République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique UNIVERSITE MOULOUD MAMMERI DETIZI-OUZOU

FACULTE DU GENIE ELECTRIQUE ET D'INFORMATIQUE

DEPARTEMENT D'INFORMATIQUE

MEMOIRE DE FIN D'ETUDE DE MASTER ACADEMIQUE

Domaine : Mathématiques et Informatique

Filière : Informatique

Spécialité : Réseaux, Mobilités et Systèmes Embarqués

Présenté par

AZOUAOU Sadia

OUAGUENI Souhila

Thème

Mise en œuvre du protocole de communication I2c dans une smart home

Mémoire soutenu devant le jury composé de :

Présidente : Mme Belkadi Malika

Examinateur : Mme Oukfif Karima

Encadreur : Mr DAOUI Mehammed

Année universitaire : 2018/2019

Remerciements

Nous tenons particulièrement à remercier Allah le tout puissant, ce mémoire n'aurait jamais été réalisé sans sa bénédiction.

Nous adressons nos remerciements à notre encadreur Monsieur

Daoui Mehammed, professeur au département d'informatique de l'Université de Tizi Ouzou, pour la qualité de son encadrement exceptionnel, *pour son aide consistante, pour ses conseils judicieux,* pour sa patience, sa rigueur et sa disponibilité durant notre préparation de ce mémoire.

.

Nos plus grands remerciements vont surtout à Monsieur Ravi pour ces efforts et son aide dans notre travail.

Nous remercions nos chers membres de jury d'avoir accepté d'examiner notre travail.

Nous tenons à remercier nos familles de nous avoir soutenus, nous ne serons jamais assez reconnaissants envers nos parents qui ont toujours tout mis en œuvre pour qu'on s'épanouisse dans tout ce qu'on entreprend.

Enfin, nous tenons à remercier toute personne nous ayant aidé de près ou de loin durant notre travail.

Dédicaces

Je dédie ce mémoire à :

Mes parents :

Ma mère, qui a œuvré pour ma réussite, par son amour, tous ses sacrifices consentis et ses précieux conseils, pour toute son assistance et sa présence dans ma vie, reçois à travers ce travail aussi modeste soit-il, l'expression de mes sentiments et de mon éternelle gratitude.

Mon père, qui peut être fier de trouver ici le résultat de longues années de sacrifices et de privations pour m'aider à avancer dans la vie. Puisse

Dieu faire en sorte que ce travail porte son fruit ; Merci pour les valeurs nobles, l'éducation et le soutient permanent venu de toi.

Mon mari sofiane,, Mon frère Meziane, ma sœur Sabrina qui n'ont cessé d'être pour moi des exemples de persévérance, de courage et de générosité, et ma chère copine Souhila.

Mon encadreur Mr Daoui.M Qui doit voir dans ce travail la fierté d'un savoir bien acquis.

Sadia

Dédicaces

A la mémoire de mon père puisse dieu l'accueillir dans son infini miséricorde.

Je dédie ce mémoire à :

Ma mère, qui a œuvré pour ma réussite, par son amour, tous ses sacrifices consentis et ses précieux conseils, pour toute son assistance et sa présence dans ma vie, reçois à travers ce travail aussi modeste soit-il, l'expression de mes sentiments et de mon éternelle gratitude.

Mes frères(Mahdi&Samir) et sœurs (Karima, Dalila, Djamila et layla) et leurs enfants et ayi qui n'ont cessé d'être pour moi des exemples de persévérance, de courage et de générosité, et ma chère copine Samia

Sans oublier mon future marie Jérémy.

Mon encadreur Mr Daoui.M Qui doit voir dans ce travail la fierté d'un savoir bien acquis.

Souhila

Sommaire

Introduction générale 1

I.1. Introduction 3

I.2. Définition de la domotique 3

I.3. Historique 3

I.4. Fonctionnement de la domotique 3

I.5. Différents domaines de la domotique 4

I.5.1. Domotique pour le confort 4

I.5.2. Domotique pour l'énergie 4

I.5.3. Domotique pour la sécurité 4

I.5.4. Domotique pour la santé 4

I.6 Technologies de la domotique 5

I.6.1. La technologie par réseau sans fil 5

I.6.2. La technologie CPL 7

I.6.3 La technologie par réseau câblé 7

I.7. Maisons intelligentes 7

I.8. Maison intelligente et la domotique 8

I.9. Avantages et inconvénients de la domotique 9

I.9.1. Avantages 9

I.9.2. Inconvénients 9

I.10. Plug and Play 10

I.11. Protocoles de communication 10

I.11.1. One Wire-bus 10

I.11.2. SPI 12

I.11.3. I2C 13

I.11.4. KNX 15

I.11.5. CPL 16

I.11.6. Bus CAN 17

I.12. Conclusion 19

II.1. Introduction 21

II.2. Définition du bus I2C 21

II.3. Principe du bus I2C 22

II.4. Domaines d'application I2C 22

II.5. Caractéristiques 23

II.6. Le protocole I2C 23
II.6.1. La prise de contrôle du bus 24
II.6.2. Start condition 24
II.6.3. Repeated Start 25
II.6.4. Stop condition 25
II.6.5. Acknowledge (ACK) 25
II.1. Not acknowledge (NACK) 26
II.6.7. Transmission d'un bit 26
II.6.8. La transmission d'un octet 27
II.6.9. La transmission d'une adresse avec le bit R/W 27
II.7. LECTURE / ECRITURE DE DONNEES 29
II.7.1. Ecriture d'une donnée 29
II.7.2. Lecture d'une donnée 31
II.8. Avantages et inconvénients du protocole I2C 33
II.8.1. Avantages du protocole i2c 33
II.8.2. Inconvénients du protocole i2c 33
II.9. Conclusion 34
III.1. Introduction 37
III.2. Schéma synoptique global du système à réaliser 37
III.3. Conception Matérielle 38
III.3.1. Présentaon du Microcontrôleur (Pic16F877A) 38
III.3.2. Présentaon du capteur LM 35 41
III.3.3. Concepon du capteur I2C 42
III.3.4. Concepon de l'aconneur I2C 42
III.3.5. Concepon d'un module Maitre I2C 43
III.4. Conception logicielle 44
III.4.1. Schéma modulaire du programme 44
III.4.2. Implémentaon du capteur I2C 45
III.4.3. Implémentaon de l'aconneur I2C : 48
III.4.4. Implémentaon du maitre I2c 49
III.5. Conclusion 51
IV.1. Introduction 53
IV.2. Environnement de développement 53
IV.2.1. Compilateur MikroC 53
IV.2.2. Compilateur MPLAB 54

IV.2.3. Proteus Professional..... 54
IV.3. Outils de développement utilisés 56
IV.3.1. Easypic V7..... 56
IV.4. Edition des circuits 58
IV.4.1. Schémas du capteur I2C..... 58
IV.4.2. Schémas de l'aconneur I 2C..... 58
IV.4.3. Schémas du maitre I2C 59
IV.5. Conclusion 61
Conclusion générale 62
Bibliographie 63

Table des figures

Figure I.1 : illustration du confort, sécurité, énergie et santé ***5***
Figure I.2 : Le X2D (courant porteur) ***6***
Figure I.3 : Le réseau Zigbee ***6***
Figure I.4 : Le réseau Bluetooth ***6***
Figure I.5 : Figure illustrant Porteurs en Ligne (CPL) ***7***
Figure I.6 : schéma représentatif de fonctionnement général des équipements d'une maison intelligente ***8***
Figure I.7 : maison intelligente et la domotique ***9***
Figure I.8 : Plug and Play ***10***
Figure I.9: One Wire-bus ***11***
Figure I.10 : Exemple de configuration du protocole One wire ***12***
Figure I.11: SPI ***12***
Figure I.12: Exemple de configuration de bus SPI avec un maître et plusieurs esclaves ***13***
Figure I.13 : I2C ***14***
Figure I.14: fonconnement de KNX 15
Figure I.15 : CPL 16
Figure I.16: Exemple de câblage 17
Figure I.17: Tableau récapitulatif des fonctionnalités 17
Figure I.18 : bus Can 18
Figure I.19 : L'avantage du bus Can 19

Figure II. 1: Exemple de configuraon de bus I 2C avec plusieurs maîtres et plusieurs esclaves *22*
Figure II. 2: Structure d'E/S d'un module I2C *22*
Figure II. 3: Structure d'E/S d'un module I2C *23*
Figure II. 4: prise de contrôle de bus *24*
Figure II. 5: Start condition *24*
Figure II. 6: Repeated Start *25*
Figure II. 7 : Stop condition *25*
Figure II. 8: bit Acknowledge *26*
Figure II. 9: bit Not Acknowledge *26*
Figure II. 10: bit de donnée *26*
Figure II. 11: la transmission d'un octet *27*
Figure II. 12: La transmission d'une adresse avec le bit R/W *28*
Figure II. 13: Ecriture du maitre *30*
Figure II. 14: signal électrique de l'écriture *31*
Figure II. 15: Lecture d'une donnée *32*
Figure II. 16: signal électrique de la lecture *32*
Figure II. 17: Ecriture et l'écriture du maitre *32*
Figure II. 18: signal électrique de la lecture et l'écriture *33*

Figure III.1*:* Le schéma synoptique global de système à réaliser *37*
Figure III.2: Les di□érentes familles de PIC *38*
Figure III.3: Exemple de l'***identification PIC***16F877-A *39*
Figure III.4: Brochage du PIC16F877 ..... *40*
Figure III.5: brochage du LM35 ..... *41*
Figure III.6: Schéma électrique du capteur I2C *42*
Figure III.7: Schéma électrique de l'aconneur I 2C *43*
Figure III.8: Schéma électrique du maitre I2C *44*
Figure III.9: schéma modulaire du programme *45*
Figure III.10: organigramme du capteur I2C *47*
Figure III.11: organigramme du l'aconneur I 2C *49*
Figure III.12: organigramme du maitre I2C *51*

Figure IV.1 : la fenêtre ISIS.....55
Figure IV.2 : la fenêtre ARES.....56
Figure IV.3 : Easypic.....57
Figure IV.4 : la fenêtre mikroProg Suite for pic.....57
Figure IV.5 : schémas du capteur I2C.....58
Figure IV.6 : schémas de l'aconneur I 2C.....59
Figure IV.7 : schémas du maitre I2C.....60
Figure IV.8 : photo réelle de notre projet.....60

Introduction générale

L'internet des Objets (IoT) s'annonce comme une évolution sans précédent. Cette révolution facilite la création d'objets intelligents permettant des avancées dans de multiples domaines. L'un des domaines les plus affectés par l'émergence de l'IoT est la domotique. Ce dernier est un domaine technologique qui vise à automatiser et faire communiquer entre eux des équipements afin d'en faciliter l'interaction et d'interagir avec l'environnement.

La mise en place d'une centrale de domotique nécessite de nouveaux capteurs et actionneurs intelligents et auto-configurables. A titre d'exemple, un interrupteur (actionneur dans ce cas) doit pouvoir permettre d'allumer ou d'éteindre une lampe de façon manuelle, mais pouvoir aussi recevoir la commande d'allumage et d'extinction par la centrale. Il doit également retourner l'état de sa lampe (allumée ou éteinte). Le même cas est valable pour les capteurs de température, de lumière des robinets d'eau et de gaz et des télécommandes de téléviseurs, de climatisation, ...etc.

Notre mémoire a pour objectif la mise en place du protocole de communication I2C entre une centrale de domotique et ses capteurs/actionneurs. Ce protocole permettra à la centrale de reconnaitre tous les capteurs et actionneurs intelligents se trouvant dans son espace et faciliter leur gestion. Autrement dit, l'installateur des équipements (capteurs, actionneurs...) n'aura qu'à brancher ces derniers à la centrale pour qu'ils soient reconnus et utilisés par la centrale, comme c'est le cas des équipements Plug and Play classiques

Le présent travail s'articulera autour de quatre chapitres :

Le premier fait objet d'un état de l'art sur la technologie actuelle la domotique, les maisons intelligentes, Plug and Play, protocoles de communication etc...

Dans le deuxième chapitre nous allons étudier le protocole de bus I2C qui est un protocole de communication série rapide et facile à mettre en œuvre.

Le troisième chapitre est réservé pour la conception de notre travail ; nous allons donner une description détaillée sur le microcontrôleur PIC qu'on a utilisé dans notre projet, ensuite nous présenterons les différents schémas, organigrammes qui expliquent bien le fonctionnement de notre système.

Le quatrième chapitre est consacré à la réalisation en exposant les outils et l'environnement de travail utilisés, on va présenter a la fin de ce chapitre les circuits réalisés …

Ce présent mémoire se termine par une conclusion résumant les connaissances acquises au travers du mémoire réalisé, ainsi que des propositions pouvant venir améliorer notre protocole dans le futur.

Chapitre I :

Domotique, maison intelligente, protocoles de communication

I.1. Introduction

Les progrès technologiques, notamment de l'informatique, de la télécommunication et de l'électronique ont permis le développement de systèmes de transmission, des commandes à distance et favoriser l'éclosion d'une offre abondante de nouveaux services pour les occupants des logements.

Dans ce chapitre, nous allons voir une présentation générale de la domotique, ses secteurs d'application, les différents types de technologies, et les différents protocoles de communications séries utilisés dans cette dernière.

I.2. Définition de la domotique

Le terme domotique issu du latin "Domus", qui signifie maison et du suffixe "tique "pour informatique, regroupe l'ensemble des objets connectés et des applications permettant d'automatiser et d'améliorer les tâches au sein d'une maison**.** *[1]*

La domotique opère dans un champ technique et informatique très vaste permettant de programmer la plupart des appareils et dispositifs électriques de la maison, depuis l'éclairage et le chauffage jusqu'aux équipements audiovisuels et électroménagers, en passant par l'ouverture des fenêtres. Elle facilite également le contrôle en gérant les systèmes d'alarme, les préventions d'incendie, ou encore la température au sein des pièces. *[2]*

I.3. Historique

Les premiers travaux de domotique sont apparus dans les années 70 avec les problématiques énergétiques dues aux crises pétrolières. Ces crises marquent le début du développement de l'électronique pour les bâtiments. Au départ, la domotique contrôle seulement les prises, l'éclairage et les volets roulants grâce à une télécommande. Au fur et à mesure, de nouveaux objets se mettent en réseau comme les thermostats et les alarmes.

Mais c'est véritablement à partir de la fin du 20e siècle, que la domotique va se démocratiser avec le développement des technologies sans fil comme le wifi ou le Bluetooth, la miniaturisation des composants électroniques, l'avènement des appareils mobiles, l'invasion des écrans tactiles et des télévisions connectées, les ingénieurs peuvent désormais proposer au public des produits - objets connectés ou systèmes domotiques – bien plus puissants et simples d'utilisation. *[3]*

I.4. Fonctionnement de la domotique

Sa fonction est de programmer, de contrôler et d'automatiser, à distance ou sur place, tous les appareils du domicile intégrés au sein du réseau. Ce dernier fonctionne avec ou sans fil afin de recevoir et de transmettre des données et les commandes entre les différents appareils à

contrôler. Dans le sens inverse, chaque appareil peut communiquer sur son état de fonctionnement et recevoir des commandes.

Par exemple, si votre cafetière n'est pas équipée d'un programmateur, en branchant un module entre la prise murale et le cordon de votre machine, il est possible de la programmer pour préparer votre café au moment de votre réveil. Avec du matériel plus évolué, c'est toute une série de situations en chaîne, appelées scénarios, qui est possible. *[2]*

I.5. Différents domaines de la domotique

I.5.1. Domotique pour le confort

Gestion de l'éclairage, gestion du chauffage, gestion des volets roulants, par simple action d'une commande, toutes ces tâches sont simplifiées grâce à la domotique. Cette dernière permet d'améliorer le confort d'usage. Grâce à une application installée sur son Smartphone, par exemple, les habitants d'une maison connectée peuvent décider de l'heure d'ouverture des volets, de la température des pièces selon l'heure de la journée. Des capteurs installés un peu partout dans la maison détectent la présence des individus et peuvent ainsi donner le signal pour allumer ou éteindre les lumières dans une pièce, activer la température optimale et même aller jusqu'à démarrer une musique d'ambiance dans le salon si les habitants l'ont choisie. *[4]*

I.5.2. Domotique pour l'énergie

Parmi ces technologies, de nombreux automatismes : gestion des volets, gestion des équipements de chauffage rendent les maisons réactives aux conditions extérieures (climat) et intérieures (usage), l'objectif final étant de réduire les dépenses quotidiennes d'énergie tout en préservant le confort des habitants. *[2]*

I.5.3. Domotique pour la sécurité

Un des domaines d'application de la domotique est la sécurité des biens et des personnes par des systèmes d'alarme qui préviennent d'une part des risques techniques (pannes ou disfonctionnements des appareils) et d'autre part des éventuelles intrusions dans la maison (cambriolage). *[4]*

I.5.4. Domotique pour la santé

La domotique trouve aujourd'hui de nouvelles applications dans le domaine de la santé. En installant des systèmes domotiques dans les maisons des personnes en situation de handicap, atteintes de maladies neurodégénératives telles que la maladie d'Alzheimer ou encore des personnes âgées, il est possible de les aider dans leur quotidien en automatisant le plus possible des tâches considérées comme complexes. Cela permet également à la personne de rester à son domicile plus longtemps et d'être suivie à distance. *[3]*

Figure I.1 : illustration du confort, sécurité, énergie et santé

I.6 Technologies de la domotique

Dans le cadre d'une utilisation à l'échelle d'un habitat, la domotique concerne trois technologies : la technologie par réseau sans fil, la technologie par réseau câblé et la technologie CPL (Courant porteur en ligne).

I.6.1. La technologie par réseau sans fil

La technologie sans fil utilise plusieurs supports technologiques : les ondes radio ou RF (sur des fréquences en MHz) et l'infrarouge ou IR, qui a pour inconvénient de ne pas traverser les murs. Il est conseillé, pour une meilleure stabilité du système, de ne pas mixer le sans-fil avec un autre type de technologie, le CPL par exemple. Cela peut nuire à l'installation et à la qualité de la communication entre les équipements. *[5]*

Les ondes radio sont employées par de multiples protocoles comme le X10 RF, le HomeEasy, le X2D, le Zigbee, le Zwave, ou encore le Bluetooth.

Les principales fréquences utilisées dans la domotique sont le 433 MHz, le 868MHz et le 2GHz. On trouve parmi les protocoles sans fil :

.

• **Le X2D** est mixte (courant porteur ou radio 868 MHz) convient à la domotique de sécurité et la domotique du chauffage. *[6]*

Figure II.2 : Le X2D (courant porteur)

• **Le réseau Zigbee**, basé sur le standard 802.15.4, ratifié par l'IEEE (Institute of Electrical and Electronics Engineers), a de plus en plus de fidèles. Il fonctionne avec des piles très longues durées d'autonomie, sur 866 MHz (bande libre en Europe) et 915 MHz (aux États-Unis). *[4]*

Figure III.3 : Le réseau Zigbee

- **Le Bluetooth**

Mise au point en 1994, la technologie Bluetooth est un standard de communication qui permet de recevoir mais également d'émettre des données sur une très courte distance grâce aux ondes radio UHF. Démocratisé dans le secteur des téléphones portables et de l'informatique, le Bluetooth a permis de simplifier les connexions entre les périphériques. Aujourd'hui, avec l'apparition d'objets connectés de plus en plus nombreux, son usage tend à se développer davantage au cœur de notre quotidien. *[7]*

Figure IV.4 : Le réseau Bluetooth

I.6.2. La technologie CPL

Utilise les prises de courant du domicile pour transmettre les informations entre les appareils et les unités de commande. Ainsi, chaque prise reçoit les données qui lui sont destinées pour une gestion simplifiée et un coût d'installation réduit. Ce dispositif facile à mettre en œuvre est parfait pour les locataires puisque l'installation les suit lors de leurs différents déménagements. En revanche, le défaut de cette technologie est qu'elle génère davantage de pollution électromagnétique que les autres. *[8]*

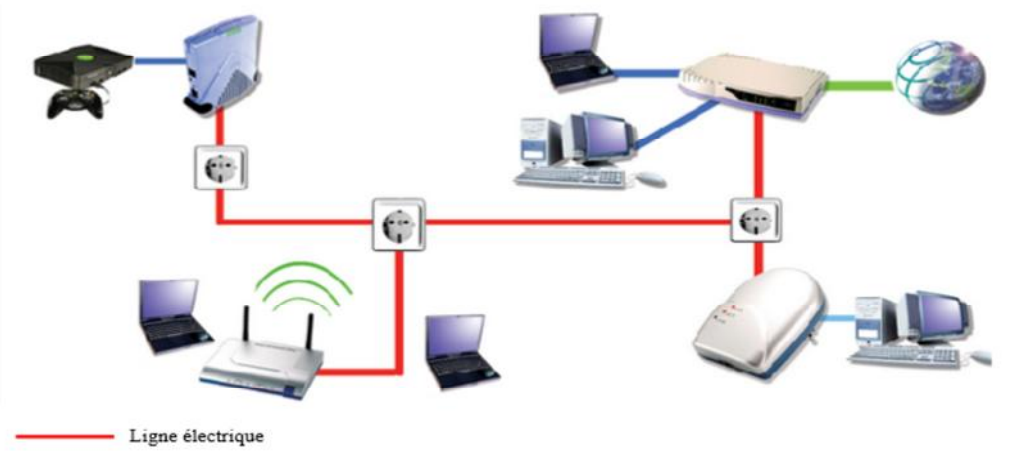


Figure V.5 : Figure illustrant Porteurs en Ligne (CPL)

I.6.3 La technologie par réseau câblé

Certains professionnels ne sont pas favorables, au sein d'une installation domotique, aux approches sans fil ou CPL. Ils leur préfèrent une domotique par câbles.

Le pré-câblage doit être souple et évolutif, car la technologie ne cesse d'évoluer. Il faut ainsi prévoir un local technique, le « local de répartition », qui centralise les points d'arrivée de toutes les liaisons externes (électricité, téléphone, Internet, télévision, fibre optique ...). *[9]*

I.7. Maisons intelligentes

Internet, objets connectés et domotique : voilà le trio qui a rendu possible l'avènement de la maison intelligente.

Pour faire simple, une maison intelligente est une maison dans laquelle plusieurs objets et appareils sont connectés à votre Smartphone. Du thermostat à l'éclairage, en passant par le système d'alarme ou le réfrigérateur, tous ces appareils intelligents (« smart devices ») communiquent entre eux par le biais d'une connexion internet sans fil.

Une maison intelligente est une maison partiellement ou totalement automatisée. Dans cette dernière, plusieurs petites applications peuvent être connectées. La transformation de l'habitation en maison intelligente peut donc se faire progressivement. Le grand intérêt d'une maison connectée est que la communication se fait de manière bilatérale. *[10]*

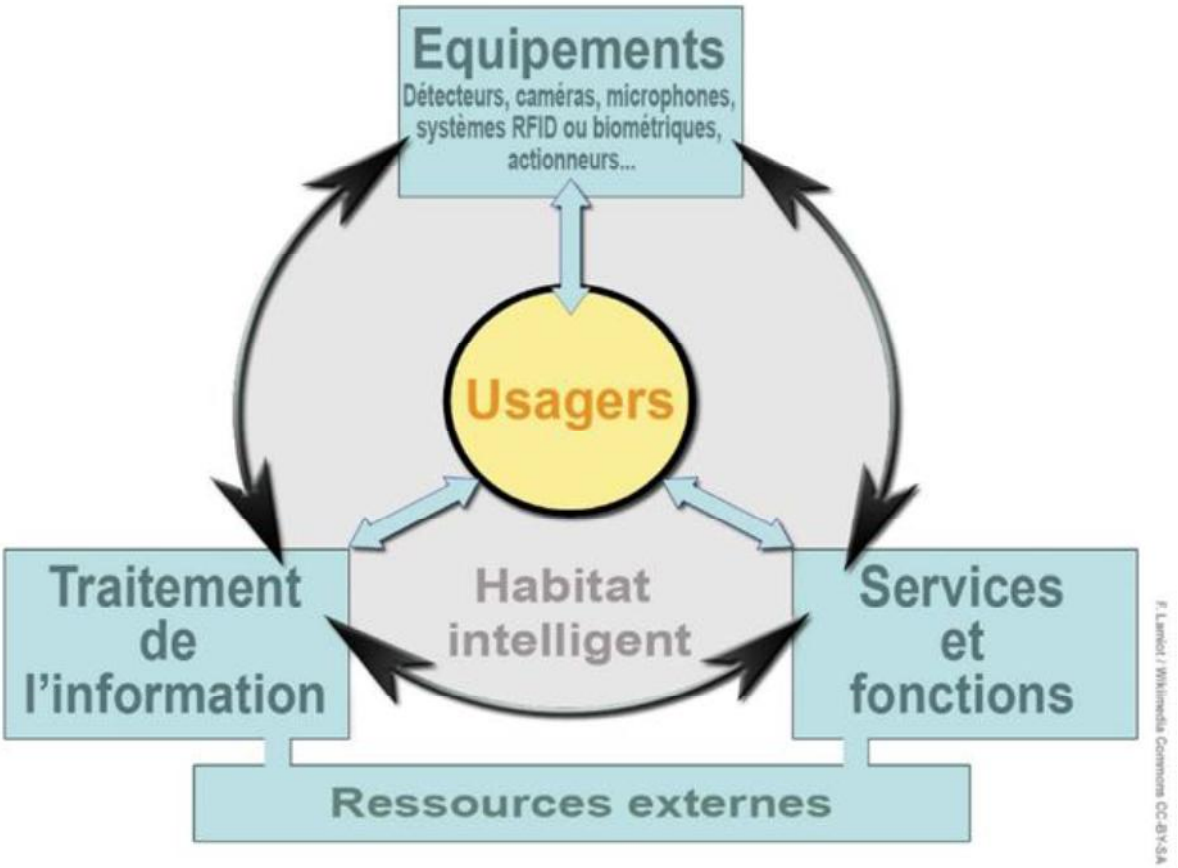


Figure VI.6 : schéma représentatif de fonctionnement général des équipements d'une maison intelligente

I.8. Maison intelligente et la domotique

Bien que foncièrement liés, les concepts de maison intelligente et de domotique présentent des différences notoires. Un système domotique constitue un ensemble intégré reprenant tous les composants d'une maison. Un tel système exige une installation spécifique effectuée par un installateur qualifié. C'est là que se situe la principale différence avec les nouvelles applications de la maison intelligente. Celles-ci sont faciles à utiliser et ne nécessitent aucune installation sophistiquée. Il suffit de posséder un Smartphone, une connexion internet et, bien entendu, une maison. *[11]*

Figure VII.7 : maison intelligente et la domotique

I.9. Avantages et inconvénients de la domotique

I.9.1. Avantages

- Simplification de la vie et optimisation du confort en adaptant une maison à différents scénarios de la vie quotidienne
- Economies d'énergie grâce à la gestion automatique du chauffage, de la climatisation et de l'éclairage et à la programmation des appareils électroménagers en heures creuses.
- Amélioration de la sécurité grâce à des alarmes, des systèmes d'ouverture automatique de la porte (reconnaissance vocale, carte magnétique…) En cas de tentative d'intrusion dans la maison, un appel téléphonique automatique peut contacter le propriétaire ou une entreprise de sécurité.
- Aide précieuse pour les personnes dépendantes et handicapées. *[12]*

I.9.2. Inconvénients

Le principal inconvénient est le prix d'achat et d'installation. Le prix est beaucoup plus élevé mais les factures d'énergie baisseront. Il faut donc le prendre en compte dans le budget initial.

Le deuxième inconvénient est le verrouillage qu'offrent certaines marques dans leurs produits ne permettant pas d'avoir un logiciel ouvert. *[12]*

I.10. Plug and Play

Le principe Plug and Play (« connecter et jouer » ou « brancher et utiliser ») se définit comme la possibilité d'utiliser un périphérique directement après branchement ; un système plus global le reconnaitra rapidement et automatiquement permettant son utilisation immédiate, avec un minimum d'intervention de la part de l'utilisateur. *[13]*

Figure VIII.8 : Plug and Play

I.11. Protocoles de communication

Pour assurer le Plug and Play, il faut utiliser des protocoles de communication. Ces derniers sont divers et varient entre les appareils électroniques et se diffèrent selon leurs utilisations.

I.11.1. One Wire-bus

- **Définition**

Dès que l'on parle de **Domotique**, le protocole 1-Wire rentre rapidement dans le jeu : il s'agit d'un bus série, nécessitant au minimum 1 fil (d'où son nom). *[14]*

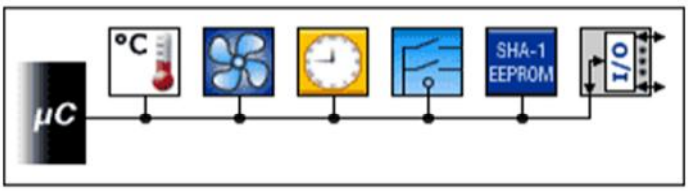


Figure IX.9: One Wire-bus

- **Fonctionnement**

Le bus 1-wire fonctionne en mode maitre-esclaves. Le maître contrôle le bus, envoie des commandes et interroge les sondes. Il ne peut y avoir qu'un seul maître sur un réseau et les sondes ne peuvent d'elles-mêmes envoyer des données sur le bus : pas d'interruptions comme en I2C.

Au repos, le bus présente un niveau électrique haut, ce qui permet d'alimenter les sondes par l'entrée data : c'est le mode « parasite ».

L'inconvénient est qu'il faut un certain temps pour qu'une sonde se « recharge » entre 2 interrogations. Mais il est aussi possible d'alimenter les sondes par un fil dédié (celles qui le peuvent, certaines ne fonctionnent qu'en parasite).

Le bus fonctionne indifféremment en 3.3 ou en 5 volts. Le 5 volts sera à privilégier pour un réseau long et/ou parasité. *[14]*

- **Caractéristiques**

- ✓ Asynchrone
- ✓ Type maître-esclave (single master, multi salves)
- ✓ Bidirectional, Half duplex *[15]*

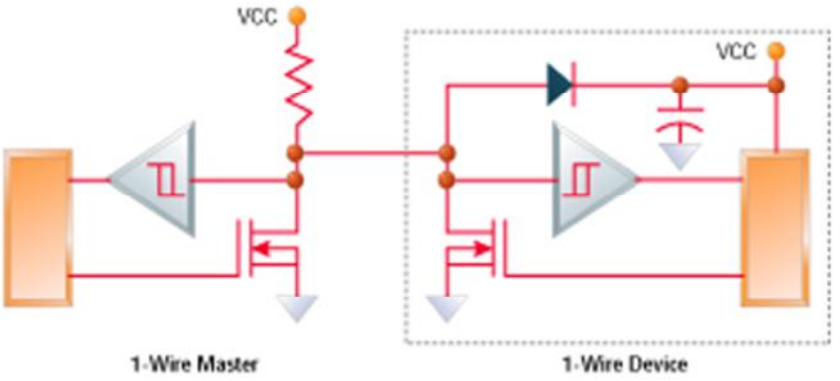


Figure X.10 : Exemple de configuration du protocole One wire

I.11.2. SPI

- **Definition**

SPI (Serial Périphérique Interface) est un bus série utilisé pour la transmission synchrone de données entre un maître et un ou plusieurs esclaves. La transmission est en full duplex. *[16]*

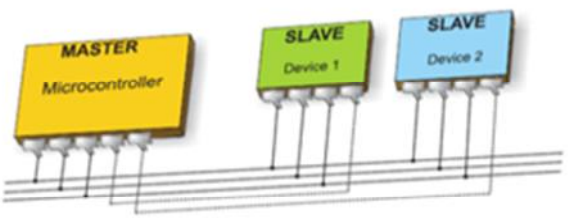


Figure XI.11: SPI

- **Fonctionnement et connexion**

Le maître génère l'horloge et initialise la transmission de données en sélectionnant l'esclave (convertisseur, registre à décalage, mémoire, …) avec qui il veut communiquer. Chaque esclave est sélectionné par une ligne SS (Slave Select) et n'est actif que lorsqu'il est sélectionné. Le bus SPI est composé de deux lignes de données et deux lignes de signal, toutes unidirectionnelles :

- ✓ MOSI (Master Out Slave In) : Sur la ligne MOSI le maître transmet des données à l'esclave.
- ✓ MISO (Master In Slave Out) : Sur la ligne MISO l'esclave transmet des données au maître.

- ✓ SCK (SPI Serial ClocK) : Signal d'horloge, généré par le maitre, qui synchronise la transmission. La fréquence de ce signal est fixée par le maître et est programmable.
- ✓ SS (Slave Select) : Ce signal placé au NL 0 permet de sélectionner (adresser) individuellement un esclave. Il y a autant de lignes SS que d'esclaves sur le bus. Le nombre possible de raccordements SS du maître limitera donc le nombre d'esclaves. *[16]*

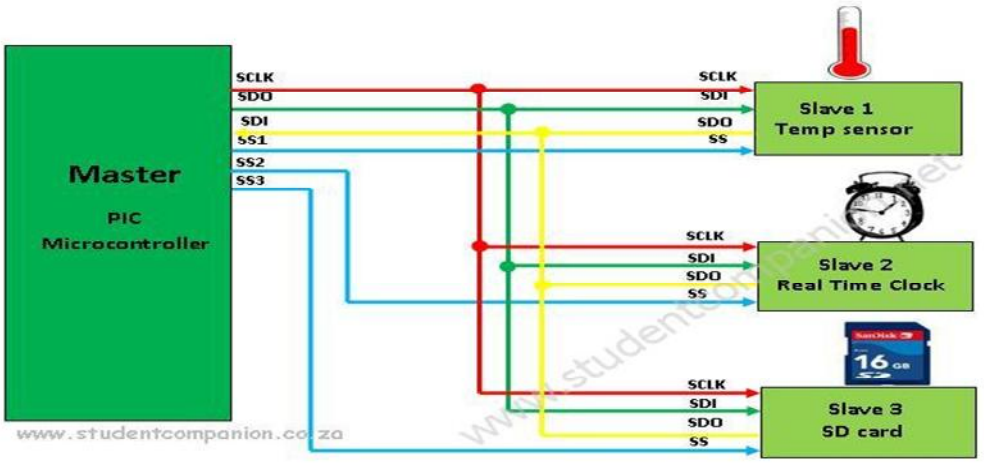


Figure XII.12: Exemple de configuration de bus SPI avec un maître et plusieurs esclaves

➢ **Caractéristiques**

- ✓ Synchrone
- ✓ Type maître-esclave (single master, multi salves)
- ✓ Unidirectionnel (Une ligne par direction)
- ✓ Full duplex *[16]*

I.11.3. I2C

➢ **Définition**

I2C est un bus série permettant de transmettre des informations de façon synchrone entre divers circuits connectés sur le bus. Le protocole de la liaison est du type MAITRE/ESCLAVE. Chaque circuit est reconnu par son adresse et peut être soit transmetteur soit receveur de l'information. Ces circuits peuvent être : Un ordinateur, un microcontrôleur, un microprocesseur, une mémoire, un périphérique (clavier, écran...) etc. *[17]*

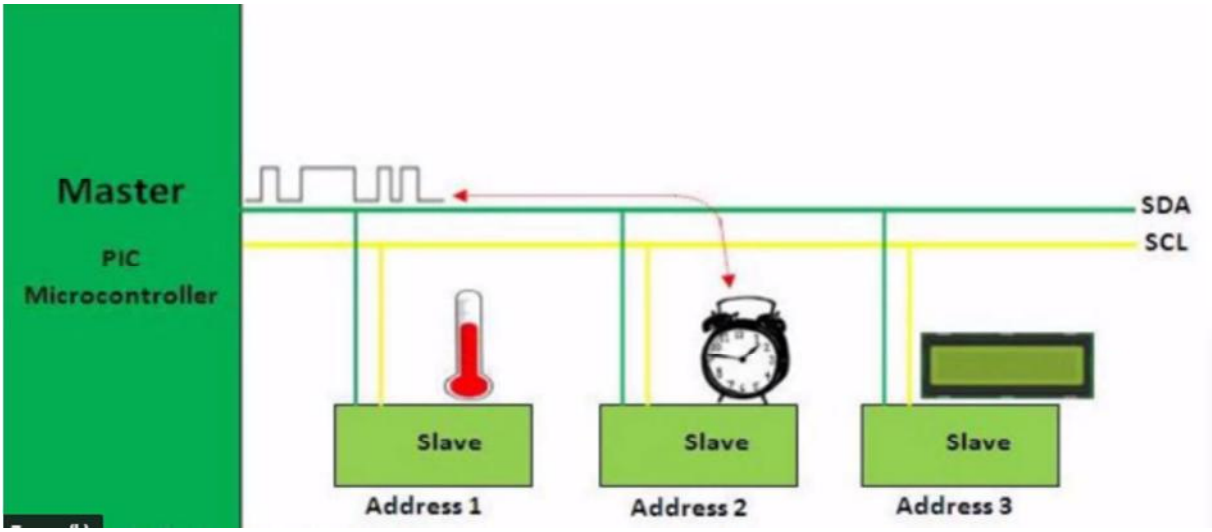


Figure XIII.13 : I2C

- **Fonctionnement**

Lors d'un échange, la ligne SCL est pilotée par l'initiateur de l'échange ou maître, quitte à ce que l'esclave agisse également dessus dans certains cas particuliers. La condition de départ, générée par le maître du bus est suivie par le premier octet de données, poids forts en tête. Après le huitième bit, l'émetteur qui est aussi le maître dans ce cas, met sa ligne SDA au niveau haut c'est à dire au repos mais continue à générer l'horloge sur SCL. Pour acquitter l'octet, le récepteur doit alors forcer la ligne SDA au niveau bas pendant l'état haut de SCL qui correspond à cet acquittement, prenant en quelque sorte la place d'un neuvième bit. Lorsque cet échange est terminé, le maître génère une condition d'arrêt. *[18]*

- **Caractéristiques**

- ✓ Synchrone
- ✓ Type maître-esclave (Multi masters, multi salves)
- ✓ Bidirectionnel, Halfduplex

Débit :

- ✓ Jusqu'à100 kbps en mode standard (version 1)
- ✓ Jusqu'à400 kbps en mode fast (version 1)
- ✓ Jusqu'à3, 4 Mbps en mode HS (version 2 / 2000) *[18]*

I.11.4. KNX

- **Définition**

KNX (ou Konnex) est essentiellement un bus de communication qui a été développé pour et par les acteurs du bâtiment. Le protocole KNX est un protocole à logique répartie. Contrairement à d'autres protocoles d'automatismes, celui-ci ne fonctionne pas en mode maître/esclave, chaque automate étant indépendant des autres. *[19]*

- **Fonctionnement**

Selon les besoins, les possibilités d'utilisation du protocole KNX sont nombreuses. Le protocole KNX offre différents types de capteur : bouton-poussoir, détecteur de présence, régulateur de niveau de luminosité, détecteur de variation de température… Ces capteurs sont sensibles aux commandes, ordres et mesures. Ils envoient les informations aux produits de sortie (actionneurs), via le bus. Les actionneurs réagissent à leur tour aux échanges d'informations et exécutent les ordres qui leur sont attribués : éclairage, gestion de l'énergie, chauffage… Le bus d'installation, lui, sert de support physique pour le réseau de commande.

L'ensemble bus/radio ou bus/infrarouge offre la possibilité d'actionner les récepteurs pilotes depuis le boîtier de télécommande, grâce au bus de commande. *[20]*

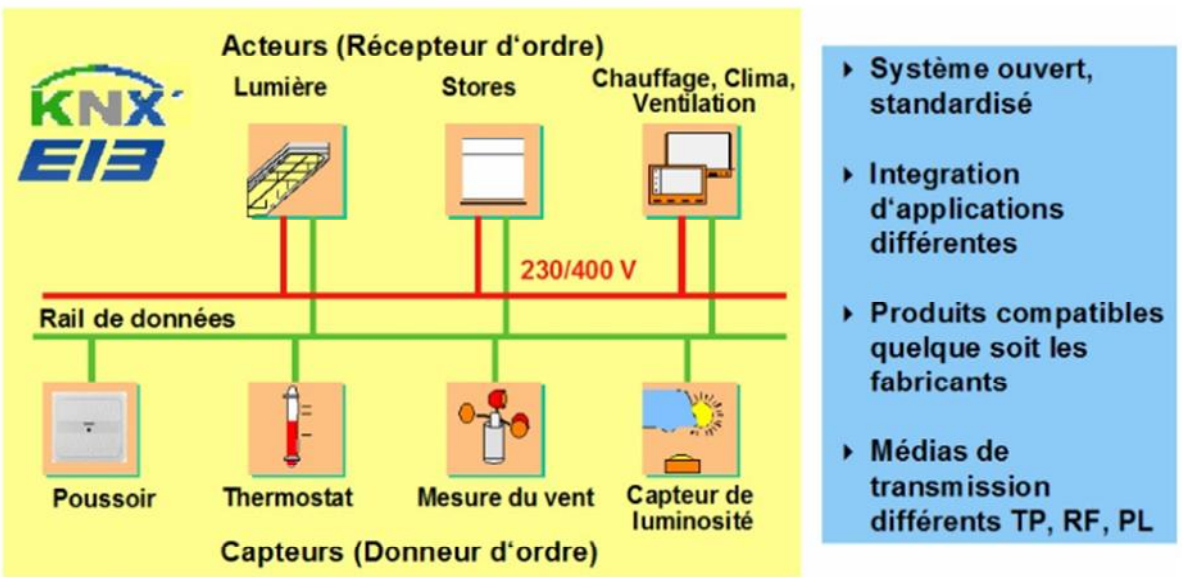


Figure XIV.14: fonctionnement de KNX

- **Caractéristiques**

Type d'interface : TP1

Récepteur-émetteur : UART TP (intégré dans l'appareil)

Vitesse de transmission : 9,6 kbit/s (fixe pour TP1)

Mode de transmission : semi Duplex

Alimentation du bus : 30 V

Consommation du bus : 5 mA (sauf QAW740)

Consommation du bus pour QAW740 :7,5 mA

Alimentation du bus et signal de communication : découplage par bobine d'arrêt (appareil avec bobine intégrée) [21]

I.11.5. CPL

- **Définition**

Le terme « courants porteurs en ligne » (CPL) réfère à une technique permet de véhiculer les informations numériques en utilisant les conducteurs électriques du secteur. Cette technique est utilisée pour les applications domotiques et pour les communications informatiques. *[22]*

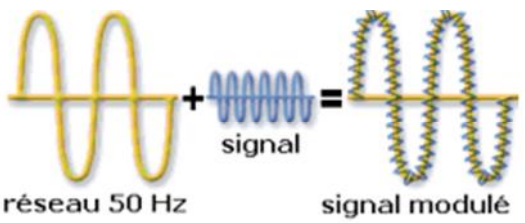


Figure XV.15 : CPL

- **Fonctionnement**

Le principe consiste à superposer sur la tension du secteur un signal (ou plusieurs signaux) de faible amplitude (quelques volts) mais dont la valeur de fréquence est caractéristique de l'information à transférer. La valeur de cette fréquence dite « porteuse » se situe dans une bande passante allant de 1,6 à 30 MHz (HF) pour le haut débit, et allant de 3 à 148KHz pour le bas débit. Concrètement les boîtiers CPL adaptent les signaux numériques issus des appareils communicants pour permettre leur transfert sur les fils électriques du secteur. *[22]*

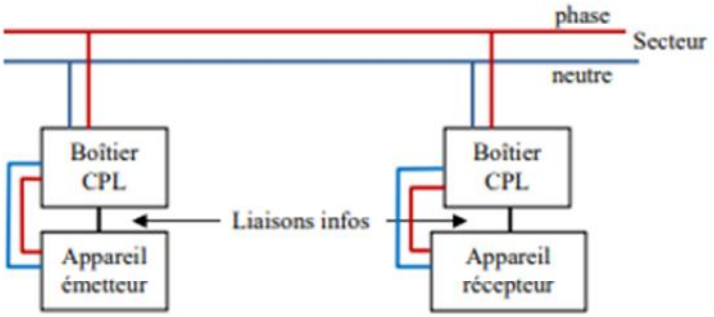


Figure XVI.16: Exemple de câblage

- **Caractéristiques** *[23]*

Interface Réseau	Ethernet RJ-45
Taux de transfert	Jusqu'à 200 Mbps
Portée	Jusqu'à 300 mètres
Sécurité	Cryptage 128 bits AES (Advanced Encryption Standard)
Compatibilité	Windows 7, Vista, XP. MacIntosh. Linux.

Figure XVII.17: Tableau récapitulatif des fonctionnalités

I.11.6. Bus CAN

- **Définition**

Le bus CAN (Controller Area Network) est né du besoin de trouver une solution de communication série dans les véhicules automobiles, qui ont tendance à intégrer de plus en plus de commandes électroniques. Jusqu'à maintenant, tous les organes de commandes des véhicules échangeaient les données par l'intermédiaire de lignes dédiées. L'augmentation du nombre d'organe embarqué a contraint les équipementiers automobiles à développer une nouvelle architecture à base de bus réseaux. *[24]*

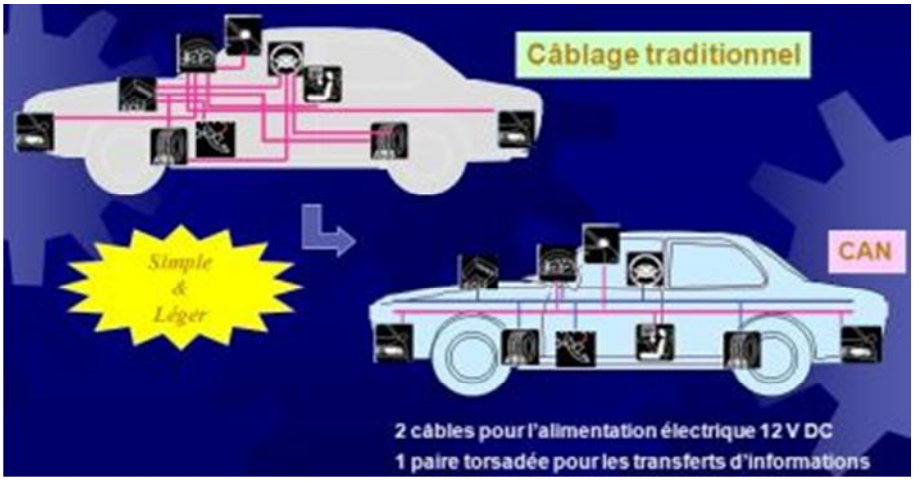


Figure XVIII.18 : bus Can

- **Fonctionnement**

Le concept de communication du bus CAN est celui de la diffusion d'information (broadcast) : chaque station connectée au réseau écoute les trames transmises par les stations émettrices. Ensuite chaque nœud décide quoi faire du message, s'il doit y répondre ou non, s'il doit agir ou non, etc...

Le protocole CAN autorise différentes stations à accéder simultanément au bus. C'est un procédé rapide et fiable d'arbitrage qui détermine la station qui émet en premier.
L'accès au bus est donc aléatoire car une station peut émettre à n'importe quel moment. Mais cet accès se fait par priorité ; cette méthode est appelée CSMA CD/AMP (Carrier Sense Multiple Acces with Collision Detection and Arbitration Message Priority). *[25]*

- **Caractéristiques**

- ✓ Domaine d'application Principalement l'automobile
- ✓ Asynchrone
- ✓ Bus Multi-Maîtres
- ✓ Bidirectionnel, Halfduplex
- ✓ Différentielle : moins sensible aux perturbations
- ✓ Hiérarchisation des messages.
- ✓ Garantie des temps de latence.
- ✓ Souplesse de configuration.
- ✓ Détections et signalisations d'erreurs.
- ✓ Débit (Dépend de la taille du réseau)
  Jusqu'à 1 Mbps (réseaux < 40 m)
  125 kbps (réseau < 500 m). *[25]*

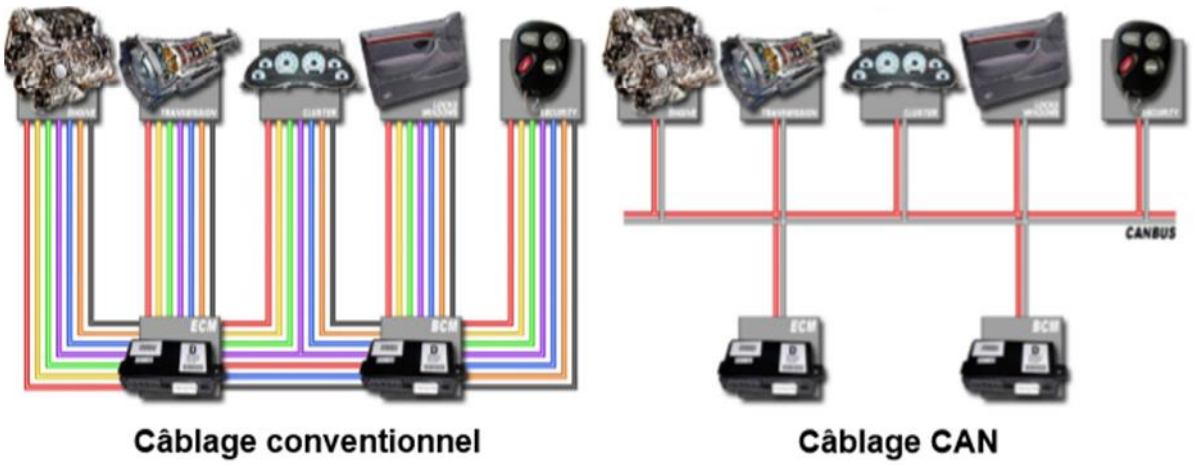


Figure XIX.19 : L'avantage du bus Can

I.12. Conclusion

Dans ce chapitre nous avons commencé par une présentation générale de la domotique ainsi ses domaines d'application et les différents types de technologies utilisées. Ensuite on a mis l'accent sur la maison intelligente et ses avantages/inconvénients. Enfin, nous avons exposé les différents protocoles de communication utilisés.

Dans le chapitre qui suit on va mettre le point sur la présentation du protocole I2C, ses caractéristiques générales et son fonctionnement.

Chapitre II :
Le protocole de bus I2C

II.1. Introduction

I2C est un bus série permettant de transmettre des informations de façon synchrone entre divers circuits connectés sur le bus. Dans ce chapitre, nous allons présenter le bus I2C, ses caractéristiques générales, les domaines d'application et le fonctionnement du protocole. Nous verrons ensuite comment les échanges se font entre le maitre et l'esclave en mode lecture et écriture.

II.2. Définition du bus I2C

Le bus I2C (Inter-Integrated Circuit) est un bus populaire développé par la société Philips dans les années 1980. Il s'agit d'une interface de communication série multi-maîtres et multi-esclaves. I ^{2} C utilise deux lignes de données bidirectionnelles à drain ouvert, Serial Data (SDA) et Serial Clock (SCL) avec des résistances pull up, comme indiqué ci-dessous. Contrairement à UART, on peut connecter et communiquer avec plusieurs périphériques en utilisant le même bus I2C.

I ^{2} C est un protocole maître-esclave. Cela signifie que les appareils connectés au bus I ^{2} C seront soit maîtres, soit esclaves. Le maître est l'appareil qui initie la communication et gère la ligne d'horloge (SCL). Les esclaves sont les appareils qui répondent au maître et ils ne peuvent pas initier une communication. *[26]*

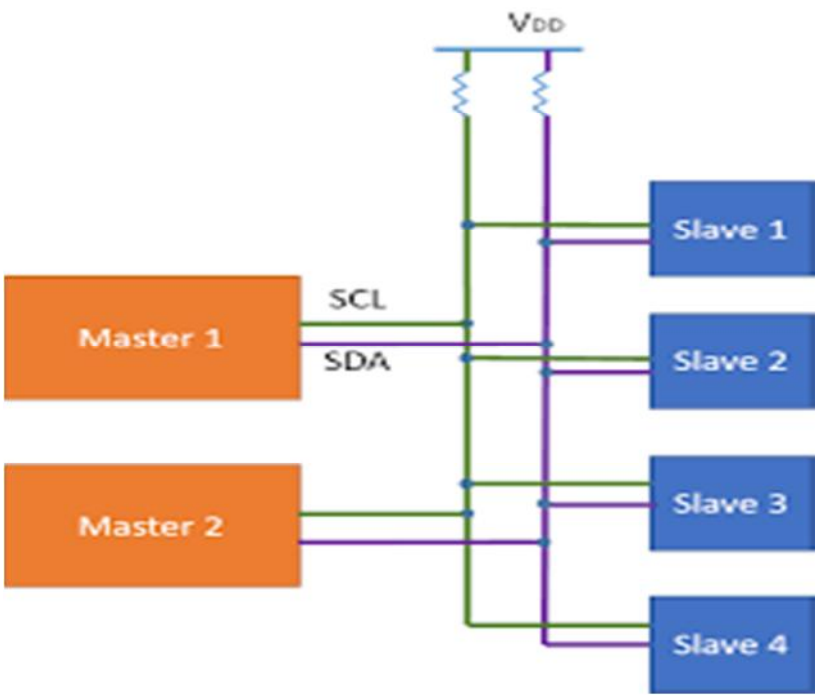


Figure II. 1: Exemple de configuration de bus I2C avec plusieurs maîtres et plusieurs esclaves

II.3. Principe du bus I2C

Entrée-Sorties de type drain ouvert (ou collecteur ouvert)

Les circuits connectés sur un bus I2C ont des sorties de type drain ouvert (ou collecteur ouvert), ce qui permet de faire un ET logique "câblé".
Deux résistances de pull-up sont bien sûr placées entre les lignes SDA et SCL et l'alimentation (VDD).

Quand le bus n'est pas utilisé, SDA et SCL sont au niveau haut (niveau de repos).
Quand une ligne (SDA ou SCL) est au repos (niveau 1), on peut la forcer à 0.
Quand une ligne (SDA ou SCL) est au niveau 0, on ne peut pas la forcer à 1. *[27]*

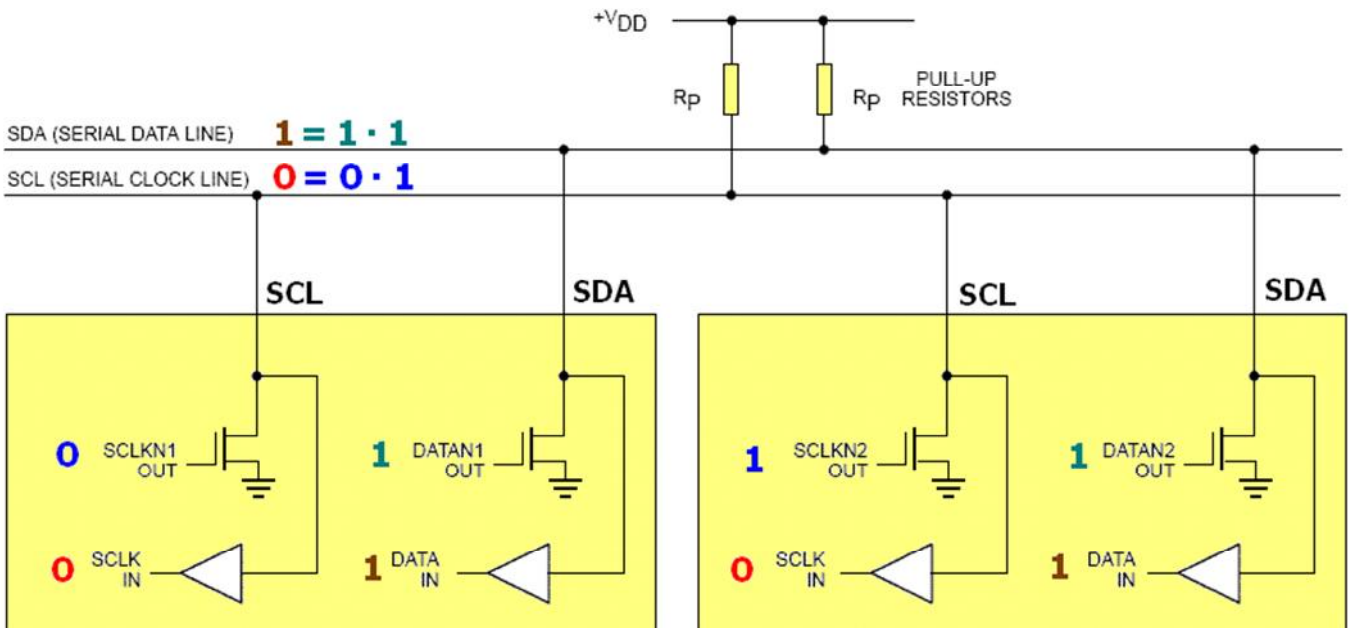


Figure II. 2: Structure d'E/S d'un module I2C

II.4. Domaines d'application I2C

La communication I2C est utilisée uniquement pour les communications à courte distance. Il est certainement fiable dans la mesure où il possède une impulsion d'horloge synchronisée pour le rendre intelligent. Ce protocole est principalement utilisé pour communiquer avec des capteurs ou d'autres appareils devant envoyer des informations à un maître. C'est très pratique lorsqu'un microcontrôleur doit communiquer avec de nombreux autres modules esclaves en utilisant un minimum de fils. *[28]*

II.5. Caractéristiques

- Synchrone
- Bidirectionnel, halfduplex

- Le support physique utilisé

Pour se connecter à un bus I²C il faut une masse, et deux fils de communication. Le premier fil, SDA (Signal Data), est utilisé pour transmettre les données. L'autre fil, SCL (Signal CLock) est utilisé pour transmettre un signal d'horloge synchrone (signal qui indique le rythme d'évolution de la ligne SDA). *[29]*

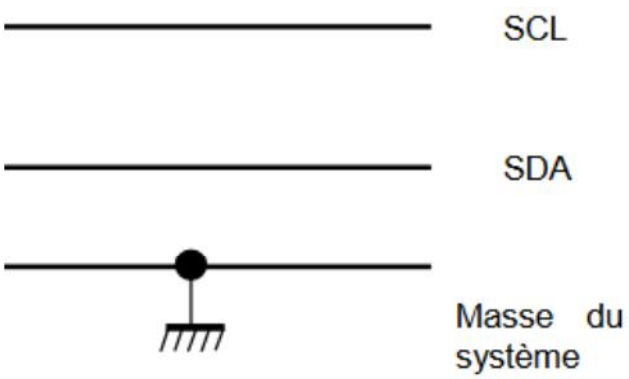


Figure II. 3: Structure d'E/S d'un module I2C

- Vitesse de transfert

Il existe trois vitesses de transfert de données pour le bus I2C : mode standard, mode rapide et mode haute vitesse.

La norme est de 100 Kbps. Le mode rapide est à 400 Kbits / s et le mode haute vitesse prend en charge des vitesses allant jusqu'à 3,4 Mbps . *[17]*

II.6. Le protocole I2C

Certains circuits sont configurés pour être des masters, d'autres des esclaves et d'autres peuvent être soit l'un soit l'autre. Pour prendre le contrôle du bus, il faut que celui-ci soit au repos (SDA et SCL à '1'). Lorsqu'un circuit prend le contrôle du bus, il en devient le maître. C'est lui qui génère le signal d'horloge et c'est lui qui initie les séquences d'échange.

II.6.1. La prise de contrôle du bus

Pour transmettre des données sur le bus I²C, il faut surveiller deux conditions particulières : la condition de départ et la condition d'arrêt.

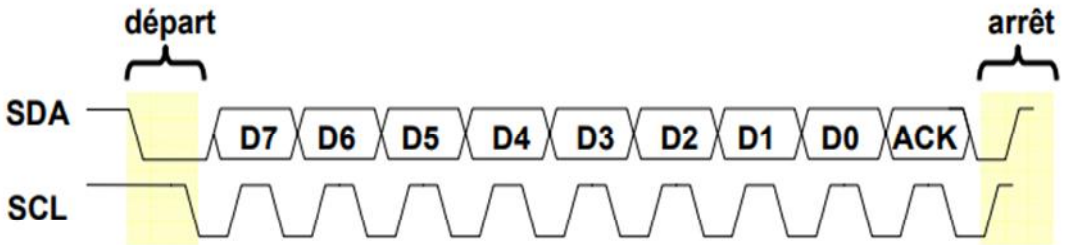


Figure II. 4: prise de contrôle de bus

Avant de tenter de prendre le contrôle du bus, un circuit doit vérifier que les lignes SDA et SCL sont au repos, c'est-à-dire à l'état haut. Si c'est le cas, le circuit indique qu'il prend le contrôle du bus en mettant la ligne SDA à 0. A partir de ce moment-là, les autres circuits savent que le bus est occupé et ils ne devraient pas tenter d'en prendre contrôle. Le circuit qui vient de prendre le contrôle du bus en devient le maître. C'est lui qui génère le signal d'horloge, quel que soit le sens du transfert. *[29]*

II.6.2. Start condition

Au début d'une séquence d'échange, le master génère une Start condition (S) avant de commencer l'échange de données. La condition Start est définie par la transition de la ligne SDA du niveau haut au niveau bas, SCL étant au niveau haut. *[30]*

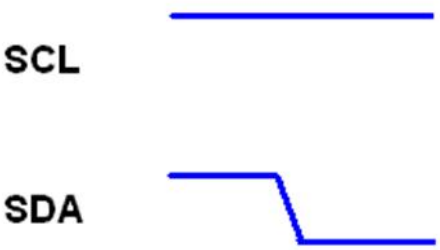


Figure II. 5: Start condition

II.6.3. Repeated Start

Une séquence de transmission peut contenir plusieurs starts conditions avant de rencontrer un Stop Condition. On parle de repeated Start condition. *[31]*

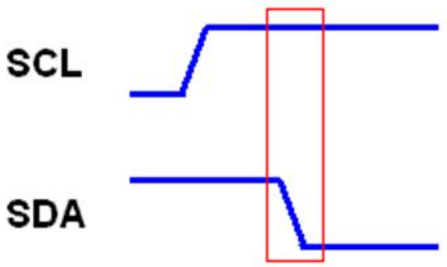


Figure II. 6: Repeated Start

II.6.4. Stop condition

A la fin d'une séquence d'échange, le master génère une stop condition (P) après laquelle le bus est de nouveau libre. La condition Stop est définie par la transition de la ligne SDA du niveau bas au niveau haut, SCL étant au niveau haut. *[32]*

Figure II. 7 : Stop condition

II.6.5. Acknowledge (ACK)

L'acknowledge est l'accusé de réception. Il est placé par le circuit qui reçoit sur la ligne SDA juste après la réception du 8ème bit. C'est l'émetteur qui le lit de la même façon qu'on lit un bit ordinaire.

Il y a deux 2 cas :

- bit Acknowledge (master to slave)

Le bit Acknowledge est généré par le maître (le maître force la ligne SDA au niveau 0).

- bit Acknowledge (slave to master)

Le bit Acknowledge est généré par un esclave (l'esclave force la ligne SDA au niveau 0). *[33]*

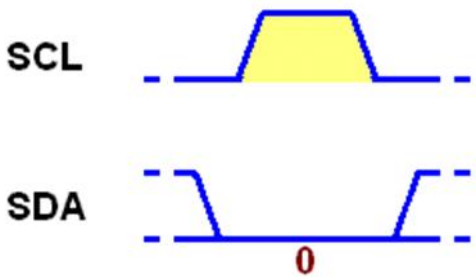


Figure II. 8: bit Acknowledge

II.1. Not acknowledge (NACK)

Le bit Not Acknowledge est toujours généré par le maître (le maître ramène la ligne SDA au niveau 1). *[32]*

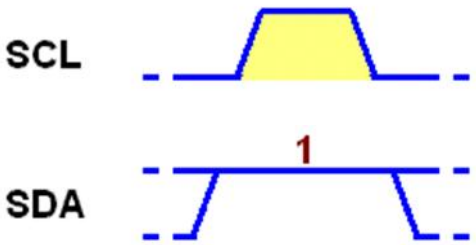


Figure II. 9: bit Not Acknowledge

II.6.7. Transmission d'un bit

Le bit à transmettre est placé sur la ligne SDA ensuite une impulsion d'horloge est envoyée sur la ligne SCL. C'est cette impulsion qui informe le slave qu'il doit lire la donnée sur SDA. *[34]*

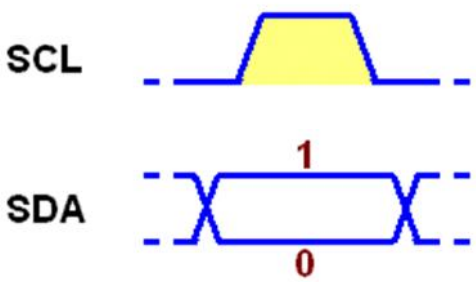


Figure II. 10: bit de donnée

II.6.8. La transmission d'un octet

La transmission d'un mot (octet) se fait bit à bit en commençant par le bit de poids fort B7 que l'émetteur (le maître) applique sur la ligne SDA. Puis il valide la donnée en appliquant pendant un instant un niveau haut sur la ligne SCL. Lorsque SCL revient au niveau bas, il recommence l'opération jusqu'à ce que l'octet complet soit transmis. *[25]*

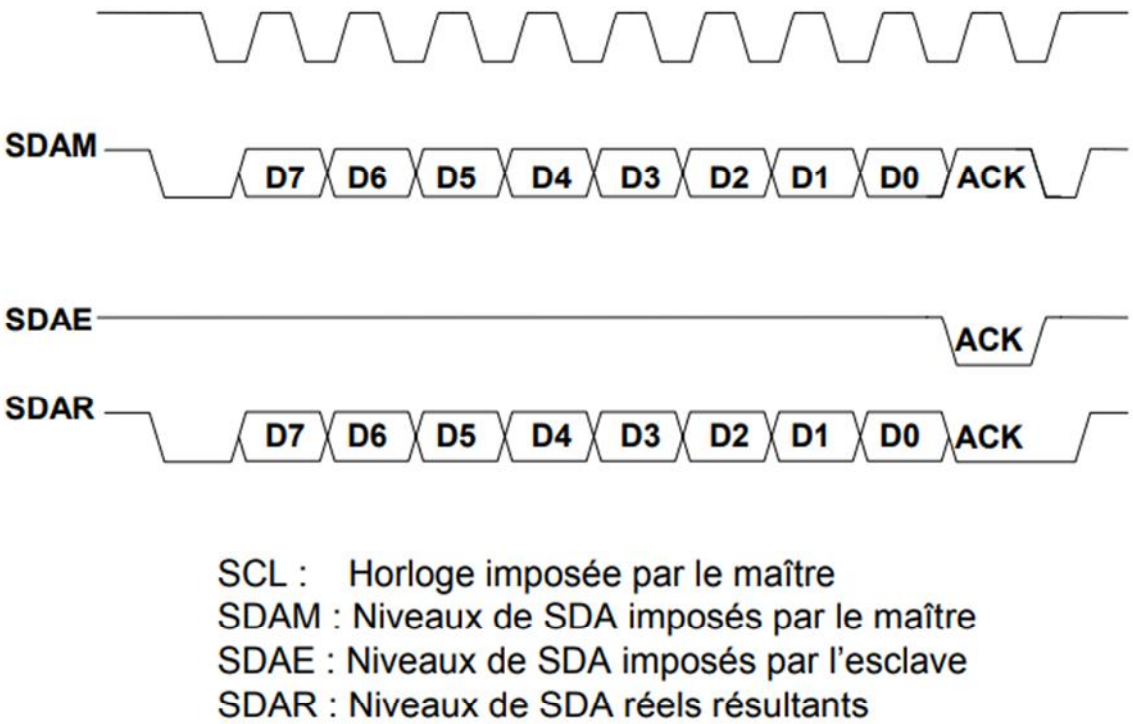


Figure II. 11: la transmission d'un octet

II.6.9. La transmission d'une adresse avec le bit R/W

Le nombre de composants qu'il est possible de connecter sur un bus I²C étant largement supérieur à deux, le maître doit pouvoir choisir quel esclave est censé recevoir les données. Dans ce but, le premier octet que le maître transmet n'est pas une donnée mais une adresse. Le format de l'octet d'adresse est un peu particulier. Comme il y a seulement 7 bits, le master envoie à la 8ème position le bit R/W pour indiquer au slave s'il désire une émission (R/W=0) ou une réception (R/W=1). *[35]*

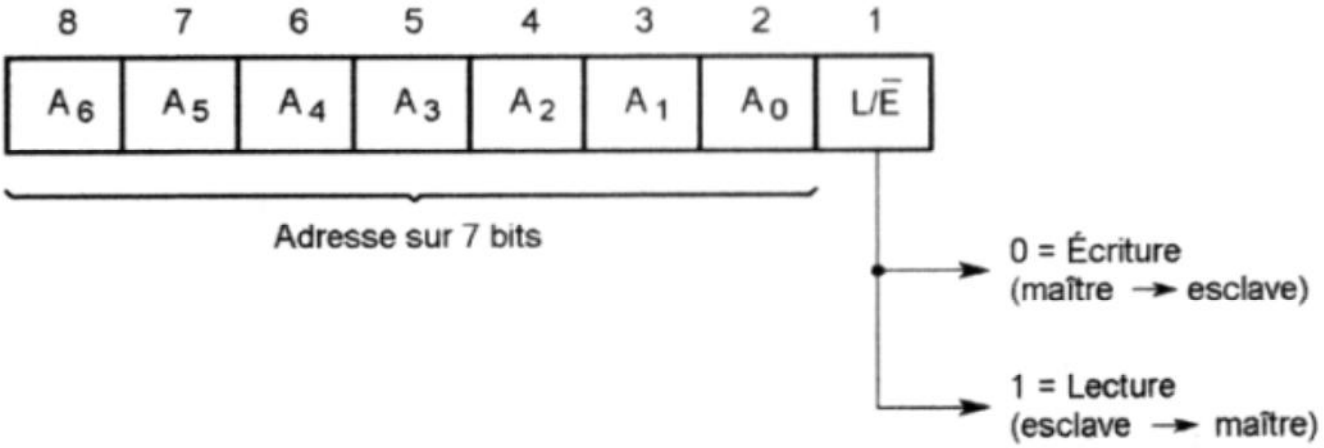


Figure II. 12: La transmission d'une adresse avec le bit R/W

Chaque circuit connecté au bus I²C possède une adresse, qui doit être unique. L'adresse associée à un composant est définie en partie par l'état de broches de sélections et d'autre part par sa fonction.

Par exemple, le circuit PCF8574, qui est un port d'entrées/sorties bidirectionnel 8 bits, décompose son adresse de la façon suivante : [0] [1] [0] [0] [A2] [A1] [A0] [R/W]. Les bits A2, A1 et A0 reflètent l'état des broches 1, 2 et 3 du circuit. Cela permet de placer 8 circuits PCF8574 sur le bus I²C. Lors de la conception d'un système, il faut donc veiller à l'unicité des adresses attribuées aux différents composants.

Une fois l'adresse envoyée sur le bus, l'esclave concerné doit répondre en plaçant le bit ACK à 0. Si le bit ACK vaut 1, le maître comprend qu'il y a une erreur de sélection et il génère la condition arrêt. En revanche, si le bit ACK vaut 0, le maître peut continuer les opérations. *[35]*

L'octet d'adressage est codé soit sur 7 bits soit sur 10 bits :

- ## Adressage sur 7 bits

L'octet d'adressage peut être scindé en deux parties :

Les sept premiers bits correspondent à l'adresse proprement dite,
Le dernier bit est nommé R/W :

On peut également considérer que l'adresse est codée sur les 8 bits, chaque esclave a alors deux adresses, l'adresse paire qui sert à lui envoyer des données, l'adresse impaire pour lui demander d'en envoyer.

Plusieurs adresses sont réservées :

« 00000000 » : utilisée pour adresser tous les esclaves (« broadcast »),

« 0000001X » : utilisée pour accéder aux composants CBUS (ancêtre de l'I²C),

« 0000010X » : réservée pour d'autres systèmes de bus,

« 0000011X » : réservée pour des utilisations futures,

« 00001XXX » : pour les composants haute-vitesse,

« 11111XXX » : réservée pour des utilisations futures,

« 11110yzX » : permet de préciser une adresse sur 10 bits. *[36]*

➢ Adressage sur 10 bits

Dans le cas d'un adressage sur 10 bits, il faut utiliser deux octets.

Le premier est l'octet « **11110yz0** », les bits « **yz** » sont les 2 bits de poids forts de l'adresse, le bit R/W est toujours placé à 0.

Le deuxième octet est utilisé pour les 8 bits de poids faibles de l'adresse, il n'y a pas de bit R/W.

À la suite de l'émission du premier octet, plusieurs esclaves parmi ceux ayant une adresse sur 10 bits peuvent répondre par un ACK (ceux qui ont les mêmes 2 bits de poids fort). Peu importe, à l'issue du 2^{éme} octet, seul l'esclave auquel on s'est adressé répondra.

Le bit R/W étant toujours placé à 0, pour demander à un esclave d'écrire, à la suite de l'émission des 2 octets précédents, il faut renvoyer une condition de RESTART suivie de l'octet « 11110yz1 » (avec le bit R/W à 1), pour que l'esclave sache qu'il s'agit d'une commande de lecture. *[36]*

II.7. LECTURE / ECRITURE DE DONNEES

II.7.1. Ecriture d'une donnée

START + adresse

le maitre démarre par un bit START, la ligne SDA transite de l'état logique « 1 » à l'état logique « 0 », et ensuite le maitre envoi une adresse .
Le 8ème bit est bien à l'état logique « 0 » ce qui veut dire que le maitre écrit sur le bus et s'adresse à l'esclave qui lui a une adresse (l'adresse doit être unique par esclave). l'esclave qui à cette adresse va se reconnaitre et donc envoyer un bit d'acquittement en signalant au maitre qu'il a bien reçu l'adresse en mode écriture.

1ère donnée

Une fois le maitre ayant reçu le bit d'acquittement (le 9ème bit pour être précis) qui émane de l'esclave, le maitre va écrire la 1ére donnée qu'il va transférer à l'esclave.
Cette dernière va donc être reçue par l'esclave et ce dernier va renvoyer un bit d'acquittement en signalant qu'il a bien reçu cette 1ére donnée.

2ème donnée

Le maitre ayant reçu le bit d'acquittement qui émane de l'esclave va renvoyer une 2ème donnée, et ainsi de suite l'esclave va renvoyer un bit d'acquittement en signalant qu'il a bien reçu cette 2ème donnée.

STOP

Le maitre a bien reçu le bit d'acquittement de l'esclave mais il n'enverrait rien d'autre et la ligne passera à l'état logique « 1 » en envoyant un bit de STOP pour dire à l'esclave qu'il a terminé la transmission de données. Chaque fois qu'un bit d'acquittement est généré derrière par l'esclave, c'est le maitre qui doit envoyer les données. *[37]*

Figure II. 13: Ecriture du maitre

En signal électrique :

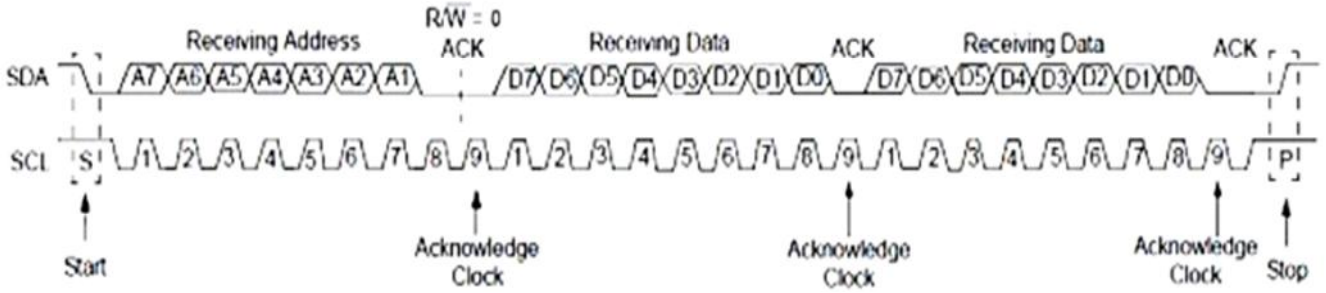


Figure II. 14: signal électrique de l'écriture

II.7.2. Lecture d'une donnée

START + adresse

En mode lecture, le principe reste le même on retrouve le bit de START ainsi que l'envoi de l'adresse en mode lecture puisque le 8ème bit est à l'état logique « 1 » (couleur bleu). Le maitre réclame donc une lecture sur le bus à l'esclave afin que ce dernier puisse lui envoyer les données.

1ère donnée

Lorsque l'esclave à bien reçu son adresse avec le 8ème bit à l'état logique « 1 », il sait que c'est le maitre qui s'adresse à lui en mode lecture, et doit générer un bit d'acquittement afin de donner une réponse au maitre du type : « oui c''est bien moi !! Je t'envoie les données comme demandé !! » Ainsi la 1ére donnée est envoyé par l'esclave au maitre.

2ème donnée

Lorsque le maitre aura reçu la 1ére donnée qui émane de l'esclave, c'est le maitre à son tour qui va répondre à l'esclave par un bit d'acquittement (couleur bleu) en signalant à l'esclave qu'il a bien reçu la 1er donnée mais qu'il attend maintenant la 2ème donnée !!
Ainsi, l'esclave va renvoyer la 2ème donnée au maitre.

STOP

Lorsque le maitre aura reçu la 2ème donné, il n'enverra pas de bit d'acquittement, car et ce ceci est IMPORTANT, chaque fois que le maitre envoi un bit d'acquittement (couleur bleu) c'est qu'il réclame un retour de l'esclave et l'esclave doit répondre en transmettant ses données.
Afin d'arrêter cette transmission, le maitre va émettre un bit de STOP pour dire à l'esclave que la transmission est terminé et qu'il n'attend plus rien de l'esclave.
Ce qu'il faut comprendre d'une manière général c'est qu'à chaque fois qu'un bit d'acquittement est généré par le maitre, l'esclave doit envoyer les données. *[37]*

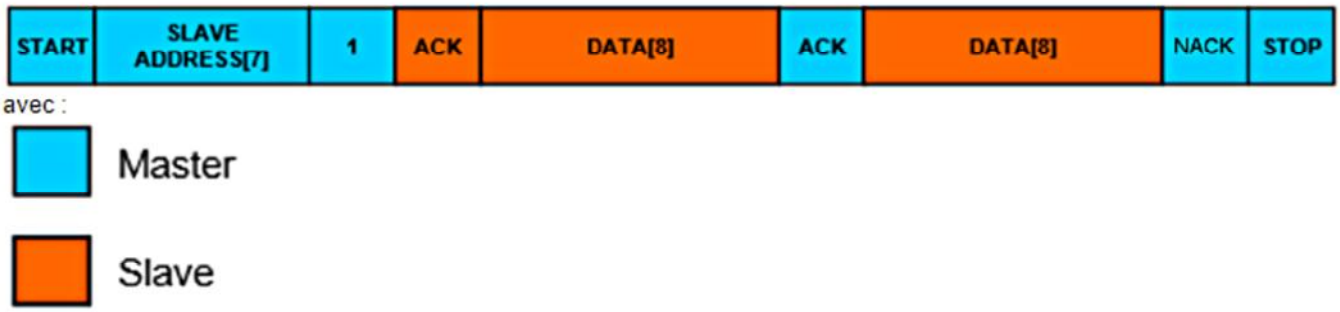


Figure II. 15: Lecture d'une donnée

En signal électrique :

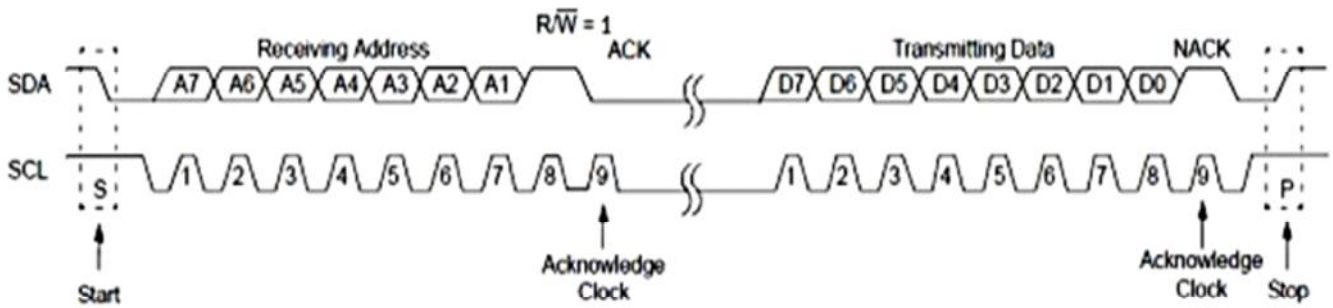


Figure II. 16: signal électrique de la lecture

Dans le cas où le maître écrit (envoie une commande) puis lit ce que l'esclave envoi :

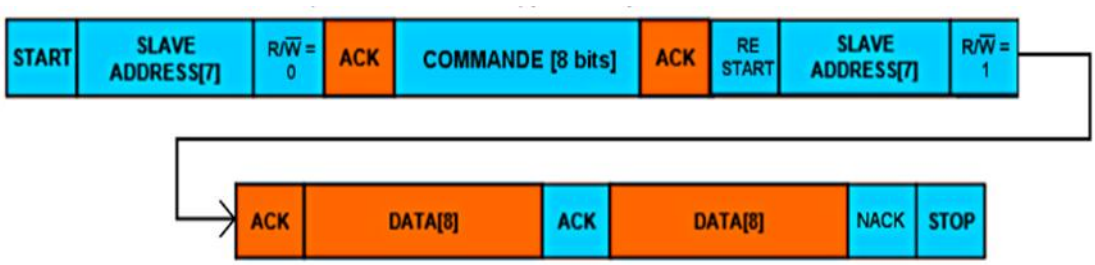


Figure II. 17: Ecriture et l'écriture du maitre

En signal électrique :

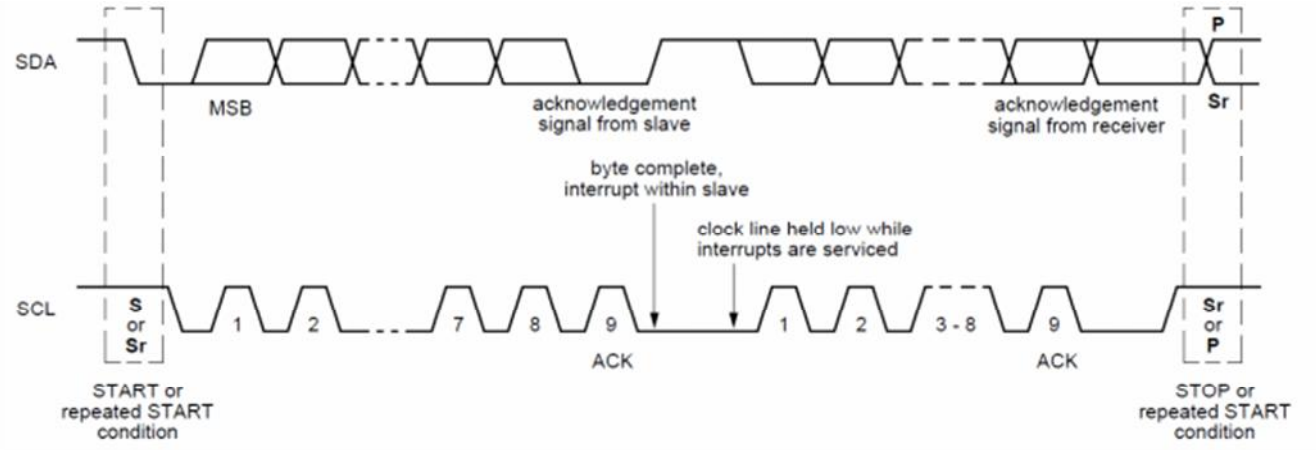


Figure II. 18: signal électrique de la lecture et l'écriture

II.8. Avantages et inconvénients du protocole I2C

II.8.1. Avantages du protocole i2c

• Utilisation de 2 lignes seulement pour transmettre les informations.

• Les données peuvent circuler dans les deux sens sur le bus (half-duplex).

• Le bus est multi-maître.

• Chaque équipement relié au bus dispose d'une adresse codée sur 7 bits, soit une possibilité de connecter 128 équipements.

• Les composants programmables récents comprennent des fonctions permettant de gérer le protocole I2C. *[28]*

II.8.2. Inconvénients du protocole i2c

• Le principe de dialogue série est complexe et difficile à maitriser.

• Le bus limite le débit et ne permet pas des applications dans lesquelles la vitesse de transmission est importante. (100Kbits maximum, 400Kbits sur des équipements récents)

• En plus d'être limité par les adresses, le nombre d'équipement maximum est limité par la charge capacitive du bus maximale : 400pF. *[28]*

II.9. Conclusion

L'I2C est un protocole de communication rapide et facile à mettre en œuvre. Il permet de créer des systèmes supportant plusieurs maîtres et esclaves. Ce protocole prévoit la gestion automatique des conflits entre composants. Ce sont les composants eux-mêmes qui s'en occupent.

Notre système repose sur cette norme. Etant en mode mono-maître, nous n'avons pas à nous préoccuper de la gestion des conflits.

Chapitre III :
Conception

III.1. Introduction

Notre mémoire a pour objectif la mise en place du protocole de communication I2C entre une centrale de domotique et ses équipements pour permettre à cette dernière de détecter automatiquement les équipements branchés et leurs types (capteur ou actionneur). Ce protocole permettra à la centrale de reconnaitre tous les capteurs et actionneurs intelligents se trouvant dans son espace et faciliter leur gestion. Cette solution facilitera l'intégration d'un ensemble de capteurs dans une smart home. A titre d'exemple, nous intégrerons nos modules dans une smart home réalisé sous Android Things.

III.2. Schéma synoptique global du système à réaliser

La figure ci-dessous donne le schéma synoptique global d'une centrale de domotique contenant plusieurs capteurs et actionneurs.

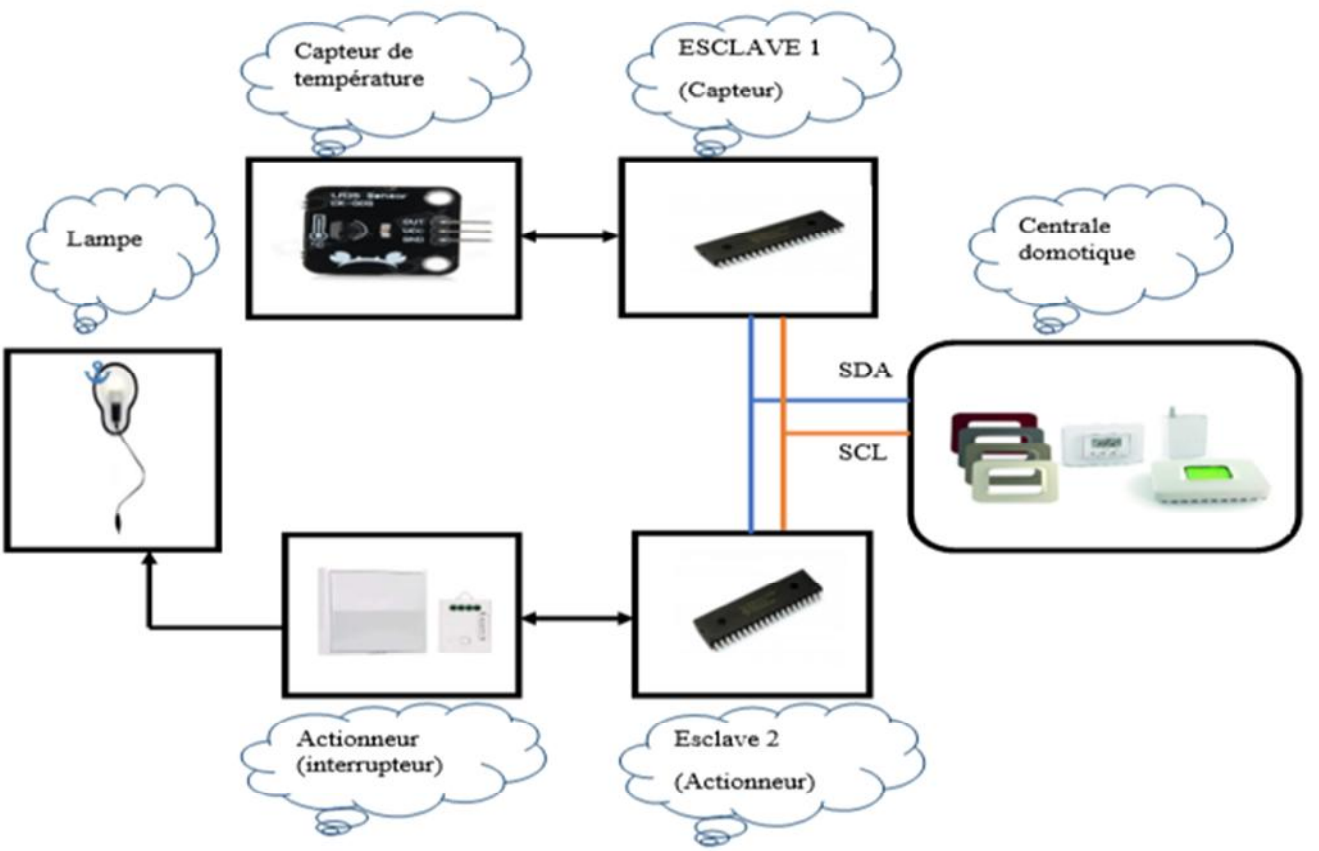


Figure III.1: **Le schéma synoptique global de système à réaliser**

Ce système s'articule autour de deux parties, la première partie est le maitre I2C. C'est la centrale de domotique à laquelle sont connectés les différents équipements. Elle aura pour rôle de scanner le bus I2C à la recherche d'équipements connectés. Ensuite elle doit

reconnaitre le type d'équipement (capteur ou actionneur) et son adresse de gestion. Une fois l'équipement reconnu et l'adresse déterminée, elle doit pouvoir le gérer. Si, par exemple l'équipement est un capteur de température, elle doit prélever la température à la demande de l'utilisateur ou cycliquement. Si c'est une porte, elle doit pouvoir ouvrir ou fermer cette dernière à la demande de l'utilisateur. Ce procédé pourra être généralisé plus tard pour prendre en charge tous les équipements de la domotique (alarme, Gaz, Chauffage, ...etc.).
La deuxième partie est constituée des différents esclaves. Chacun est réalisé au tour d'un capteur ou d'un actionneur. A titre d'exemple, nous allons réaliser un capteur de température et un actionneur pour allumage d'une lampe. L'esclave doit répondre à la requête de recherche émise par le maitre et s'identifier en donnant son type, et sa valeur

Les différents composants seront réalisés à l'aide du microcontrôleur PIC 16F877 disposant d'implémentation matérielle pour le protocole I2C.

III.3. Conception Matérielle

III.3.1. Présentation du Microcontrôleur (Pic16F877A)

Le PIC (Programmable Interface Contrôler) est un microcontrôleur développé par Microchip permettant de développer des systèmes embarqués peu chers et ne nécessitant pas l'ajout de composants externes. Les Pics sont des composant dits RISC (Reduced Instructions Set Computer), ou encore composant à jeu d'instruction réduit. Le microcontrôleur se trouve, dans plusieurs appareils telle que : les téléphones portables, machines à laver, télévisions vidéos .... Etc. *[38]*

- Différentes familles de PIC

Les Pics sont subdivisés en 3 grandes familles : *[39]*

- La famille **Base-Line**, qui utilise des mots d'instructions de 12 bits.
- La famille **Mid-Range**, qui utilise des mots de 14 bits (et dont font partie les 16F876 et 16F877)
- La famille **High-End**, qui utilise des mots de 16 bits.

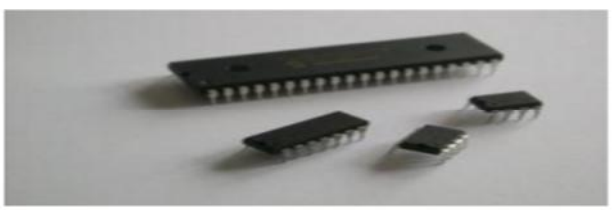


Figure III.2: Les différentes familles de PIC

➢ Identification des PIC *[40]*

Un PIC est généralement identifié par une référence de la forme suivante : xx(L)XXyy-zz

- Xx : famille du composant, actuellement « 12, 14, 16, 17 et 18 ».
  - ✓ L : tolérance plus importante de la plage de tension.
- XX : type de mémoire programme.
  C: EPROM ou EEPROM.
  CR: PROM
  F : Flash
  Yy : indique le modèle du PIC (son identité).
  Zz : vitesse maximale du quartz de pilotage. [11]

Concernant le PIC 16F877-A

- 16 : indique la famille du PIC (Mid-Rang).
- F : indique le type de la mémoire utilisée (Flash).
- 877 : identité du PIC.
- A : fréquence maximale d'horloge qui est de 20MHZ.

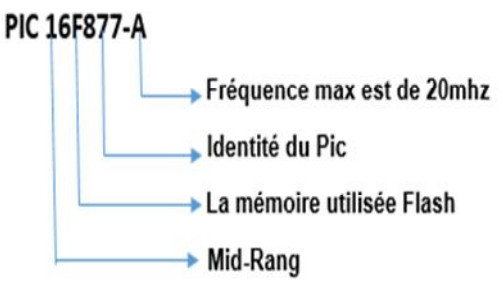


Figure III.3: Exemple de l'identification PIC16F877-A

➢ **Propriétés** : [41]

- Il a 40 pins.
  Il supporte de nombreux protocoles de communication tels que :

  Protocole série.

  Protocole parallèle.

  Protocole I2C.

- 2entrées de captures et de comparaison.
- Un convertisseur Analogique Numérique 10 bits avec 8 entrées multiplexées.
- Une interface de communication série asynchrone et synchrone (USART/SCI).

- Une tension d'alimentation entre 2 et 5.5 V.
- Vous pouvez recevoir des données de RX et transmettre des données de TX.
- Il prend en charge le protocole I2C et SPI.
- Un oscillateur à cristal allant de 4 MHz à 40 MHz.
- Trois TIMERS avec leurs Prescalers, TMR0, TMR1, TMR2.
- Deux modules de comparaison et Capture CCP1 et CCP2.
- 13 sources d'interruption.
- Fonctionnement en mode sleep pour réduction de la consommation.
- Programmation par mode ICSP (In Circuit Serial Programming) 12V ou 5V

➢ Brochage du PIC16F877 : *[39]*

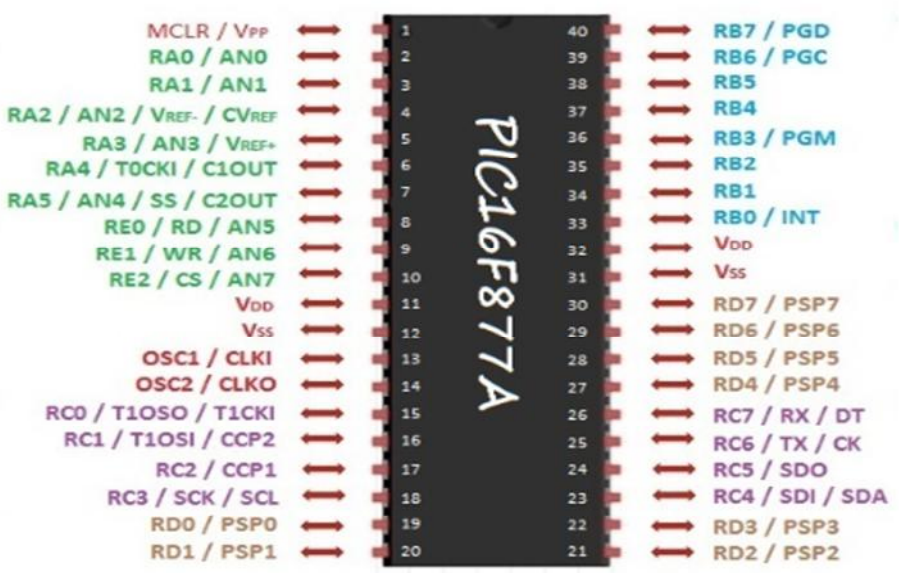


Figure III.4: Brochage du PIC16F877

On peut distinguer sur ce schéma :

L'alimentation : VDD (+5V) et Vss (0V).
Les bornes du quartz (oscillateur a quartz) : OSC1 et OSC2.
L'entrée RESET (MCLR: Master clear).
Les différents ports d'Entrées/Sorties : RAx, RBx, RCx, RDx, Rex.

- Le bus I2C et le PIC 16F877A

Le PIC16F877A possède des entrées/sorties spécialement prévues pour la communication I2C. Ce sont les broches nommées RC3 (SCL) et RC4 (SDA). Elles sont directement reliées à un contrôleur I2C.

Ce contrôleur prévoit la gestion de tout le protocole I2C (Prise de parole, collision, ligne occupée...). Il permet une gestion facile de la liaison. *[27]*

III.3.2. Présentation du capteur LM35

Le capteur LM35 est un capteur de température où la tension de sortie est linéairement proportionnelle à la température en Celsius centigrade. Ce capteur ne nécessite pas de calibrage externe pour fournir une précision de #177 ; °C sur une gamme de température de -55°C à +150°C. Son coefficient est de 10mV/°C et dans notre cas le capteur est alimenté par 0-5V, on ne peut mesurer par conséquent que des températures positives. *[39]*

- Brochage du LM35

Le capteur LM35 est commercialisé dans boiter 3 broches classiques, comme illustré dans la figure ci-dessous

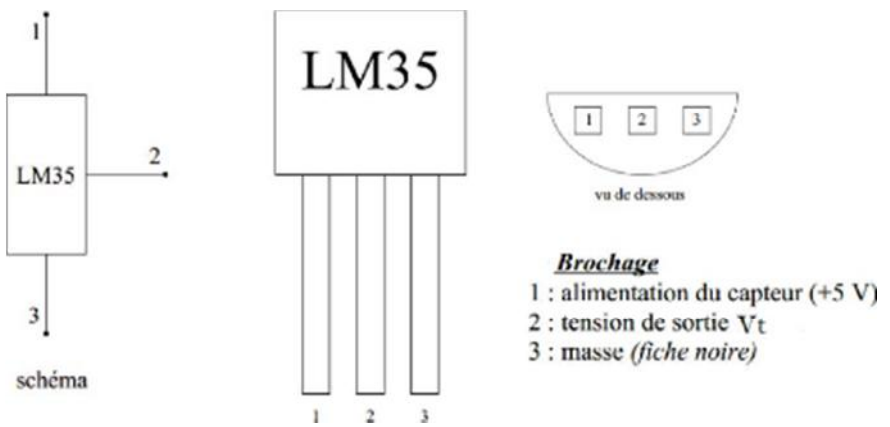


Figure III.5: brochage du LM35

- Caractéristiques *[42]*

Une fois alimenté, il va délivrer une tension analogique proportionnelle à la température.

Tension d'alimentation : 4V à 30V

Etendu de mesure : 0°C à 100°C

Précision : ±0,75 °C (typique)

Echelle : 10mV/°C

Calibration : 0mV à 0°C, 1000mV à 100°C

III.3.3. Conception du capteur I2C

La figure III.6 donne le schéma électrique du capteur I2C. Il est conçu à base d'un capteur LM35 relié à la broche analogique A5 du pic16f877A pour recevoir la valeur de température.

Pour spécifier l'adresse de l'esclave (capteur), on a utilisé les jumpers (2, 3, 4) qui sont reliés aux GPIO (3, 34, 35). Pour la fréquence d'horloge, on a utilisé un oscillateur qui est composé d'un quartz (X2) relié aux deux broches (osc1 [13] osc2 [14]) du pic et deux capacités c1, c2 reliées à la masse. Concernant le connecteur 1, il est utilisé pour l'alimentation et la liaison filaire avec le maitre I2C (SCL [18], SDA [23]).

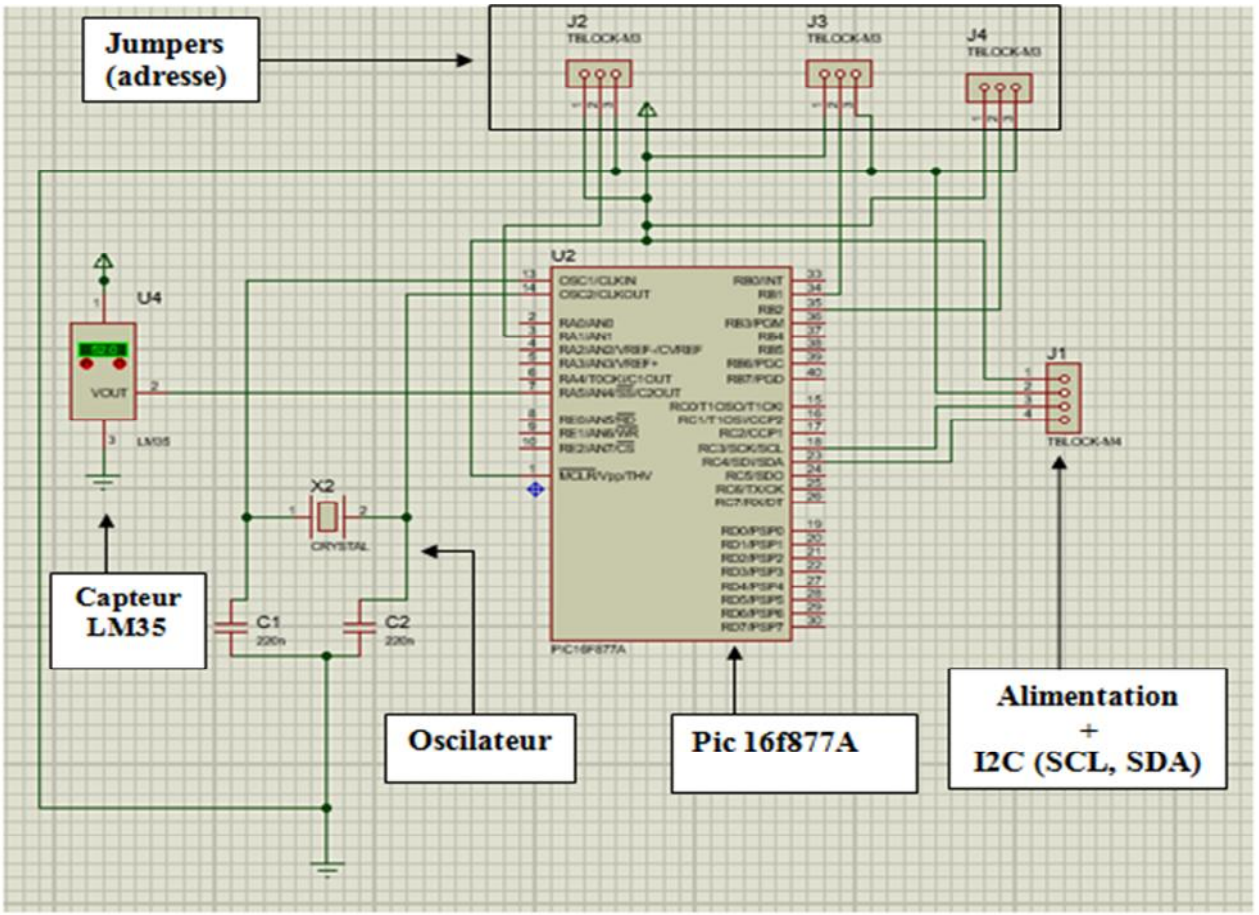


Figure III.6: Schéma électrique du capteur I2C

III.3.4. Conception de l'actionneur I2C

La figure III.7 donne le schéma d'un actionneur pour lampe. La broche 33 du pic est reliée à l'entrée A de l'optocoupleur qui sert à activer le transistor de commande du relais tout en isolant la charge et le signal venant du pic.

Le transistor MOSFET est utilisé comme un interrupteur. Une tension de 5V DC active le Gâte (gâchette) et le courant passe de drain à source. La bobine du relais est liée en séries avec le drain et le source +5V.

Quand il y a un signal sorti de 5V du pic (pin 33), le courant passe dans la diode de l'optocoupleur et le phototransistor est activé.

La diode est liée avec la bobine du relais pour protéger le transistor de haute tension qui va être générer le moment ou la tension sur la gâchette devienne zéro.

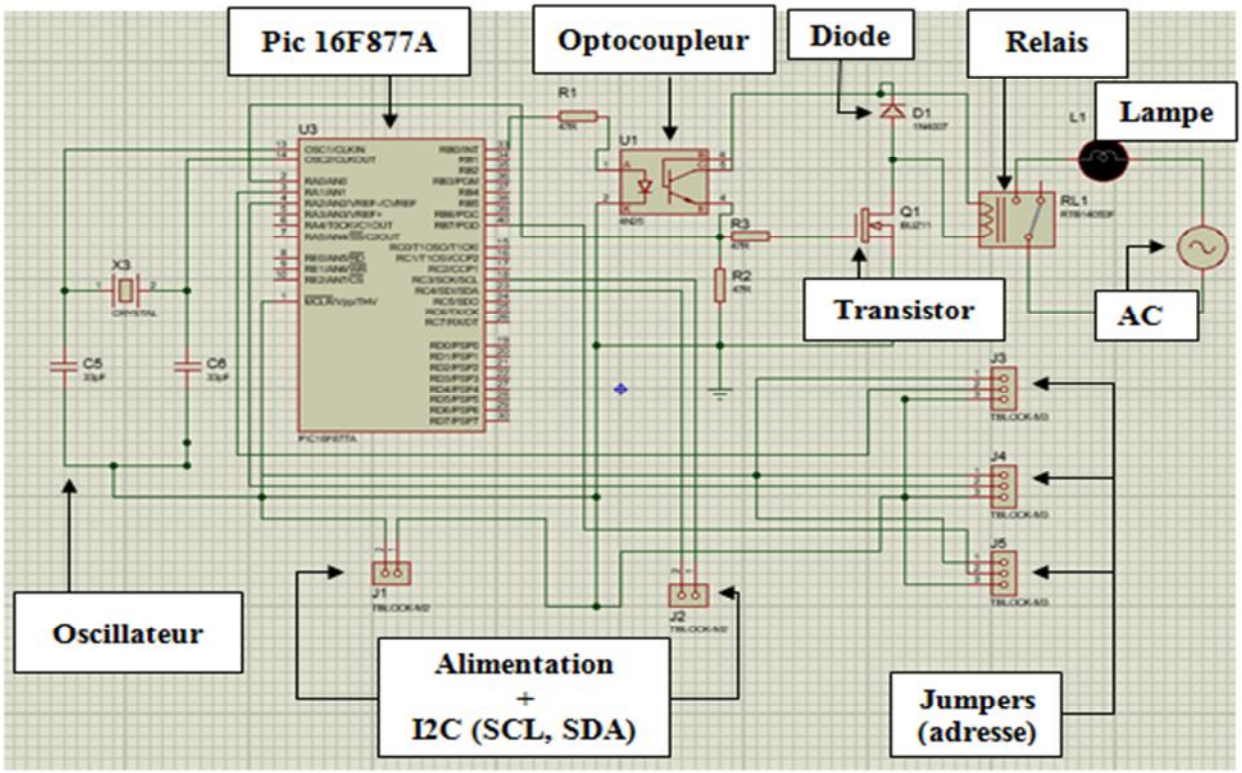


Figure III.7: Schéma électrique de l'actionneur I2C

III.3.5. Conception d'un module Maitre I2C

Pour la démonstration des fonctionnalités de nos capteurs et actionneurs, nous avons conçu un maitre I2C dans lequel nous avons embarqué une application de démonstration de l'utilisation des modules. La figure III.8 donne le schéma électrique du maitre qui commande les esclaves par les deux lignes SCL (18) et SDA (23) du pic relié aux deux résistances pull up. Pour l'affichage, on a utilisé un lcd 2x16 relié à une résistance variable pour l'ajustement de la lumière.

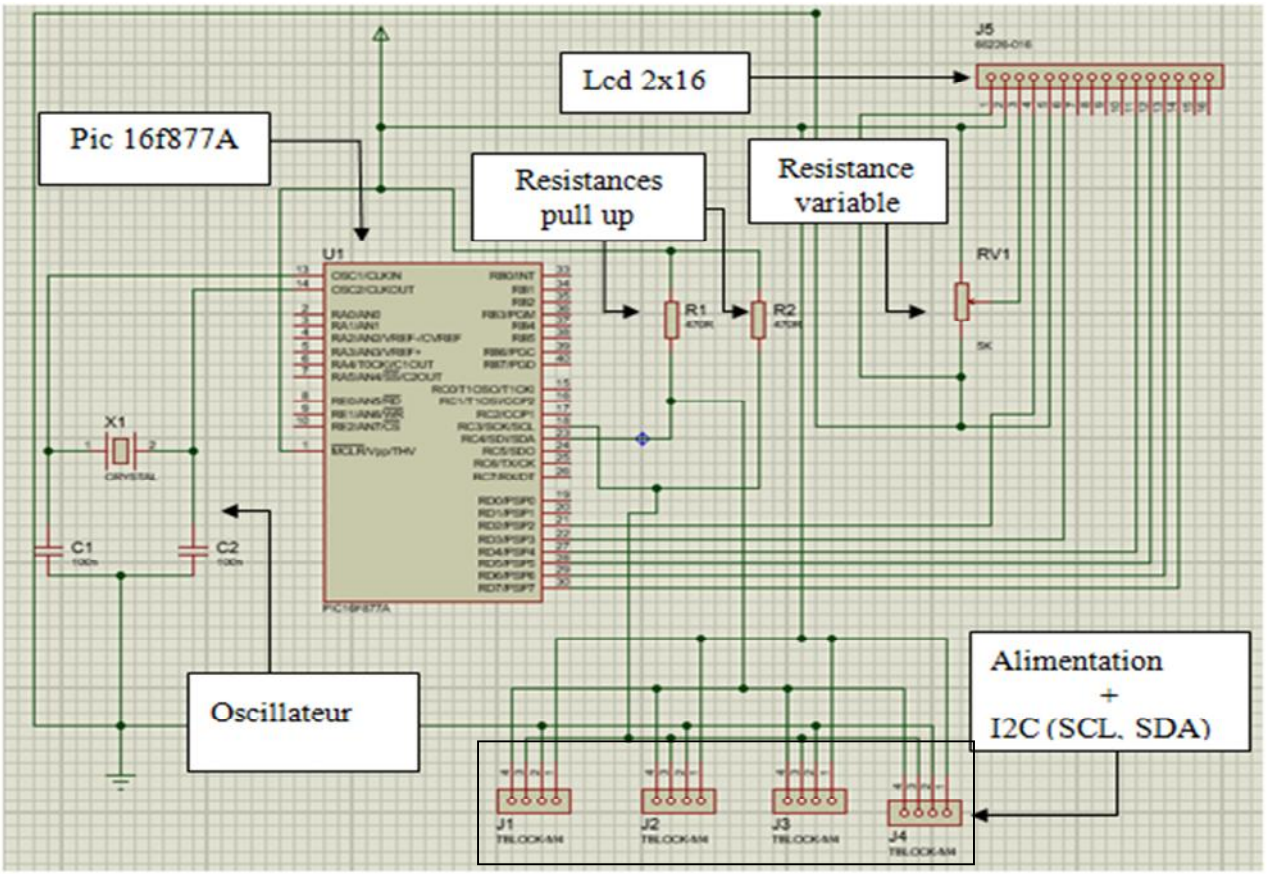


Figure III.8: Schéma électrique du maitre I2C

III.4. Conception logicielle

III.4.1. Schéma modulaire du programme

Chaque équipement utilise des drivers que nous avons implémentés au niveau registres (le niveau le plus bas). Chaque driver est implémenté de façon modulaire. Nous avons donc créé trois modules définis chacun par un fichier d'entête (.h) et un fichier implémentation (.c). Cette implémentation facilité la lisibilité et la réutilisabilité des drivers. Les trois modules sont:

- Le driver du LCD: définit par les deux fichiers lcd.h et lcd.c contenant les fonctions de gestion d'un LCD 2x16
- Le Driver du convertisseur analogique numérique: définit par les deux fichiers adc.h et adc.c contenant les fonctions nécessaires à la conversion analogique numérique
- Le Driver I2C définit par les deux fichiers i2c.h et i2c.c contenant les fonctions nécessaires à l'établissement du protocole I2C

La figure III-9 donne le diagramme d'interaction entre les différents programmes réalisés avec les drivers

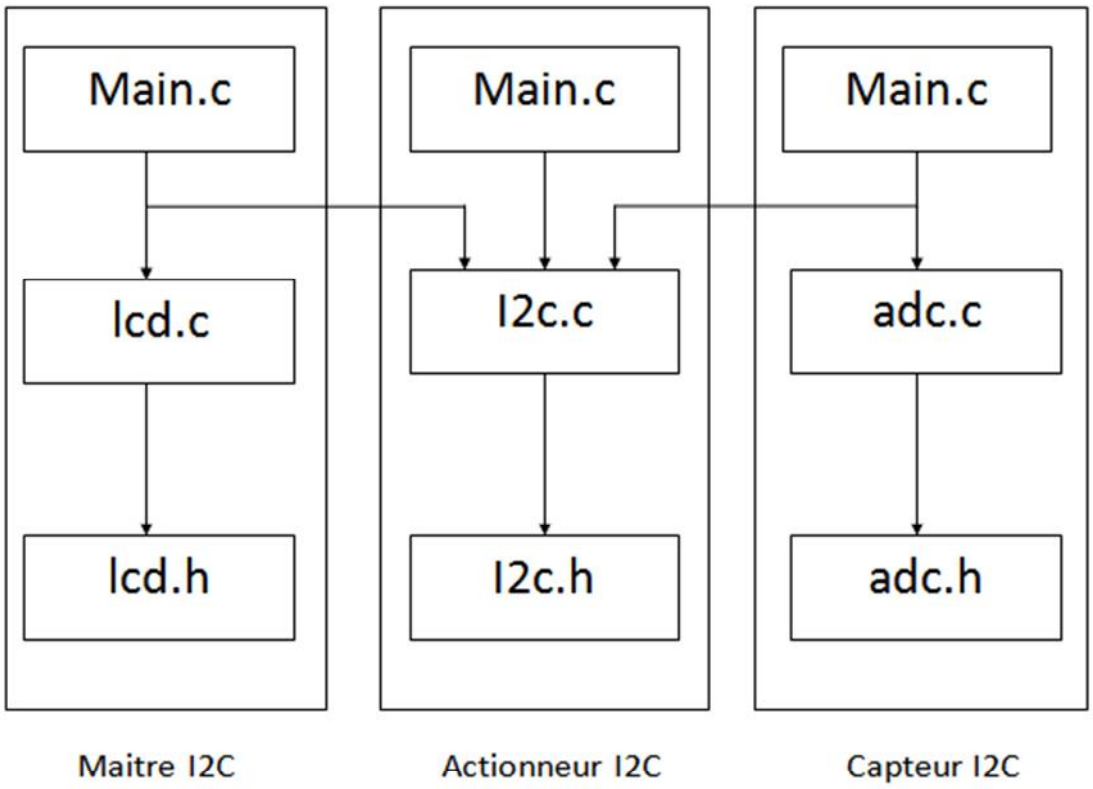
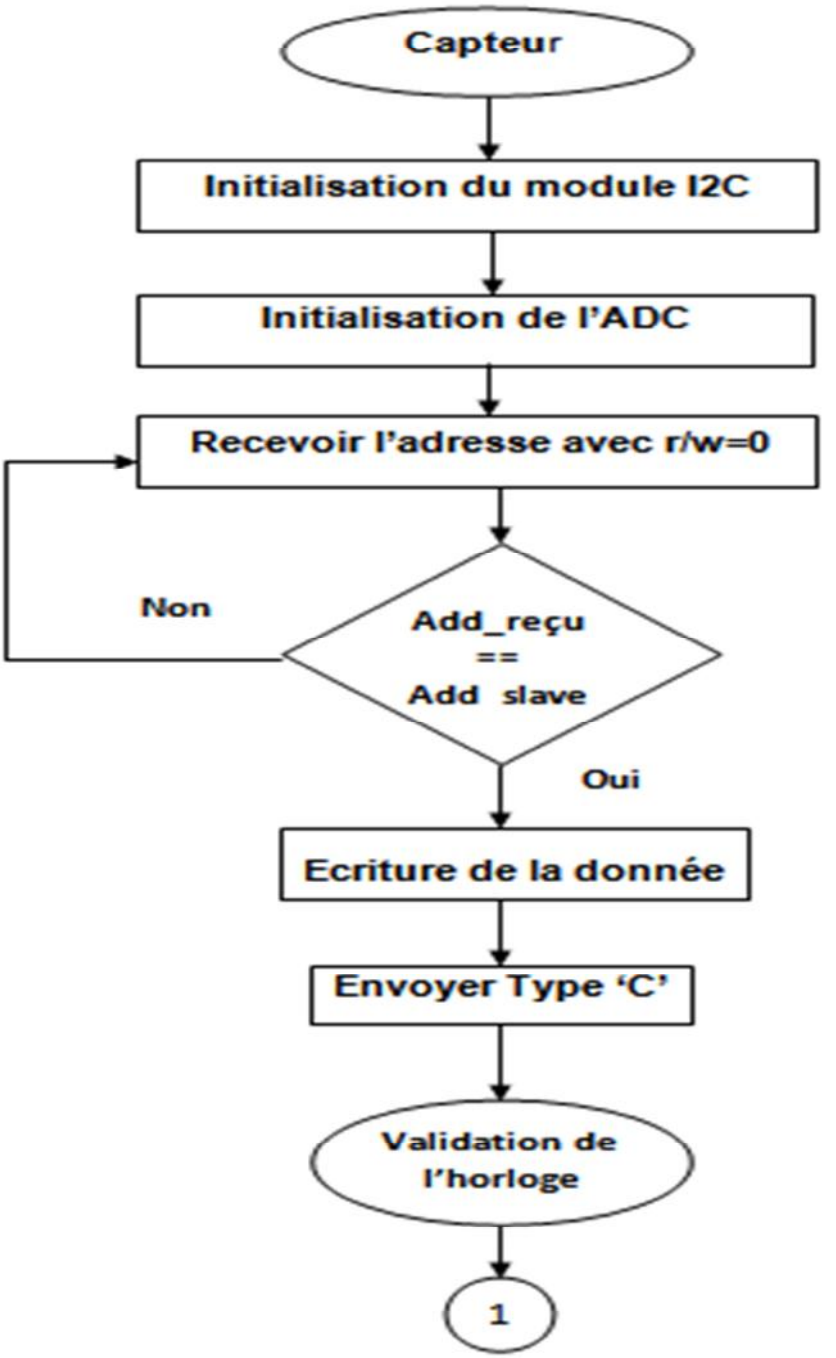


Figure III.9: schéma modulaire du programme

III.4.2. Implémentation du capteur I2C

La figure III.10 donne l'organigramme du capteur i2c qui décrit son fonctionnement global. En premier temps, nous allons initialiser le module i2c (capteur) et son convertisseur l'ADC, ensuite dès que le capteur reçoit une adresse du maitre i2c, il va faire une comparaison avec sa propre adresse, (add reçue === add slave), si c'est l'adresse d'un autre esclave, il ne réagit pas.

Si elle correspond à son adresse une interruption sera déclenchée et le capteur va faire une écriture, il envoie son type (capteur ou actionneur) et la valeur de température en deux parties : Partie entière (valeur) et partie fractionnelle (valeur2).


Capteur
Initialisation du module I2C
Initialisation de l'ADC
Recevoir l'adresse avec r/w=0
Non
Add_reçu
==
Add slave
Oui
Ecriture de la donnée
Envoyer Type 'C'
Validation de
l'horloge
1

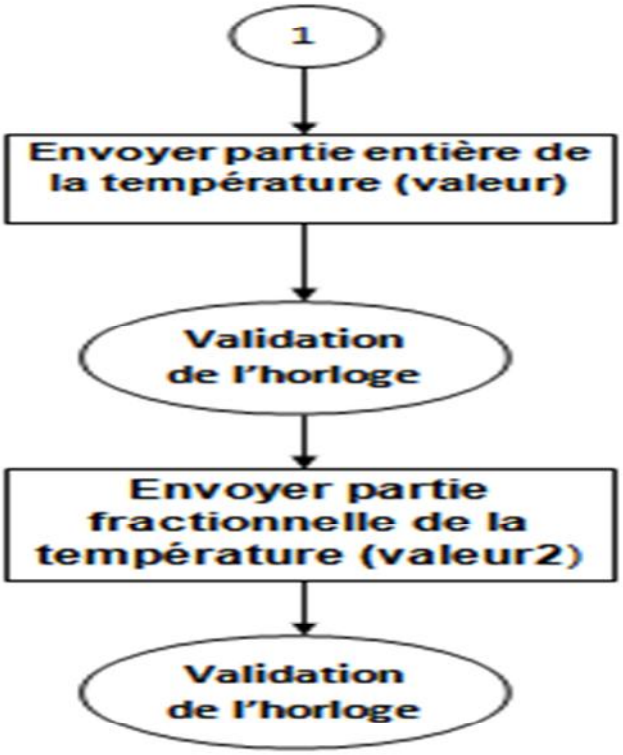
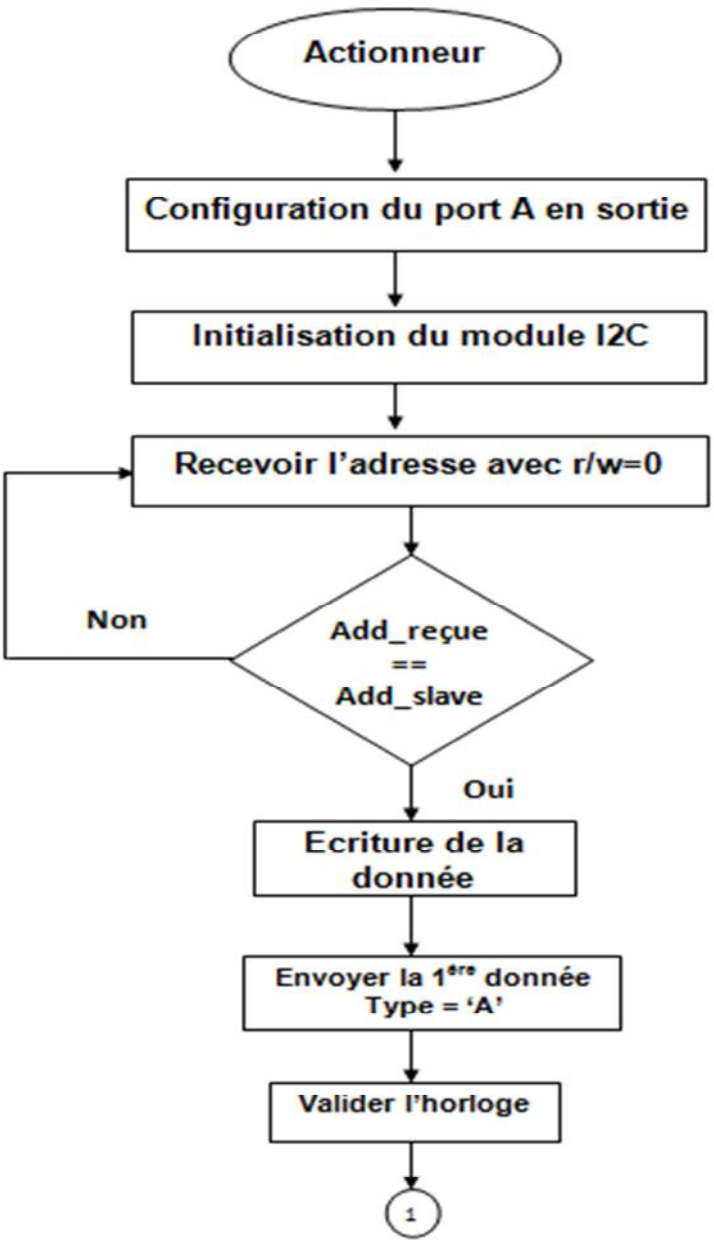


Figure III.10: organigramme du capteur I2C

III.4.3. Implémentation de l'actionneur I2C :

La figure III.11 donne l'organigramme de l'actionneur I2C illustrant son écriture pour le maitre en envoyant son type et son état (On/Off) ensuite il fait la lecture et l'exécution de la commande reçue par le maitre soit allumage ou extinction.



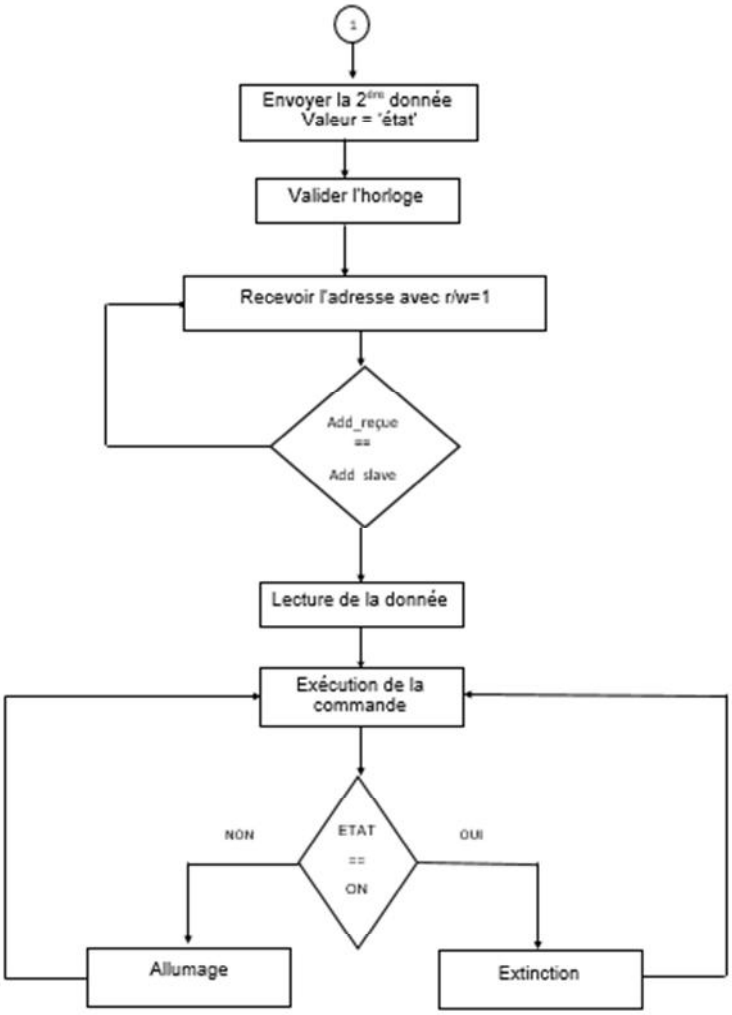
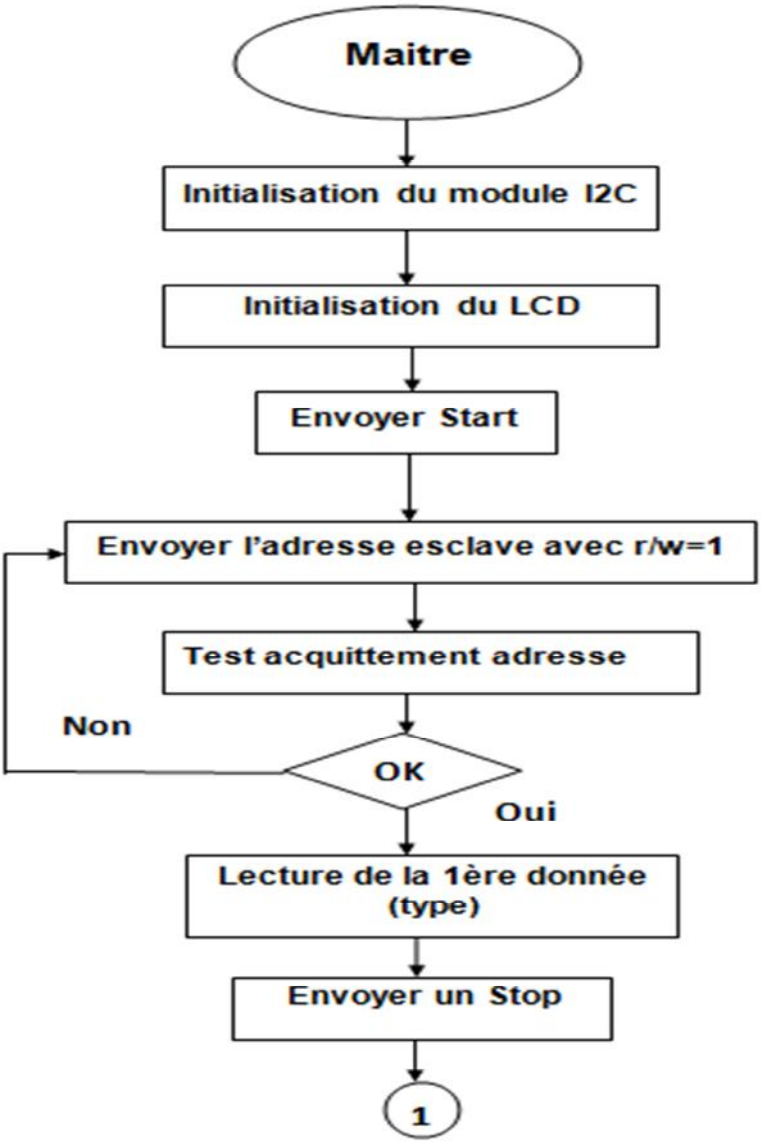


Figure III.11: organigramme du l'actionneur I2C

III.4.4. Implémentation du maitre I2c

La figure III.12 donne l'organigramme du maitre I2C qui initie la communication maitre-esclave en envoyant un Start ensuite il fait la recherche des modules qui sont connectés a lui, si le test d'acquittement reçu par ces derniers est positif, le maitre fait la lecture de la 1ére donnée type (actionneur/capteur), il envoie un ACK à l'esclave concerné en signalant qu'il a bien reçu la 1ére donnée mais qu'il attend maintenant la 2ème donnée. Lorsque le maitre aura reçu la deuxième donnée qui est la valeur de température dans le cas d'un capteur en deux partie : entière (valeur) et fractionnelle (valeur2) ou l'état (valeur) dans le cas d'un actionneur et selon ce dernier le maitre va faire l'écriture de la commande (allumage ou extinction).


Maitre
Initialisation du module I2C
Initialisation du LCD
Envoyer Start
Envoyer l'adresse esclave avec r/w=1
Test acquittement adresse
Non
OK
Oui
Lecture de la 1ère donnée (type)
Envoyer un Stop
1

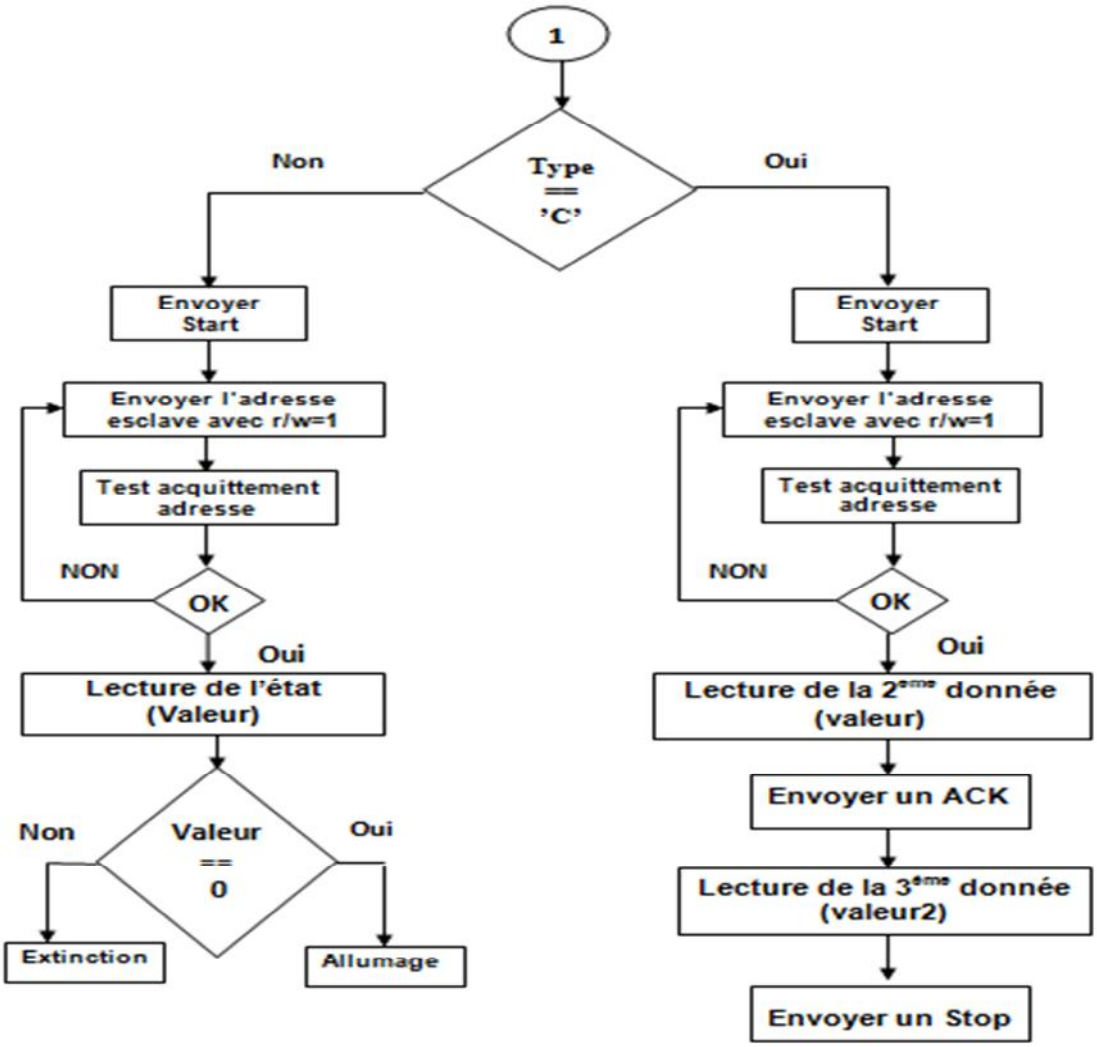


Figure III.12: organigramme du maitre I2C

III.5. Conclusion

Au début de ce chapitre, nous avons commencé par un schéma synoptique détaillé, ensuite nous avons donné une description et des caractéristiques du pic 16f877A et du capteur LM35 utilisé, puis nous avons fourni une explication détaillée de la conception de notre système illustrés avec des organigrammes et des schémas.

Chapitre IV :
Mise en œuvre

IV.1. Introduction

Après avoir terminé la conception de notre système, nous passons à la dernière phase de notre étude, qui est la réalisation. Dans ce chapitre nous allons présenter les outils utilisés pour le développement des circuits du capteur, de l'actionneur ainsi que le module maitre. Puis Nous allons montrer nos instruments électroniques que nous avons réalisés.

IV.2. Environnement de développement

La programmation des microcontrôleurs PIC est supportée par plusieurs langages de programmation tel que : MikroC for pic, MPLAB, Mikrobasic, HI-TECH C for PIC, flow code, etc... Pour la simplicité et la facilité de la programmation, En cherchant le compilateur le plus adapté aux microcontrôleurs PIC, on trouve Mplab xc8 et MikroC qu'on a utilisé au premier temps avec la carte easypic V7 et pour la simulation on trouve le logiciel PROTEUS 8 Professional. *[39]*

IV.2.1. Compilateur MikroC

L'outil *MikroC PRO for PIC* n'est pas qu'un simple compilateur, c'est un environnement de développement complet comprenant un éditeur de code source convivial, un compilateur, un *linker*, un débogueur, un programmateur et des outils pratiques, comme des terminaux pour port série, USB, UDP ou cartes MMC, des éditeurs pour bitmap, caractères LCD ou EEPROM et un décodeur d'afficheur à 7 segments.

Le grand nombre de bibliothèques pour périphériques fournies accélère l'écriture de programmes complexes avec *MikroC*. Comme bibliothèques de communication on trouve, parmi d'autres, Ethernet, I2C, RS-232/485, PS/2, CAN ou encore SPI. Sinon il y a des bibliothèques pour le stockage (EEPROM, carte MMC/SD/compact flash, etc.), pour l'interface homme-machine (clavier, afficheur graphique ou alphanumérique, son, USB, etc.), des bibliothèques de commande (PWM, E/S, etc.) et d'autres encore. La grande collection d'exemples pratiques qui mettent en œuvre les bibliothèques, ainsi que la documentation étendue, permettent d'arriver rapidement à un programme fonctionnel sans trop se perdre dans les registres du microcontrôleur ou des périphériques. Le code est enregistré dans un fichier ayant l'extension *. Hex, ce fichier est le résultat principal de la compilation, avec lequel le microcontrôleur sera programmé ou pour être utilisé pour une simulation sur ordinateur. *[43]*

IV.2.2. Compilateur MPLAB

L'environnement de développement intégré MPLAB fournit par le constructeur Micro chip, est un outil qui regroupe un éditeur de texte, un compilateur MPASM pour l'assembleur et XC8 pour le C, un outil de simulation, et un logiciel de programmation. Par conséquent il y a une possibilité de l'associer à PROTEUS pour constituer une plateforme de développement et de simulation très utile. *[44]*

IV.2.3. Proteus Professional

Proteus Professional est une suite logicielle destinée à la simulation de circuits électroniques et des systèmes embarqués. Développé par la société Labcenter Electronics, les logiciels incluent dans Proteus Professional permettent la CAO (Construction Assistée par Ordinateur) dans le domaine électronique. Deux logiciels principaux composent cette suite logicielle : (ISIS, ARES, PROSPICE) et VSM. Cette suite logicielle est très connue dans le domaine de l'électronique. De nombreuses entreprises et organismes de formation (incluant lycée et université) utilisent cette suite logicielle. Outre la popularité de l'outil, Proteus Professional possède d'autres avantages

• Pack contenant des logiciels facile et rapide à comprendre et utiliser

• Le support technique est performant

• L'outil de création de prototype virtuel permet de réduire les coûts matériel et logiciel lors de la conception d'un projet. *[45]*

- ISIS

Le logiciel ISIS de Proteus Professional est principalement connu pour éditer des schémas électriques. Par ailleurs, le logiciel permet également de simuler ces schémas ce qui permet de déceler certaines erreurs dès l'étape de conception. Indirectement, les circuits électriques conçus grâce à ce logiciel peuvent être utilisés dans des documentations car le logiciel permet de contrôler la majorité de l'aspect graphique des circuits. *[46]*

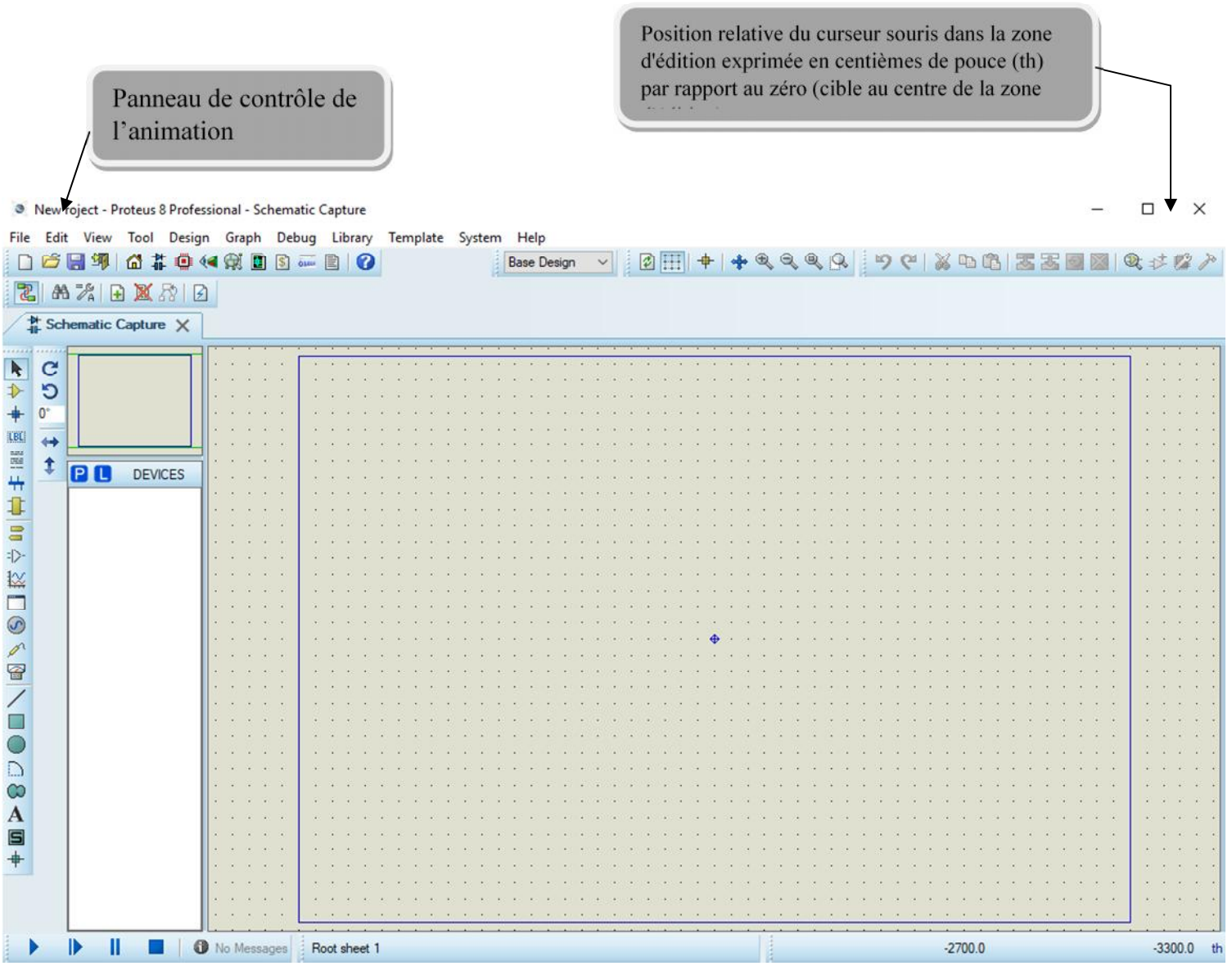


Figure IV.1 : la fenêtre ISIS

- ARES

Le logiciel ARES est un outil d'édition et de routage qui complète parfaitement ISIS. Un schéma électrique réalisé sur ISIS peut alors être importé facilement sur ARES pour réaliser le PCB (Printed circuit board) de la carte électronique. Bien que l'édition d'un circuit imprimé soit plus efficiente lorsqu'elle est réalisée manuellement, ce logiciel permet de placer automatiquement les composants et de réaliser le routage automatiquement. *[46]*

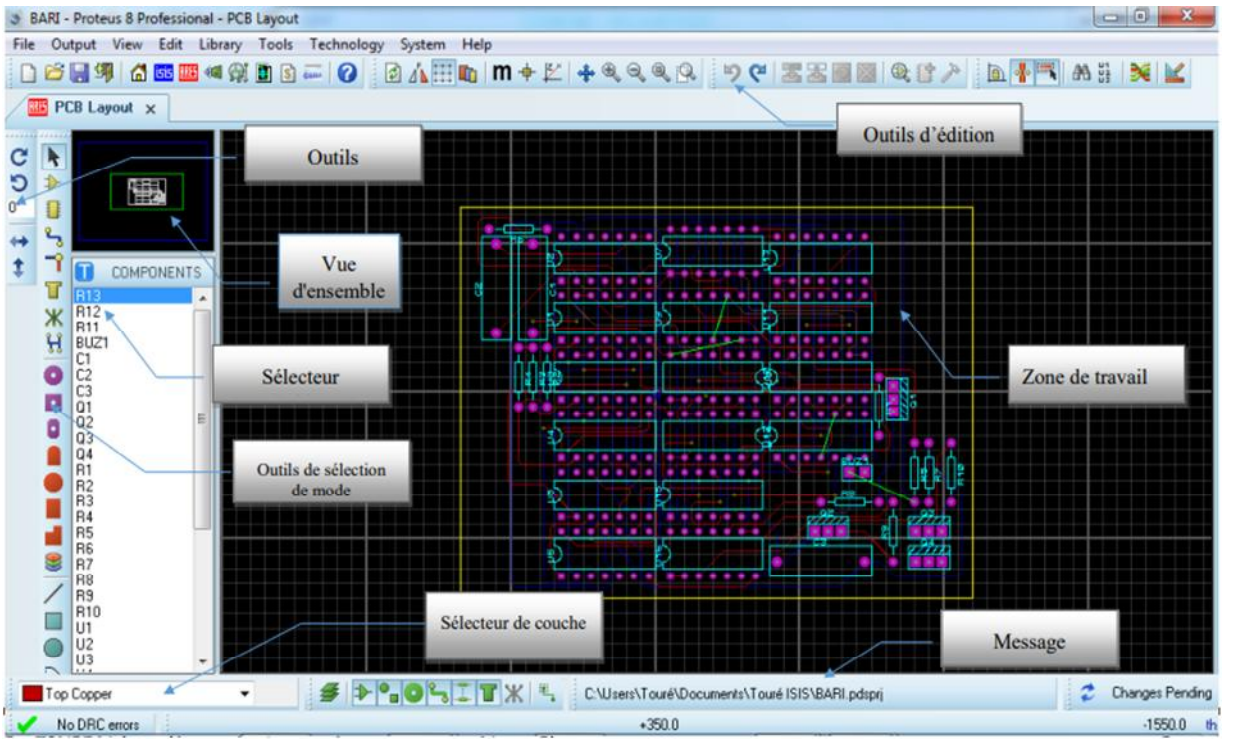


Figure IV.2 : la fenêtre ARES

IV.3. Outils de développement utilisés

IV.3.1. Easypic V7

La carte de développement EasyPIC v7 de la société Mikroelektronica permet de développer des projets utilisant les microcontrôleurs Microchip de la famille PIC 12/16/18, A l'achat elle est équipée du PIC 18F45K22.
EasyPIC v7 intègre un grand nombre de fonctionnalités :

- Leeds et boutons poussoirs connectables aux différents ports.
- Switches et cavaliers de configurations afin de s'adapter aux microcontrôleurs et à leurs caractéristiques Des connecteurs E/S au format HE10 reliés aux différents ports des microcontrôleurs.
- Programmateur sur la carte permettant la programmation in situ (sans avoir à enlever le microcontrôleur).
- Module ICD permettant le débogage du programme en simulation et en temps réel sur la carte.
- Des connecteurs afin de relier des capteurs (DS18S20 ...) Directement sur la carte.
- Un afficheur LCD et un afficheur GLCD tactile.
- 4 afficheurs 7 segments.
- Module de communication
- RS232, USB.
- Connecteur d'extension microbus sockets permettant l'ajout de « click Boards » Bluetooth, MP3, CAN SPI etc...
- Potentiomètres reliés aux entrées des convertisseurs analogique/numérique.
- Une mémoire E²PROM I²C.
- Un buzzer. *[47]*

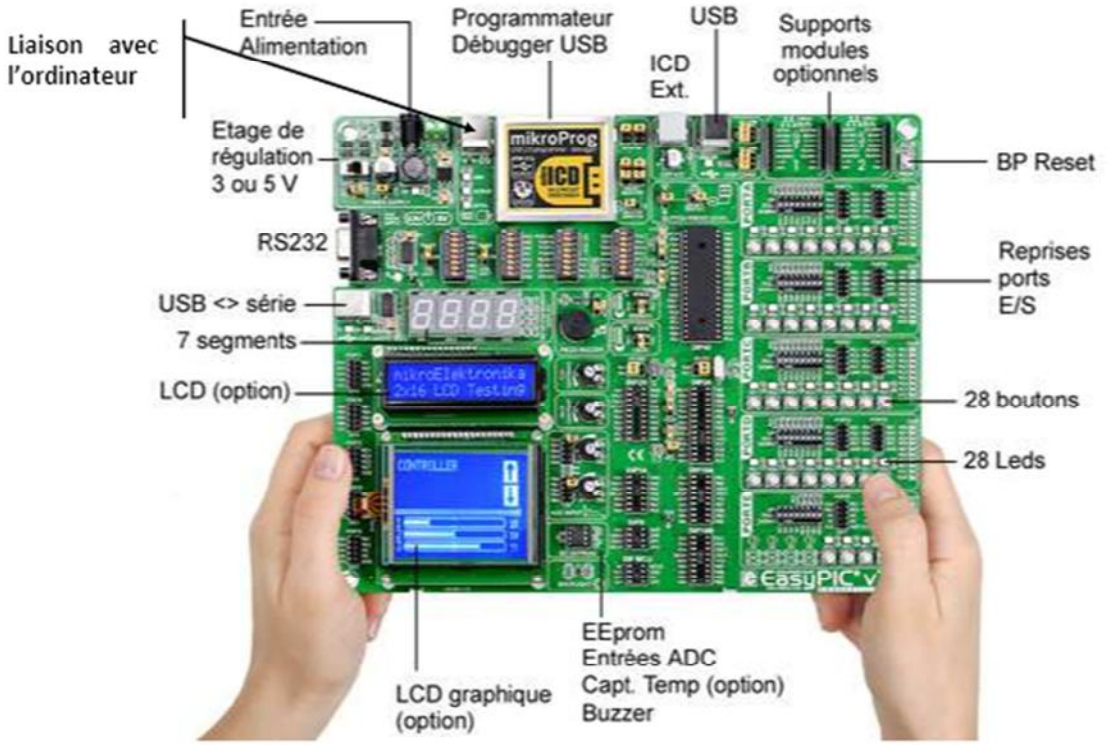


Figure IV.3 : Easypic

L'édition du programme en langage C ainsi que la compilation est effectué par l'intermédiaire du logiciel mikroC PRO La programmation est effectuée par le logiciel mikroProg Suite For PIC À travers une liaison USB.

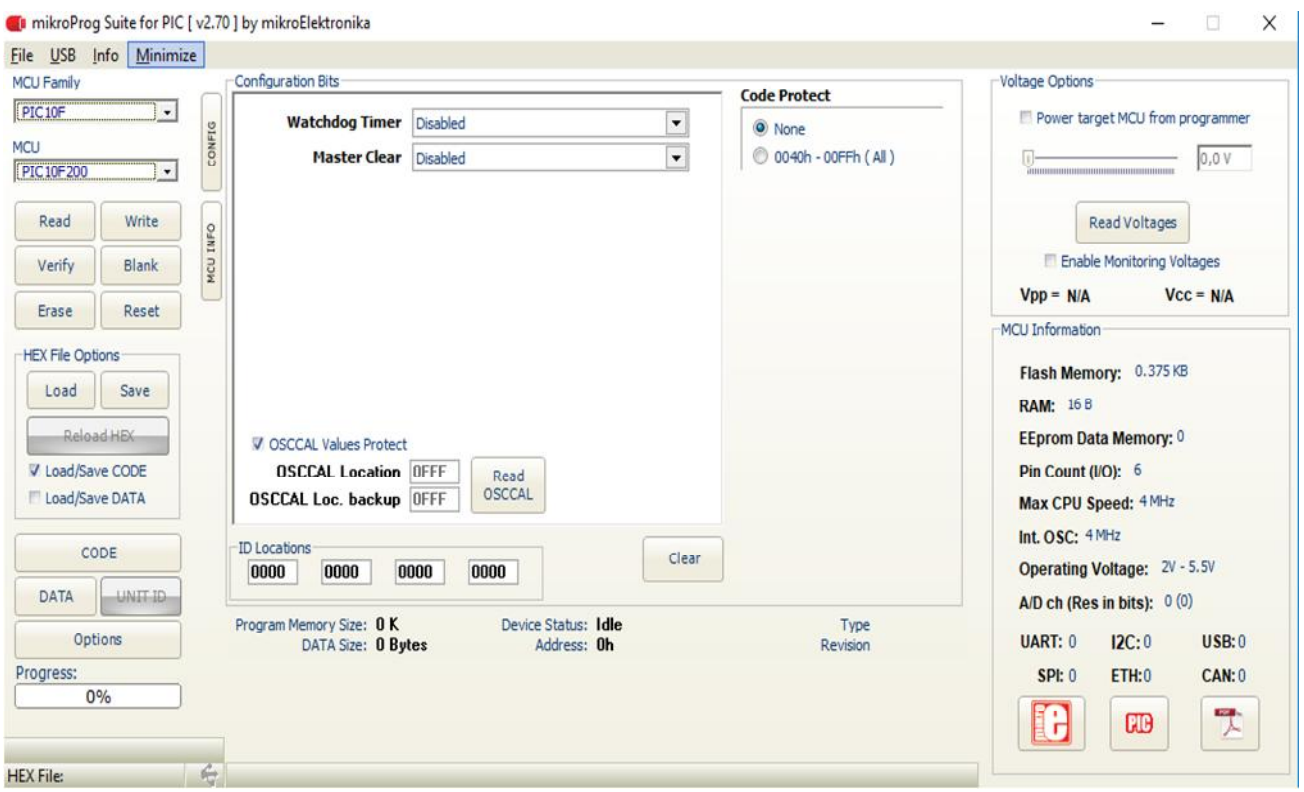


Figure IV.4 : la fenêtre mikroProg Suite for pic

IV.4. Edition des circuits

IV.4.1. Schémas du capteur I2C

Sous l'environnement ISIS et ARES, nous avons pu dessiner les schémas du capteur présentés ci-dessous

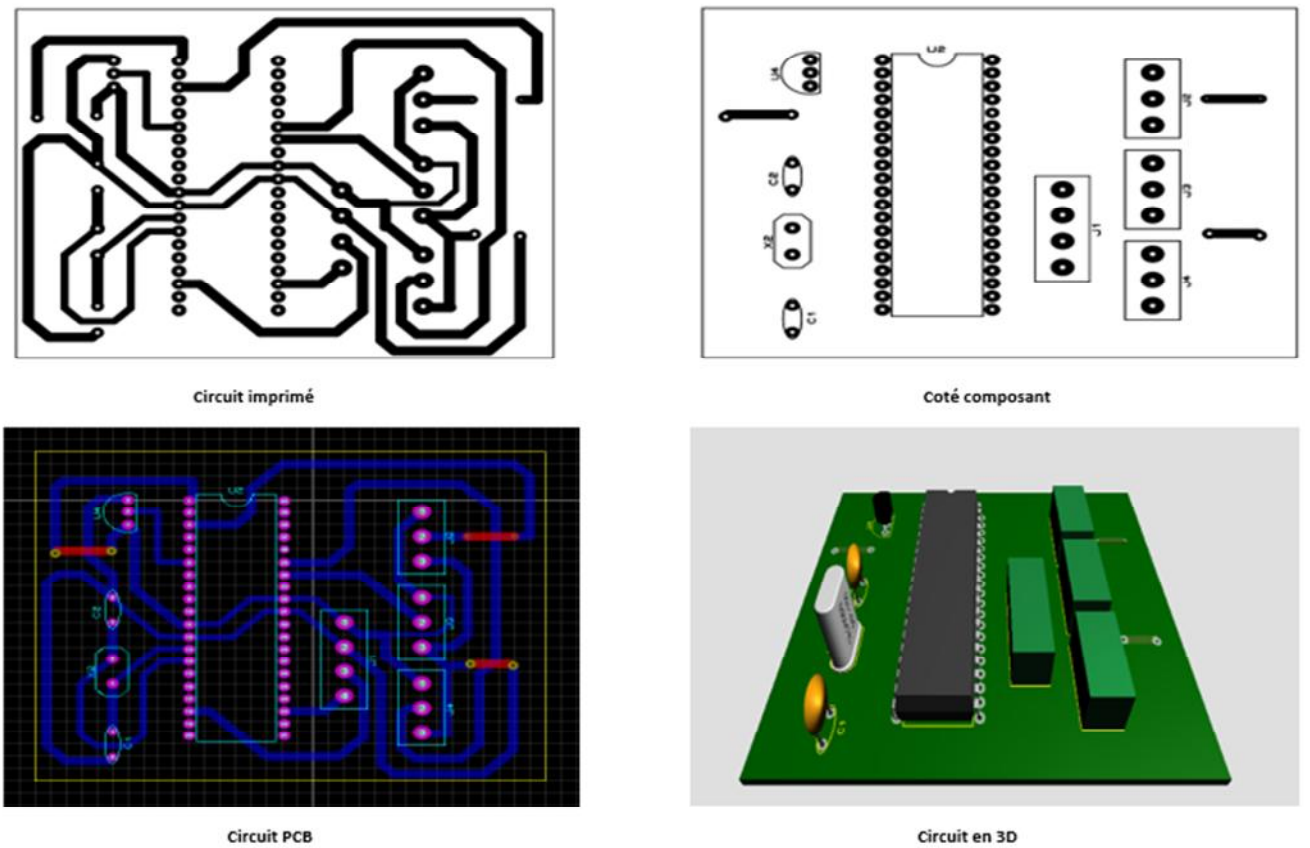
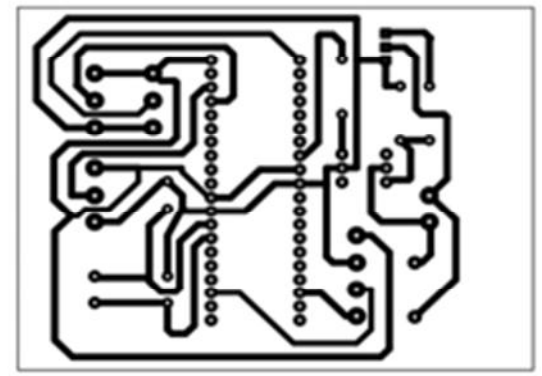


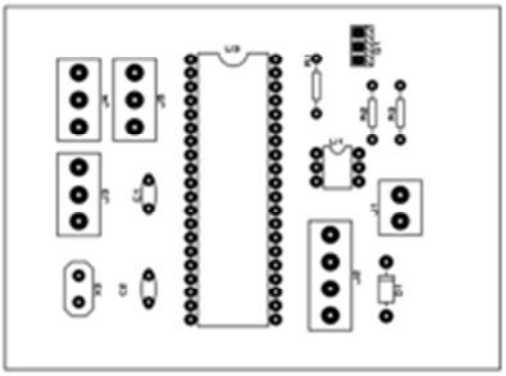
Figure IV.5 : schémas du capteur I2C

IV.4.2. Schémas de l'actionneur I2C

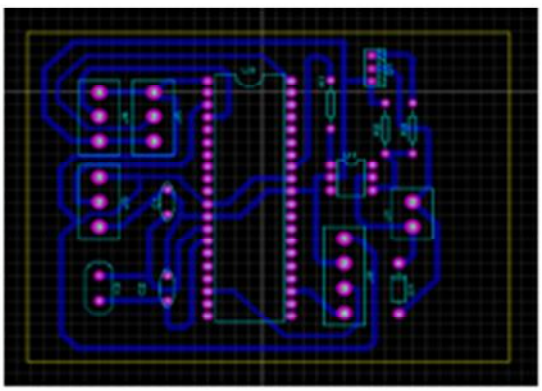
Sous l'environnement ISIS et ARES, nous avons pu dessiner les schémas de l'actionneur présentés ci-dessous



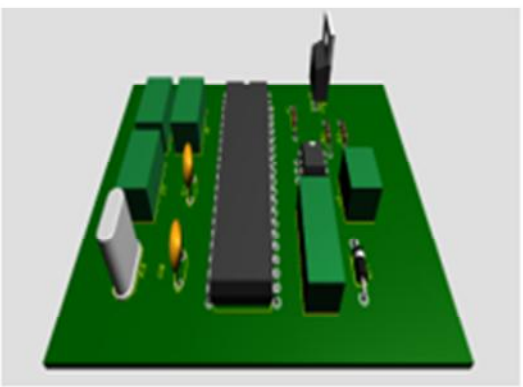
Circuit imprimé



Coté composant



Circuit PCB



Circuit en 3D

Figure IV.6 : schémas de l'actionneur I2C

IV.4.3. Schémas du maitre I2C

Sous l'environnement ISIS et ARES, nous avons pu dessiner les schémas du maitre présentés ci-dessous

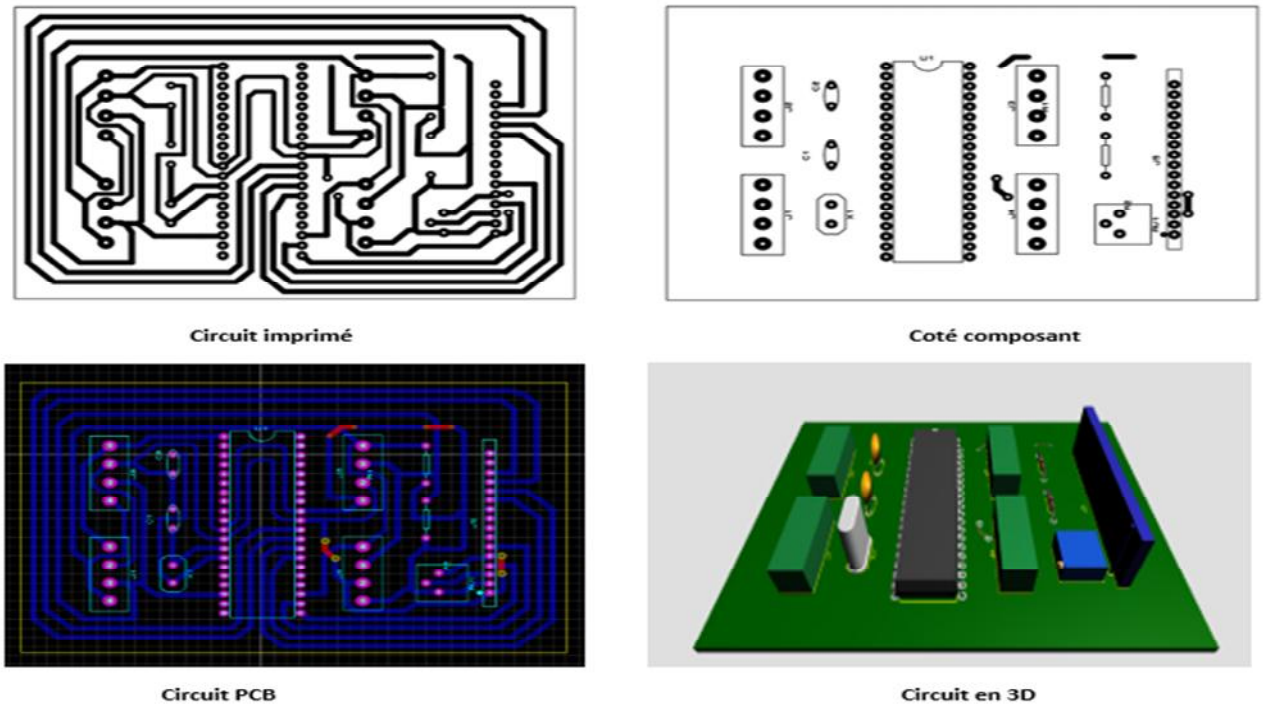


Figure IV.7 : schémas du maitre I2C

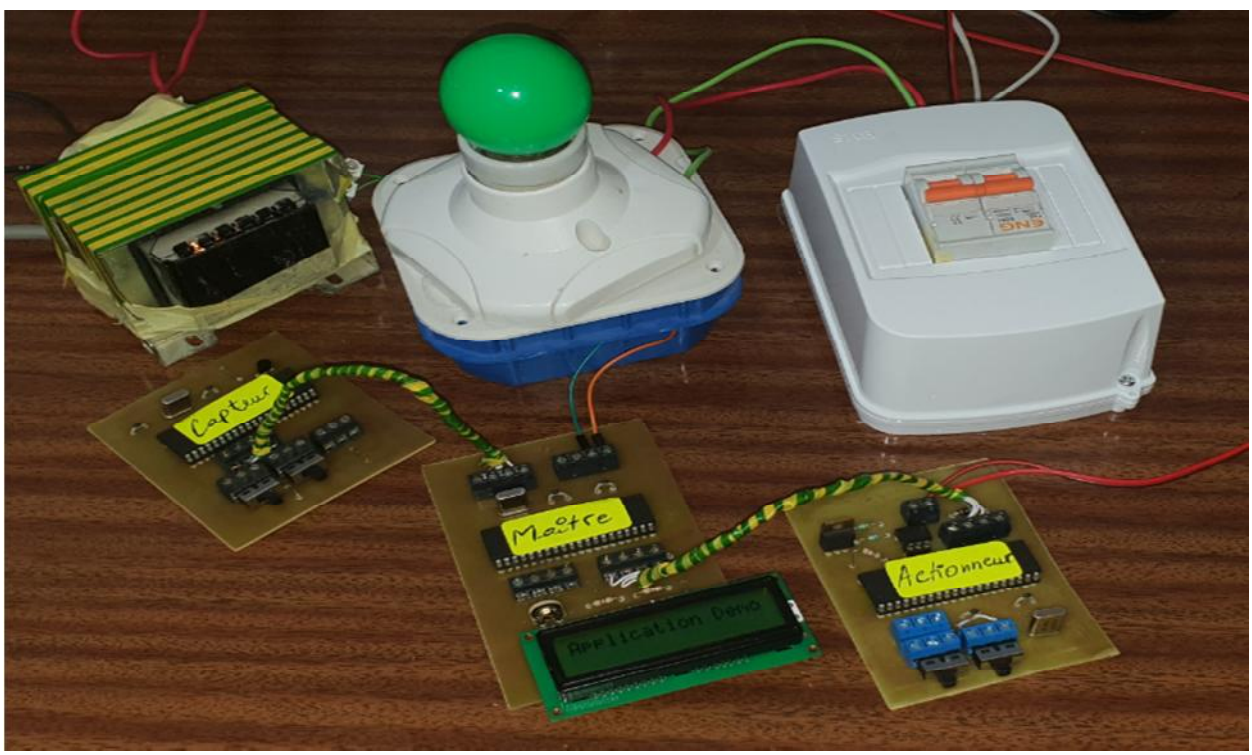


Figure IV.8 : photo réelle de notre projet

IV.5. Conclusion

Dans ce chapitre, nous avons présenté en détail la réalisation de notre projet. Nous avons commencé par la description des environnements matériels et logiciels.

Ensuite, nous avons réalisé une description de notre système en présentant des captures d'écran témoignant les différentes tâches. Enfin, nous avons présenté les différentes étapes dans un chronogramme des tâches.

Conclusion générale

Notre projet de fin d'étude à pour but l'étude et la réalisation d'un protocole de communication I2C entre une centrale de domotique et ses capteurs/actionneurs dans l'objectif de faciliter la gestion et l'auto-configurabilité de ces périphériques (capteurs, actionneur …), nous avons aussi réalisé un module maitre dans lequel on a implémenté une application démonstrative du fonctionnement de notre système réalisé (maitre-esclave).

Le système a été simulée sous le logiciel ' Proteus ' et validé avant d'être réalisé conformément aux objectifs visés.

Le programme est développé en langage C, compilé par mplab xc8 et injecté dans les pics 16F877A associés via la carte easypicV7.

Ce projet a été bénéfique pour notre formation il nous a permis de :

- Comprendre la philosophie des systèmes embarqués, leur architecture et l'utilisation des systèmes électroniques standards.
- Connaitre la famille des microcontrôleurs MICROCHIP, connus par leur richesse en terme d'élément d'E / S, possibilités de programmabilité, souplesse de développement et de mise en œuvre pratique.
- Programmer en langage C le microcontrôleur pic 16f877A.

-acquérir des connaissances en programmation pour les microcontrôleurs PIC sous le compilateur MikroC et de dépasser les problèmes trouvés dans la simulation sous ISIS PROTEUS.

Comme perspective pour notre travail nous proposons d'intégrer nos modules (capteurs, actionneur ...) dans une smart home qui facilite l'envoi des messages par l'utilisateur via une application Android Things ainsi qu'une amélioration dans le programme pour effectuer d'autres effets d'affichage.

Enfin, nous souhaitons que notre travail sera bénéfique à toute personne souhaitant réaliser ce genre d'application et qu'il pourra être poursuivi par d'autres étudiants dans le but d'améliorer et de simplifier encore plus son fonctionnement.

Bibliographie

[4] : **GHADBANE, Toufik MEKHALFIA Toufik et.** *"Etude et réalisation d'un système de commande à distance des installations électriques pour la domotique".* M'sila : Université Mohamed Boudiaf, 2018.

[5] : **Hamid, HAMOUCHI.** *"Conception et réalisation d'une centrale embarquée de la domotique «Smart Home »".* Rabat : Université Mohammed V, 2015.

[9] : **Eckert, Alleguede et.** *Le guide de la domotique.* 2012.

[10] : **BOUDELLAL, MEZIANE.** *"Smart home - Habitat connecté, 361 installations domotiques et multimédia".* [éd.] Dunod. 2014.

[12] : **Selma, METAHRI Mohammed El habib et ABDELLI.** *Smart House.* Tlemcen : Université ABOU BEKR BELKAID, 2017.

[15] : **MONTAGNY, Sylvain.** *Bus de communication.* s.l. : Université de Savoie.

[37] : **MALIK, SMADI OUSSAMA et BENYALLOUL.** *" Utilisation d'une souris pour mesure de distance ".* TLEMCEN : UNIVERSITE ABOU BAKR BELKAID, 2015.

[38] : **SMAIL, Soufiane.** *"Commande de l'éclairage public et mesure de la température à base de pic18f4550".* Biskra- Algérie : Université Mohamed Khider, 2013. Mémoire Master.

[40] : **BALI, Chaher.** *Réalisation d'un robot mobile avec évitement d'obstacle et trajectoire programmée.* Biskra : Mohamed Khider, 2012.

[1] : Union confédérale CFDT des retraités. [En ligne] 16 9 2014. [Citation : 5 7 2019.] https://www.xn--cfdt-retraits-mhb.fr/Claude-Titre-a-trouver.

[2] : **Beautiful Company.** le Mag de la Domotique. [En ligne] 2018. [Citation : 5 7 2019.] https://www.lemagdeladomotique.com/dossier-1-domotique-definition-applications.html.

[3] : **Cea.** La domotique ou la maison connectée. [En ligne] 14 octobre 2016. [Citation : 20 juillet 2019.] http://www.cea.fr/comprendre/Pages/nouvelles-technologies/essentiel-sur-domotique-maison-connectee.aspx.

[6] : **Noémie Vialard.** *Domotique sans fil : principe - Ooreka.* [En ligne] 2017-2018. [Citation : 20 7 2019.] https://domotique.ooreka.fr/comprendre/domotique-sans-fil.

[7] : **Cyril Byen.** *ZONEADSL.* [En ligne] 21 11 2016. [Citation : 30 7 2019.] https://www.zoneadsl.com/dossiers/maison-connectee/bluetooth.html.

[8] : **SARL Gallop.** [En ligne] 2012. [Citation : 21 7 2019.] https://www.gallop-renovation.com/appartement/domotique/.

[11] : luminus. [En ligne] 9 8 2017. [Citation : 25 7 2019.] https://lumiworld.luminus.be/fr/up-to-date-fr/la-maison-intelligente-permet-des-economies-denergie/.

[13] : *la renovation.* [En ligne] 21 juin 2004. [Citation : 7 aout 2019.] http://www.larenovation.fr/le-blog-de-la-mutation/1002-1002.

[14] : **Faillie, Laurent.** *Cultura.* [En ligne] 2001. [Citation : 7 aout 2019.] http://destroyedlolo.info/BananaPI/1wire/.

[16] : Liaison SPI. [En ligne] https://lewebpedagogique.com/isneiffel/files/2017/06/Liaison-SPI.pdf.

[17] : *LE BUS I2C.* [En ligne] http://www.gecif.net/articles/genie_electrique/cours/terminale/cours/le_bus_I2C.pdf.

[18] : [En ligne] https://www.futura-sciences.com/maison/definitions/maison-konnex-10741/.

[19] : [En ligne] http://arteco-electricite.com/blog/protocole-knx-cest-quoi/.

[20] : *Siemens.* [En ligne] aout 2003. [Citation : 1 aout 2019.] http://www.siemens.ch/sbt/datenblaetter/fr/downloads/n3127fr.pdf.

[21] : [En ligne] http://ww2.ac-poitiers.fr/electronique/sites/electronique/IMG/pdf/Fiche_de_synthese_Principe_CPL.pdf.

[22] : [En ligne] https://www.ovhtelecom.fr/telephonie/accessoires/cpl_fiche_technique.xml.

[23] : [En ligne] https://www.abcelectronique.com/annuaire/cours/cache/551/le-bus-can.html.

[24] : [En ligne] https://www.electronique-mixte.fr/wp-content/uploads/2018/07/Formation-Interface-communication-42.pdf.

[25] : [En ligne] 2018. https://electrosome.com/i2c-pic-microcontroller-mplab-xc8/?fbclid=IwAR2FmsxtGMXsbhTlaarwa6ohJCy9FtFap9NqeVJKQxrG1IxwtLcJXnoqxgw.

[26] : [En ligne] http://fabrice.sincere.pagesperso-orange.fr/cm_electronique/projet_pic/aidememoire/16F876A_bus_I2C/bus_I2C_16F876A.htm.

[27] : [En ligne] https://les-electroniciens.com/sites/default/files/cours/cours_i2c.pdf.

[28] : **Jarno, Aurélien.** [En ligne] 2008. https://www.aurel32.net/elec/i2c.php.

[29] : [En ligne] http://infoindustrielle.free.fr/Les_reseaux/Reseaux_htm/22N-I2C.htm.

[30] : [En ligne] http://dlnware.com/dll/Repeated-START-Sr-Condition.

[31] : [En ligne] http://dspace.univ-biskra.dz:8080/jspui/bitstream/123456789/12411/1/ZEMALI_Mounir.pdf.

[32] : [En ligne] https://askri-t.jimdo.com/app/download/13669213924/Le+module+MSSP+en+mode+I2C.pdf?t=1476482142

[33] : [En ligne] https://www.abcelectronique.com/annuaire/cours/cache/1156/le-pic-16f876-877.pdf.

[34] : [En ligne] http://www.mytopschool.net/mysti2d/activites/polynesie2/ETT/C044/32/BusI2C/index.html?LebusI2C.html.

[35] : [En ligne] http://tvaira.free.fr/projets/activites/activite-bus-i2c.html.

[36] : [En ligne] http://electronique71.com/.

[39] : [En ligne] https://www.technologuepro.com/microcontroleur/les-microcontroleurs-pic.htm.

[41] : [En ligne] https://boutique.semageek.com/fr/413-capteur-de-temp%C3%A9rature-lm35dz.html.

[42] : [En ligne] https://www.elektormagazine.fr/news/compilateur-mikroc-pro-for-pic-2009.

[43] : [En ligne] http://paristech.institutoptique.fr/site.php?id=10&fileid=15948.

[44] : [En ligne] http://www.magoe.net/coursUniv/coursUniv_146_pdf.pdf.

[45] : [En ligne] http://www.elektronique.fr/logiciels/proteus.php.

[46] : [En ligne] https://fr.rs-online.com/web/p/kits-de-developpement-pour-processeurs-et-microcontroleurs/7916343/.

[47] : [En ligne] http://www.simius.be/interfacage/bus-i2c/i2c.html.

ANNEXE

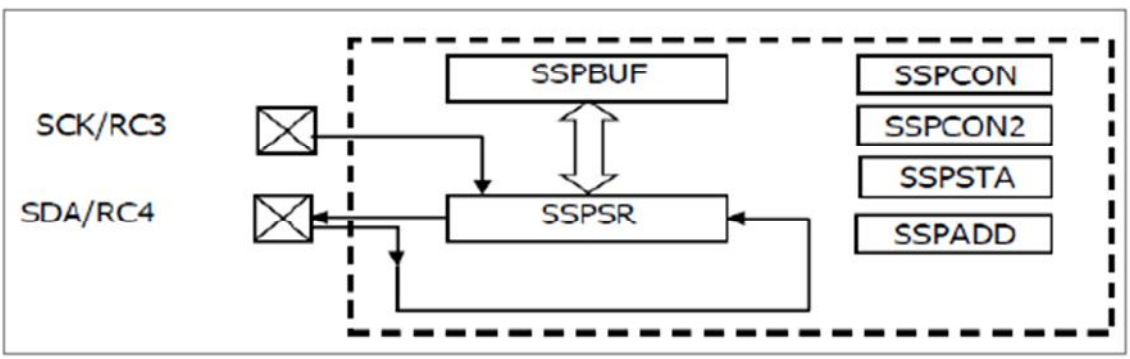
Le module MSSP (Master Synchronous Serial Port)

Le MSSP est un des deux modules de communication série du PIC 16F876/877. Il permet d'échanger des données en mode synchrone avec d'autres circuits qui peuvent être des microcontrôleurs, des mémoires EEPROM série, des convertisseurs A/N, des modules d'affichage . . . Il peut fonctionner selon deux modes : le mode SPI (Serial Peripheral Interface) et le mode I2C (Inter-Integrated Circuit)

Le module MSSP en mode I2C

Le module MSSP du PIC peut être configure en master ou en slave. Il utilise les broches RC3/SCL (Horloge) et RC4/SDA (données). Ces broches doivent être configurées en ENTREE a l'aide du registre TRISC et doivent être munie de Resistances de pull-up externes nécessaires au fonctionnement I2C. Les fréquences d'horloges supportées sont 100 kHz, 400 kHz et 1 MHz.

L'accès au module en lecture et écriture se fait à l'aide du registre tampon (buffer) SSPBUFF. La transmission et la réception se fait a l'aide du registre a décalage SSPSR auquel nous n'avons pas directement accès.



Transmission d'un octet

Pour transmettre un octet, il suffit de le copier dans le registre SSPBUF, et le module MSSP s'occupe du reste. Au moment de l'écriture dans SSPBUF, le bit BF passe à 1 et la transmission commence.

A la fin de la transmission, le bit SSPSTAT.BF repasse à 0 et le drapeau d'interruption PIR1.SSPIF passe à 1.

Le bit BF apparait donc comme un bit très important, c'est lui qui nous permet de savoir si le registre SSPBUF est libre ou non Si on tante d'écrire dans SSPBUF alors que BF=1, le bit SSPCON.WCOL passe a 1 pour indiquer une collision et l'écriture n'a pas lieu.

Réception d’un octet

A la fin de la réception d'un octet, celui-ci est transfère dans SSPBUF, l’indicateur SSPSTAT.BF et le drapeau d’interruption PIR1.SSPIF passent a 1. BF repasse automatiquement à 0 au moment de la lecture de SSPBUF alors que SSPIF soit être remis a 0 par soft. Si le PIC termine la réception d'un octet avant que l'octet précédent qui se trouve dans SSPBUF n'ait été lu, on à un Over flow qui sera signale par le drapeau SSPOV. Le transfert n'a pas lieu, l’octet arrive est perdu.

Les registres de configuration

Le control du module se fait à l'aide des registres :

SSPCON : registre de control (BANK0)

SSPCON2 : registre de control (BANK1)

SSPSTAT : registre d'état (BANK1)

SSPADD : registre d'adresse (BANK1)

Le registre SSPSTAT

SSPSTAT	SMP	CKE	D/A	P	S	R_W	UA	BF

SMP : Control de slew rate : 1 pour 100 kHz et 1 MHz, 0 pour 400 kHz

CKE : Control des niveaux de tension a l'entrée, 0 : I2C, 1 : SMBUS

D/A : Indicateur d'état, 0 : dernier octet émis/reçu = adresse, 1 : dernier octet

émis/reçu = donnée

P : Indicateur de Stop Condition. Ce bit passe à 1 quand on détecte un Stop Condition. Il est automatiquement remis a 0 quand un autre événement et reçu.

S : Indicateur de Start Condition. Ce bit passe a 1 quand on détecte un Start Condition. Il est automatiquement remis a 0 quand un autre événement et reçu

R_W : Indicateur de lecture écriture

Mode master : 0 : pas de transmission en cours, 1 : transmission en cours

Mode slave : image du bit R/W qui constitue le bit 0 du premier octet suivant le start condition. 0 : écriture, 1 : lecture

UA : Indicateur utilise en mode esclave avec adressage 10 bits pour nous informer qu'il faut placer la 2eme partie de l'adresse dans le registre adresse. Ce dernier ayant seulement 8 bits, l'adresse est traitée en 2 parties. Ce bit est remis automatiquement a zéro âpres l'écriture dans SSPADD.

BF : Indicateur sur l'état du registre SSPBUF.

Le registre SSPCON

SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0

WCOL : Collision d'écriture. En mode I2C master, ce bit passe a 1 si on tente d'écrire dans

SSPBUF à mauvais moment de la séquence de transmission (BF=1). En mode I2C slave, il est positionne si on tente d'écrire dans SSBUF avant la fin de la transmission de l'octet précédant. Ce bit doit être remis à zéro par programme.

SSPOV : Overflow, cet indicateur d'erreur est positionne quand une nouvelle donnée est reçue alors que le registre SSBUF n'est pas encore lu. Dans ce cas la nouvelle donnée est perdue. Ce bit doit être remis à zéro par programme.

SSPEN : Validation du module MSSP (1 = valide)

CKP : Utilise en mode I2C slave pour générer des pauses en inhibant l'horloge. 0 : horloge forcée a zero, 1 : horloge validée

SSPM3:SSPM0 : mode de fonctionnement du module

0110: I2C Slave, adresse 7 bits

0111: I2C Slave, adresse 10 bits

1000: I2C master, fréquence = F_{OSC} / (4 * (SSPADD+1))

1011 : I2C Slave force a l'état de repos

1110 : I2C master, adresse 7 bits, interruption sur START et STOP

1110 : I2C master, adresse 10 bits, interruption sur START et STOP

Le registre SSPCON2

SSPCON2	GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN

Le bit 7 concerne le mode slave, les autres bits concernent le mode master :

GCEN : General call enable bit : Si ce bit est valide, le PIC répond a son adresse et a l'adresse générale 0000h

ACKSTAT : Acknowledge Status bit (mode master) : ce bit est l'image de l'accuse de réception envoyé par le slave. 0=accuse positif, 1=accuse négatif

ACKDT : Acknowledge Data bit (master mode) : Valeur du bit envoyé au slave en fin de réception. Ce bit est envoyé au moment ou on positionne le bit ACKEN.

ACKEN : Acknowledge Séquence Enable bit (master mode) : quand il est place a un, ce bit démarre une séquence accuse de réception qui consiste à placer les broches SDA et SCL dans le mode correspondant au bit ACKDT

RCEN : Receive Enable bit : (master, réception) Quand on le place a 1, ce bit démarre la réception d'un octet. Il revient automatiquement a 0 après la transmission du 8_{eme} bit. La réception ne démarre que si l'horloge n'est pas stretch (forcée a 0: pause) par le slave.

PEN : STOP Condition Enable bit, démarre une séquence stop condition sur les broches SDA et SCL. Ce bit est remis à 0 automatiquement âpres la séquence du STOP

RSEN: Repeated START Condition Enable bit: démarre une sequence Repeated Start condition. Il est remis à 0 automatiquement après la séquence du R-START

SEN : START Condition Enable bit : démarre une séquence Start condition. Il est remis à 0 automatiquement après la séquence du START

Le registre SSPADD

En mode I2C **master**, ce registre permet de déterminer la fréquence de communication

$$F = \frac{Fosc}{4 \times (SSPADD + 1)} \; Hz$$

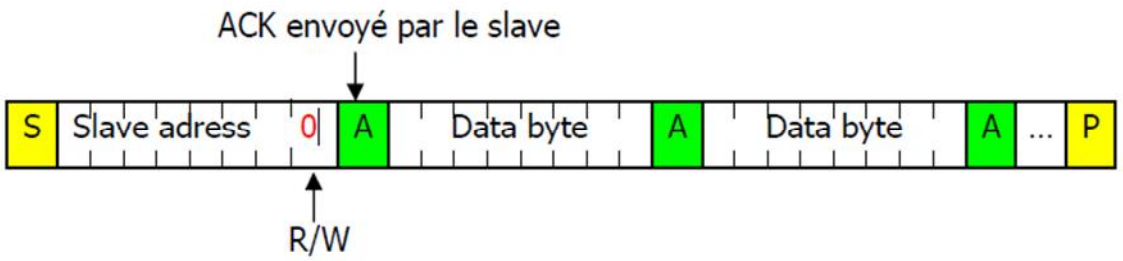
Les fréquences possibles sont : 100 kHz, 400 kHz et 1 MHz

$$SSPADD = \frac{Fosc}{4 \times F} - 1 \; Hz$$

En mode I2C **slave**, ce registre doit contenir l'adresse du slave. Le bit 0 est réservé, il doit être toujours place a zero, l'adresse doit être écrite a partir du bit 1. En cas d'adresse 7 bits, cela ne pose pas de problème. En cas d'adresse 10 bits, l'adresse est écrite en 2 temps, On commence par écrire 1 1 1 1 0 A_9 A_8 0 dans SSPADD, en attend l'indicateur UA, ensuite on écrit A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0

Une séquence de communication est toujours initiée par le master qui commence par envoyer un **S**TART bit (SSPCON2.**S**EN), suivi de l'adresse du slave. L'adresse sera codée seulement sur 7 bits, Le LSB qu'on appelle bit R/W permet de préciser le sens de l'échange qui va suivre, R/W=0 pour transmission, R/W=1 pour réception. On envoie ou on reçoit les données et on termine la communication par un sto**p** bit (SSPCON2.**P**EN) Si l'on désire changer le sens de l'échange avant l'envoi du stop, ce qui évite l'éventualité de perdre la ligne au profit d'un autre master, il faut envoyer un RepeatStart condition suivi de l'adresse accompagnée d'un R/W différent

Master en transmission



• On positionne le bit SSPCON2.SEN sur le master, celui-ci envoie le START bit sur la ligne SDA puis raz automatiquement le bit SEN et positionne le drapeau d'interruption PIR1.SSPIF, ce dernier doit être raz par soft. Le slave voit le START, positionne son bit S et se prépare a recevoir l'adresse,

• On copie l'adresse avec R/W=0 dans SSPBUF, les bits SSPSTAT.BF et SSPSTAT.R_W passent a 1, la transmission démarre, après le 8eme coup d'horloge, SSPSTAT.BF repasse a zero, après le 9eme R_W repasse a 0, et le drapeau d'interruption SSPIF passe a 1 et l'horloge est forcée a 0 (pause) jusqu'`a ce qu'un nouvel octet est copie dans SSPBUF. Le drapeau SSPIF doit être raz par soft pour qu'il puisse servir dans la suite.

• L'acquittement sur l'adresse est renvoyé par le slave pendant le 9eme coup d'horloge sur la ligne SDA : 0 (ACK) s'il a lu l'octet correctement, 1 (NoACK) s'il n'a pas pu lire l'octet correctement. Ce bit recopie par le master dans le bit SSPCON2.ACKSTAT. Le programmeur doit vérifier l'état de ce bit pour décider de la suite des événements,

• Si acquittement de l'adresse est négatif on envoie un R-START et on renvoie l'adresse

• Si acquittement de l'adresse est positif, on envoie un octet de donnée en l'écrivant dans

SSPBUF. Tout se passe comme lorsqu'on a envoyé l'adresse. Les bit BF et R_W passent a 1 au début de l'émission et repassent a 0 a la fin. Le drapeau SSPIF passe à 1. Le slave renvoie l'acquittement pendant le 9eme bit qui sera copie dans le bit ACKSTAT.

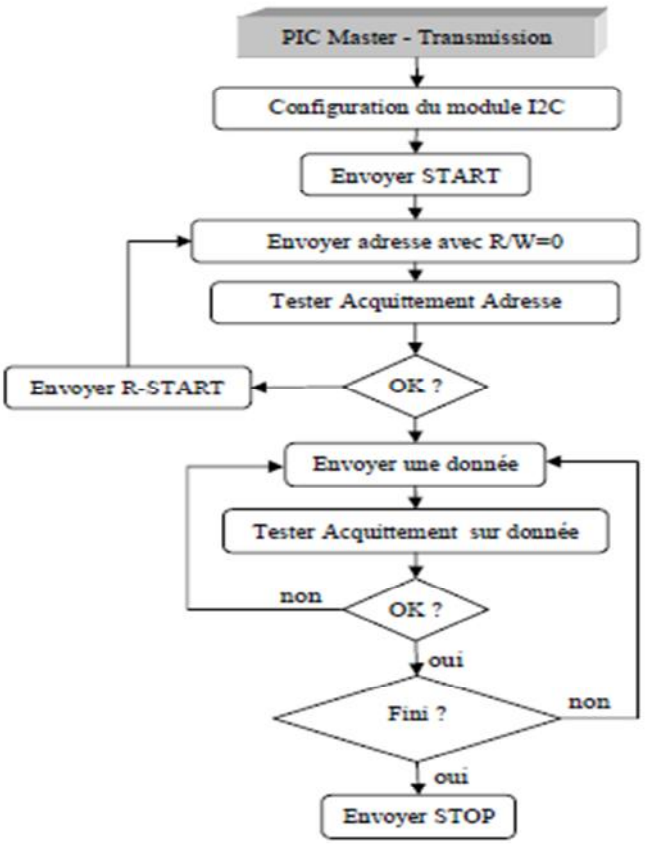
• Si l'acquittement sur la donnée est négatif, le plus simple est de retransmettre la donnée

• Si l'acquittement sur la donnée est positif :

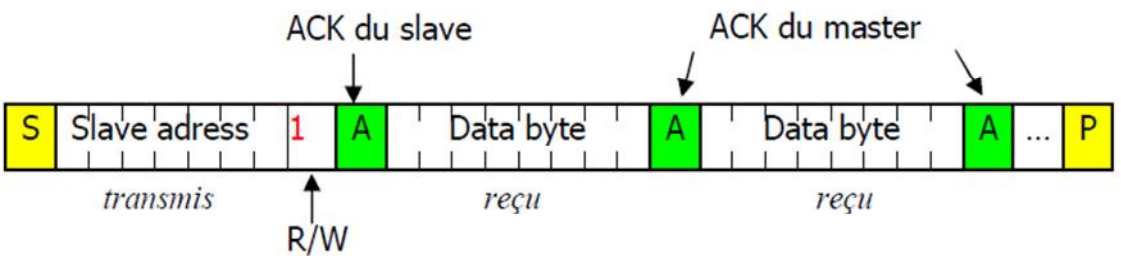
Si on a encore des données a transmettre, on démarre une nouvelle émission en écrivant dans SSPBUF.
Si on n'a plus de données a transmettre, on envoie un stop bit en positionnant le bit

SSPCON2.PEN qui est remis a zero automatiquement après la transmission d'un Stop bit sur la ligne SDA.

• Quand le slave voie le stop bit, il raz le bit S et positionne le bit P puis réinitialise son électronique pour se préparer a la réception d'un nouveau START bit.



Master en réception :



• On positionne le bit SSPCON2.SEN sur le master, celui-ci envoie le START bit sur la ligne SDA et raz automatiquement le bit SEN et positionne le drapeau d'interruption PIR1.SSPIF, ce

dernier doit être raz par soft. Le slave voit le START bit, il place le bit SSPSTAT.S a 1 et se prépare a recevoir l'adresse,

• On copie l'adresse avec R/W=1 dans SSPBUF, les bits SSPSTAT.BF et SSPSTAT.R_W passent a1, la transmission démarre, après le 8eme coup d'horloge, SSPSTAT.BF repasse a zero, après le 9eme R_W repasse a 0, et le drapeau d'interruption SSPIF passe a 1, il doit être raz par soft pour qu'il puisse servir dans la suite.

• Pendant le 9eme coup d'horloge, le slave renvoie un ACK/NOACK selon sa situation et force l'horloge a zero (pause) pour empêcher le master de lire avant que la donnée ne soit préparée sur le bus. On appelle ca : clock stretching.

• Le programme du slave prépare la donnée sur le bus puis libère l'horloge a l'aide du bit CKP

• Le programme du master doit tester l'acknowledge dans SSPCON2.ACKSTAT, si = 0(OK), il positionne le bit SSPCON2.RCEN pour demander le début d'une réception.

• Le module I2C master voyant le RCEN=1 essaye de commencer une réception par l'envoi d'une rafale de 8 coups d'horloge. Quand il place l'horloge a 1 (première impulsion), celle-ci peut ne pas passer a 1 car il se peut qu'elle soit forcée a 0 par le slave. Dans ce cas le master attend que l'horloge passe à 1 pour continuer la rafale de coup d'horloge

• En résumé, des que les deux conditions {RCEN=1 et horloge libre} sont vérifiées, le master envoie une rafale de 8 coups d'horloge qui s'accompagne de la réception de 8 bits de données.

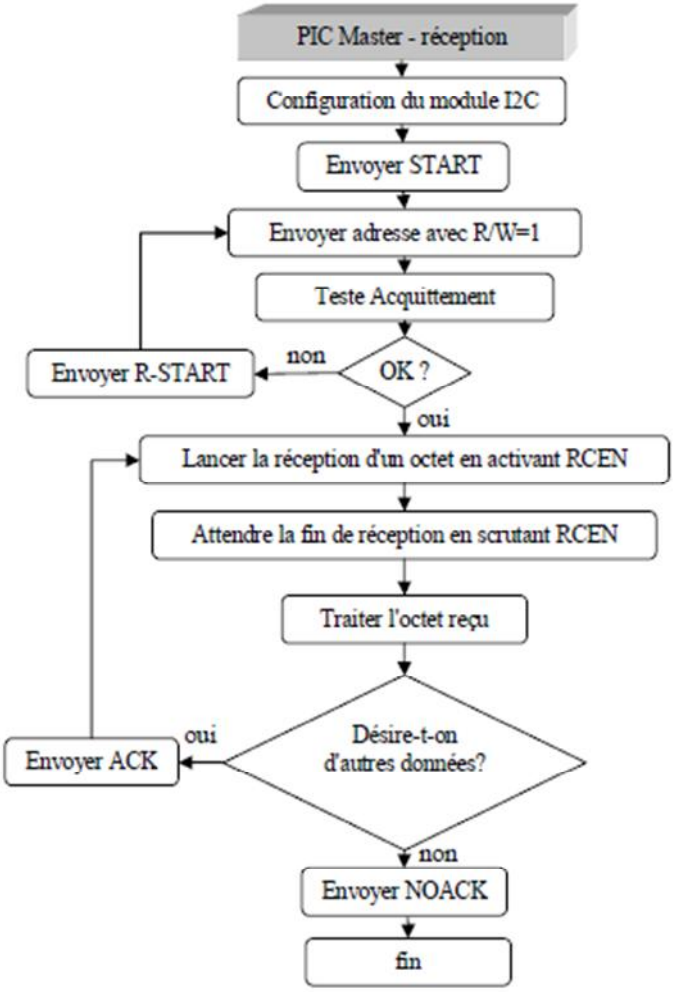
A la fin de la réception, le module I2C master ramène le bit SSPCON2.RCEN a 0, transfert l'octet reçu dans SSPBUF, l'indicateur BF passe a 1 ainsi que le drapeau d'interruption PIR1.SSPIF. Ensuite Il force l'horloge a 0 (pause) pour donner le temps au programme d'envoyer l ' Acknowledge,

• Le programme utilisateur du master vide le buffer SSPBUF, BF repasse a 0. Ensuite il envoie l'Acknowledge en plaçant ACK/NOACK (0/1) dans la bit SSPCON2.ACKDT et en validant le bit SSPCON2.ACKEN. Le bit ACKDT est place sur la ligne SDA, et ACKEN est raz automatiquement. L'horloge est libérée juste le temps d'un coup d'horloge (le 9eme) pour que le slave puisse prendre l ' Acknowledge en compte. Elle est remise en pause après. Le drapeau d'interruption SSPIF est aussi positionne après l'envoi du ACK

• Si on a envoyé ACK, ca veut dire qu'on veut recevoir d'autres données, alors notre programme peut initier la réception d'un nouvel octet en validant SSPCON2.RCEN.

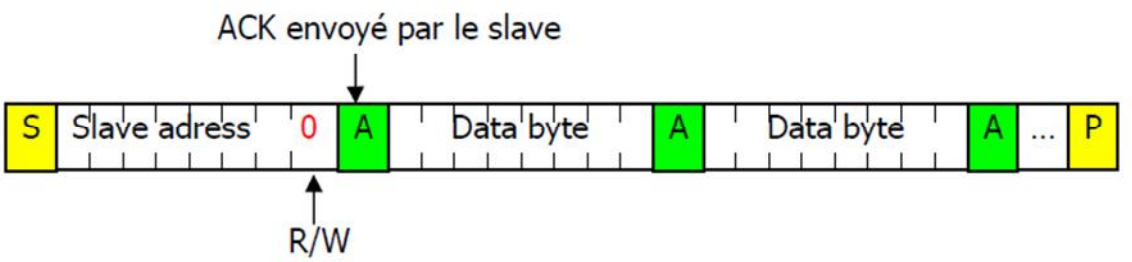
• A la réception du ACK, le slave met l'horloge en mode pose pour donner le temps a son programme de placer une donnée dans le registre SSPBUF et de libérer ensuite l'horloge a l'aide du bit CKP.

• Si on a envoyé NOACK, (on ne veut plus de donnée), Le slave qui le reçoit réinitialise sa logique comme s'il a reçu un STOP.



MSCP en mode I2C Slave

Slave en Réception de données (venant du master)



• Le PIC détecte le START sur la ligne SDA, il positionne le bit SSPSTAT.S et se prépare à recevoir l'adresse,

• Le PIC détecte la fin de réception de l'adresse, il vérifie si elle correspond a son adresse.

• Si c'est l'adresse d'un autre slave, il ne réagit pas, il se met à attendre le stop bit,

• Si l'adresse reçue correspond à celle de notre PIC :

Le LSB R/W est recopie dans SSPSTAT.R_W

Le bit SSPSTAT.D_A passe a 0 ® c'est une adresse qui est arrivée la suite dépend de l'état de BF et SSPOV (voir tableau ci-dessus pour plus de détails):

Si BF=0 et SSPOV=0 (tout est OK) SSPSR (l'adresse) est transféré dans SSPBUF

BF passe à 1

Un ACK (0) est place sur la ligne SDA pendant le 9eme coup d'horloge. C'est le PIC qui le fait, notre programme ne s'en occupe pas

Le drapeau PIR1.SSPIF passe a 1 ce qui peut déclencher une interruption. Il doit être raz par soft.

Si BF=1 (Buffer SSPBUF plein)

L'adresse ne sera pas transférée dans SSPBUFSSPOV sera positionne

L'ACK ne sera pas transmis (NoAck)

Le drapeau PIR1.SSPIF passe à 1 ce qui peut déclencher une interruption, Il doit être raz par soft

Si BF=0 et SSPOV=1 (Buffer SSPBUF vide mais il y'a eu un overflow avant)

SSPSR (l'adresse) est transférée dans SSPBUF

SSPOV reste positionne

L'ACK ne sera pas transmis (NoAck)

Le drapeau PIR1.SSPIF passe a 1 ce qui peut déclencher une interruption

Status Bits as Data Transfer is Received		SSPSR → SSPBUF	Generate $\overline{\text{ACK}}$ Pulse	Set bit SSPIF (SSP Interrupt occurs if enabled)
BF	SSPOV			
0	0	Yes	Yes	Yes
1	0	No	No	Yes
1	1	No	No	Yes
0	1	Yes	No	Yes

- Dans le cas "tout est OK", notre programme doit :

  Vider SSPBUF pour que BF passe a 0 et qu'on soit pret pour la suite.
  On vérifie le bit R_W, on trouve 0, on sait qu'on va recevoir un octet

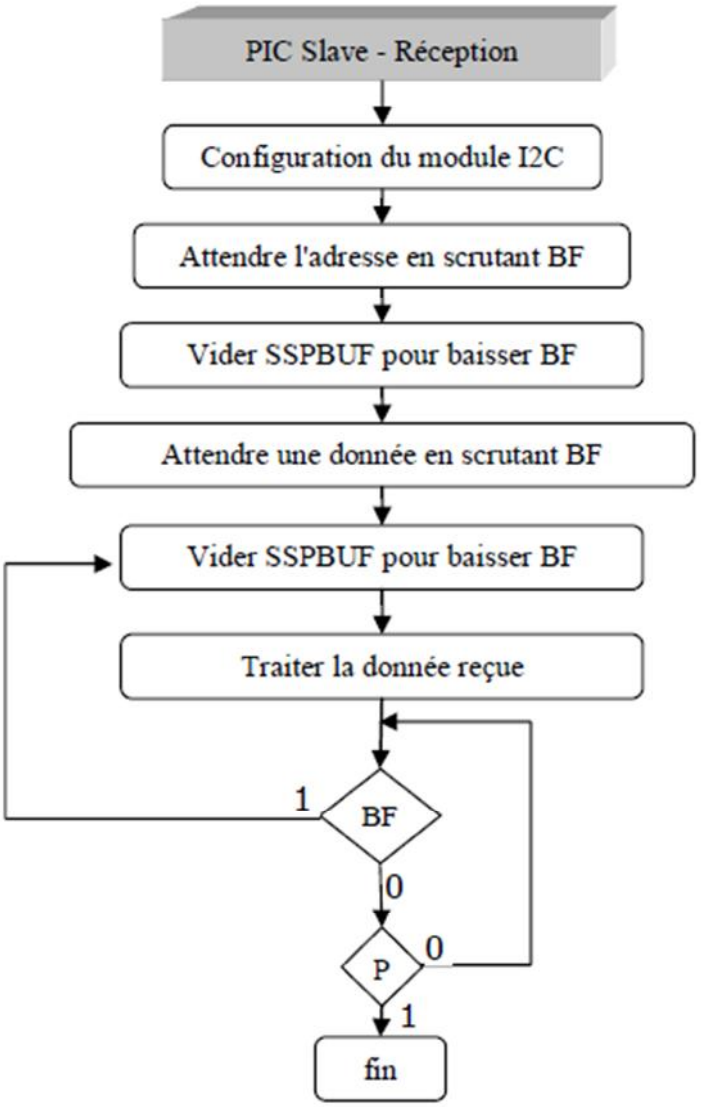
- Le PIC attend la réception du 8eme bit, vérifie BF et SSPOV (on suppose que tout est OK), il transfert l'octet arrive dans SSPBUF, BF et D_A passent a 1.

- Le PIC envoie l'acquittement pendant le 9eme bit et positionne SSPIF à 1.

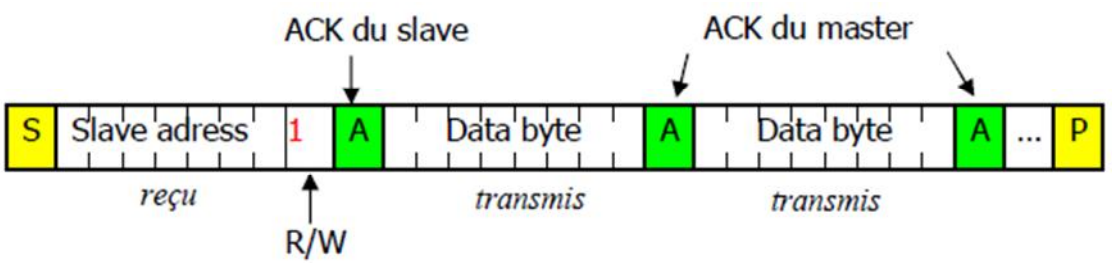
- Notre programme vide SSPBUF, BF repasse à 0

- Maintenant on peut soit recevoir, une autre donnée, soit un stop condition, soit un

Repeat_Start_Condition. Il faut donc avoir l'œil, sur les indicateurs, BF, SSPIF, P et S pour décider de la suite des opérations



Slave en transmission de données (vers le master)



Remarquons que le slave en transmission ne fait que préparer la donnée a transmettre dans son buffer SSPBUF. C'est le master qui décide du début de l'échange puisque c'est lui qui gère l'horloge. Toutefois, le slave peut bloquer un échange en bloquant l'horloge a 0.

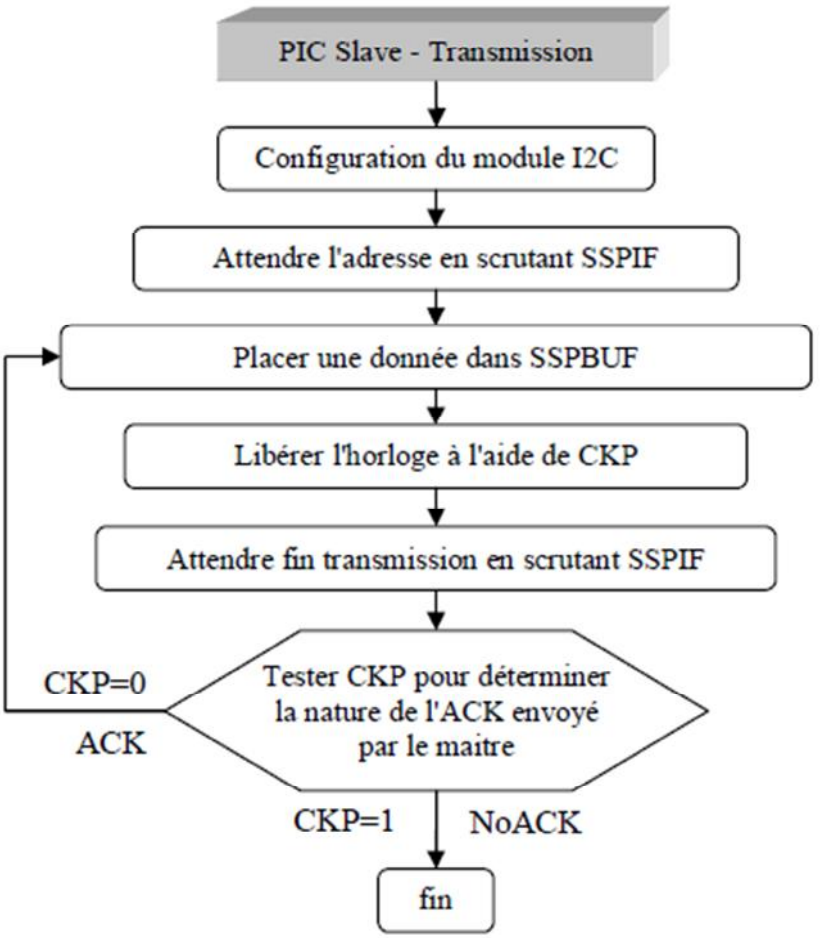
Clock Stretching.

Le drapeau BF passe à 1 quand le programme utilisateur écrit un octet dans SSPBUF et revient à 0 après la transmission du 8 bit de cet octet. Quand au début de l'échange, on reçoit l'adresse, celle-ci est transférée dans SSPBUF et BF reste à 0 dans ce cas.

Le PIC détecte le START bit sur la ligne SDA, positionne le bit SSPSTAT.S et se prépare à recevoir l'adresse,
Le PIC détecte la fin de réception de l'adresse, il vérifie si elle correspond a son adresse.
Si c'est l'adresse d'un autre slave, il ne réagit pas, il se met a attendre le stop condition,
Si l'adresse reçue correspond a notre PIC. Le module I2C Slave recopie le bit R/W (1 dans ce cas) dans le bit R_W du registre SSPSTAT, puis vérifie les bits BF et SSPOV pour savoir si le contenu précédent du registre SSPBUF a été lu ou non
Supposons que tout est OK, BF=0 et SSPOV=0
Il recopie SSPSR (l'adresse) dans SSPBUF
BF ne passe pas à 1
Il place un ACK (0) sur la ligne SDA pendant le 9eme coup d'horloge. C'est le PIC qui le fait, notre programme ne s'en occupe pas
Il place le bit CKP à 0 ce qui force l'horloge a 0. (Mode pause : Clock Stretching)
Le drapeau PIR1.SSPIF passe à 1 ce qui peut déclencher une interruption, il doit être raz par soft
Le bit SSPSTAT.D_A passe a 0 ▫ c'est une adresse qui est arrivée

· Notre programme détecte que SSPIF est passe a 1 :

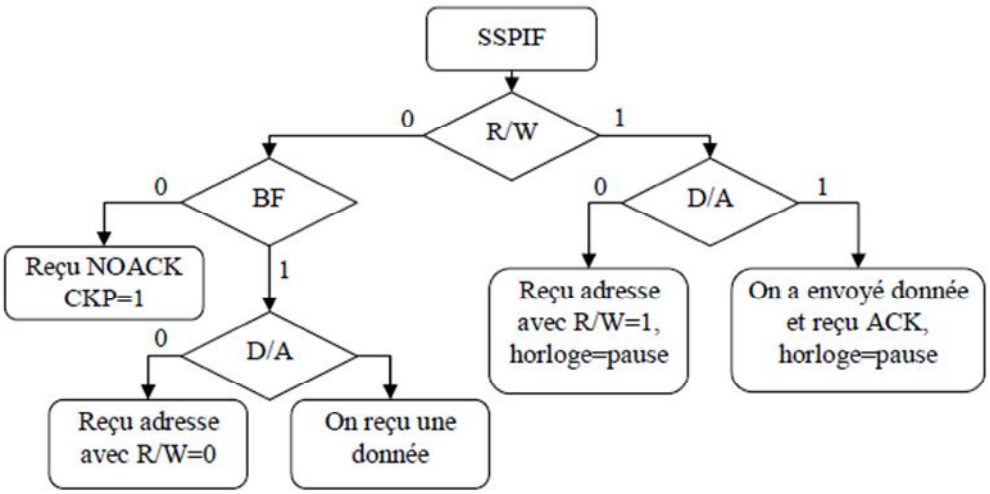
Si on vérifie le bit R_W, on trouve 1, on sait qu'on va envoyer un octet
On n'a pas besoin de lire SSPBUF car BF est égal à 0
On copie un octet de donnée dans SSPBUF. L'indicateur SSPSTAT.BF passe a 1
On replace le bit CKP à1 pour libérer l'horloge. C'est le moyen d'informer le master que la donnée est prête.
Le master détecte, la libération de l'horloge, envoie une rafale de coup d'horloge, l'octet est transmis. L'indicateur SSPSTAT.BF repasse à 0 après le 8eme
Maintenant le master peut renvoyer soit un ACK, soit NoACK :
Si c'est un ACK, le PIC (slave) positionne le drapeau SSPIF et raz le bit CKP pour mettre l'horloge en pause. Ceci donne au programme du slave le temps de préparer une nouvelle donnée dans SSPBUF et libérer l'horloge ensuite. Si le master envoie un STOP après le ACK, le slave ne le détecte pas.
Si c'est un NOACK, le PIC (slave) positionne le drapeau SSPIF, le bit CKP reste à 1 et le bit R_W passe à 0. L'électronique du slave est réinitialisée, il est prêt à recevoir un nouveau START. Normalement, le master n'a pas besoin d'envoyer un STOP dans ce cas. S'il le fait quand même, le slave le détecte et positionne l'indicateur P.



Utilisation des interruptions

Voici les 5 événements qui déclenchent l'interruption SSPI sur un PIC Slave.

Evénement	R/W	D/A	BF	CKP	Action
On a reçu une adresse avec R/W=0	0	0	1	NU	Il faut vider SSPBUF pour baisser le drapeau BF et attendre une donnée
On a reçu une donnée	0	1	1	NU	Il faut vider SSPBUF et attendre un événement (qui peut être un STOP)
On a reçu une adresse avec R/W=1. L'horloge est forcée en pause	1	0	0	0	Il faut placer une donnée dans SSPBUF et libérer l'horloge en positionnant le bit CKP
On a fini la transmission d'un octet et reçu un ACK. L'horloge est forcée en pause	1	1	0	0	Il faut placer une donnée dans SSPBUF et libérer l'horloge en positionnant le bit CKP
On dirait qu'on a reçu une donnée mais le buffer reste vide et l'horloge n'est pas pausée: anormal. C'est un NOACK qui a été reçu	0	1	0	1	La séquence d'échange est finie


SSPIF
R/W
0
1
BF
0
1
D/A
0
1
Reçu NOACK
CKP=1
D/A
0
Reçu adresse
avec R/W=1,
horloge=pause
On a envoyé donnée
et reçu ACK,
horloge=pause
Reçu adresse
avec R/W=0
On reçu une
donnée