Visvesvaraya Technological University, Belagavi.

PROJECT REPORT
on

"Advanced power regeneration from treadmill machine for gym industries with fingerprint access for authentication"

Project Report submitted in partial fulfillment of the requirement for theaward of the degree of

Bachelor of Engineering
in
Electronics and Communication Engineering
For the academic year 2019-20

Submitted by

Madhumathi	1CR16EC417
Basavarajeshwari	1CR17EC404
Suvarna	1CR17EC430

Under the guidance of

Internal
Guide 1
Dr. Imtiyaz Ahmed BK
Asst. proffesor
Department of ECE
CMRIT,Bengaluru

External
Guide 1
Designation
Company
Place

Department of Electronics and Communication Engineering
CMR Institute of Technology, Bengaluru – 560 037

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to Certify that the dissertation work **"**Advanced power regeneration from treadmill machine for gym industries with fingerprint access for authentication**"**carried out byMadhumathi (1CR16EC417), Basavarajeshwari (1CR17EC404), Suvarna (1CR17EC430)bonafide students of **CMRIT** in partial ulfillment for the award of **Bachelor of Engineering**in**Electronics and Communication Engineering**of the**Visvesvaraya Technological University, Belagavi,** during the academic year **2019-20**. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited in the departmental library. Theproject report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said degree.

Signature of Guide Signature of HOD Signature of
Principal

Dr. Imtiyaz Ahmed B K Dr. R. Elumalai Dr.Sanjay Jain
Asst. professorHead of the Department, Principal,
Dept. of ECE., Dept. of ECE., CMRIT,
CMRIT,Bengaluru.CMRIT,Bengaluru. Bengaluru.

External Viva

Name of Examiners Signature & date

.

2.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose consistent guidance and encouragement crowned our efforts with success.

I consider it as my privilege to express the gratitude to all those who guided in the completion of the project.

I express my gratitude to Principal, **Dr. Sanjay Jain,** for having provided me the golden opportunity to undertake this project work in their esteemed organization.

I sincerely thank Dr. R. Elumalai,HOD**,** Department of Electronics and Communication Engineering, CMR Institute of Technology for the immense support given to me.

I express my gratitude to our project guideDr. Imtiyaz Ahmed B K, Asst. professor for their support, guidance and suggestions throughout the project work.

Add more as necessary

Last but not the least, heartful thanks to our parents and friends for their support.

Above all, I thank the Lord Almighty for His grace on us to succeed in this endeavor.

CONTENTS

Table of Contents

List of Figures

List of Tables

Page no

Advanced Power Regeneration from Treadmill Machine for GYM Industries with Fingerprint Access for Authentication

ABSTRACT

Now a day's exercises play an important role in human life. As we know the exercising will reduce the amount of excess calorie of the body and annihilates the metabolic activities of the body. When doing exercise, a large amount of human energy is get wasted. Our project is mainly aim to convert this energy into sufficient form and for making the exercise more convenient by a new design. For that we designed a Treadmill cycle where the pedal of the cycle is fully replaced by a treadmill. The treadmill will drive the rear wheels of the cycle via a chain drive, so that its need only the effort of exercising in treadmill to travel a short distance conveniently. A stand is provided to make the Cycle inclined at stationery state and can be use the cycle as a perfect treadmill. A powerful dynamo and a battery is provided to the rotating parts of the cycle so that it can produce and store electrical energy during exercising or travelling. We can use this electrical energy in emergency situations such as power failure, it can light some lamps in a room and can give power for music systems etc, for Authentication we installed fingerprint sensor.

Chapter 1

INTRODUCTION

The treadmill cycle is totally new way of moving it is done by the combination of various mechanical part i.e. chain and gear. Motion of cycle from one place to another place will be done by human effort. Treadmill cycle basically shows a new concept for both travelling and exercising. Due to the ever-increasing demand of fuel for various purposes it also eliminates the use of any fuel in any case. The gears present in it provide us the speed which is required for travelling faster. This makes this cycle different from a simple bicycle.

A treadmill is a device used for exercises by staying in the same place, walking or running can be done on it. From the principle of treadmill working integrated with the principle of electromagnetic induction, a new cycle is designed which is capable of generating electric power while exercising. The most striking feature of this design is that it can be made possible as a mobile and power generating treadmill.

The walking cycle has a simple mechanism, operated with free wheels, gear chain, bearing shaft and links arrangement. As the straight line motion during walking gets converted to rotary motion through very simple movement by means of a gear chain and free wheel mechanism of the linkages. The rotary motion is again converted in to linear motion of the cycle through gear chain and free wheels arrangement. The conveyor system is either continuous movement or intermittent which is completely based on the person.

This invention relates to improvements in transport devices, and it relates particularly to devices for transferring people, with small in number in case of a bike or a cycle. The Walking Bicycle is the one, which combines walking and cycling into one activity. This combines the two activities into a straight line motion simply by walking on the belt provided, allowing to propel forward at desirable speed. Usually, the operation of the walking cycle machine is controlled by the user itself by simply walking on the treadmill belt and also balancing the cycle. The operating speed of the walking cycle differs on the amount of force applied by the user.

Chapter 2

LITERATURE SURVEY

The Design of Treadmill with Distributed Generation Function Yingbo Yu ; Haohua Qin IEEE 2019

Physical health is very important to economic development, and with the improvement of the living standard, indoor treadmills are more and more popular. However, traditional treadmill drive system has many drawbacks. In order to meet the requirements of low-carbon environmental protection, it is necessary to develop an electromagnetic damping control system capable of distributed generation to improve the shortcomings of traditional treadmill drive system. Therefore, we designed a treadmill that uses PWM mode to adjust the electromagnetic damping, which can guarantee the accuracy of damping adjustment during the movement process. This system integrated intelligent charging management and data networking function together, which can greatly enhance the practicability of the treadmill. If a large number of such treadmills are brought together and the electricity generated from the treadmills is so large that could be connected to the grid, which will effectively alleviate the energy shortage.

An innovative technique of electricity generation and washing machine application using treadmill Sahil ; P.K. Sharma ; N. Hari ; N. Kumar ; D. Shahi IEEE 2018

The scope of this paper focuses upon a method of generating electrical energy from the application of specially designed treadmill with the provision of specially designed washing machine, which will work when a human runs over a treadmill for exercise which is an integral part of modern society life. In this

proposed model, a permanent magnet dc generator of 1200 W rating is used which will produce dc power, which can be further used for electrical utilities and small power applications. The washing machine system involves a tub type arrangement which is connected to the treadmill via gearbox system. The power produced by the execution of aforementioned design is used to fulfill all the basic requirements of electrical appliances of household such as mobile battery charging, emergency battery backup for lights etc. The goal of this research paper is to exploit mechanical energy of rotating belt as an alternative source for the generation of electricalenergy for the applications of basic daily household needs. This technology will help in reduce considerably, the fuel consumption required for the electrical power generation for fulfilling the electrical requirements of household appliances.

Treadmill Bicycle with Regenerative Power RohitMalviya, SagarkumarBhumarkar, Aman Jain, TarunBagahe and Hemant Dawande IEEE 2018

In this project we study the treadmill exercise outdoor and their effect on health. One of the most popular types of home as well as outdoor exercise equipment is the treadmill bicycle, which provides a straightforward, efficient aerobic workout. For many, treadmills are a good choice to begin a new exercise routine because walking is well tolerated by most individuals regardless of fitness level and for most back conditions. As strength and endurance are developed, the treadmill bicycle can be used for jogging or for interval training. The modern challenge faced with the global energy situation is the growing energy demand and the strong dependence on unsustainable fossil fuels. Another concurrent issue is the adverse health and socio-economic implications of adult obesity. Human Power Generation, which uses

metabolized human energy to generate electrical power, could potentially address both these challenges. The treadmill, one of the most popular exercise machines, presently consumes large amounts of energy while dissipating a majority as heat. A heavy duty rechargeable battery pack was used to store the generated energy and additional components to measure the generated power were included. The power generating potential of the generator was determined for varying belt speeds and angles of inclination.

DESIGN AND FABRICATION OF TREADMILL CYCLE Prof. P. R. Gajbhiye1 , Prof. Dhananjay G. Dange , Shubham. C. Hingnekar , Raunak. V. Kondalwar , Nazeefuddin Jamal , Mohit. G. Sonwane , Mohit. G. Shete IEEE2018

Exercise is inevitable to keep health in good status. In this project we study the treadmill exercise outdoor and their effect on health. Also we enlisted the advantages and disadvantages of treadmill cycle exercise. One of the most popular types of home as well as outdoor exercise equipment is the treadmill cycle, which provides a straightforward, efficient aerobic workout. For many, treadmills are a good choice to begin a new exercise routine because walking is well tolerated by most individuals regardless of fitness level and for most back conditions. As strength and endurance are developed, the treadmill bicycle can be used for jogging or for interval training. The modern challenge faced with the global energy situation is the growing energy demand and the strong dependence on unsustainable fossil fuels. Another concurrent issue is the adverse health and socio-economic implications of adult obesity. Human Power Generation, which uses metabolized human energy to generate electrical power, could potentially address both these challenges. The treadmill, one of the most popular exercise machines, presently consumes large amounts of energy while

dissipating a majority as heat. The purpose of this thesis project was to design and develop a human powered treadmill generator and determine its power generation potential. A heavy duty rechargeable battery pack was used to store the generated energy and additional components to measure the generated power were included. The power generating potential of the generator was determined for varying belt speeds and angles of inclination, and compared with the American College of Sports Medicine (ACSM) metabolic walking and running prediction equations to determine efficiency. The generator was able to deliver 140W peak power for a short period of time. Regression equations related the power generated to the belt speed, covering values ranging from an average 10.8±0.36W at 1.83±0.045m/s to 90.3±3.04W at 2.38±0.054m/s. The angle of inclination did not have a significant impact on energy generation. The max average efficiency obtained for the system in this study was 37.9±2.63%, assuming 25% gait efficiency. Possible applications for this concept include energy saving equipment in a gym, low-cost, simple to operate, and low maintenance solutions for developing nations, and as a tool to educate energy conservation. Also, the need for exercise in space with low gravity makes the treadmill generator a possible source for secondary power in future extraterrestrial environments.

WALKING E-BIKE Ajan C R, Ajay Sugadan K, Akshay Balachandran IRJET 2018

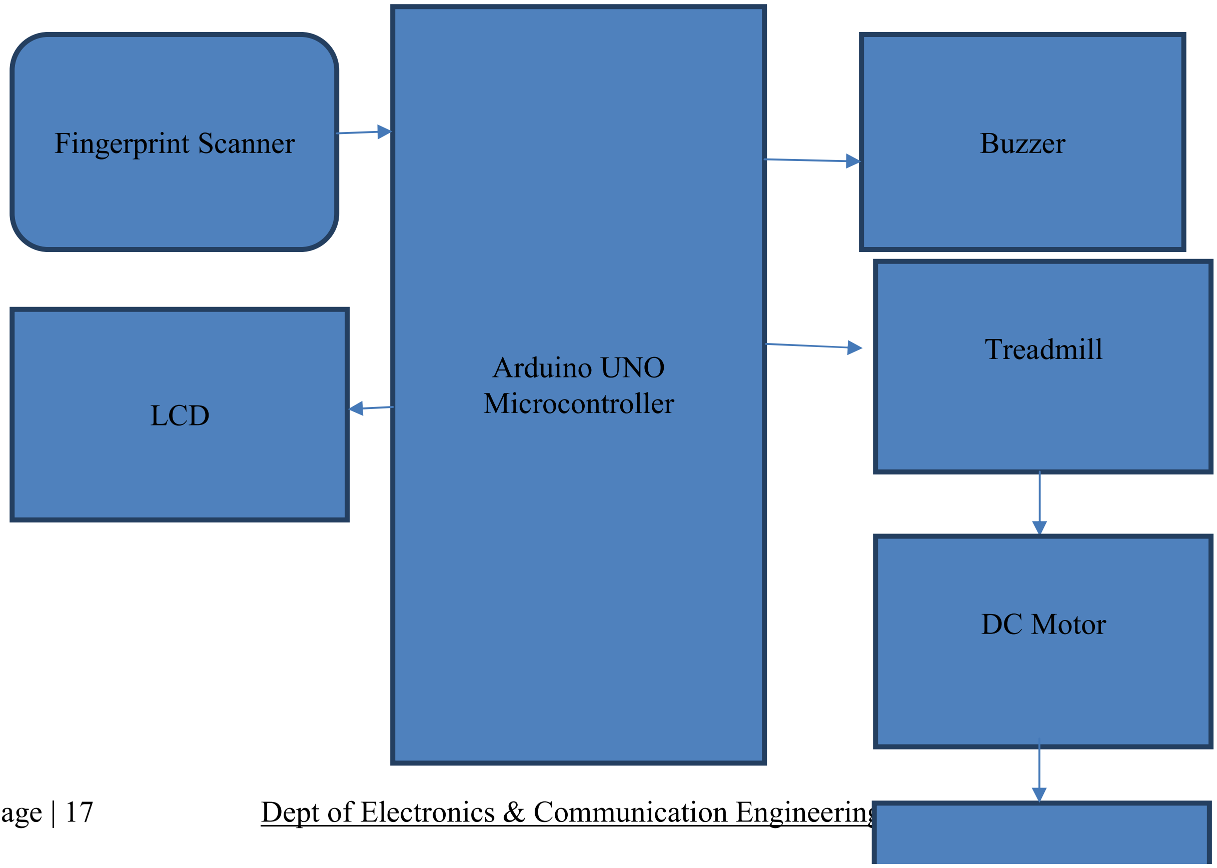
Walking E-Bike is a project which focuses on the design and fabrication of a totally new way of moving. The idea is to create economic and ecological exercise machine/transport. The bike responds to walking input on the treadmill belt and powers it. Alternative power is given to the wheel using a motor. The

transmission is done using a series of chain drives. The structure is well balanced and strongly built to withstand higher loads. The motor supports the vehicle to move in difficult terrains and steep slopes.

Chapter 3

METHODOLOGY

Block Diagram



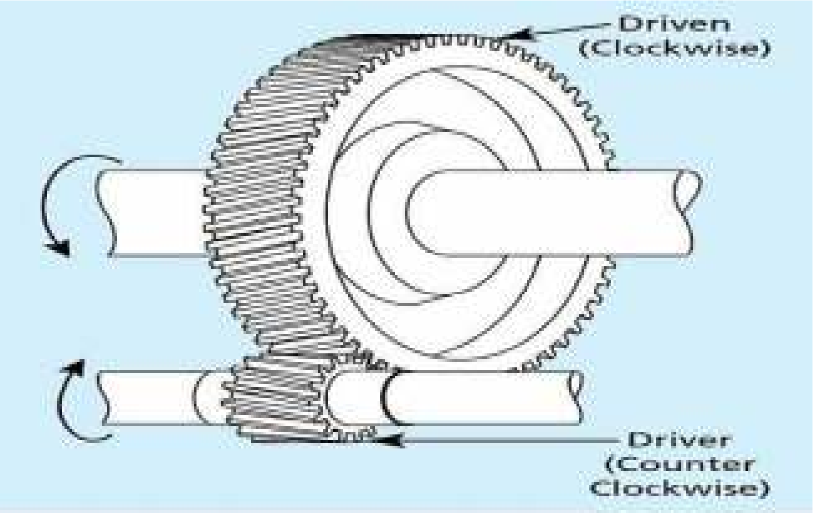
Working

The treadmill Cycle working on the basis of energy conversion. When a man is exercising on a treadmill his effort is gets wasted. But in this treadmill Cycle, the human effort is converted as useful work. Travelling and exercising can be done at a time. The conveyor is driven by manually. The main roller is fixed on the rear side and a helical driver gear is fitted to the roller. When the conveyor moves by leg power which will rotates the roller and the helical gear attached to the roller also rotates. Another shaft is fitted with a driven helical gear is then attached to the main helical gear. A sprocket is fixed on the second shaft will give drive to the rear wheels via chain drive. Rear wheel assembly will support and stabilize the arrangement and for access we initialized a Thumb Impression.

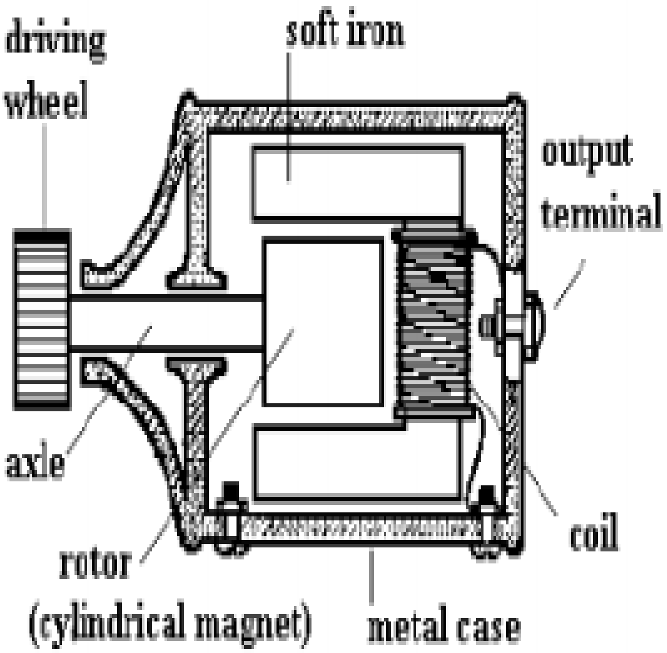
Electricity Generation

To harness the power of animals or humans for doing work, treadmills were introduced, which are a type of mill operated by a person or animal treading steps of a tread-wheel to grind grain. Treadmills are not used to harness power nowadays, but used as an exercise machinesfor running or walking. The machine provides a moving platform with a wide conveyor belt (track) rather than the user powering the mill and was driven by an electric motor. This simple, light and low budget treadmills passively resist the motion, moves only when walkers push the belt with their feet. But an addition of small DC generators were done, whose moving parts are mechanically coupled with the moving rollers of machine that moves when belt of the treadmill is moving. When the rotor of the DC generator starts moving, an emf will be produced across its output terminals. This generated emf can be used for charging of Battery or other purposes.

Gear coupling for driving mechanism



In a bicycle generator, a small dc generator is attached to one of the wheel of the bicycle. When the bicycle runs, the rotor which is attached to the cycle wheel also rotates and emf will be generated across the output terminals of the generator. This emf is then generally used for lighting the head-light of the bicycle.



Bicycle generator

Working Principle

The bicycle generator is small and a low torque is required to rotate its rotor. Here in the treadmill, instead of using one single large generator a number of small generators is used, which are electrically parallel connected and mechanically roller coupled. In a treadmill, the belt moves on some cylindrical shape of rollers and those rollers are surroundedby the belt in both upper and lower sides. Each join side (left and right) of the roller is mechanically coupled with the rotor of a small DC generator such that as the roller rotates, the rotor also starts to rotate.

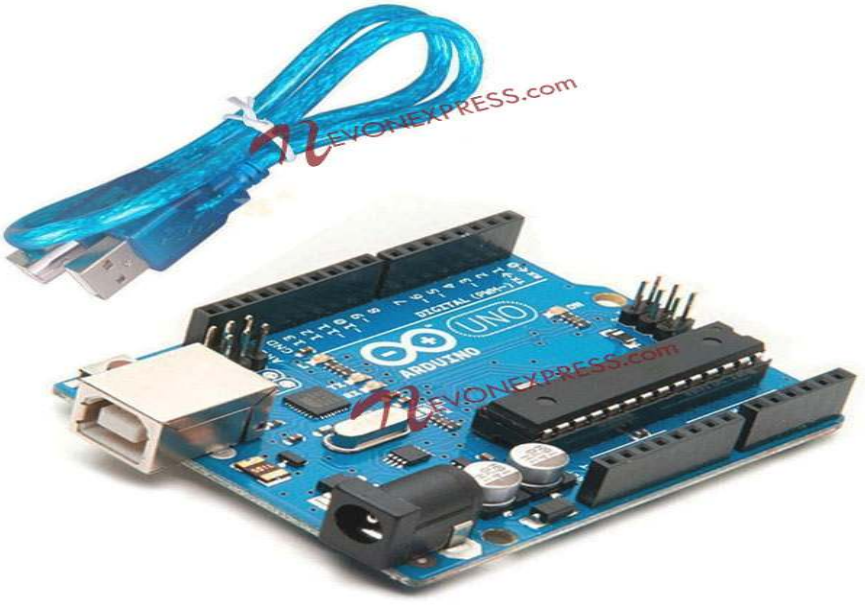
The DC generators 1, 2, 3, etc. are rotating in opposite direction with respect to the direction of the rotation of DC generators 1’, 2’, 3, etc. So, the emfs generated by them is also in 1800 out of phase with respect to the other generator, situated in opposite side of the roller. To eliminate this problem, 1, 2, 3 etc. should be connected parallel to the opposite terminals of 1’, 2’, 3’etc. Also a diode should be connected with the positive terminals of each DC generators. This will prevent them to work as a motor as they are mainly connected to the battery and their main work is to charge the battery not to take energy from it. If for any reason the belt of the treadmill is running in opposite direction, then these diodes will prevent the current to circulate in opposite direction. There is also a capacitor connected across the output terminals to prevent the fluctuations of the DC output voltage and keep it steady-state or at a constant value.

This speed is enough to generate 6W electric power for each DC generator at 12V (output voltage). Here, in the above arrangement of the treadmill we assume ten rollers which are surrounded by the belt. With each roller two DC generators are mechanically connected or coupled. So, the total number of DC generators is = (2 * 10) = 20. Thus, the total power generating capacity of this system is = (20 * 6) = 120 W. If one man run on this treadmill (with average speed of 10 km/h) for one hour then the total electrical energy produced by this system is = (120 * 1) = 120 Watt-hour. Now, this energy can be stored in rechargeable DC batteries. Once the batteries are charged then we can use this energy for lighting or other purposes. The efficiency of the whole system is varying from 80% to 90%. Let, we see some of its applications. Suppose we have a 10 Watt LED bulb, then we can be lighting this bulb by directly connecting it with battery (or with chopper circuit if needed to adjust the DCvoltage at rated value) for (120 / 10) * 0.85 = 10.2 hours (taking efficiency 85%). We can also be lighting a 40 Watt tube light (with suitable inverter circuit) for 3 to 4 hours. Special type of inverter circuit by which we can directly lighting 40 Watt tube-light without any choke and starter arrangement. Here we can use this inverter circuit for lighting tube-light.

Chapter 4

HARDWARE

ARDUINO UNO:



ARDUINO UNO is an open –source electronics platform based on easy-to-use hardware and software. ARDUINO boards are able to read inputs –light on sensor, a finger on a button, or a Twitter message and turn it into an output –activating a motor, turning on an LED, publishing something online. ARDUINO was born at the "IVERA INTERACTION DESIGN INSTITUTE "as an easy tool for fast prototyping.

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 ceramic resonator, 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 Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.

The Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.

The board has the following new features:

- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltageprovided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for futurepurposes.
- Stronger RESETcircuit.
- Atmega 16U2 replace the8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Summary

Microcontroller	ATmega32

	8
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Schematic & Reference Design

EAGLE files: arduino-uno-Rev3-reference-design.zip(NOTE: works with Eagle 6.0 and newer) Schematic: arduino-uno-Rev3-schematic.pdf

Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

Power

The Arduino Uno 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 of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be 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:

- **VIN.** The input voltage to the Arduino 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 thispin.
- **5V.**This pin outputs a regulated 5V from the regulator on the board. The board can besupplied 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 adviseit.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50mA.
- **GND.** Groundpins.

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

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 a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. 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 Serialchip.
- **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 attachInterrupt()function for details.

PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite()function.

- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPIlibrary.
- **LED: 13.** There is a built-in LED connected to digital pin 13.

When the pin is HIGH value, the LED is on, when the pin is LOW, it'soff.

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. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wirelibrary.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used withanalogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one ontheboard.

See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

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 ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows,a .in 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-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial libraryallows for serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentationfor details. For SPI communication, use the SPIlibrary.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the referenceand tutorials.

The ATmega328 on the Arduino Uno comes preburned with a bootloaderthat allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line toground, making it easier to put into DFUmode.

You can then use Atmel's FLIP software(Windows) or the DFU programmer(Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorialfor more information.

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half- second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending thisdata.

The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

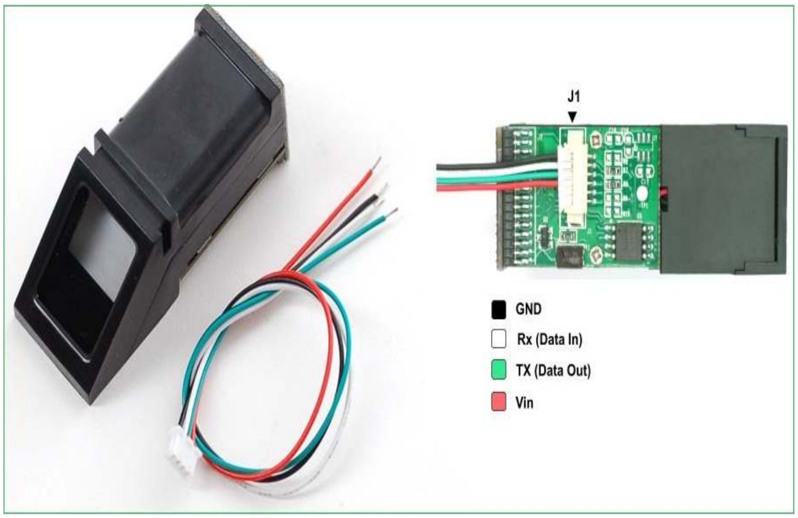
The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the boardto be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the otherpins.Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing. Using Microcontroller, we can control each macule like Fingerprint, Buzzer, LCD etc.,

Fingerprint Scanner

Here we are using Fingerprint scanner for Accessing for the Treadmill, without fingerprint we can’t access this module it is interfaced with Arduino UNO Microcontroller.



Biometrics

Biometrics is a means of identifying a person by measuring a particular physical or behavioral characteristic and later comparing it to a library of characteristics belonging to many people. Among the various characteristics measured are; Face Fingerprints, Hand geometry, Handwriting, Iris, Retinal, Vein, and Voice. Retinal identification is the most accurate of the biometric methods used at this time. Fingerprint identification is the oldest biometric to identify a person.

Authentication:

For other uses of the terms "authentication", "authentic" and "authenticity". Authentication (from Greek: real or genuine, from authentic; author) is the act of establishing or confirming something (or someone) as authentic, that is, that claims made by or about the thing are true. This might involve confirming the identity of a person, the origins of an artifact, or assuring that a computer program is a trusted one.

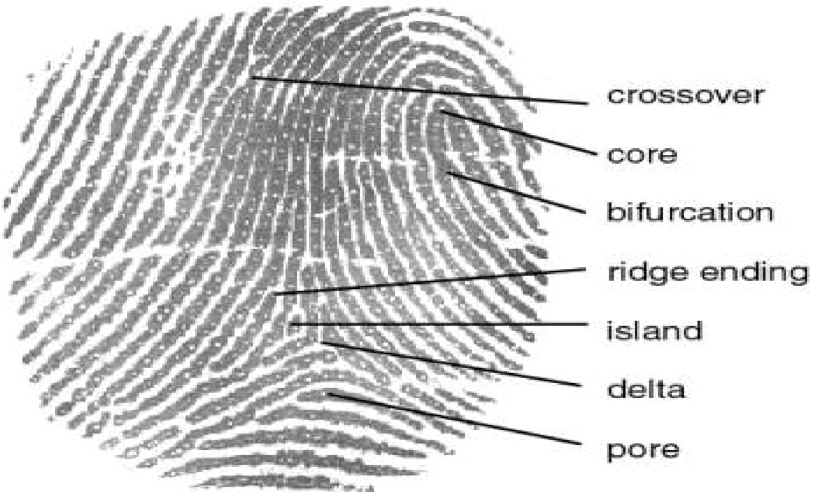
Humans have used fingerprints for personal identification for a very long time. Modern fingerprint matching techniques were initiated in the late 16th century. Henry Fauld, in 1880, first scientifically suggested the individuality and uniqueness of fingerprints. At the same time, Herschel asserted that he had practiced fingerprint identification for about 20 years. This discovery established the foundation of modern fingerprint identification. In the late 19TH century, Sir Francis Galton conducted an extensive study of fingerprints. He introduced the minutiae features for single fingerprint classification in 1888. The discovery of uniqueness of fingerprints caused an immediate decline in the prevalent use of anthropometric methods of identification and led to the adoption of fingerprints as a more efficient method of identification. An important advance in fingerprint identification was made in 1899 by Edward Henry, who (actually his two assistants from India) established the famous “Henry system” of fingerprint classification an elaborate method of indexing performing (manual) fingerprint identification. In the early 20TH century, fingerprint identification was formally accepted as a valid personal identification method by law enforcement agencies and became a standard procedure in forensics. Fingerprint identification agencies were setup worldwide and criminal fingerprint databases were established. With the advent of livescan fingerprinting and availability of cheap fingerprint sensors, fingerprints are increasingly used in government and commercial applications for positive person identification. Fingerprints very much tuned to facilitating the human experts.

FINGERPRINT DEFINITION:

A smoothly flowing pattern formed by alternating crests (ridges) and troughs (valleys) on the palmar aspect of hand is called a palmprint. Formation of a palmprint depends on the initial conditions of the embryonic mesoderm from which they develop. The pattern on pulp of each terminal phalanx is considered as an individual pattern and is commonly referred to as a fingerprint. A fingerprint is believed to be unique to each person (and each finger) 2. Fingerprints of even identical twins are different.

Fingerprints are one of the most mature biometric technologies and are considered legitimate proofs of evidence in courts of law all over the world. Fingerprints are, therefore, used in forensic divisions worldwide for criminal investigations. More recently, an increasing number of civilian and commercial applications are either using or actively considering to use fingerprint-based identification because of a better understanding of fingerprints as well as demonstrated matching performance than any other existing biometric technology.

A **fingerprint** is an impression of the friction ridges of all part of the finger. A friction ridge is a raised portion of the epidermis on the palmar (palm) or digits (fingers and toes) or plantar (sole) skin, consisting of one or more connected ridge units of friction ridge skin. These are sometimes known as "dermal ridges" or "dermal papillae".



Fingerprints may be deposited in natural secretions from the eccrine glands present in friction ridge skin (secretions consisting primarily of water) or they may be made by ink or other contaminants transferred from the peaks of friction skin ridges to a relatively smooth surface such as a fingerprint card. The term fingerprint normally refers to impressions transferred from the pad on the last joint of fingers and thumbs, though fingerprint cards also typically record portions of lower joint areas of the fingers (which are also used to make identifications).

A fingerprint is the feature pattern of one finger (Figure 1.1.1). It is believed with strong evidences that each fingerprint is unique. Each person has his own fingerprints with the permanent uniqueness. So fingerprints have being used for identification and forensic investigation for a long time.

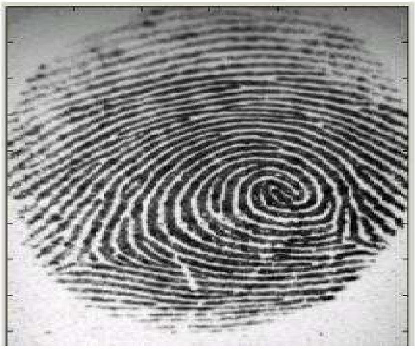


Figure 2.1 A fingerprint image acquired by an Optical Sensor

A fingerprint is composed of many ridges and furrows. These ridges and furrows present good similarities in each small local window, like parallelism and average width.However, shown by intensive research on fingerprint recognition, fingerprints are not distinguished by their ridges and furrows, but by Minutia, which are some abnormal points on the ridges (Figure 2.2). Among the variety of minutia types reported in literatures, two are mostly significant and in heavy usage: one is called termination, which is the immediate ending of a ridge; the other is called bifurcation, which is the point on the ridge from which two branches derive.

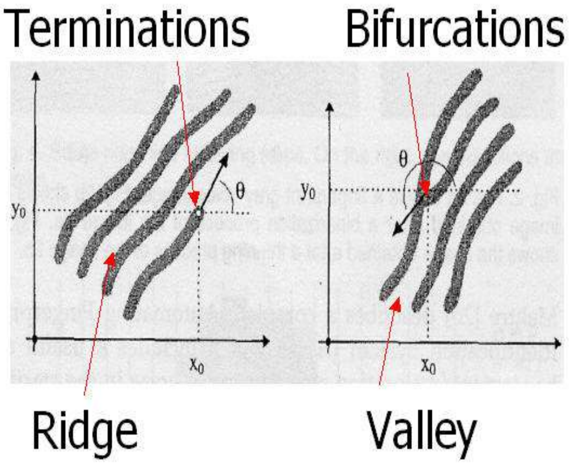


Figure 2.2 Minutia. (Valley is also referred as Furrow,
Termination is also called Ending,
and Bifurcation is also called Branch)

4. fingerprint authentication

Steps for fingerprint Authentication:

Step 1: User Registration

In any secure system, to enroll as a legitimate user in a service, a user must beforehand register with the service provider by establishing his/her identity with the provider. For this, the user provides his/her fingerprint through a finger scanner. The finger print image thus obtained undergoes a series of enhancement steps. This is followed by a Finger print hardening protocol with servers to obtain a hardened finger print FP which is stored into the server's database.

Step 2: Fingerprint Enhancement

A fingerprint is made of a series of ridges and furrows on the surface of the finger. The uniqueness of a fingerprint can be determined by the pattern of ridges and furrows. Minutiae points are local ridge characteristics that occur at either a ridge bifurcation or a ridge ending. A ridge termination is defined as the point where a ridge ends abruptly. A ridge bifurcation is defined as the point where a ridge forks or diverges into branch ridges.

The quality of the ridge structures in a fingerprint image is an important characteristic, as the ridges carry the information of characteristic features required for minutiae extraction.

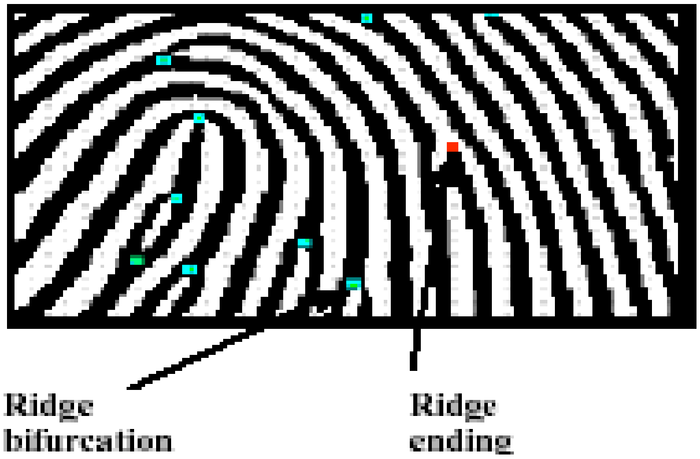


Fig 4.1: Example for ridge bifurcation and ridge ending

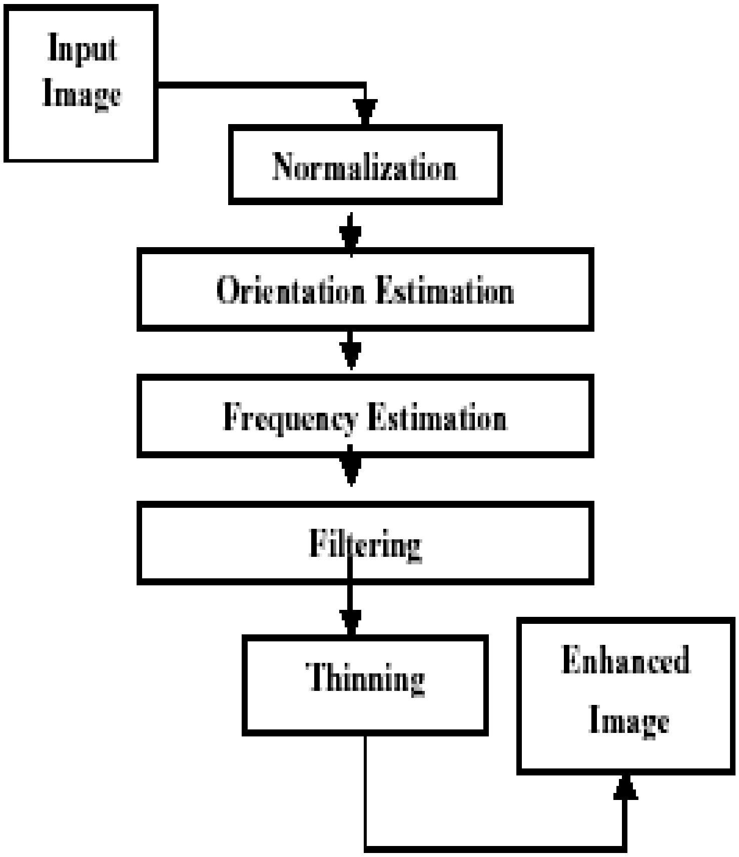


Fig 4.2: Block diagram for fingerprint enhancement

In practice, a fingerprint image may not always be well defined due to elements of noise that corrupt the clarity of the ridge structures. Thus, image enhancement techniques are often employed to reduce the noise and enhance the definition of ridges against valleys. Figure 4.2 illustrates the different steps involved in the development of the Enhancement Finger print. The details of these steps are given in the following subsections.

Step 3: Normalization

Normalization is used to standardize the intensity values in an image by adjusting the range of gray-level values so that it lies within a desired range of values. It does not change the ridge structures in a fingerprint; it is performed to standardize the dynamic levels of variation in gray-level values, which facilitates the processing of subsequent image enhancement stages. Fig. 4.3(a & b) shows the original fingerprint & the results of a normalized fingerprint.

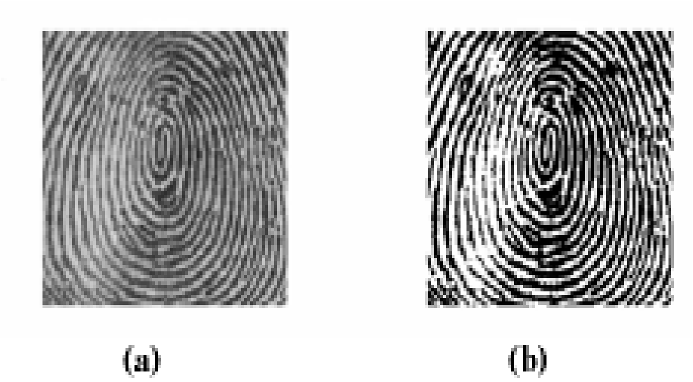


Fig 4.3 (a) Original Image (b) Normalized Image

Step 4: Orientation Estimation

The orientation field of a fingerprint image defines the local orientation of the ridges contained in the fingerprint (see Fig.4.4). The orientation estimation is a fundamental step in the enhancement process as the subsequent Gabor filtering stage relies on the local orientation in order to effectively enhances the fingerprint image. Fig. 4.5(a & b) illustrates the results of orientation estimation & smoothed orientation estimation of the fingerprint image.

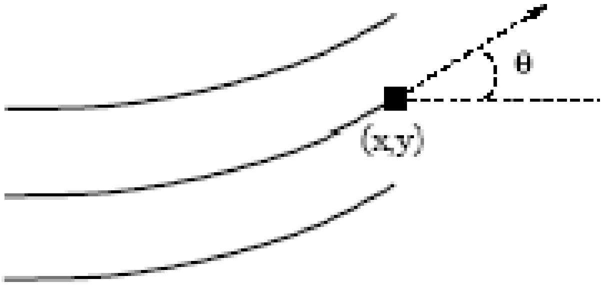


Fig 4.4: The orientation of a ridge pixel in a fingerprint

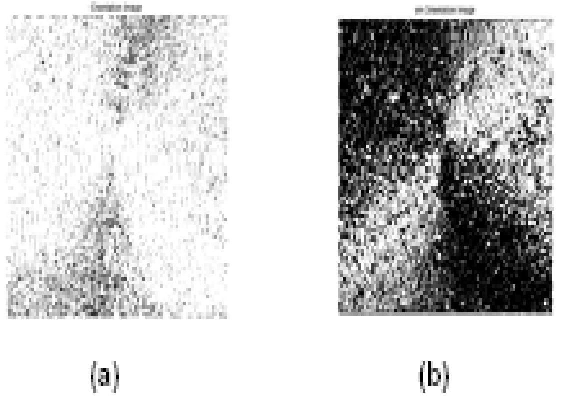


Fig: 4.5 (a)Orientation image (b)Smoothed orientation image

Step 5: Frequency Estimation

In addition to the orientation image, another important parameter that is used in the construction of the Gabor filter is the local ridge frequency. The frequency image represents the local frequency of the ridges in a fingerprint. Fig. shows the results of the local frequency estimation.

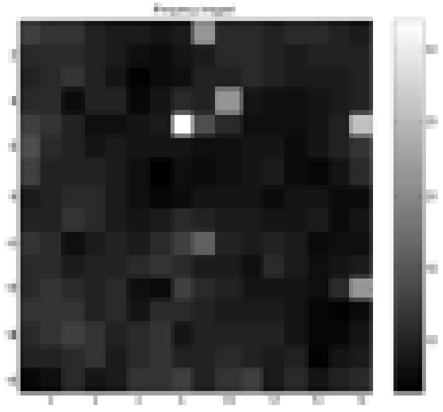


Fig 4.6: Frequency Image

Step 6: Gabor Filtering

Once the ridge orientation and ridge frequency information has been determined, these parameters are used to construct the even-symmetric Gabor filter. Gabor filters are employed because they have frequency-selective and orientationselective properties. These properties allow the filter to be tuned to give maximal response to ridges at a specific orientation and frequency in the fingerprint image. Therefore, a properly tuned Gabor filter can be used to effectively preserve the ridge structures while reducing noise. An even symmetric Gabor filter in the spatial domain is defined as,

$$G(x,y;q,f) = \exp\left\{-\frac{1}{2}\left[\frac{x_q^2}{s_x^2}+\frac{y_q^2}{s_y^2}\right]\right\}\cos(2pfx_q),$$

$$x_q = x\cos q + y\sin q$$

$$y_q = -x\sin q + y\cos q$$

where ɵ is the orientation of the Gabor filter, f is the frequency of the cosine wave, ɵxand ɵyare the standard deviations of the Gaussian envelope along the x and y axes, respectively, and xɵ and yɵ define the x and y-axes of the filter coordinate frame, respectively. Fig 4.7 illustrates the results of using gabor filter to a fingerprint image.

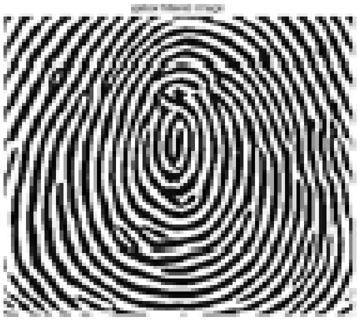


Fig 4.7:Filtered Image

Step 7:Thinning

The final image enhancement step typically performed prior to minutiae extraction is thinning. Thinning is a morphological operation that successively erodes away the foreground pixels until they are one pixel wide. The application of the thinning algorithm to a fingerprint image preserves the connectivity of the ridge structures while forming a skeleton version of the binary image. This skeleton image is then used in the subsequent extraction of minutiae.

The process involving the extraction of minutiae from a skeleton image will be discussed in the next section. Fig. 4.8 illustrates the results of thinning to a fingerprint image.

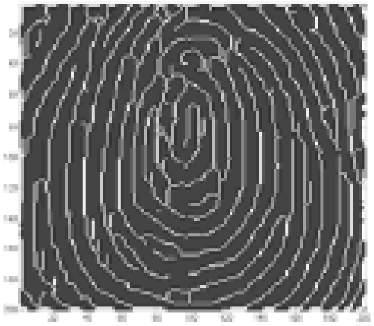


Fig 4.8:Thinned Image

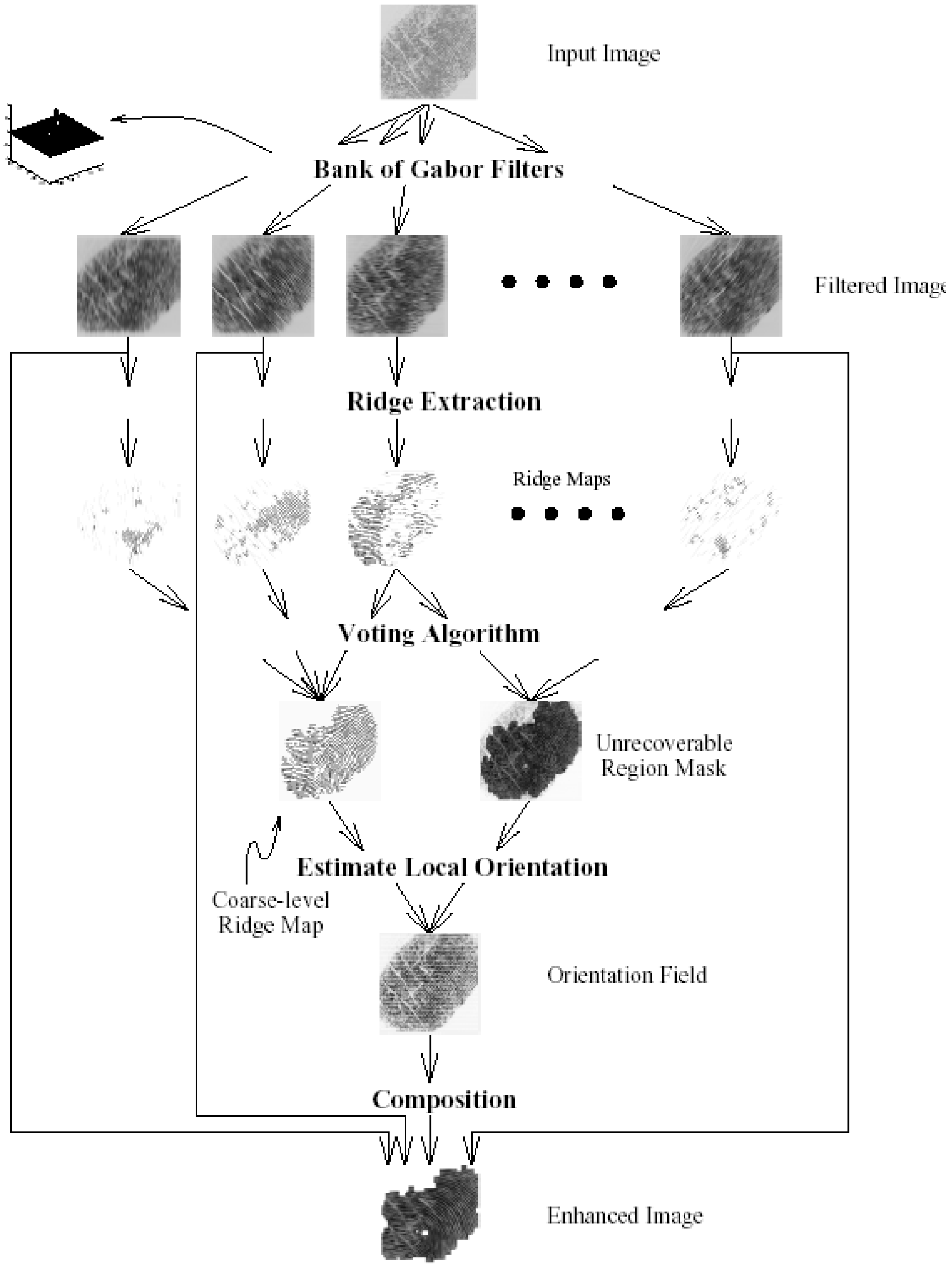


Fig:Fingerprint enhancement algorithm

Fingerprint enhancement and recognition

A while ago, I posted some pretty pictures of my toe when I figured out the image format used by the Digital Persona and Microsoft fingerprint readers.

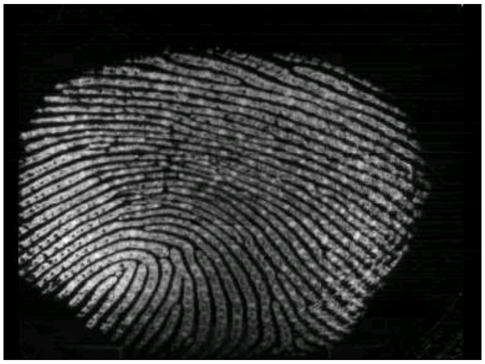
While it's pretty cool to see your fingerprint on-screen, the real question is how do we make use of these prints? We need a way of storing fingerprints, and a way of saying "does this fingerprint equal the one we stored earlier?" From that point, we can implement fingerprint-based login and other things.

There are various open-source projects aiming to do this kind of thing, I made a list of them here. Unfortunately all of them appear to be dead projects and most of them aren't useful at all, but I've made progress with one of them at least: FVS.

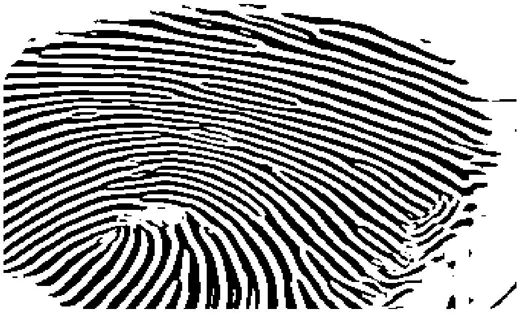
There are various different algorithms which can be used to compare fingerprints. I'll try to describe the method used by FVS: *minutiae detection*.

A fingerprint is made up of ridges, basically just the curvy lines which you see. Ridges start and finish and some of them split (bifurcate) into 2 other ridges. The points where ridge endings and bifurcations happen are known as minutiae. We can compare the positions and directions of the minutiae on two fingerprint images to decide whether they are equal. This is certainly throwing a lot of information away, but this method is very widely used in the fingerprint recognition world.

We start with the initial toe-print. I actually cheated by subtracting an image seen by the sensor before I scanned my toe from the toe-print, so what you can see below is slightly enhanced (clearer) than the original image.

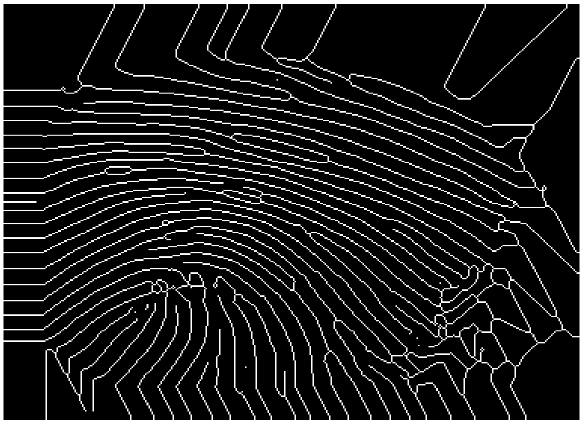


The *ridges* **of the fingerprint are visible above in white. The enhancement step involves finding the ridge direction and the ridge frequency. These details can then be used to apply aGabor filter to the image, which produces a greatly enhanced version**

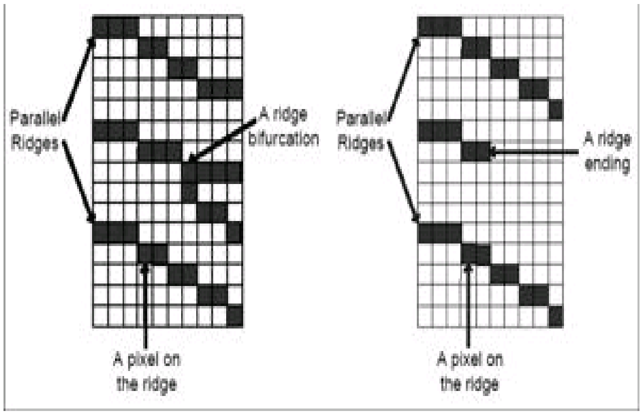


The above enhancement takes a few seconds on my system. This is OK for prototyping but is too slow for a real fingerprint login system, I hope we can find ways to optimize this. In addition to the Gabor filter, the image was further enhanced by binarization: all pixels are either black or white, no noise.

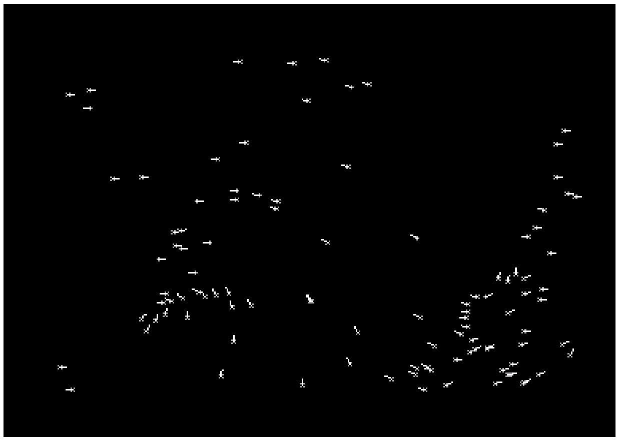
The ridges are now shown in black on a white background. The next step is to reduce each ridge line to a single pixel in width. This is known as *thinning*.



The advantage of thinning is that minutiae are now really easy to detect. We take every pixel on the image, and we ignore it if it is not a ridge (i.e. if it is white). For all the ridge pixels, we count the number of adjacent ridge pixels. If there is only one ridge pixel neighbour, we have found a ridge ending. If there are three ridge pixel neighbours, we have found a bifurcation.



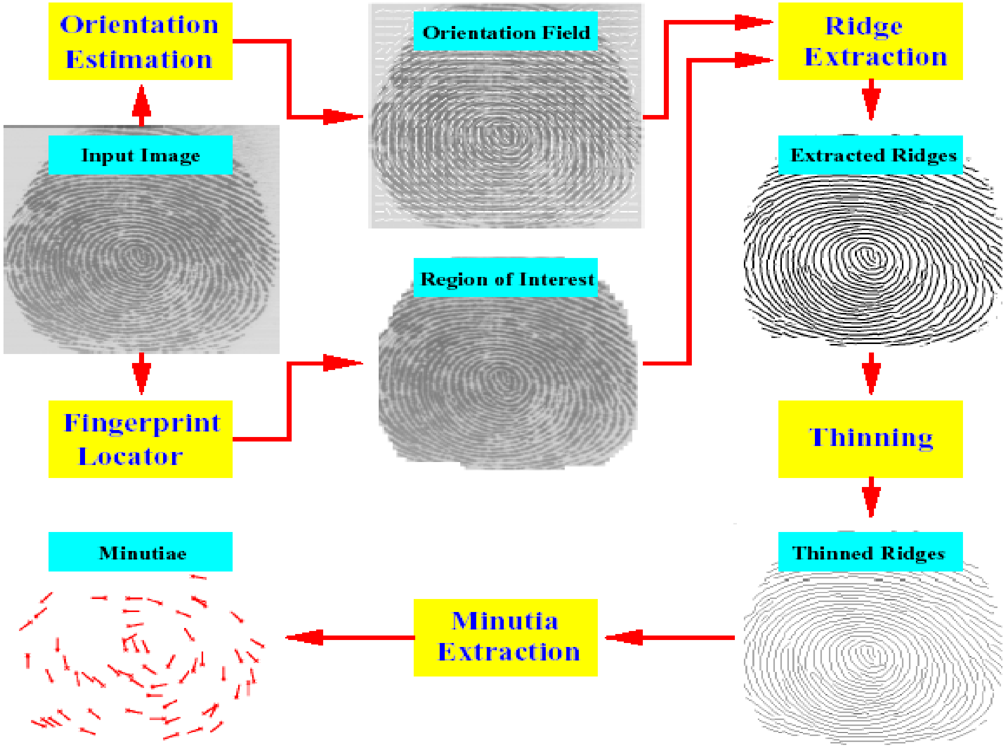
FVS includes minutiae detection code based on the above algorithm:



The code isn’t perfect, as it detected many minutiae around the edge of the fingerprint image, where they do not exist. However it should be relatively simple to exclude those as FVS already knows about the edges of the print.

The next challenge is to compare two minutiae sets and decide how similar they are. FVS includes some code to do this, but it just crashes, and I haven’t spent much time debugging it yet. This is a difficult operation: prints of the same finger are never identical: sometimes some minutiae are not visible, they can be spaced slightly differently, and the finger might even be significantly rotated since the last print.

A project called efinger has built a complete fingerprint recognition database. It uses FVS’s enhancement code, but ships it’s own code for thinning, minutiae detection, and minutiae set comparison. The code is not brilliant (does not consider rotation or anything like that) but should provide a good starting point.



LCD

Interfacing to LCD Display

Liquid Crystal Display also called as LCD is very helpful in providing user interface as well as for debugging purpose. The most common type of LCD controller is HITACHI 44780 which provides a simple interface between the controller & an LCD. These LCD's are very simple to interface with the controller as well as are cost effective.

The most commonly used *ALPHANUMERIC* displays are *1x16* (Single Line & 16 characters), *2x16* (Double Line & 16 character per line) &*4x20* (four lines & Twenty characters per line).

The LCD requires 3 control lines (RS, R/W & EN) & 8 (or 4) data lines. The number on data lines depends on the mode of operation. If operated in 8-bit mode then 8 data lines + 3 control lines i.e. total 11 lines are required. And if operated in 4-bit mode then 4 data lines + 3 control lines i.e. 7 lines are required. How do we decide which mode to use? It's simple if you have sufficient data lines you can go for 8 bit mode & if there is a time constrain i.e. display should be faster then we have to use 8-bit mode because basically 4-bit mode takes twice as more time as compared to 8-bit mode.

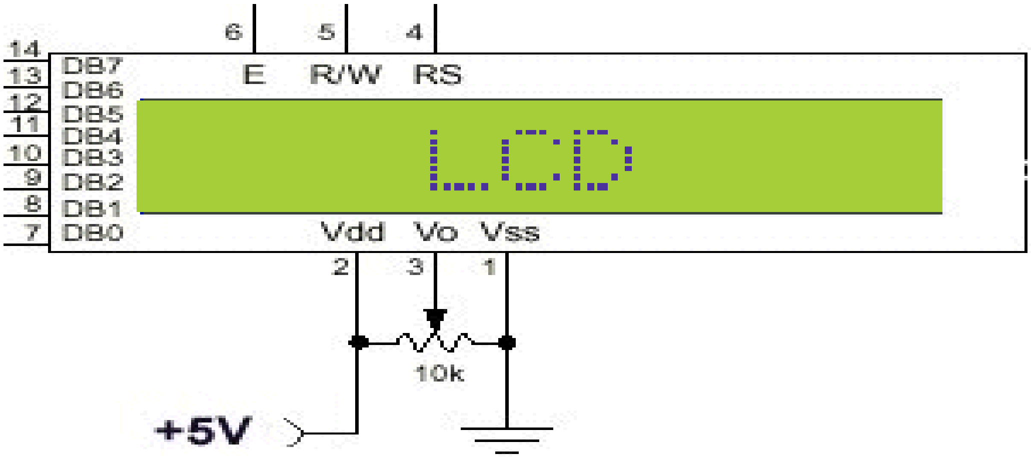
Most projects you create with the 8051 CPU require some form of display. The most common way to accomplish this is with the LCD (Liquid Crystal Display). LCDs have become a cheap and easy way to get text display for embedded system Common displays are set up as 16 to 20 characters by 1 to 4 lines.

When *RS* is low (0), the data is to be treated as a command. When RS is high (1), the data being sent is considered as text data which should be displayed on the screen.

When *R/W* is low (0), the information on the data bus is being written to the LCD. When RW is high (1), the program is effectively reading from the

LCD. Most of the times there is no need to read from the LCD so this line can directly be connected to Gnd thus saving one controller line.

The *ENABLE* pin is used to latch the data present on the data pins. A HIGH - LOW signal is required to latch the data. The LCD interprets and executes our command at the instant the EN line is brought low. If you never bring EN low, your instruction will never be executed.



UNDERSTANDING LCD:

Pin outline:

• 8 data pins D7:D0

Bi-directional data/command pins.

Alphanumeric characters are sent in ASCII format.

• RS: Register Select

RS = 0 -> Command Register is selected

RS = 1 -> Data Register is selected

• R/W: Read or Write

0 -> Write, 1 -> Read

• E: Enable (Latch data)

Used to latch the data present on the data pins.

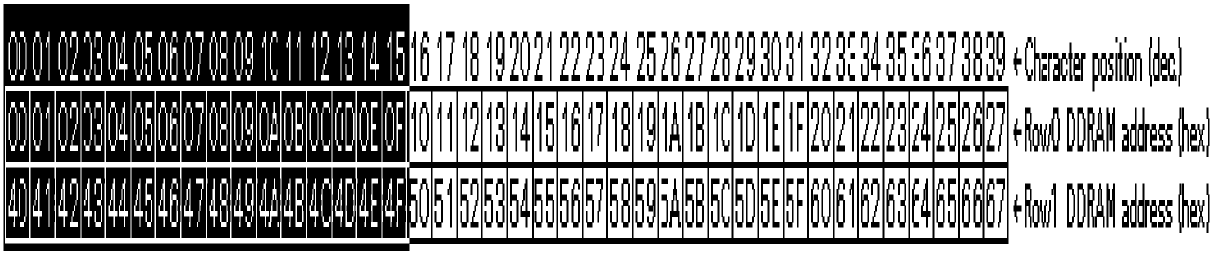
A high-to-low edge is needed to latch the data.

• VEE : contrast control

NOTE: When writing to the display, data is transferred only on the high to low transition of this signal. However, when reading from the display, data will become available shortly after the low to high transition and remain available until the signal falls low again.

Display Data RAM (DDRAM)

Display data RAM (DDRAM) is where you send the characters (ASCII code) you want to see on the LCD screen. It stores display data represented in 8-bit character codes. Its capacity is 80 characters (bytes). Below you see DD RAM address layout of a 2*16 LCD.



00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	← Character position (dec.)
00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13	← Row0 DDRAM address (hex)
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53	← Row1 DDRAM address (hex)

20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	← Character position (dec.)
14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20	21	22	23	24	25	26	27	← Row0 DDRAM address (hex)
54	55	56	57	58	59	5A	5B	5C	5D	5E	5F	60	61	62	63	64	65	66	67	← Row1 DDRAM address (hex)

In the above memory map, the area shaded in black is the visible display (For 16x2 displays). For first line addresses for first 15 characters is from 00h to 0Fh. But for second line address of first character is 40h and so on up to 4Fh for the 16th character.

So if you want to display the text at specific positions of LCD , we require to manipulate address and then to set cursor position accordingly .

Character Generator RAM (CGRAM)-User defined character RAM

In the character generator RAM, we can define our own character patterns by program. CG RAM is 64 bytes ,allowing for eight 5*8 pixel, character patterns to be defined. However how to define this and use it is out of scope of this tutorial. So I will not talk any more about CGRAM

Registers

The HD44780 has two 8-bit registers, an instruction register (IR) and a data register (DR). The IR stores instruction codes. The DR temporarily stores data to be written into DDRAM or CGRAM and temporarily stores data to be read from DDRAM or CGRAM. Data written into the DR is automatically written into DDRAM or CGRAM by an internal operation. . These two registers can be selected by the register selector (RS) signal. See the table below:

		Register Selection
RS	R/W	Operation
0	0	IR write as an internal operation (display clear, etc.)
0	1	Read busy flag (DB7) and address counter (DB0 to DB6)
1	0	DR write as an internal operation (DR to DDRAM or CGRAM)
1	1	DR read as an internal operation (DDRAM or CGRAM to DR)

Busy Flag (BF)

When the busy flag is 1, the LCD is in the internal operation mode, and the next instruction will not be accepted. When RS = 0 and R/W = 1 (see the table above), the busy flag is output to DB7 (MSB of LCD data bus). The next instruction must be written after ensuring that the busy flag is 0.

LCD Commands

The LCD's internal controller accept several commands and modify the display accordingly. These commands would be things like:

– Clear screen

– Return home

– Shift display right/left

Instruction	Decimal	HEX
Function set (8-bit interface, 2 lines, 5*7 Pixels)	56	38
Function set (8-bit interface, 1 line, 5*7 Pixels)	48	30
Function set (4-bit interface, 2 lines, 5*7 Pixels)	40	28
Function set (4-bit interface, 1 line, 5*7 Pixels)	32	20
Entry mode set	See Below	See Below
Scroll display one character right (all lines)	28	1E
Scroll display one character left (all lines)	24	18
Home (move cursor to top/left character position)	2	2
Move cursor one character left	16	10
Move cursor one character right	20	14
Turn on visible underline cursor	14	0E
Turn on visible blinking-block cursor	15	0F
Make cursor invisible	12	0C
Blank the display (without clearing)	8	08
Restore the display (with cursor hidden)	12	0C
Clear Screen	1	01
Set cursor position (DDRAM address)	128 + addr	80+ addr
Set pointer in character-generator RAM (CG RAM address)	64 + addr	40+ addr

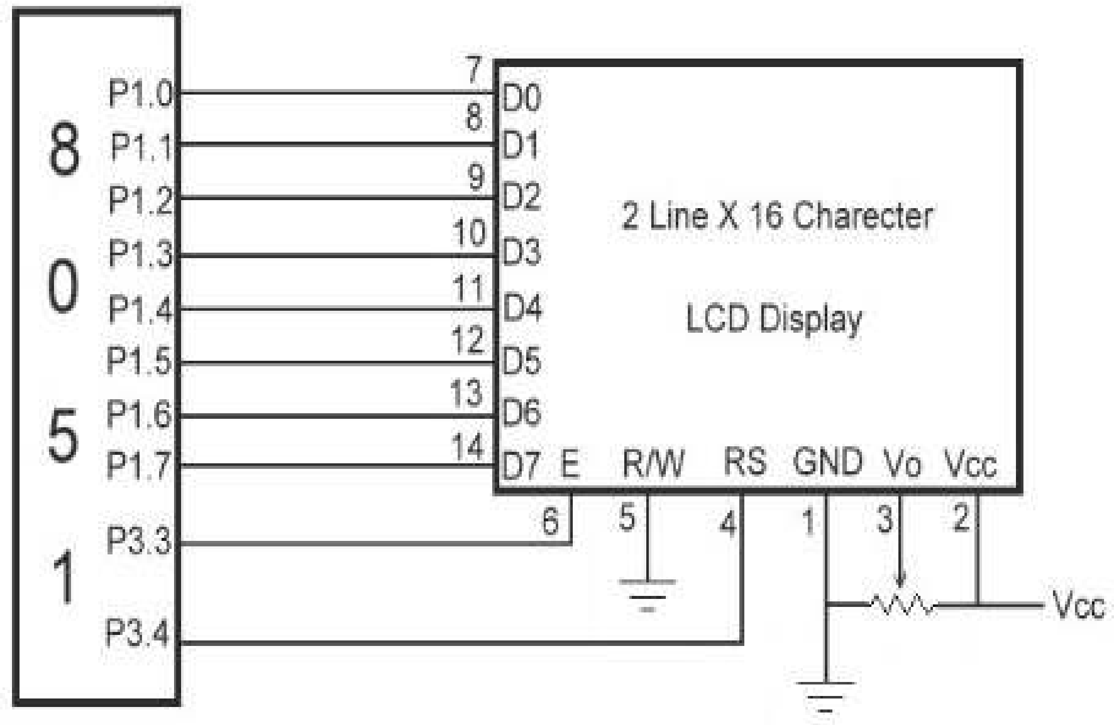
Entry mode set

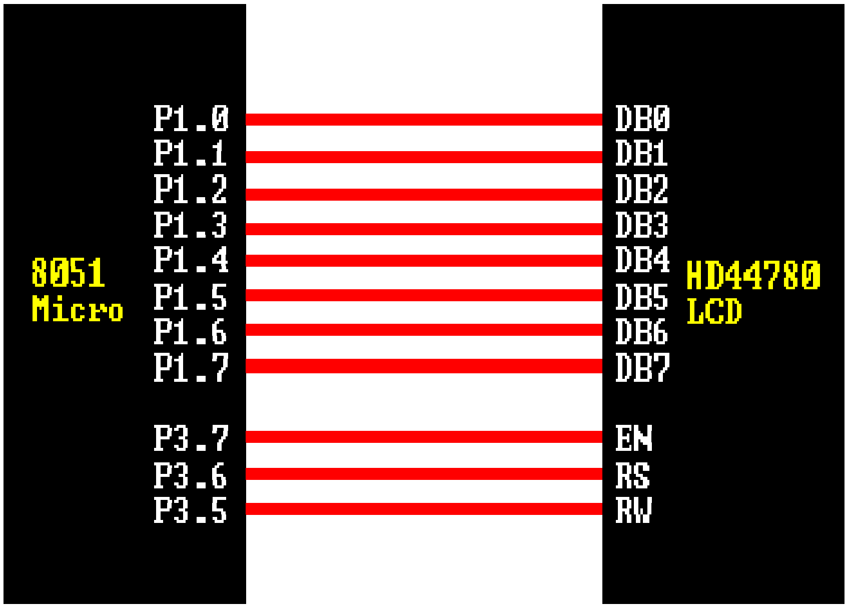
This command sets cursor move direction and display shift ON/OFF. There are 4 possible function set commands;04, 05, 06, and 07. This command changes the direction the cursor moves by setting the address counter to increment or decrement. This command is very important. If you do not understand it you may not see anything or what you actually wanted to see on LCD screen. I have created 4 animated gifs to demonstrate what the function set command is all about.

Set cursor position (DDRAM address)

As said earlier if we want to display the text at specific positions of LCD , we require to manipulate address and then to set cursor position accordingly.
I want to display "MAHESH" in message "Hi MAHESH" at the right corner of first line then I should start from 10th character.
So referring to table 80h+0Ah= 8Ah.

INTERFACING LCD TO 8051 :


8051
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.3
P3.4
7
8
9
10
11
12
13
14
D0
D1
D2
D3
D4
D5
D6
D7
E
R/W
RS
GND
Vo
Vcc
6
5
4
1
3
2
2 Line X 16 Charecter
LCD Display
Vcc


8051
Micro
P1.0
P1.1
P1.2
P1.3
P1.4
P1.5
P1.6
P1.7
P3.7
P3.6
P3.5
DB0
DB1
DB2
DB3
DB4
DB5
DB6
DB7
EN
RS
RW
HD44780
LCD

The 44780 standard requires 3 control lines as well as either 4 or 8 I/O lines for the data bus. The user may select whether the LCD is to operate with a 4-bit data bus or an 8-bit data bus.

If a 4-bit data bus is used, the LCD will require a total of 7 data lines.
If an 8-bit data bus is used, the LCD will require a total of 11 data lines.

The three control lines are **EN**, **RS**, and **RW**.

Note that the EN line must be raised/lowered before/after each instruction sent to the LCD regardless of whether that instruction is read or write, text or instruction. In short, you must always manipulate EN when communicating with the LCD. EN is the LCD's way of knowing that you are talking to it. If you don't raise/lower EN, the LCD doesn't know you're talking to it on the other lines.

Checking the Busy Flag

You can use subroutine for checking busy flag or just a big (and safe) delay.

1. Set R/W Pin of the LCD HIGH(read from the LCD)
2. Select the instruction register by setting RS pin LOW
3. Enable the LCD by Setting the enable pin HIGH
4. The most significant bit of the LCD data bus is the state of the busy flag(1=Busy,0=ready to accept instructions/data). The other bits hold the current value of the address counter.

If the LCD never come out from "busy" status because of some problems ,The program will "hang," waiting for DB7 to go low. So in a real applications it would be wise to put some kind of time limit on the delay--for example, a maximum of 100 attempts to wait for the busy signal to go low. This would guarantee that even if the LCD hardware fails, the program would not lock up.

DC Motor

DC stands for "direct current". A DC motor is an electric motor that uses electricity and a magnetic field to produce torque, which turns the DC motor. A DC motor consists of two magnets of opposite polarity and an electric coil. When a power supply is added to the coil electric current flows through in a circuit and generates a small magnetic field. The repellent and attractive electromagnetic forces of the magnets provide the torque that causes the armature to turn.

How it Works

Magnets are polarized, with a positive and a negative side. A DC motor uses the attraction between opposite poles and the repulsion of like poles to convert electric energy into kinetic energy. As the magnets within the DC motor attract and repel one another, the motor turns. The magnetic force on the armature works perpendicular to both wire and magnetic field. An electric switch called a commutator reverses the direction of the electric current in the armature twice every cycle. The poles of the electromagnet push and pull against the permanent magnets on the outside of the motor. As the poles of the armature electromagnet pass the poles of the permanent magnets, the commutator reverses the polarity of the armature electromagnet. During that instant of switching polarity, inertia keeps the motor going in the proper direction.

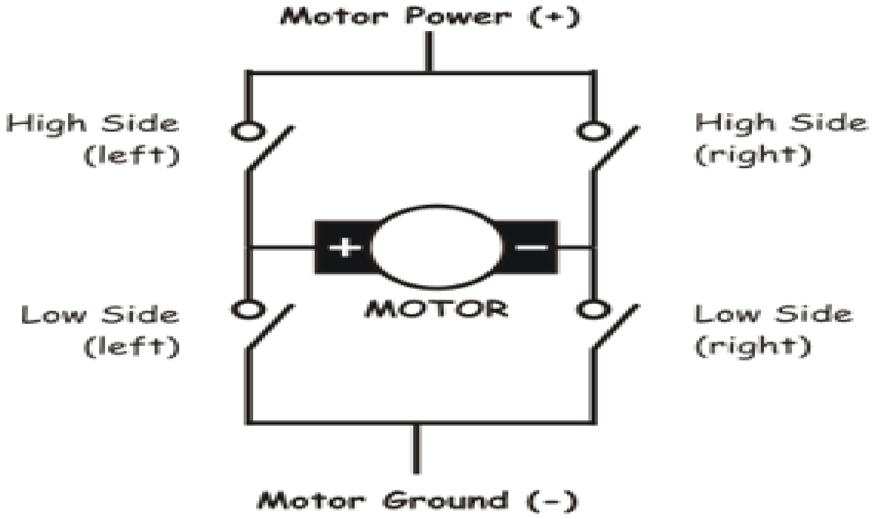
If an electric current goes through the coil, the motor will act like a generator and produce an electric motive force (EMF). When the motor spins it produces a voltage called the back EMF because it opposes the applied voltage on the motor. Therefore, the voltage drop across the motor consists of the voltage drop from the back EMF and the voltage drop from the internal resistance of the rotation of the armature.

DC motor interfacing with controller:

Usually H-bridge is preffered way of interfacing a DC motor. These days many IC manufacturers have H-bridge motor drivers available in the market like L293D is most used H-Bridge driver IC. H-bridge can also be made with the help of trasistors and MOSFETs etc. rather of being cheap, they only increase the size of the design board, which is sometimes not required so using a small 16 pin IC is preffered for this purpose.

H-BRIDGE:

The name "H-Bridge" is derived from the actual shape of the switching circuit which control the motoion of the motor. It is also known as "Full Bridge". Basically there are four switching elements in the H-Bridge as shown in the figure below.

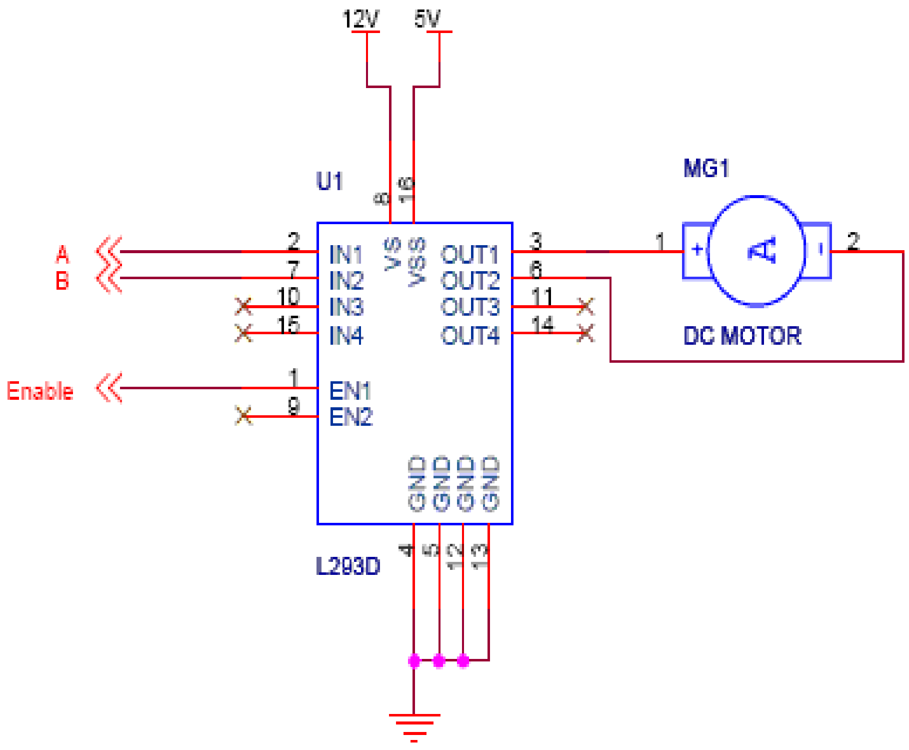


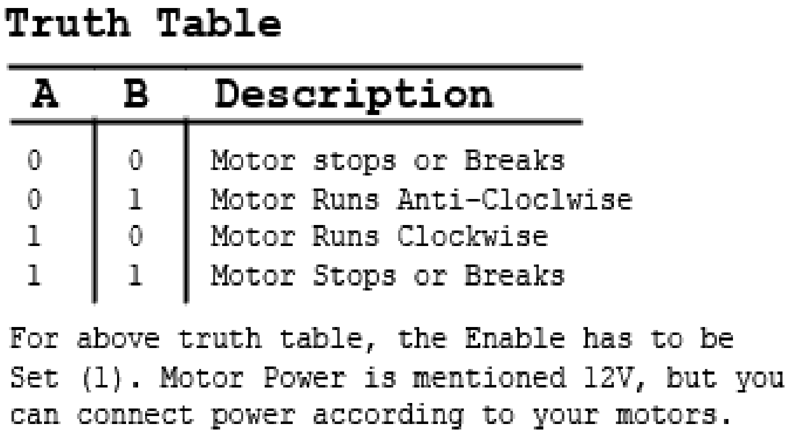
As you can see in the figure above there are four switching elements named as "High side left", "High side right", "Low side right", "Low side left". When these switches are turned on in pairs motor changes its direction accordingly. Like, if we switch on High side left and Low side right then motor rotate in forward direction, as current flows from Power supply through the motor coil goes to ground via switch low side right. This is shown in the figure below. Similarly, when you switch on low side left and high side right, the current flows in opposite direction and motor rotates in backward direction. This is the basic working of H-Bridge.

DualH-BridgeMotorDriver(L293D): :

L293D is a dual H-Bridge motor driver, So with one IC we can interface two DC motors which can be controlled in both clockwise and counter clockwise direction and if you have motorwith fix direction of motion the you can make use of all the four I/Os to connect up to four DC motors. L293D has output current of 600mA and peak output current of 1.2A per channel. Moreover for protection of circuit from back EMF ouput diodes are included within the IC. The output supply (VCC2) has a wide range from 4.5V to 36V, which has made L293D a best choice for DCmotor driver.

A simple schematic for interfacing a DCmotor using L293D is shown below.



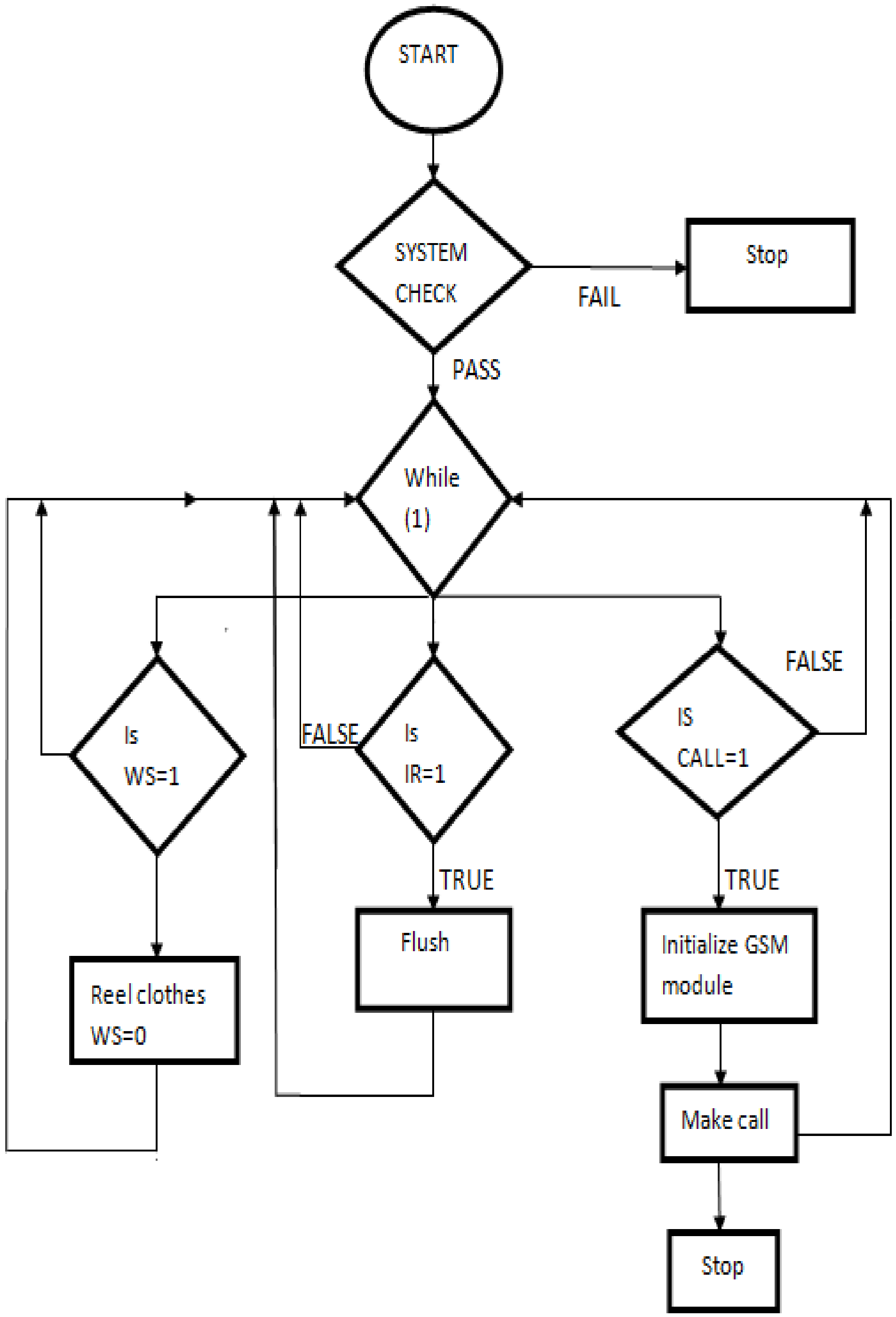


Truth Table

A	B	Description
0	0	Motor stops or Breaks
0	1	Motor Runs Anti-Cloclwise
1	0	Motor Runs Clockwise
1	1	Motor Stops or Breaks

For above truth table, the Enable has to be Set (1). Motor Power is mentioned 12V, but you can connect power according to your motors.

FLOW CHART:



Flow chart description:

1. System check access will activate the smart appliances in home by using tag
2. Here we use while loop, in order to access this smart appliances for “n” number of times.
3. If you want to make use of appliances, it is purely dependent on the situations.
    a. If water Sensor is high sensing that rainfall is occurring, it will indicate the dc motor to reel clothes inside the covered places.
    b. After the work of dc motor is done, then water sensor changes from state from 1 to 0.
    c. If IR sensor changes state from 0 to 1, then automatic flush is done after the usage of washroom.
    d. After some time interval, the IR will change to low state then automatic flush will switch to off mode.
    e. If call is in high state, then start initializing the GSM module and make a call to respective ones by taking inputs of keypad interface.
    f. After the call is disconnected, call variable is set back to 0 indicating the completion of call.

Buzzer

A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short). Typical uses of buzzers and beepers include alarm devices, timers, and confirmation of user input such as a mouse click or keystroke.

Battery

12v Lead Acid battery

The battery used for powering the systemisa12vLeadacid battery.Itisrechargeableandcanprovideupto1.3amperes inanhour.Thisisidealforcontrollingthemovementofthe 4 DC motors used in theproject.

Chapter 5

SOFTWARE

TOOL DESCRIPTION

Integrated Development Environment “IDE” For Arduino

Introduction to Arduino IDE

IDE stands for “Integrated Development Environment” :it is an official software introduced by Arduino.cc, that is mainly used for editing, compiling and uploading the code in the Arduino Device. Almost all Arduino modules are compatible with this software that is an open source and is readily available to install and start compiling the code on the go.

In this article, we will introduce the Software, how we can install it, and make it ready for developing applications using Arduino modules.

Arduino IDE Definition

1. Arduino IDE is an open source software that is mainly used for writing and compiling the code into the ArduinoModule.
2. It is an official Arduino software, making code compilation too easy that even a common person with no prior technical knowledge can get their feet wet with the learningprocess.
3. It is easily available for operating systems like MAC, Windows, Linux and runs on the Java Platform that comes with inbuilt functions and commands that play a vital role for debugging, editing and compiling the code in theenvironment.
4. A range of Arduino modules available including Arduino Uno, Arduino Mega, Arduino Leonardo, Arduino Micro and manymore.
5. Each of them contains a microcontroller on the board that is actually programmed and accepts the information in the form ofcode.
6. The main code, also known as a sketch, created on the IDE platform

will ultimately generate a Hex File which is then transferred and uploaded in the controller on theboard.

7. The IDE environment mainly contains two basic parts: Editor and Compiler where former is used for writing the required code and later is used for compiling and uploading the code into the given ArduinoModule.
8. This environment supports both C and C++languages.

How to get Arduino IDE

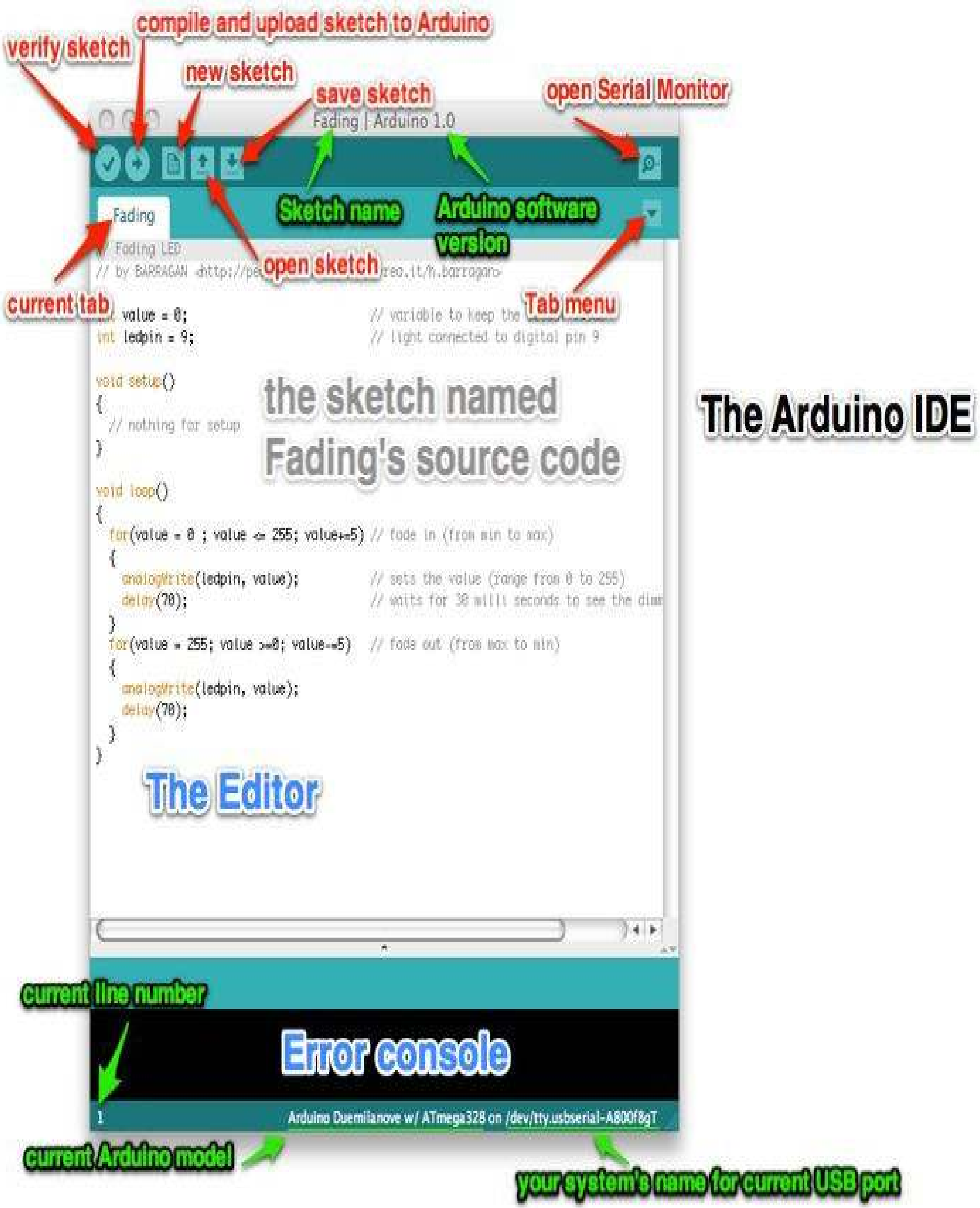
We can download the Software from Arduino main website. As I said earlier, the software is available for common operating systems like Linux, Windows, and MACos,we select to download the correct software version that is easily compatible with our operating system.

Details on IDE: The IDE environment is mainly distributed into three sections

- **1. MenuBar**
- **2. TextEditor**
- **3. OutputPane**

As we download and open the IDE software, it will appear like an image below.

Figure: IDE software



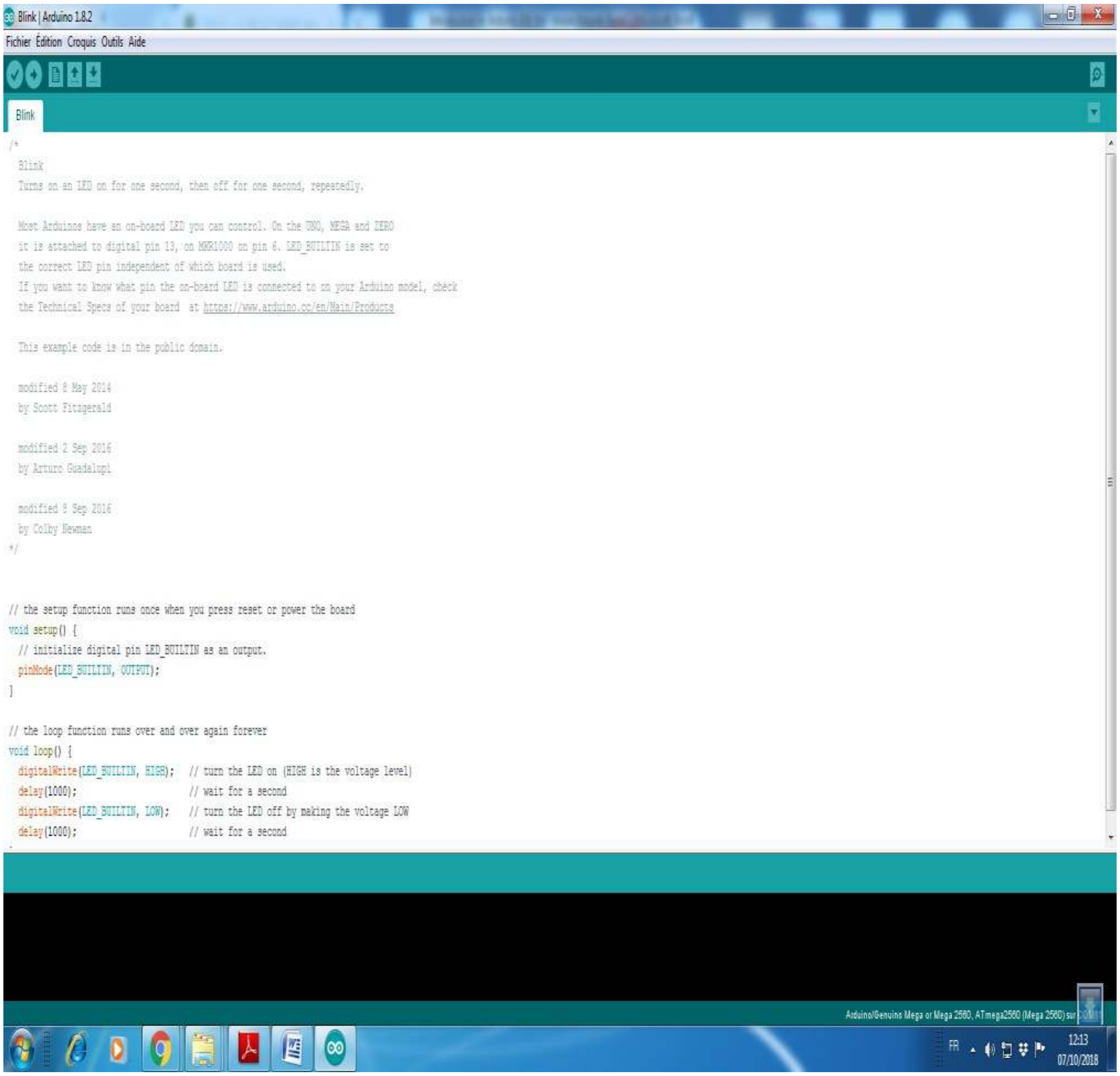
```
// Fading LED
// by BARRAGAN <http://pe                    reo.it/h.barragan>

  value = 0;                          // variable to keep the
int ledpin = 9;                       // light connected to digital pin 9

void setup()
{
  // nothing for setup
}

void loop()
{
  for(value = 0 ; value <= 255; value+=5) // fade in (from min to max)
  {
    analogWrite(ledpin, value);           // sets the value (range from 0 to 255)
    delay(70);                            // waits for 30 milli seconds to see the dim
  }
  for(value = 255; value >=0; value-=5)   // fade out (from max to min)
  {
    analogWrite(ledpin, value);
    delay(70);
  }
}
```

Arduino Duemilanove w/ ATmega328 on /dev/tty.usbserial-A800f8gT

The bar appearing on the top is called **Menu Bar** that comes with five different options as follow

- **File** – You can open a new window for writing the code or open an existing one. Following table shows the number of further subdivisions the file option is categorizedinto.

Figure: Arduino Basic example code.


Blink | Arduino 1.8.2
Fichier Édition Croquis Outils Aide
Blink
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) sur COM1
FR
12:13
07/10/2018

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
  it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
  the correct LED pin independent of which board is used.
  If you want to know what pin the on-board LED is connected to on your Arduino model, check
  the Technical Specs of your board  at https://www.arduino.cc/en/Main/Products

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald

  modified 2 Sep 2016
  by Arturo Guadalupi

  modified 8 Sep 2016
  by Colby Newman
*/


// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
```

And at the end of compilation, it will show you the hex file it has generated for the recent sketch that will send to the Arduino Board for the specific task you aim to achieve.

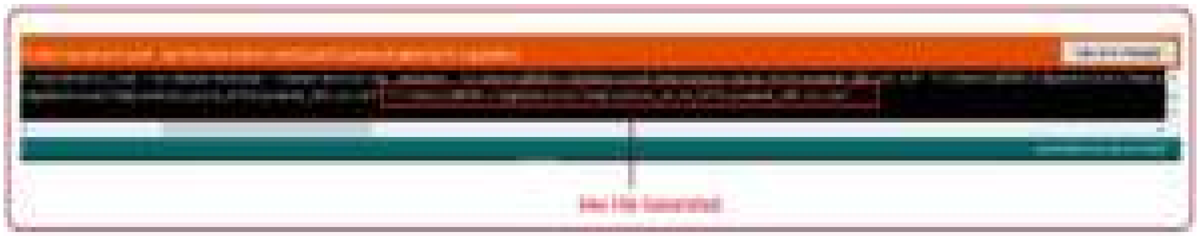


Figure: Hex file format.

- **Edit** – Used for copying and pasting the code with further modification forfont
- **Sketch** – For compiling andprogramming
- **Tools** – Mainly used for testing projects. The Programmer section in this panel is used for burning a bootloader to the newmicrocontroller.
- **Help** – In case you are feeling skeptical about software, complete help is available from gettingstarted to troubleshooting.

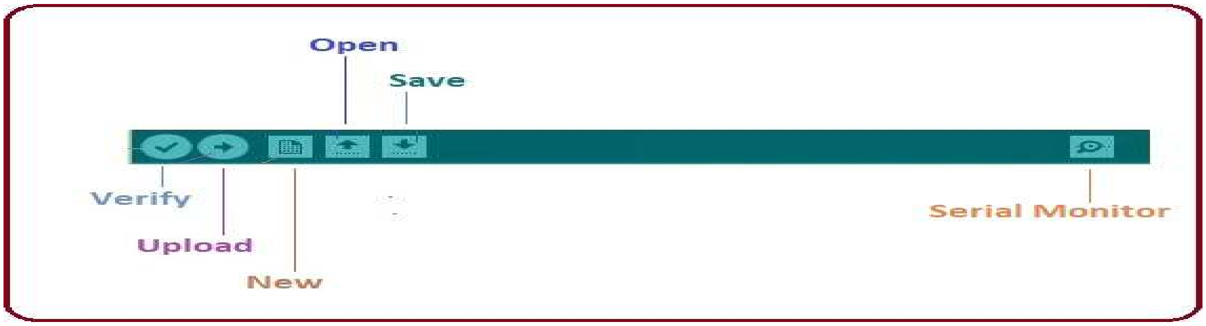
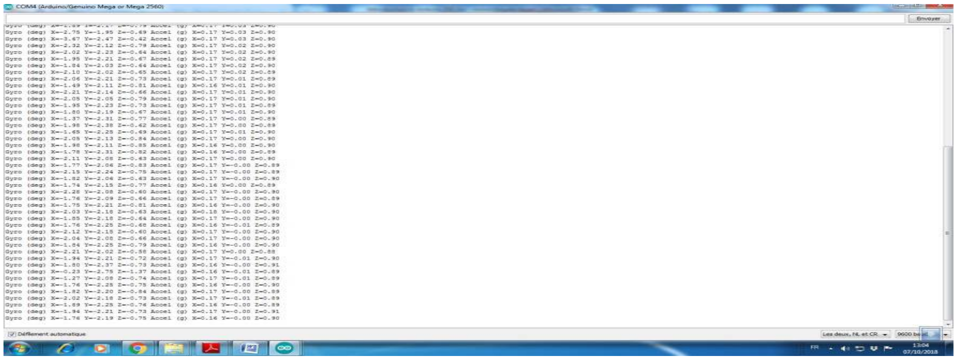


Figure: The **Six Buttons** appearing under the Menu tab are connected with the running program as follow.

- The check mark appearing in the circular button is used to verify the code.

Click this once you have writtenyourcode.

- The arrow key will upload and transfer the required code to the Arduinoboard.
- The dotted paper is used for creating a newfile.
- The upward arrow is reserved for opening an existing Arduinoproject.
- The downward arrow is used to save the current runningcode.
- The button appearing on the top right corner is a **Serial Monitor** – A separate pop-up window that acts as an independent terminal and plays a vital role for sending and receiving the Serial Data. You can also go to the Tools panel and select Serial Monitor pressing Ctrl+Shift+M all at once will open the Serial Monitor. The Serial Monitor will actually help to debug the written Sketches where you can get a hold of how your program is operating. Your Arduino Module should be connected to your computer by USB cable in order to activate the SerialMonitor.
- You need to select the baud rate of the Arduino Board you are using right now. For my Arduino Uno Baud Rate is 9600, as you right the following code and click the Serial Monitor, the output will show as the imagebelow.



COM4 (Arduino/Genuino Mega or Mega 2560)

```
Gyro (deg) X=-2.75 Y=-1.95 Z=-0.69 Accel (g) X=0.17 Y=0.03 Z=0.90
Gyro (deg) X=-3.67 Y=-2.47 Z=-0.42 Accel (g) X=0.17 Y=0.03 Z=0.90
Gyro (deg) X=-2.32 Y=-2.12 Z=-0.79 Accel (g) X=0.17 Y=0.02 Z=0.90
Gyro (deg) X=-2.02 Y=-2.23 Z=-0.64 Accel (g) X=0.17 Y=0.02 Z=0.90
Gyro (deg) X=-1.95 Y=-2.21 Z=-0.67 Accel (g) X=0.17 Y=0.02 Z=0.89
Gyro (deg) X=-1.84 Y=-2.03 Z=-0.64 Accel (g) X=0.17 Y=0.02 Z=0.90
Gyro (deg) X=-2.10 Y=-2.02 Z=-0.65 Accel (g) X=0.17 Y=0.02 Z=0.89
Gyro (deg) X=-2.06 Y=-2.21 Z=-0.73 Accel (g) X=0.17 Y=0.01 Z=0.89
Gyro (deg) X=-1.49 Y=-2.11 Z=-0.81 Accel (g) X=0.16 Y=0.01 Z=0.90
Gyro (deg) X=-2.21 Y=-2.14 Z=-0.66 Accel (g) X=0.17 Y=0.01 Z=0.90
Gyro (deg) X=-2.05 Y=-2.05 Z=-0.79 Accel (g) X=0.17 Y=0.01 Z=0.90
Gyro (deg) X=-1.95 Y=-2.23 Z=-0.73 Accel (g) X=0.17 Y=0.01 Z=0.89
Gyro (deg) X=-1.80 Y=-2.19 Z=-0.67 Accel (g) X=0.17 Y=0.01 Z=0.90
Gyro (deg) X=-1.37 Y=-2.31 Z=-0.77 Accel (g) X=0.17 Y=0.00 Z=0.89
Gyro (deg) X=-1.98 Y=-2.38 Z=-0.62 Accel (g) X=0.17 Y=0.00 Z=0.89
Gyro (deg) X=-1.65 Y=-2.25 Z=-0.69 Accel (g) X=0.17 Y=0.01 Z=0.90
Gyro (deg) X=-2.05 Y=-2.13 Z=-0.84 Accel (g) X=0.17 Y=0.00 Z=0.90
Gyro (deg) X=-1.98 Y=-2.11 Z=-0.85 Accel (g) X=0.16 Y=0.00 Z=0.90
Gyro (deg) X=-1.78 Y=-2.31 Z=-0.82 Accel (g) X=0.16 Y=0.00 Z=0.89
Gyro (deg) X=-2.11 Y=-2.08 Z=-0.63 Accel (g) X=0.17 Y=0.00 Z=0.90
Gyro (deg) X=-1.77 Y=-2.06 Z=-0.83 Accel (g) X=0.17 Y=-0.00 Z=0.89
Gyro (deg) X=-2.15 Y=-2.24 Z=-0.75 Accel (g) X=0.17 Y=-0.00 Z=0.89
Gyro (deg) X=-1.82 Y=-2.06 Z=-0.63 Accel (g) X=0.17 Y=-0.00 Z=0.90
Gyro (deg) X=-1.74 Y=-2.15 Z=-0.77 Accel (g) X=0.16 Y=0.00 Z=0.89
Gyro (deg) X=-2.28 Y=-2.08 Z=-0.60 Accel (g) X=0.17 Y=-0.00 Z=0.90
Gyro (deg) X=-1.76 Y=-2.09 Z=-0.66 Accel (g) X=0.17 Y=-0.00 Z=0.89
Gyro (deg) X=-1.75 Y=-2.21 Z=-0.81 Accel (g) X=0.16 Y=-0.00 Z=0.90
Gyro (deg) X=-2.03 Y=-2.18 Z=-0.63 Accel (g) X=0.18 Y=-0.00 Z=0.90
Gyro (deg) X=-1.85 Y=-2.18 Z=-0.64 Accel (g) X=0.17 Y=-0.00 Z=0.90
Gyro (deg) X=-1.76 Y=-2.25 Z=-0.68 Accel (g) X=0.16 Y=-0.01 Z=0.89
Gyro (deg) X=-2.12 Y=-2.15 Z=-0.60 Accel (g) X=0.17 Y=-0.00 Z=0.90
Gyro (deg) X=-2.04 Y=-2.08 Z=-0.66 Accel (g) X=0.17 Y=-0.00 Z=0.90
Gyro (deg) X=-1.84 Y=-2.25 Z=-0.79 Accel (g) X=0.16 Y=-0.00 Z=0.90
Gyro (deg) X=-2.21 Y=-2.02 Z=-0.58 Accel (g) X=0.17 Y=0.00 Z=0.88
Gyro (deg) X=-1.94 Y=-2.21 Z=-0.72 Accel (g) X=0.17 Y=-0.01 Z=0.90
Gyro (deg) X=-1.80 Y=-2.37 Z=-0.73 Accel (g) X=0.16 Y=-0.00 Z=0.91
Gyro (deg) X=-0.23 Y=-2.75 Z=-1.37 Accel (g) X=0.16 Y=-0.01 Z=0.89
Gyro (deg) X=-1.27 Y=-2.08 Z=-0.74 Accel (g) X=0.17 Y=-0.01 Z=0.89
Gyro (deg) X=-1.76 Y=-2.25 Z=-0.75 Accel (g) X=0.16 Y=-0.00 Z=0.90
Gyro (deg) X=-1.82 Y=-2.20 Z=-0.84 Accel (g) X=0.17 Y=-0.00 Z=0.89
Gyro (deg) X=-2.02 Y=-2.18 Z=-0.73 Accel (g) X=0.17 Y=-0.01 Z=0.89
Gyro (deg) X=-1.89 Y=-2.25 Z=-0.76 Accel (g) X=0.16 Y=-0.00 Z=0.89
Gyro (deg) X=-1.94 Y=-2.21 Z=-0.73 Accel (g) X=0.17 Y=-0.00 Z=0.91
Gyro (deg) X=-1.76 Y=-2.19 Z=-0.75 Accel (g) X=0.16 Y=-0.00 Z=0.90
```

☑ Défilement automatique

Figure: Serial monitor window.

The main screen below show how to select a program from examples ie: Bink.cc

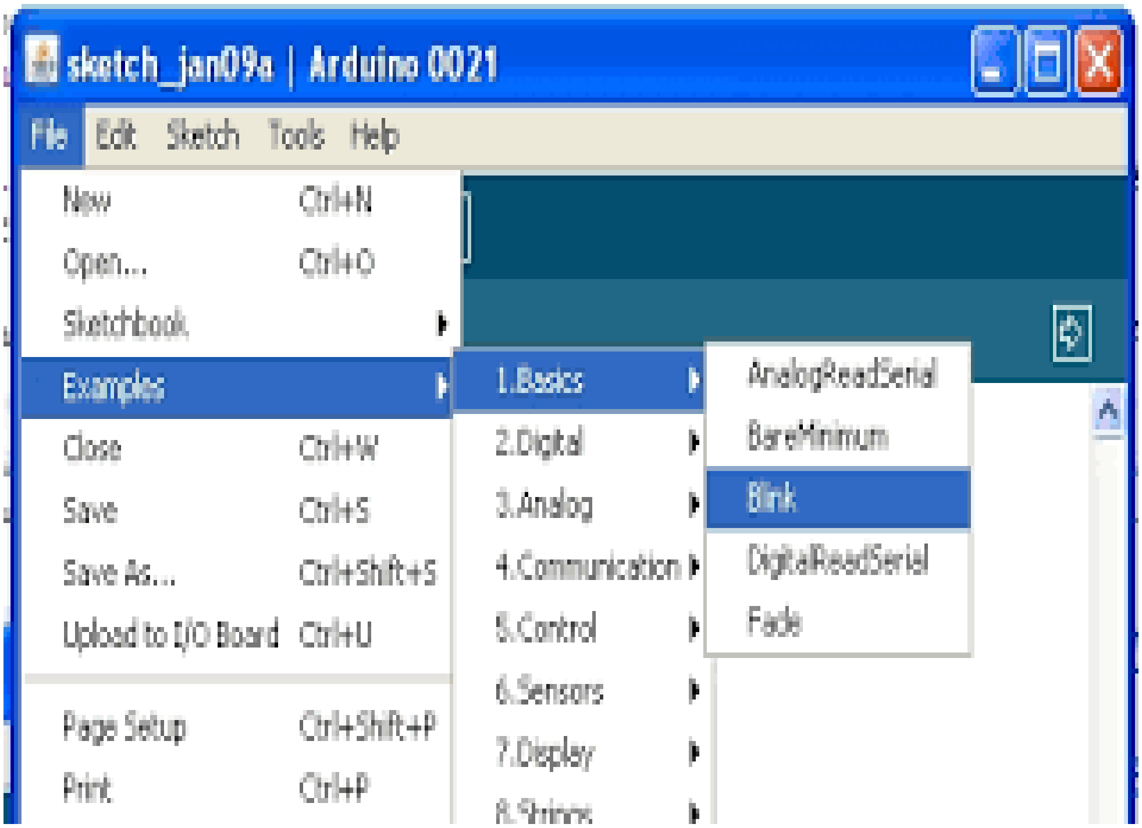


Figure: Select a program from example

Example of Code in IDE Editor: blinking a LED on pin D13

```
void setup() {
 //  initialize  digital
 pin 13 as an output.
 pinMode(13,
 OUTPUT);
}


// the loop function runs over and
over again forever void loop() {
```

```
digitalWrite(13, HIGH); // turn the LED on (HIGH is
the voltage level) delay(1000);     // wait for asecond
   digitalWrite(13, LOW);//   turn   the   LED   off   by
   making the voltage LOW delay(1000);                  //
   wait for asecond
```

```
 }
```

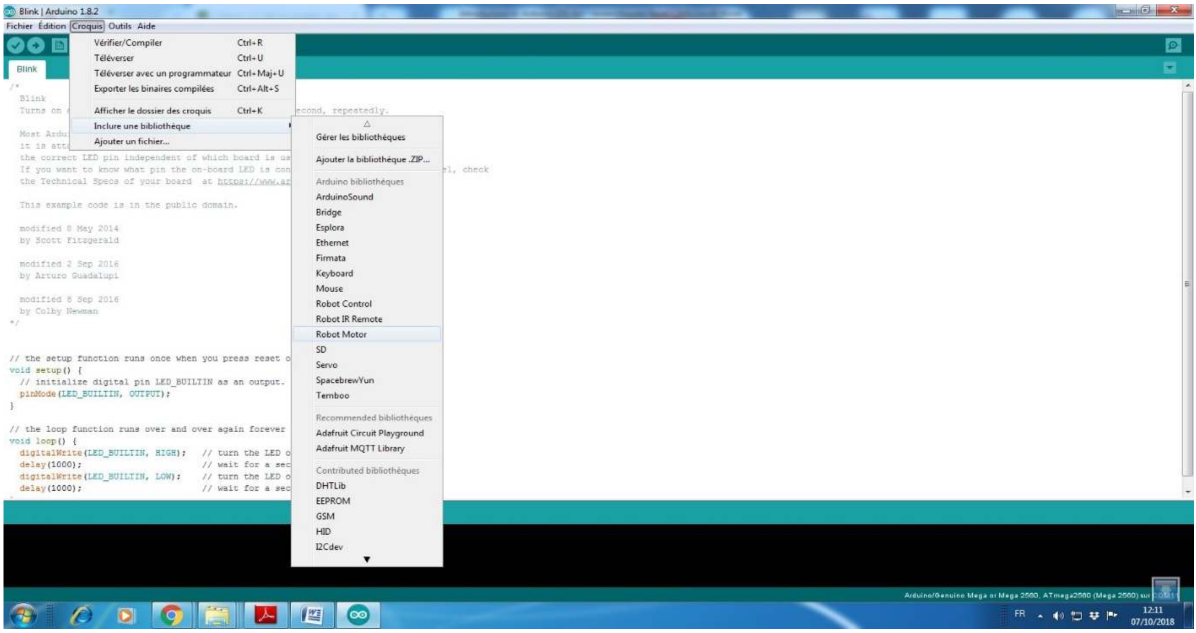
Example 2: read analog value from pin A0

```
void setup() {
 // initialize serial communication at
 9600 bits per second:
 Serial.begin(9600);
}
  // the loop routine runs over
  and over again forever: void
  loop() {
   // read the input on analog pin 0:
   intsensorValue = analogRead(A0);
  //   print   out
  the       value
  you       read:
  Serial.println
  (sensorValue
  );
  delay(1);     // delay in between reads forstability
 }
```

The bottom of the main screen is described as an Output Pane that mainly highlights the compilation status of the running code: the memory used by the code, and errors occurred in the program. You need to fix those errors before you intend to upload the hex file into your Arduino Module.

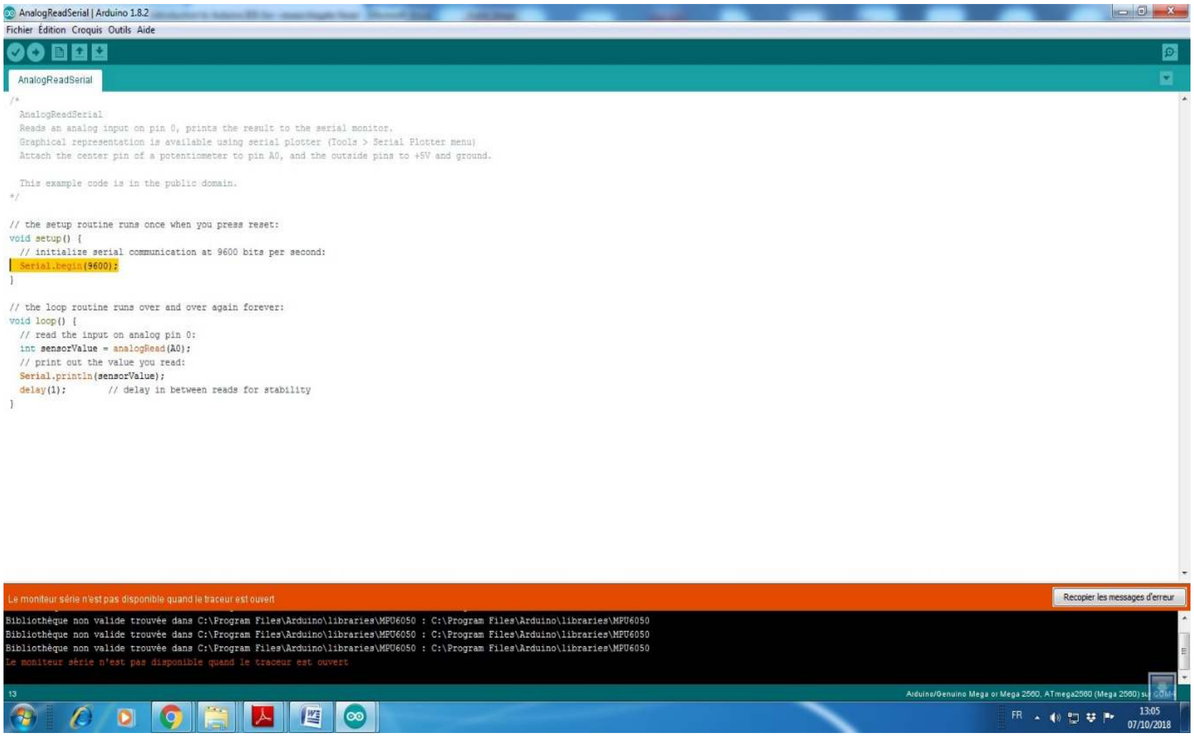
Figure: Error dialog

More or less, Arduino C language works similar to the regular C language used for any embedded system microcontroller, however, there are some dedicated libraries used for calling and executing specific functions on the board.



```
/*
  Blink
  Turns on                                                   econd, repeatedly.

  Most Ardu
  it is att
  the correct LED pin independent of which board is us
  If you want to know what pin the on-board LED is con                        el, check
  the Technical Specs of your board  at https://www.ar

  This example code is in the public domain.

  modified 8 May 2014
  by Scott Fitzgerald

  modified 2 Sep 2016
  by Arturo Guadalupi

  modified 8 Sep 2016
  by Colby Newman
*/

// the setup function runs once when you press reset o
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED o
  delay(1000);                       // wait for a sec
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED o
  delay(1000);                       // wait for a sec
```

3.4 Libraries


AnalogReadSerial | Arduino 1.8.2
Fichier Édition Croquis Outils Aide
AnalogReadSerial
Recopier les messages d'erreur
13
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) sur COM
FR
13:05
07/10/2018

```
/*
  AnalogReadSerial
  Reads an analog input on pin 0, prints the result to the serial monitor.
  Graphical representation is available using serial plotter (Tools > Serial Plotter menu)
  Attach the center pin of a potentiometer to pin A0, and the outside pins to +5V and ground.

  This example code is in the public domain.
*/

// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```

Le moniteur série n'est pas disponible quand le traceur est ouvert

```
Bibliothèque non valide trouvée dans C:\Program Files\Arduino\libraries\MPU6050 : C:\Program Files\Arduino\libraries\MPU6050
Bibliothèque non valide trouvée dans C:\Program Files\Arduino\libraries\MPU6050 : C:\Program Files\Arduino\libraries\MPU6050
Bibliothèque non valide trouvée dans C:\Program Files\Arduino\libraries\MPU6050 : C:\Program Files\Arduino\libraries\MPU6050
Le moniteur série n'est pas disponible quand le traceur est ouvert
```

Libraries are very useful for adding the extra functionality into the Arduino Module. There is a list of libraries you can add by clicking the Sketch button in the menu bar and going to Include Library.

Figure: Including libraries window.

As you click the Include Library and Add the respective library it will on the top of the sketch with a #include sign. Suppose, I Include the EEPROM library, Temperature sensors DHT11/22, LCD orI2C library it will appear on the text editor as

```
#include <EEPROM.h>.
#include <dht.h> #include <I2Cdev.h>
```

Most of the libraries are preinstalled and come with the Arduino software. However, we can also download them from the external sources.

Making Pins As Input or Output

The digitalRead and digitalWrite commands are used for addressing and making the Arduino pins as an input and output respectively.

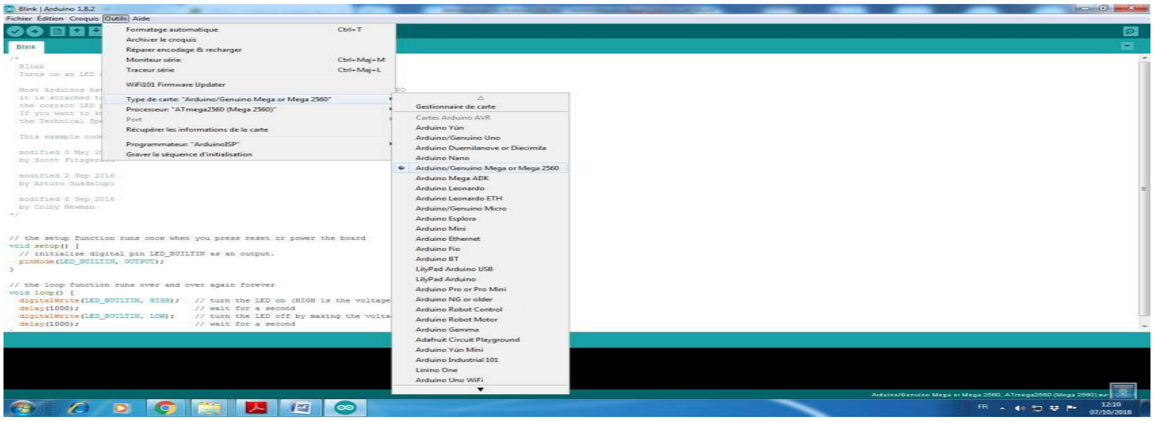
These commands are text sensitive i.e. you need to write them down the exact way they are given like digitalWrite starting with small “d” and write with capital “W”. Writing it down with Digitalwrite or digitalwritewon’t be calling or addressing any function.

Example : if we want to use Pin D13 as output , the code will be; pinMode(13, OUTPUT); followed by digitalWrite(13,HIGH);

If we want to use Pin D13 as input, the code will be : pinMode(13, INPUT);followed by x=digitalRead(13);

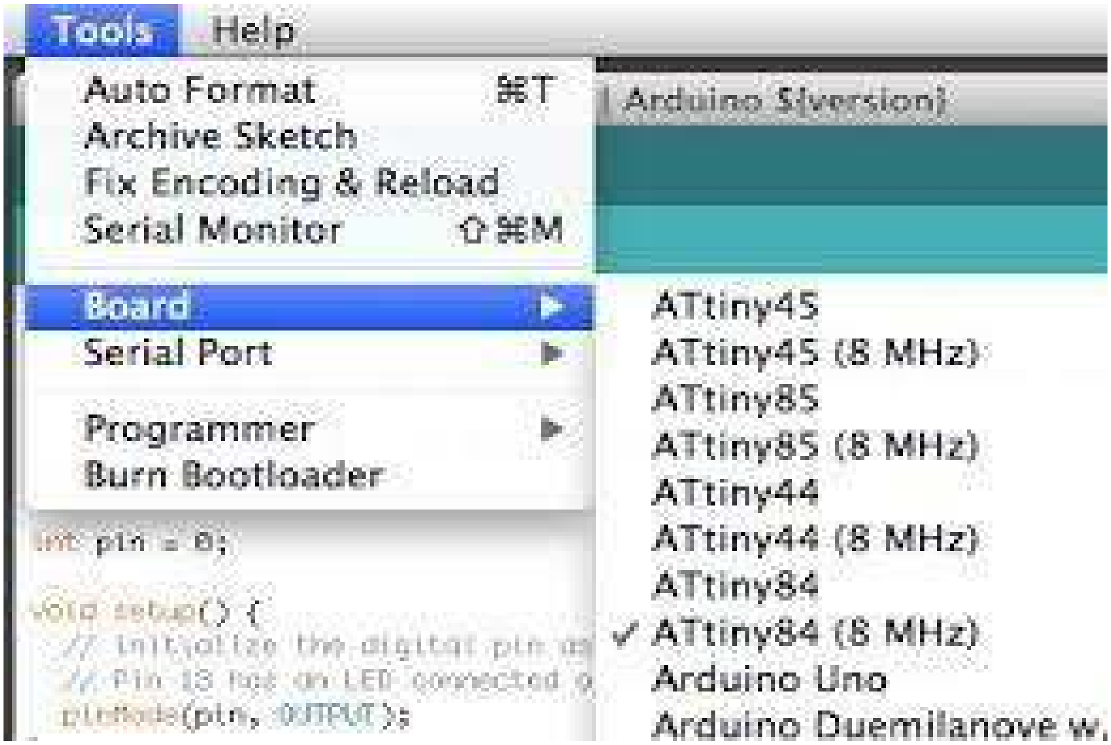
Selecting Board of Arduino

In order to upload the sketch, we need to select the relevant board we are using and the ports for that operating system. As we click the Tools on the Menu, it will open like the figure below.



```
/*
  Blink
  Turns on an LED
  
  Most Arduinos ha
  it is attached t
  the correct LED
  If you want to k
  the Technical Sp
  
  This example cod
  
  modified 8 May 2
  by Scott Fitzgera
  
  modified 2 Sep 2016
  by Arturo Guadalupi
  
  modified 8 Sep 2016
  by Colby Newman
*/

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the volta
  delay(1000);                       // wait for a second
```

Figure: a) Choose the board window.



```
int pin = 0;

void setup() {
  // initialize the digital pin as
  // Pin 13 has an LED connected o
  pinMode(pin, OUTPUT);
```

Figure: b) Choose the board window.

Just we go to the “Board” section and select the board wewould like to work on. Similarly, COM1, COM2, COM4, COM5, COM7 or higher are reserved for the serial and USB board. we can look for the USB serial device in the ports section of the Windows Device Manager.

Folloinwg figure shows the COM4 that wehave used for my project, indicating the Arduino Uno with COM4 port at the right bottom corner of the screen.

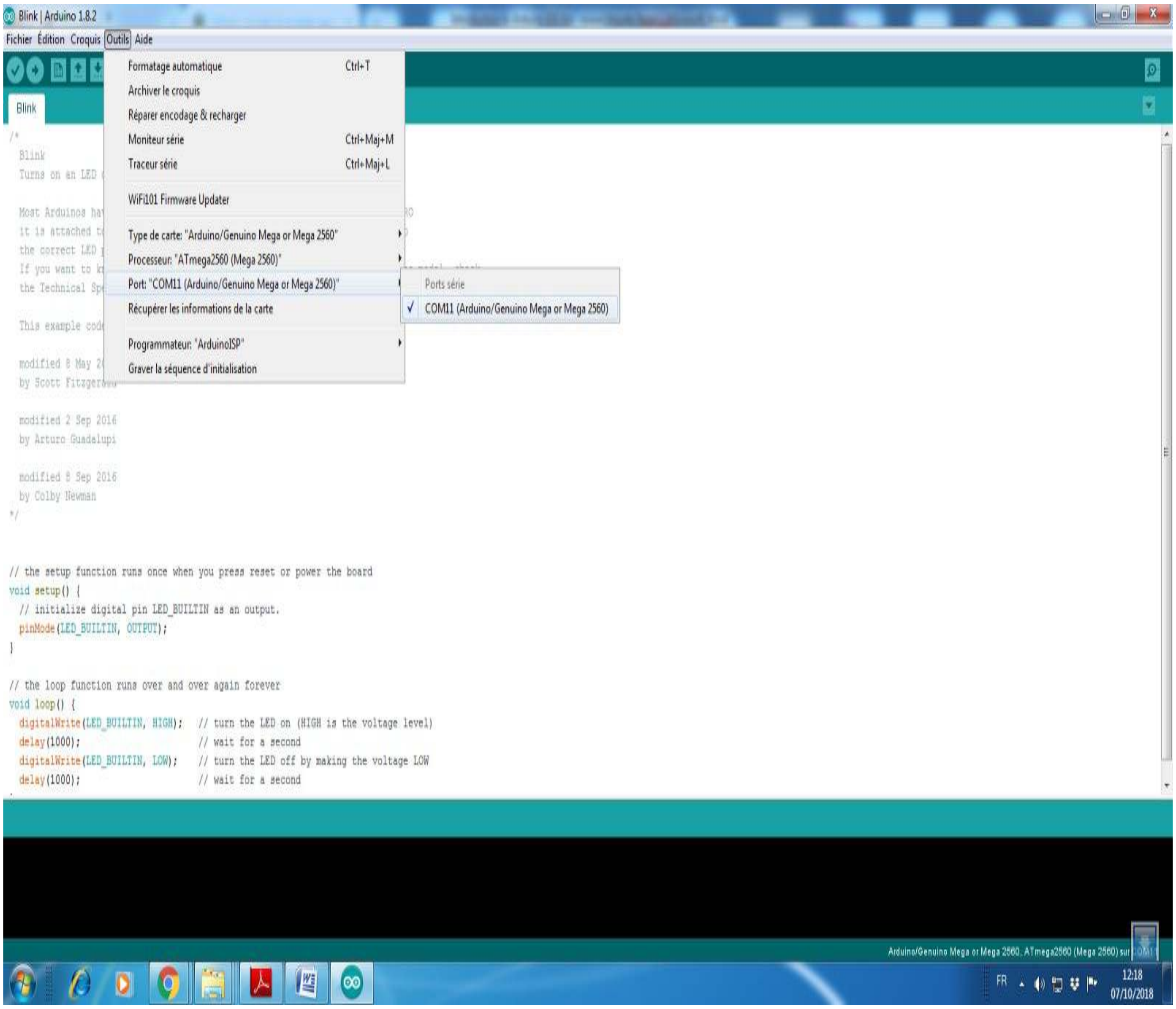
Figure: Selecting the COM port.

After correct selection of both Board and Serial Port, click the verify and then upload button appearing in the upper left corner of the six button section or you can go to the Sketch section and press verify/compile and thenupload.

- The sketch is written in the text editor and is then saved with the file extension.ino.

It is important to note that the recent Arduino Modules will reset automatically as you compile and press the upload button the IDE software, however, older version may require the physical reset on the board.

- As we upload the code, TX and RX LEDs will blink on the board, indicating the desired program is runningsuccessfully.


Blink | Arduino 1.8.2
Fichier Édition Croquis Outils Aide
Blink
Formatage automatique Ctrl+T
Archiver le croquis
Réparer encodage & recharger
Moniteur série Ctrl+Maj+M
Traceur série Ctrl+Maj+L
WiFi101 Firmware Updater
Type de carte: "Arduino/Genuino Mega or Mega 2560"
Processeur: "ATmega2560 (Mega 2560)"
Port: "COM11 (Arduino/Genuino Mega or Mega 2560)"
Récupérer les informations de la carte
Programmateur: "ArduinoISP"
Graver la séquence d'initialisation
Ports série
✓ COM11 (Arduino/Genuino Mega or Mega 2560)
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) sur COM11
FR
12:18
07/10/2018

```
/*
  Blink
  Turns on an LED (

  Most Arduinos ha
  it is attached t
  the correct LED
  If you want to k
  the Technical Sp

  This example code

  modified 8 May 2
  by Scott Fitzgera

  modified 2 Sep 2016
  by Arturo Guadalupi

  modified 8 Sep 2016
  by Colby Newman
*/


// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
```

Note: The port selection criteria mentioned above is dedicated for Windows operating system only, you can check this <u>Guide </u>if you are using MAC or

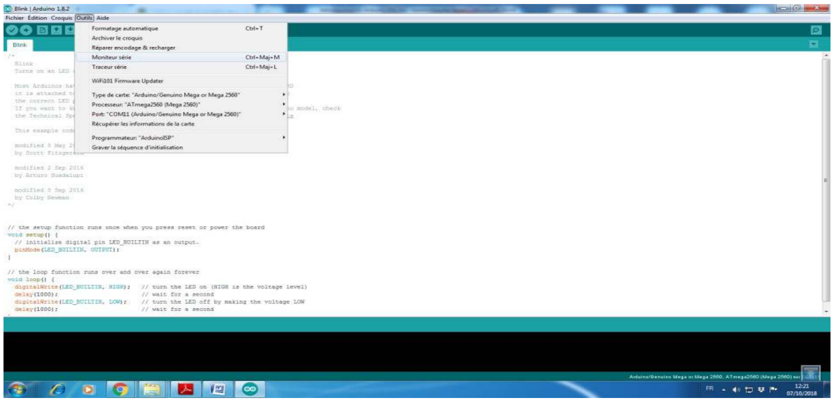
Linux.

- The amazing thing about this software is that no prior arrangement or bulk of mess is required to install this software, you will be writing your first program within 2 minutes after the installation of the IDEenvironment.

Using Serial Monitor

Example of test serial monitor on program test2.cc

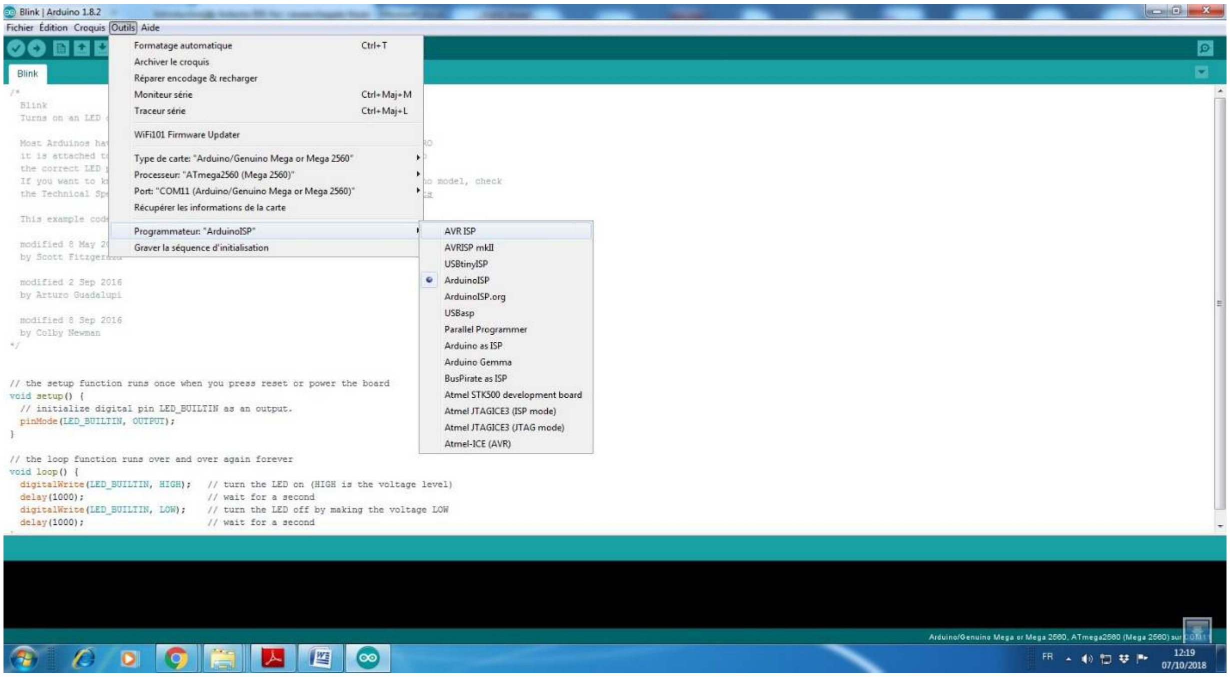
Figure: COM port chosen.


Blink | Arduino 1.8.2
Fichier Édition Croquis Outils Aide
Blink
Formatage automatique Ctrl+T
Archiver le croquis
Réparer encodage & recharger
Moniteur série Ctrl+Maj+M
Traceur série Ctrl+Maj+L
WiFi101 Firmware Updater
Type de carte: "Arduino/Genuino Mega or Mega 2560"
Processeur: "ATmega2560 (Mega 2560)"
Port: "COM11 (Arduino/Genuino Mega or Mega 2560)"
Récupérer les informations de la carte
Programmateur: "ArduinoISP"
Graver la séquence d'initialisation
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) sur COM11
FR
12:21
07/10/2018

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
```

Bootloader

As we go to the Tools section, we will find a bootloader at the end. It is very helpful to burn the code directly into the controller, setting us free from buying the external burner to burn the required code.


Blink | Arduino 1.8.2
Fichier Édition Croquis Outils Aide
Blink
Formatage automatique Ctrl+T
Archiver le croquis
Réparer encodage & recharger
Moniteur série Ctrl+Maj+M
Traceur série Ctrl+Maj+L
WiFi101 Firmware Updater
Type de carte: "Arduino/Genuino Mega or Mega 2560"
Processeur: "ATmega2560 (Mega 2560)"
Port: "COM11 (Arduino/Genuino Mega or Mega 2560)"
Récupérer les informations de la carte
Programmateur: "ArduinoISP"
Graver la séquence d'initialisation
AVR ISP
AVRISP mkII
USBtinyISP
● ArduinoISP
ArduinoISP.org
USBasp
Parallel Programmer
Arduino as ISP
Arduino Gemma
BusPirate as ISP
Atmel STK500 development board
Atmel JTAGICE3 (ISP mode)
Atmel JTAGICE3 (JTAG mode)
Atmel-ICE (AVR)
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) sur COM11
FR
12:19
07/10/2018

```
/*
  Blink
  Turns on an LED

  modified 2 Sep 2016
  by Arturo Guadalupi

  modified 8 Sep 2016
  by Colby Newman
*/


// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
```

Figure: Bootloader programming window

When we buy the new Arduino Module, the bootloader is already installed inside the controller. However, if we intend to buy a controller and put in the Arduino module, we need to burn the bootloader again inside the controller by going to the Tools section and selecting the burnbootloader.

The Program Structure

1. Pin functionality using the pinModefunction
2. Initial state ofpins
3. Initializeclasses
4. Initializevariables
5. Codelogic

Declarations

Variables

Whenever you’re using Arduino, you need to declare global variables and instances to be used later on. In a nutshell, a variable allows you to name and store a value to be used in the future. For example, you would store data acquired from a sensor in order to use it later. To declare a variable you simply define its type, name and initial value.

It’s worth mentioning that declaring global variables isn’t an absolute necessity. However, it’s advisable that you declare your variables to make it easy to utilize your values further down the line.

Instances

In software programming, a **class** is a collection of functions and variables that are kept together in one place. Each class has a special function known as a **constructor**, which is used to create an **instance** of the class. In order to use the functions of the class, we need to declare an instance for it.

Setup()

Every Arduino sketch must have a setup function. This function defines the initial state of theArduino upon boot and runs onlyonce.

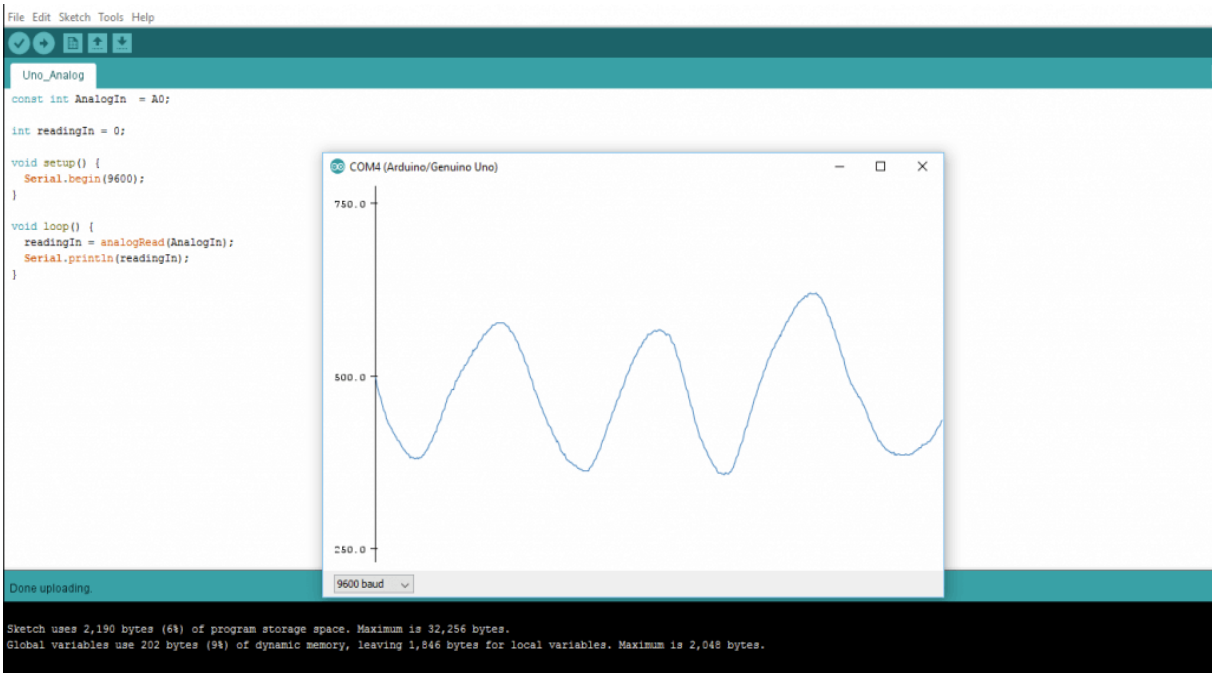
Here we’ll define the following:

Loop()

The loop function is also a must for every Arduino sketch and executes once setup() is complete. It is the main function and as its name hints, it runs in a loop over and over again. The loop describes the main logic of your circuit.

.

Serial Plotter



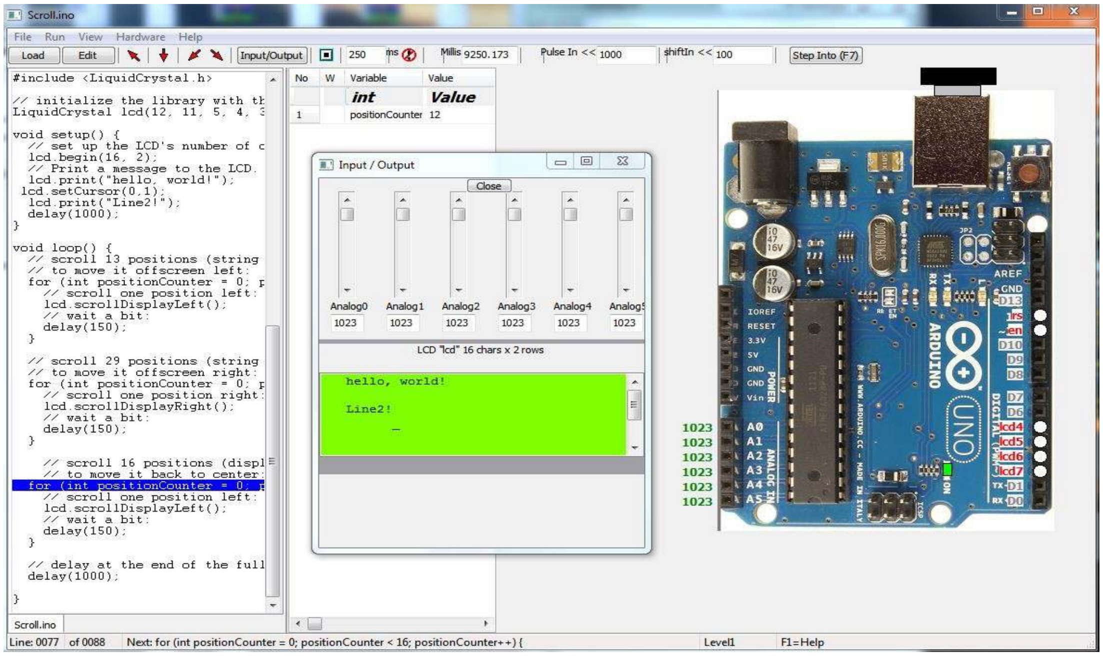
```
const int AnalogIn  = A0;

int readingIn = 0;

void setup() {
  Serial.begin(9600);
}

void loop() {
  readingIn = analogRead(AnalogIn);
  Serial.println(readingIn);
}
```

```
Sketch uses 2,190 bytes (6%) of program storage space. Maximum is 32,256 bytes.
Global variables use 202 bytes (9%) of dynamic memory, leaving 1,846 bytes for local variables. Maximum is 2,048 bytes.
```

Arduino **serial plotter** is another component of the Arduino IDE, which allows you to generate a real-time graph of your serial data. The serial plotter makes it much easier for you to analyze your data through a visual display. You're able to create graphs, negative value graphs, and conduct waveform analysis

Figure: Serial plotter.

Simulators for ARDUINO: **in the next article we will see some details on powerful simulators for Arduino, example:** Autodesk Eagle (recommended), Proteus , Autodesk Circuits, Virtronics Simulator for Arduino , Electronify , Fritzing, VBB4Arduino – Virtual Breadboard for Arduino,



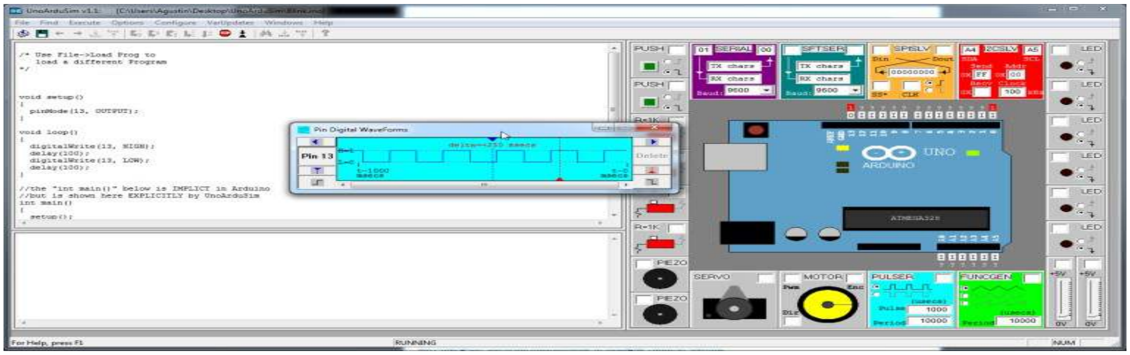
```
#include <LiquidCrystal.h>

// initialize the library with th
LiquidCrystal lcd(12, 11, 5, 4, 3

void setup() {
  // set up the LCD's number of c
  lcd.begin(16, 2);
  // Print a message to the LCD.
  lcd.print("hello, world!");
 lcd.setCursor(0,1);
  lcd.print("Line2!");
  delay(1000);
}

void loop() {
  // scroll 13 positions (string
  // to move it offscreen left:
  for (int positionCounter = 0; p
    // scroll one position left:
    lcd.scrollDisplayLeft();
    // wait a bit:
    delay(150);
  }

  // scroll 29 positions (string
  // to move it offscreen right:
  for (int positionCounter = 0; p
    // scroll one position right:
    lcd.scrollDisplayRight();
    // wait a bit:
    delay(150);
  }

    // scroll 16 positions (displ
    // to move it back to center:
  for (int positionCounter = 0; p
    // scroll one position left:
    lcd.scrollDisplayLeft();
    // wait a bit:
    delay(150);
  }

  // delay at the end of the full
  delay(1000);

}
```

```
hello, world!

Line2!
```

Line: 0077 of 0088 Next: for (int positionCounter = 0; positionCounter < 16; positionCounter++) {

Scroll.ino
File Run View Hardware Help
Load Edit Input/Output 250 ms Millis 9250.173 Pulse In << 1000 shiftIn << 100 Step Into (F7)
No W Variable Value
int Value
1 positionCounter 12
Input / Output
Close
Analog0 Analog1 Analog2 Analog3 Analog4 Analog5
1023 1023 1023 1023 1023 1023
LCD "lcd" 16 chars x 2 rows
Level1 F1=Help
1023
1023
1023
1023
1023
1023
A0 A1 A2 A3 A4 A5
IOREF RESET 3.3V 5V GND GND Vin
POWER ANALOG IN
AREF GND D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0
lcd4 lcd5 lcd6 lcd7 rs en
ARDUINO UNO
DIGITAL

Figure: Simulators.



```
/* Use File->Load Prog to
   load a different Program
*/



void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(100);
  digitalWrite(13, LOW);
  delay(100);
}

//the "int main()" below is IMPLICT in Arduino
//but is shown here EXPLICITLY by UnoArduSim
int main()
{
  setup();
```

Figure: Virtual breadboard.

TREADMILL MAIN CODE

```
#include <LiquidCrystal.h>
#include <Adafruit_Fingerprint.h>

intenA = 10;
int in1 = 9;
int in2 = 8;

SoftwareSerialmySerial(2, 3);

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);
constintrs = 12, en = 11, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
LiquidCrystallcd(rs, en, d4, d5, d6, d7);

void setup()
{
Serial.begin(9600);
  while (!Serial);  // For Yun/Leo/Micro/Zero/...
  delay(100);
Serial.println("\n\nAdafruit finger detect test");
lcd.begin(16, 2);
  // Print a message to the LCD.
lcd.setCursor(0, 0);
lcd.print("   WELCOME  TO   ");
lcd.setCursor(0, 1);
lcd.print("   THREADMILL   ");
```

```
  delay(1000);
    // set all the motor control pins to outputs
pinMode(enA, OUTPUT);
pinMode(in1, OUTPUT);
pinMode(in2, OUTPUT);
  // set the data rate for the sensor serial port
finger.begin(57600);
  delay(5);
  if (finger.verifyPassword()) {
Serial.println("Found fingerprint sensor!");
lcd.setCursor(0, 0);
lcd.print("Found fingerprint");
lcd.setCursor(0, 1);
lcd.print("    sensor!    ");
  } else {
Serial.println("Did not find fingerprint sensor :(");
    while (1) { delay(1); }
  }

finger.getTemplateCount();
Serial.print("Sensor contains "); Serial.print(finger.templateCount);
Serial.println(" templates");
Serial.println("Waiting for valid finger...");
lcd.setCursor(0, 1);
lcd.print("Fngrprnt Not fnd");
  delay(2000);
```

```
lcd.print("                ");


}


void loop()                    // run over and over again
{
intadcvalue=analogRead(A0);


intoutvalue=adcvalue*0.035;
Serial.println(outvalue);
lcd.setCursor(0, 1);
lcd.print("OUTPUT VOLTS:");
lcd.setCursor(14, 1);
lcd.print(outvalue);
getFingerprintIDez();
  delay(1000);           //don't ned to run this at full speed.
lcd.print("                ");
  delay(1000);
}


uint8_t getFingerprintID() {
  uint8_t p = finger.getImage();
  switch (p) {
    case FINGERPRINT_OK:
Serial.println("Image taken");
      break;
  case FINGERPRINT_NOFINGER:
Serial.println("No finger detected");
   return p;
  case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("Communication error");
   return p;
  case FINGERPRINT_IMAGEFAIL:
Serial.println("Imaging error");
   return p;
  default:
Serial.println("Unknown error");
   return p;
 }

 // OK success!

 p = finger.image2Tz();
 switch (p) {
  case FINGERPRINT_OK:
Serial.println("Image converted");
   break;
  case FINGERPRINT_IMAGEMESS:
Serial.println("Image too messy");
   delay(1000);
   return p;
  case FINGERPRINT_PACKETRECIEVEERR:
```

```
Serial.println("Communication error");
      return p;
    case FINGERPRINT_FEATUREFAIL:
Serial.println("Could not find fingerprint features");
      delay(1000);
      return p;
    case FINGERPRINT_INVALIDIMAGE:
Serial.println("Could not find fingerprint features");
      delay(1000);
      return p;
    default:
Serial.println("Unknown error");
      return p;
  }

  // OK converted!
  p = finger.fingerFastSearch();
  if (p == FINGERPRINT_OK) {
Serial.println("Found a print match!");
    delay(1000);
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
Serial.println("Communication error");
    delay(1000);
    return p;
  } else if (p == FINGERPRINT_NOTFOUND) {
Serial.println("Did not find a match");
```

```
    delay(1000);
    return p;
  } else {
Serial.println("Unknown error");
    delay(1000);
    return p;
  }

  // found a match!
Serial.print("Found ID #"); Serial.print(finger.fingerID);
Serial.print(" with confidence of "); Serial.println(finger.confidence);

  // Print a message to the LCD.
lcd.setCursor(0, 0);
lcd.print(" Found ID # ");
lcd.setCursor(13, 0);
lcd.print(finger.fingerID);

lcd.setCursor(0, 1);
lcd.print("FNGRPRNT MATCHED");

digitalWrite(in1, HIGH);
digitalWrite(in2, LOW);
lcd.setCursor(0, 0);
lcd.print("THREADMILL START");
lcd.setCursor(0, 1);
lcd.print("  RUNNING       ");
  return finger.fingerID;
}

// returns -1 if failed, otherwise returns ID #
intgetFingerprintIDez() {
  uint8_t p = finger.getImage();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.image2Tz();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.fingerFastSearch();
  if (p != FINGERPRINT_OK)  return -1;

  // found a match!
Serial.print("Found ID #"); Serial.print(finger.fingerID);
Serial.print(" with confidence of "); Serial.println(finger.confidence);
  // Print a message to the LCD.
lcd.setCursor(0, 0);
lcd.print(" Found ID # ");
lcd.setCursor(13, 0);
lcd.print(finger.fingerID);

lcd.setCursor(0, 1);
lcd.print("FNGRPRNT MATCHED");
```

```
  delay(2000);

digitalWrite(in1, HIGH);
digitalWrite(in2, LOW);
lcd.setCursor(0, 0);
lcd.print("THREADMILL START");
lcd.setCursor(0, 1);
lcd.print("  RUNNING       ");



  return finger.fingerID;
  return finger.fingerID;
}
```

FINGERPRINT CODE

```
#include <LiquidCrystal.h>
#include <Adafruit_Fingerprint.h>

SoftwareSerialmySerial(2, 3);

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);

constintrs = 12, en = 11, d4 = 7, d5 = 6, d6 = 5, d7 = 4;
LiquidCrystallcd(rs, en, d4, d5, d6, d7);
```

```
uint8_t id;

void setup()
{
Serial.begin(9600);
  while (!Serial);  // For Yun/Leo/Micro/Zero/...
  delay(100);
Serial.println("\n\nAdafruit Fingerprint sensor enrollment");
lcd.begin(16, 2);
  // Print a message to the LCD.
lcd.setCursor(0, 0);
lcd.print("   WELCOME  TO   ");
lcd.setCursor(0, 1);
lcd.print("FNGRPRNT ENRLMNT");
  delay(1000);
  // set the data rate for the sensor serial port
finger.begin(57600);

  if (finger.verifyPassword()) {
Serial.println("Found fingerprint sensor!");
lcd.setCursor(0, 0);
lcd.print("Found fingerprint");
lcd.setCursor(0, 1);
lcd.print("    sensor!    ");
  } else {
```

```
Serial.println("Did not find fingerprint sensor :(");
lcd.setCursor(0, 0);
lcd.print("  Did  not find  ");
lcd.setCursor(0, 1);
lcd.print("fingerprint sensr");
    while (1) { delay(1); }
  }
}

uint8_t readnumber(void) {
  uint8_t num = 0;

  while (num == 0) {
    while (! Serial.available());
num = Serial.parseInt();
  }
  return num;
}

void loop()                    // run over and over again
{
Serial.println("Ready to enroll a fingerprint!");
Serial.println("Please type in the ID # (from 1 to 127) you want to save this finger as...");
  id = readnumber();
  if (id == 0) {// ID #0 not allowed, try again!
```

```
    return;
  }
Serial.print("Enrolling ID # ");
Serial.println(id);
lcd.setCursor(0, 0);
lcd.print("Enrolling ID#");
lcd.setCursor(16, 0);
lcd.print(id);
lcd.setCursor(0, 1);
lcd.print("                ");
  while (!  getFingerprintEnroll() );
}

uint8_t getFingerprintEnroll() {


int p = -1;
Serial.print("Waiting for valid finger to enroll as #"); Serial.println(id);
lcd.print("  Image Taken   ");
lcd.setCursor(0, 1);
lcd.print("                ");
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
    case FINGERPRINT_OK:
Serial.println("Image taken");
lcd.print("  Image Taken   ");
lcd.setCursor(0, 1);
lcd.print("                ");
      break;
    case FINGERPRINT_NOFINGER:
Serial.println(".");
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("Communication error");
      break;
    case FINGERPRINT_IMAGEFAIL:
Serial.println("Imaging error");
      break;
    default:
Serial.println("Unknown error");
      break;
    }
  }

  // OK success!

  p = finger.image2Tz(1);
  switch (p) {
    case FINGERPRINT_OK:
Serial.println("Image converted");
      break;
    case FINGERPRINT_IMAGEMESS:
```

```
Serial.println("Image too messy");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("Communication error");
      return p;
    case FINGERPRINT_FEATUREFAIL:
Serial.println("Could not find fingerprint features");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
Serial.println("Could not find fingerprint features");
      return p;
    default:
Serial.println("Unknown error");
      return p;
  }

Serial.println("Remove finger");
lcd.setCursor(0, 0);
lcd.print(" Remove finger ");
lcd.setCursor(0, 1);
lcd.print("                ");

  delay(2000);
  p = 0;
  while (p != FINGERPRINT_NOFINGER) {
    p = finger.getImage();
  }
Serial.print("ID "); Serial.println(id);
  p = -1;
Serial.println("Place same finger again");
lcd.setCursor(0, 0);
lcd.print("   Place same   ");
lcd.setCursor(0, 1);
lcd.print("  finger again  ");
  while (p != FINGERPRINT_OK) {
    p = finger.getImage();
    switch (p) {
    case FINGERPRINT_OK:
Serial.println("Image taken");
lcd.setCursor(0, 0);
lcd.print("   Image taken  ");
lcd.setCursor(0, 1);
lcd.print("                ");
      break;
    case FINGERPRINT_NOFINGER:
Serial.print(".");
      break;
    case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("Communication error");
      break;
    case FINGERPRINT_IMAGEFAIL:
Serial.println("Imaging error");
```

```
      break;
    default:
Serial.println("Unknown error");
      break;
    }
  }

  // OK success!

  p = finger.image2Tz(2);
  switch (p) {
    case FINGERPRINT_OK:
Serial.println("Image converted");
      break;
    case FINGERPRINT_IMAGEMESS:
Serial.println("Image too messy");
      return p;
    case FINGERPRINT_PACKETRECIEVEERR:
Serial.println("Communication error");
      return p;
    case FINGERPRINT_FEATUREFAIL:
Serial.println("Could not find fingerprint features");
      return p;
    case FINGERPRINT_INVALIDIMAGE:
Serial.println("Could not find fingerprint features");
      return p;
  default:
Serial.println("Unknown error");
   return p;
 }

 // OK converted!
Serial.print("Creating model for #");  Serial.println(id);

 p = finger.createModel();
 if (p == FINGERPRINT_OK) {
lcd.setCursor(0, 0);
Serial.println("Prints matched!");
lcd.print("Prints matched! ");
 } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
Serial.println("Communication error");
  return p;
 } else if (p == FINGERPRINT_ENROLLMISMATCH) {
Serial.println("Fingerprints did not match");
lcd.setCursor(0, 0);
lcd.print("Fingerprints did");
lcd.setCursor(0, 1);
lcd.print("    not match    ");
  return p;
 } else {
Serial.println("Unknown error");
  return p;
  }


Serial.print("ID "); Serial.println(id);
  p = finger.storeModel(id);
  if (p == FINGERPRINT_OK) {
Serial.println("Stored!");
lcd.print(" Prints Stored! ");
  } else if (p == FINGERPRINT_PACKETRECIEVEERR) {
Serial.println("Communication error");
    return p;
  } else if (p == FINGERPRINT_BADLOCATION) {
Serial.println("Could not store in that location");
    return p;
  } else if (p == FINGERPRINT_FLASHERR) {
Serial.println("Error writing to flash");
    return p;
  } else {
Serial.println("Unknown error");
    return p;
  }
}
```

System Specifications

Hardware used

- Arduino UNO Microcontroller

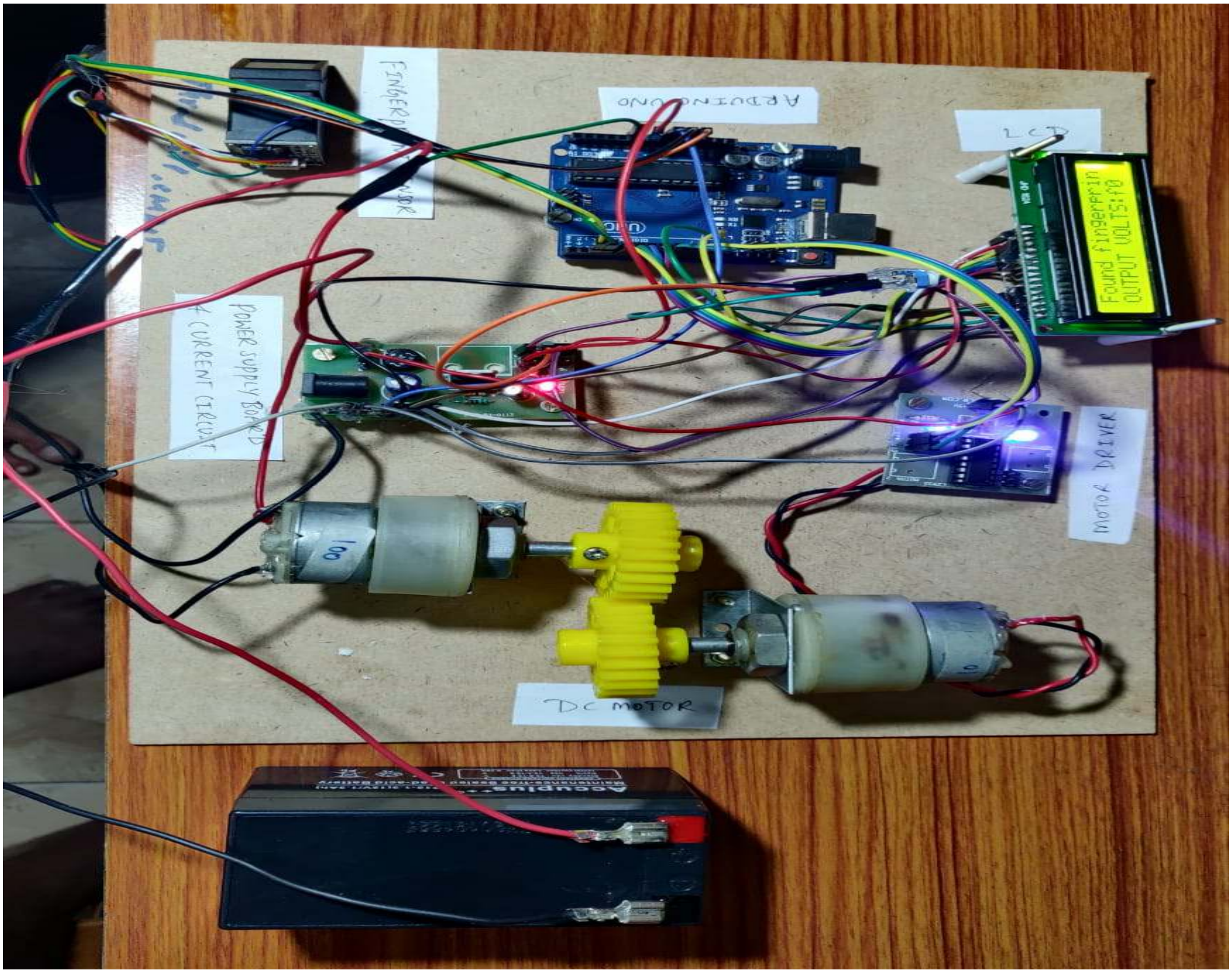
- Buzzer
- Fingerprint scanner
- Treadmill
- Battery
- LCD
- DC Motor

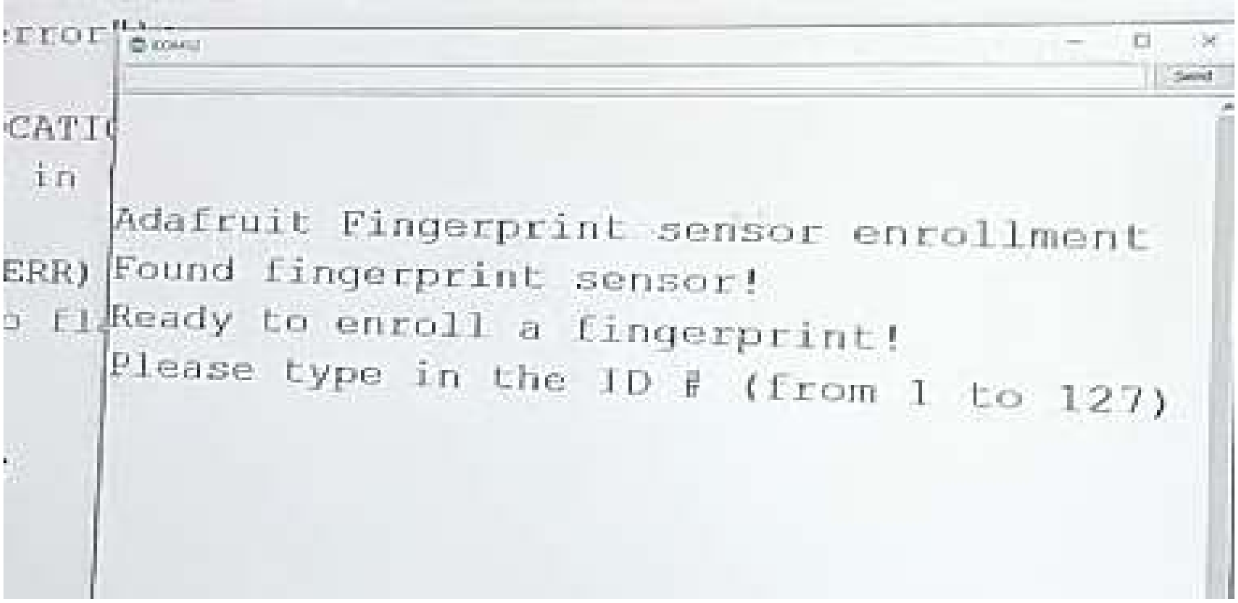
Software used

- Embedded C
- Arduino IDE

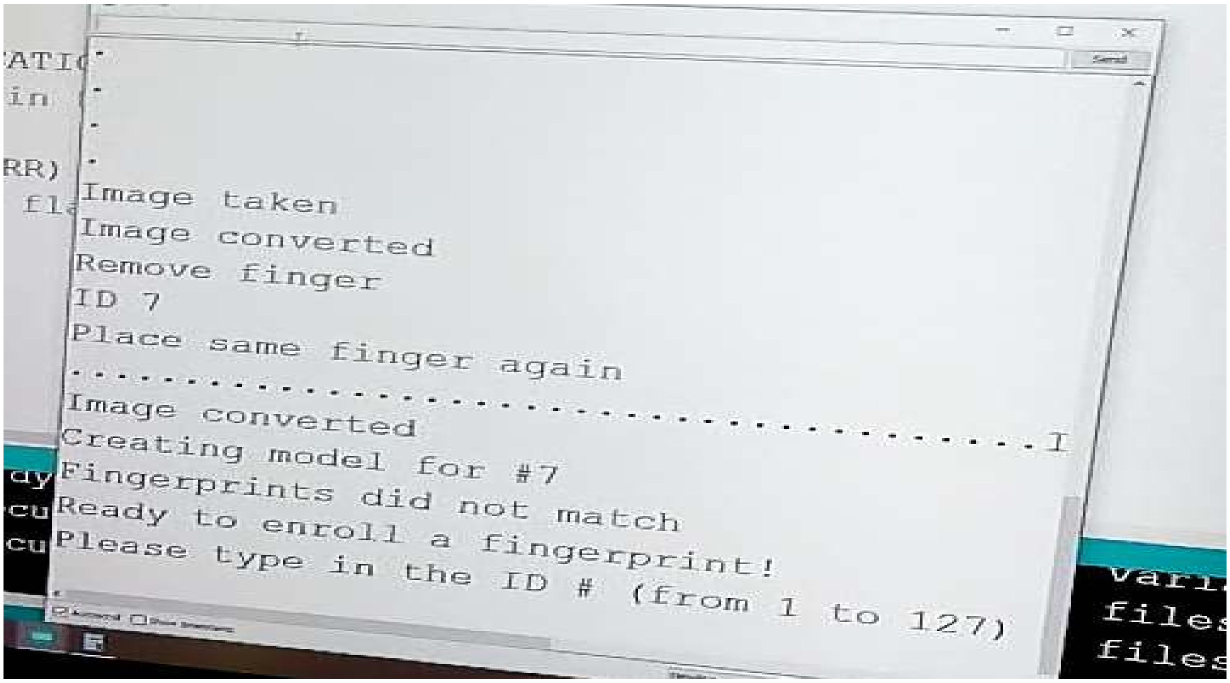
Chapter 6

RESULTS

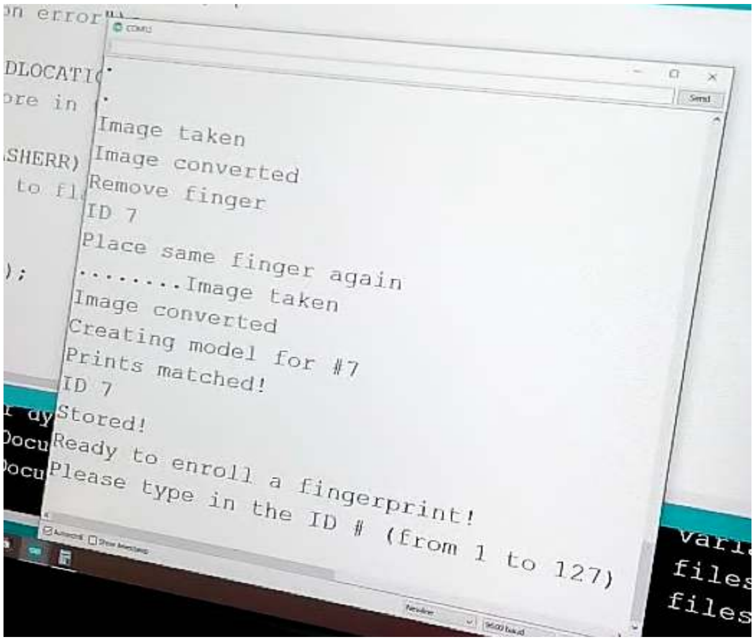

FINGERPRINT SENSOR
ARDUINO UNO
LCD
Found fingerprin
OUTPUT VOLTS:0.0
POWER SUPPLY BOARD
& CURRENT CIRCUIT
MOTOR DRIVER
DC MOTOR



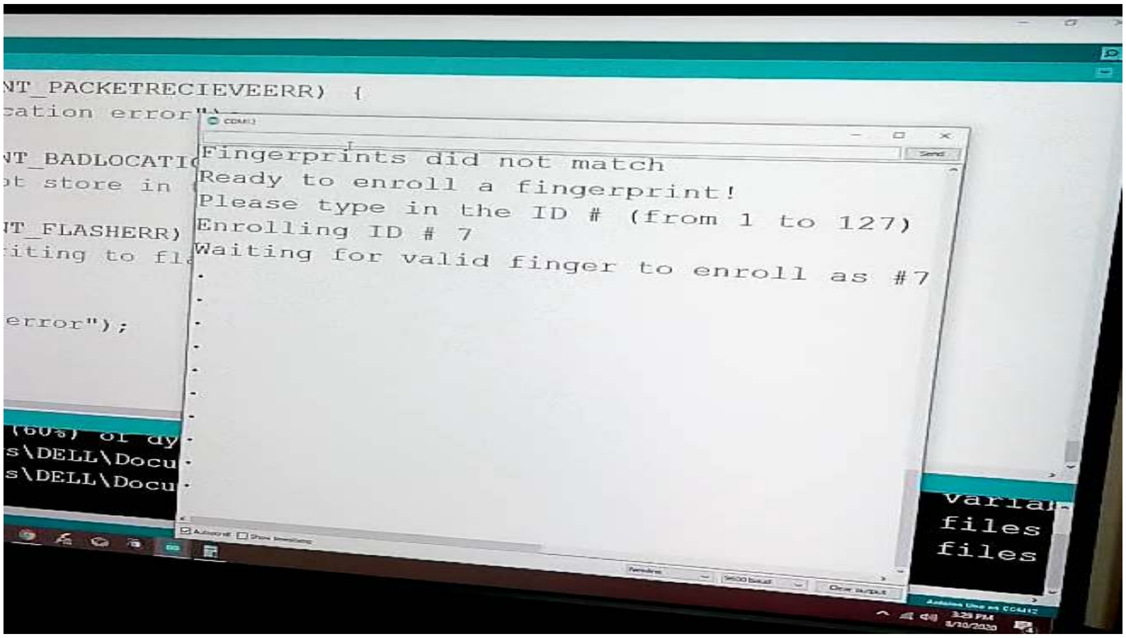
```
Adafruit Fingerprint sensor enrollment
Found fingerprint sensor!
Ready to enroll a fingerprint!
Please type in the ID # (from 1 to 127)
```



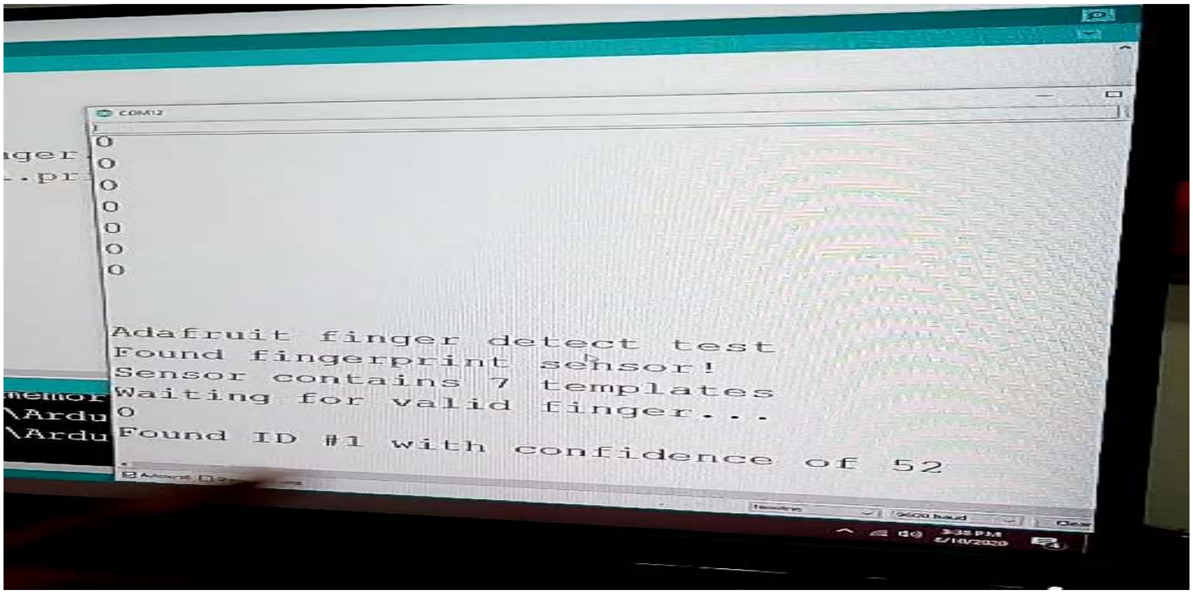
```
.
.
.
.
Image taken
Image converted
Remove finger
ID 7
Place same finger again
...................................................
Image converted
Creating model for #7
Fingerprints did not match
Ready to enroll a fingerprint!
Please type in the ID # (from 1 to 127)
```



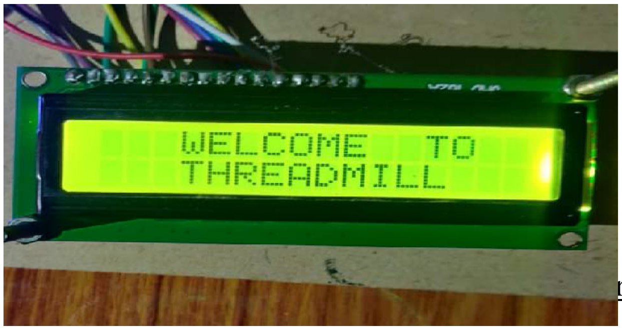
```
.
.
Image taken
Image converted
Remove finger
ID 7
Place same finger again
........Image taken
Image converted
Creating model for #7
Prints matched!
ID 7
Stored!
Ready to enroll a fingerprint!
Please type in the ID # (from 1 to 127)
```

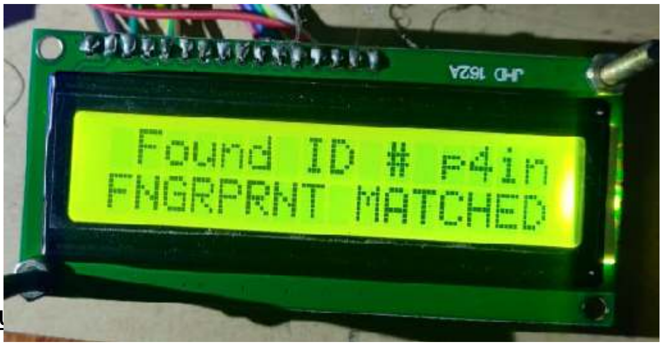


```
Fingerprints did not match
Ready to enroll a fingerprint!
Please type in the ID # (from 1 to 127)
Enrolling ID # 7
Waiting for valid finger to enroll as #7
.
.
.
.
.
.
.
.
.
.
```

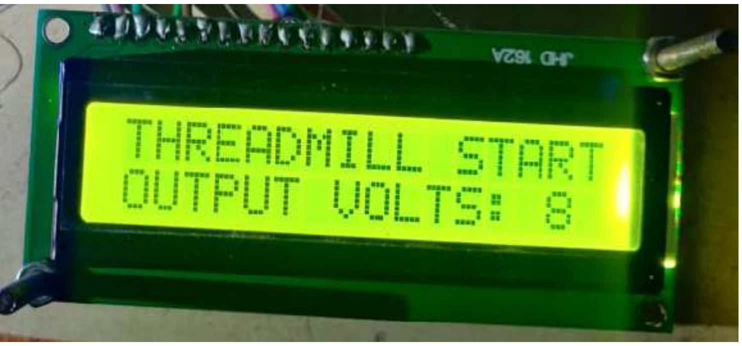


```
0
0
0
0
0
0
0

Adafruit finger detect test
Found fingerprint sensor!
Sensor contains 7 templates
Waiting for valid finger...
0
Found ID #1 with confidence of 52
```





```
THREADMIDL#START
FNRUNNINGMATCHED
```



```
THREADMILL START
OUTPUT VOLTS: 8
```

Chapter 7

ADVANTAGES, DISADVANTAGES AND APPLICATIONS

Advantages

- Enable the user to set up an exercise regime indoors that can be adhered to irrespective of the weather.
- Cushioned tread can provide slightly lower impact training than running on outdoor surfaces. Although cushioned belts have mostly been deprecated out of use and cushioned replacement belts may be hard to find, many treadmills have rubber or urethane deck elastomers (cushions) which are superior in cushioning and last longer than cushioned belts. There were, for a time, banana shaped flexible decks that were among the very best as far as cushioning that were priced at a mid-range level, but currently these are not being sold, perhaps because of the increased manufacturing cost of making the flexible deck. Cushioned belts also do not last as long as regular belts due to their construction out of weaker materials. For calorie burning, incline can be used to significantly reduce impact for a given rate of energy use.
- Incline setting can allow for consistent "uphill" training that is not possible when relying on natural features.
- Rate settings force a consistent pace.
- Some treadmills have programs such that the user can simulate terrains, e.g. rolling hills, to provide accurate, programmed, exercise periods.

- The user can watch TV whilst using the machine thus preventing TV from being a sedentary activity.
- User progress such as distance, calories burned, and heart rate can be tracked.

Disadvantages

As a cardiovascular exercise:

- Some treadmill runners develop bad running habits that become apparent when they return to outdoor running. In particular a short, upright, bouncy gait may result from having no wind resistance and trying to avoid kicking the motor covering with the front of the foot.
- Imposes a strict pace on runners, giving an unnatural feel to running which can cause a runner to lose balance.
- Treadmill running is not specific to any sport, i.e., there is no competitive sport that actually utilizes treadmill running. For example, a competitive runner would be far better off running outdoors through space since it is more specific and realistic to his/her event.

As an indoor activity:

- Many users find treadmills monotonous and lose interest after a period.[10]
- Treadmills do not offer the psychological satisfaction some runners get from running in new locations away from the distractions of home.

As a machine:

- May cause personal injury if not used properly. Of particular concern are children who reach into the treadmill belt while it is running and suffer severe friction burns that may require multiple skin grafts and result in lasting disability.[11] Injury to children can be avoided by removing the safety key when the treadmill is not in use, without which, the treadmill belt will not start.

- Costs of purchase, electrical costs, and possible repair are significantly greater than those of running outside.
- Takes up space in homes.

Applications

Exercise

Treadmill Cycle helps in maintaining proper physique. As physical fitness is important in day to day life. By using treadmill Cycle one can exercise outdoors in fresh air.

Fuel saving

People often use vehicle for travelling over short distance. This causes unnecessary wastage of fuel. Due to use of treadmill Cycle over short distance a large amount of fuel can be saved.

Travelling

Treadmill cycle can be used for travelling over short distances. One can also exercise while travelling over short distance.

Eco- friendly

Treadmill Cycle does not require any fuel. Therefore, it does not emit any pollutants. So, it is an eco-friendly vehicle.

Consumes less power

Health benefits

Treadmill maintenance

A treadmill can lose its speed and performance if not maintained from time to time. Starting from positioning of the treadmill to regular oil checks, a treadmill's longevity is decided on how it is maintained.

Placement

Ideally, a treadmill is placed on a leveled floor in order to ensure the belt and motor have a proper balanced movement. In case of uneven floors, the elevation of either of the legs (rear or front legs) are leveled out using a wood block or a brick.

Cleaning

Dirt that gets accumulated on a treadmill can also cause malfunction. Dirt on the belt or the deck is cleared by wiping the belt and the sides of the treadmill once or twice a month using a cloth or a wet sponge.

Belt maintenance

The conveyor belt is an important part that is responsible for the functioning of the treadmill. Regular maintenance for the belt includes

1. Lubrication
2. Alignment
3. Tension Maintenance

Chapter 8

CONCLUSIONS AND SCOPE FOR FUTURE WORK

CONCLUSIONS

In this project, a new way of travelling as well as exercise with the help of a new model of Cycle which is combination of treadmill and Cycle is discussed. It can be used in place of regular bike at cheaper cost and without use of fuel. The treadmill Cycle will proof to be a future vehicle as no fuel is used for travelling through this and it is pollution free. The treadmill which is used for walking helps to keep us fit as exercise is also one of the important task for a person to be fit and healthy for day to day life. Treadmill cycle is cheaper than the normal bike which also makes it efficient and economic.

Exercise

Treadmill Cycle helps in maintaining proper physique. As physical fitness is important in day to day life. By using treadmill Cycle one can exercise outdoors in fresh air.

Fuel saving

People often use vehicle for travelling over short distance. This causes unnecessary wastage of fuel. Due to use of treadmill Cycle over short distance a large amount of fuel can be saved.

Travelling

Treadmill cycle can be used for travelling over short distances. One can also exercise while travelling over short distance.

Eco- friendly

Treadmill Cycle does not require any fuel. Therefore, it does not emit any pollutants. So, it is an eco-friendly vehicle.

FUTURE SCOPE

In future, the Cycle can be run by the electrical energy stored previously in the battery. So, after travelling manually to destination, the return travel can be done by electrical power. It is the Effective way to convert human energy into electrical power while exercising. Combination of two or more process in a single system that makes time conservation in busy life.

REFERENCES

1. The Design of Treadmill with Distributed Generation Function Yingbo Yu ; Haohua Qin 2019 4th International Conference on Mechanical, Control and Computer Engineering (ICMCCE) Year: 2019 | Conference Paper | Publisher: IEEE

2. An innovative technique of electricity generation and washing machine application using treadmillSahil ; P.K. Sharma ; N. Hari ; N. Kumar ; D. Shahib 2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES)
3. Manish Debnath, "Generation of electricity by running on a leg-powered treadmill" International Journal of Latest Research in Engineering and Technology (IJLRET) ISSN: 2454-5031(Online) www.ijlret.com‖ Volume 1 Issue 7‖December 2015 ‖ PP 04-07
4. KirtishBondre, SanketBeradpatil, S. J. Thorat, "Design and Fabrication of Treadmill bicycle" International Journal of Innovative Research in Science, Engineering and Technology (An ISO 3297: 2007 Certified Organization) Vol. 5, Issue 6, June 2016
5. Adeel Ansari, Noman Raza, Farooqui Sameer, Zohaib Shaikh, Professor Arshad Rashid "Treadmill Bike" International Journal of Modern Trends in Engineering and Research e-ISSN No.:2349-9745, Date: 28-30 April, 2016
6. Caspersen CJ, Powell KE, Christenson GM. Physical activity, exercise, and physical fitness: definitions and distinctions for health-related research. Public Health Reports. 1985; 100(2):126-131.
7. Wolthuis, R. A., Froelicher, V. F., Fischer, J., Noguera, I., Davis, G., Stewart, A. J., &Triebwasser, J. H.(1977). New practical treadmill protocol for clinical use. The American journal of cardiology, 39(5), 697-700.
8. Kooijman, J. D. G., and A. L. Schwab. "Experimental validation of the lateral dynamics of a bicycle on a treadmill." ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers, 2009.
9. Kisan, Ravikiran, et al. "Treadmill and bicycle ergometer exercise:
10. Design data book -P.S.G.Tech.
11. Machine tool design handbook –Central machine tool Institute, Bangalore.