NATIONAL INSTITUTE OF ELECTRONICS AND INFORMATION TECHNOLOGY

CALICUT

PROJECT REPORT

ON

HAND MOVEMENT BASED DEVICE CONTROL SYSTEM

Submitted in Partial fulfillment of

PG Diploma in Embedded Real-Time Systems

By

MOHAMMED ASAD REHMAN

JAGADISHWAR PAMPAD

Under the Guidance of

Mr. Rajesh M.

(Scientist/Engineer 'C')

Embedded System Group, NIELIT Calicut

August 2016

ABOUT THE ORGANIZATION

National Institute of Electronics and Information Technology (NIELIT), Calicut is a Centre of NIELIT New Delhi, which is an autonomous body of Department of Electronics and Information Technology, Ministry of Communications & Information Technology, Govt. of India. The Centre is a premier organization for education, training, R&D and consultancy in IT and electronics. There are 30 units of NIELIT located in various parts of the country. The Calicut unit of NIELIT was established in 1990.

The Centre has 5 excellent laboratories, fully equipped with the latest systems and development tools in the area of Embedded Systems, Information Technology, Product Engineering, Process Control & Instrumentation and in CAD/CAM. Large collections of reference books in the above areas are accessible to the trainees from the Centre's library in addition to IEEE Online Journal access & NKN connectivity. All the labs, library and office are connected through the central network and students can retrieve information from their terminals itself and through well connected Wi-Fi system.

OBJECTIVES

- To train manpower in electronic design, product design and information technology.
- To undertake product development, contract research and consultancy.
- To develop entrepreneurs and designers in electronic and information technology.
- To impart high quality training in information technology through national level Franchise.
- To enhance technical knowledge and skills of electronic engineers including fresh Graduate engineers.

ACKNOWLEDGEMENT

We would like to express our gratitude to all who helped us directly or indirectly in the successful completion of our project work.

We express our deep sense of gratitude to **Dr. M.P Pillai** (Executive Director, NIELIT, Calicut) for providing an opportunity and facilities to do this project.

We express our sincere gratitude to **Mr. Martin** K **M (**Scientist/Engineer 'E'), Head of the department, Embedded System Group, for providing an opportunity and facilities to do this project.

We express our sincere thanks to **Mr. Shoukath Cherukat** (Scientist/Engineer 'D'), our project guide and course coordinator for his guidance, encouragement and constant support in making this project.

We also wish to thank **Mr. Rajesh M** (Scientist/Engineer 'C'), **Mr. Manoj N** (Technical Officer), **Mr. Arunagirinathan J** (Project Engineer), **Ms. Sruthi P** (Graduate Apprentice) and **Ms. Amna K** (Graduate Apprentice) for providing their valuable support.

ABSTRACT

Traditionally there are switches and buttons for every type of operations. Repeated use of switches and buttons are known to cause chronic illnesses as ‘carpal tunnel syndrome’, ‘repetitive strain injury’ and ‘Parkinson’s disease’.

The work presented here is an effort at reducing the amount of exposure human skeleton has to buttons and switches on an average daily basis. Although we cannot completely replace the use of buttons with non-conventional solutions however, least for non-intensive tasks human’s hand muscles can feel some relief.

The aim of this project is to operate real world devices with the help of pure hand movements hence providing innovative ways of control to its user.

All the motions of hand are sensed by an inertial mass unit accelerometer attached to user’s hand. This determines speed and orientation information of user’s hand and transmits an appropriate control signal over wireless (Bluetooth) link to a control unit integrated with devices.

The control unit receives these values and performs one of these applications, as desired by the user:

1. Computer multimedia control
2. Mouse operation
3. Device operation
4. Home automation

CONTENTS

1. INTRODUCTION

Traditionally devices have been controlled by buttons and switches. Over the time switching devices have evolved. Some of the advanced methods of operation include clap switches, touch switches, capacitive hover switches, motion sensor based switches, infrared block switches, etc.

All these methods require user to go to a certain position near control device to operate, and the mobility is seldom. Extended exposure to mechanical switches is known to cause chronic illnesses like carpal tunnel syndrome, repetitive strain injury and Parkinson's disease.

Apart from the necessary use of buttons and switches (like keyboards, touch phones) the generic use (like light/fan switches, leisure time entertainment media control) can be replaced with an easier way.

In here we have made an effort to control devices with the help of natural hand movements using a wireless device attached to human hand to operate various demonstrative systems.

1.1 LITERATURE SURVEY

Various types of buttons and switches:

Clap Switch has the ability to turn ON/OFF any electrical component or circuit by the clap sound.

It is known as Clap Switch, because the condenser microphone has the ability to take the sound having same pitch as the Clap sound as the input. Although it doesn't mean that the sound will have to be of Clap sound, it can be any sound having the same high pitch as of Clap. We can also say that it converts the Sound energy into the Electrical Energy, because we are give an input to the circuit as a sound whereas the circuit gives us the output as a lamp turning on (Electrical Energy).

Touch switches are the ones that operate when it is touched. There are three types of touch switches available:

1. Capacitive
2. Resistive
3. Piezo switch

A **PIR-based motion detector switch** is used to sense movement of people, animals, or other objects. They are commonly used in burglar alarms and automatically-activated lighting

systems. They are commonly called simply "PIR", or sometimes "PID", for "passive infrared detector".

Automatic **wash basin tap** controller is an **infrared** based system that detects any interruption of the IR rays by our hands or utensils and water automatically starts flowing out of the tap. The circuit mainly comprises Transmitter & Receiver which are built with the 555 Timer. Both require 5 volts D.C. Supply. The IR rays continuously emitted by the transmitter fall on the receiver. Using an IR sensor – TSOP1738 and an infra-red LED. A relay is used in the circuit along with the free-wheeling diode to drive the solenoid. Solenoid is used to lift up the valve fitted in the pipe to let the water flow out of the tap.

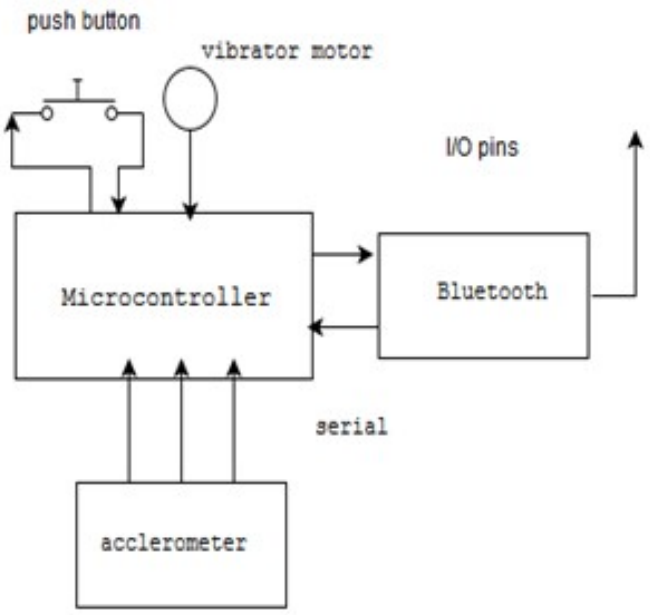
1.2. OBJECTIVE OF THE PROPOSED SYSTEM

The aim of this project is to operate real world devices with the help of pure hand movements hence providing innovative ways of control to its user. Demonstrated here is:

1. Computer multimedia control
2. Mouse operation
3. Device operation
4. Home automation

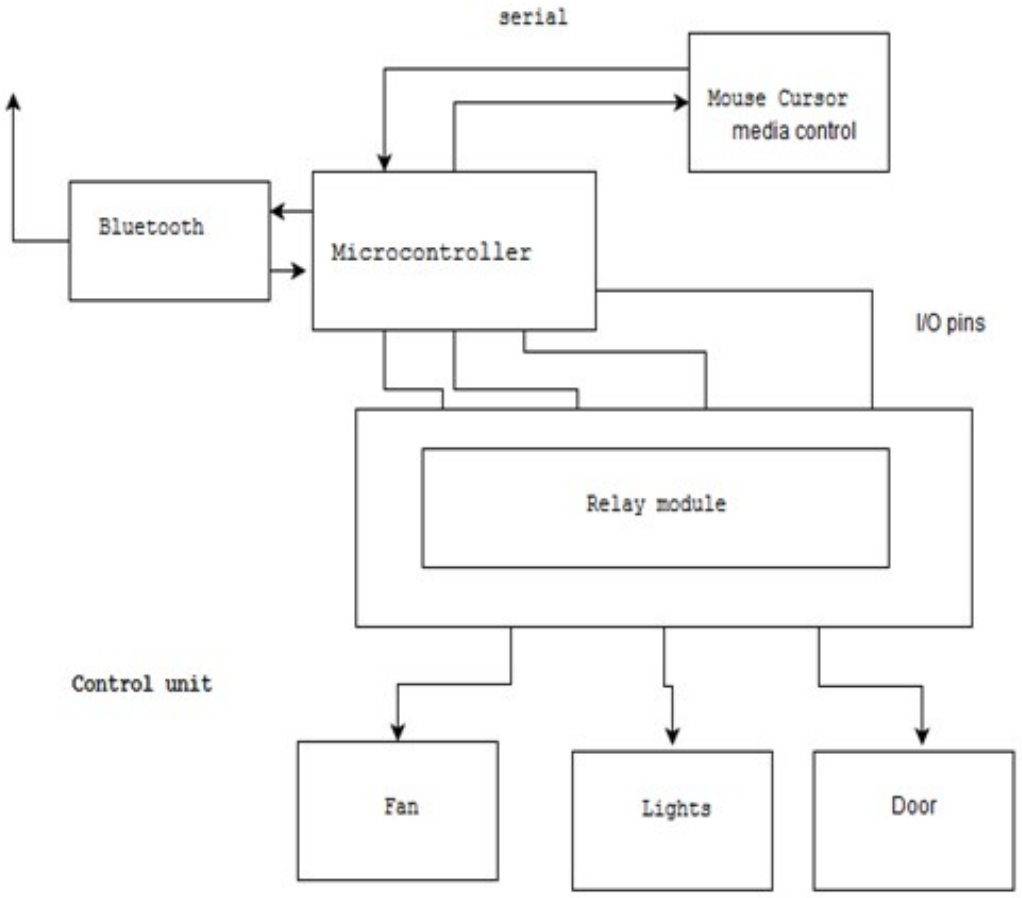
2. BLOCK DIAGRAM

Sensor Unit:



Sensor unit

Application Unit:

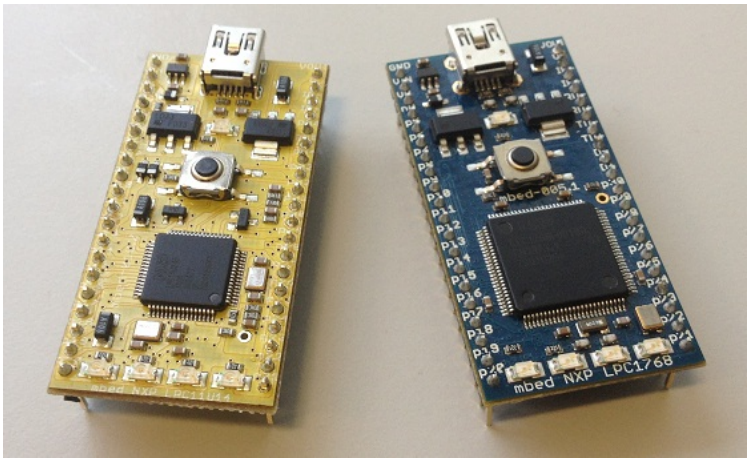


3. HARDWARE DESCRIPTION

3.1 mbed NXP LPC11U24

The mbed Microcontrollers are a series of official mbed prototyping modules based on the mbed HDK. They provide fast, flexible and low-risk and professional rapid prototyping solutions to jump-start your design.

They are packaged in a 40-pin 0.1" DIP form-factor convenient for prototyping with solder less breadboard, strip board, and through-hole PCBs. They include a built-in USB programming interface that is as simple as using a USB Flash Drive. Plug it in, drop on an ARM program binary, and it's up and running!



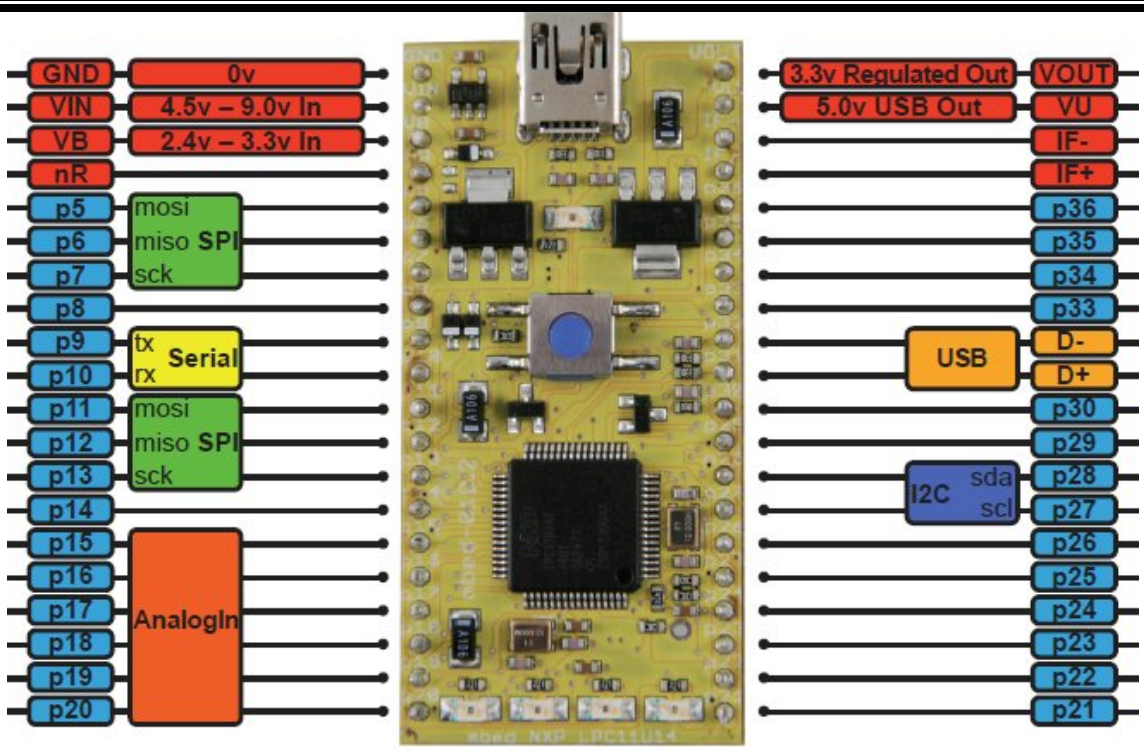
mbed NXP LPC11U24 (left) and mbed NXP LPC1768 (right)

The USB drag 'n' drop programming interface works with Windows, Mac OS X and Linux, meaning you can re-flash the microcontroller without needing drivers or a programming application. The program binary can be easily generated using the mbed Online Compiler, or alternatively using any other standard offline tool chain like Keil uVision, Code Red, Code Sourcery, GCC, or IAR.

There is also support for a virtual serial port using the same USB interface, enabling communication with a PC terminal, Labview, Matlab, and any other programming language that can communicate with a COM port.

mbed NXP LPC11U24
Rapid Prototyping for USB devices, battery applications and low-cost 32-bit ARM® Cortex™-M0 based designs

The mbed NXP LPC11U24 Microcontroller in particular is designed for prototyping low cost USB devices, battery powered applications and 32-bit ARM® Cortex™-M0 based designs. It is packaged as a small DIP form-factor for prototyping with through-hole PCBs, strip board and breadboard, and includes a built-in USB FLASH programmer.



The mbed Microcontrollers provide experienced embedded developers a powerful and productive platform for building proof-of-concepts. For developers new to 32-bit microcontrollers, mbed provides an accessible prototyping solution to get projects built with the backing of libraries, resources and support shared in the mbed community.

Features

- NXP LPC11U24 MCU
  - Low power ARM® Cortex™-M0 Core
  - 48MHz, 8KB RAM, 32KB FLASH
  - USB Device, 2xSPI, I2C , UART, 6xADC, GPIO
- Prototyping form-factor
  - 40-pin 0.1" pitch DIP package, 54x26mm
  - 5V USB, 4.5-9V supply or 2.4-3.3V battery
  - Built-in USB drag 'n' drop FLASH programmer
- mbed.org Developer Website
  - Lightweight Online Compiler
  - High level C/C++ SDK
  - Cookbook of published libraries and projects

Tools and Software

The mbed Microcontrollers are all supported by the mbed.org developer website, including a lightweight Online Compiler for instant access to your working environment on Windows, Linux or Mac OS X.

Also included is a C/C++ SDK for productive high-level programming of peripherals. Combined with the wealth of libraries and code examples being published by the mbed community, the platform provides a productive environment for getting things done.

3.2 ACCELEROMETER

An accelerometer is an electromechanical device used to measure acceleration forces. Such forces may be static, like the continuous force of gravity or, as is the case with many mobile devices, dynamic to sense movement or vibrations.

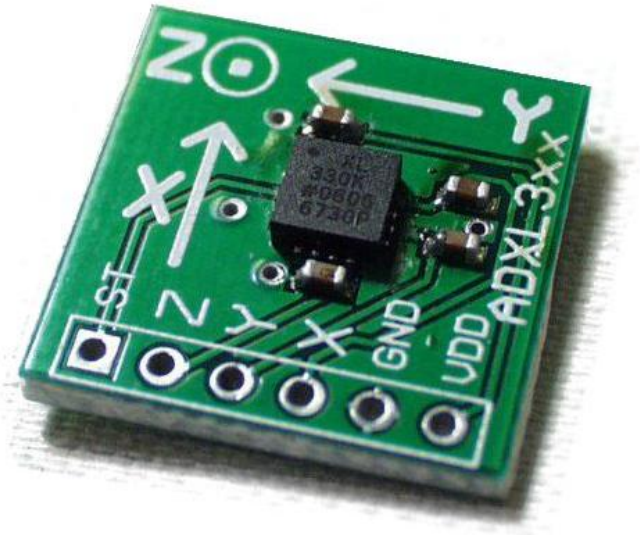
Acceleration is the measurement of the change in velocity, or speed divided by time. For example, a car accelerating from a standstill to 60 mph in six seconds is determined to have an acceleration of 10 mph per second (60 divided by 6).

The purpose of the accelerometer

The application of accelerometers extends to multiple disciplines, both academic and consumer-driven. For example, accelerometers in laptops protect hard drives from damage. If the laptop were to suddenly drop while in use, the accelerometer would detect the sudden free fall and immediately turn off the hard drive to avoid hitting the reading heads into the hard drive platter. Without this, the two would strike and cause scratches to the platter for extensive file and reading damage. Accelerometers are likewise used in cars as the industry method way of detecting car crashes and deploying airbags almost instantaneously.

In another example, a dynamic accelerometer measures gravitational pull to determine the angle at which a device is tilted with respect to the Earth. By sensing the amount of acceleration, users analyze how the device is moving.

Accelerometers allow the user to understand the surroundings of an item better. With this small device, you can determine if an object is moving uphill, whether it will fall over if it tilts any more, or whether it's flying horizontally or angling downward. For example, smart phones rotate their display between portrait and landscape mode depending on how you tilt the phone..



How they work

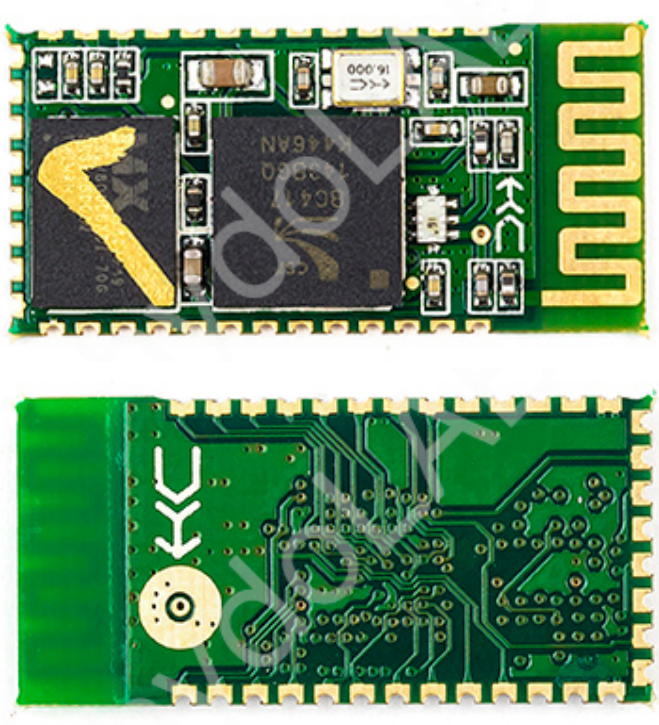
An accelerator looks like a simple circuit for some larger electronic device. Despite its humble appearance, the accelerometer consists of many different parts and works in many ways, two of which are the piezoelectric effect and the capacitance sensor. The piezoelectric effect is the most common form of accelerometer and uses microscopic crystal structures that become stressed due to accelerative forces. These crystals create a voltage from the stress, and the accelerometer interprets the voltage to determine velocity and orientation.

The capacitance accelerometer senses changes in capacitance between microstructures located next to the device. If an accelerative force moves one of these structures, the capacitance will change and the accelerometer will translate that capacitance to voltage for interpretation.

Our accelerometer is made up of 3-axes. Most smart phones typically make use of three-axis models, whereas cars simply use only a two-axis to determine the moment of impact. The sensitivity of these devices is quite high as they're intended to measure even very minute shifts in acceleration. The more sensitive the accelerometer, the more easily it can measure acceleration.

3.3 BLUETOOTH HC-05

It is a class-2 Bluetooth module with Serial Port Profile, which can configure as either Master or slave. A Drop-in replacement for wired serial connections, transparent usage. Used simply for a serial port replacement to establish connection between MCU, PC to embedded project.



HC-05 Specification:

- Bluetooth protocol: 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
- Power supply: +3.3VDC 50mA
- Working temperature: -20 ~ +75Centigrade
- Dimension: 26.9mm x 13mm x 2.2 mm

3.4 VIBRATION MOTOR

These tiny motors have offset weights that make them vibrate when they spin. They're normally called "pager motors" because they're the type found in pagers and cell phones that have a "vibrate" feature.

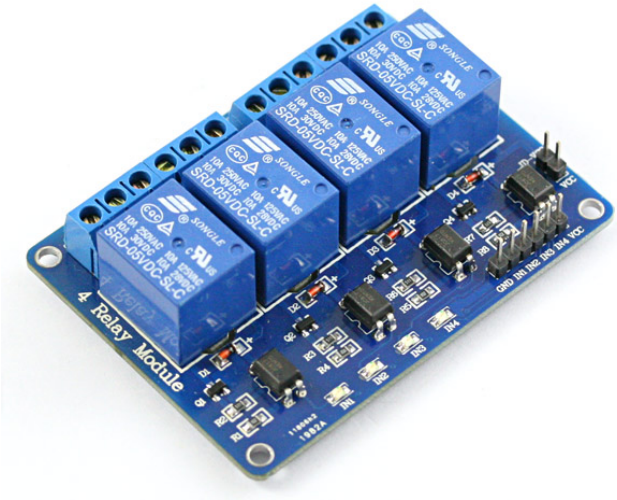
Applications
The perfect thing for making Bristlebots, vibrobots, BEAM bots, and other tiny robots! These motors can be driven with 3 V coin cells like the CR2032. Each one comes in a removable rubber boot that has one flat side for easy mounting.

Specification

- Nominal voltage: 3 V
- Operating voltage: 2.0 ~ 3.5 V
- Rated current: 85 mA
- Nominal speed: ~12000 RPM
- Diameter: 5mm
- Length: 15mm

3.5 RELAY MODULE

A **relay** is an **electrically operated switch** that allows you to turn on or off a circuit using voltage and/or current much higher than the mbed board could handle. There is no connection between the low voltage circuit operated by mbed board and the high power circuit. The relay protects each circuit from each other.



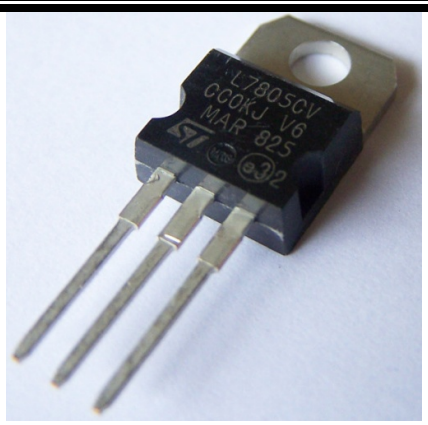
Input: The relay is a simple mechanical on/off switch. It activates when the input reaches 5v and turns off when the input is 0v. The module provides three connections labeled COM, NC and NO. NC stands for "NORMALLY CLOSED"***. This means that when the relay has no signal (LOW or 0V from mbed board), the connected circuit will be active; conversely, if you apply 5V or pull the pin HIGH, it will turn the connected circuit off. **NO stands for "NORMALLY OPEN", and functions in the opposite way; when you apply 5V the circuit turns on, and at 0V the circuit turns off. Relays can replace a manual switch. Remove the switch and connect its wires to COM and NO. When the relay is activated the circuit is closed and current can flow to the device being controlled.

Module Description: this module features four 220v 10A mounted on a4 module board, one standard 3pin connector, one transistor, a green LED that signals that the module is correctly powered and an yellow LED that indicates when the relay is active.

This module is an ACTUATOR. The connector is an INPUT which must be connected to one of the OUTPUT pins of mbed board.

3.6 7805 VOLTAGE REGULATOR

This series of fixed-voltage integrated-circuit voltage regulators is designed for a wide range of applications. These applications include on-card regulation for elimination of noise and distribution problems associated with single-point regulation. Each of these regulators can deliver up to 1.5 A of output current. The internal current-limiting and thermal-shutdown features of these regulators essentially make them immune to overload. In addition to use as fixed-voltage regulators, these devices can be used with external components to obtain adjustable output voltages and currents, and also can be used as the power-pass element in precision regulators.



Features:

- 3-Terminal Regulators
- Output Current up to 1.5 A
- Internal Thermal-Overload Protection
- High Power-Dissipation Capability
- Internal Short-Circuit Current Limiting
- Output Transistor Safe-Area Compensation

3.7 BATTERY

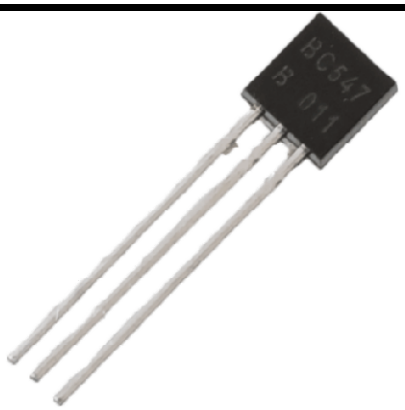
Required power 100mA@5V

Specifications:

- Capacity 11000 mAh
- Power Input DC5V / 2.1A
- Power Source Mini USB
- Output Power 5V 1A, 5V2A & 5V 2A

3.8 BC-547 Switching diode

BC547 is an NPN bi-polar junction transistor. A transistor, stands for transfer of resistance, is commonly used to amplify current. A small current at its base controls a larger current at collector & emitter terminals.



BC547 is mainly used for amplification and switching purposes. It has a maximum current gain of 800. Its equivalent transistors are BC548 and BC549.

The transistor terminals require a fixed DC voltage to operate in the desired region of its characteristic curves. This is known as the biasing. For amplification applications, the transistor is biased such that it is partly on for all input conditions. The input signal at base is amplified and taken at the emitter. BC547 is used in common emitter configuration for amplifiers. The voltage divider is the commonly used biasing mode. For switching applications, transistor is biased so that it remains fully on if there is a signal at its base. In the absence of base signal, it gets completely off.
Features:

- Switching and Applications
- High Voltage: BC546, V_{CEO}=65V
- Low Noise: BC549, BC550
- Complement to BC556 ... BC560

3.9 Push button

Single Pole Single Throw switch, these buttons are rated up to 50mA.

3.10 Laboratory Resistor, Capacitor,

Resistor:

A mixture of various resistive and capacitive elements was used for various applications in the circuit.

Resistors are the most commonly used component in electronics and their purpose is to create specified values of current and voltage in a circuit. Resistors with power dissipation below 5 watt (most commonly used types) are cylindrical in shape, with a wire protruding from each end for connecting to a circuit (see figure). The unit for measuring resistance is the OHM.

Capacitor:

The capacitor is a component which has the ability or "capacity" to store energy in the form of an electrical charge producing a potential difference (Static Voltage) across its plates, much like a small rechargeable battery.

There are many different kinds of capacitors available from very small capacitor beads used in resonance circuits to large power factor correction capacitors, but they all do the same thing, they store charge.

In its basic form, a 'capacitor' consists of two or more parallel conductive (metal) plates which are not connected or touching each other, but are electrically separated either by air or by some form of a good insulating material such as waxed paper, mica, ceramic, plastic or some form of a liquid gel as used in electrolytic capacitors. The insulating layer between a capacitor's plates is commonly called the Dielectric.

A typical capacitor.

Due to this insulating layer, DC current can not flow through the capacitor as it blocks it allowing instead a voltage to be present across the plates in the form of an electrical charge.

3.11 220V AC DEVICES

Electrical Lamp

An electric lamp is a conventional light emitting component used in different circuits, mainly for lighting and indicating purposes. The construction of lamp is quite simple, it has one filament surrounding which, a transparent glass made spherical cover is provided. The filament of the lamp is mainly made of tungsten as it has high melting point temperature. A lamp emits light energy as the thin small tungsten filament of lamp glows without being melted, while current flows through it.

The electrical fan:

The electric fan has blades similar to a water or steam turbine. A DC or AC motor drives a rotating shaft.

4. SOFTWARE TOOL DESCRIPTION

SOFTWARE OVERVIEW

During the implementation of our project we have utilized certain software. The source code for the mbed LPC11u24 was written in programming language C. The IDE used was mbed online IDE.

4.1 mbed Online Compiler

The online compiler enables you to either write your code from scratch or import an existing project and modify it to suit your needs. The only thing you need is an mbed developer account.

The compiler is always available on https://developer.mbed.org/compiler/.

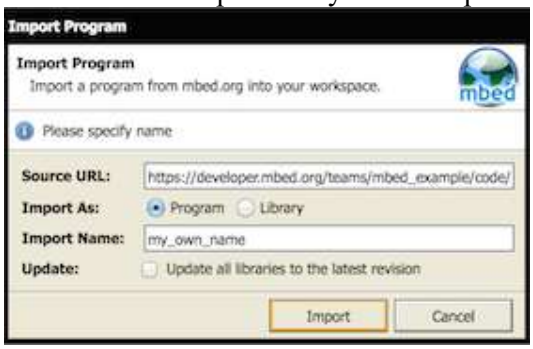
Importing code to the compiler

There are two methods of importing the code into the online compiler: directly from a program presented on the site, or using the compiler's Import button:

1. Directly from the site: wherever you see a program on the site, you should see an Import button:

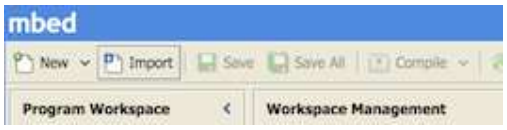
Most code snippets on the site can be directly imported

Clicking that button will take you to the compiler; you can then give the program a new name and import it to your workspace:



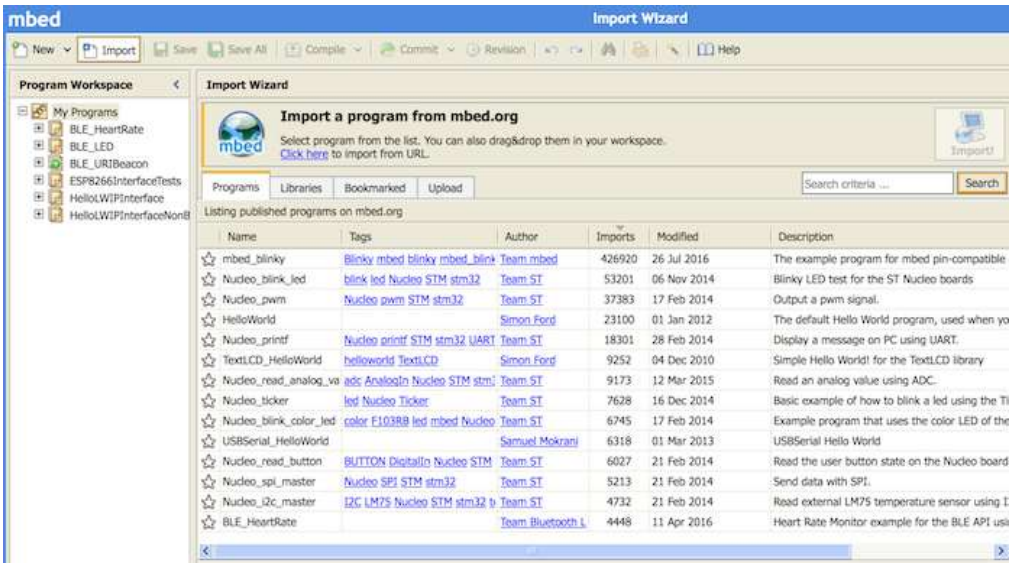
Importing to the mbed Online Compiler

2. The compiler's Import button: click the **Import** button in the compiler to open the Import Wizard:



Triggering the Import Wizard from within the mbed Online Compiler

You can search for a program by name, or perform an empty search to show all available programs:

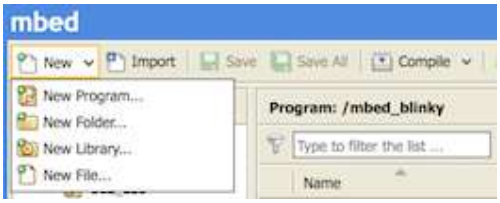


The applications list

Double click a program to import it.

Creating a new program

1. From the **New** menu, select **Program**:

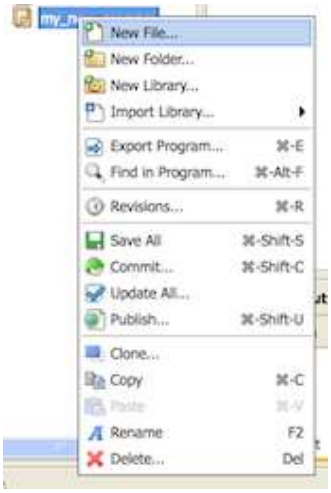


The applications listTriggering a new program

2. The **Create new program** pop-up opens.
    1. Select your platform (board).
    2. You can create from an existing template, or from an empty program.
    3. Enter a unique name.

Creating a new program

3. Create a main.cpp file in your program:
    1. Right click on the program and select **New File…**. The **Create new file** popup opens.



Adding a file

    2. Enter main.cpp as the file name.

Naming the new file

4. Import the mbed OS library so you can build your program with the mbed OS code base:
    1. Click **Import**. The Import Wizard opens.
    2. Go to the Libraries tab and search for “mbed”, or perform an empty search to show all libraries:

Searching for mbed OS

3. Double click on “mbed” to import it. The library is added to your program.

mbed OS is now in the workspace

Getting your program on your board

The mbed Online Compiler builds a file that can run on your board. All you need to do is:

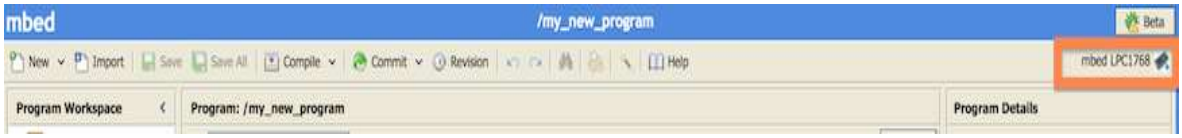
1. Select the correct board.
2. Compile the code and download the compiled file.
3. Copy the file to your board.

Selecting your board

mbed programs can be built to run on multiple boards. The hard work is done behind the scenes, by mbed OS itself. All you need to do is tell the mbed Online Compiler which board you’re building for.

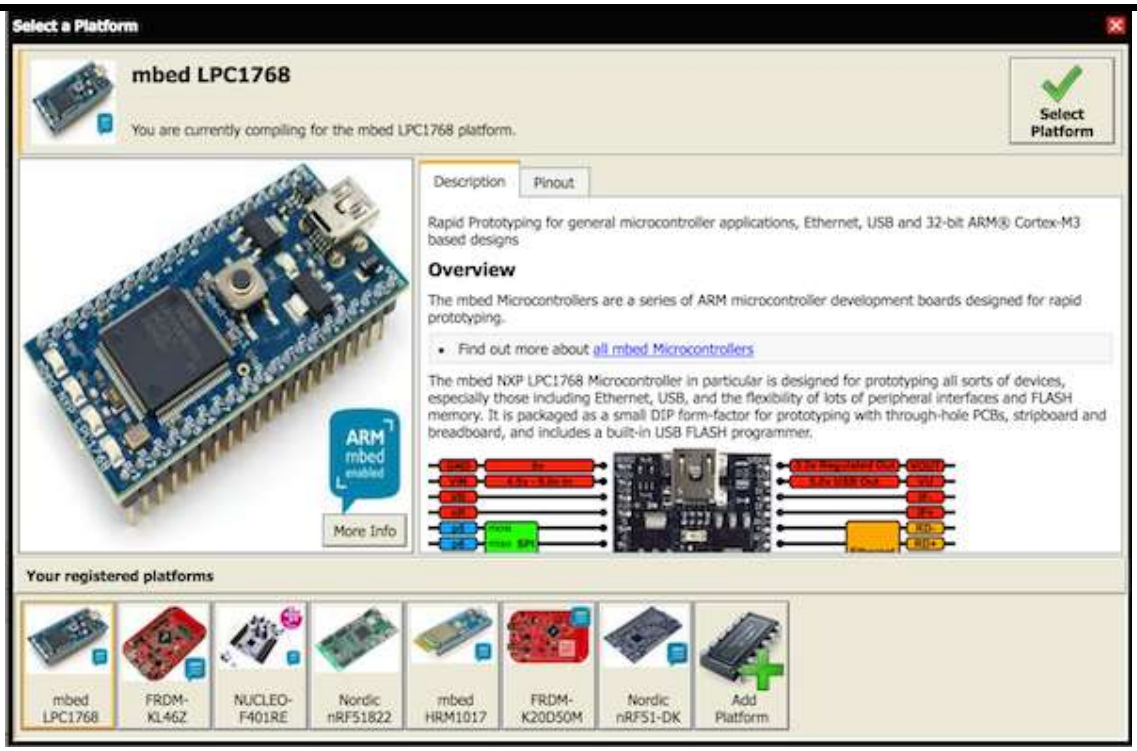
To select a board as the build target:

1. The compiler shows the current build board’s name on the upper right corner:



Showing current board. Click the board to open the full list

2. Click the name of the board you need:



Click a board to set it as the compilation board

If the board isn't already on your list, go to the board's page on mbed.com and click the **Add to your mbed Compiler** button:

Adding a board to the mbed Online Compiler list

Compile and download

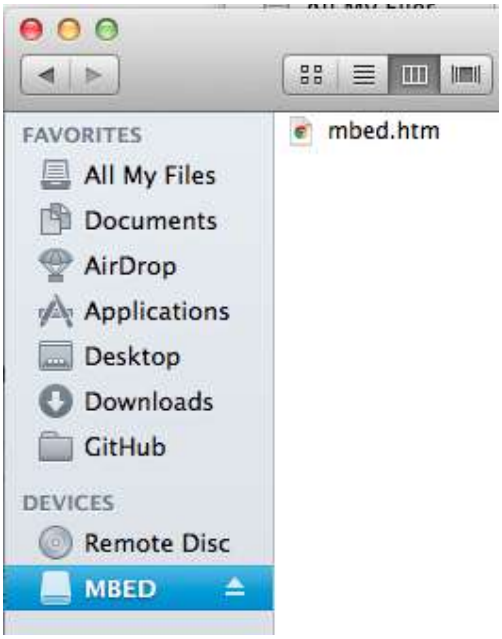
The **Compile** menu offers five options:

1. **Compile:** builds your code and downloads a binary file. Current program only.
2. **Compile All:** same as *compile*, but for all programs.
3. **Build Only:** compiles your code but doesn't download the result.
4. **Compile Macros:** defines additional macros at compile time.
5. **Update Docs**: see the guide for documenting APIs.

Copying the file to the board

Note: If you're working on Windows, you might need to install a driver to allow you to copy to your board. Please see the Windows Serial Driver section.

Your board should appear on your computer as removable storage. To run your program on the board, simply drag and drop the file you downloaded in the previous section.



The device appears as removable storage, under the name MBED

4.2Python

If you do much work on computers, eventually you find that there's some task you'd like to automate. For example, you may wish to perform a search-and-replace over a large number of text files, or rename and rearrange a bunch of photo files in a complicated way. Perhaps you'd like to write a small custom database, or a specialized GUI application, or a simple game.

If you're a professional software developer, you may have to work with several C/C++/Java libraries but find the usual write/compile/test/re-compile cycle is too slow. Perhaps you're writing a test suite for such a library and find writing the testing code a tedious task. Or

maybe you've written a program that could use an extension language, and you don't want to design and implement a whole new language for your application.

Python is just the language for you.

You could write a Unix shell script or Windows batch files for some of these tasks, but shell scripts are best at moving around files and changing text data, not well-suited for GUI applications or games. You could write a C/C++/Java program, but it can take a lot of development time to get even a first-draft program. Python is simpler to use, available on Windows, Mac OS X, and Unix operating systems, and will help you get the job done more quickly.

Python is simple to use, but it is a real programming language, offering much more structure and support for large programs than shell scripts or batch files can offer. On the other hand, Python also offers much more error checking than C, and, being a very-high-level language, it has high-level data types built in, such as flexible arrays and dictionaries. Because of its more general data types Python is applicable to a much larger problem domain than Awk or even Perl, yet many things are at least as easy in Python as in those languages.

Python allows you to split your program into modules that can be reused in other Python programs. It comes with a large collection of standard modules that you can use as the basis of your programs — or as examples to start learning to program in Python. Some of these modules provide things like file I/O, system calls, sockets, and even interfaces to graphical user interface toolkits like Tk.

Python is an interpreted language, which can save you considerable time during program development because no compilation and linking is necessary. The interpreter can be used interactively, which makes it easy to experiment with features of the language, to write throw-away programs, or to test functions during bottom-up program development. It is also a handy desk calculator.

Python enables programs to be written compactly and readably. Programs written in Python are typically much shorter than equivalent C, C++, or Java programs, for several reasons:

• the high-level data types allow you to express complex operations in a single statement;

• statement grouping is done by indentation instead of beginning and ending brackets;

• no variable or argument declarations are necessary.

Python is extensible: if you know how to program in C it is easy to add a new built-in function or module to the interpreter, either to perform critical operations at maximum speed, or to link Python programs to libraries that may only be available in binary form (such as a vendor-specific graphics library). Once you are really hooked, you can link the Python interpreter into an application written in C and use it as an extension or command language for that application.

By the way, the language is named after the BBC show "Monty Python's Flying Circus" and has nothing to do with reptiles. Making references to Monty Python skits in documentation is not only allowed, it is encouraged!

Now that you are all excited about Python, you'll want to examine it in some more detail. Since the best way to learn a language is to use it, the tutorial invites you to play with the Python interpreter as you read.

In the next chapter, the mechanics of using the interpreter are explained. This is rather mundane information, but essential for trying out the examples shown later.

The rest of the tutorial introduces various features of the Python language and system through examples, beginning with simple expressions, statements and data types, through functions and modules, and finally touching upon advanced concepts like exceptions and user-defined classes.

4.2.1.1 Purpose

The purpose of PyAutoGUI is to provide a cross-platform Python module for GUI automation for human beings. The

API is designed to be as simple as possible with sensible defaults.

For example, here is the complete code to move the mouse to the middle of the screen on Windows, OS X, and Linux:

```
>>> import pyautogui

>>> screenWidth, screenHeight = pyautogui.size()

>>> pyautogui.moveTo(screenWidth / 2, screenHeight / 2)
```

And that is all.

PyAutoGUI can simulate moving the mouse, clicking the mouse, dragging with the mouse, pressing keys, pressing

and holding keys, and pressing keyboard hotkey combinations.

4.2.1.2 Examples

```
>>> import pyautogui

>>> screenWidth, screenHeight = pyautogui.size()

>>> currentMouseX, currentMouseY = pyautogui.position()

>>> pyautogui.moveTo(100, 150)

>>> pyautogui.click()

>>> pyautogui.moveRel(None, 10) # move mouse 10 pixels down

>>> pyautogui.doubleClick()

>>> pyautogui.moveTo(500, 500, duration=2, tween=pyautogui.tweens.easeInOutQuad) # use tweening/easing function to move mouse over 2 seconds.
```

>>> pyautogui.typewrite('Hello world!', interval=0.25) # type with quarter-second pause in between each key

>>> pyautogui.press('esc')

>>> pyautogui.keyDown('shift')

>>> pyautogui.press(['left', 'left', 'left', 'left', 'left', 'left'])

>>> pyautogui.keyUp('shift')

>>> pyautogui.hotkey('ctrl', 'c')

This example drags the mouse in a square spiral shape in MS Paint (or any graphics drawing program):

>>> distance = 200

>>> while distance > 0:

pyautogui.dragRel(distance, 0, duration=0.5) # move right

distance -= 5

pyautogui.dragRel(0, distance, duration=0.5) # move down

XY coordinates have 0, 0 origin at top left corner of the screen. X increases going right, Y increases going down.

>>> pyautogui.moveTo(x, y, duration=num_seconds) # move mouse to XY coordinates over num_second seconds

>>> pyautogui.moveRel(xOffset, yOffset, duration=num_seconds) # move mouse relative to its current position

If duration is 0 or unspecified, movement is immediate. Note: dragging on Mac can’t be immediate.

>>> pyautogui.dragTo(x, y, duration=num_seconds) # drag mouse to XY

>>> pyautogui.dragRel(xOffset, yOffset, duration=num_seconds) # drag mouse relative to its current position

Calling click() just clicks the mouse once with the left button at the mouse’s current location, but the keyword

arguments can change that:

>>> pyautogui.click(x=moveToX, y=moveToY, clicks=num_of_clicks, interval=secs_between_clicks, button='left')

The button keyword argument can be ’left’, ’middle’, or ’right’.

All clicks can be done with click(), but these functions exist for readability. Keyword args are optional:

>>> pyautogui.rightClick(x=moveToX, y=moveToY)

>>> pyautogui.middleClick(x=moveToX, y=moveToY)

>>> pyautogui.doubleClick(x=moveToX, y=moveToY)

>>> pyautogui.tripleClick(x=moveToX, y=moveToY)

Positive scrolling will scroll up, negative scrolling will scroll down:

>>> pyautogui.scroll(amount_to_scroll, x=moveToX, y=moveToY)

Individual button down and up events can be called separately:

>>> pyautogui.mouseDown(x=moveToX, y=moveToY, button='left')

>>> pyautogui.mouseUp(x=moveToX, y=moveToY, button='left')

The moveTo() function will move the mouse cursor to the X and Y integer coordinates you pass it. The None value

can be passed for a coordinate to mean “the current mouse cursor position”. For example:

>>> pyautogui.moveTo(100, 200) # moves mouse to X of 100, Y of 200.

>>> pyautogui.moveTo(None, 500) # moves mouse to X of 100, Y of 500.

>>> pyautogui.moveTo(600, None) # moves mouse to X of 600, Y of 500.

Normally the mouse cursor will instantly move to the new coordinates. If you want the mouse to gradually move to

the new location, pass a third argument for the duration (in seconds) the movement should take. For example:

>>> pyautogui.moveTo(100, 200, 2) # moves mouse to X of 100, Y of 200 over 2 seconds

(If the duration is less than pyautogui.MINIMUM_DURATION the movement will be instant. By default,

pyautogui.MINIMUM_DURATION is 0.1.)

If you want to move the mouse cursor over a few pixels relative to its current position, use the moveRel() function.

This function has similar parameters as moveTo(). For example:

>>> pyautogui.moveTo(100, 200) # moves mouse to X of 100, Y of 200.

>>> pyautogui.moveRel(0, 50) # move the mouse down 50 pixels.

>>> pyautogui.moveRel(-30, 0) # move the mouse left 30 pixels.

>>> pyautogui.moveRel(-30, None) # move the mouse left 30 pixels.

4.2.1.3 Mouse Drags

PyAutoGUI's dragTo() and dragRel() functions have similar parameters as the moveTo() and moveRel()

functions. In addition, they have a button keyword which can be set to 'left', 'middle', and 'right' for

which mouse button to hold down while dragging. For example:

>>> pyautogui.dragTo(100, 200, button='left') # drag mouse to X of 100, Y of 200 while holding down left mouse button

>>> pyautogui.dragTo(300, 400, 2, button='left') # drag mouse to X of 300, Y of 400 over 2 seconds while holding down left mouse button

>>> pyautogui.dragRel(30, 0, 2, button='right') # drag the mouse left 30 pixels over 2 seconds while holding down the right mouse button

4.2.1.4 Tween / Easing Functions

Tweening is an extra feature to make the mouse movements fancy. You can probably skip this section if you don't care

about this.

A tween or easing function dictates the progress of the mouse as it moves to its destination. Normally when moving

the mouse over a duration of time, the mouse moves directly towards the destination in a straight line at a constant

speed. This is known as a linear tween or linear easing function.

PyAutoGUI has other tweening functions available in the pyautogui.tweens module. The

pyautogui.tweens.easeInQuad function can be passed for the 4th argument to moveTo(), moveRel(),

dragTo(), and dragRel() functions to have the mouse cursor start off moving slowly and then speeding

up towards the destination. The total duration is still the same as the argument passed to the function. The

pyautogui.tweens.easeOutQuad is the reverse: the mouse cursor starts moving fast but slows down as it

approaches the destination. The pyautogui.tweens.easeOutElastic will overshoot the destination and

“rubber band” back and forth until it settles at the destination.

For example:

>>> pyautogui.moveTo(100, 100, 2, pyautogui.easeInQuad) # start slow, end fast

>>> pyautogui.moveTo(100, 100, 2, pyautogui.easeOutQuad) # start fast, end slow

>>> pyautogui.moveTo(100, 100, 2, pyautogui.easeInOutQuad) # start and end fast, slow in middle

>>> pyautogui.moveTo(100, 100, 2, pyautogui.easeInBounce) # bounce at the end

>>> pyautogui.moveTo(100, 100, 2, pyautogui.easeInElastic) # rubber band at the end

These tweening functions are copied from Al Sweigart’s PyTweening module:

https://pypi.python.org/pypi/PyTweening https://github.com/asweigart/pytweening This module does not have

to be installed to use the tweening functions.

If you want to create your own tweening function, define a function that takes a single float argument between 0.0

(representing the start of the mouse travelling) and 1.0 (representing the end of the mouse travelling) and returns a

float value between 0.0 and 1.0.

4.2.1.5 Mouse Clicks

The click() function simulates a single, left-button mouse click at the mouse’s current position. A “click” is defined

as pushing the button down and then releasing it up. For example:

>>> pyautogui.click() # click the mouse

To combine a moveTo() call before the click, pass integers for the x and y keyword argument:

>>> pyautogui.click(x=100, y=200) # move to 100, 200, then click the left mouse button.

To specify a different mouse button to click, pass ’left’, ’middle’, or ’right’‘for the ‘‘button

keyword argument:

>>> pyautogui.click(button='right') # right-click the mouse

To do multiple clicks, pass an integer to the clicks keyword argument. Optionally, you can pass a float or integer to

the interval keyword argument to specify the amount of pause between the clicks in seconds. For example:

>>> pyautogui.click(clicks=2) # double-click the left mouse button

>>> pyautogui.click(clicks=2, interval=0.25) # double-click the left mouse button, but with a quarter second pause in between clicks

>>> pyautogui.click(button='right', clicks=3, interval=0.25) ## triple-click the right mouse button with a quarter second pause in between clicks

As a convenient shortcut, the doubleClick() function will perform a double click of the left mouse button. It also

has the optional x, y, interval, and button keyword arguments. For example:

>>> pyautogui.doubleClick() # perform a left-button double click

There is also a tripleClick() function with similar optional keyword arguments.

The rightClick() function has optional x and y keyword arguments.

4.2.1.6 The mouseDown() and mouseUp() Functions

Mouse clicks and drags are composed of both pressing the mouse button down and releasing it back up. If you want

to perform these actions separately, call the mouseDown() and mouseUp() functions. They have the same x, y,

and button. For example:

>>> pyautogui.mouseDown(); pyautogui.mouseUp() # does the same thing as a left-button mouse click

>>> pyautogui.mouseDown(button='right') # press the right button down

>>> pyautogui.mouseUp(button='right', x=100, y=200) # move the mouse to 100, 200, then release the right button up.

4.2.1.7 Mouse Scrolling

The mouse scroll wheel can be simulated by calling the scroll() function and passing an integer number of “clicks”

to scroll. The amount of scrolling in a “click” varies between platforms. Optionally, integers can be passed for the the

x and y keyword arguments to move the mouse cursor before performing the scroll. For example:

>>> pyautogui.scroll(10) # scroll up 10 "clicks"

>>> pyautogui.scroll(-10) # scroll down 10 "clicks"

>>> pyautogui.scroll(10, x=100, y=100) # move mouse cursor to 100, 200, then scroll up 10 "clicks"

On OS X and Linux platforms, PyAutoGUI can also perform horizontal scrolling by calling the hscroll() function. For

example:

>>> pyautogui.hscroll(10) # scroll right 10 "clicks"

>>> pyautogui.hscroll(-10) # scroll left 10 "clicks"

The scroll() function is a wrapper for vscroll(), which perform

4.3 Arduino Serial Plotter

This tool was used to plot the sensor values.

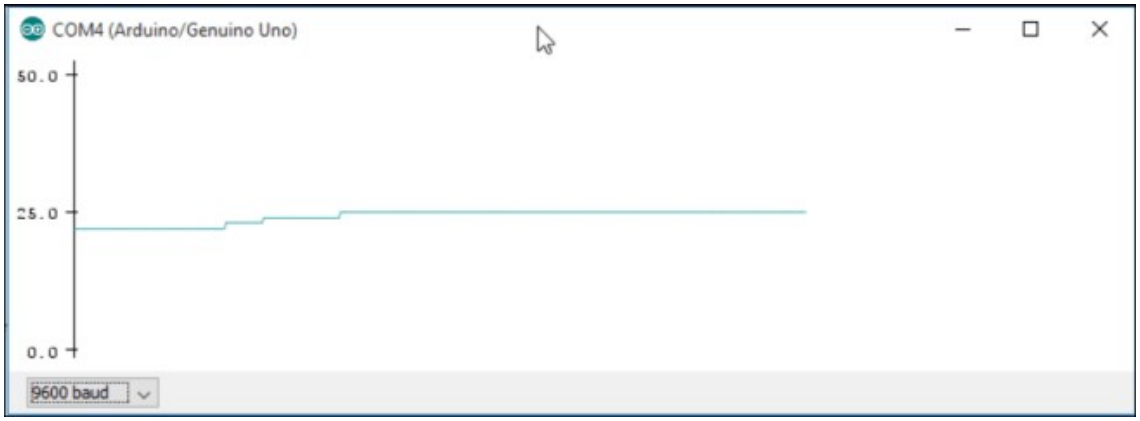
The latest release of the Arduino IDE 1.6.6 comes with a new tool called Serial Plotter. This tool gives you the ability to visualize your data in a plot that is updated in real time.

How the Arduino Serial Plotter Works

The Arduino Serial Plotter is a Tool that comes pre-installed with your Arduino IDE (version 1.6.6 and above) that takes incoming serial data and displays them in a plot.

The vertical Y axis adjusts as the value of your serial data increases or decreases. The X axis has 500 points and each tick of the axis is equal to an executed Serial.println() command.

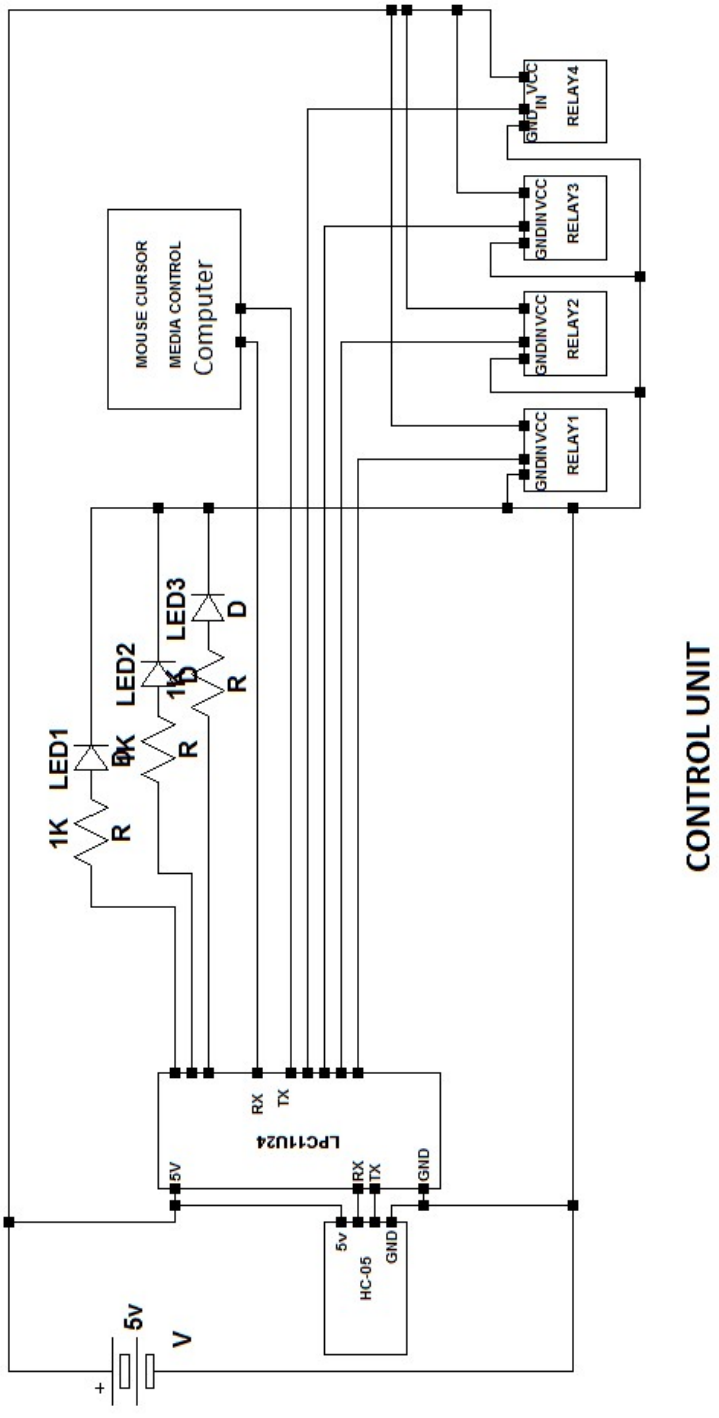
This means that the plot is updated every time you use the Serial.println() command with a new value.

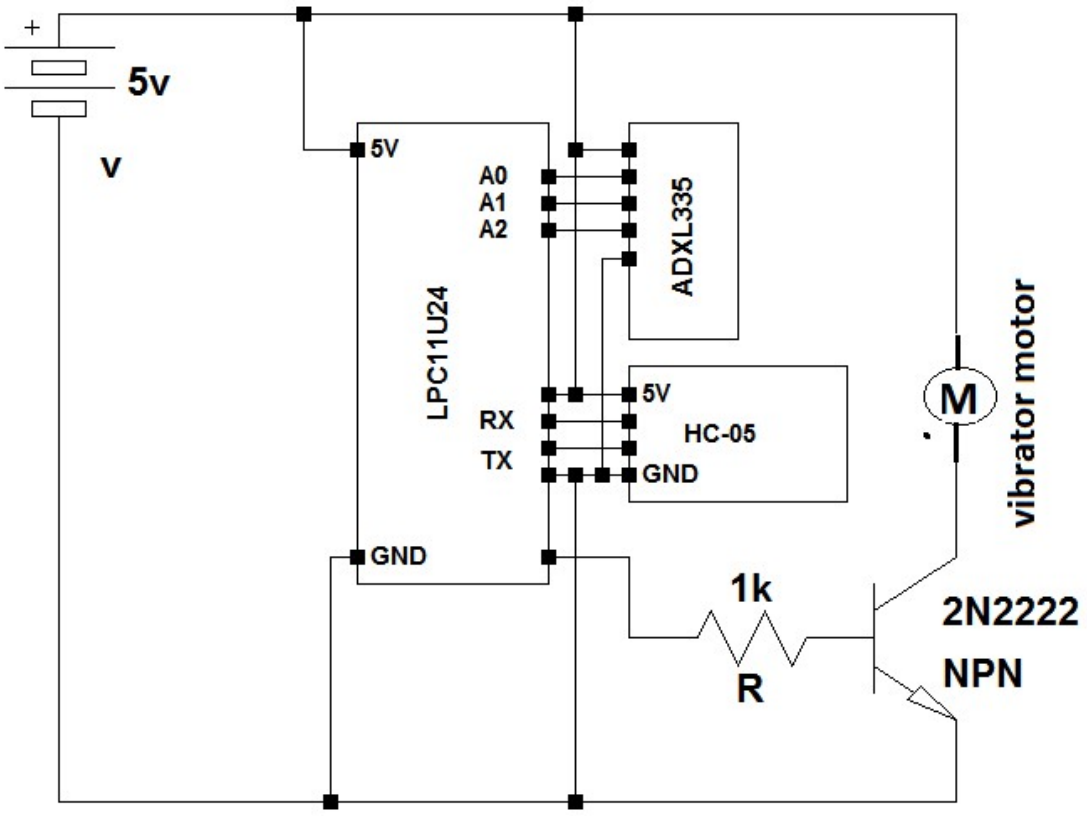


Apart from Arduino inputs this plotter works for any formatted serial numeric data. We supplied data from our mbed board to analyze over arduino serial plotter.

5.PROJECT IMPLEMENTATION

5.1 CIRCUIT DIAGRAM

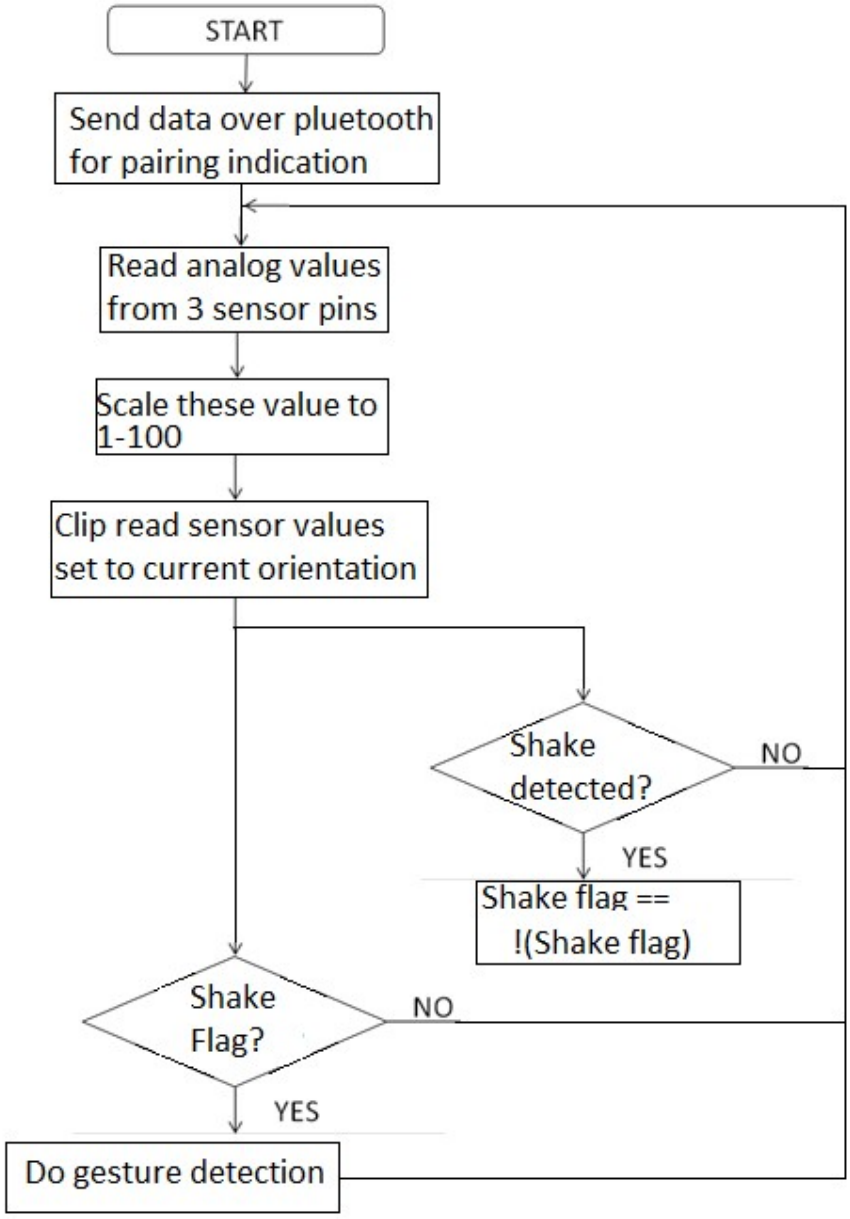



+
5v
v
5V
A0
A1
A2
ADXL335
LPC11U24
5V
RX
TX
GND
HC-05
GND
M
vibrator motor
1k
R
2N2222
NPN

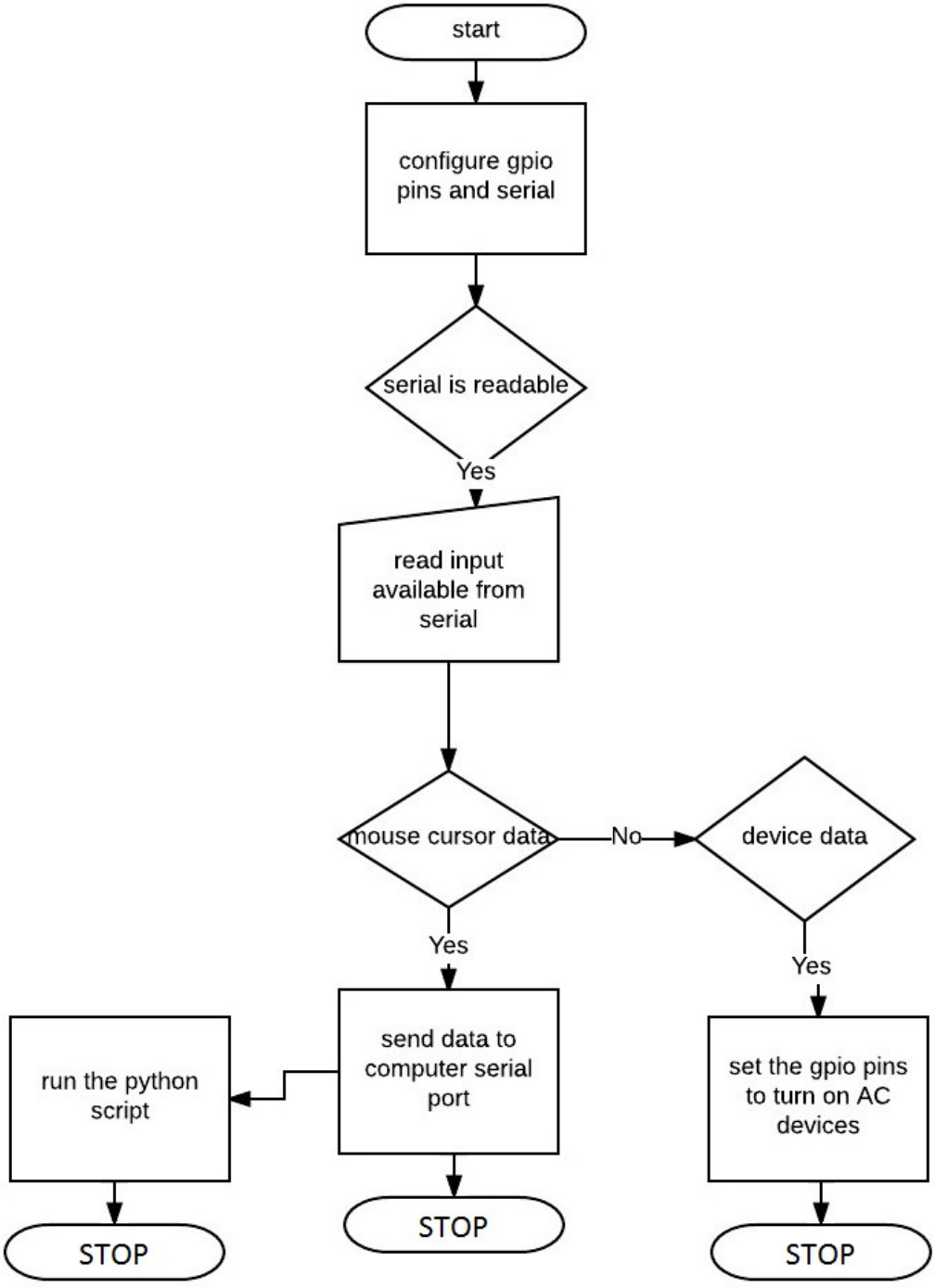
SENSOR UNIT

5.2 FLOW CHART

5.2.1 SENSOR UNIT



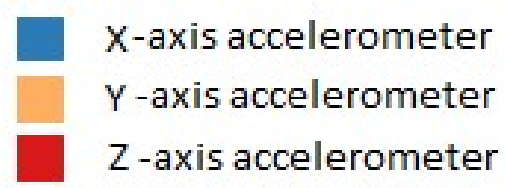
5.2.2 CONTROL UNIT



6. MODULE INTEGRATION AND CONTROL

6.1 Algorithm graphs and interpretation.

Accelerometer readings:



Interpretation

Moving
Left And Right

Moving
Forward
and
Backward

Moving
Up and down

Hand movement based

Device control system

Guide

Successful connection:

All LEDs on control unit blink twice and toggle on successful device connection.

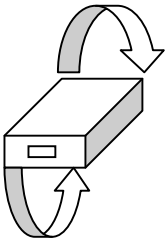
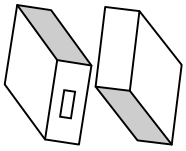
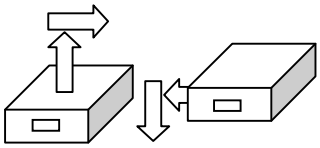
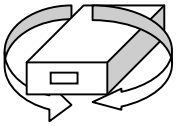
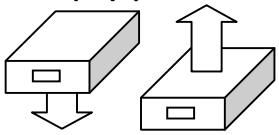
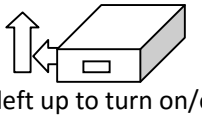
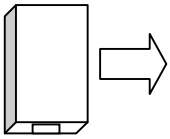
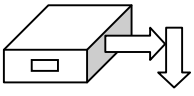
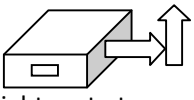
Turn on and off sensor unit:

Shaking twice along longer edge starts/stop gesture detection.

Switching between modes:

1. Pushing button once switches between media and mouse control mode.
2. Long pressing button switches to device control mode.

"In mouse mode the device continuously vibrates antonym to no vibrations in media mode."
"Device vibrates once on successful gesture detection in device mode."

 **Mouse Control**	 **Media Control**	 **AC Device control**
Hold device still parallel to floor, with power socket pointing towards you.	Hold device still parallel to floor, with power socket pointing towards you.	Hold device still parallel to floor, with power socket pointing towards you.
 Tilt up or down along shorter edge (pitch axis) to move mouse pointer **up** or **down**.	 Rest on the vertical edge to **play/pause**.	 Do up right or left down to open and close the **door**.
 Tilt left or right along longer edge (roll axis) to move mouse pointer **left** or **right**.	 Move piece up or down keeping parallel to the floor to increase/decrease **volume**.	 Do left up to turn on/off **Device-1**.
Tilt extreme left or extreme right for left and right **click**.	 Hold the device vertically pointing up and slide right for **full screen**.	 Do right-down to turn on/off **Device-2**.
		 Do right-up to turn on/off **Device-3**.

Features:

Pairing Indicator

Shake to turn on and off detection

Single button three control feature

8 bit word statement control

Movement detection: orientation and directional.

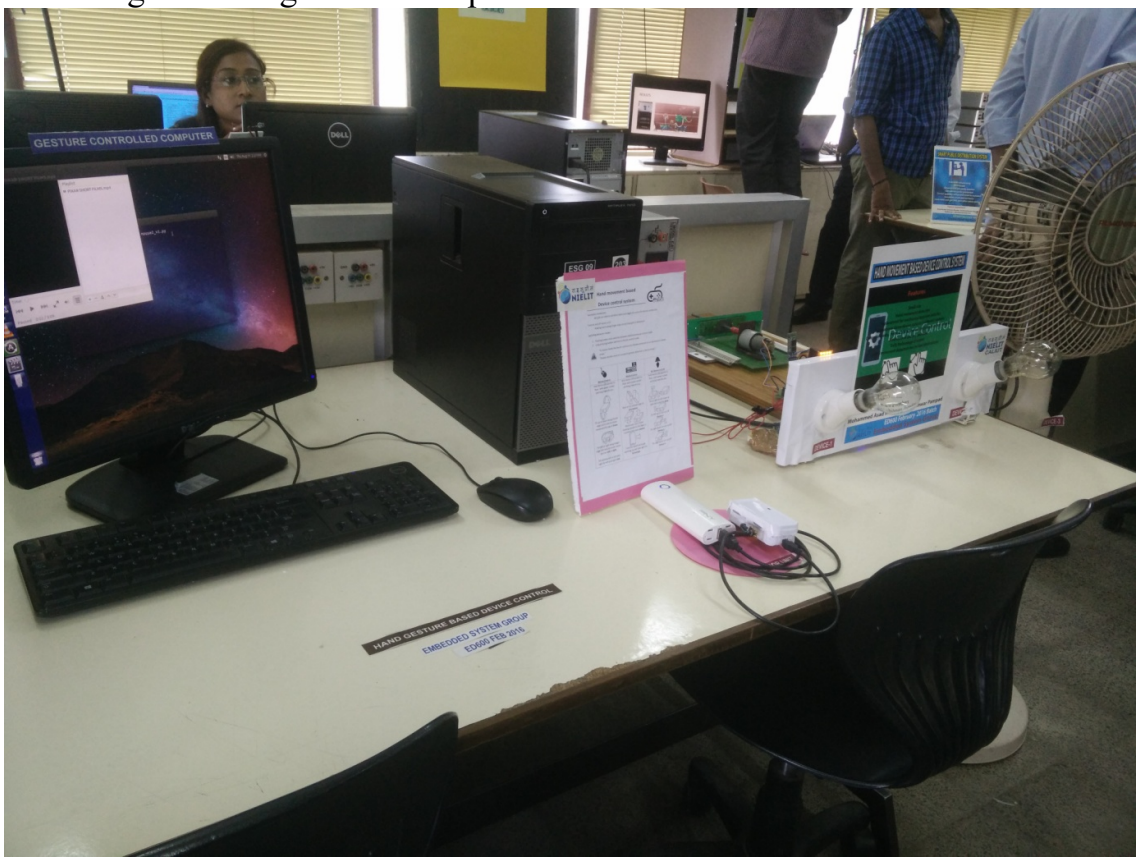
Vibrator feedback

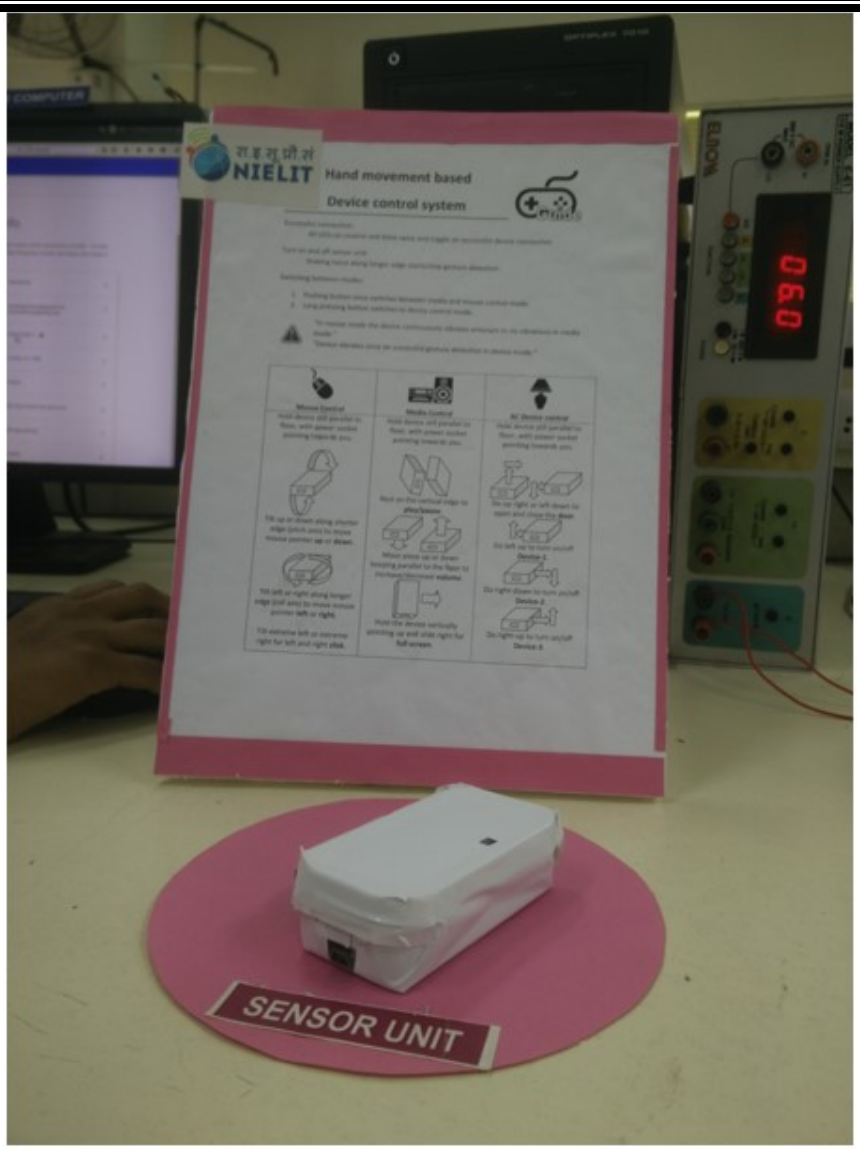
Serial data controls python based mouse and media.

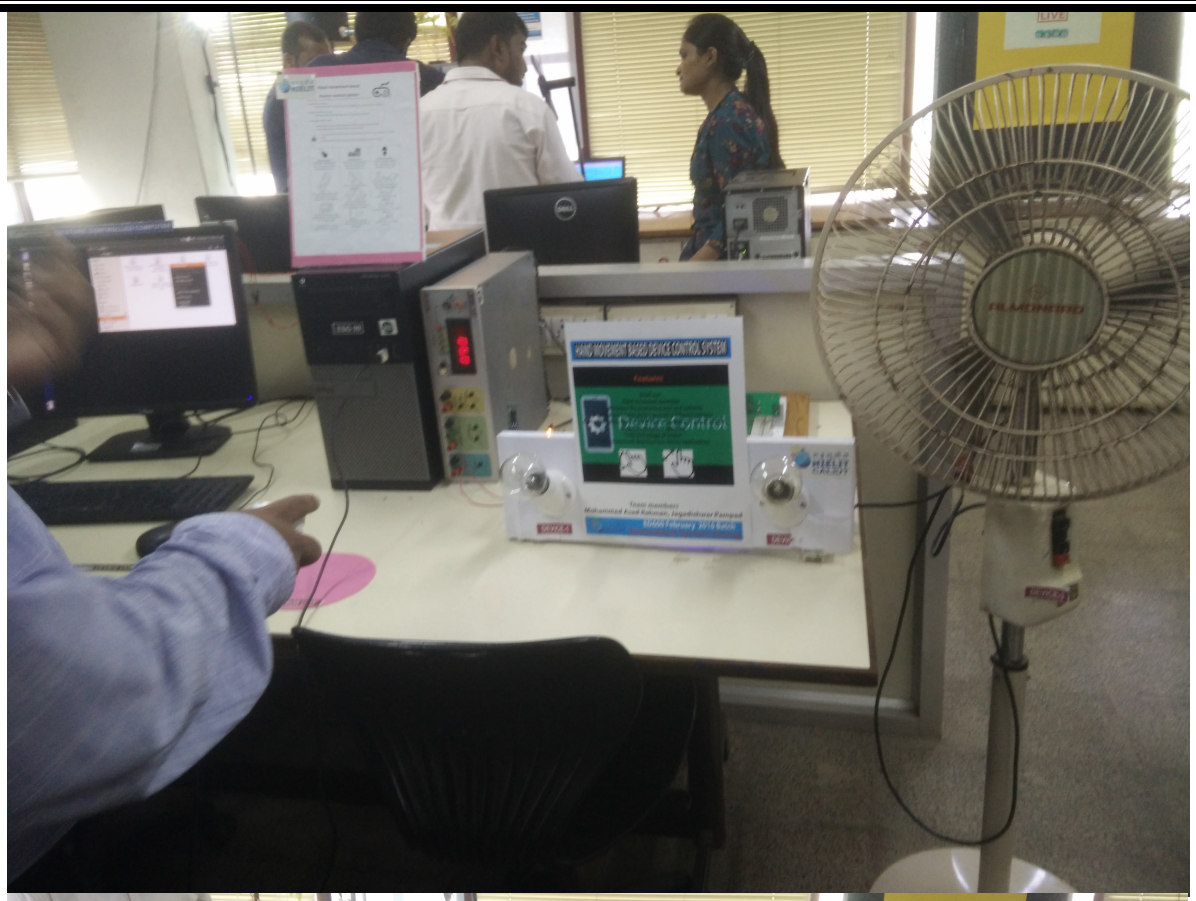
7. RESULTS

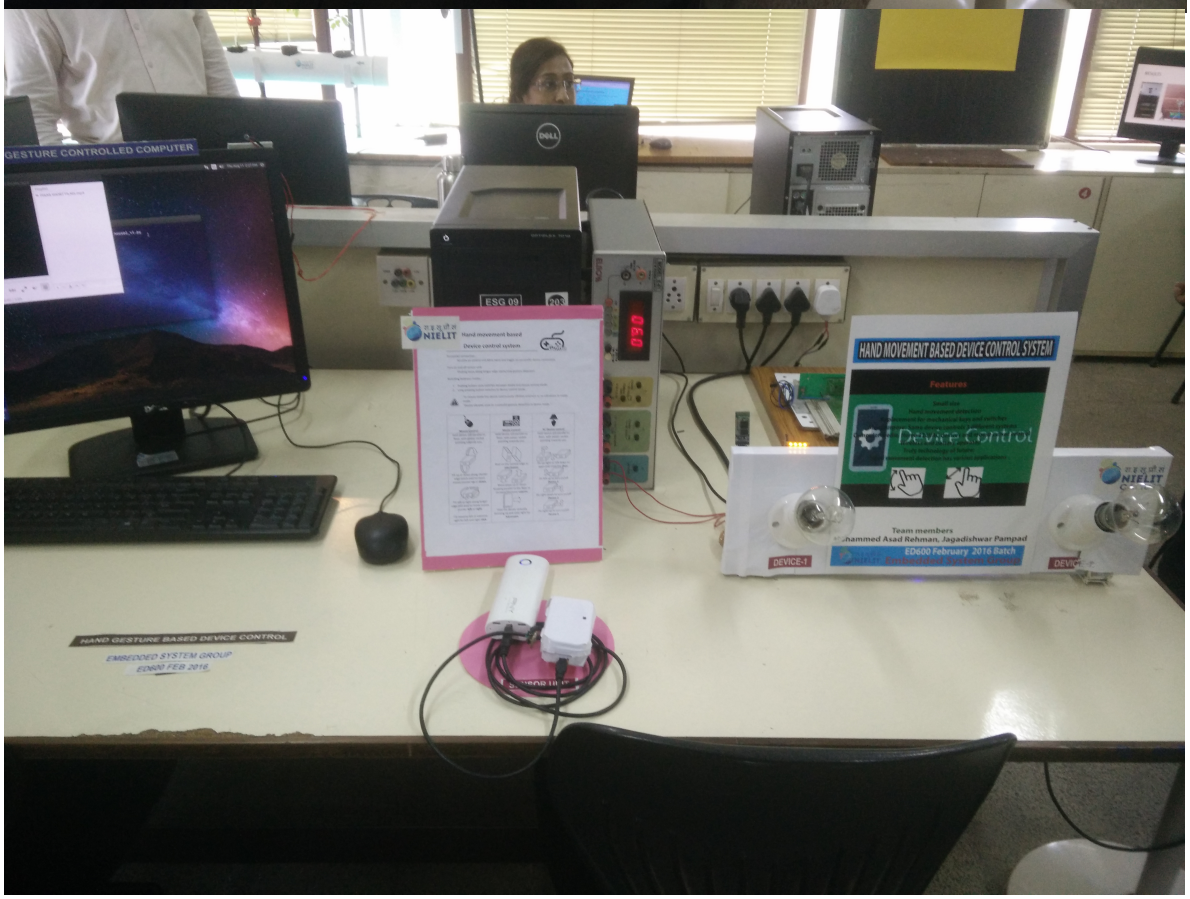
The project was implemented successfully. Three AC devices and various computer functionalities were controlled using hand gestures.

Following is an image of the setup:




NIELIT
Hand movement based
Device control system
SENSOR UNIT


HAND MOVEMENT BASED DEVICE CONTROL SYSTEM
Device Control


GESTURE CONTROLLED COMPUTER
ESG 09
NIELIT
Hand movement based
Device control system
HAND MOVEMENT BASED DEVICE CONTROL SYSTEM
Features
Device Control
Team members
ED600 February 2016 Batch
DEVICE-1
HAND GESTURE BASED DEVICE CONTROL
EMBEDDED SYSTEM GROUP
ED600 FEB 2016

8. CONCLUSION

The project has been successfully implemented.

Hand gestures are controlling following application on commands of the user:

1. Computer multimedia control
2. Mouse operation
3. Device operation
4. Home automation

The aim of this project is to operate real world devices with the help of pure hand movements hence providing innovative ways of control to its user which was successfully met.

All the motions of hand are sensed by an inertial mass unit accelerometer attached to user's hand. Determining speed and orientation information of user's hand and transmitting an appropriate control signal over wireless (Bluetooth) link to a control unit integrated with devices.

9. FUTURE SCOPE

1. Replacement for button remotes.
2. Wearable device.
3. Sensory fusion with gyroscope and magnetometer.
4. Measuring Cartesian coordinates.
5. Complete ‘media centre PC’ control solution.

10. REFERENCES

Books

Programming in C by Stephen G. Kochan.

Journal Papers

"Self-contained Position Tracking of Human Movement Using Small Inertial/Magnetic Sensor Modules"

- Xiaoping Yun, Eric R. Bachmann, Hyatt Moore IV, and James Calusdian
2007 IEEE International Conference on Robotics and Automation, Roma, Italy, 10-14 April 2007

Web links

Clap switch: http://www.electricaltechnology.org/2014/10/clap-switch-circuit-electronic-project.html

Automatic IR Tap Controller report by Ankit Jain
https://www.scribd.com/doc/19573938/Automatic-IR-Tap-Controller

Accelerometers: http://www.livescience.com/40102-accelerometers.html

http://4.bp.blogspot.com/-3NXUZRfifzA/T70rf3ktxWI/AAAAAAAAK9E/YsKWioh2HGQ/s1600/acelerometroqr4.jpg

http://www.rhydolabz.com/bluetooth-ble-c-130_132/hc05-bluetooth-module-masterslave-p-1169.html

https://store.arduino.cc/product/T010010

https://store.arduino.cc/product/T010010

A7800 SERIES POSITIVE-VOLTAGE REGULATORS 7805 datasheet

http://www.engineersgarage.com/electronic-components/transistor-bc547-datasheet

http://learn.mikroe.com/ebooks/componentsofelectronicdevices/chapter/introduction-to-resistors/

http://www.electronics-tutorials.ws/capacitor/cap_1.html

http://www.electrical4u.com/electric-lamp-types-of-electric-lamp/

http://www.edisontechcenter.org/Fan.html

https://docs.mbed.com/docs/mbed-os-handbook/en/5.1/dev_tools/online_comp/

ANNEXURES

ANNEXURE A : CODES

ANNEXURE B : DATA SHEETS