

Rapport de projet tutoré 2^{ème} année

Horloge à LED

Pilotée par un RTC en bus I2C

GRATADE Diana
TUILARD Benjamin
Groupe Q1 / P2 – 2011/2013

Professeurs référents:

- PAPAZIAN Patrick
- LEQUEU Thierry
- BESSE Dominique

Rapport de projet tutoré 2^{ème} année

Horloge à LED

Pilotée par un RTC en bus I2C

GRATADE Diana
TUILARD Benjamin
Groupe Q1 / P2 – 2011/2013

Professeurs référents:

- PAPAZIAN Patrick
- LEQUEU Thierry
- BESSE Dominique

Sommaire

Remerciements.....	5
Introduction.....	6
1.Présentation du sujet.....	7
1.1.Son fonctionnement global.....	7
1.2.Rétrospective sur le projet.....	10
1.3.Étude du cahier des charges.....	14
1.4.Organisation du projet.....	14
1.5.Planning réel et prévisionnel.....	15
2.Description technique de l'horloge.....	16
2.1.Le bus I ² C.....	16
2.2.L'ATmega : le microcontrôleur.....	18
2.3.Le RTC : Real Time Clock.....	19
2.4.Les drivers à LEDs.....	19
2.5.La CPLD.....	20
3.Mise en œuvre et réalisation de l'horloge.....	22
3.1.La partie électronique : réalisation des typons.....	22
3.2.La partie informatique : programmation.....	26
Conclusion.....	28
Résumé.....	29
Index des illustrations.....	30
Index des mots clefs du projet.....	31
Glossaire.....	32
Bibliographie.....	33
Annexes.....	34

Remerciements

Nous souhaitons tout d'abord, avant de commencer ce rapport, remercier l'IUT de nous permettre d'effectuer ce projet tutoré et de nous donner l'opportunité de mettre en application nos connaissances sur un projet concret.

Nous remercions ensuite nos professeurs respectifs Mr LEQUEU Thierry et Mr BESSE Dominique, qui nous ont donné la possibilité de mener ce projet personnel en binôme malgré des groupes de TP différents et qui ont su faire partager leurs savoir-faire notamment en matière de conception électronique.

Nous tenons également à remercier Mr VAUTIER Richard pour tous ses conseils et le soutien qu'il nous a apporté depuis le début de notre projet.

Introduction

Dans le cadre du cours d'étude et réalisation se déroulant au semestre 3, notre binôme a eu l'occasion de mener un projet venant de sa propre initiative.

Ce projet personnel est basé sur la réalisation et plus particulièrement sur la conception électronique d'une horloge à LED¹. Il s'agit là d'un projet qui a été débuté lors du semestre dernier par l'un des membres du binôme, qui avait alors effectué la majorité des recherches documentaires en matière de solutions technologiques, mais qui n'était pas parvenu jusqu'à mettre en application ses recherches. Il nous faut donc à présent réaliser cette horloge.

Dans ce rapport, nous expliquerons d'abord en quoi consiste cette horloge à LED. Nous présenterons donc le projet de manière globale en évoquant également nos attentes où encore le cahier des charges.

Nous détaillerons ensuite le fonctionnement de l'horloge dans un registre plus technique, qui nous permettra d'approfondir nos explications sur les composants essentiels qui constituent celle-ci.

Dans une dernière partie, on s'intéressera à la phase de conception de l'horloge, aussi bien d'un point de vue électronique avec la réalisation des différentes cartes, que d'un point de vue informatique en passant par la programmation de certains composants.

On finira par conclure sur notre projet, en exposant le bilan de celui-ci et nous ferons part de nos impressions.

1 LED : Light-Emitting Diode ou diode électroluminescente

1. Présentation du sujet

1.1. Son fonctionnement global

Le projet consiste à élaborer une horloge à LED fonctionnant avec un composant permettant de générer un signal d'horloge² appelé RTC³ et communiquant grâce à un langage spécifique pour l'échange de données, sous forme d'un protocole de questions/réponses.

L'horloge affichera l'heure sous 3 formats différents : traditionnel, binaire et digital.

Une autre fonctionnalité de l'horloge sera de garder l'heure en mémoire lorsque celle-ci subira une coupure d'alimentation. Ainsi, grâce à une pile interne au RTC, le temps continuera de s'écouler au sein de l'horloge, de façon à ce que lorsqu'on alimentera à nouveau l'horloge, l'heure affichée soit synchronisée avec le temps qui a continué de s'écouler durant la coupure.

Enfin, on souhaite également donner la possibilité à l'utilisateur de régler l'heure.

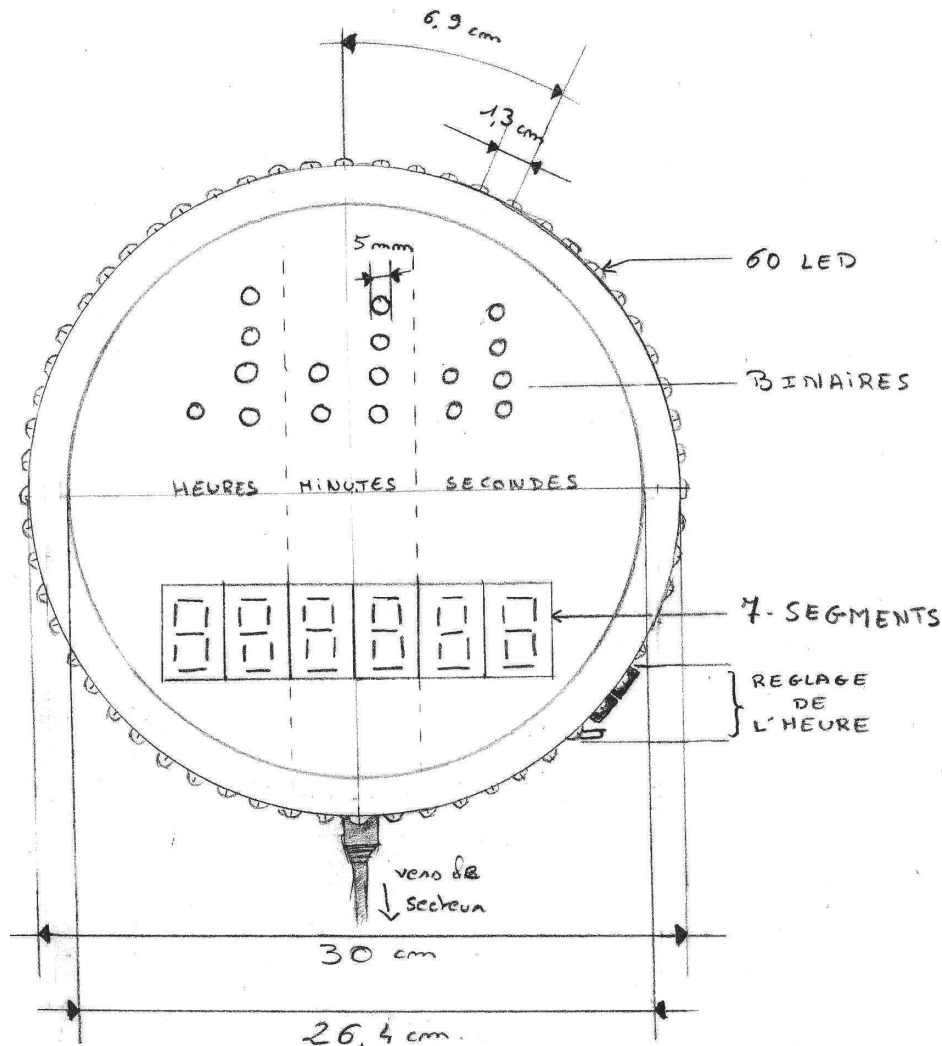


Illustration 1: Croquis de l'horloge à réaliser, fait par les rédacteurs

2 Signal d'horloge : signal électrique de type créneau variant périodiquement entre 2 états appelés état haut et état bas, correspondant à l'activation ou non d'un indicateur comme une ampoule.

3 RTC : Real Clock Time (Horloge à Temps Réel)

1.1.1. L'affichage traditionnel

L'affichage traditionnel sera réalisé par 72 LEDs qui serviront d'une part de contour pour délimiter la taille de notre horloge et d'autre part à afficher les heures et les minutes.



Illustration 2: Exemple d'affichage traditionnel à base de LEDs[1]

On distinguera deux groupements de LEDs, comme le présente l'illustration 2 ci-contre, soit :

- un premier groupement de 12 LEDs sera destiné à l'affichage des heures.
- un second groupement de 60 LEDs servira à afficher les minutes.

Le déclenchement des LEDs sera piloté par un composant programmable appelé CPLD⁴. Celles-ci s'allumeront alors à tour de rôle afin que l'utilisateur puisse visualiser l'heure, qui sera envoyée par un microcontrôleur.

Le processus de fonctionnement de cet affichage dit traditionnel, sera détaillé dans la suite de ce rapport.

1.1.2. L'affichage digital

Ce type d'affichage sera réalisé à l'aide d'afficheurs 7-segments, comme sur l'illustration 3 ci-dessous, qui permettront de visualiser l'heure dans son intégralité (c'est-à-dire : les heures, les minutes et les secondes).

Le fonctionnement des afficheurs 7-segments, au nombre de 6 dans notre projet comme indiqué sur le croquis de l'illustration 1, est dépendant du fonctionnement de l'affichage binaire. En effet, l'affichage digital réalise une retranscription de l'affichage binaire sous un autre format.



Illustration 3: Horloge utilisant un affichage digital

⁴ CPLD : Complex Programmable Logic Device, en français Circuit Logique Programmable

1.1.3. L'affichage binaire

L'affichage binaire nécessite un langage informatique particulier qui utilise des bits⁵ pour convertir un nombre décimal, dit un nombre en base 10, en un nombre en base 2, constitué exclusivement de 0 et de 1.

Le nombre binaire **B** converti du nombre décimal **D** est alors composé d'un certain nombre de bits. Chaque bit est situé à un rang **R** et on lui associe une puissance de 2^R (cf. illustration 4).

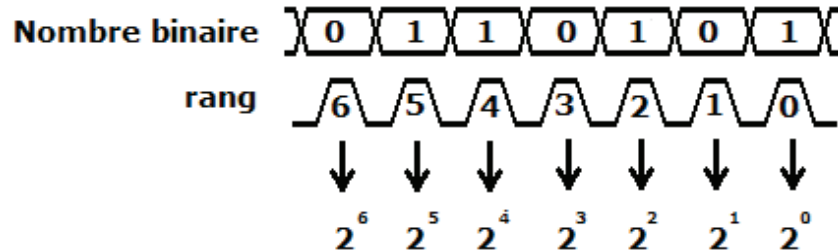


Illustration 4: Décomposition d'un nombre binaire de 7bits, faite par les rédacteurs

Pour convertir un nombre binaire **B** en un nombre décimal **D**, on additionne toutes les puissances de 2 élevée au rang **R** lorsqu'un bit a pour valeur 1. Dans l'exemple ci-dessus, le nombre binaire $(0110101)_2$ correspond à un nombre décimal égal à 53, soit :

$$0 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 32 + 16 + 4 + 1 = 53$$

Cependant, pour cet affichage, on utilise une variante du langage binaire appelé langage binaire naturel (DCB). Pour chaque chiffre d'un nombre décimal, le langage lui associe un groupement de 4 bits, que l'on mettra bout à bout pour former le nombre binaire. Chaque groupement formera un nombre binaire correspondant à un nombre décimal compris entre 0 et 9 (cf Annexe 1). La différence avec le langage binaire est que l'on converti chaque chiffre composant le nombre décimal au lieu de le convertir tel que.

Dans l'illustration 5 ci-contre, on peut voir que les secondes valent 19. Celles-ci correspondent à un nombre en binaire DCB de $(0001\ 1001)_2$ où le bit vaut 1 quand la LED est allumée sinon 0.

Soit le groupement $(0001)_2$ correspondant aux dizaines des secondes soit 1 et le groupement $(1001)_2$ correspondant aux unités c'est-à-dire 9. En détaillant la conversion, on obtient :

$$\begin{array}{r}
 (0001 \mid 1001) \\
 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \mid 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \\
 1 \mid 8 + 1 = 9
 \end{array}$$

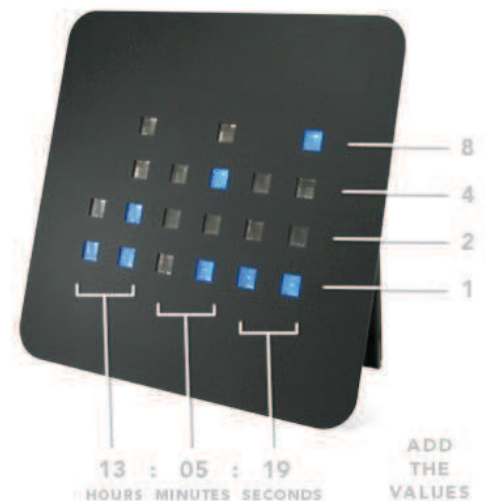


Illustration 5: Horloge binaire vendue dans le commerce[2]

⁵ Bit :Chiffre binaire valant 0 ou 1. C'est une unité de mesure informatique qui décrit l'état d'une variable pour savoir si elle est activée ou non

1.2. Rétrospective sur le projet

1.2.1. Le choix des composants

Le projet a été débuté au semestre 4 de l'année 2012. Durant ce semestre, l'essentiel des recherches de solutions et le choix des composants ont été fait. L'horloge est prête au montage et les composants sont déjà commandés.

Plusieurs fonctions de l'horloge sont indépendantes et exécutées par différents composants dont le choix a été justifié l'année précédente.

D'une part, pour pouvoir générer un signal d'horloge précis nous avons choisi un RTC (Real Time Clock) DS1307. Son rôle est de compter l'heure une fois qu'il a été initialisé. L'avantage d'un tel composant est qu'il possède une continuité de fonctionnement même lorsque l'alimentation générale de l'horloge n'est pas branchée. Cette continuité est assurée grâce à une pile interne possédant une durée de vie de dix ans.

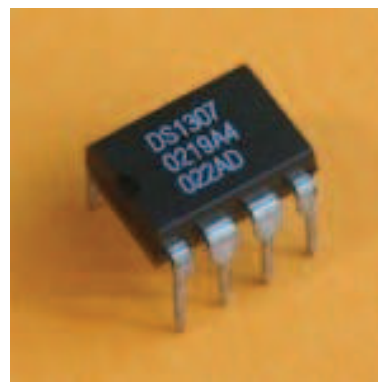


Illustration 6: un RTC DS1307 utilisé dans le projet[3]

L'inconvénient d'un tel composant est la communication imposée en bus⁶ I2C⁷. Ce bus de communication établit une liaison entre plusieurs composants à l'aide d'une paire de fils. Il permet également de simplifier la gestion des 3 affichages. Nous avons donc décidé de chercher plusieurs composants pouvant communiquer via ce bus.

Parmi ces composants, il existe les drivers à LEDs MCP23017. Ces drivers permettent de gérer l'affichage binaire directement comme nous le souhaitons par le simple envoi de données. En sortie de ces drivers, on retrouve l'affichage de l'heure en binaire comme expliqué précédemment.

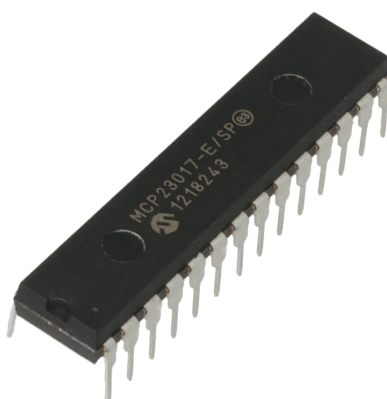


Illustration 7: Un driver à LEDs MCP23017

⁶ Bus : système de communication entre différents composants. Le bus désigne l'intégralité des supports servant à la communication (câble, fibre optique...) ainsi que les logiciels et protocoles associés

⁷ I2C : Inter Integrated Circuit, bus de communication (cf paragraphe 2.1)

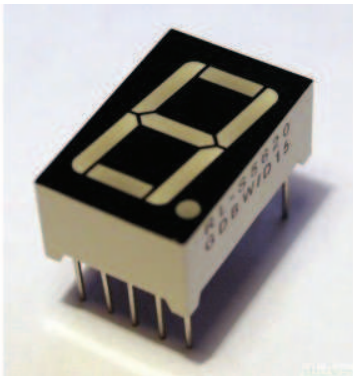


Illustration 8: Afficheur 7-segments

Ensuite, en reprenant chaque groupement de 4bits correspondant aux dizaines ou aux unités des secondes, minutes et heures, on injecte ces mêmes informations sur des décodeurs 7-segments SN74LS47 et on visualise ainsi l'heure directement en format digital.

Cependant, il reste à trouver un composant permettant de gérer la communication en I2C avec l'ensemble des circuits ainsi qu'un composant qui puisse gérer les 60 LEDs de l'affichage traditionnel.

La gestion de la communication I2C doit se faire par un microcontrôleur. On a alors choisi le microcontrôleur ATmega8535 puisqu'il est facile à programmer grâce au logiciel « Code Vision AVR » et qu'on a pu l'étudier durant le module complémentaire du S3 « Microprocesseur ».

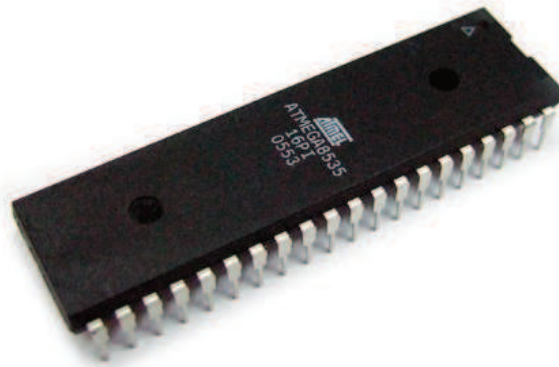


Illustration 9: Un microcontrôleur ATmega8535

Pour finir, nous avons choisi une CPLD pour gérer l'affichage traditionnel. Il s'agit d'un composant programmable, simple à manipuler, qu'on a déjà eu l'occasion de programmer lors du semestre 2. La CPLD possède un très grand nombre d'entrées et de sorties ce qui permet de centraliser la gestion de l'ensemble des LEDs nécessaire à l'affichage traditionnel.

Nous allons alors voir comment l'horloge va fonctionner d'un point de vue général puis ensuite plus en détails à travers les synoptiques de niveau 1 et 2.



Illustration 10: Une CPLD EMP7128SLC84

1.2.2. Synoptique de niveau 1

Le synoptique de niveau 1 ci-dessous permet de schématiser le principe de fonctionnement de l'horloge de manière globale.

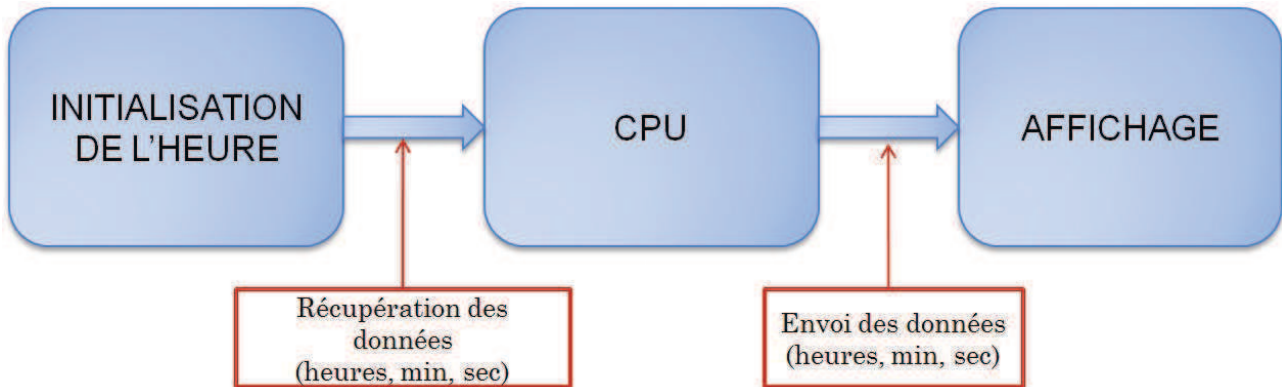


Illustration 11: Synoptique de niveau 1 du projet, fait par les rédacteurs

Le principe de fonctionnement général de l'horloge est assez simple. On doit tout d'abord initialiser l'heure de l'horloge. L'heure réglée sera récupérée par la CPU⁸. Celle-ci va traiter ces données et les envoyer aux différents modules d'affichage sur lesquels on pourra visualiser l'heure.

1.2.3. Synoptique de niveau 2

Le rôle de la CPU est tenu par l'ATmega8535 qui constitue le cœur du projet. La phase d'initialisation de l'heure est faite grâce à deux boutons poussoirs qui vont incrémenter les heures ou les minutes.

L'heure réglée, à l'aide de boutons poussoirs, sera alors récupérée par le microcontrôleur. Celui-ci traitera les informations et initialisera le RTC en envoyant l'heure réglée via la liaison I2C. À partir de ce moment, le RTC ne cessera jamais de fonctionner et assurera le bon déroulement de l'heure.

Le microcontrôleur questionnera donc sans cesse le RTC pour savoir quelle heure est-il et enverra les informations aux différents modules d'affichage. D'une part à la CPLD, qui va gérer grâce au programme qu'on lui implantera, la gestion de l'affichage traditionnel, à l'aide de six bits. Un signal d'horloge sera également envoyé à la CPLD pour assurer son bon fonctionnement et pour synchroniser l'envoi de l'ensemble des données.

Dans un second temps, l'heure sera aussi envoyée aux drivers à LED, qui géreront eux-mêmes l'affichage binaire. Comme expliqué précédemment, on récupère l'information en sortie des drivers pour l'envoyer sur des décodeurs qui afficheront sur des afficheurs 7-segments l'heure en format digital.

⁸ CPU : Central Processing Unit, autrement dit l'unité centrale du système le cœur du projet.

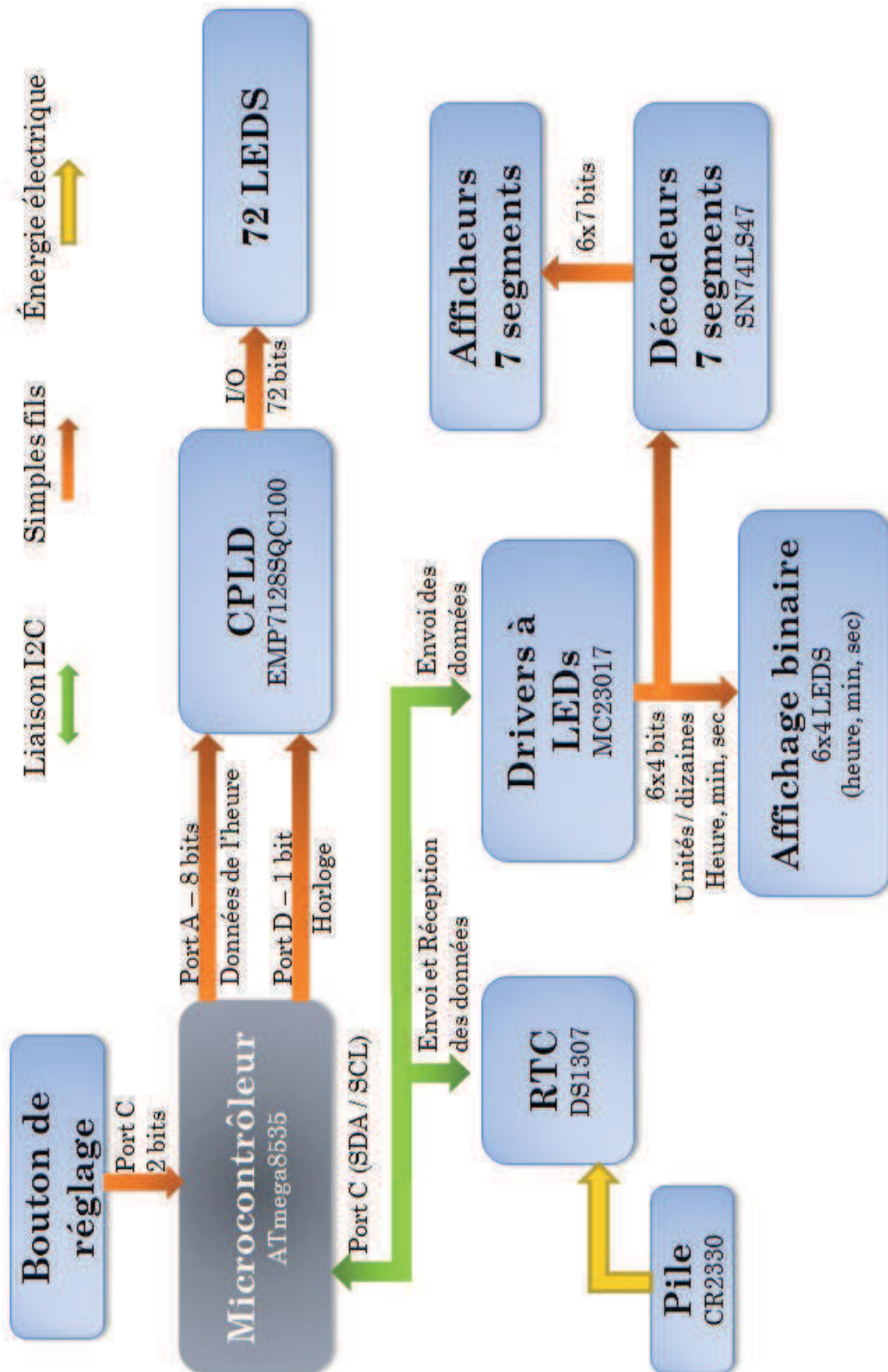


Illustration 12: Synoptique de niveau 2 du projet, fait par les rédacteurs

1.3. Étude du cahier des charges

Malgré la recherche documentaire effectuée l'an passé, le travail sur l'horloge reste cependant conséquent. Il faut à présent réaliser l'horloge et donc effectuer différentes tâches :

- l'élaboration des schémas électriques des sous-fonctions de l'ensemble du montage,
- la réalisation des cartes électroniques correspondant aux sous-fonctions,
- l'écriture du programme de l'ATmega8535 et de la CPLD,
- l'élaboration de la communication du bus I2C,
- le montage final de l'ensemble des cartes dans une horloge.

1.4. Organisation du projet

Le projet est traité par deux étudiants situés dans des groupes TD et TP différents : TUILARD Benjamin (P2) et GRATADE Diana (Q1) avec les professeurs respectifs : Mr LEQUEU et Mr BESSE.

La nécessité de séparer le travail est alors primordiale. Les deux parties du projet devront être indépendantes et permettre aux deux étudiants d'aborder des notions demandant réflexion et recherche.

Le projet a donc été divisé de la manière suivante :

TUILARD Benjamin	GRATADE Diana
<ul style="list-style-type: none">• Réalisation des cartes de l'ATMega8535 et des drivers à LEDs.• La programmation de l'ATMega8535 à la communication I2C avec le RTC et les drivers à LEDs.• L'étude des drivers à LEDs MCP23017. (réglage de fonctionnement et trame de communication)• Le fonctionnement du réglage de l'heure	<ul style="list-style-type: none">• La réalisation de la carte CPLD en CMS et des afficheurs 7 segments.• La programmation de la CPLD pour la gestion des LEDs.• La programmation de l'ATMega8535 pour envoyer les données à la CPLD.• L'étude des trames pour l'échange de données avec le RTC.• Calculs des valeurs des résistances de protection pour chacune des LED.• L'alimentation 0/5V de l'horloge.

Ainsi les tests et les travaux sont indépendants de l'avancée globale du projet et du travail de chacun.

1.5. Planning réel et prévisionnel

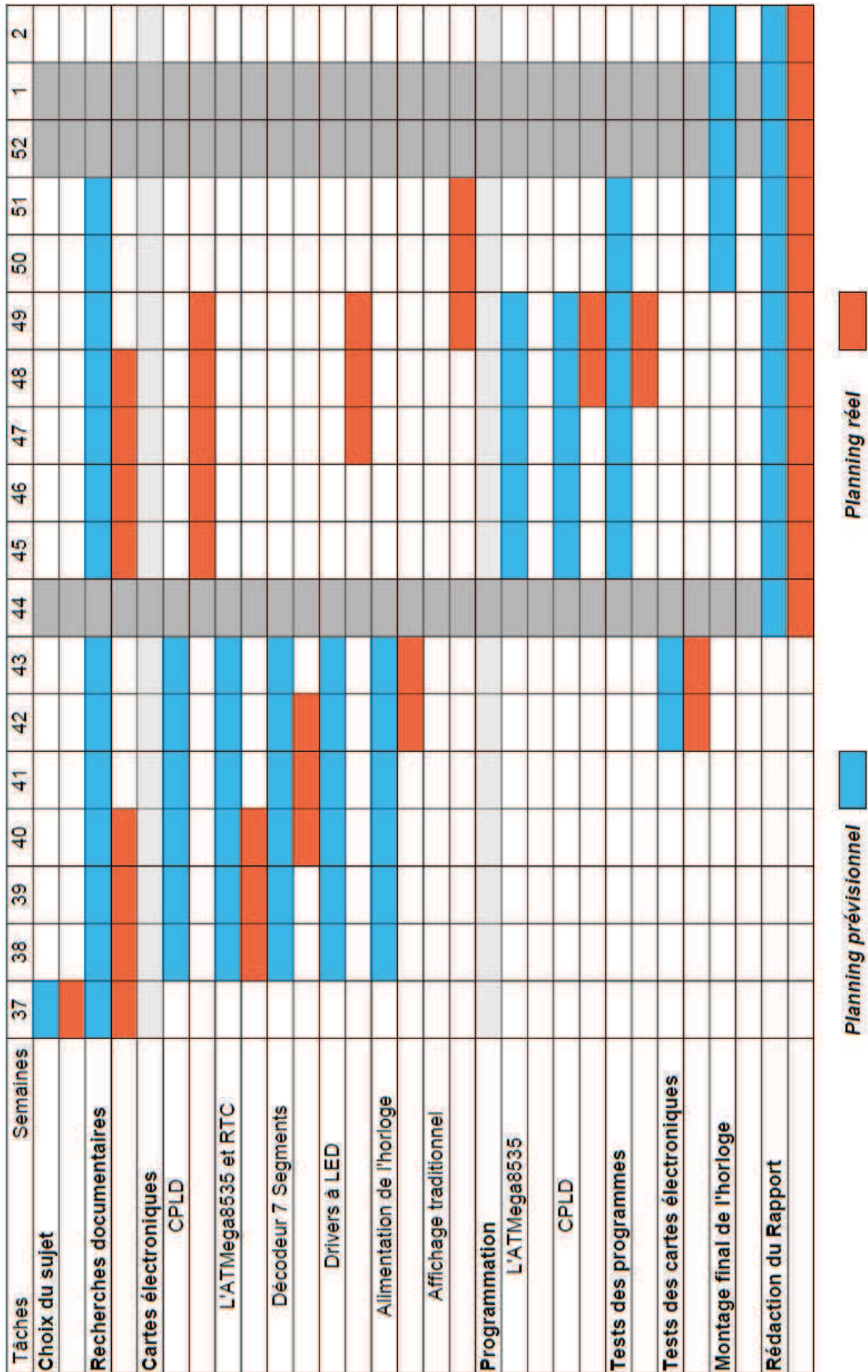


Illustration 13: Planning réel et prévisionnel, fait par les rédacteurs

Comme on peut voir sur le planning, le travail a été décomposé en deux parties, qui sont: la conception des cartes électroniques et la programmation. Ces deux parties demandant un travail conséquent, nous avons donc scindé le planning prévisionnel en deux temps, c'est-à-dire un premier temps pour la conception et un second pour la programmation.

Si l'on compare le planning prévisionnel au planning réel, on peut constater que le temps passé sur la conception des cartes électroniques a été plus long que prévu. Nous avons sous-estimé la complexité de certaines cartes, c'est pourquoi il y a eu débordement sur le temps consacré à la programmation.

De plus, nous avons choisi le logiciel ORCAD pour réaliser nos cartes électroniques. Celui-ci a l'avantage de contenir l'ensemble des éléments nécessaires à la conception de nos cartes. En revanche, nous étant inconnu et complexe d'utilisation, il nous a fallu un temps d'adaptation pour se l'approprier et le maîtriser suffisamment pour pouvoir réaliser les cartes.

Cependant, les logiciels de conceptions des cartes électroniques et de programmations étant payants, nous étions dans l'incapacité de pouvoir travailler chez nous pour pallier le retard accumulé.

2. Description technique de l'horloge

2.1. Le bus I²C

Inventé dans les années 80 par la marque Philips, le bus I2C[4] permet de faciliter la communication entre divers composants électroniques. Aussi connu sous le nom TWI (Two Wire Interface) chez certains constructeurs, ce bus a la particularité d'utiliser 3 fils pour échanger des informations :

- un fil pour le signal de données (SDA⁹)
- un fil pour le signal d'horloge (SCL¹⁰)
- un fil servant de signal de référence électrique (masse)



Illustration 14: Logo du bus I2C

Ainsi, il suffit de lier ces trois fils aux composants souhaitant utiliser la liaison I2C, pour que ceux-ci puissent communiquer entre eux.

La communication I2C est basée sur l'échange entre un maître et les esclaves. Chaque esclave correspond à un composant, qui possède une adresse spécifique sur 8 bits. Le composant maître va quant à lui gérer l'échange des informations en envoyant à la fois les signaux de données et le signal d'horloge, afin de rythmer l'envoi des données. La fonction de maître est remplie par le microcontrôleur et les rôles d'esclaves sont attribués au RTC et aux drivers à LED, dans notre projet.

⁹ SDA : Signal Data

¹⁰ SCL : Signal Clock

Pour débiter la communication en I2C, il faut veiller à ce que le SDA et le SCL soient en état de repos, ce qui correspond si on regarde l'illustration 15 ci-dessous à un état logique 1.

Chaque bit du signal de données (SDA) sera envoyé lors d'un front montant ou descendant du signal d'horloge (SCL).

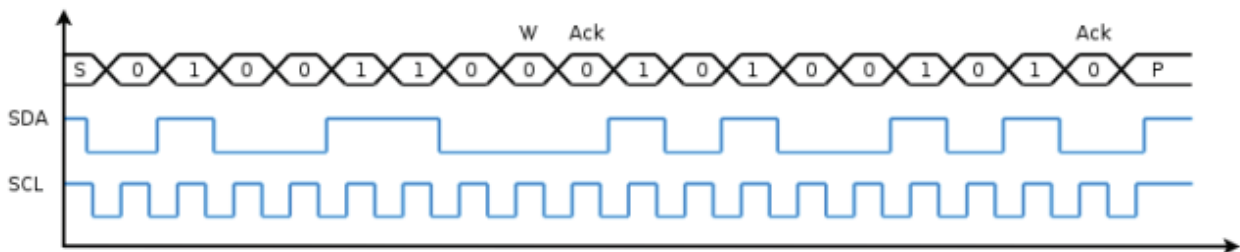


Illustration 15: Exemple de communication I2C

Ensuite, le maître envoie l'adresse de l'esclave avec lequel il veut communiquer, sur le bus I2C. Lors de cette action, on dit que le maître envoie des bits d'adressage. Puis, le maître indique le type d'action qu'il veut effectuer avec l'esclave (R/W), c'est-à-dire s'il souhaite écrire ou lire les données de celui-ci, comme on peut l'observer sur l'illustration 16 ci-dessous.

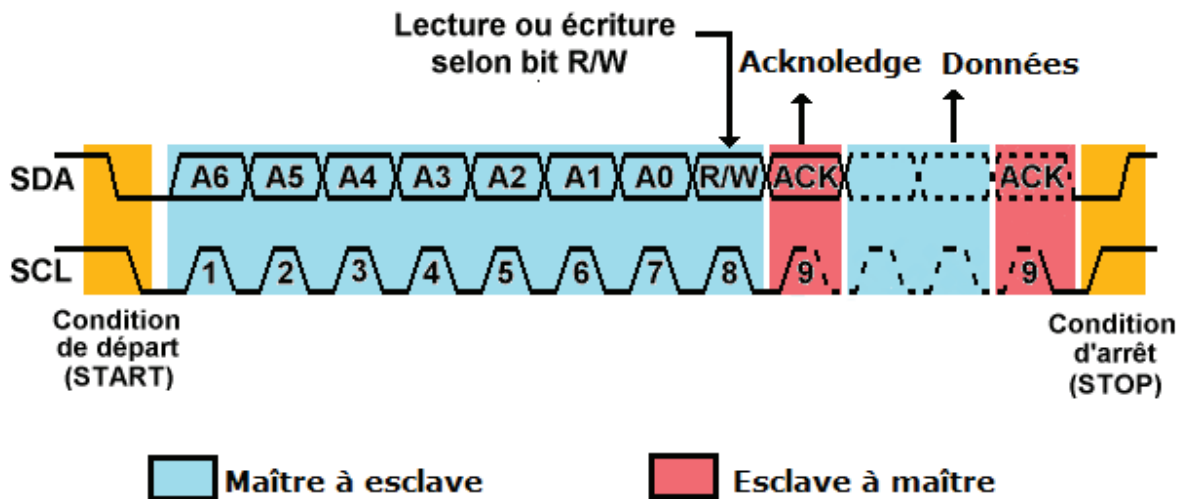


Illustration 16: Principe de fonctionnement d'une trame de communication en bus I2C, entre un maître et un esclave, fait par les rédacteurs

L'esclave va alors répondre au maître en envoyant un « Acknowledge » (ACK) qui correspond à un bit que l'on appelle : bit d'acquiescement. Ce dernier sert à indiquer si l'esclave a correctement reçu les instructions provenant du maître. Le bit d'acquiescement est symbolisé par le forçage de l'esclave à un état logique 0. Une fois l'acquiescement perçu par le maître, celui-ci peut enfin transmettre les informations à l'esclave qui acquiescera à nouveau de sorte à confirmer la réception des données. Le principe reste le même lorsque l'esclave communique des données au maître.

2.2. L'ATmega : le microcontrôleur

L'ATmega8535 est le cœur du projet. C'est un microcontrôleur, c'est-à-dire un circuit intégré rassemblant tous les éléments nécessaires d'un ordinateur comme par exemple un processeur, des mémoires et des interfaces d'entrées et sorties.

Il possède 32 broches d'entrées/sorties réparties par groupement de 8 sur 4 ports : PA, PB, PC, PD (cf. illustration 17). Comme indiqué sur sa datasheet¹¹, il permet de réaliser différentes fonctions telles qu'un comparateur, convertisseur, MLI, et autres. Dans ce projet, on utilisera avant tout le protocole I2C qui se fera au travers des broches 22 et 23 représentant les bits PC0 (SCL) et PC1 (SDA).

Il est programmé grâce au logiciel « Code Vision AVR ». Ce logiciel permet de faciliter la programmation en offrant la possibilité de proposer des lignes de codes déjà établies suivant la fonction que l'on veut mettre en place.

L'intégralité du port A permettra l'échange de données entre l'ATmega8535 et la CPLD. Le port B quant à lui, sert à la programmation du microcontrôleur. On rajoutera sur le port C, trois entrées correspondant aux trois boutons permettant le réglage de l'heure.

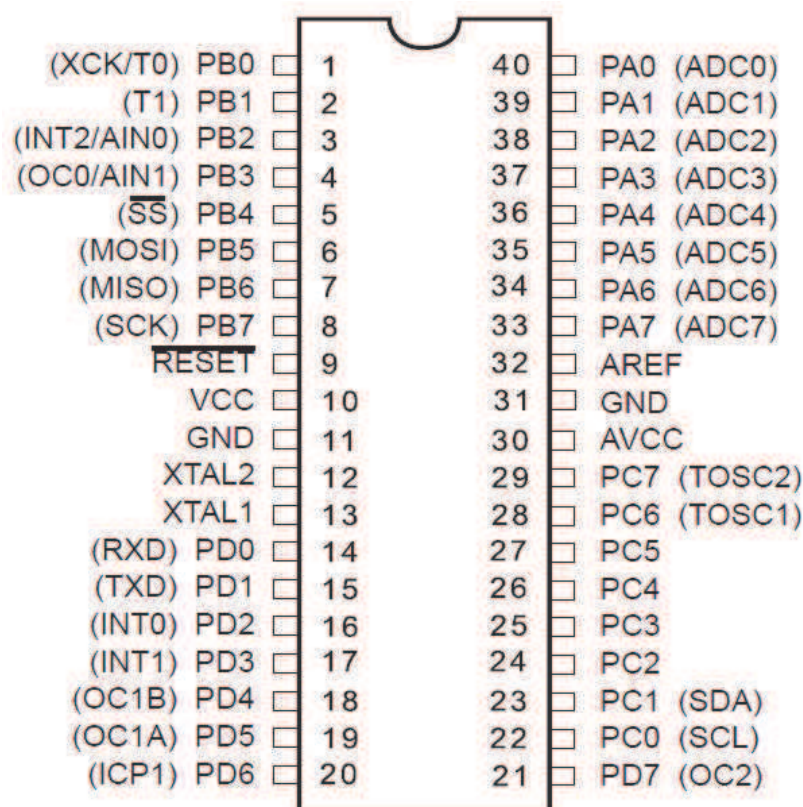


Illustration 17: Schéma de l'ATmega8535 issu de la datasheet du composant[5]

¹¹ Datasheet : notice technique d'un composant, indiquant ces fonctionnalités, des conseils d'utilisations, des explications sur sa structure interne mais aussi les cotations de dimensionnement

2.3. Le RTC : Real Time Clock

Comme expliqué précédemment, le RTC est un composant ayant pour rôle de fournir un signal d'horloge. Le RTC fonctionne grâce à un quartz[6]. C'est un cristal qui a la propriété d'osciller périodiquement si on lui applique une tension électrique. On décompose le fonctionnement du RTC en 2 phases : une phase d'écriture et une autre de lecture.

La phase d'écriture consiste à une initialisation. Le microcontrôleur va envoyer l'heure qui sera préalablement réglée. Pour cela, comme on a vu dans le paragraphe 2.1, le microcontrôleur va saisir l'adresse du RTC, en lui indiquant qu'il souhaite écrire dessus. Une fois la demande acquittée par le RTC, le microcontrôleur lui indiquera l'adresse des données qu'il souhaite modifier, puis les lui enverra. Pour cela, on s'aide du tableau de registre ci-dessous, indiquant l'adresse et la forme des données liées au RTC pour les envoyer correctement.

ADDRESS	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	FUNCTION	RANGE
00h	CH	10 Seconds			Seconds				Seconds	00-59
01h	0	10 Minutes			Minutes				Minutes	00-59
02h	0	12	10 Hour	10 Hour	Hours			Hours	1-12 +AM/PM 00-23	
		24	PM/ AM							
03h	0	0	0	0	DAY			Day	01-07	
04h	0	0	10 Date		Date			Date	01-31	
05h	0	0	0	10 Month	Month			Month	01-12	
06h	10 Year			Year			Year	00-99		
07h	OUT	0	0	SQWE	0	0	RS1	RS0	Control	—
08h-3Fh								RAM 56 x 8	00h-FFh	

Illustration 18: Tableau des registres du RTC issu de sa datasheet[7]

Une fois l'initialisation faite, le microcontrôleur n'aura plus qu'à communiquer au RTC le souhait de lecture des données ainsi que l'adresse où elles sont situées pour que le RTC puisse lui répondre. Ce dernier fonctionnera en continu grâce à sa pile interne et ne fera que décompter l'heure en se synchronisant sur le quartz qui lui est associé.

2.4. Les drivers à LEDs

Le driver à LEDs est un composant qui gère l'affichage binaire. Il permet grâce à 2 ports de 8 broches de sorties de gérer l'éclairage suivant les configurations préalable du composants. La gestion des sorties se fait alors par l'envoi de données via le bus I2C.

L'affichage binaire, comme expliqué précédemment, se repose sur le principe du langage binaire naturel dit DCB. C'est pourquoi les secondes, les minutes et les heures seront chacune gérées par un port du driver. Ce dernier communiquera avec l'ATmega8535 pour savoir quelles sorties il doit commander.

La limitation en sortie du driver nous impose d'en avoir deux. Effectivement, l'heure étant divisée en 3 nombres, chacun associés à un port d'un driver, il nous est nécessaire de correctement configurer les adresses des drivers comme le bus I2C nous l'impose, mais surtout de faire attention à ce qu'il n'y ai pas de conflit lors de l'envoi des données. Malheureusement, on n'a pas pu l'étudier plus en détails dû au retard accumulé durant notre projet.

2.5. La CPLD

La CPLD est un composant électronique programmable, qui a pour rôle dans notre projet de commander l'affichage traditionnel de l'horloge[8].

On a vu précédemment que l'affichage traditionnel se compose de 12 LEDs représentant les heures, mais aussi de 60 LEDs pour les minutes, ce qui nous fait un total de 72 LEDs. La CPLD doit donc comporter un nombre suffisant de broches (ou « pins ») pour contenir l'ensemble des LEDs nécessaire à ce type d'affichage.

Nous disposons pour notre projet de la CPLD référencée : EPM7128SQC100, qui possède au total 100 broches. Parmi ces broches, 76 sont affectées à des entrées/sorties (I/O), idéal pour accueillir nos 72 LEDs.



Illustration 19: CPLD monté CMS (Circuit monté en surface)

Cependant, il faut prendre en compte le fait que la CPLD n'est pas compatible avec le protocole I2C. Certaines broches d'entrées/sorties devront donc être réservées pour permettre l'envoi des données, c'est-à-dire pour permettre à l'ATmega8535 d'indiquer à la CPLD l'heure qu'il est afin que celle-ci puisse se synchroniser et retranscrire cette information sur les 72 LEDs.

Nous aurons donc besoin de sept broches supplémentaires, soit :

- un bit de mode permettant de savoir si les données envoyées correspondent aux heures ou aux minutes puisque l'envoi de ces 2 données ne se fait pas simultanément
- six broches nécessaires à la réception des données en base 2, d'une taille de 6 bits

La présence des 6 bits provient de la volonté de donner l'heure à la CPLD. Or, pour que celle-ci puisse prendre connaissance de l'heure, il faut alors lui envoyer les informations en binaire. Il faut également noter que la valeur maximale à envoyer à la CPLD est 59 (puisque les minutes varient de 0 à 59). Or la valeur maximale du nombre binaire envoyé sur 6 bits est $(111111)_2$. Ce nombre binaire correspond à un nombre décimal maximum égal à :

$$1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 32 + 16 + 8 + 4 + 2 + 1 = 64.$$

Il est donc nécessaire et suffisant que l'envoi des données se fasse sur 6 bits.

Au total, nous avons donc besoin de 79 broches d'entrées/sorties pour satisfaire notre cahier des charges. En revanche, la CPLD ne comporte que 76 broches prévues à cet effet. Il n'y a donc pas suffisamment de broches pour pouvoir mener à bien notre cahier des charges.

Ce composant programmable étant très coûteux, puisque sa marge de prix est comprise entre 40 et 50 euros, nous n'avons donc pas les moyens financiers pour en recommander une. Nous allons donc devoir nous adapter face à cette contrainte et modifier notre cahier des charges.

On choisira donc de n'utiliser que 60 LEDs et non 72. Une première solution consistait à mettre en commun 12 LEDs pour l'affichage des heures et des minutes. Pour différencier ces deux données on aurait mis en œuvre des LEDs RGB qui ont la particularité d'être multicolores et on aurait affecté une couleur pour reconnaître les heures et une autre pour les minutes.

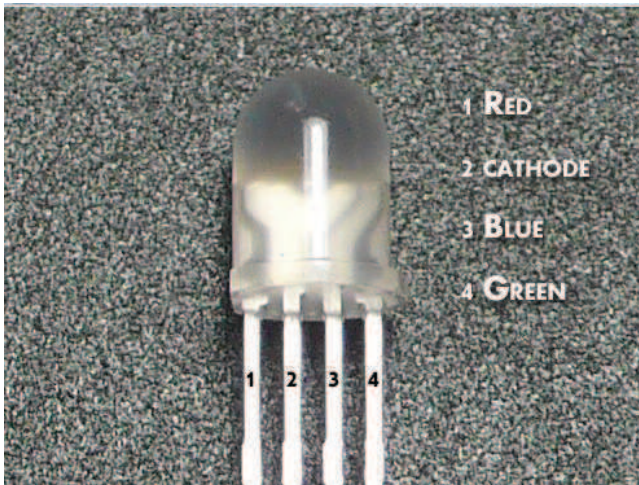


Illustration 20: LED RGB avec affectation des pattes

Cependant, les LEDs RGB nous posent encore un problème puisque chacune d'entre elle doit être reliée à non pas une mais au minimum à deux pins de la CPLD. Une solution qui nous demande encore un total de 79 broches et qui est donc impossible à réaliser.

Pour simplifier la tâche et par soucis de temps vis-à-vis de notre planning, l'affichage traditionnel ne permettra donc de visualiser les minutes et les heures que sur une même couleur et sur un même contour de LEDs. Solution qui se voit contraignante lors de la lecture globale de l'heure, pour un utilisateur extérieur.

3. Mise en œuvre et réalisation de l'horloge

3.1. La partie électronique : réalisation des typons

Étant donné l'importance du projet et le fait qu'on l'ait scindé en deux parties, on a choisi de traiter les différentes fonctions de l'horloge séparément. C'est pourquoi plusieurs cartes électroniques ont été réalisées regroupant ses fonctions et seront par la suite connectées ensemble. L'ensemble des typons¹² et des schémas électriques sera joint dans le rapport en annexe.

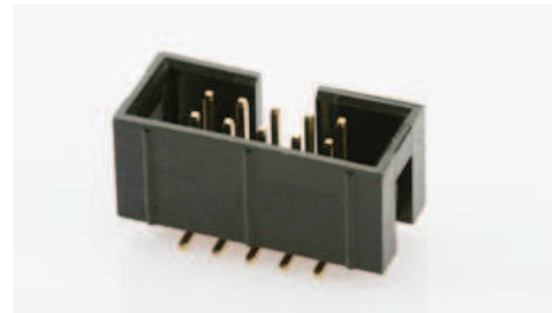


Illustration 21: Connecteur HE10

Pour connecter chaque carte ensemble, on a choisi des connecteurs HE10 (cf. illustration 21). Ces connecteurs ont été choisis pour leur encombrement réduit, la facilité de mise en place et surtout leur disponibilité en magasin.

3.1.1. Carte de l'ATmega

La carte de l'ATmega est la carte mère du projet. Elle réunit le microcontrôleur ATmega8535 ainsi que le RTC DS1307. La particularité du RTC est qu'il est monté en surface, ce qui veut dire qu'il se situera côté cuivre de la plaque, là où l'on soude les composants en temps normale.

La carte accueille aussi 4 connecteurs HE10 qui servent à :

- apporter l'alimentation 5V ainsi que le réglage de l'heure issu de la carte d'alimentation
- établir la liaison I2C avec la carte des drivers à LEDs
- permettre l'envoi des données jusqu'à la carte de la CPLD
- programmer le microcontrôleur

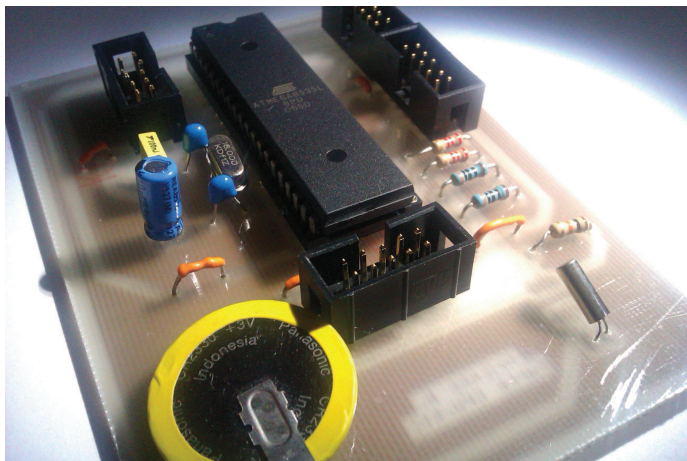


Illustration 22: Carte de l'ATmega du projet réalisée

Le câblage du connecteur de programmation était déjà établi. En ce qui concerne le RTC, on a fait face à plus de difficulté malgré que le schéma de câblage soit donné par sa datasheet. Le quartz doit être au plus près du RTC et avoir un plan de masse, c'est-à-dire un rectangle de cuivre comprenant les 2 pattes du RTC et les 2 pattes du quartz reliées ensemble, et le tout ramené au potentiel électrique de la masse. De plus une pile CR2330 de 3V, imposée par la datasheet, est reliée au RTC pour assurer la continuité de son fonctionnement.

¹² Typon : motif par ordinateur permettant de réaliser une carte électronique

3.1.2. Carte de la CPLD

La carte de la CPLD est celle qui a demandé le plus de travail au cours de notre projet. C'est une carte qui possède la particularité d'accueillir comme composant principal, une CPLD de référence EPM7128SQC100, qui devra être montée en CMS, ce qui implique donc un travail à la fois précis et minutieux. De plus, il faut noter que nous avons à faire ici à un composant programmable doté de 100 broches, ce qui rajoute une difficulté supplémentaire en termes d'encombrement. Il va donc falloir choisir des composants adaptés afin que la CPLD puisse commander l'affichage traditionnel, tout en veillant à ce que le typon ne soit pas surdimensionné par rapport aux autres cartes de l'horloge.



Illustration 23 : CPLD utilisée pour le projet déposée sur la carte électronique

Cette carte se compose de deux connecteurs HE10 qui ont pour rôle :

- La réception des données issues de l'ATmega8535, du signal d'horloge et de l'alimentation de 5V
- La programmation de la CPLD

Le typon consiste à réaliser une carte qui sera directement reliée par superposition à une seconde carte comprenant l'affichage traditionnel de l'horloge, grâce à des connecteurs deux broches.

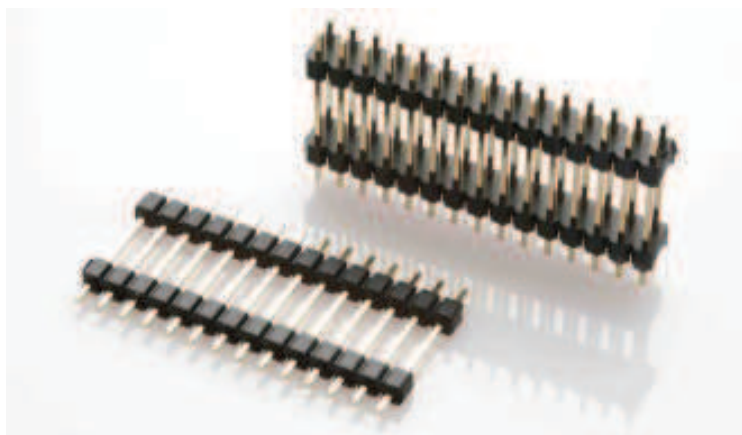


Illustration 24: Connecteurs basés sur le même principe que les connecteurs 2 broches

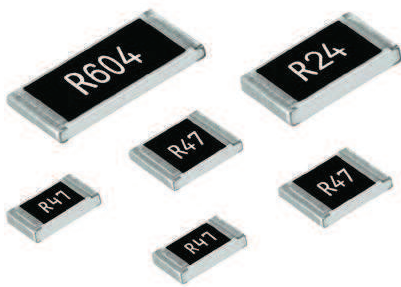


Illustration 25: Résistances de différentes tailles en CMS

De plus, les LEDs ne supportent que du 20 mA et les broches de sorties de la CPLD délivrent une tension de 5V, il faudra donc interposer des résistances entre chaque couple broche de la CPLD/LED, afin de protéger les LEDs d'une détérioration électrique. Pour connaître la valeur des résistances il suffit d'appliquer la loi d'Ohm, on obtient une valeur minimum de 250Ω qui ne correspond pas à une valeur normalisée, c'est pourquoi on choisira des résistances de 270Ω pour notre carte.

Pour réduire la taille de la carte, on décide d'utiliser des résistances en CMS, évitant au passage les courts-circuits entre les pistes très proches les unes des autres, qui proviennent de la CPLD.

Enfin, l'alimentation de l'horloge étant très éloignée de la CPLD nous avons décidé d'inclure des condensateurs de découplage permettant d'éviter une discontinuité d'alimentation.

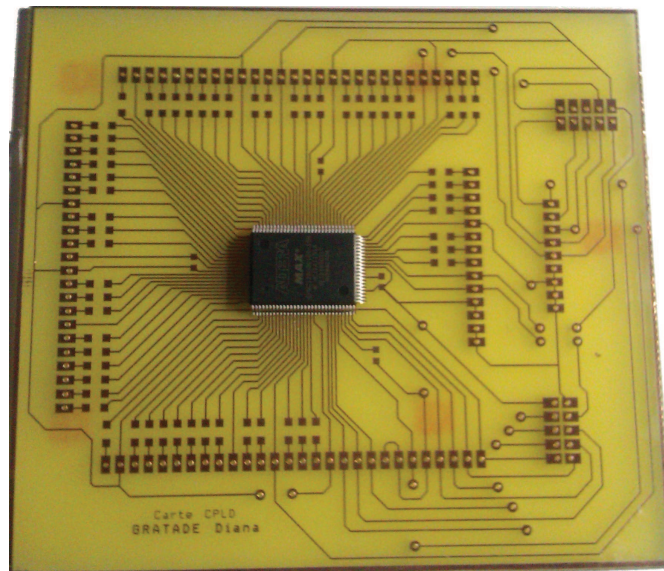


Illustration 26 : Carte de la CPLD réalisée durant le projet

Toutefois même si la carte a été imprimée, nous n'avons pas pu souder les composants n'ayant les compétences nécessaires pour le faire en autonomie.

3.1.3. Carte contenant les drivers à LEDs

La carte des drivers à LEDs doit comprendre l'affichage binaire de l'horloge à savoir les LEDs ainsi que les résistances associées à celles-ci. De plus, on compte que la présence de 4 connecteurs, l'un pour établir la liaison en I2C et l'alimentation de 5V et les autres pour relayer les états des sorties des drivers aux cartes comportant les afficheurs 7-segment.

Malgré que le typon a été débuté, celui-ci n'a pas pu être terminé puisque l'étude complète du driver n'est pas été faite.

3.1.4. Cartes composée d'afficheurs digitaux

Les cartes des afficheurs ont été les plus simples à réaliser. On a tout d'abord réalisé 3 cartes identiques pour dissocier les secondes, les minutes et les heures. Ensuite les cartes comportent simplement 2 afficheurs 7-segments, les 2 décodeurs liés aux afficheurs ainsi que un connecteur HE10. Ce connecteur fait tout simplement le lien avec la carte des afficheurs digitaux et la cartes des drivers à LEDs. Il ramène donc les valeurs de sorties des drivers ainsi que l'alimentation 5V nécessaire au fonctionnement des décodeurs.

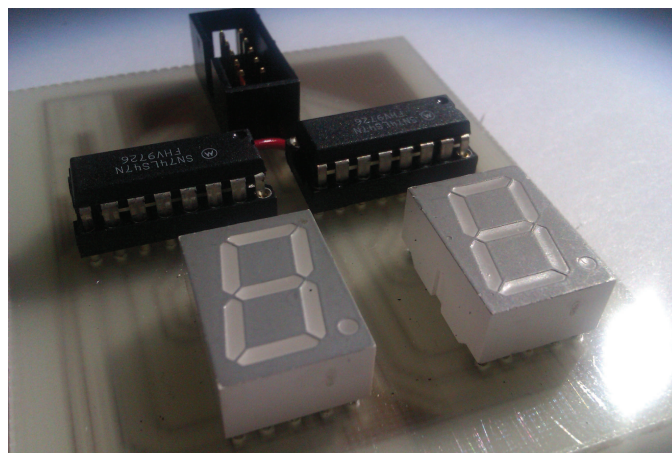


Illustration 27 : Carte des afficheurs 7-segments réalisée

3.1.5. Cartes comportant l'affichage traditionnel à Leds

Au début de projet, il était prévu que les LEDs servant pour l'affichage traditionnel soit placé dans un support circulaire en bois de diamètre de 30cm. Cela impliquait alors un long travail de câblage des LEDs à la carte de la CPLD.

Pour simplifier le câblage, on a décidé de superposer la carte de la CPLD avec une carte qui comprendra toutes les LEDs de l'affichage traditionnel. Cette carte est divisé en 2 plaques mesurant chacune 20x30 cm. Toute la difficulté de cette carte réside dans l'alignement des connecteurs. Une fois l'alignement fait, il ne restait plus qu'à disposer toutes les LEDs en cercle.

3.1.6. Carte d'alimentation 0/+5V

La carte d'alimentation de l'horloge est la plus simple à réaliser. Elle comprend tout d'abord une fiche sur laquelle on branche un câble d'alimentation semblable à un chargeur d'ordinateur portable.

De plus, on y ajoute un interrupteur et 2 boutons poussoirs qui servent à pouvoir régler l'heure. L'interrupteur a pour rôle de permettre le réglage de l'heure à l'aide des boutons poussoir. Cela évite de pouvoir dérégler l'heure lors d'une manipulation de l'horloge par un appuie malencontreux sur un des boutons.

Les boutons poussoirs ainsi que l'interrupteur vont une liaison entre une broche d'entrées du microcontrôleur et la tension d'alimentation 5V. On détectera ainsi si l'interrupteur est activé ou non par une broche d'entrées alimenté en 5V ou non et une incrémentation des heures ou des minutes à chaque fois que le microcontrôleur détectera sur les broches d'entrées associées au passage d'une tension de 0 V à 5V alimentant la broche en question.



Illustration 28 : Carte d'alimentation réalisée

3.2. La partie informatique : programmation

3.2.1. Programmation du microcontrôleur

Le programme de l'ATmega8535 doit se faire grâce au logiciel « Code VisionAVR ». Il a pour rôle de détecter lorsqu'on règle l'heure et l'appuie sur les boutons poussoir. Il doit aussi assurer la communication en protocole I2C avec le RTC et les drivers à LEDs. Pour finir, il doit envoyer les données de l'heure à la CPLD.

Malheureusement dû au retard du projet, la programmation de l'ATmega8535 n'a pas pu se faire.

3.2.2. Programmation de la CPLD

La programmation de la CPLD a été faite grâce au logiciel Quartus. Celui-ci se fait en langage Verilog. C'est un langage de description du matériel de circuits logique fortement inspiré du langage C .

Pour programmer la CPLD il faut dans un premier temps lui fournir les données relatives à l'heure en provenance de l'ATmega8535. Pour ce faire, nous ne pourrons pas communiquer ces données à l'aide du protocole I2C, elles devront donc être envoyées manuellement.

Parmi les données à envoyer, figure la valeur des heures et celle des minutes. On a expliqué précédemment qu'il était nécessaire utiliser 6 bits pour transmettre des valeurs variant de 0 à 59. Nous avons donc choisi de consacrer six ports de la CPLD à la réception des données émises par le microcontrôleur. On retrouvera dans le programme une variable nommée DATA qui correspondra à ce même mot de 6 bits, qui se déclarer de la manière suivante : « `input [5:0] data;` ».

Le principe de transmission des informations est simple. On commence par synchroniser les données à l'aide d'un signal d'horloge que l'on assimilera à la variable CLOCK dans le programme, qui sera déclaré sous la forme : « `input clock;` ». Ce signal envoyé par l'ATmega8535, servira de référence pour rythmer l'envoi des données vers la CPLD. Ainsi, à chaque détection d'un front montant du signal d'horloge (« `posedge clock` »), c'est-à-dire lorsque le signal passe d'un état logique bas à un état logique haut, on actualisera l'affichage traditionnel de l'horloge.

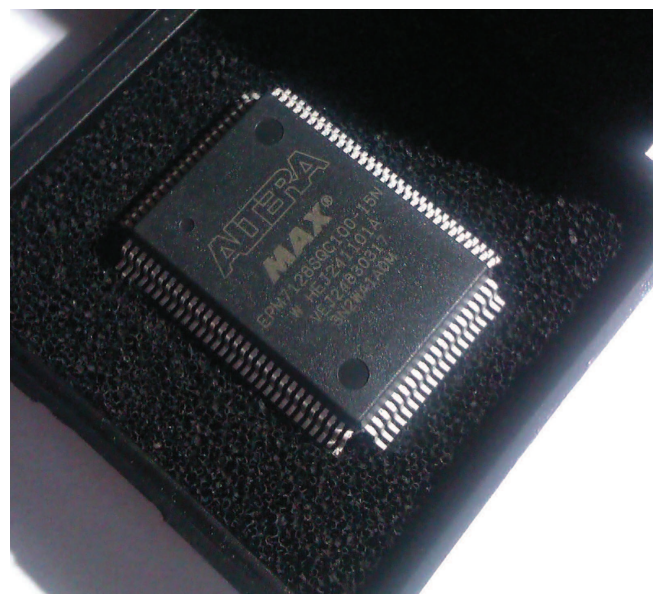


Illustration 29: CPLD utilisée pour le projet, photographie faite par les rédacteurs

Ensuite, le microcontrôleur fournit au circuit logique programmable ce qu'on appelle un bit de mode, qui permettra au composant récepteur d'identifier la nature de l'information reçue (heure ou minute). Ce bit correspondra à la variable MODE, que l'on retrouvera déclarer de cette façon dans le programme : « `input mode;` ». Par conséquent, si MODE est à 0, l'information reçu à travers DATA sera les minutes, et dans le cas où MODE est à 1, la DATA transmise correspondra aux heures.

Il faut maintenant s'adapter au cahier des charges. En effet, la CPLD a pour rôle de piloter 60 LEDs pour les minutes et 12 LEDs pour les heures. Il est donc nécessaire de déclarer deux variables dans le programme, une de 60 bits pour les minutes et l'autre d'une taille de 12 bits pour les heures, tel que : « `output [59:0] minute;` `output[11:0]heure;` ».

Le nombre binaire d'une taille de 60 bits représente en réalité les 60 LEDs de l'affichage traditionnel. Le but est d'allumer une seule LED à la fois pour les minutes ou pour les heures. Or, on a vu précédemment qu'un bit d'un nombre binaire est caractérisé par son rang **R** et qu'on lui associe un nombre décimal égal à 2^R , autrement dit on élève le chiffre 2 à la puissance du rang auquel le bit se situe. Donc dans le programme, si on souhaite allumer la LED représentant la 42ème minute, il faut que le nombre binaire soit égal en décimal à 2^{42} , c'est-à-dire que le bit situé au rang 42 soit à l'état logique 1 alors que tous les autres bits restent à 0.

Il suffit d'effectuer cette opération deux fois pour afficher distinctement les heures et les minutes. On pourra ainsi allumer uniquement les LEDs que l'on souhaite pour l'affichage traditionnel. On retrouve l'essentiel du programme commenté en annexe 11 ainsi que la simulation du programme en annexe 12.

On peut voir sur la simulation la présence de plusieurs paramètres. Tout d'abord l'horloge nommée « clock » servant à rythmer l'envoi des données. En lien avec l'horloge nous avons le paramètre « data » qui simule une incrémentation des heures. C'est la donnée qui est reçue via le microcontrôleur. On voit alors en sortie, sur la variable « heure », que seul un bit est mis à 1.

De plus lorsque la data est incrémentée, le bit mis à 1 se décale lorsqu'il y a un front montant sur l'horloge. On répond bien à notre cahier des charges et au fonctionnement attendu.

Conclusion

Durant ce projet, nous avons réalisé une horloge à LEDs fonctionnant grâce à un composant appelé RTC. Le but de cette horloge est de pouvoir afficher l'heure de 3 manières différentes : digitale, binaire et traditionnelle. Pour réaliser les 3 types d'affichages, nous nous sommes basés sur la recherche de composants qui avait été établie lors du semestre précédent et qui correspondait parfaitement à notre demande. Une fois ces recherches retravaillées, on pouvait alors dégager une planification prévisionnelle de notre travail.

Cependant, la nécessité de scinder notre projet en plusieurs sous-parties était de mise face à l'ampleur du cahier des charges. C'est pourquoi, il fallait trouver un moyen de communiquer entre chaque sous-parties de notre projet, afin que l'heure soit homogène dans chaque afficheurs, autrement dit que les informations relative à l'heure circulent partout et à intervalles régulier. C'est ainsi que nous avons été amené à étudier le Bus IC2 et à les caractéristiques des composants principaux de notre horloge, à savoir : le microcontrôleur, le RTC, les drivers à LEDs et la CPDL.

Enfin, il ne nous restait plus qu'à concevoir l'horloge. Une partie électronique comprenait la réalisation de 6 typons indépendants des uns des autres et dans une partie informatique on retrouvait la programmation du microcontrôleur et du circuit logique programmable.

Cette phase de conception imposait de la rigueur dans notre projet. Le fait d'avoir divisé notre projet en sous-parties nous demandait une certaine organisation afin de ne pas perdre le fil conducteur lors de la liaison entre nos différentes cartes électroniques, mais également une communication entre les deux membres du binôme pour veiller à ce que notre travail respectif reste cohérent .

Malgré notre organisation, la phase de conception électronique c'est avérée être plus longue que celle prévue lors du planning prévisionnel. Par conséquent, il nous a fallut réévaluer notre temps de travail et faire des compromis qui nous ont empêché d'aboutir jusqu'au montage final de l'horloge.

Néanmoins, ce projet nous a permis de tester notre faculté d'adaptation et de réaction face à diverse situations qui nous étaient inconnues. On pense notamment à la carte de la CPLD qui faisait l'objet d'un montage en CMS, la réalisation électronique exigeait un travail consciencieux, cela demandait donc une certaine maîtrise du logiciel ORCAD.

A l'avenir, ce projet pourra être repris au semestre 4. En effet, le travail restant consiste à achever la partie électronique qui comprend, la finalisation des cartes : des drivers, de la CPLD et de l'affichage traditionnel. Mais également, la programmation du microcontrôleur et enfin la réalisation du montage final de l'horloge.

Résumé

Dans le cadre de notre projet tutoré, nous avons décidé de poursuivre un projet qui fut débuté auparavant : la fabrication d'un horloge à LED.

Dans un premier temps, nous avons expliqué en quoi consister l'horloge. Cette dernière doit afficher l'heure en trois formats différents. Le premier affichage est dit traditionnel. Il est prévu que 60LEDs fassent le tour de l'horloge et que l'heure soit indiquée par des LEDs allumées. Ensuite nous avons un affichage binaire qui repose sur le langage binaire. Pour finir un dernier affichage digital sera mis en place et est semblable à un radio réveil classique.

Étant donné que la recherche documentaire et le choix des composants a été fait au début de ce projet, nous avons justifié le choix des composants. Pour cela nous avons exposé le rôle de chacun d'entre eux ainsi que leur technologie, l'importance qu'ils ont dans le fonctionnement de l'horloge et leur rapport avec les autres composants.

Pour finir nous avons décrit l'ensemble de nos réalisations, avec les problèmes rencontrés lors de la conception des cartes électroniques et les solutions émises pour essayer d'aboutir le projet. Cependant, le travail sur la conception des cartes électroniques étant trop important, nous avons pris du retard sur le projet, nous obligeant à ne pouvoir n'aboutir complètement. Certaines cartes ne sont pas terminées et il reste toute la programmation du microcontrôleur.

Nombre de mot : 226 mots

Index des illustrations

Illustration 1: Croquis de l'horloge à réaliser, fait par les rédacteurs.....	7
Illustration 2: Exemple d'affichage traditionnel à base de LEDs[1].....	8
Illustration 3: Horloge utilisant un affichage digital.....	8
Illustration 4: Décomposition d'un nombre binaire de 7bits, faite par les rédacteurs.....	9
Illustration 5: Horloge binaire vendue dans le commerce[2].....	9
Illustration 6: un RTC DS1307 utilisé dans le projet[3].....	10
Illustration 7: Un driver à LEDs MCP23017.....	10
Illustration 8: Afficheur 7-segments.....	11
Illustration 9: Un microcontrôleur ATmega8535.....	11
Illustration 10: Une CPLD EMP7128SLC84.....	11
Illustration 11: Synoptique de niveau 1 du projet, fait par les rédacteurs.....	12
Illustration 12: Synoptique de niveau 2 du projet, fait par les rédacteurs.....	13
Illustration 13: Planning réel et prévisionnel, fait par les rédacteurs.....	15
Illustration 14: Logo du bus I2C.....	16
Illustration 15: Exemple de communication I2C.....	17
Illustration 16: Principe de fonctionnement d'une trame de communication en bus I2C, entre un maître et un esclave, fait par les rédacteurs.....	17
Illustration 17: Schéma de l'ATmega8535 issu de la datasheet du composant[5].....	18
Illustration 18: Tableau des registres du RTC issu de sa datasheet[7].....	19
Illustration 19: CPLD monté CMS (Circuit monté en surface).....	20
Illustration 20: LED RGB avec affectation des pattes.....	21
Illustration 21: Connecteur HE10.....	22
Illustration 22: Carte de l'ATmega du projet réalisée.....	22
Illustration 23 : CPLD utilisée pour le projet déposée sur la carte électronique	23
Illustration 24: Connecteurs basés sur le même principe que les connecteurs 2 broches.....	23
Illustration 25: Résistances de différentes tailles en CMS.....	24
Illustration 26 : Carte de la CPLD réalisée durant le projet.....	24
Illustration 27 : Carte des afficheurs 7-segments réalisée.....	25
Illustration 28 : Carte d'alimentation réalisée.....	25
Illustration 29: CPLD utilisée pour le projet, photographie faite par les rédacteurs.....	26

Index des mots clefs du projet

- LED
- Microcontrôleur
- RTC
- Quartz
- Bus I2C
- Drivers à LED
- Affichage binaire
- Décodeur 7-segments
- CPLD
- Registre à décalage

Glossaire

- **Bit** (page 9) : Chiffre binaire valant 0 ou 1. C'est une unité de mesure informatique qui décrit l'état d'une variable pour savoir si elle est activée ou non
- **Bus** (page 10) : Système de communication entre différents composants. Le bus désigne l'intégralité des supports servant à la communication (câble, fibre optique..) ainsi que les logiciels et protocoles associées
- **CPLD** (page 8) : Complex Programmable Logic Device, en français Circuit Logique Programmable
- **CPU** (page 12) : Central Processing Unit, autrement dit l'unité central du système
- **Datasheet** (page 18): notice technique d'un composant, indiquant ces fonctionnalités, des conseils d'utilisations, des explications sur sa structure interne mais aussi les cotations de dimensionnement
- **I2C** (page 10) : Inter Integrated Circuit, bus de communication
- **LED** (page 6) : Light-Emitting Diode ou diode électroluminescente
- **RTC** (page 7) : Real Clock Time ou Horloge à Temps Réel
- **SCL** (page 16) : Signal Clock
- **SDA** (page 16) : Signal Data
- **Signal d'horloge** (page 7) : signal électrique de type créneau variant périodiquement entre 2 états appelés état haut et état bas correspondant à l'activation ou non d'un indicateur comme une ampoule
- **Typon** (page 22) : Motif par ordinateur permettant de réaliser une carte électronique

Bibliographie

- [1] **Braw Knaapen**. *Equinox Clock*, 2012, [En ligne]. (Page consultée le 16/10/12)
<<http://www.bramknaapen.com/?p=549>>
- [2] **Corolab**. *Horloge BCD*, 2009, [En ligne]. (Page consultée le 16/10/12)
<<http://corolab.unblog.fr/2009/05/04/horloge-bcd>>
- [3] **Samuel Goutenoir**. *DS1307*, 2010, [En ligne]. (Page consultée le 16/10/2012)
<<http://samuel.goutenoir.com/archives/134>>
- [4] **Aurelien Jarno**. *le bus I2C*, N.C., [En ligne]. (Page consultée le 16/10/2012)
<<http://www.aurel32.net/elec/i2c.php>>
- [5]. *Datasheet ATmega8535*, , [En ligne]. (Page consultée le 16/10/2012)
<http://datasheetcatalog.com/datasheet_pdf/A/T/M/E/ATMEGA8535.html>
- [6] **N.C.**. *Le quartz en oscillateur*, N.C., [En ligne]. (Page consultée le 12/11/2012)
<<http://etronics.free.fr/dossiers/analog/analog60.htm>>
- [7] **Maxim**. *Datasheet DS1307*, , [En ligne]. (Page consultée le 16/10/2012)
<<http://datasheets.maximintegrated.com/en/ds/DS1307.pdf>>
- [8] **Gabrielle Bonnet**. *Les horloges*, 2004, [En ligne]. (Page consultée le 12/10/12)
<http://culturesciencesphysique.ens-lyon.fr/XML/db/csphysique/metadata/LOM_CSP_Horloges.xml>

Annexes

Table des Annexes

Annexe 1 - Tableau de conversion du langage DCB.....	36
Annexe 2 – Nomenclature des composants.....	37
Annexe 3 - La carte de la CPLD.....	38
Annexe 4 - La carte de l'ATmega.....	40
Annexe 5 - La carte de l'alimentation.....	42
Annexe 6 - La carte des drivers à LEDs.....	44
Annexe 7 - La carte des afficheurs 7-segments.....	46
Annexe 8 - La carte de l'affichage traditionnel	48
Annexe 9 – Programme de la CPLD.....	49
Annexe 10 – Simulation du programme de la CPLD.....	50
Annexe 11 – Datasheet de la CPLD.....	51
Annexe 12 – Datasheet de l'ATmega.....	52
Annexe 13 – Datasheet du driver à LED.....	53
Annexe 14 – Datasheet du décodeur 7-segments.....	54
Annexe 15 – Datasheet du RTC.....	55

Annexe 1 - Tableau de conversion du langage DCB

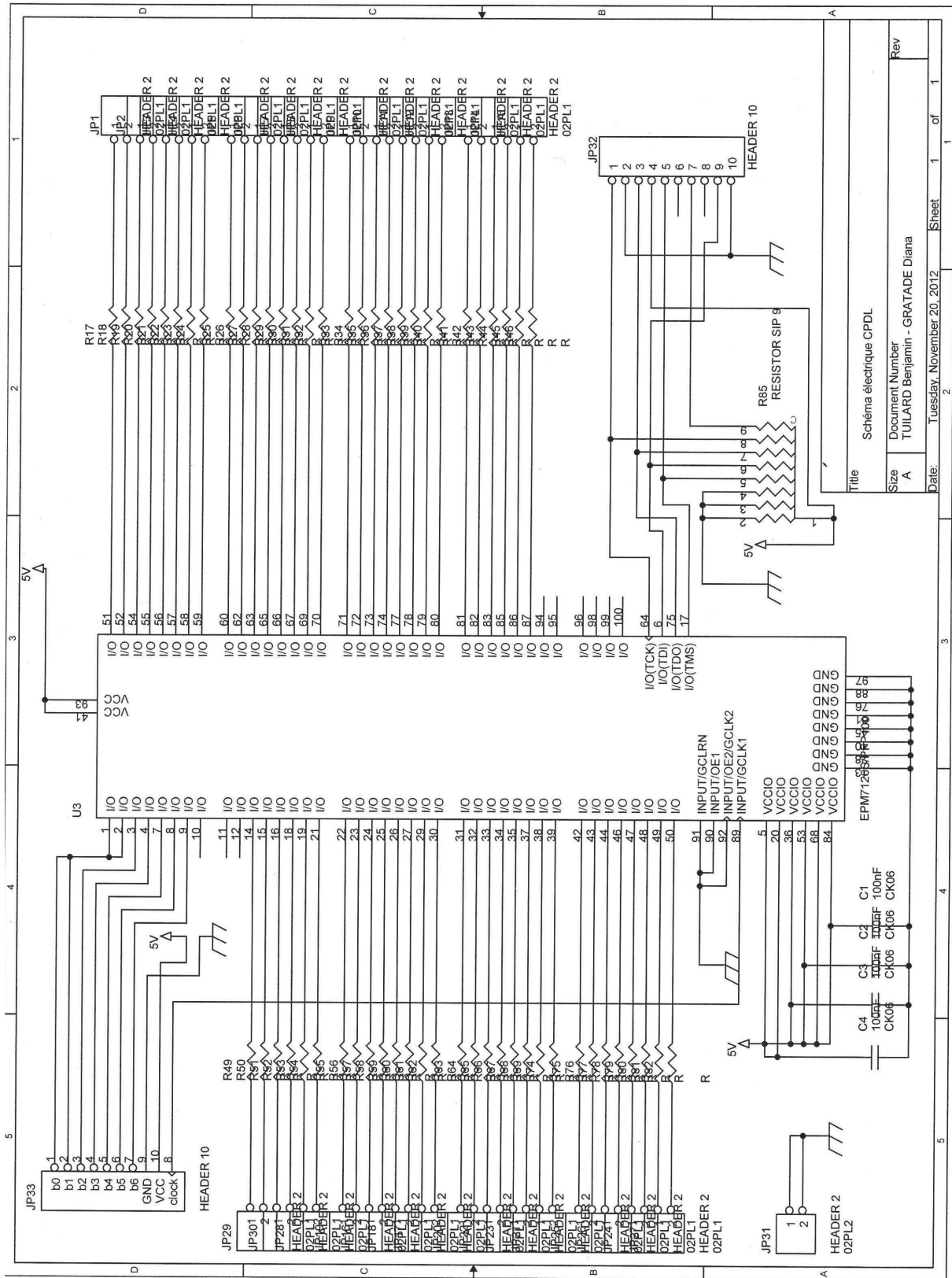
Nombre en binaire				Chiffre en décimal
Bit 3	Bit 2	Bit 1	Bit 0	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9

Annexe 2 – Nomenclature des composants

Nom du composant	Désignation	Quantité	Technologie
ATmega	ATmega8535	1	Boîtier PDIP
CPLD	EPM7128SQC100	1	Boîtier PQFP
RTC	DS1307	1	CMS
Driver à LED	MCP23017	2	
Décodeur 7-segments	SN74LS47	6	
Connecteur 10 broches	Header 10	11	
Connecteur 2 broches	Header 2	46	
Afficheur 7-segment	HDSP-5501	6	
Bouton poussoir		2	
Interrupteur	2 positions	1	6A – 125 VAC
Fiche d'alimentation		1	
Pile	CR2330 - 3V	1	265A/h
Quartz RTC	32,768 kHz	1	
Quartz Atmega	16 MHz	1	
LED	5 mm – 630 nm	85	
Résistance		60	CMS
Résistance			¼ W - 5%
Réseau de résistances		1	
Alimentation	5V - 4A		

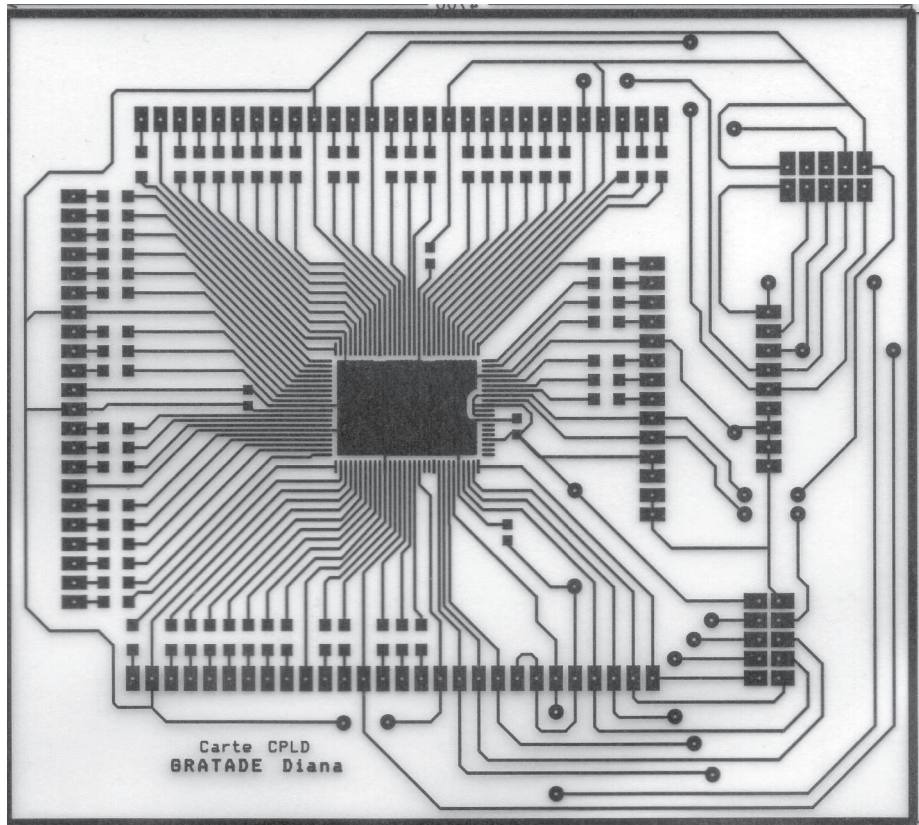
Annexe 3 - La carte de la CPLD

Le schéma électrique

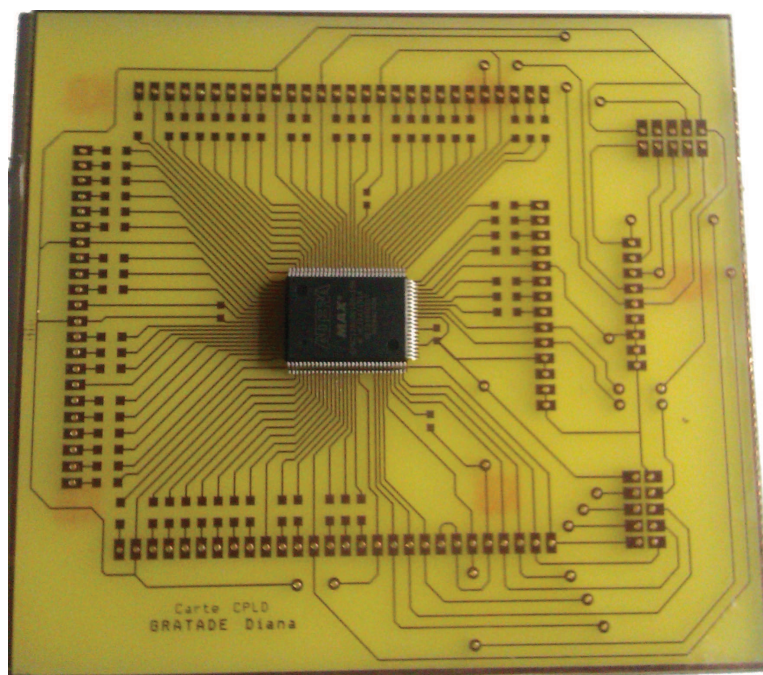


Title	Schéma électrique CPLD	
Document Number	TULLIARD Benjamin - GRATADE Diana	
Size	A	
Date:	Tuesday, November 20, 2012	Sheet 1 of 1
Rev	1	

Le typon

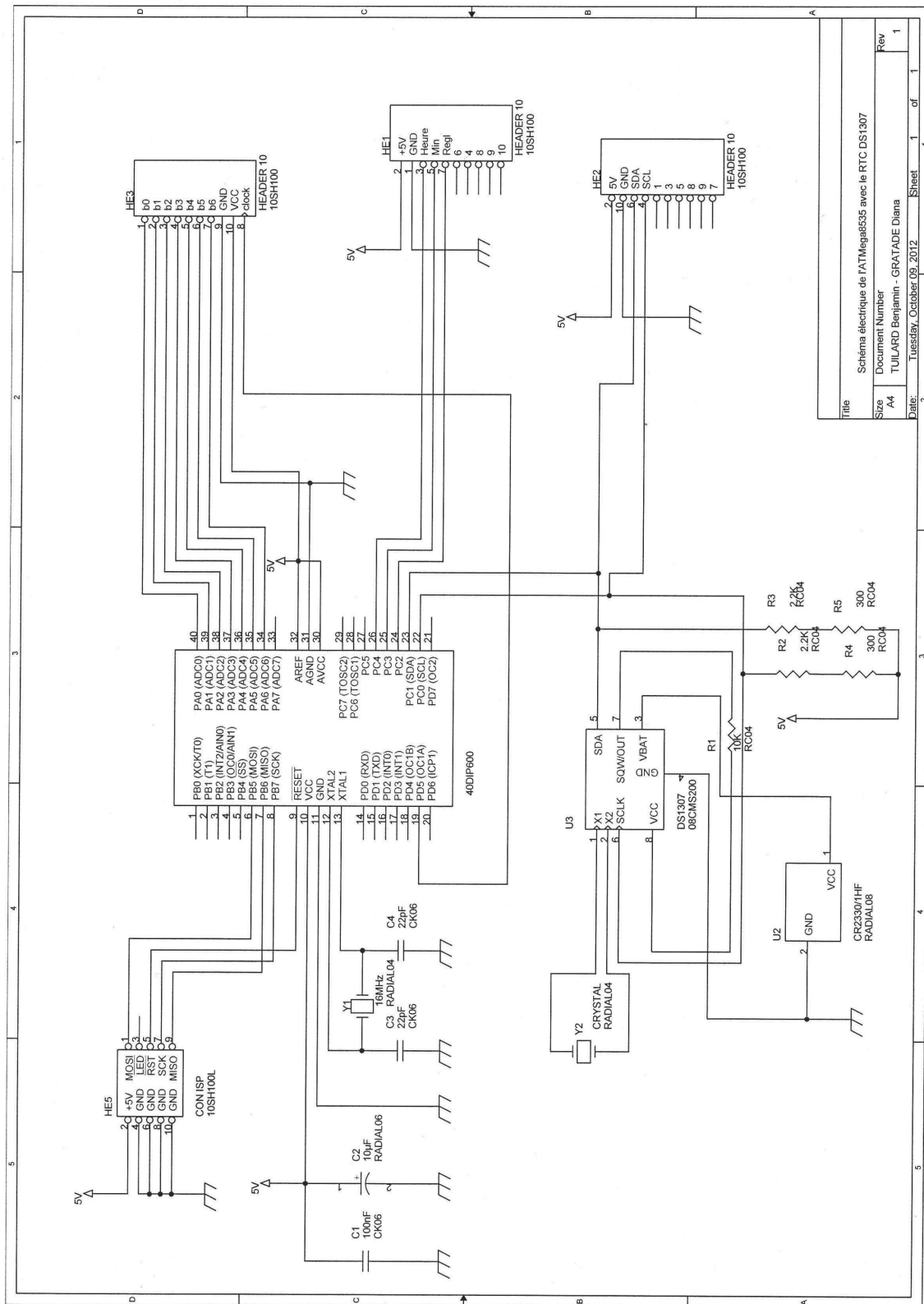


Carte finale



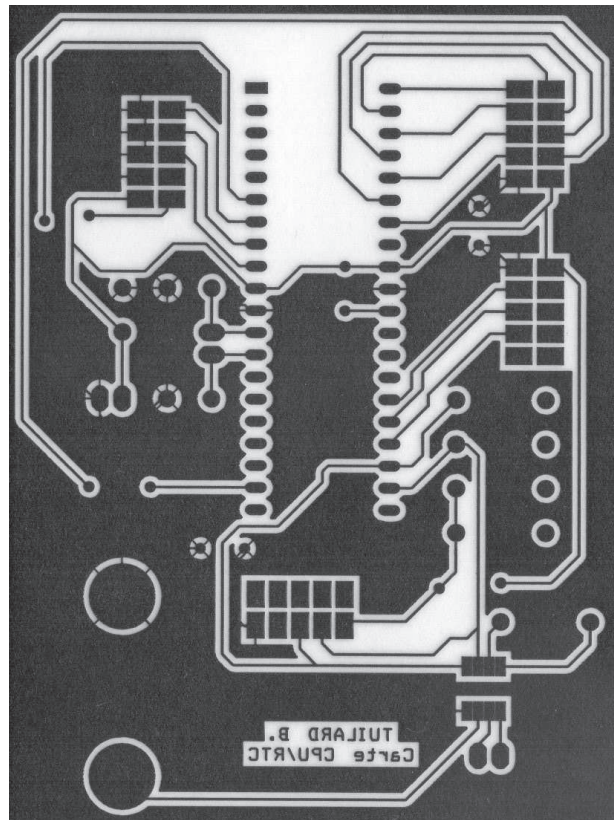
Annexe 4 - La carte de l'ATmega

Le schéma électrique

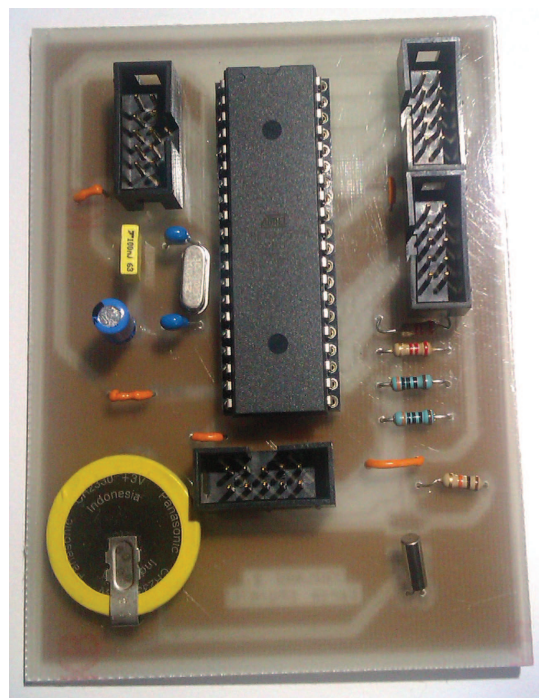


Titre	Schéma électrique de l'ATMega655 avec le RTC DS1307		
Document Number	TUILIARD Benjamin - GRATADE Diana		
Size	A4	1	of 1
Rev	1	1	of 1
Date:	Tuesday, October 09, 2012	Sheet	1

Le typon

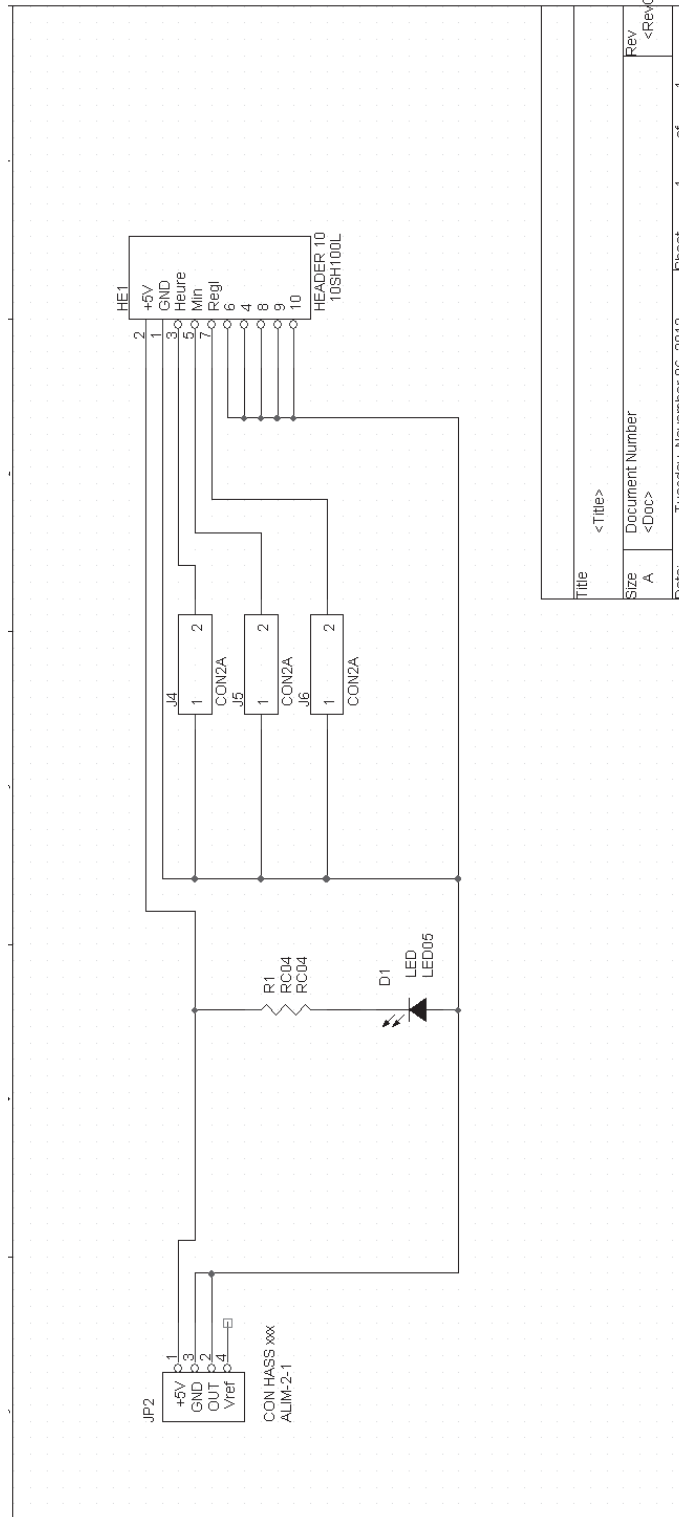


Carte finale



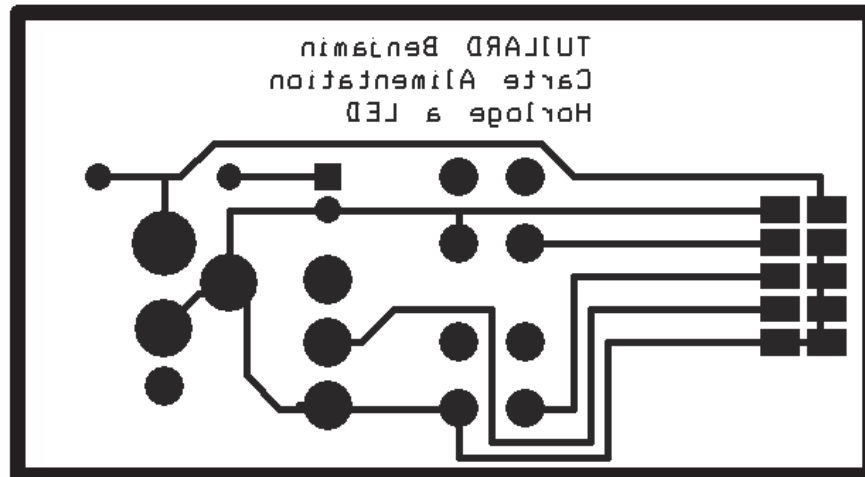
Annexe 5 - La carte de l'alimentation

Le schéma électrique



Title	<Title>
Size	Document Number
A	<Doc>
Date:	Tuesday, November 06, 2012
Sheet	1 of 1
Rev	<Rev>

Le typon

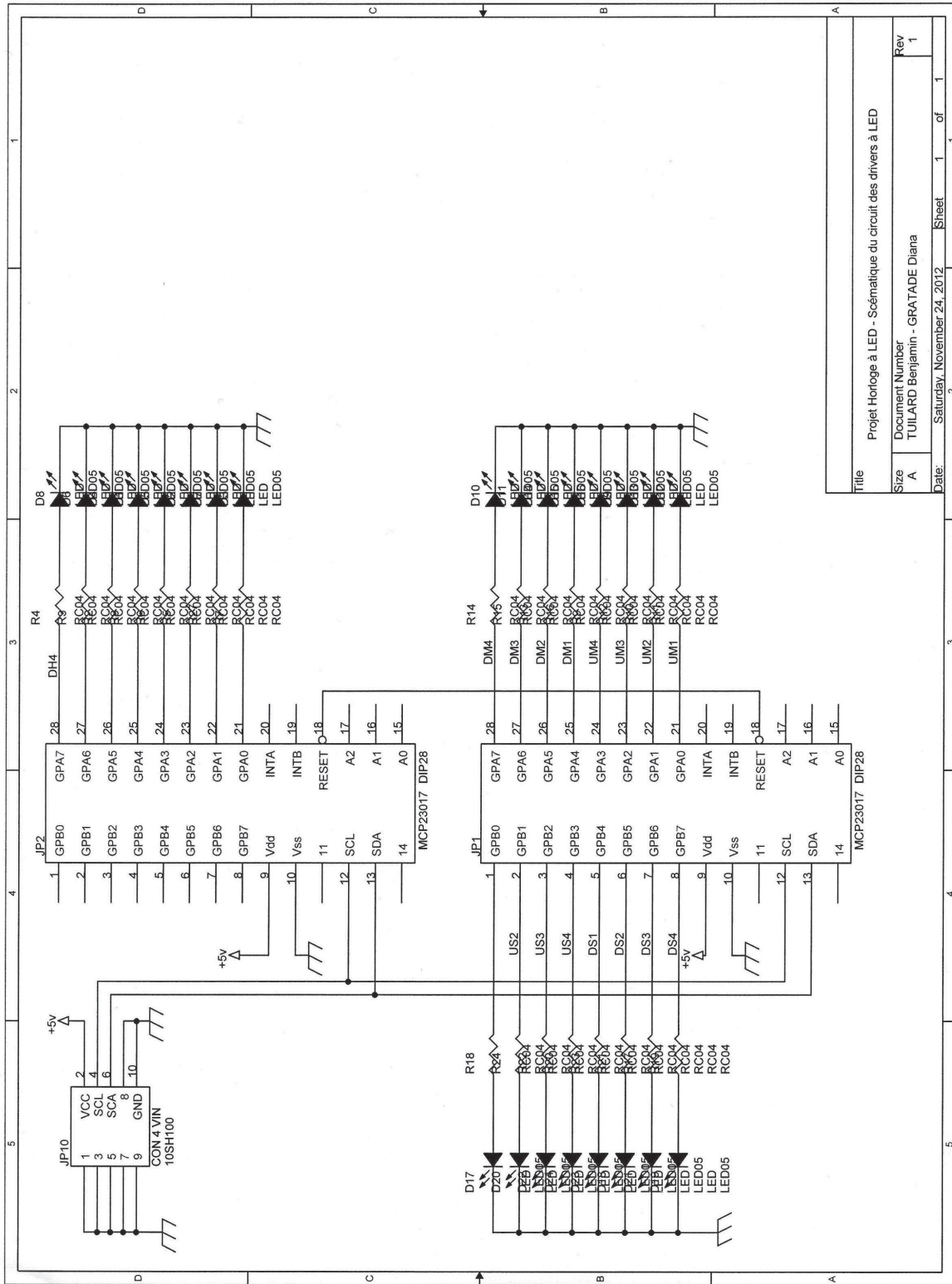


Carte finale



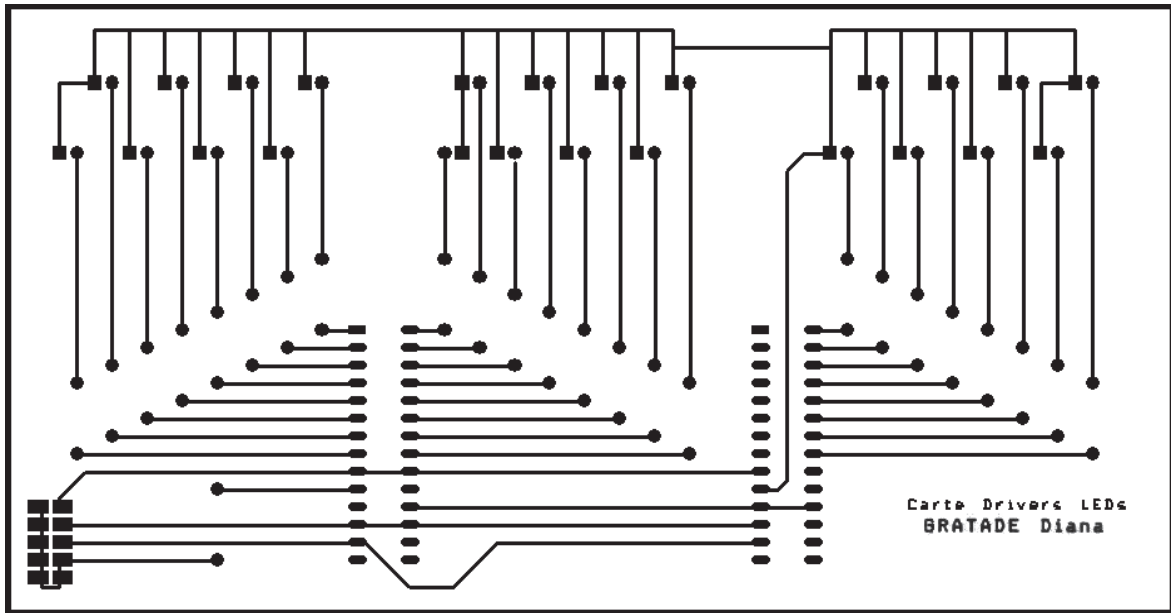
Annexe 6 - La carte des drivers à LEDs

Le schéma électrique



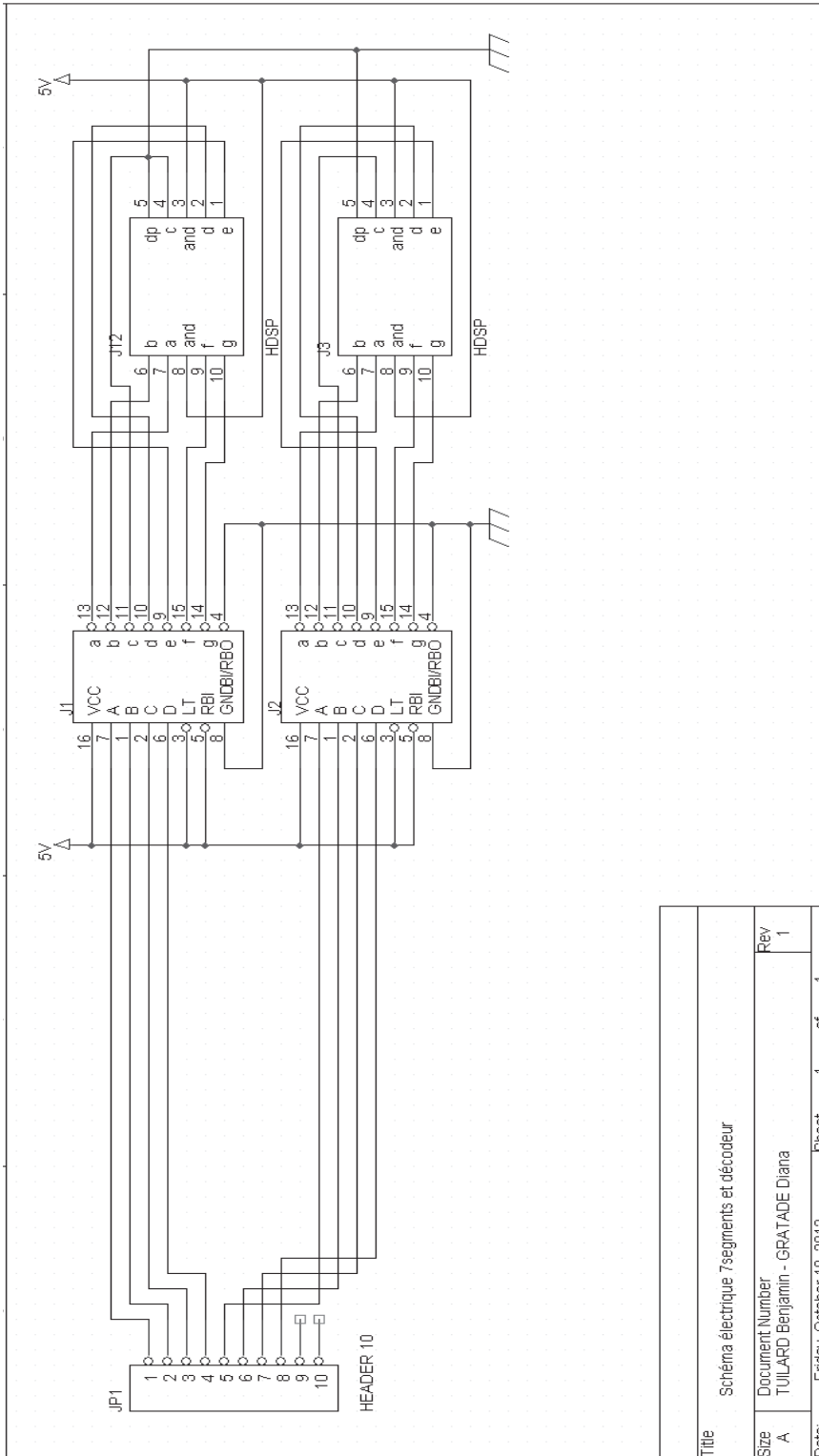
Titre		Projet Horloge à LED - Schématique du circuit des drivers à LED	
Size	A	Document Number	TUILARD Benjamin - GRATADE Diana
Date:	Saturday, November 24, 2012	Sheet	1 of 1

Le typon



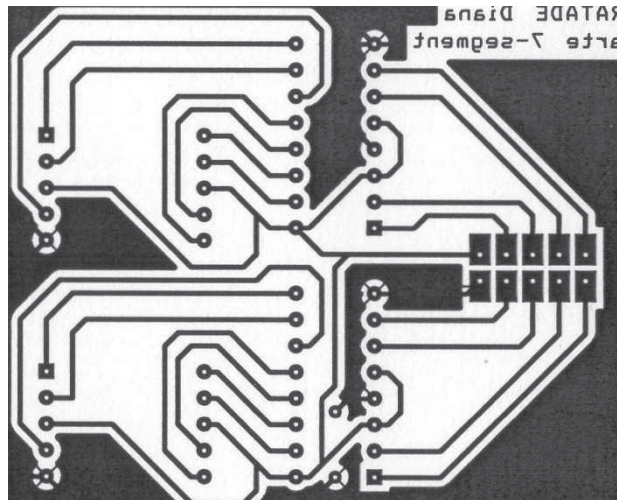
Annexe 7 - La carte des afficheurs 7-segments

Le schéma électrique

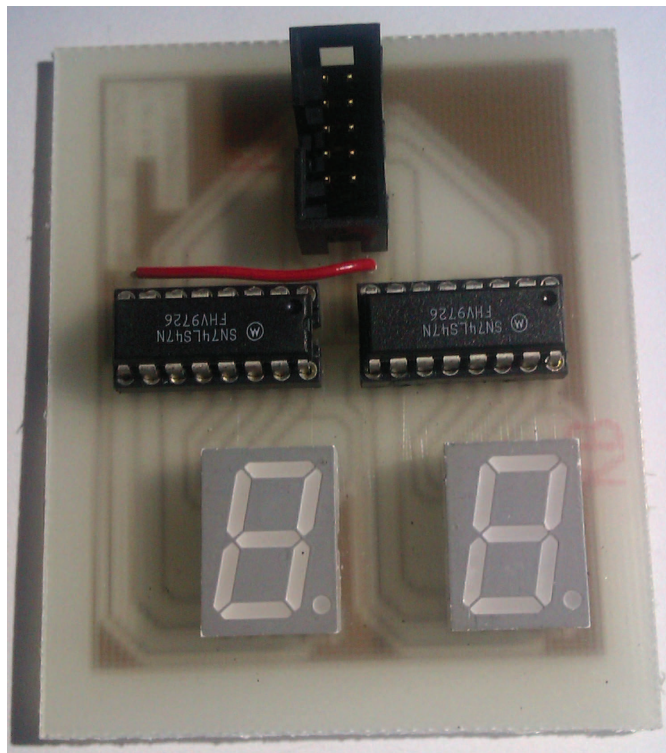


Titre		Schéma électrique 7segments et décodeur	
Size	A	Document Number	TULLARD Benjamin - GRATADE Diana
Date:	Friday, October 19, 2012	Sheet	1 of 1
		Rev	1

Le typon

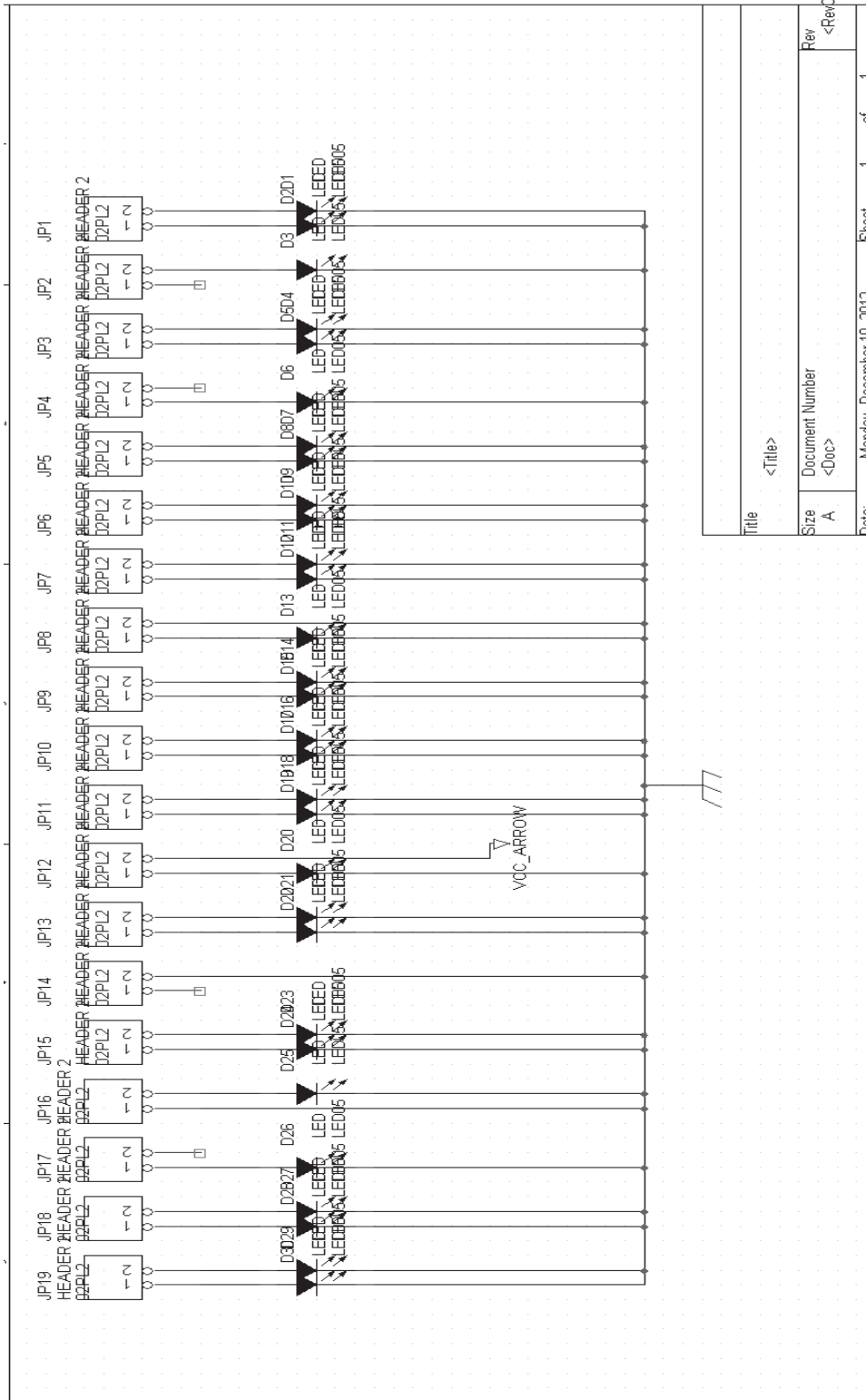


Carte finale



Annexe 8 - La carte de l'affichage traditionnel

Le schéma électrique



Annexe 9 – Programme de la CPLD

```
module horloge (clock, data, heure, minute, mode); // Déclaration du nombre du programme avec // toutes les variables utilisées

    input clock;
    input [5:0] data; // Déclaration des entrées ( [5:0] nombre de // bit soit ici 6)
    input mode;

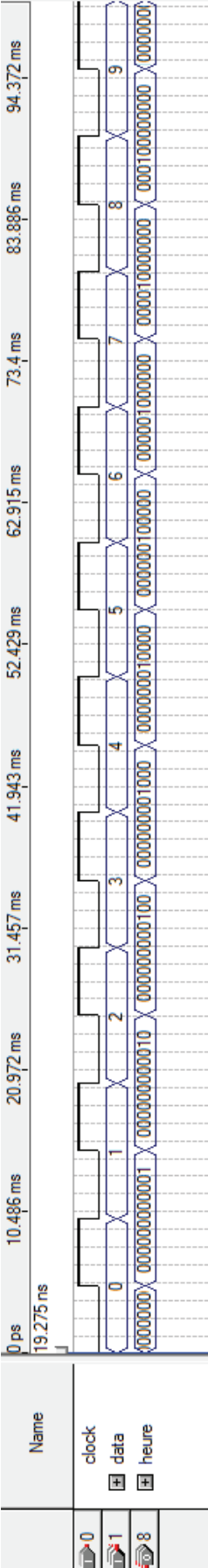
    output [59:0] minute; // Déclaration des sorties
    output [11:0] heure;

    reg [59:0] minute; // Déclaration des registres pour garder en // mémoire la valeur des valeurs
    reg [11:0] heure;

    always@ ( posedge clock ) // A chaque front montant de clock
    begin
        if (mode == 0) // on regarde si MODE est à 0
        begin
            minute = 2**data; // on affecte à minute la valeur 2DATA
        end

        if (mode == 1) // on regarde si MODE est à 1
        begin
            heure = 2** data; // On affecte à heure la valeur 2DATA
        end
    end
end
endmodule // Fin du programme
```


Annexe 10 – Simulation du programme de la CPLD



Annexe 11 – Datasheet de la CPLD



MAX 7000 Programmable Logic Device Family

September 2005, ver. 6.7

Data Sheet

Features...

- High-performance, EEPROM-based programmable logic devices (PLDs) based on second-generation MAX[®] architecture
- 5.0-V in-system programmability (ISP) through the built-in IEEE Std. 1149.1 Joint Test Action Group (JTAG) interface available in MAX 7000S devices
 - ISP circuitry compatible with IEEE Std. 1532
- Includes 5.0-V MAX 7000 devices and 5.0-V ISP-based MAX 7000S devices
- Built-in JTAG boundary-scan test (BST) circuitry in MAX 7000S devices with 128 or more macrocells
- Complete EPLD family with logic densities ranging from 600 to 5,000 usable gates (see [Tables 1 and 2](#))
- 5-ns pin-to-pin logic delays with up to 175.4-MHz counter frequencies (including interconnect)
- PCI-compliant devices available



For information on in-system programmable 3.3-V MAX 7000A or 2.5-V MAX 7000B devices, see the [MAX 7000A Programmable Logic Device Family Data Sheet](#) or the [MAX 7000B Programmable Logic Device Family Data Sheet](#).

Table 1. MAX 7000 Device Features

Feature	EPM7032	EPM7064	EPM7096	EPM7128E	EPM7160E	EPM7192E	EPM7256E
Usable gates	600	1,250	1,800	2,500	3,200	3,750	5,000
Macrocells	32	64	96	128	160	192	256
Logic array blocks	2	4	6	8	10	12	16
Maximum user I/O pins	36	68	76	100	104	124	164
t_{PD} (ns)	6	6	7.5	7.5	10	12	12
t_{SU} (ns)	5	5	6	6	7	7	7
t_{FSU} (ns)	2.5	2.5	3	3	3	3	3
t_{CO1} (ns)	4	4	4.5	4.5	5	6	6
f_{CNT} (MHz)	151.5	151.5	125.0	125.0	100.0	90.9	90.9

Annexe 12 – Datasheet de l'ATmega

Features

- High-performance, Low-power AVR[®] 8-bit Microcontroller
- Advanced RISC Architecture
 - 130 Powerful Instructions – Most Single Clock Cycle Execution
 - 32 x 8 General Purpose Working Registers
 - Fully Static Operation
 - Up to 16 MIPS Throughput at 16 MHz
 - On-chip 2-cycle Multiplier
- Nonvolatile Program and Data Memories
 - 8K Bytes of In-System Self-Programmable Flash
 - Endurance: 10,000 Write/Erase Cycles
 - Optional Boot Code Section with Independent Lock Bits
 - In-System Programming by On-chip Boot Program
 - True Read-While-Write Operation
 - 512 Bytes EEPROM
 - Endurance: 100,000 Write/Erase Cycles
 - 512 Bytes Internal SRAM
 - Programming Lock for Software Security
- Peripheral Features
 - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
 - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
 - Real Time Counter with Separate Oscillator
 - Four PWM Channels
 - 8-channel, 10-bit ADC
 - 8 Single-ended Channels
 - 7 Differential Channels for TQFP Package Only
 - 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x for TQFP Package Only
 - Byte-oriented Two-wire Serial Interface
 - Programmable Serial USART
 - Master/Slave SPI Serial Interface
 - Programmable Watchdog Timer with Separate On-chip Oscillator
 - On-chip Analog Comparator
- Special Microcontroller Features
 - Power-on Reset and Programmable Brown-out Detection
 - Internal Calibrated RC Oscillator
 - External and Internal Interrupt Sources
 - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
 - 32 Programmable I/O Lines
 - 40-pin PDIP, 44-lead TQFP, 44-lead PLCC, and 44-pad QFN/MLF
- Operating Voltages
 - 2.7 - 5.5V for ATmega8535L
 - 4.5 - 5.5V for ATmega8535
- Speed Grades
 - 0 - 8 MHz for ATmega8535L
 - 0 - 16 MHz for ATmega8535




8-bit AVR[®]
Microcontroller
with 8K Bytes
In-System
Programmable
Flash

ATmega8535
ATmega8535L

2502K-AVR-10/06



Annexe 13 – Datasheet du driver à LED



MICROCHIP MCP23017/MCP23S17

16-Bit I/O Expander with Serial Interface

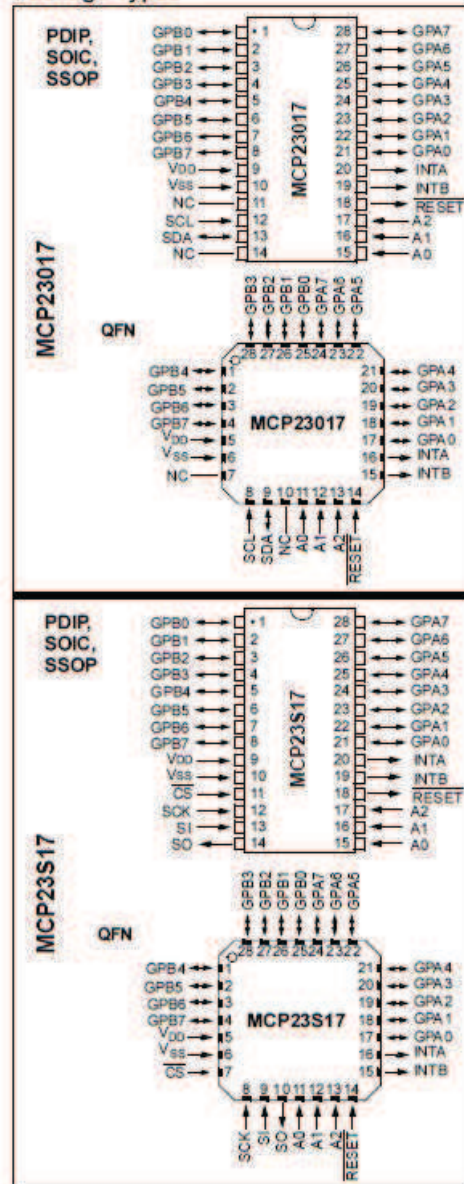
Features

- 16-bit remote bidirectional I/O port
 - I/O pins default to input
- High-speed I²C™ interface (MCP23017)
 - 100 kHz
 - 400 kHz
 - 1.7 MHz
- High-speed SPI interface (MCP23S17)
 - 10 MHz (max.)
- Three hardware address pins to allow up to eight devices on the bus
- Configurable interrupt output pins
 - Configurable as active-high, active-low or open-drain
- INTA and INTB can be configured to operate independently or together
- Configurable interrupt source
 - Interrupt-on-change from configured register defaults or pin changes
- Polarity Inversion register to configure the polarity of the input port data
- External Reset input
- Low standby current: 1 µA (max.)
- Operating voltage:
 - 1.8V to 5.5V @ -40°C to +85°C
 - 2.7V to 5.5V @ -40°C to +85°C
 - 4.5V to 5.5V @ -40°C to +125°C

Packages

- 28-pin PDIP (300 mil)
- 28-pin SOIC (300 mil)
- 28-pin SSOP
- 28-pin QFN

Package Types



Annexe 14 – Datasheet du décodeur 7-segments

SN74LS47

BCD to 7-Segment Decoder/Driver

The SN74LS47 are Low Power Schottky BCD to 7-Segment Decoder/Drivers consisting of NAND gates, input buffers and seven AND-OR-INVERT gates. They offer active LOW, high sink current outputs for driving indicators directly. Seven NAND gates and one driver are connected in pairs to make BCD data and its complement available to the seven decoding AND-OR-INVERT gates. The remaining NAND gate and three input buffers provide lamp test, blanking input/ripple-blanking output and ripple-blanking input.

The circuits accept 4-bit binary-coded-decimal (BCD) and, depending on the state of the auxiliary inputs, decodes this data to drive a 7-segment display indicator. The relative positive-logic output levels, as well as conditions required at the auxiliary inputs, are shown in the truth tables. Output configurations of the SN74LS47 are designed to withstand the relatively high voltages required for 7-segment indicators.

These outputs will withstand 15 V with a maximum reverse current of 250 μ A. Indicator segments requiring up to 24 mA of current may be driven directly from the SN74LS47 high performance output transistors. Display patterns for BCD input counts above nine are unique symbols to authenticate input conditions.

The SN74LS47 incorporates automatic leading and/or trailing-edge zero-blanking control (RBI and RBO). Lamp test (LT) may be performed at any time which the BI/RBO node is a HIGH level. This device also contains an overriding blanking input (BI) which can be used to control the lamp intensity by varying the frequency and duty cycle of the BI input signal or to inhibit the outputs.

- Lamp Intensity Modulation Capability (BI/RBO)
- Open Collector Outputs
- Lamp Test Provision
- Leading/Trailing Zero Suppression
- Input Clamp Diodes Limit High-Speed Termination Effects

GUARANTEED OPERATING RANGES

Symbol	Parameter	Min	Typ	Max	Unit
V_{CC}	Supply Voltage	4.75	5.0	5.25	V
T_A	Operating Ambient Temperature Range	0	25	70	$^{\circ}$ C
I_{OH}	Output Current – High BI/RBO			-50	μ A
I_{OL}	Output Current – Low BI/RBO BI/RBO			3.2	mA
$V_{OL(off)}$	Off-State Output Voltage a to g			15	V
$I_{OL(on)}$	On-State Output Current a to g			24	mA



ON Semiconductor
Formerly a Division of Motorola
<http://onsemi.com>

**LOW
POWER
SCHOTTKY**



PLASTIC
N SUFFIX
CASE 648



SOIC
D SUFFIX
CASE 751B

ORDERING INFORMATION

Device	Package	Shipping
SN74LS47N	16 Pin DIP	2000 Units/Box
SN74LS47D	16 Pin	2500/Tape & Reel

Annexe 15 – Datasheet du RTC



DS1307

64 x 8, Serial, I²C Real-Time Clock

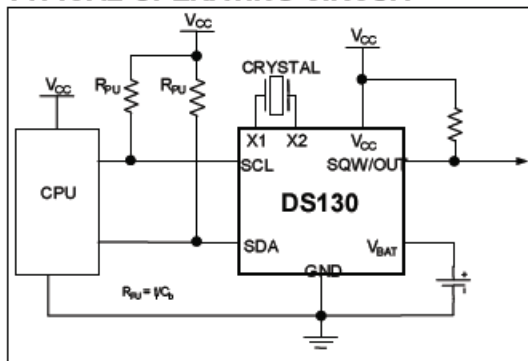
GENERAL DESCRIPTION

The DS1307 serial real-time clock (RTC) is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially through an I²C, bidirectional bus. The clock/calendar provides seconds, minutes, hours, day, date, month, and year information. The end of the month date is automatically adjusted for months with fewer than 31 days, including corrections for leap year. The clock operates in either the 24-hour or 12-hour format with AM/PM indicator. The DS1307 has a built-in power-sense circuit that detects power failures and automatically switches to the backup supply. Timekeeping operation continues while the part operates from the backup supply.

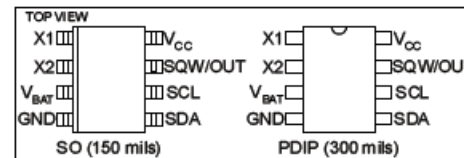
FEATURES

- Real-Time Clock (RTC) Counts Seconds, Minutes, Hours, Date of the Month, Month, Day of the week, and Year with Leap-Year Compensation Valid Up to 2100
- 56-Byte, Battery-Backed, General-Purpose RAM with Unlimited Writes
- I²C Serial Interface
- Programmable Square-Wave Output Signal
- Automatic Power-Fail Detect and Switch Circuitry
- Consumes Less than 500nA in Battery-Backup Mode with Oscillator Running
- Optional Industrial Temperature Range: -40°C to +85°C
- Available in 8-Pin Plastic DIP or SO
- Underwriters Laboratories (UL) Recognized

TYPICAL OPERATING CIRCUIT



PIN CONFIGURATIONS



ORDERING INFORMATION

PART	TEMP RANGE	VOLTAGE (V)	PIN-PACKAGE	TOP MARK*
DS1307+	0°C to +70°C	5.0	8 PDIP (300 mils)	DS1307
DS1307N+	-40°C to +85°C	5.0	8 PDIP (300 mils)	DS1307N
DS1307Z+	0°C to +70°C	5.0	8 SO (150 mils)	DS1307
DS1307ZN+	-40°C to +85°C	5.0	8 SO (150 mils)	DS1307N
DS1307Z+T&R	0°C to +70°C	5.0	8 SO (150 mils) Tape and Reel	DS1307
DS1307ZN+T&R	-40°C to +85°C	5.0	8 SO (150 mils) Tape and Reel	DS1307N

+Denotes a lead-free/RoHS-compliant package.

*A "+" anywhere on the top mark indicates a lead-free package. An "N" anywhere on the top mark indicates an industrial temperature range device.

For pricing, delivery, and ordering information, please contact Maxim Direct at 1-888-629-4642, or visit Maxim's website at www.maximintegrated.com.

REV: 100208