IOT
ch-10
MOVING TO MANUFACTURE

1) What are the different possibilities that we should consider before producing a Product. Also Explain Designing KITS in detail.

- Depending on your motives and desires for the product, and the amount of time and money that you have available to devote to it, a full spectrum of possibilities is open to you.
- If your ambition is just to enable others to build a version of it for themselves, you only need to document the build process and share it, along with any source code and design files you used, on the Internet—either on your own blog or website, or on a site like Instructables.
- When you decide that you want to get your invention into the hands of many more people way), you can generally split your choices into three categories:
- 1 ▪ A kit that your customers can assemble themselves
- 2 ▪ A ready-made electronics board which users can use as a sub-assembly in their own projects
- 3 ▪ A fully fledged product, complete with its housing, instructions, and even packaging, just waiting to be put on the shelf at your local department store.

- **DESIGNING KITS**
- Many of those people would jump at the chance to buy a kit in which someone else has done the hard (to them) part of selecting and sourcing the right components and putting them together with a step-by-step guide.
- Most kits tend to provide only the electronics and software for a particular application rather than any physical housing.
- Kits tend to piggyback on existing microcontrollers and often take the form of a standard format plug-on board—for example, a *shield* in the Arduino.
- Given that you've built your prototype, you've already done most of the work needed to create a kit.
- You've worked out which components you need, where to source them, and how to wire them up to create a functioning circuit.
- And you've written the necessary software to interface to and control the electronics.
- The parts you might not have done include designing a PCB, documenting the build process, and working out the costs.
- You can make the PCBs a few different ways after they are designed.
- Documenting the build process isn't too tricky.
- When you have everything together for one of your kits, run through assembling one of them yourself.
- As you go, take photos or video of each step and write down the order of each step.
- Working out what price to charge is harder.
- Some rules of thumb can help, though.
- The most obvious is to understand what your costs are.
- You should create a spreadsheet (or at least a list) of all the items that make up your product, along with their cost to you to buy.
- You should list every single electronic component, connector, cable, PCB, case, and so on, in addition to the packing box you'll ship it in and the time taken to put things together.

- This list is called the *bill of materials* (BOM), and it forms the starting point for all your costings and prices.
- A good starting point for working out your price (how much you will charge consumers) is to take the total cost of the BOM and multiply it by 4 or 5.
- That calculation gives you a margin to cover that item's portion of the fixed costs and also some profit.
- It also provides enough margin for resellers to cover *their* fixed costs and make some profit.
- The final step from kit to consumer product is to manufacture and assemble the housings, linkages, and whatever else is part of the finished device.

Q2) How to design PRINTED CIRCUIT BOARDS? Explain software choices for it.

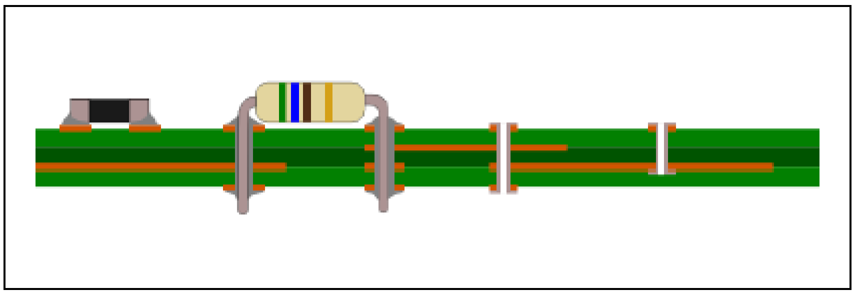
- Soldering things up is a good step towards making your prototype more robust.
- After you've done this, you should have something that will survive being given to an end user.
- You will soon get fed up with soldering each item by hand.
- Moving beyond stripboard, designing and etching your own custom PCBs gives you more options on how to lay out the circuit and makes it easier to solder up as the only holes in the board will be those for components.
- Homemade boards will still lack that fully professional finish.
- That's because they won't have the green solder mask or silkscreen layers that a PCB manufacturer will give you.
- Moving to professionally manufactured boards further simplifies the assembly process because the solder mask will make the soldering a bit easier, and, more importantly, the silkscreen provides outlines of where each component should be placed.

- SOFTWARE CHOICES
- You have many different choices when looking for some software to help you design your PCB.
- If you are working with a contract electronics design house, the staff may well use something like Altium Designer.
- But you're more likely to use one of the following lower-end (and cheaper) options.
- Fritzing (http://fritzing.org) is a free, open source design package aimed particularly at beginners in PCB design.
- It deliberately starts with a design screen resembling a breadboard and lets you map out your circuit by copying whatever you have prototyped in real life.
- It then converts that design to a schematic circuit diagram and lets you arrange components and route the traces on the PCB view.
- When you are happy with your design, you can export it for manufacture as a PCB.
- Fritzing even offers a fabrication service to make it simple to make your design a reality.
- KiCad (www.kicad-pcb.org) is another open source offering but with a more traditional workflow.
- It has a more comprehensive library of predefined parts and can be used to design boards with up to 16 layers of copper, compared to the double-sided boards that Fritzing produces.
- In addition to letting you see what the finished board will look like, it also enables you to export the 3D model. Doing so enables you to import it into your CAD software when designing the enclosure to ensure that the relevant clearances are allowed for all the components.

- Probably the most popular PCB layout software for the hobbyist and semi-professional market is EAGLE from CadSoft.
- The reason for its popularity most likely comes down to its long having a free version for non-commercial use, allowing beginners to get started.
- Although in addition to its noncommercial licence, the free version is also restricted to two layers and a maximum board size of 100mm × 80mm.
- A more recent rival to EAGLE in the commercial pro-sumer market is DesignSpark PCB.
- Unlike EAGLE, DesignSpark PCB does not restrict the size of boards you can design (up to 1m2) or the number of layers (it supports up to 14 layers).
- However, it is the only program described here which isn't available for Linux or Mac.

Q3)What makes up a PCB?

- The PCB is made up of a number of layers of fibreglass and copper, sandwiched together into the board.
- The fibreglass provides the main basis for the board but also acts as an insulator between the different layers of copper, and the copper forms the "wires" to connect the components in the circuit together.
- Given that you won't want to connect all the components to each other at the same time, which would happen if you had a solid plate of copper across the whole board, sections of the copper are etched away—usually chemically,
- But it is possible to use a CNC mill for simple boards.
- These remaining copper routes are called *tracks* or *traces* and make the required connections between the components.
- The points on the tracks where they join the leg of a component are known as *pads*.
- Single-sided boards have only one layer of copper, usually on the bottom of the board; because they're often for home-made circuits with through-hole components, the components go on the top with their legs poking through the board and soldered on the underside.
- Double-sided boards have two layers of copper: one on the top and one on the bottom.
- More complicated circuits may need even more layers to allow room to route all the traces round to where they are needed.
- Additional layers require a more complex manufacturing procedure in which alternating layers of copper and fibreglass are built up, a bit like you would make a sandwich.
- When the holes drilled through the board are *plated*—a process in which the walls of the holes are coated in
- A thin layer of copper—any layers with copper at that point are connected together.
- When you need to connect traces on two layers together at a point where there isn't a hole for the leg of a component, you use a *via*.
- It is smaller, hole through the board purely to connect different layers of copper once plated.
- You also can use blind vias, which don't pierce all the way through the board , when you don't want to connect every layer together.

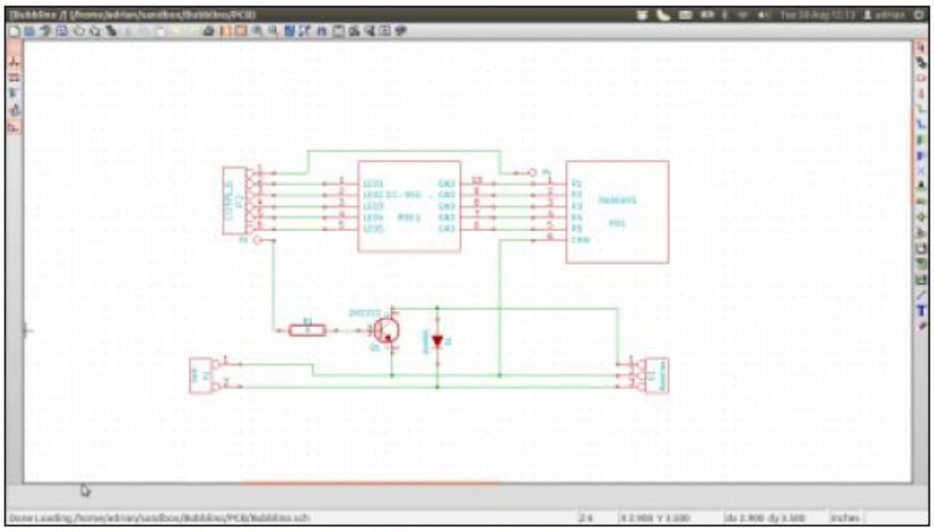

PCB features (left to right): surface-mount component; through-hole component; via; blind via.

-
- In places where you have many connections to a common point, rather than run lots of tracks across the circuit, you can more easily devote most of a layer of the board to that connection and just leave it as a filled area of copper.
- This is known as a *plane* rather than a trace and is frequently used to provide a route to ground.

- The surfaces of professionally manufactured PCBs undergo processes to apply two other finishes which make them easier to use.
- First, all the parts of the board and bare copper which aren't the places where component legs need to be soldered are covered in solder mask.
- Solder mask is most commonly green.
- The mask provides a solder-resistant area, encouraging the solder to flow away from those areas and to adhere instead to the places where it is needed to connect the components to the tracks.
- This reduces the likelihood of a solder joint accidentally bridging across two points in the circuit where it shouldn't.
- Then, on top of the solder mask is the silkscreen.
- This is a surface finish of paint applied, as the name suggests, via silkscreen printing.
- It is used to mark out where the components go and label the positions for easy identification of parts.
- It generally also includes details such as the company or person who designed the board, a name or label to describe what it is for, and the date of manufacture or revision number of the design.

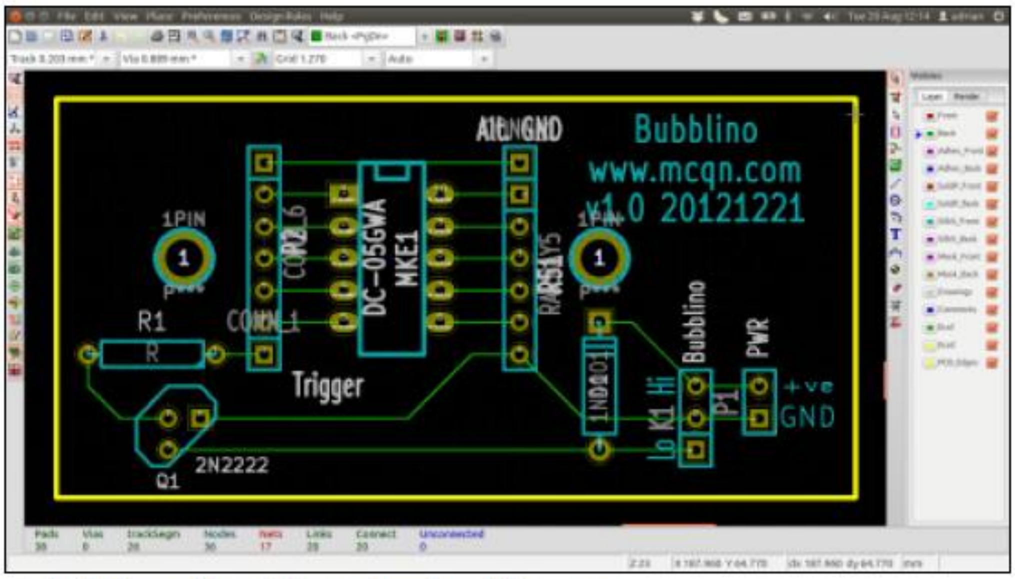
Q4) Explain THE DESIGN PROCESS for PCB.

- Regardless of the exact program you decide to use to lay out your PCB, your task in creating the design is split between two main views: the schematic and the board.
- **The Schematic**
- You usually start the design in the schematic view. It lets you lay out the components logically and make the necessary connections without having to worry about exactly where they'll sit in physical space or whether any of the tracks cross.
- The schematic provides a conceptual view of how the circuit is laid out.
- Components are represented by their standardised symbols, and you usually group them into areas of common functionality instead, which won't necessarily match the groupings they'll have in the physical layout.
- Your software package includes most of the common items you need in its library: common resistors, diodes, capacitors, transistors, integrated circuits (ICs), and more.


The schematic view of the Bubblino PCB, as designed in KiCad.

-
- You should also make sure you choose the correct *package* for the component; this is the physical format for it. Many parts are available in a range of different formats, depending on the target application.
- Making the right choices when initially choosing the component will save you having to rework that down the line
- When you come to the board view.
- When you are happy with the schematic design of your board, you can proceed to laying out the physical board.
- **The Board**
- There are no hard-and-fast rules on how to arrange the board.
- It makes sense to keep together groups of components that constitute an area of functionality, such as the power supply, for a more logical layout.
- The design has certain fixed aspects—things such as connectors which all need to be along one side for access once in a case, for example, or arrangements of headers to mate with an Arduino board.
- Start by placing the components which have the most connections first and then fitting the remainder around those constraints.
- As you position the components, you should try to reduce how tangled they are (rotating or moving them around for the best combination).
- The thin, straight-line connections from point-to-point are known as *air wires*.
- They show where the connections need to be made but won't appear in the finished PCB design until you turn them into real tracks on the PCB.
- Each one needs to be turned into a track on the PCB and routed round the board so that it makes the connection without crossing any of the other tracks or running too close to the pads of other components.
- By finding enough room for the tracks and stopping them from crossing, you can make the different layers of the PCB come into play.
- Placing tracks on different layers of the board means that they won't make electrical contact when they cross, and for more complex designs you might run one track on one layer for part of its path and switch it to another layer with a via to continue the rest of its route.
- Your PCB software has an auto-route function, which you can use to route all the tracks for you.
- But in practice, you will find it best to at least lay out the more important tracks first by hand.
- After all the tracks have been routed, your PCB design is almost finished.
- You should add any labels to the silkscreen layer to explain things.

- Also include a version number so that you can differentiate any future revisions, and also add maybe your name or the company name and a website for people to find out more.



The finished view of the Bubblino PCB board, ready for export.

- Then give your PCB a final once-over.
- Run the design rules check to catch any missed connections or problems where tracks are too close to each other or too thin for your PCB manufacturer to reliably manufacture.
- The design rules effectively contain the manufacturing tolerances for your PCB manufacturer.
- They define things like the minimum track width or the minimum distance between pads.
- Then print out a copy on paper; this way, you can compare the components to their location on the PCB in real life and spot any errors in their footprints.
- After you fix any issues thrown up by those last checks, you're ready to make the PCBs.

Q5)List and explain possible ways of Manufacturing PCB.

- If you want only a couple of boards, or you would like to test a couple of boards (a very wise move) before ordering a few hundred or a few thousand, you may decide to make them in-house.
- **ETCHING BOARDS**
- The most common PCB-making technique for home use is to etch the board.
- Some readily available kits provide all you need.
- The first step is to get the PCB design onto the board to be etched.
- This process generally involves printing out the design from your PCB design software onto a stencil.
- If you're using photo-resist board, it will be onto a stencil which masks off the relevant areas when you expose it to UV light.
- Your stencil then needs to be transferred to the board.
- For photo-resist board, you will expose it under a bright lamp for a few minutes.
- With the board suitably prepared, you can immerse it into the etching solution, where its acidic make-up eats away the exposed copper, leaving the tracks behind.
- After all the unnecessary copper has been etched away, and you've removed the board from the etching bath and cleaned off any remaining etchant, your board is almost ready for use.

- The last step is to drill the holes for any mounting points or through-hole components.
- **MILLING BOARDS**
- In addition to using a CNC mill to drill the holes in your PCB, you can also use it to route out the copper from around the tracks themselves.
- To do this, you need to export the copper layers from your PCB software as Gerber files.
- These were first defined by Gerber Systems Corp., hence the name, and are now the industry standard format used to describe PCBs in manufacture.
- To translate your Gerber file into the G-code that your mill needs requires another piece of software.
- Some CNC mills come with that software already provided, or you can use a third-party program such as Line Grinder.
- The mill effectively cuts a path round the perimeter of each track to isolate it from the rest of the copper.
- As a result, PCBs which have been milled look a bit different from those which are etched because any large areas of copper that aren't connected to anything are left on the board.
- **THIRD-PARTY MANUFACTURING**
- If your design has more than two layers, if you want a more professional finish, or if you just don't want to go to the trouble of making the PCBs yourself, many companies can manufacture the boards for you.
- The price for getting the boards made varies based on the complexity and the size of the design but also varies quite a bit from company to company.
- If you need the boards quickly, a local firm is best.
- If you have more time you can give it outside country such as china, it might reduce cost.
- Either way, the Gerber files are what you need to provide to the manufacturer.
- Make sure you export all the relevant layers from your design, meaning each of the copper layers you're using, plus the solder mask, silkscreen and drill files.

Q6)Explain the concept of ASSEMBLY and testing with respect to PCBs.

- **ASSEMBLY**
- After your PCBs have been manufactured, you still need to get the components soldered onto them.
- If you're selling them as kits, the customers will solder things up, so you just need to pack everything into bags and let them get on with it.
- Otherwise, you have to take responsibility for making that happen.
- For small runs, you can solder them by hand.
- For through-hole boards, break out your soldering iron.
- For assembling surface-mount boards, you need one more item from your PCB design Gerber collection: the solder paste layer.
- You use it to generate a stencil that allows you to apply the solder.
- You can laser-cut one from a thin sheet of Mylar plastic or have one made for you out of thin steel.
- When you have all the components on the board, you need to melt the solder to fix everything in place.
- You can do this with a soldering iron, but doing it by hand is easier if you use a hot-air rework station.
- It uses hot air to provide the necessary heat.
- You can solder all the connections at once if you use a reflow oven.

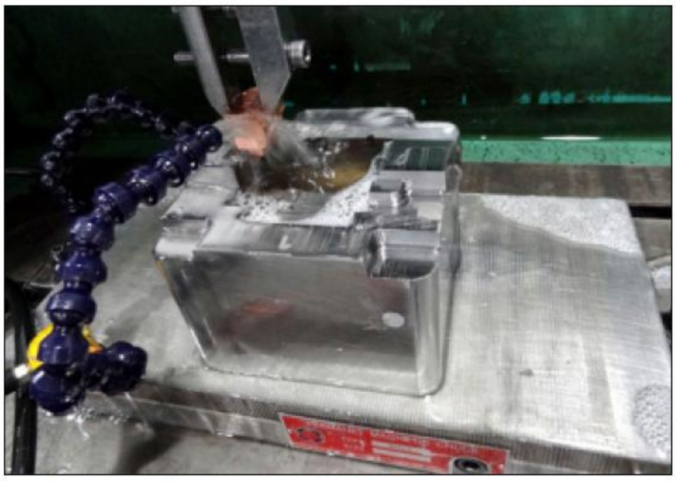
- As the name suggests, this oven heats up the PCB and components evenly until the solder melts.
- Professional reflow ovens allow you to set different temperature profiles, allowing you to match the specification in the manufacturers' datasheets for more exacting components.
- After you outgrow hand assembly, you will need some help from robots.
- In this case, you will need robots that can pick up components using a tiny vacuum nozzle, rotate and place them in the right location on the PCB, and then repeat that process at a rate of tens of thousands of components per hour. These robots are known as *pick-and-place assembly machines*.
- It is worth to offload some of the work to an *assembly house*, also known as a *contract manufacturer*.
- Contract manufacturers are firms geared up to helping people produce finished, populated PCBs.
- Often they offer a range of services from PCB design, through dealing with PCB manufacturers, to soldering up the components and even testing the completed boards
- Using an assembly house saves you from buying the expensive machinery yourself.
- If you do decide to use a contract manufacturer, having a conversation about components is worthwhile.
- For common parts, if you don't have specific requirements for tolerances, and the like, you might be able to specify that the assembler should use the parts it already holds in stock.
- That might mean you don't have to buy an entire reel just to get a few components.
- Even for the parts the contract manufacturer doesn't already carry and that you need to buy in, it may make sense to work with the manufacturer to place the order.
- If the manufacturer has dealt with the supplier before, its reputation might let you negotiate a better deal.

- **TESTING**
- Now your boards are all ready and assembled, but how do you know that they all work as they're meant to?
- This is where testing comes in.
- A better approach is to build a specific test rig to exercise the different parts of the circuit and measure the voltages at set points on the board.
- These measurements can then be compared against known-good values and a decision made automatically as to whether or not the device under test (DUT) has passed.
- Because you don't want to spend time making individual connections for each test, the normal practice for the test rig is to use the mounting holes for the PCB for alignment and then have it held by some clips against a number of carefully prepositioned, spring-loaded pins.
- These pins are known as *pogo pins*, and the spring means they can make a good connection to the board without any extra work, such as soldering, when the board is placed into the test rig.
- The test program can then run through its tests and measure voltages at different pogo pins at the relevant time in the test to see how the board being tested performs.

- If the DUT includes a microcontroller chip, the test rig could flash it with a test program to help with the testing, and even at the end of the test, assuming it has passed, flash it with the final firmware image ready for deployment.

Q7) How to go for MASS-PRODUCING THE CASE AND OTHER FIXTURES

- A good rule of thumb for keeping down the costs of production is to minimise the amount of time a person has to work on each item.
- Machines tend to be cheaper than people, and the smaller the proportion of labour is in your costs, the more you'll be able to afford to pay a decent wage to the people who *are* involved in assembling your devices.
- Production rates are also important.
- Though they're fairly labour free, 3D printers and laser cutters aren't the fastest of production techniques.
- Waiting a couple of hours for a print is fine if you just want one, but a production run of a thousand is either going to take a very long time or require a *lot* of 3D printers.
- We look at what must be the most common method of mass production: injection moulding of plastic.
- As the name suggests, the process involves injecting molten plastic into a mould to form the desired shape.
- After the plastic has cooled sufficiently, the mould is separated and the part is popped out by a number of ejection pins and falls into a collection bin.
- The whole cycle is automated and takes much less time than a 3D print, which means that thousands of parts can be easily churned out at a low cost per part.
- The expensive part of injection moulding is producing the mould in the first place; this is known as *tooling up*.
- Often for a super-smooth surface, the moulds are finished with a process called *electrical discharge machining* (EDM), which uses high-voltage sparks to vaporise the surface of the metal and gives a highly polished result.
- The moulds are machined out of steel or aluminium and must be carefully designed and polished so that the moulding process works well and the parts come out with the desired surface finish.
- The mould also needs to include space for the ejection pins to remove the part after it's made and a route for the plastic to flow into the mould.
- Because the parts need to be extracted from the mould after they're formed, very sharp corners and completely vertical faces are best avoided.
- A slight angle, called the *draft*, from vertical allows for clean parting of the part and its mould, and consistent wall thicknesses avoid warping of the part as it cools.
- The simplest moulds are called *straight-pull* and consist of the mould split into two halves.
- If your design needs to include vertical faces or complex overhangs, more complicated moulds which bring in additional pieces from the side are possible but add to the tooling-up cost.
- One way to reduce the tooling-up costs and also increase the production rate is to mould more than one part at a time.
- In a process known as *multishot moulding*, you can even share parts of different colours on the same mould.
- With carefully measured volumes for each part, one of the colours of plastic is injected first to fill the parts which need to be that colour.
- Then the other colour is injected to fill the remainder of the mould.


One of the moulds for BERG's Little Printer being finished on an EDM machine.

Q8)Explain Manufacturing Process with case study of : BERG's little printer

- The Little Printer, made by London design firm BERG, is a delightful, tiny Internet connected Printer.
- In 2006, Matt Webb, one of the founders of BERG, shared some thinking-out-loud about how the printer connected to his computer could be shared with close friends and family regardless of where they are in the world (http://berglondon.
- com/blog/2006/10/06/my-printer-my-social-letterbox/).
- He called this a social letterbox, with the idea being that people he was close to socially would be able to share printed things with him even when they weren't close to him physically.
- In the comments to his initial blog post, there's a remark from a certain Adrian McEwen, suggesting that a better approach than repurposing the printer connected to your computer would be to give the printer its own network connection and intelligence.
- This initial contemplation led to a number of people, including Matt, some of his friends, and the wider maker community, experimenting with repurposed receipt printers hooked up to Arduino boards and connected to the Internet.
- They experimented with printing out the weather forecast, their appointments for the day, tweets from friends, and much more.
- From those early investigations, they worked through what made sense when printed out and what did not.
- These experiments continued over a few years, operating almost as a background project for BERG as they engaged in a combination of client work and self-directed products.
- To bring Little Printer into existence the physical form required careful design of the printer mechanism and tooling for both injection moulding and metal pressing to provide the housing.
- The team ran into issues with the way that the injection-moulded plastic cooled around the holes for the button and status light on the top of the case: it introduced a ripple on the surface.
- Solving this problem resulted in a two-stage process for that part of the case. It is now injection-moulded without the holes, which gives the correct finish, and then drilled to provide the holes.
- On the electronics and software side, they wanted to minimise the amount of setup required and to be able to control the look and feel of the whole user experience, from ordering the printer through to using it in the home.

- This led them to choose a ZigBee-based wireless option which didn't require any username or password to pair with other endpoints.
- However, while home routers support WiFi, they don't have ZigBee support built in.
- This dictated a two-box solution: in addition to your Little Printer, you receive a BERG Cloud Bridge, a box with both ZigBee wireless and Ethernet which you plug into an Ethernet port on your network to talk to the Internet.
- That allows the printer to communicate with the BERG Cloud server software, which provides for both the delivery of "publications" to the printers and the user-facing website.
- Through the user interface of their website, BERG allows users to register their Little Printer and to choose which publications they want to subscribe to and when they should be delivered.
- Details of all this work broke cover in November 2011.
- By August 2012, things were nearing completion, and they had started taking pre-orders.
- A combination of the switching frequency of their switch-mode power supply and the PCB design for the Bridge units conspired to make it fail the electromagnetic compatibility emissions test.
- Solving that problem caused the schedule to slip by more than a month, and then a smaller issue in assembly with the magnetic catches on the case delayed things by another week or so.
- At the end of November 2012, these playful Little Printers started winging their way out to customers, bringing more Internet of Things goodness into numerous homes.

Q9) Discuss CERTIFICATION issues with IOT products.

- One of the less obvious sides of creating an Internet of Things product is the issue of certification.
- These regulations are there for good reason.
- They make the products you use day in, day out, safer for you to use; make sure that they work properly with complementary products from other suppliers; and ensure that one product doesn't emit lots of unwanted electromagnetic radiation and interfere with the correct operation of other devices nearby.
- The regulations that your device needs to pass vary depending on its exact functionality, target market (consumer, industrial, and so on), and the countries in which you expect to sell it.
- The best approach is to work with a local testing facility.
- They not only are able to perform the tests for you but also are able to advise on which sets of regulations your device falls under and how they vary from country to country.
- Such a testing facility subjects your device to lots of tests.
- Testers check over the materials specifications to ensure you're not using paint containing lead; zap it with an 8KV static shock of electricity to see how it copes; subject it to probing with a hot wire—heated to 500 degrees Celsius—to check that it doesn't go up in flames; and much more.
- Of particular interest is the electromagnetic compatibility, or EMC, testing.
- This tests both how susceptible your device is to interference from other electronic devices, power surges on the main's electricity supply, and so on, and how much electromagnetic interference your product itself emits.
- Electromagnetic interference is the "electrical noise" generated by the changing electrical currents in circuitry.

- When generated intentionally, it can be very useful: radio and television broadcasts use the phenomenon to transmit a signal across great distances, as do mobile phone networks and any other radio communication systems such as WiFi and ZigBee.
- The problem arises when a circuit emits a sufficiently strong signal *unintentionally* which disrupts the desired radio frequencies.
- In addition to the test report, you need to gather together PCB layouts, assembly certificates, the certificates for any precertified modules that you have used, and datasheets for critical components.
- This information is all held in a safe place by the manufacturer (that is, you) in case the authorities need to inspect it.
- The location of the technical file is mentioned on the declaration of conformity, which is where you publicly declare to which directives in the regulations your device conforms.
- In Europe, you must also register for the Waste Electrical and Electronic Equipment Directive (WEEE Directive).
- It doesn’t cover any of the technical aspects of products but is aimed instead at reducing the amount of electronic waste that goes to landfill.
- Each country in the EU has set up a scheme for producers and retailers of electronic and electrical products to encourage more recycling of said items and to contribute towards the cost of doing so.

Q10) Which things affect the COST of PCB?

- You have many things to consider as you move to higher volume manufacturing.
- Unfortunately, lots of them involve sizeable up-front costs.
- In fact, the further you get into the process, the less you will need your hardcore coding or critical design skills, and the more time you’ll spend balancing cash flow and fund-raising.
- If you don’t have much experience with managing delivery of inter-related tasks and deadlines, it is worth seeking out a trusted advisor or partner to help.
- *Someone* on the team needs to keep an eye on how things are progressing; which things must be done before other tasks can proceed; and, peering further into the project timeline, spot (and deal with) problems before they become a roadblock for the rest of the team.
- Many of the online PCB services include a quoting tool, so even before the design is finished, you can get a feel for the likely price.
- You should be able to make a reasonable guess on the size of PCB you need and the number of layers it’s likely to require; those are the main factors that the quoting tools use to work out the cost, plus extras if you want something out of the ordinary such as a different coloured solder mask.
- For assembling the PCBs, you should budget something around €150–€800 for setup costs—getting the solder paste stencil made, programming the pick-and-place machine, and so on—and then between 3 and 14 pence for placing each component.
- This doesn’t include the cost of the components themselves.
- Lawrence Archard, a hardware engineer with much more experience of going to manufacturing suggests that after you know the price for a few hundred of a component, each time you increase quantity by a factor of 10, the item cost will drop to three-quarters of the price.
- Pricing the plastics or other physical components depends far too much on the design for us to be able to give meaningful numbers here, but getting a hardened

steel tool made, which will be good for churning out 100,000 parts before it needs replacing, could easily run to €10,000.
- And to get through certification, you are likely to need a similar amount again.
- Simpler devices being certified in fewer territories might get through certification for around €2,000; more complicated designs, particularly those involving uncertified RF modules, could see costs 10 times that amount (€20,000) to get all the certifications in place.
- Naturally, as the project progresses, you will get more accurate quotes (and eventually completely accurate invoices…) as you talk to suppliers and get ready to place orders with them.
- This will let you update your BOM and cost spreadsheets and have the new information ripple through your plans.

Q11) How to SCALE UP SOFTWARE? Also explain various factors that require polish.
- Software is intangible and malleable.
- There are no parts to order, no Bill of Materials to pay for.
- The bits of information that make up the programs which run in the device or on the Internet are invisible.
- The software you wrote during project development and what ends up in production will be indistinguishable to the naked eye.
- Software has to be polished before it can be exposed to the real world.

- **The various factors that require this polish: correctness, maintainability, security, performance, and community.**

- **DEPLOYMENT**
- Copying software from a development machine to where it will be run from in production is typically known as *deployment*, or sometimes, by analogy to physical products, *shipping*.
- In the case of the firmware running on the microcontroller, this will (hopefully) be done once only, during the manufacture of the device.
- For a simple device like the Arduino which usually only runs a single program, the process will be identical to that in development.
- For a device like a BeagleBone or Raspberry Pi which runs an entire operating system, you will want to "package" the various program code, libraries, and other files you have used in a way that you can quickly and reliably make a new build—that is, install all of it onto a new board with a single command.
- The software for an online service will tend to run in a single location.
- However, it will generally be visible to the whole of the Internet, which means that it will potentially be subject to greater load and to a greater range of accidental or malicious inputs that might cause it to stop working.
- It is necessary, to update the online software components more regularly than those on the device.
- Having a good deployment process will allow you to do this smoothly and safely.
- Ideally, with one trigger (such as running a script, or pushing code to a release branch in your code repository), a series of actions will take place which update the software on the server.

- **CORRECTNESS AND MAINTAINABILITY**
- So, you've sold a thousand Internet-connected coffee machines. Congratulations!

- Now it’s time to cross your fingers and hope that you don’t get a thousand complaints that it doesn’t actually work. Perhaps it makes cappuccinos when the customer asked it for a latte.
- Maybe it tweets “Coffee’s on!” when it isn’t, or vice versa.
- Clearly, as a publicly available product, your software has to do what you claimed it would, and do it efficiently and safely.
- Testing your code before it is deployed is an important step in helping to avoid such a situation, and your chosen language and development framework will have standard and well-understood testing environments to help ease your task.
- The embedded code in the device, however, is particularly important to test, as that is the hardest to update once the product has been sent out to the users.
- It may be possible, given that the devices will be connected to the Internet anyway, for the code to be updated *over-the-air.*
- If your device can update its own code, the channels for delivering and authenticating any updates must be rock solid, lest they be compromised and, as mobile technologist Brian Proffitt warns, “The Internet of Things might try to kill you”.

- **SECURITY**
- These concerns lead us back to the important issue of security.
- Following are some of the more important guidelines:
- 1.Make sure that your servers are kept up-to-date with the latest security patches, are hardened with the appropriate firewalls, and detect and mitigate against password hacking attempts and rootkit attacks.
- 2. User passwords should never be stored in plain text.
- If your database were ever compromised, an attacker could easily log in as any user.
- 3. Never simply trust user input.
- Check that anything that is entered into a web application fits the type of data you expect, and refuse or clean anything which doesn’t. Although you may think input from your connected devices would be okay (because you wrote the code), it is possible that it has been compromised or an attacker may be “spoofing” it.
- In particular, be wary of passing user input to your database without checking it (otherwise, you risk an SQL injection attack were it to include SQL commands)
- 4. Be aware of cross-site request forgery (CSRF) attacks from other malicious or compromised websites.
- For example, if one of your users browses a bad site which uses JavaScript to open http://some.example.com/heating?switch=off on your site, and the user is already logged in, he may come home to a cold house.

- **PERFORMANCE**
- The first thing many people think of when considering scaling up software is whether it will be fast enough and handle a large number of users.
- If your web service is running on a modern framework, it should be easy to scale up by deploying the code onto a more powerful machine, or by running multiple servers, with a front-end server or proxy managing the load.
- You should concentrate on running your web application with appropriate standard infrastructure which is good enough for most problems.
- If you’re lucky enough to have so much traffic that you need to scale, you should always optimise using this algorithm:
    - Identify that you actually have a problem.
    - Measure and profile the tasks which are slow and identify the problem.

    - Fix that problem. The web community does a good job in sharing best practice in how to address such problems, so you often find that others have trodden the path before and documented their tried-and-tested solutions.

- **USER COMMUNITY**
- Whether you are launching a mass-market commercial product or an open-source community effort, your project will be successful only if people actually use it.
- At a minimum, you need a support email or a bug-reporting tool (ideally a public one, so that users can easily find related issues), but you will probably have a presence on various social media forums (Twitter, Facebook, and the like).
- As well as responding to queries, however, you should think about the user experience before launching: does your website have documentation, introductory videos, and tutorials?
- Blogging regularly about product development and related topics helps build a readership of users, both current and potential.
- Finally, some form of forum, mailing list, or chat room lets users support each other, which reduces your workload, but more importantly helps to build up a community and expertise around your product.