AS Systems & Control

Manufacture Coursework

Musical Cycle Training Aid

Jacob Burge — 9030

TWGSB — 61985

Situation

Training for cycling using rollers or a turbo trainer can be very boring as it involves being stationary in a single place, often for over an hour. To make the time more bearable many people listen to music. Training sessions whilst stationary are often cadence (how fast the pedals spin) based. However what happens if a song with a very slow beat is playing whilst you are trying to ride with a high cadence? I aim to make a device capable of measuring the riders cadence and selecting a song with a suitable beat to match.

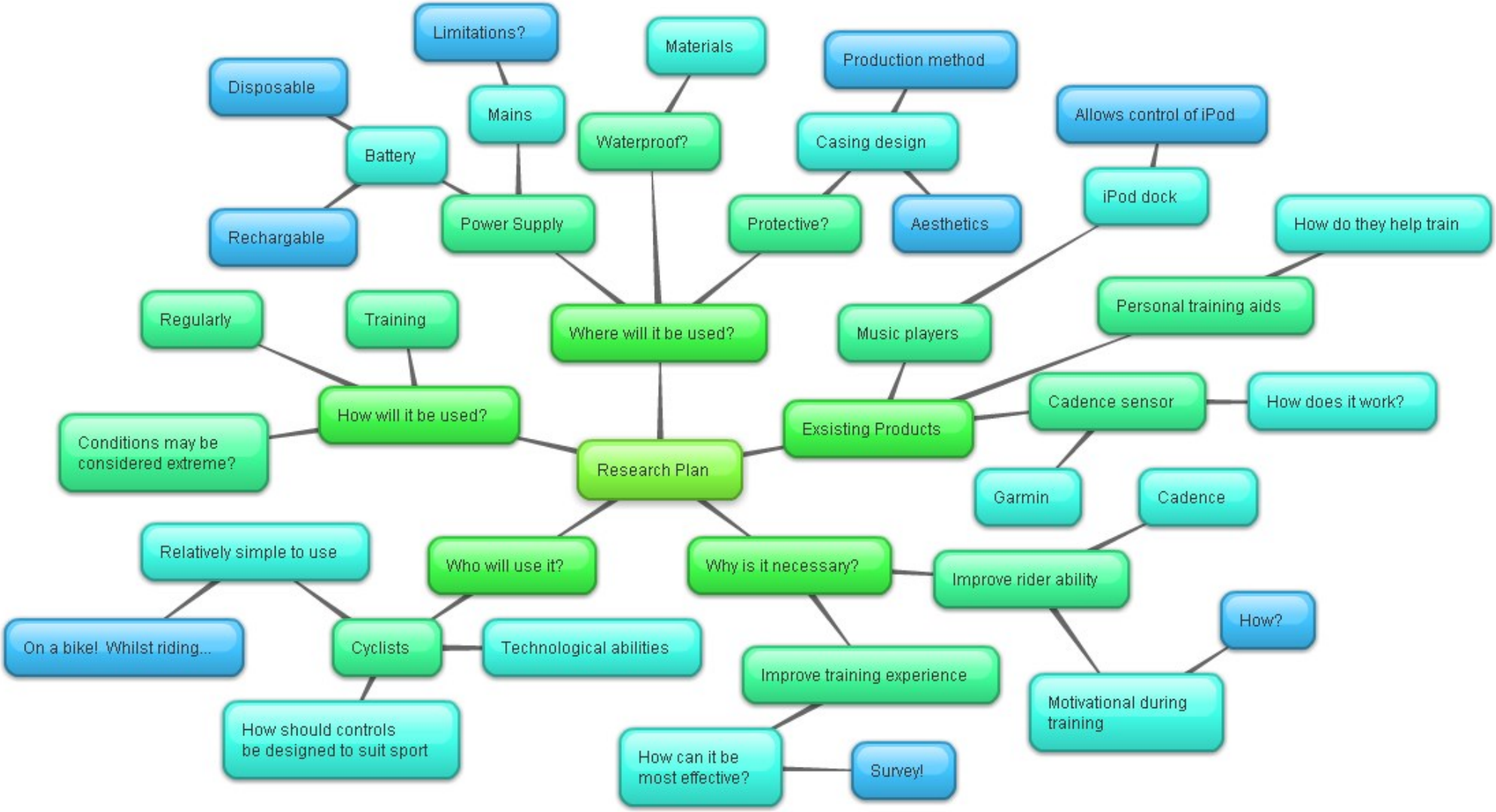
http://collinscycleshop.files.wordpress.com/2008/12/rollers.jpg

Design Brief

The product will be capable of monitoring rider's cadence and playing a song with a relevant beat with the intention of motivating the rider whilst training. The user will also be able to pause the music and manually skip forwards and backwards between songs. This will be done using mainly pre designed PCBs as this is a manufacture focused project.

Because the device will only be used during training, it is not critical that the device is as light or compact as possible. Also it will be designed to stay dry as people generally train indoors if the weather is poor.

Research Plan



User Profile

Based on the needs of my client, I have put together a user profile to help me design a suitable product for the end user:

Purpose	To reduce boredom whilst training, make it easier to reach the target cadence as well as being motivational and fun.
Where does user train	If it is dry then outside, but when raining in the garage.
How long is an average session	Roughly 1 hour. Assuming 3 minute average song length that's 20 songs.
Aesthetics — how important is it to user	Not critical as training takes place on your own, but poor aesthetics may damage the motivational effect of the product.
Weight — how important is it to user	Not important at all, because the training is static it would have little effect. Also it's only training!
Size — how important is it to user	Quite important so that the product doesn't get in the way.
Potential realistic price range	£20—£30
Do users already own an iPod	Yes, or have access to a family member's.
Do you users use headphones to listen to music	Sometimes headphones but sometimes out loud.
How technologically able are users likely to be	Moderately, can't do complicated computer things but can manage most generic software applications.
Does user measure cadence, why?	Very important in training because the faster you can spin your legs, the lower gear you can be in to achieve the fastest speed possible making endurance riding easier.
Would the user feel confident in ordering songs	Not very, some software that could do it for me or at least help me would be useful.
Other training aids used by user	Garmin cycle computer which monitors speed, cadence and heart rate. It is either mounted on the stem or out front of the handlebars.
Would user like manual control also	Yes, manual controls would be a nice feature, however the buttons would need to be easier to press whilst riding and sweat proof!
Power source preference	Ideally a rechargeable power source should be used, Id prefer not to have to remove the batteries to charge.

Benefits of Good Cadence

There are three main factors that the rider has control overt that will influence there speed:

- The effort they are putting in — this can be measured using there heart rate
- The gear they are in — a gear ratio of 1:2 will give more speed but less torques were as a gear ratio such as 3:1 will give much more torque but less speed
- The speed at which they spin the cranks (**Cadence**) -

By far the most important factor to how fast you go is how fast you pedal; the faster you pedal, the faster you can go. But there are more benefits to being able to pedal fast than just speed: Firstly, if a rider can pedal faster than everyone they are riding with, they can use a higher gear ratio e.g. 1:2 instead of 1:4. By doing this there legs are having to exert less force to maintain the same speed; this should have a beneficial effect on the riders endurance.

By spinning there legs faster, riders naturally increase the blood flow around there body whilst maintaining the same power output. This supplies the muscles with energy and oxygen at a faster rate allowing the muscles to continue working efficiently.

So Cadence can have a big impact on a riders speed and endurance which is why many riders base there training sessions on cadence readings. Many training programs have set cadence zones which must be achieved constantly for a certain time. An example of a 60 minute training routine is below:

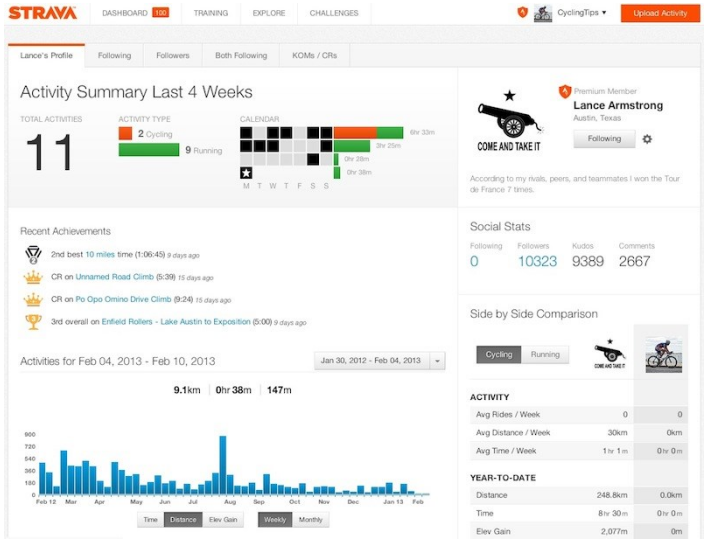
Source: http://www.bikeradar.com/gear/article/technique-cadence-matters-16394/

Warm up	**Drill set**	**Main Set**	**Cool Down**
10 minutes of easy spinning	Single leg intervals — 3 x 5 minutes • Minute 1 — Right leg only • Minute 2 — Left leg only • Minute 3 — Both legs highest cadence possible • Minute 4-5 — Recovery	Maintain 17mph for 2 x 20 minutes + 5 minutes recovery • Minutes 0-2 — 75 RPM • Minutes 3-4 — 80 RPM • Minutes 5-6 — 85 RPM • Minutes 7-8— 90 RPM • Minutes 9-10— 95 RPM	10 minutes easy spinning

http://triathlon.competitor.com/2014/07/training/two-cycling-workouts-to-improve-your-cadence_36744/2

The purpose of my product is to aid the training routine by making the time more enjoyable and providing riders with motivational support.

Training Aid Research



http://cyclingtips.com.au

http://www.ironman.com/

http://www.greatelectronicdeals.com/

Strava

Strava allows users to upload GPS files of rides they have done allowing them to track progress through a series of graphs and graphical representations of the data. It also adds an element of competition by comparing riders time on set routes and rewarding them through awards such as king / queen of the moutains.

Tacx Turbo Trainer System

Makes static training a little more interesting through the use of footage, filmed by team cars, from events such as Le Tour de France and Giro d'Italia. By playing this through a telly in front of the rider training is made more interesting and a feeling of actually moving rather than being stationery is achieved.

Arm strap for iPhone

A very simple product used a lot by runners and cyclists for listening to music whilst training. By strapping it to your upper arm cables have less far to go, so get less tangled and apps can be used to track speed and distance through the phones GPS.

Conclusion… There are already many training aids designed to make training more interesting. This is achieved through competition or entertainment using multimedia. My product will be orientated more around entertainment than competition although one thing all of these products have in common is being motivational. All of the products also look aesthetically pleasing which I think is another important feature of a training aid. Therefor I need to design my product to functional and aesthetically pleasing; combines these two key features should also make the product motivational.

Making Music

https://www.sparkfun.com/

One of the main features of this project is the ability to play music. Mores specifically the final product must be able to select a specific track based on the users cadence. Therefor the product will need some way of controlling the playback of music. This could be done in several ways: One way might be to use a chip such as the VS1053B (pictured left). This chip is capable of decoding and playing mp3 files from an SD card—perfect! However there are a few problems; firstly the chip is not cheap, this would push the price of the product up unnecessarily. Another problem is the fine pin pitch of the chip which would make manufacture very difficult. And finally because this project is focused on mainly manufacture, we are provided with a modular circuit design. Unfortunately there isn't a module featuring this particular chip, so to allow the project to play the users music, I will have to use an alternative solution.

Another option would be to simply connect a loudspeaker to a digital pin of the Picaxe chip on the main circuit board. However this would limit the user to playing RTTL ring tones which may not have the desired effect of motivating them during training! The problem with the ring tones is they are very simple and can be quite annoying.

The final option is to use an iPod to play the music. I think this is a good option because in this day and age almost everybody has an iPod or iPhone due to the recent advances in technology; this means that the overall product can be made cheaper by utilizing products the customer already owns. This will allow the product to produce excellent quality music, however it ill make the control of the music quite complicated because the product will have to interface with the iPod. This is definitely possible as it is done by many iPod docks already but is something I will have to research thoroughly. The only drawback is it limits the potential customers because another product is required in conjunction with it to work.

http://mos.musicradar.com//images/Product%20News/Tech/Sep09/apple/ipod-nano-family.jpg

However several people I have spoken to have said they would not feel able to sort their music into different speed groups. There is a simple solution to this though, DJ software (such as serato) will save the BPM of a song to its ID3 tag allowing the BPM to be viewed in the properties of an MP3 file. This will allow the user to easily sort their music so it can be played based on there cadence. If the option of a more powerfull microcontroller was available, beat detection algorithms could be built into the final product, allowing for automatic sorting of playlists.

serato DJ

www.thebeatcorner.com

Existing Product Research

https://www.thebcgpsstore.ca/Garmin-fitness-speed-cadence-sensor-for-Edge-GPS-systems-theBCGPSstore.ca/

Garmin Edge wireless cadence sensor

This is used in conjunction with a Garmin edge GPS. It sends cadence data to the main GPS wirelessly allowing the rider to monitor there cadence and analyse it after there ride or training session. It works by detecting the magnetic field from a magnet attached to one of the cranks on the riders bike; this is probably done using a reed switch or hall effect sensor. This is probably the best way to measure the cadence in my product, however I would like it to be able to select appropriate music based on the riders cadence, rather than just display the data.

http://www.zanda.com/items/things/9108016oooo068/bose-sounddock-series-ii-iphone-speaker

Bose SoundDock Series II

Although the iPod dock has nothing to do with bikes, it does provide a good way of playing music and controlling the music that is playing. This is done by the iPod being connected via the 30 pin connector giving the dock access to various control protocols. The circuitry will be very similar in my project (without the amplification and speakers) however it will have to be a lot smaller if it to practically fit on a bike. Also the iPod dock itself does not allow for a specific song to be selected—this is something I need to be able to do for my product to work effectively.

http://reviewresources.bugs3.com/cheap-waterproof-bicycle-bike-mount-case-for-apple-iphone-3-4-5-3gs-4g-4s-s-ipod-

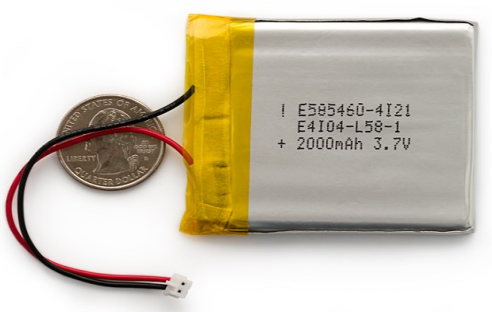
Phone mount for bikes

This design is good as it allows for a phone to be viewed whilst riding but it doesn't clog up the handlebars. Also the pouch underneath the phone can be used for an additional power supply. If my casing design were to be like this I could also mount the circuitry in this position. I think mounting to the top tube is a good option as it does not add extra weight to the handlebars meaning steering is not effected. The case also claims to be waterproof, however for my application this would not be necessary as people normally train under cover.

Power Supply Research

http://media.digikey.com/photos/CUI%20Photos/

http://www.electricswitches.com/images/



https://www.sparkfun.com/tutorials/380

Mains Transformer

Advantages:

- Never runs out of charge
- Would take up little space on the actual bike

Disadvantages:

- Requires a cable to be connected to the bike which could get in the way
- Means a mains outlet must be near by in order to use the product

Disposable Batteries

Advantages:

- Relatively compact so can be mounted on bike
- Requires no cables going to the bike

Disadvantages:

- Runs out of charge after extensive use
- Can't be recharged meaning they must be thrown out—bad for the environment
- User must buy new batteries every time they run out

Lipo Battery

Advantages:

- Relatively compact so can be mounted on the bike
- Can be recharged saving the user money and conserving resources

Disadvantages:

- Requires a separate charger
- Output is only 3.7V so some kind of step up is required to power a 5V microcontroller

Production Methods and Materials

http://cdn2.digitalartsonline.co.uk/cmsdata/features/3469825/ultimaker-2.jpg

3D Printer

3D printing in the form of Additive Layer Manufacturing (ALM) is very useful in rapid prototyping as it allows for complex models to be produced from a CAD file. This technique could be used for making parts for my product which require high tolerances or complex shapes. However there are a few issues, for example warping can occur as the part cools, effecting the final product. Also any overhangs must be supported by support material which leaves rough sides on the finished item which is not very aesthetically pleasing. 3D printers commonly use ABS or PLA plastic.

http://www.apa21.org/wp-content/uploads/2013/04/hot-font-b-sale-b-font-wood-engraving-font-b-machine-b-font-.jpg

CNC Router

Using a CNC router is also highly accurate however it only really operates in 2D meaning the design must be constructed after cutting. Materials such as acrylic and ply wood can be cut using a CNC router. The designing is done using “D design software and from this a tool path is generated and then executed by the machine. One limitation is the size of the milling bit, this limits the size of holes that can be made and only reflex angles can be cut to full effect.

http://www.crclarke.co.uk/products/img/8.jpg

Vacuum Forming

Vacuum forming does not require and CAD modelling, instead a mould is produced, using either styrene or MDF usually. This method of moulding uses Rigid Foam PVC and produces a hard wearing case that looks aesthetically pleasing. Another method using High Impact Polystyrene (HIPS) can also be used called plug and yolk moulding. This produces a much stronger casing, however it is much thicker and therefor less complex shape can be moulded. There is a limit to what can be done using a vacuum former, because if the shape is to complex it is likely that it will not be possible to remove the mould.

Mounting Options

http://blog.artscyclery.com

http://www.dcrainmaker.com/images/

http://reviewresources.bugs3.com/

Out Front Mount

Advantages:

- Easy to see whilst looking forwards
- Doesn't get in the way of the stem whilst turning

Disadvantages:

- Adds weight to the handlebars which may effect turning
- Adds to handlebar congestion!

Stem Mount

Advantages:

- Doesn't get in the way of the stem whilst turning
- Is out of the way of the handlebars — very few things are ever attached to the stem

Disadvantages:

- The rider has to look down to view the device
- May not be compatible with short stems

Top Tube Mount

Advantages:

- Doesn't effect the steering in any way
- More storage space for batteries

Disadvantages:

- Rider has to look down a long way to see device
- Depending on style may interfere with stem as the rider moves the bars

Specification

Purpose

- *To reduce boredom whilst training, make it easier to reach the target cadence as well as being motivational and fun.*

Function

- *Able to sense a riders cadence*
- *Make informed song choices based on the cadence information*
- *Select a specific song from a playlist on an Apple device*
- *Provide an audio out which can either be played through headphone, or plugged into an amp and speakers*
- *Give the user the option of manually controlling the music*
- *Provide a utility for helping the user to sort their music by speed*
- *An LED will display rough position in a cadence 'zone' by changing in brightness*

Aesthetics

- *The manufactured product must have as higher quality finish as is possible with the equipment available at school.*
- *The aesthetics must not damage the motivational effect of the product.*

Performance requirements

- *The manufactured product must provide the required functions and all should work consistently over a prolonged testing period.*
- *The manufactured product should be easy to use with all inputs clearly labelled.*
- *The product must aid training not hinder it by getting in the way*

Materials

- *The manufactured product should use casing materials appropriate to its purpose that can be processed using the tools at school.*

Components

- *The manufactured product should use the relevant electronic components as detailed in the accompanying parts list.*
- *The manufactured product should make use of any relevant casing assembly components in stock at school so as to ensure the highest quality outcome (e.g. PCB mounts)*

Specification

Market and user requirements

- *The product must be in the price range of £20—£30*
- *The product must fit as many of the points identified in the user profile as possible*

Manufacturing processes

- *The manufactured product should be manufactured using only the available tools and processes at school.*

Technology

- *The product must use the PICAXE system of Peripheral Interface Controller for its main processing needs.*
- *The product must be able to interface with the iPod Serial protocol*
- *The product will be powered from a lithium polymer battery, rechargeable via USB*

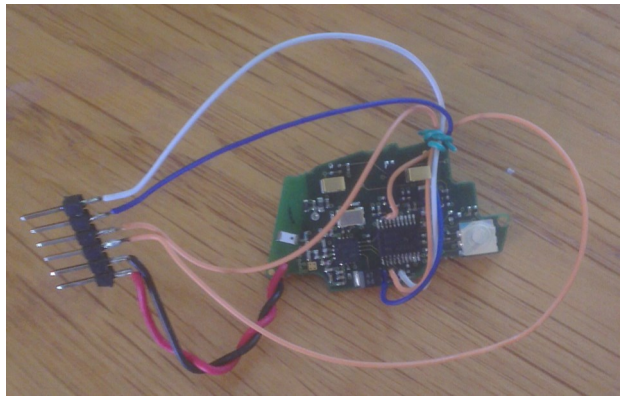
Scale of production

- *The product must be completed as a prototype with a view to the product being mass/batch produced*

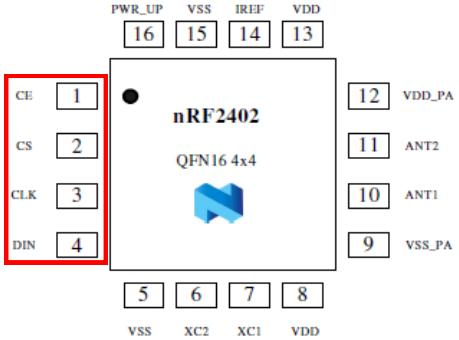
Quality and safety issues

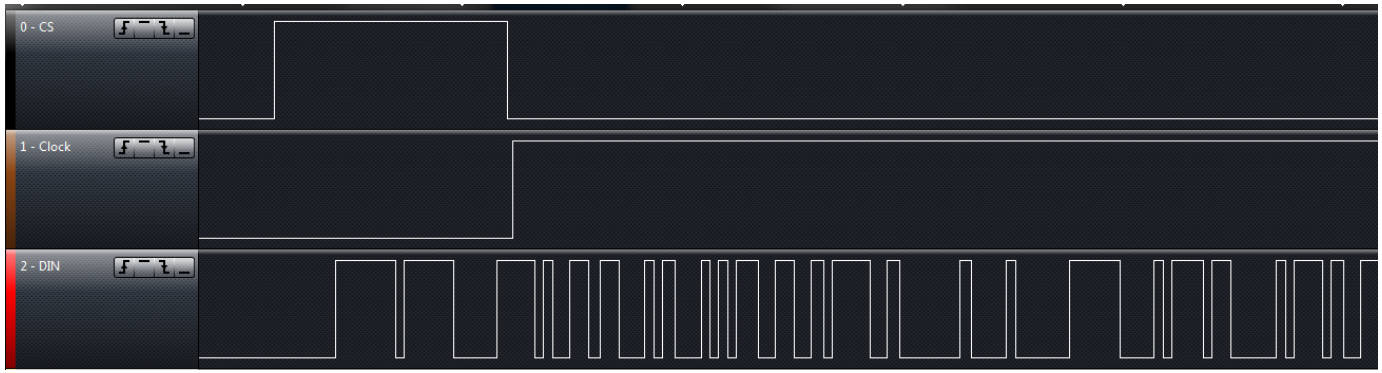
- *The product must be safe to use by an adult.*

Cadence Sensor Hack

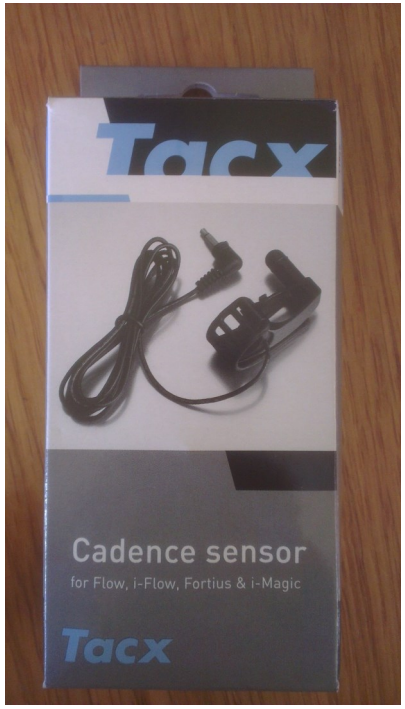


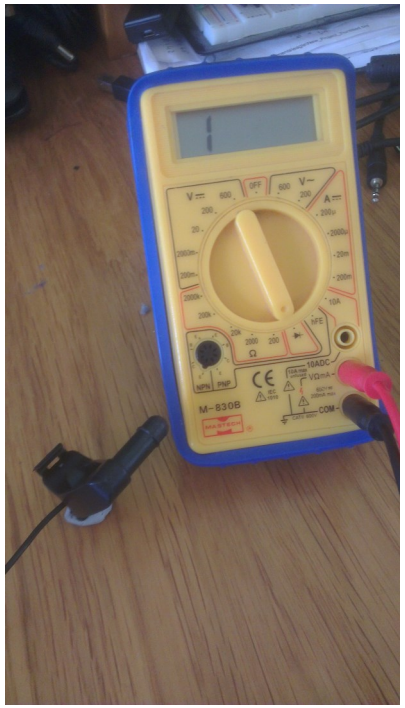
One of the significant features of my project is its ability to sense a riders cadence. There are already many products on the market for this so I decided to try and hack one instead of build my own. This is the Garmin Edge 800 cadence sensor which is wireless — this made it very challenging to hack but also this is a working unit which my mum had spare so I couldn't just chuck out the PCB and use the sensors. I intended to read the data going to the RF chip (nRF2402) and then if I was successful purchase a similar chip to simply read the wireless signals. I opened the unit vey carefully, it was only glued shut so a scalpel around the seam allowed me to split the shell open. On the left you can see the PCB which features an MCU and separate RF chip as well as a number of passive components. I looked up datasheets for all the chips and was able to establish the pins used to send data to the RF chip. I soldered some very fine wires to each of these pins and then two further wires for +3v and ground to the battery terminals. Finally I hooked the setup to my logic analyser and sampled the pins. I established that data was transmitted 4 times a second and each time the RF chip would be powered up, configured, sent data to transmit and then shut down. I was able to read the configuration data as a 14 bit data packet however the rest of the data was of varying packet size. This caused me problems because the software I use can only analyse serial which has a constant serial packet size. Consequently I was unable to make any more progress and hence I have had to reconsider how I might sense cadence.





Cadence Sensor



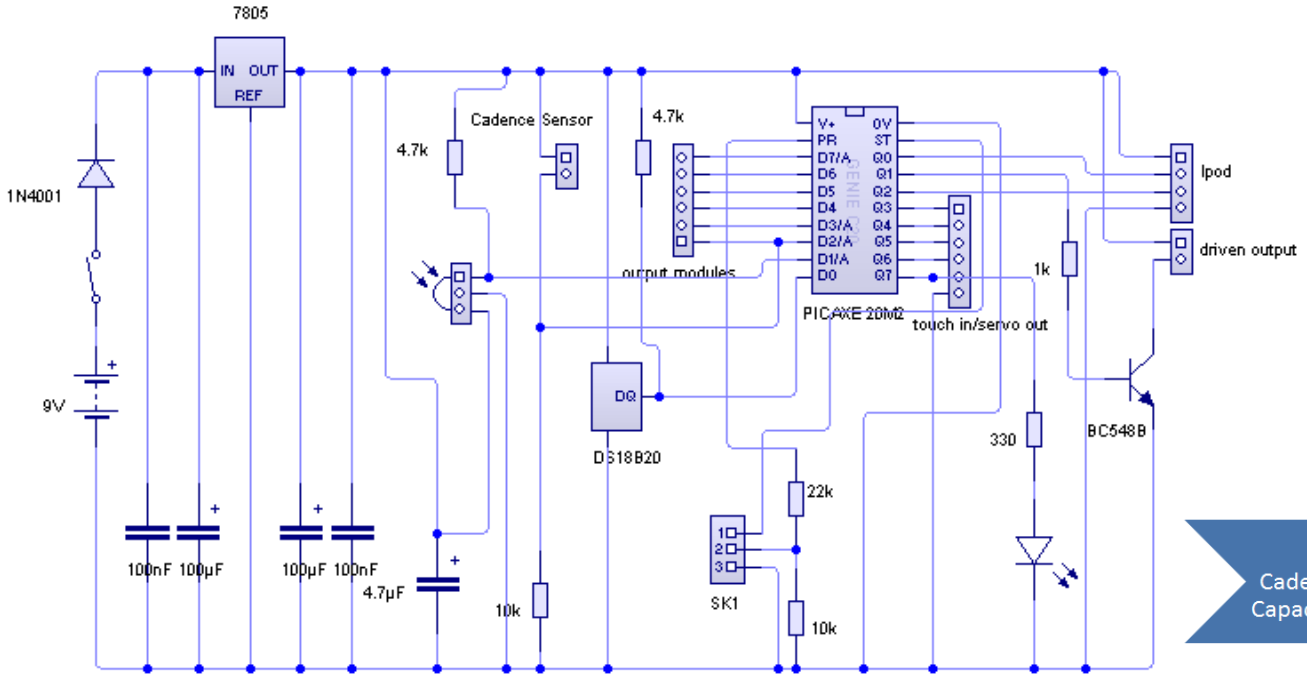


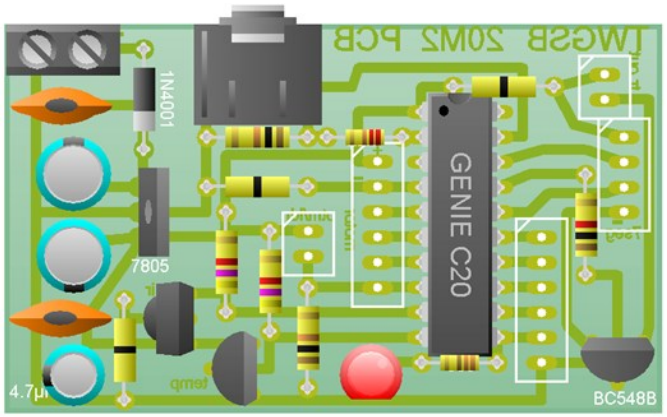
Because this project is purely focusing on manufacture and not design, I can use an 'off the shelf' cadence sensor instead of designing my own. I found this Tacx cadence sensor on eBay and decided to use it in my project as it has a simple wired interface and a 3.5mm audio jack which will make connecting it to my project very easy.

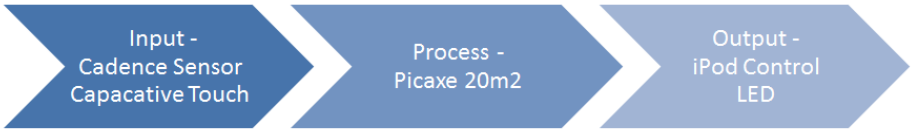
This is the sensor and connector. The sensor requires a magnet to be attached to the inside of the clean side crank. Every time the crank with the magnet passes it the sensor detects it. From this the microcontroller will be able to count how many rotations occur every minute and determine the riders cadence in RPM.

In order for the microcontroller to count the crank revolutions it needs to know how the sensor will behave when the magnetic field, produced by the magnet on the crank is present. To determine this I connected a multi-meter to the two connections on the jack and set the multi-meter to measure resistance. When the magnet wasn't present the sensor had no continuity; however when a magnet was placed near the sensor the resistance began to drop. This means that the sensor can be used in a similar way to how an LDR might be used—in a potential divider. This is good as it means no modifications will be required to get the sensor working with the pre-designed motherboard.

Circuit Modules—Motherboard







Here is the motherboard PCB design we are provided with, fortunately there is no modification which must be done for my project to work:

1. The cadence sensor acts just like an LDR and the motherboard module had a potential divider feeding into an analogue read pin for this purpose. Therefor I will just be able to substitute the LDR for the cadence sensor and perhaps change the value of the pull down resistor
2. There is already a header designed for capacitive touch which is connected to 4 Touch pins on the 20M2
3. The 20M2 is capable of counting the revolutions of the cranks using the “count” command and it can also output serial data on any pin using the “serout” command in order to send commands to the iPod
4. There is an LED built onto the board and if I need to I can extend this on jumper wires so it fits the casing
5. Although the board can’t interface with the iPod directly, the header originally designed for the seven segment add on board has power, ground and two I/Os which provides all that is needed to. I plan to design a very basic board to connect to this header and then the iPod connector.

Research—iPod Serial Protocol

Field	Size	Value
Header	2	0xFF 0x55
Length (A)	1	Size of Mode + Size of Command + Size of the Parameter
Mode (B)	1	Mode relating to command
Command (C)	2	Two byte Command
Parameter (D)	0...N	Optional Parameter, depending on the command
Checksum	1	0x100 - ((Sum of all data in A, B, C & D) & 0xFF)

0x00 0x35	none	Get number of songs in playlist
0x00 0x36	number(4)	number of songs in playlist
0x00 0x37	number(4)	jump to specified songnumber in playlist

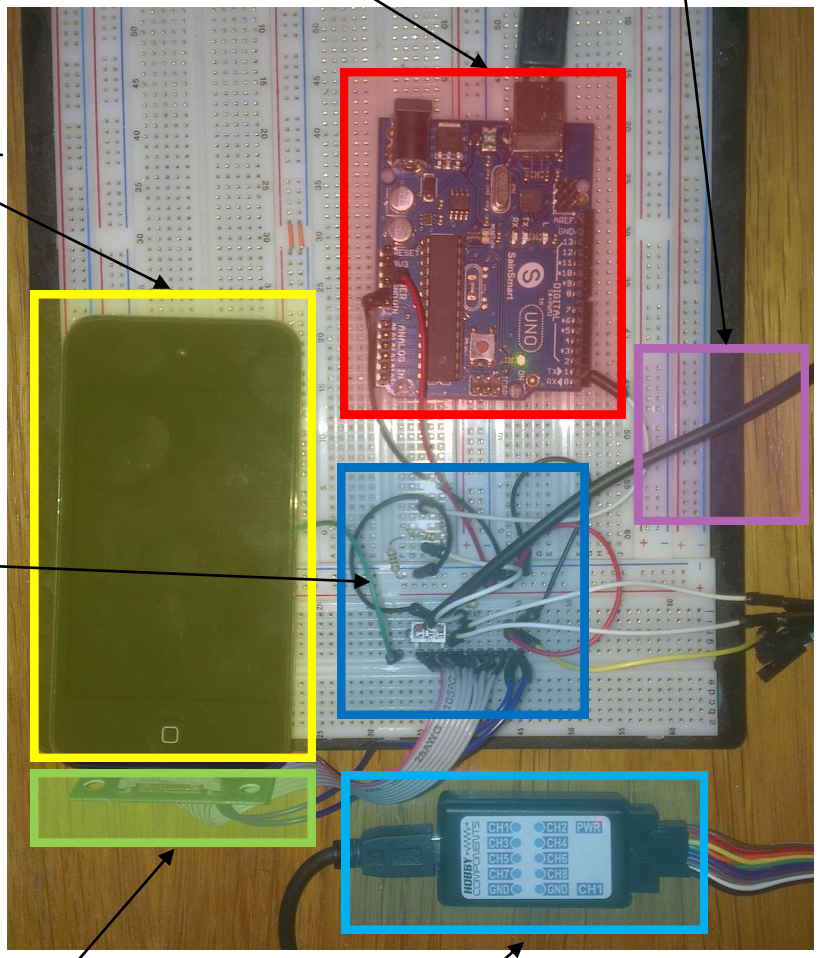
Having decided that my product would be iPod compatible, I set about researching how to communicate with an "I-device". It turns out that it isn't too complicated on devices 4th gen and lower. This is due to the 30 pin connector having serial data lines and there being a special protocol for controlling all aspects of music playback. I initially found this out from an instructables page (http://www.instructables.com/id/Simple-Ipod-Controller/?ALLSTEPS) which was very simple and easy to understand. However for my project I would like to be able to jump to a specific track depending upon the users cadence. This is not something that is covered in the instructables, however there is a link to another useful page which has detailed information on the whole protocol (http://www.adriangame.co.uk/ipod-acc-pro.html). From this I was able to establish that jumping to a specific number song in a play list would be possible, however it would have to be done in a different mode to the one covered in the instructables documentation. Mode 4 is that used by AIR remotes and is capable of much more useful commands, for example retrieving song name or elapsed time compared to the mode 2 commands which were much more simple. On the left at the top is the basics of the iPod serial protocol, and underneath is the commands relating to song number. Because I was unsure of how the song number should be represented in 4 bytes (what "number(4)" represents) I decided to call the command "Get number of songs in playlist" first to see how the iPod would structure its response. I worked out the data that would need to be sent in order to request this information in the following way:

1. **I started off with the header — 0xFF 0x55**
2. **I added the length of the Mode and the Command — 0xFF 0x55 0x03**
3. **I added the mode (AIR mode — 4) — 0xFF 0x55 0x03 0x04**
4. **I added the command to get number of songs in playlist (0x00 0x35) — 0xFF 0x55 0x03 0x04 0x00 0x35**
5. **I calculated the sum of Length, Mode and Command (0x03 + 0x04 + 0x35) = 60**
6. **I Bitwise ANDed 60 with 0xFF (255) -**

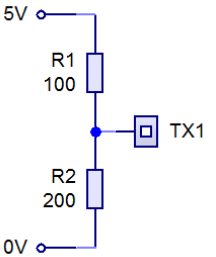
```
1 1 1 1 1 1 1 1  - 255
0 0 1 1 0 1 1 0  - 60
---------------
0 0 1 1 0 1 1 0  - 60
```

7. **I subtracted the result (60) from 0x100 (256) = 196**
8. **I converted 196 to HEX (0xC4) and added it to my data — 0xFF 0x55 0x03 0x04 0x00 0x36 0xC4**
9. **I entered this into my program and ran it...**

Investigation—iPod Serial Protocol



On the left is the set up I used to test my codes. An Arduino uno is used to generate the Serial data. Because the Arduino has a logic level HIGH of 5V, and the iPod is 3.3V a small amount of circuitry is required to shift the logic levels. This is done by having a potential divider on the data line. Other circuitry you can see in the dark blue box is the audio line out connection and a 500K resistor telling the iPod to operate in Serial mode. Then a 30-pin connector is plugged into the iPod providing power, ground, data lines and audio out connections. Because I need to know what data is being sent I have a USB logic analyser attached to the data lines and I can analyse the data being sent on my computer.




```
void setup(){

Serial.begin(19200);//Initialize the serial port to baud rate 19200 (used by ipod)

delay(1000);//Wait to ensure serial is up an running

//Switch to AIR remote mode
Serial.write(0xff);//Header bytes
Serial.write(0x55);
Serial.write(0x03);//Length - 3
Serial.write(0x00);//Mode - Switching
Serial.write(0x01);//Command - Switch to AIR remote
Serial.write(0x04);
Serial.write(0xF8);//Check sum - 0x100 - ((3 + 1 + 4) & 0xFF)
}

void loop(){

//Skip to song 5 in playlist
Serial.write(0xff);//Header bytes
Serial.write(0x55);
Serial.write(0x07);//Length - 7
Serial.write(0x04);//Mode - 4
Serial.write(0x00);//Command - Jump to specified songnumber in playlist
Serial.write(0x37);
Serial.write(0x00);//Parameter - 4 bytes - song number - 5
Serial.write(0x00);
Serial.write(0x00);
Serial.write(0x05);
Serial.write(0xB9);//Check sum - 0x100 - ((7 + 4 + 55 + 5) & 0xFF)

delay(5000);

}
```

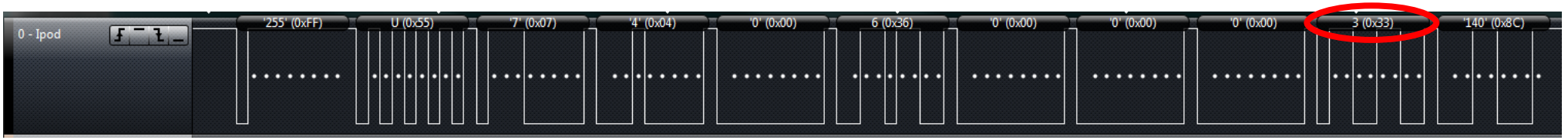
Here is the code which is written in the Arduino IDE and uploaded to the Arduino. It basically switches the iPod into Mode 4 and then sends out the serial that I have programmed it to in individual bytes.

The final product will use a Picaxe chip, but the code will be very similar so it will be easily transferred.

Results—iPod Serial Protocol

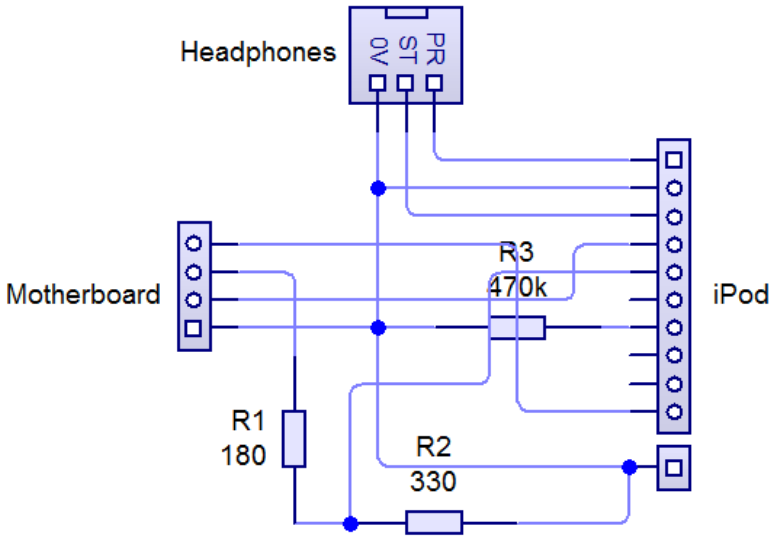
Here is the data from the logic analyser, from the picture below you can see the Arduino (Labelled “Picaxe”) sending a command and the iPod responding saying the command was completed successfully or returning data depending on the command sent.

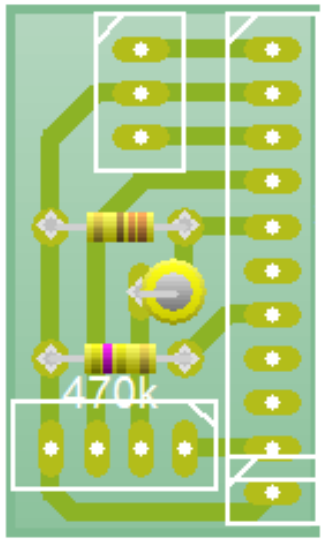
In this example I sent (**0xFF 0x55 0x03 0x04 0x00 0x36 0xC4**) requesting the number of songs in playlist. In this case the result was 0x33 (51). This was useful to me as it demonstrated how a parameter should be sent in 4 bytes.



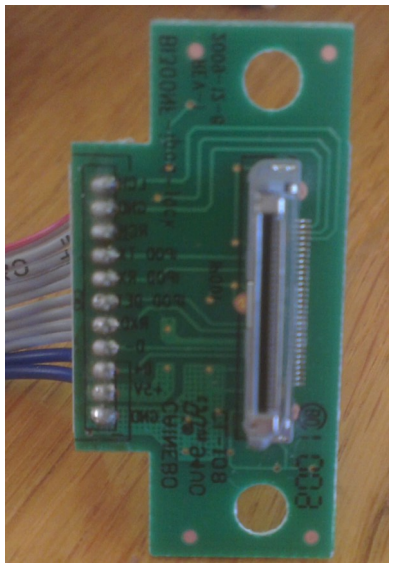
Next I tried requesting the iPod to skip to a certain number track (**0xFF 0x55 0x07 0x04 0x00 0x37 0x00 0x00 0x00 0x05 0xB9**), in this case 5. The iPod changed to track 5 and the iPod responded telling me that that the command was successfully executed (0x00).

Circuit Modules—iPod Connector





Left (Audio)
GND
Right (Audio)
iPod Tx
iPod Rx
iPod detect
Serial enable
D- (USB)
D+ (USB)
+5v
GND



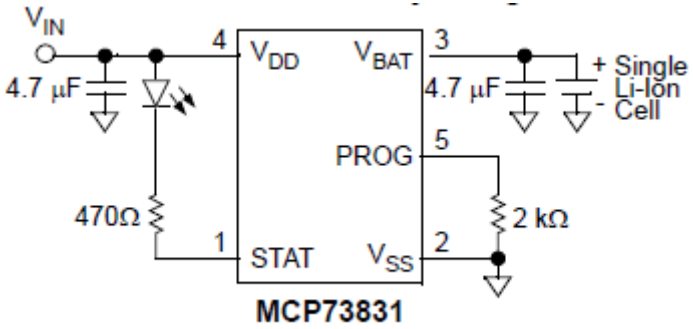
Based on my research into the iPod serial protocol I have decided to use an iPod to allow my music to play music. This is because most people already own an iPod which will bring the total cost of my product down and also provide better sound quality than using RTTL ringtones or a dedicated MP3 decoder IC.

To prevent me having lots of wires soldered to random points on the main motherboard, I have designed this very basic PCB based on my research into the iPod serial protocol. It will plug into the header on the motherboard that is designed for the seven segment display add on. It features an 11 pin header for the iPod connector breakout to connect too (top right). There is a resistor pulling the serial enable pin to ground to enable the serial and also a potential divider to stop the 5v logic high of the Picaxe from damaging the iPod which only has a logic high of 3.3v. There is also a 3 pin header which a 3.5mm headphone jack will be connected too allowing the user to plug there headphones into the device so they can hear the music. This is connected directly to the left and right audio lines as well as ground.

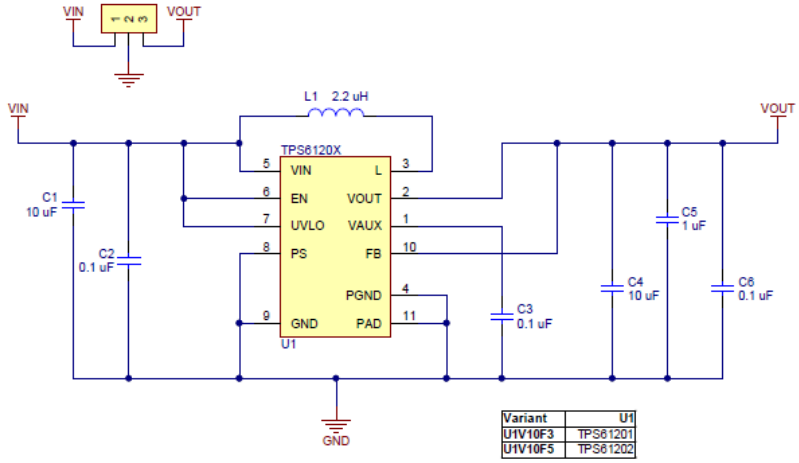
http://beginnerstech.co.uk

One problem with this solution is the updated apple connecter, lightning. Because there are far fewer pins the dedicated iPod serial pins are not broken out. However I think this problem can be solved through the use of the Apple 30 pin to lightning connector. This contains internal circuitry which encodes data from the serial pins of the 30 pin connector and sends them to 5th gen and above apple devices through the lightning connector.

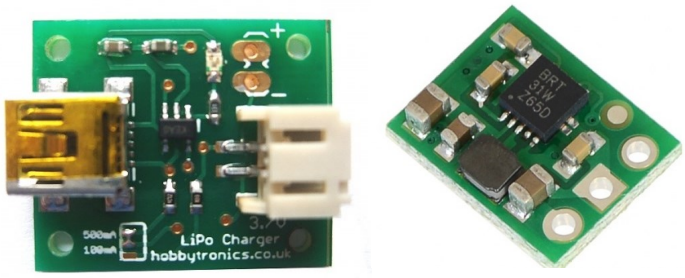
Final Circuit Diagram

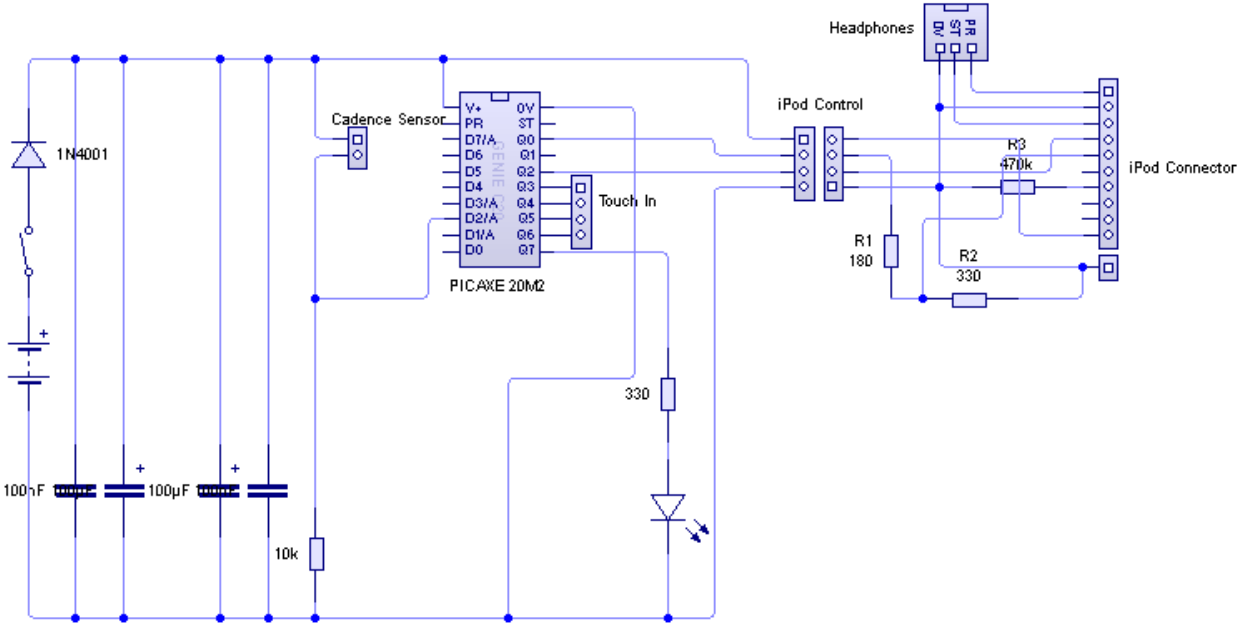


www.hobbytronics.co.uk — LiPo battery charger



www.pololu.com — 5v Step up voltage regulator

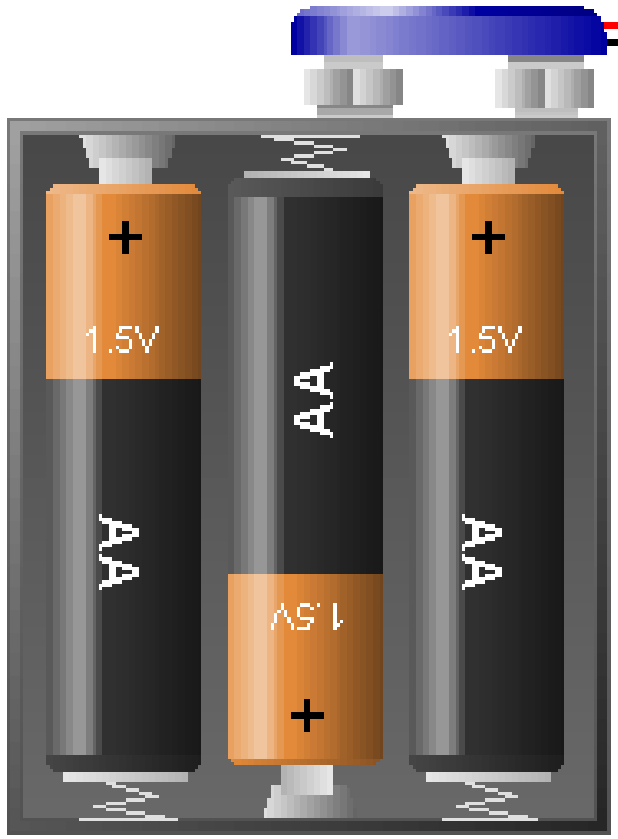




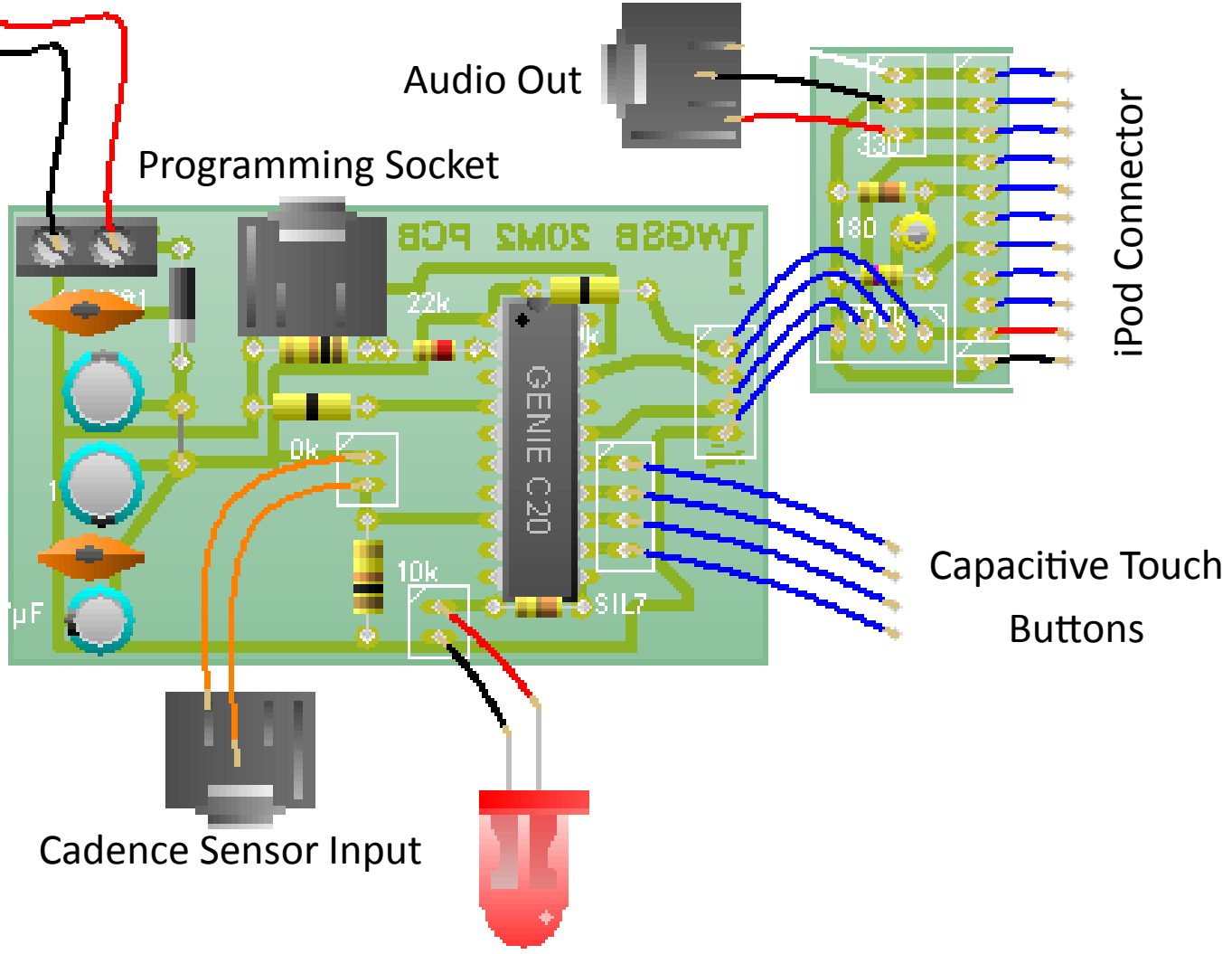
Here is my final circuit diagram, on the right is the main circuit which is a modified version of the supplied motherboard PCB. On the left is two circuit diagrams from the LiPo charger and 5v boost converter that I will be using to power my project. I bought both of these from hobbytronics.com as we do not have the facilities in school to manufacture such items.

The charger circuit uses a dedicated IC to ensure the LiPo battery is not overcharged and that it is charged using the correct current. Because the LiPo's maximum voltage is 3.7v I need to use a boost step up converter to provide the 5v I need to operate my circuit. This also ensures the voltage supply to the circuit remains constant throughout the use of the battery.

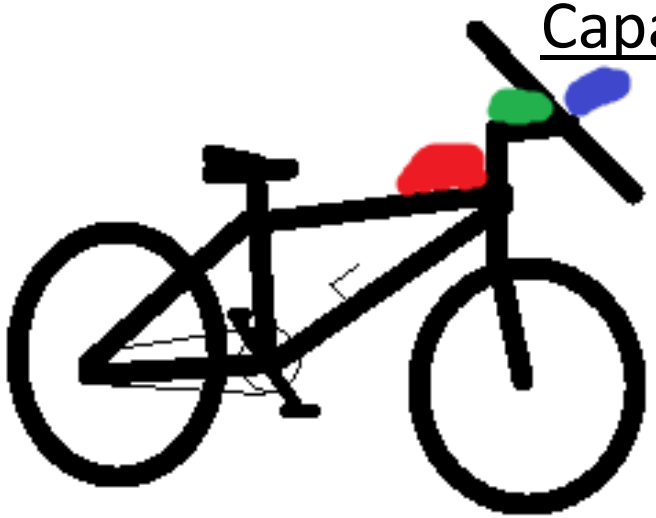
Final Wired Diagram

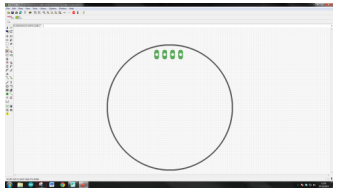


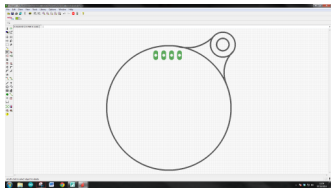
LiPo battery, charger and boost converter

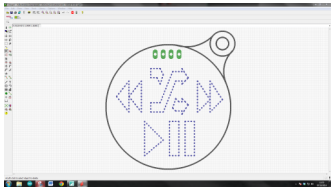


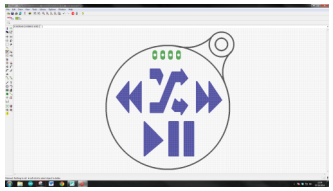
Capacitive Touch Board Development

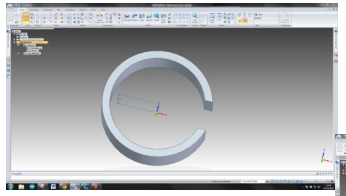


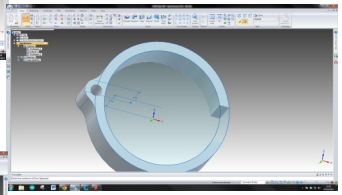


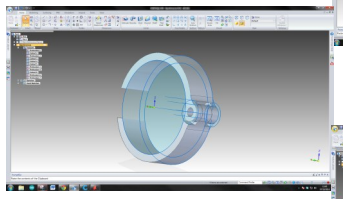


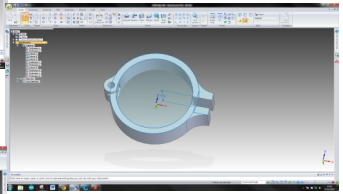


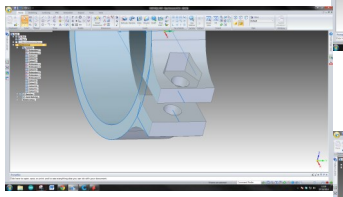


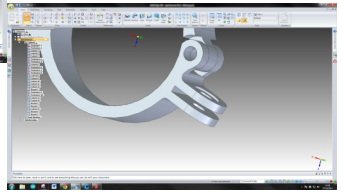






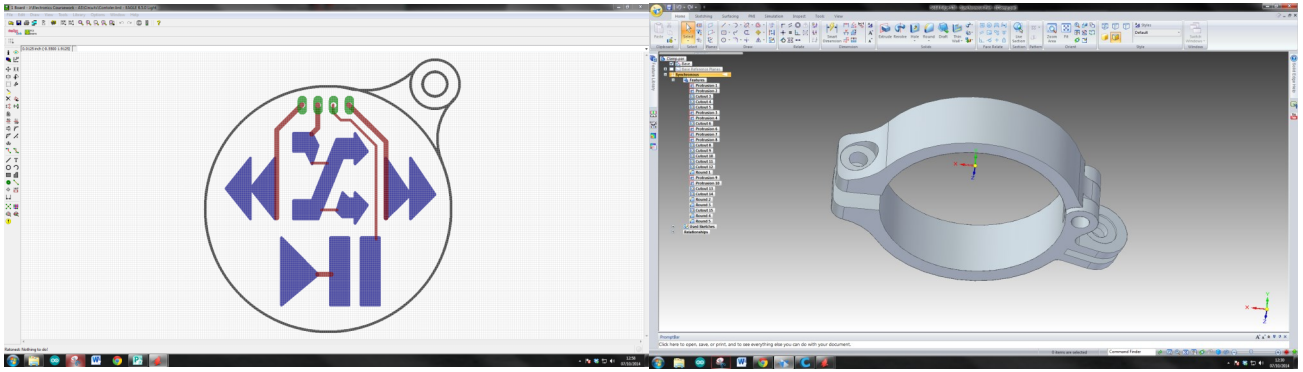




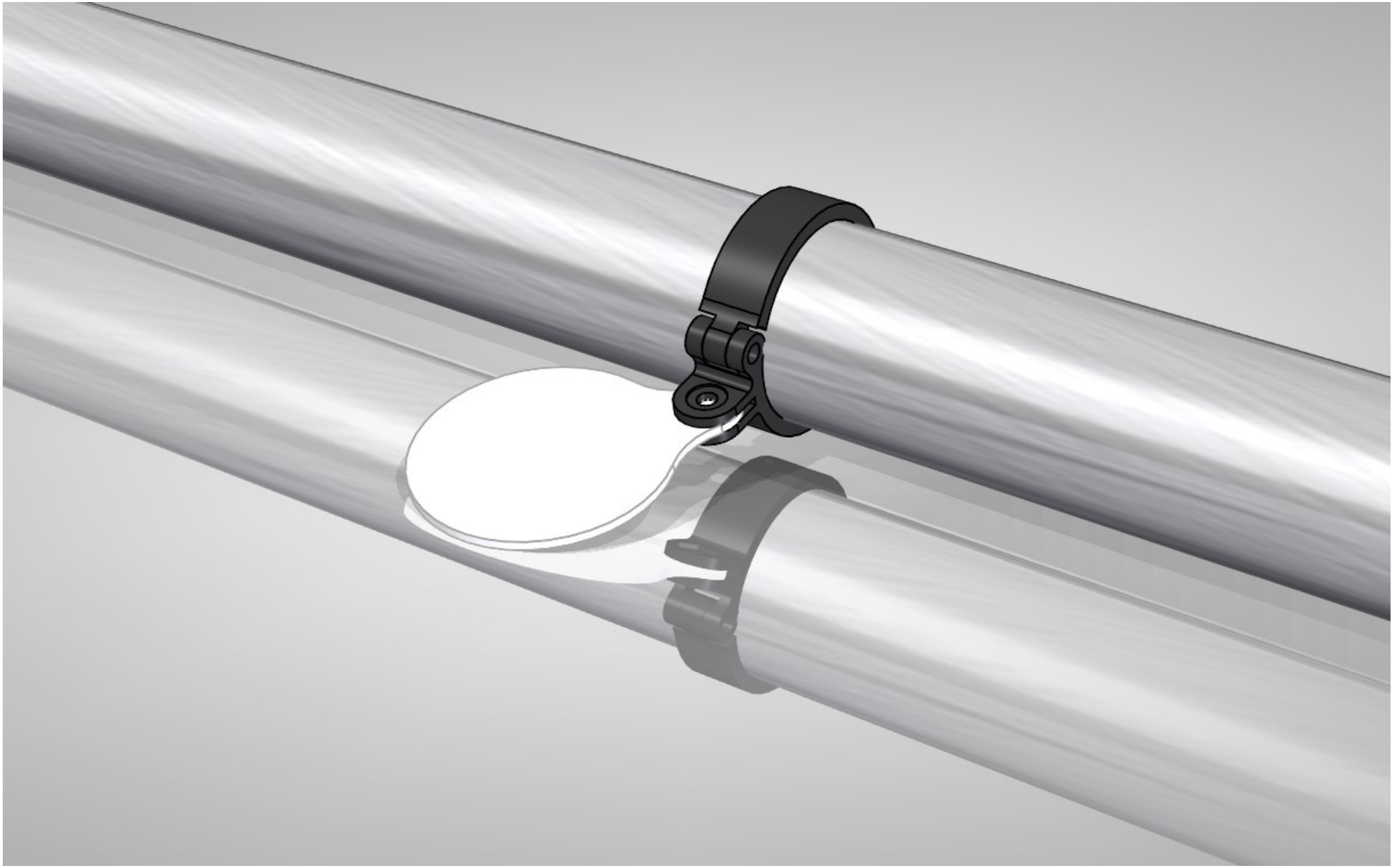


Because the main device is going to be central on the bike, I think it is important the buttons are separate from the main casing. Whilst riding hard, if the rider wants to manually control the device reaching for the centre of the bar could be very difficult. Therefor I plan to attach the controls closer to the edge of the bars, potentially allowing the rider to control the music with no change to their hand position.

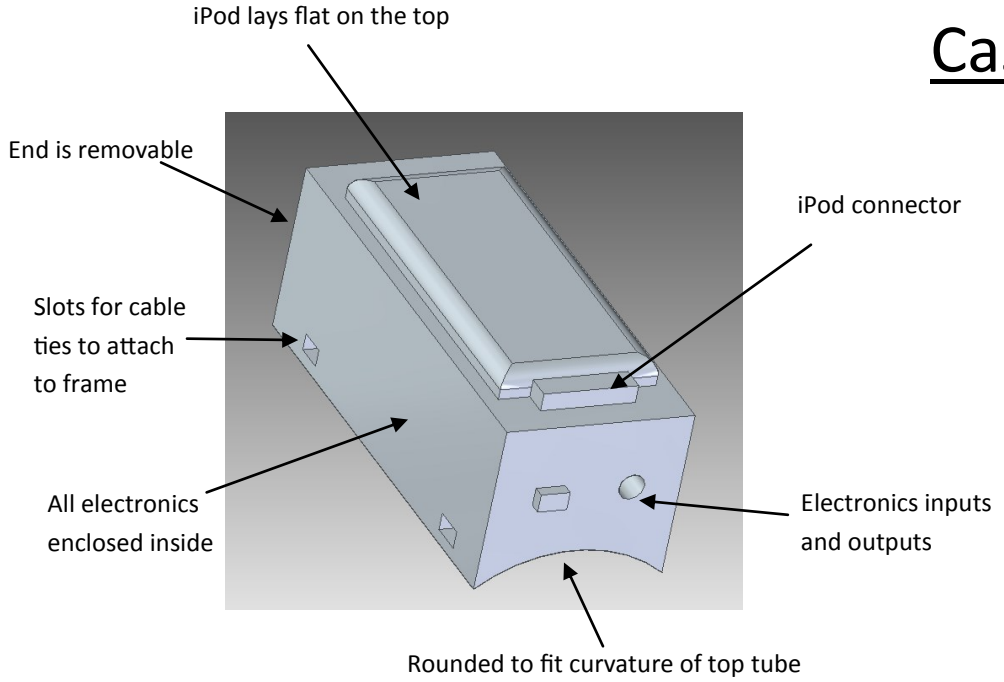
I have designed a clamp which will be 3D printed in PLA and be used to attach my PCB to the handlebars. It has been designed to fit 31.5mm bars — standard on road bikes. It allow the PCB two directions of freedom allowing the rider to adjust where it is on the bars. The PCB has a 4 pin header, one for each button. The buttons consist of forwards, backwards, play/pause and shuffle. Although the buttons are intended to control music, they will also be used for setup.



Capacitive Touch Buttons

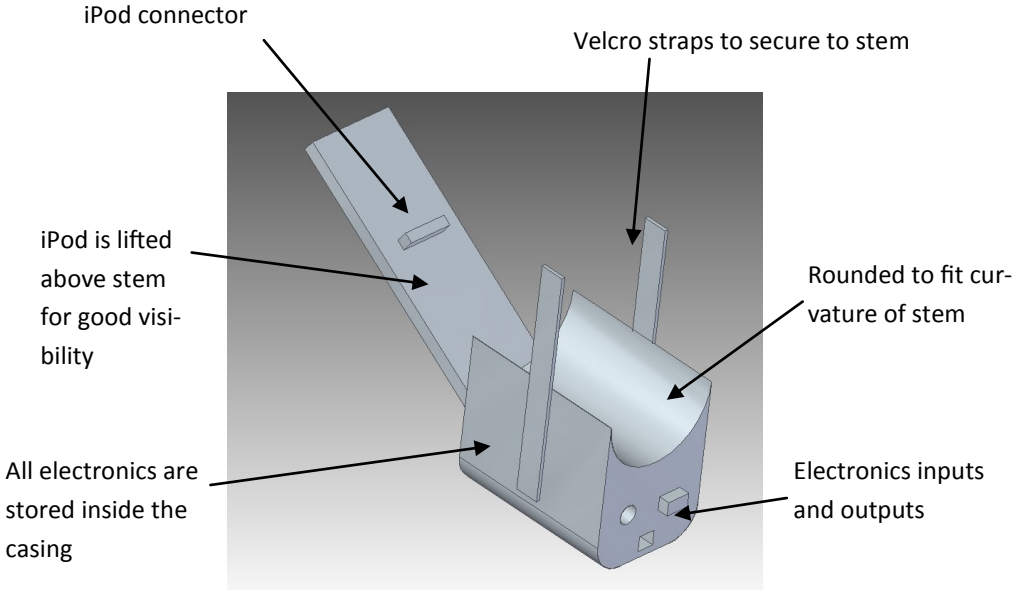


Casing Ideas



Idea #1 — Top Tube Mount

This idea involves the main casing being cable tied to the top tube of the bike. This will provide a secure fixing and prevent the product having an effect on the riders steering. The iPod will sit on top of the casing which houses the electronics and power supply. One issue with this is that for the rider to see the display of the iPod they will have to look down, potentially putting them off balance. The power supply could be any form of batteries, however I think a mains power supply would get in the way especially in this position. There will be a piece of 4 way ribbon cable going to the capacitive touch sensor on the handlebars; I think as long as the wire is sufficient in length this shouldn't be too much of a problem.



Idea #2 — Stem / Out front mount

Instead of attaching to the top tube, this design will hang (although secured tightly) from the stem of the bike. The main circuitry and power supply will be enclosed in this casing however to ensure the rider can easily see the iPod display, the iPod will be mounted out in front of the casing. Instead of cable ties this design uses Velcro straps to make it easily removable between training and actual riding. Although this will add some weight to the handlebars and may influence the riders steering, there will be no need for a wire to connect the frame and handlebars because everything will be mounted on the handlebars.

Environmental Limitations

	Top Tube	Stem width	Stem Length
Willier	43	30	46
Pinarello	39	33	53

Frame Measurements

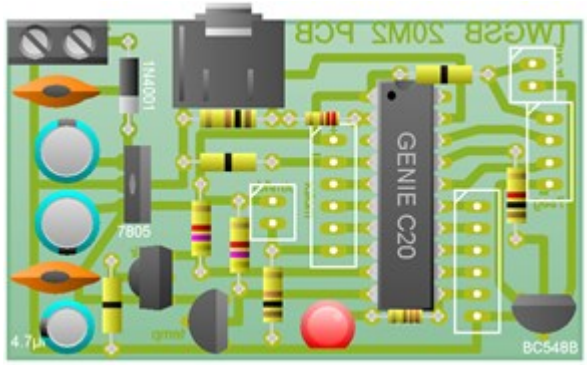
Cables may get in the way

https://www.sparkfun.com/products/339

1000 mAh Lipo Battery + Charger

50.8 x 33.5 x 5.9 mm

25 x 21.5 x 12 mm



Main PCB

65 x 40 mm

Shuffle

29.0 × 31.6 × 8.7 mm

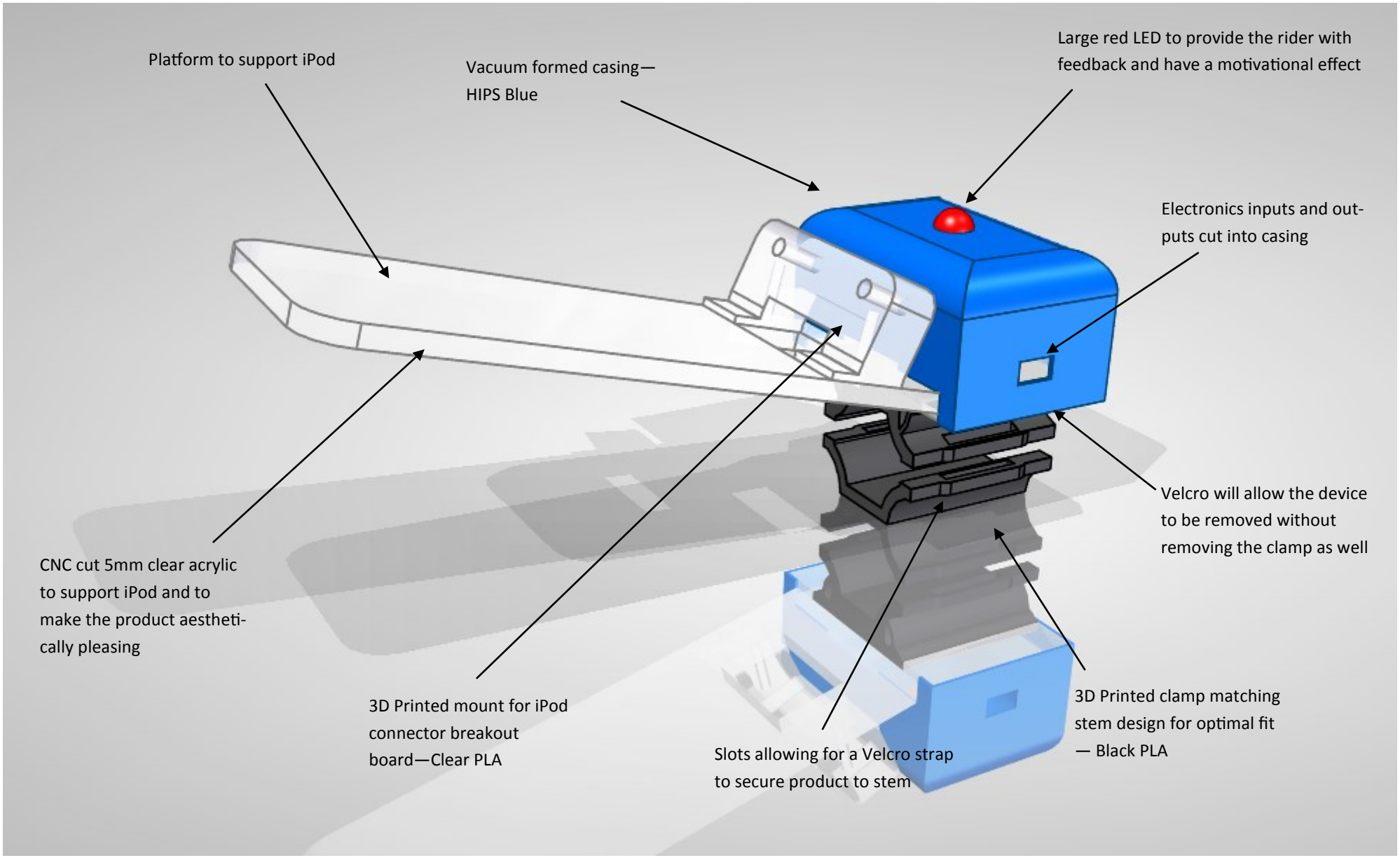
Nano

76.5 × 39.6 × 5.4 mm

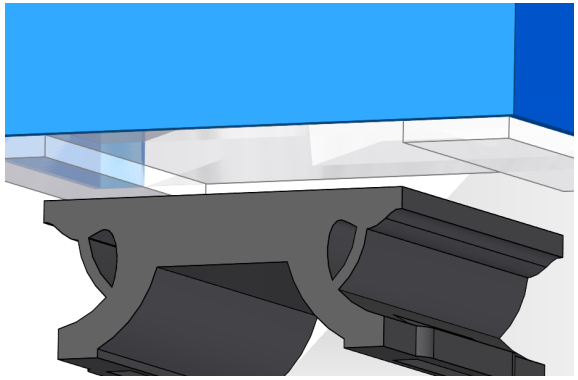
Touch

123.4 × 58.6 × 6.1 mm

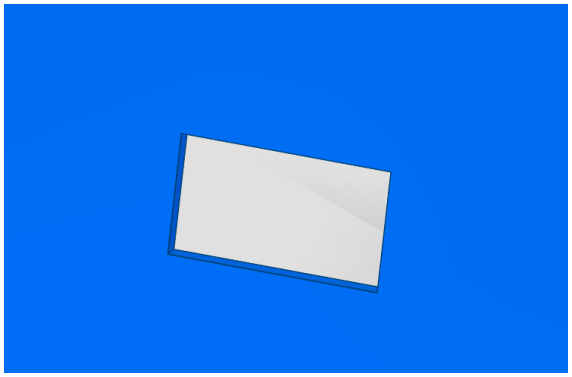
Final Casing Design



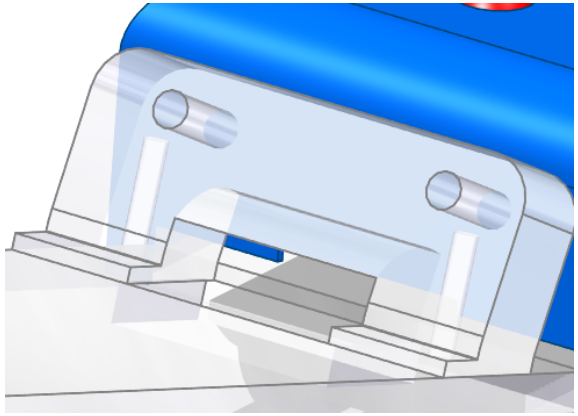
Final Casing Design — Key Features



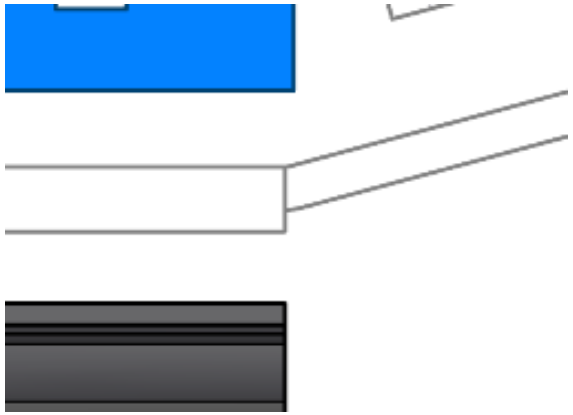
The stem clamp will be removable and will temporarily attach using Velcro. The two acrylic tabs visible in the picture will ensure the parts fit together correctly and do now twist.



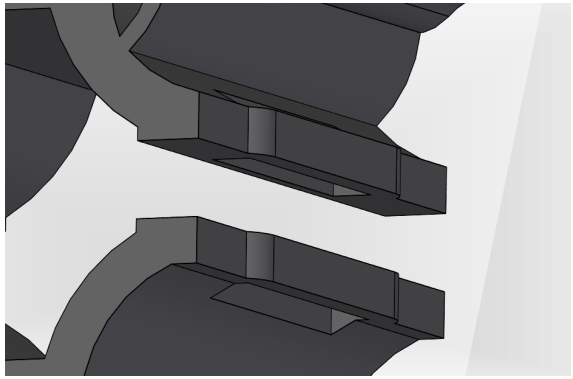
Once the vacuum formed casing has been made, I will trim holes in the sides to allow connectors to protrude through the case.



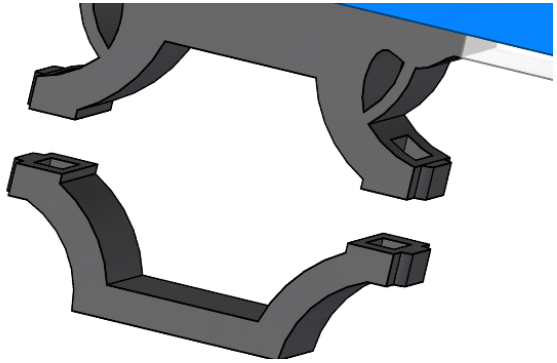
The iPod connector breakout board will attach to this 3D printed mount. It will ensure the ipod lays flat with the acrylic and will raise it up a little making it easier to see



This is a side view of the piece of 5mm acrylic which will support the iPod. It is angled at roughly 10 degrees to make it easier for the rider to see the iPod screen.

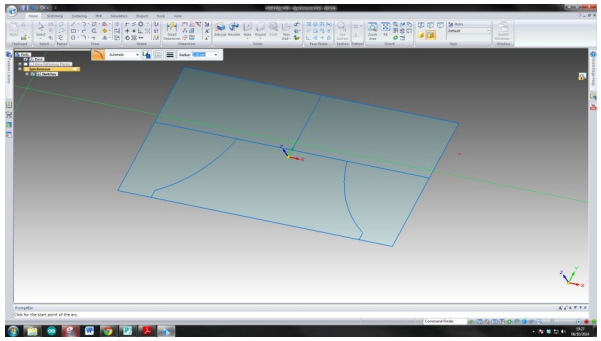


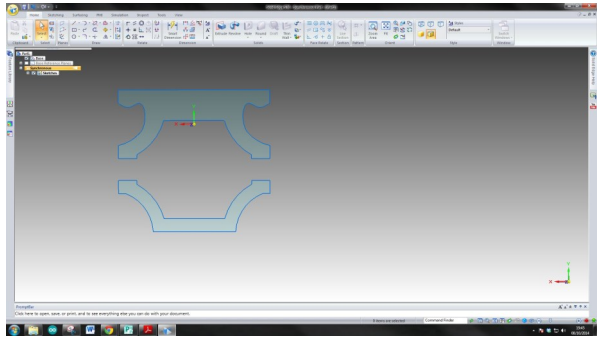
The tabs built onto each side of the two clamp pieces will be used for holding the two pieces together. This will be done through a Velcro strap attaching the two.

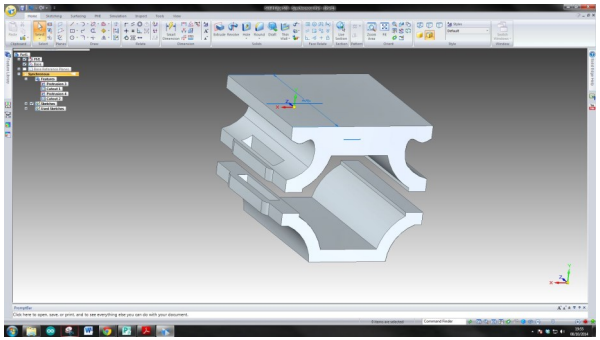


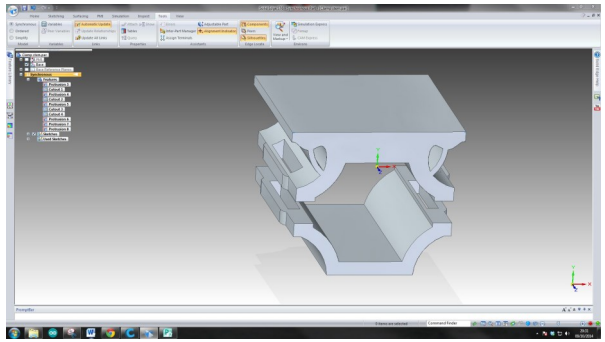
The clamp assembly will be 3D printed and can be custom designed to the stem in question. This particular stem is an odd shape making 3D printing the ideal manufacturing method to produce it.

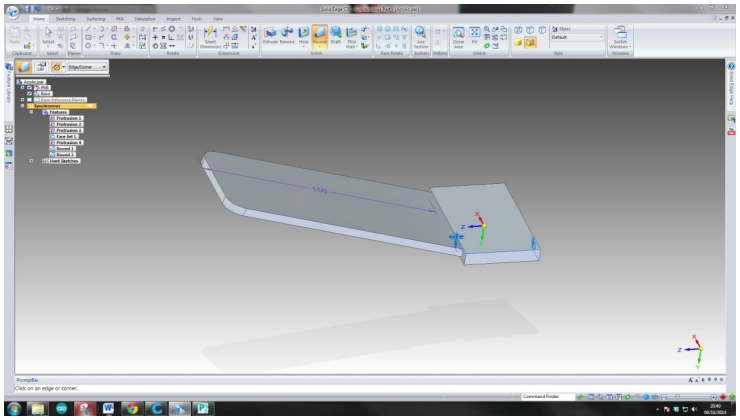
Design Process

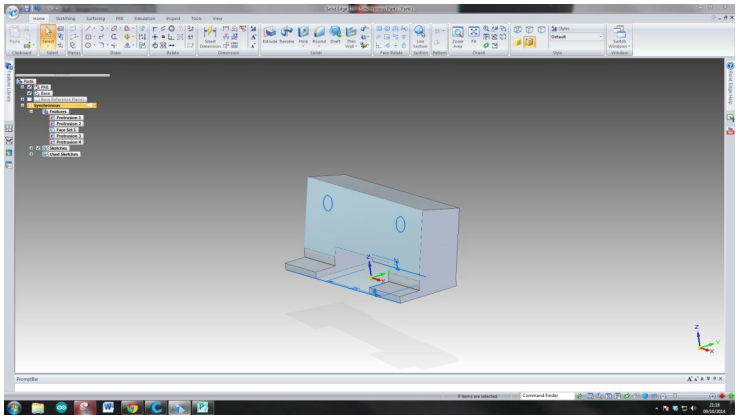


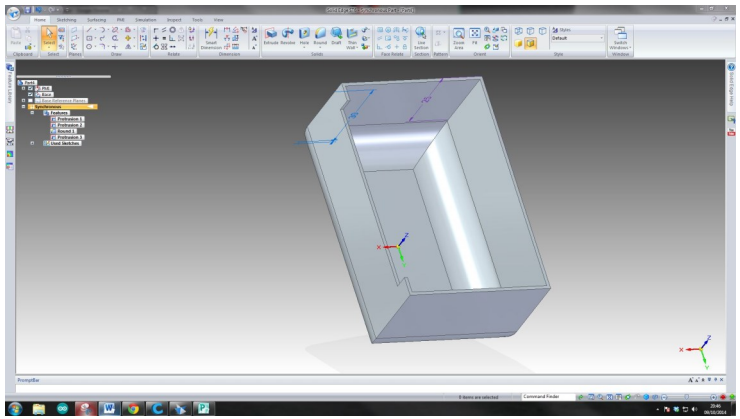




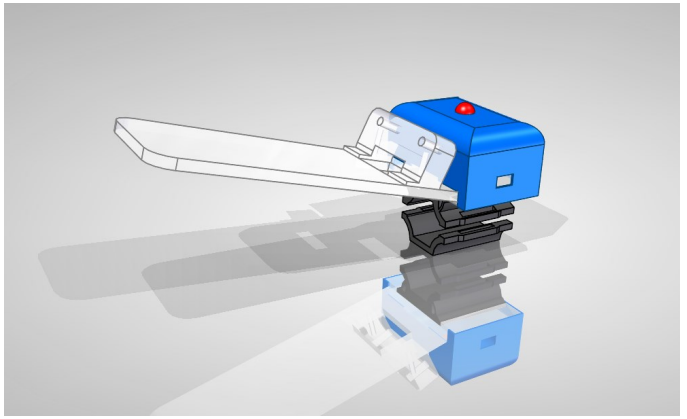




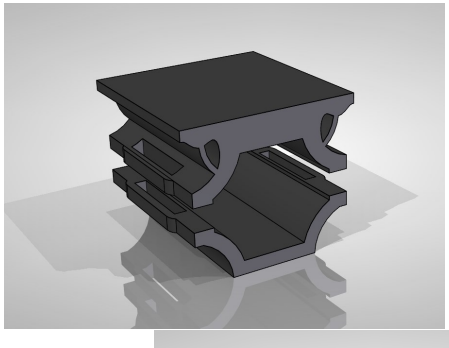


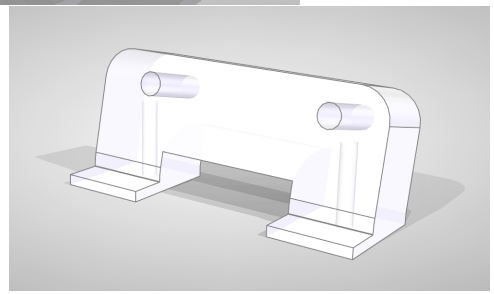


I have designed my whole product in CAD to ensure each of the separate elements fits together well. Also for the parts which will be 3D printed it is necessary to design a CAD file in order to print the part. For the main stem clamp I started of drawing the profile and then I extruded it. Once I had done this I added some tabs to allow it to be secured to the stem and finally I added some reinforcements for extra strength. I then did a rough design of the remaining parts: the acrylic, iPod connector mount and vacuum formed casing. The acrylic design will need to be copied across into 2D design so it can be cut on the CNC router. The iPod connector mount will be 3D printed so the CAD file is ready to go. I will create the mould for the vacuum forming using measurements from the CAD software. Once everything was designed, I created an assembly to ensure everything fitted together well.



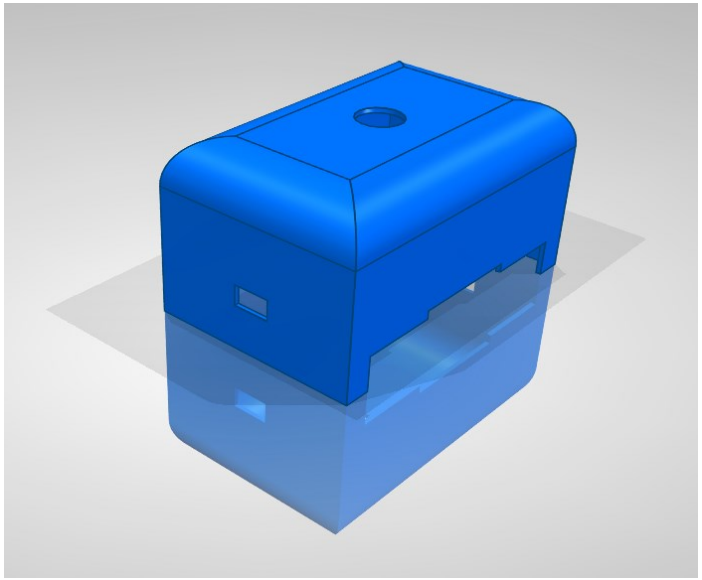
Manufacture Methods





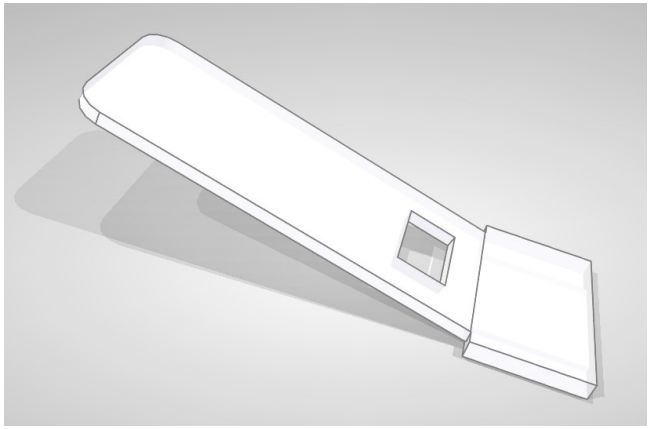
3D Printed parts

Because these parts are of a complex shape and require a high accuracy to interface with other parts, I have decided to 3D print them. The stem clamp will be printed in black PLA and the iPod connector will be printed in clear PLA to blend in with the clear acrylic. I have designed the two parts in CAD meaning they can be exported as an STL file and then sliced to produce the appropriate G-code file which the 3D printer will run from. Although PLA is not the strongest material, it can be printed at very low temperatures, massively reducing the amount it warps during cooling.



Vacuum Formed

I have opted to vacuum form the main casing as it will provide a glossy finish and light but strong casing. I will either take measurements from the CAD file and handcraft the mould out of MDF or I will 3D print it based on the CAD file. Once the basic shape is formed I can use a rotary tool such as a Dremel to cut out the holes for USB charging port, capacitive touch buttons, LED and the slit for the cable. The material used will be HIPS (high impact polystyrene) as it is thin but strong.



CNC Router + Strip Bender

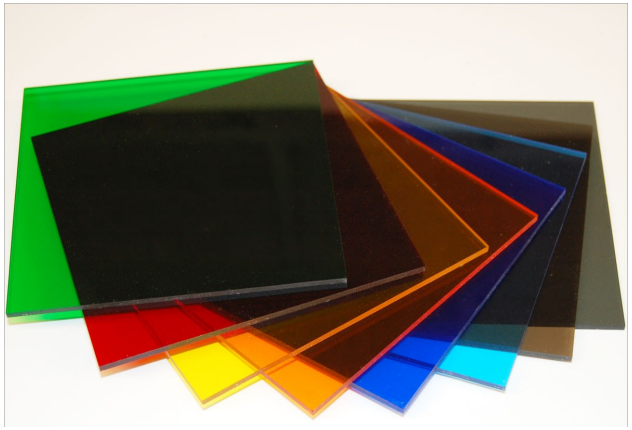
The first thing I will need to do is generate a 2D design file based on the measurements from my CAD file. From this I will use the CNC router to cut a piece of clear 5mm acrylic to the correct shape and also remove the hole for the iPod connector. Once the acrylic is cut I will polish the edges and finally use the strip bender to bend the acrylic as shown in the render above. I chose to use clear acrylic as it is strong and will make the product much more aesthetically pleasing.

Materials

http://imaginationinspace.com/

PLA — Polylactic Acid

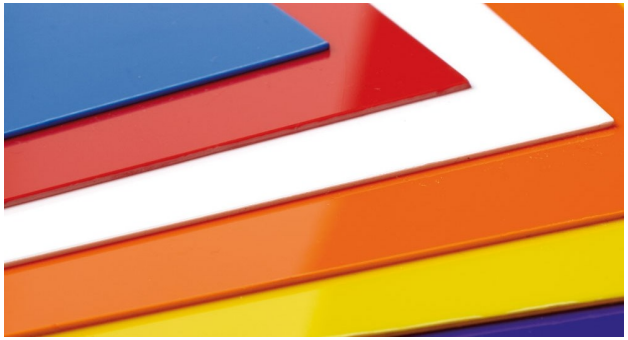
PLA is a common material used for 3D printing. This is because it has a relatively low melting temperature and it has proven to be very effective. It is referred to as a bio polymer as it produced from corn starch and is part of a fast growing family of natural plastics. Because it is made from corn starch it is far more environmentally friendly than most plastics and more sustainable too. Due to its low melting point it can also be heated up and returned to its monomer state — this means it can be easily recycled to a virgin state.



http://www.tapplastics.com/

Acrylic — Perspex

Acrylic is formed from acrylic acid and takes the form of PMMA plastic (polymethyl methacrylate). It is a good material as it has very similar properties to glass however it is considerably stronger and more lightweight. It is also very easy to work with as it can be easily cut using saws, drills or milling machines making it a good material for my project. The main disadvantage to it is the price however its other properties make up for this. Unfortunately unlike PLA it is significantly harder to recycle, this is because its manufacturing process involves several highly toxic chemicals and a lot of energy is required to reuse it. Therefor it is not the most sustainable material.

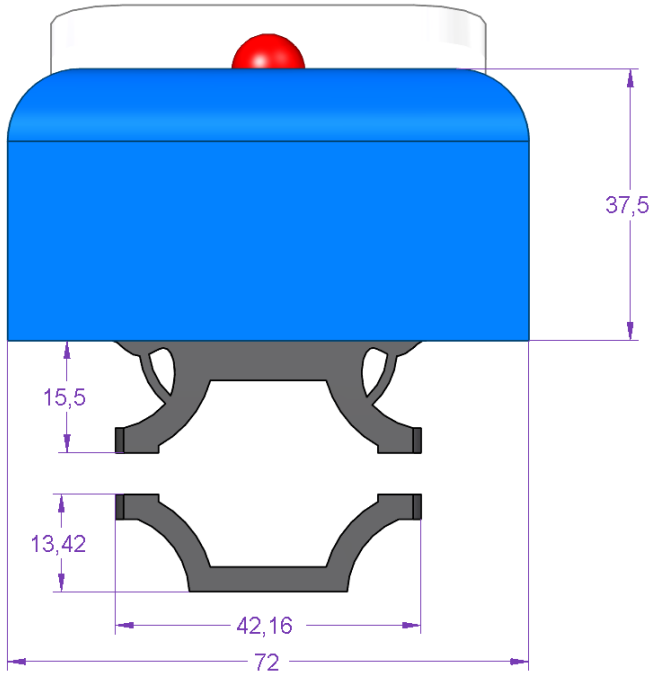


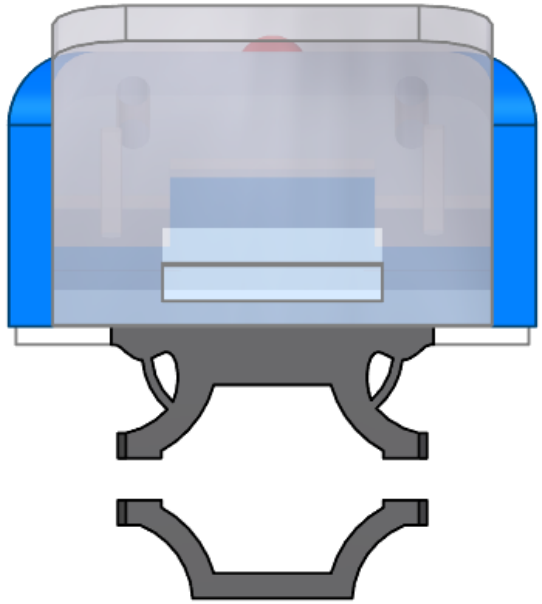
http://www.homecrafts.co.uk/

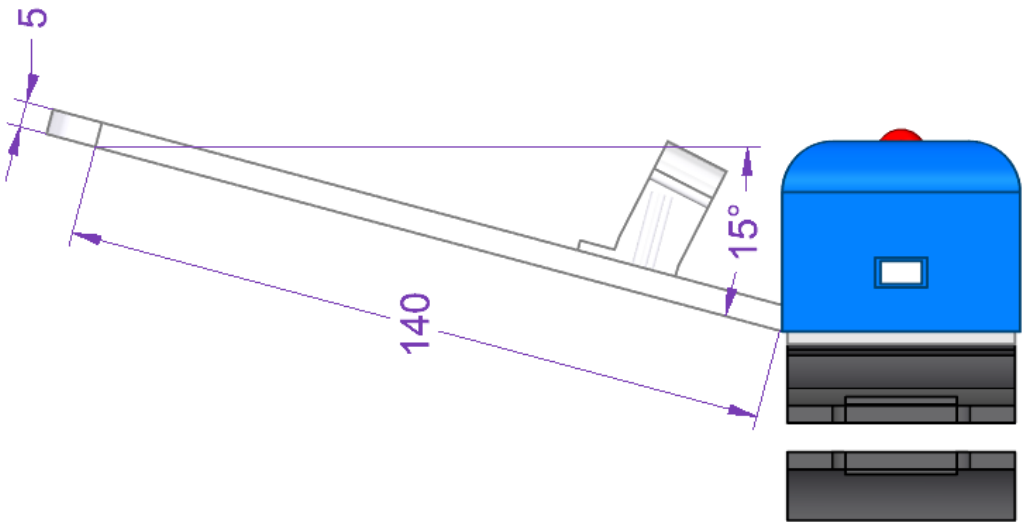
HIPS — High Impact Polystyrene

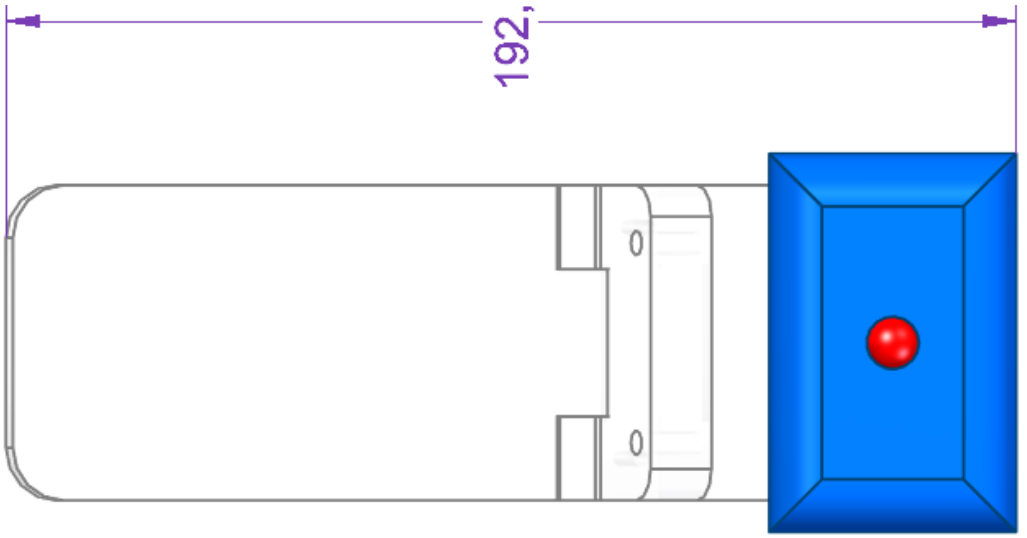
High impact polystyrene comes in sheets roughly 1mm thick and is used for vacuum forming. This is because it softens at a relatively low temperature — 95 degrees. Similar to acrylic or PMMA, HIPS is very easy to work with as it is easy to cut and manipulate making it ideal for this project. It is also produced in a wide range of colours, like both other plastics, making it aesthetically pleasing. HIPS is also relatively easy to recycle making it much more environmentally friendly than the PMMA. Not only this but it is much cheaper than acrylic and considered to be very cost effective.

Orthographic View

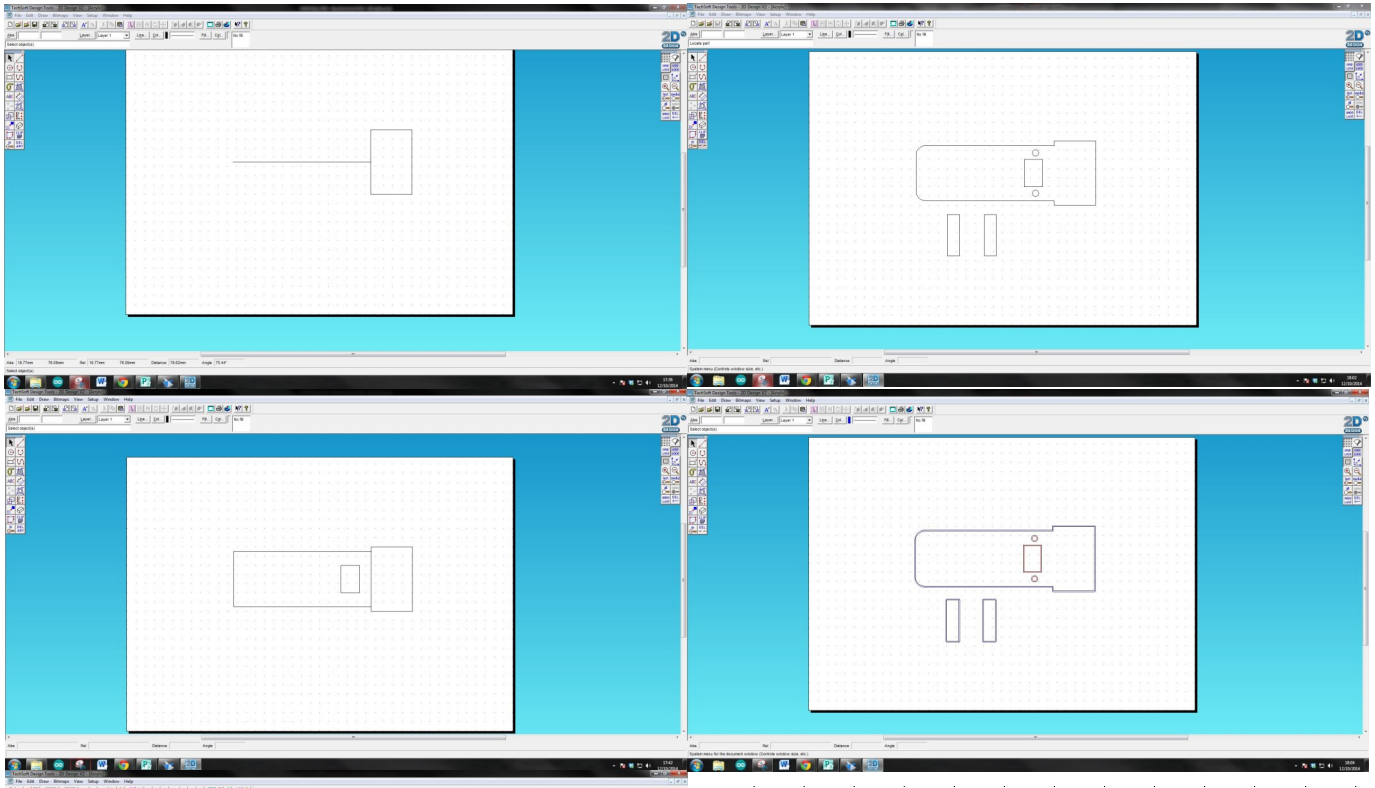


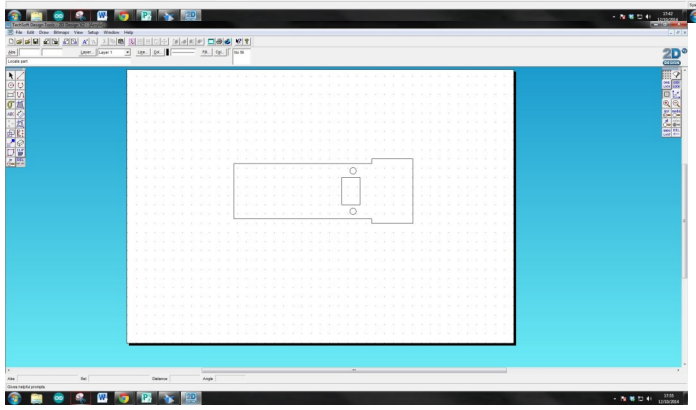




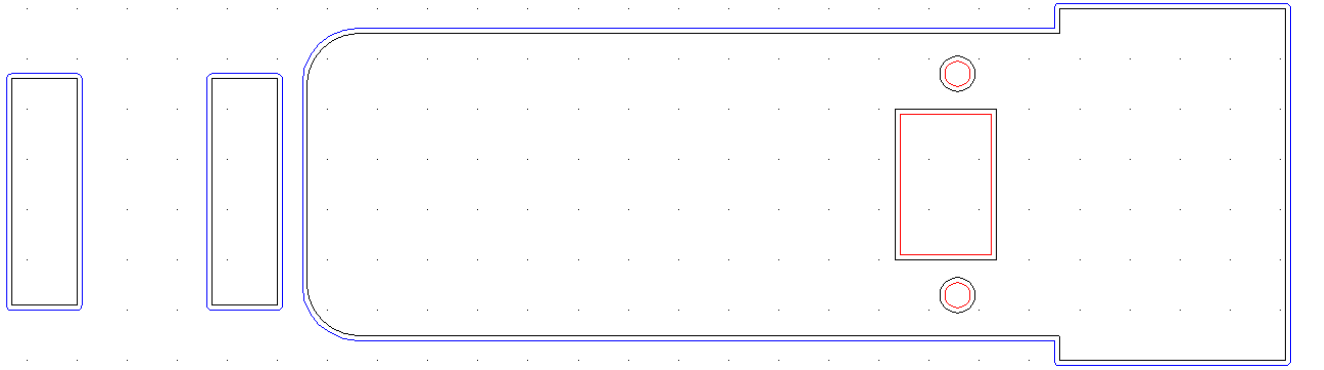


2D Design





Here is the 2D design of the piece of 5mm acrylic which will act as a base to the electronics and hold the iPod for the rider to see. The 2D design file allows it to be cut on the CNC machine for high accuracy. Once cut, I will sand and polish the edges as well as filing the inner corners. After this I will use the strip bender to put roughly a 10 degree bend in it at the point where it narrows — this will raise the iPod up making it easier to read. The acrylic has the holes to allow the iPod mount to screw into place pre cut. The two tabs on the left will be glued each side on the wider section to prevent the acrylic base plate from twisting when it is Velcro'd onto the stem clamp. The acrylic will be 5mm clear acrylic for maximum strength and to be aesthetically pleasing.



Code

```
'Pinouts

'C.2 - Cadence Sensor
'B.7 - LED
'B.6 - Button - Back
'B.5 - Button - Shuffle
'B.4 - Button - Play
'B.3 - Button - Forwards
'B.0 - Serial out
'B.1 - Serial in

setfreq m32'run clock at 32mhz

symbol shuffleState = b0
shuffleState=0
symbol playlistLength = b1
playlistLength = 0
symbol song = b2
symbol checkSum = b3
symbol touchValue = b4
symbol cadence = b5
symbol revs = w3
symbol timerr = w4
symbol division = b10
symbol lastSong = b11

'====================================================================
'====================================================================
'Setup
'====================================================================
'====================================================================

setup:
pwmout pwmdiv64, C.3, 124, 250'PWM for timer
touch B.4,touchValue'Read touch value
gosub flashLed'Flash LED
gosub flashLed
gosub flashLed

'====================================================================
'Setup-number of songs
'====================================================================

do
touch B.6,touchValue'Read touch value
if touchValue>100 and playlistLength<255 then 'If back button pressed..
gosub skipBack'Move song backwards on iPod
dec playlistLength'Decrease playlist length
gosub flashLed'Flash LED
endif

touch B.3,touchValue'Read touch value
if touchValue>100 and playlistLength>1 then 'If forward button pressed.
gosub skipForwards'Move song forwards on iPod
inc playlistLength'Increase playlist Length
gosub flashLed'Flash LED
endif

touch B.4,touchValue'Read touch value
loop while touchValue>100 'While play/pause button not pressed loop
if playlistLength = 0 then'If playlist length not entered...
read 0,playlistLength'Read last playlist length from EEPROM
else
write 0,playlistLength'Otherwise write the new value to the EEPROM
gosub flashLed'Flash LED
gosub flashLed
endif
gosub flashLed'Flash LED
gosub flashLed
gosub flashLed

```

Here is the setup part of my code; this code is only executed once, this is every time the device is powered on to ensure everything is setup correctly. The first thing to happen is the clock is set to 32mhz, this is necessary in order to communicate with iPod at the correct baud rate. It is the fastest internal clock built in to the Picaxe, so it also means the chip is running as fast as possible, enhancing the user experience. Next I have defined several variables as symbols to make the code easier to understand. Once this is done, pin C.3 is setup as a PWM with a frequency of 1000Hz; this is used as timer later on in the code to measure the time of every 3 crank rotations. The LED is flashed to tell the user the initial setup is complete and they are now required to select the length of there playlist.

The playlist length is selected using the fast forwards and rewind capacitive buttons on the controller. Every time a button is pressed the LED flashes and the iPod will switch to that song in the playlist. The idea is that the user will scroll through to the last song in the playlist and then press the play/pause button t confirm this is the final song. The Picaxe can then use this value to match RPM to song BPM and select the appropriate song.

If the user skips this stage by pressing the play/pause button straight away then the last value will be read from the EEPROM to save the user time if there are no changes to be made. However if they do make changes to the length this will be stored in the variable and saved to the EEPROM for next time the device is reset. All of this is shown visually via the LED which has different number of flashes depending upon the action that has been carried out.

```
main:

'=============================================================================
'Monitors buttons
'=============================================================================

touch B.6,touchValue'If back button pressed...
if touchValue>100 then gosub skipBack
touch B.5,touchValue'If shuffle button pressed...
if touchValue>100 then gosub shuffle
touch B.4,touchValue'If play.pause button pressed...
if touchValue>100 then gosub playPause
touch B.3,touchValue'If forwards button pressed...
if touchValue>100 then gosub skipForwards

'=============================================================================
'Count pulses from cadence sensor
'=============================================================================

readadc C.2,cadence'Read value from cadence sensor
if cadence > 100 then'Is crank present...

if revs=0 then'If not currently counting...
timerr=0'Reset timer
setint %00010000,%00010000'Start interrupt on pin linked to PWM at 1000Hz
endif
inc revs'Increase revs

if revs >=3 then'If revs = 3
setint off'Turn of interrupt
timerr=timerr/3'Take average time
revs=60000/timerr'Calculate RPM
cadence = revs'Set cadence to RPM
revs = 0'Clear revs to restart timer next revolution

'=============================================================================
'select appropriate music
'=============================================================================

division=120/playlistLength'Scale playlist Length to RPM
song=cadence/division
if lastSong=0 then 'If this is first song select new song...
gosub flashLed'Flash LED
gosub flashLed
gosub selectSong 'Switch iPod to new song
endif

if lastSong!=song then 'If song is different from the last song...
gosub flashLed'Flash LED
gosub flashLed
gosub selectSong 'Switch iPod to new song
endif

lastSong=song'Set last song to song to detect change in song

endif

endif

goto main

'=============================================================================
'=============================================================================
'Interrupt
'=============================================================================
'=============================================================================

interrupt:'interrupt to count milliseconds
inc timerr 'increase
setint %00010000,%00010000'set interrupt up again
return
```

Next is the main part of the program, this part is continually looped indefinitely. Several things occur in this loop, firstly the Picaxe checks if any buttons are being touched. If they are the appropriate action is carried out by jumping to a sub routine.

There is also some code designed to detect the cyclists cadence. This is updated every 3 revolutions of the cranks, the variable “revs” is used to count this. On the first revolution a hardware interrupt is introduced which is connected to the PWM pin. The interrupt occurs at a frequency of 1000Hz and each time increases the variable “timer” by one; effectively counting the milliseconds since the first revolution of the pedals. Every rotation is counted and then on the third the RPM is calculated. This is done by dividing the total time for all 3 revolutions by 3 to calculate the average time per revolution. Then 60 is divided by this number to calculate the riders cadence in RPM.

Once the RPM has been calculated, an appropriate song is selected. That is if the RPM has changed since the last reading — the idea behind this being that the rider must sustain there cadence to continue listening to a certain song. The software assumes the max cadence is 120, this is rider specific so could be changed depending on who is using the device. This number is divided by the number of songs in the playlist and the resultant number is then used to calculate the song which should be selected to match the cadence.

At the bottom is the interrupt which is very short to ensure it isn’t called half way through. It simply increases the timer variable and then sets the interrupt up again. Then it returns to where it was in the code when the interrupt was called.

```
'=============================================================================================
'=============================================================================================
'Sub routines
'=============================================================================================
'=============================================================================================

selectSong:
serout B.0,T19200_32,(0xFF,0x55,0x03,0x00,0x01,0x04,0xF7)'change to mode 4 - AIR Remote
pause 1
checkSum = 190 - song'generate checksum
checksum = checksum & 255
serout B.0,T19200_32,(0xFF,0x55,0x07,0x04,0x00,0x37,0,0,0,song,checkSum)'jump to song
return

skipBack:
serout B.0,T19200_32,(0xFF,0x55,0x03,0x00,0x01,0x04,0xF7)'change to mode 4 - AIR Remote
pause 1
serout B.0,T19200_32,(0xFF,0x55,0x04,0x04,0x00,0x29,0x04,0xCB)'skip back
pause 1
return

skipForwards:
serout B.0,T19200_32,(0xFF,0x55,0x03,0x00,0x01,0x04,0xF7)'change to mode 4 - AIR Remote
pause 1
serout B.0,T19200_32,(0xFF,0x55,0x04,0x04,0x00,0x29,0x03,0xCC)'skip forwards
pause 1
return

playPause:
serout B.0,T19200_32,(0xFF,0x55,0x03,0x00,0x01,0x04,0xF7)'change to mode 4 - AIR Remote
pause 1
serout B.0,T19200_32,(0xFF,0x55,0x04,0x04,0x00,0x29,0x01,0xCE)'toggle play/pause
pause 1
return

shuffle:
serout B.0,T19200_32,(0xFF,0x55,0x03,0x00,0x01,0x04,0xF7)'change to mode 4 - AIR Remote
pause 1
serout B.0,T19200_32,(0xFF,0x55,0x03,0x04,0x00,0x2C,0xCD)'Gets shuffle mode
pause 1
if shuffleState=0 then
serout B.0,T19200_32,(0xFF,0x55,0x04,0x04,0x00,0x2E,0x01,0xC9)'Sets shuffle on
else
serout B.0,T19200_32,(0xFF,0x55,0x04,0x04,0x00,0x2E,0x00,0xCA)'Sets shuffle off
endif
return

flashLed:
high B.7'LED on
pause 200
low B.7'LED off
pause 200
return
```

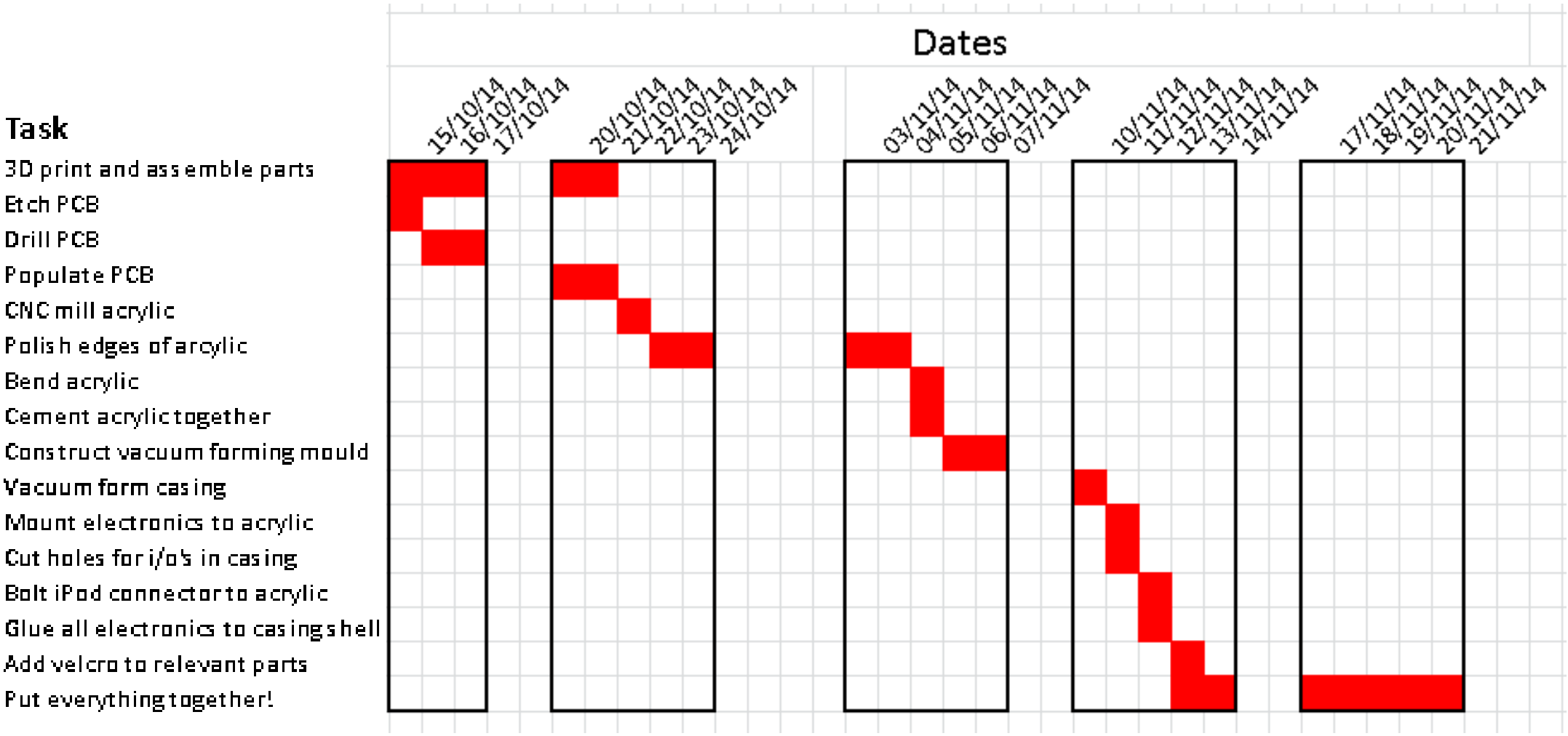
Finally there are 5 sub routines written to control the iPod in 5 different ways. There are 4 standard commands, skip back, skip forwards, play/pause and shuffle. All of these commands, I have calculated the serial data using a spreadsheet I created (pictured bottom). This saves memory on the Picaxe and makers programming easier; however for the command used to select a song number this would be inefficient. So in this case the Picaxe has to calculate the value of the check sum and substitute this into the command. This is done by subtracting the song number from 190 and then ANDing the value with 255. The rest of the command is pre defined so once this is calculated the command can be sent.

There is also a sub routine used to flash the LED. This makes the code more efficient as it is something which is done repetitively on many occasions.

Below is the spreadsheet used to calculate iPod control commands; it does all of the DEC to HEX conversions, calculates the check sum and then concatenates the data into a string which is the final command.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Header		Length	Mode	Command				Parameters				Checksum							
2			7	4					0											
3	0xFF	0x55	0x07	0x04	0x	0	0x	37	0x00				0xBE		0xFF	0x55	0x07	0x04	0x0	0x37	0x00	0xBE
4			07	04					00				BE									
5			7	4		0		55	0	0	0	0	190									
6																						

Plan of Manufacture



Plan of Manufacture

Task	Dangers	Safety Precautions	Tools
3D print and assemble parts	Hot parts	Act sensibley	3D printer
Etch PCB	Corrosive acid	Use protective equipment - goggles gloves etc	Etch tank
Drill PCB	Sharp drill bit	Act sensibley	Mini drill
Populate PCB	Hot soldering iron	Act sensibley	Soldering iron, wire cutters
CNC mill acrylic	Moving machinery	Always shut the lid	CNC mill
Polish edges of acrylic	Sharp edges	Don't run finger down edge	Glass paper, polish
Bend acrylic	Hot strip bender	Act sensibley	Strip bender
Cement acrylic together	Strong glue!	Act sensibley	Solvent cement
Construct vacuum forming mould	Sharp tools	Ensure hands are not in the way of sharp tools	Saw, sanding block
Vacuum form casing	Hot plastic	Wear heat proof gloves	Vacuum former
Mount electronics to acrylic	None		
Cut holes for i/o's in casing	Sharp tools	Ensure hands are not in the way of sharp tools	Rotatry tool
Bolt iPod connector to acrylic	None		
Glue all electronics to casing shell	Hot glue is hot	Act sensibley	Hot glue
Add velcro to relevant parts	None		
Put everything together!	None		

Risks will be minimized sufficiently if I act with caution and sensibly at all times whilst in the workshop.

Capacitive Touch Button Clamp

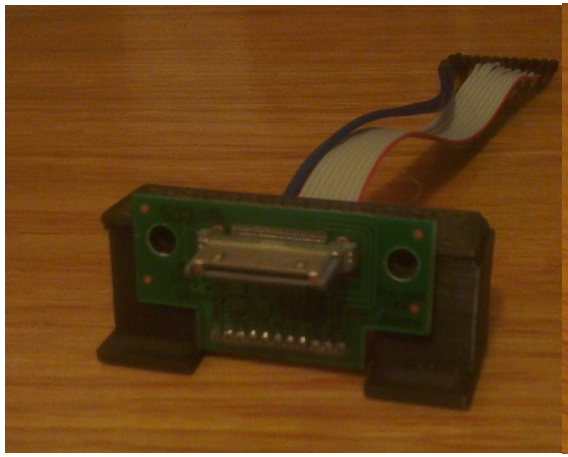
Here is one part of the two part assembly being 3D printed. It is printed in two separate parts to ensure the hinge works freely. The part was printed with thick walls (1.2mm) for maximum strength when the clamp is tightened and a thin layer height (0.6mm) making the layers less obvious and therefor more aesthetically pleasing.

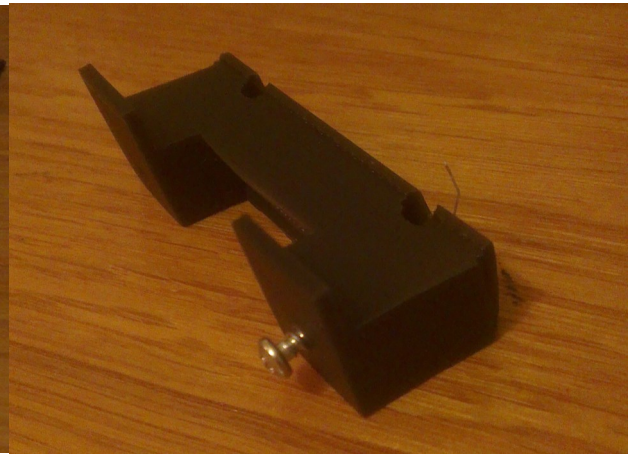
Here is the other half of the clamp after printing. A little bit of clean-up is required before the two parts can be fitted together. Support material for the hinge must be removed and the strings were the print head has jumped across must be snapped off and sanded down. The holes did not have to be drilled out as I compensated for print error in the CAD software.

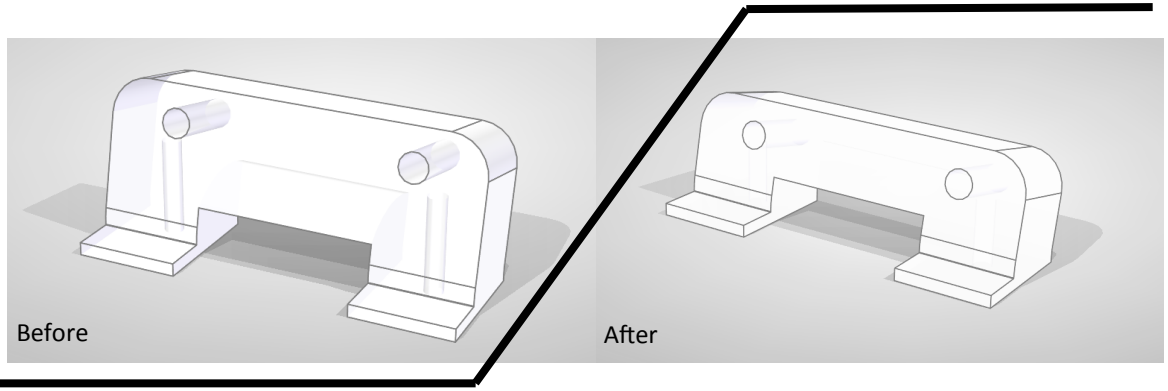
And the final product. Once the two parts had been cleaned up I was able to bolt it all together using M3 bolts and nyloc bolts to prevent vibrations from loosening them. Although the PCB is not finished yet, I have tested it with a piece of PCB and it fits very nicely.

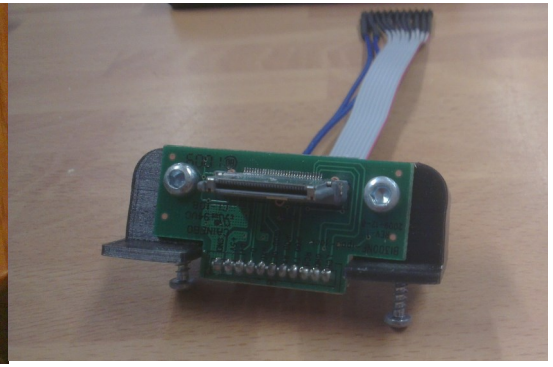
Here is the clamp fitted onto a bike, it clamps effectively and will allow the buttons to be moved on two different planes.

iPod Connector Mount







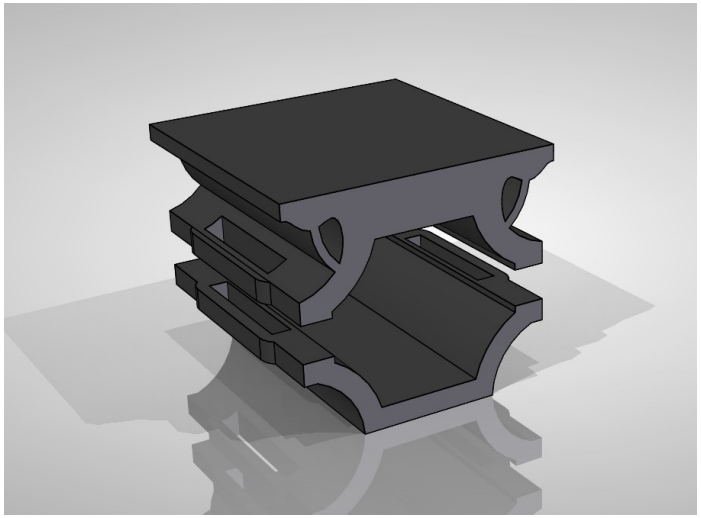


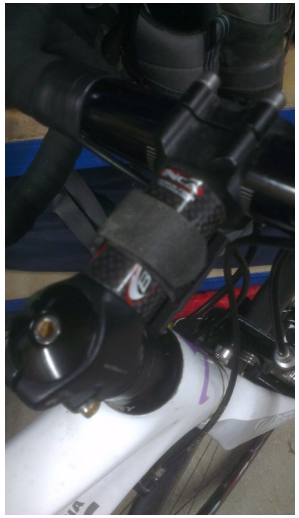
Here is my first prototype of the iPod connector mount; unfortunately both the x and y axis on the 3D printer slipped about 60% of the way through the print. However it was far enough to provide me with a good test item.

1. After connecting my iPod to the connector and then sitting in flush on the mount I have established that the mount is at the correct angle — although the iPod will need some extra support to take some weight of the connector.
2. The holes for the screws to clamp the mount to the acrylic are both the correct width and depth to allow for good purchase.
3. Unfortunately my measurements let me down and I positioned the mounting holes roughly 6mm to high. This makes the connector quite bulky and does not use spaces effectively. Although other than this the holes are of the correct spacing and size for M4 bolts.

So as a result of this prototype I have made a few changes to the CAD model. Firstly I have lowered the two mounting holes by roughly 6mm. This will mean more of the connector breakout board is in the hole cut through the acrylic making it more compact. As a result of this adaption I have had to reduce the depth of the mounting holes for the acrylic — this may mean I have to trim the screws down.

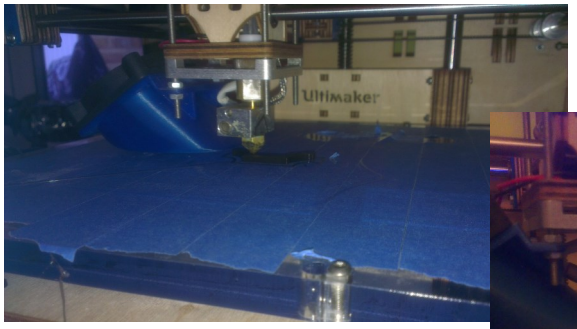
Stem Mounting Bracket





Quality	
Layer height (mm)	0.20
Shell thickness (mm)	0.8
Enable retraction	☑
Fill	
Bottom/Top thickness (mm)	0.8
Fill Density (%)	30
Speed and Temperature	
Print speed (mm/s)	50
Printing temperature (C)	205
Support	
Support type	None
Platform adhesion type	None
Filament	
Diameter (mm)	3
Flow (%)	100.0

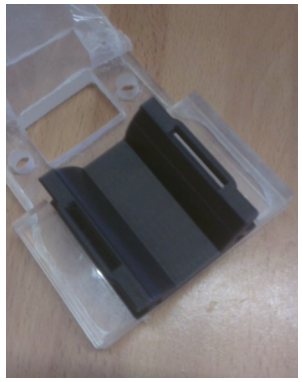
Machine	
Nozzle size (mm)	0.4
Retraction	
Speed (mm/s)	40.0
Distance (mm)	4.5
Quality	
Initial layer thickness (mm)	0
Cut off object bottom (mm)	0
Dual extrusion overlap (mm)	0.15
Speed	
Travel speed (mm/s)	100
Bottom layer speed (mm/s)	20
Infill speed (mm/s)	150
Outer shell speed (mm/s)	20
Inner shell speed (mm/s)	50
Cool	
Minimal layer time (sec)	6
Enable cooling fan	☑



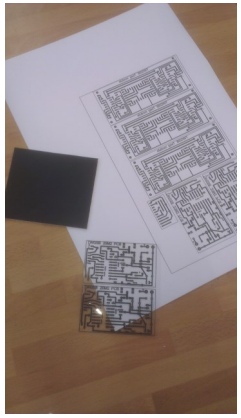
I used a 3D printer to produce this part because it allows for a complex shape to be made easily and accurately. I used the CAD model I designed earlier to produce the G-CODE file for the printer using Cura as the slicer. I used the above settings to get a high quality print suitable for the job. It has a thick shell (0.8mm) to increase the strength wrapping around the stem. 30% infill also provides a lot of strength without compromising the print time and weight of the final product. The layer height was 0.2 because the print does not have any curved over hangs and accuracy on the z axis is not essential. Finally I printed it at 150mm/s infill and 20mm/s outer wall. This keeps the print fast whilst giving a clean finish on the outside. This makes it aesthetically pleasing for the user. Once one half had printed, I tested it on the stem of the bike to check it fit; it did. So I printed out the second half.

Milling and Sanding Acrylic

From the 2D design file I was able to use the CNC milling machine to cut the 6mm clear acrylic to the shape I designed earlier. Because of the thickness of the acrylic multiple passes were required to cut all the way through. The combination of this and the milling bit meant when the acrylic was finally cut the edges were very rough. For my product the visual aspect is important so I first filed the edges down. Once I had done this I used some coarse glass paper and then progressed to finer grade paper. This left me with very smooth edges however they were dull and still did not have the visual aspect that I was looking for. Finally I polished the edges using Brasso polish, leaving me with smooth clear edges which are very aesthetically pleasing. Once the edges were polished I used solvent cement to attach the tabs onto the main piece of acrylic. Then I hot glued the Velcro to the stem mount and the acrylic. This allows for the device to be easily attached and removed.



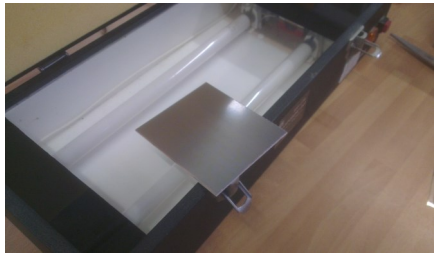
PCB Manufacture



Printout of design onto acetate

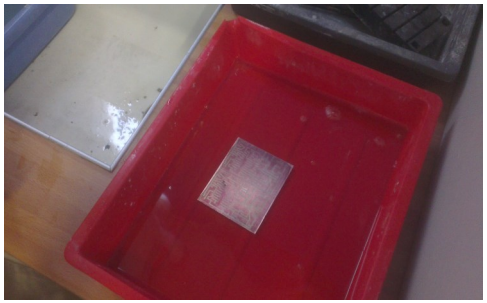
Design taped to photosensitive side of PCB

PCB exposed to UV light, transferring design

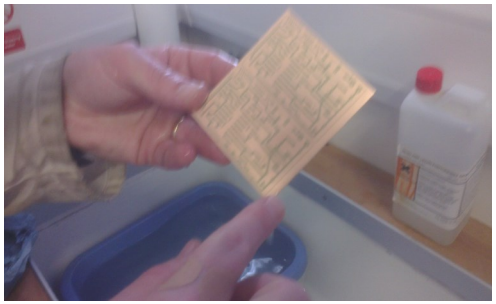


Design transferred to photoresist

PCB developed removing unwanted photoresist



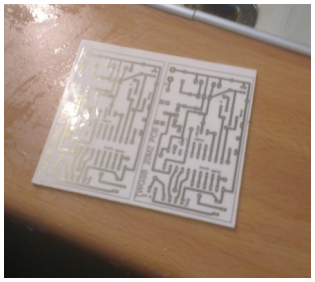
Board partially developed



Design covers over copper in photoresist

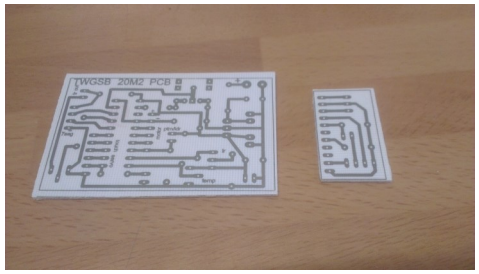
PCB put in bubble etch tank to remove unwanted copper

Once the all unwanted copper is removed by acid, board rinsed



Final PCB

PCB cut to size using guillotine



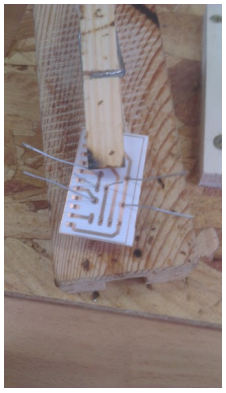
PCBs ready for drilling, photoresist removed from copper using PCB rubber

Holes drilled using 1mm drill bit

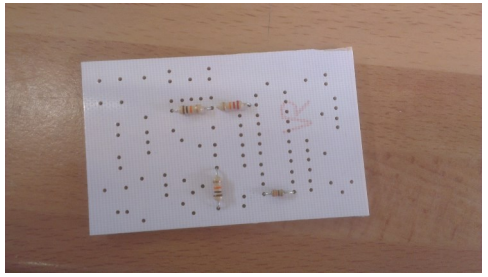
Larger holes for mounting drilled

Board sanded using disc sander for rounded edges

PCB Population

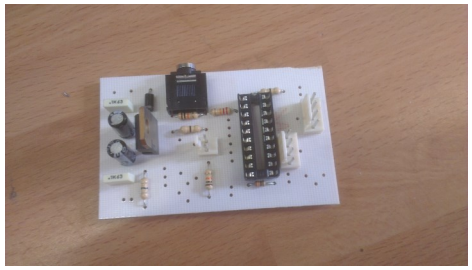
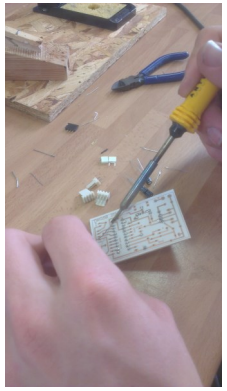
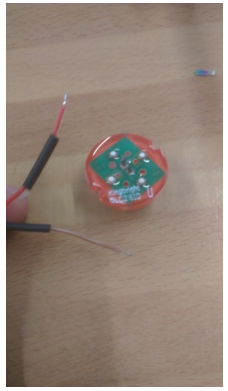
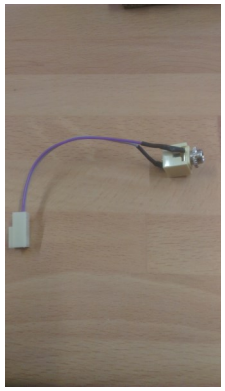
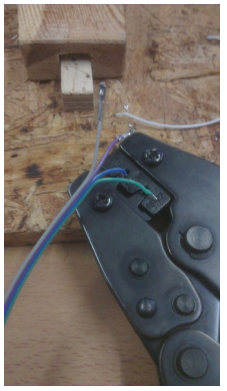
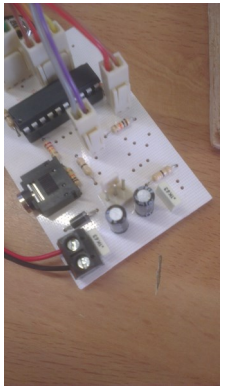
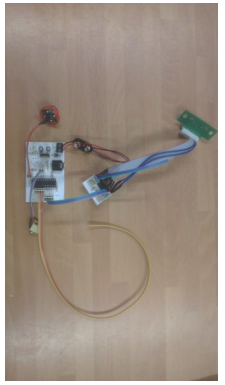





Collecting components

iPod connector breakout population

Motherboard population

Motherboard population

Crimping off board components

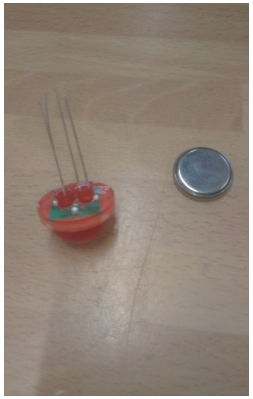
Final Assembly

Testing and

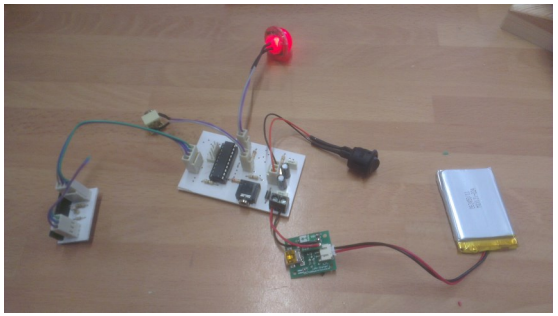
20mm Red LED

To make the LED easier to see I have opted to use one which has a 20mm diameter. Unfortunately they have a forward voltage of 6V. This exceeds the capabilities of the PIC and therefor the first time I loaded up the code to simply flash the LED nothing happened. After realising this I connected the LED to a 6V battery pack to verify it worked, it did. To get around this problem I decided to mount 2 smaller 5mm LEDs inside the 20mm dome. This would allow me to light up the big LED using the PICs 5V supply. To do this I drilled two 5mm holes and then checked the effect using a coin cell battery. Once I knew it worked I soldered the legs of the second LED to the first so they were in a parallel configuration. Finally I secured everything in place with a large amount of hot glue. Then I hooked the LED back up to the PIC and it worked fine, flashing on and off like it should have done.

```
main:
high B.7
pause 1000
low B.7
pause 1000
goto main
```

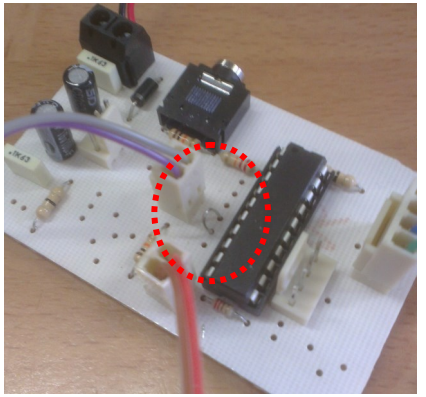







Timer

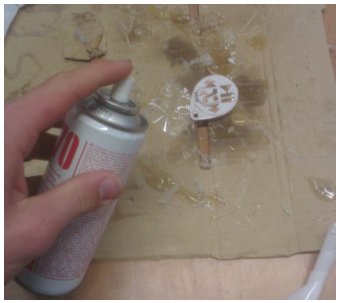
In order for the product to calculate the RPM whilst monitoring the capacitive touch buttons, the PIC needs a timer which it can receive an accurate time from. Unfortunately the Picaxe 20M2 doesn't have this as a built in feature so instead the solution is to have a hardware interrupt on a pin which is connected directly to a PWM pin oscillating at a pre determined frequency. In this case I have set it to 1000Hz to enable me to count the number of milliseconds between each crank revolution. From this number I can then calculate the current RPM of the riders cadence. Although this is not 100% accurate it will provide a good indication of how much time has elapsed and is the best possible way of doing such a thing using a Picaxe chip.



Testing and Modifications - PCB

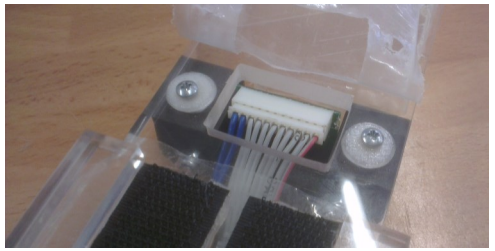
Capacitive Touch Buttons

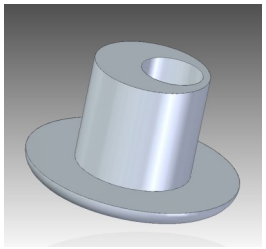
```
main:
touch16 B.6,w0
touch16 B.5,w1
touch16 B.4,w2
touch16 B.3,w3
debug
goto main
```



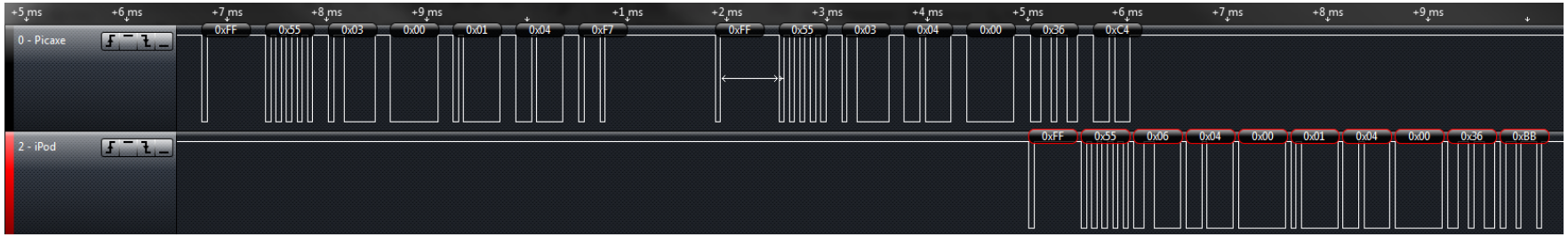
Several changes had to be made to the physical and software side of things for the4 capacitive touch buttons. To start with I was using just the touch command to detect a touch on the capacitive touch buttons. However this was not the raw data and the data outputted was not detailed enough to accurately detect a touch. Instead I am now using the touch16 command which provides an output in word format. This gives a much higher resolution making sensing touch events much more accurate. During my testing I also discovered I had not fully connected all of the elements of the play/pause button. To fix this I soldered a very fine wire to connect the two parts; I used as little solder as possible so that the pad would remain flat and not spikey at all as this would have been a serious quality control issue. Once this problem was fixed I used a plastic spray to apply a fine protective layer over the copper pads. This will insulate the user of the product from the circuit and also prevent the copper from oxidising.

iPod Connector

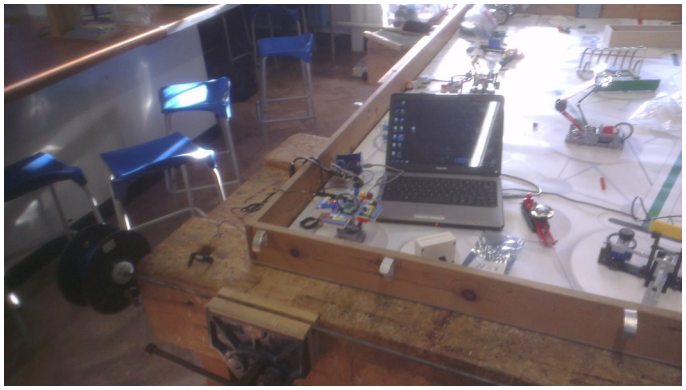




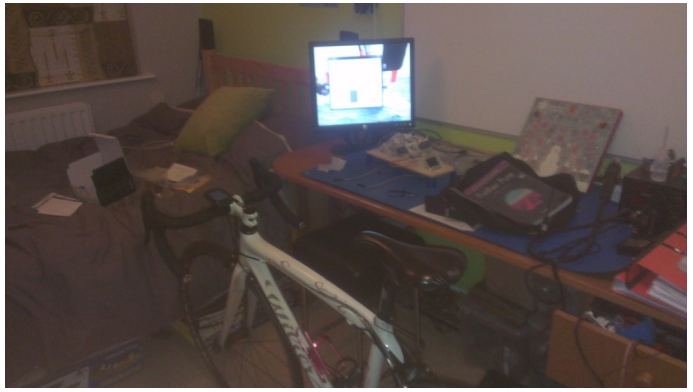
Unfortunately something went wrong when I designed the acrylic base on CAD, this lead to the holes to mount the iPod connector too were too big and not lined up. I solved this by designing and 3D printing the 'washer' shown too the right. The hole is offset to compensate for the holes not being lined up. Once I had done this I started testing the iPod serial connection. I did this by sending a few commands to the iPod; these are shown below. Firstly I sent the mode set command to set the iPod to AIR mode, then I sent the command requesting the current position in playlist. This shows the serial connection working because the iPod returns the current song number correctly. After this I tested each of the commands to skip backwards and forwards, play and pause, and toggle shuffle mode.



Testing and Modifications - Cadence Sensor



I have already tested the cadence sensor to see if it works, but now I have to check my code for detecting the RPM of the cyclist works. Initially I set up a test rig using some nylon wheels hot glued to the side of 3D printer filament spool. This then went on a steel rod and was able to spin freely. I had a rare earth magnet glued to the edge of the spool and then I held the cadence sensor in place of the bench. However this rig made for difficult testing as the spool would not spin for long enough at enough of a constant speed for me to successfully debug my code.

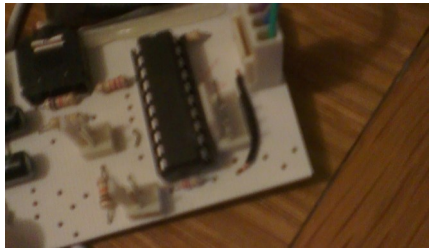


Instead I decided to use a real bike to make my life easier. I put the bike on a turbo trainer which lifts the back wheel off the ground and enables riders to train indoors—this is how my product will finally be used so it seemed like a good way of testing! I used a Garmin cadence sensor so that I knew the actual cadence and then I played with the code until I got my product producing a reading for the riders RPM very similar to that of the Garmin. I used a monitor on the bench in front of me with the Picaxe debug screen so that I could pedal at different rates and see how the cadence reading was effected. Once I had got the code working well my tests concluded that up to about 110 RPM the device was very accurate however above this the Picaxe chip was not quick enough to take accurate enough measurements to achieve a precise result.

Testing and Modifications - Modified Code

```
'================================================================================
'Count pulses from cadence sensor
'================================================================================
if shufflestate = 1 then
      readadc C.2,cadence'Read value from cadence sensor
      if cadence > 50 then'Is crank present...

            if revs=0 then'If not currently counting...
                  timerr=0'Reset timer
                  setint %00010000,%00010000'Start interrupt on pin linked to
            endif

            inc revs

            if revs >=2 then
                  setint off'Turn of interrupt

                  timerr=timerr/1'Take average time
                  revs =0
                  timerr = 60000/timerr

                  if timerr>160 then
                        timerr =160
                  endif

                  if timerr<30 then
                        timerr =30
                  endif

                  touchvalue=timerr'needs comenting out
                  oldsong=song
                  division=130/playlistLength'Scale playlist Length to RPM
                  song=touchValue-30
                  song =song/division

                  maths = division*song
                  maths = maths + 30
                  maths = timerr-maths

                  division = division/2

                  if maths>division then
                  maths=maths-division

                  else if maths<division then
                  maths = division-maths
                  else if maths = division then
                  maths = 0
                  endif

                  division = 100/division
                  duty = maths*division*8

                  pwmduty B.1, duty    '10000 Hz 10%

                  gosub selectSong

                  endif
                        do
            readadc C.2,cadence'Read value from cadence sensor
            loop while cadence > 50'Is crank present..


      debug
            endif


endif
goto main
```

Whilst the setup of my program as remained the same I have had to make quite a few changes to code which detects the riders cadence and selects the music. This is the new code. It uses the timer I discussed earlier to time one revolution of the pedals to the thousandth of a second. I experimented with averaging out a number of revolutions but it made little difference and increased the lag of the data. Once it has the time in milliseconds it divides 60000 by this to calculate the RPM of the rider. This is then scaled between 30 and 160 to make the next bit easier. Once the RPM has been detected the program allocates a song to this particular RPM. It does this by dividing 130 by the playlist length. Subtracting 30 from the current cadence and then dividing this value by the value that was calculated earlier. The brightness for the LED is calculated and then the function which produces the appropriate serial command to be sent to the item is called and the iPod changes track. To give the rider an idea of roughly where they are within the cadence 'zone' the LED changes its brightness. When the rider is right in the middle the LED switches off completely and as they get closer to either boundary the LED become brighter. This is done using PWM on pin B.1; unfortunately the motherboards LED was assigned to a pin which was not capable of PWM so I had to use a flying wire to connect the LED to pin B.1. The way the code works is the duty cycle for the PWM is calculated based on the riders position in the cadence 'zone'. At the end of every cadence reading the LEDs PWM is changed to reflect this enabling the rider to know weather to speed up or slow down in order to continue listening to the same song.



Here is the modification I made to the PCB. I also had to bend pin 11 of the IC so it wasn't in the chip holder. Without doing this the pin would just sink all the current being supplied from the other pin.

Testing and Modifications - Modified Code

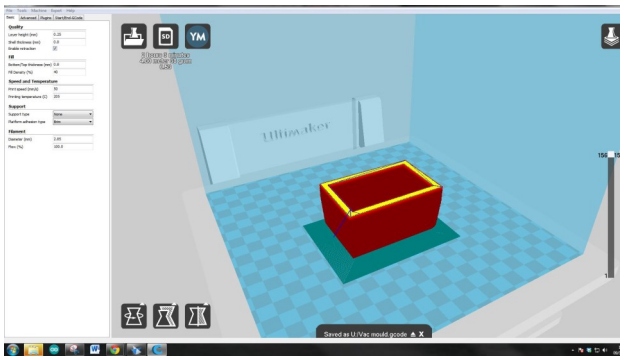
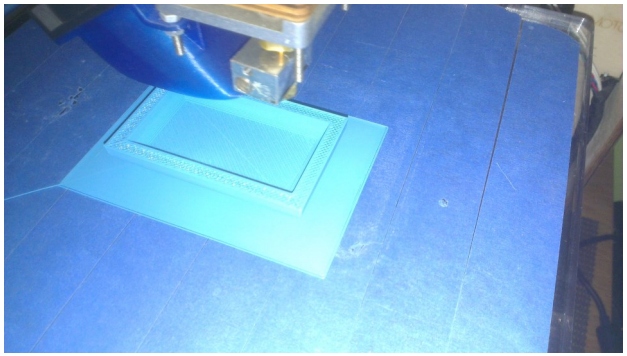
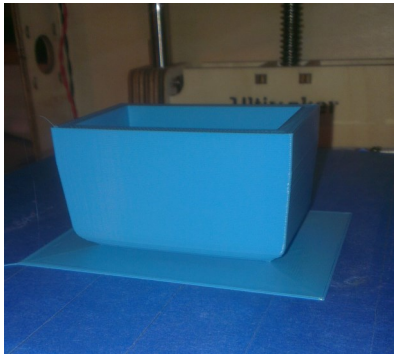
```
interrupt:'interrupt to count milliseconds

if togg = 0 then
      inc timerr 'increase timer
      setint %00000000,%00010000'set interrupt up again for low
      togg=1
else
      inc timerr 'increase timer
      setint %00010000,%00010000'set interrupt up again for high
      togg=0
endif

return

'================================================================================
'================================================================================
'Sub routines
'================================================================================
'================================================================================

selectSong:
checkSum = 190 - song'generate checksum
checksum = checksum & 255
serout B.0,T19200_32,(0xFF,0x55,0x07,0x04,0x00,0x37,0,0,0,song,checkSum)'jump to song
return

skipBack:
serout B.0,T19200_32,(0xFF,0x55,0x04,0x04,0x00,0x29,0x04,0xCB)'skip back
pause 1
return

skipForwards:
serout B.0,T19200_32,(0xFF,0x55,0x04,0x04,0x00,0x29,0x03,0xCC)'skip forwards
pause 1
return

playPause:

serout B.0,T19200_32,(0xFF,0x55,0x04,0x04,0x00,0x29,0x01,0xCE)'toggle play/pause01  CE
pause 100
return

shuffle:
if shuffleState=0 then
shuffleState=1
gosub flashLed'Flash LED
pause 1000
gosub flashLed
pause 1000
gosub flashLed
pwmout pwmdiv4, B.1, 199, duty    '10000 Hz 10%
else
shuffleState=0
PWMOUT B.1, OFF
gosub flashLed'Flash LED
pause 1000
gosub flashLed
pause 1000
gosub flashLed
endif
debug

return
```

I also had to make some changes the interrupt timing system because initially I wasn't getting very accurate results. The timer was significantly faster than it should have been because the interrupt occurred when the pin went high, it increased the "timer" variable and then reset the interrupt which fired again before the next PWM pulse. To fix this I changed it so the next interrupt to be set is opposite to the one which has just occurred. So if a high interrupt has just happened the next interrupt will be set up as a low interrupt. This is done by changing the mask when the interrupt is set up. This system worked well providing highly accurate timing over the space of 10 seconds.

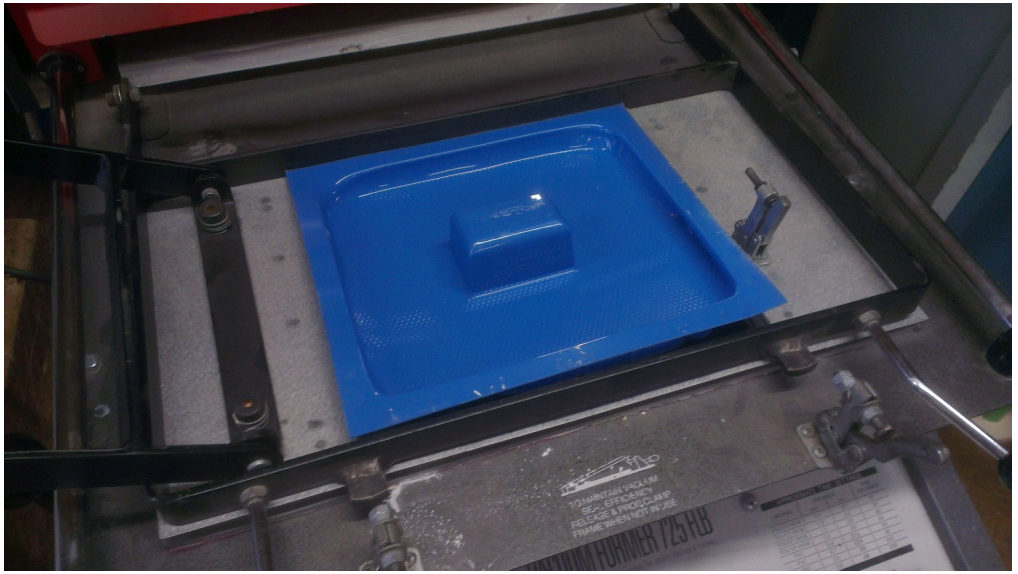
The iPod command sub procedures all work well however I realised that sending the mode change command along with every command lead to some funny issues with the play pause function. Consequently I have moved this command so it is only executed once at the beginning of the program once the iPod has been plugged in.

I have also removed the shuffle toggle function from the program as this I felt was unnecessary and also effected the playlist song selection function. Instead I am using the button to switch between modes; mode 1 is simple iPod controller and mode 2 disables the buttons and lets the Picaxe control the iPod based on the riders cadence. This works very well allowing the rider to just listen to music if that's what they want.

Vacuum Forming

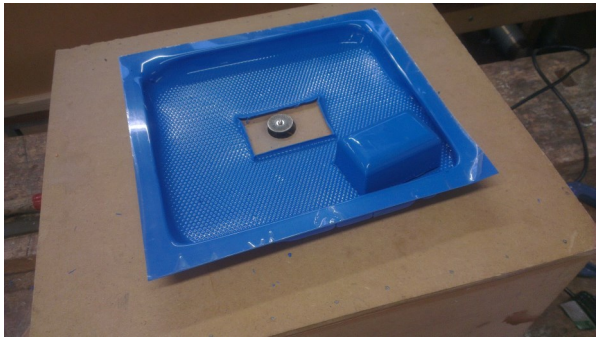




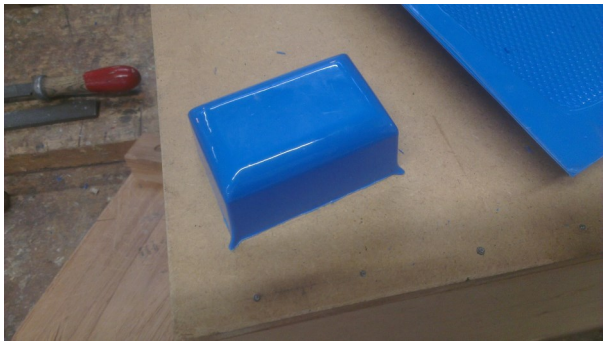

I decided to try 3D printing the mould to begin with because I already had the CAD design for it. I printed it quickly with a layer height of 0.2mm so there were some rough edges which required filing down. I also used a brim to reduce how much the mould warped during printing.

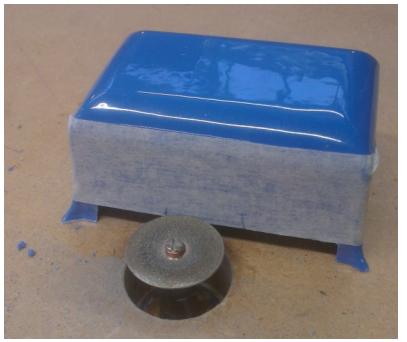



Once the mould was ready I vacuum formed my casing. First I covered the mould in talcum powder to make the mould easier to remove. Then I heated the sheet of HIPS and switched on the vacuum. Unfortunately the first time I heated up the plastic too much , this leadto webbing near the bottom of the mould. After trying again the results were better although there was still some webbing, this was probably because at the bottom my mould had flat sides rather than angled.

Casing Manufacture

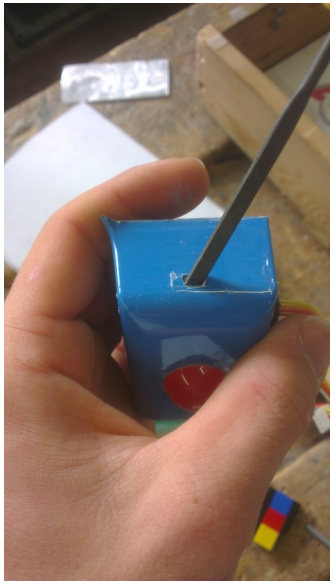


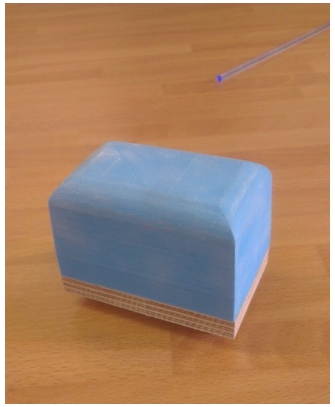


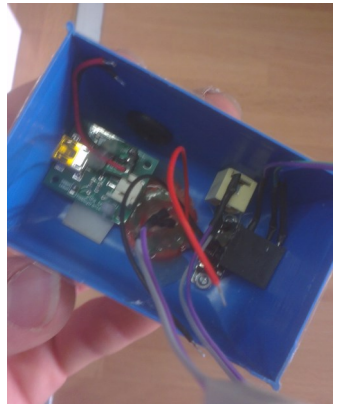


I used the gerbel cutter to remove the unwanted vacuum formed HIPs and then cut out the slot to allow the casing so slide over the acrylic. I marked were to cut with masking tape.

Drilling of holes for cadence and touch sensors





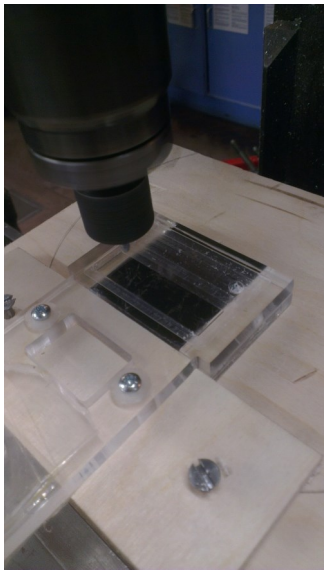


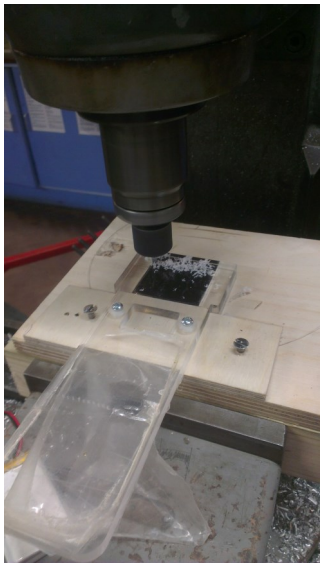
Test fit on acrylic with holes for LED and switch now cut using drill and scalpel

Hole for USB socket cut using a scalpel and then filed down to the correct size

The first casing was too small so I made the next one larger by gluing a piece of wood to the bottom of the mould. I also glued a smaller piece of wood underneath to try and reduce webbing in the corners which helped a lot.

Adaptation to Acrylic

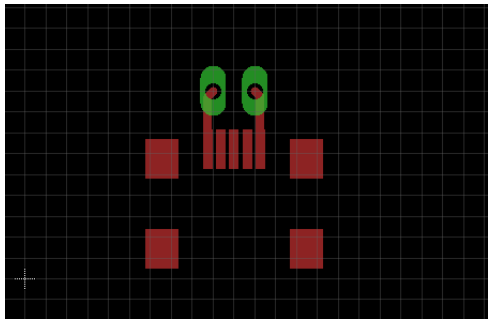




After having fully fitted out my vacuum formed case I soon realised that I was not going to have enough space inside my casing. The battery took up a lot of space and I did not want to re-build the casing. Consequently I decided to mill out some of the acrylic base to make some more room. I securely fixed the acrylic to some wood and then clamped it into the milling machine vice. I milled out a rectangular whole the correct height and width for the battery. I made it as deep as possible however I did not want to compromise the strength of the acrylic so I decided to stop milling after the rectangle was roughly 4.5mm deep; 3/4 of the total depth.

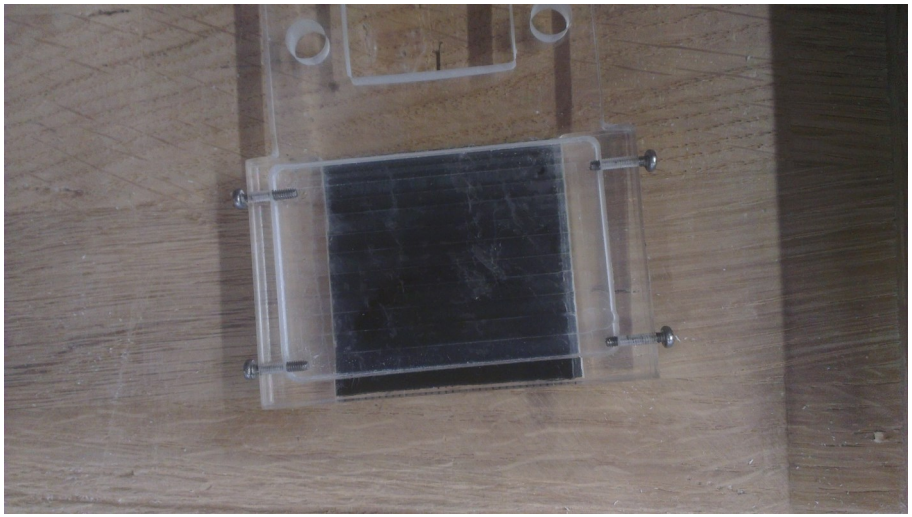
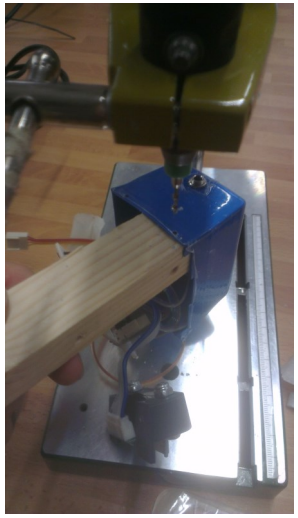
I put the battery in and then tried to fit the vacuum formed shell again — it was still a very tight fit. When I took the lid back off, it turned out the component legs had damaged the LiPo battery. As a result I used a small piece of HIPS to protect the battery in future. In the end though I concluded that I would not be able to fit the battery in and consequently I have opted to use an external battery to power the product.

To allow for the use of a battery in the future I decided to use a mini USB to power the product. This is the same connection as the charger uses so in the future if needed a rechargeable LiPo battery could be fitted inside the case. As I result I have had to make USB power board. This involves a mini USB socket mounted to a small piece of PCB with the +5v and ground pins broken out. This is the design in eagle CAD. The manufacture process is documented on the next page.



Casing Manufacture



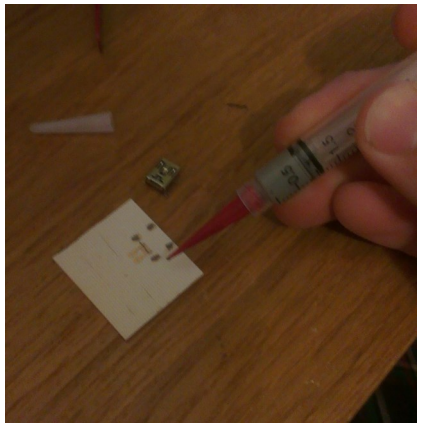

To secure the casing to the acrylic base I decided to use some M2 bolts. This was because I was still short of space so I needed as little overlap with the base acrylic as possible and M2 bolts were the smallest we had. First of all I marked and drilled a hole for each one, two on each side of the acrylic. I then used an M2 tap to put the correct thread in the acrylic so the bolt would be a secure fit. Once this was done I added a small washer to each bolt and put all four in the acrylic. The next thing was too drill corresponding holes in the vacuum formed casing. I used the mini drill with a 2.5mm bit. Unfortunately my holes were drilled slightly off centre because of how fiddly it was to drill. However this turned out to be a good thing as the casing clips securely onto the bolts now. I am not putting the outer shell on until the last minute as the tapped acrylic is very fragile because of the tiny tap.

Because I have opted to not use a battery in the case I can now use the space I milled out in the acrylic for mounting the PCB too. I drilled two holes in the PCB in a gap between the tracks. Because one was close to the edge I had to remove part of the self adhesive PCB stand off which now allows the PCB to be mounted central on the base.

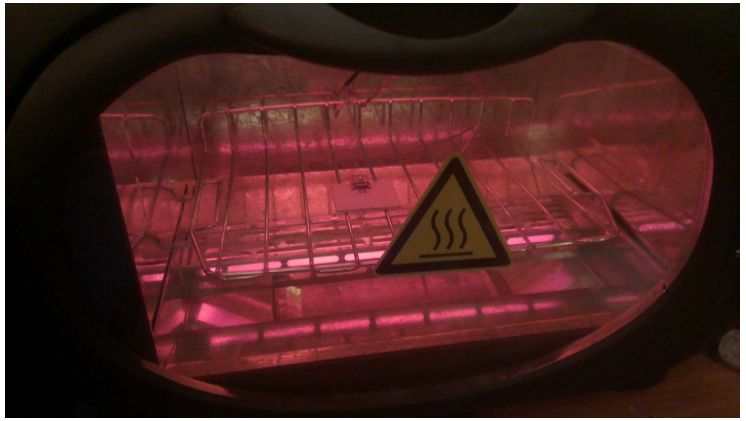
One thing which has varied slightly from my original design is the lack of bend in the acrylic. I opted not to bend the acrylic because the milled out section has significantly weakened the acrylic and I think bending the acrylic would put too much strain on this weak point.

Mini USB Power Sockets

PCB and SMD Mini USB socket ready to be reflowed

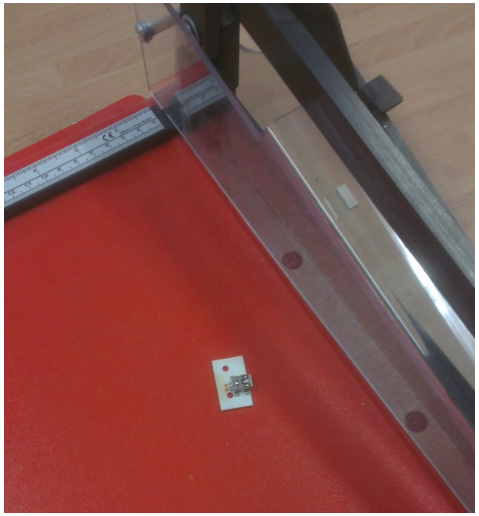


Appling solder paste to PCB and then positioning Mini USB Socket



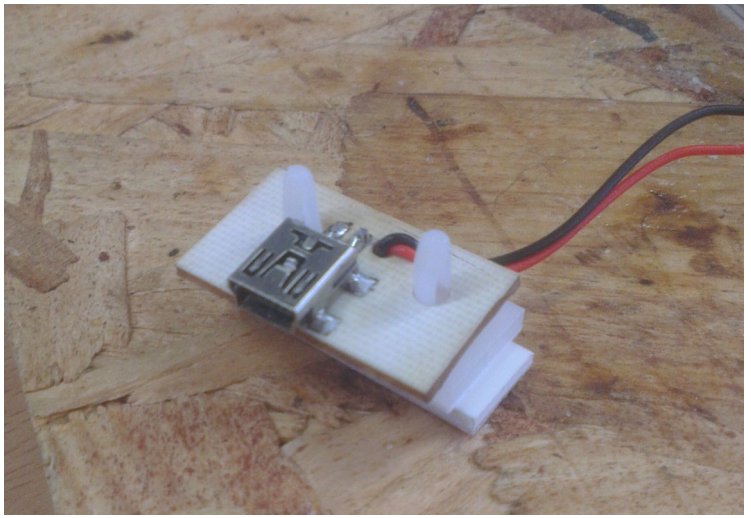
PCB being reflowed in a toaster oven

Socket mounted to PCB ready for use

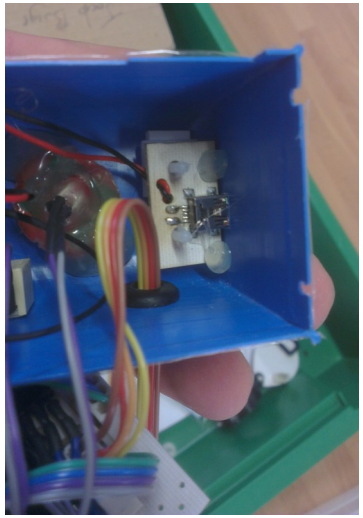


Cutting PCB down to size using PCB guillotine

Drilling holes for PCB self adhesive stand offs

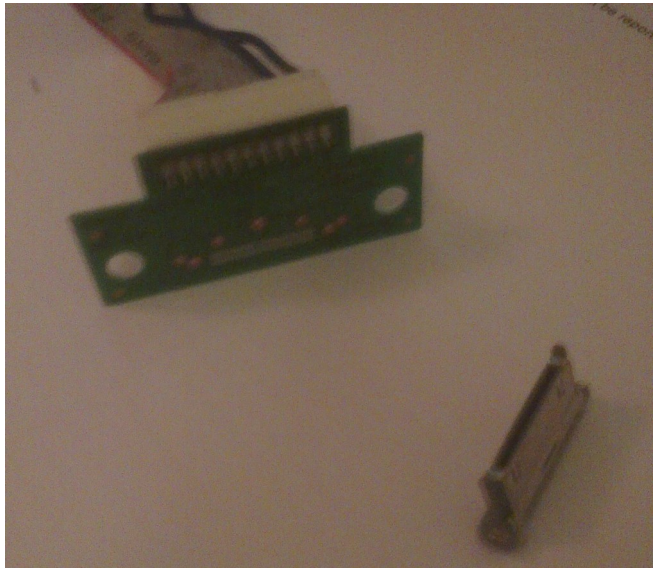


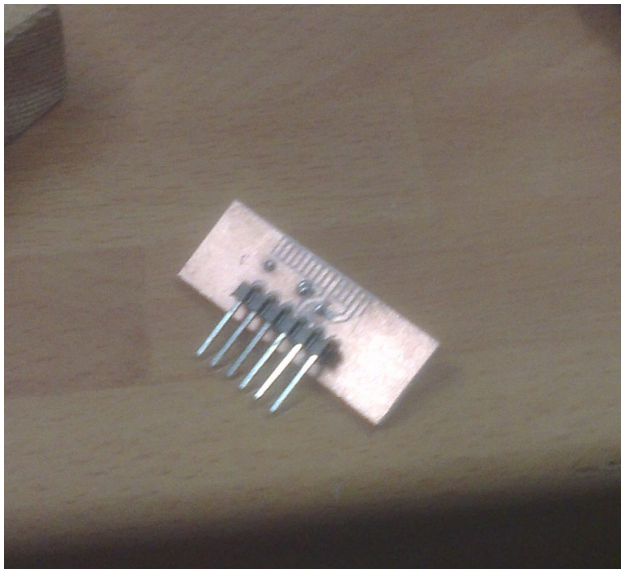
Power cables soldered onto board with strain relief holes to prevent cables breaking as well as PCB stand-offs mounted



USB mounted in casing

iPod Connector



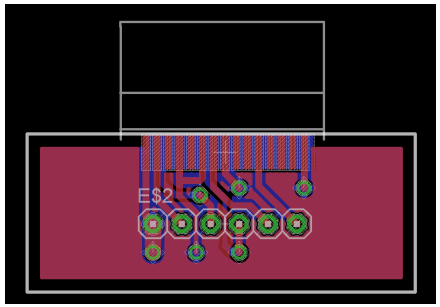


Unfortunately due to some poor quality manufacturing of the iPod connector breakout I was using the connector snapped off. This was because only the SMD pads had been soldered, not the heavier duty through hole fixings. After trying and failing to solder the connector back on I had a huge problem; no ability to interface with an iPod, the main feature of the project!

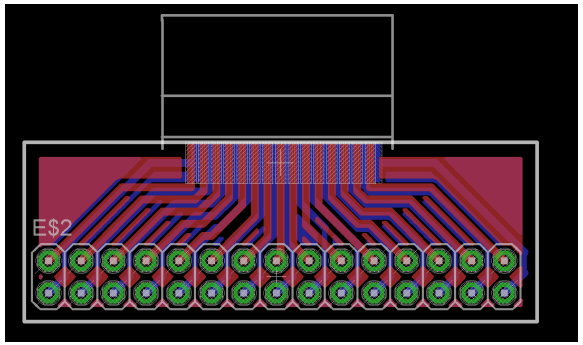
Due to time constraints I first tried manufacturing my own PCB. I inspected the original closely until I thought I had reproduced it correctly using eagle CAD. I then started making the PCB, however this time because of the way the iPod connector works I had to manufacture a double sided PCB. This is difficult because both sides must be lined up as closely as possible so the vias can be connected and the iPod connector itself soldered on correctly.

I have developed a technique for doing this using a jig made from clear acrylic. Firstly both acetates are lined up with each other, and one taped to one half of the jig. The second acetate, the one on top, has a piece of tape positioned upside down on it. Then the second half of the jig is lined up in each corner and dropped onto the acetates. The upside down tape secures the second acetate to its half of the jig and then the two pieces can be separated. Finally the PCB is taped to one half in position and then the second half is fitted into the first again, ready for exposure and etching.

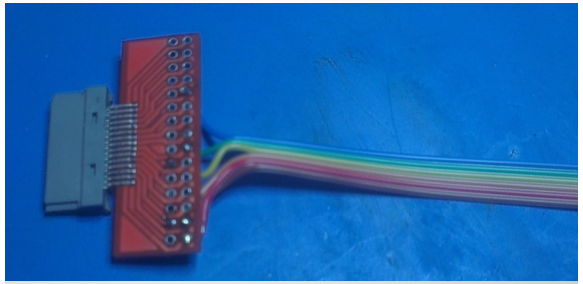
iPod Connector

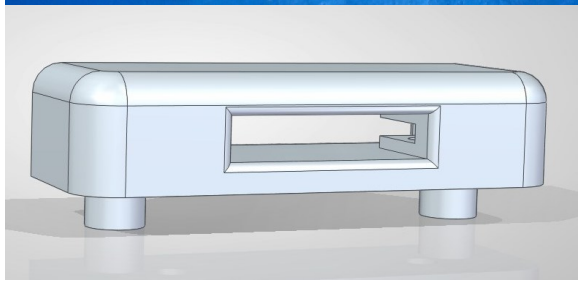


Unfortunately whilst my PCB was faultless I had no luck in sending commands to the iPod (PCB design top left). To increase my chances of getting something working I sent off for both the PCBs on the left. The top one in theory with all the connections already made and then the bottom one which is a backup breakout board in case the first board still doesn't work. After receiving the boards I soon established that there was a problem with the first boards design, as even the professionally manufactures one failed to work. So I soldered an iPod connector to my second, breakout board and hooked up only the necessary pins (demonstrated in this tutorial: http://www.instructables.com/id/Simple-Ipod-Controller/?ALLSTEPS) to an Arduino running the same code used in my iPod Serial Protocol research. It worked!

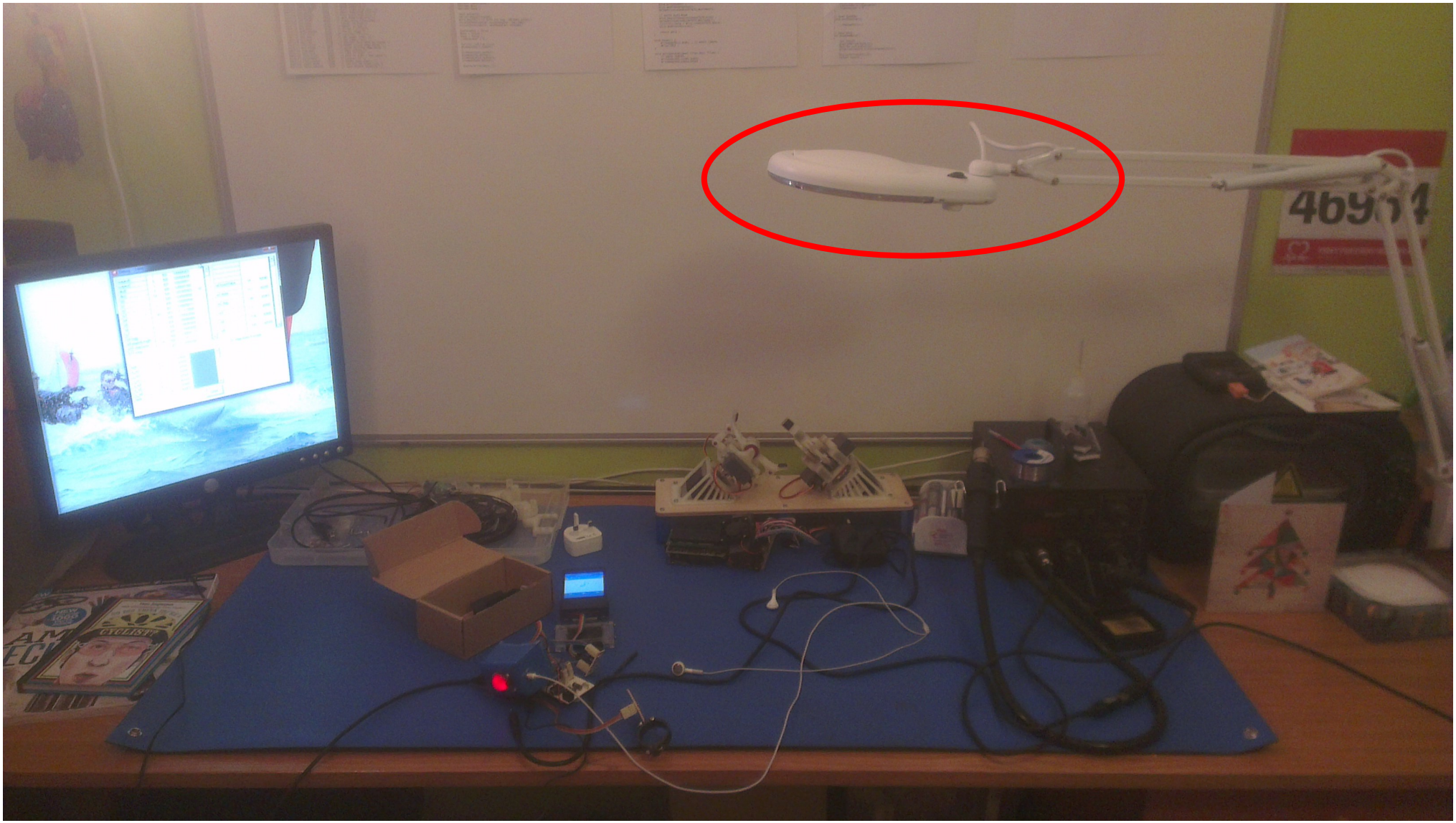


Because of the breakout style of the PCB I had to solder wires straight through the holes which was not ideal however time constraints meant it was not possible to get another revision of the first PCB made. It also meant I had to redesign the PCB holder for the 3rd time! Although I made a few nice changes; firstly grub screws coming up from the bottom of the print to clamp the PCB in place — this turned out to be very effective. Secondly I made the slot slightly tighter fitting on the connector, making the finished product look a bit neater. And finally I had to make the PCB slot slightly wider to accommodate the new design of PCB.





Capacitive Touch Problem



After a long break from coding whilst waiting for the new PCBs to arrive I got a new florescent lamp for my work bench. After half an hour of getting some very funny things happening with the capacitive touch, I finally worked out it was this causing all of the buttons to read the same value. I think this has something to do with the electrons being fired through the fluorescent tube as the problem stopped as soon as I switched the lamp off!

Parts List

PCB Parts List

Name	Cost	Quantity	Total
0R Resistor (1/4W)	0.003	3	£0.009
66.04 (W) x 39.37 (H) mm PCB	1.680	1	£1.680
17.78 (W) x 30.48 (H) mm PCB	0.680	1	£0.680
100µF Electrolytic Capacitor	0.050	2	£0.100
10k Resistor (1/4W)	0.003	2	£0.006
180R Resistor (1/4W)	0.003	1	£0.003
1N4001 Diode	0.010	2	£0.020
22k Resistor (1/4W)	0.003	1	£0.003
2-pin Terminal Block	0.060	1	£0.060
Download Socket	0.070	3	£0.210
PICAXE 20M2	1.200	1	£1.200
SPST Switch	0.320	1	£0.320
SIL Pins	2.440	1	£2.440
Dil Socket	0.02	3	£0.060
Crimp Sockets	0.82	1	£0.82
330R Resistor (1/4W)	0.003	2	£0.006
4.7µF Electrolytic Capacitor	0.050	1	£0.050
100nF Electrolytic Capacitor	0.050	2	£0.100
iPod breakout PCB	3.400	1	£3.400
Ribbon Cable	0.100	1	£0.100
Red 5mm LED	0.050	2	£0.100
Red 20mm LED	0.870	1	£0.870
USB socket	0.750	1	£0.750
iPod 30 pin connector	2.990	1	£2.990
Self tapping screws	0.020	2	£0.040
M2 Bolts	0.003	4	£0.012
Grub Screws	0.050	2	£0.100
velcro	0.500	1	£0.500
			£16.629

Casing Parts

Name	Cost	Quantity	Total
PLA	£14 per 1kg	1	£1.00
HIPS	£1.50 per sheet	1	£1.50
6mm Acrylic	£43 per m2	1	£1.00
			£3.50

Total Cost
£20.13

Testing

Many of these test points are shown in the final video of my product...

- *Able to sense a riders cadence —* ***Test: Displaying calculated cadence using debug and comparing to Garmin edge cadence read out?***

  When compared to the Garmin my device was able to provide an accurate reading up to 100 RPM; however past this point the Picaxe's clock speed was not fast enough to accurately measure the time per revolution and calculate the riders cadence accurately.

- *Make informed song choices based on the cadence information —* ***Test: Does the product select appropriate songs as speed changes?***

  Yes, as demonstrated in the video the product will change the song based on the riders cadence. As the speed increases it selects a song higher up in the playlist, which will be one with a faster beat and as it decreases it selects one lower in the playlist.

- *Select a specific song from a playlist on an Apple device —* ***Test: Can it jump between songs, for example track 3 to 5?***

  Yes, if the rider jumps a cadence zone it will jump a track. This is possible because in the code a new iPod serial command is generated for each song change allowing it to jump to a specific track.

- *Provide an audio out which can either be played through headphone, or plugged into an amp and speakers —* ***Test: Can it play music through headphones and speakers?***

  Yes, the product has an audio out which can be plugged into speakers or just headphones, it is directly connected to the iPod audio out just like the headphone socket.

- *Give the user the option of manually controlling the music —* ***Test: Can the music be controlled directly by the user?***

  Yes, as demonstrated in the video the device has two modes. One allows for manual control of the iPod using the capacitive touch buttons and the other automatically selects the music for them.

- *Provide a utility for helping the user to sort their music by speed —* ***Test: Is there something available to help sort users music?***

  As documented earlier in the project there is software such as Serato which will automatically detect the BPM of a track thus allowing the used to easily sort there music into an ordered playlist.

- *An LED will display rough position in a cadence 'zone' by changing in brightness—* ***Test: Can the position of the rider in a cadence zone be determined from the LED alone?***

  Yes, as shown in the video when the rider is comfortably in a cadence zone the LED switches off. As they get closer to the boundry the LED gets increasingly brighter.

- *The product must be able to interface with the iPod Serial protocol —* ***Test: Can the product control an iPod using the apple serial protocol?***

  Yes, the iPod can send and receive serial commands from the iPod and I have a spreadsheet for calculating new commands.

- *The product will be powered from a lithium polymer battery, rechargeable via USB— **Test: Can it?***

  Unfortunately I broke the LiPo battery that I was going to use in this project however if it was needed the product could easily have a battery in stalled in it. For now I am using either a mains USB supply or a rechargeable USB supply so there has been no compromise to the project.

- *The product must be in the price range of £20—£30—— **Test: Is it?***

  Yes, the parts list totals to £20.13, which is between £20 and £30.

Evaluation

The majority of the specification has been met for the product which is good. There are several things I would have done differently if I were to repeat the product:

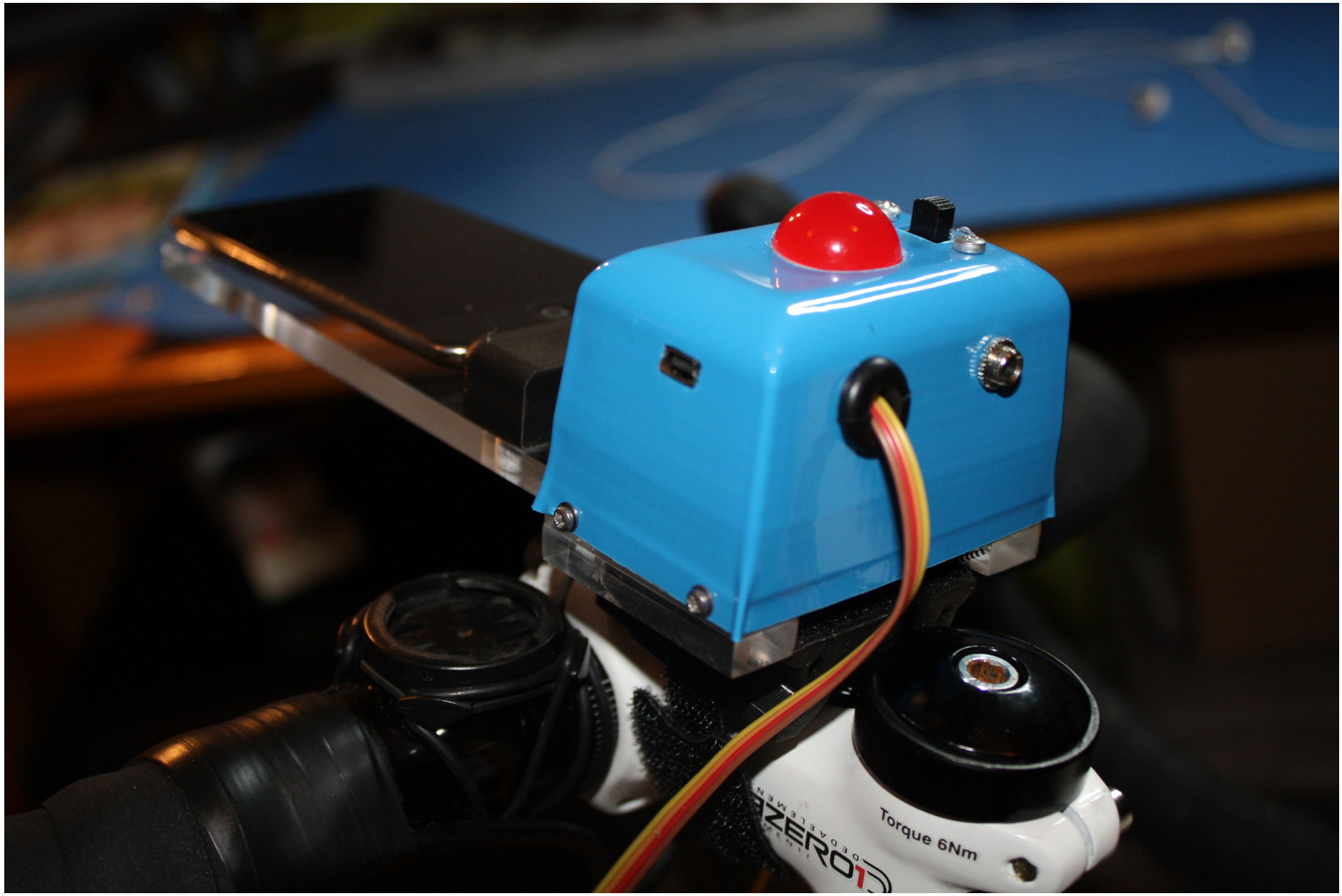
- I would make the casing much larger so it wouldn't be as much as a struggle to fit everything in as it was this time.
- I would also have checked the quality of the iPod connector PCB and reinforce it where necessary as this caused me the most problems in the whole project!
- I would try to design the vacuum forming mould slightly differently to reduce the webbing around the bottom of the edges. This could be done by increasing the draft angle of the sides and because I 3D printed the mould this could be done very easily in CAD.
- I might use an external IC dedicated to timing to record the time for each pedal revolution as this would increase the accuracy of the cadence reading.
- Get a second revision of my iPod connector PCB so I could connect to it using some crimped ribbon cable.
- A clearer visual indication of cadence might make it more useful as a standalone device for the user.

Things that I would keep the same:

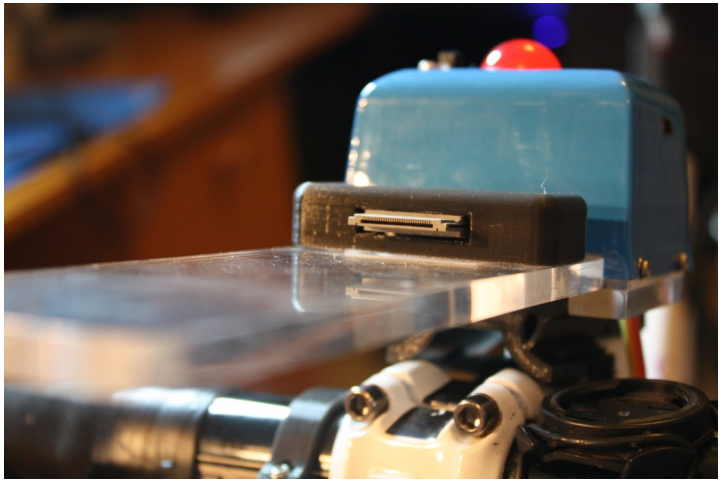
- The 6mm clear acrylic base with polished edges as this makes the product very aesthetically pleasing.
- The strong velcro used to make the product detachable from the bike for easy transition between riding on the road and riding turbo.
- Use of the iPod serial protocol to interface with an iPod for playing music as this makes the product very versatile.
- The use of 3D printing for rapid prototyping of components in the build. This also means that the CAD files are ready when the product goes into manufacture and needs to injection moulded.
- The use of capacitive touch to control the device

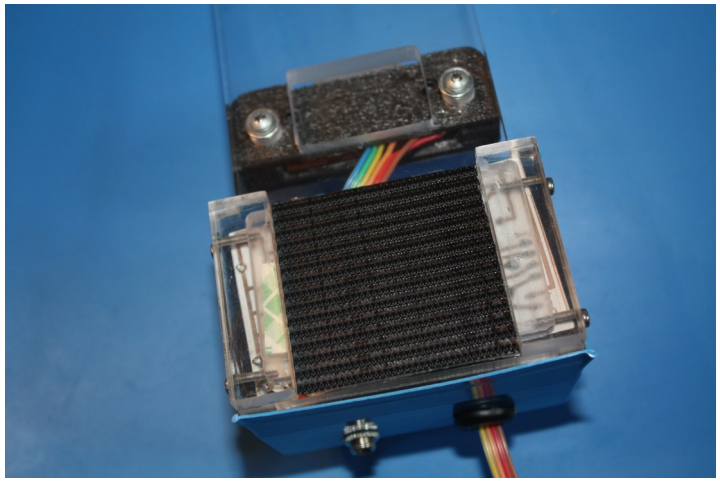
So in conclusion I am pleased with how the prototype product has turned out as I believe I have fulfilled my specification and my clients needs successfully.

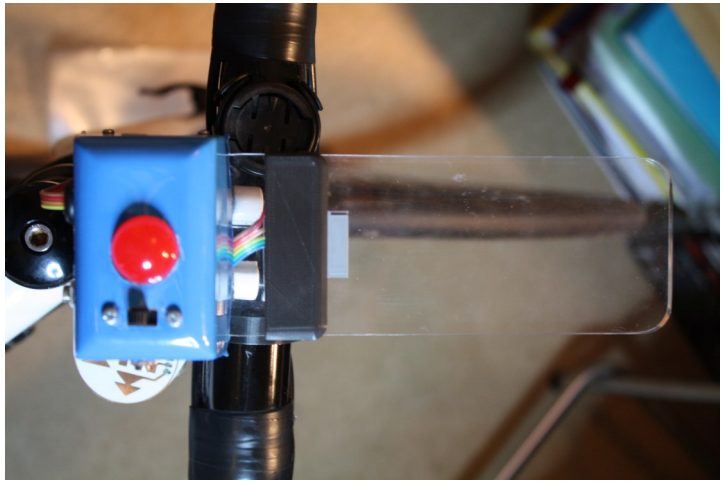
Finished Product



Finished Product







Video of product also included!