

POLICE ASSISTANT DEVICE - CLOUD DATABASE FOR CRIMINAL RECORDS

A PROJECT REPORT

Submitted by

MRUDUL C U	211419105082
NARENDRAN M	211419105087
SANTHOSH KUMAR L	211419105126
THIRUMURUGAN A	211419105156

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRICAL AND ELECTRONICS ENGINEERING



PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

APRIL 2023

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **“POLICE ASSISTANT DEVICE-CLOUD DATABASE FOR CRIMINAL RECORDS”** is the bonafide work of **“MRUDUL. C.U (211419105082), NARENDRAN.M(211419105087), SANTHOSHKUMAR.L (2114191050126), THIRUMURUGAN.A (211419105156),”** who carried out the project work under my supervision.

SIGNATURE

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

HEAD OF THE DEPARTMENT

PROFESSOR

Department of Electrical and
Electronics Engineering,
Panimalar Engineering College,
Chennai-600 123

SIGNATURE

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

SUPERVISOR

ASSOCIATE PROFESSOR

Department of Electrical and
Electronics Engineering,
Panimalar Engineering College,
Chennai-600 123

Submitted for End Semester Project Viva Voce held on.....at
Panimalar Engineering College, Chennai.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We would like to express our sincere thanks to the people who extend their help during the course of our project work.

We would like to express our deep gratitude respected Secretary and Correspondent **Dr.P.CHINNADURAI,M.A.,Ph.D.**, for his kind words and motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI, Dr. C. SAKTHI KUMAR, M.E.Ph.D,** and **Dr. SARANYASREE SAKTHI KUMAR B.E., M.B.A.Ph.D**, for providing us with necessary facilities for completion of this project.

We thank our Principal **Dr.K.MANI, M.E, Ph.D.** for expressing his keen interest during this project work and encouragement provided to us throughout the course.

We are deeply indebted and extremely grateful to our Head of the Department **Dr.S.SELVI, M.E, Ph.D**, for the support extended throughout the project.

We would like to express our sincere thanks and deep sense gratitude to our project supervisor **Dr. P.HARIRAMAKRISHNAN M.E, Ph.D.**, for his variable guidance, suggestions paved wayfor the successful completion of the project work and all other faculty members of **EEE** department for their advice suggestions.

ABSTRACT

In IOT applications, Database security and safety is one of the important aspects in the area of networks system. The crime related to criminal actions and alert systems has been a tremendous rise in every day. This generates a crucial need for an effective criminal record system. In this project, a compact, efficient system is studied, designed and explored. An integrated UI system, which is to monitors criminal's record and crimes related to them with conditions and the right authentication using fingerprint sensor. In this project, we provide a trustworthy and solid Criminal Record System (CRS) design with elements boosting traffic signal security. In addition to the recognition system, our proposed recording system also includes a number of additional features. Internet of Things technology is one of the key aspects enabled by this system. The system is kept redundant to provide reliability even in the worst-case scenario, however due to budgetary restrictions, a trade-off between redundancy and cost was required. This method is made to work with practically every location that provides crime assistance.

Additional to this we have also implied the concept of powering a vehicle if only the person is not drunk. Alcohol presence is continuously monitored using sensors and whenever alcohol detected, the vehicle engine refuse to start, to avoid accidents.

TABLE OF CONTENTS

CHAPTER NO	TOPIC	PG.NO
	ABSTRACT	iv
	LIST OF TABLES	vi
	LIST OF ABBREVIATIONS	vii
	LIST OF FIGURES	viii
1.	INTRODUCTION	10
2.	LITERATURE SURVEY	12
3.	SYSTEM DESIGN	14
	3.1 Existing System	14
	3.2 Proposed Method	15
	3.2.1 Proposed Method Working	15
	3.3 Circuit Diagram	16
	3.4 Block Diagram Description	
	3.5 Hardware Components	17
	3.5.1 ESP32 Microcontroller	18
	3.5.1.1 Introduction to ESP32	19
	3.5.1.2 General Description	20
	3.5.1.3 Specification of ESP32	21
	3.5.2 Power supply	22

3.5.2.1	General Description	22
3.5.2.2	Product Description	22
3.5.2.3	Features	23
3.5.2.4	Application	23
3.5.3	Fingerprint Reader with Arduino	24
3.5.3.1	General Description	24
3.5.3.2	Features	24
3.5.3.3	Applications	25
3.5.4	DC Motor	25
3.5.4.1	General Description	25
3.5.4.2	Product Description	25
3.5.4.3	Features	26
3.5.4.4	Applications	26
3.5.5	IOT	35
3.5.5.1	Web Server	28
3.5.5.2	Web Server – Controlling Section	30
3.5.5.3	Features	30
3.5.5.4	Applications	31
3.5.6	16 × 2 LCD	38
3.5.6.1	General Description	31
3.5.6.2	Product Description	31
3.5.6.3	Features	32
3.5.7	Gas Sensor	33

3.5.7.1	General Description	45
3.5.7.2	Product Description	33
3.5.7.3	Features	34
3.5.7.4	Application	34
3.5.8	IR Sensor	34
3.5.8.1	General Description	34
3.5.8.2	Product Description	35
3.5.8.3	Features	36
3.5.8.4	Applications	36
4.	SYSTEM IMPLEMENTATION AND RESULT	36
4.1	Software Requirements	36
4.1.1	Arduino IDE	46
4.1.2	Embedded C	47
4.1.2.1	About Embedded C	48
4.1.2.2	Named Registers	48
4.1.2.3	Embedded C Portability	49
4.1.3	Python	57
4.1.3.1	Software Description	58
4.2	System Prototype	58
4.3	Software Result	58
5.	CONCLUSION	59
	REFERENCE	60

LIST OF TABLES

TABLE NO.	NAME OF THE TABLE	PAGE NO
1.1	Microcontroller Arduino Description	43

LIST OF ABBREVIATION

Abbreviation	Description
CRMS	Criminal Record Management System
FIR	First Information Report
UART	Universal Asynchronous Receiver-Transmitter
LCD	Liquid Crystal Display
IR sensor	Infra-Red sensor
I/O	Input/Output
IOT	Internet Of Thinking
WSN	Wireless Sensor Network
DC Motor	Direct Current Motor
Arduino IDE	Integrated Development Environment
ESP32	Espressif Systems
UI	User Interface
API	Application Program Interface

LIST OF FIGURES

FIGURE NO.	NAME OF THE FIGURE	PAGE NO
3.1	Block diagram of proposed system	16
3.2	Circuit diagram of proposed system	17
3.3	Function diagram of microcontroller	20
3.4	Diagram of ESP32	21
3.5	Power adapter	23
3.6	Fingerprint reader with display	24
3.7	DC Motor	26
3.8	Web Server	28
3.9	IOT load Control	30
3.10	16×2 LCD Display	32
3.11	Gas sensor	34
3.12	IR sensor	35
3.13	Arduino display	38
4.1.3.2	Code snippet	60
4.2	System prototype	60

CHAPTER 1

INTRODUCTION

Police provides safety to citizens. It always remains steady for arresting any criminal who is a threat for the safety of society. After registering the FIR from any citizen, police starts its work and on that basis it arrests the criminals if proofs are found against them. Once the criminals are arrested, police starts investigation from them. After getting all the proofs against the criminal, it is the duty of the police to present all the proofs honestly to the court so that the right man can get right punishment.

The true and right information provided by the people to police helps a lot in arresting the criminals who try to spoil the peaceful environment of society. Along with low salary scale, facilities of modern technology such as computerized system of keeping records are not provided to police department which causes low efficiency. As it is the age of computers and all the organizations today use computers to maintain their records, so this facility should also be given to police department in order to increase their efficiency and to save their time.

In our Project we are going to implement a CRMS (Criminal Record Management System). This is a database system in which police will keep the record of criminals who have been arrested, to be arrested or escaped. This will help the police department to manage their records easily. In police system when an incident occurs, a petitioner reports an FIR (First Information Report). Police starts investigation according to law on this FIR. An investigation officer supervises the investigation process. The main concerning people in the whole process are petitioner (The person who files an FIR), Victim, Accused/Criminal, investigation officer.

The Online Criminal Record Management System applies to police stations across the district and specifically looks into the subject crime detection, conviction of criminals depending on a highly responsive backbone of information management. The efficiency of the police and the effectiveness with which it tackles crime depend on what quality of information it can derive from its existing records and how fast it can have access to it. This system is made to keep the records about the prisoners and about the crimes. Police can login as a user and can add the details of prisoners like name, age, address, crime and punishment. They can also write the First Information Report and can save it. FIR's date, time, number and details can be seen any time if required by the registered user. This system gives unique id to every FIR as required and the prisoner number will also be unique. Additionally, it tells about any crime that is done through the id of complaint made to the police or if any FIR is done and if the criminal gets caught, the information can be updated about the case.

The proposed CRMS enhances the crime recording operations of the NSF (National Security Force). The data used by the CRMS is stored in a centralized database which holds information about criminals, crime and users of the system. The database is the basis for all actions in the system and can be easily updated and used to aid in all of the system's processes, that is, all of the required information is stored in one central location and thus is easily accessible. Furthermore, the correctness of the centralized database will allow functions such as crime report generation and statistical analysis of crime data. This is a more effective storage method than a paper-based file system.

In addition to the functions highlighted above, the system performs the basic functions of storage, retrieval and manipulation of crime and criminal data and information.

The benefits of the proposed system are as follows:

- Interstation communication in real time
- Centralized data handling
- Reduced time consumption
- Computerized record keeping with manpower

CHAPTER 2

LITERATURE SURVEY

I. In this paper [1] titled “A Real-Time Records Management System for National Security Agencies”, Oludeli Awodeli, Onulri Ernest.E focus on the implementation of criminal records management system. It is database system in which the police keep the record of criminals who have been arrested, to be arrested, or escaped. This will help the police department in enhanced management of information. The main entities in the whole process include: the petitioner (the person who files a First Incident Report (FIR)), victim, accused or criminal, case and investigating officer.

II. In this paper [2] titled “Criminal Report Management System”, Sourav Bhowmick developed a web based application that provides managing the data and various information about the criminals and their crimes. Not only this but also it provides the information and current status about the police stations. It stores the GD, FIR, number of cases and each and every detail of the criminals. This system also provides a search facility to know if there is any criminal record about any person.

III. In this paper [3] titled “Crime Reporting and Crime Updates”, Mohammad Shahnawaz developed an application that applies to all Police stations across the country and specifically looks into the subject of Crime Record Management. It is well understood that Crime Prevention, Detection and Conviction of criminals depend on a highly responsive backbone of Information Management. The efficiency of the police function and the effectiveness with which it tackles crime depend on what quality of information it can derive from its existing records and

how fast it can have access to it. Initially, the system will be implemented across Cities and Towns and later on, be interlinked so that a police detective can access information across all records in the state thus helping speedy and successful completion to cases. The project has been planned to be having the view of distributed architecture, with centralized storage of the database. The application for the storage of the data has been planned.

IV. In this paper [4] titled “Crime Reporting Interface Design using Mobile Technology”, Srinidhi Eragam Reddy, Ramya Sahithi Amathi and Priyanka Vakkalagadda designed a project that aims to develop a crime file to maintain computerized records of all the FIR against crime. The system is a desktop application that can be accessed throughout the police department. This system can be used as an application for the crime file of the police department to manage the records of different activities related to first information report.

V. In this paper [5] titled “A Scalable Online Crime Reporting System”, R. G. Jimoh, K. T. Ojulari, and O. A. Enikuomehin designed a project that aims to assist the Nigerian Police in their bid to solve crimes with timely and useful information about criminals and/or their mode of operations so as to nip in the bud criminal activities in a given locality. Finally, a prototype crime reporting system was designed that relies on four reporting forms: a complaint or dispatch reporting form, a crime event report form, follow-up investigation report form, and an arrest report form. The system consists of three functional modules: a data capture module, a report management and control module, and a data utilization module. Future work on crime reporting system can be tailored towards accessibility (mobile version), awareness and improvement on the usage.

CHAPTER 3

SYSTEM DESIGN

3.1 EXISTING SYSTEM

The existing system that is being used by police department pertaining to the information of the prisoners, stores the name of the prisoners, information of the crime, date of FIR, background of the criminal and duration of the prison. However, once the period comes to an end, and the individual is released, it becomes difficult for the personnel serving the police forces to keep a track of the date and time of the release of the particular offender. Since the information of the release date of the offender is not present with each and every personnel, they are not notified on time, leading to chaos.

DISADVANTAGE

- Fast recording is not possible.
- Time-management is an issue in the existing approach.
- Consumes large volume of paper work.

3.2 PROPOSED SYSTEM

The aim of the proposed system is to cut through the chaos and workload, by innovating the current system. The proposed system notifies each and every personnel with the application on his android phone, about the release of an offender, including the other details. It eliminates the existing issue where the department of police of different area of jurisdiction remains unaware of the release of the criminal and his/her extra details, by creating a notification every time there is a sight of a criminal. This approach can thereby reduce the effort of the police and save their time, which could possibly be used in a productive manner.

ADVANTAGE

- It ensures information accuracy.
- This is a time-efficient approach.
- It minimizes manual information passages.
- The possibility of getting the work done is quite certain.
- It reduces redundancy.

3.3 BLOCK DIAGRAM

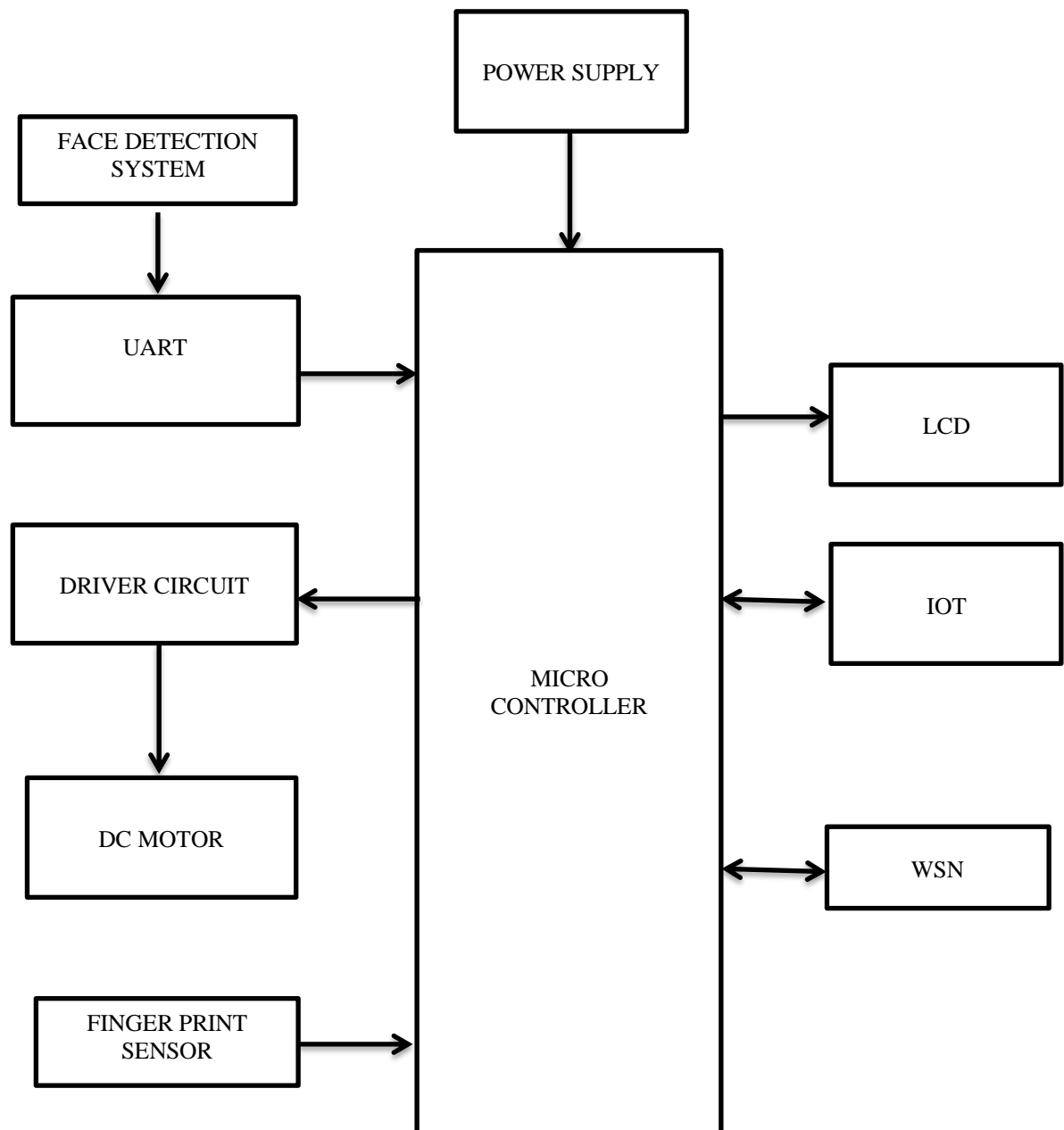


Figure 3.1 Block Diagram of Proposed System

3.3 Proposed Model Diagram

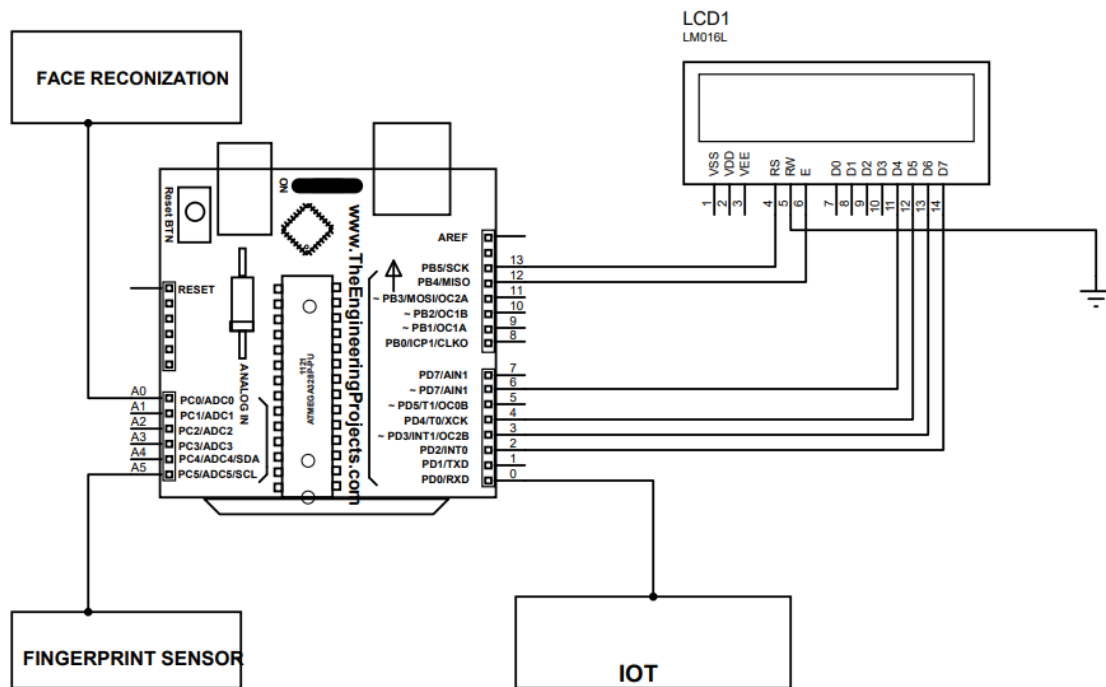


Figure 3.2 Circuit Diagram of Proposed System

Block Diagram Description

- Initially power supply comes from ac to dc adapter (230V ac) converted to 12V dc to the ESP32 Microcontroller.
- IN2596 Buck regulator step-downs 12V dc to 5V for UART and 5V for fingerprint sensor.
- For data transmission and API calling we use the WSN hardware module, so that we use the mobile phone network to connect with the microcontroller.
- If the person is detected to have consumed alcohol neither wearing a seat belt nor helmet
- If the person is to be suspected as a violator, the face of the person is recognized using the face recognition IOT system. The crime is uploaded in the database.
- After recording the face and data related to the violator, the user can access whenever he/she wants using the fingerprint sensors.
- The data is also stored in the cloud network or in local backup server used by the police servers.
- These data contain the criminal violation, type of violation, suspect's personal information, date and time of crime occurred.
- These database are to be maintained in the cloud server provided by the police network and it can also be impenetrable.

3.4 HARDWARE COMPONENTS

- MICROCONTROLLER
- FINGER PRINT SENSOR
- UART
- DRIVER CIRCUIT
- DC MOTOR
- IOT
- GAS SENSOR
- LCD DISPLAY
- IR SENSOR
- POWER SUPPLY
- WSN

SOFTWARE COMPONENTS:

- EMBEDDED C
- ARDUINO IDE
- PYTHON IDE

3.4 HARDWARE DESIGN

3.4.1 ESP32 MICROCONTROLLER

Arduino is a great platform for beginners into the World of Microcontrollers and Embedded Systems. With a lot of cheap sensors and modules, you can make several projects either as a hobby or even commercial.

As technology advanced, new project ideas and implementations came into play and one particular concept is the Internet of Things or IoT. It is a connected platform, where several “things” or devices are connected over internet for exchange of information.

In DIY community, the IOT projects are mainly focused on Home Automation and Smart Home applications but commercial and industrial IoT projects have far complex implementations like Machine Learning, Artificial Intelligence, Wireless Sensor Networks etc.

The important thing in this brief intro is whether it is a small DIY project by a hobbyist or a complex industrial project, any IoT project must have connectivity to Internet. This is where the likes of ESP8266 and ESP32 come into picture.

If you want to add Wi-Fi connectivity to your projects, then ESP8266 is a great option. But if you want build a complete system with Wi-Fi connectivity, Bluetooth connectivity, high resolution ADCs, DAC, Serial Connectivity and many other features, then ESP32 is the ultimate choice.

3.4.1.1 Introduction to ESP32?

ESP32 is a low-cost System on Chip (SoC) Microcontroller from Espressif Systems, the developers of the famous ESP8266 SoC. It is a successor to ESP8266 SoC and comes in both single-core and dual-core variations of the Tensilica's 32-bit Xtensa LX6 Microprocessor with integrated Wi-Fi and Bluetooth.

The good thing about ESP32, like ESP8266 is its integrated RF components like Power Amplifier, Low-Noise Receive Amplifier, Antenna Switch, Filters and RF Balun. This makes designing hardware around ESP32 very easy as you require very few external components.

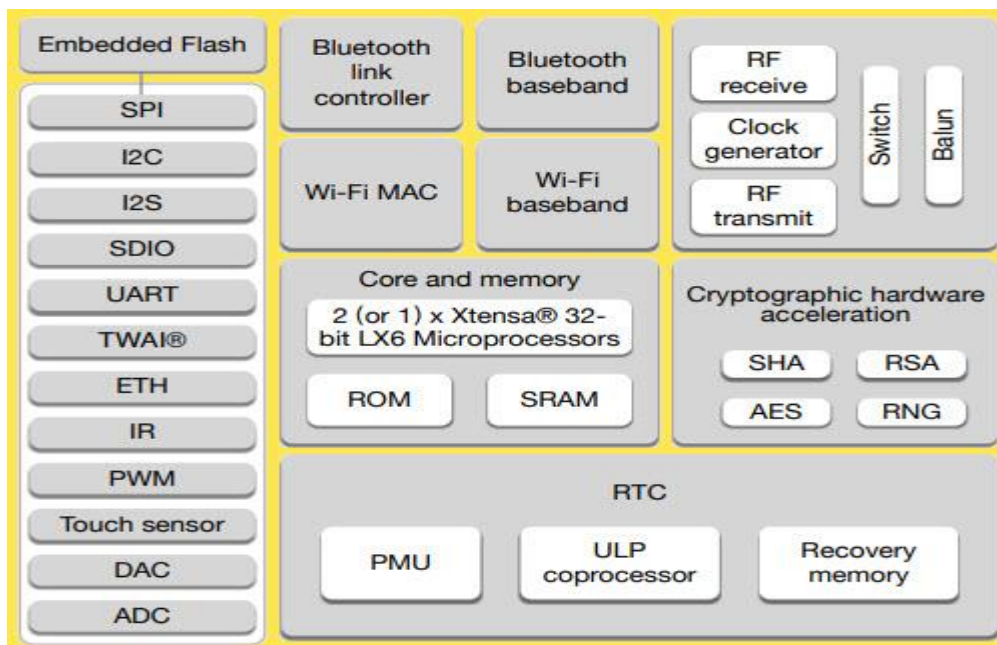


Figure 3.3 Function Diagram of Microcontroller

Another important thing to know about ESP32 is that it is manufactured using TSMC's ultra-low-power 40 nm technology. So, designing battery operated applications like wearables, audio equipment, baby monitors, smart watches, etc., using ESP32 should be very easy.

3.4.1.2 GENERAL DESCRIPTION

ESP32 is a series of low-cost, low-power system on a chip microcontroller with integrated Wi-Fi and dual-mode Bluetooth. The ESP32 series employs either a Tensilica Xtensa LX6 microprocessor in both dual-core and single-core variations, Xtensa LX7 dual-core microprocessor or a single-core RISC-V microprocessor and includes built-in antenna switches, RF balun, power amplifier, low-noise receive amplifier, filters, and power-management modules. ESP32 is created and developed by Espressif Systems, a Shanghai-based Chinese company, and is manufactured by TSMC using their 40 nm process.



Figure 3.4 Diagram of E S P 3 2

3.4.1.3 Specifications of ESP32

ESP32 has a lot more features than ESP8266 and it is difficult to include all the specifications in this Getting Started with ESP32 guide. So, I made a list

of some of the important specifications of ESP32 here. But for complete set of specifications, I strongly suggest you to refer to the Datasheet.

- Single or Dual-Core 32-bit LX6 Microprocessor with clock frequency up to 240 MHz.
- 520 KB of SRAM, 448 KB of ROM and 16 KB of RTC SRAM.
- Supports 802.11 b/g/n Wi-Fi connectivity with speeds up to 150 Mbps.
- Support for both Classic Bluetooth v4.2 and BLE specifications.
- 34 Programmable GPIOs.
- Up to 18 channels of 12-bit SAR ADC and 2 channels of 8-bit DAC
- Serial Connectivity include 4 x SPI, 2 x I²C, 2 x I²S, 3 x UART.
- Ethernet MAC for physical LAN Communication (requires external PHY).
- 1 Host controller for SD/SDIO/MMC and 1 Slave controller for SDIO/SPI.
- Motor PWM and up to 16-channels of LED PWM.
- Secure Boot and Flash Encryption.
- Cryptographic Hardware Acceleration for AES, Hash (SHA-2), RSA, ECC and RNG.

3.4.2 POWER SUPPLY

3.4.2.1 GENERAL DESCRIPTION

An adapter is a device that converts attributes of one electrical device or system to those of an otherwise incompatible device or system. Some modify power or signal attributes, while others merely adapt the physical form of one electrical connector to another. In a computer, an adapter is often built into a card that can be inserted into a slot on the computer's motherboard. The card adapts information that is exchanged between the computer's microprocessor and the devices that the card supports.

3.4.2.2 PRODUCT DESCRIPTION

An electric power adapter may enable connection of a power plug, sometimes called, used in one region to a AC power socket used in another, by offering connections for the disparate contact arrangements, while not changing the voltage. An AC adapter, also called a "recharger", is a small power supply that changes household electric current from distribution voltage) to low voltage DC suitable for consumer electronics. Some modify power or signal attributes, while others merely adapt the physical form of one electrical connector to another. For computers and related items, one kind of serial port adapter enables connections between 25-contact and nine-contact connectors, but does not affect electrical power- and signalling-related attributes.



Figure 3.5 Power Adapter

3.4.2.3 FEATURES

- Output current:1A
- Supply voltage: 220-230VAC
- Output voltage: 12VDC
- Reduced costs
- Increased value across front-office and back-office functions

- Access to current, accurate, and consistent data
- It generates adapter metadata as WSDL files with J2CA extension.

3.4.2.4 APPLICATIONS

- Back-end systems which need to send purchase order data to oracle applications send it to the integration service via integration server client.
- SMPS applications.

3.4.3 FINGERPRINT READER WITH BOARD

3.4.3.1 GENERAL DESCRIPTION

SM-621 is RS232 /UART fingerprint module scanner for the demand of access control system, door lock, T&A and safety box OEM POS Consisting of high function DSP, large capacity FLASH and color CMOS, etc, SM621 optical fingerprint module can conduct fingerprint enrollment, image processing, templates storage, fingerprint matching and fingerprint searching. This Optical biometric fingerprint reader is with great features and can be embedded into a variety of end products, such as: access control, attendance, safety deposit box, car door locks. **PRODUCT DESCRIPTION**

R305 fingerprint module is fingerprint sensor with TTL UART interface for direct connections to microcontroller UART or to PC through MAX232 / USBSerial adapter. The user can store the fingerprint data in the module and can configure it in 1:1 or 1: N mode for identifying the person. The FP module can directly interface with 3v3 or 5v Microcontroller. A level converter (like MAX232) is required for interfacing with PC serial port.



Figure 3.6 Fingerprint Reader with Display

3.4.3.2 FEATURES

- Input voltage: 5v
- Interface: RS232.
- Matching Mode: 1:1 and 1:N.
- Baud rate: 9600 – 115200.
- Storage Capacity: 256.

3.4.3.3 APPLICATIONS

- Attendance system.
- Safety deposit box system.
- Car door locking system.

3.4.4 DC MOTOR

3.4.4.1 GENERAL DESCRIPTION

The relationship between torque vs speed and current is linear as shown left; as the load on a motor increases, Speed will decrease. The graph pictured

here represents the characteristics of a typical motor. As long as the motor is used in the area of high efficiency (as represented by the shaded area) long life and good performance can be expected. However, using the motor outside this range will result in high temperature rises and deterioration of motor parts. A motor's basic rating point is slightly lower than its maximum efficiency point. Load torque can be determined by measuring the current drawn when the motor is attached to a machine whose actual load value is known.

3.4.4.2

PRODUCT DESCRIPTION

Geared dc motors can be defined as an extension of dc motors. A geared DC Motor has a gear assembly attached to the motor. The speed of motor is counted in terms of rotations of the shaft per minute and is termed as RPM .The gear assembly helps in increasing the torque and reducing the speed. Using the correct combination of gears in a gear motor, its speed can be reduced to any desirable figure. This concept where gears reduce the speed of the vehicle but increase its torque is known as gear reduction. A DC motor can be used at a voltage lower than the rated voltage. But, below 1000 rpm, the speed becomes unstable, and the motor will not run smoothly.



Figure 3.7 DC Motor

3.4.4.3 FEATURES

- Supply voltage: 12VDC
- Speed: 60rpm
- Long Lifetime, Low Noise, Smooth Motion
- Equipped with high efficiency

3.4.4.3 APPLICATIONS

- Coin Changing equipment
- Peristaltic Pumps
- Damper Actuators
- Fan Oscillators
- Photo copier
- Ticket printer
- Low Current Consumption: 400 μ A
- Sleep Mode: 3 μ A
- Low Voltage Operation: 2.2 V – 3.6 V
- High Sensitivity (800 mV/g @ 1.5g)
- Selectable Sensitivity (± 1.5 g, ± 6 g)
- Fast Turn on Time (0.5 ms Enable Response Time)
- Self Test for Freefall Detect Diagnosis

APPLICATIONS

- Self-balancing robots
- Tilt-mode game controllers
- Model airplane auto pilot

- Car alarm systems
- Crash detection/airbag deployment

3.4.5 IOT

The Internet of things (IoT) is the network of everyday objects — physical things embedded with electronics, software, sensors, and connectivity enabling data exchange. Basically, a little networked computer is attached to a thing, allowing information exchange to and from that thing. Be it lightbulbs, toasters, refrigerators, flower pots, watches, fans, planes, trains, automobiles, or anything else around you, a little networked computer can be combined with it to accept input (especially object control) or to gather and generate informational output (typically object status or other sensory data).

This means computers will be permeating everything around us — ubiquitous embedded computing devices, uniquely identifiable, interconnected across the Internet. Because of low-cost, networkable microcontroller modules, the Internet of things is really starting to take off.

3.4.5.1 WEB SERVER

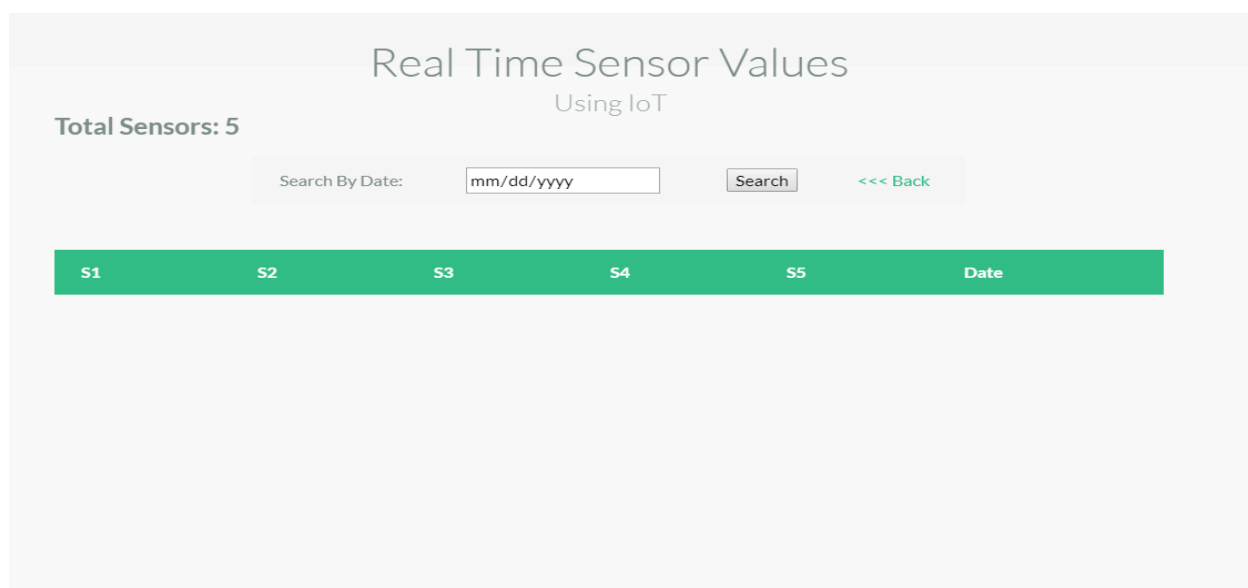


Figure 3.8 Web Server

Espresso's ESP32 delivers highly integrated Wi-Fi SoC solution to meet users' continuous demands for efficient power usage, compact design and reliable performance in the Internet of Things industry. With the complete and self-contained Wi-Fi networking capabilities, ESP32 can perform either as a standalone application or as the slave to a host MCU. When ESP32 hosts the application, it promptly boots up from the flash. The integrated high speed cache helps to increase the system performance and optimize the system memory. Also, ESP32 can be applied to any microcontroller design as a Wi-Fi adaptor through SPI / SDIO or I2C / UART interfaces. ESP32 integrates antenna switches, RF balun, power amplifier, low noise receive amplifier, filters and power management modules. The compact design minimizes the PCB size and requires minimal external circuitries. Besides the Wi-Fi functionalities, ESP32 also integrates an enhanced version of Tensilica's L106 Diamond series 32-bit processor and on-chip SRAM. It can be interfaced with external sensors and other devices through the GPIOs. Software Development Kit (SDK) provides sample codes for various applications.

Espressif Systems' Smart Connectivity Platform (ESCP) enables sophisticated features including fast switch between sleep and wakeup mode for energy-efficient purpose, adaptive radio biasing for low-power operation, advance signal processing, spur cancellation and radio co-existence mechanisms for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation.

Channel Frequencies:

- The RF transceiver supports the following channels according to IEEE802.11b/g/n standards

GHz Receiver:

- The 2.4 GHz receiver down-converts the RF signals to quadrature baseband signals and converts them to the digital domain with 2 high resolution high speed ADCs. To adapt to varying signal channel conditions, RF filters, automatic gain control (AGC), DC offset cancelation circuits and baseband filters are integrated within ESP32.

GHz Transmitter:

- The 2.4 GHz transmitter up-converts the quadrature baseband signals to 2.4 GHz, and drives the antenna with a high-power CMOS power amplifier. The function of digital calibration further improves the linearity of the power amplifier, enabling a state of art performance of delivering +19.5 dBm average power for 802.11b transmission and +16 dBm for 802.11n transmission.
- Additional calibrations are integrated to offset any imperfections of the radio, such as:
 - Carrier leakage
 - I/Q phase matching
 - Baseband nonlinearities
- These built-in calibration functions reduce the product test time and make the test equipment unnecessary

3.4.5.2 WEB SERVER: Controlling Section

Smart IOT Based Load Control



Figure 3.9 IOT Load Control

3.4.5.3 FEATURES

- Power Supply: DC +12v 1Amp.
- Auto data updating: 30sec
- Digital Output port Pins: +5V DC
- Message Format: *message or Data # (Start with * and End with #)
- Provided with 3 links
- Data updating to a specific web site
- Device controlling web site
- Data updating to a social network

3.4.5.4 APPLICATIONS

- Online Traffic monitoring

- Online Health monitoring
- Real time Transport and Logistics monitoring
- Daily life and domestics

3.4.6 16×2 LCD

3.4.6.1 GENERAL DESCRIPTION

LCD stands for liquid crystal display. They come in many sizes 8x1 , 8x2 , 10x2 , 16x1 , 16x2 , 16x4 , 20x2 , 20x4 , 24x2 , 30x2 , 32x2 , 40x2 etc . Many multinational companies like Philips Hitachi Panasonic make their own special kind of LCD'S to be used in their products. All the LCD'S performs the same functions (display characters numbers special characters ASCII characters etc). Their programming is also same and they all have same 14 pins (0-13) or 16 pins (0 to 15). Alphanumeric displays are used in a wide range of applications, including palmtop computers, word processors, photocopiers, point of sale terminals, medical instruments, cellular phones, etc.

3.4.6.2 PRODUCT DESCRIPTION

This is an LCD Display designed for E-blocks. It is a 16 character, 2-line alphanumeric LCD display connected to a single 9-way D-type connector. This allows the device to be connected to most E-Block I/O ports. The LCD display requires data in a serial format, which is detailed in the user guide below. The display also requires a 5V power supply. Please take care not to exceed 5V, as this will cause damage to the device. The 5V is best generated from the E-blocks Multi programmer or a 5V fixed regulated power supply. The 16 x 2 intelligent alphanumeric dot matrix displays is capable of displaying 224 different characters and symbols. A full list of the characters

and symbols is printed on pages 7/8 (note these symbols can vary between brand of LCD used). This booklet provides all the technical specifications for connecting the unit, which requires a single power supply (+5V).



Figure 3.10 16 x 2 LCD Display

3.4.6.3 FEATURES

- Input voltage: 5v
- E-blocks compatible
- Low cost
- Compatible with most I/O ports in the E-Block range
- Ease to develop programming code using Flow code icons

3.4.7 GAS SENSOR

3.4.7.1 GENERAL DESCRIPTION

In current technology scenario, monitoring of gases produced is very important. From home appliances such as air conditioners to electric chimneys and safety systems at industries monitoring of gases is very crucial. Gas sensors spontaneously react to the gas present, thus keeping the system updated about any alterations that occur in the concentration of molecules at gaseous state. The gas sensor module consists of a steel exoskeleton under which a sensing element is housed.

This sensing element is subjected to current through connecting leads. This current is known as heating current through it, the gases coming close to the sensing element get ionized and are absorbed by the sensing element. This changes the resistance of the sensing element which alters the value of the current going out of it. The connecting leads of the sensor are thick so that sensor can be connected firmly to the circuit and sufficient amount of heat gets conducted to the inside part. They are casted from copper and have tin plating over them.

3.4.7.2 PRODUCT DESCRIPTION

MQ-8 gas sensor composed by micro Al_2O_3 ceramic tube, Tin Dioxide (SnO_2) sensitive layer, measuring electrode and heater are fixed into a crust made by plastic and stainless steel net. MQ-8 gas sensor has high sensitivity to hydrogen gas and has anti-interference to gases. The enveloped MQ-8 have 6 pin, 4 of them are used to fetch signals, and other 2 are used for providing heating current. The MQ-8 gas module is mounted on a pcb board which has an operating voltage of 5VDC. The sensor output values can be get by means of both analog and digital.



Figure 3.11 Gas Sensor

3.4.7.3 FEATURES

- Operating voltage: 5VDC
- Operating current: 100-150mA
- Both analog and digital output
- Simple drive circuit

3.4.7.4 APPLICATIONS

- Hydrogen gas leakage detection
- Portable gas detector
- Fire safety detection system

3.4.8 IR SENSOR

3.4.8.1 GENERAL DESCRIPTION

IR LED emits infrared radiation. This radiation illuminates the surface in front of LED. Depending on reflectivity of the surface, amount of light reflected varies. This reflected light is made incident on reverse biased IR sensor. The amount of electron-hole pairs generated depends on intensity of incident IR radiation. Thus as intensity of incident ray varies, voltage across resistor will vary accordingly.

3.4.8.2 PRODUCT DESCRIPTION

An infrared sensor is an electronic device, that emits in order to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. Usually in the infrared spectrum, all the objects

radiate some form of thermal radiations. These types of radiations are invisible to our eyes, that can be detected by an infrared sensor. The emitter is simply an IR LED (Light Emitting Diode) and the detector is simply an IR photodiode which is sensitive to IR light of the same wavelength as that emitted by the IR LED. When IR light falls on the photodiode, The resistances and these output voltages, change in proportion to the magnitude of the IR light received.



Figure 3.12 IR Sensor

3.4.8.3 FEATURES

- Operating voltage:5VDC
- Output voltage: 0 or 5VDC
- Easy to assemble and use
- Onboard detection indication

- Effective distance range of 2cm

3.4.8.4 APPLICATIONS

- Augmentative communication devices
- Car locking systems
- Computers
- Signage
- Telephones

CHAPTER 4

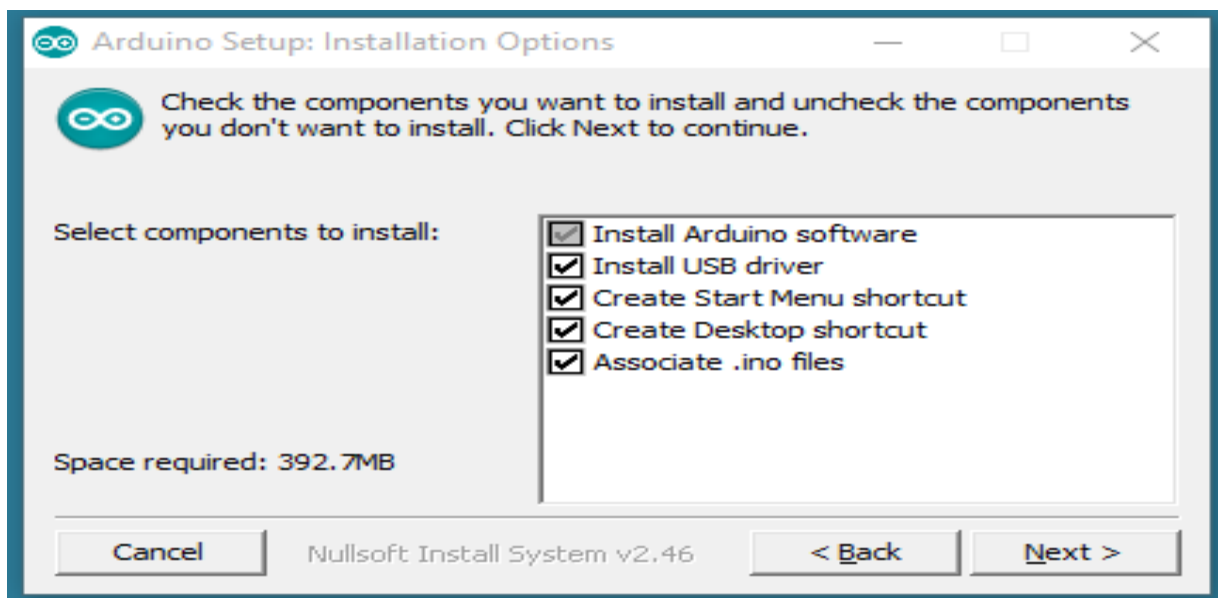
SOFTWARE REQUIREMENT

4.1 SOFTWARE DESCRIPTION:

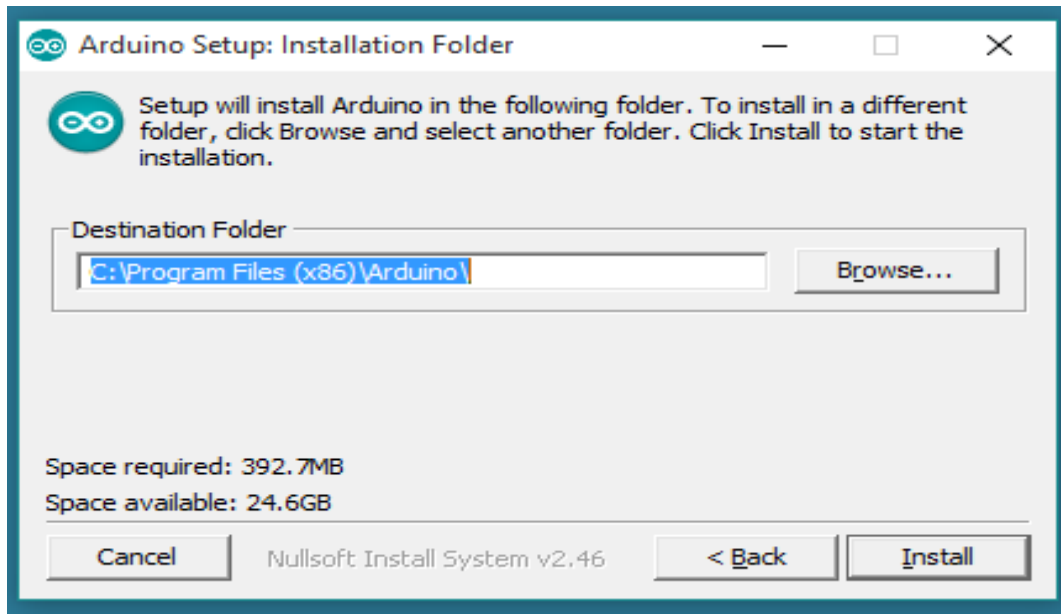
4.1.1 ARDUINO SOFTWARE (IDE)

Get the latest version from the download page. You can choose between the Installer (.exe) and the Zip packages. We suggest you use the first one that installs directly everything you need to use the Arduino Software (IDE), including the drivers. With the Zip package you need to install the drivers manually. The Zip file is also useful if you want to create a portable installation.

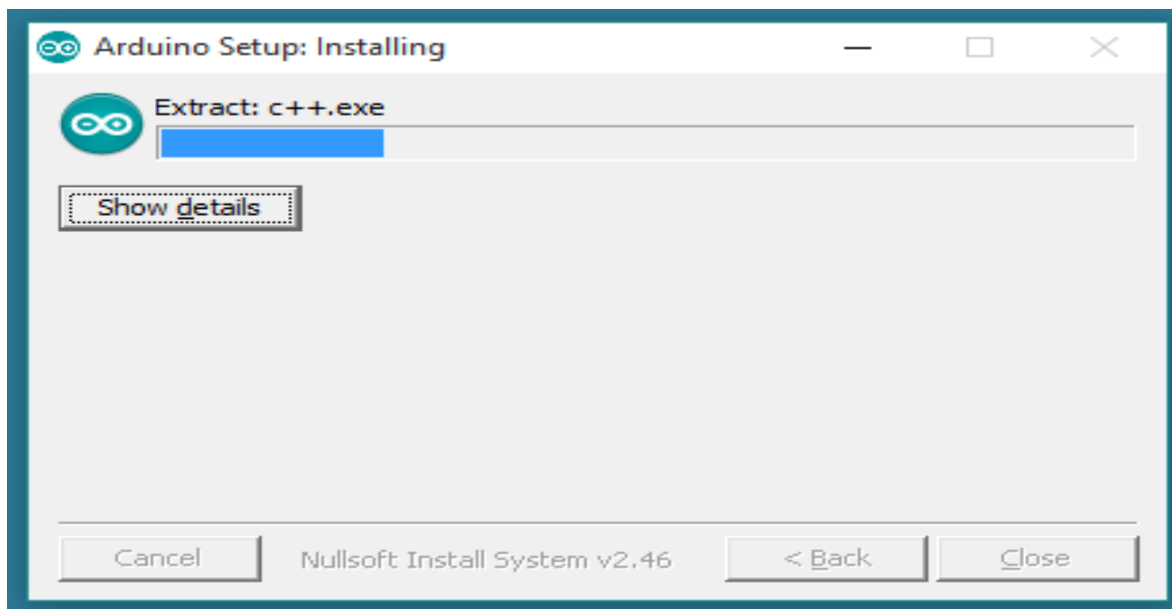
When the download finishes, proceed with the installation and please allow the driver installation process when you get a warning from the operating system.



Choose the components to install



Choose the installation directory (we suggest to keep the default one)



The process will extract and install all the required files to execute properly the Arduino Software (IDE)

Arduino Boot loader Issue

The current boot loader burned onto the Arduino UNO is not compatible with ROBOTC. In its current form, you will be able to download

the ROBOTC Firmware to the ArduinoUNO, but you will not be able to download any user programs.

The reason for this is because there is a bug in the Arduino UNO firmware that does not allow flash write commands to start at anywhere but the beginning of flash memory (0x000000). See the bottom of this page for more technical details.

Because ROBOTC is not able to burn a new bootloader as of today, you will need to use the Arduino's Open Source language with a modified bootloader file to re-burn your bootloader on your Arduino UNO boards. The enhanced bootloader is backwards compatible with the original one. That means you'll still be able to program it through the Arduino programming environment as before, in addition to ROBOTC for Arduino.

Hardware Needed

To burn a new version of the Arduino boot loader to your UNO, you'll need an AVR ISP Compatible downloader.

Using an AVR ISP (In System Programmer)

- Your Arduino UNO (to program)
- An AVR Programmer such as the AVR Pocket Programmer
- An AVR Programming Cable (the pocket programmer comes with one)

If you have extra Arduino boards, but no ISP programmer, SparkFun.com has a cool tutorial on how to flash a bootloader using an Arduino as an ISP.

Using another Arduino as an ISP

- Your Arduino UNO (to program)
- A Working Arduino (doesn't matter what kind)
- Some Male-to-Male Jumper Cables

For instructions on this method, take a look at the SparkFun.com website:
<http://www.sparkfun.com/tutorials/247>

Software Needed

ROBOTC is not currently able to burn a bootloader onto an Arduino board, so you'll need to download a copy of the latest version of the Arduino Open-Source programming language.

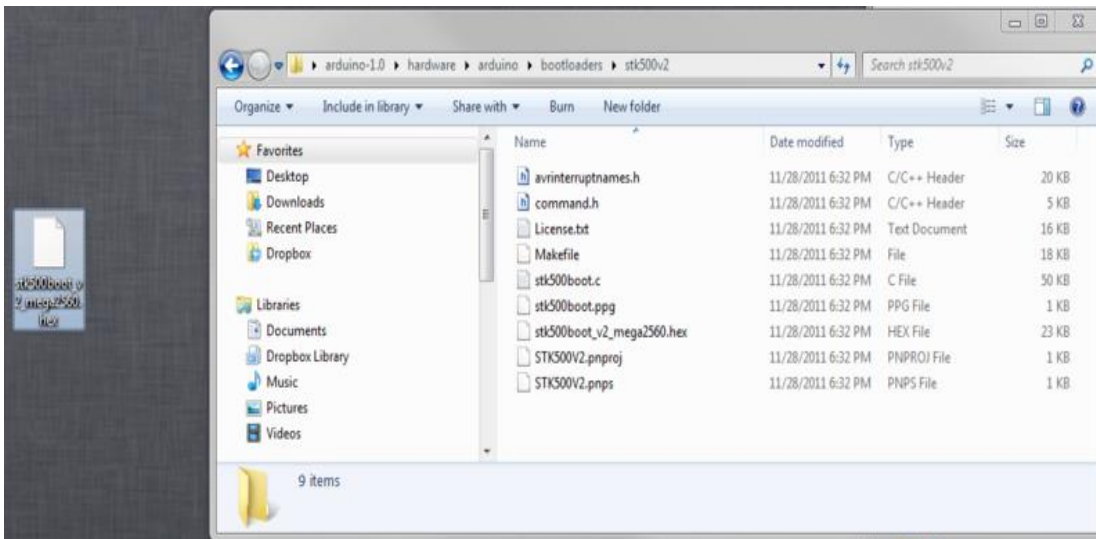
- Arduino Official Programming Language - Download Page

In addition, you'll need the ROBOTC modified bootloader. You can download that here:

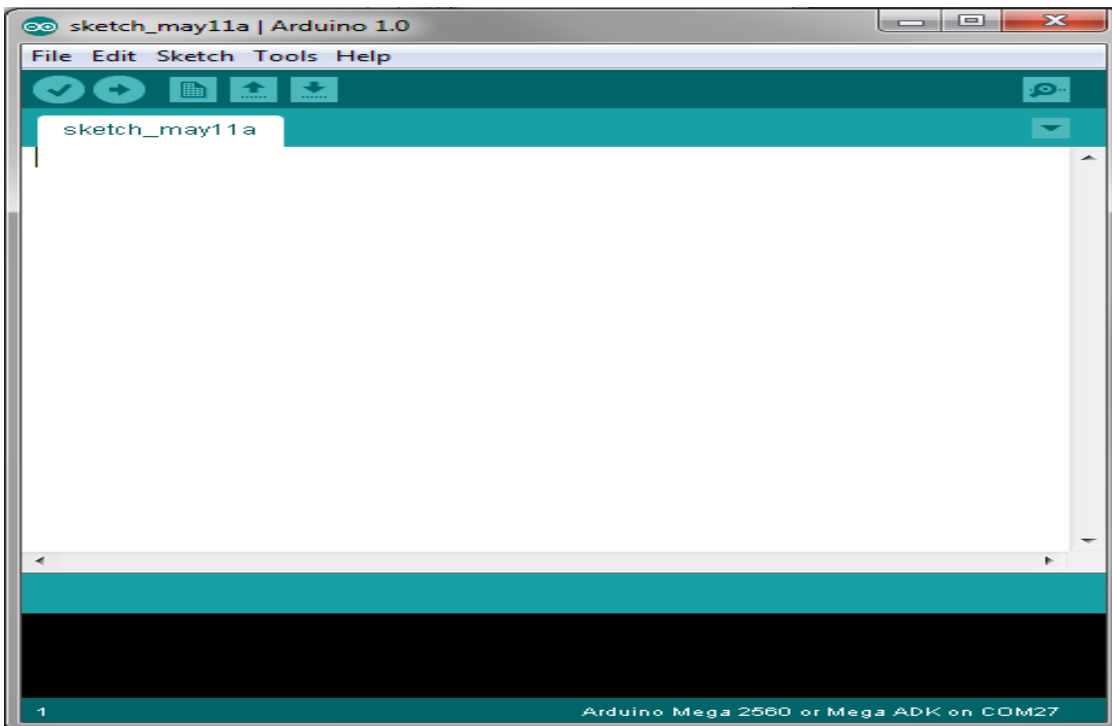
- ROBOTC Modified UNO Bootloader - Modified Bootloader

Bootload Download Instructions

- Download the Arduino Open Source Software and a copy of the Modified Bootloader File
- Copy the Modified Bootloader File into the /Arduino-1.0/hardware/arduino/bootloaders/stk500v2/ and overwrite the existing bootloader.



- Power up your Arduino UNO (either via USB or external power)
- Plug in your AVR ISP Programmer to your computer (make sure you have any required drivers installed)
- Connect your AVR ISP Programmer into your Arduino UNO Board via the ISP Header (the 2x3 header pins right above the Arduino Logo)
- Launch the Arduino Open Source Software



- Change your settings in the Arduino Software to look for an Arduino UNO
- Change your settings in the Arduino Software to select your ISP Programmer Type (Check your programmer's documentation for the exact model)

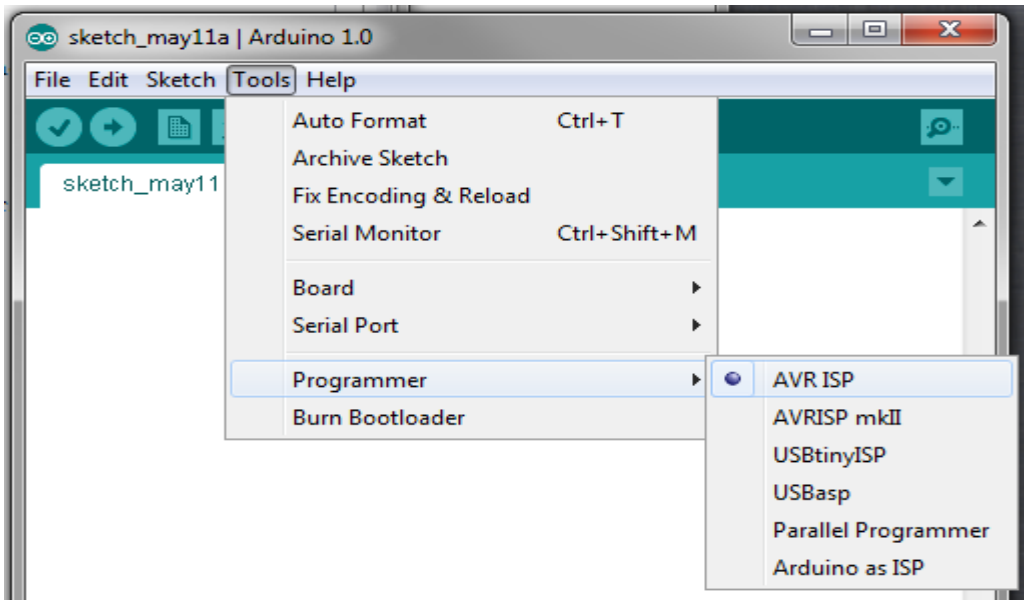
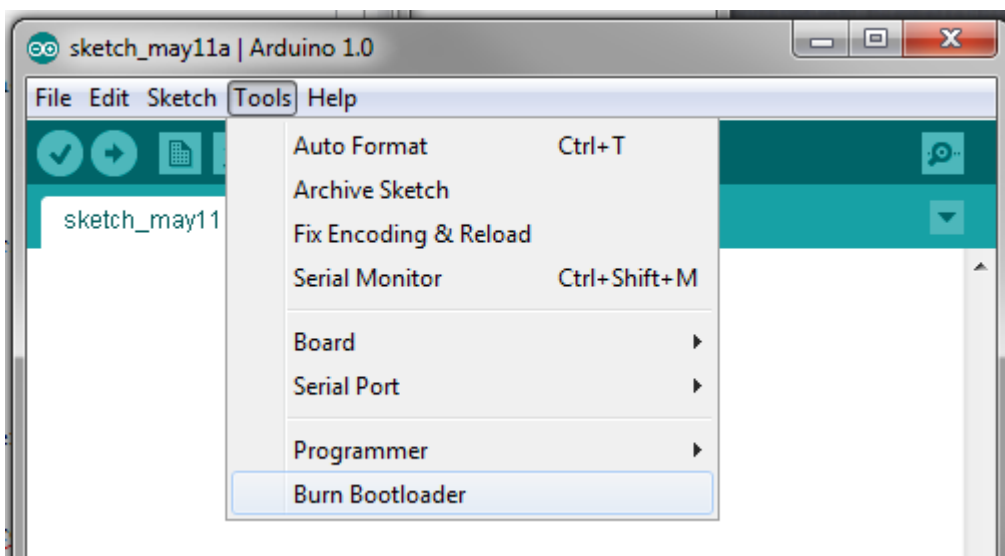


Figure 3.13 Arduino Display

- Select the "Burn Bootloader" option under the "Tools" menu. The modified bootloader will now be sent to your Arduino. This typically take a minute or so.



- You should be all set to download ROBOTC firmware and start using your Arduino UNO with ROBOTC.

Technical Details

The Arduino Boot loader sets the "erase Address" to zero every time the boot loader is called. ROBOTC called the "Load Address" command to set the address in which we want to write/verify when downloading program.

When writing a page of memory to the arduino, the Arduino boot loader will erase the existing page and write a whole new page.

In the scenario of downloading firmware, everything is great because the Erase Address and the Loaded Address both start at zero.

In the scenario of writing a user program, we start writing at memory location 0x7000, but the Boot loader erases information starting at location zero because the "Load Address" command doesn't update where to erase.

Our modification is to set both the Load Address and the Erase Address so the activity of writing a user program doesn't cause the firmware to be accidentally erased.

Microcontroller	Arduino UNO
Operating Voltage	5V Input Voltage
Input Voltage (Limits)	6 – 20 V
Digital I/O Pins	54 (Of which 14 provide PWM output)

Analog Input Pins	16
DC Current per I/O Pin	40mA
DC Current for 3.3V Pin	50mA
Flash Memory	256 Kb of which 8 Kb used by bootloader
SRAM	8KB
EEPROM	4KB
Lock Speed	16MHz

Table 1.1 Microcontroller Arduino Description

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 V_{in} 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.

They differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the programmed as a USB-to-serial converter.

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.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via a non-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50mA.
- **GND.** Ground pins.

The ATMEGA has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Each of the 54 digital pins on the Mega 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 20-50k Ohms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATMEGA USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be

configured to trigger an interrupt on a low value, a rising or falling edge, or a changing value. See the attach Interrupt() function for details.

- **PWM: 0to13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Duemilanove and Diecimila.
- **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.
- **I²C: 20 (SDA) and 21 (SCL).** Support I²C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I²C pins on the Duemilanove.

The Arduino UNO has 16 analog inputs, 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 analog Reference() function.

There are a couple of other pins on the board:

- **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

The Arduino UNO has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The Arduino UNO provides four hardware UARTs for TTL (5V) serial communication.

An ATMEGA on the board channels one of these over USB and provides a virtual comport to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the digital pins.

The Arduino UNO also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. To use the SPI communication, please see the Arduino UNO datasheet.

Programming

The Arduino UNO can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The Arduino UNO on the Arduino UNO comes preburned with a bootloader that 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.

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 is connected to the reset line of the Arduino UNO 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 Arduino 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 this data.

The Mega 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.

5.10 USB Over current Protection

Physical Characteristics and Shield Compatibility

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

The maximum length and width of the UNO PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to 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 other pins.

The UNO is designed to be compatible with most shields designed for the Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega and Duemilanove / Diecimila.

Please note that I²C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).

How to use Arduino

Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings by controlling lights, motors, and

other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino projects can be stand-alone or they can communicate with software on running on a computer (e.g. Flash, Processing, MaxMSP).

Arduino is a cross-platform program. You'll have to follow different instructions for your personal OS. Check on the Arduino site for the latest instructions. <http://arduino.cc/en/Guide/HomePage>

Once you have downloaded/unzipped the arduino IDE, you can plug the Arduino to your PC via USB cable.

4.1.2 EMBEDDED C

4.1.2.1 ABOUT EMBEDDED C

High-level language programming has long been in use for embedded-systems development. However, assembly programming still prevails, particularly for digital-signal processor (DSP) based systems. DSPs are often programmed in assembly language by programmers who know the processor architecture inside out. The key motivation for this practice is performance, despite the disadvantages of assembly programming when compared to high-level language programming.

If the video decoding takes 80 percent of the CPU-cycle budget instead of 90 percent, for instance, there are twice as many cycles available for audio processing. This coupling of performance to end-user features is characteristic of many of the real-time applications in which DSP processors are applied. DSPs have a highly specialized architecture to achieve the

performance requirements for signal processing applications within the limits of cost and power consumption set for consumer applications. Unlike a conventional Load-Store (RISC) architecture, DSPs have a data path with memory-access units that directly feed into the arithmetic units. Address registers are taken out of the general-purpose register file and placed next to the memory units in a separate register file.

A further specialization of the data path is the coupling of multiplication and addition to form a single cycle Multiply-accumulate unit (MAC). It is combined with special-purpose accumulator registers, which are separate from the general-purpose registers. Data memory is segmented and placed close to the MAC to achieve the high bandwidths required to keep up with the streamlined data path. Limits are often placed on the extent of memory-addressing operations. The localization of resources in the data path saves many data movements that typically take place in a Load-Store architecture.

The most important, common arithmetic extension to DSP architectures is the handling of saturated fixed-point operations by the arithmetic unit. Fixed-point arithmetic can be implemented with little additional cost over integer arithmetic. Automatic saturation (or clipping) significantly reduces the number of control-flow instructions needed for checking overflow explicitly in the program. Changes in technological and economic requirements make it more expensive to continue programming DSPs in assembly. Staying with the mobile phone as an example, the signal-processing algorithms required become increasingly complex. Features such as stronger error correction and encryption must be added. Communication

protocols become more sophisticated and require much more code to implement. In certain markets, multiple protocol stacks are implemented to be compatible with multiple service providers. In addition, backward compatibility with older protocols is needed to stay synchronized with provider networks that are in a slow process of upgrading.

Today, most embedded processors are offered with C compilers. Despite this, programming DSPs is still done in assembly for the signal processing parts or, at best, by using assembly-written libraries supplied by manufacturers. The key reason for this is that although the architecture is well matched to the requirements of the signal-processing application, there is no way to express the algorithms efficiently and in a natural way in Standard C. Saturated arithmetic.

For example, is required in many algorithms and is supplied as a primitive in many DSPs. However, there is no such primitive in Standard C. To express saturated arithmetic in C requires comparisons, conditional statements, and correcting assignments. Instead of using a primitive, the operation is spread over a number of statements that are difficult to recognize as a single primitive by a compiler.

DESCRIPTION

Embedded C is designed to bridge the performance mismatch between Standard C and the embedded hardware and application architecture. It extends the C language with the primitives that are needed by signal-processing applications and that are commonly provided by DSP processors. The design of the support for fixed-point data types and named

address spaces in Embedded C is based on DSP-C. DSP-C [1] is an industry-designed extension of C with which experience was gained since 1998 by various DSP manufacturers in their compilers. For the development of DSP-C by ACE (the company three of us work for), cooperation was sought with embedded-application designers and DSP manufacturers.

The Embedded C specification extends the C language to support freestanding embedded processors in exploiting the multiple address space functionality, user-defined named address spaces, and direct access to processor and I/O registers. These features are common for the small, embedded processors used in most consumer products. The features introduced by Embedded C are fixed-point and saturated arithmetic, segmented memory spaces, and hardware I/O addressing. The description we present here addresses the extensions from a language-design perspective, as opposed to the programmer or processor architecture perspective.

MULTIPLE ADDRESS SPACES

Embedded C supports the multiple address spaces found in most embedded systems. It provides a formal mechanism for C applications to directly access (or map onto) those individual processor instructions that are designed for optimal memory access. Named address spaces use a single, simple approach to grouping memory locations into functional groups to support MAC buffers in DSP applications, physical separate memory spaces, direct access to processor registers, and user-defined address spaces.

The Embedded C extension supports defining both the natural multiple address space built into a processor's architecture and the

application-specific address space that can help define the solution to a problem.

Embedded C uses address space qualifiers to identify specific memory spaces in variable declarations. There are no predefined keywords for this, as the actual memory segmentation is left to the implementation. As an example, assume that **X** and **Y** are memory qualifiers. The definition:

```
X int a[25] ;
```

Means that **a** is an array of 25 integers, which is located in the **X** memory. Similarly (but less common):

```
X int * Y p ;
```

Means that the pointer **p** is stored in the **Y** memory. This pointer points to integer data that is located in the **X** memory. If no memory qualifiers are used, the data is stored into unqualified memory.

For proper integration with the C language, a memory structure is specified, where the unqualified memory encompasses all other memories. All unqualified pointers are pointers into this unqualified memory. The unqualified memory abstraction is needed to keep the compatibility of the **void *** type, the **NULL** pointer, and to avoid duplication of all library code that accesses memory through pointers that are passed as parameters.

4.1.2.2 NAMED REGISTERS

Embedded C allows direct access to processor registers that are not addressable in any of the machine's address spaces. The processor registers are defined by the compiler-specific, named-register, storage class for each supported processor. The processor registers are declared and used like conventional C variables (in many cases volatile variables). Developers using Embedded C can now develop their applications, including direct access to the condition code register and other processor-specific status flags, in a high-level language, instead of inline assembly code.

Named address spaces and full processor access reduces application dependency on assembly code and shifts the responsibility for computing data types, array and structure offsets, and all those things that C compilers routinely and easily do from developers to compilers.

I/O HARDWARE ADDRESSING

The motivation to include primitives for I/O hardware addressing in Embedded C is to improve the portability of device-driver code. In principle, a hardware device driver should only be concerned with the device itself. The driver operates on the device through device registers, which are device specific. However, the method to access these registers can be very different on different systems, even though it is the same device that is connected. The I/O hardware access primitives aim to create a layer that abstracts the system-specific access method from the device that is accessed. The ultimate goal is to allow source-code portability of device drivers between different systems. In the design of the I/O hardware-addressing interface, three requirements needed to be fulfilled:

1. The device-driver source code must be portable.
2. The interface must not prevent implementations from producing machine code that is as efficient as other methods.
3. The design should permit encapsulation of the system-dependent access method.

The design is based on a small collection of functions that are specified in the `<iohw.h>` include file. These interfaces are divided into two groups; one group provides access to the device, and the second group maintains the access method abstraction itself.

To access the device, the following functions are defined by Embedded C:

```
unsigned int iord( ioreg_designator );  
void iowr( ioreg_designator, unsigned int value );  
void ioor( ioreg_designator, unsigned int value );  
void ioand( ioreg_designator, unsigned int value );  
void ioxor( ioreg_designator, unsigned int value );
```

These interfaces provide read/write access to device registers, as well as typical methods for setting/resetting individual bits. Variants of these functions are defined (with **buf** appended to the names) to access arrays of registers. Variants are also defined (with **l** appended) to operate with **long** values.

All of these interfaces take an I/O register designator **ioreg_designator** as one of the arguments. These register designators are an abstraction of the real registers provided by the system implementation and hide the access method from the driver source code. Three functions are defined for managing the I/O register designators. Although these are abstract entities for the device driver, the driver does have the obligation to initialize and release the access methods. These functions do not access or initialize the device itself because that is the task of the driver. They allow, for example, the operating system to provide a memory mapping of the device in the user address space.

```
void iogroup_acquire( iogrp_designator );  
void iogroup_release( iogrp_designator );  
void iogroup_map( iogrp_designator, iogrp_designator );
```

The **iogrp_designator** specifies a logical group of I/O register designators; typically this will be all the registers of one device. Like the I/O register designator, the I/O group designator is an identifier or macro that is provided by the system implementation. The map variant allows cloning of an access method when one device driver is to be used to access multiple identical devices.

4.1.2.3 EMBEDDED C PORTABILITY

By design, a number of properties in Embedded C are left implementation defined. This implies that the portability of Embedded C programs is not always guaranteed. Embedded C provides access to the

performance features of DSPs. As not all processors are equal, not all Embedded C implementations can be equal. For example, suppose an application requires 24-bit fixed-point arithmetic and an Embedded C implementation provides only 16 bits because that is the native size of the processor. When the algorithm is expressed in Embedded C, it will not produce outputs of the right precision.

In such a case, there is a mismatch between the requirements of the application and the capabilities of the processor. Under no circumstances, including the use of assembly, will the algorithm run efficiently on such a processor. Embedded C cannot overcome such discrepancies. Yet, Embedded C provides a great improvement in the portability and software engineering of embedded applications. Despite many differences between performance-specific processors, there is a remarkable similarity in the special-purpose features that they provide to speed up applications.

Writing C code with the low-level processor-specific support may at first appear to have many of the portability problems usually associated with assembly code. In the limited experience with porting applications that use Embedded C extensions, an automotive engine controller application (about 8000 lines of source) was ported from the eTPU, a 24-bit special-purpose processor, to a general-purpose 8-bit Freescale 68S08 with about a screen full of definitions put into a single header file. The porting process was much easier than expected. For example, variables that had been implemented on the processor registers were ported to unqualified memory in the general-purpose microprocessor by changing the definitions in the header definition and without any actual code modifications. The exercise was to identify the

porting issues and it is clear that the performance of the special-purpose processor is significantly higher than the general-purpose target.

4.1.3 PYTHON

Python is a widely used high-level programming language for general-purpose programming, created by Guido van Rossum and first released in 1991. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++ or Java. The language provides constructs intended to enable writing clear programs on both a small and large scale.

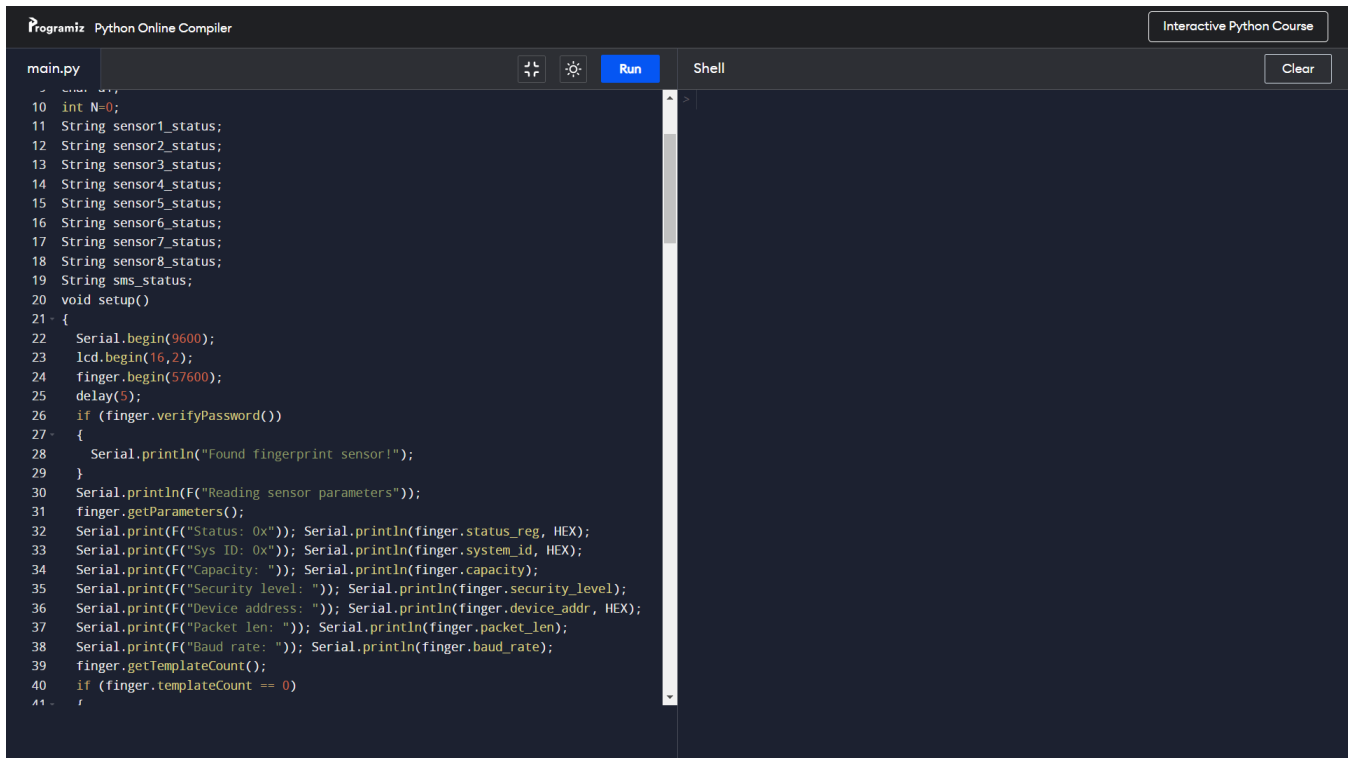
Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard library.

Python interpreters are available for many operating systems, allowing Python code to run on a wide variety of systems. C Python, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. C Python is managed by the non-profit Python Software Foundation.

Software Description

- We use the python programming for face recognition and identification purposes.
- For now, we have uploaded the sample data for face detection.
- The sample data are:
 - Signal violation
 - Theft Identification
 - No data (no-crime performed)
- To access the IOT data's we use an existing URL known as IOT beginner.com using The API data.

4.1.3.2 Code Snippet



The screenshot shows the Programiz Python Online Compiler interface. The left pane contains a Python script named `main.py` with the following code:

```
10 int N=0;
11 String sensor1_status;
12 String sensor2_status;
13 String sensor3_status;
14 String sensor4_status;
15 String sensor5_status;
16 String sensor6_status;
17 String sensor7_status;
18 String sensor8_status;
19 String sms_status;
20 void setup()
21 {
22   Serial.begin(9600);
23   lcd.begin(16,2);
24   finger.begin(57600);
25   delay(5);
26   if (finger.verifyPassword())
27   {
28     Serial.println("Found fingerprint sensor!");
29   }
30   Serial.println(F("Reading sensor parameters"));
31   finger.getParameters();
32   Serial.print(F("Status: 0x")); Serial.println(finger.status_reg, HEX);
33   Serial.print(F("Sys ID: 0x")); Serial.println(finger.system_id, HEX);
34   Serial.print(F("Capacity: ")); Serial.println(finger.capacity);
35   Serial.print(F("Security level: ")); Serial.println(finger.security_level);
36   Serial.print(F("Device address: ")); Serial.println(finger.device_addr, HEX);
37   Serial.print(F("Packet len: ")); Serial.println(finger.packet_len);
38   Serial.print(F("Baud rate: ")); Serial.println(finger.baud_rate);
39   finger.getTemplateCount();
40   if (finger.templateCount == 0)
41   {
```

The right pane is labeled "Shell" and contains a "Clear" button. The top of the interface includes the "Programiz Python Online Compiler" logo and an "Interactive Python Course" button.

Figure 3.14 Code Snippet

4.2 System Prototype

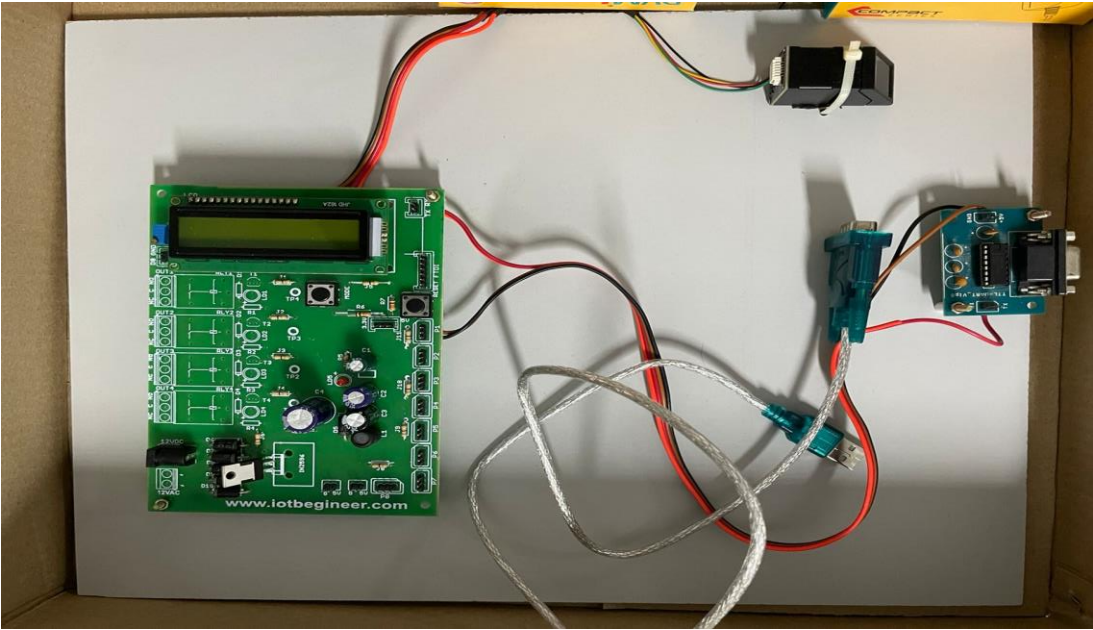


Figure 3.15 System Prototype

4.3 SOFTWARE RESULT

IoT - Real Time Sensor Values

iotbeginner.com/sensors

IoT

- Dashboard
- Title Update
- View IOT Data
- No. of sensors
- No. of loads
- Pin Selection
- Code Download
- Mobile Number Update
- Load Control
- Location Details
- Reset Sensor Data
- Download Sensor Data

Real Time Sensor Values

Filter By Date: 04-04-2023 Find

Show 10 entries

#	Sensor 1	Sensor 2	Date & Time	Action
1	criminal activities found	signal violation found	2023-04-04 23:01:41	
2	criminal activities found	signal violation found	2023-04-04 23:01:35	
3	criminal activities found	signal violation found	2023-04-04 23:01:26	

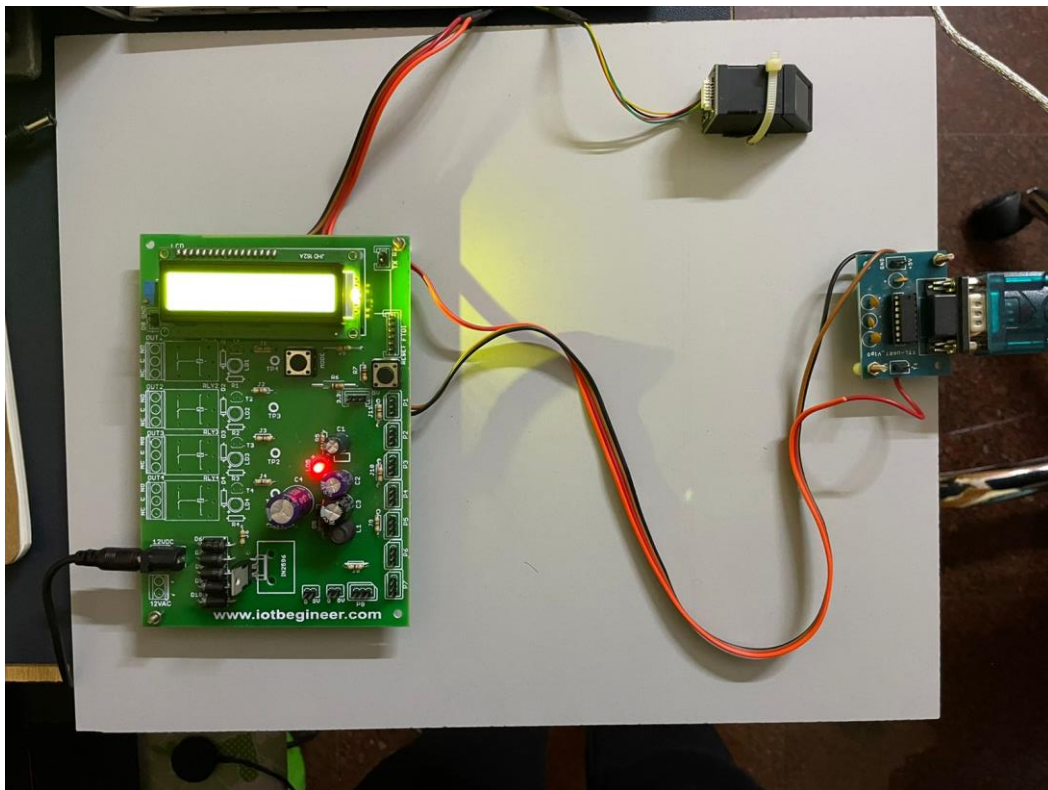
Showing 1 to 3 of 3 entries

Previous 1 Next

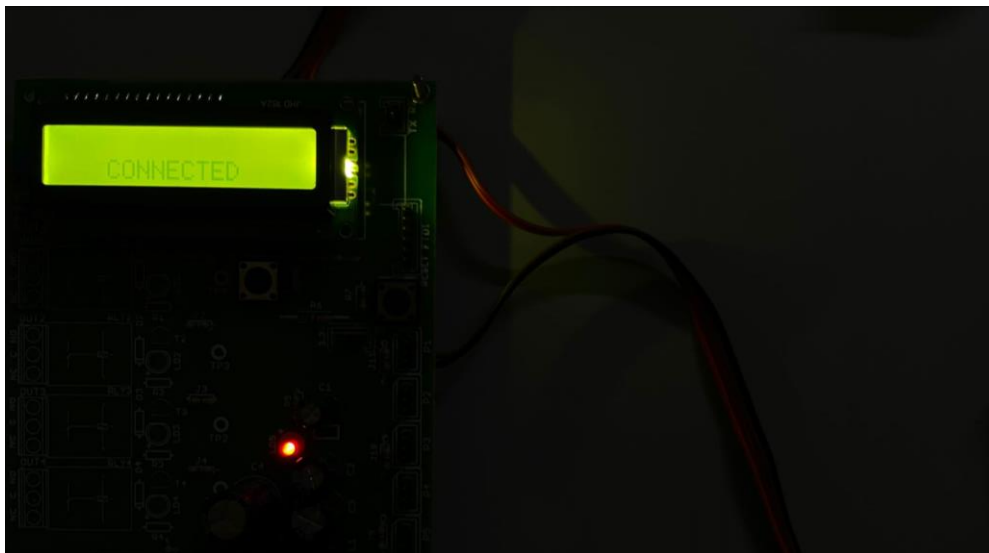
Chat

ENG IN 23:03 04-04-2023

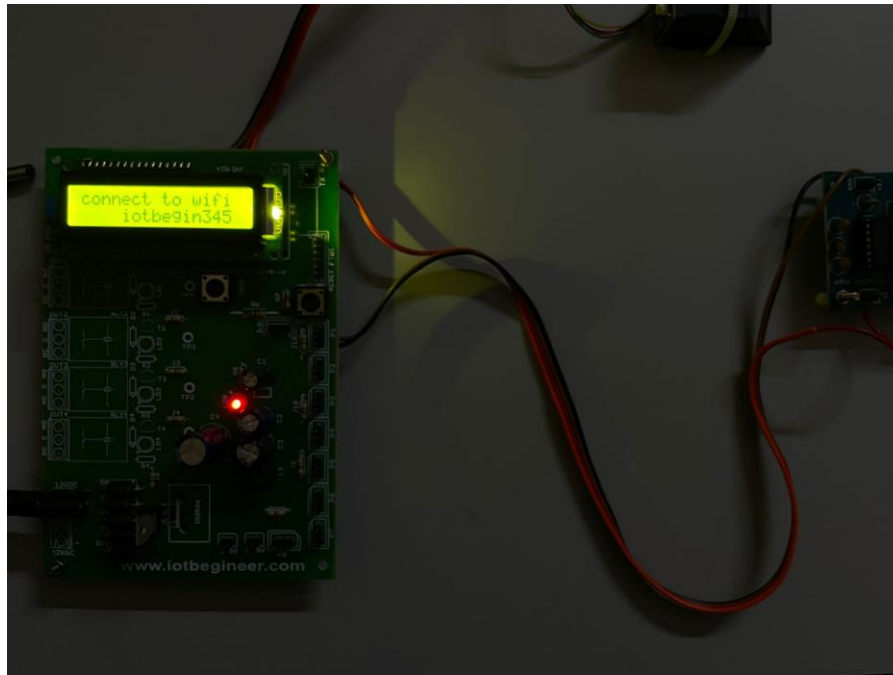
Final IOT database



Hardware Model



LED Display



Connecting to Wifi network



Face recognition of the suspect

CHAPTER 5

CONCLUSION

In the modern world, the use of computers and mobile phones is becoming rampant. As a result, the crime recording system needs to embrace the new technologies. This application will present a simple, convenient, cost-effective and efficient online crime recording system with a sensitive and intelligible web interface, thereby it reduces the amount of manual data entry. It is software which helps the policemen to work with the crimes and criminals easily. Additionally, it notifies the registered policemen about the release of criminals. This crime management system is a solution to all the problems related to the crime reports, criminal details and their crimes.

REFERENCES:

- [1]. Oludeli Awodeli, Onulri Ernest. E, "A Real-Time Records Management System for National Security Agencies Department of Computer Science, Babcock University, Ilishan-Remo, Ogun State, Nigeria, Vol.3, Issued 12, May 2015.
- [2]. Sourav Bhowmick, "Criminal Report Management System", Department of Computer Science and Engineering, ADMAS Institute of Technology, 2013.
- [3]. Mohammad Shahnawaz, "Crime Reporting and Crime Updates", 3rd International Conference on System Modeling in Research Trends (SMART) College of Computer Science and Information Technology (CCSIT), Teerthanker Mahaveer University, Moradabad, 2014.
- [4]. Srinidhi Eragam Reddy, Ramya Sahiti Amathi and Priyanka Vakkalagadda "Crime Reporting Interface Design using Mobile Technology", 2nd February, 2015.
- [5]. R. G. Jimoh, K. T. Ojulari, and O.A. Enikuomelin, "A Scalable Online Crime Reporting System", Department of Computer Science, University of Ilorin, Nigeria, Vol.7.No.1, January, 2014.
- [6] Highlights of 2009 Motor Vehicle crashes, Trunc Safety Facts, Research Notes, NHTSA (National Highway traffic Safety Administration) [Online], Accessed on 16 October 2011.
- [7] N. Virtanen, A Schirokoff and I. Luom, "Impacts of an automatic emergency call system on accident consequences," 18th ICTCT, Workshop Transport telemetric and safety, 2005.
- [8] S. M. Tang and H. I. Gao, "Trunc-incident detection-algorithm based on

nonparametric regression," IEEE Transactions on Intelligent Transportation Systems, 2005.

[9] G. Rose, "Mobile Phones as Traffic Probes: Practices, Prospects and Issues," Transport Reviews, 2006.

[10] P. Mohan, Y. N. Padmanabhan, and R. Ramjee, "Nericell: rich monitoring of road and traffic conditions using mobile smartphones," in Proceedings of the 6th ACM conference on Embedded network sensor systems, 2008.

[11] C. Thompson, I. White, B. Dougherty, A. Albright and D. C. Schmidt, "Using Smartphones to Detect Car Accidents and Provide Situational Awareness to Emergency Responders," in Proceedings of 3'd Mobile Wireless Middleware, Operating Systems, and Applications Conference,

[12] J. Yoon, B. Noble and M. Liu, "Surface street traffic estimation," in Proceedings of 5th International Conference on Mobile Systems, Applications, and Services, 2007.

[13] Md. Syedul Amin, Jubayer Jalil, M. B. I. Reaz, "Accident Detection and Reporting System using GPS, GPRS and GSM Technology," in Proceedings of International Conference on Informatics, Electronics & Vision, 2012.

[14] Chalerm Pol Saiprasert and Wasan Pattara-Atikom, "Smartphone Enabled Dangerous Driving Report System," in Proceedings of 46th Hawaii International Conference on System Sciences, 2013.