa de google maps que es visualitza a la web i que es va actualitzant cada cinc segons. Les posicions que es reben al servidor, es van guardant en un arxiu de text .txt. Un altre document del tipus php és el que visualitza el mapa de google maps en el domini web, i agafa les coordenades del arxiu de text per visualitzar la última posició enviada i a part posar un marcador al mapa.

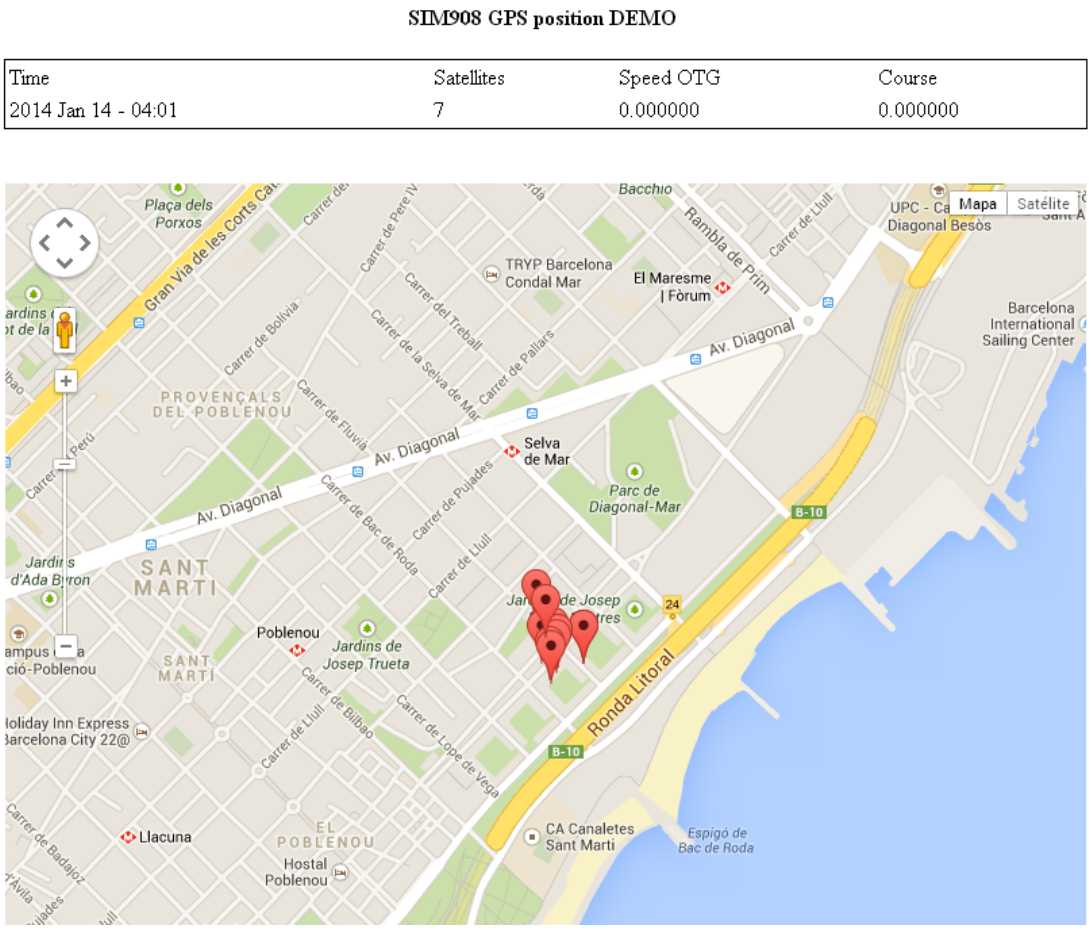

Fig. 7.20. Visualització de la pàgina web

Per a veure més clar, a la Fig. 7.20. es mostra el contingut de la pàgina web, on a la part de dalt es poden veure algunes dades de la última posició rebuda pel servidor, i a la part de

sota es visualitza el mapa de google maps amb els marcadors posats de totes les posicions rebudes, o sigui, totes les posicions que conté l'arxiu de text GPS.txt.

Al document Sim_908.php, que és un dels tres documents d'exemple esmentats, se li han de fer unes petites modificacions perquè funcioni. Primer de tot, per a mostrar el mapa de google maps a la pàgina web que es fa servir (www.agotalo.com), es necessita posar una clau API a la part de codi del document php Sim_908 que es mostra a continuació: http://maps.googleapis.com/maps/api/js?key=CLAU_API&sensor=false. El text CLAU_API s'ha de substituir per una API. Una clau API és simplement un identificador que google obliga a utilitzar si es vol mostrar un mapa de google maps en un domini web o aplicació mòbil. D'aquesta manera google dóna una interfície amb amb la qual es pot interactuar i així poder posar dades i informació sobre els seus mapes utilitzant un domini propi. Aquest identificador és gratuït i per a aconseguir-lo s'ha d'anar a la pàgina: https://code.google.com/apis/console. Seguidament s'ha d'entrar a un compte de correu de google, ja que sinó no es pot obtenir la clau. Un cop fet això, s'ha d'anar a l'apartat de Services del menú que es troba a l'esquerra de la pantalla i activar els serveis google maps API v3, google Coordinate API i google Engine API tal i com es mostra a la imatge de sota (Fig 7.21.).

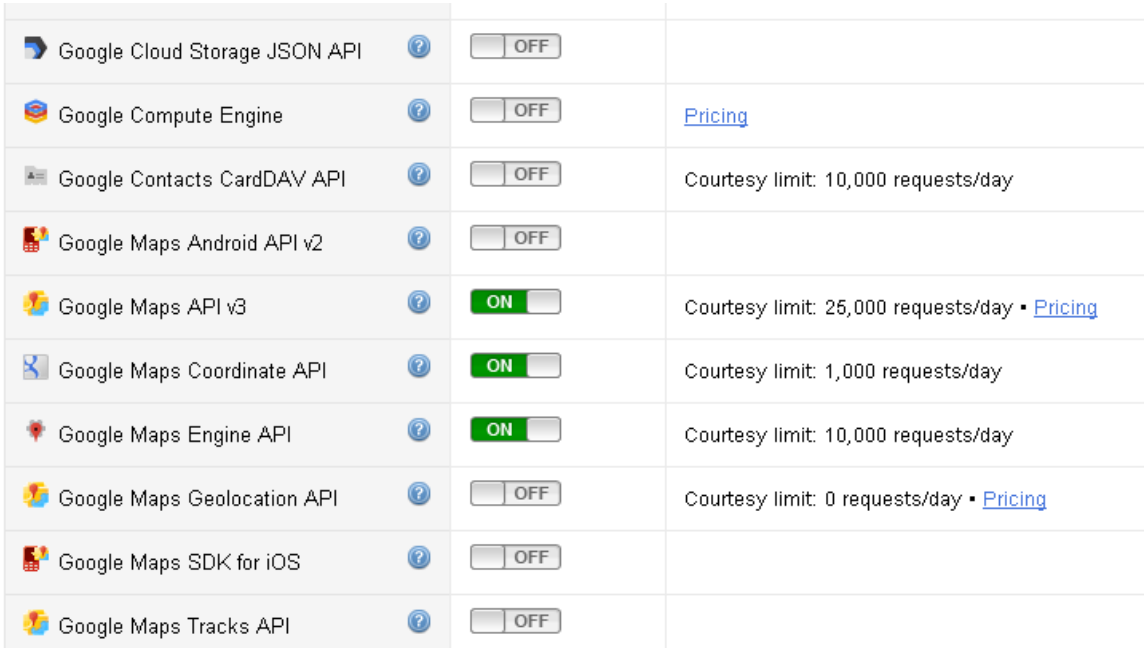

Fig. 7.21. Serveis activats dins la consola del compte de google

Un cop s'han activat aquests serveis, s'ha d'anar a l'apartat API access del menú de l'esquerra i allà clicar Create new Browser key... que es troba a la part de sota de la finestra. Un cop fet això apareixerà una clau API que ha generat google i que ja pot ser utilitzada. Com a resum del que s'acaba de comentar seguidament es mostra una figura per a veure-ho de forma més visual.

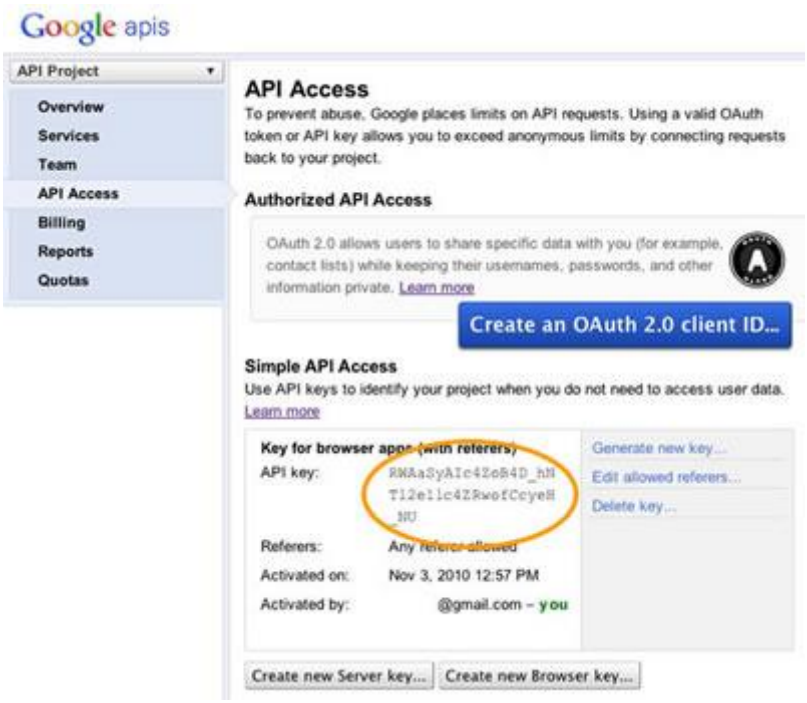

Fig. 7.22. Com generar la clau API

Com es pot apreciar a la figura de dalt, entrant dins l'apartat API accés i havent clicat el botó Create new Browser key..., es genera un identificador que, en el cas de la figura es mostra dins d'un cercle de color taronja.

Un cop es té la clau, s'ha de fer el pas anteriorment esmentat de canviar en el document sim_908.php el text CLAU_API per l'identificador que s'ha generat.

Una altra modificació és la de poder centrar el mapa, això vol dir, mostrar un punt inicial en el mapa de google maps que es mostra a la web. Per a fer això s'ha de modificar la part de codi següent: var myCenter=new google.maps.LatLng(X,Y); on s'ha de canviar la X i la Y per les coordenades de latitud i longitud, respectivament, per a poder visualitzar una posició d'inici del mapa. En el cas del projecte, el mapa s'ha centrat al Tecnocampus Mataró-Maresme, és per això que s'han posat les coordenades d'aquesta ubicació:

var myCenter=new google.maps.LatLng(41.52786301406348,2.4347056499999553);

Un cop fetes aquestes petites modificacions es pugen els documents al servidor, dins el domini que es fa servir (www.agotalo.com) i es guarden a la carpeta creada en aquest domini, que és l'anomenada modul_cotxe. D'aquesta manera entrant a la direcció http://www.agotalo.com/modul_cotxe/demo_sim908.php es mostra el contingut del domini web tal i com es pot veure a la Fig. 7.20. (amb la diferència que en aquesta figura el mapa de google maps està centrat a Barcelona i no al Tecnocampus Mataró-Maresme). S'ha de tenir en compte que la direcció del domini on es mostren les coordenades ha de ser el mateix que el del programa del Arduino, és per aquest fet que la variable url del programa Arduino (veure Fig. 7.4.) conté la cadena de caràcters www.agotalo.com/modul_cotxe. Amb tot això explicat queda configurat el servidor i la possibilitat de poder veure les coordenades GPS del SIM908 en una pàgina web d'internet.

8. Planificació del projecte

A: Posada en marxa i programació del prototip establint la comunicació a través de mòbil.

B: Configuració d'un servidor al PC per a la preparació de la comunicació del sistema prototip amb l'ordinador.

C: Posada en marxa i programació del prototip establint la comunicació a través del PC.

D: Programació del prototip per a poder establir comunicació a través de mòbil i PC conjuntament.

E: Elecció de materials necessaris per la solució pròpia proposada, prenent com a referència el prototip. S'ha tingut en compte el fet d'usar un microcontrolador que es pugui programar fent servir la IDE de Arduino ja que és la utilitzada en el prototip del projecte.

F: Disposició dels elements en una placa per reduir components i obtenir un nou producte.

G: Disseny final de la solució pròpia establint les connexions entre ells.

H: Estudi del cost i les possibilitats de fabricació del producte.

I: Redacció de la part de la memòria relacionada amb el desenvolupament de l'aplicació utilitzant el prototip.

J: Redacció de la part de la memòria relacionada amb l'aplicació d'una solució propietària

K: Redacció de la part de la memòria relacionada amb l'impacte ambiental i revisió de la resta d'apartats.

L: Redacció de la part de la memòria relacionada amb les conclusions del treball i repàs final del treball.

Activitats	Inici	Duració (dies)	final
A	08/11/2013	4	12/11/2013
B	12/11/2013	7	19/11/2013

C	19/11/2013	3	22/11/2013
D	22/11/2013	3	25/11/2013
E	25/11/2013	7	02/12/2013
F	28/11/2013	7	05/12/2013
G	02/12/2013	7	09/12/2013
H	09/12/2013	11	20/12/2013
I	18/11/2013	11	29/11/2013
J	02/12/2013	11	13/12/2013
K	20/12/2013	14	04/01/2014
L	04/01/2014	6	10/01/2014

Taula 8.1. Duració de les tasques realitzades en el projecte

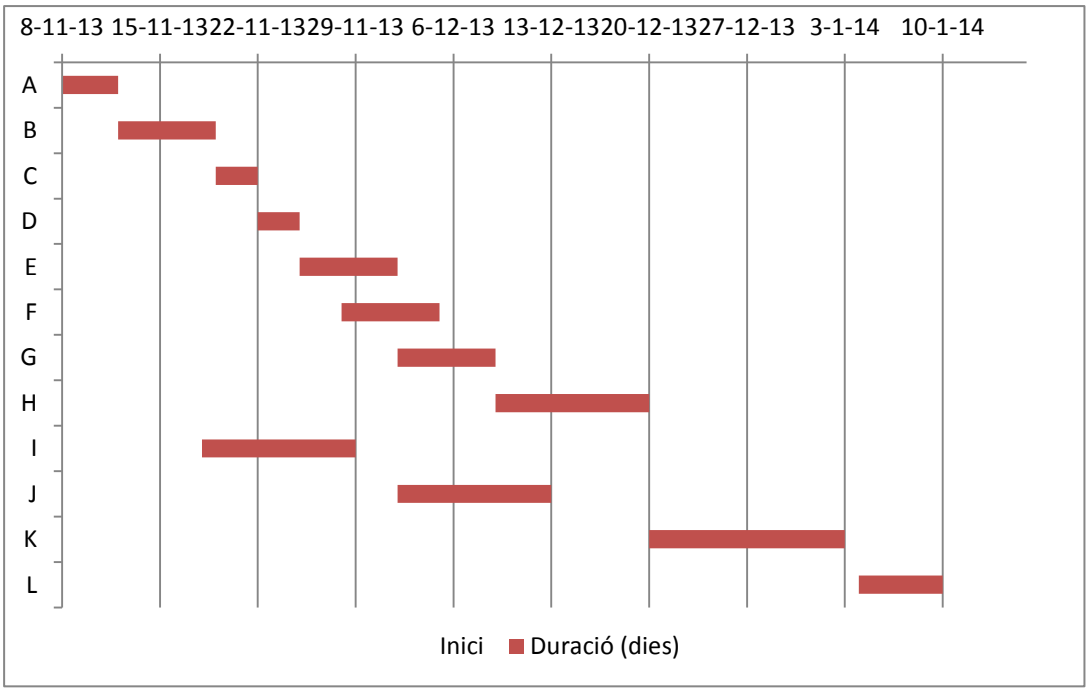

Fig. 8.1. gràfica del diagrama de Gantt amb les tasques del projecte

9. Impacte mediambiental

La generació de residus elèctrics i electrònics és una gran debilitat mediambientalment parlant. I en el cas del dispositiu que es desenvoluparà de manera real en el projecte, s'ha de tenir en compte que quan s'acabi la vida útil d'aquest, es convertirà en un residu.

Segons l'article 25 de la directiva 2012/19/UE del parlament Europeu i del consell, del 4 de Juliol de 2012, sobre residus d'aparells elèctrics i electrònics (RAEE), un aparell d'informàtica i de telecomunicacions petit (sense cap dimensió exterior superior a 50 cm) haurà de ser valoritzat en un 75% i haurà de ser preparat per la seva reutilització i reciclatge en un 55%. Per a aconseguir això, la directiva diu que s'hauran d'organitzar uns sistemes que permetin retornar als posseïdors finals i als distribuïdors, almenys gratuïtament, els residus elèctrics i electrònics. És per això que s'hauran d'establir punts de recollida selectiva i s'hauran de gestionar els residus elèctrics i electrònics tal i com indica aquesta directiva Europea, i en el cas de l'estat espanyol, acompanyada pel Reial Decret 208/2005, del 25 de febrer, sobre aparells elèctrics i electrònics i la gestió dels seus residus.

10. Conclusions i possibles millores

En aquest projecte s'ha obtingut un prototip que és capaç de comunicar-se a través d'un mòbil i una pàgina web. Complir aquest objectiu no ha estat senzill. Una de les parts més complicades i la que ha demandat més temps ha estat la d'arribar entendre tota la part electrònica i de programació que es presenta, però a part de ser de les més complicades, també ha servit per adonar-se de la importància de tots o gran part dels coneixements que s'han adquirit durant la carrera i que s'apliquen directament. Realitzar aquest treball ha estat un aprenentatge constant i és una gran motivació per a seguir aprenent sobre l'electrònica en el futur.

El que s'ha aconseguit en aquest projecte de manera més detallada ha estat poder entendre el funcionament de les plaques de desenvolupament basades en microcontroladors i les tecnologies GPS i GPRS/GSM, entendre un suficientment un llenguatge de programació per a poder desenvolupar una aplicació que es comuniqués a través de les tecnologies esmentades i poder fer un nou disseny d'una placa de desenvolupament donant-li autonomia i reduint components.

Aquest projecte és obert i es poden plantejar línies futures de treball per a ampliar-lo i millorar-lo. Comentant el que s'ha aconseguit i el que es podria millorar, a la part de hardware s'ha desenvolupat una nova placa de desenvolupament que dóna autonomia i té menys components que la original, però no incorpora el mòdul GPS+GPRS/GSM a la mateix placa, sinó que aquest és un perifèric que s'hi connecta, per tant com a primera millora es podria integrar el xip SIM908 al mateix esquemàtic que la placa de desenvolupament i no com un perifèric. Una altra millora seria la de poder garantir que el mòdul no s'apagui per falta de consum, ja que amb les connexions actuals del disseny propi, el SIM908 pot necessitar en algun moment més corrent que el que es pot proporcionar. Per a realitzar aquesta millora es podria prescindir del convertidor CC/CC i posar una línia de condensadors des de la bateria fins el SIM908 directament. Aquesta és una solució però n'hi pot haver de diferents, com per exemple tenir un convertidor que proporcioni més corrent de sortida o inserir una segona bateria al SIM908.

En l'apartat de software, es podria millora el fet que el pin de la SIM està configurat dins el codi, per tant, si es canvia la targeta SIM s'hauria de canviar el codi de

programa. Per a millorar això es podria programar un primer missatge que s'enviés al dispositiu el codi PIN de la targeta SIM, i que aquest es guardés a la memòria EEPROM que incorpora el microcontrolador. D'aquesta manera es podria canviar la SIM sense haver de canviar el codi de programa.

Com a possibles línies de treball futures, en hardware el que es podria fer és dissenyar les plaques de circuit imprès del dispositiu amb el SIM908 integrat i fabricar-ne un prototip propi. En Software es podrien ampliar les possibilitats del dispositiu i que aquest faci més coses, com poden ser activar una alarma o avisar amb un SMS quan s'excedeix una velocitat concreta o quan està a una posició concreta.

Una altra dificultat que s'ha trobat en el projecte ha estat, la de realitzar les comunicacions entre el dispositiu i la pàgina web ja que finalment s'ha necessitat ajuda externa per a aconseguir-ho.

L'antena GPS ha estat un dels motius pels quals s'ha perdut molt de temps. A l'hora de realitzar proves experimentals i provar els programes més d'un cop s'ha hagut d'esperar més de 30 minuts perquè el GPS fixes la posició en que es troba inicialment. És per això que com a millora, també es proposa canviar l'antena GPS i cercar-ne una de més garanties.

La planificació ha estat un dels temes a millorar també, ja que no s'han tingut en compte possibles problemes que generessin pròrrogues en els terminis acordats a la planificació i això ha provocat que es concentrés molta feina al final.

Tot i els entrebancs i problemes, ha estat molt reconfortant poder aprendre tantes coses i agafar motivació per a seguir en aquest camp professionalment parlant.

11. Bibliografia i referències

[1] http://info.lineadirecta.com/buscador?p_auth=s9Hig2ND&p_p_auth=wcF59Vm3&p_p_id=20&p_p_lifecycle=1&p_p_state=exclusive&p_p_mode=view&_20_struts_action=%2Fdocument_library%2Fget_file&_20_groupId=10538&_20_folderId=73282&_20_name=4002

[2] www.grupodetector.com/uploaded/noticias/pdf/2012_09_24_08_36_14.PDF

[3] http://www.mobilefleet.es

[4] http://xexun.es/

[5] http://www.arduino.cc

[6] http://openaccess.uoc.edu/webapps/o2/bitstream/10609/664/1/40103tfc.pdf

[7] http://upcommons.upc.edu/pfc/bitstream/2099.1/3729/1/53820-1.pdf

[8] http://www.atmel.com/images/doc7799.pdf

[9] http://www.atmel.com/images/atmel-8271-8-bit-avr-microcontroller-atmega48a-48pa-88a-88pa-168a-168pa-328-328p_datasheet.pdf

[10] http://www.cooking-hacks.com

[11] http://wm.sim.com/

[12] http://educacionadistancia.juntadeandalucia.es/pre/profesorado/pluginfile.php/2883/mod_resource/content/1/Apuntes_ARDUINO_Nivel_ENTERAILLO.pdf

[13] https://www.sparkfun.com/datasheets/DevTools/FTDI%20Cable%205V.pdf

[14] http://datasheets.maximintegrated.com/en/ds/MAX1551-MAX1555.pdf

[15] http://datasheets.maximintegrated.com/en/ds/MAX1674-MAX1676.pdf

[16] http://www.cooking-hacks.com/documentation/tutorials/arduino-solar

[17] http://www.atmel.com/images/atmel-2521-avr-hardware-design-considerations_application-note_avr042.pdf

[18] https://developers.google.com/maps/documentation/javascript/tutorial#api_key