Sudan University of Science and Technology

College of Graduate Studies

A secure Low Power Implementation of Internet of Things Health Monitoring System

تنفيذ آمن ذو طاقة منخفضة لنظام مراقبة صحية تابع لإنترنت الأشياء

A Research Submitted In Partial fulfillment for the Requirements of the Degree of M.Sc. degree in Electronics Engineering (Communications)

Prepared By:

Akram Mobarak Mohammed Ahmed Osman.

Supervisor:

Dr. Fath Elrhman Ismael Khalifa Ahmed

AUG 2017

الآيـة

قال تعالى :

﴿ فَتَعَالَى اللَّهُ الْمَلِكُ الْحَقُّ ۗ وَلَا تَعْجَلْ بِالْقُرْآنِ مِن قَبْلِ أَن يُقْضَىٰ إِلَيْكَ وَحْيُهُ ۖ وَقُل رَّبِّ زِدْنِي عِلْمًا ﴾

سورة طه(114)

DEDICATION

For my loving parents,

For my brothers and sisters

For all friends, especially Mustafa.

For the amazing girl that I loved her

I thank my God day and night for having me you in my life

AKNOWLEDGMENT

First, we need to thank our god (Allah) that without his blessing this work will not complete.

Then I thank my supervisor **Dr. Fath Elrhman Ismael** for his patience and countless hours and valuable efforts to guide and advise me to complete the work in his fair way.

Lastly, I need to thank our teachers in department of electronic engineering for their efforts in help and support.

ABSTRACT

The Internet of Things (IoT) makes smart objects the ultimate building blocks in the development of cyber-physical smart pervasive frameworks. The IoT has a variety of application domains, including health care. The IoT revolution is redesigning modern health care with promising technological, economic, and social prospects. Sometimes doctors need to periodically monitor patient's status of body and visualize, although patients may be somewhere out of the medical centers, the information's of patients are very important such that it must be keep from hacking by unauthorized persons. In this thesis, a human health-monitoring platform is designed and developed under the application framework of body sensor network of IoT. The platform can collect the physical information of user by constructing the human state acquisition system. Then the collected human physiological data is transmitted to data processing platform through ZigBee wireless network for further data processing, preservation and display, with which real-time rescue and treatment can be sent out by guardian, doctors and healthcare caregivers. In addition, this project analyzes distinct IoT security and privacy features, including security requirements, threat models, and attack taxonomies from the health care perspective.

المستخلص

ان انترنت الاشياء لعب دوراً مهما في وضع لبنات البناء للكثير من الحقول الذكية. ويتواجد انترنت الاشياء في الكثير من الحقول والتطبيقات ومن هذه التطبيقات هو حقل الرعاية الصحية. مما جعل من انترنت الاشياء كرائد ثوري في جعل حياة المجتمعات اكثر تطورا اقتصاديا وتقنيا واجتماعيا. في هذا البحث، تم تصميم وتطوير منصة لمراقبة صحة الإنسان باستخدام نموذج شبكات التحسس اللاسلكية. في بعض الاحيان يحتاج الطبيب الى مراقبة حالات بعض المرضى بصورة دورية، وهؤلاء المرضى قد يتواجدون خارج الوحدة الصحية كما ان بيانات هؤلاء المرضى تجب حمايتها من الاعتداءت بواسطة اي شخص غير مسموح له بالاطلاع عليها. في هذا البحث تم تصميم نظام مراقبة صحية للمرضى حيث تقوم هذه المنصة بتجميع البيانات الطبيعية للشخص بواسطة حساسات ملامسة للانسان وبعد ذلك يقوم النظام بتشفير البيانات الطبيعية المجمعة ومن ثم ارسالها الى وحدة معالجة هذه البيانات لا سلكيا من اجل استخلاصها ومعالجتها وعرضها وبالتالي يمكن الاستفادة منها بواسطة الاطباء والمسؤولين ووحدات الرعاية الصحية المختلفة.بالإضافة الى ذلك قمنا بدراسة المحددات الامنية لخصوصية المريض ومهدداتها وعلى هذا الاساس تم بناء نظام تشفيري بسيط لحماية بيانات المريض من الاعتداء وتقليل استهلاك الطاقة.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

2FA	Two-Factor Authentication
3D	Three Dimension
3DES	Three Dimension Encryption Scheme
ADC	Analog to Digital Convertor
AES	Advance Encryption Scheme
AREF	Analog Voltage Reference
ARM	Advance RISC Micro-Controller
ARPANET	Advanced Research Projects Agency Network
AVR-MCU	Allen Vargin Reduced Instruction Set Micro-Controller Unit
BCM	Broad-Com Microprocessor
BIST`	
CMOS	Complementary Metal Oxide Semiconductor
GPS	Global Positioning System
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DARPA	Defense Advanced Research Projects Agency
DC	Direct Current
DIP	Dual In-Line Package
DMA	Direct Memory Access
DNS	Domain Name Service
DSP	Digital Signal Processor
DTR	Data Terminal Ready
DVD	Digital Video Disk
ECC	Error Correction Code

EDAC	
EEPROM	Electrically Erasable Programmable Read Only Memory
EPC	
FEC	Forward Error Correction
FFIEC	Federal Financial Institutions Examination Council’s
GPIO	General Purpose input output
GPU	Graphical Processing Unit
HD	High Definition
HDMI	High Definition Multi Interface
HH	Home to Home
HTML	Hyper Text Mark-able Language
HTTP	Hyper Text Transfer Protocol
I2C	Inter interconnection
ICSP	In Chip System Programmer
ID	Identity
IoT	Internet of Things
ISO	International Standardization Organization
KEK	key encryption key
LED	Light Emitting Diode
M2M	Machine to Machine
MISO	Master In Slave Out
MOSI	Master Out Slave In
PC	Programmable Counter
PCB	Printed Circuit Board
RCA	Radio Corporation of America
RFID	Radio Frequency Identification
RISC	Reduce Instruction Set
RX	Reciver

SCK	Shift Clock
SD	Storage Disk
SDRAM	Storage Disk Random Access Memory
SoC	System on Chip
SRAM	Static Random Access Memory
SS	Slave Select
SSL	Secure Socket Layer
TCP	Transferee Control Protocol
TTL	Transistor Transistor Logic
TV	Television
TWBR	Two Wire Bitrate Register
TWCR	Two Wire Control Register
TWI	Two Wire Interface
TX	Transmitter
UART	Universal Asynchronous Receiver Transmitter
USART	Universal synchronous Receiver Transmitter
USB	Universal Serial Bus
WSNs	Wireless Sensor Network

CHAPTER ONE
INTRODUCTION

Chapter One

Introduction

1.1 Preface

The term IoT refers to an interactive connection with low power, low cos short-range radio devices to allow more man touch independent for the modern society by allowing sensors that may be attached to a small digital node for the data collection from the environment[1]. The main features of the Internet of Things first are comprehensive perception, using RFID, sensor, two-dimensional code to access to the information of the object anytime, anywhere. Scope of data collection terminal connected through the Internet of Things is very broad, involving hundreds of millions of heterogeneous devices ubiquitous access, including mobile phones, computers that already have powerful computing, storage and communication capabilities of the terminal; appliances, railways, bridges, buildings such as the embedded sensor device; radio frequency identification (RFID) devices, infrared sensors, global positioning systems, laser scanners. Followed by a reliable transfer, through the wide variety of data collection terminal including the Internet, mobile Internet and other network interconnection, to achieve real-time acquisition of external environment information, the dynamic information of the object is convert it into a data format suitable for network transmission, and transfer to the data center through network. Finally the collected data is intelligent processing, using cloud computing, fuzzy identify and other intelligent computing technology to analyze and process vast amounts of data and information to achieve intelligent control for objects [2]. The huge increase in the amount of available data, as well as the urgent need to link these data sources together directly affects the process of information exchange. Internet of things (IoT) is not only interested in connecting devices with each other but it means connecting devices together with the possibility of interaction between them in an innovative way Machine to Machine M2M. These peripherals are often of different functions, as well as from various sources such as sensors collect data on natural disasters, earthquakes and hurricanes indicators, or you may have medical devices attached to collect the status of patients that are distributes in distant areas, it can also be a range of sensors to control peripheral devices in an industrial institution or a smart home. From the previous narrative it is clear that the network in this way must contain millions of absolutely different hardware functions and that should work in harmony with each other [2]. The IoT is increased rapidly in academia, industry as well as government that has the potential to bring significant personal, professional and economic benefits.

Figure 1.1 IoT system architecture[1]

1.2 Problem Statement

In some medical cases, doctors need to monitor the status of patient's body periodically and visually, these patients may be somewhere out of the medical centers, the information's of patients are very important such that it must be keep from hacking by unauthorized persons.

1.3 Proposed Solution

A small Arduino Nano connected with a number of sensors is attached to the human body representing the sensor node, this node collects the sensory data from sensor, random unique cryptographic keys are used for the encryption of scenery data and then transmitted via wireless module. A raspberry Pi based local server is responsible for the collection and classification of patient health information's and then visualized the result to be used for diagnoses of the patients.

1.4 Aims and Objectives

The aims of this project is to design IoT based health treatment application to achieve the following goals:

- To Design and implement a low power, short range health treatment sensor nodes.
- To design raspberry Pi based local server that managing the transmission , collection and analysis of raw health treatment data
- To Exchange the health information with Raspberry Pi based local server using simple encryption scheme.
- To Maintains the consumed power and bandwidth in tolerable level with cost.

1.5 Methodology

- Design a sensor node that implement a health treatment using SPIO2 sensor, blood-pressure sensor and temperature sensor using proteus simulator.
- Writing C code that collect the sensory data from the various sensors, and to implement an encryption scheme using cryptographic scheme.
- Emulate this data by transmission of information's using computer COM port to insure that the transmission is done properly.
- Batch a Raspbian OS on an SD card and then insert the SD card in raspberry Pi to complete the installation of the OS.
- Install and configure the raspberry Pi as apache server to exchange the transmission with IP networks.
- Connect both the raspberry pi and the sensor node with RF module using USART transmission protocol.
- Configure the raspberry pi to allow the exchanging of data with public server.
- After collect the data received by the raspberry Pi, it then analyzed and visualized using numerical tools I used Matlab and .ods systems.

1.6 Thesis Outline:

In chapter one: an introduction about the concept of IoT is presented in addition to the problem and the proposed solution, also the objectives and the methodology is stated.

In chapter two: a broad background is discussed, also a brief discussion about related work to the thesis is discussed.

In chapter three: the system design will be described in addition to the operation of whole system.

In chapter four: a variety system results are discussed and approved.

In chapter five: a brief conclusion is presented in addition to the recommendations.

CHAPTER TWO

LITERATURE REVIEW

Chapter Two
Literature Review

2.1 Background

Smart devices. Smart-phones. Smart cars. Smart homes. Smart cities. A smart world. These notions have been espoused for many years. Achieving these goals has been investigated, to date, by many diverse and often disjoint research communities. Five such prominent research communities are: Internet of Things (IoT), Mobile Computing (MC), Pervasive Computing (PC), Wireless Sensor Networks (WSN), and most recently, Cyber Physical Systems (CPS). However, as technology and solutions progress in each of these fields there is an increasing overlap and merger of principles and research questions. Narrow definitions of each of these fields are no longer appropriate. Further, research in IoT, PC, MC, WSN and CPS often relies on underlying technologies such as real-time computing, machine learning, security, privacy, signal processing, big data, and others. Consequently, the smart vision of the world involves much of computer science, computer engineering, and electrical engineering. Greater interactions among these communities will speed the progress [1].

Current industrial trends and initiatives aim to connect the unconnected." Today, millions of embedded devices are used in safety and security of critical applications such as industrial control systems, modern vehicles, and critical infrastructure. In the last decades, classical production engineering, automation, and intelligent computation systems merged into the industrial Internet of Things (IoT). The number of computation components integrated into industrial control systems, production systems, and factories is steadily increasing. Programmable logic controllers are replaced by more advanced cyber physical systems (CPS), which are freely programmable embedded devices that control physical processes. CPS typically communicate over closed industrial communication networks but are often also connected to the Internet [2].

Apart from benefits of IoTs, there are several security and privacy concerns at different layers viz; Front end, Back end and Network. the survey in several security and privacy concerns related to Internet of Things (IoTs) defining some open challenges. Then, discussion on some applications of IoTs in real world [3].

The introduction of IPv6 and web services as fundamental building blocks for IoT applications promises to bring a number of basic advantages including, a homogeneous protocol ecosystem that allows simple integration with Internet hosts; simplified development of very different appliances; a unified interface for applications, removing the need for application-level proxies.

Such features greatly simplify the deployment of the envisioned scenarios ranging from building automation to production environments to personal area networks, in which very different things such as a temperature sensor, a luminaire, or an RFID tag might interact with each other, with a human carrying a smart phone, or with backend services [4].

2.1.1 History and industrial drivers of WSNs

The development of WSNs was inspired by military applications, notably surveillance in conflict zones. Today, they consist of distributed independent devices that use sensors to monitor the physical conditions with their applications extended to industrial infrastructure, automation, health, traffic, and many consumer areas[1][7].

Research on WSNs dates back to the early 1980s when the United States Defense Advanced Research Projects Agency (DARPA) carried out the distributed sensor networks (DSNs) program for the US military. At that time, the Advanced Research Projects Agency Network (ARPANET) had been in operation for a number of years, with about 200 hosts at universities and research institutes [4]. DSNs were assumed to have many spatially distributed low-cost sensing nodes, collaborating with each other but operated autonomously, with information being routed to whichever node that can best use the information. Even though early researchers on sensor networks had the vision of a DSN in mind, the technology was not quite ready. More specifically, the sensors were rather large (i.e. the size of a shoe box and bigger), and the number of potential applications was thus limited. Furthermore, the earliest DSNs were not tightly associated with wireless connectivity [5].

2.1.1.1 Characteristic features of WSNs

A WSN in figure 2-1 can generally be described as a network of nodes that cooperatively sense and control the environment, enabling interaction between persons or computers and the surrounding environment. WSNs nowadays usually include sensor nodes, actuator nodes, gateways and clients. A large number of sensor nodes deployed randomly inside of or near the monitoring area (sensor field), form networks through self-organization. Sensor nodes monitor the collected data to transmit along to other sensor nodes by hopping [5][11]. During the process of transmission, monitored data may be handled by multiple nodes to get to gateway node after multi hop routing, and finally reach the management node through the internet or satellite. It is the user who configures and manages the WSN with the management node, publish monitoring missions and collection of the monitored data. As related technologies mature, the cost of WSN equipment has dropped dramatically, and their applications are gradually expanding from the military areas to industrial and commercial fields. Meanwhile, standards for WSN technology have been well developed, such as Zigbee®1, Wireless Hart, ISA 100.11a, wireless networks for industrial automation – process automation (WIA-PA), etc. Moreover, with new application modes of WSN emerging in industrial automation and home applications, the total market size of WSN applications will continue to grow rapidly [5].

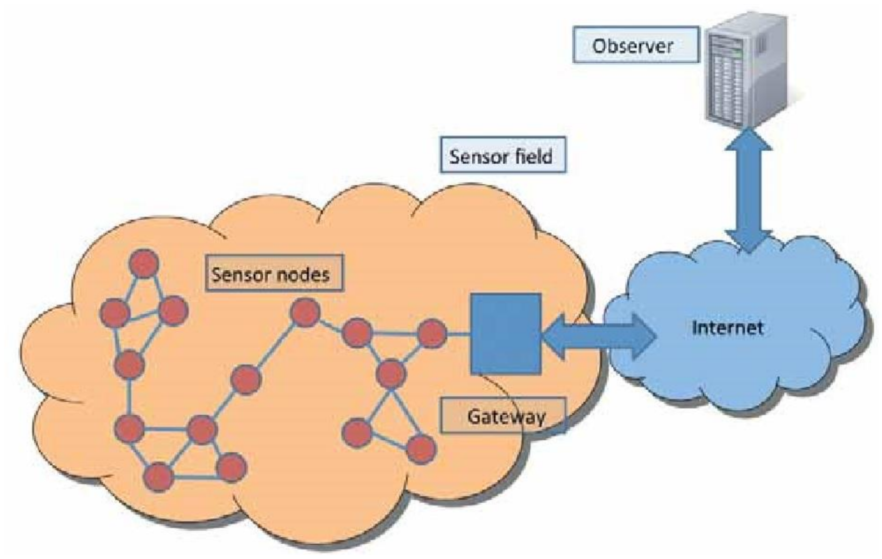


Figure 2.1: Wireless sensor networks[1]

2.1.2 Cloud infrastructure attacks

A smart home device may include a back-end cloud service, depending on the category of the device. In our tests, 68 percent of the devices offered a cloud service. Such a service could be used for statistical purposes, such as logging the home's electricity usage or CO2 levels over a number of months. Other cloud systems allow the remote management of IoT devices, such as light bulbs or heating. Some vendors even force the user to connect to their cloud back-end system and do not provide users with the option of locally managing their devices [6]. The companies either provide access to the cloud service through a smartphone application or a web portal, where users can log in.

Unfortunately, nearly all of the tested IoT cloud services in figure 2-2 allow the user to choose weak passwords, such as “1234”. Even worse, many services prevent the user from using strong passwords with a sufficient level of complexity, due to unreasonable restrictions. One service, for example, restricted the user to a PIN code with a maximum length of four numbers. This makes it easy for any attacker that knows the user’s email address to brute-force their PIN code and take over their account.

Most of the analyzed services don’t lock users out of their accounts after a number of failed login attempts, further allowing attackers to brute-force accounts. None of the analyzed back-end cloud services provided the option of two-factor authentication (2FA) [6].

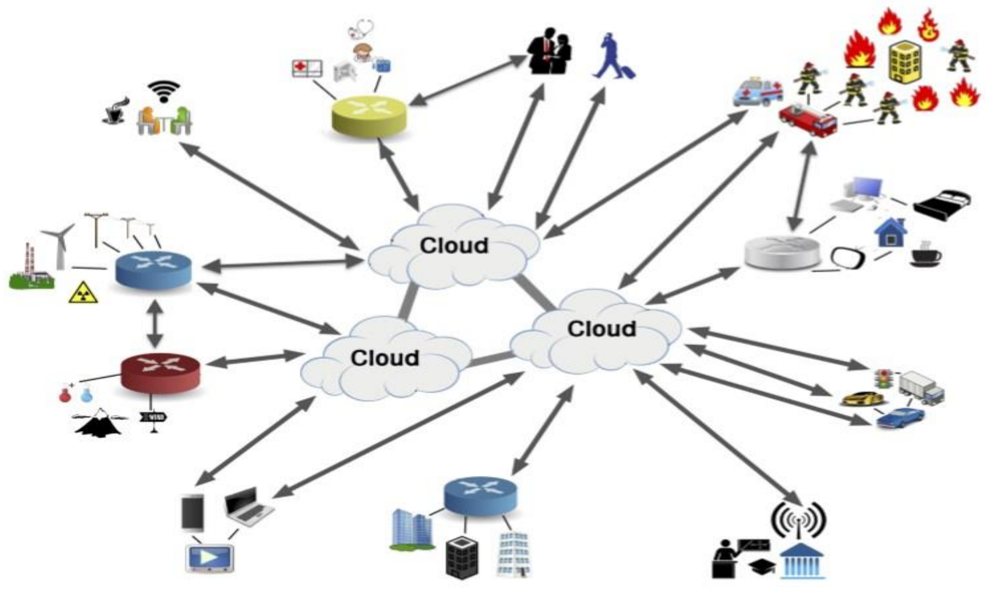


Figure 2.2: An illustration of IoT including cloud services (IoT-Cloud) [7].

2.1.3 Systemic Approach for IoT Security

Here will present the main actors of the systemic approach to security in IoT, introduced in figure 2-3. It is worth to note that the real novelty of the scheme is the introduction of the “Intelligent Object” at the center of the interactions among Person, Process and Technological Ecosystem. In the following we will introduce each of the mentioned actors and their functions in the scheme.

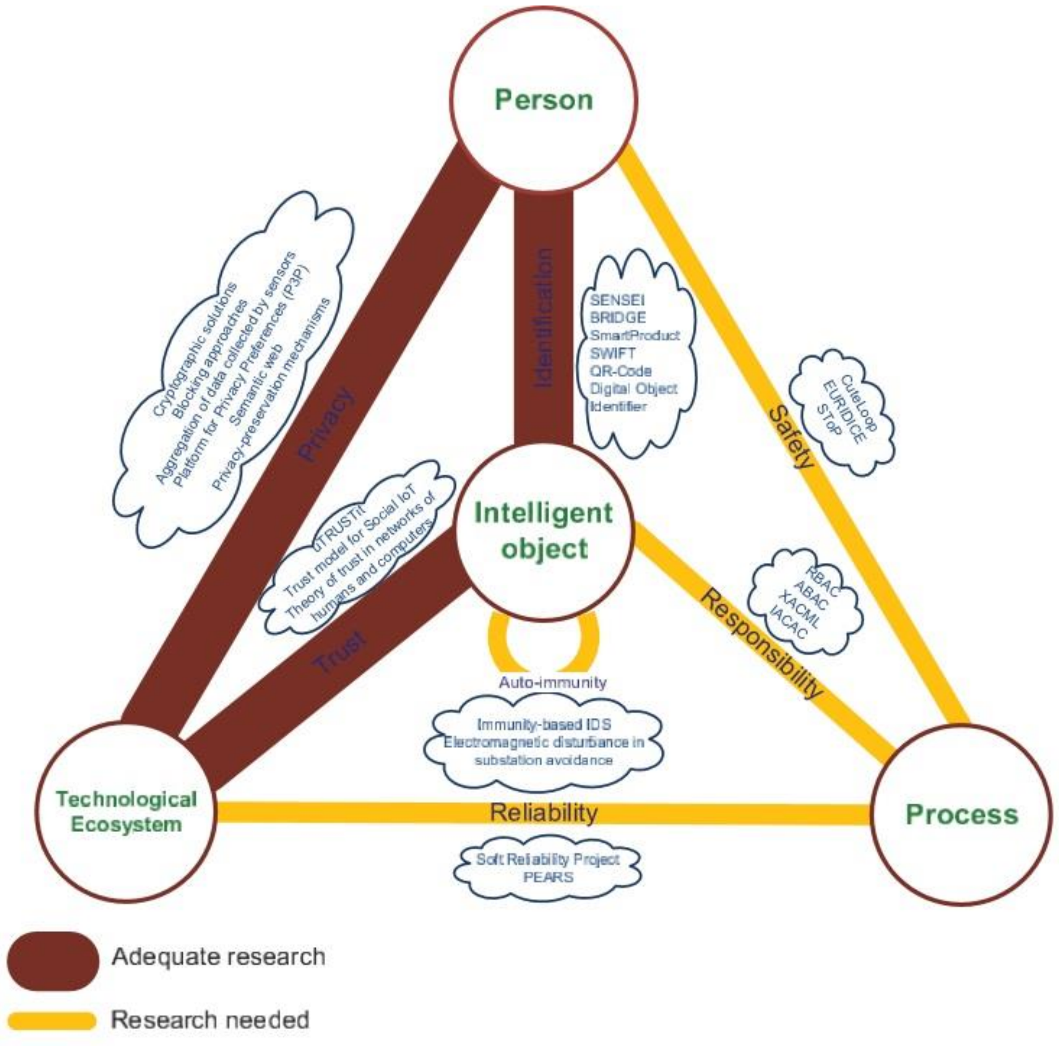


Figure 2.3: A systemic approach for IoT security[3]

A. Person

The first node plays a fundamental role in the IoT security framework. The human resources are responsible for security rules management, which includes:

• Defining security practices and rules.

• Auditing practices and rules efficiency.

• Applying practices and rules when into operational mode.

Due to the complex environment of the IoT, this node is a vital component in security management and enhancement. To this purpose, the human component should be able to analyses the context of IoT, individuate its advantages and limitations, and exploit the technology evolution to bring adequate solutions.

B. Process

The second node refers to a means to accomplishing tasks in the IoT environment according to some security requirements. The process is required to be compliant with the security policies in order to keep the environment secure at different levels. Furthermore, due to the complexity of the model and the presence of different interactions originating from this node, security processes are difficult to implement.

The Federal Financial Institutions Examination Council's (FFIEC) of USA presented a first classification of standard areas to deal with when considering security processes:

• Information Security Risk Assessment.

• Information Security Strategy.

• Security Controls Implementation.

• Security Monitoring.

• Security Process Monitoring and Updating.

In practices, security process need to meet requirements of standards, strategies, policies, procedures and other afferent documents. Thus, an adequate compromise must be found between complexity of security process practices and the needed security level.

C. Intelligent Object

This node is the heart of the new approach. It refers to an “object” augmented by the electronic features needed to let it communicate with other objects in the surrounding environment. These objects will become active participants in business, information and social processes. In fact, objects in the IoT framework will be able to cooperate, share and exchange information about the environment, and respond to events happened in the environment by accomplishing adequate operations. Due to their expected pervasively, the correct design and development of security practices within the conception of intelligent objects is fundamental to ensure the right level of security to the whole environment surrounding them.

D. Technological ecosystem

This node refers to technological choices made to ensure IoT security. According to, information security technology falls into several broad categories:

• Security Design and Configuration

• I&A: Identification and Authorization

• Enclave internal

• Enclave boundary

• Physical and environmental

The choices related to each of these elements may include system architecture, communications protocols, implemented algorithms, access control methods, performance, etc. It is evident that a trade-off among security requirements, feasibility and technology evolution should be found in order to ensure the appropriate level of security without degrading the performance of the system [8].

2.1.4 Machine to Machine

Describes a set of interconnected devices that allow both wireless and wire line systems to communicate with other devices, mostly in vertical segments, i.e., the plumbing/connectivity that enables the IoT ecosystem.

The M2M world of today seems limited only by the imaginations of application developers, device manufacturers, operating systems developers, infrastructure vendors and service providers.

The virtuous cycle of innovation and investment in M2M development embraces both wire line and wireless connectivity, but mobile uses are predominant. By 2018, M2M devices are projected to account for more than 40 percent of connected devices in the United States, as compared to 19.7 percent globally.13 Wireless connectivity is frequently the most efficient, flexible and cost-effective option, especially in geographies lacking reliable wire line access. The result is that this growing segment of the IoT is made possible by mobile connectivity.

To ensure that society realizes future benefits of the IoT, the wireless industry is committed to continuing efforts to develop best practices and create solutions that will support security, transparency and data integrity, including the protection of sensitive and personally identifiable information. As interest in this expansion of the IoT has grown, so has attention to the cyber security and data privacy implications of M2M devices [9].

Table 2-1: Evolving features of the IoT today, in the past and future [10]

	2000	2013	2020	Uncertainty
Technology	RFID	Sensors, cloud phones.	ICT inside things, New technologies.	Invisibility, ubiquity.
Size	Millions	Billions	Billions to trillions	Billions to trillions
Interconnection	Wired stationary	Wireless mobile H2M	E2E, all IP, M2M, interoperability	Ubiquity standards
Data collection	Identifier	Sensory, limited areas,	Active humans increasing coverage, passive humans.	Extent penetration
Things interaction	None	Button, touch Displays haptic	Web interface	Prevalence of web interface
System interaction	None	Smartphone, gestures, speech	Web interfaces ,Haptic using the environment	Using all human senses
Lifecycle	Ownership transfer	Ownership transfer	Product history log, exchangeable	Dynamic-ness
Vertical Vs horizontal	None	Mainly vertical	both	Central solution prevail

2.1.5 EMBEDDED SECURITY SOLUTION

There are many existing solutions to counter different attacks. Encryption of information is used for confidentiality. The most popular cipher algorithms are: RSA, ECC, AES, 3DES.The hash of information is used to check the integrity of a message by providing a signature which is unique for each message. The most known algorithms are MD5 and SHA. In addition, non-repudiation, availability and authenticity are guaranteed by communication protocols like IPsec for example. Most of these algorithms and processes are very much computationally intensive. So, we require dedicated hardware or Digital Signal Processors (DSP).

System designs for embedded devices are complicated, including multiple independent processor cores, secondary bus masters such as DMA engines, and large numbers of memory and peripheral bus slaves. In addition to these functional components there is typically a parallel system infrastructure that provides invasive and non-invasive debug capabilities, as well as component boundary scan and Built-In-Self-Test (BIST) facilities [8]. Due to this kind of importance, complexity as well as the pervasive deployment of embedded devices from home to big enterprises, embedded device security becomes a big issue. Many research initiatives have been undertaken to counter the issues of security in embedded systems. We find great treatment on the issues of embedded system security in [10], where authors have described security requirements, design challenges, basic concepts, different security protocols like Secure Socket Layer (SSL), open SSL, architectures. The SSL protocol is typically layered on top of the transport layer of the network protocol stack, and is either embedded in the protocol suite or is integrated with applications such as web browsers. This is shown in figure 2-4 [11].

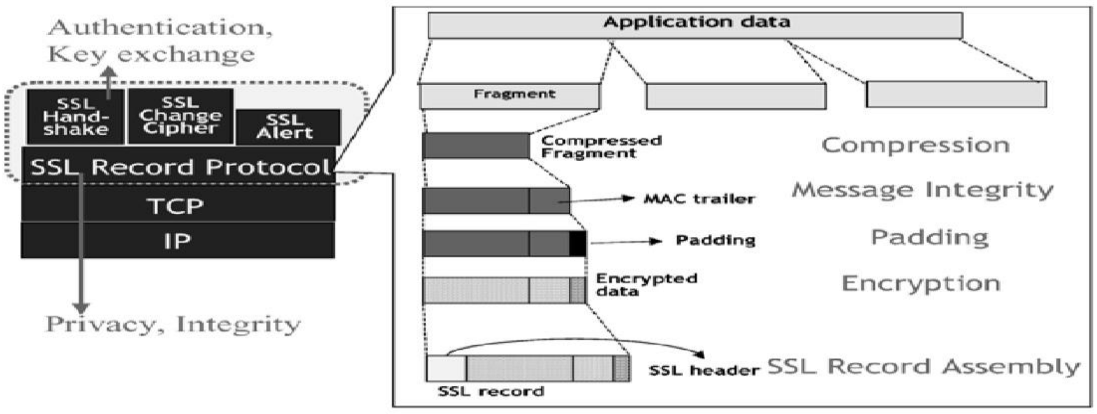


Figure-2-4: SSL protocol, with an expanded view of the SSL record protocol

2.1.6 RFID and the EPC Network

RFID (Radio Frequency Identification) in figure 2-5 is primarily used to identify objects from a distance of a few meters, with a stationary reader typically communicating wirelessly with small battery-free transponders (tags) attached to objects [1]. As well as providing two important basic functions for an Internet of Things – identification and communication – RFID can also be used to determine the approximate location of objects provided the position of the reader is known. At the end of the 1990s, RFID technology was restricted to niche applications such as animal identification, access control and vehicle immobilizers. High transponder prices and a lack of standards constituted an obstacle to the wider use of the technology[5][1].

Since then, however, its field of application has broadened significantly, mainly thanks to MIT's Auto-ID Center, which was founded in 1999. The Auto-ID Center and its successor organization EPC global have systematically pursued a vision of cheap, standardized transponders identifying billions of everyday objects, and they have developed the necessary technology jointly with commercial partners. The use of RFID technology in the supply chains of retail giants such as Wal-Mart and Metro is the result of these efforts. While the adoption by major retailers represents a remarkable success, the evolution of RFID and its associated infrastructure technologies in recent years also highlights challenges involved in realizing an Internet of Things in the broader sense of the term [2].

The development of RFID over recent years is reflected not only in technical progress but also in cost reductions and standardization. For example, the power consumption of the latest generation of transponders is less than 30 µW, with reading distances of up to ten meters possible under favorable conditions. Increasing miniaturization has also led to a unit price of close to five cents for bulk orders of simple RFID transponders. Major progress has also been made in the field of standardization, with the ISO 18000-6C RFID protocol – also referred to as EPC global Gen2 – being supported by several manufacturers, dominating the market and guaranteeing interoperability[12].

High cost pressure and the absence of batteries in transponders means that RFID communications protocols cannot be based on established Internet protocols due to a scarcity of resources. For example, a typical RFID microchip merely consists of a few hundred thousand transistors, contains no microcontroller and has minimal storage capacity – usually just a few bytes. Instead of using a battery, passive RFID microchips are supplied with power remotely from a reading device. Since the power supply can frequently be interrupted due to "field nulls", the transmission of large data packets is avoided – at 128 bits, these are typically much shorter than IP packets. Everyday objects that are to be addressed in an Internet of Things using RFID technology will therefore not behave in exactly the same way as Internet nodes. Instead, it is likely that a highly optimized wireless protocol will be used over the last few meters due to scarce resources and the adverse conditions encountered in the physical world. The RFID reader would act as a gateway between the two different protocols. TCP and HTTP-based protocols have been developed for use in RFID environments, where they are used to configure readers and distribute the data captured via the Internet [12].

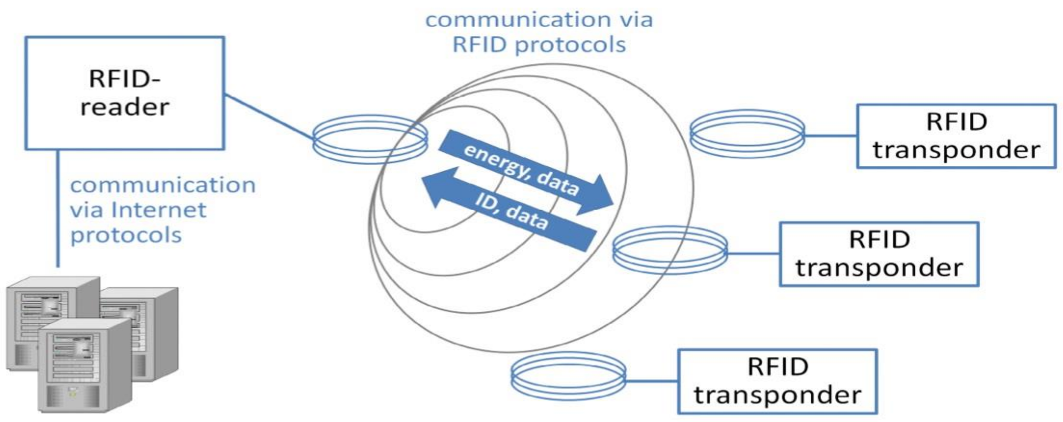


Figure 2.5: RFID Telecommunication [7]

2.2 Related Works

The authors of [2] introduced Industrial IoT systems, the related security and privacy challenges, and outlook possible solutions towards a holistic security framework for Industrial IoT systems. Cyber-attacks on IoT systems are very critical since they may cause physical damage and even threaten human lives. The complexity of these systems and the potential impact of cyber-attacks bring upon new threats.

In addition, authors in [3] introduced Internet of Things (IoTs), which offers capabilities to identify and connect worldwide physical objects into a unified system. As a part of IoTs, serious concerns are raised over access of personal information pertaining to device and individual privacy.

Direct interpretation of the term Internet of Things refers to the use of standard Internet protocols for the human-to-thing or thing-to-thing communication in embedded networks. Although the security needs are well-recognized in this domain, it is still not fully understood how existing IP security protocols and architectures can be deployed. In this paper, we discuss the applicability and limitations of existing Inter-net protocols and security architectures in the context of the Internet of Things [4].

While in [13] a general survey of all the security issues existing in the Internet of Things (IoT) along with an analysis of the privacy issues that an end user may face as a consequence of the spread of IoT. The majority of the survey is focused on the security loopholes arising out of the information exchange technologies used in Internet of Things. No counter measure to the security drawbacks has been analyzed.

But in [14] creates by itself an interesting challenge when adding new things and enabling new services on the Internet. Without public IP addresses, the Internet of Things capabilities would be greatly reduced. Most discussions about IoT have been based on the illusionary assumption that the IP address space is an unlimited resource or it is even taken for granted that IP is like oxygen produced for free by nature. Hopefully, the next generation of Internet Protocol, also known as IPv6 brings a solution.

In [15] after outlining key challenges in data security and privacy, we summarize research directions for securing IoT data, including efficient and scalable encryption protocols, software protection techniques for small devices, and fine-grained data packet loss analysis for sensor networks.

The Internet-of-Things (IoT) has quickly moved from the realm of hype to reality with estimates of over 25 billion devices deployed by 2020. While IoT has huge potential for societal impact, it comes with a number of key security challenges—IoT devices can become the entry points into critical infrastructures and can be exploited to leak sensitive information.

Traditional host-centric security solutions in today’s IT ecosystems (e.g., antivirus, software patches) are fundamentally at odds with the realities of IoT (e.g., poor vendor security practices and constrained hardware) [16].

In the recent years, people need to use Internet at anytime and anywhere. Internet of Things (IOT) allows people and things to be connected Anytime, Anyplace, with anything and anyone, ideally using any path/network and Any service. IOT can be distinguished by various technologies, which provide the creative services in different application domains. This implies that there are various challenges present while deploying IOT. The traditional security services are not directly applied on IOT due to different communication stacks and various standards. So flexible security mechanisms are need to be invented, which deal with the security threats in such dynamic environment of IOT. In [17] survey we present the various research challenges with their respective solutions. Also, some open issues are discovered and some hints for further research direction are advocated.

As IoT is built on the basis of the Internet, security problems of the Internet will also show up in IoT. And as IoT contains three layers: perception layer, transportation layer and application layer, also analyzes the cross-layer heterogeneous integration issues and security issues in detail and discusses the security issues of IoT as a whole and tries to find solutions to them [18].

We propose to have a device which is the integration of multiple devices, hardware comprises of a wearable “Smart band” which continuously communicates with Smart phone that has access to the internet. The application is programmed and loaded with all the required data which includes Human behavior and reactions to different situations like anger, fear and anxiety [19].

CHAPTER THREE

DESIGN OF HEALTH TREATMENT SYSTEM USING RASPBERRY PI AND ARDUINO

Chapter Three
Design of Health Treatment System Using Raspberry Pi and Arduino

3.1 Introduction

The figure 3.1 represent the proposed health treatment IoT system include various stages of the signal road map from entry of the data up to visualization of the collected data.

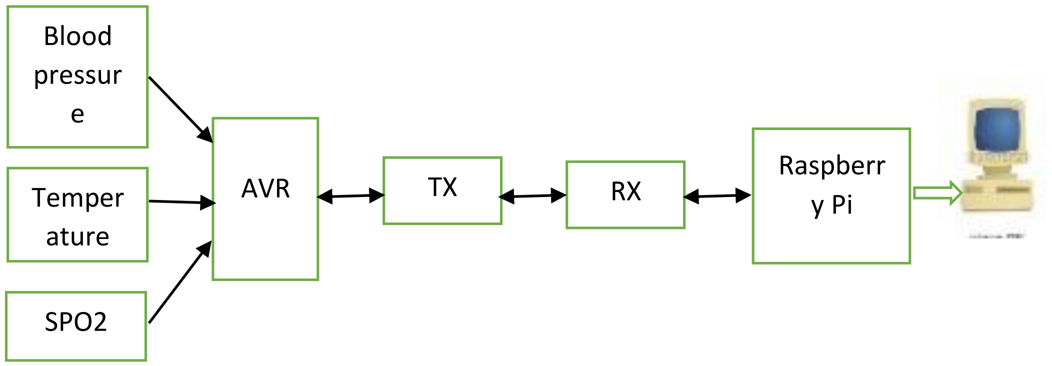


Figure 3-1: The proposed Health treatment system

Here the sensors data is to be collected by the AVR microcontroller, thus the AVR microcontroller can be named as the brain of the sensor node, the AVR then add a simple encryption algorithm to the collected sensory data and then the data is transmitted via serial port to APC220 wireless module, the wireless module broadcast the information into space, at the right hand side locate the receiver that receives the information and then handoffs the received data to the raspberry Pi via serial port, the raspberry pi writes these information's to a file and then it can analyzed and visualized using appropriate numerical tools.

3.2 Acquisition of Sensory data

At this part three type of sensors are introduced beginning by pressure sensor, temperature sensor and SPO2 sensor as it illustrated below in detail.

3.2.1 Blood pressure sensor

Blood pressure is the pressure of the blood in the arteries as it is pumped around the body by the heart. When your heart beats, it contracts and pushes blood through the arteries to the rest of your body. This force creates pressure on the arteries. Blood pressure is recorded as two number the systolic pressure and over the diastolic pressure.

Figure 3-2: pressure Sensor

The blood pressure does not stay the same all the time. It changes to meet your body’s needs. It is affected by various factors including body position, breathing or emotional state, exercise and sleep. It is best to measure blood pressure when you are relaxed and sitting or lying down.

Table 3-1: Human Pressure Range

Human Situation	**Systolic(mm Hg)**	**Diastolic(mm Hg)**
Hypotension	<90	<60
Desired	90-119	60-79
Prehypertension	120-139	80-89
Stage1 Hypertension	140-159	90-99
Stage2 Hypertension	160-179	100-109
Hypertensive Crisis	>= 180	>=110

High blood pressure (hypertension) can lead to serious problems like heart attack, stroke or kidney disease. High blood pressure usually does not have any symptoms, so you need to have your blood pressure checked regularly.

3.2.2 Temperature sensor

Body temperature depends upon the place in the body at which the measurement is made, and the time of day and level of activity of the person. Different parts of the body have different temperatures.

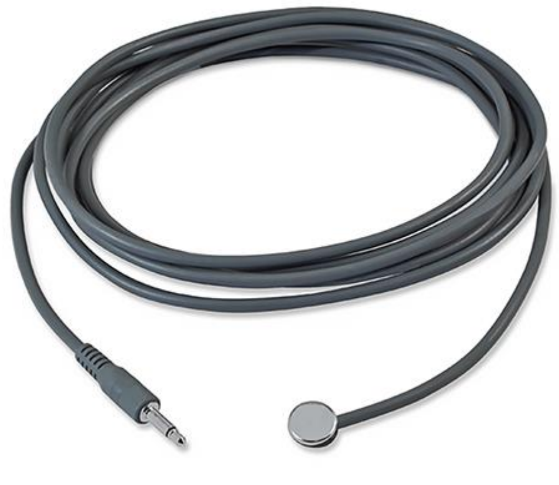


Figure 3-3: Temperature Sensor

The commonly accepted average-core body-temperature is 37.0°C (98.6°F). In healthy adults, body temperature fluctuates about 0.5°C (0.9°F) throughout the day, with lower temperatures in the morning and higher temperatures in the late afternoon and evening, as the body's needs and activities change.

It is of great medical importance to measure body temperature. The reason is that a number of diseases are accompanied by characteristic changes in body temperature. Likewise, the course of certain diseases can be monitored by measuring body temperature, and the physician can evaluate the efficiency of a treatment initiated.

Table 3-2: Human Temperature Range

Human Situation	**Temperature**
Hypothermia	<35.0 ℃ (95.0 °F)
Normal	36.5-37.5 ℃ (97.7-99.5 °F)
Fever Or Hyperthermia	>37.5-38.3 ℃ (99.5-100.9 °F)
Hyperpyrexia	>40.0-41.5 ℃ (104.0-106.7 °F)

The precision of the Body Temperature Sensor can improved by a calibration process. Calibration is a process of measuring voltage and resistance real values.

When using temperature sensor, you are actually measuring a voltage, and relating that to what the operating temperature of the sensor must be. If you can avoid errors in the voltage measurements, and represent the relationship between voltage and temperature more accurately, you can get better temperature readings.

3.2.3 SPO2 sensor

Pulse oximetry a noninvasive method of indicating the arterial oxygen saturation of functional hemoglobin.

Oxygen saturation is defined as the measurement of the amount of oxygen dissolved in blood, based on the detection of Hemoglobin and Deoxyhemoglobin. Two different light wavelengths are used to measure the actual difference in the absorption spectra of HbO2 and Hb. The bloodstream is affected by the concentration of HbO2 and Hb, and their absorption coefficients are measured using two wavelengths 660 nm (red light spectra) and 940 nm (infrared light spectra). Deoxygenated and oxygenated hemoglobin absorb different wavelengths.

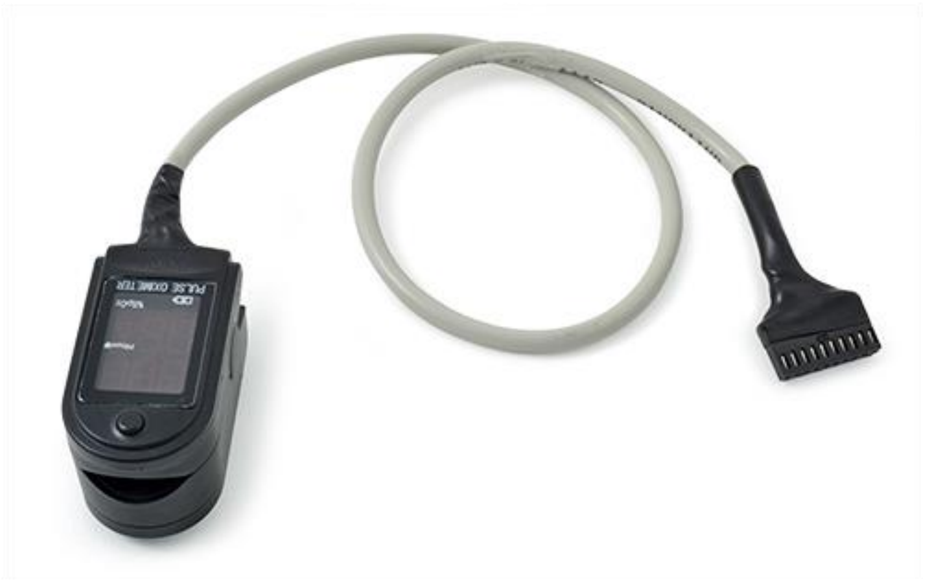


Figure 3-4 SPO2 Sensor

Deoxygenated hemoglobin (Hb) has a higher absorption at 660 nm and oxygenated hemoglobin (HbO2) has a higher absorption at 940 nm . Then a photo-detector perceives the non-absorbed light from the LEDs to calculate the arterial oxygen saturation.

A pulse oximeter sensor is useful in any setting where a patient's oxygenation is unstable, including intensive care, operating, recovery, emergency and hospital ward settings, pilots in unpressurized aircraft, for assessment of any patient's oxygenation, and determining the effectiveness of or need for supplemental oxygen.

Acceptable normal ranges for patients are from 95 to 99 percent, those with a hypoxic drive problem would expect values to be between 88 to 94 percent, values of 100 percent can indicate carbon monoxide poisoning.

3.3 Description of Arduino Nano Board

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. 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 in the figure 3.8 is programmed with many programming languages such as assembly, C as well Arduino development programming language.

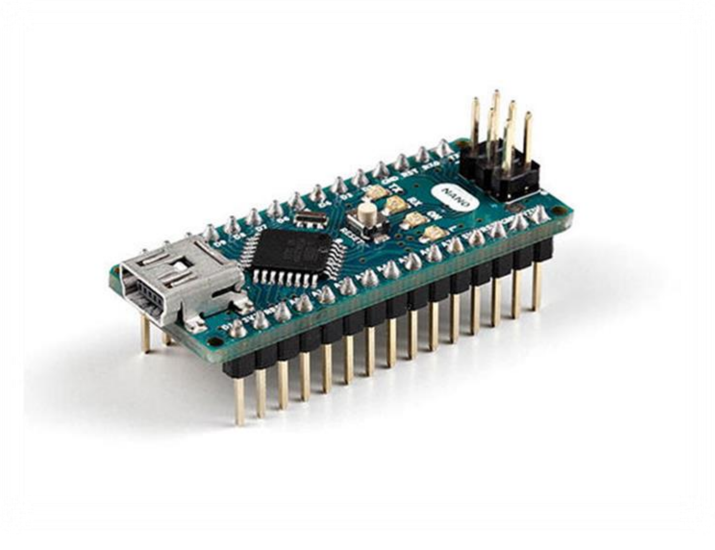


Figure 3.5: Arduino Nano[7]

Arduino Nano is a surface mount breadboard embedded version with integrated USB. It is a smallest, complete, and breadboard friendly. It has everything that Diecimila/Duemilanove has (electrically) with more analog input pins and onboard +5V AREF jumper. Physically, it is missing power jack. The Nano is automatically sense and switch to the higher potential source of power, there is no need for the power select jumper.

Table 3-3: Specifications of Arduino Nano

Microcontroller	**Atmel ATmega328**
Operating Voltage (logic level)	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O	14 Pins
Analog Input	8 Pins
DC Current per I/O Pins	40 mA
Flash Memory	32 KB
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz

3.3.1 Power of Arduino Nano

The Arduino Nano can be powered via the Mini-B USB connection, 6-20V unregulated external power supply, or 5V regulated external power supply. The power source is automatically selected to the highest voltage source.

3.3.2 Memory of AVR Atmega328P

The ATmega328 has 32 KB, (also with 2 KB used for the bootloader. The ATmega328 has 2 KB of SRAM and 1 KB of EEPROM.

3.3.3 Input and Output of Arduino Nano

Each of the 14 digital pins on the Nano can be used as an input or output. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 KOhms. Figure 3.9 shows the layout pf Arduino Nano. Some pins have specialized functions:

- Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip.
- External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value.
- PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output.
- SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
- 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.
- The Nano has 8 analog inputs, each of which provide 10 bits of resolution. By default, they measure from ground to 5 volts. Analog pins 6 and 7 cannot be used as digital pins. Additionally, some pins have specialized functionality:

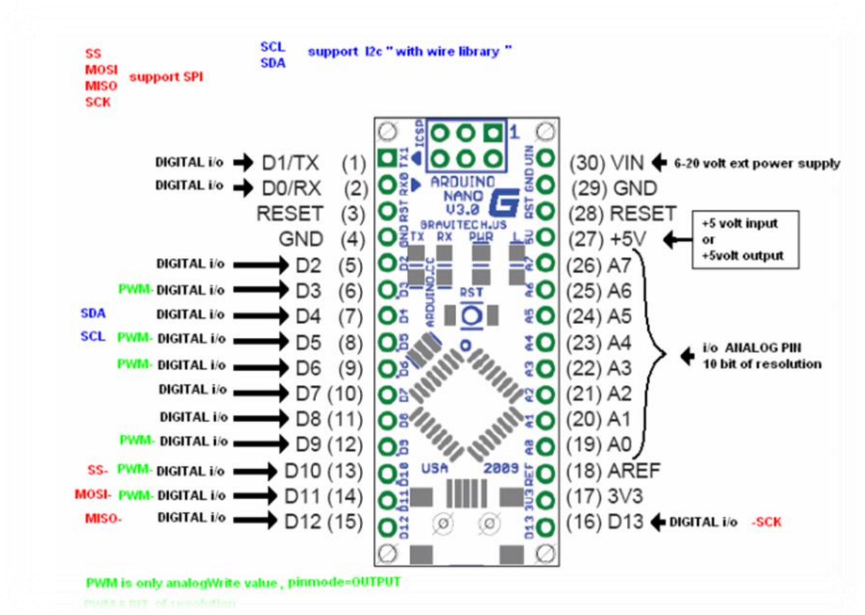


Figure 3-6: Arduino Nano Layout

- I2C: A4 (SDA) and A5 (SCL). Support I2C (TWI) communication.

There are a couple of other pins on the board:

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

3.3.4 Communication between Arduino Nano with host computer

The Arduino Nano has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provide UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An FTDI FT232RL on the board channels this serial communication over USB and the FTDI drivers provide a virtual com port to software on the computer. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board.

The RX and TX LEDs on the board will flash when data is being transmitted via the FTDI chip and USB connection to the computer Programming the Arduino Nano can be programmed with the Arduino software. The ATmega328 on the Arduino Nano comes preboned 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.

You can also bypass the boot loader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using Arduino ISP or similar.

3.3.5 Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Nano 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 FT232RL is connected to the reset line of the ATmega328 via a 100nf 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 boot loader can have a shorter timeout, as the lowering of DTR can be well.

3.4 APC220 RF Module Specifications

The APC220 radio module provides a simple and economic solution for wireless data communications. Figure 3.4 show an APC220 Integrates an embedded high speed microprocessor and high performance IC that creates a transparent UART/TTL interface, and eliminates any need for packetizing and data encoding. If you are looking for a low-cost solution with better range performance, this has been our customer choice for a long time now.

It inventively uses efficient cycle cutting and EDAC which can correct maximum of 24 bits' continuous burst error with coding gain nearly 3dBm, this scheme is much higher than FEC. The anti-burst interference capability and sensitivity are greatly improved. Therefore, this module is suitable and powerful for use in harsh environment.

Figure 3-7: RF Module AP220

3.4.1 Connect APC220 to PC

To connect APC220 to Raspberry, a TTL to RS232 or TTL to USB converter is required.

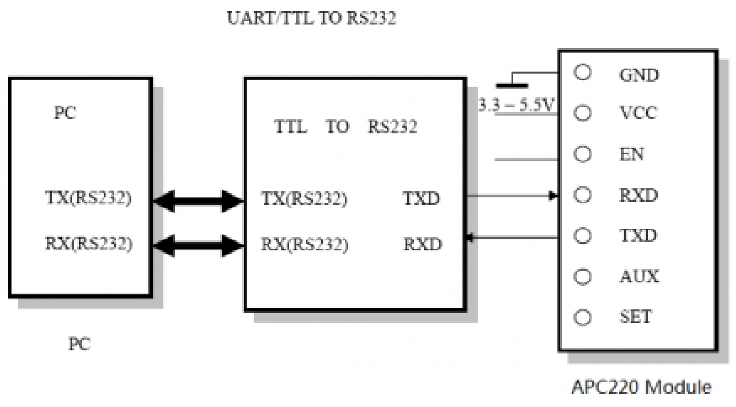


Figure 3-8: RF Module Layout

The block diagram in the figure 3.5 the UART to TTL converter used to achieve the data level transmission between the host computer and the APC220 module. The USB to TTL converter combine the USB-232-1 and TTL-232-1 allows you to convert USB to TTL/CMOS compatible levels and vice versa.

3.4.2 Connect APC220 to MCU

Any MCU which has TTL port is able to talk to APC220. A diagram is shown below for easy connection. In figure 3.6 The GPIO pins of the Arduino Nano are connected to corresponding pins in the RF module.

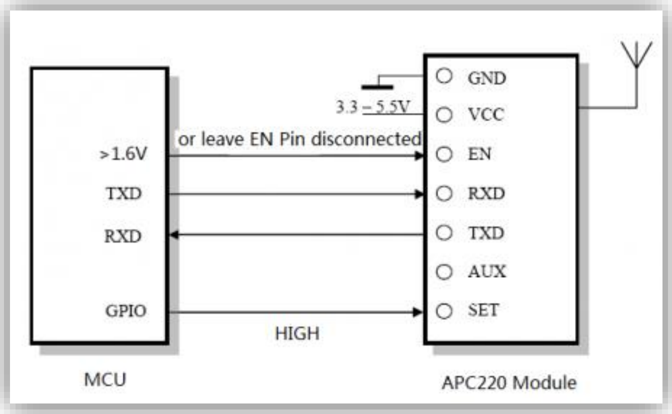


Figure 3-9: TTL interface board Layout

3.5 Raspberry Pi Specification

A Raspberry Pi is a credit card-sized computer originally designed for education, inspired by the 1981 BBC Micro. It uses a many different kinds of processors, so can’t install Microsoft Windows on it. But can install several versions of the Linux operating system that appear and feel very much like Windows. Raspberry Pi is also used to surf the internet, to send an email to write a letter using a word processor, but you can to do so much more. Simple to use but powerful, affordable and in addition difficult to break, Raspberry Pi is the perfect device for aspiring computer scientists [1].

This small computer features amazing HD (high-definition) quality, video playback, also sports high quality audio and has the capability to play 3D games. The device use the ARM processor which does nearly all of the hard work in order to run the Raspberry Pi. RASPBIAN, PIDORA, OPENELEC, RASPBMC, RISC OS, and ARCH LINUX these are few software's which are used. All this software's can be downloaded easily and these are free from the official forum under the NOOBS (new out of the box software) category. It supports Python as the main programming language for functioning and coding. It also supports BASIC, C, C++, JAVA, and Perl and Ruby languages [2].

The Raspberry Pi is slower than a modern laptop or desktop but is still a complete Linux computer and can provide all the expected abilities that implies, at a low-power consumption level. The Raspberry Pi is open hardware, with the exception of the primary chip on the Raspberry Pi, the Broadcom SoC (System on a Chip), which runs many of the main components of the board–CPU, graphics, memory, the USB controller, etc. Many of the projects made with a Raspberry Pi are open and well-documented as well and are things you can build and modify yourself.

The Raspberry Pi Foundation has just recently released a new model, the Raspberry Pi 3, which supersedes some of the previous boards, although the older boards will still be produced as long as there is a demand for them. It is generally backwards compatible with previous versions of the board.

There are a two Raspberry Pi models, the A and the B, named after the aforementioned BBC Micro, which was also released in a Model A and a Model B. The A comes with 256MB of RAM and one USB port. It is cheaper and uses less power than the B. The current model B comes with a second USB port, an Ethernet port for connection to a network, and 512MB of RAM. The Raspberry Pi A and B boards been upgraded to the A+ and B+ respectively. These upgrades make minor improvements, such as an increased number of USB ports and improved power consumption, particularly in the B+. The A+ and B+ have been reviewed on Opensource.com here.

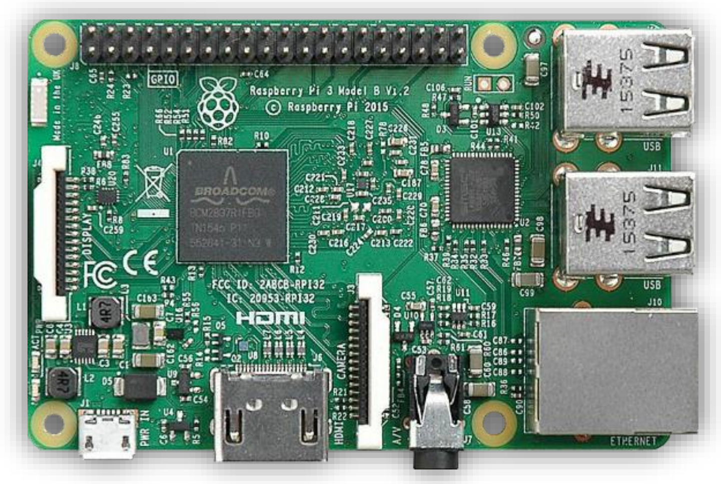


Figure 3-10: Raspberry pi V2 Board

3.5.1 Raspberry Pi hardware specifications

We will briefly go over some of the core components that make up the Raspberry Pi to give you a better feel for what it is capable of. The Raspberry Pi is built off the back of the Broadcom BCM2835. The BCM2835 is a multimedia application processor geared towards mobile and embedded devices. On top of this, several other components have been included to support USB, RCA, and SD card storage.

We will now look at some of the core-components of the Raspberry Pi board. The following figure highlights some of these with a description of each provided:

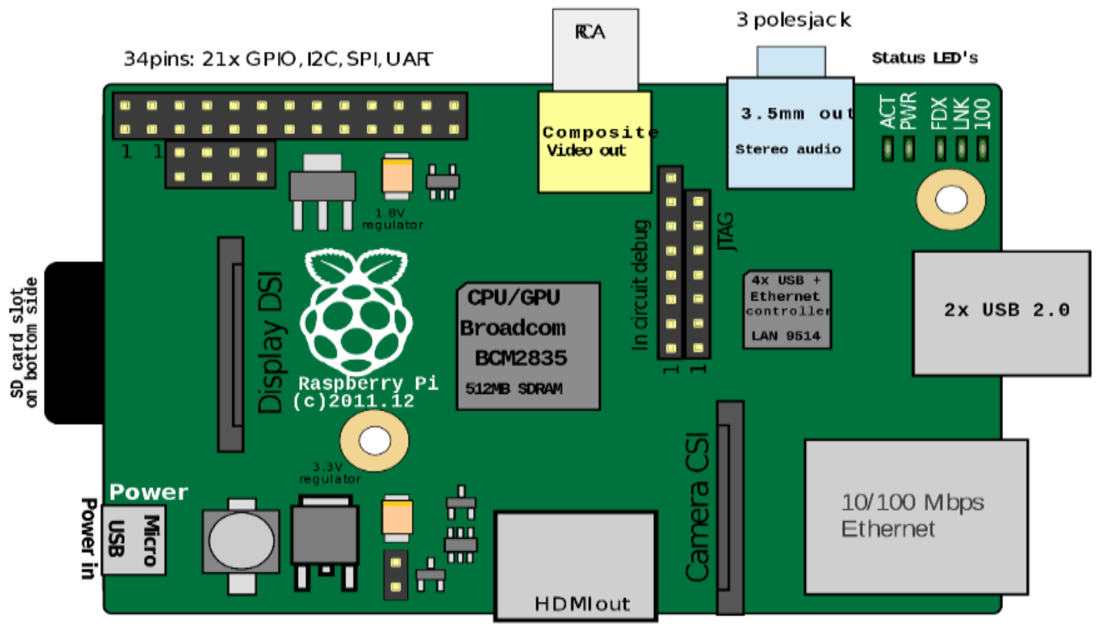


Figure 3-11: Raspberry pi Layout

The Raspberry Pi is a small device coming in at 85.60mm x 53.98mm x 17mm and weighing only 45g. This makes it perfect for home automation, where a small device can be placed in a case and mounted inside an electrical box, or replace an existing thermostat device on a wall. boards such as the Arduino.

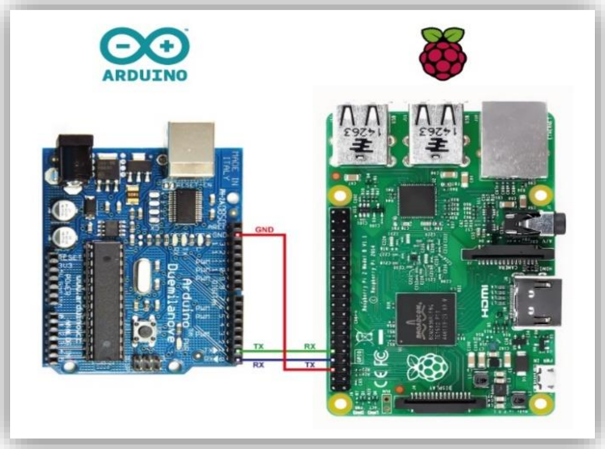


Figure 3-12: Raspberry pi Arduino Peripherals Connections

As the name suggests, the GPIO pins can accept input and output commands and thus can be programmed on the Raspberry Pi. The Arduino shields will be attached to the GPIO via a bridge shield allowing us to transfer data from sensors soldered to the device back to the Raspberry Pi. This is especially useful in home automation projects, where we may wish to store sensor data or manipulate motors based upon a program running on the Raspberry Pi's operating system. Having touched upon the technical capabilities of the Raspberry Pi, we will now look at the Arduino and the Raspberry Pi to Arduino shield, a way to connect the two technologies via the GPIO pins.

3.6 System implementation, and operation

In this chapter, we are going to describe the system implementation including design, operation as well as discussing results that achieves to aims of the project. The proposed health treatment system is constructed from two subsystems, one which is attached to the human body and is call Sensor Station (SS) and the Home station (HS) which coordinate the collection, analysis and transmission of information between the Sensor Station and the public server.

3.7 Implementation of Sensors Node

The Sensor Station is responsible of reading the human body sensing status like the blood pressure, body temperature and the level of hemoglobin of the patient and then encrypt this information in a simple lightweight cryptographic scheme. The encrypted collected information is then send serially to a low cost, short range wireless module APC220.

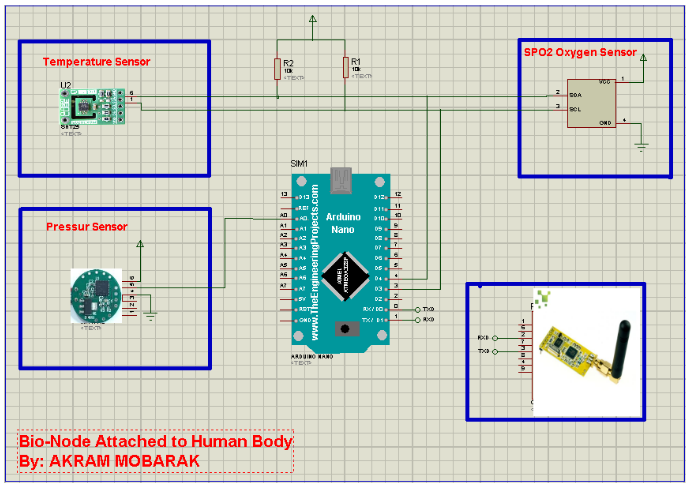


Figure 3-13: Sensor Station Circuit

3.7.1 Reading Sensors Values

The MAX30205 temperature sensor accurately measures temperature and provide an over- temperature alarm/interrupt/shutdown output. This device converts the temperature measurements to digital form using a high-resolution, sigma-delta, analog-to-digital converter (ADC). Communication is through an I^{2}C-compatible 2-wire serial interface.

The communication between MAX30205 sensor and the Arduino is going through I2C communication bus, the SDA pin of the MAX30205 is connected to the SDA of the atmega328p and the while the SCL pin of the sensor is connected to the SCL of the MCU. The MCU in appendix A initializes the operation of the tow wire interface TWI to allow the sensors to stablish a valid connection with the MCU. This initialization can be done by setting up the Two Wire Bitrate Register TWBR and Two Wire Control Register TWCR.

The I^{2}C serial interface accepts standard write byte, read byte, send byte, and receive byte commands to read the temperature data and configure the behavior of the open-drain over-temperature shutdown output. Therefore, the TWI bus line check whether there is incomplete operation, this operation can be either an ongoing transmission of SPO2 data or previous reading from MAX30205. The check could be done by continuously monitoring if there is an interrupt is occurring on the TWIE bit on TWCR. At appendix B the function is return 1 if operation is successful else it will return 0.

The MAX30205 features three address select lines with 32 available addresses. The sensor has a 2.7V to 3.3V supply voltage range, low 600µA supply current, and a lockup-protected I^2C-compatible interface that make them ideal for wearable fitness and medical applications.

The MCU will send the address of the MAX30205 using unsigned char TWI_drvr_writeregister (); This function accesses the MAX30205 slave at slave address and writes REG_DATA to its internal register at REG_ADDR. The function returns 1 if successful, else, it returns 0.

Now the sensor has been ready to send the scenery data to the Arduino board, the microcontroller receives the data using unsigned char TWI_readregister (). This function accesses the slave at slave address, reads its internal register at REG_ADDR, and stores the data in received Data pointer. The function returns 1 if successful, else, it returns 0

The same procedure is done with MAX30100 oxygen sensor since it’s also a two wire interface module.

In order to read the data from a blood pressure sensor we must allow the ADC of the microcontroller by setting the sampling rate, the reference voltage as well as to insure that the selected channel is configured as analogue input pin.

Here channel 0 in DIDR0 register is select as analogue input channel while the reference value is set to 5V as the AVCC value so the AREF pin of the Atmega328P must be connected to the ground via coupling capacitor.

When the ADC channel detect an analogue signal the MCU start the conversion by sampling this signal and the quantize it into a finite level, and then coded the quantized levels into 10-bit resolution digital form stored in ADC data register (ADH and ADL). During the conversion of the analogue signal the MCU remains busy until ADIF bit in the ADSR is set to 1 after that the MCU is going through next conversion.

3.7.2 Sensors Data encryption

The sensing data is encrypted by advanced encryption standard (AES) using a Secure Rolling Code Algorithm transmission protocol and its cipher-based message authentication code (CMAC) mode of operation for transmitter authentication supporting 128-bit key size with less than 30ms response time. The transmission protocol can be used in a system consisting of one receiver and a limited number of associated transmitters.

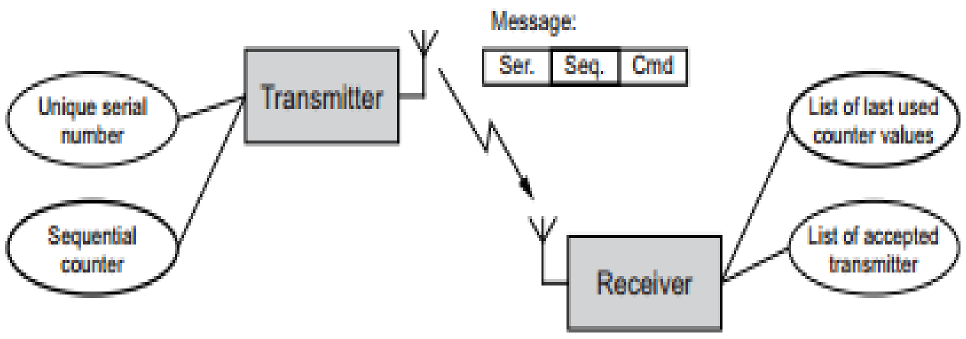


Figure 3-14: Secure System with all goals satisfied

The transmitters are preprogrammed with a unique identification number and a secret encryption key. To be able to recognize and accept a new transmitter, the secret key must be transferred to the receiver in a secure manner. It is important that the transmission from the transmitter is unreadable for all but the receivers belonging to the same system. It is also important that the receiver only accept transmitters belonging to the same system.

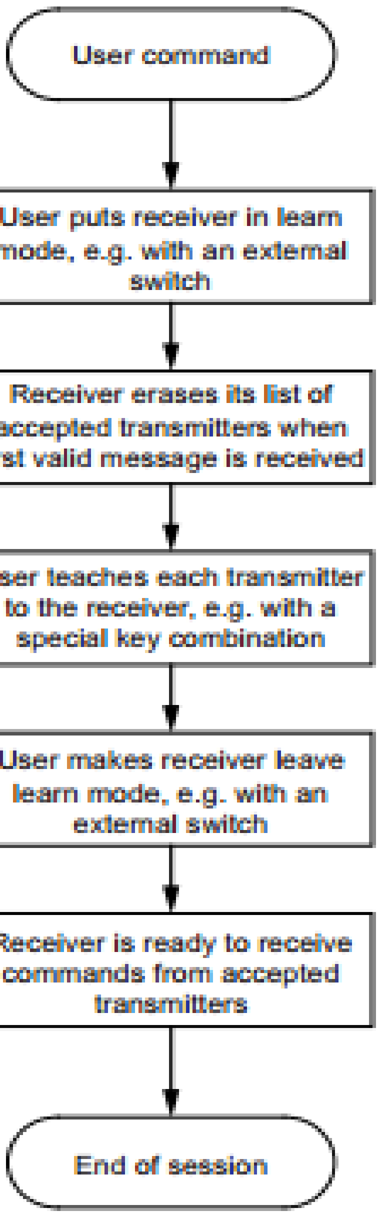

User command
User puts receiver in learn mode, e.g. with an external switch
Receiver erases its list of accepted transmitters when first valid message is received
User teaches each transmitter to the receiver, e.g. with a special key combination
User makes receiver leave learn mode, e.g. with an external switch
Receiver is ready to receive commands from accepted transmitters
End of session

Figure 3-15: Flow off learning Session

The key transfer process is initiated by putting the receiver into a special learn mode using an external signal, e.g., a switch hidden inside the car. On the transmitter side, the learning process is initiated by e.g., a special key combination. The transmitter encrypts its secret key using the shared key and transmits it to the receiver together with the serial number and optionally the sequential counter state. Due to this scheme, the shared key is often referred to as a key encryption key (KEK)

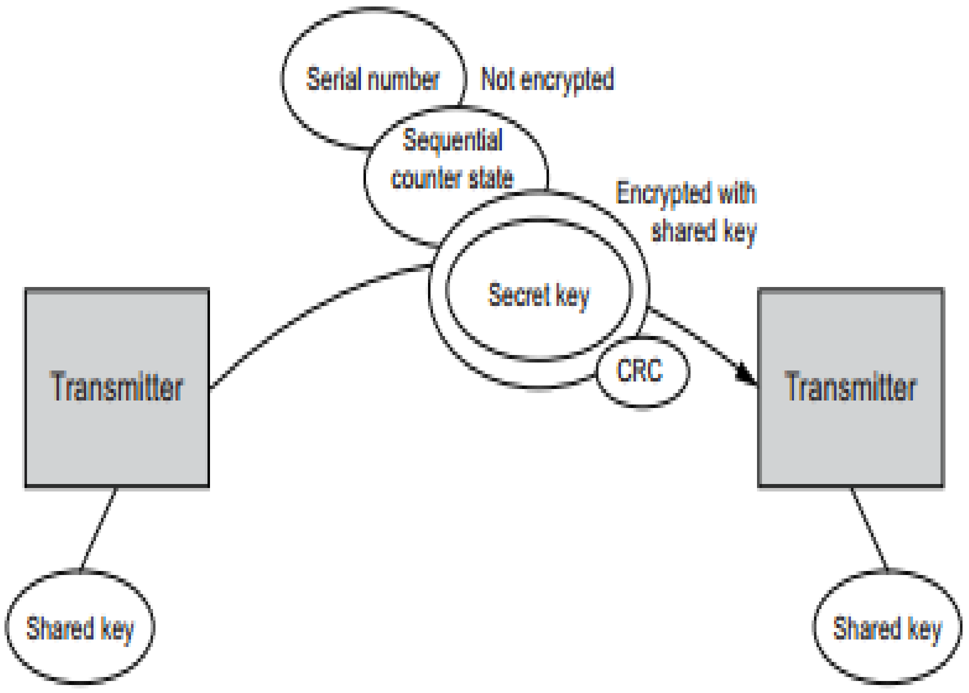


Figure 3-16: Secret Key Transfer Session

If one can assume that the transmitter is not operated before it is used in a key transfer session, the counter state can be omitted from the transmission, resulting in a shorter transmission. Even if the transmitter has been operated a few times, it will be accepted as long as the counter has not counted past the rolling window size. The implementation in this application note includes the counter value in the transmission. To ensure message integrity, a CRC code generated from the whole message frame is also appended to the message.

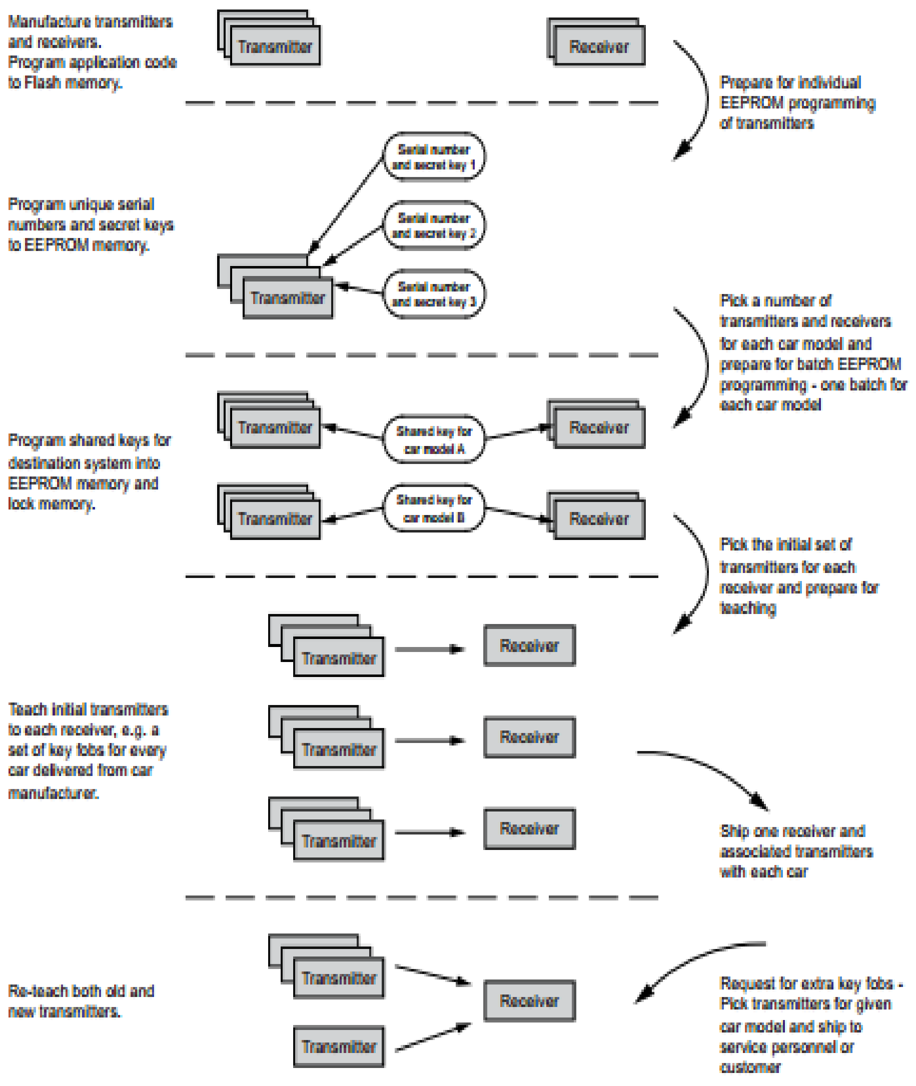

Manufacture transmitters and receivers. Program application code to Flash memory.
Transmitter
Receiver
Prepare for individual EEPROM programming of transmitters
Serial number and secret key 1
Serial number and secret key 2
Serial number and secret key 3
Program unique serial numbers and secret keys to EEPROM memory.
Transmitter
Pick a number of transmitters and receivers for each car model and prepare for batch EEPROM programming - one batch for each car model
Transmitter
Shared key for car model A
Receiver
Program shared keys for destination system into EEPROM memory and lock memory.
Transmitter
Shared key for car model B
Receiver
Pick the initial set of transmitters for each receiver and prepare for teaching
Transmitter
Receiver
Teach initial transmitters to each receiver, e.g. a set of key fobs for every car delivered from car manufacturer.
Transmitter
Receiver
Ship one receiver and associated transmitters with each car
Transmitter
Receiver
Re-teach both old and new transmitters.
Transmitter
Receiver
Transmitter
Request for extra key fobs - Pick transmitters for given car model and ship to service personnel or customer

Figure 3-17: Overview of Transmitter and Receiver lifecycle

3.7.3 Operation

The transmitter spends most of its time in Power Down Sleep Mode, waiting for the user to press a button. When a Pin Change Interrupt wakes up the ATmega328p, the button selection is compared to a defined Teach Command button. (In the appendix C) If it matches, the transmitter encrypts and transmits its secret key using the system's shared key. Before entering Power Down Sleep Mode again, the sequential counter value is incremented.

3.8 Wireless Interface

After the sensors information's are encrypted now this information's are ready for transmission through air interface using APC220 wireless module.

For the first time we need to initiate by setting up the baud rate, the number of bits per and the buffer size of the USART module inside the AVR-MCU.

The configuration at appendix D allows the transmission of 9600bps baud-rate with CPU frequency equal to 16Mhz and 8bit data per frame while enabling the transmitter and the receiver as well as receive interrupt is enabled also. Now the microcontroller is ready to transmit the information to the raspberry by using APC220 wireless module; the following code at appendix E is responsible for the establishment of the transmission.

Configuring APC220 wireless module Download and install the configuration software for the USB to TTL converter and then go to the device manager in the windows computer check if the software was install successfully.

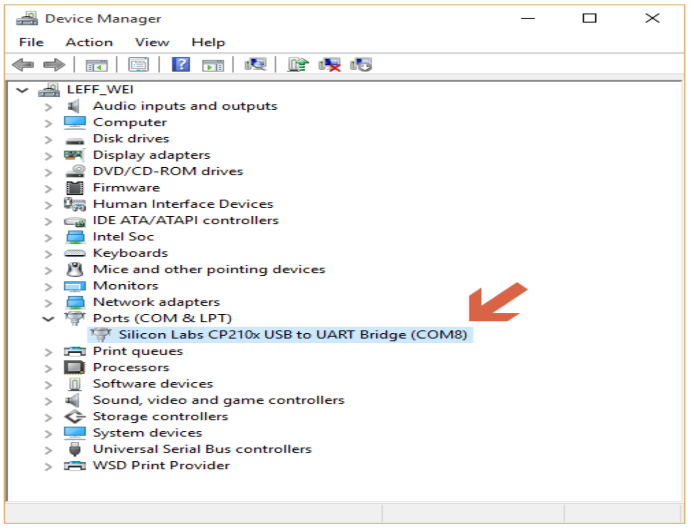


Figure 3-18: Serial Port Configuration

Figure 3.18 show the final configuration of the RF module APC220, after installation of the driver is completed, go the device manager of your windows computer by write click on computer icon then choose mange, the above navigation list will be open, go to port setting and find what is the preferred port for the module, its optional and can be changed.

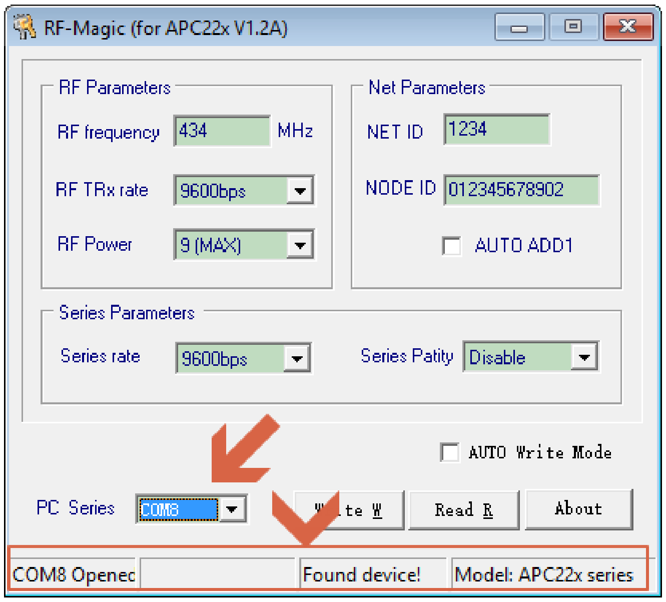
CHAPTER FOUR

RESULTS AND DISCUSSIONS

Chapter Four

Results and Discussions

Now by running the RF-Magic executable file by write click on ap22x.exe application and then choose open as administrator, the window in the figure 4.7 will immediately opened, this allows the programmer to select the desired configuration for his own project by choosing the bandwidth of transmission, the network ID as well as node ID and also the user has to choose the physical and air baud rate of transmission.



RF-Magic (for APC22x V1.2A)

RF Parameters
RF frequency 434 MHz
RF TRx rate 9600bps
RF Power 9 (MAX)

Net Parameters
NET ID 1234
NODE ID 012345678902
AUTO ADD1

Series Parameters
Series rate 9600bps
Series Patity Disable

AUTO Write Mode
PC Series COM8

COM8 Openec | Found device! | Model: APC22x series

Figure 4-1: RF Transmitter Setting

In the figure 4.18 we connect the APC220 to the windows computer through USB to TTL converter, then select the appeared COM port, the module will appear in the status bar along with the port number and the model series of the RF Module. Figure 4.7 shows the selected node parameters as an example, the demonstration in my thesis is worked under 434 Mhz band, the network ID can be any number between 0-0xFFFF, but the network ID must be unique number in each network because it tells that all nodes have the

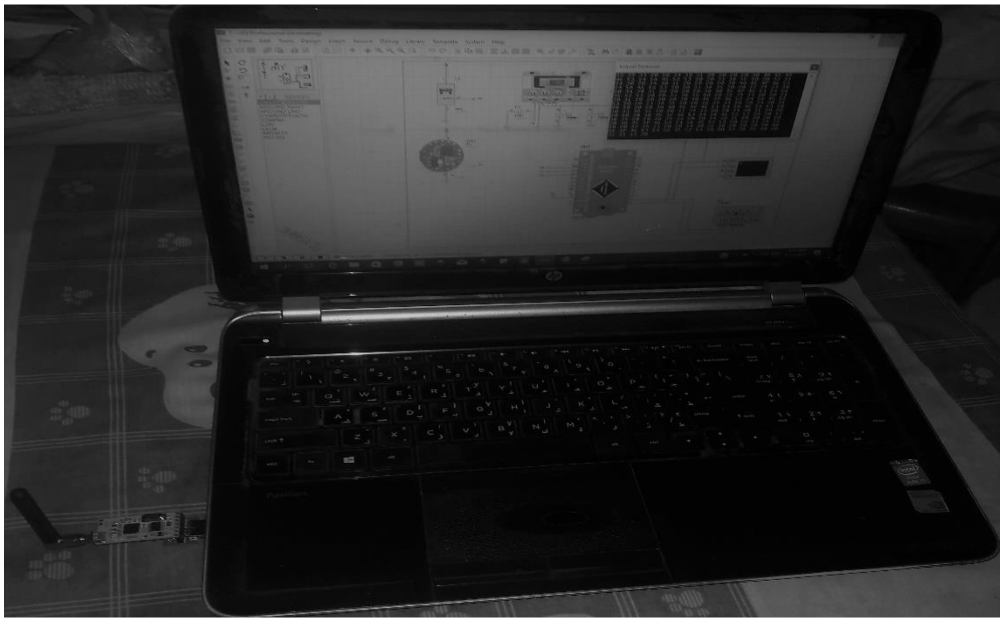
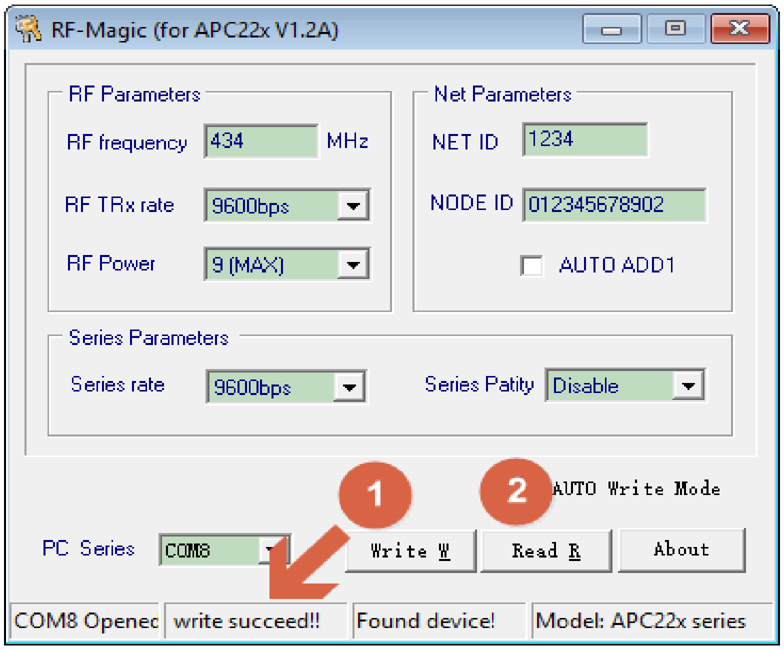


Figure 4-2: APC220 connection with PC

same network ID are allowed to be communicate with each other but not with the same nodes having different network ID, while the node



RF-Magic (for APC22x V1.2A)

RF Parameters

RF frequency 434 MHz

RF TRx rate 9600bps

RF Power 9 (MAX)

Net Parameters

NET ID 1234

NODE ID 012345678902

AUTO ADD1

Series Parameters

Series rate 9600bps

Series Patity Disable

AUTO Write Mode

PC Series COM8

COM8 Openec | write succeed!! | Found device! | Model: APC22x series

Figure 4-3: RF Receiver Setting

The node id is an optional 32-bit number to define the node of the sensor, also we use a baud rate of 9600 bps for both the physical and air baud rates.

Table 4-1: Specification of RF Sensor

Parameter	Range	Default
RF frequency	Resolution 1KHz， Accuracy ±100Hz	434MHz
RF TRx Rate	1200， 2400， 4800， 9600， 19200bps	9600bps
RF power	0-9	9
Series Rate	1200， 2400， 4800， 9600， 19200， 38400， 57600bps	9600bps
Net ID	0-65535（16 bit）	12345
Node ID	123456789012	
Series Parity	Disable， Odd Patity， Even Patity	Disable

4.1 Writing the Raspbian image

Insert the SD card into the laptop/pc and run the image writer. Once open, browse and select the downloaded Raspbian image file. Select the correct device, that is the drive representing the SD card. If the drive (or device) selected is different from the SD card, then the other selected drive will become corrupted. After that, click on the "Write" button in the bottom.

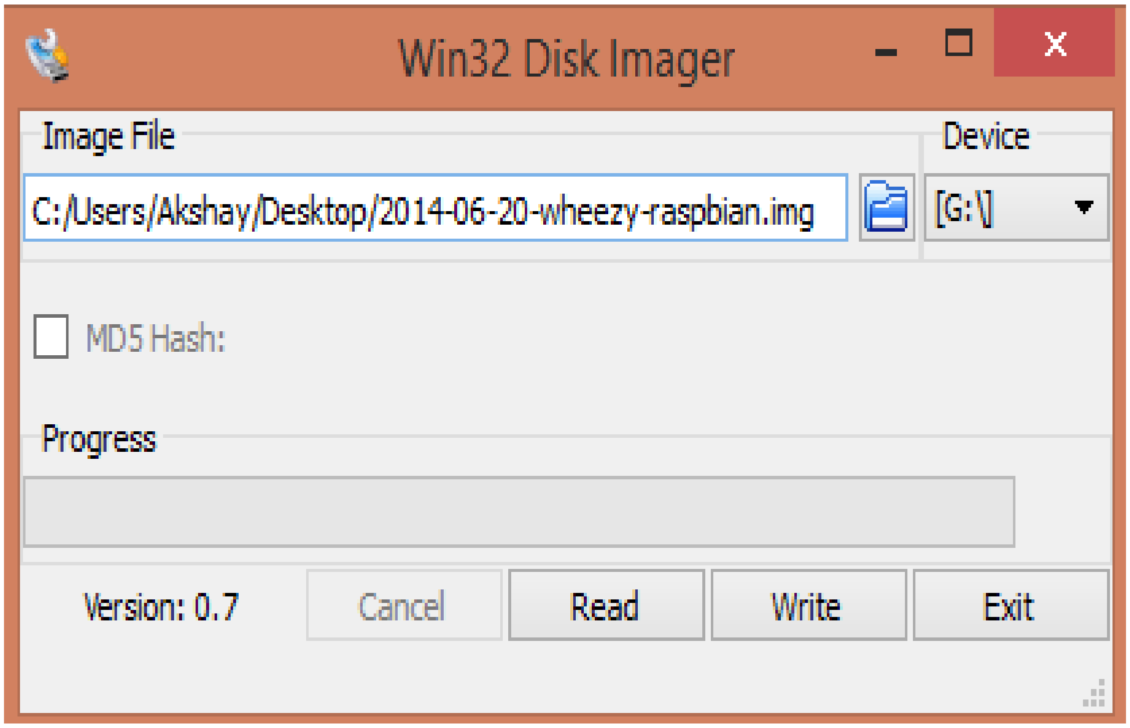


Image File: C:/Users/Akshay/Desktop/2014-06-20-wheezy-raspbian.img

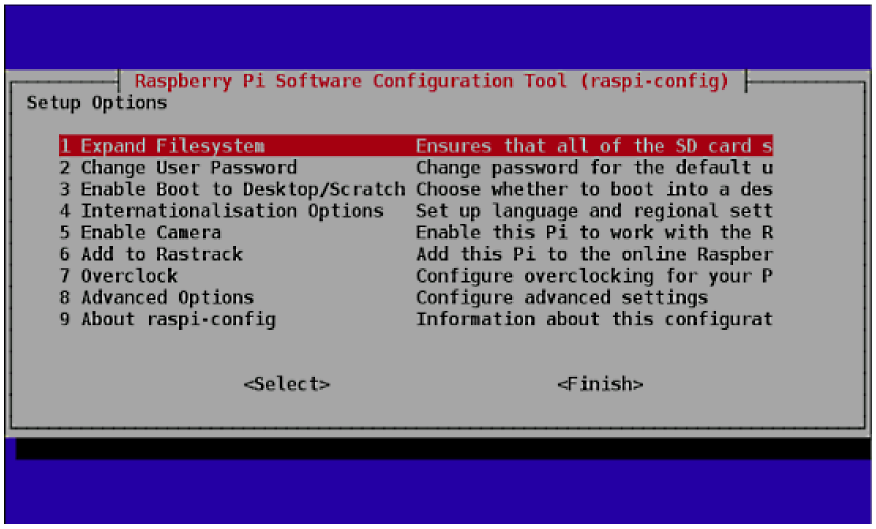
Device: [G:\]

Figure 4-4: Image File Burn at SD Card

Once the write is complete, eject the SD card, insert it into the Raspberry Pi, and turn it on. It should start booting up. Setting up the Pi Raspberry Pi comes with a default user name and password and so always use it whenever it is being asked.

```
login: pi
password: raspberry
```

When the Pi has been booted for the first time, a configuration screen called the "Setup Options" should appear and it will look like the image below.



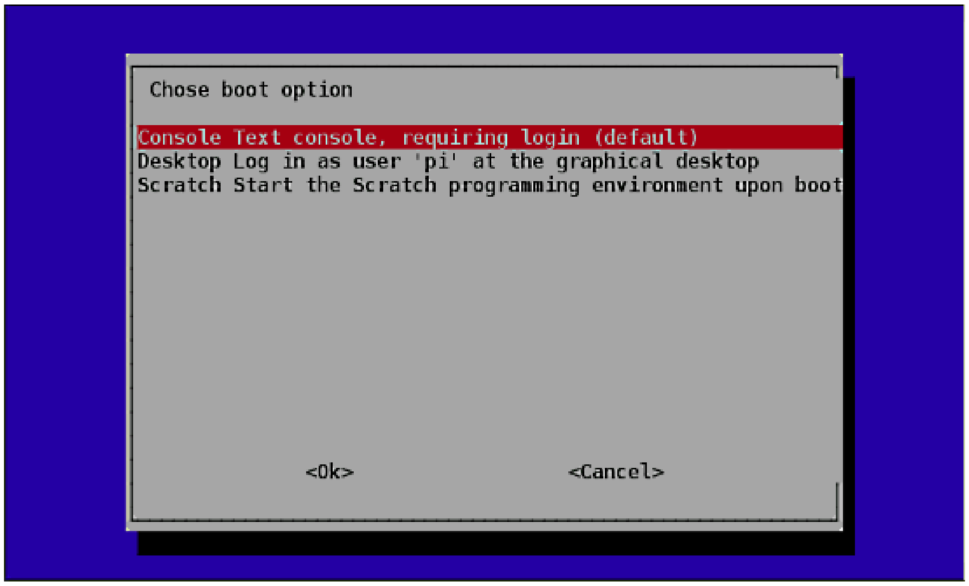
```
Raspberry Pi Software Configuration Tool (raspi-config)
Setup Options

    1 Expand Filesystem              Ensures that all of the SD card s
    2 Change User Password           Change password for the default u
    3 Enable Boot to Desktop/Scratch Choose whether to boot into a des
    4 Internationalisation Options   Set up language and regional sett
    5 Enable Camera                  Enable this Pi to work with the R
    6 Add to Rastrack                Add this Pi to the online Raspber
    7 Overclock                      Configure overclocking for your P
    8 Advanced Options               Configure advanced settings
    9 About raspi-config             Information about this configurat

                  <Select>                     <Finish>
```

Figure 4-5: Raspberry Pi Configuration Tool

Now that the Setup Options window is up, we will have to set a few things. After completing each of the steps below, if it asks to reboot the Pi, please do so. After the reboot, if you do not get the "Setup Options" screen, then follow the command given above to get the screen/window.

Select the first option in the list of the setup options window, that is select the "Expand Filesystem" option and hit the enter key. We do this to make use of all the space present on the SD card as a full partition. All this does is, expand the OS to fit the whole space on the SD card which can then be used as the storage memory for the Pi.

Select the third option in the list of the setup options window, that is select the "Enable Boot To Desktop/Scratch" option and hit the enter key. It will take you to another window called the "choose boot option" window that looks like the figure 4-6 below.



```
Chose boot option

Console Text console, requiring login (default)
Desktop Log in as user 'pi' at the graphical desktop
Scratch Start the Scratch programming environment upon boot

            <Ok>                    <Cancel>
```

Figure 4-6: Option for Raspberry Pi

In the "choose boot option window", select the second option, that is, "Desktop Log in as user 'pi' at the graphical desktop" and hit the enter button. Once done you will be taken back to the "Setup Options" page, if not select the "OK" button at the bottom of this window and you will be taken back to the previous window. We do this because we want to boot into the desktop environment which we are familiar with. If we don't do this step, then the Raspberry Pi boots into a terminal each time with no GUI options.

Once, both the steps are done, select the "finish" button at the bottom of the page and it should reboot automatically. If it does not, then use the following command in the terminal to reboot.

```
sudo reboot
```

After the reboot from the previous step, if everything went right, then you will end up on the desktop, which looks like the image below.

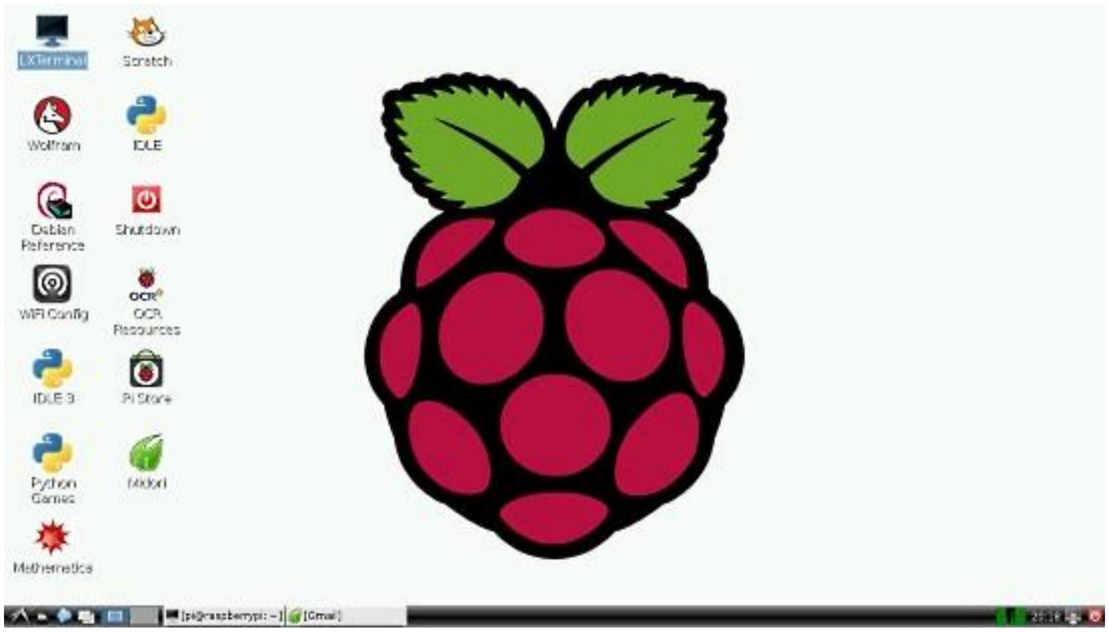


Figure 4.7 Raspberry Pi Desktop

Once you are on the desktop, open a terminal and enter the following command to update the firmware of the Pi.

```
sudo rpi-update
```

Updating the firmware is necessary because certain models of the Pi might not have all the required dependencies to run smoothly or it may have some bug. The latest firmware might have the fix to those bugs, thus it’s very important to update it in the beginning itself.

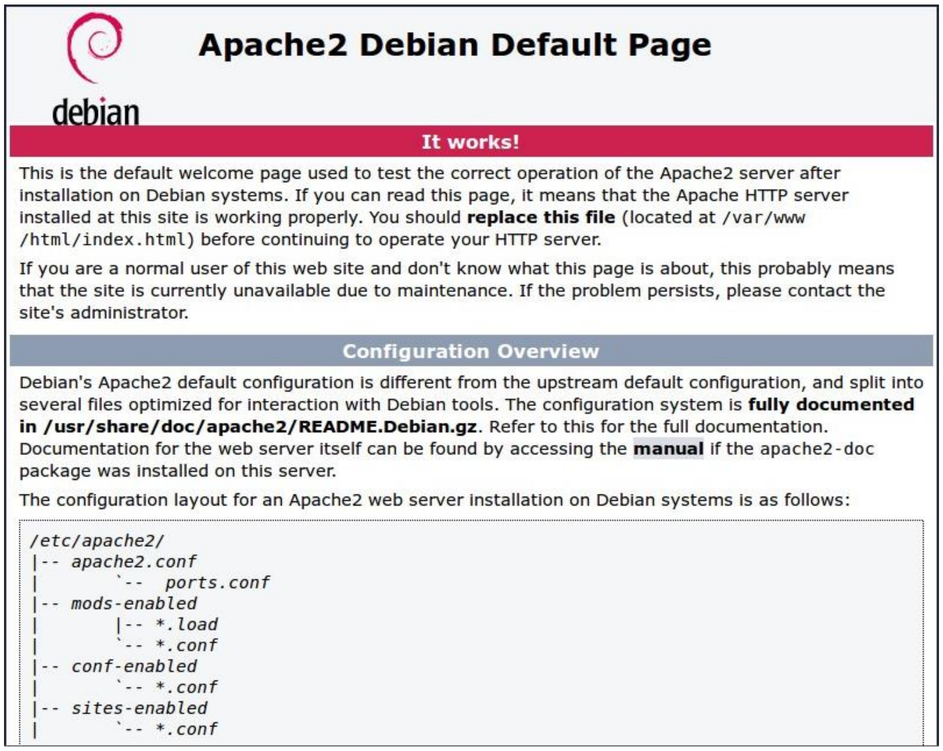
4.2 Installing Apache server on Raspberry Pi

Apache is a popular web server application you can install on the Raspberry Pi to allow it to serve web pages. First install the apache2 package by typing the following command in to the Terminal:

```
sudo apt-get install apache2 -y
```

By default, Apache puts a test HTML file in the web folder. This default web page is served when you browse to http://localhost/ on the Pi itself, or http://192.168.1.13 from another computer on the network. To find the Pi's IP address, type hostname -I at the command line.

Browse to the default web page either on the Pi or from another computer on the network and you should see the following:



Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at /var/www/html/index.html) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

Figure 4-8: Apache Default Page

This means you have Apache working. This default web page is just a HTML file on the file system. It is located at

```
/var/www/html/index.html.
```

Navigate to this directory in a terminal window and have a look at what is inside

```
cd /var/www/html
ls -al
```

This will show you:

```
total 12
drwxr-xr-x  2 root root 4096 Jan  8 01:29 .
drwxr-xr-x 12 root root 4096 Jan  8 01:28 ..
-rw-r--r--  1 root root  177 Jan  8 01:29 index.html
```

4.3 Pi Sensor Node

The proposed model allows Arduino and Raspberry Pi users to perform biometric and medical applications where body monitoring is needed by using different kinds of bio-sensors. This information can be used to monitor in real time the state of a patient or to get sensitive data in order to be subsequently analyzed for medical diagnosis. Biometric information gathered can be wirelessly sent using any of the 6 connectivity options available: Wi-Fi, 3G, GPRS, Bluetooth, 802.15.4 and ZigBee depending on the application.

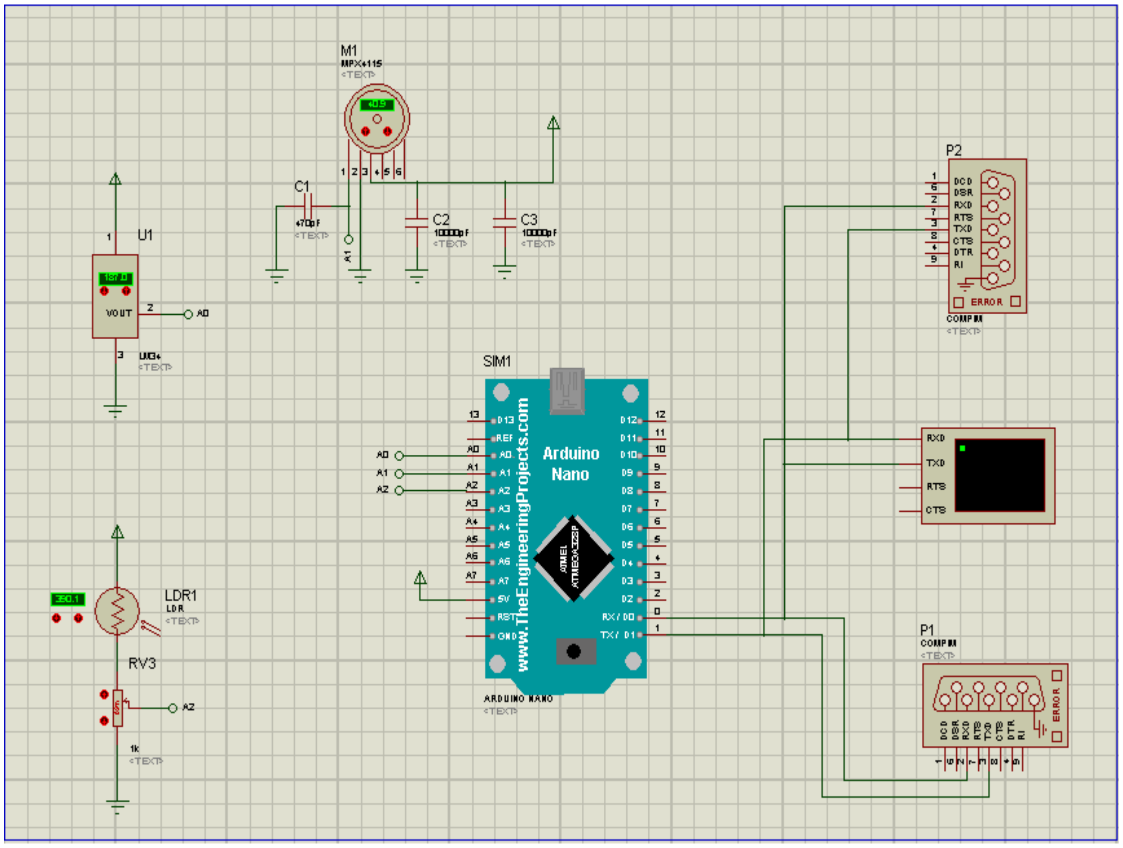
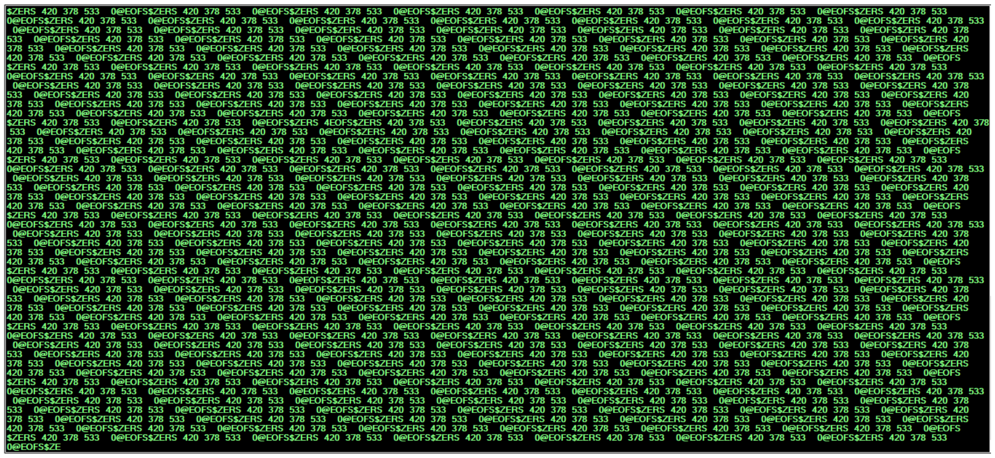


Figure 4-9a: Sensors Node

In figure 4.14a the Arduino NANO sensor node connected to the RF module using COM4 with baud rate of 9600 bps, at the same time it connected to COM2 to visualize the physical data from the terminal, the sensor node collects all sensors data from the ADC channel as well as I2C data, this collected data will have stored in a buffer and then performs the encryption process for the gathered data, the collected data is then being transferred in 9600bps baud rate. In the figure 4.14b we show that this information’s are absolutely secure and safe to be transferred to APC220 wireless module.

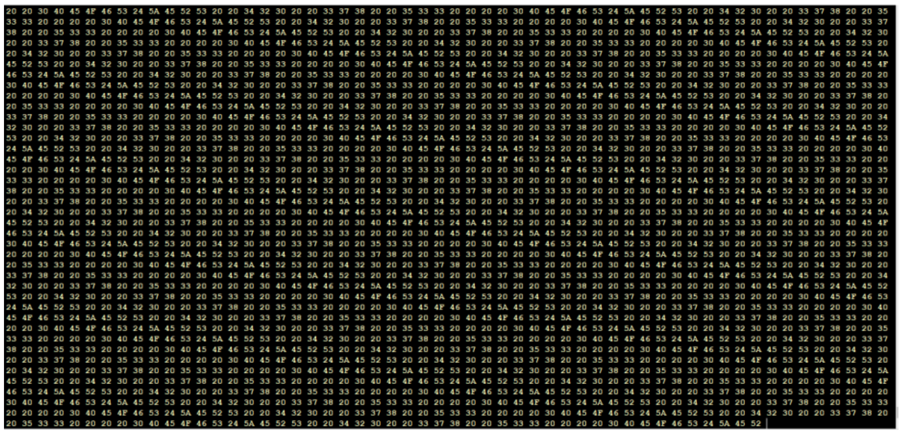


```
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
 0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378
533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420
378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS
420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
 0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378
533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420
378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS
420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 4EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378
 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420
378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS
420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
 0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378
533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420
378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS
420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
 0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378
533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420
378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS
420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
 0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378
533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420
378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS
420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
 0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378
533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420
378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS
420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
 0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378
533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420
378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS
420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS
$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533  0@EOFS$ZERS 420 378 533
0@EOFS$ZE
```

Figure 4-9b: Transmit Data from Sensor Node to Analysis

In the window of figure 4.14b it shows that the encrypted sensors before transmission, this includes headers, counter and tail of the transmitted frame. While the window in figure 4.14c shows that the air transmitted data is exactly the as the data at the microcontroller terminal thus the transmission speed is no effected by extra redundancy. This information can be used to monitor in real time the state of a patient or to get sensitive data in order to be subsequently analyzed for medical diagnosis at the raspberry Pi side.

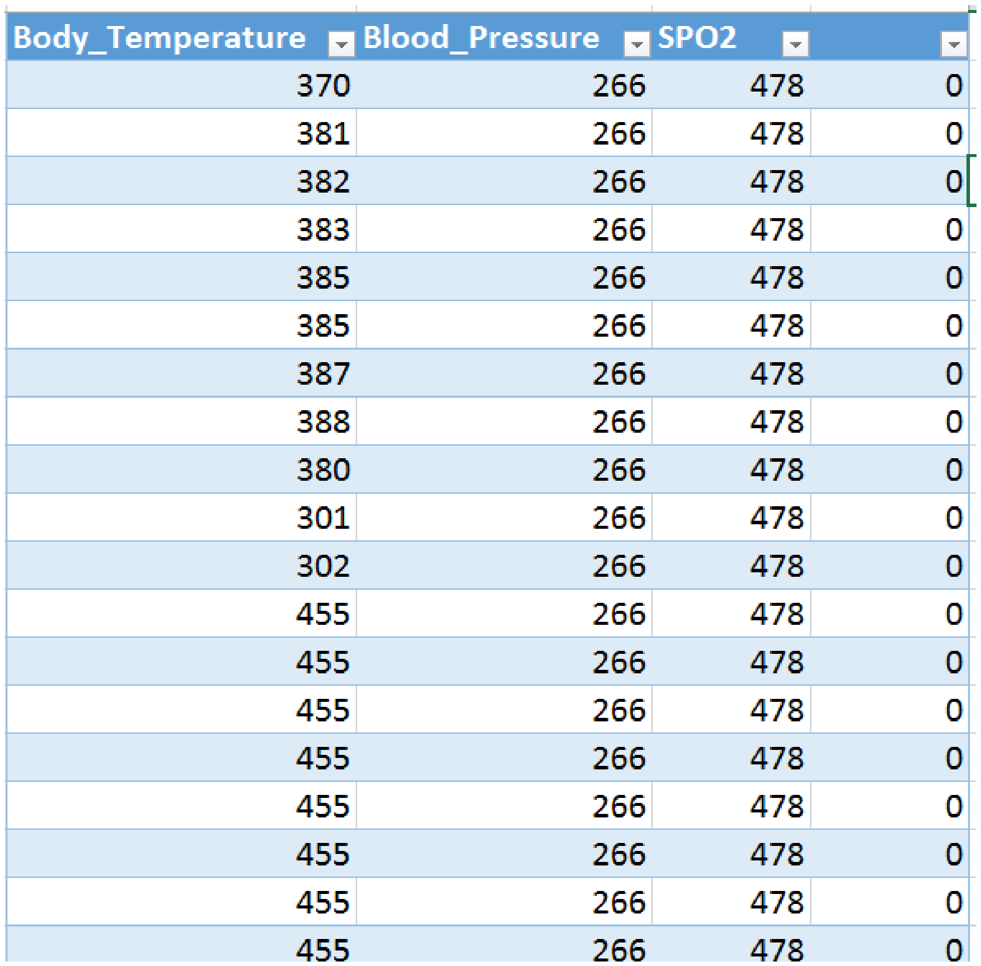
The data is received by the raspberry pi serial port and then the raspberry pi goes to extract the real data from the received frame.



```
20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35
33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37
38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30
20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20
20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A
45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F
46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20
30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33
20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20
20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20
33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34
32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52
53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53
24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40
45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20
20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35
33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37
38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30
20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20
20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A
45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F
46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20
30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33
20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20
20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20
33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34
32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52
53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53
24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40
45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20
20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35
33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37
38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30
20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20
20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A
45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F
46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20
30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33
20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20
20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52 53 20 20 34 32 30 20 20 33 37 38 20 20 35 33 33 20 20 20 20 30 40 45 4F 46 53 24 5A 45 52
```

Figure 4.10 Transmitting Data to MATLAB and Raspberry

The received data is forwarded by the raspberry Pi to an excel file, in the figure 14.5a the received data as mentioned is extracted from the burst frame and then and then reported in its origin as temperature of the patient body, the blood pressure and the oxygen concentration in the blood respectively, the monitored data is represent the real time value of the human body status of all time thus these data has to be averaged over the scale and then it can be visualized using excel or other numerical technique.



Body_Temperature	Blood_Pressure	SPO2	
370	266	478	0
381	266	478	0
382	266	478	0
383	266	478	0
385	266	478	0
385	266	478	0
387	266	478	0
388	266	478	0
380	266	478	0
301	266	478	0
302	266	478	0
455	266	478	0
455	266	478	0
455	266	478	0
455	266	478	0
455	266	478	0
455	266	478	0
455	266	478	0
455	266	478	0

Figure 4-11a: Data at Excel and analysis data

Here the data is visualized in figure 14.15b using drawing technique embedded with excel sheet.

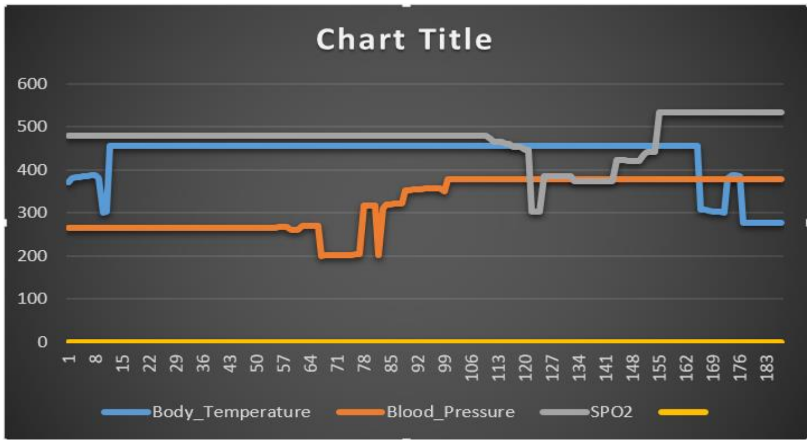
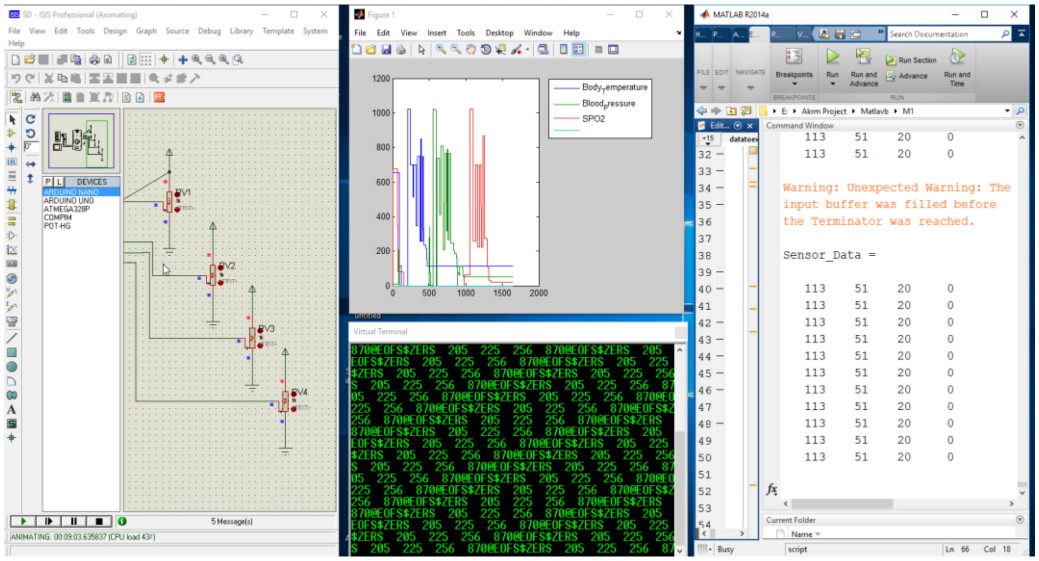


Figure 4-11b: Excel data analysis data

The figure 14.15b represent the visualization of three sensors, the blue one represent the temperature sensory data read from the terminal, while the gray and orange curves represent the pressure of the blood as well as oxygen sensory data.

On the other hand, the Matlab is also used to analyzed the received data, the figure 14.6 shows that the data is received by Matlab and then the received data is extracted using appendix F, the data is then plotted in the computer screen.



```
870@EOFS$ZERS  205  225  256  870@EOFS$ZERS  205
EOFS$ZERS  205  225  256  870@EOFS$ZERS  205  225
$ZERS  205  225  256  870@EOFS$ZERS  205  225  256
S  205  225  256  870@EOFS$ZERS  205  225  256  87
05  225  256  870@EOFS$ZERS  205  225  256  870@EO
225  256  870@EOFS$ZERS  205  225  256  870@EOFS$Z
256  870@EOFS$ZERS  205  225  256  870@EOFS$ZERS
870@EOFS$ZERS  205  225  256  870@EOFS$ZERS  205
EOFS$ZERS  205  225  256  870@EOFS$ZERS  205  225
$ZERS  205  225  256  870@EOFS$ZERS  205  225  256
S  205  225  256  870@EOFS$ZERS  205  225  256  87
05  225  256  870@EOFS$ZERS  205  225  256  870@EO
225  256  870@EOFS$ZERS  205  225  256  870@EOFS$Z
256  870@EOFS$ZERS  205  225  256  870@EOFS$ZERS
870@EOFS$ZERS  205  225  256  870@EOFS$ZERS  205
EOFS$ZERS  205  225  256  870@EOFS$ZERS  205  225
$ZERS  205  225  256  870@EOFS$ZERS  205  225  256
S  205  225  256  870@EOFS$ZERS  205  225  256  87
```

```
   113    51    20     0
   113    51    20     0

Warning: Unexpected Warning: The
input buffer was filled before
the Terminator was reached.

Sensor_Data =

   113    51    20     0
   113    51    20     0
   113    51    20     0
   113    51    20     0
   113    51    20     0
   113    51    20     0
   113    51    20     0
   113    51    20     0
   113    51    20     0
   113    51    20     0
   113    51    20     0
```

Figure 14-12: the received data at the MATLAB

The data is then plotted using Matlab, this data can be then used by the doctors for diagnosis purpose and can be archived to a data base in the public server.

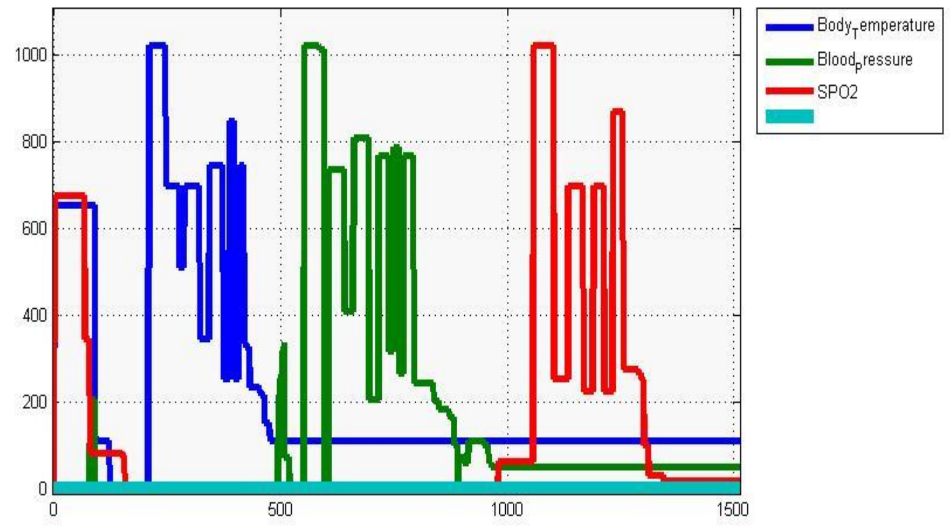


Figure 4-17: Sensors data plot using Matlab

Data can be sent to the Cloud in order to perform permanent storage or visualized in real time by sending the data directly to a laptop or Smartphone. iPhone and Android applications have been designed in order to easily see the patient's information.

CHAPTER FIVE
CONCLUSION AND RECOMMENDATIONS

Chapter Five
Conclusion and Recommendations

5.1 Conclusion

As the conclusion, in this project a prototype of raspberry pi based health treatment system is represented. Where the sensory data is collected with small size, low power consumption and low cost Arduino NANO node achieving the first goal. Where the body temperature data is collected using analogue to digital converter of the atmega328P while the blood pressure data and the oxygen's is captured via two-wire communication. The collected sensors information is then encrypted via simple encryption scheme and transferred wirelessly via APC220 ZigBee wireless module, this data is then captured, analyzed, and stored and visualized using raspberry Pi. The patients' temperature signal, blood pressure and O2 can be monitored remotely anywhere and anytime. This information also can be send to a cloud server to be permanently stored and can be used for future requirements.

5.2 Recommendations

We recommend that to use real life sensors to demonstrate this prototype, and analyze this data in a real network case, also we recommend that to use Wi-Fi network instead of RF module that increase the level of security and keeps the consuming levels of power as low as possible [look at the datasheet].

References

[1] L. F. John A. Stankovic, "Research Directions for the Internet of," 2014.

[2] S. S. D. R. P. P. D. o. C. E. S. J. Sathish Kumar Department of Computer Engineering, "A Survey on Internet of Things: Security and Privacy Issues," *international Journal of Computer Applications,* vol. 90 –No 11, p. 0975 –8887, 2014.

[3] C. W. Ahmad-Reza Sadeghi and M. Waidner, "Security and Privacy Challenges in Industrial Internet of Things," *Technische Universität Darmstadt, Germany Intel CRI-SC at TU Darmstadt, GermanyFraunhofer Institute for Secure Information Technology, Darmstadt,Germany,* no. text book, pp. 7-11, 2015.

[4] O. G.-M. R. H. ,. L. K. S. K. a. K. W. C. G. R. A. G. a. P. R. Tobias Heer, "Security Challengesin the IP-based Internet of Things," *Journal on Wireless Personal Communications,* 2012.

[5] International Electro technical Commission, "Inrenet of Things: Wireless Sensor Networks," *Registered trademark of the International Electro technical Commission,* Copyright © IEC, Geneva, Switzerland 2014..

[6] M. B. B. C. Wueest, "Insecurity in the Internet of," vol. Version 1.0, March 12, 2015.

[7] T. P. J. B. H. K. a. D. E. Jatinder Singh, "Twenty security considerations for cloud-supported Internet of Things," *IEEE 1http://www.rfidjournal.com/articles/view?4986,,* 2015.

[8] Y. C. ,. E. N. Z. C. B. Arbia Riahi, "A systemic approach for IoT security," *Université de Technologie de Compiègne, Compiègne, France.,* 2015.

[9] M. C. a. t. I. o. T. E. M. Communication.

[10] O. G. M. a. K. W. Jan Henrik Ziegeldorf1, "Privacy in the Internet of Things: Threats and Challenges," SECURITY AND COMMUNICATION NETWORKS Security Comm., 2013.

[11] J. S. S. K. Arijit Ukil, "Embedded Security for Internet of Things," 2010.

[12] F. M. a. C. Floerkemeier, "From the Internet of Computers to the Internet of Things," *Distributed Systems Group, Institute for Pervasive Computing,* no. ETH Zurich, 2011.

[13] T. B. D. o. I. Engineering, "Corresponding author "Survey of Security and Privacy Issues of Internet of Things," 2015.

[14] L. L. a. A. S. V.-c. C. I. E. U. o. A. S. W. S. (.-S. S. V. S. j. C. Antonio J. Jara, "The Internet of Everything through IPv6: An Analysis of Challenges, Solutions and Opportunities," *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications,* vol. 4, pp. 97-118, 2014.

[15] E. B. D. o. C. S. P. U. W. Lafatette, "Data Security and Privacy in the IoT," in *19th International Conference on Extending Database Technology*, Bordeaux, France, March 15-18 2016.

[16] V. S. S. S. Y. A. C. X. C. M. U. P. U. Tianlong Yu, "Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the Internet-of-Things," Philadelphia, PA USA, 15 November 16.

[17] A. K. B. P. C. E. &. P. Ashvini Balte, "Security Issues in Internet of Things (IoT): A Servey," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 5, p. 450, 2015.

[18] Q. J. •. A. V. V. •. J. W. •. J. L. •. Q. Dechao, "Security of the Internet of Things: perspectives and challenges," in *Springer Science+Business Media* , New York, 2014.

[19] M. S. S. E. D. G C Harikiran Karthik, "Smart Security Solution for Women based on Internet Of Things(IOT)," in *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, 2016.

Appendix A

```
void twi_initial(void)
{
  TWBR = TWI_TWBR;
  TWSR = TWI_TWPS;
  TWDR = 0xFF;
  TWCR = (1<<TWEN)|
         (0<<TWIE)|(0<<TWINT)|
         (0<<TWEA)|(0<<TWSTA)|(0<<TWSTO)|
               (0<<TWWC);
        }
```

Appendix B

```
unsigned char TWI_drvr_writeregister(unsigned char slave_address, unsigned char REG_ADDR, unsigned
char REG_DATA)

{

        twi_start_transceiver_with_data(   &REG_DATA,   1,     ((slave_address<<TWI_ADR_BITS)   |
(FALSE<<TWI_READ_BIT)), REG_ADDR );

        switch(twi_get_state_info())

        {

                case TWI_MTX_ADR_NACK:

                case TWI_MRX_ADR_NACK:

                case TWI_MTX_DATA_NACK:

            case TWI_BUS_ERROR:

                        return 0;

                default:

                        return 1;

        }

}
```

Read Data at Raspberry:

```
unsigned char TWI_drvr_readregister(unsigned char slave_address, unsigned char REG_ADDR, unsigned
char* receivedData)

{

        unsigned char outData[TWI_BUFFER_SIZE];

        twi_start_transceiver_with_data( 0, 0, (slave_address<<TWI_ADR_BITS) |
(FALSE<<TWI_READ_BIT), REG_ADDR );


        twi_start_transceiver_with_data( 0, 0,  ((slave_address<<TWI_ADR_BITS) |
(TRUE<<TWI_READ_BIT)),  REG_ADDR );

        twi_get_data_from_transceiver( outData, 2 );

        *receivedData = outData[1];

        switch(twi_get_state_info())

        {

                case TWI_MTX_ADR_NACK:

                case TWI_MRX_ADR_NACK:

                case TWI_MTX_DATA_NACK:

            case TWI_BUS_ERROR:

                        return 0;

                default:

                        return 1;

        }

    }
```

Initial_ADC:

```
void InitADC()

{

 DIDR0 = 1<<ADC0D;

 ADMUX |= (1<<REFS0);

 ADCSRA |= (1<<ADSC);

 ADCSRA |= (1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADEN);

}
```

Read ADC:

```
uint16_t ReadADC(uint8_t ADCchannel)

{

 ADMUX = (ADMUX & 0xF0) | (ADCchannel & 0x0F);

 _delay_us(10);

 ADCSRA |= (1<<ADSC);

 while(ADCSRA & (1<<ADSC) );

 return ADC;

}
```

Appendix C

```
unsigned char twi_busy( void )
{
  return ( TWCR & (1<<TWIE) );
}
```

Appendix D

```
static void mixColumn( byte * column )
{
        byte result0, result1, result2, result3;سيب
        byte column0, column1, column2, column3;
        byte xor;
        column0 = column[0];
        column1 = column[1];
        column2 = column[2];
        column3 = column[3];
        result0 = column1 ^ column2 ^ column3;
        result1 = column0 ^ column2 ^ column3;
        result2 = column0 ^ column1 ^ column3;
        result3 = column0 ^ column1 ^ column2;
        xor = 0;
        if (column0 & 0x80) {
                xor = BPOLY;
        }
        column0 <<= 1;
        column0  ^= xor;
        xor = 0;
        if (column1 & 0x80) {
                xor = BPOLY;
        }
        column1 <<= 1;
        column1  ^= xor;
        xor = 0;
        if (column2 & 0x80) {
                xor = BPOLY;
        }
        column2 <<= 1;
        column2  ^= xor;
        xor = 0;
        if (column3 & 0x80) {
                xor = BPOLY;
        }
        column3 <<= 1;
        column3  ^= xor;
        column[0] = result0 ^ column0 ^ column1;
        column[1] = result1 ^ column1 ^ column2;
        column[2] = result2 ^ column2 ^ column3;
        column[3] = result3 ^ column0 ^ column3;
}
```

Appendix E

```
void USART1_Init()
{
  #include<util/setbaud.h>
        unsigned cshar x;

        UBRR0H = UBRRH_VALUE;
        UBRR0L = UBRRL_VALUE;
        UCSR0B = ((1 << RXCIE0) | (1 << RXEN0) | (1 << TXEN0));
        UCSR0C = (1 << UCSZ01) | (1 << UCSZ00);
        x = 0;
        USART_RxTail = x;
        USART_RxHead = x;
        USART_TxTail = x;
        USART_TxHead = x;
}
```

Appendix F

```
int USART_Transmit(char data, FILE *stream)
{
        unsigned char tmphead;
        if(data == '\n')
                USART_Transmit('\r',stream);
        tmphead = (USART_TxHead + 1) & USART_TX_BUFFER_MASK;
        while (tmphead == USART_TxTail);
        USART_TxBuf[tmphead] = data;
        USART_TxHead = tmphead;
        UCSR0B |= (1<<UDRIE0);
        return 0;
}
```

Appendix G

```
filename = 'testdata.xlsx';
%A = {'Time','Temperature'; 12,98; 13,99; 14,97};
sheet = 6;
xlRange = 'E1';

% figure(1)
% ylim([-10 260 ]);
a=0;
ts=0;
Ds=0;
Df=0;
c=0;
rev=0;
RD=0;
%fopen(s)
%%
filename = 'Data_From_Node.xlsx';
sheet = 4;
xlRange = 'E1';
s= serial('COM2');
data={0,0,0,0};
P_S=[0,0,0,0];
while(1)
% Time=clock;

fopen(s)

tline=fgets(s);
pause(1);
T_mat=S_tline(tline,30);
Sensor_Data=extrat_num(T_mat)
if(sum(Sensor_Data(:,4))==0)
%     if(sum(Sensor_Data(:,1))/11==sum(Sensor_Data(1,1)) && ...
%        sum(Sensor_Data(:,2))/11==sum(Sensor_Data(1,2)) && ...
%        sum(Sensor_Data(:,3))/11==sum(Sensor_Data(1,3)) )
P_S=[P_S ;Sensor_Data];
```

```
plot(P_S);
legend('Body_Temperature','Blood_Pressure','SPO2',' ','Location','NorthEastOutside')
 data=cat(1,data,num2cell(Sensor_Data));
 A = {'Body_Temperature','Blood_Pressure','SPO2',' '};
 A=cat(1,A,data);
 xlswrite(filename,A,sheet,xlRange);
 end
fclose(s);
end
fprintf(s,'AT+CMGF=1');
%plot(int8(tline))
axis
```