NAS Integration with Microcontroller for Data Transfer

Cold-storage Backup Automation using ESP32-S3 and USB-C

Project Planning

Prepared by:

Alexandre Silva

Date: June 29, 2025

1 Abstract

This project explores the integration of a *Network-Attached Storage (NAS)* system with a *microcontroller*, specifically leveraging the *ESP32-S3*, to automate and optimize *cold-storage* backup processes. The primary goal is to facilitate seamless data transfers between the *NAS* and an external *1TB USB-C disk* using the microcontroller's *USB On-The-Go (OTG)* capabilities, enhancing the system's functionality without relying on additional computers or network dependencies.

The project employs the *ESP32-S3* microcontroller for its dual *USB-C* ports, which enable efficient bidirectional data communication and device control. One port is designated for interfacing with the *NAS*, while the other connects to the external storage device, ensuring rapid and reliable data transfers.

Software-wise, the *ESP32-S3* will simply be copying every file specified within a $cold - backup.spec$ file present at the root of the *NAS*' shared storage directory into the hard-drive's root storage, overwriting any files where newer changes occured, ensuring up-to-date backups. Additionally, the *microcontroller* will only be responsible for constructively writing data into the external hard-drive, never deleting files from it.

Additional features include the design of a custom cooling mechanism housed within a 3D-printed enclosure to address potential heat dissipation challenges. Active cooling is achieved through a miniature fan controlled by the *ESP32-S3*, ensuring long-term reliability and performance stability during intensive data transfers.

Contents

2 Components

2.1 NAS

The *NAS* in this project is a **Lenovo ThinkPad T460** laptop that has been modified to serve as a dedicated file server. It is equipped with a **750GB hard drive** to provide ample storage capacity for various files and backups. The ThinkPad is running a standard version of *Ubuntu 20.04*, with the *Samba* service configured to create a shared folder, allowing access to the storage across the network. This *NAS* provides the backbone of the cold-storage backup solution and acts as the primary data hub for file transfers.

- **Model:** Lenovo ThinkPad T460
- **Storage:** 750GB Hard Drive (integrated into the system)
- **Operating System:** Ubuntu 20.04 (with Samba for file sharing)
- **Role:** Centralized data storage and management
- **Additional Features:**
  - Dedicated for cold-storage backup automation
  - Provides network access for data retrieval and management
  - Connected to the microcontroller (ESP32-S3) for data transfer and monitoring

2.2 ESP32-S3 Microcontroller

The **ESP32-S3** is a highly versatile and powerful microcontroller from *Espressif Systems*, designed for a wide range of embedded applications. In this project, the *ESP32-S3* serves as the central control unit, responsible for managing data transfer between the *NAS* system (*Lenovo ThinkPad T460*) and the external storage device.

Programming the *ESP32-S3* is facilitated by the **Arduino IDE**, which provides an easy-to-use environment for writing and uploading **C/C++ code**. This makes the microcontroller highly accessible for developers, hobbyists, and researchers. The ability to integrate various hardware components with the *ESP32-S3*'s rich set of features ensures flexibility and scalability in the project.

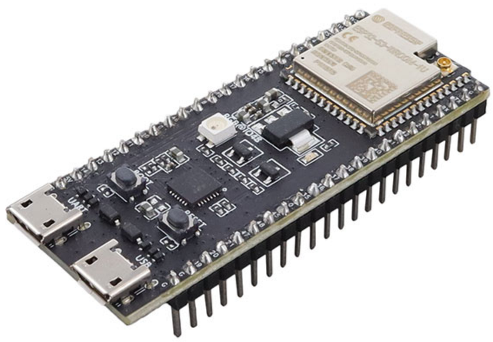

Figure 1: ESP32-S3 Microcontroller

- **Model:** ESP32-S3
- **Processor:** Dual-core 32-bit CPU, capable of up to 240 MHz clock speed
- **Connectivity:**
  - Wi-Fi (802.11 b/g/n) for network communication
  - Bluetooth 5.0 (BLE) for low-energy wireless communication

  - Two USB-C ports, one for USB-to-serial and one for USB OTG connection

- **Memory:** 512MB RAM (available depending on the board variant)
- **Storage:** Flash memory for firmware and program storage (up to 16MB)
- **I/O Pins:** 34 programmable GPIO pins supporting digital I/O, analog inputs, PWM, and communication protocols
- **Programming Environment:** Arduino IDE with support for C/C++ libraries
- **Key Role in Project:**
  - Managing the communication between the NAS system and external storage
  - Handling data transfer operations between devices
  - Enabling peripheral control (e.g., LED indicators)
- **Power Efficiency:** Low power consumption, suitable for continuous operation in embedded systems
- **Additional Features:**
  - Support for a wide range of external devices via USB OTG
  - Capability to control external components like fans for cooling and temperature regulation

2.3 External Hard-Drive

For the external storage device in this project, the **Toshiba Canvio Basics 1TB USB 3.2 External Hard Drive** is being considered. This drive offers a convenient, high-capacity solution for storing backups and cold-storage files. Its compact and portable design, combined with I connectivity, ensures fast data transfers between the external drive and the *NAS* setup, while its 1TB capacity provides ample room for the project's storage needs.

Figure 2: Toshiba Canvio Basics 1TB

- **Model:** Toshiba Canvio Basics 1TB
- **Capacity:** 1TB
- **Interface:** USB 3.2 (Backward compatible with USB 3.0/2.0)
- **Form Factor:** 2.5-inch portable external hard drive
- **Power Supply:** Bus-powered via USB connection (No external power required)
- **Features:**

  - Plug-and-play functionality for easy setup
  - Fast read and write speeds with USB 3.2
  - Compact, portable design suitable for both stationary and mobile use
  - Ideal for use as backup storage in conjunction with NAS

- **Role:** External storage for backup and file management within the NAS system

NOTE: This might not end up being the actual hard-drive being used.

2.4 3D Printed Parts

The project includes custom-designed 3D printed parts intended to house and protect the microcontroller and other components. These parts are designed to provide a sturdy enclosure, facilitate heat dissipation, and support efficient cable management. While the exact shape and measurements are still to be finalized, the 3D printed enclosure will be designed to fit the *ESP32-S3 microcontroller*, a button, LEDs, and any necessary cooling components, such as a fan.

The enclosure may feature cutouts for *USB* ports, cooling fans, and *GPIO connections*, ensuring that all components remain accessible while being securely housed. Ventilation holes or slots will be incorporated for passive heat dissipation, with the option to add a mini fan for active cooling.

Once the design is finalized, the parts will be 3D printed using **PLA** or **ABS** filament for durability and ease of fabrication.

- **Role:** Protective enclosure for electronic components, including microcontroller, hard drive, and cooling system
- **Features:**
  - Sturdy design to house all essential parts securely
  - Ventilation holes for heat dissipation and optional fan integration
  - Cutouts for ports, connections, and external hardware
- **Material:** *PLA* or *ABS* filament (to be determined based on durability and heat resistance needs)
- **Customization:** Measurements and shape will be finalized during the design phase

2.5 Others

In addition to the primary components of the project (*microcontroller, NAS, external storage, and 3D printed parts*), several auxiliary components are necessary for assembly, testing, and ensuring proper functionality. These include basic prototyping materials, cables, and tools.

- **Breadboard:** A breadboard will be used for prototyping the circuit and testing connections between components before finalizing the design. It will allow easy placement and rearranging of the wires without soldering.
- **Jumper Wires:** A set of jumper wires is essential for connecting components on the breadboard, such as the *ESP32-S3 microcontroller*, buttons and cooling systems.
- **USB-C Cables:** High-quality *USB-C* cables will be needed to connect the *ESP32*-S3 to external devices (such as the hard drive) and for data transfer.
- **Cooling Fan (Optional):** Depending on the power requirements and the amount of heat generated by the components, an optional mini cooling fan could be used for active cooling, especially if the 3D printed enclosure design necessitates additional heat management.
- **Resistors, Buttons and LEDs:** Basic *resistors*, *LEDs* and *Buttons* will be required for the circuit to work.
- **Screws and Mounting Hardware:** To secure the components (such as the *ESP32-S3* and external hard drive) inside the 3D printed enclosure,screws and other mounting hardware, such as glue, might be necessary.

3 Acquisitions

The following table lists the items that need to be acquired, along with the unit costs and total costs:

Item Name	**Unit Cost**	**Quantity**	**Total Cost**
Toshiba Canvio Basics 1TB	59.90€	1	59.90€
3D Printer (Creality Ender 3 V3 SE) + 3D Printing Filament	90€	1	90€
ESP32-S3 Microcontroller	6.50€	1	6.50€
2-Pin Push-On Button	0.34€	1	0.34€
Green LED	0.41€	1	0.41€
Total			157.15€

Alternatively, using on-site 3D printing services, a significantly lower price for the case printing can be achieved, at the risk of having to re-print the casing several times in case there are any issues.

Item Name	**Unit Cost**	**Quantity**	**Total Cost**
Toshiba Canvio Basics 1TB	59.90€	1	59.90€
3D Printing Services	15-80€	1	15 - 80€
ESP32-S3 Microcontroller	6.50€	1	6.50€
2-Pin Push-On Button	0.34€	1	0.34€
Green LED	0.41€	1	0.41€
Total			82.15€ - 147.15€

4 Hardware Conceptualisation

The hardware setup for this project revolves around the integration of several key components: the *ESP32-S3 microcontroller*, the *NAS system* (Lenovo ThinkPad T460), the *external storage device* (Toshiba Canvio Basics 1TB hard drive), and any additional peripherals like cooling systems and sensors. This section outlines the conceptualization of how these components will be wired and assembled for optimal performance.

At the core of the system is the *ESP32-S3 microcontroller*, which will serve as the main controller responsible for managing data transfer between the *NAS* and the external storage. The *ESP32-S3* will be connected to the Toshiba hard drive via the *USB-C OTG port*. The *OTG* (On-The-Go) functionality will allow the microcontroller to interface directly with the external storage, enabling data transfer without the need for an intermediary device.

To power the system, the *ESP32-S3* and the hard drive will both be powered by the *NAS*. The *USB-C cables* will be used to establish the necessary connections, one for the communication between the microcontroller and the hard drive, and the other for programming and debugging the *ESP32-S3*.

Initially, the circuit will be assembled on a *breadboard* for prototyping. This will allow for easy testing and adjustments before finalizing the design. The breadboard will serve as a temporary testing platform to connect the *GPIO pins* of the *ESP32-S3* to external peripherals such as *buttons* and *LEDs*. Jumper wires will be used to create the connections between the microcontroller and the peripherals, ensuring that all components are properly integrated for data transfer and system monitoring.

To manage heat dissipation, a *mini cooling fan* may be integrated into the system if temperature levels exceed acceptable thresholds. The fan will be powered by the microcontroller and will run in parallel to the circuit once it turns on. Whilst the fan is optional, a few holes on the sides of the casing to manage heat dissipation will be included.

Once the components are tested on the breadboard, they will be transferred to the final *3D printed enclosure*, which will only be designed once this testing phase is over in order to create a case big enough for the components.

5 Software Conceptualisation

The software component of the project will focus on ensuring that the *ESP32-S3* microcontroller efficiently handles the file transfer and backup process between the *NAS* and the external storage device. The microcontroller will be programmed to interact with a file specified in a configuration format, namely a *cold-backup.spec* file, which will be located at the root of the *NAS*' shared storage directory.

The *ESP32-S3* will read the contents of this specification file, which will list all the files and directories that need to be backed up to the external storage. The *microcontroller* will proceed by copying each of the specified files to the root directory of the external *hard drive*, ensuring that the backup data is updated whenever newer files are found on the *NAS*. If any file on the *NAS* has a more recent modification timestamp than the corresponding file on the external storage, it will be overwritten to ensure the external hard drive always contains the most current version of the data.

An important aspect of the software design is that the *ESP32-S3* will only perform constructive operations, meaning it will never delete or modify any data on the external hard drive beyond copying and updating existing files. This approach ensures that the external storage remains a secure backup of the *NAS* system, preventing the accidental loss of any files. Only new or updated files will be written to the external drive, preserving the integrity of previously backed-up data.

This approach makes the system reliable, with the *ESP32-S3* acting as a simple yet effective data transfer tool, ensuring that the backup process remains efficient and secure, while the *NAS system* maintains its role as the central data repository.