**An-Najah National University**

**Faculty of Engineering**

**Computer Engineering Department**

**Graduation Project 2 Documentation**

**Project Name:**

**GMAIL Messenger & Mail System Using Ethernet Interfaced PIC Microcontroller**

**Done By:**

**Osama Tawfiq Al-Aqel**

**Firas Zahi Shaka’a**

**Supervised By:**

Dr. Anas To’meh

**☺Abstract**

**No one can deny the importance of the GMAIL Messenger or Mail system in the life of people, especially in the business, because every day thousands or millions of emails are sent all over the world.**

**All of us know that there is mobile applications that do the same as the suggested above, but our idea is different.**

**Our idea is to build a device that enables clients from checking their Mail system or chatting with their friends online using an existing internet connection through Ethernet card manually build and economically cheap.**

**Our device is a standalone product that enables our clients from checking their mail and chatting with others through nice hardware interface.**

**Why Gmail?**

**If somebody ask us, why especially Gmail system not other??! We answered him that Google offers the developers a free access to their servers to check Mail system or to access their GTALK Messenger through some powerful APIs offered free by them.**

**So, we decide to make that device using the Gmail system and it can be applied to any other free system such as yahoo system.**

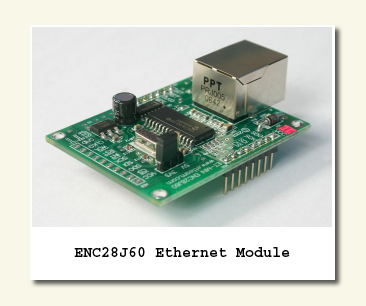
**So, the project is divided into three main sections, which are:**

1. **Ethernet Interfaced PIC microcontroller as Network Card.**
2. **Gmail Mail & Messenger System.**
3. **PIC18 microcontroller boot loader.**

**The project takes 3 month to be done as mentioned above, and many difficulties face us and we handled them very well.**

**☺Project Main Sections**

1. **Ethernet Interfaced PIC microcontroller as Network Card**

****

**In order to have a device that access internet easily in order to reach Google servers, we need like a small computer that takes an IP Address on the internet, which is, PIC18F4620 microcontroller that implements the common 5 network layers and their protocols.**

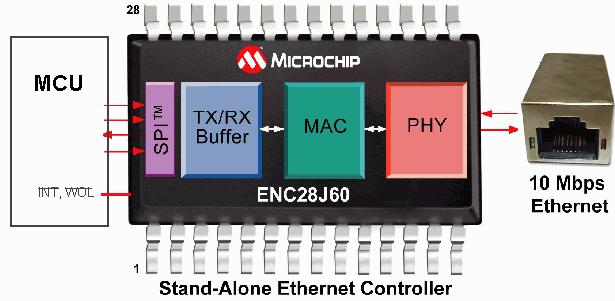
**So we use the following components in building the network card:**

1. **PIC18F4620 microcontroller.**
2. **ENC28J60 Ethernet chip.**
3. **PPT RJ-45 connector.**

**The PIC microcontroller is programmed by the boot loader we made with the TCP/IP stack which implements the 5 network layers and their protocols on the PIC, in order to make the PIC take an IP Address on the internet.**

**All networks protocols are enabled such as TCP, UDP, ICMP, FTP, HTTP Server, … etc.**

**So that we can ping to the PIC to make sure that it is online and has a unique IP Address.**

****

**Here we start, we use the TCP protocol to open a TCP socket with a web service that we build, this web service job is to take the information needed for the interaction with Gmail servers and deal with them, and respond back with any data or feedback.**

**\*\* The Web Service**

**We build a .Net web service that uses the .Net package Jabber-Net that is a Google API free for developers in order to access Gmail servers through it.**

**This web service takes the Gmail user ID and password and checks them with Gmail servers and returns authentication for the user.**

**Many operations and interactions can be made after logging in Gmail servers; we will talk about them later.**

**Jabber-Net is a Gmail API package written in .Net to implement the Extensible Messaging & Presence Protocol (XMPP Protocol) that used by different Mail companies in order to implement their messaging systems including Gmail.**

1. **Gmail Mail & Messenger System**

****

**Our Gmail system hardware consists of the following:**

1. **Graphical LCD to show output and input.**
2. **2 Keypads for user inputs.**
3. **PIC18F4620 microcontroller to talk with the Ethernet system.**

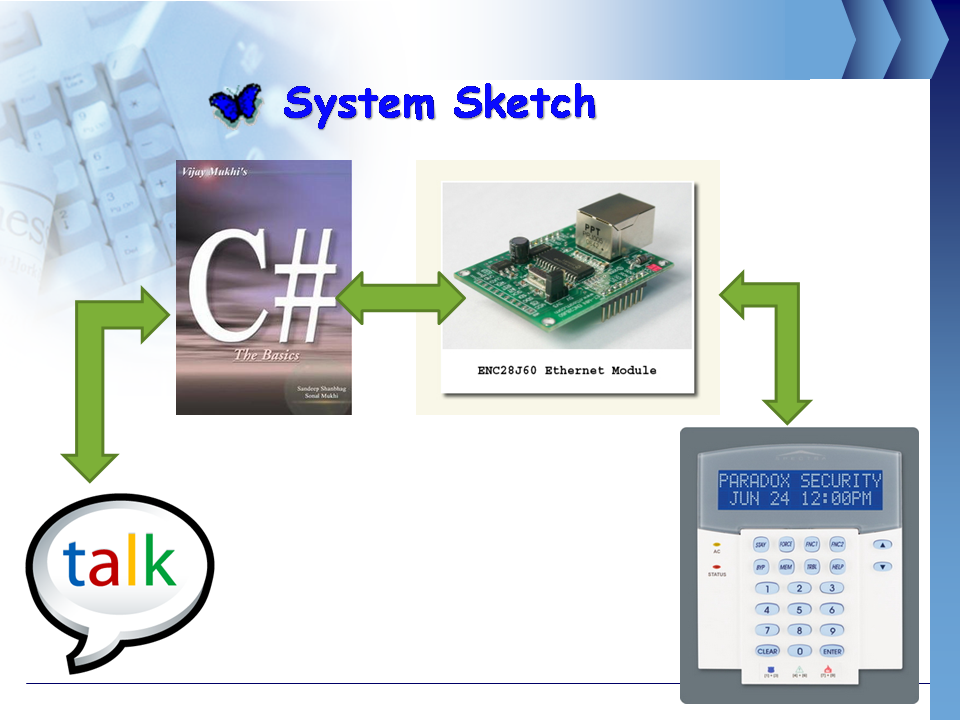
**With these components the user can interact with the Gmail servers in order to access Gmail Services either the Mail system or the GTALK messenger.**

**When the device starts, a welcome screen appears to the user and asks him to enter his Gmail email and password in order to login to his/her account.**

**If the login process succeeded, a menu will appear to the user in order to choose what to do, users options are as follows:**

1. **Checking user’s Gmail Inbox to read messages.**
2. **Sending an email to his/her friends.**
3. **Login to GTALK messenger account.**
4. **Import GTALK contact list of the user account.**
5. **Chat with your friends which are on your contact list.**
6. **Logout from Gmail system.**

**So, when we merge the Ethernet Interface module with the hardware Gmail module, we will get our portable device, which is:**

****

1. **PIC18F4620 Boot Loader**

****

**It is a PIC18F4620 Boot loader consists of two parts, which are:**

1. **PIC side:**

**Written in PIC-C and assembly code, fully protected, this code is the one that will interact with the application that controls the programming process.**

**This code is put at the end of the PIC18F4620 program memory and takes very small size that will not affect the original size of the user’s code program.**

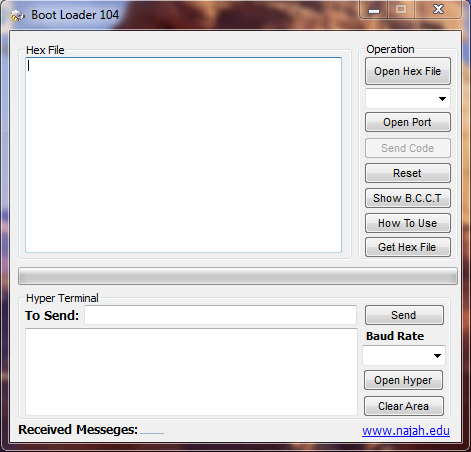
1. **Computer side:**

**It is a C# application that sends the user’s code to the PIC and receives acks from it after each operation.**

**The boot loader has the following properties:**

1. **Can be modified easily to any frequency and already exists for 4, 10, 25 and 40 MHz.**
2. **Applicable for any PIC18 that has 64K program memory.**
3. **It has a built-in HyperTerminal so that the user will not go every time to the windows terminal to test his/her code.**
4. **Simple and nice graphical user interface.**
5. **Spread widely between students.**

**The boot loader application looks like:**

****

**☺Future Plans**

**We are going in the future to accomplish the following:**

1. **Make the PIC access the internet Wireless without the need to the Ethernet cable.**
2. **Support the voice calls in the Gmail system.**
3. **Build the device as printed circuit portable product.**