# Replicated Secret Self-Destruct USB

ECE 445 Design Document - Spring 2025

Project #69

Alexander Clemens, Varun Siva, Danny Metzger

Professor: Viktor Gruev

TA: Micheal Gamota

# Contents

# 1 Introduction

## 1.1 Problem

Traditional flash drives were originally invented to provide a convenient and portable way to store or transfer a large amount of data [1]. Over time, many have used flash drives to hold highly secure information, from keys, classified documents, and personal data. However, their small size and portability also make them susceptible to theft, loss, and unauthorized access. Although software encryption may solve some issues, it is not a foolproof solution. Attackers can attempt to brute force passwords or exploit software vulnerabilities. In some cases, a stolen or deleted drive can be recovered with some forensic analysis. Many solutions, such as BitLocker, VeraCrypt, and others offer strong encryption, but are heavily reliant on the integrity of the system and the strength of the password itself. Password-based security is only as strong as the credentials used; weak passwords and poor management can make encrypted drives vulnerable. When analyzing the most common passwords globally, over 90% of them can be cracked by modern tools in less than one second [2]. To increase security, more complicated passwords are often created but run the risk of being forgotten or stored in an unsecure location. Remembering or managing passwords for many flash drives can be a difficult task. Although hardware-based solutions exist, they still have limitations. These solutions often mitigate the risks associated with software encryption, but still associate a password into the hard drive. In the event of a brute force attack or tampering, most existing solutions do not provide a way to automatically erase the data. For a high security drive, a solution that solves these security issues could be useful.

## 1.2 Solution

Our solution is a custom PCB flash drive that implements a variety of security features through its hardware. Instead of using a password to encrypt the drive, an encryption key will be split among three authentication cards. This will be done using Replicated Secret Sharing, requiring at least two out of three physical authentication cards connected to unlock the device. Through custom hardware, this flash drive will ensure that decryption can only occur with the correct physical authentication, deterring software attacks.

In addition, the flash drive will contain a tamper-resistant self-destruct circuit. In the event of multiple failed authentications or physical tampering of the drive, it will automatically corrupt and erase the stored data. Although the data is encrypted, this will deter most brute force attempts. There will be a small battery as part of the device, ensuring destruction even when the drive is unplugged. With all these features, this flash drive should provide a higher level of security to the user's data without the need for a complex password.

## 1.3 Visual Aid

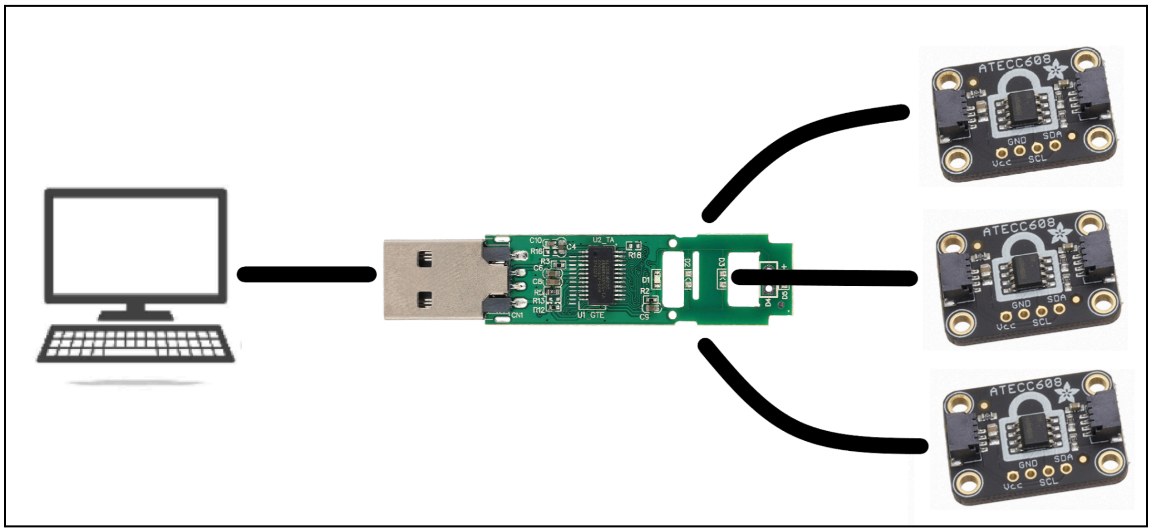

Figure 1: Visual aid for authentication cards and drive

The design will consist of two separate designs. The flash drive will be a standalone piece that functions as a normal flash drive, with extra security features. In addition, three separate authentication cards will be able to connect to the flash drive via sets of pins. Fundamentally, the flash drive will contain components essential to a normal flash drive, such as a controller, flash storage, and a USB connector.

All authentication cards may be plugged into the flash drive at once, but only two are required to encrypt and decrypt the storage. A button controls the encrypt and decrypt function. A user can simply plug in the flash drive into a computer, connect the authentication cards, and encrypt or decrypt the drive as necessary. Upon first use, the button will create a secret and split it among the cards, allowing them to be tied to that specific flash drive.

## 1.4 High-Level Requirements:

1. The flash drive must allow a maximum of 5 failed authentication attempts before triggering the self-destruct.
2. The flash drive must require at least 2 out of 3 physical authentication cards to decrypt the hidden partition.
3. The flash drive's various modes of encryption should all utilize at least 256-bit keys.

# 2 Design

## 2.1 Physical Design

For the external enclosure of our USB PCB, we will be creating a 3-D printed case as seen in the Figure 2 below. The casing will be separated into a top and bottom half that connect, with the USB PCB mounted flat on the bottom half of the enclosure. We will leave open holes on the top half of the enclosure for the USB connector, the GPIO pins that connect to the Authentication Cards, an LED to indicate different states of the USB, and a button to initiate the cryptographic initialization and authentication process. Another key component to this enclosure will be the magnet held to the inside ceiling of the top enclosure. This magnet is necessary to assist the Tamper Detection Subsystem (Section 2.3.3) discover when an attacker may be removing the USB casing. The Hall Effect Sensor on the PCB will be located below the magnet across the enclosure for guaranteed detection of its presence.

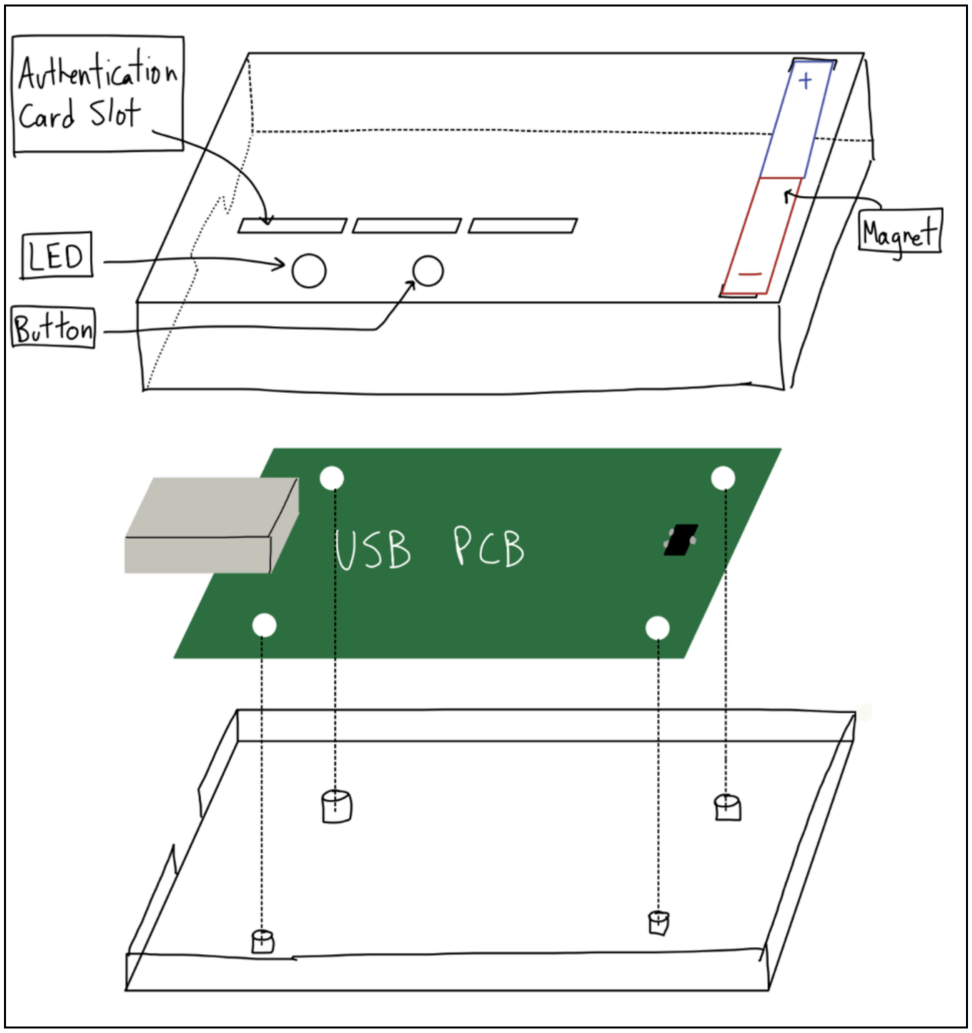

Figure 2: Physical USB Casing Design

## 2.2 Block Diagram

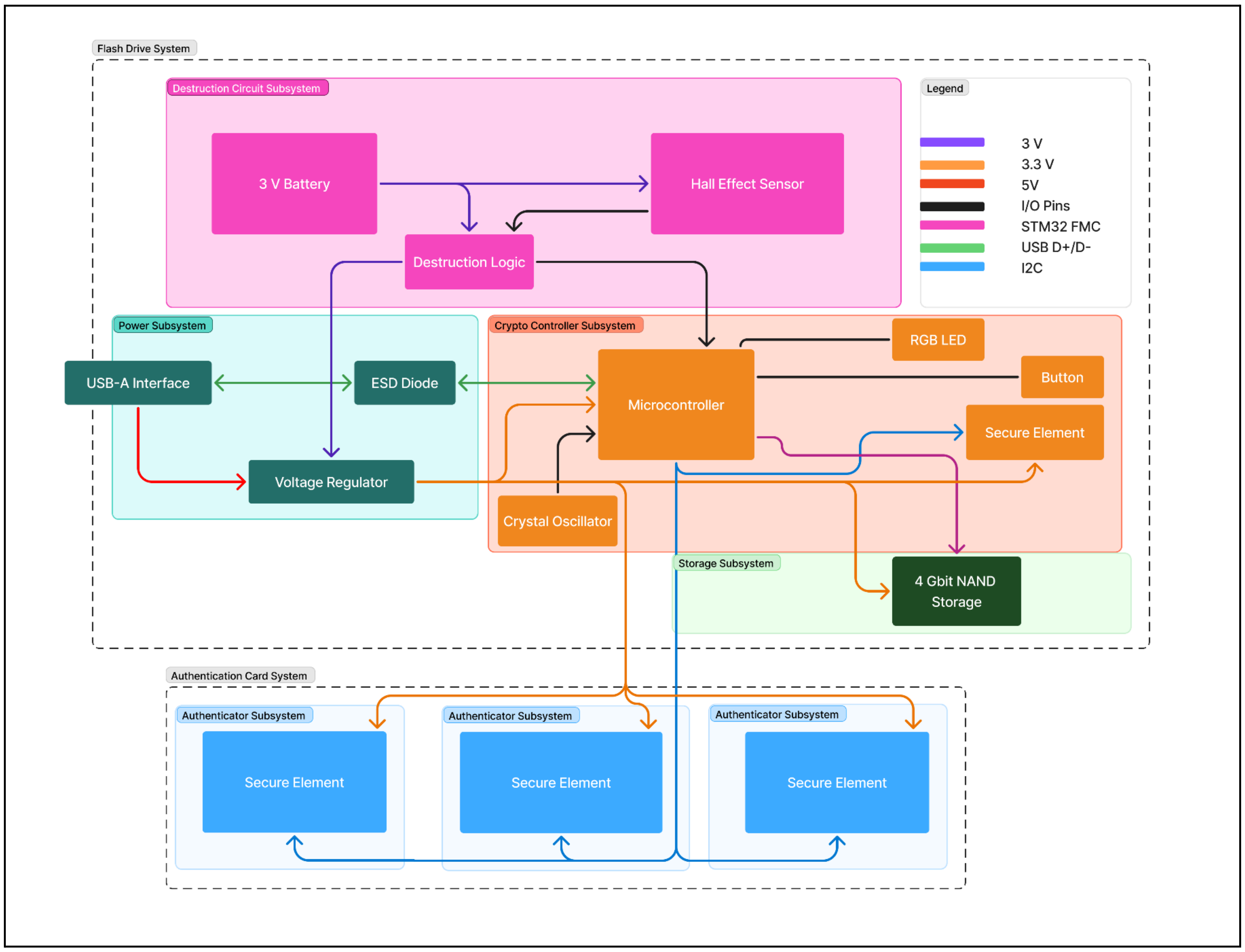

Figure 3: Replicated Secret Shared Self-Destruct USB Block Diagram

## 2.3 Subsystem Overview

### 2.3.1 Authentication Card Subsystem

The Authentication Card Subsystem allows for secure, hardware-based authentication before the USB grants access to the NAND storage data on the USB. This subsystem is located on an entirely separate PCB that connects to the main USB PCB for the authentication process. There will be 3 of these

Authentication Card PCBs in total. This subsystem communicates with the Crypto Controller Subsystem, sending or withholding the key shares that ultimately enable (or deny) access to encrypted data when 2 of them are plugged in via GPIO pins. This subsystem will mainly consist of a ATECC608B Secure Element chip placed on each Authentication Card, which will have the responsibility of holding the cryptographic key-pair that will validate its identity. When plugged into the USB via GPIO pins, the secure element will send its key-pair with I2C communication, which will travel to the microcontroller and either be confirmed or denied as the correct value. This subsystem will be driven by a 3.3V output from the voltage regulator on the main USB PCB  as well as the USB's common ground that both come through the GPIO pins. To ensure the Authentication Subsystem is fulfilling its responsibilities of sensing its connection to the USB and translating its cryptographic key-pair to the USB microcontroller, a requirements and verification table can be found below. For more information on the cryptographic capabilities of the Authentication cards, refer to section 2.4.

| REQUIREMENTS | VERIFICATION |
|---|---|
| ● All 3 authentication cards are able to be plugged into the USB via GPIO pins and initialized at the same time. | ● Before any authentication cards have been initialized with the USB, connect all 3 of the Authentication Card PCBs into the USB.<br>● Next, press the button present on the USB to initialize the cryptographic keys for the system.<br>● Measure the encrypted values of the cards once initialized for testing purposes. To do this, connect the 3 Authentication Cards again and have the USB microcontroller send the encrypted values to the NAND memory to be viewed as verified during production. |
| ● Once initialized, the K-pair held in each authentication card can not be altered or changed. | ● Insert one initialized authentication card and two uninitialized cards into the USB.<br>● Press the USB button to attempt reinitialization of |

| | |
|---|---|
| Additionally, no further authentication cards can be initialized. | cryptographic keys.<br>● Test the two newly initialized cards by inserting them into the USB and pressing the button—ensure access is denied.<br>● Measure the encrypted values of the previously initialized card after reinitialization, then insert it along with another original card. Send the encrypted values to NAND memory and confirm they remain unchanged for production verification. |
| ● When connected to the USB PCB, the Authentication Card Secure Element is automatically prompted to send its K-pair via I2C communication. | ● Insert the authentication card into the USB PCB and monitor I²C lines (SDA/SCL) with an oscilloscope or logic analyzer.<br>● Verify that the secure element powers on and responds to an I²C request from the USB PCB.<br>● Confirm that the secure element transmits the K-pair via I²C.<br>● Cross-check the received K-pair against expected values stored in the secure element. |

### 2.3.2 Storage Subsystem

The Storage Subsystem is responsible for handling all read/write/erase operations to user data. This subsystem simply consists of the MT29F4G08ABADAWP-IT NAND Flash chip in order to hold and interact with the USB data [3]. Decryption of the USB data only occurs only after the Crypto Controller Subsystem authorizes it, at which point the microcontroller can securely read and write to NAND Flash. If a tamper event is triggered or the authentication cards are denied, this subsystem will be instructed by the microcontroller to erase the data present on the NAND Flash. This data is deleted by enacting the *BLOCK ERASE* function over all the already encrypted blocks of data within the NAND Flash [4]. The storage subsystem contains at least 0.5GB of NAND flash storage. This subsystem is powered by the 3.3V drawn from the USB connection with the computer. In the case that the USB case is tampered with, it is instead powered by the 3V coin battery located in the Tamper Detection Subsystem. The microcontroller itself handles all of the encryption and decryption of data, so the Storage Subsystem simply just has to store the already encrypted data. To ensure the Storage Subsystem is fulfilling its responsibilities of storing, reading, writing, and erasing all of its encrypted data, a requirements and verification table can be found below.

| REQUIREMENTS | VERIFICATION |
|---|---|
| ● The NAND Flash correctly reads and writes data when the correct Authentication Cards are utilized. | ● Insert a correct authentication card into the USB PCB and attempt to write test data to the NAND flash. Record the operation status via serial debugging.<br>● Remove the USB and place it back in with the correct Authentication Cards. Read the stored data from and compare it to the original input to confirm accuracy.<br>● Remove the correct authentication card and attempt to write or read data again. Verify that access is denied. |
| ● The NAND Flash contains only encrypted data, nothing that would be understandable without an encryption key. | ● Write test data to the NAND flash using the correct authentication card and encryption process.<br>● Read back the stored data directly from the NAND flash using debugging tools or a memory reader.<br>● Verify that the retrieved data appears as encrypted, with no readable plaintext or identifiable patterns.<br>● Attempt to decrypt the stored data using the correct encryption key and confirm that it successfully restores the original input. |
| ● All the valid data blocks stored on the | ● Ensure the NAND flash contains valid |

| | |
|---|---|
| NAND Flash are deleted once the destruction sequence is enacted with the physical tampering or incorrect Authentication Cards. | encrypted data by writing test data using a correct authentication card.<br>● Trigger the destruction sequence by either physically tampering with the device (e.g., removing the USB casing/magnet) or using an incorrect authentication card.<br>● Attempt to read data from the NAND flash after the destruction sequence and verify that all valid data blocks have been erased.<br>● Confirm that only erased or randomized data remains by checking for unreadable or zeroed-out memory regions with the microcontroller. |

### 2.3.3 Tamper Detection Subsystem

The Tamper Detection Subsystem ensures tight protection against physical attacks to tamper with the physical USB PCB that we have created. This subsystem is located on the USB PCB and consists of a A3214ELHLT-T Hall Effect Sensor(HES) that will detect when the USB enclosure is tampered or removed and initiates the sequence to erase the data present on the NAND memory. The Hall Effect Sensor sends a signal based on the presence or absence of magnetic fields to a series of MOSFETs referred to as the Destruction Logic in the Block Diagram above [5]. The enclosure designed for the USB will have a small magnet attached on its inside ceiling, which will constantly cause the Hall Effect Sensor's output signal to be sent to the Microcontroller. Once the enclosure is tampered and removed, the magnetic field will be absent from the Hall Effect Sensor's radius and it will halt its output signal, which is fed into the MOSFET system Destruction Logic. The output of the Destruction Logic will indicate to the Microcontroller that the USB has been tampered with, initiating the data erasure sequence. This

Tamper Detection will be powered by the CR2477X-HO 3V coin battery within the subsystem, so that the Hall Effect Sensor is able to detect and communicate while the USB is unplugged, when an attack such as this is likely to occur. The Destruction Logic will switch the Microcontroller and NAND memory to be powered by this coin battery when the Hall Effect Sensor detects tampering so they are able to carry out the data erasure whilst the USB is unplugged. In order to avoid premature tampering signals by the Hall Effect Sensor before the USB enclosure and magnetic field are secured, there will be a switch to disconnect the communication between the Destruction Logic and the Microcontroller until the Cryptographic keys have been initialized. To ensure the Tamper Detection Subsystem is fulfilling its responsibilities of detecting USB enclosure removal and initiating the data erasure sequence, a requirements and verification table can be found below.

| REQUIREMENTS | VERIFICATION |
|---|---|
| ● The signals sent from the Destruction Logic are not able to interface with the Microcontroller until the Cryptographic Keys are initialized. | ● Plug in the USB without initializing the cryptographic keys and monitor the connection between the Destruction Logic and the microcontroller.<br>● Utilize a multimeter to verify that no signals from the Destruction Logic are received or processed by the microcontroller, as they are not connected yet.<br>● Initialize the cryptographic keys and check that the microcontroller now recognizes and connects to signals from the Destruction Logic after a switch has been closed. |
| ● Once the USB casing and magnet are removed, the Hall Effect Sensor stops sending its signal to the Destruction Logic. | ● During testing and before implementing the firmware to delete the USB’s data after tampering, connect an LED to the signal sent out by the Hall Effect sensor. |

| | |
|---|---|
| | ● Repeatedly put on and take off the USB enclosure with the magnet attached. Ensure that the signal lights up the LED when the case is closed, and turns the LED off once the case and magnet are removed. |
| ● The Destruction Logic sends signals to the microcontroller to initiate data deletion and connect routing of the 3V coin battery to power the microcontroller and NAND memory. | ● Ensure the system is in an idle state with the USB casing and magnet in place. Record the signal from the Destruction Logic to the microcontroller via serial debugging.<br>● Before implementing data deletion firmware, test by removing the USB casing and magnet to trigger the Hall Effect Sensor. Record the signal from the Destruction Logic to the microcontroller via serial debugging. Confirm the signal is now present.<br>● Also affirm that voltage has now been routed from the 3V coin battery to both the microcontroller and the NAND memory.<br>● Once implemented the data deletion firmware, repeat the process and see that previous files on the NAND storage have been deleted once plugged back into the computer with the correct authentication cards. |

### 2.3.4 Crypto Controller Subsystem

The Control Subsystem orchestrates operations across all other subsystems. The main component of this subsystem is the STM32U5A microcontroller, which needs to send and receive data through three main peripherals. The first of these peripherals is the USB port, which serves as both input and output for our flash drive. To facilitate communication with the USB port, the data+ and data- pins are connected to the microcontroller. An internal PHY converter translates the signals into a form readable by both the microcontroller and the computer connected to the USB port. Additionally, an external crystal oscillator is necessary to provide the stable clock on which the USB connection is established.

The next peripheral the microcontroller must communicate with is the NAND flash. It does this through an on-board Flexible Memory Controller (FMC), which supports an 8-bit wide bus and the control signals required to operate the NAND flash. This configuration enables the microcontroller to write data to, and read data from, the NAND flash. Essentially, the microcontroller takes signals from the USB interface, encrypts them, and stores them in the NAND flash. When authorized, it decrypts data from the NAND flash and sends it out through the USB port.

The last peripheral the microcontroller must interface with is the set of secure elements that hold the cryptographic keys. All secure elements will be connected via an $I^2C$ bus. Because $I^2C$ is address-based, the address of the specific secure element is sent so that only that element is written to, avoiding simultaneous writes to all secure elements. The microcontroller will initially generate the keys to be stored on these secure elements, then verify them when prompted. One secure element is located inside the Crypto Controller Subsystem itself and holds the private key, while three additional secure elements are located on authentication cards, each storing part of the public keys used to decrypt data.

The user will have five authentication submissions in total before the microcontroller wipes all memory on the NAND flash. An RGB LED displays system status—such as pending authentication, successful unlock, or warning states—and a physical button is used to indicate that the user wishes to authenticate using the connected authentication cards.

| REQUIREMENTS | VERIFICATION |
|---|---|
| The microcontroller must be able to successfully transmit data in and out of the USB port | ● Connect the device to a PC using the USB port. Transfer a 1 MB test file from the PC to the device.<br>● Read back the file stored on the device to ensure the data matches the original file by comparing checksums.<br>● Transfer that same 1MB test file back to the PC, and ensure that the file received on the PC is identical to the one originally |

| | |
|---|---|
| | sent.<br>● Monitor USB status logs (through microcontroller’s debug interface) to confirm no errors occur during read/write operations |
| The microcontroller must communicate with the NAND flash using the Flexible Memory Controller (FMC) | ● Load a test firmware onto the microcontroller that writes a recognizable pattern to a specific range of addresses in the NAND flash.<br>● Read the same range of addresses back into microcontroller memory<br>● Verify that the data read from NAND matches what was written, indicating proper FMC operation. |
| The microcontroller must communicate with the secure elements using I2C | ● Power on the system and run a diagnostic routine that attempts to detect all secure elements on the I2C bus<br>● Ensure the microcontroller successfully identifies each secure element by its unique address<br>● Write a test value to one secure element and confirm it is not written to any others<br>● Read back the value to verify that the correct secure element received it<br>● Repeat the write/read process for each secure element to ensure reliable communication across all devices. |
| The LED must display the correct status when the button is pushed. | ● Load a firmware sequence that changes the LED status when the button is pressed (e.g., from “off” to “blue,” or from “green” to “red,” depending on the |

| | |
|---|---|
| | intended system state).<br>● With the system powered, observe the LED while pressing the button:<br>● Confirm that pressing the button once switches the LED to the expected color or mode (e.g., authentication mode).<br>● Release the button and confirm that the LED returns to its previous state or moves to the next intended state as specified by the system design.<br>● Repeat the test multiple times to ensure consistent behavior and no software/hardware debounce issues |

### 2.3.5 Power Subsystem

While plugged in, the components on the flash drive, as well as the authentication devices, need to be powered. The USB power from a computer is enough to power everything on the board, but comes in as +5V. Every component on our PCB takes in about 3.3V for power. The USB connector contains a pin for +5V, which is routed to the voltage regulator. The voltage regulator takes in this input and outputs a 3.3V source at 500mA, which is connected to the secure elements, NAND flash, and microcontroller. Although one of the voltage inputs for the microcontroller can be scaled down to as little as 1.71V for power saving mode, that is not necessary as we are not running the microcontroller off a battery when plugged in.

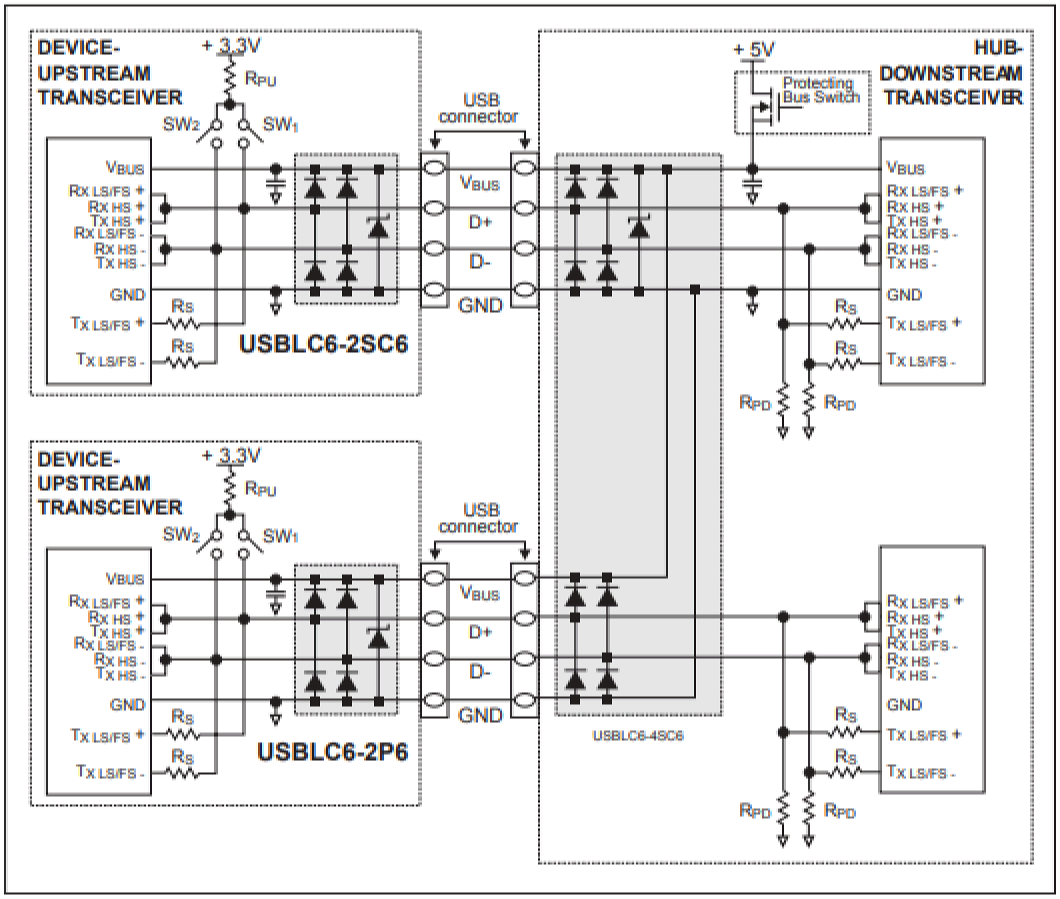

Figure 4: STM32 ESD diode circuitry example

Along with power and ground, the USB connector also contains a differential pair for data. This differential pair is routed through an ESD diode to prevent static discharge through the housing of the USB connector. This is necessary to avoid discharge from affecting the microcontroller. After the differential pair is connected to the ESD diode, it is then routed directly to the microcontroller, which has an embedded high speed PHY module to read and write data to and from the computer. High speed USB 2.0 should allow data transfer at 480 Mbps.

When unplugged, the USB connector is not able to power all the components necessary. This is necessary for tamper detection when someone attempts to open the physical casing of the flash drive. In this scenario, the 3V battery from the destruction subsystem will have to power the microcontroller and NAND flash storage for a short time while the NAND is being erased. Note that this is not intended to be done often. For more information on the design and function of the battery power, refer to sections 2.3.3 and 2.5.

| REQUIREMENTS | VERIFICATION |
|---|---|
| ● Must be able to regulate USB power to power components throughout the duration of connectivity to the computer. | ● Measure the voltage regulator output with a multimeter while plugged into a computer. Ensure it consistently provides 3.0 - 3.6V to all components under multiple load conditions.<br>● Use an oscilloscope to measure power draw upon startup and exit, ensuring all components maintain a stable 3.0 - 3.6 V before any data transactions begin. |
| ● Proper ESD protection on USB Data Lines | ● Apply a controlled electrostatic discharge to the USB housing and verify that the microcontroller is functional.<br>● Perform multiple USB plug/unplug cycles to confirm data lines remain stable.<br>● Use an oscilloscope to monitor signals before and after the ESD diode, ensuring that signals are protected. |
| ● Must be able to protect against variable changes in USB power input, as it may overvolt or draw too much current. | ● Test with a custom USB power source and introduce a small overvoltage to verify that the voltage regulator safely regulates to 3.3V.<br>● Apply a load that exceeds the current draw and verify the regulator limits current appropriately. |

## 2.4 Software and Cryptography

Suppose we have a secret key K. This key is 256-bit and will allow us to use AES-256 to encrypt and decrypt the flash storage on the drive. We want to be able to distribute this key across multiple authentication cards while ensuring that any two can reconstruct it. Although Shamir Secret Sharing is a viable solution for this, it is a bit overcomplicated for this use case. It involves reconstructing lines on a graph, and a different method may be simpler, but just as secure, for a two out of three system.

Replicated Secret Sharing stems from multi-party computation and involves K being split into three components $K_0$, $K_1$, and $K_2$ [6]:

$$K = K_0 \oplus K_1 \oplus K_2$$

Since XOR is a secure operation, it ensures that all 3 shares are needed to reconstruct K. It is good practice to also encrypt communication between the secure elements and the microcontroller on the flash drive. The flash drive will use a 256-bit public key cryptography pair to handle communication with the authentication cards' secure elements. The flash drive's Secure Element has a public key, $PK_{flash}$ and a secret key, $SK_{flash}$. Therefore, the cards will hold:

$$\text{Card 1: } Enc((K_0, K_1), PK_{flash})$$
$$\text{Card 2: } Enc((K_0, K_2), PK_{flash})$$
$$\text{Card 3: } Enc((K_1, K_2), PK_{flash})$$

The flash drive, which possesses $SK_{flash}$ is now capable of decrypting these pairs from each of the cards. No other device will be able to extract these keys, even with physical access to an authentication card. Given any two cards, it is clear that all three keys are available, allowing the flash drive to reconstruct K. This design enhances security while maintaining a level of fault tolerance against lost keys. With K reconstructed, we still need to verify that the reconstructed K is correct. The Secure Element on the flash drive holds a SHA-256 hash of K. We can then easily verify if K was reconstructed properly and determine whether to decrypt or log a failed attempt.

When the threshold is met for failed attempts, the microcontroller also contains a function to erase the entire flash storage. This can be done by calling the *BLOCK ERASE* instruction on the NAND for every block, ensuring the device is entirely reset.

## 2.5 Tolerance Analysis

While we are confident in our flash drive's ability to remain secure when plugged in, a key concern arises if the battery depletes and disables our Hall effect sensor. In that scenario, the device could fail to detect physical tampering. An attacker might open the enclosure, connect probes to the microcontroller's pins, and attempt to brute force the cryptographic keys. Although our encryption scheme aims to prevent brute force attacks, our priority is to ensure data cannot fall into adversarial hands in the first place.

To address this concern, we are planning to minimize power consumption when not plugged as much as possible. The only two components that must be plugged in in order to erase data are the hall effect sensor and the microcontroller. The Allegro A3214ELHLT-T hall effect sensor we have selected has a max current consumption of 6µA at 3V. Additionally, the STM32U5A microcontroller we have selected has a standby mode which dramatically reduces the current consumption of the microcontroller. While in this standby mode, the typical current consumption of the microcontroller at 3V is 195nA [7]. Therefore, with the hall effect sensor and microcontroller in a standby mode, the average power draw can be calculated as shown below.

$$V_{battery} * (i_{hall} + i_{microcontroller}) = 3 * (6\mu A + 195nA) = 18.585\ \mu Ah$$

So, we draw 18.585 µAh of power per hour, or 0.446 mAh per day. We have selected the CR2477X-HE battery, which has 1 Ah of power, so we will be able to keep the tamper detection circuit active for 1Ah / .446mAh = 2242 days, or approximately 6.14 years.

However, this calculation we have just performed does not account for the power that will be consumed by wiping the data from the NAND flash, so this figure is not accurate. In the event that our flash drive detects a physical tamper to its system it will need to wake up the microcontroller, power up the NAND memory, and then the microcontroller will send garbage data to the NAND memory which will wipe the data. In this case, our MT29F4 has a max current draw of 35mA, and our STM32U5A microcontroller has a max current draw of 200mA. We are assuming the worst case maximum current draw of our components to ensure that we will be able to wipe all of our data. At 3V, this means that while wiping data from the NAND we will consume the following amount of power per hour.

$$V_{battery} * (i_{microcontroller} + i_{NAND}) = 3 * (200mA + 35mA) = 705mAh$$

Dividing this into seconds, we get 0.19583 mAh per second.

Now, we must figure out how much time it will take us to write over the NAND's entire memory. The datasheet of the NAND flash lists 20 ns as a typical time to do a write cycle.[8] Since we are working with an 8-bit bus, that means that 8 bits will be delivered to the NAND every 20 ns. Therefore, we can extrapolate that the NAND flash can receive $8/(20 * 10^{-9})$ = 400,000,000 bits per second. Since we are working with 4 gigabits of data (4,294,967,296 bits), we can safely expect the NAND to be cleared in (4,294,967,296 / 400,000,000) = 10.737 seconds. Multiplying this time by the amount of power consumed by clearing the memory, we get the total power consumed by clearing the NAND. 0.19583 mAh * 10.737 seconds = 2.103 mAh of power.

Now, knowing that we will need to use 2.103 mAh of power in order to wipe all data on the NAND, we can compute how long our tamper detection functionality will be able to stay active for. With (1Ah - 2.103 mAh) = 997.897 mAh left over, we can compute our tamper detection's active time, which is 997.897 mAh / .446mAh = 2237 days, or 6.13 years.

Additionally, the Hall effect sensor only requires power to send its output signal; it will still physically switch states when the magnetic field changes, even if unpowered. This means that if the battery depletes, the sensor can still register a tamper event. The next time the device is powered via the USB interface, the control subsystem will detect the changed sensor state and trigger NAND destruction if necessary. Given the sensor's extremely low power consumption and its ability to switch states without a continuous power source, we are confident this solution provides a high level of security even under adverse conditions.

# 3 Cost and Schedule

## 3.1 Cost Analysis

The total cost for parts as seen below before shipping is $220.86. Adding in 5% of shipping cost and 10% of sales tax and our total parts cost is $253.98. We can estimate a salary of $40 * 2.5hr * 100hrs = $10,000 per team member. With 3 team members the total labor cost is $30,000. Therefore, our total cost is $30,253.98.

| Description | Manufacturer | Description | Quantity | Price | Link |
|---|---|---|---|---|---|
| STM32U5A9ZJT6Q | STMicroelectronics | Microcontroller | 4 | $16.50 | Link |
| ATECC608B-SSHDA-T | Microchip Technology | Secure Element | 15 | $0.90 | Link |
| USB1061-GF-L-A | GCT | USB Port | 4 | $0.86 | Link |
| A3214ELHLT-T | Allegro MicroSystems | Hall Effect | 5 | $1.11 | Link |
| MT29F4G08ABAFAWP-IT:F | Micron | NAND Flash | 5 | $3.56 | Link |
| LD39050PU33R | STMicroelectronics | Voltage Reg. | 5 | $1.08 | Link |
| ECS-120-20-1X-EM | ECS Inc. | Crystal | 5 | $0.91 | Link |
| CR2477X-HE | Murata Electronics | Battery | 10 | $2.78 | Link |
| USBLC6-2SC6 | STMicroelectronics | ESD diode | 6 | $0.36 | Link |
| SLW-1276864-4A-D | Same Sky | Switch | 4 | $0.80 | Link |
| KGQ05FCG1H300JH | KYROCERA AVX | 30pf Capacitor | 8 | $0.19 | Link |
| KGF21BR71H104KT | KYROCERA AVX | 0.1uf Capacitor | 50 | $0.31 | Link |
| C1608X7T1A106M080AC | TDK | 10uf Capacitor | 8 | $0.23 | Link |
| CRCW25124K70JNEGIF | Vishay / Dale | 4.7k Resistor | 30 | $0.86 | Link |

| | | | | | |
|---|---|---|---|---|---|
| IAUCN10S7N021ATMA1 | Infineon Technologies | MOSFET | 5 | $4.04 | Link |
| D6C10F1LFS | C&K | Button | 4 | $1.20 | Link |
| CLV1H-FKB-CMPRTHKBB2367453 | Cree LED | RGB LED | 5 | $0.36 | Link |
| Total | | | | $220.86 | |

## 3.2 Schedule

| **Week** | **Task** | **Person** |
|---|---|---|
| March 2 - March 6 | Work on design document | Everyone |
| | Research datasheets for microcontroller and NAND flash | Alex |
| | Update lab notebook with recent progress, create and order PCB for Authentication card | Danny |
| | PCB Layout wiring for flash drive design | Varun |
| March 6 - March 13 | Configure I2C communication on the development board for Breadboard Demo. **Order parts and second PCB** | Everyone |
| | Get STLink IDE and development environment set up | Alex |
| | Fix up PCB layout with added components and wiring issues | Danny |
| | Work on voltage routing for self-destruct circuit | Varun |
| March 13 - March 23 (Spring Break) | Begin shrinking the PCB design for 3rd round order and begin working through firmware | Everyone |
| | Update lab notebook with recent progress and plans for testing | Varun |
| March 23 - March 31 | Finalize PCB design with self-destruct logic and smaller board.<br>Finish up initial firmware for base operations of USB and authentication system.<br>**Order third round PCB** | Everyone |
| | Solder rest of Crypto Controller Subsystem and Storage Subsystem | Alex |
| | Solder Authentication Cards and Power Subsystem | Danny |
| | Solder microcontroller and components | Varun |
| March 31 - April 7 | Fix any issues with existing PCB that we have discovered through testing.<br>**Order final PCB** | Everyone |
| | Write and test USB file storage system for the NAND flash | Alex |
| | Verify Data Deletion system | Danny |

| | | |
|---|---|---|
| | Verify cryptography and encryption work as expected with correct AES modes | Varun |
| April 7 - April 14 | Finalize 3-D printed enclosures for both authentication cards and flash drive | Everyone |
| | Run through firmware and double check all states | Alex |
| | Verify operation of Tamper Detection subsystem with magnet | Danny |
| | Ensure firmware redundancy checks on PCB | Varun |
| April 14 - April 18 | Prepare multiple drives for mock demos<br>Fill out Team Contract Assessment | Everyone |
| | Create outline for Final Paper, deal with any remaining issues | Alex |
| | Ensure Secure Element keys cannot be extracted or changed | Danny |
| | Go to lab and 3-D print enclosure designs | Varun |
| April 18 - April 30 | Solve remaining issues with self-destruct system and encryption<br>**Final Demo** | Everyone |
| | Finish first draft of Final Paper | Alex |
| | Create and solder enough authentication cards for failed authentication attempts for demo | Danny |
| | Pre-initialize drives to work as expected under enclosure | Varun |
| April 30 - May 7 | Finish final presentation and final paper.<br>Fix any issues or critiques that arose during the mock presentation | Everyone |

# 4 Ethics & Safety

Our project potentially raises a few ethical and safety concerns that must be considered during the design, development, and production phases of our project. Some of these include the following.

## 4.1 Ethical Considerations

### Honesty and Transparency

The IEEE tells us that we must “be honest and realistic in stating claims or estimates based on available data" (IEEE CoE Section 5). The USB device’s self-destructive feature is powerful but risky. Users must be fully aware of its existence, how it works, and the potential consequences of triggering it. Transparency is key to building trust and ensuring users understand what they’re dealing with.

In order to ensure this, we can provide clear documentation and warning messages on the device and in user instructions about the self-destruct mechanism and any potential consequences. This ensures that users are not blindsided by accidental data loss and can make informed decisions about using the device.

### Contribution to Society

The ACM Code of Ethics says that innovations should "Contribute to society and human well-being, acknowledging that all people are stakeholders in computing" (ACM CoE Section 1.1). Our device contributes to the protection of sensitive data, which is in the public interest, especially given the increasing concerns over data breaches and privacy violations. However, as with any security device, there is a need to balance protection with usability.

We must make sure the device not only helps protect data but also takes into account user experience. By designing the self-destruct feature with careful safeguards and clear communication to the user, we can enhance both security and user confidence, improving the overall well-being of those who use it.

## 4.2 Safety Considerations

### Secure Data Destruction

According to the IEEE, we must agree "To avoid harm to others, their property, reputation, or employment by false or misleading claims, and to avoid injuring others" (IEEE CoE Section 9). In our case with our USB, we must ensure that this device safely deletes data and has safeguards to protect the data in case of accidental deletion or malicious deletion.

The self-destructing USB is intended to protect sensitive data, but if not properly designed, it could also lead to unintended data loss or harm on purpose by an attacker attempting to delete the data. While this is an inherent risk of having a super secure hard drive such as this, we could possibly necessitate the replicated secret to fail 5 times in a row to actually delete the data as a backup plan.

### Testing for Reliability

The ACM requires one to "Ensure that software and hardware are developed and tested to meet the highest quality standards, and are fit for their intended purposes" (ACM CoE Section 2.1). In our case, reliable functioning of the device is vital, as the self-destruct mechanism could malfunction, leading to either data loss without authorization or failure to protect the data when needed.

In order to prevent this we perform rigorous testing (including edge cases and stress tests) to ensure the self-destruct mechanism works as intended under various conditions for each copy of the USB. We could test the majority of the copies we make to ensure the batch of PCBs was created as intended.

## 4.3 Regulations

### Illinois Data Protection Laws

The Personal Information Protection Act (815 ILCS 530)
governs the protection of personal data in Illinois. It requires entities to implement reasonable security measures to protect sensitive data from unauthorized access or theft.

For our USB, we must ensure that the self-destruct mechanism securely erases sensitive information, such as passwords or other personally identifiable information (PII). If the data is not properly erased, or if there is a risk of recovery after destruction, this could violate state data protection regulations.

## 4.4 Other

Besides some of these ethical and safety concerns we must consider in relation to our project, it is also vital that we keep in line with the ethical concerns in relation to the operations of ECE 445. These rules we must adhere to include:

- **Honesty:** Ensure that we are honest about the development of our project and the data we collect as the semester progresses. We should not be faking or skewing data to make our project look more successful.
- **Record Keeping:** Be sure to document everything, including our failures, into our lab notebooks. Record keeping will also keep our intellectual property safe for the future.
- **Stealing/Plagiarism:** If we use the work or ideas of others, we must be sure to give credit and abide by fair use and copyright laws. Also must not try to pass off other's work as our own.
- **Other:** Treat our team members with respect, even through arguments or disagreements. Agree to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to properly credit the contributions of others.

# 5 Citations

[1] Volle, A. (2025, February 14). *USB flash drive*. Encyclopædia Britannica.
https://www.britannica.com/technology/USB-flash-drive

[2] Stouffer, C. (2023, June 27). *139 password statistics to help you stay safe*. Official Site.
https://us.norton.com/blog/privacy/password-statistics

[3] Bigelow, Stephen J., and Margaret Jones. "What Is NAND Flash Memory?: Definition from TechTarget." *Search Storage*, TechTarget, 12 May 2023,
www.techtarget.com/searchstorage/definition/NAND-flash-memory.

[4] "Erasing Data on a Flash Storage Device Is Not as Easy as It Seems." *How to Securely Erase Flash Storage*, www.atpinc.com/blog/secure-erase-ssd-data-sanitization. Accessed 6 Mar. 2025.

[5] Soltero, M. (n.d.). *What is a hall-effect sensor?*. SSZT164 Technical article | TI.com.
https://www.ti.com/document-viewer/lit/html/SSZT164

[6] Foundation, P. B. (2021, June 4). *MPC techniques series, part 1: Secret sharing*. Medium.
https://medium.com/partisia-blockchain/mpc-techniques-series-part-1-secret-sharing-d8f98324674a

[7] Datasheet - STM32U5Axxx - Ultra-low-power arm&lt; ... (n.d.).
https://www.st.com/resource/en/datasheet/stm32u5a5aj.pdf

[8] *Global leaders in semiconductors*. Micron Technology. (n.d.).
https://www.micron.com/?login=&returnUrl=%2F-%2Fmedia%2Fclient%2Fglobal%2Fdocuments%2Fproducts%2Fdata-sheet%2Fnand-flash%2F70-series%2Fm70a_4gb_3v_nand_spi.pdf%3Frev%3D14a6ca3df4e046d09f2db87b126018ed