“RING-O”
MAJOR PROJECT

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

SUBMITTED BY

Akshay Baweja
00713202814

Divya Koneti
00913202814

2017-2018

GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY, NEW DELHI

GURU TEGH BAHADUR INSTITUTE OF TECHNOLOGY G-8 AREA, RAJOURI GARDEN, NEW DELHI-110064

CONTENTS

CERTIFICATE

Certified that **AKSHAY BAWEJA** (00713202814) has carried out the research work presented in this thesis entitled **"RING - O"** for the award of **Bachelor of Technology** from GGSIPU DWARKA, DELHI under my supervision. The thesis embodies results of original work, and studies as carried out by student himself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

External Examiner

_____

_____

Project Guide

Mukesh Sahu

HOD (ECE)

Date: 14 May 2018

CERTIFICATE

Certified that **DIVYA KONETI** (00913202814) has carried out the research work presented in this thesis entitled **"RING - O"** for the award of **Bachelor of Technology** from GGSIPU DWARKA, DELHI under my supervision. The thesis embodies results of original work, and studies as carried out by student herself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

External Examiner

_____

_____

Project Guide

Mukesh Sahu

HOD (ECE)

Date: 14 May 2018

ACKNOWLEDGEMENT

The successful completion of this project would not have been possible without the kind support and help of many individuals and organisations. We would like to extend our sincere thanks to all of them. We are highly indebted to our guide and Head of Department, ECE **Mr. Mukesh Sahu** for his guidance and constant supervision as well as for providing necessary information regarding the project.

We would also like to express our gratitude towards our parents and staff of **Guru Tegh Bahadur Institute Of Technology** for their kind co-operation and encouragement which helped us in the completion of this project. Our thanks and appreciation also goes to our fellow colleagues for their help in developing the project and people who have willingly helped us out with their abilities.

Akshay Baweja
00713202814/ECE-1/2014
akshaybaweja.1996@gmail.com

Divya Koneti
00913202814/ECE-1/2014
divyakoneti0001@gmail.com

DECLARATION

We hereby declare that the work which is being presented in this Major Project entitled, "**RING - O**" submitted to **GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY, NEW DELHI** in the partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** in **ELECTRONICS & COMMUNICATION ENGINEERING**, is an authentic record of our own work carried out from January, 2018 to May, 2018 under the supervision of Mr. Mukesh Sahu.

The matter embodied in this project report has not been submitted by me for the award of any other.

Place: New Delhi, Delhi
Date: May 10, 2018

Akshay Baweja
(00713202814/ECE1/2014)

Divya Koneti
(00913202814/ECE1/2014)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Mr. Mukesh Sahu
Project Guide

ABSTRACT

Machine learning is one of the latest technologies which is widely used in industries worldwide to reduce the human efforts. This teaches the device to work on its own without anyone else's help. The algorithm is designed in such a way that the device learns on its own. The more precise the algorithm is, the more efficient the device works. Ring-O is a three dimensional gesture controlled device which enables us to type anywhere and the text appears on the screen. Moreover, it allows you to perform various day-to-day gestures to ease your work, some of these include music player control, IoT enabled home appliance control, etc.

Ring-O has been trained a number of times for accurate results. Initially it yielded 62% efficiency but later on it reached the 84% mark. The final Ring-O model works perfectly without any time lag or errors. It is user friendly and because of its compact size, it can be carried anywhere without any problem. Ring-O is a combination of both keyboard and mouse. It is a smart keyboard which doesn't require much of efforts to type or to perform any other task.With the evolution of technology and the minimization in embedded technology - Ring-O is one of the best examples of smart devices and is going to be the largely used device in the coming days.

The system works on ATMega328 and the MPU6050 which is a combination of Gyroscope and accelerometer sensors. MPU6050 gives 84% prediction accuracy that made it possible to design Ring-O in a compact package. Currently, it is working as a python app on computer but can be integrated with the Android application and can be used as a smart keypad and to swipe between apps, control them and could be used for gaming as well.

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

INTRODUCTION

Ring-O is a three dimensional, gesture-based input tool that lets you type in the air and display the content on a screen. The whole device is designed in the form of a ring which connects to the PC via Bluetooth. The ring is packed with micro-controller, IMUs like accelerator and gyroscope, battery along with protection circuit and a bluetooth module. These internal mechanics allow Ring-O to work more like a smart keyboard than a traditional one. Because it relies on sensors, Ring-O doesn't need complex gesture recognition algorithms, and can therefore understand movements more quickly and reliably. Furthermore, since Ring-O is designed to be worn on the hand and carried around, it allows you to interface with a wider range of objects. Also the second mode, that is, the gesture mode allows you to perform gestures and perform various tasks to be performed using it. These tasks include music control such as volume up/down, play/pause, skip song and many more, Home Appliance control such as turning IoT enabled devices on or off, etc. Also, you can perform various other gestures such as brightness control of your laptop, presentation control, etc.

While physical gestures are key to the controller, the Ring-O also understands that everyone moves differently. Users train the ring according to the gestures that are most natural to their own style and workflow, and assign those movements to common actions relevant to whatever they're working on.

Ring-O is a better way to interact. This a device that got back to the basics of how we use computers. We're used to old tools like the keyboard and mouse, but they're starting to make less sense as our devices get smaller and more embedded in our lives. And keyboard being the basic input device to a computer and widely used worldwide needs a new design with the growing technology and the shrinking size of the computers as small as a palm. The Ring-O is by far the lightest typing device without any interface constraints. Its a smartly designed gesture controlled keyboard. Ring-O lets you work with your hands in a more intuitive way.

Ring-O is extremely versatile. We can program custom gestures into actions on your devices. Just make a movement with your hand, then tie it to any action you want.

Ring-O is based on machine learning algorithms that help it understand the movements that are being performed in a 3D Space. Each and every action of respective character is being trained to the device to get maximum efficiency and then according to designed algorithm the machine adapts the ongoing process to function further for the similar actions. Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

With the rapid growth and advancement of technology the keyboard and mouse will not be around forever, and the Ring-O will make that happen sooner rather than later. It will not only help in accelerating the work process but will also be helpful in various ways.

The project has been designed using various technologies like Machine Learning, PCB designing, 3D printing. This project has been designed by keeping in mind the upcoming technology and the growing needs of people to make tasks easy and smooth.

CHAPTER-2

PROJECT WORK

CHAPTER-2.1

MPU6050

INTRODUCTION AND WORKING

MPU6050

The MPU-6050 parts are the world's first MotionTracking devices designed for the low power, low cost, and high-performance requirements of smartphones, tablets and wearable sensors.

The MPU-6050 incorporates InvenSense's MotionFusion and run-time calibration firmware that enables manufacturers to eliminate the costly and complex selection, qualification, and system level integration of discrete devices in motion-enabled products, guaranteeing that sensor fusion algorithms and calibration procedures deliver optimal performance for consumers.

The MPU-6050 devices combine a 3-axis gyroscope and a 3-axis accelerometer on the same silicon die, together with an onboard Digital Motion Processor (DMP), which processes complex 6-axis MotionFusion algorithms. The device can access external magnetometers or other sensors through an auxiliary master I²C bus, allowing the devices to gather a full set of sensor data without intervention from the system processor. The devices are offered in a 4 mm x 4 mm x 0.9 mm QFN package.

Figure: 2.1.1

Block Diagram of MPU6050

The accelerometer and gyro values are called the "raw" values. This is just as with other accelerometer and gyro sensors. A more sophisticated application is using the DMP to retrieve specific computed values from the sensor. Reading the raw values for the accelerometer and gyro is easy. The sleep mode has to be disabled, and then the registers for the accelerometer and gyro can be read.

But the sensor also contains a 1024 byte FIFO buffer. The sensor values can be programmed to be placed in the FIFO buffer. And the buffer can be read.

The FIFO buffer is used together with the interrupt signal. If the MPU-6050 places data in the FIFO buffer, it signals the controller with the interrupt signal so the controller knows that there is data in the FIFO buffer waiting to be read. A little more complicated is the ability to control a second I2C- device.The MPU-6050 always acts as a slave to the controller with the SDA and SCL.

But beside the normal I2C-bus, it has it's own I2C controller to be a master on a second (sub)-I2C-bus. It uses the pins AUX_DA and AUX_CL for that second (sub)-I2C-bus. It can control, for example, a magnetometer. Things get really complex with the “DMP”. The sensor has a "Digital Motion Processor" (DMP), also called a "Digital Motion Processing Unit". This DMP can be programmed with firmware and is able to do complex calculations with For this DMP, InvenSense has a discouragement policy, by not supplying enough information how to program the DMP. However, some have used reverse engineering to capture firmware. The DMP (Digital Motion Processor) can do fast calculations directly on the chip.

This reduces the load for the micro-controller. The DMP is even able to do calculations with the sensor values of another chip, for example a magnetometer connected to the second (sub)-I2C-bus.

The pin "AD0" selects between I2C address 0x68 and 0x69. That makes it possible to have two of these sensors in a project. Most breakout boards have a pull-up or pulldown resistor to make AD0 default low or high. Connect AD0 to GND or 3.3V for the other I2C address.

The InvenSense MotionApps Platform that comes with the MPU-6050 abstracts motion-based complexities, offloads sensor management from the operating system, and provides a structured set of APIs for application development.

For precision tracking of both fast and slow motions, the parts feature a user-programmable gyro full-scale range of ±250, ±500, ±1000, and ±2000 °/sec (dps), and a user-programmable accelerometer full-scale range of ±2g, ±4g, ±8g, and ±16g.

Additional features include an embedded temperature sensor and an on-chip oscillator with ±1% variation over the operating temperature range.

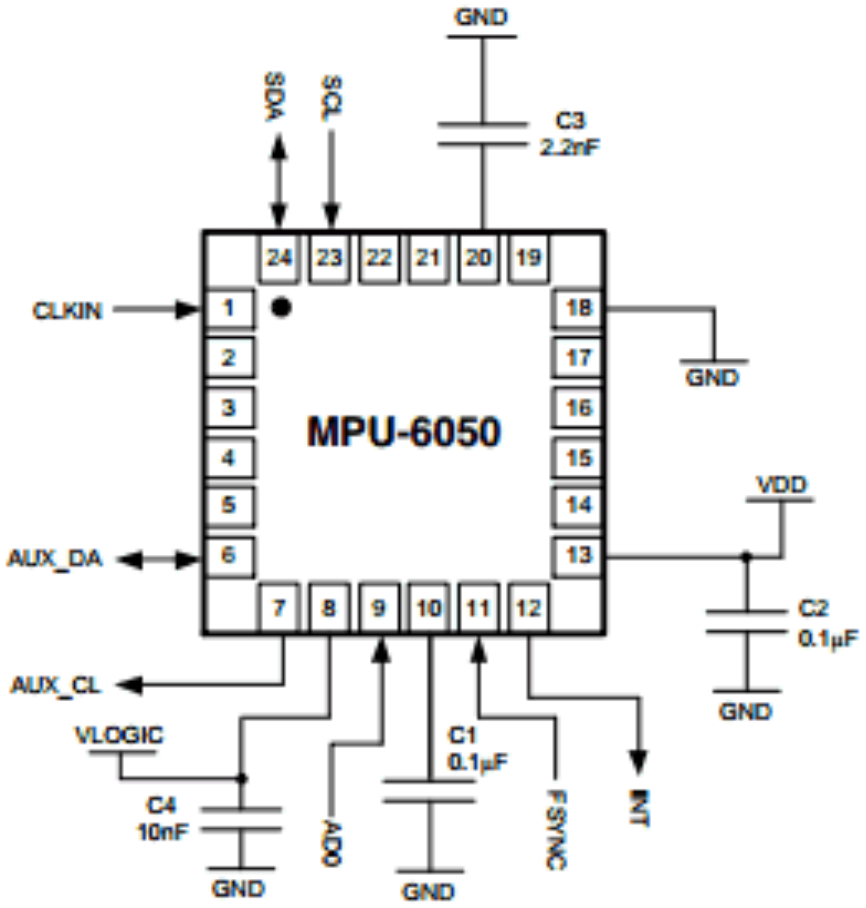


Figure:2.1.2

Pin Diagram Of MPU6050

CHAPTER-2.2

3D PRINTING TECHNOLOGY :

IMPLEMENTATION IN PROJECT

3D Printing technology:

3D printing is a form of additive manufacturing technology where a three dimensional object is created by laying down successive layers of material. It is also known as rapid prototyping, is a mechanised method whereby 3D objects are quickly made on a reasonably sized machine connected to a computer containing blueprints for the object. The 3D printing concept of custom manufacturing is exciting to nearly everyone. This revolutionary method for creating 3D models with the use of inkjet technology saves time and cost by eliminating the need to design; print and glue together separate model parts. Now, you can create a complete model in a single process using 3D printing. The basic principles include materials cartridges, flexibility of output, and translation of code into a visible pattern.

3D Printers are machines that produce physical 3D models from digital data by printing layer by layer. It can make physical models of objects either designed with a CAD program or scanned with a 3D Scanner. It is used in a variety of industries including jewellery, footwear, industrial design, architecture, engineering and construction, automotive, aerospace, dental and medical industries, education and consumer products.

History of 3d Printing

The technology for printing physical 3D objects from digital data was first developed by Charles Hull in 1984. He named the technique as Stereo lithography and obtained a patent for the technique in 1986. While Stereo lithography systems had become popular by the end of 1980s, other similar technologies such as Fused Deposition Modelling (FDM) and Selective Laser Sintering (SLS) were introduced. In 1993, Massachusetts Institute of Technology (MIT) patented another technology, named "3 Dimensional Printing techniques", which is similar to the inkjet technology used in 2D Printers. In 1996, three major products, "Genisys" from Stratasys, "Actua 2100" from 3D Systems and "Z402" from Z Corporation, were introduced.

In 2005, Z Corp. launched a breakthrough product, named Spectrum Z510, which was the first high definition colour 3D Printer in the market. Another breakthrough in 3D Printing occurred in 2006 with the initiation of an open source project, named Rep-rap, which was aimed at developing a self-replicating 3D printer.

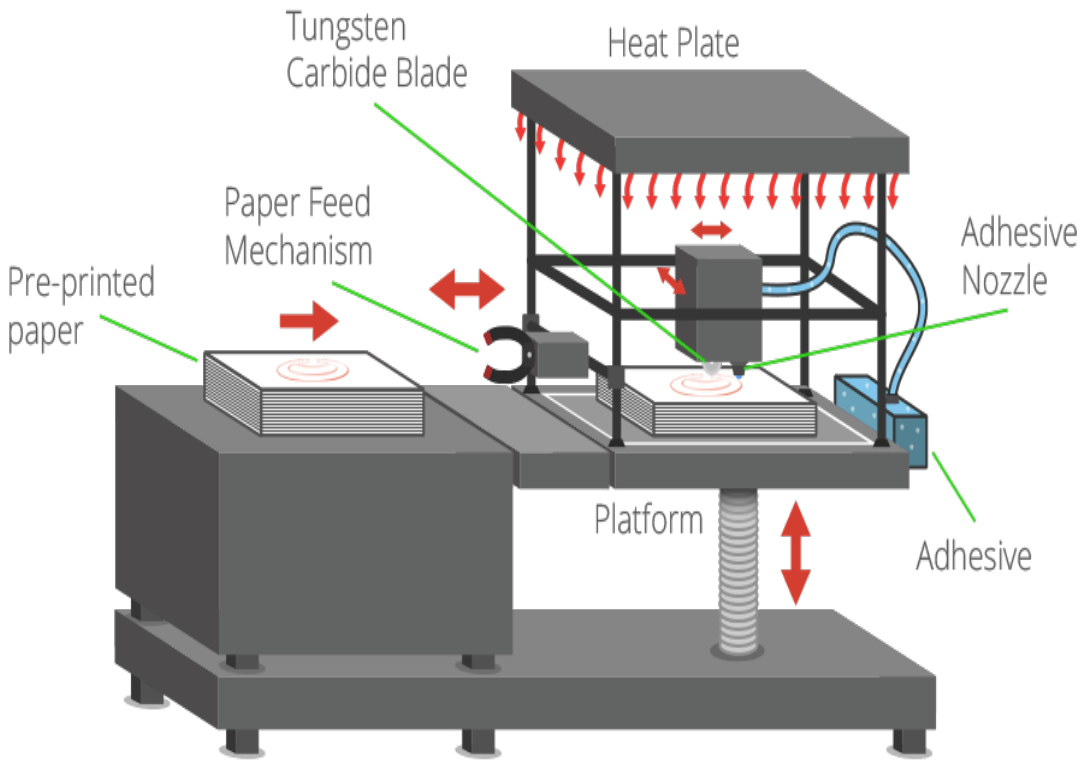


Figure:2.2.1

Manufacturing a Model With The 3D Printer

Current 3D Printing Technologies

- Stereo lithography - Stereo lithographic 3D printers (known as SLAs or stereo lithography apparatus) position a perforated platform just below the surface of a vat of liquid photo curable polymer. A UV laser beam then traces the first slice of an object on the surface of this liquid, causing a very thin layer of

photopolymer to harden. The perforated platform is then lowered very slightly and another slice is traced out and hardened by the laser. Another slice is then created, and then another, until a complete object has been printed and can be removed from the vat of photopolymer, drained of excess liquid, and cured.

- Fused deposition modelling - Here a hot thermoplastic is extruded from a temperature-controlled print head to produce fairly robust objects to a high degree of accuracy.

- Selective laser sintering (SLS) - This builds objects by using a laser to selectively fuse together successive layers of a cocktail of powdered wax, ceramic, metal, nylon or one of a range of other materials.

- Multi-jet modelling (MJM)- This again builds up objects from successive layers of powder, with an inkjet-like print head used to spray on a binder solution that glues only the required granules together.

- The VFlash printer, manufactured by Canon, is low-cost 3D printer. It's known to build layers with a light-curable film. Unlike other printers, the VFlash builds its parts from the top down.

- Desktop Factory is a startup launched by the IdeaLab incubator in Pasadena, California.

- Fab@home, an experimental project based at Cornell University, uses a syringe to deposit material in a manner similar to FDM. The inexpensive syringe makes it easy to experiment with different materials from glues to cake frosting.

- The Nano-factory 3D printing technologies are introduced that are related to the nanotechnologies. .

3D Printing Capabilities:

As anticipated, this modern technology has smoothed the path for numerous new possibilities in various fields. The list below details the advantages of 3D printing in certain fields.

1.Product formation is currently the main use of 3D printing technology. These machines allow designers and engineers to test out ideas for dimensional products cheaply before committing to expensive tooling and manufacturing processes.

2.In Medical Field, Surgeons are using 3d printing machines to print body parts for reference before complex surgeries. Other machines are used to construct bone grafts for patients who have suffered traumatic injuries. Looking further in the future, research is underway as scientists are working on creating replacement organs.

3. Architects need to create mockups of their designs. 3D printing allows them to come up with these mockups in a short period of time and with a higher degree of accuracy.

4.3D printing allows artists to create objects that would be incredibly difficult, costly, or time intensive using traditional processes.

3D Saves Time and Cost: Creating complete models in a single process using 3D printing has great benefits. This innovative technology has been proven to save companies time, manpower and money. Companies providing 3D printing solutions have brought to life an efficient and competent technological product.

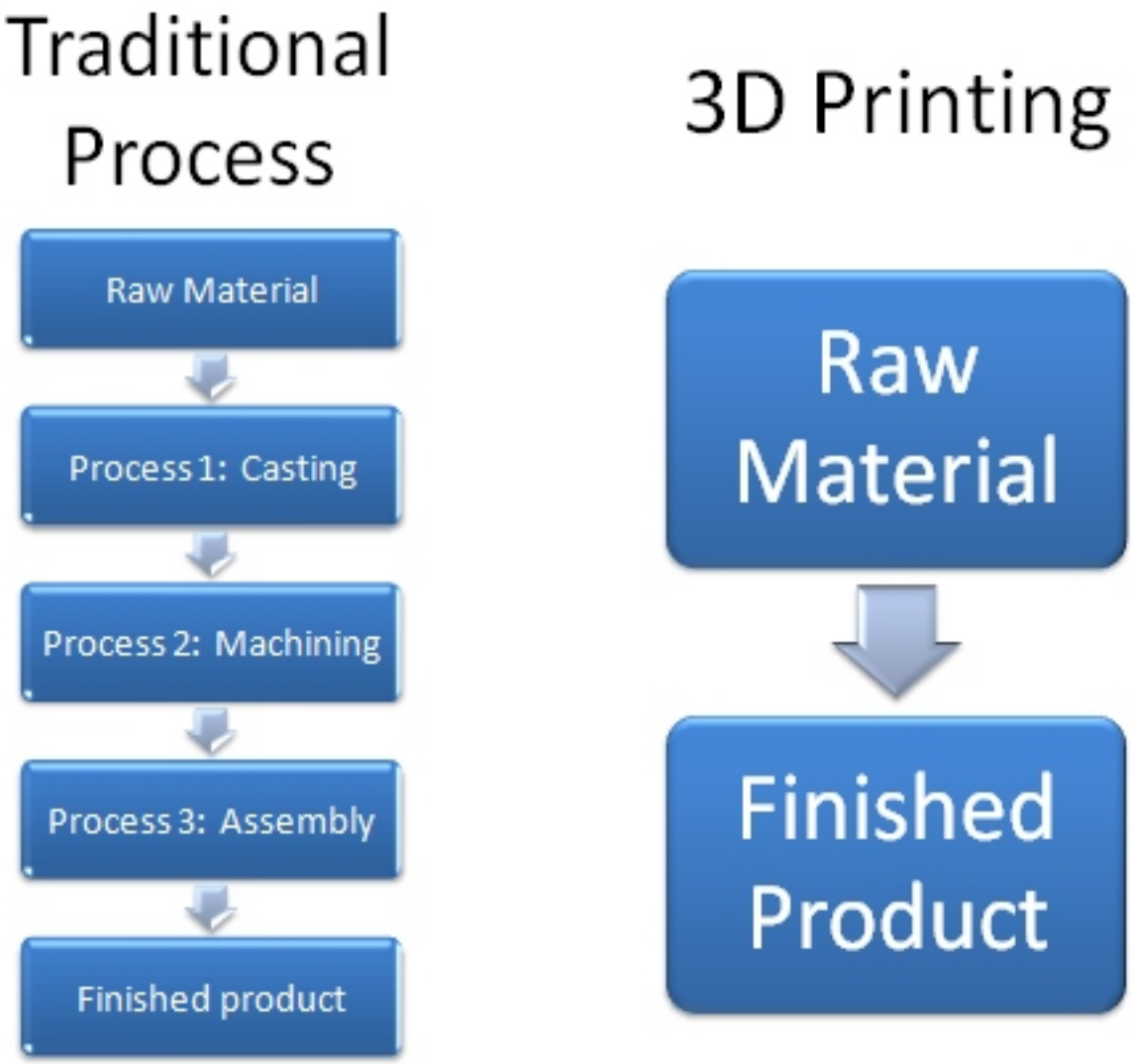


Figure:2.2.2

Flowchart of a 3D Printing Process

3D Printing in Ring-O

Ring-O is manufactured using 3D Printing technology with PLA being the print material. Ring-O is designed in two parts to house the PCBs and battery inside.

Perhaps the most important part of 3D printing is using the right material for your print job. When considering what building material to use, there are numerous factors to consider, including strength, flexibility, accuracy, and special conditions the material may require in order to print properly and accurately. ABS and PLA are the

two types of materials 3D printers currently use, but the differences between the two aren't immediately apparent. The raises the question: which one is best suited for your 3D printing needs?

Both ABS and PLA are thermoplastics. Thermoplastics become malleable when superheated, thus allowing you to mould and sculpt them into different shapes prior to cooling. Moreover, you can repeat the process without affecting the integrity of the material. While both are used for making objects in 3D printing via similar processes, ABS and PLA differ in some key ways, and therefore some printers will only utilize ABS or PLA — or both, depending on the machine at hand.

PLA

PLA, or Poly Lactic Acid, is made from organic material — specifically cornstarch and sugarcane. This makes the material both easier and safer to use, while giving it a smoother and shinier appearance that's more aesthetically pleasing. The thermoplastic is also more pleasant on the nose, as the sugar-based material smells slightly sweet when heated opposed to the harsh smell often associated with ABS. However, while PLA might seem like a better overall choice at first glance, it features a far lower melting point than ABS. This means that using printed parts for mechanical operations, or even storing them in high-temperature locations, can result in the part warping, cracking, or melting. The material is also weaker than ABS, though, it can achieve a superior level of print detail and is less prone to errors while printing.

ABS

ABS, short for Acrylonitrile Butadiene Styrene, is an oil-based plastic. It is a strong, sturdy material that businesses widely use for constructing things such as plastic car parts, musical instruments, and the ever-popular Lego building blocks. ABS has a high melting point, and can experience warping if cooled while printing. Because of this, ABS objects must be printed on a heated surface, which is something many at-home printers do not have. ABS also requires ventilation when in use, as the fumes can be unpleasant. The aforementioned factors make ABS printing difficult for hobbyist printers, though, it's the preferred material for professional applications.

Conclusion

All things considered, there are some similarities between the two. Each requires a dry location for storage given the materials are prone to melting and warping, and furthermore, each is susceptible to moisture. Both ABS and PLA also smell while printing, as heating the thermoplastic gives off fumes. That said, it's the print temperature that primarily affects intensity of said fumes — not so much the material itself. Yes, ABS will smell like hot plastic and require ventilation while PLA will smell mildly sweet, but the strength of these smells is going to be dependent on your printer.

However, there are also major differences between the two thermoplastics. ABS is going to give your projects better structural integrity and will be more suited to mechanical use given the material can better withstand the elements, but it will also require specific types of printers and printing surfaces. On the flip side, PLA will give you more precise prints and better aesthetic quality, as well as more flexibility with printing conditions if you can do without the strength and resilience of ABS. In simple terms, PLA is for hobbyist printers while ABS caters to those looking to produce commercial-grade parts that need to endure more rigorous wear and tear.

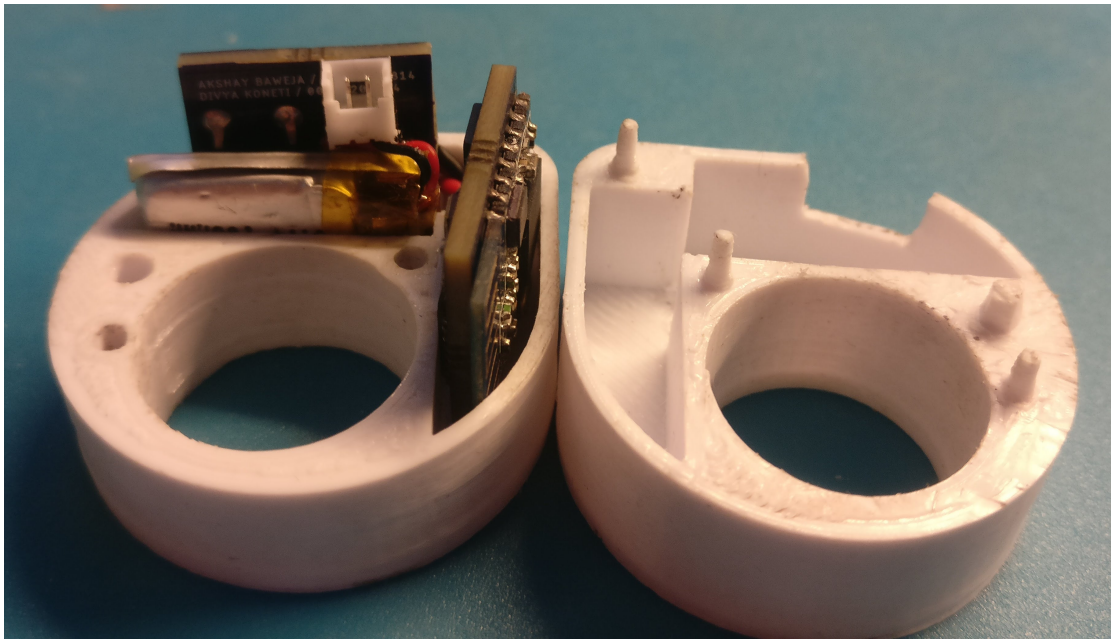


Figure 2.2.3
3D printed Ring

3D PRINTER USED FOR PRINTING

The 3D printer used in project is **FlashForge Creator Pro** for high level of precision. Upgraded from the outside in, Creator Pro is a very reliable 3D printer with excellent precision and professional quality. The sturdy metal frame increases stability of the printer's moving parts. Metal platform support and 10mm Z-axis guide rods allow for precise movement of the z-axis. The build plate is made from 6.3mm thickness alloy of Aluminium —the same grade used in the aerospace industry, excellent heat distribution and never deformed. The versatile chamber provides different printing environment for different materials. As a result, precise and top-tier print quality shows up.

The 2016 Creator Pro comes with stylish and user-friendly design. All the metal and plastic parts go together with each other so well because they are moulded. Thoughtful design details include unibody top cover, external handles, 180°opening front panel, plastic levelling knobs, 45°viewing control interface and full-range power supply are added to its friendly user experience. Creator Pro features a metal frame instead of wooden one of original Creator. It is optimised for the printing process by adding much needed stiffness to reduce vibration. The metal support and 10mm Z-axis guide rods work together to increase precision of vertical movement. The Creator Pro build plate is made from the same grade of aluminium used in the aerospace industry, with 6.3mm thickness. This high level aluminium can always stay flat, even after continued exposure to high heat. And yes, it can be heated to deliver desired temperature for different materials. Then reliable and high quality prints show up.

Creator Pro features a closable door and removable top cover to keep out dust and foreign particles. This fully enclosed chamber also eliminates temperature interference from the exterior environment and stabilises the printing temperature for more successful and accurate ABS prints, less warps. Open the door, remove the top cover and let air flows in. The chamber is ventilated to cool down extruded PLA quickly. This yields out smoother curves on finished prints, less hung materials.

From outside in, all major parts such as motors and power supply are from world's top suppliers, and all metal and plastic parts are moulded. These bring forth a consistent and stable Creator Pro. We set rigorous standards for assembly lines, where each machine is tested over 48hrs for top quality.

Creator Pro is a straightforward 3D printer based on open source technology. It allows for flexible upgrades and various modifications. Meanwhile, you are able to choose your favourite software such as FlashPrint, Simplify3D, Cura and more.

For beginners, the simple and intuitive interface lets you convert your design to a 3D printed model with a few clicks. It has a list of presets for build quality, so just select what build quality you want and FlashPrint takes care of the rest. And for experts seeking more control over the software, it has a wide range of parameters can be changed manually as well, from temperature to printing speed and even the first layer thickness and height where to pause printing. Once sliced, you can get layer-by-layer visualisation and time and material estimates.

Featured with cut and split function, FlashPrint allows you to divide your model into several separate parts when it is too big as for one print. Limited build volume can print bigger models.

Creator Pro comes with an proprietary dual-extruder structure. It allows you to print the main part with ABS filament and the support with dissolvable filament which can be melt in limonene. And this gives you solid infill for delicate overhangs and objects with internal moving parts. Also it can be tuned to print with a wide range of experimental materials such as Flex, T-glass and composite materials such as woodFill, copperFill, and brassFill.

Figure:2.2.4
FlashForge Creator Pro 3D printer

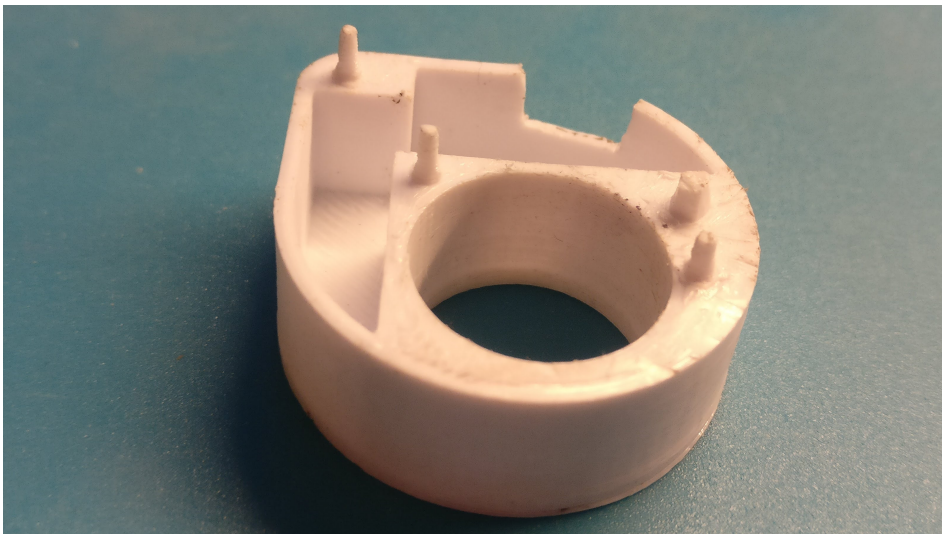


Figure:2.2.5
3D Printed Model of ring

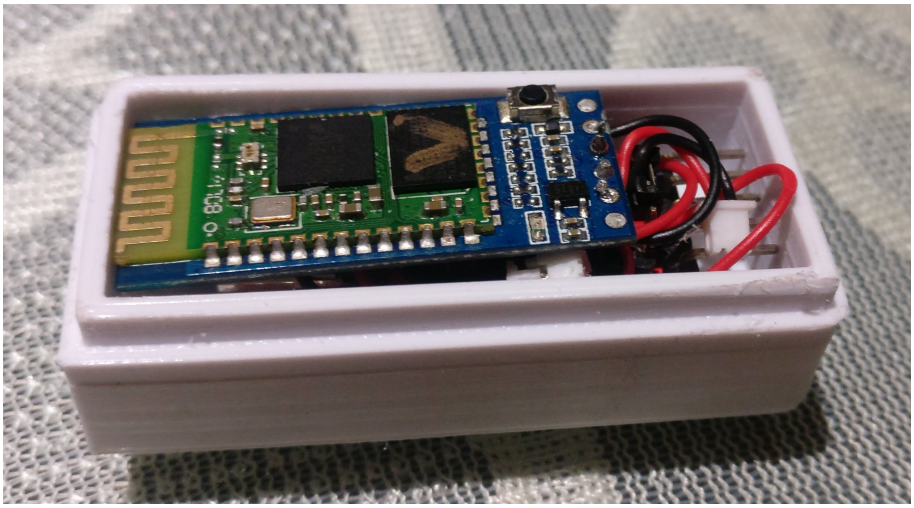


Figure:2.2.6

3D Model of Receiver with components inside

CHAPTER-2.3

SOFTWARE USED TO DESIGN MODEL

FUSION 360

FUSION 360

Figure:2.3.1
Autodesk Fusion 360 Logo

Fusion 360 is a cloud-based CAD/CAM/CAE tool for collaborative product development. Fusion 360 combines fast and easy organic modelling with precise solid modelling, allowing you to make your designs manufacturable. Start by communicating your designs with renderings and drawings the ease of collaboration using Fusion 360 increases team productivity. As a design comes together it can be optimised and validated using the Simulation features of Fusion 360. When your design is done Fusion 360 isn't, it provides capabilities to take you into manufacturing. The CAM capabilities allow quick generation of tool-paths for CNC machines, or you can send the design to a 3D printer for rapid prototyping.

Your data is kept safe and secure on the cloud, with unlimited storage and access. Fusion 360 is compatible with Mac and PCs, with unlimited installs, and unlimited team member invites to your Fusion 360 projects.

User interface overview

Application bar: Access the Data Panel (that allows quick access to your design files), file operations, save, undo and redo.

Profile and help: In profile you can control your profile and account settings, or use the help menu to continue your learning or get help in troubleshooting.

Toolbar: Use the Toolbar to select the workspace you want to work in, and the tool you want to use in the workspace selected.

ViewCube: Use the ViewCube to orbit your design or view the design from standard view positions.

Browser: The browser lists objects in your design (everything from planes and sketches to parts and assemblies). Use the browser to make changes to objects and control visibility of objects.

Canvas and marking menu: Left click to select objects in the canvas. Right-click to access the marking menu. The marking menu contains frequently used commands in the wheel and all commands in the overflow menu.

Timeline: The timeline lists operations performed on your design. Right-click operations in the timeline to make changes. Drag operations to change the order they are calculated.

Navigation bar and display settings: The navigation bar contains commands used to zoom, pan, and orbit your design. The display settings control the appearance of the interface and how designs are displayed in canvas.

Workspaces

What are workspaces?

Fusion 360 has more capabilities than typical CAD software. To simplify your experience with the interface, capabilities are grouped into various workspaces. These workspaces organise tools and commands according to particular design objectives. Each workspace has a unique toolbar across the top. However, certain menus, and

their commands, are repeated in multiple workspaces. Currently, the following workspaces are available in Fusion 360.

MODEL

The MODEL workspace allows you to create and edit solid 3D geometry. This workspace is most like a traditional 3D CAD environment. In addition to standard modelling features, like extrude or revolve, you can also access the sculpting workspace from within the MODEL workspace. Check the Model section of the help for more information or exercises directly related to the MODEL workspace.

SCULPT

The SCULPT workspace is a sub-environment of the MODEL Workspace. The sculpting tools allow you to push and pull the geometry from vertices and edges into desired shapes. You can create and modify 2D or 3D surface geometry and 3D solid objects in the SCULPT workspace. Read through the Sculpt section of the help for more details.

PATCH

The PATCH workspace allows you to create and edit 2D or 3D surface geometry. Working with surface geometry is slightly different from working with traditional 3D solid bodies, so it has been broken out into a separate workspace. Many designers use surface modelling techniques along with 3D modelling so it is not uncommon to bounce back and forth between the PATCH and other modelling workspaces. The Patch section of the help has more details about surface modelling.

RENDER

The RENDER workspace is used to generate realistic pictures of your designs. Using capabilities like lighting and adding decals, you can show your design as if it were already a live prototype. More information about rendering capabilities can be found here.

ANIMATION

The ANIMATION workspace provides tools for creating videos. Easily share your videos to communicate your design features and functionality. Videos provide needed insight about your designs and can assist others in understanding and evaluating them. More information about the ANIMATION workspace can be found here.

SIMULATION

The SIMULATION workspace allows you to use finite element analysis to simulate how the design performs under various loads and conditions. Understanding the physical limitations of your design (with regard to stress and temperature) is invaluable. Knowing if a design is in danger of failure, or is possibly over-engineered, helps you to make informed decisions about design changes. Creating a more efficient and better design on the first manufacturing pass is always a good idea. Simulation can also minimise or eliminate the necessity of building prototypes and performing destructive testing. Read more in the Simulation section of the help.

CAM

When the time comes to take your design from digital to fabricated, take advantage of Computer-Aided Manufacturing capabilities in the CAM workspace. You can produce tool-path strategies for fabricating your designs. Export your tool-paths to CNC (Computer Numerically Controlled) machines to make your design a reality. Check out the CAM section of the help for more information.

MESH

Using the MESH workspace, you can repair and re-mesh mesh bodies. This workspace must be enabled in the Preview section of the Preference dialog. A mesh body is a representation of a solid volume using many short line segments arranged in triangles or quadrilaterals to form the faces. Use meshes for 3D printing. You can also manipulate meshed bodies in the SCULPT workspace. More information about the Mesh workspace.

Note: Do not confuse mesh bodies with the finite element analysis (FEA) mesh produced in the SIMULATION workspace. FEA mesh lines indicate the edges of individual solid elements, and the elements extend throughout the volume of solid parts, not just at the faces. The endpoints of FEA mesh lines are nodes (or grid points) at which simulation results are calculated.

DRAWING

Create standard 2D drawings from your 3D geometry by entering the DRAWING workspace. See the Drawings section of the help for more information.

Direct Modelling and how it affects workspaces

Fusion 360 also lets you switch between direct modelling and history modelling, or use both at the same time. If the history timeline is turned off, the Sculpt and Mesh environments become their own workspaces, accessible through the workspace drop-down menu.

With the design history turned on, creating a Base-feature allows you to go into a direct modelling "sandbox". This adds a Base-feature into the history timeline, but does not capture any more actions you perform for as long as you are in this Base-feature. Clicking on Finish Base Feature allows you to exit out of Base-feature, and go back into history modelling.

MODEL

- **Start a solid body using basic shapes**
  Create solid bodies using primitive shapes.
- **Create a 3D sketch**
  Sketches can contain 2D and 3D sketch entities.
- **About direct and parametric modeling**
  Learn about two basic ways how to work in a model workspace - direct and parametric modelling.

- **Use materials to control the appearance of a design**

  This video shows how to use physical materials and visual materials. It also demonstrates how to remove material overrides and adjust the projection of texture maps.

Assembly Modelling

A Fusion 360 assembly model is a collection of parts and subassemblies that function as a single unit. Parts and subassemblies are connected by assembly relationships

Since there is no special file type for assemblies, each Fusion design can be a single part with only bodies, or an assembly with jointed components, or a mix of the two. Components that themselves have sub-components are considered to be assemblies.

When you first make components your single component design becomes an Assembly.

Workflow for assemblies:

- If your intent is to create an assembly of components, it is best to create the components early using New Component, and then create sketches and bodies inside the components they belong to from the start. Use Activate Component radio button before creating more sketches and bodies.
- If you have no need to re-use design, simply use the New Component options inside dialogs like Extrude will create a component, and thus your design becomes an Assembly.

The following video gives you an introduction to the assemblies, and it covers following areas:

- Creating assembly structure
- Using joints to position components
- Simulating motion in assemblies

Fusion 360 allows for models to be fully designed within a single environment, starting from a few base sketches and working all the way up to a full-scale assembly that includes motion. Fusion 360 makes it simple to design all of the necessary bodies in context to one another, and when you want to add mechanical motion to the model, bodies can be converted into components in order to enable motion using "Joints", which I'll dive deeper into later in this course. This converts the entire design into an assembly.

CHAPTER-2.4

PCB DESIGNING

PCB DESIGNING

The success of any creation is often dependent on the foundations it is built upon, be it the strength of a character, depth of a building's foundations or the extent of a tree's roots. Much in the same way, the success of any electronic device depends on what it is built on. The motherboard of any electronics device serves as a playground and a host to every form of electrical signal that performs some function for the equipment. Be it the communication signal between the North Bridge and processor on a computer, or a simple on-off signal in a routine school project, the effectiveness of the design is a function of the capabilities offered by the base board itself.

A **Printed Circuit Board** doesn't just connect electrical components using etched copper pathways, but also provides mechanical strength to it. Printed Circuit Boards, or more appropriately, Printed Wiring Boards are found in almost all of the commercial products as a packaging medium as building blocks.

PCBs are a composite of organic and/or inorganic dielectric materials with many layers with wiring interconnects and also house components like inductors and capacitors. There isn't any standard printing board as such and each board is unique, often a function of the product itself. There are industry standards for almost every aspect of **PCB design**, controlled by IPC, for example the IPC-2221, 'Generic Standard on Printed Board Design'.

History of PCB designing

PCBs have evolved from the electrical connection systems developed in the 1850s. The first patents on Printed Wires were issued in 1903. Albert Hanson explained a layered structure of foil conductors laminated to insulation boards. Arthur Berry patented a 'Print-and-Etch' method in 1913 and Max Schoop patented Flame Spraying metal onto a board via a mask. Thomas Edison had experimented with chemicals for plating conductors on linen paper way back in 1904, but the method of electroplating circuit patterns was finally successfully patented to Charles Durcase in the year 1927. Charles Ducas had earlier patented a technique of creating electrical paths directly using stencils and electrically conductive ink in 1925.

World War II saw the invention of circuit boards that could withstand gunshots. But, the credit of developing the first PCB is given to Paul Eisler in 1943, for developing a method of etching conductive circuits on copper foil bonded to a non-conductive base reinforced by glass. The method remained dormant until late 50s when the transistors were introduced for commercial use. The presence of wire leads on electronic components led to the development of 'Through Hole' technology where holes were drilled into the PCB and the components soldered on to the board at those points. It was patented by a U.S. firm Hazeltyne in 1961. However, this process being slightly expensive and wasteful as the extra wire is cut off and not used much. Nowadays, 'surface mount' technology is gaining impetus as the demand for smaller, high density circuits is increasing.

Modern PCBs are designed with dedicated layout software, generally in the following steps

1. Schematic capture through an electronic design automation (*EDA*) tool.
2. Card dimensions and template are decided based on required circuitry and case of the PCB.
3. The positions of the components and heat sinks are determined.
4. Layer stack of the PCB is decided, with one to tens of layers depending on complexity. Ground and power planes are decided. A power plane is the counterpart to a ground plane and behaves as an AC signal ground while providing DC power to the circuits mounted on the PCB. Signal interconnections are traced on signal planes. Signal planes can be on the outer as well as inner layers. For optimal EMI performance high frequency signals are routed in internal layers between power or ground planes.
5. Line impedance is determined using dielectric layer thickness, routing copper thickness and trace-width. Trace separation is also taken into account in case of differential signals. Microstrip, stripline or dual stripline can be used to route signals.
6. Components are placed. Thermal considerations and geometry are taken into account. Vias and lands are marked.

7. Signal traces are routed. Electronic design automation tools usually create clearances and connections in power and ground planes automatically.
8. Gerber files are generated for manufacturing.

MANUFACTURING : PCB CAM

Manufacturing starts from the PCB fabrication data generated by computer aided design, such as Gerber layer images, Gerber or Excellon drill files, IPC-D-356 netlist and component information. The Gerber or Excellon files in the fabrication data are never used directly on the manufacturing equipment but always read into the CAM (Computer Aided Manufacturing) software. CAM performs the following functions:

1. Input of the fabrication data.
2. Verification of the data; optionally DFM
3. Compensation for deviations in the manufacturing processes (e.g. scaling to compensate for distortions during lamination)
4. Panelization
5. Output of the digital tools (copper patterns, solder resist image, legend image, drill files, automated optical inspection data, electrical test files,...)

Panelization

Panelization is a procedure whereby a number of PCBs are grouped for manufacturing onto a larger board - the panel. Usually a panel consists of a single design but sometimes multiple designs are mixed on a single panel. There are two types of panels: assembly panels - often called arrays - and bare board manufacturing panels. The assemblers often mount components on panels rather than single PCBs because this is efficient.The bare board manufactures always uses panels, not only for efficiency, but because of the requirements of the plating process. Thus a manufacturing panel can consist of a grouping of individual PCBs or of arrays, depending on what must be delivered.

The panel is eventually broken apart into individual PCBs; this is called de-paneling. Separating the individual PCBs is frequently aided by drilling or routing perforations along the boundaries of the individual circuits, much like a sheet of postage stamps. Another method, which takes less space, is to cut V-shaped grooves across the full dimension of the panel. The individual PCBs can then be broken apart along this line of weakness. Today de-paneling is often done by lasers which cut the board with no contact. Laser panelization reduces stress on the fragile circuits.

Copper patterning

The first step is to replicate the pattern in the fabricator's CAM system on a protective mask on the copper foil PCB layers. Subsequent etching removes the unwanted copper. (Alternatively, a conductive ink can be ink-jetted on a blank (non-conductive) board. This technique is also used in the manufacture of hybrid circuits.)

1. **Silk screen printing** uses etch-resistant inks to create the protective mask.
2. **Photo-Engraving** uses a photomask and developer to selectively remove a UV-sensitive photoresist coating and thus create a photoresist mask. Direct imaging techniques are sometimes used for high-resolution requirements. Experiments were made with thermal resist.
3. **PCB milling** uses a two or three-axis mechanical milling system to mill away the copper foil from the substrate. A PCB milling machine (referred to as a 'PCB Prototyper') operates in a similar way to a plotter, receiving commands from the host software that control the position of the milling head in the x, y, and (if relevant) z axis.
4. **Laser resist** ablation Spray black paint onto copper clad laminate, place into CNC laser plotter. The laser raster-scans the PCB and ablates (vaporizes) the paint where no resist is wanted. (Note: laser copper ablation is rarely used and is considered experimental.

The method chosen depends on the number of boards to be produced and the required resolution.

Large volume

- Silk screen printing – Used for PCBs with bigger features
- Photoengraving – Used when finer features are required

Small volume

- Print onto transparent film and use as photo mask along with photo-sensitized boards (i.e., pre-sensitized boards), then etch. (Alternatively, use a film photo-plotter)
- Laser resist ablation
- PCB milling

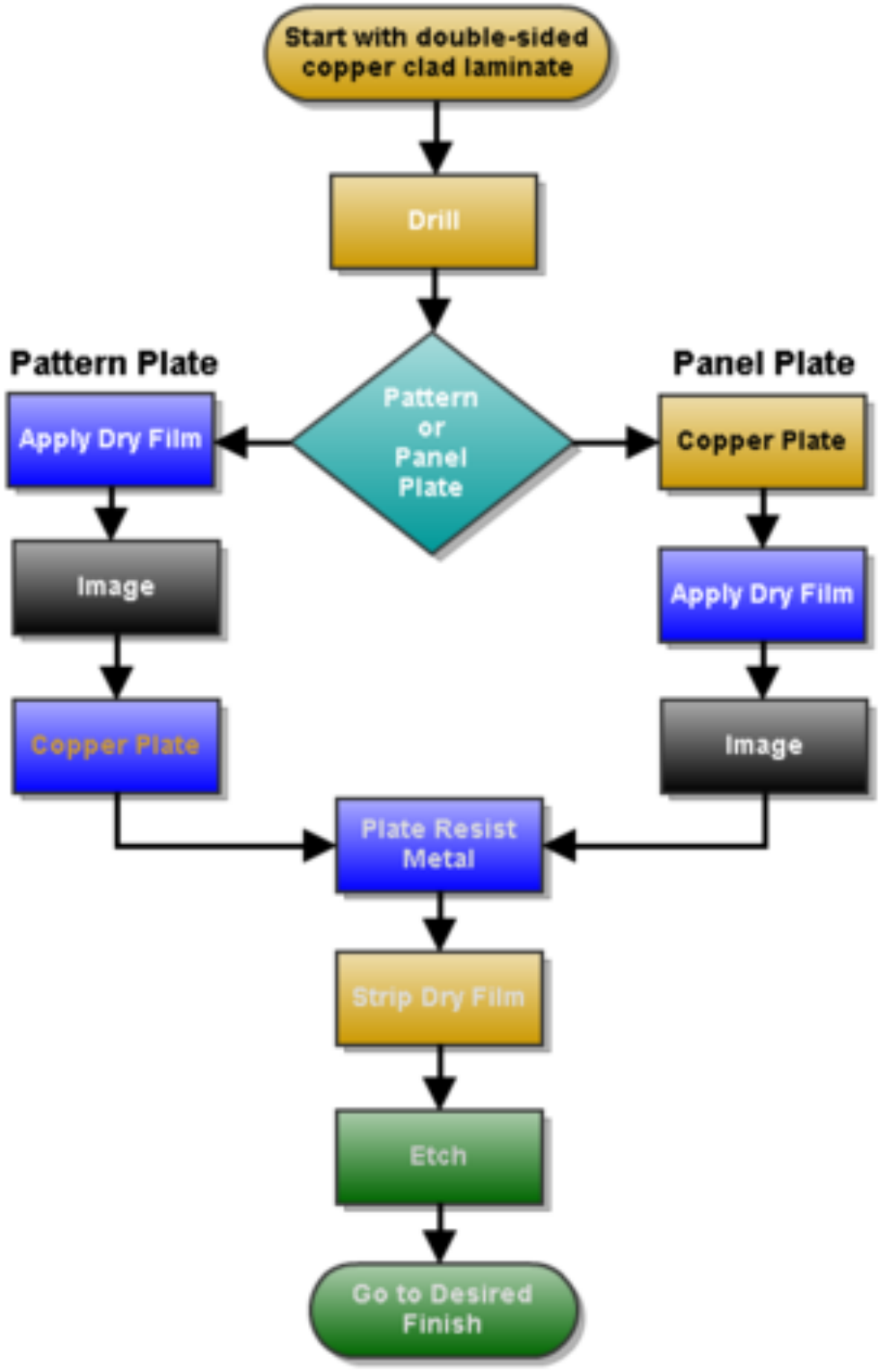


Figure: 2.4.1

The two processing methods used to produce a double sided PCB with plated-through holes

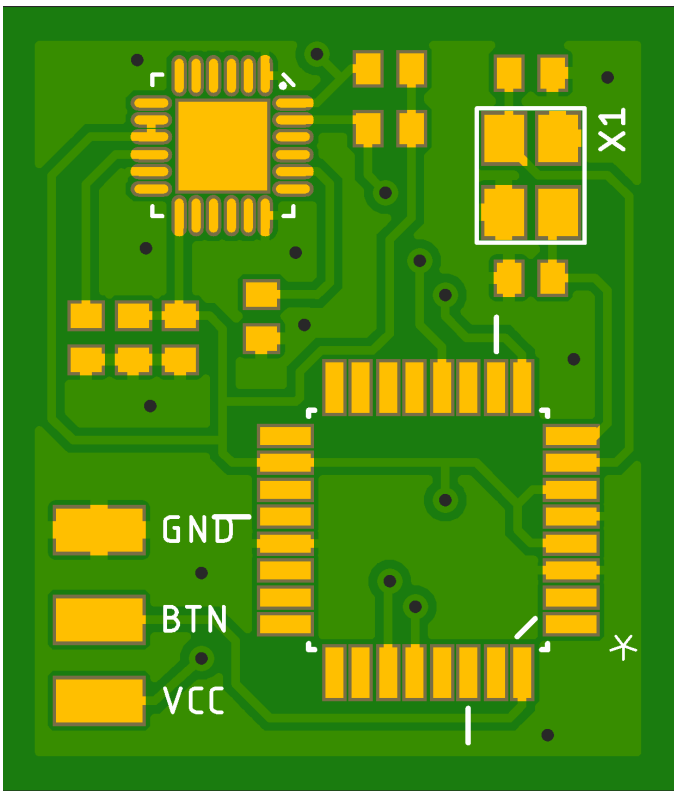


Figure 2.4.2
Main PCB Top View

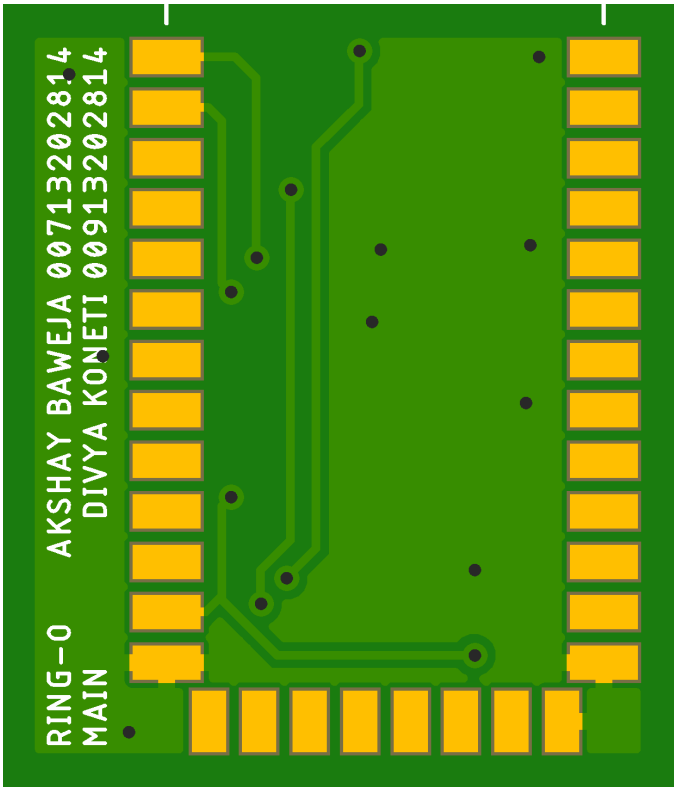


Figure 2.4.3
Main PCB bottom view

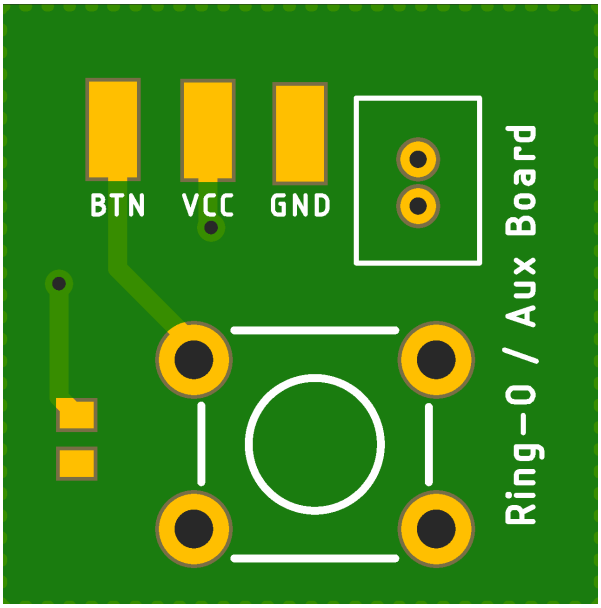


Figure 2.4.4
Aux PCB Top View

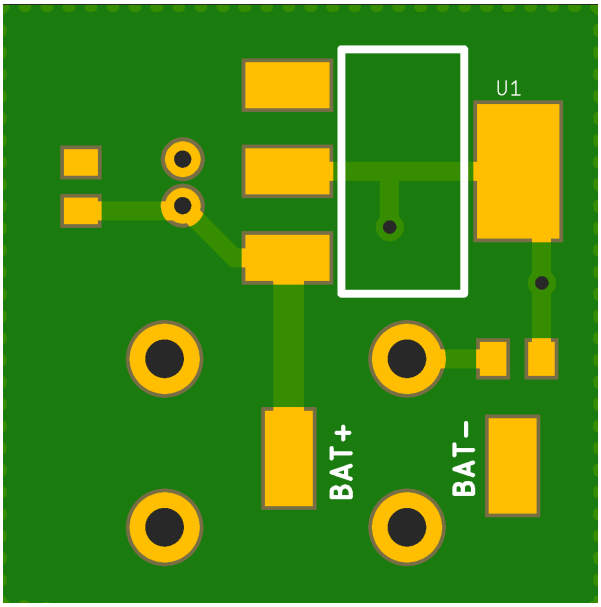


Figure 2.4.5
Aux PCB Bottom View

- The final PCB was manufactured by OSHPARK. OSH Park is a community printed circuit board (PCB) order.
- It brings you high quality, lead free boards (ENIG finish), manufactured in the USA, and shipped for free to anywhere in the world.

- This is a community printed circuit board (PCB) order. It take designs from lots of people, put them all together on a panel and then order the panel from a fab.

  Since they're all splitting the panel setup cost, this lets us make circuit boards inexpensively.

- 2 layer boards are $5 per square inch (with 3 copies of your board included in that price) and ship in under 12 calendar days from ordering.

  4 layer boards are $10 per square inch (also including 3 copies of your board), go to the fab once a week, and have a 2 week turn time from the fab.

- They offer purple solder-mask over bare copper (SMOBC) and an ENIG (Electroless Nickel Immersion Gold) finish.

  Anyone involved within the printed circuit board (PCB) industry understand that PCB's have copper finishes on their surface. If they are left unprotected then the copper will oxidise and deteriorate, making the circuit board unusable. The surface finish forms a critical interface between the component and the PCB. The finish has two essential functions, to protect the exposed copper circuitry and to provide a solderable surface when assembling (soldering) the components to the printed circuit board.

Hot Air Solder Levelling (HASL) was once the tried and true method of deliver consistent assembly results. However, the ever-increasing circuit complexity and component density has stretched the capabilities of even horizontal solder levelling systems to their limits.

As component pitches became finer and a need for a thin coating became greater, HASL represented a process limitation for PCB manufacturers. As an alternative to HASL, alternative coatings have been around for several years now, both electrolytic and immersion processes.

HASL / Lead Free HASL

HASL is the predominant surface finish used in the industry. The process consists of immersing circuit boards in a molten pot of a tin/lead alloy and then removing the excess solder by using 'air knives', which blow hot air across the surface of the board.

One of the unintended benefits of the HASL process is that it will expose the PCB to temperatures up to 265°C which will identify any potential delamination issues well before any expensive components are attached to the board.

Advantages:

- Low Cost
- Widely Available
- Re-workable
- Excellent Shelf Life

Disadvantages:

- Uneven Surfaces
- Not Good for Fine Pitch
- Contains Lead (HASL)
- Thermal Shock
- Solder Bridging
- Plugged or Reduced PTH's (Plated Through Holes)

Immersion Tin

According to IPC, the Association Connecting Electronics Industry, Immersion Tin (ISn) is a metallic finish deposited by a chemical displacement reaction that is applied directly over the basis metal of the circuit board, that is, copper. The ISn protects the underlying copper from oxidation over its intended shelf life.

Copper and tin however have a strong affinity for one another. The diffusion of one metal into the other will occur inevitably, directly impacting the shelf life of the deposit and the performance of the finish. The negative effects of tin whiskers growth are well described in industry related literature and topics of several published papers.

Advantages:

- Flat Surface
- No Pb
- Re-workable
- Top Choice for Press Fit Pin Insertion

Disadvantages:

- Easy to Cause Handling Damage
- Process Uses a Carcinogen (Thiourea)
- Exposed Tin on Final Assembly can Corrode
- Tin Whiskers
- Not Good for Multiple Reflow/Assembly Processes
- Difficult to Measure Thickness

OSP / Entek

OSP (Organic Solder-ability Preservative) or anti-tarnish preserves the copper surface from oxidation by applying a very thin protective layer of material over the exposed copper usually using a conveyorized process.

It uses a water-based organic compound that selectively bonds to copper and provides an organometallic layer that protects the copper prior to soldering. It's also extremely green environmentally in comparison with the other common lead-free finishes, which suffer from either being more toxic or substantially higher energy consumption.

Advantages:

- Flat Surface
- No Pb
- Simple Process
- Re-workable
- Cost Effective

Disadvantages:

- No Way to Measure Thickness

- Not Good for PTH (Plated Through Holes)
- Short Shelf Life
- Can Cause ICT Issues
- Exposed Cu on Final Assembly
- Handling Sensitive

Electroless Nickel Immersion Gold (ENIG)

ENIG is a two layer metallic coating of 2-8 µin Au over 120-240 µin Ni. The Nickel is the barrier to the copper and is the surface to which the components are actually soldered to. The gold protects the nickel during storage and also provides the low contact resistance required for the thin gold deposits. ENIG is now arguably the most used finish in the PCB industry due the growth and implementation of the RoHs regulation.

Advantages:

- Flat Surface
- No Pb
- Good for PTH (Plated Through Holes)
- Long Shelf Life

Disadvantages:

- Expensive
- Not Re-workable
- Black Pad / Black Nickel
- Damage from ET
- Signal Loss (RF)
- Complicated Process

Gold – Hard Gold

Hard Electrolytic Gold consists of a layer of gold plated over a barrier coat of nickel. Hard gold is extremely durable, and is most commonly applied to high-wear areas such as edge connector fingers and keypads.

Unlike ENIG, its thickness can vary by controlling the duration of the plating cycle, although the typical minimum values for fingers are 30 µin gold over 100 µin nickel for Class 1 and Class 2, 50 µin gold over 100 µin nickel for Class 3.

Hard gold is not generally applied to solderable areas, because of its high cost and its relatively poor solder-ability. The maximum thickness that IPC considers to be solderable is 17.8 µin, so if this type of gold must be used on surfaces to be soldered, the recommended nominal thickness should be about 5-10 µin.

Advantages:

- Hard, Durable Surface
- No Pb
- Long Shelf Life

Disadvantages:

- Very Expensive
- Extra Processing / Labor Intensive
- Use of Resist / Tape
- Plating / Bus Bars required
- Demarcation
- Difficulty with Other Surface Finishes
- Etching Undercut can Lead to Slivering / Flaking
- Not Solderable Above 17 µin
- Finish Does Not Fully Encapsulate Trace Sidewalls, Except in Finger Areas

It is important to select the appropriate surface finish for your project by considering the various options while factoring in performance requirements and material costs.

For an example, if you are looking for the lowest cost then Tin-Lead HASL might seem like a good choice, but it is not suitable for RoHS-compliant products. If your product does require RoHS, you might consider lead-free HASL. That is only if there are no fine pitch components, since LFHASL cannot be applied perfectly flat. If your design needs to be RoHS compliant and does use fine pitch components,

then you'll need to select a flat, lead-free finish, such as Immersion Silver or ENIG. Bear in mind that doing so will necessitate the use of more costly high temperature laminate.

If you are unsure of what you will need ,consult with a PCB fabricator prior to you making a selection. This will ensure that the combination of the surface finish and material will result in a high-yielding, cost-effective design that will perform as expected.

Advantages of PCBs

- Low labor
- Automatic assembly
- Good for production runs of 10 pieces to 1 million copies at extremely low cost and high production rates
- Low capacitance
- Thin size allows close packing of multiple boards or low profile products
- Dense packing of components allows for short circuit paths – low delay – fast circuits and small area
- Very consistent mechanical properties allows board to be part of physical packaging solution – support and bracing
- Very consistent electrical properties reduces circuit variation and improves overall quality
- High electrical properties allows for high performance and consistent performance circuits

CHAPTER-2.5

SOFTWARE

SOFTWARE

Language used -PYTHON

Python is an interpreted, object-oriented programming language similar to PERL, that has gained popularity because of its clear syntax and readability. Python is said to be relatively easy to learn and portable, meaning its statements can be interpreted in a number of operating systems, including UNIX-based systems, Mac OS, MS-DOS, OS/2, and various versions of Microsoft Windows 98. Python was created by Guido van Rossum, a former resident of the Netherlands, whose favourite comedy group at the time was Monty Python's Flying Circus. The source code is freely available and open for modification and reuse. Python has a significant number of users.

A notable feature of Python is its indenting of source statements to make the code easier to read. Python offers dynamic data type, ready-made class, and interfaces to many system calls and libraries. It can be extended, using the C or C++ language.

Python can be used as the script in Microsoft's Active Server Page (ASP) technology. The scoreboard system for the Melbourne (Australia) Cricket Ground is written in Python. Z Object Publishing Environment, a popular Web application server, is also written in the Python language.

Features and Philosophy

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing, and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name

resolution (late binding), which binds method and variable names during program execution.

Python's design offers some support for functional programming in the Lisp tradition. It has filter(), map(), and reduce() functions; list comprehensions, dictionaries, and sets; and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

The language's core philosophy is summarised in the document The Zen of Python (PEP 20), which includes aphorisms such as:

- Beautiful is better than ugly
- Explicit is better than implicit
- Simple is better than complex
- Complex is better than complicated
- Readability counts

Rather than having all of its functionality built into its core, Python was designed to be highly extensible. This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications. Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach.

While offering choice in coding methodology, the Python philosophy rejects exuberant syntax (such as that of Perl) in favour of a simpler, less-cluttered grammar. As Alex Martelli put it: "To describe something as 'clever' is not considered a compliment in the Python culture." Python's philosophy rejects the Perl "there is more than one way to do it" approach to language design in favour of "there should be one —and preferably only one—obvious way to do it".

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of CPython that would offer marginal increases in speed at the cost of clarity. When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-

time compiler. Cython is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

An important goal of Python's developers is keeping it fun to use. This is reflected in the language's name—a tribute to the British comedy group Monty Python—and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (from a famous Monty Python sketch) instead of the standard foo and bar.

Statements and control flow

Python's statements include :

- The assignment statement (token '=', the equals sign). This operates differently than in traditional imperative programming languages, and this fundamental mechanism (including the nature of Python's version of variables) illuminates many other features of the language. Assignment in C, e.g., x = 2, translates to "typed variable name x receives a copy of numeric value 2". The (right-hand) value is copied into an allocated storage location for which the (left-hand) variable name is the symbolic address. The memory allocated to the variable is large enough (potentially quite large) for the declared type. In the simplest case of Python assignment, using the same example, x = 2, translates to "(generic) name x receives a reference to a separate, dynamically allocated object of numeric (int) type of value 2." This is termed binding the name to the object. Since the name's storage location doesn't contain the indicated value, it is improper to call it a variable. Names may be subsequently rebound at any time to objects of greatly varying types, including strings, procedures, complex objects with data and methods, etc. Successive assignments of a common value to multiple names, e.g., x = 2; y = 2; z = 2 result in allocating storage to (at most) three names and one numeric object, to which all three names are bound. Since a name is a generic reference holder it is unreasonable to associate a fixed data type with it. However at a given time a name will be bound to some object, which **will** have a type; thus there is dynamic typing.

- The if statement, which conditionally executes a block of code, along with else and elif (a contraction of else-if).
- The for statement, which iterates over an iterable object, capturing each element to a local variable for use by the attached block.
- The while statement, which executes a block of code as long as its condition is true.
- The try statement, which allows exceptions raised in its attached code block to be caught and handled by except clauses; it also ensures that clean-up code in a finally block will always be run regardless of how the block exits.
- The class statement, which executes a block of code and attaches its local namespace to a class, for use in object-oriented programming.
- The def statement, which defines a function or method.
- The with statement (from Python 2.5), which encloses a code block within a context manager (for example, acquiring a lock before the block of code is run and releasing the lock afterwards, or opening a file and then closing it), allowing Resource Acquisition Is Initialization (RAII)-like behaviour.
- The pass statement, which serves as a NOP. It is syntactically needed to create an empty code block.
- The assert statement, used during debugging to check for conditions that ought to apply.
- The yield statement, which returns a value from a generator function. From Python 2.5, yield is also an operator. This form is used to implement coroutines.
- The import statement, which is used to import modules whose functions or variables can be used in the current program.There are two ways of using import . from <module name> import * or import <module name>.
- The print statement was changed to the print() function in Python.

Libraries

Python's large standard library, commonly cited as one of its greatest strengths, provides tools suited to many tasks. For Internet-facing applications, many standard formats and protocols such as MIME and HTTP are supported. It includes modules for creating graphical user interfaces, connecting to relational databases, generating pseudorandom numbers, arithmetic with arbitrary precision decimals,[86] manipulating regular expressions, and unit testing.

Some parts of the standard library are covered by specifications (for example, the Web Server Gateway Interface (WSGI) implementation wsgiref follows PEP 333), but most modules are not. They are specified by their code, internal documentation, and test suites (if supplied). However, because most of the standard library is cross-platform Python code, only a few modules need altering or rewriting for variant implementations.

LIBRARIES OF PYTHON USED:

matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, the jupyter notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

numpy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions

- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

NumPy is licensed under the BSD license, enabling reuse with few restrictions.

scipy

The SciPy library is one of the core packages that make up the SciPy stack. It provides many user-friendly and efficient numerical routines such as routines for numerical integration and optimisation.

scikit-learn

Machine Learning library in Python

- Simple and efficient tools for data mining and data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Tkinter

Tkinter is Python's de-facto standard GUI (Graphical User Interface) package. It is a thin object-oriented layer on top of Tcl/Tk. Tkinter is not the only GuiProgramming toolkit for Python.

CHAPTER-2.6

Bluetooth HC-05

HC−05 module is an easy to use Bluetooth SPP (Serial Port Protocol) module, designed for transparent wireless serial connection setup. The HC-05 Bluetooth Module can be used in a Master or Slave configuration, making it a great solution for wireless communication.This serial port bluetooth module is fully qualified Bluetooth V2.0+EDR (Enhanced Data Rate) 3Mbps Modulation with complete 2.4GHz radio transceiver and baseband. It uses CSR BlueCore 04 − External single chip Bluetooth system with CMOS technology and with AFH (Adaptive Frequency Hopping Feature).

The Bluetooth module HC-05 is a MASTER/SLAVE module. By default the factory setting is SLAVE.The Role of the module (Master or Slave) can be configured only by AT COMMANDS. The slave modules cannot initiate a connection to another Bluetooth device, but can accept connections. Master module can initiate a connection to other devices. The user can use it simply for a serial port replacement to establish connection between MCU and GPS, PC to your embedded project, etc.

Hardware Features

- Typical −80dBm sensitivity.
- Up to +4dBm RF transmit power.
- 3.3 to 5 V I/O.
- PIO(Programmable Input/Output) control.
- UART interface with programmable baud rate.
- With integrated antenna.
- With edge connector.

Software Features

- Slave default Baud rate: 9600, Data bits:8, Stop bit:1,Parity:No parity.
- Auto−connect to the last device on power as default.
- Permit pairing device to connect as default.
- Auto−pairing pin-code: ”1234” as default.

Figure 2.6.1
Bluetooth Module HC-05

CHAPTER - 2.7
WORKING OF RING-O

WORKING

Ring-O works on principle on Correlation and Machine Learning. It intakes data in form of values of accelerometer and gyroscope sensors i.e. MPU6050 from InvenSense and then stores them into a file. This file is then processed by various machine learning algorithms and correlated with the pre-trained model by the user. This pre-trained model returns the correlation percentage the sub-model that is the base character with highest percentage is returned to the user.

Ring-O gives user an option to self train the model whenever user make the gesture or character while using the Ring-O, not only this but also Ring-O also gives you the next letter suggestions based on currently typed letter(s). This is achieved with help of Python English dictionaries.

Working with the Dataset

This project was originally conceived to make a keyboard, so each gesture is associated with a character (case sensitive). This means that you can teach the algorithm a maximum of about 100 different gestures.

Let's start a new recording batch, where you will record new samples for a specific gesture. Open the terminal and write:

```
python start.py target=a:0
```

Explanation:

- The "target" argument tells the module that we want to record new samples for a specific gesture.
- The "a" character is the one that characterise a gesture, it must be unique for every gesture and it must have a length of 1.
- The "0" character is the *batch number*, it must be different every time you register a new batch to avoid sample overriding. For example, the first time you record a batch set it to 0, the next time 1 and so on.

- The "port" represents the serial port your Arduino is connected to.

When `start.py` is running, you will record different samples by pushing and releasing the integrated button on the Arduino. When the button is pressed, the library records the data from the accelerometer. For each gesture, a good number of samples is 40.

Every sample get saved as a different file in the data folder.

Train the Model

When your dataset is ready, you can use it to train the machine learning algorithm. This is pretty straightforward, open the terminal and write:

```
python learn.py
```

Use the Model

To check out if the model is working, open the terminal and write:

```
python start.py predict #to start keyboard mode
python start.py gesture #to start gesture mode
```

CHAPTER-3
RESULTS
and
DISCUSSION

3.1. RESULT

- A gesture controlled Ring-O has been designed successfully. Also it is well trained so as to give maximum efficiency. There is no error while detecting the characters or gestures thus each and every character is being recognised perfectly without any time lag and the required 3D printed model has been designed properly to hold the Micro-controller and Bluetooth Module and MPU6050 together.

- The hardware required for the project is minimal and is very handy thus can be carried anywhere. Additionally, it is user friendly, it does not require much of human efforts to run the device, just the knowledge of basic characters and some specific gestures is enough to use the device. This, being a compact device can be used anywhere and there are no interface constraints as well.

- The algorithm designed is working efficiently. It is also easy to understand and not too complex. On a whole the hardware and software are both user friendly and working perfectly.

- The Ring-O is fully functional and is exactly same as the proposed model in terms of its structure and functionalities with some advancements to it. The response of the device is well within the specified range thus there is no time lag between input and output. Thus it is working perfectly without any hardware or software flaws.

3.2. DISCUSSION

This device has been designed to reduce the human efforts and to shrink the mouse and keyboard into a single device. One of the major uses of this device is security, people can unlock their personal data secretly without someone else's knowledge. Its advanced version can be used to make calls in emergency conditions for women

protection. Further people can use it while answering an important message while driving without leading to accidents which proves to be a boon for this fast moving generation. Its compact size helps the user to carry everywhere without any efforts. Moreover it consumes minimal power. There is no problem like heating or any other losses.

CHAPTER-4

CONCLUSION
and
FUTURE SCOPE

4.1. CONCLUSION

A gesture controlled Ring-O has been designed successfully. Also it is well trained so as to give maximum efficiency. There is no error while detecting the characters or gestures thus each and every character is being recognised perfectly without any time lag and the required 3D printed model has been designed properly to hold the raspberry pi circuit and the PCB together and the MPU6050.

This device has been designed to reduce the human efforts and to shrink the mouse and keyboard into a single device.One of the major uses of this device is security, people can unlock their personal data secretly without someone else's knowledge. Its advanced version can be used to make calls in emergency conditions for women protection. Further people can use it while answering an important message while driving without leading to accidents which proves to be a boon for this fast moving generation.Its compact size helps the user to carry everywhere without any efforts. Moreover it consumes minimal power. There is no problem like heating or any other losses.

4.2. FUTURE SCOPE

The Ring-O is a combination of keyboard and mouse. It can further be expanded into a smart keypad which can be interpreted by an android application and can be run on smartphones as well. As a result people can place a call or they can send messages just by typing in the air.

Ring-O lets you map hand gestures to keyboard shortcuts, so the advanced version of it can be used to control almost any application. We can simply tune into our favourite songs just by changing the gestures or just flick our finger left up to increase the volume and left down to decrease it. Thus we can control the volume bar as well.

It can further be used to swipe left or right for space and delete respectively.

REFERENCES

References

[1] Alex Fu and Yangyang Yu. Real-time Gesture Pattern Classification with IMU Data. [Online]. Available: http://stanford.edu/class/ee267/Spring2017/report_fu_yu.pdf . [Accessed: March 4, 2018]

[2] A. Abdullah, N. A. Abdul-Kadir and F. K. Che Harun. Research and Development of IMU Sensors-based Approach for Sign Language Gesture Recognition [Online]. Available: http://journal.utem.edu.my/index.php/jtec/article/view/3122/2260. [Accessed: March 4, 2018]

[3] "MPU-6050 | TDK". [Online]. Available: https://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/. [Accessed: March 5, 2018]

[4] "Atmel ATMega328 Documentation". [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Summary.pdf. [Accessed: March 7, 2018]

[5] "OSH Park - An electric ecosystem". [Online]. Available: https://oshpark.com [Accessed: March 12, 2018]

[6] "Fusion 360 | Free Software for Students, Educators | Autodesk". [Online]. Available: https://www.autodesk.com/products/fusion-360/students-teachers-educators. [Accessed: April 12, 2018]

[7] "PCB Design & Schematic Software | EAGLE | Autodesk". [Online]. Available: https://www.autodesk.com/products/eagle/overview. [Accessed: Nov. 12, 2018]

[8] "FLASHFORGE NEW Creator Pro Dual Extrusion 3D Printer". [Online]. Available: http://www.flashforge-usa.com/shop/flashforge-new-creator-pro-dual-extrusion-3d-printer.html. [Accessed: April 15, 2018]

[9] "About Python | Python.org". [Online]. Available: https://www.python.org/about/. [Accessed: April 13, 2018]

[10] "Scientific Computing Tools for Python". [Online]. Available: https://www.scipy.org/about.html. [Accessed: April 20, 2018]

ANNEXURE A

CODE

CODE

• start.py

```
import serial, os, signals, sys, suggestions
from sklearn.externals import joblib

'''

This module is used to register new signals,
and test the predictions.

You can use different modes to achive different things:

TARGET MODE: Used to register new samples
You must specify the target sign and the batch with this sintax:

python start.py target=a:3

Where 'a' is the target sign and 3 is the batch.
It saves the recorded sign in a file named "a_sample_3_N.txt" in the
directory specified by the "target_directory".
PS. N is a progressive number in the batch used to make each
recording unique.

TARGET_ALL MODE: Used to register new samples by providing a sentence
that contains all letters. The user have to write the sentence using
the
device and the module save the corresponding recorded sample.

WRITE MODE: Register new samples and translate them into text by
predicting
the correct character. If the parameter "noautocorrect" is used,
the predictions will not be cross-corrected with the dictionary

The if the predicted char is upper case, it has a special meaning:

D = delete
A = delete all ( Not enabled by default, you must toggle
"DELETE_ALL_ENABLED" )

'''

def print_sentence_with_pointer(sentence, position):
    print sentence
    print " "*position + "^"

#Sentence used to get samples because it contains all letters.
#ALTERNATIVE: the quick brown fox jumps over the lazy dog
test_sentence = "pack my box with five dozen liquor jugs"

#Mode parameters, controlled using sys.argv by the terminal
TRY_TO_PREDICT = False
SAVE_NEW_SAMPLES = False
FULL_CYCLE = False
ENABLE_WRITE = False
```

```
TARGET_ALL_MODE = False
AUTOCORRECT = True
DELETE_ALL_ENABLED = False

#Serial parameters
SERIAL_PORT = "/dev/cu.usbserial-A50285BI"
BAUD_RATE = 38400
TIMEOUT = 100

#Recording parameters
target_sign = "a"
current_batch = "0"
target_directory = "data"

current_test_index = 0

#Analyzes the arguments to enable a specific mode

arguments = {}

for i in sys.argv[1:]:
    if "=" in i:
        sub_args = i.split("=")
        arguments[sub_args[0]]=sub_args[1]
    else:
        arguments[i]=None

#If there are arguments, analyzes them
if len(sys.argv)>1:
    if arguments.has_key("target"):
        target_sign = arguments["target"].split(":")[0]
        current_batch = arguments["target"].split(":")[1]
        print "TARGET SIGN: '{sign}' USING BATCH:
{batch}".format(sign=target_sign, batch = current_batch)
        SAVE_NEW_SAMPLES = True
    if arguments.has_key("predict"):
        TRY_TO_PREDICT = True
    if arguments.has_key("write"):
        TRY_TO_PREDICT = True
        ENABLE_WRITE = True
    if arguments.has_key("test"):
        current_batch = arguments["test"]
        TARGET_ALL_MODE = True
        SAVE_NEW_SAMPLES = True
    if arguments.has_key("noautocorrect"):
        AUTOCORRECT=False
    if arguments.has_key("port"):
        SERIAL_PORT = arguments["port"]

clf = None
classes = None
sentence = ""
hinter = suggestions.Hinter.load_english_dict()

#Loads the machine learning model from file
if TRY_TO_PREDICT:
    print "Loading model..."
    clf = joblib.load('model.pkl')
```

```
    classes = joblib.load('classes.pkl')


print "OPENING SERIAL_PORT '{port}' WITH BAUDRATE
{baud}...".format(port = SERIAL_PORT, baud = BAUD_RATE)

print "IMPORTANT!"
print "To end the program hold Ctrl+C and send some data over serial"

#Opens the serial port specified by SERIAL_PORT with the specified
BAUD_RATE
ser = serial.Serial(SERIAL_PORT, BAUD_RATE, timeout = TIMEOUT)

output = []

in_loop = True
is_recording = False

current_sample = 0

#Resets the output file
output_file = open("output.txt","w")
output_file.write("")
output_file.close()

#If TARGET_ALL_MODE = True, print the sentence with the current
position
if TARGET_ALL_MODE:
    print_sentence_with_pointer(test_sentence, 0)

try:
    while in_loop:
        #Read a line over serial and deletes the line terminators
        line = ser.readline().replace("\r\n","")
        #If it receive "STARTING BATCH" it starts the recording
        if line=="STARTING BATCH":
            #Enable the recording
            is_recording = True
            #Reset the buffer
            output = []
            print "RECORDING...",
        elif line=="CLOSING BATCH": #Stops recording and analyzes the
result
            #Disable recording
            is_recording = False
            if len(output)>1: #If less than 1, it means error
                print "DONE, SAVING...",

                #If TARGET_ALL_MODE is enabled changes the target
sign
                #according to the position
                if TARGET_ALL_MODE:
                    if current_test_index<len(test_sentence):
                        target_sign =
test_sentence[current_test_index]
                    else:
                        #At the end of the sentence, it quits
                        print "Target All Ended!"
```

```
                        quit()

                #Generates the filename based on the target sign,
batch and progressive number
                filename = "{sign}_sample_{batch}
_{number}.txt".format(sign = target_sign, batch = current_batch,
number = current_sample)
                #Generates the path
                path = target_directory + os.sep + filename

                #If SAVE_NEW_SAMPLES is False, it saves the recording
to a temporary file
                if SAVE_NEW_SAMPLES == False:
                    path = "tmp.txt"
                    filename = "tmp.txt"

                #Saves the recording in a file
                f = open(path, "w")
                f.write('\n'.join(output))
                f.close()
                print "SAVED IN {filename}".format(filename =
filename)

                current_sample += 1

                #If TRY_TO_PREDICT is True, it utilizes the model to
predict the recording
                if TRY_TO_PREDICT:
                    print "PREDICTING..."
                    #It loads the recording as a Sample object
                    sample_test = signals.Sample.load_from_file(path)

                    linearized_sample =
sample_test.get_linearized(reshape=True)
                    #Predict the number with the machine learning
model
                    number = clf.predict(linearized_sample)
                    #Convert it to a char
                    char = chr(ord('a')+number[0])

                    #Get the last word in the sentence
                    last_word = sentence.split(" ")[-1:][0]

                    #If AUTOCORRECT is True, the cross-calculated
char will override the predicted one
                    if AUTOCORRECT and char.islower():
                        predicted_char =
hinter.most_probable_letter(clf, classes, linearized_sample,
last_word)
                        if predicted_char is not None:
                            print "CURRENT WORD: {word}, PREDICTED
{old}, CROSS_CALCULATED {new}".format(word = last_word, old = char,
new = predicted_char)
                            char = predicted_char

                    #If the mode is WRITE, assigns special meanings
to some characters
                    #and builds a sentence with each char
```

```
                    if ENABLE_WRITE:
                        if char == 'D': #Delete the last character
                            sentence = sentence[:-1]
                        elif char == 'A': #Delete all characters
                            if DELETE_ALL_ENABLED:
                                sentence = ""
                            else:
                                print "DELETE_ALL_ENABLED = FALSE"
                        else: #Add the char to the sentence
                            sentence += char
                        #Prints the last char and the sentence
                        print "[{char}] -> {sentence}".format(char =
char, sentence = sentence)
                        #Saves the output to a file
                        output_file = open("output.txt","w")
                        output_file.write(sentence)
                        output_file.close()
                    else:
                        print char
            else: #In case of a corrupted sequence
                print "ERROR..."
                current_test_index -= 1

            #If TARGET_ALL_MODE=True it shows the current position in
the sentence
            if TARGET_ALL_MODE:
                current_test_index += 1
                print_sentence_with_pointer(test_sentence,
current_test_index)
        else:
            #Append the current signal line in the recording
            output.append(line)
except KeyboardInterrupt: #When Ctrl+C is pressed, the loop
terminates
    print 'CLOSED LOOP!'

#Closes the serial port
ser.close()
```

• learn.py

```
from sklearn import datasets
from sklearn import svm
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import scipy as sp
import os, signals
from sklearn.externals import joblib
from sklearn.model_selection import GridSearchCV

#Check if the module is executed as main, needed for parallel
processing
if __name__ == '__main__':
      #List of parameters
      SHOW_CONFUSION_MATRIX = False

      x_data = []
      y_data = []

      classes = {}

      root="data" #Default directory containing the dataset

      print "Loading the dataset from
'{directory}'...".format(directory=root),

      #Fetch all the data files from the root directory of the
dataset
      for path, subdirs, files in os.walk(root):
            for name in files:
                  #Get the filename
                  filename = os.path.join(path, name)
                  #Load the sample from file
                  sample = signals.Sample.load_from_file(filename)
                  #Linearize the sample and then add it to the x_data
list
                  x_data.append(sample.get_linearized())
                  #Extract the category from the file name
                  #For example, the file "a_sample_0.txt" will be
considered as "a"
                  category = name.split("_")[0]
                  #Get a number for the category, as an offset from
the category
                  #to the a char in Ascii
                  number = ord(category) - ord("a")
                  #Add the category to the y_data list
                  y_data.append(number)
                  #Include the category and the corresponding number
into a dictionary
                  #for easy access and referencing
                  classes[number] = category

      print "DONE"

      #Parameters used in the cross-validated training process
      #The library automatically tries every possible combination to
```

```
      #find the best scoring one.
      params = {'C':[0.001,0.01,0.1,1], 'kernel':['linear']}

      #Inizialize the model
      svc = svm.SVC(probability = True)
      #Inizialize the GridSearchCV with 8 processing cores and
maximum verbosity
      clf = GridSearchCV(svc, params,verbose =10, n_jobs=8)

      #Split the dataset into two subset, one used for training and
one for testing
      X_train, X_test, Y_train, Y_test = train_test_split(x_data,
                        y_data, test_size=0.35, random_state=0)

      print "Starting the training process..."

      #Start the training process
      clf.fit(X_train, Y_train)

      #If SHOW_CONFUSION_MATRIX is true, prints the confusion matrix
      if SHOW_CONFUSION_MATRIX:
            print "Confusion Matrix:"
            Y_predicted = clf.predict(X_test)
            print confusion_matrix(Y_test, Y_predicted)

      print "\nBest estimator parameters: "
      print clf.best_estimator_

      #Calculates the score of the best estimator found.
      score = clf.score(X_test, Y_test)

      print "\nSCORE: {score}\n".format(score = score)

      print "Saving the model...",

      #Saves the model to the "model.pkl" file
      joblib.dump(clf, 'model.pkl')
      #Saves the classes to the "classes.pkl" file
      joblib.dump(classes, 'classes.pkl')

      print “DONE"
```

• gesture.py

```
import applescript

script='''
on volumeUp()
   set curVolume to output volume of (get volume settings)

    if curVolume < 90 then
       set newVolume to curVolume + 10
    else
       set newVolume to 100
    end if

   set volume output volume newVolume
end volumeUp

on volumeDown()

   set curVolume to output volume of (get volume settings)

    if curVolume > 10 then
       set newVolume to curVolume - 10
    else

       set newVolume to 1
    end if

   set volume output volume newVolume
end volumeDown

on volumeMute()

   set isMuted to output muted of (get volume settings)

   set newMuted to not isMuted

   set volume output muted newMuted
end volumeMute

on nextSlide()

    tell application "Keynote"
        show next
    end tell
end nextSlide
'''

scpt = applescript.AppleScript(script)

def gesture_callback(gesture):
    if gesture == "a":
        scpt.call('volumeMute')
        print "Muted."
    elif gesture == "b":
        scpt.call('volumeUp')
        print "Volume Up by 10"
```

```
elif gesture == "c":
    scpt.call('volumeDown')
    print "Volume Down by 10"
elif gesture == "d":
    scpt.call('nextSlide')
    print "Next Slide"
```

- **plot_signal.py**

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import scale
from scipy.interpolate import interp1d
import sys

'''
This module simply plots a signal.
You can specify the signal filename as a parameter in the terminal.
'''

filename = sys.argv[1]
sample_size_fit = 50
data_raw = [map(lambda x: float(x), i.split(" ")[1:-1]) for i in
open(filename)]

data = np.array(data_raw)
print data

data_norm = scale(data)

acx = data_norm[:,0]
acy = data_norm[:,1]
acz = data_norm[:,2]

gx = data_norm[:,3]
gy = data_norm[:,4]
gz = data_norm[:,5]

x = np.linspace(0, data.shape[0], data.shape[0])
f_acx = interp1d(x, acx)
f_acy = interp1d(x, acy)
f_acz = interp1d(x, acz)

f_gx = interp1d(x, gx)
f_gy = interp1d(x, gy)
f_gz = interp1d(x, gz)

xnew = np.linspace(0, data.shape[0], sample_size_fit)
acx_stretch = f_acx(xnew)
acy_stretch = f_acy(xnew)
acz_stretch = f_acz(xnew)

gx_stretch = f_gx(xnew)
gy_stretch = f_gy(xnew)
gz_stretch = f_gz(xnew)

plt.plot(acx_stretch)
plt.plot(acy_stretch)
plt.plot(acz_stretch)

plt.plot(gx_stretch)
plt.plot(gy_stretch)
plt.plot(gz_stretch)
plt.show()
```

• plot_signal_process.py

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.preprocessing import scale
from scipy.interpolate import interp1d
import sys

'''
This module simply plots a signal.
You can specify the signal filename as a parameter in the terminal.
'''

ALL_AXES=False

filename = sys.argv[1]
sample_size_fit = 50

data_raw = [map(lambda x: float(x), i.split(" ")[1:-1]) for i in
open(filename)]

data = np.array(data_raw)

f, axarr = plt.subplots(3)

axarr[0].set_title("Raw Data")
if ALL_AXES:
      axarr[0].plot(data)
else:
      axarr[0].plot(data[:,1])

#print data

data_norm = scale(data)

axarr[1].set_title("Y Normalized Data")
if ALL_AXES:
      axarr[1].plot(data_norm)
else:
      axarr[1].plot(data_norm[:,1])

acx = data_norm[:,0]
acy = data_norm[:,1]
acz = data_norm[:,2]

gx = data_norm[:,3]
gy = data_norm[:,4]
gz = data_norm[:,5]

x = np.linspace(0, data.shape[0], data.shape[0])
f_acx = interp1d(x, acx)
f_acy = interp1d(x, acy)
f_acz = interp1d(x, acz)

f_gx = interp1d(x, gx)
f_gy = interp1d(x, gy)
f_gz = interp1d(x, gz)
```

```
xnew = np.linspace(0, data.shape[0], sample_size_fit)

acx_stretch = f_acx(xnew)
acy_stretch = f_acy(xnew)
acz_stretch = f_acz(xnew)

gx_stretch = f_gx(xnew)
gy_stretch = f_gy(xnew)
gz_stretch = f_gz(xnew)


axarr[2].set_title("X Normalized to 50 samples")
axarr[2].plot(acx_stretch)
if ALL_AXES:
      axarr[2].plot(acy_stretch)
      axarr[2].plot(acz_stretch)

      axarr[2].plot(gx_stretch)
      axarr[2].plot(gy_stretch)
      axarr[2].plot(gz_stretch)

plt.show()

# print data
# print data_norm
```

• signals.py

```
import numpy as np
from sklearn.preprocessing import scale
from scipy.interpolate import interp1d

'''
This library contains the classes needed to process the signals.
'''

class Sample:
      '''
      Sample is used to load, store and process the signals obtained
      from the accelerometers.
      It provides a method to load the signals from file and process
them.
      '''
      def __init__(self, acx, acy, acz, gx, gy, gz):
            self.acx = acx
            self.acy = acy
            self.acz = acz
            self.gx = gx
            self.gy = gy
            self.gz = gz

      def get_linearized(self, reshape = False):
            '''
            Linearize the data, combining the 6 different axes.
            Useful to feed the data into a machine learning
algorithm.

            If reshape=True it reshape it (Useful when feeding it to
the predict method)
            '''
            if reshape:
                  return np.concatenate((self.acx, self.acy,
self.acz, self.gx, self.gy, self.gz)).reshape(1,-1)
            else:
                  return np.concatenate((self.acx, self.acy,
self.acz, self.gx, self.gy, self.gz))


      @staticmethod
      def load_from_file(filename, size_fit = 50):
            '''
            Loads the signal data from a file.

            filename: indicates the path of the file.
            size_fit: is the final number of sample an axe will have.
                        It uses linear interpolation to increase or
decrease
                        the number of samples.

            '''
            #Load the signal data from the file as a list
            #It skips the first and the last line and converts each
number into an int
```

```
            data_raw = [map(lambda x: float(x), i.split(" ")[1:-1])
for i in open(filename)]

            #Convert the data into floats
            data = np.array(data_raw)

            #Standardize the data by scaling it
            data_norm = scale(data)

            #Extract each axe into a separate variable
            #These rapresent the acceleration in the 3 axes
            acx = data_norm[:,0]
            acy = data_norm[:,1]
            acz = data_norm[:,2]

            #These rapresent the rotation in the 3 axes
            gx = data_norm[:,3]
            gy = data_norm[:,4]
            gz = data_norm[:,5]

            #Create a function for each axe that interpolates the
samples
            x = np.linspace(0, data.shape[0], data.shape[0])
            f_acx = interp1d(x, acx)
            f_acy = interp1d(x, acy)
            f_acz = interp1d(x, acz)

            f_gx = interp1d(x, gx)
            f_gy = interp1d(x, gy)
            f_gz = interp1d(x, gz)

            #Create a new sample set with the desired sample size by
rescaling
            #the original one
            xnew = np.linspace(0, data.shape[0], size_fit)
            acx_stretch = f_acx(xnew)
            acy_stretch = f_acy(xnew)
            acz_stretch = f_acz(xnew)

            gx_stretch = f_gx(xnew)
            gy_stretch = f_gy(xnew)
            gz_stretch = f_gz(xnew)

            #Returns a Sample with the calculated values
            return Sample(acx_stretch, acy_stretch, acz_stretch,
gx_stretch, gy_stretch, gz_stretch)
```

- **suggestions.py**

```
import os

'''
This library contains the classes needed to manipulate words and
obtain suggestions
'''

class Hinter:
	'''
	Hinter is used to load a dictionary and obtain some suggestions
	regarding the next possible letters or compatible words.
	'''
	def __init__(self, words):
		self.words = words

	@staticmethod
	def load_english_dict():
		'''
		Loads the english dictionary and returns a Hinter object
with
		the words loaded into the self.words list
		'''
		ENGLISH_FILENAME = "dict" + os.sep + "english.txt"
		words = [i.replace("\n","") for i in
open(ENGLISH_FILENAME)]
		return Hinter(words)

	def compatible_words(self, word, limit = 10):
		'''
		Returns the words that starts with the "word" parameter.
		The "limit" parameter defines how many words the function
		returns at maximum
		'''
		output = []
		word_count = 0
		#Cycle through all the words to find the ones that starts
with "word"
		for i in self.words:
			if i.startswith(word):
				output.append(i)
				word_count+=1
			if word_count>=limit: #If limit is reached, exit
				break
		return output

	def next_letters(self, word):
		'''
		Returns a list of compatible letters.
		A letter is compatible when there are words that starts
with "word"
		and are followed by the letter.
		'''
		#Get 100 compatible words
		words = self.compatible_words(word, 100)
		letters = []
```

```
            #Cycle through all the compatible words
            for i in words:
                  if len(i)>len(word): #if the "word" is longer than
a compatible word, skip
                        letter = i[len(word):len(word)+1] #Get the
following letter
                        if not letter in letters: #Avoid duplicates
                              letters.append(letter)
            return letters

      def does_word_exists(self, word):
            '''
            Check if a specific word exists in the loaded dictionary
            '''
            if word in self.words:
                  return True
            else:
                  return False

      def most_probable_letter(self, clf, classes, linearized_sample,
word):
            '''
            Get the most probable letter for a given recorded sign
and the current word
            '''
            if word=="":
                  return None

            probabilities = clf.predict_log_proba(linearized_sample)
            ordered = sorted(classes)
            values = {}
            for i in xrange(len(probabilities[0])):
                  values[round(probabilities[0,i], 5)] =
classes[ordered[i]]
            ordered = sorted(values, reverse=True)
            possible_letters = self.next_letters(word)
            for i in ordered:
                  if values[i] in possible_letters:
                        return values[i]
            return None
```

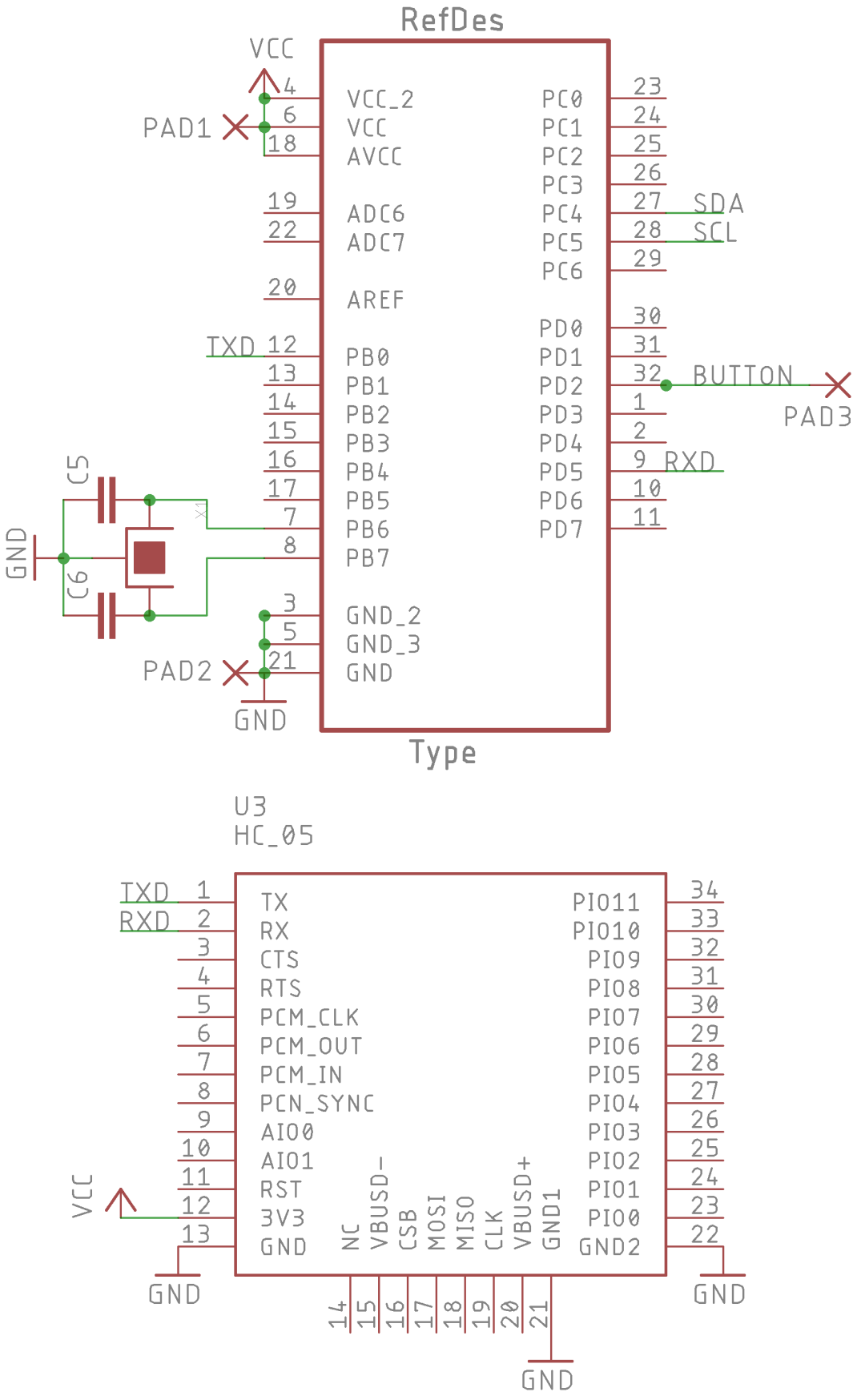
- **window.py**

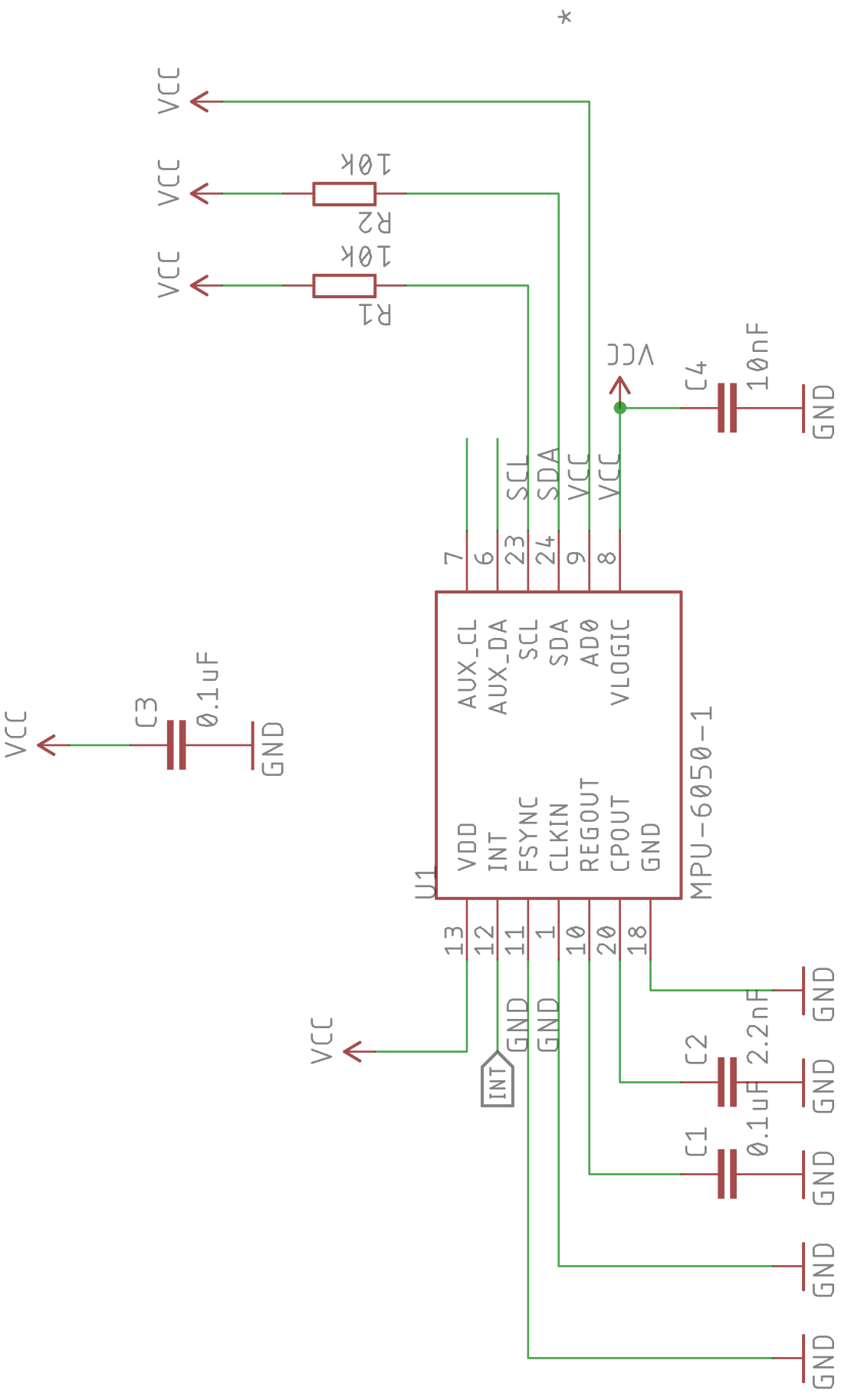
```
from Tkinter import *

'''
This module just display a basic window to show the output of the
"start.py" module
'''

class TextWindow:

    def __init__(self, master):
      self.root = master

        frame = Frame(master)
        frame.pack()
        master.title("Gesture Keyboard Output")

        self.label = Label(master, text="Connecting...",
font=("Helvetica", 80))
        self.label.configure(wraplength=600)
        self.label.pack()

        self.update_clock()

    def update_clock(self):
        input_file = open("output.txt")
        self.label.configure(text=input_file.read())
        input_file.close()
        self.root.after(100, self.update_clock)

root = Tk()

app = TextWindow(root)

root.mainloop()
root.destroy()
```

ANNEXURE B

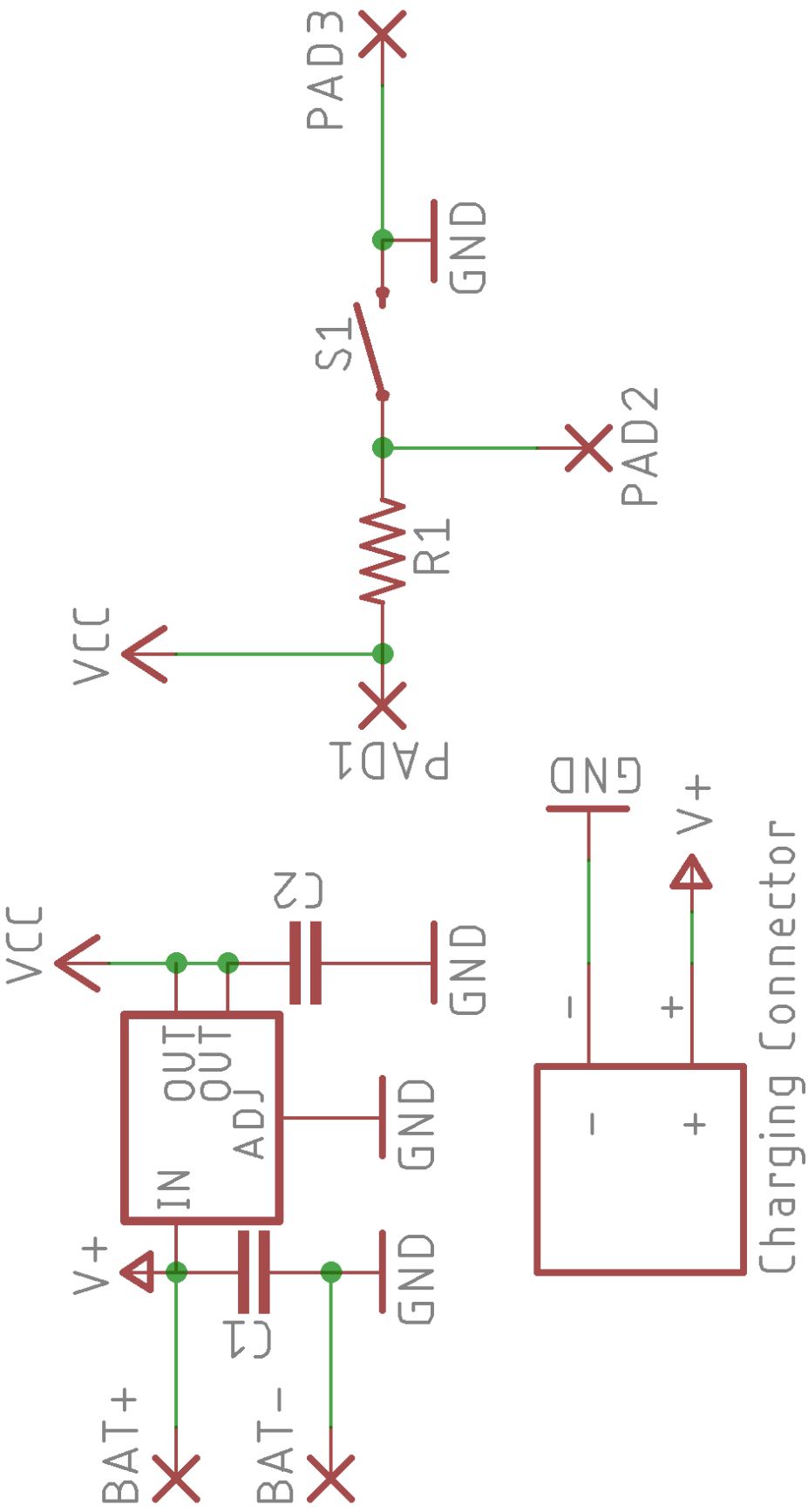
SCHEMATICS

Schematic for Ring-O Main PCB



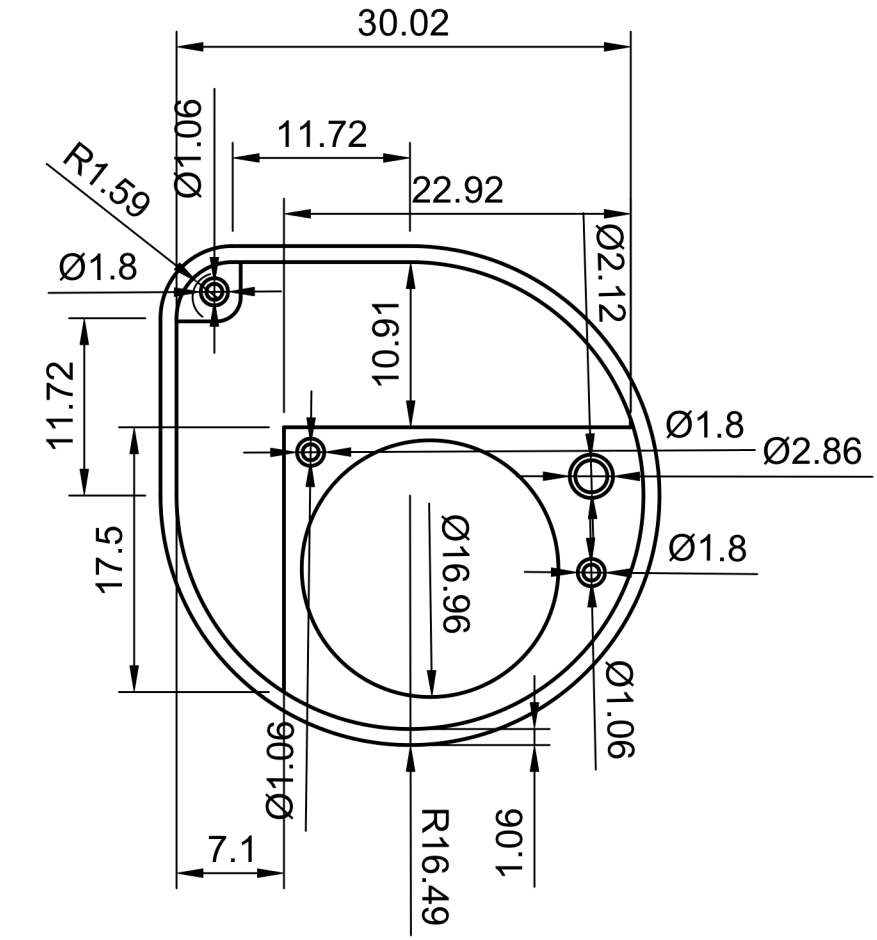

VCC
10K
R2
10K
R1
VCC
C4
10nF
GND
7
6
23
24
9
8
AUX_CL
AUX_DA
SCL
SDA
AD0
VLOGIC
SCL
SDA
VCC
VCC
C3
0.1uF
GND
U1
VDD
INT
FSYNC
CLKIN
REGOUT
CPOUT
GND
MPU-6050-1
13
12
11
1
10
20
18
VCC
INT
GND
GND
C2
2.2nF
C1
0.1uF
GND
GND
GND
GND
GND

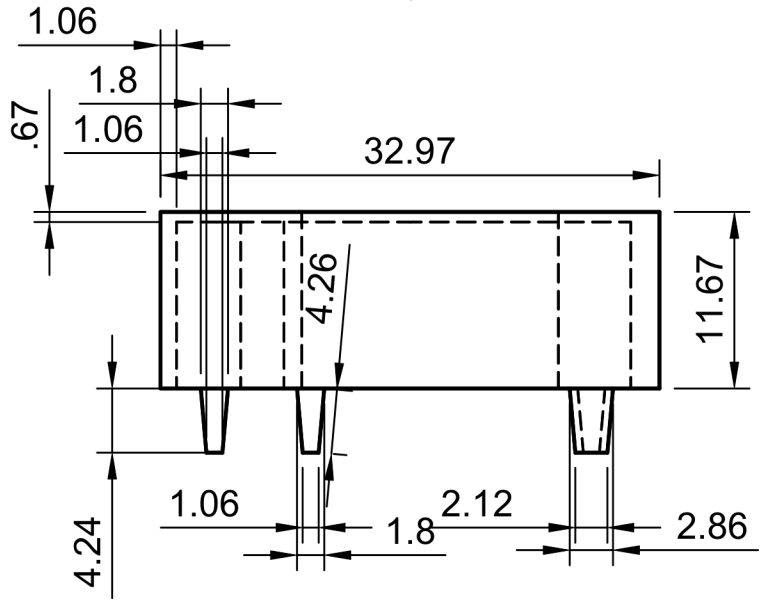
Schematic for Ring-O Aux PCB



ANNEXURE C

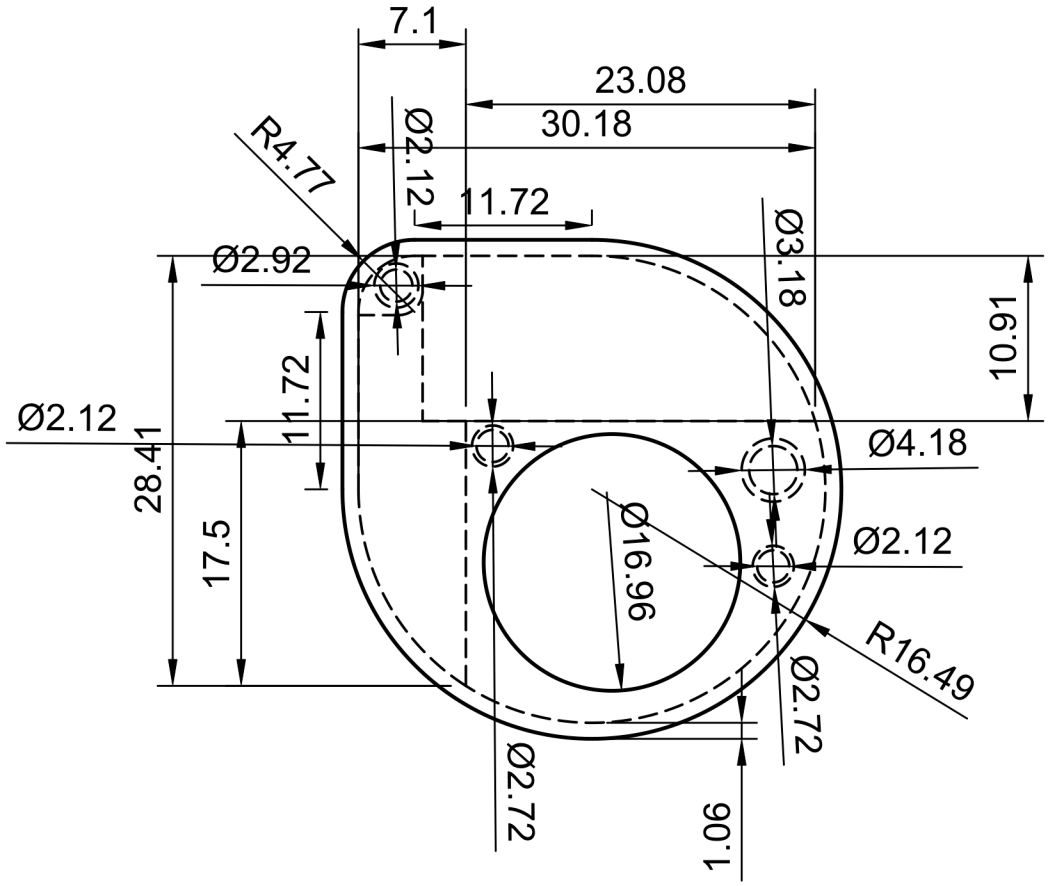
CAD DRAWINGS

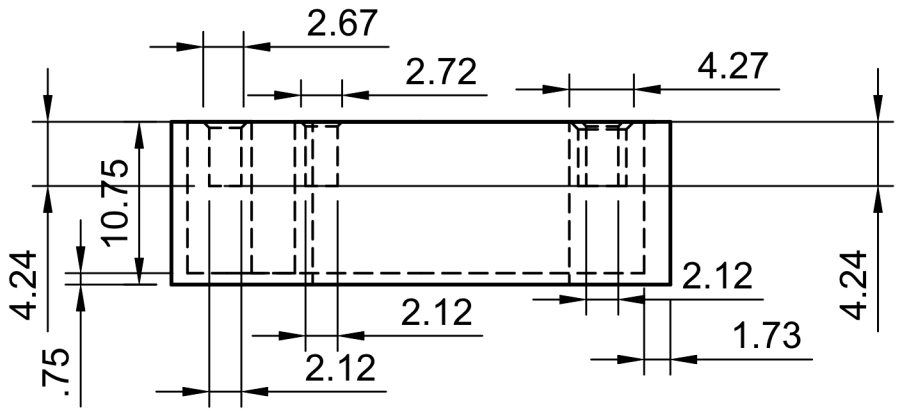




***all dimesnions are in mm**

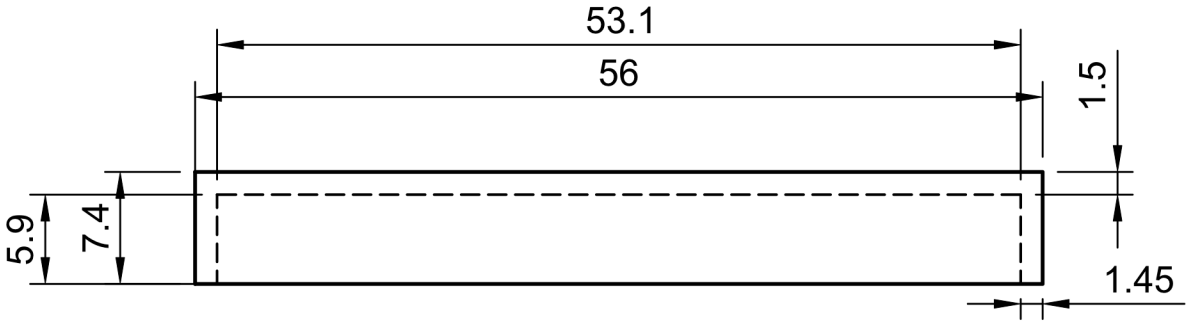
Dept.	Technical reference	Created by	Approved by
ECE		Akshay Baweja	Mr. Mukesh Sahu
		Document type: Major Project Drawing	Document status: Passed
		Title: Ring-O Ring Module Male	DWG No.: GTBIT/2018/ECE/1/07/001
			Rev.: 1 | Date of issue: 15/04/2018 | Sheet: 1/2

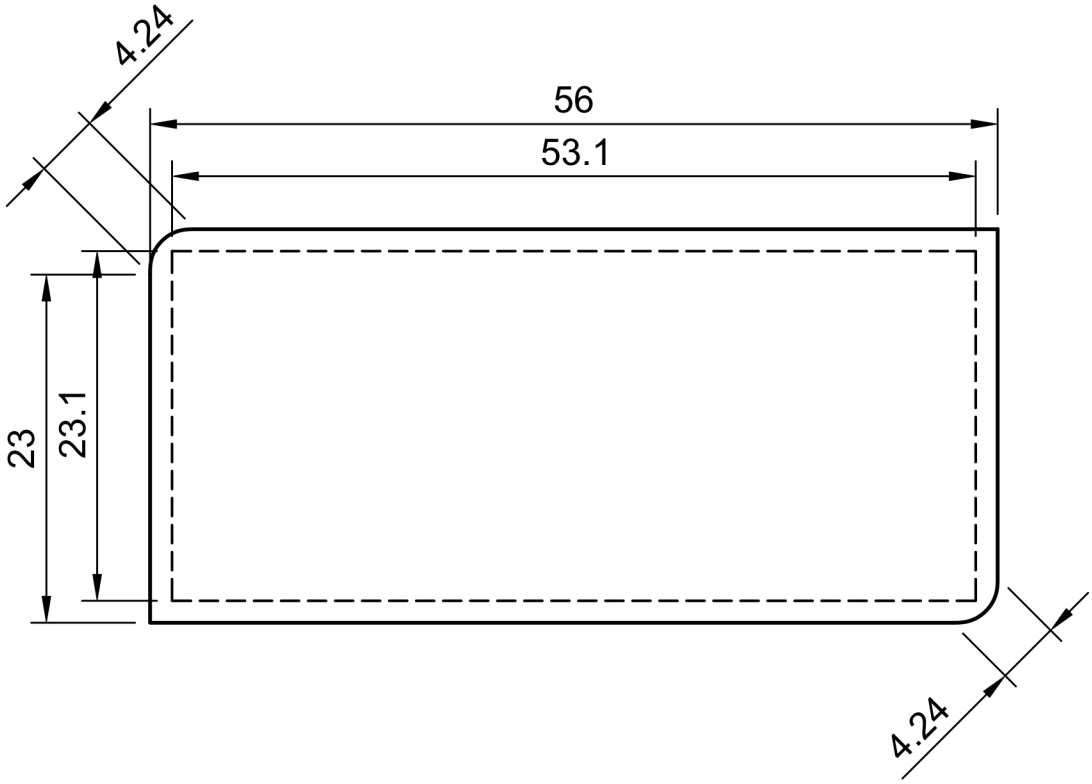




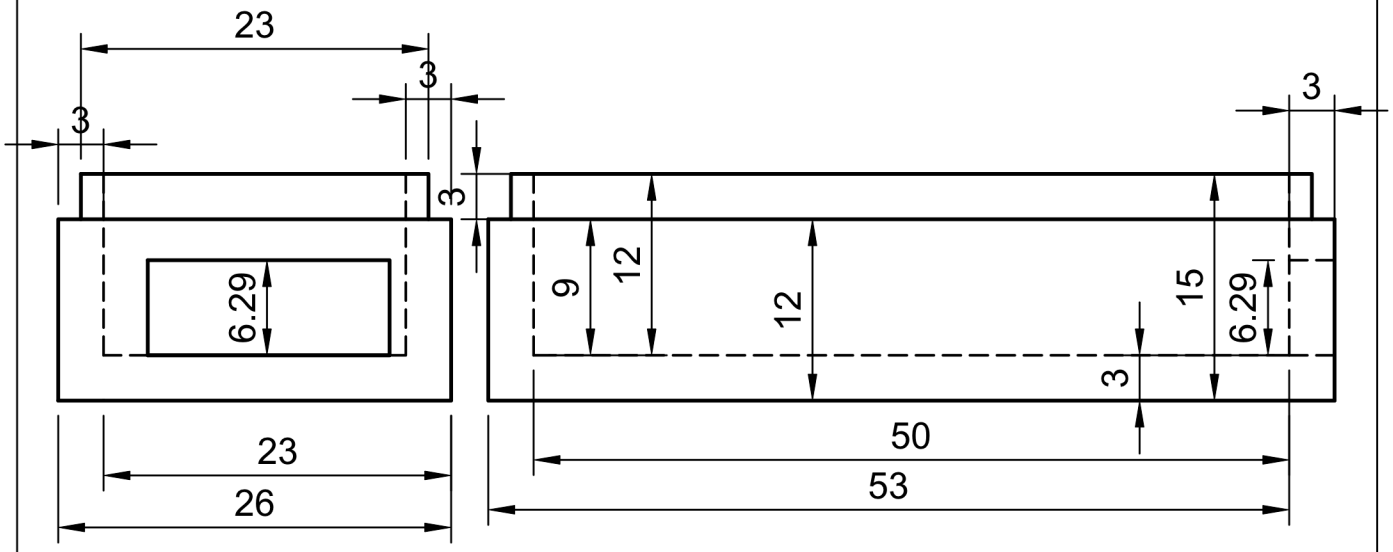
***all dimesnions are in mm**

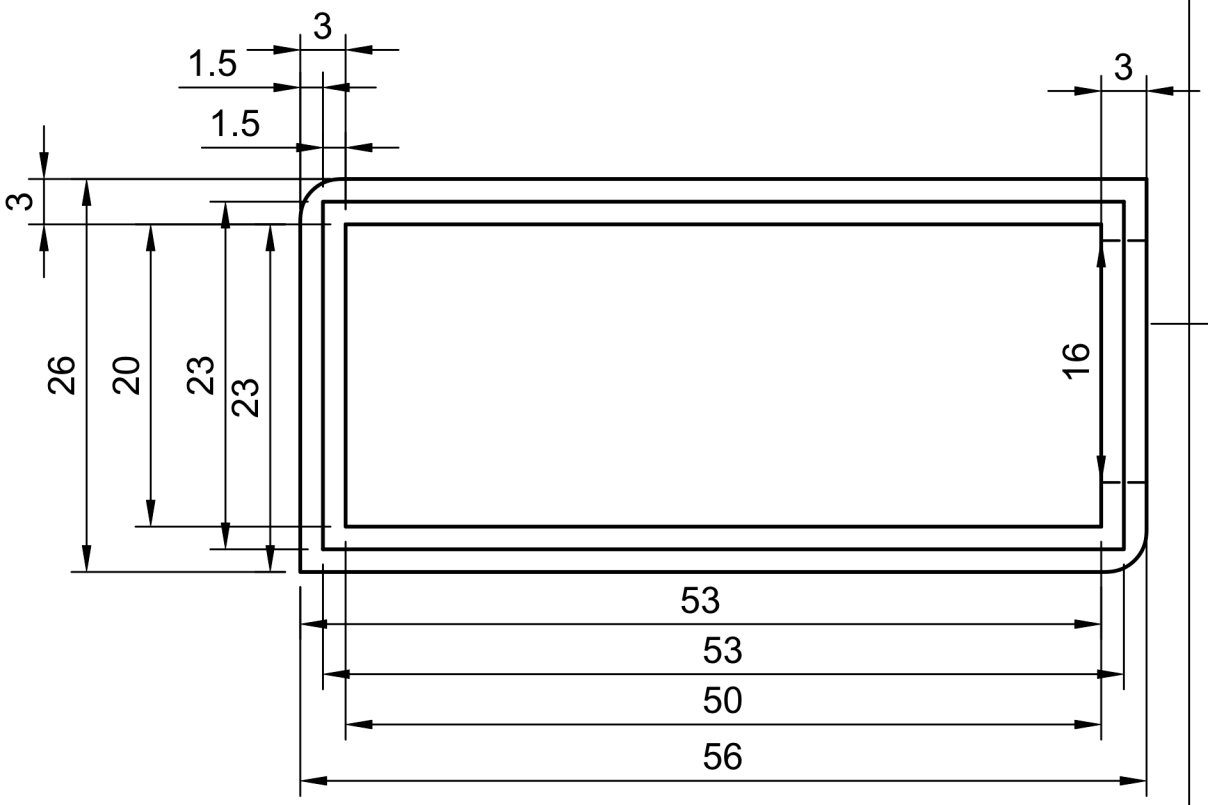
Dept.	Technical reference	Created by	Approved by
ECE		Akshay Baweja	Mr. Mukesh Sahu
		Document type: Major Project Drawing	Document status: Passed
		Title: Ring-O Ring Female	DWG No.: GTBIT/2018/ECE/1/07/002
			Rev.: 1 | Date of issue: 15/04/2018 | Sheet: 2/2





Dept.	Technical reference	Created by	Approved by
ECE		Akshay Baweja	Mr. Mukesh Sahu
		Document type: Major Project Drawing	Document status: Passed
		Title: Ring-O Bluetooth Reciever Box Top	DWG No.: GTBIT/2018/ECE/1/07/003
			Rev.: 1 | Date of issue: 15/04/2018 | Sheet: 1/2





*all dimensions are in mm

Dept.	Technical reference	Created by	Approved by
ECE		Akshay Baweja	Mr. Mukesh Sahu
		Document type: Major Project Drawing	Document status:
		Title: Ring-O Bluetooth Reciever Box Bottom	DWG No.: GTBIT/2018/ECE/1/07/004
			Rev.: 1 | Date of issue: 15/04/2018 | Sheet: 2/2