



N° d'ordre :

UNIVERSITE DE M'SILA
FACULTE DES MATHEMATIQUES ET DE
L'INFORMATIQUE

Département d'Informatique

MEMOIRE

Présenté pour l'obtention du diplôme de :

MASTER

Domaine : Mathématiques et Informatique

Filière : Informatique

Spécialité : Systèmes d'informations avancés

Par: Siham ZOUAOUI

SUJET

Implémentation et validation de la pile TCP/IP de Microchip sur un dsPIC

Soutenu publiquement le : 28/06/2012 devant le jury composé de :

Mr. Adel MOUSSAOUI

Université de M'sila

Président

Mr. Marouane BENARROUDJ

Université de M'sila

Rapporteur

Mr. Foudil BELHADJ

Université de M'sila

Rapporteur

Mr. Aimad eddine DEBBI

Université de M'sila

Examineur

Promotion : 2011 /2012

بِسْمِ الرَّحْمَنِ الرَّحِيمِ



Remerciements

Je remercie tout d'abord Allah pour m'avoir donnée le courage et la santé pour accomplir ce travail.

Je tiens à témoigner ma reconnaissance à mon encadreur Monsieur Marouane BENARROUDJ, Maître assistante à l'université de M'sila, pour tous ses conseils précieux, ses critiques constructives et ses encouragements. Qu'il trouve ici l'expression de ma profonde gratitude.

Mes sincères remerciements vont à Monsieur Foudil BELHADJ Maître assistante à l'université de M'sila, Pour avoir accepté de suivre notre projet, Je vous remercie pour vos conseils utiles, qui ont grandement contribué à ma formation.

Je remercie aussi les membres du jury qui me font honneur en jugeant ce travail.

Je tiens à remercier tous mes enseignants de la deuxième année de Master, ainsi que tous mes collègues, et tout le personnel du département Informatique.

Un grand merci également à tous ceux qui de près ou de loin, m'ont apporté un support quelconque durant l'achèvement de ce projet. Qu'ils trouvent tous ici l'expression de notre gratitude et notre parfaite considération spécialement MR. Roussafi MAHJOUBI et Mlle Amel MELIOUH.

Dédicaces

À mes chers parents, pour leur amour et leurs sacrifices,

À mes chers frères Adel ,Abed elhamid & Mohamed

À mes chères sœurs Ahlam & Soundous,

À toute ma famille,

À tous mes amis,

À toutes les personnes qui me sont chères,

Je dédie ce modeste travail.

....  Siham

Résumé

Notre travail porte sur l'implémentation et la validation sur une cible matérielle de la pile TCP/IP fournie en projet source ouvert par la société Microchip. Cette pile qui est disponible, très documentée et contient tous les protocoles utiles et nécessaires pour l'élaboration d'un projet de système embarqué interagissant avec le réseau informatique. Nous avons embarqué cette pile sur une plateforme matérielle autour du circuit microcontrôleur dsPIC30F6014A et le circuit dédié au réseau ENC28J60. Pour la validation du fonctionnement de cette pile nous l'avons exploité pour développer un système de contrôle d'accès basé sur le réseau informatique.

Mots-clés :

la Pile TCP/IP de Microchip, dsPIC30F6014A, ENC28J60, MPLAB IDE, MPLAB C30, Proteus, MPLAB ICD2, PC, dsPICPRO4, contrôle d'accès, conception UML.

Abstract

Our work focuses on the implementation and validation of Microchip TCP / IP stack on a material target. This stack is open source project provided by Microchip Company. It is available, free of use helpful and contains all the useful and necessary protocols for the design of embedded system board interacting with the computer network. We embarked this stack hardware platform around dsPIC30F6014A microcontroller dedicated network circuit ENC28J60. To validate the operation of this stack, we used it to develop a control access system based on the computer network.

Keywords: Microchip TCP/IP Stack, dsPIC30F6014A, ENC28J60, MPLAB IDE, MPLAB C30, Proteus, MPLAB ICD2, PC, dsPICPRO4, control access systems, UML design.

Sommaire

Introduction générale	1
Chapitre I : Les systèmes embarqués	
1. Introduction.....	4
2. Définition d'un système embarqué.....	4
3. Caractéristiques d'un système embarqué.....	5
4. Architecture générale et les composants possibles d'un système embarqué.....	6
5. Développement de systèmes embarqués.....	8
6. Conclusion.....	8
Chapitre II : Généralités sur le modèle OSI et la pile TCP/IP	
1. Introduction.....	10
2. le modèle architectural OSI.....	10
2.1. Les sept couches du modèle OSI.....	10
3. La pile de protocoles TCP/IP	14
3.1. Bref historique.....	14
3.2. Le modèle TCP/IP.....	16
3.3. Caractéristique de la pile TCP/IP.....	16
3.4. Comparaison entre le modèle OSI et la pile TCP/IP.....	17
3.4.1. Couche application.....	17
3.4.2. Couche transport.....	17
3.4.3. Couche Internet	18
3.4.4. Couche accès réseau.....	18
4. Les différents protocoles TCP/IP.....	18
4.1. Format d'une trame Ethernet MAC 802.3 10-Base T.....	19
4.2. Le protocole ARP.....	20
4.2.1. Format du datagramme ARP.....	20
4.3. Le protocole IP.....	21
4.3.1. Format du datagramme IP.....	21
4.4. Le protocole ICMP.....	23
4.4.1. Format du datagramme ICMP.....	24
4.5. Le protocole UDP.....	25
4.5.1. Format du datagramme UDP.....	25

4.6. Le protocole TCP.....	26
4.7. Le protocole HTTP.....	28
4.7.1. Communication entre navigateur et serveur.....	29
5. Etude de cas du la pile TCPIP de Microchip.....	30
5.1. Présentation de la pile TCP/IP de Microchip.....	30
5.2. L'architecture de la pile TCP/IP de Microchip.....	31
5.3. Configuration de la pile TCP/IP de Microchip.....	32
5.4. Fonctionnement de la pile TCP/IP de Microchip.....	34
5.4.1. Initialisation de la pile	34
5.5. La boucle Main (super boucle).....	34
5.6. Le Multitâche coopératif.....	35
6. Conclusion.....	36

Chapitre III : Etude de l'environnement matériel

1. Introduction.....	38
2. Présentation de l'environnement matériel.....	38
2.1. Description de la carte de développement dsPICPRO4.....	38
2.2. Les Microcontrôleur	40
2.2.1. Définition d'un PIC.....	41
2.2.2. Définition d'un dsPIC.....	41
2.3. Interface Série pour périphérique.....	41
2.3.1. Fonctionnement de protocole SPI.....	42
3. Programmation de la carte dsPICPRO4.....	43
3.1. Moyens et outils utilisés	43
3.1.1. Environnement de développement.....	43
3.1.2. Le compilateur MPLAB C30.....	45
3.1.3. L'outil de simulation Protus.....	46
3.2. Configuration de la pile TCP/IP de Microchip.....	46
3.2.1. Configuration de la pile à l'aide du fichier HardwareProfile.h.....	47
3.2.2. Configuration de la pile à l'aide du fichier TCPIPConfig.h.....	47
3.2.3. Programmation du fichier BerkeleyTCPCClientDemo.c.....	49
3.2.4. Le fichier MainDemo.c.....	49
3.2.4.1. Algorithme de maindemo.c.....	49
4. Conclusion.....	50

Chapitre IV : Conception et réalisation d'un système de contrôle d'accès

1. Introduction.....	52
2. Description du système de contrôle d'accès mise en œuvre.....	52
2.1. protocole de communication utilisé.....	53
3. Environnement de travail.....	54
4. Conception du système.....	54
4.1. Identification des acteurs.....	55
4.2. Diagramme de cas d'utilisation.....	55
4.3. Diagramme de séquence.....	56
4.4. Diagramme d'état –transition.....	61
Conclusion.....	64

Liste des figures

Fig.1: Architecture d'un système embarqué	7
Fig.2 : Modèle en 7 couches de l'OSI.....	11
Fig.3 : Le modèle OSI et l'architecture TCP/IP.....	17
Fig.4: Trame Ethernet 802.3.....	19
Fig.5: Datagramme ARP.....	20
Fig.6 : Datagramme IP.....	22
Fig.7 : Message ICMP.....	24
Fig.8 : Datagramme ICMP.....	24
Fig.9 : Datagramme UDP.....	25
Fig.10 : Datagramme du pré en-tête UDP.....	26
Fig.11 : Datagramme de l'en-tête TCP.....	27
Fig.12: Datagramme du pré en-tête TCP.....	28
Fig.13: Communication entre client et serveur	29
Fig.14 : Les protocoles supportés par la pile TCP/IP Microchip.....	31
Fig.15 : La pile TCP/IP de Microchip vs modèle de référence TCP/IP.....	31
Fig.16 : La carte dsPICPRO4.....	39
Fig.17 : Liaison SPI avec un maître et un esclave.....	42
Fig.18: Diagramme de circuit Ethernet.....	43
Fig.19: les outils logiciels et matériels de MPLAB.....	44
Fig.20: Interface MPLAB V8.80.....	45
Fig.21 : Configuration du protocole utilisé par la pile TCP/IP.....	48
Fig.22 : l'architecture de système à réaliser.....	53
Fig.23 : Diagramme de cas d'utilisation.....	55
Fig.24 : Diagramme de séquence « ouvrir ».....	56
Fig.25 : Diagramme de séquence « fermer ».....	57
Fig.26 : Diagramme de séquence « vérifier l'état d'une porte ».....	58
Fig.27 : Diagramme de séquence «bloquer une porte(s) ».....	59
Fig.28 : Diagramme de séquence « accéder à une porte ».....	60
Fig.29 : Diagramme d'état transition « serveur ».....	61
Fig.30 : Diagramme d'état transition « client ».....	62
Fig.31: interface pour « système de contrôle 'accès ».....	63

Introduction Générale

Les systèmes embarqués, qui peuvent être vus comme des systèmes mélangeant électronique et informatique ayant un fonctionnement autonome, ont envahi notre vie au quotidien. Ce sont des dispositifs intelligents qui ont des puissances de calcul de plus en plus grandes, et dont les moyens de communication leur permettent d'interagir avec le monde environnant. Ces objets communicants, des plus petits « étiquettes électroniques dites TAG placées sur des objets » jusqu'aux plus gros « satellites », en passant par les téléphones portables, appareils électroménagers, consoles de jeux vidéo, automobiles, etc.... Ces objets sont le plus souvent soumis à des contraintes fortes, de cout, de fiabilité et de sécurité (dispositifs d'atterrissage automatique en avionique), de consommation énergétique (gestion de la batterie), de réponse en temps réel. Ils sont munis des capteurs, d'actionneurs, d'écrans tactiles et de dispositifs de communication reliés à un microcontrôleur ou un microprocesseur qui fonctionne avec un système d'exploitation temps réel.

L'interaction avec les systèmes embarqués est généralement obtenue en raccordant ces appareils à un réseau informatique. Les protocoles de la pile TCP/IP utilisés pour la communication à travers l'Internet sont relativement complexes et difficile à maîtriser.

L'objectif de ce travail est d'étudier, d'implémenter la pile TCP/IP de Microchip sur un dsPIC et de développer un système de contrôle d'accès simple, basé sur PC, pour but de validation.

Pour ce faire, nous avons divisé le travail en deux parties : la première partie consiste à configurer et embarquer la pile TCP/IP de Microchip sur un microcontrôleur dsPIC30F6014A en utilisant la carte de développement dsPICPRO4.

La deuxième partie est consacrée à la validation du fonctionnement de la pile sur un système simple de contrôle d'accès permettant le contrôle d'ouverture d'une porte par une carte sans contacts.

Ce mémoire est formé de quatre chapitres, à travers lesquels nous décrivons le travail effectué pour la conception et la réalisation de notre système:



Dans le premier chapitre, nous commençons par une définition des systèmes embarqués. Nous donnons les différentes caractéristiques qui paraissent importantes, et décrivons leur Architecture générale.

Dans le deuxième chapitre, nous donnons quelques généralités sur le modèle OSI et ce protocole TCP/IP, tout en présentant la pile TCP/IP de Microchip.

Dans le troisième chapitre, Nous focalisons notre travail sur la présentation de l'environnement matériel et la description de la carte de développement dsPICPRO4. En plus, les outils et les moyens utilisés pour la configuration de la pile TCP/IP de Microchip est donnée.

Le chapitre 4 est consacré aux tests et validation du bon fonctionnement de la pile. La conception et la mise en œuvre d'un système simple de contrôle d'accès est aussi entamée.

Finalement, nous clôturons ce mémoire par une conclusion et des perspectives.



Chapitre I

Les systèmes embarqués

1. Introduction :

Les systèmes embarqués sont d'une importance stratégique pour les économies modernes. Ils sont massivement utilisés dans des produits et services pour assurer de nouvelles fonctionnalités et de nouveaux usages et améliorer leur compétitivité.

Le logiciel embarqué prend une part de plus en plus importante par rapport au matériel.

Par exemple, grâce aux technologies embarquées, il est possible d'optimiser la consommation d'essence dans une voiture en fonction de l'état du moteur. Il est également possible d'améliorer la sécurité et le confort des passagers, avec des systèmes d'airbag plus fiables et performants, et avec le freinage assisté.

Ces systèmes peuvent être utilisés dans différents domaines, comme le transport (avionique, espace, automobile, ferroviaire), les appareils électriques et électroniques (appareils photo, jouets, postes de télévision, Machine à laver, systèmes audio, téléphones portables), les systèmes de contrôle, la distribution d'énergie, l'automatisation, etc. Leur utilisation étendue aux produits grand public courants, est une évolution majeure pour les technologies de l'information.

2. Définition d'un système embarqué :

De pratique, les systèmes embarqués sont impliqués dans des nombreux domaines. Il est difficile de donner une définition précise.

Un système embarqué peut être défini comme étant un système électronique et informatique autonome, dédié à une tâche bien précise. Ses ressources disponibles sont généralement limitées.

Il ne possède généralement pas des entrées/sorties standards et classiques comme un clavier ou un écran d'ordinateur. Le matériel et le logiciel ne sont pas aussi facilement discernables comme dans un environnement de travail classique de type ordinateur PC. Le logiciel embarqué étant enfoui, noyé dans le matériel. [1]

Le logiciel créé pour les systèmes embarqués est appelé firmware. Il est stocké dans la mémoire en lecture seule ou dans une mémoire flash plutôt que dans un disque dur.



3. Caractéristiques d'un système embarqué :

Les caractéristiques d'un système embarqué sont les suivants :

❖ L'autonomie :

Les systèmes embarqués doivent en général être autonomes, c'est-à-dire remplir leur mission pendant de longues périodes sans intervention humaine. Cette autonomie est indispensable lorsque l'intervention humaine est impossible. Les temps de réponses de ces systèmes sont aussi importants que l'exactitude des résultats.

❖ Fonctionnement en Temps Réel :

Les applications embarquées, comme des applications de système de contrôle, sont conduites par événements et doivent répondre rapidement à ces événements.

Il se peut qu'un système embarqué aura besoin d'opérations de calcul, elles doivent être faites en réponse à un événement extérieur.

❖ Environnement :

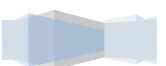
Ils sont soumis à de nombreuses contraintes dictées par l'environnement telles que la température, l'humidité, les vibrations, les chocs, les variations d'alimentation, la corrosion, l'eau, le feu, les radiations...etc.

❖ Faible coût :

Un assez grand nombre de produits d'embarqué sont sur les marchés où l'utilisateur ne veut pas payer le supplémentaire pour la performance ou fonctionnalité de plus, Les concepteurs ont donc dû concevoir avec des rapports optimaux entre le prix et la performance Le résultat de ceci est que les ressources disponibles sont minimales possibles. C'est pour quoi un système embarqué a rarement plus de quelques Méga octets de mémoire disponible.

❖ Faible consommation :

La minimisation de la consommation est essentielle pour les systèmes autonomes afin de maximiser l'autonomie des batteries. Une consommation excessive augmente le prix de revient du système embarqué car il faut alors des batteries de forte capacité.



❖ Faible encombrement et faible poids :

Ils doivent cohabiter sur une faible surface électronique analogique, électronique numérique. Notamment, c'est très important pour les applications portables où l'on doit minimiser la taille et les poids.

- Electronique *Pocket PC*, applications portables où l'on doit minimiser la consommation électrique
- Faible consommation : Batterie de 8 heures et plus (PC portable : 2 heures).

Les systèmes embarqués sont la plupart du temps dans des machines qui doivent fonctionner en continu pendant de nombreuses années, sans erreurs et dans certains cas réparent les erreurs quand elles arrivent. C'est pourquoi les logiciels sont toujours développés et testés avec plus d'attention que ceux pour les PC. [1] [2]

4. Architecture générale et les composants possibles :

L'architecture d'un système normal, un ordinateur, se compose de trois couches: application, système d'exploitation et matériel. De même, un système embarqué dispose de 3 couches. Chaque couche a la même fonctionnalité qu'un système normal. Mais Il y a des différences de sous composants du système. Les deux premières couches s'agissent de logiciel. Cependant, ce n'est pas un système qui contient tous les composants comme le système complet. Car le but de conception est de servir quelques tâches spécifiques et de concentrer à un unique travail. Le système d'exploitation est une couche logicielle sur laquelle on va se placer l'ensemble des applications lancées par les utilisateurs. Il est possible de modifier leurs composants, et aussi ajouter ou supprimer certains modules au besoin dépendant du but du système.

La dernière couche, de matériel c'est un ensemble d'éléments physiques employés pour le traitement de données. Les composants matériels sont limités et, peut-être, il y aurait un composant dédié aux tâches spécifiques.



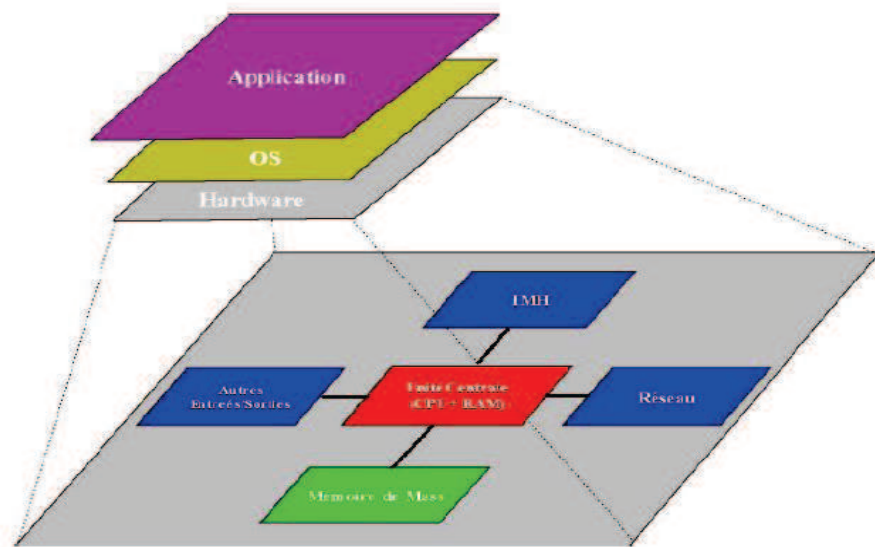


Fig.1: Architecture d'un système embarqué

Les systèmes embarqués utilisent généralement des microprocesseurs à basse consommation d'énergie ou des microcontrôleurs, dont la partie logicielle est en partie ou entièrement programmée dans le matériel, généralement dans une mémoire morte ROM, EPROM, EEPROM, FLASH, etc. On peut équiper plusieurs composants supplémentaires : une carte dédiée aux applications spécifiques, mémoire de Mass (DiskOnChip, FlashDisk, CDROM, carte SD), carte réseau, entrées sorties spéciaux.

Ci-dessous quelques dispositifs matériels:

➤ **Entrées:**

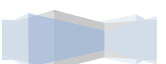
- Les capteurs/convertisseurs (pression, audio, température...)
- Le clavier, boutons poussoirs ou télécommandes (infrarouge, Bluetooth, radio...)
- Les lecteurs de tags ou de codes-barres, ...

➤ **Sorties:**

- L'écran et afficheurs LCD.
- Le Système d'alarme ou synthèse vocale.
- L'imprimante en tous genres comme papier, étiquettes, photos...

➤ **Mémoire de mass :**

- L'utilisation de ROM: Disque virtuel CD, DVD.
- Le disque à distance: NFS, TFTP.



- La mémoire flash: FlashDisk, CompactDisk, DiskOnChip, SDCard, Mémoire stick, clés USB.

➤ **IHM:**

Normalement, un système embarqué est un dispositif qui sert à communiquer entre l'humain et la machine. Un exemple réaliste d'IHM est l'écran avec les dispositifs «TouchScreen» qui est visé aux PDAs ou Handheld. [1]

5. Développement de systèmes embarqués :

Le développement de systèmes embarqués nécessite des connaissances à la fois en électronique et en informatique. Parmi le matériel nécessaire pour réaliser un système embarqué on trouve :

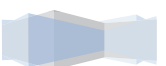
- La documentation (datasheet) sur les composants utilisés. C'est la première source d'informations pour le développement.
- L'outillage de base de l'électronicien (fer à souder, insoleuse...)
- Les outils d'analyse temporelle : oscilloscope, analyseur logique...
- Des composants de base (résistances, condensateurs...)
- Un microprocesseur ou un microcontrôleur
- Un compilateur croisé (dit aussi en anglais cross-compiler)
- Un programmeur de microcontrôleur ou un programmeur in-situ
- Un émulateur in-circuit ou ICE (In Circuit Emulator) Considéré comme l'équipement roi pour le debug matériel et logiciel.

6. Conclusion :

Aujourd'hui les systèmes embarqués sont enfouis dans la majeure partie des équipements du quotidien et concernent la quasi-totalité des secteurs d'activité, Ils jouent un rôle très important dans la vie humaine.

Dans ce chapitre, on a défini et décrit les systèmes embarqués et ses différentes caractéristiques, leur Architecture générale et les différents composants possibles.

Dans ce qui suit, nous allons voir une généralité sur le modèle OSI et la pile TCP/IP



Chapitre II

Généralités sur le modèle
OSI et la pile TCP/IP.

1. Introduction :

Les systèmes embarqués sont aujourd'hui fortement communicants .Cela est possible grâce aux puissances de calcul offertes par les microprocesseurs et les microcontrôleurs et grâce aussi à l'explosion de l'usage la connectivité Internet ou connectivité IP.

La connectivité Internet permet de raccorder tout système embarqué au réseau Internet. L'ajoute d'une connectivité IP à un système embarqué permet fondamentalement le contrôler à distance de n'importe où par :

- une application réseau.
- plus simplement par le Web en utilisant un navigateur Internet.

La connectivité IP demande d'embarquer une suite de protocoles Internet sur le système électronique pour pouvoir être mise en œuvre.

2. Le modèle architectural OSI :

Le modèle OSI (Open System Interconnection) est un modèle d'architecture permettant à des réseaux hétérogènes de communiquer. Il s'agit d'une norme officielle, ouverte à des développements ultérieurs.

Ce modèle composé de 7 couches décrivant l'architecture des réseaux et les protocoles OSI correspondants. Chaque couche doit réaliser une fonction bien précise, elles sont numérotées de la plus matérielle à la plus logicielle, les fonctions propres à chaque niveau sont assurées par des entités (matériel et/ou logiciel). Une architecture réseau qui respecte le modèle OSI offre une grande fiabilité et une transparence dans le fonctionnement du réseau.
[3]

2.1. Les sept couches du modèle OSI :

Dans ce modèle, le problème consistant à déplacer des informations entre des ordinateurs est divisé en sept problèmes plus petits et plus faciles à gérer. Chacun des sept petits problèmes est représenté par une couche particulière du modèle. Voici les sept couches du modèle de référence OSI :



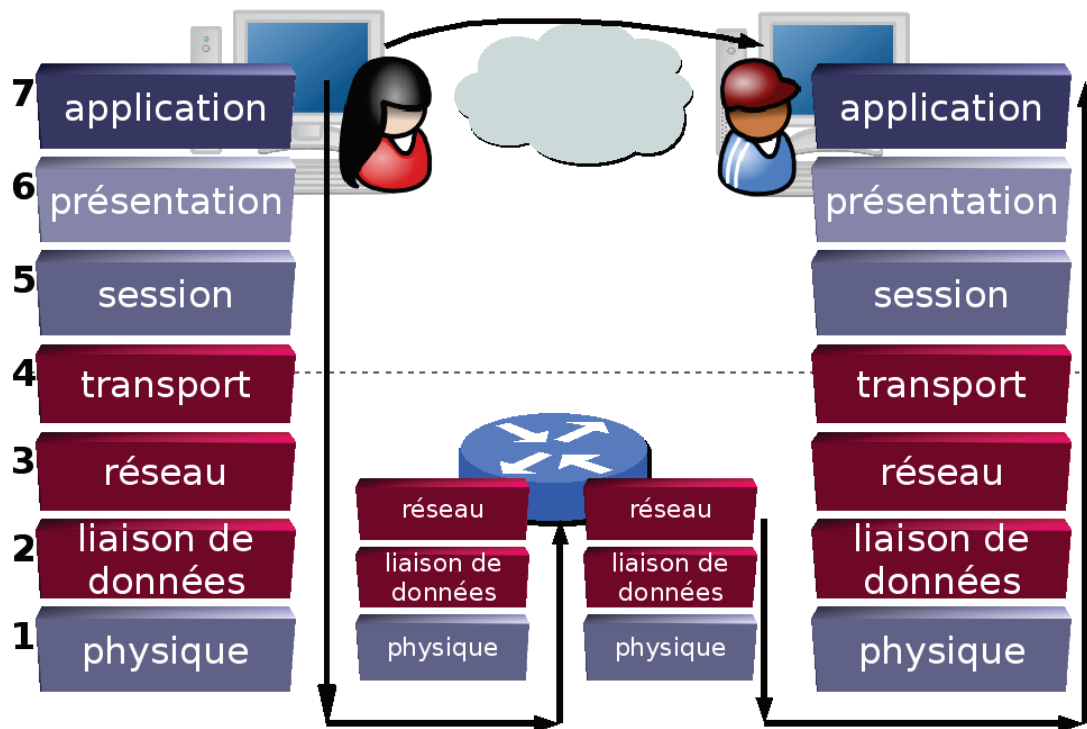


Fig.2 : Modèle en 7 couches de l'OSI

Une couche ne définit pas un protocole, elle délimite un service qui peut être réalisé par plusieurs protocoles de différentes origines. Ainsi chaque couche peut contenir tous les protocoles que l'on veut, pourvu que ceux-ci fournissent le service demandé à ce niveau du modèle.

Un des intérêts majeurs du modèle en couches est de séparer la notion de communication, des problèmes liés à la technologie employée pour véhiculer les données.

Au cours de cette partie, nous verrons toutes les couches du modèle OSI, en commençant par la couche 1 :

- **La couche physique :**

Décrit des caractéristiques électriques (niveaux de tension, signaux...), mécaniques (forme des connecteurs, caractéristiques des câbles...), physiques (distances maximales de transmission...) et logiques (débit, codage...) de connexion du poste au réseau. Elle fait en sorte que physiquement, l'émission d'un bit à 1 (bit « brut » en bande de base ou signal analogique...) ne soit pas considérée à la réception comme un bit 0. Elle gère le type de transmission (synchrone/asynchrone) et procède



éventuellement à la modulation/démodulation du signal. L'unité d'échange à ce niveau est le bit.

Comme La paire torsadée, les connecteurs RJ45 et la carte réseau Ethernet travaillent en couche 1 OSI.

- **La couche 2 : est la couche liaison de données :**

Cette couche reçoit les données brutes de la couche physique, elle les organise en trames, elle détecte les erreurs, elle retransmet les trames erronées, elle gère les acquittements (ACK) qui indiquent si les données sont bien transmises Une fois ceci fait, elle transmet ses données formatées à la couche réseau. Cette couche travaille aussi avec la couche qui la précède à savoir la couche physique (couche 1). L'unité de données est la trame (data frame) et les informations qui circulent sur le réseau sont donc structurées en trames, Elle transforme les paquets fournis par la couche réseau (couche 3) en trame.

La carte réseau (en partie – adresse MAC), le protocole HDLC (High-level DataLink Control) travaillent au niveau 2.

- **La couche 3 : est la couche réseau :**

Son rôle est chargée de l'acheminement des paquets de données qui transitent sur l'ensemble du réseau .Elle doit donc utiliser des informations d'adressage. Ces paquets seront sans doute amenés à traverser des nœuds intermédiaires, un routage est donc nécessaire (c'est-à-dire que la couche réseau doit déterminer le meilleur chemin de communication c'est le routage). Si un nœud est surchargé ou hors service, le contrôle de flux doit éviter les congestions en régulant la charge du trafic ou en déroutant les paquets vers un autre nœud. L'unité de données est le paquet. La couche réseau assure également un service de traduction des adresses logiques (adresse IP) en adresses physiques (adresse MAC).

- **La couche 4 : est la couche transport :**

Assure l'interface « middleware » entre les couches hautes orientées « traitement » (session, présentation, application) et les couches basses orientées « réseaux » (réseau, liaison, physique). Elle doit assurer à la demande de la couche session le transport correct des messages, même au travers de plusieurs réseaux, de manière fiable et économique et ce de bout en bout entre l'émetteur et le récepteur.



Ce service, transparent pour l'utilisateur même au travers de plusieurs réseaux, lui impose de gérer le flux et de corriger les ultimes erreurs. Elle doit donc garantir les services qui n'auraient pas été assurés dans les couches inférieures. C'est en effet le dernier niveau chargé de l'acheminement de l'information.

L'unité de données est le message. La couche Transport de l'émetteur segmente les données issues de la couche Session en messages et la couche Transport du récepteur reconstitue les messages en remplaçant les paquets issus de la couche Réseau dans le bon ordre. Elle permet de multiplexer plusieurs flux d'informations sur le même support et inversement de démultiplexer.

TCP, UDP, SPX, NetBEUI... ces sont des protocoles de transport qui travaillent au niveau 4

- **La couche 5 : est la couche session :**

Est la première couche orientée traitement .Elle permet l'ouverture et la fermeture d'une session de travail entre systèmes distants et assure la synchronisation du dialogue. C'est à ce niveau que l'on décide également du mode de transmission (simplex, half-duplex, full-duplex).La synchronisation du dialogue se fait au moyen de points de contrôle ainsi, Lorsqu'un problème se produit, seules les données émises après le dernier point de Contrôle correctement reçu seront réexpédiées.

Le langage de requête SQL (Structured Query Language), le système de partage de fichiers NFS (Network File System) TLS (Transport Layer Security), RPC (Remote Procedure Call) ... travaillent au niveau 5.

- **La couche 6 : est la couche présentation :**

Cette couche s'occupe de la partie syntaxique et sémantique de la transmission de l'information. Elle s'occupe par exemple du codage des caractères permettant ainsi à deux systèmes hétérogènes de communiquer. Un caractère A restera un caractère A. C'est le système d'encodage, qui traduit les informations pour qu'elles soient compréhensibles indépendamment du système utilisé. On retrouve aussi à ce niveau la partie cryptage/décryptage des communications.

La conversion d'un fichier texte MS-DOS en un fichier texte Unix, le logiciel de cryptage PGP (Pretty Good Privacy) ASN (Abstract Syntax Notation), SMB (Server Message Block)... travaillent au niveau 6.



- **La couche 7 : est la couche application :**

Cette couche fournit des services utilisables par les applications de l'utilisateur. Pour simplifier on peut dire que « l'application de l'utilisateur va utiliser la couche application OSI pour fournir aux processus de l'application utilisateur les moyens d'accéder à l'environnement OSI et de communiquer ». Les applications de l'utilisateur communiquent entre elles en faisant appel à des éléments de service d'application ou ASE (Application Service Element). Cette technique permet de partager des modules communs OSI entre plusieurs applications utilisateurs.

Les principaux services proposés sont :

- Transfert de fichiers (FTP File Transfert Protocol),
- Messagerie ou courrier électronique (SMTP Send Mail Transfert Protocol),
- Soumission de travaux à distances (client-serveur),
- Accès aux fichiers distants (NFS Network File System),
- Terminal virtuel (Telnet...),
- Web (HTTP Hyper Text Transfert Protocol)

Les systèmes d'exploitation réseau, le protocole HTTP utilisé avec le web...travaillent au niveau 7. [4]

3. La pile de protocoles TCP/IP (Transmission Control Protocol/Internet Protocol) :

3.1. Bref historique :

En 1969 aux Etats Unis, l'agence gouvernementale DARPA (Defense Advanced Research Project Agency) lance un projet de réseau expérimental, basé sur la commutation de paquets. Ce réseau, nommé ARPANET, fut construit dans le but d'étudier les technologies de communications, indépendamment de toute contrainte commerciale.

Un grand nombre de techniques de communication par modems datent de cette époque.

En 1975, le réseau passe officiellement du stade expérimental au stade opérationnel. Le développement d'ARPANET ne s'arrête pas pour autant, les bases des protocoles TCP/IP sont développées à ce moment, donc après que ARPANET soit opérationnel.

En Juin 1978 Jon Postel définit IPv4 et en 1981 IP est standardisé dans la RFC 791.



En 1983 les protocoles TCP/IP sont adoptés comme un standard militaire et toutes les machines sur le réseau commencent à l'utiliser. Pour faciliter cette reconversion, la DARPA demande à l'université de Berkeley d'implémenter ces protocoles dans leur version (BSD) d'Unix. Ainsi commence le mariage entre ce système d'exploitation et les protocoles TCP/IP. L'apport de l'Université de Berkeley est majeur, tant au niveau théorique (concept des sockets) qu'au niveau de l'utilisateur, avec des utilitaires très homogènes qui s'intègrent parfaitement au paradigme d'usage existant (rcp, rsh, rlogin...).

Depuis cette époque, un nouveau terme est apparu pour désigner cette interconnexion de réseaux, l'Internet avec un " i " majuscule.

Le succès de cette technologie est alors très important et suscite un intérêt croissant de la part d'acteurs très divers, et en particulier La " National Science Foundation " qui y voit un intérêt majeur pour la recherche scientifique et soutient donc ce nouveau moyen de mettre en communication tous les chercheurs.

Depuis 1990, ARPANET n'est plus, pourtant le terme Internet demeure il désigne maintenant un espace de communication qui englobe la planète tout entière. Des millions de sites partout sur la surface du globe y sont connectés.

Depuis 1994, l'Internet s'est ouvert au commerce, surtout avec l'apparition en 1991 d'un nouvel outil de consultation, le « World Wide Web » ou « Web » et ses interfaces populaires : Mosaic3, Netscape, Mozilla, Firefox...

Depuis 1995, pour faire face à sa popularité fortement croissante et aux demandes de transactions sécurisées, le protocole évolue et une nouvelle version, la version 6 (IPv6), est définie et en cours de déploiement expérimental.

Les services réseaux les plus populaires sont principalement :

- Le courrier électronique qui permet l'échange de messages entre usagers.
- Les innombrables forums de discussion « news ».
- Le transfert de fichiers entre machines « FTP ».
- Le « remote login », ou ses équivalents cryptés (ssh), qui permet à un utilisateur de se connecter sur un site distant depuis son poste local.
- Puis maintenant la radio, la vidéoconférence, la réalité virtuelle avec le VRML, le chat,... etc.



En conclusion sur cette historique on peut dire que l'Internet est une collection apparemment anarchique (il n'y a pas de structure hiérarchique et centralisée) de réseaux inter-connectés et appartenant à divers propriétaires. [3]

3.2. Le modèle TCP/IP:

Le modèle OSI plutôt théorique a été remplacé par un modèle plus pratique, le modèle TCP/IP. A la différence du modèle OSI, qui a d'abord été normalisé avant d'être appliqué, le modèle TCP/IP, a tout d'abord été déployé avec succès avant d'être normalisé.

TCP/IP ou la « pile TCP/IP » est une suite de protocoles. Le sigle TCP/IP signifie «Transmission Control Protocol/Internet Protocol». Il provient des noms des deux protocoles majeurs de la suite de protocoles, c'est-à-dire les protocoles TCP et IP. [5]

3.3. Caractéristiques de la pile TCP/IP :

Le succès de TCP/IP s'il vient d'abord d'un choix du gouvernement Américain s'appuie ensuite sur des caractéristiques intéressantes :

1. C'est un protocole ouvert, les sources (C) en sont disponibles gratuitement et ont été développés indépendamment d'une architecture particulières, d'un système d'exploitation particulier, d'une structure commerciale propriétaire. Ils sont donc théoriquement transportables sur n'importe quel type de plate-forme, ce qui est prouvé de nos jours.
2. Ce protocole est indépendant du support physique du réseau. Cela permet à TCP/IP d'être véhiculé par des supports et des technologies aussi différents qu'une ligne série, un câble coaxial Ethernet, une liaison louée, un réseau token-ring, une liaison radio (satellites, wireless 802.11a/b/g, infrarouge, xDSL, fibre optique, la liste des supports et des technologies n'est pas exhaustive. . .
3. Le mode d'adressage est commun à tous les utilisateurs de TCP/IP quelle que soit la plate-forme qui l'utilise. Si l'unicité de l'adresse est respectée, les communications aboutissent même si les hôtes sont aux antipodes.
4. Les protocoles de hauts niveaux sont standardisés ce qui permet des développements largement répandus et inter-opérables sur tous types de machines. [3]



3.4. Comparaison entre le modèle OSI et la pile TCP/IP :

La suite de protocoles désignée par TCP/IP est construite sur un modèle en couches moins complet que la proposition de l'ISO. Quatre couches sont suffisantes pour définir l'architecture de ce protocole.

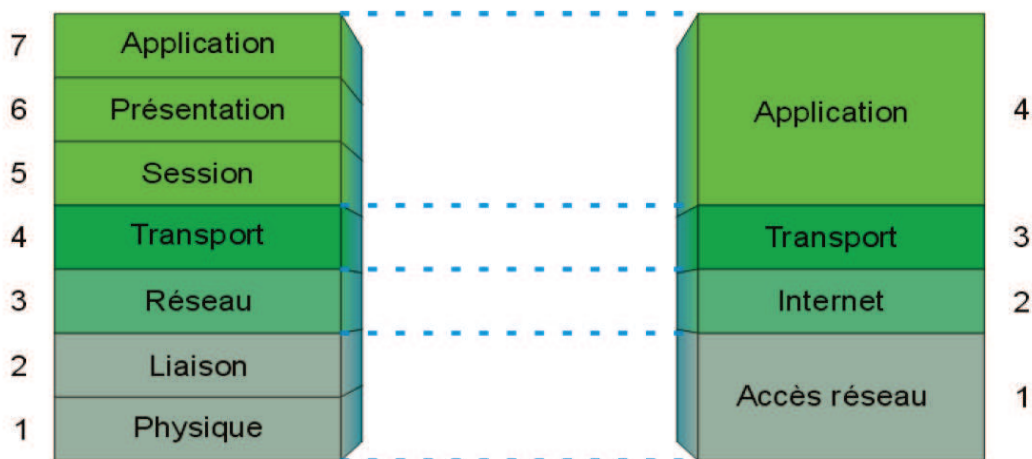


Fig.3 : Le modèle OSI et l'architecture TCP/IP.

4 Couche Application (Application layer).

3 Couche Transport (Transport layer).

2 Couche Internet (Internet layer).

1 Couche Accès réseau (Network access layer).

3.4.1. Couche application :

C'est la couche de haut niveau, elle correspond directement avec l'utilisateur, elle englobe les couches OSI d'application, de présentation et de session. Elle s'assure que les données soient correctement "empaquetées" pour qu'elles soient lisibles par la couche suivante.

3.4.2. Couche Transport :

La couche transport permet d'identifier les applications qui communiquent. Pour faciliter la communication on a défini non pas des noms d'applications, mais des ports de communication spécifiques à chaque application.



La couche transport gère 2 protocoles de livraison des informations : UDP est dit "sans connexion" et TCP est dit "avec connexion".

3.4.3. Couche Internet :

Ce sont ici des protocoles de haut niveau de la couche réseau. IP permet le routage des informations entre réseaux, c'est ici que l'adresse IP est utilisée.

ICMP est un protocole de contrôle il met à disposition des outils de dépistage d'erreur et de signalisation. C'est un protocole important qui mérite que l'on s'y arrête. Nous en reparlerons plus en détail. Ne pas oublier le protocole ARP.

3.4.4. Couche Accès réseau :

Cette couche est regrouper les couches physique et liaison de données du modèle OSI. Il assure la bonne gestion du médium (détection de collisions) et permet l'acheminement des informations entre émetteur et destinataire au niveau des adresses MAC. [6]

4. Les différents protocoles TCP/IP :

Les différents protocoles utilisés sur le réseau Internet sont : le protocole IP, le protocole ARP, le protocole ICMP, le protocole UDP, le protocole TCP, le protocole DHCP, ainsi que le protocole HTTP.

Un protocole est une méthode standard qui permet la communication entre deux machines, c'est-à-dire un ensemble de règles et de procédures à respecter pour émettre et recevoir des données sur un réseau.

Sur Internet, les protocoles utilisés font partie d'une suite de protocoles, c'est-à-dire un ensemble de protocoles reliés entre-eux.

La réalisation de notre projet nous a fait travailler sur quelques protocoles énoncés précédemment. Nous allons décrire ces différents protocoles dans les paragraphes qui suivent.

Les protocoles Internet seront transportés par des trames Ethernet compatible MAC 802.3.



4.1. Format d'une trame Ethernet MAC 802.3 10-Base T :

Préambule	SOF	Adresse Destination	Adresse Source	longueur/t ype	Données	Bourrage	FCS
7 octets	1 octet	6 octets	6 octets	2 octets	46 à 1500 octets		4 octets

Fig.4 : Trame Ethernet 802.3

- **Le préambule** : est une suite de 0 et de 1 alternés. Il permet à l'horloge du récepteur de se synchroniser sur celle de l'émetteur.
- **Délimiteur de début de trame SOF (*Start of Frame*)** : permet de trouver le début de trame et qui marque la fin des informations de synchronisation et qui contient la séquence de bits 10101011.
- **Adresse Destination** : Adresse physique de la carte Ethernet destinataire de la trame. Elle est codée sur 6 octets.
- **Adresse Source** : Adresse physique de la carte Ethernet émettrice de la trame. Elle est codée sur 6 octets.
- **Type** : indiquer le type de protocole de niveau 3 employé pour transmettre le message.
- **longueur/type** : peut être utilisé de deux façons. Si la valeur est inférieure à 1536 en notation décimale, alors elle indique la longueur. L'interprétation de la longueur est utilisée lorsque la couche LLC fournit l'identification de protocole. La valeur de type précise le protocole de couche supérieure qui reçoit les données une fois le traitement Ethernet terminé. La longueur indique le nombre d'octets du champ des données.

Quelques types courants :

0x0800 Internet Protocol, Version 4 (IPv4)

0x0806 Address Resolution Protocol (ARP)

0x8035 Reverse Address Resolution Protocol (RARP)

0x86DD Internet Protocol, Version 6 (IPv6)

- **Données** : les données de l'utilisateur.
- **Le champ de Bourrage**: Le champ de bourrage permet de compléter le champ de données dans le cas où celui-ci contient moins de 46 octets.
- **Le FCS (*Frame Check Sequence*)** : est un champ de contrôle qui permet de valider l'intégrité de la trame .Il utilise un CRC (*Cyclic Redundancy Check*) qui englobe tous



les champs de la trame. Ainsi, la station réceptrice peut décider si la trame est correcte et doit être transmise à la couche supérieure.

Chaque machine connectée au réseau possède un numéro d'identification de 48 bits. Ce numéro est un numéro unique qui est fixé dès la fabrication de la carte en usine. Toutefois la communication sur Internet ne se fait pas directement à partir de ce numéro mais à partir d'une adresse dite logique attribuée par un organisme: l'adresse IP.

Ainsi, pour faire correspondre les adresses IP aux adresses physiques, le protocole ARP interroge les machines du réseau pour connaître leur adresse physique.

4.2. Le protocole ARP (Address Resolution Protocol) :

C'est un protocole situé entre la couche liaison de données et la couche réseau. Il propose une solution au problème de résolution d'adresse où un protocole de niveau réseau a besoin d'envoyer un message à une destination et ne dispose que de l'adresse de niveau réseau(IP) de cette destination. ARP fait la correspondance entre les adresses IP et les adresses MAC, Les adresses MAC sont les adresses physiques propres à chaque carte réseau. Ce protocole étant présent sur tout (ou la plupart) des LAN.

4.2.1. Format du datagramme ARP :

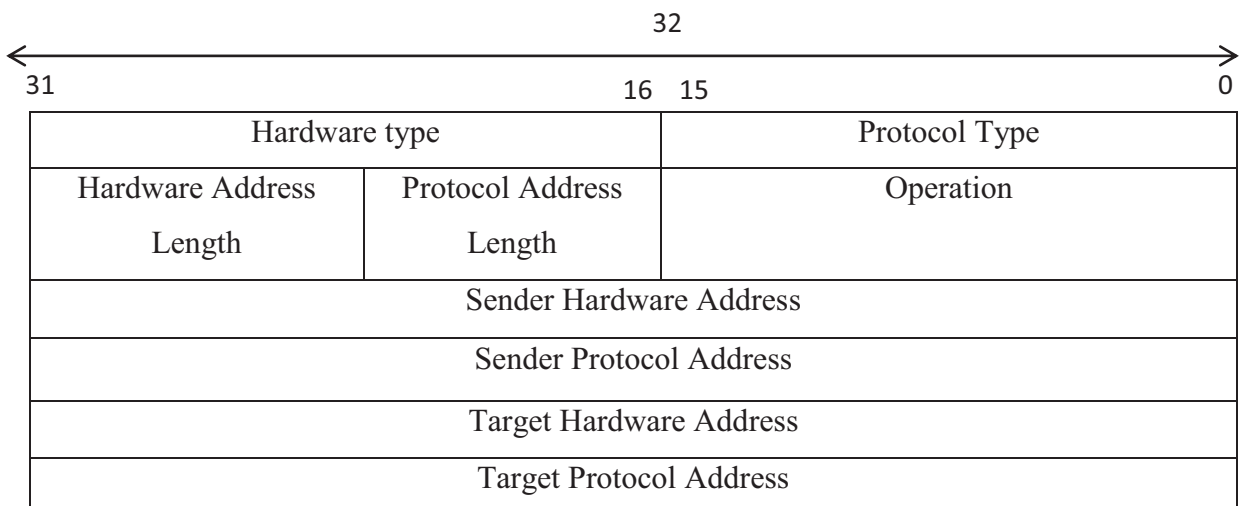


Fig.5 : Datagramme ARP

- **Hardware type (type de matériel) :** il s'agit du type du matériel sur lequel sont transmises les trames du protocole choisit. Ce champ est codé sur 2 octets. c'est 1 pour Ethernet.

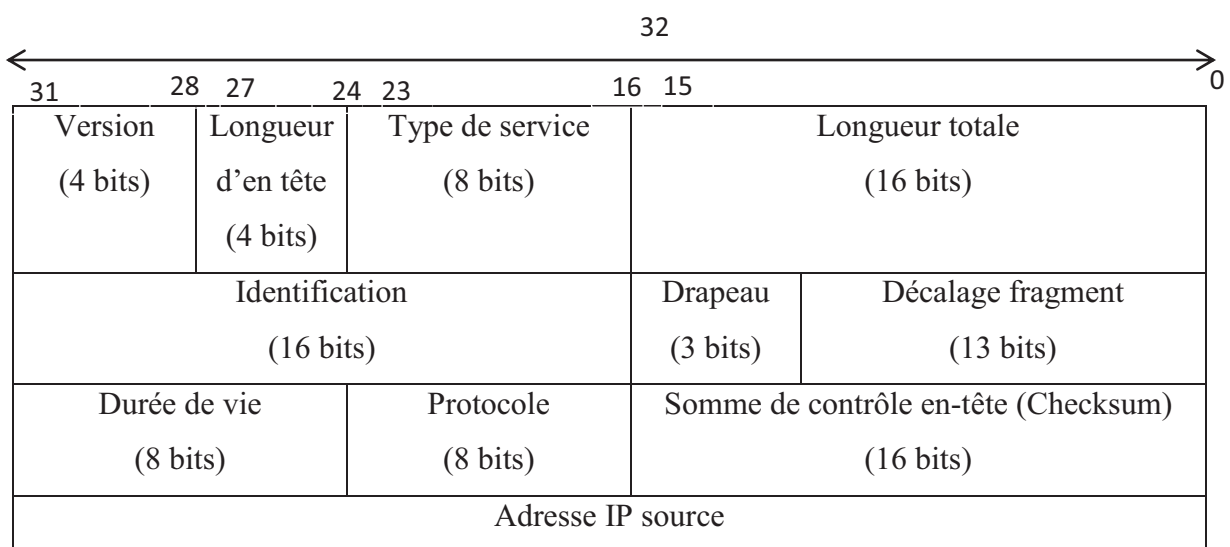


- **Protocol Type** : il s'agit du type de protocole que nous utiliserons pour la communication. Ce champ est codé sur 2 octets, c'est 0x0800 pour des adresses IP.
- **Hardware Address Length** : nous définissons la longueur des adresses physiques en octets. Ce champ est codé sur 1 octet. (6 octets pour Ethernet).
- **Protocol Address Length** : nous définissons la longueur de l'adresse logique .Ce champ est codé sur 1 octet. (4 octets pour IP).
- **Operation** : il s'agit de définir le type d'opération de la trame ARP envoyé, Ce champ est codé sur 1 octet. QUESTION (1) ou REPONSE(2).
- **Sender Hardware Address** : adresse physique de l'émetteur.
- **Sender Protocol Address** : adresse logique de l'émetteur.
- **Target Hardware Address** : adresse physique du destinataire.
- **Target Protocol Address** : adresse logique du destinataire.

4.3. Le protocole IP (Internet Protocol) :

Le protocole IP fait partie de la couche Internet de la suite de protocoles TCP/IP. C'est un des protocoles les plus importants d'Internet car il permet l'élaboration et le transport des datagrammes IP (les paquets de données), sans toutefois en assurer la « livraison ». En réalité, le protocole IP traite les datagrammes IP indépendamment les uns des autres en définissant leur représentation, leur routage et leur expédition.

4.3.1. Format du datagramme IP :



(32 bits)
Adresse IP destination (32 bits)
IP OPTIONS
Données

Fig.6 : Datagramme IP

- **Version (4 bits)** : il s'agit de la version du protocole IP que l'on utilise actuellement on utilise la version 4 (IPv4). Elle est codée sur 4 bits : 0100 pour IPv4 et 0110 pour IPv6.
- **Longueur d'en-tête ou IHL pour Internet Header Length**: Longueur de l'en-tête en mots de 32 bits. Cette valeur est utilisée pour distinguer la partie en-tête de la partie données du paquet. La représentation usuelle de l'en-tête se fait sur 32 bits de largeur. Comme le champ Options n'est pas obligatoire, la valeur minimum du champ IHL est 5 (0101). Elle est codée sur 4 bits
- **Type de service** : le type de service donne une indication sur la qualité de service souhaitée, Ce paramètre est utilisé pour "guider" le choix des paramètres des services actuels lorsqu'un datagramme transite dans un réseau particulier.
Certains réseaux offrent un mécanisme de priorité, traitant préférentiellement un tel trafic par rapport à un trafic moins prioritaire. Principalement, le choix offert est une négociation entre les trois contraintes suivantes : faible retard, faible taux d'erreur, et haut débit. Ce champ est codé sur 8 bits.
- **Longueur totale**: Cette partie indique la longueur totale du datagramme exprimée en octets, en-tête et données comprises. Ce champ étant codé sur 2 octets.
- **Identification** : une valeur d'identification unique assignée par l'émetteur pour identifier les fragments d'un même datagramme. Ce champ est codé sur 16 bits.
- **Drapeaux (flags)** : 3 bits Ce champ contient 3 indicateurs d'état :
 - **Reserved flag** : doit toujours être à 0.
 - **Don't Fragment (DF)** : à 0 si le paquet peut être fragmenté ; à 1 s'il ne doit pas être fragmenté.
 - **More Fragments (MF)** : à 1 si d'autres fragments sont attendus, à 0 s'il n'y a pas/plus de fragments.



- **déplacement de fragment** (13 bits) : champ permettant de connaître la position du début du fragment dans le datagramme initial. L'unité de mesure de ce champ est de 8 octets (le premier fragment ayant une valeur de zéro).
- **Durée de vie** appelée aussi **TTL**, pour *Time To Live* (8 bits) : ce champ indique le nombre maximal de routeurs à travers lesquels le datagramme peut passer. Ainsi ce champ est décrémenté à chaque passage dans un routeur, lorsque celui-ci atteint la valeur critique de 0, le routeur détruit le datagramme. Cela évite l'encombrement du réseau par les datagrammes perdus.
- **Protocole**: ce champ indique quel protocole de niveau supérieur est utilisé dans la section donnée du datagramme Internet. Ce champ est codé sur 8 bits.
 - CMP : 1
 - UDP : 17
 - IGMP : 2
 - TCP : 6
- **Somme de contrôle de l'en-tête, ou *header checksum* (16 bits)**: pour s'assurer de l'intégrité de l'en-tête. A la réception de chaque paquet, la couche calcule cette valeur, si elle ne correspond pas à celle trouvée dans l'en-tête le datagramme est oublié sans message d'erreur.
- **Adresse IP Source**: ce champ représente l'adresse IP de la machine émettrice, il permet au destinataire de répondre. Ce champ est codé sur 32 bits.
- **Adresse IP destination**: adresse IP du destinataire du message. Ce champ est codé sur 32 bits.
- **IP OPTIONS** : 24 bits pour préciser des options de comportement des couches IP traversées et destinataires. Les options les plus courantes concernent :
 - Des problèmes de sécurité.
 - Des enregistrements de routes.
 - Des enregistrements d'heure.
 - Des spécifications de route à suivre.

4.4. Le protocole ICMP (Internet Control Message Protocol) :

Le protocole ICMP (Internet Control Message Protocol) est un protocole faisant partie de la couche IP qui permet de gérer les informations relatives aux erreurs du protocole IP. Il ne permet pas de corriger ces erreurs, mais d'en informer les différents émetteurs des



Datagrammes en erreurs. Chaque pile IP, que ce soit des routeurs ou des stations de travail, gèrent ICMP par défaut.

Les messages ICMP sont généralement envoyés automatiquement dans l'un des cas suivants :

- Un datagramme IP ne peut pas atteindre sa destination.
- Un routeur IP (passerelle) ne peut pas transmettre les datagrammes sur l'état de la transmission.

Un routeur IP redirige l'hôte d'envoi pour utiliser un meilleur itinéraire vers la destination. Les messages ICMP sont encapsulés et envoyés dans les datagrammes IP comme le montre l'illustration suivante :

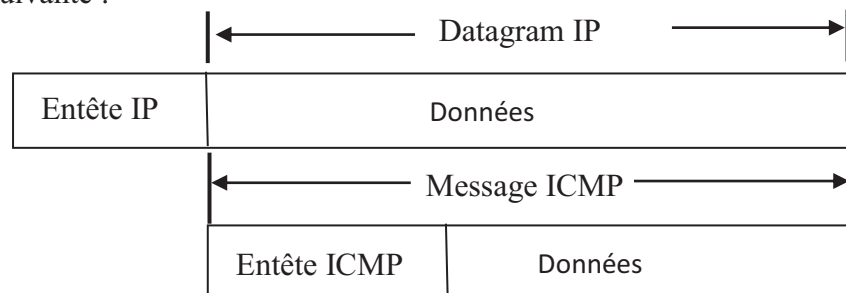


Fig.7 : Message ICMP

4.4.1. Format du datagramme ICMP :

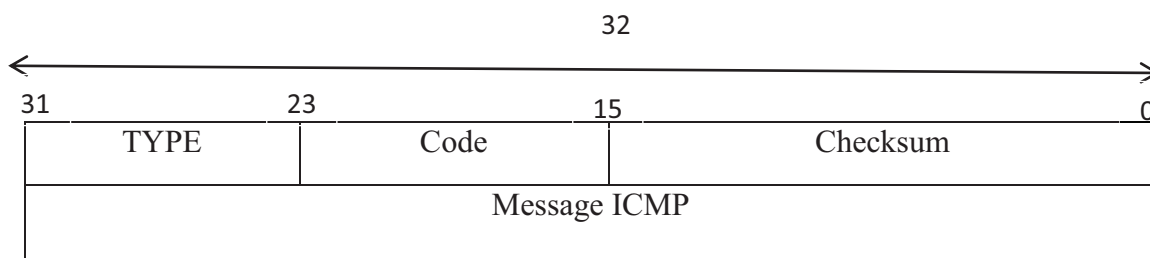


Fig.8 : Datagramme ICMP

- **Type :** (8 bits) Indique le type du message.
- **Code :** (8 bits) Fournit des informations supplémentaires sur le type de message.
- **Total de Contrôle :** Checksum (16 bits) Permet de contrôler l'intégrité du message ICMP.



➤ Signification des messages ICMP :

Type	Code	Message	Signification du message
8	0	Demande d'ECHO	Ce message est utilisé lorsqu'on utilise la commande <i>PING</i> . Cette commande, permettant de tester le réseau
3	5	Destinataire inaccessible	Le routage a échoué
3	3	Destinataire inaccessible	Le port n'est pas accessible
11	1	Temps de réassemblage de fragment dépassé	Ce message est envoyé lorsque le temps de réassemblage des fragments d'un datagramme est dépassé.
13	0	Timestamp request	ne machine demande à une autre son heure et sa date système (universelle)
15	0	Demande d'adresse réseau	Ce message permet de demander au réseau une adresse IP
12	0	En-tête erroné	Ce message est envoyé lorsqu'un champ d'un en-tête est erroné. La position de l'erreur est retournée

4.5. Le protocole UDP (User Datagram Protocol) :

User Datagram Protocol est un protocole de transport sans connexion. Le rôle de ce protocole est de permettre la transmission de données et de manière très simple entre deux entités, étant donné qu'il ne fournit pas de contrôle d'erreurs (perte de messages, messages non ordonnés, etc....) donc est un protocole non fiable.

4.5.1. Format du datagramme UDP :

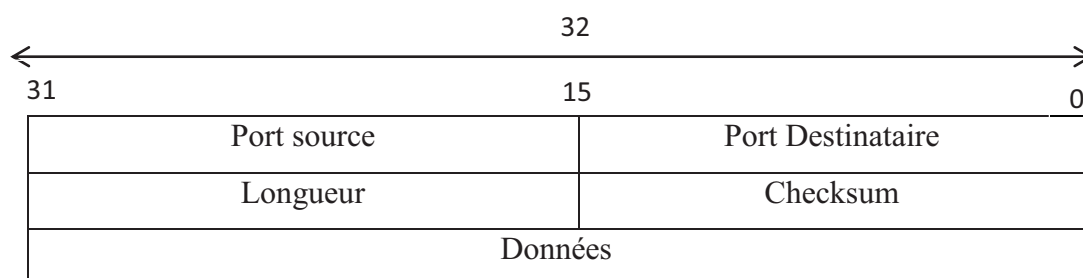


Fig.9 : Datagramme UDP



- **Port source** : c'est un champ optionnel. indique depuis quel port le paquet a été envoyé.
- **Port Destinaire** : indique à quel port le paquet doit être envoyé.
- **Longueur** : indique la longueur totale du segment UDP (en-tête et données). La longueur minimale est donc de 8 octets (taille de l'en-tête).
- **Checksum** : celle-ci permet de s'assurer de l'intégrité du paquet reçu quand elle est différente de zéro. Elle est calculée sur l'ensemble de l'en-tête UDP et des données, mais aussi sur un pseudo en-tête constitué de l'information typique d'un en-tête IP.

Le pseudo entête Utilisé pour calculer le checksum à la source et à la destination ajoutée avant l'en-tête UDP contient l'adresse IP source, l'adresse IP destinataire, le code de protocole, et la longueur du segment UDP. Cette information permet d'augmenter l'immunité du réseau aux erreurs de routage de datagrammes.

Le datagramme du pré en-tête est décrit dans la figure suivante :

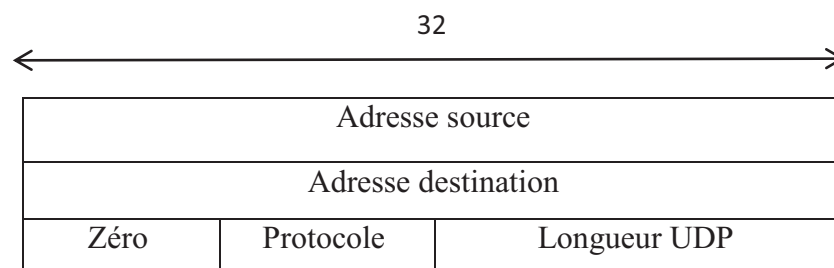


Fig.10 : Datagramme du pré en-tête UDP

4.6. Le protocole TCP (Transmission Control Protocol) :

Le protocole TCP est défini dans le but de fournir un service de transfert de données de haute fiabilité entre deux machines.

TCP est un protocole orienté connexion, c'est-à-dire qu'il permet à deux machines qui communiquent de contrôler l'état de la transmission.

Les caractéristiques principales du protocole TCP sont les suivantes:

- TCP permet de remettre en ordre les datagrammes en provenance du protocole IP.
- TCP permet de vérifier le flux de données afin d'éviter une saturation du réseau.
- TCP permet de formater les données en segments de longueur variable afin de les "remettre" au protocole IP.



- TCP permet de multiplexer les données, c'est-à-dire de faire circuler simultanément des informations provenant de sources (applications par exemple) distinctes sur une même ligne.
- TCP permet enfin l'initialisation et la fin d'une communication de manière courtoise.

Grâce au protocole TCP, les applications peuvent communiquer de façon sûre , indépendamment des couches inférieures.

Lors d'une communication à travers le protocole TCP, les deux machines doivent établir une connexion. La machine émettrice (celle qui demande la connexion) est appelée client, tandis que la machine réceptrice est appelée serveur. On dit qu'on est alors dans un environnement client-serveur.

Les machines dans un tel environnement communiquent en full-duplex, c'est-à-dire que la communication se fait dans les deux sens.

La figure suivante montre la structure d'un en-tête TCP.

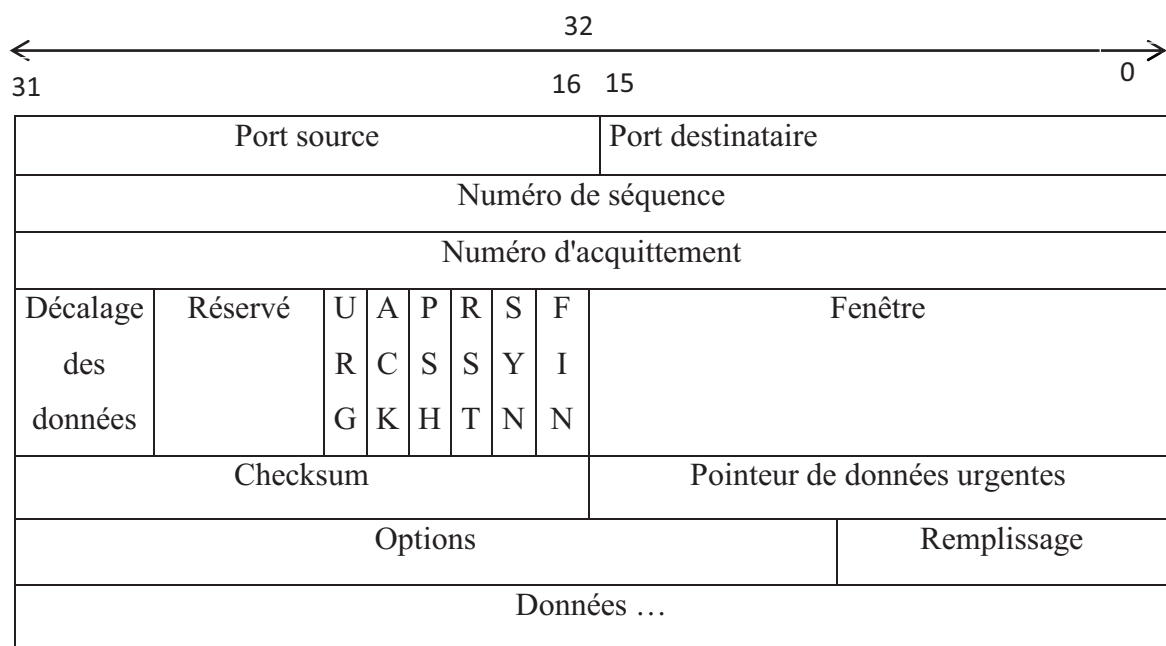


Fig.11 : Datagramme de l'en-tête TCP

- **Port source** : Numéro du port source.
- **Port destination** : Numéro du port destination.
- **Numéro de séquence** : Numéro de séquence du premier octet de ce segment.
- **Numéro d'acquittement** : Numéro de séquence du prochain octet attendu.



- **Décalage des données** (4 bits): il permet de repérer le début des données dans le paquet. Le décalage est ici essentiel car le champ d'options est de taille variable
- **Réservé** : Réservé pour un usage futur.
 - URG : Signale la présence de données URgentes
 - ACK : Signale que le paquet est un accusé de réception (ACKnowledgement)
 - PSH : Données à envoyer tout de suite (PuSH)
 - RST : Rupture anormale de la connexion (ReSeT)
 - SYN : Demande de synchronisation (SYN) ou établissement de connexion
 - FIN : Demande la FIN de la connexion
- **Fenêtre** : Taille de fenêtre demandée, c'est-à-dire le nombre d'octets que le récepteur souhaite recevoir sans accusé de réception
- **Checksum** : Somme de contrôle calculée sur l'ensemble de l'en-tête TCP et des données, mais aussi sur un pseudo en-tête (extrait de l'en-tête IP). Ce pseudo en-tête comporte les adresses Internet source et destinataires, le type de protocole et la longueur du message TCP. Ceci protège TCP contre les erreurs de routage. Cette information sera véhiculée par IP, et est donnée comme argument par l'interface TCP/Réseau lors des appels d'IP par TCP. Le datagramme du pré en-tête TCP est décrit dans la figure suivante :

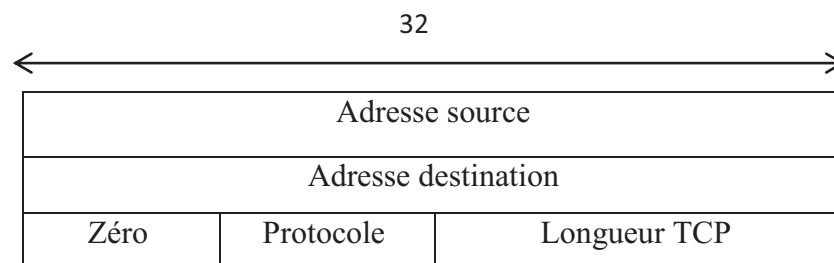


Fig.12: Datagramme du pré en-tête TCP

- **Pointeur de données urgentes** : Position relative des dernières données urgentes
- **Options** : Facultatives
- **Remplissage** : Zéros ajoutés pour aligner les champs suivants du paquet sur 32 bits, si nécessaire
- **Données** : Séquences d'octets transmis par l'application. [3] [5] [6]

4.7. Le protocole HTTP (Hyper Text Transfer Protocol) :

Le protocole HTTP (Hyper text Transfert Protocol) est l'un des plus utilisés sur Internet car il concerne le World Wide Web. Ce protocole décrit de quelle façon un navigateur peut interroger un serveur Web, et lui demander le contenu d'une URL (Uniform Resource Location). Ce protocole utilise TCP. La communication s'effectue en deux temps : le client (un navigateur par exemple) envoie une requête Http à un serveur, qui lui répond. Le port utilisé par le protocole Http est le port 80.

4.7.1. Communication entre navigateur et serveur :

La communication entre le navigateur et le serveur se déroule en deux temps:

- Le navigateur effectue une **requête http**.
- Le serveur traite la requête puis envoie une **réponse http**.

La figure suivante décrit le processus correspondant.

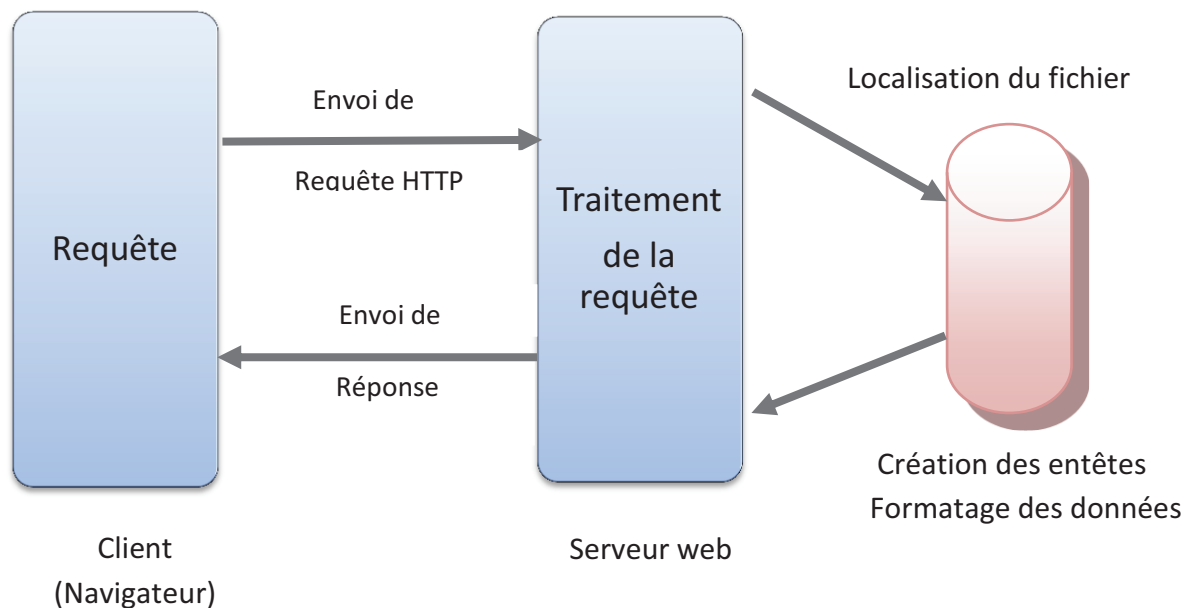


Fig.13: Communication entre client et serveur



5. Etude de cas du la pile TCPIP de Microchip:

Les implémentations du service TCP/IP destinées aux systèmes embarqués sont très nombreuses, elles ciblent différentes plateformes matérielles et supportent des fonctionnalités variées. Parmi les piles les plus complètes on cite: μ C/TCP-IP , openTCP, lwIP, Microchip TCP/IP, NicheLite TCP/IP, CMX MicroNet, TI MSP430 TCP/IP etc...

Pour la réalisation de notre projet nous avons utilisé la pile TCP/IP de Microchip qui est d'abord disponible, très documentée et contient tous les protocoles utiles et nécessaires pour l'élaboration d'un projet d'un système embarqué interagissant avec le réseau informatique.

5.1. Présentation de la pile TCP/IP de Microchip :

La société Microchip propose une pile logicielle TCP/IP libre de droits et optimisée pour les PIC18, PIC24, PIC32, et les dsPIC. Cette pile est une suite des programmes qui fournissent des services à toutes les applications basées sur TCP/IP. Les utilisateurs n'ont pas besoin de connaître les complexités et caractéristiques de la pile TCP/IP afin de l'employer.

Certaines couches de TCP/IP sont dites « vivent », elles réagissent automatiquement quand un service est demandé ou lors de l'arrivée de paquet. La pile est de conception modulaire, elle est écrite en langage C.

La pile TCP/IP de Microchip est considérée comme un portail qui permet l'échange de données entre la plateforme matérielle et l'application (sur PC) via le réseau informatique. Elle comprend des modules pour plusieurs couches applicatives couramment utilisés, y compris le protocole HTTP pour servir des pages web, SMTP pour l'envoi d'e-mails, SNMP pour fournir l'état et de contrôle, Telnet, TFTP et bien plus encore. En outre, la pile comprend l'implémentation légère et de haute performance de TCP et UDP de la couche transport, ainsi que d'autres modules de soutien tels que IP, ICMP, DHCP, ARP et DNS.



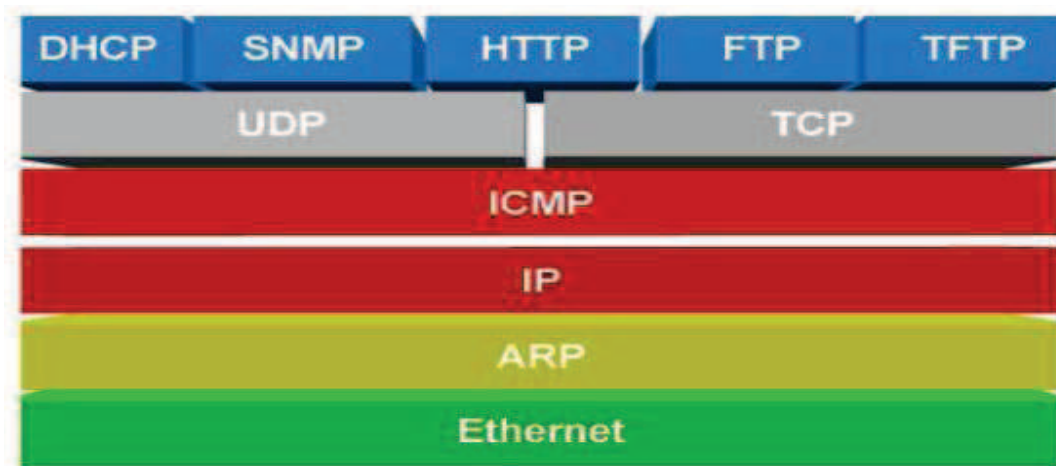


Fig.14 : Les protocoles supportés par la pile TCP/IP de Microchip

5.2. L'architecture de la pile TCP/IP de Microchip :

Comme le modèle de référence TCP/IP, la pile TCP/IP de Microchip divise la pile TCP/IP en couches multiples Comme le montre la figure ci-dessous.

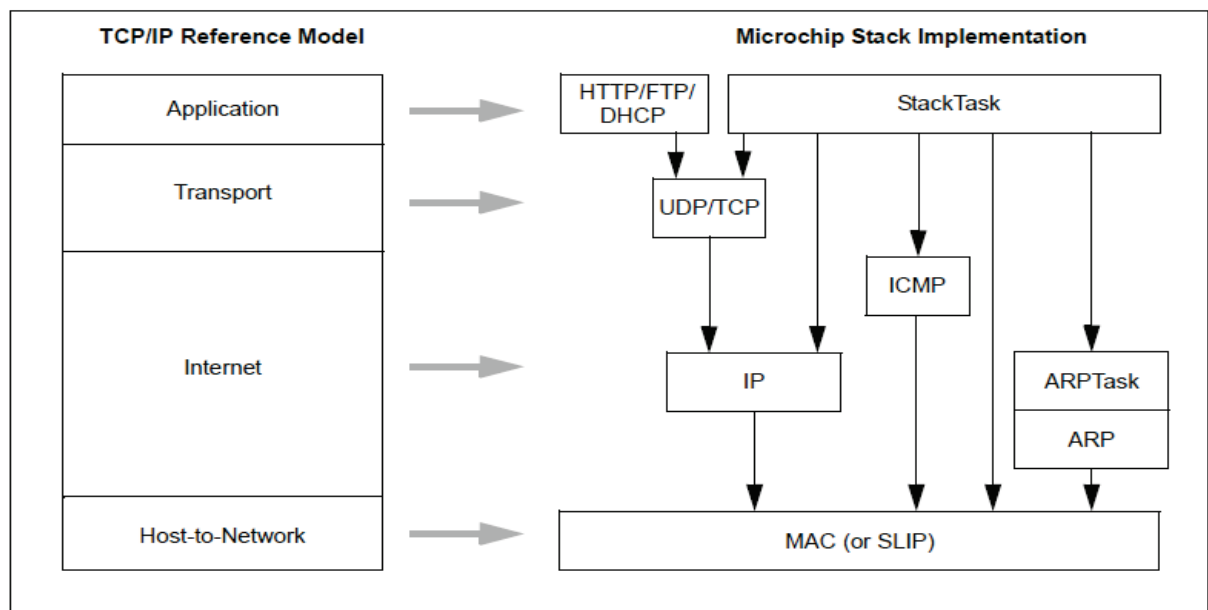


Fig.15 : la pile TCP/IP de Microchip vs modèle de référence TCP/IP.

Le code d'application de chaque couche est trouvé dans un fichier source séparé, tandis que les services et les API (Application Programming Interfaces) sont définis par des entêtes des fichiers.



La pile TCP/IP de Microchip repose sur la fonction « StackTask ». Cette fonction doit être exécutée en permanence pour que la pile soit fonctionnelle. Si un système d'exploitation temps-réel RTOS est utilisé, une tâche doit être créée et réservée uniquement pour cette fonction. Un départ supplémentaire majeur de la mise en œuvre de la pile TCP/IP traditionnelle est l'ajout de deux nouveaux modules : "StackTask" et "ARPTask".

"StackTask" gère les opérations de la pile et l'ensemble de ses modules, tandis que "ARPTask" gère les services de la couche Address Resolution Protocol (ARP).

Comme indiqué précédemment, la pile TCP/IP de Microchip est une pile où certains de ses couches doivent être en mesure d'effectuer certaines opérations programmées de manière asynchrone. Pour être en mesure de répondre à cette exigence tout en restant relativement indépendant de l'application principale en utilisant le service "StackTask" et "ARPTask". La pile TCP / IP Microchip utilise une technique bien connue appelée *multitâche coopératif*.

Dans un système multitâche coopératif, il n'y a plus d'une tâche, chaque un effectue son tâche et retourne son contrôle afin que la prochaine tâche peut effectuer son travail. Dans ce contexte, "StackTask" et "ARPTask" sont des tâches de coopération.

Notez que la pile TCP/ IP de Microchip ne pas mettre en œuvre tous les protocoles qui sont normalement présents dans la pile TCP / IP .Microchip permet d'intégrer des protocoles additionnels sur la base de cette pile.

5.3. Configuration de la pile TCP/IP de Microchip:

La réalisation de Multitâche coopératif permet à l'application principale de l'utilisateur d'effectuer son propre code, sans avoir à gérer la pile TCP/IP ainsi que cette fonctionnalité signifie que toutes les applications utilisant la pile TCP/IP de Microchip doivent également être écrites en mode multitâche coopératif. En plus de la conception multitâche coopératif, l'utilisateur doit d'abord comprendre certains détails de la configuration de base.

Pour faciliter le processus de configuration, la pile utilise la macro « defines ». Pour activer, désactiver ou définir un paramètre particulier.

La pile TCP/IP de Microchip contient des fichiers de base qui doivent être inclus dans tous les projets qui utilisent cette pile ils sont les suivent :



- Un fichier Main.c c'est le fichier qui contient le programme principal de l'application.
- le fichier ARP.c et ARP.h sont utilisés par la pile pour trouver l'adresse MAC associée à une adresse IP donnée.
- Le fichier Delay.c et Delay.h sont utilisés pour fournir des retards pour certaines fonctions de la pile
- Les fichiers de la couche physique comme ENC28J60.c, ENC28J60.h ENC24J600, ETH97J60.c .sont utilisés pour activer une couche physique spécifiée.
- Helpers.c et Helpers.h : Ces fichiers contiennent des fonctions auxiliaires utilisés pour des tâches diverses de la pile.
- IP.c et IP.h : Ces fichiers fournissent des fonctionnalités de la couche Internet de la pile.
- StackTsk.c et StackTsk.h : Ces fichiers contiennent le code pour initialiser la pile et d'effectuer des rappels qui maintiennent la pile passé.
- Tick.c et Tick.h : Ces fichiers mettre en place un timer tick qui est utilisé pour mettre en œuvre certaines fonctionnalités de synchronisation.
- HardwareProfile.h : Ce fichier de configuration est utilisé pour mettre en place des options matérielles.
- TCPIPConfig.h : Ce fichier de configuration est utilisé pour configurer les options du firmware.
- MAC.h : Ce fichier d'en-tête fournit des macros et des structures relatives à la couche MAC du matériel.
- TCPIP.h : Ceci est le fichier primaire incluse pour la pile.

On peut choisir d'inclure des fichiers supplémentaires pour supporter des protocoles et des fonctionnalités supplémentaires.

Nous verrons dans le chapitre III comment configurer cette pile.



5.4. Fonctionnement de la pile TCP/IP de Microchip :

5.4.1. Initialisation de la pile :

Premièrement on va commencer par l'initialisation de matériel. Cela inclut les oscillateurs, LED, LCD, tout SPI que vous utilisez pour contrôler la puce matériel MAC/ PHY.

Ensuite, appeler les fonctions d'initialisation du matériel. La fonction `TickInit ()` doit être appelé en premier, il va initialiser le timer tick qui gère le timing de la pile.

Ensuite, appelez les fonctions d'initialisation supplémentaires qui nécessitent l'initialisation du matériel. Par exemple, la fonction `MPFSInit()` aura besoin d'initialiser un port SPI pour communiquer avec un stockage de la mémoire dispositif pour stocker les pages web.

Enfin, vous pouvez initialiser la pile TCP/IP en appelant la fonction `StackInit()`. Cette fonction appelle automatiquement les fonctions d'initialisation pour les autres protocoles de la pile si elles ont été activées dans `TCPIPConfig.h`. Par exemple `TCPInit ()` pour Le protocole TCP, `HTTPInit ()` pour http, `FTPInit ()` pour FTP.

5.5. La boucle Main (super boucle) :

Une fois que la pile a été initialisée, il doit entrer dans une boucle infinie qui se chargera les tâches de l'application. Dans cette boucle, il y a deux fonctions principales qu'il doit appeler régulièrement: `StackTask` et `StackApplications`.

La fonction `StackTask` va affecter toutes les opérations programmées que la pile nécessite, et gérer la transmission et la réception de paquets de données.

La fonction `StackApplications` appellera les modules actifs dans l'application. Par exemple, si une application utilise un serveur HTTP, `StackApplications` appelle automatiquement la fonction `HTTPServer()` pour traiter toute http tâches qui sont en attente.

On à faire un appel à `StackTask ()` pour chaque appel de `StackApplications()` mais aussi on n'est pas obligé d'appeler ces fonctions avec une certaine fréquence spécifique.

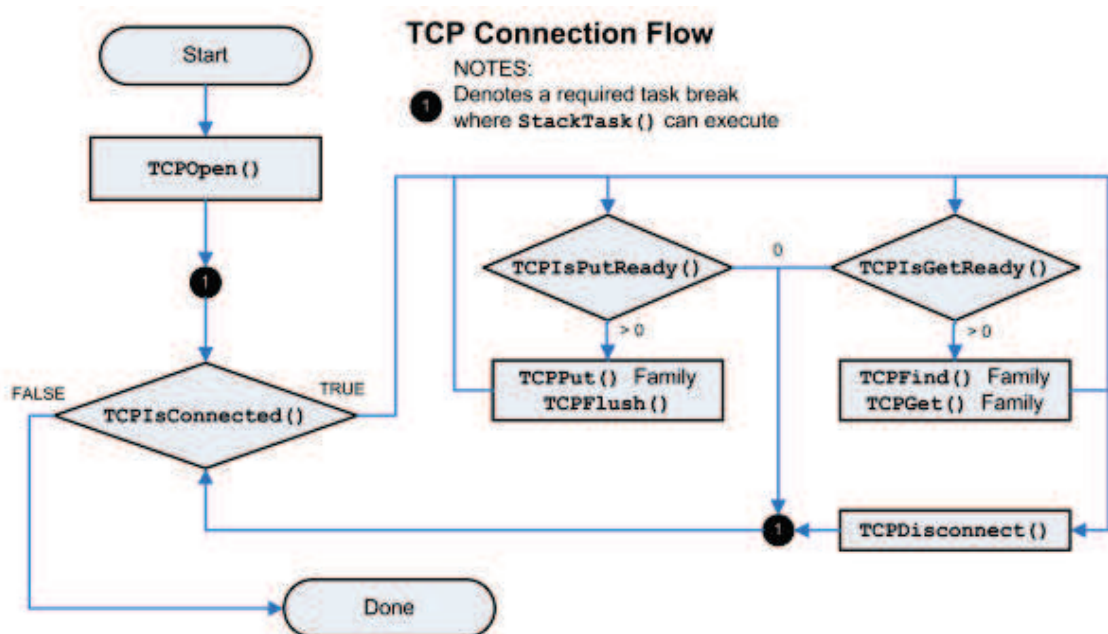


5.6. Le Multitâche coopératif :

Comme nous l'avons indiqué précédemment, la pile TCP/IP utilise une approche multitâche coopérative, elle doit faire des appels périodiquement à des fonctions spéciales pour transmettre/recevoir des paquets et de maintenir la fonctionnalité du protocole.

L'implémentation d'un protocole ou d'une tâche d'application se fait par multiples étapes, il peut être avantageux de les répartir entre les États pour donner un temps partagé. Ensuite la pile utilise une variable globale ou statique pour suivre l'état de protocole ou tâche.

Le diagramme suivant montre comment la pile TCP/IP de Microchip est implémentée le protocole TCP par la stratégie de multiple étapes:



❶ : conception d'une interruption nécessaire pour exécuter la fonction `StackTask()`.



5. Conclusion :

Dans ce chapitre, Nous avons présenté, le principe d'un modèle en couches et donné l'exemple incontournable du modèle OSI. Puis, nous avons présenté une description générale sur le modèle en couches TCP/IP.

Ensuite nous avons énoncé les différents protocoles du modèle TCP/IP et quelques notions de base sur ce dernier.

Dans une autre section nous présentons la pile TCP/IP de Microchip, l'architecture de cette pile, la manière de son fonctionnement avec plus de détail, et ce pour mieux comprendre la suite du travail qui est principalement basée sur cette pile.



Chapitre III

Etude de

l'environnement

matériel

1. Introduction :

L'objectif de cette partie est d'étudier la pile TCP/IP de Microchip et l'embarquer ensuite sur la plateforme matérielle dsPICPRO4, qui est autour du circuit microcontrôleur dsPIC30F6014A et le circuit ENC28J60, qui gère toute la couche physique et MAC du modèle OSI.

Pour commencer, nous allons donc faire une étude détaillée sur matériel utilisé et nous allons établir la configuration de la pile TCP/IP de Microchip sur ce matériel et ce en utilisant les outils de développement qu'ils lui sont relatifs. La pile est ensuite embarquée sur la carte.

2. Présentation de l'environnement matériel :

2.1. Description de La carte de développement dsPICPRO4 :

La société allemande MikroElektronika offre une carte de développement qui permet de créer des applications communiquant via l'interface Ethernet. Cette carte dispose d'un programmeur à connexion USB intégré qui permet de télécharger les programmes développés en assembleur ou avec n'importe quel type de compilateur, cette carte contient le microcontrôleur dsPIC30F6014A de Microchip qui intègre un générateur de signaux 16 bits, 144 KO de mémoire Flash et 8 KO de RAM. [7]

La carte dsPICPRO4 est munie de tous les composants nécessaires pour l'accomplissement de notre projet dans sa totalité. Le microcontrôleur dsPIC30F6014A offre suffisamment de mémoire Flash pour stocker la pile TCP/IP. De plus, il est relié à une connexion RJ-45 pour accéder à un réseau avec interface Ethernet 10Base T de type « ENC28J60 Stand-Alone Ethernet Controller with SPI Interface » avec une multitude de fonctionnalités pour gérer la plupart des exigences du protocole réseau. Cette interface est connectée directement au microcontrôleur avec une interface standard SPI. La figure 16 ci-dessous illustre la carte de développement dsPICPRO4 :



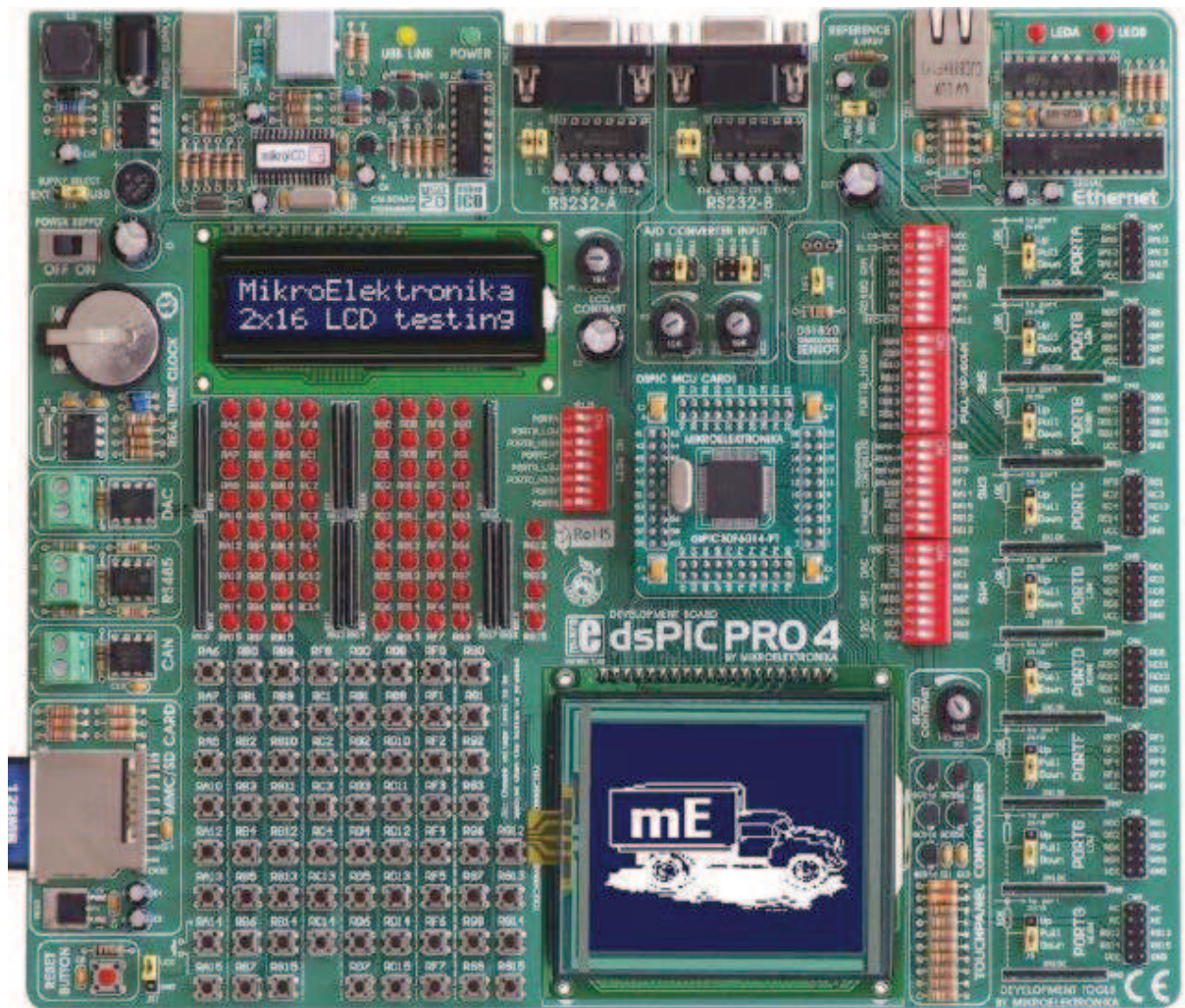


Fig.16 : la carte dsPICPRO4

La carte sur laquelle nous travaillons est composée des éléments suivants :



Emplacement pour DS1820 (option)



Alimentation externe possible



Connecteur pour ICD2 (option)



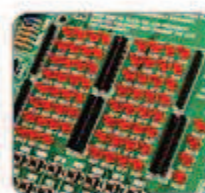
Programmeur USB intégré



Double port série



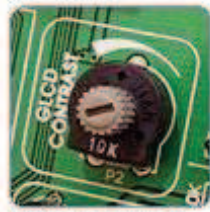
Liaison Ethernet à base de ENC28J60



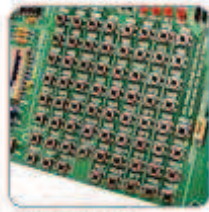
67 Leds intégrée à la platine



Référence 5 V ou 4,096 V



Réglage contraste des LCD



67 BP dispos sur la platine



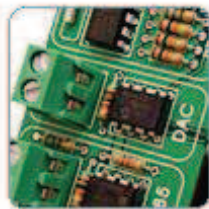
Sélection +VCC ou GND sur BP



Alim. externe ou via USB



Configurations diverses via jumpers



Convertisseur "N/A" 12 bits intégré



Sélection pull-up pull-down sur port



Leds de signalisation



CPU 64 ou 80 broches (dsPIC30F6014A livré)



Connecteur pour carte SD™



BP Reset intégré



Potentiomètres sur conv. "A/N"

L'afficheur LCD permet d'afficher des informations intéressantes. Les boutons sont utiles pour simuler un envoi de données. La connexion RJ-11 permet de brancher le module MPLAB ICD 2 afin de programmer le microcontrôleur sous l'environnement MPLAB IDE.

2.2. Les Microcontrôleurs :

Un microcontrôleur est un composant programmable capable d'exécuter un programme et qui possède des circuits d'interface intégrés avec le monde extérieur. Il regroupe dans un seul boîtier compact un processeur de calcul, de la mémoire vive (RAM), la mémoire permanente (FLASH, EEPROM) et des périphériques.

Les microcontrôleurs se caractérisent par un plus haut degré d'intégration, une plus faible consommation électrique, une vitesse de fonctionnement plus faible (de quelques mégahertz jusqu'à plus d'un gigahertz) et un coût réduit par rapport aux microprocesseurs polyvalents utilisés dans les ordinateurs personnels . [8]



Il existe plusieurs modèles des microcontrôleurs quelques exemples courants : les 8051 d'Intel, les 68HC11 de Motorola et les PIC de Microchip.

2.2.1. Définition d'un PIC :

Un PIC (Nom déposé par Microchip) n'est rien d'autre qu'un microcontrôleur, c'est à dire une unité de traitement de l'information de type Microprocesseur à laquelle on a ajouté des périphériques internes permettant de réaliser des montages sans nécessiter l'ajout des composants externes.

Les Pics sont alors des composants dits RISC (Reduced Instructions Set Computer, ou composant à jeu d'instruction réduits). Plus facile et plus rapide en est le décodage, et plus vite le composant fonctionne. [8]

2.2.2. Définition d'un dsPIC :

Les DSPIC (Digital Signal Processing Integratedcircuits) sont des microcontrôleurs 16 bits fabriqués par Microchip, ils correspondent aux familles 30Fxxxx et 33Fxxxx. Ces microcontrôleur possèdent des instructions de type DSP (Digital Signal Processor) et peuvent donc convenir pour des applications de traitement du signal. Leur architecture est très différente des célèbres microcontrôleurs PIC (qui sont de type 8 bits). On peut dire globalement qu'ils offrent des performances très intéressantes aussi bien en tant que DSP qu'en tant que microcontrôleurs d'usage général.

2.3. Interface Série pour périphérique(SPI) :

L'interface Série pour périphériques (SPI) est un protocole de communication série synchrone simple utilisé par les microcontrôleurs pour communiquer avec un ou plusieurs composants périphériques rapidement sur de courtes distances. Les composants communiquent selon un schéma maître-esclaves, où le maître s'occupe totalement de la communication. Ce protocole peut aussi être utilisé pour des communications entre deux microcontrôleurs.



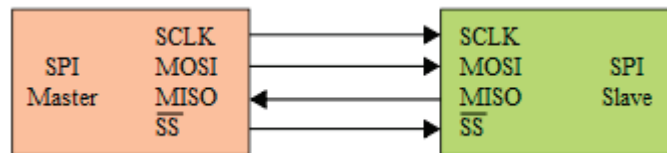


Fig.17 : Liaison SPI avec un maître et un esclave

Avec une connexion SPI, il y a toujours un composant maître, habituellement un microcontrôleur lequel commande le ou les composants périphériques. Typiquement, il y a 3 lignes communes à tous les composants et qui les relient entre eux :

- La ligne Maître entrant - Esclave sortant (ou MISO pour *Master In Slave Out*) : c'est la ligne utilisée pour envoyer des données depuis le périphérique vers le maître.
- La ligne Maître sortant - Esclave entrant (ou MOSI pour *Master Out Slave In*) : c'est la ligne utilisée pour envoyer des données du maître vers le périphérique.
- La ligne du signal d'horloge (ou SCLK pour *Serial Clock*) : cette ligne transmet les impulsions de signal d'horloge générées par le composant maître et qui synchronisent la transmission des données.

A ces trois lignes communes, s'ajoutent une ligne de sélection par périphérique :

- La ligne de sélection de l'esclave (ou SS pour *Slave Select*) : chaque composant périphérique dispose d'une broche de sélection (active sur le niveau bas) que le composant maître peut utiliser pour activer ou désactiver le composant périphérique voulu. [9]

2.3.1. Fonctionnement de protocole SPI :

Une transmission SPI entre un maître et un esclave est faite comme suit :

- Le maître génère l'horloge et sélectionne l'esclave avec qui il veut communiquer.
- L'esclave répond aux requêtes du maître.

À chaque coup d'horloge le maître et l'esclave s'échangent un bit. Après huit coups d'horloges le maître transmet un octet à l'esclave. [10]

Dans notre projet nous avons utilisé le protocole SPI pour la communication entre le dsPIC30F6014A et l'interface réseau ENC28J60 comme indiqué dans la figure suivante :



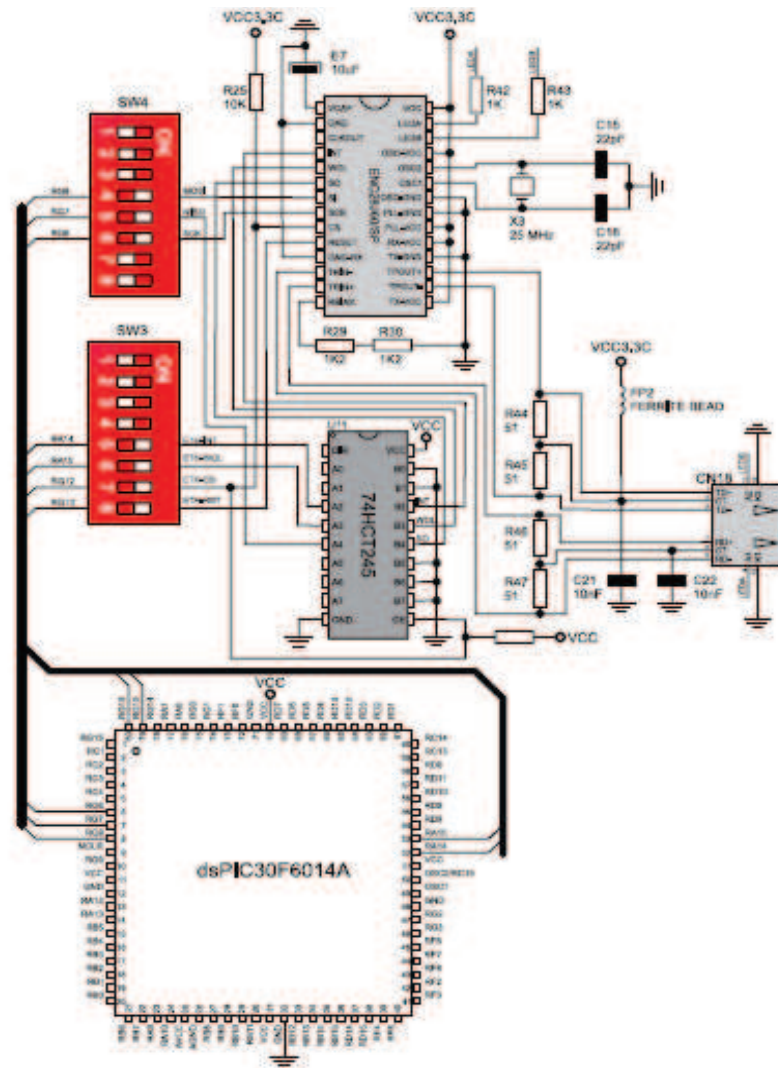


Fig.18: Diagramme de circuit Ethernet.

3. Programmation de la carte dsPICPRO4 :

3.1. Moyens et outils utilisés :

3.1.1. Environnement de développement :

Pour la configuration de la pile TCP/ IP de Microchip nous avons utilisé MPLAB IDE V8. MPLAB est un Environnement de Développement Intégré (IDE) gratuit fourni par la société Microchip. Il permet le développement logiciel des microcontrôleurs PIC et les contrôleurs de signal numériques dsPIC, et ensuite il permet de tester l'application avec différents logiciels et composants matériels.

Voici un aperçu parapluie de MPLAB IDE et les catégories d'outils logiciels et matériels qui y sont intégrés:



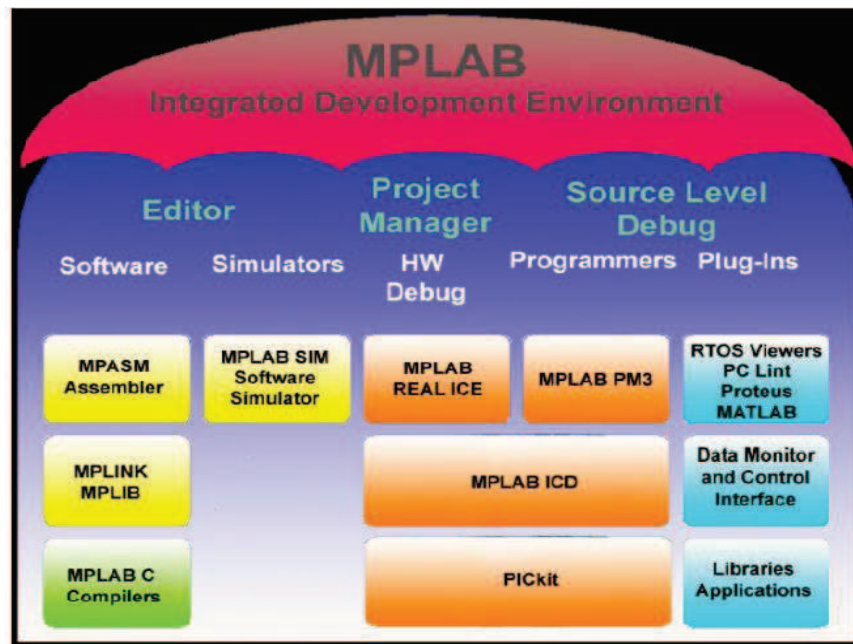


Fig.19:les outils logiciels et matériels de MPLAB

Construit dans MPLAB IDE, un éditeur de programmeur, un gestionnaire de projet et de puissants outils de débogage. D'autres outils libres sont des éléments standards de MPLAB IDE, comme le montrent les blocs jaunes, assembleur MPASM, MPLINK / Mplib éditeur de liens et de bibliothèque, ainsi que MPLAB SIM, le simulateur logiciel. Les compilateurs C MPLAB (bloc vert) sont vendus séparément et fonctionnent de façon transparente au sein de cet environnement. Les outils matériels sont indiqués en orange et comprennent les programmeurs et les débogueurs ainsi que des émulateurs de circuit complexes. Notez que MPLAB ICD et PICKits permettent d'avoir des attributs de débogage à faible coût, Emulation / analyse et de programmation de périphériques dans un seul outil. Les plug-ins (en bleu) comprennent des tiers partis des compilateurs C, systèmes d'exploitation temps réel, Surveillance de données et contrôle d'interface graphique, ainsi que des bibliothèques qui peuvent être liés aux applications spécifiques.[11]



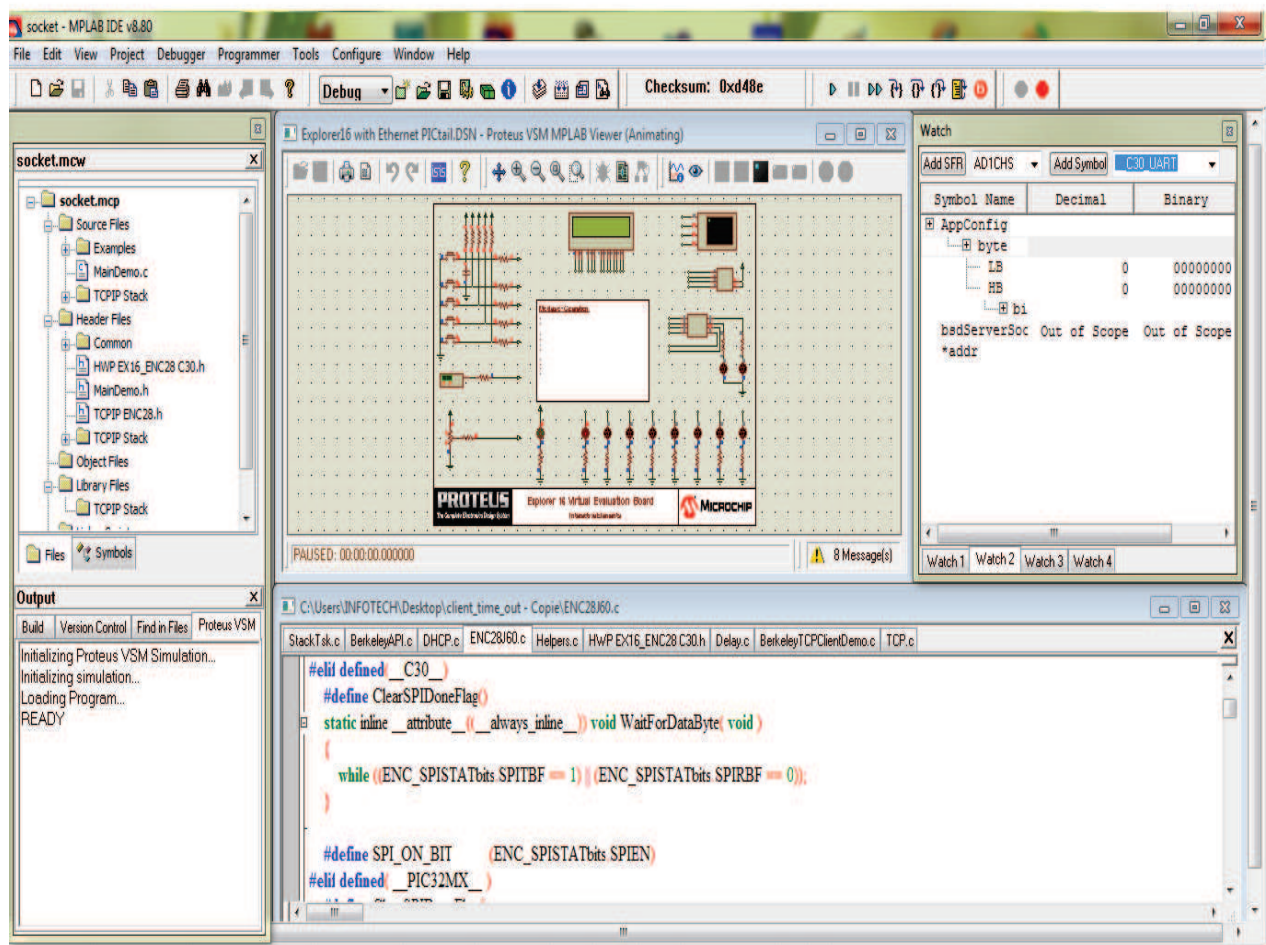


Fig.20: Interface MPLAB V8.80

3.1.2. Le compilateur MPLAB C30 :

Le compilateur C pour PIC24 et dsPIC connu sous le nom MPLAB C30 est plein de fonctionnalités dédiées aux contrôleurs 16 Bits de Microchip qui sont PIC24, dsPIC30F et dsPIC33F.

MPLAB C30 qui dérive du projet GCC de GNU est un compilateur complètement intégré dans l'Environnement MPLAB (IDE), ce qui permet le débogage au niveau source avec le MPLAB REAL ICE émulateur, MPLAB ICD 2 In-Circuit Debugger et MPLAB SIM Simulator.

3.1.3. L'outil de simulation Proteus :

Proteus est une suite logicielle destinée à l'électronique. Développé par la société LabcenterElectronics, les services inclus dans Proteus permettent la conception assistée par ordinateur (CAO) dans le domaine électronique

Proteus possède d'autres avantages :

- Pack contenant des logiciels faciles et rapides à comprendre et utiliser
- Le support technique est performant
- L'outil de création de prototype virtuel permet de réduire les coûts matériel et logiciel lors de la conception d'un projet.

ISIS : le logiciel ISIS de Proteus est principalement connu pour éditer des schémas électriques. Par ailleurs, le logiciel permet également de simuler ces schémas et en particulier les circuits dsPIC et le composant ENC28J60 ce qui permet de déceler les éventuelles erreurs durant la phase de programmation du firmware de la carte de développement. [12]

3.2. Configuration de la pile TCP/IP de Microchip :

Pour pouvoir programmer le microcontrôleur dsPIC30F6014A de la carte dsPICPRO4, On doit relier cette dernière avec le module MPLAB ICD 2 .Par la suite, il faut connecter MPLAB ICD 2 à l'ordinateur à l'aide d'un câble USB. Pour poursuivre, il suffit d'exécuter la pile TCP/IP Microchip sous le programme MPLAB IDE.

La pile TCPIP de Microchip est présentée sous forme d'un projet source ouvert. Il est donc possible de voir en entier les programmes de la majorité des protocoles dont TCP et IP. De plus, pour faciliter la compréhension de chaque fonction établie par Microchip, il y a un document complet qui explique en détail chacune d'entre elles. Ce document s'intitule *Microchip TCP/IP StackHelp*.

Les fichiers de la pile TCPIP de Microchip ne doivent pas être modifiés, en particulier les fichiers suivants qui doivent être adaptés à l'application (matériels et logiciels) :

- HardwareProfile.h : description du matériel.
- TCPIPConfig.h : description des applications TCPIP activées.
- BerkeleyTCPCClientDemo.c : application de démonstration de socket TCP client.
- MainDemo.c : un exemple complet d'application.



3.2.1. Configuration de la pile TCP/IP à l'aide du fichier HardwareProfile.h:

Le fichier HardwareProfile.h décrit le matériel, il a été modifié et complété pour l'utilisation d'un dsPIC30F6014A sur la carte dsPICPRO4.

Les lignes suivantes permettent de définir la carte utilisée :

```
#if defined (YOUR_BOARD)
#include "Configs/HWP YOUR_BOARD.h"
```

Dans notre projet nous avons remplacé ses lignes par :

```
#define dsPICPRO4
#if defined (dsPICPRO4)
#include "HWP dsPIC30f_ENC28.h"
```

Dans le fichier HWP dsPIC30f_ENC28.h on a décrit le matériel de la carte dsPICPRO4 :

```
#define GetSystemClock()      (10000000ul)    // 10MHz
#define GetInstructionClock() (GetSystemClock()/2)
#define GetPeripheralClock() (GetSystemClock()/2)

// ENC28J60 I/O pins

#define ENC_CS_TRIS           (TRISGbits.TRISG12)
#define ENC_CS_IO             (LATGbits.LATG12)
#define ENC_SPI_IF            IFS2bits.SPI2IF
#define ENC_SSPBUF            SPI2BUF
#define ENC_SPISTAT           SPI2STAT
#define ENC_SPISTATbits       SPI2STATbits
#define ENC_SPICON1           SPI2CON
#define ENC_SPICON1bits       SPI2CONbits
#define ENC_SPICON2           SPI2CON
```

3.2.2. Configuration de la pile TCP/IP à l'aide du fichier TCPIPConfig.h :

Le fichier TCPIPConfig.h définit les services TCPIP utilisés, toutes les fonctions sont actives par défaut, il suffit de commenter celles qui sont inutiles. Ainsi la figure 21 démontre les fonctions utilisées.



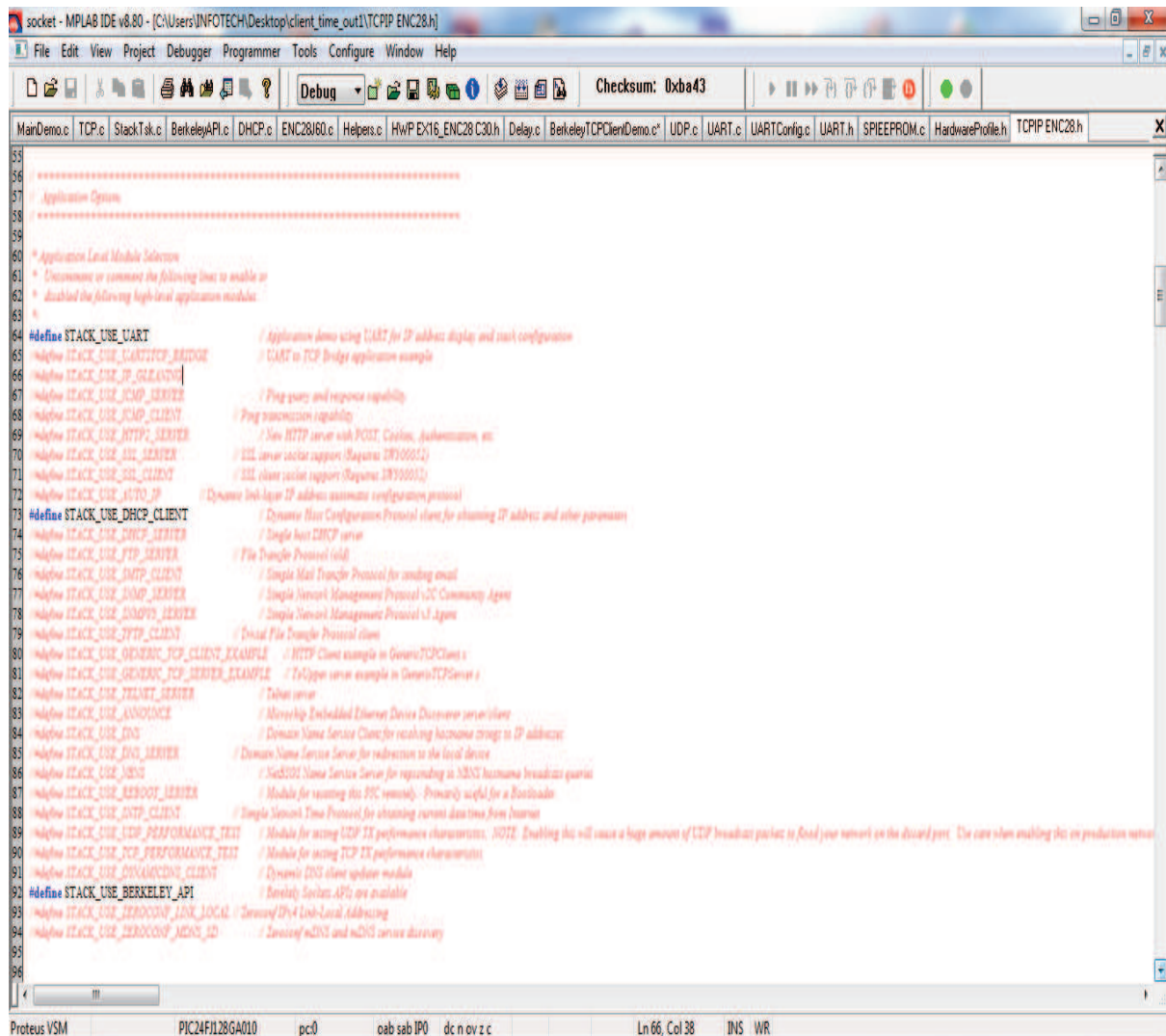


Fig.21 : Configuration du protocole utilisé par la pile TCP/IP.

- **STACK_USE_BERKELEY_API** : Cette fonction est la plus utilisée Pour le cas de ce projet. Elle inclut le fichier où le microcontrôleur peut envoyer des données par socket TCP au serveur. Le nom de ce fichier est BerkeleyTCPClientDemo.c.

Les lignes suivantes sont configurées en fonction du réseau (IP par défaut, masque de sous réseau, serveur DNS etc....)

```
#define MY_DEFAULT_HOST_NAME    "dsPICPRO4"
#define MY_DEFAULT_MAC_BYTE1    (0x00)
#define MY_DEFAULT_MAC_BYTE2    (0x04)
#define MY_DEFAULT_MAC_BYTE3    (0xA3)
#define MY_DEFAULT_MAC_BYTE4    (0x00)
#define MY_DEFAULT_MAC_BYTE5    (0x01)
```

```

#define MY_DEFAULT_MAC_BYTE6      (0x00)

#define MY_DEFAULT_IP_ADDR_BYTE1  (192ul)
#define MY_DEFAULT_IP_ADDR_BYTE2  (168ul)
#define MY_DEFAULT_IP_ADDR_BYTE3  (204ul)
#define MY_DEFAULT_IP_ADDR_BYTE4  (15ul)


#define MY_DEFAULT_MASK_BYTE1      (255ul)
#define MY_DEFAULT_MASK_BYTE2      (255ul)
#define MY_DEFAULT_MASK_BYTE3      (255ul)
#define MY_DEFAULT_MASK_BYTE4      (0ul)


#define MY_DEFAULT_GATE_BYTE1      (192ul)
#define MY_DEFAULT_GATE_BYTE2      (168ul)
#define MY_DEFAULT_GATE_BYTE3      (204ul)
#define MY_DEFAULT_GATE_BYTE4      (15ul)


#define MY_DEFAULT_PRIMARY_DNS_BYTE1  (192ul)
#define MY_DEFAULT_PRIMARY_DNS_BYTE2  (168ul)
#define MY_DEFAULT_PRIMARY_DNS_BYTE3  (204ul)
#define MY_DEFAULT_PRIMARY_DNS_BYTE4  (15ul)
#define MY_DEFAULT_SECONDARY_DNS_BYTE1  (0ul)
#define MY_DEFAULT_SECONDARY_DNS_BYTE2  (0ul)
#define MY_DEFAULT_SECONDARY_DNS_BYTE3  (0ul)

```

3.2.3. Programmation du fichier BerkeleyTCPClientDemo.c :

À l'origine, le fichier BerkeleyTCPClientDemo.c était un exemple de requête GET effectué par Microchip. Ce fichier a été modifié selon les besoins du notre projet.

3.2.4. Le fichier MainDemo.c :

MainDemo.c est une application complète de serveur WEB, nous avons modifié ce fichier selon les besoin de notre application comme le montre l'algorithme suivant :

3.2.4.1. Algorithme de maindemo.c :

1. La fonction *InitializeBoard()* initialise l'horloge, les E/S ,l'UART, autorise les interruptions.



2. La fonction *LCDInit()* Initialiser l'afficheur LCD.
 3. La fonction *TickInit()* initialise la base de temps de la machine d'état, utilise timer0 .
 4. La fonction *InitAppConfig()* initialise les adresses (IP, masque , DNS, routeur etc...) .
 5. La fonction *StackInit()* Initialise les services de TCP/IP(MAC, ARP, TCP).
 6. Début de la boucle While(1).
- 6.1. Clignotement de la LED0 chaque seconde.
- 6.2. la fonction *StackTask()* valide la prochaine tâche TCPIP à effectuer.
7. Exécute les services activés .
 8. Vérifie que l'adresse IP n'a pas changé (par DHCP).
 9. La fonction *BerkeleyTCPClientDemo()* c'est ici que l'on place le programme de l'application.
- Fin de la boucle While(1).

4. Conclusion :

Dans ce chapitre, nous avons pu identifier les besoins matériels et fonctionnels et les points qui permettront d'entamer la partie réalisation de notre système.

Dans le chapitre suivant, on représentera les fonctions du système à réaliser, et on abordera les divers diagrammes de la conception UML.



Chapitre IV

Conception et réalisation d'un système de contrôle d'accès en utilisant la pile TCP/IP

1. Introduction :

Après avoir achevé la partie configuration de la carte de développement, nous présentons dans ce chapitre la partie de conception et de réalisation d'un environnement de test et validation de notre système. Pour cela, nous avons développé un système de contrôle d'accès simple basé sur PC.

Le contrôle d'accès devient de plus en plus populaire dans beaucoup d'entreprises. Les systèmes de contrôle d'accès permettent de limiter l'accès aux locaux en fonction des heures et des personnes. Ils conservent une trace de tous les accès autorisés ou refusés.

Dans ce chapitre, nous présenterons en premier lieu, la description du système de contrôle d'accès mis en œuvre, le protocole d'échange des données utilisé entre le client et le serveur, l'environnement de travail, ainsi que les outils de développement utilisés.

En second lieu, nous présenterons la phase de conception du système en utilisant le formalisme UML. Enfin, nous décrirons les tests d'évaluation du système de contrôle d'accès par carte sans contacte.

2. Description du système de contrôle d'accès mise en œuvre :

Il s'agit dans ce travail à développer un système de contrôle d'accès simple qui permet de contrôler l'ouverture et la fermeture de portes. Il consiste à vérifier qu'une personne a bien les droits nécessaires pour accéder à un lieu, un bâtiment ou un local. Il permet d'organiser la circulation des personnes à l'intérieur d'un site et de gérer l'accessibilité des différentes zones de façon sélective.

Il se compose des cartes de contrôle en utilisant la carte dsPICPRO4, aux quelles sont raccordés des lecteurs de carte. Chaque carte d'identification est individuel ce qui consiste à vérifier qu'une personne est autorisée à accéder à une zone. Ainsi, L'accès est effectué comme suit :

- La carte d'identification est présentée devant le lecteur de carte.
- Le lecteur lit l'ID de la carte et le transmet à la carte de contrôle.
- La carte contrôle (client) formulé une requête d'autorisation d'accès et l'envoi via l'interface réseau (objet de test) à l'unité de traitement (serveur).



- Le serveur valide, en consultant la base de données, l'autorisation ou le refus d'accès en transmettant une réponse à la carte contrôle.
- La carte de contrôle déclenche l'ouverture de la porte en cas d'une réponse favorable.
- La carte contrôle procède, éventuellement en suite à la fermeture de la porte.

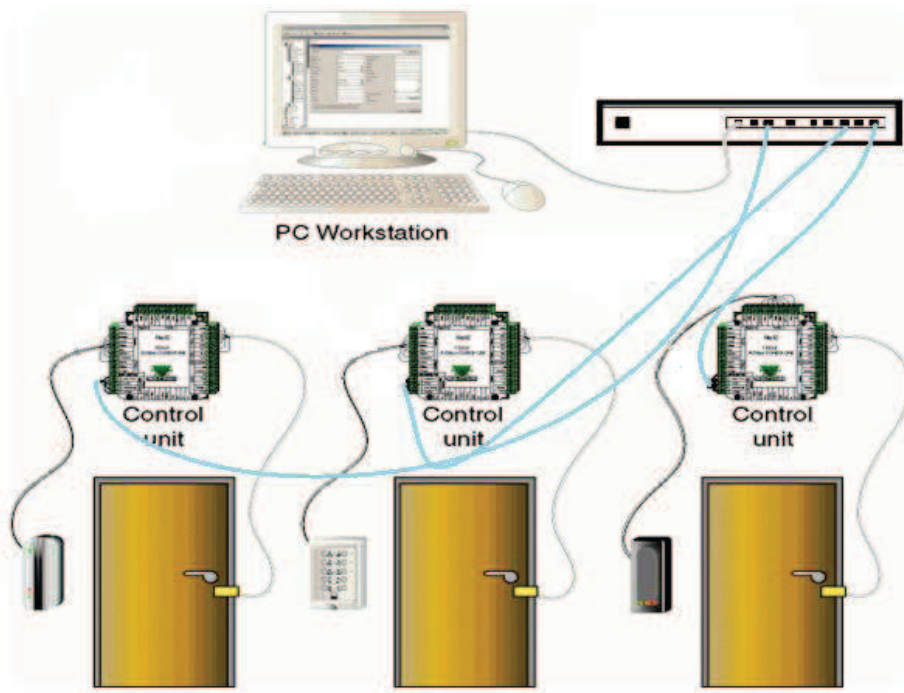


Fig.22 : l'architecture de système à réaliser

2.1. Protocole de communication utilisé :

Les systèmes de contrôle d'accès nécessitent des systèmes de sécurité pour le traitement efficace de communication. Il est donc nécessaire de trouver un équilibre contrôlé entre l'accès au réseau et la limitation des risques de sécurité.

Dans ce cadre, nous avons proposé un protocole de communication entre le client et le serveur. Les données sont envoyées selon la grammaire suivante :

Commande	Donnée à envoyer	Début de donnée		Code opération		Donnée à envoyer	
		caractère	signification	caractère	signification	caractère	signification
Envoyer le numéro De la porte	«Y!0!n!»	Y	Début de donnée	0	Numéro de l'opération	n	Numéro de la porte
Envoyer un ID	«Ç!1!m!»	Ç	Début de donnée	1	Numéro de l'opération	m	Mot de passe
Ouvrir la porte	«Ä!3!o!»	Ä	Début de donnée	3	Numéro de l'opération	o	Ouvrir (action)
Fermer la porte	«Ö!4!f!»	Ö	Début de donnée	4	Numéro de l'opération	f	Fermer (action)
Voir l'état d'une porte	«Ë!2!n!»	Ë	Début de donnée	2	Numéro de l'opération	n	Numéro de la porte
Bloquer une porte dans un état ouvert	«B!5!bo!»	B	Début de donnée	5	Numéro de l'opération	bo	Bloquer la porte (action)
Bloqué une porte dans un état fermé	«F!6!bf!»	F	Début de donnée	6	Numéro de l'opération	bf	Bloquer la porte (action)

3. Environnement de travail :

Ce travail a été réalisé sur un environnement de Windows 7. Le développement de cette application s'est fait en optant les outils suivant :

- L'utilisation de MyEclipse Bleu Edition 8.6 pour le développement de l'application en Java.
- L'SGBD Microsoft Access 2010.
- L'utilisation de l'Objecteering 6.1 pour la conception UML de l'application.

4. Conception du système :

Toute démarche de conception nécessite une modélisation .notre choix s'est porté sur le formalisme UML.

UML « Unified Modeling Language » est un langage utilisant une représentation graphique.

L'usage d'une représentation graphique est un complément excellent à celui de



représentations textuelles. Ainsi, le dessin permet bien souvent d'exprimer clairement ce qu'un texte exprime difficilement et un bon commentaire permet d'enrichir une figure.

Nous illustrerons notre conception par le développement du modèle dynamique, Nous présenterons le diagramme de cas d'utilisation, les diagrammes de séquence et le diagramme d'état transition.

4.1. Identification des acteurs

Un acteur représente l'abstraction d'un rôle joué par des entités externes. Un acteur peut Consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages éventuellement porteurs de données.

Les acteurs du notre système sont :

- **Administrateur** L'administrateur système visualise l'état du système et gère les portes.
- **Utilisateur** : demander l'accès par l'utilisation de carte sans contacte.

4.2. Diagramme de cas d'utilisation :

Les cas d'utilisation représentent l'ensemble des fonctionnalités du système à étudier. Ils permettent d'identifier toutes les fonctions du système.

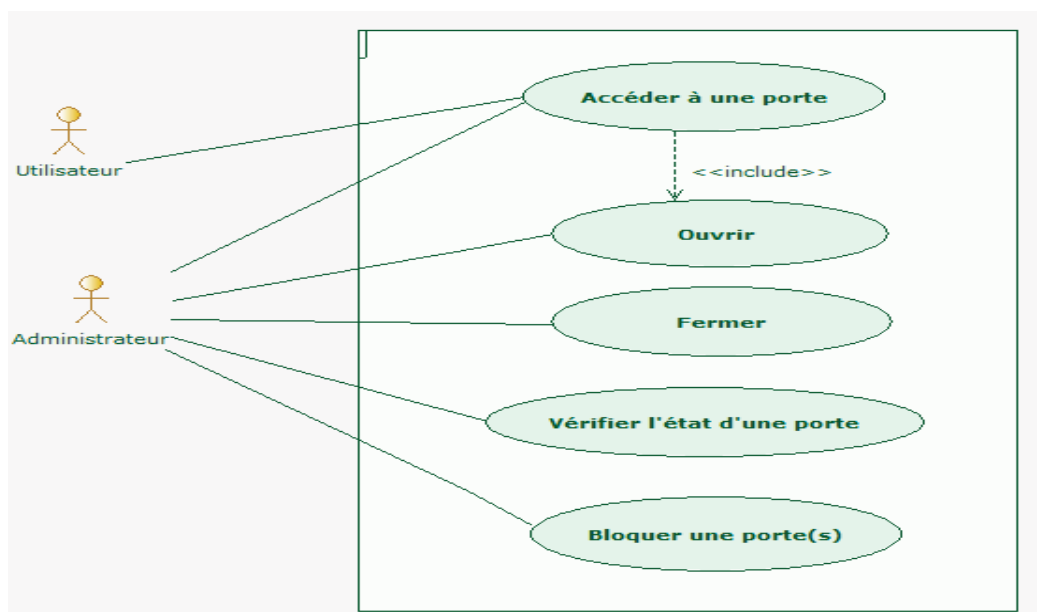


Fig.23 : Diagramme de cas d'utilisation



4.3. Diagramme de séquence :

Le diagramme de séquence permet de représenter les scénarios d'un cas d'utilisation. Un scénario est une instance d'un cas d'utilisation. un message reçu par un objet déclenche l'exécution d'une opération et renvoie un message qui correspond au résultat de l'opération.

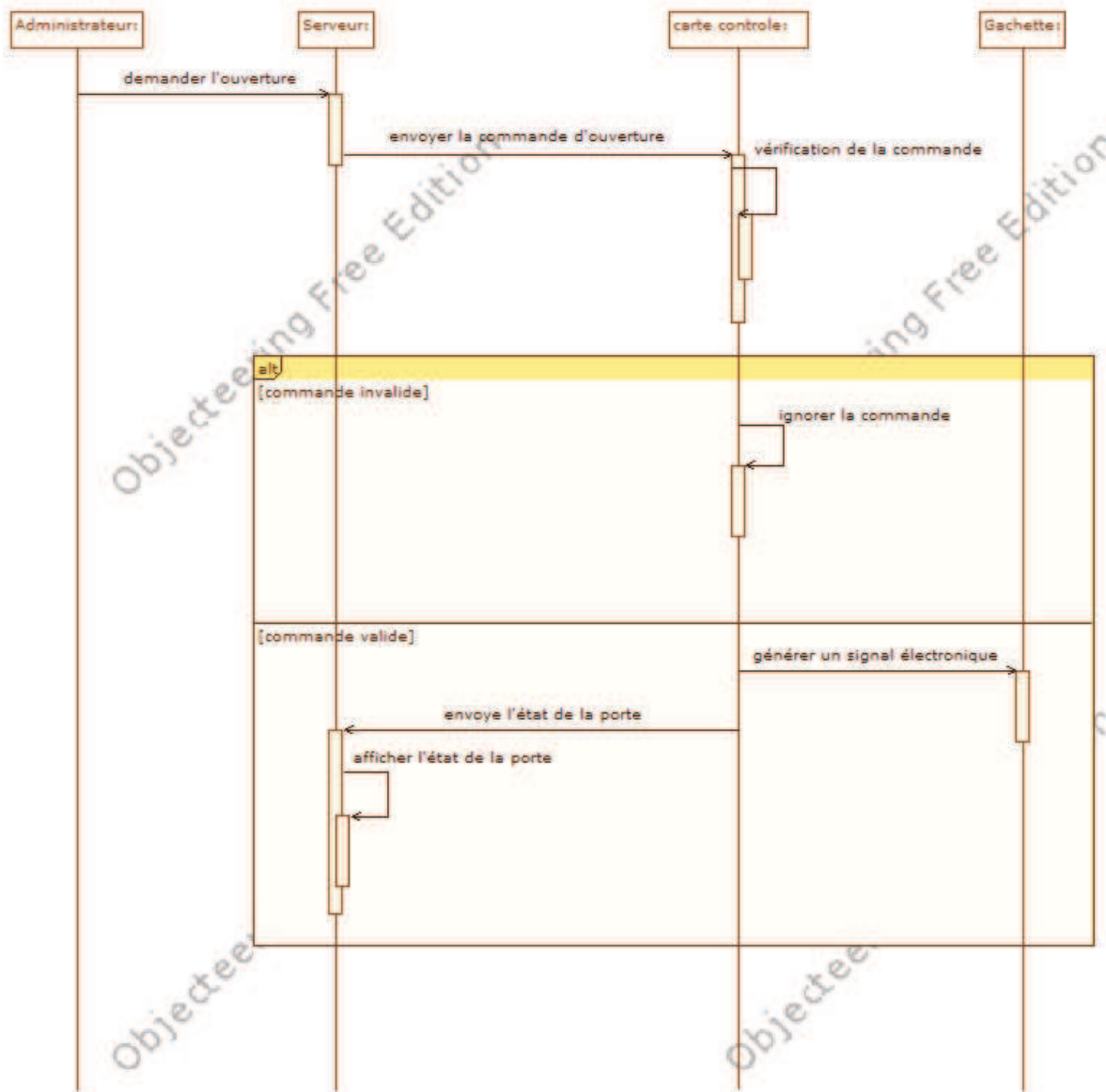
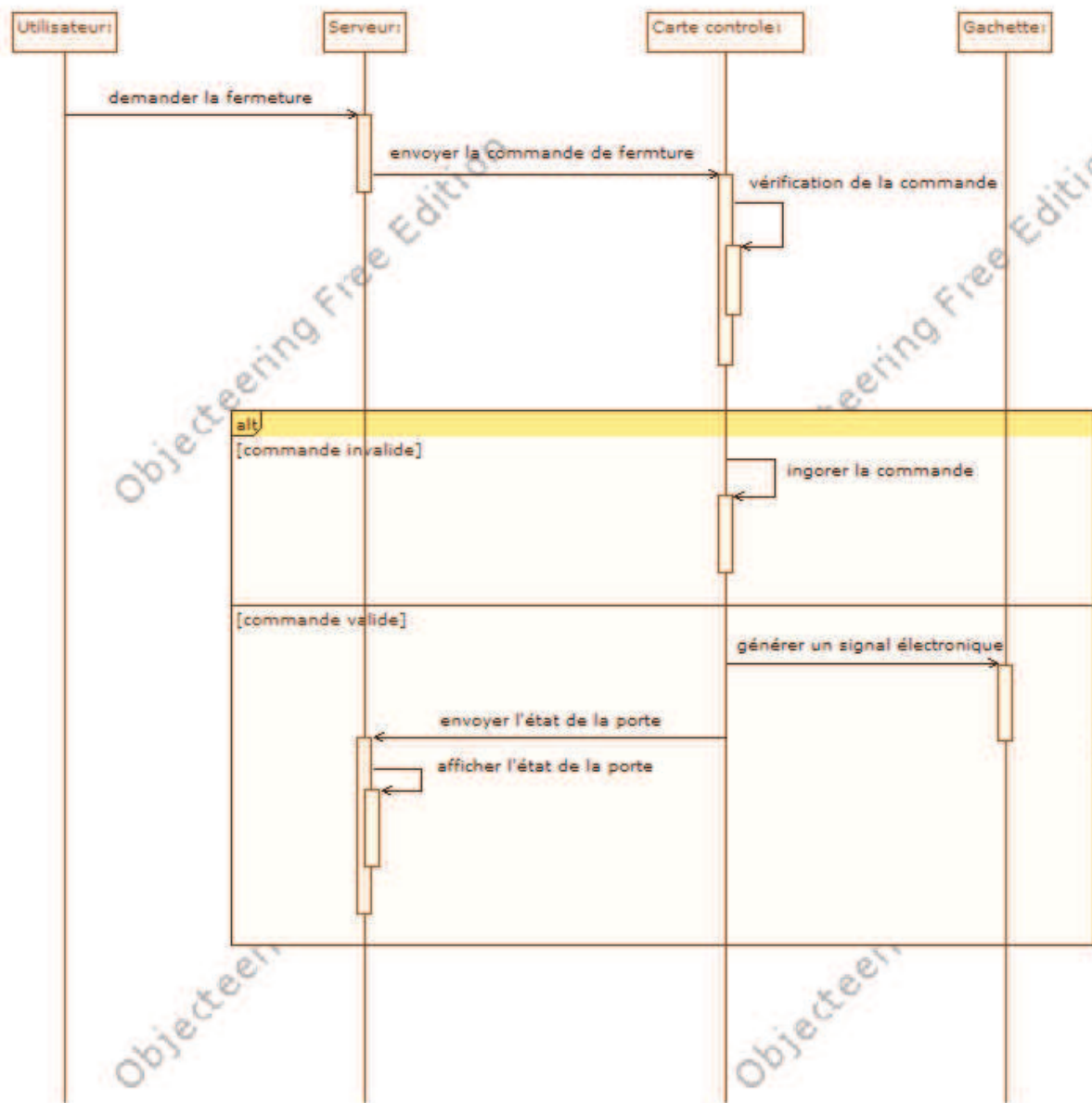
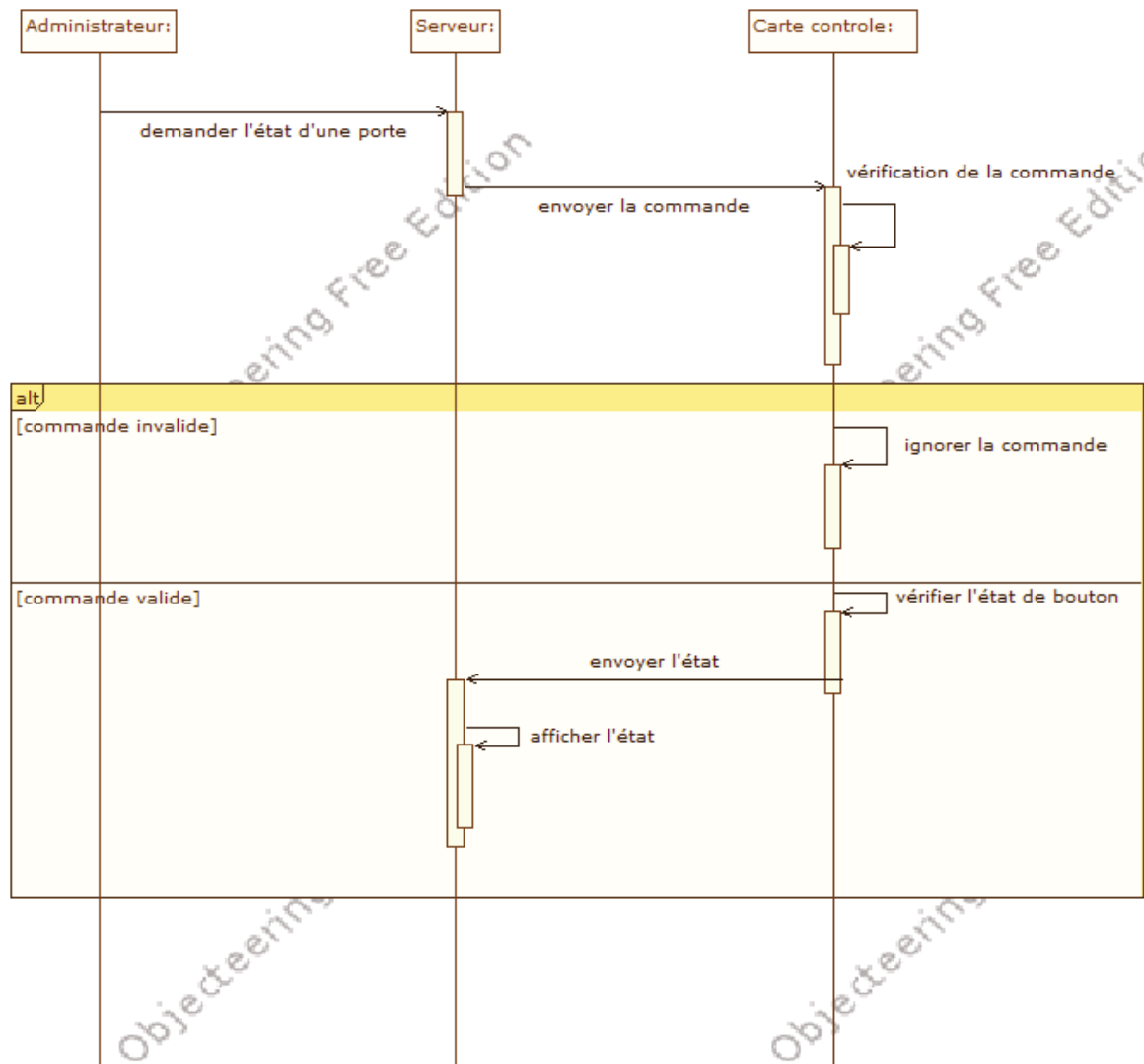


Fig.24 : Diagramme de séquence « ouvrir ».

**Fig.25 :** Diagramme de séquence « fermer »

**Fig.26 :** Diagramme de séquence « vérifier l'état d'une porte »

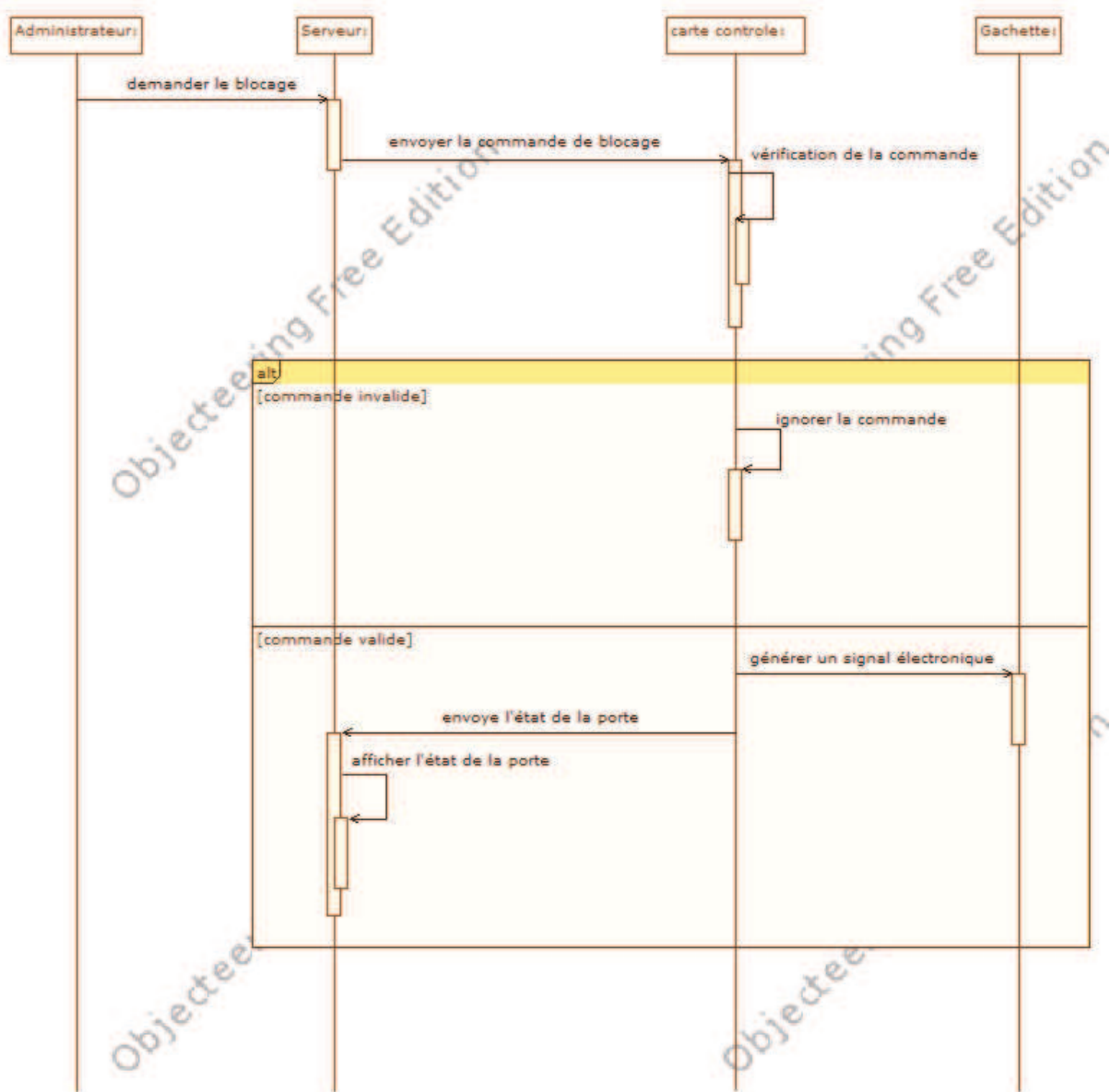


Fig.27 : Diagramme de séquence «bloquer une porte(s) »



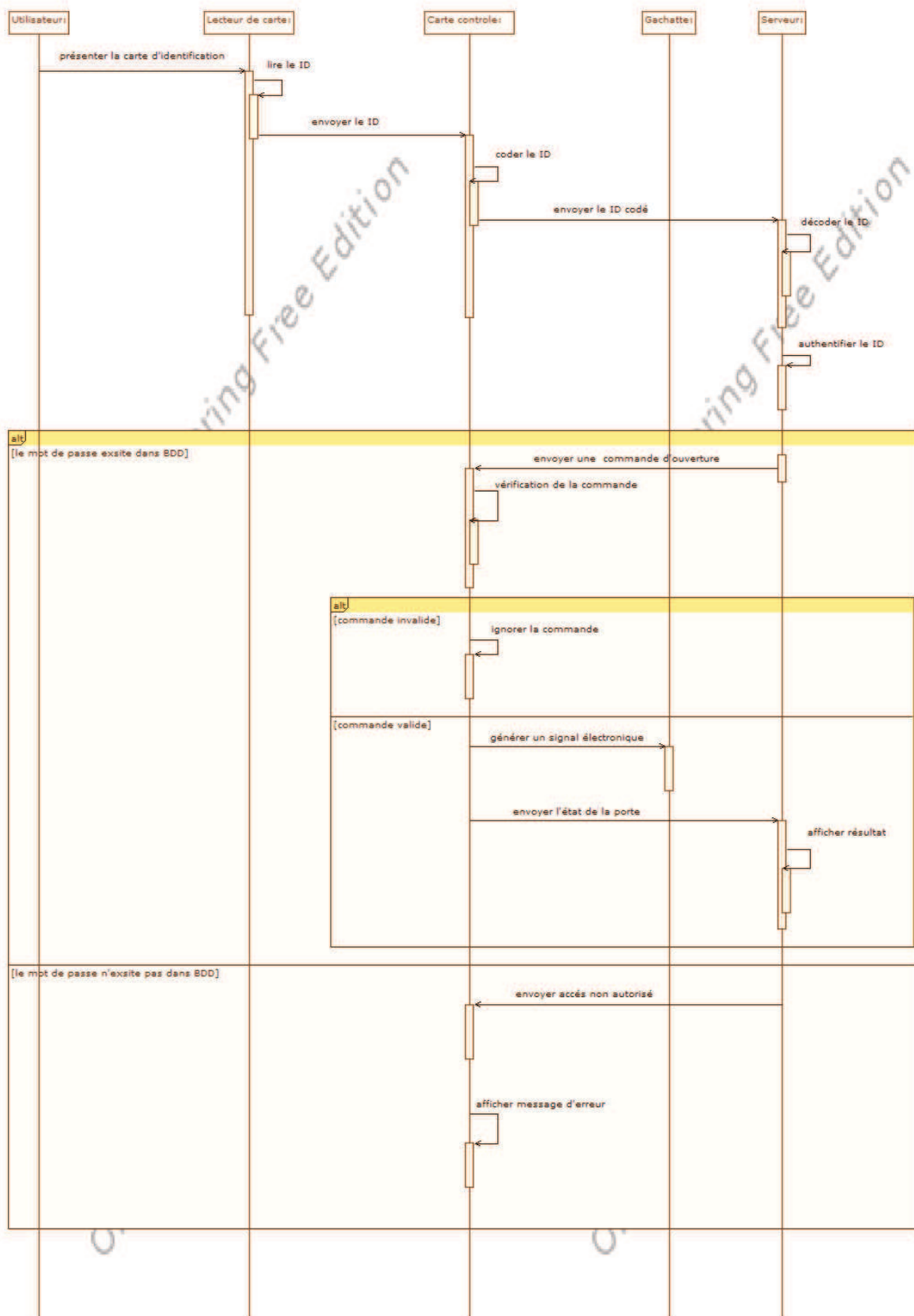


Fig.28 : Diagramme de séquence « accéder à une porte »

4.4. Diagramme d'état-transition:

Les diagrammes d'état-transition servent à représenter des automates à états-finis sous forme de graphes d'états, reliés par des arcs orientés qui décrivent les transitions. Les diagrammes d'états-transitions permettent de décrire les changements d'états d'un objet ou d'un composant, en réponse aux interactions avec d'autres objets, composant ou acteurs.

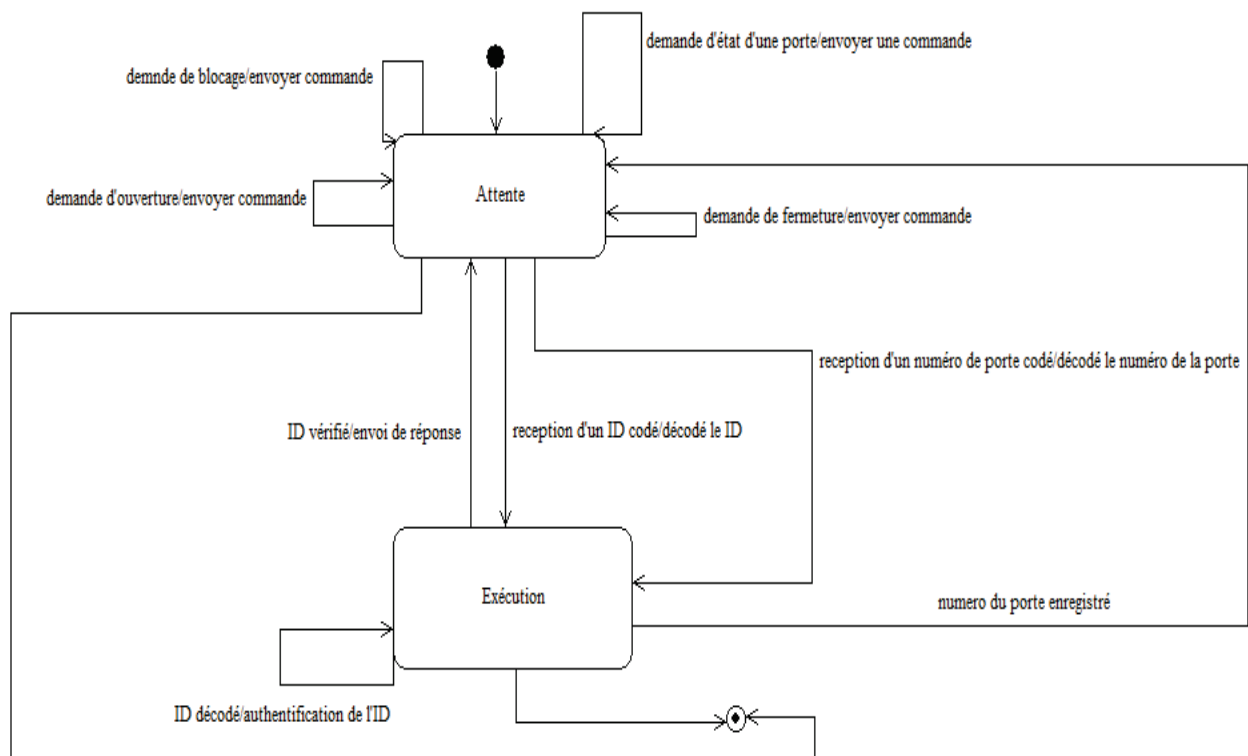


Fig.29 : Diagramme d'état transition « serveur »

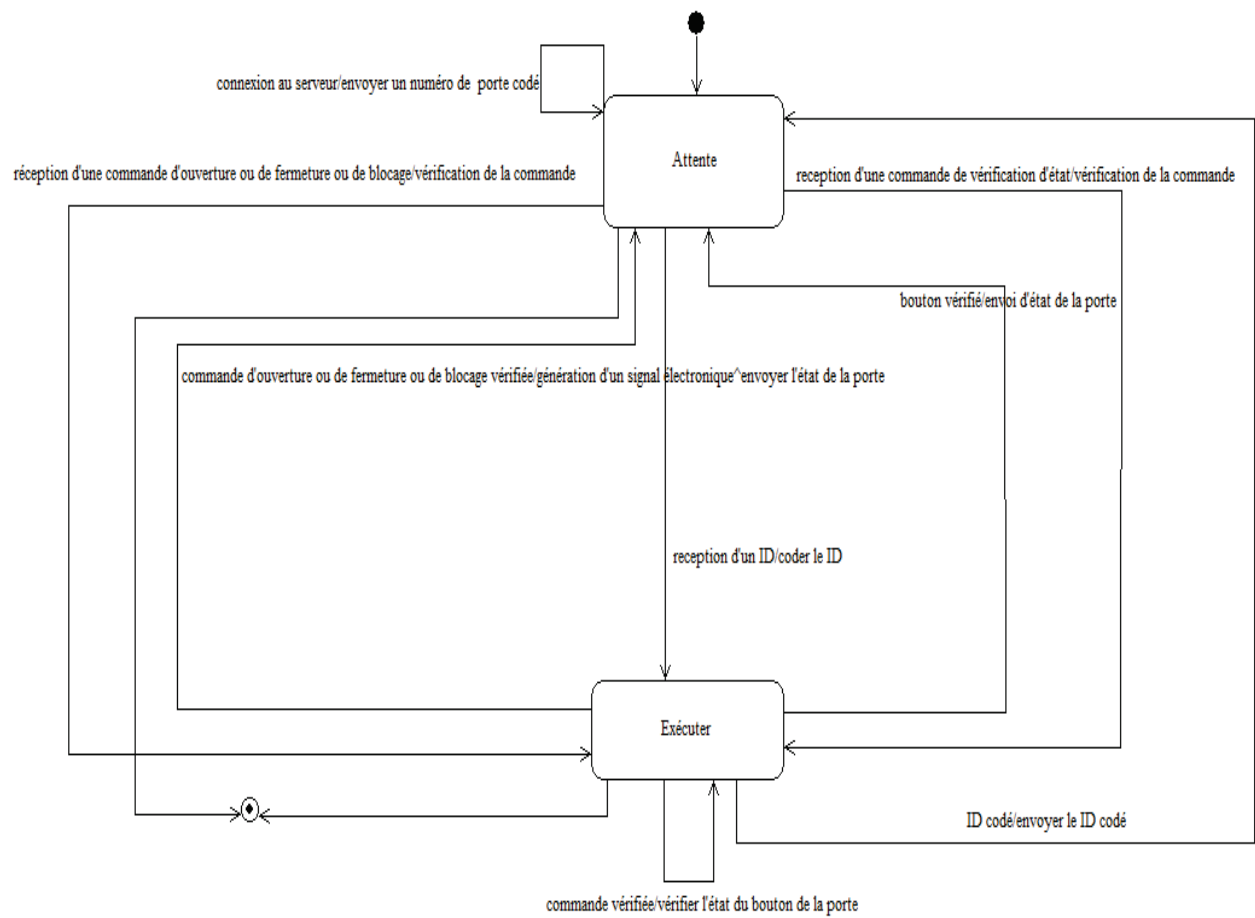


Fig.30 : Diagramme d'état transition « client »

L'outil que nous avons développé est constitué de l'interface suivante :



Fig.31: interface pour « système de contrôle d'accès »

Conclusion

Nous pouvons dire que l'objectif qui a été visé au départ, consistant en l'étude et l'implémentation de la pile TCP/IP de Microchip sur un dsPIC et le développement d'un système de contrôle d'accès simple a été plus ou moins atteint par l'aboutissement de ce modeste travail qui, nous espérons, reflétera la richesse des notions et des connaissances acquises lors de son élaboration.

Le travail dans le domaine de systèmes embarqués exige la maîtrise de plusieurs notions techniques aussi bien dans le domaine de l'électronique et celui de l'informatique, ce qui a nous donné l'occasion de découvrir le protocole de communication série, la programmation bas niveau par MPLAB, les techniques nécessaires pour l'utilisation des microcontrôleurs dans les montages électroniques.

En effet malgré les grosses difficultés que nous avons rencontré, nous citons : le manque de ressource dans ce domaine et le peu de temps consacré à cette étude, nous pouvons dire que nous avons très bien profité de ce travail et qu'en toutes circonstances, nous avons appris beaucoup de choses : le fonctionnement des protocoles Internet dans le détail, Cela nous a permis d'acquérir beaucoup de connaissances dans le domaine des réseaux, les compétences acquises dans le domaine de la Sécurité civile « les systèmes de contrôle d'accès » et ses technologies, la maîtrise et le savoir-faire dans le domaines de simulation avec Proteus, ainsi que beaucoup d'autres connaissances concernant le milieu du système embarqué qui sont des atouts très précieux que nous estimons avoir bien assumé .

Enfin, nous ne prétendons guère avoir contourné tout le sujet, plusieurs améliorations et ajouts peuvent être apportés tels que l'utilisation de protocole ICMP, DNS, implémentation d'un algorithme de cryptage pour le traitement efficace de communication, l'installation des capteurs et des détecteurs contre l'incendie et l'intrusion, autorisation d'accès selon l'heure, Instantanéité de la mise à jour des droits d'accès, Base de données des journaux d'événements téléchargeable ...plusieurs choses que nous espérons réaliser au futur et qui reste également à la portée de certains qui pourraient être intéressés dans le cadre d'un projet de fin d'études ou autre.

Toutefois, nous pouvons dire que nous sommes satisfaits de nos résultats et souhaitons qu'il soit bien valorisé.



Bibliographie

- [1] Pham Viet Hung , « Système Embarqué », 2005
- [2] Patrice KADIONIK, « Les Systèmes embarqués, Linux pour l'embarqué »,2003
- [3] François Laissus, « Cours d'introduction à TCP/IP », 20 février 2005
- [4] Pierre-Alain Goupille, « TECHNOLOGIE DES ORDINATEURS ET DES RÉSEAUX »,8e édition
- [5] Frédéric Jacquenod, « Normalisation des réseaux »,03 janvier 2008
- [6] Laurent Signac, « Réseaux : Tcp/Ip, Internet, sécurité »
- [11] «Démarrer avec les outils de développement de Microchip »
- [13] Microchip Technology, « Microchip TCP/IP StackHelp »,2011
- [14] Microchip Technology, “The Microchip TCP/IP Stack”,2008, AN833
- [15] Mathieu Lemoine, « Le Protocole http »,2006
- M. Simmons,« Ethernet Theory of Operation »,2008, AN1120
- « PIC24FJ128GA010 FamilyData Sheet », 2007, 39747D
- Abdul Rafiq,« Microchip TCP/IP Stack with BSD Socket API»,2007, AN1108
- « dsPIC30FFamily Reference Manual », 2006, 70046E
- « ENC28J60Data Sheet », 2006, 39662B
- « dsPIC30F6011A/6012A/6013A/6014A Data Sheet»,2006, 70143C
- « dsPICPRO4 user manual»

Webographie

- [7] <http://www.lextronic.fr/P1888-starter-kit-dspic-pro4.html>
- [8] http://clubelek.insa-lyon.fr/joomla/fr/Base_de_connaissances/electronique/initiation_au_micro-controleur.php
- [9] http://www.mon-club-elec.fr/pmwiki_reference_arduino/pmwiki.php?n=Main.LibrairieSPI
- [10] http://fr.wikipedia.org/wiki/Serial_Peripheral_Interface
- [12] <http://www.elektronique.fr/logiciels/proteus.php>