SMART STRETCHER AND INTEGRATED MEDICAL INTELLIGENCE SYSTEMS FOR UNCONSCIOUS PERSON

A PROJECT REPORT

Submitted by

SHANMUGA RAJ R **REG NO: 113016106089**

HARIHARAN B **REG NO: 113016106029**

VIGNESH KUMAR M **REG NO: 113016106106**

In partial fulfilment for the award of the degree

Of

BACHELOR OF ENGINEERING

In

ELECTRONICS AND COMMUNICATION ENGINEERING

VEL TECH HIGH TECH Dr. RANGARAJAN Dr. SAKUNTHALA ENGINEERING COLLEGE

AVADI, CHENNAI - 600 062.

ANNA UNIVERSITY : CHENNAI 600 025

APRIL 2020

CERTIFICATE OF EVALUATION

COLLEGE : VEL TECH HIGH TECH Dr. RANGARAJAN & Dr. SAKUNTHALA ENGINEERING COLLEGE, AVADI, CHENNAI- 600 062.

BRANCH : ELECTRONICS & COMMUNICATION ENGINEERING

YEAR/SEM : IV / VIII

NAME OF THE STUDENT	UNIVERSITY NO	TITLE OF THE PROJECT	NAME OF THE SUPERVISOR
SHANMUGA RAJ R	11301610089	SMART STRETCHER AND INTEGRATED MEDICAL INTELLIGENCE SYSTEMS FOR UNCONSCIOUS PERSON	Mr. N. PRABHAKARAN, B.E., M.E.,
HARIHARAN B	113016106029		
VIGNESH KUMAR M	113016106106		

The report of the project work submitted by the above students in the partial fulfilment for the award of Bachelor of Engineering Degree in Electronics and Communication Engineering of Anna University was evaluated and confirmed to be the reports of the work done by the above students and then evaluated.

---------- ----------

INTERNAL EXAMINER **EXTERNAL EXAMINER**

DATE:

ACKNOWLEDGEMENT

We wish to express our sincere thanks and heartfelt gratitude to our founder and chairman **Dr. RANGARAJAN B.E(Elec)., B.E(Auto)., M.S(Auto)., D.Sc.,** and our vice chairman **Dr. SAKUNTHALA RANGARAJAN., MBBS.,** Vel Tech High Tech Dr.Rangarajan Dr.Sakunthala Engineering College, for their constant support and facilities provided by the institution.

We express our sincere thanks to our Principal **Dr. E KAMALANABAN B.E., M.E., Ph.D.,** and Head of the Department **Mrs. N DURAICHI B.E., M.E.,** for their constant encouragement and valuable assistance.

We take this opportunity to express our profound gratitude and deep regard to our project guide **Mr. N. PRABHAKARAN B.E., M.E.**, Department of Electronics and Communication Engineering for his guidance and constant encouragement throughout the course of the project.

We also thank our parents, various staff, friends and all the good hearts for rendering their valuable suggestions and ideas during the course of our project thereby making it successful.

ABSTRACT

The Internet of Things (IoT) creates opportunity for more direct integration of the physical world into computer – based system, resulting in efficiency improvement, economic benefits and reduce human exertions.

The Internet of Things (IoT) systems are typically controlled by event driven smart apps that take as input either sensed data, user inputs or other external triggers (from the internet) and command one or more actuators towards providing different form of automations. To save a life is auspicious as well as precious.

The idea here is to provide an intelligent smart health system using sensors and Arduino which are implemented in stretcher. It will sense the body condition and send the data to the hospital and also informs to nearby police station where accident occurs to avoid law issues through server.

If this system is implemented, many human lives can be saved by preparing intensive care unit in hospital, as their physical parameters are updated to hospital before their arrival to hospital. The main advantages of this system are that it is easy to install in all hospital ambulance, easy to decide treatment process prior before the patient's arrival to hospital and it's a time consumption.

Obviously, this system can be implemented in any stretcher, chair. The product has been tested with a small group of people and worked properly. It still requires a deep test before commercialized in practice.

TABLE OF CONTENTS:

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTERNET OF THINGS

1.1 INTRODUCTION:

The internet of things or IoT, is a system of interrelated computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

A thing in the internet of things can be a person with a heart monitor implant, a farm animal with a biochip transponder, an automobile that has built-in sensors to alert the driver when tire pressure is low or any other natural or man-made object that can be assigned an IP address and is able to transfer data over a network. Increasingly, organizations in a variety of industries are using IoT to operate more efficiently, better understand customers to deliver enhanced customer service, improve decision-making and increase the value of the business.

1.1.2 History of IoT

Kevin Ashton, co-founder of the Auto-ID Center at MIT, first mentioned the internet of things in a presentation he made to Procter & Gamble (P&G) in 1999. Wanting to bring radio frequency ID (RFID) to the attention of P&G's senior management, Ashton called his presentation "Internet of Things" to incorporate the cool new trend of 1999: the internet. MIT professor Neil Gershenfeld's book, *When Things Start to Think*, also appearing in 1999, didn't use the exact term but provided a clear vision of where IoT was headed.

IoT has evolved from the convergence of wireless technologies, microelectromechanical systems (MEMS), microservices and the internet. The convergence has helped tear down the silos between operational technology (OT)

and information technology (IT), enabling unstructured machine-generated data to be analyzed for insights to drive improvements. Although Ashton's was the first mention of the internet of things, the idea of connected devices has been around since the 1970s, under the monikers *embedded internet* and *pervasive computing*.

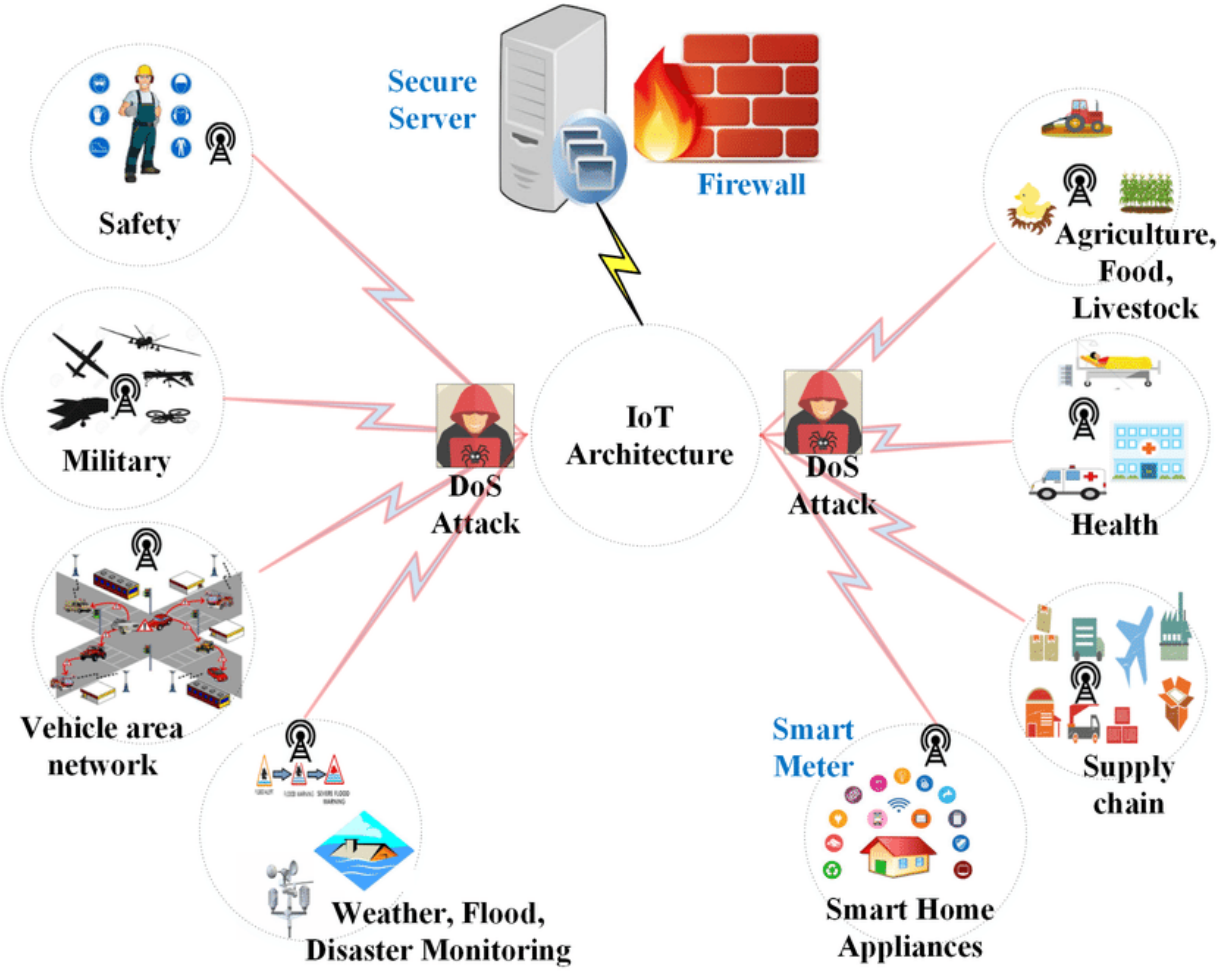


FIG 1.1 EXAMPLE OF AN IoT SYSTEM

The first internet appliance, for example, was a Coke machine at Carnegie Mellon University in the early 1980s. Using the web, programmers could check the status of the machine and determine whether there would be a cold drink awaiting them, should they decide to make the trip to the machine. IoT evolved

from machine-to-machine (M2M) communication, i.e., machines connecting to each other via a network without human interaction. M2M refers to connecting a device to the cloud, managing it and collecting data. Taking M2M to the next level, IoT is a sensor network of billions of smart devices that connect people, systems and other applications to collect and share data. As its foundation, M2M offers the connectivity that enables IoT.

The internet of things is also a natural extension of SCADA (supervisory control and data acquisition), a category of software application program for process control, the gathering of data in real time from remote locations to control equipment and conditions. SCADA systems include hardware and software components. The hardware gathers and feeds data into a computer that has SCADA software installed, where it is then processed and presented it in a timely manner. The concept of the IoT ecosystem, however, didn't really come into its own until the middle of 2010 when, in part, the government of China said it would make IoT a strategic priority in its five-year plan. An IoT ecosystem consists of web-enabled smart devices that use embedded processors, sensors and communication hardware to collect, send and act on data they acquire from their environments.

IoT devices share the sensor data they collect by connecting to an IoT gateway or other edge device where data is either sent to the cloud to be analyzed or analyzed locally. Sometimes, these devices communicate with other related devices and act on the information they get from one another.

The devices do most of the work without human intervention, although people can interact with the devices -- for instance, to set them up, give them instructions or access the data. The connectivity, networking and

communication protocols used with these web-enabled devices largely depend on the specific IoT applications deployed.

1.1.3 Applications:

There are numerous real-world applications of the internet of things, ranging from consumer IoT and enterprise IoT to manufacturing and industrial IoT. IoT applications span numerous verticals, including automotive, telco, energy and more. In the consumer segment, for example, smart homes that are equipped with smart thermostats, smart appliances and connected heating, lighting and electronic devices can be controlled remotely via computers, smartphones or other mobile devices.

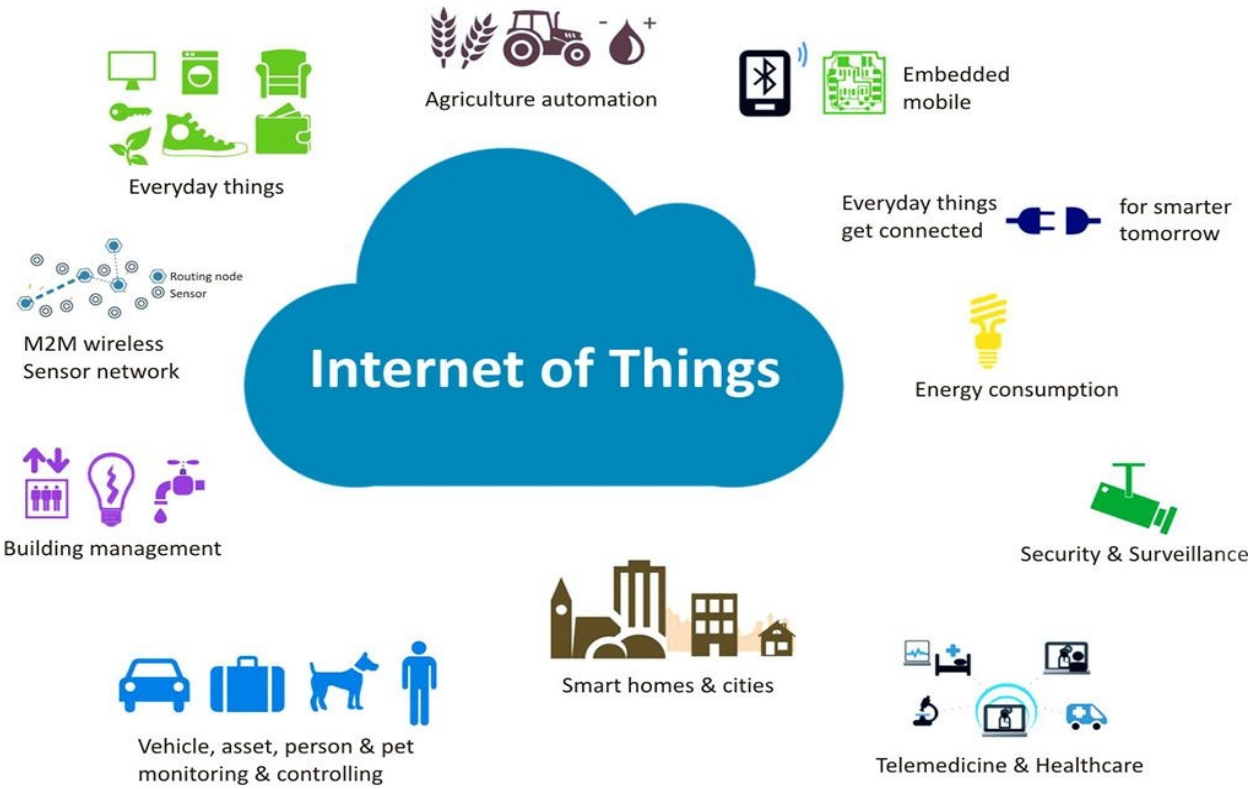


FIG 1.2 IOT APPLICATIONS

Wearable devices with sensors and software can collect and analyze user data, sending messages to other technologies about the users with the aim of making users' lives easier and more comfortable. Wearable devices are also used for public safety -- for example, improving first responders' response times during emergencies by providing optimized routes to a location or by tracking construction workers' or firefighters' vital signs at life-threatening sites.

In healthcare, IoT offers many benefits, including the ability to monitor patients more closely to use the data that's generated and analyze it. Hospitals often use IoT systems to complete tasks such as inventory management, for both pharmaceuticals and medical instruments.

In agriculture, IoT-based smart farming systems can help monitor, for instance, light, temperature, humidity and soil moisture of crop fields using connected sensors. IoT is also instrumental in automating irrigation systems.

In a smart city, IoT sensors and deployments, such as smart streetlights and smart meters, can help alleviate traffic, conserve energy, monitor and address environmental concerns, and improve sanitation.

1.1.4 Benefits of IoT

The internet of things offers a number of benefits, enabling them to:

- ❖ Monitor their overall business processes
- ❖ Improve the customer experience
- ❖ Enhance employee productivity
- ❖ Integrate and adapt business models
- ❖ Make better business decisions and
- ❖ Generate more revenue.

IoT encourages companies to rethink the ways they approach their businesses, industries and markets and gives them the tools to improve their business strategies.

1.1.5 IOT Adoption Barriers

The internet of things connects billions of devices to the internet and involves the use of billions of data points, all of which need to be secured. Due to its expanded attack surface, IoT security and IoT privacy are cited as major concerns. Because IoT devices are closely connected, all a hacker has to do is exploit one vulnerability to manipulate all the data, rendering it unusable. And manufacturers that don't update their devices regularly -- or at all -- leave them vulnerable to cybercriminals.

Additionally, connected devices often ask users to input their personal information, including names, ages, addresses, phone numbers and even social media accounts -- information that's invaluable to hackers.

However, hackers aren't the only threat to the internet of things; privacy is another major concern for IoT users. For instance, companies that make and distribute consumer IoT devices could use those devices to obtain and sell users' personal data.

CHAPTER 2

LITERATURE REVIEW

TITLE:

ASSESSMENT OF PRE-EJECTION PERIOD IN AMBULATORY SUBJECTS USING SEISMOCARDIOGRAM IN A WEARABLE BLOOD PRESSURE MONITOR.

ABSTRACT:

The ability to monitor arterial blood pressure continuously with unobtrusive body worn sensors may provide a unique and potentially valuable assessment of a patient's cardiovascular health. Pulse wave velocity (PWV) offers an attractive method to continuously monitoring blood pressure. However, PWV technologies based on timing measurements between the ECG and a distal PPG suffer from inaccuracies on mobile patients due to the confounding influence of pre-ejection period (PEP). In this paper, we presented a wearable, continuous blood pressure monitor (ViSi Mobile) that can measure and track changes in PEP. PEP is determined from precordial vibrations captured by an accelerometer coupled to the patient's sternum. The performance of the PEP measurements was evaluated on test subjects with postural change and patient activity. Results showed potential to improve cNIBP accuracy in active patients.

AUTHORS: Guanqun Zhang, Amber C. Cottrell, Isaac C. Henry and Devin B. McCombie.

PUBLISHED IN: 2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)

TITLE:

IOT BASED JOYSTICK CONTROLLED PIBOT USING SOCKET COMMUNICATION.

ABSTRACT:

The Internet of things (IoT) and automated frameworks is a key driver of robotic development technology. In current technology, robots are controlled using Smartphone. In our approach Driving Force GT Joystick is used to control the robot wirelessly and which provide precise manual control on a robotic vehicle. Compared to web-controlled or Smartphone controlled Pibot, joystick controlled Pibot is more effective since it provides speed control. In this work, Raspberry Pi is used as a base controller to control robotic car called Pibot to work as real time system and vehicle operations are controlled remotely at the ground station using USB joystick. The camera on the Pibot is used for live video streaming on a webpage using HTTP server for surveillance. The server-client socket communication (via strong Wi-Fi) is performed for controlling the vehicle in the remote station using python programming.

AUTHORS: Radhika K A, Raksha B , Pruthviraj U

PUBLISHED IN: 2018 IEEE Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)

TITLE:

ENERGY AWARE COMMUNICATION BETWEEN SMART IOT MONITORING DEVICES.

ABSTRACT:

One of the most important issue that must be addressed in designing communication protocols for wireless sensor networks (WSN) is how to save sensor node energy while meeting the needs of applications. Recent researches have led to new protocols specifically designed for sensor networks where energy awareness is an essential consideration. Internet of Things (IoT) is an innovative ICT paradigm where a number of intelligent devices connected to Internet are involved in sharing information and making collaborative decision. Integration of sensing and actuation systems, connected to the Internet, means integration of all forms of energy consuming devices such as power outlets, bulbs, air conditioner, etc. Sometimes the system can communicate with the utility supply company and this led to achieve a balance between power generation and energy usage or in general is likely to optimize energy consumption as a whole. In this paper some emerging trends and challenges are identified to enable energy-efficient communications in Internet of Things architectures and between smart devices. The way devices communicate is analyzed in order to reduce energy consumption and prolong system lifetime. Devices equipped with WiFi and RF interfaces are analyzed under different scenarios by setting different communication parameters, such as data size, in order to evaluate the best device configuration and the longest lifetime of devices.

AUTHORS: Floriano De Rango, Domenico Barletta

PUBLISHED IN: 2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS)

TITLE:

AN IOT - BASED REMOTE MONITORING SYSTEM FOR ELECTRICAL POWER CONSUMPTION VIA WEB-APPLICATION.

ABSTRACT:

Electricity is a fundamental need of the human being that is commonly used for domestic, industrial and agricultural purposes. In this sense, the waste of energy generates millionaire losses for the countries. Technological solutions such as Internet of Things allow connecting the physical with the digital world in order to optimize resources and in this particular case, manage and/or monitor the energy consumption. In addition, the advance of micro and nano electronics has allowed the development of communication modules such as the XBee that allows the implementation of a wireless sensor network quickly & efficiently with minimal energy consumption being widely used for monitoring and control tasks. The presented prototype takes advantage of the previously mentioned advantages by developing hardware and software solution. It allows remote monitoring of electricity consumption in a home through a scalable & modular platform using XBee technology and customized protocol for data communication between the four modules that make up the system. Results are presented and demonstrate the accuracy of prototype compared to readings with conventional electricity meter.

AUTHOR: Darwin Alulema, Mireya Zapata

PUBLISHED IN: 2018 International Conference on Information Systems and Computer Science (INCISCOS)

TITLE:

DESIGN OF SMART STRETCHERS AND VITAL SIGNS MONITORING SYSTEM FOR REDUCED-MOBILITY PATIENTS.

ABSTRACT:

Technology advances have been focused for the last few years in different areas. For medical care, these technological advantages could provide mechanisms to obtain and process data that determine important clinical aspects of the patients. The purpose of this study is to emphasis on the main features of a smart stretcher prototype controlled by voice commands, and vital signs monitoring system for disabled persons or reduced mobility patients. Raspberry Pi and Arduino are the selected devices for processing the data, which are integrated to the data collecting software in order to develop a global system. The key contribution of this prototype is to facilitate medical treatments to the patients, since this system is developed by using affordable technology and merging different functions to offer a pragmatic solution.

AUTHORS: Wilmer Calle, Manuel Eduardo Flores Morán

PUBLISHED IN: 2018 13th Iberian Conference on Information Systems and Technologies (CISTI)

CHAPTER 3

3.1 EXISTING METHOD :-

Based on the survey, till now, even hospital aide informs the live status of patient, there is no any smart system to inform patients live condition to nursing home through online. Also it is must to inform immediately about accident to relatives and police station to proceed legal activities. To overcome these issues, we are proposing an idea to implement a smart system in stretcher in ambulance itself. In existing, it is also noted that minimum two workers are required to push stretcher safely. The proposed system will help to overcome this problem also.

3.2 PROPOSED METHOD :-

The proposed system consists of two sections namely:

1. Stretcher section
2. Hospital section.

In stretcher section, there are many health monitoring sensors to monitor patient live health parameters that are implemented in stretcher. This information is updated to hospital via server for every second. Also it is necessary to inform about accident and patient personal details to nearby police station along with location. For this, a biometric sensor is used to get patient personal information. This information is updated to hospital and police station through Internet of Things. With this information, an alert will be messaged to patient's relatives by cop.

Second section is hospital section. After reaching hospital, it is must to admit the patient in intensive care unit as soon as possible. But there are many obstacles that may interfere between entrance and care unit. Till now, hospital workers move the stretcher manually. It may take some time delay due to applying human power. To solve this issue, a stretcher control mechanism is implemented in

this system which is controlled by microcontroller. By initiating this mechanism, stretcher moves automatically by stretcher moving mechanism with guidance of human. If the stretcher moves very fast or uncontrollable, then an emergency stop switch will be activated automatically to stop litter.

A sonic humidifier sensor is implemented to detect whether any person or other object interferes in stretcher path. If any interfere occurs, alerting system is activated to avoid such interferes.

3.3 BLOCK DIAGRAM:

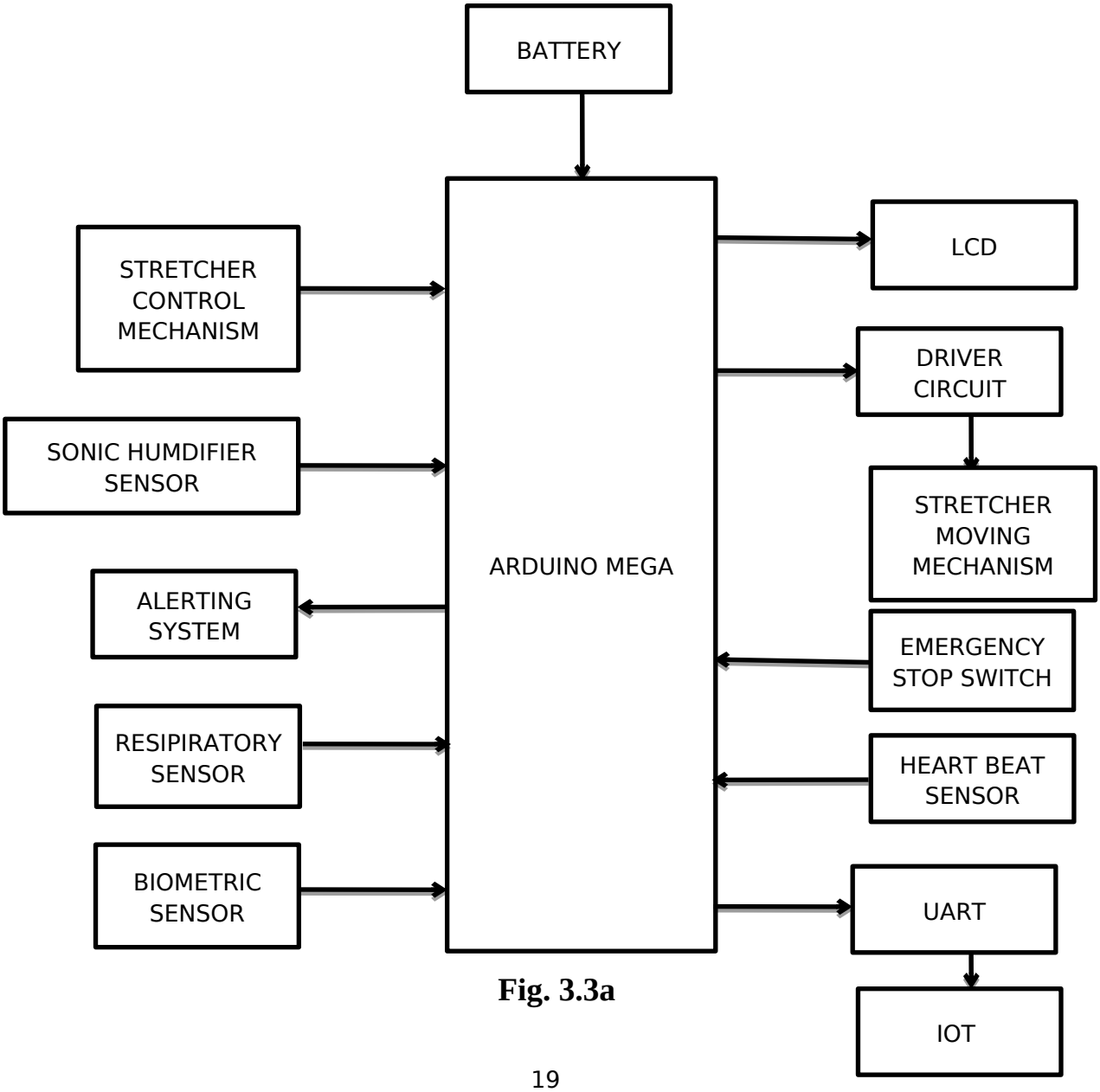


Fig. 3.3a

3.4 CIRCUIT DIAGRAM:

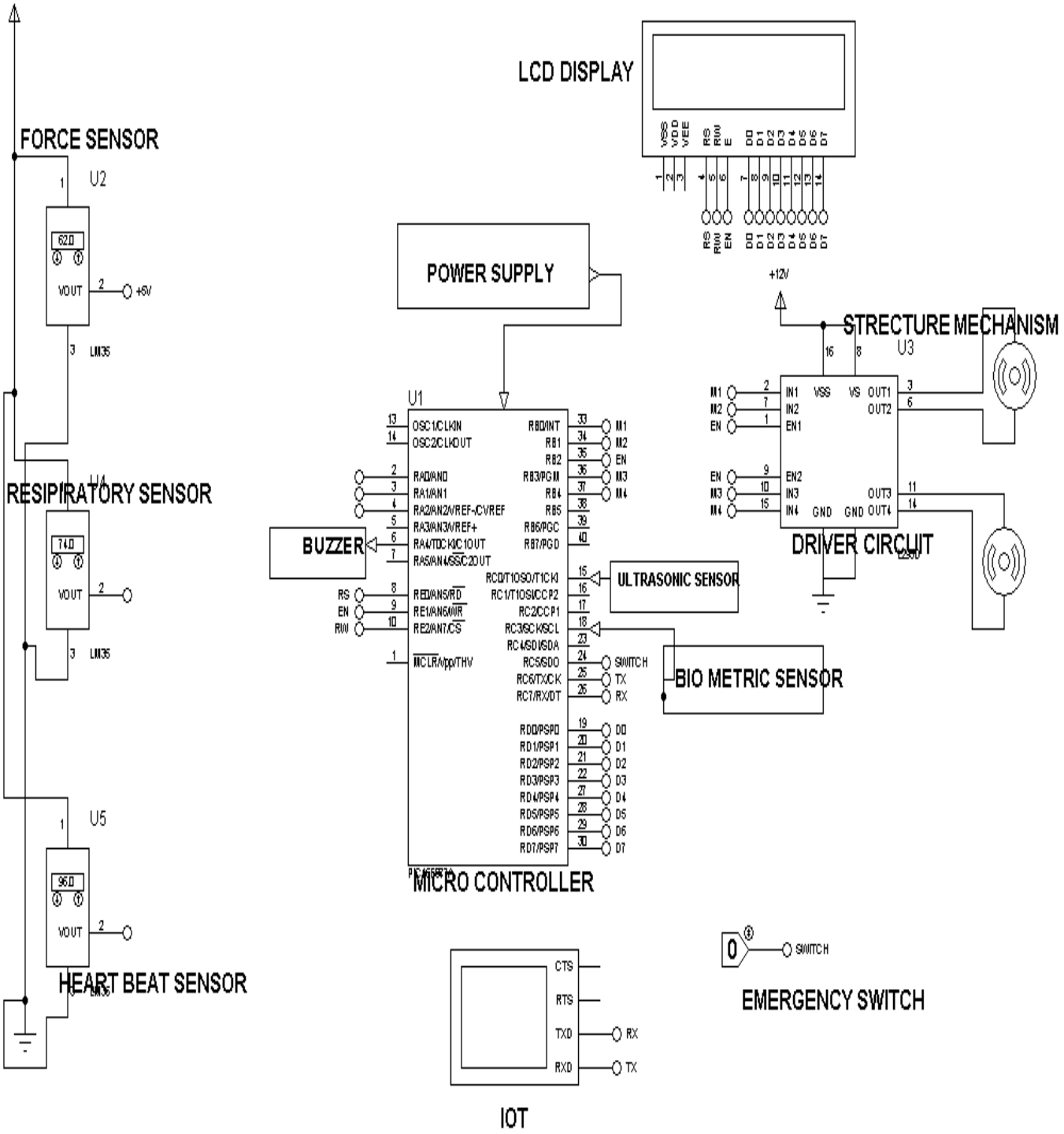


Fig. 3.4a

CHAPTER 4

HARDWARE DESCRIPTION

4.1 MICRO CONTROLLER: ARDUINO MEGA

4.1.1 GENERAL DESCRIPTION:

Mega2560- CORE is a small, complete and breadboard-friendly board base on the ATMega2560. Its design is based on the Arduino Mega2560 so, we can use it as a Arduino Mega2560 development board. In a different place, it lacks only a 6-foot download port and a reset switch. Reducing the hardware circuit that can we reduce the power consumption and the cost. Mega2560-CORE has a matching download line and the other one end of the download cable is a USB interface, so it is very convenient for use. 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.

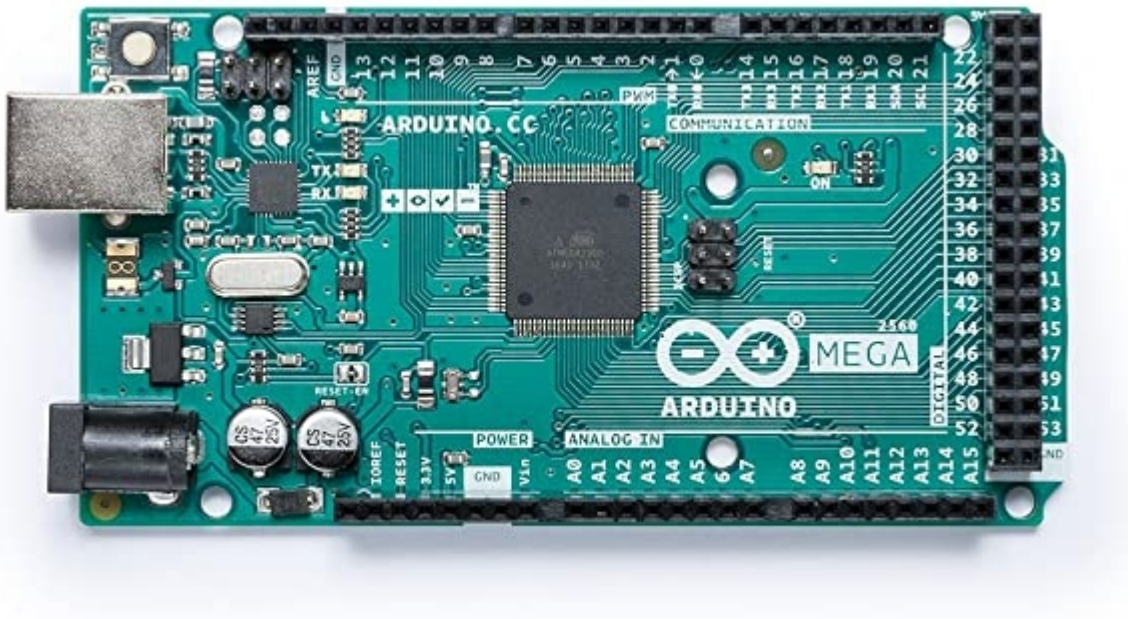


Fig. 4.1a

4.1.2 PRODUCT DESCRIPTION:

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

4.1.3 FEATURES:

- Operating Voltage: 5V
- Input Voltage (recommended): 7-12V
- Input Voltage (limits): 6-20V
- Digital I/O Pins: 54 (of which 14 provide PWM output)
- Analog Input Pins: 16
- DC Current per I/O Pin: 40 mA

4.1.4 APPLICATIONS:

- Multiple projects requires high speed operations
- Multiple DIY projects having larger code size
- Atmega32 based robot
- Real time biometrics

4.2 DRIVER CIRCUIT

4.2.1 PRODUCT DISCRIPTION:

The ULN2003 is a monolithic high voltage and high current Darlington transistor arrays. It consists of seven NPN Darlington pairs that feature

high-voltage outputs with common-cathode clamp diode for switching inductive loads. The collector-current rating of a single Darlington pair is 500mA. The Darlington pairs may be paralleled for higher current capability. Applications include relay drivers, hammer drivers, lamp drivers, display drivers (LED gas discharge), line drivers, and logic buffers.

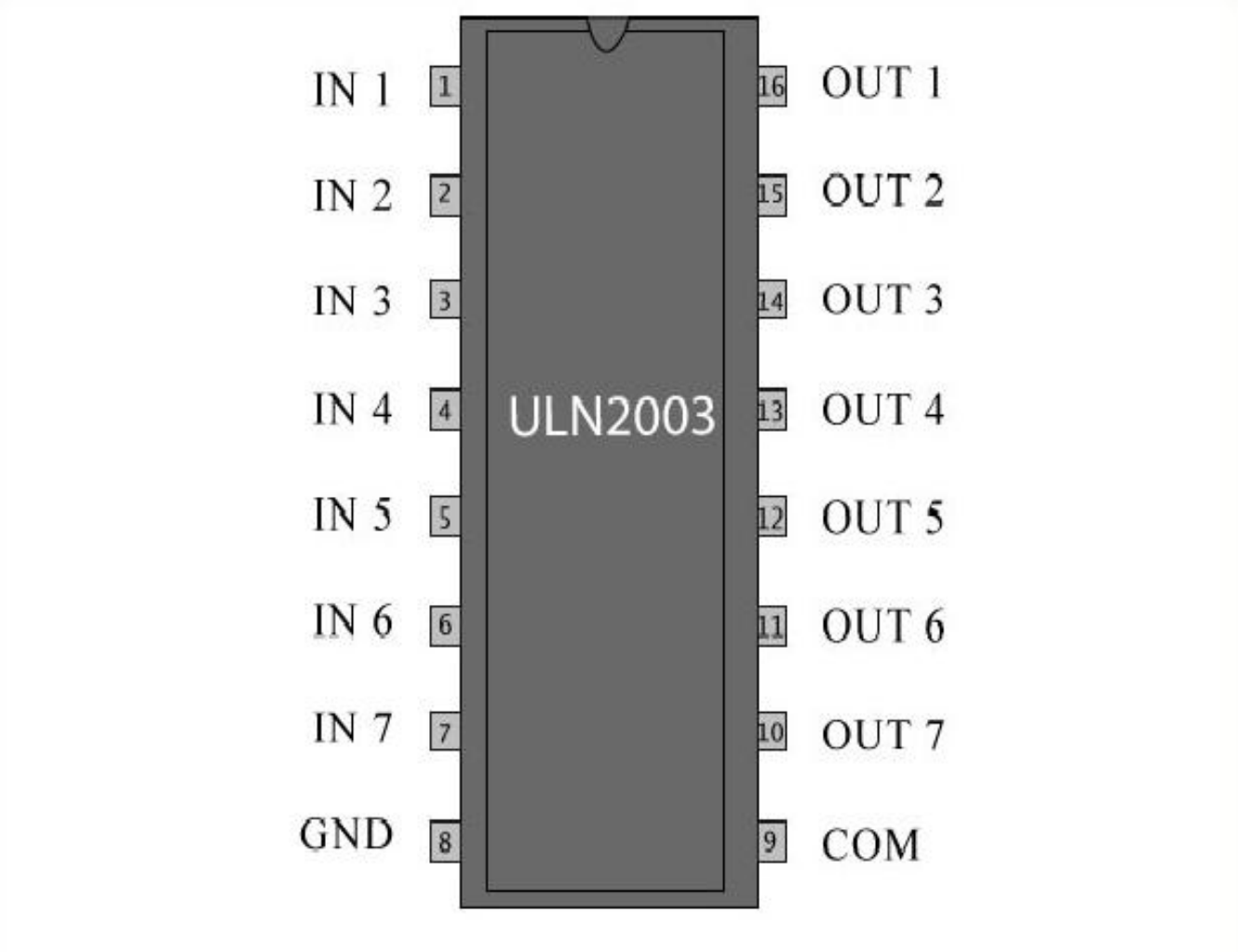


Figure: 4.2.a

4.2.2 FEATURES:

- 500mA rated collector current (Single output)
- High-voltage outputs: 50V
- Inputs compatible with various types of logic.

- Relay driver application.

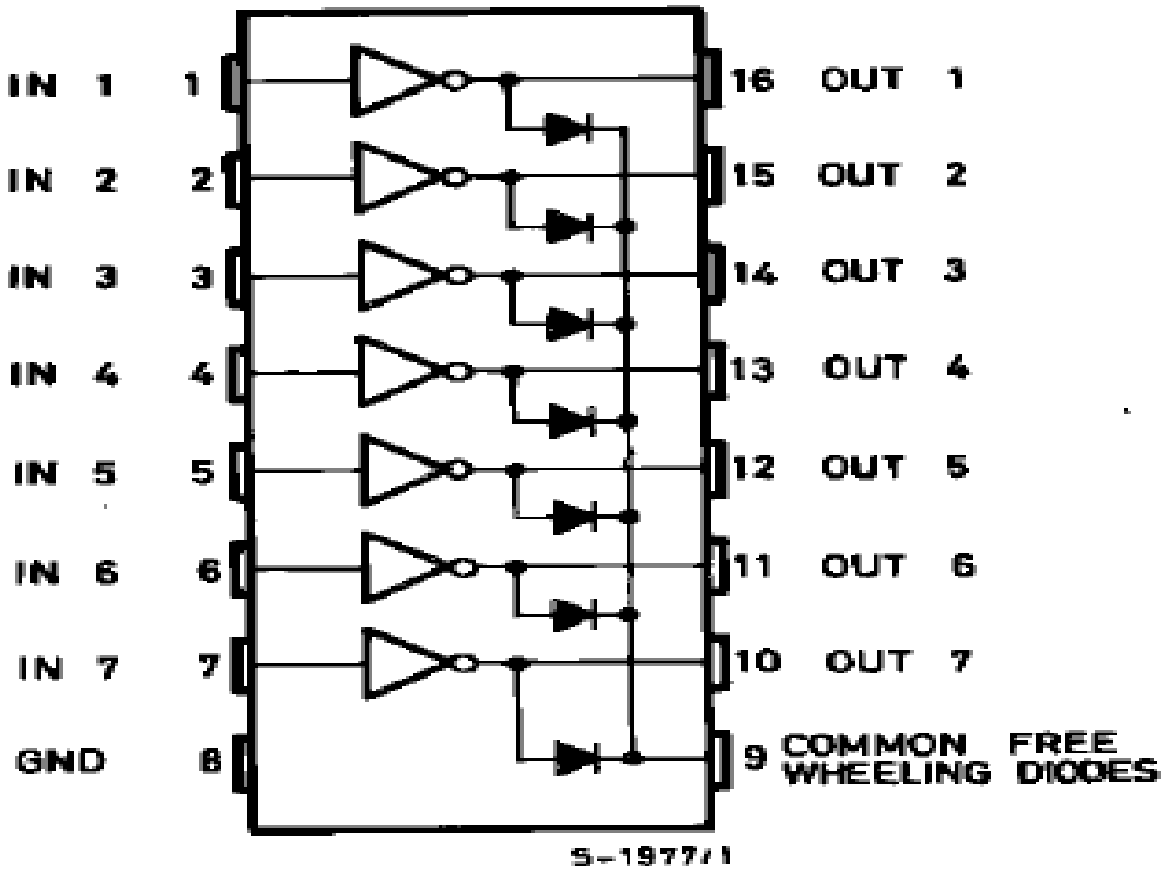


Figure: 4.2.b

The ULN2003 series input resistors selected for operation directly with 5 V TTL or CMOS. These devices will handle numerous interface needs particularly those beyond the capabilities of standard logic buffers. It has input resistors for operation directly from 6 V to 15 VCMOS or PMOS logic outputs. The ULN 2003 is the standard Darlington arrays. The outputs are capable of sinking 500mA and will withstand at least 50 V in the OFF state. Outputs may be paralleled for higher load current capability. The ULx2823A/LW and ULx2824A/ LW will withstand 95 V in the OFF state. These Darlington arrays are furnished in 18-pin dual in-line plastic packages or 18-lead small-outline plastic packages.

All devices are pinned with outputs opposite inputs to facilitate ease of circuit board layout. Prefix ‘ULN’ devices are rated for operation over the temperature range of -20°C to +85°C; prefix ‘ULQ’ devices are rated for operation to -40°C.

4.3 RESPIRATORY SENSOR:

Fig. 4.3a

4.3.1 GENERAL DESCRIPTION:

The Respiration Sensor is used to monitor abdominal or theoretical breathing, in biofeedback applications such as stress management and relaxation training. Besides measuring breathing frequency, this sensor also gives you an indication of the relative depth of breathing.

The Respiration Sensor for Nexus can be worn over clothing, although for best results we advise that there only be 1 or 2 layers of clothing between the sensor and the skin.

The Respiration Sensor is usually placed in the abdominal area, with the central part of the sensor just above the navel. The sensor should be placed tight enough to prevent loss of tension.

4.3.2 PRODUCT DESCRIPTION:

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.

The first step in this process is breathing in air, or inhaling. The taking in of air rich in oxygen into the body is called inhalation and giving out of air rich in carbon dioxide from the body is called exhalation.

The second step is gas exchange in the lungs where oxygen is diffused into the blood and the carbon dioxide diffuses out of the blood. The third process is cellular respiration, which produces the chemical energy that the cells in the body need, and carbon dioxide.

Finally, the carbon dioxide from cellular respiration is breathed out of body from the lungs.

4.3.3 FEATURES:

- Input voltage: 5v
- Output voltage: 5v
- Output: Analog
- Range: 30% – 65%
- Size (Approx.):132cm (52" Long)

4.3.4 APPLICATIONS:

- Medical purpose.
- Environmental Control System.
- Emergency response System.

4.4 LCD:

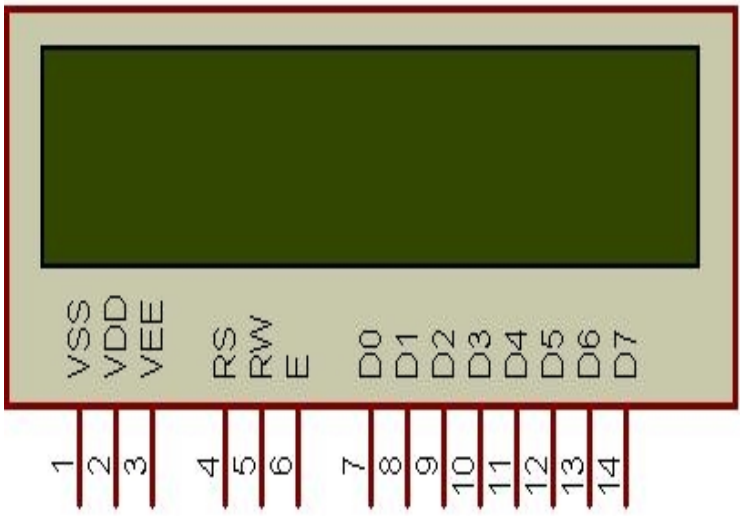


Fig3.4a LCD module

4.4.1 INTRODUCTION:

The most commonly used Character based LCDs are based on Hitachi's HD44780 controller or other which are compatible with HD44580.In this tutorial, we will discuss about character based LCDs, their interfacing with various microcontrollers, various interfaces (8-bit/4-bit), programming, special stuff and tricks you can do with these simple looking LCDs which can give a new look to your application

The most commonly used LCDs found in the market today are 1 Line, 2 Line or 4 Line LCDs which have only 1 controller and support at most of 80 characters, whereas LCDs supporting more than 80 characters make use of 2HD44780 controllers.

Most LCDs with 1 controller has14 Pins and LCDs with 2 controller has 16 Pins (two pins are extra in both for back-light LED connections).

PIN DESCRIPTION:

Pin No.	Name	Description
Pin no. 1	D7	Data bus line 7 (MSB)
Pin no. 2	D6	Data bus line 6
Pin no. 3	D5	Data bus line 5
Pin no. 4	D4	Data bus line 4
Pin no. 5	D3	Data bus line 3
Pin no. 6	D2	Data bus line 2
Pin no. 7	D1	Data bus line 1
Pin no. 8	D0	Data bus line 0 (LSB)
Pin no. 9	EN1	Enable signal for row 0 and 1 (1stcontroller)
Pin no. 10	R/W	0 = Write to LCD module 1 = Read from LCD module
Pin no. 11	RS	0 = Instruction input 1 = Data input
Pin no. 12	VEE	Contrast adjust
Pin no. 13	VSS	Power supply (GND)
Pin no. 14	VCC	Power supply (+5V)
Pin no. 15	EN2	Enable signal for row 2 and 3 (2ndcontroller)
Pin no. 16	NC	Not Connected

Table. 4.4.1a

No.	Instruction	Hex	Decimal

1	Function Set: 8-bit, 1 Line, 5x7 Dots	0x30	48
2	Function Set: 8-bit, 2 Line, 5x7 Dots	0x38	56
3	Function Set: 4-bit, 1 Line, 5x7 Dots	0x20	32
4	Function Set: 4-bit, 2 Line, 5x7 Dots	0x28	40
5	Entry Mode	0x06	6
6	Display off Cursor off (clearing display without clearing DDRAM content)	0x08	8
7	Display on Cursor on	0x0E	14
8	Display on Cursor off	0x0C	12
9	Display on Cursor blinking	0x0F	15
10	Shift entire display left	0x18	24
12	Shift entire display right	0x1C	30
13	Move cursor left by one character	0x10	16

14	Move cursor right by one character	0x14	20
15	Clear Display (also clear DDRAM content)	0x01	1
16	Set DDRAM address or cursor position on display	0x80+add*	128+add*
17	Set CGRAM address or set pointer to CGRAM location	0x40+add**	64+add**

Table 4.4.1b

4.5 HEART BEAT SENSOR:

Heart beat sensor is designed to give digital output of heat beat when a finger is placed on it. When the HEART BEAT detector is working, the beat LED flashes in unison with each heartbeat. This digital output can be connected to microcontroller directly to measure the Beats per Minute (BPM) rate.

It works on the principle of light modulation by blood flow through finger at each pulse.

Medical heart sensors are capable of monitoring vascular tissue through the tip of the finger or the ear lobe. It is often used for health purposes, especially when monitoring the body after physical training.

HEART BEAT is sensed by using a high intensity type LED and LDR. The finger is placed between the LED and LDR. As Sensor a photo diode or a photo transistor can be used. The skin may be illuminated with visible (red) using transmitted or reflected light for detection.

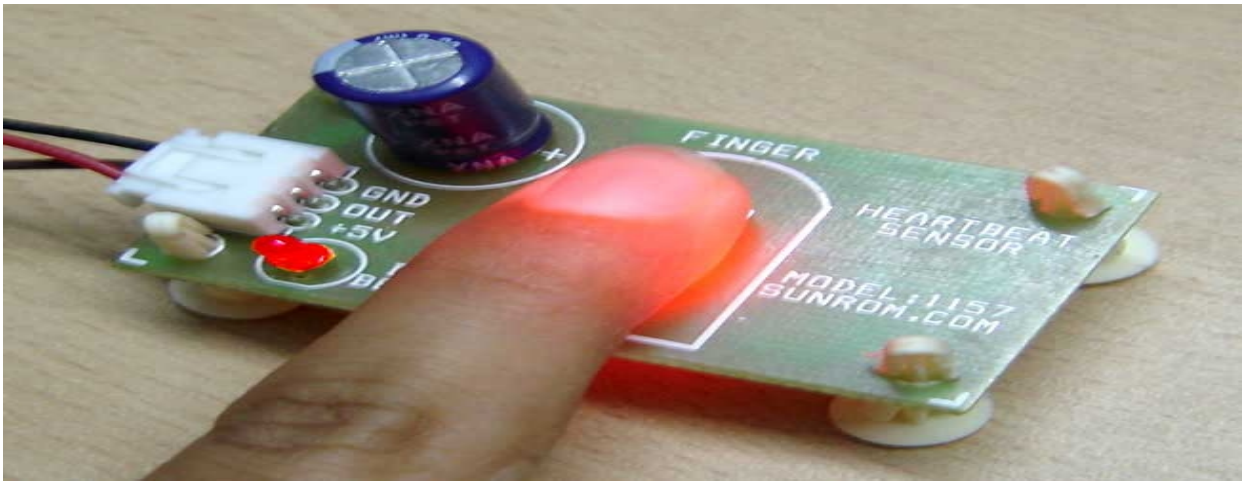


Fig. 4.5a

The very small changes in reflectivity or in transmittance caused by the varying blood content of human tissue are almost invisible. Various noise sources may produce disturbance signals with amplitudes equal or even higher than the amplitude of the pulse signal. Valid pulse measurement therefore requires extensive preprocessing of the raw signal. The new signal processing approach presented here combines analog and digital signal processing in a way that both parts can be kept simple but in combination are very effective in suppressing disturbance signals

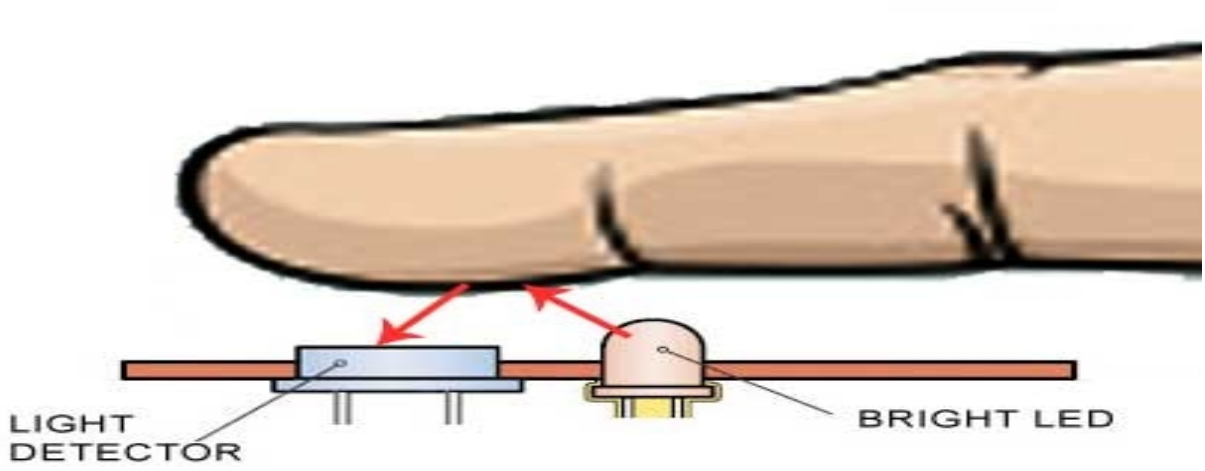


Fig 4.5b

The setup described here uses a red LED for transmitted light illumination and a LDR as detector. With only slight changes in the preamplifier

circuit the same hardware and software could be used with other illumination and detection concepts. The detectors photo current (AC Part) is converted to voltage and amplified by an operational amplifier (LM358).

4.5.1 FEATURES:

- Microcontroller based SMD design.
- Heat beat indication by LED.
- Instant output digital signal for directly connecting to microcontroller.
- Compact Size.
- Working Voltage +5V DC.

4.5.2 APPLICATIONS:

- Digital Heart Rate monitor.
- Patient Monitoring System.
- Bio-Feedback control of robotics and applications.

4.6 BATTERY:

A battery is an electrochemical cell (or enclosed and protected material) that can be charged electrically to provide a static potential for power or released electrical charge when needed. A battery generally consists of an anode, a cathode, and an electrolyte. A battery is a device that converts chemical energy directly to electrical energy. It consists of a number of voltaic cells; each voltaic cell consists of two half cells connected in series by a conductive electrolyte containing anions and cat ions.

One half-cell includes electrolyte and the electrode to which anions (negatively charged ions) migrate i.e., the anode or negative electrode; the other half-cell includes electrolyte and the electrode to which cat ions (positively charged ions) migrate, i.e., the cathode or positive electrode. In the red ox reaction that powers the battery, cat ions are reduced (electrons are added) at the cathode, while anions are oxidized (electrons are removed) at the

anode. The electrodes do not touch each other but are electrically connected by the electrolyte. Some cells use two half-cells with different electrolytes. A separator between half-cells allows ions to flow, but prevents mixing of the electrolytes. Common types of commercial batteries and some of their characteristics and

Battery Type	Characteristics	Typical Uses	Advantages
Sealed Lead Acid (SLA) battery	Can hold a charge for up to 3 years	Backup emergency power source	Inexpensive
Nickel-Cadmium (Ni-Cd) battery	Fast, even energy discharge	Appliances, audio and video equipment, toys; most popular batter	Relatively inexpensive; widely available
Nickel-Metal Hydride (Ni-MH) battery	Typical power capacity i1.2 V - 1200 to 1500 mAh; extended life 2300 mAh; 2.5 to 4 hours battery life	Portable computers; cellular phones; same as for Ni-Cd batteries	No memory effect; unused capacity remains usable
Lithium Ion (Li-Ion) battery	Stable and safe; highest energy capacity	Portable computers; cellular phones; same as for Ni-Cd batteries	Twice the charge capacity of Ni-Cd; slow self-discharge

advantages are summarized in the following table.

Table 4.6a

4.7 BUZZER:

Fig 3.7a

4.7.1 GENERAL DESCRIPTION

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or key stroke.

Buzzer is an integrated structure of electronic transducers, DC power supply, widely used in computers, printers, copiers, alarms, electronic toys, automotive electronic equipment, telephones, timers and other electronic products for sound devices.

Active buzzer 5V Rated power can be directly connected to a continuous sound, this section dedicated sensor expansion module and the board in combination, can complete a simple circuit design, to "plug and play."

4.7.2 PRODUCT DESCRIPTION

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

It generates consistent single tone sound just by applying D.C voltage. Using a suitably designed resonant system, this type can be used where large sound volumes are needed. At Future Electronics we stock many of the most common types categorized by Type, Sound Level, Frequency, Rated Voltage, Dimension and Packaging Type.

4.7.3 FEATURES

- Input supply: 5 VDC.
- Current consumption: 9.0 mA max.
- Oscillating frequency: 3.0 ±0.5 KHz.
- Sound Pressure Level: 85dB min.

4.7.4 APPLICATIONS

- Confirmation of user input. (ex: mouse click or keystroke)
- Electronic metronomes.
- Sporting events.
- Judging Panels.
- Annunciator panels.
-

4.8 BIOMETRIC SENSOR: FINGERPRINT READER WITH BOARD

4.8.1 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,

Fig4.8a

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.

4.8.2 PRODUCT DESCRIPTION

R305 fingerprint module is fingerprint sensor with TTL UART interface for direct connections to microcontroller UART or to PC through MAX232 / USB Serial 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.

4.8.3 FEATURES

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

4.8.4 APPLICATIONS

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

4.9 FORCE SENSOR

FORCE SENSOR

Fig. 4.9a

4.9.1 GENERAL DESCRIPTION

This is a force sensitive resistor with a round, 0.5" diameter, sensing area. This FSR will vary its resistance depending on how much pressure is being applied to the sensing area. The harder the force, the lower the resistance.

When no pressure is being applied to the FSR its resistance will be larger than 1MΩ. This FSR can sense applied force anywhere in the range of 100g-10kg. Interlink Electronics FSRTM 400 series is part of the single zone Force Sensing Resistor family.

Force Sensing Resistors, or FSRs, are robust polymer thick film (PTF) devices that exhibit a decrease in resistance with increase in force applied to the surface of the sensor.

4.9.2 PRODUCT DESCRIPTION

This force sensitivity is optimized for use in human touch control of electronic devices such as automotive electronics, medical systems, and in industrial and robotics applications.

The standard 400 sensor is a round sensor 7.62mm in diameter. Custom sensors can be manufactured in sizes ranging from 5mm to over 600mm. Female connector and short tail versions can also be ordered.

When external force is applied to the sensor, the resistive element is deformed against the substrate. Air from the spacer opening is pushed through the air vent in the tail, and the conductive material on the substrate comes into contact with parts of the active area. The more of the active area that touches the conductive element, the lower the resistance.

4.9.3 FEATURES

- Overall length: 2.375"
- Sensing diameter: 0.5"
- Cost effective.
- Ultra-thin; 0.35mm.
- Robust; up to 10M actuations.

4.9.4 APPLICATIONS

- Use force for UI feedback.
- Enhance tool safety.
- Detect presence, position, or motion.
- Relative change in force or applied load.
- Rate of change in force.

4.10 ULTRASONIC SENSOR

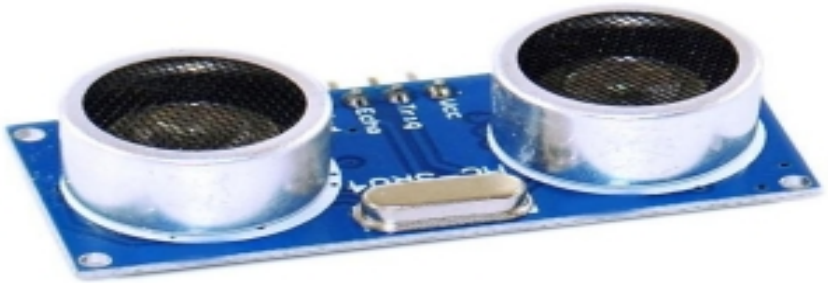


Fig 4.10a

4.10.1 GENERAL DESCRIPTION

Ultrasonic sensors emit ultrasonic pulses, and by measuring the time of ultrasonic pulse reaches the object and back to the transducer.

The sonic waves emitted by the transducer are reflected by an object and received back in the transducer.

After having emitted the sound waves, the ultrasonic sensor will switch to receive mode. The time elapsed between emitting and receiving is proportional to the distance of the object from the sensor.

4.10.2 PRODUCT DESCRIPTION

Ultrasonic transmitter emitted an ultrasonic wave in one direction and started timing when it launched. Ultrasonic spread in the air and would return immediately when it encountered obstacles on the way. At last the ultrasonic receiver would stop timing when it receives the reflected wave.

The distance of sensor from the target object is calculated. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package. It operation is not affected by sunlight or black material. The supply voltage to the sensor is 5VDC. The sensor has two pins namely trig and echo which is connected to the controller to give digital input.

4.10.3 FEATURES

- Working Voltage: 5VDC
- Quiescent Current: <2mA
- Working Current: 15mA
- Detecting Range: 2cm - 4.5m
- Trigger Input Pulse width: 10uS

4.10.4 APPLICATIONS

- Robot navigation
- Obstacle avoidance

CHAPTER 5

SOFTWARE DESCRIPTION

5. EMBEDDED C:

5.1 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 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.

5.2 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.

5.3 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.

5.4 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.

5.5 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:

- The device-drive source code must be portable.
- The interface must not prevent implementations from producing machine code that is as efficient as other methods.
- 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 **buff** 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.

5.6 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.

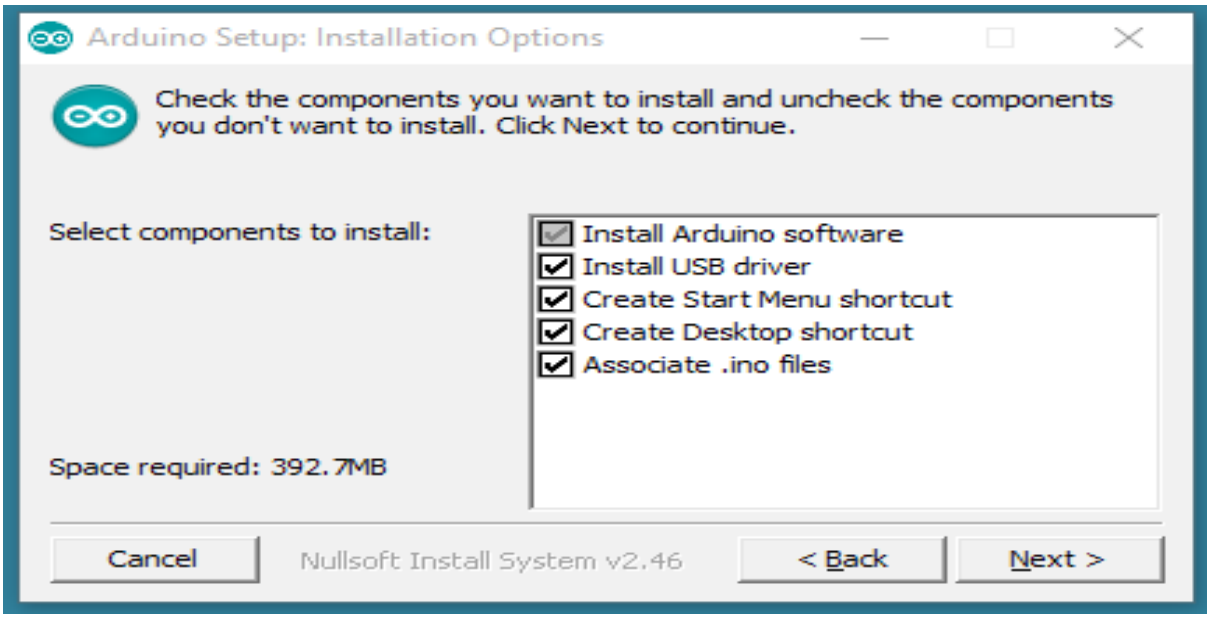
The porting process was much easier than expected. For example, variables that had been implemented on the processor registers were ported to unqualified memory in the general-purpose microprocessor by changing the definitions in the header definition and without any actual code modifications.

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.

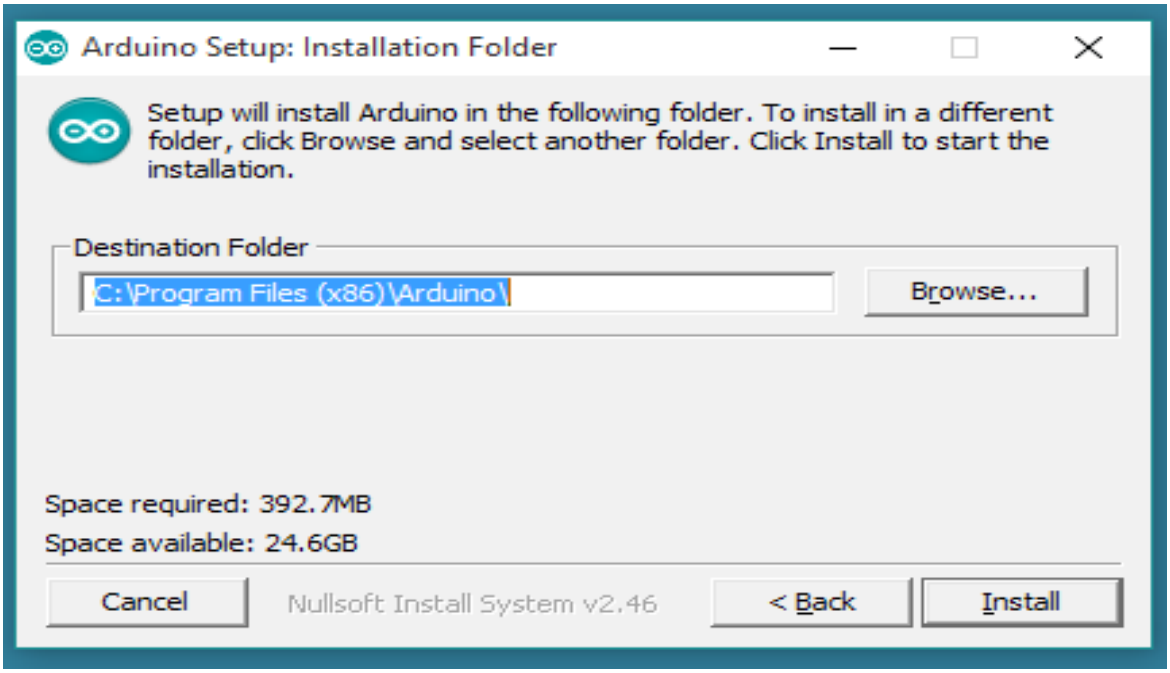
5.7 Arduino Software (IDE)

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.

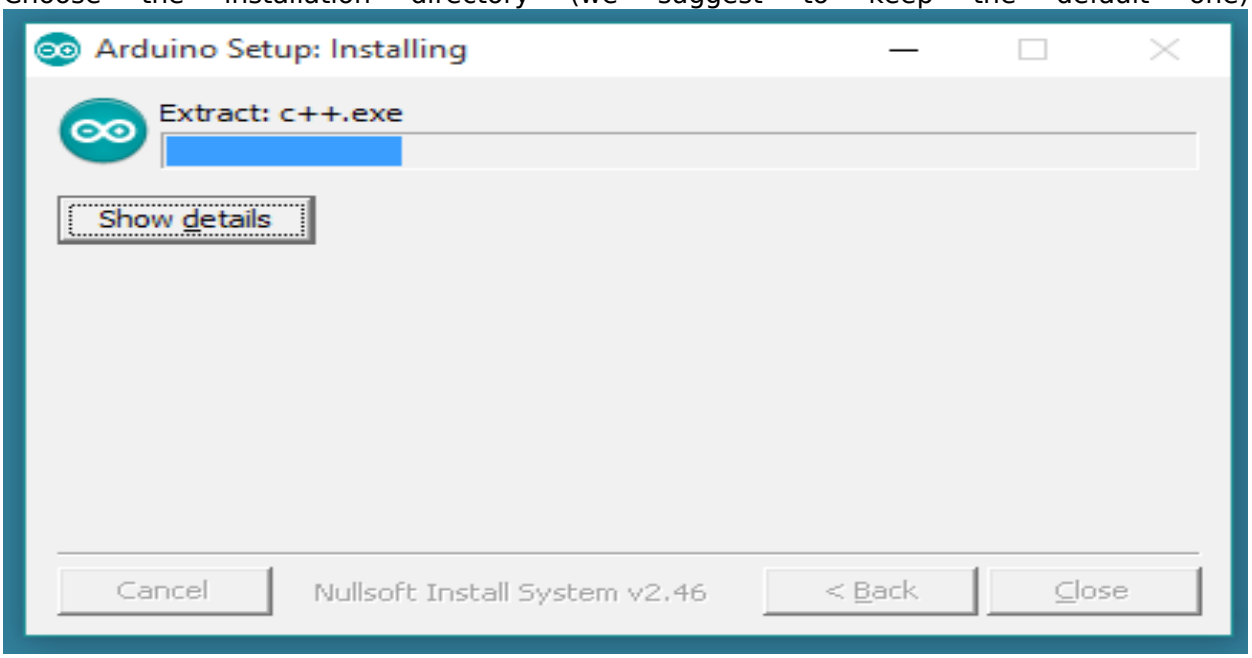


Choose the components to install :



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

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



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 Arduino UNO, but you will not 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.

5.7 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
- 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

5.8 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.

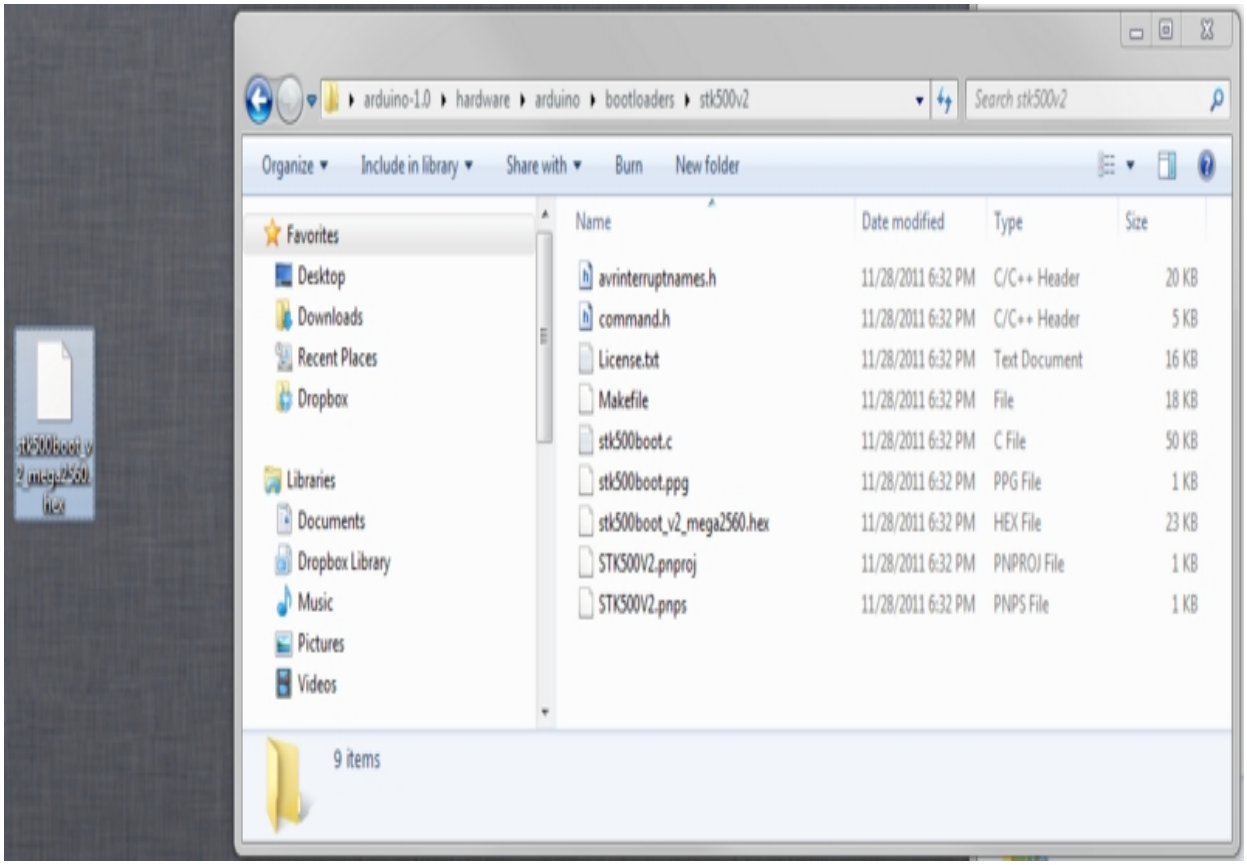
- Arduino Official Programming Language

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

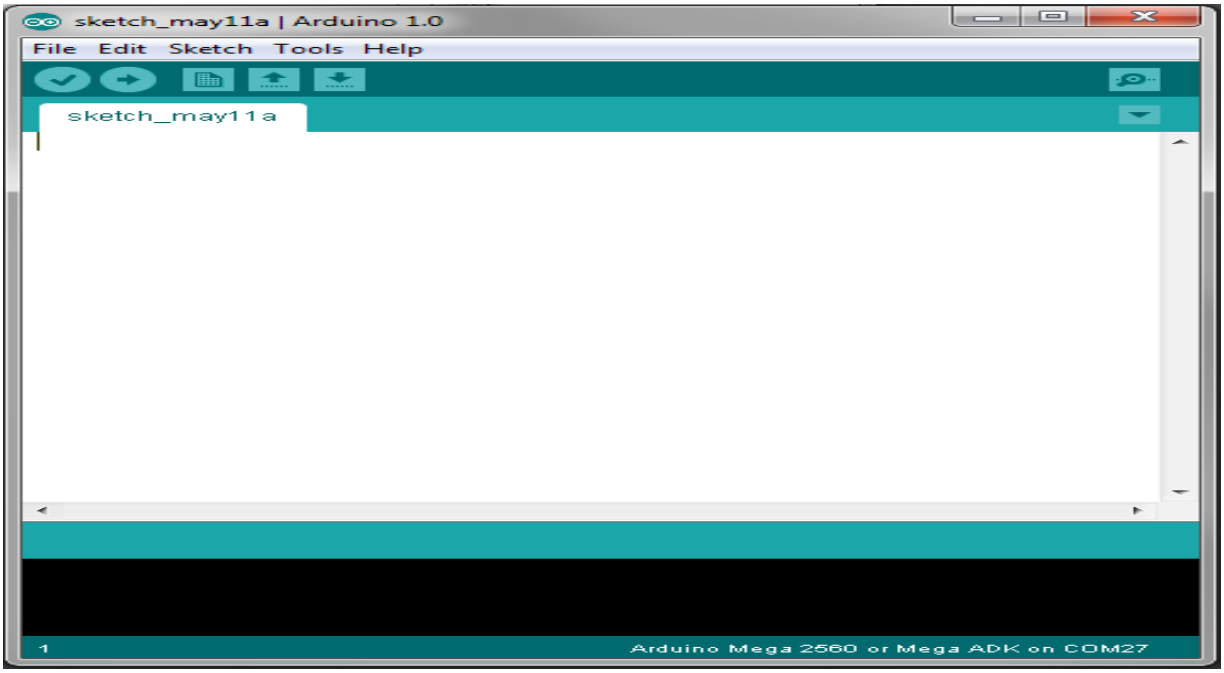
- ROBOTC Modified UNO Bootloader - Modified Bootloader

5.9 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.

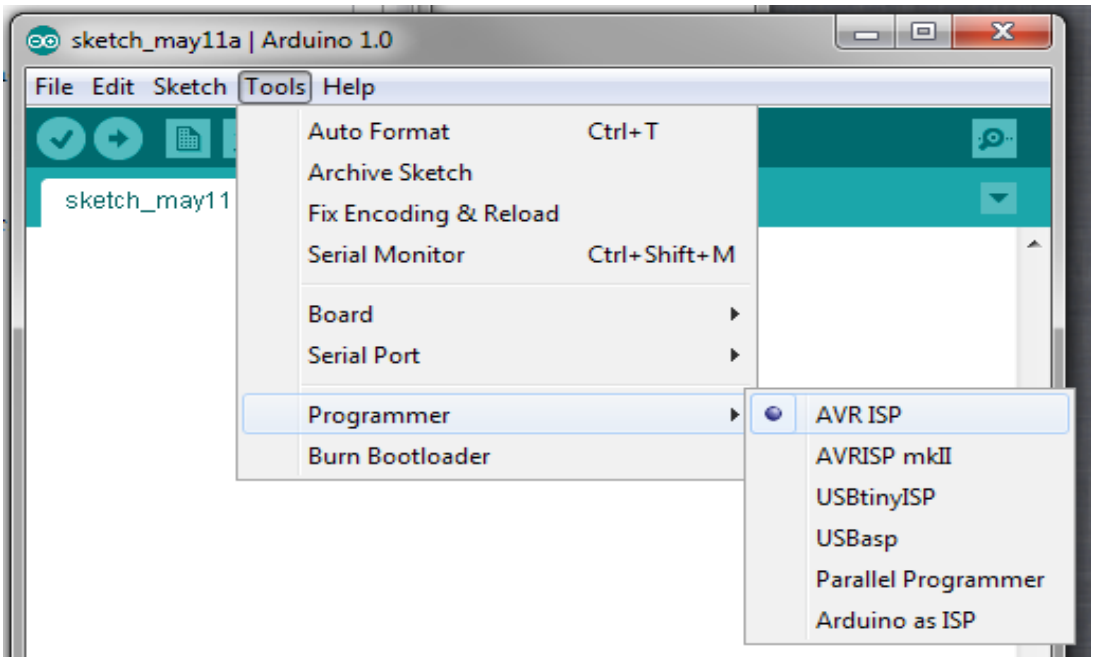


- Power up your Arduino UNO (either via USB or external power)
- Plug in your AVR ISP Programmer to your computer (make sure you have any required drivers installed)
- Connect your AVR ISP Programmer into your Arduino UNO Board via the ISP Header (the 2x3 header pins right above the Arduino Logo)
- Launch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

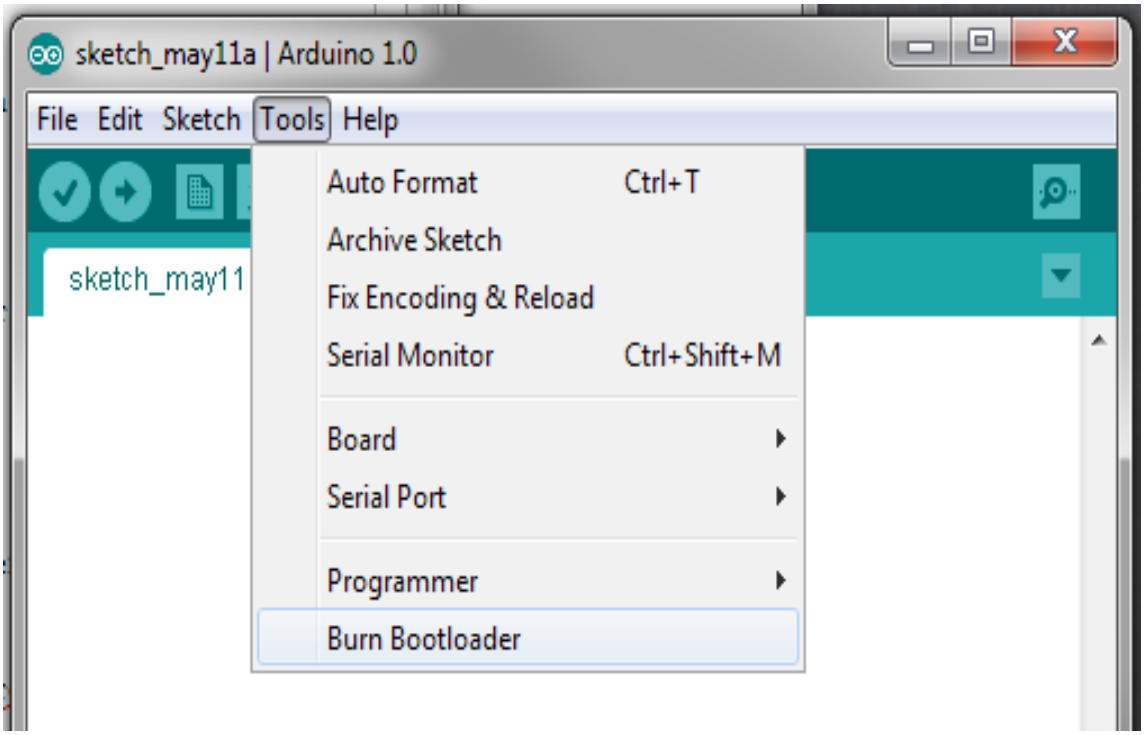


- Change your settings in the Arduino Software to look for an Arduino UNO

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



- 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.

CHAPTER 6

SUMMARY

Microcontroller	Arduino UNO
Operating Voltage	5V Input Voltage
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (14 provide PWM output)
DC Current per I/O Pin	40mA
DC Current for3.3VPin	50mA
Flash Memory	256 KB (8 KB used by bootloader)
SRAM	8KB
EEPROM	4KB
Lock Speed	16MHz
Analog input pin	16

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 volt supply generated by the on board regulator. Maximum current drawn 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.
- **PI: 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.
- **I^2C: 20 (SDA) and 21 (SCL).** Support I^2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I^2C pins on the Duemilanove.

The Arduino UNO has 16 analog inputs, each of which provides 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 analog inputs. Used with analog Reference ().
- **eset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

6.1 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 and1).

A Software Serial 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.

6.2 PROGRAMMING:

The Arduino UNO can be programmed with the Arduino software. The Arduino UNO on the Arduino UNO comes preburned with a boot loader 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.

6.3 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 nano farad 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.

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 100mil 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 the Mega and Duemilanove / Diecimila.

Please note that I^2C is not located on the same pins on the Mega (20and21) 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

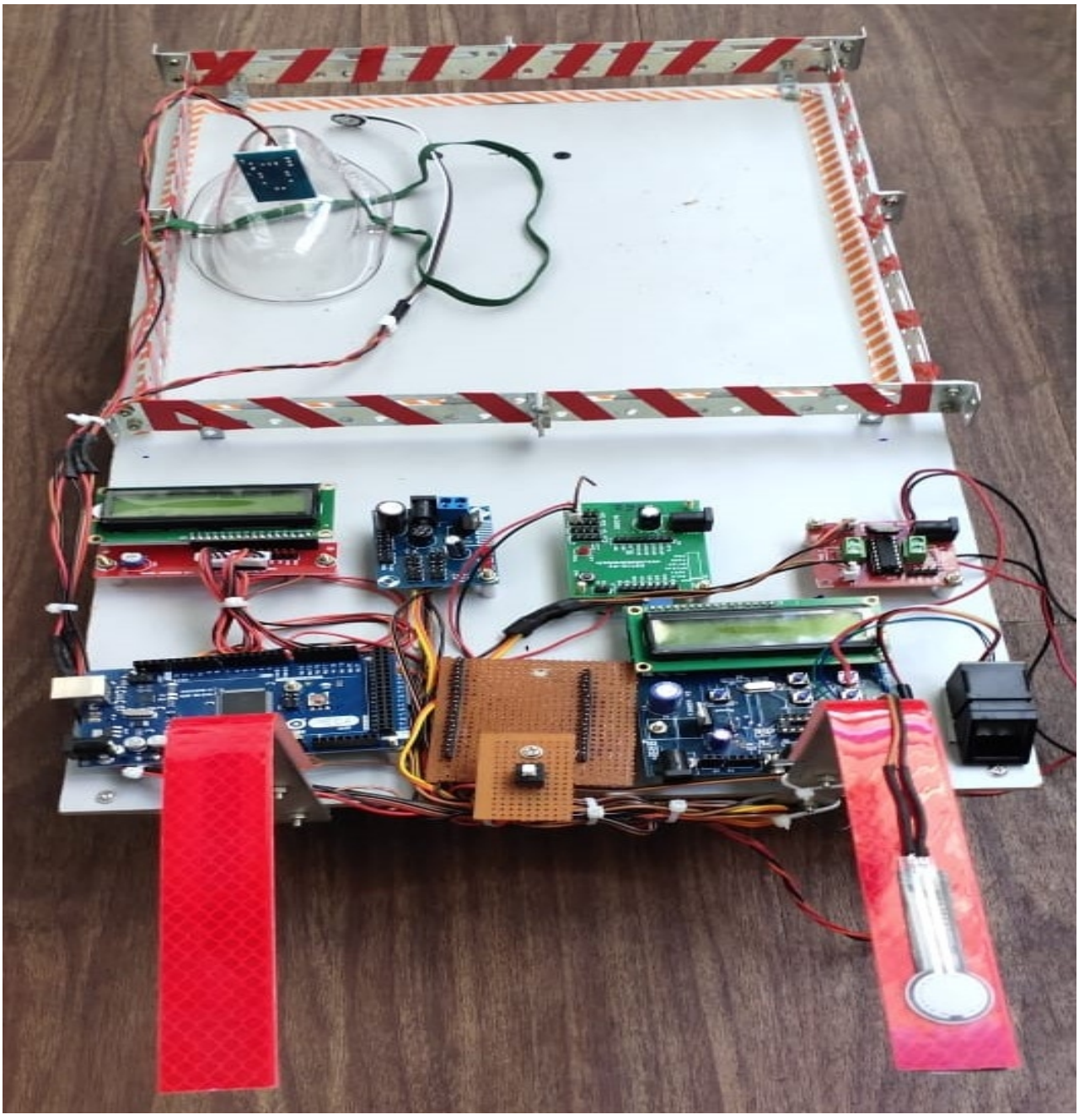
Once you have downloaded/unzipped the arduino IDE, you can plug the Arduino to your PC via USB cable.

CHAPTER 7

RESULT & CONCLUSION

RESULT:

As result of this project the unknown person's identity and their health is monitored and their current situation is been sent to the respective persons parents or family through the police station.



CONCLUSION:

As health care services are important part of our society, automating these services lessen the burden on humans and eases the measuring process. Also, the transparency of this system helps patients to trust it.

As continuous monitoring of patient condition is performed, physician can analyze current status of patient which will help to take decision him to provide suitable treatment.

REFERENCES:

- M. E. Mlinac, M. C. Feng,"Assessment of activities of daily living, selfcare, and independence", Archives of Clinical Neuropsychology, Vol. 31, Issue 6, Aug 2016.
- H. Liu, H. Darabi, P. Banerjee, J. Liu,"Survey of wireless indoor positioning techniques and systems", IEEE Trans. Syst. Man Cybern.C: Applications and Reviews, vol. 37, no. 6, Nov. 2007.
- V. Pasku et al.,"Magnetic Field Based Positioning Systems." IEEE Comm. Surveys & Tutorials, Mar 2017.
- K. Nguyen, Z. Luo, "Dynamic route prediction with the magnetic field strength for indoor positioning", Int. Journal of Wireless and Mob. Computing, Vol.12 Issue 1, Jan 2017.
- B. Gozick, K. P. Subbu, R. Dantu, T. Maeshiro,"Magnetic maps for indoor navigation", IEEE Trans. Instrum. Meas., vol. 60, no. 12, Dec. 2011.
- Alarifi et al., "Ultra wideband indoor positioning technologies: Analysis and recent advances," Sensors (Basel), vol. 16, no. 5, May 2016.
- Rambla, J. Huerta, "Providing databases for different indoor positioning technologies: Pros and cons of magnetic field and Wi-Fi based positioning," in Proc. Mobile Inf. Syst., vol. 2016, Jan. 2016.

- X. Zhao, Z. Xiao, A. Markham, N. Trigoni, and Y. Ren, “Does BTLE measure up against Wi-Fi? A comparison of indoor location performance,” in Proc. Eur. Wireless 20th Eur. Wireless Conf., Barcelona, Spain, 2014.
- P. Davidson and R. Piche, “A survey of selected indoor positioning methods for smartphones”, IEEE Comm. Surveys & Tutorials, 2016.
- Y. Li, Y. Zhuang, H. Lan, P. Zhang, X. Niu, N. El-Sheimy, “Wi-FiAided magnetic matching for indoor navigation with consumer portable devices”, Micromachines, vol. 6, no. 6, 2015.
- E. Wang, M. Wang, Z. Meng, X. Xu. ”A Study of Wi-Fi-Aided Magnetic Matching Indoor Positioning Algorithm.” Journal of Computer and Comm. 5.03, 2017.
- S. Dernbach, B. Das, N. C. Krishnan, B. L. Thomas, D. J. Cook, ”Simple and complex activity recognition through smart phones”, Proc. 8th IE, Jun. 2012.
- J. Torres, O. Belmonte, R. Montoliu, S. Trilles, and A. Calia, “How feasible is WiFi fingerprint-based indoor positioning for in-home monitoring?” in Proc. 12th Int. Conf. Intel. Environ. (IE), Sep. 2016.
- R. Jimenez and F. Seco, “Event-driven Real-time Location-aware Activity Recognition in AAL Scenarios,” in Proc. 12th Int. Conf. on Ubiq. Computing and Amb. Intel., UCAmI 2018, 4-7 Dec, 2018; Punta Cana, Dominican Repu