The Creation of Novostok, a Handheld Gaming Console

Naomi E Silver
ID: 2425281
AC40001 Honours Project
BSc (Hons) Applied Computing
University of Dundee, 2025
Supervisor: Prof. G Webster

Abstract – *Novostok is a do-it-yourself console that is reviving classic games while showing how modern and low-cost hardware can underpin grassroots preservation. Built around a Raspberry Pi 5, a Raspberry Pi Pico 2 microcontroller and an entirely 3D printed enclosure, the prototype delivers reliable emulation from 8-bit systems up to the PlayStation 2. The project defined strict functional requirements of full digital and analogue input, hot swappable ROM management and non-functional requirements such as low input-latency, thermal stability and reproducibility from readily available parts. With the Pi 5 selected for its balance of performance and community support, while CircuitPython driven USB HID firmware on the Pico ensure sub-millisecond input delivery. A two-stage evaluation combined quantitative ergonomic testing with a five-person user study, confirming the Novostok is comfortable for extended play and intuitive to operate, despite its compact footprint. The paper analyses hurdles in dual-display support and joystick calibration, and documents hardware workarounds to guide future builders. Novostok demonstrates that enthusiast-built hardware can provide sustainable access to ageing software libraries and offers a reproducible blueprint for hobbyists and preservationists who want to bridge the gap between yesterday's games and today's open platforms.*

1 Introduction

Video game history is vanishing faster than it can be rereleased. Official storefront closures, aggressive litigation and the slow decay of cartridges mean that whole generations of titles are sliding into "lost media" status. Meanwhile, demand for authentic retro experiences is booming, with dedicated devices such as the Analogue Pocket faithfully recreating 8-bit handhelds, and premium "PC handhelds" like the Steam Deck chase modern performance. Yet both extremes miss a crucial middle ground. An affordable, hackable console that is small enough to live in a backpack, powerful enough to emulate sixth generation home console hardware and open enough to be rebuilt or repaired long after corporate support evaporates.

Novostok tackles that gap. The project began with a single question. Can one person, using off the shelf components and open-source tools, build a handheld that feels as cohesive as a commercial product while prioritising preservation and reproducibility? Answering it required merging disciplines that rarely meet in a single assignment. Embedded systems design, 3D CAD and additive manufacturing, low-level firmware development, Linux system administration and human computer interaction testing.

This paper documents that journey. Section 2 surveys the technological and cultural backdrop, from Nintendo's dual-display innovations to contemporary DIY emulation projects. Section 3 creates the functional and non-functional requirements and precures them into a specification to follow. Section 4 details the hardware and software design from informed decisions based on the specification. Enclosure fabrication and post processing, soldering and implementation of firmware is covered in section 5. Section 6 presents the quantitative ergonomics and qualitive user study results, while section 7 reflects on the finished device's capabilities and limitations. Finally, section 8-10 discuss lessons learned, guidance for future builders and outline the upgrades for a future revision.

In short, Novostok is both a functioning and a proof of concept: a template for how hobbyists can safeguard digital heritage through hands-on engineering rather than apathetic nostalgia.

2 Background

2.1 Retro handheld consoles

Video game consoles are categorised in a particular way, through 'generations', with the fourth generation being the first to see a handheld console achieve commercial success by selling over one million units. This generation included some less popular entries from Atari's Lynx [13] and Sega's Game Gear [14] which both saw minor success. In contrast, Nintendo's introduction of the Game Boy [15] becoming an enormous commercial success, achieving lifetime sales of approximately 64.42 million units globally [1]. However, this figure includes the consolidated sales with the Game Boy Color [16]. By subtracting the initial year of Game Boy Color sales, it can be estimated that the original Game Boy (and its variations) accounted for most of these sales during its lifetime. In contrast to Atari's and Sega's consoles, the

Game Boy did not have a colour display and relied on less powerful hardware. Despite these limitations, its long battery life and superior video games library allowed it to out-sell the other consoles within fourth console generation, this would cement Nintendo's lead within the handheld console market. Following the success of the original Game Boy, Nintendo developed the Game Boy Color and released in 1998. This console included a colour display and more powerful hardware [17] and while not as successful as the console that it followed, it still sold at least 54.3 million units [1].

While it's important to highlight the consoles that defined their respective era, the focus and main inspiration for this project lies in the 7th and 8th generations of consoles with the Nintendo DS [18] and 3DS [19] line of consoles. These consoles introduced dual displays, with the bottom display featuring a resistive touchscreen, innovating significantly on other consoles within these generations [20]. This offered unique mechanics that incorporated stylus-based controls and gesture input, such as those within the game Legend of Zelda: Phantom Hourglass [21]. This entry incorporates the use of a stylus for the resistive touchscreen to display a map of the environment and allowing the user to plot points within the map. This approach to handheld gaming further solidified the DS as a bridge between console generations, allowing players to experience a vast library of games originating from the late 80's through backwards compatibility. In addition to Nintendo's unique take on handheld consoles within this generation, they pioneered features such as local wireless multiplayer allowing users to trade Pokémon [22] to racing other users on Mario Kart [23] to communicating using PictoChat [20]. These innovations made the DS line of consoles the most versatile and influential handheld system of its time, shaping the future of portable gaming and inspiring subsequent designs, including this project.

2.2 Modern commercial devices

The appeal of retro style gaming consoles continues well beyond the late 1900s, with many modern devices aiming to recreate the feel and experience of early handheld systems. Dedicated communities have formed around the preservation and modernisation of these experiences, often through new hardware. One of these products is the Analogue Pocket [24], a device that imitates the appearance of the original Game Boy but with a modernised approach. While it follows the same design language as the Game Boy, the Analogue Pocket improved on some of the criticisms of the original device through a full colour and backlit display [24]. As opposed to other similar devices in the space, the Analogue Pocket is intended only for use with original game cartridges from the Game Boy line of consoles and other generation four consoles and not as a device to facilitate piracy [24]. It achieves this using an FPGA (field programmable gate array) which replicates the original hardware at a lower level, unlike conventional emulators which replicate through software. The use of an FPGA in the Analogue Pocket provides many advantages such as the accuracy and timing of the emulation and modularity which allows the Pocket to replicate more than a single console [2]. However, while these advantages provide a precise recreation, there are two limitations preventing their use in other devices and this project. The cost of an FPGA is significantly higher than a conventional processor and the complexity of programming the FPGA to replicate the hardware requires a deep understanding of the original system's design [2]. While the Analogue Pocket focusses on faithfully recreating the retro gaming experience, other modern devices are exploring how dual-screen designs can push technology forward.

The innovation of dual displays is not a feature that often gets added to modern devices. A recent example aimed at gaming is the Ayaneo Flip DS [25], a device that pays homage to the Nintendo DS in more than just name and is positioning itself a handheld Windows PC competing within the same market as the Valve Steam Deck [26]. Unlike the Analogue Pocket which focusses on replicating retro gaming experiences, the Ayaneo Flip DS targets modern PC gaming but has been faced with a mixed reception. According to PC Gamer [3], the device's short battery life of less than two hours under load, the small lower display measuring only 3.5 inches, the noticeable gap between the top and bottom displays, and its prohibitively high starting price of $949, make it a niche product for enthusiasts rather than a practical gaming solution.

This reinvention of the wheel does highlight a trend within the tech industry back towards multiple displays and clamshell designs on technology. With devices from Samsung like the Z Flip [27] and the Z Fold [28], both of which are devices that incorporate the clamshell design to present the user with more screen real estate within a smaller footprint. Similarly, devices like the Microsoft Surface Duo line of products aimed at increasing productivity through giving users more options when it comes to workflow. While devices like the aforementioned may be marketed as innovative and new, the tech industry has seen this trend before with mobile phones like the Motorola Razr which prioritised compactness over the bulkier "brick" phones of their time. These modern devices, however, expand on that legacy by offering practical solutions that address contemporary demands, allowing users the luxury of a larger display without sacrificing portability.

2.3 Modern DIY and open-source devices

Given the flexibility of modern single board computers and system on module boards, enthusiasts have created their own handheld emulation devices. While none, as of writing, have targeted Nintendo DS emulation, many exemplify the viability of low-cost hardware with open-source software for retro console emulation.

One example is the Pico-GB [48], a handheld emulation device based on the Raspberry Pi Pico [53]. This project recreates the original Game Boy form factor and is capable of emulating Game Boy games through bare metal programming, without a traditional operating system. The Pico-GB, is constrained by the Pico's limited memory and processing power, making it suitable only for simpler 8-bit consoles. However, it perfectly demonstrates the strength of DIY. It is low-cost, community supported with the underlying code being maintained and modular construction using widely available components. It also demonstrates the potential of microcontrollers as valid platforms for retro emulation, albeit within specific technical boundaries.

Microcontroller-based devices such as the Pico-GB demonstrate the feasibility of extremely low-cost handheld emulation, but they are inherently limited by the SoC's computing power. For emulating more demanding platforms, particularly generation six and earlier seventh generation consoles, many DIY projects utilise the Raspberry Pi Zero [54].

For Pi Zero-based handheld emulation, the Null2 [49] uses a Raspberry Pi Zero to run lightweight emulation frameworks such RetroPie [54]. As an open-source project, every aspect of the device is freely available. The enclosure design files are available to be either, laser cut, 3D print, or CNC machine by the end user or purchased through occasional sales. Similarly, the hardware is provided as a part list and can be purchased individually or bought just like the enclosure. While the performance of the Pi Zero limits emulation of later generation consoles, the Null2 is designed in such a way that newer Pi Zero models can be used as a drop-in replacement, namely the Pi Zero 2, provided the software has been updated to support the newer hardware.

Devices like the Pico-GB and Null2 showcase the accessibility and creativity of the DIY and open-source emulation community, but their limited SOCs (system on chip) prevent the emulation of demanding consoles. This is where the most powerful of the Raspberry Pi Foundation's hardware becomes useful. The higher performance SBCs (single board computer), such as the Raspberry Pi 4 and 5, and the SoM (system on module) offerings like the compute module 4 [57] and 5 [56], provide the computational resources required for more advanced emulation tasks.

The full-size Raspberry Pi boards are more commonly used in stationary, home console-style builds rather than handheld projects. Their larger footprint allows for improved thermal management and easy access to standard IO like HDMI, USB and ethernet. As a result, there are comparatively few open-source projects targeting these boards. Instead, the ecosystem is populated by commercial enclosures and kits that repurpose the Pi as a plug-and-play retro console. For example, the PiStation [50], an enclosure designed to mimic the appearance of classic home consoles while housing a Raspberry Pi 4.

In projects and products that prioritise portability, a compute module is commonly used. The Raspberry Pi Foundation's Compute Module 4 and 5 match their SBC counterparts in performance and IO capabilities, however, they need to be placed in a carrier board. These boards break out the module's features such as display links and USB, but unlike the Raspberry Pi which has fixed IO, these carrier boards can be made to target specific features like PCIe and GPIO only. This means that the module can consume less power as it is not supporting unnecessary features, have an overall smaller footprint and easier reproducibility.

An open-source example that utilises the compute module 4 is the Retro Lite CM4 [51], a project designed to replicate the form factor of the Nintendo Switch Lite. The device integrates a custom carrier board which provides USB, HDMI, dual display interfaces, integrated controls, audio and power management. This implementation allows for far more control over the size of the device and is not constrained by the height of full-sized single board computers meaning an overall slimmer device can be achieved with the performance of an SBC. With the use of a carrier board, hardware upgrades are as easy as removing the old compute module and putting in another one like a Pi Zero or Pico could in the previously mentioned projects.

2.4 Emulation, the ethicality and legality

While emulation itself is not illegal, distribution or downloading of copyrighted ROMs typically is. This distinction was set in 2000 where Sony, following the launch of the PlayStation 1 [29], attempted to sue the corporation Connectix over their emulator the Virtual Game Station [30]. The court deemed the creation and sale of the emulator fair use and allowed Connectix to continue selling the emulator, until Sony bought the rights to it and subsequently discontinued its sale [5]. This case established a legal precedent for the legitimacy of emulators, provided they are developed without infringing on proprietary software or distributing copyrighted ROMs.

However, the use of third-party tools, such as ROM dumpers [31], complicates this ethical and legal landscape. They allow users to plug in a proprietary cartridge like those from the Nintendo DS and extract the data, allowing it to be playable on emulators. End users who do this are not often targeted by companies like Nintendo but rather communities who provide these ROMs for anyone to download, like in the case of Nintendo of America v. Gary Bowser, where Gary was found guilty of fraud over his connection with Team Xecuter, a group responsible for hacking Nintendo consoles and distributing ROMs. He was ordered to pay a non-dischargeable debt of $10,000,000 [4]. This case has set a significant precedent as the fine Gary

was issued in perpetuity and a provision of 25-30% of his gross monthly income until it is paid off.

Although the legality of Gary Bowser's actions are indisputable, Nintendo has a reputation for heavy-handed litigation. And as such, despite emulators being deemed legal and fair use, Nintendo targeted Yuzu, a popular emulator for the Nintendo Switch, ordering the group Tropic Haze to pay a total of $2,400,000 [6]. Following this, in October 2024 Nintendo contacted the project lead of Ryujinx, another Switch emulator, where Nintendo offered an agreement to stop working on the project and to remove the organisation and all related assets out of fear of a similar outcome [32].

Despite this, emulation and piracy will continue to happen as they often go hand in hand. Piracy is often cited as a service issue and not a cost issue [33] and as Nintendo continues to close online marketplaces as a legitimate source of these games [34], users must choose between purchasing on the secondary market at a steep price at very limited quantities or turning to emulation and piracy.

2.5 Media preservation efforts

With Nintendo's tight fist on their intellectual property, including the closure of first party online marketplaces [34], the reluctance to bring forward these titles on their newer consoles and the litigation against emulators, the only reliable way to access retro games is through dedicated preservation groups. These groups ensure the availability of ROMs for games that are no longer sold or supported. As game cartridges degrade over time and older consoles also degrade, 'lost media' becomes a serious concern [35] and with growing discontentment towards these explicitly anti-consumer practices. Initiatives like the 'Stop Killing Games' petition within the EEA aims to hold video game publishers to account as games become unplayable as the publisher stops supporting a video game, with the aim to either allow the public access to the tools to maintain a game, or force publishers to have an 'end-of-life plan' which would allow customers to play the game they paid for [7].

The importance of media preservation, particularly video game preservation, may not be immediately clear. However, content from the late 1990s holds significant cultural importance, shaping the evolution of modern video games, meaning archiving these works is critical. Yet, István Harkai highlights in his journal article, video games are still subject to copyright law are not yet included in the traditional notion of cultural heritage. As a result, they lack the same legal protections afforded to more conventional forms of media [8]. This gap is further underscored in a 2009 paper presented at the Game Developers Conference, which warns "if we fail to address the problems of game preservation, the games currently being made will disappear, perhaps within a few decades" [9].

2.6 HCI

Throughout history, handheld consoles have evolved significantly with innovative features to enhance how users interact with the console and experience content. One of the more ambitious but ultimately unsuccessful features is the inclusion of an autostereoscopic 3D display [10] on the Nintendo 3DS. While revolutionary for a device from 2011 to offer cheap, glasses-less 3D was not without flaws. Common criticisms of the console included headaches, nausea and vision fatigue, largely attributed to the device's sensitivity with viewing angles. Slight deviations from the optimal position disputed the 3D effect, causing a distorted image and causing discomfort. [11]. These issues were further exacerbated by a phenomenon known as vergence-accommodation conflict (VAC). This occurs when the brain receives mismatched signals between the convergence of the eyes (focussing on the 3D effect) and the accommodation of the eyes (focussing on the display surface). VAC is a well-documented limitation in autostereoscopic displays and is often cited as a primary contributor to the visual discomfort experienced by users of such technology, including the Nintendo 3DS family of consoles [10].

With increasing demand for flexibility in gaming. The industry has shifted towards hybrid consoles that can function both as handheld and home systems. The Nintendo Switch exemplifies this, becoming the second-best selling console of all time, with over 139 million units sold as of 2024 [36] with the successor releasing in 2025 [37]. These consoles aim to bridge two markets, the handheld and home console market, by leveraging modularity. The Nintendo Switch achieves this through its innovative Joy-Con controllers, allowing users to dock the system and play on a TV with detached Joy-Cons, or seamlessly reattach then to "switch" to handheld mode.

This modular approach presents a unique challenge. Designing controllers that are ergonomic in both handheld mode and intuitive when in docked mode. Many users have found the Joy-Cons uncomfortable, citing issues such as their small size, unconventional button placement and lack of a traditional D-pad [38]. Despite the challenges, the hybrid design is being adopted within the industry, with companies like OneXPlayer and Lenovo contributing to this growing trend with the OneXPlayer 2 [39] and Lenovo Legion Go [40] respectively. These systems highlight the increasing demand for "do it all" consoles that adapt to different gaming scenarios while pushing innovation in controller design and usability.

While many of the features that distinguish handheld consoles are mentioned above, significant time and money go into the creation of effective ergonomics in consoles and will play a large role in this project as the comfort and long-term enjoyment of a console is heavily dependent on the

shape. These considerations often go overlooked as well implemented ergonomics allow for better accessibility and can enhance gameplay. However, as Norman states, “Good design is actually a lot harder to notice than poor deign, in part because good designs fit our needs so well that the design is invisible” [12, p. 6]. This makes identifying excellent ergonomic designs challenging, as effective designs often fade when considering user experience. Conversely, poor designs, such as, the flat joystick implementations in the Nintendo Switch Joy-cons and Nintendo 3DS family of consoles, highlight the impact of inadequate ergonomics, with users frequently reporting issues like their thumbs slipping and hand cramps.

3 Specification

Before beginning implementation, it is crucial to first define the device requirements and establish a consistent design language. These steps ensure that the 3D printing and soldering process can be completed with a clear direction to avoid unnecessary delays and wasted materials.

3.1 Requirements

This section will outline the overall requirements of the Novostok device. These have been divided into functional and non-functional requirements. While requirements would typically be gathered based on user stories and feedback, Novostok’s requirements are instead derived from its technical goals. Specifically, the need to allow for input parity across the consoles the Pi 5 can emulate.

3.1.1 Functional Requirements

As Novostok leverages a newer Raspberry Pi 5, the software used to emulate does not have direct support for it and does not have a list of supported consoles at the time of writing. Some inferences based on the Pi’s hardware specifications and hardware testing are required to outline what emulation cores can reliably run.

Among many of the emulation frameworks that exist, they use emulation cores, these are often shared between each different framework so the device can target the cores themselves as input requirements. The earliest consoles like the Nintendo Entertainment System only require a D-Pad and two primary buttons with subsequent consoles introducing more inputs like additional face buttons, dual analogue joysticks and shoulder and trigger buttons. Additionally, as this device is targeting Nintendo DS emulation, a secondary display would be required.

From this, a core set of input requirements can be derived to ensure compatibility across most consoles the Pi 5 is likely able to emulate.

- D-Pad for directional input.
- Four face buttons (A/B/X/Y) to support SNES, GBA and PlayStation 1 era games.
- Start and select buttons to match standard controller layouts.
- Dual analogue joysticks for compatibility with PlayStation 1 and PSP and later systems.
- Four shoulder buttons (LT/RT/LB/RB) to support consoles requiring both trigger and shoulder buttons.
- A primary display for conventional gameplay and typical library browsing.
- A secondary display to support Nintendo DS emulation.
- Automatic display switching when the secondary display is revealed.
- ROM browsing and selection via physical input
- Operation using the official Raspberry Pi 5 USB-C power supply (5V 5A).

While the highest possible fidelity with each individual console is not the primary goal, Novostok aims to offer reliable performance across a wide range of systems, prioritising consistency and affordability over perfect feature emulation. That said, support for additional features such as motion or microphone input could be revisited for future revisions of the Novostok.

The system must support ROM selection, execution, and save/load state functionality, as these are essential features of most emulation platforms. While the specific emulation framework has not yet been finalised, these features are consistently available across the most viable options such as RetroPie, Batocera, Lakka and RecalBox. Meaning that these requirements are expected to be satisfied by default, provided there is correct input mapping and accessible file storage.

Additionally, display functionality, like outputting to the primary display, is typically handled by the emulation framework. However, given the requirement of dual displays, a choice on implementation is required. Between a clamshell design like that of the Nintendo DS itself or something different, the Novostok will use a sliding display that when stowed hides the secondary display allowing for a shorter footprint. And when slid up, will reveal the secondary display and automatically swapping the display resolution to account for it.

Power for the Novostok will be supplied using the official Raspberry Pi 5 USB-C power supply, providing the stable 5V 5A needed to power the Pi and the additional components. While a portable battery system would be more ideal for a fully self-contained handheld console, it is considered out of scope for this project due to the additional complexity, safety concerns and risk of destroying components through testing. This approach will allow for more focused development on the core components with future versions exploring internal power.

3.1.2 Non-functional requirements

The non-functional requirements describe how the system should behave under normal and extended use. Including performance expectations, usability goals, stability, cost constraints and long-term maintainability. While these requirements do not define specific features, they are essential to the overall user experience of the Novostok console.

- Deliver playable performance across the targeted emulation cores, maintaining low input latency
- Provide a plug and play experience with minimal setup and intuitive navigation using physical inputs only (beyond the build guide).
- Maintain performance during extended usage without crashing or thermal throttling.
- Be designed to allow for future expansion or hardware upgrades with minimal to no disruption to the design of the device.
- Remain within a constrained budget using widely available components to ensure reproducibility.

Performance is a critical consideration, particularly for genres that require precise timing such as platformers and fighting games. Low input latency and consistent framerates are essential. The Novostok must be able to meet these standards for its target console range.

The usability of the device plays a key role in delivering a seamless user experience. The system should boot directly into the emulation framework and be fully operable using the onboard input controls, with no reliance on external peripherals such as a keyboard or mouse.

Although the device adopts a compact and handheld form factor, its dependence on wall power limits portability. This is an acknowledged constraint of the current iteration and the use of the Raspberry Pi 5 particularly but was defined this way to avoid the complexity and risks associated with engineering an internal battery and management system. Future versions will revisit this aspect as the design matures.

Reliability is essential for any consumer-facing hardware. The system should maintain consistent performance under prolonged use, supported by adequate thermal and power management to prevent overheating or throttling.

Finally, the Novostok is intended to be an accessible and reproduceable design. While not recommended on this iteration, it is still a core principle, this means components should be affordable and sourced from reliable suppliers. Where feasible, the internal layout and software should allow for future upgrades to the hardware.

3.2 Prototype and early concepts

With both functional and non-functional requirements defined, initial concept sketches were developed to explore how the Novostok device could meet its technical and ergonomic goals. Early sketches were produced to outline features like form factor, control layout and the display configuration given the engineering challenges it posed.

Given that the sliding display mechanism serves as the device's defining feature, establishing how it would operate was crucial. While the device could imitate the clamshell design of the Nintendo DS, it presents a major limitation, if a user chooses to play a single-display game, such as those from the Game Boy, the second display remains unused, occupying unnecessary space, increasing the footprint and drawing additional power.

To address this, the concept illustrated in figure 1 introduces a vertically sliding display system. Inspired by the rail-based sliding mechanisms used in late 2000s mobile phones. This design allows, with one fluid motion, to reveal the secondary display. Utilising one part of a reed switch embedded in the bottom portion of the enclosure, and the other part within the top display fixture. Then when the magnet disengages, this activates a script that would power on the secondary display, set the resolution to form a singular tall display and match the brightness of the two displays. Then, when the user slides the display back down, the magnet within the reed switch engages and performs the same actions in reverse.

This design would resolve the inefficiency of clamshell-based designs while maintaining flexibility. However, it introduces new challenges, such as ensuring the top display is protected from scuffs and scratches when in the closed position, an issue that has been noted for refinement in the future.

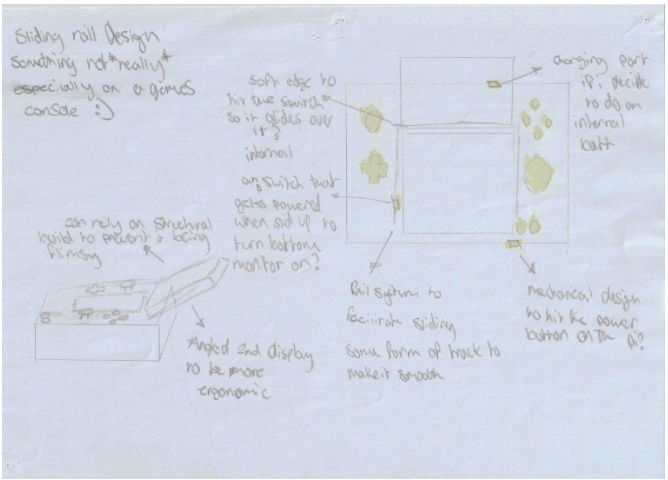

Figure 1. Early design sketch illustrating sliding display functionality

To further evaluate the feasibility of the proposed sliding display feature, a low fidelity cardboard prototype was constructed based on the initial sketches. The prototype was created using the dimensions of the Raspberry Pi 5, displays, and control layout. This prototype helped in understanding the engineering challenges of a sliding rail system and informing the size and ergonomics of the final system (Figure 2).

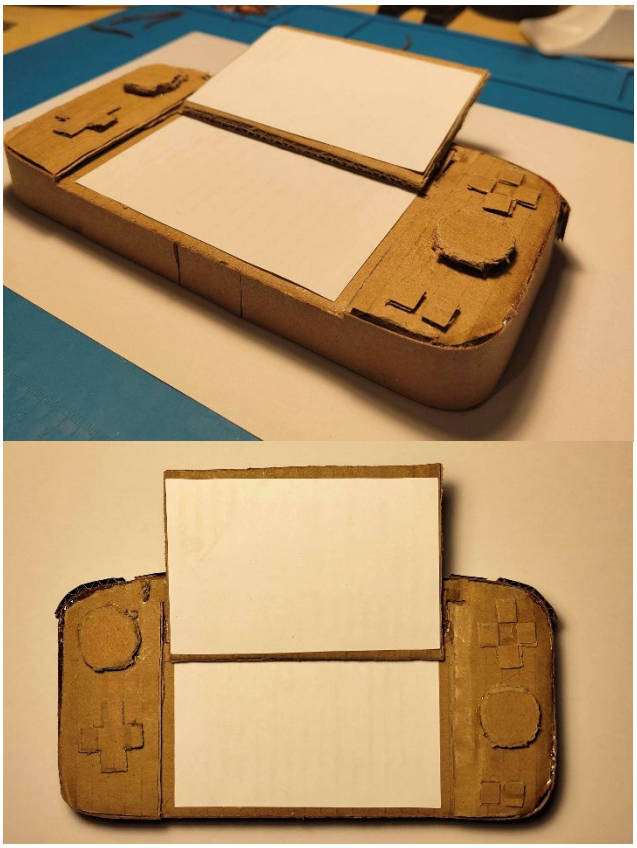

Figure 2. Cardboard prototype. (a) side view illustrating ergonomics and thickness. (b) top-down view showing device footprint

3.3 Initial testing

Through a limited hardware test, this approach will no longer be viable. The initial device specifications were written up to use the DSI standard [41] to connect both displays, which would allow for lower power consumption compared to HDMI and the convenience and space savings of a singular thin ribbon cable as the interface. While in theory the Raspberry Pi 5 [42] can support dual independent DSI displays and does successfully handle it with a desktop environment and window manager like that found in Raspberry Pi OS, it does not support it natively on a headless operating system like Pi OS Lite. Furthermore, the BCM2712 SoC of the Pi 5 uses a unified display pipeline under the Direct Rendering Manager (DRM) causing the DSI panels to not be seen as independent framebuffers, this means that the firmware only exposes one primary framebuffer for DSI usage and the handling for the secondary DSI display is not implemented in Pi OS Lite.

To overcome this, testing using HDMI displays was done, as this would bypass the framebuffer issue. however, software limitations still prevented HDMI displays from working for this approach for several reasons. Emulation station, the graphical interface used to launch games on RetroPie, as it is headless, it does not run within a window manager or include conventional ways to manage display settings. Because of this, typical tools like xrandr [43] will not work and without direct support to extend displays to act as a singular tall display in the DRM (Direct Rendering Manager) this approach is not feasible. Although support for this use case could be achieved through custom configurations and firmware modifications, the software used does not include direct support for this functionality. Given the significant development effort required and the inherent instability of such an approach, the cost outweighs the negatives.

Given the low-level nature and obscurity of this aspect of the device, documentation and discussion online is sparse meaning prior research only returned that this aspect of Novostok was plausible and did not highlight any issues with implementing dual displays. However, shifting towards a singular display device is still possible and the requirements will be adjusted accordingly.

3.4 Revised requirements

As outlined above, testing revealed that current emulation frameworks do not support dual display configurations on the Raspberry Pi 5. This directly impacts the Novostok's ability to emulate Nintendo DS titles as original intended, while still possible the experience would be severely degraded as they rely on a taller total screen resolution size. The initial requirement for a secondary display is therefore no longer possible within the current software and hardware constraints.

Despite the removal of this feature, the remaining functional and non-functional requirements will remain unchanged. The system continues to target reliable emulation of a wide range of retro consoles, up to and including the PlayStation 2. The input scheme remains valid, as it supports all necessary controls for the revised console targets. Display output, thermal stability, and interface usability are still critical components of the overall specification and are unaffected by this revision.

While Nintendo DS support has been deprioritised in this iteration of the device, the design remains modular and can be adapted in the future to reintroduce dual display support, should software compatibility improve, or alternative emulation platforms become viable.

4 Design

4.1 Software

There are several popular emulation options available for Raspberry Pi, such as Lakka [59], Recalbox [60], and Batocera [61], each offering unique features and user experiences. Lakka features a simple and PlayStation-like UI that would be an ideal fit for this project. However, like Batocera and Recalbox, Lakka is a standalone operating system meaning it operates as a closed system with no package manager or support for additional software installations because of its read-only filesystem. This limitation prevents customisation beyond its built-in features, making it unsuitable for this project which requires functionality beyond standard emulation like python for scripting.

Given these limitations, a more flexible emulation tool like RetroPie [54] will be used instead, as it allows for greater system customisation and software integration while still providing robust emulation support. Unlike the above mentioned, RetroPie is not a standalone OS but rather a software package that can be installed on a lightweight Linux distribution. This allows for additional software installations, including Samba which, when configured, means users can share files to and from the device on a network.

RetroPie is a widely used emulation framework that combines multiple emulators, called cores, under one unified system, it streamlines the process of playing games from a variety of platforms. Leveraging RetroArch [62], it ensures a consistent UI, configuration management and control schemes across all emulated consoles and with RetroArch supporting over fifty emulation cores, this is important for parity and ease of use for users of this device.

Since RetroPie is designed to run on Raspberry Pi hardware, it includes various performance optimisation features to ensure smooth emulation across a range of consoles. While, at the time of writing, the Raspberry Pi 5 does not have an official download image, it still benefits significantly from the increased performance from more demanding emulators like those for the N64 and PSP.

While this device modernises retro gaming to a degree, within RetroPie the user can tune, not just the performance of games, but the look and feel of how they play. Using RetroArch, users can add shaders and overlays to simulate the appearance of CRT displays with scan lines, more muted colours and aspect ratio changes. It also includes many quality-of-life changes to many emulators like save states, fast-forwarding and game rewinding and making it possible to pause and resume gameplay at any time.

With the emulation software covered, the device now needs an underlying operating system. As previously alluded, a full featured desktop Linux distribution could have the potential to fix the software issues with dual displays. However, the benefits of that approach are outweighed by the overhead incurred from using it, with significantly more resources used to run the OS, window manager, and background processes which would hinder the experience. Similarly, there are many headless and lite operating systems to choose from

There are many viable options for ARM-compatible, lightweight, headless operating systems such as DietPi [45], which is designed for ultra-minimal resource usage, or Ubuntu Server [46], which provides a feature-rich but heavier alternative. However, Raspberry Pi OS Lite [44] is the best choice in this project due to its first-party support, stability, low system overhead and extensive software availability. As the official operating system for the Raspberry Pi, it provides the highest level of hardware compatibility which is necessary given the DIY nature of this project. Unlike standalone operating systems such as Lakka or Batocera, Raspberry Pi OS Lite allows for full software customisation which allows Samba to be configured to add or remove games and python for scripting of display brightness and input methods.

A further key advantage of Raspberry Pi OS Lite is its long-term viability and community support, as it is actively maintained by the Raspberry Pi Foundation. While DietPi offers extreme resource efficiency, which would provide better performance while playing games, its smaller development team and aggressive modifications to the default Debian system could introduce compatibility issues when installing additional software or further additions to the hardware. Similarly, Ubuntu Server, though feature-rich, is designed as a general-purpose OS and many of those features are not necessary here and would reduce performance in games, while minimal, it is still a reduction. Raspberry Pi OS Lite, in contrast, maintains a balance between flexibility, stability, and hardware-specific optimisations, ensuring long-term support and seamless package updates.

Additionally, stability and security are vitally important considerations for a project of this nature, given it is positioned as a replicable project by anyone, no matter the technical skill. Raspberry Pi OS Lite follows Debian's long-term security update cycle, ensuring consistent protection against vulnerabilities while remaining lightweight. Other distributions, such as DietPi, sacrifice some system security configurations to reduce overhead, while Ubuntu Server focuses on enterprise level system features, of which, many are not relevant or needed in a standalone handheld gaming device.

For input handling, the implementation is more complex, while each individual button could be mapped using the GPIO pins onboard the Raspberry Pi 5 the joysticks could not. As the joysticks output analogue data where each axis of the potentiometer return a voltage value which can be

converted to usable value ranges. This requires specific hardware to read and transmit, for this, the device will make use of a ADS1115 which provides the necessary number of analogue pins. This would then allow the reading of the outputs on the Raspberry Pi 5 using a script to translate each button press to a keystroke and translate the voltage output from the joysticks to a usable range of 0 to 65535. While this would be a simple solution, it has limitations like GPIO polling rates which would case input-delay, made worse when the system is stressed which would be the case when any emulator is running and while negligible with so few inputs, offloading the processing of this data will reduce the resource load and therefore increasing performance of the Raspberry Pi 5.

That is why the Novostok will use a Raspberry Pico microcontroller as the input manager, filtering noise, handling debounce logic and sending only when processed, providing clean data to the Raspberry Pi 5. Initially, the microcontroller would perform the same steps as outlined above, translating each button press to a keystroke and using UART or I2C to send data. But again, this approach would suffice but there is latency overhead associated with UART and would likely result in certain button presses not registering.

This is where Adafruit's CircuitPython utility, with the 'usb_hid' library, becomes particularly advantageous. It allows the microcontroller to function as a USB HID device, eliminating key ghosting and supporting polling rates up to 1000 Hz, ensuring near instantaneous input response times, especially useful for emulated retro gaming as a common complaint of other implementations is input delay. While USB connectivity introduces some physical constraints in a compact enclosure, it provides reliable, low-latency communication between the MCU and the Raspberry Pi.

Additionally, USB HID offers broad compatibility across operating systems requiring no additional drivers or software mapping. Unlike GPIO-based solutions, which rely on software polling and may suffer from higher latency and wiring complexity, USB HID is a plug-and-play solution that ensures consistent performance. Furthermore, its scalability and futureproofing allow for firmware-based upgrades, such as adding macro support or multi-key bindings, without modifying the hardware. This is key for any future iterations of this device, allowing for a more modular upgrade path.

This solution also has the added benefit of being fully supported in RetroPie as it acts as a USB game controller, allowing far easier mapping of inputs. Within RetroArch users would easily be able to remap any given buttons with another. With a GPIO based solution, the bindings for button inputs would be handled using a non-user facing script, requiring some level of technical competency from the user to simply adjust what a given button would do, or creation of a program with the sole purpose of providing the user with a GUI to change key bindings. This would introduce a significant point of failure, adding unnecessary complexity to what should be a straightforward user experience. By leveraging RetroArch's built-in input remapping feature, users could effortlessly customise controls without needing manual script modifications of external configuration tools. This ensures the device remains accessible, user-friendly, and easily adaptable to user preferences, eliminating the technical barriers with a GPIO-based input handler.

Despite the space limitations introduced by USB, the stability, speed and flexibility of this approach make it the most suitable for this project.

4.2 Hardware

A wide range of single board computers are available that could serve as drop-in replacement for the Raspberry Pi 5 in this project. Thanks to input control being handled via USB HID, the choice of SBC is largely determined by computing power, efficiency, software compatibility and crucially, the cost. While many options exist, most of them cater to specific use cases beyond what is required for this project.

One such alternative is the Orange Pi 5 Plus [68], which, on paper, offers superior raw CPU performance with an octa-core Rockchip RK3588S processor and PCIe 3.0 support for expandable and faster storage, like the Pi 5. While a compelling option for high-speed computing tasks, its less refined software ecosystem means that many drivers and optimisations must be manually configured. This creates potential compatibility issues, particularly when setting up emulation or custom Linux configurations. Further compounded as this project is intended to be replicable by users with varying levels of technical competency, if a particular software update breaks compatibility, then it is not guaranteed a given user could solve it.

A slightly more established competitor is the Odroid N2+ [69], which has a quad-core Cortex-A73 and a Mali-G52 GPU. While it performs well in multimedia applications and support similar level of GPIO functionality as the Raspberry Pi, it lacks the same level of software support as seen in Raspberry Pi OS, requiring additional configuration for RetroPie and third-party applications. Meaning this SBC is discarded as an option for the same reasons as the above.

The Raspberry Pi 5 delivers a significant boost in performance over its predecessor, making it well-suited for use in the Novostok. Equipped with a 2.4GHz quad-core Cortex-A76 processor, while the Pi 5 is a contender because of its dual DSI display connectors, the improved CPU and GPU provide welcome performance uplift. This would ensure smooth emulation. Unlike alternatives like the

Orange Pi 5 Plus, the Raspberry Pi 5's performance is optimised for ARM-based applications, ensuring broad software compatibility and a streamlined user experience. While this additional power does come with higher delivery requirements, it remains a versatile and accessible choice for this project.

Beyond raw hardware capabilities, the software and hardware support is unrivalled. With Raspberry Pi OS developed in-house ensuring compatibility and an expansive online marketplace for hardware components which was heavily utilised to obtain all the necessary components for this project. The detailed documentation, software and hardware support far surpasses that seen in other SBCs allowing this project to exist and be done without extreme difficulty aside from the complications already mentioned.

Cost-effectiveness and accessibility are also key factors in making the Raspberry Pi 5 the most practical choice. Priced at £76.50 for the 8gb model, the one used in this project, it delivers a well-rounded feature set without the premium costs associated with AI or ultra-high performance computing SBCs. Moreover, the Raspberry Pi foundation ensures long-term availability, unlike niche competitors that suffer from stock shortages or inconsistent updates. The stability makes the Raspberry Pi 5 a sustainable, replicable, and well-support option for this project.

While there are several viable SBCs, each catering to specific computing needs, the Raspberry Pi 5 emerges as the most practical and well-rounded choice for this project. It offers a balanced mix of processing power, software compatibility and affordability. Without introducing excessive costs of complexity as seen in other SBCs. With native support for RetroPie, extensive documentation, and a thriving community, the Pi 5 ensures both ease of implementation and long-term maintainability, making it the ideal foundation for this build.

As for the MCU, there is no single defining feature that dictated the selection of the Raspberry Pi Pico over alternatives. Many MCUs in this category such as Arduino Nano [70] and ESP32 [71] offer similar pricing and features, with only minor variations in performance, memory, or peripheral support. However, the Raspberry Pi Pico is used in this project primarily for consistency within the Raspberry Pi ecosystem and familiarity with documentation and tools.

In terms of hardware performance, the Raspberry Pi Pico features a dual-core ARM Cortex-M0+ processor running at up to 133MHz, with 264KB of SRAM and 2MB of onboard flash storage. While this is significantly less powerful than higher-end MCUs like the ESP32, the computational demands of this project are minimal, the MCU is only required to parse button and joystick inputs and send them through USB HID signals, a task that requires very little processing power. This means that any MCU that supports MicroPython would, in essence, perform identical.

From a cost perspective, the Pico is one of the most affordable microcontrollers available, retailing at £4.80 for the model in this project. Given that alternative MCUs such as the Arduino Nano (£18) or ESP32 (£7-£10) cost significantly more without offering any meaningful benefits for this specific application, the Pico provides the best balance of affordability and usability.

5 Implementation and Testing

The implementation of Novostok is documented chronologically, following the logical progression of the device's construction and integration stages. Each major subsystem, including enclosure design and fabrication, soldering, software configuration and final assembly, is presented under its own subheading to clearly separate the development of individual components.

Given this shift from conventional development, neither a waterfall nor agile approach could be used. Instead, a Gantt chart was produced at the outset, outlining key milestones across both documentation and hardware and development tasks. This included separate planning for major phases such as component integration, user testing and report writing. While the initial plan scheduled multiple user tests, these were consolidated into a single final evaluation phase to better align with the project (Figure 3).

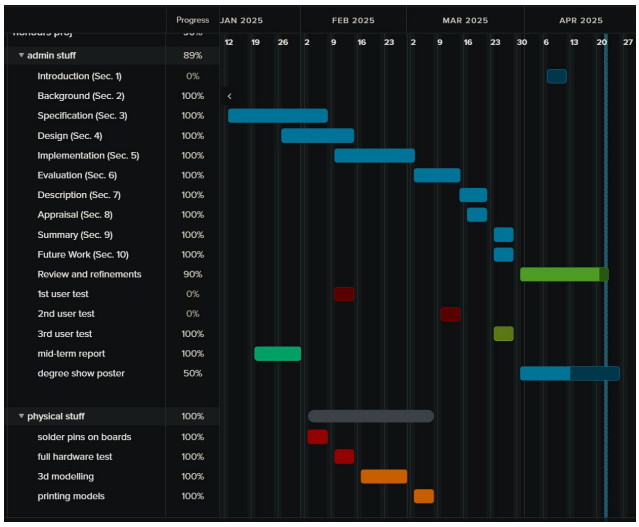

Figure 3. Gantt chart outlining project timeline.

5.1 Enclosure design and fabrication

The enclosure serves as one of the most important aspects of the device and plays a critical role in both functionality and ergonomics. A finalised sketch was produced to guide the 3D modelling process (figure 4). While not detailed in measurement, the sketch established a visual design

language inspired by the Nintendo DS [18], which informed decisions on button placement and chamfered edges.

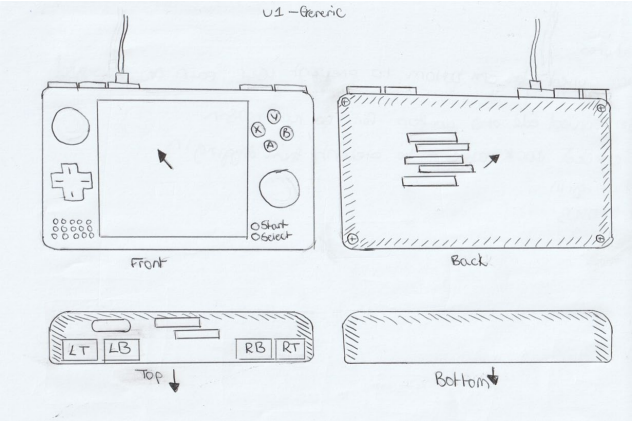

Figure 4. Finalised design sketch.

5.1.1 Enclosure modelling

Following the initial paper sketches and cardboard prototype, the design was refined to accommodate the physical dimensions of the hardware and the precise position of inputs. This process required more than simply measuring the external dimensions, each internal component was individually modelled using Autodesk Fusion to allow for accurate placement, proper internal clearance and efficient use of internal space. All parts were modelled to a dimensional accuracy of ±0.1mm, allowing for proper fitment and alignment for the full enclosure (figure 5).

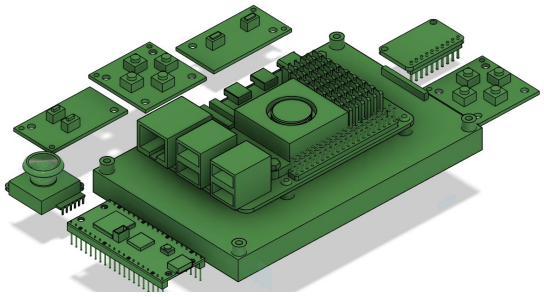

Figure 5. 3D modelled internal components.

Beginning the enclosure modelling process, the internal components like the Raspberry Pi, were arranged in Fusion according to the proportions laid out in the design sketch (figure 4). This alignment was essential to determine the overall footprint and internal spacing for the final device (figure 6).

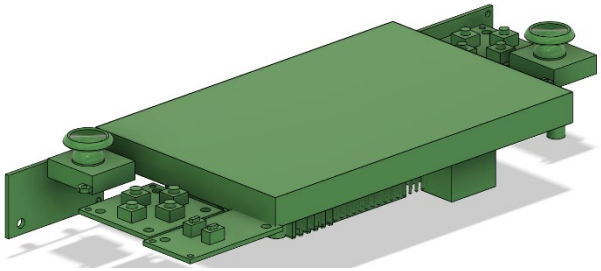

Figure 6. Aligned internal components, isometric view.

A bounding volume was then created by extruding a solid enclosure body around the aligned components, ensuring vertical clearance and overall compactness (figure 7). This defined the outer shape of the device prior to any cutouts or interior detailing.

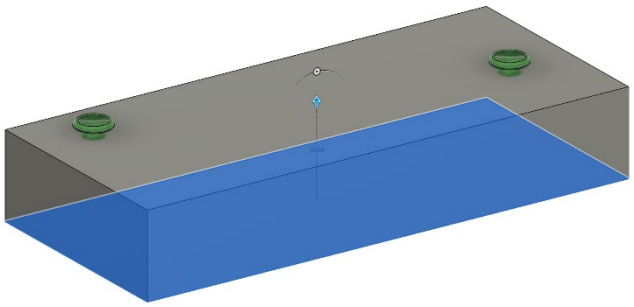

Figure 7. Extrusion of the enclosure body.

A feature that is found across most modern device is chamfered edges, the process of smoothing sharp edges, this will help with ergonomics of the device and prevent the any corners to reduce discomfort during prolonged use (figure 8). This step was completed prior to shelling out the enclosure as the order of operations significantly impacts the overall manufacturing process.

Figure 8. Fillets applied to exterior edges of the enclosure.

Once the external form was finished, a shell operation was applied to hollow the enclosure with a uniform thickness of 2mm (figure 9). In 3D printing and modelling, a common design guideline is to ensure that wall thickness is at least twice the nozzle diameter to prevent warping and deformation. Additionally, the material used, like PLA, PETG and ABS have implications on the design like lower glass transition temperatures and sturdiness. For materials like TPU, a minimum wall thickness of 2mm is

recommended, whereas materials such as ABS 1.5mm is recommended [52]. As these variables were not guaranteed at the time of modelling, a conservative 2mm thickness was used to ensure compatibility and mechanical strength across any possible material and nozzle size.

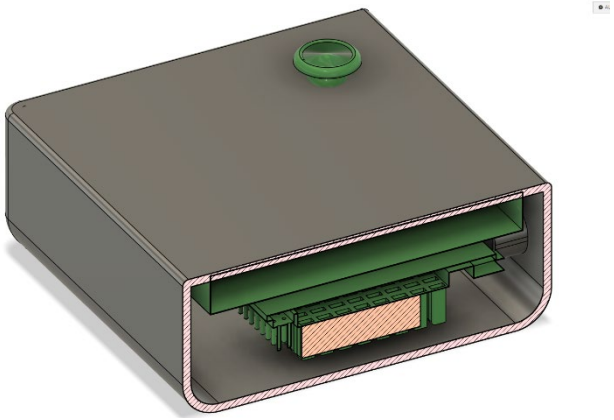

Figure 9. Hollowed enclosure viewed using slice tool.

The top face was marked and cut to fit the display and the physical inputs. Including recesses for the display and face buttons (Figure 9). The placement driven by the alignment of the PCBs and following the placement of consoles like the Nintendo DS.

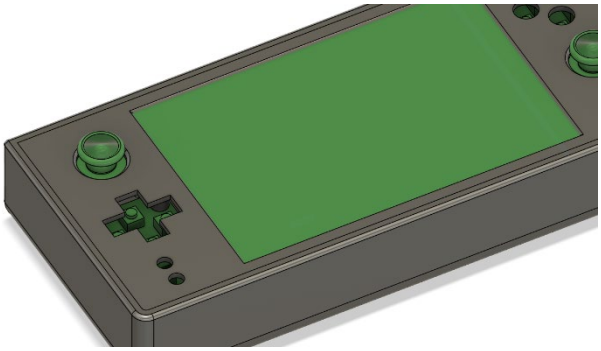

Figure 10. Top face of enclosure after initial cutouts.

Mounting stems were created to support each button and joystick module. These were modelled to match each component height and tolerances. Following early test prints, the stems were thickened to avoid shearing then tapped with M3 screws, they would otherwise split when the tap was inserted (Figure 11).

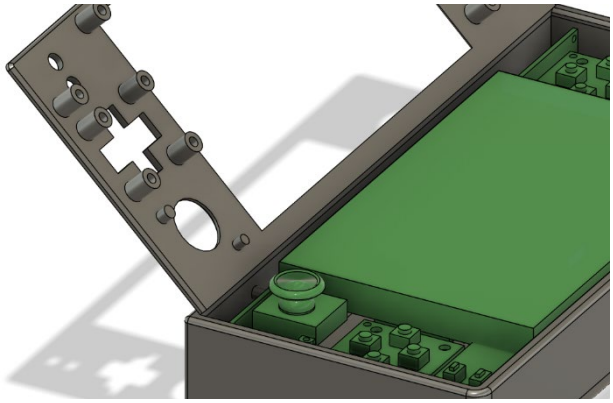

Figure 11. Revised top enclosure showing mounting posts for all input components.

Each button, including the D-Pad and action buttons, were modelled individually to match the cutouts and mounting clearance. With a wide base than the button itself means when the button is secured it would not fall out of the hole (figure 12). These were adjusted based on the fit to ensure smoother travel and reliable actuation.

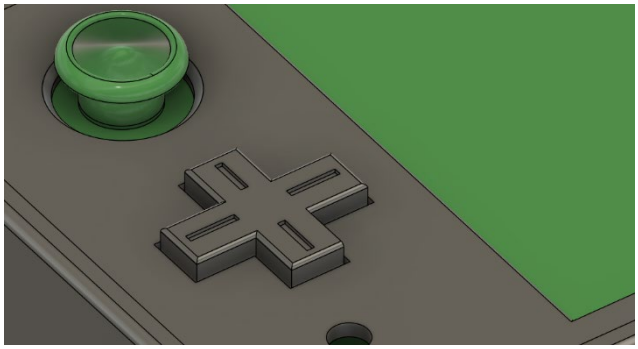

Figure 12. D-Pad button positioned within the top face.

With the faceplate accommodating buttons and joysticks, the enclosure's shell requires cutouts for the shoulder and trigger buttons, input for power and ventilation grills to aid in cooling (figure 13). The power input cutout has guiding extrusion to prevent the use from missing the USB-C power input on the Pi and accidently shorting any of the internal circuits or disrupting the internal wiring.

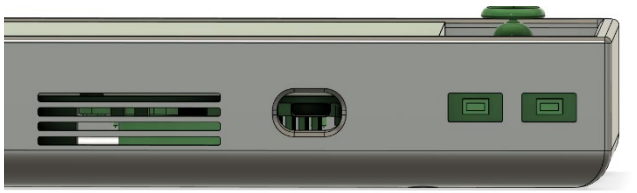

Figure 13. External IO and ventilation on rear of enclosure.

The final modelling steps focused on refining the bottom of the enclosure for thermal performance and mechanical support. The Mounting stems for the shoulder buttons to mount the PCBs were added, a rear-facing vent was positioned above the active cooler on the Raspberry Pi 5,

aiding in airflow. Finally, passthrough mounting stems were introduced to allow for the display and Pi to be fastened securely to the enclosure (Figure 14).

Figure 14. Final enclosure shell with mounting holes and ventilation.

5.1.2 Slicing, 3D printing and post-processing

The final enclosure model was prepared for fabrication. This involved adding sprues to loose components like the buttons and exporting them from Fusion to the STL file format and setting the resolution settings of the exported files.

With the availability of printers and filament types not being guaranteed, the models for the top of the enclosure and buttons were sliced using 50% infill with a layer height of 0.2mm and 1.2mm wall thickness in PrusaSlicer. The infill amount would allow the print to have far higher durability as there would be less hollow space, the layer height would ensure visual quality without significantly impacting print times.

Multiple filament types were used across different parts of the device, selected based on material properties, printing time, and availability. The faceplate and the buttons were printed in ABS due to its improved heat resistance and durability. While the device is unlikely to generate significant internal heat, ABS was chosen over PETG primarily for ABS' rigidity and print times, while PETG has overall better strength its impact resistance is lower and for a device like a handheld console it makes ABS the preferred choice. The faceplate and buttons were printed using a Prusa MK4s, requiring approximately 30 minutes for the buttons and 90 minutes for the faceplate. In contrast, the shell required an estimated 8-hour print time and was instead produced on an older, reliable in-house printer by the university's technical staff. Due to time and scheduling constraints, the model was sliced and printed externally, likely using PLA, with soluble support material removed in a caustic bath the following day.

The post-processing on the 3D printed pieces are minimal with models positioned to reduce the need for supports. However, preparing the faceplate for the mounting of the PCBs for the buttons is required, with thread tapping for an M3 screw thread and glueing of the analogue joysticks. Once complete, the PCBs can be mounted to the faceplate (Figure 16) and enclosure (Figure 16) ready for soldering.

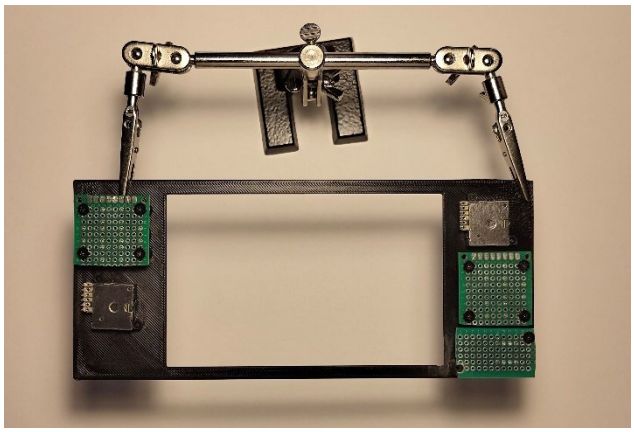

Figure 15. Face place with mounted PCBs and joysticks.

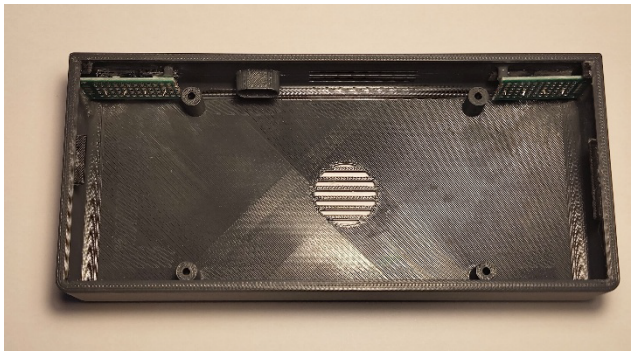

Figure 16. Enclosure with mounted PCBs.

5.2 Soldering

Shared across all fifteen components, is the requirement for seventeen total ground connections. Due to the highly size constrained enclosure, stripboard was used to minimise the total length of wire and to create two common ground rails, each of which could be connected to one of the Raspberry Pi Pico's eight available ground pins. This approach simplified the wiring complexity and reduced the number of discrete wire runs along the base of the enclosure. With this need, a wire colour standard was established, using brown for ground, red for signal and yellow for power and this was used throughout the soldering process.

However, several risks are introduced by this method. Most notably, the long wire bridging the two stripboard sections on the faceplate (Figure 17) may act as an antenna and introduce electrical noise. This could result in inputs not being accurately registered by the Pico. While this issue is difficult to mitigate in such a confined space, it will be monitored and can be further addressed through filtering or debouncing in the input handler script.

Figure 17. Face plate with the long, bridging ground wire.

Another potential issue involves cumulative resistance across common ground paths. Given the length of wire used and the manual nature of the soldering on the stripboard, with some excess solder and residual flux, there is a risk of inconsistent electrical performance. To mitigate this, all ground connections were repeatedly tested using a multimeter's continuity mode, ensuring that the resistance between any two points on the shared ground rail did not exceed 0.2Ω.

Following the soldering of the ground rail of the faceplate, to prevent more complex wiring, a ground rail was added within the enclosure for the shoulder and trigger buttons. During this addition, polyimide tape was placed on the base of the enclosure under where the Pi 5 and Pico would be, this would aid in both thermal management and to prevent electrical shorts (Figure 18).

Figure 18. Enclosure with common ground and polyimide tape.

Next, polyimide tape was placed on components that are at risk of causing shorts. This includes the ADS1115 (Figure 19a), Pico (Figure 19b) and the metal shielding on the right joystick where the ADS1115 would be mounted (Figure 19c). While layering polyimide tape is not necessary, these components will have wires soldered directly to them which has a minor risk of melting through the tape, despite polyimide tape's high thermal resistance.

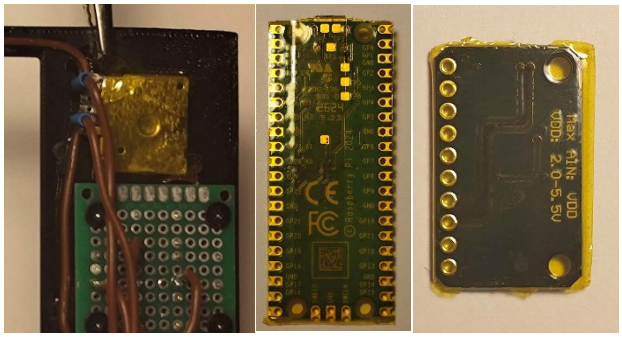

Figure 19. Collection of components with polyimide tape. (a) Right joystick with polyimide tape. (b) Pico with polyimide tape. (c) ADS1115 with polyimide tape.

To keep the faceplate as a mostly contained unit, the ADS1115 was mounted on the only remaining available space. This increased local wiring complexity but significantly reduced it overall. The GND and ADDR pads were soldered to ground, with the latter intentionally grounded to set the I^2C address to its default value (0x48), which was used throughout the project. The A0-A3 analogue input pads were then connected to the corresponding joystick axis, following the manufacturer's wiring diagram [65]. The path for the analogue signal wires were routed along the existing ground wire which reduces the space required (Figure 20).

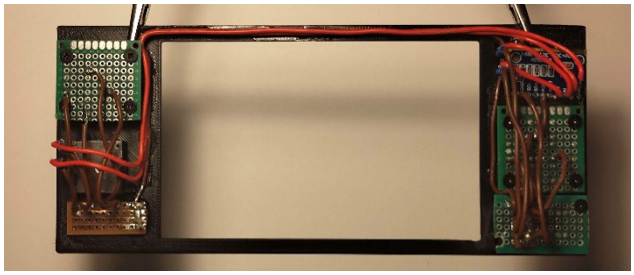

Figure 20. Faceplate with analogue signal wires soldered.

With the faceplate being complete, the Pico was mounted within the enclosure to have signal wires soldered to the GP pads. With order of operations being a necessary consideration to prevent snapping solder joints when the enclosure is shut later, this was done from the bottom components to the top components, starting with the shoulder buttons and 3V rail (Figure 21).

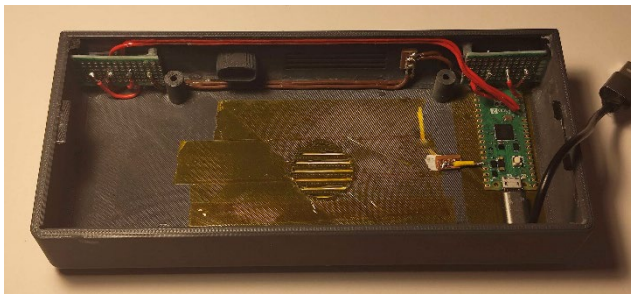

Figure 21. Enclosure with Pico mounted.

Because of the limited number of analogue inputs on the Pico, the SCL and SDA outputs from the ADS1115 were soldered first to ensure that these were secure. Then 3V power was soldered from the power rail to both the left joystick and the ADS1115 (Figure 22a). Finally, the remaining seven signal wires for the D-Pad, start and select buttons and the left joystick click (Figure 22b).

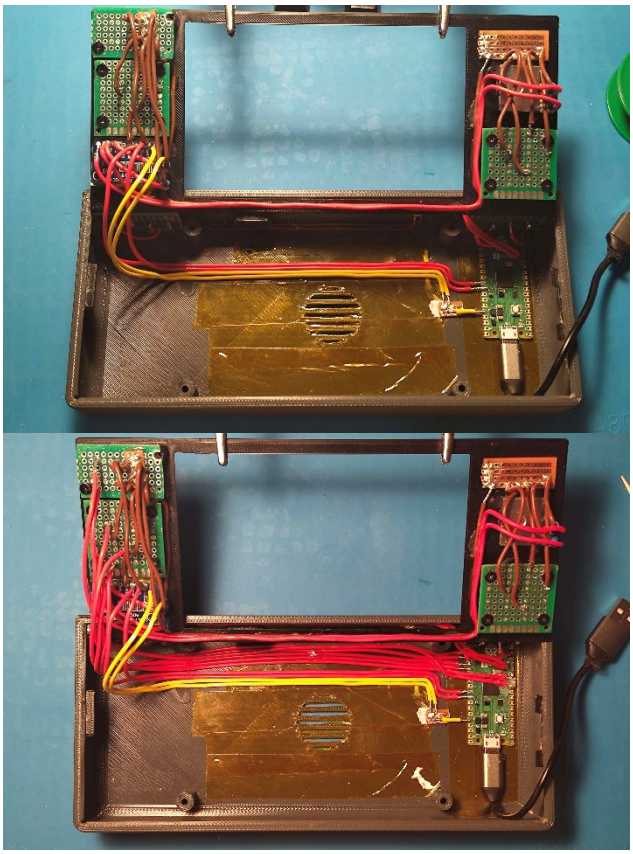

Figure 22. Enclosure and faceplate left side soldered. (a) ADS1115 soldered. (b) signal wires for buttons soldered.

To finish the enclosure wiring, both ground rails were soldered to separate ground pads on the Pico. The signal wires for the right-side joystick switch and the A/B/X/Y buttons were soldered to the Pico, and lasly 3V power to the right joystick (Figure 23).

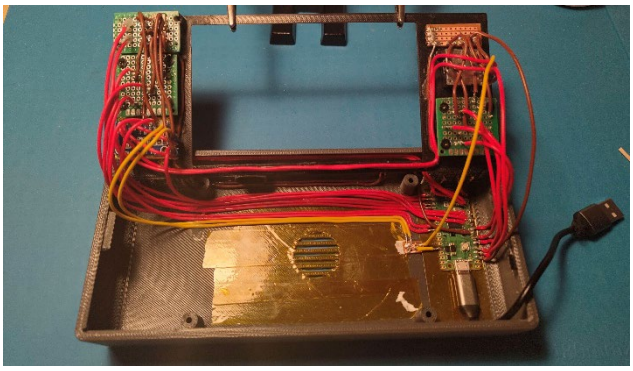

Figure 23. Enclosure with all inputs soldered.

5.3 Software

5.3.1 Raspberry Pi Pico 2

During research and following feasibility testing, this project relied on Adafruit's 'adafruit_hid' library, which previously included the 'hid_gamepad.py' module to simplify implementation of USB gamepad functionality. However, with the final device using the Raspberry Pi Pico 2, only CircuitPython versions 9.x.x and above are supported. By which point 'hid_gamepad.py' had been deprecated as of version 7.x.x.

While the overall 'adafruit_hid' library remains usable in version 9.2.7, the absence of 'hid_gamepad.py' requires the manual definition of a HID report descriptor. Select elements from earlier versions of 'hid_gamepad.py' were still adapted to suit the custom descriptor and input structure required by Novostok.

Because of this, custom HID descriptors need to be defined manually to describe the gamepad's input structure to the host system (Figure 24). The 'boot.py' script contains a six-byte report descriptor that outlines a HID-compliant gamepad with 16 digital buttons and four 8-bit analogue axes: X, Y, Z and Rz. These values are encoded using the standard USB HID Usage Tables, setting logical and physical ranges to ensure the device is recognized as a fully functional gamepad. The descriptor is passed to CircuitPython's 'usb_hid.Device()' class, which is then activated using 'usb_hid.enable()' during the Pico's startup sequence. This ensures the Pico 2 is enumerated as a USB gamepad upon connection, even before any higher-level input logic is executed in the main script. With Adafruit's documentation serving as the basis for the report descriptors used in this project [66].

```
GAMEPAD_REPORT_DESCRIPTOR = bytes((
    0x05, 0x01,       # Usage Page (Generic Desktop)
    0x09, 0x05,       # Usage (Game Pad)
    0xA1, 0x01,       # Collection (Application)
    0x85, 0x04,       # Report ID 4
    0x05, 0x09,       #   Usage Page (Button)
    0x19, 0x01,       #   Usage Minimum (1)
    0x29, 0x10,       #   Usage Maximum (16)
    0x15, 0x00,       #   Logical Minimum (0)
    0x25, 0x01,       #   Logical Maximum (1)
    0x75, 0x01,       #   Report Size (1)
    0x95, 0x10,       #   Report Count (16)
    0x81, 0x02,       #   Input (Data,Var,Abs)
    0x05, 0x01,       #   Usage Page (Generic Desktop)
    0x15, 0x81,       #   Logical Minimum (-127)
    0x25, 0x7F,       #   Logical Maximum (127)
    0x09, 0x30,       #   Usage X
    0x09, 0x31,       #   Usage Y
    0x09, 0x32,       #   Usage Z
    0x09, 0x35,       #   Usage Rz
    0x75, 0x08,       #   Report Size (8)
    0x95, 0x04,       #   Report Count (4)
    0x81, 0x02,       #   Input (Data,Var,Abs)
    0xC0              # End Collection
))

gamepad = usb_hid.Device(
    report_descriptor=GAMEPAD_REPORT_DESCRIPTOR,
    usage_page=0x01,           # Generic Desktop
    usage=0x05,                # Gamepad
    report_ids=(4,),
    in_report_lengths=(6,),
    out_report_lengths=(0,)
)

usb_hid.enable((gamepad,))
```

Figure 24. Code snippet. Boot.py, HID report descriptor structure.

For the ‘code.py’ script, it begins by configuring the ADS1115 to read four analogue inputs from the joysticks axes (Figure 25). These inputs are accessed over I^2C using the ‘busio’ module, with GPIO pins, GP27_A1 and GP26_A0 acting as SCL and SDA respectively. The ‘AnalogIn’ class from the ‘adafruit_ads1x15’ library is used to assign each axis, left X/Y and right X/Y, to one of the ADS1115’s four channels. Following this, the ‘button_pins’ dictionary maps each digital input to the GPIO pad they were soldered to on the Pico 2. These pins are configured with pull-up resistors and stored in a dictionary of ‘DigitalInOut’ objects to streamline input checking within the main loop as reducing latency as much as possible is critical. Finally, a ‘gamepad’ object is instantiated to send HID input to the host device.

```
# setup ADS1115
i2c = busio.I2C(scl=board.GP27, sda=board.GP26)
ads = ADS.ADS1115(i2c)
ads.gain = 1

# analogue joystick inputs from ADS1115
left_x = AnalogIn(ads, ADS.P1)   # A1
left_y = AnalogIn(ads, ADS.P0)   # A0
right_x = AnalogIn(ads, ADS.P3)  # A3
right_y = AnalogIn(ads, ADS.P2)  # A2

button_pins = {
    "rt": board.GP13,
    "rb": board.GP12,
    "lt": board.GP11,
    "lb": board.GP10,
    "dpad_up": board.GP21,
    "dpad_down": board.GP18,
    "dpad_left": board.GP19,
    "dpad_right": board.GP20,
    "start": board.GP9,
    "select": board.GP8,
    "a": board.GP3,
    "b": board.GP5,
    "x": board.GP4,
    "y": board.GP6,
    "l_stick_btn": board.GP7,
    "r_stick_btn": board.GP2,
}

buttons = {}
for name, pin in button_pins.items():
    btn = digitalio.DigitalInOut(pin)
    btn.switch_to_input(pull=digitalio.Pull.UP)
    buttons[name] = btn
```

Figure 25. Code snippet. Code.py, input assignment.

The main loop handles the real-time reading and translation of input signals into HID-compatible data (Figure 26). The ‘scale_axis() function centres the joystick inputs around 0, applying a small deadzone to filter out noise and jitter, and then scales the analogue values to the HID-expected range of -127 to 127. These values are then passed into the ‘move_joysticks()’ methods, which updates the USB gamepad’s axis data. For button input, the script iterates through the mapped buttons, checking for logic low (pressed state), and updates the HID report using ‘press_buttons()’ and ‘release_buttons()’. A short 10ms sleep is added between cycles to reduce system load and debounce input readings.

```
def scale_axis(val):
    delta = val - JOY_CENTER
    if abs(delta) < JOY_DEADZONE:
        return 0
    return max(-127, min(127, int((delta / JOY_RANGE) * 127)))

while True:
    lx = scale_axis(left_x.value)
    ly = -scale_axis(left_y.value)
    rx = scale_axis(right_x.value)
    ry = -scale_axis(right_y.value)
    gp.move_joysticks(x=lx, y=ly, z=rx, r_z=ry)

    for i, (name, btn) in enumerate(buttons.items()):
        if not btn.value:
            gp.press_buttons(i + 1)
        else:
            gp.release_buttons(i + 1)

    time.sleep(0.01)
```

Figure 26. Code snippet. Code.py, input handling logic.

With input handling complete, testing was conducted to verify that HID inputs were received correctly beyond the serial console environment (Figure 27a) and could be recognised by the RetroPie input configuration tool and jstest tool within Linux. While an issue was noted with the Raspberry Pi 5 not detecting the Pico if it was connected at boot, the controller was successfully recognised after hot plugging once the system had started (Figure 27b).

```
Gamepad running...
Pressed: b
Left stick:  X= -98  Y=  77   | Raw X=7341 Y=12739
Right stick: X= -98  Y=  76   | Raw X=7339 Y=13046
Released: b
Left stick:  X= -98  Y=  77   | Raw X=7349 Y=12885
Right stick: X= -98  Y=  76   | Raw X=7345 Y=13049
Pressed: a
Released: a
Pressed: a
Left stick:  X= -98  Y=  77   | Raw X=7350 Y=12866
Right stick: X= -98  Y=  76   | Raw X=7356 Y=13049
Released: a
Pressed: a
Released: a
Left stick:  X= -98  Y=  77   | Raw X=7349 Y=12870
Right stick: X= -98  Y=  76   | Raw X=7342 Y=13050
Left stick:  X= -98  Y=  77   | Raw X=7349 Y=12995
Right stick: X= -98  Y=  76   | Raw X=7347 Y=13049
Left stick:  X= -98  Y=  78   | Raw X=7349 Y=12602
Right stick: X= -98  Y=  76   | Raw X=7349 Y=13046
Left stick:  X= -98  Y=  78   | Raw X=7343 Y=12575
Right stick: X= -98  Y=  76   | Raw X=7340 Y=13046
Left stick:  X= -98  Y=  78   | Raw X=7345 Y=12601
```

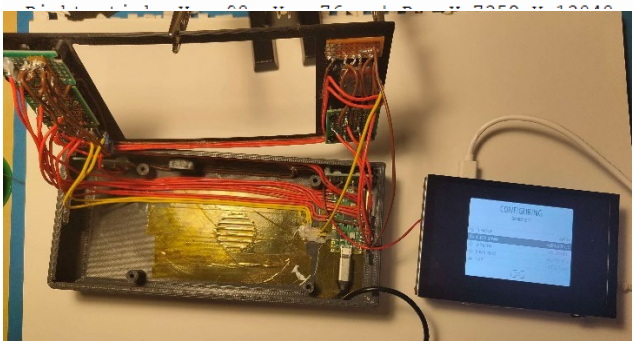

Figure 27. Input testing. (a) Serial editor shell output. (b) input detection on final hardware.

5.3.2 Raspberry Pi 5

To begin software implementation for the Raspberry Pi 5, Raspberry Pi OS Lite was installed using the official Raspberry Pi Imager. RetroPie was then setup by cloning the official repository and running the setup script, choosing the full installation to ensure all available emulation cores and configuration utilities were included. More beta cores can be installed directly within RetroPie later.

With the base system configured, the '/boot/firmware/config.txt' file was modified to support DSI display output (Figure 28a). The Waveshare brightness script was installed and a systemd service and script were made to set the brightness on boot otherwise the display would not turn on (Figure 28b) (Figure 28c).

```
[all]

dtoverlay=vc4-kms-v3d
# DSI1 Use
# dtoverlay=vc4-kms-dsi-waveshare-panel,7_0_inchC
# DSI0 Use
dtoverlay=vc4-kms-dsi-waveshare-panel,7_0_inchC,dsi0

disable_touchscreen=1
```

```
#!/bin/bash
echo 122 | tee /sys/class/backlight/*/brightness
```

```
[Unit]
Description=set display brightness at boot
After=multi-user.target

[Service]
Type=oneshot
ExecStart=/usr/local/bin/set-brightness.sh
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

Figure 28. Configuration files. (a) config.txt additions. (b) set brightness bash script. (c) set brightness systemd service file.

Following this, Samba was configured to set the '…/roms/' directory of RetroPie to be a network share, allowing for ROMs to be transferred to the Pi 5 over a network as opposed to relying on the ROMs being pre-baked into the OS image or transferring them to and from a USB drive (Figure 29). It is assumed that ROMs used for testing and demonstration were obtained legally, such as through personal backups of owned media.

```
[RetroPie]
path = /home/naomi/RetroPie/roms
writeable=Yes
create mask=0777
directory mask=0777
public=no
```

Figure 29. Samba share configuration for RetroPie.

As noted earlier, the Pico would only be detected by RetroPie and 'jstest' if it was fully disconnected and reconnected after the Pi had booted. Various methods were attempted to resolve this, including forcefully re-enumerating the Pico in code.py, dynamically unlinking and re-linking the USB device and cutting power using 'uhubctl'. However, none of these approaches succeeded in performing a full hardware reset of the Pico.

While this limitation would not pose a problem in an open system with accessible USB ports, where a user could

manually disconnect and reconnect the controller, it is problematic in a sealed enclosure. To resolve this, a hardware-level workaround was implemented by wiring the Raspberry Pi 5's GPIO to the Pico's 'RUN' pad (Figure 30). This allows the Pico to be reset programmatically. A dedicated systemd service (Figure 31a) and accompanying Python script (Figure 31b) were created to pull the 'RUN' pin low after a short delay following system boot. Ensuring the Pico is properly reinitialised and recognised as a USB HID device.

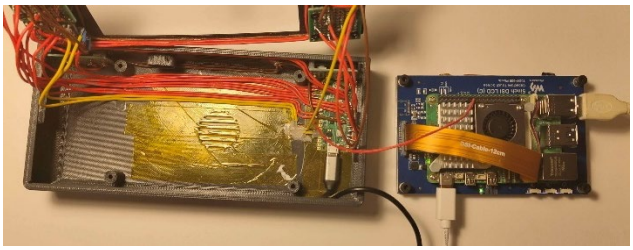

Figure 30. Pi 5 GPIO 22 to Pico 'RUN' pad.

```
from gpiozero import LED
from time import sleep

sleep(10)

reset = LED(22)

reset.off()
sleep(0.2)
reset.on()
```

```
[Unit]
Description=reset raspberry pi pico via GPIO
After=multi-user.target

[Service]
Type=oneshot
ExecStart=/usr/bin/python3 /usr/local/bin/pico-reset.py
RemainAfterExit=true

[Install]
WantedBy=multi-user.target
```

Figure 31. Configuration files. (a) reset Pico script. (b) reset Pico systemd service file

5.4 Final assembly

With each subsystem complete, implemented and tested individually, the final assembly of the Novostok device was completed. This involved mounting the Pi 5 and display assembly within the enclosure. All the wiring was routed through the base of the enclosure, and the faceplate was closed with aid from a heat gun to make the wiring more malleable and a small amount of hot glue was used to hold the faceplate down.

During final integration testing, an issue was identified with the analogue joystick modules. On each joystick, only one axis produced meaningful data, while the other axis exhibited minimal change, likely just electrical noise. Multiple troubleshooting steps were undertaken, including rewiring, modifying the ADS1115 gain and testing alternate analogue pins, but the issue persisted. Notably, the joystick modules did not specify a rated operating voltage in their documentation, although they were supplied with 3.3V in line with the rest of the system. It remains unclear whether the root cause was insufficient voltage or a hardware defect.

Due to time constraints in the final phase of development, the joystick input code was commented out of in the final 'code.py' implementation. The remaining digital inputs were unaffected, and the device retained full functionality for D-Pad based control. Restoring joystick functionality is a priority for future revisions.

The images below show the completed Novostok device, fully assembled and powered on.

Figure 32. Novostok. (a) Top-down view. (b) Side view.

6 Evaluation / Testing

The evaluation of the Novostok was conducted through two processes. First, a quantitative evaluation was performed by assessing button placements against the parameters outlined in the "The Ergonomic Development of Video Game Controllers" [67]. This paper provides a pass/fail system based on the dimensions of input buttons, with ideal size ranges specified. Secondly, a user study was conducted to

evaluate user response to the Novostok and to contrast with the data from the critical evaluation.

6.1 Critical Evaluation

Evaluation of the A/B/X/Y button group was conducted using the criteria in the "The Ergonomic Development of Video Game Controllers" [67]. The paper specifies an ideal input diameter range of 13-25mm. Measurements taken from the Novostok show an average button diameter of 6.9mm, falling below the recommended range and resulting in failure according to the established criteria.

The D-pad was then evaluated, the same ideal range of 13-25mm applied [67]. Measurements of the Novostok's D-pad show an average diameter of 21.5mm, which falls within the acceptable range, resulting in a pass.

Following this, the analogue stick diameter was assessed. With the same ideal range of 13-25mm [67], the Novostok's analogue stick measured 14.7mm, also meeting the criteria indicating a pass.

Trigger width was then evaluated. Although the height of the trigger was not considered, width was measured against an idea range of 6-13mm [67]. The Novostok's triggers measured 11.5mm in width, falling within the recommended range and thus passing.

Finally, the face button spacing was assessed, defined as the internal distance between each button within the A/B/X/Y group. The ideal spacing is specified as greater than 13mm [67]. Measurements of the Novostok showed an average spacing of 9.3mm, indicating a failure to meet the recommended standard.

Due to equipment limitations, direct measurement of actuation force could not be performed. Although the paper outlines ideal actuation force ranges (expressed in newtons), the available product descriptions only provide actuation force in gram-force (gf). Additionally, the conditions under which these gf values were measured are not publicly disclosed, making them unsuitable for reliable evaluation, as a result, no force-based comparison was conducted. A summary of the button evaluations is provided in Table 1.

Button/group	Width(mm)	Ideal Width	Result
A/B/X/Y	6.9	13-25mm	Fail
D-pad	21.5	13-25mm	Pass
Analogue stick	14.7	13-25mm	Pass
Triggers	11.5	6-13mm	Pass
A/B/X/Y spacing	9.3	>13mm	Fail

Table 1. Novostok evaluation results summary

In recognition of the size differences between handheld and home console devices, a comparative evaluation was conducted using a Nintendo 2DS XL as that device is a more appropriate analogue for comparison. Measurements of button sizes and spacing on the 2DS XL were compared against the same ergonomic criteria [67]. Despite the 2DS XL being a commercially successful device, several of its inputs also failed to meet the strict ideal ranges outlined for console controllers. This suggests that while the Novostok does not fully meet all ideal measurements, the deviations are consistent with accepted norms in handheld device design. A summary of these results can be seen in Table 2.

Button/group	Width(mm)	Ideal Width	Result
A/B/X/Y	7.8	13-25mm	Fail
D-pad	18.2	13-25mm	Pass
Analogue stick	14.5	13-25mm	Pass
Triggers	18	6-13mm	Fail
A/B/X/Y spacing	8.9	>13mm	Fail

Table 2. 2DS XL evaluation results summary

6.2 User Feedback

To further evaluate the ergonomics and usability of the Novostok device, a user study was conducted involving five participants. Given the influence of hand size in perceived comfort and control accessibility, participants' hand lengths were measured and compared against an approximate benchmark value.

As no universal standard for global hand size exists, and no reliable regional average for the UK is available, a 19.05cm hand length was used as a general reference point. This benchmark was derived from a control measurement of the author's hand, measured from the tip of the middle finger to the proximal wrist crease, consistent with standard anthropometric practice. However, it is acknowledged that this control measurement is not representative of broader populations and is only used to provide a consistent internal reference point for comparative purposes within this study.

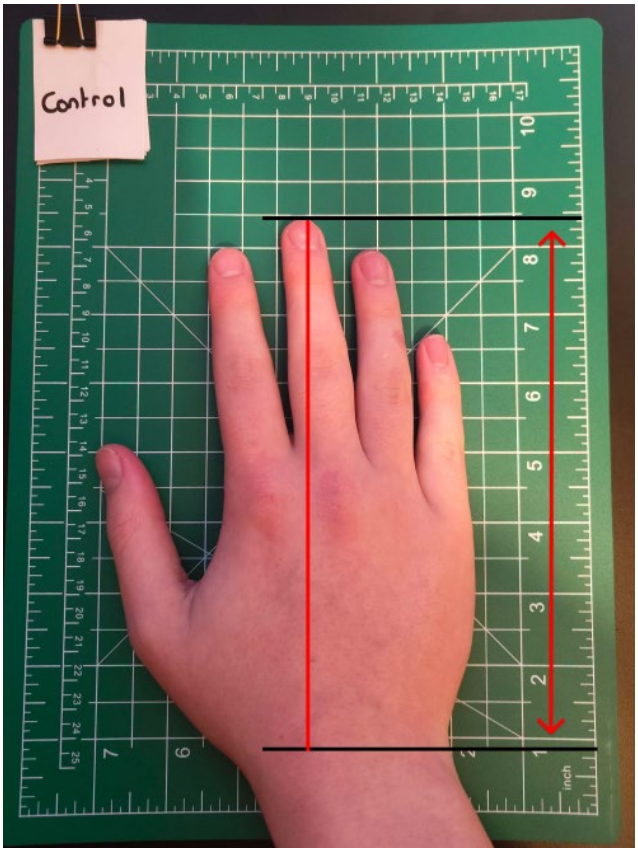

Figure 33. Control hand size measurement.

The following evaluation combines subjective questionnaire feedback with hand size estimations to investigate whether device usability trends correlate with participant hand size.

Participant Number	Hand size (cm)	Compared to control (19.05cm)
1	19.36	Slightly larger (+1.63%)
2	20.63	Larger (+8.29%)
3	17.14	Smaller (-10.03%)
4	16.95	Smaller (-11.02%)
5	16.51	Much smaller (-13.33%)

Table 3. Hand size of participants.

With participant hand sizes recorded relative to the 19.05cm internal benchmark, the following tables present key findings from the user questionnaire. Five questions were selected for focussed analysis based on their relevance to ergonomics, usability and controls. Participant responses to each question are presented alongside a brief discussion of observed trends.

Participant Number	Response
1	Somewhat easy
2	Very easy
3	Somewhat easy
4	Very easy
5	Very easy

Table 4. User responses to question “2. How easy is it to reach all the buttons and controls naturally?”

All participants reported a positive experience regarding the ease of reaching all buttons and controls, with responses ranging from “somewhat easy” to “very easy.” Notably, this trend held consistent regardless of hand size, including those with significantly smaller than average measurements relative to the 19.05cm internal benchmark. This suggests that the positioning and spacing of the controls were sufficiently ergonomic for users’ tests, despite the A/B/X/Y cluster failing the paper’s criteria [67].

Participant Number	Response
1	No
2	No
3	Yes
4	No
5	No

Table 5. User responses to question "3. Do you feel any strain or discomfort when holding the device?”

Most users reported no strain or discomfort when using the device during initial use, with only one participant indicating minor strain. However, later responses revealed that three participants, those with smaller than average hands sizes, expressed a preference for additional contouring or curved features to improve grip comfort. This suggests that while short-term use did not produce significant discomfort for most users, extended use may reveal underlying ergonomic issues, particularly for those with smaller hands. However, as noted later, all users reported the device to be bulky to some degree meaning that a slimmer enclosure could solve this problem. As such, this would be a point of focus on the next iteration.

Participant Number	Response
1	Very comfortable
2	Somewhat comfortable
3	Very comfortable
4	Very comfortable
5	Very comfortable

Table 6. User responses to question "4. Are the face buttons (A/B/X/Y) positioned comfortably for your thumb?”

All participants reported that the A/B/X/Y face buttons were comfortable to reach with their thumb, with four users reporting “very comfortable”. This positive trend persisted despite the Novostok’s face button spacing failing the ideal measurements outlined in the quantitative evaluation [67]. These findings further reinforce that slight deviations from established ergonomic standards did not negatively impact user comfortable for the participants tested.

Participant Number	Response
1	Neither easy nor difficult
2	Very easy

3	Somewhat easy
4	Unsure
5	Very easy

Table 7. User responses to question "6. Are the shoulder buttons easy to reach without shifting your grip too much?"

Participant responses regarding the accessibility of the shoulder buttons were generally positive, though less uniformly than for face buttons or main controls. Three participants reported finding the shoulder buttons "easy" to reach without shifting their grip. With one user selecting "neither easy nor difficult," suggesting an acceptable but unremarkable experience. The final participant reported being "unsure," explaining that the games tested did not heavily utilise the shoulder buttons, limiting their ability to form a strong impression. Overall, these results suggest that while the shoulder button placement is generally effective, further testing with gameplay that relies more heavily on shoulder inputs may be necessary to fully evaluate their accessibility and comfort.

Participant Number	Response
1	Yes
2	No
3	Yes
4	Yes
5	No

Table 8. User responses to question "18. If this were a commercial product, would find it comfortable enough to use daily?"

Responses to whether the Novostok would be comfortable enough for daily use were mixed. Three participants responded positively, indicating that they would find the device sufficiently comfortable for regular use, while two participants responded negatively. While this trend reflects a generally positive baseline of comfort, it is important to recognise that the Novostok remains a prototype, with a level of polish that differs from a commercial device. Participant feedback suggests that while the core ergonomic design is functional, further refinements such as enhanced contouring and material improvements would likely be necessary to meet broader commercial expectations.

One thing of note is that while all users reported a positive experience overall with the device, all respondents but one reported the device to either be "very bulky" or "somewhat bulky". While not indicative of a negative experience, it places further emphasis on the limitations of the device because of the hardware choices. Additionally, the thickness of the device may play a role in reducing strain on the users and contribute to the ergonomics of the buttons and will require further study in the next iteration.

While the quantitative evaluation provided a useful initial baseline for assessing the Novostok's ergonomics, it became clear through user testing that real-world experiences are more nuanced than pass/fail metrics would suggest. Although several inputs failed to meet the strict criteria, participants generally reported positive experiences when using the device. This highlights the limitations of purely quantitative data when evaluating ergonomics, particularly in handheld devices where subjective comfort, familiarity and personal physical characteristics play a significant role. A pass/fail system is valuable for establishing design guidelines, but it cannot capture the full range of human variability or user perception. As such, both quantitative and qualitative evaluations were necessary to gain a complete understanding of the Novostok's ergonomics.

For the full responses to the user study see appendix [1].

7 Description of the final product

The final product, Novostok, is a handheld retro-style gaming console built with the Raspberry Pi 5 and a Raspberry Pi Pico 2 microcontroller. Designed to emulate the experience of early 2000s handhelds while integrating the flexibility of modern emulation. The device delivers a self-contained, semi-portable gaming experience, featuring a 5-inch DSI display within the 3D printed enclosure, whose physical form draws inspiration from many retro handhelds like the Nintendo DS and consoles within the fourth through to the seventh generation.

Users interact with the console through a comprehensive input layout, including a D-pad, start and select buttons, A/B/X/Y face buttons, dual shoulder/trigger buttons and two analogue joysticks. While the joysticks are physically present on the faceplate, only one axis on each produced consistent data during testing. As a result, analogue input support was disabled in the final Pico firmware due to unresolved issues with their electrical behaviour. Despite this limitation, the remaining inputs function as intended and the device supports full digital input-based navigation and gameplay.

Upon powering on, the Novostok directly boots into RetroPie, which acts as a front-end to the RetroArch emulator framework. The user is greeted with a streamlined, console-style UI that enables game browsing and launching across a range of platforms up to the PlayStation 2. While PS2 emulation is possible due to the improved performance of the Pi 5, the lack of analogue joystick support presents usability challenges for titles that utilise it, and as such, the users would find the system better suited to earlier generations of games that do not use joystick inputs.

ROMs can be transferred to the device over a network via Samba, allowing for simple drag-and-drop management of games from another computer. This avoids the legal concerns of pre-baking ROMs into the OS image or relying on USB-based file transfers. Systemd services manage

display brightness and input reinitialisation at boot, ensuring the console always starts in a usable state without user intervention.

While the final product is a functional and complete handheld emulation console, it remains a prototype, with some compromises due to time and hardware constraints. The analogue joystick issue is the most significant limitation and while not critical for most 2D and early 3D games it does restrict compatibility with certain systems. Nonetheless, Novostok achieves its goal of delivering a self-contained and replicable platform for retro gaming and it lays a strong foundation for future iterations that may include refined input handling, additional display support or even modular hardware upgrades.

In keeping with the open-source philosophy that inspired much of this project, the Novostok design files and source code are release under the MIT license, allowing for free use, modification and redistribution with the documentation being licensed under CC BY-ND 4.0 license. The full project repository is available in Appendix [2]. To further support replicability and community development, a detailed build guide and bill of materials have also been produced and are available separately in Appendix [3].

7.1 Legal, Ethical, Professional and Social Considerations.

The legality and ethicality associated with the development of Novostok were covered in detail in section 2.4. In particular, the handling of copyrighted material which presents some significant challenges. To align with ethical practices, Novostok was developed without including any ROMs, BIOS files or proprietary software within the system image. Users are required to transfer their own ROMs manually and install the OS from the RetroPie repository themselves. This maintains a clear separation between the device's functionality and the potential misuse of copyrighted content. The project utilises freely available and legally compliant emulation frameworks such as RetroArch and RetroPie, which themselves do not infringe upon intellectual property rights.

Beyond legal concerns, Novostok was also developed with broader professional and social responsibilities within the emulation community as a focus. Emulation plays a significant role in preserving gaming history, especially as older titles become harder to access through legitimate means. However, the potential for enabling piracy is recognised as a risk. By focussing the design on user responsibility and flexibility, Novostok aims to support ethical use while giving users tools to enjoy their own game collections. It is also acknowledged that changes in the legal definitions of emulation and how systems like the Novostok could be affected by that.

8 Appraisal

Throughout this project, the complexity has been the most challenging factor. As the author had no, or little, working knowledge of the technologies and tools that would need to be utilised. While incredibly rewarding, creating the Novostok was not trivial. Admittedly however, the author would do it all again. The approach would not change drastically as it served as an introduction to many different mediums for creation, like soldering, 3D modelling and printing, microcontrollers and SBCs.

Some of the improvements to this project, if it were to be completed again, would centre around project logistics, time management and risk planning rather than the technical approach itself. Utilising the university's resources earlier such as 3D printing and soldering equipment. This could have significantly streamlined several stages of the build process. In addition to allocating more time to unfamiliar tasks, especially around the microcontroller HID configuration and components soldering, would have allowed for a more iterative development cycle with extra time for troubleshooting and refinement. In hindsight, it became clear that accurately scoping a multi-disciplinary project with limited prior experience is inherently difficult.

Specific challenges encountered included the inconsistent behaviour of the analogue joystick modules. As mentioned briefly within section 5.4, the components lacked clear documentation specifying their operating voltage, leading to uncertainty whether their limited functionality was caused by insufficient 3.3V supply or defective modules. Without a safe means to provide 5V, as splicing a USB cable or soldering directly to a 5V trace would risk permanent hardware damage, it was not feasible to attempt a workaround ahead of final integration and user testing. As a result, full analogue input support was disabled in the final implementation. Additionally, as mentioned in section 3.3, the original plan for dual displays had to be revised to a single display due to software and hardware incompatibilities.

8.1 Advice to others

For hardware aspects of this project, particularly soldering, extensive practice is strongly encouraged before beginning work on final components. Soldering in compact enclosures with multiple connections is inherently challenging. Improper technique can lead to cold joints, shorts and signal interference. Careful pre-planning of wire runs can significantly reduce assembly complexity and help ensure components fit as intended without straining solder joints.

A difficulty encountered by the author during this project is the seemingly endless need for components, tools, or materials. Preparing a complete list of everything required and likely required will save a significant amount of time and stress later. Similarly, allocating more time than

initially anticipated, delays in printing, hardware faults and iteration on designs will cost in the long term.

While this project is certainly achievable, it demands considerable time, patience and a willingness to learn across multiple disciplines, including hardware, software and mechanical design. For those starting with limited experience, as was the case in this project, it is important to embrace the learning process and not be discouraged by difficulties. The technical and practical challenges can be substantial, but they provide valuable opportunities for growth.

9 Summary and Conclusions

9.1 Summary

The Novostok represents more than personal passion and technical exploration. What began as a response to the ongoing challenges in retro gaming and media preservation became a deep dive into physical computing and the wider legal landscape. Despite some of the limitations mentioned in the previous section, the final product proves the viability of a device like this in aiding media preservation. While not a direct tool for preserving retro games, legislative action already being taken within the European Union will allow devices like the Novostok to be a viable console amidst an inflated second-hand market and storefront closures.

More than just a technical exercise, this project reflects a broader movement. One where individuals and consumers, not corporations, are required to take responsibility in preserving digital history. Through open-source software, readily available components and shared knowledge, it is now more possible than ever to build tailored devices that meet specific needs. In Novostok's case, a handheld console that places preservation, performance and accessibility first.

In doing so, Novostok answers the question posed in the introduction. Can one person, using off the shelf components and open tools, build a console that feels as cohesive as a commercial product? While not every goal set out in the report was achieved with many areas open for refinement, this project handily meets that goal.

9.2 Conclusion

The Novostok represents the intersection of personal passion and technical exploration. What began as a response to ongoing challenges in retro game and media preservation became a deep dive into physical computing, emulation and low-level input handling. Despite the project's technical hurdles, particularly around power delivery, display integration and analogue input handing, the final product proves that custom built hardware can serve as a viable and modern solution for enjoying and preserving retro games.

More than just a technical exercise, this project reflects a broader movement: one where individuals, not corporations, must take responsibility for preserving digital history. Through open-source software, accessible components and shared knowledge, it is now possible to build tailored devices that meet specific needs, in this case, a handheld console that puts preservation, performance and accessibility first.

While not every goal was met, many areas remain for refinement, the foundation laid here sets the stage for future iterations. In that way, Novostok is not an endpoint, but the first chapter. One that demonstrates how even limited resources, and time can still produce meaningful, working outcomes in the face of technological and systemic obstacles.

10 Future Work

As this device served as a proof of concept and a way to gain the knowledge that creates a device like the Novostok, there are many areas for improvements for hardware, user testing and interviews as noted throughout this report.

10.1 Hardware improvements

One of the most significant limitations of the Novostok prototype is the use of the Raspberry Pi 5. While it served effectively as a placeholder for more suitable hardware like a compute module, as the complexity involved in designing and ordering a small run of carrier boards is large. As such the next iteration would implement a compute module 5 with a carrier board that would significantly reduce the space required as the device does not require features like HDMI, RJ45 and full size GPIO support. Connections like DSI could be placed more conveniently as well as using surface mount points for user input which would remove the need for intricate wire runs as traces on the carrier board would provide these connections.

This change in hardware would introduce some considerations around storage. EMCC variants of the module offer fast, onboard storage, but they require a more complex flashing process and are permanently soldered. Alternatively, a micro-SD card slot could be included on the carrier board, offering user flexibility at the cost of reduced performance and reliability. For future revisions, the integration of an NVMe slot for a 2260-format SSD may provide an optimal balance between speed, reliability and upgradability, albeit with increased complexity in carrier board design and thermal management.

10.2 Power

Integrated battery power was determined to be out of scope for this version of Novostok due to the Pi 5's high and non-standard power requirement of 5V at 5A. Providing this level of power via internal battery introduced significant design and safety concerns. A viable approach would

require high output lithium-ion batteries, a battery management system and a buck converter to step down from, for example, 9V 2.8A to 5V 5A. This not only adds complexity but also increases risks of electrical damage or thermal runaway if implemented incorrectly.

Future designs using the compute module are more accommodating of internal battery power. With lower power requirements and greater design flexibility, it becomes feasible to implement a regulated Li-ion battery system. This could be achieved via a standard off-the-shelf USB power bank or a custom battery and charging circuit designed to output 5V 3A. This shift would allow the device to be become fully portable, aligning more closely with its intended use case.

10.3 Enclosure design

With a change in core hardware, the enclosure will require adjustments to thickness. The current enclosure's form factor is entirely driven by the height and footprint of the Raspberry Pi 5. Transitioning to a compute module would allow a substantially slimmer and more ergonomic shape while maintaining the same input configuration. User testing would be critical to inform contouring, grip comfort and input spacing in the updated design.

A more speculative but promising future direction involves the integration of detachable controls. Given growing industry interest in modular input schemes, the Novostok could adopt a split controller layout in future revisions. This could be achieved using large reed switch and embedded microcontrollers like a Pico or ESP32 in each controller segment, powered by small rechargeable batteries. These could communicate wirelessly via a 2.4GHz link, either by connecting to a hosted access point on the main board or via Bluetooth Low Energy. While this approach is currently conceptual, it presents a possible feature for enhancing modularity and accessibility.

10.4 Dual display support

Dual display support was the original motivator behind the Novostok's form factor, particularly for Nintendo DS emulation. However, due to limited support in both the Raspberry Pi's rendering stack and mainstream emulation frameworks, this feature was ultimately removed to keep scope manageable. Achieving this in future iterations depends on one of three factors, upstream support from the Pi foundation and emulation developers, manual patching by the author, or avoiding dual display rendering altogether.

The latter is currently the most feasible. An alternative approach could involve using a single flexible OLED display configured in a clamshell arrangement. This would allow dual display emulation using one physical display split through configuration. While promising in concept, the cost of flexible OLEDs and uncertainty regarding driver support make this approach speculative. It would also conflict with Novostok's core principle of affordability and replicability through accessible components and is therefore unlikely to be featured in the next revision.

Acknowledgments

I would like to extend my sincere thanks to my supervisor, Gemma Webster, for her guidance, support and encouragement throughout the course of this project. Her expertise and guidance were invaluable throughout the project.

Thanks are also due to the School of Science and Engineering at Dundee University whose facilities and technical staff provided essential support, particularly for 3D printing.

Special thanks also go to Jeff Geering, whose work on YouTube and GitHub was a major source of inspiration. His detailed documentation and exploration of embedded systems and Raspberry Pi hardware helped shape the technical foundation of this project.

Finally, I am deeply grateful to the individuals who volunteered their time to participate in the user evaluation of Novostok. Their feedback provided critical insight into the usability and ergonomics of the device.

References

[1] Nintendo Co., Ltd., "Consolidated Sales Transition by Region," Dec. 2015. [Online]. Available: https://www.nintendo.co.jp/ir/library/historical_data/pdf/consolidated_sales_e1512.pdf. [Accessed: Nov 15, 2024]

[2] S. Vyazigin, A. Dyusembaev, and M. Mansurova, "Emulation of x86 Computer on FPGA," in *2020 IEEE 8th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE) - Proceedings*, Riga, Latvia, 2020, pp. 1–6. doi: 10.1109/AIEEE51419.2021.9435812.

[3] J. Doe, "Ayaneo Flip DS review: Dual-screen gaming PC faces uphill battle," PC Gamer, Oct. 15, 2023. [Online]. Available: https://www.pcgamer.com/hardware/handheld-gaming-pcs/ayaneo-flip-ds-review/ [Accessed: Nov. 22, 2024]

[4] Nintendo of America Inc. V. Bowser, No. 2:21-cv-00519 (W.D Wash. Filed Apr. 16, 2021). [Online]. Available: https://www.courtlistener.com/docket/59831145/nintendo-of-america-inc-v-bowser/ Accessed: Dec. 7, 2024

[5] Sony Computer Entertainment, Inc. V. Connectix Corp., 203 F.3d 596 (9th Cir. 2000). [Online]. Available: https://www.copyright.gov/fair-use/summaries/sony-connectix-9thcir2000.pdf [Accessed: Dec 7, 2024.]

[6] Nintendo of America Inc. V. Tropic Haze LLC, No. 2:24-cv-00123 (W.D. Wash. Filed Feb. 1, 2024). [Online]. Available: https://www.courtlistener.com/docket/68284505/nintendo-of-america-inc-v-tropic-haze-llc/ Accessed: Dec 7. 2024.

[7] “Stop Killing Games: European Citizens’ Initiative,” Stop Killing Games, 2024. [Online]. Available: https://www.stopkillinggames.com/. [Accessed: Dec 9, 2024.]
[8] G. Fekete, “Preservation of Video Games and Their Role as Cultural Heritage,” Journal of Intellectual property Law & Practice, vol. 17, no. 10, pp. 844-851, 2022. [Online]. Available: https://academic.oup.com/jiplp/article/17/10/844/6712222. [Accessed: Dec 14, 2024.]
[9] H. Lowood et al., “Before It’s Too Late: Preserving Games Across the Industry/Academia Divide,” in Proceedings of DiGRA 2009 Conference: Breaking New Ground: Innovation in Games, Play, Practice and Theory, Tampere, Finland. [Online]. Available: https://dl.digra.org/index.php/dl/article/view/468. [Accessed: Dec 15, 2024.]
[10] D. M. Hoffman, A. R. Girshick, K. Akeley, and M. S. Banks, "Vergence–Accommodation Conflicts Hinder Visual Performance and Cause Visual Fatigue," Journal of Vision, vol. 8, no. 3, pp. 1–30, 2008. [Online]. Available: https://jov.arvojournals.org/article.aspx?articleid=2122611. [Accessed: Dec 25, 2024.]
[11] S. Fisher, "Viewpoint Dependent Imaging: An Autostereoscopic Display," in Proceedings of the SIGGRAPH '83 Conference on Graphics Interface, 1983, pp. 136–141. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/344779.344933. [Accessed: Dec 25, 2024.]
[12] D. A. Norman, The Design of Everyday Things, Revised and Expanded Edition. New York, NY, USA: Basic Books, 2013, p. 6.
[13] Atari Lynx Vault, “The Atari Lynx Handheld Console.” [Online] Available: https://atarilynxvault.com/. [Accessed: Jan 24, 2025]
[14] Sega, Sega Game Gear Hardware Reference Manual, 1990. [Online]. Available: https://segaretro.org/images/1/16/Sega_Game_Gear_Hardware_Reference_Manual.pdf?ref=blog.m4ra.net. [Accessed Jan 24, 2025]
[15] Nintendo, Game Boy Owner’s Manual. [Online]. Available: https://www.videogameconsolelibrary.com/images/handhelds/manuals/89_nintendo-game-boy-usa.pdf [Accessed Jan 24, 2025]
[16] Nintendo, Game Boy Color Owner’s Manual. [Online]. Available: https://www.manualslib.com/manual/1049644/Nintendo-Gameboy-Color.html. [Accessed: Jan 24, 2025]
[17] Nintendo, “Technical Data: Game Boy Pocket & Color,” Nintendo Support, [Online]. Available: https://www.nintendo.com/en-gb/Support/Game-Boy-Pocket-Color/Product-information/Technical-data/Technical-data-619585.html. [Accessed: Jan. 24, 2025]
[18] Nintendo, Nintendo DS Operations Manual, UK/DE/FR ed. [Online]. Available: https://www.nintendo.com/eu/media/downloads/support_1/nintendo_ds_19/NDS_Manual_UK_DE_FR.pdf. [Accessed: Jan 24, 2025]
[19] Nintendo, Nintendo 3DS XL Operations Manual, UK ed. [Online]. Available: https://www.nintendo.com/eu/media/downloads/support_1/nintendo_3ds_14/Nintendo3DSXL_OperationsManual_UK.pdf. [Accessed: Jan. 24, 2025]
[20] Nintendo, “Product Information: Nintendo DS Technical Data,” Nintendo Support, [Online]. Available: https://www.nintendo.com/en-gb/Support/Nintendo-DS/Product-Information/Technical-data/Product-Information-619794.html. [Accessed: Jan 24, 2025]
[21] Nintendo, “The Legend of Zelda: Phantom Hourglass,” Nintendo Website, [Online]. Available: https://www.nintendo.com/en-gb/Games/Nintendo-DS/The-Legend-of-Zelda-Phantom-Hourglass-273289.html. [Accessed: Jan 24, 2025]
[22] Nintendo, “Pokémon Platinum Version,” Nintendo Website, [Online]. Available: https://www.nintendo.com/en-gb/Games/Nintendo-DS/Pokemon-Platinum-Version-272321.html. [Accessed: Jan 24, 2025]
[23] Nintendo, “Mario Kart DS,” Nintendo Website, [Online]. Available: https://www.nintendo.com/en-gb/Games/Nintendo-DS/Mario-Kart-DS-271518.html. [Accessed: Jan. 25, 2025]
[24] Analogue Pocket User Guide, v1.0. [Online]. Available: https://assets.analogue.co/pdf/66b5bf11907751940e6f0d36d5fab2c9/Analogue+Pocket+User+Manual+v1.0.pdf. [Accessed: Feb. 3, 2025]
[25] AYANEO, AYANEO Flip DS User Manual. [Online]. Available: https://manuals.plus/m/8de6d5bc129d47bb9e08df879257e6b6214b8bccc2f52b0556cb2fc34035a4f6. [Accessed: Feb 3, 2025]
[26] Valve, Steam Deck Booklet. [Online]. Available: https://media.steampowered.com/apps/valve/2022/steamDeck_booklet_EN.pdf. [Accessed: Feb. 3, 2025]
[27] Samsung, “Galaxy Z Flip6 – Overview,” Samsung Official Website, [Online]. Available: https://www.samsung.com/uk/smartphones/galaxy-z-flip6/. [Accessed: Feb. 3, 2025]
[28] Samsung, “Galaxy Z Fold6 – Overview,” Samsung Official Website, [Online]. https://www.samsung.com/uk/smartphones/galaxy-z-fold6/. [Accessed: Feb. 3, 2025]
[29] Sony Computer Entertainment Inc., PlayStation Classic Instruction Manual, SCPH-1000R. [Online]. Available: https://manuals.playstation.net/document/pdf/SCPH-1000R_A-1.pdf. [Accessed] Feb. 6, 2025]
[30] Macintosh Repository, “Connectix Virtual Game Station,” Macintosh Repository, 1999. [Online]. Available: https://www.macintoshrepository.org/11632-connectix-virtual-game-station. [Accessed: Feb. 6, 2025]
[31] M. Guttenbrunner, “Digital Preservation of Console Video Games,” Master’s thesis, Vienna University of

Technology, 2007. [Online]. Available: https://repositum.tuwien.at/handle/20.500.12708/10895. [Assessed: Feb. 6, 2025]
[32] J. Alexander, "Nintendo pressures Ryujinx emulator developers, leading to removal of downloads," The Verge, Oct. 1, 2024. [Online]. Available: https://www.theverge.com/2024/10/1/24259791/nintendo-ryujinx-switch-emulator-gdkchan-removed-downloads-github. [Accessed: Feb. 6, 2025]
[33] C. Pereira, "Gabe says piracy isn't about price", IGN, Nov. 25, 2011. [Online]. Available: https://www.ign.com/articles/2011/11/25/gabe-says-piracy-isnt-about-price. [Accessed: Feb. 6, 2025].
[34] Nintendo, "Notice of end of purchases in Nintendo eShop for Wii U and Nintendo 3DS (Update April 2024), " Nintendo Support, Apr. 2024. [Online]. Available: https://www.nintendo.com/en-gb/Support/Purchasing/Download-games/Nintendo-eShop/Notice-of-End-of-Purchases-in-Nintendo-eShop-for-Wii-U-and-Nintendo-3DS-Update-April-2024-2174073.html. [Accessed: Feb. 6, 2025].
[35] S. W. Morris, "Preservation of the Video Game," Provenance, Journal of the Society Georgia Archivists, vol. 29, no. 1, pp. 57-76. 2011. [Online]. Available: https://digitalcommons.kennesaw.edu/provenance/vol29/iss1/4/. [Accessed: Feb. 6, 2025].
[36] Nintendo, "Sale units of Nintendo hardware and software," Nintendo Investor Relations, 2024. [Online]. Available: https://www.nintendo.co.jp/ir/en/finance/hard_soft/index.html/ [Accessed: Feb. 6, 2025].
[37] Nintendo, "Nintendo Switch 2 to be released in 2025," Nintendo Official Website, 2025. [Online]. Available: https://www.nintendo.com/us/whatsnew/nintendo-switch-2-to-be-released-in-2025/. [Accessed: Feb. 6, 2025].
[38] G. Ramolete, J, Almirante, J. Mondragon, C. Ting, M. Cohen, and B. Custodio, "Physical design of the Nintendo Switch controller configurations," in Advances in Ergonomics in Design pp. 198-205 DOI:10.1007/978-3-030-51038-1_29.
[39] OneXPlayer, "OneXPlayer 2 Pro (8840U) – Product Page," OneXPlayer Official Store, 2025 [Online]. Available: https://onexplayerstore.com/products/onexplayer-2-pro-8840u?variant=48427096965414. [Accessed: Feb. 6, 2025]
[40] Lenovo, "Legion Go – The ultimate hybrid gaming handheld," Lenovo Official Website, 2025. [Online]. Available: https://www.lenovo.com/gb/en/p/handheld/legion-go-series/legion-go/len106g0001. [Accessed: Feb. 6, 2025]
[41] MIPI Alliance, "MIPI DSI Specification" v1.1, Aug. 2023. [Online]. Available: https://www.zhuoxunweihong.com/wp-content/uploads/2023/08/mipi-DSI-specification-v1.1.pdf. [Accessed: Feb. 19, 2025]
[42] Raspberry Pi Ltd., "Raspberry Pi 5," Raspberry Pi, 2024. [Online], Available: https://www.raspberrypi.com/products/raspberry-pi-5/. [Accessed: Feb. 19, 2025]
[43] Arch Linux, "Xrandr," ArchWiki, 2025. [Online]. Available: https://wiki.archlinux.org/title/Xrandr. [Accessed: Feb. 19, 2025]
[44] Raspberry Pi Foundation, "Raspberry Pi OS," Raspberry Pi Documentation, 2025. [Online]. Available: https://www.raspberrypi.com/documentation/computers/os.html. [Accessed: Feb. 28, 2025]
[45] DietPi Project, "DietPi: Lightweight justice for your single-board computer," DietPi, 2025. [Online]. Available: https://dietpi.com/. [Accessed: Feb 28, 2025]
[46] Canonical Ltd., "Ubuntu Server Guide," Ubuntu Documentation, 2025. [Online]. Available: https://documentation.ubuntu.com/server/. [Accessed: Feb. 28, 2025]
[47] YouMakeTech, Pico-GB: A Game Boy clone powered by a Raspberry Pi Pico, GitHub Repository, 2023. [Online]. Available: https://github.com/YouMakeTech/Pico-GB?tab=readme-ov-file. [Accessed: Mar. 25, 2025]
[48] Null2, "Null 2: Retro Gaming Handheld," Null2, 2024. [Online]. Available: https://www.null2.co.uk/. [Accessed: Mar. 28, 2025]
[49] The Pi Hut, "PiStation case for Raspberry Pi 4," [Online]. Available: https://thepihut.com/products/pistation-case-for-raspberry-pi-4. [Accessed: Mar 28, 2025]
[50] StonedEdge, "Retro Lite CM4," GitHub, 2023. [Online]. Available: https://github.com/StonedEdge/Retro-Lite-CM4. [Accessed: Mar. 28, 2025]
[51] Raise3D, "3D Printing Wall Thickness: The Essential Guide," Raise3D, 2023. [Online]. Available: https://www.raise3d.com/blog/3d-printing-wall-thickness/. [Accessed: Mar. 29, 2025]
[52] Raspberry Pi Ltd., "Raspberry Pi Pico," Raspberry Pi, 2023. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-pico/. [Accessed: Mar. 31, 2025]
[53] Raspberry Pi Ltd., "Raspberry Pi Zero," Raspberry Pi 2023. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-zero/. [Accessed: Mar. 31, 2025]
[54] RetroPie, "RetroPie: Retro-gaming on the Raspberry Pi," RetroPie, 2024. [Online]. Available: https://retropie.org.uk/. [Accessed: Mar. 31, 2025]
[56] Raspberry Pi Ltd., "Raspberry Pi Compute Module 5," Raspberry Pi 2023. [Online]. Available: https://www.raspberrypi.com/products/compute-module-5/?variant=cm5-104032. [Accessed: Mar. 31, 2025]
[57] Raspberry Pi Ltd., "Raspberry Pi Compute Module 4," Raspberry Pi 2023. [Online]. Available: https://www.raspberrypi.com/products/compute-module-4/?variant=raspberry-pi-cm4001000. [Accessed: Mar. 31, 2025]
58] Raspberry Pi Ltd., "Raspberry Pi 4," Raspberry Pi 2023. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/. [Accessed: Mar. 31, 2025]

[59] Lakka, “Lakka – A lightweight Linux distribution that transforms a small computer into a fill blown game console,” Lakka.tv. [Online]. Available: https://www.lakka.tv/. [Accessed: Apr. 10, 2025]
[60] Recalbox, “Recalbox – Retrogaming and media center on your Raspberry Pi,” Recalbox.com. [Online]. Available: https://www.recalbox.com/. [Accesses: Apr. 10, 2025].
[61] Batocera, “Batocera.linux – A retrogaming operating system,” Batocera.org. [Online]. Available: https://batocera.org/. [Accessed: Apr. 10, 2025].
[62] RetroArch, “RetroArch – frontend for emulators, game engines and media players,” RetroArch.com. [Online]. Available: https://www.retroarch.com/. [Accessed: Apr. 10, 2025].
[63] Adafruit Industries, “CircuitPython HID Library,” Adafruit Documentation, [Online]. Available: https://docs.circuitpython.org/projects/hid/en/latest/. [Accessed: Apr. 10, 2025].
[64] Raspberry Pi Foundation, “Remote access – Samba File Sharing,” Raspberry Pi Documentation. [Online]. Available: https://www.raspberrypi.com/documentation/computers/remote-access.html#samba. [Accessed: Apr. 10, 2025].
[65] Adafruit Industries, “Mini Analog Thumbstick – Datasheet,” [Online]. Available: https://cdn-shop.adafruit.com/product-files/5628/P5628_C17894-001-A+datasheet.png. [Accessed: Apr. 15, 2025]
[66] G.21 Gamblor, A beginner’s Guide to Writing USB HID Report Descriptors (by a beginner), Adafruit Playground. [Online]. Available: https://adafruit-playground.com/u/Gamblor21/pages/a-beginners-guide-to-writing-usb-hid-report-descriptors-by-a-beginner. [Accessed: Apr 15, 2025]
[67] E. Thoresen, "The Ergonomic Development of Video Game Controllers," *Journal of Ergonomics*, vol. 6, no. 2, pp. 1–8, 2016. [Online]. Available: https://www.longdom.org/open-access/the-ergonomic-development-of-video-game-controllers-2165-7556-1000209.pdf. [Accessed: Apr. 18, 2025]
[68] Orange Pi, “Orange Pi 5 Plus,” [Online]. Available: http://www.orangepi.org/html/hardWare/computerAndMicrocontrollers/service-and-support/Orange-Pi-5-plus.html. [Accessed: Apr. 18, 2025]
[69] Odroid UK, “Odroid N2 4GB,” [Online]. Available: https://www.odroid.co.uk/Odroid-N2-4GB. [Accessed: Apr. 18, 2025]
[70] Arduino, “Arduino Nano,” [Online]. Available: https://store.arduino.cc/products/arduino-nano. [Accessed: Apr. 18, 2025]
[71] Espressif Systems, “Products: SoCs,” [Online]. Available: https://www.espressif.com/en/products/socs. [Accessed: Apr. 18, 2025]

Appendices

[1] User study responses. Excel file containing the full responses from all participants to the user evaluation questionnaire.

[2] Novostok Project Repository. Full project files including source code, STL models and documentation. Available at: https://github.com/naomisilver/Novostok.

[3] Build Guide and Bill of Materials. Step by step guide to assembling the Novostok device, including required components, wiring and tools. Available at: https://github.com/naomisilver/Novostok/wiki.

[4] User consent form. Blank version of the consent form signed by participants prior to taking part in the user study, outlining their rights and the use of collected data.

[5] Ethics declaration form. Completed ethics declaration form submitted prior to commencing the user study, in accordance with university research policy.

[6] Risk assessment form. Submitted risk assessment form detailing potential hazards during soldering, hardware testing and 3D printing.

[7] Supplementary project photos. Additional photographs of the full development of Novostok, including implementation, testing and final assembly.

[8] Mid-term progress report. The report submitted at the project’s halfway point, outlining the initial progress, key decisions and the development roadmap for the second phase.