

HEALTHCARE MANAGEMENT FOR PREGNANT LADY USING IOT

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

KM KAARTHIK (Reg. No. 37110302)

HEMANTH REDDY M (Reg. No. 37110274)



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(*DEEMED TO BE UNIVERSITY*)

Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI
SALAI, CHENNAI – 600 119

MARCH - 2021



SATHYABAMA



INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with "A" grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **KM KAARTHIK (Reg. No. 37110302)** and **HEMANTH REEDY M (Reg No. 37110274)** who carried out the project entitled "**HEALTH CARE MANAGEMNET FOR PREGNANT LADY USING**

IOT" under my supervision from **August 2020** to **March 2021**.

Internal Guide

Dr. P. Asha., M.E., Ph.D.,

Head of the Department

Dr. S. Vigneswari, M.E., Ph.D.,

Dr. L. Lakshmanan, M.E., Ph.D.,

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I **KM KAARTHIK** hereby declare that the Project Report entitled “**HEALTH CARE MANAGEMENT FOR PREGNANT LADY USING IOT**” is done by me under the guidance of **Mrs. DR.P.ASHA, M.E., PH.D.**, Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D.,** and **Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Mrs. Dr. P.ASHA, M.E., Ph.D., Professor**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

HEALTHCARE MANAGEMENT FOR PREGNANT LADY USING IOT

ABSTRACT

Nowadays woman percentage was literally decreasing all over the world and this leads to face many problems in the future. According to many surveys, women are dying in their pregnancy period due to the less care and negligence in her medication, particularly pregnant women should have to be monitored or checked every hour and her health condition should be noted to move on to further process. Generally, women in developed areas are being checked regularly during their pregnancy times so that they can easily receive fine medication from the doctors and the respective hospitals. Pregnant women should have to be monitored every second because no one can predict the ladies situation how it changes in a serious way. But we should have to predict her health condition to know her graph about her health that's how we can manage any medicine for her, so we should be attentive in device that monitor's pregnant lady condition that we can receive her health condition continuously so that we will get the alerts when her health condition was serious.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No.
	ABSTRACT	I
	LIST OF FIGURES	II
1.	INTRODUCTION	1
2.	LITERATURE SURVEY	3
3.	AIM & SCOPE OF THE INVESTIGATION	7
	3.1. Overview of Embedded Systems	7
	3.2. Characteristics of Embedded Systems	9
	3.2.1. Reactive Systems	9
	3.2.2. Distributed Systems	10
	3.2.3. Heterogeneous Architectures	10
	3.2.4. Harsh environment	11
	3.3. System Safety and Reliability	11
	3.4. Control of Physical Systems	11
	3.4.1. Small and low weight	12
	3.4.2. Cost sensitivity	12
	3.4.3. Power management	12
	3.5. Power Supply	12
	3.5.1. Transformer	13

	3.5.2. Rectifier	13
	3.5.3. Filters	13
	3.5.4. Registers	13
	3.5.5. Regulators	14
	3.6. Features and Description of Regulators	14
	3.7. Existing System	14
	3.8. Proposed System	15
4.	EXPERIMENTAL METHODS	16
	4.1. Software	16
	4.2. Hardware	17
	4.2.1. Pin Definition	18
	4.2.2. Blink Test	22
	4.2.3. Flashing Nodemcu	24
	4.3. Arduinouno and its programming 25	
	4.3.1. Overview	26
	4.3.2 Pin Configuration	28
	4.3.3. Input and Output	29
	4.3.4. Communication	30
	4.4. Arduino Software	31
	4.4.1. Humidity sensor	36
	4.4.2. Temperature sensor	36

	4.4.3. Pressure sensor	39
	4.4.4. Respiratory sensor	40
	4.5. Motivation	41
	4.6. IoT	41
	4.7. Heartbeat Sensor	42
	4.7.1. Principle of a Heartbeat Sensor	43
	4.7.2. Working of a Heartbeat Sensor	43
	4.7.3. Practical Heartbeat sensor	44
5.	RESULTS & DISCUSSION	46
	5.1. Data Collection	47
	5.2. Graphs	48
6.	SUMMARY & CONCLUSION	49
7.	REFERENCES	50
8.	APPENDIX	53
	A. Source code	53
	B .Screenshots	58

LIST OF FIGURES

FIGURE	FIGURE NAMES
3.1	BLOCK DIAGRAM OF TYPICAL EMBEDDED SYSTEM
3.2	EMBEDDED SYSTEMS HAVING HETEROGENEOUS ARCHITECTURES
3.3	GENERAL BLOCK OF POWER SUPPLY UNIT
4.1	ARCHITECTURE OF HEALTH MOTORING SYSTEM
4.2	NODEMCU
4.3	TCP CONNECTIONS
4.4	ARDUINO PAKAGE
4.5	PREFERENCES
4.6	SETUP
4.7	CPU FREQUENCY
4.8	RST BUTTON
4.9	ESP BOARD
4.10	ESP8266 USING WINDOWS
4.11	ESP8266 BREADBOARD WITH HEAD CONNECTOR
4.12	PIN ASSIGNMENT
4.13	INTERNAL CABLES
4.14	ARDUINO

4.15	ARDUINO BOARD
4.16	ARDUINO IDE
4.17	COM PORT
4.18	TOOL SELECTION
4.19	SKETCH OF ARDUINO
4.20	UPLOADING ARDUINO SKETCH TO ARDUINO BOARD
4.21	TEMPERATURE SENSOR
4.22	DIGITAL SIGNAL
4.23	TRANSMITTING SIGNAL
4.24	TIME DIFFERENCE FOR TRANSMITTING "1's" & "0's"
4.25	PRESSURE SENSOR
4.26	RESPIRATORY SENSOR
4.27	AC SIGNAL
4.28	HEARTBEAT SENSOR
4.29	DATABASE
4.30	GOOGLE LOCATION
4.31	GRAPHICAL STORAGE OF DATA
4.32	HEARTBEAT ALERT
4.33	TEMPERATURE ALERT
4.34	RESPIRATORY ALERT

CHAPTER 1

INTRODUCTI

ON

More time taking and long-distance traveling is difficult for pregnant women. Because pregnant women living in backward areas and mostly they cannot receive any immediate healthcare in serious conditions. A frequent work has been going on and few methods are implemented to check the continues health condition of the pregnant lady. A remote-controlled device is introduced for the health monitoring of the pregnant lady and that by using that we can easily save her in serious conditions. Pregnant women healthcare is a type of taking care of both the mother and the baby who already growing in her womb. It's a risky task for taking care of the pregnant lady because of the sudden issues of her health lead to death.

The pregnant women's healthcare system is crucial and IoT is open deals with this type of methods so this helps to know and find the accurate values of the ladies' health condition. This system describes the design and implementation part of wearable with the smart phone for the real time monitoring of health. In this system smart shirt are developed with sensors and can be worn by any type of patient for monitoring his or her health in real time and get required treatment or prescription. These systems are mainly developed considering elder people in mind as they live alone in their homes. Therefore this system basically monitors the elderly people for self diagnosis purpose.

The result of this system was the system could monitor and diagnose patients' heart conditions in real time, when they wear a sports-shirt with a ECG sensor in it. In addition to this, the system also provides graphical information with history management tools and an automatic emergency call system to the patient to get the required treatment in time. The drawback of the system is that it only concentrates on elder people and it includes glouse for wearing which is used in this system, we can arrange temperature, pressure, respiratory and heartbeat sensor that which helps to monitor the pregnant ladies health situation in every second that we can monitor everything in the serial monitor that directly functions in the Arduino application which is connected to the Wi-Fi called Nodemcu and it helps to store the data to the Ubdots storage.

OBJECTIVE

This project proposed to construct a health monitoring system that would allow a pregnant woman in the rural areas to interact with a Doctor/ Physician with almost full functional capability. Illiteracy in women and lack of access to health facilities in many rural areas increases the risk of death for child-bearing women. Preventive measures by constant health monitoring of patient at early stages and guidance to prevent costly hospitalization to avoid such risks is the main objective of the project. The health monitoring device suggested in this project constantly measures the Temperature and Heart Beat of a pregnant Lady and whenever there are fluctuations from the normal value GSM sends the information to gynaecologist at remote place through her mobile. Also, this device allows the user to hear the advices and diet prescriptions as advised by the doctor. This system operates as a health monitoring unit, but the user has the access to hear the advices and diet prescriptions as advised by the doctor.

CHAPTER 2

LITERATURE

SURVEY

1. Health Monitoring Laboratories by Interfacing Physiological Sensors to Mobile Android Devices

This paper describes, Android Java-DSP (AJDSP) as a mobile application that interfaces with sensors and enables simulation. This also helped in visualization of signal processing. In this system firstly there is creation of interface between both external sensors and on-board device sensors for monitoring the physiological parameter of human being. This paper also explored the trend of mobile sensing and adapted it towards improving digital signal processing (DSP), by building interfaces to medical sensor and external sensors. In this paper there is use of SHIMMER. It is a small wireless low power sensor International Journal of Advance Foundation and Research in Computer platform.

2. Smart Elderly Home Monitoring System with an Android Phone

This paper considers or takes into account certain facts, which are heart attack and stroke as they are the major cause of hospitalization of the elder people. There is more chances of survival if the older people gets the treatment within an hour. it had also been developed. An android smart phone with accelerometer is used to detect a fall of the carrier, and this android device is known as healthcare device. The android phone is then connected to the monitoring system by using the TCP/IP protocol through Wi-Fi. Because of this system, elderly and chronically ill patients can stay independently in their own home and secure in the knowledge that they are being monitored

3. Design and Implementation of Wearable ECG System

This system describes the design and implementation part of wearable ECG with the smart phone for the real time monitoring of health. In this system smart shirt are developed with ECG sensors and can be worn by any type of patient for monitoring his or her health in real time and get required treatment or prescription. These systems are mainly developed considering elder people in mind as they live alone in their homes.

Therefore this system basically monitors the elderly people for self diagnosis purpose. The result of this system was the system could monitor and diagnose patients' heart conditions in real time, when they wear a sports-shirt with a ECG sensor in it. In addition to this, the system also provides graphical information with history management tools and an automatic emergency call system to the patient to get the required treatment in time.

4. Remote patient monitoring system

This paper provides the image based system which acquires the ECG signal via digital camera; this information is performed on the tool like MATLAB and data sending through the internet network and stored in database. Then the original image is then availed to the doctor via Android mobiles. The purpose of this system is the vital signs and parameters from the ICU monitoring system and makes this data to be available to the doctor who may not be in the hospital and in the country. In case of any abnormality, the doctor is alerted by sending a notification from C2DM server to his mobile. The drawback of this paper is that, due to the slow internet connection the data will not be send to the doctor which is located remotely. The image is captured through the camera, which must be HD which cost a lot.

5. Hospital Health Care Monitoring System using Wireless Sensors Network

In this paper there is continuous observation of the patient's physiological parameters such as blood pressure of patient as well as heart rate. This system is mainly useful for pregnant women to measure the various parameters like blood pressure, heart beat and fetal movement to control the health issue. This system has to monitoring more than one patient at a time and easily able to sense the blood pressure (BP) and heart rate of the patients. In this system, there is a sensor node attached to body of patient to measure signals from the wireless sensors and sends these signal to the database. This system can detect the abnormal conditions of the patient, raise an alarm to the patient and sends a SMS/Email to the doctor for treatment.

6. Wireless Health and the Smart Phone Conundrum

This paper tells us that, they implemented wireless healthcare by sensor communication, data processing, data visualization etc on different mobile platforms. There is central controller for wireless health application. They developed libraries for particular mobile devices to implement the health care on different platforms. CDMA and UTMS have more network than Bluetooth and Wi-Fi, and all this cellular connectivity is supported by mobile platforms. They started by, assessing the best smart phone platform for the health care, next they determine the runtime environment to the smart phone platform and finally they developed the wireless health application for that particular platform. They provide different and better debugging environment for different platforms. The drawback of this paper is that they do not represent the complete wireless health software library.

7. Android ECG application development

The paper describes the design and implementation part of ECG application. The main purpose or objective of this paper was to use the power of android platform and produce a prototype model of ECG application. It performs its works with VS100, Bluetooth ECG monitoring device. This paper also describes the architecture, development environment and tools that are used in android mobile application development. The tools used in development process or phase are Eclipse, Android SDK, ADT plugging for Eclipse. This device monitors Heart Rate via Bluetooth communication. The result of this paper was communicating with vital sense device via Bluetooth communication.

8. Advanced IoT based combined remote health monitoring, home automation, and alarm system:

Jayeeta Saha, Arnab Kumar Saha, Aiswarya Chatterjee,
Suyash Agrawal, Ankita Saha, Avirup Kar, Himadri Nath Saha

Nowadays, automation and the Internet of Things are changing the world. The day-by-day development of the Internet of Things causes a revolution in modern technology, which makes our life easier and automated. Due to a busy schedule and irregular lifestyle, the health hazard is not an age-dependent factor in the recent era. Under these

circumstances, the Internet of Things has provided a much easier solution for remote real-time health monitoring of patients from the hospital as well as home. Sensors acquire the data of various parameters regarding patients' health, and the Internet of Things stores that data and displays through the website, which provides access for remote monitoring. The use of Sensor reduces the human error, and the size of the system reduces the occupied space of the room. The unique part of this proposed solution is the alarm generation to provide the prescribed medicine to the patient in time. The other beneficial area of the system is the scheme of sending the notification through email and SMS alert if any of the health parameters crosses the threshold value.

CHAPTER 3

AIM & SCOPE OF THE PRESENT INVESTIGATION

AIM & SCOPE

The present project monitors the heart beat and temperature of mother but not the fetus. So the upcoming projects can take a look at measuring heart beat and temperature of both mother and fetus. Hence both can step out of danger if there is any problem during pregnancy. We used zigbee in this project the future projects can use the higher version of connectivity. This project is open for developments from all sides. It is user's imagination which limits the working of this project. One can go on adding the extra, rich features to this project.

OVERVIEW OF EMBEDDED SYSTEMS

An embedded system is a special-purpose computer system designed to perform one or a few dedicated functions, often with real-time computing constraints. It is usually embedded as part of a complete device including hardware and mechanical parts. In contrast, a general-purpose computer, such as a personal computer, can do many different tasks depending on programming. Embedded systems have become very important today as they control many of the common devices we use.

Since the embedded system is dedicated to specific tasks, design engineers can optimize it, reducing the size and cost of the product, or increasing the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale. Physically, embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, or the systems controlling nuclear power plants. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

In general, "embedded system" is not an exactly defined term, as many systems have some element of programmability. For example, Handheld computers share some elements with embedded systems such as the operating systems and microprocessors which power them but are not truly embedded systems, because they allow different

applications to be loaded and peripherals to be connected. Embedded systems provide several functions

- Monitor the environment; embedded systems read data from input sensors.

This data is then processed and the results displayed in some format to a user or users

- Control the environment; embedded systems generate and transmit commands for actuators.

- Transform the information; embedded systems transform the data collected in some meaningful way, such as data compression/decompression

Although interaction with the external world via sensors and actuators is an important aspect of embedded systems, these systems also provide functionality specific to their applications. Embedded systems typically execute applications such as control laws, finite state machines, and signal processing algorithms. These systems must also detect and react to faults in both the internal computing environment as well as the surrounding electromechanical systems.

There are many categories of embedded systems, from communication devices to home appliances to control systems. Examples include;

- Communication devices

e.g.: modems, cellular phones

- Home Appliances

e.g.: CD player, VCR, microwave oven

- Control Systems

e.g.: Automobile anti-lock braking systems, robotics, satellite control.

BLOCK DIAGRAM OF EMBEDDED SYSTEM

An embedded system usually contains an embedded processor. Many appliances that have a digital interface microwaves, VCRs, cars utilize embedded systems. Some embedded systems include an operating system. Others are very specialized resulting in the entire logic being implemented as a single program. These systems are embedded into some device for some specific purpose other than to provide general purpose computing . A typical embedded system is shown in Fig 3.1

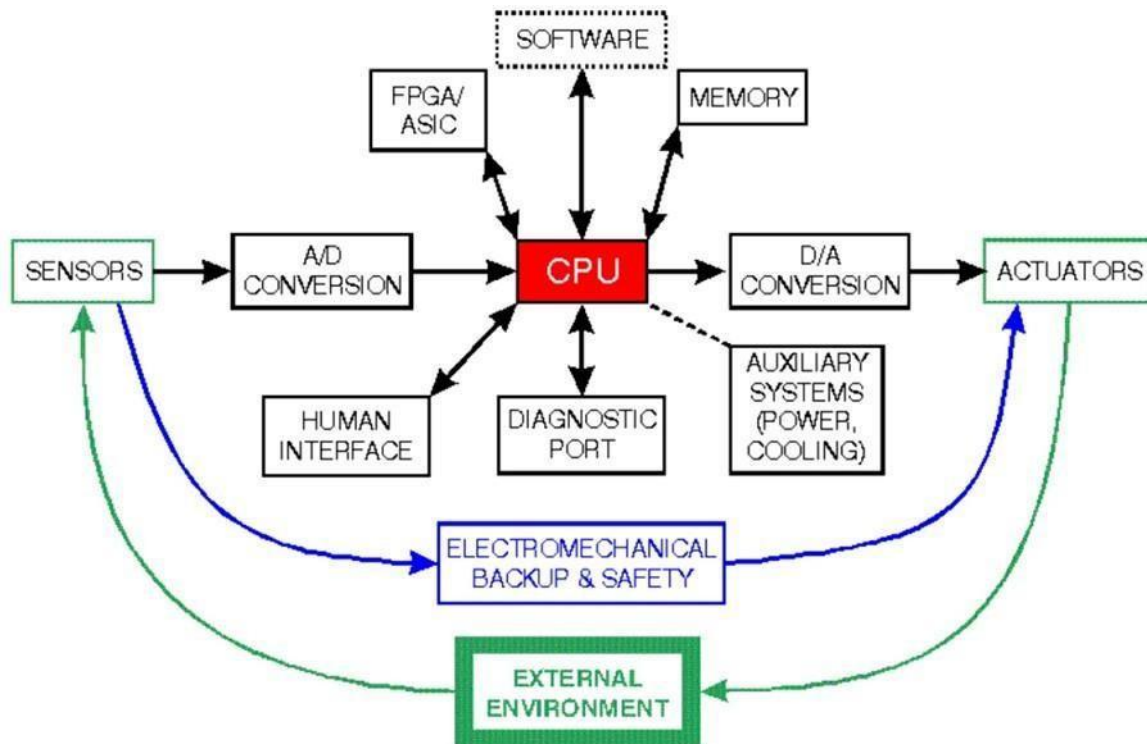


Fig 3.1 Block diagram of a typical embedded

system CHARACTERISTICS OF EMBEDDED SYSTEMS

Embedded systems are characterized by a unique set of characteristics. Each of these characteristics imposed a specific set of design constraints on embedded systems designers. The challenge to designing embedded systems is to conform to the specific set of constraints for the application.

Application Specific Systems

Embedded systems are not general-purpose computers. Embedded system designs are optimized for a specific application. Many of the job characteristics are known before the hardware is designed. This allows the designer to focus on the specific design constraints of a well-defined application. As such, there is limited user reprogram ability. Some embedded systems, however, require the flexibility of reprogram ability. Programmable DSPs are common for such applications.

Reactive Systems

As mentioned earlier, a typical embedded systems model responds to the

environment via sensors and control the environment using actuators. This requires embedded systems to run at the speed of the environment. This characteristic of embedded system is called “reactive”. Reactive computation means that the system (primarily the software component) executes in response to external events. External events can be either periodic or aperiodic. Periodic events make it easier to schedule processing to guarantee performance. Aperiodic events are harder to schedule. The maximum event arrival rate must be estimated in order to accommodate worst case situations. Most embedded systems have a significant reactive component.

Distributed Systems

A common characteristic of an embedded system is one that consists of communicating processes executing on several CPUs or ASICs which are connected by communication links. The reason for this is economy. Economical 4 8-bit microcontrollers may be cheaper than a 32-bit processors. Even after adding the cost of the communication links, this approach may be preferable. In this approach, multiple processors are usually required to handle multiple time-critical tasks. Devices under control of embedded systems may also be physically distributed.

Heterogeneous Architectures

Embedded systems often are composed of heterogeneous architectures (Fig 3.2). They may contain different processors in the same system solution. They may also be mixed signal systems.



Fig 3.2. Embedded Systems having Heterogeneous Architectures

Harsh environment

Many embedded systems do not operate in a controlled environment. Excessive heat is often a problem, especially in applications involving combustion (e.g., many transportation applications). Additional problems can be caused for embedded computing by a need for protection from vibration, shock, lightning, power supply fluctuations, water, corrosion, fire, and general physical abuse.

SYSTEM SAFETY AND RELIABILITY

As embedded system complexity and computing power continue to grow, they are starting to control more and more of the safety aspects of the overall system. These safety measures may be in the form of software as well as hardware control. Mechanical safety backups are normally activated when the computer system loses control in order to safely shut down system operation. Software safety and reliability is a bigger issue. Software doesn't normally "break" in the sense of hardware. However software may be so complex that a set of unexpected circumstances can cause software failures leading to unsafe situations. Discussion of this topic is outside the scope of this book, but the challenges for embedded designers include designing reliable software and building cheap, available systems using unreliable components. The main challenge for embedded system designers is to obtain low-cost reliability with minimal redundancy.

CONTROL OF PHYSICAL SYSTEMS

One of the main reasons for embedding a computer is to interact with the environment. This is often done by monitoring and controlling external machinery. Embedded computers transform the analog signals from sensors into digital form for processing. Outputs must be transformed back to analog signal levels. When controlling physical equipment, large current loads may need to be switched in order to operate motors and other actuators. To meet these needs, embedded systems may need large computer circuit boards with many non-digital components.

Small and low weight

Many embedded computers are physically located within some larger system. The form factor for the embedded system may be dictated by aesthetics. For example, the form factor for a missile may have to fit inside the nose of the missile. One of the challenges for embedded systems designers is to develop non-rectangular geometries for certain solutions. Weight can also be a critical constraint. Embedded automobile control systems, for example, must be light weight for fuel economy. Portable CD players must be light weight for portability purposes.

Cost sensitivity

Cost is an issue in most systems, but the sensitivity to cost changes can vary dramatically in embedded systems. This is mainly due to the effect of computer costs have on profitability and is more a function of the proportion of cost changes compared to the total system cost.

Power management

Embedded systems have strict constraints on power. Given the portability requirements of many embedded systems, the need to conserve power is important to maintain battery life as long as possible. Minimization of heat production is another obvious concern for embedded systems.

POWER SUPPLY

All electronic circuits works only in low DC voltage, so we need a power supply unit to provide the appropriate voltage supply for their proper functioning .This unit consists of transformer, rectifier, filter & regulator. AC voltage of typically 230volts rms is connected to a transformer voltage down to the level to the desired ac voltage. A diode rectifier that provides the full wave rectified voltage that is initially filtered by a simple capacitor filter to produce a dc voltage. This resulting dc voltage usually has some ripple or ac voltage variation . A regulator circuit can use this dc input to provide dc voltage that not only has much less ripple voltage but also remains the same dc value even the dc voltage varies some what, or the load connected to the output dc voltages changes.

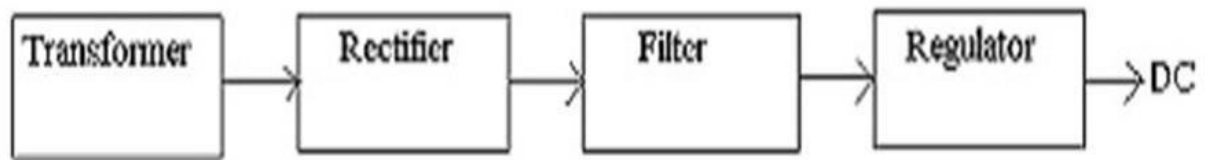


Fig.3.3. General Block Of Power Supply Unit

Transformer

A transformer is a static piece of which electric power in one circuit is transformed into electric power of same frequency in another circuit. It can raise or lower the voltage in the circuit, but with a corresponding decrease or increase in current. It works with the principle of mutual induction. In our project we are using a step down transformer to providing a necessary supply for the electronic circuits. Here we step down a 230volts ac into 12volts ac.

Rectifier

A dc level obtained from a sinusoidal input can be improved 100% using a process called full wave rectification. Here in our project for full wave rectification we use bridge rectifier. From the basic bridge configuration we see that two diodes(say D2 & D3) are conducting while the other two diodes (D1 & D4) are in off state during the period $t = 0$ to $T/2$. Accordingly for the negative cycle of the input the conducting diodes are D1 & D4. Thus the polarity across the load is the same.

In the bridge rectifier the diodes may be of variable types like 1N4001, 1N4003, 1N4004, 1N4005, 1N4007 etc.... can be used. But here we use 1N4007, because it can withstand up to 1000v.

Filters

In order to obtain a dc voltage of 0 Hz, we have to use a low pass filter. So that a capacitive filter circuit is used where a capacitor is connected at the rectifier output & a dc is obtained across it. The filtered waveform is essentially a dc voltage with negligible ripples & it is ultimately fed to the load.

Registers

The controller IC has two 8 bit registers, an instruction register (IR) and a data register (DR). The IR stores the instruction codes and address information for display

data RAM (DD RAM) and character generator RAM (CG RAM). The IR can be written, but not read by the MPU. The DR temporally stores data to be written to /read from the DD RAM or CG RAM. The data written to DR by the MPU, is automatically written to the DD RAM or CG RAM as an internal operation. Since all the functions such as display RAM, character generator, and liquid crystal driver, required for driving a dot-matrix liquid crystal display are internally provided on one chip, a minimal system can be interfaced with this controller/driver.

Regulators

The output voltage from the capacitor is more filtered & finally regulated. The voltage regulator is a device, which maintains the output voltage constant irrespective of the change in supply variations, load variations & temperature changes. Here we use fixed voltage regulator namely LM7805. The IC LM7805 is a +5v regulator which is used for microcontroller.

FEATURES AND DESCRIPTION OF REGULATORS

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24V
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

EXISTING SYTEM

In the existing system, when the pregnant woman faces the problem with her health she has to take some precautions and she can able to send some alerts messages or at least she should have to inform the family members when they are alone in the house. If the women were unable to inform that will be so serious for her. So this is considered one of the drawbacks.

PROPOSED SYSTEM

The work attempts at monitoring the health condition of a pregnant women by every second and the received data stores in the database which we have created and controlled privately that no one can check her results. Sensors are connected to the lady which is connected to the network and that this works to monitor and analysis the pregnant ladies health, these sensors are used to measure and give accurate values and graphs with all limitations like body temperature and Heartbeat rate and handover so that the patient's situation can be studied by the family members so that they can decide to take the pregnant lady to hospital or not. Pregnant ladies can't move from one place to another place fast and regularly to hospitals to consult a doctor in the starting time of pregnancy. So by using this process we can monitor and check the pregnant lady's health can store the received health data in the Ubdots database. This particular temperature, heartbeat sensors are monitored and noted privately with the exact and accurate values and these sensors are considered as important functioning sensors in the health monitoring process. This project paper basically provides a wearable device that will continuously monitor the important parameters for a patient and do data storing continuously. If any dangerous condition arises for a patient we will get the G-mail alert to the specific person who has connected with the device in online in Ubdots storage along with the GPS location, the data stores and transfer the information to the family or doctor and that helps to rescue the pregnant lady, because of IoT and the product is compact and also a wearable. In the pregnant women's health and wellness observation system, it would be guaranteed that the right Client is being detected, that information is forwarded out to the accurate health information systems.

A wearable sensor measures the health limitations enthusiastically and is connected according to the concept of IPv6 wireless personal network, to the node for wireless transmission through the internet.

CHAPTER 4

EXPERIMENTAL METHODS

ARCHITECHTURE

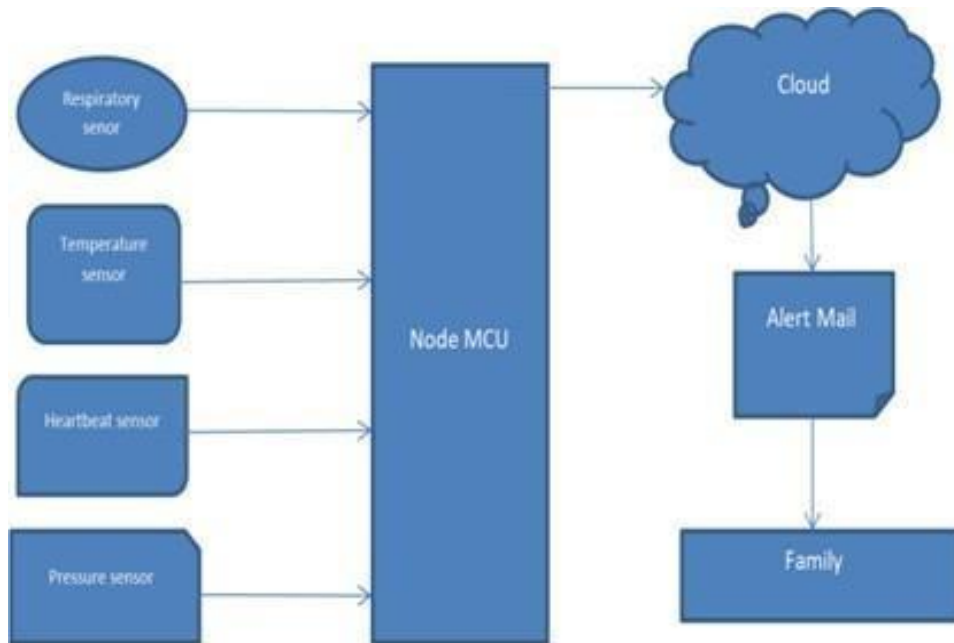


Fig. 4.1. Architecture of Health monitoring system

- Pressure sensor
- Temperature sensor
- Heart beat sensor
- Respiratory sensor

SOFTWARE

(i) Arduino ide:

The Arduino core functions are simply a set of C++ classes and libraries you can use. It is built and compiled using the GNU gcc/g++ compiler. Your setup() and loop() functions are simply placed into the mandatory.

(ii) Embedded c:

Embedded C is a generic term given to a programming language written in C, which is associated with a particular hardware architecture. Embedded C is an extension to the C language with some additional header files. These header files may change from controller to controller.

HARDWARE

ESP8266 Nodemcu Wi-Fi Device kit



Fig. 4.2. Node mcu

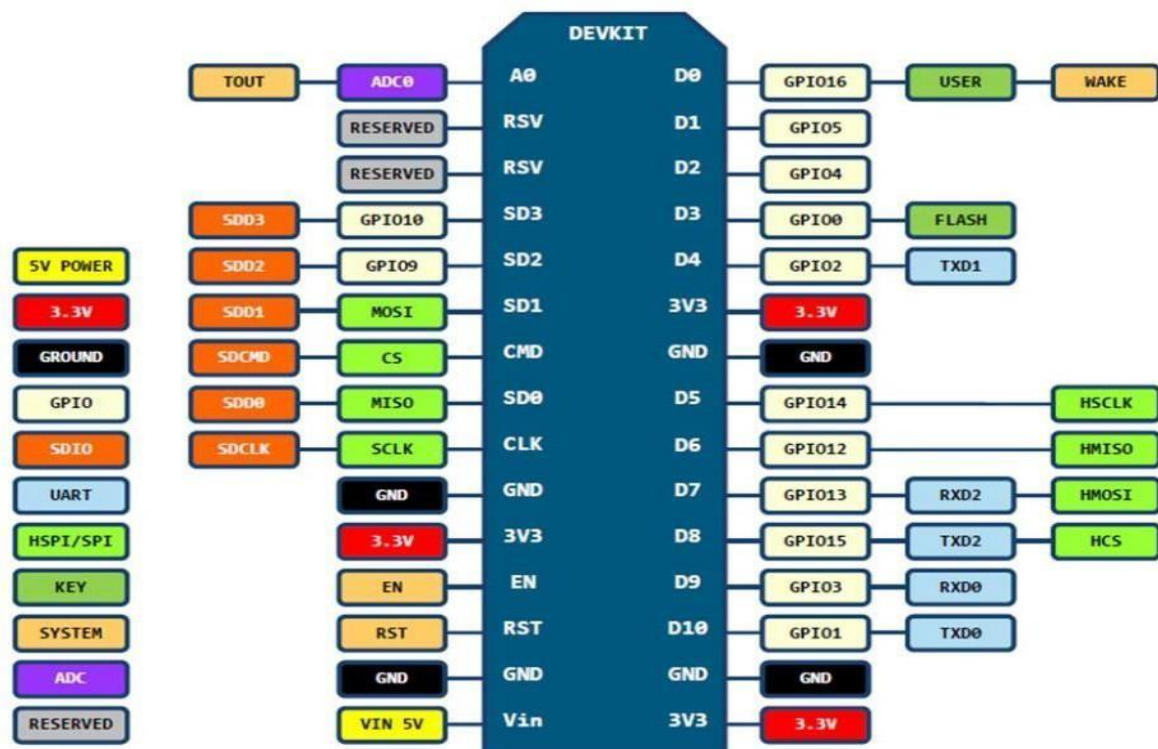
Specification:

- Voltage: 3.3V.
- Wi-Fi Direct (P2P), soft-AP.
- Current consumption: 10uA~170mA.
- Flash memory attachable: 16MB max (512K normal).
- Integrated TCP/IP protocol stack.

- Processor: Tensilica L106 32-bit.
- Processor speed: 80~160MHz.
- RAM: 32K + 80K.
- GPIOs: 17 (multiplexed with other functions).
- Analog to Digital: 1 input with 1024 step resolution. • +19.5dBm output power in 802.11b mode
- 802.11 support: b/g/n.

Maximum concurrent TCP connections

Pin Definition



D0(GPIO16) can only be used as gpio read/write, no interrupt supported, no pwm/i2c/ow supported.

Fig.4.3. TCP connections

Using Arduino IDE

The most basic way to use the ESP8266 module is to use serial commands, as the chip is basically a WiFi/Serial transceiver. However, this is not convenient.

What we recommend is using the very cool Arduino ESP8266 project, which is a modified version of the Arduino IDE that you need to install on your computer. This

makes it very convenient to use the ESP8266 chip as we will be using the well-known Arduino IDE. Following the below step to install ESP8266 library to work in Arduino IDE environment.

Install the ESP8266 Board Package

Enter

http://arduino.esp8266.com/stable/package_esp8266com_index.json into Additional Board Manager URLs field in the Arduino v1.6.4+ preferences.

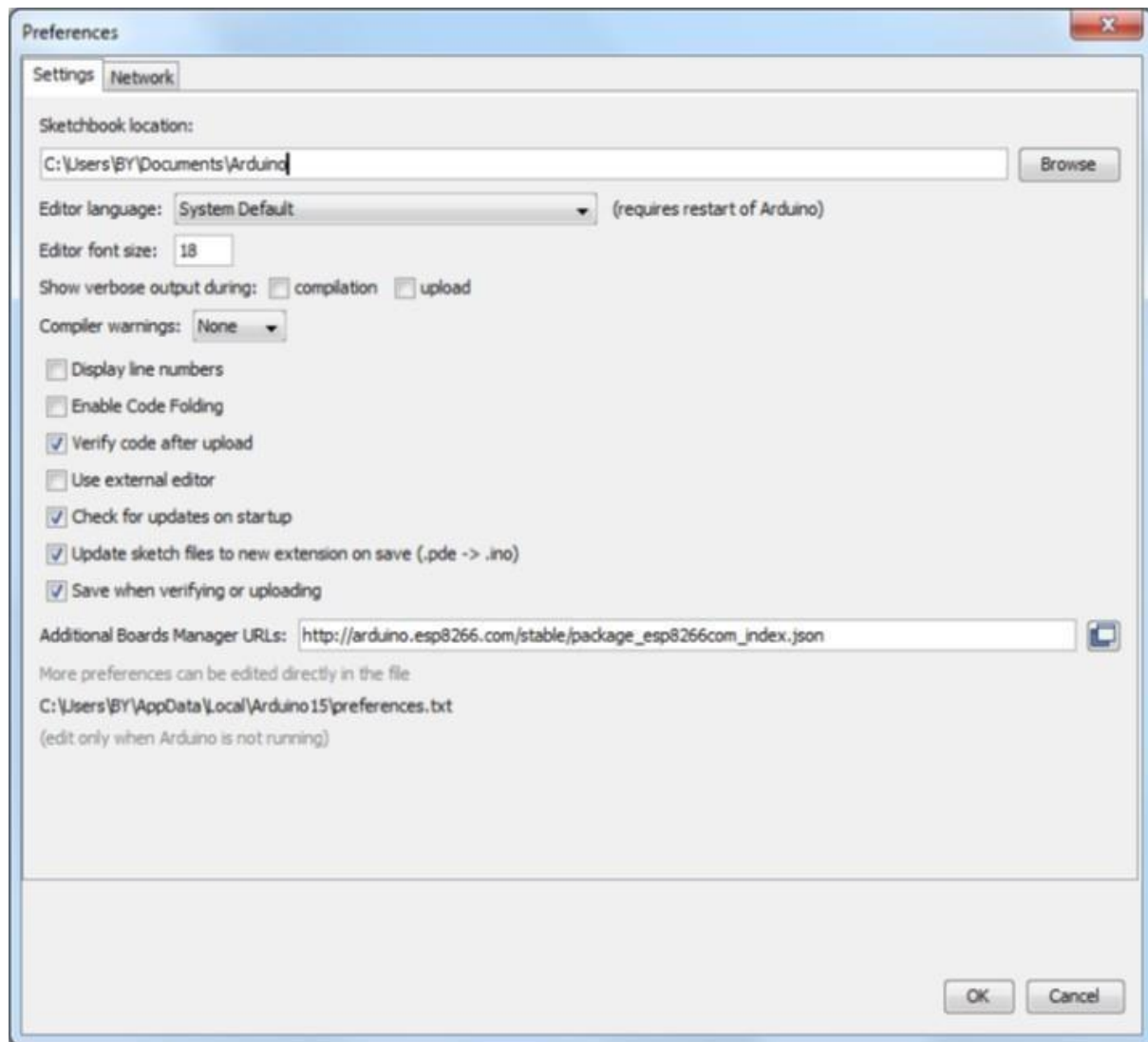


Fig4.4. http://arduino.esp8266.com/stable/package_esp8266com_index.json

Click 'File' -> 'Preferences' to access this panel. Next, use the Board manager to install the ESP8266 package.

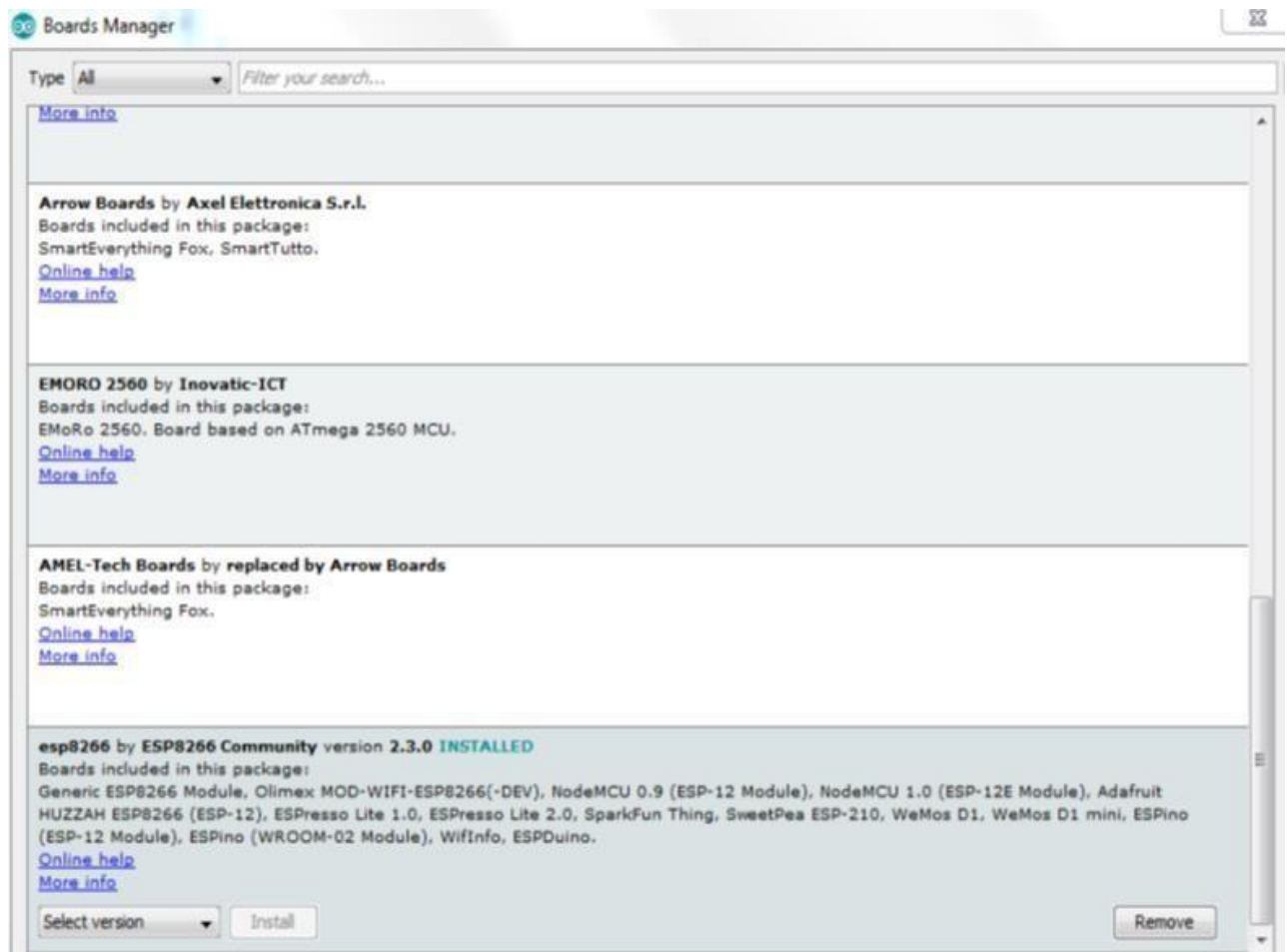


Fig.4.5. Preferences

Click 'Tools' -> 'Board:' -> 'Board Manager...' to access this panel. Scroll down to 'esp8266 by ESP8266 Community' and click "Install" button to install the ESP8266 library package. Once installation completed, close and re-open Arduino IDE for ESP8266 library to take effect.

Setup ESP8266 Support

When you've restarted Arduino IDE, select 'Generic ESP8266 Module' from the 'Tools' -> 'Board:' dropdown menu

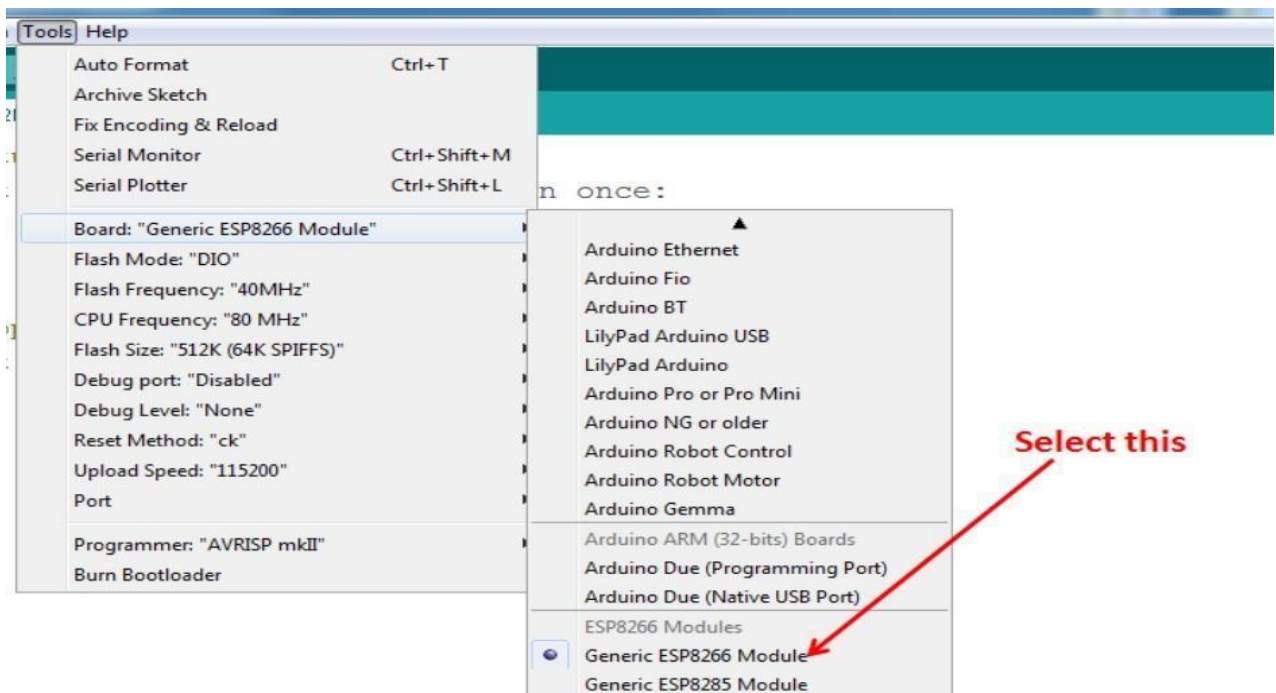


Fig.4.6. Setup ESP8266 Support

Select 80 MHz as the CPU frequency (you can try 160 MHz overclock later)

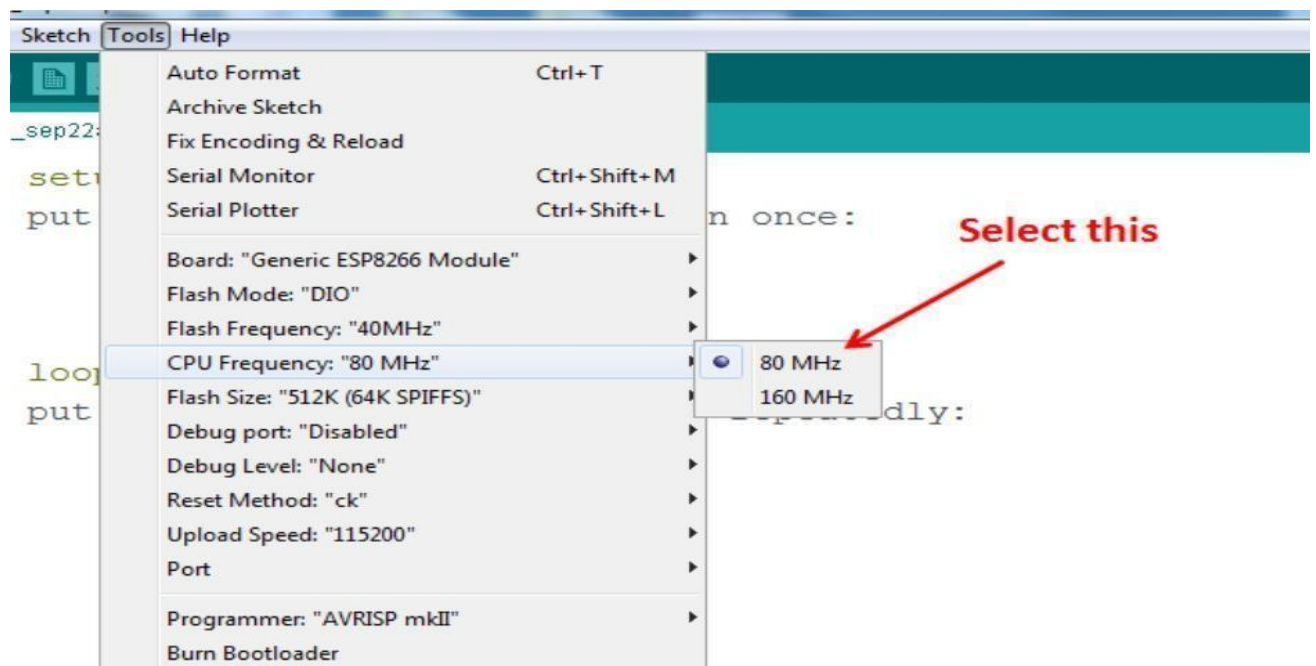


Fig.4.7. CPU frequency

Blink Test

We'll begin with the simple blink test. Enter this into the sketch window (and save since you'll have to). Connect a LED as shown in Figure3-1.

Now you'll need to put the board into bootload mode. You'll have to do this before each upload. There is no timeout for bootload mode, so you don't have to rush.

- Hold down the 'Flash' button.
- While holding down 'Flash', press the 'RST' button.
- Release 'RST', then release 'Flash'

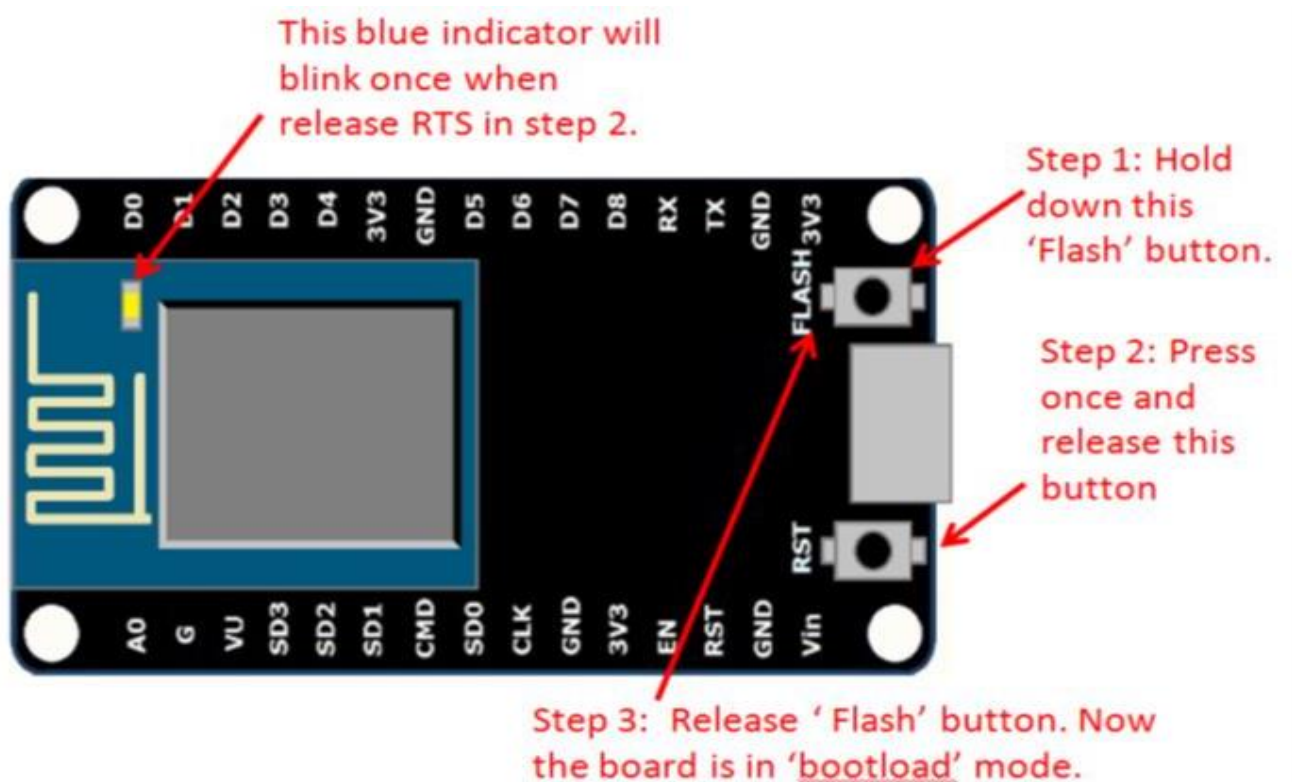


Fig.4.8. RST button

Once the ESP board is in bootload mode, upload the sketch via the IDE,

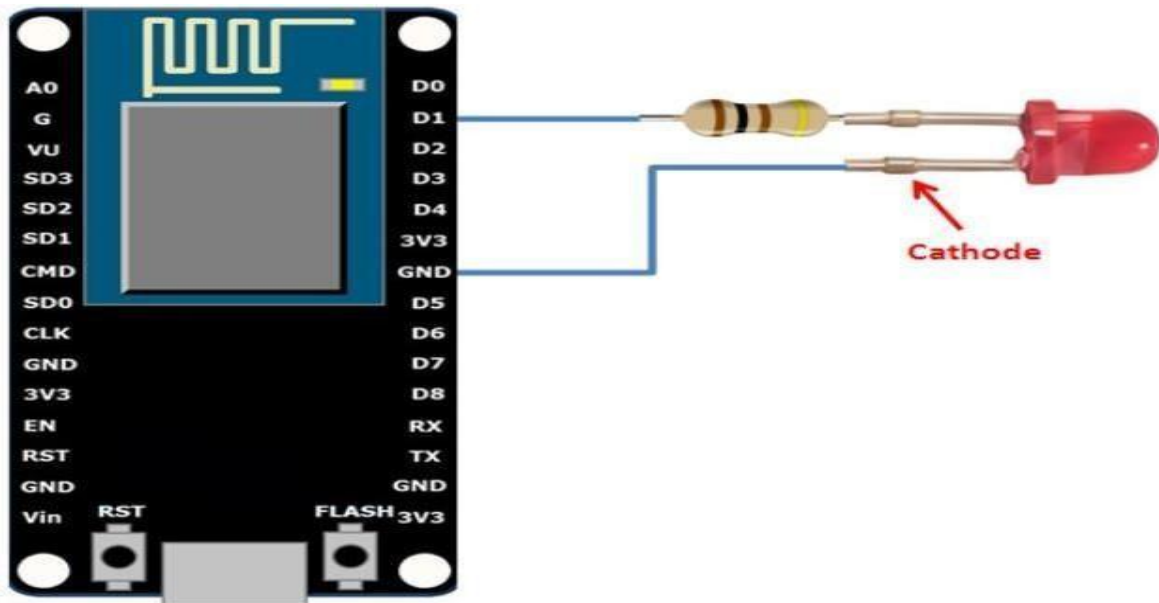


Fig.4.9. ESP board

Connection diagram for the blinking test

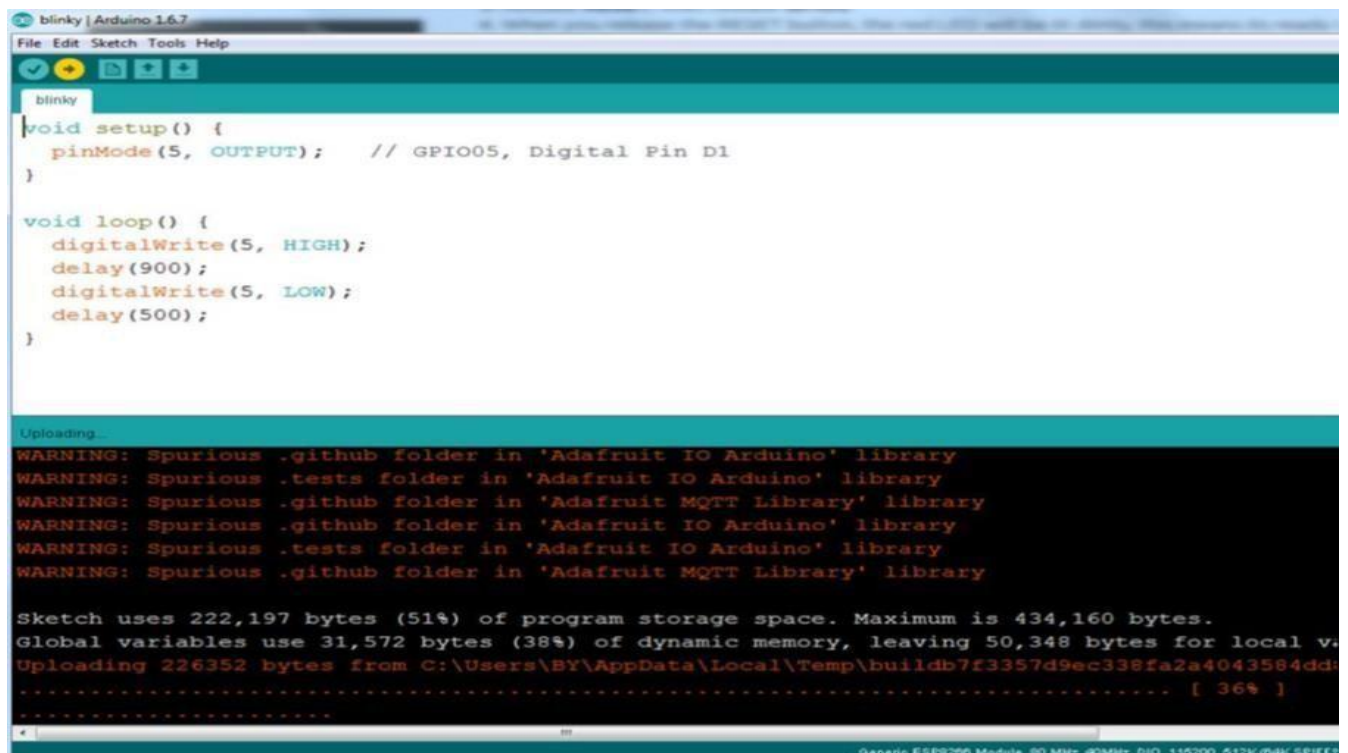


Fig.4.10. ESP8266 using Windows

Flashing NodeMCU Firmware on the ESP8266 using Windows

NodeMCU is a firmware that allows you to program the ESP8266 modules with LUA script. And you'll find it very similar to the way you program your Arduino. With just a few lines of code you can establish a WiFi connection, control the ESP8266 GPIOs, turning your ESP8266 into a web server and a lot more. In this tutorial we are going to use another ESP8266 module with pin header adapter board which is breadboard friendly.



Fig. 4.11. ESP8266 Module Breadboard Friendly with Header Connector



Fig.4.12. Pin Assignment

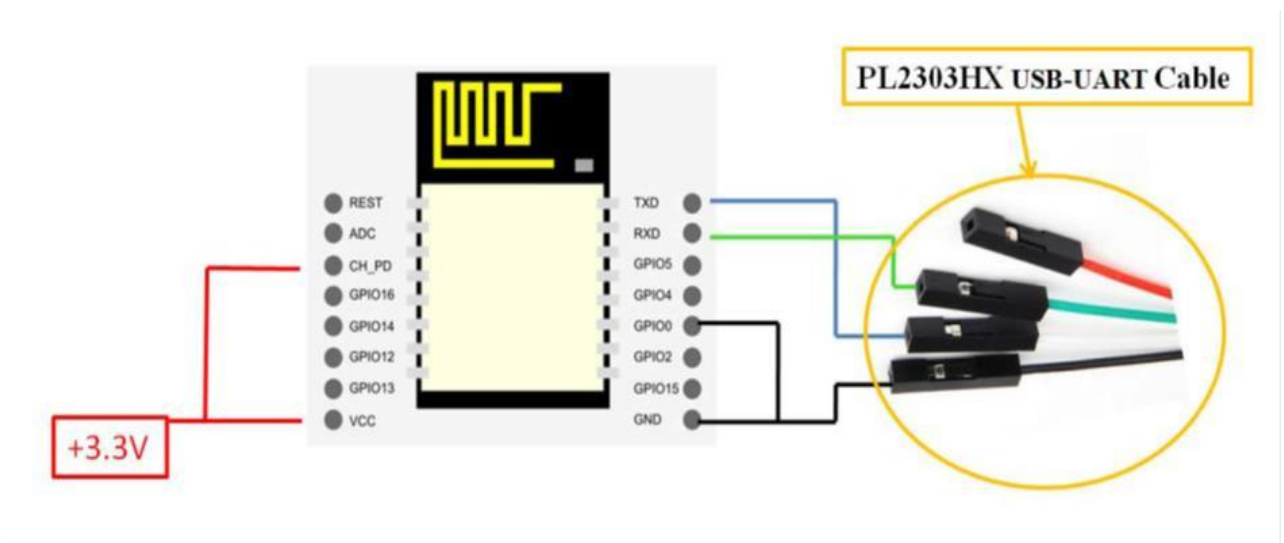


Fig.4.13. Internal

Cables ARDUINO UNO AND ITS

PROGRAMMING

Arduino is a tool for making computers that can sense and control more of the physical world than your desktop computer.



Fig.4.14. Arduino

It's an open-source physical computing platform based on a simple microcontroller board, and a development environment for writing software for the board. Arduino can be used to develop interactive objects, taking inputs from a variety of switches or sensors, and controlling a variety of lights, motors, and other physical outputs.

Arduino projects can be stand-alone, or they can be communicating with software running on your computer. The boards can be assembled by hand or purchased preassembled; the open-source IDE can be downloaded for free. The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment.

Overview

The Arduino microcontroller is an easy to use yet powerful single board computer that has gained considerable traction in the hobby and professional market. The Arduino is open-source, which means hardware is reasonably priced and development software is free. This guide is for students in ME 2011, or students anywhere who are confronting the Arduino for the first time. For advanced Arduino users, prowl the web; there are lots of resources. This guide covers the Arduino Uno board (Spark fun DEV-09950, \$29.95), a good choice for students and educators. With the Arduino board, you can write programs and create interface circuits to read switches and other sensors, and to control motors and lights with very little effort. Many of the pictures and drawings in this guide were taken from the documentation on the

This is what the Arduino board looks like.



Fig.4.15. Arduino board

The Duemilanove board features an Atmel ATmega328 microcontroller operating at 5 V with 2 Kb of RAM, 32 Kb of flash memory for storing programs and 1 Kb of EEPROM for storing parameters. The clock speed is 16 MHz, which translates to about executing about 300,000 lines of C source code per second. There is a USB connector for talking to the host computer and a DC power jack for connecting an external 6-20 V power source, for example a 9 V battery, when running a program while not connected to the host header connectors. The Arduino programming language is a simplified version of C/C++. If you know C, programming the Arduino will be familiar. If you do not know C, no need to worry as only a few commands are needed to perform useful functions. An important feature of the Arduino is that you can create a control program on the host PC, download it to the Arduino and it will run automatically. Remove the USB cable connection to the PC, and the program will still run from the top each time you push the reset button. Remove the battery and put the Arduino board in a closet for six months. When you reconnect the battery, the last program you stored will run. This means that you connect the board to the host PC to develop and debug your program, but once that is done, you no longer need the PC to run the program. 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.

- Microcontroller ATmega328
- 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 MH

4.3.2Pin Configuration

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 this pin.
- **5V.** this pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50mA. **GND.** Ground pins.
- **IOREF.** This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs for working with the 5V or 3.3V.

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.

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using `pinMode()`, `digital Write()`, and `digital Read()` 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 Serial chip.

- **External Interrupts:** 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the `attachInterrupt()` function for details.

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

- **SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.

- **LED:** 13. There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

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

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

Communication

Microcontrollers depend on a host computer for developing and compiling programs. The software used on the host computer is known as an integrated development environment, or IDE. For the Arduino, the development environment is based on the open source Processing platform (www.processing.org) which is described by its creators as a “programming language and environment for people who want to program images, animation, and interactions. If you are unfamiliar with this language, don’t worry; it’s not hard to learn, and the Arduino IDE provides some feedback when you make mistakes in your programs. 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 documentation for details. For SPI communication, use the SPI library. As you go through the list of programming statements available in the Arduino IDE (choose Help->Reference), you might think there isn’t much power for doing things like running servos, operating stepper motors, reading potentiometers, or displaying text on an LCD. Like most any language based on C. A good example of a library you’ll use with the Robot and likely many other robot projects is Servo. This library allows you to connect one or more hobby R/C servos to the Arduino’s digital I/O pins. The Servo library comes with the standard Arduino installation package Library->Servo.

This adds the line

```
#include <Servo.h>
```

Which tells the Arduino IDE that you wish to include the Servo library in your sketch. With the functionality of the library now available to you, you can use its various functions to control one or more servos. For example, you can use the write function to rotate a servo to a specific position, from 0 to 180 degrees. The following code

```
myServo.write(90);
```

Moves a servo to its midpoint, or 90 degree position. Structurally, Arduino sketches are very straightforward and are pretty easy to read and understand. The Arduino program contains two main parts: setup () and loop (). These are programming functions that do what their names suggest: setup () sets up the Arduino hardware.

ARDUINO SOFTWARE:

- A. Open Arduino IDE, which the program should be entered directly from this Arduino and it will be used to check and run the program.

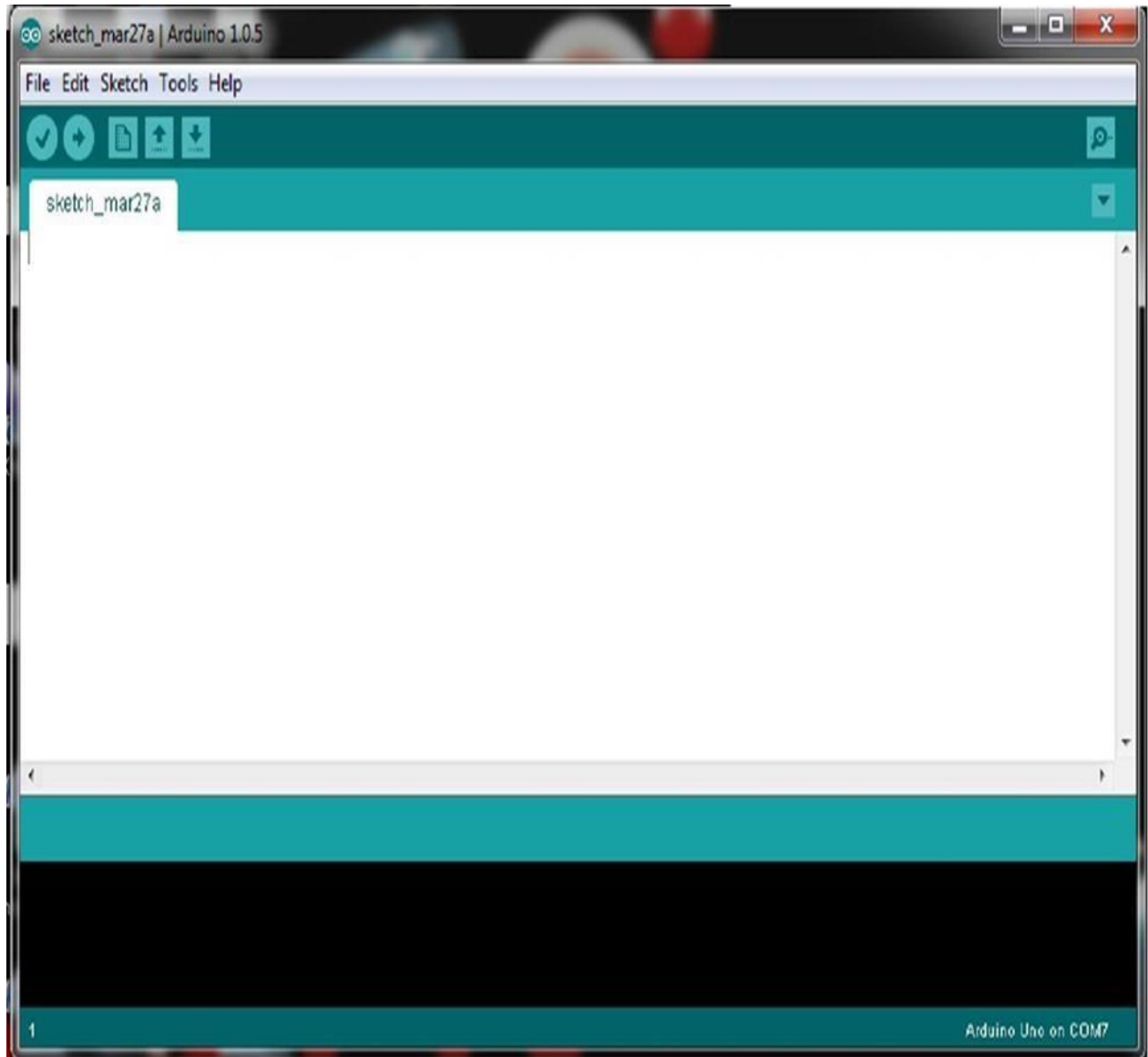


Fig.4.16. Arduino IDE

B. Select the COM Port from tool, where we can select the options and that we can change the modes and the serial points. This will be on the top panel that we can select and make the changes.

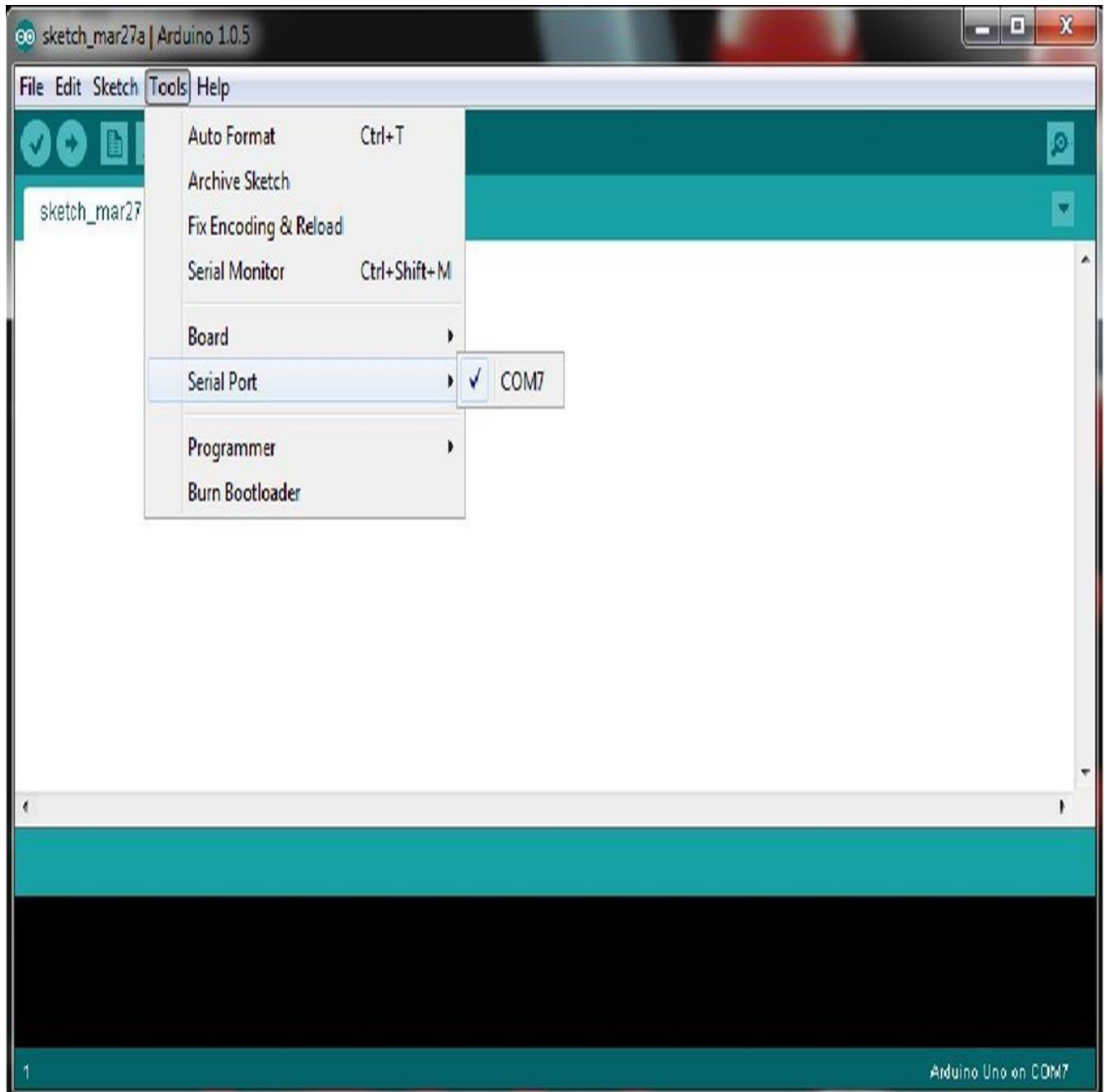


Fig.4.17. COM Port

C. Select the required Arduino board and there will be many options from the Arduino board it will be displayed from inside.

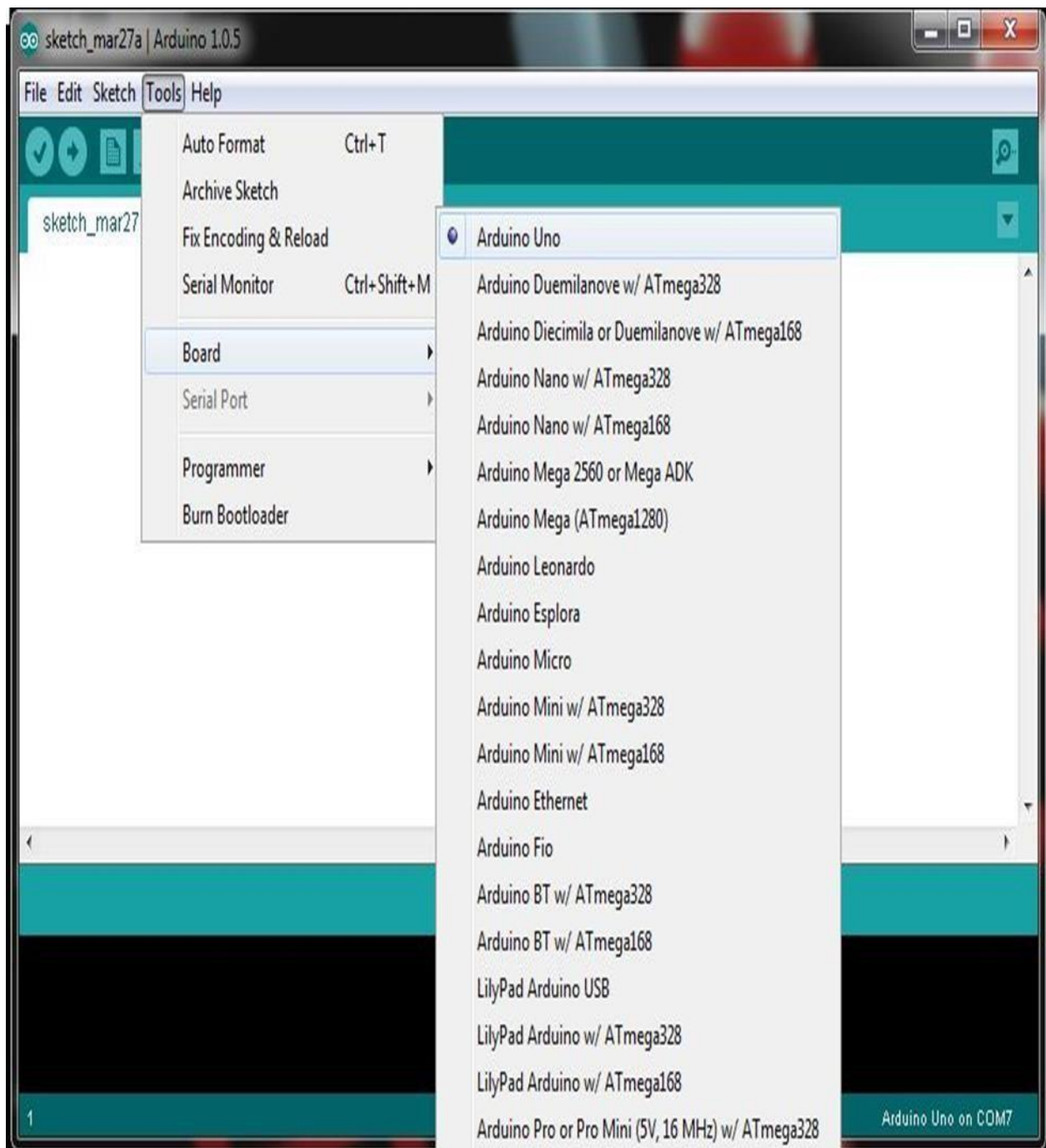
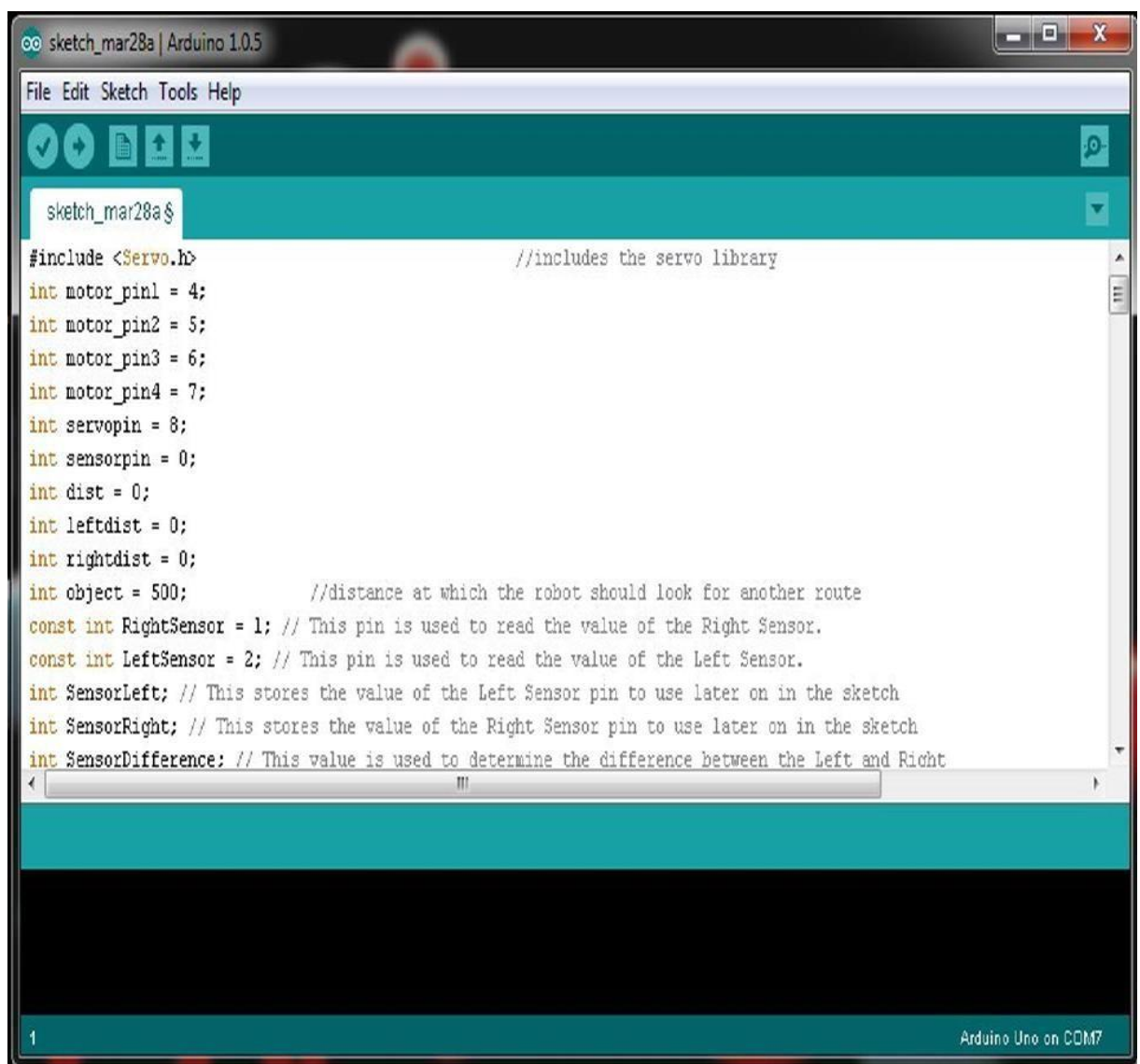


Fig.4.18. Tool selection

D. Sketch in Arduino IDE, here we can add the code directly and we can monitor the every point from the Arduino itself.



```
sketch_mar28a $
#include <Servo.h> //includes the servo library
int motor_pin1 = 4;
int motor_pin2 = 5;
int motor_pin3 = 6;
int motor_pin4 = 7;
int servopin = 8;
int sensorpin = 0;
int dist = 0;
int leftdist = 0;
int rightdist = 0;
int object = 500; //distance at which the robot should look for another route
const int RightSensor = 1; // This pin is used to read the value of the Right Sensor.
const int LeftSensor = 2; // This pin is used to read the value of the Left Sensor.
int SensorLeft; // This stores the value of the Left Sensor pin to use later on in the sketch
int SensorRight; // This stores the value of the Right Sensor pin to use later on in the sketch
int SensorDifference; // This value is used to determine the difference between the Left and Right
```

1 Arduino Uno on COM7

Fig.4.19. Sketch of Arduino

E. Compile and upload the Sketch to Arduino board here and we can easily handle the program that too with the errors can be identified.

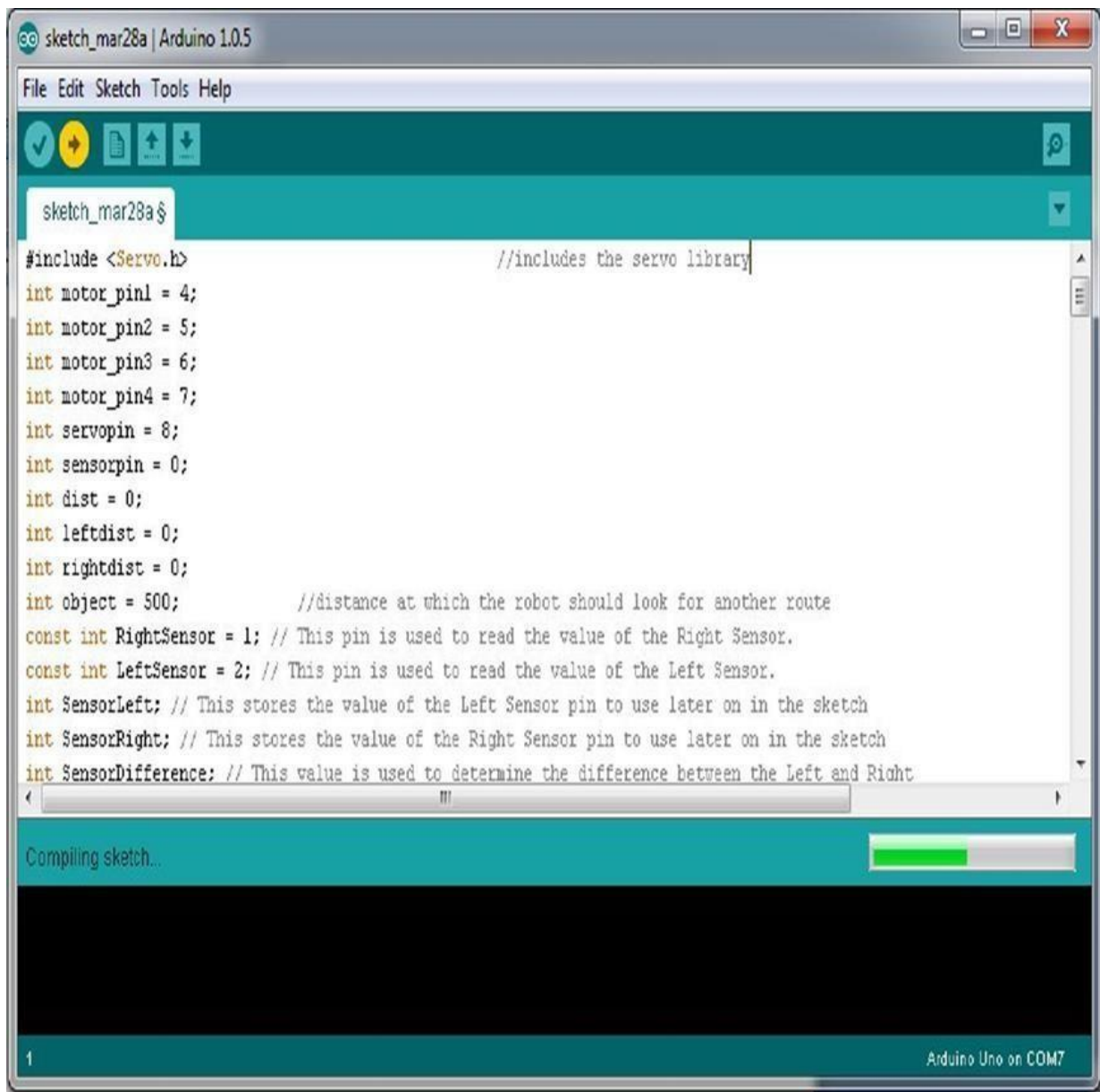


Fig.4.20. uploading the Sketch to Arduino board

Humidity sensor

Measurement and control of temperature and relative humidity finds applications in numerous areas. These days devices are available which have both temperature and humidity sensors with signal conditioning, ADC, calibration and communication interface all built inside them. The use of such smart sensors greatly simplify the design and reduces the overall cost. We discussed in past about Humidity and temperature measurements with Sensirion's SHT1x/SHT7x sensors. These sensors are capable of measuring both temperature and relative humidity and provide fully calibrated digital outputs. While SHT1x/SHT7x are very accurate sensors, they are still expensive for hobbyists use. This articles discusses the DHT11 sensor which also provides calibrated digital outputs for temperature and humidity but is relatively lot cheaper than the Sensirion sensors. The DHT11 sensor uses a proprietary 1-wire protocol which we will be exploring here and implementing with the PIC16F628A microcontroller that will receive the temperature and humidity values from the sensor and display them on a 16×2 character LCD.

Temperature sensor

The DHT11 sensor comes in a single row 4-pin package and operates from 3.5 to 5.5V power supply. It can measure temperature from 0-50 °C with an accuracy of $\pm 2^{\circ}\text{C}$ and relative humidity ranging from 20-95% with an accuracy of $\pm 5\%$. The sensor provides fully calibrated digital outputs for the two measurements. It has got its own proprietary 1-wire protocol, and therefore, the communication between the sensor and a microcontroller is not possible through a direct interface with any of its peripherals. The protocol must be implemented in the firmware of the MCU with precise timing required by the sensor. The following timing diagrams describe the data transfer protocol between a MCU and the DHT11 sensor. The MCU initiates data transmission by issuing a "Start" signal. The MCU pin must be configured as output for this purpose. The MCU first pulls the data line low for at least 18 ms and then pulls it high for next 20-40 μs before it releases it.

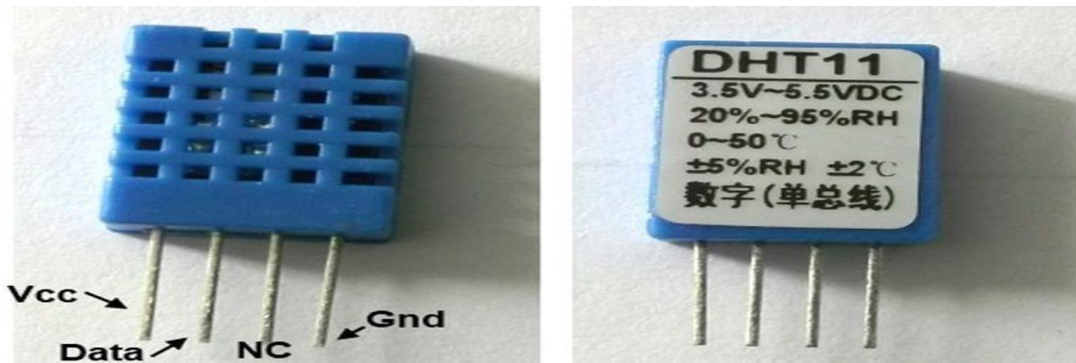


Fig.4.21. Temperature sensor

DHT11 sensor comes in a single row 4-pin package. Next, the sensor responds to the MCU “Start” signal by pulling the line low for 80 μ s followed by a logic high signal that also lasts for 80 μ s. Remember that the MCU pin must be configured to input after finishing the “Start” signal. Once detecting the response signal from the sensor, the MCU should be ready to receive data from the sensor. The sensor then sends 40 bits (5 bytes) of data continuously in the data line. Note that while transmitting bytes, the sensor sends the most significant bit first.

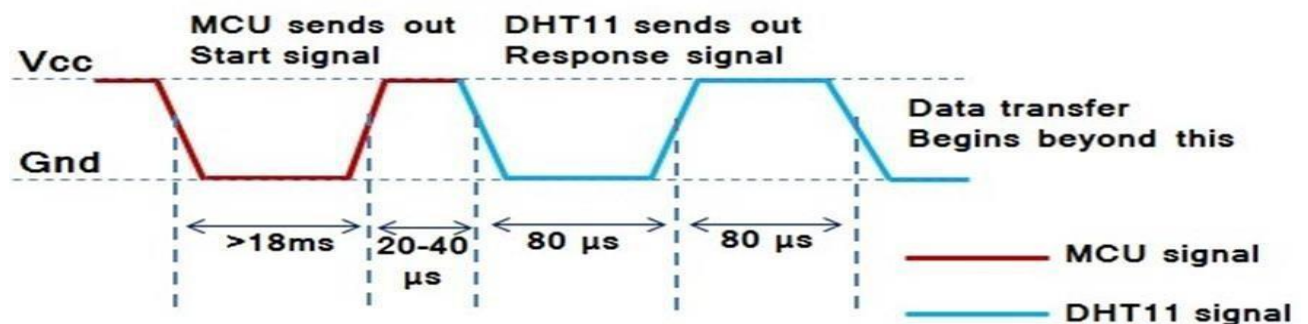


Fig.4.22. Digital signal

The 40-bit data from the sensor has the following structure.

Data (40-bit) = Integer Byte of RH + Decimal Byte of RH + Integer Byte of Temp. + Decimal Byte of Temp. + Checksum Byte

For DHT11 sensor, the decimal bytes of temperature and humidity measurements are always zero. Therefore, the first and third bytes of received data actually give the numeric values of the measured relative humidity (%) and temperature (°C). If all the five bytes are transferred successfully then the checksum byte must be equal to the last 8 bits of the sum of the first four bytes, i.e. Checksum = Last 8 bits of (Integer Byte of RH + Decimal Byte of RH + Integer Byte of Temp. + Decimal Byte of Temp.).

Now lets talk about the most important thing, which is signalling for transmitting "0" and "1". In order to send a bit of data, the sensor first pulls the line low for 50 ?s. Then it raises the line to high for 26-28 ?s if it has to send "0", or for 70 ?s if the bit to be transmitted is "1". So it is the width of the positive pulse that carries information about 1 and 0.

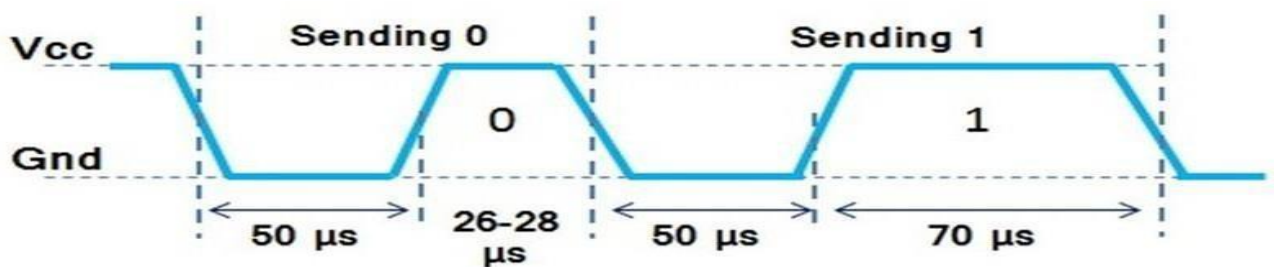


Fig.4.23. Transmitting signal

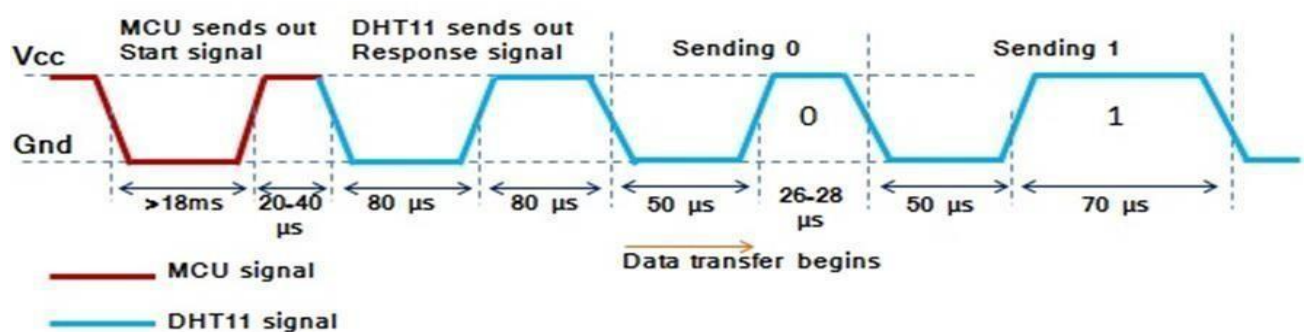


Fig.4.24. Timing difference for transmitting "1s" and "0s"

At the end of the last transmitted bit, the sensor pulls the data line low for 50 ?s and then releases it. The DHT11 sensor requires an external pull-up resistor to be

connected between its Vcc and the data line so that under idle condition, the data line is always pulled high. After finishing the data transmission and releasing the data line, the DHT11 sensor goes to the low-power consumption mode until a new “Start” signal arrives from the MCU.

Pressure sensor

A pressure sensor measures pressure, typically of gases or liquids. Pressure is an expression of the force required to stop a fluid from expanding, and is usually stated in terms of force per unit area. A pressure sensor usually acts as a transducer; it generates a signal as a function of the pressure imposed. For the purposes of this article, such a signal is electrical. Pressure sensors are used for control and monitoring in thousands of everyday applications.

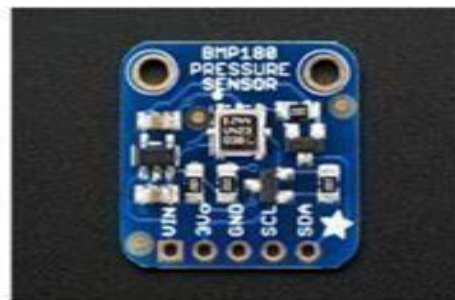


Fig.4.25. Pressure sensor

Pressure sensors can also be used to indirectly measure other variables such as fluid/gas flow, speed, water level, and altitude. Pressure sensors can alternatively be called pressure transducers, pressure transmitters, pressure senders, pressure indicators, piezometers and manometers, among other names. Pressure sensors can vary drastically in technology, design, performance, application suitability and cost. A conservative estimate would be that there may be over 50 technologies and at least 300 companies making pressure sensors worldwide.

Respiratory sensor

Air and gas flows as well as line and therapy pressures have to be monitored and controlled in numerous medical instruments such as respiratory devices, aesthetic devices, sleep diagnosis devices, sleep apnea therapy devices (CPAP), spirometers and oxygen concentrators. High-quality sensors in respiratory devices measure minute flow rates around the zero point of the respiratory flow and also detect flow rates of several hundred l/min. Applications in aesthetic devices call for sensors with high resistance to anaesthetics.

Figure 4.26



Fig.4.26. Respiratory sensor

In controlled CPAP devices, pressure sensors continuously monitor the therapy pressure, thereby improving the comfort and quality of the treatment. Spirometers utilize special pneumotacho graphs that measure the respiratory flow using efficient differential pressure sensors. First Sensor develops and manufactures highly reliable sensors and customized sensor systems as a strategic partner to medical product manufacturers in the area of breathing and respiration. Our complete advisory, development, production and service processes comply with the high requirements for medical products according to EN ISO 13485. The respiration sensor is fixed to a long hook and loop strap that is placed around the chest or abdomen.

MOTIVATION

IoT-based system is in charge of providing knowledge from an environment to a non-expert user. IoT-based system can be used in different environments, so it needs to be able to address many heterogeneous devices. Thus, a major concern within developing an IoT-based system is how to handle the interaction with the heterogeneous devices for non-expert users. This concern can be addressed by a middleware layer between devices and non-expert users. The middleware we found in the literature can provide satisfaction by facilitating the interaction with devices, but they do not support low-level device configuration [4].

IoT

Internet of Things (IoT) has increasingly gained attention in industry to interact with different types of devices. IoT can have influence on industry and society by integrating physical devices into information networks [8]. IoT impacts can be on different perspectives, namely for private and business users. From the perspective of a private user, IoT has effect on both working and personal fields, such as smart homes and offices, e-health and assisted living. From the aspect of a business user the impacts would be in fields such as automation and industrial manufacturing, logistics, business process management, intelligent transportation of people and goods [9] IoT integrates physical things into information networks. IoT covers the overall infrastructure, including software, hardware and services, which is used to support these information networks.

1. Edge Technology layer

This is a hardware layer that consists of embedded systems, RFID tags, sensor networks and all of the other sensors in different forms. This hardware layer can perform several functions, such as collecting information from a system or an environment, processing information and supporting communication.

2. Access Gateway layer

This layer is concerned with data handling, and is responsible for publishing and subscribing the services that are provided by the Things, message routing, and hovelling the communication between platforms.

a. Middleware layer

This layer has some critical functionalities, such as aggregating and filtering the received data from the hardware devices, performing information discovery and providing access control to the devices for applications.

b. Application layer

This layer is responsible for delivering various application services. These services are provided through the middleware layer to different applications and users in IoT-based systems. The application services can be used in different industries such as, logistics, retail, healthcare, etc.

HEARTBEAT SENSOR

A person's heartbeat is the sound of the valves in his/her's heart contracting or expanding as they force blood from one region to another. The number of times the heart beats per minute (BPM), is the heart beat rate and the beat of the heart that can be felt in any artery that lies close to the skin is the pulse.

Two Ways to Measure a Heartbeat

Manual Way

Heart beat can be checked manually by checking one's pulses at two locations- wrist (the radial pulse) and the neck (carotid pulse). The procedure is to place the two fingers (index and middle finger) on the wrist (or neck below the windpipe) and count the number of pulses for 30 seconds and then multiplying that number by 2 to get the heart beat rate. However pressure should be applied minimum and also fingers should be moved up and down till the pulse is felt.

Using a sensor: Heart Beat can be measured based on optical power variation as light is scattered or absorbed during its path through the blood as the heart beat changes.

Principle of Heartbeat Sensor

The heartbeat sensor is based on the principle of photo plethysmography. It measures the change in volume of blood through any organ of the body which causes a change in the light intensity through that organ (a vascular region). In case of applications where heart pulse rate is to be monitored, the timing of the pulses is more important. The flow of blood volume is decided by the rate of heart pulses and since light is absorbed by blood, the signal pulses are equivalent to the heart beat pulses.

There are two types of photoplethysmography:

Transmission: Light emitted from the light emitting device is transmitted through any vascular region of the body like earlobe and received by the detector.

Reflection: Light emitted from the light emitting device is reflected by the regions.

Working of a Heartbeat Sensor

The basic heartbeat sensor consists of a light emitting diode and a detector like a light detecting resistor or a photodiode. The heart beat pulses causes a variation in the flow of blood to different regions of the body. When a tissue is illuminated with the light source, i.e. light emitted by the led, it either reflects (a finger tissue) or transmits the light (earlobe). Some of the light is absorbed by the blood and the transmitted or the reflected light is received by the light detector. The amount of light absorbed depends on the blood volume in that tissue. The detector output is in form of electrical signal and is proportional to the heart beat rate. This signal is actually a DC signal relating to the tissues and the blood volume and the AC component synchronous with the heart beat and caused by pulsatile changes in arterial blood volume is superimposed on the DC signal. Thus the major requirement is to isolate that AC component as it is of prime importance.

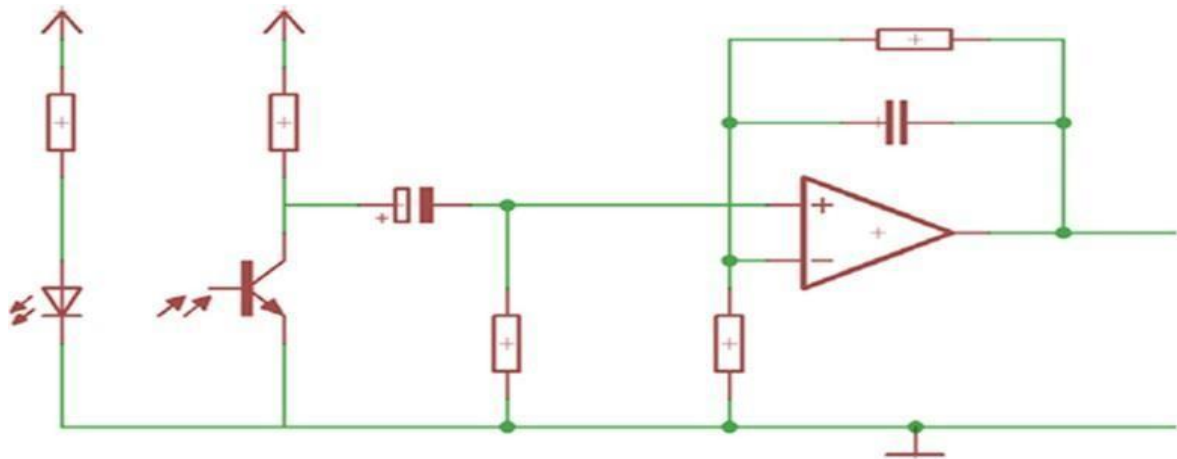


Fig.4.27. AC signal

To achieve the task of getting the AC signal, the output from the detector is first filtered using a 2 stage HP-LP circuit and is then converted to digital pulses using a comparator circuit or using simple ADC. The digital pulses are given to a microcontroller for calculating the heart beat rate,

$$\text{BPM(Beats per minute)} = 60 \cdot f$$

Where f is the pulse frequency

Practical Heartbeat sensor

Practical heartbeat Sensor examples are Heart Rate Sensor (Product No PC-31470).



Fig.4.28.Heartbeat sensor

It consists of an infrared led and an ldr embedded onto a clip like structure. The clip is attached to the organ (earlobe or the finger) with the detector part on the flesh. It is embedded on a wearable device which can be worn on the wrist and the output can be sent wirelessly (through Bluetooth) to the computer for processing.

Application

Developing your own Heartbeat Sensor System.

A basic Heartbeat Sensor system can also be built using basic components like a ldr, comparator ICLM358 and a Microcontroller as given. As described above regarding the principle of heart beat sensor, when the finger tissue or the earlobe tissue is illuminated using a light source, the light is transmitted after getting modulated i.e. a part getting absorbed by the blood and the rest being transmitted. It works on the principle that when light falls on the resistor, its resistance changes. As the light intensity increases, the resistance decreases. Thus the voltage drop across the resistor decreases. Here a comparator is used which compares the output voltage from the LDR to that of the threshold voltage. As this reduced light intensity falls on the LDR, the resistance increases and as a result the voltage drop increases. When the voltage drop across the LDR or the non-inverting input exceeds that of the inverting input, a logic high signal is developed at the output of the comparator and in case voltage drop being lesser a logic low output is developed. These pulses can be fed to the Microcontroller which accordingly processes the information to get the heart beat rate and this is displayed on the Display interfaced to the Microcontroller.

CHAPTER 5

RESULTS AND DISCUSSION

DATASET COLLECTION

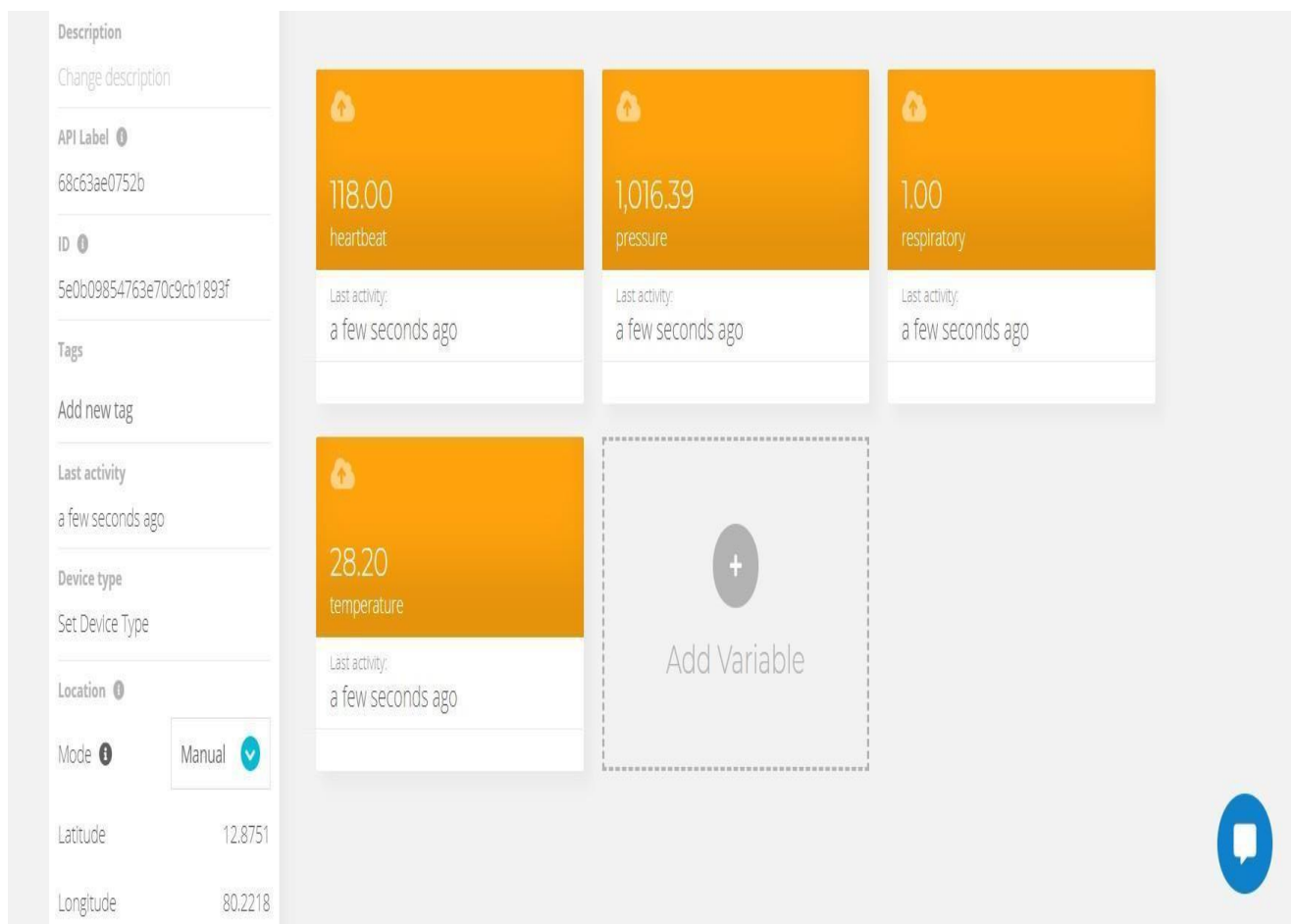


Fig.4.29. Database

In this database, we can monitor the present health condition of the pregnant lady so that we can easily reach for the situation of that lady. This will help to identify the wrong situation or any right situation of that pregnant lady.

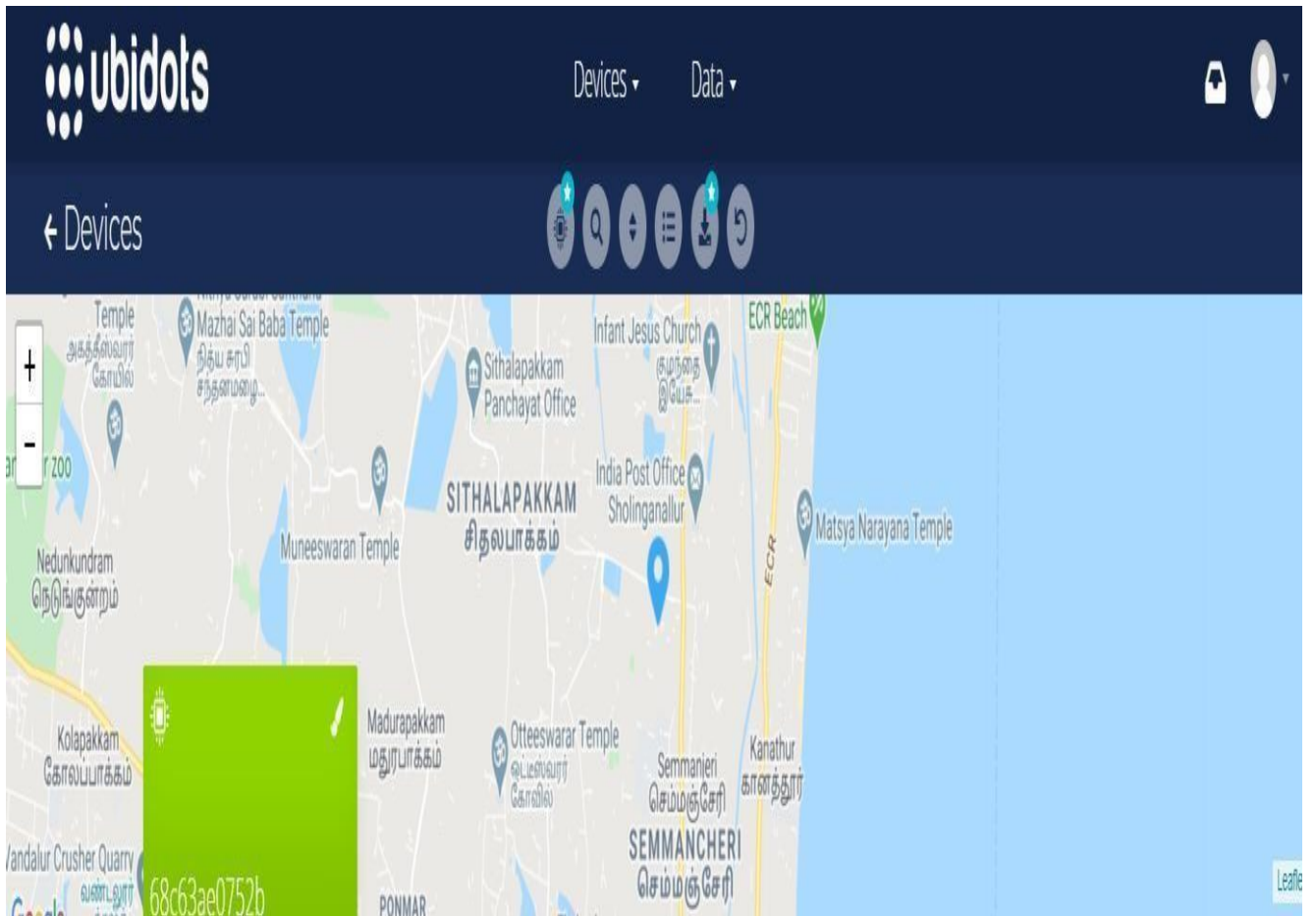


Fig.4.30. Google Location

The Google location shows the exact and correct position of the pregnant lady who wore the monitoring device. There are locations and colors which we can change as per the project or as per the database which we use for the verified project.

GRAPHS



Fig.4.31. Graphical storage of data

A graphical storage data shows the fluctuating ranges and the situations of the receiving data continuously from the device which was worn by the pregnant lady. The data varies as per the received data from the lady who was pregnant.

Mail Alerts



Fig.4.32. Heartbeat Alert

Here there is the received mail for the data that monitored from the device and when the range of the heartbeat raises then randomly such type of mail alert will get received to the registered mail.



Fig.4.33. Temperature Alert

Here there is the received mail for the data that monitored from the device and when the range of the temperature raises then randomly such type of mail alert will get received to the registered mail.



Fig.4.34. Respiratory alert

Here there is the received mail for the data that monitored from the device and when the range of the Respiratory was 1 in a normal way when it changes in inhale and exhale process it turns as 0 and randomly mail alert will get received to the registered mail.

CHAPTER 6

SUMMARY

&CONCLUSION

SUMMARY

Currently, the available system is used as a compact and wearable device. This system collects and stores the data which was continuously monitored from the database. So these are the concepts that we can implement over this IoT platform that we can easily find the temperature, pressure, heartbeat and respiratory measures which can be stored to cloud with the particular time, date and location.

CONCLUSION

Most revisions of maternal mortality are hospital based concepts. However, in evolving and under developed countries, most of the maternal deaths take place at home. In order to reduce these problems, a compact assistive device is designed and the vital parameters such as the temperature, pressure, heart rate, pressure of the pregnant lady is measured by using different sensors. The device is lightweight and highly sensitive even for small movements, thus preferred as a home monitoring device. Regular monitoring of the vital parameters of women in the rural areas reduces the infant mortality. The measured parameters are transferred through the IoT, It provides quality and timely health support for women. The results are viewed in the mobile phone and also we can check from the database in the computer through the IoT.

REFERENCES

- [1] Aleksandra C. Zoric, Sinisa S. Ilic, "PC Based Electrocardiography & Data Acquisition", TELSIKS, IEEE, pp 619-622, September 28- 30 2005.
- [2] Tia Gao, Dan Greenspan, Matt Welsh, Radford R. Juang, and Alex Alm, "Real Time Patient Monitoring System Using Lab view", International Journal of Scientific & Engineering Research, April-2012.
- [3] Sherin Sebastian, Neethu Rachel Jacob, "Remote Patient Monitoring System Using Android Technology", IJDPS, September 2012. Prema Sundaram, International Journal of Computer Science and Mobile Computing Vol.2 Issue. 5, May- 2013, pg. 191-201 © 2013, IJCSMC All Rights Reserved 201.
- [4] C. Wen, M. Yen, K. Chang and R. Lee , "Realtime ECG telemonitoring system design with mobile phone platform", Measurement, Volume 41, Issue 4, May 2008, Pages 463-470.
- [5] Wilkoff BL, Auricchio A, Brugada J, Cowie M, Ellenbogen KA, Gillis AM et al. "HRS/EHRA Expert Consensus on the Monitoring of Cardiovascular Implantable Electronic Devices (CIEDs): Description of Techniques, Indications, Personnel, Frequency Ethical Considerations", Euro pace 2008;10:707–25.
- [6] Varma N, Epstein A, Schweikert R, Love C, Shah JA, Irimpen " Evaluation of efficacy and safety of remote monitoring for ICD follow-up" the TRUST trial, Circulation 2008, Vol. 118, No. 22, 2316, Abstract 4078.
- [7] Kiely DK, "Resident characteristics associated with wandering in nursing home", Int J Geriatric Psychiatry. 2000; 15(11):1013-1020.
- [8] K. Lorincz et al., "Sensor Networks for Emergency Response: Challenges and Opportunities," IEEE Pervasive Computing, IEEE Press, pp. 16-23, October/December 2004.

- [9] J. Hill et al., "System Architecture Directions for Networked Sensors," in Proc. 9th Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000)", ACM Press, pp.93-104, 2000.
- [10] K. Lorincz and M. Welsh, A Robust, "Decentralized Approach to RF Based Location Tracking, tech. report TR-19-04, Division of Eng. And Applied Sciences " , Harvard Univ., Cambridge, MA, 2004.
- [11] Batz, D., Batz, M., Lobo, N.D.V. and Shah, M. "A computer vision system for monitoring medication intake", Canadian Conf. on Computer and Robot Vision, May, pp. 362-369.2005.
- [12] Zao, J.K., Wang, M.Y., Peihuan, T. and Liu, J.W.S., "Smart Phone Based Medicine In-take Scheduler, Reminder and Monitor", IEEE e-Health Networking Applications and Services (Healthcom), pp 162 – 168., 2010.
- [13] Media Aminian, Hamid Reza Naji, "A Hospital Healthcare Monitoring System Using Wireless Sensor Networks", Department of Computer Science and Research branch Islamic Azad University Kerman Iran and College of Electrical and Computer Engineering Kerman Graduate University of Technology Iran, vol. 4, 2013.
- [14] B.K Bhoomika, K N Muralidhara, "Secured Smart Healthcare Monitoring System Based on Iot", International Journal on Recent and Innovation Trends in Computing and Communication, vol. 3, no. 7, pp. 4958-4961, July 2015.
- [15] L. Wang, G.-Z. Yang, J. Huang, J. Zhang, L. Yu, Z. Nie et al., "A wireless biomedical signal interface system-on-chip for body sensor networks", IEEE Trans. Biomed. Circuits Syst., vol. 4, no. 2, pp. 112-117, Apr. 2010.
- [16] S Anbumani, R2 Kavin, A3 Saranya, T4 Dineshkumar, R5 Premkumar, "An Intelligent Patient Tele-Monitoring System Using Android Technology", International Journal of Research in Engineering and Technology, vol. 4, no. 02, pp. 477-482, Feb 2015.

- [17] LeBellego G, Noury N, Virone G, Mousseau M, Demongeot J (2006) A model for the measurement of patient activity in a hospital suite. *IEEE Transactions on information technology in biomedicine*.
- [18] Lee C, Kim T, Hyun SJ (2016) A data acquisition architecture for healthcare services in mobile sensor networks. In: 2016 International conference on big data and smart computing (BigComp). IEEE, pp 439–442.
- [19] Lin CT, Ko LW, Chang MH, Duann JR, Chen JY, Su TP, Jung TP (2010) Review of wireless and wearable electroencephalogram systems and brain-computer interfaces—a mini-review. *Gerontology* 56(1):112–9.
- [20] Pham M, Mengistu Y, Do HM, Sheng W (2016) Cloud-Based Smart Home Environment (CoSHE) for home healthcare. In: 2016 IEEE international conference on automation science and engineering (CASE). IEEE, pp 483–488.

APPENDIX

A. SOURCE CODE

```
#include "Ubidots.h"

#include

<SFE_BMP180.h>

#include <Wire.h>

#include "DHT.h"

#define DHTPIN D5

const int analogPin =

A0; int respi = D3;

#define DHTTYPE DHT11

DHT dht(DHTPIN,

DHTTYPE);

SFE_BMP180 pressure;

#define ALTITUDE

1655.0

const char* UBIDOTS_TOKEN = "BBFF-d4Sd5j3TipgfvAg3VRh18t7s5Cp4VT";      //
Put here your Ubidots TOKEN

const char* WIFI_SSID = "admin";      // Put here your Wi-Fi SSID

const char* WIFI_PASS = "12345678";      // Put here your Wi-Fi

password Ubidots ubidots(UBIDOTS_TOKEN, UBI_HTTP);

void setup()

{
```

```
Serial.begin(115200);
```

```

ubidots.wifiConnect(WIFI_SSID,
WIFI_PASS); if (pressure.begin())

Serial.println("BMP180 init
success"); else

{

Serial.println("BMP180 init
fail\n\n"); while(1);

}

pinMode(respi, INPUT);

Serial.println(F("DHTxx
test!")); dht.begin();

}

void loop()

{

//_____temp-_____//

delay(2000);

float h = dht.readHumidity();

float t =

dht.readTemperature();

float f =

dht.readTemperature(true); if

(isnan(h) || isnan(t) || isnan(f)) {

```

```
Serial.println(F("Failed to read from DHT sensor!"));
```

```

return;

}

float hif = dht.computeHeatIndex(f, h);

float hic = dht.computeHeatIndex(t, h,
false); Serial.print(F("% Temperature: "));

Serial.print(t);

Serial.println(F("°C "));

ubidots.add("Temperature",

t);

//_____//

//-----respiratory    //

int State = digitalRead(respi);

Serial.print("respiratory::");

Serial.println(State);

ubidots.add("respiratory",

State);

//_____//

//____hb-_____//

int sensorValue =

analogRead(analogPin); delay(1000);

if (sensorValue >= 1000)

{

```

```
int Heartbeat = random(60, 120);
```

```

Serial.print("Heartbeat::");

Serial.println(Heartbeat);

ubidots.add("Heartbeat",

Heartbeat);

}

//_____//

//_____pressure_____//

char status;

double

T,P,p0,a;

status

=pressure.startTemperature(); if

(status != 0)

{

delay(status);

status

=pressure.getTemperature(T); if

(status != 0)

{

status

=pressure.startPressure(3); if

(status != 0)

```

```
{  
  
delay(status);  
  
status =  
  
pressure.getPressure(P,T); if  
  
(status != 0)
```

```

{

Serial.print("absolute pressure:

"); Serial.print(P,2);

Serial.print(" mb, ");

ubidots.add("pressure",

P); }

else Serial.println("error retrieving pressure measurement\n");

}

else Serial.println("error starting pressure measurement\n");

}

else Serial.println("error retrieving temperature measurement\n");

}

else Serial.println("error starting temperature measurement\n");

bool bufferSent = false;

bufferSent = ubidots.send(); // Will send data to a device label that matches the device
Id

if (bufferSent) {

Serial.println("Values sent by the device");

}

delay(1000);

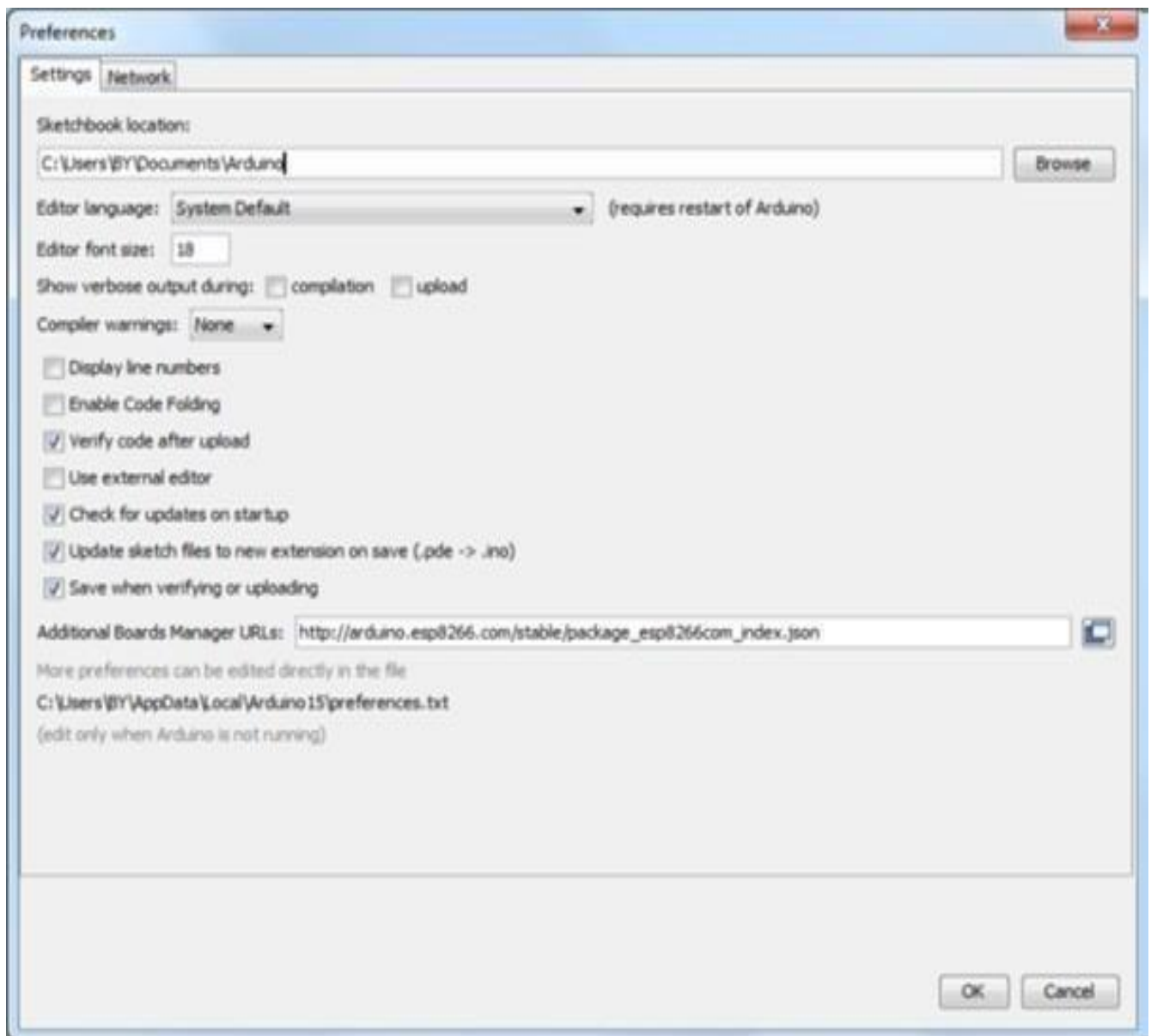
// _____//

}

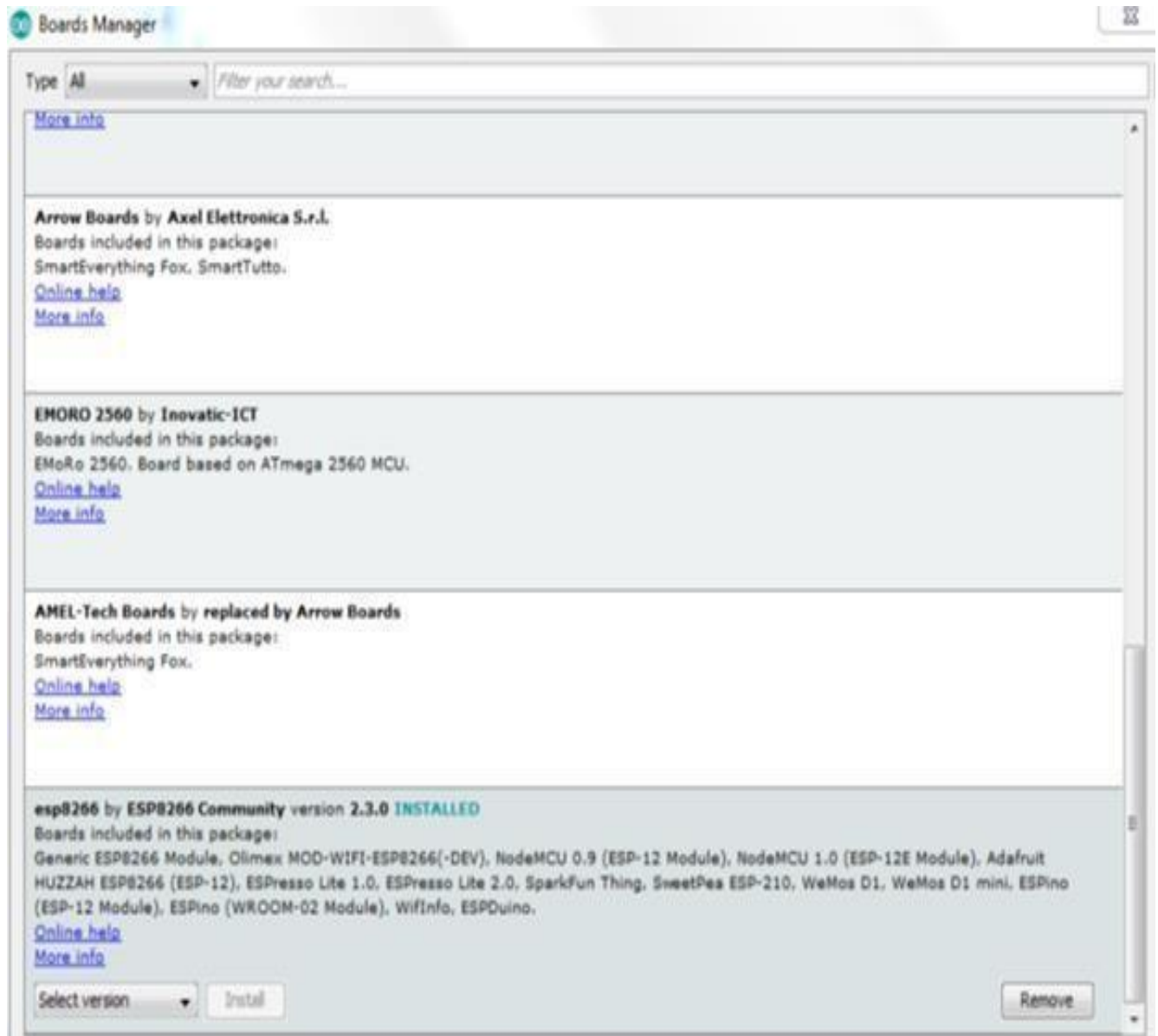
```

B.

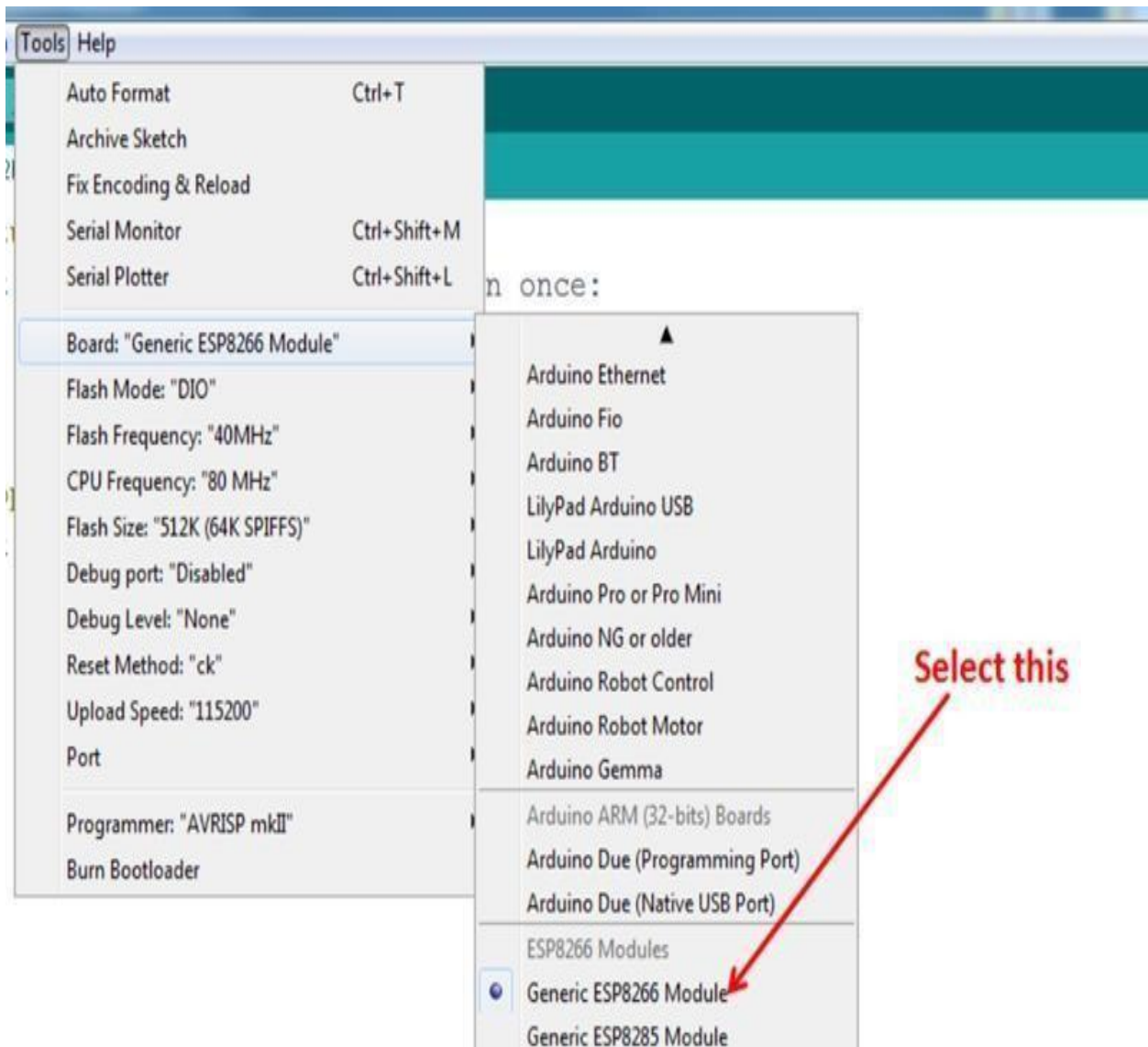
SCREENSHOTS



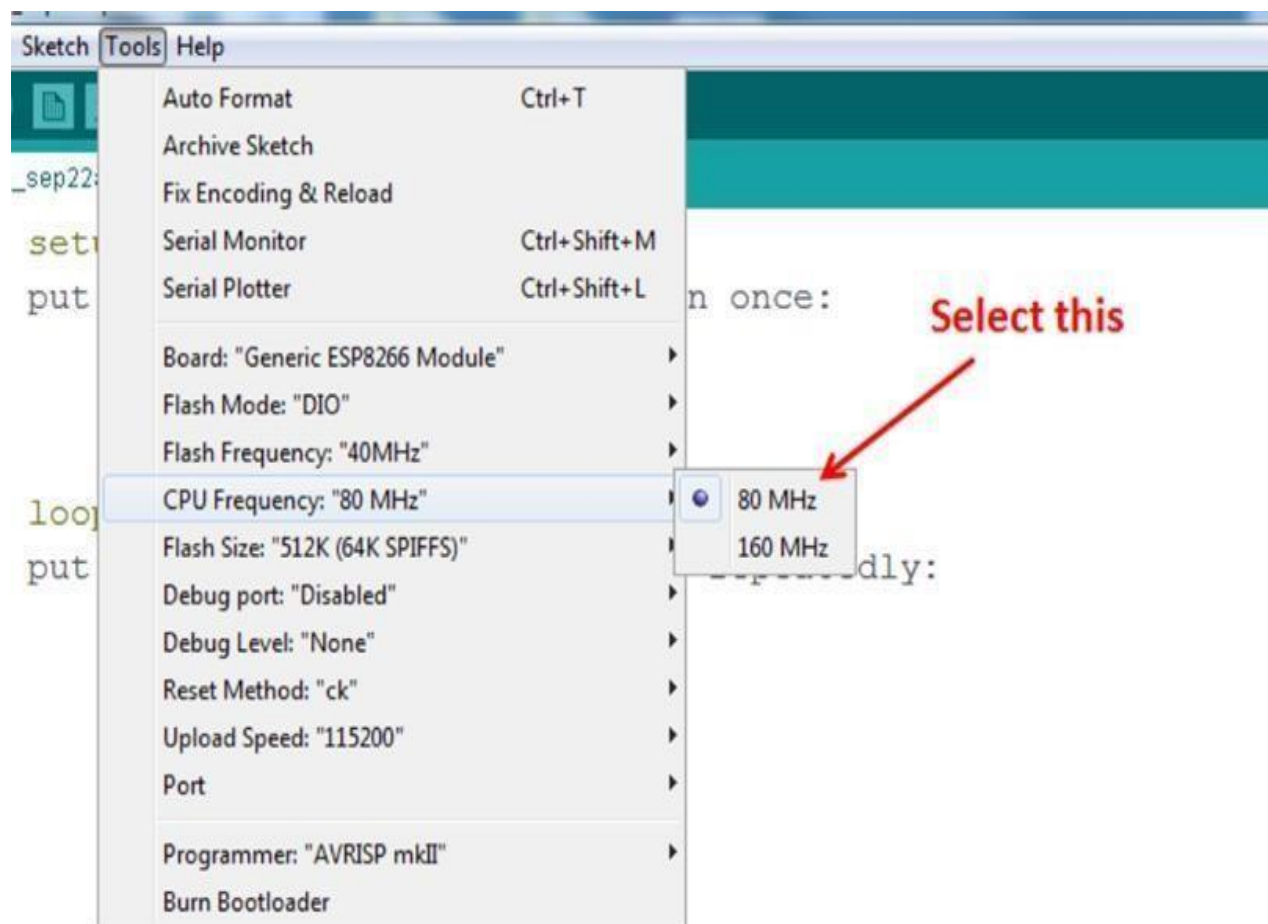
B1. Additional Board Manager URLs field in the Arduino v1.6.4+ preferences.



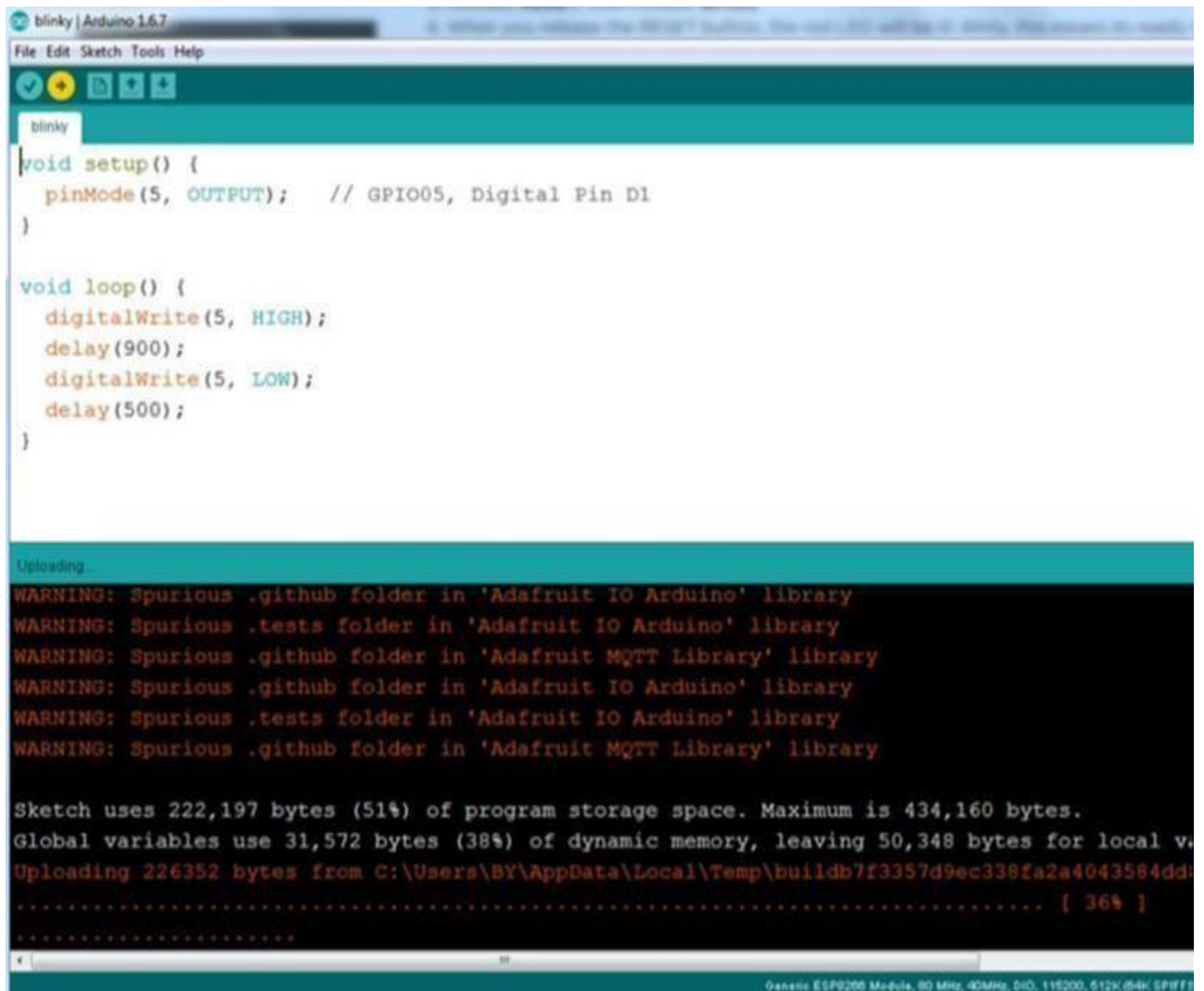
B2. Preferences, ESP8266 Community ' and click "Install" button to install the ESP8266 library package



B3. Select the required Arduino board and there will be many options from the Arduino board it will be displayed from inside.



B4. 80 MHz as the CPU frequency



The screenshot shows the Arduino IDE interface. The top menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu is a toolbar with icons for opening files, saving, and running. The sketch editor displays a 'blink' sketch with the following code:

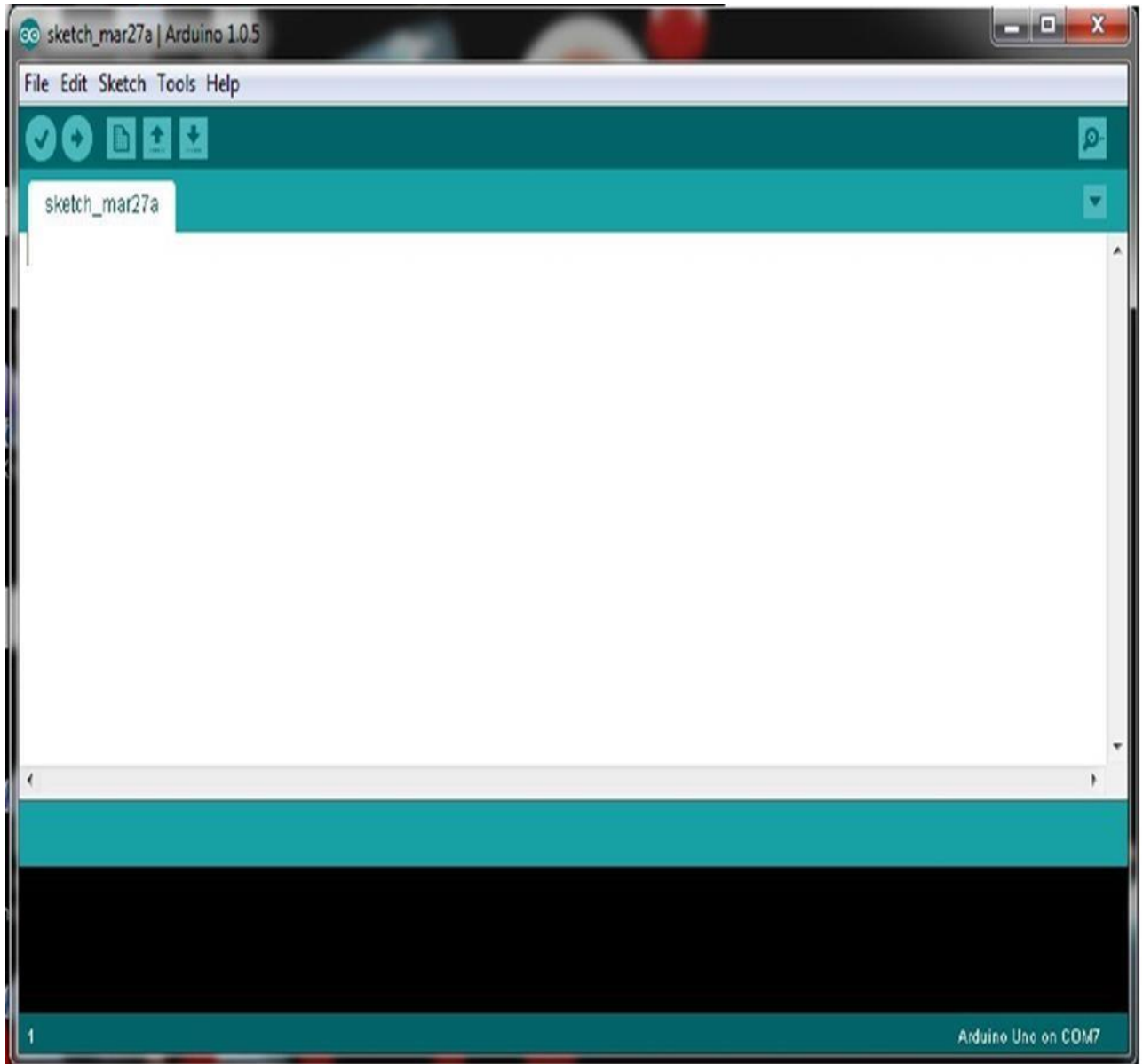
```
void setup() {  
  pinMode(5, OUTPUT); // GPIO05, Digital Pin D1  
}  
  
void loop() {  
  digitalWrite(5, HIGH);  
  delay(900);  
  digitalWrite(5, LOW);  
  delay(500);  
}
```

Below the editor, the 'Serial' monitor is open, showing the upload progress. The status bar at the bottom indicates 'Uploading' and 'Genetic ESP8266 Module, 80 MHz, 40MHz, DIO, 115200, 512K (54K SPIFFS)'.

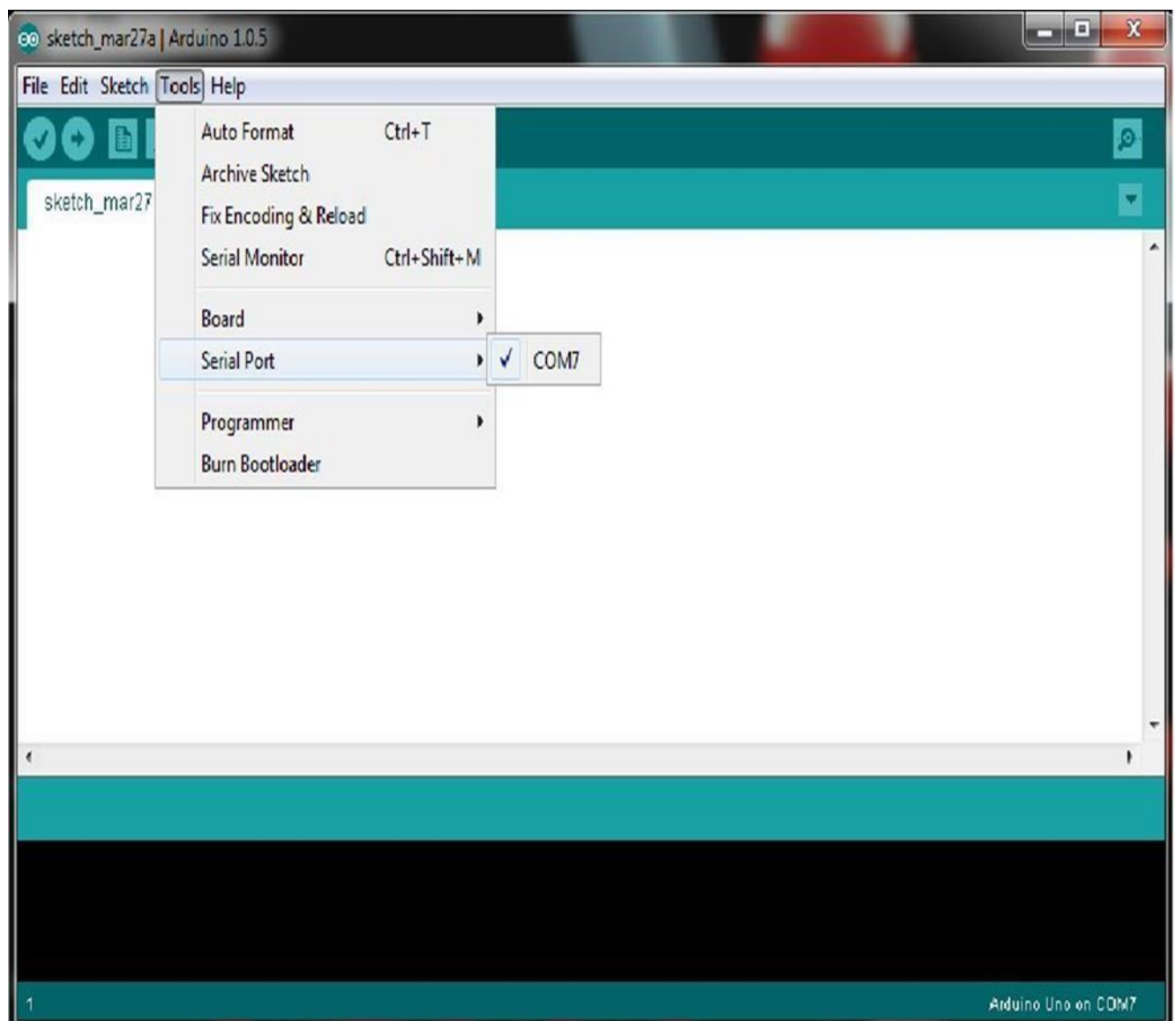
Serial Monitor Output:

```
Uploading  
WARNING: Spurious .github folder in 'Adafruit IO Arduino' library  
WARNING: Spurious .tests folder in 'Adafruit IO Arduino' library  
WARNING: Spurious .github folder in 'Adafruit MQTT Library' library  
WARNING: Spurious .github folder in 'Adafruit IO Arduino' library  
WARNING: Spurious .tests folder in 'Adafruit IO Arduino' library  
WARNING: Spurious .github folder in 'Adafruit MQTT Library' library  
  
Sketch uses 222,197 bytes (51%) of program storage space. Maximum is 434,160 bytes.  
Global variables use 31,572 bytes (38%) of dynamic memory, leaving 50,348 bytes for local variables.  
Uploading 226352 bytes from C:\Users\BY\AppData\Local\Temp\buildb7f3357d9ec338fa2a4043584ddi  
..... [ 36% ]  
.....
```

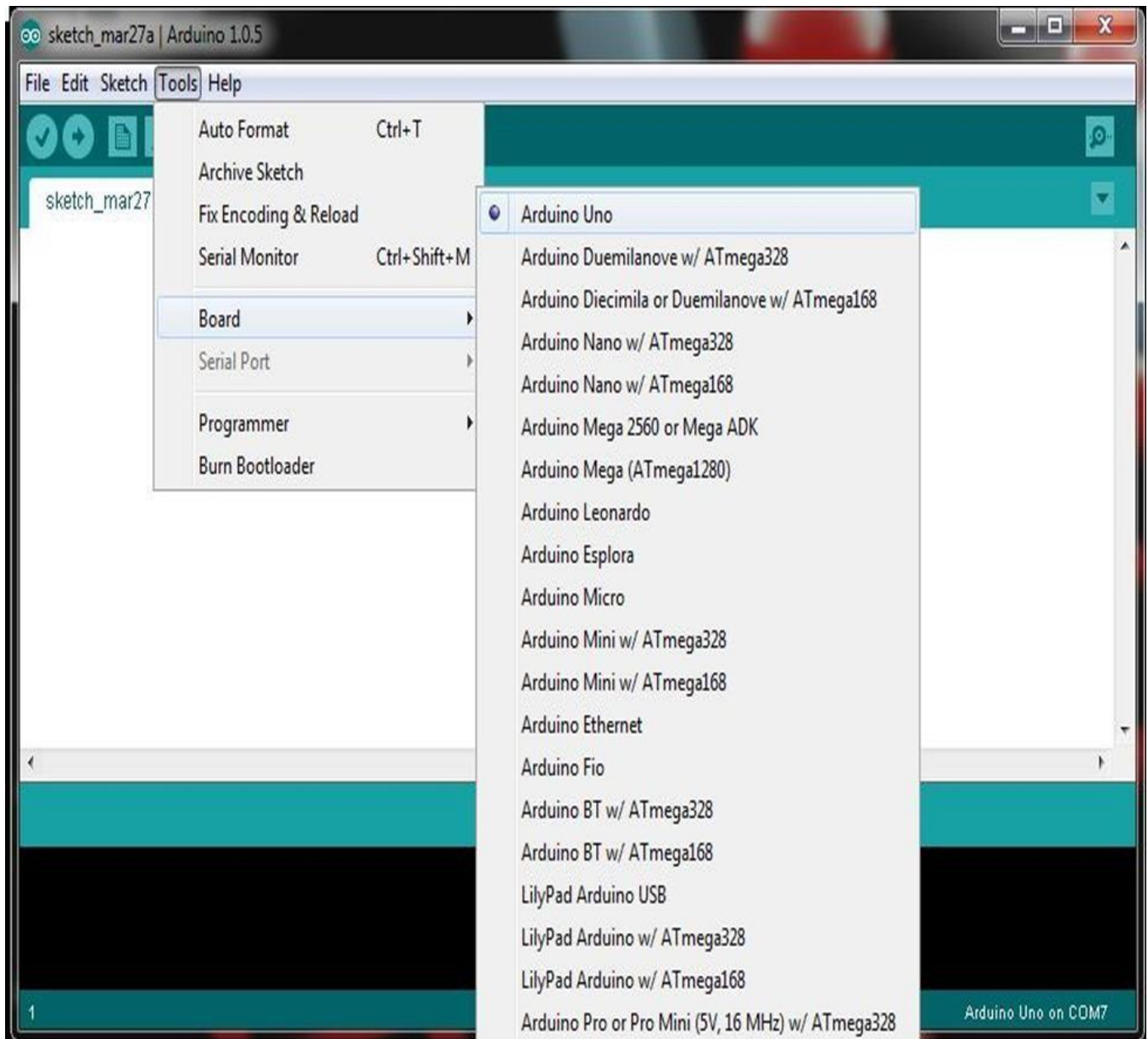
B5. ESP8266 using Windows



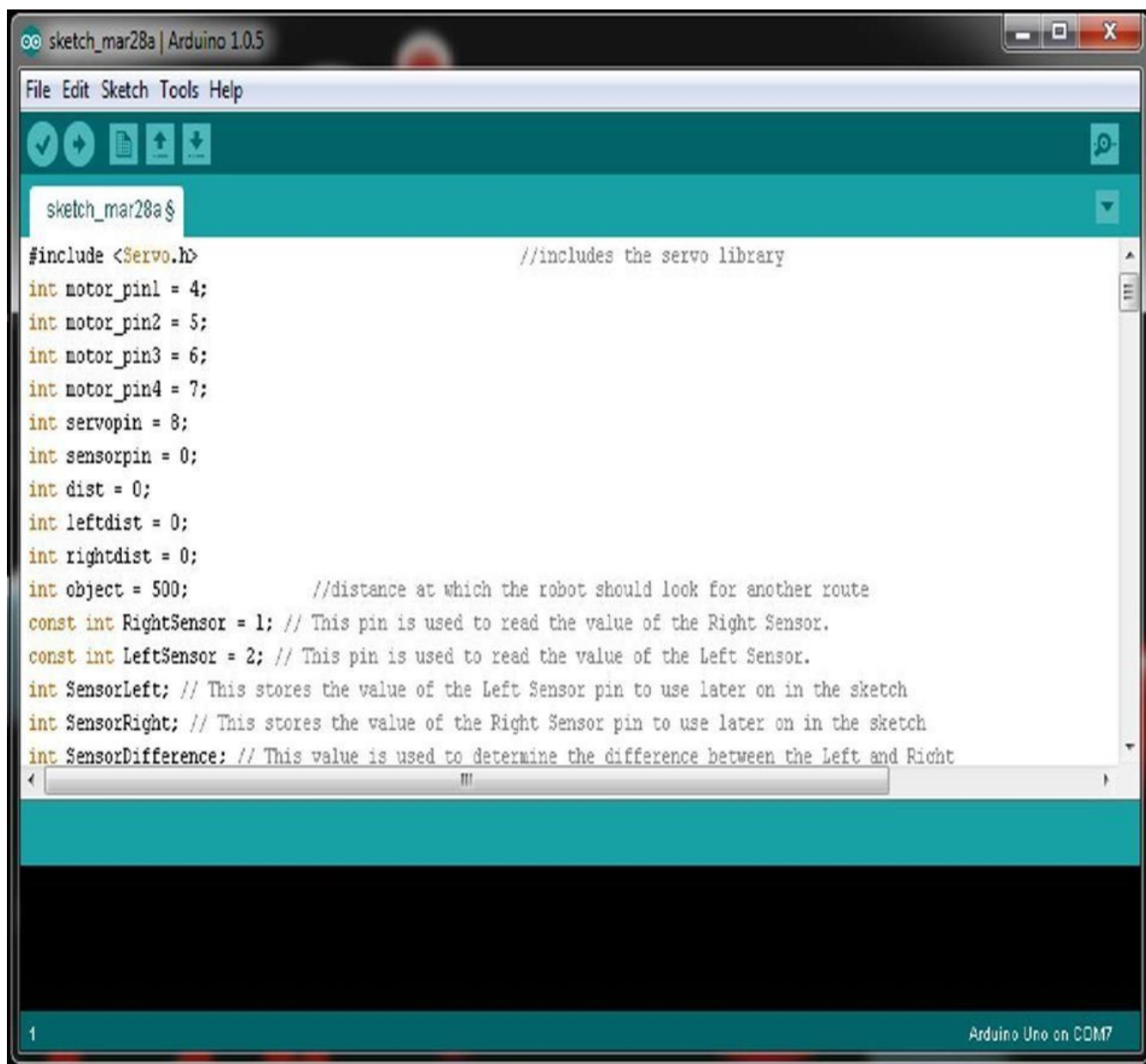
B6. Open Arduino IDE, which the program should be entered directly



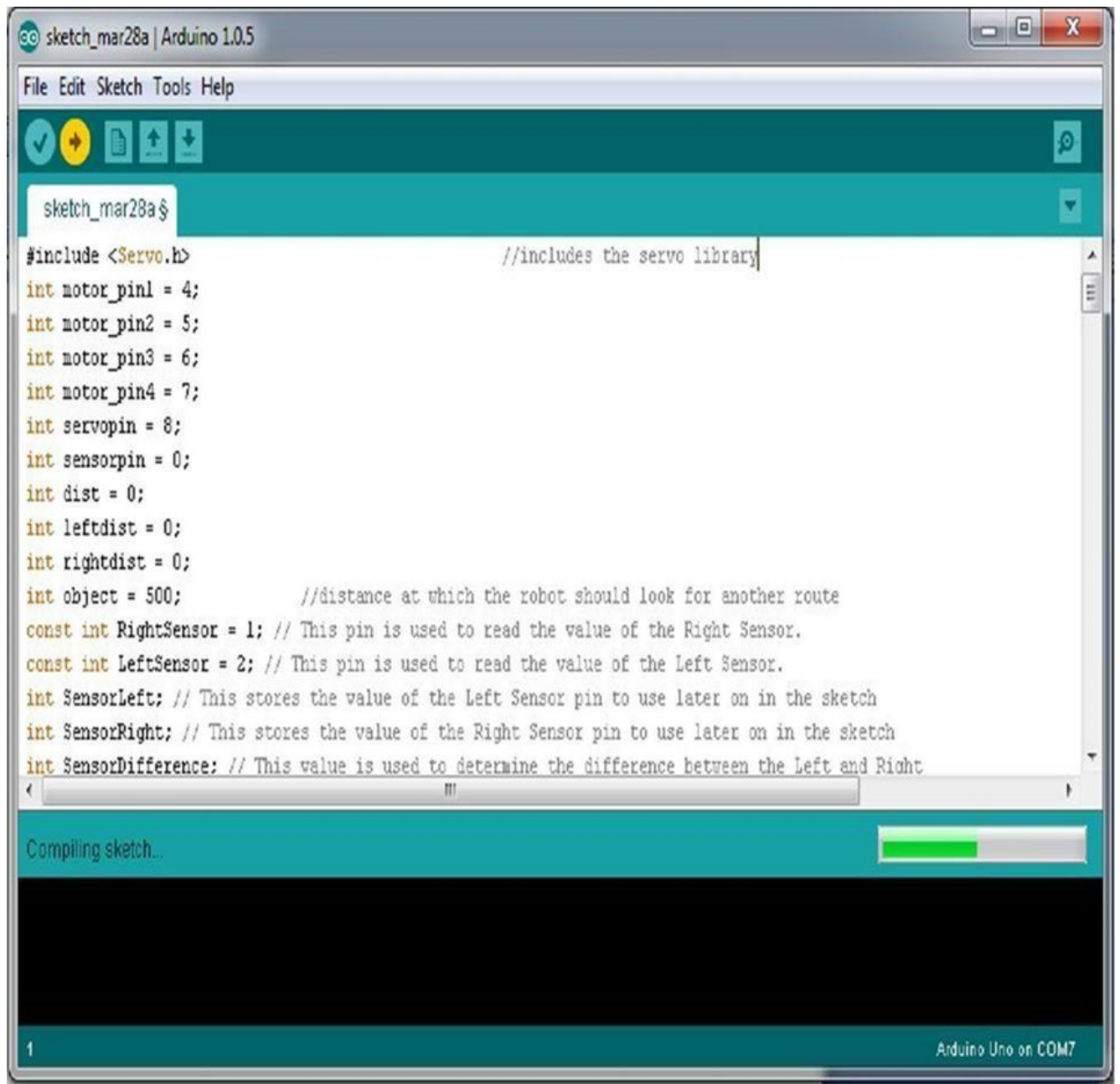
B7. COM Port from tool, where we can select the options and that we can change the modes and the serial points



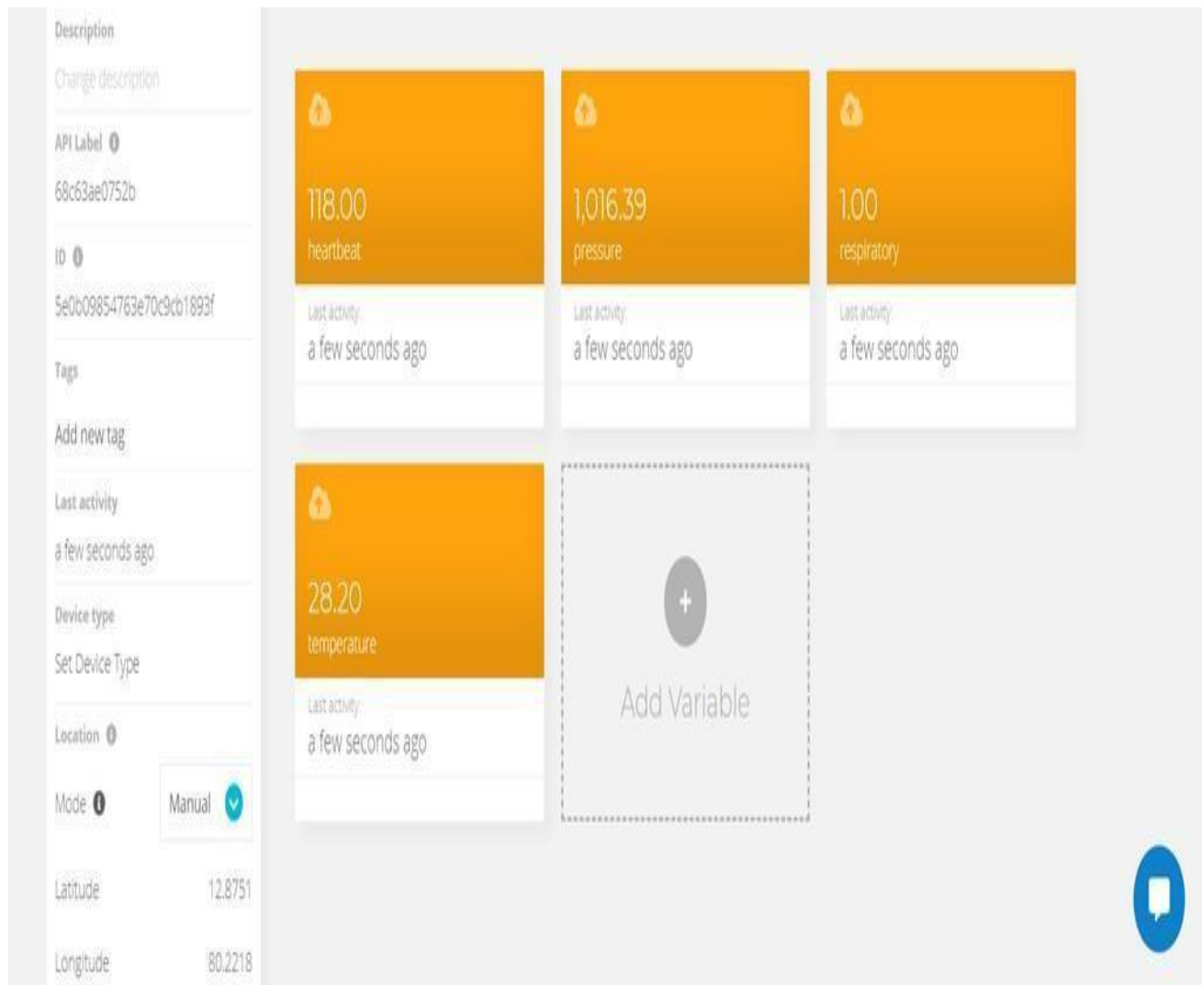
B8. Select the required Arduino board and there will be many options from the Arduino board



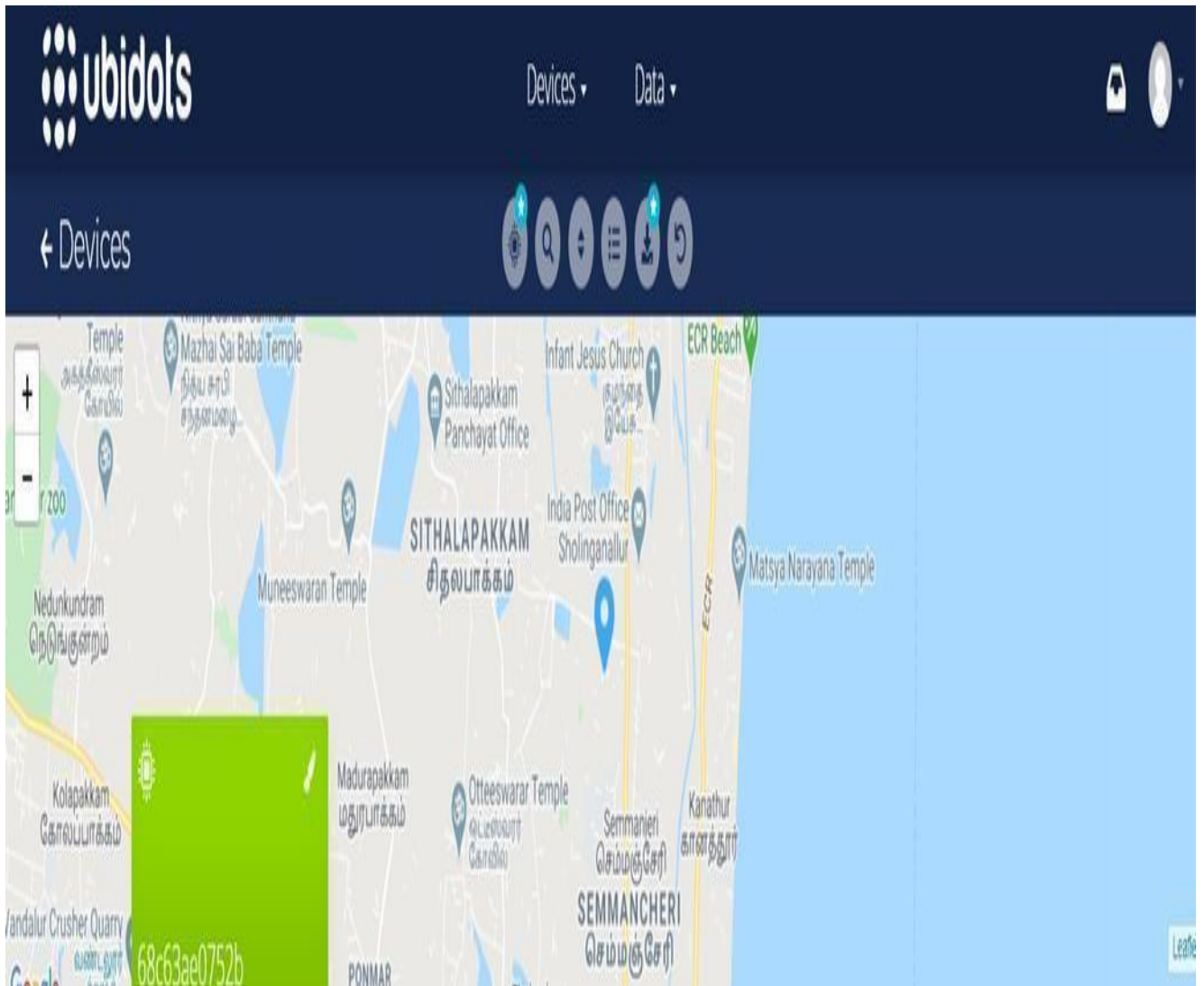
B9. Sketch in Arduino IDE, here we can add the code directly and we can monitor



B10. Compile and upload the Sketch to Arduino board here and we can easily handle the program



B11. In this database, we can monitor the present health condition of the pregnant lady so that we can easily reach for the situation of that lady



B12. The Google location shows the exact and correct position of the pregnant lady who wore the monitoring device



B13. A graphical storage data shows the fluctuating ranges and the situations of the receiving data continuously from the device

heartbeat alert! Inbox x



Notifications Ubidots <service@ubidots.com>

to me ▾

Hey there, heartbeat was 111.0 at 2020-02-17 10:13:00 +0530.

B14. Heartbeat Alert received for the registered mail ID

temperature alert! Inbox x



Notifications Ubidots <service@ubidots.com>

to me ▾

Hey there, temperature was 27.4 at 2020-02-17 15:39:11 +0530.



Reply



Forward

B15. Temperature Alert received for the registered mail ID

respiratory alert! Inbox x



Notifications Ubidots <service@ubidots.com>

to me ▼

Hey there, respiratory was 1.0 at 2020-02-18 10:23:13 +0530.

B16. Respiratory Alert received for the registered mail ID