REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique

Département d'Electronique

Mémoire de projet de fin d'études

Pour l'obtention du diplôme d'ingénieur d'état en électronique

---

Gestion par ordinateur d'un banc d'accumulateurs

Billel DJELLOULI

Sous la direction de **M. Mourad HADDADI** (Professeur)

Présenté et soutenu publiquement le **22/06/2016**

Composition du Jury :

Président	M. Med Salah AIT CHEIKH	Pr.	ENP
Rapporteur/ Promoteur	M. Mourad HADDADI	Pr.	ENP
Examinateur	M. Hicham BOUSBIA-SALAH	MCA.	ENP

ENP 2016

Remerciements

En premier lieu, je remercie Dieu le tout puissant de m'avoir donné le courage, la volonté et la patience pour réaliser ce travail.

Je remercie mes parents, mes chers frères et sœurs qui m'ont soutenu durant mes études.

Je remercie mes six neveux qui m'ont toujours apporté la joie et la bonne humeur.

Mes sincères gratitudes à M. Mourad Haddadi qui m'a orienté et dirigé durant mon projet.

Je remercie également les membres de jury: M. Med Salah Ait Cheikh Président de jury / Chef du département d'électronique, ainsi que M. Hicham Bousbia-Salah comme examinateur, qui nous ont honorés en accordant du temps et de l'énergie à la lecture de ce document et qualifier notre travail.

Je remercie chaleureusement tous les membres de l'équipe du Laboratoire des Dispositifs de Communication et de Conversion Photovoltaïque qui m'ont éclairé avec leur savoir et leur expérience.

Finalement je remercie tous mes amis qui m'ont soutenu moralement, et toute personne qui m'a aidé de près ou de loin afin d'effectuer ce travail.

ملخص

المشروع الذي قمنا به طيلة الأشهر الأخيرة يتمثل في صنع شاحن لبطاريات الرصاص ذو توتر 12 فولط، و ذلك وفق خوارزمية تنفذها وحدة التحكم و المعالجة (PIC18F2550)، بالإضافة إلى تطوير نظام للتواصل و التحكم عن طريق الكمبيوتر.

يتركز المشروع على أربعة محاور رئيسية هي:

- إنشاء الدارة الكهربائية، و تتألف من محول الطاقة (Buck) مع وحدة التحكم (PWM)، العناصر المسؤولة عن الاستشعار (لقياس التوتر، شدة التيار و درجة الحرارة) و أخيرا شاشة العرض LCD.
- برمجة وحدة التحكم و المعالجة لضمان خوارزمية شحن عبر ثلاث مراحل، الضبط الكامل للدارة الكهربائية، قراءة مختلف البيانات لعرضها في شاشة LCD، وإرسالها في الناقل التسلسلي USB.
- تصميم واجهة المستخدم الرسومية، مشفرة بلغة البرمجة # C، التي تتمثل في واجهة للتواصل بين الدارة الكهربائية و المستخدم. يمكن من خلالها ارسال توجيهات، بالإضافة إلى استقبال البيانات، عرضها و حفظها على جهاز الكمبيوتر.
- إنشاء الاتصال التسلسلي USB بهدف ربط الدارة الكهربائية و واجهة المستخدم الرسومية.

الكلمات المفتاحية: شاحن كهربائي، بطاريات الرصاص، محول Buck ، متحكم، PIC18F2550، MikroC، اتصال تسلسلي USB، واجهة المستخدم الرسومية، لغة # C، Visual Studio.

Abstract

Our project aims to make a charger for lead-acid batteries with a nominal voltage of 12V, using a charging algorithm implemented in a microcontroller (PIC18F2550) and the development of a communication and management system via computer.

The project is based on four main axes:

• The manufacturing of the electrical part (hardware part) consisting of the power supply (manufacturing of Buck converter) and its control with PWM, the elements of data conditioning (used for measuring the voltage, current and temperature) and finally the LCD screen display.

• Programming the microcontroller (firmware part) to ensure a three-stage charging algorithm, controlling the entire hardware part, read different data with its embedded ADC and finally write on the LCD screen and on the USB bus.

• Design of a Graphical User Interface (software part), coded in C #, which makes a connection between the charging circuit and the user. That one can send commands, acquire, display and save data on computer.

• Establishing a USB serial communication linking the charger circuit and the GUI.

Keywords: Charger, Lead Acid Batteries, Buck Converter, Microcontroller, PIC18F2550, MikroC, USB communication, Graphical User Interface, C # language, Visual Studio.

Résumé

Notre projet consiste à réaliser un chargeur pour les batteries plomb – acide ayant une tension nominale de 12V, suivant un algorithme de charge implanté dans un microcontrôleur (PIC18F2550) ainsi que l'élaboration d'un système de communications et de gestion par ordinateur de ce dernier.

Le projet est basé sur quatre axes principaux :

- La réalisation de la partie électrique (partie hardware) qui est constituée de l'alimentation (réalisation d'un hacheur Buck) avec sa commande PWM, les éléments de conditionnement (pour la mesure de la tension, du courant et de la température) et affichage sur écran LCD.
- Programmation du microcontrôleur (partie firmware) afin d'assurer un algorithme de charge à trois phases, commander toute la partie hardware, lire des différents données avec le convertisseur analogique-numérique qu'il possède et enfin écrire sur l'afficheur LCD et sur le bus USB.
- Conception d'une interface graphique (partie software), codée en langage C#, qui fait une liaison entre le circuit de charge et l'utilisateur. Celle-là peut envoyer des commandes, acquérir, afficher et sauvegarder les données sur ordinateur.
- Etablissement d'une communication série USB faisant le lien entre le circuit de charge et l'interface graphique.

Mots clés : Chargeur, Batteries Acide-Plomb, Hacheur Buck, Microcontrôleur, PIC18F2550, MikroC, Communication USB, Interface graphique, Langage C#, Visual Studio.

Tables des matières

Liste des figures

Liste des tableaux

Introduction générale

L'électricité est une source d'énergie à fort consommation. De nos jours, une grande partie de l'activité humaine exige des systèmes de stockage d'énergie. Ces derniers serviront comme un support de sécurité et d'approvisionnement pour garantir une continuité de service des appareils électriques. Ce stockage est assuré par un banc d'accumulateurs connectés en série et/ou en parallèle pour fournir l'énergie nécessaire.

Une des technologies de stockage qui est largement utilisée est la batterie au plomb. La maturité et le faible coût de cette technologie en sont les raisons principales. Cependant, une bonne utilisation nécessite une optimisation de la méthode de charge. Car les performances et la longévité de ces batteries dépendent de la qualité des chargeurs. Pour cela le choix d'un chargeur efficace est indispensable.

Dans ce présent travail, nous allons concevoir un chargeur de batterie au plomb d'une tension nominale de 12V. Ce chargeur sera géré et commandé d'une façon autonome par un microcontrôleur. Une interface graphique sera élaborée, elle servira comme liaison entre l'utilisateur et le circuit de charge.

Chapitre 1

Généralités sur les batteries

1.1. Introduction

Dans ce chapitre nous allons introduire des généralités sur les accumulateurs et les batteries. Nous allons donner la classification des batteries. Leurs grandeurs caractéristiques et leurs phases de fonctionnement seront également mentionnées. Ensuite, nous allons citer les types les plus courants des batteries en s'étalant davantage sur les batteries au plomb. Et enfin, nous décrirons les méthodes de charge pour les batteries au plomb.

1.2. Terminologie

1.2.1. Définition d'un accumulateur

L'accumulateur est un appareil qui emmagasine de l'énergie pour la restituer à mesure des besoins. [1]

1.2.2. Définition d'une batterie

Une batterie est un ensemble d'accumulateurs électriques reliés entre eux de façon à créer un générateur électrique de tension et de capacité désirée. Ces accumulateurs sont parfois appelés éléments de la batterie ou cellule. [2]

1.2.3. Nuance de vocabulaire

Pour les éléments rechargeables on utilise les termes de batteries ou d'accumulateurs, à la différence d'une pile qui n'est pas rechargeable. Cette nuance de vocabulaire peut avoir lieu en français mais beaucoup moins dans la littérature anglo-saxonne, dans laquelle le terme « battery » désigne indifféremment une pile ou un accumulateur. [3]

1.3. Classification des batteries

Les batteries se regroupent sous deux grandes classes : accumulateurs primaires (non-rechargeables) et autres secondaires (rechargeables), on distingue aussi d'autres genres de classifications basées soit sur une structure particulière (conception technologique) ou un domaine d'utilisation bien défini.

1.3.1. Les batteries primaires

Elles sont dans l'incapacité d'être chargées électriquement. On ne les utilise qu'une seule fois, après on devra les changer car les réactions chimiques qui les gouvernent sont irréversibles. Malgré leur prix relativement élevé, les batteries primaires sont très commodes pour certaines applications : lampe à torche, appareillage d'instrumentation, jouets et lanceurs dans le domaine spatial.

Les avantages principaux d'une batterie primaire sont : une densité d'énergie très élevée, une durée de vie appréciable, aucune maintenance à prévoir, et une facilité d'utilisation.

1.3.2. Les batteries secondaires

Une batterie secondaire est un dispositif électrochimique destiné à emmagasiner de l'électricité pour la restituer ensuite à la demande. Ce genre de batterie peut être chargé électriquement une fois déchargé par le passage d'un courant électrique à travers ses électrodes en sens inverse du courant de décharge. C'est ce qui lui donne son aspect de dispositif de stockage d'énergie électrique connue sous le nom « Accumulateur ».

Les accumulateurs utilisés comme un moyen de stockage d'énergie, sont généralement connectés à une source électrique pour être chargés (alternateur, réseau électrique, générateur photovoltaïque… etc.), et une charge qui consomme l'énergie délivrée par l'accumulateur en régime de décharge, exemple : voiture, installations électriques en avion, satellite, alimentation de secours non interruptible (UPS), …. Ils sont également utilisés à la place des batteries primaires en régime de décharge; avec l'avantage d'être rechargeables plusieurs fois (cycles) plutôt que jetables. [4]

1.4. Grandeurs caractéristiques des accumulateurs

La diversité des domaines d'utilisation des batteries a imposé plusieurs types de critères qui sont utiles pour déterminer les performances d'une batterie à travers ces paramètres afin d'en classer les différents types des batteries suivant leur conformité aux exigences.

1.4.1. La tension

La tension est le paramètre le plus apparent et facile à déterminer. Différentes types de tension sont définis:

- **Tension théorique « Eth » :** C'est une tension qui est en fonction des matières actives (Anode, Cathode et électrolyte) et la température. Elle est déduite à partir de la loi de Nernst.

Remarque

La loi de Nernst est une équation qui met en relation la FEM de la batterie ($E_{éq}^{Cell}$) avec la différence de potentiel standard pour un couple d'électrode (E^0), la température, la constante des gaz parfait (R), la constante de Faraday (F) et les différentes concentrations des différents éléments participant dans la réaction d'oxydo-Réduction.

$$E_{éq}^{(Cell)} = E^0 - \frac{RT}{nF} Ln\left(\left[\frac{[Réd_1]^c[Ox_2]^d}{[Ox_1]^a[Réd_2]^b}\right]\right)$$

- **Tension nominale, « Vn » :** (Nominal voltage) c'est la valeur par laquelle les batteries sont démarquées. C'est l'une des tensions typiques, recommandée en mode de fonctionnement. C'est en général la tension selon laquelle une batterie a été nommée ou est généralement en fait référence. Pour une cellule au plomb-acide, la tension nominale est de 2.0V (Qui peut être différente de la tension en circuit ouvert). [4,5]

- **Tension de fin de décharge, «VCut-Off » :** (Cut-off voltage) quand la cellule (ou batterie) atteint cette tension elle a donc délivrée la majorité de sa capacité, elle est considérée comme vide. Le fabriquant estime qu'opérer en dessous de cette tension dégrade la batterie car on est en phase de sous décharge qui implique des réactions irréversibles. La tension de fin de décharge pour une cellule « LA » (Acide-Plomb) et «Li-ion» est de 1.75Vet 2.5V respectivement.

- **Tension de fin de charge, «Vfull » :** lorsque la cellule (ou batterie) atteint cette tension, elle récupère toute la matière active disponible et elle est considérée comme chargée. Des cellules « LA » et «Li-ion» sont pleinement chargées à 2.15V [6] et 4.2V respectivement.

- **Tension à circuit ouvert « VOC »** la tension mesurée de la cellule (ou batterie) sous aucune charge (courant nul), cette tension est directement liée à l'état de charge de la cellule (SOC) qui est à son tour liée à la force électromotrice. La VOC ne coïncide pas instantanément avec force électromotrice, pour avoir un rapport fidèle il va

falloir ajouter un temps de relaxation pour la cellule après la charge et/ou décharge. Ce temps dépend de la température, du courant avant le repos. [4]

1.4.2. La capacité

La capacité est la caractéristique principale d'un accumulateur, c'est la charge électrique que peut fournir l'accumulateur complètement chargé pendant un cycle complet de décharge. Sa valeur initiale théorique doit être indiquée par le constructeur. Elle diminue au fur et à mesure dans la vie de l'accumulateur. Cette capacité est exprimée en Ampères heure (Ah). [7]

Par exemple: un accumulateur de 60 Ah est capable de fournir 60 A pendant 1 heure, ou encore 30 A pendant 2 heures. Le courant est imposé éventuellement par la charge.

1.4.3. La densité d'énergie

La densité d'énergie est l'énergie emmagasinée par rapport à la masse ou au volume. La densité d'énergie s'exprime en Wattheure/kilogramme (Wh/kg) ou en Wattheure/litre (Wh/l). [8]

Par exemple : soit deux batteries, la première ayant une densité d'énergie de 100 Wh/kg et l'autre ayant 60 Wh/kg. Pour que les deux batteries aient la même énergie emmagasinée, il faut que la deuxième ait une masse plus grande, donc elle est plus lourde. (Le même raisonnement pour le cas du volume).

1.4.4. Résistance interne

Un accumulateur (ou une batterie) peut être modélisé comme une source de tension en série avec une résistance. En pratique, la résistance interne d'une batterie dépend de sa taille, les propriétés chimiques, l'âge, la température et le courant de décharge. [9]

Il y a deux composantes qui influent sur la résistante interne de l'accumulateur : la résistance électronique et la résistance ionique, la résistance interne étant la somme des deux. La résistance électronique en raison de la résistivité des matériaux de composants. Quant à la résistance ionique est due à des facteurs électrochimiques tels que la conductivité de l'électrolyte, la mobilité des ions, et la surface de l'électrode. [10]

1.4.5. L'état de charge (SOC)

La batterie peut être vue comme un réservoir d'énergie dont la quantité évolue constamment. En conséquence, son état de charge est identifié comme la capacité contenue dans cette batterie, elle est affectée par les conditions d'opération (le courant, la température…etc.). Ce paramètre est exprimé en pourcentage (%). On attribue 100% (ou 1) pour une batterie pleinement chargée et 0% (ou 0) pour une batterie dite vide. [4]

$$\mathbf{SOC}_{\%} = \frac{\mathbf{Capacité\ actuelle\ (Ah)}}{\mathbf{Capacité\ maximale\ (Ah)}} * \mathbf{100}$$

1.4.6. La profondeur de décharge (DOD)

Le DOD est la quantité de capacité retirée durant la phase de décharge lors d'un cycle à partir d'une batterie pleinement chargée. Ce paramètre est exprimé, en pourcentage (%). [4]

$$\mathbf{DOD}_{\%} = \frac{\mathbf{La\ capacité\ rétiré\ d'une\ batterie\ chargé\ (Ah)}}{\mathbf{Capacité\ maximale\ (Ah)}} * \mathbf{100}$$

1.5. Les phases de fonctionnement

Quelle que soit la technologie employée, les accumulateurs passent par au moins deux phases de fonctionnement : la charge et la décharge. C'est le principe même de l'accumulateur : on stocke de l'énergie (charge) pour la restituer ensuite (décharge).

1.5.1. La charge

La charge est la phase de stockage d'énergie dans l'accumulateur. Pour l'effectuer, on utilise un chargeur qui sera spécifique pour chaque technologie de batterie.

Le rendement de la charge n'est pas de 100% mais plutôt de 50 à 75%. On apportera donc plus d'énergie à l'accumulateur qu'il ne sera capable d'en restituer ensuite.

1.5.2. La surcharge

Lorsque l'accumulateur est totalement chargé et qu'on continue de le charger, il passe en surcharge. Les effets peuvent être une simple élévation de la température, une destruction partielle de l'accumulateur ou même une l'explosion de l'élément.

1.5.3. La décharge

Une fois que l'accumulateur est chargé, on peut utiliser l'énergie qui y est emmagasinée. L'énergie est donc transformée d'une forme chimique vers une forme électrique. L'élément fournit alors de l'énergie tant qu'on lui en demande et surtout tant qu'il lui en reste.

Si on demande trop d'énergie à un accumulateur pendant une longue durée, ce dernier va être en état de décharge profonde, et cela endommage l'accumulateur.

Lorsque l'accumulateur est vide, on observe une chute brutale de la tension à ses bornes.

1.5.4. L'autodécharge

C'est la perte de capacité lors du stockage en raison du courant de fuite interne entre les plaques positives et négatives. Cela veut dire que même si l'accumulateur n'est pas utilisé, il se décharge. Elle est exprimée en pourcentage par mois et varie selon le type de la batterie. [4, 8, 11]

1.6. Les différents types de batteries et d'accumulateurs

1.6.1. La batterie au plomb

C'est le type de batterie le plus largement répandu sur le marché et c'est aussi la plus ancienne technologie de stockage d'énergie. Elle est très utilisée dans les voitures pour alimenter le démarreur.

Remarque: plus du tiers de la production mondiale de plomb est destinée à la fabrication de batteries au plomb.

Figure 1.1: Batterie au plomb pour les automobiles

1.6.1.1. Historique

Les premières sources chimiques de courant électrique apparaissaient à la fin du XVIIIème siècle. En 1800, Volta en fait la première démonstration en réalisant un empilement successif d'une lame de zinc, d'une lame de feutre imbibée d'eau vinaigrée et d'une lame de cuivre. Il crée le premier dispositif de production électrochimique d'énergie électrique, on lui attribue le nom de « pile » de par sa structure.

C’est en 1859 que Gaston Planté réalise le premier accumulateur. Il est composé de deux feuilles de plomb roulées en spirale, séparées par une toile de lin et plongées dans un bac contenant une solution d’acide sulfurique à 10%.

Figure 1.2: Première batterie d'accumulateurs conçue par Gaston Planté

En 1880, Camille Faure facilite la création de l’accumulateur au plomb : une pâte à base d’oxyde de plomb et d’acide sulfurique est appliquée directement sur les lames de plomb ; elles sont maintenues en place en enroulant le tout dans du feutre.

En 1881, Henri Owen Tudor développa une batterie formée d'une plaque négative en plomb spongieux et d'une plaque positive en oxyde de plomb. Il fait breveter son invention et devient l’inventeur de la batterie au plomb.

Depuis, les batteries au plomb n’ont pas cessé d’évoluer, ce qui leur permet aujourd’hui d’être très largement utilisées (elles représentent 60 à 65% du marché mondial des batteries). Leur fonctionnement reste cependant basé sur celui bâti par Henri Tudor.

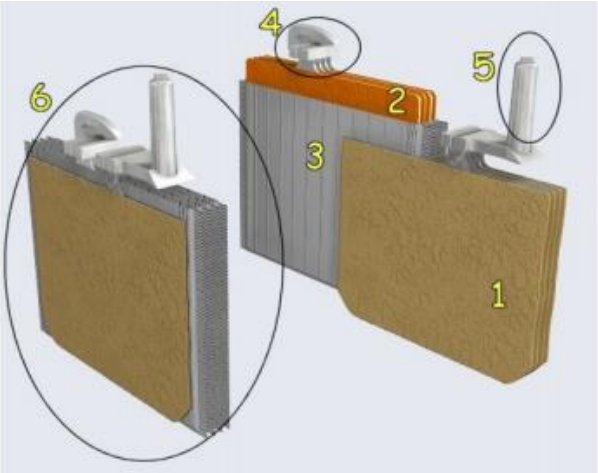
1.6.1.2. Constitution générale

Une batterie au plomb est composée de plusieurs éléments dont le nombre détermine la tension que la batterie délivrera. Un élément est une association d’électrodes positives et négatives baignant dans de l’électrolyte. L’ensemble a une différence de potentiel entre ses bornes d’environ 2 Volts (pour les batteries au plomb).

L'électrode positive est composée d'oxyde de plomb et l'électrode négative de plomb. L'électrolyte, dont la fonction est d'assurer le transfert des ions entre les électrodes pendant la réaction chimique, est une solution acide. [8]

Un élément de batterie de démarrage (technologie « plomb ouvert ») comprend des plaques planes positives **(2)** et négatives **(1)** assemblées en alternance (Voir figure 1.3). Le nombre de plaques pour chaque polarité et leur surface sont les paramètres qui définissent

la capacité de l'élément. Par exemple, l'électrode positive comporte ici 4 plaques en parallèle, reliées par un connecteur **(4)**. Pour éviter les courts circuits entre les plaques de polarité différente, un séparateur microporeux isolant est placé entre ces plaques lors du montage **(3)**. Les plaques positives et négatives sont assemblées en faisceaux **(6)** et plongées dans une solution d'acide sulfurique et d'eau distillée. Chaque faisceau constitue ainsi un élément. [11]



(**1**) Électrode négative, composée de 4 plaques en plomb spongieux (Pb)

(**2**) Électrode positive, composée de 4 plaques de dioxyde de plomb (PbO_2)

(**3**) Séparateur micro poreux (pochette en polyéthylène)

(**4**) Pontet de connexion en plomb

(**5**) Borne terminale négative

(**6**) Un élément Pb/PbO_2

Figure 1.3: Constitution d'une cellule de batterie au plomb

1.6.1.3. Réaction chimique

Le procédé chimique d'une batterie au plomb est constitué de deux électrodes : l'électrode négative est formée du plomb métallique (Pb) et l'électrode positive est formée de l'oxyde de plomb (PbO_2). Toutes les deux sont immergées dans une solution d'acide sulfurique (H_2SO_4), comme illustré sur la figure ci-contre.

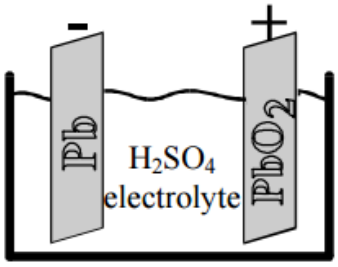


Figure 1.4: Représentation des électrodes dans une batterie au plomb

Lors de la décharge de la cellule, les deux électrodes accumulent le sulfate de plomb $PbS0_4$ (solide) et l'électrolyte est converti en H_2O, alors que l'inverse se produit lors de la charge. A l'électrode négative, ce processus peut être décrit par les équations chimiques suivantes:

$$Pb \leftrightarrow Pb^{2+} + 2é$$

$$Pb^{2+} + SO_4^{2-} \leftrightarrow PbSO_4$$

De même à l'électrode positive, nous pouvons écrire:

$$PbO_2 + 4H^+ + 2é \leftrightarrow Pb^{2+} + H_2O$$

$$Pb^{2+} + SO_4^{2-} \leftrightarrow PbSO_4$$

Donc la réaction globale s'écrit

$$Pb + PbO_2 + 2H_2SO_4 \leftrightarrow 2PbSO_4 + H_2O$$

Ou la réaction de charge se produit de droite vers la gauche. Quant à la réaction de décharge elle se produit de gauche vers la droite.

1.6.1.4. Dissipation de gaz (phénomène du Gassing)

Quand la cellule atteint la charge complète et la majorité du $PbSO_4$ a été converti en Pb et PbO_2, la tension de charge de la cellule devient supérieure à un certain seuil appelé la « tension de gazage ». A ce point, la réaction de surcharge commence, en plus des réactions de la charge normale, ce qui entraîne la production du gaz d'hydrogène et du gaz d'oxygène (réaction de l'électrolyse de l'eau).

$$H_2O \rightarrow H_2 + \frac{1}{2}O_2$$

La production excessive de gaz de la batterie est indésirable car elle entraîne un gaspillage d'énergie et une augmentation importante du temps de charge. En même temps, si le gaz n'est pas correctement ventilé, il sera recueilli et aura un potentiel explosif, en particulier dans des environnements où des étincelles électriques sont possibles. Dans les batteries VRLA (Voir 1.6.1.5.2), ce problème est minimisé car ces gaz se recombinent à une pression élevée, mais le gazage excessif à la suite d'une mauvaise charge oblige la valve de protection de l'évacuer. [12]

1.6.1.5. Les différents types de batteries au plomb

Il existe deux principaux types d'accumulateurs au plomb, la batterie dite ouverte et la batterie scellée ou à recombinaison de gaz

1.6.1.5.1. La batterie au plomb ouverte

Lors du fonctionnement de la batterie au plomb dite ouverte, il y a une production de gaz suite aux réactions secondaires de décomposition de l'eau. Ces gaz s'échappent naturellement par les orifices prévus au niveau des bouchons. Le dégagement du dihydrogène H_2 dans le lieu de stockage des batteries est une source de danger, car son mélange avec l'air ambiant est potentiellement explosif à partir de 4% en volume. Pour cela, dans le cadre du stationnaire de secours, une installation en locaux spécifiques ventilés est obligatoire.

Les batteries ouvertes produites aujourd'hui, sont souvent qualifiées de batteries « maintenance free » ou « sans entretien ». Ces appellations ont été choisies car la

consommation d'électrolyte est si faible que la réserve d'électrolyte d'origine est suffisante pour assurer le bon fonctionnement de la batterie pendant toute sa durée de vie.

1.6.1.5.2 La batterie à recombinaison de gaz

Les batteries à recombinaison de gaz sont appelées aussi batteries VRLA (Valve Regulated Lead-Acid) ou parfois batteries étanches. Les premières batteries à recombinaison de gaz sont apparues fin des années 1950, grâce à la fabrication d'un électrolyte gélifié. Puis dans les années 1970, la maîtrise des procédés d'absorption d'acide dans la fibre de verre a permis l'élaboration de séparateurs imbibés d'électrolyte : le marché de cette technologie à électrolyte immobilisé prenait son essor. Ce type d'électrolyte offre plusieurs avantages :

* Il permet la formation de chemins gazeux facilitant le transfert rapide du dioxygène, qui suit alors un cycle interne : produit à l'électrode positive, sa diffusion vers l'électrode négative est optimisée (10^5 fois plus rapide qu'en électrolyte liquide) et il atteint l'électrode négative où il y est réduit (formation de molécules d'eau).

Cette propriété se traduit par une très faible consommation en eau lors de surcharges de la batterie ; avantage séduisant pour le domaine du stationnaire de secours.

* Il supprime quasiment le phénomène de stratification de l'électrolyte.

* Il autorise le placement des batteries dans des locaux quelconques, dans n'importe quelle position (souvent horizontale, ce qui facilite l'accès aux bornes).

L'électrolyte immobilisé est la clef du processus de recombinaison, dont le rendement est élevé. Toutefois, 2 à 3% des gaz produits ne sont pas recombinés, et pour se prévenir de tout risque de surpression, une soupape d'aération régulée par pression est nécessaire. C'est pourquoi les batteries à recombinaison de gaz sont appelées aussi batteries VRLA (pour Valve-Regulated Lead-Acid) et parfois improprement batteries étanches. Cette technologie est par construction "sans maintenance", donc cette précision n'est jamais mentionnée, contrairement aux batteries ouvertes où ce qualificatif est employé lorsque la consommation d'eau est amoindrie en surcharge.

La soupape limite le dégagement des gaz vers l'extérieur et empêche l'entrée de l'oxygène atmosphérique en ne s'ouvrant que pour une surpression de l'ordre de 0,1 bar. La production d'hydrogène, bien que réduite, est inévitable et risquerait à terme de causer une surpression interne destructrice pour la batterie. Ainsi, plutôt que de fabriquer une batterie complètement étanche, l'échappement a été prévu. [3]

Tableau 1.1: Principales comparaisons des 2 familles technologiques de la batterie au plomb

Type de batterie au plomb	Ouverte	A recombinaison de gaz	
Electrolyte	liquide	Gélifié	Absorbé
Appellation Anglo-Saxonne	Flooded (ou Vented) battery	Gel VRLA (ou sealed battery)	VRLA (ou sealed) AGM separator battery
Avantages	* Durée de vie pouvant être importante (5 à 15 ans) * Technologie la moins chère	* Recombinaison ➔ Pas de perte en eau (pas d'entretien) * Très faible taux de dégagement de gaz (sécurité)	
Inconvénients	* Consommation d'eau (maintenance) * Installation en locaux spécifiques pour la ventilation (à cause du dégagement de gaz)	* Plus faible durée de vie * Plus sensible à la température	

1.6.2. Les accumulateurs Nickel-Cadmium et Nickel-Hydrure Métallique

Dans la famille des accumulateurs au nickel, on retrouve deux types d'accumulateurs qui correspondent à deux couples électrolytiques différents :

- le Nickel-Cadmium Ni-Cd découvert en 1899 par Jungner ;
- le Nickel-Hydrure Métallique Ni-MH (composé permettant de stocker de l'hydrogène) commercialisé (en 1990, en anglais : « Nickel-Métal Hydride »)

Ces accumulateurs sont les plus répandus dans les appareils portatifs.

Les batteries Ni-MH équipent aujourd'hui les voitures hybrides telles que la Toyota Prius ou la Honda Civic IMA. Ces batteries sont prévues pour durer toute la durée de vie de la voiture (garantie 8 ans).

A partir du 1er juillet 2006, une directive Européenne interdira la commercialisation du Cadmium dans les accumulateurs. Il s'agit de la directive 2002/95/CE du parlement européen et du conseil du 27 janvier 2003 relative à la limitation de l'utilisation de certaines substances dangereuses dans les équipements électriques et électroniques. Les batteries Ni-Cd sont donc appelées à disparaître.

1.6.2.1. Caractéristiques

Le Ni-MH remplace largement le Ni-Cd car il possède un meilleur rapport prix/longévité et a une densité d'énergie plus élevée. Le Ni-MH dispose néanmoins de quelques désavantages par rapport au Ni-Cd :

- une résistance interne plus élevée ;
- une plus grande fragilité car il ne supporte pas les surcharges ;
- son autodécharge est plus élevée ;
- le Ni-Cd est capable de fournir des pointes de courant plus importantes (de l'ordre de 10 fois).

La tension nominale d'un élément Ni-Cd ou Ni-MH est de 1,2V ; en fait, elle évolue de 1,35V (lorsque l'élément est complètement chargé) à 1V (lorsque l'élément est déchargé). La tension peut néanmoins atteindre 0,8V, mais c'est l'extrême limite car en-dessous l'élément risque d'être endommagé ou détruit.

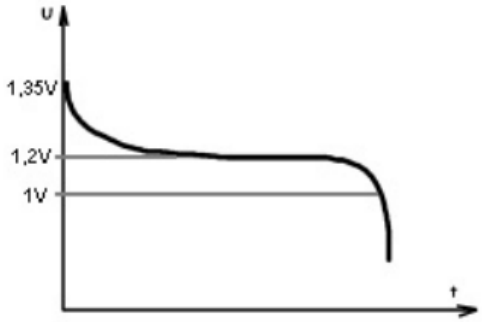


Figure 1.5: Courbe de décharge des cellules en Ni-Cd et Ni-MH

1.6.2.2. Constitution

L'électrode positive est toujours composée de nickel. Dans le cas des batteries Ni-MH, l'électrode négative est composée d'Hydrure Métallique alors que pour les batteries Ni-Cd, elle est composée du cadmium.

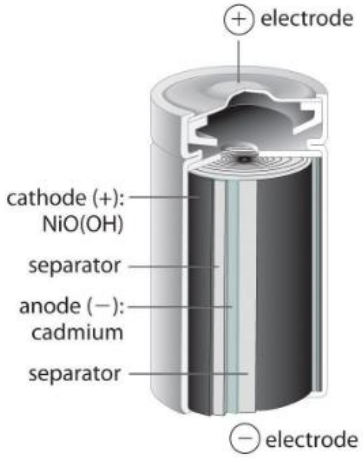


Figure 1.6: **Constitution d'une batterie Ni-Cd. [14]**

1.6.2.3. Effet mémoire

1.6.2.3.1. Le véritable effet mémoire

Ce phénomène concerne exclusivement les batteries Ni-Cd. Il est lié à des décharges périodiques parfaitement identiques. La batterie se décharge toujours jusqu'à la même valeur, et au bout d'un certain temps, il devient impossible de la décharger en dessous de cette valeur, même s'il lui reste de l'énergie. C'est comme si elle se souvenait du seuil de décharge habituel et comme si elle considérait que ce seuil est devenu son minimum. D'où le terme d'effet mémoire. Ce phénomène n'a jamais été constaté en utilisation courante mais dans des utilisations spécifiques.

1.6.2.3.2. L'effet mémoire que nous constatons

Pourtant, dans certains cas on parle « d'effet mémoire ». Pourquoi ? En fait, cet "effet" est constaté sur les appareils portatifs qui se coupent dès que le seuil de tension de la batterie devient trop faible. Cela est mis en place pour protéger les batteries des décharges profondes qui leurs nuisent. Quand l'appareil se coupe, nous considérons que la batterie est vide… Malheureusement « l'effet mémoire » est peut-être à l'origine de cette coupure, l'appareil s'éteint bien avant la fin de l'autonomie normale de la batterie.

Explication : ce phénomène provient de la modification de la structure de l'électrolyte, la modification est due à des surcharges de la batterie qui entraînent un palier de tension lors de la décharge.

Au début de la décharge, tout se passe normalement. Au bout d'un moment, une chute de tension de quelques millivolts se produit (dût à la modification de l'électrolyte). La décharge devrait se poursuivre normalement (avec une tension très légèrement inférieure à ce qu'elle devrait être) jusqu'à ce que la batterie soit vide. Mais l'appareil interprète cette légère chute de tension comme si c'était le reflet de la chute de tension de la fin de décharge normale. Et donc il s'arrête avant que la batterie soit complètement déchargée.

1.6.2.3. Conditions d'utilisation

Une batterie Ni-Cd ou Ni-MH doit permettre au moins 5OO cycles de charge/décharge; pour cela, il faut prendre des précautions d'utilisation :

- éviter les surcharges ;
- proscrire les décharges profondes (<1V par élément)) ;
- ne jamais les court-circuiter ;
- attendre 1 heure avant d'utiliser une batterie qui vient d'être chargée ;
- attendre 15 minutes avant de recharger une batterie qui vient d'être déchargée ;
- une batterie Ni-Cd doit être stockée déchargée, une batterie Ni-MH doit être stockée chargée.

1.6.3. Les accumulateurs au Lithium ion (Li-ion)

C'est la dernière génération d'accumulateurs. Le principe est connu depuis la fin des années 70 mais le lithium étant instable à la charge, la commercialisation n'a été possible qu'en 1991.

Les accumulateurs Li-ion sont très utilisés car :

- Ils offrent une densité d'énergie très supérieure aux autres technologies entre 80 à 160 Wh/kg (un poids inférieur pour une même énergie fournie.
- Il n'y a pas d'entretien particulier à apporter.

Par contre :

- Ces batteries s'usent même lorsqu'elles ne servent pas ;
- Durée de vie de 2 à 3 ans après leur fabrication ;
- Risque d'explosion si elles ne sont pas chargées correctement (les constructeurs intègrent dans les batteries des circuits qui coupent la charge si les caractéristiques de la batterie sont anormales) ;
- Capacité faible (150 à 4500mAh)

Depuis 1999 est apparue une nouvelle génération d'accumulateurs Li-po : le Lithium-ion Polymère (Li-po). L'électrolyte est un polymère gélifié qui permet d'obtenir des éléments très fins, souples et se présentant sous la forme de paquets. Elle doit, à terme, revenir moins chère à la réalisation que le Li-ion classique. [8]

1.6.4. Tableau récapitulatif

Le tableau suivant montre une comparaison des caractéristiques essentielles des différentes technologies de batteries décrites auparavant.

Tableau 1.2: Résumé des caractéristiques essentielles des batteries au plomb, Ni-Cd, Ni-Mh, Li-ion et Li-po [13].

Caractéristiques	**Plomb**	**Ni-Cd**	**Ni-MH**	**Li-ion**	**Li-po**
Energie **[Wh/kg]**	30 à 50	45 à 80	60 à 120	110 à 160	100 à 130
Tension nominale d'un élément **[V]**	2	1,25	1,25	3,6	3,6
Résistance interne (mΩ)	<100 (pour un pack 12V)	100 à 200 (pour un pack de 6V)	200 à 300 (pour un pack de 6V)	150 à 250 (pour un pack de 7.2V)	200 à 300 (pour un pack de 7.2V)
Nombre de cycles charge/décharge (pour un DOD de 80%)	200 à 300	1500	300 à 500	500 à 1000	300 à 500
Temps de charge rapide **[h]**	8 à 16h	1h (typique)	2 à 4h	2 à 4h	2 à 4h
Autodécharge par mois	5%	20%	30%	10%	10%
Tolérance à la surcharge	Haute	Moyenne	Faible	Très faible	Faible

1.7. Charge d’une batterie au plomb

1.7.1. Mode de charge à tension constante (CV)

Le mode de charge à tension constante est le moyen le plus simple pour charger la batterie. Sa courbe de charge est représentée sur la figure 1.7. On peut voir que le courant de charge diminue progressivement lorsque la batterie tend vers l'état de la charge complète. Cette méthode ne provoque pas l’augmentation significative de la température de la batterie, et le problème de la surcharge ne produira pas. Cependant, elle a besoin d'un temps de chargement long, et le courant au début de charge peut excéder le courant nominal.

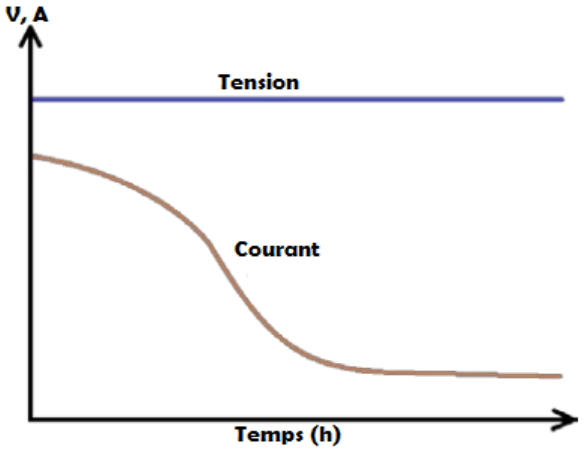


Figure 1.7: Courbe de charge à tension constante [15]

1.7.2. Mode de charge à courant constant (CC)

La courbe de charge, en utilisant le mode à courant constant, est représentée dans la figure 1.8. Dans ce mode, une source de courant est utilisée pour entraîner un courant uniforme à travers la batterie dans une direction opposée au sens de la décharge. Il est possible que le courant de charge puisse être réglé en dessous du courant nominal de sorte qu'il ne sera pas supérieur à la limite nominale. D'autre part, la tension de charge dépend du courant de charge et le temps de charge peut être estimé facilement. Cependant, l'inconvénient est que cela peut causer le problème de surcharge, et la température de la batterie peut augmenter rapidement.

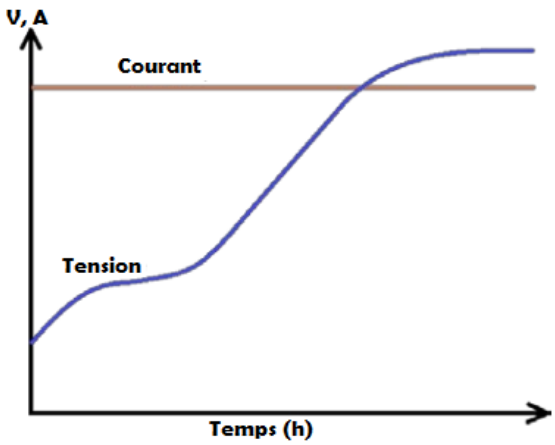


Figure 1.8: Courbe de charge à courant constant [15]

1.7.3. Mode de charge à courant constant - tension constante (CC/CV)

Ce mode combine à la fois la méthode de charge courant constant et tension constante. Dans le premier stade, appelé bulk, le courant constant est utilisé pour charger la batterie jusqu'à ce que la tension de la batterie atteigne la tension de surcharge. Ensuite, le mode de charge passe à l'étape de la tension constante, appelé phase d'absorption, ou la tension de la batterie est maintenue sur une valeur bien définie. La figure 1.9 indique la courbe de charge. L'avantage de cette méthode est que le temps de charge peut être réduit de façon considérable. [15]

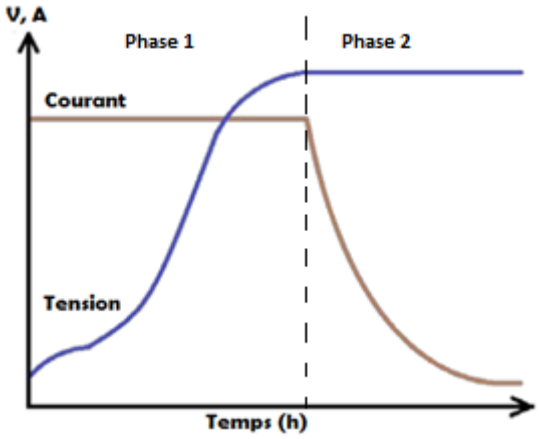


Figure 1.9: Courbe de charge à tension constante – courant constant [15]

A noter que la valeur typique du courant de charge typique est de C / 10. Il pourrait être augmenté pour une charge rapide mais peut causer des problèmes pour la batterie. Pour la tension maximale, son choix est très critique. La valeur typique est de 2,30 V à 2.45V. Si une charge lente est acceptable, ou la température ambiante peut dépasser 30 ° C, la limite de tension recommandée est de 2.35V / cellule. Si une charge plus rapide est nécessaire, et la température ambiante reste inférieure à 30 °C, 2,40 à 2,45 V / cellule peut être utilisée. [13]

1.7.4. Mode de charge à trois phases

En plus des phases du CC/CV, une 3ème phase de tension constante, dite de maintien ou floating, est ajouté à la fin.

Une fois la batterie est complètement chargée et peut être opérationnelle, un niveau inférieur de tension est maintenu aux bornes de la batterie. Cette étape est ajoutée afin de compenser les pertes par le phénomène d'autodécharge.

La tension de maintien recommandée de la plupart des batteries au plomb à basse pression est comprise entre 2,25 à 2.30V par cellule. Un bon compromis est 2.27V. [13]

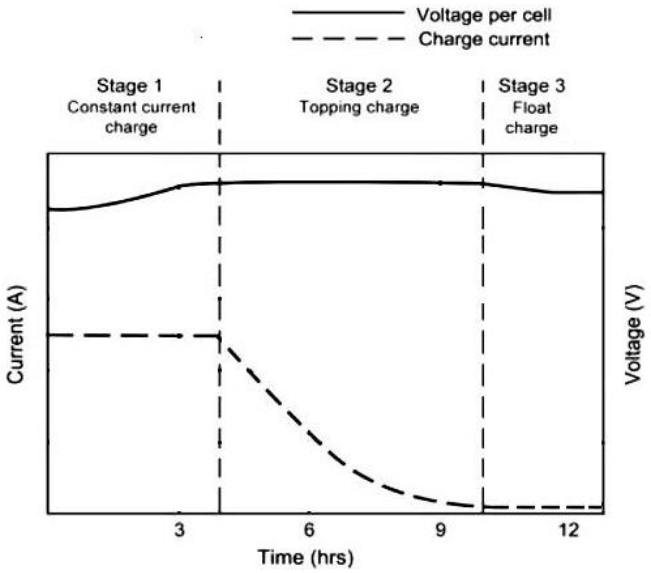


Figure 1.10: Courbe de charge à trois phases [13]

1.7.5. Mode de charge avec impulsions

La figure 1.11 représente le profil les cycles de la charge avec impulsions. Chaque cycle de charge comprend la une étape de « charge » et une étape de « repos ». Dans la période de « charge », la batterie est en train d'être chargée. Dans la période de «repos», la batterie est dans un état de repos où la batterie a plus de temps pour équilibrer la réaction batterie chimique. Par conséquent, la tension de la batterie devient plus stable, et sa durée de vie peut être prolongée. Un autre avantage est que le temps de charge peut diminuer en utilisant un courant qui peut être surélevé. [15]

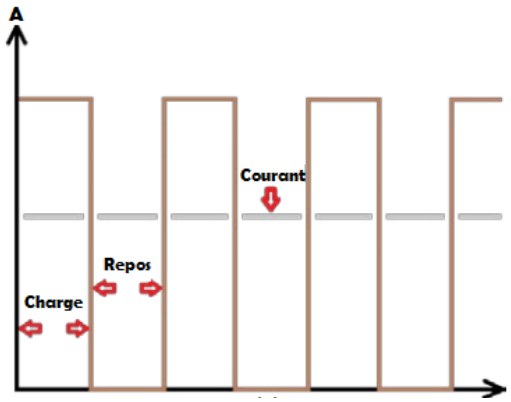


Figure 1.11: Courbe de charge avec impulsions [15]

1.8. Conclusion

Dans ce chapitre nous avons introduit des généralités sur les accumulateurs et les batteries. Plusieurs grandeurs peuvent caractériser les batteries, les plus essentielles sont : les différentes types de tensions, la capacité, la densité d'énergie, la résistance interne, l'état de charge et la profondeur de décharge. Les batteries passent par différents états de fonctionnement, nous citons l'état de charge, de décharge, d'autodécharge et de surcharge. Il existe plusieurs types de batterie, chacune a ses propres constituants, caractéristiques et ses domaines d'utilisation.

Chapitre 2

Circuit de charge

2.1. Introduction

Dans notre travail nous avons réalisé un chargeur de batterie au plomb avec une tension nominale de 12V. Dans ce chapitre nous décrirons la partie électrique du circuit de charge après avoir présenté le fonctionnement global du circuit avec son schéma synoptique. Nous présenterons enfin les détails de chaque bloc avec les composants choisis et leurs caractéristiques.

2.2. Fonctionnement global du chargeur

Le circuit de charge conçu met en œuvre l'algorithme de charge à trois phases : Une première à Courant constant (Bulk), une seconde tension constante (absoption) et enfin une dernière à tension constante de maintien (dite de floating). Ce circuit contient plusieurs blocs fonctionnels: Régulateurs, Amplificateurs opérationnels, transistors, éléments passifs, capteurs, … ils sont tous bâtis autour d'un seul cerveau qui est le microcontrôleur.

Le circuit nécessite une alimentation d'au moins 18V. L'algorithme de charge à 3 phases impose la prise en considération du type d'alimentation (en tension ou en courant). Pour cela il devrait avoir deux blocs responsables de chaque type d'alimentation. Le niveau de tension responsable à la charge doit être régulé en fonction de la phase et la température ambiante, nous avons donc réalisé un hacheur qui va convertir les 18V en une tension d'environ 14.4V pour la $2^{ème}$ phase et de 13.8V pour la $3^{ème}$ phase. Nous avons prévu aussi une option pour que l'utilisateur puisse ajuster le niveau de courant souhaité. Cela permettra d'avoir un chargeur compatible avec des batteries de différentes capacités. Pour l'alimentation des différents circuits intégrés et autre modules nous avons ajouté des régulateurs de 15V (pour les amplificateurs, le driver des MOSFET et l'optocoupleur) et de 5V (pour le microcontrôleur, l'afficheur LCD et le capteur de température).

La figure suivante montre le schéma synoptique global du circuit de charge

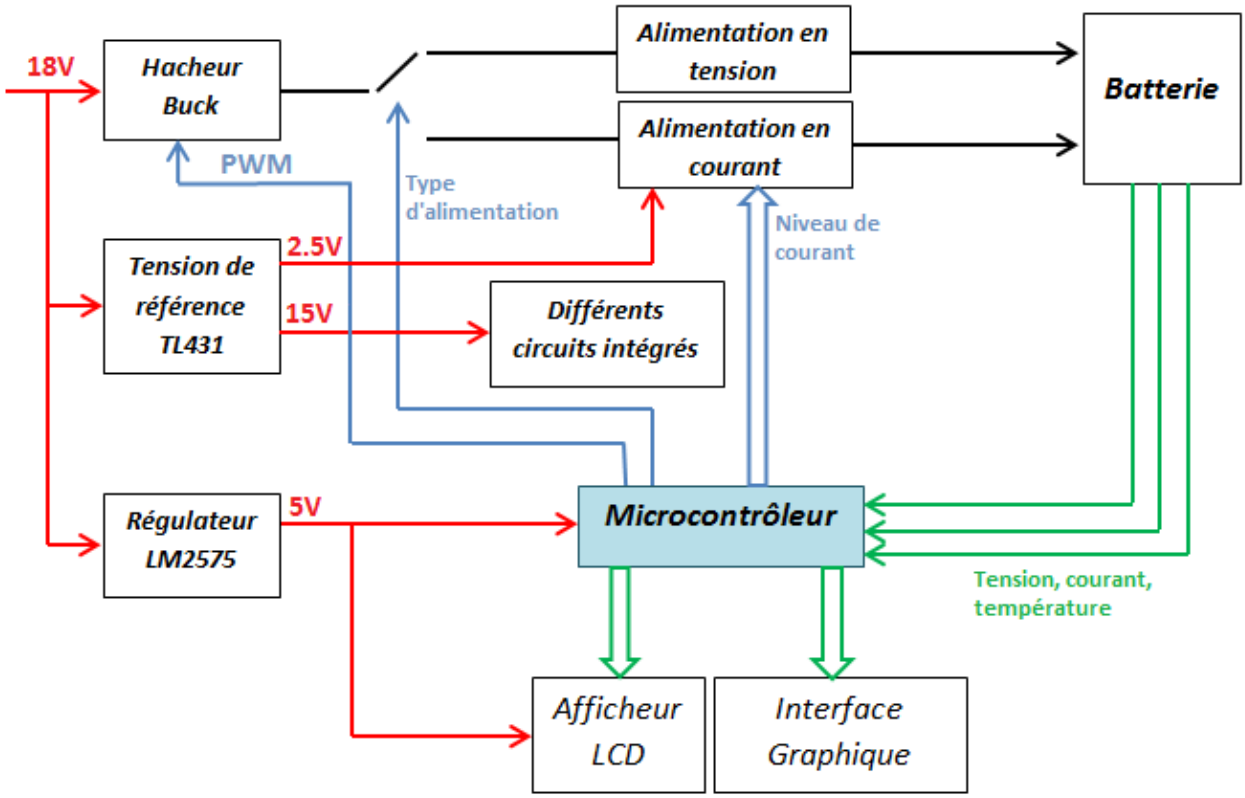


Figure 2.1: Schéma synoptique global du chargeur

2.3. Détails de chaque bloc et caractéristiques des composants choisis

Nous allons décrire le fonctionnement de chaque bloc. Nous allons présenter leurs schémas électriques, leurs caractéristiques, leurs choix et dimensionnement.

2.3.1. Le microcontrôleur PIC18F2550

En général, les chargeurs de batteries simples ne s'adaptent pas facilement avec le changement de l'état de charge ou avec le changement des paramètres de la batterie. Cela peut laisser la batterie mal chargée, ou réduire sa durée de vie ou même provoquer des risques sécuritaires [16]. Pour cela nous avons utilisé un microcontrôleur pour fournir l'intelligence nécessaire pour surmonter ces problèmes et assurer une charge efficace et autonome. Les microcontrôleurs fournissent également une flexibilité dans le choix de l'algorithme de charge optimal grâce à sa facilité de développement.

Pour notre projet nous avons choisi le PIC comme microcontrôleur avec le C comme langage de programmation. Nous avons choisi ce microcontrôleur car il est populaire, disponible sur le marché et nous sommes plus familiarisés avec son utilisation. Le composant choisi est le PIC18F2550 car il la capacité de communication série USB, c'est-à-dire la possibilité d'échanger les informations avec l'ordinateur et d'enregistrer et de suivre des données en temps réel.

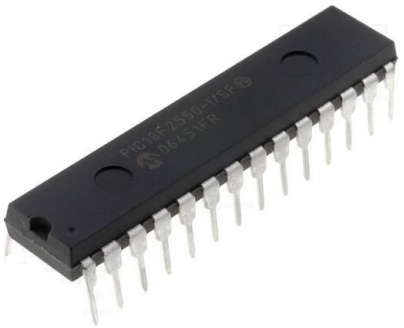


Figure 2.2: Boitier du microcontrôleur PIC1F2550

Le PIC18F2550 est un microcontrôleur de la famille 18F fabriqué par Microchip Technology. C'est un microcontrôleur 8-bits avec 28 pins. Il peut exécuter jusqu'à 12 millions d'instructions par seconde. Il présente 3 ports d'entrée/sortie bidirectionnels : PORTA et PORTB avec 8 bits et le PORTC avec 7 bits. Il comprend un oscillateur interne avec une fréquence allant de 31 KHz à 8 MHz. Il contient un module de conversion analogique-numérique et possède 10 entrées (canaux). Ce module permet la conversion avec une résolution de 10 bits. Le PIC18F2550 dispose 4 Timers et de 2 sorties PWM. Il se distingue de nombreux autres PIC par son module de communication USB.

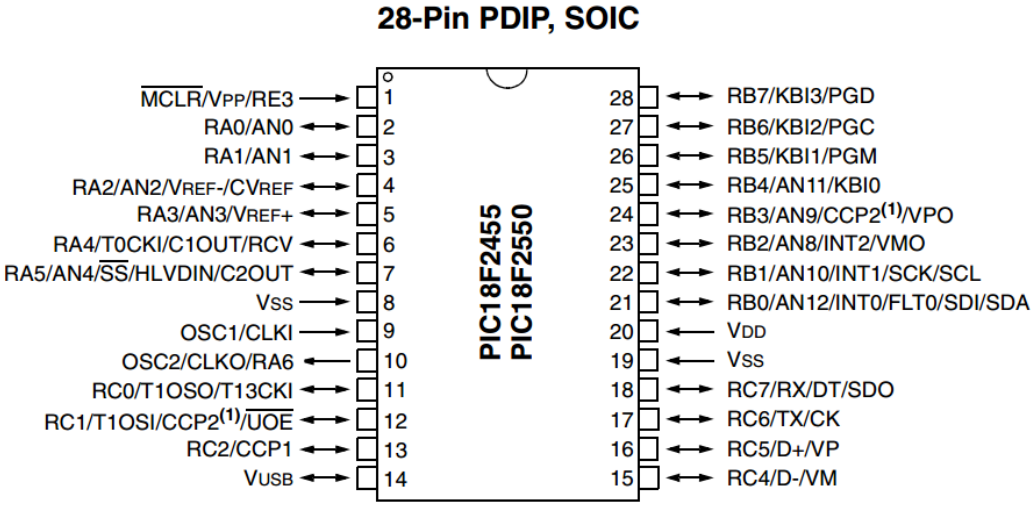


Figure 2.3: Brochage du PIC18F2550. [17]

Résumé des connexions du PIC

Le tableau suivant résume les configurations des différentes broches du PIC et ces connexions avec le reste du circuit.

Tableau 2.1: Configurations et fonctions des broches du PIC

Port	Configuration du port	Fonction
RA0	Entrée analogique	Mesure de la tension
RA1	Entrée analogique	Mesure du courant en mode CC
RA2	Entrée analogique	Mesure du courant en mode CV
RA3	Entrée analogique	Mesure de la température
RB0, RB1, RB2, RB3	Sortie numérique	Donnée vers le LCD
RB4	Sortie numérique	Sélection du registre (Pin **RS** du LCD)
RB5	Sortie numérique	Activation (Pin **E** du LCD)
RC0	Sortie numérique	Activation/Désactivation générateur de courant
RC1	Sortie numérique	Activation/Désactivation générateur de tension
RC2	Sortie	PWM (commande du hacheur)
RC4	/	Communication USB (D+)
RC5	/	Communication USB (D-)
RC6	Sortie numérique	Commande du R-2R (1)
RC7	Sortie numérique	Commande du R-2R (2)
MCLR	Entrée	Réinitialisation
VDD	Entrée	Alimentation (5V)
GND	Entrée	Alimentation (0V)

2.3.2. Conception de l'alimentation du chargeur

2.3.2.1. Le convertisseur DC-DC Buck et son dimensionnement

Le niveau de tension dans la 2ème et la 3ème phase de charge étant dépendante de la température, il est impératif de faire une bonne régulation. Pour cela nous avons utilisé un convertisseur abaisseur dont le niveau de tension de sortie est commandé par PIC via le rapport cyclique du PWM.

Un convertisseur Buck est un convertisseur DC-DC qui abaisse la tension à partir de son entrée (alimentation) à sa sortie (charge). C'est une classe des alimentations à découpage comprenant typiquement au moins deux semi-conducteurs (une diode et un transistor ou deux transistors) une inductance (pour stockage d'énergie) et un condensateur (pour filtrage en sortie). Ces convertisseurs ont un rendement meilleur que les régulateurs linéaires.

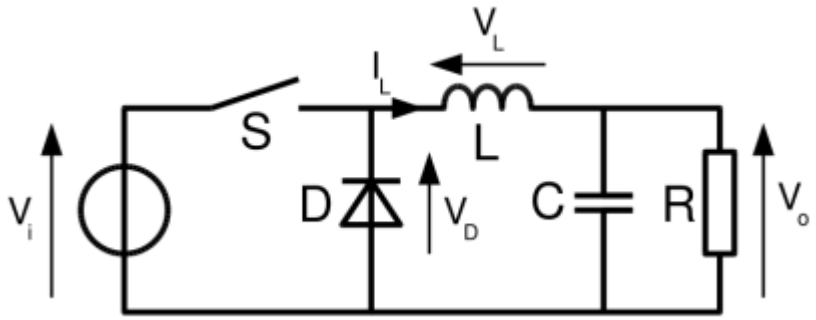


Figure 2.4: Circuit général du convertisseur Buck. [18]

V_i : Tension d'entrée

V_O : Tension de sortie

V_L : Tension aux bornes de la bobine

V_D : Tension aux bornes de la diode

I_L : Courant traversant la bobine

S : Commutateur (MOSFET)

D : Diode

L : Bobine

R : Résistance

C : condensateur

Lorsque le commutateur du Buck est passant, la tension aux bornes de la bobine est $\boldsymbol{V_i - V_o}$. En utilisant les équations d'inductance, le courant dans la bobine augmente d'un taux de $\frac{\boldsymbol{V_i - V_o}}{\boldsymbol{L}}$. Dans ce cas la diode est polarisé en inverse et ne conduit pas.

Lorsque le commutateur est bloquant, le courant doit encore couler vu que l'inductance fonctionne pour maintenir le même courant circule. Par conséquent, le courant circule toujours dans l'inductance et dans la charge. La diode fait alors le trajet de retour avec un courant égal à l'écoulement à travers elle.

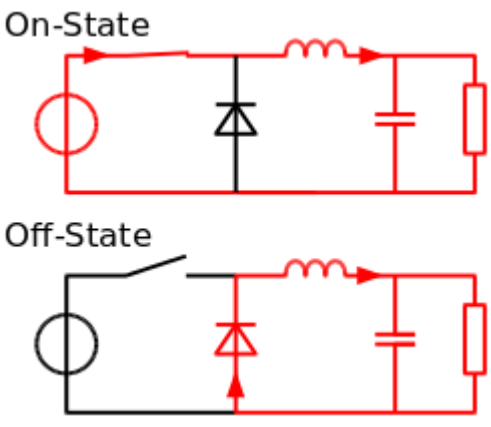


Figure 2.5: Fonctionnement du Buck en mode passant et en mode bloquant. [18]

Dans ce même cas, la polarité de la tension aux bornes de l'inductance est inversée et par conséquent le courant dans l'inductance décroît avec une pente égale à $\frac{-V_o}{L}$. [18]

La relation liant tension d'entrée avec celle de la sortie est linéaire et donnée par : $V_o = \alpha V_i$

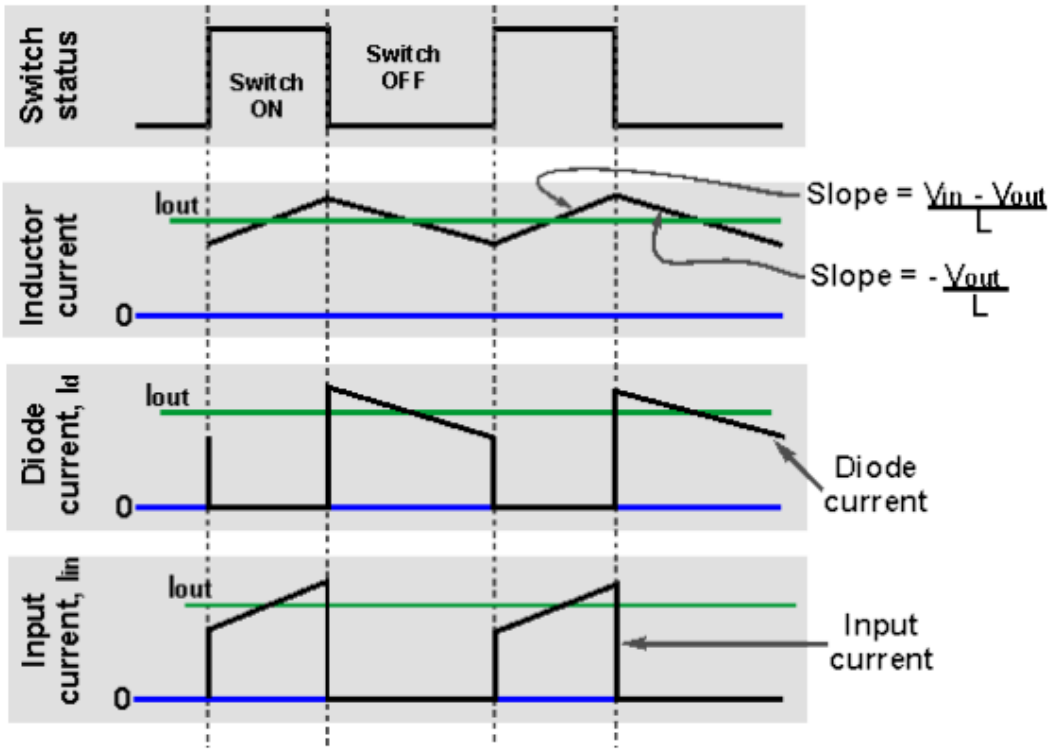


Figure 2.6: Ondes de courant et de tension dans les éléments du convertisseur Buck. [18]

Choix du MOSFET

Il faut choisir le MOSFET rapide qui permet des fréquences de commutation très grande (supérieure à 50 KHz), une résistance drain-source ($R_{DS(ON)}$) la plus faible possible, un courant direct supérieur à 6 A et une tension drain supérieure à 22V. [20]

Nous avons choisi le MOSFET IRFZ44 canal N pour sa résistance interne très faible 17.5 mΩ. Ce MOSFET se caractérise par un courant direct maximal est de 49A, quant à la tension drain maximale, elle est de 55V.

Choix de la diode

La diode utilisée doit être : d'une part, de puissance pour supporter le fort courant qui la traverse. Et d'une autre part rapide pour fonctionner correctement dans les hautes fréquences.

Nous avons choisi la diode Schottky : SBL1640CT.

Dimensionnement de la bobine

La bobine a été calculée par la formule du cours ([19]) en prenant l'ondulation du courant ΔI_{Lmax} circulant la bobine égale à 1A.

$$L = \left(V_{in\_max} - V_{out}\right) * \frac{V_{out}}{V_{in\_max}} * \frac{1}{f} * \frac{1}{\Delta I_{Lmax}}$$

$$L = (22 - 14.5) * \frac{14.5}{22} * \frac{1}{50000} * \frac{1}{1} = 98.8\mu H$$

Donc on prend une bobine d'environ 100µH.

Dimensionnement des condensateurs

- **A l'entrée :** La valeur du condensateur est prise supérieure à 22µF chaque 1 ampère de courant. Donc C_1doit être supérieure à 6 * 22 = 132 µF. Nous prenons un condensateur chimique de 220µF/40V.
- **A la sortie :** Nous avons pris l'ondulation de tension $\Delta V_{out} = 0.1V$. La formule du cours ([19]) est :

$$\boldsymbol{C_2 = \frac{(\Delta I_L)_{Max}}{8 f \Delta V_{out}} = \frac{1}{8 * 50000 * 01} = 25\mu F}$$

Nous avons choisi de mettre 2 condensateurs (22µF/25V chacun) en parallèle de capacité équivalente supérieure pour un bon filtrage. La raison du parallélisme est la réduction de la résistance équivalente série (ESR).

En résumé les paramètres du hacheur sont listés dans le tableau suivant :

Tableau 2.2: Résumé des paramètres du convertisseur Buck

Paramètres	Valeur
Tension d'entrée minimale $\boldsymbol{V_{in\ min}}$	16 V
Tension d'entrée maximale $\boldsymbol{V_{in\ max}}$	22 V
Tension de sortie $\boldsymbol{V_{OUT}}$	14.5 V
Courant de sortie $\boldsymbol{I_{OUT}}$	6 A
Ondulation maximale de la tension $\boldsymbol{\Delta V_{out}}$	1 A
Ondulation maximale du courant $\boldsymbol{\Delta I_L}$	1 A
Fréquence choisie	50 KHz
Inductance ***L***	100 µH
Capacité d'entrée $\boldsymbol{C_1}$	132 µF
Capacité d'entrée $\boldsymbol{C_2}$	44 µF

2.3.2.2 L'optocoupleur HCPL 2200

L'optocoupleur est un composant qui transfère des signaux électriques entre deux circuits isolés en utilisant la lumière. Un optocoupleur contient une source (émetteur) qui convertit le signal d'entrée électrique en lumière. Cet émetteur est généralement des diodes électroluminescentes (LED). Un détecteur optique capte la lumière entrante et génère de l'énergie électrique, soit directement, soit par modulation du courant électrique circulant à partir d'une alimentation externe. Ce détecteur optique peut être une photorésistance, une photodiode, un phototransistor, ...

Dans notre circuit nous utilisons un optocoupleur, d'une part pour isoler la partie commande (PWM du PIC) et la partie puissance (Driver qui commande le MOSFET) et d'une autre part pour augmenter le niveau de tension, car le PIC délivre un signal de commande de 5V et le driver IR2111 ne peut fonctionner qu'avec un signal d'entrée supérieur, approximativement, à 10V. Nous avons choisi l'optocoupleur HCPL 2200 grâce à ses bonnes caractéristiques de commutation: temps de montée et de descente très faible (55ns et 15 ns respectivement) ce qui permet d'aller en hautes fréquences [21]. L'optocoupleur HCPL 2200 permet une alimentation allant de 4.5 à 20V. Dans notre cas il est alimenté par 15V avec la sortie du TL431.

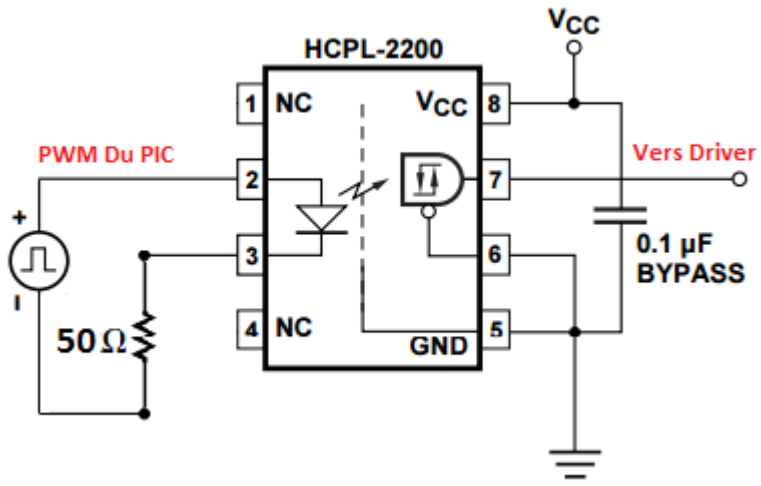


Figure 2.7: Schéma de l'optocoupleur HCPL avec les éléments externes

2.3.2.3. Le driver IR2111

Dans la commande du MOSFET et afin d'avoir une faible résistance de conduction $R_{DS(ON)}$, la tension grille-source V_{GS} doit être suffisamment élevée, mais inférieure à la tension de claquage du MOSFET. La fiche technique de l'appareil suggère $V_{GS} = 10V$ pour une bonne opération de commutation. Cependant, la source n'est pas liée à la masse ainsi que sa tension est toujours égale à la tension aux bornes de la batterie qui est dans notre cas environ 12V. D'où le besoin d'un circuit de commande capable de maintenir une tension de 10V entre la grille est la source.

La solution que nous avons adopté c'est l'utilisation d'un circuit intégré dédié pour cela. Ces circuits sont appelés « High Side Driver ». Ce qui est le cas du circuit intégré du constructeur International Rectifier IR2111. Ce circuit est conçu pour commander des MOSFET et IGBT en se basant sur un condensateur de Bootstrap externe. Ce condensateur se charge pendant l'état bas, puis se décharge pendant l'état haut produisant la tension V_{GS} nécessaire. [22]

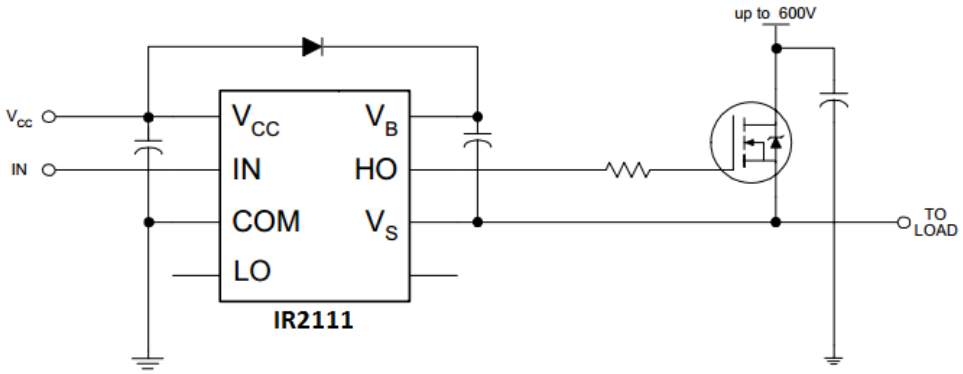


Figure 2.8: Schéma typique d'utilisation du driver IR2111 avec commande High Side. [23]

2.3.2.4. Schéma global de l'alimentation

Le schéma global de l'alimentation tenant compte de l'optocoupleur, le driver et les éléments du Buck est donné dans la figure suivante.

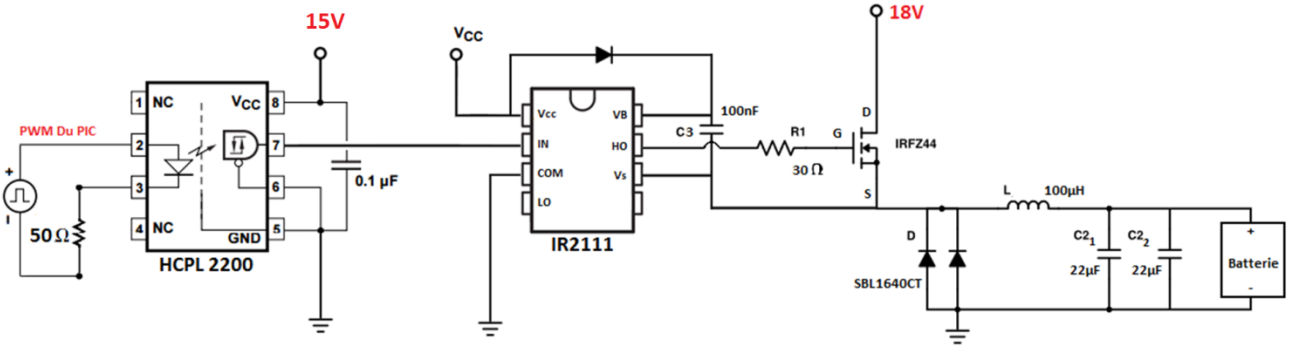


Figure 2.9: Schéma global de l'alimentation du chargeur

Remarque importante sur le driver IR2111

Bien que le driver fonctionne normalement sous une alimentation de 15V avec une charge résistive, il n'a pas pu fonctionner avec la batterie comme charge, car le condensateur du bootstrap n'a pas pu être chargé suffisamment, puisque la tension de la source du MOSFET est sur le potentiel de la batterie. Donc la solution proposée est d'utiliser une alimentation qui vaut 24V afin qu'il puisse commander le MOSFET. Mais la meilleure solution trouvée est d'utiliser le circuit optocoupleur/driver TLP250 avec une alimentation externe (ou sa masse doit être séparée à celle de la tension 18V). Le circuit complet est donné en annexe.

La figure suivante montre le montage réalisé.

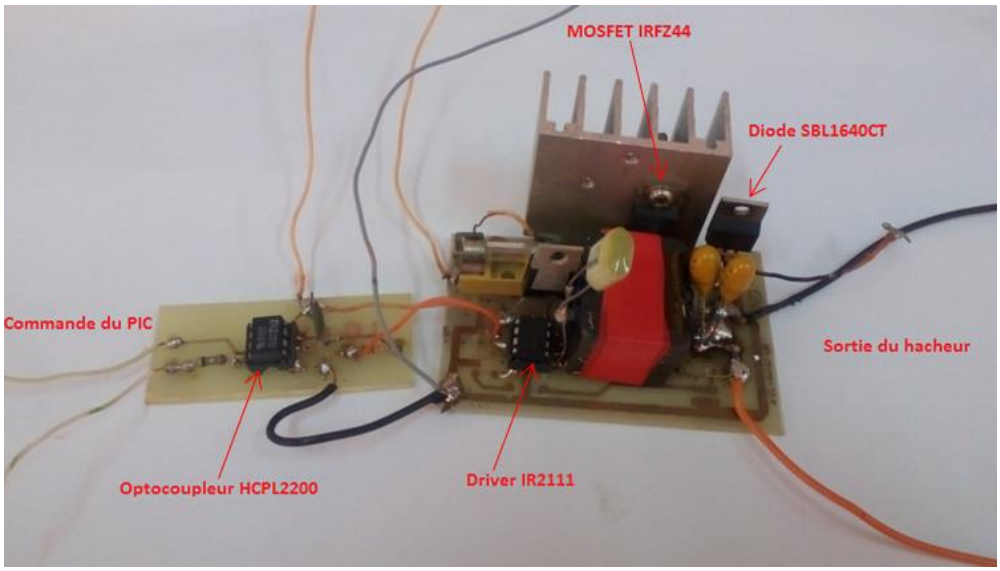


Figure 2.10: Montage réalisé du hacheur

Les figures suivantes montrent les résultats expérimentaux relatifs au hacheur Buck pour la fréquence 50KHz et un rapport cyclique de 60%.

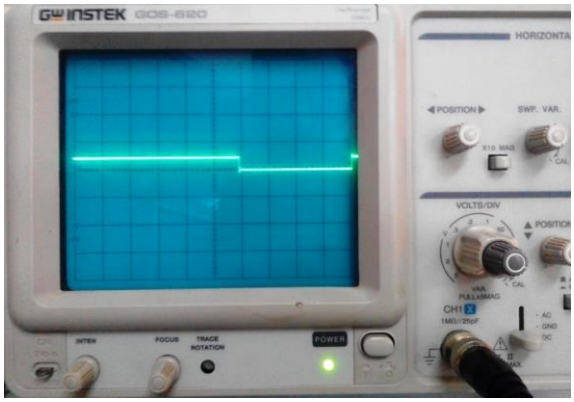


Figure 2.11: Visualisation de la sortie du PIC

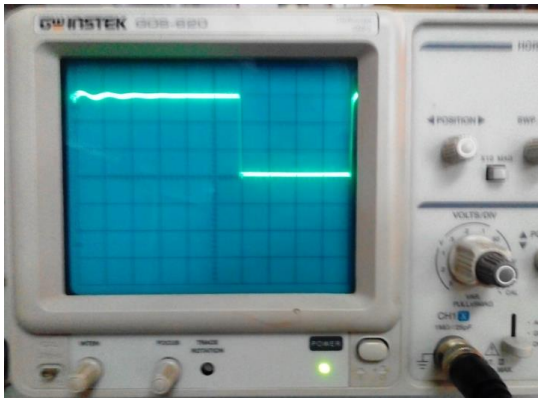


Figure 2.12: Visualisation de la sortie de l'optocoupleur

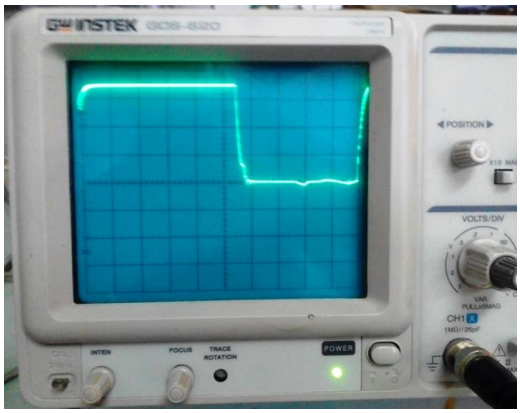


Figure 2.13: Visualisation de la tension V_{gs}

2.3.3. Le régulateur LM2575

Le LM2575 un régulateur à circuit intégré. Il est parfaitement adapté pour la conception facile et pratique d'un régulateur de commutation (convertisseur abaisseur Buck). Tous les circuits de cette série sont capables de délivrer un courant de 1 A à la charge. Ces régulateurs ont été conçus pour minimiser le nombre de composants externes pour simplifier la conception de l'alimentation. Etant donné que le convertisseur LM2575 est une alimentation à découpage, son efficacité est bien meilleure en comparaison avec les régulateurs linéaires populaires à trois broches, en particulier avec des tensions d'entrée plus élevées. Il présente un large intervalle de tension d'entrée s'étalant de 4.75 à 40V. Dans de nombreux cas, la puissance dissipée par le régulateur de LM2575 est si faible qu'aucun radiateur n'est nécessaire ou sa taille peut être réduite de façon spectaculaire. Les caractéristiques LM2575 garantie une tolérance de ± 4% sur la tension de sortie. [24]

Dans notre circuit, la tension d'entrée est de 18V et la sortie est de 5V afin d'alimenter les le PIC18F2550, l'afficheur LCD et le capteur de température. Les composants associés au LM2575 sont les suivants: Un condensateur d'entrée, un condensateur de sortie, une bobine et une diode shotteky. Le shéma est le suivant :

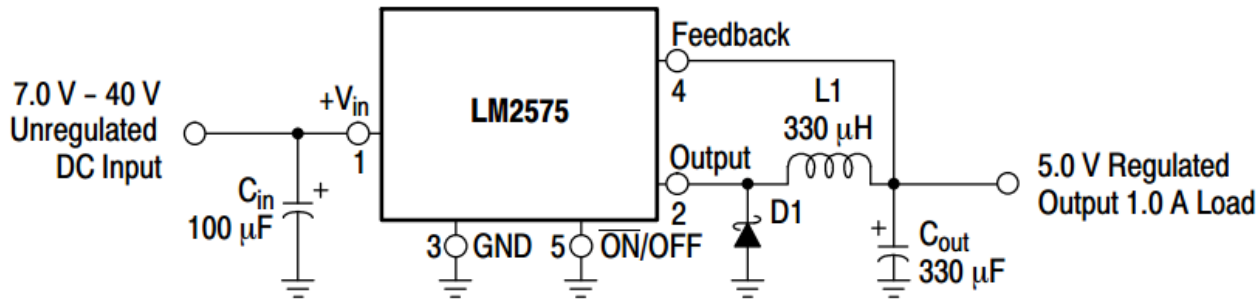


Figure 2.14: Régulateur LM2575 avec les éléments externes. [24]

2.3.4. Le circuit TL431

Le circuit TL431 est un régulateur shunt ajustable à trois broches, avec une stabilité thermique permettant un fonctionnement dans les plages de température pour les applications automobile, commerciale et militaire (de −40°C à 125°C). La tension de sortie peut être réglée, via deux résistances externes, à une valeur comprise entre V_{ref} (environ 2,5 V) et 36 V. Ce dispositif a une impédance de sortie typique de 0,2 Ω. Ces dispositifs remplacent les diodes Zener dans de nombreuses applications, comme les alimentations ajustables et les alimentations à découpage. Le TL431 est offert en trois catégories, avec des tolérances initiales (à 25 °C) de 0,5%, 1% et 2%, désignées par B, A et qualité standard, respectivement. En outre, la faible dérive de tension en fonction de la température assure une bonne stabilité sur toute la plage de température. [25]

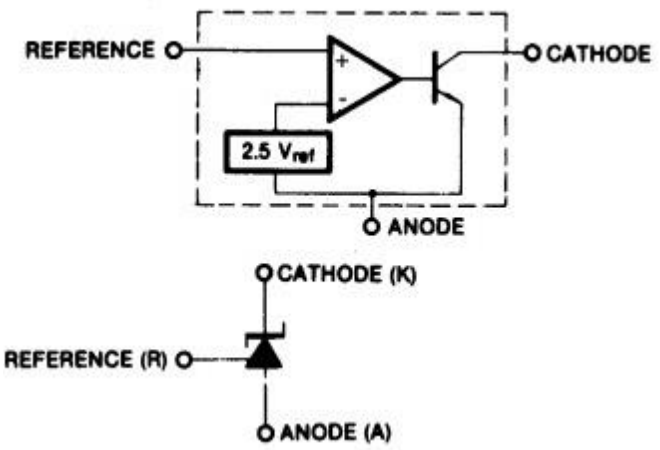


Figure 2.15: Principe du circuit TL431. [25]

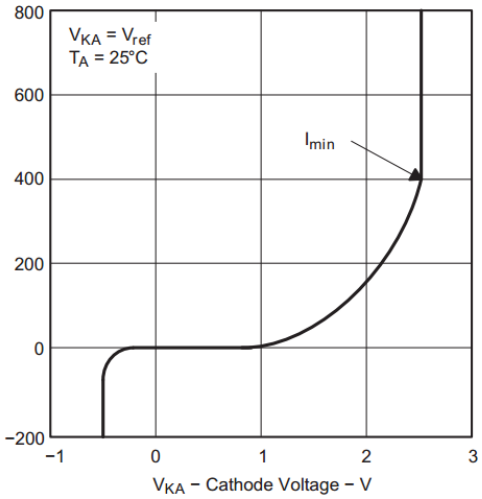


Figure 2.16: Courbe de la tension V_{AK} en fonction de la température. [25]

Montage du TL431 en référence de tension (2.50V)

C'est le montage le plus simple qui fournit une tension de 2.50V. La résistance R doit assurer un courant minimum de 1mA dans le TL431, même au plus fort courant de sortie.

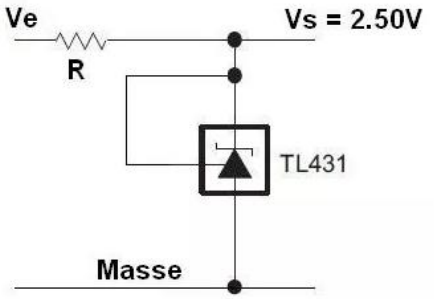


Figure 2.17: Schéma basique du TL431 avec tension de sortie fixe (2.5V). [25]

Montage du TL431 en référence de tension variable (2.5V et plus)

Dans ce montage la tension de sortie V_s est régulée selon les valeurs des résistances R_1 et R_2. Ceci suivant l'équation du diviseur de tension :

$$V_S = (1 + \frac{R_1}{R_2})V_{ref}$$

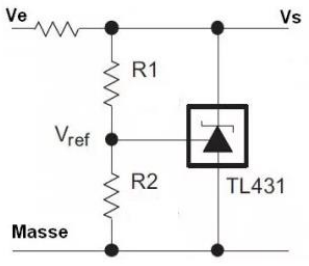


Figure 2.18: Schéma basique du TL431 avec tension de sortie variable. [25]

Le courant de sortie étant faible, il sera augmenté en ajoutant un transistor NPN.

Dans notre cas le TL431 est alimenté par la tension d'entrée de 18V. Les résistances R_1 et R_2 sont choisies de telle sorte d'avoir une tension de sortie égale à 15V suivant la formule précédente (R_1=5.1KΩ et R_2=1KΩ). Cette tension va servir comme alimentation des circuits LM311, LM310, INA118 et comme commande des MOSFET.

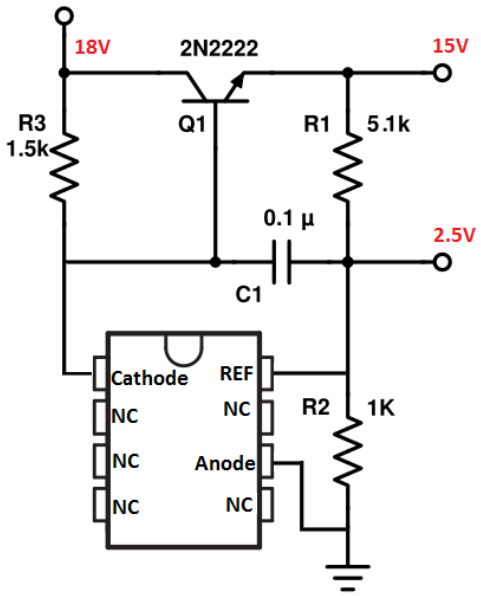


Figure 2.19: Circuit de génération des tensions 2.5V et 15V à base du TL431

2.3.5. Le montage suiveur LM310

Le LM310 est un amplificateur opérationnel connecté en interne pour être utilisé comme suiveur. Il utilise des transistors à gain élevé dans l'étage d'entrée pour obtenir un faible courant de polarisation. Ce dispositif fournit une annulation du courant d'offset. Il est conçu pour travailler de 0 à 70°C. [26]

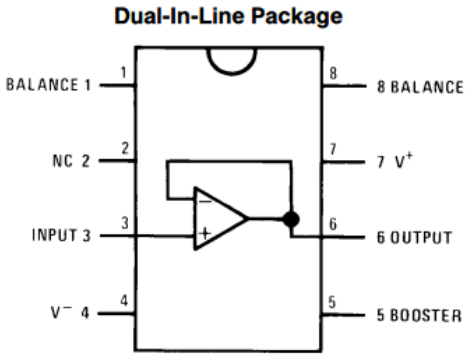


Figure 2.20: Brochage du suiveur LM310. [26]

Dans notre montage nous avons utilisé le suiveur pour pouvoir conserver et transmettre la tension de 2.5V qui provient du TL431. Sa tension de sortie va être injectée vers le diviseur de tension du montage R-2R pour commander le courant de charge choisit par l'utilisateur.

2.3.6. Le comparateur LM311

Le LM311 est un comparateur High Speed. Il est conçu pour fonctionner dans une large gamme de tensions d'alimentation, ± 18V. Les niveaux de sortie sont compatibles avec la plupart des circuits TTL et MOS. [27]

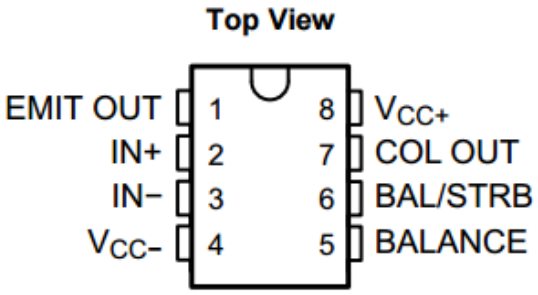


Figure 2.21: Brochage du comparateur LM311. [27]

Dans notre circuit nous avons utilisé ce comparateur pour maintenir une tension constante aux bornes de la résistance R_{20} pour que le circuit puisse travailler comme générateur à courant constant.

2.3.7. L'amplificateur différentiel INA118

Le circuit INA118 est un amplificateur d'instrumentation à usage général, offrant une excellente précision. L'alimentation de ce dernier doit être comprise entre ±1.35 to ±18 V. Le gain peut être ajusté entre 1 et 10000 avec une résistance externe unique. Le INA118 peut supporter jusqu'à ± 40 V sans dommage grâce à une protection d'entrée interne. Il est disponible dans le boîtier en plastique DIP à 8 broches et SO-8 en CMS. Sa plage de température de fonctionnement est de -40 °C à + 85 °C. [28]

Dans notre circuit, puisque la borne négative de la batterie n'est pas reliée à la masse, nous avons utilisé l'INA118 pour pouvoir lire la tension aux bornes de la batterie. Nous n'avons pas mis de résistance externe R_g entre les broches 1 et 8 afin d'avoir un gain de 1 et lire directement la tension.

La tension aux bornes de la batterie est lue par le PIC avec une entrée à convertisseur analogique numérique. Cette tension étant grande dépassant 5V (tension maximale admissible par le PIC), il impératif d'utiliser un diviseur de tension. Donc on choisit R_2 et R_3 de telle façon à ce que la tension varie de 0 à 5V au lieu de 0 à 20V. On choisit donc R_1 = 3KΩ, R_2 = 1KΩ.

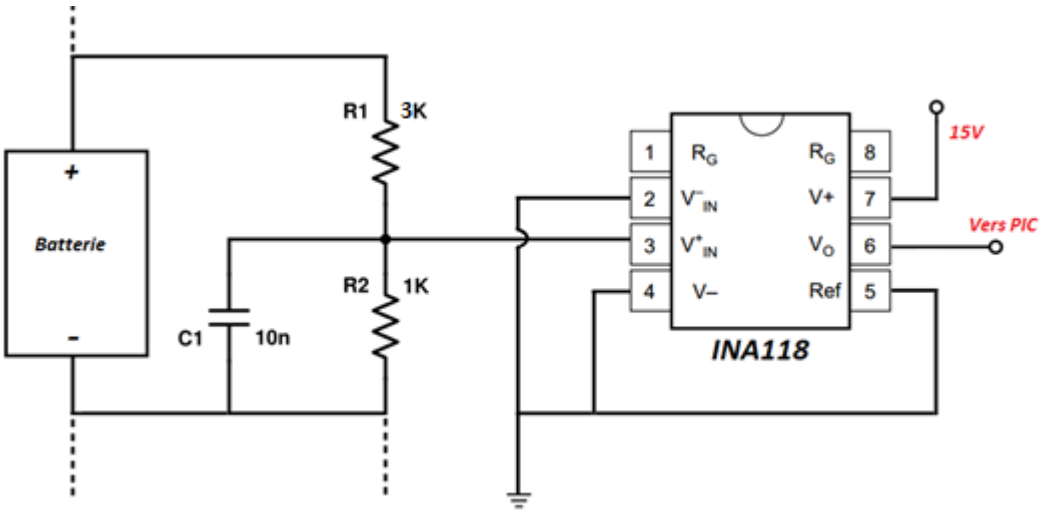


Figure 2.22: Bloc de la mesure de tension a base du circuit INA118

2.3.8. Le capteur de température LM35

Le circuit LM35 est un capteur de température à circuit intégré de précision avec une variation de tension de sortie linéairement proportionnel à la température (en Celsius). Sa sensibilité est de 10mV/°C. Le LM35 ne nécessite aucune calibration externe pour fournir une précision typique de ± ¼ % °C à la température ambiante et de ± ¾ % °C sur une plage de température entre -55 °C et 150 °C. L'alimentation recommandée de ce dernier doit s'étaler de 4V à 5.5V. Il est avantageux grâce à un très faible auto-échauffement de moins de 0.1° C dans de l'air immobile. Le LM35 utilisé est mis dans le boitier en plastique TO-92. [29]

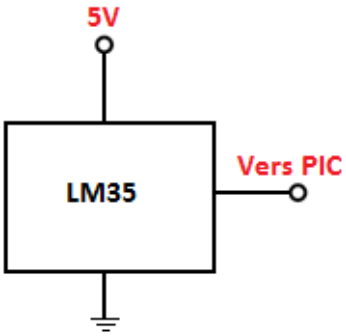


Figure 2.23: Utilisation du capteur de température LM35. [29]

2.3.9. Alimentation en courant et alimentation en tension

Le circuit de charge met en œuvre deux types d'alimentation pour assurer le fonctionnement dans les 3 phases. Cela est possible en saturant ou en bloquant les MOSFET M_1 et M_2 via les transistors Q_2, Q_3 et Q_4.

- M_1 saturé et M_2 bloqué (Q2 satuté, Q3 et Q4 bloqué): Alimentation en courant
- M_1 bloqué et M_2 saturé (Q2 bloqué, Q3 et Q4 satuté): Alimentation en tension

La commande des transitors se fait par le PIC.

2.3.9.1. Alimentation en tension et Conversion 5V à 15V

Le circuit d'alimentation en tension assure une tension constante aux bornes de la batterie suivant l'algorithme de charge. La commande du niveau de la tension comme déjà expliqué précédemment est commandée par le PWM du PIC. Pour que cela puisse fonctionner il faut envoyer approximativement 15V aux grilles des MOSFET M_2 et 0V aux grilles des MOSFET M_1. Cependant, le PIC de ne peut donner que 5V. Pour cela le circuit ci-contre a été ajouté afin de convertir les 5V aux 15V.

Quand V-ON/OFF est à 0V, le transistor Q_4 est bloqué, le courant au collecteur est nul. Ce qui va bloquer aussi Q_2. Donc la tension du collecteur de Q_2 devient nulles, d'où les MOSFET M_2 sont bloqués. Quand V-ON/OFF est à 5V, le transistor Q_4 est saturé, le courant au collecteur est non nul. Ce qui va saturer aussi Q_2. Donc la tension du collecteur de Q_2 devient égale à Vcc, d'où les MOSFET M_2 sont saturé.

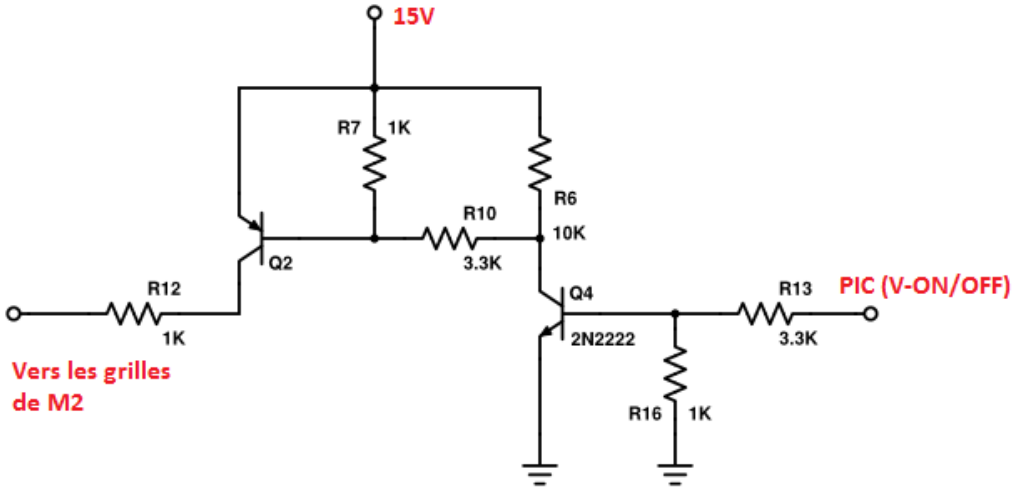


Figure 2.24: Bloc de conversion 5V vers 15V

2.3.9.2 Alimentation en courant

Pour charger la batterie à courant constant dans la 1ère phase, il faut utiliser un générateur de courant. Le circuit d'alimentation en tension assure un courant constant indépendamment de la tension et l'état de la batterie. Pour que cela puisse fonctionner et que ce générateur soit actif il faut envoyer approximativement 15V aux grilles des MOSFET M_1 (envoyer dans I_Stop 0V vers le transistor Q_3) et 0V aux grilles des MOSFET M_2 (envoyer dans V_ON/OFF 0V vers le transistor Q_4)

Le principe du fonctionnement est de maintenir une tension constante aux bornes d'une résistance à faible valeur qui est en série avec la batterie. Appliquer cette tension constante, et suivant la loi d'Ohm, est équivalent à imposer un courant constant qui va traverser la batterie. Pour maintenir cette tension constante, nous avons utilisé un comparateur.

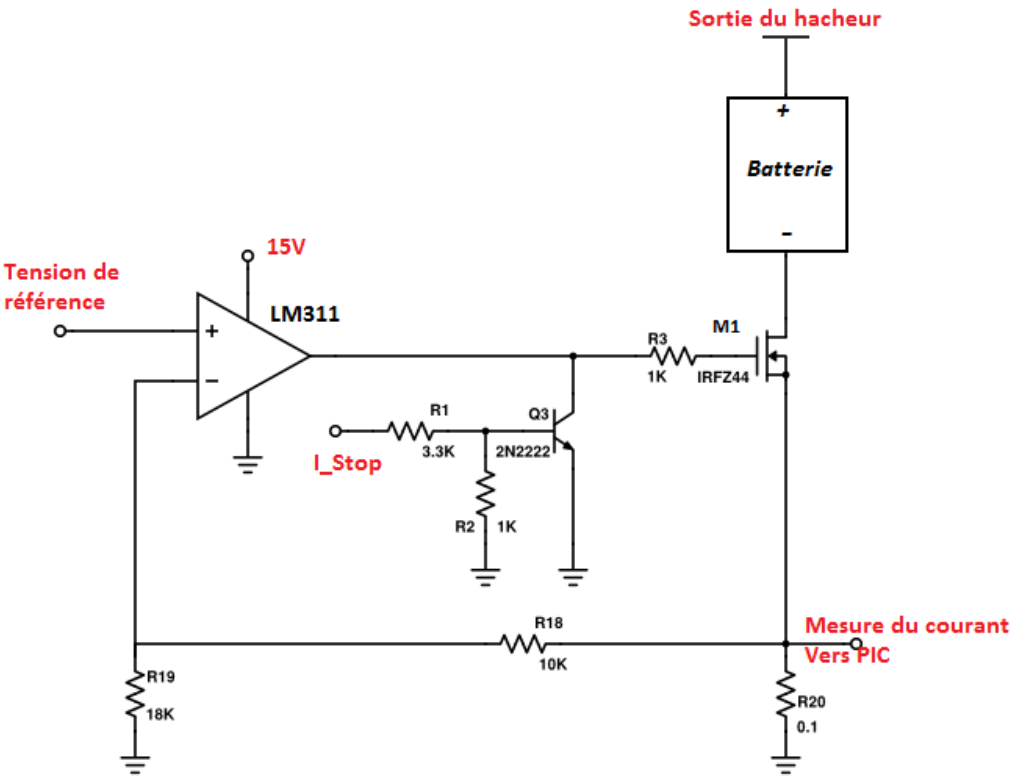


Figure 2.25: Circuit d'alimentation en courant

L'entrée non-inverseuse du comparateur est reliée à une tension fixe, quant à l'entrée inverseuse elle est reliée à la résistance R_{19} vers la masse.

Ecriture des équations :

Considérons l'amplificateur opérationnel comme idéal :

$$V_+ = V_- = V_{R_{19}} = R_{19} * i \quad \text{donc} \quad i = \frac{V_{R_{19}}}{R_{19}} \quad \textbf{(1)}$$

$$V_{R_{20}} = R_{20} * I \quad \text{d'où} \quad I = \frac{V_{R_{20}}}{R_{20}} \quad \textbf{(2)}$$

D'autre part :

$$V_{R_{20}} = (R_{18} + R_{19}) * i \ \textbf{(3)}$$

Des équations (1) et (3) :

$$V_{R_{20}} = (R_{18} + R_{19}) * \frac{V_{R_{19}}}{R_{19}} \quad \textbf{(4)}$$

Des équations (2) et (4) :

$$I = (R_{18} + R_{19}) * \frac{V_{R_{19}}}{R_{19}} * \frac{1}{R_{20}} = \frac{(R_{18}+R_{19})}{R_{19}*R_{20}} * V_{R_{19}} \quad \textbf{(5)}$$

La formule (5) indique qu'il est possible de changer le courant de charge rien que par variation de la tension $V_{R_{19}}$, c'est-à-dire la tension d'entrée à la borne non-inverseuse.

Nous avons choisi la résistance R_{20} très faible pour ne pas introduire une grande chute de tension.

Et nous avons choisi les résistances R_{18} et R_{19} très grande (par rapport à R_{20}) pour ne permettre qu'un très faible courant i qui les traverse. Donc le courant parcouru par batterie et approximativement identique à celui parcourant par la résistance R_{20} et vaut I. Cela va permettre une lecture simple et précise du courant de la batterie.

$R_{18} = 10K\Omega$

$R_{19} = 18K\Omega$

$R_{20} = 0.1\Omega$

2.3.9.3. Choix du courant de charge avec le montage R-2R

Comme démontré précédemment le courant de charge dépend de la tension d'entrée du comparateur V_+. Il est possible de varier cette tension avec un potentiomètre alimenté par la tension 2.5V (sortie du suiveur), sauf qu'il est plus adéquat de faire ce réglage par l'utilisateur sur l'interface graphique conçue au lieu de l'ajuster mécaniquement. Donc il faut utiliser un montage qui permet la variation de la résistance R_{12} suivant une commande du PIC. Au lieu d'utiliser le montage de la figure 2.26 nous avons donc choisi le montage R-2R de la figure 2.27 qui va être expliqué. Le point A étant le point de la borne non inverseuse du comparateur LM311.

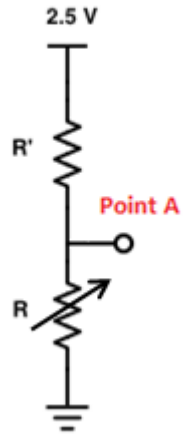


Figure 2.26: Circuit de commande du courant avec résistance variable

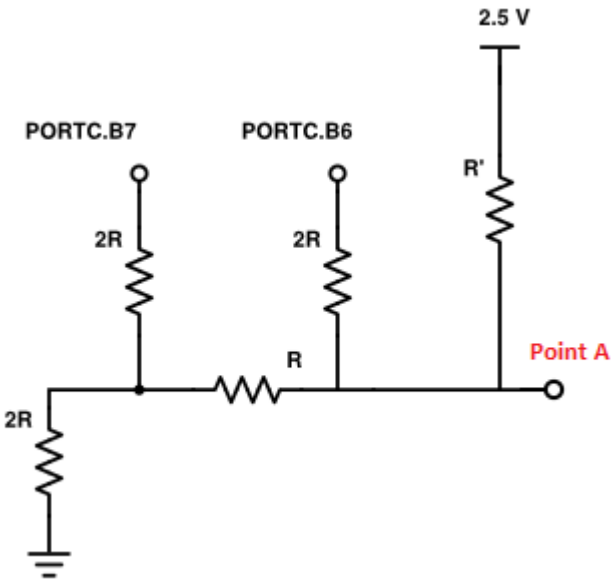
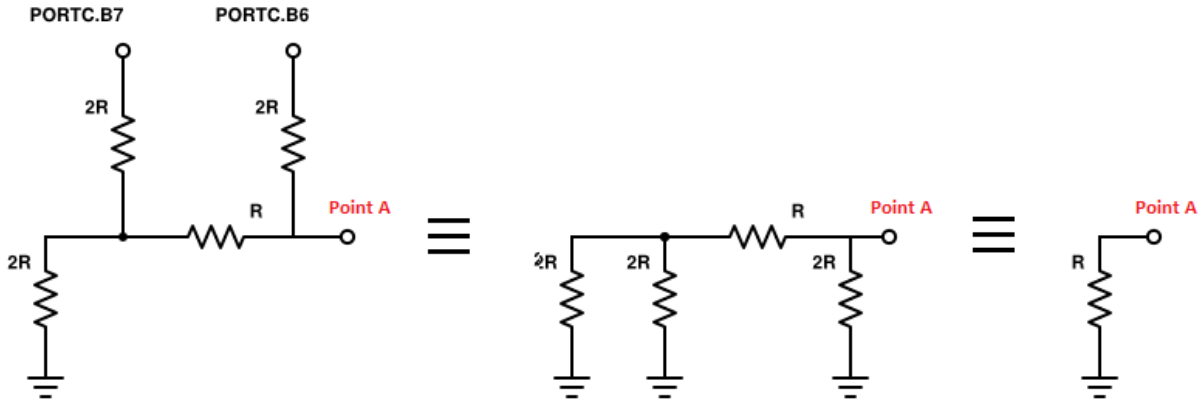


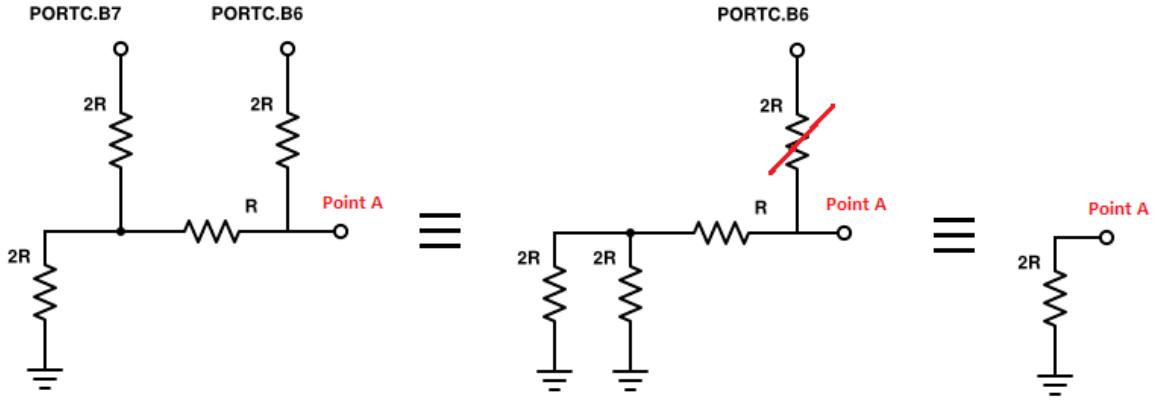
Figure 2.27: Circuit de commande du courant avec le PIC

Le montage R-2R permet de changer la résistance équivalente suivant la commande du PIC :

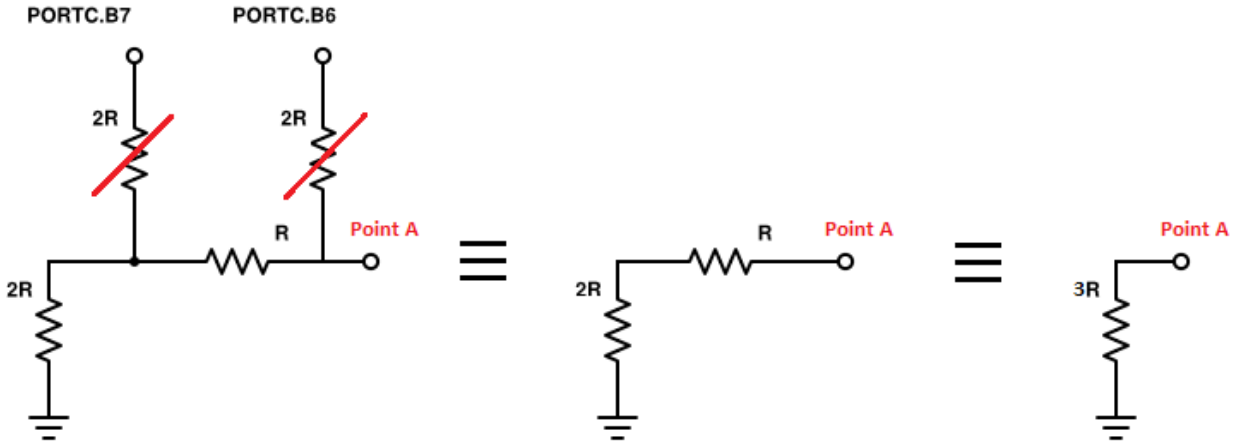
- Si les deux pins PORTC.B7 et PORTC.B6 du PIC sont à l'état bas, la résistance équivalente R_{eq} vaut R:



- Si le PORTC.B6 est à l'état haute impédance (programmé en entrée) et le pin PORTC.B7 est à l'état bas, la résistance équivalente R_{eq} vaut $2R$:



- Si les deux pins PORTC.B7 et PORTC.B6 du PIC sont à l'état haute impédance (programmés en entrée), la résistance équivalente R_{eq} vaut $3R$:



La tension V_+est égale donc, par la simple relation du diviseur de tension, à :

$$V_{R_{19}} = V_+ = \frac{R_{eq}}{R_{eq}+R'} * 2.5\ (V) \qquad \textbf{(5)}$$

Cette configuration permet le choix de 3 niveaux de courant pour charger la batterie lors de la première phase $R_{eq} = R, 2R, 3R$. De l'équation (4) et (5):

$$I = \frac{(R_{18}+R_{19})}{R_{19}*R_{20}} * V_{R_{19}} = \frac{(R_{18}+R_{19})}{R_{19}*R_{20}} * \frac{R_{eq}}{R_{eq}+R'} * 2.5 \textbf{(6)}$$

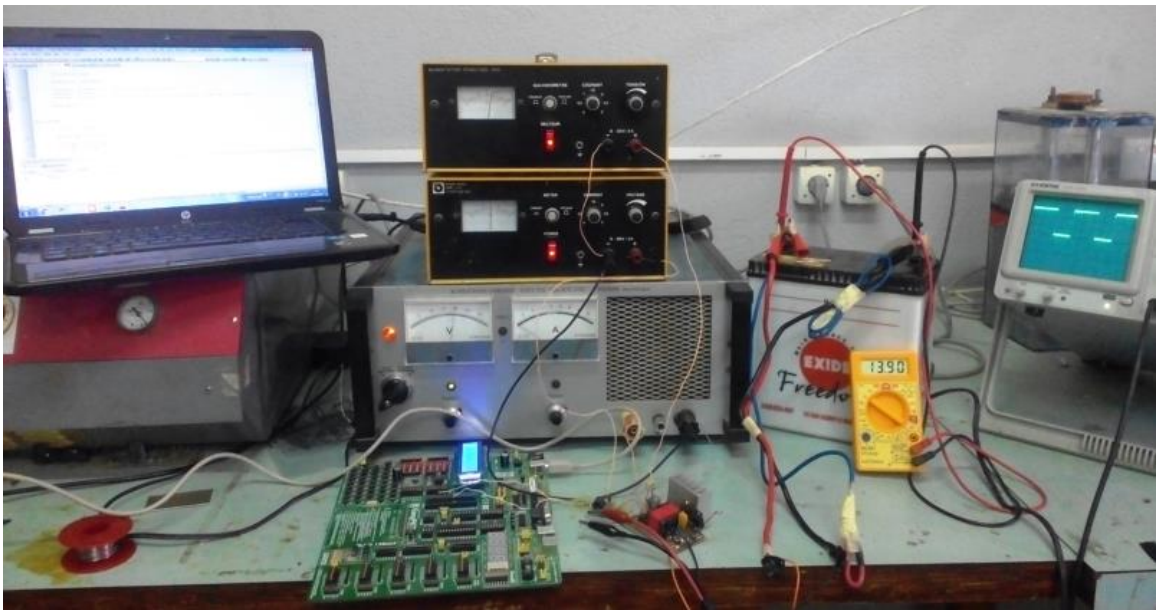


Figure 2.8 : Banc de test de la charge à tension constante

2.4. Conclusion

Le circuit de charge conçu est constitué de plusieurs blocs fonctionnels gérés par le PIC18F2550. Cela a pour but de mettre en œuvre l'algorithme de charge à trois phases. Dans ce chapitre nous avons présenté le fonctionnement global du circuit avec son schéma synoptique. Nous avons expliqué les détails de chaque bloc et nous avons justifié le choix des composants choisis.

Chapitre 3

Programmation du PIC et algorithme de charge

3.1. Introduction

Le cerveau de notre circuit réside dans le microcontrôleur. Ce dernier mesure tous les paramètres de la batterie, gère toutes les phases de charge, et échange les données avec l'utilisateur pour un but de commande et de visualisation.

Dans ce chapitre, nous allons discuter de l'algorithme de charge choisi et présenter son organigramme. Nous allons aussi montrer toutes les étapes de configuration et de programmation du PIC. Nous allons éclaircir le choix des fréquences d'oscillation, la génération du PWM, du TIMER, l'utilisation du convertisseur analogique numérique et enfin la gestion de l'affichage LCD.

3.2. Algorithme de charge

L'algorithme de charge choisi est celui à trois phases :

Première phase: nous devons travailler à courant constant avec une tension de sortie qui augmente. Le courant de charge est choisi par l'utilisateur via l'interface graphique. Cette phase se poursuit jusqu'à ce que la tension batterie atteigne la tension maximale. Cette tension est donnée par le constructeur et liée à la technologie de la batterie. Cependant, la valeur typique que nous allons prendre à 25°C est de 14.4V. Cette tension dépend de la température et varie avec un facteur négatif de –5 mV chaque 1 °C par cellule. C'est-à-dire 30 mV /°C pour une batterie de 12V [30]. Donc l'équation reliant la température avec la tension maximale est:

$$\boldsymbol{V_{Max} = 14.4 - 0.030 * (T - 25)} \quad \boldsymbol{(V)}$$

Seconde phase : Dès que la tension maximale est atteinte (la batterie dans ce cas est chargée à 80% approximativement), nous aiguillons vers la charge à tension constante. Nous devons conserver la tension constante (la tension maximale), pendant que le courant de la batterie diminue. Si le courant de la batterie diminue au-delà de C/100, la batterie devient chargée à 100%, et on procède à l'application de la tension de maintien.

Troisième phase : Dans cette phase nous appliquons une tension constante inférieure à la tension maximale qui vaut 13.8 V pour maintenir la batterie chargée et compenser l'autodécharge. Cette tension dépend également de la température et nous avons pris le même facteur que pour la tension maximale.

$$\boldsymbol{V_{Float} = 13.8 - 0.030 * (T - 25)} \quad \boldsymbol{(V)}$$

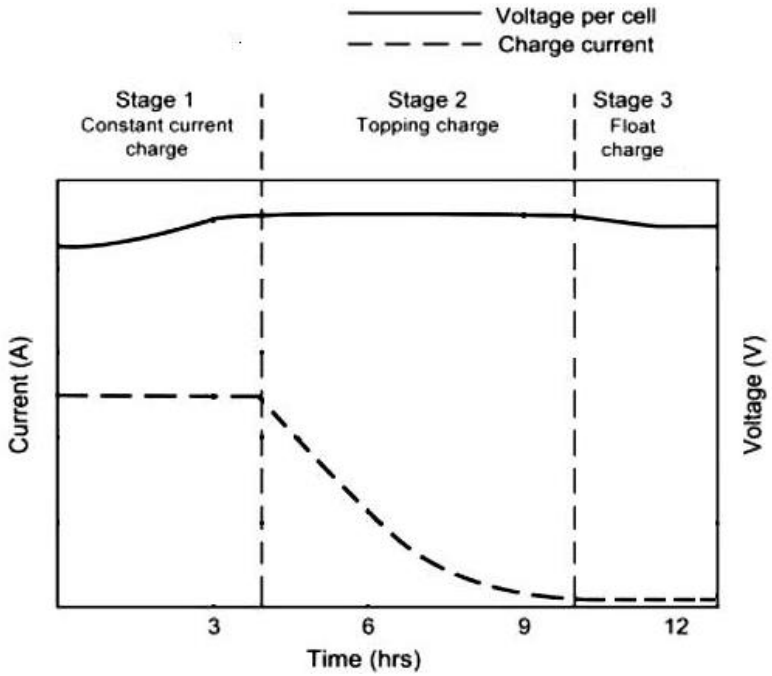


Figure 3.1: Courbe de la tension et du courant dans une charge à trois phases. [13]

3.3. Organigramme de charge (fonction main())

Suivant l'algorithme de charge choisi, le circuit réalisé et les considérations pratiques nous avons conçu l'organigramme de la figure suivante. Cet organigramme représente la fonction principale (fonction main) qui va être exécuté par le PIC pour effectuer la charge de la batterie.

Remarque

La fonction charge () et les autres fonctions seront décrites par la suite.

3.3.1. Fonction Main ()

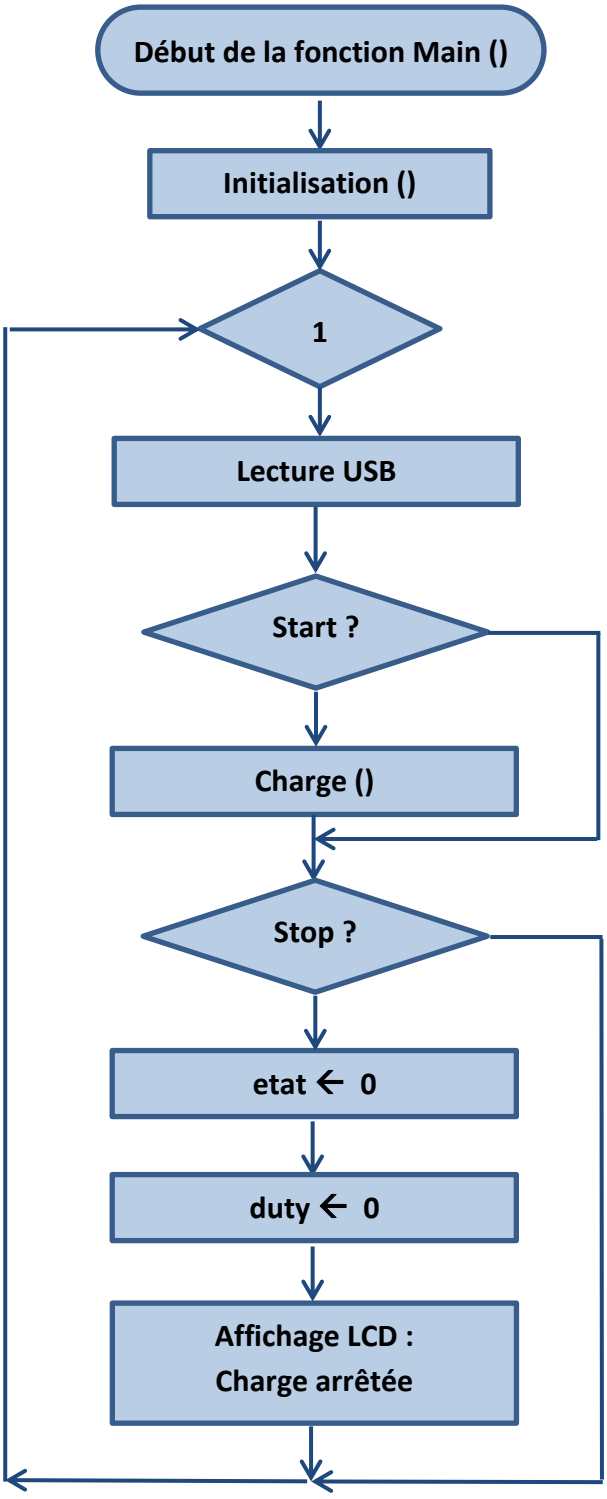


Figure 3.2: Organigramme de gestion de charge implémenté dans le PIC

3.3.2. Explication générale de l'organigramme

Au début le PIC procède par la fonction ***initialisation.*** Puis, il va tester si l'utilisateur a cliqué sur le bouton ***« Start »*** de l'interface graphique; si c'est le cas il va exécuter la fonction ***charge*** ou il va lire les données essentielles sur la batterie: la tension, le courant et la température avec les fonctions ***lectTens, lectCour, lectTemp***. Ensuit il effectue les traitements nécessaires pour la gestion des phases de charge. Egalement, il va envoyer les données à l'afficheur LCD et à l'interface graphique avec les fonctions ***communicationTens, communicationCour, communicationTemp***.

Si l'utilisateur clique sur le bouton « Stop » de l'interface graphique, la charge va s'arrêter en désactivant le PWM (rapport cyclique nul) et en revenant vers l'état initial (***etat*** = 0). L'afficheur LCD va afficher « Charge arrêtée ».

Ce processus va se boucler durant toute l'alimentation du PIC.

Remarque

Les détails de programmation et les corps de fonctions seront détaillés par la suite.

3.4. Configurations et programmation du PIC

3.4.1. Configuration des fréquences de travail

Les microcontrôleurs PIC18F2455/2550/4455/4550 possèdent différents mode d'utilisation des oscillateurs pour la gestion des horloges dans le système. L'ajout du module USB, avec ses exigences uniques pour une source d'horloge stable, rendent nécessaire de fournir une source d'horloge séparée qui est conforme à la fois avec l'USB à faible vitesse (low speed) et les spécifications pleine vitesse (full speed). Pour répondre à ces exigences, les PIC18F2455/2550/4455/4550 comprennent un nouveau branchement pour fournir une horloge de 48 MHz pour le fonctionnement USB pleine vitesse. Etant donné qu'il est entraîné à partir de la source l'oscillateur primaire, un système supplémentaire de pré-diviseurs (prescalers) et post-diviseurs (postscalers) a été ajouté afin de pouvoir recevoir une large gamme de fréquences.

Les PIC18F2455/2550/4255/4550 comprennent une boucle à verrouillage de phase (PLL). Ceci est prévu spécifiquement pour les applications USB avec des oscillateurs à faible vitesse et peut également être utilisée comme une horloge du microcontrôleur. La PLL est conçue pour produire une horloge de référence de 96 MHz fixe à partir d'une entrée fixe de 4 MHz (elle travaille comme multiplicateur de fréquence). La sortie pourrait alors être divisée et utilisée pour l'USB et/ou pour l'horloge du système. Puisque la PLL comporte une entrée et une sortie à fréquence fixe, une option à huit prescalers est prévue (1, 1/2, 1/3, 1/4, 1/5, 1/6, 1/10, 1/12) pour faire une correspondance entre la fréquence de l'oscillateur d'entrée

(quartz) et celle de l'entrée de la PLL. Il y a également une option de postscalers séparée pour obtenir l'horloge de travail du système à partir de la PLL. Cela permet au périphérique USB et au système d'utiliser la même entrée de l'oscillateur et fonctionner encore à des vitesses d'horloge différentes. Contrairement aux postscalers mis en place dans les modes XT, HS et EC, les valeurs disponibles sont : 1/2, 1/3, 1/4 et 1/6 à la sortie de la PLL. [17]

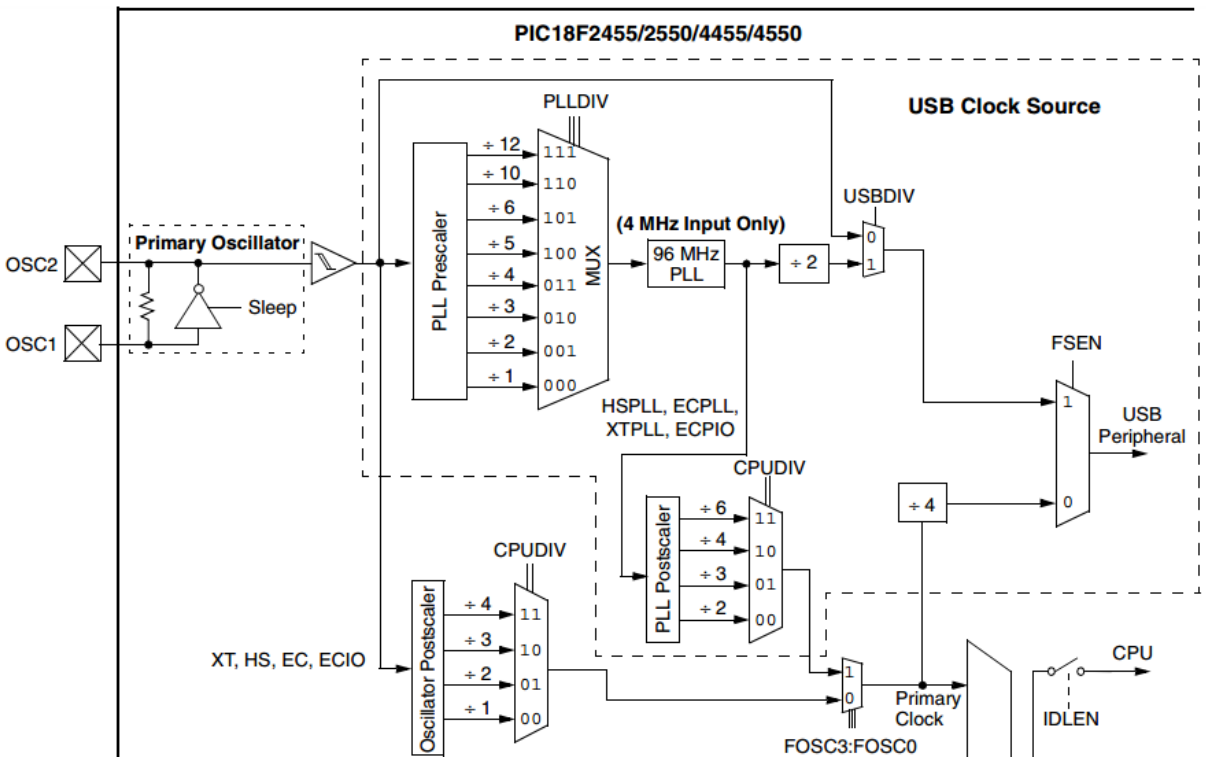


Figure 3.3: Schéma du système d'horloge à l'intérieur du PIC18F2550. [17]

Pour notre cas nous avons utilisé un oscillateur de quartz d'une fréquence de 8MHz et nous avons travaillé en mode USB Full speed, et enfin nous avons choisi que notre CPU travaille avec 48MHz. Donc pour cela nous avons utilisé le mode HSPLL. Nous avons choisi le prescaler à 2 (8MHz / 2 = 4 MHz) pour avoir nos 4 MHz à l'entrée du la PLL. La PLL va générer une horloge de 96MHz qui va être divisée par 2, ce qui donne 48MHz. Cette dernière sera la fréquence de travail du l'USB. Quant à la fréquence du CPU elle sera également 48MHz. Elle est obtenue après division des 96MHz sortie de la PLL par le postscaler qui est égale à 2.

Toute cette configuration se fait d'une manière graphique avec les propriétés du projet MikroC Pro. Les voilà les configurations à faire :

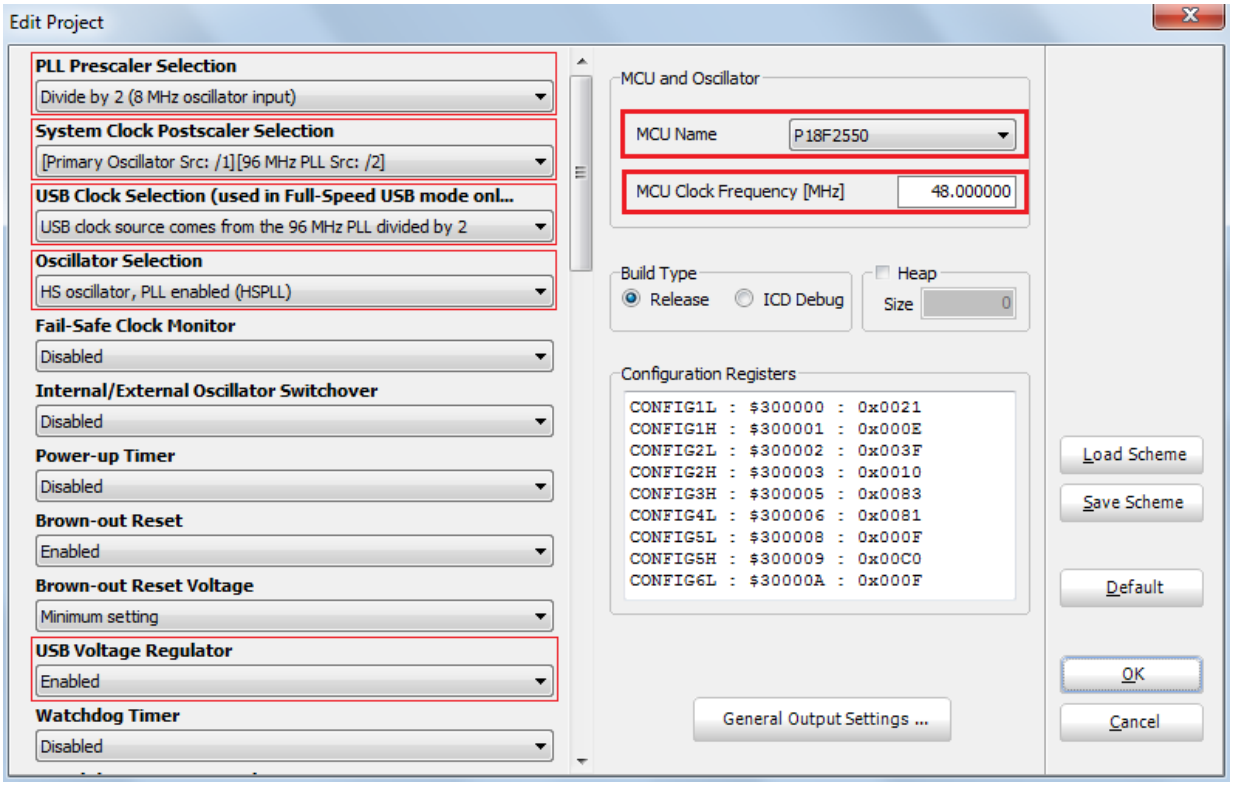


Figure 3.4: Configurations du projet sous MikroC

3.4.2. Configuration du PWM

Dans le PIC18F2550 il existe deux sorties de PWM: CCP1 (relié à RC2) et CCP2 (relié soit à RC1 ou RB3 suivant le bit **CCP2MX** du registre **CONFIG3H**). Le module **CCPx** est géré par le registre **CCPxCON** (x = 1,2). Le mode PWM est relié au **TIMER2**. La fréquence du signal impulsionnel est établie en écrivant sur le registre PR2. Pour obtenir un rapport cyclique spécifié, il faut écrire le MSB dans le CCPRxL, et les 2 bits LSB sur CCPxCON<5:4>. Quant aux facteurs diviseur de fréquence et l'activation du TIMER2 se font par le registre T2CON. Toutes ces opérations seront détaillées par la suite.

- **La période du PWM**

La période de PWM est spécifiée par écriture sur le registre PR2. La période de PWM peut être calculée selon la formule suivante:

$$\boldsymbol{T_{PWM} = [(PR2) + 1] * 4 * T_{OSC} * (TMR2\ Prescal\ value)}$$

$\boldsymbol{T_{PWM}}$**:** Période du PWM

$\boldsymbol{PR2}$**:** C'est un registre 8 bit

$\boldsymbol{T_{OSC}}$**:** C'est la période de l'horloge de travail du microcontrôleur (pas nécessairement celle du quartz d'entrée)

$\boldsymbol{TMR2\ Prescal\ value}$**:** C'est la valeur du facteur diviseur de fréquence spécifié dans **T2CON<1:0>** (1, 4 ou 16)

Donc pour une valeur de période donnée il faut chercher la bonne valeur à écrire dans le PR2 ainsi que choisir le facteur diviseur qui convient. Pour une période de PWM souhaitée, un facteur diviseur choisi, et une période d'oscillation du MCU connue, l'équation devient:

$$\boldsymbol{PR2 = \frac{T_{PWM}}{4 * T_{OSC} * (TMR2\ Prescal\ value)} - 1}$$

Comme il est plus pratique de travailler en fréquence, l'équation devient :

$$\boldsymbol{PR2 = \frac{F_{OSC}}{4 * F_{PWM} * (TMR2\ Prescal\ value)} - 1}$$

Avec

$\boldsymbol{F_{OSC}}$ **:** C'est la fréquence de l'horloge de travail du microcontrôleur.

$\boldsymbol{F_{PWM}}$ **:** C'est la fréquence du PWM.

- **Rapport cyclique**

Le rapport cyclique du PWM est spécifié en écrivant sur le registre CCPRxL et aux deux bits CCPxCON <5: 4>. Le CCPRxL contient les huit bits MSB et CCPxCON <5: 4> contiennent les deux bits LSB. Cela permettra une résolution étendue jusqu'à 10 bits. Cette valeur de 10 bits est représentée par CCPRxL: CCPxCON <5: 4>. L'équation suivante est utilisée pour calculer le rapport cyclique du PWM (il a la même unité du $\boldsymbol{T_{OSC}}$).

$$\boldsymbol{T_{PWM-ON} = (CCPRxL{:}CCPxCON) * T_{OSC} * (TMR2\ Prescale\ Value)}$$

Cela veut dire que pour une fréquence d'oscillation du MCU connue et un facteur diviseur choisi, on doit écrire correctement CCPRxL et les deux bits CCPxCON <5: 4>. L'équation précédente sera réarrangée comme suit :

$$(CCPRxL{:}CCPxCON) = \frac{T_{PWM-ON}}{T_{OSC} * (TMR2\ Prescale\ Value)}$$

En fréquence, l'équation devient:

$$\boldsymbol{(CCPRxL{:}CCPxCON) = \frac{\alpha * F_{OSC}}{F_{PWM} * (TMR2\ Prescale\ Value)}}$$

Avec $\boldsymbol{\alpha}$ un réel entre sans unité entre 0 et 1 résignant le rapport cyclique

- **Activation du PWM avec CCPxCON**

CCPxCON (x = 1,2) sont deux registres standards pour le contrôle des modules de CCPx. Les 4 bits du LSB servent à activer/désactiver et configurer les module du CCPx. Dans notre cas pour activer le PWM il faut affecter un 0x11xx à ses 4 bits LSB. Les bits 4 et 5 sont les bits LSB du rapport cyclique comme expliqué précédemment.

- **Activation du TIMER2 avec T2CON**

Le module TIMER2 est contrôlé pas le registre T2CON. Ce dernier active ou désactive le TIMER2 ainsi que configure les valeurs diviseur de fréquence prescaler et postscaler. Les voilà les détails du registre T2CON.

Bit 7: Non implémenté

Bit 6-3 (**T2OUTPS3:T2OUTPS0)**: Bits de sélection du Postscaler liés au Timer2

0000 = 1:1 Postscale

0001 = 1:2 Postscale

•

•

•

1111 = 1:16 Postscale

Bit 2 (TMR2ON): Timer2 On bit

1 = Timer2 est activé

0 = Timer2 est désactivé

Bit 1-0 (T2CKPS1:T2CKPS0): Bits de sélection du Prescale liés au Timer2

00 = Prescaler égale à 1

01 = Prescaler égale à 4

1x = Prescaler égale à 16

En résumé pour configurer le PWM il faut suivre les étapes suivantes:

1- Fixer la fréquence du signal PWM en écrivant sur le registre PR2
2- Fixer le rapport cyclique en écrivant sur le registre CCPRxL et les 2 bits CCPxCON<5:4>.
3- Mettre le pin correspondant au CCPx en sortie en écrivant un 0 au registre TRIS.
4- Fixer la valeur du prescale puis activer le TIMER2 en écrivant sur le T2CON.
5- Configurer le module CCPx pour l'opération de PWM en écrivant 0x11xx sur CCPxCON<3:0>. [17]

Pour notre cas, la fréquence du PWM étant 50KHz, la fréquence d'oscillation est de 48MHz, le prescale étant choisi à 4 et en utilisant les équations, voici les configurations qu'il faut respecter :

$$\boldsymbol{PR2 = \frac{F_{osc}}{4 * F_{PWM} * (TMR2\ Prescal\ value)} - 1 = \frac{48000}{4 * 50 * 4} - 1 = 59 = 0b00111011}$$

L'activation du TIMER2 et la fixation du prescale se fait par en écrivant T2CON = 0b00000101

L'activation du mode PWM et la fixation des 2 bits LSB du rapport cyclique se fait en écrivant: CCP1CON = 0b00101100. Quant au MSB il se fera par programme d'une manière adaptative suivant la tension de sortie voulue.

En résumé pour une configuration complète du PWM, voici le code qu'il faut utiliser

```
T2CON = 0b00000101;  // prescaler + activer le TMR2;

PR2 = 0b00111011;  // Pour 50KHz

CCPR1L = 0b00100101;   // MSB du rapport cyclique

CCP1CON = 0b00101100;  // LSB du rapport cyclique + activation du mode PWM
```

Remarque

Pour la fixation des 2 bits LSB du rapport cyclique, nous avons choisi de la mettre par défaut égale à 2 (0b10) et se contenter uniquement pour l'écriture du MSB qui est plus significatif.

3.4.3. Configuration et utilisation du convertisseur analogique-numérique

Dans le PIC18F2550 il y a 10 entrées reliées au convertisseur analogique-numérique. Ce module permet une conversion du signal analogique en entrée vers un signal numérique de 10 bits. Cela donne une résolution de $5/2^{10}$ = 4.48 mV.

La fonction du port I/O (numérique ou analogique) se fait par l'écriture sur les 4 bits LSB du registre ADCON1 suivant le tableau du Data Sheet ([17]).

Pour les fonctions basiques de gestion de la conversion, nous avons utilisé les différentes fonctions incluses dans la librairie ADC de MikroC Pro. Voilà les fonctions utilisées :

- **La fonction ADC_Init**

Cette fonction initialise le module ADC interne du PIC à travailler avec l'oscillateur RC interne.

Prototype : **void ADC_Init();**

- **La fonction ADC_Get_Sample**

Elle prend en argument le numéro du canal ou le signal analogique est entré. Elle retourne un nombre de type unsigned correspondant à la valeur du signal lu. Elle requiert l'exécution de la fonction ADC_Init et la configuration appropriée du registre TRIS.

Prototype : **unsigned ADC_Get_Sample(unsigned short channel);**

- **La fonction ADC_Read**

Cette fonction d'une part initialise le module ADC à travailler avec l'oscillateur RC interne, et d'autre part, elle retourne un nombre de type unsigned correspondant à la valeur du signal lu en spécifiant le canal souhaité en argument. Elle requiert la configuration appropriée du registre TRIS.

Prototype : **unsigned ADC_Read(unsigned short channel);**

3.4.4. Configuration et utilisation de l'afficheur LCD

L'afficheur LCD utilisé est sous forme d'une matrice de 2 lignes et 16 colonnes, donnant ainsi la possibilité d'afficher 32 caractères. Chaque caractère est sous forme de 35 pixels (7 pixels en lignes et 5 pixels en colonnes). La communication avec le microcontrôleur se fait avec une interface parallèle. L'interface se compose des pins suivants:

- **Un pin de sélection du registre (RS) :** Il contrôle où dans la mémoire de l'écran LCD nous écrivons les données: Soit sélectionner le registre de données, qui contient ce qui va être affiché dans l'écran, soit sélectionner le registre d'instruction, qui est l'endroit où le driver de l'écran LCD cherche les instructions à exécuter par la ensuite.
- **Un pin Lecture / écriture (R/W)** : Il sélectionne le mode lecture ou en mode écriture.
- **Un pin d'activation (Enable) :** Il permet d'écrire dans les registres.
- **8 pins de données (D0 -D7) :** Les états de ces pins (haut ou bas) déterminent les données que nous écrivons sur un registre pour l'affichage (en mode écriture), ou les valeurs que nous lisons (en mode lecture).
- Les deux broches **+ 5V et GND** pour l'alimentation.
- **Le pin Vo :** Il est utilisé pour régler le contraste de l'écran.
- **LED + et LED - :** Pour allumer/ éteindre le rétro-éclairage LED. [31]

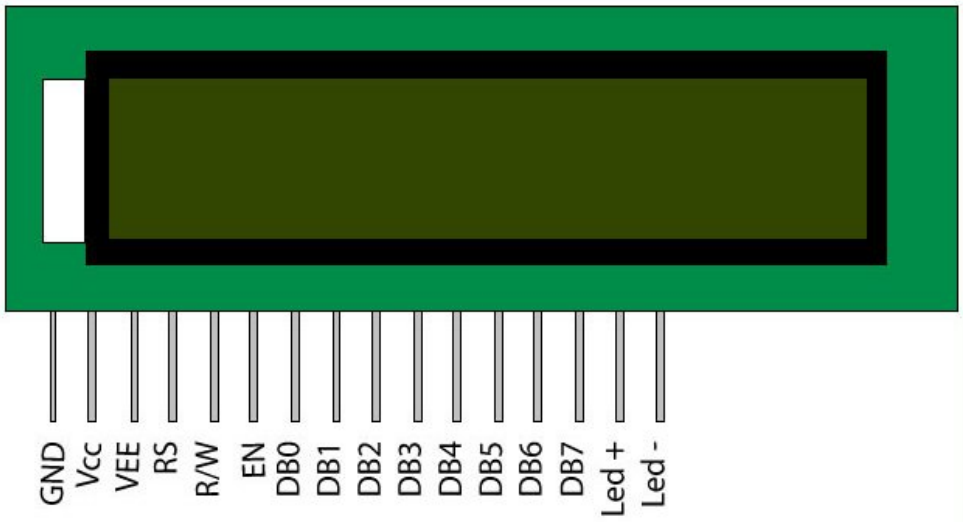


Figure 3.5: Brochage de l'afficheur LCD. [32]

Pour la programmation, MikroC PRO For PIC fournit une bibliothèque dédiée à la communication avec les afficheurs LCD (qui sont conformes avec le driver HD44780) par le biais de l'interface 4 bits. Parmi les fonctions de la bibliothèque, qui ont été utilisées dans notre projet, sont les suivantes :

- **La fonction Lcd_Init**

Cette fonction initialise l'afficheur LCD sans avoir aucun argument et ne retourne aucun résultat. Néanmoins, elle requiert la déclaration des variables expliquées précédemment.

Prototype : **void Lcd_Init();**

- **La fonction Lcd_Out**

Elle permet l'écriture sur l'afficheur LCD en spécifiant en argument le numéro de la ligne (row) et le numéro de la colonne (column) de la position de départ ainsi que le texte à écrire (text). Cette fonction requiert l'initialisation de l'afficheur LCD avec la fontion Lcd_Init.

Prototype : **void Lcd_Out(char row, char column, char *text);**

- **La fonction Lcd_Out_Cp**

Elle permet l'écriture sur l'afficheur LCD en prenant la position actuelle du curseur comme position du départ de l'écriture. Elle n'a besoin en argument que le texte à écrire (text). Cette fonction requiert, également, l'initialisation de l'afficheur LCD avec la fonction Lcd_Init.

Prototype : **void Lcd_Out_Cp(char *text);**

3.5. Listes des variables

unsigned char readbuff[64] : C'est un tableau de 64 caractères il est inscrit dans la mémoire dans la case 0x500. Il travaille comme buffer, dans lequel on enregistre momentanément les données qu'on reçoit de l'USB.

unsigned char writebuff[64] : C'est un tableau de 64 caractères il est inscrit dans la mémoire dans la case 0x540. Il travaille comme buffer, dans lequel on enregistre momentanément les données qu'on veut envoyer vers l'USB.

FloatTemp, FloatTens, FloatCour : Sont les variables dans lesquelles on stocke la tension, le courant et la température.

float tension_max : Cette variable de type float représente la tension maximale avec laquelle on effectue la charge dans la 2ème phase.

float tension_maintien : Cette variable de type float représente la tension de maintien avec laquelle on effectue la charge dans la 3ème phase.

float courant_min : Cette variable de type float représente le courant minimal, au-delà duquel on passe de la 2ème à la 3ème phase de charge.

float aa: Cette variable de type float représente le coefficient directeur de la fonction liant la tension de sortie et le rapport cyclique. Sa valeur, calculée après calibration, est égale à 0.1789.

float bb: Cette variable de type float représente l'ordonnée à l'origine de la fonction liant la tension de sortie et le rapport cyclique. Sa valeur, calculée après calibration, est égale à -0.8092.

float coef_temp : Cette variable de type float représente le coefficient de température, qui sera utilisé pour déterminer la tension maximale dans la $2^{ème}$ et $3^{ème}$ phase.

float TensMaxRef et **float TensFloatRef** : sont des variables de type float qui représentent la tension de référence de charge de la batterie dans la $2^{ème}$ et $3^{ème}$ phase. Ils sont données pour la température de référence **TempRef.**

char CMD_Tens [8], char CMD_Cour [8], char CMD_Temp [8], char CMD_Msg [8] : Ce sont des tableaux de 8 caractères avec les valeurs: "0__tensi", "0__cour" et "0__tempe", "0__messa" respectivement. Ils sont utilisés comme codes de commande pour désigner la tension, le courant, la température ou un message dans la trame de bits envoyée dans l'USB.

float duty : Cette variable de type float qui représente le rapport cyclique.

int etat : C'est une variable entière qui conserve l'état de la charge.

- etat = 0: Début.
- etat = 1: Charge en courant
- etat = 2: Charge en Tension (Absorption)
- etat = 3: Charge en Tension (maintien)

int timer : C'est une variable entière qui conserve le temps pour effectuer les instructions comme montré dans l'organigramme de la fonction ***charge***.

char cnt : une variable de type char utilisé comme indice dans les boucles for.

bit boo : C'est une variable booléenne pour montrer si l'utilisateur a cliqué sur « start » dans l'interface graphique (boo = 0) ou pas encore (boo = 1)

3.6. Programmation des fonctions

3.6.1. La fonction Initialisation ()

Ça consiste à définir les ports que nous allons utiliser, initialiser et établir les configurations du PWM, initialiser le TIMER0, initialiser le convertisseur analogique numérique, activer la communication USB, initialiser l'afficheur LCD, ensuite attendre jusqu'au déclenchement de la charge par l'utilisateur dans l'interface graphique.

- Initialisation des ports :

Les ports utilisés sont : le **PORTA**, le **PORTB** et le **PORTC**. Le **PORTA** contenant le convertisseur analogique sera donc utilisé en entrée pour mesurer la tension, le courant et la température. Les 3 premiers bits sont utilisés. Le **PORTB** sera programmé en sortie car il sera relié à l'afficheur **LCD** (les 6 premiers bits du LSB) et au programmateur **ICD** (les 2 bit du MSB). Les bits du PORTC seront programmés en sortie (à l'exception du **RC4** et **RC5** qui sont les D+ et D- de l'USB). **RC0** et **RC1** commandent le type d'alimentation, le **RC2** jouera le rôle de la sortie du signal PWM qui commandera le MOSFET, et enfin **RC6** et **RD7** commandent le niveau du courant.

```
TRISA = 0XF;

TRISB = 0;

PORTB = 0;

TRISC &= 0X38; // mettre uniquement les 3 premiers bits + Bit 6 et 7 en sortie et les autres par défaut (à cause du D+ et D-)

PORTC = 0;
```

- Configurations du PWM :

La configuration complète du PWM a été expliquée dans la partie 3.4.2. (Configuration du PWM). Dans l'initialisation on se contente uniquement par mettre le prescaler, activer le TIMER2 et écrire le PR2 selon la valeur de 50 KHz. La suite (activation du mode PWM et écrire le rapport cyclique) se fera au cours du traitement.

```
T2CON = 0b00000101;  // prescaler + activer le TMR2;

PR2 = 0b00111011;  // Pour 50KHz
```

- Initialisation du convertisseur analogique numérique :

Tous d'abord il faut configurer les 4 premiers pins du PORTA pour qu'ils fonctionnement comme entrées analogiques. Et ceci en écrivant sur le registre **ADCON1**. Ensuite nous utilisons la fonction **ADC_Init** qui va initialiser le module ADC.

```
ADCON1 = 0x0B;

ADC_Init(); // initialiser le module ADC
```

- Activation de la communication USB HID :

L'activation de la communication USB HID se fait par la fonction **HID_Enable** en introduisant en argument les deux buffers **readbuff** et **writebuff.** (Voir chapitre suivant pour plus de détails sur les fonctions de l'USB)

```
HID_Enable (&readbuff, &writebuff);      /* Activer la
communication USB HID */
```

- Initialisation l'afficheur LCD :

```
LCD_Init (); // Initialisation du module LCD

LCD_Cmd (_LCD_CURSOR_OFF); // Désactiver le curseur

LCD_Cmd (_LCD_CLEAR); // Supprimer le contenu du
```

3.6.2. La fonction lectTens ()

La fonction lectTens a pour objectif la lecture de la tension en faisant appel à la fonction ADC_Read et stocker la valeur retournée dans la variable AD_Read_Tens du type unsigned. La valeur retournée étant une séquence de 10 bits venant directement du convertisseur analogique numérique, il faut la convertir en tension avec l'équation de la règle de trois, puis récupérer sa valeur réelle (avant le diviseur de tension).

```
AD_Read_Tens  = ADC_Read(1);  // Lecture

FloatTens =  AD_Read_Tens; // Conversion Unisigned long => float

FloatTens = FloatTens  * 5 / 1023 ; //Conversion bits => tension

FloatTens = FloatTens  * 3 ;  //Récuperer la tension réelle (diviseur de tension)
```

3.6.3. La fonction lectCour ()

La fonction lectCour a pour objectif la lecture du courant en faisant appel à la fonction ADC_Read et stocker la valeur retournée dans la variable AD_Read_Cour du type unsigned. La valeur retournée étant une séquence de 10 bits venant directement du convertisseur analogique numérique, il faut la convertir en tension avec l'équation de la règle de trois. Cette tension est l'image du courant puisqu'elle est aux bornes de la résistance série de 0.1 Ω (loi d'Ohm).

```
AD_Read_Cour = ADC_Read(2); // Lecture

FloatCour =  AD_Read_Cour; // Conversion Unisigned long => float

FloatCour = FloatCour  * 5 / 1023 ; //Conversion bits => tension

FloatCour = FloatCour * 10;  //Récuperer le courant réel (Conversion Tension
=> Courant dans la résistance)
```

3.6.4. La fonction lectTemp ()

La fonction lectTemp a pour objectif la lecture de la température en faisant appel à la fonction ADC_Read et stocker la valeur retournée dans la variable AD_Read_Temp du type unsigned. La valeur retournée étant une séquence de 10 bits venant directement du convertisseur analogique numérique, il faut la convertir en tension avec l'équation de la règle de trois. Cette tension est l'image de la température puisque le capteur LM35 est linéaire avec une sensibilité de 10mV /°C.

```
AD_Read_Temp = ADC_Read(3); // Lecture

FloatTemp =  AD_Read_Temp; // Conversion Unisigned long => float

FloatTemp = FloatTemp  * 5000 / 1023 ; //Conversion bits => tension

FloatTemp = FloatTemp  / 10 ; // Récuperer la temprérature réelle  (sensibilté
du capteur: 10 mV / C)
```

3.6.5. Les fonctions de "communications"

Ces fonctions ont pour objectif l'affichage de la tension, le courant et la température sur le LCD ainsi que leur communication à l'USB (vers notre interface graphique).

Tout d'abord, chaque donnée est formatée du type float vers le type char qui permet son affichage/communication et on ne prend que 3 chiffres après la virgule.

Ensuite pour l'affichage LCD, la donnée est affichée directement (avec indication du type de la donnée évidement) avec la fonction LCD_OUT ou LCD_OUT_CP (Voir: 3.3.4. Configuration et utilisation de l'afficheur LCD).

Après pour la communication USB, le buffer **writebuff** est lui le responsable de transmission de données du PIC vers le PC (interface graphique). Donc pour spécifier quelle donnée a été transmise dans l'USB, il faut ajouter un identifiant dans les premiers 8 octets (le choix de 8 octet est arbitraire uniquement). Suivant cet identifiant la partie software pourra distinguer le type de donnée qu'elle a reçu (Tension, Courant, Température ou un message).

L'identifiant est :

"0__tensi" pour une tension

"0__coura" pour un courant

"0__tempe" pour une température

"0__messa" pour un message

Enfin la donnée est envoyée avec la fonction HID_Write.

Exemple du format du writebuff pour un courant :

'0'	'_'	'_'	'c'	'o'	'u'	'r'	'a'	6	.	3	4	1	'\0'	….	….

Chaque case représente un octet. La donnée envoyée est un courant et sa valeur est de 6,341 A.

3.6.6. La fonction GetTensMax()

Comme décrit au début du chapitre, la tension maximale est, d'une part, un indicateur de fin de la 1ère phase (courant constant) et d'une autre part elle joue le rôle de la tension à appliquer dans la 2ème (tension constante). Cette tension maximale dépend de la température suivant un facteur négative nommé **coef_temp.** La fonction GetTensMax sert à déterminer cette tension suivant la valeur de la température lue.

3.6.7. La fonction GetTensFloat ()

Comme décrit au début du chapitre, la tension de maintien qui doit être appliquée dans la 3ème dépend de la température suivant un facteur négative nommé **coef_temp.** La fonction GetTensFloat sert à déterminer cette tension suivant la valeur de la température lue.

3.6.8. La fonction GetDuty (float tension_max_maintien)

Les calibrations en pratique du hacheur indiquent que l'équation liant le rapport cyclique et la tension de sortie est une droite de la forme

$$\boldsymbol{V_s = a * \alpha + b}$$

Les valeurs expérimentales de a et b sont connues (elles ont été déterminées pratiquement). Donc La fonction GetDuty calcule le rapport cyclique qu'il faut appliquer sur le PWM suivant la tension de sortie désirée (inscrite en argument **tension_max_maintien**). La tension de sortie peut être soit la tension maximale (2ème phase) ou bien la tension de maintien (3ème phase).

3.6.9. La fonction charge()

Cette fonction a pour but de traiter les informations reçues sur le convertisseur analogique numérique (tension, courant et température) pour déterminer le mode de charge (en courant ou en tension) et dans quelle phase, déterminer le courant de charge, varier le rapport cyclique pour une tension de sortie désirée. Tout cela pour une gestion correcte du chargement de la batterie.

1ère condition :

Tester le bit flag INTCON.TMR0IF pour savoir s'il y a débordement dans le registre TMR0H:TMR0L (2 registre de 8 bit chacun) c'est-à-dire passage de FFFF à 0000. Si c'est le cas nous faisons un remise à zéro du TIMER0 et mettre la position de départ (de tel façon le débordement se fait après 1 seconde) ensuite une incrémentation de la variable timer.

2ème condition :

Tester si la variable ***timer*** a atteint 3, c'est dire passage de 3 secondes. Si c'est le cas: on lit les données avec les fonctions ***lectTens, lectCour*** et ***lectTemp*** pour les afficher sur l'interface graphique avec les fonctions ***communicationTens***, ***communicationCour*** et ***communicationTemp*** puis parcourir les autres conditions, sinon on sort de la fonction.

3ème condition :

Tester l'état de charge. S'il est dans la position de départ (***etat = 0***), on teste si la tension batterie est supérieure à 12.6 V. Si c'est le cas ça veut dire que la batterie est chargée et ne nécessite pas la charge. Donc nous passons en mode de maintien: on active le générateur de tension, on cherche la tension adéquate à appliquer, on lance la commande du PWM. Si non (si la tension batterie est inférieur à 12.6) cela nécessite le début du processus de charge en affectant 1 à la variable ***etat***.

4ème condition :

Tester l'état de charge. S'il est dans la phase de charge à courant constant (etat = 1), activer le générateur de courant tant que la tension batterie est inférieure à la tension maximale et la valeur du courant est déterminée avec la fonction ***choixcourant.*** Cette dernière réagit aux commandes envoyées par l'utilisateur sur l'interface graphique. Pour des raisons de fiabilité de la lecture de tension batterie, on coupe la charge pendant 2 secondes pour laisser la batterie d'équilibre et on mesure la tension. Dans le cas où la tension dépasse la tension maximale la phase de charge à tension constante commence en affectant 2 à la variable etat. L'utilisateur sera informé également via l'afficheur LCD et l'interface graphique.

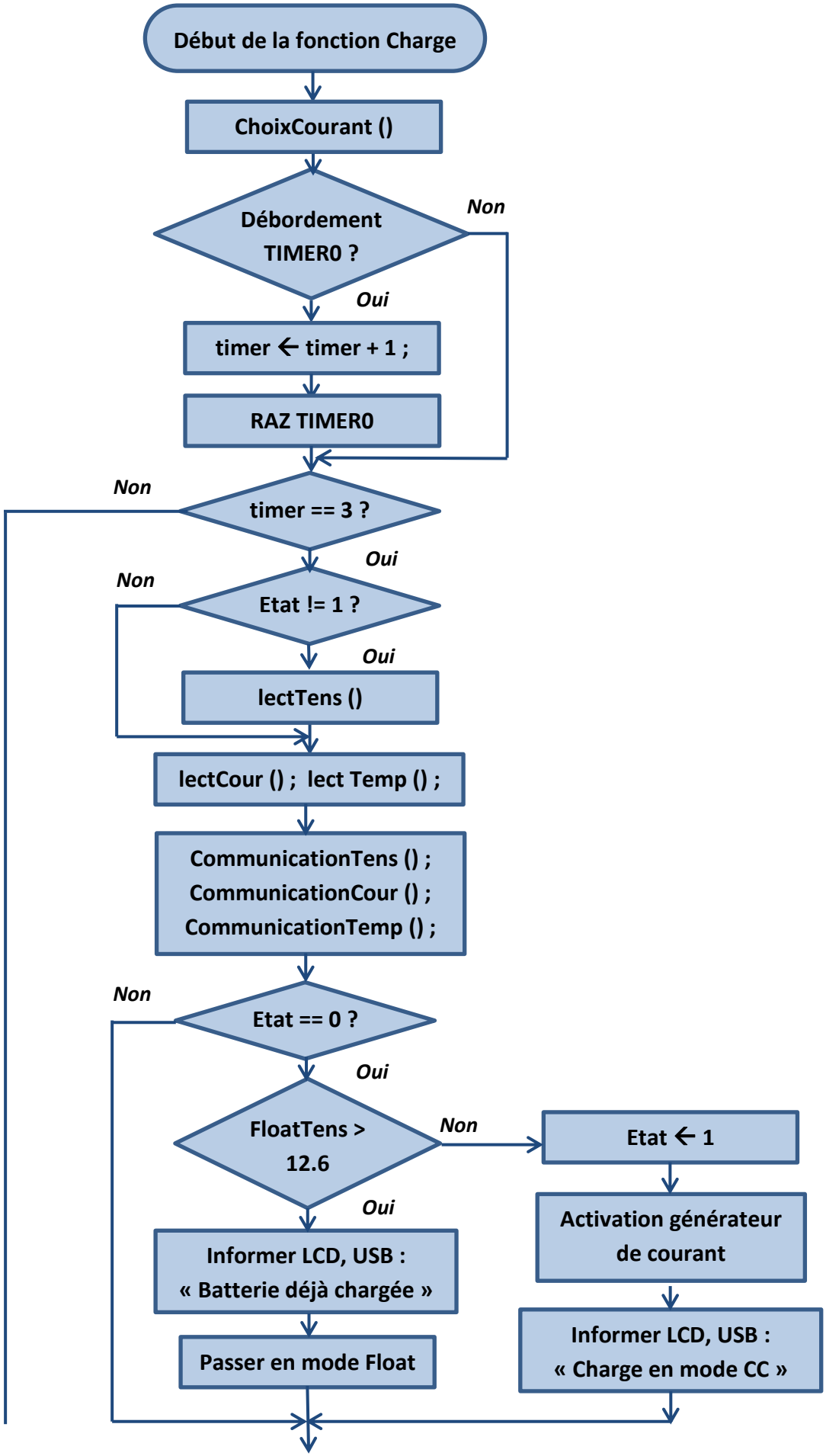
5ème condition :

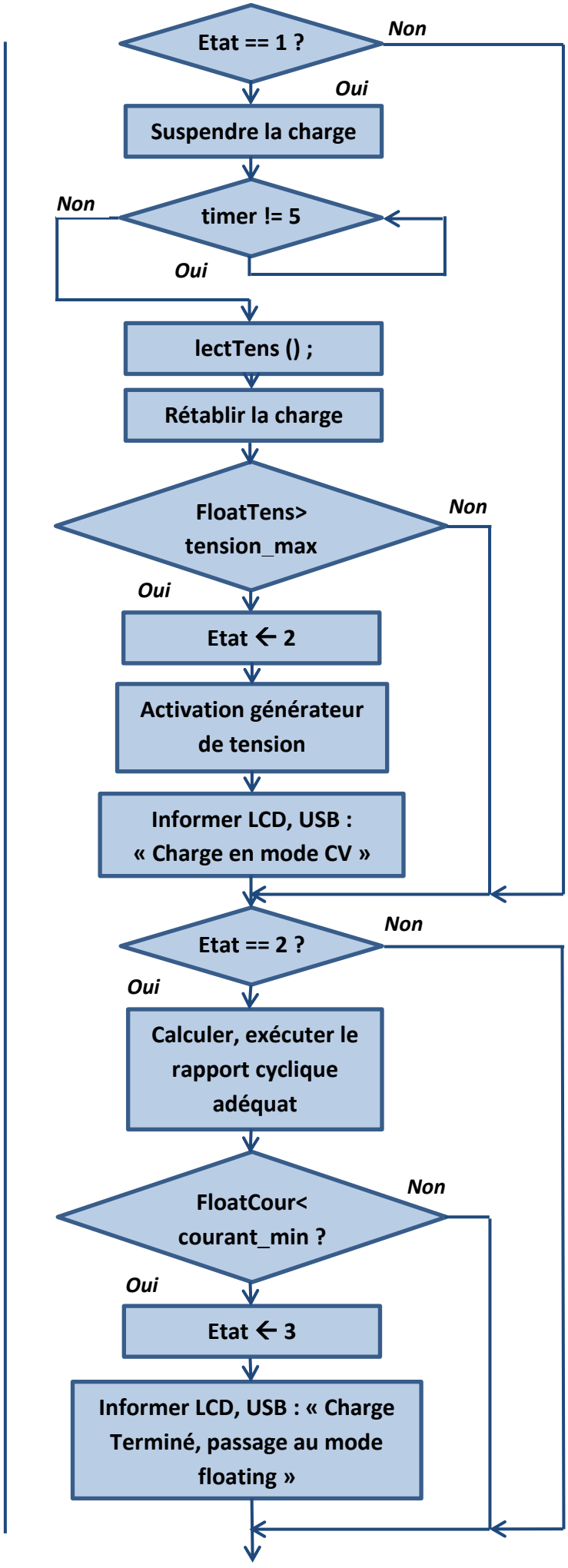
Tester l'état de charge. S'il est dans la phase de charge à tension constante (etat = 2), activer le générateur de tension, chercher la tension adéquate à appliquer, et lancer la commande du PWM. Puis on teste : si le courant est inférieur à C/100 la 2ème phase prend fin et appelle au début de la 3ème phase en affectant 2 à la variable etat. L'utilisateur sera informé également via l'afficheur LCD et l'interface graphique.

6ème condition :

Tester l'état de charge. S'il est dans la phase de maintien (etat = 3), informer l'utilisateur sur la fin de charge en affichant sur le LCD en communiquant vers l'interface. Ensuite, chercher la tension de maintien adéquate à appliquer, et lancer la commande du PWM.

La figure suivante montre l'organigramme de la fonction ***charge*** en détail :




Etat == 1 ?
Non
Oui
Suspendre la charge
timer != 5
Non
Oui
lectTens () ;
Rétablir la charge
FloatTens> tension_max
Non
Oui
Etat ← 2
Activation générateur de tension
Informer LCD, USB : « Charge en mode CV »
Etat == 2 ?
Non
Oui
Calculer, exécuter le rapport cyclique adéquat
FloatCour< courant_min ?
Non
Oui
Etat ← 3
Informer LCD, USB : « Charge Terminé, passage au mode floating »

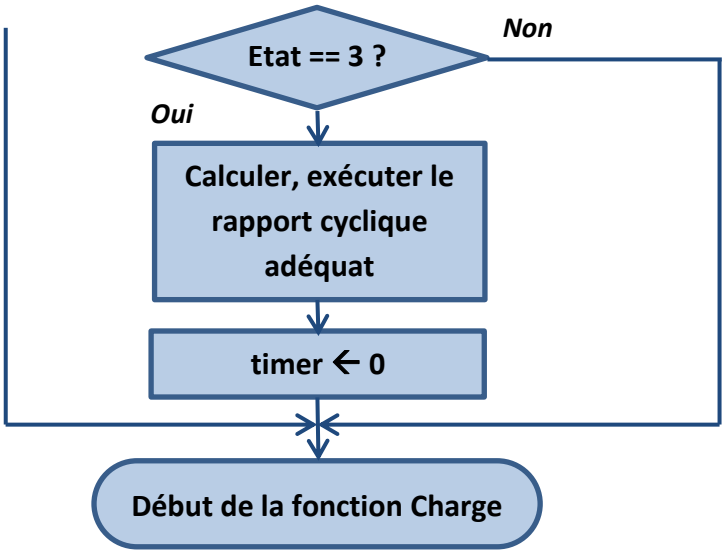


Figure 3.6: Organigramme de la fonction Charge

3.7. Conclusion

Pour une gestion flexible et efficace l'utilisation du microcontrôleur est indispensable. Ce dernier nécessite plusieurs configurations, une maitrise en programmation du langage C et une familiarisation avec l'environnement MikroC.

Le microcontrôleur s'occupe de la lecture de tension, courant et température afin de commander le PWM du hacheur et la commutation entre les deux types d'alimentation. Ainsi que leur écriture dans l'afficheur LCD et sur l'interface graphique. Tous cela afin d'assurer un algorithme de charge à trois phases.

Chapitre 4

Communication USB

4.1. Introduction

Le cahier de charge de notre projet impose une liaison avec l'ordinateur. Cela permettra une communication entre l'utilisateur et le circuit de charge pour un but de commande et d'un suivi direct de l'état de charge de la batterie. Comme la communication USB a pris de l'ampleur ces dernières années par rapport à la liaison série RS232, nous l'avons donc choisie dans la réalisation de notre projet.

Dans ce chapitre nous allons parler en général sur la communication USB, la classe USB HID utilisée ainsi que sur le fichier descripteur et la méthode de sa génération.

4.2. Généralités sur l'USB

USB, acronyme de « Universal Serial Bus » (Bus série universel), est une norme industrielle développée au milieu des années 1990, qui définit les câbles, connecteurs et protocoles de communication utilisés dans un bus pour but de communication et d'alimentation entre les ordinateurs et les appareils électroniques. Il est actuellement développé par le « USB Forum Implementers » (USB IF).

Figure 4.1: Connecteur USB.

USB a été conçu pour normaliser la connexion des périphériques d'ordinateurs (y compris les claviers, les dispositifs de pointage, appareils photo numériques, imprimantes, lecteurs multimédia portables, lecteurs de disques et les cartes réseau) à des ordinateurs personnels, à la fois pour communiquer et fournir de l'énergie électrique. Il a remplacé une variété d'interfaces antérieures, telles que les ports parallèles, ainsi que des chargeurs de puissance séparés pour les appareils portables. [33]

La connexion USB se fait via un câble à quatre fils simples. Deux fils dédié pour l'alimentation (Vcc et GND), et la liaison série se fait avec les autres deux fils (D+ et D-). Cette liaison fonctionne suivant plusieurs vitesses selon la version. Le tableau ci-dessous. [34]

Tableau 4.1: Versions de l'USB et vitesse de transfert. [33]

Version de l'USB	**Vitesse**	**Remarque**
USB 1.0	1.5 Mbit/sec	Low Speed
	12 Mbit/sec	Full Speed
USB 2.0	480 Mbit/sec	High Speed
USB 3.0	5 Gbit/sec	Super Speed
USB 3.1	10 Gbit/sec	Super Speed+

4.3. Les classes de périphérique

Les protocoles USB peuvent configurer des périphériques au démarrage ou quand ils sont branchés au moment de l'exécution. Ces dispositifs sont répartis en différentes classes de périphériques. Chaque classe de périphérique définit le comportement et les protocoles communs pour les appareils qui servent à des fonctions similaires. [34]

La fonctionnalité des périphériques USB est définie par des codes de classe, communiqués à l'hôte USB (PC par exemple) pour affecter le chargement des modules appropriés du pilote de logiciel pour chaque périphérique connecté. Celle-ci permet l'adaptabilité et l'indépendance du dispositif de l'hôte pour supporter de nouveaux appareils de différents fabricants. [33]

Des exemples de classes de dispositifs USB sont présentés dans le tableau suivant:

Tableau 4.2: Classes de périphérique et exemples. [34]

Classe du périphérique	**Exemple**
Affichage	Ecran
Communication	Modem
Audio	Haut-parleur
Mass Storage	Disque dur
Human interface	Souris

Dans notre projet nous avons utilisé la classe « Human Interface Device » HID.

4.4. Détails sur la classe HID

La classe HID se compose principalement de dispositifs qui sont utilisés par les humains pour contrôler le fonctionnement des systèmes informatiques. Des exemples typiques de dispositifs de classe HID comprennent:

* Clavier et dispositifs de pointage, par exemple, les souris standards, trackballs et joysticks.

* Commandes pour panneau frontal exemple: boutons, interrupteurs, curseurs …

* Les contrôles qui pourraient être trouvés sur des appareils tels que les téléphones, magnétoscopes volants de jeux …

* Les appareils qui ne nécessitent pas l'interaction humaine, mais fournissent des données dans un format similaire aux appareils qui utilisent la classe HID, par exemple: les lecteurs de codes à barres, des thermomètres ou voltmètres.

Les dispositifs typiques de la classe HID comprennent des indicateurs, des affichages spécialisés, des sorties audio, pavés tactile. Par conséquent, la définition de la classe HID comprend un support pour les différents types d'interface dirigés vers l'utilisateur final. [34]

4.5. Pourquoi la classe HID ?

L'un des avantages de la classe USB HID est l'abondance des pilotes (drivers) de périphériques qui sont disponibles dans la plupart des systèmes d'exploitation modernes (Dans le cas de Windows les pilotes de cette classe sont définis dans une librairie contenue dans le fichier HID.dll qui se trouve dans le dossier system32. [35]).

Les périphériques de classe USB HID et leurs fonctions de base sont définies dans la documentation USB-IF (USB Implementers Forum) sans aucun logiciel spécifique. En raison de ces descriptions génériques, il est facile pour les concepteurs d'inclure des pilotes qui fonctionnent pour les périphériques tels que les claviers, les souris, les microcontrôleurs qui prennent en charge l'USB et les autres périphériques génériques d'interface humaine. L'inclusion de ces pilotes génériques permet un déploiement plus rapide des appareils et une installation plus facile par les utilisateurs finaux. [36]

4.6. Fichier descripteur

Lorsqu'un système, agissant en tant que périphérique USB, communique avec un autre système agissant comme hôte (ordinateur), il faut spécifier les descripteurs USB qu'on souhaite exposer à l'hôte. [37]

Il s'agit d'un ensemble de structures de données qui permettra à l'hôte d'apprendre d'avantage sur le dispositif connecté [38]. Ces données sont stockées dans des segments sur la ROM (mémoire en lecture seule) [34].

Ce fichier descripteur est sous forme d'un fichier source qui doit être ajouté à chaque projet basé sur la bibliothèque USB. Ce fichier indique la fonction du système de l'appareil (par exemple, le stockage de masse, le transfert de données en série, interface homme machine, …). Il contient également l'identifiant du fournisseur et le nom, l'identifiant du produit et le nom, la longueur du rapport, et d'autres informations pertinentes. [39]

Tous les appareils USB ont une hiérarchie de descripteurs qui détaillent pour le compte de l'hôte des informations l'instruisant sur la nature de l'appareil, qui l'a réalisé, quelle version USB il supporte, de combien de manières il peut être configuré, le nombre de terminaisons et leurs types, etc.

Les descripteurs les plus courants sont :

· Descripteurs d'appareils ;

· Descripteurs de configurations ;

· Descripteurs d'interfaces ;

· Descripteurs de terminaisons ;

· Descripteurs de chaînes.

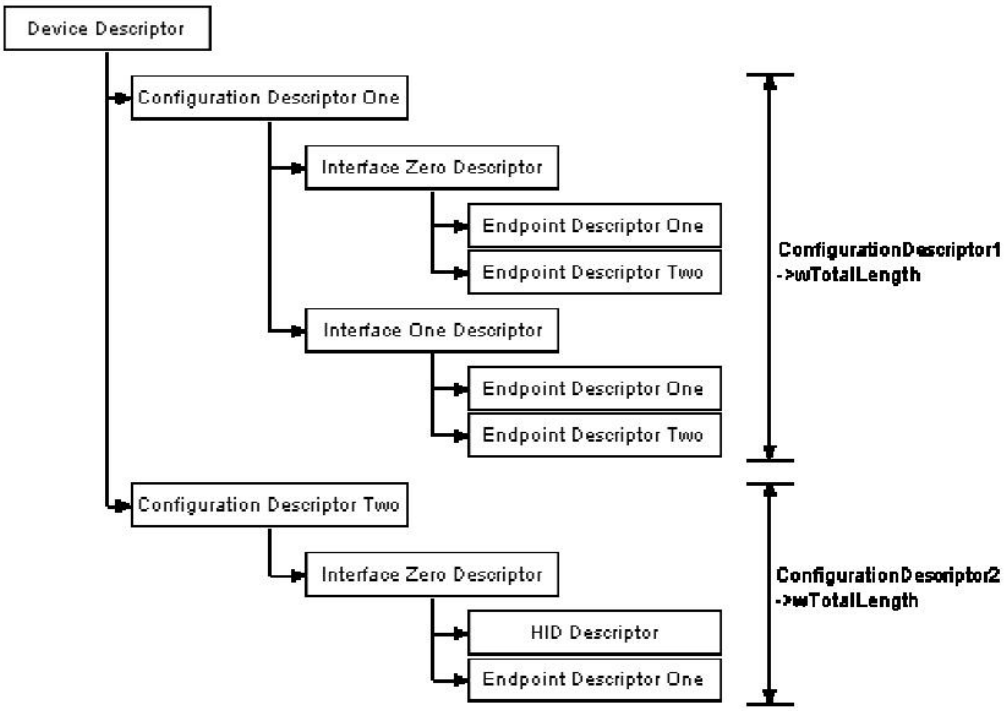


Figure 4.2: Représentation des descripteurs dans la communication USB. [40]

Les dispositifs USB ne peuvent avoir qu'un seul **descripteur d'appareil**. Le descripteur d'appareil inclut des informations qui précisent la révision USB à laquelle l'appareil se soumet, les ID (Identificateurs d'Appareils) du produit et du constructeur utilisés pour charger les pilotes logiciels appropriés et le nombre possible de configurations que l'appareil peut avoir. Le nombre de configurations indique combien de ramifications de descripteurs de configurations sont appelées à suivre.

Le **descripteur de configuration** précise des valeurs comme la quantité de puissance qu'utilise cette configuration particulière, si l'appareil est autoalimenté ou alimenté par le bus et le nombre d'interfaces qu'il possède. Quand un appareil est énuméré, l'hôte lit les

descripteurs d'appareils et peut décider de la configuration à valider. Il peut seulement valider une configuration à la fois.

Par exemple, il est possible d'avoir une configuration d'alimentation de bus de grande puissance et une configuration autoalimentée. Si l'appareil est branché à un hôte possédant une alimentation électrique secteur, le pilote logiciel de l'appareil choisira, peut-être, de permettre la configuration d'alimentation du bus de grande puissance tolérant ainsi que l'appareil soit alimenté sans être relié au secteur. Cependant s'il est connecté à un ordinateur (par exemple), il pourra valider également la seconde configuration (autoalimentée) exigeant de l'utilisateur de brancher son appareil sur un point d'alimentation (secteur).

Le **descripteur d'interface** peut être vu comme un « entête » ou un regroupement de terminaison à l'intérieur d'un groupe fonctionnel accomplissant une seule fonctionnalité de l'appareil. Par exemple, nous pouvons avoir un appareil multifonction: fax – scanner – imprimante. Le descripteur d'interface 1 pourra décrire les terminaisons de la fonction fax, le descripteur d'interface 2, la fonction scanner et le descripteur d'interface 3 la fonction imprimante. Contrairement au descripteur de configuration, il n'y a pas de limitations à avoir une seule interface validée à la fois, donc un appareil peut avoir un ou plusieurs descripteurs d'interfaces validés en même temps.

Les **descripteurs de terminaison** sont utilisés pour décrire les terminaisons autres que la terminaison zéro. Chaque descripteur de terminaison est utilisé pour spécifier le type de transfert, la direction, l'intervalle d'interrogation et la taille maximale de paquet pour chaque terminaison. L'Hôte utilisera l'information renvoyée par ces descripteurs pour déterminer les besoins de bande passante du bus. La terminaison zéro, la terminaison de commandes par défaut et est configurée avant que n'importe quel autre descripteur ne soit sollicité.

Les **descripteurs de chaînes** fournissent une information lisible pour l'homme et sont optionnels. Par exemple l’inclusion du nom de produit et du nom de constructeur.

S'ils ne sont pas utilisés, tout champ d'index de descripteurs de chaînes doit être mis à zéro indiquant qu'il n'y a pas de descripteur de chaîne disponible.

Les chaînes de caractères sont codées au format Unicode et les produits peuvent être prévus pour comprendre les diverses langues. [40]

4.7. Des explications sur les points de terminaisons

Les points de terminaison (Endpoint) peuvent être décrits comme des sources de données et existent dans des périphériques USB seulement. Ils agissent comme une sorte de mémoire tampon. Les données stockées dans un point de terminaison peuvent être soit reçues en provenance d'un hôte USB (ordinateur par exemple) ou en attendant d'être envoyé vers ce dernier.

Le client d'un hôte USB peut envoyer des données à « Endpoint 1 » par exemple: En venant de l'hôte USB, les données seront envoyées à Endpoint 1 OUT. Le programme sur le microcontrôleur va alors lire les données dès qu'il est prêt à le faire. De retour des données doivent être écrites à Endpoint 1 IN, que le programme ne peut pas accéder librement au bus USB (le bus USB étant commandé par l'hôte). Les données dans Endpoint 1 IN restent là-bas jusqu'à ce que l'hôte envoie un paquet IN pour Endpoint 1 demandant les données.

Les règles suivantes sont applicables à tous les périphériques type microcontrôleur:

- Un dispositif peut avoir jusqu'à 16 OUT et 16 IN extrémités.
- Chaque paramètre peut avoir qu'un seul sens de transfert.
- Endpoint 0 est utilisé pour les transferts de contrôle seulement et ne peut être attribuée à une autre fonction.
- OUT se réfère toujours à la direction de pointage de l'hôte vers le périphérique.
- IN se rapporte toujours à la direction pointant vers l'hôte. [41]

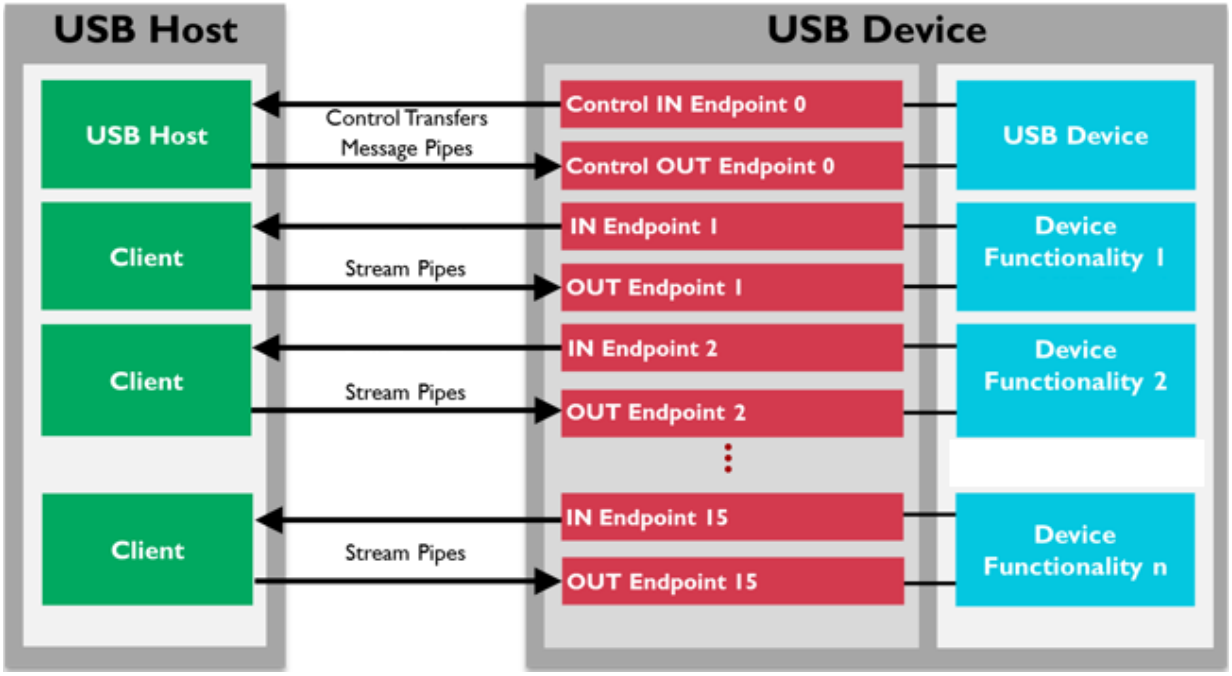


Figure 4.3: Représentation des points de terminaison et communication entre un périphérique USB et un hôte. [41]

4.8. Génération du fichier descripteur

La programmation du descripteur est complexe. Elle nécessite une grande concentration et une bonne maîtrise. Car elle exige la spécification de plusieurs paramètres des cinq différents types de descripteurs cités précédemment. Pour cela nous avons utilisé l'outil « HID Terminal » intégré avec MikroC PRO For PIC. Ce dernier offre la possibilité de générer un fichier descripteur en spécifiant uniquement : les identifiants du constructeur et du produit (VID et PID) et leurs noms, la longueur du rapport (entrée et sortie), le type d'alimentation (autoalimenté ou alimenté par le bus) ainsi que sa consommation en courant, en enfin l'intervalle d'interrogation du point de terminaison.

Le fichier pourra être créé en allant vers **Outils** puis **USB HID Terminal**. La figure suivante représente la fenêtre qui va s'ouvrir permettant la génération du fichier source.

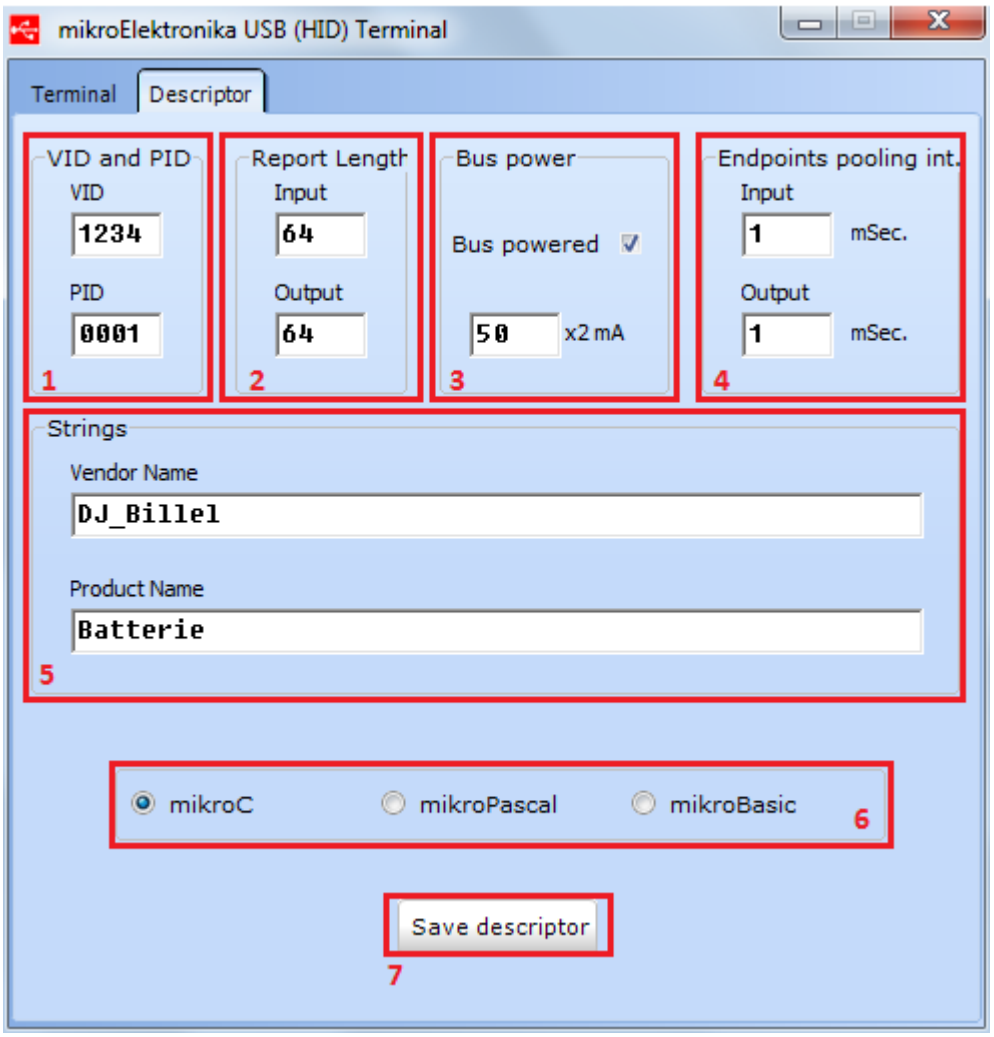


Figure 4.4: Boîte de dialogue pour la génération du fichier descripteur

1- Spécification des identifiants du constructeur et du produit (VID et PID)
2- Longueur des rapports (entrée et sortie) en octet.
3- Le type d'alimentation : autoalimenté (décoché) ou alimenté par le bus (coché) ainsi que les courants en x2 milliampères
4- Intervalle d'interrogation du point de terminaison pour les transferts de données. Exprimé en millisecondes.
5- Spécification des noms du constructeur et du produit
6- Choix du langage (C, Pascal ou Basic)
7- Génération et enregistrement du fichier

Le nom par défaut du fichier généré s'appelle **« USBdsc.c »**. On doit l'ajouter par la suite dans notre projet MikroC. Ce fichier décrira tout le protocole de communication entre notre microcontrôleur PIC18F2550 et le PC.

4.9. Description des différentes fonctions de communication USB HID dans la programmation du PIC

MikroC PRO for PIC fournit une librairie USB qui contient des fonctions de communication HID qui prennent en charge les dispositifs de classe HID. Nous citons par la suite les fonctions essentielles utilisées dans notre projet.

- La fonction HID_Enable : Elle active la communication USB HID en introduisant deux buffers de lecture et d'écriture. Cette fonction ne retourne rien.

Prototype : **void HID_Enable(char *readbuff, char *writebuff);**

Remarque

Les deux buffers de lecture et d'écriture doivent être de type « unsigned char » et inscrit en mémoire.

- La fonction HID_Read : Elle reçoit des messages de l'hôte et elle les stocke dans le buffer de lecture. Elle retourne 0 en cas d'échec. Autrement, elle retourne le nombre de caractères reçus. Elle requiert l'activation de la communication USB HID (c'est à dire l'exécution de la fonction HID_Enable).

Prototype : **char HID_Read(void);**

- La fonction HID_Write : Cette fonction envoie des données du buffer d'écriture vers l'hôte. Cette fonction a besoin du buffer d'écriture et la longueur des données à transmettre comme paramètres. Elle requiert également l'activation de la communication USB HID (c'est à dire dis l'exécution de la fonction HID_Enable).

Prototype : **char HID_Write(char *writebuff, char len);**

- La fonction HID_Disable : Cette fonction désactive la communication USB HID. Elle n'a besoin d'aucun paramètre et elle ne retourne rien.

Prototype : **void HID_Disable(void);**

4.10. Conclusion

L'USB est un système de communication largement utilisé pour relier un ordinateur avec différentes sortes de périphériques et devient de plus en plus populaire. Dans notre projet, et pour la connectivité PC – PIC18F2550, la classe la plus adaptée pour cela est la classe HID (Humain Interface Device). Cependant sa programmation et sa gestion nécessite une grande maitrise des protocoles de communication, c'est pour cela nous avons utilisé l'outil « HID Terminal » et les fonctions prédéfinies du logiciel MikroC For PIC pour réduire la complexité du problème.

Chapitre 5

Interface graphique

5.1. Introduction

Dans ce chapitre nous allons présenter la partie interface graphique codé en C# en utilisant l'environnement de Microsoft Visual Studio Community 2015. Nous allons montrer la partie graphique, et comment l'utiliser, ainsi qu'éclaircir les lignes de programmation. Evidemment, nous allons expliquer le fonctionnement de la communication USB et la bibliothèque utilisée dédiée pour cela.

5.2. Généralités sur les interfaces graphiques

Les interfaces graphiques GUI (Graphic User Interface) sont des interfaces de communication entre une application avec l'utilisateur. Il existe différents mécanismes de contrôle tels que les boutons, les listes, des éléments de choix, et les cases à cocher…. Il existe aussi les zones de texte qui permettent à l'utilisateur de saisir les données, les boîtes de dialogue pour entrer des configurations supplémentaires ou affichage d'erreurs. L'utilisateur interagit avec le programme en utilisant les différents dispositifs de pointage, généralement la souris et le clavier. [42, 43]

5.3. Langage et environnement de programmation

Pour la conception de notre logiciel, nous avons opté pour le choix d'un environnement crée par Microsoft qui est le Visual Studio et ceci pour plusieurs raisons. Visual Studio est un environnement de développement très riche et convivial, il est utilisé pour la création d'applications pour Windows, Android et iOS, ainsi que des applications web modernes et de services de « Cloud Computing ». Cet environnement supporte plusieurs langages de programmation : C#, Visual Basic, F#, C++, Python, Node.js et HTML / JavaScript. Il offre également un débogage avancé. Pour notre cas, nous avons utilisé « Visual Studio Community », avec sa dernière version 2015, qui un environnement gratuit avec plein de fonctionnalités. [44]

Le choix du langage se faisait sur 3 critères : La disponibilité d'une bibliothèque dédiée à la communication USB HID, le bon support technique, et enfin la familiarité et la convivialité du langage.

Pour le langage de programmation nous n'avons pas eu un choix bien clair. La communication USB HID étant difficile et complexe et nécessite une grande expertise, il était impératif de chercher des bibliothèques qui facilitent cette tâche. Une bibliothèque officielle n'existe pas donc il a fallu chercher des bibliothèques codées par des programmeurs indépendants. Les bibliothèques qui fonctionnent bien étant rares donc nous avons fait le choix du langage de programmation suivant la bibliothèque disponible. Après plusieurs recherches nous avons trouvé une bibliothèque qui s'appelle **« EasyUSBHidNetClass »** codé

en C#. Donc nous avons choisi le C# comme langage de programmation de notre interface graphique.

5.4. Fonctionnement global du logiciel

Le logiciel que nous avons conçu présente une interface conviviale pour la gestion de la charge d'une batterie. Elle contient un ensemble de boutons, des zones de textes, des cases à cocher, …

Au début une fenêtre de bienvenue s'affiche pour accueillir l'utilisateur :

Figure 5.1: Fenêtre d'accueil de l'interface graphique

Dès que l'utilisateur clique sut le bouton « suivant » la fenêtre principale s'affiche.

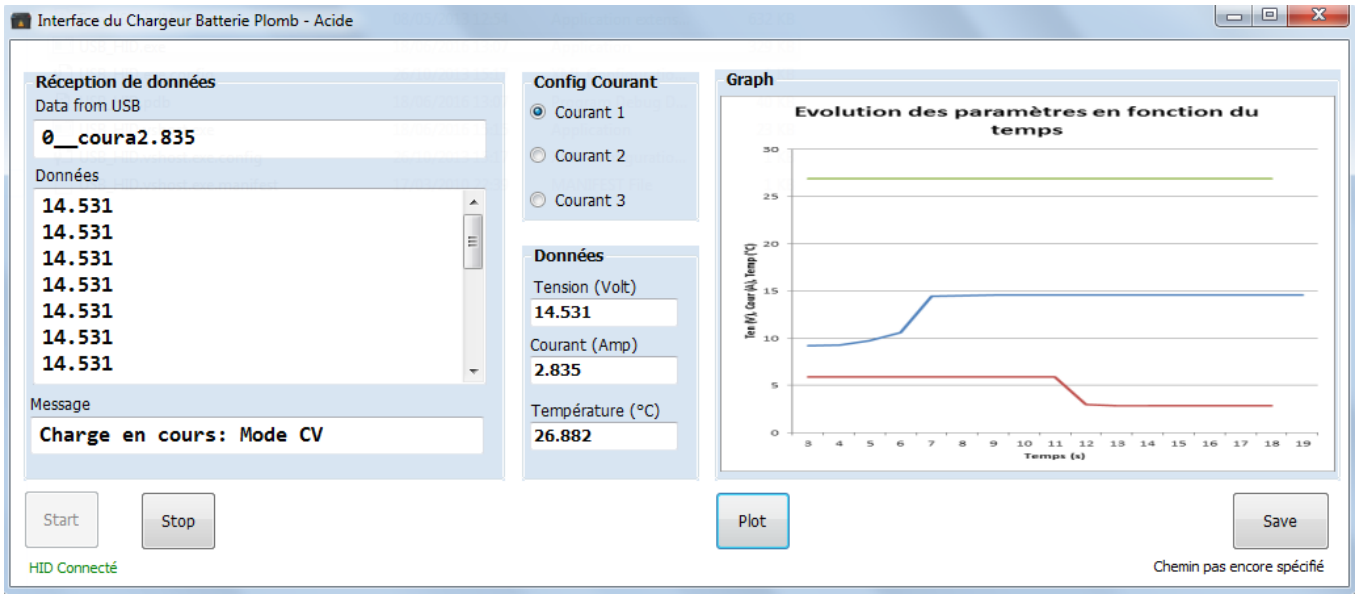


Figure 5.2: Fenêtre principale de l'interface graphique

La fenêtre principale se divise en plusieurs parties :

5.4.1. Zones de textes

Il y a plusieurs zones de texte dans notre interface. La première affiche toute la donnée reçue de l'USB. Et la deuxième affiche le cumul de toutes les tensions reçues. Quant à la troisième elle affiche les messages textuels reçus du PIC pour indiquer la phase de charge (Courant constant, tension constante, charge terminées …)

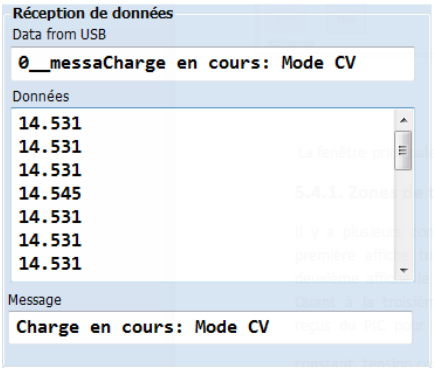


Figure 5.3: Zone de texte pour affichage d'informations

5.4.2. Configuration du courant

Comme indiqué dans la configuration R-2R du chapitre 2, il est possible de charger la batterie avec trois niveaux de courant, donc ces cases d'option donne à l'utilisateur la possibilité de choisir le courant de charge voulu (dans la phase charge à courant constant).

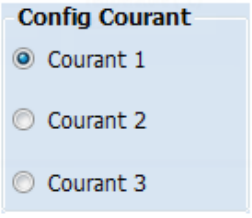


Figure 5.4: Cases d'options pour le choix du courant

5.4.3. Partie données

Elle affiche en temps réel les données envoyées par le PIC (la tension, le courant et la température)

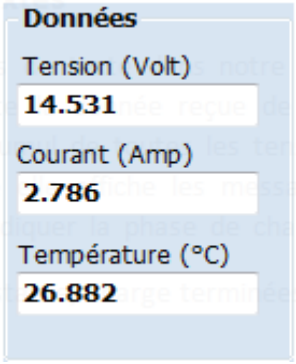


Figure 5.5: Zone d'affichage des données reçues

5.4.4. Traçage graphique

Cette zone visualise les paramètres de charge (courant, tension et température) depuis le début de la charge avec un traçage graphique qui s'actualise automatiquement.

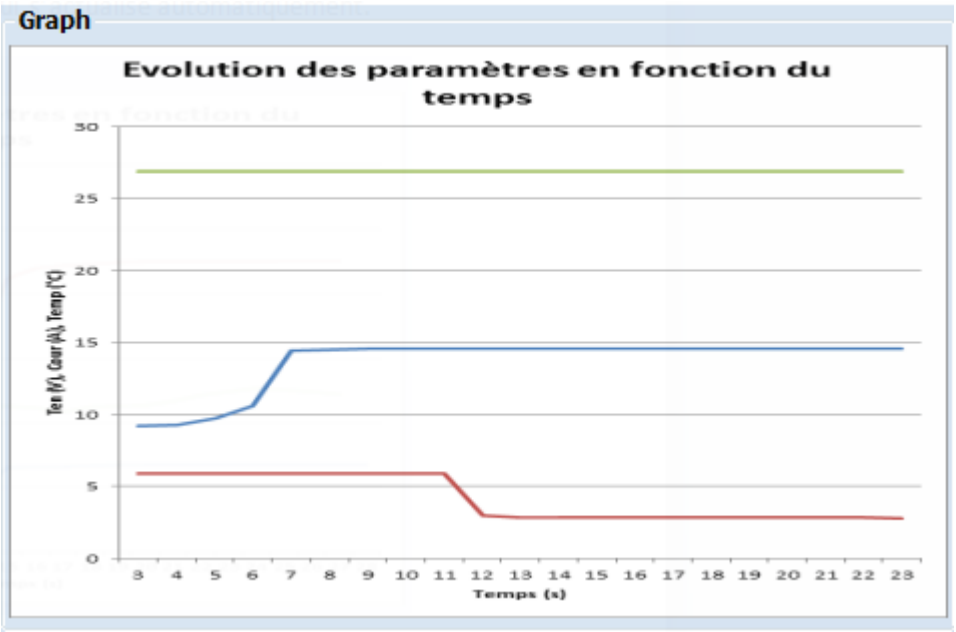


Figure 5.6: Zone de traçage des graphes

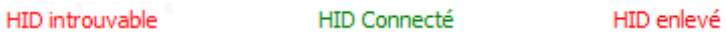
5.4.5. Boutons utiles

Pour commencer l'algorithme de charge on clique sur le bouton **« Start »,** et pour arrêter l'acquisition de données on clique sur **« Stop ».** Les données reçues peuvent être enregistré dans un fichier Excel, donc le bouton **« Save »** fera l'affaire. Une boite de dialogue s'affiche pour demander à l'utilisateur de nommer le fichier et donner l'endroit où le sauvegarder.

Figure 5.7: Boutons de l'interface graphique

5.4.6. Etiquettes

Les étiquettes ont pour but de donner des informations supplémentaires à l'utilisateur. Il est possible suivre le statut de la connectivité USB : une étiquette affiche **« USB introuvable »** si la communication n'a pas eu lieu, ou **« USB connecté »** si le lien est établi entre le PIC et l'interface, ou **« USB enlevé »** si la communication a eu lieu mais elle a été interrompue. Une autre étiquette en haut rappelle l'utilisateur de chemin de sauvegarde du fichier Excel.



5.5. Détails sur la programmation de l'interface

5.5.1. Explication des méthodes

5.5.1.1. Communication USB

5.5.1.1.1. Bibliothèque de communication USB

Comme déjà cité la bibliothèque utilisée s'appelle « EasyUSBHidNetClass ». C'est une bibliothèque Windows Forms de contrôle USB spécialement pour le développement de la communication USB de la classe HID. Elle est dédiée pour la communication entre une plateforme Visual C# ou Visual Basic.NET Microsoft avec les microcontrôleurs PIC, 8051, AVR, ARM et Freescale. Elle est sous forme de plusieurs classes et méthodes compilées sous format d'un seul fichier DLL (Dynamic Link Library). [45]

5.5.1.1.2. Méthodes de la bibliothèque de communication USB

La bibliothèque « EasyUSBHidNetClass » fournit plusieurs méthodes dédiées pour la lecture, l'écriture, le déclenchement d'événements dans le cas où un périphérique se connecte ou se déconnecte, … Et tout ce qui permette une bonne gestion de l'USB HID.

Par la suite nous allons expliquer par la suite l'ensemble des méthodes rattachées aux différents événements qui pourraient être déclenchés lors de la communication USB.

- `private void easyUSBHidNetClass1_DeviceUSB_dataReceived(object sender, easyUSBHidNetClass.DataRecievedEventArgs args)`

C'est la méthode de lecture depuis l'USB. Si une donnée vient d'être reçue dans le PC un événement se déclenche et cette méthode, étant rattachée à ce dernier, s'exécute automatiquement.

- `private void easyUSBHidNetClass1_DeviceUSB_dataSent(object sender, EventArgs e)`

Quand une donnée est envoyée, événement se déclenche. Cette méthode, étant rattachée à ce dernier s'exécute automatiquement.

- `private void easyUSBHidNetClass1_DeviceUSB_found(object sender, EventArgs e)`

Quand notre périphérique est trouvé, un événement se déclenche. Cette méthode, étant rattachée à ce dernier s'exécute automatiquement, et le label référant au statut de la connectivité s'écrit : « HID Connecté ».

- `private void easyUSBHidNetClass1_DeviceUSB_removed(object sender, EventArgs e)`

Quand notre périphérique vient d'être enlevé, un événement se déclenche. Cette méthode, étant rattachée à ce dernier s'exécute automatiquement, et le label référant au statut de la connectivité s'écrit : « HID enlevé ».

- `private void easyUSBHidNetClass1_anyDeviceUSB_found(object sender, EventArgs e)`

Quand n'importe quel périphérique USB est connecté, un événement se déclenche. Cette méthode, étant rattachée à ce dernier s'exécute automatiquement.

- `private void easyUSBHidNetClass1_anyDeviceUSB_removed(object sender, EventArgs e)`

Quand n'importe quel périphérique USB vient d'être enlevé, un événement se déclenche. Cette méthode, étant rattachée à ce dernier s'exécute automatiquement.

5.5.1.2. Chargement de la forme

Dès que l'interface se charge la méthode `Form1_Load` est exécutée. Cette dernière envoie un message d'erreur en cas Excel n'est pas bien installé sur le PC hôte. Ainsi qu'elle crée le fichier Excel (xlWorkBook) et la feuille de calcul (xlWorkSheet).

Le prototype de cette méthode est : `private void Form1_Load(object sender, EventArgs e)`

5.5.1.3. Activation de la forme

Après chargement, la forme s'active et exécute la fonction `Form1_Activated`. Cette méthode fait appel à la méthode `easyUSBHidNetClass1.DeviceUSB_Config_VID_PID (0x1234, 0x0001);` c'est une des méthodes de la bibliothèque « easyUSBHidNetClass » elle fournit respectivement le VID et le PID de notre périphérique microcontrôleur.

Le prototype de cette méthode est : `private void Form1_Activated (object sender, EventArgs e)`

5.5.1.4. Lecture de données

Dès qu'une donnée est reçue du port USB la méthode easyUSBHidNetClass1_DeviceUSB_dataReceived s'exécute. En premier lieu on met la donnée dans un tableau en prenant en considération qu'elle est codée en ASCII. Puis, on teste les premiers 8 caractères pour savoir quelle est le type de la donnée ("0__tensi" désigne la tension, "0__coura" désigne le courant, "0__tempe" désigne la température ou un message) pour l'afficher ensuite dans la zone de texte respective et l'écrire dans un fichier Excel. Enfin, on procède vers les mesures de création / rafraichissement du graphe.

5.5.1.5. Ecriture de données

L'écriture de données vers l'USB ce fait avec la méthode `Send Data`. Cette dernière prend en argument l'identifiant du rapport et un vecteur d'octets. Elle teste au début la connectivité avec l'USB HID (avec notre microcontrôleur) et essai d'envoyer le tableau des octets avec la méthode `easyUSBHidNetClass1.DeviceUsbHID.writeData` qui a à son tour les mêmes arguments en entrée. Si le microcontrôleur n'est pas connecté ou l'écriture n'a pas eu lieu un message d'erreur s'affiche.

Le prototype de cette méthode est : `public void Send_Data(int report_id, byte[] Data)`

5.5.1.6. Gestions des boutons

Quand l'utilisateur clique sur les boutons « Start » ou « Stop » on écrit, respectivement, les chaines `"__start"` ou `"__stop"` sur le buffer l'écriture « `DataWrite`», puis on l'envoie vers le PIC avec la méthode `Send_Data.`

Le clic du le bouton « Save » provoque dans la première fois la création d'une fenêtre de dialogue de type `SaveFileDialog` puis la montrer à l'utilisateur, ensuite sauvegarder le fichier (Save as …) ou annuler. Pour les autres fois le clic provoque la sauvegarde uniquement (save).

5.5.1.7. Gestion des cases d'option

Quand l'utilisateur clique sur une option, une commande sera envoyée vers l'USB. Chaque commande est représentée par une chaîne de caractères pour que le PIC puisse l'interpréter et charger la batterie suivant le courant désiré.

"__cour1" désigne le code envoyé pour une charge avec un niveau de courant faible ;

"__cour2" désigne le code envoyé pour une charge avec un niveau de courant moyen ;

"__cour3" désigne le code envoyé pour une charge avec un niveau de courant fort ;

5.5.1.8. Fermeture de la forme

Dès que l'utilisateur ferme le logiciel la méthode `Form1_FormClosing` s'exécute. Cette dernière rappelle l'utilisateur avec un message s'il a oublié d'enregistrer. Puis, elle ferme le fichier et l'application Excel, ainsi qu'elle libère toutes les ressources créées.

5.5.1.9. Méthodes en relation avec la charte graphique

Pour le tracé de la courbe des paramètres de charge, nous avons créé deux méthodes :

- `static private void SaveChartInit () :`

C'est une méthode qui initialise la charte graphique. Elle contient la création et l'initialisation des variables. Ainsi que les instructions de configuration comme la taille de la zone de traçage, le nommage du titre général et le titre des axes vertical et horizontal.

- `static private void SaveChart() :`

Cette méthode crée / rafraichi le tracé de la courbe avec les nouvelles données reçues.

5.5.2. Explication des variables

- `byte[] DataWrite :` C'est un tableau de type `byte` d'une taille de 64. Il travaille comme buffer, dans lequel on enregistre momentanément les données qu'on veut envoyer vers l'USB.
- `byte[] DataRead :` C'est un tableau de type `byte` d'une taille de 64. Il travaille comme buffer, dans lequel on enregistre momentanément les données reçu de l'USB.

- `int i_tens = 3, i_cour = 3, i_temp = 3 :` sont des indices entiers utilisés pour l'incrémentation et l'enregistrement dans le fichier Excel. Ils sont initialisés à 3 (3ème ligne).
- `string sub :` une variable de type string (chaîne de caractères) elle sert à enregistrer les 8 premiers caractères de la donnée reçue servant à distinguer la nature de la donnée (tension, courant ou température).
- `int save :` un entier indiquant si l'utilisateur a enregistré pour la 1ère fois le fichier Excel (save = 0) ou pas (save = 1).
- `string path:` Une chaîne de caractères stockant le chemin d'enregistrement du fichier Excel.
- `Excel.Application xlApp`: Variable qui représente l'application Excel
- `Excel.Workbook xlWorkBook` : Variable qui représente le fichier Excel
- `Excel.Workbook xlWorkSheet` : Variable la feuille de calcul du fichier Excel
- `Excel.ChartObjects xlCharts` : Représente une collection de toutes les chartes graphique dans un fichier Excel
- `Excel.ChartObject myChart` : Représente le conteneur de la charte graphique, celui-ci contrôle la taille, l'apparence de « chartePage », …
- `Excel.Chart chartPage` : Représente la charte graphique incluse dans le fichier Excel
- `Excel.Axis xAxis :` Représente l'axe horizontal de la charte graphique
- `Excel.Axis yAxis :` Représente l'axe vertical de la charte graphique

5.6. Conclusion

L'interface graphique est un moyen très important pour faire une liaison entre l'utilisateur et le circuit de charge. Elle était codée en C# en utilisant l'environnement de Microsoft Visual Studio Community 2015. A travers cette interface, l'utilisateur pourra déclencher et arrêter la charge, commander le niveau du courant de charge. Il recevra, également, en temps réel les paramètres: tension, courant et température de la batterie, et il pourra les stocker dans un fichier Excel.

Conclusion générale

Durant la période de réalisation de notre projet de fin d'études, nous avons essayé de concevoir un chargeur de batterie au plomb évolué. Bien que nous n'ayons pas eu le temps suffisant pour réaliser un produit fini près à l'utilisation, tous les blocs constituant ce chargeur fonctionnent correctement, à savoir le circuit de charge (le convertisseur Buck, l'alimentation en courant et l'alimentation en tension), l'algorithme implémenté sur le PIC, et enfin l'interface graphique et la communication USB. Nous sommes très enthousiastes à le finir jusqu'au prototype final.

A l'avenir, il sera possible de continuer d'améliorer le chargeur en utilisant un algorithme avancé utilisant les techniques de charge impulsionnelles. Ce genre de méthode peut augmenter considérablement le cycle de vie de la batterie et réduire le temps de charge [46]. Il est nécessaire de chercher d'optimiser davantage le chargeur en augmentant le rendement du hacheur, réduire la taille du circuit et ajouter des composants de sécurité pour le protéger.

Le projet de chargeur de batterie est très ambitieux, et c'est un bon début dans la réalisation des chargeurs. Les prochains étudiants pourront améliorer ce qui a été fait, puis essayer d'adapter ce travail pour qu'il fonctionne en utilisant les panneaux photovoltaïques. Egalement, pour les projets futurs, il sera possible de penser d'adapter le circuit pour des batteries de différentes tensions nominales (12V, 24V et 48V). Ainsi qu'adapter l'algorithme de charge et l'interface graphique pour qu'il fonctionne pour différentes technologies de batteries. Tout cela pour obtenir un système complet efficace et flexible.

Bibliographie

[1] : Dictionnaire Français-Français Larousse. Version numérique. Consulté le 25/Mai/2016. Disponible sur: www.larousse.fr

[2] : Wikipedia: Batterie d'accumulateurs. Consulté le 20 Mai 2016.

Disponible sur https://fr.wikipedia.org/wiki/Batterie_d%27accumulateurs

[3] : DILLENSEGER, Guillaume. Caractérisation de nouveaux modes de maintien en charge pour batteries stationnaires de secours.

Thèse de doctorat : Electronique, Université Montpellier II : 2004.

[4] : BOUTTE, Aïssa. Identification des paramètres internes d'une batterie pour des applications photovoltaïques. 2015.

Thèse de doctorat : Electronique, Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf. 2015.

[5] : Battery University: Battery Definitions and what they mean, édité le 14 Juin 2016

Disponible sur: http://batteryuniversity.com/learn/article/battery_definitions

[6] : Battery Education: Battery Voltage, édité le 07 Avril 2006

Disponible sur: http://www.batteryeducation.com/2006/04/battery_voltage.html

[7] : Wikipedia: Accumulateur Electrique, Charge électrique, consulté le 14 Mars 2016

Disponible sur:
https://fr.wikipedia.org/wiki/Accumulateur_%C3%A9lectrique#Charge_.C3.A9lectrique

[8] : DONDEL Théo, ESNAULT Jérémy. Pré Projet d'étude et réalisation: Chargeur de Batterie au plomb 48V, 2006. Institut Universitaire de technologie de Tours

[9] : Wikipedia: Internal Resistance, consulté le 14 Mars 2016

Disponible sur: https://en.wikipedia.org/wiki/Internal_resistance

[10] : Energizer. Technical Bulletin, Battery Internal Resistance, Version 1.0.0, Décembre 2005.

[11] : Amtex Electronics PTY LTD. Battery Charging Terminology: Battery Charging Topology

[12] : J J A Wilkinson, G A Covic. A new pulse charging methodology for lead acid batteries. IPENZ Transactions, 1998.

[13] : BUCHMANN, Isidor, Batteries in a Portable World: A handbook on rechargeable batteries for non-engineers, 2ème édition. 2001.

[14] : Photo Ni-Cd: Commercial Galvanic Cells

http://chemwiki.ucdavis.edu/Core/Analytical_Chemistry/Electrochemistry/Case_Studies/Commercial_Galvanic_Cells

[15] : Cheng Siong Lee, Hsiung Cheng Lin, Sin-You Lai, Development of Fast Large Lead-Acid Battery Charging System Using Multi-state Strategy, International Journal on Computer, Consumer and Control, 2013.

[16] : Microchip, PICREF-2, Intelligent Battery Charger Reference Design, 1998.

[17] : Microchip, PIC18F2455/2550/4455/4550 Data Sheet, 2006.

[18] : Amirul fikri bin jafri. Phone charger powered by wind energy using buck converter

Thèse de doctorat, Université de Technologie de la Malaisie, 2015

[19] : Haddadi, Mourad. Les alimentations stabilisées. Cours Conception de Maquettes.

[20] : International Rectifier, Data Sheet IRFZ44n

[21] : Avago Technologies, HCPL-2200 / HCPL-2219 Data Sheet

[22] : Hector E. Alberti, Mohammad A. Ghani, Maximum Power Point Tracker

Mémoire de licence, Worcester Polytechnic Institute, 2014.

[23] : International Rectifier, Data Sheet IR2111 Half-Bridge Driver

[24] : On Semiconductor, LM2575, NCV2575 Data Sheet, 2009

[25] : Texas Instruments, TL43xx Precision Programmable Reference, 2015.

[26] : National Semiconductor, LM110/LM210/LM310 Voltage Follower, 1994.

[27] : Texas Instruments, LMx11 Quad Differential Comparators, 2015.

[28] : Texas Instruments, INA118 Precision, Low Power Instrumentation Amplifier, 2016.

[29] : Texas Instruments, LM35 Precision Centigrade Temperature Sensors, 2016.

[30] : W. Merrouchea, A. Malek. Charging Algorithm for Increasing Lead-Acid Battery Cycle Life in Photovoltaic Systems. First International Conference on Renewable Energies and Nanotechnology impact on Medicine and Ecology, Constantine, Algeria. ICREN-01/2013 February 16-17, 2013

[31] : Tutoriel: LiquidCrystal library, 17/08/2015, disponible sur https://arduino.cc/en/Tutorial/LiquidCrystal

[32] : Allaboutcircuits.com, Interface an LCD with an Arduino, 2015

Disponible sur: http://www.allaboutcircuits.com/projects/interface-an-lcd-with-an-arduino/

[33] : Wikipedia: USB, consulté le 20 Avril 2016

Disponible sur: https://en.wikipedia.org/wiki/USB

[34] : Device Class Definition for Human Interface Devices (HID), Firmware Specification— 27/6/01, Version 1.11

[35] : VSKsoft PC Knowledge, hid.dll fix tool, publié le 16 Juin 2015

Disponible sur http://www.vsksoft.com/pck/hid-dll-fix-tool/

[36] : Wikipedia: USB Human Interface Device class, Drivers, consulté le 20 Avril 2016

Disponible sur: https://en.wikipedia.org/wiki/USB_human_interface_device_class#Drivers,

[37] : Support technique en ligne du « QNX® Software Development Platform », consulté le 1 juin 2016

Disponible sur :
http://www.qnx.com/developers/docs/660/index.jsp?topic=%2Fcom.qnx.doc.dev_pub.ref_guide%2Ftopic%2Fusblauncher_config_usb_descriptors.html

[38] : Axelson, Jan. USB Complete: Everything You Need to Develop USB Peripherals, 3ème édition.

[39] : Aide de MikroC For PIC Pro: USB HID Library

Disponible sur: http://download.mikroe.com/documents/compilers/mikroc/pic/mikroc-pic-manual-v101.pdf

[40] : Bernard Acquier. L'USB en bref : Donner un sens au standard USB, publié le 30 avril 2005

Disponible sur: http://acquier.developpez.com/cours/USB/#LV-E

[41] : Support technique en ligne, Arm Keil. USB Component: MDK Middleware for USB Device and Host Communication, USB Communication. Version 6.7

http://www.keil.com/pack/doc/mw/USB/html/_u_s_b__endpoints.html

[42] : Bruce E. Wampler, The Essence of Object Oriented Programming with Java and UML, 2011
Disponible sur: http://www.lcc.uma.es/~amg/ISE/OOP-Java-UML/Chapter1.html#1039175

[43] : Derek Molloy, Module EE402 - Object-oriented Programming with Embedded Systems, Dublin City University, Ireland, 2015/2016.

[44] : Site officiel du Microsoft Visual Studio: https://www.visualstudio.com/

[45] : Site officiel de USBHidNetClass: http://www.usbhidnetclass.com/

[46] : Liang-Rui Chen. A Design of an Optimal Battery Pulse Charge System by Frequency-Varied Technique. IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, VOL. 54, NO. 1, 2007

Annexe: Récapitulatif des différents circuits