**CHAPTER 1**

**INTRODUCTION**

**1.1 PROJECT SPECIFICATION**

The “Home Automation” concept has existed for many years. The terms “Smart Home”, “Intelligent Home” followed and has been used to introduce the concept of networking appliances and devices in the house. Home automation Systems (HASs) represents a great research opportunity in creating new fields in engineering, architecture and computing (Huidobro and Millan, 2004). HASs becoming popular nowadays and enter quickly in this emerging market. However, end users, especially the disabled and elderly due to their complexity and cost, do not always accept these systems.

**1.2 OVERVIEW**

With increasing technology, Automation has become a need whether it is home, office or some other place. At home we come across many appliances be it Fan, AC, TV, Lights, etc. What if you could operate all of them with the Android Phone you’re holding in your Hand.

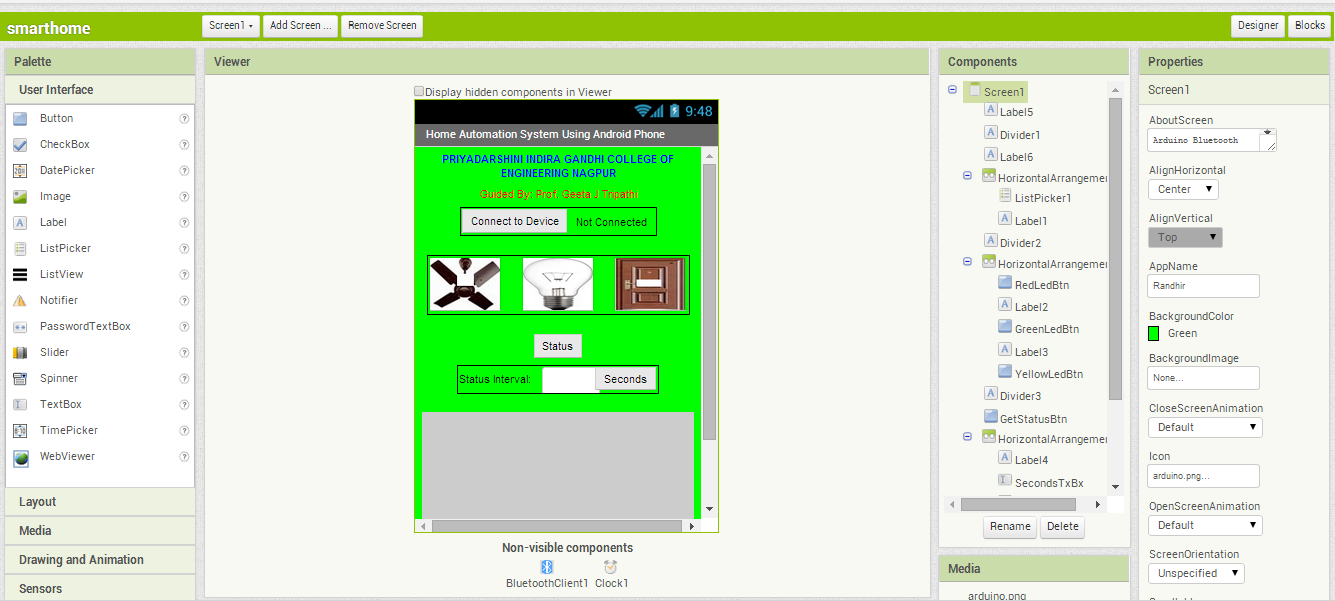
This Project aims to incorporate android phone control over electrical appliances. We use Bluetooth communication between Android phone and a Receiver (control unit) that is connected to the appliances.

Bluetooth is being chosen with its suitable capability. Bluetooth with globally available frequencies of 2400Hz is able to provide connectivity up to 100 meters at speed of up to 3Mbps depending on the Bluetooth device class [1]. In addition, a Bluetooth master device is able to connect up to 7 devices in a “Piconet”. The capabilities of Bluetooth are more than enough to be implemented in the design. Also, most of the current laptop/notebook or cell phones are come with built-in Bluetooth adapter. It will indirectly reduce the cost of this system.

**1.3 FEATURES**

The ability to manage your home’s electronic systems from one main control system can make your household run smoother, feel better and save energy.

The trick is to find a system that will meet all the demands of your household, now and in the future. Most systems can be tailored by a custom electronics professional to provide all the benefits you desire, but there are some key features that will make his job easier and your interaction with your system more enjoyable.



**1.3.1 Interoperability**

The beauty of an automation system is its ability to tie diverse electronic devices together so they can perform as one unified system. Getting these devices to work cohesively can be simple or complex, depending on the “openness” of the automation system. The more open a system is, the easier it will be for the lights, thermostats, audio/video equipment, security devices, motorized shades and other electronics to communicate with each other. A good example of interoperability is having the lights turn off, the thermostats set back when you press a “goodbye” button on a keypad or when a motion sensor notices that you have exited a room.

To support interoperability between multiple electronic devices, manufacturers of home automation systems often form connectivity partnerships with other manufacturers. For example, [Home Automation Inc.](http://www.homeauto.com/main.asp) has partnered with 40 other companies to ensure that its Omni line of automation products can communicate seamlessly with a wide variety of other systems—from architectural lighting and irrigation, to multiroom audio.

Another way automation manufacturers are fostering interoperability is through adherence to technology standards. [Control4](http://www.electronichouse.com/specials/homecontrol), for example, has embedded Zigbee wireless control technology into automation products so those products can network easily with other Zigbee-enabled products.

The more connectivity partners a manufacturer has formed and standards it has adopted, the more choices you’ll have as a consumer. More importantly, says custom electronics pro Bill Charney with [Advanced Home Audio](http://www.advancedhomeaudio.com/) in Shelton, Conn., “It allows installers to select the best suite products for their clients.”

**1.3.2. Remote Access**

“Automation is all about being able to control things in your home,” says Jay McClellan, president of [Home Automation Inc.](http://www.homeauto.com/main.asp), “and part of that is being able to change the settings quickly and easily if your plans change.” More often than not, plans change when you’re not at home. Remote access capabilities allow you to monitor your home’s environment and alter the settings of the lights, thermostats and other gear if necessary all from your laptop, cellphone or iTouch. McClellan believes that remote monitoring should be a service manufacturers and installers provide free of charge

**1.3.3. Expandability**

The way you live in your home five years from now will probably be much different than the way you live in your home today. Moreover, technology will continue to evolve, introducing a completely new generation of products to the marketplace. In the future, you may also want to add new rooms—like a recently finished basement or an addition off the back—to your automation network. Or, you may simply want to start out with just a few features when you first put in your system then add new capabilities later as you have the money. For these reasons, it’s important that a home automation system can be easily expanded both vertically to incorporate additional products and horizontally to support additional rooms.

Manufacturers can support vertical and horizontal expandability by designing their systems to speak a common network language, like IP (Internet Protocol), and by offering wireless retrofittable products that can communicate with a home’s existing network of wired products.

**1.3.4. Upgradeability**

Those touchscreens and black boxes may look impressive, but it’s what you don’t see that holds the true power of an automation system. Software is the driving force of an automation system. The more sophisticated that software is, the more the system can do. As technology changes, so must the software. Before you buy any system, be sure the manufacturer (or your installer) will be able to unlock and download software updates automatically.

**1.3.5. Variety of Interfaces**

There are a number of different ways you can control the electronic systems in your home: by pressing the buttons of a handheld remote or wall-mounted keypad, by touching colorful icons on a portable touchpanel or by sliding your finger across your iTouch. Depending on your family dynamic, budget and preferences, you might like to utilize a variety of different controllers (most people do, says McClellan), so make sure the automation manufacturer offers a wide selection of interfaces.

**1.3.6. Time-Tested**

No one, except for serious early-adopters, likes to be the guinea pig, so choose an automation system with a proven track record. Same goes for the person who installs the system into your home. “[Look for an installer](http://www1.electronichouse.com/installer/)  who’s been installing the same systems for a number of years,” suggests Derek Cowburn of  [Distinct AV](http://www.distinctav.com/" \t "_blank" \o "Distinct AV) in McCordsville, Ind. You should be able to gather some historical background about manufacturers and installers from their company websites.

**1.3.7. Strong Dealer Network**

“You can have great equipment,” says Jeff Singer of automation system manufacturer[Crestron](http://www.crestron.com/), “but you’ll need a highly trained and certified installer in order to get your money’s worth.” Good home automation manufacturers go above and beyond to create a strong dealer network, by offering continual education and training and by supporting multiple dealers in a single geographic area. For consumers, having more than one dealer to choose from is important. When more than one dealer carries a particular product in your area, pricing is more competitive and should one dealer go out of business, demand that he provide access to your project file, advises Eric Smith of home automation manufacturer Control4.

**1.3.8. Commitment to Energy-Savings**

One of the hottest topics in the consumer media is energy conservation. Automation systems can help save energy by turning off electronics devices automatically, and some do this better than others. Be sure to check out the energy-saving features of a system before you buy.

**1.3.9. Layer of Protection**

Everyone always wonders what happens to an automated house when the power goes out. If an automation system has the appropriate back-up protection, you won’t have to worry about that.

**1.3.10. Can-do Attitude**

This goes both for the installer and the manufacturer. Automation is only beneficial and practical if it fits your lifestyle. Since everyone’s lifestyle is different, the manufacturer should provide its installers with the tools to customize the system to your specific needs. If there’s something that you want your system to do and your installer says it’s impossible, either he or the manufacturer has failed you. Keep looking.

**1.4 LIMITATIONS**

When you think of home automation for your home, commonly the first response to it, is that its a rather expensive system to have installed into your home.

But have you ever thought, how it can benefit your home and  lifestyle? At the press of a button you are able to literally have ful control over your home. It adds additional security to you and your family, and helps reduce costs.As technology is growing and advancing, we too are “advancing” by means of allowing technology to merge into our lifestyles. Gone are the days where technology products were so expensive which only big corporations or the more richer people could afford.

Nowadays, alot of those products prices have reduced, allowing a average person such as you and I to afford these products. Before you do decide to purchase a home automation system, lets take a look at the limitations to home automation:

* **Costs:**Its no secret that installing a home automation system can be quite costly. But, it all depends on the equipment you wish to have installed. Remember, the more advanced system you wish to have in your home the more expensive it will be.
* **Human Error:**If the equipment is not handled & installed safely, this can lead to the equipment being damaged, and the risk of the system crashing is high.
* **Reliability:** This occurs on a very rare occasions, depending on the age of the equipment it can have a affect on the system, but otherwise the technology in the home automation systems are all up to date.
* **1.5 BENEFITS**
* Home automation makes your home safer: Its visible around us that crime is on the increase, families are beginning to aim at purchasing security alarm systems for their homes. Home Automation adds a extra sense of security to your home, using the automation system you can monitor your homes security cameras using a remote device. You can recieve notifications through email or message, when your alarm has been armed/disarmed or tripped.
* Home automation saves you money and energy:  Home automation systems are designed not to use alot of electricity, a added bonus to having a home automation system is that you can schedule your lighting in your home to go on and off at selected times of the day. Using a remote device, you’ll have remote access to operate any lighting or electrical appliances.
* Home Automation is great for the parents: If you have a newborn or toddler in your home, having a home automation system is perfect for any parent. You can be anywhere in your home, and still be able to monitor your child in a different part of the house.
* A home automation system is a all in one user friendly system: For any technological fanatic this is the ultimate device. From the press of a button on a remote device, the user is able to control Lighting, Air Conditioning, Audio & Video systems and Security Video cameras.

CHAPTER 2

**LITARATURE SURVEY**

When people think about home automation, most of them may imagine living in asmart home: One remote controller for every household appliance, cooking the riceautomatically, starting air conditioner automatically, heating water for bath automaticallyand shading the window automatically when night coming. To some extent homeautomation equals to smart home. They both bring out smart living condition and make ourlife more convenient and fast.People at that time understood that a smart home is not owing to how well it isbuilt, not how effectively it uses space, not due to how it is environmentally friendly. It isonly because of how interactive technologies that it contains. Those are still useful rules forhome automation technology today.Home automation technology and Smart home appeared very much in sciencefiction of the 1920s. But no one knows the exact date of the invention of home automation.

Based on human’s smart technology improving process, the home automation

system doesnot come by immediate invention. It comes step by step with only insignificantimprovement. The previous step is almost same with the next step.Manufactures of laborsaving appliances have been promising homeowners an

automated “Home of the Future” since the World’s Fair days of the 1930s. The intelligent

home has been a popular vision for a few decades.The first time people noticed the high technology in dwelling, they did someconnection with home automation, and it was 1960s. Experts considered that by the end ofthe century people would live in smart homes that contained independent householdmachined. Although many of the machines in these visions are today technically possible tomanufacture of course, the present situation is not exactly the same as what the expert’s decades ago.

Integration of Android and Bluetooth technology in Controlling home appliances can help and improve lifestyle of all user groups especially to the disabled and elderly people in term of safety and comfortable. The implementation of combined wired and wireless systems would be of most practical in designing a smart home system especially in cutting the system’s installation cost for conventional home.

CHAPTER 3

**COMPONENTS**

The components of a home automation system can be broken down into several categories. Click on the menu on the left side of this page to see them.

**Controlled Devices**

Controlled devices include the tremendous range of equipment that a home automation system is capable of controlling. They include household appliances, door openers, power door locks, sprinkling systems, lighting systems, HVAC systems, audio/video systems, home theater equipment, power drapes, security systems, telephone systems, intercoms, messaging systems, information systems, and many other types of equipment. There are far too many types of devices to list here.

**Controllers**

Controllers provide the intelligent control functions in a home automation system. They can range from a simple lamp timer unit to a smart keypad to a powerful computer. Controllers include any intelligent device capable of sending commands that are understood by the controlled devices.

The control functions may be contained in a single central controller, or there may be other controllers besides the central controller that have a limited subset of control functions.

All controllers must have sufficient data in order to control the controlled devices. Data can come from user input, sensor input, a timer, a control program, or some combination of these. To obtain user input, the system must have one or more user interfaces.

**I/0 Interface Devices**

I/O (input/output) interface devices provide the logical communication link between the controller(s) and the controlled devices in a system. They are the means of making various devices compatible with the physical and logical structure of the system. I/O interface devices may be separate equipment items, or they may be built in to a controlled device, in which case it would just be considered a feature of the controlled device rather than a separate equipment item.

Many equipment items that are typically part of home automation systems come with some kind of built-in industry-standard control interface. For example, VCRs, TVs, home theater, and audio equipment may have either infrared or serial interfaces, or both. Most use standardized protocols that allow remote controls from many manufacturers to interact with them.

Some controlled devices require optional or third party I/O interface devices that allow those devices to be integrated into the home automation system. Lighting equipment can be controlled with a variety of proprietary or third party dimmers or switches that have built-in I/O capabilities. Other equipment items that are less commonly used in home automation systems may require the use of "generic" I/O interface devices. As an example, to control a coffee maker, you would need some kind of generic on/off switch device with its own I/O interface. To control a reversible irrigation pump, you might need a two-relay device with its own I/O interface.

Most I/O interface devices provide one-way communications from the controller(s) to the controlled devices, although there are some types, such as RS-232 interfaces, which allow two-way communications. An example would be a smart thermostat with a built-in RS-232 interface.

An I/O interface device can serve several communications functions, including:

Converting analog signals to digital signals that can be used by the controller. This would, for example, convert the analog voltage from a thermocouple (temperature sensor) to a digital signal that could be used by the controller. Converting signals from the controller to a physical and logical form that can be received and understood by the controlled devices. Converting commands from the controller into a different set of commands that a controlled device understands. System interface devices have varying amount of built-in intelligence. For example, you might have a specialized I/O interface device that receives standardized commands from a central controller, translates those commands into a new set of commands that is understood by the device it is designed to control, and transmits those commands in the correct form and syntax to the device. This kind of "smart" device may sound like a combination of a controlled device and a controller, but it is more appropriate to call it an I/O interface device. Still another type of system interface device would be a dimmer. The dimmer would interpret the commands from the system and raise or lower lighting levels accordingly.

**User Interfaces**

User interfaces allow the user to interact with the system by sending information to the controller or by presenting information to the user about the system. The form and capabilities vary widely. Depending on the system and the type of user interface, you could do any or all of the following:

Issue direct commands to the system (such as "turn on the kitchen lights"). Obtain feedback from the system (such as the temperature of the bedroom, via a thermostat). Program the controller to carry out certain functions automatically, based on time, sequence, or conditions. Some controllers may have an integral user interface (like the keyboard on a computer), or there may be remotely located user interfaces with varying degrees of built-in intelligence (such as wall-mounted or hand-held keypads or touch panels). A dumb keypad is only used as a remote user input for a controller. A smart keypad might function as a remote user interface for a controller, but also could have enough on-board intelligence to issue its own commands to certain controlled devices; hence it could function as both user interface and controller.

Similarly, a user interface device does not have to be entirely devoted to user input and/or feedback. For example, a keypad might incorporate a temperature sensor or a light dimmer and a microphone for use with an intercom system within the same physical enclosure.

A user interface may accept a variety of user input types. Keystrokes or button presses are the most common modes of entering data, but some systems may accept voice input or other forms of communication.

Not all controllers have a "user" interface. Some specialized controllers may simply use input from sensors (or other equipment), and programming to make intelligent decisions based on that input.

Typical user interface devices include:

* Push-button panels, with or without visual displays.
* Touch-panel displays, with fixed or programmable screen layouts.
* Mobile keyboards and monitors.
* Hand-held remote controls.
* Bluetooth interfaces to allow long-distance remote control.

Most high-level home automation systems are hybrid systems, using a variety of user interface devices, each suited to certain tasks. For example, some systems might allow you to carry out all control, programming, and feedback functions from a personal computer, but might limit the number of functions available from a hand-held remote control unit or a wall-mounted push-button LCD panel.

Don't confuse user interfaces with sensors. Even though there can be some overlap in function (both can accept "input" from a user), for the purposes of this discussion, a user interface is designed to provide a means of conscious, intentional interaction with the system. For example, a home automation system might use an infrared motion sensor to determine when you enter a room and automatically send the equivalent of a button press to turn on the room lights or to trigger a whole sequence of events, but it is something that happens automatically, without any conscious choice on the part of the user.

**System Network**

The system network includes all of the controllers, wires, cables, Bluetooth, adapters, connectors, junction boxes, dimmers, power supplies, etc. that connect the various system components. This might sound like the simplest part of the system, but it can actually get quite complex. In many systems, this is the area that requires the most planning and can be the most labor-intensive part of a total system installation.

**Programming Computer**

Some system controllers allow the user to program the system with the system's own user interface(s). Other systems require the use of a separate computer (typically a PC) to program the system controller. Still others may allow certain functions to be programmed with the system's own user interface(s), but require a separate computer to program the more advanced functions or change certain basic operating parameters.

CHAPTER 4

**AIM & OBJECTIVES**

**AIM**

With increasing technology, Automation has become a need whether it is home, office or some other place. At home we come across many appliances be it Fan, AC, TV, Lights, etc. What if you could operate all of them with the Android Phone you’re holding in your Hand

This Project aims to incorporate android phone control over electrical appliances. We use Bluetooth communication between Android phone and a Receiver (control unit) that is connected to the appliances.

**OBJECTIVE**

The objective of this project is to impliment a low cost, reliable and scalable home automation system that can be used to remotely switch on or off any household appliance, using a microcontroller to achieve hardware simplicity, low cost devices using android phone

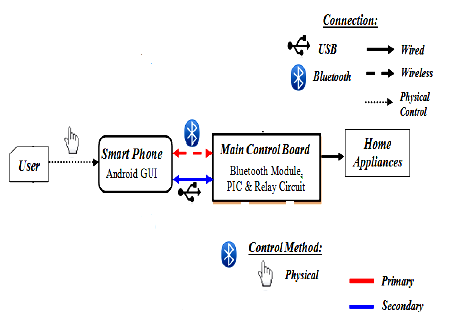
The terms “Smart Home”, “Intelligent Home” followed and has been used to introduce the concept of networking appliances and devices in the house. Home automation Systems (HASs) represents a great research opportunity in creating new fields in engineering, architecture and computing (Huidobro and Millan, 2004). HASs becoming popular nowadays and enter quickly in this emerging market. However, end users, especially the disabled and elderly due to their complexity and cost, do not always accept these systems

CHAPTER 5

**SYSTEM ARCHITECTURE**

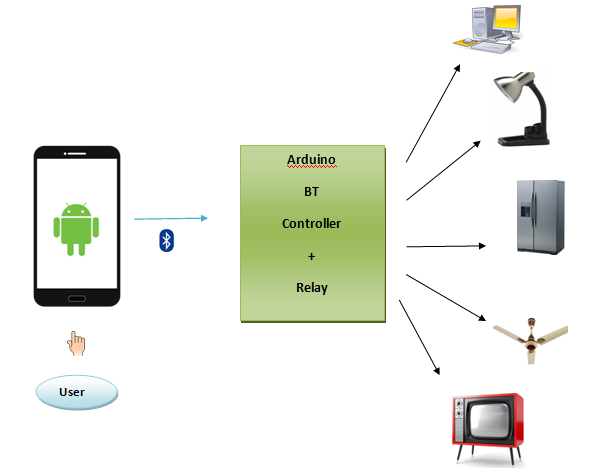
**SYSTEM ARCHITECTURE**

The system is directly installed beside the conventional electrical switches on the wall. The Bluetooth wireless connection enabled the system communicates with graphical user interface (GUI) on PC/laptop or smart phone without cable. The target home appliances are controlled by the system Main Control Board. In order improve the standard living in home, this system provides three different types of physical control methods to the Main Control Board. control method is done by Android GUI installed in Smart Phone. The user can easily touch on the screen of the phone to control the home appliances. This portable method is able to assist the disabled people who have problem with locomotion difficulty.



Smart phone can directly connect to main board. While in case of Bluetooth connection issue occurred between personal computer or laptop and control board, personal computer or laptop can connect to board by wired USB connection. The secondary connection is act as the backup solution in the system.

System Block Diagram



Home Automation System Block Diagram

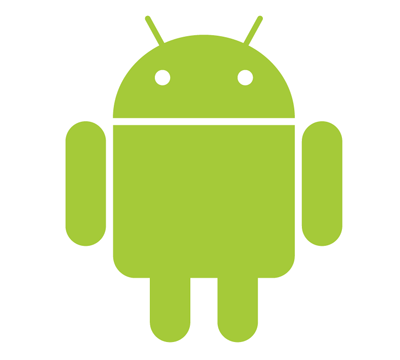
**CHAPTER 6**

**TOOLS AND PLATFORMS USED**

**6.1 ANDROID**

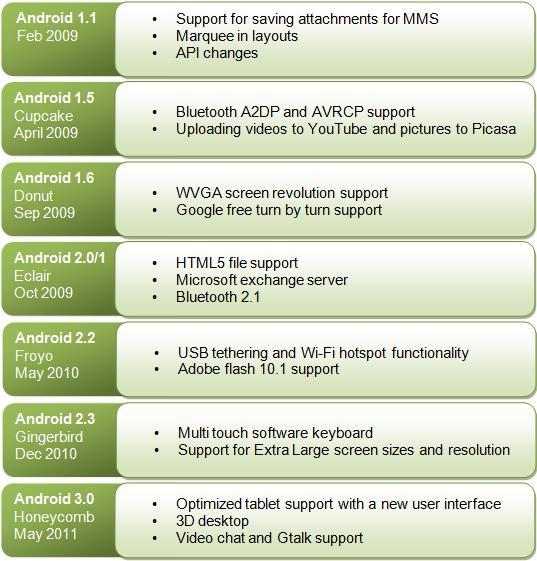
For this home automation and security system we are targeting Android platform since it has huge market and open source. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android OS is based on Linux. Android Applications are made in a Java-like language running on a virtual machine called ‘Dalvik’ created by Google. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.

World is contracting with the growth of mobile phone technology. As the number of users is increasing day by day, facilities are also increasing. Starting with simple regular handsets which were used just for making phone calls, mobiles have changed our lives and have become part of it. Now they are not used just for making calls but they have innumerable uses and can be used as a Camera , Music player, Tablet PC, T.V. , Web browser etc . And with the new technologies, new software and operating systems are required.



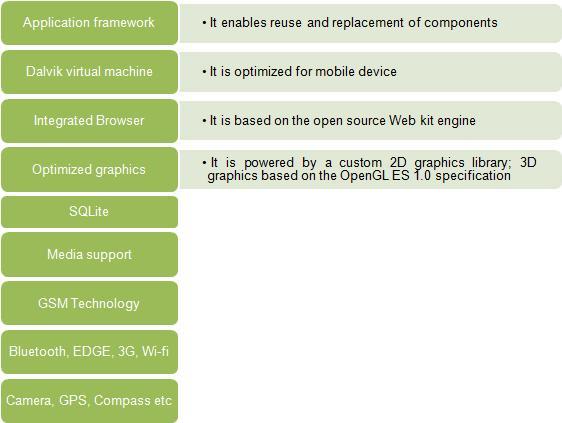
[Operating Systems](http://www.engineersgarage.com/articles/operating-systems-tutorial) have developed a lot in last 15 years. Starting from black and white phones to recent smart phones or mini computers, mobile OS has come far away. Especially for smart phones, Mobile OS has greatly evolved from Palm OS in 1996 to Windows pocket PC in 2000 then to Blackberry OS and Android.

One of the most widely used mobile OS these days is **ANDROID**. **Android** is a software bunch comprising not only operating system but also middleware and key applications. Android Inc was founded in Palo Alto of California, U.S. by Andy Rubin, Rich miner, Nick sears and Chris White in 2003. Later Android Inc. was acquired by Google in 2005. After original release there have been number of updates in the original version of Android.



**Features & Specifications**

**Android** is a powerful Operating System supporting a large number of applications in [Smart Phones](http://www.engineersgarage.com/articles/smart-phones). These applications make life more comfortable and advanced for the users. Hardwares that support Android are mainly based on [ARM architecture](http://www.engineersgarage.com/articles/arm-advanced-risc-machines-processors) platform. Some of the current features and specifications of android are:



Android comes with an Android market which is an online software store. It was developed by Google. It allows Android users to select, and download applications developed by third party developers and use them. There are around 2.0 lack+ games, application and widgets available on the market for users.

Android applications are written in java programming language. Android is available as open source for developers to develop applications which can be further used for selling in android market. There are around 200000 applications developed for android with over 3 billion+ downloads. Android relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. For software development, Android provides **Android SDK** (Software development kit). Read more about [open source software](http://www.engineersgarage.com/articles/open-source-software-history-advantages).

**Applications**

These are the basics of Android applications:

•      Android applications are composed of one or more application components (activities, services, content providers, and broadcast receivers)

•      Each component performs a different role in the overall application behavior, and each one can be activated individually (even by other applications)

•      The manifest file must declare all components in the application and should also declare all application requirements, such as the minimum version of Android required and any hardware configurations required

•      Non-code application resources (images, strings, layout files, etc.) should include alternatives for different device configurations (such as different strings for different languages)

Google, for software development and application development, had launched two competitions ADC1 and ADC2 for the most innovative applications for Android. It offered prizes of USD 10 million combined in ADC1 and 2. ADC1 was launched in January 2008 and ADC 2 was launched in May 2009. These competitions helped Google a lot in making Android better, more user friendly, advanced and interactive.

**6.2 MIT APPS INVENTOR 2**

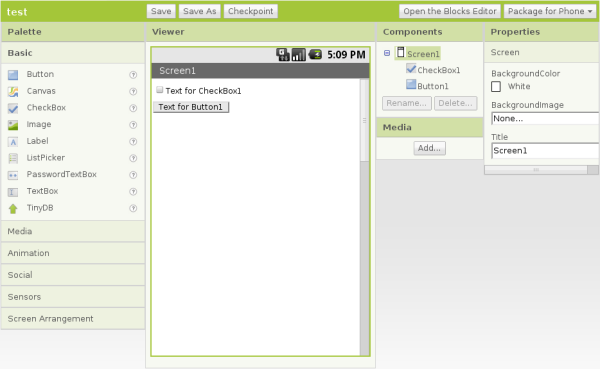
We are developing Android application. The application consists of main function like Light controlling, Fan controlling, Door locking and unlocking. The main screen has a list of all functions among which user can select any one function which he want to control. After selecting a function he would be able to see a current status of a particular device. If user wishes, he can enable or disable intended device.

We are developing android software using enviroment called MIT Apps Inventor. It allows newcomers to [computer programming](http://en.wikipedia.org/wiki/Computer_programming) to create [software applications](http://en.wikipedia.org/wiki/Application_software) for the [Android](http://en.wikipedia.org/wiki/Android_(operating_system)) operating system (OS). It uses a graphical interface, very similar to [Scratch](http://en.wikipedia.org/wiki/Scratch_(programming_language)) and the [StarLogo TNG](http://en.wikipedia.org/wiki/StarLogo_TNG" \o "StarLogo TNG) [user interface](http://en.wikipedia.org/wiki/User_interface), which allows users to [drag-and-drop](http://en.wikipedia.org/wiki/Drag-and-drop) visual objects to create an application that can run on Android devices. In creating App Inventor, Google drew upon significant prior research in educational computing, as well as work done within Google on online development environments.[[1]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-MIT_roots-1)

App Inventor and the projects on which it is based are informed by [constructionist learning](http://en.wikipedia.org/wiki/Constructionist_learning) theories, which emphasizes that programming can be a vehicle for engaging powerful ideas through active learning. As such, it is part of an ongoing movement in computers and education that began with the work of [Seymour Papert](http://en.wikipedia.org/wiki/Seymour_Papert) and the MIT Logo Group in the 1960s and has also manifested itself with [Mitchel Resnick](http://en.wikipedia.org/wiki/Mitchel_Resnick)'s work on [Lego Mindstorms](http://en.wikipedia.org/wiki/Lego_Mindstorms) and [StarLogo](http://en.wikipedia.org/wiki/StarLogo" \o "StarLogo).

App Inventor includes:

* A *designer*, in which a program's *components* are specified. This includes visible components, such as buttons and images, which are placed on a simulated screen, and non-visible components, such as sensors and web connections.
* A *blocks editor*, in which the program's logic is created.
* A compiler based on the [Kawa language framework](http://en.wikipedia.org/wiki/Kawa_(Scheme_implementation)" \o "Kawa (Scheme implementation)) and Kawa's dialect of the [Scheme](http://en.wikipedia.org/wiki/Scheme_(programming_language)) programming language, developed by Per Bothner[[3]](http://en.wikipedia.org/wiki/App_Inventor_for_Android" \l "cite_note-kawa-3) and distributed as part of the [GNU](http://en.wikipedia.org/wiki/GNU) operating system by the Free Software Foundation.[[4]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-gnu-4)
* An app for real-time debugging on a connected Android device

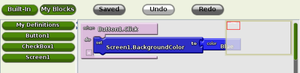


**History**

The application was made available through request on July 12, 2010, and released publicly on December 15, 2010. The App Inventor team was led by [Hal Abelson](http://en.wikipedia.org/wiki/Hal_Abelson)[[1]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-MIT_roots-1) and Mark Friedman.[[5]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-Book-5) In the second half of 2011, Google released the source code, terminated its server, and provided funding for the creation of The MIT Center for Mobile Learning, led by App Inventor creator Hal Abelson and fellow MIT professors Eric Klopfer and Mitchel Resnick.[[6]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-MIT_App_Inv-6)[[7]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-cml-7) The MIT version was launched in March 2012.[[8]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-ai2-8)

On December 6, 2013 (the start of the [Hour of Code](http://en.wikipedia.org/wiki/Hour_of_Code)),[[8]](http://en.wikipedia.org/wiki/App_Inventor_for_Android" \l "cite_note-ai2-8) MIT released App Inventor 2, renaming the original version "App Inventor Classic"[[9]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-classic-9) Major differences are:

* The blocks editor in the original version ran in a separate Java process, using the Open Blocks Java library for creating visual blocks programming languages and programming

[](http://en.wikipedia.org/wiki/File:App_Inventor_Block_Editor.png)

App Inventor Classic Blocks Editor

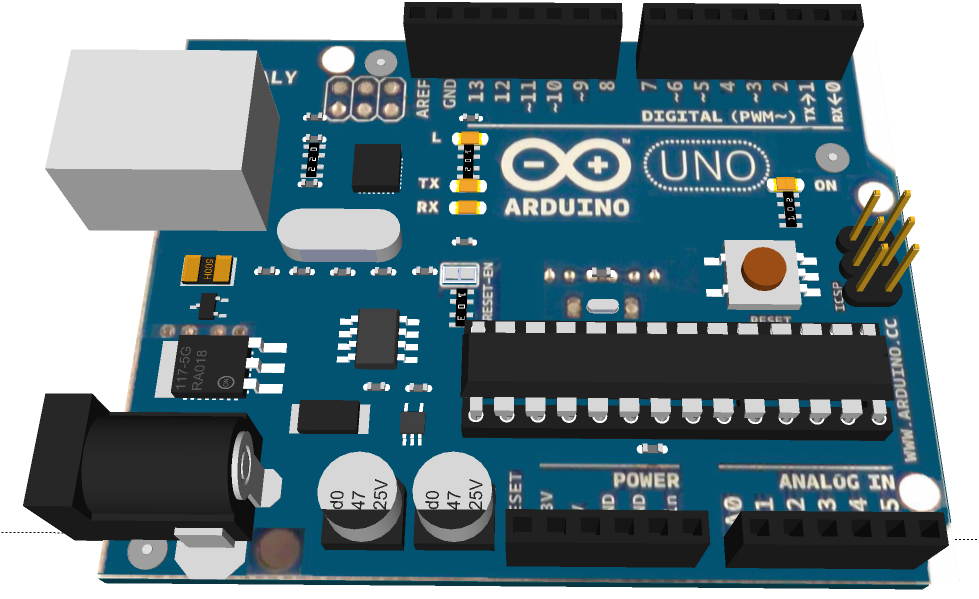
Open Blocks is distributed by the [Massachusetts Institute of Technology](http://en.wikipedia.org/wiki/Massachusetts_Institute_of_Technology)'s Scheller Teacher Education Program (STEP) and is derived from master's thesis research by Ricarose Roque. Professor Eric Klopfer and Daniel Wendel of the Scheller Program supported the distribution of Open Blocks under an [MIT License](http://en.wikipedia.org/wiki/MIT_License).[[2]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-Giants-2) Open Blocks visual programming is closely related to[StarLogo TNG](http://en.wikipedia.org/wiki/StarLogo_TNG), a project of STEP, and [Scratch](http://en.wikipedia.org/wiki/Scratch_(programming_language)), a project of [MIT Media Laboratory's](http://en.wikipedia.org/w/index.php?title=MIT_Media_Laboratory%27s&action=edit&redlink=1) [Lifelong Kindergarten Group](http://en.wikipedia.org/wiki/Mitchel_Resnick). App Inventor 2[[9]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-classic-9) replaced Open Blocks with [Blockly](http://en.wikipedia.org/wiki/Blockly" \o "Blockly), a blocks editor that runs within the browser.

* The [| MIT AI2 Companion app](https://play.google.com/store/apps/details?id=edu.mit.appinventor.aicompanion3) enables real-time debugging on connected devices via Wi-Fi, not just USB.

As of May 2014, there were 87 thousand weekly active users of the service and 1.9 million registered users in 195 countries for a total of 4.7 million apps built.[[8]](http://en.wikipedia.org/wiki/App_Inventor_for_Android#cite_note-ai2-8)

**6.3 ARDUINO UNO R3**

|  |  |
| --- | --- |
| Microcontroller | ATmega328 |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limits) | 6-20V |
| Digital I/O Pins | 14 (of which 6 provide PWM output) |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 40 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (ATmega328) of which 0.5 KB used by bootloader |
| SRAM | 2 KB (ATmega328) |
| EEPROM | 1 KB (ATmega328) |
| Clock Speed | 16 MHz |



**Power**

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts. The power pins are as follows:

* **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
* **5V.**This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
* **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
* **GND.** Ground pins.
* **IOREF.** This pin on the Arduino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs for working with the 5V or 3.3V.

**Memory**

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the [EEPROM library](http://www.arduino.cc/en/Reference/EEPROM)).

**Input and Output**

Each of the 14 digital pins on the Uno can be used as an input or output, using [pinMode()](http://arduino.cc/en/Reference/PinMode), [digitalWrite()](http://arduino.cc/en/Reference/DigitalWrite), and [digitalRead()](http://arduino.cc/en/Reference/DigitalRead) functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

**Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.

**External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the [attachInterrupt()](http://arduino.cc/en/Reference/AttachInterrupt) function for details.

**PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the [analogWrite()](http://arduino.cc/en/Reference/AnalogWrite) function.

**SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the [SPI library](http://arduino.cc/en/Reference/SPI).

**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 Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the [analogReference](http://arduino.cc/en/Reference/AnalogReference)() function. Additionally, some pins have specialized functionality:

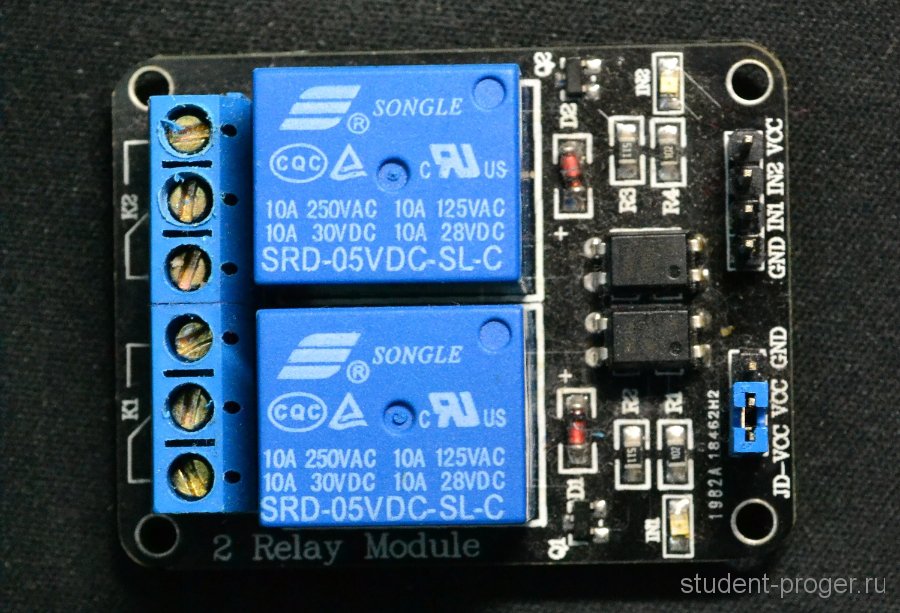
**TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the [Wire library](http://arduino.cc/en/Reference/Wire).

**6.4 RELAY**

A **relay** is an [electrically](http://en.wikipedia.org/wiki/Electric) operated [switch](http://en.wikipedia.org/wiki/Switch). Many relays use an [electromagnet](http://en.wikipedia.org/wiki/Electromagnet) to mechanically operate a switch, but other operating principles are also used, such as [solid-state relays](http://en.wikipedia.org/wiki/Solid-state_relay). Relays are used where it is necessary to control a circuit by a low-power signal (with complete electrical isolation between control and controlled circuits), or where several circuits must be controlled by one signal. The first relays were used in long distance [telegraph](http://en.wikipedia.org/wiki/Electrical_telegraph) circuits as amplifiers: they repeated the signal coming in from one circuit and re-transmitted it on another circuit. Relays were used extensively in telephone exchanges and early computers to perform logical operations.

A type of relay that can handle the high power required to directly control an electric motor or other loads is called a [contactor](http://en.wikipedia.org/wiki/Contactor). [Solid-state relays](http://en.wikipedia.org/wiki/Solid-state_relay) control power circuits with no [moving parts](http://en.wikipedia.org/wiki/Moving_parts), instead using a semiconductor device to perform switching. Relays with calibrated operating characteristics and sometimes multiple operating coils are used to protect electrical circuits from overload or faults; in modern electric power systems these functions are performed by digital instruments still called "[protective relays](http://en.wikipedia.org/wiki/Protective_relay)".

A simple electromagnetic relay consists of a coil of wire wrapped around a [soft iron core](http://en.wikipedia.org/wiki/Magnetic_core), an iron yoke which provides a low [reluctance](http://en.wikipedia.org/wiki/Magnetic_reluctance" \o "Magnetic reluctance)path for magnetic flux, a movable iron [armature](http://en.wikipedia.org/wiki/Armature_(electrical_engineering)), and one or more sets of contacts (there are two in the relay pictured). The armature is hinged to the yoke and mechanically linked to one or more sets of moving contacts. It is held in place by a [spring](http://en.wikipedia.org/wiki/Spring_(device)) so that when the relay is de-energized there is an air gap in the magnetic circuit. In this condition, one of the two sets of contacts in the relay pictured is closed, and the other set is open. Other relays may have more or fewer sets of contacts depending on their function. The relay in the picture also has a wire connecting the armature to the yoke. This ensures continuity of the circuit between the moving contacts on the armature, and the circuit track on the [printed circuit board](http://en.wikipedia.org/wiki/Printed_circuit_board) (PCB) via the [yoke](http://en.wikipedia.org/wiki/Yoke), which is soldered to the PCB.



When an [electric current](http://en.wikipedia.org/wiki/Electric_current) is passed through the coil it generates a [magnetic field](http://en.wikipedia.org/wiki/Magnetic_field) that activates the armature, and the consequent movement of the movable contact(s) either makes or breaks (depending upon construction) a connection with a fixed contact. If the set of contacts was closed when the relay was de-energized, then the movement opens the contacts and breaks the connection, and vice versa if the contacts were open. When the current to the coil is switched off, the armature is returned by a force, approximately half as strong as the magnetic force, to its relaxed position. Usually this force is provided by a spring, but gravity is also used commonly in industrial motor starters. Most relays are manufactured to operate quickly.

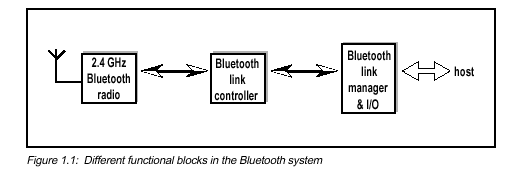
**6.5 BLUETOOTH**

**6.5.1 ABOUT**

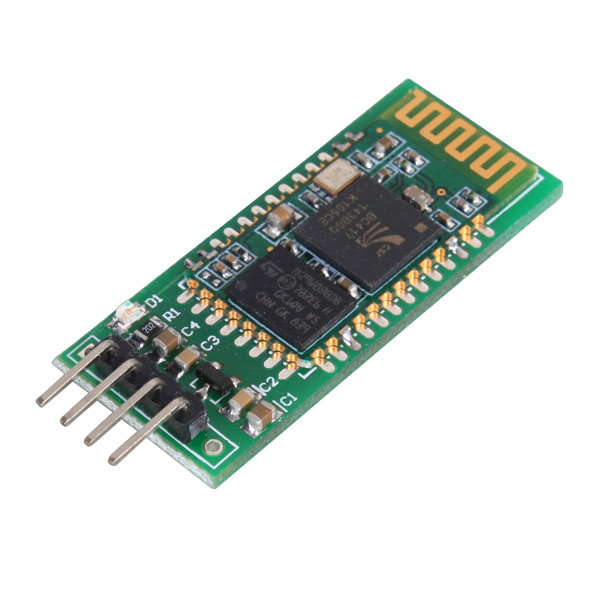
Bluetooth is a wireless technology standard for exchanging data over short distances (using short-wavelength UHF radio waves in the ISM band from 2.4 to 2.485 GHz) from fixed and mobile devices, and building personal area networks (PANs). Invented by telecom vendor Ericsson in 1994, it was originally conceived as a wireless alternative to RS-232 data cables. It can connect several devices, overcoming problems of synchronization.

**6.5.2 HOW IT WORKS?**

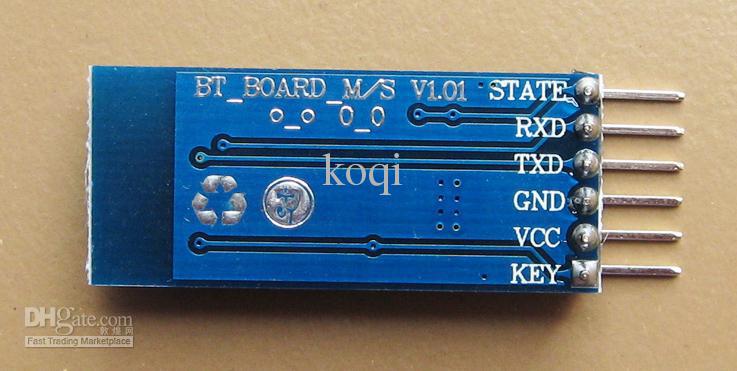
This technology achieves its goal by embedding tiny, inexpensive, short-range transceivers into the electronic devices that are available today. The radio operates on the globally-available unlicensed radio band, 2.45 GHz (meaning there will be no hindrance for international travelers using Bluetooth-enabled equipment.), and supports data speeds of up to 721 Kbps, as well as three voice channels. The bluetooth modules can be either built into electronic devices or used as an adaptor. For instance in a PC they can be built in as a PC card or externally attached via the USB port.



Each device has a unique 48-bit address from the IEEE 802 standard. Connections can be point-to-point or multipoint. The maximum range is 10 meters but can be extended to 100 meters by increasing the power. Bluetooth devices are protected from radio interference by changing their frequencies arbitrarily upto a maximum of 1600 times a second, a technique known as frequency hopping. They also use three different but complimentary error correction schemes. Built-in encryption and verification is provided.



HC-05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. HC-05 is 6-pin Module. The module has 6 pins labelled on the back, but most modules only have 4 of those populated with pogo pins. KEY & STATE seem to be not required, as KEY is used for flashing the device and STATE simply indicates if the device is awake or not. So that leaves only GND, VCC, TXD, RXD.



**6.5.3 SPECIFICATION**

* Default serail port setting :  9600, N, 8, 1Pairing code : 1234
* Bluetooth protocal :  Bluetooth Specification v2.0+EDR
* Frequency :  2.4GHz ISM band
* Modulation :  GFSK(Gaussian Frequency Shift Keying)
* Emission power :  <=4dBm, Class 2
* Sensitivity :  <=-84dBm at 0.1% BER
* Speed : Asynchronous:  2.1Mbps(Max) / 160 kbps, Synchronous: 1Mbps/1Mbps
* Security :  Authentication and encryption
* Profiles :  Bluetooth serial port
* CSR chip : Bluetooth v2.0
* Wave band : 2.4GHz-2.8GHz, ISM Band
* Protocol : Bluetooth V2.0
* Power Class : (+6dbm)
* Reception sensitivity: -85dBm
* Voltage : 3.3 (2.7V-4.2V)
* Current : Paring - 35mA, Connected - 8Ma
* Temperature : -40~ +105 Degrees Celsius
* User defined Baud rate :  4800, 9600, 19200, 38400, 57600, 115200, 230400,460800,921600 ,1382400.
* Dimension : 26.9mm\*13mm\*2.2mm

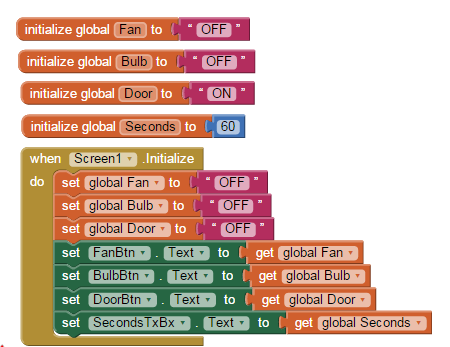
CHAPTER 7

**IMPLIMENTATION**

**7.1 ANDROID APPLICATION**

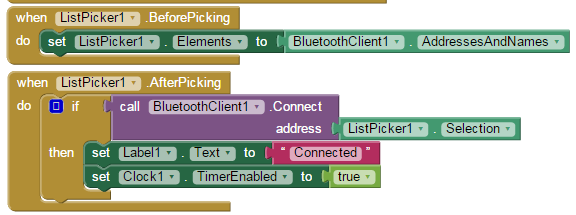
What follows is an app code description based upon the pictures below.

**BLOCK 1**



Variables for Appliance's status and interval are set and initialized when the app screen is opened.

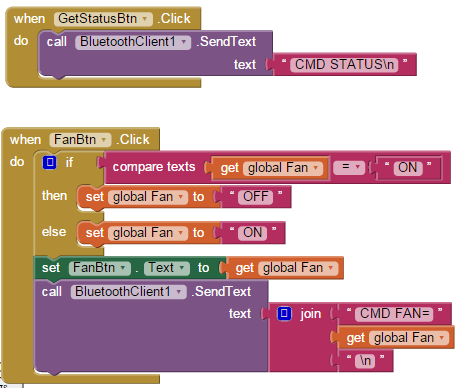
**BLOCK 2**



The connection ListPicker object ListPicker1 works with two methods:

1. a list of available (paired) Bluetooth devices is prepared and showed to the user
2. when the user selects a device, the Connect method of the Bluetooth Client object is called in order to start the connection: if it succeeds, this is displayed in the appropriate label and the Clock interrupt is activated so that messages from the device can be received.

**BLOCK 3**

****

Here we show how to send commands to the Arduino board.

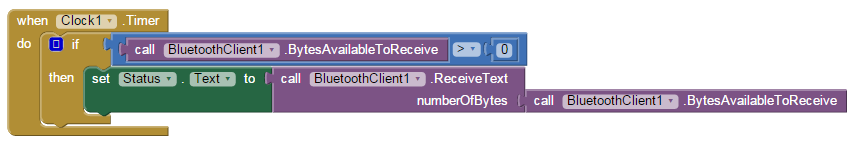
When the GetStatusBtn is pressed, the SendText method of the Bluetooth Client object is invoked and the text command is issued: note the "\n" added at the end of the "CMD STATUS" string so that the serialEvent() function in the Arduino sketch is able to know when the message is over.

The code to toggle a Appliance on or off is slightly more complex:

* we use the corresponding variable to keep track of its current status:if it's on, we want to turn it off and the other way round; so, first, we toggle the variable as it were boolean
* we then update the button label with the new state
* finally, BluetoothClient1.SendText is called to transmit the command.

The code for the other commands is not shown since it is very similar.

**BLOCK 4**



Every time the Clock1 timer fires, this routine is executed: it is the equivalent of Arduino's serialEvent(); if there are bytes to be received by BluetoothClient1, they are copied in the Status label. Note that the on a particular device, there must be support for the accessory-mode, a special means of connecting over the USB port. This allows data transfer between devices and external peripherals

**7.2 ARDUINO SKETCH**

A sketch is loaded on the Arduino board which initializes all the pins attached to servo motors to low. The sketch tells the Arduino board to initialize the required servo motor pin to high corresponding to the serial data received from matlab . The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, [on Windows, a .inf file is required](http://arduino.cc/en/Guide/Windows#toc4). 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 USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A [Software Serial library](http://www.arduino.cc/en/Reference/SoftwareSerial) allows for serial communication on any of the Uno's digital pins. The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus. For SPI communication, use the [SPI library](http://arduino.cc/en/Reference/SPI).

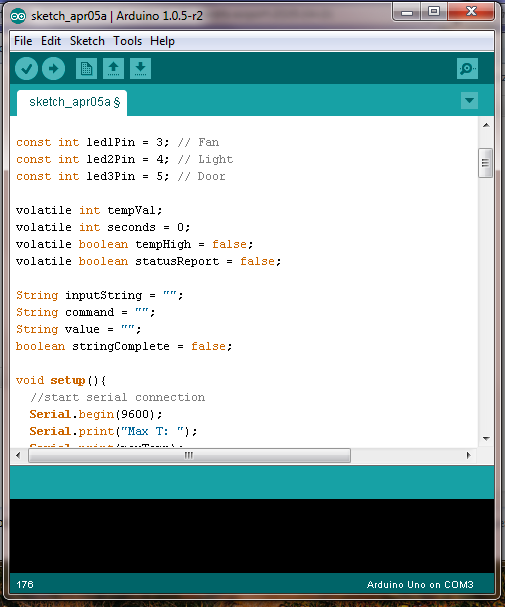
Programming: The Arduino Uno can be programmed with the Arduino software ([download](http://arduino.cc/en/Main/Software)). Select "Arduino Uno from the Tools > Board menu (according to the microcontroller on your board). For details, see the [reference](http://arduino.cc/en/Reference/HomePage) and [tutorials](http://arduino.cc/en/Tutorial/HomePage). The ATmega328 on the Arduino Uno comes preburned with a [bootloader](http://arduino.cc/en/Tutorial/Bootloader) that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol ([reference](http://www.atmel.com/dyn/resources/prod_documents/doc2525.pdf), [C header files](http://www.atmel.com/dyn/resources/prod_documents/avr061.zip)).

You can also bypass the boot loader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header using [Arduino ISP](http://arduino.cc/en/Main/ArduinoISP) or similar; see [these instructions](http://arduino.cc/en/Hacking/Programmer) for details.

The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available. The ATmega16U2/8U2 is loaded with a DFU boot loader, which can be activated by:

* On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
* On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use [Atmel's FLIP software](http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3886) (Windows) or the [DFU programmer](http://dfu-programmer.sourceforge.net/) (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU boot loader).



After developing program compile it and after successfully compilation of program upload it on the microcontroller ATMega 328 installed on Arduino UNO R3 via USB cable

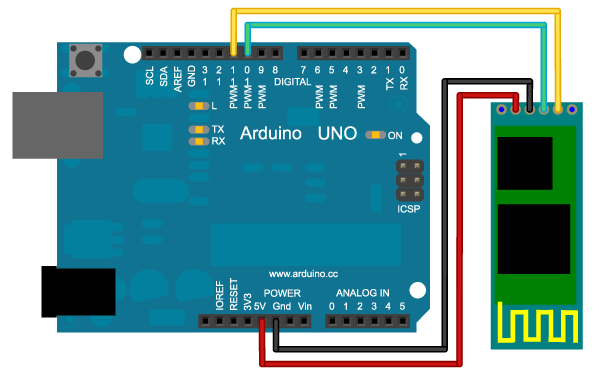
**7.3 CONNECTION OF COMPONENTS**

**7.3.1 Bluetooth Connection**

Connect Ground and 5V from Arduino to Ground and 5V of Bluetooth.

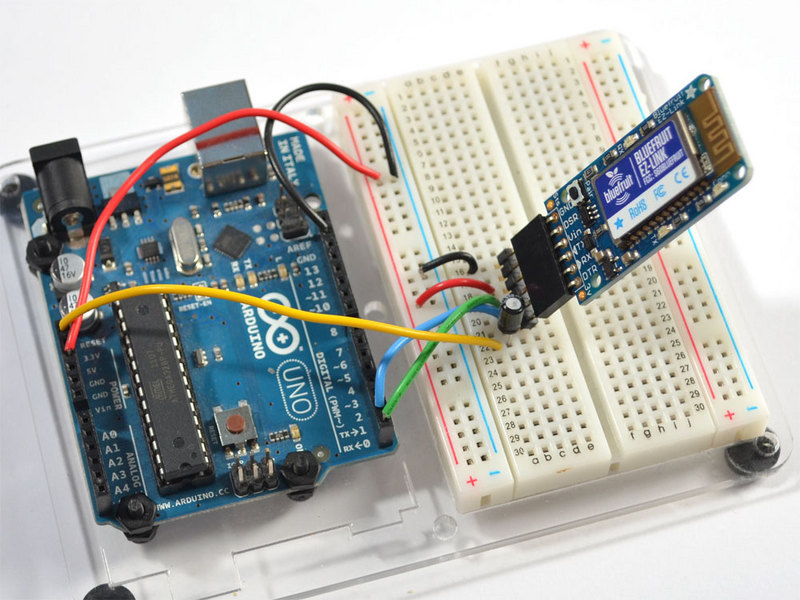
* Connect the provided cable to the JY-MCU Bluetooth module on one side and to the breadboard to the other side; connections are as follows:

VCC <--> 5V  
 GND <--> GND  
 TXD <--> Pin 0 (Rx)  
 RXD <--> Pin 1 (Tx)



* The sketch will also make use of Arduino built-in LED on Digital Pin 13.

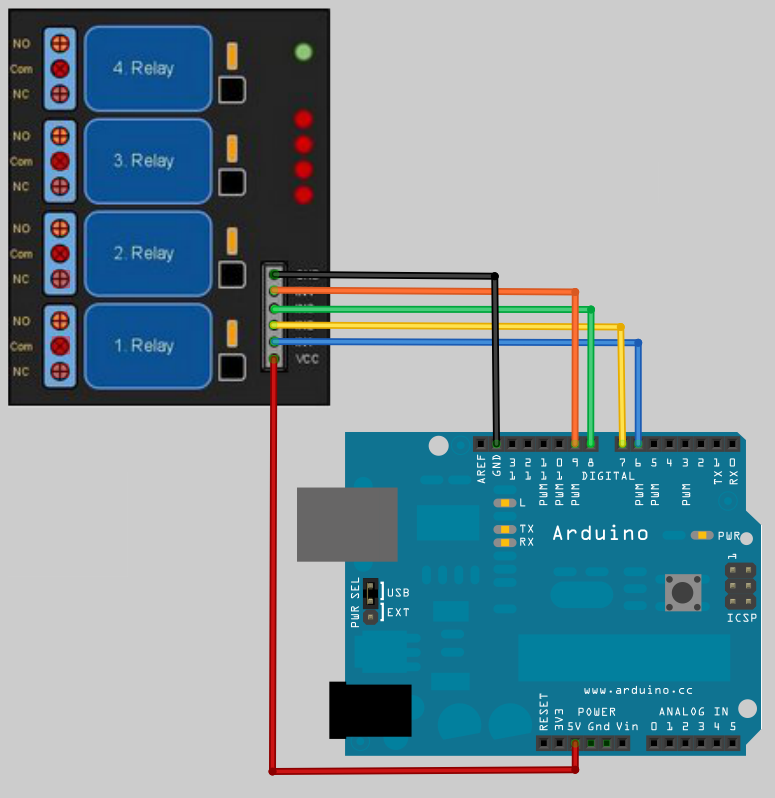
The connections of the Bluetooth module can be a little confusing since TXD goes to Rx and RXD goes to Tx: here's an explanation. Transmit and Receive refer to each device, therefore a transmission coming out of TXD pin of the Bluetooth module must be received by Arduino on the Rx Pin 0; similarly a transmission going out of Arduino Tx Pin 1 must reach the JY-MCU Bluetooth module on its RXD pin.



Warning: The Bluetooth module may interfere with PC to Arduino communication: disconnect VCC when programming the board. (in my tests this has not been the case, but for sure it won't do any harm).

**7.3.2 Relay Arduino Connection**

* Connect Vcc pin of relay to 12V and GND pin to both GND Pin of power supply and GND pin of Arduino
* Connect each channel pin to corresponding output pin of arduino as shown in figure.

****

**7.3.3 Relay to Appliances connection**

COM - Common connection--> it is the center terminal, It is hot as power to the load is connected at this terminal.

NO Normally open ---> It acts like a switch,since it is open - there will be no contact between COM and NO, When we trigger the relay module, it connects to COM by the electromagnet inside the relay and supply to the load is provided,which powers up the light.Thus the circuit is closed until we trigger the state to low in relay.

NC Normally closed---->It is always in contact with COM, even when relay is not powered.when we trigger the relay it opens the circuit, so the connection is lost. it behaves just opposite to NO.  
  
im using NO connection,but here in this type of relay "HIGH" state in code turns off the relay(opens the circuit).  "LOW" state in code turns on the relay.

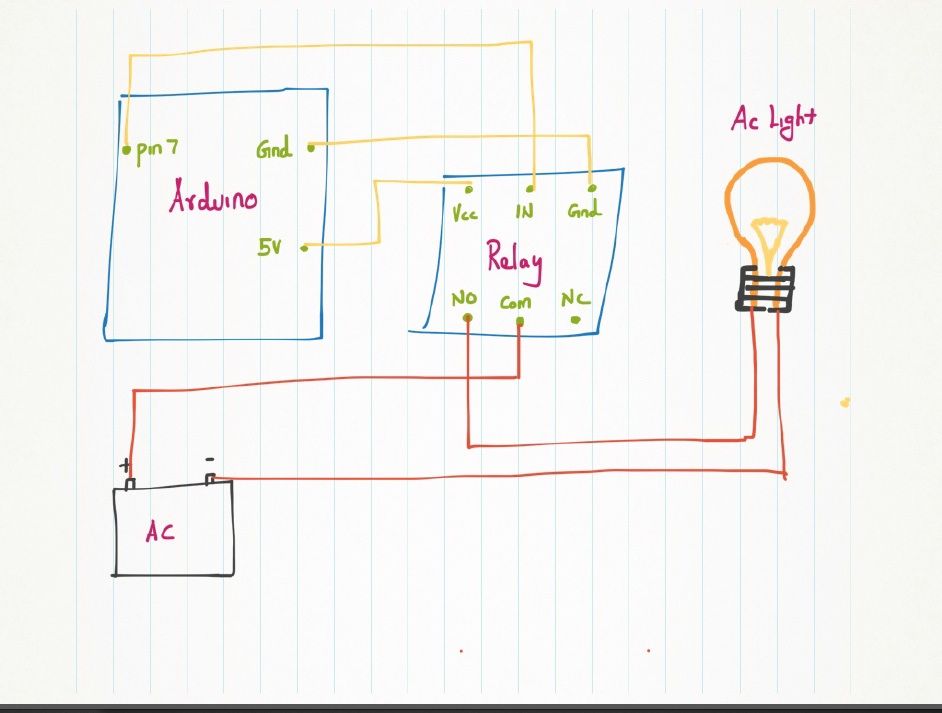
[[](http://cdn.instructables.com/FJQ/CT1J/HTVICQXY/FJQCT1JHTVICQXY.LARGE.jpg)](http://cdn.instructables.com/FJQ/CT1J/HTVICQXY/FJQCT1JHTVICQXY.LARGE.jpg)

We are using Normally open connection in relay. so that we can trigger on and off the light .

Hot line from supply is connected to COM

Supply line to the Ac light is connected to NO

Gnd or - or other terminal in light is connected directly.



CHAPTER 8

**CODING**

**8.1 ARDUINO SKETCH CODE**

float maxTemp = 30.0;

int maxTempSensor = (int) ((maxTemp / 100 + .5) \* 204.8);

float temperature = 0.0;

int maxSeconds = 10;

const int ledPin = 13;

const int tempPin = A0;

const int led1Pin = 3; // Fan

const int led2Pin = 4; // Light

const int led3Pin = 5; // Door

volatile int tempVal;

volatile int seconds = 0;

volatile boolean tempHigh = false;

volatile boolean statusReport = false;

String inputString = "";

String command = "";

String value = "";

boolean stringComplete = false;

void setup(){

//start serial connection

Serial.begin(9600);

Serial.print("Max T: ");

Serial.print(maxTemp);

Serial.print(" Sensor: ");

Serial.println(maxTempSensor);

inputString.reserve(50);

command.reserve(50);

value.reserve(50);

pinMode(ledPin, OUTPUT);

digitalWrite(ledPin, LOW);

pinMode(led1Pin, OUTPUT);

pinMode(led2Pin, OUTPUT);

pinMode(led3Pin, OUTPUT);

digitalWrite(led1Pin, LOW);

digitalWrite(led2Pin, LOW);

digitalWrite(led3Pin, LOW);

cli();

TCCR1A = 0;

TCCR1B = 0;

OCR1A = 15624;

TCCR1B |= (1 << WGM12);

TCCR1B |= (1 << CS10);

TCCR1B |= (1 << CS12);

TIMSK1 |= (1 << OCIE1A);

sei();

}

ISR(TIMER1\_COMPA\_vect)

{

tempVal = analogRead(tempPin);

if (tempVal > maxTempSensor) {

digitalWrite(ledPin, HIGH);

tempHigh = true;

}

else {

digitalWrite(ledPin, LOW);

tempHigh = false;

}

if (seconds++ >= maxSeconds) {

statusReport = true;

seconds = 0;

}

}

void loop(){

int intValue = 0;

if (stringComplete) {

Serial.println(inputString);

boolean stringOK = false;

if (inputString.startsWith("CMD ")) {

inputString = inputString.substring(4);

int pos = inputString.indexOf('=');

if (pos > -1) {

command = inputString.substring(0, pos);

value = inputString.substring(pos+1, inputString.length()-1);

if (command.equals("RED")) { // RED=ON|OFF

value.equals("ON") ? digitalWrite(led3Pin, HIGH) : digitalWrite(led3Pin, LOW);

stringOK = true;

}

else if (command.equals("GREEN")) { // GREEN=ON|OFF

value.equals("ON") ? digitalWrite(led2Pin, HIGH) : digitalWrite(led2Pin, LOW);

stringOK = true;

}

else if (command.equals("YELLOW")) { // YELLOW=ON|OFF

value.equals("ON") ? digitalWrite(led1Pin, HIGH) : digitalWrite(led1Pin, LOW);

stringOK = true;

}

else if (command.equals("TMAX")) { // TMAX=value

intValue = value.toInt();

if (intValue > 0) {

maxTemp = (float) intValue;

maxTempSensor = (int) ((maxTemp / 100 + .5) \* 204.8);

stringOK = true;

}

}

else if (command.equals("SECONDS")) { // SECONDS=value

intValue = value.toInt();

if (intValue > 0) {

maxSeconds = intValue;

stringOK = true;

}

}

} // pos > -1

else if (inputString.startsWith("STATUS")) {

Serial.print("STATUS RED=");

Serial.println(digitalRead(led3Pin));

Serial.print("STATUS GREEN=");

Serial.println(digitalRead(led2Pin));

Serial.print("STATUS YELLOW=");

Serial.println(digitalRead(led1Pin));

Serial.print("STATUS TMAX=");

Serial.println(maxTemp);

Serial.print("STATUS SECONDS=");

Serial.println(maxSeconds);

Serial.print("STATUS TEMP=");

Serial.println(temperature);

Serial.print("STATUS THIGH=");

Serial.println(tempHigh);

stringOK = true;

}

}

stringOK ? Serial.println("Command Executed") : Serial.println("Invalid Command");

inputString = "";

stringComplete = false;

}

if (statusReport) {

temperature = (tempVal \* 0.0048828125 - .5) \* 100;

Serial.print("STATUS TEMP=");

Serial.println(temperature);

Serial.print("STATUS THIGH=");

Serial.println(tempHigh);

statusReport = false;

}

}

void serialEvent() {

while (Serial.available()) {

char inChar = (char)Serial.read();

inputString += inChar;

if (inChar == '\n' || inChar == '\r') {

stringComplete = true;

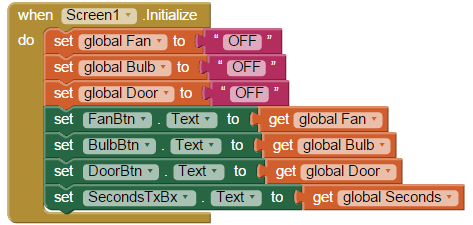
}

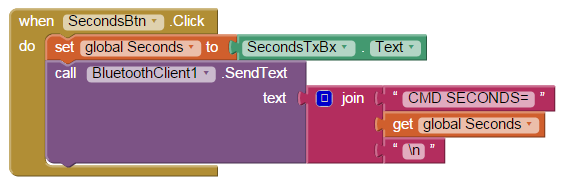
}

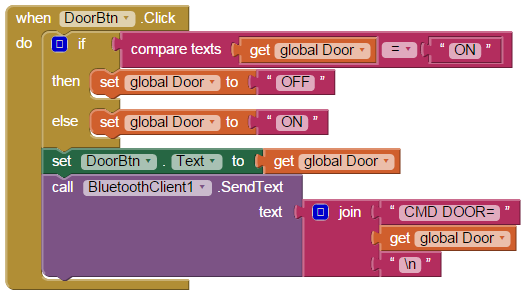
}

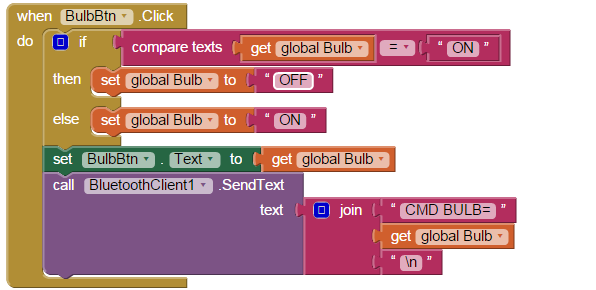
**8.2 ANDROID CODE BLOCK DIAGRAM**

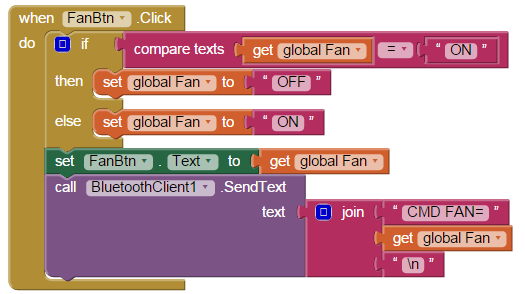
****

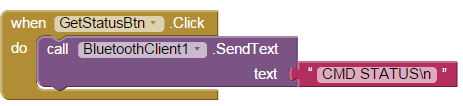
****

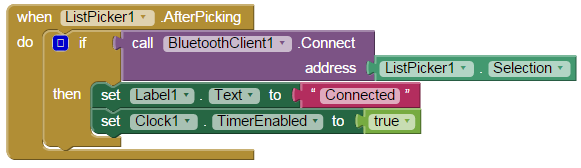
****

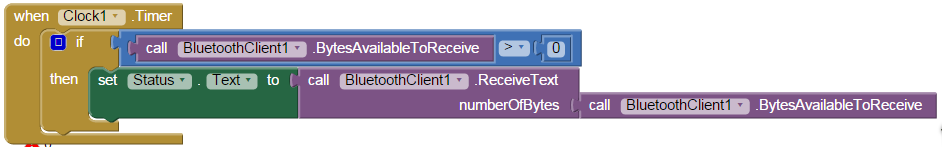
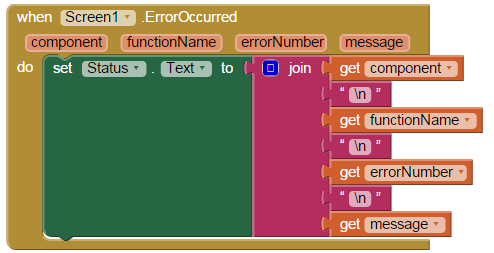
****

****

****

****

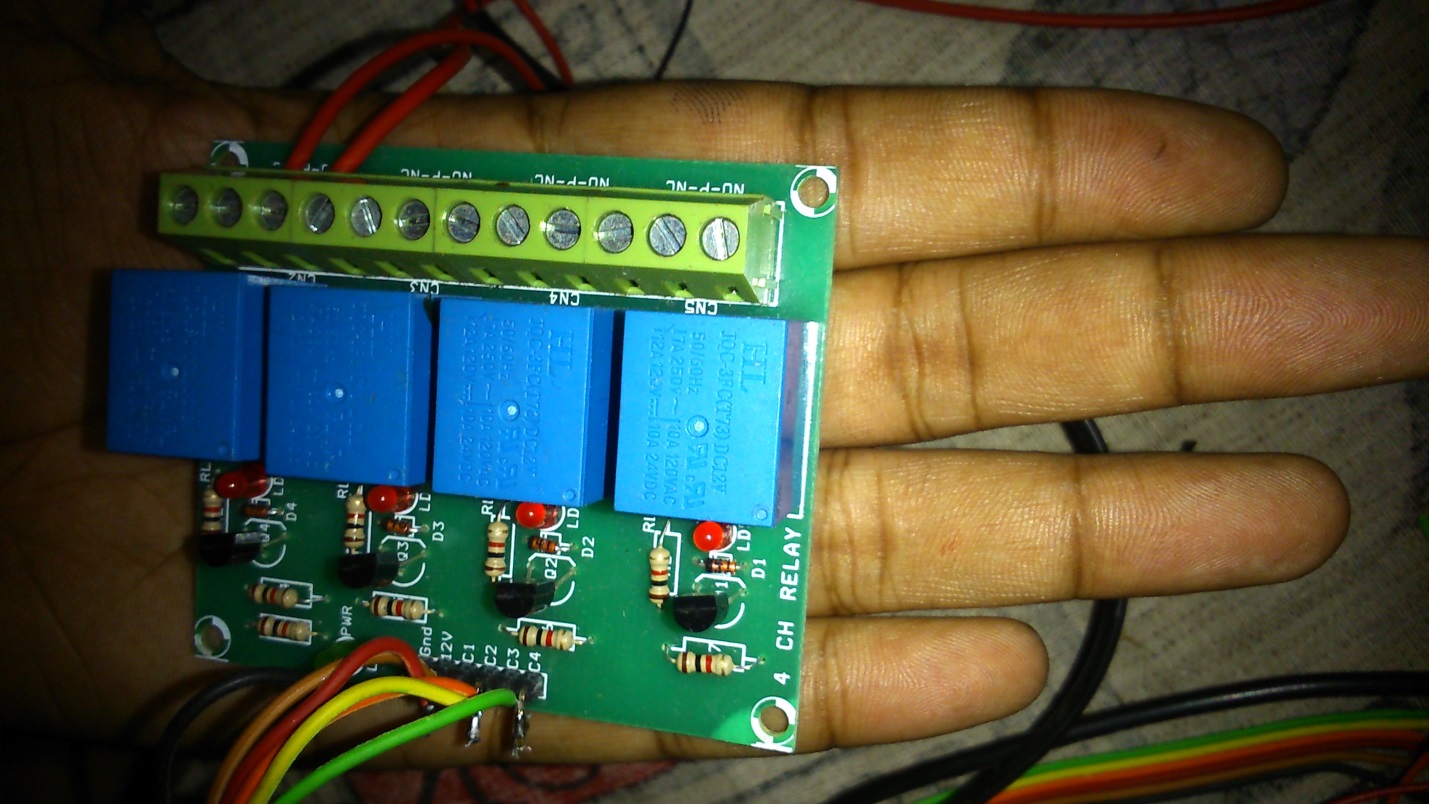
****

****

CHAPTER 9

**SCREENSHOT**

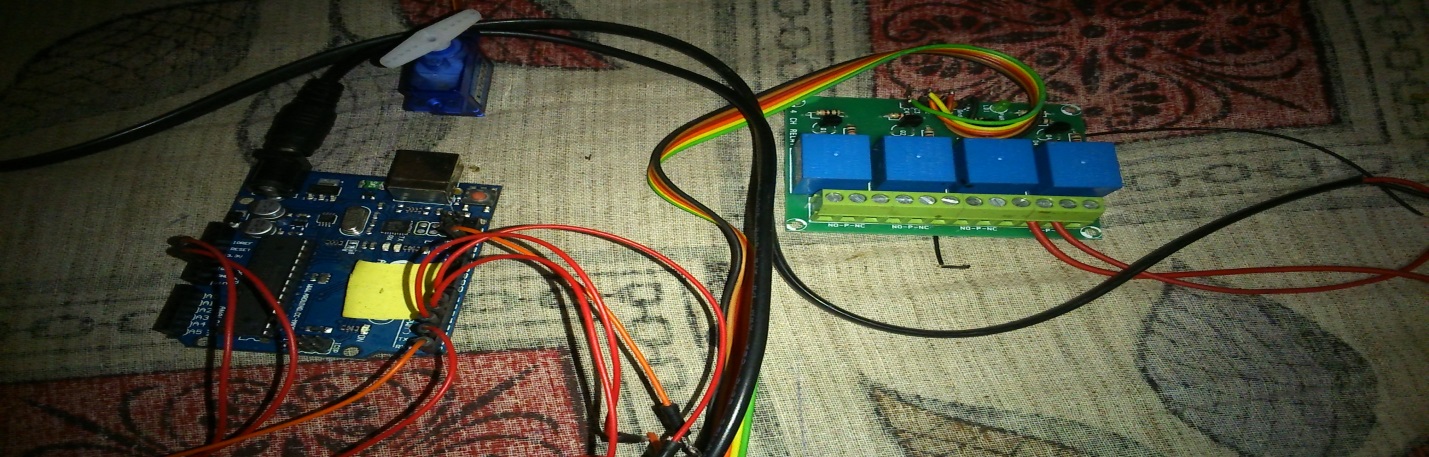
**RELAY**

****

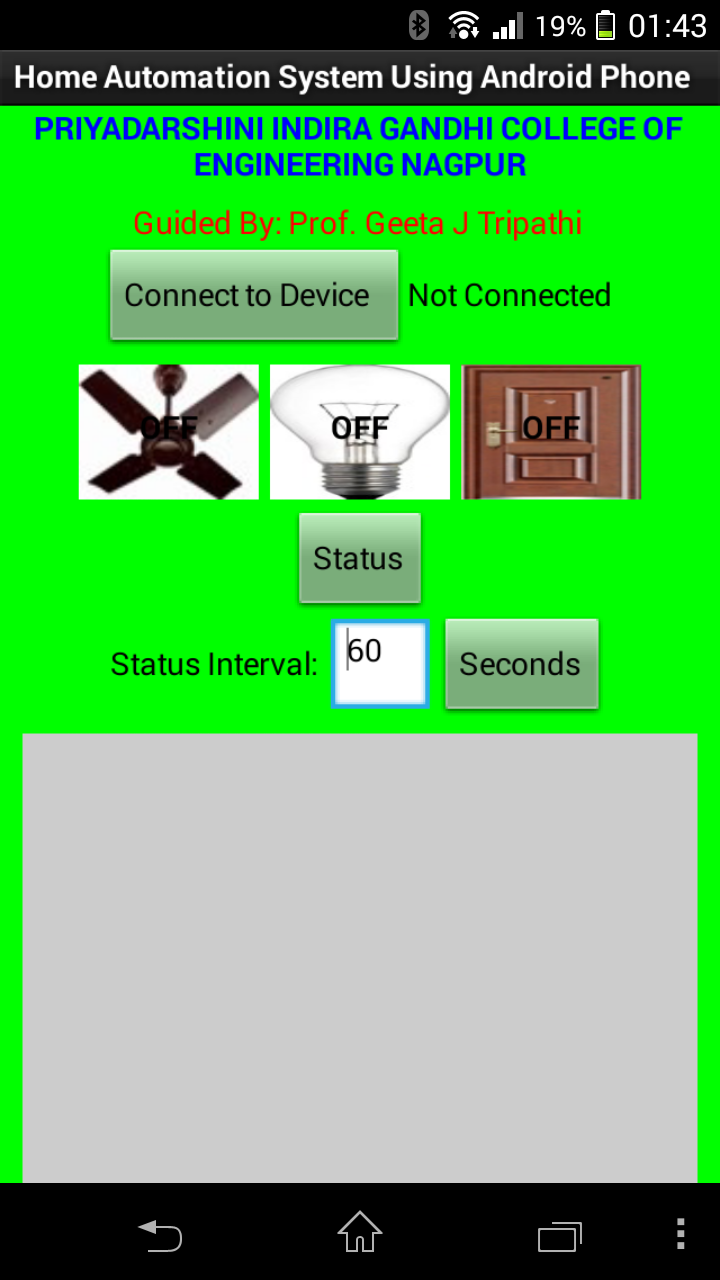
**ARDUINO & BLUETOOTH**

****

**ALL IN ONE**

****

**ANDROID APPLICATION GUI**

****

CHAPTER 10

**ADVANTAGES & APPLICATIONS**

Advantages of Home Automation System

* Home automation makes your home safer: Its visible around us that crime is on the increase, families are beginning to aim at purchasing security alarm systems for their homes. Home Automation adds a extra sense of security to your home, using the automation system you can monitor your homes security cameras using a remote device. You can recieve notifications through email or message, when your alarm has been armed/disarmed or tripped.
* Home automation saves you money and energy:  Home automation systems are designed not to use alot of electricity, a added bonus to having a home automation system is that you can schedule your lighting in your home to go on and off at selected times of the day. Using a remote device, you’ll have remote access to operate any lighting or electrical appliances.
* Home Automation is great for the parents: If you have a newborn or toddler in your home, having a home automation system is perfect for any parent. You can be anywhere in your home, and still be able to monitor your child in a different part of the house.
* A home automation system is a all in one user friendly system: For any technological fanatic this is the ultimate device. From the press of a button on a remote device, the user is able to control Lighting, Air Conditioning, Audio & Video systems and Security Video cameras.

Following are the applications of Home Automation and Security System

* Medical alert / teleassistance.
* Precise and safe blind control.
* Detection of fire, gas leaks and water leaks.
* Smoke detector can detect a fire or smoke condition, causing all lights in the house to blink to alert any person of the house to the possible emergency.
* The system can call the home owner on their mobile phone to alert them, or call the fire department or alarm monitoring company.
* In terms of lighting control, it is possible to save energy when hours of wasted energy in both residential and commercial applications by auto on/off light at night time in all major city office buildings, say after 10pm.
* Control and integration of security systems and also the potential for central locking of all perimeter doors and windows.
* Security cameras can be controlled, allowing the user to observe activity around a house or business right from a Monitor or touch panel.
* Security systems can include motion sensors that will detect any kind of unauthorized movement and notify the user through the security system or via cell phone.
* An intercom system allows communication via a microphone and loud speaker between multiple rooms.

CHAPTER 11

**CONCLUTION & FUTURE SCOPE**

**CONCLUSION**

This is an ongoing project. Our prime objective is to assist handicapped/old aged people. This paper gives basic idea of how to control various home appliances and provide a security using Android phone/tab. This project is based on Android and Arduino platform both of which are FOSS(Free Open Source Software). So the overall implementation cost is very cheap and it is affordable by a common person. Looking at the current scenario we have chosen Android platform so that most of the people can get benefit.

The design consists of Android phone with home automation application, Arduino Mega ADK. User can interact with the android phone and send control signal to the Arduino ADK which in turn will control other embedded devices/sensors. We have discussed a simple prototype in this paper but in future it can be expanded to many other areas.

**FUTURE WORK**

Looking at the current situation we can build cross platform system that can be deployed on various platforms like iOS, Windows. Limitation to control only several devices can be removed by extending automation of all other home appliances. Security cameras can be controlled, allowing the user to observe activity around a house or business. Security systems can include motion sensors that will detect any kind of unauthorized movement and notify the user. Scope of this project can be expanded to many areas by not restricting to only home. It will be flexible to support various wired as well as wireless technologies like Bluetooth, Zigbee, Wi-Fi, World Wide Web.

CHAPTER 12

**REFERENCES**

1. Al-Ali, Member, IEEE & M. AL-Rousan,“Java-Based Home Automation System R.” IEEE Transactions on Consumer Electronics, Vol. 50, No. 2, MAY 2004
2. E. Yavuz, B. Hasan, I. Serkan and K. Duygu. “Safe and Secure PIC Based Remote Control Application for Intelligent Home”. *International Journal of Computer Science and Network Security*, Vol. 7, No. 5, May 2007.
3. R.Piyare, M.Tazil” Bluetooth Based Home Automation System Using Cell Phone”, 2011 IEEE 15th International Symposium on Consumer Electronics
4. http://googleblog.blogspot.in/2011/05/android-momentum-mobile-and-more-at.html
5. http://developer.android.com/about/index.html
6. http://research.microsoft.com/en-us/projects/homeos/
7. http://source.android.com/tech/accessories/index.html
8. http://developer.android.com/guide/topics/connectivity/usb/accessory.html#manifest
9. http://source.android.com/tech/accessories/aoap/aoa.html
10. http://arduino.cc/en/Main/ArduinoBoardADK
11. http://source.android.com/tech/accessories/aoap/aoa2.html.