Universitat de Girona
Escola Politècnica Superior

Treball final de grau

Estudi:Grau en Enginyeria Electrònica Industrial i Automàtica

Títol: Instrumentació, adquisició i tractament de dades del prototip SMC 2015

Document: 1.Memòria

Alumne: David Abad Ayats

Tutor: Bianca Mariela Innocenti Badano
Departament: EEEA
Àrea: ESA

Convocatòria (mes/any) setembre/2015

INDEX

1.INTRODUCCIO

En aquest projecte es detallen la instrumentació i les especificacions que s'han usat en el prototip de moto elèctrica presentat per l'UdG Racing Team a la Smart Moto Challenge 2015, celebrada al Circuit de Catalunya. Bàsicament s'expliquen les diferents normes que el prototip ha de complir i la solució que s'ha implementat per aconseguir-ho.Com que aquest projecte ha estat pensat per ser implementat per l'equip UdG Racing Team, s'ha tingut molt en compte que les solucions fossin fàcils de implementar i de baix cost.

En la primera part s'explicaran tant la competició com els requeriments que imposen als equips, ja que son el que estableixen el marc d'aquest projecte. En aquesta primera part també s'explicarà amb detall el material cedit per la companyia ELMOTO a tots el equips. En la segona meitat del treball s'explica les possibles solucions i es detalla quina s'ha escollit i perquè. En aquesta segona part on també s'explica com s'han implementat posteriorment aquestes solucions.

1.1. Antecedents

Per a la realització d'aquest projecte s'ha partit de tota l'experiència que l'equip va adquirir en la competició de l'any 2014, ja que el material donat per la organització de la competició és el mateix. La normativa ha estat ampliada per a l'edició 2015, així com els requeriments que regulen la quantitat de sistemes intel·ligents que ha tenir el prototip.

1.2 Objecte

Aquest projecte ha servit per dissenyar la instrumentació i els perifèrics necessaris perquè el projecte faci el requerit per a la competició. Per a això s'ha fet un petit estudi per escollir la solució mes adequada. Al material bàsic que aporta la organització s'hi ha incorporat un sensor de velocitat, un sensor de l'estat de la bateria, un sistema de detecció de l'estat de paviment, un sensor de cavallet, així com un sistema de connectivitat per a dispositius intel·ligents basat en Bluetooth. També s'ha adaptat el sistema de llums i de seguretats per poder complir les normatives establertes.

S'ha utilitzat un Arduino Uno per gestionar la informació de tota la instrumentació externa que s'ha introduït, això com l'enviament de dades entre la moto i el mòbil intel·ligent. També s'ha programat una màquina d'estats per assegurar que les mesures de seguretat es compleixen abans de poder fer servir la moto.

1.3 Especificacions i abast

Aquest projecte es centra en escollir i adaptar sensors que es poden trobar al mercat a la motocicleta. També es farà la programació de l'Arduino per tal que pugui gestionar aquests sensors i es comuniqui amb el dispositiu mòbil.

En aquest projecte no s'explicarà de forma exhaustiva el codi de l'aplicació mòbil ni les bases de dades.

2. COMPETICIO SMART MOTO CHALLENGE

La Smart Moto Challenge, a la que ens referirem a partir d'ara com a SMC, es una competició universitària que s'inspira en la Formula Student. L'objectiu de la competició és el de dissenyar i crear un prototip de ciclomotor elèctric que ha d'estar destinat a un us específic. En l'edició 2015 es demanava un prototip de motocicleta enfocat al mon de la missatgeria i l'entrega de menjar a domicili. La competició segueix una normativa pròpia que regula els prototips. Per al certament d'enguany s'ha seguit la última revisió de la competició que és la 6.2.

2.1 Funcionament de la competició

La competició esta dividida en dues fases que es realitzen de forma altercada durant els dies de les proves. La primera varietat de proves son les proves dinàmiques. En els esdeveniments dinàmics es compara el rendiment dels prototips presentats per als diferents equips. Per poder participar en les proves dinàmiques s'ha de passar una revisió tècnica visual per assegurar que el ciclomotor compleix les normatives de seguretat establertes i una prova de moll en que es ruixa el prototip en aigua i es comprova que no hi ha cap filtració i que la moto pot córrer en cas de pluja o asfalt moll.

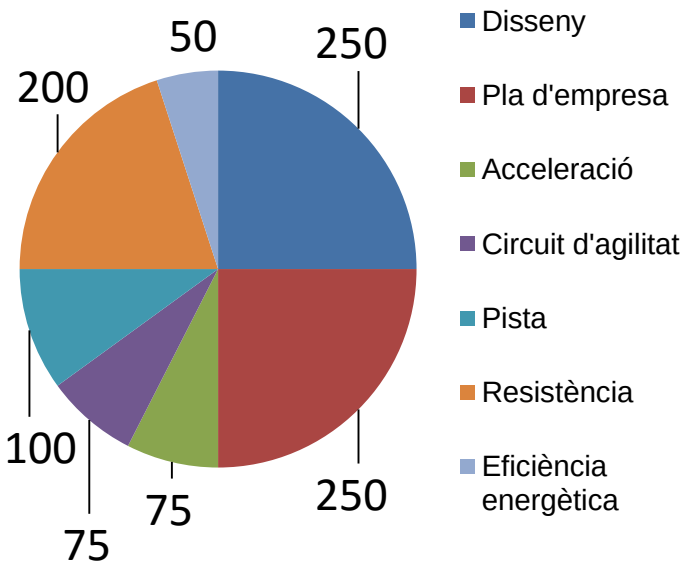

Figura 1. Repartiment de punts

Dins de les proves dinàmiques hi podem trobar la prova d'acceleració, l'autocross, la prova de resistència i la prova dels cons. Totes les proves es realitzen amb dos pilots,

es a dir que cada equip fa les proves dues vegades, una amb cada pilot. En la prova d'acceleració es mesura el temps que triguen els pilots en recórrer una distancia de cinquanta metres. Tot seguit, a la prova de pista cada equip fa quatre voltes a un circuit, dos amb cada pilot, i es registra el millor temps. A la prova de resistència cada equip ha de fer cinc voltes a un circuit amb cadascun dels pilots. El prototip ha de dur tot el sistema de llums encès. En aquesta prova es registra el millor temps per pilot. Per últim tenim la prova dels cons que consisteix en una línia recta de cons que els pilots han de fer d'anada i de tornada.

Durant la competició també es participa en dues proves estàtiques. Aquestes proves son la prova de disseny i la presentació del pla d'empresa de l'equip. La primera prova estàtica que es passa és el pla d'empresa. Aquest pla d'empresa és un exercici en que es te en compte tot el que implicaria fer una empresa per comercialitzar el prototip dissenyat. Aquest pla d'empresa estudia un període de tres anys i te en compte des de un estudi de mercat, la producció i la venta anuals, les contractacions, el local, les despeses i fins i tot el tipus de estratègia de ventes que s'utilitzaria. Aquesta presentació es fa en anglès i ha de ser inferior a deu minuts.

La prova de disseny es la última prova estàtica. Un grup d'experts pregunta fa preguntes a cada equip sobre els seus prototips. També es demana que s'expliquin o es defensin les solucions escollides. Cadascuna d'aquestes proves representa una quarta part de la puntuació total de la competició com es mostra a la figura 1.

2.2. Normativa electrònica

Tot seguit s'expliquen les normatives especifiques que el reglament detalla per a la part d'electrònica i electricitat. Bàsicament aquesta part detalla les mesures de seguretat que el prototip ha de complir i els requeriments mínims de la part de connectivitat i components intel·ligents.

2.2.1. Seguretats

Es demana que el prototip tingui un avisador acústic per saber quan la moto esta engegada. Aquest avis ha de tenir una durada d'entre un i tres segons i ha d'arribar a

70dB en un radi de dos metres. En accionar els frens s'ha de desactivar la senyal del puny de gas, fent impossible frenar i accelerar alhora. En desplegar el cavallet la moto ha de quedar sense tracció, és a dir que el motor no pot rodar amb la moto aixecada de terra. També es demana que hi hagi un boto d'emergència que pugi desactivar completament la moto. Per últim es demana que la moto tingui un avisador per saber que el prototip es troba en un l'estat normal de funcionament. Es considera estat normal de funcionament aquell estat en que la clau esta donada, el cavallet esta plegat i el boto d'emergència no esta activat.

2.2.2. Especificacions per als components intel·ligents

En aquest apartat s'expliquen els requeriments mínims que la competició imposa a tots els equips participants pel que fa a sistemes intel·ligents. La organització demana que com a mínim es llegeixi l'estat de la bateria, l'estat de l'asfalt i les coordenades GPS de la moto. Aquestes dades s'han d'enviar a un dispositiu mòbil a traves de Bluetooth, i aquest a la seva vegada ha d'enviar aquestes dades a un servidor. Cada equip ha de ser capaç de veure les dades emmagatzemades al servidor a traves de una aplicació web.

L'equip UdG Racing Team ha decidit afegir la velocitat, l'acceleració, i l'estat de la moto. L'estat de la moto consisteix a saber si la moto esta en l'estat de tracció o no, aquest estat es dóna quan es compleixen totes les condicions perquè el motor pugi girar. Tot i que la competició no estableix un límit màxim d'informació per enviar, l'equip ha cregut que amb aquesta ampliació cobríem les necessitats que un potencial client pogués tenir.

3. SISTEMA DE BATERIA

Com s'ha dit anteriorment la competició proporciona la bateria de 54.6V i 31A/h que s'ha d'utilitzar per al la motocicleta. Aquesta bateria be tancada i segellada en una capsa d'alumini per tal que ajudi a dissipar la calor de les cel·les al descarregar-se. Aquesta bateria també inclou el seu propi sistema de manteniment o BMS.

El BMS es el sistema que s'encarrega de regular la càrrega i descarrega de les bateries. El BMS també s'encarrega de assegurar-se que les cel·les no es descarreguin mai per sota d'un nivell de seguretat, ja que si les cel·les queden molt de temps en un estat de càrrega molt baix no es podrien recuperar. El sistema que regula les bateries també incorpora dues sondes de temperatura per assegurar que no s'arriba mai a un punt perillós.

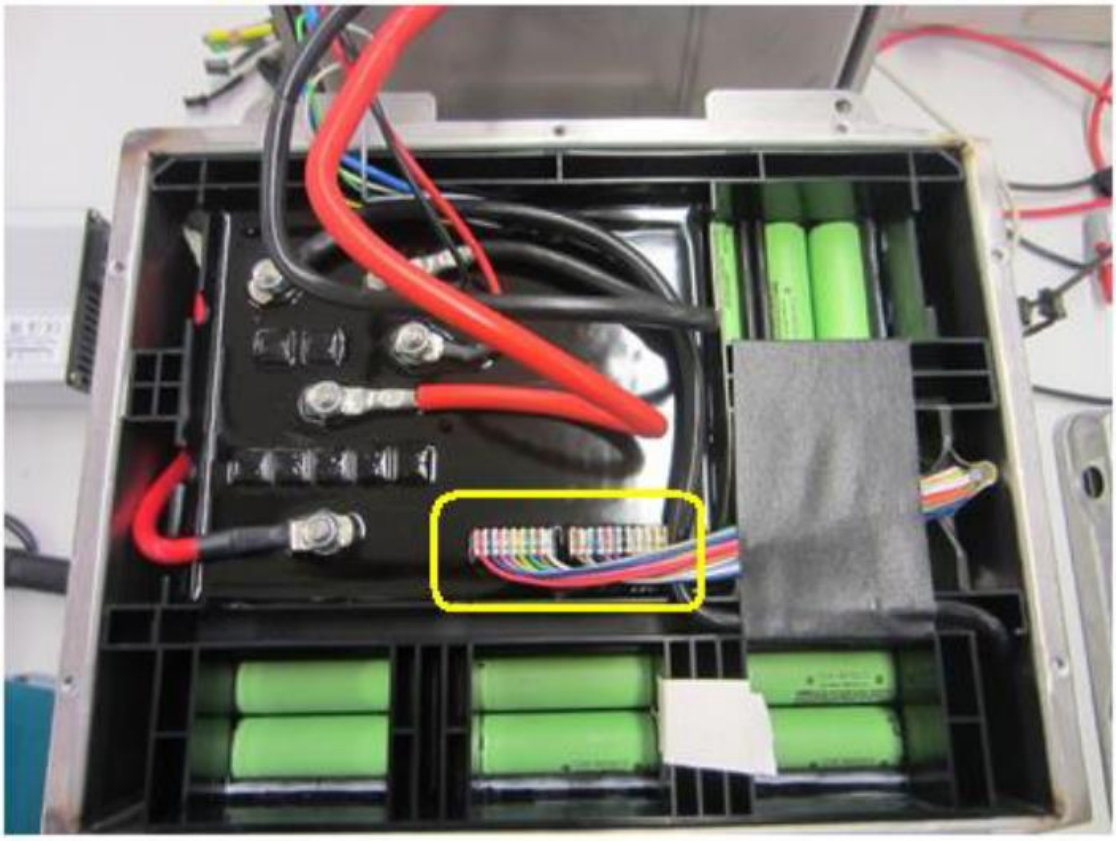

Figura 2. BMS

Dins la placa de resina negra de la que surten els terminal de la bateria, es on s'allotja el BMS. Marcat amb un requadre groc a la figura 2 podem apreciar les sondes de voltatge de cada cel·la, així com les dues sondes de temperatura que te la bateria.

3.1. Cel·les

La bateria esta formada per cel·les de iò de liti recarregables. Cada cel·la te un voltatge nominal de 3.6V i un voltatge màxim de 4.2V. La bateria esta formada per un paquet de tretze cel·les en sèrie que donen el voltatge nominal de 46.8V, i un voltatge màxim de 54.6V.

Figura 3. Pack de cel·les

Per donar més capacitat a la bateria el fabricant ha optat per posar un altre pack de catorze cel·les mes en paral·lel. Això fa que la capacitat d'emmagatzematge augmenti sense augmentar el voltatge de treball. Cada cel·la té una capacitat nominal de 2250mAh i el fabricant assegura una capacitat minia de 2150mAh. Les cel·les tenen garantida una capacitat del 80% després de 300 cicles de càrrega.

3.2. Carregador

Juntament amb la bateria s'incorpora un carregador extern. Aquest carregador es connecta directament a la xarxa i rectifica el corrent altern en continu. Un cop rectificat el corrent el carregador transforma els 230V al nivell al que treballa el BMS, ja que el carregador es connecta directament contra el BMS de la bateria.

4. SISTEMA MOTRIU

El motor que proporciona l'organització es un motor sense escombretes de corrent altern i imants permanents. El motor es trifàsic i es munta acoblat a la roda, es a dir, que el motor substitueix la boixa de la roda on es munta. Te una potencia de 1.75kW, que equival a uns 2.38CV i te un parell de 28N·m.

El sistema motriu que s'ha proporcionat te una particularitat, i es que al anar muntat sobre la roda el rotor i l'estator estant invertits. Això permet que la roda giri tot i actuar a la vega de motor i de boixa de la roda. Normalment els motors elèctrics tenen el debanat del rotor en l'eix, mentre que l'estator se sol muntar a la carcassa. En la figura 3 ho veiem explicat gràficament.

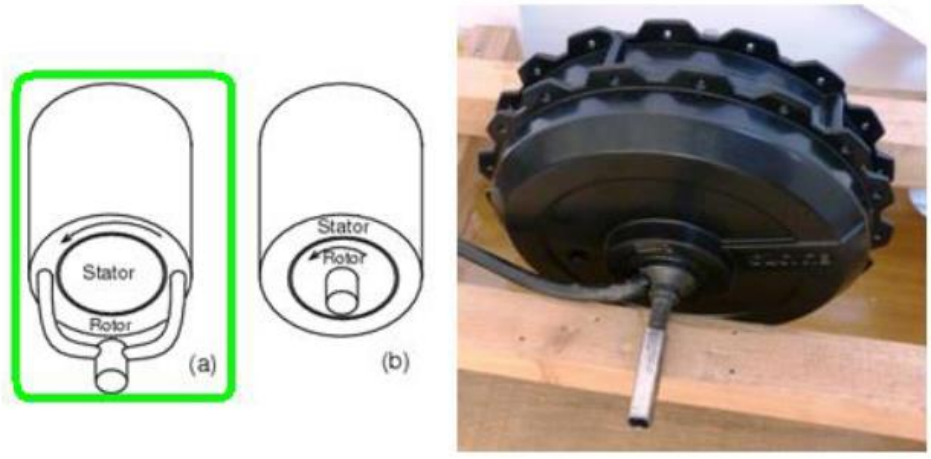

Figura 4. Sentit de gir del motor brushless

.El fet que la carcassa del motor sigui el que rodi i que l'eix quedi fixe ens elimina la necessitat de fer servir sistemes de transmissió, ja siguin cadenes o corretges. Els motors sense escombretes presenten mes bon rendiment que els motors amb escombretes, però la seva electrònica es mes complicada i per tant menys robusta. Com s'aprecia a la Figura... el motor que cedeix la companyia ELMOTO, ve segellat amb una carcassa que e protegeix de la pols i de l'aigua. Tot i que el fabricant no ho especifica, el motor te una protecció equivalent a un nivell de IP 66.

5. UNITAT DE CONTROL ELECTRÒNICA

La unitat de control electrònica, també coneguda com a ECU per les seves sigles en anglés, es la placa electrònica que s'encarrega de gestionar tot el material que proporciona la competició. La ECU te dues funcions clarament diferenciades. D'una banda s'encarrega de gestionar tots els senyals que provenen de les pinyes i els indicadors del manillar, així com de l'indicador de velocitat i de carrega de la bateria. Per altra banda també s'encarrega de la part de control del motor.

Figura 5. ECU

De color vermell a la figura 5 podem apreciar els connectors de potència. Aquí connectem el positiu i el negatiu de la bateria, així com les tres fases que surten cap al motor. El requadre blau remarca el connector de control principal. Aquest connector s'utilitza per a totes les entrades i sortides que no són de potència, com ara les llums, els frens, comunicacions, etc. En color turquesa podem apreciar el connector de convertidor DC/DC. El requadre groc és una sonda Hall que s'utilitza per mesurar el corrent que surt de la placa cap al motor. En verd es pot apreciar una sèrie de condensadors que s'utilitzen per a injectar pics de corrent en cas que la bateria no pugui fer-ho. Aquests condensadors també s'utilitzen per a absorbir les puntes de corrent que el motor entrega al passar a funcionar com a generador. En porpra veiem una de les parts mes importants, el pont trifàsic de MOSFET. I per últim en taronja

veiem el DSP, que es qui s'encarrega de interpretar les entrades de tot el sistema i pren les decisions corresponents. Un cop fet això genera les sortides que van cap al pont trifàsic i cap a la resta de la moto.

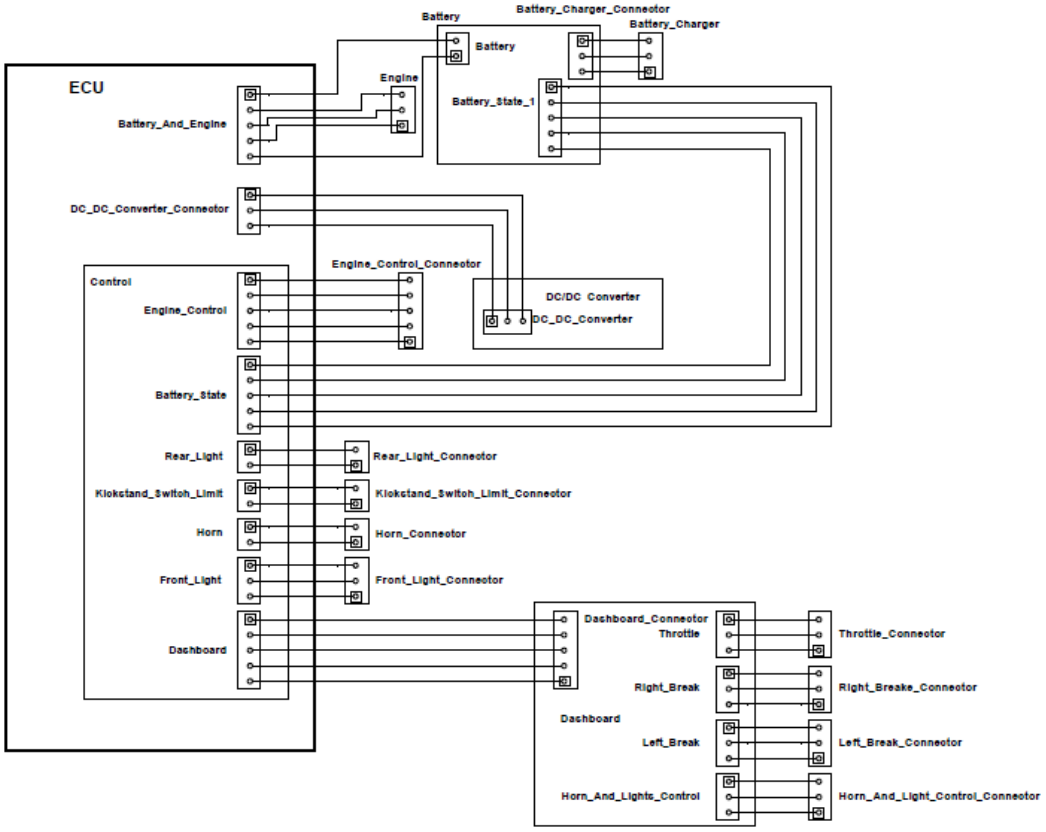

Figura 6. Esquema bàsic de ELMOTO

La normativa de la competició obliga a utilitzar aquesta ECU, però no es proporciona cap mitja per poder llegir les comunicacions amb la resta de components. Tampoc es proporciona cap eina per poder modificar el mapa de potència del motor.

La controladora tendeix a escalfar-se degut a l'alt corrent que hi circula. La ECU te una sonda de temperatura interna que deshabilita la tracció si se sobrepassa una temperatura de seguretat. Per a dissipar aquesta calor i impedir que la ECU es bloquegi, s'utilitza la pròpia carcassa d'alumini com a radiador. En fer proves de circulació ens hem adonat que aquesta dissipació a traves de la carcassa no és suficient. Per a solucionar aquest fet s'ha instal·lat un radiador d'alumini on s'hi acobla la controladora. Aquest radiador queda situat amb la part radiant a l'exterior, on el propi aire que es mou al circular dissipa la calor per convecció.

5.1. Sistema de control dels indicadors

Com ja hem comentat una de les funcions de la ECU és el de processar tot el sistema de llums i seguretats i enviar informació al panell indicador principal. Aquest indicador ens mostra la velocitat instantània, els quilometres fets i el nivell de la bateria desglossat en franges del deu per cent. S'ha de tenir en compte que encara que l'indicador ens marqui que el nivell de la bateria es troba al zero per cent, això no vol dir que la bateria estigui completament descarregada. Com ja s'ha comentat el BMS sempre manté un nivell de voltatge mínim a les cel·les per tal que aquestes no es facin malbé.

Aquesta part de la ECU també s'encarrega dels sistemes de seguretat i de les llums. La deshabilitacio del puny de gas al frenar i al tenir el cavallet desplegat ja es troben implementades en aquesta placa. El sistema de llums en canvi no compleix cap dels requisits demanats per la competició o per les normatives europees. Al engegar la moto, la ECU no engega ni les llums de posició davanteres ni les posteriors. Tampoc te implementada la llum de frenada ni els intermitents.

5.3. Sistema de control del motor

Aquesta es la funció principal de la ECU, que es la de donar tracció al motor de forma proporcional a la consigna que estableix el gas. El DSP intern es qui s'encarrega de governar un pont trifàsic de MOSFET de potencia. Aquest pont s'encarrega de passar el corrent continu de la bateria, a un corrent altern a cada una de les fases del motor. Com que el motor es de tipus brushless, el control del pont trifàsic es fa en funció de a velocitat a la que gira el motor.

6. PANELLS INDICADORS I COMANDAMENTS

En aquest apartat es detallen la resta de components que conformen el material de ELMOTO. Com que ja hem explicat el funcionament bàsic del sistema de bateries, la ECU i el motor, ja tant sols ens queda explicar el funcionament de les pinyes del manillar i el tablier principal.

Figura 7. Indicador principal

En la figura 7 podem veure l'indicador principal que proporciona ELMOTO

6.1. Indicador principal

L'indicador principal es qui s'encarrega de mostrar la informació fonamental per a que el pilot pugi conduir. S'hi mostra la velocitat instantània a que circula la moto, tot i que aquesta informació no es correcta en el nostre prototip. Això es deu a que la velocitat esta calculada per a un diàmetre de roda diferent al que utilitzem nosaltres. Aquest càlcul es fa la ECU i aquesta no es pot reprogramar. Així doncs tant la velocitat com el quilometratge mostrat no son correctes. El panell també mostra l'estat de la bateria, que es mostra com una columna de deu sectors, i si les llums llargues estant activades o no.

6.2. Botoneres

La organització també proporciona les botoneres que comanden les llums i la botzina. Aquesta botonera es un model molt senzill amb dos polsadors. Aquests comandaments funcionen amb senyal de 5V proporcionat per la ECU.

Com que aquest comandament es massa senzill per a poder implementar el sistema de llums i intermitències demanats, s'ha optat per utilitzar una altra botonera.

Aquesta pinya te moltes mes funcionalitats i ens permet incorporar els intermitents i el boto d'emergència. També manté les funcions de la botonera original que eren la de seleccionar el mode de llums i el boto per al clàxon.

Figura 8. Botonera

El boto d'emergència de la botonera esta cablejat al connector de dades de la bateria. En activar l'emergència el que fem es obrir una realimentació del connector que

permet al BMS saber si el connector de dades esta connectat o no. En cas que el BMS no detecti la comunicació amb la ECU, obre el relé principal de potència. En obrir el relé assegurem que la moto quedarà apagada i que no circularà corrent per cap sistema de prototip.

7. UNITAT DE CONTROL DEL VEHICLE

Com ja s'ha explicat la organització demana que el prototip tingui una sèrie de instrumentació addicional que el material base no porta. Aquesta instrumentació necessita d'un processador que adquireixi els senyals i els interpreti. Aquesta funció estaria destinada a la ECU que proporciona la competició, que es qui s'encarrega dels senyals dels comandaments i del panell indicador. El problema que ens trobem es que la competició no permet que es manipulin les funcions que la controladora ja te programades.

Això ens obliga a afegir un altre microprocessador per gestionar tota la instrumentació addicional. Per a fer aquesta tasca hem escollit una placa de microprocessador prefabricada Arduino UNO. S'ha escollit per el seu baix cost i per ser un dels microcontroladors mes emprats. L'Arduino ja inclou el processador, una font d'alimentació estable i el circuit necessari per a establi una connexió amb el PC via USB.

7.1. Arduino UNO

Tot seguit mostrem les principals característiques del microprocessador. L'Arduino Uno utilitza un chip ATMega 328 de la marca Atmel.

Processador	ATMega 328
SRAM	2 kB
EEPROM	1 kB
FLASH	32 kB
Tensió d'alimentació	7 a 12 Vdc
Tensió de treball	5 Vdc
E/S Digitals	14
Entrades Analògiques	6
Corrent màxim per pin digital	40 mA
Corrent màxim per font de 3.3V	50 mA
Freqüència de rellotge	16 MHz

Figura 9. Característiques Arduino

L'Arduino es pot alimentar a través del port USB o mitjançant una font externa. L'origen de l'alimentació es selecciona de forma automàtica així que quan es connecta el port USB l'alimentació s'obté sempre d'aquest.

Si es vol alimentar mitjançant una font externa (ja sigui una bateria o un adaptador de xarxa) es pot connectar indiferentment pel connector de 2.1 mm o pel corresponent pin Vin. Per tal de poder alimentar circuits externs l'Arduino disposa de pins de sortida que proporcionen 5 Vdc i 3.3 Vdc.

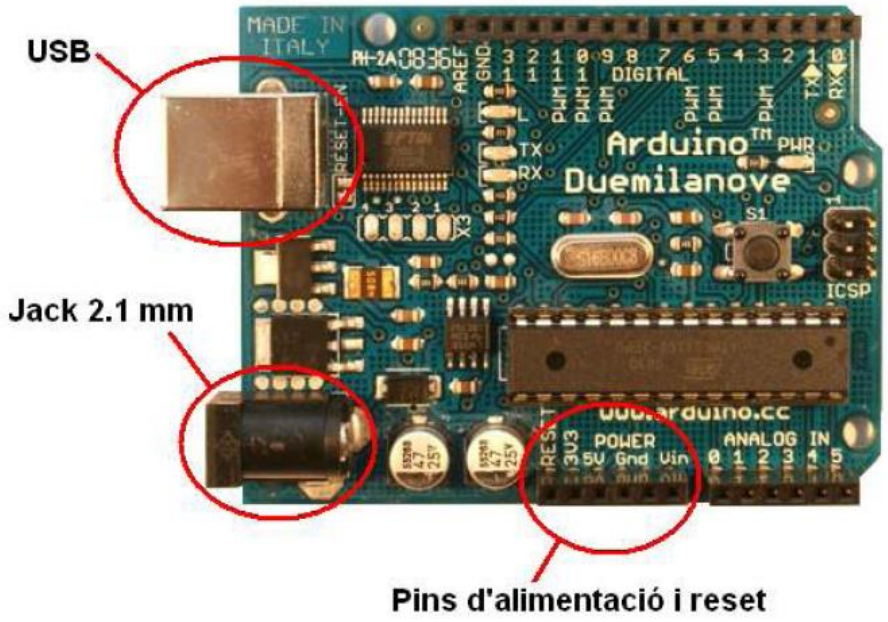

Figura 10. Alimentacions Arduino

En el nostre projecte alimentem l'Arduino a 12V a traves del pin jack de 2.1mm. Aquest voltatge a la seva vegada procedeix del convertidor DC/DC que alimenta els integrats de la ECU i totes les llums.

7.1.1. Entrades i sortides

És molt important definir quines entrades i sortides es faran servir i de quina manera ja que Arduino te pins que poden realitzar funcions especials. Tot seguit es fa una llista de les funcions especials dels pins.

Comunicació sèrie: 0 (Rx i 1(Tx). Mitjançant aquests pins es pot establir una comunicació sèrie amb un ordinador, un altre Arduino o qualsevol altre dispositiu amb port sèrie per tal de transmetre dades.

Interrupcions externes: 2 i 3. Si configurem aquests pins com entrades digitals un canvi d'estat en aquestes entrades pot generar una interrupció en el programa del processador. Les interrupcions en un programa permeten respondre de forma excepcionalment ràpida al canvi d'estat ja que es deixa d'executar el programa principal per anar a executar una funció especialitzada.

PWM: 3, 5, 6, 9, 10 i 11. Aquests pins son capaços de subministrar un tren de polsos d'ample modulat. Aquesta funcionalitat es fa servir per controlar servomotors, la intensitat lluminosa de leds, la velocitat de motors de corrent continu, etc.

SPI: 10 (SS), 11 (MOSI), 12 (MISO) i 13 (SCK). Mitjançant aquests quatre pins es pot crear un bus SPI. Actualment el llenguatge d'Arduino no permet configurar un bus SPI per la qual cosa cal fer-ho en un llenguatge de més baix nivell.

LED: 13. El pin digital 13 està connectat a un led de la placa de forma que si es configura aquest pin com a sortida es pot controlar el funcionament del led.

I2C: 4 (SDA) i 5 (SCL). Permet crear un bus de comunicacions I2C mitjançant aquests dos pins. El bus I2C es fa servir típicament per comunicar amb memòries i altres microcontroladors.

En aquest projecte s'utilitzen els pins de comunicació sèrie per a enviar dades al mòdul de Bluetooth. Els pins d'interrupció son utilitzats per el mòdul Bluetooth i per a llegir el sensor de velocitat.

7.2. Màquina d'estats

En aquest apartat expliquem la maquina d'estat que hem configurat amb Arduino. En implementar-la aconseguim assegurar un nivell mes alt de seguretat per a pilot, ja que bàsicament impedeix que aquest pugui engegar la moto amb el gas donat.

La màquina d'estats te tres estats. En el primer activem la botzina durant un segon. Això és un requeriment que demana la competició perquè la gent sàpiga que la moto s'ha engegat. Per a engegar la botzina el que fem es posar a nivell alt una sortida digital que ataca a un transistor NPN. Aquest transistor NPN esta en paral·lel amb el botó de la botzina. En activar-se el transistor generem un senyal com is estiguéssim prement el boto de la botzina.

```
void loop(){

  switch (estat){
    case 0:
    digitalWrite(D3,HIGH);//Activem la botzina
    digitalWrite(D5,LOW);//Assegurem que el pin esta en mode baix
    delay(1000);//aixo fara que la botzina soni durant 1s
    estat=1;
    break;
```

Tot i que utilitzem un delay, aquest no interfereix amb les interrupcions del sensor de velocitat, ja que en aquest estat la moto es troba quieta. El segon estat és un estat d'espera del que tant sols es surt si l'usuari activa la maneta de fre. S'ha decidit utilitzar la maneta de fre perquè la ECU d'ELMOTO desactiva la funció de gas si es prem la maneta de fre, fent la maniobra d'engegada mes segura.

```
case 1://Aqui posem la moto com a engegada i saltem d'estat si toquem el fre
    digitalWrite(D3,LOW);
    digitalWrite(D5,HIGH);
    on=digitalRead(D6);
    if(on=0){
      estat=2;
    }
    break;
```

Això ens porta a l'últim estat o estat de conducció. Aquest és l'estat de funcionament normal. En aquest estat es on l'Arduino rep i envia senyals a traves de Bluetooth i llegeix els sensors. D'aquest estat cal destacar l'activació de la tracció.

```
case 2:
    digitalWrite(D4,HIGH);
    traccio=1;
```

Semblant al cas de la botzina, l'Arduino posa a nivell alt una sortida digital. Aquest assortida fa conduir un transistor NPN que permet que s'enviï la senyal del gas a la ECU.

8. SENSOR DE VELOCITAT

Una de les dades fonamentals per al pilot a l'hora de conduir es la velocitat. En les motos elèctriques el fet de mostrar aquesta dada encara es mes important, ja que el pilot no es pot guiar pel soroll del motor.

En el nostre cas, el motor ja porta incorporat un sensor de velocitat. Aquest sensor es troba en el motor perquè es imprescindible per al seu control. Com ja hem explicat, el control del motor brushless necessita saber a quina velocitat gira el rotor per poder commutar les polaritats ens consonància.

El tractament de la informació el fa la ECU, que fa passa la velocitat angular a lineal per tot seguit mostrar-ho per l'indicador. La ECU fa la conversió de velocitat lineal utilitzat el radi de roda que fan servir utilitza ELMOTO en els seus models.

El problema es que nosaltres utilitzem una mida de roda diferent que la dels ciclomotors de ELMOTO, i com que no podem accedir a la ECU no podem canviar la conversió de la velocitat. Aquest fet ens obliga a posar un sensor de velocitat extern que puguem llegir. S'ha optat per fer servir un model molt utilitzat el sector industrial i de l'automoció, el sensor de velocitat d'efecte Hall.

Figura 11. Sensor de velocitat

Si en una sonda d'efecte Hall s'aplica corrent i s'aproxima a un camp magnètic, el sensor crea un voltatge a la sortida. Per a que el camp pugi ser detectat, el sensor s'ha de col·locar en una posició on el camp magnètic sigui vertical al sensor. Nosaltres hem

col·locat el sensor a la roda del darrere. Hem escollit la roda posterior perquè es la roda tractora.

El sensor esta col·locat de tal manera que detecta els punts de fixació del disc de fre. Com que tenim trens punts de fixació el sensor ens donarà tres polsos per roda. S'ha de tenir molt en compte que els cargols amb que es fixa el disc de fre han de ser ferromagnètics, ja que sinó ho son el sensor detectarà sis polsos enlloc de tres.

8.2. Comprovació de requisits

Com ja hem explicat, el sensor de velocitat ens dona un pols cada vegada que detecta alguna el camp magnètic provocat per un objecte metàl·lic. Així doncs el sistema d'adquisició ha d'estar preparat per a obtenir aquests polsos. El primer que farem serà calcular la freqüència màxima del senyal provocat per la velocitat.

Per a poder fer el càlcul de la freqüència màxima, primer de tot necessitem saber quina serà la velocitat màxima a la que circularà el vehicle. En el nostre cas la ECU impedeix que el prototip circuli a mes de 45 km/h. Sabem que la roda de darrere, que es la roda tractora i per tant la roda on mesurarem la velocitat, fa 16" de diàmetre.

$$V = w \cdot r \qquad \text{(Eq.1)}$$

Fem la conversió de polzades a metres i utilitzem l'equació X per a obtenir la velocitat angular. Ara ja sabem el nombre de voltes que fa la roda en un segon. Com que el sensor detecta els tres punts de fixació del disc de fre, ens donarà tres polsos per volta. Així doncs multipliquem per tres les voltes per segon i obtenim la freqüència màxima del senyal.

Hem vist en el punt anterior que la freqüència màxima a la que pot treballar el sensor de velocitat es de 8000Hz. Comprovem que el sensor es suficient.

8.3. Software d'adquisició

Arduino processa el codi executant-lo de dalt cap a baix cada vegada. Tot i fer-ho a una velocitat molt alta, això pot fer que perdem fiabilitat del senyal de velocitat, ja que es pot donar el cas que es generi un pols i l'Arduino no estigui escoltant. Per impedir que això passi hem utilitzat interrupcions de codi.

Les interrupcions de codi son entrades que prenent prioritat sobre la resta del codi i forcen al microprocessador a executar una ordre concreta. Bàsicament quan es detecta una interrupció el codi s'atura i s'executa una acció predeterminada. Quan l'acció s'ha executat, el codi torna a córrer des del punt on es trobava abans de la interrupció. Com podem veure això es molt útil en un cas com la velocitat, en que no podem perdre cap pols del senyal.

A l'hora d'implementar interrupcions però s'han de tenir en compte dues coses. La primera es que les accions que s'executin dins de la interrupció s'han de limitar el màxim possible. Això es aconsellable perquè en una interrupció el micro deixa de comptar els polsos de rellotge, i si es passa molta estona en la interrupció podem tenir problemes. La segona cosa a considerar, es que les interrupcions son incompatibles amb els delays. Durant el temps que dura un delay, el microcontrolador lo escolta cap entrada i això inclou les entrades de interrupció. Això provoca que si hi hagués un pols mentre el codi es troba aturat en un delay, la interrupció no s'executaria. Tot seguit mostrem el codi de Arduino per a configurar una interrupció.

```
attachInterrupt(1,velocitat, RISING);//Configurem interrupcio
```

El primer valor de la funció *attatchInterrupt* fa referencia al pin de l'Arduino que provoca la interrupció. El 0 equival al pin D2, mentre que un 1 habilita el pin D3. El segon valor indica quina funció es crida amb la interrupció. Per ultim el tercer valor indica com ha de ser el senyal que dispari la interrupció. Tenim quatre opcions de disparament: *RISING, FALLING, HIGH, LOW*. La opció *Rising* indica que la interrupció es dispara per flanc de pujada, mentre que la funció *Falling* ho fa per flanc de baixada. Per altre banda la opció *High* dispara quan l'entrada esta en estat alt mentre que *Low* la dispara en estat baix.

Ara ja tant sols ens queda programar l'acció que s'executarà durant la interrupció. El que necessitem es un codi que ens permeti saber quan triga la roda en fer una volta. Un cop sabem això, a traves de factors de conversió podem calcular la velocitat lineal. Per fer això tenim dues maneres d'enfocar el codi.

La primera manera es fer que la interrupció compti els polsos i que al arribar a tres compari el temps que s'ha trigat en adquirir aquests polsos. Això ens donarà els

segons que triga en fer una volta. La segona manera es calcular el temps que passa entre cada pols i fer el càlcul de velocitat. Un cop fet això es divideix la velocitat final entre tres per compensar el fet que nomes mirem un sol pols.

Nosaltres ens hem decantat per la segons opció, ja que ens estalviar d'implementar un comptador i d'haver de posar un condicional dins de la interrupció. Això farà que la interrupció sigui mes ràpida.

```
void velocitat()
{
  temps=millis()-T;
  vel=1536/temps;
  T=millis();
}
```

Per acabar guardem el valor en una variable entera per no tenir decimals i ja tenim la velocitat instantània de la moto. Aquest valor serà enviat a traves del mòdul Bluetooth al dispositiu mòbil.

9. SENSOR D'ESTAT DE L'ASFALT

Com a novetat en l'edició 2015 es demana que el prototip tingui un sistema per a poder determinar l'estat de l'asfalt. Aquesta informació s'ha de sincronitzar amb la posició GPS. Això ha de permetre de marcar sobre el mapa l'estat de l'asfalt en la ruta que fa la moto.

Per a determinar l'estat de la carretera tenim dues maneres bàsiques. La primera es fer servir un sistema que detecti la distancia entre la moto i el terra. Això ens hauria de permetre de detectar els forats que hi pugui haver a la carretera. La segona manera es utilitzant un sensor per mesurar la vibració de prototip al circular.

Com que no l'Arduino tant sols pot utilitzar dos pins per a comunicació sèrie, no hem pogut instal·lar un mòdul de GPS. EL que hem fet per a compensar això ha estat fer que la aplicació mòbil agafi la posició GPS del mòbil cada vegada que rep dades del prototip.

Per a mesurar doncs l'estat de l'asfalt hem decidit de fer servir un acceleròmetre de tres eixos. Amb l'acceleròmetre podem mesurar l'acceleració a la moto provocada per els sotracs de l'asfalt.

9.2. Acceleròmetre

Hem utilitzat un acceleròmetre ADXL335. Aquest sensor esta format per una placa de material dielèctric microscòpica suspesa per molles sofre una sèrie de superfícies de silici, creant així un condensador diferencial. En sentir un acceleració la placa es mou i fa variar el voltatge que hi ha als borns d'aquest condensador. Aquest condensador esta format per tres plaques diferents. Això es el que ens permet traduir les variacions d'acceleració en variacions de voltatge per als tres eixos

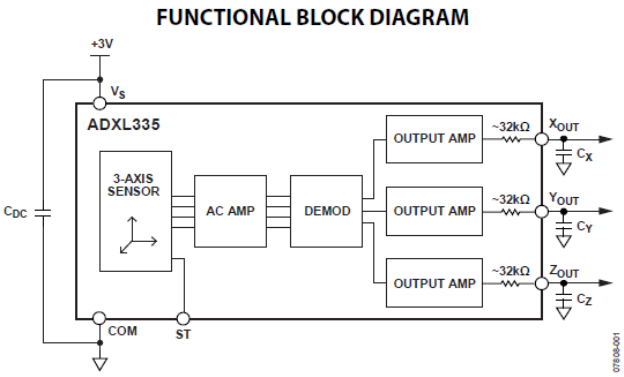

Figura 12. Components que formen l'ADXL335

Aquest sensor mesurar l'acceleració en els tres eixos, tot i que es menys sensible en el seu eix Z. Aquest sensor s'alimenta a 5V i te una sensitivitat de 330mV/g. Com veiem a la figura X a casa sortida trobem un amplificador. Aquest amplificador provoquen el sensor proporciono un senyal de 0V a -3G i un senyal de 3.3V a 3G.

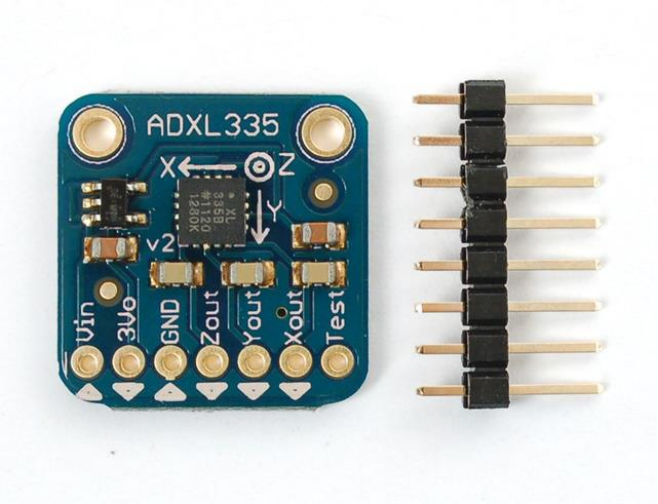

Figura 13. Placa ADXL335

Cada eix incorpora un petit filtre de pas baix. Podem regular la freqüència de tall dels filtres de cada eix acoblant un condensador. Nosaltres hem escollit col·locar el condensador que recomana el fabricant, que es de 10nF. Tot seguit veiem una taula on es mostra la relació entre els condensador i les freqüències del filtre.

Bandwidth (Hz)	Capacitor (µF)
1	4.7
10	0.47
50	0.10
100	0.05
200	0.027
500	0.01

Figura 14. Relació condensadors bandwidth

Tot i que alimentem el circuit a 5V, l'integrat treballa a 3,3V. El circuit porta incorporat un regulador que alimenta l'integrat, cosa que fa mes senzill la seva implementació en Arduino.

9.3. Adquisició i calibratge de l'ADXL335

Ara que ja tenim el mòdul connectat utilitzem l'Arduino per a adquirir el senyal que produeix aquest sensor. Com ja hem dit l'acceleròmetre dona un rang de valors entre 0 i 3,3V. Aquest senyal es proporcional a l'acceleració de +/-3G que el sensor es capaç de detectar. Aquesta entrada no ens bona per a enviar directament al mòbil, ja que ens es una mica difícil de interpretar. Per a això hem de mapejar el senyal d'entrada a l'Arduino per tenir un valor que el dispositiu mòbil pugi interpretar directament.

El primer pas es declarar es variables que farem servir. Haurem d'utilitzar dues variables, ja que una serà la variable bruta sense modificar i l'altre agafarà el valor ja re escalat.

```
int xRaw = 0;
int yRaw = 0;
int brake, accel;
```

Un cop tenim les variables definim els pins per on llegirem el senyal.

```
const int xInput = A1;
const int yInput = A2;
pinMode(xInput,INPUT);//Input Accel eix X
pinMode(yInput,INPUT);//Input Accel eix Y
```

Ara que ja tenim les entrades definides passem a adquirir les dades i a explicar com fem el nou escalat de les variables. Per a llegir utilitzem a funció *analogRead()* que s'encarrega de llegir el valor del pin analògic indicat.

```
bateria=analogRead(A0);
xRaw=analogRead(A1);
yRaw=analogRead(A2);
bat=map(bateria, 2, 5, 0, 100);
brake=map(xRaw, 0, 3.3, -3, 3);
accel=map(xRaw, 0, 3.3, -3, 3);
vibracio=map(yRaw, 0, 3.3, -3, 3);
```

Un cop hem llegit el valor de la tensió que tenim a les entrades analògiques, passem a fer servir la funció *map()*. La funció de mapa el que fa es dona una nova escala a una variable. Primer indiquem la variable a la que anem a canviar les escales. En segon lloc indiquem quins son els límits del rang en que treballava anteriorment, mentre que els dos valors finals son els límits del nou rang de treball. Com veiem nosaltres hem escalat les acceleracions perquè ens marquin un rang de +/-3G. Això tant sols es pot utilitzar sempre que la proporció sigui lineal. Amb això ja tenim unes dades que podem ser fàcilment interpretades.

Es pot veure que les variables accel i brake prenen el mateix valor. Aixo es així perquè per a l'aplicacio mòbil e mes pràctic tenir-los ja en variables separades i després eliminar els valors que no interessin. En el cas de accel s'eliminen les puntes negatives i es fa al reves per a brake.

10. SISTEMA DE CONNECTIVITAT

Tal i com es demana a la normativa a la competició, el prototip ha de dur un sistema de connectivitat que ha d'enllaçar el vehicle amb el dispositiu mòbil. També es demana específicament que aquest dispositiu sigui de tecnologia Bluetooth. Si mirem actualment el mercat podem trobar dos versions diferents: la 3.0 i la 4.0. Els dispositius 3.0 son mes senzills de implementar i ja hi ha moltes marques que tenen models expressament pensats per a Arduino. El problema d'aquest dispositius es que son lents a l'hora de connectar-se i la seva connexió no es molt estable. També s'ha de comptar que el seu consum d'energia es mes gran que les versions noves. El Bluetooth 4.0 es mes estable i de mes rapida connexió que l'antiga versió 3.0, a mes de tenir un consum mes baix. El problema es que no tots els dispositius mòbils en porten, ja que es van començar a incorporar a principis del 2014. Per tant hem decidit fer servir el Bluetooth 4.0 per al nostre prototip. S'ha escollit fer servir el dispositiu Bluefruit-LE nRF8001 del fabricant Adafruit

10.1. Funcionament del modul Bluetooth-LE nRF8001

El mòdul de Bluetooth funciona simulant un dispositiu UART i enviant dades ASCII del mòdul al microcontrolador i viceversa. El que fa aquest mòdul es crear una interfície que comunica les dades en ASCII del mòbil amb l'Arduino que treballa amb protocol sèrie. Per a fer això el mòdul transforma els bytes que rep en una única trama de dades en sèrie i reverteix el procés per transformar les dades que arriben en sèrie de l'Arduino a bytes que pugi interpretar el mòbil. Aquest mòdul també incorpora un bit de paritat a l'hora de fer les conversions per assegurar que les dades que envia son correctes. Incorpora els delimitadors de inici i de fi per poder saber quan comença i acaba la trama de dades.

10.2. Característiques

Aquest dispositiu treballa a cinc volts i es comunica amb el microcontrolador a traves de protocol SPI, es a dir utilitzant els dos pins sèrie de l'Arduino. El mòdul te un rang efectiu de deu metres. El mòbil amb que es faci la connexió ha de tenir instal·lada una versió de Android 4.3 o superior, que son les versions compatibles amb Bluetooth 4.0.

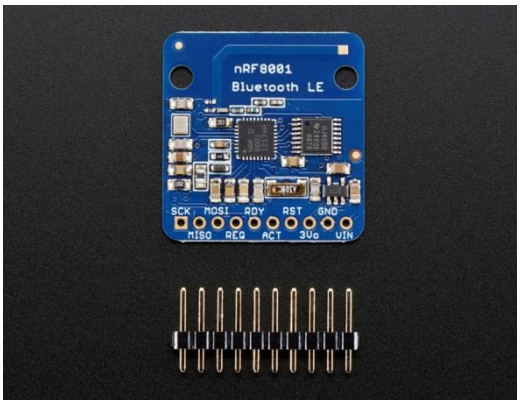

Figura 15. Mòdul Bluetooth-LE nRF8001

En la figura 15 podem veure el dispositiu de Bluetooth i tots els seus pins. A la figura 16 descrivim la funció de cada pin, així com el connexionat amb la placa Arduino.

Pin nRF8001	Descripció	Pin Arduino
Vin	Es el pin d'alimentació del circuit. El circuit s'ha d'alimentar entre 3 i 5V	5V
GND	Es la massa del circuit	GND
3Vo	Es la sortida del regulador intern del circuit que dona 3.3V i fins a 100mA	No es fa servir
RST	Aquest es el pin de l'entrada de reset	Digital 9
ACT	Es una sortida que indica al microcontrolador que el mòdul esta ocupat	No es fa servir
RDY	Pin de Ready. Aquest genera un senyal de interrupció que indica al microcontrolador que el mòdul esta preparat per llegir	Digital 2
REQ		Digital 10
MOSI	Es el pin que rep la informació que li envia el microcontrolador	Digital 11
MISO	Es el pin que envia les dades al microcontrolador. Aquest pin treballa a 3V	Digital 12
SCK	Es el pin de rellotge del circuit	Digital 13

Figura 16. Pinejat nRF8001

La connexió dels pins RST i REQ, es pot canviar a qualsevol altra pin digital que es vulgui, nosaltres hem escollit aquesta configuració perquè era mes còmoda i compacte. La connexió del pin RDY també es pot canviar però s'ha de connectar a un pin de l'Arduino que pugi ser configurat per a interrupcions.

10.3. Software d'enviament de dades

Tot seguit expliquem com hem programat l'Arduino per poder enviar les dades a traves del mòdul Bluetooth. A l'hora de fer la programació ens vam adonar que el mòdul nRF8001 tant sols pot enviar fins a X bytes de informació a la vegada. Això ens ha obligat a fragmentar la informació i fer l'enviament en tres etapes, fent una mica mes complicat el codi. Primer de tot configurem les variables i les entrades.

```
#include <SPI.h>
#include "Adafruit_BLE_UART.h"

// Connect CLK/MISO/MOSI to hardware SPI
// e.g. On UNO & compatible: CLK = 13, MISO = 12, MOSI = 11
#define ADAFRUITBLE_REQ 10
#define ADAFRUITBLE_RDY 2     // This should be an interrupt pin, on Uno thats
#2 or #3
#define ADAFRUITBLE_RST 9

Adafruit_BLE_UART     BTLEserial     =     Adafruit_BLE_UART(ADAFRUITBLE_REQ,
ADAFRUITBLE_RDY, ADAFRUITBLE_RST);

//Arrays for the 4 inputs**********************************************
String sensorValue[9] = {"0","1","2","3","4","5","6","7","8"};
```

Comencem carregant les llibreries de la comunicació sèrie i del mòdul Bluetooth. Tot seguim definim els pins necessaris per al funcionament del mòdul.

Per acabar amb la comunicació creem una variable de tipus *String* que emmagatzemar les dades dels sensor per poder-los enviar. El següent pas es crear les funcions que llegeixen el valor dels sensors i la part de codi que envia les dades des del bucle principal del programa.

```
aci_evt_opcode_t laststatus = ACI_EVT_DISCONNECTED;

void loop()

  // Tell the nRF8001 to do whatever it should be working on.
  BTLEserial.pollACI();
```

```
  // Ask what is our current status
  aci_evt_opcode_t status = BTLEserial.getState();
  // If the status changed....
  if (status != laststatus) {
```

Aqui inicialitzem la comunicació sèrie amb el nRF8001 i posem com a estat del mòdul l'estat de desconnectat. Despres esperem a que hi hagi un canvi d'estat.

```
if (status != laststatus) {
    // print it out!
    if (status == ACI_EVT_DEVICE_STARTED) {
        Serial.println(F("* Advertising started"));
    }
    if (status == ACI_EVT_CONNECTED) {
        Serial.println(F("* Connected!"));
    }
    if (status == ACI_EVT_DISCONNECTED) {
        Serial.println(F("* Disconnected or advertising timed out"));
    }
    // OK set the last status change to this one
    laststatus = status;
  }
```

L'Arduino mira en quin dels tres estat ha canviat el mòdul i ens envia la informació per tal de poder ser llegida per la consola sèrie. Això ens permet saber si el codi entra be a l'estat que li pertoca.

```
if (status == ACI_EVT_CONNECTED) {
    // Lets see if there's any data for us!
    if (BTLEserial.available()) {
      Serial.print("*         ");         Serial.print(BTLEserial.available());
Serial.println(F(" bytes available from BTLE"));
    }
     //Read arduino sensor values
      readSensors();
      //Send values to android
      sendAndroidValues();
    }
 }
```

Si el mòdul es troba en mode connectat, el programa comprova si hi ha informació per enviar i si es així l'envia. Tot seguit executa les funcions que llegeixen i envien les dades dels sensors.

```
void readSensors()
{
    bateria=analogRead(A0);
    xRaw=analogRead(A1);
    yRaw=analogRead(A2);
    bat=map(bateria, 2, 5, 0, 100);
    brake=map(xRaw, 0, 3.3, -3, 3);
    accel=map(xRaw, 0, 3.3, -3, 3);
    vibracio=map(yRaw, 0, 3.3, -3, 3);
```

```
      if(counter == 0)
      {
        //Moto on
        sensorValue[0] = on;
        //Moto off
        sensorValue[1] = on;
        //Moto in traction
        sensorValue[2] = traccio;
        counter = counter + 1;
      }
```

Com que el xip nRF8001 te un enviament de dades limitat, tant sols podem enviar el valor de tres variables a la vegada. Això ens obligar a fer la operació de enviar les dades tres vegades. El comptador es qui s'encarrega de determinar les vegades que em fet l'enviament.

Ara repetim el procés amb la resta de dades dels sensors. Veiem que el que fem es escriure els valors en la mateixa taula i aquesta la enviem cada vegada. Mes endavant l'aplicació mòbil s'encarrega de aglutinar tota la informació en una sola taula.

```
else if(counter == 1)
      {
        //Moto no traction
        sensorValue[0] = "0";
        //Moto vibracy
        sensorValue[1] = "5";
        //Moto battery
        sensorValue[2] = "6";
        //sensorValue[2] = String(analogRead(analogPin));
        counter = counter + 1;

      }
      else
      {
        //moto velocity
        sensorValue[0] = vel;
        //moto acceleration
        sensorValue[1] = A2;
        //moto brake
        sensorValue[2] = A1;
        counter = 0;
      }
}
```

Abans d'enviar el programa converteix les dades dins de la taula en bytes. Aquest procés carrega molt el buffer del mòdul nRF8001 i es el que fa que tant sols puguem enviar tres valors cada vegada.

```
void sendAndroidValues()
```

```
{
    String s = "";
    for(int k=0; k<3; k++)
    {
      s= s + sensorValue[k] + ";";
    }
    s = s + ";";

      // We need to convert the line to bytes
      uint8_t sendbuffer[s.length()];
      s.getBytes(sendbuffer, s.length());
      char sendbuffersize = min(s.length(), s.length());

      // write the data
      BTLEserial.write(sendbuffer, sendbuffersize);
}
```

11. SISTEMA DE LLUMS

Com ja s'ha comentat anteriorment el sistema de llums original de la motocicleta no compleix totes les especificacions necessàries per a la competició. L'electrònica per a solucionar aquests problemes s'ha dissenyat de tal manera que no necessiti passar per la VCU. Això s'ha fet per assegurar que en cas de fallada del microcontrolador, el sistema de llums sempre funcionarà i per tant el ciclomotor es podrà conduir encara que els sistemes intel·ligents no funcionin. En el cas de les llums de posició, com que sempre han d'estar connectats, s'ha escollit fer arribar una línia de dotze volts directe del DC/DC tant a la llum posterior com a la davantera.

11.1. Llum de frenada

La llum de fre consta de dues línies de LED diferenciades però amb la mateixa massa. Una de les línies de LED sempre esta alimentada amb una línia que ve directe del convertidor, i que fa de llum de posició posterior. Per tal de solucionar el problema de la llum de frenada, s'ha decidit de crear un petit sistema que interactuï amb els senyal de les manetes de fre i el convertidor DC/DC. El senyals del sensor de fre treballa a cinc volts. El sensor de fre es un final de cursa normalment obert. Aquest final de cursa esta activat sempre que es toqui la maneta de fre, es a dir, quan anem circulant normalment el sistema veu cinc volts. Quan es prem la maneta de fre el final de cursa passa a esta obert i passem a tenir zero volts. Per aprofitar aquest funcionament, el que s'ha fet es fer servir un transistor de tipus PNP que serveix per comandar un relé que s'enclava i deixa passar els dotze volts a la llum.

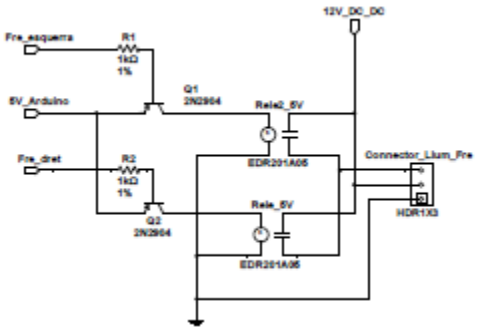

Figura 17. Esquema del circuit llum de frenada

Com es veu a la figura xxx s'ha fet servir un transistor PNP, ja que volem que condueixi quan la base es troba a zero volts. El transistor deixa passar els cinc volts

que venen del convertidor per atacar el relé. El relé s'enclava i deixa passar els dotze volts que venen del convertidor per alimentar els LED.

Una altra solució aparentment senzilla hagués estat fer el control per software des de l'Arduino. Això comportava dos problemes bàsics. El primer problema es que si l'Arduino deixava de funcionar, el sistema de senyalització de frenada també deixava de funcionar. El segon problema es que les sortides digitals de l'Arduino no treuen prou corrent per atacar la bobina del relé.

11.2. Llums d'indicació de gir

Com demana la normativa europea, el ciclomotor ha de tenir un sistema d'indicació de gir per a poder circular per la via publica. Com que la ECU proporcionada a l'equip per la organització no contempla aquesta mena d'indicadors s'ha fet un sistema nou. El reglament espanyol estableix que els indicadors de gir han de ser de color taronja i que han de ser intermitents amb una freqüència de un hertz. Per a assolir aquest objectiu s'ha dissenyat el següent sistema.

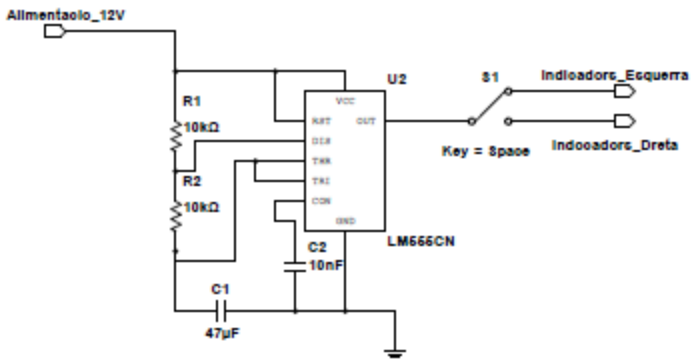

Figura18. Circuit d'intermitència

El primer de tot es aconseguir un senyal amb un intermitència de un hertz i que treballi a dotze volts. S'ha utilitzat un integrat LM555 en mode astable per fer aquest senyal. En el seu mode astable el 555 genera un senyal intermitent amb una freqüència determinada per la velocitat de carrega i descarrega del condensador connectat al pin de trigger.

Per a regular la freqüència i el duty cycle del circuit, haurem de calcular els valors de les resistències i del condensador. Per a això utilitzem la següent equació

$$t = 1.1RC \qquad \text{(Eq.2)}$$

Com que l'equació te dues incògnites, primer de tot escollim un dels components. Escollim el valor del condensador i posem un valor de 10nF. Un cop fet això ja podem trobar la resistència del circuit de carrega. Com que volem que el senyal generat sigui simètric, es a dir que el duty cycle sigui del 50%, haurem d'escollir la segona resistència amb el mateix valor que la calculada.

El LM555 dona un senyal amb una amplitud igual al valor d'alimentació de l'integrat. Nosaltres hem alimentat l'integrat a dotze volts, així que el senyal de sortida que tenim ja te una amplitud suficient per ser enviat als intermitents. La qüestió ara es saber si tindrà suficient intensitat. Com que s'han escollit intermitents de LED, aquests tenen un consum molt baix. El consum d'un sol intermitent es de 0.5W, però sempre alimentem dos llums a la vegada ja que alimentem els dos intermitents be de la banda dreta o be de la banda esquerra. Això ens deixa un consum de 1W. Com que les llums treballen a dotze volts, si apliquem la equació 3 de la potencia veiem que el consum es de 83mA.

$$P = V \cdot I \qquad \text{(Eq.3)}$$

El LM555 pot donar una corrent de sortida de fins a 200mA que es mes que suficient per a les nostres necessitats.

11.2.1. Altres solucions

La majoria dels vehicles utilitzen un relé d'intermitència per crear el senyal d'intermitència. Aquest relé consisteix en fer servir un termoparell. En passar corrent per les lamines del termoparell, aquestes es dilaten. Com que les lamines son de materials diferents, una es dilata mes de l'altra. Això fa que es deformin fins al punt que deixen de fer contacte i el relé s'obre. Quan el termoparell es refreda torna a la posició inicial i torna a fer contacte. Si les lamines estant ben ajustades això produeix un efecte de intermitència de un hertz.

Aquesta solució no s'ha pogut aplicar en el nostre prototip, ja que al utilitzar llums de tecnologia LED, les llums no consumeixen prou intensitat per a escalfar els termoparells i fer que deixin de fer contacte durant el temps desitjat. Amb llums de tecnologia LED els relés d'intermitència augmenten de freqüència, igual que si una de les llums estigues fosa. Es possible evita aquest efecte afegint resistències en sèrie amb les llums. Això fa que les llums consumeixin mes intensitat i per tant fan funcionar el termoparell de la manera adequada. El problema es que es crea un consum de corrent innecessari que elimina la gran avantatge que ens suposa el baix consum dels LED.

12. RESUM DEL PRESSUPOST

El cost d'aquest projecte puja a cinc mil cinc-cents vuitanta-dos euros amb setze cèntims sense I.V.A.

13. CONCLUSIONS

Els dissenys presentats en aquest projecte compleixen amb les especificacions establertes tant per l'equip UdG Racing Team com per l'organització de l'Smart Moto Challenge. Les solucions esposades anteriorment s'han utilitzat per participar a la SMC 2015, obtenint la cinquena millor posició absoluta i la segona millor de l'estat espanyol.

Les seguretats dissenyades compleixen amb escreixos les establertes a la normativa de la competició. La maquina d'estats que s'ha dissenyat per a la unitat de control del vehicle, així com el sensor de porta oberta, ajuden a millorar encara mes aquesta seguretat.

El condicionament dels sensors incorporats a la moto, permet que la VCU pugui adquirir i interpretar correctament els senyals. Aquest senyals son enviat a traves del mòdul de Bluetooth al dispositiu intel·ligent. A la vegada aquest dispositiu envia la informació a una base de dades.

Tot i haver assolit els resultats desitjats, de cares a un futur s'hauria de buscar una manera de transmetre totes les dades dels sensors a traves d'algun protocol de comunicació. Això permetria assegurar la validesa de les dades dels sensors, sobretot d'aquells que es troben lluny del dispositiu d'adquisició. En cas de utilitzar-se el protocol CAN, seria possible accedir a la informació que envien tant la bateria com el motor a la ECU, així com monitoritzar possibles errors dels microcontroladors gracies als timeouts del CAN.

David Abad Ayats

Graduat en Enginyeria Electrònica Industrial i Automàtica

Girona, 4 de setembre de 2015

14. RELACIO DE DOCUMENTS

Aquest projecte consta de cinc documents. Els documents son memòria, plànols, plec de condicions, estat d'amidaments i pressupost. En aquest projecte es considera que la memòria, el plec de condicions, els plànols i l'estat d'amidaments són contractuals. En canvi es considera que el pressupost pot estar subjecte a canvis.

En cas que es trobi una discrepància entre els diferents documents d'aquest projecte, el document que preval sobre els altres es la memòria.

15. BIBLIOGRAFIA

Adaftuit ADXL335, https://learn.adafruit.com/adafruit-analog-accelerometer-breakouts, 15 d'agost de 2015.

Adafruit Bluetooth-LE nRF8001, https://learn.adafruit.com/getting-started-with-the-nrf8001-bluefruit-le-breakout, 10 d'agost 2015.

Arduino, https://www.arduino.cc/, 19 d'agost 2015.

Farnell, http://es.farnell.com/, 17 juny 2015.

Viafi, http://www.viafi.net/ca/cables-accesoris-electrics, 10 juny 2015.

16. GLOSSARI

BMS = Battery Management System

BJT = Bipolar Junction Transistor

VCU = Vehicle Control Unit

ECU = Electronical Control Unit

DSP = Digital Signal Processor

MOSFET = Metal Oxid Semiconductor Field Effect Transistor

SPI = Serial Peripheral Interface

UART = Universal Asynchronous Receiver Transmitter

A. CODI D'ARDUINO

```
#include <SPI.h>
#include "Adafruit_BLE_UART.h"

// Connect CLK/MISO/MOSI to hardware SPI
// e.g. On UNO & compatible: CLK = 13, MISO = 12, MOSI = 11
#define ADAFRUITBLE_REQ 10
#define ADAFRUITBLE_RDY 2     // This should be an interrupt pin, on Uno thats
#2 or #3
#define ADAFRUITBLE_RST 9

Adafruit_BLE_UART     BTLEserial     =     Adafruit_BLE_UART(ADAFRUITBLE_REQ,
ADAFRUITBLE_RDY, ADAFRUITBLE_RST);
const int xInput = A1;
const int yInput = A2;


//Arrays for the 4 inputs*****************************************
String sensorValue[9] = {"0","1","2","3","4","5","6","7","8"};
int counter = 0;
volatile int count;  //Volatil es recomenable per a interrupcions
int n = count ;
int count1=0;
long temps = 0 ;  // Varaible per el temps
int estat, traccio, vel, accel, brake, bat, on, vibracio, bateria;

int xRaw = 0;
int yRaw = 0;
int D3=3;
int D4=4;
int D5=5;
int D6=6;

void setup() {
  pinMode(D3,INPUT);//Habilitem el pin del brunzidor
  pinMode(D4,OUTPUT);//Pin 4 utilitzat per a habilitar traccio
  pinMode(D5,OUTPUT);//Pin 5 marca l'estat de ences o apagat
  pinMode(A0,INPUT);//Input bateria
  pinMode(xInput,INPUT);//Input Accel eix X
  pinMode(yInput,INPUT);//Input Accel eix Y
  pinMode(D6,INPUT);//Entrada del fre
  Serial.begin(9600);
  while(!Serial); // Leonardo/Micro should wait for serial init
  Serial.println(F("Adafruit Bluefruit Low Energy nRF8001 Print echo demo"));
  // BTLEserial.setDeviceName("NEWNAME"); /* 7 characters max! */
  attachInterrupt(1,velocitat, RISING);//Configurem interrupcio

  BTLEserial.begin();
}
/**************************************************************************/
/*!
    Constantly checks for new events on the nRF8001
*/
/**************************************************************************/
aci_evt_opcode_t laststatus = ACI_EVT_DISCONNECTED;

void loop(){

  switch (estat){
    case 0:
    digitalWrite(D3,HIGH);//Activem la botzina
    digitalWrite(D5,LOW);//Assegurem que el pin esta en mode baix
    delay(1000);//aixo fara que la botzina soni durant 1s
    estat=1;
    break;
```

```
    case 1://Aqui posem la moto com a engegada i saltem d'estat si toquem el
fre
    digitalWrite(D3,LOW);
    digitalWrite(D5,HIGH);
    on=digitalRead(D6);
    if(on=0){
      estat=2;
    }
    break;

    case 2:
    digitalWrite(D4,HIGH);
    traccio=1;
    // Tell the nRF8001 to do whatever it should be working on.
  BTLEserial.pollACI();

  // Ask what is our current status
  aci_evt_opcode_t status = BTLEserial.getState();
  // If the status changed....
  if (status != laststatus) {
    // print it out!
    if (status == ACI_EVT_DEVICE_STARTED) {
        Serial.println(F("* Advertising started"));
    }
    if (status == ACI_EVT_CONNECTED) {
        Serial.println(F("* Connected!"));
    }
    if (status == ACI_EVT_DISCONNECTED) {
        Serial.println(F("* Disconnected or advertising timed out"));
    }
    // OK set the last status change to this one
    laststatus = status;
  }

  if (status == ACI_EVT_CONNECTED) {
    // Lets see if there's any data for us!
    if (BTLEserial.available()) {
      Serial.print("*         ");         Serial.print(BTLEserial.available());
Serial.println(F(" bytes available from BTLE"));
    }
      //Read arduino sensor values
      readSensors();
      //Send values to android
      sendAndroidValues();
    }

   }

 }


void readSensors()
{
    bateria=analogRead(A0);
    xRaw=analogRead(A1);
    yRaw=analogRead(A2);
    bat=map(bateria, 2, 5, 0, 100);
    brake=map(xRaw, 0, 3.3, -3, 3);
    accel=map(xRaw, 0, 3.3, -3, 3);
    vibracio=map(yRaw, 0, 3.3, -3, 3);

      if(counter == 0)
      {
        //Moto on
        sensorValue[0] = on;
        //Moto off
        sensorValue[1] = on;
        //Moto in traction
```

```
        sensorValue[2] = traccio;
        counter = counter + 1;
      }
      else if(counter == 1)
      {
        //Moto no traction
        sensorValue[0] = traccio;
        //Moto vibracy
        sensorValue[1] = vibe;
        //Moto battery
        sensorValue[2] = bat;
        counter = counter + 1;

      }
      else
      {
        //moto velocity
        sensorValue[0] = vel;
        //moto acceleration
        sensorValue[1] = accel;
        //moto brake
        sensorValue[2] = brake;
        counter = 0;
      }
}

//sends the values from the sensor over serial to BT module
void sendAndroidValues()
{
    String s = "";
    for(int k=0; k<3; k++)
    {
      s= s + sensorValue[k] + ";";
    }
    s = s + ";";

      // We need to convert the line to bytes
      uint8_t sendbuffer[s.length()];
      s.getBytes(sendbuffer, s.length());
      char sendbuffersize = min(s.length(), s.length());

      // write the data
      BTLEserial.write(sendbuffer, sendbuffersize);
}
void velocitat()
{
  temps=millis()-T;
  vel=1536/temps;
  T=millis();
}
```