PPU College Of Engineering and Technology
The Home of Competent Engineers and Researchers

Electrical and Computer Engineering Department

Communication Engineering Program

Bachelor Thesis

Graduation Project

GPRS Based Water Tele-Control System

Project Team

Abdellatif Al-Karaki **Eman Awad**

Mahmoud Malkawi **Walaa Shawar**

Project Supervisor

Dr. Eng . Murad Abu Subaih

Hebron – Palestine

2011-2012

الملخص

تقوم فكرة المشروع بشكل عام على بناء نظام تحكم شامل بشبكات المياه, حيث يقوم بإدارة الشبكة من ناحية توزيع المياه و التحكم بالشبكة بطريقة سلسة و سريعة و غير مكلفة, بالإضافة إلى ذلك سيوفر المشروع الكثير من الوقت و الجهد المستخدمين لادارة الشبكة بالطريقة التقليدية. حيث سيقوم بحساب كميات المياه المتوفرة في جميع الخزانات الرئيسية و بناء الجدول توزيع مناسب لكافة مناطق الشبكة, و من ثم توزيع المياه بطريقة عادلة و مناسبة لجميع المستخدمين عن طريق التحكم بالمحابس المتواجدة بالنقاط الرئيسة و الفرعية.

يستخدم المشروع تقنية (GPRS) اللاسلكية التي تدعمها شبكة الهواتف المحمولة (GSM) للقيام بعملية نقل المعلومات و الاوامر من المركز الرئيسي للشبكة الى النقاط الفرعية و نقاط المستخدمين و بالعكس.

كل ذلك سيتم التعامل معه عن طريق برنامج مخصوص, سوف يبنى باستخدام لغة البرمجة (C#), الداعمة لاوامر الاتصال بالمودم (AT commands), و الداعمة ايضا بناء و اجهة مستخدم لتسهيل التعامل مع البرنامج.

مقابل ذلك سوف يتم التحكم بالمحابس والمجسات و دارات الإنذار في النقاط الفرعية باستخدام متحكم دقيق (PIC) , و سيقوم ايضا بوظيفة جمع البيانات و التواصل مع المودم و ارسال هذه البيانات الى المركز الرئيسي ليتم تخزينها. موفرا بذلك كل الوقت المستخدم لعملية نقل البيانات والجهد المبذول للقيام بذلك يدويا, بالاضافة الى توفير الحماية و الامان في عملية توزيع المياه, حيث ان النظام سيكون قادرا على اكتشاف التسريب و السرقات في مياه الشبكة.

Abstract

The water network in the cities extends fast. That makes the process of managing the network hard and tiring. In addition, the field of monitoring is an exhausting process and takes long hours. By using the advantages of the development that occurs in the wireless communication field, we can build a tele-control system that facilitates the managing, and monitoring of the water network from the main office. The control and management will be based on a well prepared schedule by either a responsible person at the central office or by the system itself.

To guarantee a fair distribution of water for all subscribers, and to save time and efforts that are needed to control and monitor the water network, several sensors with a microcontroller and GPRS modem will be added at each network node, which allows data transmission between the nodes and the central office by GPRS technology.

Using the C and C# programming languages, a special software modules will be built for the controlling and monitoring purposes. Then by connecting the hardware components, a complete wireless system will be ready to be applied.

Dedication

To the memory of those brave men and women who so nobly give their lives to our holy beloved Palestine

To our gorgeous supporters in life to our parents

To our great university, Palestine Polytechnic University

To all the friends

To everyone who has helped us in this project

To you we dedicate this project

Acknowledgements

This project was funded by the Deanship of scientific research, within the program of support for outstanding graduate.

It's honor to us to express our warm and sincere thanks to our great supervisor, Dr. Eng. Murad Abu Sbaih, for his encouragement, guidance, support and his great efforts to explain things clearly and simply.

We are grateful to Hebron Municipality, for supplying us with useful and beneficial information about their system.

We owe our deepest gratitude to Watania Mobile Company, for granting free data SIMs needed for the project.

Many thanks to our proficient teachers and the staff at Palestine Polytechnic University.

Table of contents

Table of Figures

Table of Abbreviations

Abbreviation	Statement
A/D	Analog to digital
ALU	Arithmetical Logical Unit
AMR	Automatic Reader
AT	Attention
CPU	Central processing unit
DHCP	Dynamic Host Configuration Protocol
DSP	digital signal processor
EEPROM	Electrically Erasable Programmable ROM
FM	Factory Mutual
GPRS	General Packet Radio Service
GSM	Global system for mobile communication
GTP	GPRS Tunneling Protocol
GUI	Graphical user interface
HLR	Home location register
HMI	Human-Machine Interface
IC	Integrated circuit
IP	Internet protocol

PC	Personal Computer
PCMCIA	(Personal Computer Memory Card international Association
PIC	Peripheral Interface Controller
QoS	Quality of service
RAM	Random access memory
ROM	Read only memory
RTU	Remote Terminal Unit
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SFR	Special Function Registers
SGSN	Serving GPRS Support Node
SIM	subscriber identity module
SMS	short message service
SQL	Structured Query Language
UART	universal asynchronous receiver/transmitter
UL	Underwriters Laboratories
UTMS	Universal Mobile Telecommunications System

1

Chapter one

Introduction

1.1 Overview

This report details a system for remote control and management of a water network. The control and management will be based on a well prepared schedule by either a responsible person at the central office or by the system itself.

The system will be based on one of the modern wireless communication technologies which is GPRS "General Packet Radio Service". The nodes of the network distributed across the city, where at each node a wireless equipment will be fixed. These equipments will allow monitoring information to be sent to the central office from any node in the network.

Each node of the network will have a suitable group of sensors to gather the needed information and perform the needed actions.

The central office will have a PC to perform the monitoring and controlling operations. The PC will be supported by a software which will be developed specially for this system. The software will be built using C# programming language and will be supported with a graphical user interface to ease operating the system.

This technique in monitoring and controlling the water network is applied in many countries around the world, and has been developed by professional companies in this field. the cost of these systems is too high. In this project, we aim to build and develop the system in the lowest possible cost.

1.2 Project Objectives and Motivation

Due to the lack of water in Palestine, the aim is to help in solving this problem by designing and building a system that helps in distributing the limited amount of water to all subscribers fairly by supplying the available amount of water to the whole network according to the demanding percentage. In addition to reduce the amount of wasted water and save the network's maintenance money by offering the leakage and theft alarms.

The system will save the time and efforts that are needed to control and monitor the network by offering the monitoring and controlling operations from a central point instead of visiting each node, also protect the workers of the authority of water from the occupation harassments and attacking, because there will be no need to visit the water network nodes located in areas that are under the control of the occupation authorities.

The system will also provide an economical running cost by requesting the data when needed instead of the continuous transmission. In addition, it will allow a flexible dealing with different hardware component's manufacturers with the lowest possible cost.

1.3 Approach

The system is divided into two main parts, the central office, and the distributed nodes over the water network.

The central node will consist of a PC which is supported by a special software, and a GPRS modem that will enable the PC to communicate with other nodes.

The software will have all the functions that help the operator monitor and control the network. And it will support AT commands that will allow the PC communicate with the GPRS modem. This software will be built using the C# programming language depending on the visual C# program which helps in adding a graphical user interface to the software.

The techniques that will be used at the other nodes is varying depending on the function of each one. At the water source node, a pressure sensor will be used in order to measure the amount of water in the tank. And by using a specific equation, the amount of water will be calculated, also an electrical valve will be used in order to supply the water or not, depending on an electrical signal received from the central office.

At the main distribution node, the same electrical valve will be used for the same purpose, also a flow meter will be used in order to measure the amount of water that has been supplied to the specified area, and a pressure sensor will be used to turn on pumps when the water pressure is low, so the water can reach the high areas.

At the end consumer node, the amount of the consumed water will be measured by the same technique using a flow meter in order to be sent to the central office.

A microcontroller and a GPRS modem will be used at each node. The microcontroller will be the interface between the node components and the GPRS modem, and it will use AT commands to let it deal with the modem, also the microcontroller will have a specified instructions to link the components of the node to the central office. The GPRS modems will be used to connect the nodes to the central office using the GSM network.

As a protection method, a group of security sensors will be added to the equipments of each node to generate an alarm signal at any attempt of tempering in the equipments or at cases of equipment loss prospects.

1.4 Literature Review

The Tele-control system of the municipality of Hebron [1]

An Italian GPRS based SCADA system for controlling and monitoring the distribution of water in Hebron. It concerns at the water supply and the main distributing nodes. It provides several functions; some of them are not used due to the shortage of water in Palestine.

The system offers a control over the valves in the network due to a schedule or due the operator's orders. The valves at the system can be partially or completely opened depending on the operator's order, but the partially opened is not used. And the state of the valve at any time can be known to the operator from the system's software.

The system used the flow meter as a water quantity counter, and by measuring the speed of the flow of water and the pipe's cross section with using the volume equation, the amount of water can be calculated.

The system uses pressure sensors for several reasons; the value of the water pressure is used to protect the pipes of the network that can accept a specified pressure values. Pressure sensors are also used in order to measure the amount of the water in the tanks from the pressure value of the tanks water. The system can control the water pressure using regulating valves.

The system had a smart water distribution, but this option is not used.

The components that the system uses are mention in the following points:

1) A Computer: to gather the data and sends commands, with a special software to do that.
2) GPRS modems: with an interface, to connect the central office computer to the node using the GPRS technology
3) RTUs: Remote Terminal Unit, to control the equipments in the node, and it contains alerts for security, Electrical switches in order to control every element in the node, Fuses, Batteries, Temperature regulation, and PLC "Programmable Logic Controller" to convert the sensor's analog signals to a digital data and send it to the central computer.

4) A Human-Machine Interface or HMI (touch screen) is the apparatus which presents the data to the human operator, and through this, the human operator monitors and controls the node.

Water well SCADA system based on GPRS. [2]

One of the applications of the GPRS technology that has been used for controlling the water wells automatically instead of the manual controlling, it realizes the remote control of the water plant network system by using the GPRS, they achieved the remote monitoring and management of several wells in an effective way.

Remote Data Acquisition Using Wireless - SCADA System. [3]

A paper about a system that monitors and accesses remotely a suited device parameters like temperature, pressure, humidity on real time basis using the GPRS technology, and so, by eliminating the need of the human's visit to each site for the data collection, the system saves time and money.

Trends in SCADA for Automated Water Systems. [4]

A paper that shows the basic knowledge that needed to choose a technology and design a SCADA system that will reduces the cost and improve the performance by giving an overview of a recent trends in SCADA for water systems and water waste systems.

Design and Implementation of Secure Low Cost AMR system using GPRS Technology. [5]

A paper about a low cost secure Automatic Reader Wireless System that can calculate the total electrical energy consumption, making a billing service, and by using the GPRS technology it can transmit the information to the main server wirelessly.

In this project we are attending to build a similar system to the one of the Hebron Municipality, with the used functions of the current system and some extra ones in a lower cost.

The current system controls a small number of nodes, and focuses on the main nodes of the water network, our aim is to develop a system that can reach the end consumer, and can cover every node in the water network.

1.5 Cost Analysis :-

this project has different parts of software and hardware , in software we use a free edition of visual studio 2010 , and a free edition of MPlab PIC programming software .

the following table shows the specific hardware cost:

Part	Price * amount	part	Price
GPRS modem telit	700 * 2	Flow meter	50 * 3
GPRS modem sm5100b	510 * 1	Analog water meter	150 * 1
GPRS USB modem	270 * 1	Sensors	25 * 3
PIC microcontroller	65 * 8	valves	25 * 4
PIC bridges	10 * 8	Water network parts	310
Circuit boards	10 * 4	Water tank	150 * 1
Connectors	6 * 30	Water motor pump	150 * 1
LCD	45 * 3	DC sockets	42
Relays	10 * 5	cables	50
AC Clements	5 * 5	Ethernet connecters	21 * 2
AC electrical parts	46	Power supply	30 * 1
Wooden works	345	Switch	10 * 1
Buttons	4 * 4	transistors	1 * 30
Resistors	0.5 * 60	Pump	50 * 1
Documents printing	300	Data SIM cards	Over 4000 Funded by Wataniya mobile

Witch totally costs about 5286 NIS

2

Chapter Two

Theoretical Background

2.1 Introduction

This chapter focuses on the technologies and components that the project needs, and provides an overview about GPRS technology, PIC microcontroller, AT commands, programming language, and other parts of the system.

2.2 General Packet Radio Service (GPRS)

2.2.1 Overview

General Packet Radio Service (GPRS) is a data network architecture that is designed to integrate with the existing GSM networks and offer mobile subscribers “always on” packet switched data services access to the corporate networks and the Internet.

GPRS provides mobile operators with an opportunity to offer higher-margin data access services to subscribers. In return, subscribers benefit from GPRS by being able to use higher bandwidth mobile connections to the Internet and corporate networks. GPRS Tunneling Protocol (GTP) is the protocol used by GSM or UTMS operators to convert radio signals from subscribers into data packets, and then to transport them in non-encrypted tunnels. GTP does not provide inherent security.

With the addition of GPRS to GSM, mobile operators are adding mobile Internet and virtual private network services to their existing mobile voice services. GPRS networks are connected to several external data networks including those of roaming partners, corporate customers, GPRS Roaming Exchange providers, and the public Internet.

From a high level, GPRS can be thought of as an overlay network onto a second generation GSM network. This data overlay network provides packet data transport at rates from 9.6 to 171 kbps. In addition, multiple users can share the same air-interface resources [6].

2.2.2 How Does GPRS Work?

When a user turns on a GPRS device, typically it will automatically scan for a local GPRS channel. If an appropriate channel is detected, the device will attempt to attach to the network. The SGSN receives the attach request, fetches subscriber profile information from the subscriber’s HLR node, and authenticates the user. Ciphering may be established at this point [7].

2.2.3 IP Address

An IP address is a unique address that is assigned to devices so that they can identify and communicate with each other on a network.

There are two types of IP addresses:

1) Static IP address, which is a unique and fixed IP address on a network.
2) Dynamic IP address, which is assigned by means of the Dynamic Host Configuration Protocol (DHCP) every time a device starts [8].

2.2.4 Advantages of GPRS

GPRS provides faster data transfer rates, "always on" connection, robust connectivity, broad application support, and strong security mechanisms. The main benefits of GPRS are that it reserves radio resources only when there is data to send, and reduces reliance on traditional circuit-switched network elements.

2.2.5 GPRS Tariffing

GSM circuit switched based voice services are charged mainly on duration of calls. GPRS uses packet switching and charging will be based on totally different and has many more dimensions, because the user is always logged on the network, whether they are actively using it or not.

Some of the tariff dimensions that could be used for GPRS subscribers:
Number of packets transported

1) Volume in terms of kilo bites or mega bites
2) Uplink and downlink volume
3) Type of content
4) Time delay and quality of service (QoS) , charge more for real time applications [9].

2.3 Microcontrollers

2.3.1 Definition of microcontroller

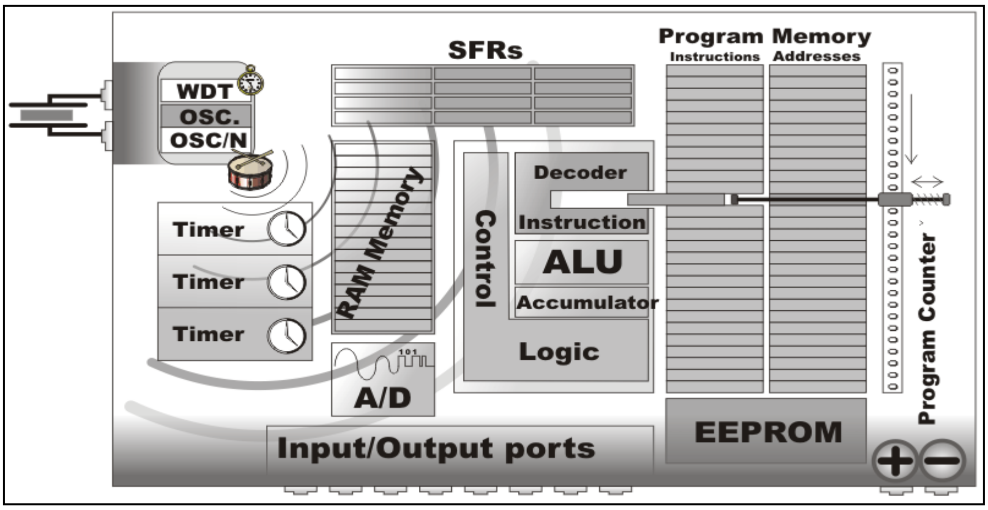
A microcontroller is a small economical computer-on-a-chip built for dealing with specific tasks, containing a processor core, memory, and programmable input/output peripherals. Program memory in the form ROM is also included on chip, as well as a typically small amount of RAM. Microcontrollers are designed for embedded applications, in contrast to the microprocessors used in personal computers or other general purpose applications.

Microcontrollers are used in automatically controlled products and devices, such as automobile engine control systems, implantable medical devices, remote controls, office machines, appliances, power tools, toys and other embedded systems. By reducing the size and cost compared to a design that uses a separate microprocessor, memory, and input/output devices, microcontrollers make it economical to digitally control even more devices and processes. Mixed signal microcontrollers are common, integrating analog components needed to control non-digital electronic systems.

Some microcontrollers may use four-bit words and operate at clock rate frequencies as low as 4 kHz, for low power consumption (milliwatts or microwatts). They will generally have the ability to retain functionality while waiting for an event such as a button press or other interrupt; power consumption while sleeping (CPU clock and most peripherals off) may be just nanowatts, making many of them well suited for long lasting battery applications. Other microcontrollers may serve performance-critical roles, where they may need to act more like a digital signal processor (DSP), with higher clock speeds and power consumption.

2.3.2 What is inside the microcontroller?

As we can see, all the operations within the microcontroller are performed at high speed and quite simply, but the microcontroller itself would not be so useful if there are not special circuits which make it complete. In this section , we are going to call attention to them.



Figure(2.1) Microcontroller’s Components

Read Only Memory (ROM)

Read Only Memory (ROM) is a type of memory used to permanently save the program being executed. The size of the program that can be written depends on the size of this memory. The size of ROM ranges from 512B to 64KB.

Random Access Memory (RAM)

Random Access Memory (RAM) is a type of memory used for temporary storing data and intermediate results created and used during the operation of the microcontrollers. The content of this memory is cleared once the power supply is off.

Electrically Erasable Programmable ROM (EEPROM)

The EEPROM is a special type of memory not contained in all microcontrollers. Its contents may be changed during program execution (similar to RAM), but remains permanently saved even after the loss of power (similar to ROM). It is often used to store values, created and used during operation.

Special Function Registers (SFR)

Special function registers are part of RAM memory. Their purpose is predefined by the manufacturer and cannot be changed therefore. Since their bits are physically connected to particular circuits within the microcontroller, such as A/D converter, serial communication module etc., any change of their state directly affects the operation of the microcontroller or some of the circuits.

Central Processor Unit (CPU)

As its name suggests, this is a unit which monitors and controls all processes within the microcontroller and the user cannot affect its work.

Input/output ports (I/O Ports)

In order to make the microcontroller useful, it is necessary to connect it to peripheral devices. Each microcontroller has one or more registers (called a port) connected to the microcontroller pins.

Oscillator

Even pulses generated by the oscillator enable harmonic and synchronous operation of all circuits within the microcontroller. It is usually configured as to use quartz-crystal or ceramics resonator for frequency stabilization. It can also operate without elements for frequency stabilization (like RC oscillator).

Timers/Counters

Most programs use these miniature electronic "stopwatches" in their operation. These are commonly 8- or 16-bit SFRs the contents of which are automatically incremented by each coming pulse. Once the register is completely loaded, an interrupt is generated!

If these registers use an internal quartz oscillator as a clock source, then it is possible to measure the time between two events (if the register value is T1 at the moment measurement has started, and T2 at the moment it has finished, then the elapsed time is equal to the result of subtraction T2-T1). If the registers use pulses coming from external source, then such a timer is turned into a counter.

Serial Communication, RS232, UART

A serial connection on your microcontroller is very useful for communication. You can use it to program your controller from a computer, use it to output data from your controller to your computer (great for debugging), or even use it to operate other electronics such as digital video cameras. Usually the microcontroller would require an external IC to handle everything, such as an RS232.

2.3.3 (PIC) Programming Languages

Unlike other integrated circuits which only need to be connected to other components and turn the power supply on, the microcontrollers need to be programmed first. And the process of programming the microcontroller is basically very simple.

In order to write a program for the microcontroller, several "low-level" programming languages can be used such as Assembly, C and Basic (and their versions as well). Writing program procedure consists of simple writing instructions in the order in which they should be executed. There are also many programs running in Windows environment used to facilitate the work providing additional visual tools.

The lowest form of programming languages is the **machine language**. Microcontrollers need to be programmed with this.

An example of machine language:

```
3A 10 51
E6 DF
32 38 00
```

Obviously neither of us could ever memorize what all those seemingly random numbers and letters do, so we would program in a higher language that makes much more sense:

```
If(language==easy)
print "yay!";
```

These higher languages would then be compiled automatically into a machine language, which then you can upload into your robot. Probably the easiest language to learn would be **BASIC**, with a name true to itself. The BASIC Stamp microcontroller uses that language.

But BASIC has its limitations, so the programming with c will be faster and more efficient to detect errors in codes.

2.4 sensors and other equipment needed for the project:

2.4.1 Modems

The project basically depends on the GPRS technology, and this technology needs data SIM card and a modem.

GPRS/GSM modem:

A GSM modem is a wireless modem that works with a GSM wireless network. A wireless modem behaves like a dial-up modem. The main difference between them is that a dial-up modem sends and receives data through a fixed telephone line while a wireless modem sends and receives data through radio waves.

A GSM modem can be an external device or a PC Card / PCMCIA Card. Typically, an external GSM modem is connected to a computer through a serial cable or a USB cable.

A GSM modem in the form of a PC Card / PCMCIA Card is designed for use with a laptop computer. It should be inserted into one of the PC Card / PCMCIA Card slots of a laptop computer.

Like a GSM mobile phone, a GSM modem requires a SIM card from a wireless carrier in order to operate.

As mentioned in earlier sections of this chapter, computers and microcontrollers use AT commands to control modems. Both GSM modems and dial-up modems support a common set of standard AT commands.

In addition to the standard AT commands, GSM modems support an extended set of AT commands. These extended AT commands are defined in the GSM standards. With the extended AT commands, you can do things like:

1) Reading, writing and deleting SMS messages.
2) Sending SMS messages.
3) Monitoring the signal strength.
4) Monitoring the charging status and charge level of the battery.
5) Reading, writing and searching phone book entries.

The number of SMS messages that can be processed by a GSM modem per minute is very low (only about six to ten SMS messages per minute).

What is a GPRS Modem?

A GPRS modem is a GSM modem that additionally supports the GPRS technology for data transmission. GPRS stands for General Packet Radio Service. It is a packet-switched technology that is an extension of GSM.

GPRS can be used as the bearer of SMS. If SMS over GPRS is used, an SMS transmission speed of about 30 SMS messages per minute may be achieved. This is much faster than using the ordinary SMS over GSM, whose SMS transmission speed is about 6 to 10 SMS messages per minute. A GPRS modem is needed to send and receive SMS over GPRS. Note that some wireless carriers do not support the sending and receiving of SMS over GPRS.

2.4.2 Pressure sensor (pressure transmitter)

As mentioned in chapter1, an important data type is the pressure of the water in the pipes and in the main tanks; this pressure will be measured with a pressure sensor which output will be a voltage or current according to the pressure value.

A pressure sensor measures pressure, typically of gases or liquids. Pressure is an expression of the force required to stop a fluid from expanding, and is usually stated in terms of force per unit area. A pressure sensor usually acts as a transducer; it generates a signal as a function of the pressure imposed.

Pressure sensors are used for controlling and monitoring in thousands of everyday applications. Pressure sensors can also be used to indirectly measure other variables such as fluid/gas flow, speed, water level, and altitude. Pressure sensors can alternatively be called pressure transducers, pressure transmitters, pressure senders, pressure indicators and piezometers, manometers, among other names.

In this system the pressure will be used to determine two parameters, the first is the amount of water in the tanks, and the second is the pressure of the water in the pipes to determine the height that the water can move up.

2.4.3 Flow meter

The flow of water needed in the project to determine the amount of water that passes through the pipe or the amount of water consumed monthly by the consumer.

Flow measurement

It is the quantification of bulk fluid movement. Flow can be measured in a variety of ways. Positive-displacement flow meters accumulate a fixed volume of fluid and then count the number of times the volume is filled to measure flow. Other flow measurement methods rely on forces produced by the flowing stream as it overcomes a known constriction, to indirectly calculate flow. Flow may be measured by measuring the velocity of fluid over a known area.

Turbine flow meter

The turbine flow meter (better described as an axial turbine) translates the mechanical action of the turbine rotating in the liquid flow around an axis into a user-readable rate of flow (gpm, lpm, etc.). The turbine tends to have all the flow traveling around it.

The turbine wheel is set in the path of a fluid stream. The flowing fluid impinges on the turbine blades, imparting a force to the blade surface and setting the rotor in motion. When a steady rotation speed has been reached, the speed is proportional to fluid velocity.

Turbine flow meters are used for the measurement of natural gas and liquid flow. Turbine meters are less accurate than displacement and jet meters at low flow rates, but the measuring element does not occupy or severely restrict the entire path of flow. The flow direction is generally straight through the meter, allowing for higher flow rates and less pressure loss than displacement-type meters. They are the meter of choice for large commercial users, fire protection, and as master meters for the water distribution system. Strainers are generally required to be installed in front of the meter to protect the measuring element from gravel or other debris that could enter the water distribution system. Turbine meters are generally available for 1-1/2" to 12" or higher pipe sizes. Turbine meter bodies are commonly made of bronze, cast Iron, or ductile iron. Internal turbine elements can be plastic or non-corrosive metal alloys. they are accurate in normal working conditions to 0.2l/s however are affect greatly with dog mix interference.

Fire meters are a specialized type of turbine meter with approvals for the high flow rates required in fire protection. They are often approved by Underwriters Laboratories (UL) or Factory Mutual (FM) for use in fire protection.

Fire hydrant meters are a specialized type of portable turbine meter that are attached to a fire hydrant to measure water out of the hydrant. The meters are normally made of aluminum to be light weight, and are usually 3" capacity. Utilities often require them for measurement of water used in construction, pool filling, or where a permanent meter is not yet installed.

2.4.4 Valves:

A **butterfly valve** is a valve which can be used for isolating or regulating flow. The closing mechanism takes the form of a disk. Operation is similar to that of a ball valve, which allows for quick shut off. Butterfly valves are generally favored because they are lower in cost to other valve designs as well as being lighter in weight, meaning less support is required. The disc is positioned in the center of the pipe, passing through the disc is a rod connected to an actuator on the outside of the valve. Rotating the actuator turns the disc either parallel or perpendicular to the flow. Unlike a ball valve, the disc is always present within the flow; therefore a pressure drop is always induced in the flow, regardless of valve position.

A butterfly valve is from a family of valves called **quarter-turn valves**. The "butterfly" is a metal disc mounted on a rod. When the valve is closed, the disc is turned so that it completely blocks off the passageway. When the valve is fully open, the disc is rotated a quarter turn so that it allows an almost unrestricted passage of the fluid. The valve may also be opened incrementally to throttle flow.

There are different kinds of butterfly valves, each adapted for different pressures and different usage. The resilient butterfly valve, which uses the flexibility of rubber, has the lowest pressure rating. The high performance butterfly valve, used in slightly higher-pressure systems, features a slight offset in the way the disc is positioned, which increases the valve's sealing ability and decreases its tendency to wear. The valve best suited for high-pressure systems is the triple offset butterfly valve, which makes use of a metal seat, and is therefore able to withstand a greater amount of pressure.

2.5 Project software

Overview

Now a days, computers are able to perform different tasks in different fields, but they can't do that by themselves! They need detailed procedures for each task. That what we call a computer program.

There are a lot of programming languages that is used to build out a computer program, choosing any language depends on the tasks of the program.

C# language:

(Pronounced 'see sharp'), is one of the C family programming languages (C, Objective C, C++, etc.) and therefore share a similar syntax [10]. It inherits many of the best features of C++ and Microsoft Visual Basic, but few of inconsistencies, and anachronisms, resulting in a cleaner and more logical language [11]. Its build process is simple compared to C++ and more flexible than in Java. There are no separate header files, and no requirement that methods and types be declared in a particular order.

You can use C# to create traditional windows client applications, XML Web services, distributed components, client-server applications, database applications, and much more [12]. It allows you to overload operators, as well as to create structure and build a variety of robust applications that run on the .NET framework.

Visual C# is an implementation of the C# language by Microsoft. Visual studio supports visual C# with a full-featured code editor, compiler, project templates, designers, and other tools.

The use of the C# and visual studio has several features:

1) Build an application, fast.
2) Design a great looking user interface.
3) Create and interact with database. It integrates seamlessly with SQL Server Compact Edition, and many other popular database systems.

4) Focus on solving the real problems, by:
    - Keeping track of all projects.
    - Make It easy to edit the project's code.
    - Keeping track of project's graphics, audio, icons, and other resources.
    - Managing and interacting with database [13].

2.6 SQL database:

SQL stands for Structured Query Language. It's a programming language for accessing data in databases. It's got its own syntax, keywords, and structure. SQL code takes the form of statements and queries, which access and retrieve the data. A SQL database can hold stored procedures, which are a bunch of SQL statements and queries that are stored in the database and can be run at any time.

Data in a SQL database lives in tables. A table likes a spreadsheet. It organizes the information into columns and rows. The columns are the data categories, like a contact's name and phone number, and each row is the data for one contact card [13].

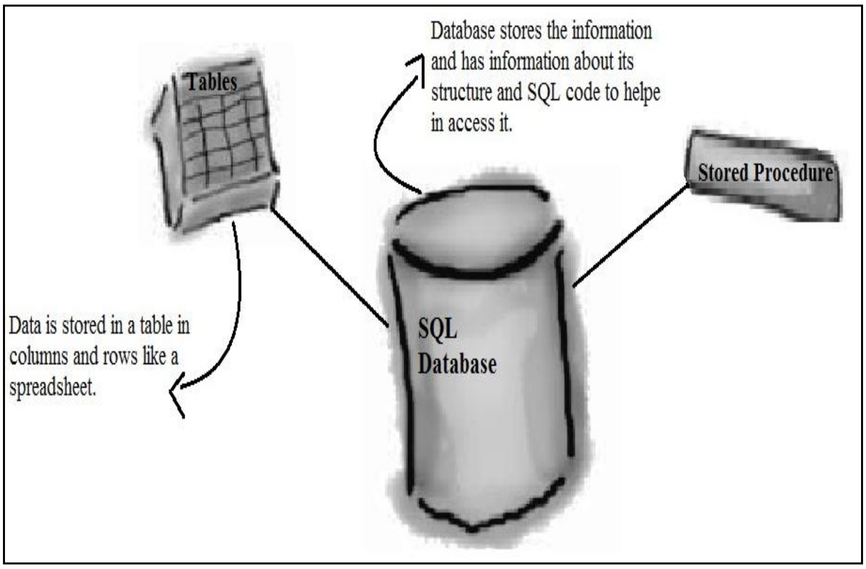


Figure (2.2) SQL Database.

What Can SQL do?

SQL can retrieve data from a database, insert records in a database, update records in a database, delete records from a database, create new databases, create new tables in a database, create stored procedures in a database, create views in a database, and it can set permissions on tables, procedures, and views.

The SQL language is sub-divided into several language elements, including:

1) Clauses, which are constituent components of statements and queries. (In some cases, these are optional.

2) Expressions, which can produce either scalar values or tables consisting of columns and rows of data.

3) Predicates, which specify conditions that can be evaluated to SQL three-valued logic or Boolean (true/false/unknown) truth values and which are used to limit the effects of statements and queries, or to change program flow.

4) Queries, which retrieve the data based on specific criteria. This is the most important element of SQL.

5) Statements, which may have a persistent effect on schemata and data, or which may control transactions, program flow, connections, sessions, or diagnostics.

2.7 AT Commands

AT commands (Hayes command set)

The Hayes command set is a specific command-language originally developed for the Hayes Smart modem 300 baud modem in 1981. The command set consists of a series of short text strings which combine together to produce complete commands for operations such as dialing, hanging up, and changing the parameters of the connection.

Introduction to AT Commands

AT commands are instructions used to control a modem. AT is the abbreviation of ATtention. Every command line starts with "AT" or "at". That's why modem commands are called AT commands. The starting "AT" is the prefix that informs the modem about the start of a command line. It is not part of the AT command name.

GSM/GPRS modems designed for wireless applications have better support of AT commands than ordinary mobile phones. Because the mobile phone manufacturers usually do not implement all AT commands, command parameters and parameter values in their mobile phones. Also, the behavior of the implemented AT commands may be different from that defined in the standard [14].

Basic Commands and Extended Commands

There are two types of AT commands:

1) Basic commands are AT commands that do not start with "+".
2) Extended commands are AT commands that start with "+". All GSM AT commands are extended commands.

Some Examples Of The GPRS AT Commands:

1) AT+CGACT: to activate or deactivate specified PDP (Packet Data Protocol) context.

2) AT+SDATACONF: to config the configure parameters of data sent by AT commands based on GPRS.

3) AT+SDATASTART: to enable GPRS service.

4) AT+SDATATSEND: to send the data specified by user in transparent mode.

5) AT+SDATATREAD: to read the received data and display in transparent mode.

6) AT+SDATASEND: to send the character string data specified by user.

7) AT+SSTRSEND: to send the character string specified by customer.

8) AT+SDATAREAD: to read the received data from the puffer.

3

Chapter Three

System Design

3.1 Introduction

This chapter introduces the system design. It divides it into parts and describes the design of each part independently. Also it shows the interacting process between them.

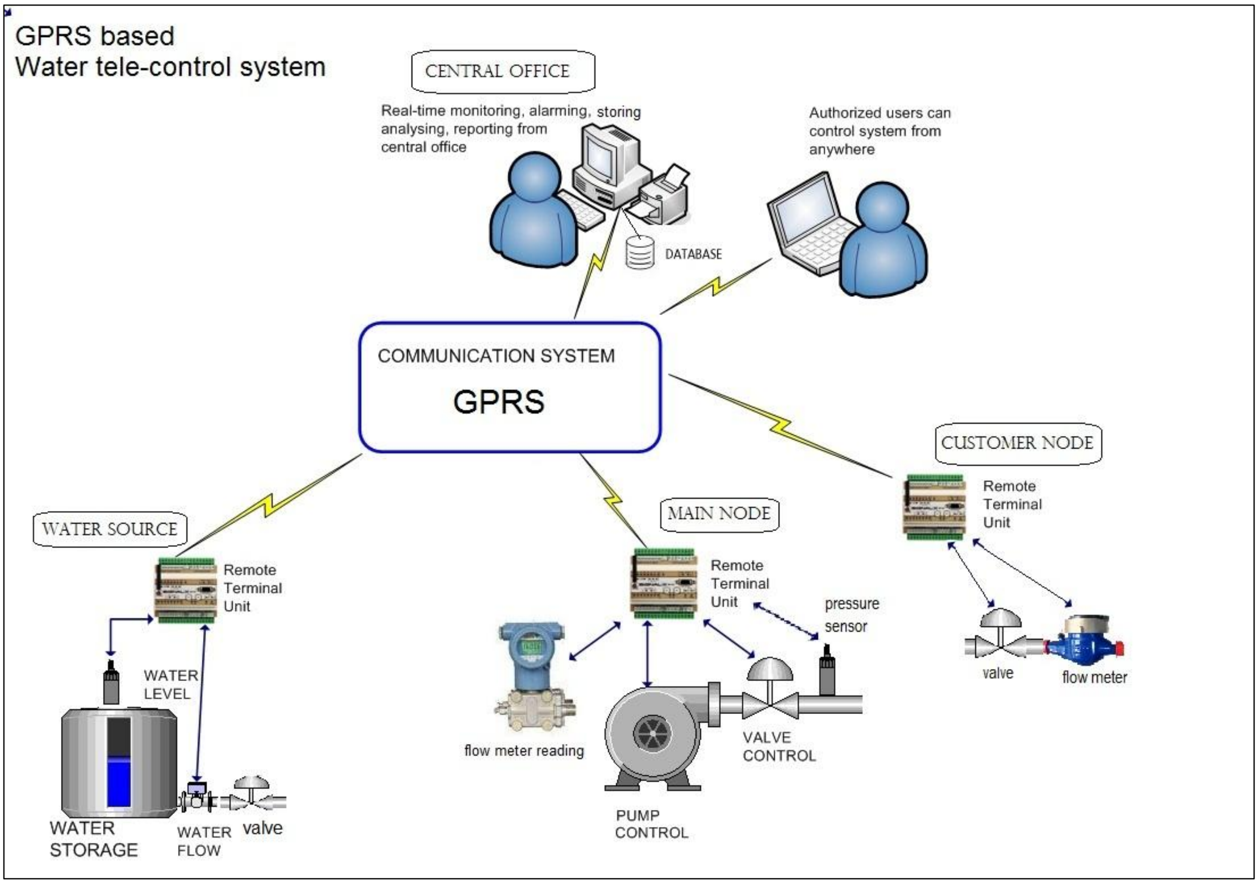


Figure (3.1)The System Design.

The figure above shows the system parts and how they are integrated with each other via GPRS network which is the system backbone. Also, the figure shows the type of each node and its components. That will be discussed in this chapter.

The system is divided into four subsystems:-

1) Water source node.
2) Main distribution node.
3) End-user node.
4) The central office.

3.2 Remote Terminal Unit (RTU)

Each node in the system will contain an RTU. It is the main part of the system representing the processing core and the communication equipment.

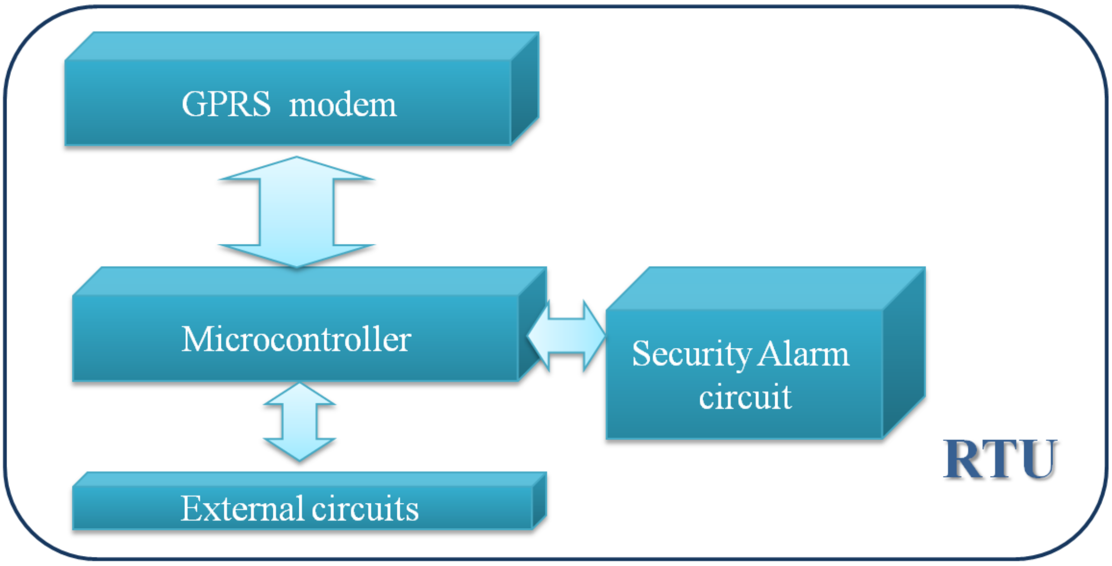


Figure (3.2) The RTU's Components.

The RTU is connected to the node's equipment. Typically, an RTU converts the electrical signals from the equipment to digital values such as the open/closed status from a switch or a valve, or measurements such as pressure, flow, voltage or current. The RTU sends these measurements to the central office and receives orders from it. By analyzing and sending these electrical signals out to the equipment, the RTU can control this equipment. The control includes: opening or closing a switch or a valve, or turning on/off a pump.

The RTU will contain two main parts which are the microcontroller and the GPRS modem, and some other external circuits. At the starting of the system the microcontroller will connect to the GPRS modem and initiate connection to the network.

The microcontroller in the RTU will perform all the processing on the data received from the sensors and valves. It also communicates with the modem to receive commands from the central office. It applies these commands on the desired part of the node. Another task of the microcontroller in the RTU is to observe the alarms circuits and inform the central office with these alarms. The RTU is also supported by an alarm circuit that notifies the PIC at any unauthorized attempt to reach the RTU using a door switch.

3.3 Water source node

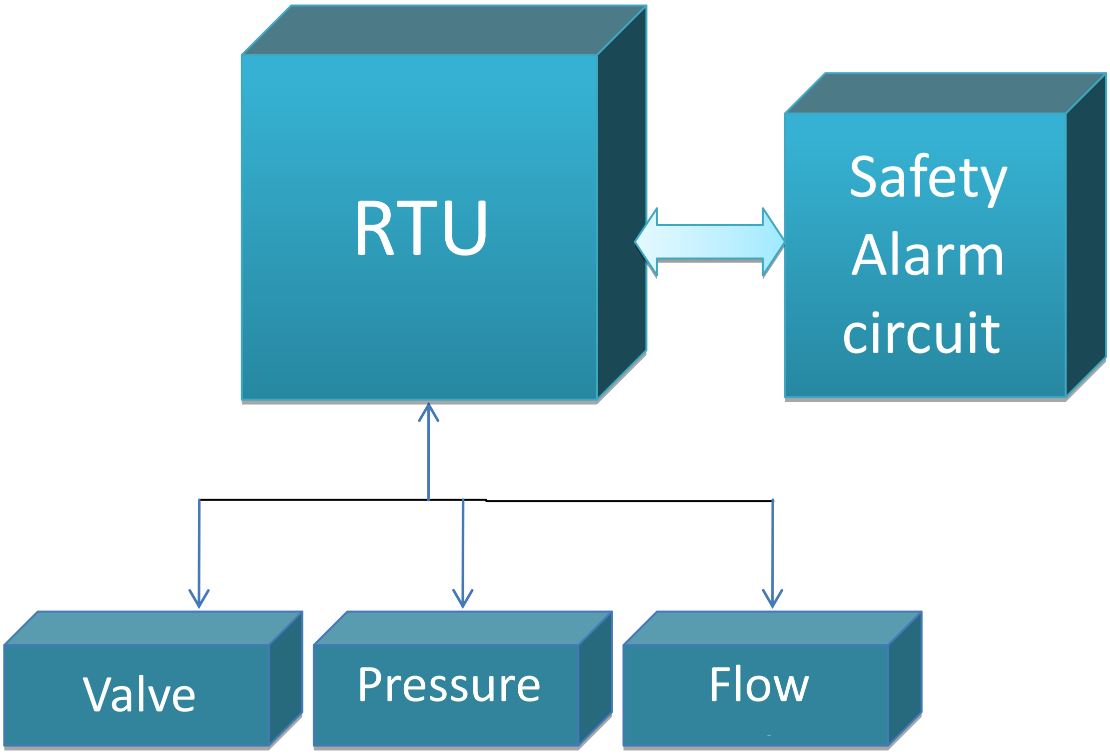


Figure (3.3) The Main Components Of The Water Source Node.

Water source is the main tank that contains the amount of water to be distributed across the whole network.

The main concern is the amount of water in the tank. Therefore, a pressure sensor will be inserted at the bottom of the tank getting used the cleaning pipe of the tank. It will measure the water pressure inside the tank. The microcontroller will use that value to calculate the amount of water.

The valve will be used to control the supplying of water in the network according to the command it receives from the microcontroller based on the distribution schedule prepared at the main office.

Also the status of the switch or a valve (on/off) may be sent to the central office upon request. All equipment will be kept in a box in the site. So a magnetic sensor (door switch) will be used to alarm in the case of any unauthorized attempt to access to the box content.

3.4 Main distribution node

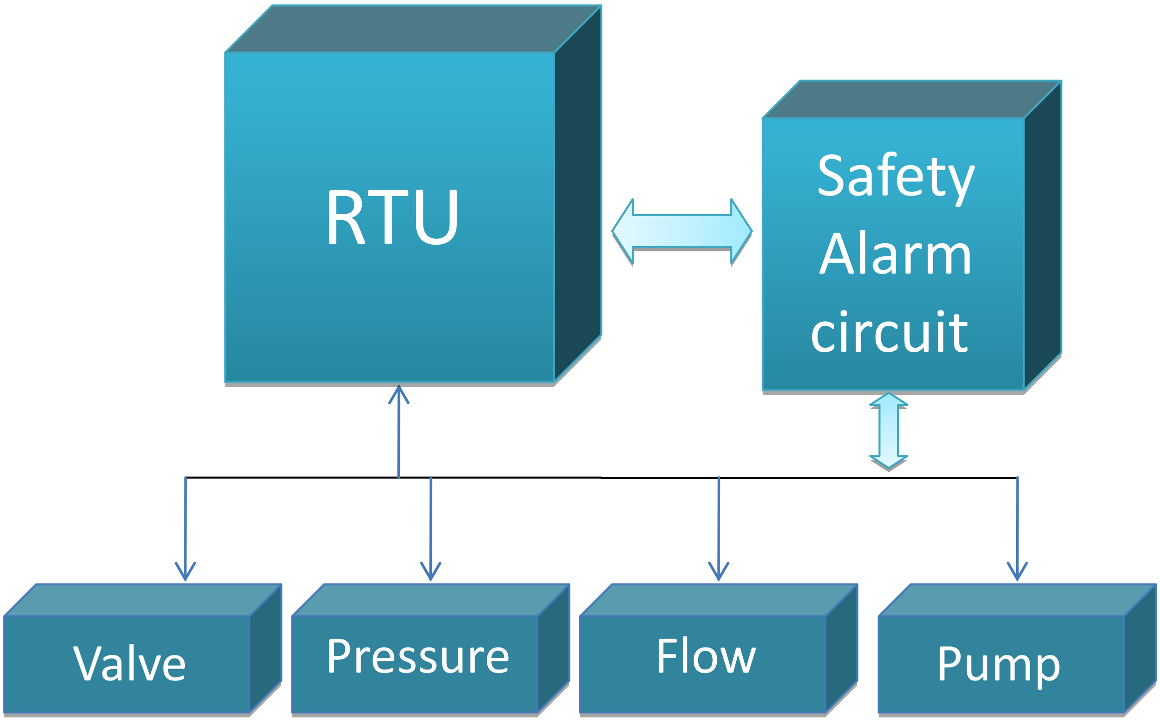


Figure (3.4) The Main Components Of The Main Distribution Node.

The main distribution node is the water gateway that is responsible for supplying water to a specific area. The equipment at the nodes differs from one to another depending on the nature of the terrain and the population density in the area. Water arrives to the main node in order to be distributed to other smaller areas according to orders from the central office. The main node also sends information, alarms and notifications to the central office to inform the operator that the system performs its functions without any problem.

At the main node there are some significant equipment that must exist to allow the main node to perform its operations. The valve is one of those equipment, that is needed to control the supply of water to each subarea from the distribution node by receiving orders from the central office. the flow meter is also a basic equipment in the main distribution node due to its turn in counting the amount of passed water through the pipe. Its function is useful for guarantee supplying a specific amount of water to a specific subarea from the main node.

The pressure sensor in the main distribution node is the last basic equipment that is needed to guarantee the safety of the network components, by alarming the central office when the pressure inside the pipes exceeds a specific safety threshold. Also the pressure is used to check whether the water have enough pressure to reach the end user or not. If not, the PIC will turn on the pump and notify the central office about that. The pump doesn't exist in all main distribution nodes. It's attendance depends on the terrain of the subarea.

Also, there will be a safety alarm at each node in the system to protect the equipment. Some main nodes equipment is located in the manholes, if any leakage in the water happens, these equipment will be damaged, so, a water level sensor will be used in these manholes to alarm the central office.

3.5 End-user node

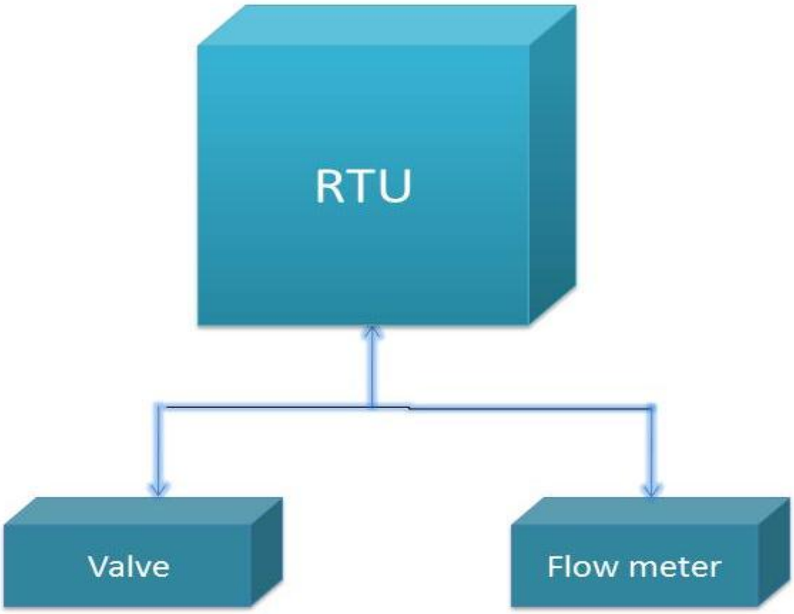


Figure (3.5) The Main Components Of The End-User Node.

The main purpose at the end user node is to measure the amount of consumption for each user, this can be done by two methods, the first one is to use a flow meter that generates a digital signal representing the water flow value that can be used to calculate the quantity of water by the following equation:

$F=(7.8Q-3)$ Q:quantity L/min F:frequency Hz (Equation 3.1)

The second method is to use the analog water counter that the end user is already has. A small sensing circuit will be added to it, in order to get an electrical signal for its reading.

A valve will also be used at the end user node to control whether to supply the water or not. The PIC will have a monitor ability over the valve status that will be used to ensure that the valve is responding or not.

3.6 The Central Office

It's the main node of the system. It will have a GPRS modem connected to the PC to allow it to communicate with the other nodes of the system. It will have a special software that will allow the operator to monitor and control the network. The software will contain a database, GUI, and several code functions.

The system can be partially controlled from anywhere by using a portable PC supported by almost the same software and connected to a GPRS modem.

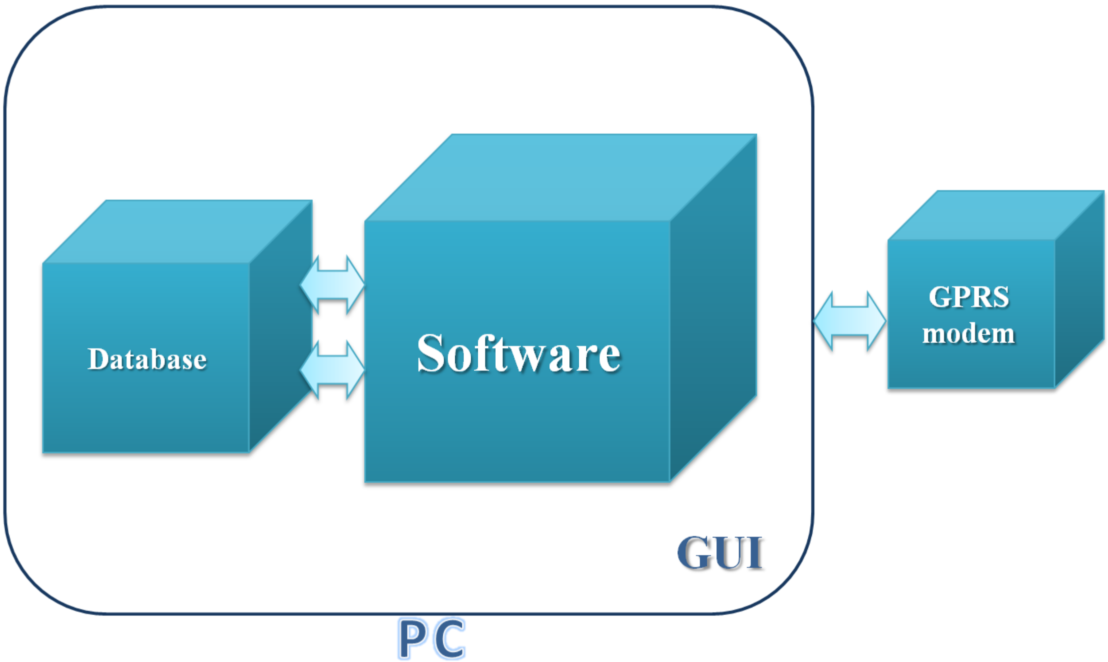


Figure (3.6) The Basic Components Of The Central Office

3.7 Software Design

The node types in the system are different. Each one of them has its own functions and its own parameters. The software will be able to identify the parameter’s type of each component in any node and order it.

3.7.1The Software of the Central office

3.7.1.1 The Main tasks

One of the main tasks of the software is establishing the connection with GPRS modem at the central office. The needed AT commands to establish the connection are the core of this part.

The software needs to collect information and data about the whole network, and store it in the corresponding database, so the software sends requests to the other nodes asking for the required information or ordering some functions to be performed.

From the water source node, the software allows the operator to order the amount of the water in the tank, the flow meter and the pressure sensor readings. At the main node there is also a flow meter and a pressure sensor. The software will be able to order their parameters. Also, at the end user the amount of consumption can be ordered.

The most common equipment between the nodes is the valve. There is a valve in each system node. The software enables the operator to control (open, close) any valve in any node from the main source until reaching the end user. Also, the software allows controlling the pumps in order to deliver water to some high areas.

3.7.1.2 Database

A database will be needed as an archive. This archive will have the history of each parameter in the network. Adding and updating data will depend on the type of each parameter. For some types, the new data will update the old one. In the others, the new data will be inserted as new items. The time and the date will be added or updated at each new data action.

At the end user node, the amount of consumption and the status of the valve will be added to the archive as new items. Also, the same will happen to the main node valve statuses and flow meters readings. Some of the parameters will not be included in the database because there is no need to store them like the reading of the pressure sensor. At the water source node, the amount of water in each tank is needed to be stored. Also, the state of the valve and the flow meter read. All these data will be added to the database as new items.

The code of the software will perform some calculations over the data in order to offer the operator some alarms like the leakage and theft alarms. The software will compare between the total consumption of the users in an area with the amount of supplied water to this area. If the result exceeds a specified threshold, a leakage alarm will appear on the software screen. The appearance of the theft alarm will have some calculation over the amount of consumption for each user, taking in concern the time of consumption.

One of the main functions of the system is distributing the water according to a built schedule. This schedule will be saved in a database and the valves will open and close accordingly. This schedule can be built by the operator or automatically depending on the available amount of water and the percentage of usage for each area in the network.

3.7.1.3Graphical User Interface

A GUI will be added to the software. This will ease the operation of using the different software functions. By the GUI, the alarms will be noticeable for the operator. Ordering any parameter from the database will be performed simply. Also, building a distribution schedule or opening and closing any valve will be performed by just a click. Any function over the system will be performed in an arranged eased way using the GUI.

3.7.2 The Microcontroller's Software

The software of the microcontroller, that will be designed, will make the microcontroller capable of receiving data from the different system node's elements. It will receive the pressure value from the tanks, and then the software will be able to calculate the amount of water in that tank. Also it will receive the flow meter readings and calculate the amount of water delivered to a certain area, or calculate the water consumption.

The software will make it easier to the microcontroller to discriminate the receiving requests from the central office and send back the appropriate data.

4

Chapter Four

Detailed System Design

4.1 Introduction

This chapter will focus on the software and hardware design of each part of the system independently.

The software part takes in consideration the design of the central office PC program step by step. It will include the way of transmitting commands, receiving acknowledgments and data, and it will store the data in a database. This part also details the design of the PIC software, it focuses on the way that the PIC communicate with the peripherals, and the method of treating their data. In order to explain that, flowcharts have been used to clarify the idea behind each design simply.

A detailed hardware design will also be shown in this chapter. It includes the design of the circuits that have been used for several purposes, as the alarms, PIC interface and the protection of the PIC from current leakage.

4.2 Software design

This section describes the software part in the project which includes the PC (central office) software and the PIC software in details.

4.2.1 Central office software

The central office software can be divided into two parts, the basic functions that includes adding and modifying information, and the communication process that includes the communication details.

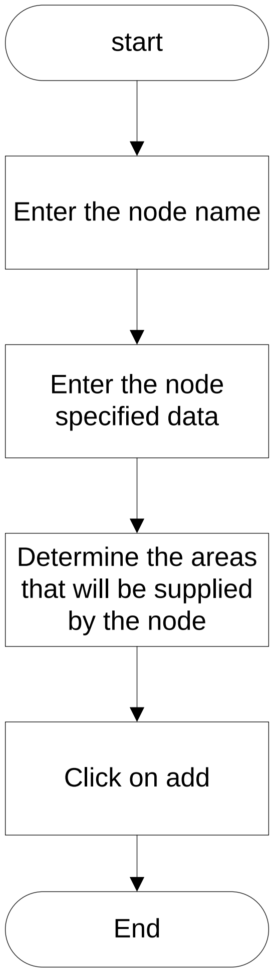
4.2.1.1 Basic Functions

At the beginning, the user of the software program must log in the program with his name and specific password in order to use it’s functions.

After checking the user authority, several functions can be done as following.

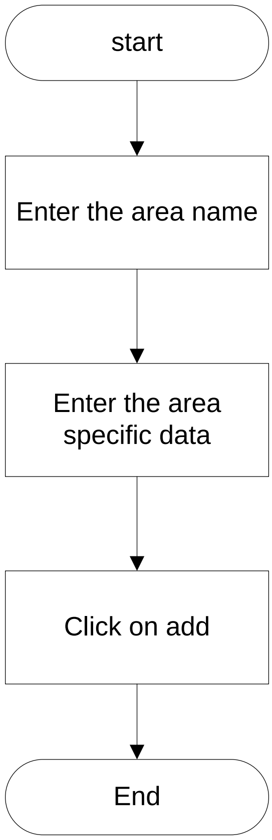
Network nodes and area:

The program gives the user the ability of adding a new node and areas to the software when needed as shown in the following tow flowcharts.



Figure(4.1) Adding New Node To The System flowchart.

The user of the program in this case must enter the name of the new node with a specific data about the node wants to add such as the node address, the node IP, the node type if it's a main area or sub main or small area, also the number of valves and flow meters it contains, and then determine the areas that must be supplied by each valve of the node. The user also can remove or search about a certain node by entering the same specified data, and can modify the node IP and type or adding a new valves and flow meters to the node.



Figure(4.2)Adding a New Area To The System.

When adding a new area to the system, the user must enter some data about it such as the area name, area type if it is a main area or sub main area or small area. And if it is a sub main or small area, then the user must determine the main area it belongs to. Also the user must add the average demand, the percentage it needs of the total water from the source. After that the software will give a second priority to any new added area.

The first priority will be given by the software to the areas that the water supplying starts for it, and before the supplying finish, the user may pause the operation for some reasons and stops the water supply. When the user starts the operation again, the software will continue the water supply for that areas, which have the first priority.

When some areas take their need of water and the supplying ended for them while the water supplying continued for the rest areas, a third priority will be given for them by the software.

Customers:

As the number of customers in the network increase the software allows of adding customers to the program and controlling their information as the following flowchart explain.

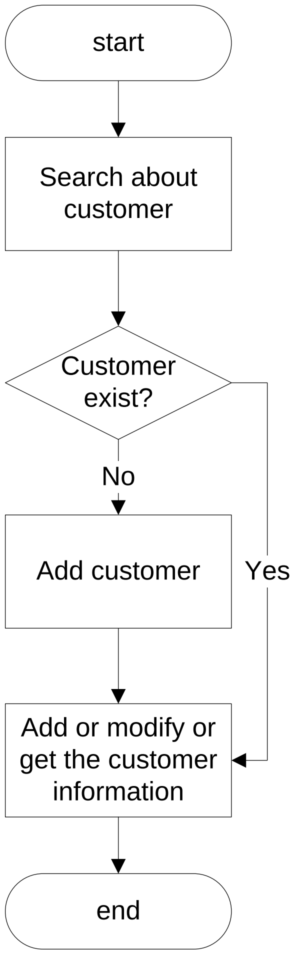
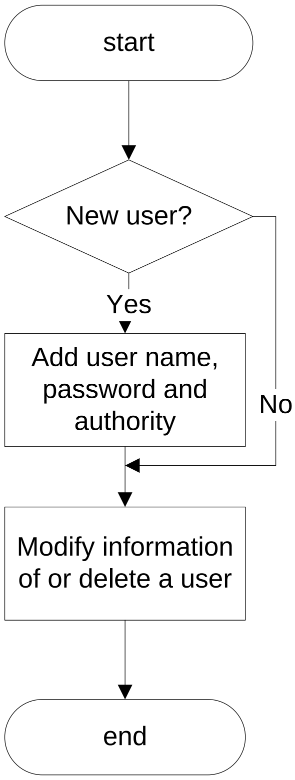


Figure (4.3) Adding A New Customer.

The user can first search about the existence of the customer in the program database by entering a specific information about him. If the customer does already exist, then the user has the choice of getting his consumption or modifying his information or even remove it . If the customer doesn't exist and is newly subscribed, then he can add him with the specified information like his name, address, area, telephone number and subscription date and type.

System users:

The software allows of adding users of different authorities: admins, supervisors or normal users to control the network from the central office. The admin has the power of using and controlling every part of the program, and the normal user has the least control of the program.



Figure(4.4) Adding A System User.

As can be seen from the flowchart, if a new user of the program is added, then his name, and a specified password for him will be added, also the authority of this user must be determined. The software allows of modifying the user password or authority in any time, or even delete the certain user from controlling the software.

4.2.1.2 Communication Process

The program in the central office is responsible for doing the main tasks and decisions in the system, and will support the GUI as explained in the previous chapter.

The connection setup:

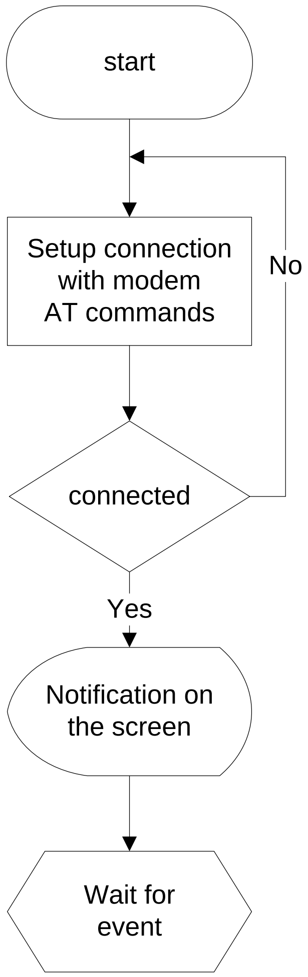


Figure (4.5) The Connection To The Modem in The Central Office Program.

At the beginning of the program a special function will initiate the connection to the GPRS modem.

The program will setup the connection with the modem by using AT commands, which will need a definition of the baud rate (9600 kbps), packet volume (8 bits), and other information required for the connection. As the connection completed the program will present a notification message on the screen and wait for an event to occur.

The data request in the central office:

As explained before the central office will request the data from the different system nodes of the system, the following flowchart shows exactly this process.

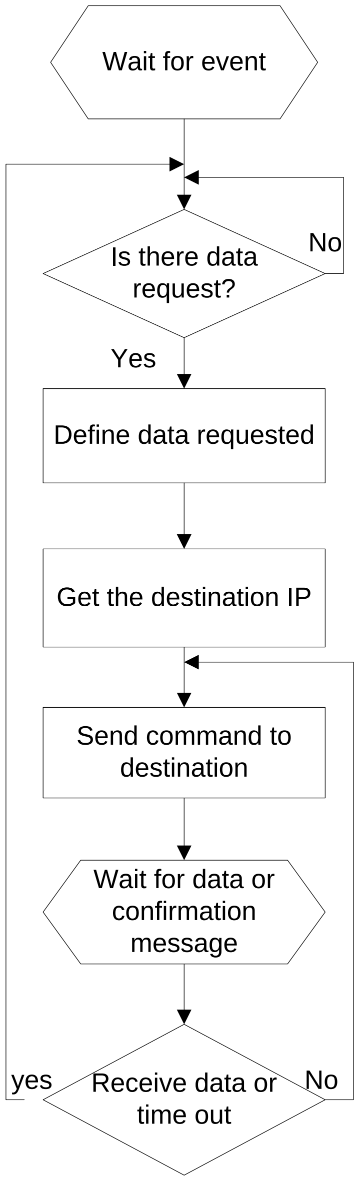


Figure (4.6) The Data Request In The Central Office Program.

The request module will be in the standby mode until the officer or the program request a data by applying a command. So the program will keep checking if there is any request, then analyzing, defining and sending the request to the node, after getting the destination's IP.

The program will wait for an acknowledgment or data for a certain period of time. If the command doesn't arrive the PIC of the node, the acknowledgment will not be received by the central office at the time, so the program will send the request again.

The data reception

The reception process in the program is shown in the following flowchart.

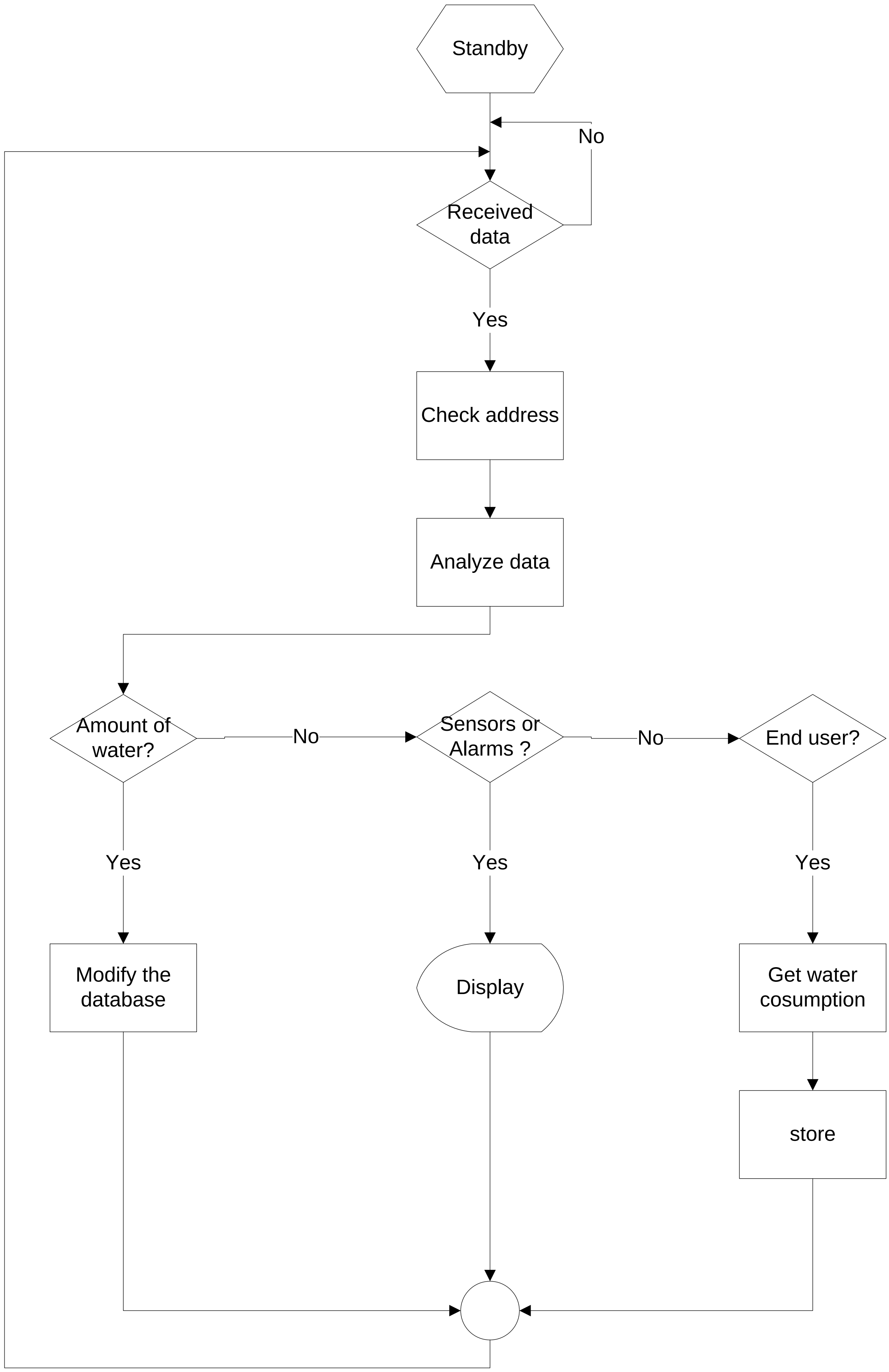


Figure (4.7)The Reception In The Program At The Central Office.

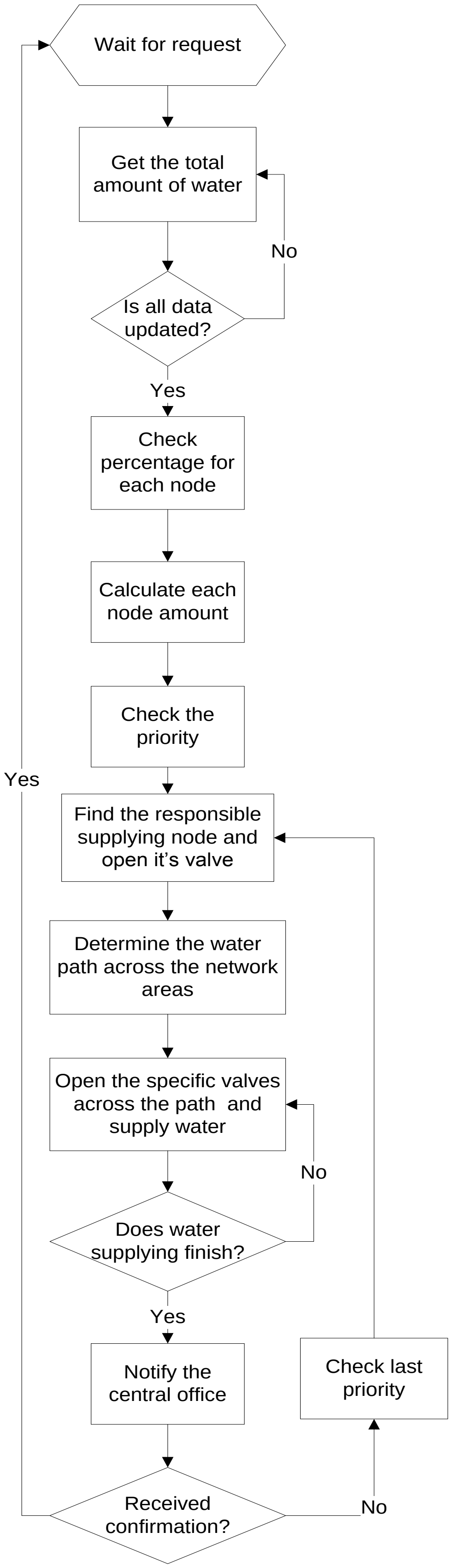
The program will keep waiting data from the modem, and if any data is received, the program first check the address of transmitter as the first step; to define the node that sending the data. Then it will analyze the data. If the data was the amount of the water in a tank, the program modifies the database of that tank.

If the data was from the end user node, then the program will get the water consumption and store it in the data base with the date and time.

If the data was a valve state or a sensor reading as a pressure, flow, and alarm, then the program displays it on the screen.

The automatic And Manual Water Distribution:

The water distribution can be done in a manual way by the program user, or by an automatic way as the next flowchart shows.



Figure(4.8) Automatic Distribution Of Water flowchart.

In the automatic distribution, at the beginning, all water quantities of water sources must be updated in the database, after that the program will check each area percentage to get the amount that it needs of water, then find the responsible valve specified to supply the area by determine the node it belongs to, then find the specified way the water will pass till reaching the area by determine the higher order areas and nodes the water will pass through and give an order to its valves to open. And after completing the water supplying, the valves will be closed and a notification to the central office will be sent.

In the manual distribution, the only difference is that the user of the program himself can determine the amount of water each area needs and give the order to open the valves.

4.2.2 PIC software for the nodes

As explained in the previous chapter, each node in the system will have a PIC that performs different operations depending on the task of that node.

The PIC Connection To The Modem:

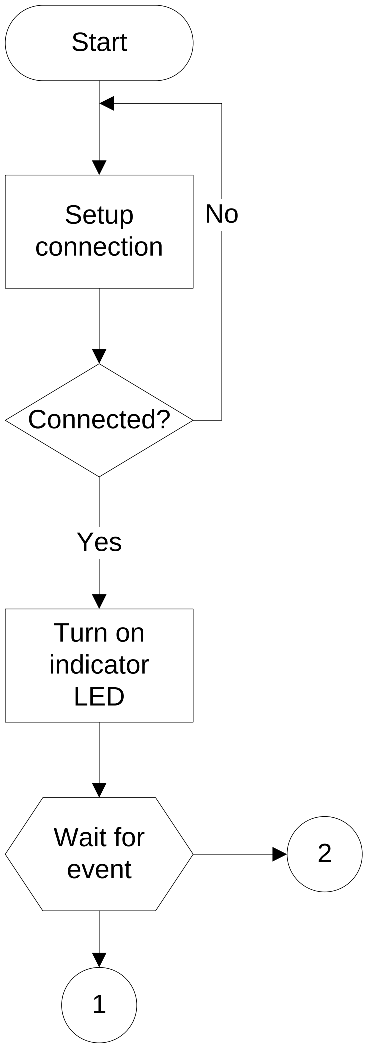


Figure (4.9) The Connection To The Modem In The Nodes.

First of all in each node, the PIC needs to setup a connection with the GPRS modem, as shown in the flowchart above, by using AT commands the PIC initiate the connection, and define the baud rate (9600 kbps), the packet size (8 bits), and other needed information. If the connection succeeds then a LED indicator will turn on as a notification. Then the PIC will wait for an event coming from the modem or sensors.

Receiving Data:

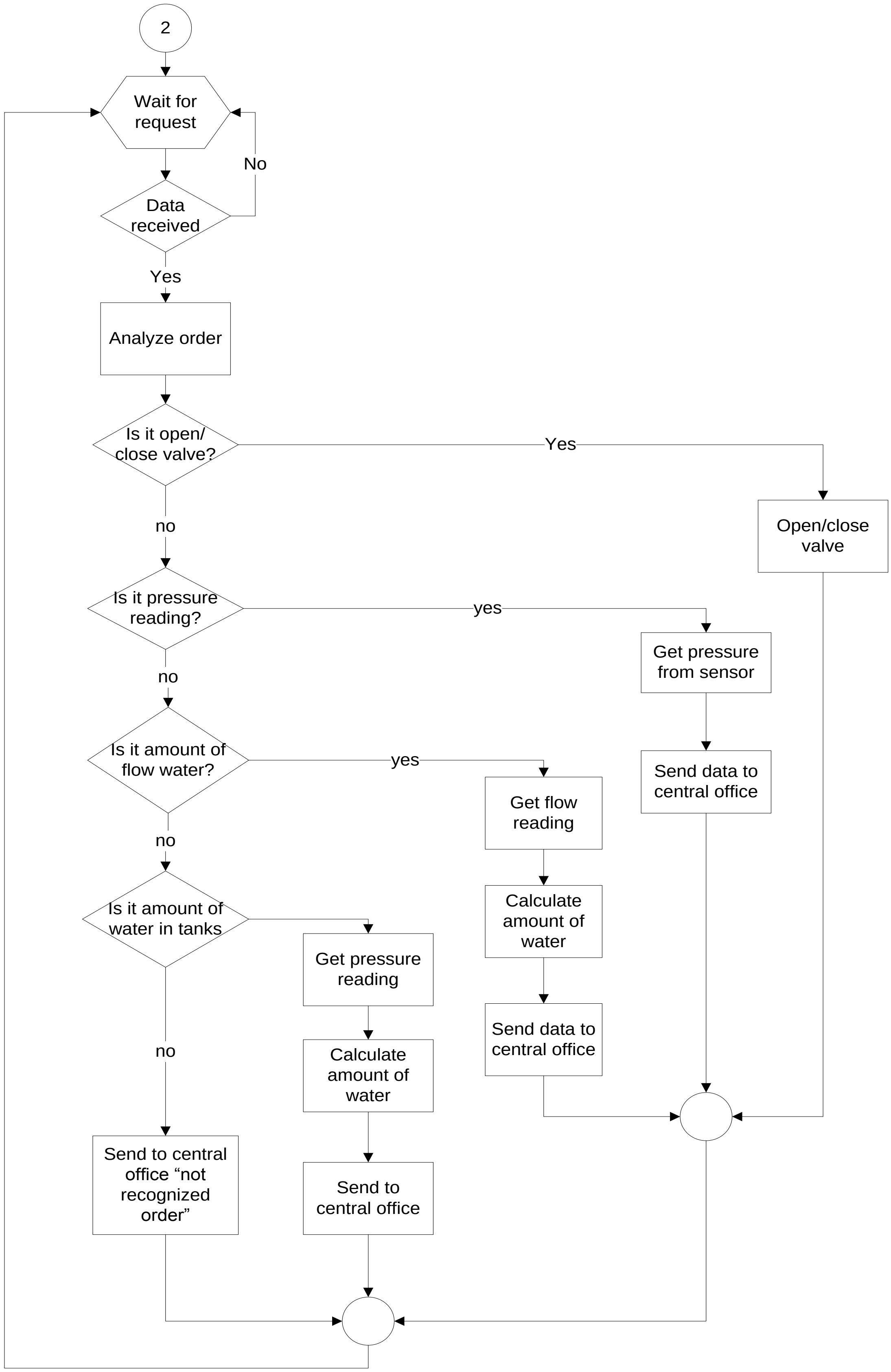


Figure (4.10) Wait From Modem In RTU.

The PIC in the nodes will also keep waiting a command data from the modem, if the data received then the program will analyze it to get the commands, and If it was an open/close valve the PIC sends an open/close signal to the valve, if it was a pressure reading then the PIC gets the pressure value from the sensor and sends it to the central office.

If the data required from the central office was the amount of water, then the PIC will get the flow mater reading in a defined period and calculate the amount of water passed from the node and send it to the central office.

If the requested data was the amount of water in the tank, then the PIC will get the water pressure value from the sensor at the bottom of the tank and calculate the amount of water in that tank, and then sends it to the central office.

The main Node PIC Program:

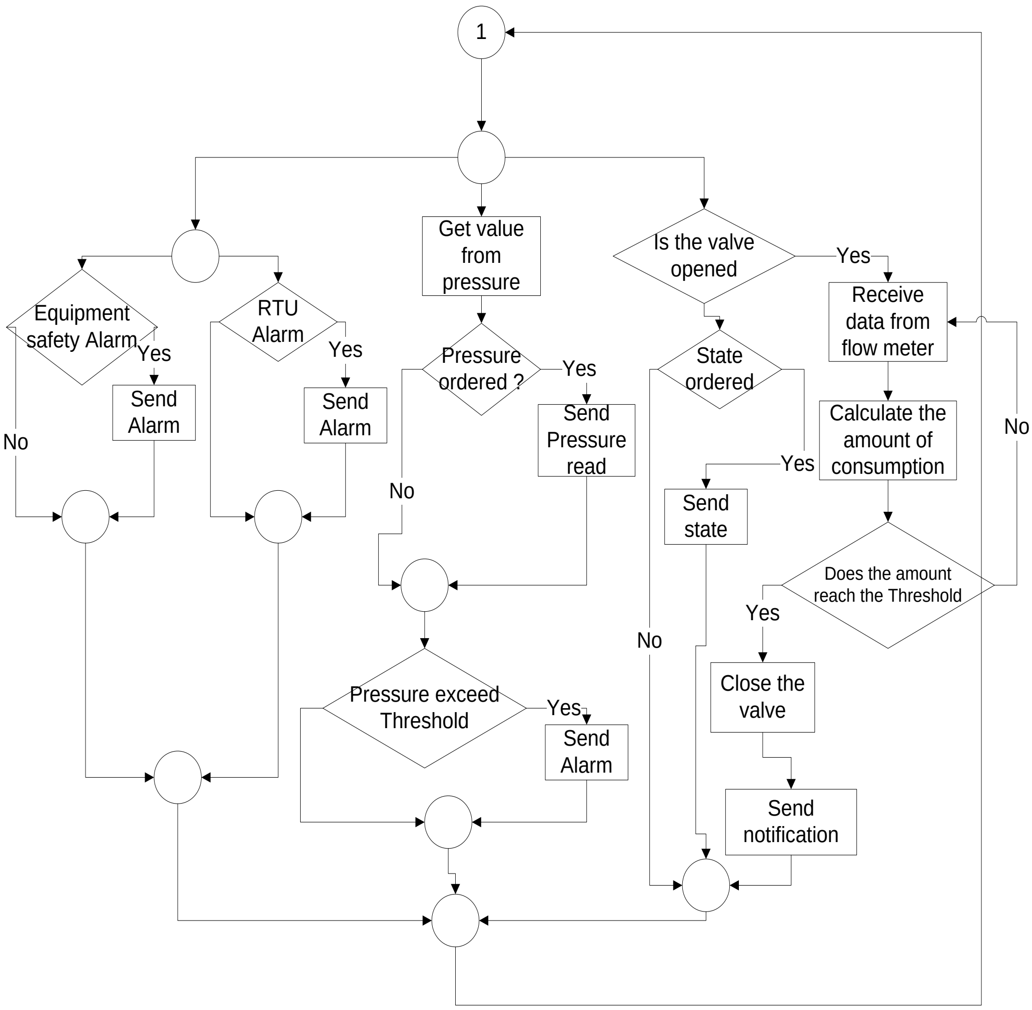


Figure (4.11) Main Node PIC Program.

Also, the PIC will wait for any data from other peripherals, some of the data are in parallel and others will receive in a form of an interrupt. The PIC will get the pressure value all the time and check if its required from the central office in order to send it, also will check if the pressure value exceeds a determined threshold or not, in order to protect the equipment, if it exceeds the threshold, the PIC will send alarm to the central office. In parallel to that, the PIC will get the status of the valve, if it requested then the PIC will send it to the central office. If the valve was opened for a certain time, the PIC will keep reading the flow of water in the pipe from the flow meter, until the flow reaches a very small value (nearly zero), then the PIC will close the valve and calculate the amount of water passed within that time and sends it to the central office.

If an alarm occurred, then a signal will interrupt the operating PIC program and send an alarm to the central office.

The End User PIC Program:

The end user node will have an RTU to get the amount of water consumed, and to control the valve. The flowchart bellow shows the end user's PIC programming.

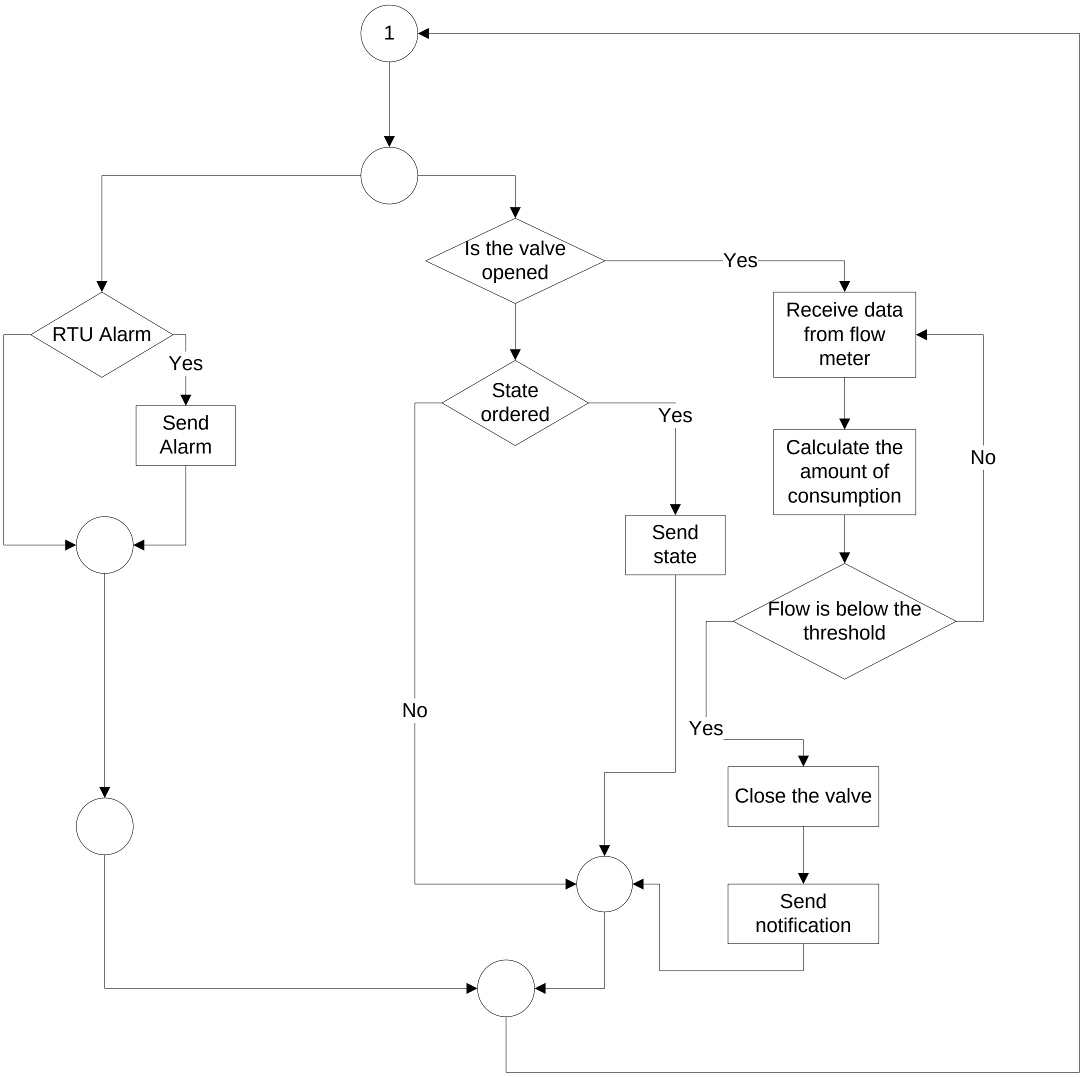


Figure (4.12) The End User PIC Program.

The PIC checks the status of the valve all the time, if it was opened then the PIC gets the flow reading from the flow meter as long as there is a flow of water in the pipe. Then calculates the amount of water consumed by the user and sends it to the central office. After sending the amount of water value, the PIC will reset the flow reading in its memory, and start counting again when the flow comes back. And if any alarm occurs, the central office will be notified.

4.3 Project Equipment

4.3.1 GPRS modem

There are different types of GPRS modems, that differs in the interface and module types. In this project the SM5100b-D modem will be used due to its capability of supporting the AT command that is needed to allow the PIC and the PC to communicate with the modem. In addition to that, it has an economical power consumption. The maximum output current is 50 mA and the maximum output voltage is 3.0 v which can be used to power some external functions, such as LCD. This module can be integrated into a great number of wireless projects. It can be used to accomplish almost anything, including GSM/GPRS and TCP/IP services.

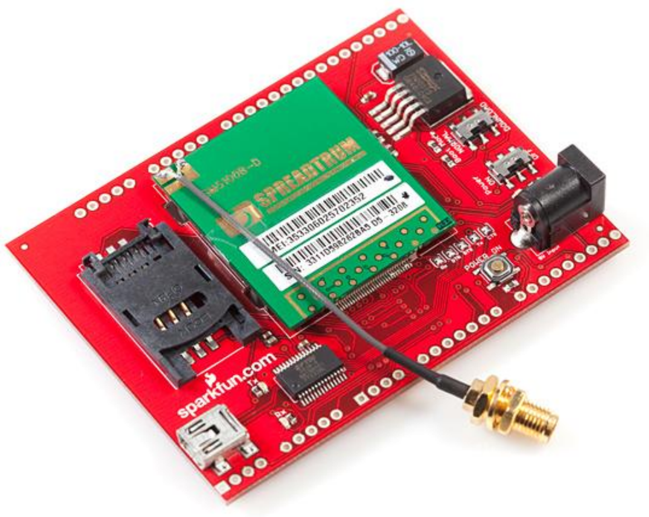


Figure (4.13) GPRS Modem.

4.3.2 Water valve

The valve that has been used in the system is a pilot valve, this type of valves performs the closing function by blocking the inlet port depending on the water pressure at the other side, this helps in offering an accurate valve closing using the water pressure over the blocker, also the used product is a zero pressure valve that works in a low pressure mediums and offer reliable functions. The pilot valve is a self holding valve that needs a one pulse to open the valve and another one to close it.

Figure (4.14) The Valve.

4.3.3 Flow meter

The main advantage of the used flow meter is that it works at low pressure medium and offers a digital output signal for the flow of the water depending on the output signal's frequency. In addition, the output voltage is 12 VDC that allows using a battery to supply it in the far areas.

Figure (4.15) the Flow Meter.

4.3.4 Pressure sensor

The main parameters that considered for choosing a pressure sensor is the kind of the output signal and the pressure range, the used sensor has been chosen to output a signal that satisfies the PIC inputs (0-5) volts. And a pressure range of (0-1) Bar satisfying the low pressure value in our prototype.

Figure (4.16) Pressure sensor.

4.4 Hardware design

This part will deal with the design of several electronic circuits that will be used to perform safety and security functions and also to work as an interface between the PIC and some peripherals.

The following circuit uses the opto-coupler to perform the separation in electrical signals between the PIC and the flow meter. This method has been used to protect the PIC from any high voltage or high current signals that can be produced by the flow meter if any problem happened.

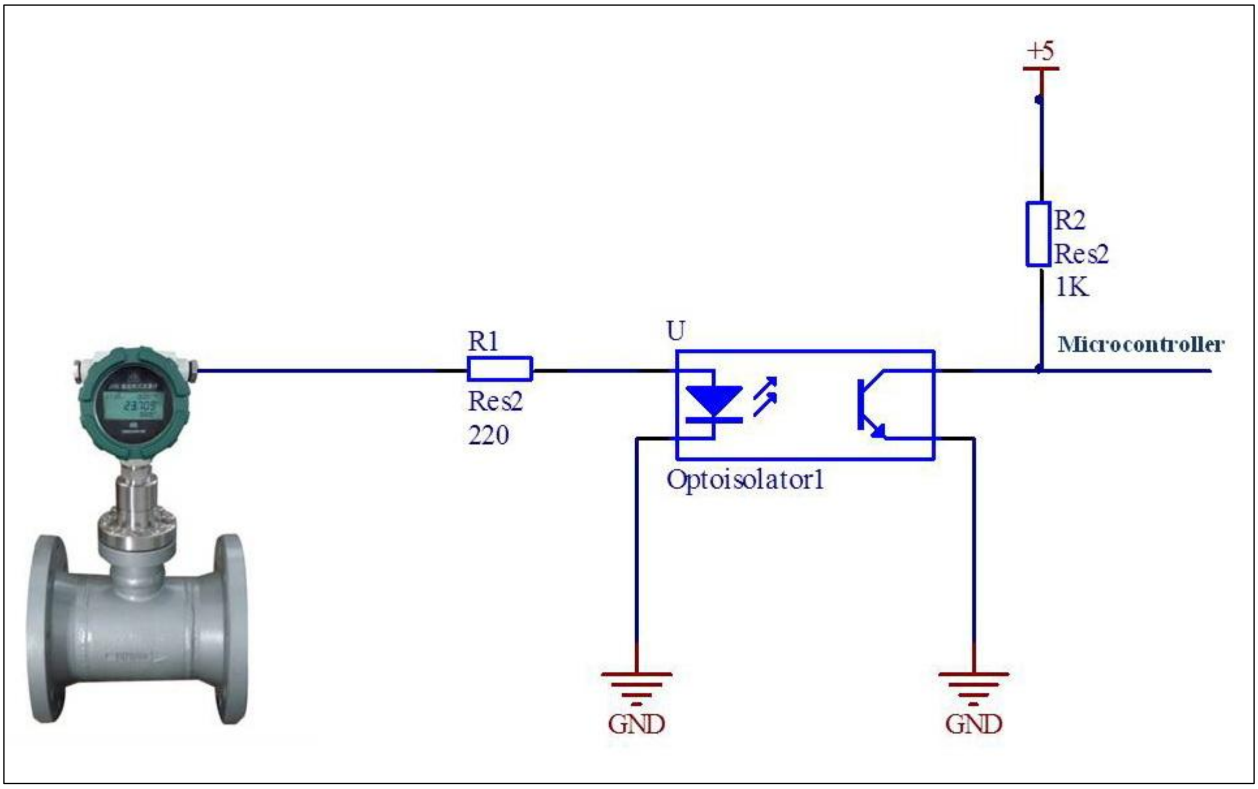


Figure (4.17) Flow Meter Separation Circuit.

The opto-coupler transfers the flow meter digital signal to the PIC input with no need of delivering the same electrical signal from the flow meter to the PIC. A 220 Ohm resistor was used at the cathode of the opto-coupler LED to protect it from the voltage increment. Also another resistor 1KOhm was used at the collector of the phototransistor, this resistor was used to drop the V_{cc} voltage across the collector in case of zero digital signal “zero volt”.

At some nodes of the system, there is a need to add a pump that push water in the pipes to guarantee delivering the water to some high areas.

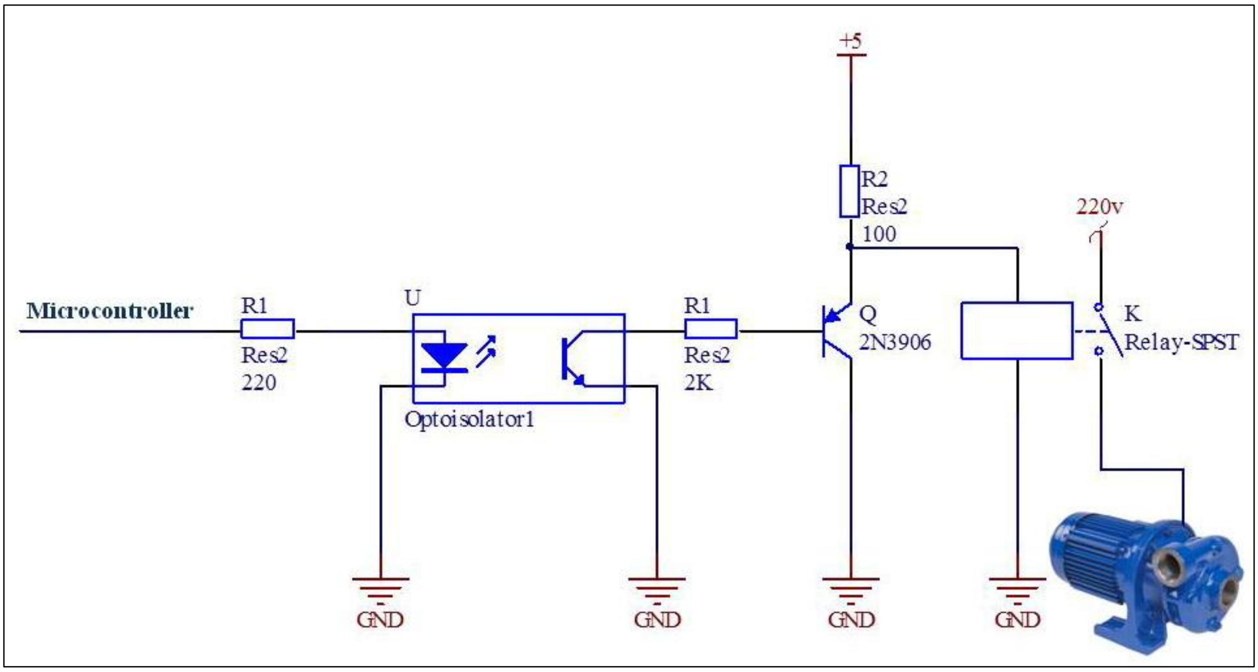


Figure (4.18) Pump Circuit.

The pumps is working over 220v ac voltage. This voltage can't be supported by the PIC or by an opto-coupler, so a relay is needed to perform the on off function over the pump from the PIC. And, in order to perform a PIC protection, an opto-coupler will be used to offer that as before, but it must be taken in consideration that the opto-coupler can't hold the current that is needed by the relay. So, a PNP transistor was added to perform as switch transistor and offer the relay required current. This circuit is also valid and will be used with the valves of the network to perform controlling it open or close.

At the end user nodes, there is already an analog water counter that can be used to count the amount of consumption without needing to add a flow meter. A small sensing circuit will be added to the water counter to have a digital value from it as it shown in the following figure.

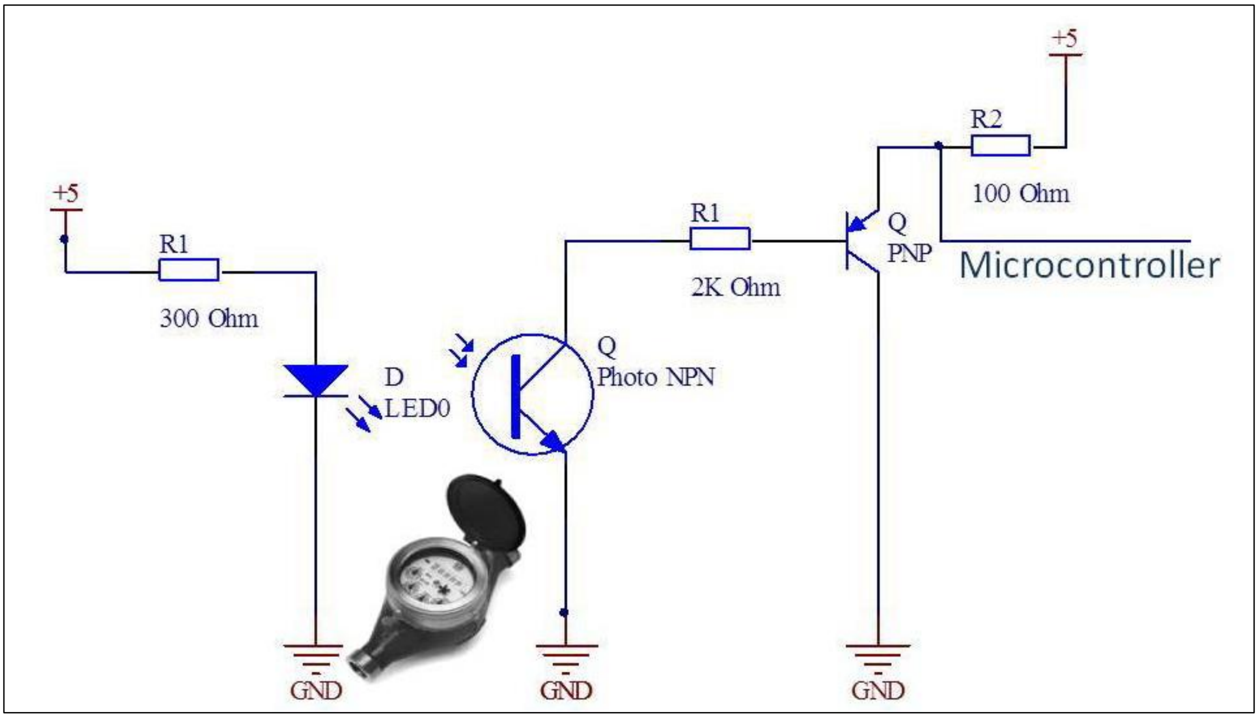


Figure (4.19) Analog Water Counter Circuit.

A small size black and white transceiver has been used over the pointer of the lowest range. This transceiver differs between white and black colors, which output high voltage at the white one and low voltage at the black one. The sensitivity of this transceiver reach 1400 rpm that is suitable for reading the fast pointer and offering a high accurate reading. Also, a PNP transistor has been used to protect this transceiver from high voltages that may damage it, sense it is so sensitive for the high voltages.

One of the main tasks of the hardware circuits is offering an alarm at any attempt of an unauthorized entry to RTU at any node in the system. The system can differs between the authorized and the unauthorized entry by a notification from the software to the required RTU. The figure below shows the circuit that offers this kind of alarms at the door of the RTU.

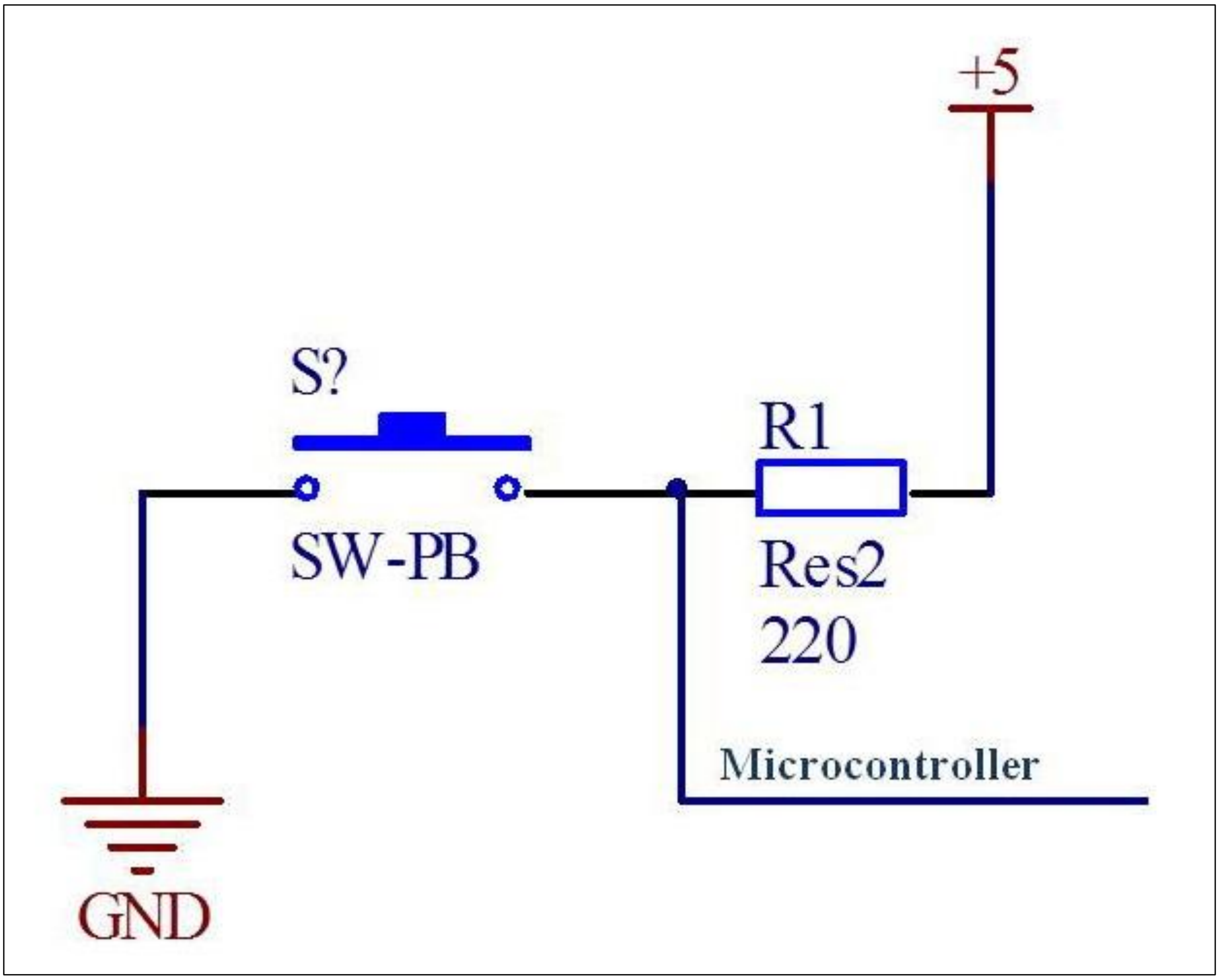


Figure (4.20) RTU Security Alarm Circuit.

The idea behind this circuit is simple. It depends on a normally open switch that gives a falling edge digital signal at any attempt to open the RTU since the switch is fixed at the door edge. The PIC can analyze this alarm wither it is authorized or not and alarm the central office when needed.

All of these circuits above and some extra equipments will be connected to the PIC, that acts as a brain that control all of them. The PIC also acts as an interface between the node components and the other network nodes through the GPRS modem. The figure below shows where each circuit or peripheral could be connect on the PIC.

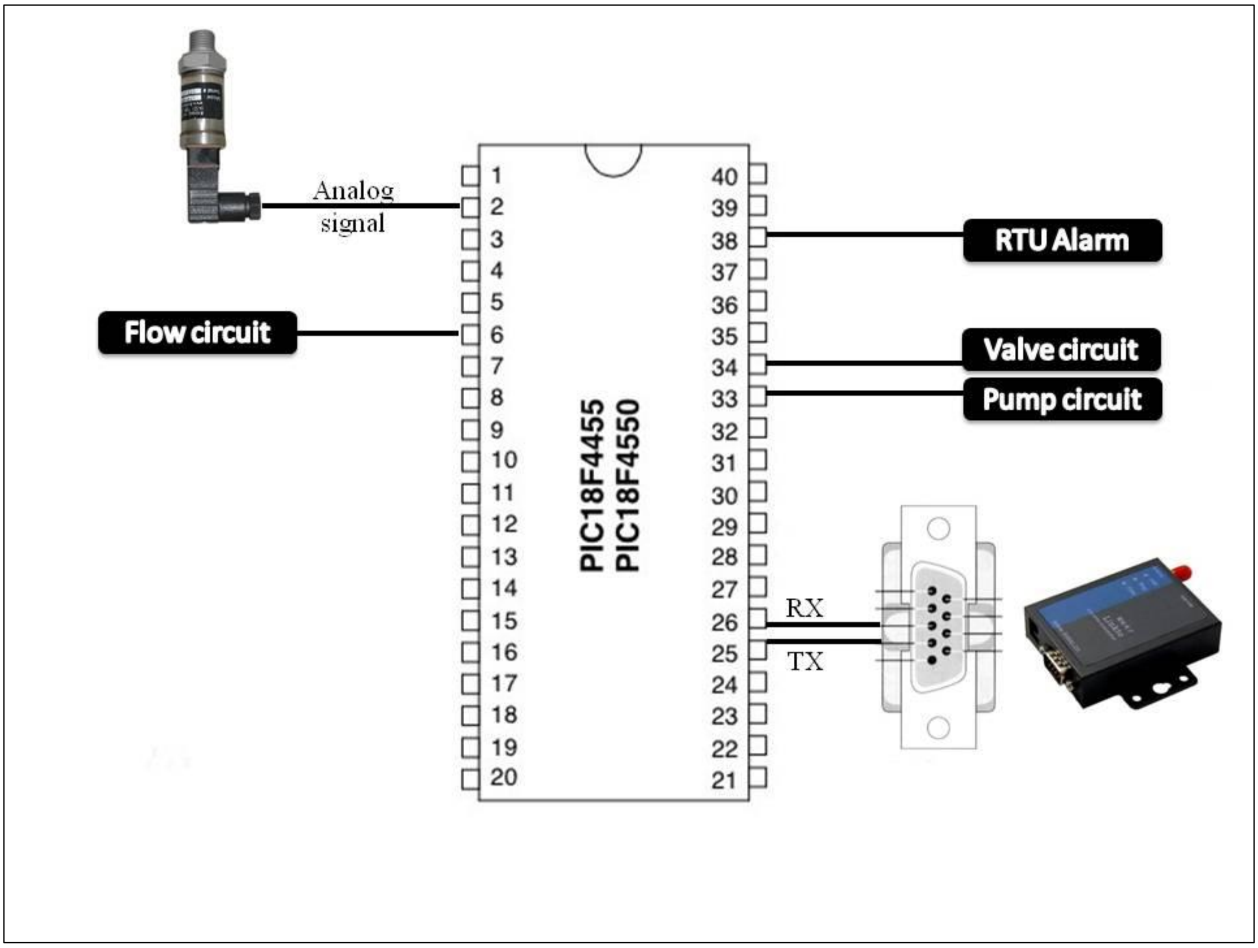


Figure (4.21) PIC Connections.

Not all of the above circuits must be used on each PIC, may be one of them added instead of the other like the flow meter circuit and the analog water counter. Each one of them is connected to the pin6 that acts a the counter to count the amount of input pulses and calculate the amount of water from it. The pressure sensor has been connected directly to the PIC, no protection circuits is needed due to the 5 volts maximum voltage that the microcontroller can perform without any problem. Pin2 is an input for an analog to digital converter, so the pressure sensor has been connected to it in order to analyze its analog signal.

Pin25 and 26 is special for serial input and output from the PIC. So, they have been used with the GPRS modem always to enable the PIC communicate the central office. The remained circuits like the valve, the pump and the Alarm can be connected to any Data input output port on the PIC depends on some declaration in the software.

5

Chapter Five

System Implementation

5.1 Introduction

This chapter describes the hardware and software implementation of the system. The system can be divided into three main parts: Microcontroller and sensors part, C# software program part and GPRS communication part. Each one of these parts will be described in details in this chapter.

5.2 Sensors and PIC interfacing and programming

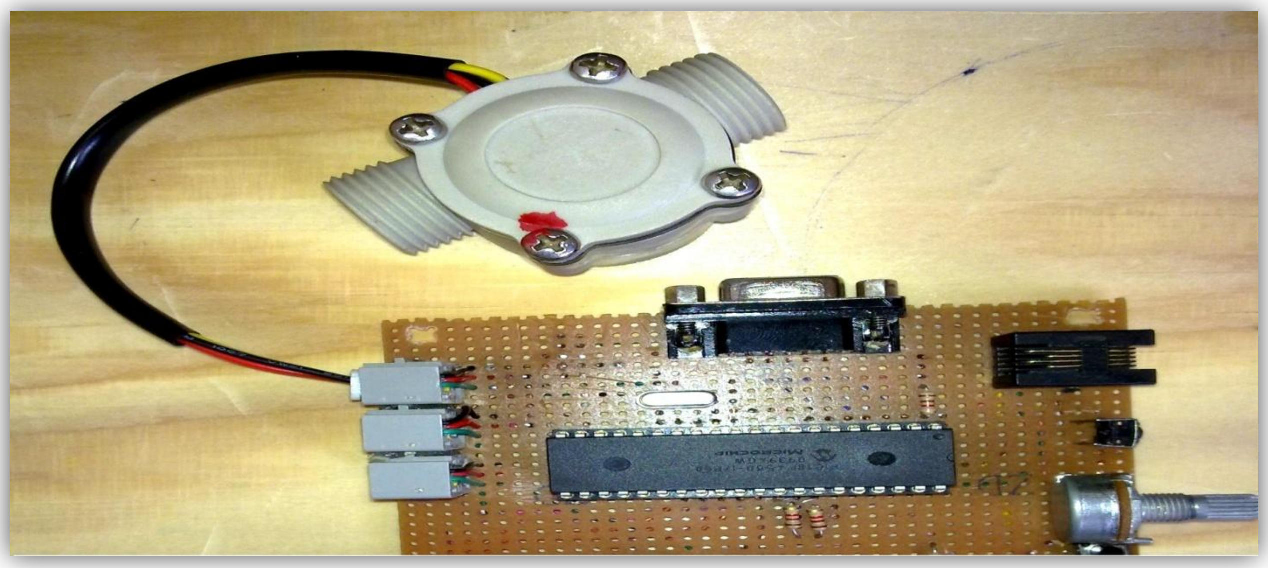
PIC 18F4550 is the main component used to process the signals it receives from the sensors distributed all over the water network nodes, these signals describe the water network status. To make the microcontroller be able to read the input signals correctly and respond accordingly different interface circuits will be used to connect these sensors with the microcontroller.

C18 programming language and MPLAB IDE editor are used to write the different programs for the PIC.

5.2.1 PIC and digital flow meter :

Flow meters are used to calculate the amount of water being supplied to a certain area. As water flows through the flow meter a small disk will rotate, then it will give a frequency value proportion with the amount of water passed through it, so Amount of Water = F(frequency).

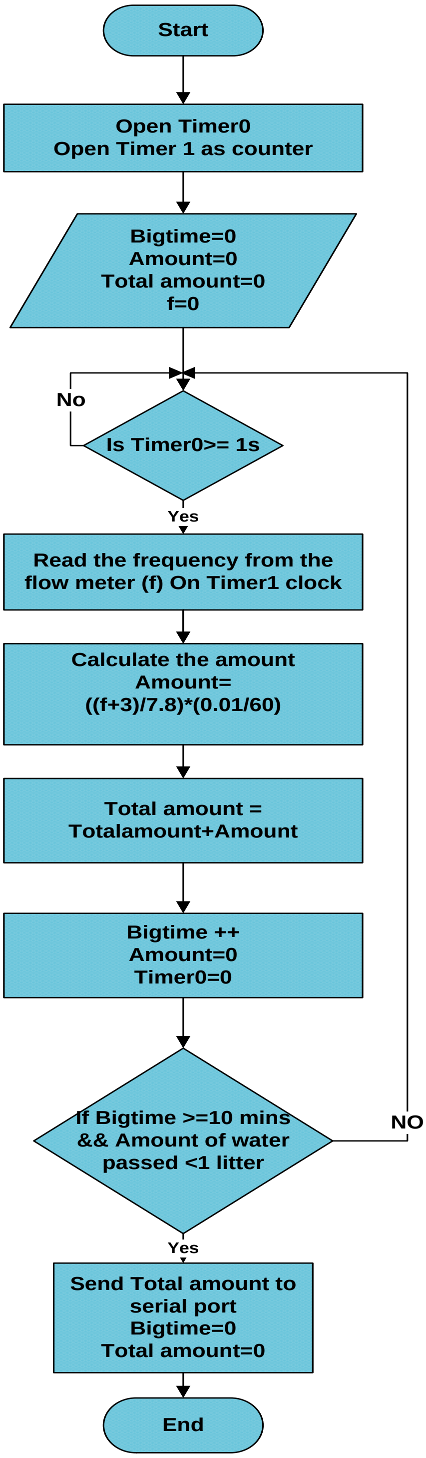
The flow meter has been connected to the microcontroller as shown in the following figure:



Figure(5.1) Microcontroller with digital flow meter

5.2.1.1 Software description:

The flow chart below shows the algorithm used to calculate the amount of water passed, using the flow meter reading:



Figure(5.2) Analog flow meter software flowchart

Timer1 has been used as a counter that reads the input frequency pulses from the flow meter every second, and Timer0 has been used as a timer to synchronize the process. After reading the frequency pulses from the flow meter, the amount of water will be calculated using the following equation:

$$Amount = ((f+3)/7.8)*(0.01/60) \; ; \quad \ldots\ldots\ldots\ldots\ldots \text{ Equation (1)}$$

This equation has been designated by the flow meter manufacturer.

The instantaneous amount will be calculated and added to the total amount every second to ensure precise reading, if the total amount of water doesn’t change for time of ten minutes, it will be sent to the serial port to be transmitted via the GPRS modem to the central office, to be added to the data base.

5.2.2 PIC and analog flow meter :

The analog flow meter is the one used currently by the customers. The idea here is developed to use the currently equipment available at the customers’ houses instead of replace them with new ones.

A simple sensing circuit has been used to interface the analog meter to the microcontroller to take its reading as a digital input to the microcontroller.

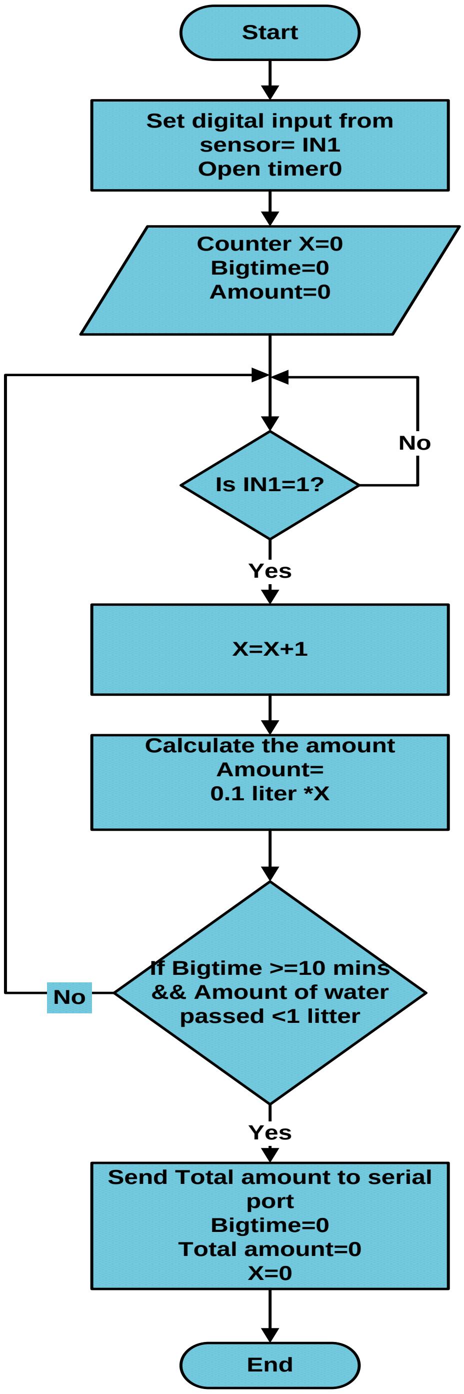
The idea and the interface circuit were explained in chapter four in figure (4.14).

The following picture shows the transceiver sensor that has been placed on the top of the analog meter and the connection with the microcontroller:

Figure(5.3) Analog flow meter and the transceiver sensor

5.2.2.1 Software description:

The flow chart below shows the algorithm used to calculate the amount of water passed, using the flow meter reading:

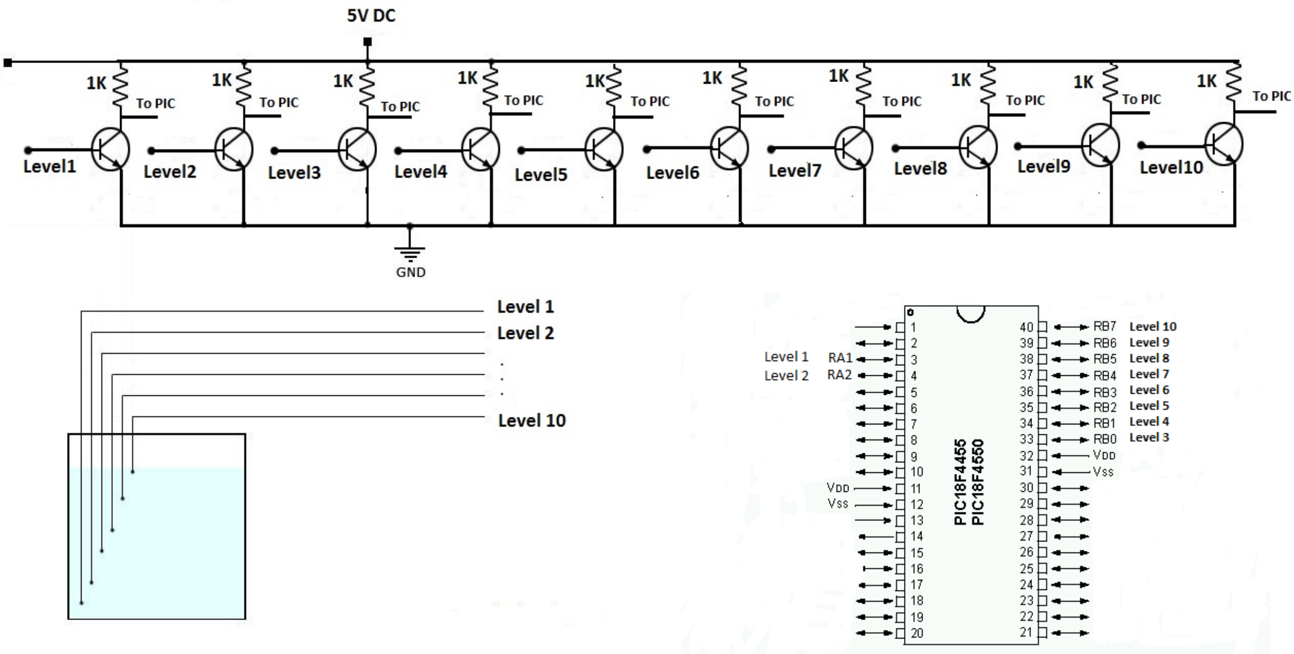


Figure(5.4) Analog flow meter software flowchart

As water follows through the meter, its pointer will rotate accordingly, a small sensor fixed above the pointer will give a high voltage every time the pointer rotate, that makes logic one on microcontroller, in its turn, the microcontroller will store these voltage pulses in a counter. Each one of the pulses correspond to 0.1 litter of water, so the microcontroller will multiply the pulses counter will 0.1 to get the amount of water has passed through the meter. If ten minutes passed and the amount of water passed less than one litter, the amount of water being calculated will be sent to the serial port to be transmitted via the GPRS modem to the central office.

5.2.3 PIC and water level indicator :

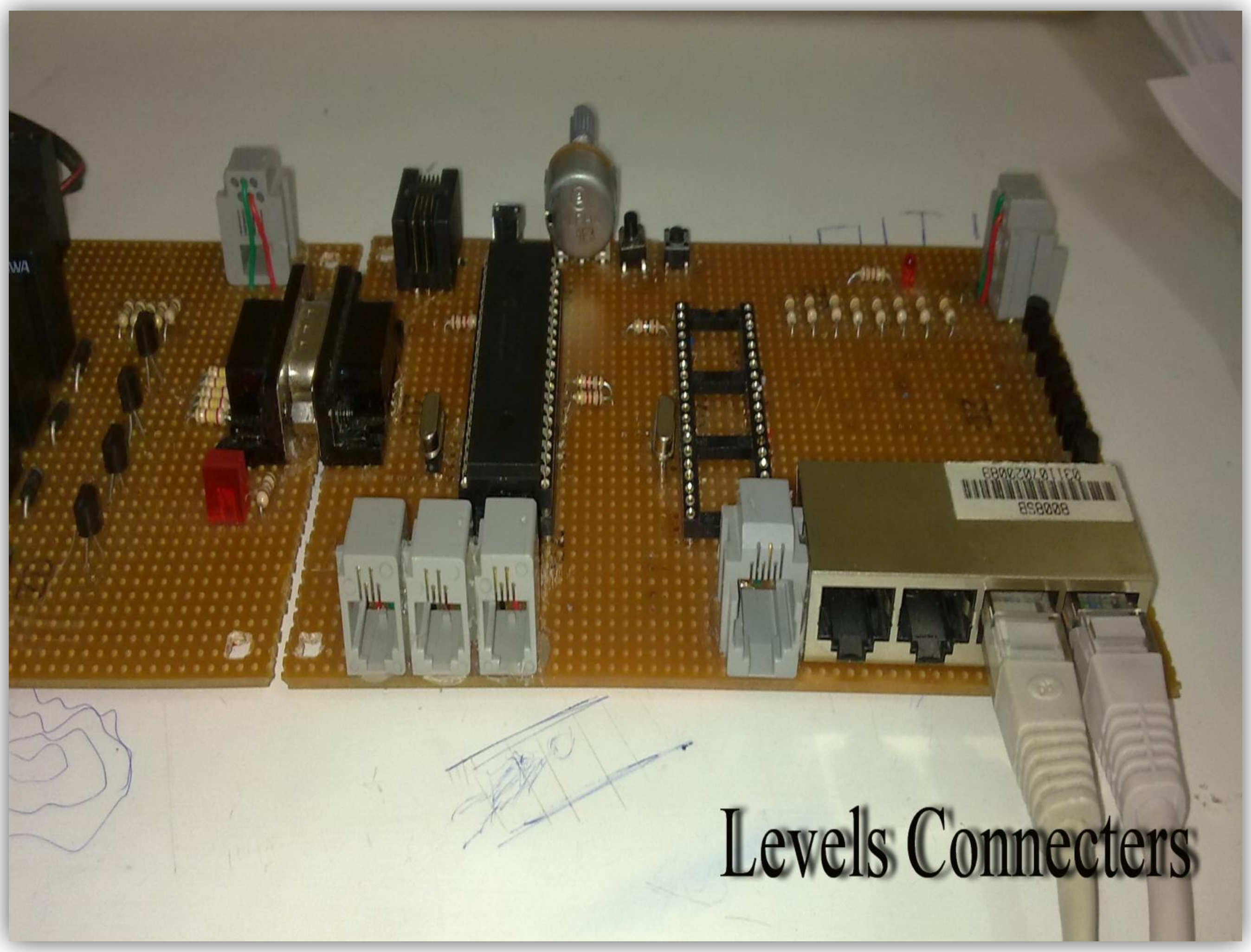
The water flow through the water network is a conductive liquid, so we can use the water level indicator shown in the circuit diagram below to calculate the amount of water inside the water tank.



Figure(5.5) Water level indicator idea

The tank has been divided into ten levels using wires. When water is rising in the tank the base of each transistor gets electrical connection to 5V DC, that signal will make logic one at the corresponding pin of the microcontroller.

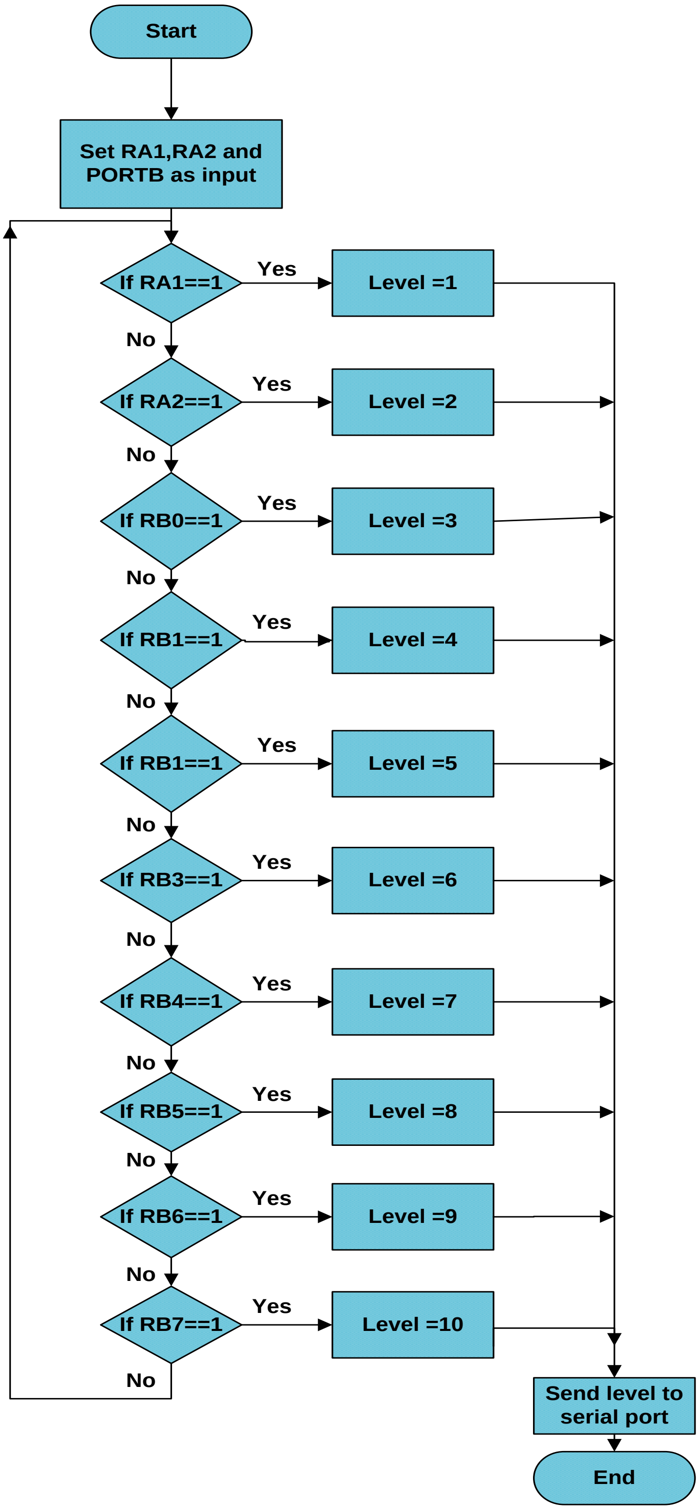
The following figure shows the circuit were built to connect the microcontroller with the water level indicator:



Figure(5.6) Microcontroller with the Water level indicator

5.2.3.1 Software description:

The flow chart below shows the algorithm used to determine the water level inside the tank:



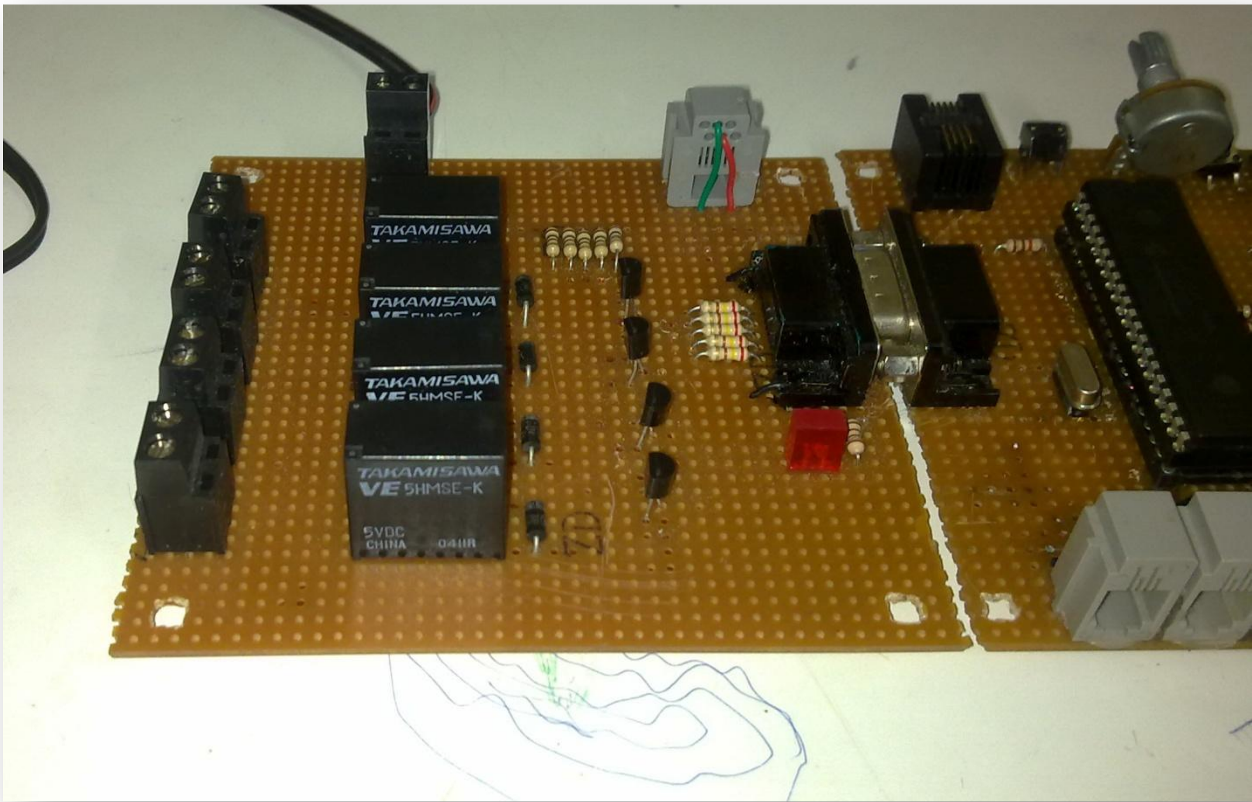
Figure(5.7) Water level indicator software flowchart

The wires indicate the levels; (level1, level2, level3,…..level10) have been connected to the microcontroller pins (RA1,RA2,RB0,RB1,RB3,…….RB7) respectively.

When the water rises inside the tank, logic one will exist on the corresponding pin on microcontroller. The program will start to check the levels from the highest one; if it is activated, the tank is full. If it's not, the program will check the lower levels.

5.2.4 PIC with AC control circuit :

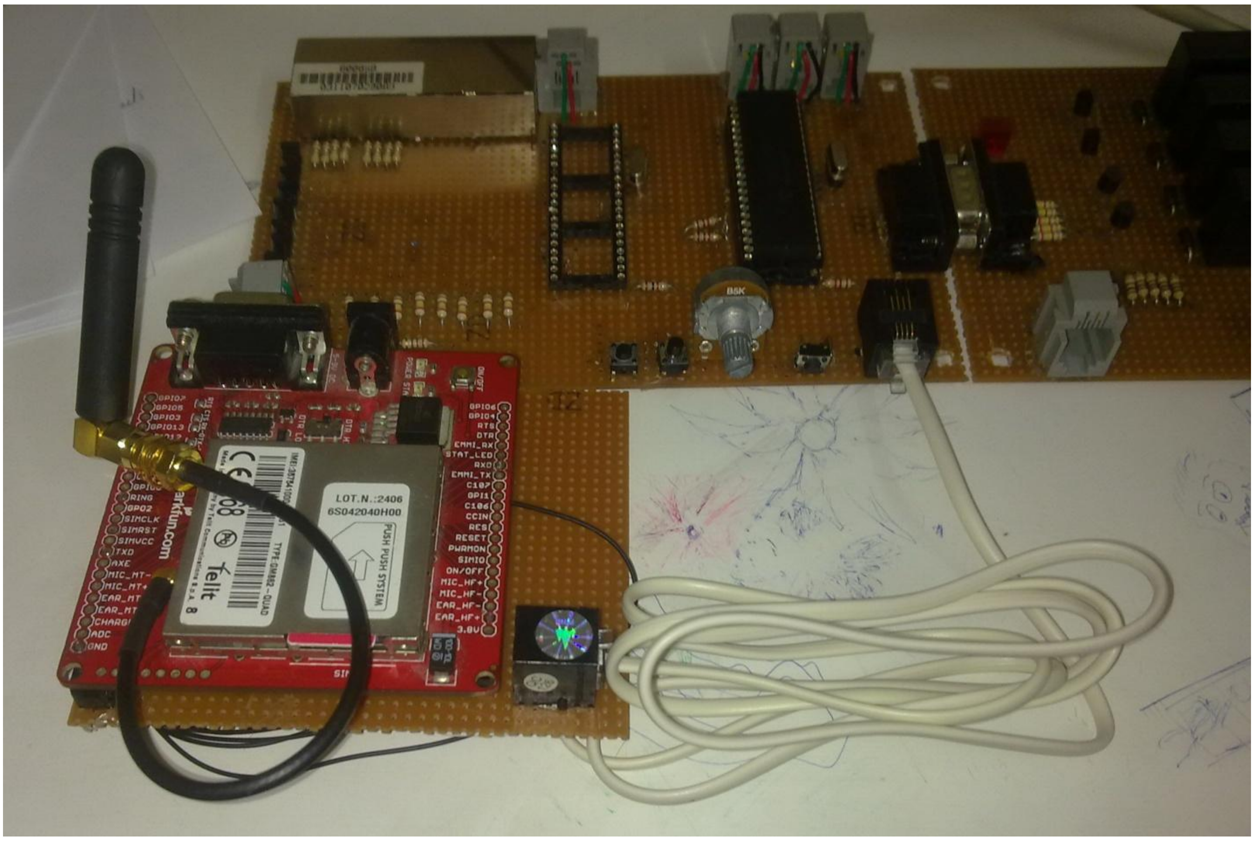
The Following circuit has been used to connect the AC components (Valves and Pump) to the PIC (Logical output). When an Order to open a valves has been initiated from the PIC (Logic one), this will close the Relay and deliver an AC220V to valve to open it.



Figure(5.8) Microcontroller with AC control circuit

5.2.5 PIC with GPRS modem:

The following picture shows the GPRS modem connected to the PIC, the modem was connected serially (Tx, Rx) , and using a buffer circuit to rise up the logic of the modem(3.3 v), to the PIC logic(5 v).



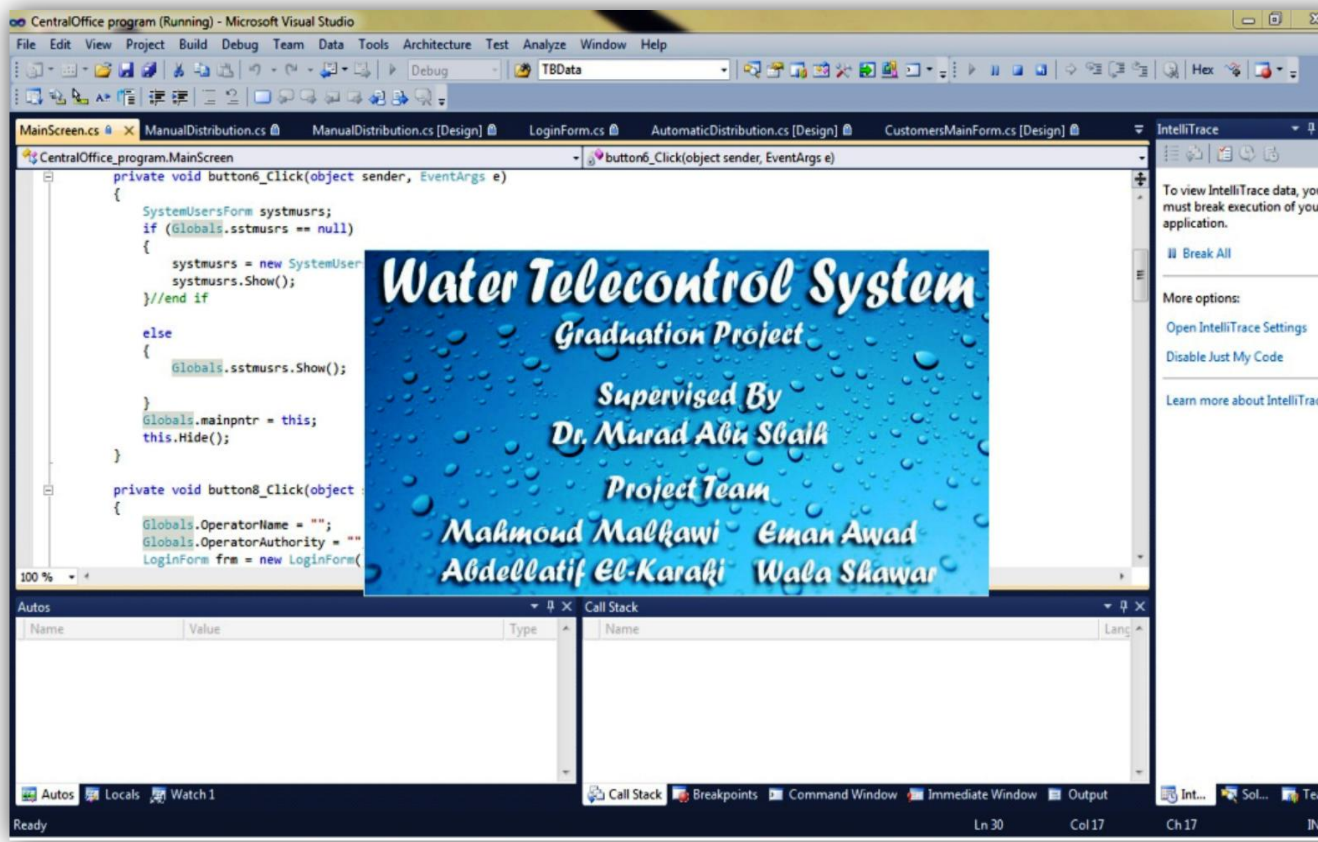
Figure(5.9) Modem Board with PIC

5.3 C# Software Program Design

The C# program at the central office acts as the brain that controls the whole water network. It receives the signals from the nodes' microcontrollers via the GPRS modem, from the received information the program get the IP of modem to determine the source node, and the ID to determine the source sensor, after determining the source of the information the program adds them to the corresponding data base in order to process them and make the necessary actions.

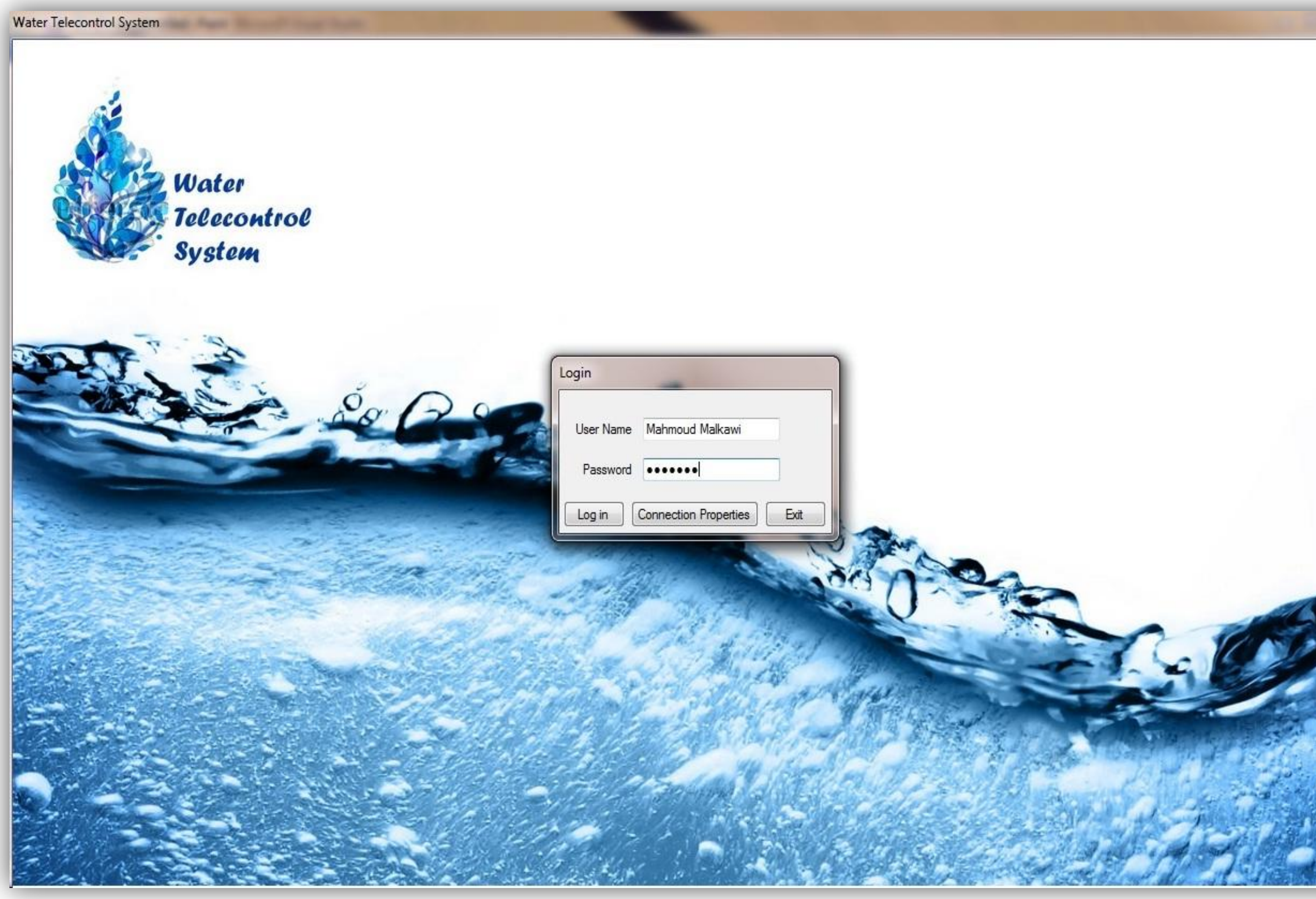
A graphical user interface has been designed to enable the systems users to deal with the program in an easy way.

After running the program the following welcome screen will appear



Figure(5.10) Welcome Screen

5.3.1 log in window

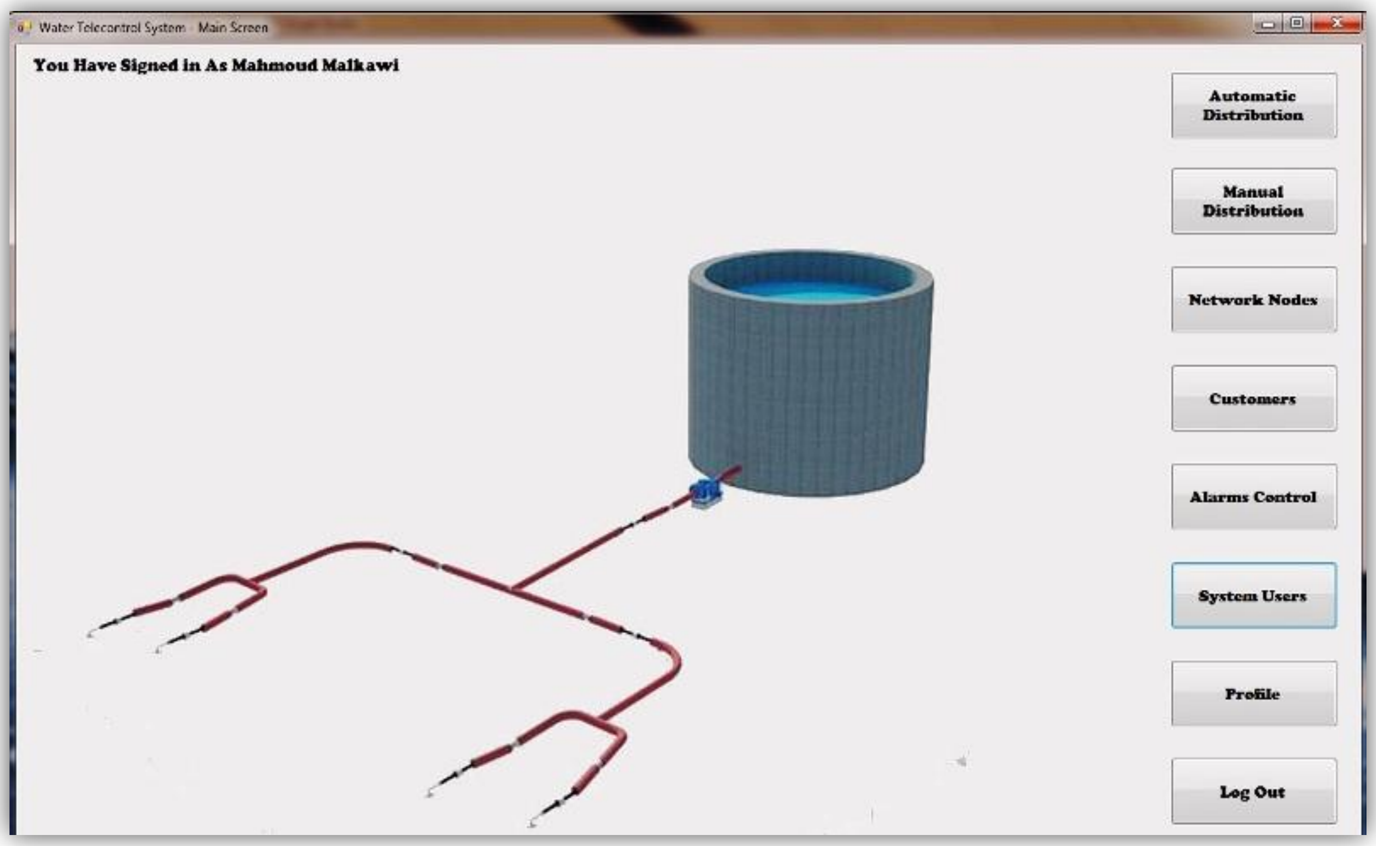


Figure(5.11) log in widow

After running the program, it will ask the user to enter a user name and a password, and here the user can log in as an administrator, a supervisor, or as a normal user, who every one of them has his own authority.

If a modem is connected to the computer, the program will connect directly to it and open a serial port and be ready to store any received data into the database. Otherwise the user will be log in offline.

5.3.2 The main window



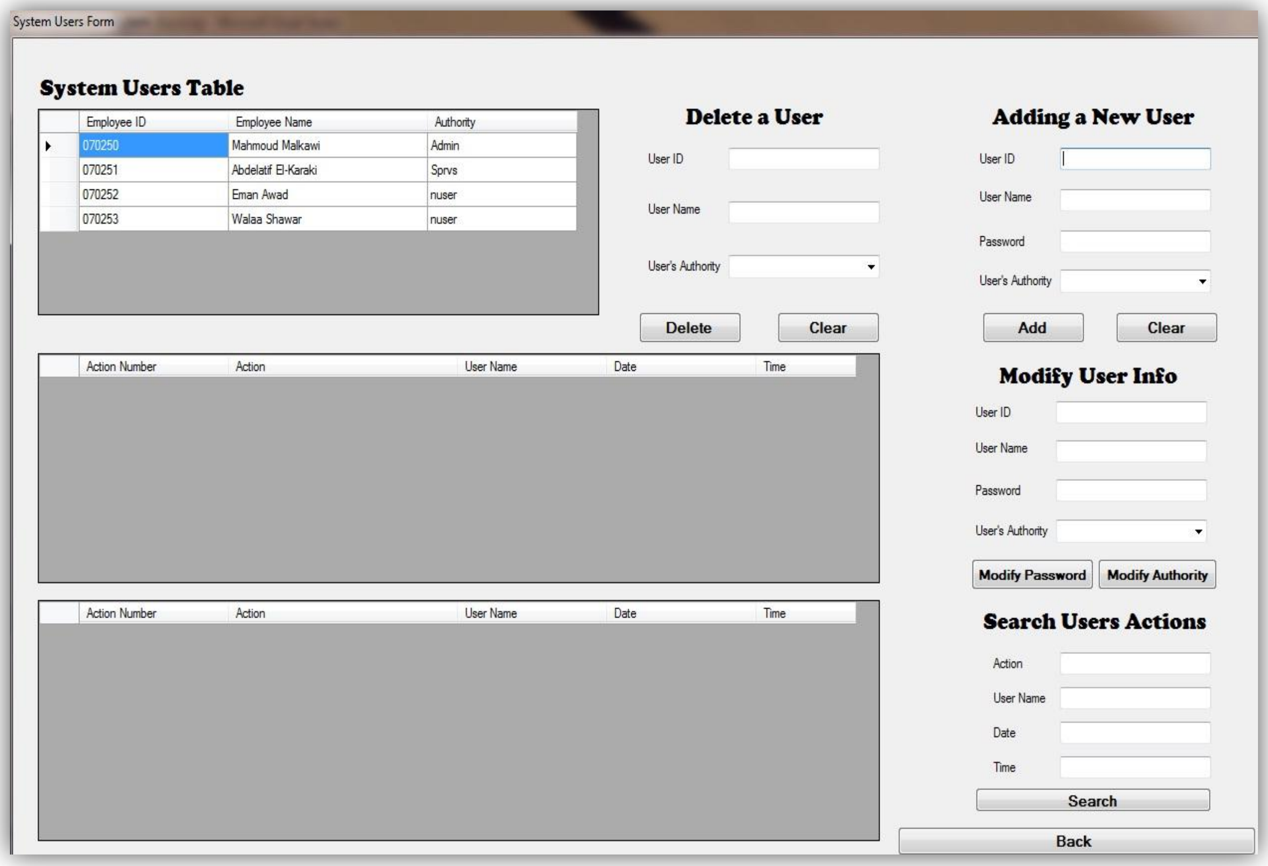
Figure(5.12) Main window

After the users enter a valid user name and password, the main window shown above will appear. The main window contains different buttons as automatic and manual distribution, network nodes, customers, alarms control, system users, profile, and log out. The appearance of these buttons depends on the user authority, which some of these buttons will not appear for the normal user. Each one of these buttons will be explained in details below.

5.3.2.1 The main window buttons

System Users button:

The following window will appear after clicking the system users button:



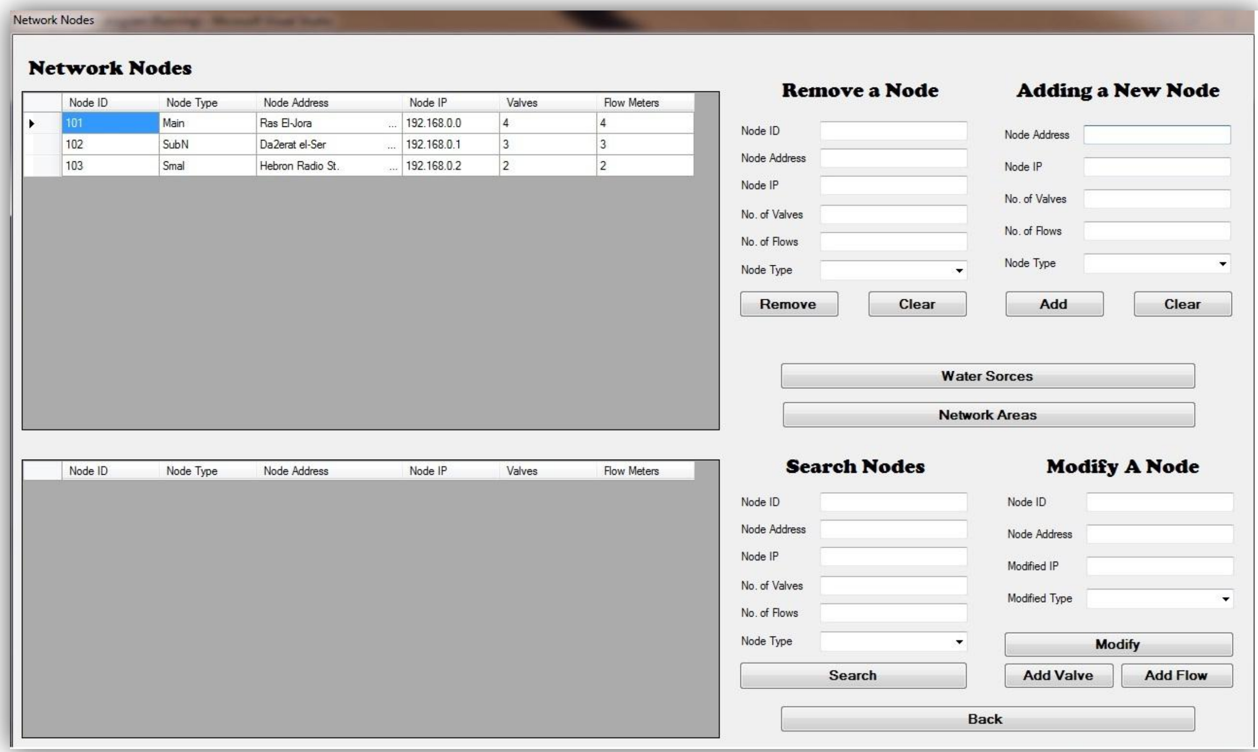
Figure(5.13) System users window

System users are the authorized people who have a user name and a password to enable them log in the program and process according to their authority. The program defines three levels of authority which are: administrator, supervisor, and normal user. The administrator can access the database of the system users and delete , add or modify their information, as shown in the figure above.

The system administrator can see the actions that have been done by other users through search users action button.

Network Nodes button:

The following window will appear after clicking network nodes button:

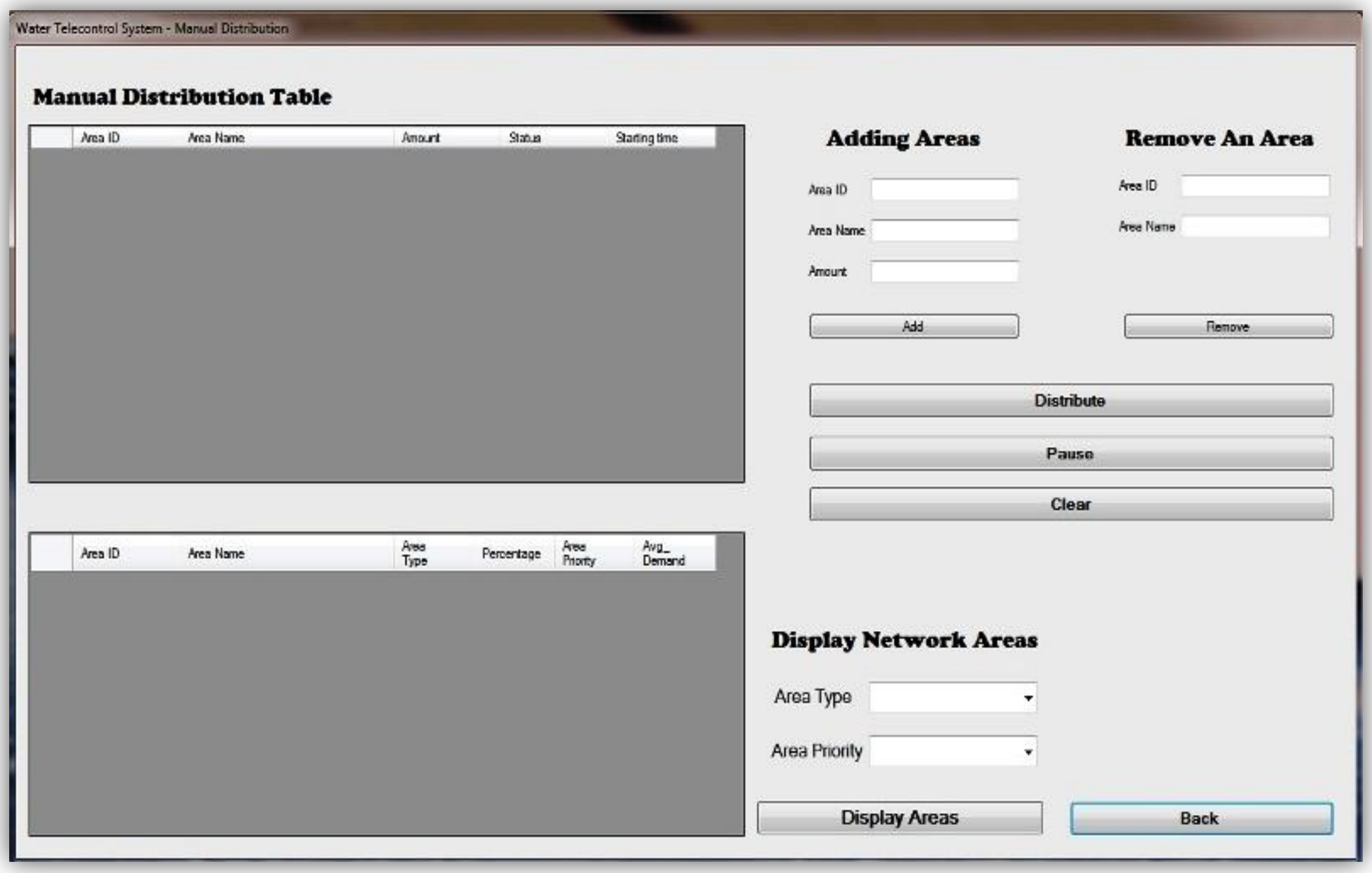


Figure(5.14) Network Nodes window

The city will be divided geographically into main nodes, each main node will be divided into sub nodes, finally each sub node will be divided into small nodes. These areas and the number of their valves and flow meters will be stored in a database. The systems' administrator can remove any node or add a new one to the network according to changes made to the water network in the field. The administrator can modify the node information according to the changes made to the node in the field, he can modify the node IP if the data SIM in the node is changed. But the node ID and address can't be modified as they are software parameters.

Manual Distribution button:

The following window will appear after clicking manual distribution button:



Figure(5.15) Manual Distribution window

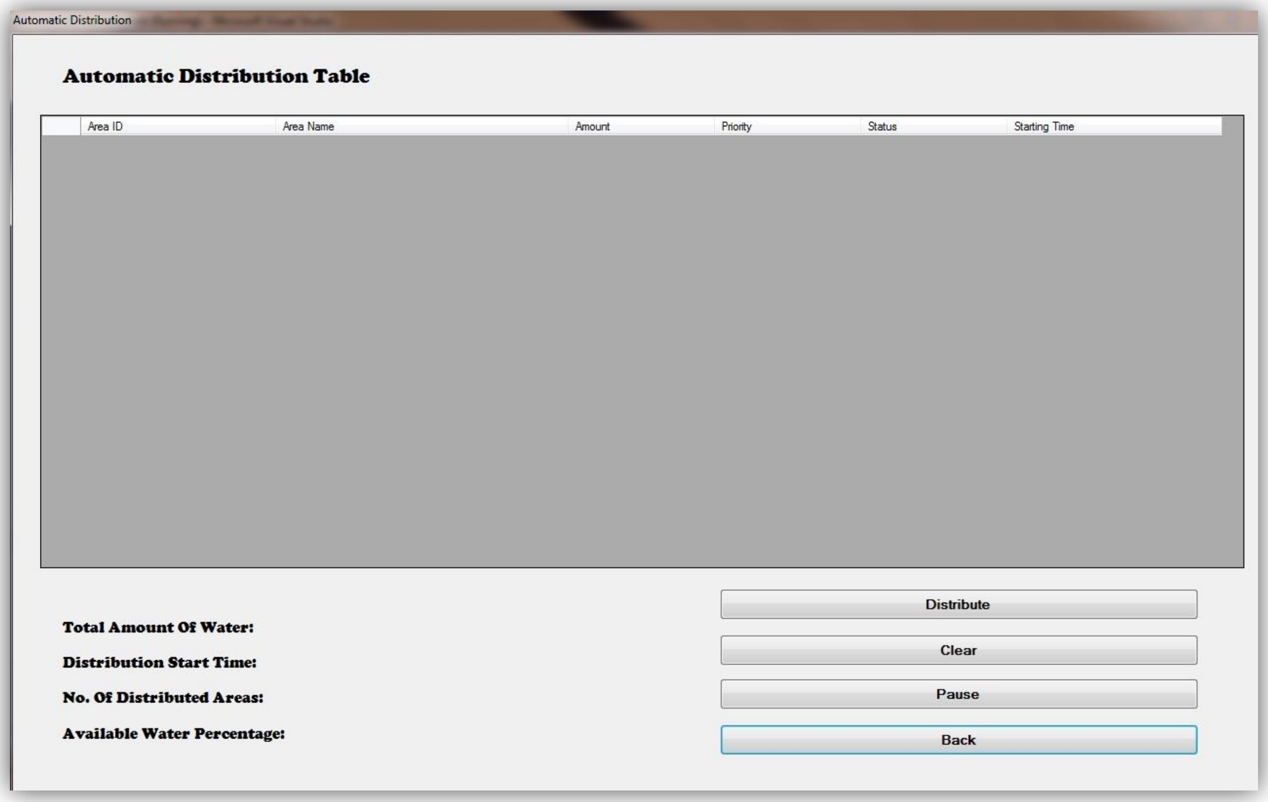
Distributing water over the areas in the network can be done automatically based on a pre-defined schedule or manually by the user.

Through the manual distribution window the user can display the network areas according to their type (main, sub, or small area), or according to their priority; there are three levels of priory: first priority is given to the area that was supplying with water but supplying has stopped suddenly before that area takes its total amount of water. The third priority is given to the area that's already has taken its total amount of water. All other areas take second priority by default.

After displaying the network areas, the user can choose any area and add it to the distribution table and define the amount of water that will be supplied to it. If the user adds an area other than the one desired to the distribution table, it can be removed using the remove button, but after clicking the distribute command, the user can't add or remove any area to, or from the distribution table.

Automatic Distribution button:

The following window will appear after clicking automatic distribution button:

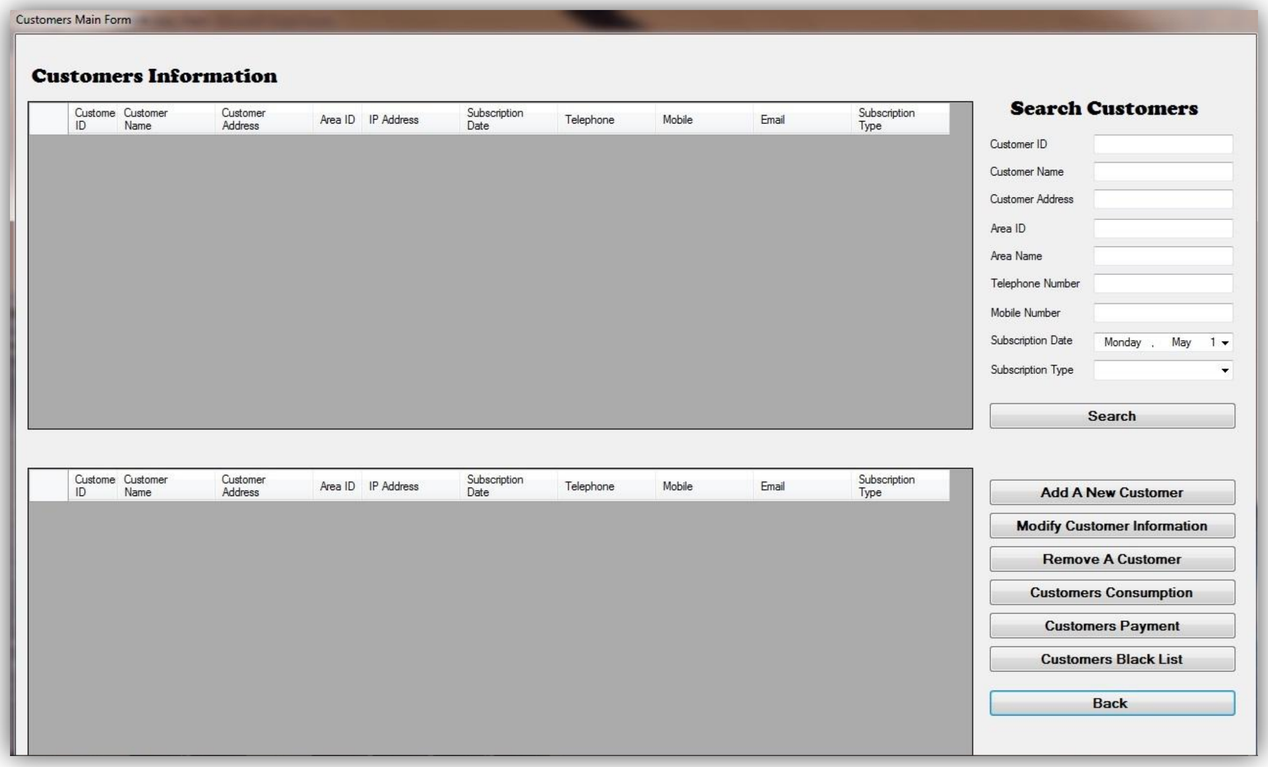


Figure(5.16) Automatic Distribution window

After the system user clicks on the distribute button, the program will get the total amount that exists in the main tank, then the program will multiply the total amount with the area percentage and distribute the amount of water to that area.

Customer button:

The following window will appear after clicking customers button:



Figure(5.17) Customers window

As shown above in the customers window, there are two tables, the first one contains personal information about the customers as: name, address, telephone number, email..etc, and the second one records of the customer water consumption.

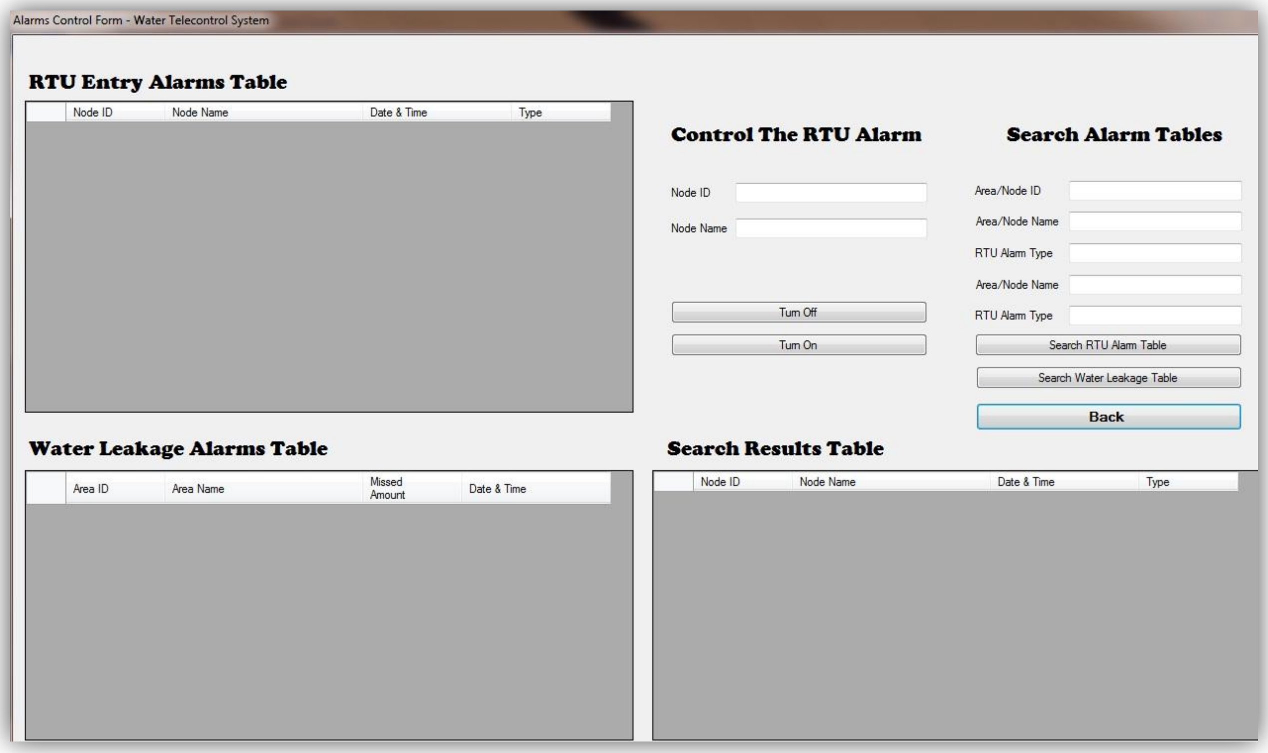
The program offers to the customers two ways for payment, pre paid or billing. Pre paid can be done using SMS sent to the central office by the user contains its ID, and the price will be deducted from the customers' mobile credit. Or the customer can pay the value of the bill he receives at his home in the end of each month.

The program give expectation for the users who may steal water according to their monthly consumption and store their names in a table called customers black list. An equation has been built to do that based on the time series analysis.

Time series: An ordered sequence of values of a variable at equally spaced time intervals. Used to obtain an understanding of the underlying forces and structure that produced the observed data, and to fit a model and proceed for forecasting, monitoring, or even feedback, and feed forward control.

Alarm control button:

As explained in chapters three and four, alarms circuits will be built on the RTUs to protect the equipment from any unauthorized access. In the case of any unauthorized access to the equipment a signal will be sent from the sensors to the central office, and the program will give a buzzer to inform the system user with the situation. Then, all the information about the alarm such as data, time, location ,,,etc will be stored in the alarm database.



Figure(5.18) Alarm Control window

6

Chapter Six

Results And Performance

6.1 Introduction

Measuring and testing the performance of the system is the final stage to complete the project. And this chapter gives a detailed description of the project testing performance.

6.2 Testing And Results

Checking of and testing the results of the project sensors, microcontrollers, modems, valves, and software are illustrated in this section individually.

6.2.1 Sensors Testing

Flow meters, level sensors are tested as following:

6.2.1.1 Flow Meter

When the flow meter is connected to the PIC at each node and let the water path through it, the flow meter gave a square wave signal that changes in frequency due to the water flow changes as in figure (6.1) on the oscilloscope. The PIC takes the flow meter readings at pin 15 and calculate the water quantity.

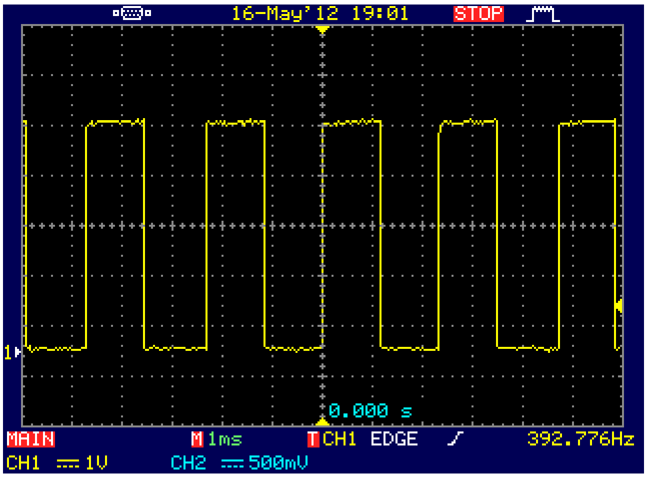


Figure (6.1) The Flow Meter Output.

The percentage error of this flow meter type is 1%, and this percentage will not even appear in this project as the average reading is taken every 10 ms and the PIC calculation takes 2 ms, so every 12 ms there is a reading from the PIC, a very accurate reading that has been checked through the usage of the function generator and the code of calculating the water quantity that we made. As a certain frequency is put by the function generator to the PIC connected with the LCD, the reading that appears is what expected to notice without any error.

Also by testing the result of letting 6 liters of water passing through the flow meter, the LCD reading was 6 as the water quantity reading.

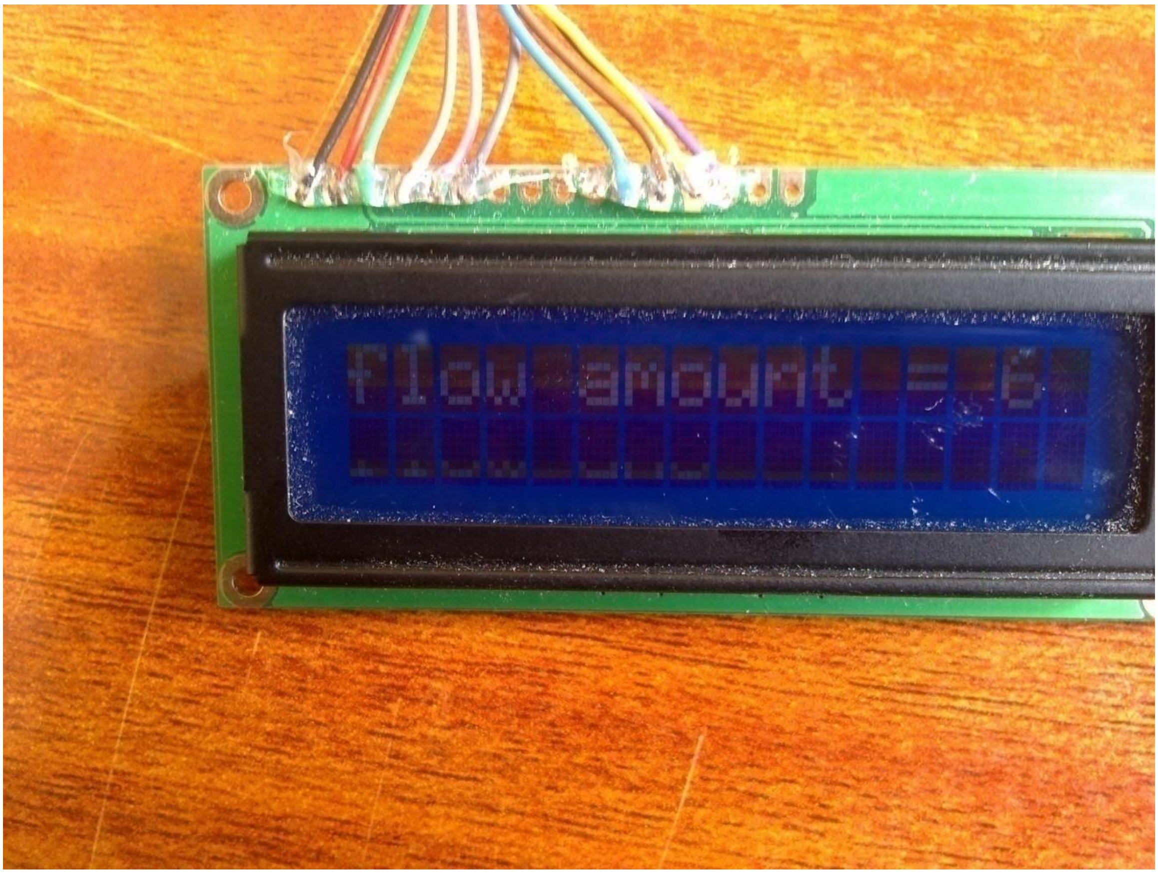


Figure (6.2) The LCD Reading.

Another choice for calculating the amount of water passing is the usage of the analog meter that is currently used in the water networks, but after adding a photo interrupter sensor with it .

After testing it, it has been made sure that this sensor can read till 1200 rpm, which the usual analog meter can't reach.

6.2.1.2 Level Sensor

In this project, an eight level sensor that spread all over the tank to calculate the amount of water in the tank was used. Each level gives a certain amount almost 16 liters of water. The levels are connected to the PIC, when the water reaches a certain level, the PIC gives the reached level with the amount of water that is described.

The level sensor was tested by filling the tank with water reached firstly to level one, the data received by the software indicates the amount of the water source with: level 1 and 16 litters. Then the tank was filled to level three and the reading was: level 3 and 48 litters.

Figure (6.3) The level Sensor in the Main Tank.

6.2.2 Valves and Pumps

The valve and pump were tested by giving a command from the central office directly to the node to open, and after approximately 10 seconds, the valve was opened. The following photo shows the connected valve and pump.

Figure (6.4) The Valve and Pump.

6.2.3 Modems Testing

The modems were tested through the C# software by sending the GPRS AT commands through the serial port.

At the sender side, a GPRS connection and attachment were made then, a session was opened to send or receive the data for both the Telit and SM5100b modems

Then, completed the opening of the session at the receiver side, and we made sure that the data transferred between them. The following commands were for the SM5100b:

-> Query and waiting for GPRS attached to network
AT+CGATT?
+CGATT : 1 (GPRS Attached)
OK
-> Setup PDP Context
AT+CGDCONT=1,"IP","SLINE"
OK
-> Activate PDP Context
AT+CGACT=1,1
OK
-> Configure the remote port, we can open a TCP connection
AT+SDATACONF=1,"TCP","10.25.100.238",1090
OK
-> Start TCP/UDP connecting
AT+SDATASTART=1,1
OK
-> Query the socket connecting status
AT+SDATASTATUS=1
+SOCKSTATUS: 1,1,0102,0,0,0 (1 means socket connected)
-> Start sending data
AT+SDATATSEND=1,10
>1234567890 <Ctrl+Z>
OK (data is sent to socket buffer and waiting to send)
-> Read the data
AT+SDATATREAD=1 (read the data from connection 1)
+SDATA: 1, 10,1234567890 (received 10 bytes, hex format)
-> Close the TCP/UDP connection
AT+SDATASTART=1,0 (close the connection 1)
OK

6.2.4 Software Testing

The ability of the software was tested to send different commands, to receive and analyze different data collected from the nodes, and to save the needed ones in the appropriate database.

The following photo shows the water source database before receiving a new data to update the database.

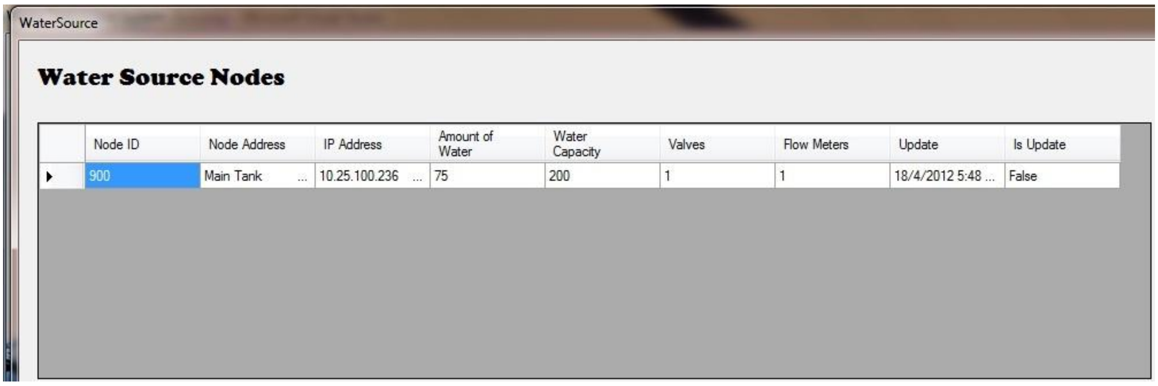


Figure (6.5) Database Before Updating.

After receiving a new data, the water source database updated as can be seen from the following photo

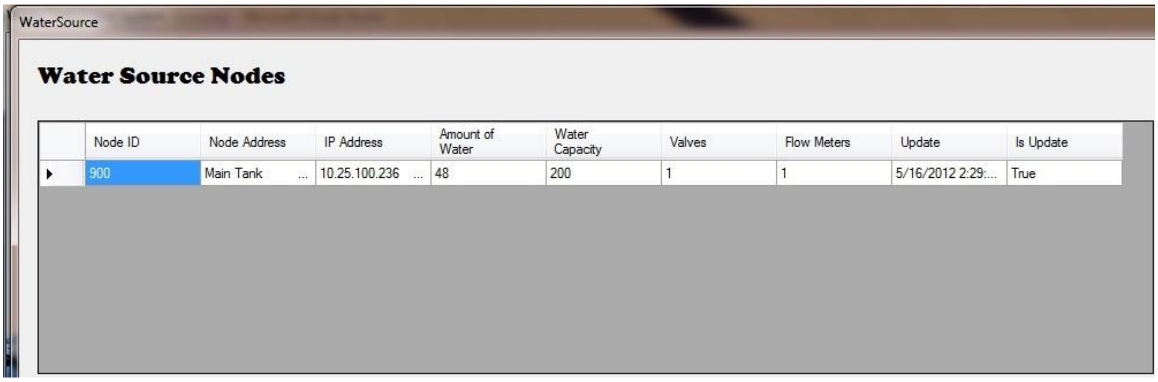


Figure (6.6) Database After Updating.

Finally, all the project components have been completely connected together to configure the whole system, and it works as all as we expected to work.

6.3 Performance Evaluation

The performance evaluation in the project can be made by measuring the delay, data rate, and the coverage of the GPRS signal.

6.3.1 Delay

The delay is the most important factor in this project, due to the fact that the data transfer need a fast transmission.

The delay is a measurement of how long a packet takes to completely arrive to the destination from the time the first bit is sent out from the source.

The transmission time between the central office and the node till the response happens takes 18 s on average, which is an acceptable delay that depends on the GSM network that the data were sent through (Al Wataniya Mobile network), also depends on the CPU of the PC that holds the software, and on the PIC and the node components response.

The GSM scenario for transferring the data gives a delay of 8 s, and the node components response takes a delay of 6 s on average. If the size of the data increases, the delay also increases. The software of the PC takes a delay of 4 seconds on average to send the commands.

6.3.2 Coverage

The GPRS signal can reach almost everywhere as it depends on the GSM network that covers the whole country. The GSM signal provides a coverage of several miles or more from a cell site. In the case of covering the areas at the network boundaries, a roaming mechanism by the network itself can be used to reach all areas that exist out of the network.

6.3.3 Data Rate

This project is a GPRS based system with 54 Kbps, which is an appropriate data rate for the packets transferring needed for the project.

7

Chapter Seven

Conclusion and Recommendations

7.1 Introduction

7.2 Problems

7.3 Acquired Learning Outcomes

7.4 Recommendations

7.1 Introduction

This chapter describes the real learning outcomes have been acquired during the work on the project, and recommendations and suggestions for future work and development.

7.2 Problems

Many problems, challenges, and issues have been raised during the work on the project. Many experiments, suggestions, ideas and researches have been carried out to deal with the different situations. Some of these problems are:

1. The Availability of the quantity and the quality of some of the Project's equipment.
2. The Israeli Restrictions on the imported equipment and the delay that occurs accordingly.
3. Many problems have been raised while establishing GPRS communication, due to the restrictions on the frequency band imposed on the local mobile networks that cause a low coverage in the GPRS service. Also some of the GSM modems were not able connect easily to the GPRS due to its low coverage.
4. The Difference between commands of each modem type.

7.3 Acquired Learning Outcomes

After accomplishing the the project tasks many talents and abilities have been achieved as:

1. We have learnt C18 programming language that is used to program the microcontroller, and how to deal with MPLAB editor.
2. We have learnt C# programming language and database building.
3. We have learnt how to interface the microcontroller with different sensors and with the GSM modem.
4. We have developed our abilities in troubleshooting and problem solving.

7.3 Recommendations

At the end, some ideas can be given to develop the system or extend its duties and functions, and some recommendations can be given to avoid the problems that may happen in the future as:

1. A small wireless network can be built to connect each group of customers that are close to each other geographically, so their consumption can be sent to the central office at one time from a central customer.
2. Instead of using a microcontroller in the main nodes, a PLC (Programmable Logic Controller) or a computer will be more efficient in dealing with flow meters and communication.
3. In similar projects SMS will be suitable in data transmission such as the water amount for the end user, there is no need to use GPRS with large packets of data.
4. If the project has a hardware part, try to get all the components of the project at the preparation stage not to postpone it to the last moment.

References

1- Shuyu, Shumei, *Water Well SCADA system based on GPRS*, 3rd International Conference on Information and Financial Engineering, Singapore, 2011.

2- A. Goel, R. Mishra, Remote *Data Acquisition Using Wireless – Scada System*, International Journal of Engineering (IJE), Volume (3) : Issue (1).

3- Synchrony Inc., *Trends in SCADA for Automated Water Systems*, Roanoke, Virginia, 2001.

4- A. Al-Omary, W. El-Medany, S. Al-Irhayim, *Design and Implementation of Secure Low Cost AMR system using GPRS Technology*, International Conference on Telecommunication Technology and Applications, Singapore, 2011.

5- Cisco System Inc., *GPRS White Paper*, 2000.

6- A. Sicher, R. Heaton, *GPRS Technology Overview*, Dell Computer Corporation, 2002.

7- Yokogawa Electric Corporation, *SCADA-RTU Communications Using GPRS,*

http://www.yokogawa.com/iab/appnotes/iab-app-gprs-en.htm

8- Usha Communications Technology, *GPRS General Packet Radio Service*, 2000.

9- A. Troelsen, *Pro C# 2008 and the .NET 3.5 platform* , Springer-Verlag New York, New York, 2008.

10- J. Sharp, *Microsoft Visual C# 2010 Step by Step*, Microsoft Press, Redmond, Washington 98052-6399, 2010.

11- .Net languages, *Introduction to the C# Language and the .NET Framework*

http://www.dotnetlanguages.net

12- A. Stellman, J. Greene, *A Brain-Friendly Guide Head First C#*, O'Reilly Media,1005 Gravenstein Highway North, Sebastopol, 2010.

13- Developer's Home, *Introduction to AT commands,*

http://www.developershome.com/sms/atCommandsIntro.asp

Flow meter

```
#include<p18f4550.h>

#include"gamelinit.h"

#include<timers.h>

#include<stdlib.h>

#include "gamelcd_v3.h"

#include "Data.h"

#include <usart.h>

#include <delays.h>

#define intr INTCONbits.TMR0IF

#define  valve  PORTAbits.RA3

#define  led  PORTAbits.RA4


//#pragma config FOSC = INTOSC_HS

//#pragma config WDT = OFF

//#pragma config LVP = OFF

/*

void Data_send1();

void Data _send2();

void Data _send3();

void Data _send4();

void Data _send5();

void Data _senda();

*/

void main (void)

{

int x, a=0;
```

```
int b=0;

float q,f;

float Q=0;

float w=0;

char  ab[5];

char sendu=1;

char text_emission[4]= " AT" ,u ;

char data1[29]= "at+cgdcont=1,IP,Sline,0.0.0.0";
char data2[12]= "at#sgact=1,1";
char data3[28]="at#sd=1,0,1080,10.25.100.238";


char send[12]= "//Tamount900";

int speed=12, i , j, l, m, c ;

//OSCCON=OSCCON | 0b01110000;



int level=0;

int lev=0;

ADCON1=0x0f;



TRISAbits.TRISA0=1;

TRISAbits.TRISA1=1;

TRISAbits.TRISA2=1;

TRISBbits.TRISB0=1;

TRISBbits.TRISB1=1;

TRISBbits.TRISB2=1;

TRISBbits.TRISB3=1;

TRISBbits.TRISB4=1;

TRISBbits.TRISB5=1;

TRISBbits.TRISB6=1;
```

```
TRISBbits.TRISB7=1;

TRISAbits.TRISA3=1; // valve output

TRISAbits.TRISA4=0;

//lcd_init();

//lcd_gotoyx(1,6);

//lcd_putrs("Hz");

//lcd_gotoyx(2,6);

//lcd_putrs("RPM");


OpenTimer1( TIMER_INT_OFF&T1_16BIT_RW&T1_SOURCE_EXT&T1_PS_1_2

&T1_OSC1EN_OFF&T1_SYNC_EXT_OFF );            //    input clck for floww frequency


OpenTimer0(TIMER_INT_OFF &

T0_16BIT &

T0_SOURCE_INT &

T0_PS_1_4 &T0_EDGE_RISE   ); // timer


//speed = 77 ;   //48000000/(64*9600) -1   9600 bauds

//OpenUSART

//(USART_TX_INT_OFF & USART_RX_INT_OFF & USART_ASYNCH_MODE &
USART_EIGHT_BIT & USART_CONT_RX & USART_BRGH_LOW,speed);


WriteTimer1(0);

 WriteTimer0(0);//10ms

 //WriteTimer0(500);


led=1;

while(1)
```

```
{
Delay10KTCYx(10);
 if(intr)
{
a=a+1;


}
if (a==15)
{
x=ReadTimer1();
f=(x*2);
q=((f)/78)/(4) ;
intr=0;
WriteTimer1(0);
WriteTimer0(0);
Q=Q+q;
w=w+q;
q=0.0;
//lcd_gotoyx(1,1);
//lcd_putrs("    ");
//lcd_gotoyx(1,1);
//lcd_puti(f);


//Bigtime = Bigtime+1;
a=0;
b=b+1;
}
```

```
if(b>65)

{

		if (w<2)

		{


//lcd_gotoyx(2,1);

//lcd_puti(w);

itoa( (int)Q , ab );

//lcd_puts(ab);

OpenUSART(USART_TX_INT_OFF & USART_RX_INT_OFF & USART_ASYNCH_MODE &
USART_EIGHT_BIT & USART_CONT_RX & USART_BRGH_LOW,speed);

//lcd_init();

//lcd_gotoyx(1,1);

for (j=0;j<4;j++)

{

while(BusyUSART()==1);

u=text_emission[j];

putcUSART(u);

//lcd_putc(u);

Delay10KTCYx(1);

}// end of at

while(BusyUSART()==1);

putcUSART(0x0D);

Delay10KTCYx(10);


for (l=0;l<29;l++)

{
```

```
while(BusyUSART()==1);
u=data1[l];
putcUSART(u);
//lcd_putc(u);
Delay10KTCYx(1);
}//  end of data1
while(BusyUSART()==1);
putcUSART(0x0D);
Delay10KTCYx(10);

for (m=0;m<12;m++)
{
while(BusyUSART()==1);
u=data2[m];
sendu=1;
if (u==0x5C )
sendu =0;

if(sendu==1)
{
putcUSART(u);
Delay10KTCYx(1);
//lcd_putc(u);
}
Delay10KTCYx(10);
}  //   end data2
```

```
while(BusyUSART()==1);
putcUSART(0x0D);
Delay10KTCYx(10);

for (c=0;c<28;c++)
{
while(BusyUSART()==1);
u=send[c];
putcUSART(u);
//lcd_putc(u);
Delay10KTCYx(1);
}  // end of data3
 for (c=0;c<2;c++)
{
while(BusyUSART()==1);
u=ab[c];
putcUSART(u);
//lcd_putc(u);
Delay10KTCYx(1);
}  // end of ab  data
while(BusyUSART()==1);
putcUSART(0x1A);
Delay10KTCYx(10);
CloseUSART();


}// end of sending data
```

```
w=0;
b=0;
//lcd_gotoyx(1,10);
//lcd_putrs("good");
}
//lcd_puti((int)Q);
}


}
```

Level sensor

```
#include<p18f4550.h>

#include <delays.h>

#include <usart.h>

#include <i2c.h>

#include"timers.h"

#include"gamelinit.h"

//#include"gamelcd_v3.h"

#include"Data.h"

#define pb0 PORTAbits.RA5

//#define pb2 PORTAbits.RA2

//#define pb3 PORTAbits.RA3

#define intr INTCONbits.TMR0IF

void Data_send1();

void Data _send2();

void Data _send3();

void Data _send4();

void Data _send5();

void Data_senda();


void main(void)

{


int level=0;

int lev=0;

ADCON1=0x0f;
```

```
TRISAbits.TRISA0=1;

TRISAbits.TRISA1=1;

TRISAbits.TRISA5=1;

TRISAbits.TRISA2=1;

TRISBbits.TRISB0=1;

TRISBbits.TRISB1=1;

TRISBbits.TRISB2=1;

TRISBbits.TRISB3=1;

TRISBbits.TRISB4=1;

TRISBbits.TRISB5=1;

TRISBbits.TRISB6=1;

TRISBbits.TRISB7=1;

PORTAbits.RA2==0;

  //  if(PORTAbits.RA2==0)

//{

     //    level=1;


//                      lcd_gotoyx(1,7);

             //          lcd_putrs("send1");

//}

 //else

//{

//lcd_init();

while(1)

{

Delay10KTCYx(1000);

//if(PORTAbits.RA5==0)
```

```
//{
    if(PORTAbits.RA0==0)
           {
        level=2;
            //          sms_send2();
            //          lcd_gotoyx(1,7);
            //          lcd_putrs("send1");
}
     else
{
    if(PORTBbits.RB7==0)
        {
        level=3;
            //          sms_send3();
            //          lcd_gotoyx(1,7);
            //          lcd_putrs("send1");
}
  else
{
    if(PORTBbits.RB5==0)
  {
        level=4;
//                      sms_send4();
            //          lcd_gotoyx(1,7);
            //          lcd_putrs("send1");
}
else
```

```
{
    if(PORTBbits.RB6==0)
        {
        level=5;
            //      sms_send5();
            //      lcd_gotoyx(1,7);
            //      lcd_putrs("send1");
}
     else
{
    if(PORTBbits.RB4==0)
         level=6;
     else
{
    if(PORTBbits.RB3==0)


         level=7;
    else
{
    if(PORTBbits.RB2==0)
        level=8;


}
}
}
}
}
```

```
//}

}

//lcd_gotoyx(1,7);
//lcd_puti(level);
if(level!=lev)
{
if(level==8)
Data_send1();
if(level==7)
Data_send2();
if(level==6)
Data_send3();
if(level==5)
Data_send4();
if(level==4)
Data_send5();
if(level==3)
Data_senda();
//level=lev;
lev=level;
}
```

```
}// while 1
}
```

Gprs part

```
#include <p18f4550.h>
#include "gamelcd_v3.h"
#include <usart.h>
#include "gamelinit.h"

void main(void)  // Test USART
{
int i,N; // 16 bits
char text_emission[20]= "HELLO WORLD\n\r";
char reception,u;
unsigned int speed;

// init hardware
TRISA = 0xBF;  // LED output
TRISB = 0xFF;  // PB inputs
TRISE = TRISE & 0xFC;   // Jack is an input

//lcd_init();
//lcd_gotoyx(1,1);

speed = 77 ;   //48000000/(64*9600) -1   9600 bauds
OpenUSART
```

```
(USART_TX_INT_OFF & USART_RX_INT_OFF & USART_ASYNCH_MODE & USART_EIGHT_BIT
& USART_CONT_RX & USART_BRGH_LOW,speed);



// 8bits

// test emission

// sends a message twice


for(i=0;text_emission[i] != 0;i++) // character by character

{

while(BusyUSART()==1);  // wait for buffer empty

u = text_emission[i];

putcUSART(u);

}


putsUSART(text_emission); // sends the message as a string


// test reception

// waits for 10 caracteres  and displays them on the LCD

for(i=0;i<10;i++)

{

while(!DataRdyUSART());  // wait for buffer full

reception = getcUSART();  // read the character in the buffer

//lcd_putc(reception);  // displays it on the LCD display

}


CloseUSART();

while(1);           // simulates the end of the program
```

```
}
//..........................................................................................
//..........................................................................................
/*void delay(int c)
{
while(c)
OpenTimer0(TIMER_INT_OFF&T0_16BIT&T0_SOURCE_INT&T0_EDGE_RISE&T0_PS_1_16);
 WriteTimer0 (50000);
}*/
//..........................................................................................
//..........................................................................................
void send_at(const char *s)
{ while(*s){ WriteUSART(*s++);}
  usart_write(0x0D);
}
//..........................................................................................
//..........................................................................................
void send_text(const char *s)
{ while(*s){WriteUSART(*s++);}}
//..........................................................................................
//..........................................................................................
void send_char(char s)
{ WriteUSART(s);}
//..........................................................................................
//..........................................................................................
void rcv_text(const char r)
{ while(r){WriteUSART(r++);}}
```

```
//......................................................................................................

//......................................................................................................


void setupAT()
 {
 char i=0;
 PORTD.F1=1;
 DELAY_MS(2300);
 PORTD.F1=0;
 DELAY_MS(3000);


   for(i=0;i<20;i++)
 {
 send_at("AT");
 delay_ms(1000);
 if (response_rcvd==1&&response==GSM_OK) break;
 }
// send_at("ATE0");
 delay_ms(2500);
  send_at("at+cgdcont=1,IP,Sline,0.0.0.0");//
  delay_ms(2500);
 response_rcvd=0;
//+cgdcont=1,"ip","sline","0.0.0.0",0,0
 delay_ms(1000);
}
//......................................................................................................
```

```
//............................................................................................................

void main (void)
{
int x;
int Bigtime=0;
float q,f;
float Q=0;
float w=0;
ADCON1=0x0f;

OpenTimer1( TIMER_INT_OFF&T1_16BIT_RW&T1_SOURCE_EXT&T1_PS_1_2
&T1_OSC1EN_OFF&T1_SYNC_EXT_OFF );

OpenTimer0(TIMER_INT_OFF &
T0_16BIT &
T0_SOURCE_INT &
T0_PS_1_1 &T0_EDGE_RISE   );

 lcd_init();

WriteTimer1(0);
```

```
 WriteTimer0(5536);
 lcd_init();



while(1)
{

if (intr)
{
x=ReadTimer1();
f=x/.01;
q=((f+3)/7.8)*(.01/60) ;
intr=0;
WriteTimer1(0);
Q=Q+q;
q=0.0;
w=w+q;
if(Bigtime>70000)
{
        if (1<w<3)
        {
                //serial communication for sending
                //zeros
lcd_puti((int)Q);
        }
w=0;
```

```
Bigtime=0;

}
Bigtime = Bigtime+1;

}

}

}
//::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
```

Pic receive and controlling

```
#include<delays.h>
#include<p18f4550.h>
#include<adc.h>
//#include <i2c.h>
#include"gamelcd_v3.h"
#include"gamelinit.h"
//#include"Data.h"
#include<usart.h>
/*
#pragma config FOSC = INTOSC_HS
```

```
#pragma config WDT = OFF

#pragma config LVP = OFF


void Data_send1();

void Data _send2();

void Data _send3();

void Data _send4();

void Data _send5();

*/

void main(void)

{


char reception ;

unsigned int spbrg;

int i,a=0, label=1;

int j=0, m=0, label2=0;

            int b=0  ;

char text_emission[5]= " AT\n" ,  u ;

char read[12]= "  at#sgact=1,1" ;
char MyByte[8];

ADCON1=0x0F;

TRISAbits.TRISA5=1;

TRISAbits.TRISA2=0;

TRISAbits.TRISA0=0;

TRISAbits.TRISA1=0;

TRISAbits.TRISA3=0;

TRISAbits.TRISA3=0;

//OSCCON=126;
```

```
spbrg =77 ;

lcd_init();                     //initialize the LCD module
lcd_gotoyx(1,1);
//PORTA=0;
PORTAbits.RA2=0;
PORTAbits.RA0=0;
PORTAbits.RA1=0;
PORTAbits.RA3=0;
PORTAbits.RA5=1;



        ADCON1=0x0f;
        lcd_init();
lcd_gotoyx(1,1);
        Delay10KTCYx(100);
while(1)
{
OpenUSART(USART_TX_INT_OFF & USART_RX_INT_OFF & USART_ASYNCH_MODE &
USART_EIGHT_BIT & USART_CONT_RX & USART_BRGH_LOW,spbrg);
        Delay10KTCYx(100);
while(BusyUSART()==1);
u = text_emission[i];
putcUSART(0x00);
while(BusyUSART()==1);
u = text_emission[i];
putcUSART('A');
```

```
while(!DataRdyUSART());
reception = ReadUSART();

while(BusyUSART()==1);
u = text_emission[i];
putcUSART('T');

while(!DataRdyUSART());
reception = ReadUSART();

while(BusyUSART()==1);
putcUSART(0x0D);

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
```

```
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

putcUSART(0x00);

for (i=0;i<12;i++)
{

while(BusyUSART()==1);
u = read[i];
putcUSART(u);

while(!DataRdyUSART());
reception = ReadUSART();

}

while(BusyUSART()==1);
putcUSART(0x0D);

while(!DataRdyUSART());
reception = ReadUSART();
```

```
while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

while(!DataRdyUSART());
reception = ReadUSART();

putcUSART(0x00);

label=1;
//CloseUSART();
//while(1)
//{

//OpenUSART(USART_TX_INT_OFF & USART_RX_INT_OFF & USART_ASYNCH_MODE &
USART_EIGHT_BIT & USART_CONT_RX & USART_BRGH_LOW,spbrg);
```

```
		Delay10KTCYx(100);

	for (i=0;i<12;i++)
	{
	while(BusyUSART()==1);
	u = read[i];
	putcUSART(u);

	while(!DataRdyUSART());
	reception = ReadUSART();

	}

	lcd_gotoyx(1,10);
	lcd_puti(m);//mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm
mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm
	while(BusyUSART()==1);
	putcUSART(0x0D);
	lcd_putrs("3adda");



	while(!DataRdyUSART());
	reception = ReadUSART();

	while(!DataRdyUSART());
	reception = ReadUSART();

	while(!DataRdyUSART());
```

```
reception = ReadUSART();

for(i=0;i<62;i++)

{

while(!DataRdyUSART());
reception = ReadUSART();

if(reception!='E')
label=1;

if(reception=='E')  // if no sms (error)
{
while(!DataRdyUSART());
reception = ReadUSART();
//i++;
if(reception=='R')
{
for(j=0;j<9;j++)
{
while(!DataRdyUSART());
reception = ReadUSART();
}
lcd_putrs("error");
label=0;
break;
```

```
}
}

if(label==1)
{
if(i>57)
{
lcd_putc(reception);
label2=1;

if (reception=='v')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='l')
{
PORTAbits.RA2=1;
PORTAbits.RA3=1;
}
}
if (reception=='c')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
```

```
i++;
if (reception=='o')
{
PORTAbits.RA2=0;
PORTAbits.RA3=0;
}
}//if close
if (reception=='n')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='g')
{
PORTAbits.RA0=1;
PORTAbits.RA3=1;
//break;
}
}//if open port a0
if (reception=='m')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='d')
```

```
{
PORTAbits.RA0=0;
PORTAbits.RA3=0;
//break;
}
}//if close a0
if (reception=='f')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='g')
{
PORTAbits.RA1=1;
PORTAbits.RA3=1;
}
}//if open port a1
if (reception=='h')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='i')
{
PORTAbits.RA1=0;
```

```
PORTAbits.RA3=0;
}
}//if close a1
/*
if (reception=='j')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='k')
{
PORTAbits.RA3=1;
}
}//if open port a3
if (reception=='l')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='m')
{
PORTAbits.RA3=0;
}
}//if close a3
```

```
if (reception=='o')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='p')
{
PORTAbits.RA5=0;
}
}//if open port a3
if (reception=='q')
{
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putc(reception);
i++;
if (reception=='r')
{
PORTAbits.RA5=1;
}
}//if close a3
*/
/*******************************************/
}// if valve open
} // label
```

```
}

lcd_gotoyx(2,9);

while(BusyUSART()==1);
putcUSART(0x00);
//lcd_putrs("4OK");
Delay10KTCYx(100);

if(label==1)
{
while(!DataRdyUSART());
reception = ReadUSART();
//lcd_putrs("5");
while(!DataRdyUSART());
reception = ReadUSART();
lcd_putrs("6");

} // label

Delay10KTCYx(65000);
Delay10KTCYx(65000);
```

```
if(label2==1)

{

label2=0;

for (i=0;i<9;i++)

`{

while(BusyUSART()==1);

u = deleteSMS[i];

putcUSART(u);


}

while(BusyUSART()==1);

putcUSART(0x0D);


lcd_gotoyx(2,5);

lcd_putrs("3bed");

label=0;


} // if label


Delay10KTCYx(65000);

Delay10KTCYx(65000);

Delay10KTCYx(65000);


m++;

lcd_gotoyx(1,7);

lcd_puti(m);//mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm
mmmmmmmmmmmmmmmmmmmmm
```

```
}

}
```

I2c slave

```
#include <p18f4550.h>

#include <i2c.h>

#include "gamelinit.h"

#include "gamelcd_v3.h"

#define L1 PORTAbits.RA1

#define L2 PORTAbits.RA2

void main (void)

{                       char MyByte;

            ADCON1=0x0f;

            TRISAbits.TRISA1=0;

            TRISAbits.TRISA2=0;



            OpenI2C(SLAVE_7,SLEW_ON);

            SSPADD = 0x32;

            while (1)

            {


            while (!DataRdyI2C());

            MyByte=ReadI2C();


            if (MyByte =='k')
```

```
            {

            L1=1;
            L2=0;

Delay10KTCYx();
L1=!L1;
L2=!L2;
Delay10KTCYx();
            }

            }
}
```

I2c master

```
#include <p18f4550.h>
#include <i2c.h>
#include "gamelinit.h"
#include "gamelcd_v3.h"

void main (void)
{
            int i=0;
            int j=0;
            char G[8]="ABCDEFGH";
```

```
        char MyByte[8];
        ADCON1=0x0f;
        OpenI2C(MASTER,SLEW_ON);
        SSPADD = 118;
        IdleI2C();
        StartI2C();
        while ( SSPCON2bits.SEN );
        IdleI2C();
        putcI2C(0x28);
        IdleI2C();
        lcd_gotoyx(1,1);
        while (i<8)
{
        putcI2C(0x28);
        IdleI2C();
        putcI2C(G[i]);
        IdleI2C();
        lcd_putc(G[i]);
        i++;

}
        StopI2C();
        IdleI2C();
        CloseI2C();
}
```