Université François-Rabelais de Tours
Institut Universitaire de Technologie de Tours
Département Génie Électrique et Informatique Industrielle

Projet tutoré : Lampe torche à LED

Florian COUROUGE
Thomas JOULAIN
2e K4A – Promotion 2010/2013

Enseignants
M. LEQUEU
M. AUGER

Université François-Rabelais de Tours
Institut Universitaire de Technologie de Tours
Département Génie Électrique et Informatique Industrielle

UNIVERSITE FRANCOIS-RABELAIS
TOURS
Institut Universitaire de Technologie
Département
GENIE ELECTRIQUE ET
INFORMATIQUE INDUSTRIELLE

Projet tutoré : Lampe torche à LED

Florian COUROUGE
Thomas JOULAIN
2e K4A – Promotion 2010/2013

Enseignants
M. LEQUEU
M. AUGER

Sommaire

Introduction....4
1.Cahier des charges....5
1.1.Contraintes....5
2.Planning de répartition des tâches effectuées....6
3.Analyse générale du projet....7
4.Première carte....8
4.1.Analyse....8
4.1.1.Alimentation à découpage....8
4.1.2.Alimentation de la LED....9
4.1.3.Pont diviseur....10
4.1.4.ATtiny 10....10
4.1.5.Connecteur ISP-6pattes....11
4.1.6.Réalisation de la carte 1....12
5.Deuxième carte....14
5.1.Connecteur OLIMEX....14
5.2.Connecteur TPI....15
5.3.CMS....16
5.3.1.1er étape....16
5.3.2.2e étape....16
5.3.3.3e étape....16
5.4.Modifications apportées....17
6.Troisième carte....18
6.1.ATtiny 13....18
6.2.Connecteur ISP – 10 broches....19
6.3.Réalisation de la carte....20
7.Quatrième carte....21
7.1.Ordinogramme....22
Conclusion....23
Résumé....24
Index des illustrations....25
Index des tables....26
Bibliographie....27

Introduction

Dans le cadre de l'étude et réalisation, nous avons choisi de faire une lampe torche qui utilisera une diode électroluminescente de puissance.

Tout d'abord, nous allons voir comment nous avons fait notre première carte. Surtout quels composants nous avons mis en œuvre afin d'alimenter correctement la diode électroluminescente et le micro-contrôleur. On montrera aussi quel micro-contrôleur on a choisi et comment on va le programmer.

Ensuite, nous verrons notre deuxième carte. Et surtout, on montrera une nouvelle technique afin de programmer le micro-contrôleur et quelle liaison on doit utiliser. Et on expliquera comment souder un composant monté en surface (CMS[1]).

Puis, nous montrerons notre troisième carte qui utilise un autre micro-contrôleur et comment on doit le programmer et quel connecteur on doit utiliser.

Enfin, on expliquera l'ordinogramme qui permet de programmer le micro-contrôleur placé sur notre carte.

1 CMS : Composants montée en surface

1. Cahier des charges

Dans ce projet nous allons créer une lampe torche à LED. On utilisera un gradateur afin de faire varier l'intensité lumineuse de la LED.

1.1. Contraintes

–Un petit micro-contrôleur fonctionnant en 5V. (Atmel)

–Batterie de 9,6V à pile.

–Utilisation d'une alimentation à découpage pour adapter la tension de la batterie afin d'alimenter le micro-contrôleur.

–Utilisation d'une LED[2] qui supporte 700mA maximum, puissance 3W : Flux lumineux de 51,6Lm.

–Utilisation d'un gradateur, permet d'adapter la luminosité de la LED grâce à un potentiomètre.

2 LED : diode électroluminescente

2. Planning de répartition des tâches effectuées

Semaines / Tâches	S37-S37	S38-S38	S39-S39	S40-S40	S41-S41	S42-S43	S43	S44 Vacance
Analyse et développement								
Typon								
Gravure /Soudure								
Test Tension 5V								
Programmation								
Test et validation								
Rédaction du dossier								
Oral								

Planning prévisionnel	
Planning réel	

3. Analyse générale du projet

La lampe torche sert à éclairer le soir ou pendant la nuit. On peut régler l'intensité de la lampe torche afin d'éclairer un livre sous la tente ou si on est dehors.

Schéma fonctionnel de niveau 1

Le fonctionnement principal peut se représenter ainsi :

Illustration 1: Schéma fonctionnel de niveau 1 du système

La fonction principale de ce projet est d'éclairer tous les lieux sombres. Pour cela, on utilise une lampe torche à LED avec une LED de luminescence de 51,6Lm.

Schéma fonctionnel de niveau 2

Ce schéma fonctionnel représente la chaîne de gestion de la lampe torche.

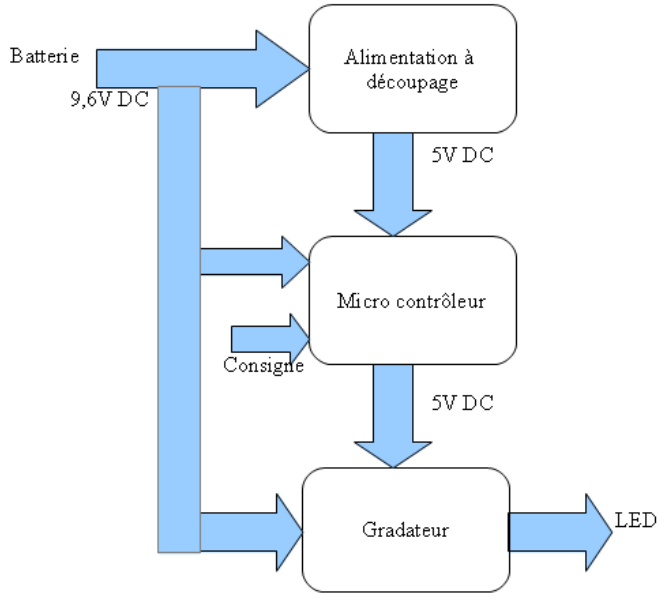


Illustration 2: Schéma fonctionnel de niveau 2 du système

Notre projet est alimenté par une batterie qui fournit une tension de 9,6V continu. Avec cette tension, on alimente une alimentation à découpage afin d'obtenir une tension continue de 5V. Cette tension permet d'alimenter le micro-contrôleur. Dans le micro-contrôleur, on mesure la tension de la batterie afin que lorsque cette tension est trop faible on fasse clignoter la LED. La tension de consigne permet de faire varier la luminosité de la LED.

4. Première carte

4.1. Analyse

4.1.1. Alimentation à découpage

Nous utilisons une batterie pour alimenter notre montage, cependant nous utilisons un micro-contrôleur alors on rajoute une alimentation à découpage.

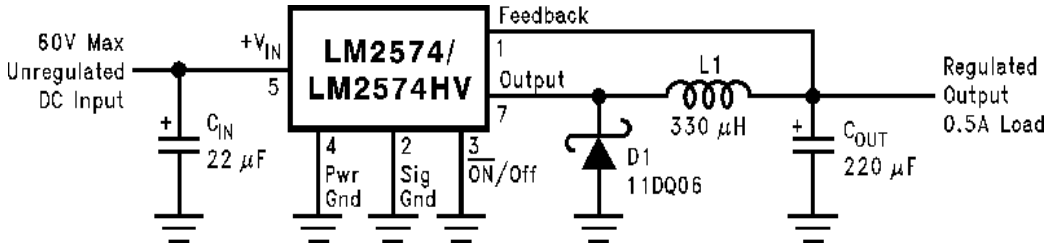


Illustration 3: Alimentation à découpage

Nous utilisons cette alimentation à découpage, car elle permet d'avoir en sorti une tension régulée fixe de 5V continue et un courant de 0,5A. Cette alimentation accepte 60V continue maximum, ce qui nous permet de l'alimenter avec les 9,6V fournit par la batterie.

Donc on prend cette alimentation à découpage pour alimenter notre micro-contrôleur, car elle délivre 5V continue fixe. Si la tension de 5V n'était pas fixe, on détériorerait le micro-contrôleur.

La diode Schottky ne modifie pas la tension et le courant de sorti. Elle a une faible chute de tension qui permet d'avoir une tension continue. De plus, son temps de réponse est très faible ce qui est bien, car nous avons une fréquence de 50kHz.

Le feedback permet de réguler la tension de sortie.

Le condensateur Cout sert à filtrer la tension de sortie afin qu'elle soit fixe à 5V.

La bobine L1 permet de stocker l'énergie magnétique ce qui permet un fonctionnement en continu, car elle restitue l'énergie magnétique.

4.1.2. Alimentation de la LED

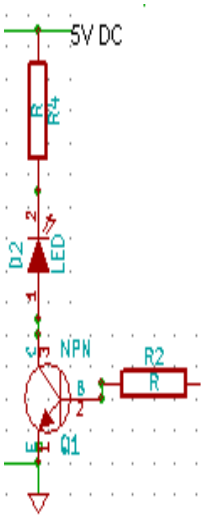


Illustration 4: Alimentation de la LED

On utilise une LED de puissance 3W qui supporte un courant de 700mA maximum. Aux bornes de la LED, on a une tension de 2.8V maximum. Nous avons choisi de mettre une tension de 2,1V. Aux bornes du transistor, on a une tension de 0,7V donc on a une tension de 2,2V aux bornes de la résistance R4.

Calcul de R4 :

$U4 = R4 * I4 \; donc \; R4 = U4 \div I4 = 3{,}14\,\Omega$

On prendra une résistance de puissance de 2.2Ω et 1W.

Calcul de R2 :

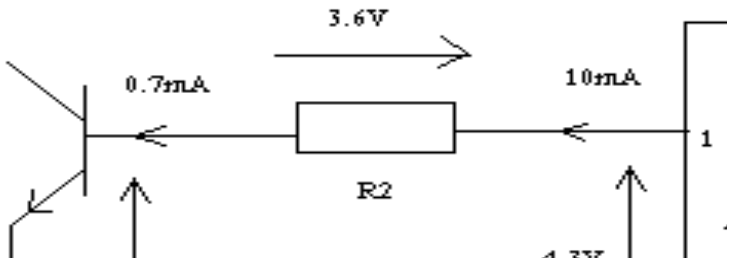


Illustration 5: Schéma de R2

$$R2 = \frac{3.6}{0.007} = 514.285\Omega$$

On prendra une résistance de 470Ω.

4.1.3. Pont diviseur

Les résistances R1 et R3 représentent un pont diviseur, on fixe R1 et R3 à 10kΩ.

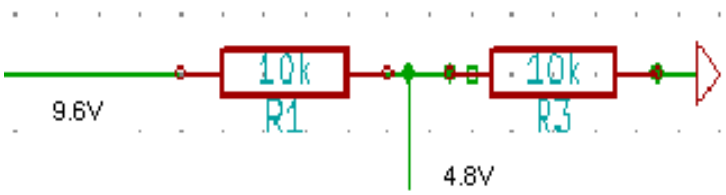


Illustration 6: Pont diviseur

En entrée de notre pont diviseur, on a la tension de la batterie qui est de 9.6V. On veut la moitié de la tension de la batterie donc pour se faire on met deux résistances de même valeurs. On prend que la moitié de la tension de la batterie car on ne doit pas dépasser une tension de 5V aux bornes du micro-contrôleur.

$$V3 = \frac{R3}{R1+R3} * Vbatt = \frac{10000}{20000} * 9.6 = 4.8\text{V}$$

La tension de 4.8V va sur la patte 4 du micro-contrôleur. Cette tension permet de mesurer la tension de la batterie afin que si la batterie est déchargée, on fait clignoter la LED.

4.1.4. ATtiny 10

On utilise un ATtiny10 car, on veut un micro-contrôleur qui est en CMS et qui n'a pas besoin de beaucoup de pattes.L'ATtiny10 n'a que 6 broches.

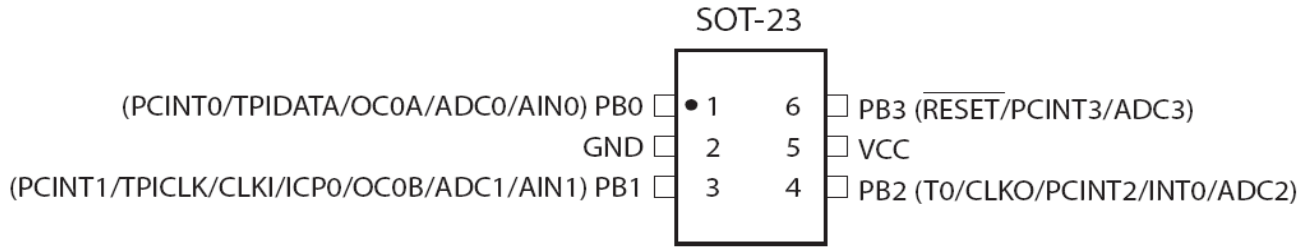


Illustration 7: Connections ATtiny10

Pour notre projet, on utilise la patte 1 pour les données, la patte 2 sera à la masse, la patte 3 sera notre horloge, on n'utilise pas la patte 4, on alimente le micro-contrôleur en +5V sur la patte 5, et la patte 6 permettra de réinitialiser le micro-contrôleur.

4.1.5. Connecteur ISP-6pattes

Afin de programmer l'ATtiny10, on utilise un connecteur ISP de six pattes.

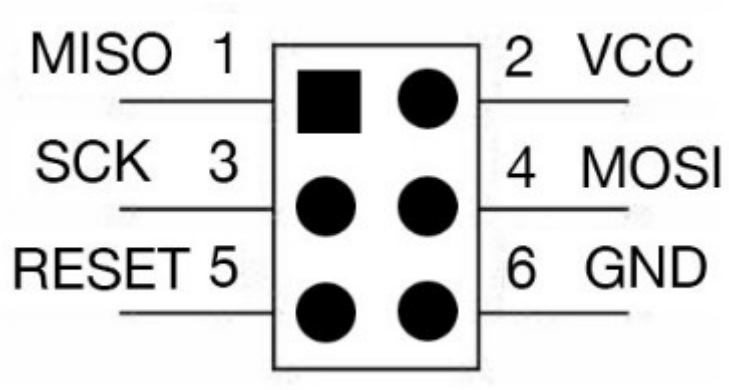


Illustration 8: Connecteur ISP-6 pattes

Pour programmer le micro-contrôleur, on doit connecter correctement l'ATtiny10 au connecteur ISP.

AVRISP mkII connector	Target pin name	ATTINY10 pin number
1 MISO	TPIDATA	1
2 VCC	VCC	5
3 SCK	TPICLK	3
4 MOSI	–	–
5 RESET	RESET	6
6 GND	GND	2

Illustration 9: Connexion ATtiny10 avec Connecteur ISP

4.1.6. Réalisation de la carte 1

On a fait le schéma de la carte avec le logiciel KiCad.

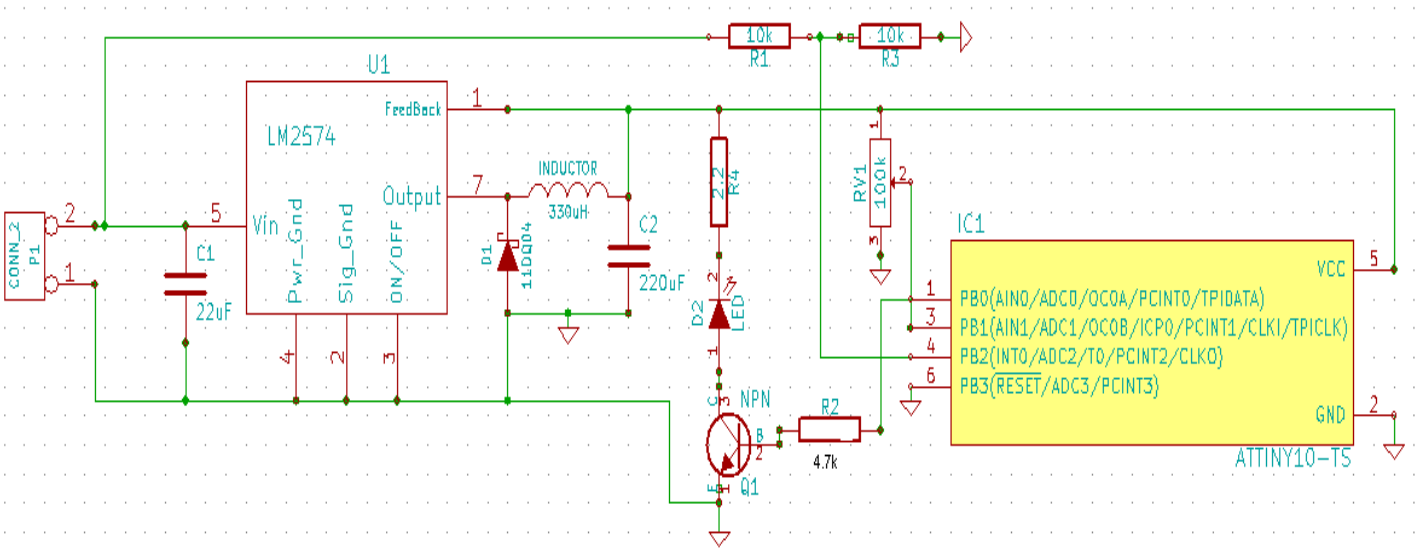


Illustration 10: Schéma structurel de la carte

On a créé le typon aussi avec KiCad.

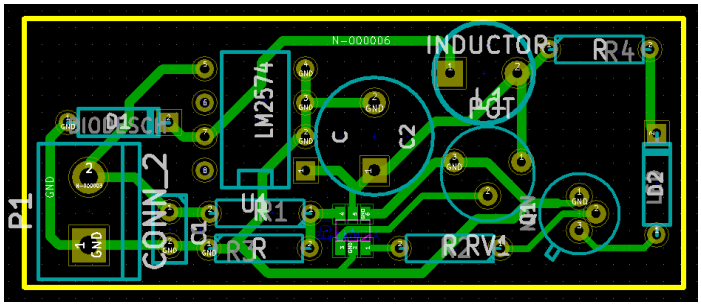


Illustration 11: Typon de la carte 1

Nous avons remarqué que l'empreinte du potentiomètre n'était pas bonne car elle était trop petite, et nous avons oublié de mettre un connecteur ISP-6 pattes. Donc sur notre carte, nous avons branché le potentiomètre et le connecteur ISP grâce à des câbles. Nous avons soudé le composant ATtiny10 au fer à souder. Nous n'avons pas prévu de connectique pour programmer, car on pensait trouver un support CMS sur internet. Donc nous avons commandé une carte annexe pour programmer le composant et ensuite le souder sur la carte test. Mais en recevant la carte annexe, nous nous sommes aperçu que cette carte disposait bien d'un support pour ATtiny10 mais faisant l'objet d'une partie d'un kit d'un projet complètement différent. Nous avons pensé pouvoir dessouder le support mais n'ayant aucune documentation à notre disposition, nous n'étions pas en mesure de pouvoir le réutiliser.

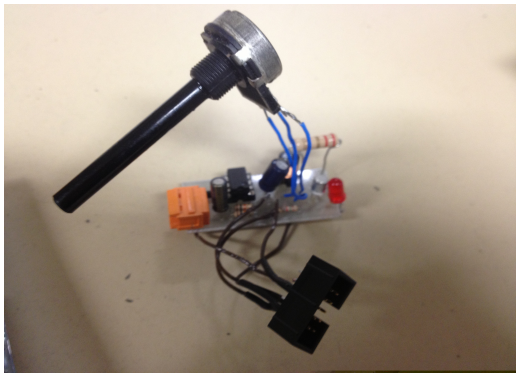


Illustration 12: Carte 1

Finalement nous avons opté par relier le connecteur ISP 6 directement à la carte test pour pouvoir programmer. Grâce au connecteur ISP – 6 pattes que l'on a rajouté sur la carte, on a pu connecter la carte à l'ordinateur grâce à la carte de programmation.

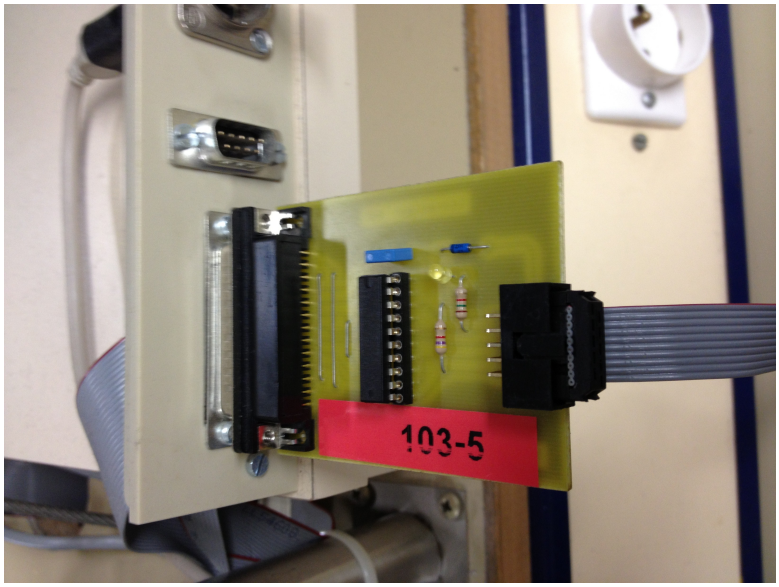


Illustration 13: Carte de programmation

Grâce à cette carte de programmation, on peut se connecter via le cordon ISP à la carte lampe torche. On utilise le logiciel de programmation CodeVisionAVR pour programmer le composant ATtiny10. On a dû télécharger une nouvelle version pour ce composant. Grâce au nouveau logiciel on peut programmer l'ATtiny10, cependant on a constaté que le logiciel ne peut pas programmer l'ATtiny10 via un cordon ISP.

Donc on recherche dans les méthodes de programmation de l'ATtiny10 sur CodeVisionAVR, on a vu que l'on pouvait le programmer via de l'USB.

5. Deuxième carte

5.1. Connecteur OLIMEX

Sur le logiciel CodeVisionAVR, on a choisi de prendre une autre méthode de programmation. On a choisi de prendre un convertisseur Atmel AVRISP MkII qui utilise de l'USB. Les principaux intérêts d'utiliser l'USB sont sa vitesse transmission et son type de connectique présent sur tous les types d'équipement.

On a trouvé sur le site de FARNELL un connecteur OLIMEX-AVR-ISP-Mk2 (code commande : 2144332).

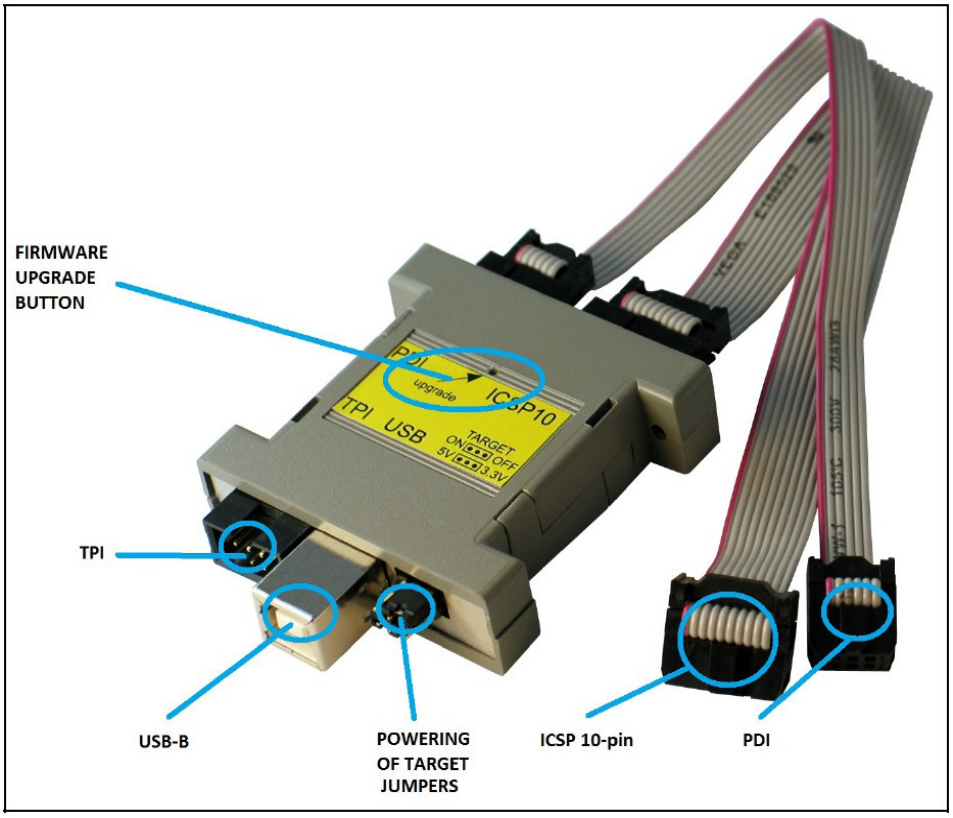


Illustration 14: Connecteur OLIMEX

Ce connecteur permet de convertir le ISP-10 pattes ou le PDI, en TPI ou en USB. Pour notre projet, nous utiliserons le ISP-10 pattes pour le relier à la carte lampe torche. Et pour le relier à l'ordinateur, on utilisera le port USB.

5.2. Connecteur TPI

Pour programmer l'ATtiny10, on utilise un connecteur TPI qui est un connecteur 6 pattes.

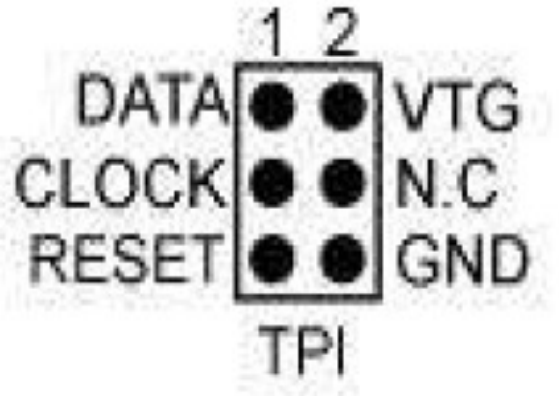


Illustration 15: Connecteur TPI

Afin de programmer le micro-contrôleur, on doit connecter correctement l'ATtiny10 au connecteur TPI.

Pattes TPI	Nom pattes	Pattes ATtiny10
1 DATA	TPIDATA	1
2 VTG	VCC	5
3 CLOCK	TPICLK	3
4 N.C	–	–
5 RESET	RESET	6
6 GND	GND	2

Tableau 1: Configuration des pattes

5.3. CMS[3]

5.3.1. 1er étape

Illustration 16: Pâte à braser

On dépose de la patte à braser grâce à une doseuse manuelle sur les emplacements des pattes de l'attiny10. La pâte ou crème à braser est utilisée pour le brasage des CMS. Elle est constituée d'un alliage et d'un flux dont les caractéristiques sont identiques à ceux utilisés pour une vague[4] ou avec un fer à souder. L'alliage est composé de billes sphériques dont la taille de 20 à 160 µm.

5.3.2. 2e étape

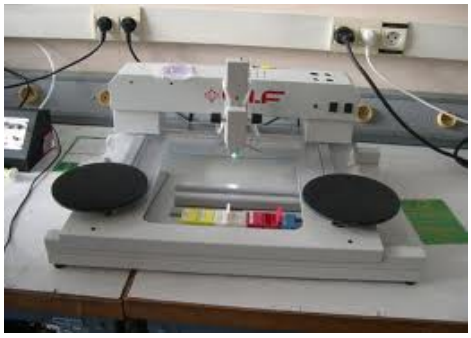


Illustration 17: Placement composant CMS

Ensuite on pose les composants CMS, dans notre cas on pose l'ATtiny10 sur son emplacement grâce à l'outil ci-dessus qui permet d'être plus précis dans ses déplacements et permet de tourner le composant en toute simplicité.

5.3.3. 3e étape

Et enfin on passe la carte dans le four à CMS, qui fixe les pattes des composants via la patte à braser. Le four à un cycle pré-défini qui comporte plusieurs phases : la première phase est une monté en température à 180 degrés puis la température est stabilisé quelques secondes ensuite la température du four augmente à 250 degrés pour souder les composants.

Illustration 18: Four à refusion

3 CMS : Composant Monté en Surface
4 Technique de soudage industrielle

5.4. Modifications apportées

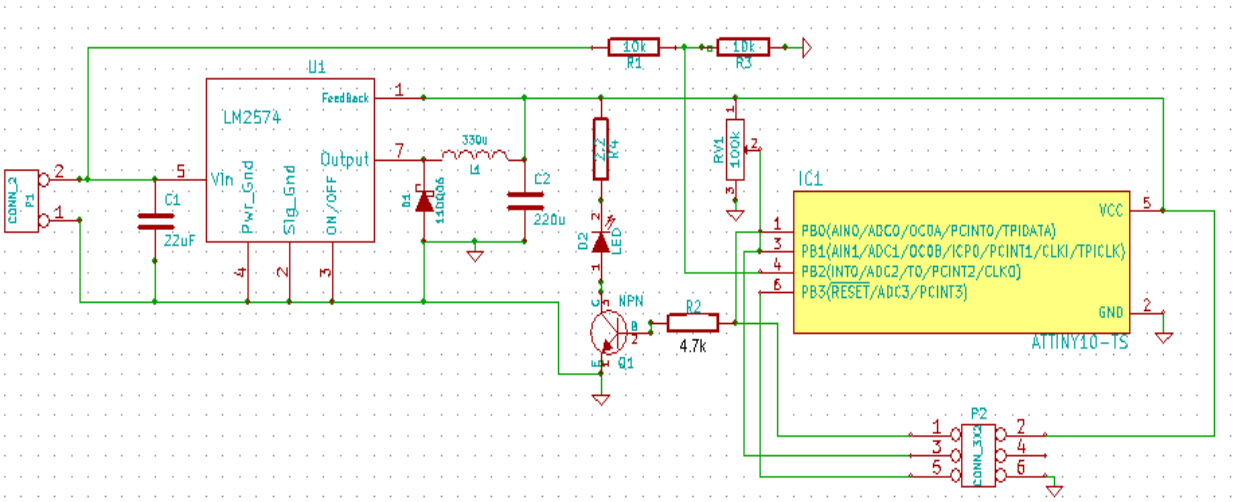


Illustration 19: Schéma structurel deuxième carte

Sur le schéma structurel nous avons rajouté le connecteur ISP – 6 pattes et nous l'avons relié correctement à l'ATtiny10.

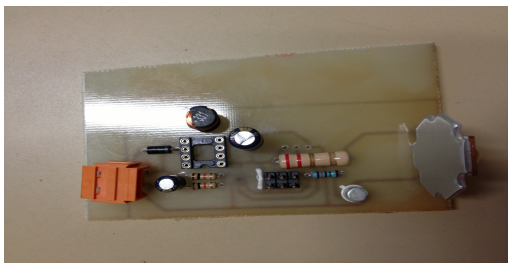


Illustration 20: Deuxième carte

Nous avons constaté que l'empreinte du potentiomètre n'était toujours pas la bonne.

Nous avons testé que nous obtenons bien la tension de 5V qui doit alimenter le micro-contrôleur et la LED.

Cependant, nous ne pouvons pas finir cette carte car on n'a toujours pas reçu la commande du connecteur OLIMEX, donc nous avons abandonné la programmation de l'ATtiny10.

6. Troisième carte

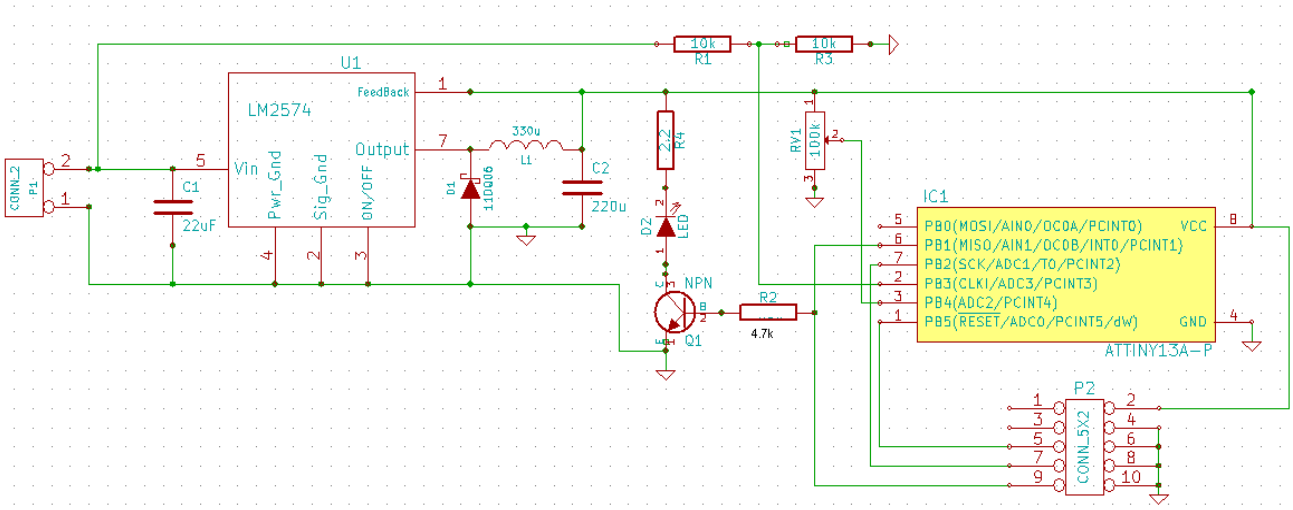


Illustration 21: Schéma structurel troisième carte

Les modifications apportées par rapport à la carte précédente sont que l'on a remplacé l'ATtiny10 par l'ATtiny13.

6.1. ATtiny 13

On utilise un ATtiny13 car, c'est un composant qui peut se programmer grâce à de l'ISP. Ce micro-contrôleur a plus de broches que l'ATtiny10, cependant il a que 8 broches.

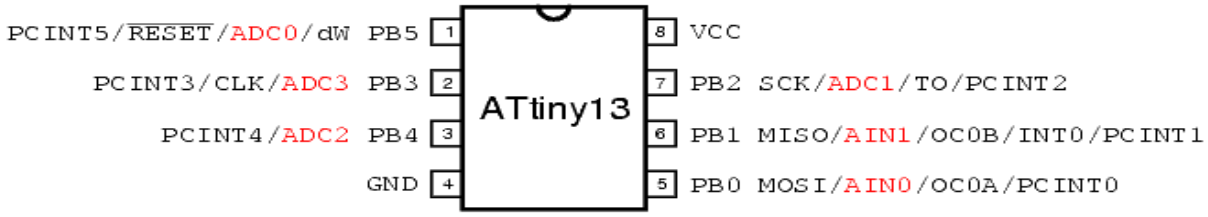


Illustration 22: Connexion ATtiny13

Pour notre projet, on utilise la patte 6 pour les données, la patte 4 sera à la masse, la patte 7 sera notre horloge, on n'utilise pas la patte 5, on alimente le micro-contrôleur en +5V sur la patte 8, et la patte 1 permettra de réinitialiser le micro-contrôleur.

6.2. Connecteur ISP – 10 broches

Afin de programmer l'ATtiny13 on utilise un connecteur ISP – 10 broches.

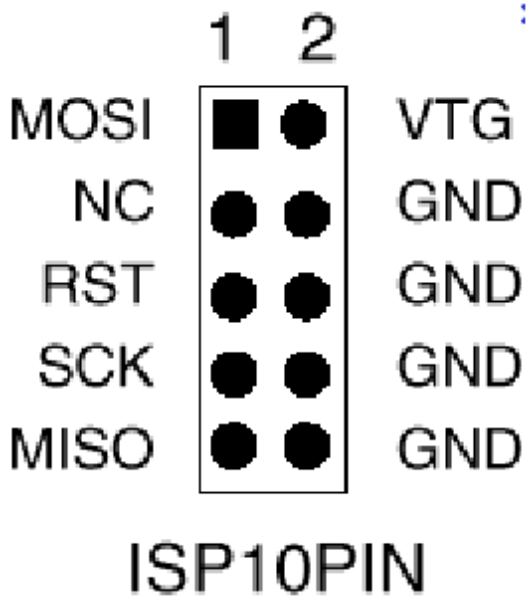


Illustration 23: Connecteur ISP-10 broches

Pour correctement programmer l'ATtiny13, on doit connecter les broches ainsi :

Broches ISP-10	Nom Broche	Broches ATtiny13
1 MOSI	–	N.C
2 VTG	VCC	8
3 N.C	–	N.C
4 GND	GND	MASSE
5 RST	RESET	1
6 GND	GND	MASSE
7 SCK	TPICLK	7
8 GND	GND	MASSE
9 MISO	TPIDATA	6
10 GND	GND	MASSE

Tableau 2 : Configuration des broches

6.3. Réalisation de la carte

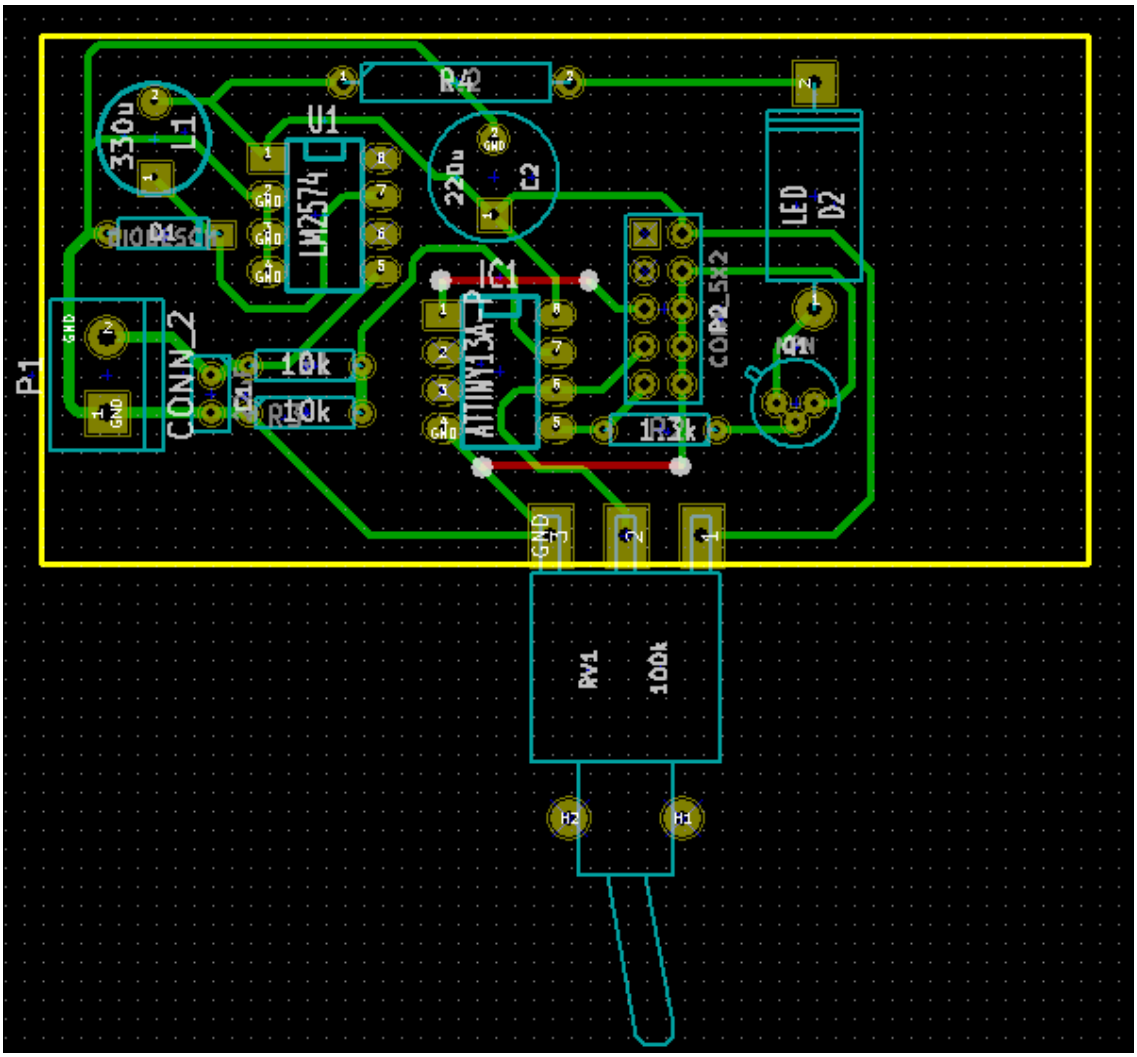


Illustration 24: Typon troisième carte

Sur le typon, on a constaté qu'il y avait de mauvaises liaisons entre l'ATtiny13 et le connecteur ISP-10 broches. Les liaisons des broches 5 et 6 devraient être sur les broches 6 et 7 de l'ATtiny13. Cette erreur est due à la NetList car en actualisant celle-ci, la NetList conserve les défauts de brochage de l'ancienne carte.

On a constaté que afin de programmer, il fallait connecter la broche 5 de l'ATtiny13 à la broche 1 du connecteur ISP.

Nous avons bien la tension de 5V qui doit alimenter le micro-contrôleur et la LED.

7. Quatrième carte

On a ajouté la liaison entre la broche 5 de l'ATtiny13 et la broche 1 du connecteur ISP.

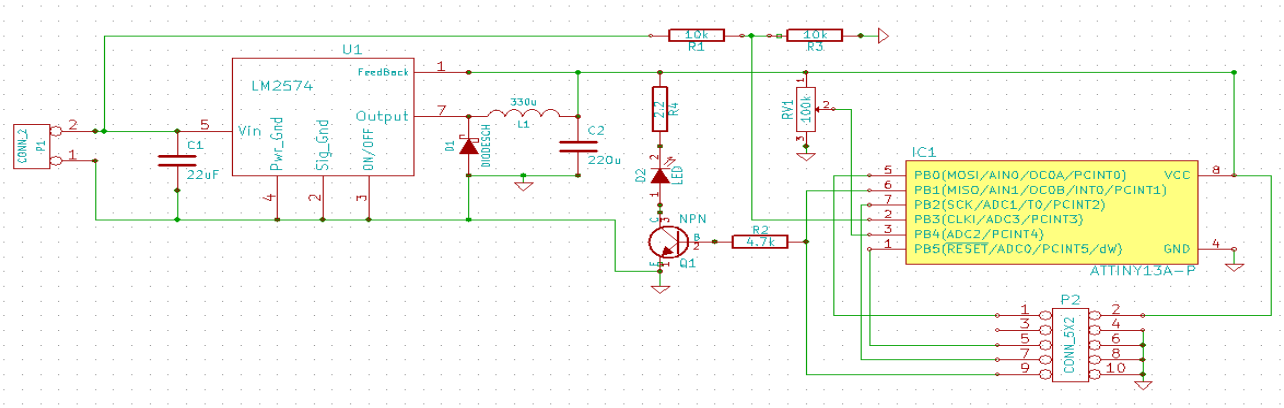


Illustration 25: Schéma structurel de la quatrième carte

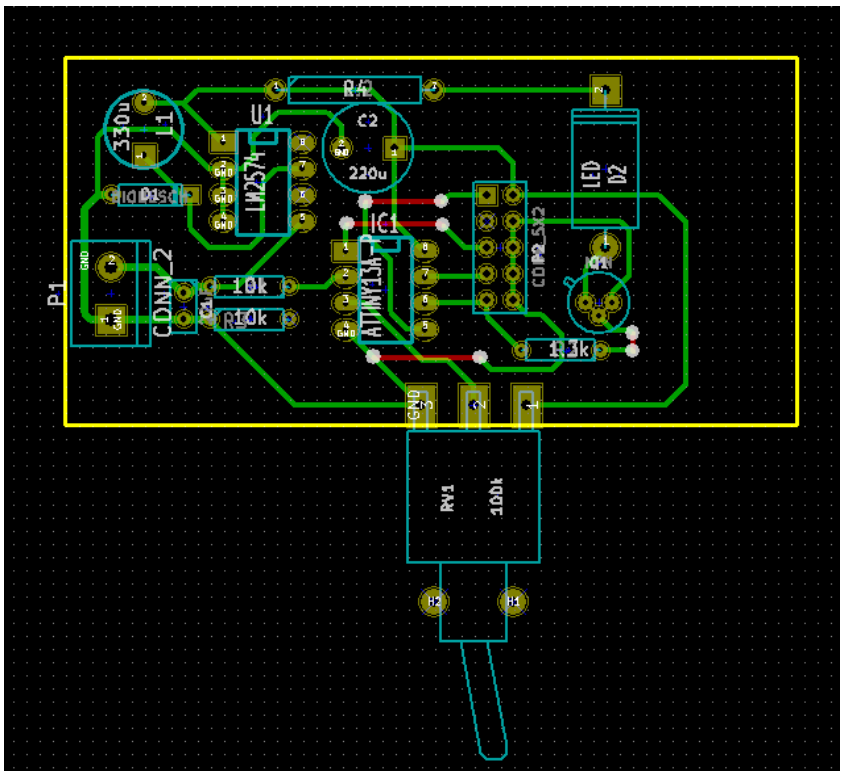
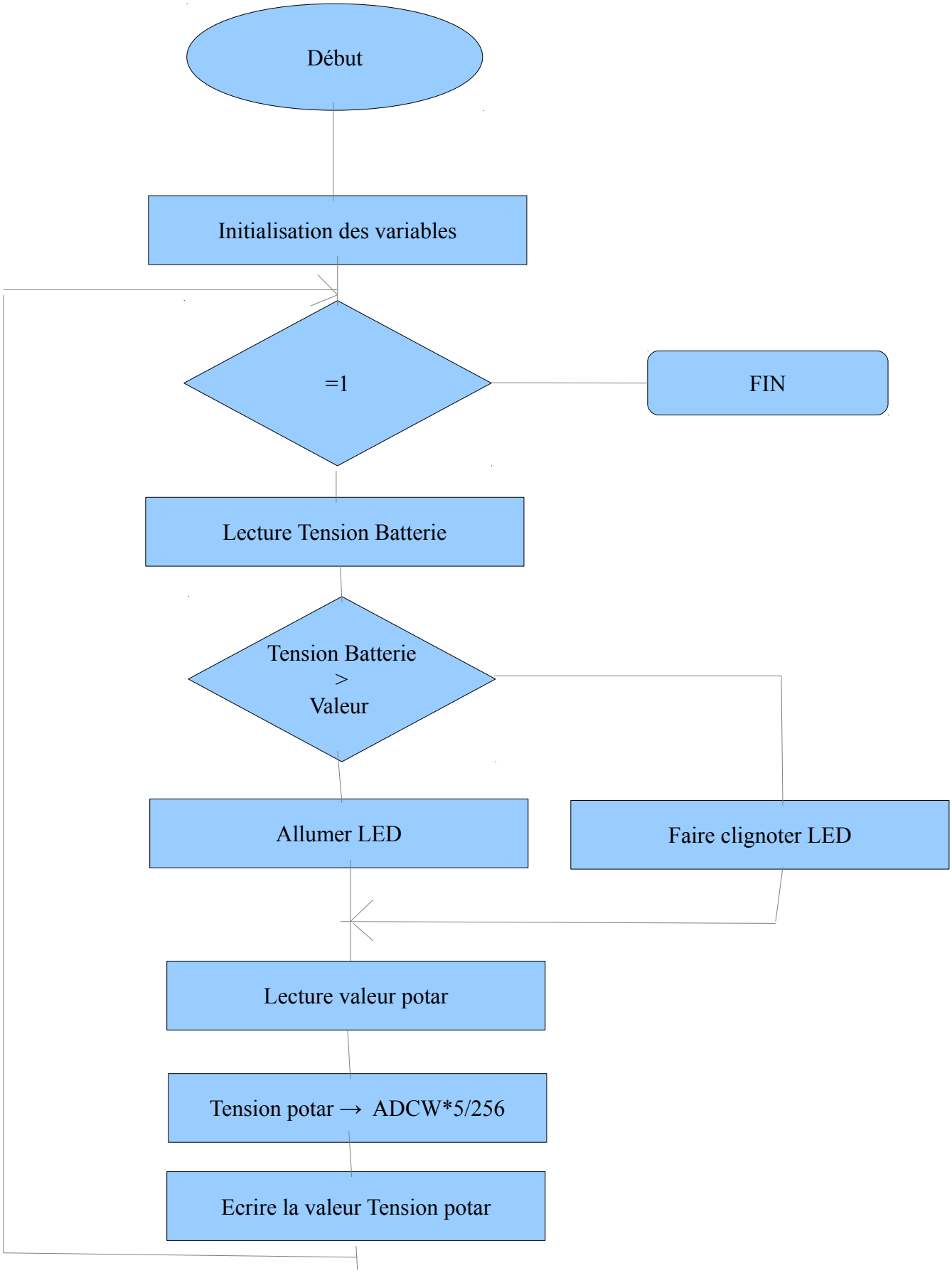


Illustration 26: Typon carte 4

On a bien la tension de 5V en sortie de l'alimentation à découpage.

Lors de la programmation de l'ATtiny13, on a constaté que l'on pouvait le programmer qu'une seule fois. Pour résoudre ce problème, il faut dans Code Vision Avr modifier dans le menu Project / Configure/ afterBuild et il faut décocher l'option Programm FuseBit.

7.1. Ordinogramme



Conclusion

Ce projet nous permis de comprendre le fonctionnement de l'ATtiny10 et l'ATtiny13. Nous avons réutilisé le logiciel KiCad afin de réaliser les schémas électriques des cartes et on a découvert le logiciel CodeVisionAVR qui nous permet de programmer les micro-contrôleurs. On a appris à mettre en œuvre les composants CMS, grâce à la salle de réalisation. On a compris le mode de communication entre l'ATtiny les compilateurs et les interfaces de programmation.

Ce projet nous permis de mettre en œuvre nos acquis en électronique et en informatique, et nous avons su être autonomes. Nous avons acquit une stratégie de solution face à tous les problèmes rencontrés lors de ce projet.

Nous remercions Monsieur LEQUEU qui nous a apporté des solutions techniques face aux problèmes rencontrés. Nous remercions aussi le magasinier Monsieur VAUTIER qui nous a fourni tous les composants et la carte de programmation.

Résumé

Tout d'abord, dans ce dossier, on a fixé notre cahier des charges où nous avons présenté toutes les contraintes de notre sujet.

Puis, on a fait une analyse générale du projet afin de montrer le fonctionnement principal de la lampe torche. On a montré les différents éléments que comporte la carte de la lampe torche.

Ensuite, plusieurs cartes qui nous ont permis de comprendre le fonctionnement d'une alimentation à découpage qui fournit en sortie une tension de 5V.

Puis, on a vu le fonctionnement de l'ATtiny10, ainsi que les différents connecteurs que l'on doit utiliser pour programmer un ATtiny.

L'ATtiny10 nous a permis de souder un composant CMS grâce à la salle de réalisation.

On a mis en œuvre une LED de puissance, et on a vu son fonctionnement.

Enfin, nous avons étudié le mode de fonctionnement de l'ATtiny13.

Index des illustrations

Illustration 1: Schéma fonctionnel de niveau 1 du système....7
Illustration 2: Schéma fonctionnel de niveau 2 du système....7
Illustration 3: Alimentation à découpage....8
Illustration 4: Alimentation de la LED....9
Illustration 5: Schéma de R2....9
Illustration 6: Pont diviseur....10
Illustration 7: Connections ATtiny10....10
Illustration 8: Connecteur ISP-6 pattes....11
Illustration 9: Connexion ATtiny10 avec Connecteur ISP....11
Illustration 10: Schéma structurel de la carte....12
Illustration 11: Typon de la carte 1....12
Illustration 12: Carte 1....13
Illustration 13: Carte de programmation....13
Illustration 14: Connecteur OLIMEX....14
Illustration 15: Connecteur TPI....15
Illustration 16: Pâte à braser....16
Illustration 17: Placement composant CMS....16
Illustration 18: Four à refusion....16
Illustration 19: Schéma structurel deuxième carte....17
Illustration 20: Deuxième carte....17
Illustration 21: Schéma structurel troisième carte....18
Illustration 22: Connexion ATtiny13....18
Illustration 23: Connecteur ISP-10 broches....19
Illustration 24: Typon troisième carte....20
Illustration 25: Schéma structurel de la quatrième carte....21
Illustration 26: Typon carte 4....21

Index des tables

Tableau 1: Configuration des pattes.....15
Tableau 2 : Configuration des broches.....19

Bibliographie

[1] . *FARNELL*, , [En ligne]. (Page consultée le 24 octobre 2012) <http://fr.farnell.com>
[2] **OLIMEX Ltd**, "*AVR-ISP-MK2 programmer*", , juillet 2012.
[3] . *Troubleshooting*, , [En ligne]. (Page consultée le 24 octobre 2012) <http://www.nongnu.org/avrdude/user-manual/avrdude_20.html#Troubleshooting>
[4] . *Programming an Attiny10 with AVRISP mkII and AVR Studio 5*, 27 janvier 2012, [En ligne]. (Page consultée le 24 octobre 2012) <http://minisystem.blogspot.fr/2012/01/programming-attiny10-with-avrisp-mkii.html>
[5] **ATMEL**, "*ATtiny4/5/9/10*", ATMEL, 2011.