N° Ordre ..... /Faculté/UMBB/2016

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENTSUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE M'HAMED BOGARA-BOUMERDES

Faculté des Sciences

Thèse de Doctorat-LMD

Présentée par

DICHOU Karima

Filière : Génie électrique

Option **:** Systèmes électroniques Avancés.

Contribution à l'étude des cartes à puce avancées

Devant le jury :

ACHELI	Dalila	Prof	UMBB	Présidente
TOURTCHINE	Victor	Prof	UMBB	Encadreur
SOUISSI	Boularbah	MCA	USTHB	Examinateur
SMAALI	Kacem	MRA	CDTA	Examinateur
RAHMOUNE	Fayçal	Prof	UMBB	Invité

Année Universitaire : 2015/2016

« Le succès n’est pas la clé du bonheur. Le bonheur est la clé du succès. Si vous aimez ce que vous faites, vous réussirez »

Albert Schweitzer

Remerciement

Je remercie dieu le tout puissant de nous avoir donné la santé et la volonté d'entamer et de terminer cette thèse.

Tout d'abord, ce travail ne serait pas aussi riche et n'aurait pas pu avoir le jour sans l'aide de mon directeur de thèse ***Pr. Victor TOURTCHINE****, je le remercie pour son encadrement exceptionnel, pour sa patience, sa rigueur et sa disponibilité durant toutes les années de la thèse.*

Je suis consciente de l'honneur que m'ont fait ***Pr. Fayçal RAHMOUNE*** *et* ***Mr. Yacine MERAIHI*** *pour leurs aide et encouragements.*

Mes remerciements s'adressent également à toute l'équipe de formation ***Infotronique*** *pour sa générosité et la grande patience dont ils ont su faire preuve malgré leurs charges académiques et professionnelles en particulier, le chef de la structure le* ***Dr. Karim BAICHE****.*

Mes profonds remerciements vont également à toutes les personnes qui m'ont aidés et soutenue de près ou de loin.

__ A tous ceux que j'aime…

Résumé :

Parmi les applications les plus spectaculaires de l'électronique à la fin du XXe siècle, c'est la carte à puce qui concerne chacun d'entre nous. Elle est devenue l'un des moyens les plus utilisés dans notre vie quotidienne. Elle a désormais fait ses preuves dans de nombreux secteurs : médical, banques, carte étudiant et systèmes de vote électroniques …etc.

L'évolution de la technologie a permet la conception de cartes à puce de plus en plus performantes et sécurisées ainsi, l'implémentation d'applications est devenue plus rapide mais leurs coûts augmentent aussi.

Le choix de la carte à puce qui convient le mieux à une application donnée est devenu une tâche difficile, en particulier pour celles destinées à un grand nombre d'utilisateurs. Il faut réduire les coûts tout en assurant le bon fonctionnement.

Notre démarche consiste à programmer et simuler les protocoles de communication les plus connus des cartes à puce (RS 232, I^2C et T=0) en langage assembleur pour réduire l'espace mémoire utilisé sur la carte. Ces protocoles représentent la base de n'importe quelle application faite avec carte à puce.

Au cours de ce travail, nous avons exploité deux applications de la carte à puce : la carte à puce destinée à l'étudiant et le système de vote électronique. Notre objectif est de les améliorer tout en optimisant les ressources utilisées. Cela, afin de réduire les coûts et assurer leurs sécurités et fiabilités.

Nous avons abordé la problématique des coûts très élevés des cartes à puce destinées aux étudiants en proposant un système pas cher et optimisé en ressource matériel et en espace mémoire et qui assure les mêmes fonctionnalités que les systèmes conventionnels.

Concernant les systèmes de vote électroniques, il existe deux types : le premier est basé sur la visite d'un bureau de vote et le deuxième se fait à distance. Chacun a des avantages et des inconvénients. La première partie de la contribution consiste à améliorer le système de vote électronique qui se fait dans les bureaux de vote en rendant ce système rapide en manipulation en utilisant une carte à puce et en assurant la sureté des résultats.

La deuxième partie de cette contribution consiste à fusionner les deux systèmes de vote électroniques existant, en prenant en compte les avantages de chacun d'entre eux, pour concevoir un nouveau qui à la fois rapide, sûr et économique que les systèmes conventionnels.

Mots-clés : *Cartes à puce, lecteur de carte à puce, microcontrôleur, Protocole de communication, cryptographie, biométrie, carte étudiant, vote électronique.*

Abstract:

Among the most spectacular applications of electronics in the late twentieth century, the smartcard concerns all of us. It became one of the most used means in everyday life. Now, it has proven in many sectors: medical, bank, student card and electronic voting systems ... etc.

Technology evolution has enabled the design of smart cards increasingly efficient and secure way so, applications implement became faster but costs also increase.

The choice of the best suits smart card to a given application become a difficult task, especially for those for many users. We must reduce costs while ensuring the proper functioning.

Our approach is to program and simulate the most popular communication protocols for smart cards (RS232, I²C and T = 0) in assembly language to reduce the used memory on the card. These protocols are the basis of any application made with smart card.

In this work, we exploited two applications of the smart card: Student smart card and the electronic voting system. Our objective is the improved them while optimizing the used resources. This, to reduce costs and ensure their safety and reliability.

We addressed the issue of the high costs of student smart cards by offering a cheaper and optimized system in hardware resource and memory and providing the same functionality as conventional systems.

Concerning electronic voting systems, there are two types: the first one based on the visit a polling station and the second is done remotely. Each has advantages and disadvantages. The first part of the contribution is to improve the electronic voting system of the polling stations, making it fast handling system using a smart card and ensuring the safety of operations.

The second part of this contribution is to merge the two existing systems of electronic voting, taking into account the benefits of each of them, to design a new one that is fast, safe and economical than conventional systems.

Keywords: *Smart card, Smart card reader, microcontroller, communication Protocol, cryptography, biometry, student card, electronic voting.*

ملخص:

من بين التطبيقات الأكثر إثارة للإلكترونيات في أواخر القرن العشرين، البطاقة الذكية تخص كل واحد فينا. لقد أصبحت أحد الوسائل الأكثر استخداما في حياتنا اليومية. لقد أثبتت وجودها الأن في الكثير من المجالات: الطبية، البنوك، بطاقة الطالب وأنظمة التصويت الإلكترونية ... الخ.

مكّن تطور التكنولوجيا من تصميم بطاقات ذكية أكثر فعالية وآمنة، وعليه أصبح تنفيذ التطبيقات أسرع ولكن التكاليف زادت أيضا.

أصبح اختيار البطاقة الذكية التي تناسب تطبيق معين مهمة صعبة، خصوصا بالنسبة للذي يستهدف الكثير من المستخدمين. يجب علينا خفض التكاليف مع ضمان حسن سير العمل.

نهجنا هو برمجه ومحاكاة بروتوكولات الاتصال الأكثر شعبية بالنسبة للبطاقات الذكية (RS232، I^2C وT=0) في لغة التجميع لتقليل الذاكرة المستخدمة على البطاقة. هذه البروتوكولات هي أساس أي تطبيق على البطاقة الذكية.

في هذا العمل، تطرقنا إلى تطبيقين للبطاقة الذكية: البطاقة الذكية للطالب ونظام التصويت الإلكتروني. هدفنا هو تحسينها مع التقليل من الموارد المستخدمة. هذا، من أجل خفض التكاليف وضمان سلامتهم وموثوقيتهم.

تناولنا مسألة ارتفاع تكاليف البطاقات الذكية الخاصة بالطلاب من خلال توفير نظام اقتصادي مع التقليل من الموارد المادية والمساحة التي تحتلها الذاكرة ويوفر نفس الوظائف مع الأنظمة التقليدية.

فيما يخص أنظمة التصويت الإلكترونية، هناك نوعان: الأول يقوم على أساس زيارة مركز الاقتراع ويتم ثاني عن بعد. لكل منها مزاياه وعيوبه. الجزء الأول من المساهمة يتمثل في تحسين نظام التصويت الإلكتروني الذي يتم في مراكز الاقتراع، مما يجعل النظام أسرع باستخدام البطاقة الذكية وذلك مع ضمان سلامة العمليات.

الجزء الثاني من هذه المساهمة هو دمج نظامي التصويت الإلكتروني، مع أخذ إيجابيات كل واحد منهما وذلك لتصميم نظام جديد، سريع، آمن واقتصادي في آن واحد مقارنة بالأنظمة التقليدية.

الكلمات الرئيسية : *البطاقات الذكية، قارئ البطاقة الذكية، ميكروكونترولر ، بروتوكول الاتصالات، والترميز، القياسات الحيوية، بطاقة الطالب، التصويت الالكتروني.*

Sommaire

Chapitre 2 : cryptographie sur cartes à puce

Chapitre 3 : Programmation et simulation des protocoles de communication utilisés par les cartes à puce

Chapitre 4 : Conception et réalisation d'une carte à puce destinée aux étudiants

Chapitre 5 : Conception d'une machine de vote électronique embarquée à base d'un microcontrôleur et utilisant une carte à puce

Liste des figures

Chapitre 1

Chapitre 2

Chapitre 3

Chapitre 4

Chapitre 5

Liste des tableaux

Chapitre 1

Chapitre 2

Chapitre 3

Chapitre 4

Chapitre 5

Liste des abréviations

Abréviation	Signification
3DES	Triple DES
AES	Advanced Encryption Standard
AFNOR	Association Française de Normalisation (France)
APDU	Application Protocol Data Unit
ATR	Answer To Reset
CLK	Clock
COS	Chip Operating System
CPU	Central Processing Unit
DES	Data Encryption Standard
DF	Dedicated File
ECC	Elliptic Curve Cryptography
EEPROM	Electrically-Erasable Programmable Read-Only Memory
EF	Elementary File
EMV	Europay, Mastercard, Visa
EPROM	Electrically Programmable Read-Only Memory
FID	File Identify
GND	Ground (supply voltage -)
GSM	Global System for Mobile communications
I/O	Input/Output
I²C	Inter Integrated Circuit bus
IC	Integrated Circuit
ID	Identity
ISO	International Standardization Organization
MF	Master File
MSB	Most Significant Bit
LSB	Less Significant Bit
NBS	National Bureau of Standard
NIST	National Institute of Standards and Technology
NSA	National Security Agency
OS	Operating System
PCB	Printed Circuit Board
PIN	Personal Identification Number
RAM	Random Access Memory
RF	Radio Frenquency
RFID	Radio Frenquency Identification
RFU	Reserved for Future Use
ROM	ROM
RSA	Rivest-Shamir-Adleman
RST	Reset
S-Box	Substitution Box
SCOS	Smart Card Operating System
SIM	Subscriber Identification Module
SW1	Status Word 1
SW2	Status Word
TPDU	Transport Protocol Data Unit
Vcc	Supply voltage (+)
Vpp	Programming voltage
RS232	Recommended Standard 232

Introduction Générale

De nos jours, la carte à puce ne cesse de conquérir de nouveaux marchés. Elle est considérée comme un élément essentiel et incontournable de la vie quotidienne. Ceci est dû non seulement au fait que l'on puisse la transporter, mais aussi au niveau de la sécurité élevé qu'elle assure. Elle est l'appareil ultime pour transporter des données raison de la sécurité qu'elle assure.

L'évolution de la technologie a permet la conception de cartes à puce de plus en plus performantes et sécurisées ainsi, l'implémentation d'applications est devenue plus rapide mais leurs coûts augmentent aussi. Le choix de la carte à puce qui convient le mieux à une application donnée est devenu une tâche difficile, en particulier pour celles destinées à un grand nombre d'utilisateurs. Il faut réduire les coûts tout en assurant le bon fonctionnement. Il serait donc crucial pour les développeurs d'applications de s'interroger sur les performances des cartes à puce utilisées tout en réduisant les couts.

Notre démarche consiste à programmer et simuler les protocoles de communication les plus connus des cartes à puce (RS 232, I²C et T=0) en langage assembleur pour réduire l'espace mémoire utilisé sur la carte. Ces protocoles représentent la base de n'importe quelle application faite avec carte à puce.

Au cours de ce travail, nous avons exploité deux applications de la carte à puce : la carte à puce destinée à l'étudiant et le système de vote électronique. Notre objectif est de les améliorer tout en optimisant les ressources utilisées. Cela, afin de réduire les coûts et assurer leurs sécurités et fiabilités.

Nous avons abordé la problématique des coûts très élevés des cartes à puce destinées aux étudiants en proposant un système pas cher et optimisé en ressource matériel et en espace mémoire et qui assure les mêmes fonctionnalités que les systèmes conventionnels.

Concernant les systèmes de vote électroniques, il existe deux types : le premier est basé sur la visite d'un bureau de vote et le deuxième se fait à distance. Chacun a des avantages et des inconvénients. La première partie de la contribution consiste à améliorer le système de vote électronique qui se fait dans les bureaux de vote en rendant ce système rapide en manipulation en utilisant une carte à puce et en assurant la sureté des résultats.

La deuxième partie de cette contribution consiste à fusionner les deux systèmes de vote électroniques existant, en prenant en compte les avantages de chacun d'entre eux, pour concevoir un nouveau qui à la fois rapide, sûr et économique que les systèmes conventionnels.

La thèse est organisée comme suite :

Chapitre 1 : Dans ce chapitre nous faisons le point sur le concept de carte à puce, son évolution et ses différents types. Cela inclus les standards internationaux et leurs caractéristiques physiques et électriques. Nous décrivons également sa communication avec lecture.

Chapitre 2 : L'objectif de ce chapitre est d'expliquer comment les nouvelles applications bénéficient de stocker des informations en toute sécurité sur une carte à puce. Nous introduisons quelques notions sur la cryptographie moderne ainsi que son historique. Quelques détails sur les différentes implémentations de l'algorithme cryptographique DES sur FPGA ont été présentés.

Chapitre 3 : Dans ce chapitre, nous avons proposés des techniques afin de programmer et simuler les deux protocoles de communication les plus connus des cartes à puce à contacts : le protocole RS 232 et le protocole I²C ainsi que le protocole de transmission, le plus utilisé jusqu'à

présent, T= 0. Cela pour une meilleure compréhension de ces échanges. En plus de ça, nous avons optimisé l'espace mémoire utilisé en programmant ces protocoles en langage assembleur.

Chapitre 4 : Dans ce chapitre, Nous avons abordé la problématique des coûts très élevés des cartes à puce destinées aux étudiants. Notre objectif est d'améliorer cette application tout en optimisant les ressources utilisées (matériel et espace mémoire). Nous présentons la conception et la réalisation d'une carte à puce, destinée aux étudiants, qui servira non seulement à faciliter le travail de l'administration mais aussi à sauvegarder toutes les informations de l'étudiant durant ses études universitaires.

Chapitre 5 : Après avoir fait une recherche bibliographique sur les différents systèmes de vote électronique existants, dans ce dernier chapitre nous proposant l'élaboration d'un nouveau système de vote électronique embarqué utilisant une carte à puce. Nous entamons la description, la programmation et la simulation du fonctionnement de notre système amélioré.

Chapitre 1

Les cartes à puce

1.1. Introduction

Depuis son apparition, la carte à puce ne cesse de conquérir de nouveaux marchés tout en gagnant la confiance du consommateur. Elle est considérée aujourd'hui comme un élément essentiel et incontournable de la vie de tous les jours. Ceci est dû non seulement au fait que l'on puisse la transporter, mais aussi au niveau de sécurité élevé qu'elle assure.

Dans ce chapitre nous faisons le point sur le concept de carte à puce, son évolution et ses différents types. Cela inclus les standards internationaux et leurs caractéristiques physiques et électriques. Nous décrivons également sa communication avec lecture.

1.2. Qu'est-ce qu'une carte à puce ? [1]

Une carte à puce (ou carte à microcircuit) est une carte en matière plastique. L'innovation consiste à loger dans l'épaisseur habituelle de la carte, un ou plusieurs circuits intégrés et un connecteur plat capable d'assurer la liaison électrique avec un « lecteur » spécial.

1.3. Evolution De La Carte A Puce [2-5, 10]

Dès 1967et 1968 deux ingénieurs allemands : Jürgen Dethloff et Helmut Grötrupp inventèrent la carte à puce. Ils déposèrent d'ailleurs un brevet à ce sujet dès 1969. Parallèlement à cela, le Japonais Kunitaka Arimura de l'institut Armura Technology déposait au Japon un brevet relatif à la carte à puce. En mars 1970 et l'année suivante, Paul Castrucci de chez IBM déposait à son tour, mais aux Etats-Unis cette fois, un brevet intitulé Information Card. Plus tard, entre 1974 et 1979, le français Roland Moreno déposait 47 brevets relatifs à la carte à puce et à ses diverses applications possibles dans 11 pays différents ; fondant ensuite la société Innovatron pour le développement et l'exploitation de ces brevets et applications. La véritable « sortie publique » de la carte à puce eut lieu en 1983 avec les premières cartes de paiement téléphonique de France Télécom (cartes à mémoires) (Voir la figure 1.1).

Fig.1. 1: La première véritable application de la carte à puce a été la télécarte.

Paradoxalement, ce n'est que trois ans plus tard, soit en 1987, qu'étaient publiées les premières normes relatives à la carte à puce. Le tableau 1.1 réalise la synthèse de l'évolution des cartes à puce.

Année	Evénement
1979	Première carte à puce fabriquée par Motorola pour Bull CP8.
1980-1981	Premières expérimentations de télévision payante.
1983	Première cartes à puce téléphonique France Télécom.
1984	Première version de la carte bleue à puce à base de carte Bull CP8.
1987	Publication des normes ISO 7816.

1989	Premières cartes GSM pour téléphones mobiles (Gemplus).
1998	Premières cartes à puce programmables en Java ou « Java Cards ».

Tab.1. 1: Quelques années significatives de l'évolution de la carte à puce.

1.4. Marche des cartes à puce [3-8]

Aujourd'hui on trouve plus de 5 milliards de cartes. Les domaines d'utilisation des cartes à puce sont de plus en plus vastes, on les trouve dans :

- **Monétique** : Carte bancaire, Porte-monnaie.
- **Identification** : cartes d'identité nationales, passeport, passeport biométrique.
- **Enseignement** : carte d'étudiant et/ou de restauration.
- **Téléphonie mobile** : carte SIM.
- **Secteur médical** : carte Vitale en France, chifa en Algérie.
- **Titre de transport.**
- **Sécurité informatique** (authentification forte et signature électronique).

1.5. Les différents types de cartes [2, 5]

Les cartes à puce se distinguent surtout les unes des autres par les fonctionnalités de leur circuit intégré interne. Il existe différents types de cartes à puce, on trouve les cartes avec contact (cartes à puce classique) (voir la figure 1.2), et les cartes à puce sans contact (voir la figure 1.3), et dans certain cas mixtes (avec et sans contact).

Fig.1. 2: Une carte avec contact.

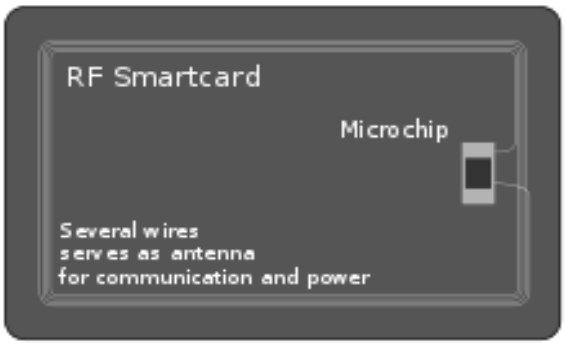


Fig.1. 3: Une carte sans contact.

1.5.1. Cartes à contact [1, 2, 5]

On peut distinguer deux grandes familles de cartes à puce à contact :

- Les cartes à mémoire ;
- Les cartes à microprocesseur.

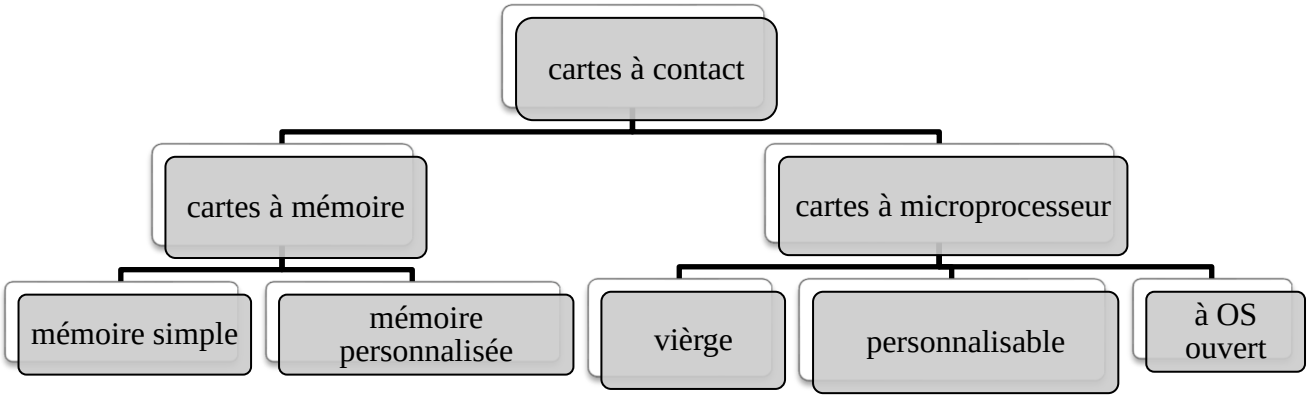


Fig.1. 4: Différents types de cartes.

1.5.1.1. Les cartes à mémoire [1, 2, 10]

La carte à mémoire appelée aussi carte synchrone en raison de son protocole de dialogue. On trouve :

- **les cartes à mémoire simple [1-5]**

Comme son nom l'indique, une carte à mémoire simple renferme une certaine quantité de mémoire, sans aucune protection particulière. Cela signifie que n'importe qui peut y lire ou y écrire des informations.

La plupart des cartes à mémoire simple sont réalisées en technologie EEPROM, et son donc recyclables (effaçables et réinscriptibles). Leur capacité est généralement de l'ordre de quelques K bits, voire quelques dizaines. La figure 1.5 représente la Synoptique interne d'une mémoire simple.

Ces cartes sont en principe destinées à des applications n'ayant pas à être sécurisées : fichier portable non secret, cartes de suivi de production…

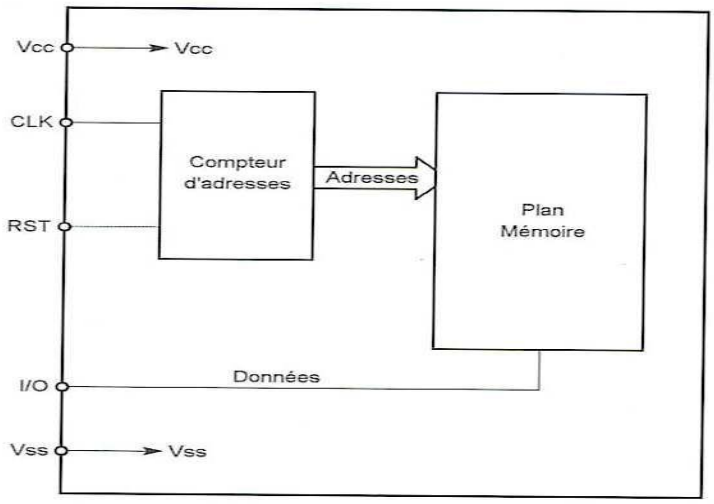


Fig.1. 5: Synoptique interne d'une mémoire simple [2].

- **Les cartes à mémoire personnalisée [1-5]**

En revanche, pour mériter l'appellation de « carte à mémoire personnalisée », une carte à puce doit contenir au moins l'un des quatre systèmes de protection suivants, réalisés en logique câblée sans le secours d'un microprocesseur :

- Zone protégée en écriture après destruction d'un fusible ;
- Zone protégée en lecture et écriture par un « code porteur » (code confidentiel dit « PIN » ou Personal Identification Number) ;
- Blocage de la carte au bout d'un nombre donné de présentations d'un PIN erroné ;
- Protection par un « code émetteur » (l'émetteur étant l'organisme qui délivre les cartes et décide de leur contenu).

Deux technologies coexistent dans cette famille : EPROM ineffaçable puisque la puce est enfermée dans un enrobage opaque aux ultraviolets, et EEPROM effaçable puis reprogrammable. Les capacités disponibles sont généralement inférieures à 1 K bit. La figure 1.6 représente la Synoptique interne d'une carte mémoire personnalisée.

Ces cartes sont un peu plus perfectionnées d'envisager des applications exigeant à la fois une meilleure sécurité et de fréquentes mises à jour d'informations : cartes prépayées rechargeable, porte-monnaie électronique simple, contrôle d'accès, dossier portable sécurisé, carte d'abonnement, cartes de fidélité.

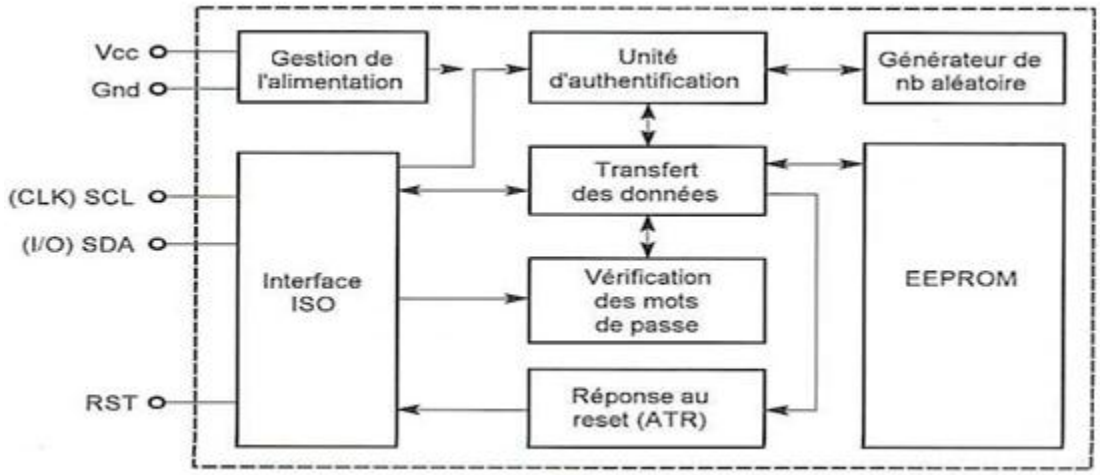


Fig.1. 6: Synoptique interne d'une carte mémoire personnalisée [2].

1.5.1.2. Les cartes à microcontrôleur [1-5, 10]

La carte à microcontrôleur (dite aussi à microprocesseur ou bien à microcalculateur), appelées aussi cartes asynchrones en raison de leur protocole de dialogue.

Ces cartes renferment un microcontrôleur complet ; c'est à dire l'association en un seul circuit d'une unité centrale de microprocesseur, de mémoire morte, de mémoire vive, de mémoire EEPROM, d'une interface d'entrée/sortie série et de toute la logique nécessaire pour faire fonctionner tout cela. La figure 1.7 représente la Synoptique interne d'une carte à microcontrôleur.

Produit informatique à part entière, elle contient plusieurs systèmes de protection :

- Zone protégée en écriture ou en écriture et lecture par un code secret émetteur, ceci après personnalisation par l'émetteur ;
- Zone protégée en lecture et écriture par un code secret porteur (PIN) ;
- Blocage de la carte après présentation d'un nombre donné de codes secrets erronés, mais avec possibilité de réhabilitation par l'organisme émetteur ;
- Mise en œuvre d'algorithmes cryptographiques (par exemple DES ou RSA) pour assurer la sécurité des transferts de données.

L'unité centrale utilisée varie selon la fonction ou la vocation de la carte. On peut trouver de simples microcontrôleurs 8 bits des familles PIC de Microchip ou AVR d'Atmel mais aussi des processeurs beaucoup plus puissants, associés parfois à un crypto processeur comme dans certaines carte de décryptage télévision récentes.

Ces cartes conviennent aux applications les plus « sensibles en sécurité» : carte bancaire, carte santé, carte de TV à péage, contrôle d'accès sécurisé, mono-service ou multiservices. On distinguer trois catégories différentes de cartes asynchrone :

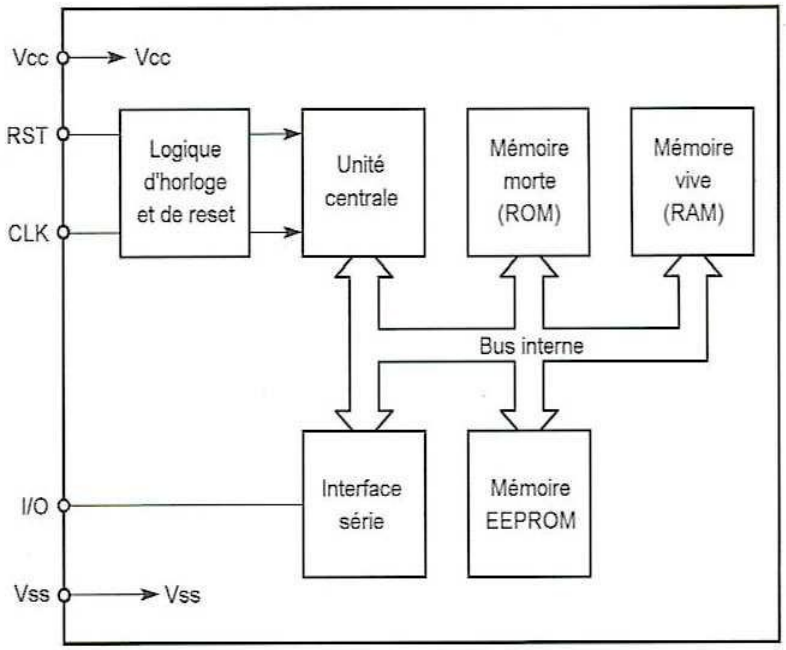


Fig.1. 7: Synoptique interne d'une carte à microcontrôleur [2].

- **Les cartes à puce "vierges" ou "spécifiques" [1, 2, 5]**

La mémoire de programme du microcontrôleur ne contient rien lorsqu'on les achète. C'est donc à nous d'écrire l'intégralité de leur OS (operating system). Cela demande beaucoup de travail mais permet de disposer de cartes réellement "sur mesure". Les cartes Gold, Silver, Fun ou bien encore Jupiter font partie de cette famille.

- **Les cartes à puce dites "personnalisables" [1, 2, 5]**

La plupart des cartes à microprocesseur sont supportées par un puissant système d'exploitation appelé, par exemple, « COS » (Chip Operating System ou Card Operating System).

Il s'agit d'un logiciel « masqué » en ROM et dont les caractéristiques sont les suivantes :

- Organisation de la mémoire en zone banalisées, protégées ou non ;
- Gestion dynamique de l'espace mémoire ;
- Gestion des codes confidentiels ;
- Changement éventuel de sous-programmes spécifiques à l'application, lors de la personnalisation.

Ces cartes permettent de développer très rapidement une application quasiment sans programmer.

- **Les cartes à puce dites "à OS ouvert" [1, 2, 5]**

C'est en quelque sorte une carte à puce "non terminée", c'est à dire une carte dans laquelle vous allez pouvoir programmer votre propre interpréteur de commande, et donc votre propre jeu d'instructions et vos propres mécanismes de gestion de fichiers.

Dans ces cartes, le fabricant a programmé un interpréteur de P-code ; ce P-code provenant lui-même de la compilation de langage évolué tel que Java dans la Java Card ou Basic dans la Basic Card. Comme pour les cartes à puce vierges, on peut ainsi réaliser une application "sur mesure" mais avec une programmation plus facile car elle est réalisée en langage évolué et non en langage machine. En outre, l'interpréteur de P-code prend en charge tous les protocoles de dialogue de bas niveau que l'on n'a pas ainsi à programmer. Tout ceci est synthétisé visuellement sur la figure 1.8 dans le cas d'une carte à puce de ce type.

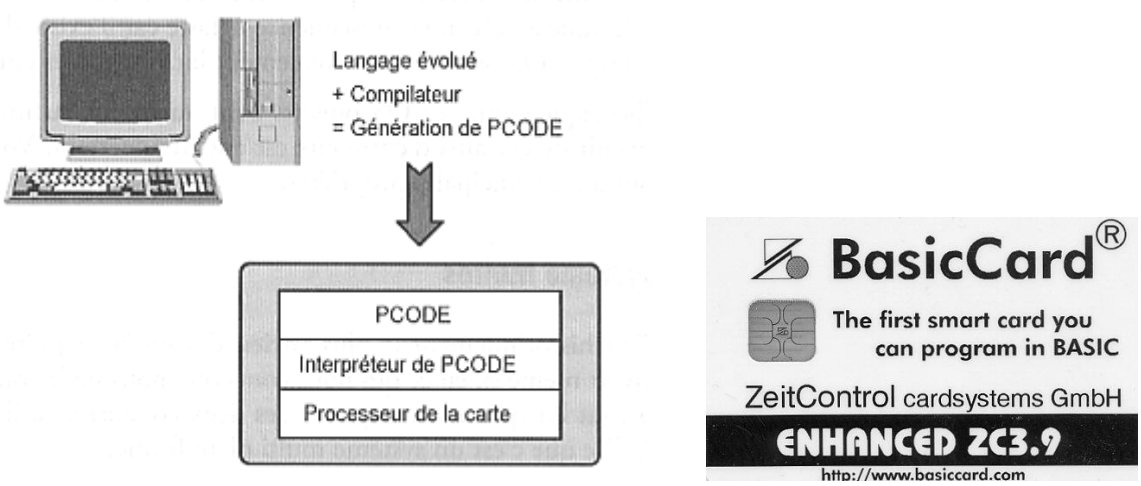


Fig.1. 8: Principe des cartes à puce dites "à OS ouvert" (exp. Basic Card) [5].

Une telle approche offre de multiples avantages que l'on peut résumer de la manière suivante :

- La carte est réellement programmable en fonction des besoins de l'application, comme lors d'une programmation par masque.
- La carte se programme en langage évolué, Java, C, Basic, accessible au commun des mortels et surtout indépendant du processeur contenu dans la carte.
- Les contraintes de programmation en grande série, imposées par la programmation par masque, n'existent plus car les cartes de ce type se programment unitairement si nécessaire.
- L'application ainsi développée est portable puisqu'elle peut être exécutée, du moins en théorie, dans toute carte compatible du langage utilisé.

1.5.2. Cartes sans contact [2, 5]

Le principe « à contact » de la carte à puce traditionnelle constitue un handicap, et parfois même un vice rédhibitoire, pour bon nombre d'applications potentielles. Outre les problèmes de fiabilité que cela peut poser (pollution et usure des contacts, vulnérabilité des lecteurs), ce principe exclut d'emblée de nombreuses familles d'applications.

Débarrassée de ses contacts, la carte à puce peut gagner énormément en robustesse, et donc affronter les pires environnements : intempéries, eau de mer, hautes températures, produits chimiques… C'est particulièrement appréciable en milieu industriel et en extérieur (applications automobiles, marines ou sportives, par exemple).

Ce lecteur pouvant fort bien télé-alimenter la « puce » par induction, l'absence de pile dans le badge constitue un avantage déterminant sur le plan de la fiabilité et de la compacité.

Mais la technique à mettre en œuvre est délicate malgré la simplicité de son principe. Des procédés de modulation et de supervision forte élaborés sont nécessaires pour garantir le très haut degré de sûreté que l'on exige dans les applications visées.

1.6. Composition d'une carte

1.6.1. Carte à contact [1]

Une carte à puce est constituée de 3 éléments (voir figure 1.9) :

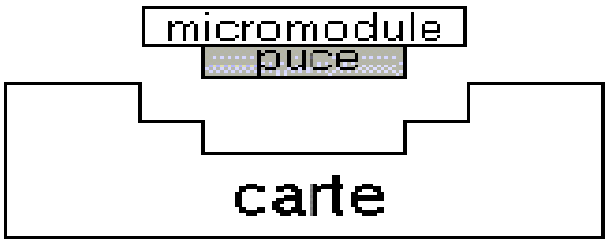


Fig.1. 9: Composition d'une carte.

- **La carte plastique**

Deux principaux types de plastique sont utilisés :

- Le PVC (Chlorure de PolyVinyle) non recyclable mais embossable.
- L'ABS (Acrylonitrille-Budadiène-Styrène) non embossable mais recyclable.

- **Le micro module**

Très mince circuit imprimé logé dans l'épaisseur de la carte qui accueille les contacts (visibles) du connecteur sur une face et la puce (cachée sous les contacts du micromodule) sur l'autre.

- **La puce électronique**

C'est un microcircuit (circuit intégré) construit à partir d'une galette de silicium.

1.6.2. Carte sans contact [2]

Les cartes à puce sans contact contiennent en plus d'une puce classique, une interface radiofréquence RFID (Radio Frequency Identification Devices) permettant la récupération des données à distance qui varie de 10cm à quelques mètres. La figure 1.10 illustre sa composition.

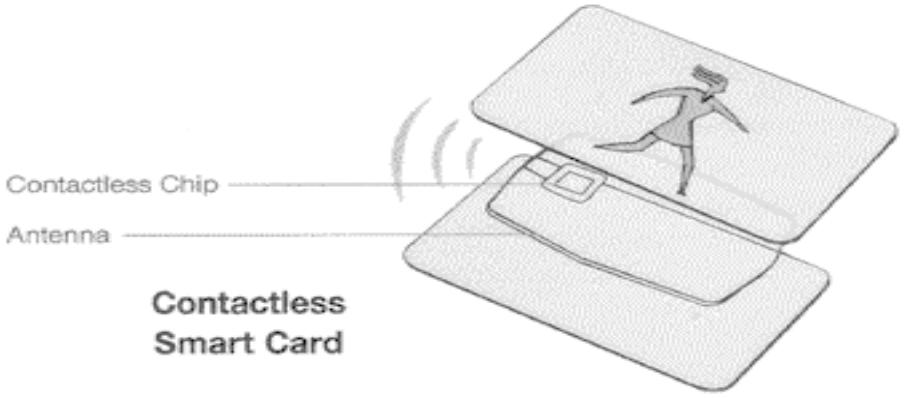


Fig.1. 10: Composition d'une carte à puce sans contact.

Elles sont alimentées par le lecteur à travers l'interface radiofréquence, leurs avantages c'est qu'elles sont plus rapide en manipulation, et subissent mois d'usure. Par contre, le transfert de données est plus long par rapport aux cartes avec contact, leurs coût est plus élevé, et elles ne sont pas sécurisé.

1.7. La standardisation des cartes à puce [2, 3, 5, 6]

Les cartes sont très standardisées car elles doivent être utilisables avec la gamme la plus large possible de lecteurs dans le monde entier. Pour garantir cette interopérabilité, la normalisation concerne au moins 3 points :

- Des paramètres physiques : taille de la carte, position de la puce et ses contacts.
- Des paramètres électriques : tension d'alimentation et niveaux électriques mise en œuvre ainsi que le brochage de la puce sur la carte.
- Des paramètres logiciels qui définissent le mode de dialogue avec la carte (commandes).

C'est la raison pour laquelle les caractéristiques des cartes à puce ont été fixées par des règles reconnues universellement qui appartiennent à une famille de standards et protocoles internationaux.

1.7.1. Les standards des cartes à puce à contact [3, 5]

Les normes principales des cartes à contact sont celles de l'ISO 7816. En plus de ces normes on trouve :

- ETSI, (télécommunications, GSM).
- EMV, (carte de paiement).
- ICAO, (agence de l'ONU, biométrie, passeport).
- …

ISO « International Organisation for Standardisation » est le plus important standard définissant les caractéristiques des cartes à puce qui fonctionnent avec un contact électrique. Sachant que 15 normes sont proposées pour les cartes à contact, nous décrivons brièvement ici uniquement les 4 principales normes.

- **ISO 7816-1**

Cette norme définit les caractéristiques physiques des cartes à puce à contact : la géométrie, la résistance, les contacts, etc.

- **ISO 7816-2**

Cette norme spécifie le dimensionnement physique (extérieur) des contacts de la puce. Deux des huit contacts réservés à une utilisation future (RFU) sont redéfinis pour l'utilisation USB dans la norme ISO 7816-12.

- **ISO 7816-3**

Cette norme définit l'interface électrique et les protocoles de transmission :

- Les protocoles de transmission (TPDU, Transmission Protocole Data Unit) : T=0 : protocole orienté octet, T=1 : protocole orienté paquet, T=14 : réservé pour les protocoles propriétaires.
- La sélection d'un type de protocole.
- La réponse à un reset (ATR, ou Answer To Reset en anglais) qui correspond aux données envoyées par la carte immédiatement après la mise sous tension.
- Les signaux électriques, tels que le voltage, la fréquence d'horloge et la vitesse de communication.

- **ISO 7816-4**

Cette norme vise à assurer l'interopérabilité des échanges. Elle définit les messages APDU (Application Protocol Data Units), par lesquels les cartes à puce communiquent avec le lecteur. Les échanges s'effectuent en mode client-serveur, le terminal ayant toujours l'initiative de communication.

1.7.2. Les standards des cartes à puce sans contact [3, 5]

Concernant les cartes à puce sans contact, on trouve les normes suivantes :

- **ISO 10 536 :** Couvent les cartes à puce à couplage très proche (quelques mm) ;
- **ISO 14 443 :** Couvrent les cartes à puce à couplage de proximité (quelque cm) ;
- **ISO 15 693 :** Couvent les cartes à puce à couplage éloigné ou voisinage (quelques dizaines de cm).

1.8. Caractéristiques physiques et électriques des cartes à puce à contacts [1, 2]

Même si nous savons à quoi ressemble physiquement une carte à puce, il n'est pas inutile de rappeler ici quelles sont ses caractéristiques matérielles principales que ce soit physique ou bien électriques.

1.8.1. Caractéristiques mécaniques [2]

Même si on connaît en général deux formats de la carte à puce, celui de la carte bancaire et celui de la carte SIM, 3 formats normalisés existent : ID1, ID00 et ID000 (voir la figure 1.11).

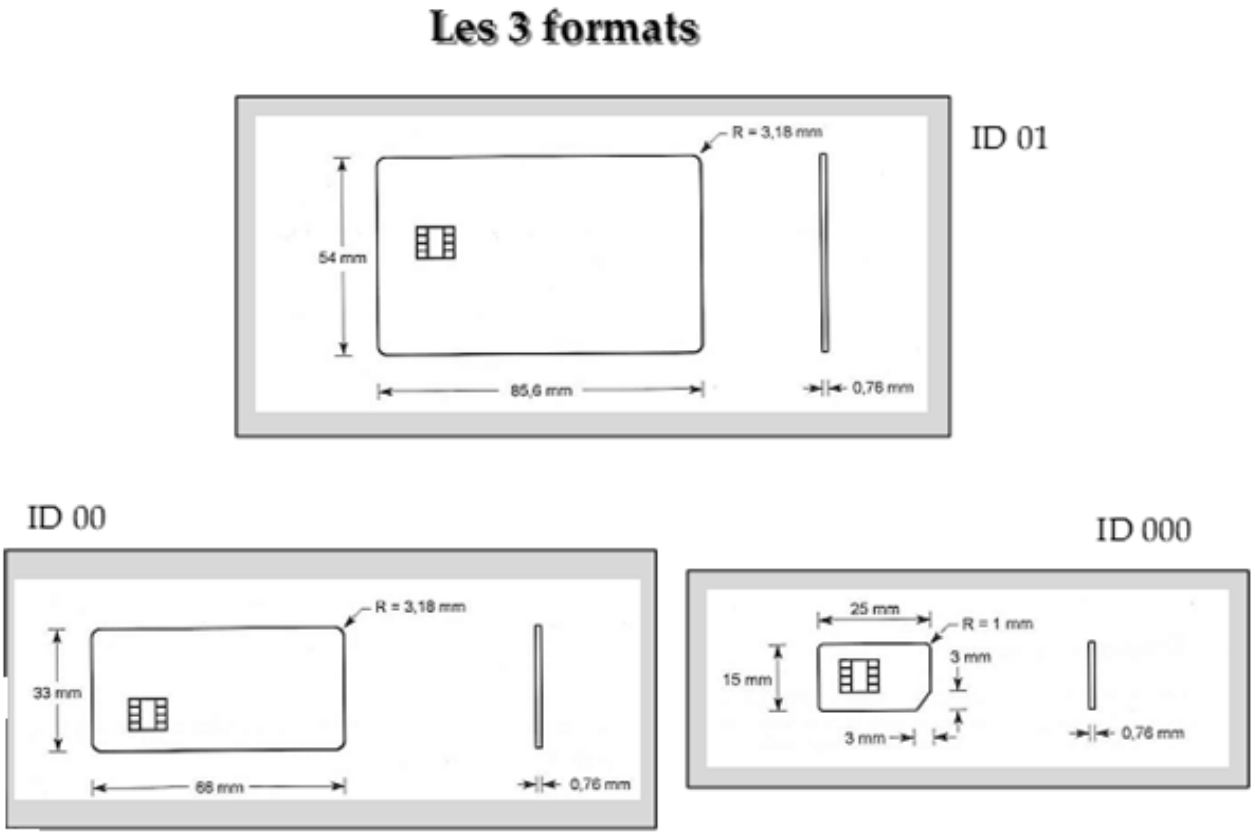


Fig.1. 11: Les dimensions des 3 formes de cartes à contact [2].

Le fabricant produit une seule taille (ID1), le client final pourra réduire ses dimensions au format ID00 ou ID000 (ex. carte SIM) (voir la figure 1.12).

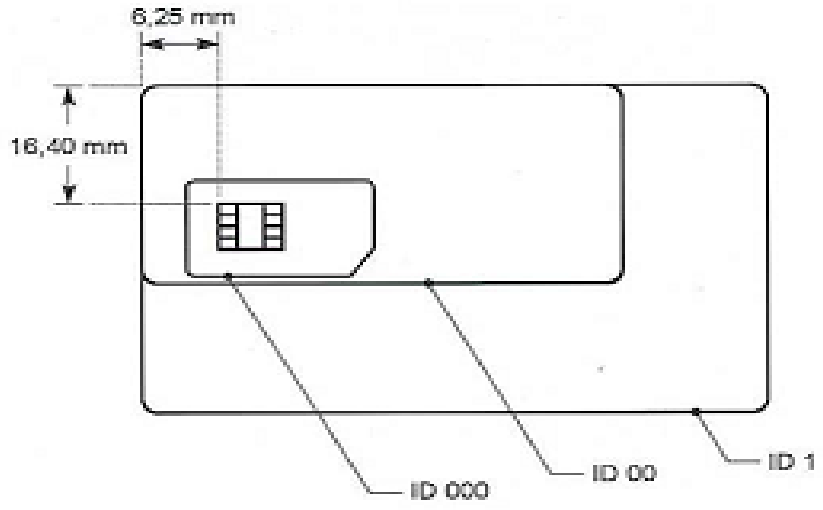


Fig.1. 12: Mise en évidence de la comptabilité entre les 3 formes de cartes [2].

En plus :

- La carte doit être opaque aux rayons UV (la puce insensible aux rayons UV).
- La carte doit résister aux détériorations de sa surface.

- La carte doit protéger la puce lors de manipulation de stockage lors d'une utilisation normale.
- La zone des contacts doit résister à la pression causée par une bille d'acier de 1,55mm de diamètre appliquée avec une force intérieure <ou=1,5N.
- La puce doit résister aux rayons X.
- La carte ne doit pas être endommagée par un champ magnétique statique de 79 500 A/m.
- Etc. [1, 3]

1.8.2. Brochage des cartes à puce [1, 10]

La carte possède 8 contacts et parmi eux on trouve 2 réservés à une future utilisation. Les différentes connections sont présentées dans la figure 1.13. Les fonctionnalités de ces connections sont données dans le tableau 1.2.

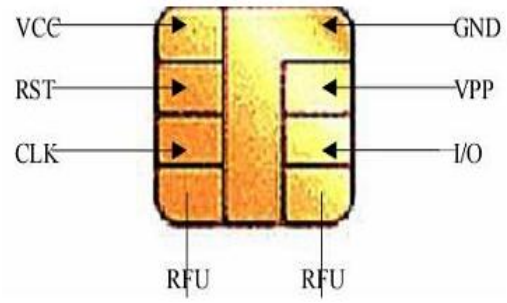


Fig.1. 13: Brochage des contacts d'une carte à puce selon la norme ISO 7816-3.

Position	Abréviation technique	Opération
C1	Vcc	Tension d'alimentation
C2	MCLR	Remise à zéro
C3	CLK	Fréquence d'horloge (Entrée)
C4, C8	RFU	Réservé à une Future Utilisation
C5	Vss	Masse
C6	Vpp	Tension de programmation de l'EPROM
C7	I/O	Données série (Entrée / Sortie)

Tab.1. 2: Fonctionnalités des contacts de la carte à puce Wafer 2.

1.8.3. Position des contacts [2, 5, 10]

Encore plus peut-être que les dimensions des cartes, les contacts de connexion avec la puce doivent avoir une position parfaitement normalisée faute de pouvoir lire n'importe quelle carte dans n'importe quel lecteur. La figure 1.14 donne la position des contacts selon la norme ISO 7816-2 actuelle.

	A	B	C	D
C1	10,25	12,25	19,23	20,93
C2	10,25	12,25	21,77	23,47
C3	10,25	12,25	24,31	26,01
C4	10,25	12,25	26,85	28,55
C5	17,87	19,87	19,23	20,93
C6	17,87	19,87	21,77	23,47
C7	17,87	19,87	24,31	26,01
C8	17,87	19,87	28,85	28,55

Position ISO 7816

Valeurs en mm

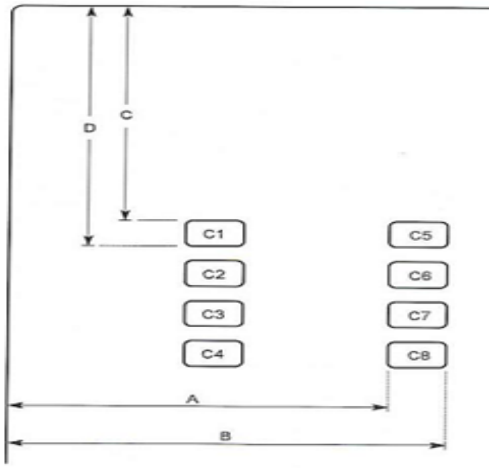


Fig.1. 14: Position des contacts selon la norme ISO 7816-2 actuelle [2].

1.8.4. Caractéristiques électriques [2, 5]

- Vcc : 4.75 ≤Vcc≤5.25v
- RST : valeur min = 4V ou Vcc-0.7V
- CLK : Min=2.4 ou 0.7Vcc ou encore Vcc-0.7V
  Max= Vcc
- E/S : entrée : Min= 2V ou 0.7 Vcc
  Max= Vcc
  Sortie : Min=2.4V ou 3.8V
  Max = Vcc

1.9. Insertion de la carte dans un lecteur [2, 6]

1.9.1. Etapes d'activation de l'interface avec la carte

Une fois la carte placée dans le lecteur, les opérations suivantes sont déclenchées :

- Mise au niveau bas de l'entrée Reset ;
- Alimentation de la carte via son entrée Vcc ;
- Mise en mode réception de la ligne I/O du circuit d'interface du lecteur ;
- Mise au niveau repos de Vpp de la carte ;
- Génération d'une horloge stable sur l'entrée CLK de la carte ;
- Un reset est alors provoqué par le circuit d'interface.

Après la connexion de la carte et l'activation de ses contacts, un reset est alors provoqué par le circuit d'interface, la carte répond par une réponse au reset ou ATR. Ensuite, un dialogue entre la carte et l'application à lieu via le circuit d'interface. Et à la fin les contacts sont désactivés par le circuit d'interface, et à ce moment-là on peut retirer la carte.

1.9.2. Désactivation de l'interface avec la carte [2, 6]

La désactivation normale a lieu lorsque la transaction en cours se termine et le terminal nous invite à retirer la carte. Avant l'affichage du message de retrait, il y a :

- Mise au niveau bas de l'entrée RST ;
- Mise au niveau bas de CLK ;

- Mise au niveau inactif de Vpp ;
- Mise au niveau inactif d'I/O ;
- Coupure de l'alimentation Vcc.

Retrait de la carte (retrait de la carte avant la fin de la transaction doit être prise en charge par l'application).

1.10. ATR (Answer To Reset) [1, 2, 5, 6]

Dès que la carte est mise sous tension, elle envoie un message de réponse d'initialisation appelé ATR.

Fig.1. 15: La réponde au RESET (ATR).

1.10.1. Caractéristiques de l'ATR [2]

- L'ATR est la réponse de la carte au Reset du terminal ;
- L'ATR au minimum = 2octets, au maximum = 33 octets ;
- Transmission en mode asynchrone semi-duplex ;
- La fréquence d'horloge comprise entre 1 et 5 MHz pour permettre à n'importe quel lecteur de lire le 1er caractère ;
- Communication entre le lecteur et la carte via la ligne bidirectionnelle I/O.

1.10.2. Chronogramme de la réponse au Reset [1]

- Commande Asynchrone (protocole RS232) : bit start + 8bits de données+ bit de parité paire + temps de garde (un ou plus bits Stop) (voir figure 1.16) ;
- L : niveau bas et H : niveau haut ;
- Le délai entre 2 caractères est au moins de 12 etu et TG = 2 etu.

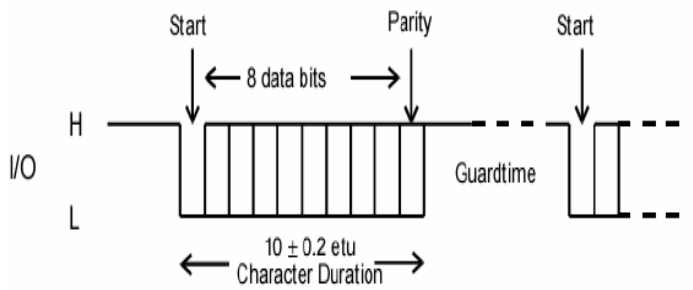


Fig.1. 16: Chronogramme d'échange d'un octet entre une carte et un lecteur.

1.10.2.1. Caractère initial de l'ATR [2]

- Premier caractère de l'ATR = TS (caractère initial)
- TS peut prendre 2 valeurs : $(HHLLLLLL)_2$ ou $(HHLHHHLL)_2$
  - 1 : Convention inverse : TS=3F (en héxa)
    - Niveau bas L = « un » logique ;
    - Niveau haut H = « zéro » logique ;
    - Bit transmis en premier = bit 7 de poids fort ;
    - Bit transmis en dernier = bit 0 de poids faible ;
  - 2 : convention directe : TS=3B (en héxa)
    - Niveau bas L = « 0 » logique ;
    - Niveau haut H = « 1 » logique ;
    - Bit transmis en premier = bit 0 de poids faible ;
    - Bit transmis en dernier = bit 7 de poids fort ;

1.10.2.2. Caractère T0 [2, 5]

Appelé aussi caractère de format, c'est le 2ème caractère de l'ATR dont la présence est également obligatoire.

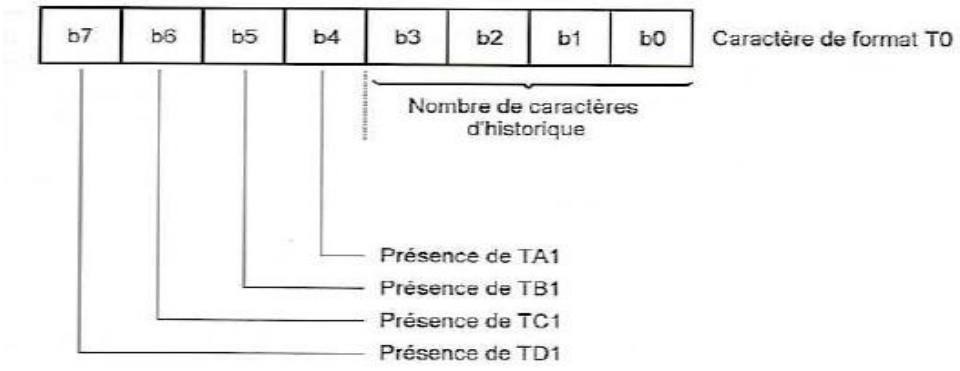


Fig.1. 17: Contenu du caractère T0 ou caractère de format de l'ATR [2].

La figure 1.17 présente le contenu du caractère T0. La partie basse, appelée K dans la norme, (b0 à b3) code le nombre de caractères d'historique (il ne peut y avoir plus que 15 caractères d'historique).

La partie hausse, appelée Y1 dans la norme, (b4 à b7) dont chaque bit indique la présence ou l'absence d'un des caractères TA1, TB1, TC1 et TD1.

La présence de TA1 et TD1 est codée dans T0 tandis que la présence des caractères TAi à TDi avec (i>1) est codée dans le caractère TD précédent.

La figure 1.18 donne l'allure générale de l'ATR.

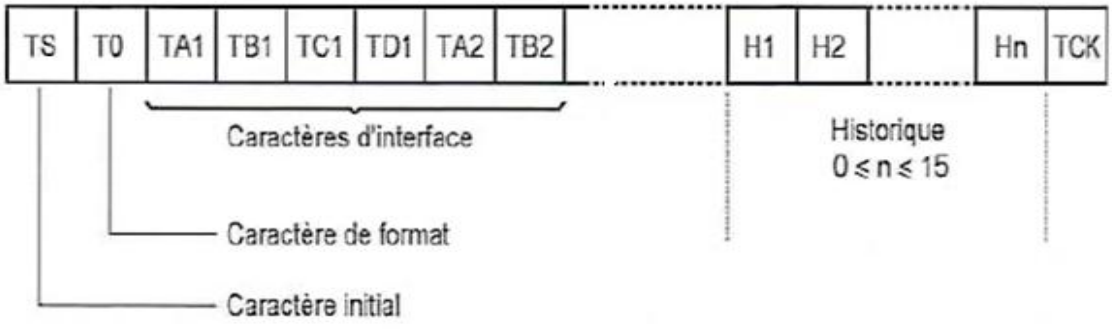


Fig.1. 18: Allure générale de l'ATR [2].

1.10.2.3. Caractère TA1

Permet de définir la vitesse de transmission utilisée après la phase de réponse au RESET.

1.10.2.4. Caractère TB1

L'utilisation de ce caractère est de plus en plus rare, il sert à coder la valeur de la haute tension de programmation Vpp ainsi que le courant nécessaire.

1.10.2.5. Caractère TC1

Code un paramètre appelé N qui est un temps de garde supplémentaire (le temps de garde est le temps qui s'écoule entre la fin d'un caractère et le début du suivant).

Ce temps de garde supplémentaire ne doit exister que lors des dialogues qui ont lieu dans le sens lecteur vers carte. Dans le sens carte vars lecteur ce temps supplémentaire n'existe pas.

1.10.2.6. Caractère TD1

Outre le fait qu'il code la présence éventuelle (et peu fréquente) des autres caractères TAi, TBi, TCi et TDi, code également sur ses 4bits de poids faible le numéro du protocole utilisé (voir la figure 1.19).

Présence de TA_2, TB_2, TC_2, TD_2 protocole

	b8	b7	b6	b5	b4	b3	b2	b1	
T=1	1	0	0	0	0	0	0	1	=(81)

Fig.1. 19: Contenu du caractère TD1 de l'ATR.

1.10.2.7. Caractères d'historique

Leurs présence est facultative, cas caractères ne sont pas définis dans une norme et leurs signification est laissée à l'appréciation du fabricant de la carte.

1.10.2.8. Caractère TCK

Le caractère TCK ne doit être présent que si un protocole différent de zéro a été spécifié au moyen de l'octet TD1.C'est un caractère de contrôle qui est calculé de telle façon que le OU exclusif entre tous les octets compris entre T0 (inclus) et TCK lui-même (également inclus) soit nul.

1.11. Les protocoles TPDU/APDU [2, 5, 6]

- TPDU : Transmission Protocol Data Unit.
- APDU : Application Protocol Data Unit.

1.11.1. Protocoles TPDU [10]

Il existe deux protocoles T=0 et T=1, le T=0 est le plus utilisé.

- **Protocole T=0 :** est de type caractère, son mode de fonctionnement est de type commande/réponse. Le terminal est l'initiateur des échanges.
- **Protocole T=1** : est un protocole ambitieux très peu utilisé, il est plus proche du modèle OSI car les échanges s'effectue en blocs structurés.

1.11.2. Protocoles APDU [2, 5]

La communication entre l'hôte et la carte est half-duplex. Elle se fait à l'aide de paquets appelés APDU.

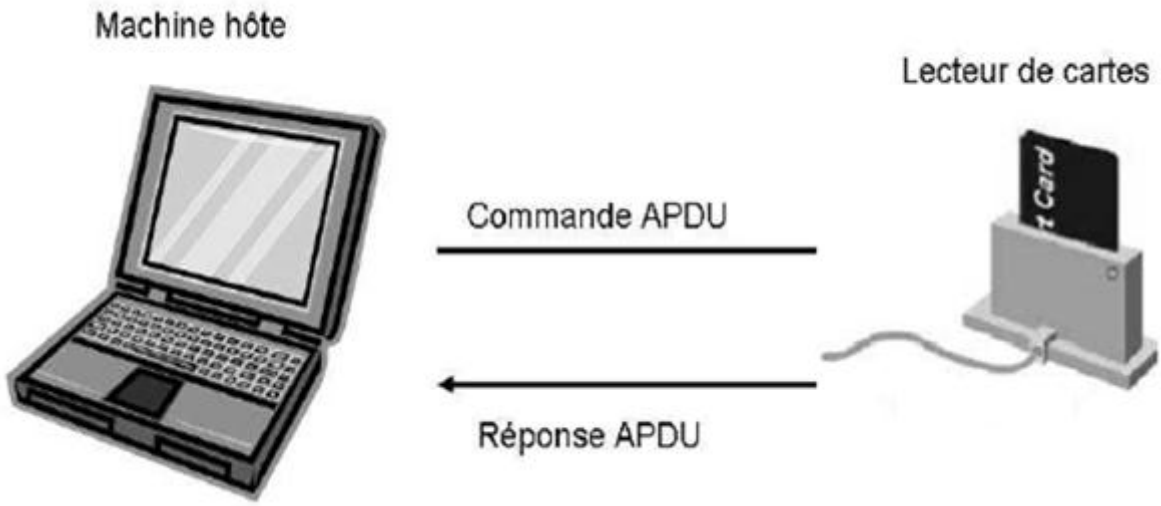


Fig.1. 20: Le modèle de communication de la carte à puce.

Un APDU contient une commande ou une réponse (voir la figure 1.20).

Le mode Maître/Esclave est utilisé. Ainsi la carte joue un rôle passif et attend une commande APDU à partir de l'hôte. Elle exécute l'instruction spécifiée dans la commande et retourne une réponse APDU.

1.11.3. Format des commandes/réponses APDU [2, 5]

Entête obligatoire				**Corps optionnel**		
CLA	INS	P1	P1	Lc	données	Le
1 octet	1 octet	1 octet	1 octet	1 octet	Non définie	Non définie

Tab.1. 3: Format des commandes APDU [10].

CLA : classe ;
INS : code de l'instruction ;
P1, P2 : paramètres de l'instruction ;
Lc : nombre d'octet présents dans le champ de données ;
Données : données à envoyer vers la carte ;
Le : nombre d'octet à recevoir de la carte ;

Corps optionnel	**partie obligatoire**	
données	SW1	SW2
varie	1 octet	1 octet

Tab.1. 4: Format des réponses APDU [10].

SW1, SW2 : Status Words (mots d'état) donnent l'état de traitement par la carte ;
Données : données reçues de la carte.

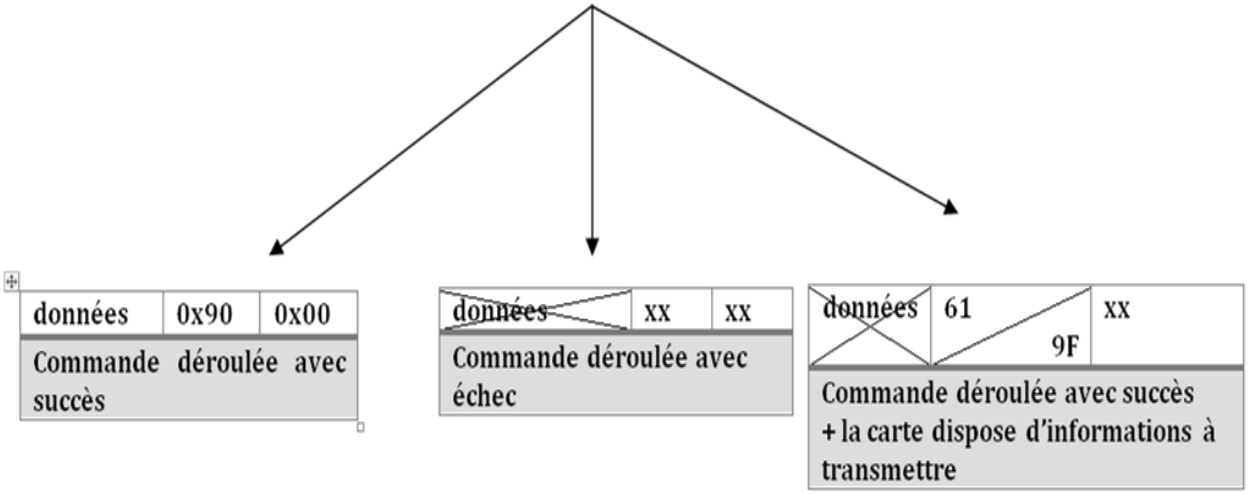


Fig.1. 21: Les différentes réponses APDU de la carte.

1.11.4. Commandes APDU [5, 10]

Il existe 5 types de commandes APDU selon qu'il y'a ou non échange de données utiles.

- **Envoi d'une commande sans données utiles**

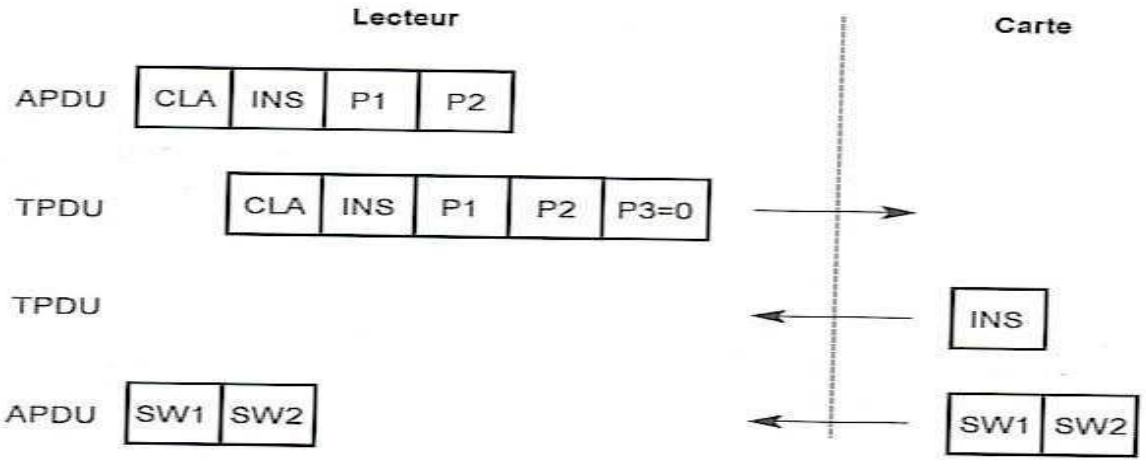


Fig.1. 22: Envoi d'une commande sans données utiles [2].

- **Commande avec réception de données utiles depuis la carte**

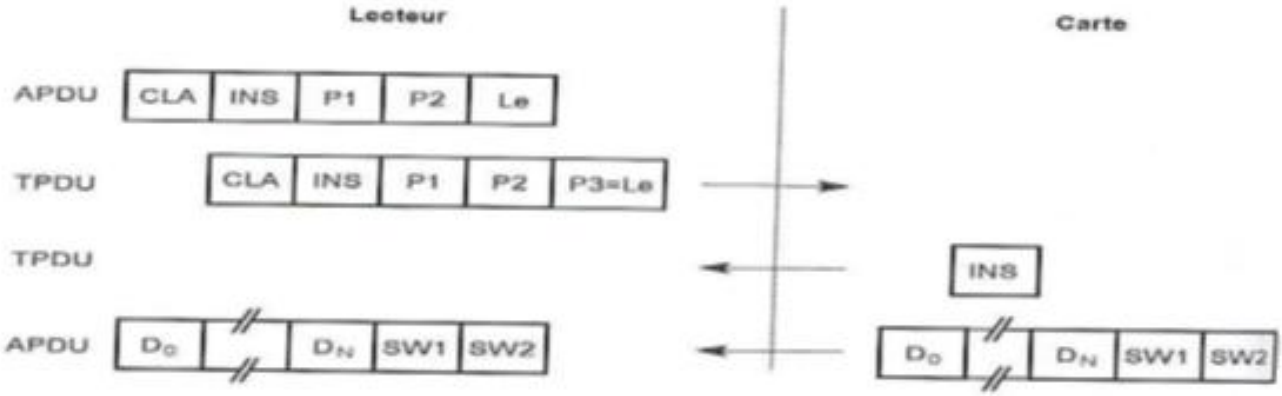


Fig.1. 23: Commande avec réception [2].

- **Commande avec invitation à lire des données depuis la carte**

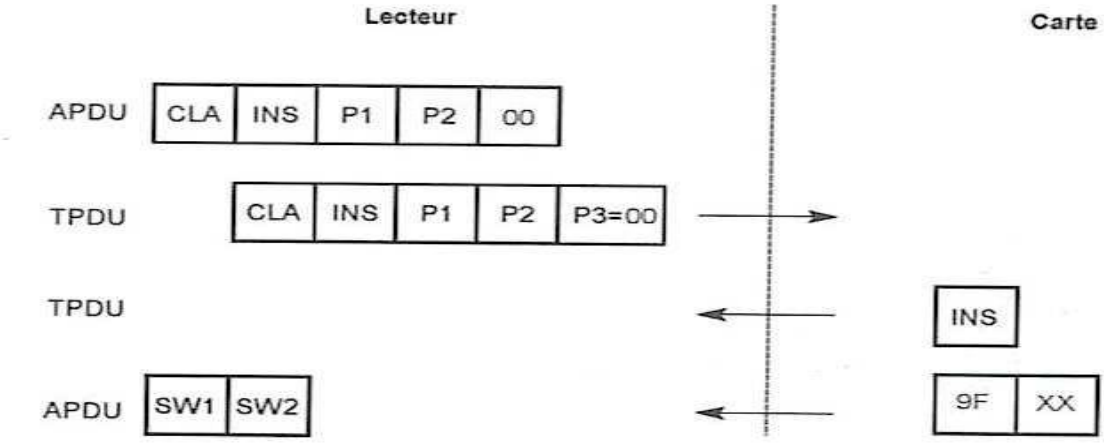


Fig.1. 24: Commande avec invitation à lire des données depuis la carte [2].

- **Commande avec envoi de données utiles vers la carte**

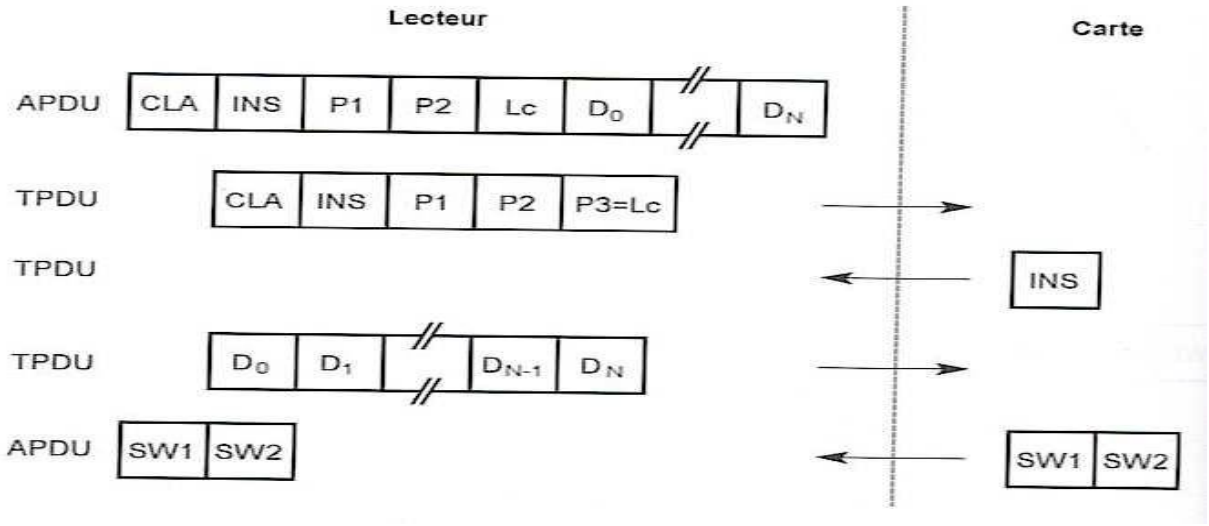


Fig.1. 25: Commande avec envoi de donnée utiles vers la carte [2].

- **Commande avec envoi et réception de données utiles**

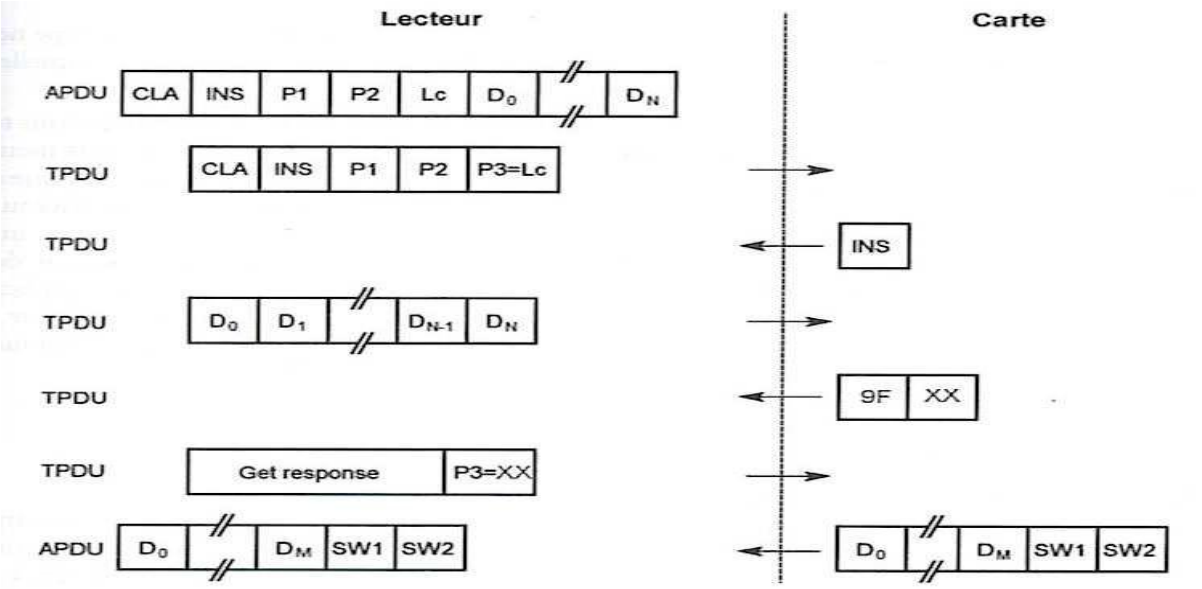


Fig.1. 26: Commande avec envoi et réception de données utiles [2].

1.12. Gestion des fichiers d'une carte à puce [2, 5]

1.12.1. Arborescence des fichiers et répertoires :

L'arborescence des fichiers et répertoires qu'il est possible de créer ou de rencontrer dans une carte à puce est (voir figure 1.27) :

- **EF (Elementary Files)** : fichiers élémentaires (feuilles de l'arbre) ;
- **DF (Dedicated Files)** : qu'on peut appeler Directory Files (répertoire) ;
- **MF (Master File)** : fichier maître, c'est le répertoire racine.

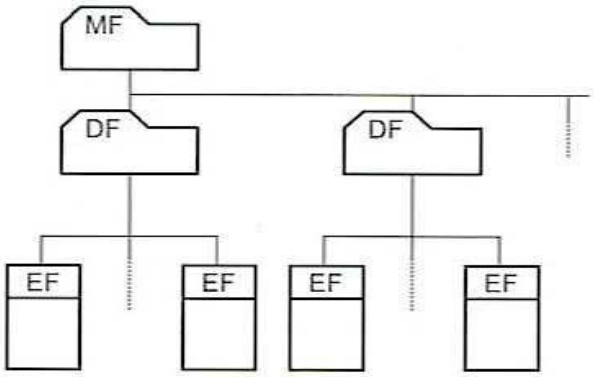


Fig.1. 27: Organisation de répertoire et sous répertoire (DF) et de fichiers de données (EF) [2].

1.12.2. Identification et nommage des fichiers

Plusieurs façons d'identifier un fichier dans la norme 7816-4 :

- Un DF ou un EF peut être référencé à l'aide d'un FID (File Identifier) sur 2 octets ;
- Un EF peut être référencé à l'aide d'un chemin d'accès : concaténation des FID en commençant de MF ;
- Un EF peut être référencé à l'aide d'un short FID sur 5 bits (valeurs autorisées 0 à 30) ;
- Un DF peut être référencé par un nom codé sur 1 à 16 octets.

1.12.3. Structure des fichiers

- Entête : contient le FID du fichier+autorisation d'accès ;
- Corps : données utiles du fichier ;
- Les entêtes sont stockés sur une page mémoire ;
- Les données sont hébergées sur une autre page.

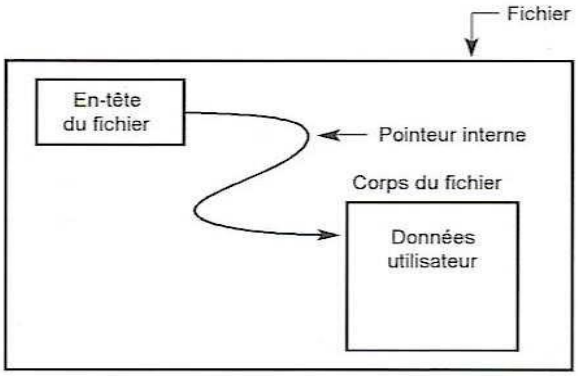


Fig.1. 28: Structure réelle des fichiers d'une carte à puce [2].

1.12.4. Les différents types de fichiers

- **Fichier a structure transparente**

Ce Fichier est une suite d'octet permettant de stocker une faible quantité de données (taille min=1octet, taille max : 255 octets).

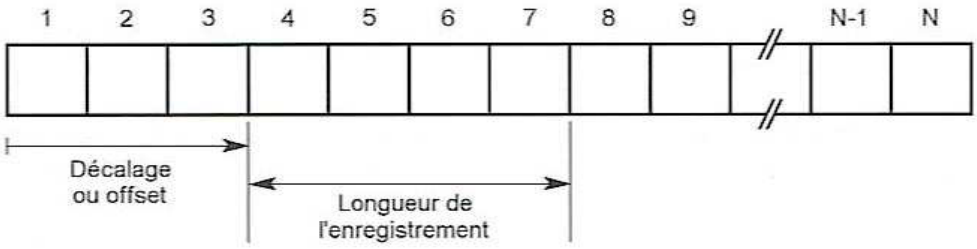


Fig.1. 29: Aspect d'un fichier à structure transparente [2].

- **Fichier a structure linéaire fixe**

Ce fichier est une suite d'enregistrement dont chaque enregistrement contient un nombre fixe d'octet.L'accès à un enregistrement se fait sur le numéro qui doit toujours commencer à partir de 1 (Taille min=1 octet, taille max : 255 octets).

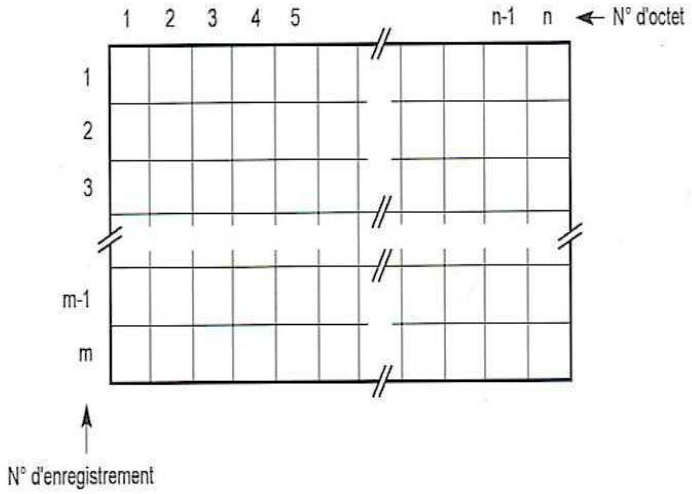


Fig.1. 30: Aspect d'un fichier à structure linéaire fixe [2].

- **Fichier a structure linéaire variable**

Il est utilisé pour stocker des noms dans un répertoire (carte SIM).Chaque enregistrement correspond à un nombre d'octets variable, l'indice commence à 1 (taille min = 1 octet, taille max : 254 octets).

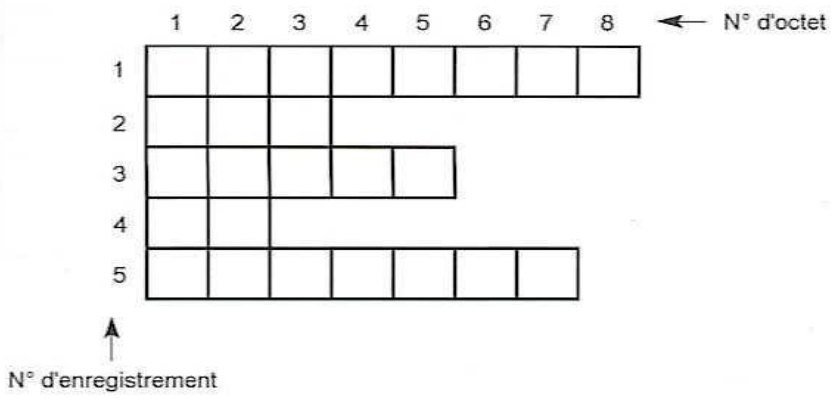


Fig.1. 31: Aspect d'un fichier à structure linéaire variable [2].

- **Fichier à structure linéaire cyclique**

Il s'agit d'une structure linéaire fixe fermée.

- L'indice 1 correspond toujours au dernier enregistrement écrit ;
- L'indice 2 correspond toujours à l'enregistrement écrit juste avant, etc. ;
- Taille max : 254 octets.

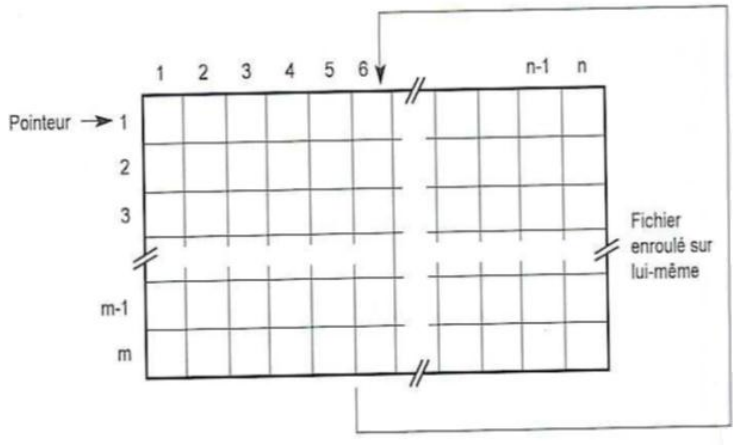


Fig.1. 32: Aspect d'un fichier à structure cyclique [2].

1.13. Les fausses cartes [1, 2]

Sous cette appellation un peu osée se cachent tout simplement des circuits en époxy de 8/10mm, aux dimensions des cartes à puce courantes, munis de contacts placés comme ceux des micromodules (voir la figure 1.33). Associées aux connecteurs de carte, ces fausses cartes permettent d'utiliser une carte conforme à n'importe quelle norme sur un lecteur ou encodeur prévu pour n'importe quelle autre norme. Ce principe ouvre la voie au développement, à l'aide de composants courants, de circuits destinés à un futur « encartage ».

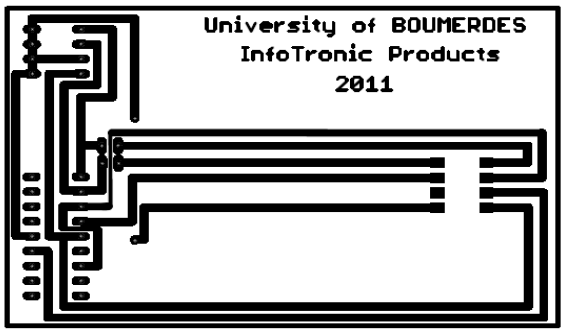


Fig.1. 33: Réalisation d'une « fausse carte » ISO (côté cuivre).

On remarque que la zone correspondant aux contacts de la norme concurrente est vide de cuivre, afin d'éviter tout risque de court-circuit dans les connecteurs « mixtes ». Bien entendu, la découpe du pourtour doit être effectuée avec précision, et suivie d'un rodage au papier abrasif fin. Après soudure des picots de raccordement du câble d'interconnexion (implantés côté composants), il est à conseiller de vernir le coté cuivre de la carte, non sans avoir provisoirement protégé la zone des contacts par un petit rectangle de ruban adhésif que l'on retirera délicatement après quelques minutes de séchage.

Si on en a la possibilité, on dorera les contacts du « faux micromodule ».On peut aussi se contenter de les étamer chimiquement (et surtout pas au fer à souder qui causerait une surépaisseur inacceptable) ou de laisser le cuivre à nu (on le frottera alors de temps en temps avec une gomme pour en éliminer l'inévitable oxydation).

La conception de toutes ces « fausses cartes » est basée sur le fait que la plupart des connecteurs courants « avalent » à peine la moitié de la carte. Il reste donc une place confortable pour implanter des composants d'une certaine épaisseur. Bien entendu, ces outils ne seront pas utilisables avec les lecteurs motorisés qui, pour des raisons évidentes de sécurité, entrainent la carte dans les profondeurs de leur mécanisme, ni dans ceux qui referment un volet par-dessus.

1.13.1. La carte Gold ou carte Wafer 1 [2]

Cette carte est la première à avoir été réalisé, elle est appelée Gold en raison de la couleur dorée de certaines versions.

Fig.1. 34: La carte Gold possède avec couleur dorée.

Elle est plus simple que certaines cartes actuelles, compte tenu de son âge et donc des composants qui étaient disponibles à l'époque de sa conception. Comme le montre son schéma de la figure 1.35, elle ne contient en effet qu'un classique microcontrôleur PIC 16F84 associé à une mémoire EEPROM série externe de type 24LC16 d'une capacité de 2 K octets.

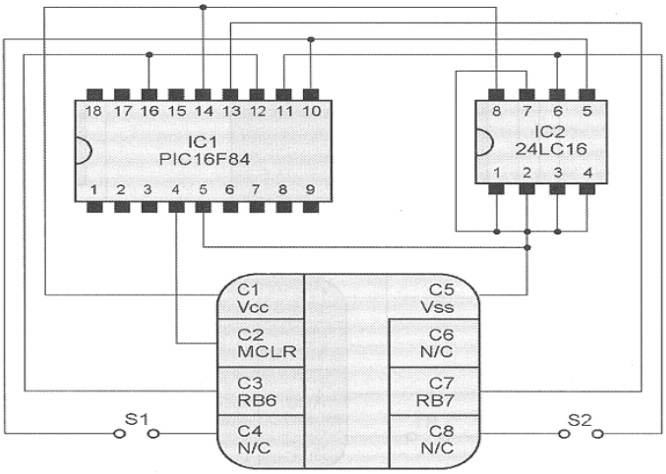


Fig.1. 35: Wafer 1.

Il est possibilité d'accéder aux lignes SDA et SCL de la mémoire EEPROM via les contacts C4 et C8. Cette possibilité optionnelle, d'où la présence des straps S1 et S2 permet de programmer directement le contenu de l'EEPROM sans passer "au travers" du microcontrôleur.

Ces deux liaisons n'existent que sur les cartes Gold que l'on réalise soi-même, elles n'existent pas dans les vraies cartes Wafer.

Pour programmer depuis l'extérieur la mémoire de ces cartes, il faut faire appel à un "loader" préalablement programmé dans le PIC, et qui rend en quelque sorte celui-ci «transparent » pendant la phase de programmation de l'EEPROM.

1.13.2. La carte Gold 64 ou carte Wafer 2 [2]

Son schéma est identique à celui de la Wafer 1, comme le montre la figure 1.36 mais la mémoire EEPROM devient de la 24LC64 c'est à dire qu'elle offre une capacité de 8 K octets.

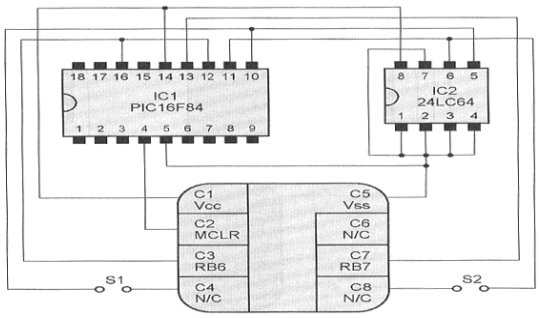


Fig.1. 36: Wafer 2.

1.13.3. La carte Silver ou carte Wafer 3 [2]

Appelée carte Wafer 3, car elle est arrivée après les deux précédentes, ou bien encore carte Silver car de nombreuses versions de cette carte sont disponibles sous une belle livrée argentée (voir les figures 1.37 et 1.38).

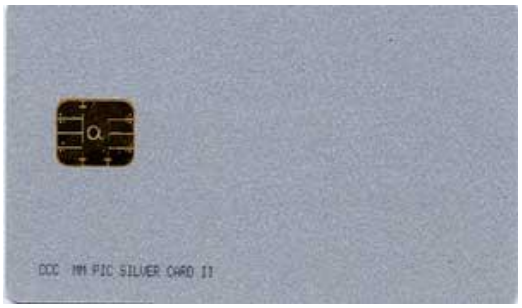


Fig.1. 37: La carte Silver classique.

Fig.1. 38: Une carte Silver plus"design".

Cette carte reste fidèle à la famille PIC de Microchip mais fait appel à un circuit doté en ressources internes avec le 16F876, comme le montre son schéma visible ci-dessous.

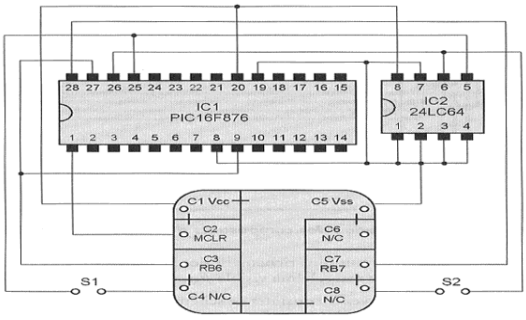


Fig.1. 39: wafer 3.

Le schéma est identique dans son principe à celui des cartes Gold mais utilise une mémoire EEPROM de 8 K octets. Cette carte est utilisée dans les applications les plus gourmandes en ressources mémoires.

1.13.4. La carte Fun ou Purple ou encore Wafer 4 [2]

Appelée carte Purple (pourpre en anglais), en raison de la couleur des premières versions qui ont été commercialisées, ou bien Fun Card (voir la figure 1.40).

Fig.1. 40: Une carte Fun 7 avec 26 K Byte.

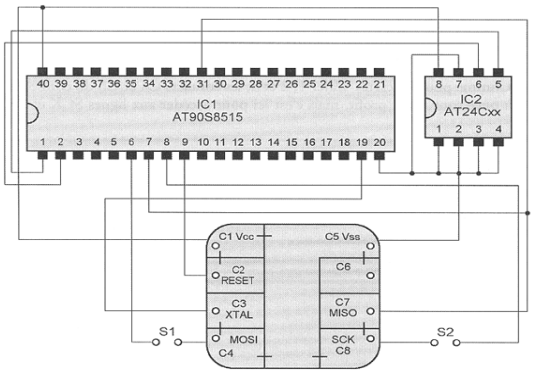


Fig.1. 41: Wafer 4.

Sur le schéma de la Fun Card on trouve un microcontrôleur Atmel AVR de type AT90S8515 qui est un circuit situé dans la partie haute de la famille AVR. Comme pour les cartes précédentes, il est associé à de la mémoire EEPROM externe dont la taille varie selon le type de Fun Card choisie.

Les contacts C4 et C8 du connecteur carte à puce sont mis pour accéder aux lignes SCK et MOSI du microcontrôleur en phase de programmation de ce dernier. Sur les cartes réelles et contrairement à ce qui est fait pour les Gold et Silver, ces liaisons sont établies en permanence. Dans le cas contraire, il serait impossible de programmer l'AT90S8515.

1.13.5. La carte Jupiter ou Pink ou encore Wafer 5 [2]

La couleur rose de certaines versions lui a valu son appellation de Pink Card, mais cette carte s'appelle aussi Wafer 5 ou bien encore Jupiter Card (voir la figure 1.42).

Mais contrairement à la Fun Card, elle fait appel à un des plus petits microcontrôleurs de la famille AVR avec l'AT90S2343, associé ici encore à de l'EEPROM externe de type AT24C16 c'est-à-dire de 2K octets (voir la figure 1.43).

Fig.1. 42: La carte Pink classique.

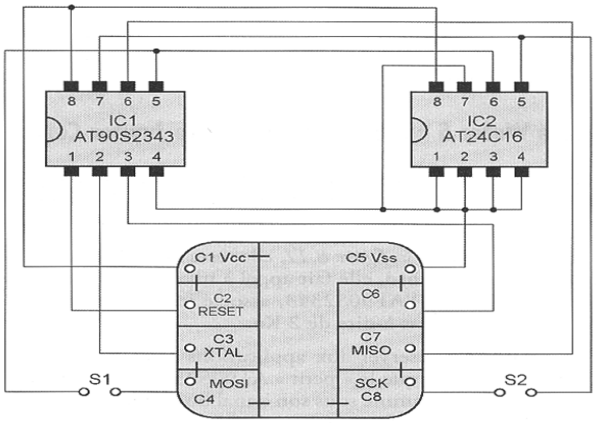


Fig.1. 43: Wafer 5.

1.14. Lecteurs de cartes à puce [1, 3, 4]

On appelle « lecteurs » de cartes à puce des appareils souvent capable aussi bien de lire que d'écrire dans celles-ci. Le terme le plus approprié est par conséquent « lecteur-encodeur ».

1.14.1. Les interfaces matérielles des lecteurs [1, 2]

Un lecteur de cartes à puce est la réunion dans un boitier ou sur un simple circuit imprimé d'un :

- Coupleur ;
- Connecteur de cartes ;
- Système d'alimentation ;
- Circuit assurant la liaison avec un système informatique « hôte ».

Certains lecteurs pourront être autonomes, et donc dotés d'un afficheur et, éventuellement, d'un clavier complet ou simplement de quelques boutons. D'autres lecteurs se connectent à un PC, on y trouve deux types :

- Lecteurs à interface série ;
- Lecteurs à interface USB.

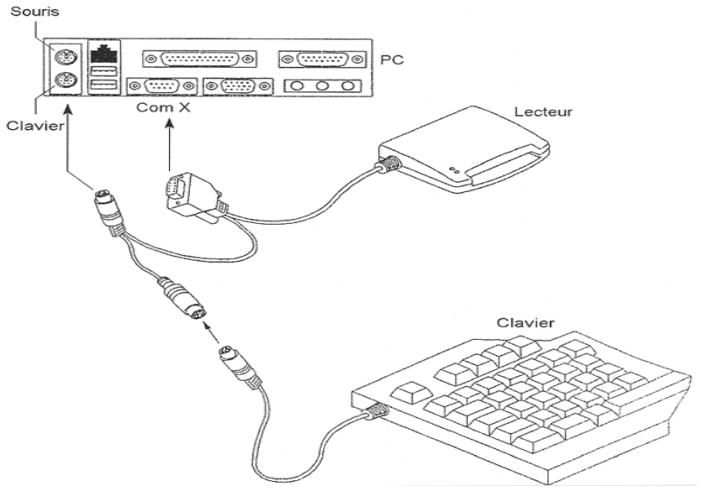


Fig.1. 44: La connexion d'un lecteur à interface avec un ordinateur [2].

Les lecteurs USB sont alimentés par l'intermédiaire du bus USB. Par contre les lecteurs à interface série utilisent une alimentation externe.

Des versions équipées d'un émetteur-récepteur radio permettent même de communiquer avec les cartes à puce sans contacts».

1.14.2. Le choix d'un lecteur [2, 9]

Lorsque l'on se préoccupe de développer une application carte à puce, le choix d'un lecteur est presque aussi important que celui de la carte elle-même. Le choix d'un lecteur de carte dépend de ce qu'on veut faire mais aussi et surtout des types de cartes qu'on souhaite lire ou programmer.

Si on utilise des cartes vierges telles que les cartes Gold, Silver, Fun ou Jupiter, il nous faut un programmateur, au moins pendant la phase de développement de notre application. C'est en effet ce programmateur qui va programmer le microcontrôleur de la carte et/ou sa mémoire EEPROM. Une fois cette carte programmée, et si nous avons écrit un programme compatible des normes ISO 7816 3 et 4, un lecteur classique pourra ensuite être utilisé pour lire et écrire dans notre carte.

Si on utilise des cartes personnalisables ou des cartes à puce à OS ouvert comme la Basic Card, un lecteur classique suffit.

Si nous ne voulons que lire dans une carte, que ce soit une carte bancaire, Vitale ou tout autre carte conforme aux normes ISO 7816 3 et 4, le lecteur classique convient aussi.

Fig.1. 45: Un programmateur classique en boîtier Le Multipro 2000.

Fig.1. 46: Le lecteur Cyber Mouse d'ACS.

1.15. Conclusion

L'évolution de la technologie a permis la conception de cartes à puce de plus en plus performantes et sécurisées ainsi, l'implémentation d'applications est devenue plus rapide mais leurs coûts augmentent aussi. Le choix de la carte à puce qui convient le mieux à une application donnée est devenu une tâche difficile, en particulier pour celles destinées à un grand nombre d'utilisateurs. Il faut réduire les coûts tout en assurant le bon fonctionnement.

Il serait donc crucial pour les développeurs d'applications de s'interroger sur les performances des cartes à puce utilisées tout en réduisant les couts.

Dans le chapitre suivant, nous introduisons le contexte de la sécurité des cartes à puce qui est la raison de sa popularité. Ensuite dans les 3 autres chapitres, nous présentons nos contributions concernant l'optimisation et l'amélioration de deux applications de cartes à puce.

Chapitre 2

Cryptographie sur cartes à puce

Certains des travaux présentés dans ce chapitre ont fait l'objet d'un article présenté lors de la conférence [ICEE 2015].

2.1. Introduction

La carte à puce a subit des évolutions majeures qui ont permis de faciliter son utilisation et de garder la confiance des utilisateurs. Elle est devenue un élément incontournable et essentiel qui permit d'assurer un niveau de sécurité important.

L'objectif de ce chapitre est d'expliquer comment les nouvelles applications bénéficient de stocker des informations en toute sécurité sur une carte à puce grâce aux algorithmes cryptographiques.

Nous introduisons quelques notions sur la cryptographie moderne ainsi que son historique. Les crypto processeurs des cartes à puce ainsi qu'une étude comparative sur différentes implémentations de l'algorithme cryptographique DES sur FPGA afin de choisir la meilleure implémentation pour une carte à puce. En fin, nous expliquons aussi comment on peut associer la cryptographie avec la biométrie sur une carte à puce pour une meilleure sécurité.

2.2. Cryptologie [11-13]

Cryptologie peut être divisée en deux catégories : la cryptographie et la cryptanalyse, comme illustré à la figure 2.1. La cryptographie est l'étude des méthodes de cryptage et de décryptage des données, tandis que la cryptanalyse se concentre sur les tentatives pour briser les systèmes cryptographiques existants.

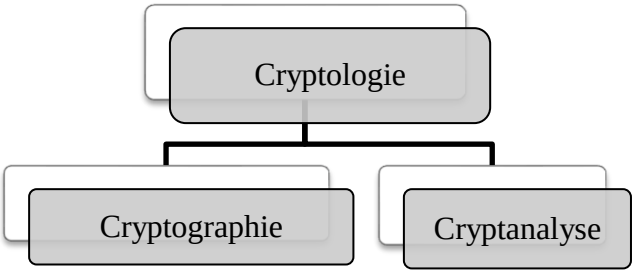


Fig.2. 1: Les deux principaux domaines de la cryptologie : la cryptographie et la cryptanalyse.

2.3. Historique de la cryptographie moderne [5, 11-13]

Avec la généralisation du traitement électronique de données dans les années 1960, la discipline de la cryptographie a connu une sorte de saut quantique. Le matériel et les logiciels de haute performance ont permis de mettre en œuvre des algorithmes mathématiques complexes et sophistiqués dans les processeurs mono-puce, ce qui a permis d'atteindre des niveaux de sécurité inégales précédemment. De plus, cette nouvelle technologie était disponible à tout le monde, contrairement à la situation antérieure dans laquelle la cryptographie était une science secrète réservée aux services militaires.

De l'antiquité jusqu'aux années 1970, tous les systèmes de cryptographie étaient symétriques (donc la principale préoccupation était la confidentialité) et il n'existait pas de méthode sûre pour échanger les clés secrètes. Historiquement la cryptographie symétrique est le premier type de chiffrement utilisé et elle fournit le seul chiffrement théoriquement indéchiffrable (chiffrement de Vernam ou one-time pad) d'après la théorie de Shannon (1949). Elle est d'une grande efficacité en termes de temps de calculs.

C'est en 1976 que deux chercheurs White Diffie et Martin Hellmann à l'université de Standford, dans leur papier "New Direction In Cryptography", ont donné une solution au problème

d'échange des clés et ont proposé en même temps le modèle théorique de la cryptographie à clé publique qui est appelé modèle de cryptographie asymétrique.

2.4. Les objectifs de la cryptographie [5, 12, 14]

Le rôle de la cryptographie est de garantir la sécurité des communications c'est dire de permettre à des entités qui ne se font pas confiance de communiquer en toute sécurité en présence de potentiels adversaires (susceptibles entre autres d'intercepter et de modifier les informations échangées ou d'usurper des identités).

Comme le montre la figure 2.2, les quatre objectifs de la cryptographie maintiennent le secret ou la confidentialité des messages, tout en assurant l'intégrité, l'authenticité et le non répudiation des messages. Ces objectifs sont mutuellement indépendants, et ils placent des exigences différentes sur le système concerné.

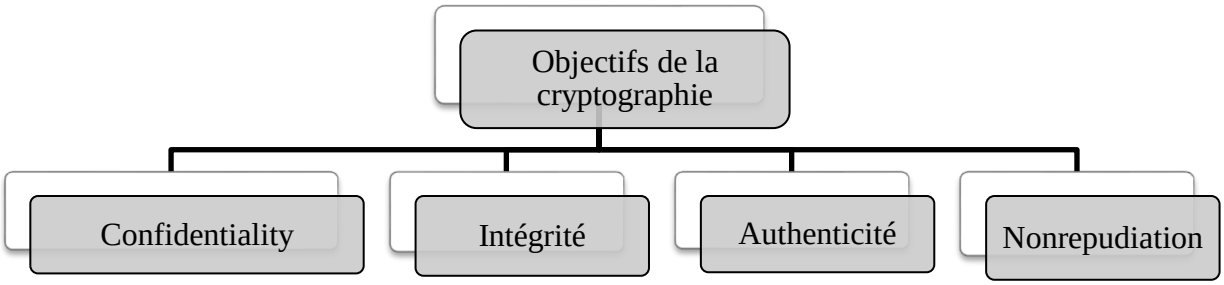


Fig.2. 2: Les quatre objectifs distincts de la cryptographie [5].

- **La confidentialité** signifie que seul le destinataire d'un message ne peut décrypter le contenu.
- **Intégrité** signifie que le destinataire peut vérifier que le message reçu n'a pas été modifié pendant la transmission.
- **Authenticité** assure que l'émetteur est bien l'auteur du message.
- **Non Répudiation** signifie que l'expéditeur peut vérifier qu'un certain destinataire a reçu un message particulier.

2.5. Principes de la cryptographie [5, 14]

Les algorithmes cryptographiques modernes sont généralement basés sur le principe de Kerckhoff (1835-1903). Il dit que l'ensemble de la sécurité d'un algorithme doit être basée uniquement sur la confidentialité de la clé, et non sur la confidentialité de l'algorithme lui-même. La conséquence de ce principe généralement connu mais souvent ignoré est que de nombreux algorithmes utilisés dans le secteur civil ont été publiés, et certains d'entre eux ont été normalisés.

Le contraire du principe de Kerckhoff est le principe de la sécurité par l'obscurité. Avec ce principe, la sécurité d'un système est basée sur l'idée qu'un attaquant ne sait pas comment le système fonctionne. Ce principe est très ancien, et il est encore utilisé fréquemment, même aujourd'hui. Toutefois, les systèmes cryptographiques (et d'autres systèmes) ne doivent pas être élaborés uniquement sur la base de ce principe. Jusqu'à présent, tous les systèmes basés sur ce principe ont été rompu, généralement dans un temps très court. Dans notre société de l'information, il n'est généralement pas possible de garder les détails techniques d'un système secret pendant une longue période, ce qui est précisément ce que ce principe exige.

2.6. Terminologies de la cryptographie [14, 15]

En termes simplifiés, la technologie de chiffrement traite trois types de données. La première est des données non chiffrées, ce qui est appelé Texte en clair. À l'opposé de cela les données cryptées, qui est appelé cryptogramme. En outre, une ou plusieurs clés sont nécessaires pour le chiffrement et le déchiffrement. Ces trois types de données sont traités par un algorithme de chiffrement.

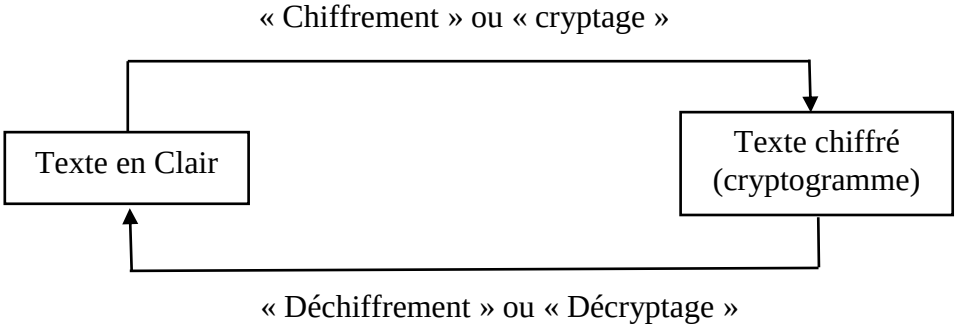


Fig.2. 3: Terminologies de la cryptographie : cryptage et décryptage.

Un terme qui se pose souvent en relation avec des algorithmes de chiffrement est la taille de la clé. Ceci fait référence au nombre de clés possibles qui peut être utilisée avec un algorithme cryptographique particulier. Une clé avec une taille plus grande est l'une des caractéristiques essentielles d'un algorithme cryptographique sécurisé.

2.7. Protocole d'échange de clé [11, 15]

L'échange de clé de Diffie-Hellman a été développé par ces deux auteurs en 1976 et publié dans l'article : W. Diffie and M.E. Hellman, New directions in cryptography, IEEE Transactions on Information Theory 22 (1976), 644-654. Ce protocole de Diffie-Hellman est le premier protocole d'échange de clés.

L'échange d'une clé secrète est fondamental en cryptographie. En effet tout chiffrement d'une grande quantité de données ne peut se faire qu'avec du chiffrement à clé secrète, surtout si cet échange a lieu en temps réel, en raison de la lenteur relative des chiffrements à clé publique.

Le développement rapide des réseaux de communication numériques, ainsi que l'augmentation du nombre des utilisateurs, ont posé de façon de plus en plus critique le problème de l'échange des clés de chiffrement.

2.8. Classification des algorithmes cryptographiques [3, 5, 6, 7, 14, 16]

Les algorithmes cryptographiques sont classés en deux types : symétriques et asymétriques. Cette classification est basée sur la clé utilisée.

Les algorithmes symétriques utilisent la même clé pour le chiffrement et le déchiffrement, il paraît donc nécessaire d'avoir un protocole d'échange de clés sûr et efficace, tandis que les algorithmes asymétriques, qui ont d'abord été postulées en 1976 par Whitfield Diffie et Martin E. Hellman, utilisent différentes clés pour le chiffrement et le déchiffrement.

2.8.1. Les algorithmes cryptographiques symétriques [5, 14, 16]

Les algorithmes de chiffrement symétriques sont basés sur le principe de l'exécution de cryptage et de décryptage en utilisant la même clé secrète - d'où la dénomination «symétrique». On trouve les algorithmes : DES, TDES, AES …etc.

Les principales caractéristiques des algorithmes de chiffrement symétriques utilisés dans les cartes à puce sont résumées dans le tableau 2.1.

Name	Plaintext size	Ciphertext size	Key size
DES	8 bytes	8 bytes	56 bits
Triple DES with two keys	8 bytes	8 bytes	112 bits (2 × 56 bits)
Triple DES with three keys	8 bytes	8 bytes	168 bits (3 × 56 bits)
IDEA	8 bytes	8 bytes	128 bits
AES	16 bytes	16 bytes	128 bits (16 bytes)
			192 bits (24 bytes)
			256 bits (32 bytes)

Tab.2. 1: Les paramètres d'entrées sorties des algorithmes cryptographiques symétriques utilisés pour les cartes à puce [5].

2.8.1.1. L'algorithme DES (Data Encryption Standard) [5, 6, 8, 12]

Le DES (Data Encryption Standard), ou DEA (Data Encryption Algorithm), a été développé dans les années 1970 par IBM (avec l'aide de la National Security Agency) et adopté en 1977 comme Federal Information Processing Standard pour les Etats-Unis (FIPS 46). DES est un algorithme de chiffrement symétrique par blocs de 64 bits. Il est largement utilisé dans certains domaines tels que les cartes magnétiques et cartes à puce.

Implementation	Time and throughput
Smart card with 3.5-MHz clock, software implementation	17.0 ms (3.8 kbit/s)
Smart card with 3.5-MHz clock and DES processing unit	112 µs (571 kbit/s)
Smart card with 3.,5-MHz clock and triple-DES processing unit	130 µs (492 kbit/s)
Smart card with 4.9-MHz clock, software implementation	12.0 ms (5.3 kbit/s)
Smart card with 4.9-MHz clock and DES processing unit	80 µs (800 kbit/s)
Smart card with 4.9-MHz clock and triple-DES processing unit	93 µs (688 kbit/s)
PC (Pentium, 200 MHz)	4 µs (16 Mbit/s)
PC (Pentium, 2 GHz)	800 ns (80 Mbit/s)
DES hardware component	64 ns (1 Gbit/s)

Tab.2. 2: Temps de calcul du DES avec un bloc de 8 octets [5].

- **Description de l'algorithme DES**

Dans cette section, nous fournissons un aperçu sur les principales composantes du DES.

Le bloc de chiffrement DES est un réseau de Feistel de 16-ronde avec une longueur de bloc de 64 bits et une longueur de clé de 56 bits. La même fonction de tour (F) est utilisée dans chacun des 16 tours. A partir de la clé principale de 56 bits on dérive une séquence de sous-clés $k1,\ldots, K16$ de de 48 bits . Un aperçu du DES est représenté sur la figure 2.4.

- **Processus de chiffrement**

Le texte brut de 64 bits est converti par la permutation initiale (*IP*), chiffre en 16 tours, suivis par l'inverse de la permutation initiale (*IP-1*). A chaque tour, les 32 bits du côté droit du bloc sont transformées avec la fonction marquée (*F*) et une sous-clé, puis OU exclusif (XOR) avec le côté gauche de 32 bits. Après chaque tour, les deux côtés du bloc de données sont inversés et l'algorithme continue de la même façon pour les autres tours.

- **La génération des clés**

Pour chaque tour du DES, une sous-clé de 48 bits doit être générée. La clé d'entrée est sur 64 bits, mais 8 bits sont utilisés pour le contrôle de parité. Après une permutation initiale de clé (*CP-1*), 16 sous-clés, une pour chaque tour, sont dérivées de la clé de 56 bits sélectionnés pour le chiffrement. La sous-clé est obtenue après un décalage à gauche, et après une permutation (*CP-2*) de 56 à 48 binaire [].

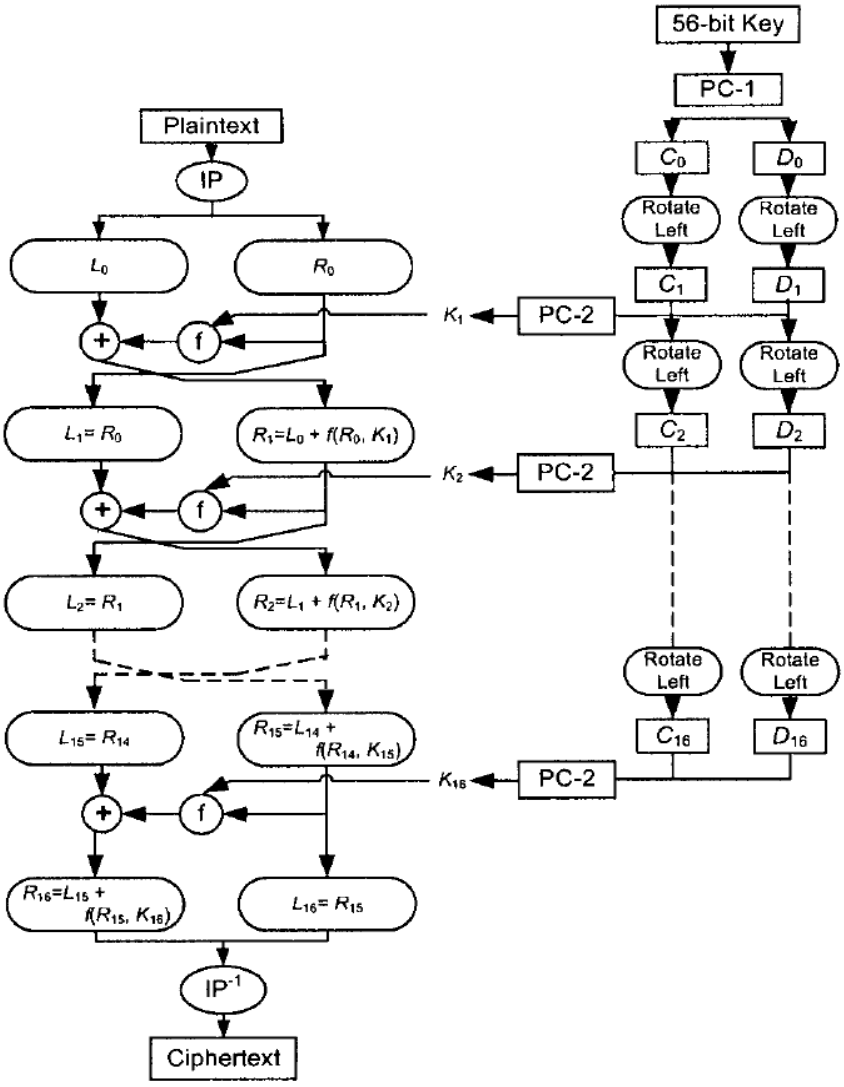


Fig.2. 4: L'Algorithme DES [13].

2.8.1.2. L'algorithme TDES (Triple DES) [5, 13]

L'algorithme Triple DES est connue par d'autres noms, y compris TDES, DES-3, 3-DES et le 2-DES. Ces noms expriment le fait que les trois clés de 56 bits sont utilisés, bien que les première et

troisième clés sont souvent les mêmes. Par conséquent, la taille de la clé doit toujours être indiquée en relation avec triple DES, afin de préciser l'algorithme sans ambiguïté. Cela vaut également pour les désignations 2TDES et 3TDES fréquemment utilisés.

La méthode TDES est beaucoup plus sûre que le double cryptage séquentiel à l'aide de deux clés différentes parce que l'attaque meet-in-the-middle n'est pas efficace contre cette méthode. Il faut trois clés de 56 bits au lieu d'une seule, ce qui donne une taille de clé de 2 × 56 bits ou 3 x 56 bits. Cette méthode est compatible avec l'algorithme DES normal et encourt aucun coût supplémentaire autre que la taille de la clé double ou triple. Cette compatibilité est l'une des principales raisons de l'utilisation généralisée du triple DES dans les cartes à puce.

2.8.1.3. L'algorithme AES [5, 12]

À la fin des années 1990, la quantité de puissance de calcul disponible en raison des progrès technologiques et de réseaux internationaux d'ordinateurs était suffisant pour permettre même à des particuliers ambitieux avec la capacité organisationnelle de monter avec succès des attaques en force sur l'algorithme DES. En conséquence, la viabilité du DES est devenu si discutable que les autorités nationales compétentes de plus en plus consacré leur attention à la spécification d'un nouveau cryptographique symétrique.

L'AES est un algorithme de chiffrement symétrique par bloc de 128 bits (16 octets) qui peuvent être utilisés avec trois tailles de clés différentes : 128 bits (16 octets), 192 bits (24 octets) et 256 bits (32 octets). Il est ainsi appelé AES-128, AES-192 ou AES-256, en fonction de la taille de la clé. L'AES est adapté pour l'implémentation matérielle, et il peut également être mis en œuvre facilement dans le logiciel en cours d'exécution à faible performance des processeurs 8 bits ou de haute performance 16 bits et 32 bits des processeurs.

En outre, il est internationalement sans licence, et selon la déclaration officielle du NIST sa durée de vie utile est plus de 20 ans. AES est normalisée par FIPS 197, qui est disponible gratuitement sur l'Internet [NIST].

Implementation	Time and throughput
Smart card, 16-bit CPU, 4.9-MHz clock, software implementation; encryption using a 128-bit key	20 ms (6.4 kbit/s)
Smart card, 16-bit CPU, 4.9-MHz clock, software implementation; decryption using a 128-bit key	25 ms (5.1 kbit/s)
Smart card with 3.5-MHz clock, software implementation	12.3 ms (5.2 kbit/s)
Smart card with 4.9-MHz clock, software implementation	8.8 ms (7.3 kbit/s)
PC (Pentium 4, 3.5 GHz)	160 ns (400 Mbit/s)
AES hardware component	1.9 ns (34 Gbit/s)

Tab.2. 3: Temps de calcule typique de l'AES avec une clé de 128 bits et un bloc de 16 octets [5].

2.8.2. Les algorithmes cryptographiques asymétriques [3, 6, 7, 16]

En 1976, Whitfield Diffie et Martin E. Hellman décrit la possibilité de développer un algorithme de chiffrement basé sur deux clés différentes. L'une de ces clés devait être publique, l'autre privée (confidentiel). Cela permettrait à toute personne possédant la clé publique pour chiffrer les messages, alors que seules les personnes possédant la clé privée pourraient les déchiffrer.

Cela permettrait d'éliminer les problèmes liés à l'échange et la distribution des clés symétriques confidentielles, et pour la première fois, il permettrait à certains autres processus, tels que la génération

de signatures numériques qui peuvent être vérifiés par tous. Parmis ces algorithme on trouve : RSA, ECC …etc.

2.8.2.1. L'algorithme RSA [11-13]

Deux ans plus tard, Ronald L. Rivest, Adi Shamir et Leonard Adleman ont proposé un algorithme qui satisfait aux critères énoncés par Diffie et Hellman. Cet algorithme, qui est appelé RSA après les initiales de ses inventeurs, est l'algorithme de chiffrement asymétrique le plus connu et le plus polyvalent actuellement en usage. Il est très simple, son principe de fonctionnement est basé sur l'arithmétique de grands nombres entiers. Les deux clés sont générées à partir de deux grands nombres premiers.

Le chiffrement et le déchiffrement utilisant l'algorithme RSA peut être exprimé mathématiquement comme suit, où x est le texte en clair, y est le texte chiffré, e est la clé publique, d est la clé privée, n représente le module public, et p et q sont des nombres premiers secrets :

$$\text{cryptage: } y = x^e \bmod n$$
$$\text{décryptage: } x = y^d \bmod n$$
$$\text{where } n = p.q$$

L'expéditeur et le destinataire doivent connaître la valeur de n. L'expéditeur connaît la valeur de e, et seul le récepteur connaît la valeur de d. Ainsi, la clé publique est définie comme $\{e, n\}$ et la clé privée est définie comme $\{d, n\}$.

La sécurité de l'algorithme est basée sur la difficulté de factoriser de grands nombres. Il est facile de calculer le module public des deux nombres premiers par la multiplication, mais il est très difficile de décomposer le module en ses deux premiers facteurs, car il n'y a pas des algorithmes efficaces pour ce processus.

2.9. Carte à puce et cryptographie [5, 14, 15]

La carte à puce est un moyen très sécurisé (basé sur la cryptographie) à la disposition de tout le monde, car il pourrait stocker en toute sécurité les clés secrètes et exécuter des algorithmes cryptographiques. En outre, les cartes à puce sont si petites et faciles à manipuler qu'ils peuvent être transportés et utilisés partout par tout le monde dans la vie quotidienne. Ce fut une idée naturelle de tenter d'utiliser ces nouvelles fonctionnalités de sécurité pour les cartes bancaires, afin de venir à bout avec les risques de sécurité liés à l'utilisation croissante des cartes à bande magnétique.

Les cartes à puce sont principalement utilisés pour fournir une autorisation pour des actions spécifiques ou d'identifier le titulaire de la carte.

2.9.1. Algorithme cryptographiques utilisées pour les cartes à puce [5, 14, 15]

Les algorithmes cryptographiques actuellement utilisés dans les cartes à puce sont : DES, AES, RSA et ECC (voir Figure 2.5). Ces algorithmes sont implémentés en logiciel (software) ou bien matériel (crypto processeur).

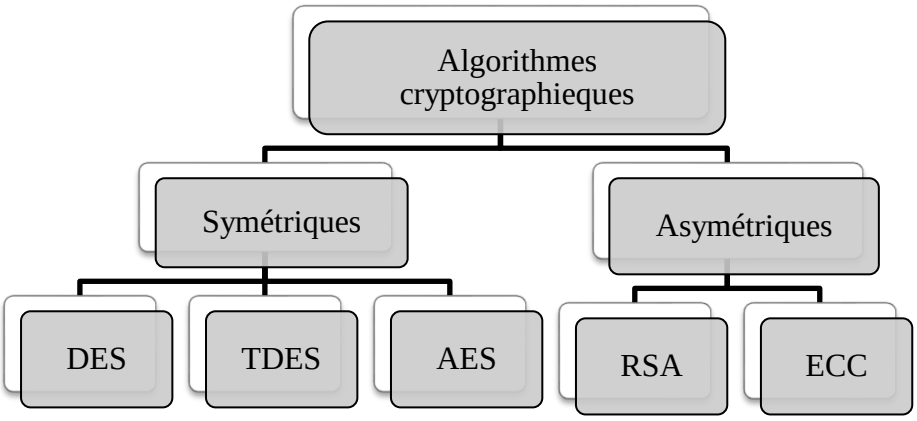


Fig.2. 5: Classification des algorithmes cryptographiques utilisés dans l'environnement de carte à puce.

2.9.2. Crypto processeur des cartes à puce [5, 8, 17, 18]

Il y a des exigences spécifiques pour les cartes à puce qui ne peuvent pas être satisfaites en utilisant du matériel de microcontrôleur classique et ne peut pas être pleinement satisfait en utilisant un logiciel, ils doivent donc être satisfaites par le matériel supplémentaire. Par conséquent, les fabricants de microcontrôleurs de cartes à puce offrent une large gamme de fonctions supplémentaires sous la forme de matériel sur puce.

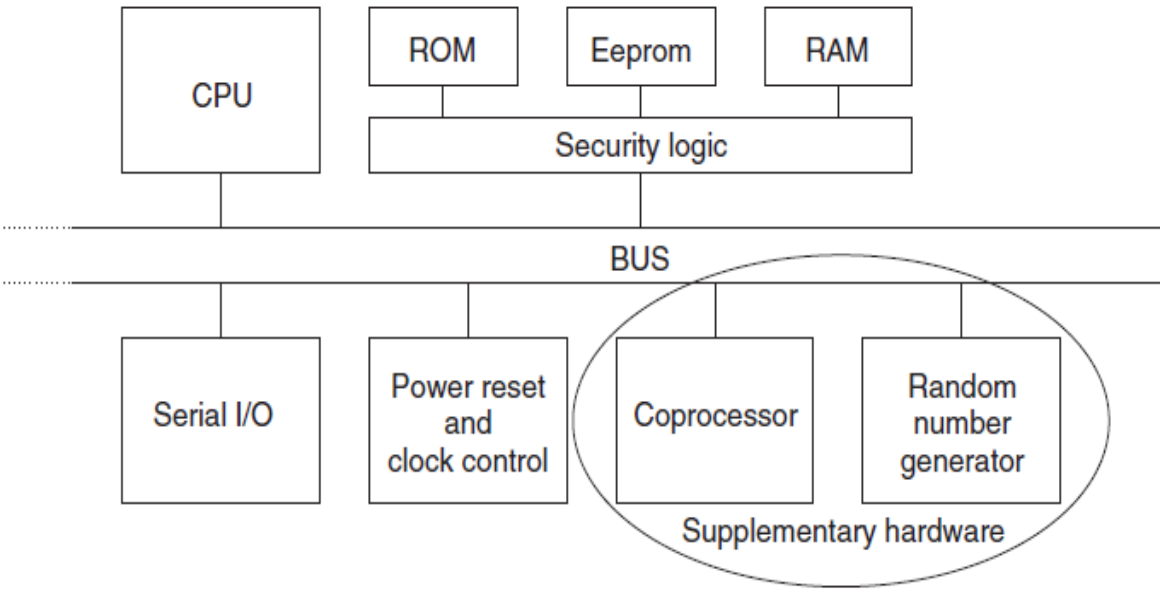


Fig.2. 6: Architecture d'une carte à puce avec un coprocesseur [17].

Les composants les plus couramment utilisés pour les fonctions supplémentaires sont décrits dans la figure 2.6. Il n'est en général pas nécessaire que tous ces composants soient présents dans un microcontrôleur particulier. Les composants qui sont présents dépendent fortement de l'application cible, entre autres choses. Par exemple, il serait économique d'inclure un coprocesseur RSA dans un microcontrôleur dont l'application cible utilise des algorithmes cryptographiques symétriques. Cependant, il existe quelques microcontrôleurs disponibles dans le commerce qui comprennent presque tous les composants décrits ci-dessous.

2.9.3. Coprocesseur pour les algorithmes cryptographiques symétriques [5, 8]

Jusqu'à présent, le DES a été utilisé comme algorithme de chiffrement standard pour les systèmes de paiement et des applications de télécommunication. Avec ce potentiel de marché, il est intéressant pour les fabricants de semi-conducteurs d'équiper les microcontrôleurs de cartes à puce avec leurs propres unités de calcul DES.

Il y a aussi un nombre croissant de processeurs pour l'algorithme AES disponibles dans les microcontrôleurs de cartes à puce. Ils soutiennent généralement les trois longueurs de clés possibles (128, 196, et 256 octets). En termes de faisabilité technique, ils sont comparables aux coprocesseurs DES parce que l'algorithme AES est aussi relativement facile à mettre en œuvre dans le matériel.

Les coprocesseurs de ce genre augmentent la fréquence d'horloge et donne une réduction proportionnelle du temps de calcul. En outre, une unité de calcul intégrée DES ou AES dans le microcontrôleur ne nécessite pas de manière significative une plus grande surface de la puce à l'espace de mémoire ROM ou Flash nécessaire pour un algorithme DES ou AES implémenté dans le logiciel, donc ils n'augmentent pas l'encombrement global sur la puce.

2.9.4. Coprocesseur pour les algorithmes cryptographiques asymétriques [5, 8]

Pour les calculs dans le domaine des algorithmes à clé publique tels que RSA et des algorithmes de courbe elliptique, ils sont spécialement conçus en unités de calcul qui sont situés sur le silicium associés aux composants habituels d'un microcontrôleur de carte à puce.

Ces unités de calcul sont limitées à effectuer des calculs de base qui sont nécessaires pour ces types d'algorithmes : exponentiation et de calcul de modulo sur un grand nombre. Ces deux opérations sont des éléments essentiels des algorithmes de chiffrement à clé publique, comme RSA et des algorithmes de courbe elliptique.

La vitesse de ces composants, qui sont optimisés pour ces deux opérations arithmétiques, les résultats de leurs très larges architectures et des taux d'horloge élevée. Dans leur domaine d'application spécifique, certains d'entre eux peuvent même surpasser un PC de haute performance.

L'unité de calcul est appelée par le processeur, qui soit transmet les données directement ou passe un pointeur vers les données, puis émet une instruction pour démarrer le calcul. Une fois la tâche terminée et le résultat a été stocké dans la RAM, le contrôle de la puce est retournée au processeur. En général, ces processeurs peuvent gérer toutes les longueurs de clés jusqu'à 1024 bits pour l'algorithme RSA, et à moyen terme, cela va augmenter à 2048 bits. Avec les courbes elliptiques, les capacités de l'ordre de 160 à 256 bits sont actuellement communes.

2.10. Comparaissions de différente implémentation du DES sur FPGA

Pas mal de travaux ont été fait afin d'avoir une meilleure implémentation de l'algorithme cryptographique DES sur PFGA et cela dans deux objectifs : réduire l'espace occupé sur l'FPGA et augmenter la vitesse d'exécution.

N°	Auteurs	Dispositif utilisé	Nombre de CLB utilisés	CLB utilisés (%)	Fréquence (MHz)	Débit (Mbits/s)
1	K.M.A Abd El-Latif et al. [19]	XC3S500E	2062	44	124.73	7983
2	V.Patel et al. [20]	XC3S500E	2814	60	111.882	7160
3	Leonard et al. [21]	XC4013	520	51	6.6	26.4
4	Wong et al. [22]	XC4013E	438	55	10	26.7
5	Wong et al. [22]	XC4020E	432	76	10	26.7

6	Kaps and Paar [23]	XC4028EX	741	72	25.18	402.7
7	Patterson [24]	XCV150	1584	41	168	10752
8	Free-DES [25]	XCV400	5263	49	47.7	3052
9	S. Vajpayee et al. [26]	XCV400E	117	1	68.05	274
10	McLonne and Mccanny [27]	XCV1000	6446	23	59.5	3808

Tab.2. 4: Caractéristiques de quelques implémentations de l'algorithme DES sur FPGA.

Les auteurs des travaux [19-27] ont essayé d'augmenter la vitesse d'exécution. Afin de choisir la bonne implémentation FPGA pour une carte à puce, nous avons fait une étude comparative sur des travaux existants selon le critère de l'espace occupé par l'algorithme sur la carte FPGA (nombre de CLB utilisé par rapport à l'espace de la carte FPGA (%)) ; car la carte à puce est dotée d'un espace très réduit. Ce qui fait que nous allons choisir une implémentation à la fois rapide et qui occupe moins d'espace.

Le tableau 2.4 montre les performances des travaux sélectionnés. Nous avons complété le tableau en calculant les valeurs qui manquent (débit, CLB utilisés (%)).

Nous avons comparé ces différentes implémentations selon la famille de la carte FPGA car le nombre de CLB est très proche.

Nous avons commencé par la famille Spartan Xilinx utilisée par les travaux N° (1- 6).

Les travaux N°1 et 2 ont utilisé la même carte XC3S500E. Le 1er est plus rapide que le 2ème avec une fréquence de 124,73 MHz et occupent moins d'espace (44% de bloc CLB) ainsi, la 1ère implémentation est bien meilleure pour une application de carte à puce en vitesse et en espace utilisé.

Concernant la 3ème et 4ème implémentation, les auteurs ont choisi la carte XC4013. Nous pouvons voir que la 4ème est plus rapide que la 3ème avec une fréquence de 10 MHz, mais la 3ème occupait moins de CLB (51% utilisé CLB), ce qui fait que la troisième implémentation est mieux pour une carte à puce selon le nombre de CLB utilisées.

En comparant les implémentations N° (1-6) sur la famille Spartan, nous remarquons que le 1er travail utilise moins de CLB que les autres (44%) avec une fréquence de 124,73 MHz, il est donc le meilleur pour une carte à puce.

Pour la famille Virtex, nous trouvons les travaux N° (7-10). Nous pouvons choisir le 9ème avec 1% CLB utilisés et une faible fréquence de 68,05 MHz.

Mais si nous travaillons avec une carte à puce avec de bonnes performances ; à la fois suffisante en espace et rapide, parmi ces implémentations, nous choisissons la 7ème application avec un débit de 10752 Mbits / s mis en œuvre sur la carte XCV150 FPGA de Xilinx.

2.11. Le système RSA biométrique [5, 6, 8, 28]

La génération de clé biométrique est définie comme le processus de crypter des données biométriques binaire avec l'algorithme cryptographique RSA. Ce processus est composé de deux modules principaux : le module d'authentification d'empreintes digitales et le module cryptographique RSA. Comme le montre la figure 2.7.

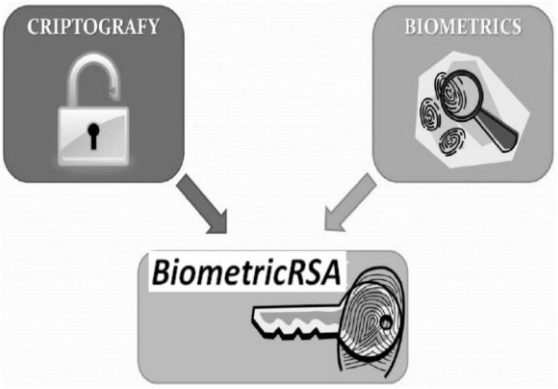


Fig.2. 7: RSA Biométrique [28].

La vérification des empreintes digitales c'est la vérification de l'authenticité d'une personne par ses empreintes digitales. Dans ce cas, La carte à puce de l'utilisateur contient son empreinte digitale. Le système de vérification obtient le modèle d'empreinte digitale selon le numéro d'identification et correspond au modèle avec l'empreinte digitale acquise.

2.12. Pour quoi a-t-on confiance en la sécurité de la carte à puce ? [15]

Tout système ou produit issu des Technologies de l'Information peut être le sujet d'une procédure de certification, menée par un organisme habilité, à l'issue de laquelle un certificat (selon une norme bien définie) est délivrée. On dit alors que le produit est conforme à la norme considérée. Ce processus de certification peut être appliqué pour une carte à puce. L'émetteur de la carte ou son utilisateur final peuvent alors s'appuyer sur un tel certificat afin de renforcer le degré de confiance qu'ils auront dans la carte.

Le processus de certification consiste en une procédure d'évaluation sécuritaire menée par des organismes reconnus et réalisée dans un cadre officiel. Il s'agit d'une procédure relativement complexe dont l'objectif est d'établir le niveau de confiance que pourra avoir un utilisateur final en la sécurité d'un produit. Le produit doit donc être conforme à une norme donnée spécifiée et satisfait l'ensemble des règles de sécurité définies par cette norme.

En France, les cartes à puce sont évaluées par des CESTI (Centre d'évaluation de la Sécurité des Technologies de l'Information) et sont certifiées par la DCCSI (Direction Centrale de la Sécurité des Systèmes d'Information). Ainsi, un utilisateur final peut être certain qu'il possède une carte sûre, et qu'il ne court pas de risque majeur même si sa carte a été la cible d'attaques.

Il existe en effet plusieurs normes dans le contexte des Technologies d'Information. Citons par exemple la norme ISO/CEI 17799 qui établit les principes généraux pour la mise en œuvre et l'entretien de la sécurité au sein d'un organisme. On trouve également la norme FIPS 140 qui décrit un ensemble de règles de sécurité (telles que la gestion des clés, l'utilisation de certains algorithmes, etc.) que doit satisfaire un module cryptographique intégré à un système de sécurité. Il existe aussi la norme ISO-15408 connue sous le nom de Critères Communs. Cette certification (ISO-15408) propose 7 niveaux d'assurance (EAL : Evaluation Assurance Level) fondés sur des tests fonctionnels et structurels, des vérifications conceptuelles et des méthodes de validation formelle.

Plusieurs modèles de cartes (tels que Gemplus ou JCOP) disposent aujourd'hui de certificats qui concernent le processeur et le système d'exploitation de la carte.

2.13. Conclusion

Les cartes à puce sont souvent utilisés dans différentes applications qui nécessitent une protection forte de sécurité et d'authentification, tels que l'organisation des entreprises, le gouvernement, banque, etc. sa popularité augmente en raison de son prix et de sa sécurité.

La cryptographie est ainsi devenue l'un des aspects clés de cartes à puce à un stade précoce de leur développement. Les méthodes et les algorithmes de cette discipline sont maintenant fermement établis comme composants de la technologie de carte à puce.

Dotées d'un espace et capacité de calcule réduits, les cartes à puce utilisent les algorithmes cryptographiques symétriques jusqu'à présent, vu l'espace occupé par ces algorithmes et leurs grande vitesse de calcule par rapport aux algorithmes cryptographiques asymétriques.

Au cours de ce chapitre, une étude comparative de différentes implémentations FPGA de l'algorithme cryptographique DES a été menée selon le critère d'espace occupé par le circuit (nombre de CLB), cela afin de choisir la bonne implémentation pour une carte à puce à la fois rapide et occupe un espace très réduit.

.

Chapitre 3

Programmation et simulation des protocoles de communication utilisés par les cartes à puce

Certains des travaux présentés dans ce chapitre ont fait l'objet d'un article présenté lors de la conférence [ICMIC 2015].

3.1. Introduction

Afin que les éléments communicants puissent se comprendre, il est nécessaire d'établir un protocole de transmission. Ce protocole devra être le même pour les deux éléments pour que la transmission fonctionne correctement.

Dans ce chapitre, nous avons proposés des techniques afin de programmer et simuler les deux protocoles de communication les plus connus des cartes à puce à contacts : le protocole RS 232 et le protocole I²C ainsi que le protocole de transmission, le plus utilisé jusqu'à présent, T= 0. Cela pour une meilleure compréhension de ces échanges. En plus de ça, nous avons optimisé l'espace mémoire utilisé en programmant ces protocoles en langage assembleur sous l'environnement MPLAB IDE V.8.56 de Microchip. Nous avons utilisé l'outil ISIS du logiciel PROTEUS V7 pour les simulations.

3.2. Protocoles de communication des cartes à puce [1, 2, 5, 9]

Les cartes à puce sont divisées en deux groupes, qui diffèrent dans leurs fonctionnalités et leurs prix : cartes à mémoire et cartes à microcontrôleur. Pour écrire et lire des données sur une carte à puce ou exécuter une commande, il est nécessaire d'avoir une connexion physique avec la carte. Pour communiquer avec une carte à puce à contacts il faut l'insérer dans un lecteur ou dans un terminal. La figure 3.1présente quelques protocoles de transmission utilisés par ces cartes.

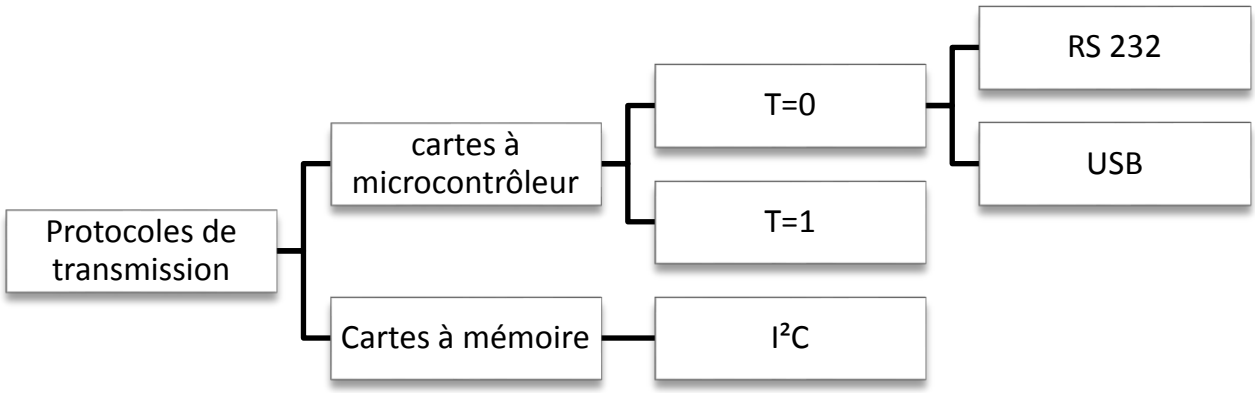


Fig.3. 1: Classification de quelques protocoles de transmission utilisés par les cartes à puce à contacts [5].

Pour la communication d'une carte à puce à mémoire avec un lecteur /terminal, un seul protocole de communication est utilisé, il correspond au protocole utilisé par la mémoire de la carte à puce.

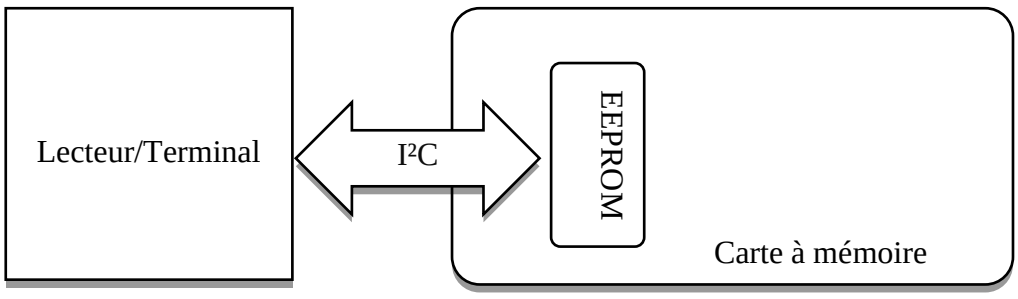


Fig.3. 2: Protocole de communications utilisé par une carte à puce à mémoire EEPROM I²C.

La figure 3.2 présence un exemple d'une carte à mémoire EEPROM. Le protocole de communication utilisé par ce type de mémoire est le protocole I²C. Il permet de lire/écrire des octets depuis/dans la mémoire EEPROM.

Pour la communication d'une carte à puce à microcontrôleur avec un lecteur /terminal, deux protocoles de communication sont utilisé entre la carte et le lecteur. Ils correspondent aux protocoles de transmission d'octets qui peuvent être le RS232 ou l'USB et au protocole utiliser pour les commandes et réponses APDUs dont le T=0 est le plus utilisé.

La figure 3.3 présence un exemple d'une carte à microcontrôleur associé à une mémoire EEPROM. Le protocole RS232 assure la communication entre le microcontrôleur de la carte et le lecteur/terminal au niveau octets ; et le protocole T=0 pour l'échange de commandes. Le protocole I²C est utilisé entre le microcontrôleur et la mémoire EEPROM de la carte.

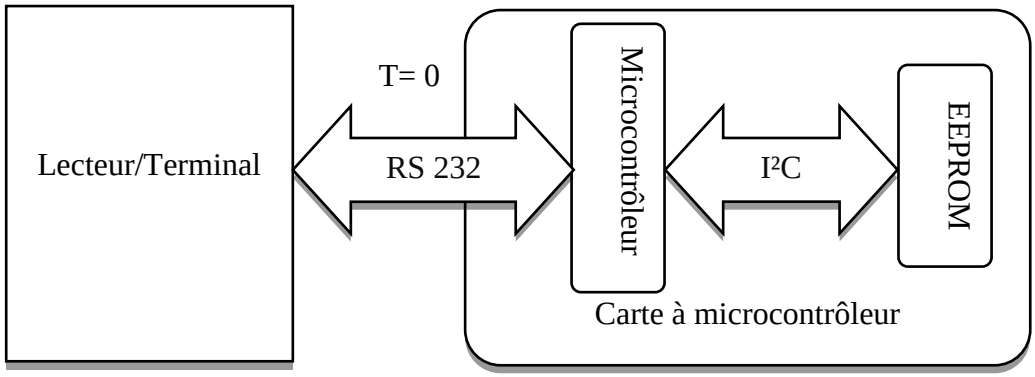


Fig.3. 3: Protocoles de communications utilisées par une carte à puce à microcontrôleur.

3.3. Description du système de simulation [29-36]

Dans ce travail, nous allons programmer et simuler les protocoles de communication utilisés par une carte à puce à mémoire I²C et une carte à puce à microcontrôleur. Pour cela, nous devrons se familiariser avec ces cartes à puce ainsi qu'avec leurs composants.

3.3.1. Les cartes à puce utilisées

3.3.1.1. La carte à microcontrôleur : Wafer 2

Fig.3. 4: Carte à puce Wafer 2.

Carte à puce Wafer 2, Appelée également Gold 64 en raison de la couleur dorée de certaines versions (voir figure 3.4). Comme le montre la figure 3.5, elle contienne un microcontrôleur PIC 16F84a de Microchip ; et une mémoire EEPROM 24LC64 d'une capacité de 8 K octets. C'est une carte vierge de tout programme ou données.

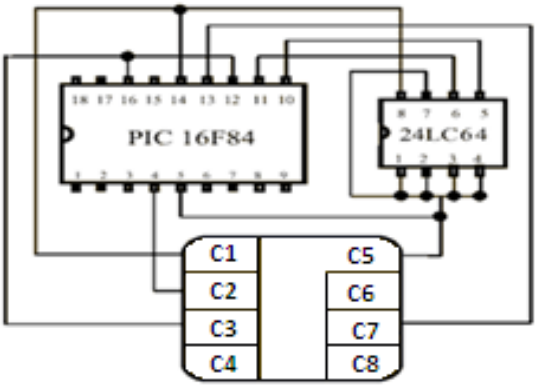


Fig.3. 5: Composition de la carte à puce Wafer 2 [37].

Cette carte possède 8 contacts et parmi eux, on trouve deux réservés à une future utilisation. Le tableau 3.1 donne les fonctionnalités ces contacts.

Position	Abréviation technique	Opération
C1	Vcc	Tension d'alimentation
C2	MCLR	Remise à zéro
C3	CLK	Fréquence d'horloge (Entrée)
C4, C8	RFU	Réservé à une Future Utilisation
C5	Vss	Masse
C6	Vpp	Tension de programmation de l'EPROM
C7	E/S	Données série (Entrée / Sortie)

Tab.3. 1: fonctionnalités des contacts de la carte à puce Wafer 2 [5].

3.3.1.2. La carte à mémoire : 24LC64

Nous utilisons la carte à mémoire I²C nommé : 24LC64 de 8K octets (voir figure 3.6). La mémoire de cette carte est divisée en deux parties : une région ROM et une zone EEPROM. Ces deux mémoire son adressable en lecture en plus de ça, la mémoire EEPROM permet l'écrire contrairement à la mémoire ROM . Elle est complètement vierge de n'importe quelle donnée.

Comme le montre la figure 3.7, cette carte possède 8 contacts et parmi eux on trouve uniquement cinq utilisés. Le tableau 3.2 donne les fonctionnalités ces contacts.

Fig.3. 6: La carte à puce 24LC64 [30].

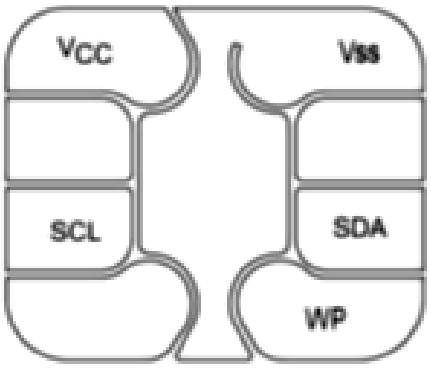


Fig.3. 7: Contactes de la carte à puce 24LC64 [30].

Position	Abréviation technique	Opération
C1	Vcc	Tension d'alimentation
C5	Vss	Masse
C3	SCL	Fréquence d'horloge (Entrée)
C7	SDA	Données série (Entrée / Sortie)
C8	WP	Protection contre l'écriture (Entrée)

Tab.3. 2: Fonctionnalités des contacts de la carte à puce 24LC64 [30].

3.3.2. Le microcontrôleur PIC 16F84A

Le PIC16F84 est un microcontrôleur performant à plusieurs titres : architecture RISC, fréquence de travail élevée, simplicité de sa structure matérielle. Il conviendra aisément aux applications les plus simples, comme aux applications les plus évoluées, malgré l'absence de certains périphériques. Il est commercialisé dans un boîtier classique de 18 broches.

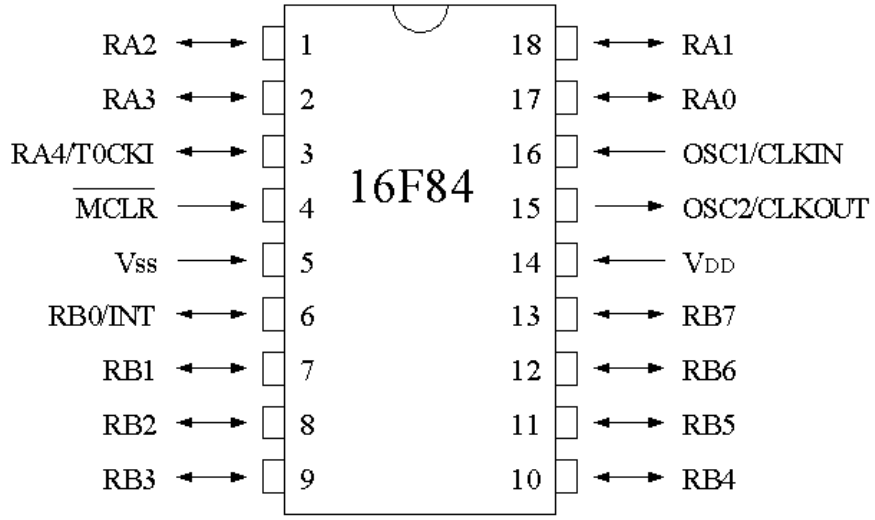


Fig.3. 8: Le microcontrôleur PIC 16F84 [31].

La figure 3.8 montre le brochage du circuit dont les fonctionnalités sont dans le tableau 3.3. Les caractéristiques du PIC 16F84 sont représentées dans le tableau 3.4. Nous remarquons que ce microcontrôleur ne prend pas en charge les protocoles de communication RS 232 et I²C. Ce qui rond leurs programmation en langage assembleur difficile.

Broches	Fonctionnalités
V_{SS}, V_{DD}	Alimentation
MCLR	Reset : 0V
OSC1, 2	Horloge
RA0-4	PORT A
RB0-7	PORT B
T0CKL	Entrée de comptage
INT	Entrée d'interruption

Tab.3. 3: Fonctionnalités des broches du microcontrôleur PIC 16F84 [34].

Entrées-Sorties	13
Flash	1024 mots (14 bits)
EEPROM	64 octets
RAM	68 octets
Watch dog	Oui, oscillateur dédié
Code protection	Oui
Instructions	35
Type d'oscillateur	Quartz, RC externe
Fréquence maximum	10Mhz
Temps d'exécution	400 ns
Interruptions externes	1
RS 232	-
I²C	-
Timer(s)	1 timer 8 bits
MLI (Modulation de Largeur d'Impulsions)	-
Comparateur analogique	-
CAN (Convertisseur Analogique Numérique)	-
Boîtier	DIP 18
Plage d'alimentation	2.0V à 5.5V
Consommation	<2 mA
Programmation	programmateur
Fonction(s) spéciales	-

Tab.3. 4: Caractéristiques du PIC 16F84 [35].

3.3.3. L'EEPROM I²C : 24LC64

Comme son nom l'indique, une EEPROM (Electrically Erasable Programmable Read Only Memory) est en principe une mémoire « morte » mais programmable et effaçable électriquement. Son grand avantage c'est de pouvoir être effacées de façon partielle, et pas forcément globale. La 24LC64 (voir figure 3.9) possède un espace mémoire de 8 K Octets et utilise le protocole de transmission de données synchrone I²C de deux lignes : une pour la transmission de données bidirectionnelle et l'autre pour le signal d'horloge. Le tableau 3.5 montre les différentes broches de cette mémoire.

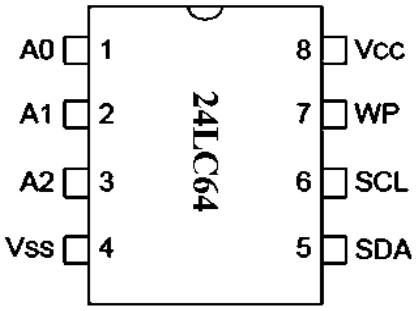


Fig.3. 9: L'EEPROM 24LC64 [29].

Broches	**Fonctionnalités**
V_{SS}	Masse
V_{DD}	Tension d'alimentation
SDA	(Serial Data/Adress) E/S (ligne de transmission de données)
SCL	Serial CLK (signal d'horloge)
WP	Write Protection (protection en écriture)
A0, A1, A2	Adresse physique

Tab.3. 5: Fonctionnalités des broches de l'EEPROM 24LC64 [29].

Lorsqu'il y'a plusieurs mémoires de type EEPROM reliées aux mêmes bus I²C. Il faut tout d'abord désigner celle avec laquelle on communique, pour cela ; les broches A0, A1 et A2 sont utilisées pour donner une adresse physique à chaque composant représentant les trois bits de poids fort de l'adresse interne de la mémoire interne de l'EEPROM. Comme l'adresse physique est sur 3 bits, elle permet notamment de faire relier jusqu'à 8 composants identiques sur un même bus, chacun bénéficiant d'une adresse distincte.

Si la broche 7 (WP : Write Protection) est reliée à la masse, toute la mémoire est accessible normalement en écriture comme en lecture. Mais si elle est connectée à la tension d'alimentation, dans ce cas la mémoire est accessible en lecture et protégée en écriture.

Le tableau 3.6 donne les principales caractéristiques de cette EEPROM.

EEPROM	8 k octets
RAM	32 octets
1 page	32 octets
Bus d'adresse	13 bits
Protection d'écriture	Oui : hardware
Instructions	2 : Lecture, Ecriture
temps d'écriture d'une page	2ms
Fréquence	400 KHz, 100 KHz
Protocole de transmission	I²C
Boîtier	DIP 8
Plage d'alimentation	2.5V à 5.5V
Programmation	programmateur
Durée de vie est	> 200 ans.

Tab.3. 6: Caractéristiques de l'EEPROM 24LC64 [32].

D'après le tableau précèdent, cette mémoire EEPROM est d'une taille de 8 K octets (8x1024=8192 octets). Elle est organisée en 256 pages de 32 octets chaque une. La figure 3.10 présente l'organisation de la 24LC64.

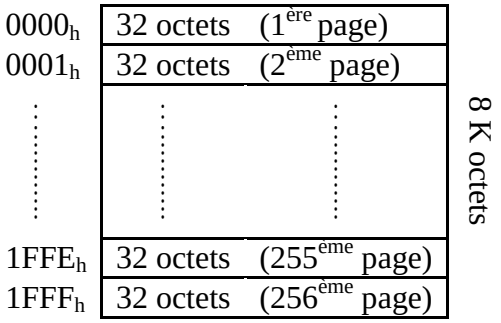


Fig.3. 10: Organisation de l'EEPROM 24LC64.

3.4. Protocole RS232 (Recommended Standard 232) [1, 2, 5]

Les paramètres généraux de la couche physique de transmission des cartes à puce sont spécifiés dans la norme ISO 7816-3. La communication série avec les cartes à puce au niveau électrique est basée sur l'interface RS232.

Les vitesses de transmission des données se mesurent en bauds (1 baud = un bit par seconde). Cette vitesse dépend de la fréquence d'horloge du lecteur de carte à puce.

3.4.1. Le format de la transmission RS232

La transmission des données entre la carte et le lecteur / terminal est asynchrone, l'émetteur ajoute un bit de start au début de chaque octet transmis pour indiquer le début d'une séquence de transmission au récepteur, et à la fin de chaque octet, l'émetteur ajoute un bit de parité pour la détection d'erreur et un ou deux bits de stop. Ceci est clairement illustré sur la figure 3.11.

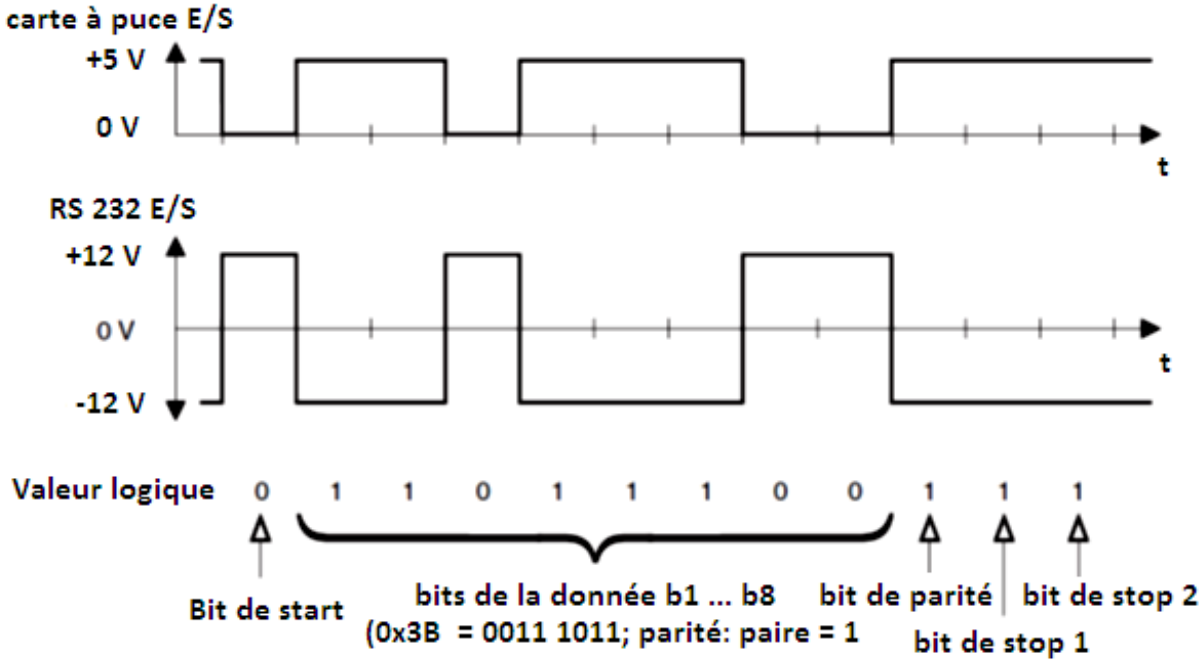


Fig.3. 11: Structure d'un caractère transmit selon le protocole RS232 [5].

La présente du bit de parité ainsi que le 2ème bit de stop est facultative. Donc on peut avoir une transmission sans les ajouter.

- Au repos la ligne de transmission est à l'état logique haut.
- La transmission débute par le passage à 0 de cette ligne pendant une période de l'horloge de transmission ce qui constitue le bit de « start ».
- Les bits du mot à transmettre sont ensuite envoyés derrière ce bit de « start ».
- Après le dernier bit utile, la ligne passe à nouveau à l'état haut pendant une ou deux périodes d'horloge pour constituer le ou les bits de « stop ».

3.4.2. Les niveaux des signaux

Les niveaux de signal sont adaptés aux niveaux de tension d'alimentation habituelles de cartes à puce (+5 V et 0) et pour le RS232 :

- Tout signal de niveau compris entre +3 et +25 V est considéré comme étant à l'état bas.
- alors que tout signal compris entre -3 et -25 V est considéré comme étant à l'état haut.

3.4.3. Les conventions logiques utilisées

Il y'a deux conventions logiques utilisées par le protocole RS232 :

- Convention logique directe :
  - Etat haut correspond à un « 1 » logique.
  - Etat bas correspond à un « 0 » logique.
  - Le bit N°0 est transmis en premier.
- Convention logique inverse :
  - Etat haut correspond à un « 0 » logique.
  - Etat bas correspond à un « 1 » logique.
  - Le bit N°7 est transmis en premier.

3.4.4. La parité

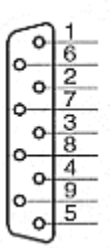
Le bit de parité consiste à compter les bits utiles du caractère transmis (c'est à dire les bits du caractère transmis seulement et non le bit de « start » et le ou les bits « stop») et selon que leur nombre est pair ou impair, on met à 1 ou à 0 le bit de parité. Cette parité peut être paire ou impaire :

- En parité paire : le bit de parité est positionné pour que le nombre total de bits à 1 (y compris celui de parité) soit pair.
- En parité impaire : le bit de parité est positionné pour que le nombre total de bits à 1 (y compris celui de parité) soit impair.

Pour les cartes à puce, la parité utilisée est toujours paire et deux bits de stop.

3.4.5. La norme RS232

La norme RS 232 précise les niveaux électriques des signaux chargés de véhiculer une liaison série asynchrone et y ajoute un certain nombre de signaux de contrôle. Le tableau 3.7 donne les fonctions des lignes de contrôle conjointement au brochage utilisé sur les prises DB9 à 9 points.



N° de broche	nom	Fonction
1	DCD	Détection de porteuse
2	RD	Réception de données
3	TD	Emission de données
4	DTR	Terminal de données prêt
5	SG	Masse électrique
6	DSR	Poste de données prêt
7	RTS	Demande d'émission
8	CTS	Prêt à émettre
9	RI	Indication de sonnerie

Tab.3. 7: Fonctionnalités des broches de la prise DB9.

3.5. Programmation et simulation du protocole RS232

Le protocole RS232 assure la communication entre le lecteur/terminal et la carte à puce à microcontrôleur.

La carte à puce utilisée pour la programmation et simulation du protocole RS232 est la Wafer2. Elle est dotée d'un microcontrôleur PIC 16F84a. Elle est représentée par son microcontrôleur ; car c'est lui qui assure la communication avec le lecteur/terminal.

Nous avons programmé le protocole RS232 en langage assembleur sur le microcontrôleur 16F84a ; sous l'environnement MPLAB de Microchip. Pour la simulation nous avons utilisé le logiciel proteus.

La convention logique directe a été utilisée, sans bit de parité et un seul bit de stop ; car nous souhaitons programmer le minimum. Une fois que ça fonctionne nous pouvons ajouter le bit de parité paire ainsi que le $2^{ème}$ bit de stop facilement.

La vitesse de transmission des données utilisée est 1200 bauds ce qui fait que le temps nécessaire pour transmettre un seul bit est 0.8ms, sachant que le microcontrôleur est cadencé avec un quartz de 4 Mhz.

3.5.1. Schéma de simulation

Le schéma de simulation réalisé sous le logiciel Proteus est présenté dans la figure 3.12.

- Le microcontrôleur PIC 16F84a représente la carte à puce Wafer2 ;
- Le terminal virtuel servira à envoyer/recevoir des caractères vers/depuis le PIC ce qui fait qu'il représente le lecteur/terminal ;
- La ligne RA0 servira à établir la liaison RS232 entre le microcontrôleur PIC 16F84 et le terminal virtuel ;
- Les LEDs serviront à afficher le caractère reçu par le PIC ;

Proteus nos offre la possibilité de simuler le fonctionnement du microcontrôleur sans compléter son montage (alimentation, reset et signal d'horloge). Ces signaux doivent être générés par le lecteur de carte, ce qui fait qu'on considère la carte insérée dans le lecteur.

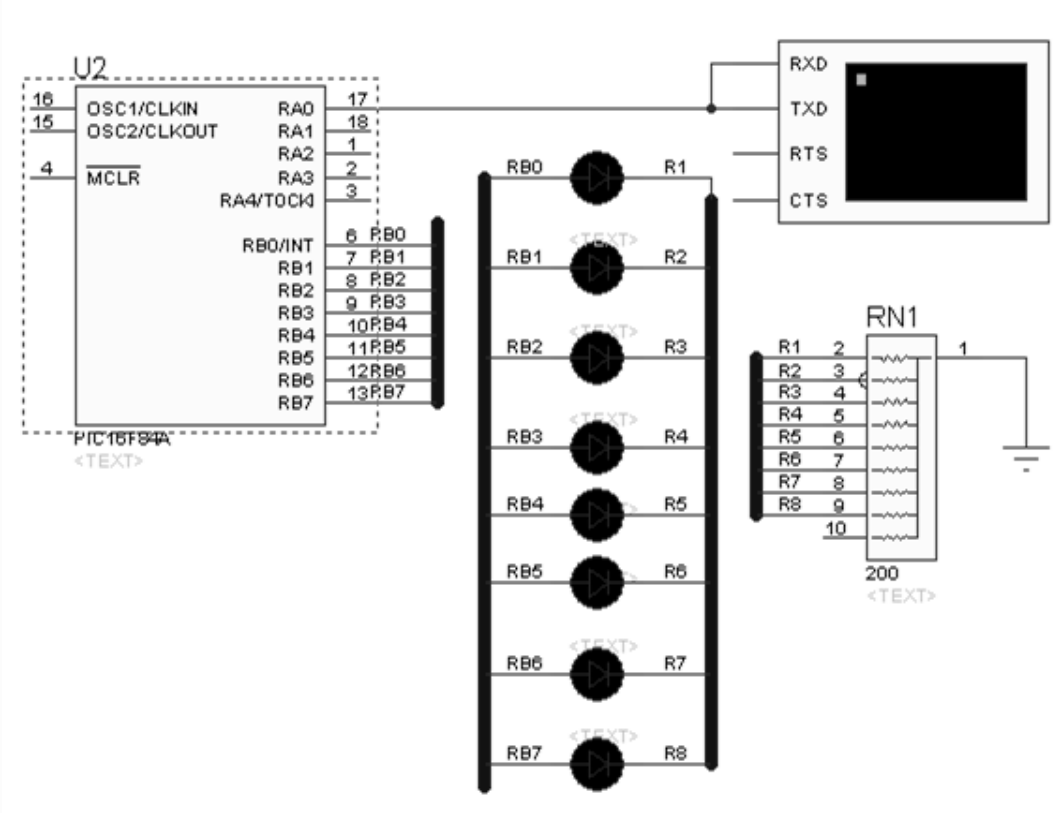


Fig.3. 12: Schéma de simulation du protocole RS232.

Le microcontrôleur doit travailler en deux modes : envoi et réception RS232. Pour cela, nous divisons la programmation en deux parties ; une pour la transmission depuis le terminal virtuel vers le microcontrôleur et l'autre dans le sens inverse.

3.5.2. Transmission depuis le terminal virtuel vers le microcontrôleur

3.5.2.1. Organigrammes

Dans ce cas, le microcontrôleur travaille en mode réception. Le terminal virtuel envoi des caractères en code ASCII vers le microcontrôleur et ce dernier les affiche sur les 8 LEDs connectées au PORTB, cela afin de vérifier que le code binaire affiché sur les LEDS correspond au code ASCII transmit par le terminal.

- **L'algorithme global**

La figure 3 .13 présente l'algorithme globale du programme assembleur sous le microcontrôleur PIC 16F84a. Après la mise sous tension, le PIC commence par une initialisation afin de préparer la ligne de transmission. Ensuite, le microcontrôleur récupère le caractère envoyé par le terminal virtuel pour l'afficher en fin sur les 8 LEDs connectées au PORTB. Les algorithmes secondaires sont présentés dans les figures 4.14 et 4.16.

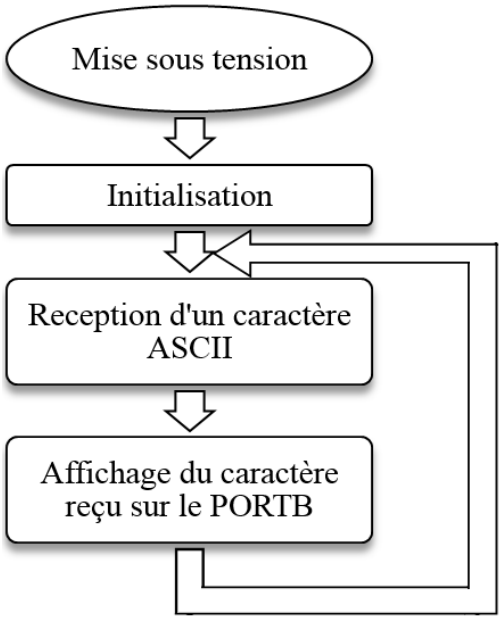


Fig.3. 13: Organigramme du programme de réception RS232.

- **L'algorithme d'initialisation**

D'après le schéma de simulation, c'est la ligne RA0 du PIC qui communique avec le terminal virtuel selon le protocole RS232. Dans la partie initialisation, nous commençons par la déclaration des registres que nous utiliserons dans le programme assembleur.

Variables utilisées sont : databyte ; contient la donnée (8 bits) ;
Bitcount ; conteur de bits ;
TXD ; récupère le contenu du PORTB ;
Temp ; utilisée pour la temporisation.

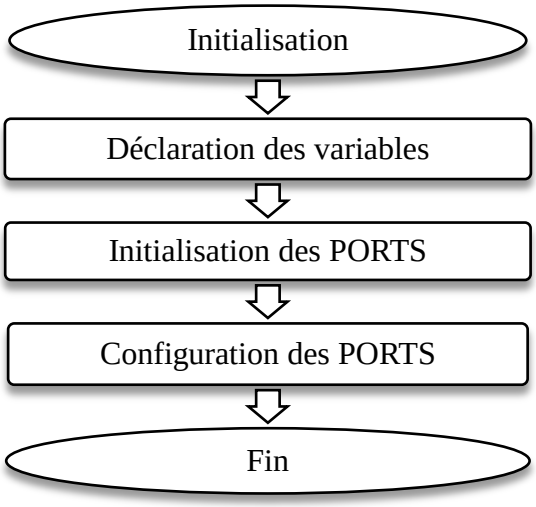


Fig.3. 14: Organigramme d'initialisation du PIC.

Comme la ligne de transmission RS232 est à l'état logique haut au repos (aucune transmission), la ligne RA0 doit être initialisée à l'état logique 1_2. Au départ il y'a aucun caractère a affiché ce qui fait que le PORTB doit être initialisé avec $(00000000)_2$.

Le microcontrôleur est en mode réception, dans ce cas nous devons configurer la ligne RA0 en entrée.

- **Algorithme de réception d'un octet**

En mode réception, nous commençant par initialiser le registre Data_byte avec $(00000000)_2$ car c'est le registre qui va sauvegarder l'octet reçu. En plus de ça, le compteur de bit Bit_count doit être initialisé à 8_{10} car l'octet à recevoir est sur 8 bits.

La ligne RA0 attend la réception d'un changement de son état logique 1_2 vers l'état logique 0_2 qui représente le bit de start. Si un changement est détecté, le PIC attend une durée de ½ bit et refait le teste (voir figure 3.16). Si le 0_2 est toujours présent, nous pouvons dire qu'il s'agit d'un vrai bit de start et non pas d'un bruit. Et s'il s'agit d'un bruit, nous revenons au départ pour attendre la réception d'un bit de start à nouveau.

Supposant qu'un vrai bit de start a été détecté, le microcontrôleur se met à l'attente du 1er bit. Afin d'éviter les perturbations des transitions, une temporisation d'un bit permet de se positionner au milieu du bit reçu.

Comme nous sommes positionnés au milieu du bit, nous pouvons récupérer le contenu du registre PORTA du PIC afin de sauvegarder la valeur reçu sur RA0. Le principe est illustré dans la figure 3.17 comme suit :

- Le contenu du registre PORTA sera déplacer dans le registre TXD ;
- Une rotation du registre TXD à droite permet de déplacer RA0 dans le bit C du registre STATUS ;
- Une rotation du registre Data_byte à droite permet d'entrer la valeur de RA0, qui se trouve dans C, dans le bit N°7 de ce registre.

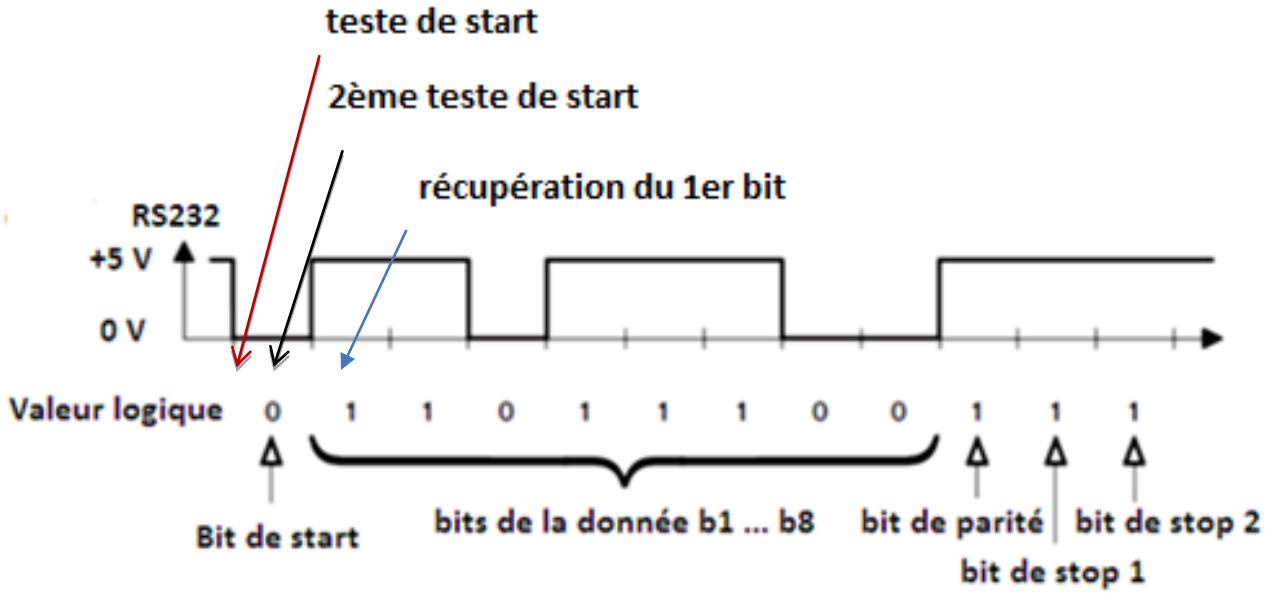


Fig.3. 15: Illustration de la détection du bit de start et du 1er bit.

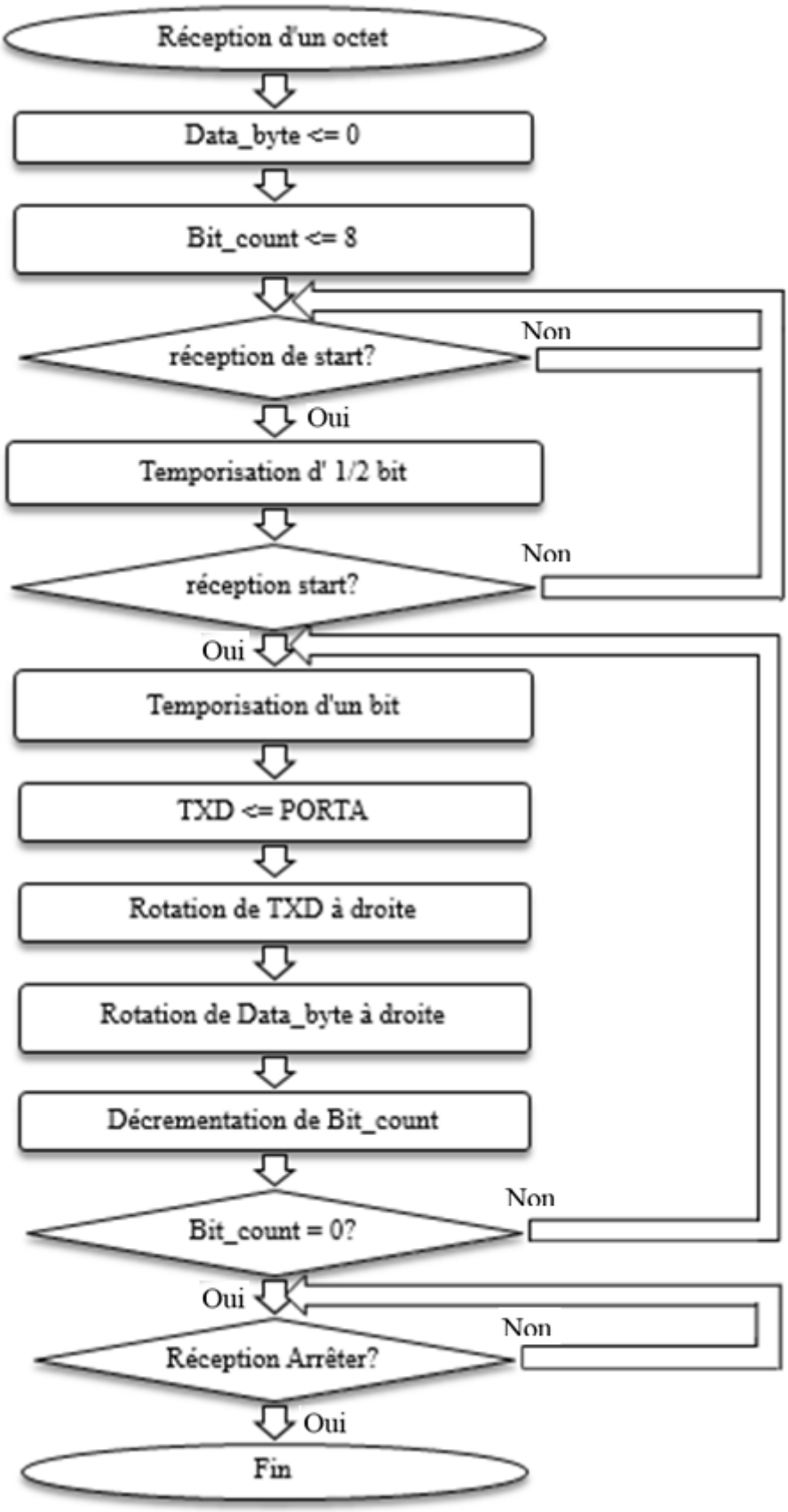
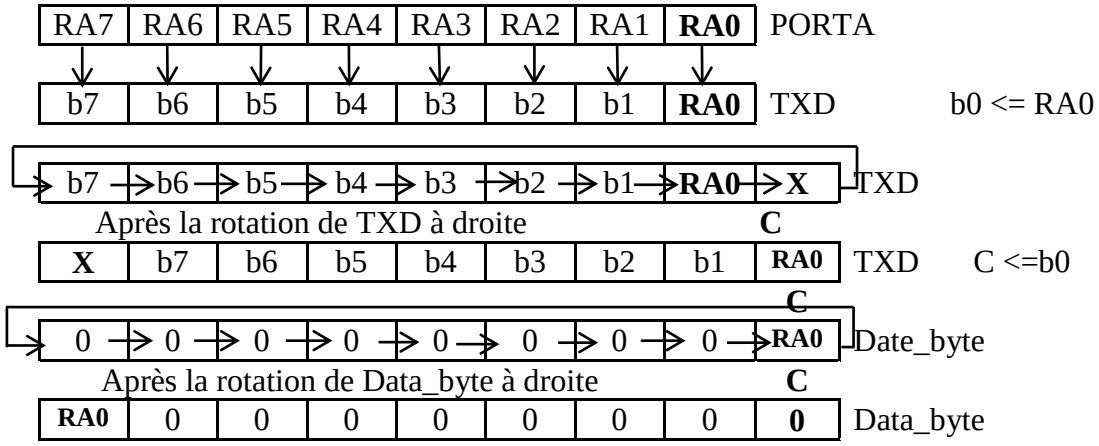


Fig.3. 16: Organigramme du programme de réception d'un octet.



X : peut-être 1_2 ou 0_2

Fig.3. 17: Récupération d'un bit reçu sur RA0 dans le registre Data_byte.

Une fois le bit récupérer, le compteur de bits Bit_count doit être décrémenté. Pour récupérer les autres bits, nous revenons au début faire une autre temporisation d'un bit et les deux opérations de rotations. Une fois que le compteur devient 0_{10}, nous passons à la réception du bit de stop.

3.5.2.2. Simulations

Une fois qu'on lance la simulation le terminal virtuel s'affiche. Les caractères tapés sur ce dernier seront envoyés vers le PIC qui les affiche en binaire sur les 8 LEDs connectées au PORTB comme le montre la figure 3.18.

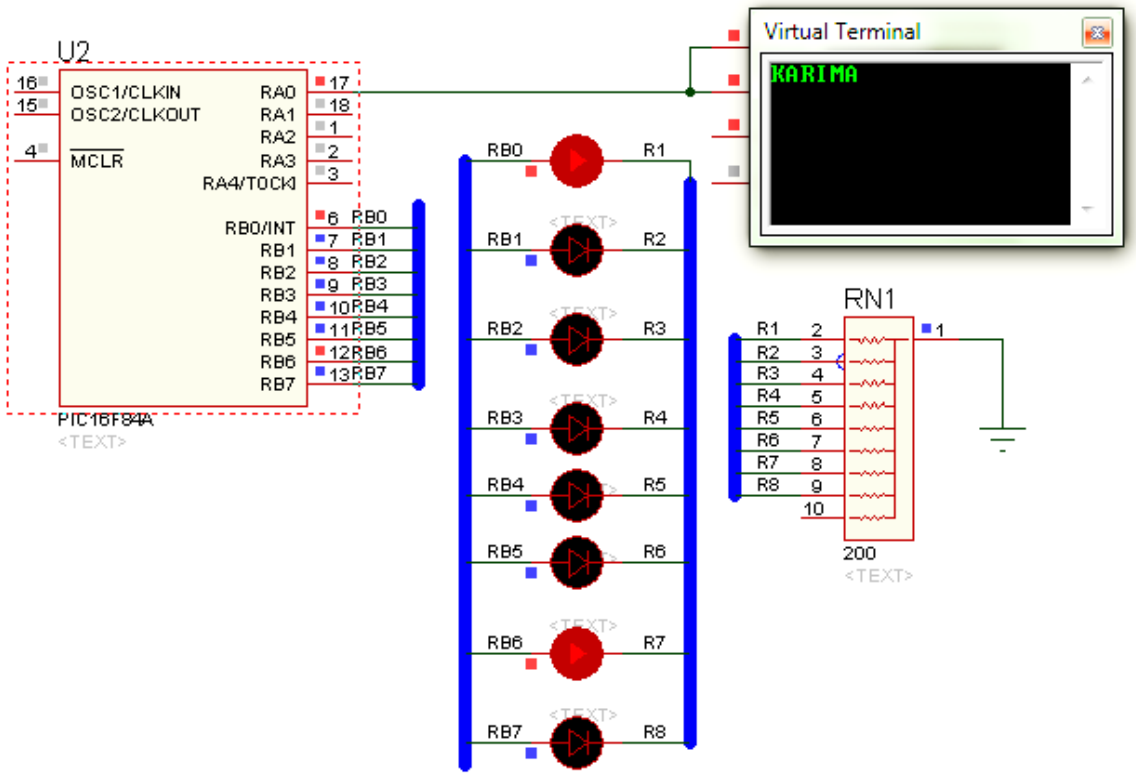


Fig.3. 18: Simulation de la transmission RS 232 depuis le terminal virtuel vers le PIC.

3.5.3. Transmission depuis le microcontrôleur vers le terminal virtuel

3.5.3.1. Organigrammes

Dans ce cas, le microcontrôleur travaille en mode envoi. Le terminal virtuel reçoit des caractères en code ASCII depuis le microcontrôleur, les caractères affichés sur le terminal doivent correspondre à ceux transmis par le microcontrôleur.

- **L'algorithme principal**

La figure 3.19 présente l'algorithme globale du programme assembleur sous le microcontrôleur PIC 16F84a. Après la mise sous tension, le PIC commence par une initialisation afin de préparer la ligne de transmission. Ensuite, le microcontrôleur prépare le caractère à envoyer vers le terminal virtuel. En fin, Il le transmit. Les algorithmes secondaires sont présentés dans les figures 4.20, 4.22.

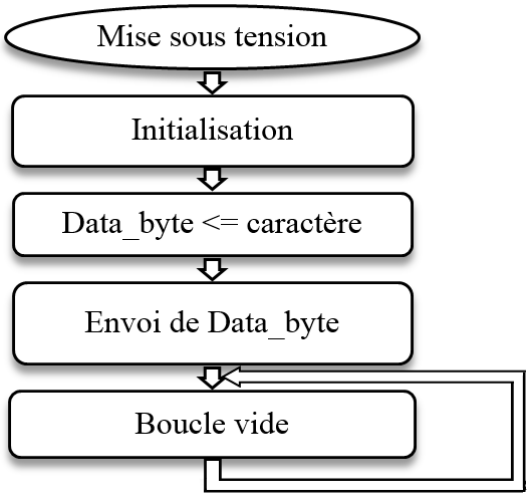


Fig.3. 19: Organigramme du programme de réception RS232 caractère ASCII vers le PIC.

- **L'algorithme d'initialisation**

D'après le schéma de simulation, c'est la ligne RA0 du PIC qui communique avec le terminal virtuel selon le protocole RS232. Dans la partie initialisation, nous commençons par la déclaration des registres que nous utiliserons dans le programme assembleur. Nous avons déclaré les mêmes registres pour l'envoi et pour la réception RS232 car les deux codes source seront implantés sur le même microcontrôleur.

Variables utilisées sont : databyte ; contient la donnée (8 bits)
Bitcount ; conteur de bits
Temp ; utilisée pour la temporisation

Comme la ligne de transmission RS232 est à l'état logique haut au repos (aucune transmission), la ligne RA0 doit être initialisée par 1_2.

Le microcontrôleur est en mode envoi, dans ce cas nous devons configurer la ligne RA0 en sortie.

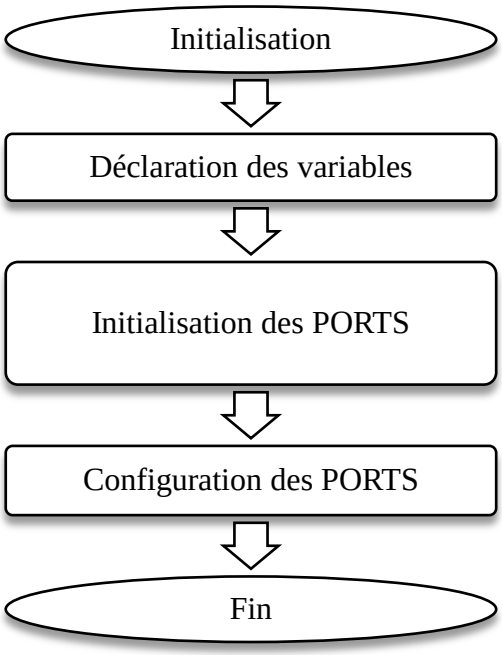


Fig.3. 20: Organigramme d'initialisation.

- **Algorithme d'envoi d'un octet**

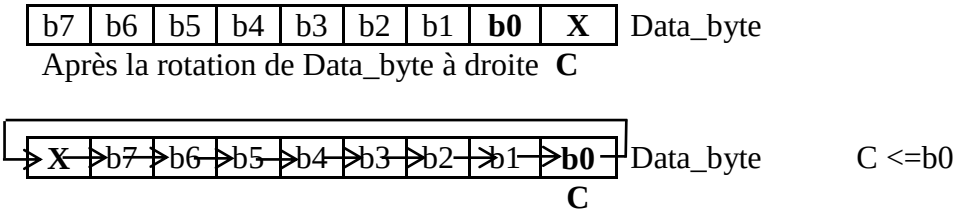
Cette fois, le microcontrôleur travaille en mode envoi. Nous commençons par l'initialisation du compteur de bit Bit_count avec 8_{10} car l'octet à transmettre est sur 8 bits.

La ligne RA0 envoi le bit de start en changeant son l'état logique de 1_2 vers 0_2 avec une temporisation d'un bit.

Afin d'envoyer les bits de l'octet un par un, nous utilisant une opération de rotation dont le principe est illustré dans la figure 3.21, comme suit :

- Une rotation du registre Data_byte à droite permet de déplacer son b0 le bit C du registre STATUS ;
- Selon la valeur qui se trouve dans C, nous changeons l'état de la ligne de transmission RA0 pendant une temporisation d'un bit.

A chaque fois qu'un bit est envoyé, le compteur de bits Bit_count doit être décrémenté. Une fois que le compteur devient 010, nous passons à la réception du bit de stop.



X : peut-être 1_2 ou 0_2

Fig.3. 21: Transmission d'un bit de Data_byte via RA0.

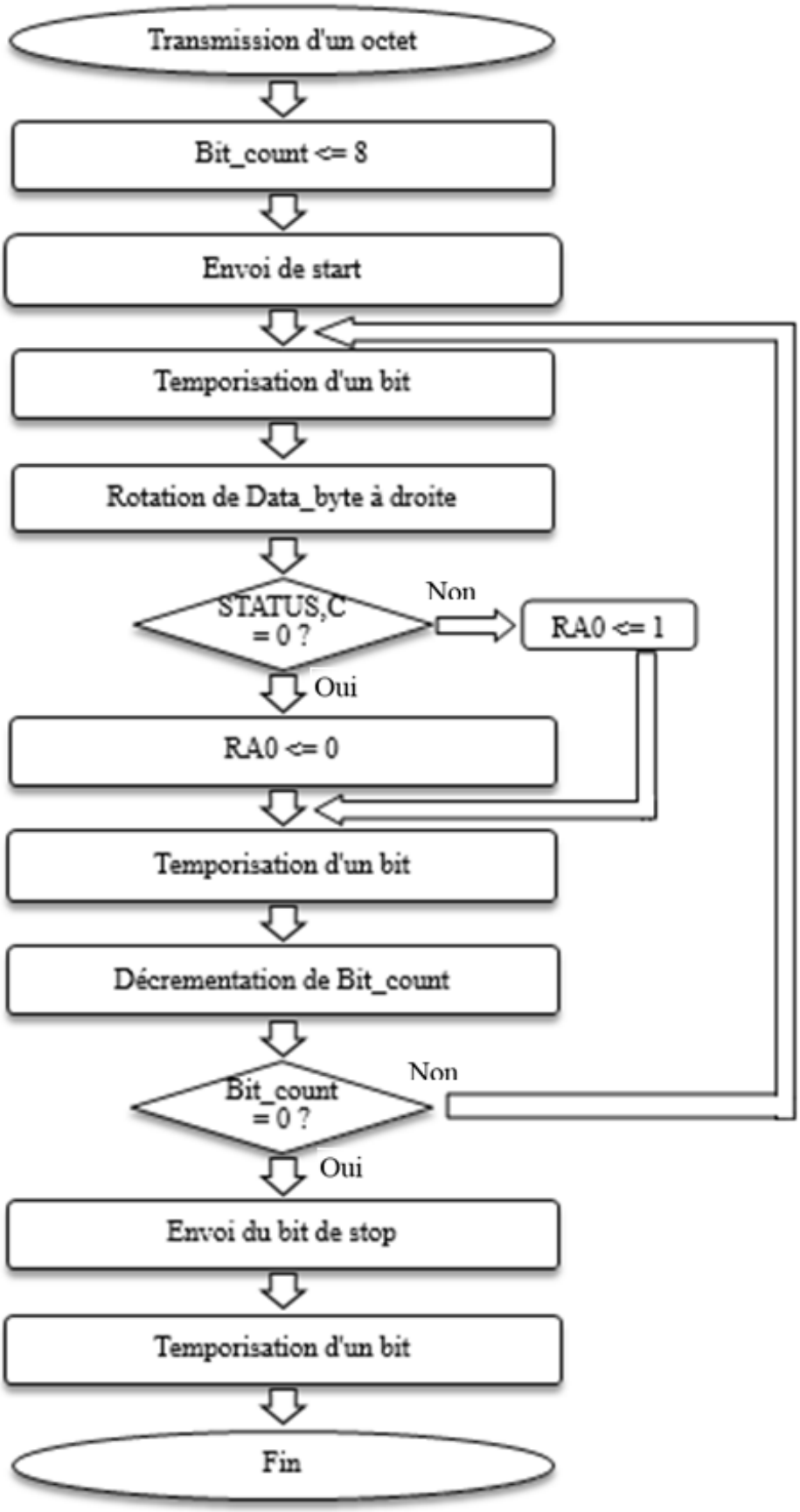


Fig.3. 22: Organigramme du programme de transmission d'un octet.

3.5.3.2. Simulations

Cette fois fois, la simulation se fait dans l'autre sens. Une fois qu'on lance la simulation le terminal virtuel s'affiche. Les caractères sauvegardés dans le PIC seront transmis vers le terminal virtuel, comme le montre la figure 3.23.

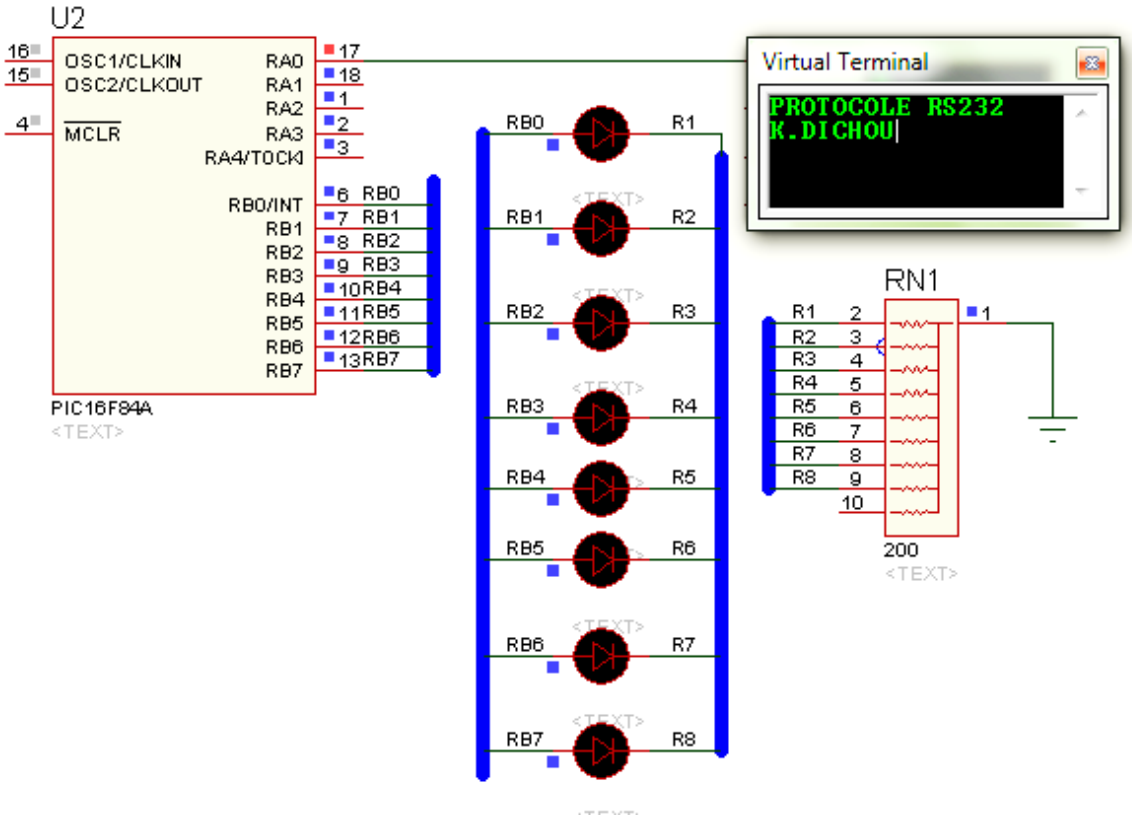


Fig.3. 23: Simulation de la transmission RS 232 depuis le PIC vers le terminal virtuel.

3.6. Protocole I²C [1, 2 , 5, 9, 29, 32]

Le bus I²C (Inter Integrated Circuit) a été proposé initialement par Philips mais est adopté de nos jours par de très nombreux fabricants. Ce protocole qui n'utilise que deux lignes de signal (et les masses correspondantes) permet à un certain nombre d'appareils d'échanger des informations sous forme série avec un débit pouvant atteindre 100 K bits par seconde ou 400 K bits par seconde pour les versions les plus récentes. Même si ces débits peuvent sembler relativement faibles.

La transmission synchrone représente le standard des cartes à mémoire. Ce protocole permet à la mémoire d'être adressé physiquement afin de lire ou d'écrire. Cela signifie que le processus de transmission de données est lié à l'adressage de la mémoire et à des fonctions de gestion de mémoire (lecture, écriture). Les commandes sont envoyées dans le sens maitre vers esclave et les données peuvent être échangées dans les deux sens ; sachant que le maître représente le générateur du signal d'horloge.

3.6.1. Le format de la transmission I²C

Le format général de la transmission I²C est présenté dans la figure 3.24. Il est constitué d'un start suivi par un octet de contrôle, l'adresse à sélectionner, la donnée à lire ou à écrire et en fin un stop. Afin de confirmer la réception d'un l'octet :

- Lors de la transmission maître vers esclave, chaque octet est suivi par un bit d'acquittement ACK envoyé par l'esclave (opération d'écriture).
- Un non acquittement No ACK est envoyé lors de la fin de l'opération de lecture par le maître.

Lorsqu'un maître désire effectuer plusieurs échanges d'adresses différentes, il n'est pas obligé de terminer le premier échange par un stop, On peut les enchaîner en générant un start dès la fin d'un échange.

Start	Octet de contrôle	Adresse	Données	Stop

Fig.3. 24: Format de la transmission I²C.

3.6.1.1. L'octet de contrôle

L'octet de contrôle est le premier octet reçu par l'esclave après la condition de start. La figure 3.25 montre son contenu.

- Il est constitué d'un code de contrôle de 4 bits, pour l'EEPROM 24LC64 c'est $(1010)_2$.
- Les trois prochain bits (A2, A1, A0) permet de sélectionner le dispositif désirer, ces bits sont en effet les bits de poids fort du mot d'adresse ou encore l'adresse physique du composant.
- Le dernier bit défini l'opération à effectuer.
  - 1 : pour lecture
  - 0 : pour écriture

Après la réception de cet octet, l'esclave doit envoyer le bit d'acquittement ACK au maître.

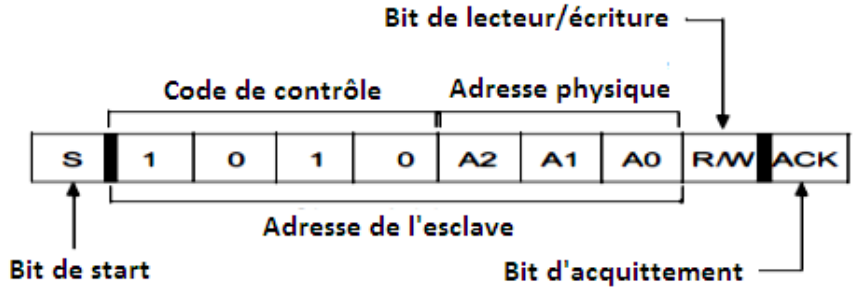


Fig.3. 25: Format du byte de contrôle [29].

3.6.1.2. L'adresse

Après l'octet de contrôle viens la sélection de l'adresse désirée dans le dispositif lui-même. La taille de cette adresse dépend de la mémoire à adresser. Dans notre cas, le protocole I²C est utilisé pour communiquer avec la mémoire EEPROM 24LC64 (voir figure 3.26). Le bus d'adresse de cette mémoire est sur 13 bits. On aura 2 octets : un est sur 8 bits et l'autre avec 5 bits utiles. Les bits d'adresse du poids fort sont transférés en premier, suivi par les bits du poids faible. Après la réception d'un octet d'adresse, l'esclave (l'EEPROM) doit envoyer un bit d'acquittement ACK.

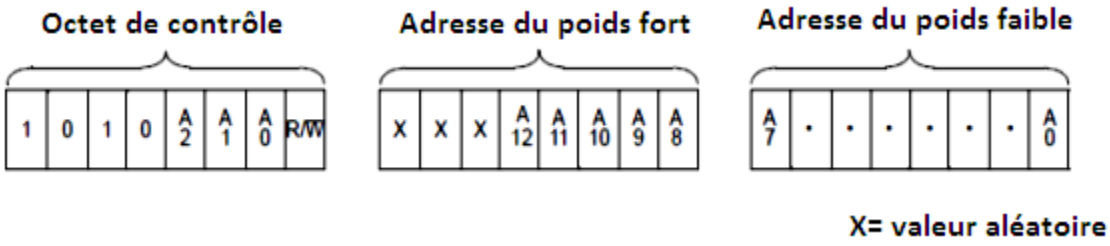


Fig.3. 26: Adressage de l'EEPROM 24LC64 avec le protocole I²C [29].

3.6.1.3. Les données

Après la transmission de l'octet de contrôle et de l'adresse interne du composant sélectionné, selon l'opération (lecture ou écriture), des données seront échangées. Les données sont envoyées par paquets de 8, même si un octet regroupe en fait 8 bits indépendants. Le bit de poids fort est envoyé le premier. La figure 3.27 donne l'exemple d'échange de données avec l'EEPROM 24LC64.

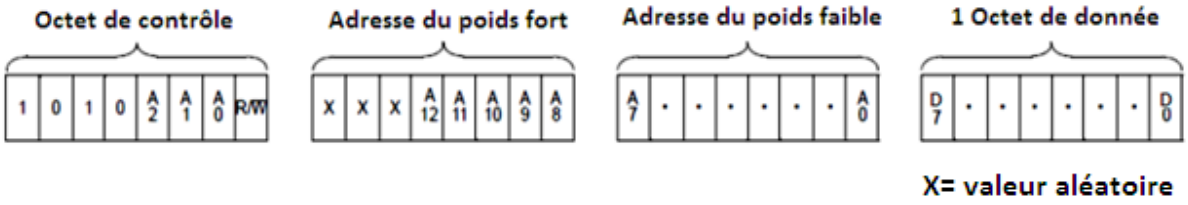


Fig.3. 27: transmission de données de l'EEPROM 24LC64 avec le protocole I²C [29].

Afin de confirmer la réception d'un octet :

- Lorsque de la lecture d'un octet, ce dernier sera suivi d'un bit de non acquittement No ACK envoyé par le récepteur de l'octet (le maître) pour indiquer la fin de la réception.
- Un bit d'acquittement ACK sera envoyé par le récepteur de l'octet (l'esclave) lors qu'une opération d'écriture.

3.6.1.4. Les niveaux des signaux

Le bus I²C est bifilaire :

- Une ligne de données appelée SDA (Serial DAta) ;
- Une ligne d'horloge appelée SCL (Serial CLock).

Le chronogramme de transmission I²C est bien illustré dans la figure 3.28 comme suit :

- Au repos les lignes SDA et SCL sont au niveau haut.
- La ligne SCL est pilotée par l'initiateur de l'échange (le maître).
- Une donnée n'est considérée comme valide sur le bus que lorsque le signal SCL est à l'état haut.
- L'émetteur doit positionner la donnée à émettre lorsque SCL est à l'état bas et la maintenir tant que SCL reste à l'état haut.
- Le bit de start est réalisé lorsque la ligne SDA passe du niveau haut au niveau bas alors que SCL est au niveau haut.
- Réciproquement, le bit de stop est réalisé lorsque SDA passe du niveau bas au niveau haut alors que SCL est au niveau haut.

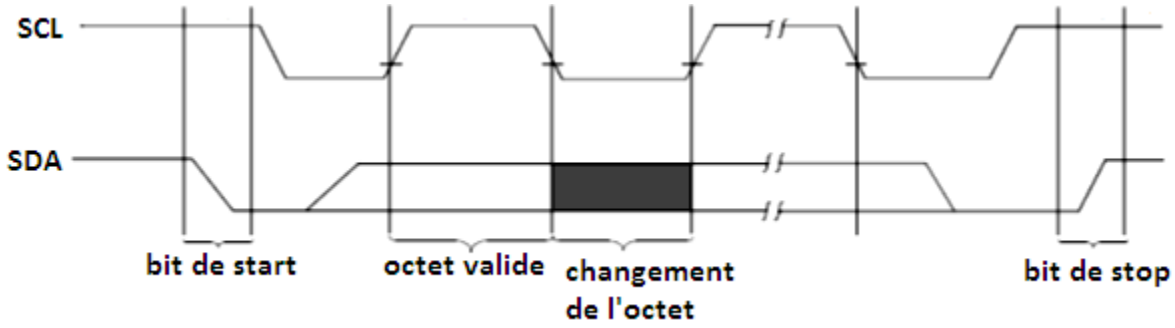


Fig.3. 28: Séquence de transfert de données selon le protocole I²C [29].

Chaque octet est suivi par un bit d'acquittement de la part du destinataire. 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, le récepteur doit alors forcer la ligne SDA au niveau bas pendant l'état haut de SCL qui correspond à l'ACK (donc l'ACK = 0_2) (voir figure 3.29).

Durant la lecture, le maitre doit signaler la fin de la transmission en ne pas générant un ACK durant l'état haut de l'horloge (No ACK=1_2).

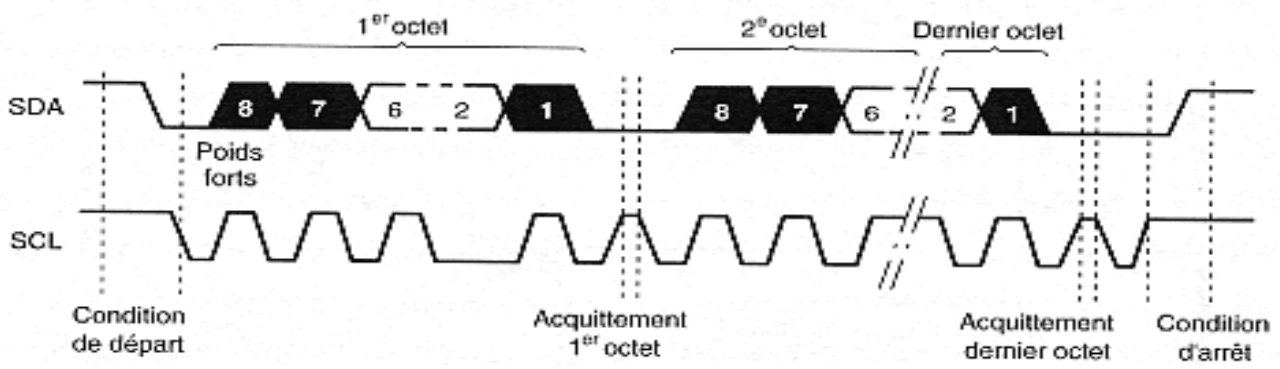


Fig.3. 29: Transfert du bit d'acquittement ACK [29].

3.6.2. Les opérations d'écriture

Il y'a deux formats d'opération d'écriture pour la mémoire 24LC64. Le 1er permet d'écrire un seul octet dans une adresse déterminée, comme l'illustre la figure 3.30. Le 2ème permet d'écrire plusieurs octets à la fois, allons jusqu'à 32 octets (voir figure 3.31).

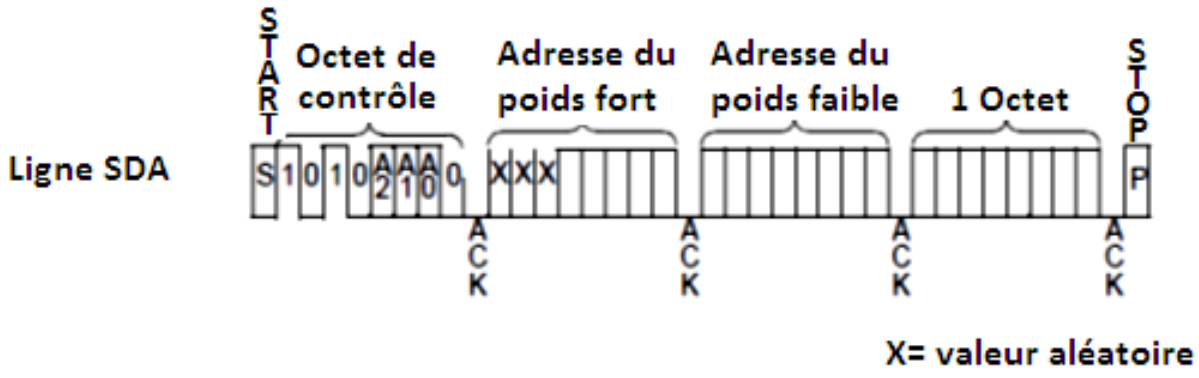


Fig.3. 30: Ecriture d'un octet [29].

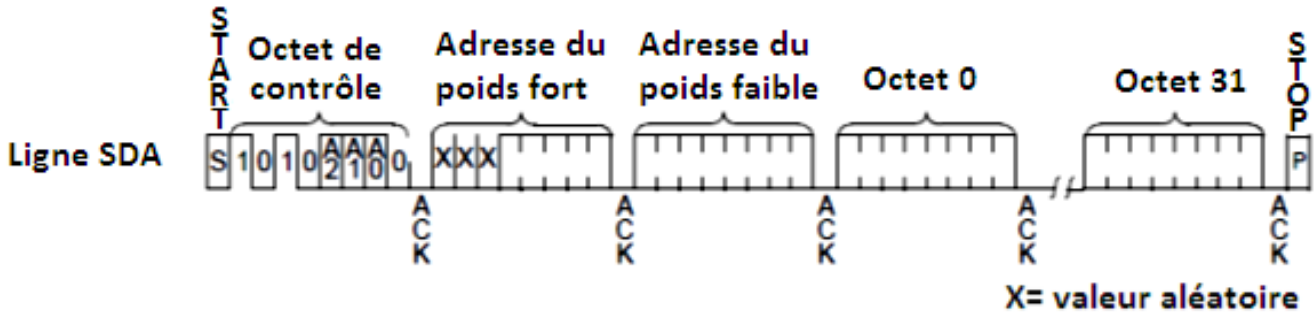


Fig.3. 31: Ecriture d'une page (32 octets) [29].

3.6.3. Les opérations de lecture

Il y'a trois formats d'opération de lecture pour la mémoire 24LC64. Le 1er permet de lire une adresse courante, comme l'illustre la figure 3.32. Le 2ème permet de lire un octet à partir d'une adresse aléatoire (voir figure 3.33) et le dernier format permet de lire plusieurs octets à partir d'une adresse aléatoire (voir figure 3.34).

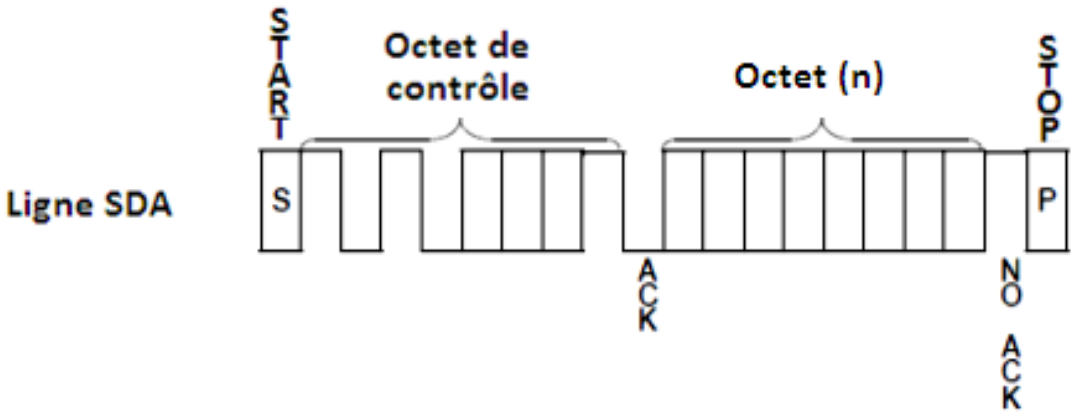


Fig.3. 32: Lecture de l'adresse courante [29].

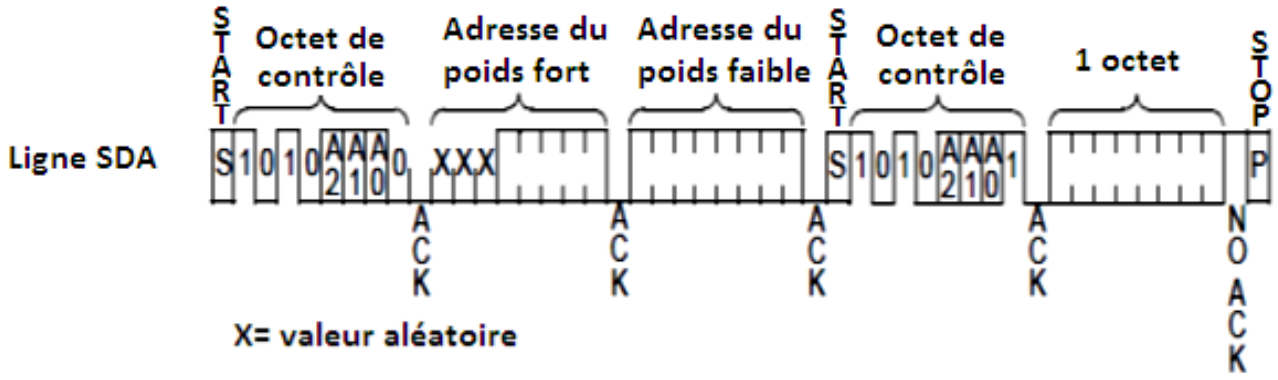


Fig.3. 33: Lecture d'une adresse aléatoire [29].

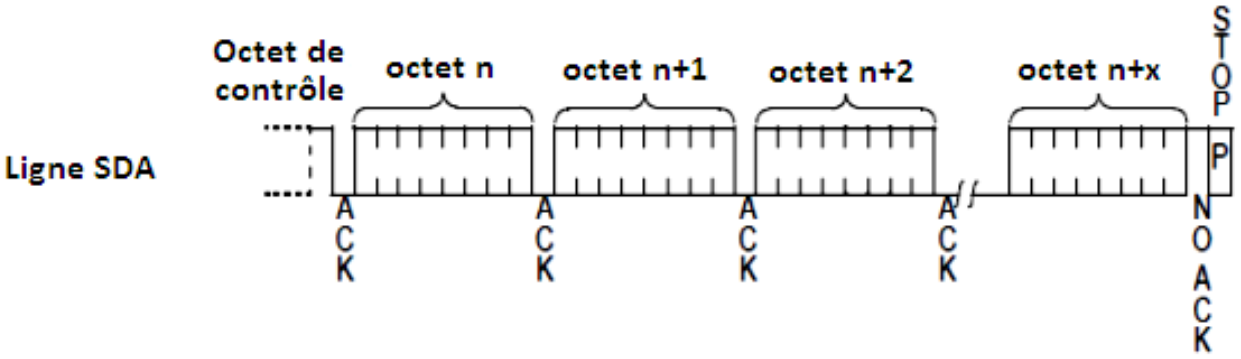


Fig.3. 34: Lecture séquentielle [29].

3.7. Programmation et simulation du protocole I²C

Le protocole I²C assure la communication entre le lecteur/terminal et la carte à puce à mémoire I²C ou bien entre le microcontrôleur de la carte et sa mémoire I²C.

Les cartes à puce utilisées pour la programmation et la simulation du protocole I²C sont : la 24LC64 et la wafer 2. La Wafer 2 est dotée d'un microcontrôleur PIC 16F84a et d'une mémoire 24LC64.

Nous avons programmé le protocole I²C en langage assembleur sur le microcontrôleur 16F84a ; sous l'environnement MPLAB de Microchip.

3.7.1. Schéma de simulation

Deux schémas de simulation ont été réalisés sous le logiciel Proteus présentés dans les figures 4.35 et 4.36.

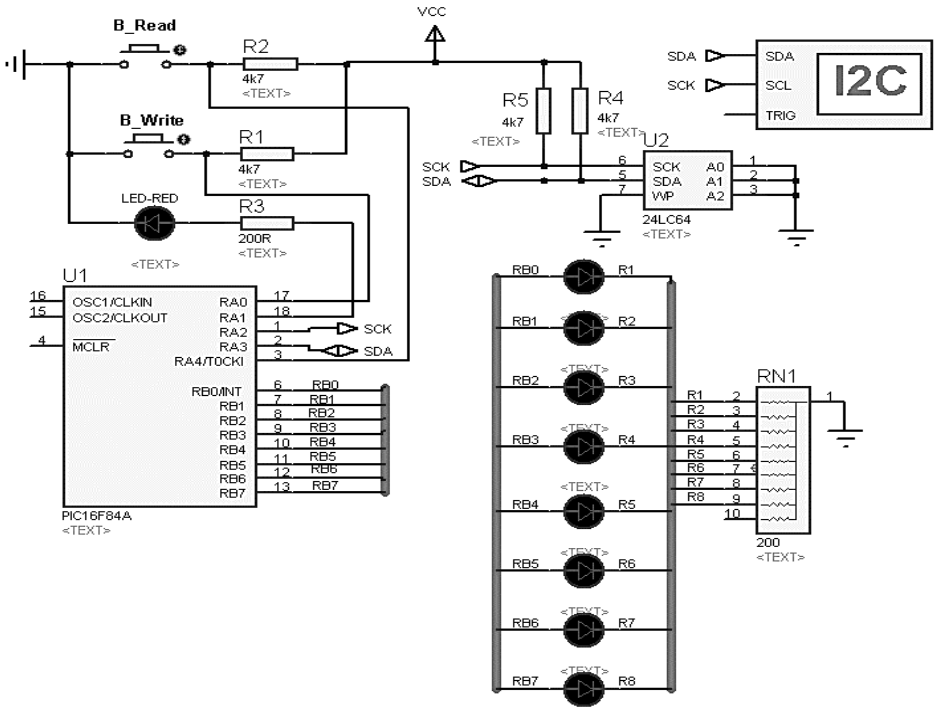


Fig.3. 35: Simulation du fonctionnement du protocole I²C entre le microcontrôleur et l'EEPROM de la carte à puce Wafer 2.

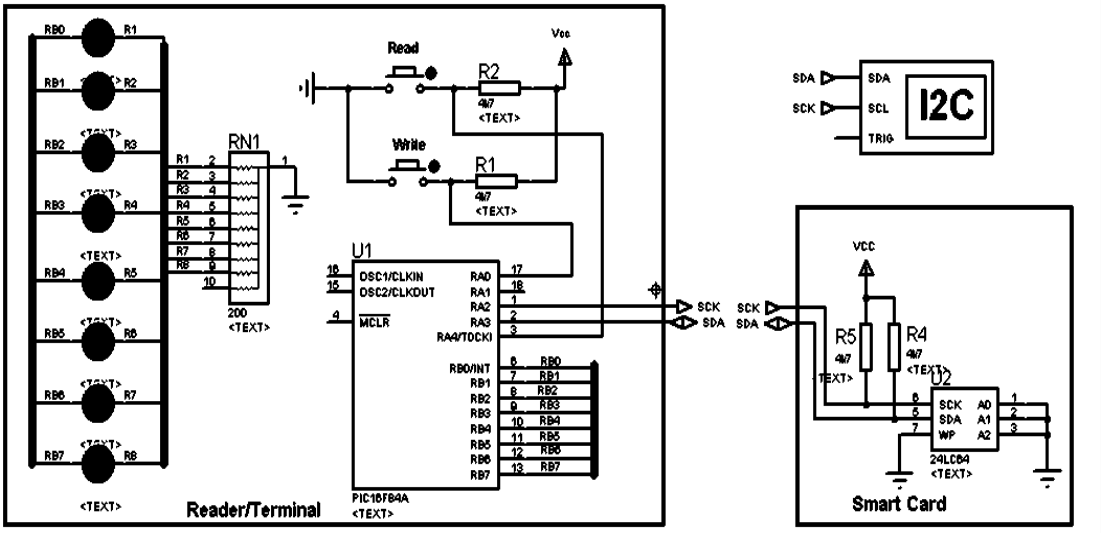


Fig.3. 36: Communication entre une carte à puce à mémoire I²C avec un lecteur/terminal.

Dans le circuit de la figure 3.35 nous avons simulé ce protocole sur une carte à puce Wafer 2 (entre son PIC et sa mémoire EEPROM). Le schéma de simulation réalisé à la figure 3.36 permet de simuler ce protocole entre une carte à puce à mémoire (la 24LC64) et un terminal représenté par le microcontrôleur PIC 16F84a.

Proteus nous donne la possibilité de visualisé le protocole I²C à l'aide d'un simulateur I²C debug. Comme le montre la figure 3.37, on peut visualiser tous les paquets échangé entre l'EEPROM et le PIC.

Pour la simulation du protocole I²C entre le microcontrôleur de la carte et sa mémoire EEPROM, nous avons proposé d'écrire des caractères dans l'EEPROM envoyé depuis le microcontrôleur et de les lire par la suite par ce dernier et les affiché sur les LEDs reliées au PORTB afin de vérifier que ce sont les bons caractères.

- Le I²C debug servira à affiché les échanges entre microcontrôleur et EEPROM ;
- Les LEDs serviront à afficher le caractère lus par le PIC ;
- Les lignes RA1, et RA2 serviront à établir la liaison I2C avec l'EEPROM.

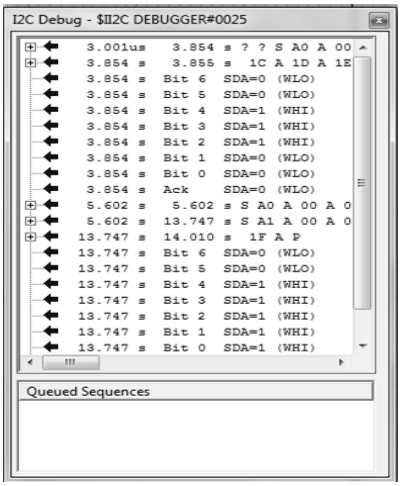


Fig.3. 37: I²C debug.

3.7.2. Ecriture dans l'EEPROM

3.7.2.1. Organigrammes

- **Algorithme global**

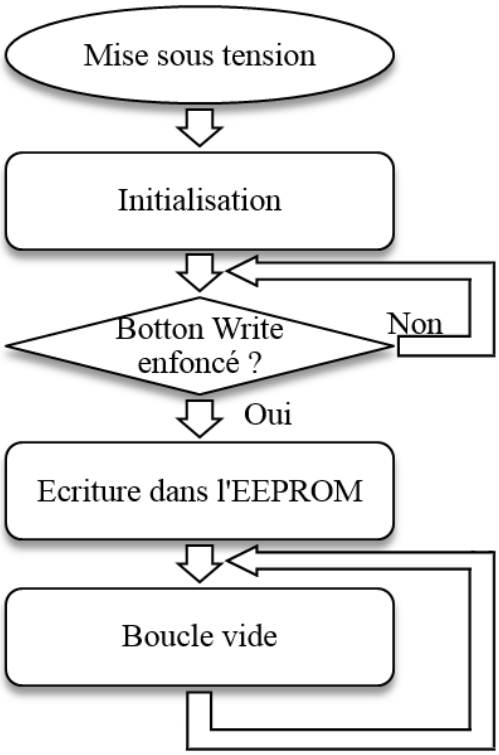


Fig.3. 38: Organigramme d'écriture dans l'EEPROM.

- **L'algorithme d'initialisation**

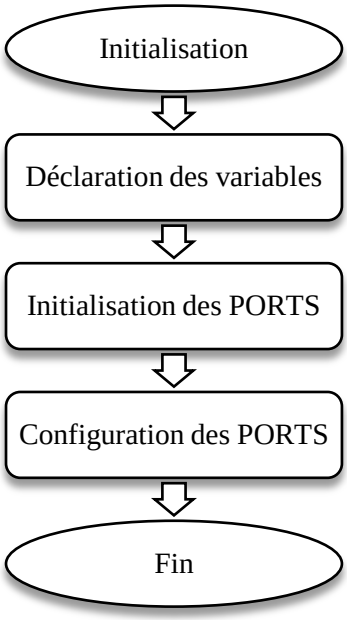


Fig.3. 39: Organigramme d'initialisation du PIC.

- **Algorithme d'écriture d'un octet**

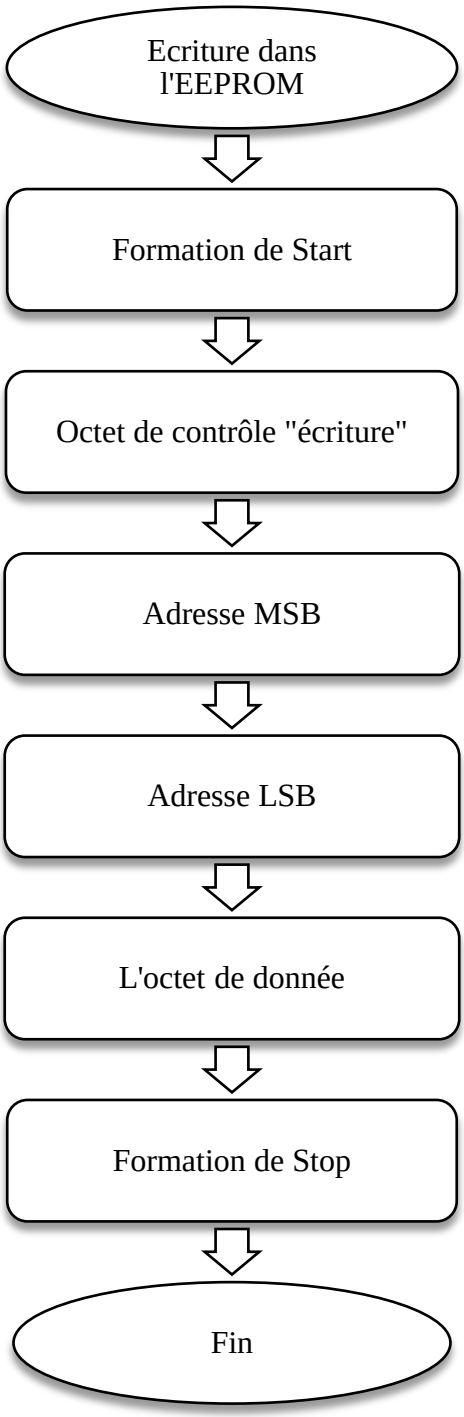


Fig.3. 40: Organigramme d'écriture d'un octet dans l'EEPROM.

- **Algorithme d'écriture d'une page de 32 octets**

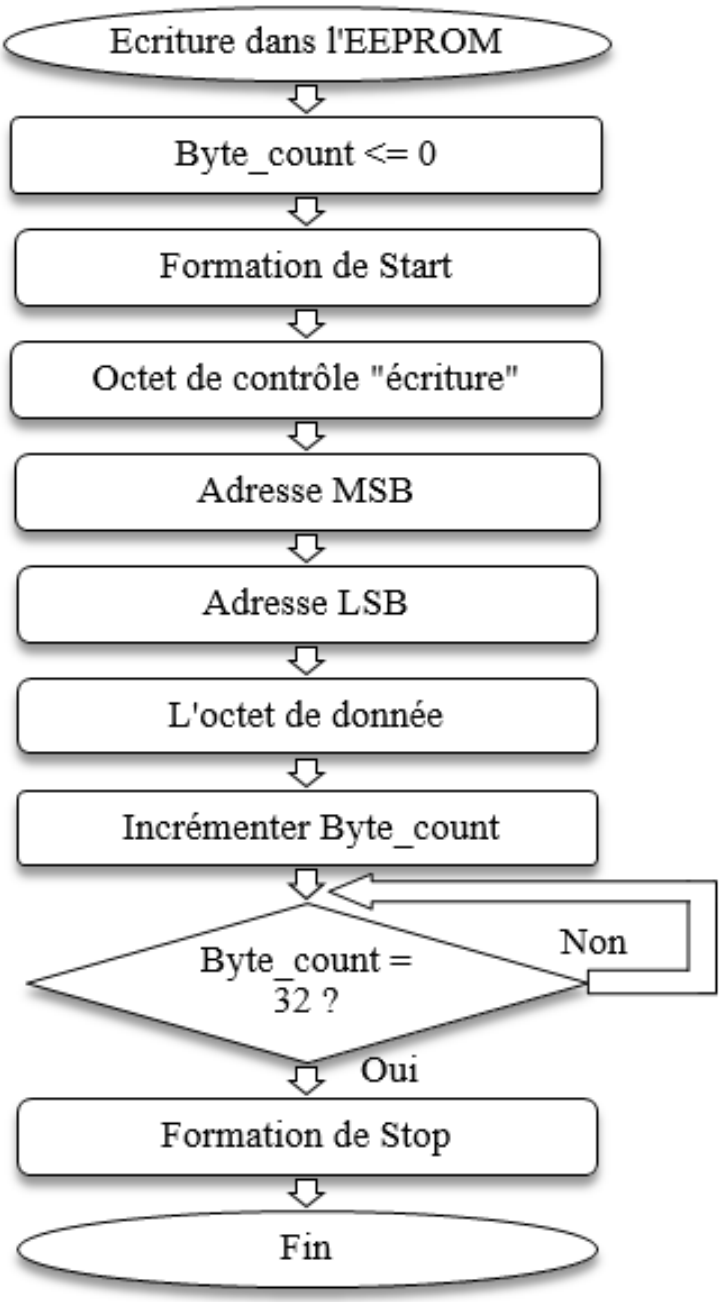


Fig.3. 41: Organigramme d'écriture d'une page (32 octets) dans l'EEPROM.

3.7.2.2. Simulations

Pour l'opération d'écriture, nous utilisons le bouton Write connecté au microcontrôleur PIC pour envoyer des données à partir du maître, qui est le PIC, à l'esclave correspondant à l'EEPROM I²C., selon le chronogramme représenté sur la figure 3.42.

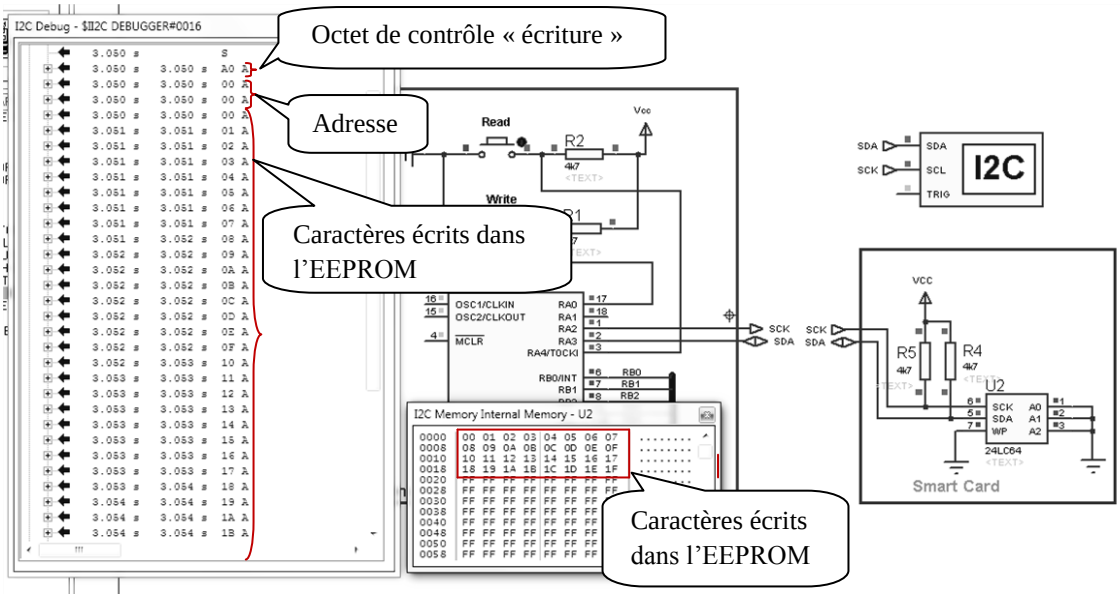


Fig.3. 42: Simulation de l'écriture de 32 caractères dans l'EEPROM I²C.

3.7.3. Lecture de l'EEPROM

3.7.3.1. Organigrammes

- **Algorithme global**

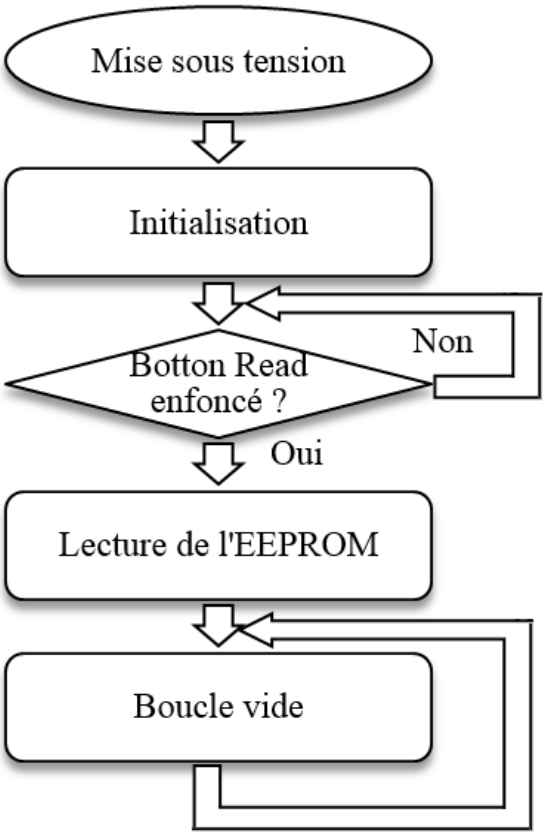


Fig.3. 43: Organigramme de lecture de l'EEPROM.

- **Algorithme d'initialisation**

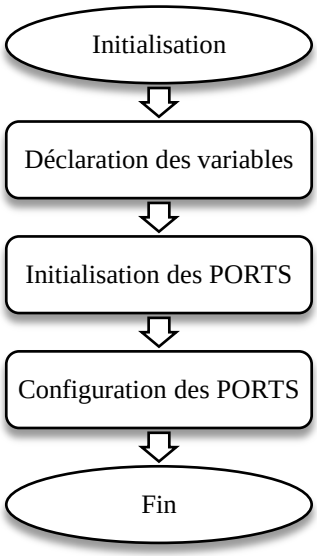


Fig.3. 44: Organigramme d'initialisation du PIC.

- **Algorithme de lecture d'un octet**

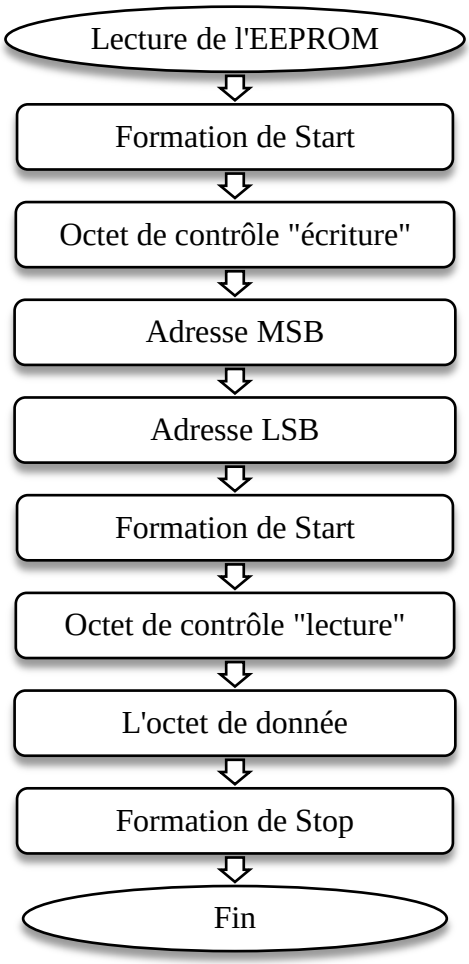


Fig.3. 45: Organigramme de lecture d'un octet de l'EEPROM.

- **Algorithme de lecture d'une page de 32 octets**

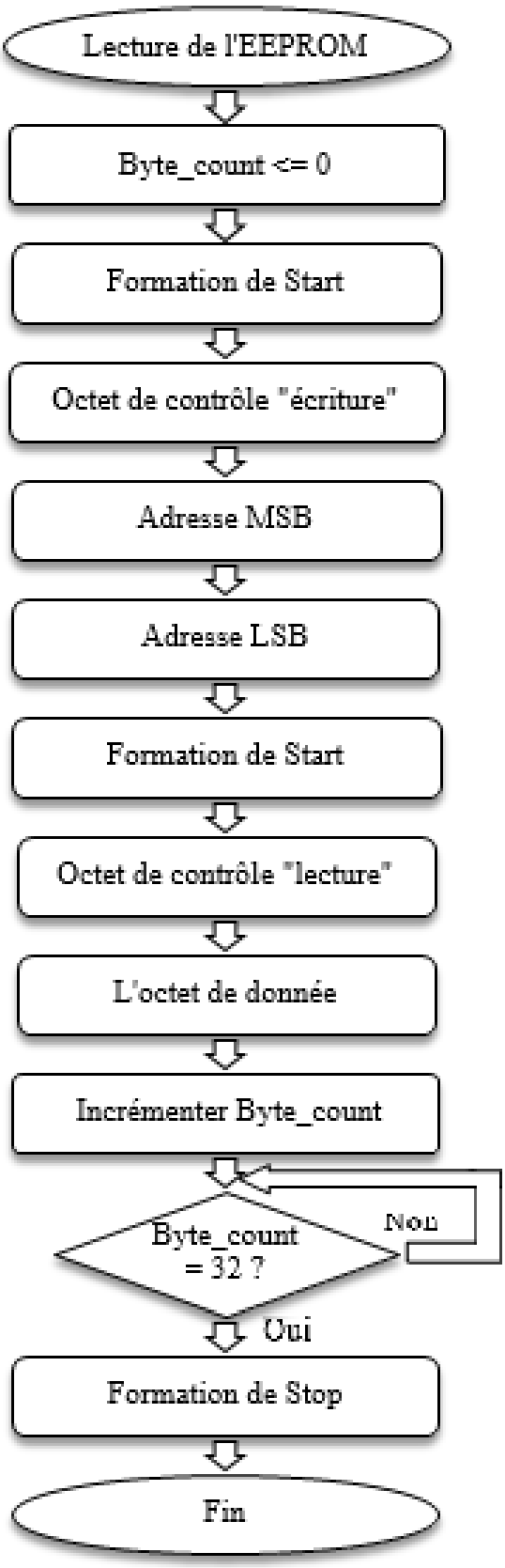


Fig.3. 46: Organigramme de lecture d'une page (32 octets) de l'EEPROM.

3.7.3.2. Simulations

En utilisant le bouton Read, le PIC lit les données enregistrées dans l'EEPROM, puis les affiche sur les 8 LEDs connectés au PORTB. Cela, afin de vérifier le bon déroulement de l'opération de lecture (Voir la figure 3.47).

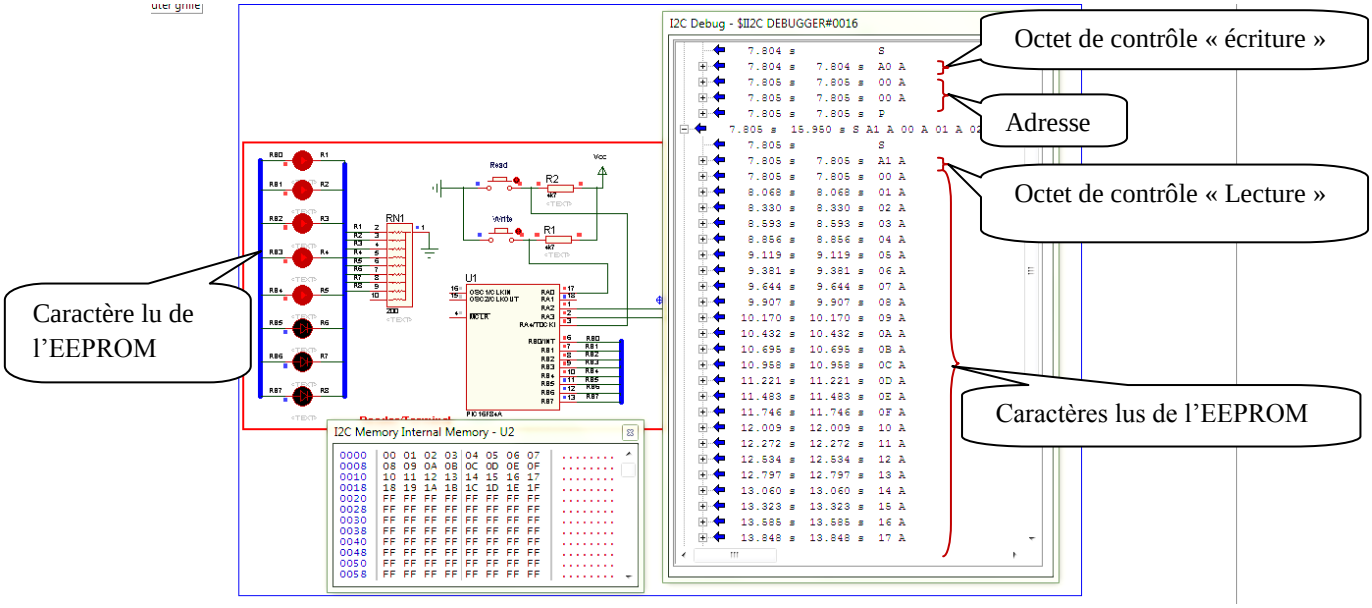


Fig.3. 47: Simulation de la lecture de 32 Caractères à partir de l'EEPROM I²C.

3.8. Protocole de transmission T=0 [3, 5, 6, 15]

Le protocole de transmission T=0 est le plus ancien et le plus utilisé par les cartes à puce à microcontrôleur avec contacts. Ce protocole est relativement optimisé pour déplacer les commandes et réponses entre la carte et le lecteur. Les octets sont transmis selon le protocole RS232.

3.8.1. Commandes APDUs (Application Protocol Data Unit)

La communication entre la carte et le lecteur se produit selon différentes transitions d'état. Le canal de communication se fait dans un seul sens ; une fois que le lecteur envoie une commande à la carte à puce (voir le tableau 3.8), il est bloqué jusqu'à ce qu'une réponse soit reçue.

Parties obligatoires				**Parties optionnelles**		
CLA	INS	P1	P2	Lc	donnée	Le
1 Octet	1 Octet	1 Octet	1 Octet	1 Octet	Lc Octets	1 Octet

Tab.3. 8: Format des commandes APDUs.

CLA : Indique la classe de la commande.
INS : Octet de l'instruction.
P1, P2 : Paramètres de l'instruction.
Lc : Indique le nombre d'octets de la donnée.
Data : La donnée à envoyer.
Le : Indique la langueur possible de la donnée à recevoir depuis la carte à puce en réponse.

3.8.2. Réponses APDUs (Application Protocol Data Unit)

La carte peut effectuer une séquence d'opérations, comme dicté par la commande du lecteur, et peut envoyer une réponse vers le lecteur et fournit une indication de l'état de l'exécution de la commande , (voir le tableau 3.9).

Parties optionnelle	**Parties obligatoires**	
Donnée	SW1	SW2
Le Octets	1 Octet	1 Octet

Tab.3. 9: Format des réponses APDUs.

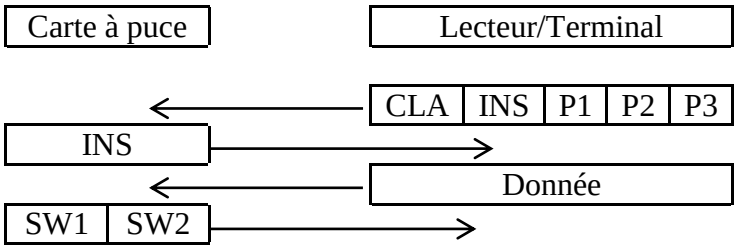
Donnée : Données reçues de la carte.
SW1, SW2 : les octets d'état.

3.8.3. Protocole TPDU (Transport Protocol Data Unit)

Les données échangées entre le lecteur et la carte avec ces commandes et réponses sont référenciées comme Transmission Protocol Data Unit (TPDU). On trouve principalement deux formats de TPDU.

Le premier consiste à envoyer des données depuis le lecteur vers la carte à puce. Comme l'explique la figure 3.48 :

- Le lecteur commence par envoyer la classe, l'instruction et ses paramètres.
- La carte répond en envoyant le code de l'instruction pour indiquer qu'elle a bien reçu la commande.
- Le lecteur compare le code reçu de la carte avec celui de la commande qu'il a envoyé, s'il est le même, le lecteur envoi l'autre moitié de la commande qui correspond à la donnée d'une taille P3.
- Après la réception, la carte répond en envoyant SW1 et SW2 afin d'indiquer l'état du déroulement de la commande.



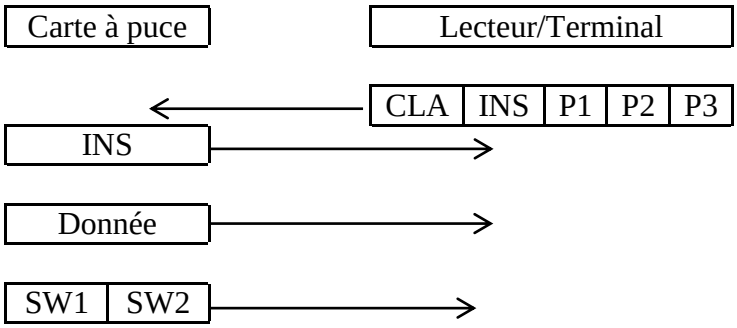
Le paramètre P3 peut être Le ou Lc.

Fig.3. 48: Donnée envoyé depuis le lecteur vers la carte à puce.

Le deuxième consiste à envoyer des données depuis la carte à puce vers le lecteur. Comme l'explique la figure 3.49 :

- Le lecteur commence par envoyer la classe, l'instruction et ses paramètres.
- La carte répond en envoyant le code de l'instruction pour indiquer qu'elle a bien reçu la commande.

- Le lecteur compare le code reçu de la carte avec celui de la commande qu'il a envoyé, s'il est le même, le lecteur attend la réception d'une donnée d'une taille P3.
- Après la soumission de la donnée, la carte envoi SW1 et SW2 afin d'indiquer l'état du déroulement de la commande.



Le paramètre P3 peut être Le ou Lc.

Fig.3. 49: Donnée envoyée depuis la carte à puce vers le lecteur.

3.9. Programmation et simulation du protocole T=0

Les octets des commandes et réponses APDUs sont échangés selon le protocole RS232. Nous avons déjà programmé ce protocole, en langage assembleur, sous l'environnement MPLAB de Microchip pour la même carte à puce Wafer 2.

Nous allons utiliser les codes d'envoi et de réception RS232 afin de programmer les différentes commandes APDUs du protocole T=0. Le logiciel Proteus est toujours utilisé pour la simulation.

3.9.1. Schéma de simulation

Le circuit de simulation est divisé en deux parties : la carte à puce et le lecteur / terminal.

La figure 3.50 montre le circuit de simulation entre une carte à puce et un lecteur. Le lecteur est simulé par un port série DB9 qui assure la communication avec un ordinateur. Le terminal virtuel envoie et reçoit des données vers / depuis la carte à puce ; il fonctionne comme une application dans l'ordinateur.

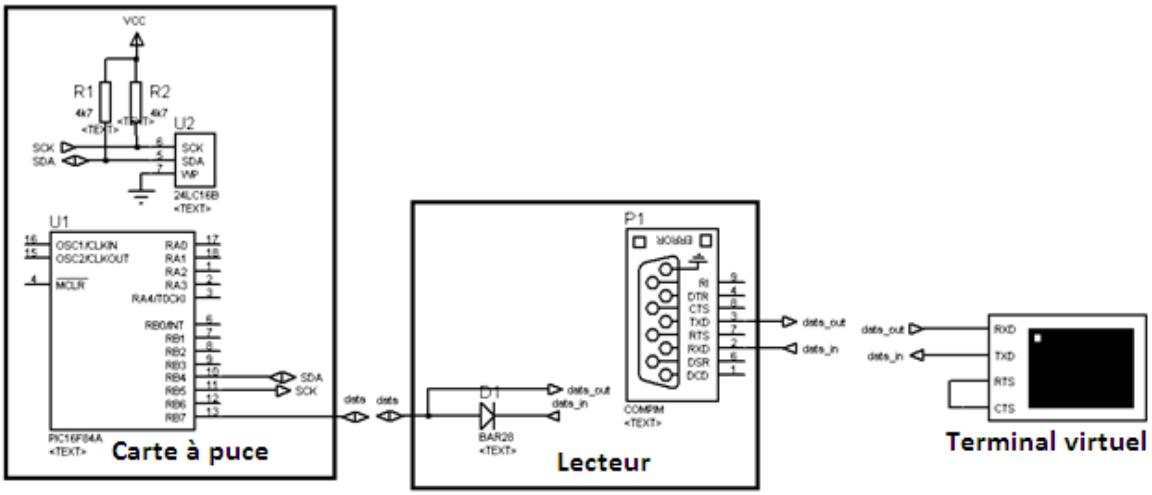


Fig.3. 50: Communication entre une carte à puce à microcontrôleur et un lecteur.

Pour simuler la communication avec un terminal, un microcontrôleur PIC 16F84a a été utilisé. Ce microcontrôleur est responsable de l'accès aux fichiers et de la communication avec la carte à puce. Le terminal virtuel est utilisé pour afficher les données échangées entre la carte et le terminal comme le montre la figure 3.51.

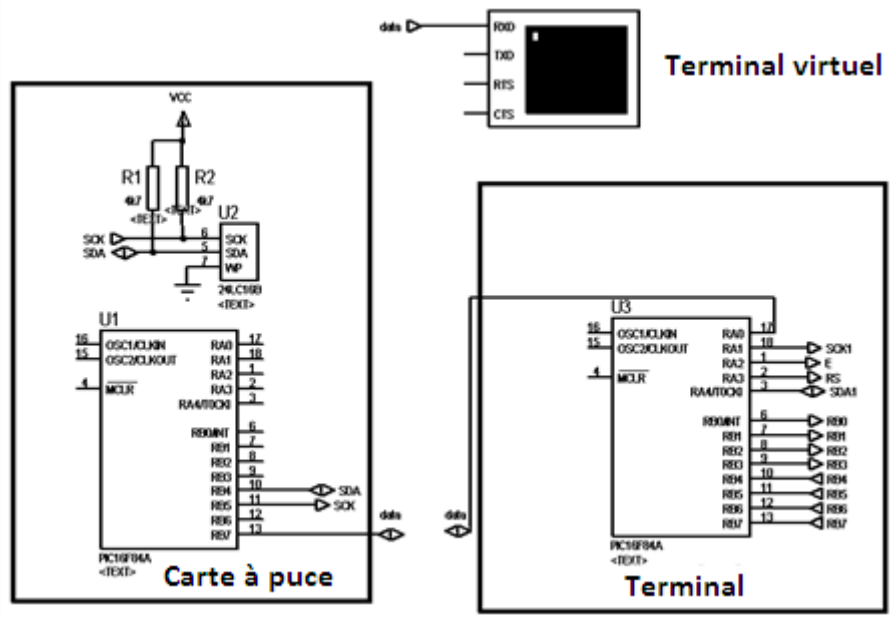


Fig.3. 51: Communication entre une carte à puce à microcontrôleur et un terminal.

3.9.2. Commandes utilisées [2]

Une fois la carte à puce est insérée, elle envoi l'ATR (Answer To Reset) pour le lecteur / terminal. Cette carte à puce utilise trois commandes APDUs :

- **La commande VERIFY** : utilise 6 caractères pour la vérification d'un mot de passe. Le format de la commande VERIFY est présenté dans le tableau 3.10. Le format de la réponse APDU est présenté dans le tableau 3.11.

Commande APDU					
CLA	INS	P1	P2	Lc	Données
00	20	00	00	06	xx xx xx xx xx xx

xx : 1 octet du mot de passe.

Tab.3. 10: Format APDU de la commande VERIFY.

	Réponse APDU	
Etat	SW1	SW2
Succès	90	00
Erreur	98	40

Tab.3. 11: Réponse APDU de la commande VERIFY.

- **La commande SELECT :** utilisée pour sélectionner une adresse d'un octet. Le tableau 3.12 donne le format de la commande APDU et le tableau 3.13 donne le format de la réponse APDU.

Commande APDU					
CLA	INS	P1	P2	Lc	Donnée
00	A4	00	00	01	xx

xx : 1 octet d'adresse.

Tab.3. 12: Format APDU de la commande SELECT.

	Réponse APDU	
Etat	SW1	SW2
Succès	90	00
Erreur	6A	82

Tab.3. 13: Réponse APDU de la commande SELECT.

- **La commande UPDATE RECORD :** Permet d'écrire 1 octet dans la carte à puce à l'adresse déjà sélectionnée. Il suffit de changer la valeur de Lc pour envoyer plus de données à la carte. Le tableau 3.14 donne le format de la commande APDU et le tableau 3.15 donne le format de la réponse APDU.

Commande APDU					
CLA	INS	P1	P2	Lc	donnée
00	DC	00	00	01	xx

xx : 1 octet de donnée.

Tab.3. 14: Format APDU de la commande UPDATE RECORD.

	Réponse APDU	
Etat	SW1	SW2
Succès	90	00
Erreur	92	40

Tab.3. 15: Réponse APDU de la commande UPDATE RECORD.

3.9.3. Simulations

Lorsque la carte à puce est insérée dans le lecteur, elle commence par envoyer l'ATR. Comme le montre la figure 3.52, l'ATR est 3B 83 00 00 00 00 03h. Après l'ATR, le lecteur commence sa communication avec la carte en utilisant les commandes APDUs.

- **La commande VERIFY :** Elle est envoyée avec les octets : 00 20 00 00 06_h, indiquant à la carte que le mot de passe qu'elle va recevoir par la suite est sur 6 caractères. La carte répond en envoyant l'octet INS qu'elle a reçu comme accusés de réception qui est : 20_h, (voir figure 3.52). Le test est fait avec deux mots de passe : le premier est incorrect et le deuxième est correct.

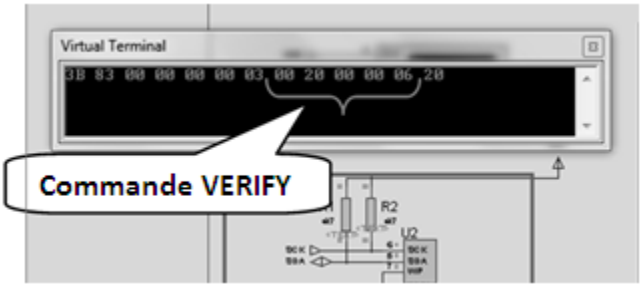


Fig.3. 52: Simulation de la commande VERIFY.

Pour un mot de passe incorrecte, nous avons utilisé les caractères : 1 2 3 4 5 6 correspondant au code ASCII : 31 32 33 34 35 36_h. Après avoir comparé le mot de passe reçu et celui enregistré au niveau de la carte à puce, cette dernière répond en envoyant SW1=98_h SW2=40_h indiquant que le mot de passe n'est pas correct. La figure 3.53 donne l'échange d'octets de cette commande au niveau TPDU, et la figure 3.54 donne le résultat de simulation.

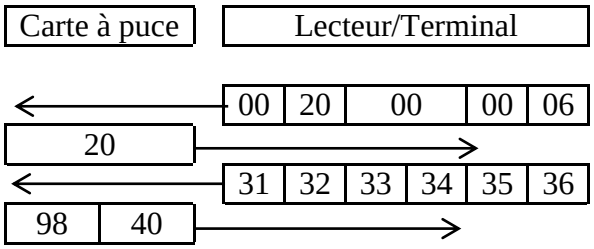


Fig.3. 53: protocole TPDU de la commande VERIFY pour un mot de passe incorrecte.

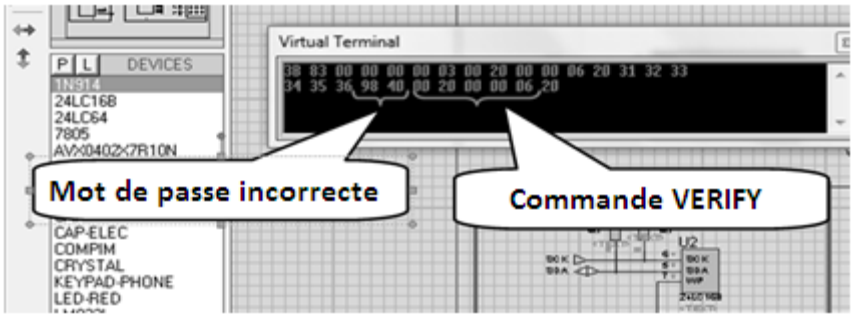


Fig.3. 54: Commande VERIFY avec un mot de passe incorrecte.

Comme le mot de passe est incorrect, le lecteur envoi à nouveau la commande VERIFY à la carte à puce. Mais cette fois le mot de passe à vérifier est correcte, nous utilisons les caractères : 0 0 0 0 0 0 0 0 qui correspondent aux codes ASCII : 30 30 30 30 30 30_h. L'échange de donnée est bien expliqué dans la figure 3.55. La carte répond à la fin par SW1=90_h SW2=00_h indiquant que le mot de passe est correct comme le montre la simulation de la figure 3.56.

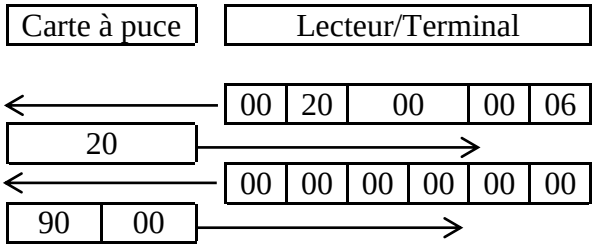


Fig.3. 55: protocole TPDU de la commande VERIFY pour un mot de passe correcte.

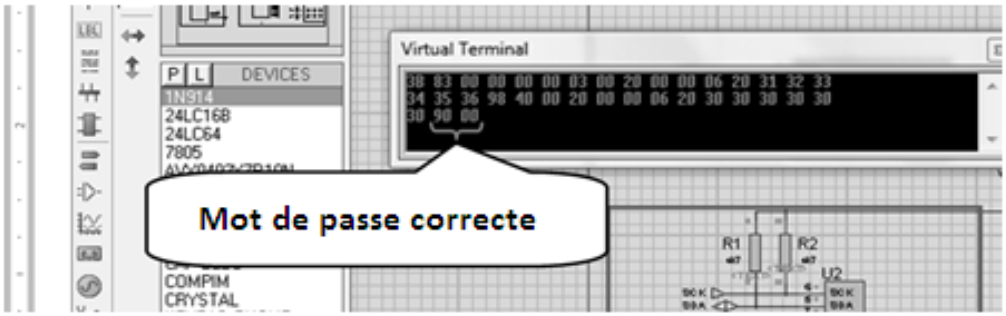


Fig.3. 56: Commande VERIFY avec un mot de passe correcte.

- **SELECT Command :** Elle est utilisée pour sélectionner une adresse. Elle est envoyée avec les octets : 00 A4 00 00 01_h, indiquant à la carte qu'une adresse de la taille d'un octet va suivre. La carte répond en envoyant l'octet INS qu'elle a reçu comme accusés de réception qui est : A4h, (voir figure 3.58). Comme le INS de la commande envoyé et celui reçu de la carte sont identique, le lecteur envoi l'adresse 00h à la carte à puce. Une fois l'adresse bien sélectionnée au niveau de la carte, cette dernière répond en envoyant $SW1=90_h$ $SW2=00_h$. La figure 3.57 donne l'échange d'octets de cette commande au niveau TPDU.

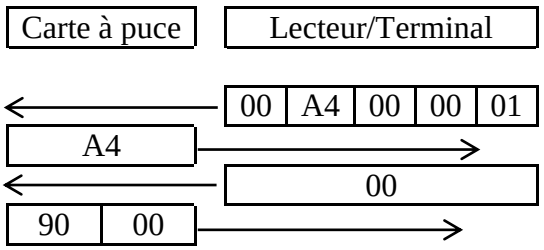


Fig.3. 57: Protocole TPDU de la commande SELECT.

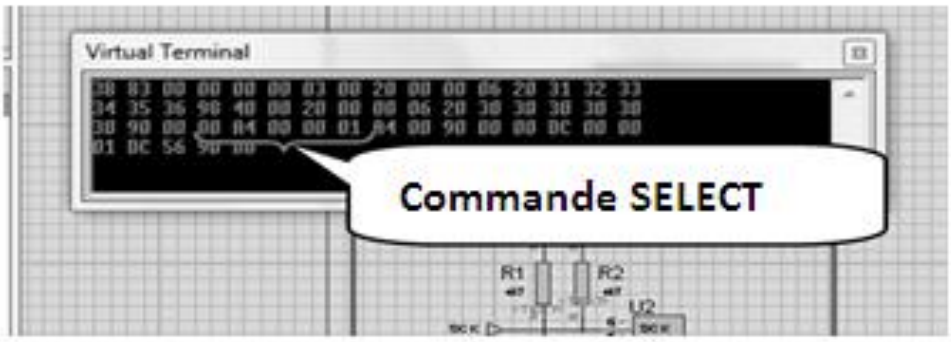


Fig.3. 58: Simulation de la commande SELECT.

- **La commande UPDATE RECORD** : Elle est utilisée pour écrire dans une adresse déjà sélectionnée. Elle est envoyée avec les octets : 00 DC 00 00 01_h, indiquant à la carte qu'une donnée de la taille d'un octet va suivre. La carte répond en envoyant l'octet INS qu'elle a reçu comme accusés de réception qui est : DC_h, (voir la figure 3.59). Comme le INS de la commande envoyée et celui reçu de la carte sont identiques, le lecteur envoi le caractère A correspondant au code ASCII 56h à la carte à puce. Une fois l'opération d'écriture est bien déroulée au niveau de la carte, cette dernière répond en envoyant $SW1=90_h$ $SW2=00_h$. La figure 3.60 donne l'échange d'octets de cette commande au niveau TPDU.

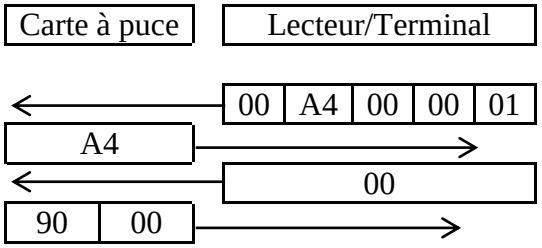


Fig.3. 59: Protocole TPDU de la commande UPDATE RECORD.

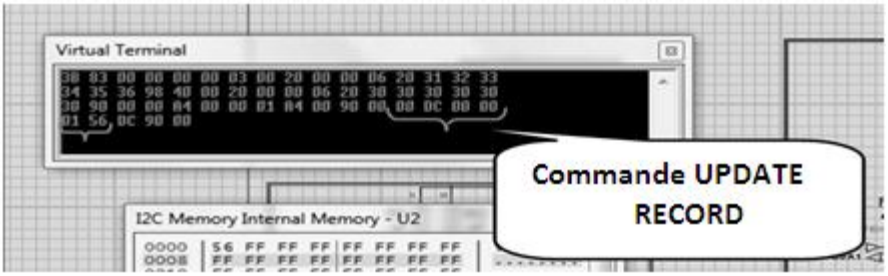


Fig.3. 60: Simulation de la commande UPDATE RECORD.

3.10. Conclusion

Ce chapitre permet de mieux comprendre comment accéder à n'importe qu'elle carte à puce depuis n'importe qu'elle lecteur ou terminal. Ce qui fait que ce travail fait la base de n'importe quelle application à base de carte à puce.

Nous avons étudié les protocoles de communication I2C et RS232, les plus connus, des cartes à puce ainsi que le protocole de transmission T=0 le plus utilisé jusqu'à nos jours.

Les contributions de ce travail sont :

- Proposition de techniques de simulation pour mieux comprendre ces échanges et détecter les erreurs de programmation plus facilement.
- et le plus important est l'optimisation de l'espace mémoire en les programmant en langage assembleur.

Chapitre 4

Conception et réalisation d'une carte à puce destinée aux étudiants

Certains des travaux présentés dans ce chapitre ont fait l'objet d'un article présenté lors de la conférence [Telecom & 9ème JFMMA 2015].

4.1. Introduction

Parmi les applications les plus récentes de la carte à puce, on trouve la carte étudiant qui remplacent la carte d'étudiant en papier.

Nous avons abordé la problématique des coûts très élevés des cartes à puce destinées aux étudiants. Notre objectif est d'améliorer cette application tout en optimisant les ressources utilisées (matériel et espace mémoire). Cela, afin de réduire les coûts tout en assurant le bon fonctionnement.

Dans ce chapitre, Nous présentons la conception et la réalisation d'une carte à puce, destinée aux étudiants, qui servira non seulement à faciliter le travail de l'administration mais aussi à sauvegarder toutes les informations de l'étudiant durant ses études universitaires. La carte à puce sélectionné est la Wafer 2 et le lecteur de cartes dénommé Phoenix. La simulation est faite en utilisant le logiciel Proteus. Le microcontrôleur de la carte à puce est programmé en langage assembleur sous le logiciel MPLAB. L'interface graphique est réalisée en Visual Basic 6.

4.2. Cartes à puce étudiant existantes

Les premières cartes qui ont remplacées les cartes à étudiant en papier étaient les cartes à bandes magnétiques. Avec l'apparition des cartes à puce, qui ont plus d'avantage en espace mémoire, sécurité, fiabilité et fonctionnalité, les cartes d'étudiants sont conçues à base de cartes à puce.

Les cartes à bande magnétique ont un très faible stockage contrairement aux cartes à puce qui peuvent sauvegarder des centaines de fois plus d'informations que la carte à bande magnétique. En plus, les cartes à puce peuvent avoir plus d'utilisation et peut être plus polyvalent que les cartes magnétiques.

La carte à puce étudiants existe sous différentes fonctionnalités : carte d'identité d'étudiant, transport, carte de paiement et comme une carte d'accès ou de connexion à des ordinateurs. Parfois, des cartes à puce polyvalentes offrent plusieurs services en une seule carte.

La majorité des travaux (pour ne pas généraliser) qui ont été fait pour la conception de cartes à puce destinées aux étudiants ont utilisés des cartes à puce avec de grandes performances pour des petites applications : Cartes d'accès à un endroit dans le campus comme (l'université, bibliothèque, cafétéria, salle de lecture ….etc). La plupart de ces cartes sont dotées avec une seule application à la fois. On peut citer les travaux : [38-41]

4.3. Cahier de charge

Afin que chaque étudiant possèdera sa propre carte à puce étudiant, la faculté des sciences de l'université M'hamed Bougara de Boumerdes à proposer la réalisation de ce projet tout en respectant le cahier des charges suivant :

- Cette carte servira à garder les informations de son titulaire ainsi que ses relevais de notes durant tout son parcours universitaire ;
- L'accès à la carte se fera à l'aide d'un mot de passe, avec un nombre d'essai limité, tel que l'administrateur peux lire et apporter des modifications à la carte, par contre l'étudiant ne peut intervenir dans aucune modification ;
- Et le plus important, La carte doit être la moins chère possible ;

Au début de chaque année universitaire, et à la fin de chaque semestre l'étudiant doit se présenter au niveau de la scolarité de la faculté afin de réactiver sa carte et obtenir ses notes. Cela facilitera énormément le travail de l'administration.

4.4. Carte à puce utilisée

Il existe différents types de carte à puce et le choix d'une carte doit être fait très soigneusement. Les cartes Java ou Basic coutent trop chère et pour cela nous avons choisi une carte vierge à base d'un microcontrôleur PIC (ca donnera les même performances de fonctionnement et de sécurité) car ce sont les moins chères du marché (Wafer 1, 2 ou 3). Et comme la taille de notre programme de gestion de la carte n'est pas si grande, un PIC avec des capacités moyennes suffit, comme le 16F84 (Wafer 1 ou 2). En plus la taille de données à sauvegarder dans la carte est de 7 K octet, ce qui implique la sélection de la carte Wafer2.

4.4.1. Carte à puce Wafer 2

Carte à puce Wafer 2, Appelée également Gold 64 en raison de la couleur dorée de certaines versions (voir figure 4.1).

Fig.4. 1: Carte à puce Wafer 2.

4.4.2. Schéma de la Wafer 2

Comme le montre la figure 4.2, cette carte contienne un microcontrôleur PIC 16F84a de Microchip ; et une mémoire EEPROM 24LC64 d'une capacité de 8 K octets. C'est une carte vierge de tout programme ou données.

Il est possibilité d'accéder aux lignes SDA et SCL de la mémoire EEPROM via les contacts C4 et C8. Cette possibilité optionnelle, d'où la présence des interrupteurs S1 et S2 permet de programmer directement le contenu de l'EEPROM sans passer "au travers" du microcontrôleur.

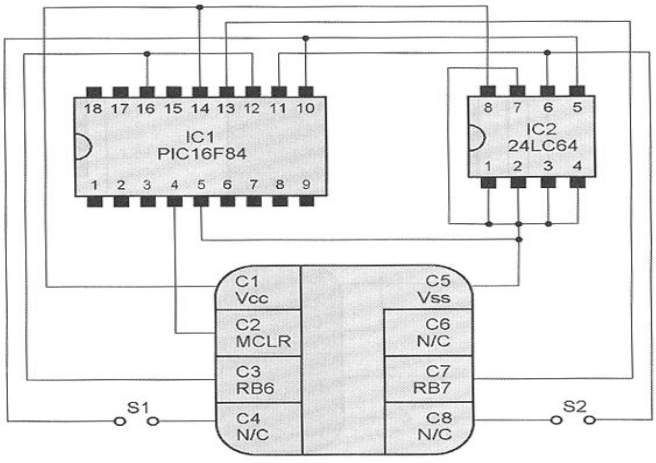


Fig.4. 2: Composition de la carte à puce Wafer 2.

Ces deux liaisons n'existent que sur les cartes Gold que l'on réalise soi-même, elles n'existent pas dans les vraies cartes Wafer.

Pour programmer depuis l'extérieur la mémoire de ces cartes, il faut faire appel à un "loader" préalablement programmé dans le PIC, et qui rend en quelque sorte celui-ci «transparent » pendant la phase de programmation de l'EEPROM.

4.4.3. Réalisation de la Wafer 2

Nous avons réalisé le montage dont le dessin du circuit imprimé est fait sous AEGLE.

4.4.3.1. Liste des composants

IC1 : PIC 16F84a

IC2 : 24LC64

S1 et S2 : interrupteurs

4.4.3.2. Circuit imprimé

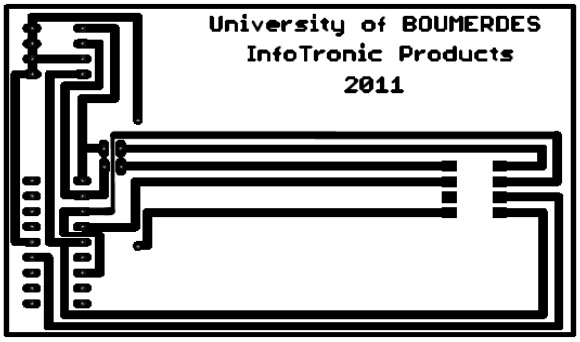


Fig.4. 3: Dessin du circuit imprimé de la carte Wafer2 (coté cuivre).

4.4.3.3. Implantation des composants

Le plan d'implantation est indiqué sur la figure ci-dessous :

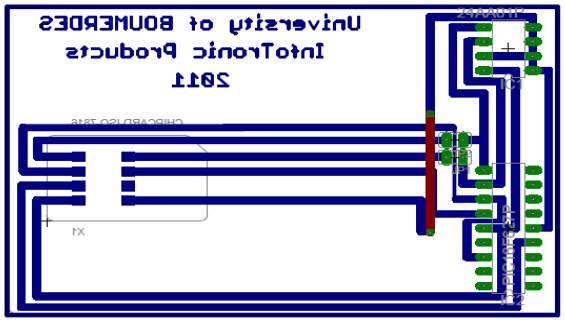


Fig.4. 4: Schéma d'implantation des composants (coté composants).

Après avoir placé tous les composants, voici notre carte :

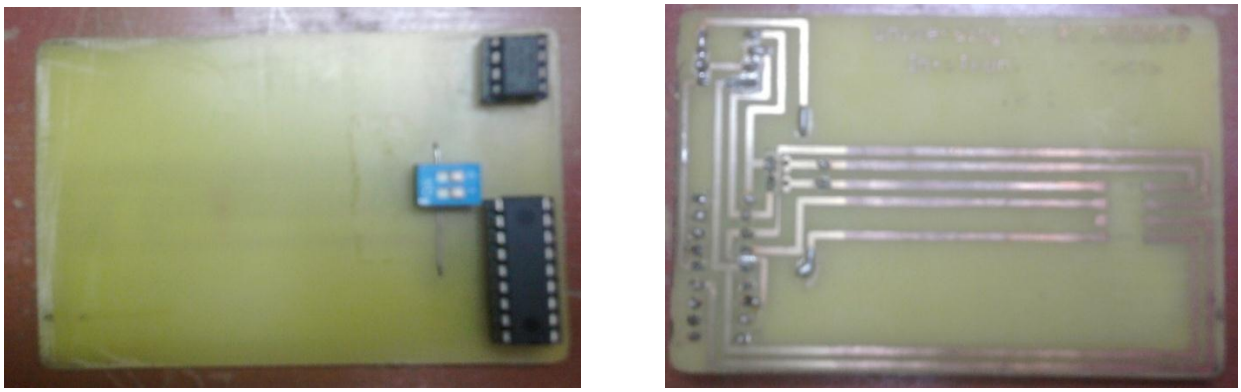


Fig.4. 5: La Wafer2 après implantation des composants.

4.5. Lecteur de carte à puce utilisé

Il existe différent types de lecteurs/programmateurs de cartes à puce, et parmi ceux qui convient comme lecteur à notre carte on trouve le phœnix, après avoir simulé son fonctionnement, nous l'avons réalisé sur circuit imprimé.

4.5.1. Lecteur PHOENIX ou SMART MOUSE

Avec l'apparition des premières cartes Gold, deux schémas de lecteur ont été développés, baptisés :

- Phoenix ;
- Et Smart Mouse.

Ces schémas étaient très proche l'un de l'autre, et pour que le lecteur soit vraiment polyvalent, on le rend compatible Phoenix et Smart Mouse.

Fig.4. 6: Le lecteur Phoenix.

4.5.2. Schéma du lecteur

- **Partie interface**

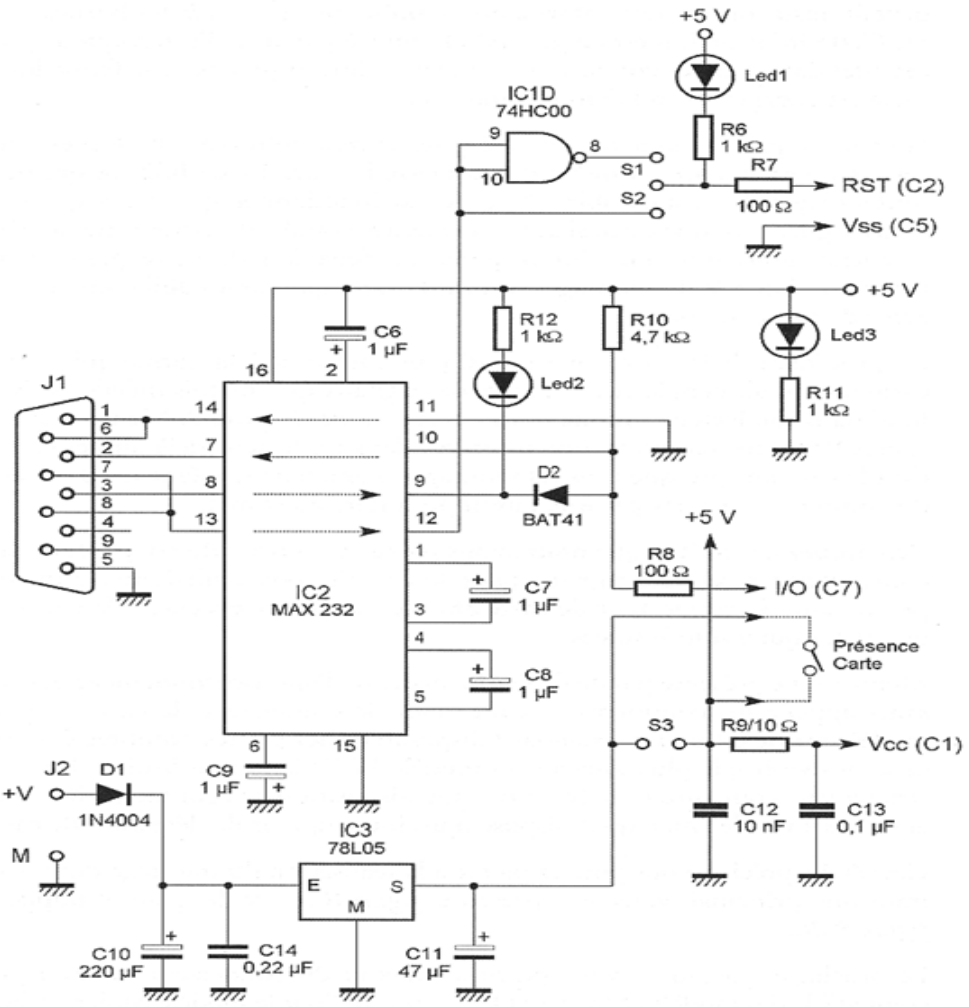


Fig.4. 7: Schéma de la partie interface [2].

- **Horloge**

L'horloge utilisée par la carte est générée par un oscillateur à quartz réalisé autour :

- d'IC1a pour fonctionner à 3,579 MHz positionnée par l'interrupteur S5 ;
- ou d'IC1b pour fonctionner à 6 MHz positionnée par l'interrupteur S4.

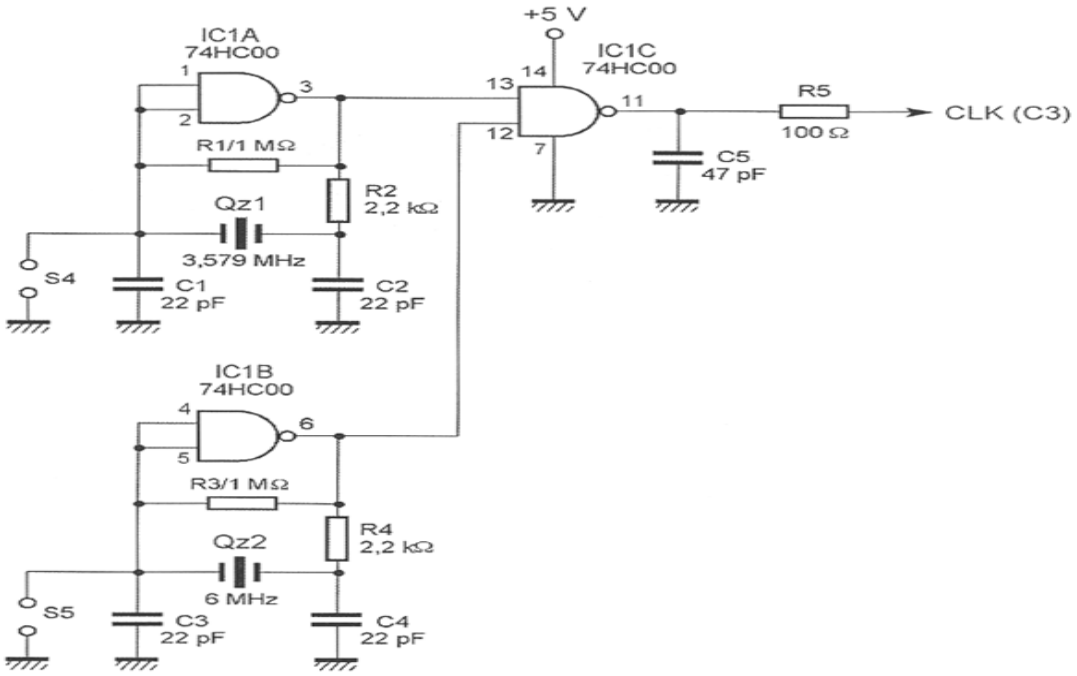


Fig.4. 8: Schéma de la partie horloge [2].

4.5.3. Simulation du fonctionnement

Afin de vérifier le fonctionnement du lecteur, nous avons réalisé son schéma sous PROTEUS comme suit :

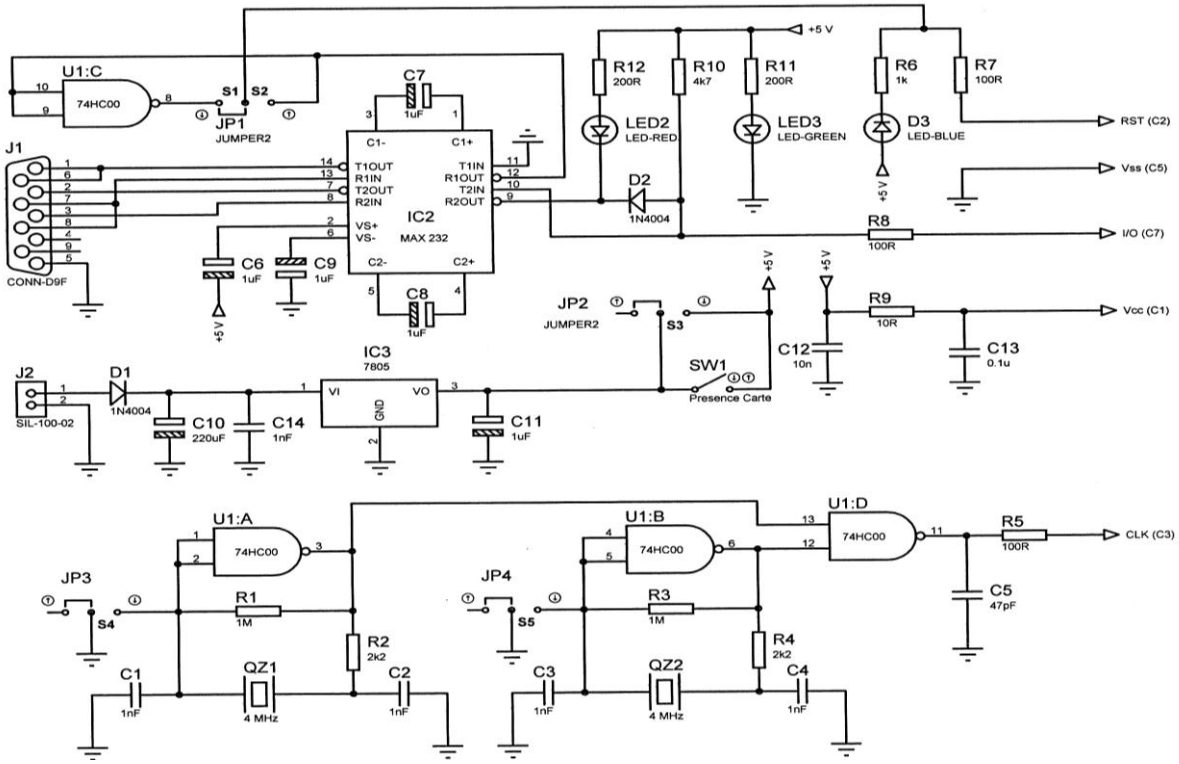


Fig.4. 9: Simulation du lecteur Phoenix sous PROTEUS.

4.5.4. Réalisation du lecteur

4.5.4.1. Liste des composants

Semi-conducteurs IC1 : 74HC00 IC2 : MAX 232 ou ICL 232 IC3 : 78L05 D1 : 1N 4004 D2 : BAT 41, BAR 28, etc. diode Schottky (impératif) LED1, LED2, LED3 : LED, couleur au choix **Résistances ¼ de watt 5 %** R1, R3 : 1 Mohms (marron, noir, vert) R2, R4 : 2,2 kohms (rouge, rouge, rouge) R5, R7, R8 : 100 ohms (marron, noir, marron) R6, R11, R12 : 1 kohms (marron, noir, rouge) R9 : 10 ohms (marron, noir, noir) R10 : 4,7 kohms (jaune, violet, rouge)	**Condensateurs** C1, C2, C3, C4 : 22 pF céramique. C5 : 47 pF céramique. C6, C7, C8, C9 : 1 µF 25 volts chimique radial. C10 : 220 µF 25 V chimique radial C11 : 47 µF 25 V chimique radial C12 : 10 nF céramique C13 : 0,1 µF polyester, Mylar, MKT C14 : 0,22 µF polyester, Mylar, MKT **Divers** QZ1 : Quartz 3,579 MHz en boîtier HC18U QZ2 : Quartz 6,00 MHz en boîtier HC18U J1 : Prise DB9 femelle J2 : Jack femelle 2,1 mm J3 : Connecteur pour carte à puce ISO format ID 1. Interrupteur de détection de présence de carte. Interrupteur de sélection de fréquence Supports de circuits intégrés : 1 x 14 pattes, 1 x 16 pattes

Tab.4. 1: Liste des composants nécessaires.

4.5.4.2. Circuit imprimé

Nous avons réalisé le montage dont le dessin du circuit imprimé est fait sous EAGLE.

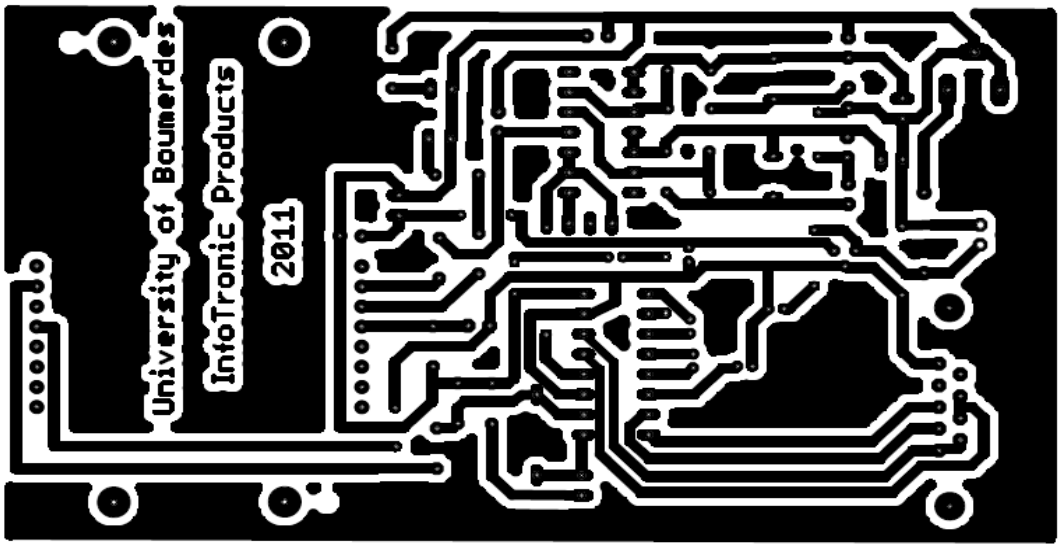


Fig.4. 10: Dessin du circuit imprimé du lecteur Phoenix (coté cuivre).

4.5.4.3. Implantation des composants

Le plan d'implantation est indiqué sur la figure ci-dessous :

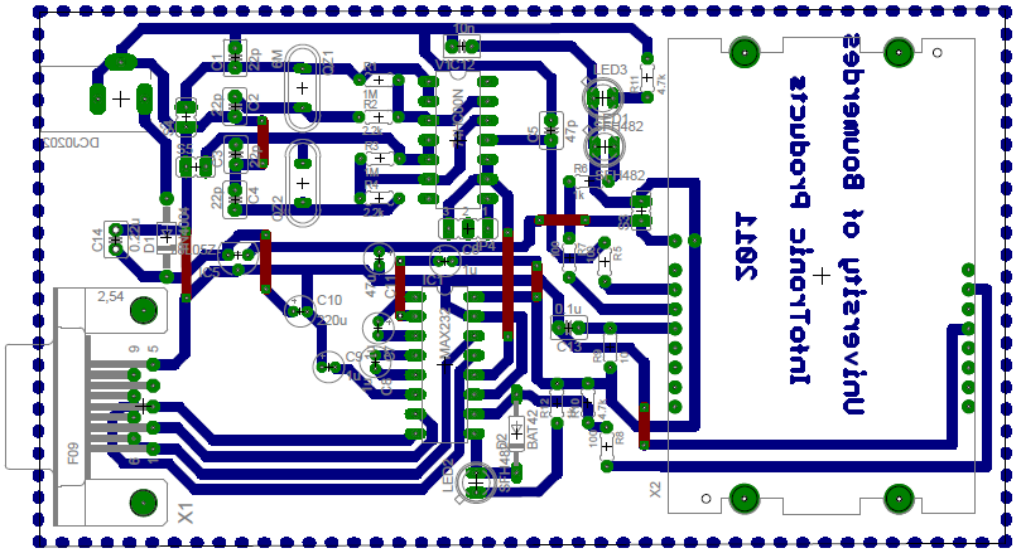


Fig.4. 11: Schéma d'implantation des composants (coté composants).

Après le placement des composants, voici notre lecteur :

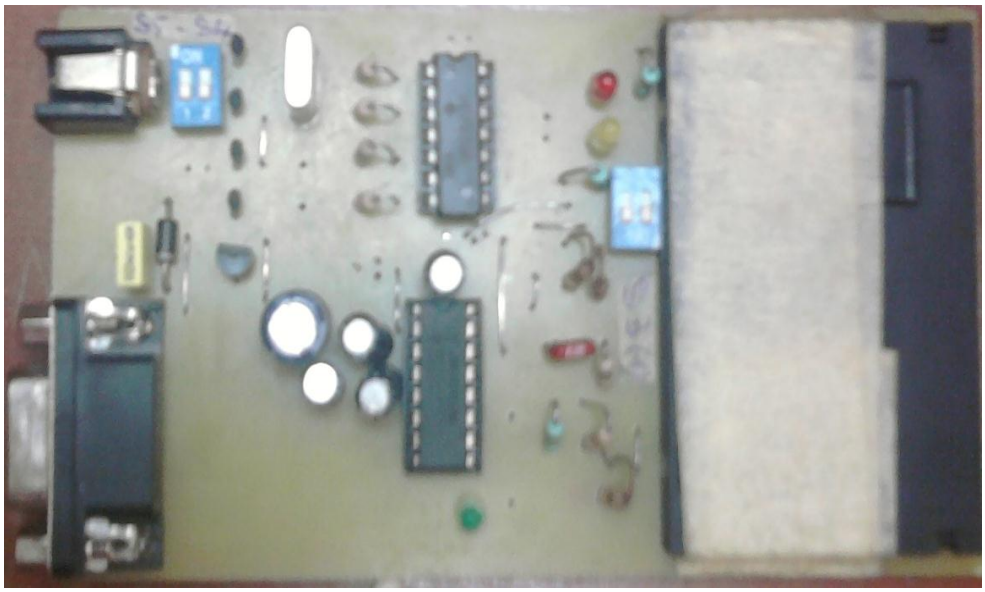


Fig.4. 12: Le lecteur phœnix après implantation des composants.

4.6. Programmation de la carte à puce

Après l'insertion de la carte dans le lecteur, elle envoi l'ATR et attend la réception d'une commande. Si c'est le cas, elle passe au traitement de la commande en question comme le montre le logigramme de la figure 4.13.

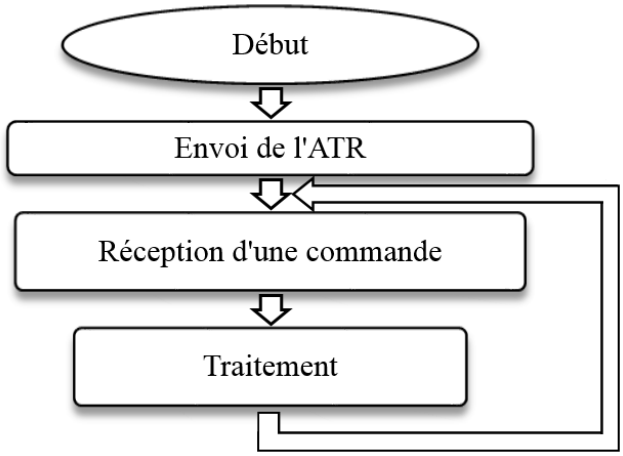


Fig.4. 13: Logigramme du fonctionnement de la carte.

4.6.1. ATR utilisée

L'ATR utilisée correspond à la matricule de l'étudiant : exemple Z060050.

4.6.2. Commandes utilisées

Les commandes traitées par la carte sont :

- La Commande SELECT;
- La commande READ;
- La commande WRITE;
- La commande VERIFY.

La réponse de ces commandes correspond à l'octet SW qui peut prendre deux valeurs :

- SW = 'Y' pour une commande déroulée avec succès.
- SW = 'N' pour une commande déroulée avec échec.

- **Commande SELECT**

L'APDU de la commande SELECT est :

CLA	INS	P1	P2	Lc	Adresse
0	S	0	0	02	xx xx

xx xx : Correspond à l'adresse du fichier à sélectionner dans l'EEPROM sur 2 octets.

Tab.4. 2: La commande SELECT.

- **Commande READ**

L'APDU de la commande READ est :

CLA	INS	P1	P2	Le
0	R	0	0	xx

XX : correspond aux nombres de lignes (enregistrements) à lire.

Tab.4. 3: La commande READ.

- **Commande WRITE**

L'APDU de la commande WRITE est :

CLA	INS	P1	P2	Lc
0	W	0	0	xx

XX : correspond aux nombres de lignes (enregistrements) à modifier.

Tab.4. 4: La commande WRITE.

- **Commande VERIFY**

L'APDU de la commande VERIFY est :

CLA	INS	P1	P2	Lc	Mot de passe
0	V	0	0	06	xx xx xx xx xx xx

xx xx xx xx xx xx : correspond au mot de passe tapé.

Tab.4. 5: La commande VERIFY.

4.6.3. Logigramme de la carte Wafer 2

- **Algorithme principale**

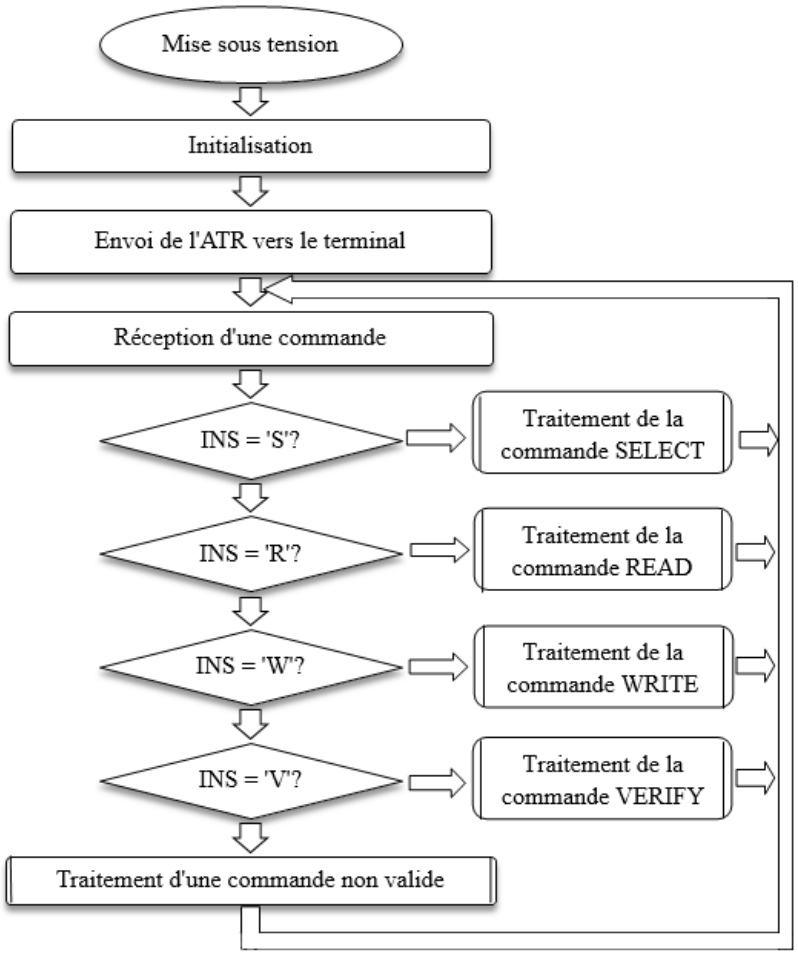


Fig.4. 14: Organigramme principal du programme de la carte.

Algorithme : Traitement de la commande ‘’SELECT’’

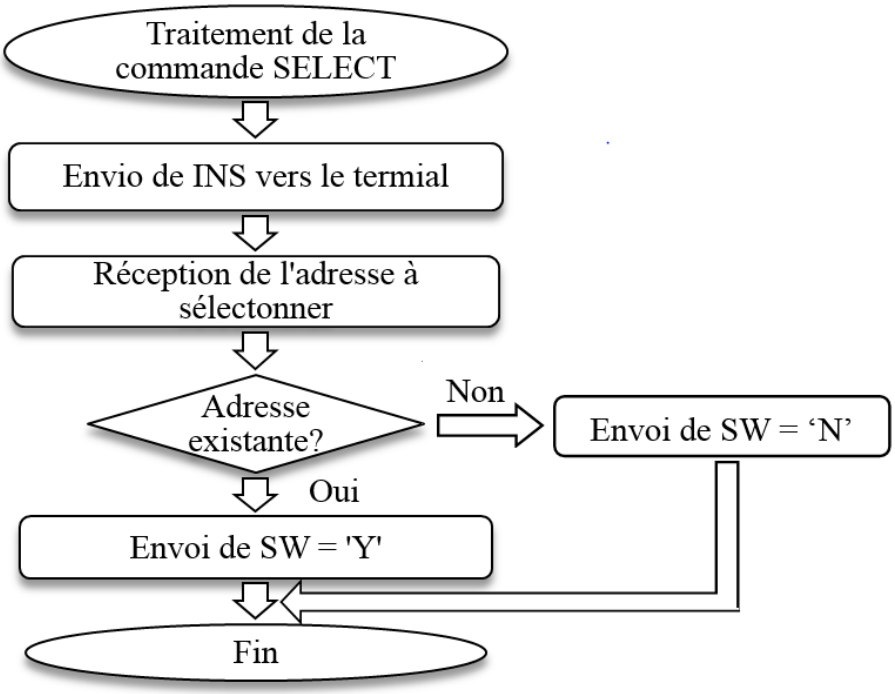


Fig.4. 15: Organigramme du traitement de la commande SELECT.

Algorithme : Traitement de la commande ‘’READ’’

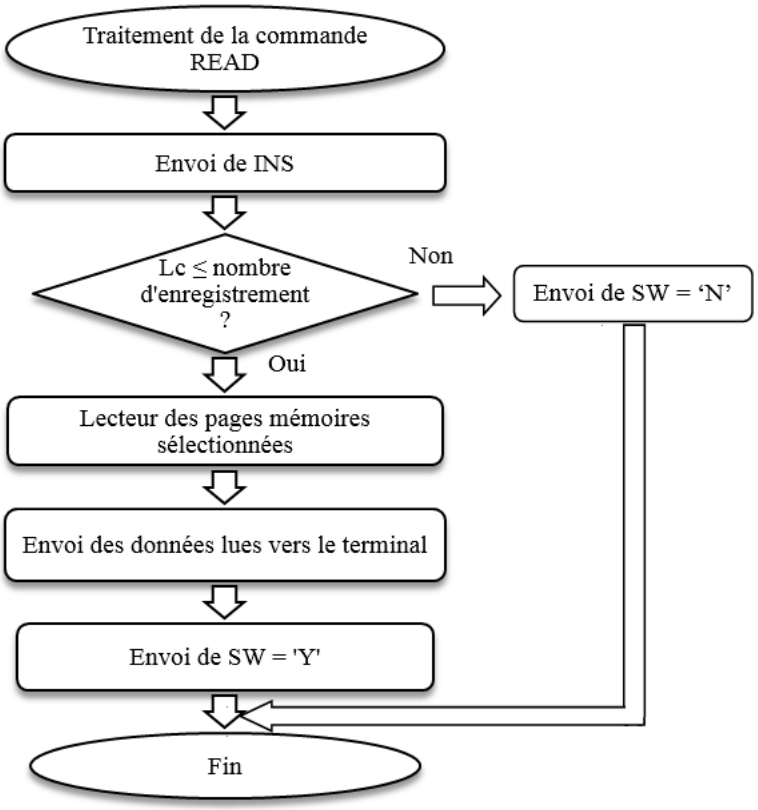


Fig.4. 16: Organigramme de traitement de la commande READ.

Algorithme : traitement de la commande 'WRITE'

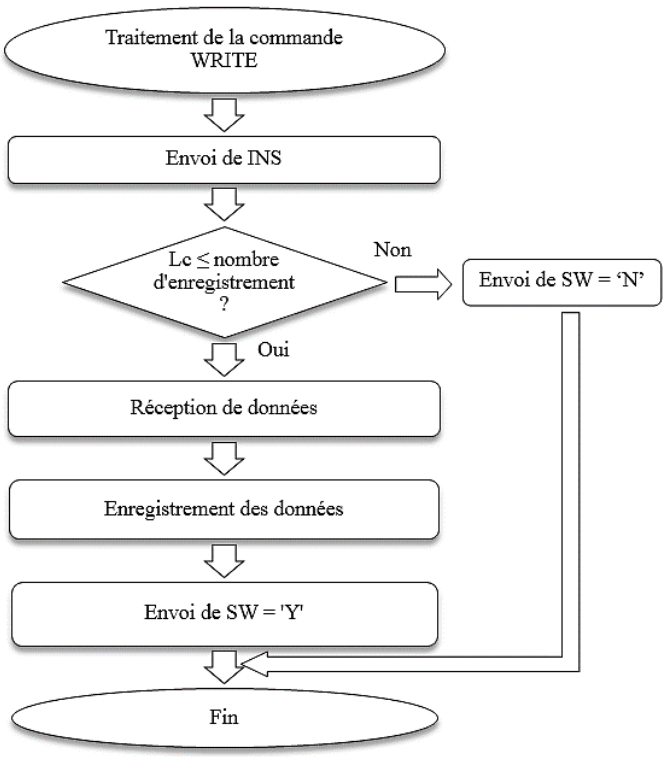


Fig.4. 17: Organigramme de traitement de la commande WRITE.

Algorithme : Traitement de la commande ''VERIFY''

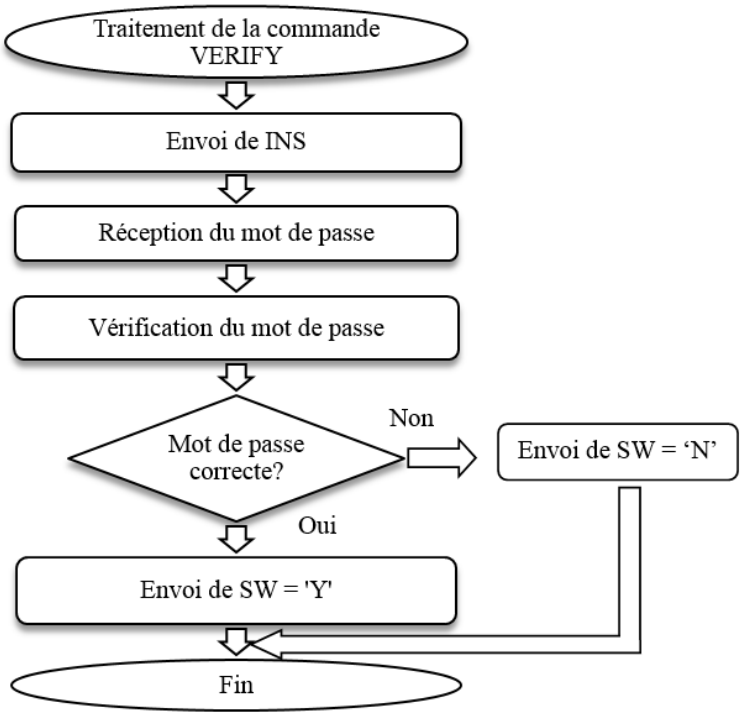


Fig.4. 18: Organigramme de traitement de la commande VERIFY.

Algorithme : Traitement d'une commande non valide

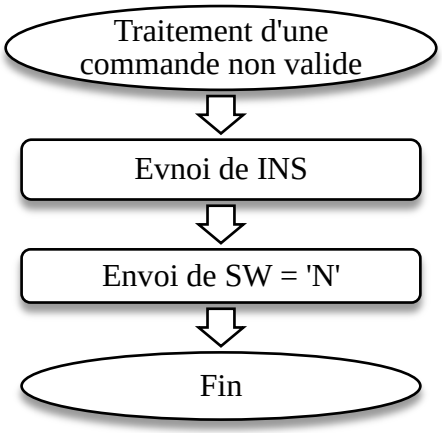


Fig.4. 19: Organigramme de traitement d'une commande non valide.

4.6.4. Organisation de la mémoire EEPROM

Pour l'organisation de la mémoire, on choisit le type de fichier à structure linéaire fixe car il est plus facile à gérer.

1 fichier = 3 blocs

1 bloc = 8 pages

1 pages = 32 bytes

Donc au totale on aura 10 fichiers et 2 blocs (la taille de notre mémoire est de 64k bytes).

4.6.5. Simulation du fonctionnement

4.6.5.1. Schéma de simulation

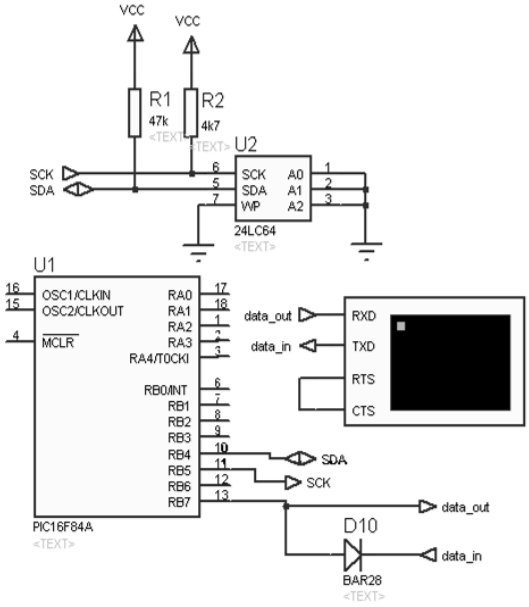


Fig.4. 20: Schéma de simulation de fonctionnement de la carte.

Ce schéma est constitué de la carte Wafer2 (PIC16f84, 24LC64), et d'un terminal virtuel.

Proteus nous offre la possibilité de simuler le fonctionnement du microcontrôleur sans compléter son montage (alimentation, RESET et signal d'horloge). Ces signaux doivent être générés par le lecteur de carte, ce qui fait qu'on considère la carte insérée dans le lecteur.

Le terminal virtuel sert à simuler la communication établit entre la carte et l'interface graphique (logiciel). Avec ses 2 lignes TXD il envoi des données et commandes à la carte, et avec la ligne RXD il reçoit les données et réponses aux commandes envoyées par la carte.

4.6.5.2. Résultats de simulation

Le premier message reçu de la carte est l'ATR (Answer To Reset).

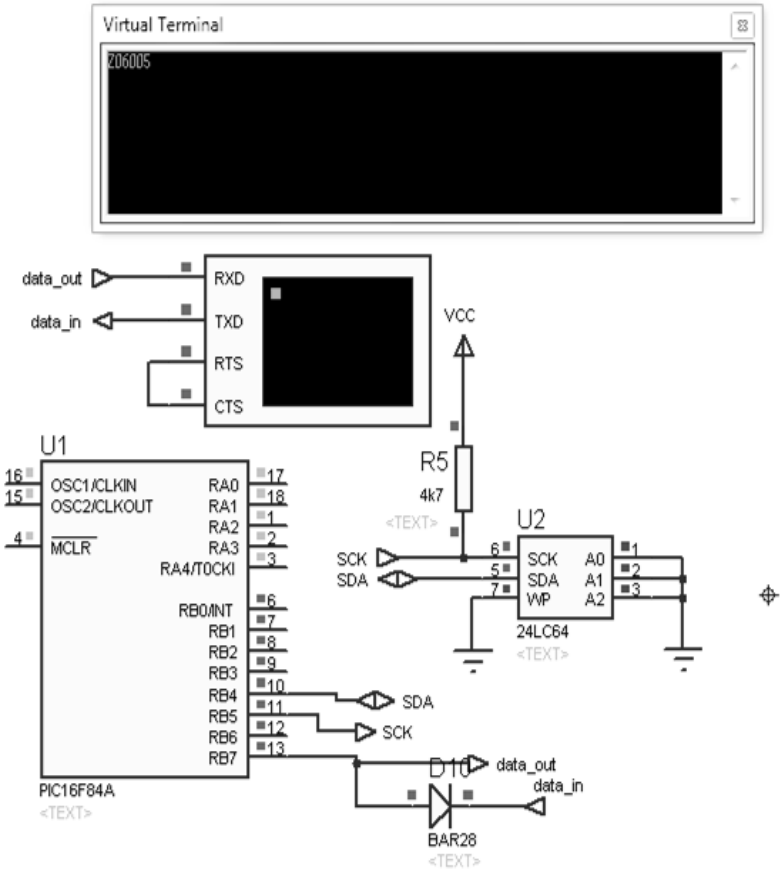


Fig.4. 21: Envoi de l'ATR vers le lecteur.

Maintenant, nous testons le fonctionnement des différentes commandes. Pour cela, la commande doive être tapée dans le terminal virtuel qui se chargera de l'envoyer vers la carte, et la réponse de la carte sera affichée sur le même terminal virtuel.

Nous avons commencé par la commande SELECT comme suite :

- Nous envoyons le TPDU (0S0001) qui indique que la taille de l'adresse est 1 octet ;
- La carte répond par (S) indiquant la réception de la commande SELECT ;
- Ensuite nous envoyons le TPDU (00) indiquant la sélection du fichier N°0 ;
- La carte répond par (Y) pour indiquer le bon déroulement de la commande.

Fig.4. 22: Teste de la commande SELECT.

De la même manière, nous avons vérifié le fonctionnement des autres commandes.

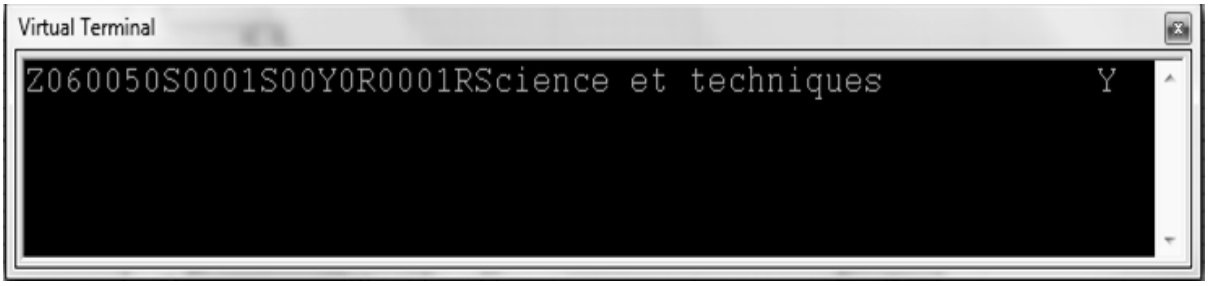


Fig.4. 23: Teste de la commande READ.

Fig.4. 24: Teste de la commande WRITE.

Fig.4. 25: Teste de la commande VERIFY.

4.7. Interface graphique

Notre but est de développer une application Windows en Visual Basic pour interfacer la communication entre la carte et le lecteur au niveau de l'ordinateur.

4.7.1. Schéma général de l'application

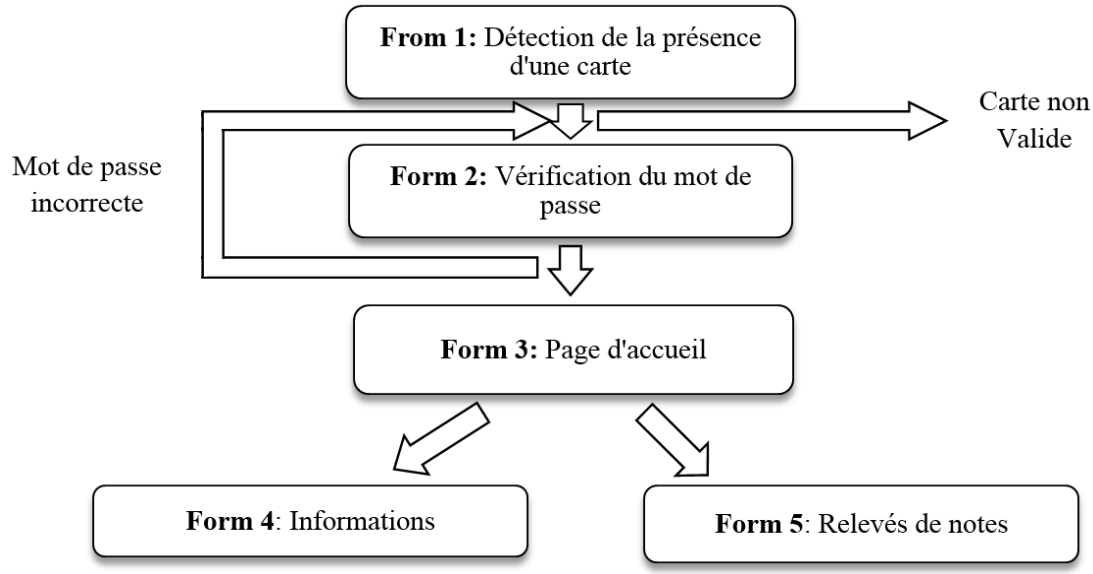


Fig.4. 26: Logigramme de l'application Visual Basic.

Une fois que la partie visuelle de l'application soit terminée et que le code soit ajouté, nous avons passé à la vérification de son fonctionnement.

4.7.2. Procédure de simulation

Nous avons pris le schéma de la carte à puce réalisé précédemment, et nous avons ajouté un port série (DB9) que nous avons relié avec la ligne data de la carte comme le montre la figure suivante :

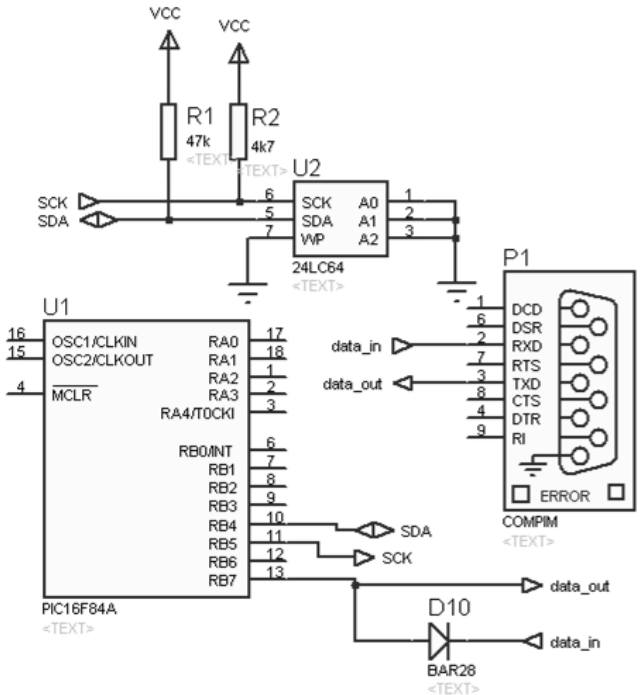


Fig.4. 27: Schéma de simulation fonctionnelle de la carte.

Ce port série va servir de liaison entre la carte et l'interface graphique déjà réalisée avec Visual Basic pour simuler le fonctionnement de tout le système.

Et pour cela, on a besoin d'un port série virtuel, qui relie le port qui se trouve ou niveau de l'interface graphique de Visual Basic, et le port qui se trouve dans le schéma fonctionnelle de la carte sous Proteus.

Pour résoudre ce problème, nous avons choisi un logiciel permettant de créer des ports virtuels dénommé Virtual Serial Driver 6.9 d'Eltima Software.

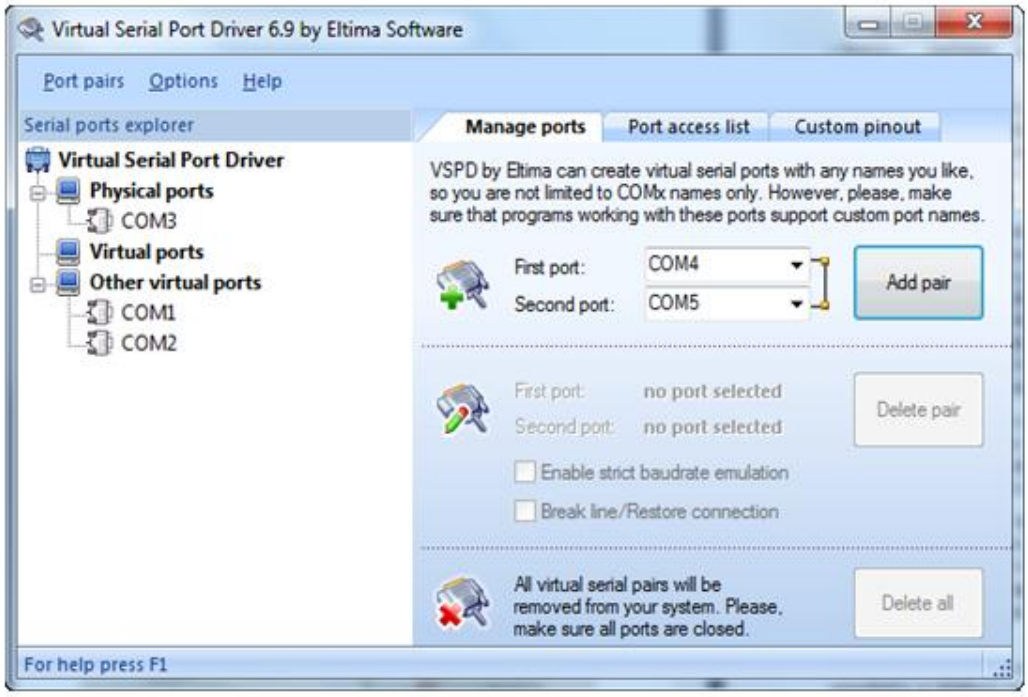


Fig.4. 28: Logiciel permettant la création de ports virtuels.

4.7.3. Simulation du fonctionnement

Pour commencer la simuler nous exécutons l'application crée sous Visual Basic. La première fenêtre qui s'affiche est Form1 (elle nous demande d'insérer une carte).

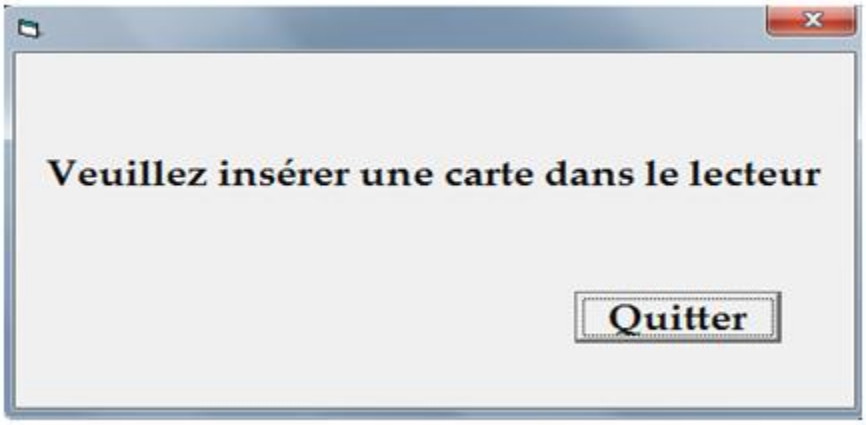


Fig.4. 29: Form1 est la 1ère fenêtre qui s'affiche après le lancement de l'interface graphique.

Si l'interface détecte une carte, qui correspond au lancement de la simulation sous Proteus, et d'après l'ATR reçu on à 2 cas :

- **La carte est non valide** : dans ce cas Form1 change de message pour indique que la carte est non valide et qu'on doit quitter.

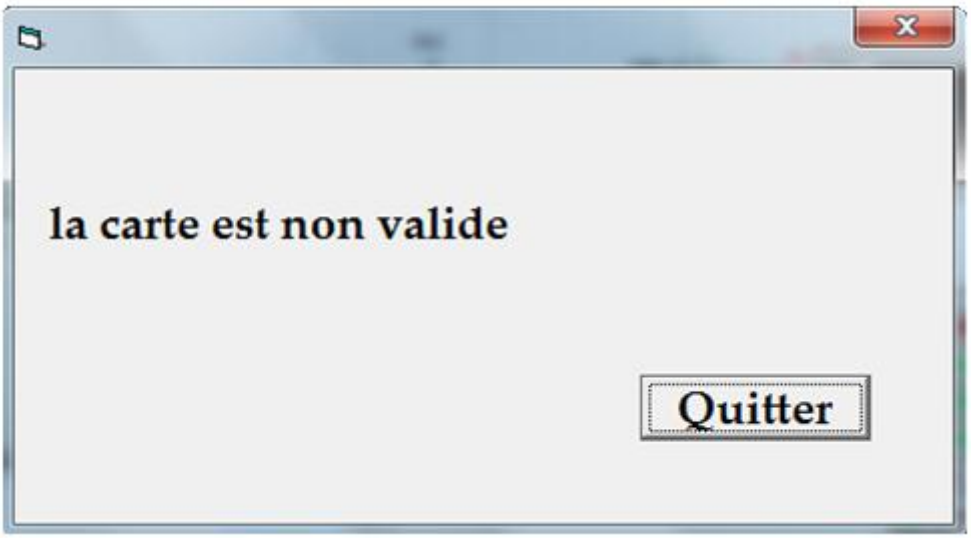


Fig.4. 30: Form1 après détection d'une carte non valide.

- **La carte est valide :** dans ce cas Form1 se ferme, et Form2 s'ouvre et nous demande de saisir le mot de passe comme le montre la figure suivante :

Fig.4. 31: Form2 demande le saisi du mot de passe.

Après la validation du mot de passe on a 3 cas :

- **Mot de passé inférieur à 6 caractères** : Dans ce cas le traitement du mot de passe ne s'effectue pas au niveau de la carte, car dès que Visual Basic détecte le mot de passe il compte le nombre de caractères, et s'ils sont inférieurs à 6 il nous affiche un message d'erreur.

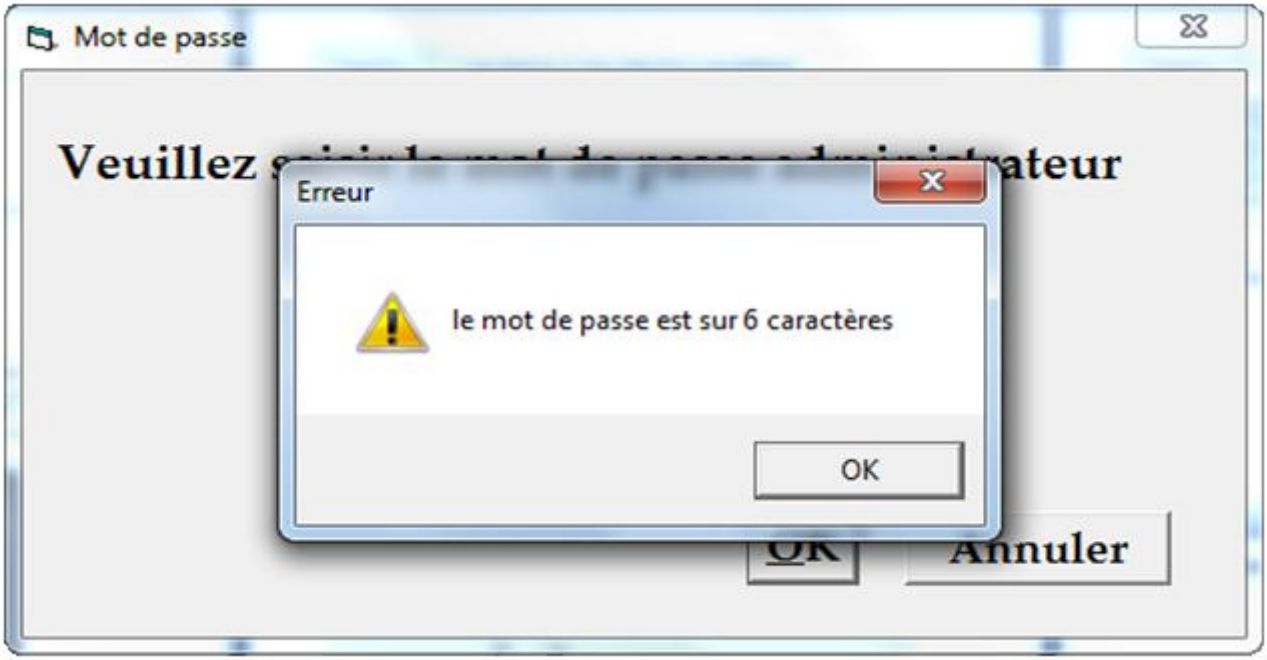


Fig.4. 32: Message d'erreur indiquant le nombre de caractère saisi inférieur à 6.

- **Mot de passé erroné** : Dans ce cas le traitement du mot de passe s'effectue au niveau de la carte, et si le mot de passe envoyer à la carte ne correspond pas à celui qui se trouve à son niveau, un message d'erreur s'affiche.

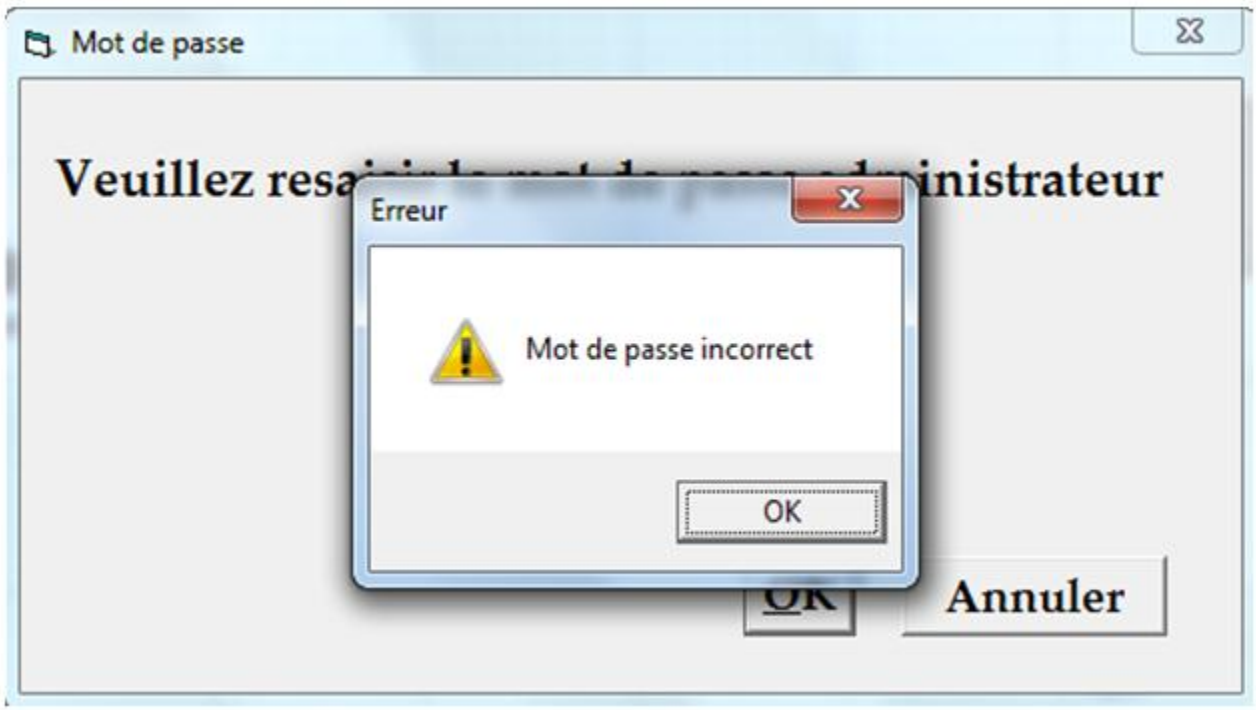


Fig.4. 33: Message d'erreur indiquant que le mot de passe est incorrect.

- **Mot de passe correct** : Dans ce cas Form3 s'affiche

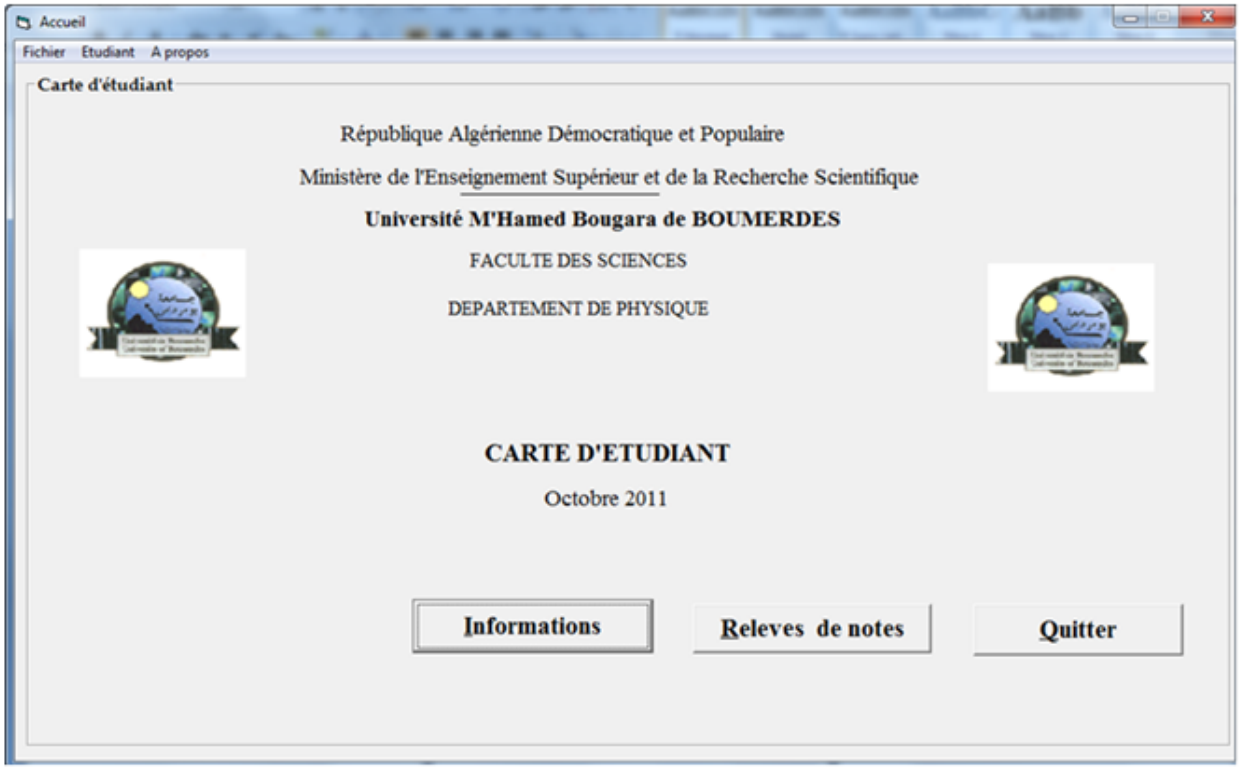


Fig.4. 34: Form3 correspond à la page d'accueil.

Cette fenêtre nous donne accès soit vers les informations de l'étudiant, soit vers ses relevés de notes.

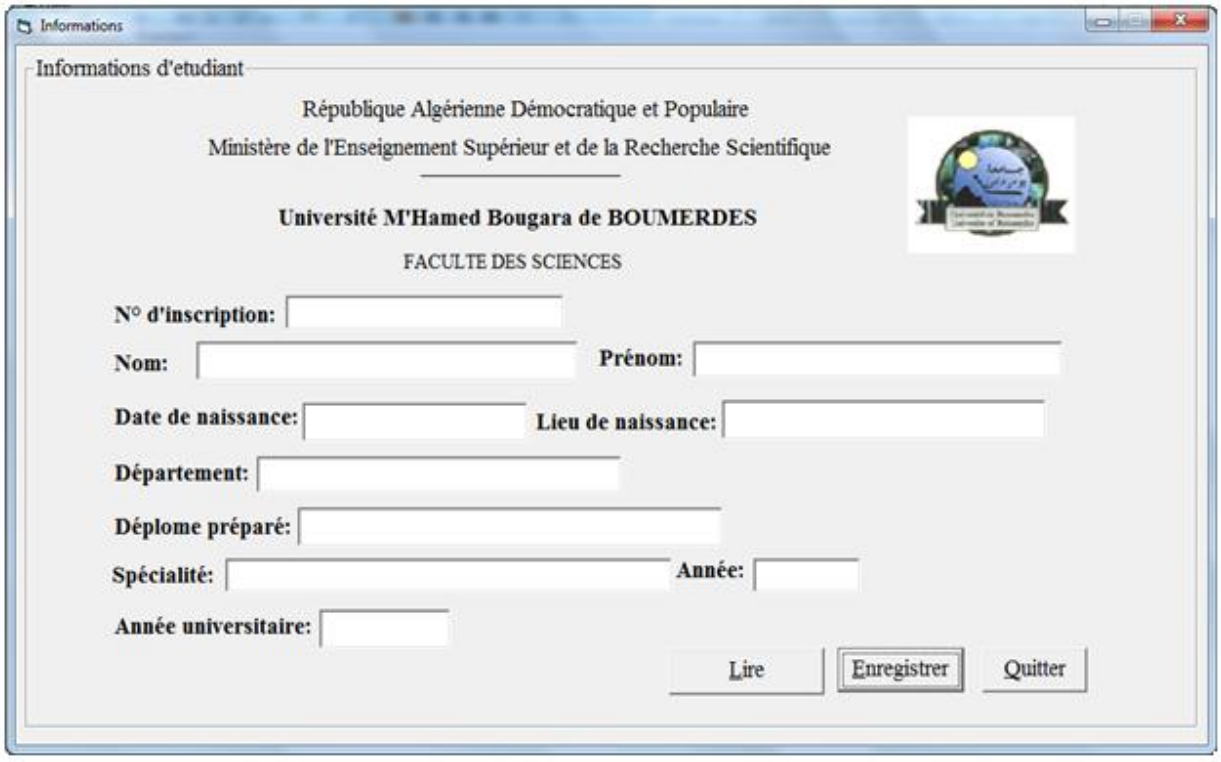


Fig.4. 35: Form4 sert à sauvegarder les informations de l'étudiant.

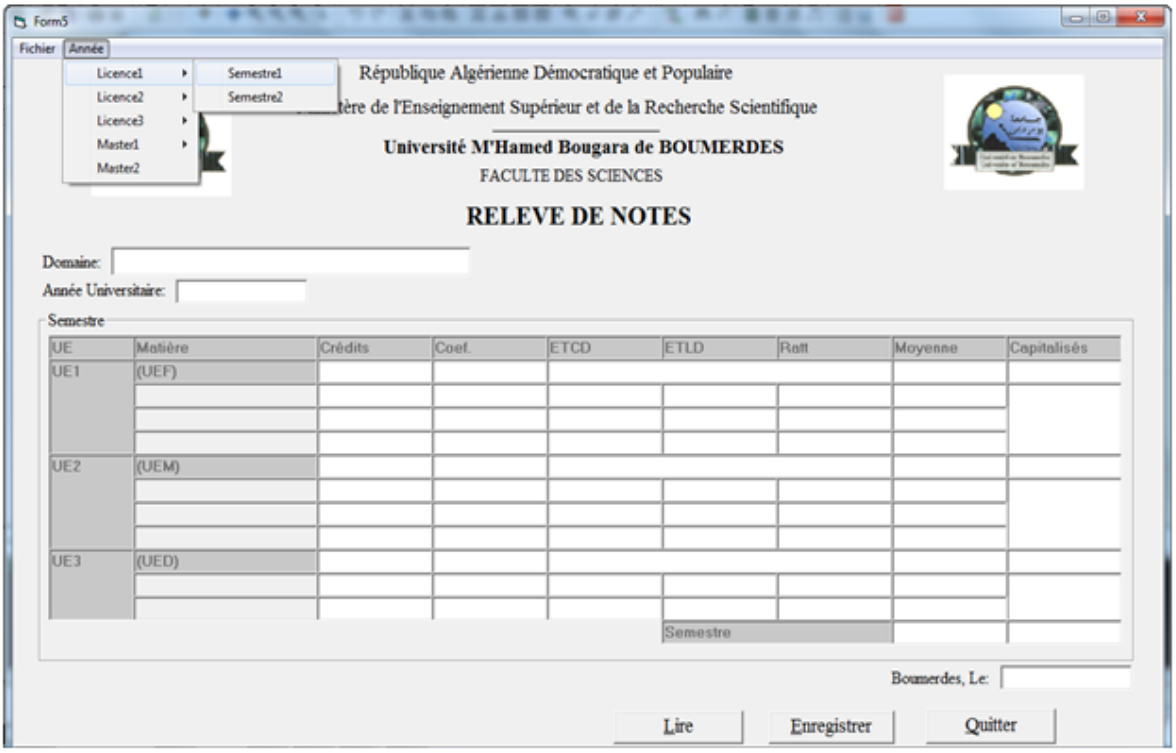


Fig.4. 36: Form5 sert à manipuler les relevés de notes de l'étudiant.

Dans les deux fenêtres on peut LIRE les données et ENREGISTRE les modifications apportées.

Pour les relevais de notes, on doit tout d'abord sélectionner l'année et le semestre désiré avant de lire ou d'enregistrer les données.

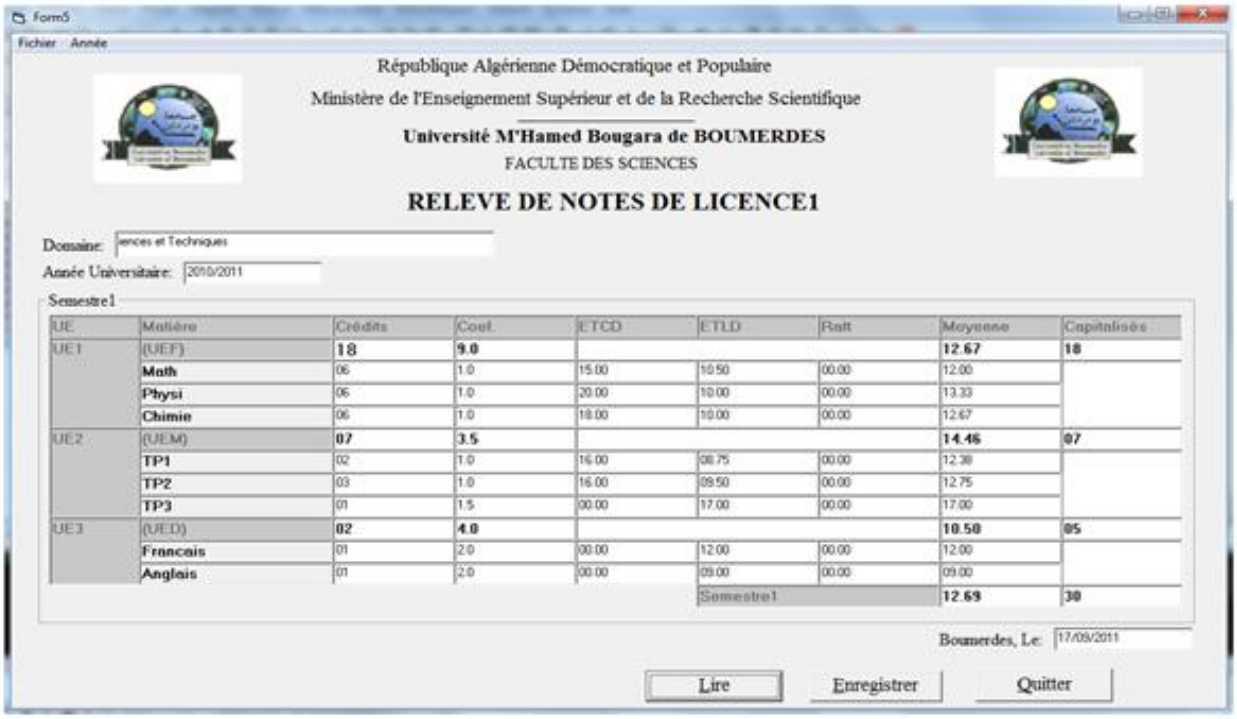


Fig.4. 37: Les données lues de la carte après sélection du semestre 1 de licence1.

Pour les fenêtres informations et relevais de notes, on trouve deux bottons de commande Lire et Enregistrer.

Ces deux commandes Lire et Enregistrer commencent par sélectionner la zone mémoire qu'on désire lire/modifier on lui envoyant la commande SELECT accompagner de l'adresse du fichier, ensuite elles sont suivies par la commande READ/WRITE.

4.8. Conclusion

Au cours de ce chapitre nous avons conçu et réalisé une carte à puce étudiant permettant largement de faciliter le travail de l'administration.

Comme prévu, la carte contient les informations de l'étudiant ainsi que ses relevais de notes durant tout son parcours universitaire. L'accès se fait à l'aide d'un mot de passe. Le saisi de données se fait uniquement par l'administrateur au niveau de la faculté, l'étudiant pourra les consulter mais ne peut apporter aucune modification.

La programmation des protocoles de communication en langage assembleur a permis de choisir une carte à puce à la fois pas cher et suffisante pour l'application et répondre au cahier de charge.

Nous avons choisi la carte à puce Wafer 2 à base du PIC16F84a et le lecteur Phoenix. Au cours de la programmation nous avons rencontré des problèmes ralentissant. Le microcontrôleur de la carte ne prend pas en charge la programmation des protocoles de communication I2C et RS232 ce qui nous a pris beaucoup de temps en particulier pour I²C. En plus de cela, la programmation du port série en Visual Basic a aussi causé des problèmes car la documentation disponible est très pauvres.

Chapitre 5

Conception d'une machine de vote électronique embarquée à base d'un microcontrôleur et utilisant une carte à puce

Certains des travaux présentés dans ce chapitre ont fait l'objet de 3 articles : Un est présenté lors de la conférence [IDT'14, IEEE], le 2ème est présenté dans la conférence [ACECS'15] et publier dans le journal [CSA 2015] et le dernier est publier dans le journal [IJSISE 2016].

5.1. Introduction

Au cours de ce chapitre, nous avons exploité le système de vote électronique dans le but de l'améliorer tout en optimisant les ressources utilisées. Cela, afin de réduire les coûts et assurer leurs sécurités et fiabilités.

Il existe deux types : le premier est basé sur la visite d'un bureau de vote et le deuxième se fait à distance. Chacun d'eux a des avantages et des inconvénients.

La première partie de la contribution consiste à améliorer le système de vote électronique qui se fait dans les bureaux de vote en rendant ce système rapide en manipulation en utilisant une carte à puce.

La deuxième partie de la contribution consiste à fusionner les deux systèmes de vote électroniques existant, en prenant en compte les avantages de chacun d'entre eux, pour concevoir un nouveau qui à la fois rapide, sûr et économique que les systèmes conventionnels.

La simulation est faite avec Proteus Professional Software V7. Les microcontrôleurs sont programmés en langage d'assemblage sous MPLAB IDE V.8.56 logiciel. L'interface graphique est conçue avec Visual Basic 6.

5.2. Systèmes de vote existants [42-44]

5.2.1. Système de vote traditionnel

Dans les systèmes de vote traditionnels, les choix et les intentions des électeurs sont représentés sous forme de papier. Comme l'illustre la figure 5.1, une fois d'un électeur entrent dans le bureau de vote, il est identifié avant de lui permettre de voter.

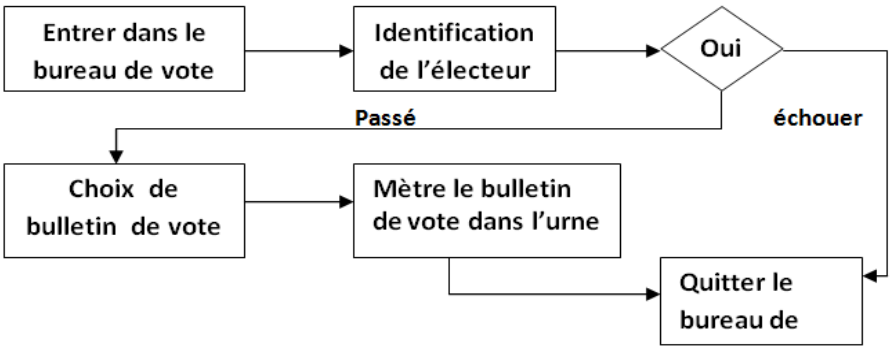


Fig.5. 1: Système de vote classique [42].

5.2.2. Systèmes de vote électroniques

Le vote papier traditionnel peut être fastidieux et incommode. Le vote électronique non seulement accélère l'ensemble du processus, mais il est moins cher et plus confortable pour les électeurs et les autorités. Il réduit également les chances d'erreurs.

Un système de vote électronique doit fournir toutes les fonctionnalités de base que le vote classique. En outre, il devrait fournir plus de services afin de rendre le processus plus fiable et sécurisé. Les données de l'élection sont enregistrées, stockées et traitées principalement comme information numérique.

La recherche sur le vote électronique est un sujet très important pour le progrès de la démocratie. De ce fait, il connaisse aujourd'hui des développements importants.

Les machines de vote électroniques sont des dispositifs électroniques simples utilisés pour enregistrer des voies à la place des bulletins de vote et des boîtes qui ont été utilisés précédemment dans le système de vote classique.

Il existe deux types de systèmes de vote électronique reconnus. Le 1er est basé sur la visite d'un bureau de vote et le 2ème se fait à distance via des réseaux informatiques et internet.

5.2.2.1. Systèmes de vote électroniques basé sur la visite d'un bureau de vote

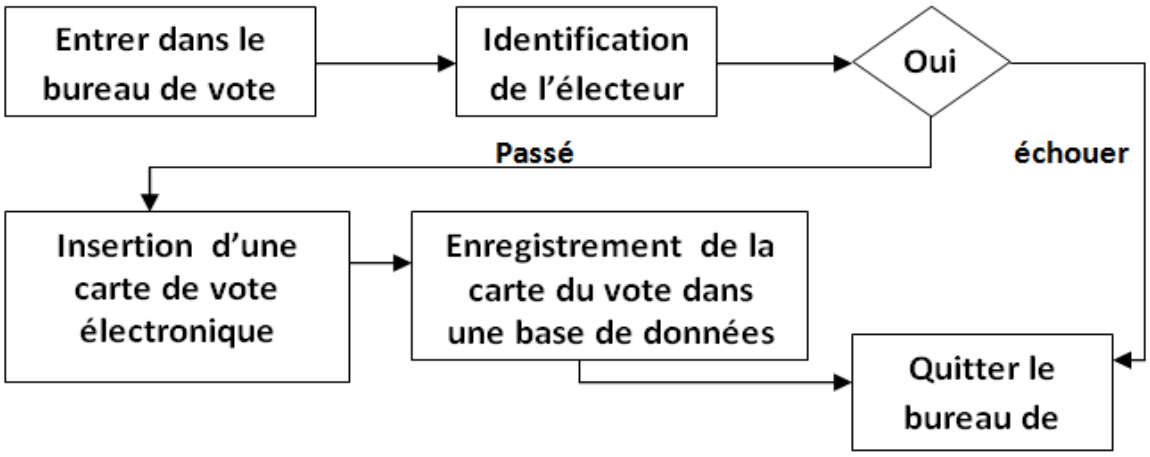


Fig.5. 2: Systèmes de vote électronique basé sur la visite du bureau de vote [42].

Pour ce type, le vote électronique s'effectue dans les bureaux traditionnels. Ou lieu d'utiliser le papier, des machines sont placées permettant à l'utilisateur de sélectionner leurs choix. Le processus est illustré sur la figure 5.2.

Dans ce cas, les électeurs sont toujours identifiés par l'utilisation de cartes d'identification. Les électeurs ne remplissent pas les cartes de vote sous la forme de papier, mais ils utilisent des boutons poussoirs sur différents appareils électroniques disponibles dans les bureaux de vote.

Il existe différentes formes de ces machines utilisées à traverser le monde. Les différentes machines de vote électronique sont bien définies dans les travaux de Prasath et Mekala publié en 2014 [45] et celui d'Ali et al publié en 2014 [46] également. La majorité de ces machines sont à base de microcontrôleur. Plusieurs travaux ont été faits pour améliorer ces machines comme [45-53]

Les avantages de ce système sont les résultats qui sont rapidement calculés. En utilisant une machine autonome sans aucune connectivité réseau, personne ne peut interfacer avec sa programmation et manipuler le résultat. Cependant, ce n'est pas sûr de prévenir le vote multiple par une seule personne. En plus de ça, l'utilisation du personnels dans le processus prennent beaucoup de temps et augmentent les coûts pour la vérification d'identité qui s'effectuée manuellement, signature de l'électeur à la fin du vote, le cachet de la carte de vote à la fin, la récupération de la liste électorale.

5.2.2.2. Systèmes de vote électroniques à distance

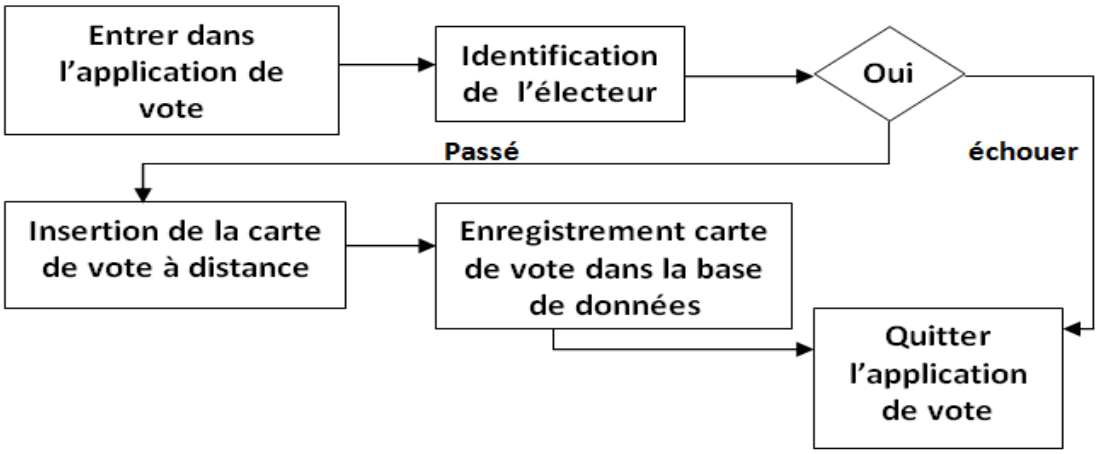


Fig.5. 3: Systèmes de vote électronique à distance [42].

Le deuxième type de système de vote électronique se fait à distance via des réseaux informatiques et Internet. Habituellement, les électeurs ont la possibilité de voter en utilisant des ordinateurs aux niveaux des bureaux de vote, à des endroits éloignés, ou bien, leurs téléphones portables. A figure 5.3 illustre le déroulement d'un de ces systèmes.

Il existe de nombreux types de systèmes de vote à distance. Plusieurs travaux ont été réalisés pour améliorer ces machines comme [54-56]. La majorité des travaux sont très récents.

Les systèmes de vote électronique à distance ont l'avantage du contrôle de l'identité qui se fait très rapidement avec différentes techniques, selon le système utilisé, avec une authentification en utilisant une carte à puce d'identité, authentification biométrique ou bien une authentification dynamique englobant plusieurs techniques à la fois. Malgré cela, il y'a beaucoup de risques de sécurité car le vote se déroule par internet en plus de cas les gens ont tendance à rejeter ce type en raison des résultats qui peuvent être manipulés ainsi que l'anonymat non assurée.

5.3. Exigences d'un système de vote [42, 44, 54]

La conception d'un « bon» système de vote, que ce soit électronique ou classique, doit satisfaire un certain nombre de critères :

- **L'anonymat** du vote d'un électeur doit être présent.
- Le système de vote doit également être **inviolable** pour contrecarrer une large gamme d'attaques, y compris le bourrage des urnes par votes et dépouillement incorrect par les initiés.
- **Précision** : un système est précis s'il est impossible de le modifié.
- **Vérifiabilité** : un système est vérifiable si quelqu'un peut vérifier indépendamment que tous les votes ont été comptés correctement.
- **Démocratie** : un système est démocratique si elle ne permet que les électeurs admissibles à voter et il veille à ce que chaque électeur peut voter qu'une seule fois.

- **Confidentialité** : un système est privé si ni les autorités électorales, ni personne d'autre ne peut lier un bulletin de vote à l'électeur qui l'a émis et aucun électeur et prouver qu'il ou elle a voté en manière particulière.
- **Commodité** : un système est pratique s'il permet aux électeurs d'exprimer leurs votes rapidement, en une seule séance, et avec un minimum d'équipement ou de compétences particulières.
- **Flexibilité** : un système est flexible s'il peut être adapté à différents types de vote et a un nombre varié de candidats.
- **Mobilité** : un système est mobile s'il n'y a pas de restrictions sur l'emplacement à partir de laquelle un électeur peut voter.
- **Fiabilité** : un système est fiable si elle exécute et maintient ses fonctions en permanence.
- **Cohérence** : un système est conforme s'il fonctionne de façon efficace à chaque endroit, dans chaque situation, et les fonctions s'effectuent exactement comme prévu.
- **L'acceptation sociale** : un système à l'acceptation sociale si elle dispose d'une réception favorable et est perçu comme étant le système aussi efficace par la population de vote.

5.4. Une machine de vote électronique améliorée utilisant un microcontrôleur et une carte à puce

5.4.1. Enoncé du problème

Les autorités administratives utilisent les machines de vote électronique pour :

- Réduire le temps de collection des résultats ;
- Diminuer le risque d'erreurs ou de manipulations ;
- Contrôler les coûts administratifs

Les problèmes qui se posent avec les machines de vote existantes sont :

- Le contrôle de l'identité qui :
  - Prend du temps ;
  - Augmente les coûts.
- Prévenir le vote multiple par une seule personne ;
- Signature de l'électeur à la fin du vote ;
- Le caché de la carte.

5.4.2. Méthodologie

Pour résoudre ces problèmes, ce travail propose la conception d'une machine de vote utilisant une carte à puce.

Ce système va :

- Assurer l'anonymat en utilisant une carte à puce qui ne contient pas les données de l'utilisateur ;
- Utilise une authentification avec carte à puce et mot de passe avec un nombre d'essai limité pour plus de sécurité ;
- Enregistre le vote dans la carte à puce pour prévenir le vote multiple et remplacer le cachet de la carte de vote ;
- Calcule les résultats ;

- Utilise une authentification administrative avec une carte à puce et un mot de passe pour afficher les résultats.

5.4.3. Description du système à concevoir

Le système est composé de deux parties : la carte à puce et la machine comme le montre la figure 5.4.

La carte à puce utilisée est la Wafer 1. Elle contient un microcontrôleur PIC 16F84a qui assure le dialogue avec la machine et la mémoire 24LC16 de type EEPROM pour sauvegarder les données. La machine aussi comporte un microcontrôleur PIC 16F84a qui représente son composant principal associé à une mémoire EEPROM (24LC64) pour enregistrer la liste des électeurs. La machine communique avec l'utilisateur en utilisant un clavier de 12 touches et un afficheur LCD de 2 lignes de 20 caractères.

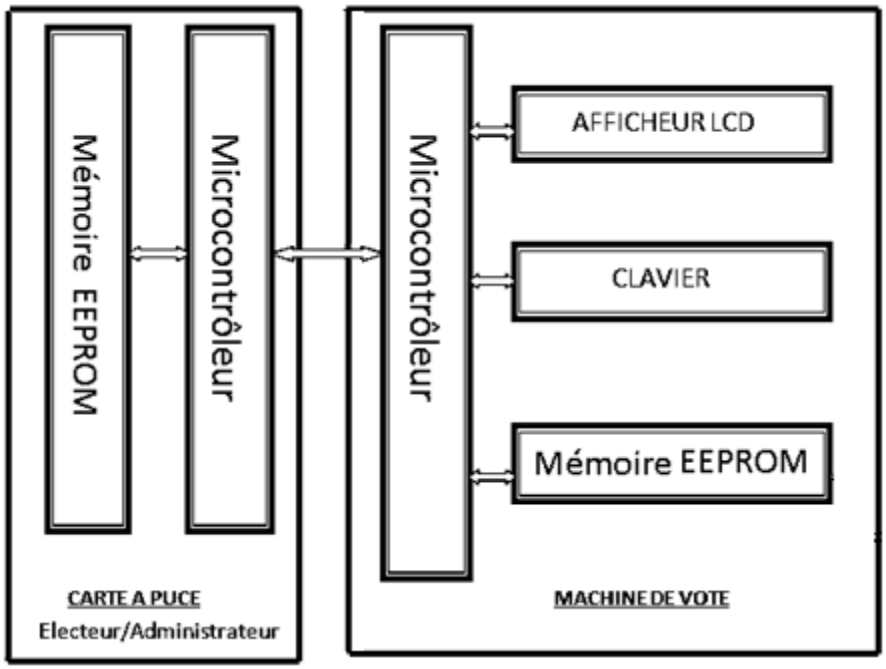


Fig.5. 4: Bloc diagramme du système à concevoir.

5.4.4. Programmation des cartes à puce

5.4.4.1. Organigramme

Dans ce système nous avons utilisé deux cartes à puce : une carte administrateur et une carte électeur mais le principe de fonctionnement est le même (voir la figure 5.5).

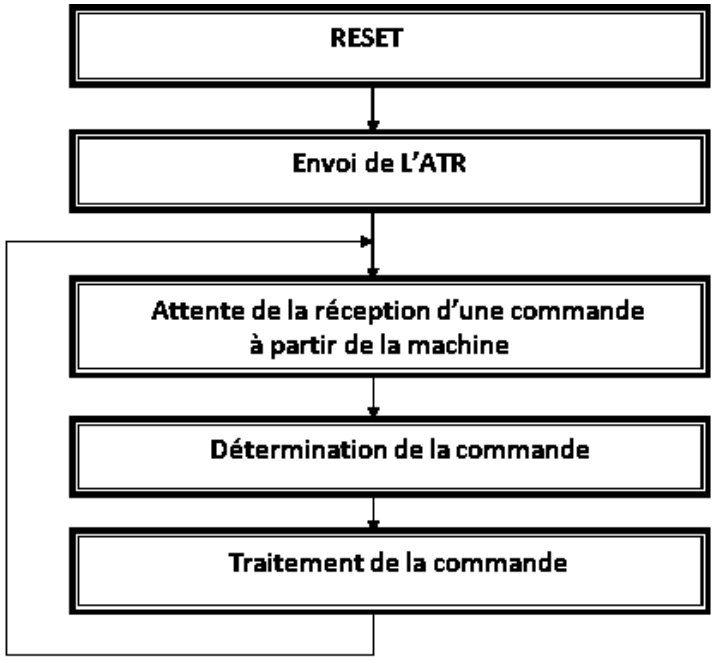


Fig.5. 5: Organigramme de la carte à puce.

Une fois que la carte à puce est insérée dans le lecteur (machine de vote électronique) elle est remise à zéro. Ensuite elle envoi l'ATR (Answer To Reset) vers son lecteur pour échanger les informations nécessaires afin d'assurer la communication. Par la suite, la carte passe à l'attente de la réception d'une commande à partir du lecteur et dès qu'elle reçoit une, elle passe à la détermination de la commande et son traitement. En fin, elle revient à l'attente de réception d'une nouvelle commande.

5.4.4.2. ATR utilisées

Comme nous avons utilisé deux types de carte à puce, leurs ATRs étaient différents (voir les figures 5.6 et 5.7).

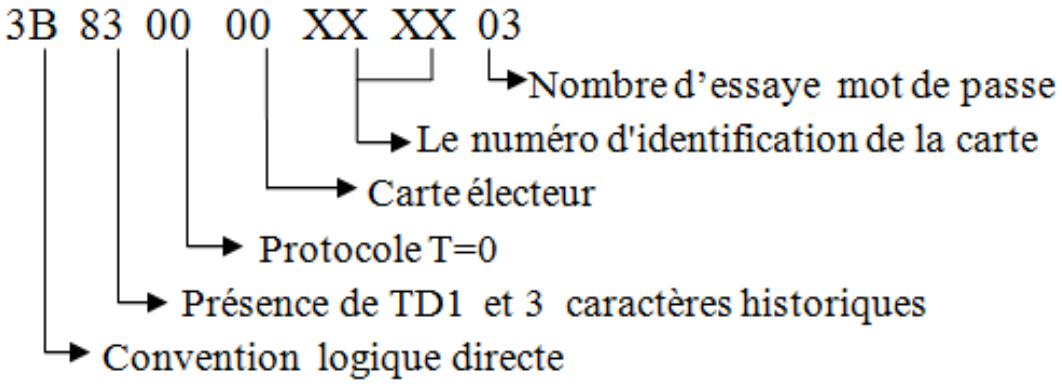


Fig.5. 6: ATR de la carte à puce électeur.

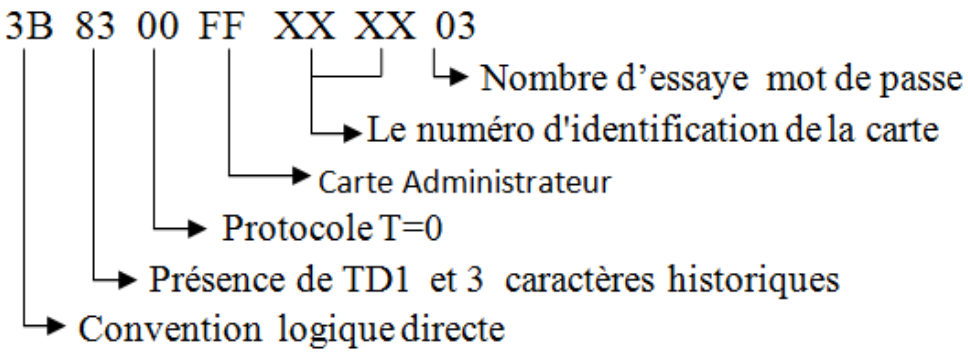


Fig.5. 7: ATR de la carte à puce administrateur.

5.4.4.3. Commandes utilisées

La carte à puce communique avec la machine en utilisant 3 commandes APDU.

- **La commande VERIFY** : utilise 6 caractères pour la vérification d'un mot de passe. Le format de la commande VERIFY est présenté dans le tableau 5.1.

Commande APDU					
CLA	INS	P1	P2	Lc	Données
00	20	00	00	06	xx xx xx xx xx xx

xx : 1 octet du mot de passe.

	Réponse APDU	
Etat	SW1	SW2
Succès	90	00
Erreur	98	40

Tab.5. 1: Format de la commande VERIFY.

- **La commande SELECT :** utilisée pour sélectionner une adresse d'un octet. Le tableau 5.2 donne le format de cette commande.

Commande APDU					
CLA	INS	P1	P2	Lc	Donnée
00	A4	00	00	01	xx

xx : 1 octet d'adresse.

	Réponse APDU	
Etat	SW1	SW2
Succès	90	00
Erreur	6A	82

Tab.5. 2: Format de la commande SELECT.

- **La commande UPDATE RECORD :** Permet d'écrire 1 octet dans la carte à puce à l'adresse déjà sélectionnée. Il suffit de changer la valeur de Lc pour envoyer plus de données à la carte. Le tableau 5.3 donne le format de la commande APDU.

Commande APDU					
CLA	INS	P1	P2	Lc	donnée
00	DC	00	00	01	xx

xx : 1 octet de donnée.

	Réponse APDU	
Etat	SW1	SW2
Succès	90	00
Erreur	92	40

Tab.5. 3: Format de la commande UPDATE RECORD.

5.4.5. Programmation de la machine de vote électronique

5.4.5.1. Organigramme

La machine authentifier les utilisateurs en utilisant un mot de passe de 6 caractères. Cela en utilisant un clavier de 12 touches. Si le mot de passe est correcte nous allons avoir accès au système, sinon le nombre d'essai du mot de passe sera décrémenté ; sachant que le nombre d'essai maximum est 3 et après la carte sera bloquée.

Pour une carte administrateur, la machine affiche les résultats de l'élection. Pour la carte électeur, la machine invite l'électeur à voter en insérant le numéro qui correspond à son candidat. (Voir l'organigramme de la figure 5.8).

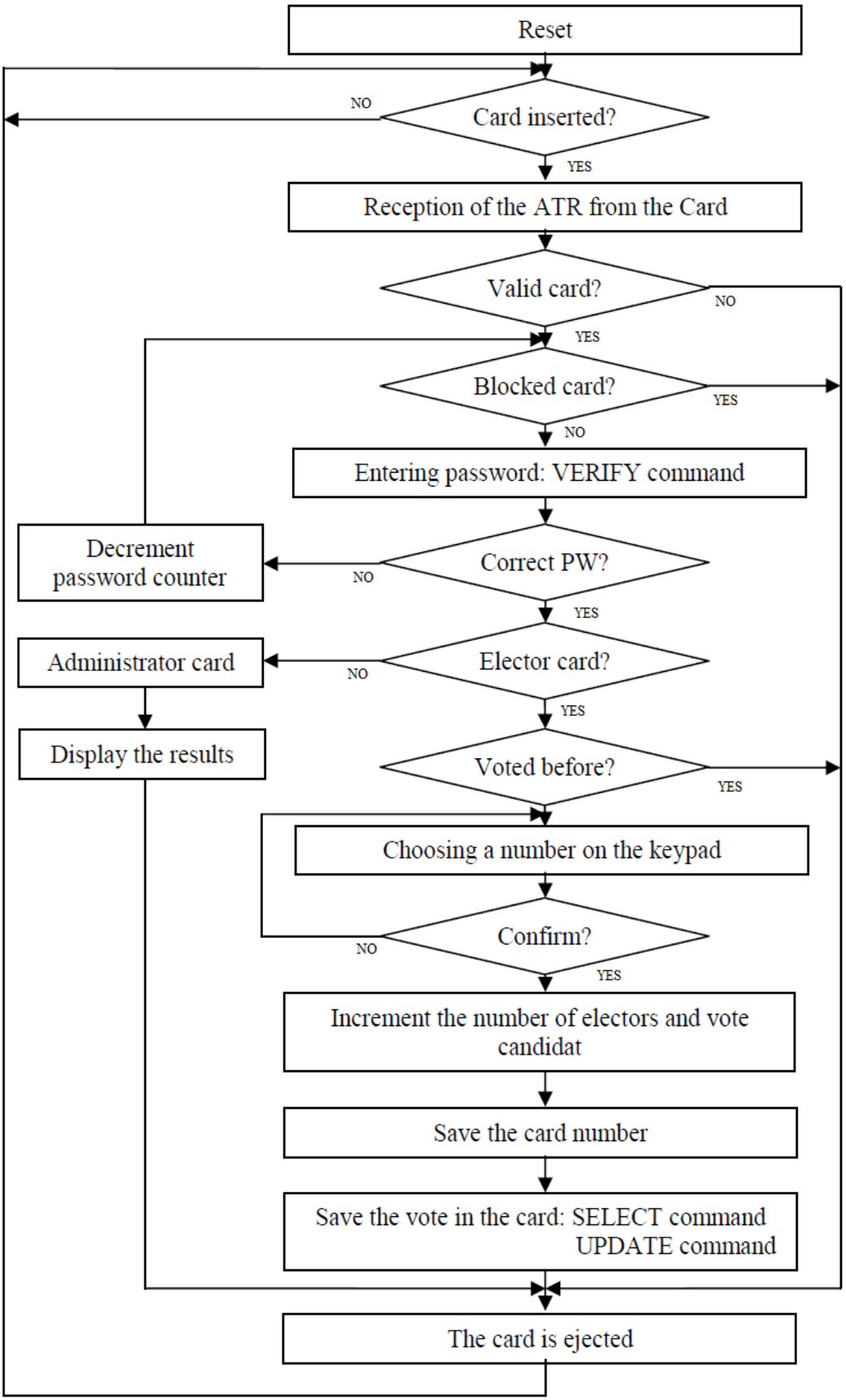


Fig.5. 8: Organigramme de la machine.

5.4.5.2. Déroulement du vote

La machine est installée dans un bureau de vote initialisée avec la liste des électeurs ayant droit. Pour la sécurité des électeurs il peut y avoir un agent de sécurité dans le bureau de vote mais à part ça ce système n'a pas besoin de personnelles.

Une fois qu'un électeur se présente au bureau, il insert sa carte à puce de vote dans la machine. La machine l'invite à entrer son mot de passe. Si le mot de passe est faux, la machine l'invite à le vérifier une autre fois (3 fois au maximum). Si le mot de passe est juste, l'électeur est invité à entrer le numéro correspondant à son candidat. Afin d'éviter des erreurs de frappe, la machine lui demande de confirmer son choix. Si ce n'est pas le bon choix, l'électeur saisi à nouveau le numéro de son candidat et le confirme. Une fois le vote confirmé, l'électeur peut retirer sa carte et sortir du bureau.

A la fin du vote, l'administrateur revient au bureau de vote et à l'aide de sa carte à puce administrateur s'authentifier au niveau de la machine afin d'afficher les résultats. De préférence, avec la présence des candidats ou de leurs représentants.

5.4.6. Simulation du système

5.4.6.1. Circuit de simulation

Afin de vérifier le fonctionnement de la machine de vote électronique , nous avons réalisé son schéma sous PROTEUS comme le montre la figure 5.9.

Le circuit de simulation est divisé en deux parties : la machine et la carte à puce. La carte à puce contient un microcontrôleur PIC 16F84a et la mémoire EEPROM 24LC16. La machine contient un microcontrôleur 16F84a, une mémoire EEPROM 24LC64, un clavier de 12 touches et un écran LCD de deux lignes de vingt caractères.

La simulation est faite avec PROTEUS Professional V7. Les microcontrôleurs sont programmés en langage d'assembleur sous MPLAB IDE V.8.56. Le virtuel terminal est utilisé pour afficher les échanges entre cartes à puce et machine.

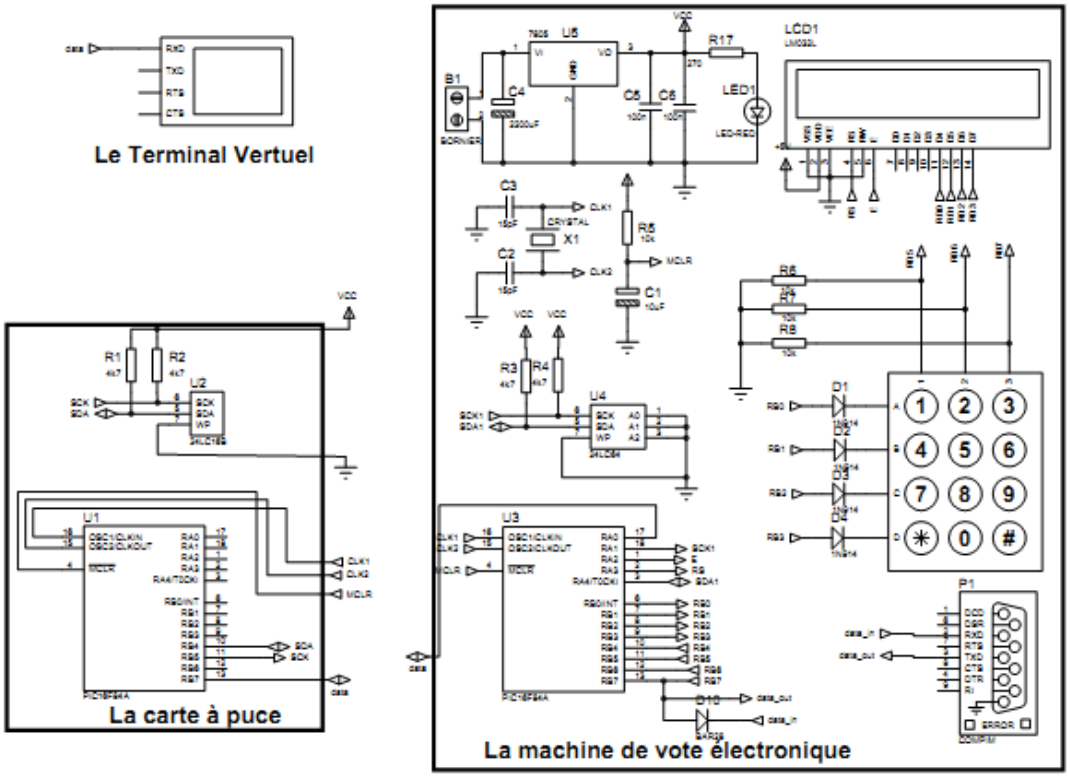


Fig.5. 9: Circuit du système de vote électronique conçu.

5.4.6.2. Simulation

Quand l'électeur insert la carte à puce la machine vérifié si la carte n'est pas bloquée et que l'électeur n'a pas voté avant. L'écran LCD affiche un message « Enter your password » Comme indiqué dans la figure suivante.

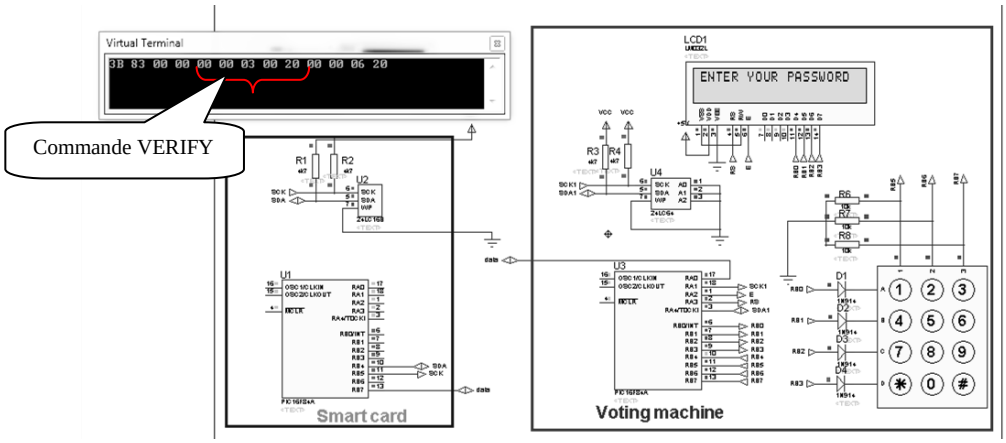


Fig.5. 10: La machine invite l'utilisateur à taper son mot de passe.

L'électeur saisi son mot de passe.

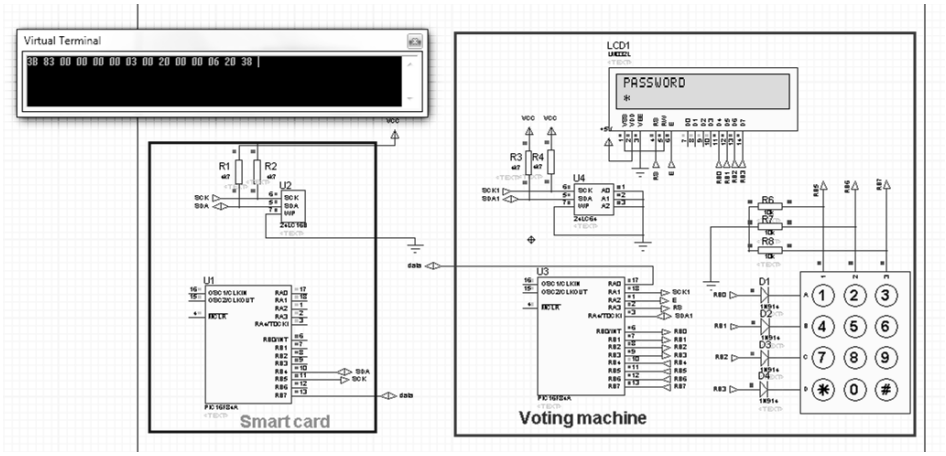


Fig.5. 11: L'utilisateur commence à saisir son mot de passe.

Le teste est effectué avec deux mots de passe : le premier est incorrecte et le second est correcte.

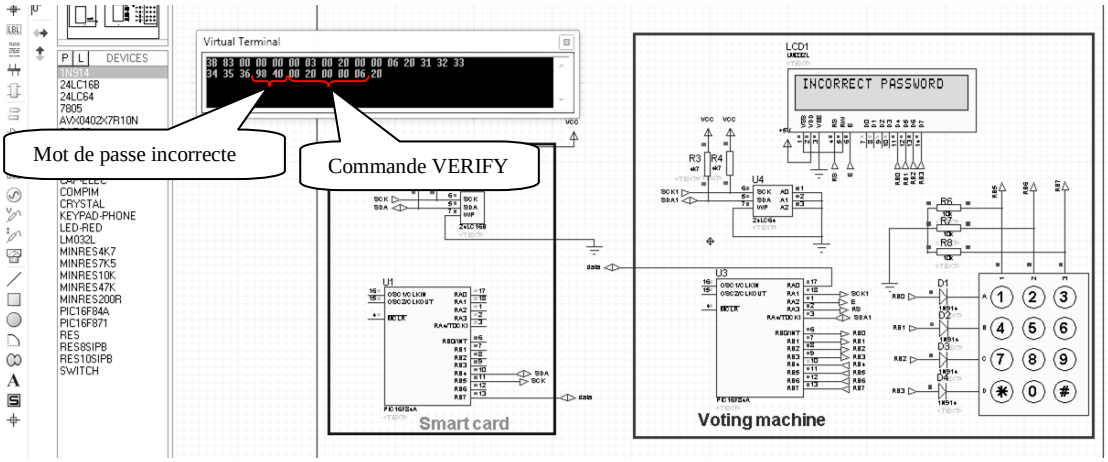


Fig.5. 12: Le mot de passe est incorrect.

Quand le mot de passé est incorrect la machine demande à nouveaux à l'électeur de saisir son mot de passé, la machine lui offre un nombre d'essai limité à 3.

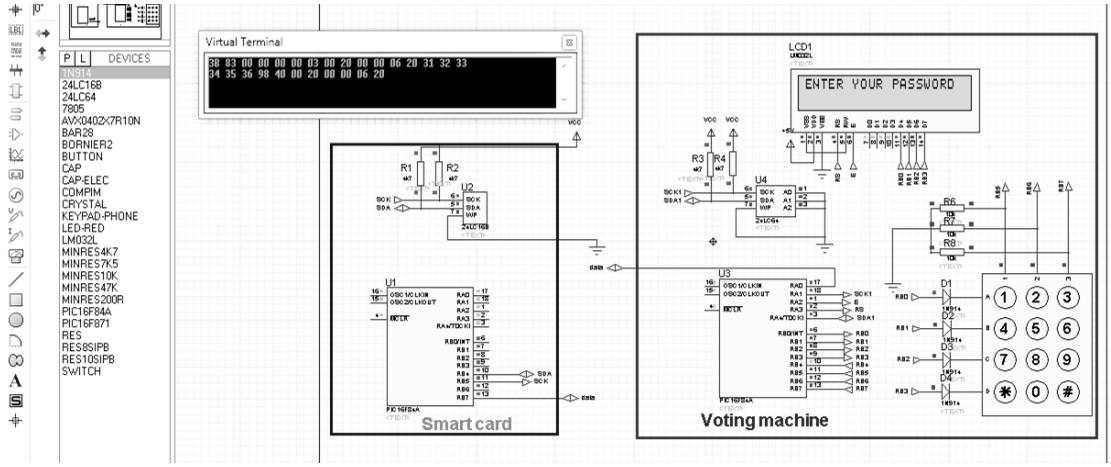


Fig.5. 13: Envoi de la commande VERIFY à nouveau.

Quand le mot de passé est correcte le machine invite l'électeur à choisir un numéro correspondant au candidat sur le qu'elle il souhait voter.

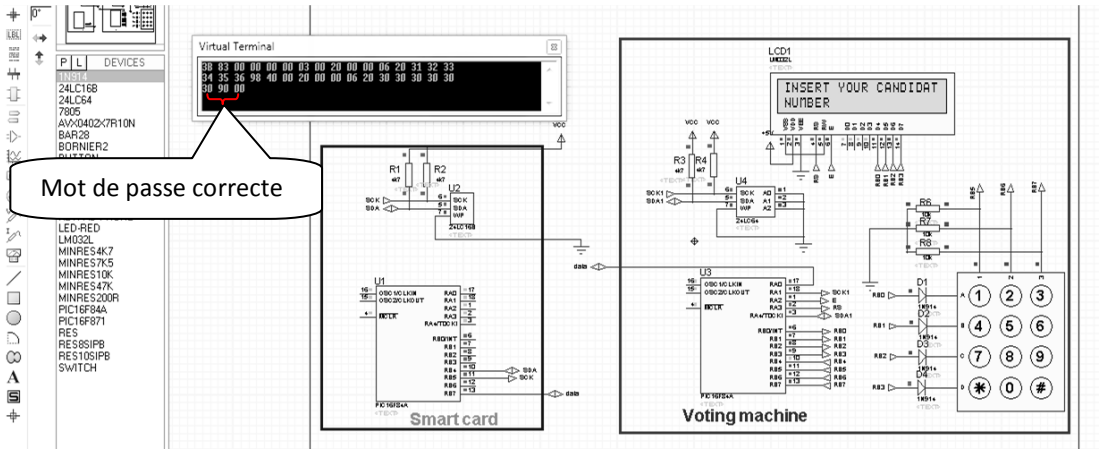


Fig.5. 14: Mot de passe correcte.

Si l'électeur choisi un numéro de candidat non existant, la machine va l'invité à nouveaux à saisir un numéro de candidat.

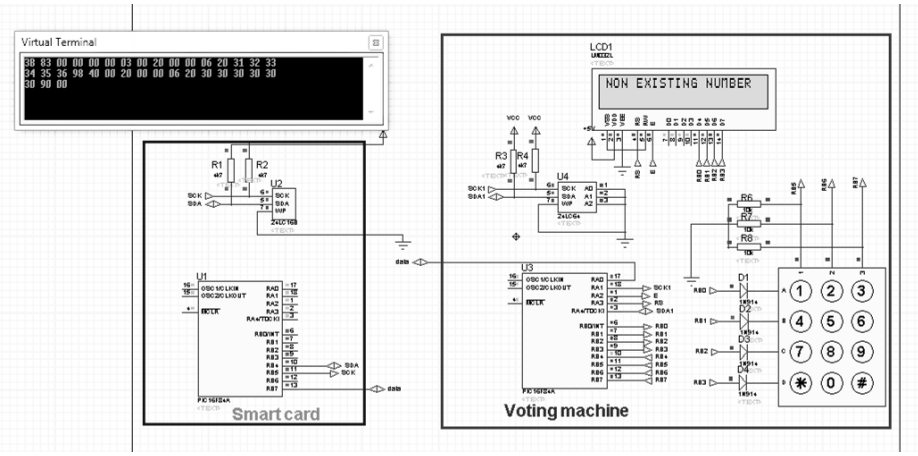


Fig.5. 15: La sélection d'un numéro de candidat non existant.

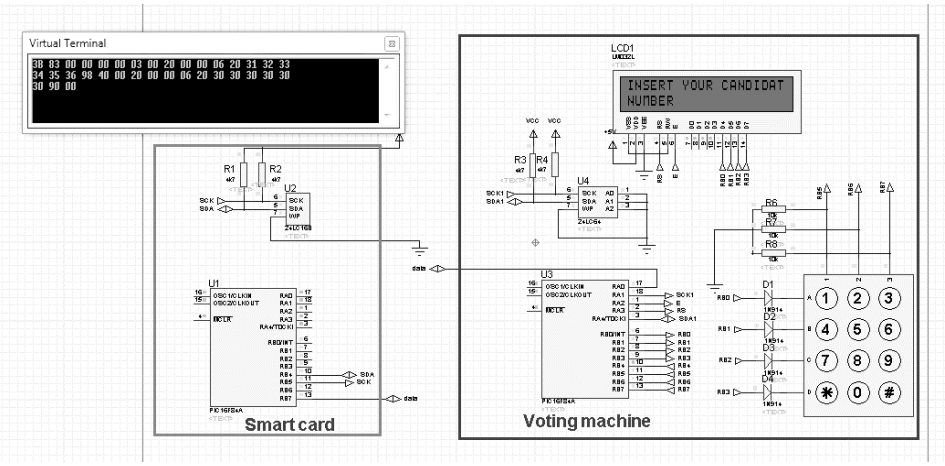


Fig.5. 16: La machine invite l'électeur à introduire le numéro du candidat choisi à nouveau.

Quand l'électeur choisi un bon numéro de candidat, la machine va afficher sur le LCD le numéro du candidat choisi.

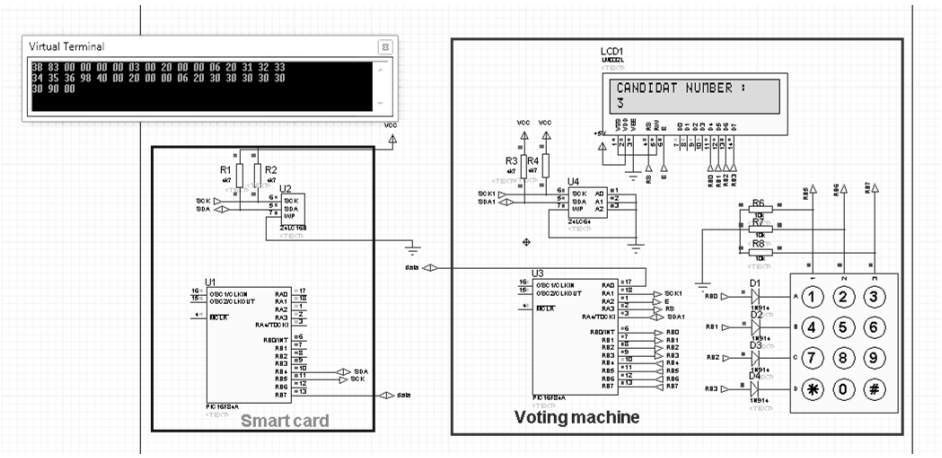


Fig.5. 17: La machine affiche le numéro du candidat sélectionné.

Pour ne pas avoir d'erreur ou un appui accidentelle sur une touche du clavier par exemple, la machine invite l'électeur à valider par la touche « * », ou annuler par « # ».

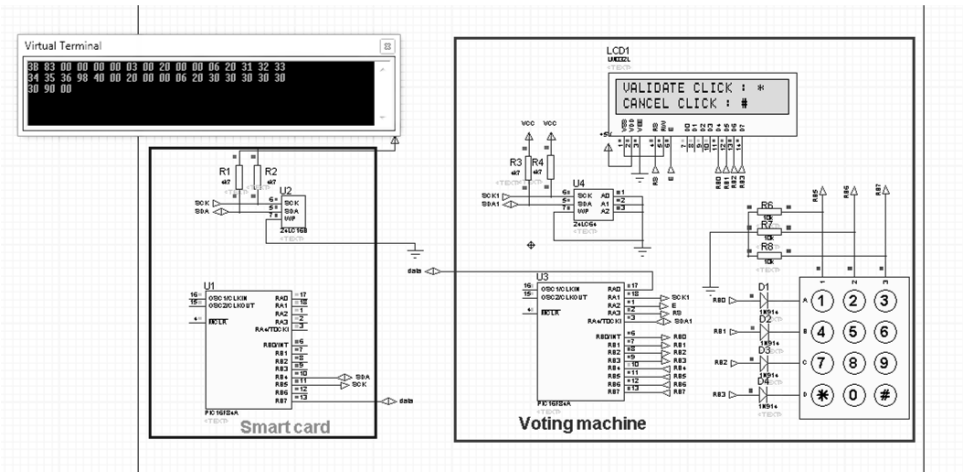


Fig.5. 18: Validation ou annulation du vote.

Si l'électeur appui sur « # » pour annulé, il sera invité a sélectionné à nouveaux un numéro de candidat.

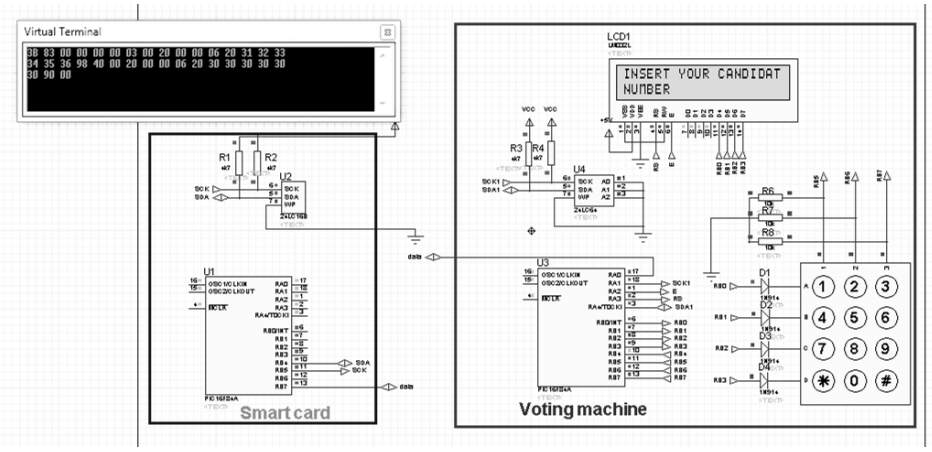


Fig.5. 19: La machine invite l'électeur à choisir un autre candidat.

Après la confirmation du choix d'un candidat, deux compteurs internes sont incrémenté dans l'EEPROM interne du microcontrôleur (U3) : le premier incrémente le nombre de voix totale dans le bureau et le 2ème incrémente le nombre de voix correspondant au candidat choisi.

Le numéro de série de la carte à puce électeur correspond à une adresse dans la mémoire EEPROM de l'appareil (U4), l'écriture du caractère 'V' dans cette adresse indique que le porteur de la carte en question a déjà voté.

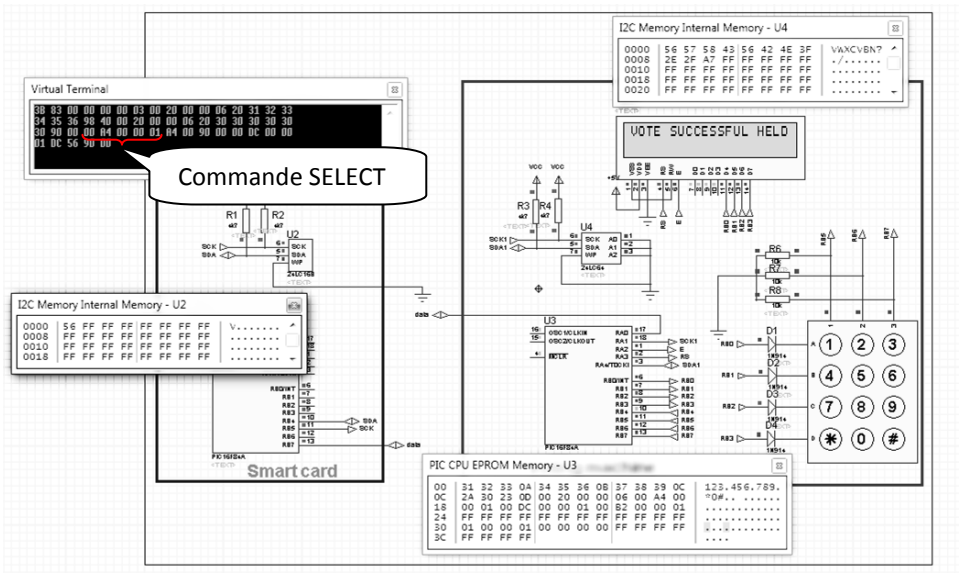


Fig.5. 20: Traitement de la commande SELECT.

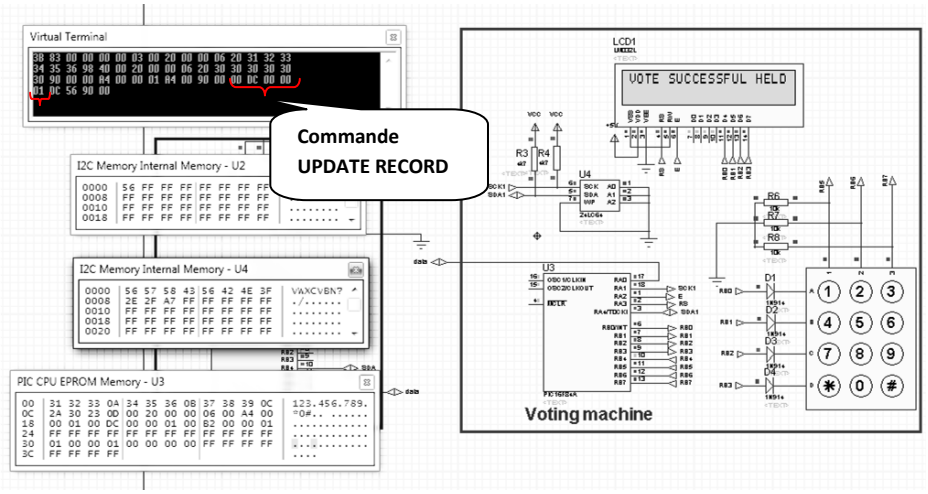


Fig.5. 21: Traitement de la commande UPDATE RECORD.

5.4.7. Résultats et discussion

La simulation de la machine fonctionne normalement avec plusieurs possibilités. Le choix d'un candidat se fait en utilisant un clavier. La machine compte le nombre d'électeurs et de voies pour chaque candidat.

Le système obtenu a permis de résoudre quelques problèmes liés à ce type de vote, ce qui a rendu le système :

- Rapide en manipulation ;
- Préviennent le vote multiple ;
- Sécurisé avec l'authentification des utilisateurs.

Tout ceci en utilisant une carte à puce dans les bureaux de vote. En plus de cas :

Le système est flexible en utilisant un clavier permettant d'avoir un nombre important de candidats (non limité par quelques boutons poussoirs). La récupération de la liste électorale est assurée par l'enregistrement des numéros de série des cartes à puce électeur ayant voté dans la machine.

5.5. Une machine de vote électronique embarquée avec une carte à puce

5.5.1. Enoncé du problème

Il existe deux types de systèmes de vote électronique reconnus.

- Le premier est basé sur la visite d'un bureau de vote. Dans ce cas :
  - Les électeurs sont encore identifiés par l'utilisation de cartes d'identification, comme le cas du système conventionnel ;
  - Les électeurs ne remplissent pas les cartes de vote en papiers, mais des machines électroniques avec des boutons poussoirs sont utilisées pour sélectionner les candidats ;
  - Les avantages de ce system sont :
    - Les résultats qui se calculent rapidement.
    - En utilisant une machine autonome sans connectivité réseau, personne ne peut interfacer avec sa programmation et manipuler les résultats.
  - Cependant, ce n'est pas sûr de prévenir le vote multiple par une seule personne.

- En plus de ça, l'utilisation du personnels dans le processus prennent beaucoup de temps et augmentent les coûts pour effectuer les tâches suivantes :
  - La vérification d'identité qui s'effectuée manuellement,
  - Signature de l'électeur à la fin du vote,
  - Le cachet de la carte de vote à la fin,
  - La récupération de la liste électorale

- Le deuxième type de système de vote électronique se fait à distance via des réseaux informatiques et Internet. Habituellement, les électeurs ont la possibilité de voter en utilisant des ordinateurs aux niveaux des bureaux de vote, à des endroits éloignés, ou bien, leurs téléphones portables.
  Les systèmes de vote électronique à distance ont l'avantage du contrôle de l'identité qui se fait très rapidement avec différentes techniques, selon le système utilisé, avec une authentification en utilisant une carte à puce d'identité, authentification biométrique ou bien une authentification dynamique englobant plusieurs techniques à la fois. Malgré cela, il y'a beaucoup de risques de sécurité car le vote se déroule par internet.

5.5.2. Méthodologie

Notre contribution consiste à fusionner les deux systèmes de vote électronique existants afin de concevoir un nouveau qui prend en compte les avantages de chacun. Nous prenons les avantages des systèmes de vote de bureau avec la rapidité de contrôle de l'identité assuré par le systèmes de vote électronique à distance.

L'authentification des électeurs qui se faisaient manuellement est éliminée et remplacée par l'utilisation d'une carte à puce associé à l'empreinte digitale du porteur de la carte. Les données échangées sont crypté avec l'algorithme cryptographique RSA pour plus de sécurité.

5.5.3. Description du système à concevoir

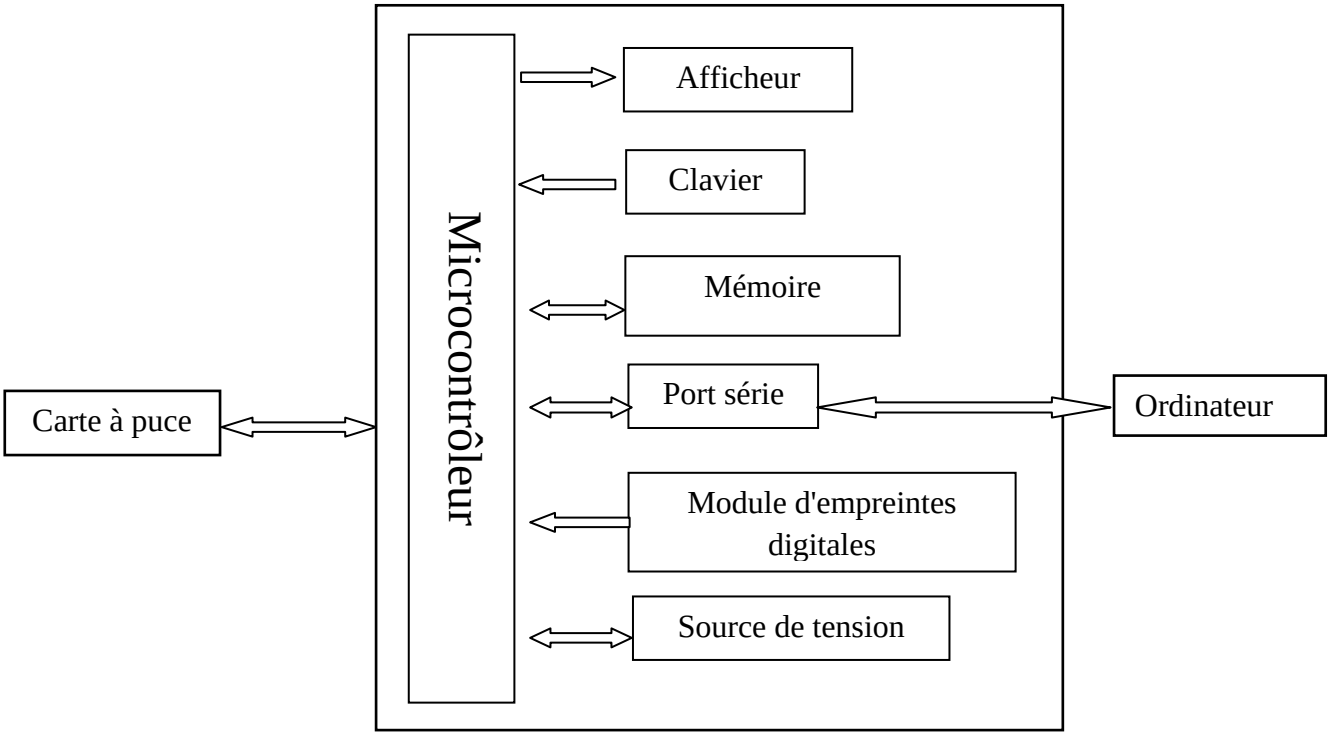


Fig.5. 22: bloc diagramme de notre machine de vote électronique embarquée.

La machine vote électronique est composée d'un microcontrôleur qui représente le composant principal qui assure la communication avec la carte à puce (voir figure5.22), et contient une mémoire pour enregistrer la liste des électeurs. La machine communique avec l'utilisateur à l'aide d'un clavier, un module d'empreinte digitale et un afficheur. A la fin du vote, la machine envoie les résultats et la liste électorale à l'ordinateur via le port série. La machine a été faite pour fonctionner sur piles pour maintenir l'alimentation dans de nombreux endroits.

5.5.4. Déroulement du vote

Cette machine de vote est présente dans les bureaux de vote initialisée avec les numéros de série correspondant aux électeurs. Sachant que, la base de données correspondant à l'identité des porteurs des cartes de vote est séparée de ce que les numéros de série. Seule l'application au niveau de l'ordinateur peut faire la correspondance. Elle est munie d'un clavier et un affichage pour communiquer avec l'utilisateur et un port série pour transmettre les résultats du vote à l'ordinateur.

Chaque électeur doit voter dans son bureau de vote parce que les machines de vote électronique ne sont pas connectées au réseau cela, afin d'éviter toute manipulation.

L'administrateur démarre la machine après l'authentification à l'aide de sa carte à puce et son empreinte digitale. Les électeurs viennent au bureau de vote munis de leurs cartes à puce, ils sont identifiés par la machine en insérant leur carte à puce de vote et en utilisant leurs empreintes digitales.

Une fois l'identité vérifiée, la machine vérifie que l'électeur n'a pas déjà voté. Ensuite, une liste de candidats apparaît avec les chiffres. L'électeur choisit un candidat et valide son choix. La machine enregistre le vote dans la machine et dans la carte à puce ensuite elle l'éjecte. L'électeur peut quitter le bureau de vote.

A la fin du vote et après l'authentification, l'administrateur affiche les résultats pour les personnes présentes. Ensuite, la machine est connectée à un ordinateur pour récupérer les résultats des bureaux de vote. Il n'y a que l'administrateur qui peut accéder à l'application au niveau de l'ordinateur pour plus de sécurité. Il publie les résultats du vote et la liste électorale sur le site web du vote pour donner plus confiance aux résultats.

5.5.5. Programmation du système

Lorsque la machine est en marche, elle attend la réception d'une carte à puce. Une fois que la carte est insérée, la machine vérifie que la bonne carte et qu'elle n'est pas bloquée et déterminé si elle est une carte administrateur ou électeur. Dans les deux cas, la machine authentifie l'utilisateur en utilisant une carte à puce et l'empreinte digitale pour accéder au système.

Dans le cas d'une carte à puce électeur (voir la figure 5.23), la machine contrôle si l'électeur a déjà voté ou non. Ensuite, il choisit un candidat en utilisant le clavier et valide son vote. Une fois que le vote est confirmé, la machine incrémente le nombre d'électeurs et celui de voix correspondant au candidat sélectionné dans sa mémoire interne. La machine enregistre le numéro d'identification de la carte dans sa mémoire et le vote dans la carte à puce.

Pour une carte administrateur, la machine affiche les résultats de l'élection. Voir l'organigramme de la figure 5.24.

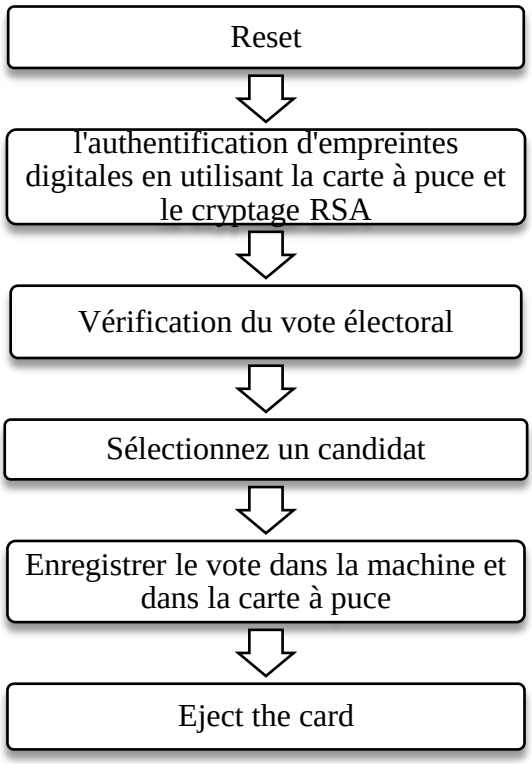


Fig.5. 23: Organigramme de la machine de vote électronique communiquant avec la carte à puce électeur.

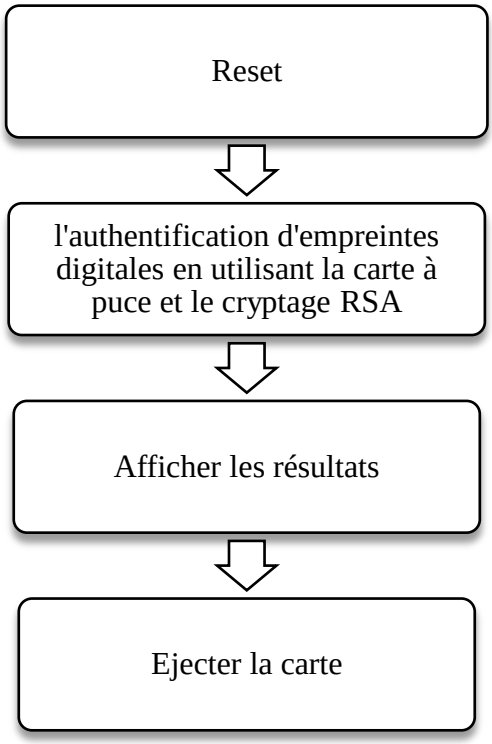


Fig.5. 24: Organigramme de la machine de vote électronique communiquant avec la carte à puce administrateur.

Lors de la connexion à l'ordinateur, comme le montre la figure 5.25, la machine de vote électronique doit être authentifié par l'ordinateur pour être sûr que ce soit la bonne machine. Après cela, l'ordinateur récupère les résultats et décoder la liste électorale et publie tout cas dans un site Web pour prouver aux gens que le vote a été fait sans fraudes.

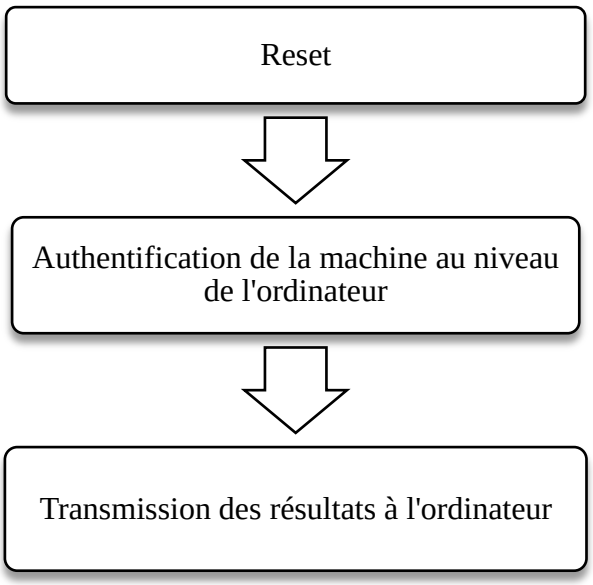


Fig.5. 25: Organigramme de la machine de vote électronique communiquant avec l'ordinateur.

5.5.6. Simulation du système

Le circuit de simulation est divisé en deux parties : la carte à puce et la machine de vote électronique. Voir la figure 5.26.

La carte à puce utilisée se compose d'un microcontrôleur PIC 18F452 qui assure le dialogue avec la machine vote électronique et la mémoire EEPROM 24LC16 pour stocker des données. Elle communique avec la machine de vote électronique avec un protocole série asynchrone RS232 avec 9600 bauds. La carte à puce est simulée avec des composants discrets mais en réalité c'est un seul circuit intégré. Toute carte à puce qui prend en charge le cryptage RSA peut être utilisé pour cette application.

La machine de vote électronique contient également un microcontrôleur PIC 18F452 qui fonctionne avec une fréquence d'horloge de 40 MHz, il représente le composant principal qui assure la communication avec la carte à puce, et contient une mémoire EEPROM 24LC64 de 8 K octets pour enregistrer la liste électorale (8192 électeurs). La machine communique avec l'utilisateur à l'aide d'un clavier 12 touches et un écran LCD avec deux lignes de vingt caractères.

A la fin du processus de vote, le microcontrôleur de la machine de vote électronique peut être connecté à un ordinateur à l'aide d'un port série.

La simulation est faite sous le logiciel PROTEUS et la programmation des microcontrôleurs sous MPLAB. Un terminal virtuel est utilisé pour afficher les données échangées entre la carte et la machine, comme le montre le circuit de simulation.

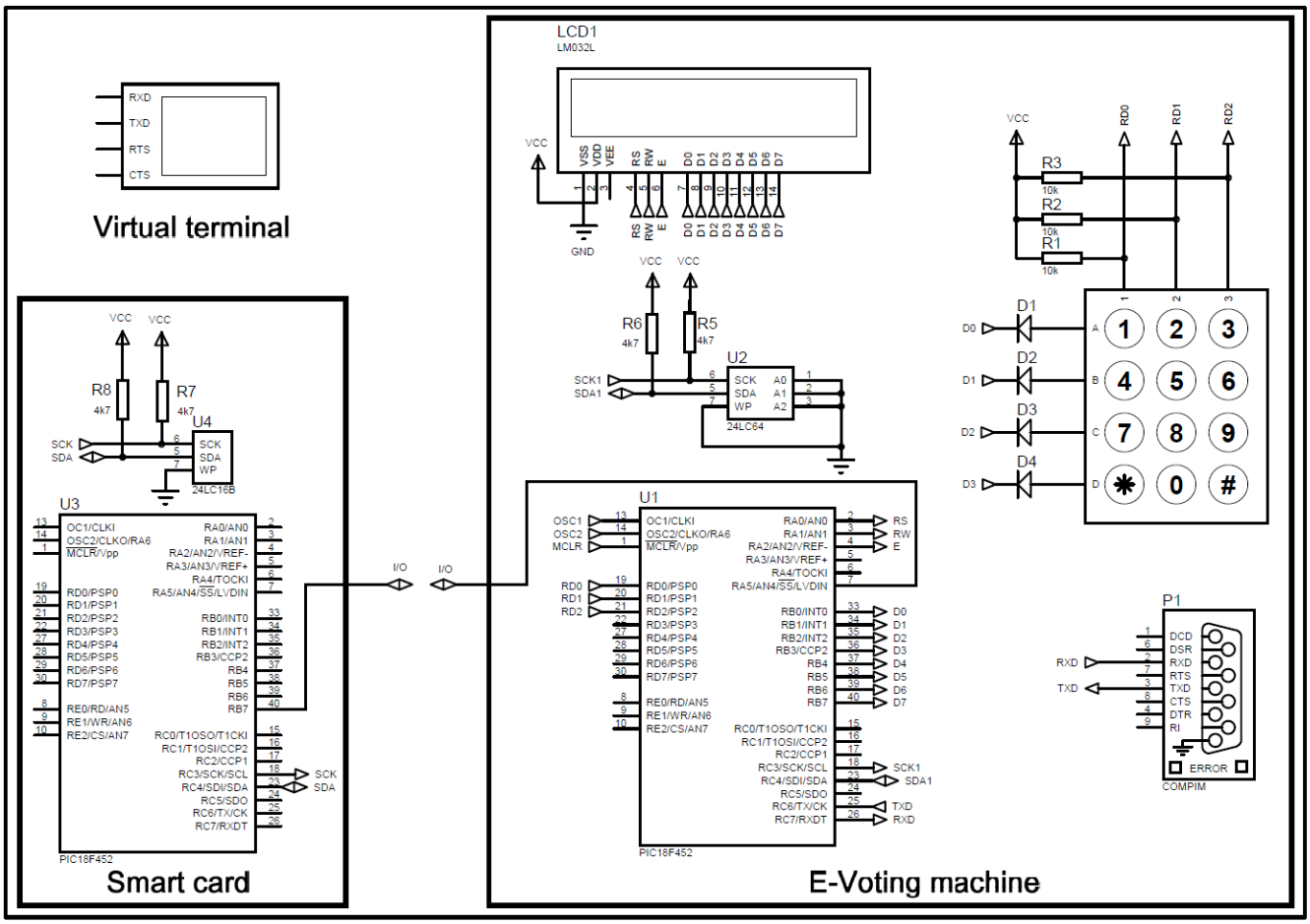


Fig.5. 26: Circuit de simulation du nouveau système de vote électronique.

Ce système a été testé pour six candidats. L'électeur entre le numéro correspondant à son candidat choisi à l'aide du clavier (numéro de candidat : 1, 2, 3, 4, 5 ou 6), ou bien «0» pour un bulletin nul. Afin d'éviter la touche accidentelle du clavier, la machine demande à l'électeur de confirmer son choix en utilisant le bouton "*" ou annuler avec le bouton "#". Une fois que le vote est confirmé, la machine incrémente le nombre d'électeurs et le vote pour le candidat choisi dans l'EEPROM interne du microcontrôleur PIC 18F452 de la machine. Cette dernière enregistre également le numéro de la carte dans sa mémoire EEPROM 24LC64 et enregistre le vote sur la carte à puce (dans la mémoire EEPROM).

Pour une authentification administrateur, la machine affiche les résultats de l'élection sur l'écran LCD.

5.5.7. Résultats et discussion

Nous avons fusionné les deux systèmes de vote électroniques existant (système de vote dans les bureaux, vote à distance), en prenant en compte les avantages de chacun d'entre eux. Nous avons conçu une machine de vote électronique embarquée, à base d'un microcontrôleur et utilise une carte à puce qu'on installe dans des bureaux de vote :

Les avantages prient des systèmes de vote électronique de bureau sont :

- Résultats calculés rapidement.
- Étant une machine autonome sans connectivité réseau, personne ne peut interfacer avec sa programmation et manipuler les résultats.

- A la fin, la machine pourras être relié à un ordinateur avec une authentification pour permettre l'affichage des résultats et le calcule d'autres statistiques du vote.

Les avantages assurés par le système de vote électronique à distance sont :

- L'authentification de l'électeur qui se faisait manuellement est éliminée et remplacé par une authentification avec une carte à puce et empreinte digitale de son titulaire, ce qui rond le contrôle d'identité rapide.
- Remplacé le cachet de la carte à la fin du vote.
- Eviter l'utilisation de personnels dans le processus.

En plus des avantages prient des deux différents systèmes de vote électronique, nous avons amélioré le système en résolvons plusieurs problèmes :

L'anonymat : Ce système permis d'assurer l'anonymat par une authentification avec une carte à puce destinée au vote, au lieu de la carte d'identité utilisée dans les systèmes de vote électronique à distance, associée à une authentification biométrique à l'aide de l'empreinte digitale de l'électeur pour plus de sécurité et de rapidité.

La sureté des résultats :

- Les résultats du vote seront enregistrés dans la mémoire EEPROM interne du Microcontrôleur de la machine qui sera protégée en écriture.
- Le calcule automatique des résultats nécessite l'utilisation d'une authentification avec une carte à puce administrative et un mot de passe pour affiché les résultats du vote.
- À la fin du vote, la machine sera authentifiée par l'ordinateur avant de recevoir les résultats.
- Pour donner plus de confiance, les résultats de l'élection et la liste électorale qui correspond au nombre total d'électeurs seront publiées sur internet.

Le vote multiple : Enregistrement du vote dans la carte à puce pour éviter le vote plusieurs fois par la même personne.

Récupération de la liste électorale : Les numéros de série des cartes de vote seront enregistrés dans la machine du vote électronique pour les autorités administratives, pour récupérer la liste des électeurs, ce qui remplace la signature de l'électeur à la fin du vote. Sachant que la base de données des numéros de séries des cartes à puce du vote est séparée des informations de leurs titulaires, après les élections les deux bases peuvent être reliées afin de récupérer la liste électorale.

5.6. Conclusion

La carte à puce est de plus en plus utilisée dans de nombreux domaines notamment celui du vote électronique. Au cours de ce travail, nous avons exploité cette application dans le but de l'améliorer tout en optimisant les ressources utilisées.

Nous avons vu qu'il existe deux types de systèmes de vote électronique, le premier est basé sur la visite d'un bureau de vote et le deuxième se fait à distance. Chacun d'eux a des avantages et des inconvénients.

La première partie de la contribution était l'amélioration du système de vote électronique qui se fait dans les bureaux. Le système obtenu a permis de résoudre quelques problèmes liés à ce type de vote. Ceci en utilisant une carte à puce dans les bureaux de vote.

La deuxième partie consistait à fusionner les deux systèmes de vote électroniques existant, en prenant en compte les avantages de chacun d'entre eux. Nous avons conçu une machine de vote électronique embarquée, à base d'un microcontrôleur et utilisant une carte à puce, qu'on installe dans

des bureaux de vote. En plus des avantages prient des deux différents systèmes de vote électronique, nous avons amélioré le système en résolvons plusieurs problèmes.

La programmation des protocoles de communication en langage assembleur a permis de choisir des cartes à puce ainsi que des microcontrôleurs de la machine ; pas chère et suffisants pour l'application.

Conclusion générale

Depuis son apparition, la carte à puce ne cesse de conquérir de nouveaux marchés tout en gagnant la confiance du consommateur. Elle est considérée aujourd'hui comme un élément essentiel et incontournable de la vie de tous les jours. Ceci est dû non seulement au fait que l'on puisse la transporter, mais aussi au niveau de sécurité élevé qu'elle assure.

L'évolution de la technologie a permis la conception de cartes à puce de plus en plus performantes et sécurisées ainsi, l'implémentation d'applications est devenue plus rapide mais leurs coûts augmentent aussi. Le choix de la carte à puce qui convient le mieux à une application donnée est devenu une tâche difficile, en particulier pour celles destinées à un grand nombre d'utilisateurs. Il faut réduire les coûts tout en assurant le bon fonctionnement. Il serait donc crucial pour les développeurs d'applications de s'interroger sur les performances des cartes à puce utilisées tout en réduisant les couts.

Les protocoles de communication représentent la base de n'importe quelle application faite avec carte à puce. Dans le but l'optimiser les ressources utilisées, nous avons programmé et simulé les protocoles les plus connus des cartes à puce (RS 232, I²C et T=0) en langage assembleur.

Cela a été appliqué pour deux applications de la carte à puce : la carte à puce destinée à l'étudiant et le système de vote électronique. Nous avons obtenu des systèmes pas cher et optimisés en ressource matériel et en espace mémoire tout en assurant le bon fonctionnement. Par la suite, nous avons amélioré chaque application à part en répondant aux besoins.

Nous avons commencé par la carte à puce étudiant en concevant et réalisant une carte à puce ainsi que son lecteur et interface graphique. L'application était destinée pour notre faculté (faculté des sciences, université M'hamed Bougara de Boumerdes, Algérie). Le système conçu facilitera énormément de travail de l'administration :

- Elle sert à garder les informations de son titulaire ainsi que ses relevés de notes durant tout son parcours universitaire.
- L'accès se fait à l'aide d'un mot de passe avec un nombre d'essai limité.
- Le saisi de données se fait uniquement par l'administrateur au niveau de la faculté,
- L'étudiant peut consulter ses données mais ne peut apporter aucune modification.

Au début de chaque année universitaire, et à la fin de chaque semestre l'étudiant doit se présenter au niveau de la scolarité de la faculté afin de réactiver sa carte et obtenir ses notes. Cela facilitera énormément le travail de l'administration.

Concernant les systèmes de vote électroniques, nous avons commencé par l'amélioration du système qui se fait dans les bureaux de vote. Le système obtenu a permis de résoudre quelques problèmes liés à ce type de vote, ce qui a rendu le système :

- Rapide en manipulation ;
- Préviennent le vote multiple ;
- Sécurisé avec l'authentification des utilisateurs.

Tout ceci en utilisant une carte à puce dans les bureaux de vote. En plus de cas :

Le système est flexible en utilisant un clavier permettant d'avoir un nombre important de candidats (non limité par quelques boutons poussoirs). La récupération de la liste électorale est assurée par l'enregistrement des numéros de série des cartes à puce électeur ayant voté dans la machine.

Afin d'amélioré ce dernier, nous avons fusionné les deux systèmes de vote électroniques existant (système de vote dans les bureaux, vote à distance), en prenant en compte les avantages de chacun d'entre eux. Nous avons conçu une machine de vote électronique embarquée, à base d'un microcontrôleur et utilise une carte à puce qu'on installe dans des bureaux de vote :

Les avantages prient des systèmes de vote électronique de bureau sont :

- Résultats calculés rapidement.
- Étant une machine autonome sans connectivité réseau, personne ne peut interfacer avec sa programmation et manipuler les résultats.
- A la fin, la machine pourras être relié à un ordinateur avec une authentification pour permettre l'affichage des résultats et le calcule d'autres statistiques du vote.

Les avantages assurés par le système de vote électronique à distance sont :

- L'authentification de l'électeur qui se faisait manuellement est éliminée et remplacé par une authentification avec une carte à puce et empreinte digitale de son titulaire, ce qui rond le contrôle d'identité rapide.
- Remplacé le cachet de la carte à la fin du vote.
- Eviter l'utilisation de personnels dans le processus.

En plus des avantages prient des deux différents systèmes de vote électronique, nous avons amélioré le système en résolvons plusieurs problèmes :

- Assurer l'anonymat par une authentification avec une carte à puce destinée au vote ;
- sureté des résultats ;
- Éviter le vote plusieurs fois par la même personne ;
- Récupération de la liste électorale.

L'utilisation du vote électronique est un vrai sujet de débat, parce que c'est vraiment difficile de concevoir un système de vote idéal qui assure une parfaire sécurité et l'anonymat sans compromis. Pour cette raison, ces systèmes ne sont pas généralisés. Nous comptons réaliser et tester ces systèmes dans des situations réelles et essayer de les améliorer afin d'avoir un meilleur système de vote électronique basé sur carte à puce.

Productions Scientifiques de l'auteur

Communications dans des actes de congrès avec comité de lecture

1. Karima DICHOU, Victor TOURTCHINE and Fayçal RAHMOUNE: "An Improved Electronic Voting Machine Using a Microcontroller and a Smart Card," IEEE. 9th International Design & Test Symposium (IDT). Algeria, pp. 219-224, December 2014.

2. Karima DICHOU, Victor TOURTCHINE and Fayçal RAHMOUNE: "Conception et Réalisation d'une Carte à Puce Etudiant," 9th International Telecom'15 & 9ème JFMMA Symposium. Morocco, pp. 1-4, May 2015.

3. Karima DICHOU, Victor TOURTCHINE and Fayçal RAHMOUNE: "Finding the best FPGA implementation of the DES algorithm to secure smart cards," IEEE. 4th International Conference on Electrical Engineering (ICEE). Algeria, December 2015.

4. Karima DICHOU, Victor TOURTCHINE and Fayçal RAHMOUNE: "Simulation of APDUs Exchanged Between a Microcontroller Smart Card and a Reader," IEEE. 7th International Conference on Modeling, Identification and Control (ICMIC). Tunisia, December 2015.

Publications dans des revues avec comité de lecture

1. Karima DICHOU, Victor TOURTCHINE and Fayçal RAHMOUNE: "Simulation of smart card interface with electronic voting machine," Computer Science and Applications (CSA), vol. 2, no 4, pp. 140-146, July 2015.

2. Karima DICHOU, Victor TOURTCHINE and Fayçal RAHMOUNE: "An embedded e-voting machine with smart card", Int. J. Signal and Imaging Systems Engineering (Inderscience), 2016, in press.

Références

[1] P. Gueule, Cartes à puce, 2ème édition DUNOD. Paris. 2001.

[2] C. Tavernier, Les cartes à puce –Théorie et mise en œuvre, 2ème édition DUNOD. Paris. 2002.

[3] U. Hansmann, M.S. Nicklous, T. Schäck, A. Schneider, and F. Seliger, Smart card application development using Java. Springer Verlag, Berlim, 2002.

[4] M. Hendry, Multi-application smart cards, technology and applications. Cambridge university press, 2007.

[5] W. Rankl, W. Effing, Smart Card Handbook, 4th ed., John Wiley & Sons, New York,2010.

[6] K. Markantonakis, K. Mayes, Secure smart embedded devices, platforms and applications. Springer, University of London, 2014.

[7] M. Hendry, Smart card security and applications, 2nd ed., Artech House, London, 2001.

[8] K. Markantonakis and K.E. Mayes, Smart cards, tokens, security and applications. Springer Science & Business Media, University of London, 2007.

[9] P. Gueule, PC et cartes à puce, DUNOD, Paris. 2000.

[10] D. Sauveron. Étude et réalisation d'un environnement d'expérimentation et de modélisation pour la technologie Java CardTM. Application à la sécurité. Thèse de doctorat, Université Bordeaux I, 3 décembre 2004.

[11] J. Katz and Y. Lindell, Introduction to modern cryptography, 2nd ed., CRC Press, Taylor & Francis Group, 2014.

[12] B. Schneider, Applied Cryptography: Protocols, algorithms, and source code in C, 2nd ed., John Wiley & Sons, 1996.

[13] F. Rodríguez-Henríquez, N.A. Saqib, A.D. Perez, and Ç.K. Koç, Cryptographic algorithms on reconfigurable hardware. Springer Science & Business Media, United States of America, 2007.

[14] C. BARRAL. Biometrics & Security: Combining Fingerprints, Smart Cards and Cryptography. Thèse de doctorat, Faculté Informatique Et Communications. Laboratoire De Sécurité et De Cryptographie, Suisse, 25 Août 2010.

[15] A. Karray. Conception, mise en oeuvre et validation d'un environnement logiciel pour le calcul sécurisé sur une grille de cartes à puce de type Java. Thèse de doctorat, Université Bordeaux I, 10 décembre 2008.

[16] C. Clavier. De la sécurité physique des crypto-systèmes embarqués. Thèse se doctorat, Université de Versailles Saint-Quentin de Paris, 23 novembre 2007.

[17] E. Trichina, M. Bucci, D. D. Seta and R. Luzzi, Supplemental Cryptographic Hardware for Smart Cards, IEEE Micro, vol.21, No.6, November/ December 2001.

[18] Z. Peng and J.J. Fang, "Comparing and Implementation of Public Key Cryptography Algorithms on smart card", IEEE, International Conference on Computer Application and System Modeling (ICCASM 2010), pp. 508-510, 2010.

[19] K.M.A. Abd Elatif, H.F.A. Hamed and E.A.M. Hasaneen, "FPGA Implementation of the pipelined data encryption standard (DES) based on variable time data permutation," the Online Journal on Electronics and Electrical Engineering (OJEEE), vol. 2, no 3, pp. 298-302, July 2010.

[20] V. Patel, R. C. Joshi, A. K. Saxena, "FPGA Implementation of DES Using Pipelining Concept With Skew Core Key-Scheduling," Journal of Theoretical and Applied Information Technology, Vol 5, No3, pp. 295-300, March 2009.

[21] J. Leonard, W.H. Mangione-Smith, "A case study of partally evaluated hardware circuits: key-specific DES," Field-Programmable Logic and Applications. Springer Berlin Heidelberg, pp. 151-160, 1997.

[22] K. Wong, M.Wark, E. Dawson, “A single-chip FPGA implementation of the data encryption standard (DES) algorithm,” IEEE Globecom Communication, pp. 827- 832, vol.2, Sydney, Australia, 1998.
[23] J.P. Kaps and C. Paar, “Fast DES Implementations for FPGAs and Its Application to a Universal Key-Search Machine,” 5th Annual Workshop on selected areas in cryptography, pp.234-247, Canada, 1998.
[24] C. Patterson, “High Performance DES Encryption in Virtex FPGAs Using Jbits, “Field-Programmable Custom Computing Machines, FCCM’00, pp. 113-121, USA, 2000.
[25] M. Mcloone and J. V. McCanny, “A high performance FPGA implementation of DES,” IEEE. Workshop Signal Processing Systems, (SiPS). Lafayette, LA, pp. 374-383, October 2000.
[26] S. Vajpayee and R. Rai, “Modeling and simulation of data encryption standard algorithm on FPGA,”International Journal of Computer Technology & Applications, Vol. 3, no 3, pp. 1015-1022, June 2012.
[27] M. McLonne and J.V. Mccanny, “High-performance FPGA implementation of DES using a noval method for implementing the key schedule,” IEE Proceedings of Circuits, Devices and Systems (IET), pp. 373-378, 2003.
[28] Filippo, ‘Fingerprint traits and RSA algorithm fusion technique’, IEEE, Sixth International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), pp. 351-356, 2012.
[29] Microchip, 64K I2C™ Serial EEPROM, Datasheet,2002.
[30] Microchip, “24AA64:24LC64, 64K I2C Serial EEPROM”,2002.
[31] Microchip, " 18-pin Enhanced FLASH/EEPROM, 8-bit Microcontroller,” 16F84 datasheet,2001.
[32] P. Gueule, Plus loin avec les cartes à puce» DUNOD. Paris. 2004.
[33] E. Mesnard, Du binaire au processeur. Ellipses. Paris, 2004,.
[34] C. Tavernier, Les microcontrôleurs PIC -description et mise en œuvre, 2ème édition DUNOD. Paris. 2002.
[35] E. Yann and L. Anger, Robots mobiles intelligents Du capteur au comportement». France. 2006.
[36] P. Gueule, Composants électroniques programmables sur PC, 3ème édition DUNOD. Paris. 2005.
[37] C. Tavernier, Electronique pratique – Lecteur pour carte à puce Wafer Gold, Silver, Fun, etc., n°267.
[38] S. Omar and H. Djuhari, “Multi-purpose student card system using smart card technology.” IEEE. the Fifth International conference on Information technology based higher education and training (ITHET), pp. 527-532, 2004.
[39] Lv. Chao, J. Zhang and Y. Ma, Student attendance management system based on campus smart card platform. Applied mechanics and materials, vol. 321-324, Pp. 3022-3025, 2013.
[40] A. Ansari, R. Joshi, A. Katariya and R. Gaikwad, Single multipurpose and resourceful techno card. International journal of engineering technology science and research (IJETSR), vol. 3, Issue 2, pp. 8-13, 2016.
[41] M. J. Voon, S. M. Yeo and N. H. Voon, “Campus access control and management system, Intelligent and evolutionary systems”, Proceedings in adaptation, Learning and optimization 5, pp. 395-404, 2016.
[42] A. Al-Ameen, S.A. Talab, The technical feasibility and security of e-voting, International Arabic Journal of Information Technology , vol. 10, Issue 4, pp. 397-404, 2013.
[43] S.M. Ali, C.A. Mehmood, A. Khawja, R.Nasim, M.Jawad and S.Usman, “Micro-controller based smart electronic voting machine system,” IEEE. International Conference on Electro/Information Technology (EIT), Milwaukee, WI, pp. 438-442, Jun. 2014.

[44] D.A. Kumar, T.U.S. Begum, Electronic voting machine—A review, in: International Conference on Pattern Recognition, Informatics and Medical Engineering (PRIME), Salem, Tamilnadu, pp. 41-48, Mar 2012.
[45] S.V. Prasath, R. Mekala M.E., A literature survey on micro-controller based smart electronic voting machine system, International Journal of Advanced Research in Electronics and Communication Engineering 3, vol. 12, pp. 1756-1761, 2014.
[46] S.M. Ali, C.A. Mehmood, A. Khawja, R.Nasim, M.Jawad, S.Usman, et al., "Micro-controller based smart electronic voting machine system," IEEE. International Conference on Electro/Information Technology (EIT), Milwaukee, WI, pp. 438-442, Jun. 2014.
[47] S. Agrawal, P. Majhi and V. Yadav, Fingerprint recognition based electronic voting machine, International Journal of Engineering and Technical Research (IJETR), p. 255-259, 2014.
[48] K.Jagriti, P. Sabi, R. Arthi, M. Prawin Angel, Electronic voting machine using ZIGBEE, International Journal of Research in Engineering and Technology, vol. 3, issue 7, pp. 11-16, 2014.
[49] Bhatia, Vaibhav and Gupta, Rahul, Design of a GSM Based Electronic Voting Machine with Voter Tracking. In: International journal of information technology, vol. 4852, p. 799-802, 2015.
[50] Bhatia, Vaibhav and Gupta, Rahul. A novel electronic voting machine design with voter information facility using microcontroller. In: Computing for Sustainable Global Development (INDIACom), International Conference on. IEEE, pp. 274-276, 2014.
[51] Anjum, B. Farhath. Advanced Microcontroller Based Bio-Metric Authentication Voting Machine. IOSR Journal of Engineering, vol. 4, no 5, pp. 29-40, 2014.
[52] Lalitha, A. and Sankar, N. Muthu, Design and Implementation of Ballot Malfunctioning System Avoidance Security Optimization. In: International Journal of science and research (IJSR), pp. 2018-2021, 2012.
[53] Hoque, Md Murshadul, A Simplified Electronic Voting Machine System .In: International Journal of Advanced Science & Technology, vol. 62, 2014.
[54] G. Z. Qadah and T. Rani, Electronic voting systems: Requirements, design, and implementation. Computer Standards & Interfaces, vol. 29, no 3, p. 376-386, 2007.
[55] Shukur, Ban Salman. Mobile Voting Prototype to Promote the Participation of All Students in Selecting Their Representatives: Iraqi Universities as a Case Study. In: International journal of advanced in computer science & its applications (IJCSIA), p.38-42, 2014.
[56] Lee, Yunho, Park, Sangjoon, Mambo, Masahiro and Towards, trustworthy e-voting using paper receipts. Computer Standards & Interfaces, vol. 32, no 5, p. 305-311, 2010.
[57] G. Perry, Visual Basic 6,CampusPress. Paris. 2005.
[58] D. I. Schneider, Computer Programming -Concepts and Visual Basic, Pearson custom
[59] Xilinx, URL: http://www.xilinx.com/ November 2015.
[60] Microchip, 28/40-pin High Performance, Enhanced FLASH Microcontrollers with 10 bit A/D (2006), 18F452 datasheet.