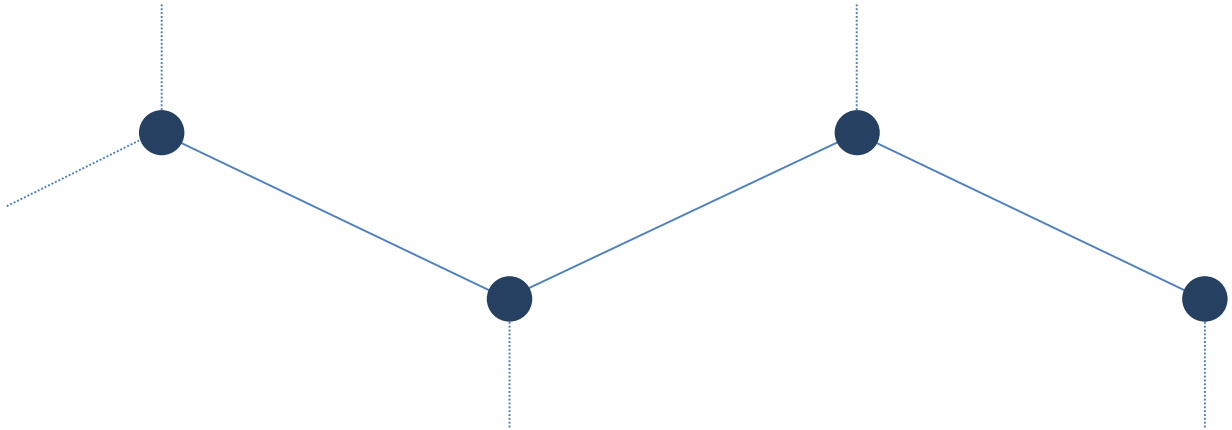
다이얼 금고

[Intel] 엣지 AI SW 아카데미(8기)
나지훈
류균봉

발표일 : 2025. 08. 08 (금)

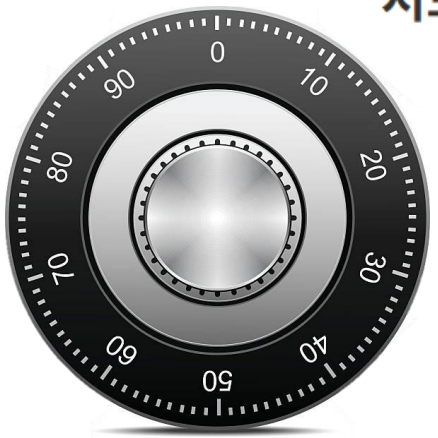
CONTENTS



프로젝트 개요

가변저항을 다이얼처럼 사용해 비밀번호를 입력하고,

서보모터로 금고를 여는 보안 시스템

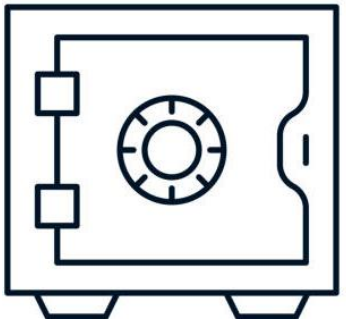


목적

🎯

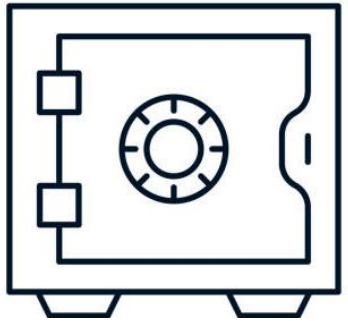
STM32 마이크로컨트롤러를 이용한 디지털 금고 시스템 구현

- STM32 기반 보안 금고 시스템 구현
- 간편한 사용자 입력 + 보안성 강화 + 인터페이스 제공



기능 요약

- 비밀번호 입력을 통한 금고 잠금/해제
- 서보모터로 물리적 금고 잠금/해제 구현
- 아날로그 다이얼(가변저항) + TACT 스위치로 숫자 입력
- LCD를 통한 실시간 상태 출력 등 사용자 인터페이스 제공
- 관리자 모드로 비밀번호 변경 기능 제공



사용 부품 및 기술

사용 부품

부품명	역할	연결 핀
가변저항 (Potentiometer)	다이얼 숫자 입력 (ADC 사용)	ADC1_IN0 (PA0)
서보모터 (SG90 등)	금고 문 개폐	TIM1_CH1 (PA8)
I2C LCD	시스템 상태 출력	I2C1 (PB7: SDA, PB6: SCL)
TACT 스위치 x2	입력 버튼 / 관리자 모드 진입	EXTI (PB5), EXTI (PB4)
STM32 보드 (예: Nucleo F401RE)	시스템 제어 (MCU)	

사용 기술

1 MCU 및 주변 장치

항목	설명
MCU	STM32F4 시리즈 (HAL 드라이버 기반)
ADC	아날로그 입력 → 0~9 숫자 변환 (가변저항)
PWM (TIM1)	서보모터 제어 (금고 잠금/해제)
TIM2	ADC 트리거용 타이머
UART2	디버깅용 시리얼 통신 (최종 코드에서는 제외)
GPIO EXTI	TACT 스위치 인터럽트 처리
I2C	LCD 제어용 통신 프로토콜

사용 기술

2️⃣ 소프트웨어 로직

🔐 상태 머신 (`enum SecurityMode`)

상태	설명	전이 조건	다음 상태
`IDLE`	대기 상태	TACT2 입력	`SUPERVISOR_INPUT`
`SUPERVISOR_INPUT`	관리자 코드 입력	일치	`CURRENT_INPUT`
`CURRENT_INPUT`	기존 비밀번호 입력	일치	`NEW_PASSWORD_INPUT`
`NEW_PASSWORD_INPUT`	새 비밀번호 입력	완료	`IDLE`

사용 기술

🛠️ 핵심 함수

함수명	기능
`handle_tact1()`	잠금/해제 처리, 비밀번호 입력 확인
`handle_tact2()`	관리자 모드 진입, 새 비밀번호 설정 3단계 보안 프로세스
`password_match()`	비밀번호 일치 여부 비교 (관리자 or 금고)
`reset_nums()`	입력 배열(insert_num[]) 초기화
`print_nums()`	LCD에 현재 입력 상태 표시
`set_password()`	금고 비밀번호 변경 처리
`HAL_ADC_ConvCpltCallback()`	ADC 값 → 숫자로 변환, LCD에 출력
`HAL_GPIO_EXTI_Callback()`	인터럽트 발생 시 TACT 버튼 처리 분기

사용 기술

🔧

📌 기타 기술 요소

- ✅ **디바운싱 처리**

  `HAL_GetTick()` 기준 200ms 이내 입력 무시

설계 포인트

주요 동작 절차

🔐 다이얼 금고 시스템 작동

👤 관리자 모드 – 비밀번호 변경 프로세스

🔐 다이얼 금고 시스템 작동

> 사용자 버튼(TACT 스위치 1) → 입력 → 검증 → 결과 처리로 이어지는 단계별 프로세스로 작동
> 모든 단계는 사용자 조작과 내부 상태에 따라 자동으로 진행

다이얼 금고 시스템 작동

1 입력 단계 – 비밀번호 입력 대기

- 사용자는 가변저항(다이얼)을 돌려 숫자 선택
- ADC 값 0 ~ 4095를 변환 값 0 ~ 9로 변환
  - 계산 방식: `입력값 410 => 1`
- 숫자 선택 후, **입력 버튼**을 눌러 저장 (인터럽트 발생)

다이얼 금고 시스템 작동

2️⃣ 비교 단계 – 비밀번호 확인

- 비밀번호 자리 수 만큼 입력되면 자동으로 비교
- 결과에 따라 아래 동작 수행:

✅ 일치 시:

- 서보모터 → 90° 회전 (열림)
- LCD 표시: `✅ UNLOCKED`

❌ 불일치 시:

- 입력 배열 초기화
- LCD 표시: `❌ FAILED`

다이얼 금고 시스템 작동

고 단

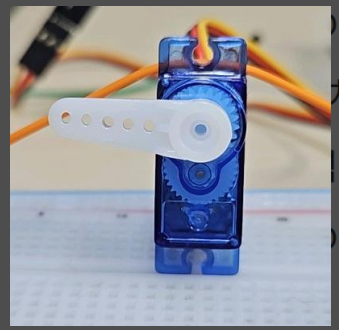
번호

에 따

시:

- 서보모터 → 90° 회전 (열림)

표시: ✅ UNLOCKED



치 시:

배열

표시

다이얼 금고 시스템 작동

3 초기화 단계 - 잠금

- 금고가 열린 상태에서 버튼을 누르면 서보모터 0°로 회전 → 금고 잠금
- LCD 표시: 🔒 `LOCKED`
- 사용자가 입력한 데이터를 초기화하여 다음 입력을 받을 준비

👤 관리자 모드 – 비밀번호 변경 프로세스

> 관리자 버튼(TACT 스위치 2)을 누르면 3단계 절차 진행
>
> 각 단계에서 통과해야 다음 단계로 진입 가능

관리자 모드

1 관리자 비밀번호 확인

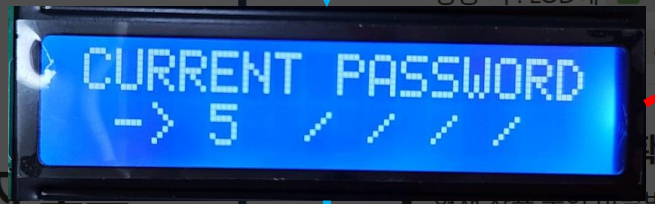
- 관리자 비밀번호 입력
- 성공 시 : LCD에 ✅ `CURRENT PASSWORD` 출력
- 실패 시 : LCD에 ❌ `ACCESS DENIED` 출력 및 금고 잠금

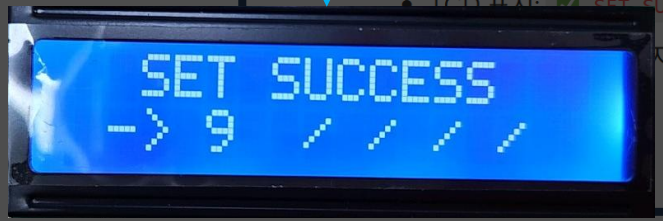
2 기존 비밀번호 확인

- 현재 사용 중인 비밀번호 입력
- 성공 시 : LCD에 ✅ `SET NEW P.W!` 출력
- 실패 시 : LCD에 ❌ `ACCESS DENIED` 출력 및 금고 잠금

3 새 비밀번호 설정

- 새로운 4자리 비밀번호 입력
- LCD 표시: ✅ `SET SUCCESS` 출력
- 완료 후 관리자 모드 자동 종료





관리자 모드

확인

성공 시 : LCD에 CURRENT PASSWORD 출력

ACCESS DENIED 출력

확인

현재 사용 중인 비밀번호 입력

성공 시 : LCD에 SET NEW P.W! 출력

ACCESS DENIED 출력 및 금고 잠금

새로운 4자리 비밀번호 입력

LCD 표시: SET SUCCESS 출력

자동 종료

주요 기능과 함수

📌

기능	담당 함수	설명 요약
비밀번호 입력 및 해제	handle_tact1()	입력 처리 및 잠금/해제 제어
관리자 모드 처리	handle_tact2()	비밀번호 변경용 FSM 처리
ADC → 숫자 변환	HAL_ADC_ConvCpltCallback	가변저항 입력값을 0~9 숫자로 변환
LCD 출력	print_nums(), reset_nums()	입력값 LCD 출력 및 초기화
비밀번호 비교/저장	password_match(), set_password()	비밀번호 확인 및 저장 처리

1️⃣ 비밀번호 입력 및 잠금/해제 제어 (`handle_tact1()`)

✅ 주요 기능 요약

- 금고가 잠긴 상태일 때 TACT1 버튼 입력:
  - ADC 값을 이용한 4자리 숫자 입력
  - 입력 완료되면 비밀번호 일치 여부 확인
  - 일치하면 모터로 잠금 해제 (`UNLOCKED`)
  - 불일치하면 LCD에 `FAILED`
- 해제 상태에서는 TACT1 눌렀을 때 다시 잠금 (`LOCKED`)

1 비밀번호 입력 및

✅ 주요 기능 요약

- 금고가 잠긴 상태일
  - ADC 값을 이용
  - 입력 완료되면 L
  - 일치하면 모터로
  - 불일치하면 LCD
- 해제 상태에서는 TA

```c
void handle_tact1()
{
    // 잠금 상태일 경우, 입력 값으로 잠금 해제 시도
    if (flag_locked)
    {
        static int i = 0;
        insert_num[i] = adc_val;
        i++;

        print_nums();  // LCD에 입력한 숫자 표시

        if (i >= PASSWORD_LEN)
        {
            if (password_match(0))  // 금고 비밀번호 검증
            {
                // 모터 90도 회전 (해제)
                __HAL_TIM_SET_COMPARE(&htim1, TIM_CHANNEL_1, 2500);

                lcd_set_cursor(0, 4);
                lcd_send_string("UNLOCKED");
                flag_locked = 0;
            }
            else
            {
                lcd_set_cursor(0, 5);
                lcd_send_string("FAILED");
            }

            reset_nums();  // 입력 초기화
            i = 0;
        }
    }
    else  // 해제 상태일 경우 -> 잠금 상태로 전환
    {
        __HAL_TIM_SET_COMPARE(&htim1, TIM_CHANNEL_1, 1500);
        lcd_set_cursor(0, 5);
        lcd_send_string("LOCKED");
        flag_locked = 1;
    }
}
```

2️⃣ 관리자 모드 – 비밀번호 변경 프로세스 (`handle_tact2()`)

✅ 주요 기능 요약

- TACT2 버튼 입력으로 진입.
- 총 3단계로 구성된 상태 기반 FSM 흐름:
  - 관리자 비밀번호 입력 (SUPERVISOR_INPUT)
  - 현재 금고 비밀번호 입력 (CURRENT_INPUT)
  - 새 비밀번호 입력 (NEW_PASSWORD_INPUT)

2 관리자 모드

✅ 주요 기능 요약

- TACT2 버튼 입
- 총 3단계로 구
  - 관리자 비
  - 현재 금고
  - 새 비밀번

```
void handle_tact2()
{
    print_nums();

    // 보안 시스템 기본 대기 상태
    if (security_mode == IDLE)
    {
        lcd_set_cursor(0, 0);
        lcd_send_string("SUPERVISOR  CODE");
        security_mode = SUPERVISOR_INPUT;
        return;
    }

    static int i = 0;
    insert_num[i] = adc_val;
    i++;

    if (i >= PASSWORD_LEN)
    {
        // 관리자 비밀번호 입력
        if (security_mode == SUPERVISOR_INPUT)
            handle_supervisor_input();

        // 현재 금고 비밀번호 입력
        else if (security_mode == CURRENT_INPUT)
            handle_current_password_input();

        // 비밀번호 재설정
        else if (security_mode == NEW_PASSWORD_INPUT)
            handle_new_password_input();

        reset_nums();
        i = 0;
        print_nums();
    }
}
```

3 보안 상태 처리 (비밀번호 검증 흐름)

▼ 🧩 핵심 코드 – 단계별 처리

```
// 관리자 비밀번호 확인
void handle_supervisor_input()
{
    if (password_match(1))  // 일치 시
    {
        lcd_send_string("CURRENT PASSWORD");
        security_mode = CURRENT_INPUT;
    }
    else  // 실패 시
    {
        lcd_send_string("ACCESS DENIED");
        security_mode = IDLE;
        __HAL_TIM_SET_COMPARE(&htim1, TIM_CHANNEL_1, 1500);
        flag_locked = 1;
    }
}
```

3 보안 상태 처리 (비밀번호 검증 흐름)

▼ 🧩 핵심 코드 – 단계별 처리

```
// 현재 금고 비밀번호 확인
void handle_current_password_input()
{
    if (password_match(0))
    {
        lcd_send_string("SET NEW P.W!");
        security_mode = NEW_PASSWORD_INPUT;
    }
    else
    {
        lcd_send_string("ACCESS DENIED");
        security_mode = IDLE;
        __HAL_TIM_SET_COMPARE(&htim1, TIM_CHANNEL_1, 1500);
        flag_locked = 1;
    }
}
```

3 보안 상태 처리 (비밀번호 검증 흐름)

▼ 🧩 핵심 코드 – 단계별 처리

```
// 새 비밀번호 설정
void handle_new_password_input()
{
    set_password();  // 비밀번호 저장
    lcd_send_string("SET SUCCESS");
    security_mode = IDLE;
    __HAL_TIM_SET_COMPARE(&htim1, TIM_CHANNEL_1, 1500);
    flag_locked = 1;
}
```

4️⃣ ADC 입력 처리 → 숫자 변환 (`HAL_ADC_ConvCpltCallback()`)

✅ **주요 기능 요약**

- 가변저항을 통해 0~4095의 아날로그 값 입력
- 이를 0~9 숫자로 변환해 입력값 처리

```c
void HAL_ADC_ConvCpltCallback(ADC_HandleTypeDef* hadc)
{
    if (hadc->Instance == ADC1)
    {
        uint32_t val = HAL_ADC_GetValue(&hadc1);

        // ADC 변환 결과를 0~9로 매핑하여 입력 숫자로 설정
        if      (val < 200)  adc_val = 0;
        else if (val < 500)  adc_val = 1;
        else if (val < 1000) adc_val = 2;
        else if (val < 1500) adc_val = 3;
        else if (val < 2000) adc_val = 4;
        else if (val < 2500) adc_val = 5;
        else if (val < 3000) adc_val = 6;
        else if (val < 3500) adc_val = 7;
        else if (val < 4000) adc_val = 8;
        else                 adc_val = 9;

        // LCD 출력
        char buffer[5];
        sprintf(buffer, "%d", adc_val);
        lcd_set_cursor(1, 4);
        lcd_send_string(buffer);
    }
}
```

4 AD

✅ 주요

- 가변
- 이를

5️⃣ LCD 출력 및 입력 초기화

✅ 주요 기능 요약

- 입력한 숫자들 LCD에 출력: `print_nums()`
- 입력 리셋: `reset_nums()`

```
void print_nums()
{
    for (int i = 0; i < 4; i++)
    {
        lcd_set_cursor(1, 7 + (i * 2));
        lcd_send_data('0' + insert_num[i]);
    }
}

void reset_nums()
{
    for (int i = 0; i < PASSWORD_LEN; i++)
        insert_num[i] = -1;
}
```

6️⃣ 비밀번호 비교 및 저장

✅ 주요 기능 요약

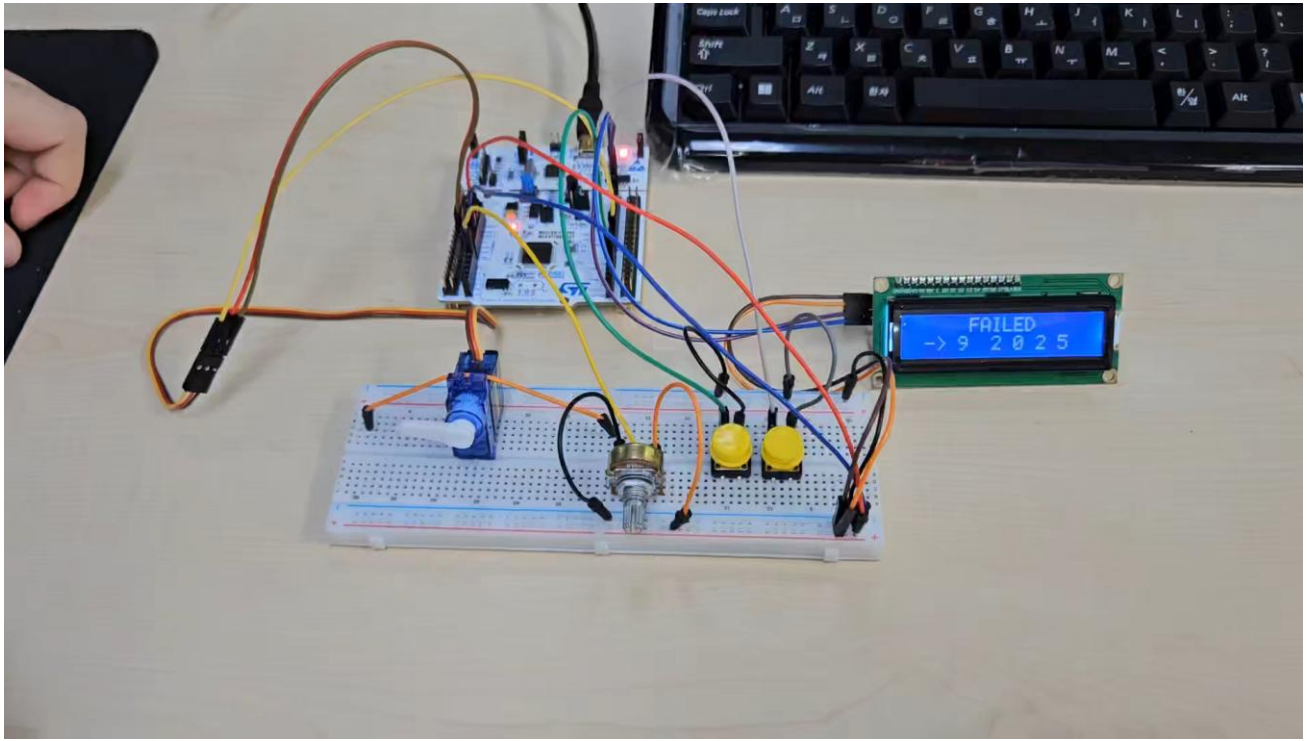
- `password_match(int security)` → 입력값과 저장된 비밀번호 일치 여부 확인
- `set_password()` → 새 비밀번호 저장

```c
int password_match(int security)
{
    for (int i = 0; i < PASSWORD_LEN; i++)
    {
        if ((security ? supervisor[i] : password[i]) != insert_num[i])
            return 0;
    }
    return 1;
}

void set_password()
{
    for (int i = 0; i < PASSWORD_LEN; i++)
        password[i] = insert_num[i];
}
```

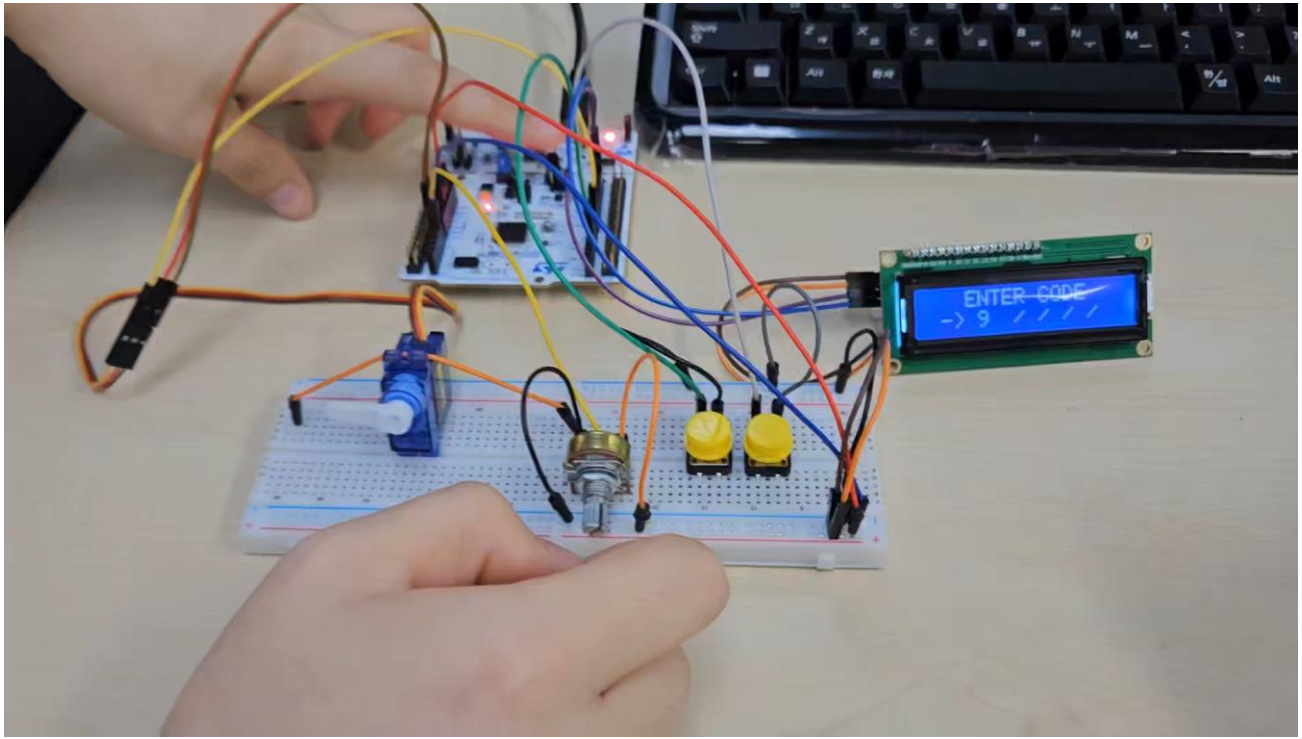
🎥 다이얼금고 전체 시연

→ 비밀번호 입력 → 일치 → 금고 열림 → 재입력 시 잠김



🎥 관리자 모드 시연

→ TACT2 → 관리자 인증 → 기존 비밀번호 → 새 비밀번호 설정



개선 아이디어

개선 아이디어

개선 항목	설명	기대 효과	우선순위
LCD 출력 개선	인터럽트 내 `HAL_Delay()` 사용 불가 문제 해결을 위해 **타이머 기반 LCD 출력 지연 처리**	사용자 경험 향상 (부드러운 UI 응답)	★★★
비밀번호 Flash 저장	비밀번호를 **STM32 내부 Flash에 저장**하여 전원 꺼져도 보존	비밀번호 유실 방지, 보안성 강화	★★★★
UPS 연동	**무정전 전원공급장치(UPS)** 연결로 전원 차단 시에도 동작 지속	전원 차단 시에도 보안 유지	★★
센서 연동 (침입 감지)	**포토센서, 볼 스위치 등** 추가해 물리적 침입 감지 및 대응	이상 접근 탐지, 보안성 향상	★★
경고 시스템 추가	**부저, Wi-Fi 모듈 연동**으로 경고음 또는 원격 알림 기능 구현	실시간 침입 경고, 원격 대응 가능	★★
비밀번호 입력 시각 피드백 강화	**숫자 입력 시 LCD에 애니메이션 효과 또는 '*' 마스킹 표시**	UX 강화, 정보 노출 방지	★★
입력 제한 및 잠금 기능	연속 실패 시 일정 시간 동안 입력 차단 (**Lock-out 기능**)	무차별 대입 공격 방지	★★★

Q&A


프로젝트 노션 URL → https://buly.kr/7bHKLKZ