Design & Development of a Microcontroller Based Water Flow Control System Using Servo Motor

Developed by:

Md. Tamzid Islam

ID: 2011-2-55-038

Rajeeb Chandra Talukder

ID: 2011-2-55-012

East West University

Department of Electronics & Communication Engineering

Project Supervisor:

Dr. Saeed Mahmud Ullah

Associate Professor

Department of Applied Physic, Electrical & Electronic Engineering

University Of Dhaka

DECLARATION

We therefore pronounce that we did the work reported in this project in the **Department of Electronics and Communication Engineering, East West University, under the supervision** of **Dr. Saeed Mahmud Ulllah**. We gravely announce that to the best of our insight, no some portion of this report has been submitted somewhere else for grant of a degree. All wellsprings of information utilized have been properly recognized.

Signature:

....	....
Md. Tamzid Islam	**Rajeeb Chandra Talukder**
ID : 2011-2-55-038	**ID : 2011-2-55-012**

Supervisor

Dr. Saeed Mahmud Ullah

Associate Professor

Department of Applied Physic, Electrical & Electronic Engineering

University Of Dhaka

APPROVAL

This is to certify that the Project titled as "**Design & Development of a Microcontroller Based Water Flow Control System Using Servo Motor**" submitted to the respected members of the Board of Examiners of the Faculty of Engineering for partial fulfillment of the requirements for the degree of Bachelor of Science in Electronics & Telecommunications Engineering by the following students and has been accepted as satisfactory.

Submitted By:

Md. Tamzid Islam
ID : 2011-2-55-038

Rajeeb Chandra Talukder
ID : 2011-2-55-012

Dr. Saeed Mahmud Ullah
Associate Professor
Department of Applied Physic, Electrical & Electronic Engineering

University Of Dhaka

Dr. M. Mofazzal Hossain
Chairperson & Professor,
Dept. of Electronics and Communication Engineering
East West University

ACKNOWLEDGEMENTS

Numerous individuals merit our debt of gratitude is in order for their assistance in finishing this project. We might want to thank our area of expertise for allowing us to do this project. We need to express our thanks and profound thankfulness to our counselor **Dr. Saeed Mahmud Ulllah** as he has depleted all his knowledge and time by catching up our day by day advance and empowering advices and even by sharing on the inconveniences. We might want to extend our gratitude to the research center staff of ECE Dept. for their quick reaction and participation with us to get a few materials we requirement for our case. We have uncommon affirmation for our gathering individuals for their seeing each other and dedicated from the earliest starting point up to the end.

At long last, we might want to thank our family and all persons who were included with this project for their significant help and polished skill amid this anticipate.

Dedication

To Our Parents

ABSTRACT

Water flow control is one of the significant issues confronting real urban communities of the world and wastage during transmission has been distinguished as a major issue; this is one of the motivations for this research, to deploy computing techniques in creating a barrier to wastage in order to not only provide more financial gains and energy saving, but also help the environment and water cycle which in turn ensures that we save water for our future. We introduced our exploration in installing a control system into a programmed water pump controller through the use of different technologies in its design, development, and implementation. The system utilized microcontroller to automate the procedure of water pumping in a water canal or other river and can distinguish the level of water, switch on/off the pump as needs be and show the status on a LCD screen. This research has successfully given a change on existing water level controllers by its utilization of calibrated circuit to demonstrate the water level.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Water flow control

Sustainability of available water resource in many reason of the word is now a dominant issue. This problem is quietly related to poor water allocation, inefficient use, and lack of adequate and integrated water management. Water is commonly used for agriculture, industry, and domestic consumption. Therefore, efficient use and water monitoring are potential constraint for home or office water management system. Last few decades several monitoring system integrated with water level detection have become accepted. Measuring water level is an essential task for government and residence perspective. In this way, it would be possible to track the actual implementation of such initiatives with integration of various controlling activities. Therefore, water controlling system implementation makes potential significance in home applications. The existing automated method of level detection is described and that can be used to make a device on/off. Moreover, the common method of level control for home appliance is simply to start the feed pump at a low level and allow it to run until a higher water level is reached in the water tank. This is not properly supported for adequate controlling system. Besides this, liquid level control systems are widely used for monitoring of liquid levels, reservoirs, silos, and dams etc. Usually, this kind of systems provides visual multi level as well as continuous level indication. Audio visual alarms at desired levels and automatic control of pumps based on user's requirements can be included in this management system. Proper monitoring is needed to ensure water sustainability is actually being reached, with disbursement linked to sensing and automation. Such programmatic approach entails microcontroller based automated water level sensing and controlling.

1.2 Surface and Ground Water Resources

Water in our planet is available in the atmosphere, the oceans, on land and within the soil and fractured rock of the earth's crust Water molecules from one location to another are driven by the solar energy. Moisture circulates from the earth into the atmosphere through evaporation and then back into the earth as precipitation. In going through this process, called the Hydrologic Cycle.

1.3 Priorities for water resources planning

Water resource projects are constructed to develop or manage the available water resources for different purposes. The water allocation priorities for planning and operation of water resource systems should broadly be as follows:

1. **Domestic consumption**

   This includes water requirements primarily for drinking, cooking, bathing, washing of clothes and utensils and flushing of toilets.

2. **Irrigation**

   Water required for growing crops in a systematic and scientific manner in areas even with deficit rainfall.

3. **Hydropower**

   This is the generation of electricity by harnessing the power of flowing water.

4. **Ecology / environment restoration**

   Water required for maintaining the environmental health of a region.

5. **Industries**

   The industries require water for various purposes and that by thermal power stations is quite high.

6. **Navigation**

   Navigation possibility in rivers may be enhanced by increasing the flow, thereby increasing the depth of water required to allow larger vessels to pass.

7. **Other uses**

   Like entertainment of scenic natural view.

1.4 Purpose of this project

In this project we show how to make a water flow controller with microcontroller and servomotor which will use as a switch gate. The servo motor in this project is use to control water flow and the sensor is use to measure the water level. By this project we can control the waste of water and by using it on the switch gate we can prevent flood or any other water related problems. This project can be used in industry as well as household. This project has two functions, one is water measure and another is flow control. By combine this two properties we can control water stages in water treatment plant, switch gates of small river, canal or lake. So this control system gives safe secured water flow control. In short the purpose of the project is

- To design a smart water flow control system
- To construct a smart water flow controller circuit in the breadboard
- Test for its functionality
- Product commercialization
- To design the control system with low cost components

Chapter 2

AN OVERVIEW OF MICROCONTROLLERS

2.1 Introduction

Arduino is an open source electronics platform based on easy-to-use hardware and software. In this project we use Arduino-UNO board. The Arduino hardware platform already has the power and reset circuitry setup as well as circuitry to program and communicate with the microcontroller over USB. In addition, the I/O pins of the microcontroller are typically already fed out to sockets/headers for easy access. On the software side, Arduino provides a number of libraries to make programming the microcontroller easier. The simplest of these are functions to control and read the I/O pins rather than having to fiddle with the bus/bit masks normally used to interface with the At mega I/O.

2.2 Microcontroller

A microcontroller is a small computer (SoC) on a solitary integrated circuit containing a processor core, memory, and programmable input/output peripherals. Program memory as Ferroelectric RAM, NOR streak or OTP ROM is regularly included on chip, and also an ordinarily little measure of RAM. Microcontrollers are intended for installed applications, rather than the microchips utilized as a part of PCs or other universally useful applications comprising of different discrete chips. Microcontrollers are utilized as a part of naturally controlled items and gadgets, for example, vehicles motor control frameworks, implantable medicinal gadgets, remote controls, office machines, apparatuses, power instruments, toys and other installed frameworks. By diminishing the size and cost contrasted with a configuration that uses a different microchip, memory, and information/yield gadgets, microcontrollers make it sparing to digitally control significantly more gadgets and procedures. Blended sign microcontrollers are regular, incorporating simple parts expected to control non-advanced electronic frameworks. Some microcontrollers may utilize four-piece words and work at clock rate frequencies as low as 4 kHz, for low power utilization. They will by and large can hold usefulness while sitting tight for an occasion, for example, a catch press or other intrude on; force utilization while resting (CPU clock and most peripherals off) might be just nanowatts, making a significant number of them appropriate for dependable battery applications. Different microcontrollers may serve execution basic parts, where they may need to act more like an advanced sign processor (DSP), with higher clock speeds and power utilization.

2.3 History

The first microprocessor was the 4-bit Intel 4004 released in 1971, with the Intel 8008 and other more capable microprocessors becoming available over the next several years. However, both processors required external chips to implement a working system, raising total system cost, and making it impossible to economically computerize eappliances.The Smithsonian Institution credits TI engineers Gary Boone and Michael Cochran with the successful creation of the first microcontroller in 1971. The result of their work was the TMS 1000, which became commercially available in 1974. It combined read-only memory, read/write memory, processor and clock on one chip and was targeted at embedded systems. Partly in response to the existence of the single-chip TMS 1000, Intel developed a computer system on a chip optimized for control applications, the Intel 8048, with commercial parts first shipping in 1977. It combined RAM and ROM on the same chip. This chip would find its way into over one billion PC keyboards, and other numerous applications. At that time Intel's President, Luke J. Valenter, stated that the microcontroller was one of the most successful in the company's history, and expanded the division's budget over 25%. Most microcontrollers at this time had concurrent variants. One had an erasable EPROM program memory, with a transparent quartz window in the lid of the package to allow it to be erased by exposure to ultraviolet light, often used for prototyping. The other was either a mask programmed ROM from the manufacturer for large series, or a PROM variant which was only programmable once; sometimes this was signified with the designation OTP, standing for "one-time programmable". The PROM was of identical type of memory as the EPROM, but because there was no way to expose it to ultraviolet light, it could not be erased. The erasable versions required ceramic packages with quartz windows, making them significantly more expensive than the OTP versions, which could be made in lower-cost opaque plastic packages. For the erasable variants, quartz was required, instead of less expensive glass, for its transparency to ultraviolet—glass is largely opaque to UV—but the main cost differentiator was the ceramic package itself. In 1993, the introduction of EEPROM memory allowed microcontrollers to be electrically erased quickly without an expensive package as required for EPROM, allowing both rapid prototyping, and In System Programming. (EEPROM technology had been available prior to this time, but the earlier EEPROM was more expensive and less durable, making it unsuitable for low-cost mass-produced microcontrollers.) The same year, Atmel introduced the first microcontroller using Flash memory, a special type of EEPROM. Other companies rapidly followed suit, with both memory types.

2.4 Advantages of Microcontroller

- The simplest computer processor is used as the "brain" of the future system.
- Single purposes and low power consumption.
- Depending on the taste of the manufacturer, a bit of memory, a few A/D converters,
- timers, input/output lines etc. are added.
- Low cost and small packaging.
- Simple software able to control it all and which everyone can easily learn about has been
- developed.

2.5 Embedded Design

A microcontroller can be viewed as an independent system with a processor, memory and peripherals and can be utilized as an implanted system. The greater part of microcontrollers being used today are installed in other hardware, for example, cars, phones, machines, and peripherals for PC frameworks. While some inserted system are exceptionally modern, numerous have negligible prerequisites for memory and project length, with no working system, and low programming multifaceted nature. Run of the mill information and yield gadgets incorporate switches, transfers, solenoids, LEDs, little or custom fluid precious stone presentations, radio recurrence gadgets, and sensors for information, for example, temperature, mugginess, light level and so on. Inserted frameworks as a rule have no console, screen, plates, printers, or other conspicuous I/O gadgets of a PC, and may need human cooperation gadgets of any sort.

2.6 Interrupts

Microcontrollers must provide real-time (predictable, though not necessarily fast) response to events in the embedded system they are controlling. When certain events occur, an interrupt system can signal the processor to suspend processing the current instruction sequence and to begin an interrupt service routine (ISR, or "interrupt handler") which will perform any processing required based on the source of the interrupt, before returning to the original instruction sequence. Possible interrupt sources are device dependent, and often include events such as an internal timer overflow, completing an analog to digital conversion, a logic level change on an input such as from a button being pressed, and data received on a communication link. Where power consumption is important as in batteried devices, interrupts may also wake a microcontroller from a low-power sleep state where the processor is halted until required to do something by a peripheral event.

2.7 Types of microcontroller

- ARM core processors (many vendors)
- ARM Cortex-M cores are specifically targeted towards microcontroller applications
- Atmel AVR (8-bit), AVR32 (32-bit), and AT91SAM (32-bit)
- Cypress Semiconductor's M8C Core used in their PSoC (Programmable System-on-Chip)
- Freescale ColdFire (32-bit) and S08 (8-bit)
- Freescale 68HC11 (8-bit), and others based on the Motorola 6800 family
- Intel 8051, also manufactured by NXP Semiconductors, Infineon and many others
- Infineon: 8-bit XC800, 16-bit XE166, 32-bit XMC4000 (ARM based Cortex M4F), 32-bit TriCore and, 32-bit Aurix Tricore Bit microcontrollers[18]
- MIPS
- Arduino
- Microchip Technology PIC, (8-bit PIC16, PIC18, 16-bit dsPIC33 / PIC24), (32-bit PIC32)
- NXP Semiconductors LPC1000, LPC2000, LPC3000, LPC4000 (32-bit), LPC900, LPC700 (8-bit)
- Parallax Propeller
- PowerPC ISE
- Rabbit 2000 (8-bit)
- Renesas Electronics: RL78 16-bit MCU; RX 32-bit MCU; SuperH; V850 32-bit MCU; H8; R8C 16-bit MCU
- Silicon Laboratories Pipelined 8-bit 8051 Microcontrollers and mixed-signal ARM-based 32-bit microcontrollers

2.8 Arduino-UNO

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Arduino consists of both a physical programmable circuit board and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board. The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware in order to load new code onto the board – simply use a USB cable. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package. The Arduino software is free, the hardware boards are pretty cheap, and both the software and hardware are easy to learn has led to a large community of users who have contributed code and released instructions for a huge variety of Arduino-based projects.

2.9 Figure of Arduino-UNO

2.10 History of Arduino-UNO

The Arduino is developed in 2005. The Arduino microcontroller was initially created as an educational platform for a class project at the Interaction Design Institute Ivrea in Milan (Italy) in 2005. It derived from a previous work of the Wiring microcontroller designed by Hernando Barragan in 2004. From the beginning, the Arduino board was developed to attract artists and designers. The Wiring microcontroller was created by Hernando Barragan to be used for parsing data to electronic devices. His aim was that it could be used by non-technical people who only had basic experience with using computers. He first of all wanted it to be used as a prototyping tool. Since he needed help to create an easy software tool to programmed the board he engaged Casey Reas and Massimo Banzi as his assistants. Reas created the visual programming language for the prototyping tool.

Chapter 3

THEORY BEHIND THE PROJECT

3.1 Introduction

This project includes Arduino Uno broad, servo motor, DC motor, 16 bit LED display, RJ45 connector. We want to connect the servo motor to an Arduino. We developed the connection by using upper equipments. At first we developed a connection without load. Then we developed the circuit with 220V power load.

3.2 Arduino-UNO

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter. "Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0.

3.3 Architecture of Arduino-UNO

There are many varieties of Arduino boards that can be used for different purposes. The Arduino UNO components are:

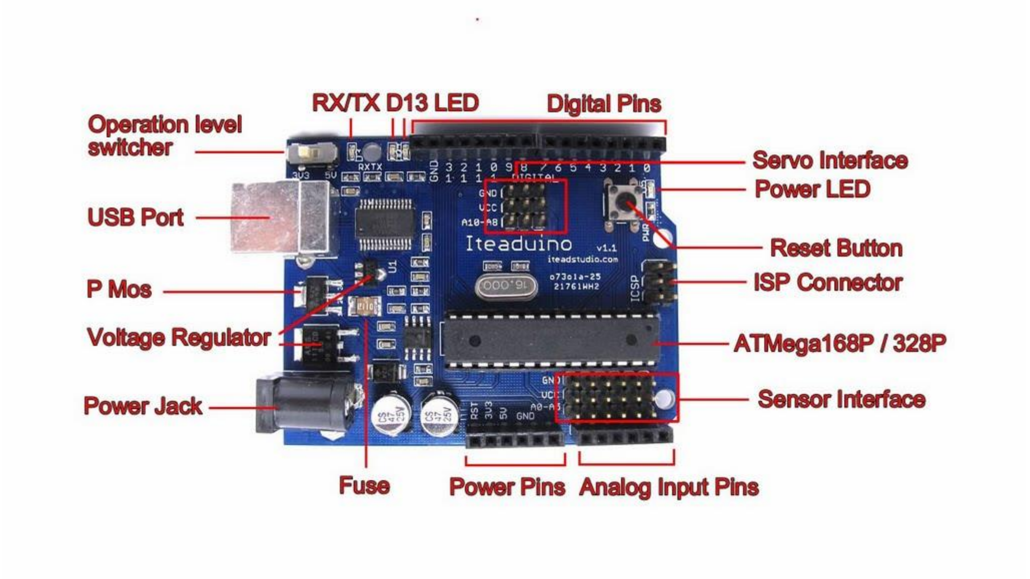


Fig: 3.1 Arduin-Uno R3 Board

3.4 Overview

The Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again."Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

3.4.1 Power

The Arduino/Genuino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector. The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.
The power pins are as follows:

- V_{in}. The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- 5V.This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- 3V3. A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- GND. Ground pins.

- IOREF. This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

3.4.2 Memory

The ATmega328 has 32 KB (with 0.5 KB occupied by the boot loader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

3.4.3 Input and output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller. In addition, some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attach Interrupt() function for details.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analog Write() function.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
- LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
- TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.
- The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function.

3.4.4 Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega8U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '8U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, an *.inf file is required.. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-toserial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A SoftwareSerial library allows for serial communication on any of the Uno's digital pins. The ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmega328 datasheet.

3.4.5 Reset Button

The Arduino has a reset button. Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if code doesn’t repeat, but we want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn’t usually fix any problems.

3.4.6 TX RX LEDs

TX is short for transmit, RX is short for receive. In our case, there are two places on the Arduino UNO where TX and RX appear once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs. These LEDs will give us some nice visual indications whenever Arduino is receiving or transmitting data.

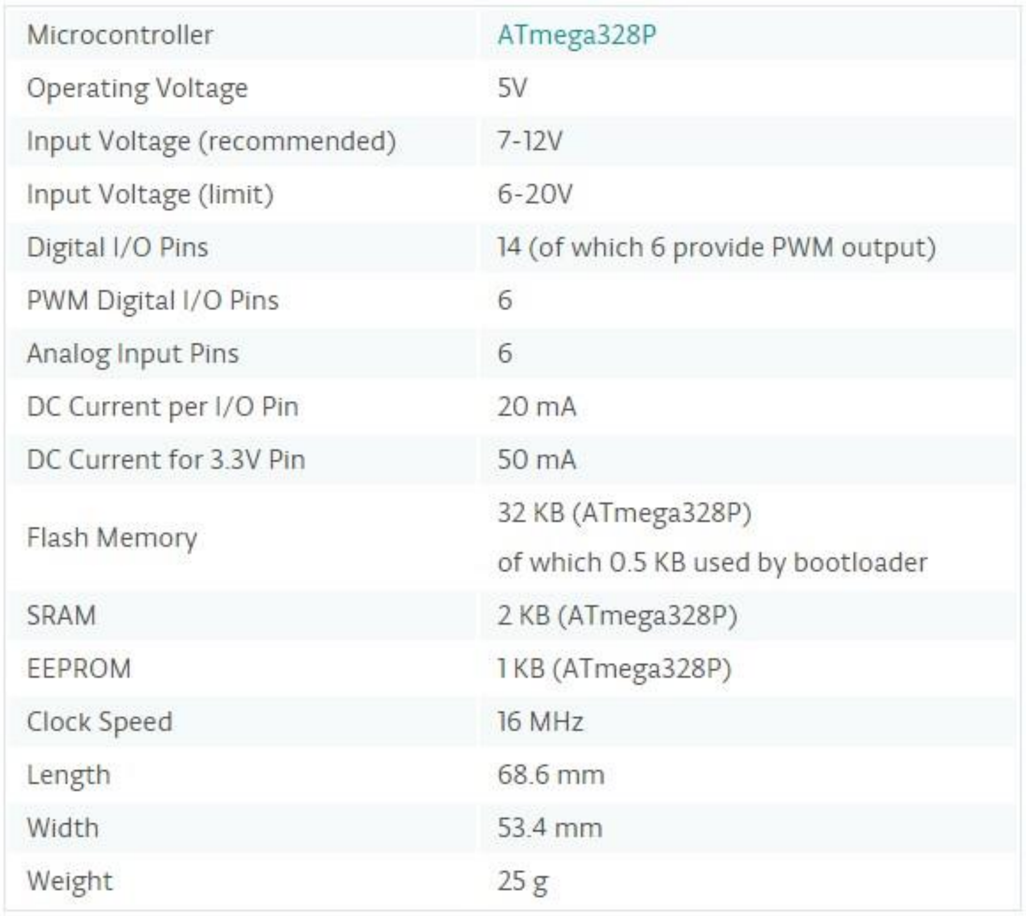
3. 4.7 Main IC

The black thing with all the metal legs is an IC, or Integrated Circuit. The main IC on the Arduino is slightly different from board type to board type, but is usually from the AT mega line of IC’s from the ATMEL Company. This can be important, as may need to know the IC type before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC.

3. 4.8 Voltage Regulator

The voltage regulator is not actually something interacting with on the Arduino. But it is potentially useful to know that it is there and what it's for. It controls the amount of voltage that is let into the Arduino board. It will turn away an extra voltage that might harm the circuit.

3.4.9 Technical specs



Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

3.4.10 Schematic Diagram

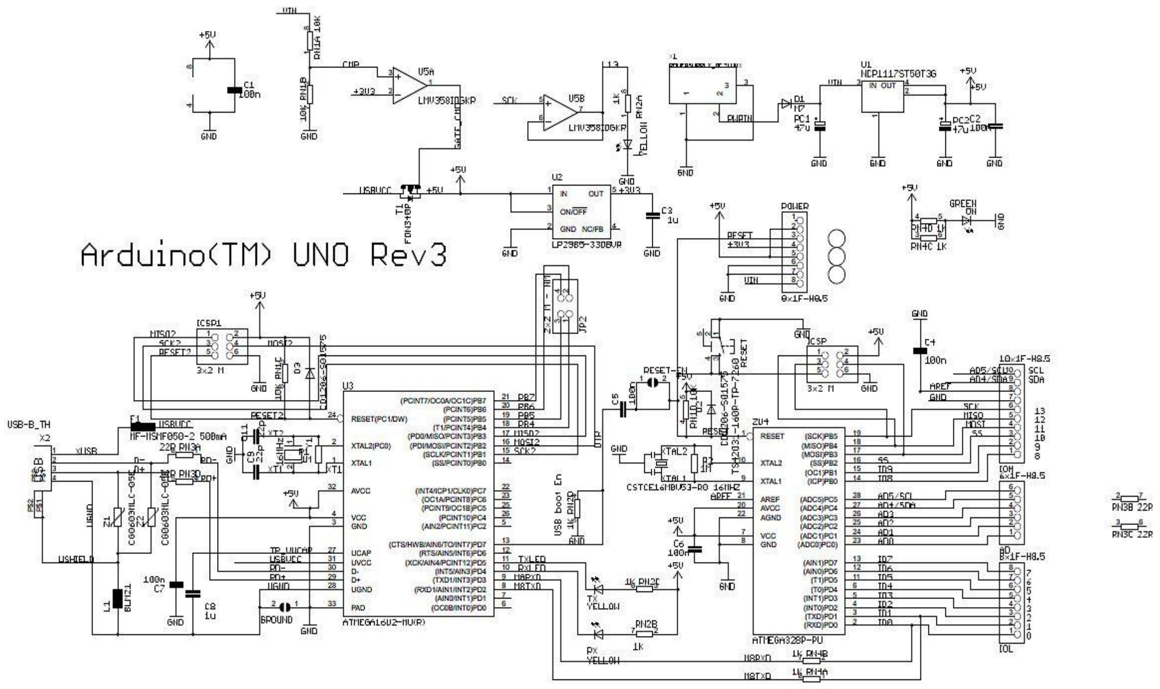


Fig: 3.2 Schematic Diagram Arduin-Uno

3.5 Getting started with Arduino Software

First download and install the Arduino for Mac, Linux or Windows from arduino.cc. Windows users also need to install a driver. Connect your board via USB, launch the Arduino application and select Arduino-Uno from the tools to board menu. Open the sketch File.
Examples: 01. Basics: Blink. Click the toolbar button to upload it to your board.

3.5.1 The Integrated Development Environment (IDE)

Every microcontroller needs software to be programmed. The Arduino board is not a case apart. It has its own integrated development environment (IDE).It is free and everyone can download it from its official website using either the Windows, Mac OS X or Linux platform. That allows Arduino Board to gain more users and it also helps it to grow.

3.5.2 IDE Parts

- Compile: Before program “code” can be sent to the board, it needs to be converted into instructions that the board understands. This process is called Compiling.
- Stop: This stops the compilation process.
- Open Existing Sketch: This loads a sketch from a file on our computer.
- Save Sketch: This saves the changes to the sketch.
- Upload to Board: This compiles and then transmits over the USB cable to our board.
- Serial Monitor: Until this point when our programs (sketches) didn’t work, we just pulled out our hair and tried harder.
- Tab Button: This lets you create multiple files in your sketch. This is for more advanced programming than we will do in this class.
- Text Console: This shows you what the IDE is currently doing and is also where error messages display if make a mistake in typing program.
- Line Number: This shows what line number your cursor is on.

3.6 Servomotor

Servo motors (or servos) are self-contained electric devices that rotate or push parts of a machine with great precision. Servos are found in many places: from toys to home electronics to cars and airplanes. If you have a radio-controlled model car, airplane, or helicopter, you are using at least a few servos. In a model car or aircraft, servos move levers back and forth to control steering or adjust wing surfaces. By rotating a shaft connected to the engine throttle, a servo regulates the speed of a fuel-powered car or aircraft. Servos also appear behind the scenes in devices we use every day.

Servomotor are used in applications such as robots, CNC machinery or automated manufacturing. The main objective of servo are given are given below:

- Design a control system
- Analyze the transient response
- Fine tune the feed back loop
- Determine the proper gear ratio for the desired speed or efficiency
- Build the mechanical sections
- Build the amplifier and motor driver
- Try to make it fit inside whatever you’re trying to control

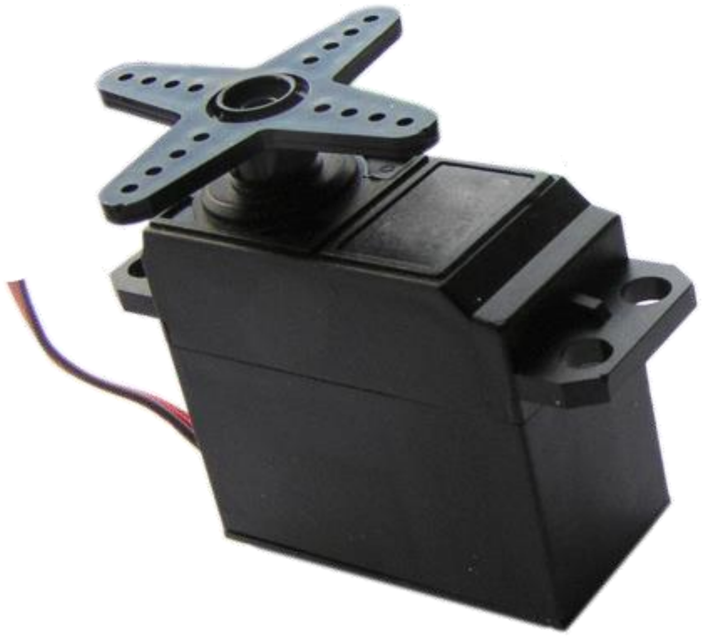


Fig 3.3 Servo Motor

3.6.1 Types of servo motors

Servos come in many sizes and in three basic types: positional rotation, continuous rotation, and linear.

- **Positional rotation servo**: This is the most common type of servo motor. The output shaft rotates in about half of a circle, or 180 degrees. It has physical stops placed in the gear mechanism to prevent turning beyond these limits to protect the rotational sensor. These common servos are found in radio-controlled cars and water- and aircraft, toys, robots, and many other applications.
- **Continuous rotation servo**: This is quite similar to the common positional rotation servo motor, except it can turn in either direction indefinitely. The control signal, rather than setting the static position of the servo, is interpreted as the direction and speed of rotation. The range of possible commands causes the servo to rotate clockwise or counterclockwise as desired, at varying speed, depending on the command signal.
- **Linear servo**: This is also like the positional rotation servo motor described above, but with additional gears (usually a **rack and pinion** mechanism) to change the output from circular to back-and-forth. These servos are not easy to find, but you can sometimes find them at hobby stores where they are used as actuators in larger model airplanes.

3.6.2 Mechanism

A servomotor is a closed-loop servomechanism that uses position feedback to control its motion and final position. The input to its control is some signal, either analogue or digital, representing the position commanded for the output shaft.The motor is paired with some type of encoder to provide position and speed feedback. In the simplest case, only the position is measured. The measured position of the output is compared to the command position, the external input to the controller. If the output position differs from that required, an error signal is generated which then causes the motor to rotate in either direction, as needed to bring the output shaft to the appropriate position. As the positions approach, the error signal reduces to zero and the motor stops. The very simplest servomotors use position-only sensing via a potentiometer and bang-bang control of their motor; the motor always rotates at full speed (or is stopped). This type of servomotor is not widely used in industrial motion control, but it forms the basis of the simple and cheap servos used for radio-controlled models. More sophisticated servomotors use optical rotary encoders to measure the speed of the output shaft and a variable-speed drive to control the motor speed. Both of these enhancements, usually in combination with a PID control algorithm, allow the servomotor to be brought to its commanded position more quickly and more precisely, with less overshooting.

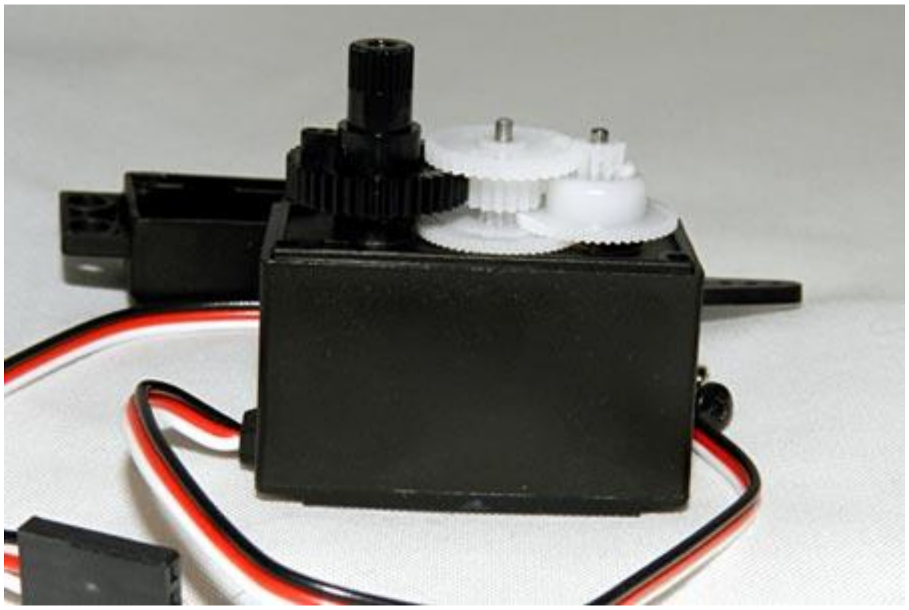


Fig 3.4 The gears in a typical standard-size servo are made of plastic.

3.6.3 Encoders

The first servomotors were developed with synchros as their encoders. Much work was done with these systems in the development of radar and anti-aircraft artillery during World War II. Simple servomotors may use resistive potentiometers as their position encoder. These are only used at the very simplest and cheapest level, and are in close competition with stepper motors. They suffer from wear and electrical noise in the potentiometer track. Although it would be possible to electrically differentiate their position signal to obtain a speed signal, PID controllers that can make use of such a speed signal generally warrant a more precise encoder. Modern servomotors use rotary encoders, either absolute or incremental. Absolute encoders can determine their position at power-on, but are more complicated and expensive. Incremental encoders are simpler, cheaper and work at faster speeds. Incremental systems, like stepper motors, often combine their inherent ability to measure intervals of rotation with a simple zero-position sensor to set their position at start-up.

3.6.4 Controlling a servo motor

Servos take commands from a series of pulses sent from the computer or radio. A pulse is a transition from low voltage to high voltage which stays high for a short time, and then returns to low. In battery devices such as servos, "low" is considered to be ground or 0 volts and "high" is the battery voltage. Servos tend to work in a range of 4.5 to 6 volts, so they are extremely hobbyist computer-friendly. Have you ever picked up one end of a rope that was tied to a tree or held one end of a jump rope while a friend held the other? Imagine that, while holding your end of the rope, you moved your arm up and down. The rope would make a big hump that would travel from your end to the other. What you have done is applied a pulse, and it traveled down the rope as a wave. As you raise your hand up and down, if you keep your hand in the air longer, someone watching this experiment from the side would see that the pulse in the rope would be longer or wider. If you bring your hand down sooner, the pulse is shorter or more narrow. This is the pulse width. If you keep your end going up and down, making a whole bunch of these pulses one after another, you have created a pulse train (see Figure 6 below). How often did you raise and lower your end? This is the frequency of your pulse train and is measured in pulses per second, or Hz (abbreviation of "hertz").

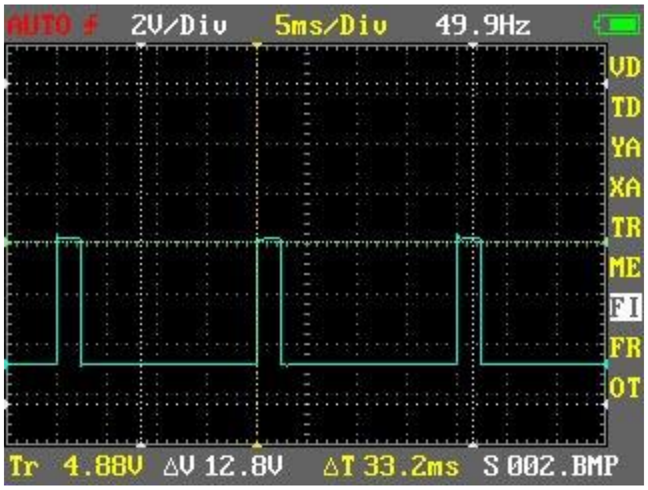


Fig 3.5 Pulse train Arduino might generate to control a servo

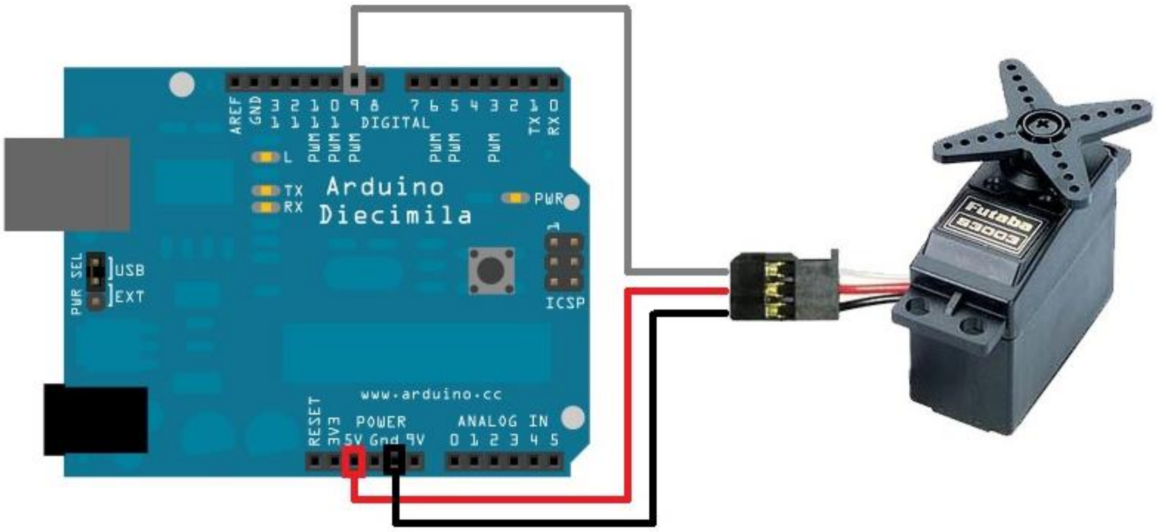


Fig 3.6 Servo motor and arduino connection

3.6.5 Specification of servo motor

Modulation	Digital
Torque	**4.8V:** 130.5 oz-in(9.40 kg-cm) **6.8V:** 152.8 oz-in(11.00 kg-cm)
Speed	**4.8V:** 0.19 sec/60̊ **6.8V:** 0.15 sec/60̊
Weight	1.94 oz(55.0g)
Dimensions	**Length:** 1.60 in (40.7 mm) **Width** 0.78 in (19.7 mm) **Height** 1.69 in (43.9 mm)
Motor Type	3- pole
Gear type	Metal
Rotation	Dual Bearings

3.7 Liquid-crystal display (LCD)

A liquid-crystal display (LCD) is a flat-panel display or other electronic visual display that uses the light-modulating properties of liquid crystals. Liquid crystals do not emit light directly. LCDs are available to display arbitrary images (as in a general-purpose computer display) or fixed images with low information content, which can be displayed or hidden, such as preset words, digits, and 7-segment displays as in a digital clock. They use the same basic technology, except that arbitrary images are made up of a large number of small pixels, while other displays have larger elements. LCDs are used in a wide range of applications including computer monitors, televisions, instrument panels, aircraft cockpit displays, and signage. They are common in consumer devices such as DVD players, gaming devices, clocks, watches, calculators, and telephones, and have replaced cathode ray tube (CRT) displays in nearly all applications. They are available in a wider range of screen sizes than CRT and plasma displays, and since they do not use phosphors, they do not suffer image burn-in. LCDs are, however, susceptible to image persistence. The LCD screen is more energy-efficient and can be disposed of more safely than a CRT can. Its low electrical power consumption enables it to be used in battery-powered electronic equipment more efficiently than CRTs can be. It is an electronically modulated optical device made up of any number of segments controlling a layer of liquid crystals and arrayed in front of a light source(backlight) or reflector to produce images in color or monochrome. Liquid crystals were first discovered in 1888. By 2008, annual sales of televisions with LCD screens exceeded sales of CRT units worldwide, and the CRT became obsolete for most purposes.

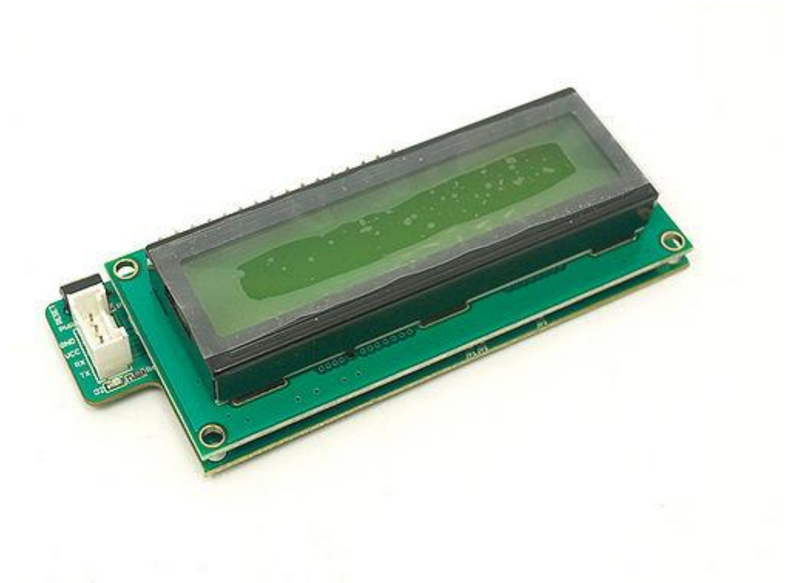


Fig 3.7 16 bit lcd display

3.8 DC motor

A DC motor is any of a class of electrical machines that converts direct current electrical power into mechanical power. The most common types rely on the forces produced by magnetic fields. Nearly all types of DC motors have some internal mechanism, either electromechanical or electronic, to periodically change the direction of current flow in part of the motor. Most types produce rotary motion; a linear motor directly produces force and motion in a straight line.

DC motors were the first type widely used, since they could be powered from existing direct-current lighting power distribution systems. A DC motor's speed can be controlled over a wide range, using either a variable supply voltage or by changing the strength of current in its field windings. Small DC motors are used in tools, toys, and appliances. The universal motor can operate on direct current but is a lightweight motor used for portable power tools and appliances. Larger DC motors are used in propulsion of electric vehicles, elevator and hoists, or in drives for steel rolling mills. The advent of power electronics has made replacement of DC motors with AC motorspossible in many applications.

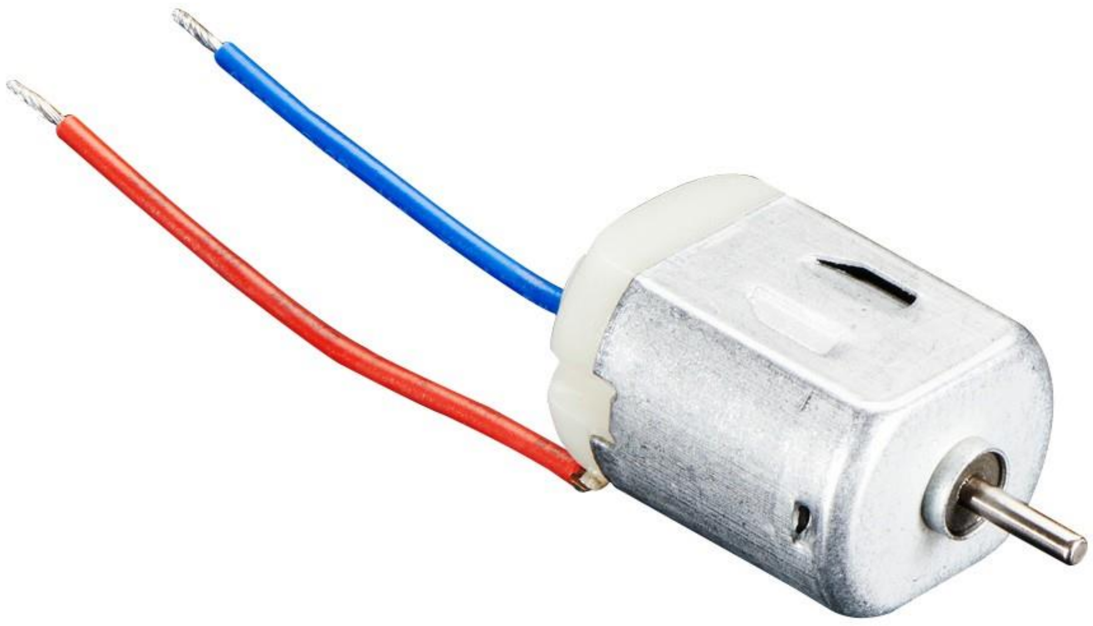


Fig 3.8 DC motor

Chapter 4

SYSTEM DESIGN AND ANALYSIS

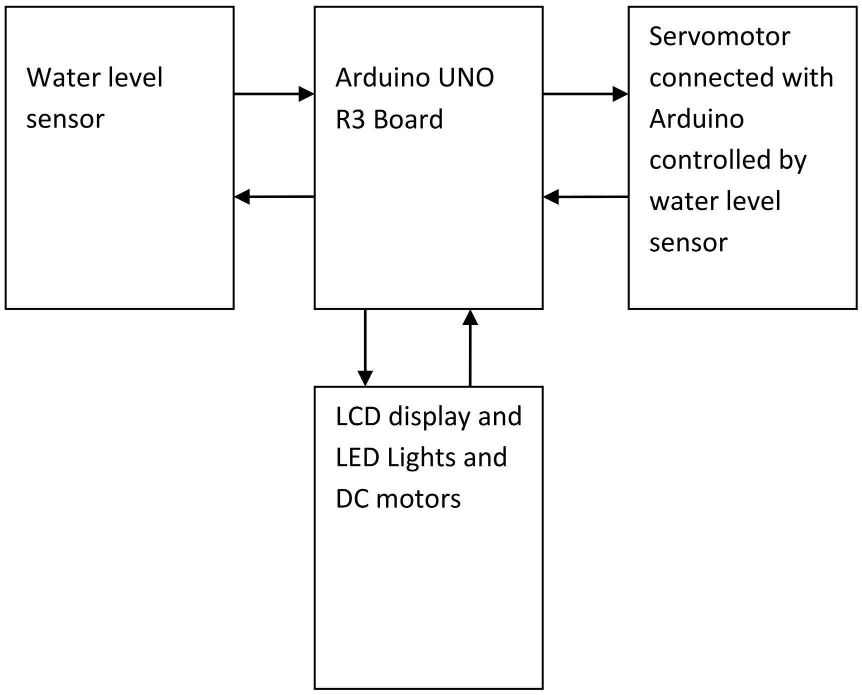
4.1 Introduction

In this project helps those people who interested to build something with Arduino. To Design a project include into two parts, one is hardware design and another part is software design. We use servo motor and 16 bit led display for the hardware design and we connected these components with microcontroller. Arduino software is downloaded from www.arduino.cc and C/C++ programmable language is used. Many examples are given in the ardiono.cc and this software is easy to usage.

4.2 Hardware Design

The whole system design is divided into two parts to design a smart water flow control system. One is the design the smart system in the breadboard and controls the designed system. Another part is the display part design to count the value in smart system. Finally, the smart water flow control system is formed a complete integrated system. In this project Arduino development board is more efficient.

4.2.1 Block Diagram of the control system


Water level sensor
Arduino UNO R3 Board
Servomotor connected with Arduino controlled by water level sensor
LCD display and LED Lights and DC motors

4.2.2 Equipments Used in this System

To design the project we use following component:

- One Arduino Uno Board.
- One bread board.
- One servo motor
- 16 bit LCD display
- RJ-45 Connector.
- UTP Cable.
- FourLED.
- 22k & 220k 420 Resistance.
- One DC motor
- Four transistor
- One diode

4.2.3 Water level measuring Circuit Design

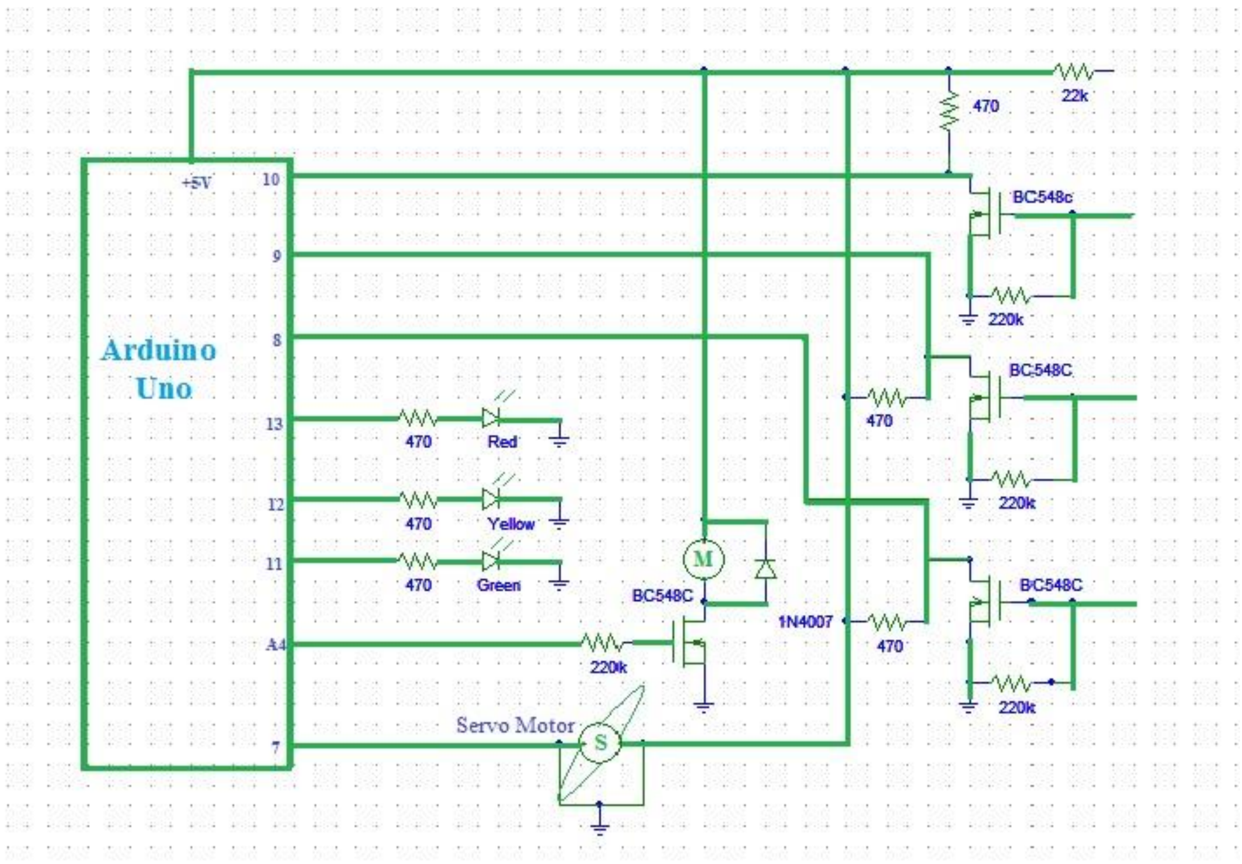
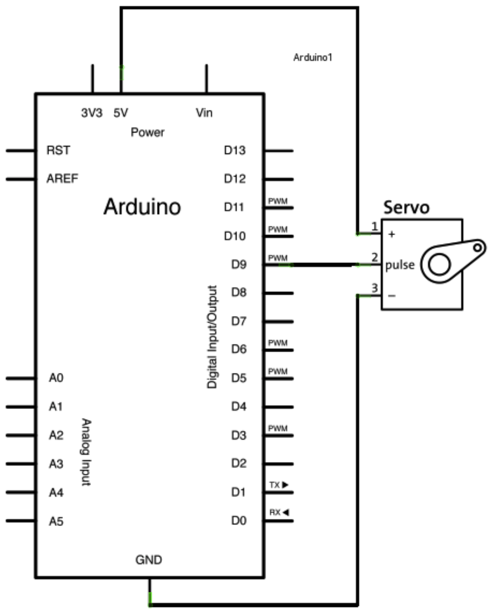


Fig 4.4 Water level measuring Circuit Design

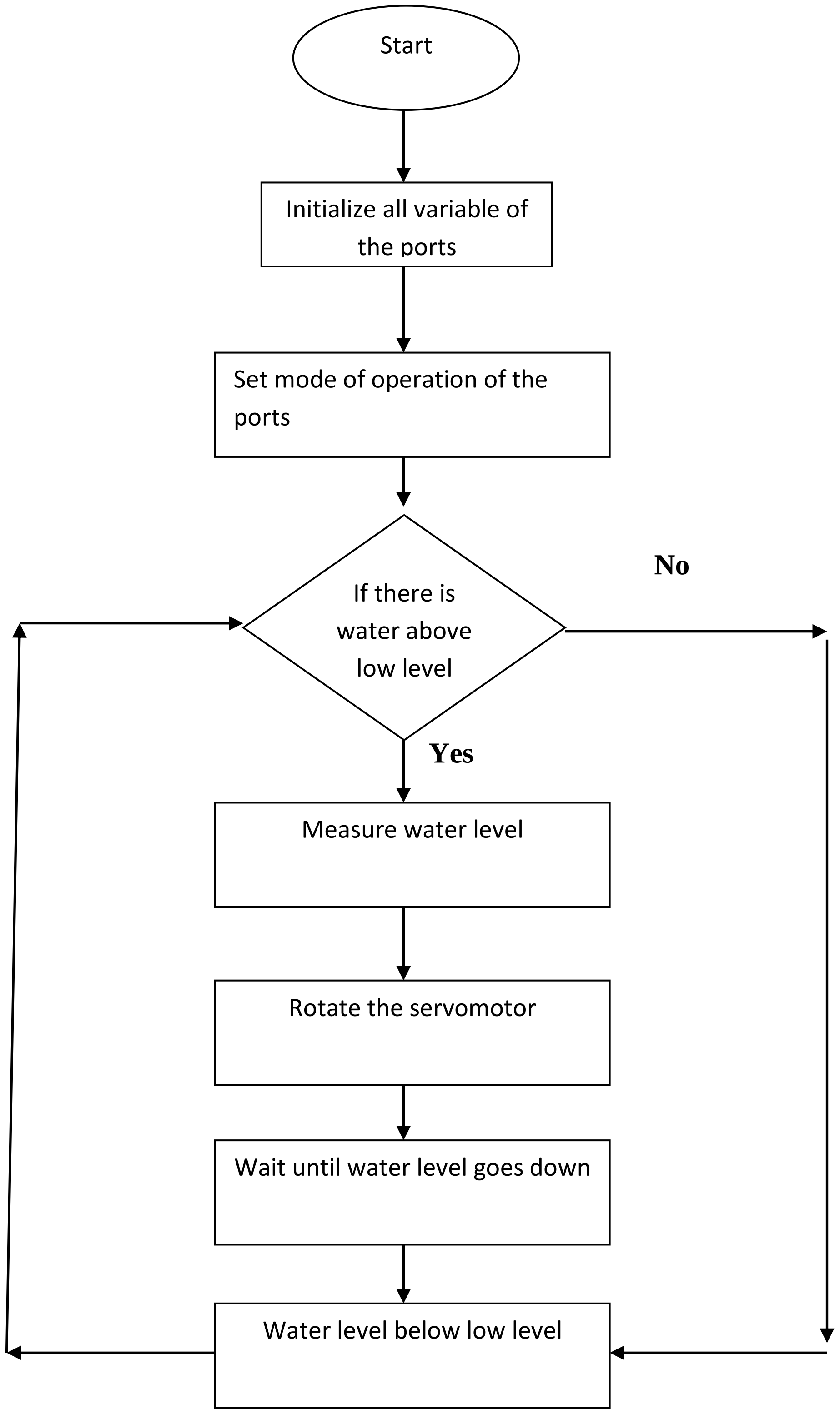
4.2.4 Power Connection With Servo Motor

In this project we use a Servo Motor to control door or entrance security system. The main operation of servo motor is to follow user instruction ; it's mainly rotate in both direction at a fixed angle by user command and can be used to control a lot of circuits. The servo motor can control the door whether it is lock or unlock at a high power maintained by this electric motor.



Motors are mainly use for control a system. Here we connect Arduino pin 3 for signaling data send from arduino with Ethernet Shield and send the command to servo motor through the net. The positive andnegative (GND) wires are connected into the breadboard, so that the board can get the actual voltage. Otherwise the systems cannot response. Servo motor has three pins are connected at the positive and negative point on the board. Third pin is connected with shield for signaling the load command. Whenever command will pass from the wed the motor words rotationally as user direction.These motors can move into forward direction and also into backward direction.

4.2.5 Flow Chart


Start
Initialize all variable of the ports
Set mode of operation of the ports
If there is water above low level
No
Yes
Measure water level
Rotate the servomotor
Wait until water level goes down
Water level below low level

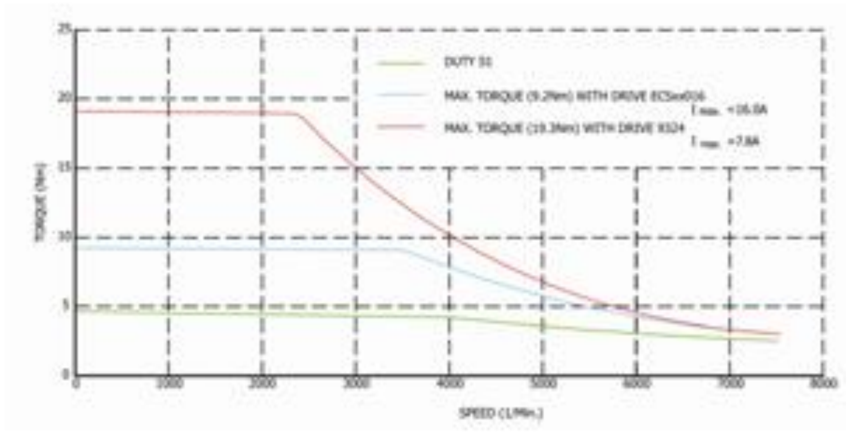
4.2.6 Motor Characteristic Curve

Using the asynchronous servomotor CV100M4 from SEW-EURODRIVE, important data for this project planning including the motor characteristics curve will be looked at in more detail below. Usually, the following motor data is known:

Motor type :	CV100M4
Rated Speed N_{rated} :	2100 l/m
Rated torque M_{rated}:	15Nm
Rated current I_{rated}:	8.1 A

Transition speed$_{trans}$: 1760 l/m (together with a 4-k servo inveter)

Special attenuation should be paid to the transition speed during project planning. The transition speed is the speed up to which the maximum torque is available for the utilizing the maximum servo inverter peak current. If the motor is operated above the transition speed, the available torque is greatly reduced. This can be clearly seen the following figure



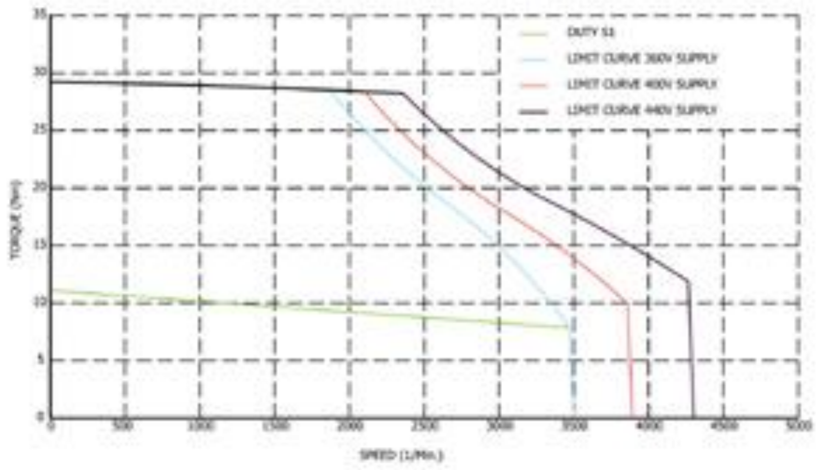


Fig 4.4 Characteristic Curve of asynchronous servo motor

4.3 Software Design

Software design is divided into two parts. First we write the Arduino program in Arduino software. Then we compile it to the Arduino hardware. This Arduino command is control the Arduino hardware and other circuit connection. For making connection between Arduino and other devices.

4.3.1 Installing Arduino

Arduino runs on Windows. Go to the Arduino software web site at http://arduino.cc/en/Main/Software and download the version of the software compatible with our system. We use Arduino 1.0.5 version.

4.3.2 Verifying the Hardware

Now that we have the Arduino IDE software installed, let's connect the computer to the Arduino board, load a small program, and verify that all components are working together. First, need to connect the USB cable to our mc board and then plug the other end of the USB cable into our computer.

4.3.3 Arduino Language

The Arduino language is implemented in C/C++ and based in Wiring. When we write an Arduino sketch, we are implicitly making use of the Wiring library, which is included with the Arduino IDE. This allows us to make run able programs by using only two functions: setup() and loop(). As mentioned, the Wiring language is inspired by Processing, and the Arduino language structure is inherited from the Processing language, where the equivalent functions are called setup(). We need to include both functions in every Arduino program, even if we don't need one of them. Let's analyze the structure of a simple Arduino sketch using again the Blink example.

4.3.4 Source Code

The source code for both the Arduino sketch and web page are a bit big to include on this page. It will show in Appendix.

Chapter 5

IMPLEMANTATION AND RESULT

5.1 Implementation

All the parts are connected as circuit design. Then we upload the programming code in the Arduino and we get positive result. It works properly according to our design. After connecting the full circuit we make some test . First we test the water sensor without servo motor and it works properly. We measure the water level by putting the sensor in a bowl and it gives us the right measurement of the water level. Then we test our servomotor controlling by the water level sensor. It also works properly as we expected. We install a sufficient program for the servo rotate. We connect three LED lights to measure the progress of the measurement. We also monitor our in LCD display we are connected to the brad board.

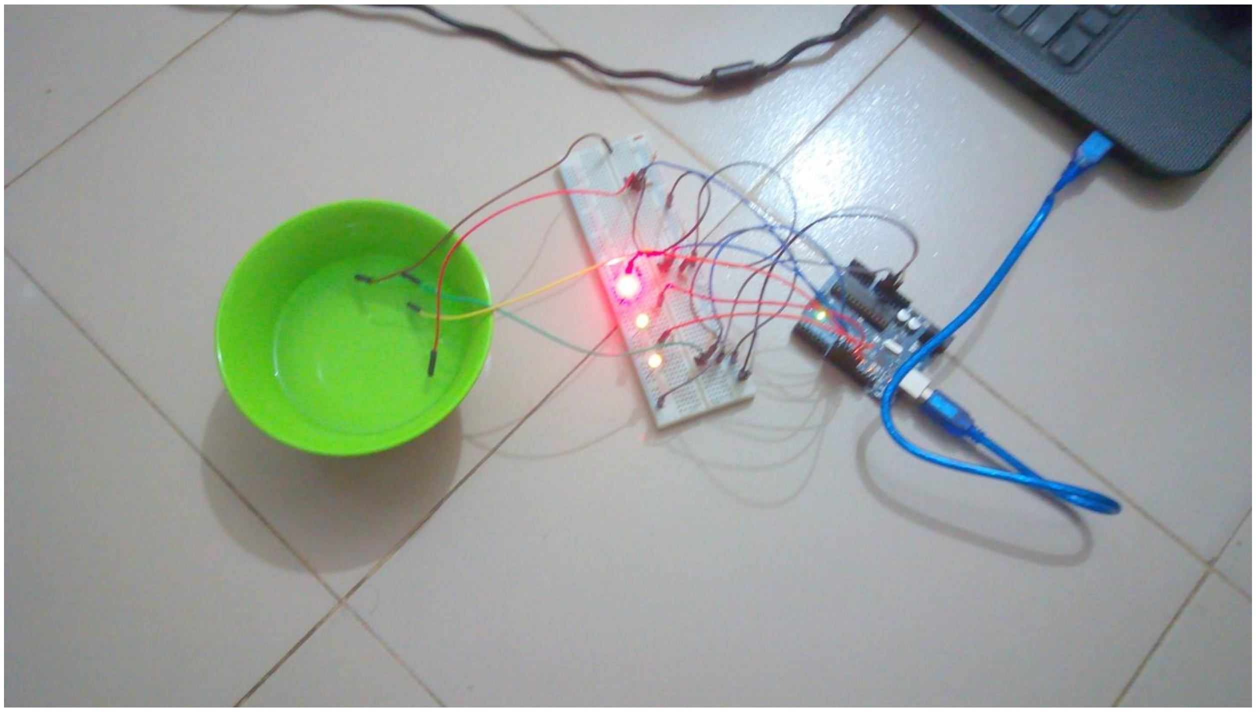


Fig 5.1 Connection without servo

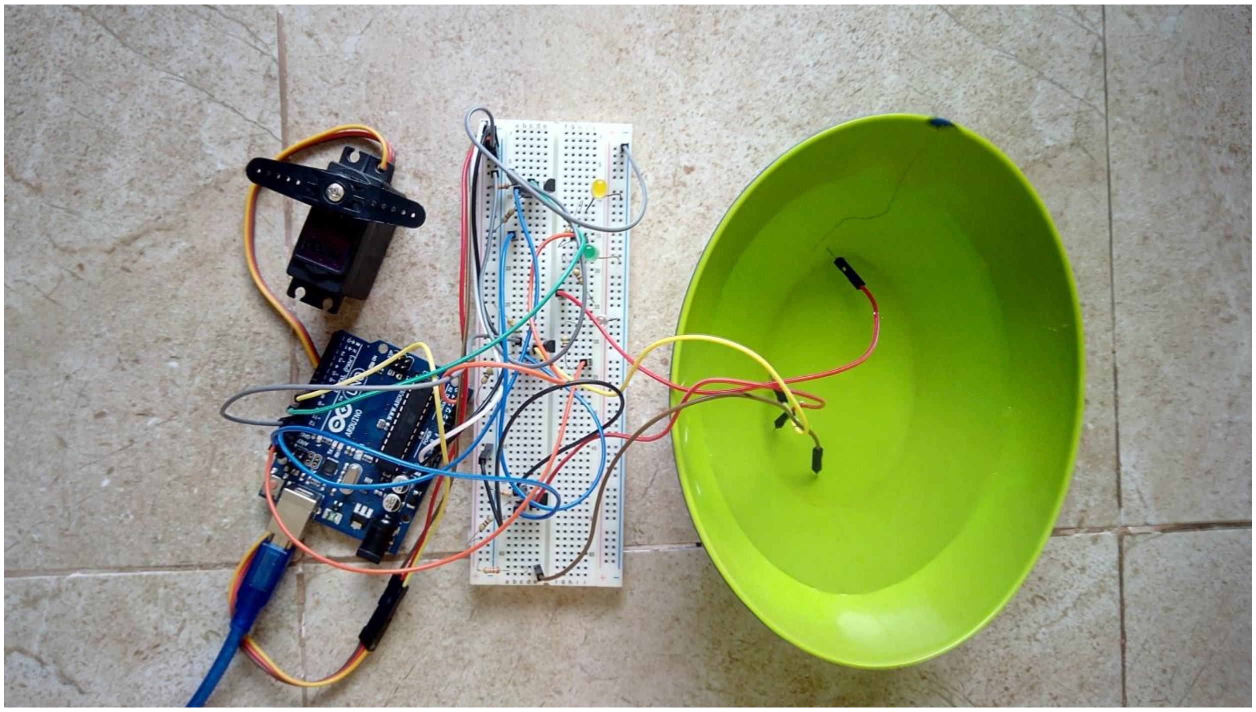


Fig 5.1 Connection with servo on

Chapter 6

CONCLUSION

6.1 Conclusion

Automatic water level measure and control system employs the use of different technologies in its design, development, and implementation. The system used microcontroller to automate the process of water level measuring in an water bowl and has the ability to detect the level of water in a bowl, switch on/off the DC motor accordingly and display the status on an LCD screen. This research has successfully provided an improvement on existing water level controllers by its use of calibrated circuit to indicate the water level and use of DC instead of AC power thereby eliminating risk of electrocution.

APPENDIX

1. Programming Code Arduino Scatch

```
#include <LiquidCrystal.h>
#include <Servo.h>
Servo servoMain; // Define our Servo
LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
byte sensorPin[] = {8, 9, 10};
byte ledPin[] = {11, 12, 13}; // number of leds = numbers of
sensors
const byte sensors = 3;
int level = 0;
int motor = A4;

void setup() {
servoMain.attach(10); // servo on digital pin 10
```

```
  for(int i = 0; i < sensors; i++) {
     pinMode(sensorPin[i], INPUT);
     pinMode(ledPin[i], OUTPUT);
  }
  pinMode(motor, OUTPUT);
  lcd.begin(16, 2);
}

void loop() {
  level = 0;
  for(int i = 0; i < sensors; i++) {
     if(digitalRead(sensorPin[i]) == LOW) {
        digitalWrite(ledPin[i], HIGH);
        level = sensors - i;
     } else {
       digitalWrite(ledPin[i], LOW);
     }
```

```
}
lcd.clear();
lcd.print("Water level");
lcd.setCursor(0,1);
switch(level) {
   case 1:
    lcd.print("HIGH");
    digitalWrite(motor, HIGH);
    servoMain.write(45);
    break;
   case 2:
     lcd.print("AVERAGE");
     servoMain.write(90);
     digitalWrite(motor, LOW);
       servoMain.write(0);


     break;
```

REFERENCE

1. https://www.arduino.cc/en/main/arduinoBoardUno
2. http://digital.csic.es/bitstream/10261/127788/7/D-c-%20Arduino%20uno.pdf
3. https://en.wikipedia.org/wiki/Microcontroller
4. http://www.sciencebuddies.org/science-fair-projects/project_ideas/Robotics_ServoMotors.shtml
5. https://en.wikipedia.org/wiki/Servomotor
6. http://www.paleotechnologist.net/?p=3594
7. https://en.wikipedia.org/wiki/Liquid-crystal_display
8. https://en.wikipedia.org/wiki/DC_motor
9. Microcontroller chip Technology, 2001, PIC16F84A Datasheet www.microchip.com
10. http://www.webopedia.com/TERM/W/Wi_Fi.html
11. 17.http://www.webopedia.com/DidYouKnow/Computer_Science/wireless_networks_explained.asp
12. 18. http://compnetworking.about.com/cs/wireless/g/bldef_ap.htm
13. 19.http://www.google.com.bd/imgres?imgurl=http://i.embed.ly/1/display/resize%253Fkey%253D1e6a1
14. 20. http://www.instructables.com/id/Arduino-Ethernet-Shield-Tutorial/step3/Get-started/
15. 21. http://startingelectronics.org/tutorials/arduino/ethernet-shield-web-server-tutorial/web-page-structure/