INTERNET OF THINGS IN HEALTHCARE

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

Bachelor of Technology.

in

Computer Science & Engineering

under the Supervision of

PUNIT GUPTA

By

DEEPIKA AGGARWAL (**111304*****)***

to

Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

Certificate

This is to certify that project report entitled “Internet Of Things In Healthcare”, submitted by ***Deepika Aggarwal*** in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science & Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date:15-05-2015

PUNIT GUPTA
Assistant Professor

Acknowledgement

I take this opportunity to express my profound gratitude and deep regards to my guide Mr. Punit Gupta for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by his time to time shall carry me a long way in the journey of life on which I am about to embark.

The in-time facilities provided by the Computer Science department throughout the project development are also equally acknowledgeable.

At the end I would like to express my sincere thanks to all my friends and others who helped me directly or indirectly during this project work.

Date: 15-05-2015 **Deepika Aggarwal**

111304

Table of Contents

List of Figures

Abstract

In the Internet of Things (IoT), devices gather and share information directly with each other and the cloud, making it possible to collect, record and analyze new data streams faster and more accurately. The rapid development of Internet of things (IoT) technology makes it possible for connecting various smart objects together through the Internet and providing more data interoperability methods for application purpose. Recent research shows more potential applications of IoT in information intensive industrial sectors such as healthcare services. In this project, I present an IoT-based system for emergency medical services to demonstrate how to collect, integrate, and interoperate IoT data flexibly in order to provide support to emergency medical services. The IoT has already brought in significant changes in many areas of healthcare. It is rapidly changing the healthcare scenario by focusing on the way people, devices and apps are connected and interact with each other.

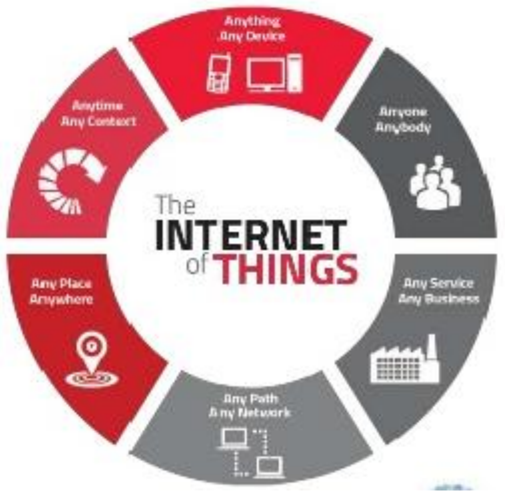
The idea of this project came to reduce the headache of patient to visit to doctor every time he needs to check his blood pressure, heart beat rate, temperature etc. With the help of my project I am hoping to save the time of both patients and doctors and also to help doctors in emergency scenario as much as possible.

In this generation where fully automatic systems are used almost in every life purpose, we have designed this project to serve the purpose of human need. This project is an application of the microcontroller that we use in our daily life. Normally, as we see that in today's world everyone is busy and has to go here and there for work so they do not care about their health and do not put attention to small healthcare problems like high blood pressure, low pulse rate etc. The project can be used to avoid the above problem. Basically I am developing this application to help doctors for emergency cases. Because in current scenario what happens is when any emergency arises emergency alert is send to hospital not to patient's doctor directly. So the problem statement of my project is to connect and collect data information through health status monitors which would include patient's heart rate, blood pressure and ECG and sends an emergency alert to patient's doctor with his current status and full medical information.

CHAPTER 1 : INTRODUCTION

1.1 Introduction : Internet of Things

IoT is a network of devices that connect directly with each other to capture and share vital data through a secure service layer (SSL) that connects to a central command and control server in the cloud. Let’s begin with a closer look at what that entails and what it suggests for the way people collect, record and analyze data—not just in healthcare, but in virtually every industry today. The idea of devices connecting directly with each other is, as the man who coined the term Internet of Things puts it, “a big deal.”1 As Kevin Ashton explained a decade after first using the phrase at a business presentation in 1999, “Today computers—and therefore, the Internet—are almost wholly dependent on human beings for information. The problem is, people have limited, time, attention and accuracy—all of which means they are not very good at capturing data about things in the real world.”1 The solution, he has always believed, is empowering devices to gather information on their own, without human intervention.



The Internet of Things, also called The Internet of Objects, refers to a wireless network between objects, usually the network will be wireless and self-configuring.Internet of Things (IoT) is one of the major component advances in present time that links the internet with everyday sensors and working devices.

“Smart” objects play a key role in the Internet of Things vision, since embedded communication and information technology would have the potential to revolutionize the utility of these objects. Using sensors, they are able to perceive their context, and via built-in networking capabilities they would be able to communicate with each other, access Internet services and interact with people. “Digitally upgrading” conventional object in this way enhances their physical function by adding the capabilities of digital objects, thus generating substantial added value. Forerunners of this development are already apparent today – more and more devices such as sewing machines, exercise bikes, electric toothbrushes, washing machines, electricity meters and photocopiers are being “computerized” and equipped with network interfaces.

From any time, any place connectivity for anyone, we will now have connectivity for anything!

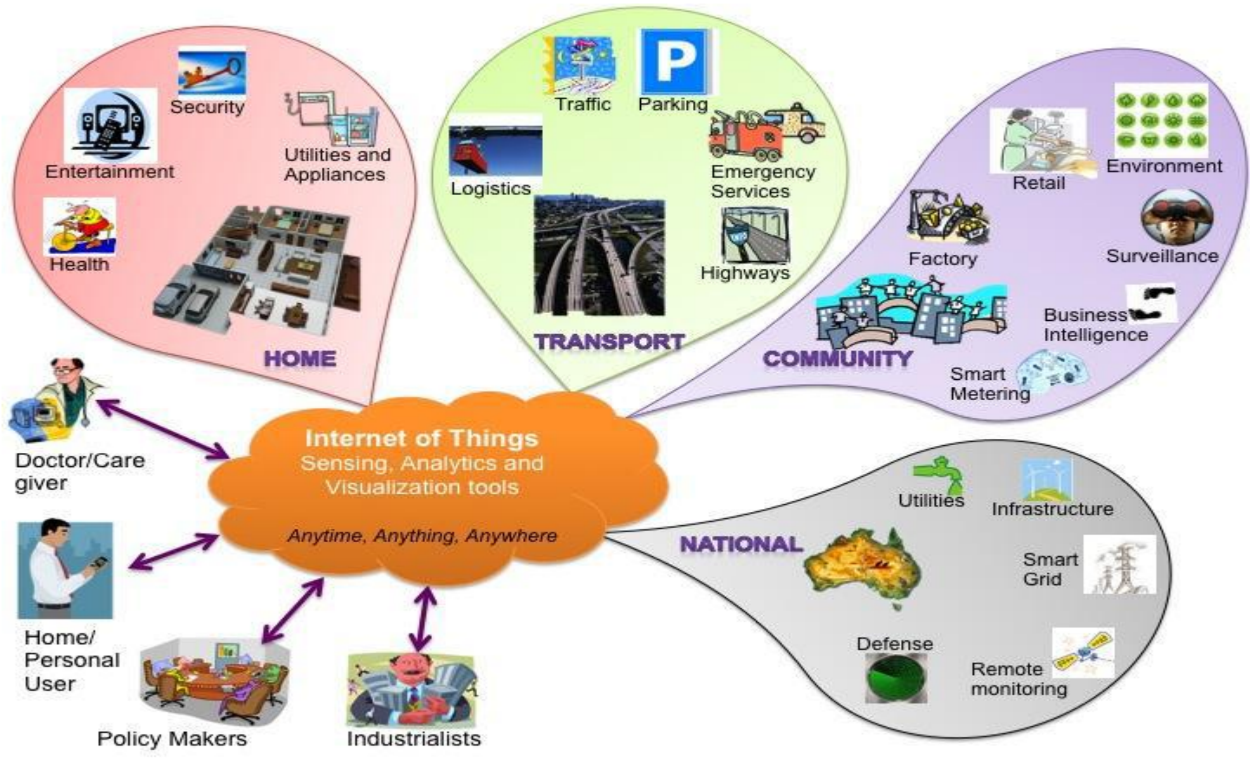


Figure 1: Internet of Things

In other application domains, Internet connectivity of everyday objects can be used to remotely determine their state so that information systems can collect up-to-date information on physical objects and processes. This enables many aspects of the real world to be “observed” at a previously unattained level of detail and at negligible cost. This would not only allow for a better understanding of the underlying processes, but also for more efficient control and management The ability to react to events in the physical world in an automatic, rapid and informed. manner not only opens up new opportunities for dealing with complex or critical situations, but also enables a wide variety of business processes to be optimized. The real-time interpretation of data from the physical world will most likely lead to the introduction of various novel business services and may deliver substantial economic and social benefits.

The use of the word “Internet” in the catchy term “Internet of Things” which stands for the vision outlined above can be seen as either simply a metaphor – in the same way that people use the Web today, things will soon also communicate with each other, use services, provide data and thus generate added value – or it can be interpreted in a stricter technical sense, postulating that an IP protocol stack will be used by smart things.

From a technical point of view, the Internet of Things is not the result of a single novel technology; instead, several complementary technical developments provide capabilities that taken together help to bridge the gap between the virtual and physical world. These capabilities include:

- **Communication and cooperation:** Objects have the ability to network with Internet resources or even with each other, to make use of data and services and update their state. Wireless technologies such as GSM and UMTS, Wi-Fi, Bluetooth, ZigBee and various other wireless networking standards currently under development, particularly those relating to Wireless Personal Area Networks (WPANs), are of primary relevance here.

- **Addressability:** Within an Internet of Things, objects can be located and addressed via discovery, look-up or name services, and hence remotely interrogated or configured.

- **Sensing:** Objects collect information about their surroundings with sensors, record it, forward it or react directly to it.

- **Actuation:** Objects contain actuators to manipulate their environment (for example by converting electrical signals into mechanical movement). Such actuators can be used to remotely control real-world processes via the Internet.

- **Embedded information processing:** Smart objects feature a processor or microcontroller, plus storage capacity. These resources can be used, for example, to process and interpret sensor information, or to give products a "memory" of how they have been used.

- **Localization:** Smart things are aware of their physical location, or can be located. GPS or the mobile phone network are suitable technologies to achieve this, as well as ultrasound time measurements, UWB (Ultra-Wide Band), radio beacons (e.g. neighboring WLAN base stations or RFID readers with known coordinates) and optical technologies.

- **User interfaces:** Smart objects can communicate with people in an appropriate manner (either directly or indirectly, for example via a Smartphone). Innovative interaction paradigms are relevant here, such as tangible user interfaces, flexible polymer-based displays and voice, image or gesture recognition methods.

1.1.1 Emergence of Internet of Things

The emergence of the IoT, in which devices connect directly to data and to each other, is important for two reasons:

1. Advances in sensor and connectivity technology are allowing devices to collect , record and analyze data that was not accessible before. In healthcare, this means being able to collect patient data over time that can be used to help enable preventive care, allow prompt diagnosis of acute

complications and promote understanding of how a therapy (usually pharmacological) is helping improve a patient’s parameters.

2. The ability of devices to gather data on their own removes the limitations of human-entered data—automatically obtaining the data doctors need, at the time and in the way they need it. The automation reduces the risk of error. Fewer errors can mean increased efficiency, lower costs and improvements in quality in just about any industry. But it’s of particular interest need in healthcare, where human error can literally be the difference between life and death.

1.1.2 IoT Building Blocks Emerging Everywhere

Even though only “1 percent of things are connected today,”2 according to Joseph Bradley, general manager of Cisco Consulting Services, businesses across a variety of industries are establishing the building blocks of the IoT infrastructure. Here are a few examples:

- **Home and building automation:** Digital marketer Lauren Fisher points to the Nest Learning Thermostat, which takes data about the home environment and owners’ temperature preferences and programs itself to operate efficiently within the context of that information. This technical framework provides energy providers with the connectivity to better manage the energy grid.

- **Automotive design and manufacturing:** Mobile virtual network operator Alex Brisbourne describes how the automotive industry is increasingly designing automated applications into vehicles to provide maintenance monitoring, fuel and mileage management, driver security and other capabilities that cost little to integrate but have significant earning potential. The addition of a cloud-based server to analyze the data and automatically act on it—automatically scheduling a maintenance appointment at the appropriate time, for example— would move this further in the direction of the IoT.

- **Public transportation/smart cities:** Technology writer Martyn Casserly cites the London iBus system, which "…works with information from over 8,000 buses that are fitted with GPS capabilities alongside various other sensors which relay data about the vehicle's location and current progress,"5 so bus stop signposts can display details of a bus's impending arrival.

IoT concepts have already been adopted in areas such as energy (e.g., smart lighting, smart grid) and industrial automation. According to a report in eWeek2 about a Cisco conference call with journalists, "…as more connections are made, the value to businesses and the global economy will only go up." The eWeek story describes a Cisco vision that goes beyond the IoT to IoE, or the Internet of Everything. This is what Cisco sees as a system of connections that includes not only devices, but also people, data and processes—"…essentially whatever is connected to or crosses over the Internet." Cisco expects the IoE to be worth $14.4 trillion to the global economy by 2020.

1.2 Introduction : Internet of Things in Healthcare

The Internet of things in healthcare encompasses heterogeneous computing and wireless communication systems, apps and devices that help patients and providers alike to monitor, track and store patients' vital statistics or medical information. Examples of such systems are smart meters, RFID, wearable health monitoring sensors, and smart video cameras. Also, smart phones, intelligent vehicles, and robotics are considered to be the part of IoT.

These IoT devices produce enormous amounts of data which becomes a challenge for providers to deal with efficiently. To harness this huge data in a technological way and make sense of it, the Internet of Things Analytics (IoTA) is implemented. Data mining, data management and data analytics techniques are used to make this deluge of data useful and medically relevant. In fact, it has been predicted that by 2017, more than 50 percent of analytics techniques will make a better use of this influx of data which is generated from instrumented machines and applications.

1.2.1 Enabling Technologies: Making the IoT in Healthcare Possible

The successful use of the IoT in the preceding healthcare examples relies on several enabling technologies. Without these, it would be impossible to achieve the usability, connectivity and capabilities required for applications in areas such as health monitoring.

Smart sensors, which combine a sensor and a microcontroller, make it possible to harness the power of the IoT for healthcare by accurately measuring, monitoring and analyzing a variety of health status indicators. These can include basic vital signs such as heart rate and blood pressure, as well as levels of glucose or oxygen saturation in the blood. Smart sensors can even be incorporated into pill bottles and connected to the network to indicate whether a patient has taken a scheduled dose of medication. For smart sensors to work effectively, the microcontroller components must incorporate several essential capabilities:

• **Low-power operation** is essential to keeping device footprint small and extending battery life, characteristics that help make IoT devices as usable as possible. Freescale, which has long offered low-power processing, is working now to enable completely battery-free devices that utilize energy harvesting techniques through the use of ultra-low-power DC-DC converters.

• **Integrated precision-analog capabilities** make it possible for sensors to achieve high accuracy at a low cost. Freescale offers this enabling technology within microcontrollers which contain analog components, such as high-resolution analog-to-digital converters (ADCs) and low-power op-amps.

• **Graphical user interfaces (GUIs)** improve usability by enabling display devices to deliver a great deal of information in vivid detail and by making it easy to access that information. Freescale’s i.MX applications processors with high graphics-processing performance support advanced GUI development.

1.2.2 Transforming Healthcare with the Internet of Things

In the US healthcare industry, the small and big healthcare organizations are using the Internet of Things tools and devices which are revolutionizing medical care in unique ways. From headsets that measure brainwaves to clothes that include sensing devices, Google Glass, BP monitors etc., all these have taken personal health monitoring to a new level.

According to ABI Research, by 2016, the sale of wearable wireless medical device will bloom and reach more than 100 million devices annually. Another report by IMS Research, the research partner of Wearable Technologies, states that the devices which are wearable or are close to the body produce more realistic results.

There has been clinical evidence that the physiological data received from wireless devices has been a valuable contributor for managing or preventing chronic diseases and monitoring patients post hospitalization. As a result, a growing number of medical devices are becoming wearable nowadays, including glucose monitors, ECG monitors, pulse audiometers, and blood pressure monitors and so on.

The market for wearable technologies in healthcare is expected to exceed by $2.9 billion in 2016. The Internet of Things enables health organizations to achieve superior technology interoperability, lift critical data from multiple sources in real-time, and a better decision-making capability. This trend is transforming healthcare sector, increasing its efficiency, lowering costs and providing avenues for better patient care.

1.2.3 IoT- A Healthcare Game Changer

The IoT has already brought in significant changes in many areas of healthcare. It is rapidly changing the healthcare scenario by focusing on the way people, devices and apps are connected and interact with each other. IoT is coming out as the most promising information communications technology (ICT) solution which enables providers to improve healthcare outcomes and reduce healthcare costs by collecting, recording, analyzing and sharing myriads of new data streams in real time and flawlessly. Moreover, as the widespread adoption of IoT grows, many of the inefficiencies in healthcare will be reduced. For example, sensors embedded in medical devices such as diagnostic equipment, drug dispensing systems, surgical robots, implantable devices, personal health and fitness sensors, etc., will perform data collections, measurements, and conduct tests digitally in no time which are currently administered and recorded manually.

This is extremely important for gaining new insights and knowledge on various issues in healthcare. For example, to study a patient's response to a specific therapy or drugs, traditionally, healthcare providers study different samples taken from the patients. These samples sometimes are not up to the required standard to give clearer results. But, IoT has made it possible for the first time to collect real-time data from unlimited number of patients for a definite period of time through connecting devices. It is anticipated that it will also improve healthcare services for people in remote locations as monitoring systems provide a continuous stream of data that enable healthcare providers to make better decisions. IoT is gaining momentum among healthcare providers and healthcare IT and is emerging as a major technology trend for improved healthcare.

1.3 Introduction : Intel Galileo Board Gen 2

The 2nd generation Intel Galileo board provides a single board controller for maker community, students and professional developers. Based on the Intel Quark SoCX1000, a 32-bit Intel Pentium processor- class system on a chip (SoC), the genuine board provides a full featured offering for a wide range of applications. Arduino certificated and designed to be hardware, software, and pin compatible with large range of Arduino Uno R3 shields, the board also provides a simple and more cost effective development environment compared to the Intel Atom processor and Intel Core processor based designs. Based on a new micro-architecture, the processor is designed for a two-chip platform consisting of a processor and Platform Controller Hub (PCH). The platform enables higher performance, lower cost, easier validation, and improved x-y footprint. The processor includes Integrated Display Engine, Processor Graphics, PCI Express ports, and Integrated Memory Controller. The processor is designed for desktop platforms. It supports up to 12 Processor Graphics execution units (EUs).

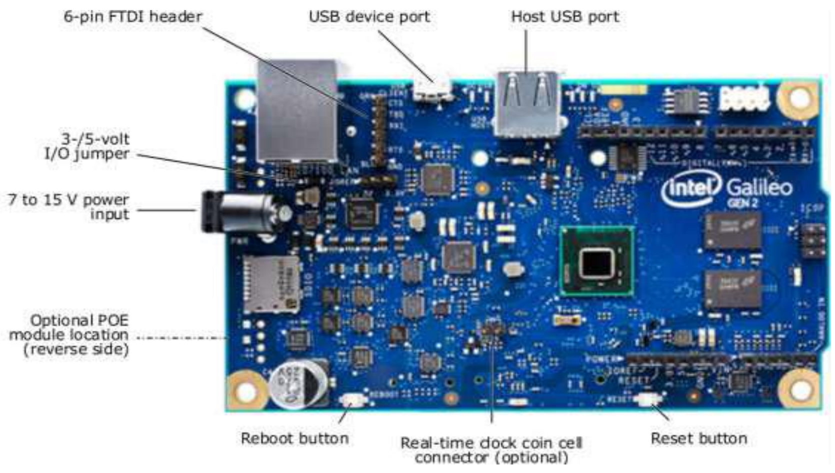


Figure 2: Outer view of Intel Galileo Board Gen 2

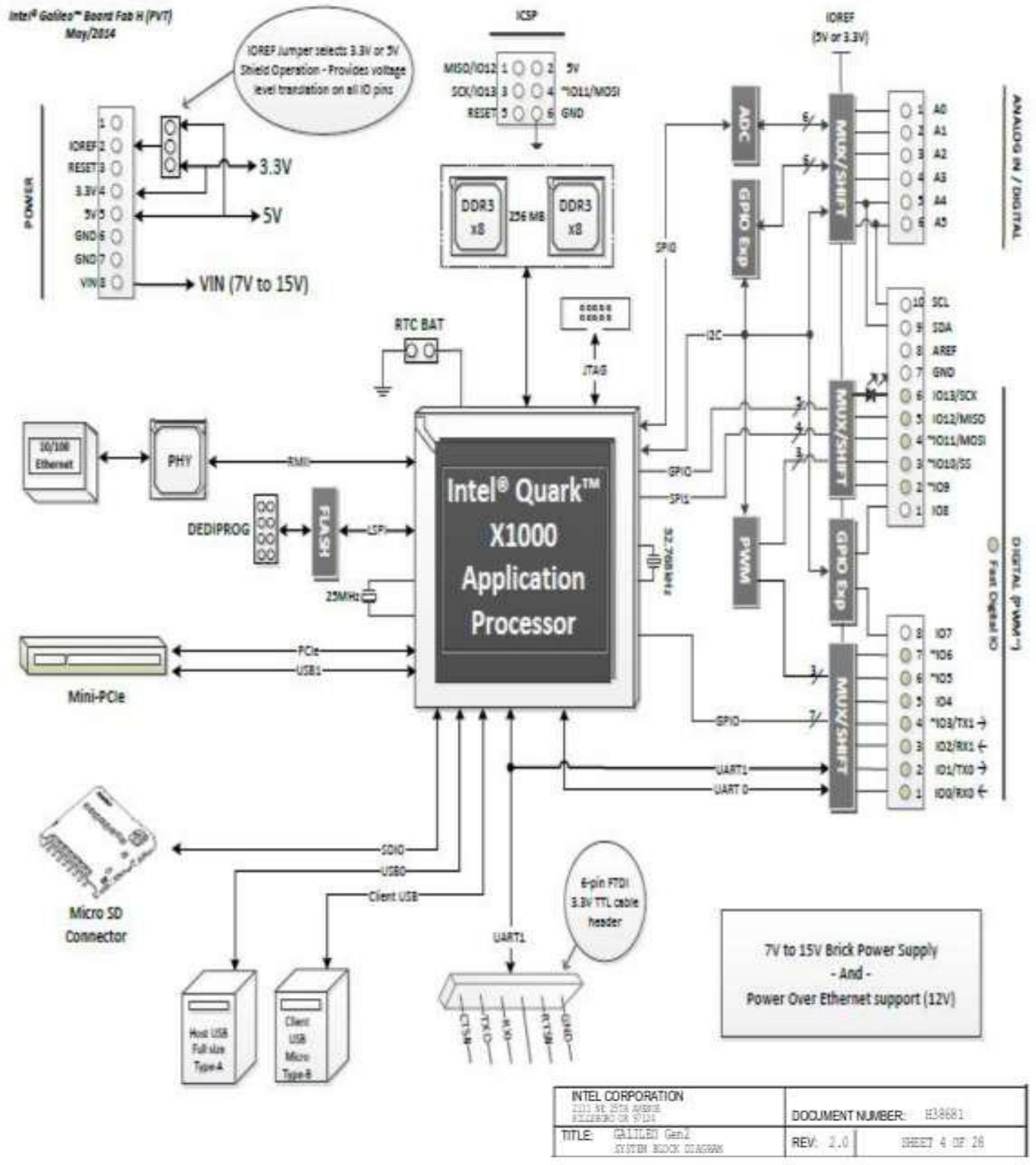


Figure 3: Pin configuration of Intel Galileo Board Gen 2

1.3.1 Physical Characteristics

- 10 cm long and 7 cm wide with the USB connectors, UART jack, Ethernet connector, and power jack extending beyond the former dimension
- Four screw holes allow the board to be attached to a surface or case
- Reset button to reset the sketch and any attached shields

1.3.2 Processor Features

Instruction set architecture (ISA)-compatible 32-bit Intel® Pentium® processor

- 16 Kbytes L1 cache
- 512 Kbytes of on-die embedded SRAM
- Simple to program: single thread, single core, constant speed
- ACPI-compatible CPU sleep states supported
- Integrated real-time clock (RTC) with optional 3V "coin cell" battery for operation between turn on cycles
- 400 MHz clock speed

1.3.3 Storage Options

- 8 Mbyte Legacy SPI Flash to store firmware (bootloader) and the latest sketch
- Between 256 Kbytes and 512 Kbytes dedicated for sketch storage
- 512 Kbytes embedded SRAM
- 256 Mbytes DRAM
- Optional micro SD card offers up to 32 Gbytes of storage
- USB storage works with any USB 2.0 compatible drive
- 11 Kbytes EEPROM programmed via the EEPROM library

1.3.4 Enhancements

The Intel Galileo board (Gen 2) delivers improved features and functionality in following areas:

- 12 GPIOs fully native for greater speed and improved drive strength.
- 12-bit PWM for more precise control of servos and smoother response.
- 12 V Power-over-Ethernet capable.
- Power supplies from 7 V to 15 V are supported.
- Serial console URAT header is compatible with FTDI USB converters.
- Console URAT1 can be redirected to Arduino headers in sketches, which can eliminate the need for soft-serial.

1.3.5 Detailed View Of Intel Galileo Board

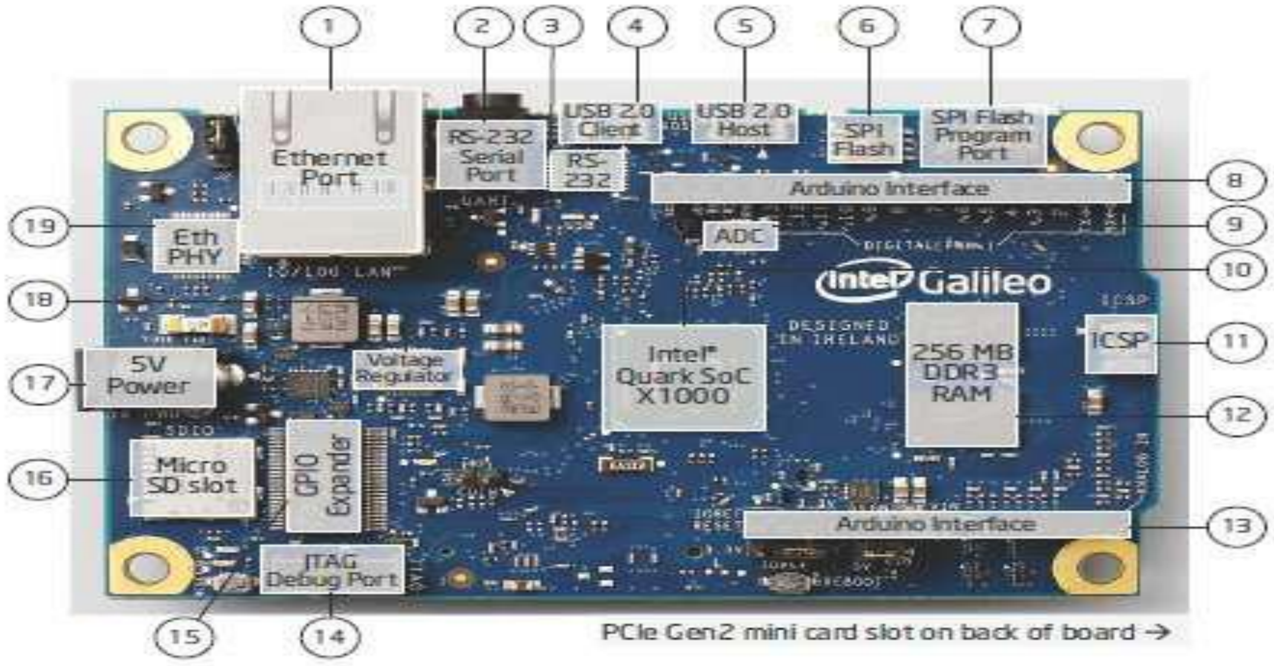


Figure 4: Detailed View of Board

- 6-pin 3.3V USB TTL UART header replaces 3.5 mm jack RS-232 console port for Linux debug. New 6-pin connector mates with standard FTDI* USB serial cable (TTL-232R-3V3) and popular USB-to-Serial breakout boards. 12 GPIOs now fully native for greater speed and improved drive strength.
- 12-bit pulse-width modulation (PWM) for more precise control of servos and smoother response.
- Console UART1 can be redirected to Arduino headers in sketches, eliminating the need for soft-serial in many cases.
- 12V power-over-Ethernet (PoE) capable (PoE module installation required).
- Power regulation system changed to accept power supplies from 7V to 15V.

Digital pins 0 to 13 (and the adjacent AREF and GND pins), Analog inputs 0 to 5, the power header, ICSP header, and the UART port pins (0 and 1), are all in the same locations as on the Arduino Uno R3. This is also known as the Arduino 1.0 pinout.

Galileo is designed to support shields that operate at either 3.3V or 5V. The core operating voltage of Galileo is 3.3V. However, a jumper on the board enables voltage translation to 5V at the I/O pins. This provides support for 5V Uno shields and is the default behavior. By switching the jumper position, the voltage translation can be disabled to provide 3.3V operation at the I/O pins.

CHAPTER 2 : LITERATURE REVIEW

2.1 Title: The Potential of Internet of m-health Things "m-IoT" for Non-Invasive Glucose level Sensing

An amalgamated concept of Internet of m-health Things (m-IoT) has been introduced recently and defined as a new concept that matches the functionalities of m-health and IoT for a new and innovative future (4G health) applications. It is well know that diabetes is a major chronic disease problem worldwide with major economic and social impact. To-date there have not been any studies that address the potential of m- IoT for non-invasive glucose level sensing with advanced opto-physiological assessment technique and diabetes management. This paper addresses the potential benefits of using m-IoT in non-invasive glucose level sensing and the potential m-IoT based architecture for diabetes management. We expect to achieve intelligent identification and management in a heterogeneous connectivity environment from the mobile healthcare perspective. Furthermore this technology will enable new communication connectivity routes between mobile patients and care services through innovative IP based networking architectures.

M-IoT introduce a new healthcare connectivity paradigm that interconnects IPV6 based communication technologies such as 6LoWPAN with emerging 4G networks for future Internet based m-health services. From the glucose level sensing end, it is well known that real-time/ continuous measurement can provide significant clinical information of the glucose levels and conditions to provide timely intervention of any hyperglycemic episodes of the diabetic patient in real-time compared to non-real time measurements as it is the case currently. In recent years several non-invasive glucose level measurement methods have been addressed and the opto-physiological modeling could provide an approach for effective interpretation of these measurement [8]. These non-invasive glucose monitoring sensors are mainly attached onto the skin, which, as the largest and outermost organ of the body, accounts for 10–15% of the body mass and has one of the lowest metabolic rates, thus having relatively low nutritive requirements. Hence the skin is employed as a blood reservoir by the body, changes in the blood requirements of other organs lead to the changes in the cutaneous blood contents including glucose level. *In-vivo* non-invasive glucose monitoring has the promise of providing great relief to many patients with diabetes mellitus for whom tight glucose control is are directly dependent on the tissue composition and desirable for reducing the effects of long-term complications or helping to avoid the potentially life-threatening condition of hypoglycemia. Currently, the control of the body glucose is performed in a blood sample requiring painful puncture of the skin in order to draw a drop of blood, typically from the fingertip. The demands from diabetes clinical management and patient self monitoring are moving onto an effective approach when integrating a high performance optoelectronic sensor into state-of-the-art m-Health system. However, to date there is no study to address the potential m- IoT connectivity with non-invasive glucose level sensing.

This paper addresses such m-IoT based architecture and the potential implementation issues and challenges. It also describes the methodology adopted to achieve the goals of this work and shows some of the preliminary and evaluation results of a test bed implementation for general m-IoT system.

2.1.2 METHOD: M-IOT FOR REAL-TIME GLUCOSE SENSING

The m-IoT system based on IPV6 and 6LoWPAN protocol architectures. In this configuration, the non-invasive diabetes sensors from the patients are linked via IPv6 connectivity to the relevant healthcare provider or the diabetes centre. The fundamental role in the architecture is the 6LoWPAN protocol enabling wireless sensor devices for all IP based wireless nodes based on IEEE 802.15.4 standard. The potential of the 6LoWPAN in addition to the IPV6 connectivity is the low power characteristics of these nodes.

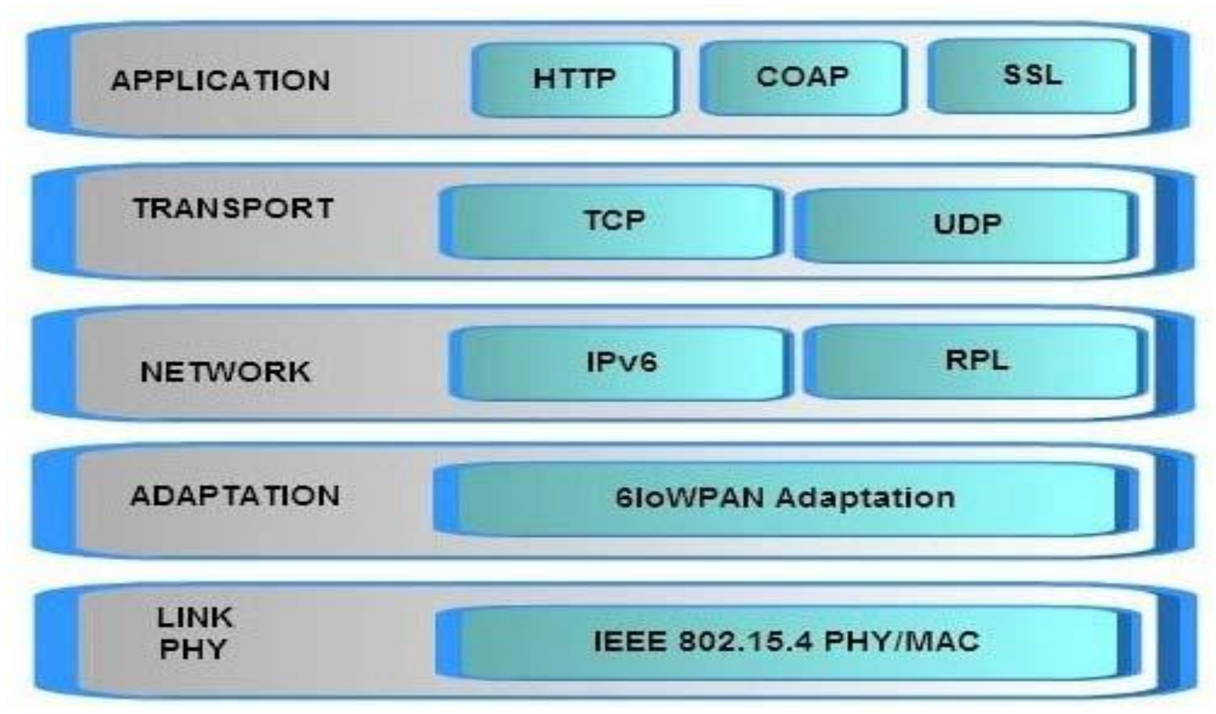


Figure 5 : 6LoWPAN based IoT architecture

An IPV6 access point sends the relevant diabetes clinical data to an IP-based connected healthcare centre. Data can be sent and collected both periodically or event-driven.

The experimental 6LoWPAN based system is shown in Figure 6. In this work a test bed platform is build and tested to verify the m-IoT concept and its related communication setup and requirements in terms of both hardware and software. The sensor used in this experiment is A Body temperature sensor that is build in the TelosB mote used. The next step is to use the non-invasive opto-physiological sensor for glucose level monitoring. The output of this noninvasive sensor can be connected to the TelosB mote. TelosB mote has a built in Analogue-to-Digital convertor that accept an analogue signal of 0-3 Volt and hence we need some signal conditioning interface between the non- Invasive sensor and the 10-pin (U2) and 6-pin (U28) expansion connectors of the TelosB.

In real medical scenario each diabetes node (representing individual mobile patient end) can have its own IPv6 address for relevant m-IoT and direct Internet connectivity from any mobile device linked or embedded with these devices. The biomedical information from these opto-physiological sensors is sent in real-time to a central access point where data is collected to special IP linked diabetes management system such as the one described elsewhere for further clinical analysis. The results show the successful implementation of the developed m-IoT system. Work is currently is ongoing to connect the non-invasive sensor within the TelosB node for the real-time glucose data measurement.

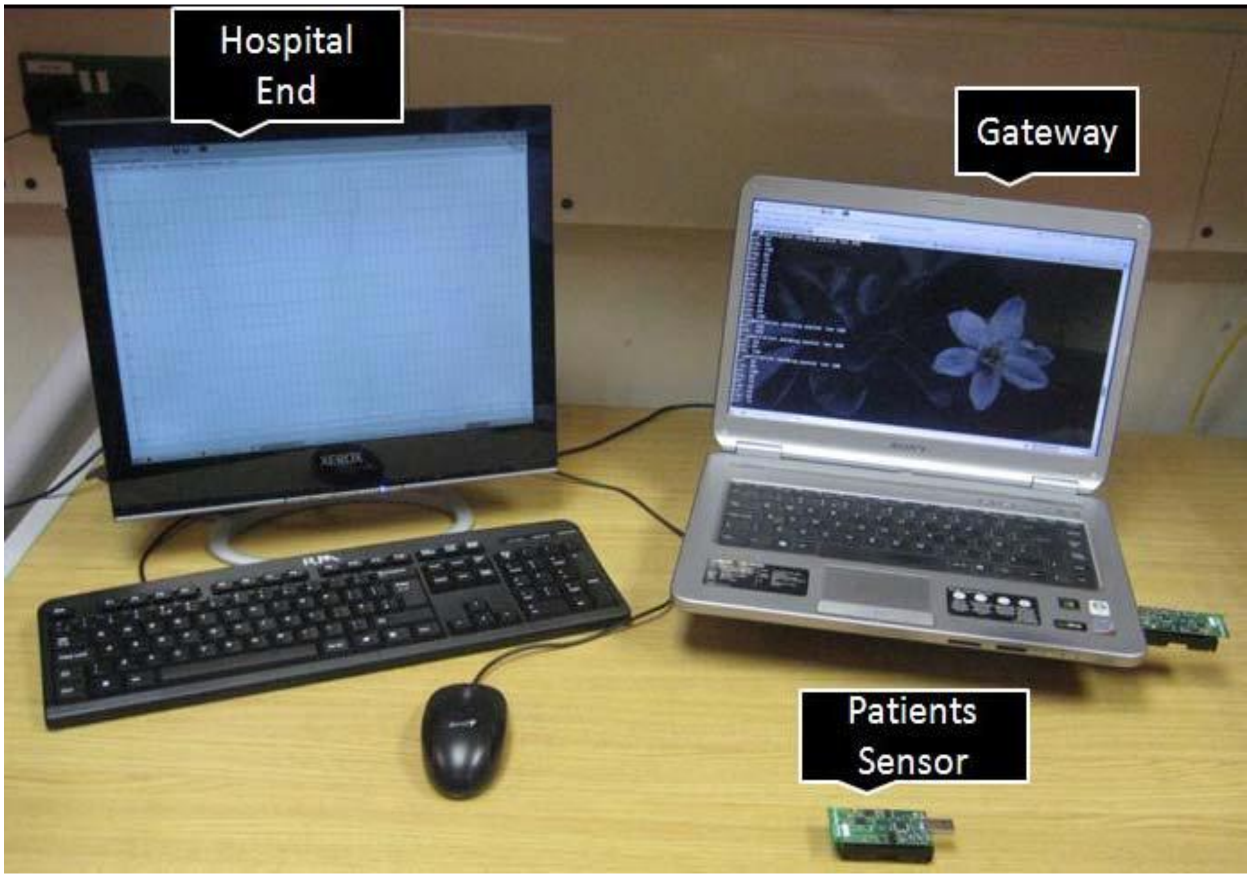
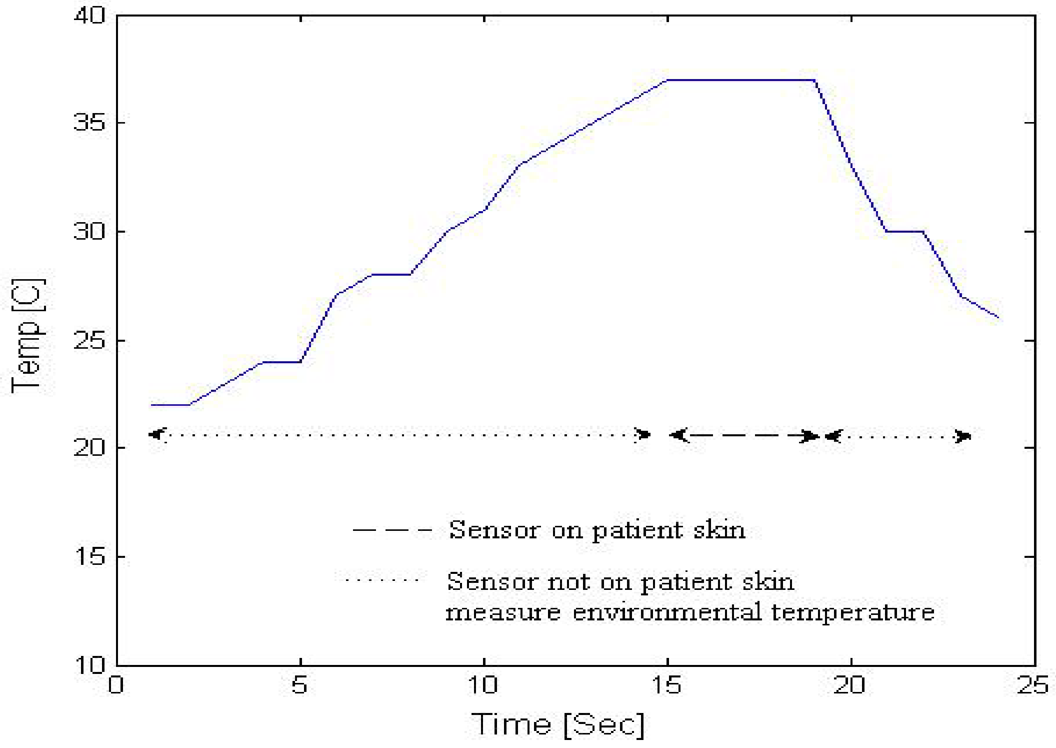


Figure 6 : Experimental m-IoT system



Sample of m-IoT based Temperature data

In this paper a new application concept has been presented that links the m-IoT connectivity for real-time non-invasive glucose level measurements in Diabetic patients. An experimental system has been implemented using the 6LoWPAN and TelsoB nodes sensor for verifying the performance of the system. Ongoing work is currently underway to build up and integrate the m-IoT node with the non-invasive opto-electronic sensor for real-time glucose levels and to compare the performance of this new sensor and system set-up with existing off-shelf glucose sensors. Above figure shows a block diagram of the m-IoT based noninvasive Opto-physiological Glucose sensor.

A typical Opto-physiological assessment sensor

A novel non-invasive sensors design and set-up based on opto-physiological modeling as shown in Fig. above is being considered to integrate with m-IoT and direct Internet connectivity from any mobile devices. To this end, an emerging integrated technique will be explored and investigated towards generating a better performance in the physiological signal capture of the non-invasive glucose monitoring sensor based upon the existing Loughborough University opto-physiological assessment technology. This senor could also combine the accelerometer motion sensor within the integrated sensor cavity. The identical wavelength light sources together with photodiode coupled will be ideally suited for in-vivo and real-time glucose and physical activity measurements and features. Hence the future work will consider to

1) research into performance and reliability of the non-invasive glucose monitoring output, refining and revising of the relevant processes of signal capturing, processing and transferring;

2) comparison with available diabetic assessment device to achieve a stand level of clinical diabetes assessment, and easy-to-operate and routine management; and

3) compliance with the regulations and standards of the applicable UK & Globe health care standards.

2.2 Title : RFID Technology for IoT-Based Personal Healthcare in Smart Spaces

This paper is aimed at drawing a landscape of the current research on RFID sensing from the perspective of IoT for personal healthcare. The survey will cover passive (i.e., battery-less) devices in the UHF band (860–960 MHz) which are capable to provide services and enough read ranges to implement a network of sensors for tracking the human wellness and monitoring the quality of the local environment. The discussion will cover both the physical issues and the signal processing, up to the application level.

A RFID-powered environment supporting new pervasive healthcare services could be a Smart-House equipped with a distributed network of readers, enforcing a uniform and robust coverage in the most relevant spaces, and an heterogeneous set of battery-less tags with sensing capability. Readers may interact, for instance, through WiFi or Bluetooth links, with a concentrator node enabling the interconnection with external services that take care of data processing and of assistance procedures' activation. Ambient sensors, able to detect physical parameters of the environment, such as the temperature, humidity, and the presence of toxic agents, could permit to quantify the wellness of the environment itself and to analyze its correlation with the person's health state. Wearable and implantable tags could instead permit to provide indirect information about the presence of a person inside a room, his motion, the interaction among people and in more futuristic body-centric applications, to produce data about the health state of prosthesis or artificial organs.

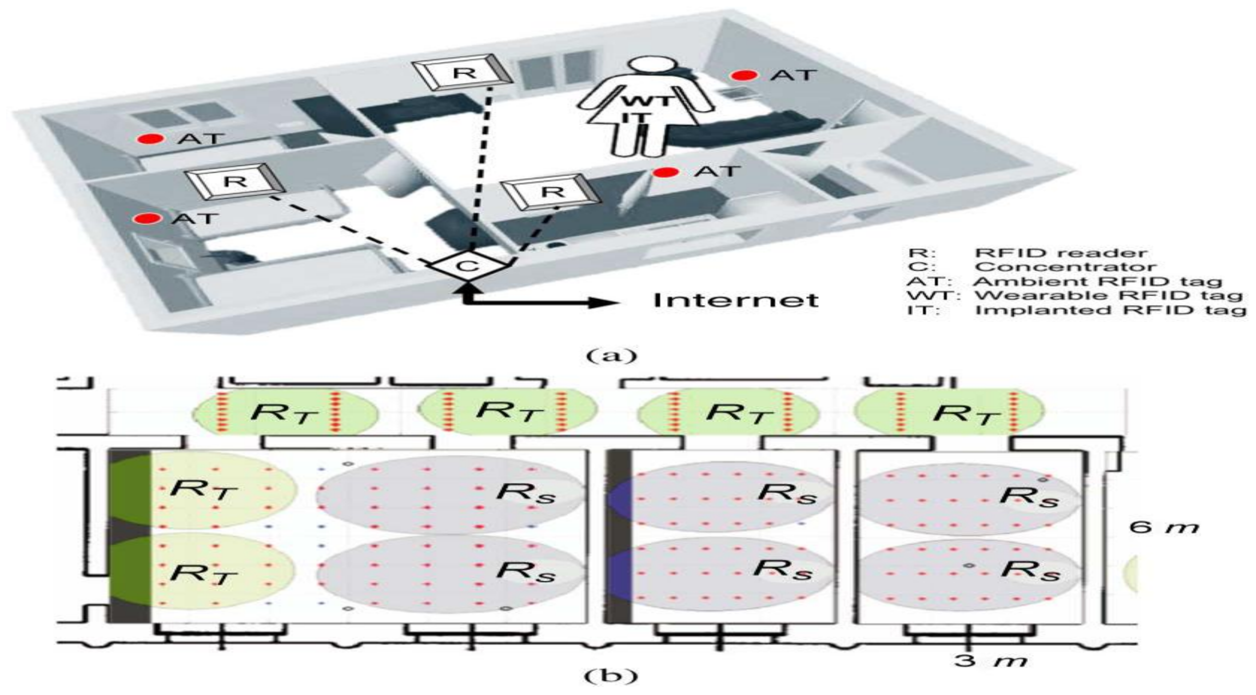


Figure 7 : Sketch of an RFID-powered habitat with various tags

Data gathered from the RFID environment, e.g., generated by ambient tags or by the interaction of the user with the house, could be finally processed by means of data-mining algorithms to analyze the movements and trajectories of the user, classify its gestures in daily and sleep conditions, recognize critical events and emit alarms. Finally, the whole technologic infrastructure has to be compliant with the electromagnetic safety standard concerning the power absorption in the human body and the emitted field strength in living environments.

By combining together wearable tags and ambient tags it is possible to develop a fully passive RFID system to monitoring the state of children, disabled and elderly people in domestic and hospital environment during the night: an Ambient Intelligence platform, capable to estimate the sleep parameters, to classify the human activity and to identify abnormal events that require immediate assistance.

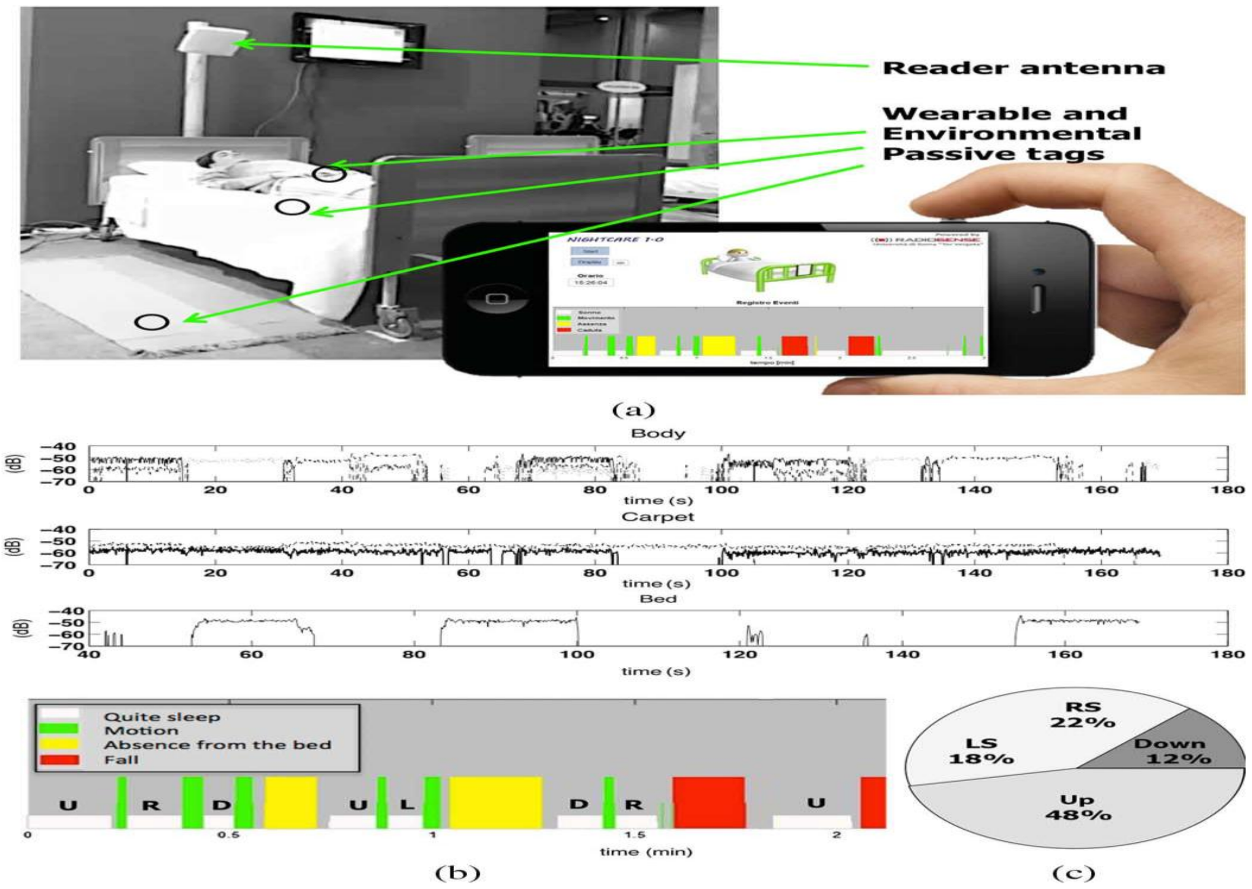


Figure 8 : Ambient intelligence system aimed to take care of the night sleep involving RFID tags placed over the body and in the surrounding environment.

CHAPTER 3 : PROBLEM STATEMENT

The problem statement is improve the healthcare scenario i.e. to provide hospitals and doctors 24 Hour tracking of multiple patient's health record.

The aim of this project is to design a system which would connect and collect data information through health sensors which would include patient's heart rate, temperature and to send it on the server from where the patient and doctor can track the health status of patient anytime.

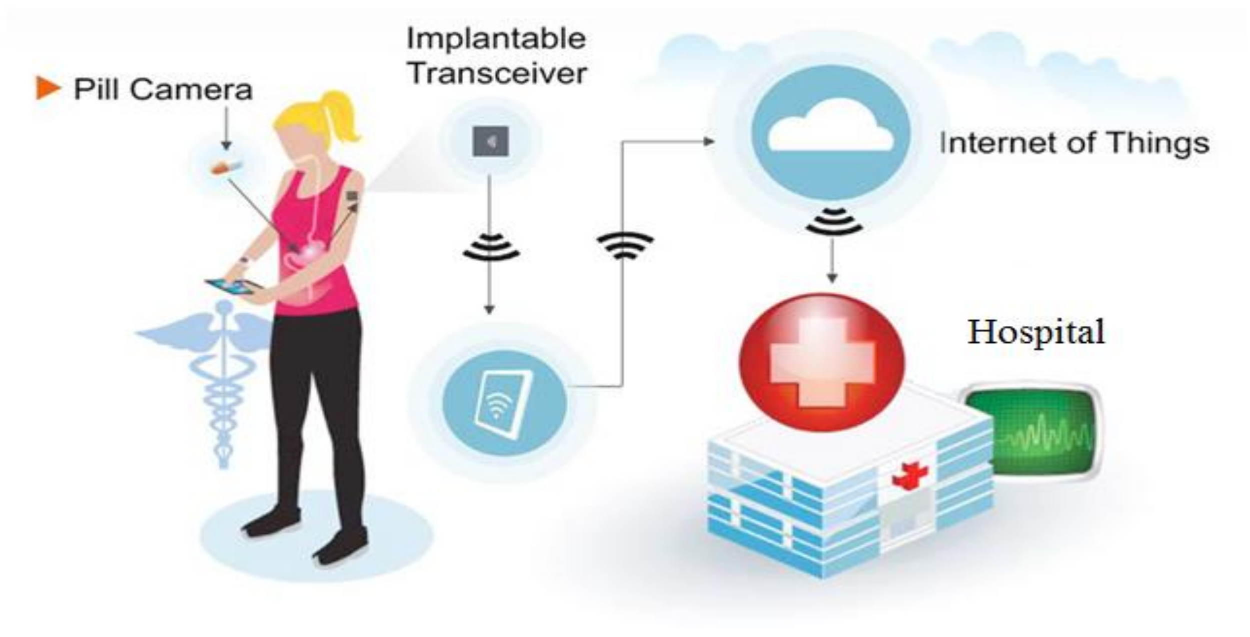
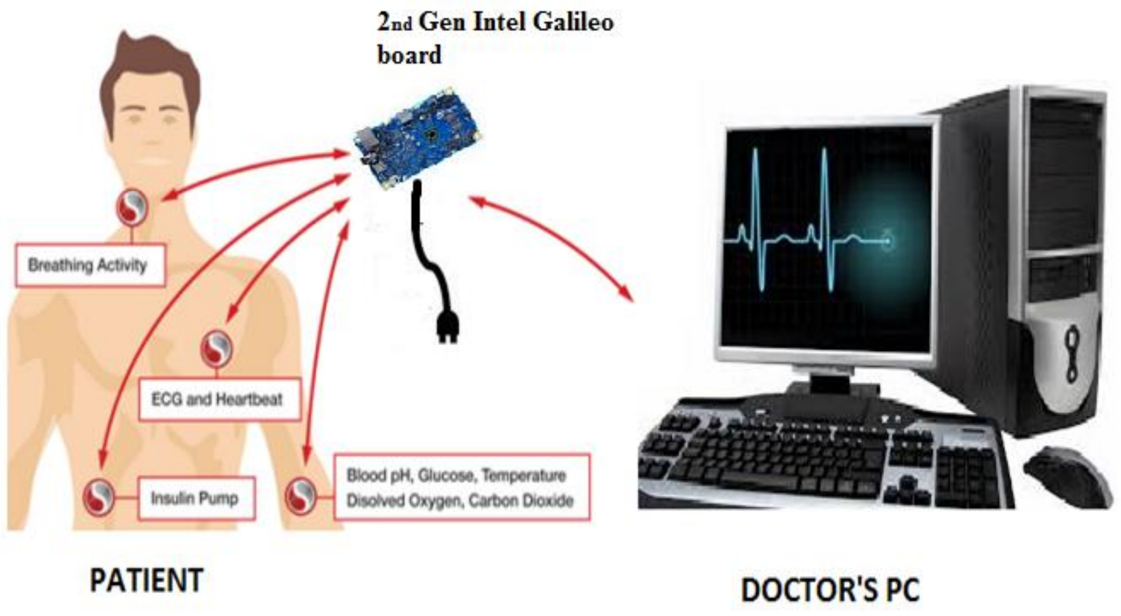


Figure 9 : Patient Monitoring System

CHAPTER 4 : PROPOSED MODEL

4.1 Project Introduction

In our day to day activities we are introduced to many intelligent to many intelligent systems and one of them would be this i.e. a system capable enough to monitor the patient automatically using IoT that will accurately collects data information through health sensors which would include patient's heart rate, temperature and to send it on the web server so that patient's doctor can see patient's current status and full medical information. It will help the doctor to monitor his patient from anywhere and also to the patient to check his health status directly without visiting to the hospital.

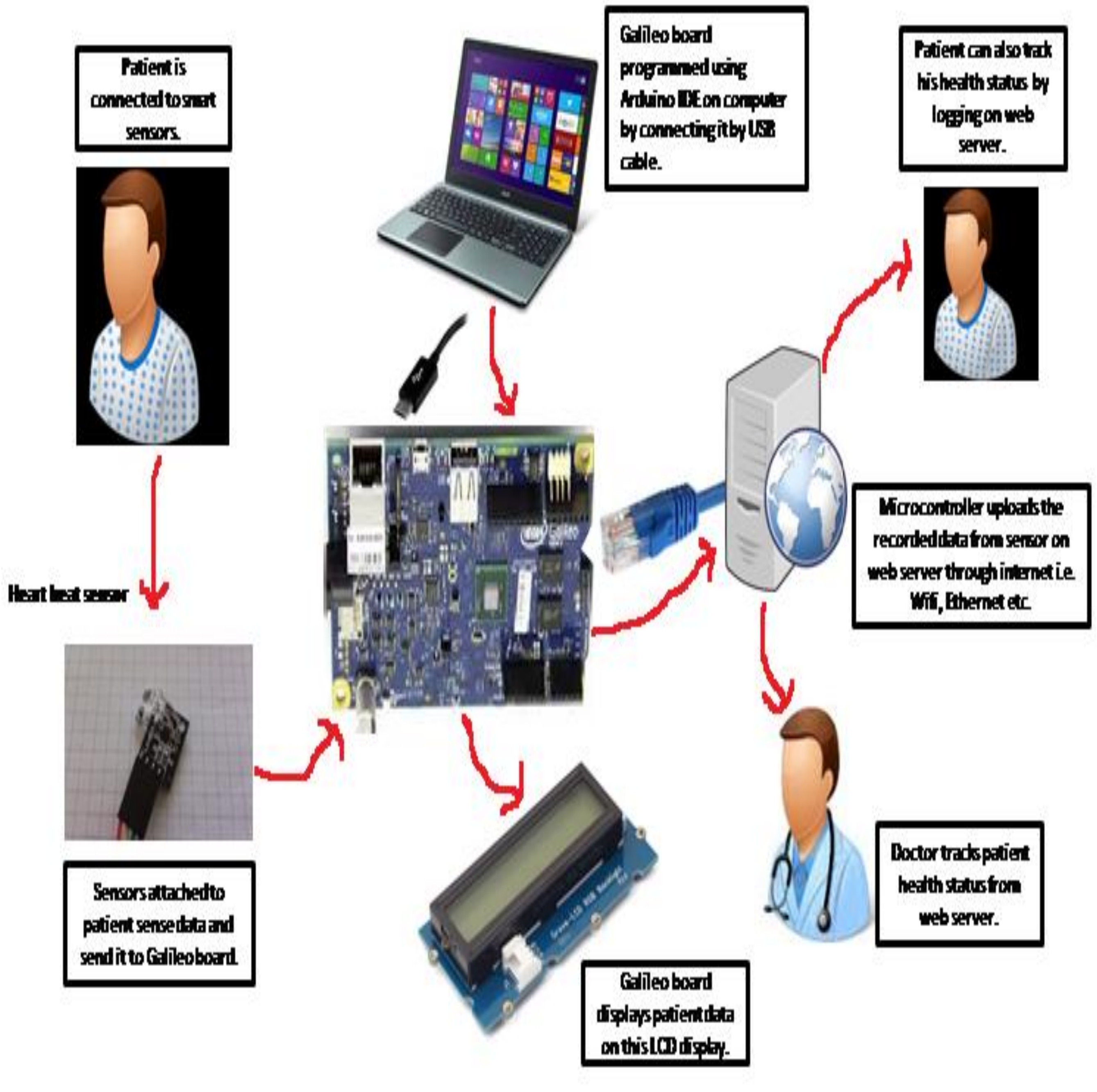


2nd Gen Intel Galileo board
Breathing Activity
ECG and Heartbeat
Insulin Pump
Blood pH, Glucose, Temperature Disolved Oxygen, Carbon Dioxide
PATIENT
DOCTOR'S PC

Figure 10 : Proposed Model

4.1.1 System Architecture

The following architecture diagram will help you to understand the system more efficiently.



Patient is connected to smart sensors.
Galileo board programmed using Arduino IDE on computer by connecting it by USB cable.
Patient can also track his health status by logging on web server.
Heart beat sensor
Microcontroller uploads the recorded data from sensor on web server through internet i.e. Wifi, Ethernet etc.
Sensors attached to patient sense data and send it to Galileo board.
Doctor tracks patient health status from web server.
Galileo board displays patient data on this LCD display.

Figure 11 : System Architecture

4.2 Hardware Requirements

The major hardware requirements required for the project are:

I. Smart Sensors
II. Intel Galileo Board Gen 2
III. Bread Board
IV. RGB LCD Display
V. Grove – Base Shield

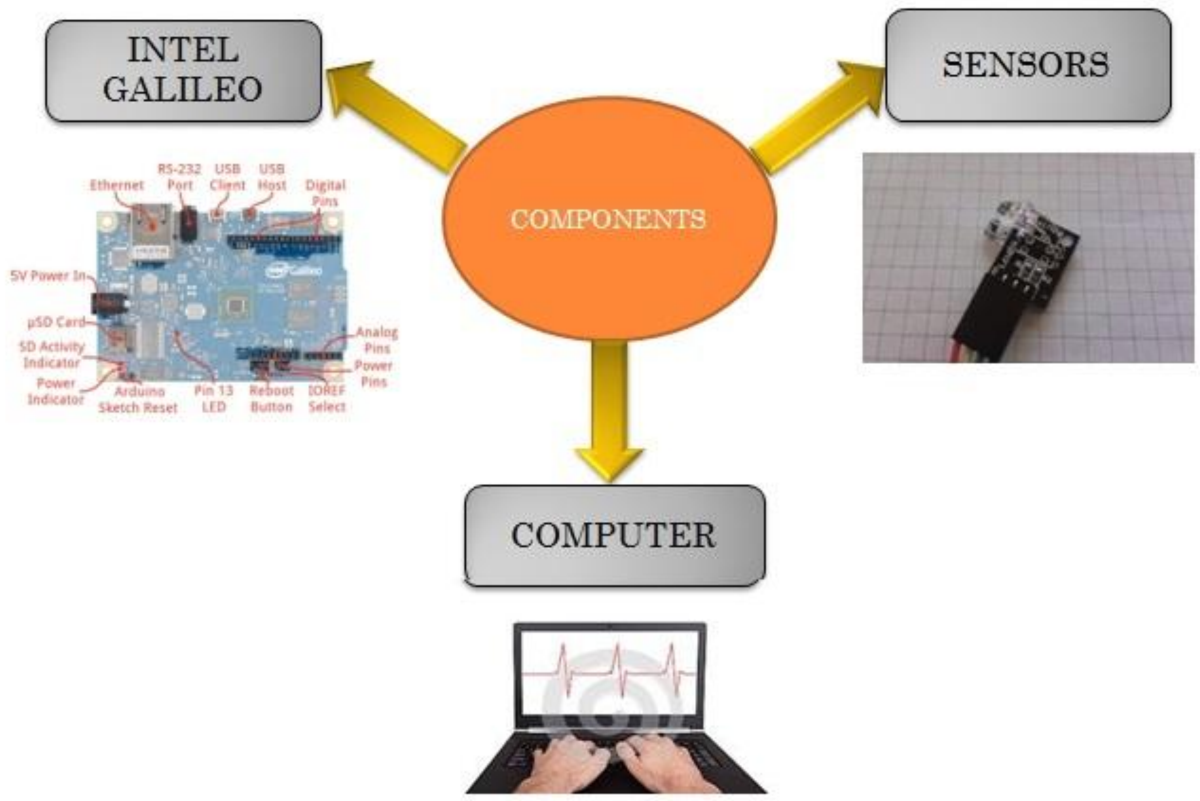


Figure 11 : Components

Smart Sensors : A **sensor** is a device that detects events or changes in quantities and provides a corresponding output, generally as an electrical or optical signal; for example, a thermocouple converts temperature to an output voltage. But a mercury-in-glass thermometer is also a sensor; it converts the measured temperature into expansion and contraction of a liquid which can be read on a calibrated glass tube.

Sensors are used in everyday objects such as touch-sensitive elevator buttons (tactile sensor) and lamps which dim or brighten by touching the base, besides innumerable applications of which most people are never aware. With advances in micro machinery and easy-to-use microcontroller platforms, the uses of sensors have expanded beyond the more traditional fields of temperature, pressure or flow measurement, for example into MARG sensors. Moreover, analog sensors such as potentiometers and force-sensing resistors are still widely used. Applications include manufacturing and machinery, airplanes and aerospace, cars, medicine and robotics.

A sensor's sensitivity indicates how much the sensor's output changes when the input quantity being measured changes. For instance, if the mercury in a thermometer moves 1 cm when the temperature changes by 1 °C, the sensitivity is 1 cm/°C (it is basically the slope Dy/Dx assuming a linear characteristic). Some sensors can also have an impact on what they measure; for instance, a room temperature thermometer inserted into a hot cup of liquid cools the liquid while the liquid heats the thermometer. Sensors need to be designed to have a small effect on what is measured; making the sensor smaller often improves this and may introduce other advantages. Technological progress allows more and more sensors to be manufactured on a microscopic scale as micro sensors using MEMS technology. In most cases, a micro sensor reaches a significantly higher speed and sensitivity compared with macroscopic approaches.

A good sensor obeys the following rules :

- Is sensitive to the measured property only
- Is insensitive to any other property likely to be encountered in its application
- Does not influence the measured property

The sensitivity is then defined as the ratio between output signal and measured property. For example, if a sensor measures temperature and has a voltage output, the sensitivity is a constant with the unit [V/K]; this sensor is linear because the ratio is constant at all points of measurement.

For an analog sensor signal to be processed, or used in digital equipment, it needs to be converted to a digital signal, using an analog-to-digital converter.

Sensors that I am going to use in my project are:

- Heart Beat Sensor
- Temperature Sensor

1. Heart beat sensor

Heart beat sensor is designed to give digital output of heat beat when a finger is placed on it.

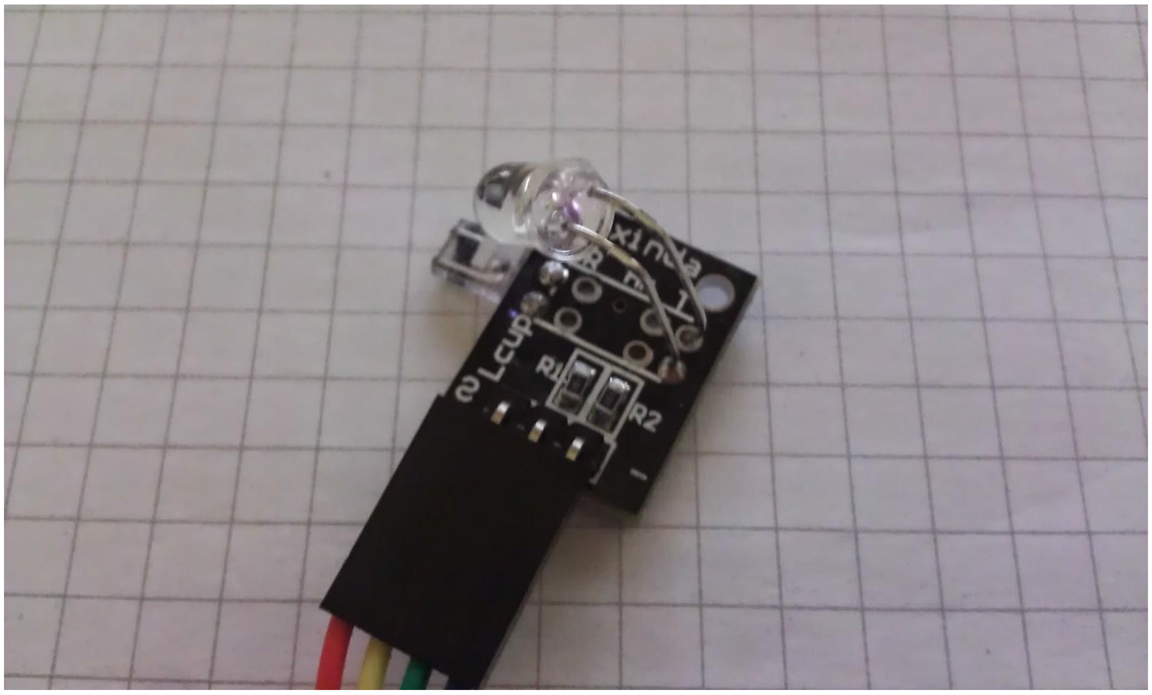


Figure 12 : HeartBeat Sensor

The sensor has 3 pins which are labelled very good. One of them has a -, here you have to connect GND. The next pin has an S, this is the pin for the sensor output. The last pin (in the middle) which has no label is connected to the LED. This pin has to be connected to +5V and the sensor output has to be connected to an analog input of the Galileo board.

2. Temperature sensor

The temperature sensor Grove uses a thermistor which returns the ambient temperature in the form of a resistance value, which is then used to alter Vcc. This board then converts this voltage value measured by an analog input pin to a temperature. The operating range is -40 to 125 degrees Celsius, with an accuracy of ±1.5ºC.

Figure 13 : Grove Temperature Sensor

Grove Temperature Sensor Schematic :

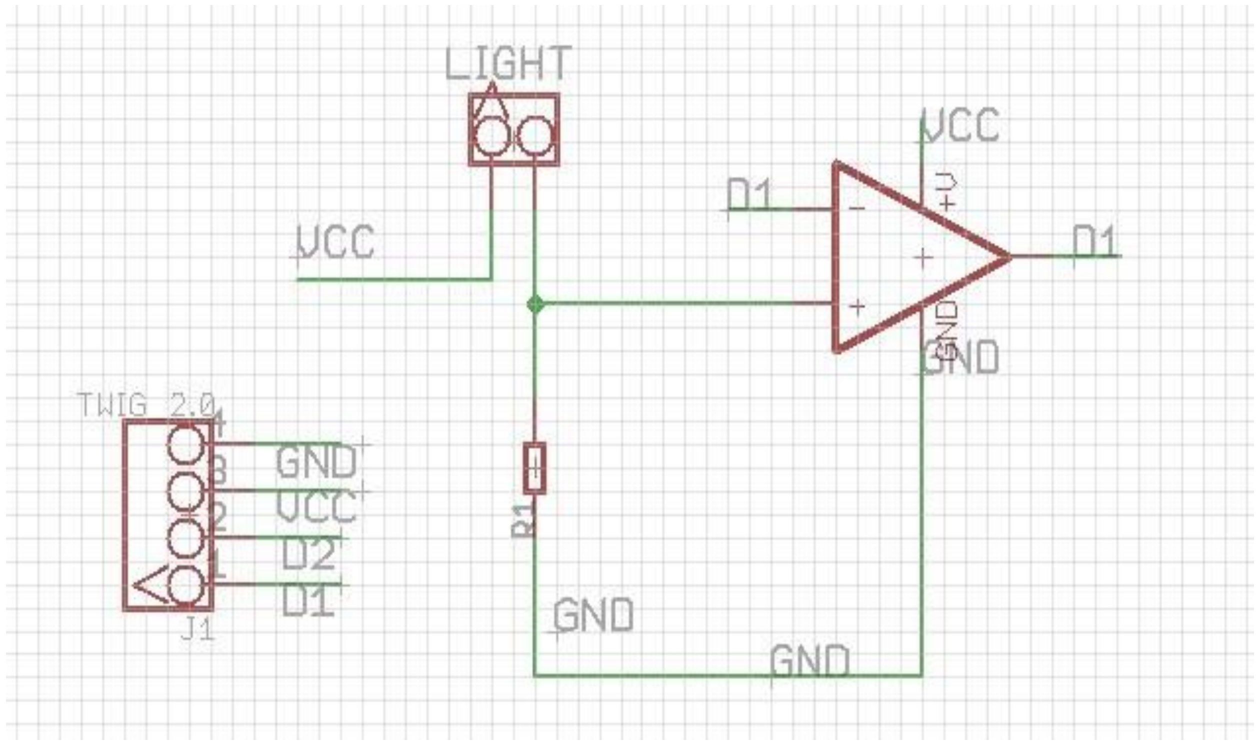


Figure : Schematic of Grove Temperature sensor

RGB LCD Display : It is done with tedious mono color backlight? This Grove enables you to set the color to whatever you like via the simple and concise Grove interface. It takes I2C as communication method with your microcontroller. So number of pins required for data exchange and backlight control shrinks from ~10 to 2, relieving IOs for other challenging tasks. Besides, Grove - LCD RGB Backlight supports user-defined characters.

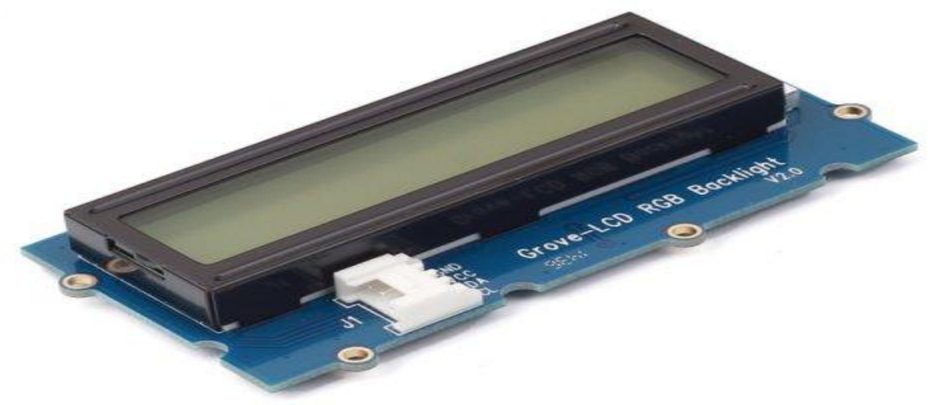


Figure 14 : Grove RGB LCD Display

Specifications:

- Input Voltage:5V
- Operating Current<60mA
- CGROM: 10880 bit
- CGRAM: 64*8 bit
- Colorful RGB Backlight
- Built-in English and Japanese fonts
- I2C communication, uses only two IOs
- Automatic power-on reset

Grove – Base Shield : Base Shield v2 has many Grove connectors, making it convenient for you to use Grove products together.

Power Compatible: Every Grove connector has four wires, one of which is Vcc. However, not every micro-controller main board needs a supply voltage of 5V, some need 3.3V. That's why we add a power switch to Base Shield v2. In this way, you can adjust the voltage of Vcc via this switch, making sure the voltage of Vcc is the same as supply power of the main board.

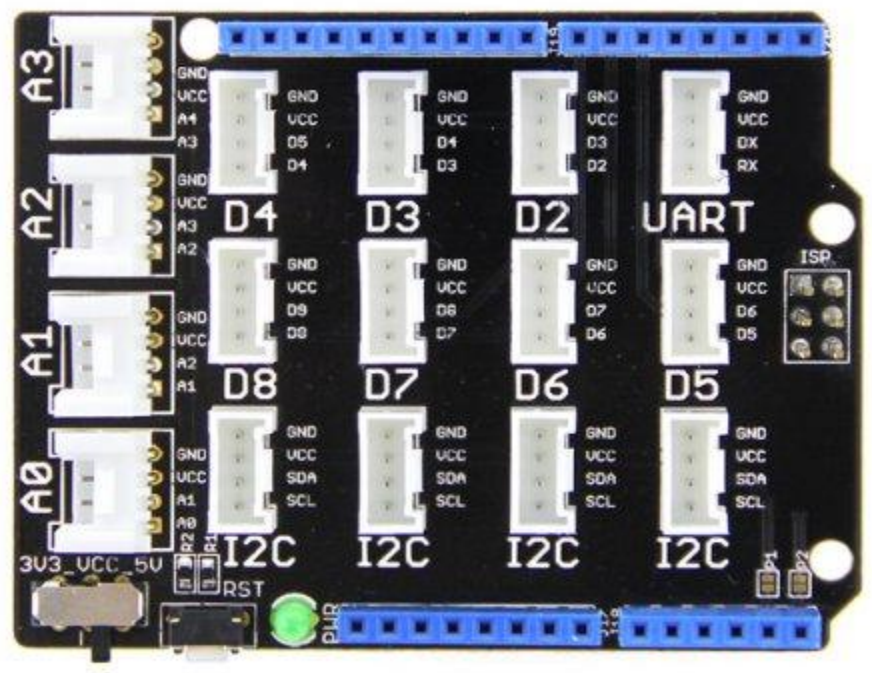


Figure 15 : Grove – Base Shield

Intel Galileo board 2nd Gen: Intel Galileo features the Intel Quark SoC X1000, the first product from the Intel Quark technology family of low-power, small-core products. Intel Quark represents Intel's attempt to compete within markets such as the Internet of Things and wearable computing. Designed in Ireland, the Quark SoC X1000 is a 32-bit, single core, single-thread, Pentium (P54C/i586) instruction set architecture (ISA)-compatible CPU, operating at speeds up to 400 MHz.

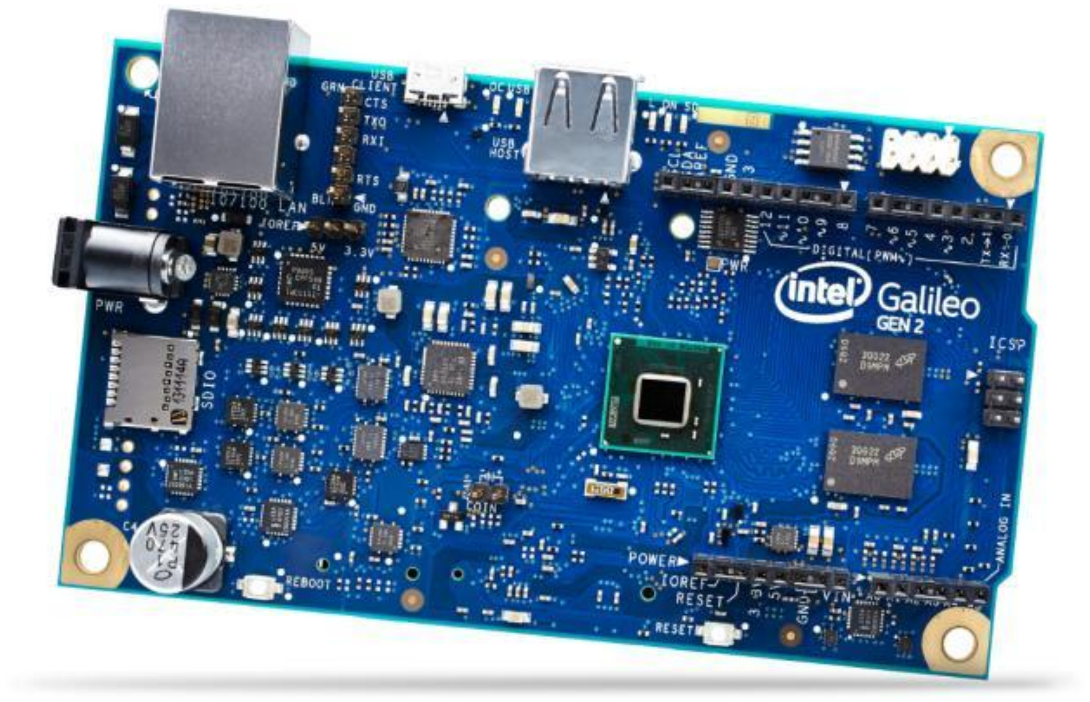


Figure 16 : The **Intel® Galileo Gen 2 board**

2.3 Software Requirements

This project needs:

- An Arduino SDK (version 1.5.3) as the SDK for the programming of the 2nd Gen Intel Galileo board.
- Wampp server for database.

1. Arduino SDK

The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and derives from the IDE for the Processing programming language and the wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a sketch.

Arduino programs are written in C or C++. The Arduino IDE comes with a software library called "Wiring" from the original Wiring project, which makes many common input/output operations much easier. Users only need define two functions to make a runnable cyclic executive program:

- setup( ): A function run once at the start of a program that can initialize settings
- loop( ): A function called repeatedly until the board powers off

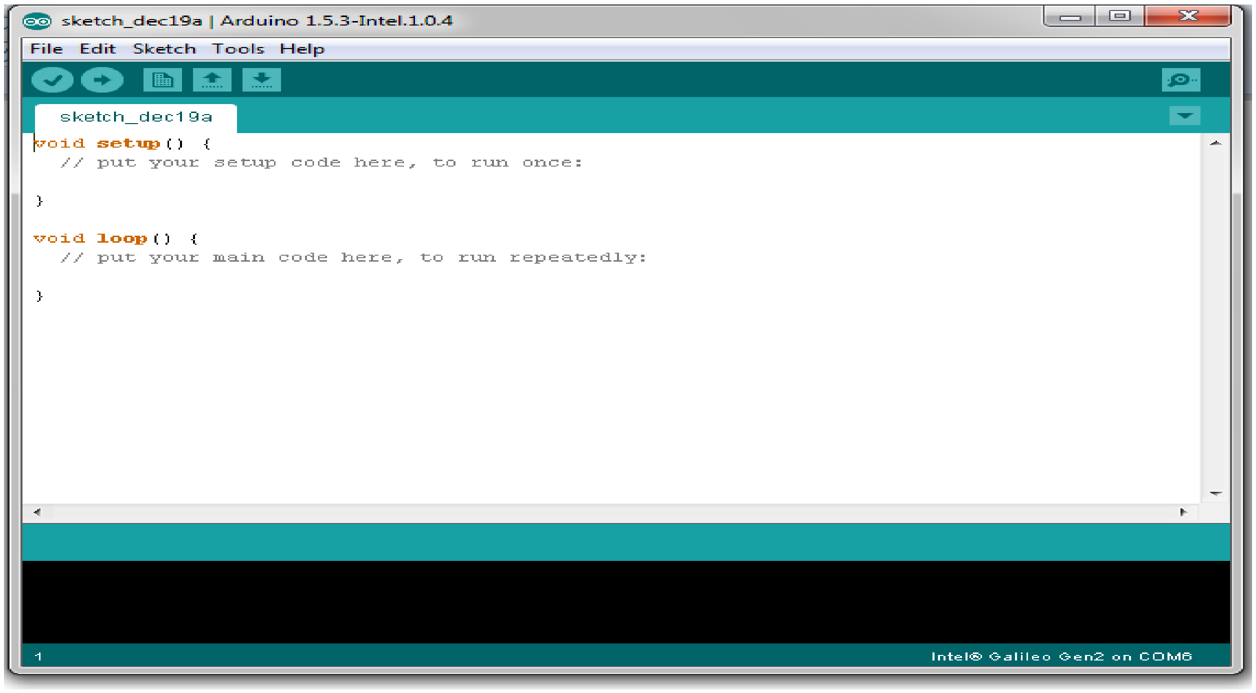
The Arduino IDE uses the GNU tool chain and AVR Libc to compile programs, and uses avrdude to upload programs to the board. Arduino SDK is a software application that provides comprehensive facilities to computer programmers for software development. We will be using Arduino version 1.5.3 which will support Intel Galileo Gen 2 board.

Modes of Communication

There are two modes of communication:

I. *Serial:* Used for communication between the Galileo board and a computer or other devices. It communicates on digital pins 0 (RX) and 1 (TX) as well as with the computer via USB. Thus, if we use these functions, we cannot also use pins 0 and 1 for digital input or output. We can use the Arduino environment's built-in serial monitor to communicate with an Galileo board.

II. *Stream:* Stream is the base class for character and binary based streams. It is not called directly, but invoked whenever you use a function that relies on it. Stream defines the reading functions in Arduino. When using any core functionality that uses a read() or similar method, you can safely assume it calls on the Stream class.


sketch_dec19a | Arduino 1.5.3-Intel.1.0.4
File Edit Sketch Tools Help
sketch_dec19a
1
Intel® Galileo Gen2 on COM6

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Figure 17 : Front view of Arduino SDK

CHAPTER 5 : SIMULATION AND RESULT

5.1 HARDWARE IMPLEMENTATION

1. Blinking Led

To blink the LED takes only a few lines of code. The first thing we do is define a variable that will hold the number of the pin that the LED is connected to. We don't have to do this (we could just use the pin number throughout the code) but it makes it easier to change to a different pin. We use an integer variable).

The second thing we need to do is configure as an output the pin connected to the LED. We do this with a call to the pinMode() function, inside of the sketch's setup() function.

Finally, we have to turn the LED on and off with the sketch's loop() function. We do this with two calls to the digitalWrite() function, one with HIGH to turn the LED on and one with LOW to turn the LED off. If we simply alternated calls to these two functions, the LED would turn on and off too quickly for us to see, so we add two calls to delay() to slow things down. The delay function works with milliseconds, so we pass it 1000 to pause for a second.

Sketch:

```
int led = 13;

void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

In this example a LED is repeatedly turned ON and OFF for one second each. Here a simple circuit is implemented on a bread board and is connected with the Intel Galileo Gen 2 board which is further connected to the power supply by the adapter and also with a micro USB cable that helps us to transfer the code serially to the board.

Output:

(A) When pin is high

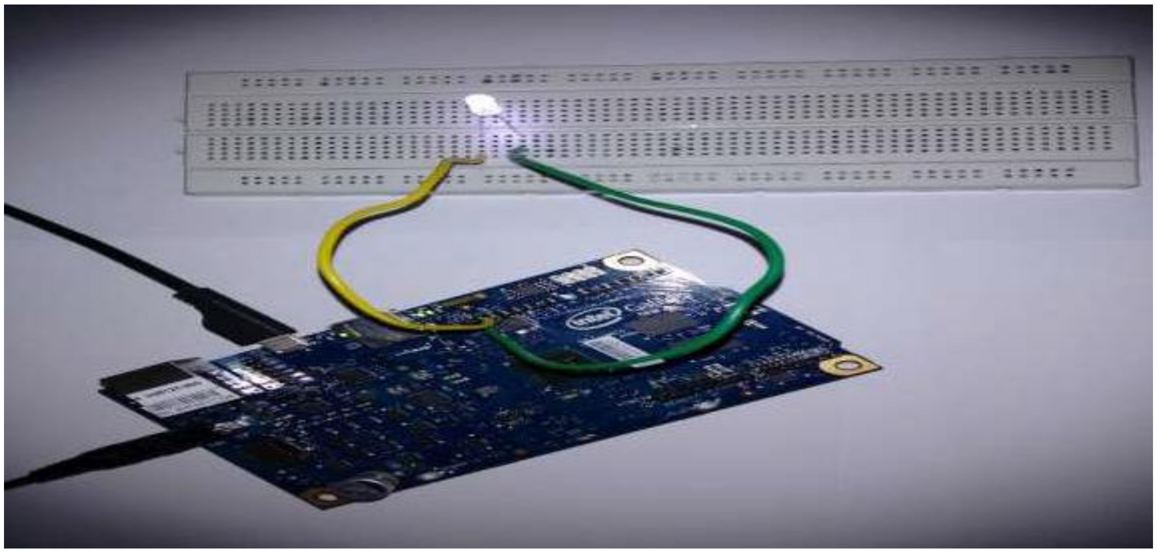


Figure 18 : Led is On

(B) When pin is low

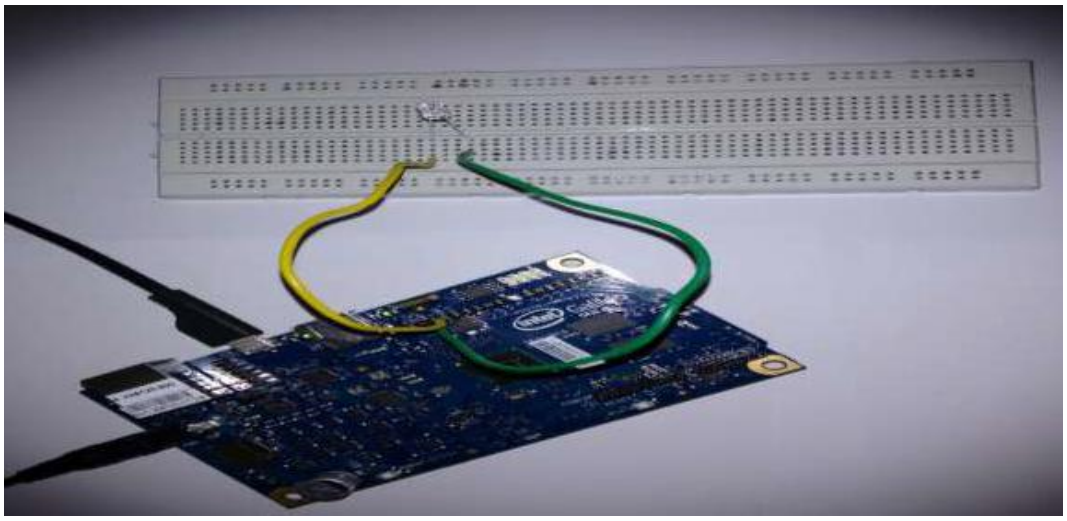


Figure 19 : Led is off

2. AnalogInput

It demonstrates analog input by reading an analog sensor on analog pin 0 and turning on and off a light emitting diode(LED) connected to digital pin 13. The amount of time the LED will be on and off depends on the value obtained by analogRead().

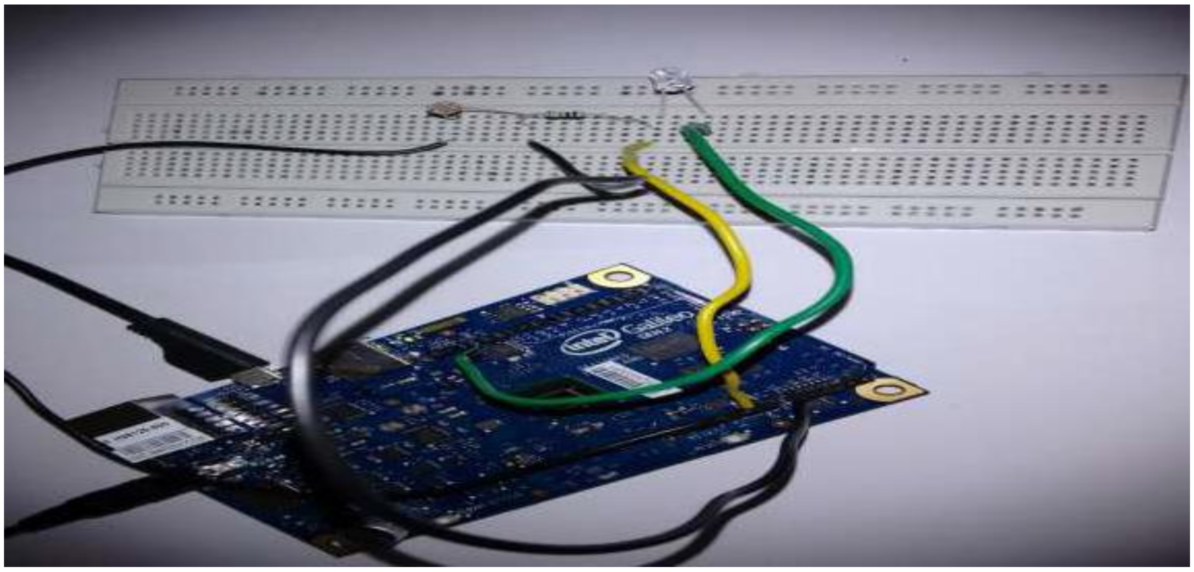
The circuit:

- Potentiometer attached to analog input 0
- center pin of the potentiometer to the analog pin
- one side pin (either one) to ground
- the other side pin to +5V
- LED anode (long leg) attached to digital output 13
- LED cathode (short leg) attached to ground

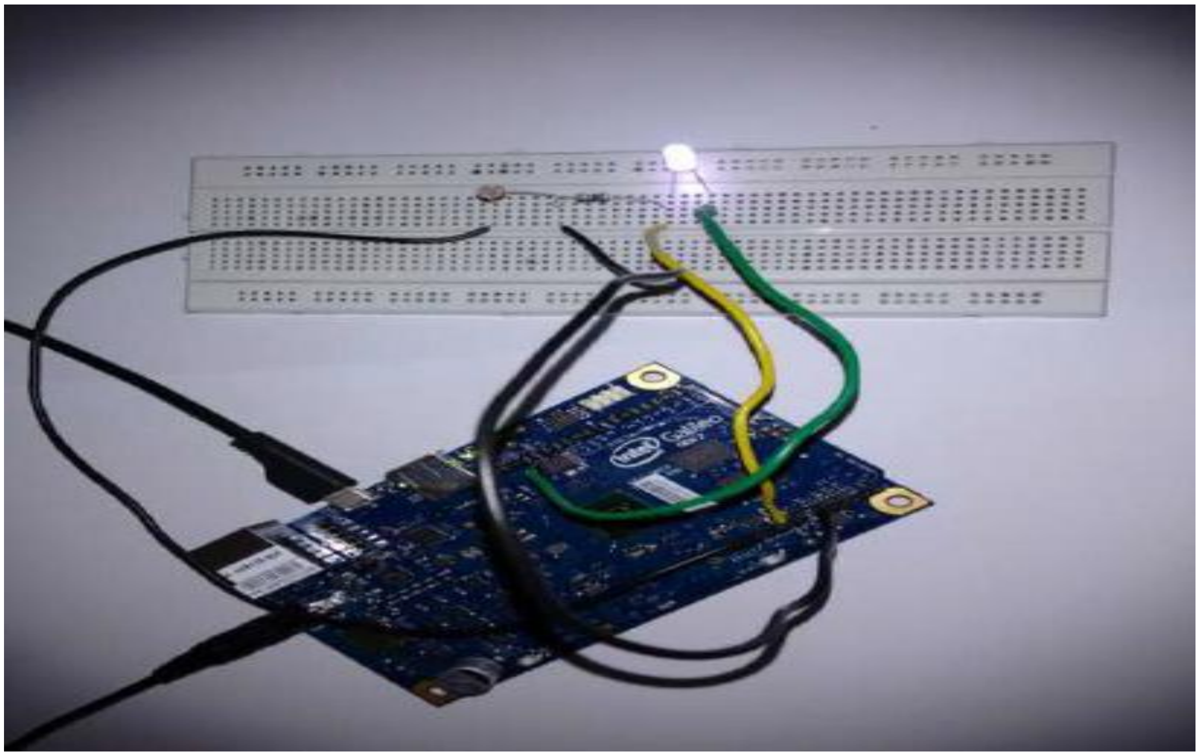
Sketch:

```
int sensorPin = A0;    // select the input pin for the potentiometer
int ledPin = 13;      // select the pin for the LED
int sensorValue = 0;  // variable to store the value coming from the sensor

void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on
  digitalWrite(ledPin, HIGH);
  // stop the program for <sensorValue> milliseconds:
  delay(sensorValue);
  // turn the ledPin off:
  digitalWrite(ledPin, LOW);
  // stop the program for for <sensorValue> milliseconds:
  delay(sensorValue);
}
```

OUTPUT:



(a) When value of sensor is high



(b) When the value of sensor is low.

Figure 20 : Output of Analog Input

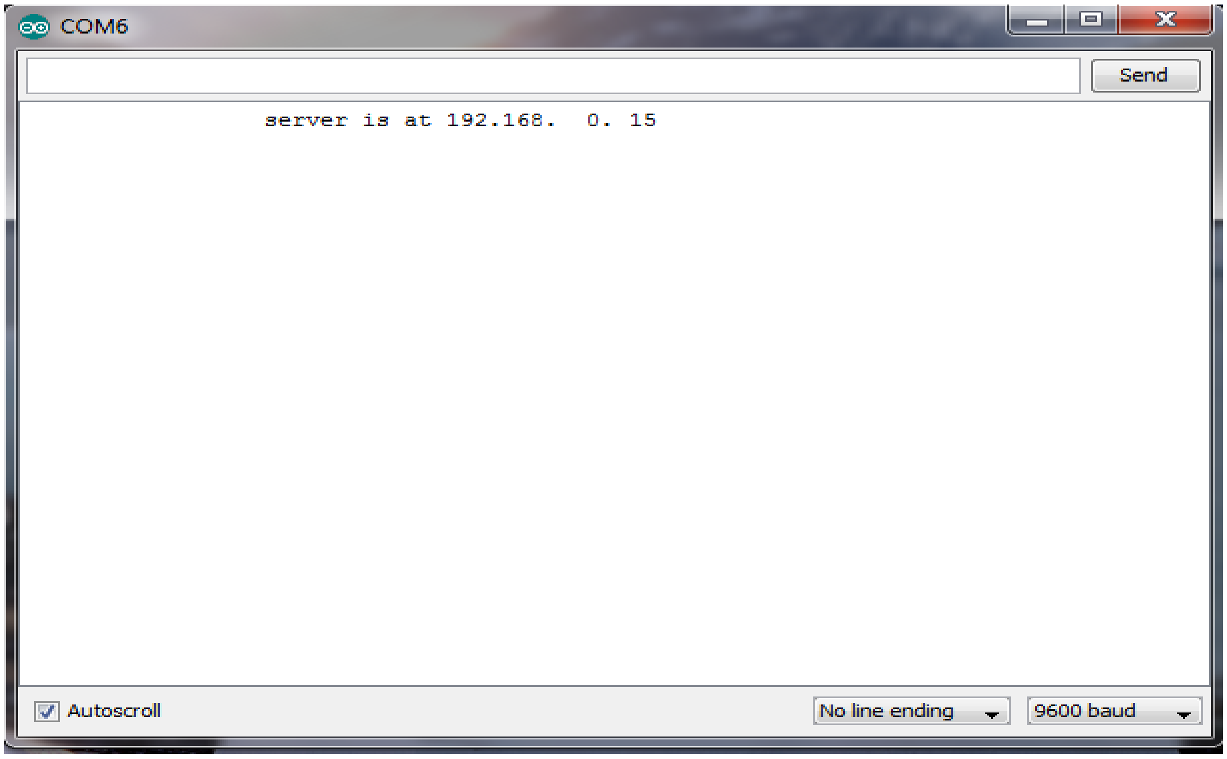
3. Web Server

A simple web server provides the ip address to controller.

Sketch:

```
#include <SPI.h>
#include <Ethernet.h>
byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress ip(192,168,0,15);
// Initialize the Ethernet server library
// with the IP address and port you want to use
// (port 80 is default for HTTP):
EthernetServer server(80);
void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
   while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }
  // start the Ethernet connection and the server:
  Ethernet.begin(mac, ip);
  server.begin();
  Serial.print("server is at ");
  Serial.println(Ethernet.localIP());
}
void loop() {
  // listen for incoming clients
  EthernetClient client = server.available();
  if (client) {
    Serial.println("new client");
    // an http request ends with a blank line
    boolean currentLineIsBlank = true;
    while (client.connected()) {
      if (client.available()) {
        char c = client.read();
        Serial.write(c);
        if (c == '\n' && currentLineIsBlank) {
          // send a standard http response header
          client.println("HTTP/1.1 200 OK");
          client.println("Content-Type: text/html");
          client.println("Connection: close");
          client.println();
          client.println("<!DOCTYPE HTML>");
          client.println("<html>");
          // add a meta refresh tag, so the browser pulls again every 5 seconds:
          client.println("<meta http-equiv=\"refresh\" content=\"5\">");
          // output the value of each analog input pin
          for (int analogChannel = 0; analogChannel < 6; analogChannel++) {
            int sensorReading = analogRead(analogChannel);
            client.print("analog input ");
            client.print(analogChannel);
            client.print(" is ");
            client.print(sensorReading);
            client.println("<br />");
          }
          client.println("</html>");
          break;
        }
        if (c == '\n') {
          // you're starting a new line
          currentLineIsBlank = true;
        }
        else if (c != '\r') {
          currentLineIsBlank = false;
        }}}
    delay(1);
        client.stop();
    Serial.println("client disonnected");
  }
```

Output on Serial Monitor :



```
server is at 192.168.  0. 15
```

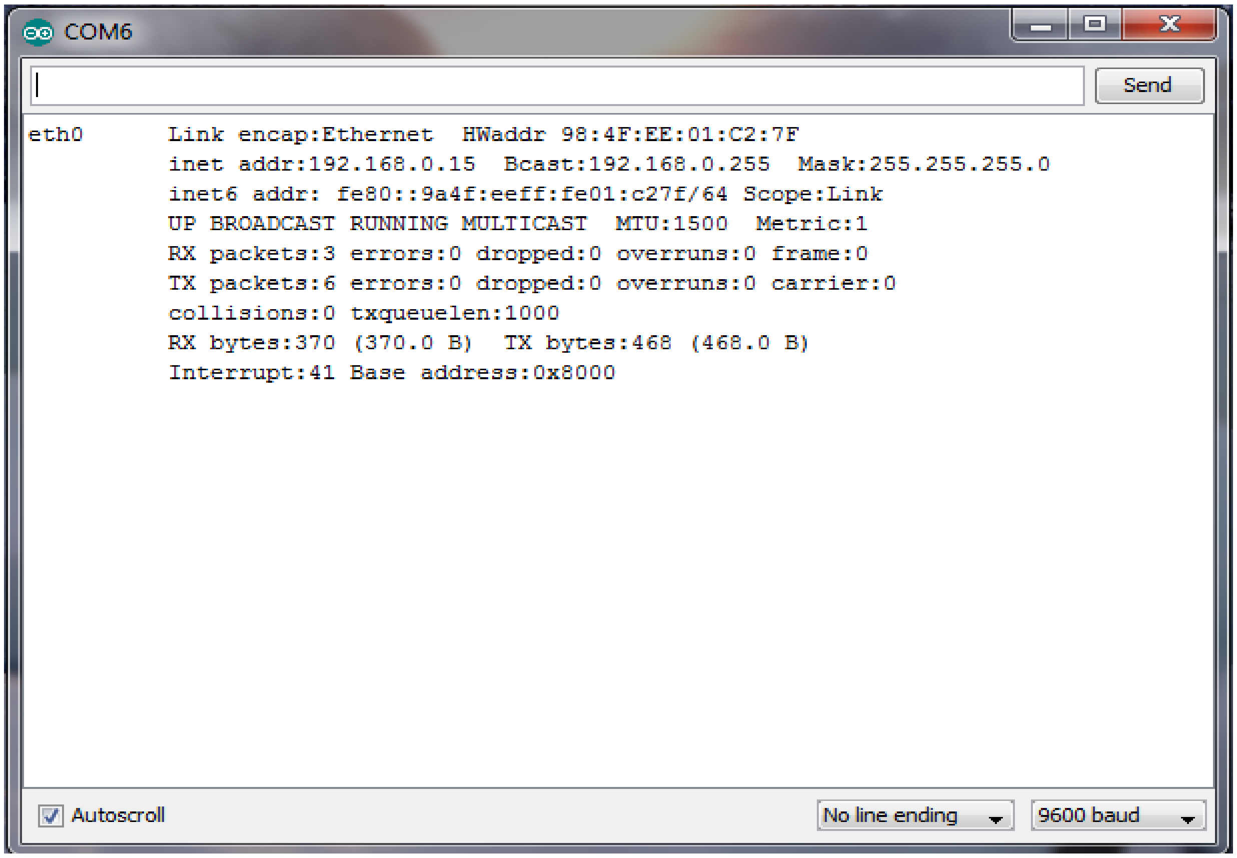
Figure 21 : Output of web server sketch

4. Telnet

Sketch:

```
void setup()
{
  system("telnetd -l /bin/sh");
}
void loop()
{
  system("ifconfig eth0 > /dev/ttyGS0");
    sleep(10);
```

Output on Serial Monitor :



```
eth0      Link encap:Ethernet  HWaddr 98:4F:EE:01:C2:7F
          inet addr:192.168.0.15  Bcast:192.168.0.255  Mask:255.255.255.0
          inet6 addr: fe80::9a4f:eeff:fe01:c27f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:370 (370.0 B)  TX bytes:468 (468.0 B)
          Interrupt:41 Base address:0x8000
```

Figure 22 : Output of Telnet sketch

This sketch provides the mac address and ip address to the controller.

5. Heart Beat Sensor

The sensor has 3 pins as you can see in below diagram. One of them has a -, which is connected to GND. The next pin has an S , this is the pin for the sensor output which is connected to analog input pin 0 of the Galileo board. The middle pin which has no label is to +5V Vcc supply.

Sketch:

```
int ledPin = 13;
int sensorPin = 0;
unsigned long lasttime;

void setup()
{
   pinMode (ledPin, OUTPUT);
  digitalWrite(ledPin,HIGH);
  Serial.begin (9600);
   lasttime = micros();
}

void loop()
{

  static double oldValue = 0;
  while(micros() - lasttime < 10000)
  {
    delayMicroseconds(300);
  }
  //Read Analog channels. You can connect accelerometer, gyro, temperature sensor etc to these channels
  int rawValue = analogRead(sensorPin);
  rawValue = rawValue-915;
    lasttime += 10000;
  //data2 = analogRead(1);
  //data3 = analogRead(2);
  //data4 = analogRead(3);

  //You can plot upto 4 channels of data. Uncomment only one of the options below

   //plot(data1,data2,data3,data4);   //Plots 4 channels of data
//  plot(data1,data2,data3);      //Plots 3 channels of data
//  plot(data1,data2);            //Plots 2 channels of data
Serial.println (rawValue);
plot(rawValue);
oldValue = rawValue;
                    //Plots 1 channel of data


  //oldValue = rawValue;
```

```
 // delay(10); //Read and plot analog inputs every 10ms.
}

//Function that takes 1 integer value and generates a packet to be sent to SimPlot.
void plot(int rawValue)
{
  int16_t comPkt[130];

  comPkt[0] = 0xCDAB;             //SimPlot packet header. Indicates start of data packet
  comPkt[1] = 2;      //Size of data in bytes. Does not include the header and size fields
  comPkt[2] = (int16_t) rawValue;

  //pktSize = 2 + 2 + (1*sizeof(int)); //Header bytes + size field bytes + data

  Serial.write((uint8_t *) comPkt, 6);
}
```

Schematic Diagram:

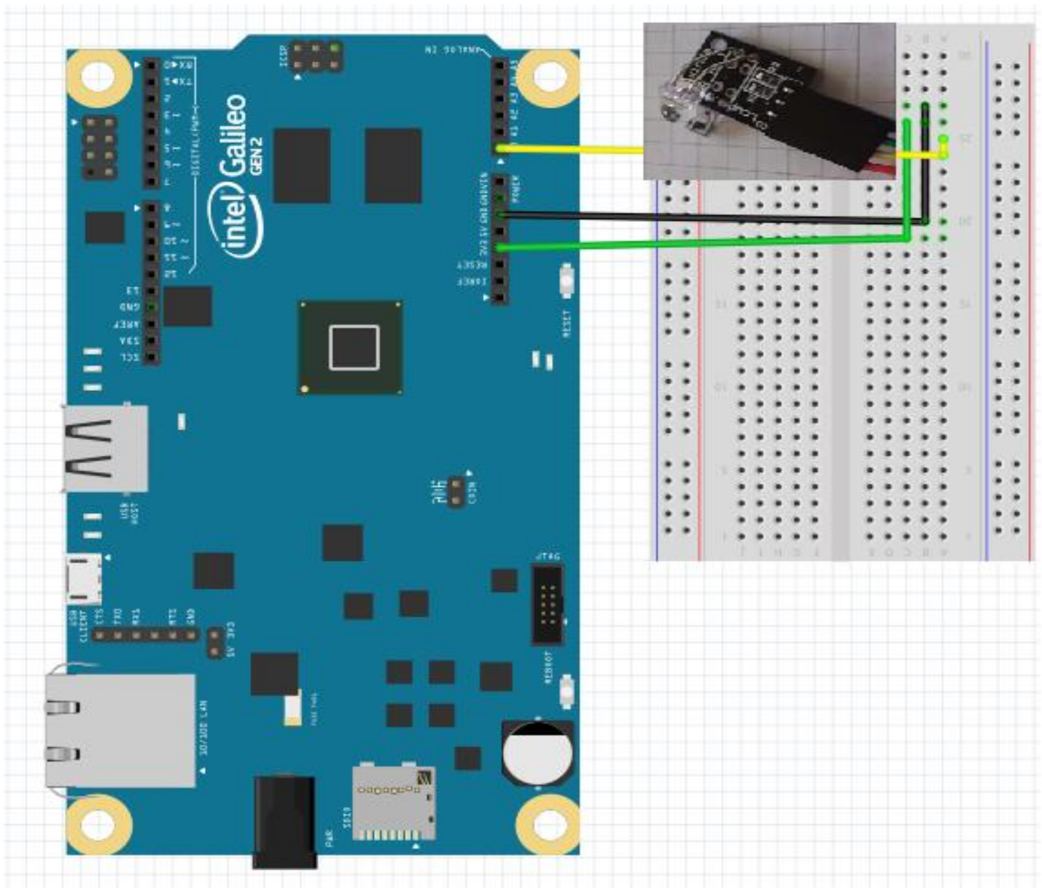


Figure 23 : Schematic diagram of Heart Beat Sensor

Circuit Diagram:

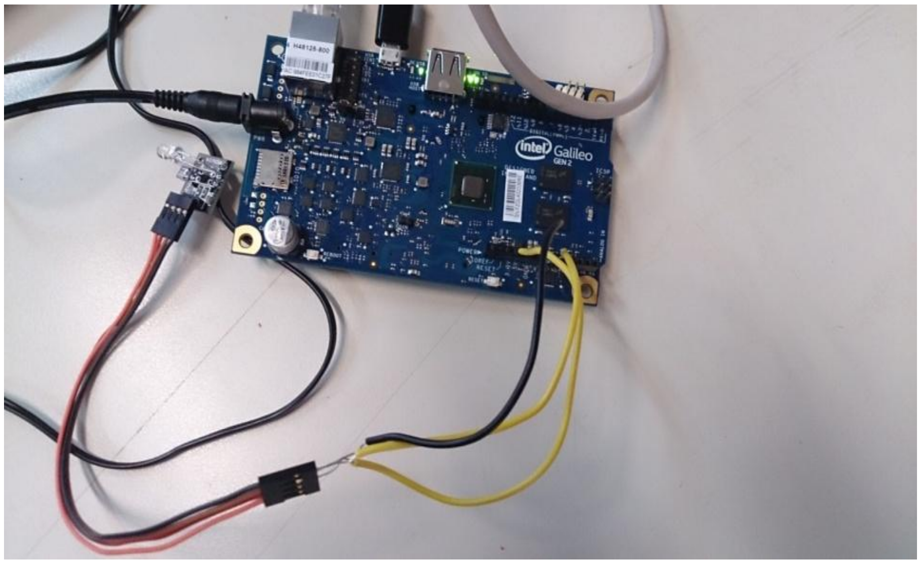
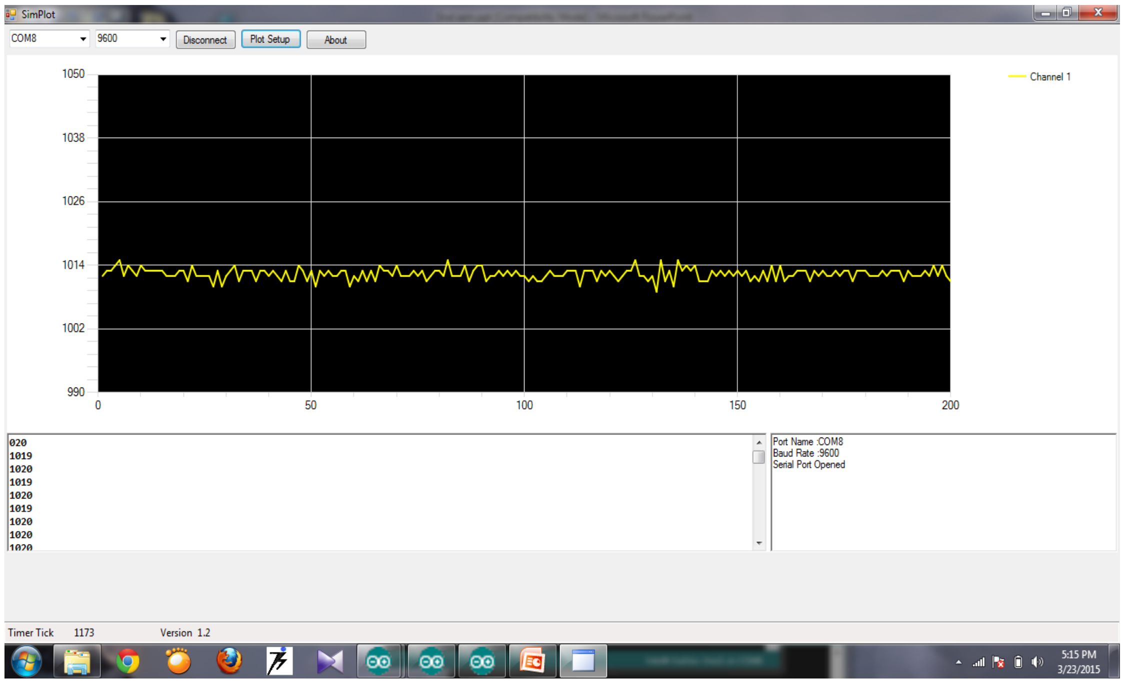


Figure 24 : Circuit diagram of Heart Beat Sensor

This figure shows how Heart Beat sensor is connected to Galileo Board.

Output on Simplot Tool:



```
020
1019
1020
1019
1020
1019
1020
1020
1020
```

```
Port Name :COM8
Baud Rate :9600
Serial Port Opened
```

Timer Tick 1173 Version 1.2

Figure 25 : Output showing real-time Heart Beat

SimPlot is a simple real time data visualization tool. This tool plots real time data from serial port. It provides 4 channels of data. Baud Rate is set to 9600 because usb port transmits data efficiently at this baud rate. Channel is set to channel 1.

6. Grove Temperature Sensor

Temperature sensor is connected to the Grove Base Shield at analog input A0 through wire connector which is attached to the Galileo board. Sensor output pin is connected to the Analog port 0 of Grove Basic Shield using the 4-pin grove cable. Plug the Grove Basic Shield into Galileo. Connect Galileo to PC by using a USB cable.

Schematic Diagram :

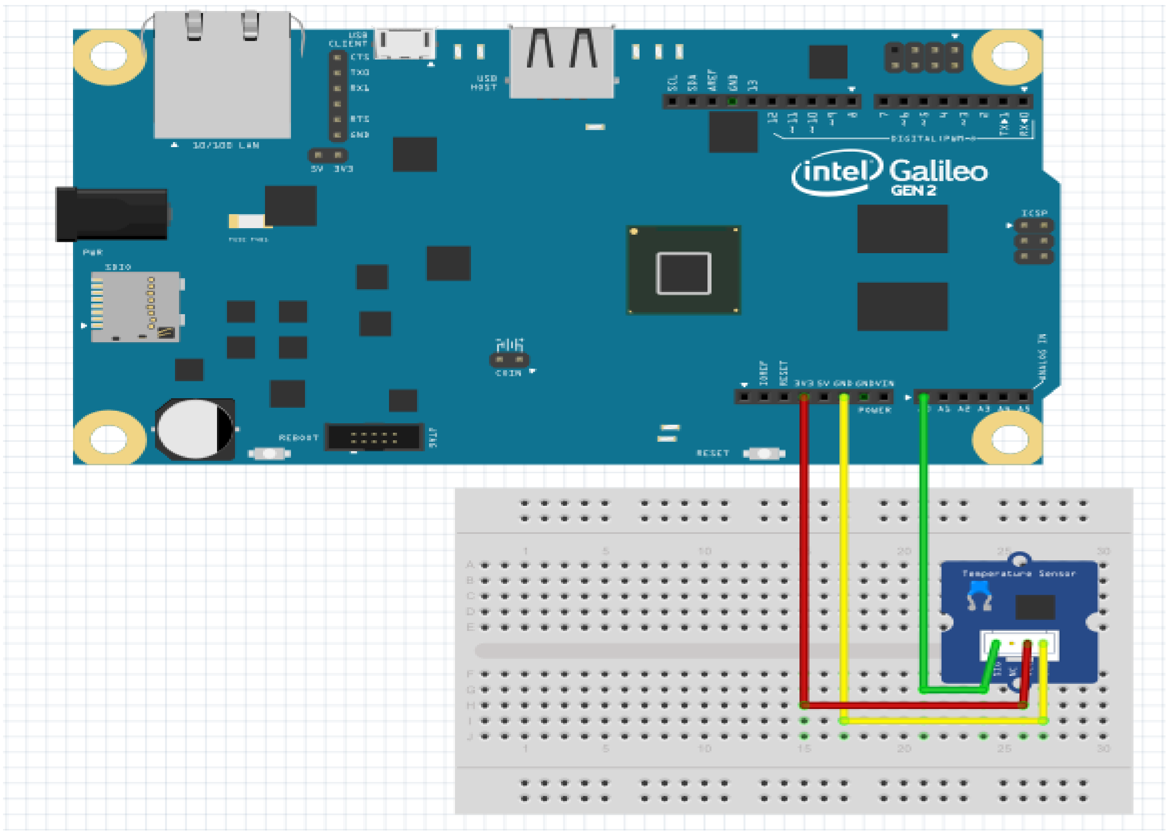


Figure 26 : Schematic diagram of Temperature Sensor

Circuit Diagram :

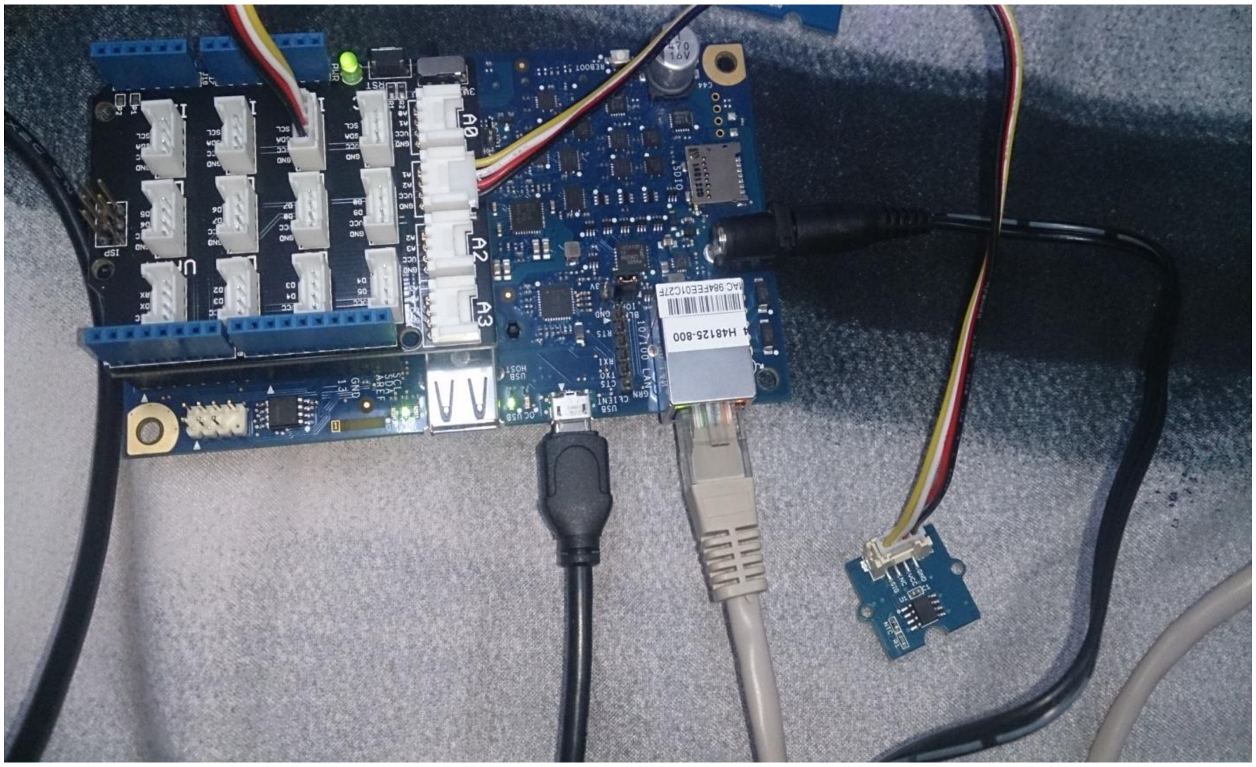
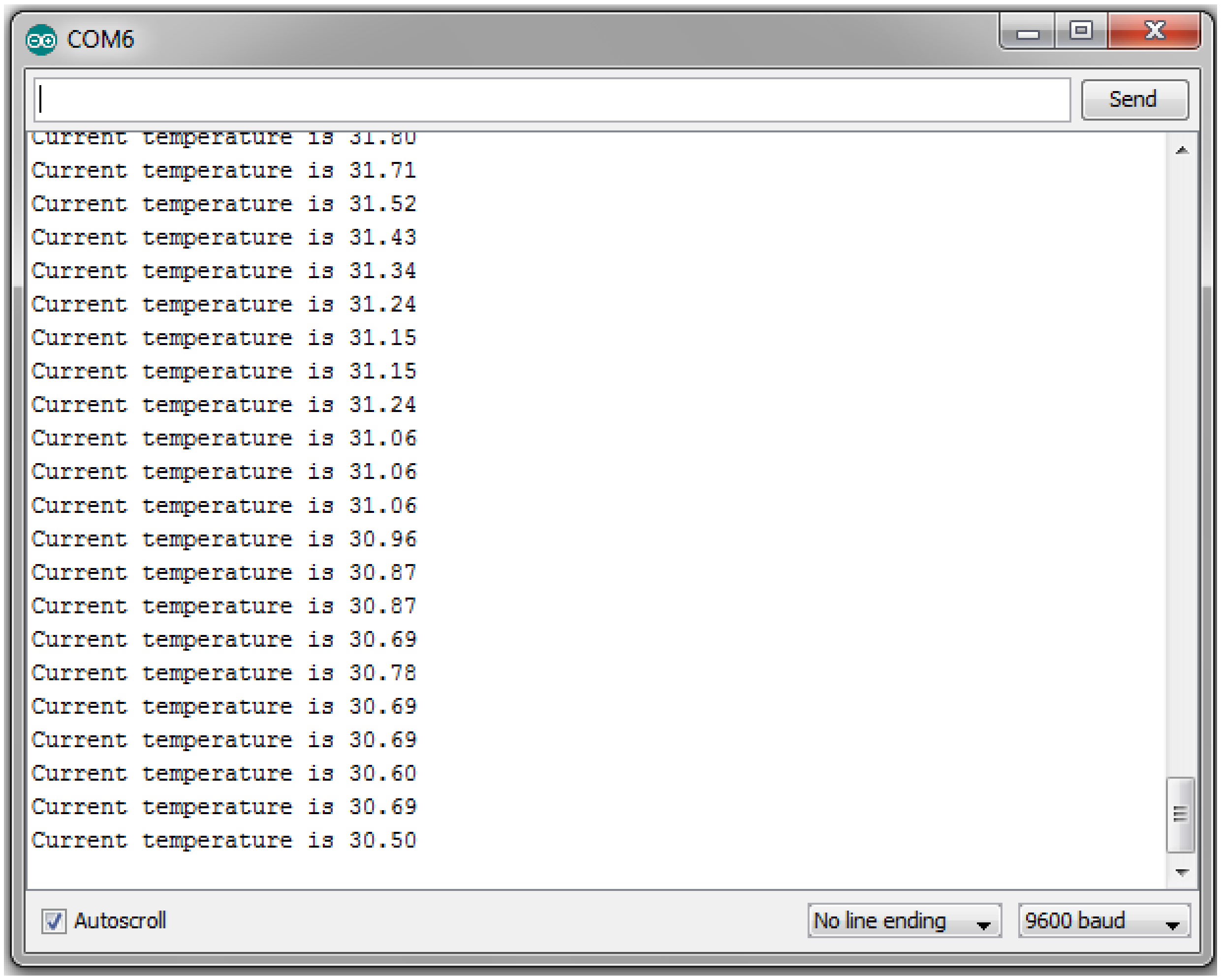


Figure 27 : Circuit diagram of Temperature Sensor

Sketch:

```
#include <math.h>
int a;
float temperature;
int B=3975;                  //B value of the thermistor
float resistance;
void setup()
{
  Serial.begin(9600);
}
void loop()
{
  a=analogRead(0);
  resistance=(float)(1023-a)*10000/a; //get the resistance of the sensor;
  temperature=1/(log(resistance/10000)/B+1/298.15)-273.15;//convert to temperature via datasheet ;
  delay(1000);
  Serial.print("Current temperature is ");
  Serial.println(temperature);
 }
```

Output on Serial Monitor :



```
Current temperature is 31.80
Current temperature is 31.71
Current temperature is 31.52
Current temperature is 31.43
Current temperature is 31.34
Current temperature is 31.24
Current temperature is 31.15
Current temperature is 31.15
Current temperature is 31.24
Current temperature is 31.06
Current temperature is 31.06
Current temperature is 31.06
Current temperature is 30.96
Current temperature is 30.87
Current temperature is 30.87
Current temperature is 30.69
Current temperature is 30.78
Current temperature is 30.69
Current temperature is 30.69
Current temperature is 30.60
Current temperature is 30.69
Current temperature is 30.50
```

Autoscroll ✓ · No line ending · 9600 baud

Figure 28 : Output of Temperature Sketch

7. Implementation with RGB-LCD Display

Sketch:

```
#include <Wire.h>
#include "rgb_lcd.h"
rgb_lcd lcd;
int tempPin = 1;
int heartPin = 0;

void setup()
{
  lcd.begin(16, 2);
}

void loop()
{
  // Display Temperature in C
  int tempReading = analogRead(tempPin);
  float tempVolts = tempReading * 5.0 / 1024.0;
  float tempC = (tempVolts - 0.5) * 100.0;
  float tempF = tempC * 9.0 / 5.0 + 32.0;
  //         ----------------
  lcd.print("Temp        F  ");
  lcd.setCursor(6, 0);
  lcd.print(tempF);
    int heartBeat = analogRead(heartPin);
  lcd.setCursor(0, 1);
  //         ----------------
  lcd.print("Heart     bpm   ");
  lcd.setCursor(6, 1);
  lcd.print(heartBeat);
  delay(500);
}
```

Circuit Diagram :

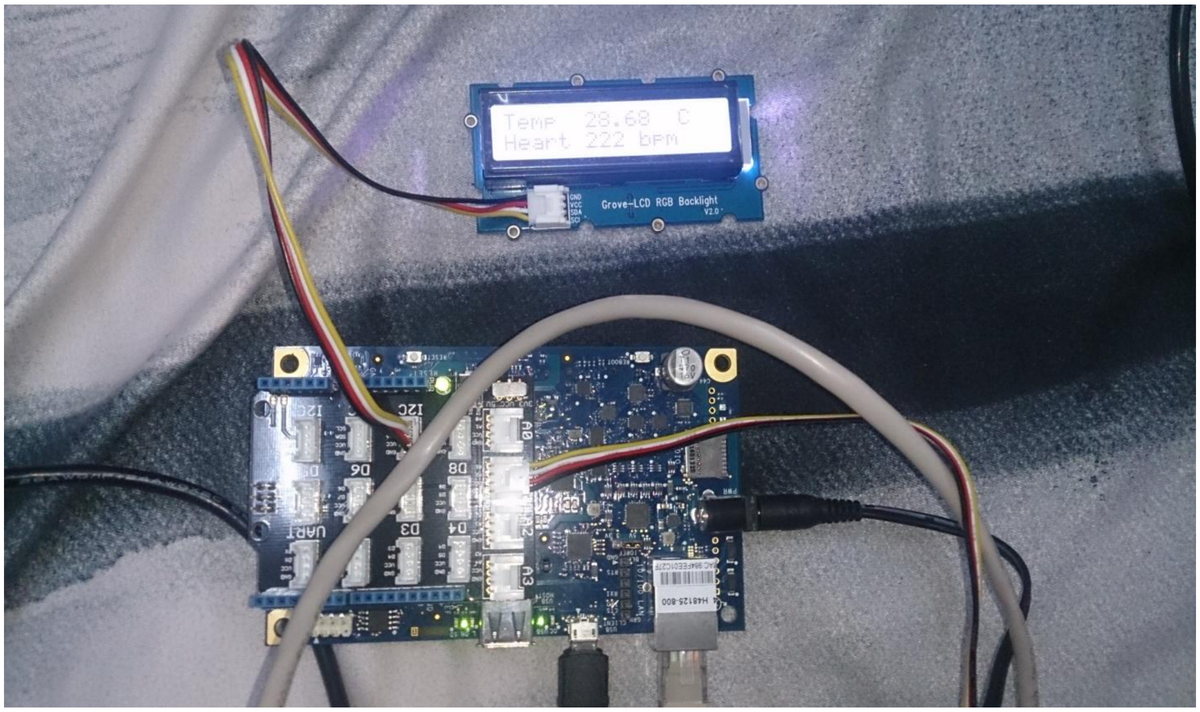


Figure 29 : Circuit diagram of RGB LCD Display

Output of LCD:

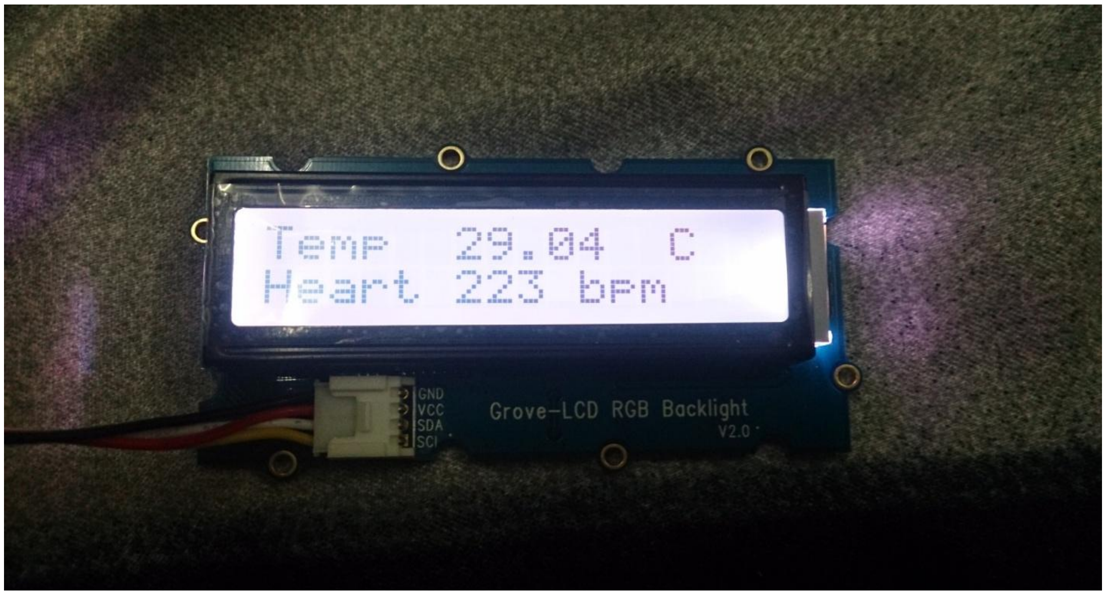


Figure 30 : Output showing RGB LCD Display

8. Live Streaming and Uploading of Data Directly on Web Sever

Circuit Diagram:

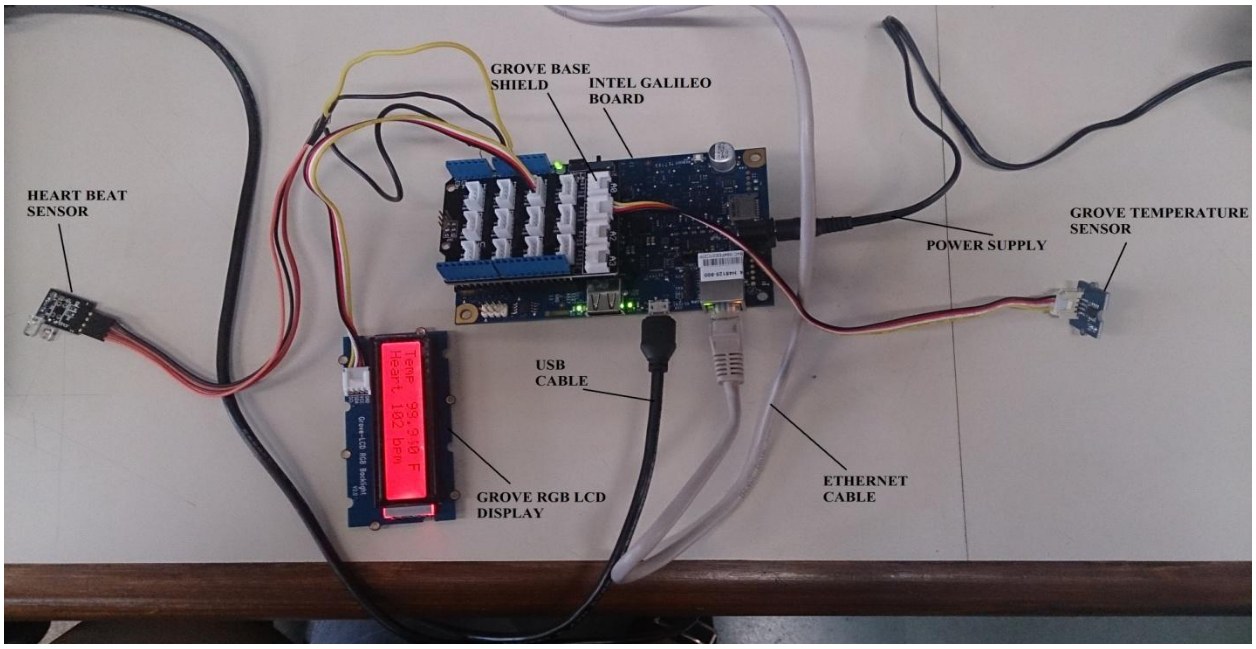


Figure 31 : Circuit diagram of Sensors to Galileo

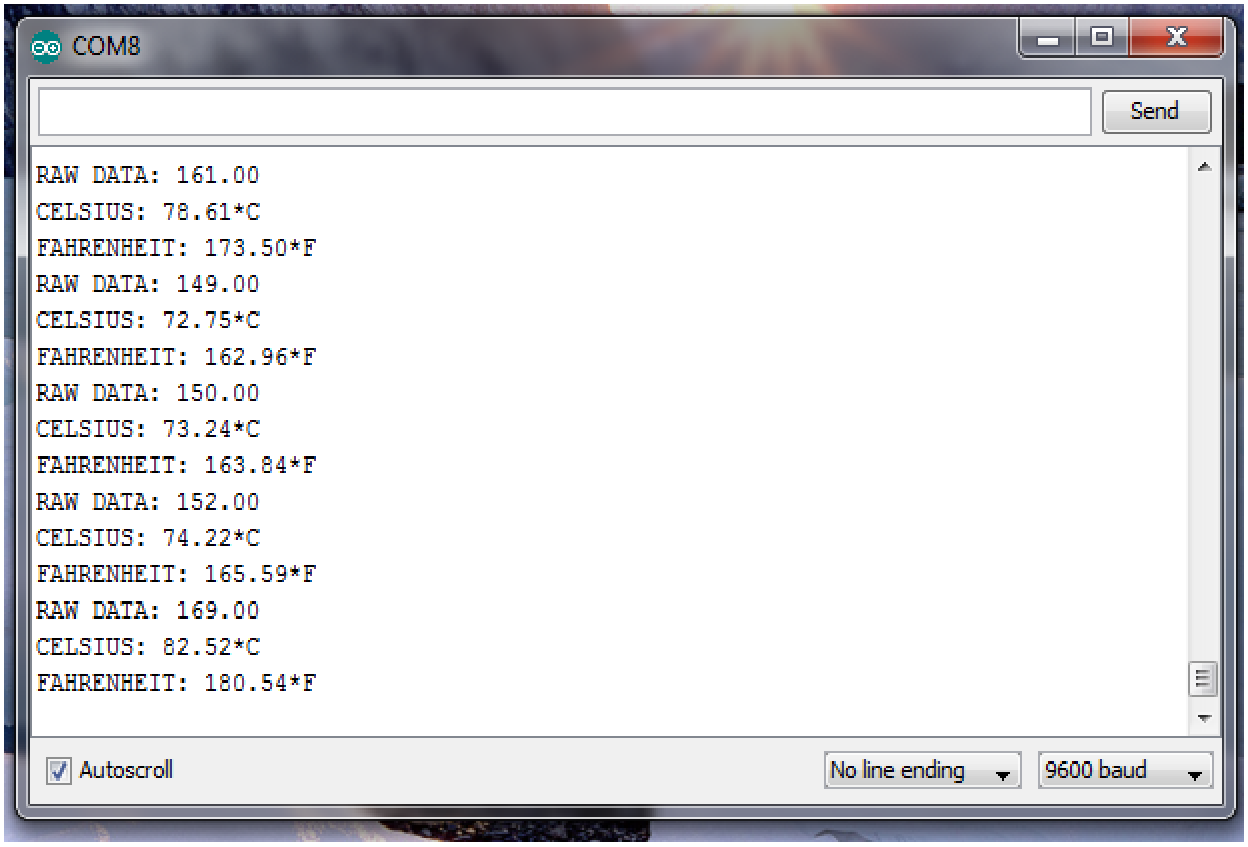
- The Heart Beat sensor is connected through connecting wires to Grove base Shield. The Pin labeled with – is connected to GND, the pin labeled with S is connected to analog input pin A0 of Galileo and the middle one is connected to Vcc.
- Temperature sensor is connected to the Grove Base Shield at analog input A1 through wire connector which is attached to the Galileo board. Sensor output pin is connected to the Analog port 1 of Grove Basic Shield using the 4-pin grove cable. Plug the Grove Basic Shield into Galileo. Connect Galileo to PC by using a USB cable.
- Grove RGB LCD Display is connected to the Grove Base Shield at I2C.

Sketch:

```
#include <SPI.h>
#include <Ethernet.h>
#include <Wire.h>
#include "rgb_lcd.h"

 rgb_lcd lcd;
const int colorR = 255;
const int colorG = 0;
const int colorB = 0;
// Enter a MAC address for your controller below.
// Newer Ethernet shields have a MAC address printed on a sticker on the shield
byte mac[] = {
  0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
IPAddress server(192,168,0,101); // Google
int a1=0,a2=0;
String aa,aa3;
int c=0;
String data;
int B=3975;
// Initialize the Ethernet client library
// with the IP address and port of the server
// that you want to connect to (port 80 is default for HTTP):
EthernetClient client;

void setup() {
  // Open serial communications and wait for port to open:
  Serial.begin(9600);
   lcd.begin(16, 2);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
  }

  // start the Ethernet connection:
  if (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    // no point in carrying on, so do nothing forevermore:
    for(;;)
      ;
  }
  // give the Ethernet shield a second to initialize:
  delay(1000);
  Serial.println("connecting...");
    system("telnetd -l /bin/sh");
      system("ifconfig eth0 > /dev/ttyGS0");
}

void loop()
{// if you get a connection, report back via serial:
  if (client.connect(server, 80)) {
    //Serial.println("connected");
    // Make a HTTP request:
    a1= analogRead(A0);
    a2= analogRead(A1);
    a1=a1-935;
     int tempReading = a2;
  float resistance = (float)(1023-tempReading)*10000/tempReading;
  float tempF = 1/(log(resistance/10000)/B+1/298.15)-273.15;
  //         ----------------
  lcd.print("Temp         C  ");
  lcd.setCursor(6, 0);
  lcd.print(tempF);

  // Display Light on second row
  int heartBeat = a1;
  lcd.setCursor(0, 1);
  //         ----------------
  lcd.print("Heart     bpm   ");
 lcd.setRGB(colorR, colorG, colorB);
  lcd.setCursor(6, 1);
  lcd.print(heartBeat);
  delay(500);
    if(a1<=80)
    {
      aa= (String)a1;
    }
    if(a1>=90)
    {
      aa3= (String)a1;
    }
    String aa1=(String)a2;
    String aa2= "m1";

    data="?pmin=" + aa+"&temp="+aa1+"&mid="+aa2+"&pmax="+aa3;
    if(c>5000)
    {
```

```
   client.println("GET /smarthealthcare/form1.php"+data+" HTTP/1.0");
   c=0;
   Serial.println(data);
  }
   //client.println("Content-Type: application/x-www-form-urlencoded");
   c++;
   client.println();
   //Serial.println(data);
  }
  else {
   // if you didn't get a connection to the server:
   Serial.println("connection failed");
  }

  if(client.connected())
  {
   client.stop();
  }}
```

Output:



```
RAW DATA: 161.00
CELSIUS: 78.61*C
FAHRENHEIT: 173.50*F
RAW DATA: 149.00
CELSIUS: 72.75*C
FAHRENHEIT: 162.96*F
RAW DATA: 150.00
CELSIUS: 73.24*C
FAHRENHEIT: 163.84*F
RAW DATA: 152.00
CELSIUS: 74.22*C
FAHRENHEIT: 165.59*F
RAW DATA: 169.00
CELSIUS: 82.52*C
FAHRENHEIT: 180.54*F
```

Figure 32 : Output of sensors

5.2 SOFTWARE IMPLEMENTATION

This project uses the wamp server to store data on the webserver using mysql database which provides 24 hour health tracking of patient.

User Interface:

1. Index.php

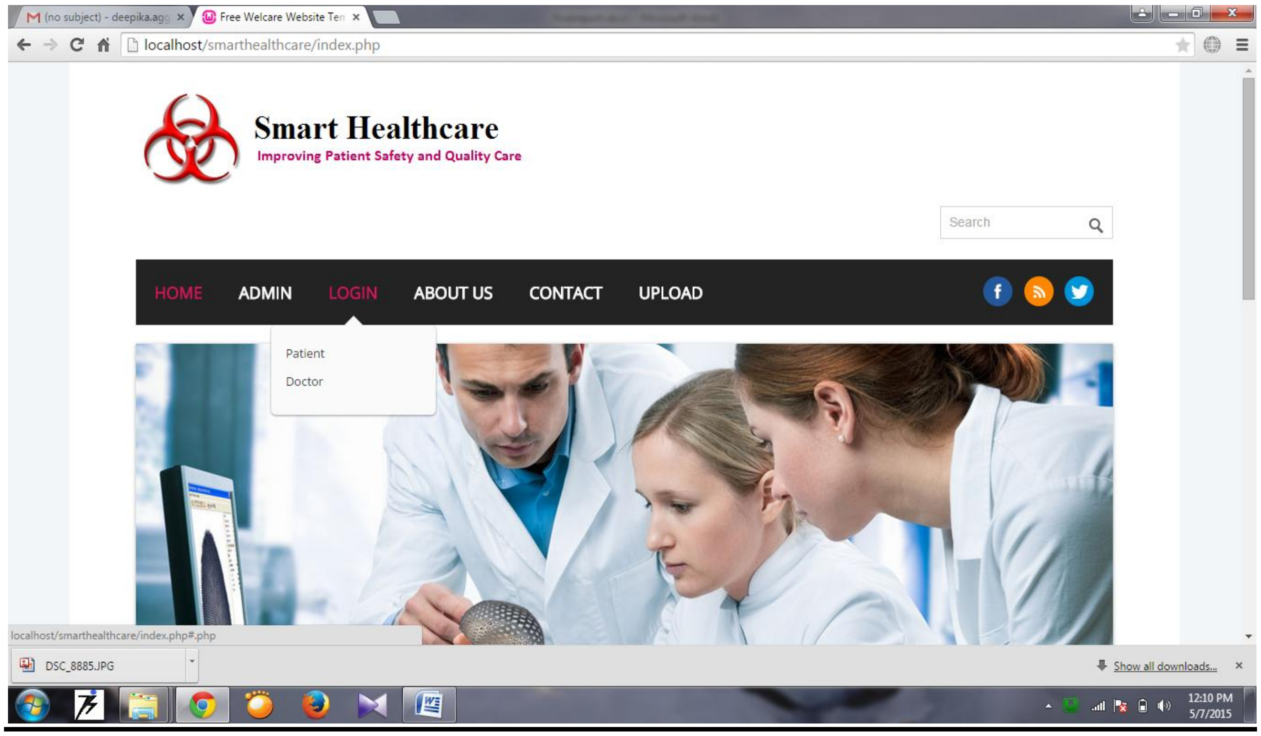


Figure 33 : Home Page

This is the Home page of the web portal where patient and doctor can visit to track the health status of patient. Both patient and doctor can track the health status of patient through this web portal anytime. It provides the live streaming of patient heart rate and temperature.

2. Upload.php

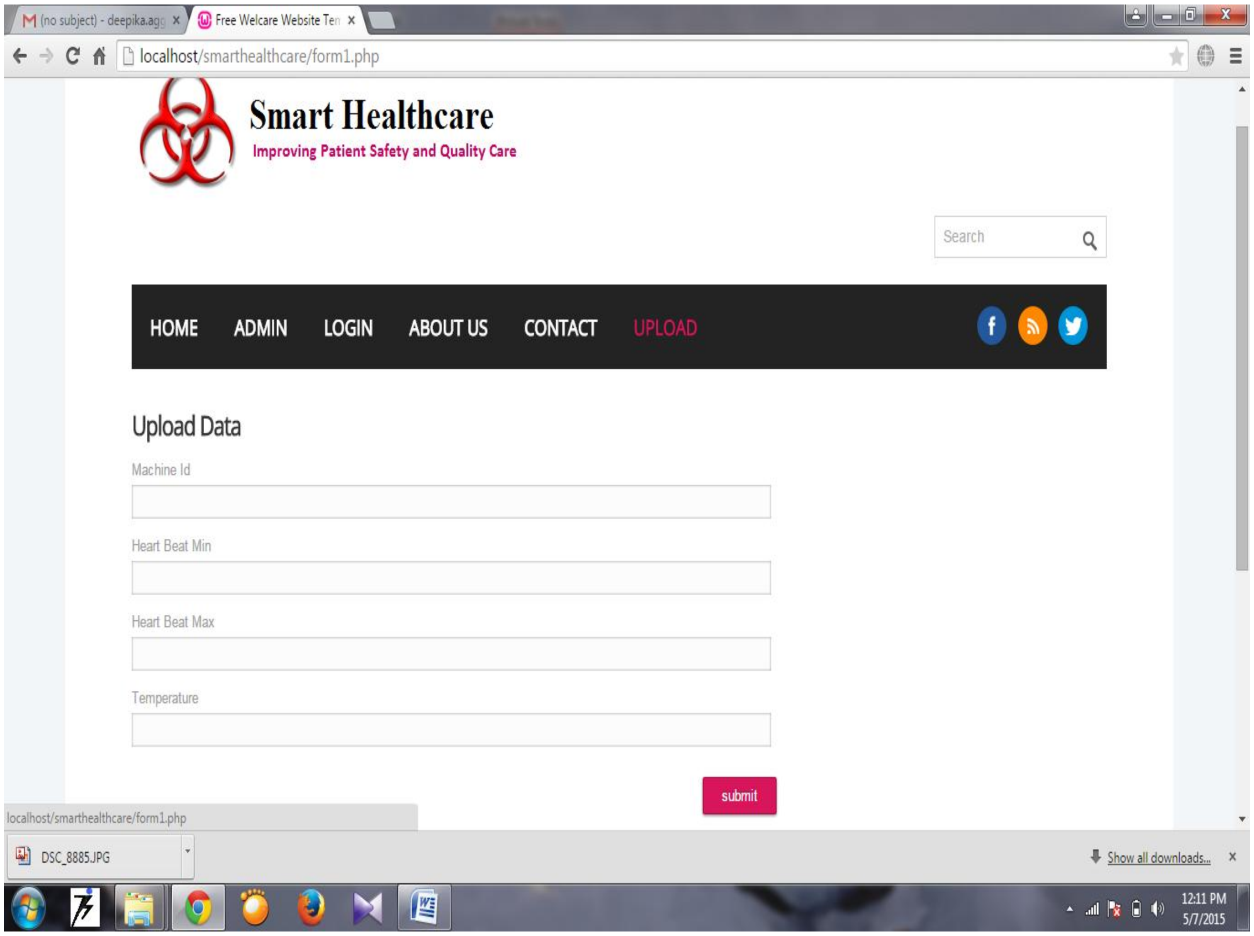
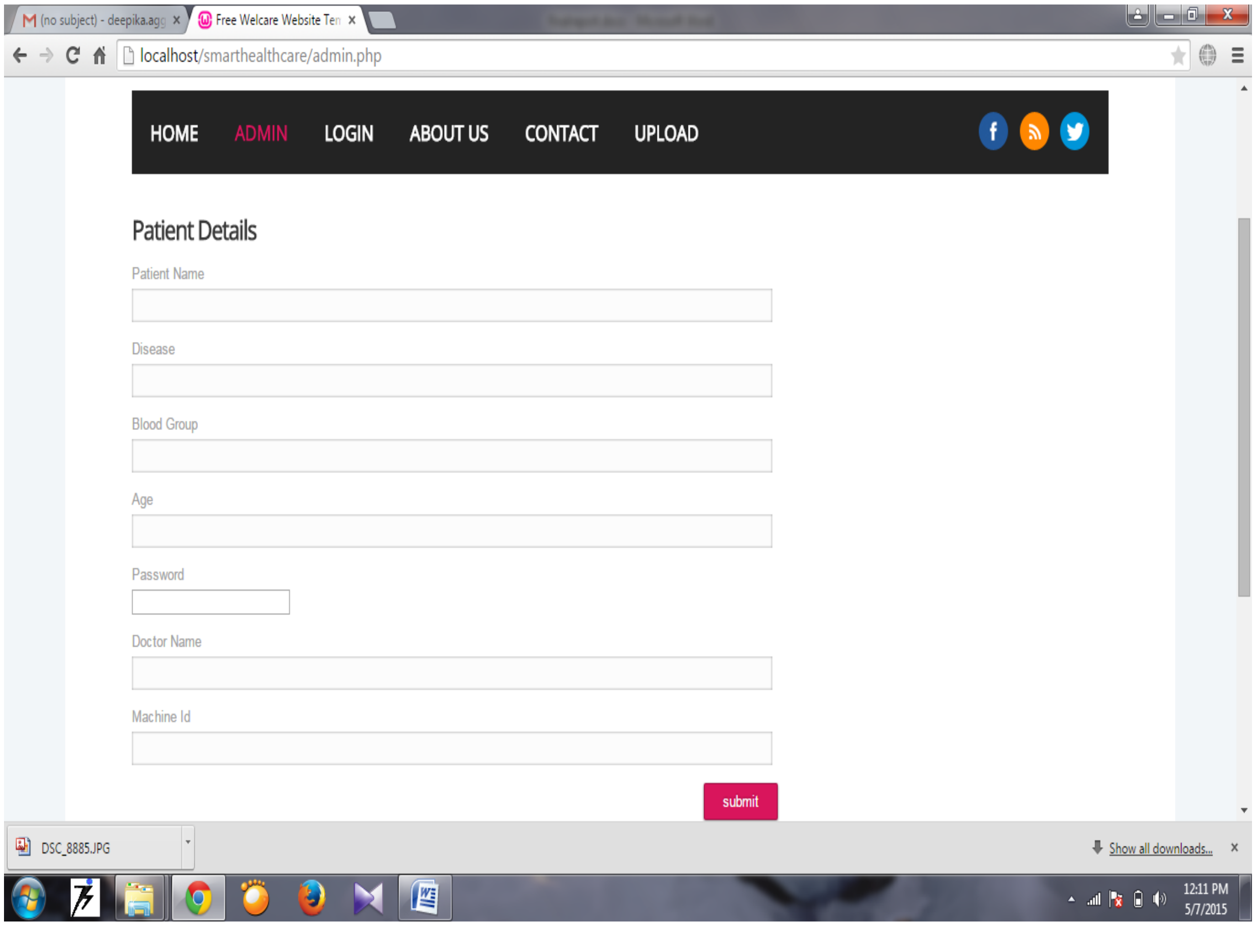


Figure 34: Form for Uploading data

This is the page from where the sensed data from sensor is getting upload on the server via controller i.e. through Galileo. Galileo collects data from both sensors i.e. Heart Beat sensor and Temperature sensor and send it to server via network. Heart Rate getting stored in min and max form. Temperature is getting stored in degree Celsius.

3. Admin page



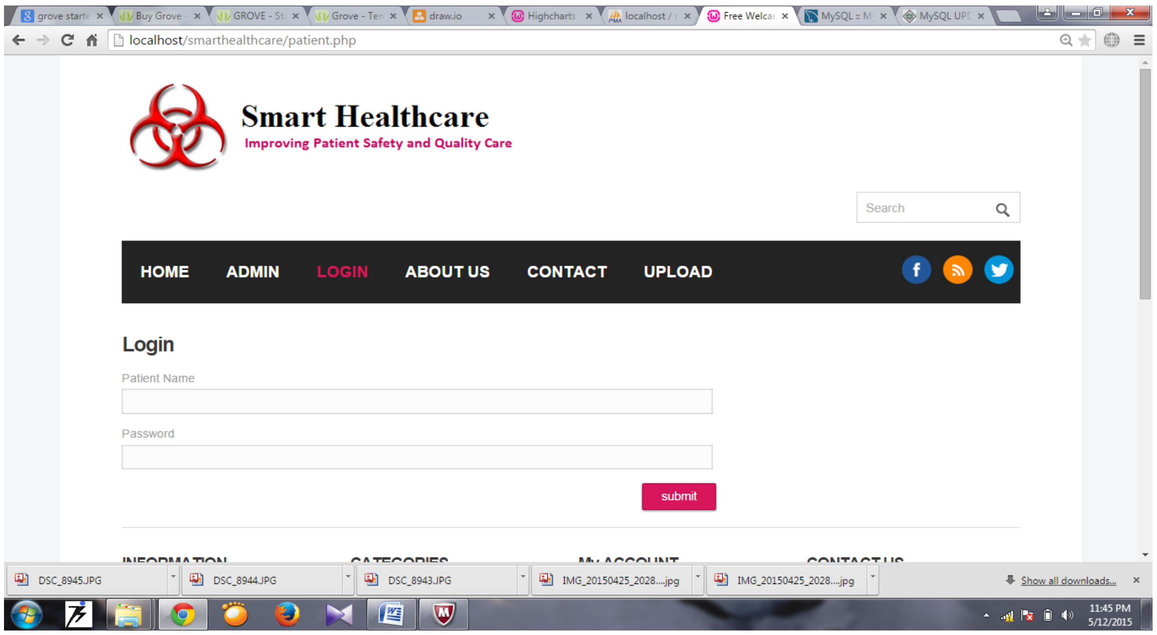
Patient Details

Patient Name	
Disease	
Blood Group	
Age	
Password	
Doctor Name	
Machine Id	

Figure 35 : Admin page

This is the admin page where admin added full details of new patient and corresponding doctor which get stored in database. Admin provides machine id and unique password to each patient. With this password patient can login and can check his health status for medical prescription or further references.

4. Patient Login


localhost/smarthealthcare/patient.php
Smart Healthcare
Improving Patient Safety and Quality Care
Search
HOME
ADMIN
LOGIN
ABOUT US
CONTACT
UPLOAD

Login

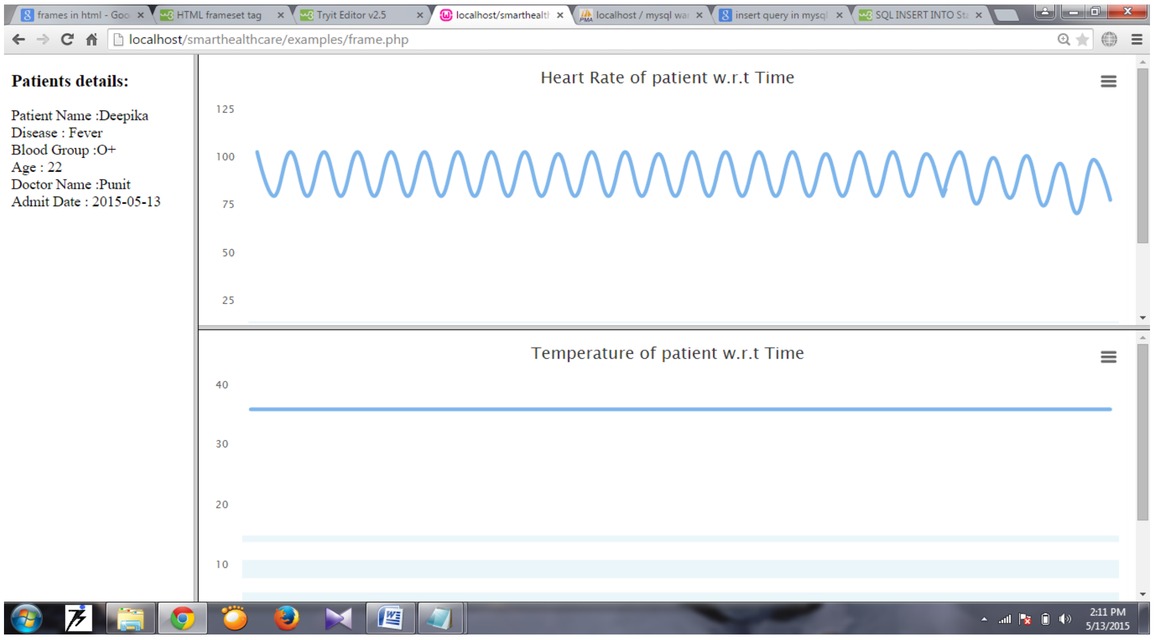
Patient Name

Password

submit

Figure 36 : Patient Login page

It's a patient login page where patient can login and can track his health status as heart rate and temperature with his full information. Patient has to use the password provided by admin to login. After login he can check his full health record and can use it for medical prescription as shown below:


localhost/smarthealthcare/examples/frame.php
Heart Rate of patient w.r.t Time
125
100
75
50
25
Temperature of patient w.r.t Time
40
30
20
10

Patients details:

Patient Name :Deepika
Disease : Fever
Blood Group :O+
Age : 22
Doctor Name :Punit
Admit Date : 2015-05-13

Figure 37: Record of patient on Web Server

5. Doctor Login

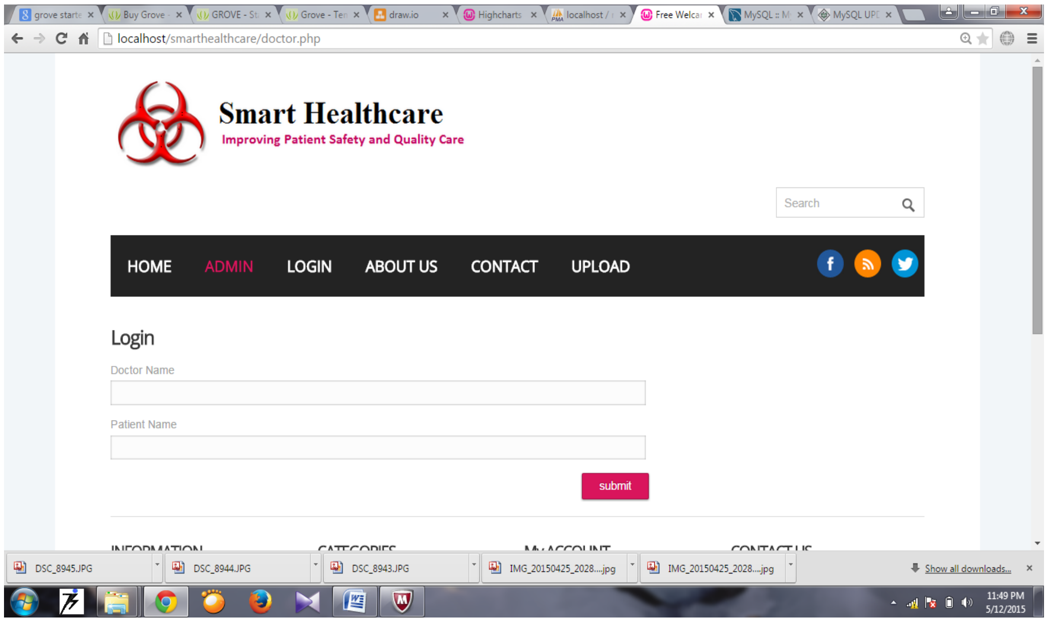
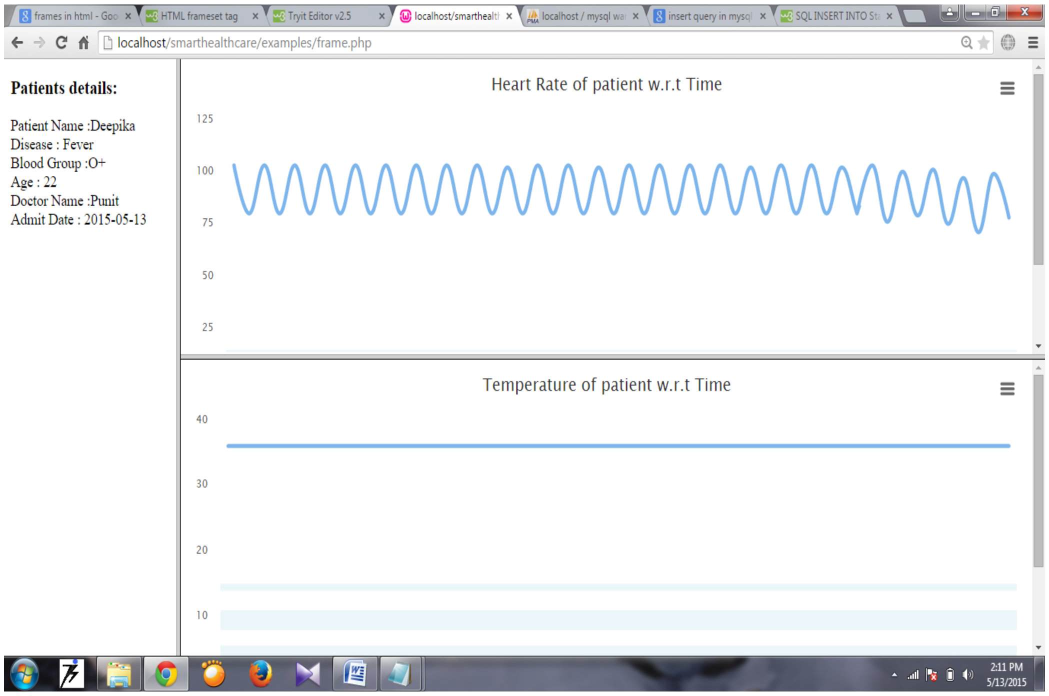


Figure 38 : Doctor Login page

It's a doctor login page where doctor can login and can track his patient's health status as heart rate and temperature with his full log information as shown below:

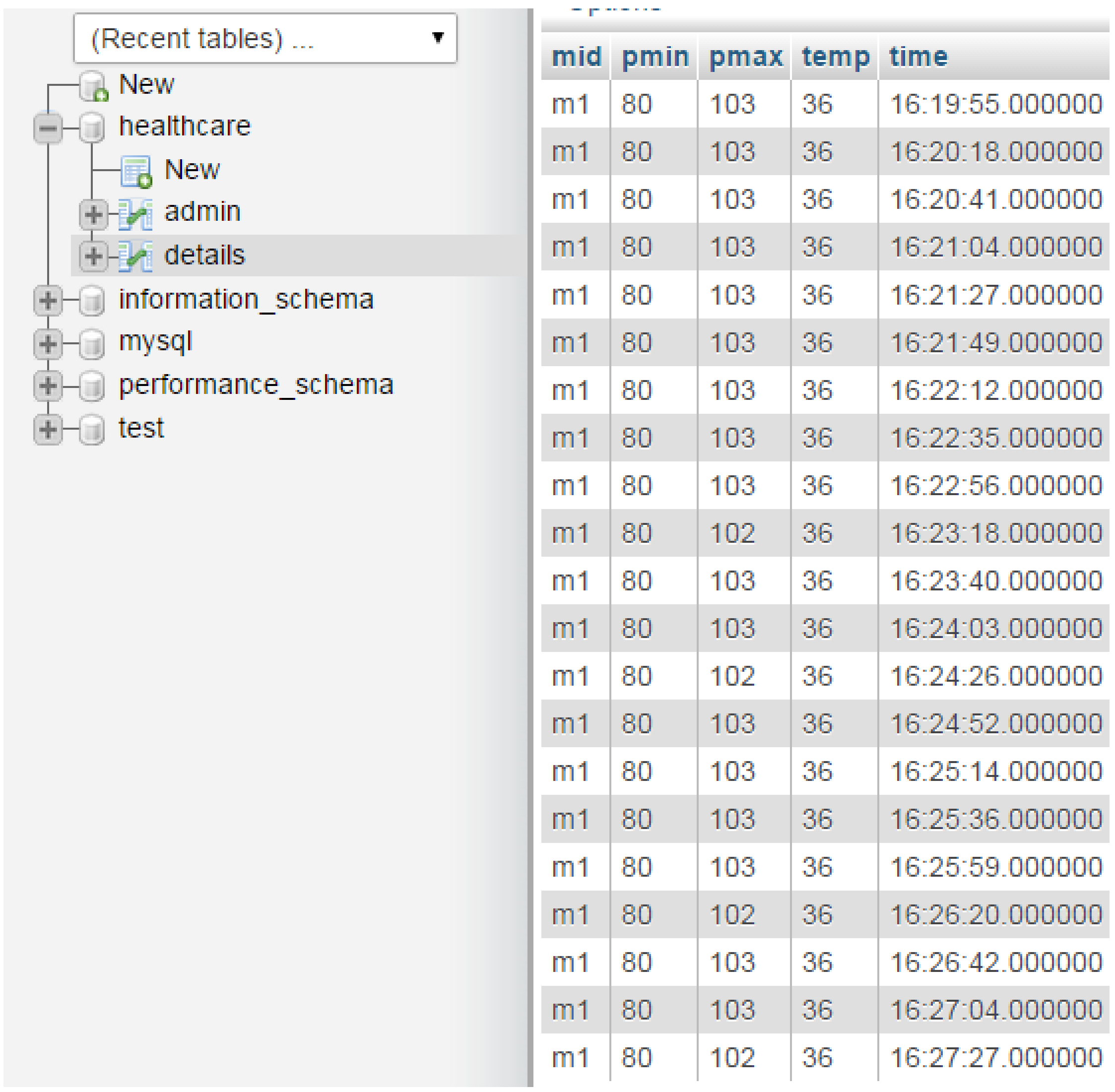

localhost/smarthealthcare/examples/frame.php
Heart Rate of patient w.r.t Time
Temperature of patient w.r.t Time

Patients details:

Patient Name :Deepika
Disease : Fever
Blood Group :O+
Age : 22
Doctor Name :Punit
Admit Date : 2015-05-13

Figure 39: Record of patient on Web Server

6. Database



mid	pmin	pmax	temp	time
m1	80	103	36	16:19:55.000000
m1	80	103	36	16:20:18.000000
m1	80	103	36	16:20:41.000000
m1	80	103	36	16:21:04.000000
m1	80	103	36	16:21:27.000000
m1	80	103	36	16:21:49.000000
m1	80	103	36	16:22:12.000000
m1	80	103	36	16:22:35.000000
m1	80	103	36	16:22:56.000000
m1	80	102	36	16:23:18.000000
m1	80	103	36	16:23:40.000000
m1	80	103	36	16:24:03.000000
m1	80	102	36	16:24:26.000000
m1	80	103	36	16:24:52.000000
m1	80	103	36	16:25:14.000000
m1	80	103	36	16:25:36.000000
m1	80	103	36	16:25:59.000000
m1	80	102	36	16:26:20.000000
m1	80	103	36	16:26:42.000000
m1	80	103	36	16:27:04.000000
m1	80	102	36	16:27:27.000000

Figure 40 : Table showing sensed data from sensors stored in database

CONCLUSION AND FUTURE WORK

The proposed system has been implemented over IoT and majors of the system has been achieved as listed below:

1. Patient’s live data can be tracked from the web portal.
2. A track of patient health record is placed for further references.
3. Live streaming of data is possible through IoT.
4. Tracking of multiple patient record over a same portal.
5. 24-Hour health tracking of patient as his Heart Rate and Temperature.

In future, analysis of the system and the records of patient can be done for prediction of diseases and further prescription of medicines through the health record.

REFERENCES

Papers:

[1] RFID Technology for IoT-Based Personal Healthcare in Smart Spaces. Authored by Sara Amendola, Rossella Lodato, Sabina Manzari, Cecilia Occhiuzzi, and Gaetano Marrocco. IEEE INTERNET OF THINGS JOURNAL, VOL. 1, NO. 2, APRIL 2014

[2] The Potential of Internet of m-health Things “m-IoT” for Non-Invasive Glucose level Sensing. R.S .H. Istepanian Senior Member, IEEE, S. Hu, Senior Member, IEEE, N. Y. Philip, Member, IEEE and A. Sungoor.33rd Annual International Conference of the IEEE EMBS Boston, Massachusetts USA,August 30 - September 3, 2011

Web Links:

[1] ieeeplore.ieee.org

[2] https://communities.intel.com/community/makers/edison/getting-started

[3] https://software.intel.com/en-us/iot/downloads

[4] http://www.hades.in/

[5] http://iotdk.intel.com/sdk/1.1/iotdk-ide-win.7z

[6] tingelectronics.com/tutorials/arduino/ethernet-shield-web-server-tutorial/SD-card-web-server-links

[7] freescale.com/healthcare