Strain Gauge Finger Position Tracking

Name: Daniel Brittle

Student ID: 21014253

Award Title: BEng Mechanical Engineering

Supervisor: Dr Alison Griffiths

1. Table of Contents

Table of Figures:

Abstract

Throughout the course of this project, a system to track the hand's digits' motion using strain gauges was developed and produced in a way in which a moderately skilled hobbyist (meaning capable of using a Soldering Iron) could replicate. This is achieved through the use of 3D printed components (allowing adaptability to the targeted user) and off-the-shelf components easily available through the internet. An instruction manual was also produced, providing the instructions to build the device alongside links to useful information for the user to read if they choose.

The overall aims of this project were:

- Build a device to track each digits' motion using strain gauges
- Make the device using off-the-shelf components & 3D printing
- Be able to replicate this device for less than £100
- Make the system portable and wireless
- Produce an instruction manual that allows the user to replicate the device with minimal difficulty

The applications of a device that can track the hands' motion are a wide field, from computer games (such as VR (Virtual Reality) systems and other simulators) to engineering and production, such as using the device to remotely control a system such as a robotic arm. These applications also stretch into the medical field, as remote piloting of a robotic hand system could allow a surgeon to operate on a patient from halfway across the world. The performing arts sector could also use the system developed as a cheaper alternative to the MiMU gloves (as they have a retail pre-order price of £2,699).

To develop this device, research was conducted into various methods of tracking each digits' motion, from a flat hand to a tight fist. This research highlighted that the two most popular current methods are flex sensors and machine vision, each of which have drawbacks which the use of strain gauges resolves. Research was also conducted into which microcontroller would be simplest to use, both in terms of ease of sourcing and programming, and as such an ESP32 DevkitC V4 was chosen due to the form factor and inbuilt wireless capability.

In summary, the device developed can be replicated for a price of £66.04 (excluding tools and expendable resources such as solder and electricity) and can be adapted by the user to suit their individual needs, such as hand size and strength.

1. Introduction

Throughout history, all of humanity's achievements have all been produced the same way; the sue of their hands. Whether it be through the forging of a wagon wheel, bringing the hammer down again and again on the iron, to modern vehicles being produced via CAD (Computer Aided Design) software on a computer using a mouse and keyboard. The human hand has always been the key method through which we interact with the world, and as such the motion of it is important to capture as more of the world becomes digital.

1.1. Aim

The aim of this project is to design, produce and test a portable device that tracks a hands' digits' movement, and wirelessly transmits this for use by a computer system. This tracking system will be produced using Strain gauges (to collect the positional data), 3D printing (to produce the non-electronic components of the system) and microcontrollers (such as an Arduino or ESP-32 development board) to collect and transmit the data collected. The system will be developed in a way that allows it to be easily adapted to different hand sizes, and to be able to be replicated by a moderately skilled hobbyist (meaning familiar with soldering).

1.2. Objectives

To achieve the aims listed in Section 1.1, the following objectives will be met:

- Research current digit position tracking methods and systems
- Research Strain gauges and how to interface with microcontrollers
- Develop a system to track the hands' digits through the average range of motion
- Produce this system in a way in which it can be replicated without the use of specialised equipment (Excluding a Soldering Iron and 3D Printer)
- The design must be portable, and as such able to function wirelessly
- The design must be adaptable in a way that allows different people to scale the design to better fit their needs, such as hand and digit size
- The device must be able to be replicated for less than £100
- An Instruction Manual will be written to accompany the device and aid in construction

1.3. Background Information

There are various methods already in existence to do just this; however, each of these have their strengths and weaknesses. An example would be a VR gaming system called the Meta Quest 2, which uses machine vision to capture the hands' motion and reproduce that within the game being played, with the limitation that the hands must be visible to the cameras used to be captured. This limits this method to only be suitable for well-lit, open areas such as a mostly empty room.

The current medical method of tracking the digits' motion is through the use of flexible sensors: while they are fairly accurate, they also are only available in fixed lengths and are costly for a single sensor (£7.60 per sensor (PiHut, 2018)).

1.4. Justification

There has been no dedicated research into a new method to track the hands' digits' motion via sensors since the commercial failure of the Nintendo Power Glove, and as such modern technology hasn't been utilised. The closest modern product is the MiMU gloves, but these use flexible sensors (the same method as the Power Glove) and have a selling price of £2699.99.

Strain gauges are the chosen are of research due to their widespread use in materials testing, meaning they are widely available and can have a high degree of accuracy due to their intended use requiring it. The use of strain gauges to successfully produce a digit tracking system with a reasonable degree of accuracy would allow hand tracking systems based on sensors to be produced and marketed for a much lower cost, increasing the availability of digitally tracking the hands' motion to a wider audience, such as VR users.

2. Project Management

2.1. Project Progression

Throughout the development of the tracking system, weekly meetings with the project's supervisor will occur; These meetings will be documented in a meeting log to show progression and provide a shared location to keep notes and queries about the project.

A project breakdown will be composed, and this will then be used to create a Gantt chart to visibly show the project's progression week on week. The Gantt chart can be found in Appendix A.

2.2. Health and Safety

As with any practical project, Health & Safety is an important factor in how the project can develop. To ensure that adequate H&S was followed, a risk assessment was produced and can be found in Appendix B

3. Design

3.1. Iterations

3.1.1. Concept

To begin designing the digit motion tracking device, it first must be planned how many gauges there will be, so that the entire hands' motion can be tracked. Figure 3-1 below shows the conceptual locations of the gauges on the back of the hand

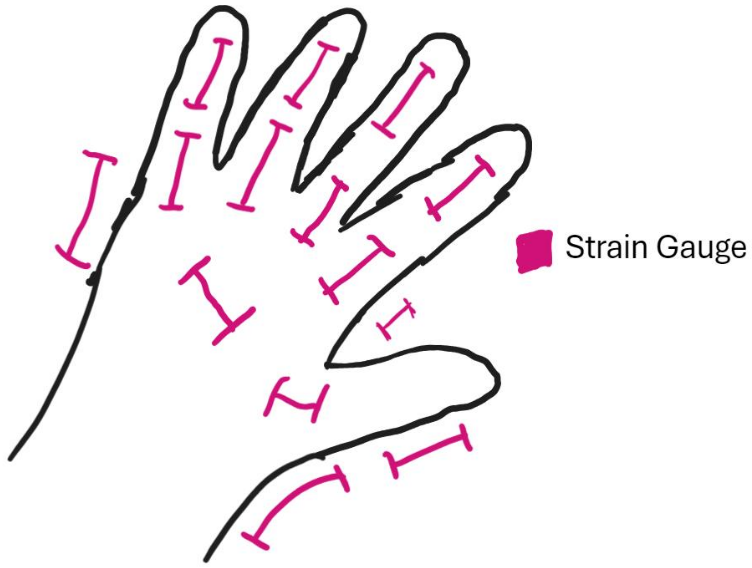


Figure 3-1: Conceptual Strain Gauge Locations

There is planned to be 14 strain gauges to track the hands' motion; 10 which cover the movement of each of the joints on the finger, two which track the spread of the fingers. One on the back of the hand to track the gripping motion and one to track the thumb moving from side to side.

Once the number of gauges has been decided, the next step is to create a flow chart of how the system will function, shown in Figure 3-2.

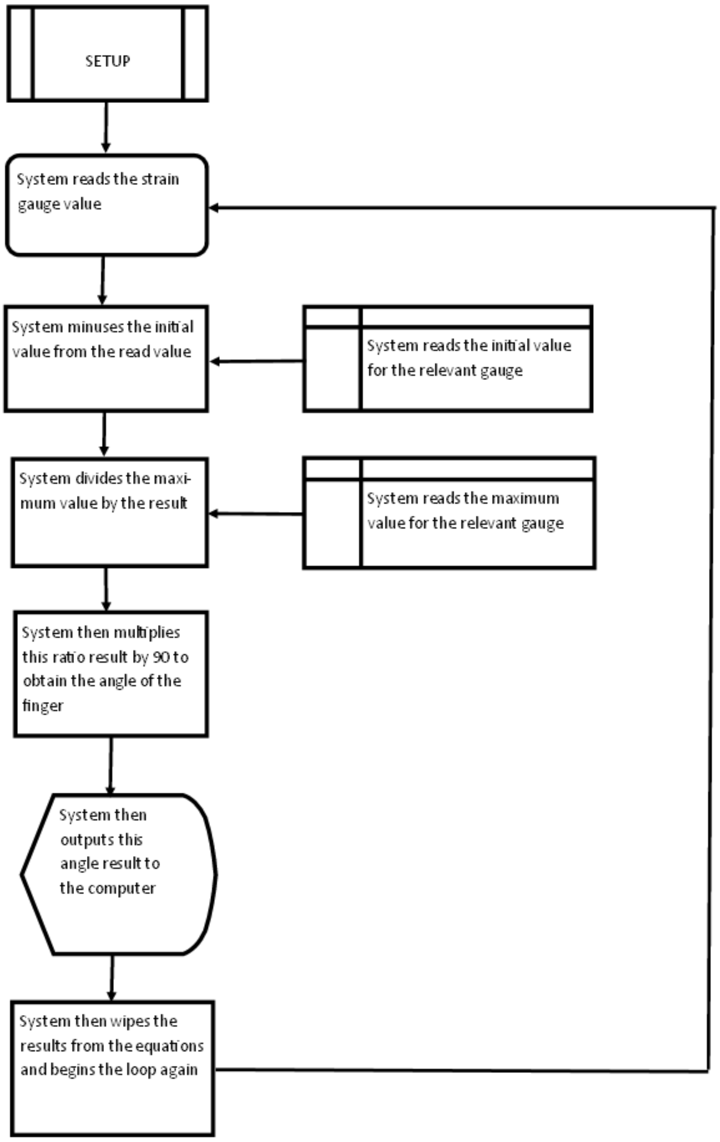


Figure 3-2: Flow chart of how the system will function

The creation of the flow chart helps to ensure that the system is functioning as intended; the flow chart highlights the key stages and decisions the system will face and decide as it is in use.

As the number of Strain gauges has been decided and the system has been roughly planned, a prototype can be designed and produced, as shown in section 3.1.2

3.1.2. Mk.1 – Arduino Based

The first prototype of the system is based on an Arduino Mega 2560, due to the ease of accessibility and the number of GPIO pins. The number of GPIO pins were necessary for this version of the tracking system as each HX711 board was initially individually wired to the Arduino, so a total of 56 input pins were required. However, further research concluded that the Clock, VCC and Ground pin could each be shared between multiple boards, leading to the circuit diagram shown in Figure 3-3.

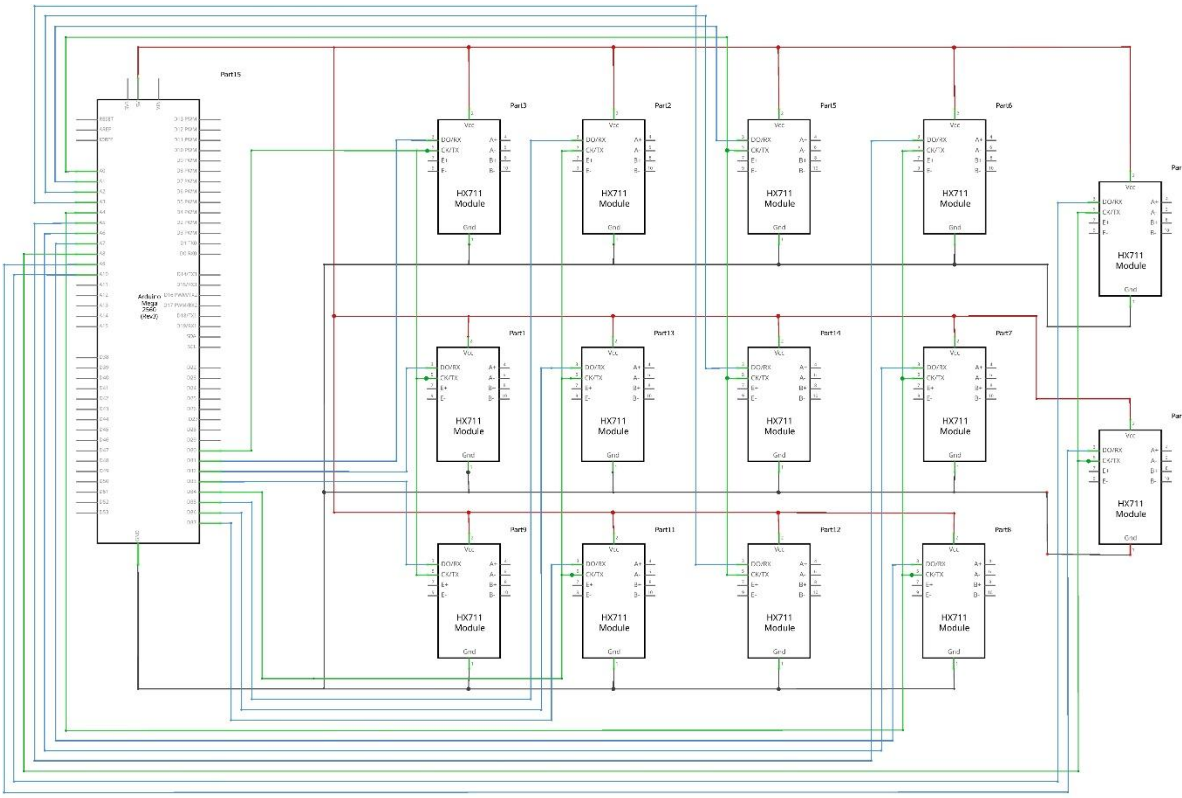


Figure 3-3: Circuit diagram for the Mk1 tracking system

To allow the shared wires to connect to multiple boards, they were soldered in parallel to one another, as shown in Figure 3-4.

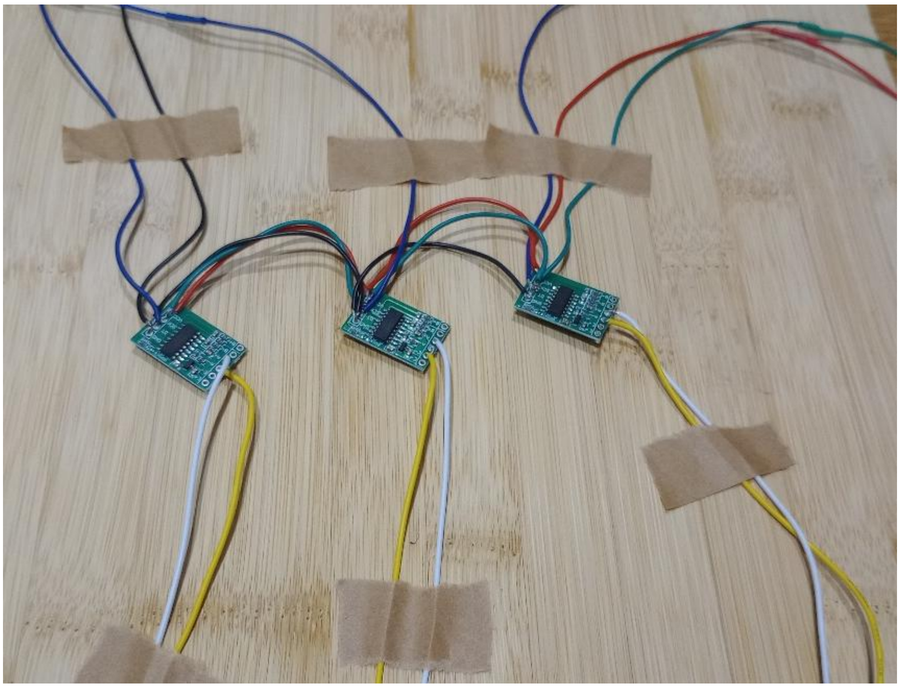


Figure 3-4: HX711 boards soldered in parallel

To ensure that a single mis-soldered joint didn't cause the entire system to fail, the gauges were divided into 4 groups of three gauges and a group of two. Each of these groups had separate VCC (Red), ground (Black) and Clock (Green) wires, and each individual board had a dedicated data (Blue) wire. The yellow and white wires shown in Figure 3-4 are those that would connect to the strain gauge on the back of the hand.

An LCD screen was also planned implemented alongside the ESP-32, as this would allow the suer to be aware of how the system is functioning when they aren't near the computer the system is communicating with.

A 3D printed case was designed to house this system on the user's wrist, shown in Figure 3-5.

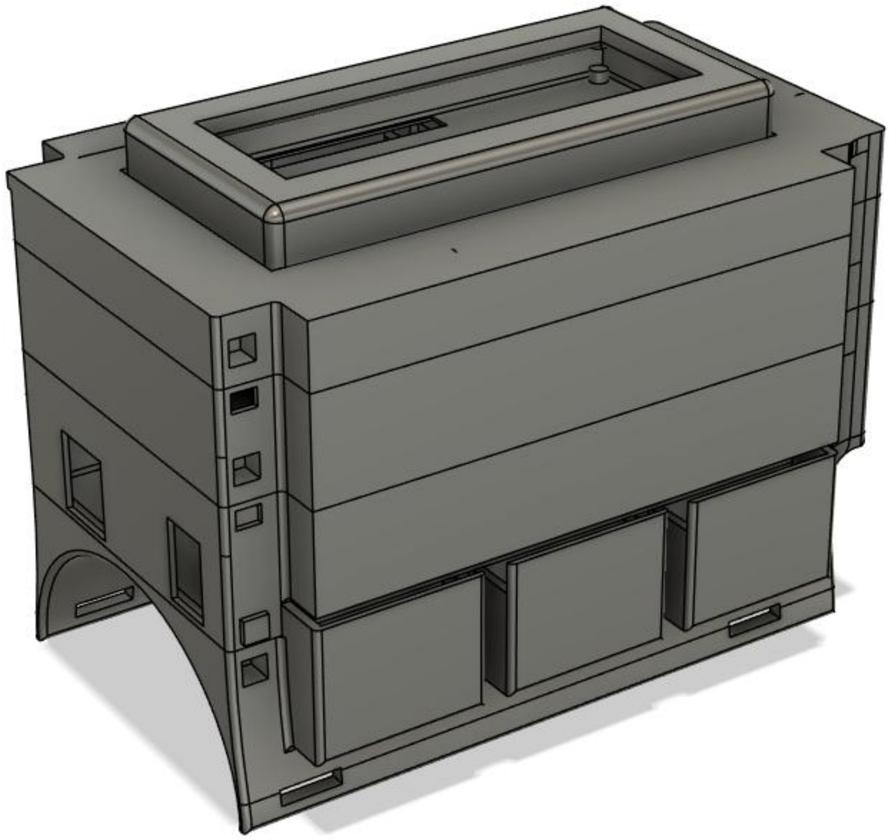


Figure 3-5: Mk.1 Casing - Arduino based

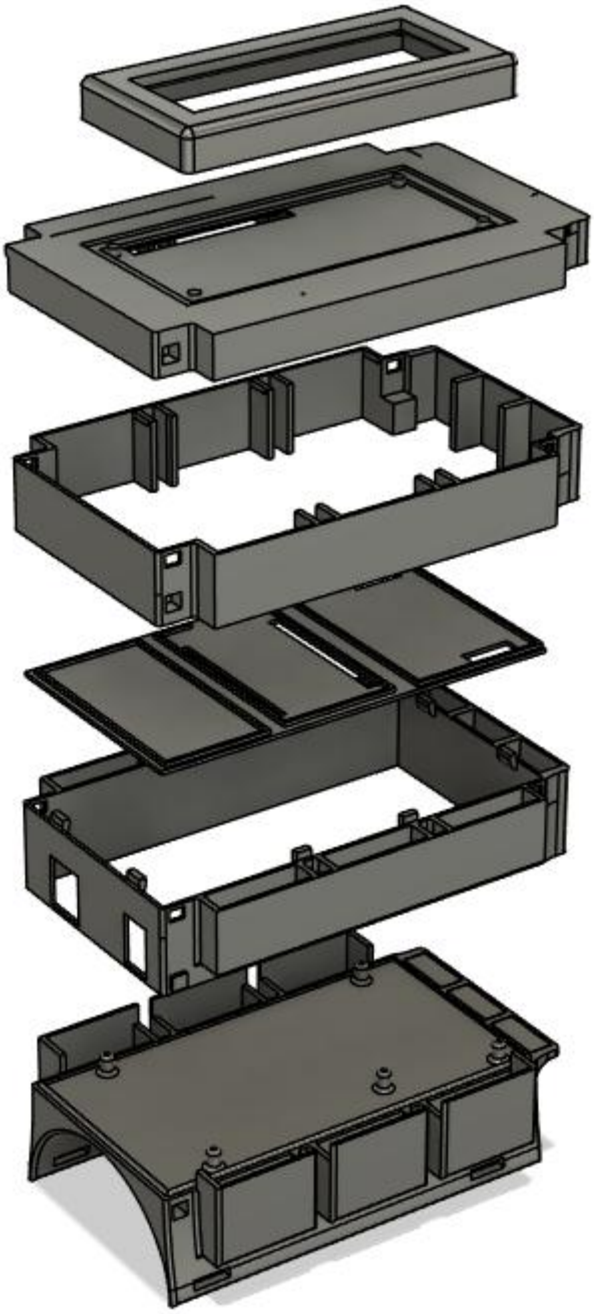


Figure 3-6:Mk.1 Casing - Arduino based (Exploded View)

This design was dismantled in favour of the Mk.2, as the Arduino doesn't have wireless capabilities. Research was conducted into implementing them in some way, such as including a wireless expansion board, but it was concluded that the best option would be an ESP-32 board to include this. However, to interface between the two boards a level shifter would have to be used as the Arduino operates on a 5V system (Haseloff, 2024) whereas the ESP-32 operates at 3.3V, and would be damaged by 5V (Barrozo, 2023).

3.1.3. Mk.2 – ESP-32 Based (Fully wired)

For the next version of the tracking system, the Mk.2, the decision was made to fully switch the Arduino for the ESP-32 DevkitC V4, as it had enough GPIO pins to be able to support the 14 gauges and an 12C OLED screen. This switch allowed the 3D printed casing to be reduced and include a dedicated transformer board, allowing the device to become battery powered by a 9V battery.

The inclusion of a dedicated power board and battery meant that the design is more adaptable by the user if they choose, as the ESP-32 can also be powered by a Micro-USB cable. Also, the power board means that the HX711 Amplifier boards can be powered at a higher voltage (5V) than the ESP-32, as the ESP-32's maximum operating voltage is 3.3V whereas the HX711 boards have an operating range from 2.7V to 5.5V (Avia Semiconductor, 2009).

A circuit diagram of the 14 gauges, OLED screen and ESP-32 is shown in Figure 3-7.

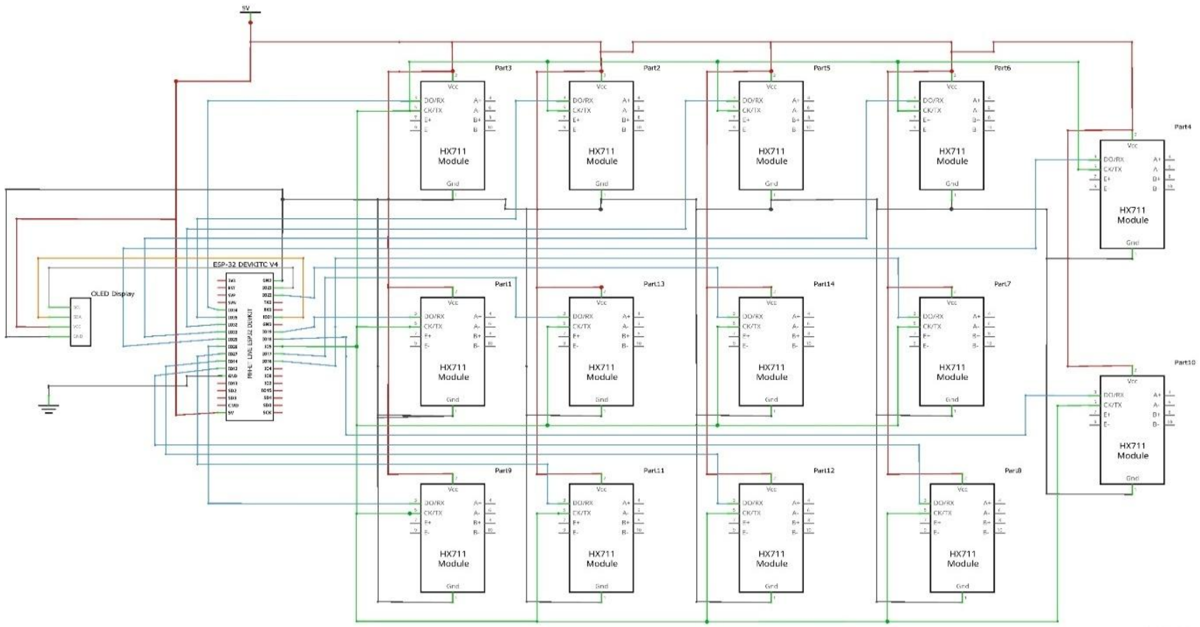


Figure 3-7: Wiring Diagram for the full 14 gauges (ESP-32)

While the circuit diagram depicts the system to be organised, the practicality of routing each of the 24 individual wires that needed to be attached to the ESP-32 is different, as shown in Figure 3-8.

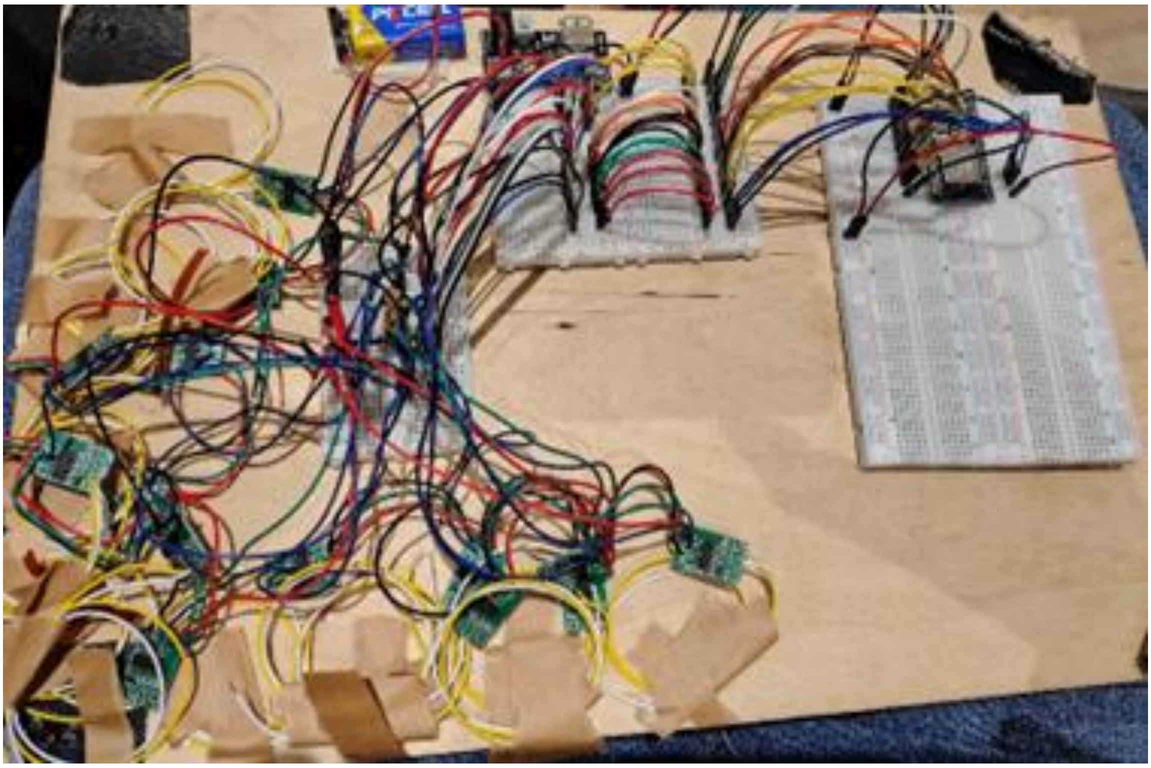


Figure 3-8: Photo showing each of the wires needed for the MK.2 tracking device connected to the ESP-32

The difficulty caused by the management of this number of wires for the user is why the decision was made to create a custom PCB (Printed Circuit Board).

This helps the project to better achieve the aims set out previously, as the use of a custom PCB, while increasing the initial price of the project, greatly simplifies the assembly process for the user. It does this by greatly reducing the amount of wire cutting, stripping and soldering required to connect each of the components together, as the PCB would allow them to be simply slotted into place by using header pins. This also means that the system uses a modular approach to bought components, allowing the user to simply switch a faulty component for a new one within 5 minutes, as opposed to the previous iterations where the faulty board would have to be de-soldered and then the new board soldered in its stead.

3.1.4. Mk.3 – ESP-32 (PCB)

This iteration of the project aimed to reduce the chaotic aspects of the entirely individually wired HX711 boards' approaches of the previous iterations by creating a custom PCB.

However, as this is a prototype in development, Perfboards (MorePCBAd, 2023) were used to allow for experimentation, and a cheaper development cost in case of issues during production.

Figure 3-9 shows the legend of the PCB schematics shown in Figure 3-10 and Figure 3-11, as they are intended for the user to follow to recreate the PCB board for their own system.

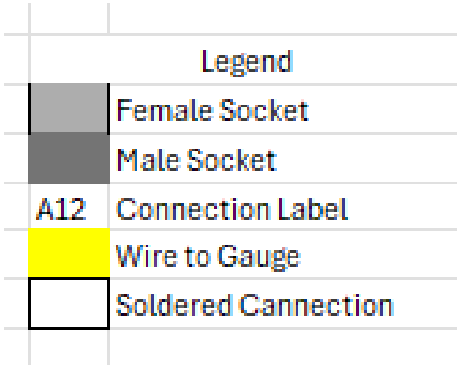


Figure 3-9: Legend for Figure 3-10 & Figure 3-11

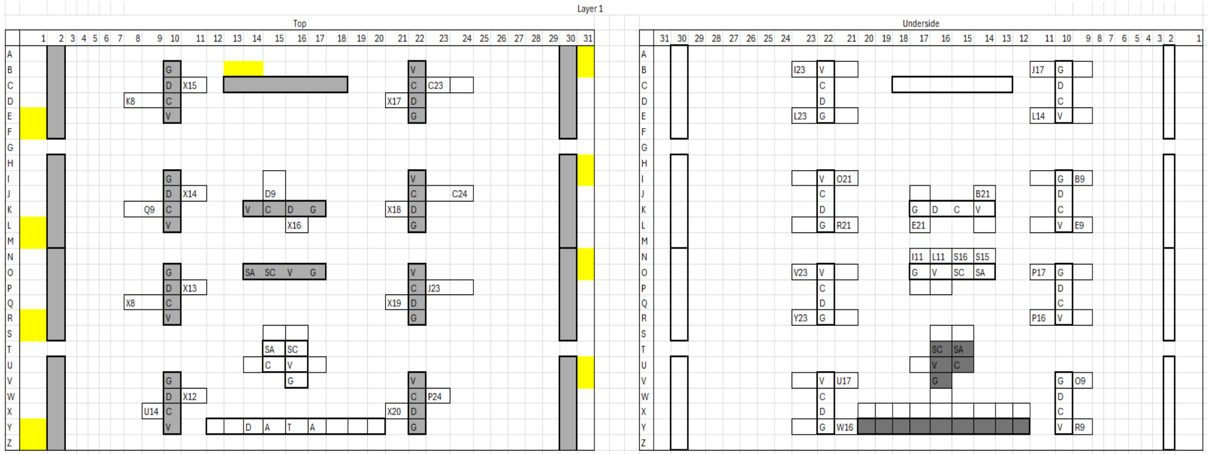


Figure 3-10: PCB Schematic for the upper Mk.3 PCB

In Figure 3-10, the layout of the upper Perfboard PCB is shown, following the legend shown in Figure 3-9. This layer contains only HX711 boards, meaning a majority of the wires follow one another around the board.

For ease of use, the VCC, Ground and Clock wires are each labelled in a way that one board leads to the next, tracing back to the male header in the centre of the board. Unshaded boxes should be connected to the nearest shaded box via solder, which will be shown in Section 3.2.

Each separate type of pin was soldered using a unique colour of wire (VCC is Red, Ground is Black, Clock is Green and Data is Blue) and divided equally between the top and bottom of the Perfboard, with the Clock and Data wires being soldered on the top of the board and the VCC and Ground on the underside.

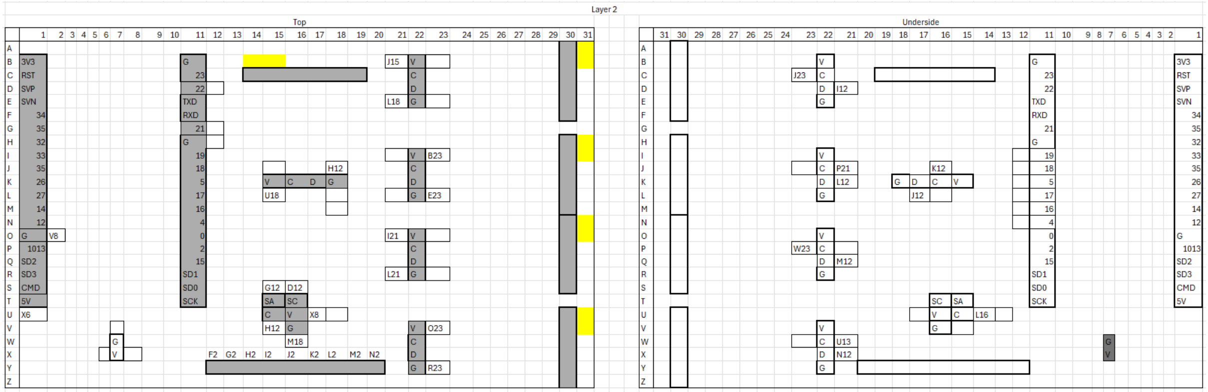


Figure 3-11: PCB Schematic for the lower Mk.3 PCB

Figure 3-11 shows the PCB layout for the lower layer of the Mk.3 PCB boards and follows the same legend and rules as Figure 3-10. This layer contains the ESP-32, and as such each of the pins is labelled to assist the user in producing the prototype PCB.

This layer draws power from the dedicated power board and 9V battery placed underneath the PCB in the housing and connects to the power board using the male header pins shown on the underside schematic in positions W7 and X7. As can be seen from the schematics, each of the components is grounded to the ESP-32, which in turn is grounded to the power board. Grounding each of the components through the ESP-32 is necessary to allow the components to function, such as the OLED screen.

The two layers of PCB connect through the corresponding male and female header pins on each board, as they are designed to slot on top of one another. However, this is not feasible due to the size of the other components present, and as such the corresponding headers can each be connected by a DuPont jumper cable (multicomp PRO, 2020), for the purposes of prototyping and DIY construction by the user.

However, the design concept this design is based upon was proven to be ineffective (explained in Section 4.1.1).

The multiple layers of PCBs, alongside the spacing required between each layer, meant that the casing designed for the Mk.3 was bulky and unwieldy when on a user's wrist. The casing shown in Figure 3-12 is that which was designed for the Mk.3 and visibly shows the size of the casing, being 95mm tall from the wrist (122mm total).

Figure 3-12: Mk.3 Casing

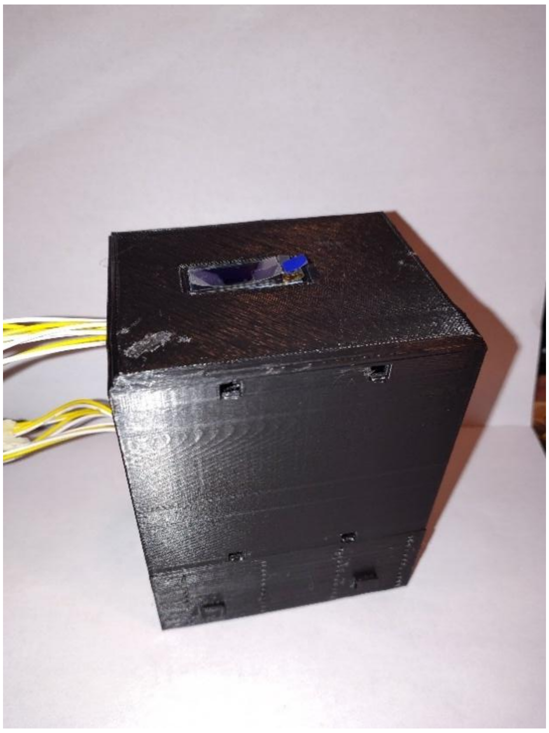


Figure 3-13: Mk.3 Casing (Assembled)

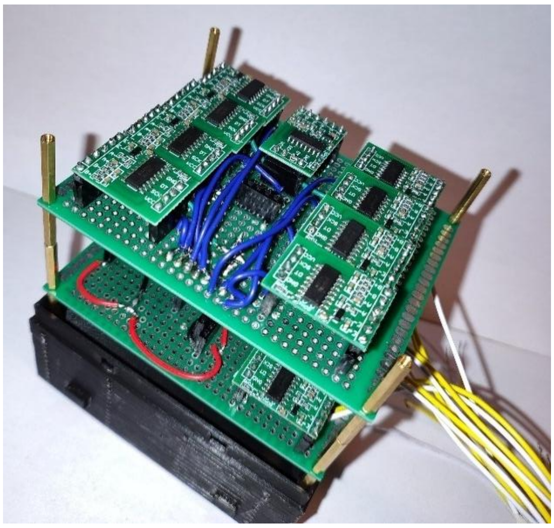


Figure 3-14: Mk.3 Electronics (Assembled)

Figure 3-14 shows what is inside the casing shown in Figure 3-13, and showcases the height that the two PCB layers stack to when space is kept empty in preparation for the Dupont cables requires to connect the two.

Further images of the upper layer's prototype PCB for the Mk.3 iteration are shown in Figure 3-15 and Figure 3-16.

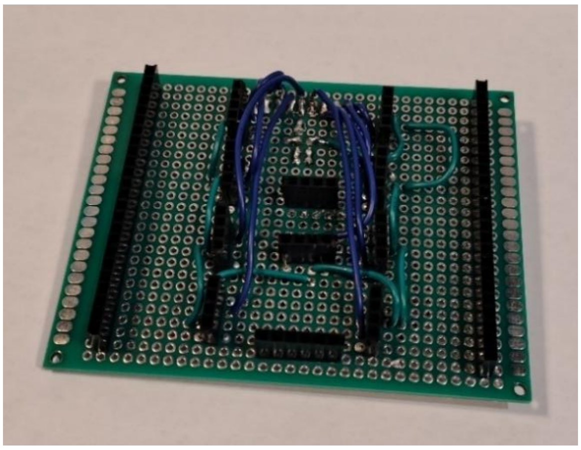


Figure 3-15: Mk.3 Upper prototype PCB wiring (Top)

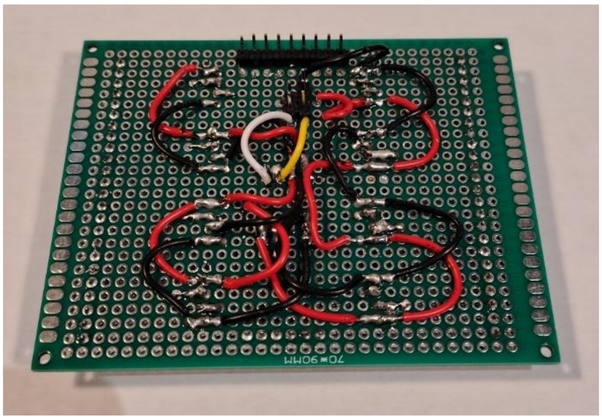


Figure 3-16:Mk.3 Upper prototype PCB wiring (Underside)

3.1.5. Mk.4 – ESP-32 (5 gauges)

As the basis of the Mk.3 was proven to be ineffective at consistently tracking motion, further research was conducted until a working method was found (Section 4.1.2).

As this new method is much bulkier, the circuit diagram (shown in Figure 3-17) was reduced to read only 5 Strain gauges, one for each digit.

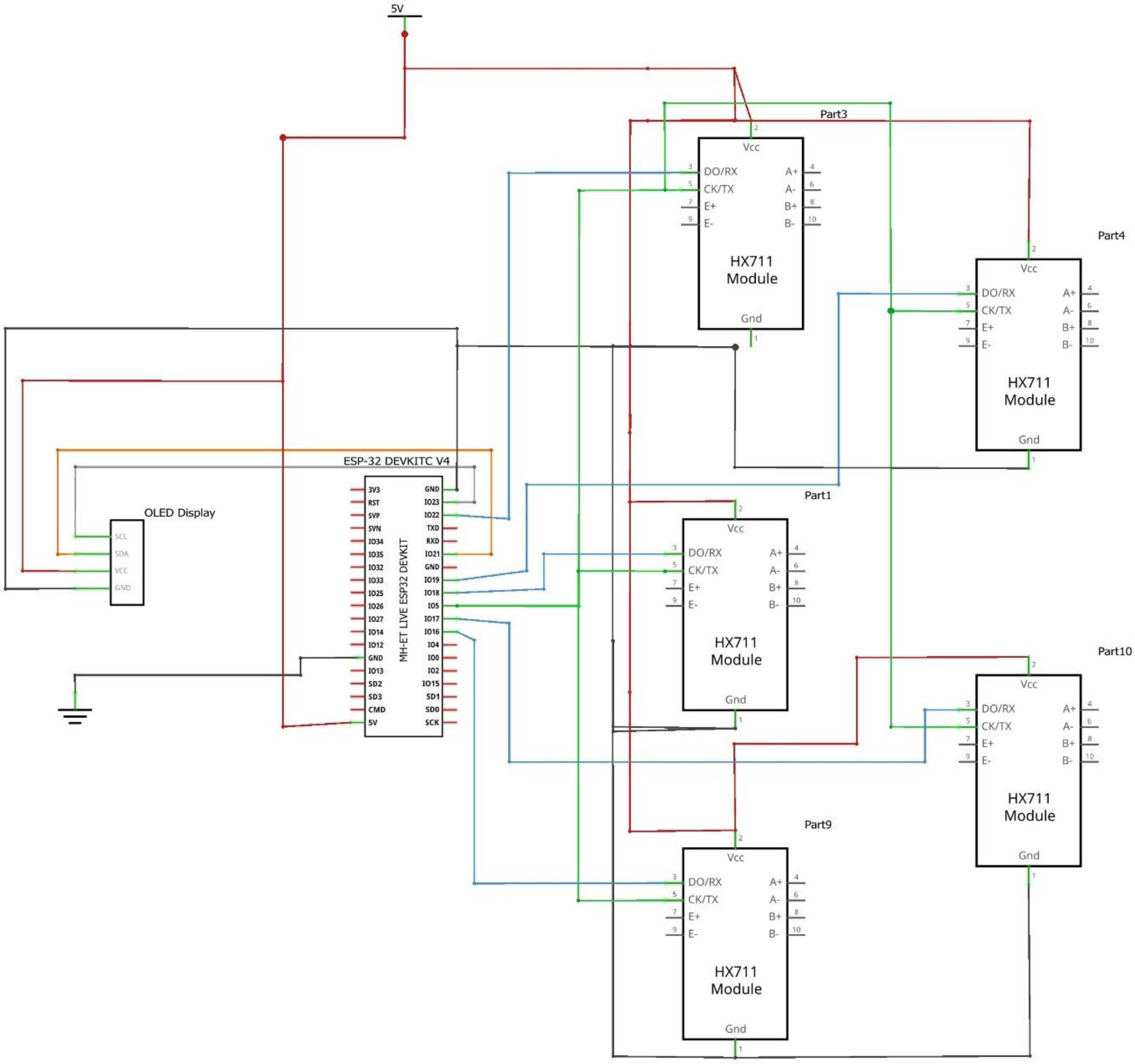


Figure 3-17: Mk.4 (ESP-32 (5 Gauges)) Circuit Diagram

As the Mk.4 is very similar to the previous iteration (Mk.3), the PCB schematic is adapted to represent the reduction from 14 Strain Gauges to just 5, as shown in Figure 3-18. Once again, this PCB schematic follows the legend in Figure 3-9.

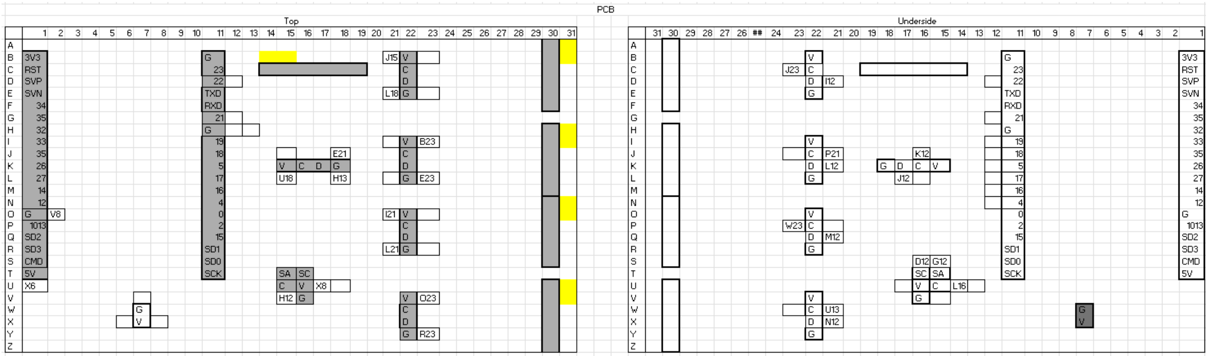


Figure 3-18: Mk.4 (ESP-32 5 Gauges) PCB Schematic

As this is the final iteration in the development of the motion tracking system, this iteration will be covered further in Section 5

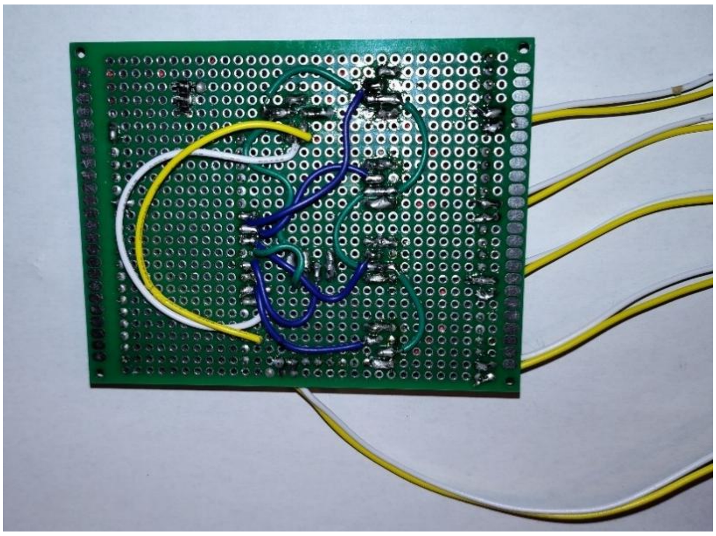


Figure 3-19: Completed Underside of the Mk.4 PCB

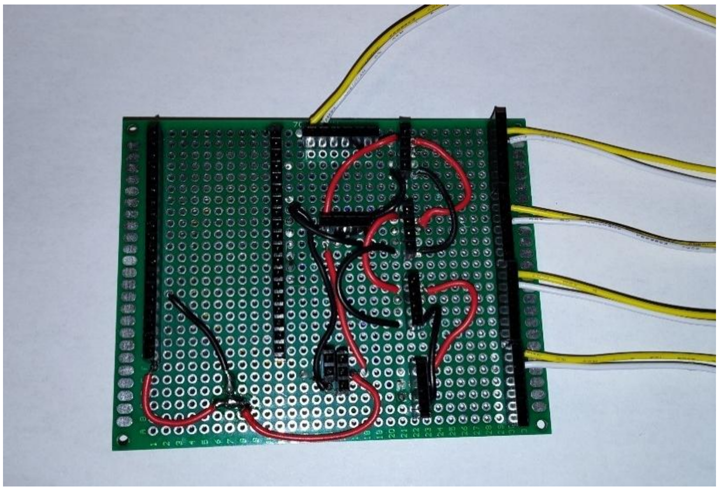


Figure 3-20: Completed Top of the Mk.4 PCB

3.2. 3D Printing

3.1.6. Testing

As one of the objectives of this project is to use 3D printed parts due to their relative ease to adapt to an individual user's needs (such as a smaller digit, slimmer wrist or less hand strength), each of the non-electric components of the digit tracking mechanism were designed with 3D printing in consideration. Figure 3-21 to Figure 3-24 show the first attempt at a 3D-printed casing, for Mk.1 of the digit position tracking device.

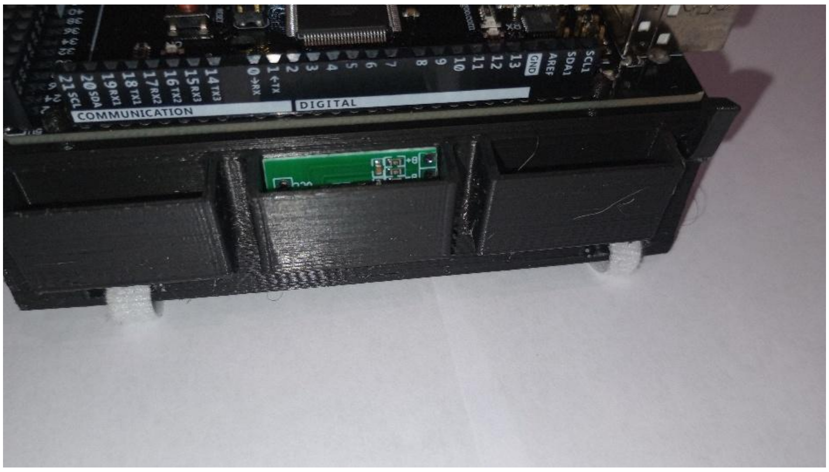


Figure 3-21: Mk.1 Casing - HX711 in place

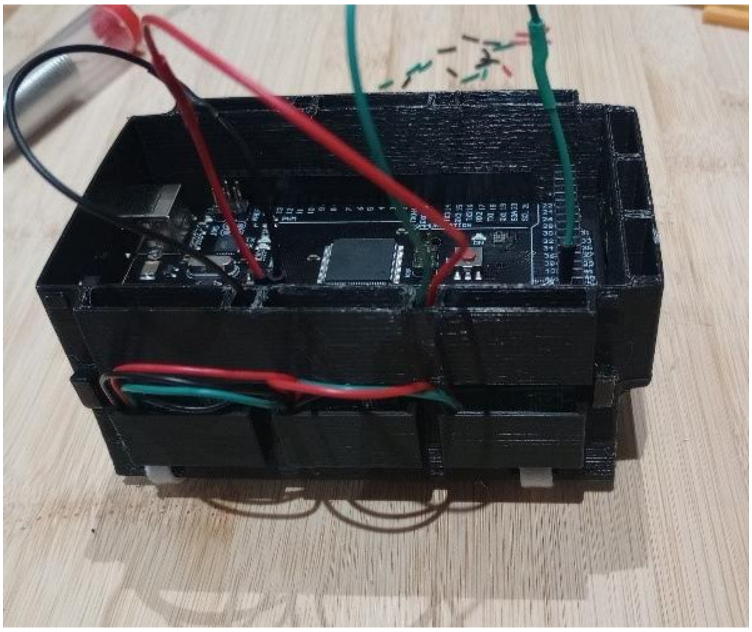


Figure 3-22: Mk.1 casing layers 1 & 2 (HX711 boards in place in Layer 1))

Figure 3-23: Mk.1 complete casing

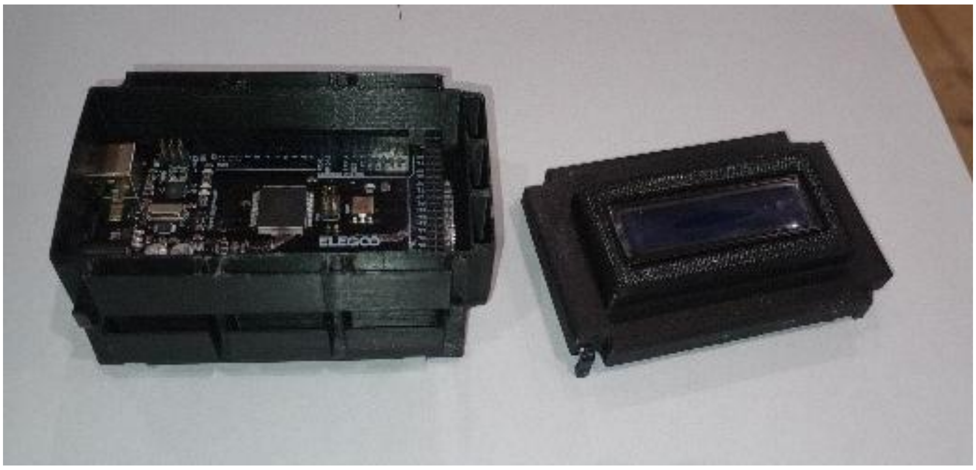


Figure 3-24: Mk.1 casing (Complete - Lid removed)

Each of the layers shown in Figure 3-21 to Figure 3-24 had multiple revisions and re-prints during the development, whether that was caused by software malfunctions (such as GCODE becoming corrupted), hardware issues (such as blocked nozzles and a faulty bed levelling sensor) or environmental factors such as the prints warping due to excessive heat within the 3D printer enclosure. A majority of these issues were solved by re-slicing the model, allowing the printer to cool more between prints and general maintenance of the printer.

For the issues that couldn't be resolved in these ways, more experimentation was needed. An example is the issue shown in Figure 3-25.

Figure 3-25: Misprint of the Mk.4 lid due to the support material failing to disconnect properly

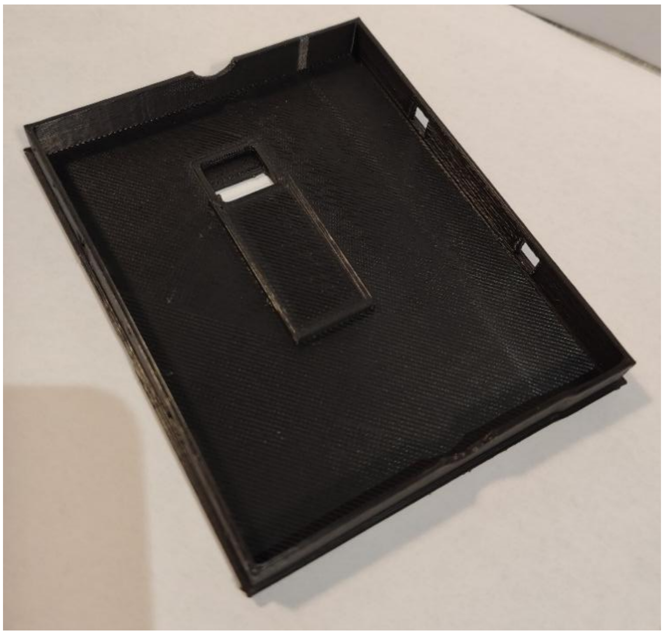


Figure 3-26: The underside of the misprinted Mk.4 lid, showing no defects

Figure 3-26 shows the underside of the Mk.4 lid shown in Figure 3-25, highlighting that the only issue with this print is that the support material failed to disconnect correctly from the printed part.

To minimise the impact of this misprint issue, the lid was printed in a different orientation so that the support structures were generated on the underside, and as such any print issues that are purely cosmetic would be hidden away inside the digit tracking device, improving the aesthetics of the outside.

Another issue that occurred while printing the Mk.4 wrist layer is that one side repeatedly became de-laminated during printing, so the part was a complete failure and waste of resources. This occurred twice, so research was conducted, and it was decided to modify the part to include manual snap-off support structures, as shown in Figure 3-27.

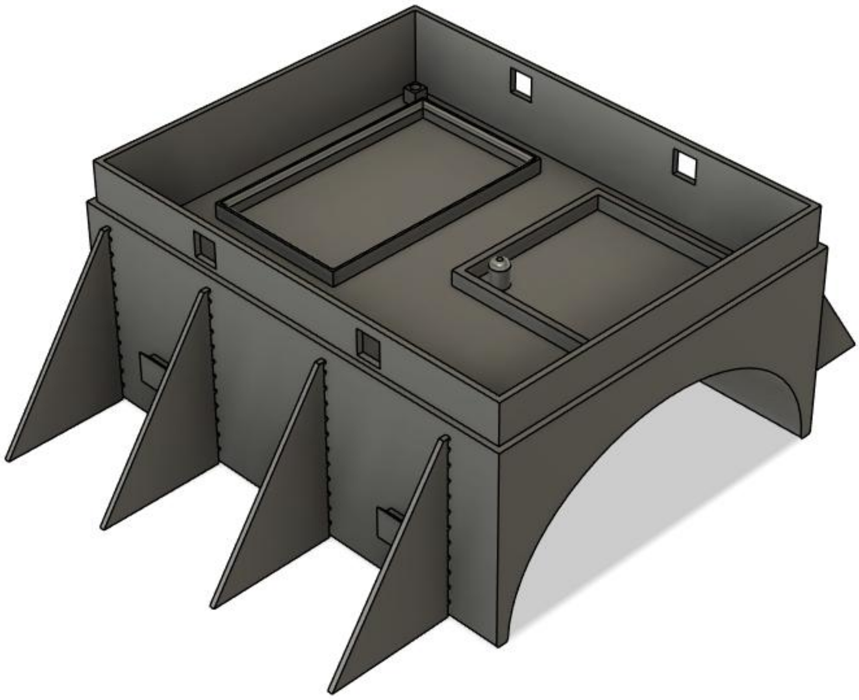


Figure 3-27: Mk.4 Wrist CAD model with manual snap-off support

The triangles shown on the sides of the Mk.4 Wrist CAD model in Figure 3-27 are designed to print alongside the model itself and ensure that the sides remain complete by providing support as the piece prints. It does this by being positioned perpendicular to the wall of the print, preventing it from falling or moving during the printing process and ruining the print.

3.1.7. Final Printing

3D printing takes time and resources, namely PLA plastic. As such, Table 3-1 shows the printing time & material cost for each component

Component	Material	Cost	Time
ESP-32 Wrist	74g	£1.34	9h 46m
ESP-32 Middle	24g	£0.43	2h 39m
ESP-32 Lid	42g	£0.75	4h 36m
Case Pin x4	2g	£0.03	14m
Back of hand	16g	£0.28	2h 55m
Thumb Panel	10g	£0.18	1h 31m
Hand beams x5	8g	£0.14	50m
Fingertip caps x4	21g	£0.38	3h 5m
Thumb cap	7g	£0.12	57m
TOTAL	204g	£3.65	27h 33m

Table 3-1: 3D printing resource cost (Mk.4)

4. Implementation

4.1. Initial Practical experimentation & Testing

Initially, it was assumed that the strain gauges could be superglued to a thin PLA (Polylactic acid, a common plastic for use by 3D Printers) (Joseph et al, 2023) beam, and this would bend along with the digit and cause a reading from the gauge. Under this assumption, tests were carried out to verify that the design would function on the testing system shown in Figure 4-1.

Figure 4-1: Initial Practical experimentation - Testing system (With Gauge attached)

A thin (0.6mm) PLA tab was 3D printed, and a strain gauge attached via two methods: attached along the length of the gauge and then attached only at the ends. Each of the small black tabs shown on the testing system each represent 10 degrees of deflection from the initial reading (as shown in Figure 4-2), and the PLA tab was bent to each tab consecutively and the strain gauge reading recorded in Table 4-1.

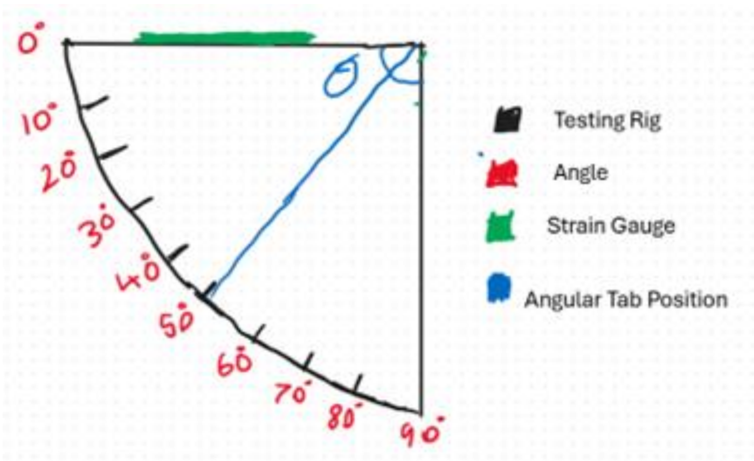


Figure 4-2: Explanation of the Initial Testing System

Degree	Fixed Ends				Fully fixed along length													
	Attempt 1	Measured Angle	Attempt 2	Measured Angle	Attempt 1	Measured Angle	Attempt 2	Measured Angle	Attempt 3	Measured Angle	Attempt 4	Measured Angle	Attempt 5	Measured Angle	Attempt 6	Measured Angle	Attempt 7	Measured Angle
0		N/A	9196	9.29	7636	17.92	10238	3.77	10736	-2.20	1941	75.19	10211	25.50	10707	22.77	15358	5.60
10	13086	9.12	8191	18.11	5074	42.10	10492	1.63	9128	11.61	-18	90.14	9695	28.76	8863	34.35	9726	36.55
20	7933	40.97	7710	22.33	4025	52.00	10110	4.85	8161	19.92	2334	72.19	8180	38.33	7837	40.79	8759	41.86
30	7677	42.55	7679	22.60	2965	62.01	8095	21.82	7749	23.45	-2	90.02	6111	51.40	6023	52.18	7327	49.73
40	8165	39.54	8191	18.11	1180	78.86	5630	42.58	7546	25.20	1948	75.13	6328	50.03	4606	61.08	4088	67.53
50	7935	40.96	7927	20.42	164	88.45	3986	56.43	5372	43.87	967	82.62	-248	91.57	4052	64.56	3173	72.56
60	7153	45.79	7159	27.17	481	85.46	36	89.70	4855	48.31	1020	82.22	-338	92.14	2983	71.27	1887	79.63
70	6943	47.09	6552	32.49	494	85.34	-342	92.88	4039	55.31	-1	90.01	-321	92.03	2090	76.88	1477	81.88
80	6652	48.89	4029	54.64	-17	90.16	-19	90.16	4223	53.73	-860	96.56	-257	91.62	1266	82.05	829	85.44
90	3827	66.35	3903	55.74	-50	90.47	-920	97.75	2430	69.13	-193	91.47	-13	90.08	530	86.67	430	87.64
Tare Value	14562		10254		9534		10686		10480		11792		14247		14333		16377	

Table 4-1: Strain Gauge practical data (Initial Testing)

Table 4-1 shows the data collected during the initial testing and is sub-divided into two sections: Fixed ends and Fixed along the length. These sections refer to how the strain gauge used was attached to the PLA tab, and repeats were taken for each method.

Degree	Fixed Ends				Fully fixed along length													
	1	Error	2	Error	1	Error#	2	Error	3	Error	4	Error	5	Error	6	Error	7	Error
0	N/A	N/A	9.29	N/A	17.92	#DIV/0!	3.77	N/A	-2.20	N/A#	75.19	N/A	25.50	N/A	22.77	N/A	5.60	N/A
10	9.12	8.78%	18.11	81.07%	42.10	321.02%	1.63	83.66%	11.61	16.11%	90.14	801.37%	28.76	187.56%	34.35	276.52%	36.55	265.51%
20	40.97	104.85%	22.33	11.64%	52.00	160.02%	4.85	75.74%	19.92	0.42%	72.19	260.93%	38.33	91.63%	40.79	0.44%	41.86	109.32%
30	42.55	41.84%	22.60	24.66%	62.01	106.70%	21.82	27.26%	23.45	21.82%	90.02	200.05%	51.40	71.32%	52.18	22.63%	49.73	65.78%
40	39.54	1.16%	18.11	54.73%	78.86	97.15%	42.58	6.46%	25.20	37.01%	75.13	87.83%	50.03	25.06%	61.08	54.49%	67.53	68.84%
50	40.96	18.08%	20.42	59.15%	88.45	76.90%	56.43	12.86%	43.87	12.27%	82.62	65.24%	91.57	83.13%	64.56	57.62%	72.56	45.13%
60	45.79	23.68%	27.17	54.72%	85.46	42.43%	89.70	49.49%	48.31	19.49%	82.22	37.03%	92.14	53.56%	71.27	55.64%	79.63	32.72%
70	47.09	32.73%	32.49	53.58%	85.34	21.91%	92.88	32.69%	55.31	20.98%	90.01	28.58%	92.03	31.47%	76.88	63.26%	81.88	16.98%
80	48.89	38.89%	54.64	31.70%	90.16	12.70%	90.16	12.70%	53.73	32.83%	96.56	20.70%	91.62	14.53%	82.05	67.84%	85.44	6.81%
90	66.35	26.28%	55.74	38.06%	90.47	0.52%	97.75	8.61%	69.13	23.19%	91.47	1.64%	90.08	0.09%	86.67	30.63%	87.64	2.63%
Average	32.92%		45.48%		93.26%		34.39%		20.46%		167.04%		62.04%		69.89%		68.19%	

Table 4-2: Percentage Error (Initial Testing system)

Table 4-2 calculates the percentage error for the data collected in Table 4-1, and highlights that there is no consistent pattern impacting the results. This wild variation within the results shows that the initial method for tracking the digits' motion using strain gauges was unsuitable, and a new method was required to make the project feasible.

4.2. Final Practical experimentation & testing

Following research into current strain gauge usages, such as in load cells (Proctor, 2022), a breakthrough was made in the method to use the strain gauge to measure the digits' position.

As strain gauges function by measuring small strains on the surface of an object, the initial method was providing too much strain and damaging the gauges. As such, a new method was devised to reduce the amount of strain the gauge was subjected to, and the testing system in Figure 4-3 was devised.

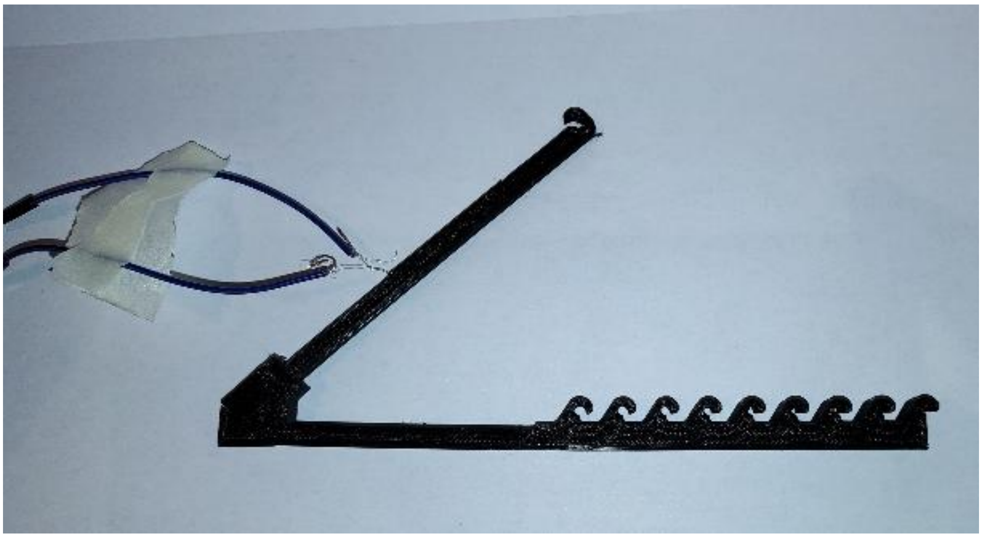


Figure 4-3: Final Practical testing system

In this testing system, the Strain gauge is fixed on the back of the "hook" (the large beam protruding upright at a 45° angle) before being calibrated to zero. Measurement can now begin; the aim was to simply act as a proof of concept, and that strain gauges could be used to measure linear movement.

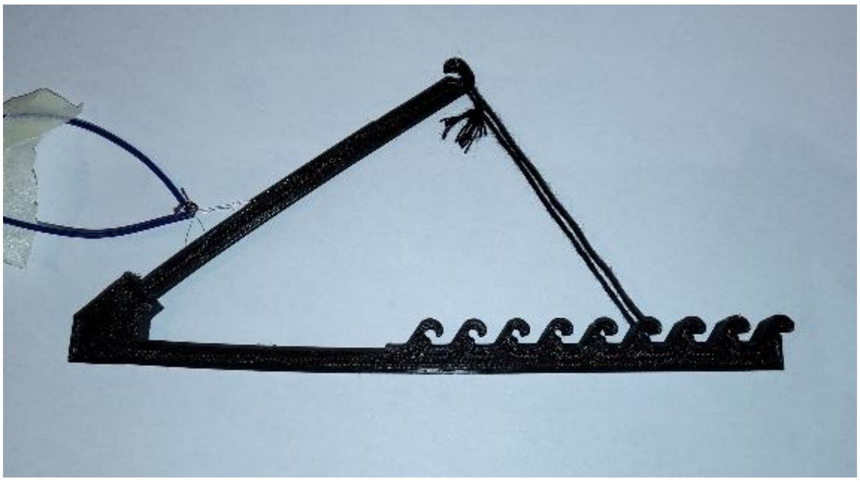


Figure 4-4:Final Practical testing system (In use)

Figure 4-4 shows the testing system in use, using a loop of strong thread to act as a fixed distance between the "hook" with the strain gauge attached and a corresponding hoop on the base.

A loop of thread was used as it has a low modulus of elasticity while still remaining flexible and is also strong enough to withstand the strain produced from the “hook” beam bending.

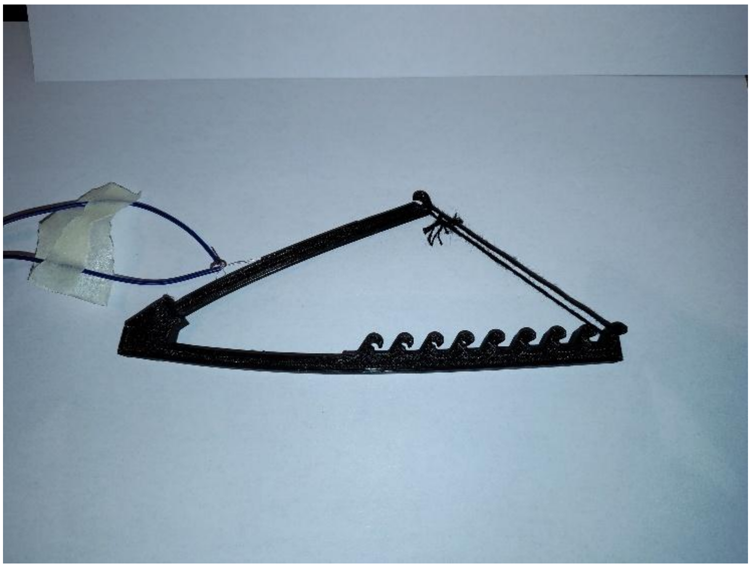


Figure 4-5: Final Practical testing system (Maximum Strain)

Figure 4-5 shows the issue that occurred as the stress and strain being measured increased; the base began to bend along with the beam. This is an issue as it means that a proportion of the strain that should be acting upon the “hook” beam is instead acting on the base, which will begin to skew the results, as shown in Figure 4-6.

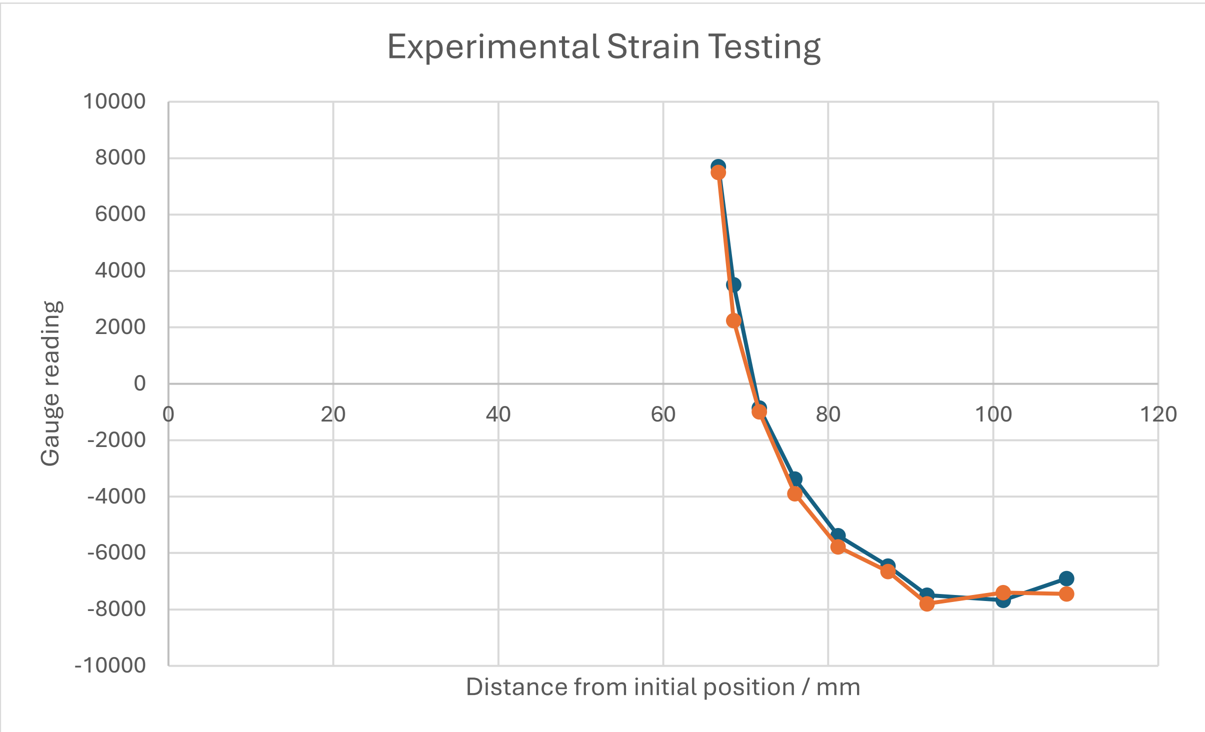


Figure 4-6: Experimental Strain Measurement against displacement (Final Testing System)

Figure 4-6 shows that there is initially a linear relationship between the strain being measured and the distance the “hook” beam is displaced. This is critical to the development of the motion tracking system as it means that a relationship between a digit's positioning and measured strain can be identified and used to calculate the position from the strain produced.

However, the later data in Figure 4-6 highlights an issue with the construction of the system that must be solved; the material of the base must have a stronger tensile strength than that of the beam.

This difference in materials will ensure that much of the strain remains acting upon the "hook" beam, and an accurate measurement can be taken to calculate the position of the digit.

5. Final Product

To reiterate, the aim of this project was to produce a device capable of tracking each digits' motion using strain gauges, while being able to be reproduced for less than £100. An instruction manual was also produced to provide the user with an assembly guide for the device, alongside the links to purchase the required components and suggested resources for further research.

5.1. Costings

A breakdown of the item pricing is shown in Table 5-1.

Name	Amount	Price	Total
ESP-32 DevkitC V4	2	£10.99	£21.98
HX711 Amplifier Boards	5	£1.79 for 5	£1.79
BF350-30AA Strain Gauge	5	£2.07 for 2	£6.21
Dupont female to female connectors	6	£5.49 for 90	£5.49
0.96" OLED screen	1	£1.63	£1.63
Prototyping PCB board – 70x90mm	1	£1.28	£1.28
9V step-down transformer board to 5V	1	£0.62	£0.62
9V Battery connector	1	£0.58	£0.58
9V Battery	1	£2.15	£2.15
M2 Brass standoffs – (Set)	-	£8.99	£8.99
PCB Male & Female pin headers (Set)	-	£1.43	£1.43
26 AWG Wire (Assorted Colours) (Set)	-	£6.99	£6.99
Switch	1	£0.61 for 10	£0.61
10mm strapping	1M	£1.59 for 5 yards	£1.59
10mm buckles	2	£1.05 for 10	£1.05
3D Printing	204g		£3.65
Total Cost			£66.04

Table 5-1: Costings for the Prototype

5.2. Accompanying documentation

The Instruction manual produced can be found in Appendix D.

5.3. Physical product

The main objective of this project is to design & produce a portable digit tracking system, this section details the practical construction of the system.

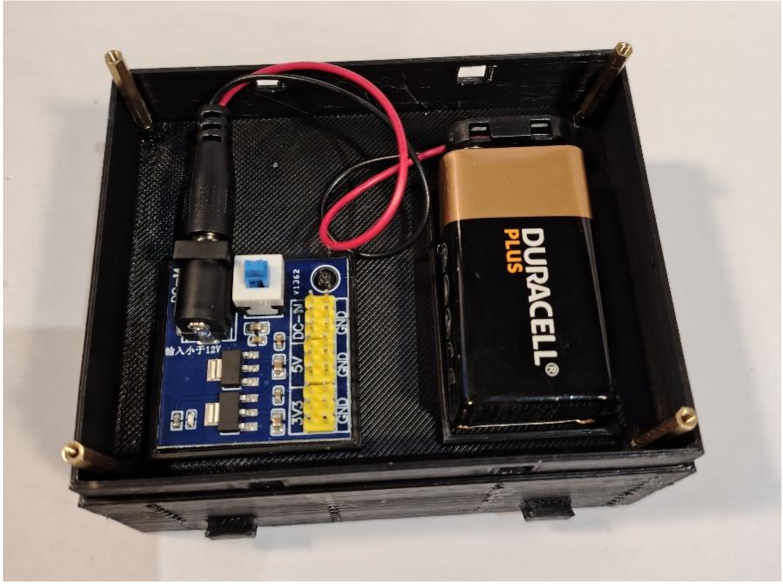


Figure 5-1: Inside Layer 1 of Mk.4

Figure 5-1 shows the full layout of the inside of the Mk.4 ESP-32 wrist base layer, with the 9v battery and 5V transformer board installed in their defined areas.

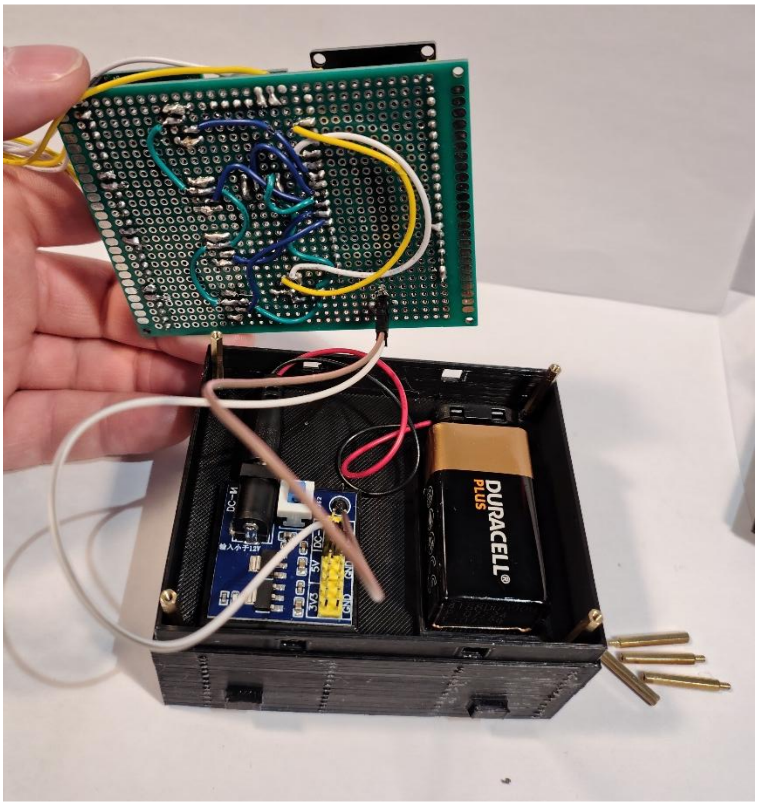


Figure 5-2; Connection between 5V transformer board and the PCB (Mk.4)

Figure 5-2 shows how the 5V transformer board connects to the male header pins on the underside of the Mk.4 PCB – this connection is achieved through the use of two Dupont Female-Female Jumper pins.

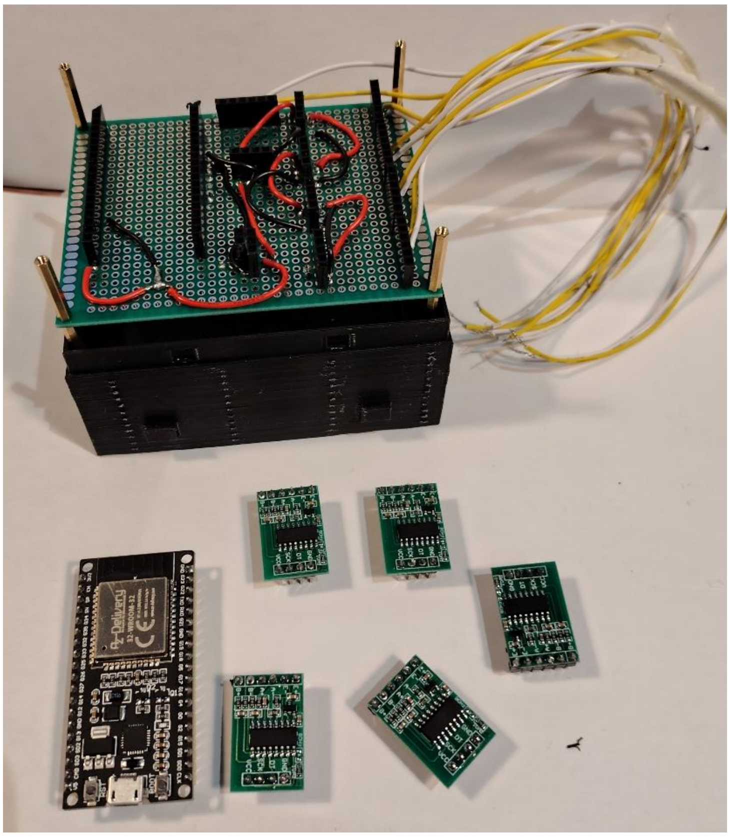


Figure 5-3: PCB in place, with components ready to insert (Mk.4)

Figure 5-3 shows the PCB installed on the M2 brass standoffs, alongside the components that will install into the female header pins on the PCB's top face.

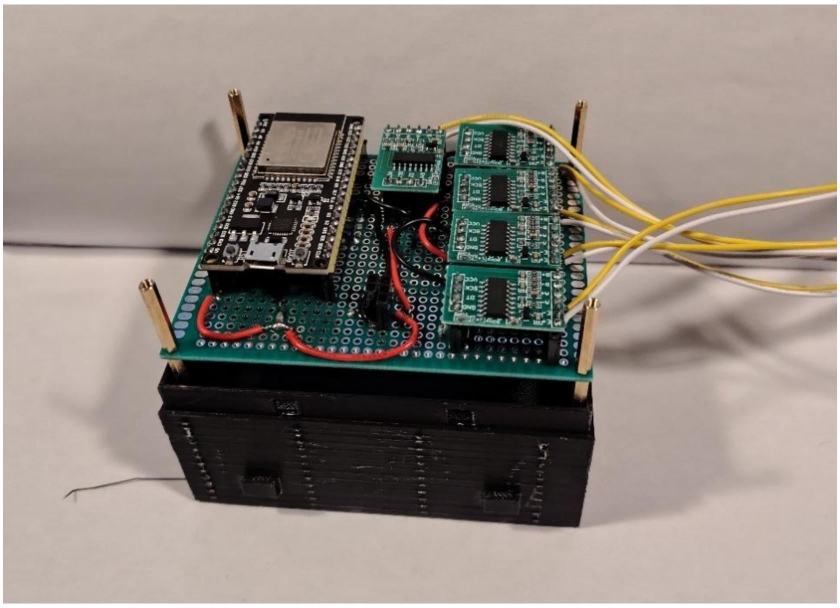


Figure 5-4: Fully assembled PCB & Wrist casing (Mk.4)

Figure 5-4 details the location of each of the 5 HX711 boards and ESP-32, and how they slot into and connect with the Mk.4 PCB.

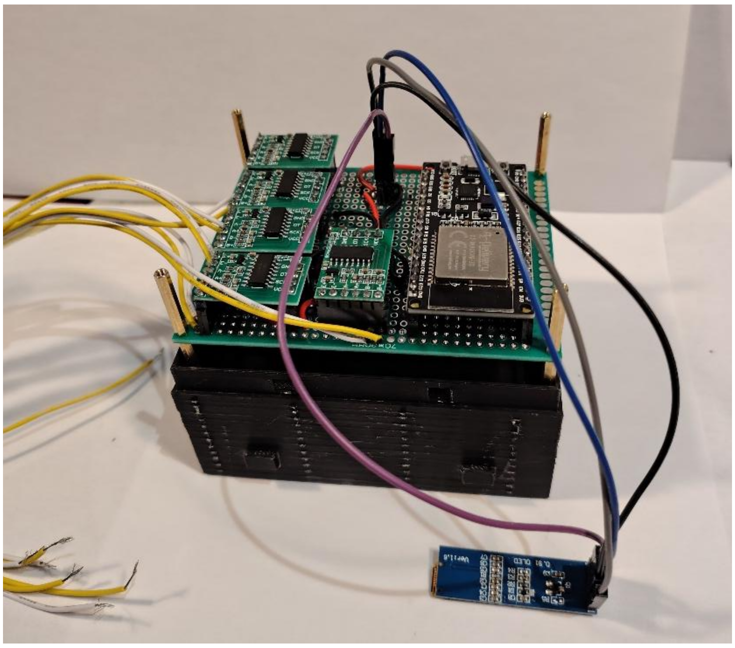


Figure 5-5: Mk.4 - showing connection to OLED screen

Figure 5-5 shows the location of the OLED connector pins, the 5 female headers located in the centre of the Mk.4 PCB. The connection to the OLED screen is made now using 4 DuPont Male to Female jumper cables and is made now when the header pins are more accessible.

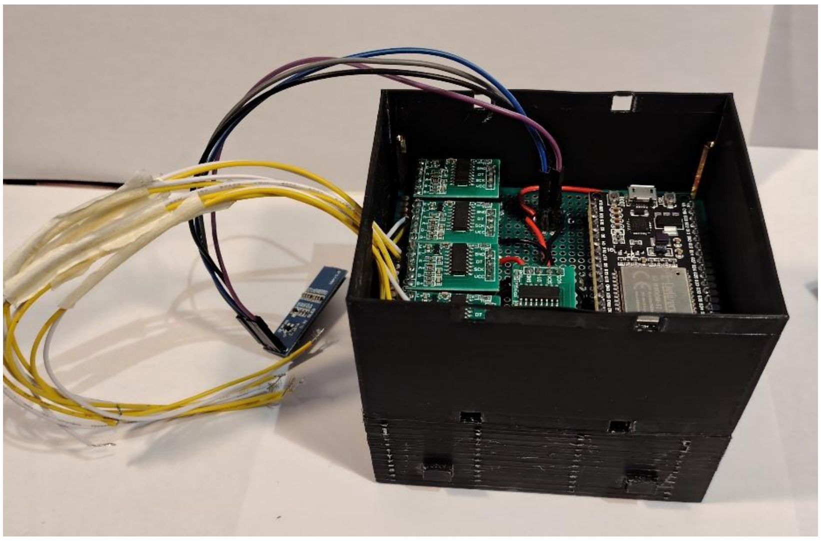


Figure 5-6: Middle layer added onto PCB & Base layer (Mk.4)

Figure 5-6 shows how the wires that lead to the strain gauges feed through the circular hole within the middle layer of the Mk.4 ESP-32 wrist casing, alongside also showcasing how the middle layer slots around the PCB onto the base layer.

Figure 5-7: Mk.4 Lid with OLED screen in place (Top)

Figure 5-7 is of the top of the Mk.4 ESP-32 wrist casing lid, showing how the OLED screen slots into the space provided while still concealing the header pins required to make it function.

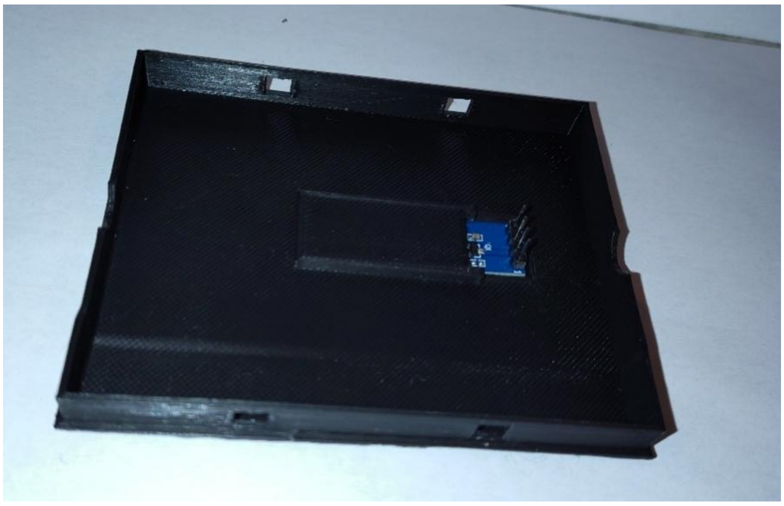


Figure 5-8:Mk.4 Lid with OLED screen in place (Underside)

Figure 5-8 shows the underside of the Mk.4 ESP-32 wrist casing lid, focusing on how the header pins for the OLED screen are still accessible for the internal circuitry.

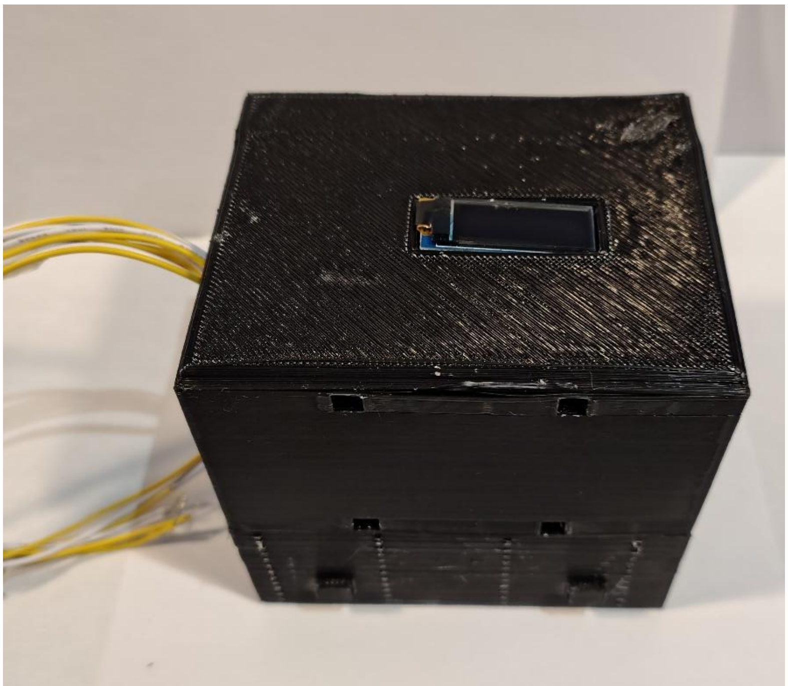


Figure 5-9: Fully assembled wrist casing (Mk.4)

Figure 5-9 shows the fully assembled Mk.4 ESP-32 Wrist casing and shows how each of the three layers slot together to form a single box with simple edges. Figure 5-9 also shows the OLED screen in place on top of the casing, and how the wires feed out of the middle layer ready to be connected to the strain gauges.

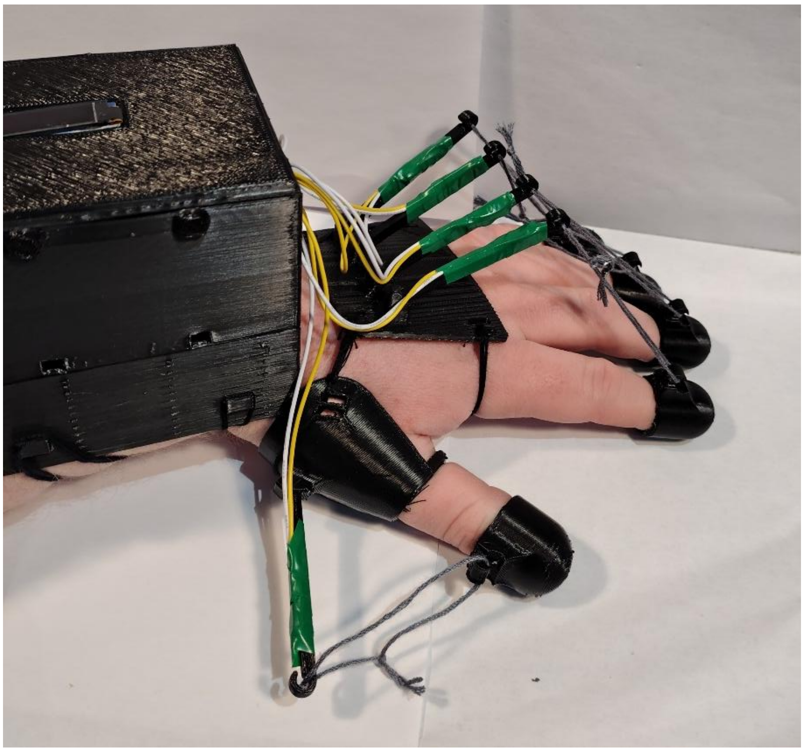


Figure 5-10: Fully assembled Mk.4 digit motion tracking systen (on Hand)

Figure 5-10 shows the fully assembled Mk.4 Digit Tracking System, with the white and yellow wires feeding from the middle of the wrist casing to each of the individual gauges on a PLA hooked beam. Each beam is wrapped in green electrical tape, covering the strain gauge and soldered connections – This is to ensure that no loose wires can cause gauges to interfere with one another, as well as ensuring that each gauge is isolated and fully follows the motion of the beam as the finger bends.

For the Mk.4 (shown in Figure 5-10), strong thread was tied in a loop of a given circumference and used to connect the beam and the fingertip cap for each digit. Strong thread was chosen to connect the beam and fingertip cap as it allows for variability in each user's hand, as using a solid PLA link of a fixed length would be a waste of resources for a user with larger or smaller hands than the designer's. Thread also allows for adjustments to be made on the amount of tension on the fingers, as a smaller loop means that the beam on the back of the hand will have to be pre-loaded with strain to achieve the distance required between the fingertip cap and the beam.

Also, once the correct size of thread loop has been found for each user, they have the option to upgrade the thread loop into something more durable, such as a 3D printed TPU (Thermoplastic Polyurethane) hoop of the given size.

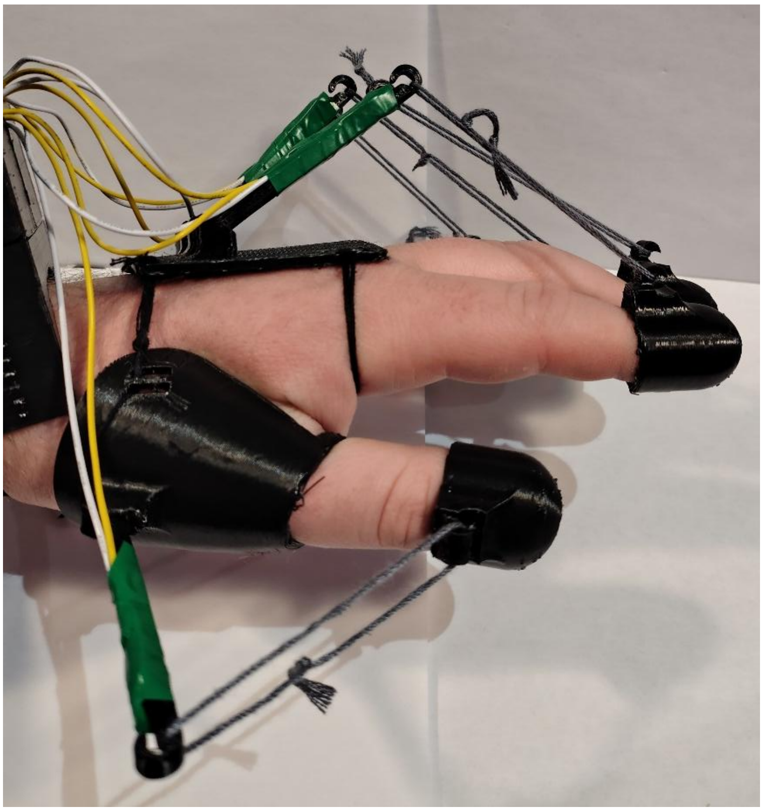


Figure 5-11: Mk.4 Practical testing - Flat hand

Figure 5-11 shows the tension in the beams when the hand is held flat, as shown by the tightness of the thread loops between the beams and fingertip caps.

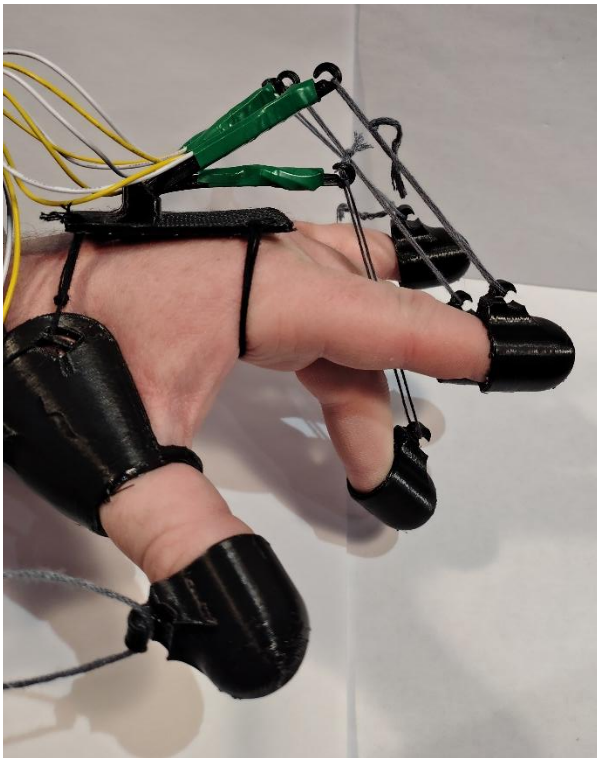


Figure 5-12: Mk.4 Practical testing - Curled finger (Singular)

Figure 5-12 highlights the difference in the beams' location when a finger is curled, as shown by the middle digit & corresponding hand beam.

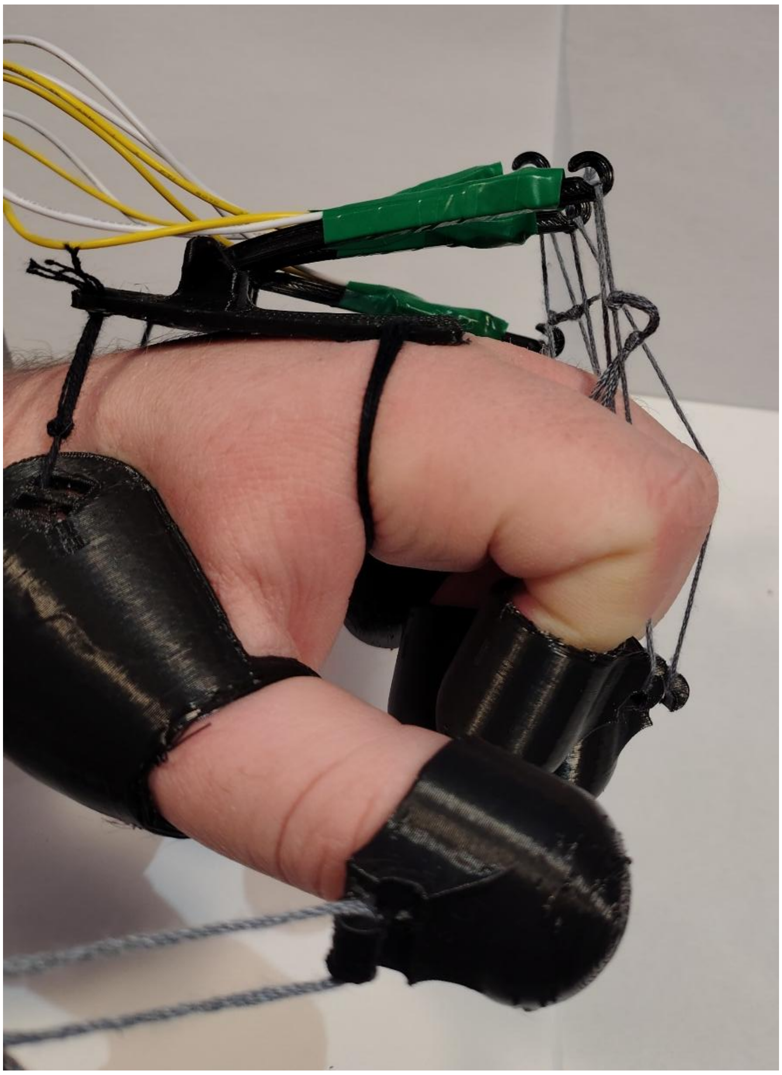


Figure 5-13: Mk.4 Practical testing – Fist

Figure 5-13 shows how each of the beams act when the hand is curled into a fist – each of the beams bend a similar amount as the others.

Figure 5-13 also highlights three issues with the current Mk.4 design – The first issue is that during testing, the panel on the back of the hand broke, causing the beam that corresponds to the little finger to break loose, as can be seen in the background of the photo.

The second issue is that the panel on the back of the hand wasn't secured properly, leading to the back of the panel (the end closes to the wrist) to lift, which is what caused the issues with the data collected during the practical experimentation (Section 4.2), as can be seen in Figure 4-6.

6. Analysis

6.1. Current Issues

After analysing the current Mk.4 design, there are three current issues that come to light. These are:

- The panel on the back of the hand:
  - Bends when the hand beams are under strain
  - Isn't currently strong enough to withstand the strain of a closed fist
- The wrist box is bulky in comparison to the remain of the system

Figure 5-12 & Figure 5-13 show the weakness of the hand panel, as it is being lifted in both photos. The action of the support for the hand beam of the little finger snapping also shows that the panel needs to be redesigned to provide sturdier support for each of the hand beams without compromising their range of motion, and the panel requires general re-enforcement to ensure that the readings from the strain gauges aren't impacted by the failure of the panel to remain fixed in place

The bulkiness of the Mk.4 wrist box can be seen by the amount of empty space inside, as shown in Figure 5-2. It is also more visible in Figure 5-10, where the size of the wrist box can be seen to be more than three times larger than the hand. The large size of the Mk.4 ESP-32 wrist casing increases the chance of it becoming damaged during use, and in turn could lead to the device breaking and requiring repair or replacement.

6.2. Suggested Improvements

To counter these issues highlighted in section 6.1, the following changes could be made:

- Redesign the back of hand & thumb to be single, wrap-around panel
- Integrate the power transformer into the PCB
- Directly solder components to the custom PCB, rather than using headers

The benefits of redesigning the panel on the back of the hand & the thumb into a single panel that wraps around the hard are that it would eliminate both issues that the hand panel has. It will do this by providing extra support from the underside (palm) of the hand, resisting the bending moment that forms when the beams are under strain from the fingers flexing. Having a larger hand panel would also mean that there would be more available space on the panel to re-enforce the connecting area between the panel and the hand beams.

The merging of the thumb and hand panel would also improve the accuracy of the thumb's strain gauge tracking, as the current panel partially moves with the thumb when it bends. Introducing experimental error into the positional readings.

The integration of the standalone transformer board into the custom PCB would allow space to be saved within the wrist layer, and the battery could be moved to an external box on the end of the current wrist box. The moving of the battery would grant better access to the power switch, as well as providing space for the ESP-32 PCB layer to move closer to the wrist, reducing the overall height of the wrist box.

A simpler improvement would be to remove the use of the female headers from the PCB and instead directly solder each component (HX711 boards and the ESP-32) to the circuit board, as this would reduce the overall height of the wrist box by 11mm (from 17mm to 6mm).

7. Conclusion

Over the course of this project, the following objectives have been met:

- Research was conducted into current methods of gesture recognition and digit positional tracking, providing a wide range of example devices and systems that function both via wired connection and wireless connections, such as Bluetooth.
- Research was conducted into strain gauges, how they work and how to interface them with microcontrollers – both an Arduino and ESP-32 have been used through the duration of the project to measure the strain a given object is placed under

A device capable of tracking a digit's motion through a regular range of motion has been produced, and complies with the following criteria from the initial aims and objectives:

- The device must be produced in a way which allows it to be replicated without the use of any specialised equipment (Excluding the use of a Soldering iron and 3D printer)

This objective has been met, as the entire device is comprised from off-the-shelf components and a 3D printed casing. The most specialised aspect of the device, excluding the coding of the microcontroller, is the fabrication of the custom PCB for the Mk.3, but this process is fully catalogued and explained within both this report and the Instruction manual in Appendix D.

- The device must be portable and able to function wirelessly

As the system includes a 9V battery for power and utilises an ESP-32 as the microcontroller, the device is both able to be powered internally by the battery alongside able to use the ESP-32 built-in wireless functionality to transmit the data collected from the strain gauges.

- The device must be designed in a way that makes it adaptable to suit a wide range of people's hand sizes, dexterity and strength

As the casings of the system are entirely 3D printed, the CAD models will be made available to any user should they wish to adapt the designs to better suit themselves, such as changing part sizes to fit a smaller or larger hand. The CAD models have been developed using dimensions wherever possible, meaning that a user would simply have to double-click the number and modify the part to suit their needs.

Also, to account for the wide range of dexterity and hand strength within the population, the hand beam part can be 3D printed using a different material (Such as TPU) or printed with lower percentage infill to change the strength required to displace it, but this would require further research into the mechanical and structural properties of different percentage infills and the various types of infill also.

- The device must be replicable for under £100

As mentioned previously in Table 5-1, the cost of the materials (excluding tools such as the Soldering iron and 3D printer, and expendable resources such as electricity and solder) the system can be replicated for £66.04, and that is operating under the assumption that the user is producing it from the ground up, and doesn't have spare components such as jumper cables and M2 standoffs in storage at home. If the user were to build two or more of these devices, the per-item cost would be cheaper due to the remaining spares from the 1st device, such as jumper wires, switches, standoffs and other miscellaneous items.

- An instruction manual must be written to accompany the device

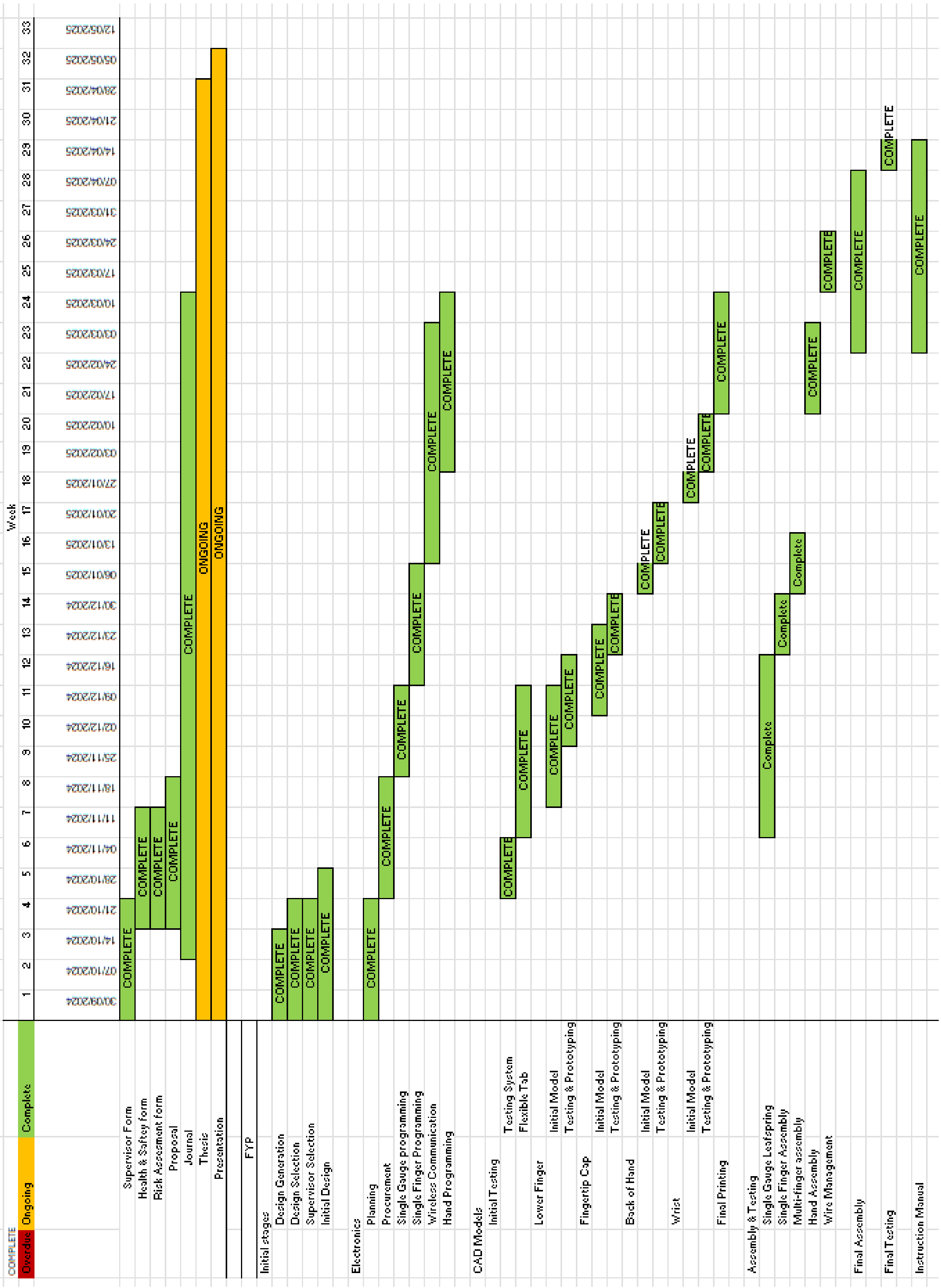
As previously mentioned in Section 5.2, the accompanying instruction manual can be found in Appendix D.

The next steps for this project are to follow the recommendations made in Section 6.2 and organise the production of a professional custom PCB that can be used within the design without the user needing to fabricate it themselves. Further research could also be conducted into improving the sampling rate of the system, and reducing the latency between the data being collected, transmitted and then received by the computer.

While the development of this project and digit tracking system has primarily had the gaming sector as a focus, the system developed could also be used in other sectors, such as the medical sector using the system to measure the strength of a person’s tendons individually, digit by digit, through the amount of force they can exert on the system. This system could also be used by companies such as Nasa, as the system weighs less than 500 grams and is more than 50% 3D printable, and therefore reparable in case of breakage in less than 10 hours for the largest 3D printed component.

8. Appendices

8.1. Appendix A: Gantt Chart



8.2. Appendix B: Health & Safety Risk Assessment

GENERAL RISK ASSESSMENT FORM			**Severity** multiplied by **Likelihood** equals **Risk Rate**. NB: Calculated after taking in to account existing precautions				
School/Service DTA		Severity / Likelihood	Insignificant (1)	Minor (2)	Moderate (3)	Serious (4)	Fatal / Critical (5)
Task/Activity/Area: FYP – Strain Gauge Finger Position Tracking		Almost Certain (5)	5	10	15	20	25
		Likely (4)	4	8	12	16	20
Assessed By: (Print Name):	Signature: Daniel Brittle	Possible (3)	3	6	9	12	15
Dept Manager: (Print Name) Dr Alison Griffiths	Signature: Alison GRIFFITHS	Unlikely (2)	2	4	6	8	10
Date of Assessment: 29/10/24	Review Date: 4/04/25	Rare (1)	1	2	3	4	5

	Activity/Process/ Machines	**Hazard**	**Persons in Danger**	**Severity 1-5**	**Likelihood 1-5**	**Risk Rate**	**Measures/Comments**	**Result**
1	Time spent on computer/ CAD Designs and Electronics Programming	• Back Pain from prolonged sitting • RSI • Eye strain	• Student	2	3	6	• Regular breaks away from screens, such as 5 minutes every 30. • Ergonomic chair • Blue light filters on screen • Follow HSE guidelines for computer work	A
2	3D Printing	• VOCs released • Moving parts • Heated parts • Microplastics released • Electrical components • Noise Levels	• Student • Those Present in the room	4	2	8	• A Sealed, filtered enclosure • HEPA filtration • PAT Tested electrical components • Follow HSE guidelines regarding VOC exposure and 3D printing	A
3	Soldering	• Heated Elements • Fumes released • Eye Strain • Electrical components	• Student • Those present in the room	3	3	9	• Adequate Ventilation when soldering • Filter mask to be worn when soldering • Regular breaks • Magnifying glass available • PAT testing • Clear, Organised workspace	A
4	System Testing	• Electric shock • Pinched skin	• Student	3	2	6	• Wear a protective layer underneath, such as a thin glove • Follow proper lab conduct • Follow lab H&S requirements	A

Key to result **T** = Trivial Risk **A** = Adequately Controlled **N** = Not Adequately Controlled **U** = Unable to decide (further information required).

8.3. Appendix C: ESP-32 Code

8.3.1. Transmitter:

```
// Including all the necessary Libraries:
#include <HX711.h>
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <esp_now.h>
#include <WiFi.h>
// Setting up the display screen
#define SCREEN_WIDTH 128 // Display width (Pixels)
#define SCREEN_HEIGHT 32 // Display height (Pixels)
#define OLED_RESET -1 // Reset pin
#define SCREEN_ADDRESS 0x3C // I2C Screen address
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET) ;

// Setting up ESP-NOW
//Put the Reciever MAC address here:
uint8_t broadcastAddress[] = {0xf8, 0xb3, 0xb7, 0x48, 0xbd, 0x1c} ;
// This part defines what sort of data is being sent
typedef struct struct_message {
  int a ;
} struct_message ;
// defining the message name to be sent via ESP-NOW
struct_message GaugeData ;
esp_now_peer_info_t peerInfo ;

// Setting up HX711 gauge names and Pins
int Index = 22 ;
int Middle = 19 ;
int Clock = 5;
int Ring = 18 ;
int Little = 17 ;
int Thumb = 16 ;

// Other Global Variables to be Used
```

```
float LocalRatio ;
float AngleRatio ;
int X = 0 ;
float DigitValue ;

// Setting up remaining stuff for the HX711 gauges
HX711 scale1 ; HX711 scale2 ; HX711 scale3 ; HX711 scale4 ; HX711 scale5 ;

// Setting up a generalised conversion function for the gauges' raw values to be converted to angle measurements
int DigitMin[5] = {0,0,0,0,0} ;
int DigitMax[5] = {20000,20000,20000,20000,20000} ;
int DigitScaleRead[5] = {0,0,0,0,0} ;

// Setting up a tare function for all gauges simultaneously
int GaugeTare() {
  scale1.tare() ; scale2.tare() ; scale3.tare() ; scale4.tare() ; scale5.tare() ;
}
// Setting up a function to read all gauges at once
int ReadScales() {
 DigitScaleRead[0] = scale1.read() ;
 DigitScaleRead[1] = scale2.read() ;
 DigitScaleRead[2] = scale3.read() ;
 DigitScaleRead[3] = scale4.read() ;
 DigitScaleRead[4] = scale5.read() ;
}
// Setting up Blank Angle Array
int DigitAngle[5] = {0,0,0,0,0} ;
```

```
// Setting up Angle Calculation Function
int AngleCalculation(int X) {
  int Max = DigitMax[X] ;
  int Min = DigitMin[X] ;
  int DigitValue = DigitScaleRead[X] ;
  LocalRatio = Max - Min ;
  AngleRatio = DigitValue / LocalRatio ;
  DigitAngle[X] = AngleRatio * 90 ;
  LocalRatio = 0 ;
  AngleRatio = 0 ;
  DigitValue = 0 ;
  return DigitAngle[X] ;
}

void setup() {
  // put your setup code here, to run once:
  Serial.begin(19200) ;
  display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS) ;
//Setting up ESP-NOW
  WiFi.mode(WIFI_STA) ;
  if (esp_now_init() != ESP_OK) {
    Serial.println("Problem encountered starting ESP-Now") ;
    return ;
  }
//Remembering the P2P reciever
  memcpy(peerInfo.peer_addr, broadcastAddress, 6) ;
  peerInfo.channel = 0 ;
  peerInfo.encrypt = false ;
  if (esp_now_add_peer(&peerInfo) != ESP_OK) {
    Serial.println("Can't add peer") ;
    return ;
  }
//Initialising the scale setting for each pin & Clock
  scale1.begin(Index, Clock) ;
  scale2.begin(Middle, Clock) ;
  scale3.begin(Ring, Clock) ;
  scale4.begin(Little, Clock) ;
  scale5.begin(Thumb, Clock) ;
// OLED screen initialization
  display.clearDisplay() ;
  display.setTextSize(1) ;
  display.setTextColor(SSD1306_WHITE) ;
  display.display() ;
  display.setCursor(0,0) ;
  display.println("Initializing....") ;
  int CurrentTime = millis() ;
//Calibrating the Gauges
  while ((CurrentTime - millis()) >5000) {
    GaugeTare() ;
    display.clearDisplay() ;
    display.setCursor(0,0) ;
    display.println("Lay your hand flat now") ;
    return ;
  }
  int IndexMin = scale1.read() ;
  int MiddleMin = scale2.read() ;
  int RingMin = scale3.read() ;
  int LittleMin = scale4.read() ;
  int ThumbMin = scale5.read() ;

  while ((CurrentTime - millis()) > 7500) {
    display.clearDisplay() ;
```

```
    display.setCursor(0,0) ;
    display.println("Now make a tight fist") ;
    display.display() ;
    return ;
  }
  int IndexMax = scale1.read() ;
  int MiddleMax = scale2.read() ;
  int RingMax = scale3.read() ;
  int LittleMax = scale4.read() ;
  int ThumbMax = scale5.read() ;

// Setting up each Digit's Max and Min values
  // Minimum Values
  int DigitMin[5] = {0,0,0,0,0} ;
  DigitMin[0] = IndexMin ;
  DigitMin[1] = MiddleMin ;
  DigitMin[2] = RingMin ;
  DigitMin[3] = LittleMin ;
  DigitMin[4] = ThumbMin ;
// Maximum Values
  int DigitMax[5] ;
  DigitMax[0] = IndexMax ;
  DigitMax[1] = MiddleMax ;
  DigitMax[2] = RingMax ;
  DigitMax[3] = LittleMax ;
  DigitMax[4] = ThumbMax ;

  if ((CurrentTime - millis()) > 15000) {
//Clearing the screen
    display.clearDisplay() ;
    display.setCursor(0,0) ;
```

```
    display.println("Setup Complete ") ;
   return ;
  }
}
void loop() {
 float gauge = 1 ;
 int i ;
 int j ;
 ReadScales() ;
 for (gauge = 1; gauge < 6; gauge++) {  // Read each gauge and store in array
  int AngleCalculation (gauge) ;
  }
//TESTING THAT THE CODE WORKS
 Serial.print("Raw Value [") ;
 for (i=1; i < 6; i++) {  //This Cycles through each gauge and stores the readings in the relevant array slot
  int ScaleRaw = DigitScaleRead[i] ;
  Serial.print(ScaleRaw) ;
  Serial.print(",") ;
 Serial.print( "]") ;
 Serial.println("Angle [") ;
  }
 for (j=1; j < 6; j++) {  // Cycles through each slot of the array and converts to an ange measurement
  int AngleRaw = DigitAngle[j] ;
  Serial.print(AngleRaw) ;
  Serial.print(",") ;
  Serial.print("]") ;
  delay(10) ;
    }
  //Code to setup ESP-NOW
  strcpy(GaugeData.a, DigitAngle{0,1,2,3,4} ;
  esp_err_t result = esp_now_sen(BroadcastAddress, (uint8_t *) &GaugeData, sizeof(GaugeData)) ;
}
```

8.3.2. Receiver:

```
#include <esp_now.h>
#include <WiFi.h>

// Structure to receive data via
// MUST MATCH the Sender Structure
typedef struct struct_message {
    int a ;
} struct_message;

struct_message GaugeData;

// callback function that will be executed when data is received
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
  memcpy(&GaugeData, incomingData, sizeof(GaugeData)) ;
  Serial.print("Information received: ") ;
  Serial.println(GaugeData.a) ;
}

void setup() {
// Letting the ESP-32 transmit the data to the computer
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);

  // Initialise ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  esp_now_register_recv_cb(esp_now_recv_cb_t(OnDataRecv));
}

void loop() {

}
```

8.3.3. Appendix D: Instruction manual

Instruction Manual for the Strain Gauge Finger Position Tracking Device

By Daniel Brittle

Contents

Introduction

Hands. Most people have them, and you use them every day, without even thinking.

Nowadays, everything is going digital. Banking, shopping, researching, hanging out with your mates.

VR has also become a much more common thing to see, from Youtubers playing VR Horror games on stream to the push for the Metaverse.

So why can't you use your hands in VR, and just have to be stuck with using the controllers that ship with whichever headset you're using?

This device aims to solve that issue, for cheaper than £100. And it'll teach you a few things about PCB design, Microcontrollers, the Internet-of-Things and how your hand itself works.

Let's get into it, shall we?

Parts List

This is the part where you're told you need to spend thousands on buying this exact part from this exact place, produced when the sun was setting on a blue moon.

Or we just don't do that.

This project was designed with the idea that people would have better ideas than I do, and cheaper ways of doing things. So, if you have better ideas than the ones here, don't be afraid to share – I'm always up for learning new things.

If you've got little technical know-how and just want a cheap hand tracker, these are the tools you'll want to make sure you have on hand:

- Soldering Iron
- Solder
- Wire cutters/strippers
- Micro-USB cable (Or USB-C, dependent on the ESP-32 you buy)

This is the expensive part - the stuff you need to buy:

Name	Amount	Link
ESP-32 DevkitC V4	2	Amazon
HX711 Amplifier Boards	5	AliExpress
BF350-30AA Strain Gauge	5	AliExpress
Dupont male to female connectors	6	AliExpress
0.96" OLED screen	1	AliExpress
Prototyping PCB board – 70x90mm	1	AliExpress
9V step-down transformer board to 5V	1	AliExpress
9V Battery connector	1	AliExpress
9V Battery	1	Amazon
M2 Brass standoffs – (Set)	-	Amazon
PCB Male & Female pin headers (Set)	-	AliExpress
26 AWG Wire (Assorted Colours) (Set)	-	Amazon
Switch	1	AliExpress
10mm strap	1M	AliExpress
Buckles	2	AliExpress

The links suggested will also be listed in the Appendices at the back, if you've printed this out for the fun of it. Can't say I'd blame you, there's nothing like the feeling of pen on paper when you're noting down ideas – I would recommend you keep a digital copy of this somewhere though, just in case you get too excited playing your game and the paper copy ends up soaked in coffee.

3D Printed Components

This is the part where you can make the device fit you, and your hand. Unfortunately, all of this was designed with my hand in mind, mostly because my hands were doing the work to make the thing. I'd recommend using some software like Ultimaker Cura (Or any slicer of your choice, if you know what a slicer is) to check the dimensions of each piece to make sure it's the right size for you – luckily enough, Cura is free to use.

If you have a 3D printer, you're in luck – most of the plastic parts needed are printable (Almost as if they were designed that way!).

If not, the good news is that there's hundreds of companies and people out there who are willing to print stuff for other people for a price, so you can check that they're the right size and send them over to one of these services. All things going well, you'll get the parts in the post in a few days.

Anyway, here's the list of parts you'll want to print to track one hand:

Part	Quantity
Finger Cap	4
Thumb Cap	1
Back of Hand	1
Thumb base	1
Hand Beam	5
ESP Wrist	1
ESP Middle	1
ESP Lid	1
Case Plugs	8

Code and how to upload

So. You've got the parts, you've got the box they go in, now you just need the instructions to make the parts work.

Good luck, you've got this.

If you don't want to write it all out yourself, I suppose I could be kind and tell you that the code you'll need is at the back of this manual, in the appendices. If you're holding a paper copy and reading this, I would highly recommend switching to a digital copy for the next bit – unless you don't like the copy and paste function and would rather type it all out yourself – each to their own, I suppose.

First things first, you're going to want to install the Arduino IDE – this lets your computer talk to the microcontroller, the brains of the device.

Once the IDE's installed, assuming you have the microcontroller in hand, just plug the microcontroller into your computer using the relevant wire – The IDE should automatically detect what it is and what COM port it uses.

If it doesn't automatically detect what device it is, click on the dropdown at the top and see if a COM port is detected to have something attached – if there's nothing, there's another step you need to do.

Don't worry, I had this issue as well and bought about 5 or 10 different ESP-32 boards before I figured it out – you just need to install a VCOM driver. There is tutorials out there to do this, I'll link the one I used here.

So, at this stage your controller is plugged into your computer, the IDE has a name and address to send the code when you paste it in – time for the next step.

If you've bought an ESP-32 board (like the one recommended in the parts list), you get to use a fancy feature of them called ESP-NOW. This sets up a P2P (Peer 2 Peer) network between the ESP's you've bought – problem is, they don't know what to call each other to get the other's attention.

This is where the MAC address of each comes in – It's a unique name every Wifi-enabled device has, and you just need to run the code from the appendices to find each ESP's name.

So, copy the Wifi code from the appendices (Credit goes to Rui & Sara Santos @ RandomNerdTutorials – the page link is here), paste it into the Arduino IDE and then click the blue circle with a tick at the top of the IDE window – This compiles the code and verifies that it'll work without blowing anything up (stuff shouldn't blow up, but be prepared just in case).

While you're waiting for the code to compile, I'll tell you a little about the Internet of things. The IoT (fancy acronym, I know) is the name for the network that all our wireless devices form around us, made of loads of things (See where the name comes from?) from a smart alarm clock (that tell you the weather for the day, so internet-enabled) to the Amazon Echo in your front room to one of those fancy smart fridges. Effectively, anything with a sensor of some sort that can connect to a network is part of the IoT.

With the IoT, all of the devices "talk" to each other and share their data, picking up on the smallest things ("The fridge has been opened 12 times today – 8:27:32am, XX/XX/XXXX") to things as large as reminding you to put your electric car on charge for work tomorrow. This sharing of data can help us, as people are predictable in large numbers (Think congestion on the road, there's no reason for it other than everyone slowing down because everyone else is). Talking of congestion, that's an example of how the IoT is used. Google maps (an example, other maps use the same data) tracks traffic and congestion through people's phones, as if a large number of people's phones are all within a short range of each other moving slowly on a road, it's more likely to be traffic than a horde of people walking in the middle of the road.

Anyway, the code should have compiled now, assuming that you're still awake after that little interlude.

The next step is to send the code to the microcontroller, so click on the blue circle with an arrow point upwards in. The code will compile again, and then you'll have a window pop up at the bottom of the screen telling you what's going on behind the scenes.

The MAC address of the microcontroller board you have plugged should start flooding your little pop-up window, so just unplug the microcontroller and take not of that MAC address before repeating the process with the other microcontroller.

Congratulations! You now know each of the microcontrollers' names.

As a bonus, you now know how to upload code to the microcontrollers, so copy & paste the tracking device code into the IDE and change the MAC address variable to the receiver, then compile & upload.

Repeat the same process for the receiver, except putting in the transmitter's MAC address where it asks for it.

And bam, you're done! At least for the coding bit, you've got the physical assembly yet.

Assembly Instructions

Now we're getting to the technical part: putting things together.

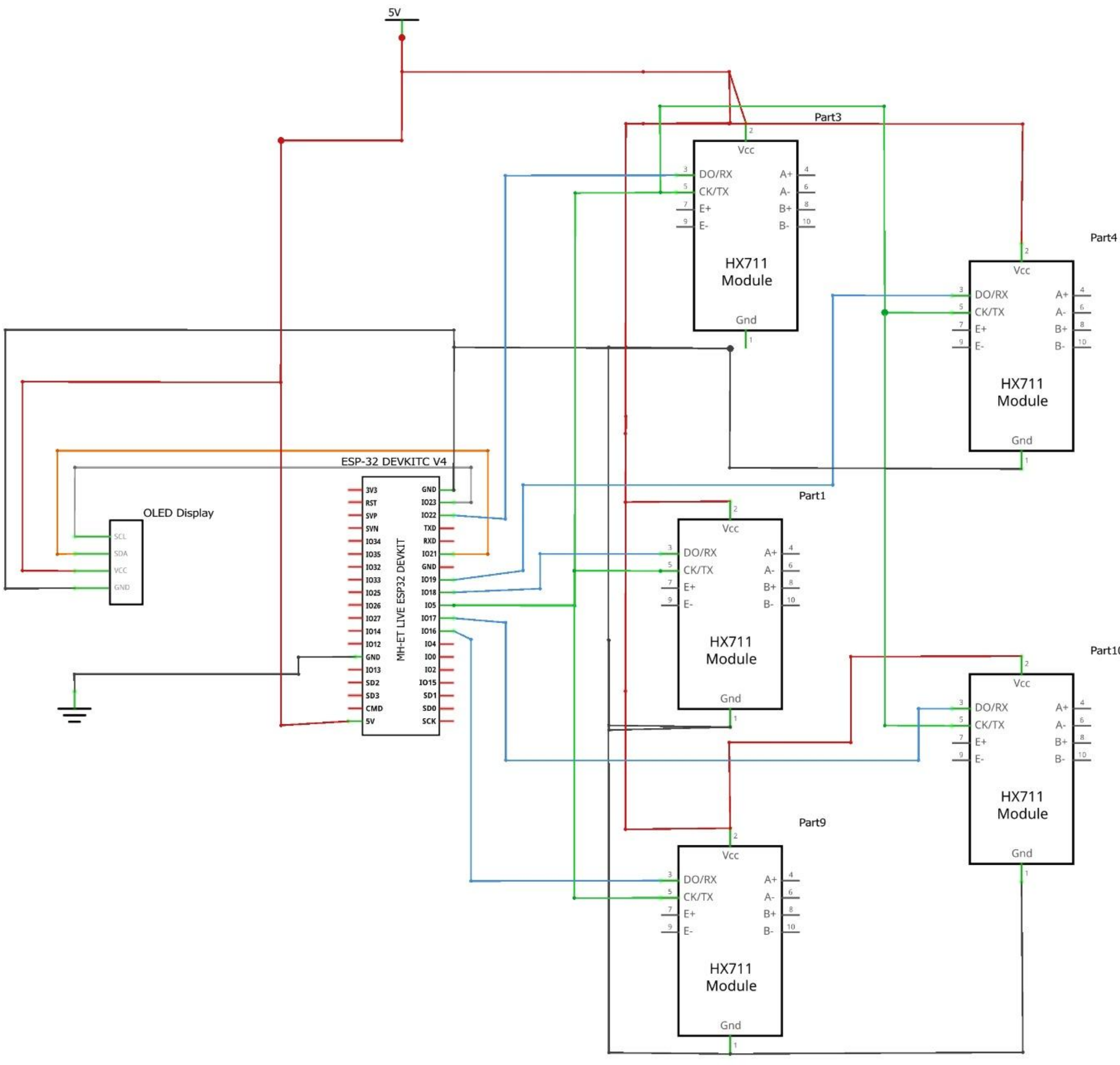
I've split this into two halves, as each of them use different tools to get the parts together. Plus, it makes this manual look a little more professional.

Electronics

This is the brains behind the digit tracking, and probably the most involved section because of the number of wires it involves.

I have tried to optimize the wiring, as you'll see as you read ahead and find out about the PCB, but this is the full bird's nest of wiring that you'll need to do.

Schematics:



The colours aren't just to make it look pretty (although it does help), they actually signify the different wires used in the device:

- Red is VCC (positive)

- Black is Ground (negative)
- Green is Clock (For HX711 boards)
- Blue is Data (each HX711 has their own individual one of these)
- Yellow is the I2C Data
- Grey is the I2C Clock

Even though the Schematic makes it look like a lot of wires, most of them are actually just linked between the boards themselves as opposed to making everything from the microcontroller to the peripherals individually.

To make the PCB, you'll use the prototyping board that was in the parts list, alongside the headers and 26 AWG wire that was bought.

To follow the PCB Schematics shown on the next two pages, my suggestion would be to define each pin type as a colour and follow one colour at a time to build up the PCB.

Trust me, I know from experience that trying to do it all at one time ends badly, with wires soldered in the wrong places and joints being made where there shouldn't be,

I would recommend following this order to make the PCB:

1. Female headers
2. Male headers
3. The top side
    a. VCC
    b. Ground
4. The underside
    a. Clock
    b. Data

The legend for the PCB schematics is shown here:

	Legend
	Female Socket
	Male Socket
A12	Connection Label
	Wire to Gauge
	Soldered Cannection

Remember, each wire needs to be joined to a pin and other wire via a solder joint between the two, rather than twisting the wires into a single hole.

You will also need to cut a power cable from the 9V in half and solder the switch in the middle before assembling the rest of the device.

Top

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
A																															
B	3V3										G										J15	V									
C	RST										23											C									
D	SVP										22											D									
E	SVN										TXD										L18	G									
F	34										RXD																				
G	35										21																				
H	32										G																				
I	33										19											V	B23								
J	35										18							E21				C									
K	26										5				V	C	D	G				D									
L	27										17				U18			H13				G	E23								
M	14										16																				
N	12										4																				
O	G	V8									0										I21	V									
P	1013										2											C									
Q	SD2										15											D									
R	SD3										SD1										L21	G									
S	CMD										SD0																				
T	5V										SCK				SA	SC															
U	X6														C	V	X8														
V															H12	G						V	O23								
W							G															C									
X							V															D									
Y																						G	R23								
Z																															

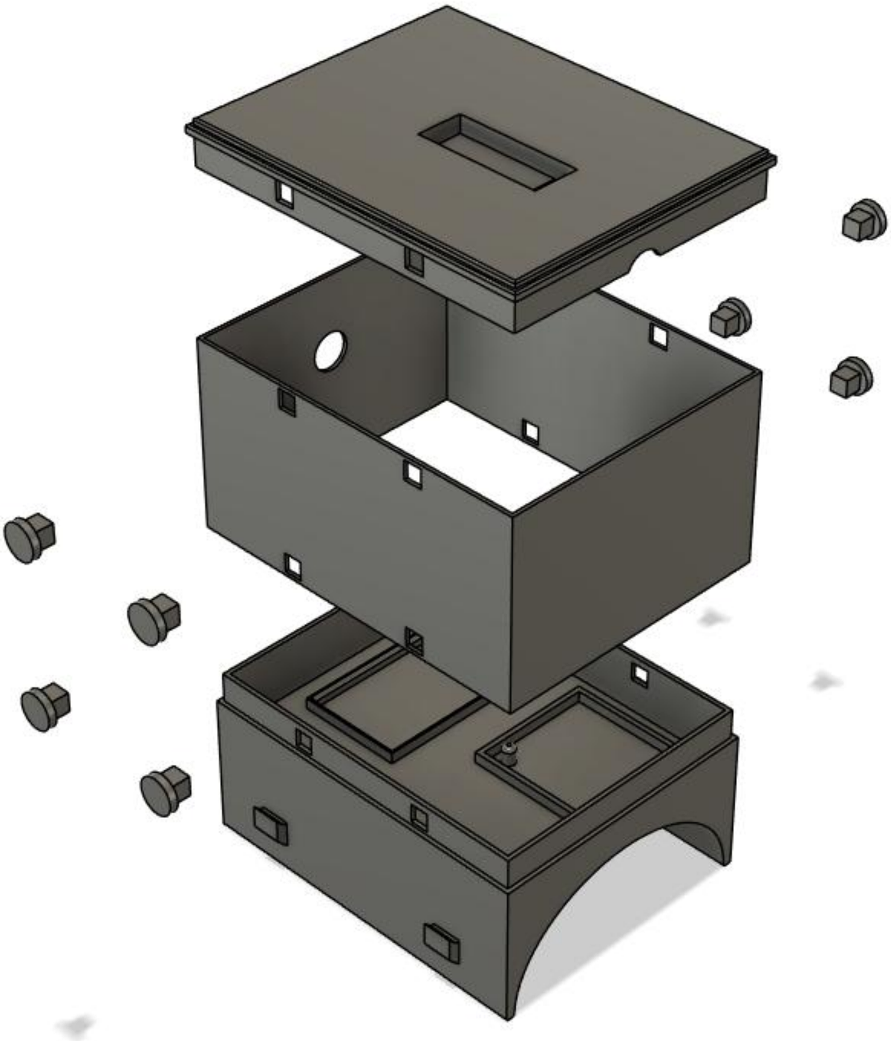
Underside

	31	30	29	28	27	26	##	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
A																															
B										V											G										3V3
C									J23	C											23										RST
D										D	I12										22										SVP
E										G											TXD										SVN
F																					RXD										34
G																					21										35
H																					G										32
I										V											19										33
J										C	P21					K12					18										35
K										D	L12			G	D	C	V				5										26
L										G					J12						17										27
M																					16										14
N																					4										12
O										V											0										G
P									W23	C											2										1013
Q										D	M12										15										SD2
R										G											SD1										SD3
S																D12	G12				SD0										CMD
T																SC	SA				SCK										5V
U																V	C	L16													
V										V						G															
W										C	U13														G						
X										D	N12														V						
Y										G																					
Z																															

Also, you will need to superglue each of the strain gauges carefully to the back if the hand tabs, one on each beam and put to one side to dry.

Main Body

The diagram below shows an exploded view of the 3D Printed wrist casing:

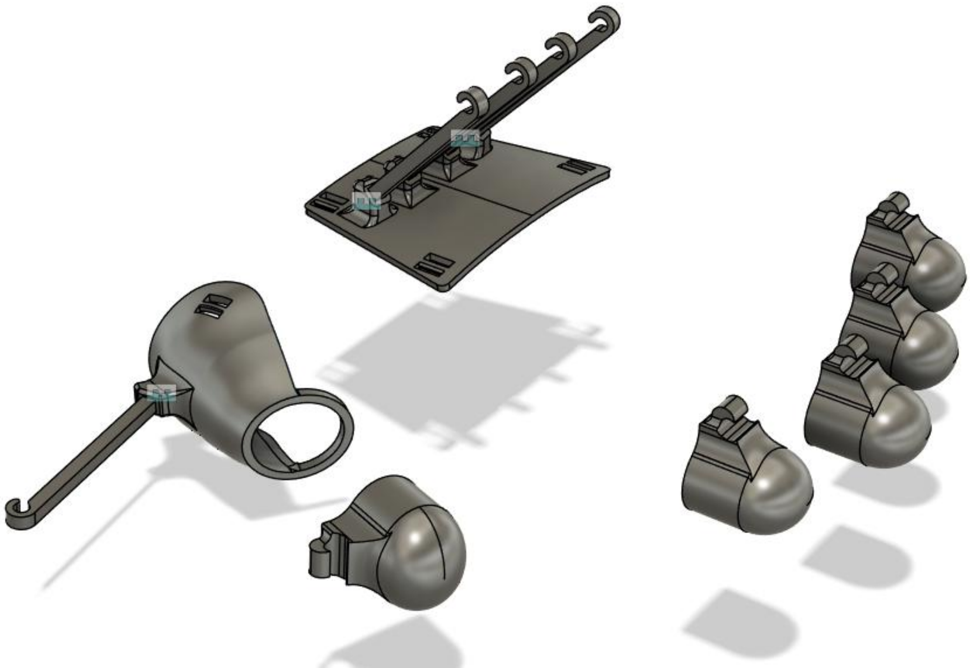


To put the electronics into the casing:

- The 9V battery goes into the larger frame on the wrist section, with the wire connector leading away from the wall
- The transformer board fits into the other frame, using the pin to help line it up - Don't forget to plug the connector in
- Slot the switch on the 9V connector wire into the space in the wall of the casing – this is the power switch to turn the device on and off
- Use 4 M2*20mm brass standoffs and screw into the small posts on the wrist section
- Use 2 jumper cables to connect the male pins on the underside of the PCB to the power board – Make sure the polarities are correct!
- Line up the PCB with the brass standoffs, and screw it in

- Slot in the microcontroller and 5 HX711 boards into the corresponding female headers
- Connect 4 jumper cables into the remaining female headers – these go to the screen
- To slot in the screen to the lid, first put the pins through the gap and keep the screen at a 45° degree angle as you slide it in, before lowering it so that it sites flush with the casing
- Put the wires soldered to the PCB to attach to the gauges through the hole in the wall of the middle of the casing, before sliding the middle casing over the PCB and onto the wrist box – the square holes around the edges will align between the two parts
- Using the 4 jumper cables you attached to the PCB earlier, connect the screen (check each wire is going to the correct pin) and slot the lid into place
- Slot each of the 8 case pins into the square holes around the edges of the middle casing, where the holes align with the wrist base and lid

Next, to assemble the back of the hand, as shown in the image below:



To assemble the back of the hand, the hand beams simply slot into the gaps on the panels, and the fingertip caps simply go on each finger.

To attach the panels together, the 10mm webbing should be woven between the slots on each panel, and once the right length is achieved, a buckle should be attached to make the device wearable.

Recommended Links

Expressif (ESP-32 manufacturer) website:

https://www.espressif.com/en/support/documents/technical-documents

Arduino Forums:

https://forum.arduino.cc/

RandomNerdTutorials:

https://randomnerdtutorials.com/

Appendices

Appendix 1: Parts List links

ESP-32 DevkitC V4 – x2

https://www.amazon.co.uk/dp/B074RGW2VQ?ref=ppx_yo2ov_dt_b_fed_asin_title&th=1

HX711 Amplifier Boards - x5

https://www.aliexpress.com/item/1005006883960671.html?spm=a2g0o.order_list.order_list_main.100.3b5c1802x9CcbM

BF350-30AA Strain Gauge – x5

https://www.aliexpress.com/item/1005001891511256.html?spm=a2g0o.order_list.order_list_main.4.449f1802LgX1me

Dupont male to female connectors – x6

https://www.aliexpress.com/item/1005003250665155.html?spm=a2g0o.productlist.main.7.64b274c4KWZ8Hf&aem_p4p_detail=202505012004024896266745130000004445320&algo_pvid=93325d74-a303-4aa3-ac1f-c485979acd84&algo_exp_id=93325d74-a303-4aa3-ac1f-c485979acd84-3&pdp_ext_f=%7B%22order%22%3A%22436%22%2C%22eval%22%3A%221%22%7D&pdp_npi=4%40dis%21GBP%213.08%213.08%21%21%213.97%213.97%21%402103856417461550420598480ef12c%2112000028309478180%21sea%21UK%212496246729%21X&curPageLogUid=bMVneRItB382&utparam-url=scene%3Asearch%7Cquery_from%3A&search_p4p_id=2025050120040248962667451300004445320_1

0.96" OLED screen – x1

https://www.aliexpress.com/item/1005006701323078.html?spm=a2g0o.order_list.order_list_main.48.449f1802LgX1me

Prototyping PCB board – 70x90mm – x1

https://www.aliexpress.com/item/1005007252022845.html?spm=a2g0o.productlist.main.33.41181df8gCCBop&algo_pvid=c4b03d42-74ab-419c-9788-696b91f59cc0&algo_exp_id=c4b03d42-74ab-419c-9788-696b91f59cc0-16&pdp_ext_f=%7B%22order%22%3A%221%22%2C%22eval%22%3A%221%22%7D&pdp_npi=4%40dis%21GBP%210.64%210.64%21%21%216.01%216.01%21%40211b6139174615488252 67978ee9c4%2112000039953644023%21sea%21UK%212496246729%21X&curPageLogUid=c0aZgalEnZ1x&utparam-url=scene%3Asearch%7Cquery_from%3A

9V step-down transformer board to 5V – x1

https://www.aliexpress.com/item/1005005691344779.html?spm=a2g0o.order_list.order_list_main.63.449f1802jOxMmJ

9V Battery connector – x1

https://www.aliexpress.com/item/1097854909.html?spm=a2g0o.productlist.main.1.2fe71eWs1eWslh&algo_pvid=5b695743-bad3-4bd0-b120-30bcbdb05aed&algo_exp_id=5b695743-bad3-

4bd0-b120-30bcbdb05aed-0&pdp_ext_f=%7B%22order%22%3A%22345%22%2C%22eval%22%3A%221%22%7D&pdp_npi=4%40dis%21GBP%210.60%210.58%21%21%210.77%210.74%21%40211b430817461546624641689e386c%2112000034582752228%21sea%21UK%212496246729%21X&curPageLogUid=Yap0PhkA08UR&utparam-url=scene%3Asearch%7Cquery_from%3A

9V Battery – x1

https://www.amazon.co.uk/Morrisons-Extra-Long-Alkaline-Battery/dp/B01LZNKYB1/ref=sr_1_2_f3_0o_morri?crid=1CEUZDDVVZVR3&dib=eyJ2IjoiMSJ9.nfzjFrmX4uHkv-u9hGdwVUQr7n_vCkOFUJJmgU3ppMgwx8YQqH3L_nHsLk3X7oSGo9ZteRhZJI4B1jo2st4bAEXqXGkDg-S6LILfByiIl-6nyAjtiI2fHZSZZIisb3F7gix7t-YYOyy9SsXxSu3F5vFE7wL_kmtfTL9PTWADIrypMeDQS3E4AecwCWR3iJ97Rkpvi_W2Se6HzbNJKvOEf-eibnihx0C2v_TgkyuazZk.-8e2uiSYKBMCdUaejxNm-09Zo3eSTW6kHMfPYzATLF4&dib_tag=se&keywords=9V+battery&qid=1746197460&refinements=p_85%3A20930949031&rnid=20930948031&rps=1&sprefix=9v+battery%2Caps%2C194&sr=8-2

M2 Brass standoffs – (Set)

https://www.amazon.co.uk/dp/B0C1BZ1MKJ?ref=ppx_yo2ov_dt_b_fed_asin_title

PCB Male & Female pin headers (Set)

https://www.aliexpress.com/item/4000873858801.html?spm=a2g0o.productlist.main.1.2b726cc7hlUpKK&algo_pvid=1ac4ebc9-b4b5-42ea-b8a3-7a86a931c1a4&algo_exp_id=1ac4ebc9-b4b5-42ea-b8a3-7a86a931c1a4-0&pdp_ext_f=%7B%22order%22%3A%223850%22%2C%22eval%22%3A%221%22%7D&pdp_npi=4%40dis%21GBP%211.47%211.47%21%21%211.90%211.90%21%40211b876717461549956492020eeb76%2110000010058190554%21sea%21UK%212496246729%21X&curPageLogUid=PzLeXhQc3llm&utparam-url=scene%3Asearch%7Cquery_from%3A

26 AWG Wire (Assorted Colours) (Set)

https://www.amazon.co.uk/dp/B0C7TKK79B?ref=ppx_yo2ov_dt_b_fed_asin_title&th=1

Switch – x1

https://www.aliexpress.com/item/1005007077815408.html?spm=a2g0o.productlist.main.5.54e9331b2rFXex&algo_pvid=099ae16d-2c71-403f-b049-4684505c31c2&algo_exp_id=099ae16d-2c71-403f-b049-4684505c31c2-2&pdp_ext_f=%7B%22order%22%3A%221453%22%2C%22eval%22%3A%221%22%7D&pdp_npi=4%40dis%21GBP%210.61%210.61%21%21%215.75%215.75%21%40210385db17461659027263463e09de%2112000039325827634%21sea%21UK%212496246729%21X&curPageLogUid=BuwroHm6Bnmm&utparam-url=scene%3Asearch%7Cquery_from%3A

10mm Nylon Webbing – 1M

https://www.aliexpress.com/item/1005005221573556.html?spm=a2g0o.productlist.main.5.6ab86b35SWesBm&algo_pvid=659cf80d-e4d8-4618-ae1e-8c3e818bdeb0&algo_exp_id=659cf80d-e4d8-4618-ae1e-8c3e818bdeb0-2&pdp_ext_f=%7B%22order%22%3A%221636%22%2C%22eval%22%3A%221%22%7D&pdp_npi=4%40dis%21GBP%211.59%211.59%21%21%212.05%212.05%21%4021038e6617461793

065783642eebd6%2112000032242477268%21sea%21UK%212496246729%21X&curPageLogUid=HPaSR0NxuxaG&utparam-url=scene%3Asearch%7Cquery_from%3A

Buckles – x2

https://www.aliexpress.com/item/32963823617.html?spm=a2g0o.productlist.main.3.324670e4W79Ph5&algo_pvid=6dff46f0-79f2-461e-8bcb-37ed4bcdc4b8&algo_exp_id=6dff46f0-79f2-461e-8bcb-37ed4bcdc4b8-1&pdp_ext_f=%7B%22order%22%3A%22187%22%2C%22eval%22%3A%221%22%7D&pdp_npi=4%40dis%21GBP%211.05%211.05%21%21%211.36%211.36%21%40211b430817461794232623792e38ab%2166559844577%21sea%21UK%212496246729%21X&curPageLogUid=rlAlMHdZXA4w&utparam-url=scene%3Asearch%7Cquery_from%3A

Appendix 2: ESP-32 Code

Wifi – MAC Addresses;

This can be found at the link below, alongside a corresponding guide:

https://randomnerdtutorials.com/esp-now-esp32-arduino-ide/

Strain Gauge Tracking Code (Wrist device):

```
// Including all the necessary Libraries:
#include <HX711.h>
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <esp_now.h>
#include <WiFi.h>
// Setting up the display screen
#define SCREEN_WIDTH 128 // Display width (Pixels)
#define SCREEN_HEIGHT 32 // Display height (Pixels)
#define OLED_RESET -1 // Reset pin
#define SCREEN_ADDRESS 0x3C // I2C Screen address
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET) ;

// Setting up ESP-NOW
//Put the Reciever MAC address here:
uint8_t broadcastAddress[] = {0xf8, 0xb3, 0xb7, 0x48, 0xbd, 0x1c} ;
// This part defines what sort of data is being sent
typedef struct struct_message {
  int a ;
} struct_message ;
// defining the message name to be sent via ESP-NOW
struct_message GaugeData ;
esp_now_peer_info_t peerInfo ;

// Setting up HX711 gauge names and Pins
int Index = 22 ;
int Middle = 19 ;
int Clock = 5;
int Ring = 18 ;
int Little = 17 ;
int Thumb = 16 ;

// Other Global Variables to be Used
float LocalRatio ;
```

```
int Thumb = 12 ;

// Other Global Variables to be Used
float LocalRatio ;
float AngleRatio ;
int X = 0 ;
float DigitValue ;

// Setting up remaining stuff for the HX711 gauges
HX711 scale1 ; HX711 scale2 ; HX711 scale3 ; HX711 scale4 ; HX711 scale5 ;

// Setting up a generalised conversion function for the gauges' raw values to be converted to angle measurements
int DigitMin[5] = {0,0,0,0,0} ;
int DigitMax[5] = {20000,20000,20000,20000,20000} ;
int DigitScaleRead[5] = {0,0,0,0,0} ;

// Setting up a tare function for all gauges simultaneously
int GaugeTare() {
  scale1.tare() ; scale2.tare() ; scale3.tare() ; scale4.tare() ; scale5.tare() ;
}
// Setting up a function to read all gauges at once
int ReadScales() {
 DigitScaleRead[0] = scale1.read() ;
 DigitScaleRead[1] = scale2.read() ;
 DigitScaleRead[2] = scale3.read() ;
 DigitScaleRead[3] = scale4.read() ;
 DigitScaleRead[4] = scale5.read() ;
}
// Setting up Blank Angle Array
int DigitAngle[5] = {0,0,0,0,0} ;
```

```

// Setting up Angle Calculation Function
int AngleCalculation(int X) {
  int Max = DigitMax[X] ;
  int Min = DigitMin[X] ;
  int DigitValue = DigitScaleRead[X] ;
  LocalRatio = Max - Min ;
  AngleRatio = DigitValue / LocalRatio ;
  DigitAngle[X] = AngleRatio * 90 ;
  LocalRatio = 0 ;
  AngleRatio = 0 ;
  DigitValue = 0 ;
  return DigitAngle[X] ;
}

void setup() {
  // put your setup code here, to run once:
  Serial.begin(19200) ;
  display.begin(SSD1306_SWITCHCAPVCC, SCREEN_ADDRESS) ;
//Setting up ESP-NOW
  WiFi.mode(WIFI_STA) ;
  if (esp_now_init() != ESP_OK) {
    Serial.println("Problem encountered starting ESP-Now") ;
    return ;
  }
//Remembering the P2P reciever
  memcpy(peerInfo.peer_addr, broadcastAddress, 6) ;
  peerInfo.channel = 0 ;
  peerInfo.encrypt = false ;
  if (esp_now_add_peer(&peerInfo) != ESP_OK) {
    Serial.println("Can't add peer") ;
    return ;
```

```
  }
//Initialising the scale setting for each pin & Clock
  scale1.begin(Index, Clock) ;
  scale2.begin(Middle, Clock) ;
  scale3.begin(Ring, Clock) ;
  scale4.begin(Little, Clock) ;
  scale5.begin(Thumb, Clock) ;
// OLED screen initialization
  display.clearDisplay() ;
  display.setTextSize(1) ;
  display.setTextColor(SSD1306_WHITE) ;
  display.display() ;
  display.setCursor(0,0) ;
  display.println("Initializing....") ;
  int CurrentTime = millis() ;
//Calibrating the Gauges
  while ((CurrentTime - millis()) >5000) {
    GaugeTare() ;
    display.clearDisplay() ;
    display.setCursor(0,0) ;
    display.println("Lay your hand flat now") ;
    return ;
  }
  int IndexMin = scale1.read() ;
  int MiddleMin = scale2.read() ;
  int RingMin = scale3.read() ;
  int LittleMin = scale4.read() ;
  int ThumbMin = scale5.read() ;

  while ((CurrentTime - millis()) > 7500) {
    display.clearDisplay() ;
```

```
    display.setCursor(0,0) ;
    display.println("Now make a tight fist") ;
    display.display() ;
    return ;
  }
  int IndexMax = scale1.read() ;
  int MiddleMax = scale2.read() ;
  int RingMax = scale3.read() ;
  int LittleMax = scale4.read() ;
  int ThumbMax = scale5.read() ;

// Setting up each Digit's Max and Min values
  // Minimum Values
  int DigitMin[5] = {0,0,0,0,0} ;
  DigitMin[0] = IndexMin ;
  DigitMin[1] = MiddleMin ;
  DigitMin[2] = RingMin ;
  DigitMin[3] = LittleMin ;
  DigitMin[4] = ThumbMin ;
// Maximum Values
  int DigitMax[5] ;
  DigitMax[0] = IndexMax ;
  DigitMax[1] = MiddleMax ;
  DigitMax[2] = RingMax ;
  DigitMax[3] = LittleMax ;
  DigitMax[4] = ThumbMax ;

  if ((CurrentTime - millis()) > 15000) {
//Clearing the screen
    display.clearDisplay() ;
    display.setCursor(0,0) ;
```

```
    display.println("Setup Complete ") ;
   return ;
  }
}
void loop() {
 float gauge = 1 ;
 int i ;
 int j ;
 ReadScales() ;
 for (gauge = 1; gauge < 6; gauge++) {  // Read each gauge and store in array
  int AngleCalculation (gauge) ;
  }
//TESTING THAT THE CODE WORKS
 Serial.print("Raw Value [") ;
 for (i=1; i < 6; i++) {  //This Cycles through each gauge and stores the readings in the relevant array slot
  int ScaleRaw = DigitScaleRead[i] ;
  Serial.print(ScaleRaw) ;
  Serial.print(",") ;
 Serial.print( "]") ;
 Serial.println("Angle [") ;
  }
 for (j=1; j < 6; j++) {  // Cycles through each slot of the array and converts to an ange measurement
  int AngleRaw = DigitAngle[j] ;
  Serial.print(AngleRaw) ;
  Serial.print(",") ;
  Serial.print("]") ;
  delay(10) ;
    }
  //Code to setup ESP-NOW
  strcpy(GaugeData.a, DigitAngle{0,1,2,3,4} ;
  esp_err_t result = esp_now_sen(BroadcastAddress, (uint8_t *) &GaugeData, sizeof(GaugeData)) ;
}
```

Receiver:

```
#include <esp_now.h>
#include <WiFi.h>

// Structure to receive data via
// MUST MATCH the Sender Structure
typedef struct struct_message {
    int a ;
} struct_message;

struct_message GaugeData;

// callback function that will be executed when data is received
void OnDataRecv(const uint8_t * mac, const uint8_t *incomingData, int len) {
  memcpy(&GaugeData, incomingData, sizeof(GaugeData)) ;
  Serial.print("Information received: ") ;
  Serial.println(GaugeData.a) ;
}

void setup() {
// Letting the ESP-32 transmit the data to the computer
  Serial.begin(115200);
  WiFi.mode(WIFI_STA);

  // Initialise ESP-NOW
  if (esp_now_init() != ESP_OK) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }

  esp_now_register_recv_cb(esp_now_recv_cb_t(OnDataRecv));
}

void loop() {

}
```

Appendix 3: Device Pinouts

ESP-32 DevkitC V4:

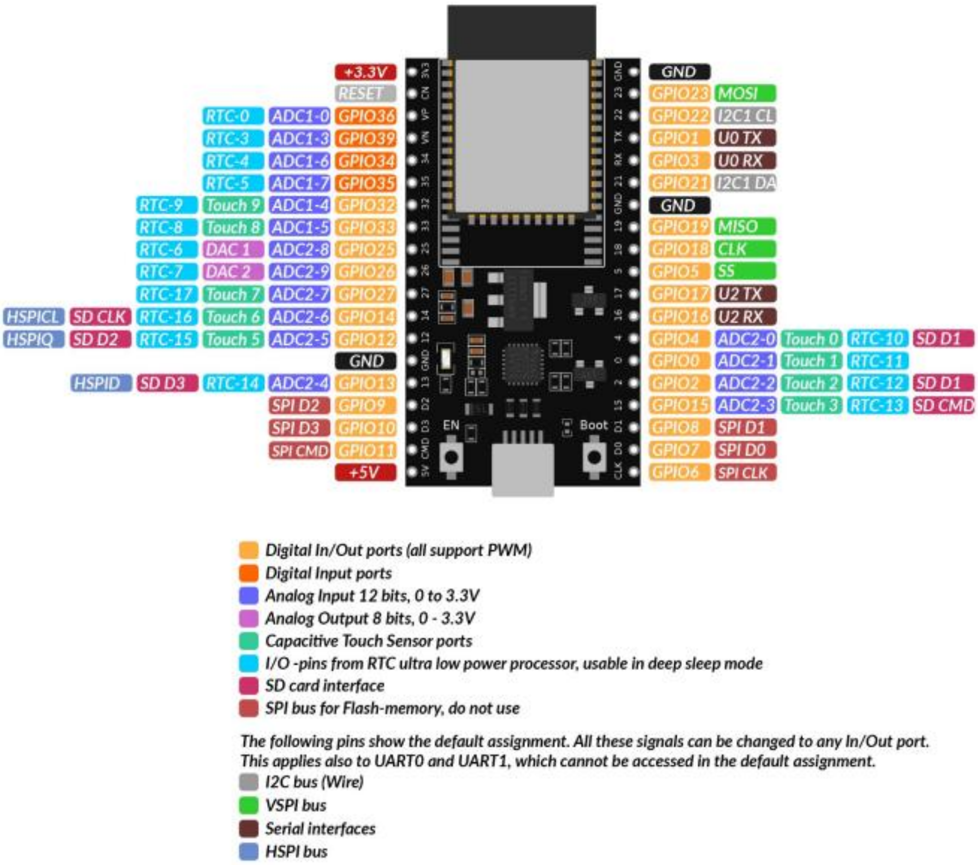


Figure 14: ESP-32 DevkitC V4 Pinout

Further information available at this Link.

HX711 Amplifier Board Pinout:

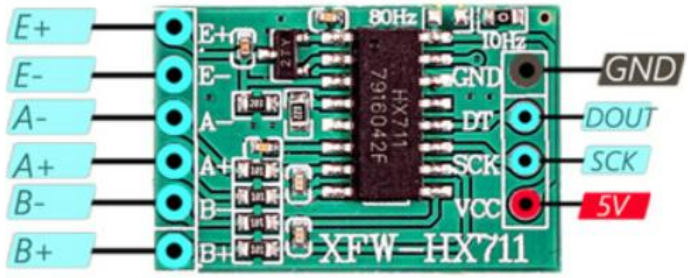


Figure 15: HX711 Amplifier Board Pinout

The full datasheet is available at this [Link.]()