CHAPTER ONE

INTRODUCTION

1.1 Introduction

Sudan economy is largely depends on agriculture resources which it is depend on water rainfall through the rain season. If we exclusive sector of irrigation agriculture, the major is depend on seasonal rainfall in most country side. So to measure rainfall using reliable and accurate gauge is of high importance.

The rain gauge (also known as udometer or pluviometer or cup) is definitely an instrument that is used by meteorologists and hydrologists to gather and measure the quantity of rainfall or precipitation over a set period of time. Rain gauges are considered the most traditional method for measuring rainfall.

Historically the first rain records were kept by the ancient Greeks, about 500 B.C, after 100 year later people in India used bowls to record the rainfall. Today rain record is a part of weather forecasting which weather warring are important forecast because they are used to protect the life and properties , also rainfall play an major role in our life which include drinking, agriculture, floods energy production travelling and all life activities.

The basic idea of most rain gauges is to collect rainwater into cylindrical vessel of a fixed diameter. Rainfall measurement is usually provided in units of water depth (inches or millimeters). The volume of collected water is divided by the area of cylinder opening and converted into a depth or rain. The new designs of rain gauge give more accurate reading and have many advantages such as that observer can record rainfall remotely. Because measure rainfall here is according tip drop not on traditionally looking at the water level of course there is possible of error .There are different types of rain gauges that can be classified into two main categories: recording gauges, and non-recording gauges.

Digital tipping bucket rain gauge consists of funnel that collect and channel the precipitation into small seesaw (seesaw is mechanically lever and fulcrum) like a container. After a pre-set amount of precipitation falls, the lever tips, dumping the collect water and sending signal by red switch (or sensor), the sensor send electrical signal to microprocessor(microcontroller) which count and give quantity of rainfall ,the result can be display on pc or seven segment screen.

1.2 Problem Statement

In conventional rain gauge, the meteorologist (observer) needs to wait until rain stop then take readings, but in digital tipping bucket rain gauge there is no need for waiting for rain stop. Vaporization happened to rain precipitation result in inaccurate reading rainfall is decided by total amount of rain that has fallen in a set period of time. Here rain gauge connected to PC through serial communication and the software converts the number of pulses into value of rainfall.

The advantage of recording rainfall records in PC is that it can be as soon as possible shared through internet, this help to rapidly transfer of rainfall rates in specific area and make early alarm about rain flooded. A digital rain gauge offers an element that

a traditional model cannot timeliness. Every drop of rain that falls into a collection tube is instantly measured in a digital model.

1.3 The Objective of the Thesis

(i) Design and implement a digital tipping bucket rain gauge with display and save rainfall in a PC. To build using available material: two balanced bucket, funnel, gauge body, electronic components. Used material is metal and plastic.

(ii) To study and understand microcontroller and its functions and other electronic components used in designing the gauge.

(iii) To test measure rainfall with precise reading. Accuracy consideration depends largely on setting of gauge and design parameters.

1.4 The Scope of the Thesis

The works needed to execute the project are hardware and software.

Hardware is through design of tipping rain gauge mechanical parts, which include tipping bucket, funnel, and unit body with standard measurement stand as possible. In addition to electronic circuits designs using the microcontroller, sensors, and serial communication.

While software include microcontroller programming though programming software, the instruction assign input from sensor (which convert mechanical motion to electrical signal) as an input and each input triggered as a value in mm in output PC display.

1.5 Out Line of Thesis

This thesis consists of five chapters. The first chapter discusses the objective and scope of the study as a summary of the works. While chapter two concentrates on theory and literature reviews of rain precipitation, types of rain gauges, methods of rain recording and microcontroller which it is used in modern rain gauges. While chapter three deals with hardware and software implementation of this project. The result and analysis are presented at chapter four. The last chapter includes conclusion and future recommendations.

Many models have computers in the weather station that allow users to see the amount of rain that has fallen during a particular period of time, the computer's memory allows users to see the total accumulation of a specific day, a week, a month and even a year in many cases.

As rainfall measurement need more accuracy, so design of tipping bucket and all parts of gauge must be compatible with each other as much as possible according to available materials for design.

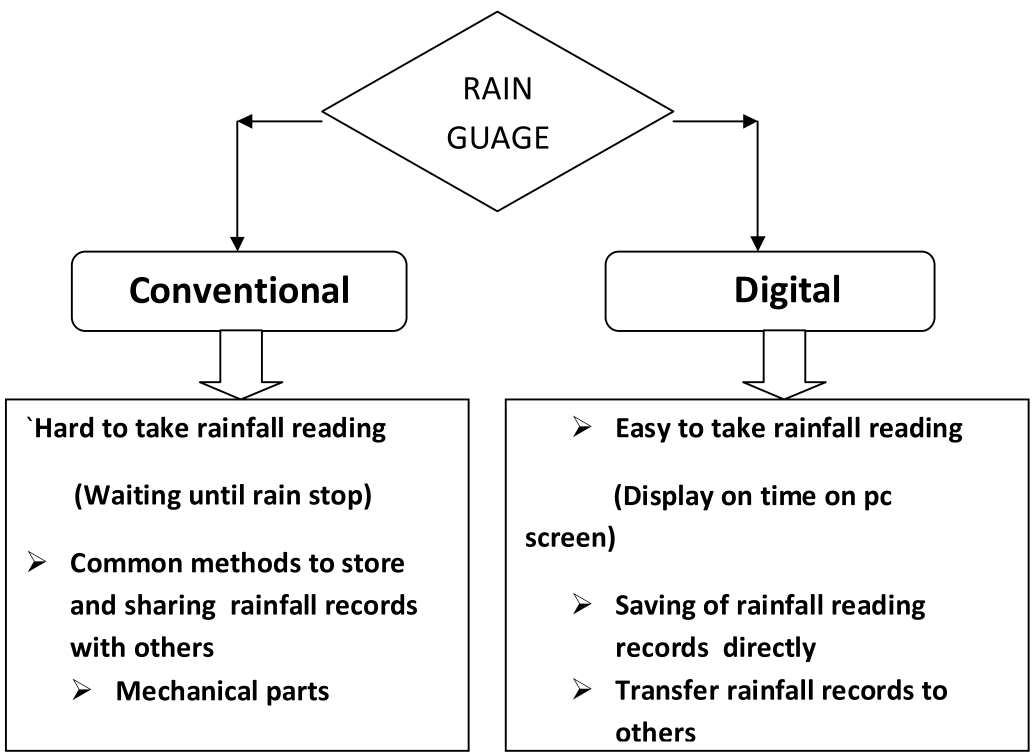


Figure 1.1 Problem Statement Representation

CHAPTER TWO

Theory and Literature Review

2.1 Introduction

This chapter includes the study of different types of rain gauges and observation instruments, rain gauge classification, condition of exposure, limitation of accuracy, microcontroller architecture, timer, voltage regulator, flip-flop, and methods of connections. Precipitation is defined as liquid (water drops) or solid (ice flakes) condensation of water vapor falling from clouds or deposited from air into ground. Precipitation is measured as the amount of water reaches horizontal ground or horizontal projection plane of earth's surface, and expressed as vertical depth of water or the water equivalent of solid precipitation the units of precipitation is millimeter.

Instrument for measuring precipitation include rain gauge and snow gauge, and various types manufacturing according to the purpose at hand. Here concentration will be on rain gauges.

2.2 Rain Gauges classification

Rain gauges are classified into **recording** and **non-recording types.**

Non-recording Includes cylindrical and ordinary rain gauges and measurement of precipitation with these types is performed manually by the observer. Some recording types such as siphon rain gauges have a built-in recorder, and the observer must physically visit the observation site to obtain data.

Recording gauges this rain gauges are used to obtain a continuous record of daily or weekly rainfall as rain gauges measure weight or volume of precipitation collected in vessel collected with affixed orifice diameter.

2.2.1 Non-recording gauges

These instruments work according to simple principle of measurement, and also have straightforward structure. They offer advantage of having a low rate of problem occurrence. This includes:

(i) Cylindrical Rain Gauges shown in Figure 2.1 type of rain gauge which can be used to measure snow, it is alternatively known as cylindrical snow/rain gauge. It consists of a cylindrical vessel with a uniform diameter from top to bottom and an orifice at the top. It does not have a funnel. Rainwater enters through the orifice and accumulates in the cylindrical vessel, which is weighed at regular intervals with a precipitation scale.

As amount of precipitation is determined by subtracting the Vessel weight from the total weight, the dry vessel is weighed before observation.

A rain-measuring glass may be used instead of a precipitation scale. To measure solid precipitation such as snow and hail with such a device, a known amount of

warm water is added to melt the precipitation; the total amount is then measured with the measuring glass, and the amount of warm water added is subtracted from the total to obtain the precipitation amount. The precipitation scale is graduated in millimeters based on the size of the rain gauge orifice.

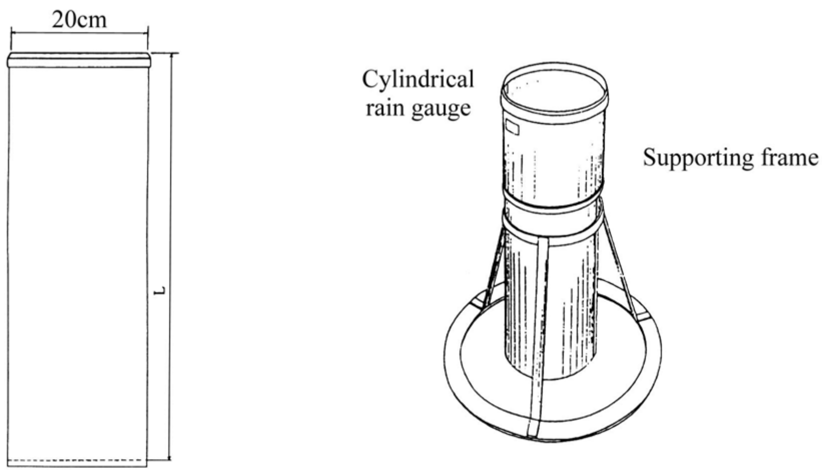


Figure 2.1 Cylindrical Rain Gauge

(ii) Ordinary Rain Gauges is shown in Figure 2.1 a type used at non-automated observatories. With such devices, the observer takes measurements using a rain-measuring glass at regular intervals. This type of rain gauge consists of a receptacle, a shell, a storage bottle, a storage vessel and a rain-measuring glass, which is a measuring cylinder Graduated in precipitation amounts based on the diameter of the receptacle’s orifice. The shell acts as a container for the storage bottle and the storage vessel. The storage vessel is a cylindrical metallic container that houses the storage bottle. The measuring cylinder is transparent, and is graduated in units of precipitation.

Rainwater entering through the receptacle accumulates in the storage bottle, and the precipitation amount is measured with the measuring glass. Rainwater that overflows from the storage bottle enters the storage vessel.

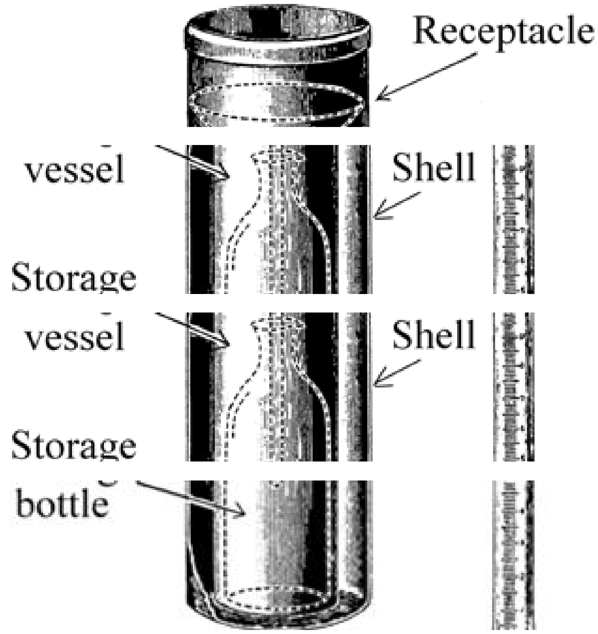


Figure 2.2. Ordinary rain gauge

The amount of overflow is also measured with the measuring glass, and is added to the amount of precipitation in the storage bottle.

(iii) Standard rain gauge shown in Figure 2.3 consists of metal canister open at the top. Inside the top is a funnel which drains into narrow cylinder. The ratio (the funnel cross-sectional area divided by the cylinder cross-sectional area) is 10 to 1 so every one inch of rain that falls from sky , 10 is collected in the cylinder. This allows accurate measurements down to 0.01 inches of rain (10 x 0.01=0.1 inches that actually collected in the cylinder). Measurements are made with specially calibrated ruler, in English or metric units, that take into account this ratio. The cylinder can hold 20 inch of water, so for rain events greater than 20 inch, the cylinder overflow into the canister. Once the full cylinder is emptied, the overflow water can be carefully poured into the cylinder and the measurement adds together. To prevent evaporation, a small mount (0.01 inch) on the water of light gauge oil is added to the cylinder weekly. The oil floats on the water surface, trapping water molecules that might otherwise evaporate. The 0.01 inch is then subtracted from the weekly precipitation measurement.

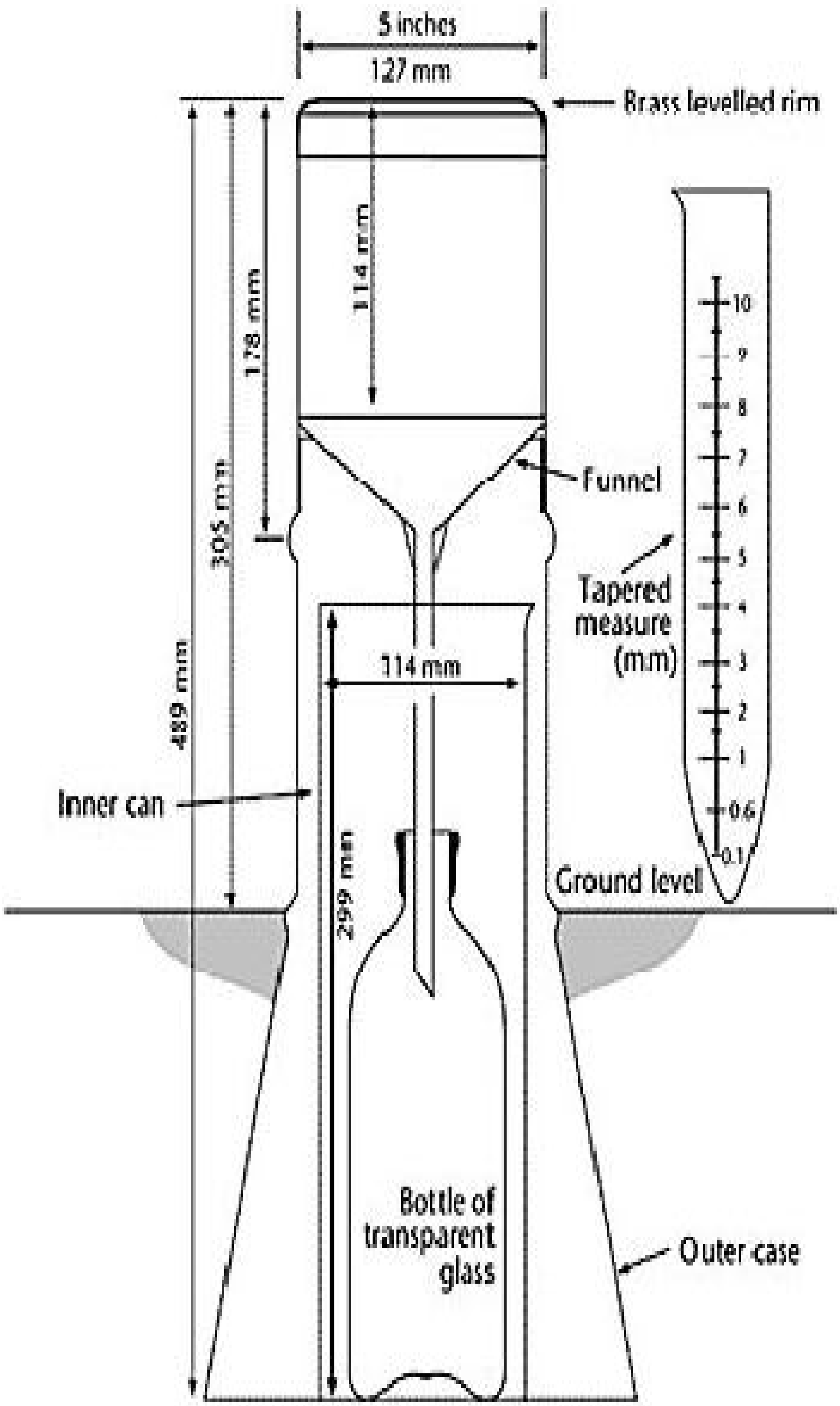


Figure 2.3 Standard Rain Gauge

2.2.2 Recording gauges

Such as tipping bucket have ability to record and store of data by itself, and remote reading can be taken by setting a recorder at a site distant from a gauge itself to enable automatic observation. This includes

(i) Siphon Rain Gauges enables automatic, continuous measurement and recording of precipitation. This type of rain gauge shown in Figure 2.4 consists of a receptacle to collect precipitation and a measuring part to measure and record its amount. The measuring part consists of a float with a recording pen attached a storage tank with a siphon to drain a fixed amount of water, and a clock-driven drum. Rainwater gathered by the receptacle is led from the rain receiver to the storage tank through an adjustment vessel. As a result, a float in the storage tank moves upward.

A recording penis connected to the float. When rainwater in the storage tank reaches a level equivalent to a fixed amount, it is drained by the siphon. This procedure is repeated as long as rainfall continues, and the pen repeats traces from zero to the maximum on the recording paper. When the rainfall stops, the pen traces a horizontal line.

. 1-receiver; 2-floater; 3-siphon; 4-recording needle;

5-drum with diagram; 6-clock mechanism

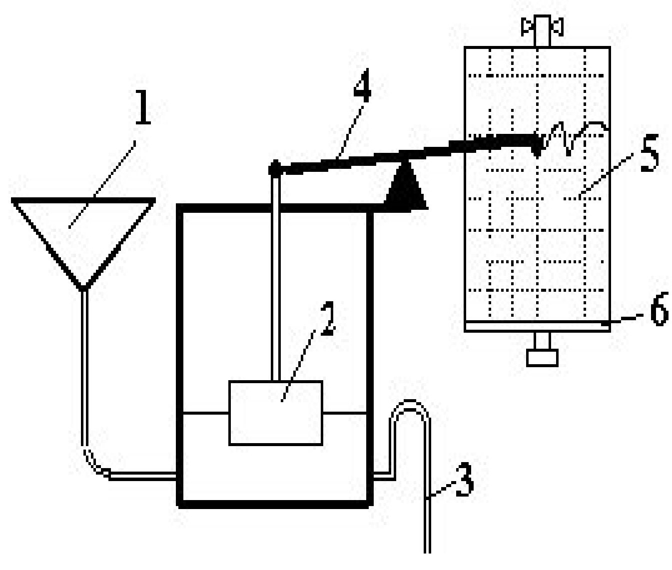


Figure 2.4 Siphon Rain Gauge

(ii) Tipping Bucket Rain Gauge generates an electric signal (i.e., a pulse) for each unit of precipitation collected, and allows automatic or remote observation with a recorder or a counter. The only requirement for the instrument connected to the rain gauge is that it must be able to count pulses. Thus, a wide selection of configurations and applications are possible for this measuring system. Solid precipitation can also be measured if a heater is set at the receptacle.

This type of rain gauge shown in Figure 2.5 consists of a receiver and a measuring part, with the receiver serving as the container for the device. The measuring part consists of a tipping bucket and a pulse-generating reed switch (or mercury switch) assembled within the receiver. The tipping buckets consist of two triangular vessels attached to the left and right of a rotation shaft, each with a capacity equivalent to a specific amount of precipitation. The reed or mercury switch is connected to these tipping buckets to generate an electrical signal (i.e., a pulse) each time the buckets tip.

Operation: Rainwater collected in the receptacle is channeled through the funnel and poured into a tipping bucket. When it reaches a predetermined amount, the bucket tips and dumps the water into a drain cylinder, causing the reed switch to generate a pulse. Subsequent rainwater is poured into the other bucket. As long as precipitation continues, this operation is repeated and a pulse is generated each time a bucket tip.

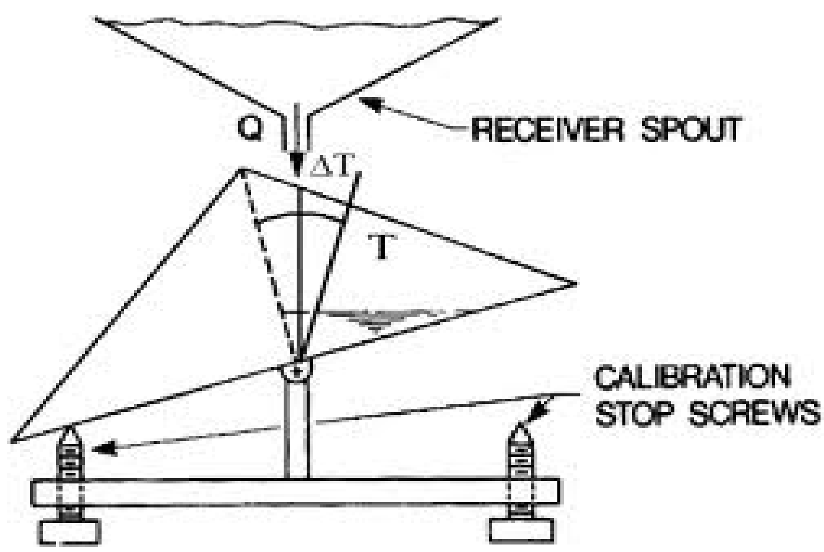


Figure 2.5 Tipping Bucket Rain Gauge

(iii) Weighing rain gauge shown in Figure 2.6 consists of a metal canister that houses a funnel, a pail that sits atop scale, and an analogue recorder (rotating drum with paper). Once 0.01

Inches of rain fall through the funnel into the pail, the weight of the water on the scale triggers the open on the recorder to move upwards. As more rain falls, the open continues to move upwards creating a time series of rain fall amount on the chart paper of the rotating drum, which is turned by a clock mechanism. Although precipitation is measure by its weight, it is converted to inches on the chart.

Advantage it is can estimate intense rain and disadvantage it more expensive and need more maintenance. The chart paper is replaced weekly, but the pail is emptied only when it becomes too full, every two month or so.

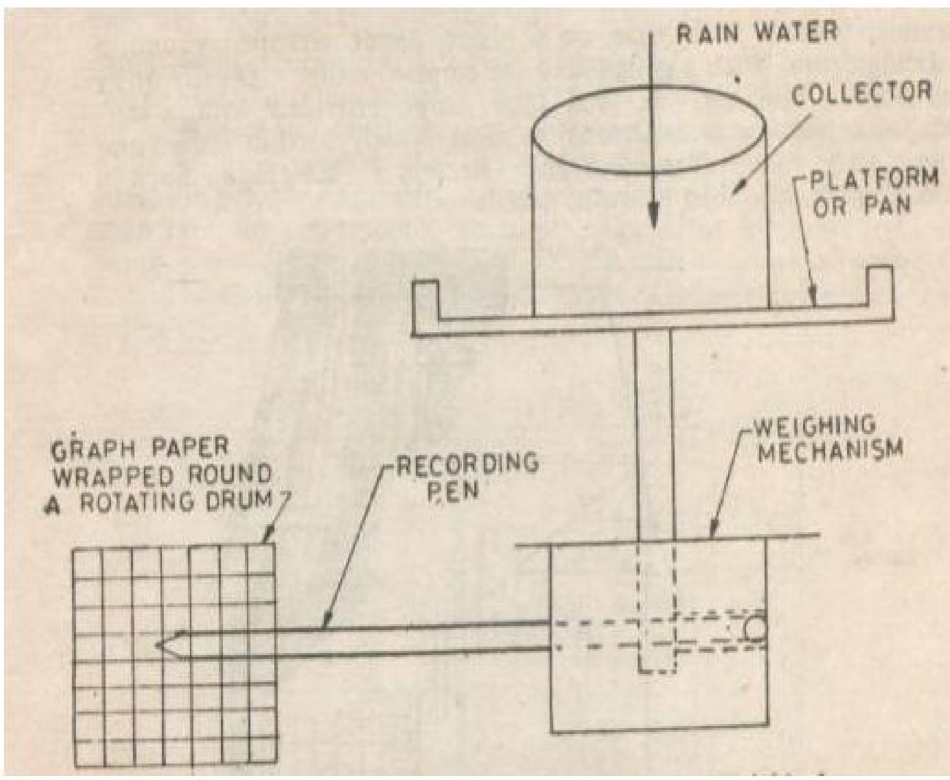


Figure 2.6 Weighing Rain Gauge

(iv) Optical rain gauge this have row of collection funnels. In an enclosed space below each is a laser diode and phototransistor detector .when enough water is collected to make a single drop, it drips from bottom, falling into laser beam bath. The sensor is set at right angles to the laser, so that enough light is scattered to be detected as sudden flash of light. The flashes from these photo detectors are then read and transmitted or recorded.

2.3 Exposure

As the environment of the instrument's location significantly influences observation of precipitation, the surroundings must be considered when selecting the observation site.

2.3.1 Conditions of exposure

To ensure representative observation, the following environmental conditions should be considered as far as possible:

a. The airflow around the rain gauge should be as horizontal as possible. Avoid sites that are concave, elevated or tilted. Choose a site far from precipices or mountain ridges, where local winds are strongly distorted. Avoid sites where wind blows through or stagnates. Building rooftops should not be considered.

b. Choose sites away from other instruments, trees or buildings. Ideally, the instrument should be installed at a distance from such objects equivalent to at least two to four times their height.

c. As the wind speed near the ground increases with height, the efficiency of precipitation collection decreases the higher a gauge is placed. Accordingly the receptacle should be placed as low as possible.

d. However, too low a setting will result in the entry of splashed rainwater from the ground or the introduction of ground snow in the case of a snowstorm.

e. The ground surface around the rain gauge should be flat and covered with short grass (lawn) or gravel to prevent raindrops from splashing into the unit from outside.

2.3.2 Limitation of accuracy:

The most significant influences on the accuracy of precipitation measurement are the environment and wind at the installation site rather than the performance of the instrument itself. Such influences are difficult to eliminate. Additionally, because precipitation is strongly characterized by locality, it is difficult to choose observation sites with a sufficient level of representation. Site selection without such consideration may result in observations that have very poor accuracy. It is strongly advisable to shield rain gauges from the wind or install them in an optimum observation environment. Satisfying the installation conditions outlined in (1) will ensure high accuracy of observation.

Wind exerts a significant influence on the observation of precipitation with snow and rain gauges, and there is no way to avoid its effects. However, accurate collection of precipitation in a rain gauge is possible when the wind around the receptacle is horizontal

And its speed is equal to that at ground level or when no vortices develop near the gauge so the windshields effective in reducing the influence of wind.

2.4 Microcontroller

Basically, a microcontroller is a device which integrates a number of the components of a microprocessor system onto a single microchip and optimized to interact with the outside world through on-board interfaces other specialized circuits all in one package.

On the other hand, a microprocessor is normally optimized to co-ordinate the flow of information between separate memory and peripheral devices which are located outside it. Connections to a microprocessor include address, control and data busses that allow it to select one of its peripherals and send to or retrieve data from it. Because a microcontroller's processor and peripherals are built on the same silicon, the devices are self-contained and rarely have any bus structures extending outside their packages so a microcontroller incorporates into the same microchip as shown in Figure (2.7)

- The CPU core
- Memory (both ROM and RAM)
- Some parallel digital I/O

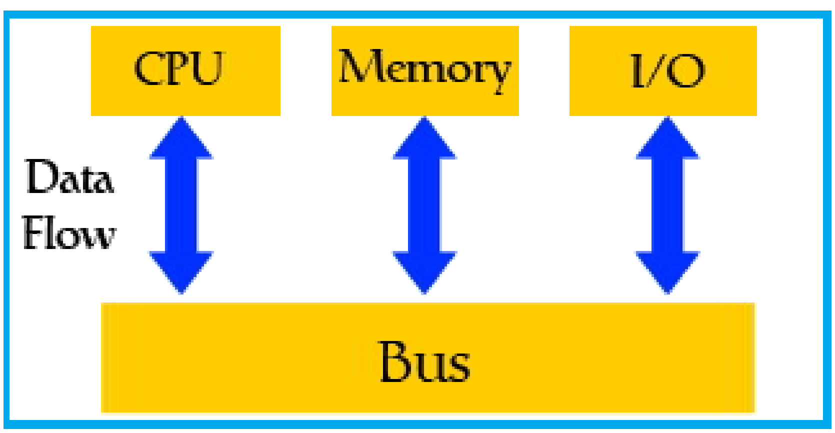


Figure 2.7 Microcontroller's Fundamental Components

Microcontrollers will also combine other devices as shown in Figure 2.8

- A Timer module to allow the microcontroller to perform tasks for certain time periods.
- A serial I/O port to allow data to flow between the microcontroller and other devices such as a PC or another microcontroller.

- An ADC to allow the microcontroller to accept analogue input data for processing.

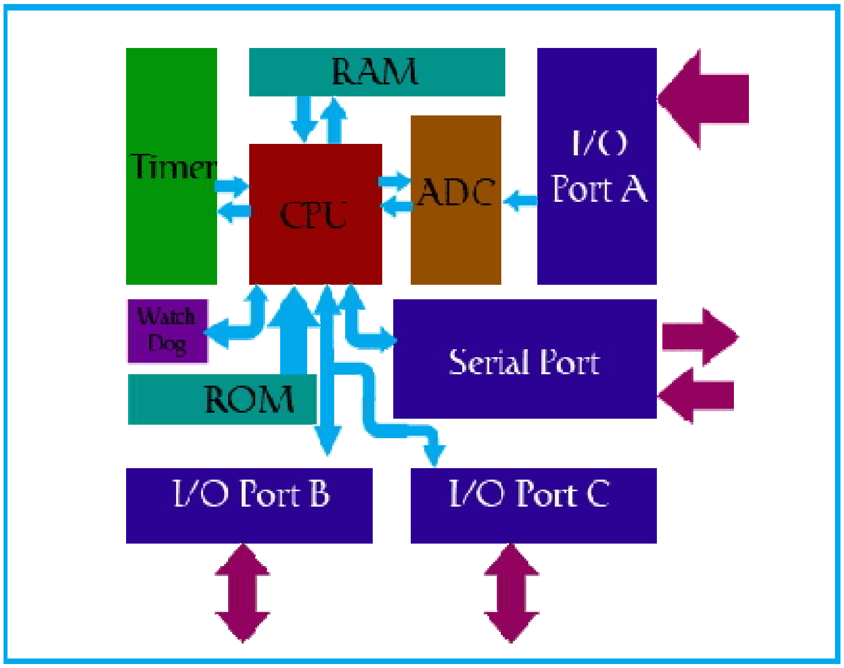


FIG 2.8 Microcontroller Building Block

2.4.1 Memory Unit

Memory is part of the microcontroller whose function is to store data. The easiest way to explain it is to describe it as one big closet with lots of drawers. If we suppose that we marked the drawers in such a way that they cannot be confused, any of their contents will then be easily accessible. It is enough to know the designation of the drawer and so its contents will be known to us for sure.

Memory components are exactly like that. For a certain input we get the contents of a certain addressed memory location and that's all. Two new concepts are brought to us: addressing and memory location.

Memory consists of all memory locations, and addressing is nothing but selecting one of them. This means that we need to select the desired memory location on one hand, and on the other hand we need to wait for the contents of that location. Besides reading from a memory location, memory must also provide for writing onto it. This is done by supplying an additional line called control line. We will designate this line as R/W (read/write). Control line is used in the following way: if r/w=1, reading is done, and if opposite is true then writing is done on the memory location. Memory is the first element, and we need a few operation of our microcontroller.

2.4.2 Central Processing Unit

The block that will have a built in capability to multiply, divide, subtract, and move its contents from one memory location onto another are called "central processing unit" (CPU). Its memory locations are called registers.

Registers are therefore memory locations whose role is to help with performing various mathematical operations or any other operations with data wherever data can be found.

Look at the current situation. We have two independent entities (memory and CPU) which are interconnected, and thus any exchange of data is hindered, as well as its functionality.

If, for example, we wish to add the contents of two memory locations and return the result again back to memory, we would need a connection between memory and CPU. Simply stated, we must have some "way" through data goes from one block to another.

2.4.3 Data Bus

That" way "is called bus, physically it represent a group of 8, 16 or more wires. There are two types of buses: address and data bus. The first one consist of as many lines as the mount of memory we wish to address, and the other one is as wide as data. First one serves to transmit address from CPU memory and the second to connect all blocks inside microcontroller

As far as functionality, the situation has improved, but a new problem has also appeared: we have a unit that's capable of working by itself, but which does not have any contact with the outside world, or with us! In order to remove this deficiency, let's add a block which contains several memory locations whose one end is connected to the data bus, and the other has connection with the output lines on the microcontroller which can be seen as pin in the electronic components.

2.4.4 Input-output unit

Those locations we've just added are called "ports". There are several types of ports: input, output or bidirectional ports. When working with ports, first of all it is necessary to choose which port we need to work with, and then to send data to, or take from the port.

2.4.5 Other Microcontroller Devices

Timer unit: Since we have the serial communication explained, we can receive, send and process data. However, in order to utilize it in industry we need a few additionally blocks. One of those is the timer block which is significant to us because it can give us information about time, duration, protocol etc.

The basic unit of the timer is a free-run counter which is in fact a register whose numeric value increments by one in even intervals, so that by taking its value during periods T1 and T2 and on the basis of their difference we can determine how much time has elapsed. This is a very important part of the microcontroller whose understanding requires most of our time.

This task is performed by a block for analog to digital conversion or by an ADC. This block is responsible for converting an information about some analog value to a binary number and for follow it through to a CPU block so that CPU block can further process it. Finally, the microcontroller is now completed, and all we need to do now is to assemble it into an electronic component where it will access inner blocks through the outside pins. The picture below shows what a microcontroller looks like inside.

Fig 2.9 shows the microcontroller architecture. Thin lines which lead from the center towards the sides of the microcontroller represent wires connecting inner blocks with the pins on the housing of the microcontroller so called bonding lines. For a real application, a microcontroller alone is not enough. Beside a microcontroller, programs that would be executed are needed, and a few more elements which make up an interface logic towards the elements of regulation.

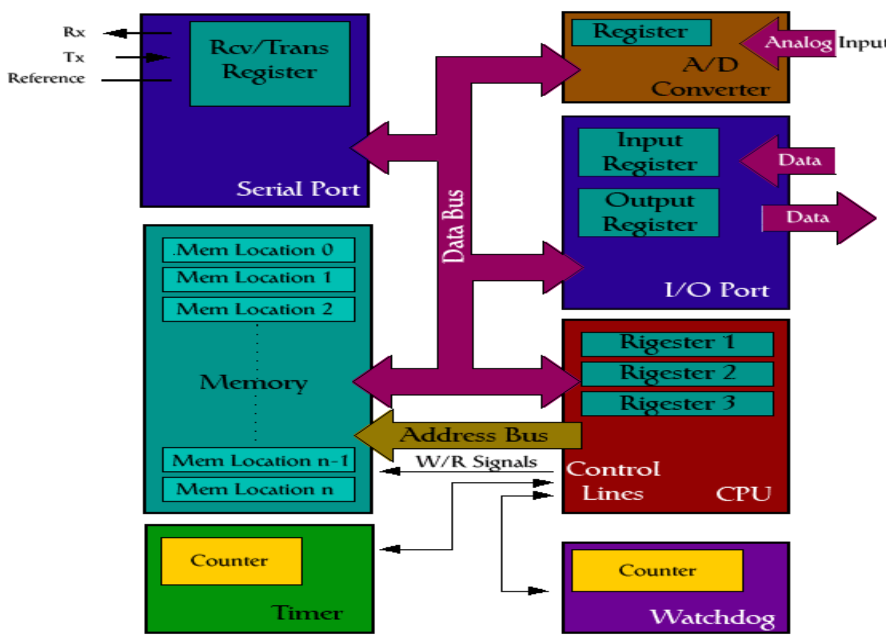


FIGURE 2.9 Microcontroller Architecture

Program writing is a special field of work with microcontrollers and is called "programming". Programming can be done in several languages such as Assembler, C and Basic which are most commonly used languages. Assembler belongs to lower level languages that are programmed slowly, but take up the least amount of space in memory and gives the best results where the speed of program execution is concerned. As it is the most commonly used language in programming microcontrollers it will be discussed in a later chapter. Programs in C language are easier to be written, easier to be understood, but are slower in executing from assembler programs. Basic is the easiest one to learn and its instructions are nearest a man's way of reasoning, but like C programming language it is also slower than assembler. In any case, before you make up your mind about one of these languages you need to consider carefully the demands for execution speed, for the size of memory and for the amount of time available for its assembly.

After the program is written, we would install the microcontroller into a device and run it. In order to do this we need to add a few more external components necessary for its work. First we must give life to a microcontroller by connecting it to a power supply (power needed for operation of all electronic instruments) and oscillator whose role is similar to the role that heart plays in a human body. Based on its clocks microcontroller executes instructions of a program. As it receives supply microcontroller will perform a small check up on itself, look up the beginning of the program and start executing it. How the device will work depends on many parameters, the most important of which is the skillfulness of the developer of hardware, and on programmer's expertise in getting the maximum out of the device with his program.

2.5 Timer 555

The 555 timer shown in Figure 2.10 the first introduced around 1971 by the Signetics Corporation as the SE555/NE555 and was called "**The IC Time Machine**".

The 555 is a single-chip version of a commonly used circuit called a multivibrator, which is useful in a wide variety of electronic circuits. The 555 timer chips probably the most popular integrated circuit ever made.

The 555 chips is used for basic timing functions, such as turning a light on for a certain length of time, or it can used to create a warning light that flashes on and off. Also it is used to produce musical notes of a particular frequency, or it is used to control positioning of a servo device.

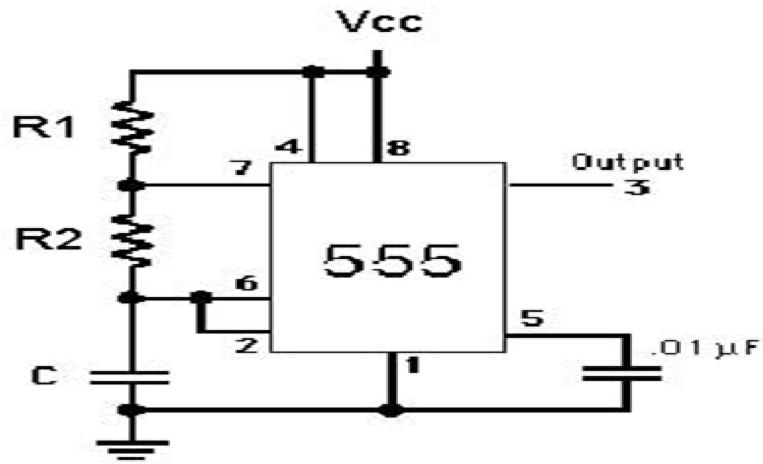


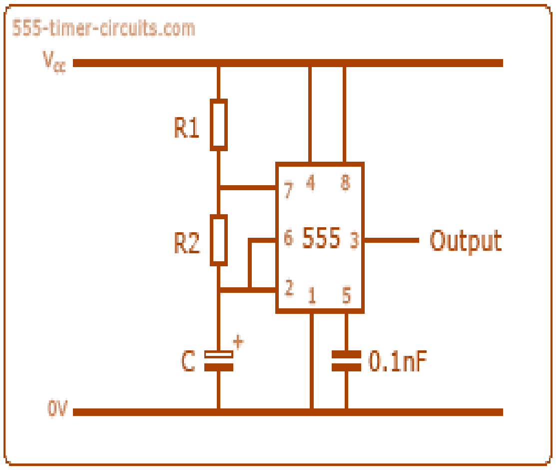
Figure 2.10 Timer555

The 555, in fig. 1 and fig. 2 above, come in two packages, either the round metal-can called the 'T' package or the more familiar 8-pin DIP 'V' package. About 20-years ago the metal-can type was pretty much the standard (SE/NE types). The 556 timer is a dual 555 version and comes in a 14-pin DIP package, the 558 is a quad version with four 555's also in a 14 pin DIP case.

The 555 has three main operating modes, Monostable, Astable, and Bistable. Each mode represents a different type of circuit that has a particular output.

An Astable Circuit shown in Figure 2.11 has no stable state - hence the name "a stable". The output continually switches state between high and low without any intervention from the user, called a 'square' wave. This type of circuit could be used to give a mechanism

intermittent motion by switching a motor on and off at regular intervals. It can also be used to flash lamps and LEDs, and is useful as a 'clock' pulse for other digital ICs and circuits.



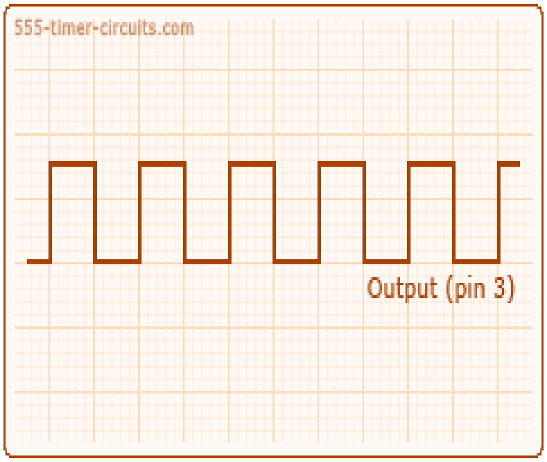


FIG URE 2.11 A stable Mode

A Monostable Circuit shown in Figure 2.12 produces one pulse of a set length in response to a trigger input such as a push button. The output of the circuit stays in the low state until there is a trigger input, hence the name "monostable" meaning "one stable state". This type of circuit is ideal for use in a "push to operate" system for a model displayed at exhibitions. A visitor can push a button to start a model's mechanism moving, and the mechanism will automatically switch off after a set time.

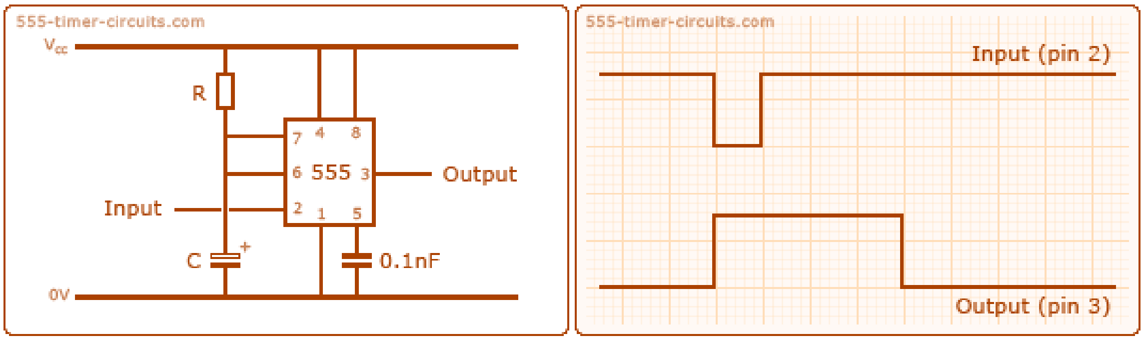


Figure 2.12 Monostable Modes

A BISTABLE MODE or what is sometimes called a Schmitt Trigger shown in Figure 2.13 has two stable states, high and low. Taking the Trigger input low makes the output of the circuit go into the high state. Taking the Reset input low makes the output of the circuit go into the low state. This type of circuit is ideal for use in an automated model railway system where the train is required to run back and forth over the same piece of track. A push button (or reed switch with a magnet on the underside of the train) would be placed at each end of the track so that when one is hit by the train, it will either trigger or reset the bitable. The output of the 555 would control a DPDT relay which would be wired as a reversing switch to reverse the direction of current to the track, thereby reversing the direction of the train.

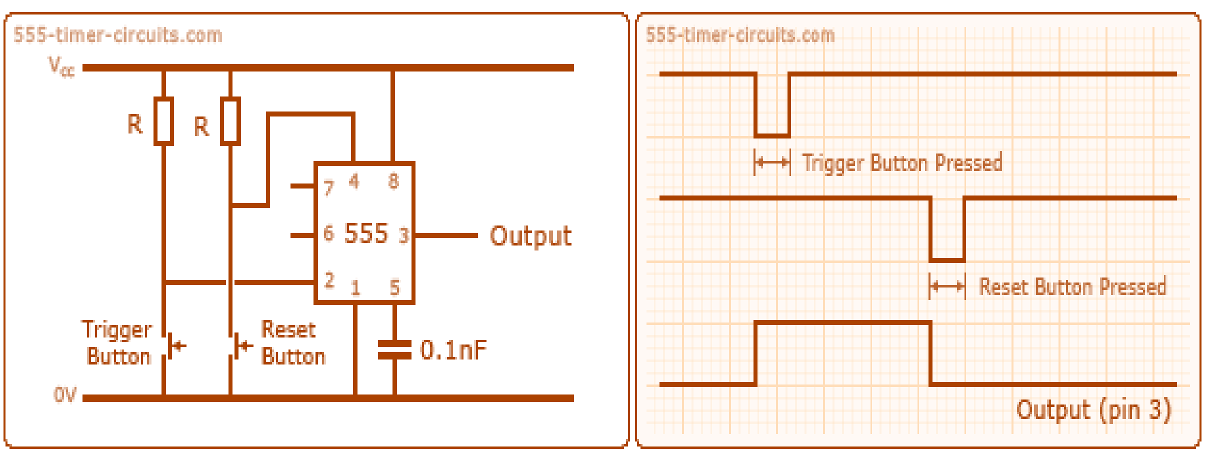


Figure 2.13 Bistable Mode

2.6 Reed switch

The **reed switch** shown in Figure 2.14 is an electrical operated by an applied magnetic field it was invented at Bell telephone laboratories in 1936 by W. B. Ellwood. It consists of a pair of contacts on ferrous metal reeds in a hermetically sealed glass envelope. The contacts may be normally open, closing when a magnetic field is present, or normally closed and opening when a magnetic field is applied. The switch may be actuated by a coil, making a reed relay or by bringing a magnet near to the switch. Once the magnet is pulled away from the switch, the reed switch will go back to its original position. An example of a reed switch's application is to detect the opening of a door, when used as proximity for a burglar alarm.

A Reed Switch consists of two ferromagnetic blades (generally composed of iron and nickel) hermetically sealed in a glass capsule. The blades overlap internally in the glass capsule with a gap between them, and make contact with each other when in the presence of a suitable magnetic field. The contact area on both blades is plated or sputtered with a very hard metal usually rhodium or ruthenium. These very hard metals give rise to the potential of very long life times if the contacts are not switched with heavy loads. The gas in the capsule usually consists of Nitrogen or some equivalent inert gas. Some reed switch to increase tier ability to switch and standoff high, have an internal vacuum. The reed blades act as magnetic flux conductors when exposed to an external magnetic field from either a permanent magnet or electromagnetic coil. Poles of opposite polarity are created and the contacts close when the magnetic force exceeds the spring force of the reed blades. As the external magnetic field is reduced so that the force between the reeds is less than the restoring force of the reed blades, the contacts open.

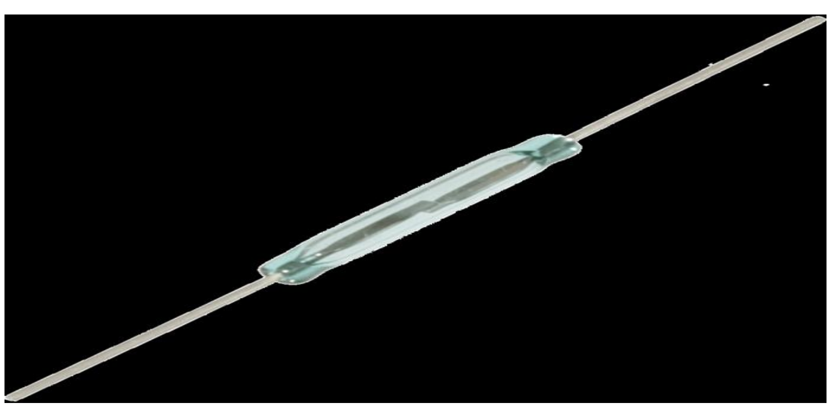


Figure 2.14 Reed Switch

2.7 Flip- flop

In electronics, a **flip-flop** or **latch** is a circuit that has two stable states and can be used to store state information. A flip-flop is a bistable multivibrator. The circuit can be made to change state by signals applied to one or more control inputs and will have one or two outputs. It is the basic storage element in sequential logic. Flip-flops and latches are a fundamental building block of digital electronics systems used in computers, communications, and many other types of systems.

Flip-flops and latches are used as data storage elements. Such data storage can be used for storage of *state* and such a circuit is described as sequential logic. When used in a finite-state machine. The output and next state depend not only on its current input, but also on its current state (and hence, previous inputs). It can also be used for counting of pulses, and for synchronizing variably-timed input signals to some reference timing signal.

Flip-flops can be either simple (transparent or opaque) or clock (synchronous or edge-triggered); the simple ones are commonly called latches. The word *latch* is mainly used for storage elements, while clocked devices are described as *flip-flops*. A latch is level-sensitive, whereas a flip-flop is edge-sensitive. That is, when a latch is enabled it becomes transparent, while a flip flop's output only changes on a single type (positive going or negative going) of clock edge.

Flip-flops can be divided into common types: the **SR** ("set-reset"), **D** ("data" or "delay")[)], **T** ("toggle"), and **JK** types are the common ones. The behavior of a particular type can be described by what is termed the characteristic equation, which derives the "next" (i.e., after the next clock pulse) output, *Q* in terms of the input signal(s) and/or the current output Q.

SR FLIP-FLOP type shown in Figure 2.15 has two inputs: Set and Reset. Two outputs: Q and Q' (Q' being the inverse of Q). The SR flip-flop can also have a clock input for a level driven circuit as opposed to a pulse driven circuit.

The operation of an SR flip-flop is as follows: The Set input will make Q go to 1 i.e. will 'set' the output. The Reset input will make the output Q go to 0 i.e. reset the

output. The scenario of having both Set and Reset at logic 1 is not allowed as this is not a logical pair of inputs.

JK FLIP-FLOP type show in Figure 2.16 consists of two data inputs: J and K, and one clock input. There are again two outputs Q and Q' (where Q' is the reverse of Q).

OPERATION:

A. **W**hen J=K=0, the current output will carry through to the next state. e.g. Current state Q = Next state Q

B. When J=0 and K=1, the next state output will be put to 0. This happens regardless of the present state output.

C. When J=1 and K=0, the next state output will be asserted (put to 1). This happens regardless of the present state output.

D. When J=K=1, the next state output will be the inverse of the current state output. e.g. Current state Q' = Next state Q.

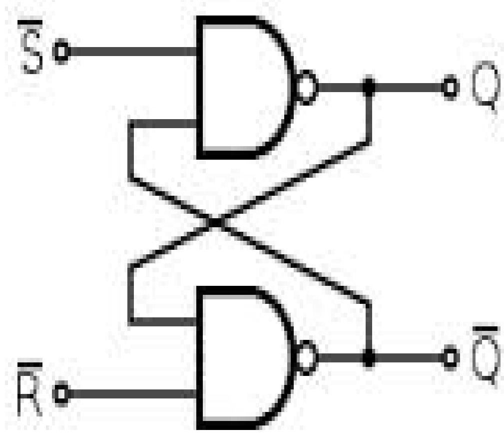


Figure 2.15 SR FLIP-FLOP

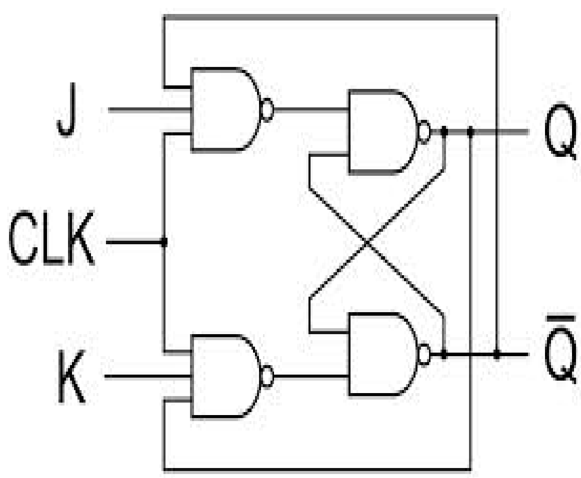


Figure 2.16 JK FLIP-FLOP

Chapter Three

Hardware Design and Instrumentation

3.1 Introduction

In this study, the methodology intended is to design, and implement a tipping bucket rain gauge by using microcontroller which receives pulse signal from sensor unit, which within microcontroller and software will be converted into reading measure value (figures) in PC screen which is serially connected to microcontroller for display and saving.

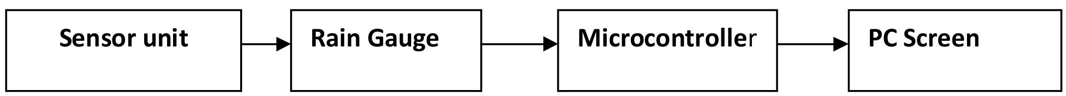


Figure 3.1 Tipping Bucker Rain Gauge

Data Acquisition System

3.2 Detailed Design

The digital rain gauge includes the collector, the microcontroller for USB conversion and software. Within the collector are the magnetic sensor and tipping bucket. These elements provide the basic mechanism for measuring and translating a unit of rainfall into a pulse. The signal then travels to the Microcontroller where it is translated into a USB signal and sent to the PC. Once in the PC the signal will be read by software and presented to the user in various useful forms. The data flow from rainfall to the user interface can be seen in figure 3.2

1. This rainfall being collected by the funnel. It is passed into the tipping bucket assembly.
2. Accumulated rainfall in the tipping bucket assembly.
3. When accumulated rain reaches a certain threshold bucket tip trigger process for reset accumulator.
4. When the bucket tips, it trips a magnetic sensor which sends an electrical pulse.
5. Converts the electric pulse into a USB digital signal. This signal is sent to host computer.
6. Host computer calculate rainfall as float variable.
7. This is the rainfall that has accumulated since the last calculation event.
8. Rainfall data can be manipulated in various ways during this step. Controlled by the user through step9.
9. Data is relayed to user. User can enact different manipulation of data through GUI. Functions such as total rainfall, rainfall rate, and unit conversions will be supported.

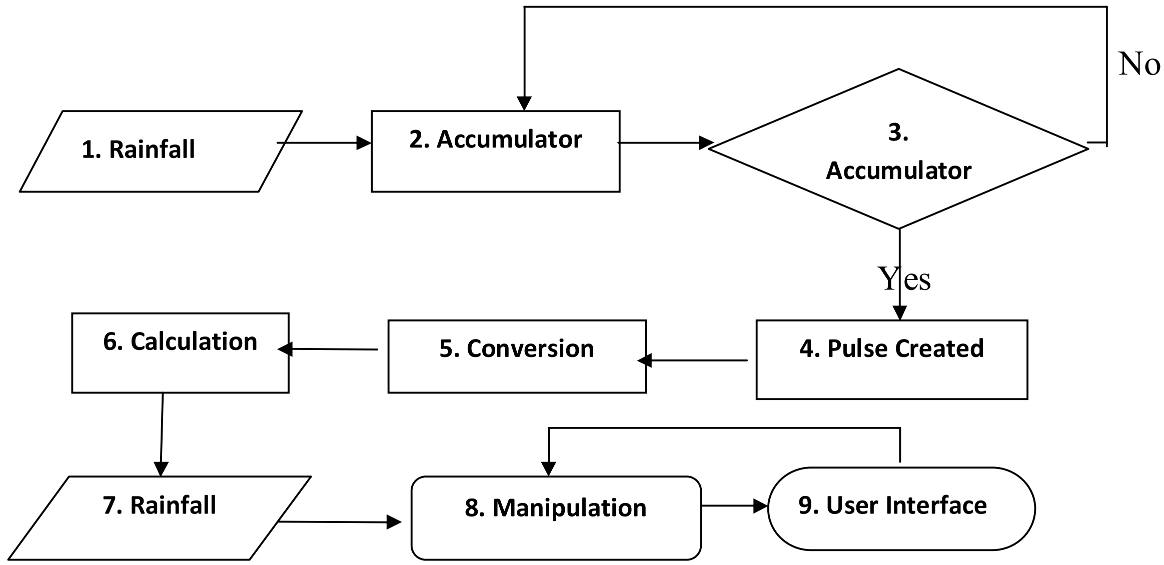


FIGURE 3.2 Block Diagram of Data Flow

This project presents two problems hardware and software design. The hardware includes mechanical and electrical components. Mechanical parts include design of tipping bucket with axial support, funnel, gauge main body, tipping bucket adjust and balance. Electrical hardware begins with choosing a sensor and transmission cable. A hardware translation must exist between the sensor output and the PC input port.

The software interface should be simple to use so that even novice computer user can utilize it.

To choose a sensor the rain collection mechanism must be first chosen. Knowing the mechanism will tell how a unit of collected rain will be translated into electrical signal. The sensor serves as translator between the physical event and the resulting signal. The software will need to read the signal from computer port, interrupt this data in appropriate manner and compute useful figures for the user in a clean GUI.

Solving the hardware problem involves design rain gauge sensor which can be place external to a house or business. A tipping bucket mechanism serves as the rain collector (Fig 3.3) each unit of collected rain will cause a small bucket within the sensor unit to shift. The shift in bucket movement will be captured by a magnetic sensor.

3.3 Hardware implementation

This section will discuss the components that had been used included sensor unit mechanism, Microcontroller (PIC16F84A), Timer555, Flip-flop, MAX232 for RS232 serial communication between Microcontroller and PC.

3.3.1 Sensor unit mechanism design

The sensor unit's purpose is to quantize rainfall amounts into electronic pulses. Solving the hardware problem involves designing a rain gauge sensor. A tipping bucket mechanism as in Figure 3.3 serves as the rain collector. Each unit of collected rain causes a small bucket within the sensor unit to shift. The shift in bucket movement will be captured by a magnetic sensor or reed switch of Figure 3.4). Also, each time that the bucket tips, the water that has accumulated is dumped out.

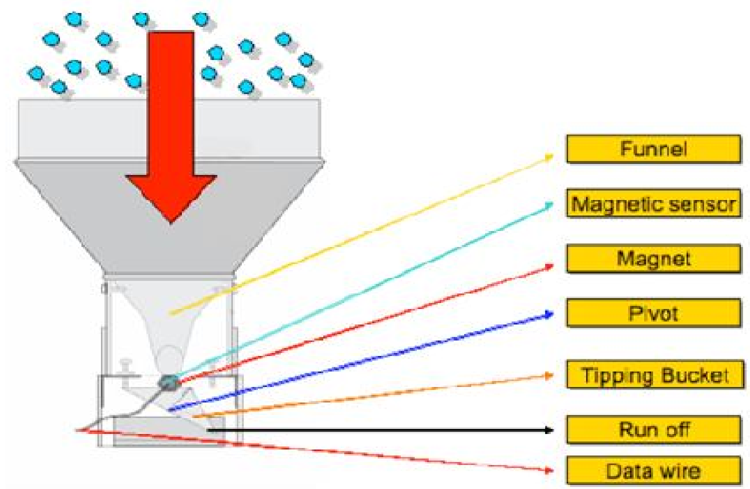


Figure 3.3 Collector Design

The signal generated by the magnetic sensor is sent down a two-conductor transmission line to the converter box which includes microcontroller and the other electronic components. The frequency of the pulses and the volume of water that each pulse represents are determined by several factors. One of which is the size of the upper orifice of the collector funnel. The larger this surface area is the more rain is concentrated into the tipping bucket mechanism.

This affects how fast the bucket will fill. Assuming that we are holding the actual volume that the bucket can hold constant, the factor that determines the accumulated volume of the water is the distance of the pivot point from the bottom of the bucket assembly. If the pivot point is moved down from the bucket, more water will be required to cause a tip, so adjustable pins are used to control this distance. Shorter distances will cause the bucket to tip with a smaller volume of accumulated water.

We are concerned with this, because the granularity of the volume to pulse ratio can be controlled in this way. The ideal pivot point would accurately measure rainfall while staying within the frequency limits of the sensor that has been chosen. For the set of equipment the distance value that will be used is 26.6 mm.

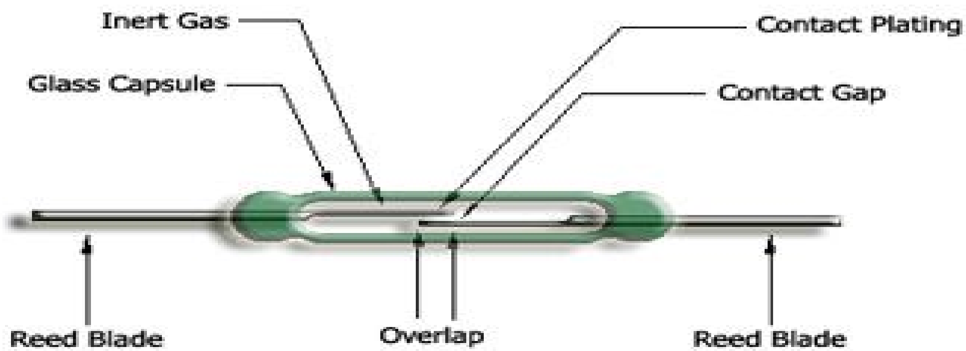


FIGURE 3.4 Reed Switch

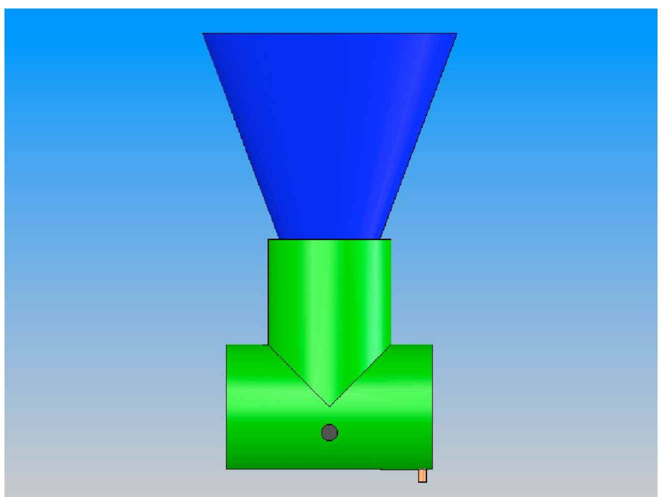


FIG 3.5 Sensor Unit Front View

The material unit as follow:

Collector funnel: Injection-molded plastic

Unit body: PVC T-Junction

Tipping bucket: folded sheet aluminum

Pivot dowel: Mild Steel

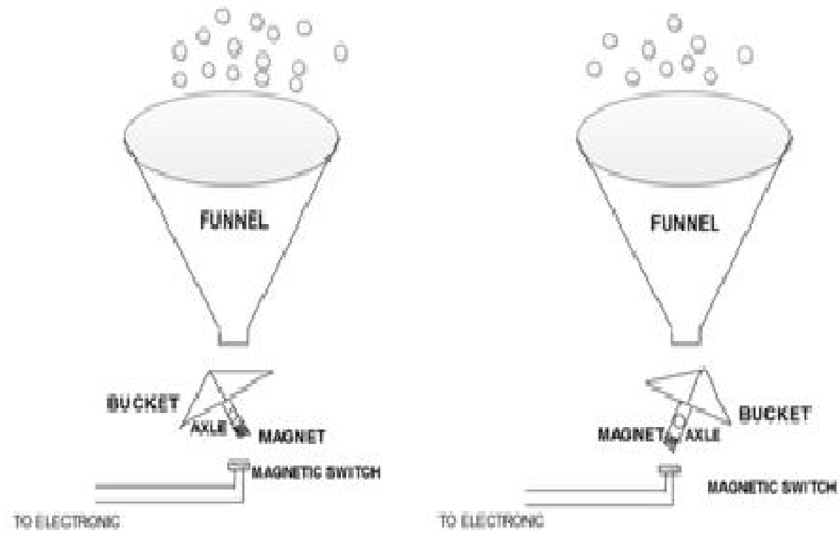


Figure 3.6 Tipping Bucker Rain Gauge Operation Method

3.3.2 PIC MICROCONTROLLER

A PIC microcontroller is a single-chip package that combines a microprocessor, room program memory, Ram variable memory, along with several input and output logic gates this make them a one –chip computer.

The PIC16F84A has 13 digital input/ output lines; there are two input-output ports, port A and port B, and each pin of each port can be set individually as an input or an output. As inputs it can either be high (+5v) or low (ground), as outputs it can either driven high or low. We can decide which lines are inputs and outputs, and can even switch a line from being an input to being an output during program execution. The CPU of course, the brain of the brain of the computer, It read and executes instructions from the program memory. As it does so, it can store and retrieve data in working memory (RAM).some CPU make distinctions between registers located within CPU and RAM located outside, the PIC doesn't and it's general-purpose working RAM also known as "file register". On F84 there are 68 bytes of general- purpose RAM located at address HEX 0C to HEX 4F.

Besides the general-purpose memory there is special working register where the CPU holds the data its working on. There are also several special function- registers each of which controls operation of the PIC on some way. The program memory of the F84 consists of flash memory EPROM, it can be recorded and erased electrically, and it can retain its contents when power off.

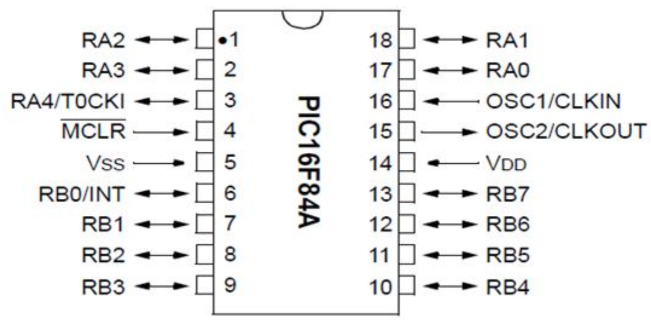


Figure3.7 PIC1684A Microcontroller Pin's

The PIC16F84A requires a few external components in order to operate, first a regulator voltage supply is needed, and the PIC16F84A can use a supply voltage from 3-6v DC, here we use voltage regulator which produce 5v.

Also an external clock signal is need; in this project we will use 4MHZ crystal for high speed and precise timing (along with two 22PF capacitors running from either lead to ground) will give an instruction speed of one million instructions per second in the PIC.

The other component is needed is a 10 k resistor to hold the reset line (pin4), this will cause PIC to reset (start executing its program from the beginning, like it would when first powered up)

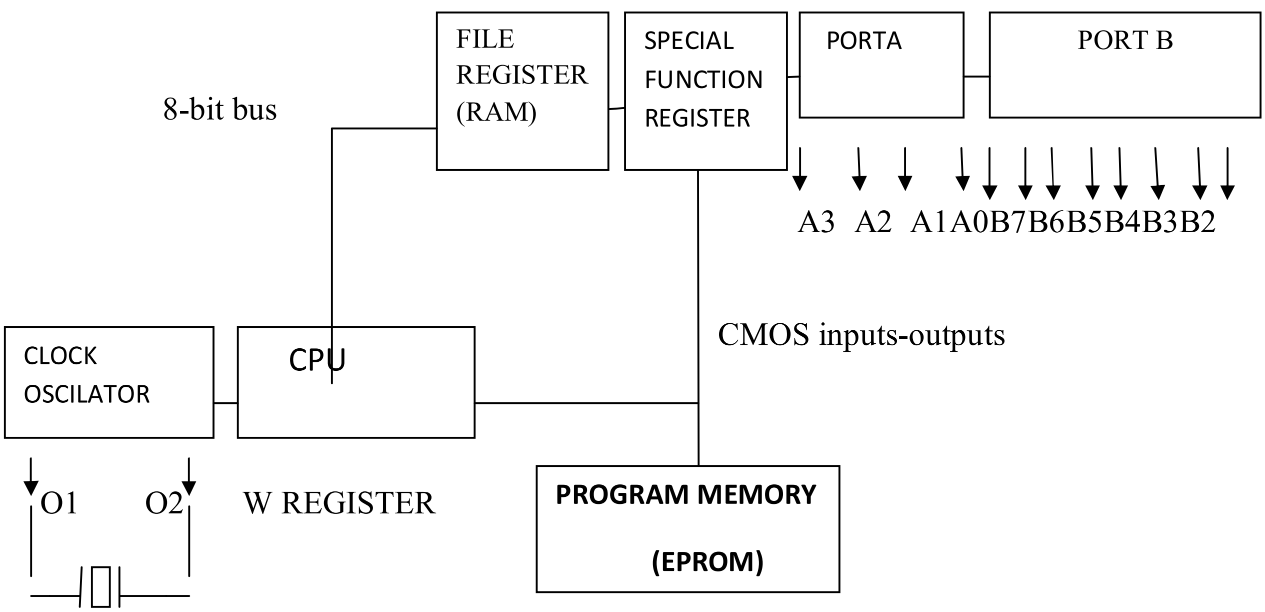


FIG 3.8 PIC16F84A Microcontroller Internal Components

3.3.3 Pulse Counting

It's fairly simple matter to connect the magnetic switch from the rain gauge to a +5v supply so we can drive one of the PIC's pin high whenever the switch closes. Then it's just a matter of watching that pin in the PIC code and incrementing a register each time the switch closes and then opens. But the PIC might miss momentary switch closure ,so the magnetic sensor require constant attention lest it misses a bucket tip and reads incorrectly, so we use FLIP-FLOP to latch onto momentary pulse from the magnetic switch and remember it long enough for the PIC to have a chance to detect it.

FLIP-FLOP is an electronic circuit made from digital gates that has ability to assume either of two a stable state depending on it's put given. The most basic FLIP-FLOP type is R-S input and Q-Q output. From the logic diagram below we can see that a momentary pulse on S input cause Q output to go high and stay high until a momentary pulse is receive high on the R input. In the S-RFLIP-FLOP, Sand R should not be allowed to go high at the same time because it results in an uncertain state for Q.

Here the more common FLIP-FLOP type in use is JK FLIP-FLOP, as shown in FIG 4.3. The JK has three inputs: J and K which function much like S and R in RS FLIP-FLPO and CK or clock input. The difference between the operation of RS FLIP-FLOP and JK FLIP-FLOP is that the output of the JK FLIP-FLOP will only change state on the leading edge of a pulse to CK line.

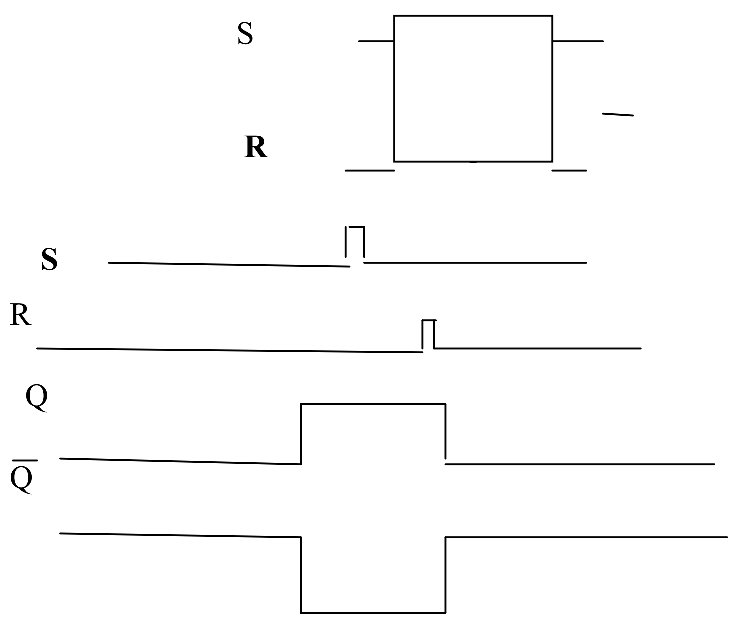


Figure 3.9 Logic Diagram of R-S Flip-Flop

From logic diagram, even though the J input goes high, the Q output does not go high until the subsequent pulse on the CK line. The same true for the K input taking Q low again- it does not happen until the pulse occur in the CK line.

The difference between JK and RS FLIP-FLOP it is okay for the both J and K inputs to be high at the same time. In fact that it's a very useful state because it causes the outputs to toggle each time a pulse is received on the clock line.

We are going to use this feature to latch onto pulses received due to momentary closing of the magnetic switch, and then all our PIC needs to do is sense state change in the corresponding input and increment its counter.

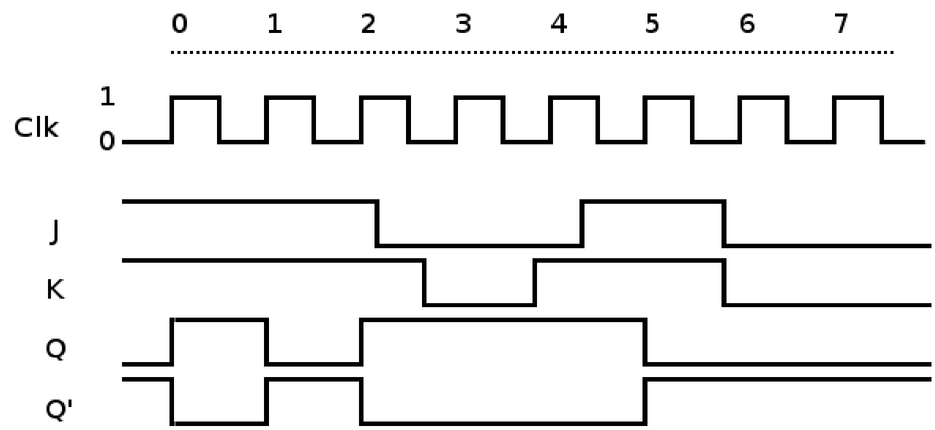


Figure 3.10 Logic Diagram For JK Flip-FLOP

3.3.4 Switch Debouncing

There is one other little issue that we need to deal with. Switches have a nasty habit of "bouncing" when closed or opened. Bouncing mean switch's contact do not closed or opened perfectly-instead, they bounce against each other a few times making and breaking the connection until come to the rest. In many cases this is no big deal, but here it results in several pulses in the clock line each time the switch closes instead of just one, so obviously this can royally mess with the accuracy of gauge. There are number of techniques that can be used to debounce a switch. We use Timer 555 to generate a fixed –width output pulse we use (mono stable mode) resistors and capacitors to set width of the output pulse (which about 0.1 second) so now we have reliable pulse train that should be easy to count with PIC.

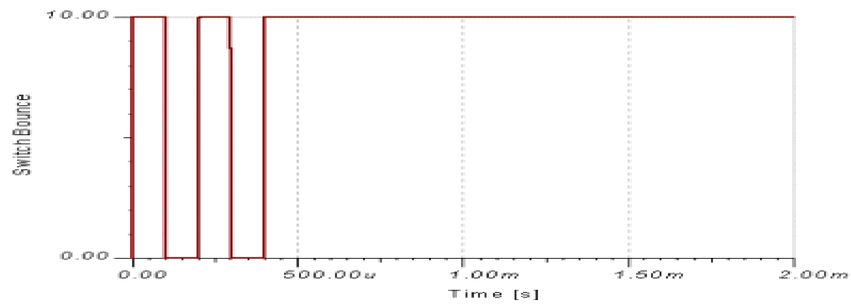


FIG 3.11 Switch Bounce

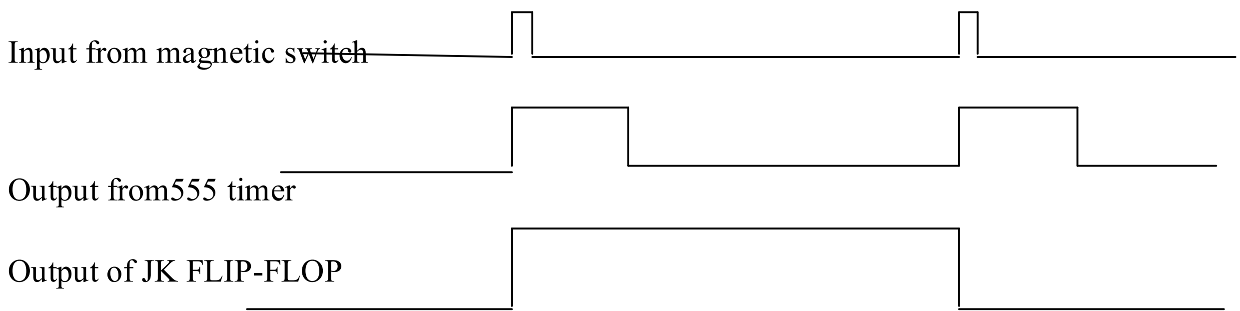


Figure 3.12 Rain Gauge Pulse-Sensing Logic Diagram

3.3.5 RS232 AND MAX232

The MAX232 IC is used to convert the TTL/CMOS logic levels to RS232 logic levels during serial communication of microcontrollers with PC. The controller operates at TTL logic level (0-5V) whereas the serial communication in PC works on RS232 standards (-25 V to + 25V). This makes it difficult to establish a direct link between them to communicate with each other.

The intermediate link is provided through MAX232. It is a dual driver/receiver that includes capacitive generator to supply RS232 voltage levels from a single 5v supply. Each receiver converts RS232 input to 5V TTL/CMOS levels. This receiver (R1, R2) can accept +-30v input. The drivers (T1, T2) also called transmitter convert the TTL/CMOS input level into RS232 level.

The transmitters take input from controller serial transmission pin and send the output to RS232's receiver. The receiver on the other hand take input from transmission pin of RS232 serial port and give serial output to microcontroller receiver pin. MAX232 needs four external capacitors that's value range from $2\mu F$ to$22\mu F$.

Microcontroller	**MAX232**		**RS232**
TX	T1/2 in	T1/2 out	RX
RX	R1/2 in	R1/2 out	TX

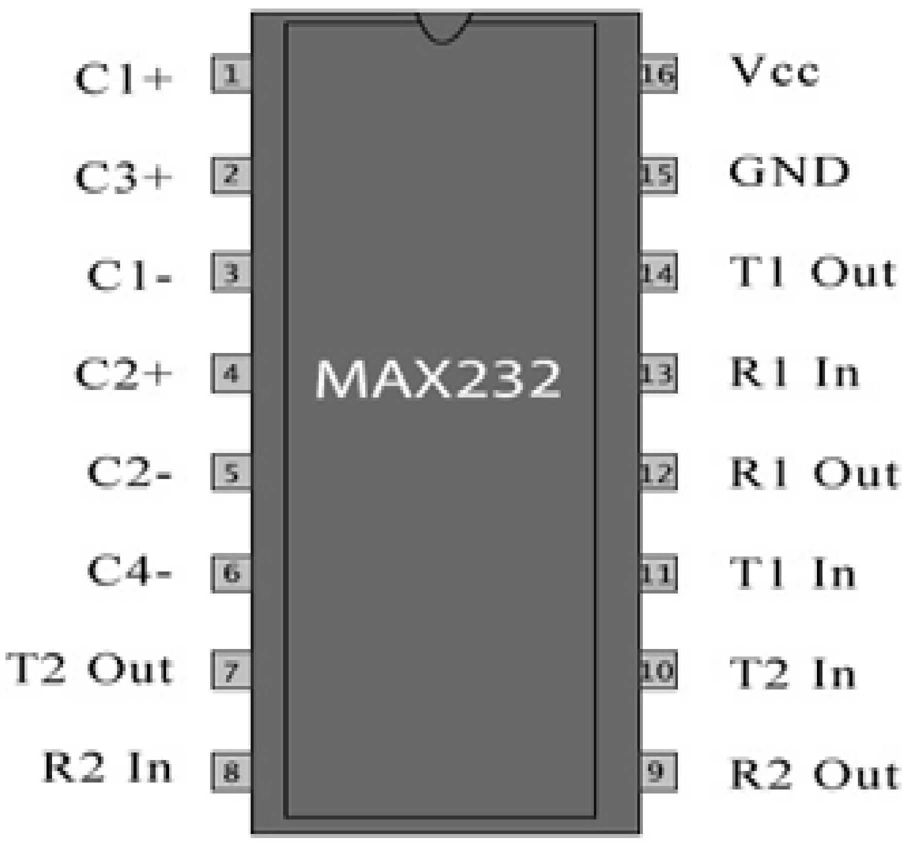


FIGURE 3.13 MAX 232 PINS

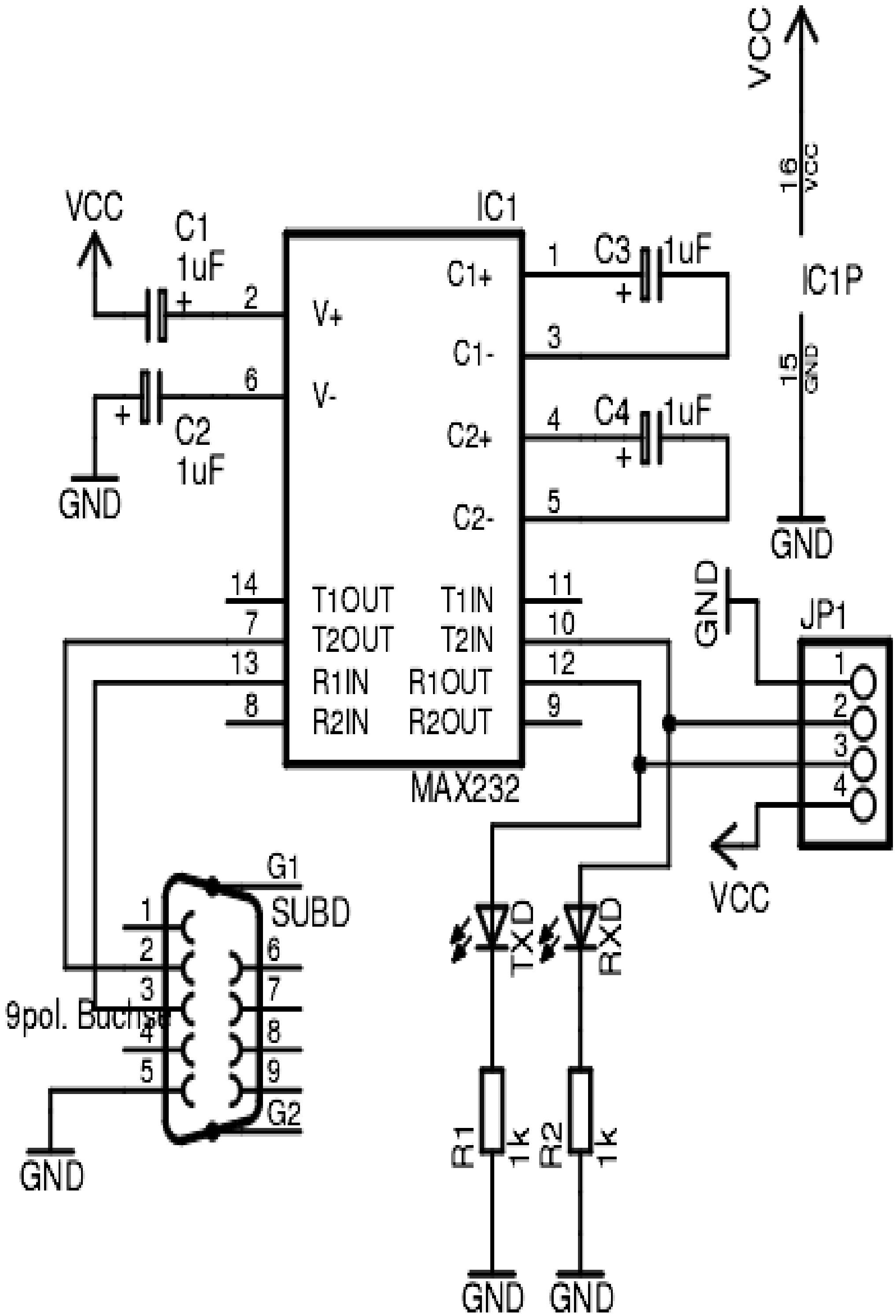


FIGURE 3.14 INTERFACING MAX232 TO RS 232

3.3.6 The rain gauge final built circuit

As shown second page this is the final construction of tipping bucket rain gauge electronic circuit diagram contain reed switch, timer555, flip-flop, microcontroller, max232, AND USB-to serial comp connect to PC. According to circuit diagram, all components will be installed on PCB board and finally the PCB board will fixed in closed metal case as seen in Figure 3.15.

Figure 3.15 Electronic circuit fixed inside metal case

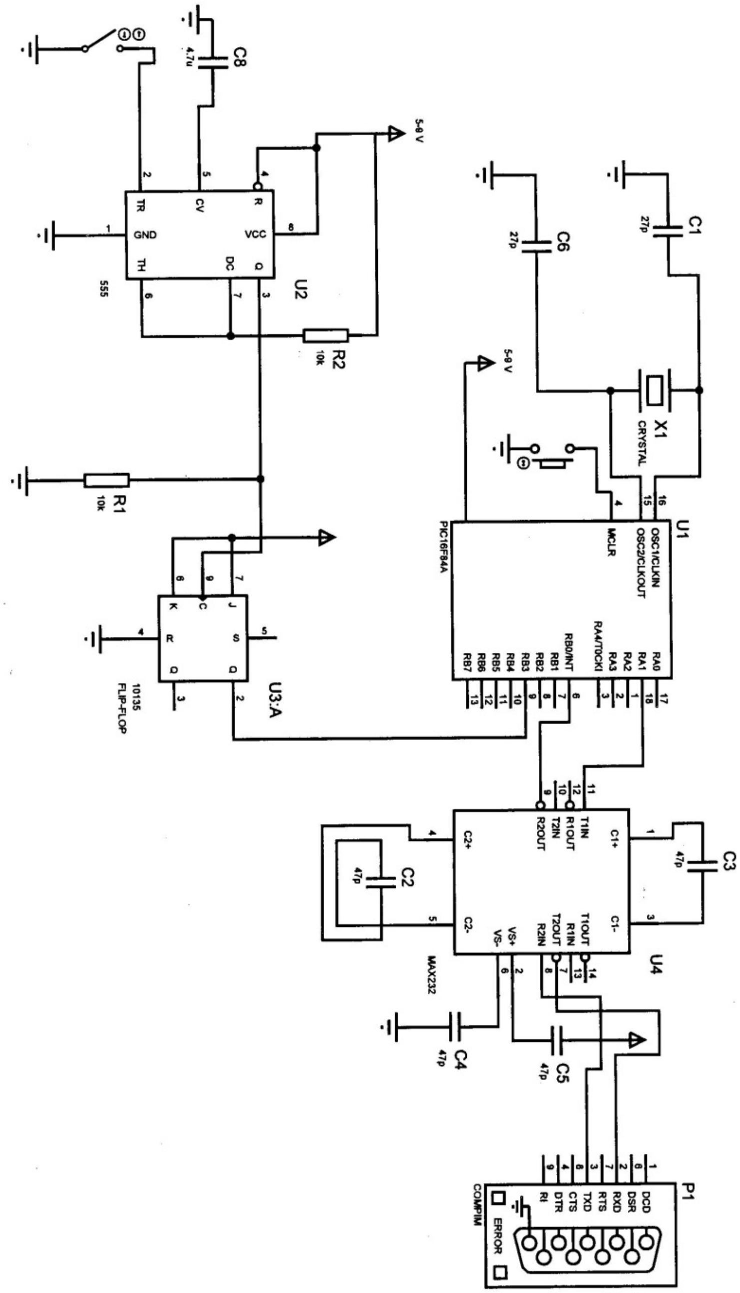


Figure 3.16 Rain Gauge Electronic Circuit diagram

3.3.7 Rain Gauge Electronic Circuit Description

The funnel shown in Figure 3.6 collects and concentrates the rain so that it ends up in either the left side or the right side of the bucket. When that side of the bucket gets full, it gets unbalanced and tips, pouring out its contents and putting the other side of the bucket in place for filling.

Each time the bucket tips, a magnet attached to the bucket passes by a magnetic switch, causing the switch to momentarily close. The PIC circuit will sense the switch closure and increment a counter for each bucket tip.

The PC software will ask the PIC circuit for the current tip count and convert that to an amount of rainfall by multiplying by a calibration value.

3.4 Software Design:

For software design or implementation M PLAB IDE is used to program microcontroller in assembly language. Besides, program was written in c code is used for user interface purpose to rainfall records in PC as seen in Appendix A.

3.4.1 Algorithm and Programming in MPLAB IDE

An algorithm has to be developed to make the microcontroller read the input and respond accordingly. Therefore, the algorithm is established and represented by a flow chart in FIG3.14 and FIG 3.15 represent flow chart of user interface program in PC.

Then the microcontroller flow chart is translated into assembly language and compiled using MPLAB, the PIC16F84A software development tool. As the funnel collects and concentrate rain so that it ends up in either the left side or right side of the bucket, when that side of the bucket gets full, it get unbalanced and tips, pouring out its contents and putting the other side of the bucket in place for filling. Each time the bucket tip, a magnet attached to the bucket passes by a magnet switch causing switch to momentarily close.

The microcontroller circuit senses the switch closure and increment a counter for each bucket tip, but rain sensor (reed switch) require constant attention lest it misses a bucket tip and read incorrectly, so that need some way of remembering that information until the PIC can get to it.

The solution is to use FLIP-FLOP to latch onto the momentary pulse from the magnetic switch and remembered it long enough for the PIC to have a chance to detect it. We use the JK FLIP- FLOP.

The PC software asks the microcontroller circuit for current tip count and converts that to an amount of rainfall by multiplying by a calibration value.

3.4.2 Signal Management

(A)Microcontroller: The technology considered here is PIC16F84A microcontroller with USB-to Serial adapter interface, reasons for using PIC microcontroller 16F84A it is:

- RISC (reduced instruction set computer) design
- Only thirty seven instructions to remember
- Its code extremely efficient, allowing the PIC to run with typically less program memory than its larger competitors.
- Its low cost, high clock speed
- Available in market.

(B) Sensor: the technology considers here is magnetic switch because the senor is both accurate and inexpensive. The switching rate is high.

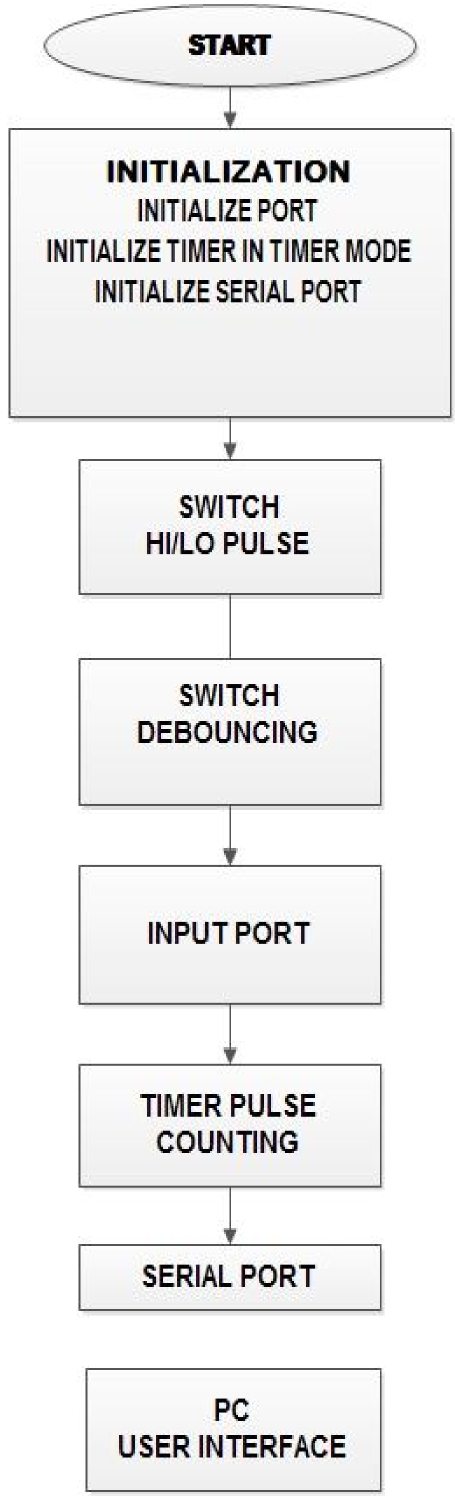


Figure 3.17 Flow Chart of Microcontroller's Main Program

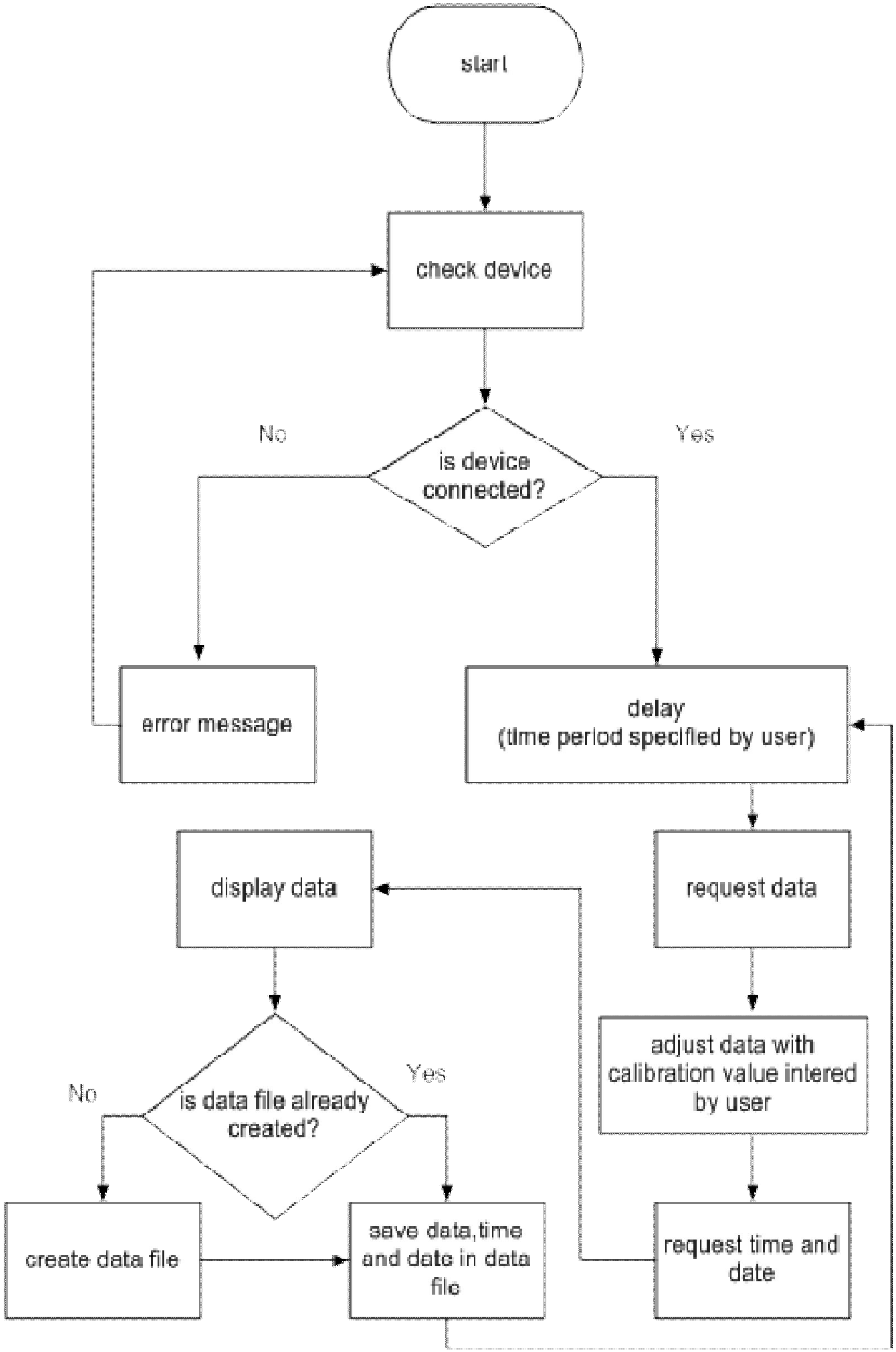


Figure 3.18 Flow Chart User Interface Main Program

3.4.3 Processing Explanation of the Main Program

There are four main parts of main program in microcontroller. There is initialization of ports, timer, setup of serial port, check situation of reed sensor (high/low).

a) Initialization of the mode of ports A ,B

In this project pin 9 of port B used as digital input where it receive input (HI/LO) from flip-flop pin 1, a register named Rain Register has been defined as rainfall counter for pulses received from RB3. Each time the input changes state (goes from low to high or high to low), the PIC needs to increment its rainfall counter. All pins of port B are set to input.

b) Timer modules

The timer module has three functions:

- Produce a time delay by means of a counter
- Count a number of input pulses arriving
- Produce an output pulse of a specified duration

Fig 16 below illustrates the basic arrangement of the PICs timer modules.

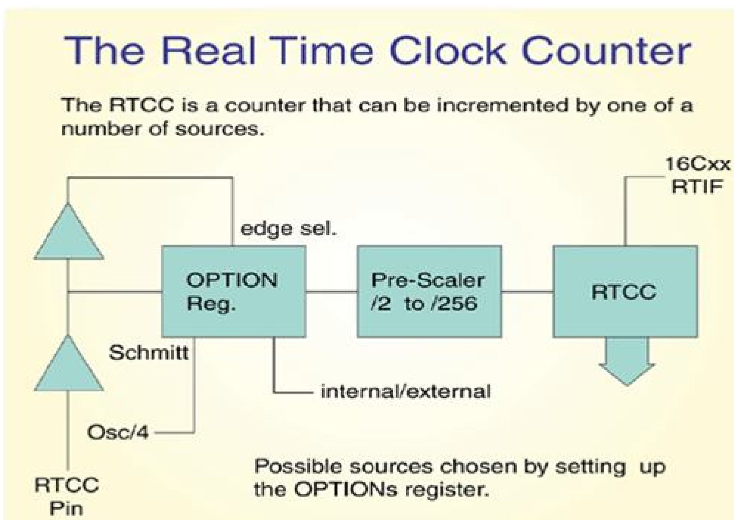


FIG 3.19 Timer Module

The timer module is called a “Real Time Clock counter” because it can make use of clock signal of the PIC to increment an 8 bit counter in real time.

In the PIC, the counter can be incremented on every clock pulse, or the clock signal can be gated into a pre-scalar unit so that the clock signal is divided to lower frequency. In this way, the timer can be made to count more slowly and hence produce longer delays. The timer register is called the TMR register referred to as TMRO in the PIC16c84. (There is only one timer in this particular PIC, but some PICs have more than one).

The TMR register can be loaded with a count value via software. It is then incremented to the maximum value 11111111 (255 decimal, FF hex) before rolling over to zero, 0000 000. The action of roll-over to zero causes a flag bit to be set in the INTCON register (bit 2, called TOIF (timer over flow interrupt flag)). The microcontroller can read the status of the INTCON register TOIF bit to determine when the timer counter has rolled over. The OPTION register can be used to select the source for the clock pulses going to the timer counter RTCC. See diagram above and table below. Osc/4 is the clock input signal, running at one quarter the crystal frequency used by the PIC. Therefore a PIC using a 4MHz clock will count at 1MHz rate.

Table 3.1 Option Register Bits (associated with RTCC)

Bits	7	6	5	4	3	2	1	0	
			RTS	RTE	PSA	PS2	PS1	PS0	
RTCC					0	0	0	0	2
WTD					1	0	0	1	4
Edg0>1				0		0	1	0	8
Edg1>0				1		0	1	1	16
internal			0			1	0	0	32
external			1			1	0	1	64
						1	1	0	128
						1	1	1	256
RTS, signal source, RTE signal active edge, PSA, pre-scalar assignment									

Bit 7 and bit 6 not associated with the control of the timer so they are not described.

Bit 5, the RTS bit, defines whether the timer clock source is internal (from the clock signal) or external

Bit 4 defines whether the rising edge or the falling edge will be used to increment the timer counter.

Bit 3 defines the source for the pre-scalar - either the timer counter or the Watchdog timer.

Bits 2, 1 and 0 determine the size of the pre-scalar in terms of 'divide by 2' up to 'divide by 256', as shown.

(c) Setup for serial port

PIC16F84A microcontroller doesn't have built in UART (Universal Asynchronous Receiver Transmitter) module, so we can create UART functionality in its software, so software code written in assembly language.

First external frequency (1MHZ) is defined, Baud rate for UART (6900) UART _TX is Ra1, and UART_RX is Rb0.

In the main function, firstly UART is initialized using initialize soft UART function, and then set the pins RT be an output and RX to be an input, TX is made high, because idle state of UART TX pin is always high.

A code for UART receive function is written, code waits for the starting bit, then after a delay of one and half bit, first bit value is sampled. Then after every 1 bit duration next data bit is sampled and its value is stored in data value variable.

In the end, stop bit is checked; if valid stop bit is found then received value in the data value variable is returned from the function. If valid stop bit is not found, then a value of 0 is returned from the function indicating an error has occurred in the reception.

At last code for the UART Transmit function, in which after every one bit duration, TX pin is made low or high depending upon the value of data value variable. Each bit of data value variable is transmitted after every one bit delay. In the end, stop bit is transmitted, Data is transmitted LSB first.

c) Check Reed Switch Situation

After each bucket tip, a magnet attached to bucket affected reed sensor cause its contacts to open and close, that mean high and low digital input reach microcontroller through pin RB3.

Two commands are used to test reed switch situation (high or low):

i) **BTFSC**: Bit test F, skip if clear (0), switch closed that mean test register bit and skip next instruction if register bit =0 or clear.

ii) **BTFSS**: Bit test F, skip if set (1), switch open that mean test register bit and skip next instruction if register bit =1 or set.

CHAPTER FOUR

EXPERIMENTAL RESULTS ANALYSIS AND COST

4.1 Introduction

Because the rain gauges of tipping bucket can easily use the digital signal, the rain gauges are widely used for the meteorological observation. In general the resolution of rain gauges of tipping bucket type can be categorized by 0.1mm, 0.5mm and 1mm classes. But the error of the tipping bucket rain gauges is made by the intensity of rainfall and expected to make the standard calibration method for error measurement. But it is not easy to obtain the exact error factor in accordance with rainfall intensity, because the inertia of tipping bucket varies according to rainfall intensity, the rainfall loss breaks out.

World meteorological organization recommends the standard for the measurement methods and equipments of rain gauges used to measure rainfall among equipments for meteorological observation. Thus, this study developed the calibration facility to measure error characteristic of rain gauge by rainfall intensity and standardized the procedure to upgrade the accuracy of calibration.

Using available facilities, this study presents a fitting rain gauge by test result and provides a compensation value to improve the data quality of rainfall.

4.2 Calibration

Calibration helps with obtainment of high quality rain measurements of tipping bucket rain gauge which are the data sources for meteorological / hydrologic research/ disaster mitigation. It is recommended to do calibration for the following statuses:

(i) Maintenance on a regular basis: Annual

(ii) After high intensity rainfall event

(iii) In doubt about the accuracy of collected rain data

4.2.1 How to Calibrate

OPTION	ACTION REASON	COST	ISSUE
No **TBRG** Calibration	No request	None	Either a higher cost or long period with no valued data
Manual calibration	With measuring cup pour of water into the funnel and manual count the tip which allow of larger error range	Few	Under large uncertainty
Factory calibration or laboratory calibration	Wait for a schedule and the calibration is performed off-site **with extra delivery cost**	High	No idea whether the error will be derived from installation
Calibration on site with a portable field calibrator	Pour a known volume of water and the water will drip at constant flow rate from the bottle into the rain gauge.	low	Desired tip of test vs. actual counts

Table 4.2.1 Calibration Methods

4.2.2 Gauge Testing and Calibration:

Figure 4.1 gauge practical Electronic circuit

As shown in Figure 4.1 of gauge practical electronic circuit contained in metallic case, the input is connected to reed switch and output is connected to PC through USB-to serial adapter, the circuit is supplied with 9v DC.

Figure 4.2 Gauge Unit Body

As shown in Figure 4.2 of gauge unit body which consist of

(1) PVC T-Junction

(2) Sensor unit: this consist of

(i) reed switch sensor to detect the movement of tipping bucket

(ii) tipping bucket mechanism which consist of two buckets fixed to movable axis on two ball bearing and two adjusting screw (to adjust the balance of two bucket)

Calibration was made on site by pouring a determined volume of water (100 mm) per specific time (50 sec) in rain gauge funnel, each bucket contain approximately 10 mm of water when tipped water. The readings result of calibration test is scheduled below:

TIP(mm)	TIME (sec)
10	5
20.1	10
30.1	15
40.2	20
50	25
50.2	30
59.8	35
70.1	40
80.3	45
90	50
100.1	55

Table 4.2.2 Gauge Testing Results

4.3 Gauge Recording Limitations

To get the best results of recording rainfall some procedures must be taken in consideration:

- The rain gauge must be positioned on a flat surface – if the surface is not flat, the see saw may tip before the bucket has filled to the calibrated level, or not tip at all. If the bucket does not tip at the calibrated level, the rainfall calculated will not be correct. Use a spirit level to determine whether a surface is flat and then fix the gauge to the flat surface to ensure you are getting an accurate reading.
- The rain gauge must be positioned on a surface that does not vibrate- surface such as porch or fence can move and vibrate. The tipping bucket gauge is very sensitive and any vibration could cause the gauge to tip even if it is not raining.
- The gauge must not be positioned near trees – being positioned near trees could allow leaves or pollen to fall inside funnel and blocked it, causing an inaccurate reading.
- The gauge must not be positioned in a sheltered area- being positioned in a sheltered location (such as beside your house or a fence) could significantly decrease or increase the amount of rain depending on the wind direction, and cause an inaccurate reading.

4.4 Gauge Overall Cost

The price cost of tipping bucket rain gauge which includes mechanical and electrical hardware is listed below:

DESCRIPTION	PRICE(SDG)
Flip-flop	10
capacitors	15
Crystal 4MHZ	10
Resistors	15
Max232	30
Press button	5
Board test	25
PCB board	12
DB9 connector	17
PIC microcontroller	80
Reed switch	20
Timer555	20
Tipping bucket unit	50
Gauge body unit and funnel	25
Hardener Resin	15
TOTOAL COST	**349 SDG**

Table 4.3 Gauge Components Total Cost

The price of new tipping bucket rain gauge in the market is found about $480 which is approximately equal to 4800 SDG, and the ours above is only about 349SDG.

By making comparison the designed gauge has a lower price cost.

CHAPTER FIVE

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Recent developments in science and technology provide wide range scope of applications of high performance and accurate rain gauge in area such as air ports, agriculture, flood, water drinking stations, climate researches changes, transport, etc...

This dissertation concentrates on recording type which is tipping bucket rain gauge rather than the second type which is the non- recording.

The digital tipping bucket rain gauge design is accomplished using microcontroller PIC16F84A, the electromechanical system in which the movement action of tip as a result of precipitation water weight is converted into electrical signal, the magnet sensor open or close due to effect of magnet piece attached in axis of tipping bucket. The on/off signal transfer to microcontroller and then to PC through USB to serial communication finally display rain precipitation in mm.

In this dissertation hardware and software design was implemented and tested.

Hardware design includes design of gauge sensor unit (tip, axis, supports, and base). While the software design includes microcontroller software and rainfall display software on PC.

5.2 Recommendation

Although the microcontroller can function as expected, and since the performance of the rain gauge deeply depend on it is accuracy, the most accurate results from a tipping bucket rain gauge need proper installation of rain gauge.

So finding the best possible site for the gauge is important and careful consideration should be given to the quality of precipitation catch prior to the final installation.

For future works, some recommendations have been listed below in order to improve the performance.

- The study can be expanded to make weather station by adding temperature, humidity, wind speed and wind direction as input values to the PIC microcontroller.
- The collected data after saving and storing in PC can be share with others by using internet. This could help to compare the accuracy of collected data by data that collected by anther stations in the same area, so this could help to fix the reading errors.
- Using PIC 18F family help to measure weather pressure, because pressure sensor produce analog output signal. The PIC18 has ability to convert the signal from analog to digital, this feature not available for PIC16F family.

REFERENCES

[1] H.M.Raghunath, “Hydrology Principle® Analysis ®Design”

, 2nd edition, India, Newdelhi, 2006.

[2] Dr. John Gorman, ARIC**,”** Introduction to the Tipping Bucket Rain Gauge”, USA, 2011

[3] Stephen Burt, “The Weather Observer’s Hand Book

UK. Cambridge”,2013.

[4] Lan.R.Sinclair,”Sensor and Transducer”, 3rd edition, UK,

Oxford, 2001.

[5] Judy Bass, “Electronic Circuit for the Evil Genius”, USA,

MCGRAW- HILL,2005.

Appendix A

”””
””””””””

Program in Microcontroller PIC16F84A for Tipping Bucket Rain Gauge

””
”””””””””””””””””””””””””””””””

””
”””””””””””””””””””””””””””””””

File Name: ptr.asm

Project : Design of Tipping Bucket Rain Gauge Using Microcontroller PIC16F84A

Date : 12/11/2013

”””
”””””;””””””””””””””””””””””””””””””””

```
        ; list =        16f84a
         Radix        dec                                                   ;
; __config _CP_OFF & _WDT_OFF & _XT_OSC
  _Clkspd             equ    1000000
  ; _BaudRate equ     9600
   _Period            equ    (_Clkspd/_BaudRate)
      Macros
```

```
loadw   macro arg1
movlw   high(arg1)
movwf   hi
movlw   low(arg1)

movwf   lo
endm
dodigit macro   arg1
movlw   high(arg1)
movwf   shi
movlw   low(arg1)
movwf   slo
call    dosub
endm
, memory locations:
cblock 0x0C
BitCount        ;number of bits left to send or
                ;receive, not including start & stop
                ;bits
RXChar          ;received character while being received
```

```
RXBuff          ;most recently received character
TXChar          ;character to transmit
SerialReg   ;status register:
            ;bit 0: on if character has been
            ;    received
            ;bit 1: on if busy with RX/TX
            ;bit 2: on if sending, off if receiving
            ;bit 3: on if next bit to send is stop bit
  WSave         ;copy of W register
  SSave         ;copy of the Status register
    SHT11Byte   ;a byte of data sent to or returned by the SHT11
  counter       ;generic counter register
  counter2  ;another generic counter register
  hi        ;hi byte for number to be converted to ascii
  lo        ;lo byte for number to be converted to ascii
  shi       ;hi byte for subtractor for conversion to ascii
  slo       ;lo byte for subtractor for conversion to ascii
  digit     ;ascii digit converted from binary. Also used as
            ;input to SendErrorCode
  MSDelay       ;register for WaitMS timing
```

```
RegSave          ;copy of PORTA or PORTB register
RegTemp          ;temp copy of RegSave
RainHi           ;hi byte of two-byte rainfall register
RainLo           ;lo byte of two-byte rainfall register
```

,,
,,,

-Serial Communication Routines-

The serial comm routines generate 1 start bit, 8 data bits, 1 stop bit,

; no parity. The baud rate is determined by the delay programmed

; into the onboard timer. The sending and receiving is interrupt driven,

; meaning other tasks can be carried on while the characters are being

; sent and received.

```
; -----Main Interrupt Routine----------------------------------------
Int         movwf       WSave
swapf STATUS,W  ;use swapf to prevent any
movwf     SSave      ;status flags from being changed
 btfsc  INTCON,T0IE
goto   DoBit
wapf   SSave,W
movwf     STATUS
```

```
swapf WSave,F
swapf WSave,W
retfie
        ;------end Main Interrupt Routine-----------------------------------
…………………………………………………………………………………………………
  ……………………………………………………………………………………
        ;------Subroutine StartRX------------------------------------------
bcf     INTCON,INTF
movlw       _StartRxDelay
movwf       BitCount
RXWait      decfsz BitCount,F
goto    RXWait
movlw
movwf       BitCount
        ;------end StartRX------------------------------------------------
        ;------Subroutine Send Char-----------------------------------------
bcf     PORTA,_SER_OUT
movlw       b'00000110'
movwf       SerialReg
```

```
movlwb'10100000'
movwf       INTCON
movlw
movwf       BitCount
return
        ;------begin Record Rainfall-----------------------------------------
RecordRainfall
;did a transition occur?
movf  RegSave,W
andlw _RAIN_MASK
movwf       RegTemp
movf  PORTB,W
andlw _RAIN_MASK
subwf RegTemp,F
btfsc  STATUS,Z
return
;it's set. increment the counter
incfsz RainLo,F
goto   SaveRainReg
incf   RainHi,F
```

```
SaveRainReg
 btfss  RegSave,_RAINSENSOR
 goto   SetRainReg
 bcf    RegSave,_RAINSENSOR
 return
SetRainReg
 bsf    RegSave,_RAINSENSOR
 return
            ;------end RecordRainfall-------------------------------------------
            ;------begin TellRainfall-------------------------------------------
 ; Send the contents of the rainfall register over the serial port
 ; as a five-digit ASCII number.
 TellRainfall
 movf   RainLo,W
 movwf      lo
 movf   RainHi,W
 movwf      hi
 ; Send the terminating CR and LF:
 call   SendCRLF
```

```
goto    MainLoop

        ;------end TellRainfall---------------------------------------------

        ;------begin Reset Rainfall------------------------------------------

; Reset the rainfall register.

Reset Rainfall

clrf    Rain Hi

clrf    Rain Lo

goto    Main Loop

                ;------end Reset Rainfall---------


        ;------begin Idle---------------------------------------------------

call    WaitMS

call    RecordRainfall

btfss   PORTB,_PBUTTON

return

        ;------end Idle-----------------------------------------------------

        ;------Main Program-------------------------------------------------

bsf     STATUS,RP0      ;switch to bank 1

movlw   b'10011111'     ;
```

```
movwf       TRISB
movlw       0x00
movwf       TRISA
bcf             STATUS,RP0
clrf    PORTA
zero the rainfall counter:
    clrf      RainHi
    clrf      RainLo
                    initialize RegSave for counting rainfall
 movf PORTB,W
  movwf     RegSave
 call   SerSetup      ;set up serial comm routines & int
MainLoop
 call   GetAChar      ;wait for a character
movf  RXBuff,W    ;move the rx char into W
 sublw      'r'            ;compare with 'r' character
 btfsc  STATUS,Z
 goto    TellRainfall ;if r, report the rainfall
 goto  MainLoop
 End        ,,,
```

Appendix B

””

Source code of C visual for tipping bucket rain gauge using PIC16F84A

””

```
    public void Starts()
  {
   // Closing serial port if it is open
    if (_serialPort != null && _serialPort.IsOpen)
       _serialPort.Close();
     _serialPort.DataReceived +=
       new SerialDataReceivedEventHandler(_serialPort_DataReceived);
   _  serialPort.Open();
   }
     void _serialPort_DataReceived(object sender, SerialDataReceivedEventArgs e)
   {
     int dataLength = _serialPort.BytesToRead;
     byte[] data = new byte[dataLength];
     int DataRead = _serialPort.Read(data, 0, dataLength);
```

```
  if (DataRead == 0)
    return;
  if (NewSerialDataRecieved != null)
    NewSerialDataRecieved(here, SerialData(data));
 }
 public void Stops()
{
 _serialPort.Close();
}
```